R1: We appreciate the reviewer's positive comments and for recognizing the novelty and importance of our work.

- "a similar algorithm already exists in the literature which the authors fail to cite nor to compare against" As per NeurIPS reviewing instructions: *"authors are excused for not knowing about all non-refereed work (e.g, those appearing on arXiv) . . . "* This already anticipates that extra care is in order when comparing to arXiv to avoid misunderstanding. We are quite aware of this paper and reiterate: our paper is original and the first with such an approach.
- "While theoretical results are sound and important from a dynamical systems perspective, it remains questionable in an optimization context." The significance of our work to optimization is that it opens the door for building new optimization algorithms from first principles by applying a general structure-preserving discretization scheme, i.e. Eq. (12), to a dynamical system. Another reason why our approach is relevant to optimization can be found in *"On Dissipative Symplectic Integration with Applications to Gradient-Based Optimization"* [arxiv.org/abs/2004.06840], Ref. [18]; see also L182–193 in the paper where it is shown that structure-preserving discretizations allow one to preserve rates of convergence. Symplectic discretizations of dissipative systems ensure that their "good-properties" are transferred automatically to discrete-time; we also provided compelling evidence why the relativistic system is relevant for optimization (L225–239, numerics, stability in the supplement, etc.). We will clarify and provide more intuition in the revision. We mention in passing that formal convergence-rate-results for the relativistic system are very challenging and beyond scope—see L307–314—however such a method approaches CM or NAG as limiting cases, hence its convergence cannot be worse than CM or NAG, which are well-understood in the literature.
- "The details of the numerics are vague . . . relevant metric." The implementation is simple, exactly as written, i.e., Algorithm 1, Eq. (1) and Eq. (2). The Bayesian optimization follows the original Ref. [34] (we used the Python library "hyperopt" provided by the authors). The parameters were optimized to give the smallest possible objective function value. We agree that we could have provided more details. Since space is short, we will include more comments about this in the supplement. Moreover, we will make our code publicly available. Thank you for bringing this up.
- "RGD never diverges." This was a misleading "abuse of language" on our part and will be corrected. Thank you.

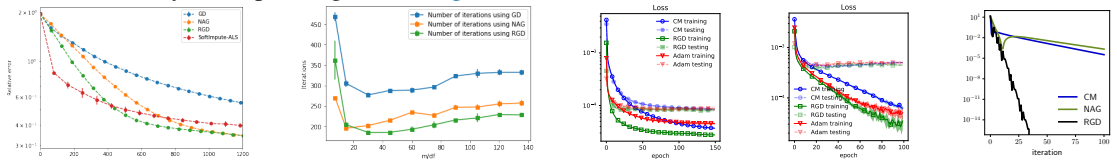
R3: We would like to thank the reviewer for comments and suggestions, which will be incorporated.

- *Weaknesses/additional feedback:* "More machine learning problems in the numerical section. A toy example . . . " We agree that practical ML problems are worth exploring. We show some preliminary results in the Figures below. We will improve these and include more ML experiments in the supplement (and also available code). In passing, let us mention that leading experts in the field have emphasized the great need for theoretical and principled approaches to construct new algorithms, as well as understand existing ones. *Our contributions are on these theoretical lines*. We hope the reviewer may appreciate their value, independent of practical experiments (although we provided some that confirm our claims). Our paper brings, and extends, important ideas from physics to ML/optimization.

Regarding the toy problem, the Rosenbrock case in the paper is a meaningful example. *We expect that our method, RGD, stands out when the objective has fast growing tails*; since RGD can control the "velocity" without having to reduce the step size, it can navigate through such landscapes more effectively. We will clarify and provide a couple of intuitive examples in the revision, such as in Fig. (e) below for a one-dimensional case. Thanks for raising this question.

"Is there a theoretical justification that shows the relativistic Hamiltonian is better than Euclidean . . . " This is a very deep question, whose answer requires some serious differential geometry; we just started to understand this and unfortunately is beyond scope. In short, a Hamiltonian dynamics happens in the cotangent bundle, $T^*\mathcal{M}$, of the base manifold $\mathcal{M}$ where the objective f is defined. The Lagrangian formalism happens on the tangent bundle $T\mathcal{M}$. The kinetic energy defines a metric on $T\mathcal{M}$ yielding different "momentum lengths." Thus different kinetic energies may yield better algorithms if they can "match/adapt" to the geometry of f; the relativistic is just one example. We believe there is a deep relation (i.e. an equation) that relates dynamical quantities to the duality gap, which may help to elucidate this.

- *Prior work:* Thank you for pointing out "Wang & Li (2019)," we will include a citation in the revised version.



Left to right. CM always close to NAG (when not shown). **RGD=ours**. (a) Movie Lens; SoftImput=[Hastie et al JMLR (2015)]. (b) Matrix Completion with noise; Error 10^{-5} of oracle bound [Candes&Plan IEEE (2009)]. (c) MNIST, feed forward CNN (3 layers), cross entropy. (d) VGG net (11 layers). (e) $f(x) = (1/8)(x^2+1)^4 - 1/8$; RGD stands out on fast growing tails.

R4: We appreciate R4's clear summary of our work. His/her understanding is very accurate, so we are surprised by his/her confidence score. We will certainly incorporate the "additional feedback." We are also encouraged to hear that the reviewer supports this line of work at the interface of dynamical systems, physics, and ML.

- "There are 2 extra hyperparameters compared to CM and NAG." (There is no free lunch!) In practice, there is actually only 1 extra parameter. We included α in Algo. 1 only to illustrate, in an unbiased manner, that preserving the symplectic structure can indeed be beneficial; all results in Fig. 2 favor $\alpha \to 1$, as opposed to $\alpha \to 0$. Thus, one should fix $\alpha = 1$. We will clarify this in the revision. Indeed, we will provide publicly available code.
- "Computational infrastructure and programming language" We used Python 3 on a Mac Book Pro, Quad-Core Intel i5, 16 GB RAM, and standard libraries such as numpy, scipy, hyperopt.