LAPLACIAN SMOOTHING GRADIENT DESCENT

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a class of very simple modifications of gradient descent and stochastic gradient descent. We show that when applied to a large variety of machine learning problems, ranging from softmax regression to deep neural nets, the proposed surrogates can dramatically reduce the variance and improve the generalization accuracy. The methods only involve multiplying the usual (stochastic) gradient by the inverse of a positive definitive matrix coming from the discrete Laplacian or its high order generalizations. The theory of Hamilton-Jacobi partial differential equations demonstrates that the new algorithm is almost the same as doing gradient descent on a new function which (i) has the same global minima as the original function and (ii) is "more convex". We show that optimization algorithms with these surrogates converge uniformly in the discrete Sobolev H^p_σ sense and reduce the optimality gap for convex optimization problems. We implement our algorithm into both PyTorch and Tensorflow platforms which only involves changing of a few lines of code. The code will be available on Github.

1 INTRODUCTION

Stochastic gradient descent (SGD) has been the workhorse for solving large scale machine learning problems (Bottou et al., 2018). It gives rise to a family of algorithms that make the training of DNNs practical, which is believed to somehow implicitly smooth the loss function of the DNNs (Jastrzebski et al., 2018). Many efforts have been carried out to improve the training and generalization of DNNs by directly searching for flat minima (Keskar et al., 2017; Chaudhari et al., 2017; 2016). An alternative view of SGD's magic comes from the theory of uniform stability (Bousquet & Elisseeff, 2002; Duchi et al., 2011; Hardt et al., 2016; Bottou et al., 2016; Gonen & Shalev-Shwartz, 2017).

The noise in SGD, on the one hand, helps gradient based optimization algorithms circumvent spurious local minima and reach those that generalize well (Schmidhuber, 2014). On the other hand, it slows down the convergence of regular gradient descent (GD). To recover the linear convergence rate for strongly convex functions, several interesting variance reduction algorithms have been proposed, e.g., SAGA (Defazio & Bach, 2014) and SVRG (Johoson & Zhang, 2013). But none of them is suitable to train deep neural nets (DNNs). Improving the performance of SGD remains of interest, especially in nonconvex optimization.

In this work, we propose a carefully designed positive definite matrix to smooth the (stochastic) gradient on-the-fly. The resulting surrogate tends to reduce noise in SGD and improve the training of DNNs. We shall refer to this procedure as Laplacian smoothing. The gradient smoothing can be done by solving a tri-diagonal linear system with the original gradient on the right hand side. More precisely, we simply pre-multiply the gradient by the inverse of the following tri-diagonal circular convolution matrix

$$\boldsymbol{A}_\sigma := \begin{bmatrix} 1+2\sigma & -\sigma & 0 & \dots & 0 & -\sigma \\ -\sigma & 1+2\sigma & -\sigma & \dots & 0 & 0 \\ 0 & -\sigma & 1+2\sigma & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ -\sigma & 0 & 0 & \dots & -\sigma & 1+2\sigma \end{bmatrix} \tag{1}$$

for some positive constant $\sigma \geq 0$. In fact, we can write $\boldsymbol{A}_\sigma = \boldsymbol{I} - \sigma \boldsymbol{L}$, where $\boldsymbol{I}$ is the identity matrix, and $\boldsymbol{L}$ is the discrete one-dimensional Laplacian which acts on indices. Define the so-called

(periodic) forward finite difference matrix as

$$\boldsymbol{D}_+ = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ 0 & 0 & -1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 0 & 0 & \dots & 0 & -1 \end{bmatrix}.$$

We can write $\boldsymbol{A}_\sigma = \boldsymbol{I} - \sigma \boldsymbol{D}_- \boldsymbol{D}_+$, where $\boldsymbol{D}_- = -\boldsymbol{D}_+^\top$ is the backward finite difference. The resulting Laplacian smoothing stochastic gradient descent (LS-SGD) requires negligible extra computational cost and generalizes better than the standard SGD. When the Hessian has a poor condition number, gradient descent performs poorly. In this case, the derivative increases rapidly in one direction, while increasing slowly in others. Gradient smoothing can avoid jitter along steep directions and help make progress in shallow directions (Li & et al, 2018). Moreover, we show that the operator $\boldsymbol{A}_\sigma^{-1}$ acts like a denoiser which enables better convergence in the presence of a very noisy stochastic gradient. Our proposed approach is linked to an unusual HJ-PDE whose solution makes the original loss function more convex while retaining its flat (and global) minima, and we essentially work on this surrogate function with a much better landscape.

2 Hamilton-Jacobi PDEs and Convexification

Machine learning problems are generally formulated as finding the optimal parameters $\boldsymbol{w}$ of a parametric function $\boldsymbol{y} = h(\boldsymbol{x}, \boldsymbol{w})$, such that for an input $\boldsymbol{x}$, the output $\boldsymbol{y}$ is as accurate as possible. The optimal $\boldsymbol{w}$ can be obtained by minimizing an empirical risk function, $f(X, Y, \boldsymbol{w}) \doteq f(\boldsymbol{w})$, given the training data $\{X, Y\}$. We consider the problem of finding a global or a relatively flat minima of $f(\boldsymbol{w})$. To this end, we introduce the following unusual HJ-PDE with $f(\boldsymbol{w})$ as initial condition

$$\begin{cases} u_t + \frac{1}{2}\langle \nabla_{\boldsymbol{w}} u, \boldsymbol{A}_\sigma^{-1} \nabla_{\boldsymbol{w}} u \rangle = 0, & (\boldsymbol{w}, t) \in \Omega \times [0, \infty) \\ u(\boldsymbol{w}, 0) = f(\boldsymbol{w}), & \boldsymbol{w} \in \Omega \end{cases} \tag{2}$$

By the Hopf-Lax formula (Evans, 2010), the unique viscosity solution to the above problem can be represented by

$$u(\boldsymbol{w}, t) = \inf_{\boldsymbol{v}} \Big\{ f(\boldsymbol{v}) + \frac{1}{2t} \langle \boldsymbol{v} - \boldsymbol{w}, \boldsymbol{A}_\sigma (\boldsymbol{v} - \boldsymbol{w}) \rangle \Big\}.$$

This viscosity solution $u(\boldsymbol{w}, t)$ makes $f(\boldsymbol{w})$ more convex by bringing down the local maxima while retaining the wide minima. An illustration of this is shown in Fig. 1. If we perform the smoothing GD with proper step size on the function $u(\boldsymbol{w}, t)$, it is easier to reach the global or at least a flat minima of the original nonconvex function $f(\boldsymbol{w})$.

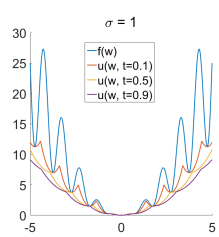


Figure 1: $f(\boldsymbol{w}) = \|\boldsymbol{w}\|^2 \big(1 + \frac{1}{2}\sin(2\pi\|\boldsymbol{w}\|)\big)$ is made more convex by solving Eq.(2). The plot shows the cross section of the 5D problem with $\sigma = 1$ and different t values.

Proposition 1. *Suppose $f(\boldsymbol{w})$ is differentiable, the Laplacian smoothing GD update on $u(\boldsymbol{w}, t)$*

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - t \boldsymbol{A}_\sigma^{-1} \nabla_{\boldsymbol{w}} u(\boldsymbol{w}^k, t)$$

permits the smoothing implicit gradient descent on $f(\boldsymbol{w})$

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - t \boldsymbol{A}_\sigma^{-1} \nabla f(\boldsymbol{w}^{k+1}). \tag{3}$$

All the proofs here and below are provided in the appendix.

2.1 LAPLACIAN SMOOTHING GRADIENT DESCENT

Laplacian smoothing implicit gradient descent requires inner iterations as used in (Chaudhari et al., 2017), which is computationally expensive. We relax the implicit scheme to the explicit

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \gamma_k \boldsymbol{A}_\sigma^{-1} \nabla f(\boldsymbol{w}^k).$$

Intuitively, compared to the standard GD, this scheme smooths the gradient on-the-fly by an elliptic smoothing operator. Here we adopt an efficient fast Fourier transform (FFT) implementation, which is available in both PyTorch (Paszke et al., 2017) and TensorFlow (Abadi et al., 2016). Given a vector $\boldsymbol{g}$, a smoothed vector $\boldsymbol{d}$ can be obtained by computing $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$. This is equivalent to $\boldsymbol{g} = \boldsymbol{d} - \sigma \boldsymbol{v} * \boldsymbol{d}$, where $\boldsymbol{v} = [-2, 1, 0, \cdots, 0, 1]^\top$ and $*$ is the convolutional operator. Hence, we have

$$\boldsymbol{d} = \mathrm{ifft}\left(\frac{\mathrm{fft}(\boldsymbol{g})}{\boldsymbol{1} - \sigma \cdot \mathrm{fft}(\boldsymbol{v})}\right),$$

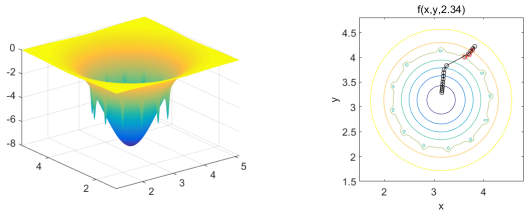
where we use component-wise division, fft and ifft are the FFT and inverse FFT, respectively. Therefore, the smoothed gradient can be obtained in quasilinear time. This additional time complexity is almost the same as performing a one step update on the weights vector $\mathbf{w}$. For many machine learning models, we may need to concatenate the parameters into a vector. This reshape might lead to some ambiguity, nevertheless, based on our tests, both row and column majored reshaping work for the LS-GD algorithm. Moreover, in deep learning cases, the weights in different layers might have different physical meanings. We then perform layer-wise gradient smoothing, instead.

Remark 1. *In image processing, the Sobolev gradient (Jung et al., 2009) involves a multi-dimensional Laplacian operator which operates on $\boldsymbol{w}$, is different from the one-dimensional discrete Laplacian operator employed in our LS-GD scheme, which operates on indices.*

To verify the improved convexity of the Laplacian smoothing gradient descent (LSGD), we show that LS-GD helps bypass sharp minima and reach the global minima for certain examples. We consider the following function, in which we 'drill' narrow holes on a smooth convex function,

$$f(x,y,z) = -4e^{-\left((x-\pi)^2+(y-\pi)^2+(z-\pi)^2\right)} - 4\sum_i \cos(x)\cos(y) e^{-\beta\left((x-r\sin(\frac{i}{2})-\pi)^2+(y-r\cos(\frac{i}{2})-\pi)^2\right)}, \quad (4)$$

where the summation is taken over the index set $\{i \in \mathbb{N} |\ 0 \le i < 4\pi\}$, r and β are the parameters that determine the location and narrowness of the local minima and are set to 1 and $\frac{1}{\sqrt{500}}$, respectively. We do GD and LS-GD starting from a random point in the neighborhoods of the narrow minima, i.e., $(x_0, y_0, z_0) \in \{\bigcup_i U_\delta(r\sin(\frac{i}{2})+\pi, r\cos(\frac{i}{2})+\pi) |\ 0 \le i < 4\pi, i \in \mathbb{N}\}$, where $U_\delta(P)$ is a neighborhood of the point P with radius δ. Our experiments (Fig. 2) show that, if $\delta \le 0.2$, GD will converge to narrow local minima, while LS-GD convergences to wider global minima.



(a) (b)

Figure 2: Demo of GD and LS-GD. Panel (a) depicts the slice of the function (Eq.(4)) with $z = 2.34$; panel (b) shows the paths of GD (red) and LS-GD (black). We take the step size to be 0.02 for both GD and LS-GD. $\sigma = 1.0$ is utilized for LS-GD.

GENERALIZED SMOOTHING GRADIENT DESCENT

We can generalize $\boldsymbol{A}_\sigma$ to the nth order discrete hyper-diffusion operator as follows

$$\boldsymbol{I} + (-1)^n \sigma \boldsymbol{L}^n \doteq \boldsymbol{A}_\sigma^n.$$

Each row of the discrete Laplacian operator $\boldsymbol{L}$ consists of an appropriate arrangement of weights in central finite difference approximation to the 2nd order derivative. Similarly, each row of $\boldsymbol{L}^n$ is an arrangement of the weights of the central finite difference to approximate the $2n$th order derivative. It is easy to prove that the smallest eigenvalue of $\boldsymbol{A}_\sigma^n$ is 1 for all σ and n, and the leading eigenvalue is no more than $1 + 4^n\sigma$.

Remark 2. *The nth order smoothing operator $\boldsymbol{I} + (-1)^n \sigma \boldsymbol{L}^n$ can only be applied to the problem with dimension at least $2n+1$. Otherwise, we need to add dummy variables to the object function.*

Again, we apply the FFT to compute the smoothed gradient vector. For a given gradient vector $\boldsymbol{g}$, the smoothed surrogate, $(\boldsymbol{A}_\sigma^n)^{-1}\boldsymbol{g} \doteq \boldsymbol{d}$, can be obtained by solving $\boldsymbol{g} = \boldsymbol{d} + (-1)^n \sigma \boldsymbol{v}_n * \boldsymbol{d}$, where $\boldsymbol{v}_n = (c_{n+1}^n, c_{n+2}^n, \cdots, c_{2n+1}^n, 0, \cdots, 0, c_1^n, c_2^n, \cdots, c_{n-1}^n, c_n^n)$ is a vector of the same dimension as the gradient to be smoothed. And the coefficient vector $\boldsymbol{c}^n = (c_1^n, c_2^n, \cdots, c_{2n+1}^n)$ can be obtained recursively by the following formula

$$\boldsymbol{c}^1 = (1, -2, 1), \quad c_i^n = \begin{cases} 1 & i = 1, 2n+1 \\ -2c_1^{n-1} + c_2^{n-1} & i = 2, 2n \\ c_{i-1}^{n-1} - 2c_i^{n-1} + c_{i+1}^{n-1} & \text{otherwise.} \end{cases}$$

Remark 3. *The computational complexities for different order smoothing schemes are the same when the FFT is utilized for computing the surrogate gradient.*

3 Reduce Optimality Gap in SGD

We show advantages of the LS-(S)GD and generalized schemes for convex optimization. Consider finding the minima $\boldsymbol{x}^*$ of the quadratic function $f(\boldsymbol{x})$ defined in Eq.(5) by different schemes.

$$f(x_1, x_2, \cdots, x_{100}) = \sum_{i=1}^{50} x_{2i-1}^2 + \sum_{i=1}^{50} \frac{x_{2i}^2}{10^2}. \tag{5}$$

To simulate SGD, we add Gaussian noise to the gradient vector, i.e., at a given point $\boldsymbol{x}$, we have

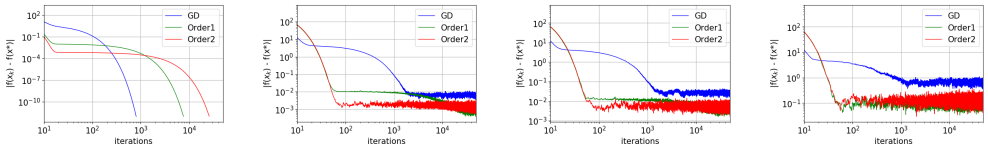
$$\tilde{\nabla}_\epsilon f(\boldsymbol{x}) := \nabla f(\boldsymbol{x}) + \epsilon \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}),$$

where the scalar ϵ controls the noise level, $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ is the vector with zero mean and unit variance in each coordinate. The corresponding numerical schemes can be formulated as

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \eta_k (\boldsymbol{A}_\sigma^n)^{-1} \tilde{\nabla}_\epsilon f(\boldsymbol{x}^k), \tag{6}$$

where σ is the smoothing parameter selected to be 10.0. We take diminishing step sizes with initial values 0.1 for SGD/smoothed SGD; 0.9 and 1.8 for GD/smoothed GD, respectively. Without noise, the smoothing allows us to take larger step sizes, rounding to the first digit, 0.9 and 1.9 are the largest suitable step size for GD and smoothed version here. We compare constant learning rate and exponentially decaying learning rate, i.e., after every 1000 iteration, the learning rate divided is by 10. We apply different schemes that corresponding to $n = 0, 1, 2$ in Eq.(6) to the problem Eq.(5), with the initial point $\boldsymbol{x}^0 = (1, 1, \cdots, 1)$.

Figure. 3 shows the iteration v.s. optimality gap when the constant learning rate is applied to different noise levels. In the noise free case, all three schemes converge linearly, but gradient smoothing has a smaller decay constant due to its increased condition number. When there is noise, our smoothed gradient helps to reduce the optimality gap and converges faster after a few iterations.



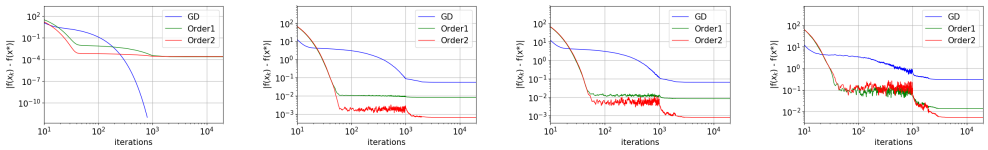
(a) $\epsilon = 0$ (b) $\epsilon = 0.05$ (c) $\epsilon = 0.1$ (d) $\epsilon = 0.5$

Figure 3: Iterations v.s. optimality gap for GD and smoothed GD with order 1 and 2 for the problem in Eq.(5). Constant step size was used.

The exponentially decaying learning rate helps smoothed SGD to reach a point with a smaller optimality gap, and the higher order smoothing further reduce the optimality gap, as shown in Fig. 4. One simple reason for this in the noisy case is because of the noise removal properties of the smoothing operators. The influence of the learning rate is still under investigation. We establish the convergence of our proposed smoothing gradient descent algorithms.

3.1 Some Properties of Laplacian Smoothing Gradient Descent

We say the objective function f has L-Lipschitz gradient, if for any $\boldsymbol{w}, \boldsymbol{u} \in \mathbb{R}^m$, we have $\|\nabla f(\boldsymbol{w}) - \nabla f(\boldsymbol{u})\| \leq L\|\boldsymbol{w} - \boldsymbol{u}\|$, and f is a-strongly convex, if $\langle \nabla f(\boldsymbol{w}) - \nabla f(\boldsymbol{u}), \boldsymbol{w} - \boldsymbol{u} \rangle \geq a\|\boldsymbol{w} - \boldsymbol{u}\|^2$. We define the vector norm induced by any matrix $\boldsymbol{A}$ as $\|\boldsymbol{w}\|_{\boldsymbol{A}} := \sqrt{\langle \boldsymbol{w}, \boldsymbol{A}\boldsymbol{w} \rangle}$.



(a) $\epsilon = 0$ (b) $\epsilon = 0.05$ (c) $\epsilon = 0.1$ (d) $\epsilon = 0.5$

Figure 4: Iterations v.s. optimality gap for GD and smoothed GD with order 1 and 2 for the problem in Eq.(5). Exponentially decaying step size is utilized here.

Proposition 2. *Suppose f is convex with the global minimizer $\boldsymbol{w}^*$, and $f^* = f(\boldsymbol{w}^*)$. Consider the following iteration with constant learning rate $\eta > 0$*

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \eta(\boldsymbol{A}_\sigma^n)^{-1}\boldsymbol{g}^k,$$

where $\boldsymbol{g}^k$ is the sampled gradient in the kth iteration at $\boldsymbol{w}^k$ satisfying $\mathbb{E}[\boldsymbol{g}^k] = \nabla f(\boldsymbol{w}^k)$. Denote $G_{\boldsymbol{A}_\sigma^n} := \sup_k \|\boldsymbol{g}^k\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}}$ and $\overline{\boldsymbol{w}}^K := \sum_{k=0}^{K-1} \boldsymbol{w}^k/K$ the ergodic average of iterates. Then the optimality gap is

$$\lim_{K\to\infty} \mathbb{E}[f(\overline{\boldsymbol{w}}^K)] - f^* \leq \frac{\eta G_{\boldsymbol{A}_\sigma^n}}{2}.$$

Note that $\|\boldsymbol{g}\|_{(\boldsymbol{A}_\sigma^n)^{-1}}$ generally decreases in n unless $\boldsymbol{g}$ is constant, which indicates that a bigger n implies smaller optimality gap. This is consistent with the experimental results above.

Proposition 3. *Suppose f is L-Lipschitz smooth and a-strongly convex with the global minimizer $\boldsymbol{w}^*$. Consider the generalized smoothing gradient descent algorithm*

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \eta_k(\boldsymbol{A}_\sigma^n)^{-1}\boldsymbol{g}^k,$$

where $\boldsymbol{g}^k$ is the sampled gradient in the kth iteration at $\boldsymbol{w}^k$ satisfying $\mathbb{E}\left[\boldsymbol{g}^k\right] = \nabla f(\boldsymbol{w}^k)$ and $\mathbb{E}\left[\|\boldsymbol{g}^k\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}}\right] \leq C_0 + C_1\|\nabla f(\boldsymbol{w}^k)\|^2$ for all $k \in \mathbb{N}$. If we take $\eta_k = \frac{C}{k+1}$ for some $C > 0$, then we have

$$\mathbb{E}\left[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}\right] = \mathbb{E}\left[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2 + \sigma\|\boldsymbol{D}_+^n(\boldsymbol{w}^k - \boldsymbol{w}^*)\|^2\right] = O\left(\frac{1}{k+1}\right),$$

i.e., we have H_σ^n uniform convergence in σ of $\{\boldsymbol{w}^k\}$ in expectation. The H_σ^n norm of $\boldsymbol{w}$ is defined by $\|\boldsymbol{w}\|_\sigma^n := \|w\|_{\boldsymbol{A}_\sigma^n} = \sqrt{\langle \boldsymbol{w}, \boldsymbol{A}_\sigma^n \boldsymbol{w}\rangle}$.

Proposition 4. *Consider the algorithm $\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \eta_k(\boldsymbol{A}_\sigma^n)^{-1}\nabla f(\boldsymbol{w}^k)$. Suppose f is convex and L-Lipschitz smooth. If the step size satisfies $0 < \underline{\eta} \leq \eta \leq \bar{\eta} < \frac{2}{L}$. Then $\lim_{t\to\infty} \|\nabla f(\boldsymbol{w}^k)\| \to 0$. Moreover, if the Hessian $\nabla^2 f$ of f is continuous with $\boldsymbol{w}^*$ being the global minimizer of f, and $\bar{\eta}\|\nabla^2 f\| < 1$, then $\|\boldsymbol{w}^k - \boldsymbol{w}^*\|_{\boldsymbol{A}_\sigma^n} \to 0$ as $k \to \infty$, and the convergence is linear and independent of σ.*

In what follows, we present the noise reduction properties of the proposed smoothing operator $\boldsymbol{A}_\sigma^{-1}$.

Proposition 5. *For any vector $\boldsymbol{g} \in \mathbb{R}^m$, $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, let $j_{\max} = \arg\max_i d_i$ and $j_{\min} = \arg\min_i d_i$. We have $\max_i d_i = d_{j_{\max}} \leq g_{j_{\max}} \leq \max_i g_i$ and $\min_i d_i = d_{j_{\min}} \geq g_{j_{\min}} \geq \min_i g_i$.*

Proposition 6. *The operator $\boldsymbol{A}_\sigma^{-1}$ preserves the sum of components. For any $\boldsymbol{g} \in \mathbb{R}^m$ and $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, we have $\sum_j d_j = \sum_j g_j$, or equivalently, $\mathbf{1}^\top \boldsymbol{d} = \mathbf{1}^\top \boldsymbol{g}$.*

Proposition 7. *Given any vector $\boldsymbol{g} \in \mathbb{R}^m$ and $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, then*

$$\|\boldsymbol{d}\| + \sigma\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^2}{\|\boldsymbol{d}\|} \leq \|\boldsymbol{g}\|.$$

The above inequality is strict unless $\boldsymbol{g} = \boldsymbol{d}$ is a constant vector. In particular, we have $\|\boldsymbol{d}\| \leq \|\boldsymbol{g}\|$ and $\|\boldsymbol{D}_+\boldsymbol{d}\| \leq \frac{1}{\sqrt{\sigma}}\|\boldsymbol{g}\|$.

Let $\boldsymbol{g}$ be the noise vector contained in the stochastic gradient, the above results imply that the extreme values in $\boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$ are smaller than those in $\boldsymbol{g}$ (in magnitude), and it also has a much smaller ℓ_2 norm.

Proposition 8. *For any $\boldsymbol{g} \in \mathbb{R}^m$, define $\mathrm{Var}(\boldsymbol{g}) := \frac{1}{m}\|\boldsymbol{g}\|^2 - \left(\frac{\mathbf{1}^\top \boldsymbol{g}}{m}\right)^2$ be the variance of components in $\boldsymbol{g}$. Let $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, then*

$$\mathrm{Var}(\boldsymbol{d}) \leq \mathrm{Var}(\boldsymbol{g}) - 2\sigma\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^2}{m} - \sigma^2\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^4}{m\|\boldsymbol{d}\|^2}.$$

The inequality is strict unless $\boldsymbol{g} = \boldsymbol{d}$ is a constant vector.

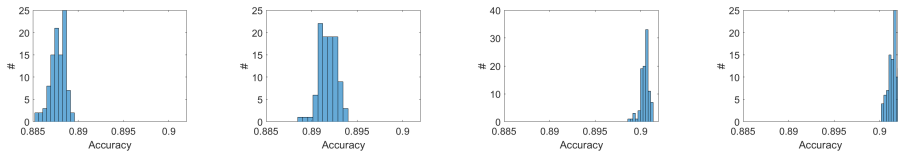
Proposition 8 shows that the component-wise variance of $\boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$ is considerably less than that of $\boldsymbol{g}$, unless $\boldsymbol{g}$ is a constant vector. Our last result shows that $\boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$ has diminishing ℓ_1 norm of finite difference of all orders. This is an excellent desnoising result.

Proposition 9. *Given vectors $\boldsymbol{g}$ and $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, for any $p \in \mathbb{N}$, it holds that $\|\boldsymbol{D}_+^p\boldsymbol{d}\|_1 \leq \|\boldsymbol{D}_+^p\boldsymbol{g}\|_1$. The inequality is strict unless $\boldsymbol{D}_+^p\boldsymbol{g}$ is a constant vector.*

Remark 4. *The above proofs generalize for $n > 1$, except for Propositions 5 and 9.*

3.2 Softmax Regression

Consider the application of the proposed optimization schemes for Softmax regression. We run 200 epochs of SGD and different order smoothing algorithms to optimize the maximum likelihood of the Softmax regression with batch size 100. Based on the results from previous section, we apply the exponentially decay learning rate with initial value 0.1 and decay 10 times after every 50 epochs. We train the model with only 10 $\%$ randomly selected MNIST training data and test the trained model on the entire testing images. We further compare the results with SVRG under the same setting. Figure. 5 shows the histograms of generalization accuracy of the Softmax regression model trained by SGD ((a)); SVRG ((b)); LS-SGD (order 1) ((c)); LS-SGD (oder 2) ((d)). It is seen that SVRG improves the generalization with higher average accuracy. But the first and second order smoothing schemes significantly improve averaged generalization accuracy by more than 1% and reduce the variance over 100 independent trials.



(a) SGD (b) SVRG (c) LS-GD: Order 1 (d) LS-GD Order 2

Figure 5: Testing accuracy of Softmax model trained on randomly selected 10% MNIST data.

4 Applications to Deep Neural Nets

4.1 Train Neural Nets with Small Batch Size

Many advanced artificial intelligence tasks make high demand on training neural nets with extremely small batch size, the milestone technique for this is group normalization (Wu & He, 2018). In this section, we show that LS-SGD successfully trains DNNs with extremely small batch size. We consider LeNet-5 devised by (LeCun et al., 1998) for MNIST classification. Our network architecture is as follows

$$\text{LeNet-5: } \mathrm{input}_{28\times 28} \to \mathrm{conv}_{20,5,2} \to \mathrm{conv}_{50,5,2} \to \mathrm{fc}_{512} \to \mathrm{softmax}.$$

The notation $\mathrm{conv}_{c,k,m}$ denotes a 2D convolutional layer with c output channels, each of which is the sum of a channel-wise convolution operation on the input using a learnable kernel of size $k \times k$, it further adds ReLU nonlinearity and max pooling with stride size m. fc_{512} is an affine transformation that transforms the input to a vector of dimension 512. Finally, the tensors are activated by a softmax function. The MNIST data is first passed to the layer $\mathrm{input}_{28\times 28}$, and further processed by this hierarchical structure. We run 100 epochs of both SGD and LS-SGD with initial learning rate 0.01 and divide by 5 after 50 epochs, and use a weight decay of 0.0001 and momentum of 0.9. Figure. 6(a) plots the generalization accuracy on the test set with the LeNet5 trained with different batch sizes. For each batch size, LS-SGD with $\sigma = 1.0$ keeps the testing accuracy more than 99.4%, SGD reduce the accuracy to 97% when batch size 4 is used. The classification become just a random guess, when the model is trained by SGD with batch size 2. Small batch size leads to large noise in the gradient, which may make the noisy gradient not along the decent direction, However, Lapacian smoothing rescues this by killing the noise.

4.2 Improve Generalization Accuracy

The skip connections in ResNet smooth the landscape of the loss function of the classical CNN (He et al., 2016; Li et al., 2017). This means that ResNet has fewer sharp minima. On Cifar10 (Krizhevsky, 2009), we compare the performance of LS-SGD and SGD on ResNet with the pre-activated ResNet56 as an illustration. We take the same training strategy as that used in (He et al., 2016), except that we run 200 epochs with the learning rate decaying by a factor of 5 after every 40 epochs. For ResNet, instead of applying LS-SGD for all epochs, we only use LS-SGD in the first 40 epochs, and the remaining training is carried out by SGD. The parameter σ is set to 1.0. Figure 6(b) depicts one path of the training and generalization accuracy of the neural nets trained by

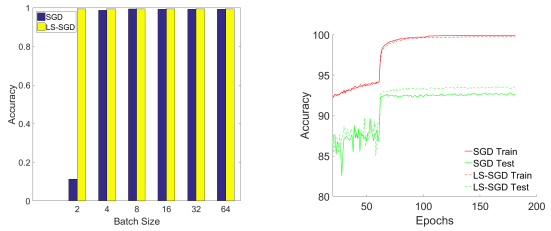


Figure 6: (a). Testing accuracy of LeNet5 trained by SGD/LS-SGD on MNIST with various batch sizes. (b). The evolution of the pre-activated ResNet56's training and generalization accuracy by SGD and LS-SGD. (Start from the 20-th epoch.)

SGD and LS-SGD, respectively. It is seen that, even though the training accuracy obtained by SGD is higher than that by LS-SGD, the generalization is however inferior to that of LS-SGD. We conjecture that this is due to the fact that SGD gets trapped into some sharp but deeper minimum, which fits better than a flat minimum but generalizes worse. We carry out 25 replicas of this experiments, the histograms of the corresponding accuracy are shown in Fig. 7.

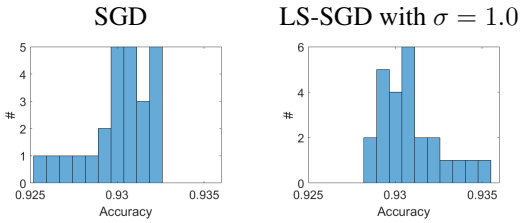


Figure 7: The histogram of the generalization accuracy of the pre-activated ResNet56 on Cifar10 trained with LS-SGD over 25 independent experiments.

4.3 Training Wassersterin GAN

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) are notoriously delicate and unstable to train (Arjovsky & Bottou, 2017). In (M. Arjovsky & Bottou, 2017), Wasserstein-GANs (WGANs) are introduced to combat the instability in the training of GANs. In addition to being more robust in training parameters and network architecture, WGANs provide a reliable estimate of the Earth Mover (EM) metric which correlates well with the quality of the generated samples. Nonetheless, WGANs training becomes unstable with a large learning rate or when used with a momentum based optimizer (M. Arjovsky & Bottou, 2017). In this section, we demonstrate that the gradient smoothing technique in this paper alleviates the instability in the training, and improves the quality of generated samples. Since WGANs with weight clipping are typically trained with RMSProp (Tieleman & Hinton, 2012), we propose replacing the gradient g by a smoothed version $g_\sigma = \boldsymbol{A}_\sigma^{-1} g$, and also update the running averages using g_σ instead of g. We name this algorithm LS-RMSProp.

To accentuate the instability in training and demonstrate the effects of gradient smoothing, we deliberately use a large learning rate for training the generator. We compare the regular RMSProp with the proposed LS-RMSProp. The learning rate for the critic is kept small and trained approximately to convergence so that the critic loss is still an effective approximation to the Wasserstein distance.To control the number of unknowns in the experiment and make a meaningful comparison using the critic loss, we use the classical RMSProp for the critic, and only apply LS-RMSProp to the generator.

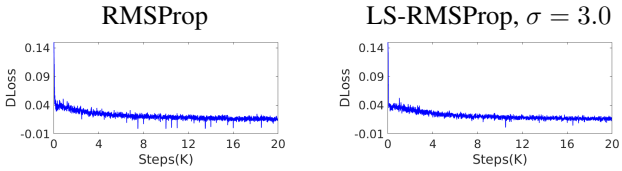
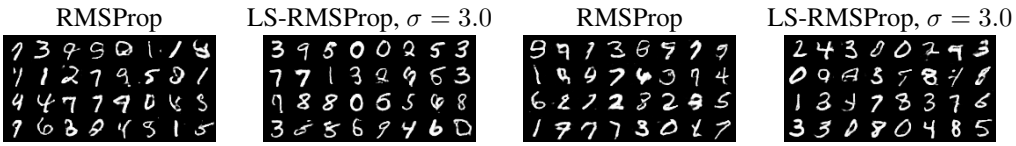


Figure 8: Critic loss with learning rate $lrD = 0.0001$, $lrG = 0.005$ for RMSProp (Left) and LS-RMSProp (Right), trained for 20K iterations. We apply a mean filter of window size 13 for better visualization. The loss from LS-RMSProp is visibly less noisy.

We train the WGANs on the MNIST dataset using the DCGAN architecture (Radford et al., 2015) for both the critic and generator. In Figure 8 (left), we observe the loss for RMSProp trained with a large learning rate has multiple sharp spikes, indicating instability in the training process. The samples generated are also lower in quality, containing noisy spots as shown in Figure 9 (a). In contrast, the curve of training loss for LS-RMSProp is smoother and exhibits fewer spikes. The generated samples as shown in Figure 9 (b) are also of better quality and visibly less noisy. A reasonable conjecture for this improved stability is that the spikes which appear in training for RMSProp correspond to sharp extrema in the approximate EM metric, and this is circumvented by smoothing the gradient. The effects are less pronounced with a small learning rate, but still result in a modest improvement in sample quality as shown in Figure 9 (c) and (d).We also apply LS-RMSProp for training the critic, but do not see a clear improvement in the quality. This may be because the critic is already trained near optimality during each iteration, and does not benefit much from gradient smoothing.



(a) (b) (c) (d)

Figure 9: Samples from WGANs trained with RMSProp (a, c) and LS-RMSProp (b, d). The learning rate is set to $lrD = 0.0001$, $lrG = 0.005$ for both RMSProp and LS-RMSProp in (a) and (b). And $lrD = 0.0001$, $lrG = 0.0001$ are used for both RMSProp and LS-RMSProp in (c) and (d). The critic is trained for 5 iterations per step of the generator, and 200 iterations per every 500 steps of the generator.

4.4 Deep Reinforcement Learning

Deep reinforcement learning (DRL) has been applied to playing games including Cartpole (Brockman et al., 2016), Atari (Mnih et al., 2013), Go (Silver & et al, 2016; Mnih & et al, 2015). DNNs play a vital role in approximating the Q-function or policy function. We apply the Laplacian smoothed gradient to train the policy function to play the Cartpole game. We apply the standard procedure to train the policy function by using the policy gradient (Brockman et al., 2016). We use the following network to approximate the policy function:

$$\text{input}_4 \to \text{fc}_{20} \to \text{relu} \to \text{fc}_2 \to \text{softmax}.$$

The network is trained by RMSProp and LS-RMSProp with $\sigma = 1.0$, respectively. The learning rate and other related parameters are set to be the default ones in PyTorch. The training is stopped once the average duration of 5 consecutive episodes is more than 490. In each training episode, we set the maximal steps to be 500. Left and right panels of Fig. 10 depict a training procedure by using RMSProp and LS-RMSProp, respectively. We see that Laplacian smoothed gradient takes fewer episodes to reach the stopping criterion. Moreover, we run the above experiments 5 times independently, and apply the trained model to play Cartpole. The game lasts more than 1000 steps for all the 5 models trained by LS-RMSProp, while only 3 of them lasts more than 1000 steps when the model is trained by vanilla RMSProp.

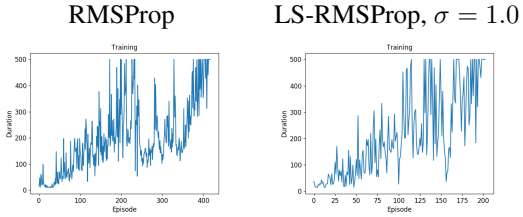


Figure 10: Durations of the cartpole game in the training procedure. Left and right are training procedure by RMSProp and LS-RMSProp with $\sigma = 1.0$, respectively.

5 Concluding Remarks

Motivated by the theory of Hamilton-Jacobi partial differential equations, we proposed Laplacian smoothing gradient descent and its high order generalizations. This simple modification dramatically reduces the optimality gap in stochastic gradient descent and helps to find better minima. Extensive numerical examples ranging from toy cases to shallow and deep neural nets to generative adversarial networks and to deep reinforcement learning, all demonstrate the advantage of the proposed smoothed gradient. Several issues remain, in particular devising an on-the-fly adaptive method for choosing the smoothing parameter σ instead of using a fixed value.

ACKNOWLEDGMENTS

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

REFERENCES

M. Abadi, A. Agarwal, and et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467, 2016.

M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. arXiv preprint arXiv:1701.04862, 2017.

D. P. Bertsekas. Nonlinear programming. Athena scientific Belmont, 1999.

L. Bottou, E. Frank, and J. Nocedal. Optimization methods for large-scale machine learning. arXiv preprint arXiv:1606.04838, 2016.

L. Bottou, E. F. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. SIAM Review, 60(2):223–311, 2018.

O. Bousquet and A. Elisseeff. Stability and generalization. Journal of Machine Learning Research, 2:499–526, 2002.

G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. arXiv preprint arXiv:1606.01540, 2016.

P. Chaudhari, A. Choromanska, S. Soatto, Y. LeCun, C. Baldassi Carlo, C. Borgs, J. Chayes, L. Sagun, and R. Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. arXiv preprint arXiv:1611.01838, 2016.

P. Chaudhari, A. Oberman, S. Osher, S. Soatto, and C. Guillame. Deep relaxation: partial differential equations for optimizing deep neural networks. arXiv preprint arXiv:1704.04932, 2017.

A. Defazio and F. Bach. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In Advances in Neural Information Processing Systems, 2014.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12:2121–2159, 2011.

L.C. Evans. Partial differential equations. 2010.

A. Gonen and S. Shalev-Shwartz. Fast rates for empirical risk minimization of strict saddle problems. arXiv preprint arXiv:1701.04271, 2017.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In Advances in Neural Information Processing Systems, pp. 2672–2680, 2014.

M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. In International Conference on Learning Representations, 2016.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778, 2016.

S. Jastrzebski, Z. Kenton, N. Ballas, A. Fischer, Y. Bengio, and A. Storkey. Dnn's sharpest directions along the sgd trajectory. arXiv preprint arXiv:1807.05031, 2018.

R. Johoson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In Advances in Neural Information Processing Systems, 2013.

M. Jung, G. Chung, G. Sundaramoorthi, L. Vese, and A. Yuille. Sobolev gradients and joint variational image segmentation, denoising, and deblurring. In Computational Imaging VII, volume 7246, pp. 72460I. International Society for Optics and Photonics, 2009.

N. Keskar, M. Shirish, N. Dheevatsa, S. Jorge, Mikhail, P. Tang, and P. Tak. On large-batch training for deep learning: Generalization gap and sharp minima. arXiv preprint arXiv:1609.04836, 2017.

A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 81:2278–2324, 1998.

F. Li and et al. Cs231n: Convolutional neural networks for visual recognition. 2018.

H. Li, Z. Xu, G. Taylor, and T. Goldstein. Visualizing the loss landscape of neural nets. arXiv preprint arXiv:1712.09913, 2017.

S. Chintala M. Arjovsky and L. Bottou. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.

Mnih and et al. Human-level control through deep reinforcement learning. Nature, 518:529–533, 2015.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing Atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602, 2013.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.

J Schmidhuber. Deep learning in neural networks: An overview. arXiv preprint arXiv:1404.7828, 2014.

D. Silver and et al. Mastering the game of go with deep neural networks and tree search. Nature, 529:484–489, 2016.

T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, 4(2):26–31, 2012.

Y. Wu and K. He. Group normalization. In European Conference on Computer Vision, 2018.

APPENDIX: PROOFS

Proposition 1. *Suppose $f(\boldsymbol{w})$ is differentiable, the Laplacian smoothing GD update on $u(\boldsymbol{w}, t)$*

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - t\boldsymbol{A}_\sigma^{-1}\nabla_{\boldsymbol{w}} u(\boldsymbol{w}^k, t)$$

permits the smoothing implicit gradient descent on $f(\boldsymbol{w})$

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - t\boldsymbol{A}_\sigma^{-1}\nabla f(\boldsymbol{w}^{k+1}).$$

Proof of Proposition 1. We define

$$z(\boldsymbol{w}, \boldsymbol{v}, t) := f(\boldsymbol{v}) + \frac{1}{2t}\langle \boldsymbol{v} - \boldsymbol{w}, \boldsymbol{A}_\sigma(\boldsymbol{v} - \boldsymbol{w})\rangle,$$

and rewrite $u(\boldsymbol{w}, t) = \inf_{\boldsymbol{v}} z(\boldsymbol{w}, \boldsymbol{v}, t)$ as $z(\boldsymbol{w}, \boldsymbol{v}(\boldsymbol{w}, t), t)$, where $\boldsymbol{v}(\boldsymbol{w}, t) = \arg\min_{\boldsymbol{v}} z(\boldsymbol{w}, \boldsymbol{v}, t)$. Then by the Euler-Lagrange equation,

$$\nabla_{\boldsymbol{w}} u(\boldsymbol{w}, t) = \nabla_{\boldsymbol{w}} z(\boldsymbol{w}, \boldsymbol{v}(\boldsymbol{w}, t), t) = \boldsymbol{J}_{\boldsymbol{w}}\boldsymbol{v}(\boldsymbol{w}, t)\nabla_{\boldsymbol{v}} z(\boldsymbol{w}, \boldsymbol{v}(\boldsymbol{w}, t), t) + \nabla_{\mathbf{w}} z(\boldsymbol{w}, \boldsymbol{v}(\boldsymbol{w}, t), t),$$

where $\boldsymbol{J}_{\boldsymbol{w}}\mathbf{v}(\boldsymbol{w}, t)$ is the Jacobian matrix of $\boldsymbol{v}$ w.r.t. $\boldsymbol{w}$. Notice that $\nabla_{\boldsymbol{v}} z(\boldsymbol{w}, \boldsymbol{v}(\boldsymbol{w}, t), t) = \mathbf{0}$,

$$\nabla_{\boldsymbol{w}} u(\boldsymbol{w}, t) = \nabla_{\boldsymbol{w}} z(\boldsymbol{w}, \boldsymbol{v}(\boldsymbol{w}, t), t) = \frac{1}{t}\boldsymbol{A}_\sigma(\boldsymbol{v}(\boldsymbol{w}, t) - \boldsymbol{w}).$$

Letting $\boldsymbol{w} = \boldsymbol{w}^k$ and $\boldsymbol{w}^{k+1} = \boldsymbol{v}(\boldsymbol{w}^k, t) = \arg\min_{\mathbf{v}} z(\boldsymbol{w}^k, \boldsymbol{v}, t)$ in the above equalities, we have

$$\nabla_{\boldsymbol{w}} u(\boldsymbol{w}^k, t) = \frac{1}{t}\boldsymbol{A}_\sigma(\boldsymbol{w}^{k+1} - \boldsymbol{w}^k).$$

In summary, the gradient descent $\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - t\boldsymbol{A}_\sigma^{-1}\nabla_{\boldsymbol{w}} u(\boldsymbol{w}^k, t)$ is equivalent to the proximal point iteration $\boldsymbol{w}^{k+1} = \arg\min_{\boldsymbol{v}} f(\boldsymbol{v}) + \frac{1}{2t}\langle \boldsymbol{v} - \boldsymbol{w}^k, \boldsymbol{A}_\sigma(\boldsymbol{v} - \boldsymbol{w}^k)\rangle$, which yields $\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - t\boldsymbol{A}_\sigma^{-1}\nabla f(\boldsymbol{w}^{k+1})$. □

Proposition 2. *Suppose f is convex with the global minimizer $\boldsymbol{w}^*$, and $f^* = f(\boldsymbol{w}^*)$. Consider the following iteration with constant learning rate $\eta > 0$*

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \eta(\boldsymbol{A}_\sigma^n)^{-1}\boldsymbol{g}^k$$

where $\boldsymbol{g}^k$ is the sampled gradient in the kth iteration at $\boldsymbol{w}^k$ satisfying $\mathbb{E}[\boldsymbol{g}^k] = \nabla f(\boldsymbol{w}^k)$. Denote $G_{\boldsymbol{A}_\sigma^n} := \sup_k \|\boldsymbol{g}^k\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}}$ and $\overline{\boldsymbol{w}}^K := \sum_{k=0}^{K-1} \boldsymbol{w}^k/K$ the ergodic average of iterates. Then the optimality gap is

$$\lim_{K\to\infty} \mathbb{E}[f(\overline{\boldsymbol{w}}^K)] - f^* \le \frac{\eta G_{\boldsymbol{A}_\sigma^n}}{2}.$$

Proof. Since f is convex, we have

$$\langle \nabla f(\boldsymbol{w}^k), \boldsymbol{w}^k - \boldsymbol{w}^*\rangle \ge f(\boldsymbol{w}^k) - f^*. \tag{7}$$

Furthermore,

$$\begin{aligned}
&\mathbb{E}[\|\boldsymbol{w}^{k+1} - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] = \mathbb{E}[\|\boldsymbol{w}^k - \eta(\boldsymbol{A}_\sigma^n)^{-1}\boldsymbol{g}^t - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] \\
&= \mathbb{E}[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] - 2\eta\mathbb{E}[\langle \boldsymbol{g}^k, \boldsymbol{w}^k - \boldsymbol{w}^*\rangle] + \eta^2\mathbb{E}[\|(\boldsymbol{A}_\sigma^n)^{-1}\boldsymbol{g}^t\|^2_{\boldsymbol{A}_\sigma^n}] \\
&\le \mathbb{E}[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] - 2\eta\mathbb{E}[\langle \nabla f(\boldsymbol{w}^k), \boldsymbol{w}^k - \boldsymbol{w}^*\rangle] + \eta^2\|\boldsymbol{g}^k\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}} \\
&\le \mathbb{E}[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] - 2\eta(\mathbb{E}[f(\boldsymbol{w}^k)] - f^*) + \eta^2 G_{\boldsymbol{A}_\sigma^n},
\end{aligned}$$

where the last inequality is due to (7). We rearrange the terms and arrive at

$$\mathbb{E}[f(\boldsymbol{w}^k)] - f^* \le \frac{1}{2\eta}(\mathbb{E}[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] - \mathbb{E}[\|\boldsymbol{w}^{k+1} - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}]) + \frac{\eta G_{\boldsymbol{A}_\sigma^n}}{2}.$$

Summing over k from 0 to $K-1$ and averaging and using the convexity of f, we have

$$\mathbb{E}[f(\overline{\boldsymbol{w}}^K)] - f^* \le \frac{\sum_{k=0}^{K-1}\mathbb{E}[f(\boldsymbol{w}^k)]}{K} - f^* \le \frac{1}{2\eta K}\mathbb{E}[\|\boldsymbol{w}^0 - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] + \frac{\eta G_{\boldsymbol{A}_\sigma^n}}{2}.$$

Taking the limit as $K \to \infty$ above establishes the result. □

Proposition 3. *Suppose f is L-Lipschitz smooth and a-strongly convex. Consider the generalized smoothing gradient descent algorithm*

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \eta_k (\boldsymbol{A}_\sigma^n)^{-1} \boldsymbol{g}^k,$$

where $\boldsymbol{g}^k$ is the sampled gradient in the kth iteration at $\boldsymbol{w}^k$ satisfying $\mathbb{E}\left[\boldsymbol{g}^k\right] = \nabla f(\boldsymbol{w}^k)$ and $\mathbb{E}\left[\|\boldsymbol{g}^k\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}}\right] \leq C_0 + C_1 \|\nabla f(\boldsymbol{w}^k)\|^2$ for all $k \in \mathbb{N}$. If we take $\eta_k = \frac{C}{k+1}$ for some $C > 0$, then we have

$$\mathbb{E}\left[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}\right] = \mathbb{E}\left[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2 + \sigma \|\boldsymbol{D}_+^n (\boldsymbol{w}^k - \boldsymbol{w}^*)\|^2\right] = O\left(\frac{1}{k+1}\right),$$

i.e., we have H_σ^n uniform convergence in σ of $\{\boldsymbol{w}^k\}$ in expectation. The H_σ^n norm of $\boldsymbol{w}$ is defined by $\|\boldsymbol{w}\|_\sigma^n := \|w\|_{\boldsymbol{A}_\sigma^n} = \sqrt{\langle \boldsymbol{w}, \boldsymbol{A}_\sigma^n \boldsymbol{w}\rangle}$.

Proof of Proposition 3. Since $\nabla f(\boldsymbol{w}^*) = \boldsymbol{0}$, by strong convexity of f, we have

$$\langle \nabla f(\boldsymbol{w}^k), \boldsymbol{w}^k - \boldsymbol{w}^* \rangle = \langle \nabla f(\boldsymbol{w}^k) - \nabla f(\boldsymbol{w}^*), \boldsymbol{w}^k - \boldsymbol{w}^* \rangle \geq a \|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2.$$

Moreover, by L-smoothness of f and the fact that $\|\boldsymbol{A}_\sigma^n\| = 1$, we also have

$$\|\nabla f(\boldsymbol{w}^k)\| = \|\nabla f(\boldsymbol{w}^k) - \nabla f(\boldsymbol{w}^*)\| \leq L \|\boldsymbol{w}^k - \boldsymbol{w}^*\| \leq L \|\boldsymbol{w}^k - \boldsymbol{w}^*\|_{\boldsymbol{A}_\sigma^n}.$$

Hence,

$$\begin{aligned}
&\mathbb{E}[\|\boldsymbol{w}^{k+1} - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] = \mathbb{E}[\|\boldsymbol{w}^k - \eta (\boldsymbol{A}_\sigma^n)^{-1} \boldsymbol{g}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] \\
=& \mathbb{E}[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] - 2\eta_k \mathbb{E}\left[\langle \boldsymbol{g}^k, \boldsymbol{w}^k - \boldsymbol{w}^* \rangle\right] + \eta_k^2 \mathbb{E}[\|\boldsymbol{g}^k\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}}] \\
=& \mathbb{E}[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}] - 2\eta_k \langle \nabla f(\boldsymbol{w}^k), \boldsymbol{w}^k - \boldsymbol{w}^* \rangle + \eta_k^2 \mathbb{E}[\|\boldsymbol{g}^k\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}}] \\
\leq& (1 - 2\eta_k a) \mathbb{E}\left[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}\right] + \eta_k^2 \left(C_0 + C_1 \mathbb{E}[\|\nabla f(\boldsymbol{w}^k)\|^2]\right) \\
\leq& \left(1 - 2\eta_k a + \eta_k^2 L^2 C_1\right) \mathbb{E}\left[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}\right] + \eta_k^2 C_0,
\end{aligned}$$

where in the first inequality we used $\|(\boldsymbol{A}_\sigma^n)^{-1}\| = 1$ for all σ and n. Taking $\eta_k = \frac{C}{k+1}$ for some proper $C > 0$ and using induction, one can show that $\mathbb{E}\left[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2_{\boldsymbol{A}_\sigma^n}\right] = \mathbb{E}\left[\|\boldsymbol{w}^k - \boldsymbol{w}^*\|^2 + \sigma \|\boldsymbol{D}_+^n (\boldsymbol{w}^k - \boldsymbol{w}^*)\|\right] = O(\frac{1}{k+1})$. □

Proposition 4. *Consider the algorithm $\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \eta_k (\boldsymbol{A}_\sigma^n)^{-1} \nabla f(\boldsymbol{w}^k)$. Suppose f is L-Lipschitz smooth and $0 < \underline{\eta} \leq \eta \leq \bar{\eta} < \frac{2}{L}$. Then $\lim_{t\to\infty} \|\nabla f(\boldsymbol{w}^k)\| \to 0$. Moreover, if the Hessian $\nabla^2 f$ of f is continuous with $\boldsymbol{w}^*$ being the minimizer of f, and $\bar{\eta}\|\nabla^2 f\| < 1$, then $\|\boldsymbol{w}^k - \boldsymbol{w}^*\|_{\boldsymbol{A}_\sigma^n} \to 0$ as $k \to \infty$, and the convergence is linear.*

Proof of Proposition 4. By the Lipschitz continuity of ∇f and the descent lemma (Bertsekas, 1999), we have

$$\begin{aligned}
f(\boldsymbol{w}^{k+1}) &= f(\boldsymbol{w}^k - \eta_k (\boldsymbol{A}_\sigma^n)^{-1} \nabla f(\boldsymbol{w}^k)) \\
&\leq f(\boldsymbol{w}^k) - \eta_k \langle \nabla f(\boldsymbol{w}^k), (\boldsymbol{A}_\sigma^n)^{-1} \nabla f(\boldsymbol{w}^k)) \rangle + \frac{\eta_k^2 L}{2} \|(\boldsymbol{A}_\sigma^n)^{-1} \nabla f(\boldsymbol{w}^k)\|^2 \\
&\leq f(\boldsymbol{w}^k) - \eta_k \|\nabla f(\boldsymbol{w}^k)\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}} + \frac{\eta_k^2 L}{2} \|\nabla f(\boldsymbol{w}^k)\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}} \\
&\leq f(\boldsymbol{w}^k) - \underline{\eta} \left(1 - \frac{\bar{\eta} L}{2}\right) \|\nabla f(\boldsymbol{w}^k)\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}}.
\end{aligned}$$

Summing the above inequality over k, we have

$$\underline{\eta} \left(1 - \frac{\bar{\eta} L}{2}\right) \sum_{k=0}^{\infty} \|\nabla f(\boldsymbol{w}^k)\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}} \leq f(\boldsymbol{w}^0) - \lim_{k\to\infty} f(\boldsymbol{w}^k) < \infty.$$

Therefore, $\|\nabla f(\boldsymbol{w}^k)\|^2_{(\boldsymbol{A}_\sigma^n)^{-1}} \to 0$, and thus $\|\nabla f(\boldsymbol{w}^k)\| \to 0$.

For the second claim, we have

$$\begin{aligned}
\boldsymbol{w}^{k+1} - \boldsymbol{w}^* &= \boldsymbol{w}^k - \boldsymbol{w}^* - \eta_k(\boldsymbol{A}_\sigma^n)^{-1}(\nabla f(\boldsymbol{w}^k) - \nabla f(\boldsymbol{w}^*)) \\
&= \boldsymbol{w}^k - \boldsymbol{w}^* - \eta_k(\boldsymbol{A}_\sigma^n)^{-1}\left(\int_0^1 \nabla^2 f(\boldsymbol{w}^* + \tau(\boldsymbol{w}^{k+1} - \boldsymbol{w}^*)) \cdot (\boldsymbol{w}^k - \boldsymbol{w}^*)\mathrm{d}\tau\right) \\
&= \boldsymbol{w}^k - \boldsymbol{w}^* - \eta_k(\boldsymbol{A}_\sigma^n)^{-1}\left(\int_0^1 \nabla^2 f(\boldsymbol{w}^* + \tau(\boldsymbol{w}^{k+1} - \boldsymbol{w}^*))\mathrm{d}\tau \cdot (\boldsymbol{w}^k - \boldsymbol{w}^*)\right) \\
&= (\boldsymbol{A}_\sigma^n)^{-\frac{1}{2}}\left(\boldsymbol{I} - \eta_k(\boldsymbol{A}_\sigma^n)^{-\frac{1}{2}}\int_0^1 \nabla^2 f(\boldsymbol{w}^* + \tau(\boldsymbol{w}^{k+1} - \boldsymbol{w}^*))\mathrm{d}\tau(\boldsymbol{A}_\sigma^n)^{-\frac{1}{2}}\right)(\boldsymbol{A}_\sigma^n)^{\frac{1}{2}}(\boldsymbol{w}^k - \boldsymbol{w}^*)
\end{aligned}$$

Therefore,

$$\|\boldsymbol{w}^{k+1} - \boldsymbol{w}^*\|_{\boldsymbol{A}_\sigma^n} \leq \left\|\boldsymbol{I} - \eta_t(\boldsymbol{A}_\sigma^n)^{-\frac{1}{2}}\int_0^1 \nabla^2 f(\boldsymbol{w}^* + \tau(\boldsymbol{w}^{k+1} - \boldsymbol{w}^*))\mathrm{d}\tau(\boldsymbol{A}_\sigma^n)^{-\frac{1}{2}}\right\| \|\boldsymbol{w}^k - \boldsymbol{w}^*\|_{\boldsymbol{A}_\sigma^n}.$$

So if $\eta_k\|\nabla^2 f\| \leq \frac{1}{\|(\boldsymbol{A}_\sigma^n)^{-1}\|} = 1$, the result follows. □

Proposition 5. *For any vector $\boldsymbol{g} \in \mathbb{R}^m$, $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, let $j_{\max} = \arg\max_i d_i$ and $j_{\min} = \arg\min_i d_i$. We have $\max_i d_i = d_{j_{\max}} \leq g_{j_{\max}} \leq \max_i g_i$ and $\min_i d_i = d_{j_{\min}} \geq g_{j_{\min}} \geq \min_i g_i$.*

Proof of Proposition 5. Since $\boldsymbol{g} = \boldsymbol{A}_\sigma \boldsymbol{d}$, it holds that

$$g_{j_{\max}} = d_{j_{\max}} + \sigma(2d_{j_{\max}} - d_{j_{\max}-1} - d_{j_{\max}+1}),$$

where periodicity of subindex are used if necessary. Since $2d_{j_{\max}} - d_{j_{\max}-1} - d_{j_{\max}+1} \geq 0$, We have $\max_i d_i = d_{j_{\max}} \leq g_{j_{\max}} \leq \max_i g_i$. A similar argument can show that $\min_i d_i = d_{j_{\min}} \geq g_{j_{\min}} \geq \min_i g_i$. □

Proposition 6. *The operator $\boldsymbol{A}_\sigma^{-1}$ preserves the sum of components. For any $\boldsymbol{g} \in \mathbb{R}^m$ and $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, we have $\sum_j d_j = \sum_j g_j$, or equivalently, $\mathbf{1}^\top \boldsymbol{d} = \mathbf{1}^\top \boldsymbol{g}$.*

Proof of Proposition 6. Since $\boldsymbol{d} = \boldsymbol{A}_\sigma \boldsymbol{g}$,

$$\sum_i g_i = \mathbf{1}^\top \mathbf{g} = \mathbf{1}^\top(\boldsymbol{I} + \sigma \boldsymbol{D}_+^\top \boldsymbol{D}_+)\boldsymbol{d} = \mathbf{1}^\top \boldsymbol{d} = \sum_i d_i,$$

where we used $\boldsymbol{D}_+\mathbf{1} = \mathbf{0}$. □

Proposition 7. *Given any vector $\boldsymbol{g} \in \mathbb{R}^m$ and $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, then*

$$\|\boldsymbol{d}\| + \sigma\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^2}{\|\boldsymbol{d}\|} \leq \|\boldsymbol{g}\|.$$

The above inequality is strict unless $\boldsymbol{g} = \boldsymbol{d}$ is a constant vector. In particular, we have $\|\boldsymbol{d}\| \leq \|\boldsymbol{g}\|$ and $\|\boldsymbol{D}_+\boldsymbol{d}\| \leq \frac{1}{\sqrt{\sigma}}\|\boldsymbol{g}\|$.

Proof of Proposition 7. By the definition of $\boldsymbol{A}_\sigma$,

$$\boldsymbol{g} = \boldsymbol{A}_\sigma \boldsymbol{d} = (\boldsymbol{I} - \sigma \boldsymbol{D}_- \boldsymbol{D}_+)\boldsymbol{d} = \boldsymbol{d} + \sigma \boldsymbol{D}_+^\top \boldsymbol{D}_+ \boldsymbol{d}. \tag{8}$$

Therefore, pre-multiplying by $\boldsymbol{d}^\top$ on both sides, we have

$$\|\boldsymbol{d}\|^2 + \sigma\|\boldsymbol{D}_+\boldsymbol{d}\|^2 = \boldsymbol{d}^\top \boldsymbol{g} \leq \|\boldsymbol{d}\|\|\boldsymbol{g}\|.$$

In particular, $\|\boldsymbol{d}\| \leq \|\boldsymbol{g}\|$ and $\sigma\|\boldsymbol{D}_+\boldsymbol{d}\|^2 \leq \|\boldsymbol{d}\|\|\boldsymbol{g}\| \leq \|\boldsymbol{g}\|^2$, so $\|\boldsymbol{D}_+\boldsymbol{d}\| \leq \frac{1}{\sqrt{\sigma}}\|\boldsymbol{g}\|$. All the inequalities are strict unless $\|\boldsymbol{D}_+\boldsymbol{d}\| = 0$, and $\boldsymbol{g} = \boldsymbol{d}$ is a constant vector. □

Proposition 8. *For any $\boldsymbol{g} \in \mathbb{R}^m$, define $\mathrm{Var}(\boldsymbol{g}) := \frac{1}{m}\|\boldsymbol{g}\|^2 - \left(\frac{\mathbf{1}^\top \boldsymbol{g}}{m}\right)^2$ be the variance of components in $\boldsymbol{g}$. Let $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, then*

$$\mathrm{Var}(\boldsymbol{d}) \leq \mathrm{Var}(\boldsymbol{g}) - 2\sigma\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^2}{m} - \sigma^2\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^4}{m\|\boldsymbol{d}\|^2}.$$

The inequality is strict unless $\boldsymbol{g} = \boldsymbol{d}$ is a constant vector.

Proof of Proposition 8. Since $\mathbf{1}^\top\boldsymbol{g} = \mathbf{1}^\top\boldsymbol{d}$ and $\|\boldsymbol{d}\| + \sigma\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^2}{\|\boldsymbol{d}\|} \leq \|\boldsymbol{g}\|$,

$$\begin{aligned}\mathrm{Var}(\boldsymbol{g}) &\geq \frac{1}{m}\left(\|\boldsymbol{d}\|^2 + 2\sigma\|\boldsymbol{D}_+\boldsymbol{d}\|^2 + \sigma^2\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^4}{\|\boldsymbol{d}\|^2}\right) - \left(\frac{\mathbf{1}^\top\boldsymbol{d}}{n}\right)^2 \\ &= \mathrm{Var}(\boldsymbol{d}) + 2\sigma\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^2}{m} + \sigma^2\frac{\|\boldsymbol{D}_+\boldsymbol{d}\|^4}{m\|\boldsymbol{d}\|^2}.\end{aligned}$$

The inequality is strict unless $\|\boldsymbol{D}_+\boldsymbol{d}\| = 0$, and $\boldsymbol{g} = \boldsymbol{d}$ is a constant vector. □

Proposition 9. *Given vectors $\boldsymbol{g}$ and $\boldsymbol{d} = \boldsymbol{A}_\sigma^{-1}\boldsymbol{g}$, for any $p \in \mathbb{N}$, it holds that $\|\boldsymbol{D}_+^p\boldsymbol{d}\|_1 \leq \|\boldsymbol{D}_+^p\boldsymbol{g}\|_1$. The inequality is strict unless $\boldsymbol{D}_+^p\boldsymbol{g}$ is a constant vector.*

Proof of Proposition 9. Since $(1+2\sigma)d_i = g_i + \sigma d_{i+1} + \sigma d_{i-1}$, for any $p \in \mathbb{N}$, we have

$$(1+2\sigma)(\boldsymbol{D}_+^p\boldsymbol{d})_i = (\boldsymbol{D}_+^p\boldsymbol{g})_i + \sigma(\boldsymbol{D}_+^p\boldsymbol{d})_{i+1} + \sigma(\boldsymbol{D}_+^p\boldsymbol{d})_{i-1}.$$

So

$$(1+2\sigma)|(\boldsymbol{D}_+^p\boldsymbol{d})_i| \leq |(\boldsymbol{D}_+^p\boldsymbol{g})_i| + \sigma|(\boldsymbol{D}_+^p\boldsymbol{d})_{i+1}| + \sigma|(\boldsymbol{D}_+^p\boldsymbol{d})_{i-1}|.$$

The inequality is strict if there are sign changes among the $(\boldsymbol{D}_+^p\boldsymbol{d})_{i-1}$, $(\boldsymbol{D}_+^p\boldsymbol{d})_i$, $(\boldsymbol{D}_+^p\boldsymbol{d})_{i+1}$. Summing over i and using periodicity, we have

$$(1+2\sigma)\sum_{i=1}^m|(\boldsymbol{D}_+^p\boldsymbol{d})_i| \leq \sum_{i=1}^m|(\boldsymbol{D}_+^p\boldsymbol{g})_i| + 2\sigma\sum_{i=1}^m|(\boldsymbol{D}_+^p\boldsymbol{d})_i|,$$

and the result follows. The inequality is strict unless $\boldsymbol{D}_+^p\boldsymbol{g}$ is a constant vector.

□