We thank the reviewers for their thoughtful comments and helpful feedback. Below, we address each reviewer in turn.

Review 1. *ℓ_p regression rates.* We agree that the ℓ_p regression runtimes obtained through our framework do not match the performance achieved by the state-the-art algorithms designed for the problem. However, we achieve these runtimes through a general framework that also yields state-of-the-art runtimes for ℓ_∞ and logistic regression. We believe that this sheds further light on the problem and is a step towards a more complete understanding of the problem class.

Other comments. In the revised version of the paper, we will emphasize in the intro that we obtain acceleration when our ball optimization oracle is restricted to affine transformations of Euclidean space (as opposed to a different norm, e.g., ℓ_1 or ℓ_∞). We will also add further comparison to notions of Hessian stability in different norms appearing in the literature; for instance, the work of Cohen et al. solves ball-optimization problems where the ball is measured in the ℓ_∞ norm. Regarding Ene and Vladu's IRLS-based algorithm for ℓ_∞, thank you very much for pointing this out. We will add a comparison to this result in our revision. Our algorithm works with a slightly different treatment of the constraint matrix $\min_x \|\mathbf{A}x - b\|_\infty$ (versus $\min_{\mathbf{A}x=b} \|x\|_\infty$), bounds iteration complexity in terms of $\|x^*\|_2$ (versus $\sqrt{m}\|x^*\|_\infty$ where m as dimension of x^*), and minimizes a softmax objective (versus ℓ_∞); we nevertheless believe our results can be adapted to their setting and our complexity is no worse (up to logarithmic factors) than theirs in the worst case and will comment on this in the revision.

Reviewer 2. *Theory vs. practice.* We agree that the main contribution of the paper lies on the theoretical side; our work provides a conceptually simple algorithmic framework with proof-of-concept complexity guarantees for a range of fundamental ML problems. Our proposed algorithms are not immediately suited for implementation, but opens the door to a promising new direction for designing practical algorithms with fewer loops and line searches.

Additional references. We appreciate the pointers to relevant papers. Using the notation of our paper, [FBR19] attains runtime bounds proportional to MR (which is at most $\sqrt{L/\mu}$), where M is the quasi-self-concordance (QSC) parameter and R is the domain size; our paper attains improved runtime guarantees proportional to $(MR)^{2/3}$. Apart from that, the runtimes of both papers depend polylogarithmically on the problem condition number L/μ. We will be sure to include this discussion in detail and the three relevant references on approximate solvers for AGD in the revision.

Reviewer 3. *Norm notation.* We use the general norm notation to include norms induced by any PSD matrix $\mathbf{M}$, which covers Euclidean ℓ_2 as a special case when $\mathbf{M} = \mathbf{I}$. This general norm is used throughout the paper, e.g. Def. 1, Cor. 12, and particularly for applications where $\mathbf{M} = \mathbf{A}^\top \mathbf{A}$. We will be sure to clarify this generality when using the notation.

Eqs. (1)–(2) notation. For ball optimization oracle with center z, radius r, the value of λ is a function of r, z which we denote by $\lambda_r(z)$; $y(\lambda)$ is the unique point y prescribed by a value λ given x, v. Thus, the implementation of the MS oracle boils down to solving the implicit equation $\lambda = \lambda_r(y(\lambda))$. We will make this clearer in the revision.

Bullins' paper. Thank you for pointing this out. While Bullins' paper does give the state-of-the-art algorithm for ℓ_p regression when $p = 4$, a more recent work by [AKPS19] (ref. [2] of our paper) achieves the state of the art polynomial dependence for ℓ_p regression for all $p \geq 2$ and matches Bullins' work in the special $p = 4$ case. We will be sure to discuss Bullins' result in our revision.

Broader impact. Thank you for the suggestion; we will include more discussion in the revision.

Reviewer 4. *Practicality and relevance to NeurIPS.* Our paper is theoretical, but we hope that our conceptual insights—particularly the ball optimization abstraction and its connection to Hessian stability—may inspire practical developments. We believe the paper is relevant to the ML audience because the problems for which we prove faster runtime bounds include soft-margin SVM and logistic regression, which are central in ML; therefore, many of the researchers likely to turn our theoretical insight into practice belong to the NeurIPS community. On a related note, we remark that many of the recent (purely theoretical) developments of MS acceleration appeared in ML venues (see refs. [9,17] in our paper and the Bullins (2020) paper pointed out by Reviewer 3).

Discussion on Theorem 8. It is true that μ-strongly convex L-smooth functions are L/μ-Hessian stable globally (over a ball of radius $r = \infty$). However, for a ball of radius $r < \infty$, the problems that we consider in our applications section have a stability parameter c much smaller than their condition number. Hessian stability is thus a distinct structural property from smoothness and strong convexity, allowing efficient minimization of certain poorly-conditioned functions.

Statement of Theorem 20. "Coin flips of an algorithm" is a common term for internal randomization in the algorithm. We explicitly construct the distribution over convex functions Appendix G by composing the the zero-chain defined in Eq. (34) with random orthogonal transformations. We will be sure to clarify this part in the revision.

Runtime comparisons for applications. For ℓ_p regression, we included known complexities from existing works in lines 109–122. For logistic and ℓ_∞ regression, the runtime comparisons are currently discussed in lines 66–85, and in more detail in the last paragraph of each application in Section 4. We will offer a more detailed comparison in the revision.