We thank the reviewers for their constructive comments. We will incorporate the discussions below in the revision.

R1-Q1 & R3-Q4: technical challenges and novelty. We apologize for the confusion. One of the main challenges we encountered was *how to model/represent the (multi-step-ahead) prediction errors*. The model should be general enough to include various prediction methods, yet clean enough to allow for insightful theoretical analysis. After many attempts, we eventually decide to represent prediction errors by how many steps ahead this prediction is made. This representation is general and can include most existing models in literature [1,15,25]. Further, this representation clearly suggests how each prediction errors affect the algorithm updates (see e.g. (6)), providing insightful understanding.

Next, we discuss the proof novelty. Similar to RHGD in [12] and [14], we design our RHIG in such a way that the online regret analysis can be converted to offline gradient-based algorithms' convergence analysis. However, our RHIG is converted to offline *inexact/stochastic* gradient descent, where the gradient errors are from prediction errors; while [12,14] only deal with the *exact* gradient, which is much simpler to analyze. To handle the prediction/gradient errors, we first establish the almost-strong-convexity and almost-smoothness inequalities in Lemma 4 & 5 as mentioned by R3. In addition, we conduct nontrivial manipulation of the inequalities and carefully choose the stepsizes and parameters so that the accumulated errors do not explode. Our procedures are different from the classic literature since our gradient errors are time-varying, not uniformly bounded, and correlated, while most literature assumes uniformly bounded and/or i.i.d. gradient errors. Moreover, we provide a concentration bound for the regret, which is not addressed in [12,14].

Finally, we note that existing papers on SOCO with noisy multi-step predictions (e.g. [1,15]) only provide regret bounds for *optimization-based* algorithms, but our paper studies a *gradient-based* algorithm, which is more applicable under computational time constraints, e.g. in robotics, connected vehicles and other large transportation systems, etc.

R1-Q2 & R2-Q2: algorithm comparison and intuitions on how we deal with noisy predictions. Good question. Firstly, most existing methods, e.g. AFHC, CHC, are optimization-based, while our RHIG is based on gradient descent, which is known to be more robust to errors. Secondly, RHIG implicitly reduces the impact of the (poorer-quality) long-term predictions and focuses more on the (better) short-term ones by using long-term predictions in the first several updates and then using short-term ones in later updates to refine the decisions; while AFHC and CHC treat predictions more equally by taking averages of the optimal solutions computed by both long-term and short-term predictions.

R2-Q1: regret bound explanation. We apologize for the confusion. Let $E_k = \|\boldsymbol{\delta}(k)\|^2$ denote the k-step-ahead prediction error, then the regret bound in Corollary 1 is $\tilde{O}(\rho^W \sqrt{V_T T} + \sum_{k=1}^{\min(W,T)} \rho^k E_k)$. The first term decays exponentially with W, suggesting that RHIG reduces the impact of the variation V_T exponentially with W. In the second term, E_k's coefficient ρ^k decays exponentially with k, suggesting that *the impact of the k-step-ahead prediction error decays exponentially with k*. This benefits the performance since long-term predictions are usually much poorer than short-term ones. **R2-Q3: proof explanation.** Please kindly refer to the second paragraph of R1-Q1 & R3-Q4.

R3-Q1: parametric costs. Good question. Section 4 (the deterministic case) can be generalized to varying function types as in [14]. However, the tricky problem is *how to <u>model</u> the stochastic prediction errors*. There are usually three types of correlation between the prediction errors, i.e. between the errors of (i) $\nabla f_{t|\tau}(x_1)$ and $\nabla f_{t|\tau}(x_2)$, (ii) $\nabla f_{t|\tau_1}(x)$ and $\nabla f_{t|\tau_2}(x)$, (iii) $\nabla f_{t_1|\tau}(x)$ and $\nabla f_{t_2|\tau}(x)$. Due to the complicated correlation, we decide to adopt the stochastic model in [1,15], which uses parametric functions $f(x;\theta)$ to capture the type-i correlation, and uses equation (9) to model the type-ii and type-iii. **R3-Q2: strongly convex and smooth costs.** Consider l_2-regularized logistic regression: $\min_x f(x; \{z_i, y_i\}_{i=1}^n) = -\sum_{i=1}^n \log(1+\exp(-y_i\langle x, z_i\rangle))/n + \lambda\|x\|^2/2$, where x is the regressor, $(z_i, y_i)_{i=1}^n$ denotes the samples, and λ is l_2 regularizer's coefficient. The loss function is strongly convex and smooth. (see [2] for more examples.) **R3-Q3: regret bound's order.** Please refer to R2-Q1 for the regret bound in the big-O notation. The prediction errors E_k can be either larger or smaller than V_T as mentioned in [20]. When $V_T = o(T)$ and $E_k = o(T)$ for $k \leq W$, the regret bound is $o(T)$. As a simple example of sublinear regrets, consider θ_{t-1} as the prediction of θ_{t+k} ($k \geq 0$) at time t, then $E_k = O(V_T)$ under proper assumptions, so when $V_T = o(T)$, the regret is $o(T)$. **R3-Q5: Cor. 3 is $O(T)$ when $W = 1$?** Yes. This is consistent with classic results in [1,15]. Intuitively, the stochastic model (9) indicates that the total variance of the prediction errors is $O(T)$, so the regret is at least $O(T)$ by the perturbation theory.

R4-Q1: comparison with RHGD. RHGD requires *accurate* W-stage predictions; while our RHIG can handle *inaccurate predictions*, which is more realistic and thus our regret bounds are more general and meaningful. For more technical discussion, please kindly refer to R1-Q1 & R3-Q4. Numerically, we observe better performance by RHIG since the carefully chosen stepsizes and parameters effectively reduce the error accumulation, while RHGD's performance is seriously downgraded by the prediction errors. **R4-Q2: comparison with RL.** Good question. This work and RL focus on different issues. Classical RL focuses on *unknown* systems in *stationary* environment, while this paper considers *time-varying/nonstationary* environment with *noisy future predictions* and a *known system*. Though one can apply RL algorithms to time-varying environment anyway, the online performance could be very undesirable without special treatment of the nonstationary environment. Our work addresses two under-explored issues in RL: *how to use the future predictions to improve the performance in nonstationary environment?* and *how to characterize the impact of the prediction errors?* Combining this work and RL is an exciting direction to promote RL's performance.