Does online gradient descent (and variants) still work with biased gradient and variance?

Ahmad Al-Tawaha[1], and Ming Jin[1]

***Abstract*— Deterministic bias and stochastic unbiased noise refer to different sources of uncertainty in the gradient that can affect the performance of the online learning algorithms. Although existing studies have provided bounds for dynamic regret under these uncertainties, these bounds provide general insight into algorithms' functionality. This paper addresses the efficacy of online gradient-based algorithms (OGD) with inexact gradients. Our analysis quantifies the degree to which gradient estimates can tolerate these uncertainties and identifies specific conditions to ensure the robustness of online gradient descent-based algorithms. Our findings indicate that bias and variance function independently of each other. Also, how much inexactness (variance and bias) online gradient-based algorithms can handle depends on factors including decision dimension, gradient norm, function variations, alignment of functions' gradients, and the curvature of the functions. Our results are verified numerically and experimentally, bridging a significant knowledge gap. Finally, as a case study, we introduce a general online optimization algorithm to explore the interplay between bias and variance with dynamic regret.**

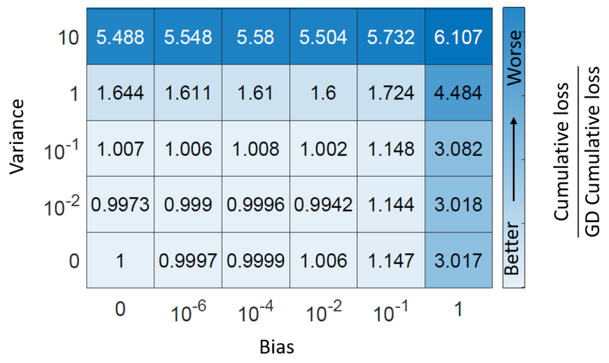

Fig. 1: Online classification on the ijcnn 1 dataset [10], [11] using the logistic loss. In every round, we implement full-batch gradient descent and adjust the gradient estimate by adding varying degrees of variance and bias as a percentage of the gradient norm. Considering the variance and bias, the numeric values indicate the cumulative loss ratio, compared to regular full-batch gradient descent over 400 rounds. The labels are inverted to imitate a changing environment. The results averaged over 25 seeds.

I. Introduction

Optimization problems are central to various applications, including control systems, state estimation of dynamic systems [1], training of deep neural nets [2], optimal power flow problems [3], and resource allocation [4]. Many of these problems are inherently time-sensitive and require adaptation to changing data. This has led to the development of online optimization approaches [5], [6], where solutions must be found sequentially over time and adapted to time-varying input data. The real-time nature of online optimization presents unique challenges and opportunities, accommodating many real-world applications [1], [7], [8].

Online optimization problems can be formulated as a sequential game between a learner and an adversary over discrete time periods. At each time step, a learner chooses an action $x_t \in \mathcal{X}$, where $\mathcal{X} \subset \mathbb{R}^d$, in order to minimize unknown cost function $f_t(.) : \mathcal{X} \to \mathbb{R}$. Then, the learner incurs cost $f_t(x_t)$. Subsequently, the environment reveals some information about the form of f_t given as full information (receiving f_t or $\nabla f_t(x_t)$) or bandit (just $f_t(x_t)$); and the learner uses this information to determine the next action [9].

Our work aims to examine the conditions that affect the inexactness of gradient estimations in online gradient descent (OGD), focusing on understanding the impact of variance and bias on these algorithms. Through theoretical analysis and empirical verification, we seek to answer the question: "Does online gradient descent (and variants) still work with biased gradient and variance?" and explore the boundaries within which inexactness becomes sub-linear, aiding the development of more reliable and effective online learning algorithms.

A particular emphasis is placed on examining the role of bias and variance in gradient estimates, which can significantly affect the performance and reliability of online gradient descent algorithms. Figure 1 demonstrates a straightforward case where the learning process can tolerate certain levels of variance and bias in the gradient estimate without causing significant changes in the cumulative loss. However, these tolerances become detrimental if the levels exceed a certain threshold. The intriguing question that Figure 1 triggers is: to what extent can an online cost functions gradient tolerate variance and bias? Understanding bias and variance in OGD is crucial. Knowing these factors helps us predict how well OGD might work in different situations and under what limit. Also, deep insights into how bias and variance affect OGD can guide the development of improved algorithms. In light of these objectives, our work makes several key contributions:

- Through theoretical analysis and empirical verification, we identified significant factors, including decision dimension, gradient norm, functions variations, alignment of functions' gradients, and the curvature of the func-

[1] Department of Electrical and Computer Engineering, Virginia Tech, VA, USA.

tions (Hessian of functions) that affect the impact of variance and bias on online learning algorithms.
- We presented specific and practical conditions under which online gradient descent and its variants perform reliably even with inexact gradient estimations.
- Our findings extend a guiding framework for understanding the bounds of variance and bias in online learning algorithms, aiming to inform the design of more reliable and effective algorithms in practice.

II. Related Work

Existing research has extensively explored the dynamic regret bounds associated with online gradient descent for convex [12]–[16] and nonconvex functions [17]–[21]. Various sources of errors or uncertainties, including deterministic bias and stochastic noise in the gradient due to algorithmic and hardware constraints, have been acknowledged [22], [23].

Several works have provided dynamic regret bounds when inexact gradients are available [22], [24], [25], often expressed as a sum of regularity measures and cumulative inexactness. The regularity measures include functional variation $V_T := \sum_{t=2}^{T} \sup_{x\in\mathcal{X}} |f_{t-1}(x) - f_t(x)|$ [26], gradient variation $G_T := \sum_{t=2}^{T} \sup_{x\in\mathcal{X}} \|\nabla f_{t-1}(x) - \nabla f_t(x)\|^2$ [27], the path length of the competitor sequence $\mathcal{P}_T := \sum_{t=2}^{T} \left\|x_t^* - x_{t-1}^*\right\|$ introduced in the seminal work [28] and the squared path length $\mathcal{S}_T := \sum_{t=2}^{T} \left\|x_t^* - x_{t-1}^*\right\|^2$ introduces by [14]. The cumulative inexactness is denoted $\mathcal{E}_T = \sum_{t=1}^{T} \epsilon_t$ encapsulating the accumulated inexactness over the considered time frame.

Previous studies commonly assume the sub-linearity of inexactness, yielding valuable insights into the behavior of online gradient algorithms. However, they often overlook the practical conditions under which inexactness is sub-linear and how this impacts the real-world effectiveness of dynamic regret bounds in guiding algorithmic performance. Our work aims to address this gap. Notably, it is understood that achieving sublinear dynamic regret — a critical goal for meaningful upper bounds — is contingent upon both the regularity measure and inexactness being sublinear. Our investigation explores the practical scenarios under which these sublinear conditions are met.

III. The Impact of Stochastic and Deterministic Noise on Decision Update

Our work takes a different path from [29]. While they looked at a single static setting, we explore a scenario where different functions are encountered in each round, which is typical in online learning settings. By drawing insights from [29], [30], we extend the discussion toward understanding sublinear dynamic regret and how online gradient-based algorithms perform in a more complex, changing environment, which helps fill an important knowledge gap. The instantaneous cost function at a specific point on the optimization trajectory is denoted as $f_t(x_t)$, and the instantaneous cost function at the subsequent point on the path, that is, after a decision update step, is represented by $f_{t+1}(x_{t+1})$. Also, $x_{t+1} = x_t + \eta\Delta x_t$, where η is the learning rate. In the remaining parts of this work, we define the cumulative loss for a single decision update $f_{t+1}(x_{t+1}) + f_t(x_t)$ as a function of the variance and bias in the decision update.

We assume that all cost functions $\{f\}_{t=1}^{T}$ are twice differentiable and the step size η is sufficiently small, allowing us to effectively approximate $f_{t+1}(x_{t+1})$ around $f_{t+1}(x_t)$ using a second-order Taylor series. We represent the bias vector as $b\vec{\beta}$, where $\vec{\beta}$ is a vector with norm $\sqrt{d}$, showing the bias direction in the d-dimensional decision space. With these premises, we formulate the decision update equation to be:

$$\Delta x_t := -\nabla_x f_t(x_t) + b\vec{\beta} + \sigma\sqrt{h(d)}\hat{d}, \tag{1}$$

In this context, $\nabla_x f_t(x_t)$ represents the first derivative of the loss function f_t at the current decision x_t, $\hat{d}$ refers to a unit vector in the d-dimensional parameter space, where each element of this vector is zero-mean i.i.d. The total variance of Δx_t is $\sigma^2 h(d)$. Here, $h(d)$ characterizes how the variance changes as a function of the decision dimension. We define $\Delta f\left(b,\sigma^2\right)$ as the change in a cumulative loss of two consecutive cost functions $f_{t+1}(x_{t+1}) + f_t(x_t)$ when performing the aforementioned decision update as compared to the change in cumulative loss when the decision update step follows the true gradient, i.e.,

$$\Delta f\left(b,\sigma^2\right) = \left[f_{t+1}(x_{t+1}) + f_t(x_t)\right]_{(b,\sigma^2)} - \left[f_{t+1}(x_{t+1}) + f_t(x_t)\right]_{(0,0)}, \tag{2}$$

where $\left[f_{t+1}(x_{t+1}) + f_t(x_t)\right]_{(b,\sigma^2)}$ is the cumulative loss with bias b and variance σ^2 for single parameter update. The formal description is in the following Lemma.

Lemma 3.1: Assuming a small learning rate η, the bias, b, and variance, σ^2, in gradient estimates can be linked to changes in a cumulative loss upon one decision update step as compared to the cumulative loss under a true online gradient descent decision update step:

$$\begin{aligned}
&\mathbb{E}_{\hat{d}}\left[\Delta f\left(b,\sigma^2\right)\right] \\
&= \mathbb{E}_{\hat{d}}\left[\left[f_{t+1}(x_{t+1}) + f_t(x_t)\right]_{(b,\sigma^2)} - \left[f_{t+1}(x_{t+1}) + f_t(x_t)\right]_{(0,0)}\right] \\
&= b\left\langle \nabla_x f_{t+1}(x_t), \vec{\beta}\right\rangle \eta + \frac{1}{2}b^2\left\langle \vec{\beta}, \nabla_x^2 f_{t+1}(x_t)\vec{\beta}\right\rangle \eta^2 \\
&- \frac{1}{2}b\left\langle \nabla_x f_t(x_t), \left(\nabla_x^2 f_{t+1}(x_t) + \nabla_x^2 f_{t+1}(x_t)^T\right)\vec{\beta}\right\rangle \eta^2 \\
&+ \frac{1}{2}\frac{\sigma^2 h(d)}{d}\operatorname{Tr}\left[\nabla_x^2 f_{t+1}(x_t)\right]\eta^2.
\end{aligned} \tag{3}$$

Due to space limitations, we provide all the proofs in this online document [31]. Unpacking the implications of Lemma 3.1, we can identify several critical elements that significantly shape the performance of online learning algorithms:

Separation of bias and variance effects: The effects of variance and bias on the expected change in cumulative loss, represented as $\mathbb{E}_{\hat{d}}\left[\Delta f\left(b,\sigma^2\right)\right]$, function independently of each other. This allows for individual optimization of each component - bias, and variance - to minimize the cumulative loss.

Dual impact of bias: Bias influences the loss in two distinct ways: linearly and quadratically. The first term in

Lemma's expression presents a linear impact with respect to bias, whereas the second term encapsulates the quadratic impact.

The insights from Lemma 3.1 pave the way for optimizing online learning algorithms. The decoupling of bias and variance facilitates targeted algorithm tuning, allowing for an individual examination of each effect in the subsequent analysis.

Lemma 3.2: The impact of variance on Δf is proportional to the trace of the Hessian.

$$\Delta f(0,\sigma^2) = \frac{1}{2}\frac{\sigma^2 h(d)}{d}\,\mathrm{Tr}\left[\nabla_x^2 f_{t+1}(x_t)\right]\eta^2.$$

Variance's impact is directly proportional to the trace of the Hessian of the subsequent loss function, $\nabla_x^2 f_{t+1}(x_t)$. The Hessian matrix represents the loss function's curvature, and the decision variable's dimensionality significantly influences the overall loss. This suggests that the structure of the loss functions can significantly influence the overall loss. While the impact of variance is easily discernible from Lemma 3.2, we further inspect the previous formula for a specific bias value denoted as b while keeping the variance, σ, equal to zero.

Lemma 3.3: The impact of bias on Δf is linearly proportional to the gradient norm.

$$\Delta f(b,0) = \\ \frac{1}{2}b^2 Q\eta^2 + b\left\|\nabla_x f_{t+1}(x_t)\right\| P\eta - \frac{1}{2}b\left\|\nabla_x f_t(x_t)\right\| R\eta^2 \quad (4)$$

where $Q = \left\langle \overrightarrow{\beta}, \nabla_x^2 f_{t+1}(x_l)\,\overrightarrow{\beta}\right\rangle$, $P = \left\langle \nabla_x \widehat{f_{t+1}}(x_t), \overrightarrow{\beta}\right\rangle$ and $R = \left\langle \nabla_x \widehat{f_t(x_t}), \left(\nabla_x^2 f_{t+1}(x_l) + \nabla_x^2 f_{t+1}(x_t)^T\right)\overrightarrow{\beta}\right\rangle$.

The terms $Q, R,$ and$P$ are independent of the gradient's norm. According to Lemma 3.3, the impact of bias on the function is intricately tied to the norm and direction of the gradient, the direction of the bias vector $\overrightarrow{\beta}$, and the eigenvectors of the loss Hessian. These dependencies lead to specific implications, which we explore in the following:

Quadratic term and curvature: The term $\frac{1}{2}b^2 Q\eta^2$ exhibits a quadratic dependency on the bias and captures the influence of the curvature of the upcoming loss function. The alignment of the bias direction $\overrightarrow{\beta}$ with the curvature of the loss function in that particular direction is embodied within the quantity Q. Interestingly, this term remains unaffected by the gradient norm, allowing it to become dominant when it is potentially small.

Linear term and gradient norm of an upcoming cost function: The term $b\left\|\nabla_x f_{t+1}(x_t)\right\| P\eta$ is linear in b and directly proportional to the gradient norm. When the gradient norm is small, this term will be reduced, potentially diminishing the linear effect of bias on Δf. Additionally, since P includes an inner product with $\overrightarrow{\beta}$, the effect of this term depends on the alignment between the bias direction and the direction of the gradient of an upcoming cost function.

Negative linear dependence on the bias with gradient norm modulation of current cost function: The term $-\frac{1}{2}b\left\|\nabla_x f_t(x_t)\right\| R\eta^2$ introduces a negative linear dependence with respect to the bias b, and is modulated by the norm of the gradient of the current cost function. The factor R encapsulates the intricate interplay between the directions of the bias, the current gradient, and the curvature of upcoming loss functions.

After examining the individual effects of bias and variance on Δf, an essential question arises: At what level does variance cause online gradient descent to degrade? To pinpoint the specific variance threshold leading to a deterioration in the learning process, we present the following lemma:

Lemma 3.4: Consider the online gradient descent algorithm. Define the dynamic regret at time t with variance σ^2 as $\mathrm{DR}(t)_{(0,\sigma^2)} = \left[f_t(x_t) - f_t(x_t^*)\right]_{(0,\sigma^2)}$. The regret is growing at a constant rate ($\mathrm{DR}(t+1)_{(0,\sigma^2)} - \mathrm{DR}(t)_{(0,\sigma^2)} \geq C_1$) if the variance satisfied:

$$\sigma^2 \geq \frac{2C_1}{\lambda_{\min}\eta^2} + \frac{2\left(f_t(x_t) - f_t^*\right) - \left(f_{t+1}(x_t) - f_{t+1}^*\right)}{\lambda_{\min}\eta^2} \\ + \frac{2\nabla f_{t+1}^\top(x_t)\nabla f_t(x_t)}{\lambda_{\min}\eta} - \frac{\nabla f_t^\top(x_t)\left(\nabla^2 f_{t+1}(x_t)\right)\left(\nabla f_t(x_t)\right)}{\lambda_{\min}}, \quad (5)$$

where $C_1 > 0$ and $\lambda_{\max} \geq \ldots \geq \lambda_{\min}$ denotes the eigenvalues of $\nabla^2 f_{t+1}$, the equality holds with variance threshold value denoted as σ_T^2. The condition in the lemma implies that the dynamic regret with variance grows by at least a constant amount at every step. The condition assumes that $\mathrm{DR}(t+1)_{(0,0)} \leq \mathrm{DR}(t)_{(0,0)}$ holds, reflecting the inherent characteristics of the environment rather than a failure in the algorithm because of the variance. The right-hand side of the inequality provides a safety threshold for the variance. The online gradient descent becomes unstable when the variance exceeds the threshold value.

Various factors contribute to this threshold, such as the curvature of the upcoming loss function. Specifically, a smaller $\lambda_{\min}$ corresponding to a flatter curvature would increase the right-hand side of the inequality, making the algorithm more robust to variance. In regions with steep curvatures, a small noise in the gradient can cause the algorithm to take a large, possibly incorrect step, leading to significant deviations. However, in flatter regions, the inherent nature of the function means it is less sensitive to these errors. A noisy gradient in a flat region will not drastically change the algorithm's step as it would in a steeper region. Interestingly, that is aligned with the result from [32]–[34] as variance help in escaping the local minimum "sharp curvature" point locally and gives better generalization. The flatter regions act as a buffer. Even if the gradient has some noise due to variance, the update step taken by the algorithm will not push the solution too far.

The second term represents the difference between the regrets of two consecutive functions f_t and f_{t+1} evaluated at the same point x_t. If this term is large, it signifies that the regret due to f_t was substantially more than that due to f_{t+1} at the same point x_t. Since variance adds uncertainty and potentially increases regret, a substantial inherent reduction (like a buffer) in regret without considering variance allows the algorithm to tolerate more variance without the total regret (inherent plus that due to variance) growing too much.

Thus, the larger it is, the more robustness you have against variance.

The inner product $\nabla f_{t+1}^{\top}(x_t)\nabla f_t(x_t)$ provides information on how aligned the gradients of the two functions are at the point x_t. If the value is positive, both gradients point in the same general direction. The "redundant information" from f_{t+1} can be an error-checking mechanism. To use an analogy, imagine two people giving you directions. If both are mostly pointing you to the same path, even if one is a little uncertain (analogous to the noise in gradient), you'd still have a clear idea of where to go based on the consistent information. This is the notion of redundancy. However, if they point in opposite directions, any uncertainty from either person makes the decision even harder.

The analysis unveils an interplay of factors determining the variance tolerance of the online gradient descent algorithm. Firstly, flatter regions serve as buffers against variance-induced deviations, with smaller $\lambda_{\min}$ values rendering the algorithm more robust against variance. Secondly, the regret difference between consecutive functions f_t and f_{t+1} at x_t acts as an inherent reduction in regret, thus allowing a greater tolerance for variance. Lastly, the alignment of gradients of f_t and f_{t+1} at x_t serves as an error-checking mechanism, enabling better decision-making even amidst gradient noise.

After analyzing the impact of variance, we now focus on bias. The critical question is: At what level does bias cause online gradient descent to degrade? While the complexities of bias make it a more challenging subject to tackle fully in this part, we will provide insights into its first-order Tayler effects. A more detailed analysis, especially concerning the second-order Tayler effects of bias, can be found in the online document [31].

Lemma 3.5: Consider the online gradient descent algorithm. Define the dynamic regret at time t with biased b in the gradient as $\mathrm{DR}(t)_{(b,0)} = [f_t(x_t) - f_t(x_t^*)]_{(b,0)}$. The regret is growing at a constant rate ($\mathrm{DR}(t+1)_{(b,0)} - \mathrm{DR}(t)_{(b,0)} \geq C_2$), if and only if the bias satisfied:

$$\mathrm{b} \geq \frac{C_2 + (f_t(x_t) - f_t^*) - \left(f_{t+1}(x_t) - f_{t+1}^*\right) + \nabla f_{t+1}^{\top}(x_t)\nabla f_t(x_t)\eta}{\sqrt{d}\ \|\nabla f_{t+1}\|\ \eta}. \quad (6)$$

The equality holds with bias threshold value denoted as b_T. To delve into the practical implications of this lemma, let's examine how the bias b in the gradient can affect the algorithm's performance. This deterioration is quantified as an increase in dynamic regret by at least a constant C_2. Moreover, the bias threshold is inversely related to the square root of the dimension d of the decision variable. This relation implies that as the dimensionality of our problem increases (i.e., the number of decision variables), the amount of bias the algorithm can tolerate decreases. In practical terms, in high-dimensional spaces (large d), even a small bias can lead to significant errors in optimization.

Two terms we have already discussed in the previous lemma. The term showing the difference in regrets between f_t and f_{t+1} at x_t tells us about natural differences. If this term is large, our algorithm can handle these differences without causing problems. Then, the other term about the gradients, $\nabla f_{t+1}^{\top}(x_t)\nabla f_t(x_t)$. In the variance context, this term helped us understand how similar the directions of the two functions are. Here, in the context of bias, this similarity tells us if one function's bias might be balanced out by the other, making the algorithm more stable.

IV. Case study: Inexact Online Multiple Gradient Descent (IOMGD) algorithm

Exploring the bias and variance with dynamic regret is an important aspect. In this section, we analyze the IOMGD algorithm. A particular interest of this algorithm is its inherent flexibility. Specifically, by setting the number of inner iterations to $K = 1$ and $h = 1$, the algorithm reduces to the traditional inexact online gradient descent. For clarification, here is the algorithm description: For each round t up to T, the learner submits a decision x_t within the set $\mathcal{X}$ and initializes z_t^1 to be equal to x_t. Then, for each step j up to K (where K is the total number of multiple steps), the learner receives an inexact gradient $\hat{\nabla} f_t\left(z_t^j\right)$. Subsequently, the learner updates an auxiliary variable $\hat{z}_t^j$ by applying projected gradient descent on the current iterate z_t^j. The variable z_t^{j+1} can be seen as a weighted blend of the previous iterations z_t^j, and the auxiliary variable, $\hat{z}_t^j$. At the end of these K steps, the decision for the next round, x_{t+1}, is set to be z_t^{K+1}. This process is repeated for each round until the final round T is reached. The general procedure is summarized in Algorithm 1.

Algorithm 1 Inexact online multiple gradient descent

Input: The number of inner iterations K and the step size η

```
1:  Let $x_1$ be any point in $\mathcal{X}$.
2:  for $t = 1, \ldots, T$ do
3:      Submit $x_t \in \mathcal{X}$
4:      $z_t^1 = x_t$
5:      for $j = 1, \ldots, K$ do
6:          Receive an inexact gradient $\hat{\nabla} f_t(z_t^j)$.
7:          Update $\hat{z}_t^j = P_{\mathcal{X}}\left(z_t^j - \eta\hat{\nabla} f_t\left(z_t^j\right)\right)$.
8:          Update $z_t^{j+1} = z_t^j + h\left(\hat{z}_t^j - z_t^j\right)$
9:      end for
10:     $x_{t+1} = z_t^{K+1}$
11: end for=0
```

In the following results, we assume the following assumptions are satisfied. The gradients of all the online functions are bounded by G, i.e.,

$$\sup_{\mathbf{x}\in\mathcal{X}, 1\leq t\leq T} \|\nabla f_t(x)\| \leq G.$$

The functions f_t are $L-$smooth and $\mu-$strongly convex over the convex set $\mathcal{X}$

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

The strong convexity assumption implies that each function f_t has a unique minimizer within the convex set, x_t^*. And the smoothness assumption is standard in differential optimization [35]. Lastly, The bound on the norm of the gradients is typical in the analysis of online algorithms for constrained optimization. Start by examining how to establish regret bound when the gradient is inexact. The proof is broken down into three main steps. The first step is to determine a limit for the quantity $\|x_{t+1}-x_t\|^2$ based on gradient error and the distance between x_t and x_t^* for one step IOGD. The second step is bound the quantity $\|x_{t+1}-x_t\|^2$ by implementing the presented algorithm. And finally, setting a limit for the cumulative sum of $\|x_{t+1}-x_t\|^2$. Our main results on the regret bound of IMOGD in dynamic settings are derived from the following:

Lemma 4.1: The sequence $\{x_t\}$ for all $t \in \mathbb{N}$ generated by IOGD algorithm with a step size $\eta \leq \frac{1}{L}$ for $L-$smooth and $\mu-$strongly convex $f(\cdot)$ satisfies the following bounds for any sequence $\{x_t^*\}$

$$\mathbb{E}_{\hat{d}}\|z_t^{j+1}-x_t^*\|^2 \leq \left(1-\frac{h\mu}{L}\right)\|z_t^j-x_t^*\|^2+\frac{h}{L}\epsilon_t,$$

$$\|x_{t+1}-x_t^*\|^2 \leq \beta^K\|x_t-x_t^*\|^2+\frac{1}{\mu}\epsilon_t,$$

$$\sum_{t=1}^{T}\|x_t-x_t^*\|^2 \leq \frac{1}{1-2\beta^K}\left(\|x_1-x_1^*\|^2+2\mathcal{E}_T+2\mathcal{S}_T^*\right),$$

where $\beta = 1-\frac{h\mu}{L}$, $\epsilon_t = 2D\left(b_t\sqrt{d}+\sigma_t\right)$, and $\mathcal{E}_T = \frac{1}{\mu}\sum_{t=2}^{T}\epsilon_t$.

Using the cumulative sum, we determine the dynamic regret upper bound. The following corollaries present two different upper bounds on the dynamic regret of the IOMGD algorithm. They are based on the path length $\mathcal{P}_T^*$, and the squared path length $\mathcal{S}_T^*$ as a regularity measures.

Corollary 4.1: (Bounding the dynamic regret by $\mathcal{P}_T^*$ and $\tilde{\mathcal{E}}_T$): Assume that $f_t(\cdot)$ is $L-$ smooth and $\mu-$ strongly convex by setting $\eta \leq \frac{1}{L}$ in Algorithm 1, we have

$$\sum_{t=1}^{T}\mathbb{E}_{\hat{d}}\left[f_t\left(x_t\right)\right]-f_t\left(x_t^*\right) \leq \frac{G}{1-\sqrt{\beta^K}}\Big(\|x_1-x_1^*\| -\beta^{\frac{K}{2}}\mathbb{E}_{\hat{d}}\left[\|x_T-x_T^*\|\right]+\mathcal{P}_T^*+\tilde{\mathcal{E}}_T\Big)$$

where $\tilde{\mathcal{E}}_T = \sqrt{\mathcal{E}_T}$.

When the value of $\tilde{\mathcal{E}}_T$ is equal to zero, we obtain the same dynamic regret bound in [36]. Also, the previous dynamic regret bound recovers the dynamic regret bound obtained in [22], [37] with an improvement as the sum of the inexactness appears under the square root.

Corollary 4.2: (Bounding the dynamic regret by $\mathcal{S}_T^*$ and $\mathcal{E}_T$): Assume that $f_t(\cdot)$ is $L-$ smooth and $\mu-$ strongly convex by setting $\eta \leq \frac{1}{L}$ in Algorithm 1, and for any $\theta > 0$, we have

$$\sum_{t=1}^{T}\mathbb{E}_{\hat{d}}[f_t(x_t)]-f_t(x_t^*) \leq \frac{1}{2\theta}\sum_{t=1}^{T}\|\nabla f_t(x_t^*)\|^2 +\mathcal{C}_1\left(\|x_1-x_1^*\|^2+2\mathcal{S}_T^*+4\mathcal{E}_T^*\right),$$

where $\mathcal{C}_1 = \frac{\theta+L}{2(1-2\beta^K)}$.

Compared to Corollary4.1, the proposed IOMGD improves the dynamic regret by reducing it from $\mathcal{O}\left(\mathcal{P}_T^*+\mathcal{E}_T\right)$ to $\mathcal{O}\left(\min\left(\mathcal{P}_T^*+\tilde{\mathcal{E}}_T, \mathcal{S}_T^*+\mathcal{E}_T^*\right)\right)$. Note that the gradient at x_t^* is zero when x_t^* is in the relative interior of the feasibility set $\mathcal{X}$; as a result, the dynamic regret in the above theorem can be simplified to $\mathcal{O}\left(\mathcal{S}_T^*+\mathcal{E}_T^*\right)$. In a slowly changing environment, where the distance between successive optimal solutions is small, the squared path length $\mathcal{S}_T^*$ can be significantly smaller than the path length $\mathcal{P}_T^*$. In this case, Corollary 4.2 can provide an improved regret bound than the result in Corollary 4.1.

The regret bounds of the IOMGD algorithm, as presented in Lemma 4.1, Corollary 4.1, and Corollary 4.2, provide upper bounds on dynamic regret. These bounds are influenced by parameters related to the gradient bound, strong convexity, and a smoothness parameter L (curvature of the function). Notably, they also include path lengths, squared path lengths, and inexactness terms – the latter being the cumulative sum of bias and variance at each iteration. However, as highlighted in the earlier discussions on the roles of bias and variance in Lemma 3.4 and 3.5, dynamic regret can exhibit complex interdependencies. For bounded dynamic regret, both bias and variance must be sub-linear. This sub-linearity, as derived in Lemma 3.4 and 3.5, hinges on factors such as the decision dimension, gradient norm, both the lower and upper bounds of the function's hessian and the alignment of consecutive gradients that is not evident from the dynamic regret upper bound alone.

V. Experiments

Investigating variance sensitivity in Online Gradient Descent: The primary objective of this experiment is to investigate the robustness of the OGD algorithm against variance and bias. This setting allows a focused exploration of the algorithm's performance under variance and bias. We utilize a quadratic loss function f defined as:

$$f(x) = \frac{1}{2}x^\top \mathbf{A}x - \mathbf{b}^\top x,$$

where the decision dimension $d = 10$. To methodically understand the impact of variance, we derive a variance threshold value σ_T^2 at each epoch, based on Equation 5. We then corrupt the gradients with various fractions of this threshold value to introduce a controlled perturbation level. The results, illustrated in Figure 2, demonstrate that once the algorithm's operations surpassed the defined σ_T^2, a marked divergence in the algorithm's performance was observed. This highlights the significance of the variance conditions delineated in Equation 5.

Subsequently, we evaluated the algorithm's sensitivity to bias by computing a bias threshold, b_T, at each epoch, as described in Equation 6. Figure 3 clarifies our findings, indicating that exceeding the bias threshold leads to the algorithm's divergence. Notably, divergence can occur even before the threshold value is reached; however, surpassing the threshold value guarantees the divergence.

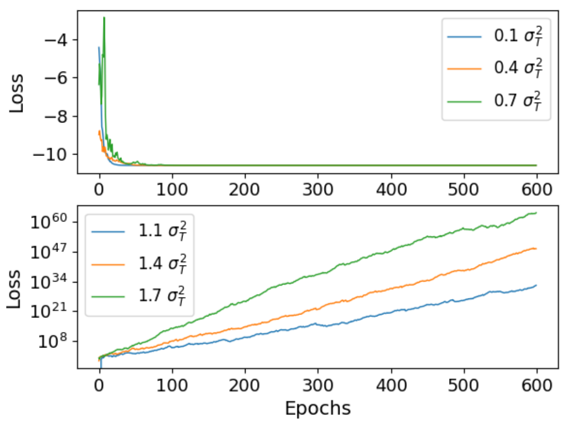

Fig. 2: Sensitivity of the OGD algorithm to variance: The plot illustrates the performance degradation of the OGD algorithm as the variance exceeds the derived threshold value (σ_T^2).

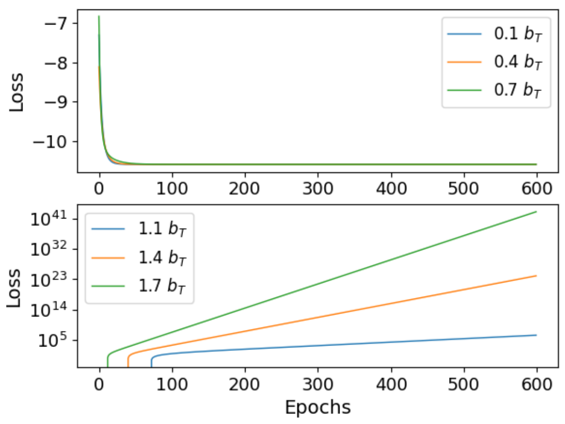

Fig. 3: Sensitivity of the OGD algorithm to bias: This plot shows the impact on the OGD algorithm performance when the bias exceeds the computed threshold value (b_T).

Synthetic Experiment: The primary objective of this experiment is to validate the theoretical findings proposed in Lemma 3.5, specifically in a synthetic environment with controlled conditions. The experiment setup is adapted from [38], facilitating a structured examination of bias sensitivity in online learning scenarios.

The loss function chosen for this experiment is a time-dependent quadratic function with a varying center point, mathematically represented as $f_t(x) = \frac{1}{4}(x - y_t)^2$ with $y_t = 100 \sin\left(\pi \frac{t}{10T}\right)$. Here, the time t spans a horizon $T = 2000$, causing a sinusoidal variation in the center of the quadratic function, which induces a dynamic shift in the function's optimum.

A bias is introduced to the function's gradient to model the gradient's inexactness. The bias threshold component b_T is derived based on the defined upper limit given in Equation 6. Different fractions of the bias threshold $\underline{T}$ are considered to mimic the gradient's inexactness.

The performance analysis is carried out by observing the cumulative loss as different bias levels are introduced. Figure 3 illustrates the resultant unbounded cumulative loss when the threshold value is exceeded.

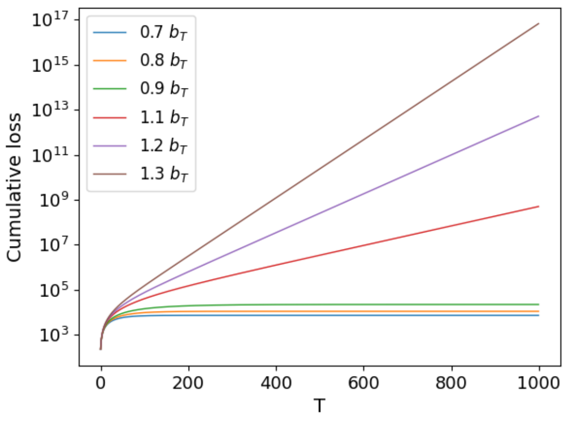

Fig. 4: Illustration of unbounded cumulative loss with exceeding bias threshold values in a synthetic environment, validating the bias sensitivity as shown in Lemma 3.5.

Foreground and Background Separation: In this part, we evaluate the efficacy of Algorithm 1 in foreground and background separation—a pivotal task in computer vision aimed at isolating objects of interest (foreground) from the background within an image or video. The complexity of this task escalates when foreground and background elements share similar color and texture characteristics or overlap. Utilizing online learning, particularly the IOMGD algorithm, enhances the accuracy and robustness of foreground and background separation, ultimately leading to improved separation outcomes.

The process involves decomposing a matrix Y_t into a low-rank component L_t and a sparse component S_t at every time instance t. The loss function employed for this purpose is formulated as:

$$f_t(L_t, S_t) = \|Y_t - L_t - S_t\|_F^2 + \mu_L \|L_t\|_F^2 + \mu_S \|S_t\|_F^2.$$

The full gradient version of the presented algorithm is computationally expensive and sometimes infeasible since it requires the knowledge of the full matrix Y_t at each t. To simulate this scenario and demonstrate the performance of the proposed algorithm, a normal white noise $X \sim \mathcal{N}(0, \sigma^2)$ is added to the gradient of the loss function $f_t(L_t, S_t)$, which satisfies the conditions outlined in 5.

Figure 5 shows the performance evaluation of IOMGD algorithm, where $K = 3$ exhibited a reduction in cumulative loss compared to a single gradient descent step.

Figure 6 provides a visual representation of low-rank component separation using both exact and inexact multiple gradients with $K = 3$. The top image portrays an original video instance Y_t, while the middle and bottom images show L_t derived through exact and inexact gradients, respectively.

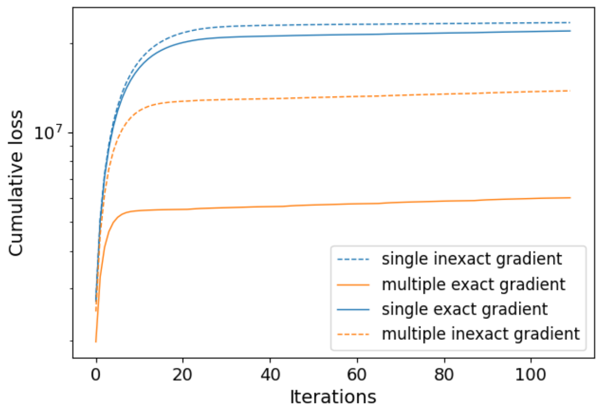

Fig. 5: Illustration of cumulative loss for the IOMGD algorithm over various iterations. The figure demonstrates the algorithm's ability to manage loss accumulation across iterations, showcasing its efficacy and stability in the foreground and background separation task.

VI. Conclusion

This paper tackled a key question of the efficacy of online gradient-based algorithms with inexact gradients. We bridged an existing knowledge gap, establishing clear conditions for these algorithms to work well. Using mathematical analysis and empirical verification, we analyzed the impact of uncertainties for single parameter updates. We found that bias and variance functioned independently of each other. Also, how much inexactness (variance and bias) online gradient-based algorithms can handle depends on factors including decision dimension, gradient norm, functions variations, alignment of functions' gradients, and the curvature of the functions. Future research includes developing adaptive algorithms that dynamically respond to varying degrees of inexactness.

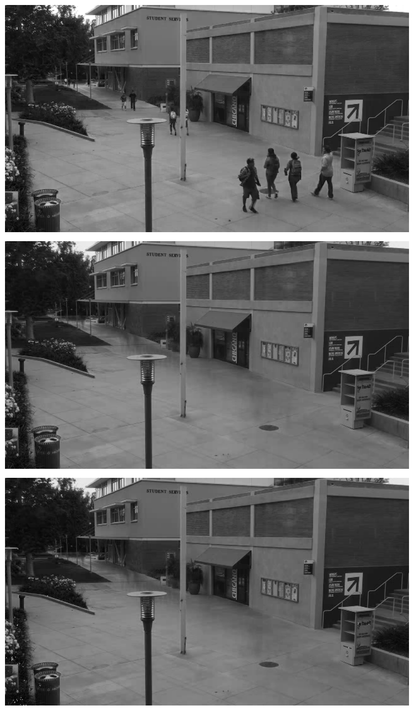

Fig. 6: Illustration of Low-rank Components Separation via exact and inexact multiple gradients ($K = 3$). The top image represents the original instance Y_t from a video; the middle and bottom images depict L_t obtained through exact and inexact gradients, respectively.

References

[1] C. V. Rao, J. B. Rawlings, and D. Q. Mayne, "Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations," *IEEE transactions on automatic control*, vol. 48, no. 2, pp. 246–258, 2003.

[2] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," *Advances in neural information processing systems*, vol. 31, 2018.

[3] F. Zohrizadeh, C. Josz, M. Jin, R. Madani, J. Lavaei, and S. Sojoudi, "Conic relaxations of power system optimization: Theory and algorithms," *European Journal of Operational Research*, vol. 287, no. 2, pp. 391–409, 2020.

[4] T. Wang and S. Wang, "Online convex optimization for efficient and robust inter-slice radio resource management," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 6050–6062, 2021.

[5] E. Hazan *et al.*, "Introduction to online convex optimization," *Foundations and Trends® in Optimization*, vol. 2, no. 3-4, pp. 157–325, 2016.

[6] N. Cesa-Bianchi and F. Orabona, "Online learning algorithms," *Annual review of statistics and its application*, 2021.

[7] M. S. Asif and J. Romberg, "Sparse recovery of streaming signals using ℓ_1-homotopy," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4209–4223, 2014.

[8] Y. Tang, K. Dvijotham, and S. Low, "Real-time optimal power flow," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2963–2973, 2017.

[9] S. Shalev-Shwartz *et al.*, "Online learning and online convex optimization," *Foundations and Trends® in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2012.

[10] C.-C. Chang, "" libsvm: a library for support vector machines," acm transactions on intelligent systems and technology, 2: 27: 1–27: 27, 2011," *http://www. csie. ntu. edu. tw/˜ cjlin/libsvm*, vol. 2, 2011.

[11] D. Prokhorov, "Ijcnn 2001 neural network competition," *Slide presentation in IJCNN*, vol. 1, no. 97, p. 38, 2001.

[12] E. C. Hall and R. M. Willett, "Online convex optimization in dynamic environments," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 647–662, 2015.

[13] A. Mokhtari, S. Shahrampour, A. Jadbabaie, and A. Ribeiro, "Online optimization in dynamic environments: Improved regret rates for strongly convex problems," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 7195–7201.

[14] L. Zhang, T. Yang, J. Yi, R. Jin, and Z.-H. Zhou, "Improved dynamic regret for non-degenerate functions," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[15] P. Zhao and L. Zhang, "Improved analysis for dynamic regret of strongly convex and smooth functions," in *Learning for Dynamics and Control*. PMLR, 2021, pp. 48–59.

[16] N. Eshraghi and B. Liang, "Dynamic regret bounds without lipschitz continuity: Online convex optimization with multiple mirror descent steps," in *2022 American Control Conference (ACC)*. IEEE, 2022, pp. 228–235.

[17] L. Yang, L. Deng, M. H. Hajiesmaili, C. Tan, and W. S. Wong, "An optimal algorithm for online non-convex learning," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 2, no. 2, pp. 1–25, 2018.

[18] O.-A. Maillard and R. Munos, "Online learning in adversarial lipschitz environments," in *Joint european conference on machine learning and knowledge discovery in databases*. Springer, 2010, pp. 305–320.

[19] E. Hazan and S. Kale, "Online submodular minimization." *Journal of Machine Learning Research*, vol. 13, no. 10, 2012.

[20] L. Zhang, T. Yang, R. Jin, and Z.-H. Zhou, "Online bandit learning

for a special class of non-convex losses," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, 2015.

[21] X. Gao, X. Li, and S. Zhang, "Online learning with non-convex losses and non-stationary regret," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018, pp. 235–243.

[22] A. S. Bedi, P. Sarma, and K. Rajawat, "Tracking moving agents via inexact online gradient descent algorithm," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 202–217, 2018.

[23] S. Bubeck *et al.*, "Convex optimization: Algorithms and complexity," *Foundations and Trends® in Machine Learning*, vol. 8, no. 3-4, pp. 231–357, 2015.

[24] W. Choi, M.-S. Lee, and S.-B. Yun, "Inexact online proximal mirror descent for time-varying composite optimization," *arXiv preprint arXiv:2304.04710*, 2023.

[25] U. Syed, E. Dall'Anese, and B. Hu, "Bounds for the tracking error and dynamic regret of inexact online optimization methods: A general analysis via sequential semidefinite programs," *arXiv preprint arXiv:2303.00937*, 2023.

[26] O. Besbes, Y. Gur, and A. Zeevi, "Non-stationary stochastic optimization," *Operations research*, vol. 63, no. 5, pp. 1227–1244, 2015.

[27] A. Jadbabaie, A. Rakhlin, S. Shahrampour, and K. Sridharan, "Online optimization: Competing with dynamic comparators," in *Artificial Intelligence and Statistics*. PMLR, 2015, pp. 398–406.

[28] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the 20th international conference on machine learning (icml-03)*, 2003, pp. 928–936.

[29] A. Ghosh, Y. H. Liu, G. Lajoie, K. Kording, and B. A. Richards, "How gradient estimator variance and bias impact learning in neural networks," in *The Eleventh International Conference on Learning Representations*, 2023.

[30] D. V. Raman, A. P. Rotondo, and T. O'Leary, "Fundamental bounds on learning performance in neural circuits," *Proceedings of the National Academy of Sciences*, vol. 116, no. 21, pp. 10 537–10 546, 2019.

[31] A. Al-Tawaha and M. Jin, "Does online gradient descent (and variants) still work with biased gradient and variance?" 2023. [Online]. Available: http://www.jinming.tech/papers/IOGD2023.pdf

[32] B. Kleinberg, Y. Li, and Y. Yuan, "An alternative view: When does sgd escape local minima?" in *International conference on machine learning*. PMLR, 2018, pp. 2698–2707.

[33] C. Jin, R. Ge, P. Netrapalli, S. M. Kakade, and M. I. Jordan, "How to escape saddle points efficiently," in *International conference on machine learning*. PMLR, 2017, pp. 1724–1732.

[34] R. Ge, F. Huang, C. Jin, and Y. Yuan, "Escaping from saddle points—online stochastic gradient for tensor decomposition," in *Conference on learning theory*. PMLR, 2015, pp. 797–842.

[35] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2003, vol. 87.

[36] P. Nazari, D. A. Tarzanagh, and G. Michailidis, "Dadam: A consensus-based distributed adaptive gradient method for online optimization," *IEEE Transactions on Signal Processing*, 2022.

[37] R. Dixit, A. S. Bedi, R. Tripathi, and K. Rajawat, "Online learning with inexact proximal online gradient descent algorithms," *IEEE Transactions on Signal Processing*, vol. 67, no. 5, pp. 1338–1352, 2019.

[38] N. Campolongo and F. Orabona, "Temporal variability in implicit online learning," *Advances in neural information processing systems*, vol. 33, pp. 12 377–12 387, 2020.

VII. Appendix

Lemma 7.1: Assuming a small learning rate η, the bias, b, and variance, σ^2, in gradient estimates can be linked to changes in a cumulative loss upon one decision update step as compared to the cumulative loss under a true online gradient descent decision update step:

$$\begin{aligned}\mathbb{E}_{\hat{d}}\left[\Delta f\left(b,\sigma^2\right)\right] &= \mathbb{E}_{\hat{d}}\left[\left[f_{t+1}\left(x_{t+1}\right)+f_t(x_t)\right]_{(b,\sigma^2)}-\left[f_{t+1}\left(x_{t+1}\right)+f_t(x_t)\right]_{(0,0)}\right], \\ &= b\left\langle \nabla_x f_{t+1}(x_t), \overrightarrow{\beta}\right\rangle\eta+\frac{1}{2}b^2\left\langle\overrightarrow{\beta},\nabla_x^2 f_{t+1}(x_t)\overrightarrow{\beta}\right\rangle\eta^2-\frac{1}{2}b\left\langle\nabla_x f_t(x_t),\left(\nabla_x^2 f_{t+1}(x_t)+\nabla_x^2 f_{t+1}(x_t)^T\right)\overrightarrow{\beta}\right\rangle\eta^2 \\ &+\frac{1}{2}\frac{\sigma^2 h(d)}{d}\operatorname{Tr}\left[\nabla_x^2 f_{t+1}(x_t)\right]\eta^2.\end{aligned}$$

A. *Proof of Lemma* ??

Proof: We use the second-order Taylor series expansion of the task error to understand how the cumulative error changes as system parameters evolve. The parameter update rule is a noisy version of the gradient descent update:

$$\Delta x_t := -\nabla_x f_t(x_t)+b\overrightarrow{\beta}+\sigma\sqrt{h(d)}\hat{d}. \tag{7}$$

For a small step η, the Taylor series expansion is given as:

$$f(x_{t+1})=f(x_t)+\langle\nabla_x f(x_t),\Delta x_t\rangle\eta+\frac{1}{2}\left\langle\Delta x_t,\nabla_x^2 f(x_t)\Delta x_t\right\rangle\eta^2, \tag{8}$$

Using this approach, let us compare the change in cumulative error after one decision update to the cumulative error using the standard online gradient descent method for decision update,

$$\Delta f\left(b,\sigma^2\right)=\left[f_{t+1}\left(x_{t+1}\right)+f_t(x_t)\right]_{(b,\sigma^2)}-\left[f_{t+1}\left(x_{t+1}\right)+f_t(x_t)\right]_{(0,0)}, \tag{9}$$

Plugging in Equations **??** and **??** in Equation **??**,

$$\begin{aligned}\Delta\mathcal{L}\left(b,\sigma^2\right) &= \left[b\left\langle\nabla_x f_{t+1}(x_t),\overrightarrow{\beta}\right\rangle+\sigma\sqrt{h(d)}\left\langle\nabla_x f_{t+1}(x_t),\hat{d}\right\rangle\right]\eta \\ &+\frac{1}{2}\left[-b\left\langle\nabla_x f_t(x_t),\nabla_x^2 f_{t+1}(x_t)\overrightarrow{\beta}\right\rangle-\sigma\sqrt{h(d)}\left\langle\nabla_x f_t(x_t),\nabla_x^2 f_{t+1}(x_t)\hat{d}\right\rangle\right. \\ &-b\left\langle\overrightarrow{\beta},\nabla_x^2 f_{t+1}(x_t)\nabla_x f_t(x_t)\right\rangle-\sigma\sqrt{h(d)}\left\langle\hat{d},\nabla_x^2 f_{t+1}(x_t)\nabla_x f_t(x_t)\right\rangle+b\sigma\sqrt{h(d)}\left\langle\overrightarrow{\beta},\nabla_x^2 f_{t+1}(x_t)\hat{d}\right\rangle \\ &\left.+b\sigma\sqrt{h(d)}\left\langle\hat{d},\nabla_x^2 f_{t+1}(x_t)\overrightarrow{\beta}\right\rangle+b^2\left\langle\overrightarrow{\beta},\nabla_x^2 f_{t+1}(x_t)\overrightarrow{\beta}\right\rangle+\sigma^2 h(d)\left\langle\hat{d},\nabla_x^2 f_{t+1}(x_t)\hat{d}\right\rangle\right]\eta^2\end{aligned}$$

Taking the expectation:

$$\begin{aligned}\mathbb{E}_{\hat{d}}\left[\Delta f\left(b,\sigma^2\right)\right] &= \left[b\left\langle\nabla_x f_{t+1}(x_t),\overrightarrow{\beta}\right\rangle\right]\eta+\frac{1}{2}\left[-b\left\langle\nabla_x f_t(x_t),\nabla_x^2 f_{t+1}(x_t)\overrightarrow{\beta}\right\rangle\right. \\ &-b\left\langle\overrightarrow{\beta},\nabla_x^2 f_{t+1}(x_t)\nabla_x f_t(x_t)\right\rangle-\sigma\sqrt{h(d)}\left\langle\hat{d},\nabla_x^2 f_{t+1}(x_t)\nabla_x f_t(x_t)\right\rangle \\ &\left.+b^2\left\langle\overrightarrow{\beta},\nabla_x^2 f_{t+1}(x_t)\overrightarrow{\beta}\right\rangle+\sigma^2 h(d)\left\langle\hat{d},\nabla_x^2 f_{t+1}(x_t)\hat{d}\right\rangle\right]\eta^2\end{aligned} \tag{10}$$

Additionally, we make the following assumptions for elements of the d-dimensional noise vector $\hat{d}$: $\mathbb{E}_{\hat{d}}\left[\hat{d}_i^2\right]=\frac{1}{d}$ and $\mathbb{E}_{\hat{d}}\left[\hat{d}_i\hat{d}_j\right]=0,\forall i\neq j\in 1\ldots d$. Therefore,

$$\begin{aligned}\mathbb{E}_{\hat{d}}\left[\left\langle\hat{d},\nabla_x^2 f_{t+1}(x_t)\hat{d}\right\rangle\right] &= \mathbb{E}_{\hat{d}}\left[\sum_{i,j}\hat{d}_i\hat{d}_j\nabla_x^2 f_{t+1}(x_t)_{ij}\right] \\ &= \sum_i\mathbb{E}_{\hat{d}}\left[\hat{d}_i^2\right]\nabla_x^2 f_{t+1}(x_t)_{ii}+\sum_{i\neq j}\sum_j\mathbb{E}_{\hat{d}}\left[\hat{d}_i\hat{d}_j\right]\nabla_x^2 f_{t+1}(x_t)_{ij} \\ &= \frac{1}{d}\sum_i\nabla_x^2 f_{t+1}(x_t)_{ii}=\frac{1}{d}\operatorname{Tr}\left[\nabla_x^2 f_{t+1}(x_t)\right],\end{aligned} \tag{11}$$

Moreover:

$$\left\langle \nabla_x f_t(x_t), \nabla_x^2 f_{t+1}(x_t)\overrightarrow{\boldsymbol{\beta}} \right\rangle + \left\langle \overrightarrow{\boldsymbol{\beta}}, \nabla_x^2 f_{t+1}(x_t)\nabla_x f_t(x_t) \right\rangle = \nabla_x f_t(x_t)^T \nabla_x^2 f_{t+1}(x_t)\overrightarrow{\boldsymbol{\beta}} + \overrightarrow{\boldsymbol{\beta}}^T \nabla_x^2 f_{t+1}(x_t)\nabla_x f_t(x_t)$$
$$= \nabla_x f_t(x_t)^T \nabla_x^2 f_{t+1}(x_t)\overrightarrow{\boldsymbol{\beta}} + \nabla_x f_t(x_t)^T \nabla_x^2 f_{t+1}(x_t)^T \overrightarrow{\boldsymbol{\beta}}$$
$$= \nabla_x f_t(x_t)^T \left(\nabla_x^2 f_{t+1}(x_t) + \nabla_x^2 f_{t+1}(x_t)^T\right)\overrightarrow{\boldsymbol{\beta}}$$
$$= \left\langle \nabla_x f_t(x_t), \left(\nabla_x^2 f_{t+1}(x_t) + \nabla_x^2 f_{t+1}(x_t)^T\right)\overrightarrow{\boldsymbol{\beta}} \right\rangle \tag{12}$$

The second equality holds because each item is a scalar value. Now, incorporating Equations **??** and **??** to simplify Equation **??**:

$$\mathbb{E}_{\hat{d}}\left[\Delta f\left(b,\sigma^2\right)\right] = \mathbb{E}_{\hat{d}}\left[\left[f_{t+1}\left(x_{t+1}\right) + f_t(x_t)\right]_{(b,\sigma^2)} - \left[f_{t+1}\left(x_{t+1}\right) + f_t(x_t)\right]_{(0,0)}\right],$$
$$= b\left\langle \nabla_x f_{t+1}(x_t), \overrightarrow{\boldsymbol{\beta}} \right\rangle \eta + \frac{1}{2}b^2\left\langle \overrightarrow{\boldsymbol{\beta}}, \nabla_x^2 f_{t+1}(x_t)\overrightarrow{\boldsymbol{\beta}} \right\rangle \eta^2 - \frac{1}{2}b\left\langle \nabla_x f_t(x_t), \left(\nabla_x^2 f_{t+1}(x_t) + \nabla_x^2 f_{t+1}(x_t)^T\right)\overrightarrow{\boldsymbol{\beta}} \right\rangle \eta^2$$
$$+ \frac{1}{2}\frac{\sigma^2 h(d)}{d}\operatorname{Tr}\left[\nabla_x^2 f_{t+1}(x_t)\right]\eta^2.$$

■

Lemma 7.2: Consider the online gradient descent algorithm. Define the dynamic regret at time t with variance σ^2 as $\mathrm{DR}(t)_{(0,\sigma^2)} = \left[f_t\left(x_t\right) - f_t\left(x_t^*\right)\right]_{(0,\sigma^2)}$. The regret is growing at a constant rate ($\mathrm{DR}(t+1)_{(0,\sigma^2)} - \mathrm{DR}(t)_{(0,\sigma^2)} \geq C_1$) if the variance satisfied:

$$\sigma^2 \geq \frac{2C_1}{\lambda_{\min}\eta^2} + \frac{2\left(f_t(x_t) - f_t^*\right) - \left(f_{t+1}(x_t) - f_{t+1}^*\right)}{\lambda_{\min}\eta^2} + \frac{2\nabla f_{t+1}^\top\left(x_t\right)\nabla f_t\left(x_t\right)}{\lambda_{\min}\eta} - \frac{\nabla f_t^\top\left(x_t\right)\left(\nabla^2 f_{t+1}\left(x_t\right)\right)\left(\nabla f_t\left(x_t\right)\right)}{\lambda_{\min}},$$

B. *Proof of Lemma* **??**

Proof:

$$DR(t+1)_{(0,\sigma^2)} - DR(t)_{(0,\sigma^2)} = \left(f_{t+1}(x_t) - f_{t+1}^*\right) - \left(f_t(x_t) - f_t^*\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle\eta$$
$$+ \frac{1}{2}\left\langle \nabla f_t(x_t), \nabla^2 f_{t+1}(x_t)\left(\nabla f_t(x_t)\right)\right\rangle\eta^2 + \frac{1}{2}\frac{\sigma^2 h(d)}{d}\operatorname{Tr}\left[\nabla^2 f_{t+1}\right]\eta^2,$$
$$\geq \left(f_{t+1}(x_t) - f_{t+1}^*\right) - \left(f_t(x_t) - f_t^*\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle\eta$$
$$+ \frac{1}{2}\left\langle \nabla f_t(x_t), \nabla^2 f_{t+1}(x_t)\left(\nabla f_t(x_t)\right)\right\rangle\eta^2 + \frac{1}{2}\sigma^2 h(d)\lambda_{\min}\eta^2,$$

The inequality holds as $\lambda_{\min} \leq \frac{1}{d}\operatorname{Tr}\left[\nabla_x^2 f_{t+1}\right] \leq \lambda_{\max}$. If the RHS of inequality is greater than $C_1 > 0$, then the decision update step necessarily implies $DR(t+1)_{(0,\sigma^2)} - DR(t)_{(0,\sigma^2)} \geq C_1$, and let $h(d) = 1$ Therefore,

$$\left(f_{t+1}(x_t) - f_{t+1}^*\right) - \left(f_t(x_t) - f_t^*\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle\eta + \frac{1}{2}\left\langle \nabla f_t(x_t), \nabla^2 f_{t+1}(x_t)\left(\nabla f_t(x_t)\right)\right\rangle\eta^2 + \frac{1}{2}\sigma^2 h(d)\lambda_{\min}\eta^2 \geq C_1,$$

$$\sigma^2 \geq \frac{2C_1}{\lambda_{\min}\eta^2} + \frac{2\left(f_t(x_t) - f_t^*\right) - \left(f_{t+1}(x_t) - f_{t+1}^*\right)}{\lambda_{\min}\eta^2} + \frac{2\nabla f_{t+1}^\top\left(x_t\right)\nabla f_t\left(x_t\right)}{\lambda_{\min}\eta} - \frac{\nabla f_t^\top\left(x_t\right)\left(\nabla^2 f_{t+1}\left(x_t\right)\right)\left(\nabla f_t\left(x_t\right)\right)}{\lambda_{\min}},$$

■

Lemma 7.3: Consider the online gradient descent algorithm. Define the dynamic regret at time t with biased b in the gradient as $\mathrm{DR}(t)_{(b,0)} = \left[f_t\left(x_t\right) - f_t\left(x_t^*\right)\right]_{(b,0)}$. The regret is growing at a constant rate ($\mathrm{DR}(t+1)_{(b,0)} - \mathrm{DR}(t)_{(b,0)} \geq C_2$), if and only if the bias satisfied:

$$b \geq \frac{C_2 + \left(f_t(x_t) - f_t^*\right) - \left(f_{t+1}(x_t) - f_{t+1}^*\right) + \nabla f_{t+1}^\top\left(x_t\right)\nabla f_t\left(x_t\right)\eta}{\sqrt{d}\left\|\nabla f_{t+1}\right\|\eta}$$

C. *Proof of Lemma* **??**

D. *First order approximation:*

Proof: For simplicity, we consider first order Taylor approximation $f_{t+1}(x_{t+1})$ as the following:

$$f_{t+1}(x_{t+1}) = f_{t+1}(x_t) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle\eta + \left\langle \nabla f_{t+1}(x_t), b\overrightarrow{\boldsymbol{\beta}}\right\rangle\eta,$$

Then,

$$DR(t+1)_{(b,0)} - DR(t)_{(b,0)} = \left(f_{t+1}(x_t) - f^*_{t+1}\right) - \left(f_t(x_t) - f^*_t\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle \eta + \left\langle \nabla f_{t+1}(x_t), b\overrightarrow{\boldsymbol{\beta}}\right\rangle \eta,$$

If the RHS of inequality is greater than $C_2 > 0$, then the decision update step necessarily implies $DR(t+1)_{(b,0)} - DR(t)_{(b,0)} \geq C_2$, then

$$\begin{aligned}
&\left(f_{t+1}(x_t) - f^*_{t+1}\right) - \left(f_t(x_t) - f^*_t\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle \eta + \left\langle \nabla f_{t+1}(x_t), b\overrightarrow{\boldsymbol{\beta}}\right\rangle \eta \geq C_2, \\
&\left\langle \nabla f_{t+1}(x_t), b\overrightarrow{\boldsymbol{\beta}}\right\rangle \eta \geq C_2 + \left(f_t(x_t) - f^*_t\right) - \left(f_{t+1}(x_t) - f^*_{t+1}\right) + \nabla f^\top_{t+1}(x_t)\nabla f_t(x_t)\,\eta, \\
&b\,\sqrt{d}\,\|\nabla f_{t+1}\|\,\eta \geq C_2 + \left(f_t(x_t) - f^*_t\right) - \left(f_{t+1}(x_t) - f^*_{t+1}\right) + \nabla f^\top_{t+1}(x_t)\nabla f_t(x_t)\,\eta, \\
&\qquad b \geq \frac{C_2 + \left(f_t(x_t) - f^*_t\right) - \left(f_{t+1}(x_t) - f^*_{t+1}\right) + \nabla f^\top_{t+1}(x_t)\nabla f_t(x_t)\,\eta}{\sqrt{d}\,\|\nabla f_{t+1}\|\,\eta}.
\end{aligned}$$

E. Second order approximation:

Now, let us analyze the second-order Taylor approximation :

$$\begin{aligned}
DR(t+1)_{(b,0)} - DR(t)_{(b,0)} &= \left(f_{t+1}(x_t) - f^*_{t+1}\right) - \left(f_t(x_t) - f^*_t\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle \eta + \left\langle \nabla f_{t+1}(x_t), b\overrightarrow{\boldsymbol{\beta}}\right\rangle \eta, \\
&\quad + \frac{1}{2}\left\langle \nabla f_t(x_t), \nabla^2 f_{t+1}(x_t)\left(\nabla f_t(x_t)\right)\right\rangle \eta^2 - \frac{1}{2}\left\langle b\overrightarrow{\boldsymbol{\beta}}, \nabla^2 f_{t+1}(x_t)\left(\nabla f_t(x_t)\right)\right\rangle \eta^2 \\
&\quad - \frac{1}{2}\left\langle \nabla f_t(x_t), \nabla^2 f_{t+1}(x_t)(b\overrightarrow{\boldsymbol{\beta}})\right\rangle \eta^2 + \frac{1}{2}\left\langle b\overrightarrow{\boldsymbol{\beta}}, \nabla^2 f_{t+1}(x_t)(b\overrightarrow{\boldsymbol{\beta}})\right\rangle \eta^2, \\
&\overset{(i)}{\leq} \left(f_{t+1}(x_t) - f^*_{t+1}\right) - \left(f_t(x_t) - f^*_t\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle \eta + \left\langle \nabla f_{t+1}(x_t), b\overrightarrow{\boldsymbol{\beta}}\right\rangle \eta \\
&\quad + \frac{1}{2}\lambda_{\max}\|\nabla f_t(x_t)\|^2\eta^2 - \frac{1}{2}\left\langle b\overrightarrow{\boldsymbol{\beta}}, \nabla^2 f_{t+1}(x_t)\left(\nabla f_t(x_t)\right)\right\rangle \eta^2 \\
&\quad - \frac{1}{2}\left\langle \nabla f_t(x_t), \nabla^2 f_{t+1}(x_t)(b\overrightarrow{\boldsymbol{\beta}})\right\rangle \eta^2 + \frac{1}{2}\lambda_{\max}\|b\sqrt{d}\|^2\eta^2, \\
&= \left(f_{t+1}(x_t) - f^*_{t+1}\right) - \left(f_t(x_t) - f^*_t\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle \eta + \frac{\eta^2}{2}\lambda_{\max}\|\nabla f_t(x_t)\|^2 \\
&\quad + \frac{\eta^2}{2}\lambda_{\max}\|b\sqrt{d}\|^2 + \left\langle \eta\nabla f_{t+1}(x_t) - \frac{\eta^2}{2}\nabla f_t(x_t)\left(\nabla^2 f_{t+1}(x_t) + \nabla^2 f_{t+1}(x_t)^T\right), b\vec{\beta}\right\rangle, \\
&\overset{(ii)}{\leq} \left(f_{t+1}(x_t) - f^*_{t+1}\right) - \left(f_t(x_t) - f^*_t\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle \eta + \frac{\eta^2}{2}\lambda_{\max}\|\nabla f_t(x_t)\|^2 \\
&\quad + \|\nabla f_{t+1}(x_t) - \eta\left(\nabla^2 f_{t+1}(x_t)\right)\nabla f_t(x_t)\|\|b\sqrt{d}\eta\| + \frac{1}{2}\lambda_{\max}\|b\sqrt{d}\eta\|^2,
\end{aligned}$$

where (i) follows from a linear algebra inequality that $\lambda_{\min}\|\mathcal{C}\|^2 \leq \left\langle \mathcal{C}, \nabla^2 f_{t+1}(x_t)\mathcal{C}\right\rangle \leq \lambda_{\max}\|\mathcal{C}\|^2$ for vector $\mathcal{C}$ and $\lambda_{\max} \geq \ldots \geq \lambda_{\min}$ denotes the eigenvalues of $\nabla^2 f_{t+1}$. Finally, (ii) follows from Cauchy-Schwarz inequality. If the RHS of inequality is less than $-C_2$, then the decision update step sufficiently implies a decrease in the dynamic regret by, at most, a constant C_2, then

$$\alpha\|b\sqrt{d}\eta\|^2 + \beta\|b\sqrt{d}\eta\| + \gamma \leq 0,$$

where $\alpha = \frac{1}{2}\lambda_{\max}$, $\beta = \|\nabla f_{t+1}(x_t) - \eta\left(\nabla^2 f_{t+1}(x_t)\right)\nabla f_t(x_t)\|$, and $\gamma = \left(f_{t+1}(x_t) - f^*_{t+1}\right) - \left(f_t(x_t) - f^*_t\right) + \left\langle \nabla f_{t+1}(x_t), -\nabla f_t(x_t)\right\rangle \eta + \frac{\eta^2}{2}\lambda_{\max}\|\nabla f_t(x_t)\|^2 + C_2$. Since $\|b\sqrt{d}\eta\|$ cannot be negative and to get intuition, let us assume f_{t+1} being smooth and $c < 0$, then the allowed value of bias is in $[0, \frac{-\beta+\sqrt{\beta^2-4\alpha\gamma}}{2\alpha\eta\sqrt{d}})$. This implies that as the dimensionality of our problem increases, the amount of bias the algorithm can tolerate decreases which is consistent with the first-order expansion condition.

■

Lemma 7.4: The sequence $\{x_t\}$ for all $t \in \mathbb{N}$ generated by IOGD algorithm with a step size $\eta \leq \frac{1}{L}$ for $L-$smooth and $\mu-$strongly convex $f(\cdot)$ satisfies the following bounds for any sequence $\{x_t^*\}$

$$\mathbb{E}_{\hat{d}}\|z_t^{j+1} - x_t^*\|^2 \leq \left(1 - \frac{h\mu}{L}\right)\|z_t^j - x_t^*\|^2 + \frac{h}{L}\epsilon_t,$$
$$\|x_{t+1} - x_t^*\|^2 \leq \beta^K\|x_t - x_t^*\|^2 + \frac{1}{\mu}\epsilon_t,$$
$$\sum_{t=1}^{T}\|x_t - x_t^*\|^2 \leq \frac{1}{1-2\beta^K}\left(\|x_1 - x_1^*\|^2 + 2\mathcal{E}_T + 2\mathcal{S}_T^*\right),$$

where $\beta = 1 - \frac{h\mu}{L}$, $\epsilon_t = 2D\left(b_t\sqrt{d} + \sigma_t\right)$, and $\mathcal{E}_T = \frac{1}{\mu}\sum_{t=2}^{T}\epsilon_t$.

F. *Proof of Lemma* **??**

Proof: Strong convexity of the function f_t implies that

$$f_t(x) - \frac{\mu}{2}\|x - z_t^j\|^2 \geq f_t\left(z_t^j\right) + \nabla f_t\left(z_t^j\right)^T\left(x - z_t^j\right), \tag{13}$$

for any $x \in \mathcal{X}$. By adding and subtracting the inner product $\nabla f_t\left(z_t^j\right)^T\left(\hat{z}_t^j - z_t^j\right)$ to the right-hand side of **??**, we obtain

$$\begin{aligned} f_t(x) - \frac{\mu}{2}\|x - z_t^j\|^2 &\geq f_t\left(z_t^j\right) + \nabla f_t\left(z_t^j\right)^T\left(\hat{z}_t^j - z_t^j\right) + \nabla f_t\left(z_t^j\right)^T\left(x - \hat{z}_t^j\right) \\ &= f_t\left(z_t^j\right) + \nabla f_t\left(z_t^j\right)^T\left(\hat{z}_t^j - z_t^j\right) + \hat{\nabla} f_t\left(z_t^j, \xi\right)^T\left(x - \hat{z}_t^j\right) + \left(\nabla f_t\left(z_t^j\right) - \hat{\nabla} f_t\left(z_t^j, \xi\right)\right)^T\left(x - \hat{z}_t^j\right) \end{aligned} \tag{14}$$

Observe that the optimality condition of the update of $\hat{z}_t^j$ in line 7 of Algorithm 1 implies that

$$\left(\hat{\nabla} f_t\left(z_t^j, \xi\right) + \frac{1}{\eta}\left(\hat{z}_i^j - z_i^j\right)\right)^T\left(x - \hat{z}_t^j\right) \geq 0, \tag{15}$$

for any $x \in \mathcal{X}$. From the result in **??**, it follows that the inner product $\hat{\nabla} f_t\left(z_t^j, \xi\right)^T\left(x - \hat{z}_t^j\right)$ is bounded below by $\frac{1}{\eta}\left(z_t^j - \hat{z}_i^j\right)^T\left(x - \hat{z}_t^j\right)$. Applying this substitution to **??** yields

$$f_t(x) - \frac{\mu}{2}\|x - z_i^j\|^2 \geq f_t\left(z_i^j\right) + \nabla f_t\left(x_i^j\right)^T\left(z_t^j - z_t^j\right) + \frac{1}{\eta}\left(z_t^j - z_i^j\right)^T\left(x - z_i^j\right) + \left(\nabla f_t\left(z_i^j\right) - \hat{\nabla} f_t\left(z_i^j, \xi\right)\right)^T\left(x - z_t^j\right). \tag{16}$$

According to the Lipschitz continuity of the instantaneous gradients ∇f_t and Taylor's series of the objective function $f_t\left(\hat{z}_t^j\right)$ near the point z_t^j we can write

$$f_t\left(\hat{z}_i^j\right) \leq f_t\left(z_i^j\right) + \nabla f_t\left(z_i^j\right)^T\left(z_i^j - z_t^j\right) + \frac{L}{2}\|\hat{z}_i^j - z_i^j\|^2.$$

Thus, the sum $f_t\left(z_t^j\right) + \nabla f_t\left(z_t^j\right)^T\left(\hat{z}_t^j - z_t^j\right)$ is bounded below by $f_t\left(\hat{z}_t^j\right) - \frac{L}{2}\|\hat{z}_t^j - z_t^j\|^2$. By applying this substitution into **??**, we obtain

$$f_t(x) - \frac{\mu}{2}\|x - z_i^j\|^2 \geq f_t\left(\hat{z}_t^j\right) - \frac{L}{2}\|\hat{z}_i^j - z_t^j\|^2 + \frac{1}{\eta}\left(z_t^j - \hat{z}_t^j\right)^T\left(x - \hat{z}_t^j\right) + \left(\nabla f_t\left(z_i^j\right) - \hat{\nabla} f_t\left(z_t^j, \xi\right)\right)^T\left(x - \hat{z}_t^j\right). \tag{17}$$

By adding and subtracting z_t^j we can expand the inner product $\left(z_t^j - \hat{z}_t^j\right)^T\left(x - \hat{z}_t^j\right)$ as the sum of $\left(z_t^j - \hat{z}_t^j\right)^T\left(x - z_t^j\right)$ and $\left(z_t^j - \hat{z}_t^j\right)^T\left(z_t^j - \hat{z}_t^j\right)$. From applying this substitution into **??** and $\eta \leq \frac{1}{L}$ it follows that

$$f_t(x) - \frac{\mu}{2}\|x - z_t^j\|^2 \geq f_t\left(\hat{z}_t^j\right) + \frac{L}{2}\|\hat{z}_t^j - z_t^j\|^2 + \frac{1}{\eta}\left(z_t^j - \hat{z}_t^j\right)^T\left(x - z_t^j\right) + \left(\nabla f_t\left(z_t^j\right) - \hat{\nabla} f_t\left(z_t^j, \xi\right)\right)^T\left(x - \hat{z}_t^j\right). \tag{18}$$

Now set $x = x_t^*$ in **??** and regroup the terms to obtain

$$f_t\left(x_t^*\right) - f_t\left(\hat{z}_t^j\right) \geq \frac{\mu}{2}\|x_t^* - z_t^j\|^2 + \frac{L}{2}\|\hat{z}_t^j - z_t^j\|^2 + \frac{1}{\eta}\left(z_t^j - \hat{z}_t^j\right)^T\left(x_t^* - z_t^j\right) + \left(\nabla f_t\left(z_t^j\right) - \hat{\nabla} f_t\left(z_t^j, \xi\right)\right)^T\left(x - \hat{z}_t^j\right). \tag{19}$$

Note that the optimal objective function value $f_t\left(x_t^*\right)$ is smaller than $f_t\left(\hat{z}_t^j\right)$. Thus, the left-hand side of **??** is nonpositive, which implies that the right-hand side is also smaller than 0 :

$$\frac{\mu}{2}\|x_t^* - z_t^j\|^2 + \frac{L}{2}\|\hat{z}_t^j - z_t^j\|^2 + \frac{1}{\eta}\left(z_t^j - \hat{z}_t^j\right)^T\left(x_t^* - z_t^j\right) + \left(\nabla f_t\left(z_t^j\right) - \hat{\nabla} f_t\left(z_t^j, \xi\right)\right)^T\left(x - \hat{z}_t^j\right) \leq 0.$$

Therefore, by dividing both sides of the inequality by L and regrouping the terms, it follows that

$$\left(z_t^j - \hat{z}_t^j\right)^T\left(z_t^j - x_t^*\right) \geq \frac{1}{2}\|\hat{z}_t^j - z_t^j\|^2 + \frac{\mu}{2L}\|x_t^* - z_t^j\|^2 + \frac{1}{L}\left(\nabla f_t\left(z_t^j\right) - \hat{\nabla} f_t\left(z_t^j, \xi\right)\right)^T\left(x - \hat{z}_t^j\right). \tag{20}$$

Now by showing a lower bound for the inner product $\left(z_t^j - \hat{z}_t^j\right)^T\left(z_t^j - x_t^*\right)$ in terms of the squared norms $\|\hat{z}_t^j - z_t^j\|^2$ and $\|x_t^* - z_t^j\|^2$ we proceed to prove the main claim. Consider the update $z_t^{j+1} = (1-h)z_t^j + h\hat{z}_t^j$ in line 8 of Algorithm **??**. By subtracting x_t^* from both sides of the equality and computing the squared norm of the resulted expression, we obtain

$$\|z_t^{j+1} - x_t^*\|^2 = \|z_t^j - x_t^*\|^2 + h^2\|z_t^j - \hat{z}_t^j\|^2 - h\left(z_t^j - x_t^*\right)^T\left(z_t^j - \hat{z}_t^j\right). \tag{21}$$

Substitute the inner product $\left(z_t^j - x_t^*\right)^T\left(z_t^j - \hat{z}_t^j\right)$ in **??** by its lower bound in **??** to obtain

$$\|z_t^{j+1} - x_t^*\|^2 \leq \left(1 - \frac{h\mu}{L}\right)\|z_t^j - x_t^*\|^2 + h(h-1)\|z_t^j - \hat{z}_t^j\|^2 - \frac{2h}{L}\left(\nabla f_t\left(z_t^j\right) - \hat{\nabla} f_t\left(z_t^j, \xi\right)\right)^T\left(x - \hat{z}_t^j\right). \tag{22}$$

First, note that h is a constant from the interval $(0,1]$; therefore, the term $h(h-1)\|z_t^j - \hat{z}_t^j\|^2$ is smaller than zero, and we can simplify the right-hand side of **??** as

$$\begin{aligned}
\|z_t^{j+1} - x_t^*\|^2 &\leq \left(1 - \frac{h\mu}{L}\right)\|z_t^j - x_t^*\|^2 - \frac{2h}{L}\left(\nabla f_t\left(z_t^j\right) - \hat{\nabla} f_t\left(z_t^j, \xi\right)\right)^T\left(x - \hat{z}_t^j\right), \\
&\leq \left(1 - \frac{h\mu}{L}\right)\|z_t^j - x_t^*\|^2 + \frac{2h}{L}\left(b\overrightarrow{\boldsymbol{\beta}} + \sigma\sqrt{h(d)}\hat{d}\right)^\top\left(x - \hat{z}_t^j\right), \\
&\leq \left(1 - \frac{h\mu}{L}\right)\|z_t^j - x_t^*\|^2 + \frac{2h}{L}\|b\overrightarrow{\boldsymbol{\beta}} + \sigma\sqrt{h(d)}\hat{d}\|\|x - \hat{z}_t^j\|, \\
&\leq \left(1 - \frac{h\mu}{L}\right)\|z_t^j - x_t^*\|^2 + \frac{2hD}{L}\left(\|b\overrightarrow{\boldsymbol{\beta}}\| + \|\sigma\sqrt{h(d)}\|\right),
\end{aligned} \tag{23}$$

Taking the expectation over the randomness on both sides of **??**, we have

$$\mathbb{E}_{\hat{d}}\|z_t^{j+1} - \mathrm{x}_t^*\|^2 \leq \left(1 - \frac{h\mu}{L}\right)\|z_t^j - x_t^*\|^2 + \frac{2hD}{L}\left(b\sqrt{d} + \sigma\sqrt{h(d)}\right), \tag{24}$$

Case(1): summation bound for $\mathcal{S}_T^*$: Note that the strong convexity constant μ is smaller than the constant of gradients Lipschitz continuity L, i.e., $\mu \leq L$. Moreover, $h \leq 1$ which implies that $0 < \frac{h\mu}{L} \leq 1$. Hence, the constant $\beta := (1 - \frac{h\mu}{L})$ is from the interval $[0,1)$. Recall the updating rule $z_{t-1}^{j+1} = P_X\left(z_{t-1}^j - \alpha\hat{\nabla} f_{t-1}\left(z_{t-1}^j\right)\right), j = 1, \ldots, K$; then, we can write that

$$\begin{aligned}
\|x_{t+1} - x_t^*\|^2 &= \left\|z_t^{K+1} - x_t^*\right\|^2 \\
&\leq \beta^K\|x_t - x_t^*\|^2 + \frac{2hD\left(1 - \beta^K\right)}{L(1-\beta)}\left(b\sqrt{d} + \sigma\sqrt{h(d)}\right), \\
&\leq \beta^K\|x_t - x_t^*\|^2 + \frac{2h}{\mu}\left(b\sqrt{d} + \sigma\sqrt{h(d)}\right),
\end{aligned}$$

where we recursively apply the result from Equation **??**. Let $\epsilon_t = \frac{2Db\sqrt{d}}{\mu}$. Now, using $\|x-y\|^2 \leq (1+\iota)\|x-z\|^2 + \left(1+\frac{1}{\iota}\right)\|z-y\|^2$, we can bound

$$\sum_{t=1}^{T}\|x_t - x_t^*\|^2 \leq \|x_1 - x_1^*\|^2 + \sum_{t=2}^{T}(1+\iota)\left\|x_t - x_{t-1}^*\right\|^2 + \sum_{t=2}^{T}\left(1+\frac{1}{\iota}\right)\left\|x_t^* - x_{t-1}^*\right\|^2.$$

$$\leq \|x_1 - x_1^*\|^2 + \sum_{t=1}^{T}(1+\iota)\beta^K\|x_t - x_t^*\|^2 + \sum_{t=1}^{T}(1+\iota)\epsilon_t + \sum_{t=2}^{T}\left(1+\frac{1}{\iota}\right)\left\|x_t^* - x_{t-1}^*\right\|^2.$$

$$\sum_{t=1}^{T}\|x_t - x_t^*\|^2 \leq \frac{\|x_1 - x_1^*\|^2}{(1-(1+\iota)\beta^K)} + \sum_{t=2}^{T}\frac{(1+\iota)\epsilon_t}{(1-(1+\iota)\beta^K)} + \frac{\left(1+\frac{1}{\iota}\right)\mathcal{S}_t^*}{(1-(1+\iota)\beta^K)}. \tag{25}$$

Then, by setting $\iota = 1$

$$\sum_{t=1}^{T}\|x_t - x_t^*\|^2 \leq \frac{1}{1-2\beta^K}\left(\|x_1 - x_1^*\|^2 + 2\mathcal{E}_T^* + 2\mathcal{S}_T^*\right). \tag{26}$$

Recap that $\beta = (1-\frac{h\mu}{L})$, then by choosing $K \propto \frac{L}{h\mu}$.i.e., $K = \lceil \frac{\ln(4)L}{h\mu} \rceil$, then

$$\sum_{t=1}^{T}\|x_t - x_t^*\|^2 \leq 2\|x_1 - x_1^*\|^2 + 4\mathcal{E}_T^* + 4\mathcal{S}_T^*. \tag{27}$$

Note that $\kappa = \frac{L}{\mu}$ is the conditional number. If κ is large, we have ill-condition. It is known that it is more difficult to optimize, and that is why we need more iterations as $K \propto \frac{L}{\mu}$.

Case(2): summation bound for $\mathcal{P}_T^*$:

$$\sum_{t=1}^{T}\|x_t - x_t^*\| \leq \frac{\|x_1 - x_1^*\| - \beta^{\frac{K}{2}}\|x_T - x_T^*\|}{1-\beta^{\frac{K}{2}}} + \frac{1}{1-\beta^{\frac{K}{2}}}\sum_{t=2}^{T}\left\|x_t^* - x_{t-1}^*\right\| + \frac{1}{1-\beta^{\frac{K}{2}}}\sqrt{\mathcal{E}_T}$$

$$= \frac{\|x_1 - x_1^*\| - \beta^{\frac{K}{2}}\|x_T - x_T^*\|}{1-\beta^{\frac{K}{2}}} + \frac{\mathcal{P}_T^*}{1-\beta^{\frac{K}{2}}} + \frac{1}{1-\beta^{\frac{K}{2}}}\sqrt{\mathcal{E}_T}$$

■