Conformal Symplectic and Relativistic Optimization

Guilherme França*, Jeremias Sulam, Daniel Robinson, René Vidal

Mathematical Institute for Data Science,
Johns Hopkins University, Baltimore, MD 21218, USA

Abstract

Recent work in machine learning has shown that optimization algorithms such as Nesterov's accelerated gradient can be obtained as the discretization of a continuous dynamical system. Since different discretizations can lead to different algorithms, it is important to choose the ones that preserve certain structural properties of the dynamical system, such as critical points, stability and convergence rates. In this paper we study structure-preserving discretizations for certain classes of dissipative systems, which allow us to analyze properties of existing accelerated algorithms as well as introduce new ones. In particular, we consider two classes of conformal Hamiltonian systems whose trajectories lie on a symplectic manifold, namely a classical mechanical system with linear dissipation and its relativistic extension, and propose discretizations based on conformal symplectic integrators which preserve this underlying symplectic geometry. We argue that conformal symplectic integrators can preserve convergence rates of the continuous system up to a negligible error. As a surprising consequence of our construction, we show that the well-known and widely used classical momentum method is a symplectic integrator, while the popular Nesterov's accelerated gradient is not. Moreover, we introduce a relativistic generalization of classical momentum, called relativistic gradient descent, which is symplectic, includes normalization of the momentum, and may result in more stable/faster optimization for some problems.

*guifranca@gmail.com

Contents

1 Introduction

Gradient descent based methods are ubiquitous in machine learning because they only require first-order information about the objective function which makes them computationally efficient. However, vanilla gradient descent can be slow. Alternatively, *accelerated gradient methods*, whose basic construction can be traced back to Polyak [1] and Nesterov [2], became popular due to their ability to achieve best worst-case complexity bounds. Gradient descent with momentum, or simply *classical momentum* (CM) for short, is often found as [3]

$$v_{k+1} = \mu v_k - \epsilon \nabla f(q_k), \tag{1a}$$

$$q_{k+1} = q_k + v_{k+1}, \tag{1b}$$

where $\mu \in (0, 1)$ is the momentum factor, $\epsilon > 0$ the stepsize or learning rate, and $f : \mathbb{R}^n \to \mathbb{R}$ the function being minimized. *Nesterov's accelerated gradient* (NAG) can also be found in

a similar form [3], namely

$$v_{k+1} = \mu v_k - \epsilon \nabla f(q_k + \mu v_k), \tag{2a}$$

$$q_{k+1} = q_k + v_{k+1}. \tag{2b}$$

Both algorithms have a long history in optimization and have been extensively applied in deep learning [3]. They are also the basic prototype for other methods such as RMSprop [4], Adam [5] and AdaGrad [6], which in addition include different normalizations of the gradient. Despite their increasing popularity, a complete understanding of these adaptive methods is still lacking and their benefits remain unclear [7].

A promising direction for understanding accelerated methods has been emerging by making connections between optimization and continuous dynamical systems [8–18]. From this perspective, both CM and NAG correspond to different discretizations of the same continuous system, namely (6). Moreover, starting from this differential equation one is free to choose a gamut of discretization techniques, each leading to a different algorithm that simulates the continuous dynamics to some degree of approximation. It is important to note that when describing natural phenomena, the continuous dynamical system is the fundamental object of study. The main goal of a discretization is thus to numerically reproduce the system's behavior as close as possible. Unfortunately, typical discretization introduce spurious artifacts and may not preserve the most important properties of the continuous system [19]. Therefore, given a continuous dynamical system with desirable properties for optimization, a reasonable approach to constructing optimization algorithms is to look for discretizations that:

- Preserve the most important properties of the system and reproduce the qualitative behavior of its trajectories.
- Preserve the system's phase portrait, i.e. critical points together with their stability.
- Preserve the rates of convergence of trajectories to critical points, at least to some degree of accuracy.

In the case of Hamiltonian systems there exists a class of discretizations that satisfy these conditions, known as *symplectic integrators* [19,20]. These methods were originally developed for conservative systems and date back to the 50's [21], but mostly went unnoticed. Only later in the 80's was this approach rediscovered and gained interest [22–27]. (See [28,29] for a historical account.) Nowadays, symplectic integrators have been widely used across many areas of physics such as statistical mechanics, nonlinear and molecular dynamics, complex systems, Monte Carlo methods, particle physics, and astrophysics. Nevertheless, only recently have they started to be considered in optimization [30]. More relevant for optimization purposes

are Hamiltonian systems with a linear dissipation, the so-called *conformal Hamiltonian systems* [31]. The main results from symplectic integrators can be naturally extended to this class of dissipative systems. Such methods are referred to as *conformal symplectic integrators* and have been recently explored [32]. This is the discretization technique that will be considered in this paper.

Outline and Contributions We begin in Section 2 with an overview of the basics about conformal Hamiltonian systems, highlighting some important consequences of their intrinsic symplectic geometry. In Section 3, we construct a conformal symplectic integrator for a generic Hamiltonian system, which leads to a large family of optimization algorithms parameterized by different choices of the kinetic energy. In Section 4, we apply this proposed approach to the classical Hamiltonian (5) and use it to analyze the CM and NAG algorithms given by (1) and (2), respectively. Specifically, we show that CM is a conformal symplectic integrator (Corollary 4) while NAG is not (Theorem 5). Both are surprising new results for the optimization literature. In Section 5, we provide general arguments supporting why conformal symplectic integrators preserve convergence rates of the continuous system up to a negligible error. In Section 6, we apply our general symplectic integrator to a relativistic system with linear dissipation. As a consequence, we derive the following new *relativistic gradient descent* (RGD) algorithm:[1]

$$v_{k+1} = \mu v_k - \epsilon \nabla f(q_k), \tag{3a}$$

$$q_{k+1} = q_k + \frac{v_{k+1}}{\sqrt{1 + \|v_{k+1}\|^2/v_c^2}}, \tag{3b}$$

where $v_c > 0$ is a constant playing the role of the speed of light. Note that in the limit $\|v\| \ll v_c$, RGD recovers CM. Thus, RGD has at least the same performance as CM but also the potential for improvement. We illustrate numerically that RGD is more stable and significantly faster for some problems. Importantly, RGD resembles the popular Adam, AdaGrad, and RMSprop, but employs a different type of normalization. However, unlike these alternatives, RGD enjoys an elegant theoretical justification in terms of relativistic mechanics and conformal symplectic integrators.

Related Work There are only a couple of papers related to this work. [30] is the first to consider symplectic integrators in optimization. However, *conformal* Hamiltonian systems—which we believe are the natural way to approach the problem—were not considered. Instead, a standard leapfrog integrator, for conservative systems, was applied to a nonautonomous system written in the extended phase space. The authors also modified the leapfrog method

[1] We actually prefer the updates (37) because they have a more physical interpretation of parameters. We simply mention (3) for purposes of comparison with (1) at this stage.

by adding a gradient flow, which ultimately breaks the symplectic structure. [33] is the first to introduce relativistic mechanics in a machine learning context, i.e. in Monte Carlo methods, but without relationship to either conformal Hamiltonian systems or (conformal) symplectic integrators. Also, their starting point is not our conformal relativistic system described by the differential equations (36). Explicit and implicit Euler discretizations of a conformal Hamiltonian system were considered in [34], but without connections to (conformal) symplectic integrators. They focus on generalized kinetic energies allowing linear convergence without strong convexity assumption. Although the relativistic kinetic energy is considered, the role of the speed of light is completely absent, which is an essential feature of relativistic discretizations as we explain in Section 6. Generalized kinetic energies were also previously considered in Monte Carlo methods [35].

2 Conformal Hamiltonian Systems

We start by introducing the basics of conformal Hamiltonian systems and focus on their intrinsic symplectic geometry (we refer the reader to [31] for more details). The state of the system is described by a point on phase space $(q,p) \in \mathbb{R}^{2n}$, where $q = q(t) \in \mathbb{R}^n$ is the generalized coordinates, $p = p(t) \in \mathbb{R}^n$ the conjugate momentum associated to q, and trajectories are parametrized by time $t \in \mathbb{R}$. The system is completely specified by a Hamiltonian $H : \mathbb{R}^{2n} \to \mathbb{R}$. Thus, a conformal vector field is required to obey the modified version of Hamilton's equations given by

$$\dot{q} = \nabla_p H(q,p), \qquad \dot{p} = -\nabla_q H(q,p) - \gamma p, \tag{4}$$

where $\dot{q} \equiv \frac{dq}{dt}$, $\dot{p} \equiv \frac{dp}{dt}$, and $\gamma > 0$ is a damping *constant* responsible for dissipating the energy of the system. A classical example is given by

$$H(q,p) = \frac{\|p\|^2}{2m} + f(q) \tag{5}$$

where m is the mass of a particle subject to the potential f. From (4) we obtain the equations of motion

$$\dot{q} = \frac{p}{m}, \qquad \dot{p} = -\nabla f(q) - \gamma p. \tag{6}$$

The Hamiltonian is the energy of the system. Taking its total time derivative along trajectories one finds $\dot{H} = -\gamma\|p\|^2 \leq 0$. Therefore, H is a Lyapunov function and all orbits tend to fixed points, which in this case must satisfy $\nabla f(q) = 0$ and $p = 0$. Note that (6) is a standard system in classical mechanics, being a nonlinear generalization of the harmonic oscillator with friction.

The most important property of conformal Hamiltonian systems is is their underlying symplectic geometry. Let us define

$$z \equiv \begin{bmatrix} q \\ p \end{bmatrix}, \qquad \Omega \equiv \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}, \qquad D = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}, \tag{7}$$

where I is the $n \times n$ identity matrix. The system (4) can thus be concisely written as

$$\dot{z} = \Omega \nabla H(z) - \gamma D z. \tag{8}$$

Note that $\Omega\Omega^T = \Omega^T\Omega = I$ and $\Omega^2 = -I$, so that Ω is real, orthogonal and antisymmetric. Let $\xi, \eta \in \mathbb{R}^{2n}$ and define the symplectic 2-form[2]

$$\omega(\xi, \eta) \equiv \xi^T \Omega \, \eta. \tag{9}$$

A transformation $G : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ is said to be *symplectic* if $\omega(G\xi, G\eta) = \omega(\xi, \eta)$, which is equivalent to $G^T \Omega \, G = \Omega$. The equations of motion define a flow $\Phi_t : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ according to $\Phi_t\big(z_0\big) \equiv z(t)$, where $z(t)$ is the trajectory at time t with initial condition $z(0) = z_0$. Let $J_t(z)$ denote the Jacobian matrix of $\Phi_t(z)$. From (8) it is possible to show that [31][3]

$$J_t^T \Omega J_t = e^{-\gamma t}\Omega \quad \implies \quad \omega_t = e^{-\gamma t}\omega_0. \tag{10}$$

Therefore, a conformal Hamiltonian flow Φ_t *contracts the symplectic form exponentially.* Moreover, it follows from (10) that volumes on phase space shrink at a rate given by

$$\mathrm{vol}(\Phi_t(\mathcal{R})) = \int_{\mathcal{R}} |\det J_t(z)| dz = e^{-n\gamma t} \, \mathrm{vol}(\mathcal{R}) \tag{11}$$

where $\mathcal{R} \subset \mathbb{R}^{2n}$. This contraction is stronger as dimension increases. In the conservative case ($\gamma = 0$) the symplectic form is preserved and volumes are left invariant. It can also be shown that any first integral $\mathcal{Q}$ of a conservative system, i.e. one such that $\dot{\mathcal{Q}} = 0$, obeys $\dot{\mathcal{Q}} = -\gamma \mathcal{Q}$, or equivalently $\mathcal{Q}_t = e^{-\gamma t}\mathcal{Q}_0$, in the conformal case ($\gamma \neq 0$) [36]. Another known property of conformal Hamiltonian systems is that their Lyapunov exponents sum in pairs to γ [37]. This imposes constraints on the admissible dynamics and controls the phase portrait near fixed points. For other properties of attractor sets we refer to [38]. Finally, conformal symplectic transformations can be composed and form the so-called *conformal group*.

[2] In the language of differential geometry, we have a 2-form $\omega_i(\xi, \eta) = (dq_i \wedge dp_i)(\xi, \eta)$, where $\wedge$ is the wedge product. For any 1-forms $\alpha, \beta : \mathbb{R}^{2n} \to \mathbb{R}$ we have $\alpha \wedge \beta(\xi, \eta) \equiv \alpha(\xi)\beta(\eta) - \alpha(\eta)\beta(\xi)$. Also, $dg(\eta)(\xi) \equiv \langle \nabla g(\eta), \xi \rangle$ defines a 1-form for any differentiable function $g : \mathbb{R}^{2n} \to \mathbb{R}$. Therefore, $\omega_i = dq_i \wedge dp_i$ and summing over all these area elements we have $\omega = \sum_i dq_i \wedge dp_i$, which is (9).

[3] Here we mean that $\omega_t \equiv \sum_i dq_i(t) \wedge dp_i(t)$, whereas $\omega_0 = \sum_i dq_i(0) \wedge dp_i(0)$, and the states at different times are related through the flow mapping $z(t) = \Phi_t(z(0))$, where $z(t) = \left[\begin{smallmatrix} q(t) \\ p(t) \end{smallmatrix}\right]$.

3 Conformal Symplectic Optimization

In this section, we derive a family of optimization algorithms through a discretization of the general conformal Hamiltonian system (4). Such construction will be employed later to obtain properties of known optimization algorithms, and also to introduce a new method.

Consider the Hamiltonian system written in the form (8), i.e.

$$\dot{z} = \underbrace{\Omega \nabla H(z)}_{C(z)} - \underbrace{\gamma D z}_{D(z)} \tag{12}$$

where we associate flow maps Φ_t^C and Φ_t^D to the respective vector fields $C(z)$ and $D(z)$. Symplectic integrators are splitting methods that approximate the true flow of the system, Φ_t, by composing the individual flows Φ_t^C and Φ_t^D [39]. The procedure to construct a numerical integrator, with a fixed stepsize $h > 0$, is to first obtain a numerical approximation to the conservative part of the system, $\dot{z} = \Omega \nabla H(z)$. This yields a numerical map $\hat{\Phi}_h^C$ that approximates Φ_h^C for small intervals of time $[t, t+h]$. One can choose any standard symplectic integrator for this task. We pick the simplest one which is a version of the *symplectic Euler* method [39]. We thus have $\Phi_h^C : (q, p) \mapsto (Q, P)$ where

$$Q = q + h\nabla_p H(q, P), \qquad P = p - h\nabla_q H(q, P). \tag{13}$$

Note that in general this method is implicit on P, however it will become explicit for separable Hamiltonians (this will be clear shortly). Now, the dissipative part of the system, $\dot{z} = -\gamma D z$, can actually be integrated exactly. Indeed, we have the two equations $\dot{q} = 0$ and $\dot{p} = -\gamma p$ thus the mapping $\hat{\Phi}_h^D : (q, p) \mapsto (Q, P)$ is given by

$$Q = q, \qquad P = e^{-\gamma h} p. \tag{14}$$

Now, consider the composition $\hat{\Phi}_h \equiv \hat{\Phi}_h^C \circ \hat{\Phi}_h^D$. Making use of (13) and (14) we obtain the mapping $\hat{\Phi}_h : (q, p) \mapsto (Q, P)$ given by

$$P = e^{-\gamma h} p - h\nabla_q H(q, P), \tag{15a}$$

$$Q = q + h\nabla_p H(q, P). \tag{15b}$$

As it stands this method is implicit on P, nevertheless it is a completely general integrator for the dynamical system (4) with an arbitrary Hamiltonian function H. Let us now assume that the Hamiltonian is separable in the form $H(q, p) = T(p) + f(q)$, where T is the kinetic energy and f is the potential energy—in this context also the objective function intended to be minimized. Let $q(t_k)$ and $p(t_k)$ denote the true states of the continuous system at discrete instants of time $t_k = kh$, for $k = 0, 1, \ldots$. Denoting the respective numerical estimate of

such states by q_k and p_k, we can write (15) in the more familiar algorithmic form

$$p_{k+1} = e^{-\gamma h} p_k - h\nabla f(q_k), \tag{16a}$$
$$q_{k+1} = q_k + h\nabla T(p_{k+1}). \tag{16b}$$

Note the these updates became explicit under a separable Hamiltonian. Moreover, only one gradient computation ∇f per iteration is required, so this method is computationally cheap. The updates (16) consists of a family of algorithms parametrized by the choice of kinetic energy T. When $\gamma = 0$ these updates reduce to the standard symplectic Euler method.

Before showing important properties of (16), or more generally (15), let us define precisely what we mean by a conformal symplectic integrator.

Definition 1. *A numerical one-step map $\hat{\Phi}_h$, where h is a stepsize, is said to be conformal symplectic if $z_{k+1} = \hat{\Phi}_h(z_k)$ is conformal symplectic, i.e. $\omega_{k+1} = e^{-\gamma h}\omega_k$, whenever $\hat{\Phi}_h$ is applied to a smooth Hamiltonian. Iterating such a map k times yields $\omega_k = e^{-\gamma t_k}\omega_0$, so that the contraction of the symplectic form* (10) *is preserved.*

Therefore, a conformal symplectic integrator is designed to preserve the contraction of the symplectic form. This automatically allows the numerical method to reproduce the qualitative behaviour of the continuous system, since all properties arising from the symplectic structure are preserved.[4]

Theorem 2. *The general mapping* (15) *is a conformal symplectic integrator. Thus, the method* (16) *is also conformal symplectic since it is a particular case.*

Proof. The variational form of (15) can be written as

$$(I + hH_{qp})dP = e^{-\gamma h}dp - hH_{qq}dq, \tag{17a}$$
$$dQ - hH_{pp}dP = dq + hH_{qp}dq, \tag{17b}$$

where $H_{qp} = \frac{\partial^2}{\partial q \partial p} H(q, P)$ denotes a matrix of second-order partial derivatives, i.e. a block of the Hessian of H. It is implicit that all the above derivatives are computed at the point (q, P), which we omit for the sake of simplicity. Taking the wedge product between both equations in (17), upon using the basic properties of the wedge product[5] and symmetry of the Hessian, one obtains

$$(I + hH_{qp})dQ \wedge dP = e^{-\gamma h}(I + hH_{qp})dq \wedge dp. \tag{18}$$

[4] For the reader unfamiliar with these concepts, a nice and short introduction is [40].

[5] We recall that such basic properties are (1) antisymmetry: $x \wedge y = -y \wedge x$; (2) bilinearity: $x \wedge (c_1 y + c_2 z) = c_1 x \wedge y + c_2 x \wedge z$; and (3) matrix multiplication: $x \wedge (My) = (M^T x) \wedge y$. Here x, y, z are vectors, c_1, c_2 are constants, and M is a matrix.

Since H is arbitrary and $I + hH_{qp}$ invertible, this implies that

$$dQ \wedge dP = e^{-\gamma h} dq \wedge dp \tag{19}$$

as desired. Therefore, by Definition 1 we conclude that (15) is conformal symplectic. □

We now address the question of how accurate (15) approximates a true trajectory of the continuous dynamical system.

Theorem 3. *The numerical scheme* (15) *is first-order accurate, namely*

$$\hat{\Phi}_h(z) - \Phi_h(z) = O(h^2). \tag{20}$$

Proof. From the equations of motion (4) and a Taylor expansion we have

$$q(t_{k+1}) = q(t_k + h) = q + h\nabla_p H(q,p) + O(h^2), \tag{21a}$$
$$p(t_{k+1}) = p(t_k + h) = p - h\nabla_q H(q,p) - \gamma h p + O(h^2), \tag{21b}$$

where we denote $q \equiv q(t_k)$ and $p = p(t_k)$ for simplicity. Under one step of the mapping (15), starting from the point $(q,p) = (q(t_k), p(t_k))$, we thus have

$$q_{k+1} = q + h\nabla_p H\left(q, p + O(h)\right) = q + h\nabla_p H(q,p) + O(h^2), \tag{22a}$$
$$p_{k+1} = e^{-\gamma h} p - h\nabla_q H\left(q, p + O(h)\right) = p - \gamma h p - h\nabla_q H(q,p) + O(h^2). \tag{22b}$$

Comparing (22) with (21) yields

$$q_{k+1} = q(t_{k+1}) + O(h^2), \qquad p_{k+1} = p(t_{k+1}) + O(h^2), \tag{23}$$

which is just another way of writting (20). □

Therefore, the generic method (15) is a conformal symplectic integrator and reproduces the trajectories of the continuous system (4) up to first-order of accuracy. The relations (20) provide a local error, i.e. the error in only one step of the method. By iterating such a map from $t = 0$ up to a finite time $t_k = kh$, it is possible to show that the global error is given by $\hat{\Phi}_h^k(z_0) - \Phi_h^k(z_0) = O(h)$ [39].

4 Relationship to Classical Momentum and Nesterov

The conformal symplectic integrator (16) is completely general, allowing one to use any—meaningful—kinetic energy T and potential function f. Restricting to the case of the classical Hamiltonian (5) such updates become

$$p_{k+1} = e^{-\gamma h} p_k - h\nabla f(q_k), \tag{24a}$$
$$q_{k+1} = q_k + (h/m) p_{k+1}. \tag{24b}$$

The reader may recognize that this is precisely the CM method (1) under the particular choice

$$m = h = \epsilon, \qquad \mu = e^{-\gamma h}. \tag{25}$$

Therefore, the well-known CM algorithm (1)—which has been extensively employed in machine learning and optimization—is nothing but a dissipative version of the symplectic Euler method! The algorithm (24) generalizes CM by incorporating a mass parameter m. The relation (25) also explains why the momentum factor must be in the range $0 < \mu < 1$, and is related to the amount of friction imposed on the system. As a direct consequence of Theorems 2 and 3, the method (24) is a first-order conformal symplectic integrator for the Hamiltonian system (6). We state this surprising result explicitly.

Corollary 4. *The classical momentum method* (1) *is a conformal symplectic integrator for the Hamiltonian system* (6)*. Moreover, it is a first-order integrator.*

Due to the close relationship between CM and NAG one might wonder if the latter is also conformal symplectic. The following result answers this question negatively.

Theorem 5. *Nesterov's accelerated gradient* (2) *is not a conformal symplectic integrator for the Hamiltonian system* (6)*.*

Proof. Consider the mapping $(q, p) \mapsto (Q, P)$ given by

$$\tilde{Q} = q + \tfrac{h}{2m} e^{-\gamma h} p, \tag{26a}$$
$$P = e^{-\gamma h} p - h \nabla f\big(\tilde{Q}\big), \tag{26b}$$
$$Q = q + \tfrac{h}{2m} P, \tag{26c}$$

which consist of a discretization of the Hamiltonian system (6). Note that with the choice

$$m = h/2 = \epsilon/2, \qquad \mu = e^{-\gamma h}, \tag{27}$$

this is precisely NAG (2). Therefore, it suffices to show that (26) is not conformal symplectic. The variational form of these updates are given by

$$d\tilde{Q} = dq + \tfrac{h}{2m} e^{-\gamma h} dp, \tag{28a}$$
$$dP = e^{-\gamma h} dp - h \nabla^2 f\big(\tilde{Q}\big) d\tilde{Q}, \tag{28b}$$
$$dQ = dq + \tfrac{h}{2m} dP. \tag{28c}$$

Using the antisymmetry of the wedge product we conclude that

$$dQ \wedge dP = dq \wedge dP = e^{-\gamma h} dq \wedge dp - h dq \wedge \nabla^2 f\big(\tilde{Q}\big) d\tilde{Q}. \tag{29}$$

Moreover,

$$dq \wedge \nabla^2 f\big(\tilde{Q}\big) d\tilde{Q} = \tfrac{he^{-\gamma h}}{2m} dq \wedge \nabla^2 f(q) dp + O(h^2) \tag{30}$$

which in general does not vanish. Therefore, $dQ \wedge dP \neq e^{-\gamma h} dq \wedge dp$ implying that (26) is not conformal symplectic. □

It is perhaps more common to find Nesterov's method written as

$$q_{k+1} = y_k - \epsilon \nabla f(y_k), \tag{31a}$$

$$y_{k+1} = q_{k+1} + \mu_{k+1}(q_{k+1} - q_k), \tag{31b}$$

where $\mu_{k+1} = \frac{k}{k+r}$ with $r \geq 3$. This system can also be written in the exact same form as (2), but now with an adaptive μ_k. This can be seen by introducing the variable $v_k \equiv q_k - q_{k-1}$ and writing the updates in terms of q and v. Thus, when μ_k is constant we already showed in Theorem 5 that (31) is not conformal symplectic. In the case $\mu_k = \frac{k}{k+r}$, the differential equation associated to (31) is equivalent to (4) after the substitution $\gamma \to r/t$ [8]. Since γ is now time dependent, the system is no longer conformal [31]. Therefore, also in its instantiation (31), NAG cannot be a conformal symplectic integrator.

From a purely optimization perspective, the implications of Theorem 5, in contrast with Corollary 4, does not imply that NAG is inferior nor superior to CM. Such results simply state that NAG is not a geometric integrator, as opposed to CM. These are interesting and important results in their own right. However, from the perspective of being a discretization of a continuous system, NAG indeed does not respect the most important property of the underlying dynamical system, contrary to CM. In this sense, NAG introduces spurious ingredients not truly present in the continuous dynamics and its stability properties might be different compared to the continuous system. Whether NAG's discretization is beneficial or not for optimization is an interesting open question whose answer could bring insights into the construction of accelerated methods.

5 Continuous versus Discrete Convergence Rates

The dynamical system (6) is interesting for optimization because the energy is being dissipated at a constant rate[6] thus trajectories converge asymptotically to a state $q^\star$ that corresponds to a minimum of f. Under certain assumptions on f, it is possible to compute convergence rates for this process. For instance, the recent results of [18] immediately apply to system (6) as a particular case, from which one concludes the following:

[6] The linear dissipation of the energy holds in general provided H is a convex function of p.

- If f is convex then for all $t_k \geq 1/\gamma$, where $t_k = kh$, it holds that

$$f(q(t_k)) - f(q^\star) = O(\gamma m/(hk)). \tag{32}$$

- If f is M-strongly convex then for all $t_k \geq 0$, and with $\gamma < \frac{3}{2}\sqrt{M/m}$, it holds that

$$f(q(t_k)) - f(q^\star) = O(\gamma^2 m e^{-2\gamma hk/3}). \tag{33}$$

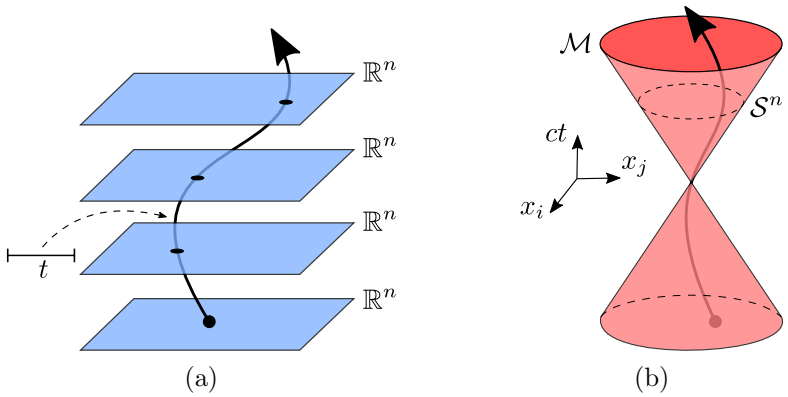
What can be said about these rates for discretizations of the system? Suppose we have a numerical integrator of order $r \geq 1$, which means that over a finite interval t_k we have the global error $\|q_k - q(t_k)\| \leq C_k h^r$, where q_k denotes the numerical estimate to the true trajectory $q(t_k)$. The constant C_k is independent on the stepsize h, but depends on t_k [39,41]. Assuming that f has a bounded gradient $\|\nabla f\| \leq L$, this implies $|f(q_k) - f(q(t_k))| \leq LC_k h^r$. Therefore,

$$f(q_k) - f(q^\star) \leq f(q(t_k)) - f(q^\star) + LC_k h^r. \tag{34}$$

This shows that any rate of the continuous system, such as (32) and (33), holds for the discretization up to an error that depends on the stepsize h and on the time interval t_k, through the constant C_k.

For typical discretizations one has $C_k = c_1(e^{c_2 t_k} - 1)$.[7] Since C_k grows *exponentially* with t_k, the estimate (34) becomes completely useless if t_k is large and h not sufficiently small, i.e. for a fixed stepsize h this estimate is valid up to $t_k \ll c_2^{-1} \log\left(1 + \frac{1}{c_1 L h^r}\right)$. However, for symplectic integrators this is not the case. Indeed, the constant C_k can actually be made *exponentially small*. For instance, $C_k = c_1(e^{c_2 t_k} - 1)e^{-c_3/h}$ holds for time intervals $t_k \ll c_3/(hc_2)$ [41,42]. It is also possible to extend this to exponentially large time intervals $t_k = he^{c_4/h}$ [42,43]. Unfortunately, elaborating more on these ideas, which involve backward error analysis and sophisticated construction of the so-called shadow Hamiltonians, are beyond the scope of this paper. The main reason behind this unique property of symplectic integrators is that they can be seen as an *exact* integrator of another Hamiltonian system, which is a very small perturbation of the original Hamiltonian system. Therefore, if the original Hamiltonian system is stable under small perturbations, a symplectic integrator will reproduce its qualitative behaviour. This is one of the reasons why symplectic integrators are extensively used in molecular dynamics and astrophysics, which require extremely large simulation intervals t_k. In optimization the situation is even better since time intervals of this order of magnitude are unlikely to occur because the dissipation brings the system to small regions around fixed points quite fast. Thus, for (conformal) symplectic integrators there are strong arguments assuring that continuous-time rates are preserved in discrete-time, up to a negligible error that can be controlled with the stepsize.

[7] Here, $c_1, c_2, \ldots > 0$ are all constants that are independent of the stepsize h or time interval t_k.



(a) (b)

Figure 1: (a) Newtonian versus (b) Minkowski space. (a) The worldline of a particle lies on $\mathbb{R}^n$ and time is just a parametrization of the curve. (b) In Minkwoski space the worldline is confined to lie inside the cone whose boundary correspond to the maximum speed of light. A slice of time corresponds to an n-dimensional hypersphere $\mathcal{S}^n$ of light frontwaves. In this geometry, velocities are bounded. When $c \to \infty$ one recovers the classical Newtonian space.

6 Relativistic Optimization

For the reader unfamiliar with special relativity, let us just briefly touch on some of its simple but fundamental concepts to motivate our approach. The previous algorithms are based on (6) which is a classical Newtonian system, where time is just a parameter that is independent of the Euclidean space $\mathbb{R}^n$. This implies that there is no restriction on the speed $\|v\| = \|dx/dt\|$ that a particle can attain. This translates to a discretization such as (24) where large gradients ∇f give rise to a large momentum p, implying that the position updates q can diverge. On the other hand, in special relativity the space and time form a unified geometric entity, the $(n+1)$-dimensional Minkowski space $\mathcal{M}$ with coordinates $X = (ct; x)$, where c is the speed of light in vacuum—which is a unified constant of nature. An infinitesimal distance on this manifold is given by $ds^2 = -(cdt)^2 + \|dx\|^2$. Null geodesics correspond to $ds^2 = 0$, implying that $\|v\|^2 = \|dx/dt\|^2 = c^2$, i.e. no particle can travel faster than c. This imposes constraints on the geometry where trajectories take place, as illustrated in Fig. 1. With that being said, the basic idea is that through discretizing a relativistic system we can incorporate these features into an optimization algorithm, which hopefully can bring some benefits such as an improved stability.

A relativistic particle subject to a potential f is described by the Hamiltonian [44]

$$H(q,p) = c\sqrt{\|p\|^2 + (mc)^2} + f(q). \tag{35}$$

In the classical limit, $\|p\| \ll mc$, one obtains $H = mc^2 + \|p\|^2/(2m) + f(q) + O(1/c^2)$, recovering (5) up to a constant $E_0 = mc^2$ which has no effect in deriving the equations of

motion (6)—this constant is precisely the famous Einstein's equivalence between mass and energy. Replacing (35) into (4) we thus obtain the dissipative relativistic system

$$\dot{q} = \frac{cp}{\sqrt{\|p\|^2 + (mc)^2}}, \qquad \dot{p} = -\nabla f - \gamma p. \tag{36}$$

Note that in the classical limit the equations (36) recover (6). Importantly, in the dynamical system (36) the momentum is normalized by the $\sqrt{\cdot}$ factor, so that $\dot{q}$ remains bounded even if p was to go unbounded. Now, applying our general conformal symplectic integrator (16) to this case, which simply amounts to replacing the kinetic energy T of (35), we obtain

$$p_{k+1} = e^{-\gamma h} p_k - h\nabla f(q_k), \tag{37a}$$

$$q_{k+1} = q_k + h\frac{cp_{k+1}}{\sqrt{\|p_{k+1}\|^2 + (mc)^2}}. \tag{37b}$$

We call this method *relativistic gradient descent* (RGD).[8] Note that RGD recovers the classical algorithm (24) in the limit $c \to \infty$, which is closely related to the CM method (1).[9] It is worth stressing that, as a consequence of Theorem 2 and Theorem 3, we immediately conclude the following.

Corollary 6. *The relativistic gradient descent method* (37) *is first-order accurate. More importantly, it is a conformal symplectic integrator for the dynamical system* (36)*.*

In special relativity c is a universal constant, but in RGD it is a free parameter that implies a cutoff to the q update and can prevent divergences, assuring at least $\|q_{k+1} - q_k\| \leq hc$. Thus, RGD can be more stable and has at least the same performance as CM, since the latter is obtained as a particular case, namelly $m = h$ and $c \to \infty$. We emphasize that relativistic effects are only noticeable when particles have a velocity comparable to c. For $\|p\| \ll mc$, classical and relativistic mechanics coincide and there should be no significant difference between RGD and the classical method (24). In practice, it is important to choose the speed of light c carefully. If c is too large we recover (24), but if c is too small the system can have very slow convergence or not even update the q variable enough. The optimal choice of c naturally depends on the problem, i.e. on the objective function f. Moreover, from physics we know that particles that travel close to the speed of light must be massless. Therefore, when tuning RGD, we should aim at making m as small as possible while allowing the algorithm to converge by controlling c.

[8] The updates (3) are obtained from (37) under the choice $h = m = \epsilon$ and $v_c = hc$. Although (3) looks closer to (1), the algorithm (37) have a more intuitive physical interpretation and is preferred.

[9] The reader more familiar with momentum based optimization methods may wonder why we choose "relativistic gradient descent" instead of "relativistic momentum". This is because any relativistic mechanical system is a second-order differential equation and therefore always has momentum, i.e. relativistic momentum is redundant from a physics point of view.

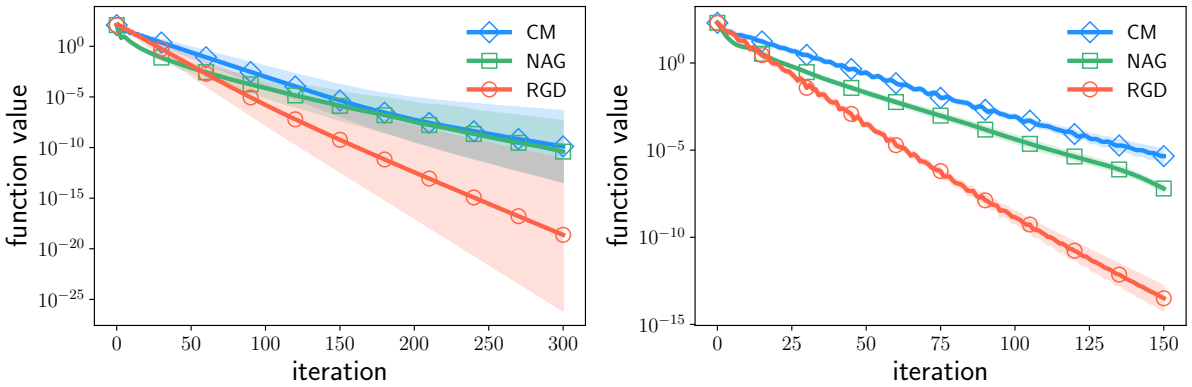


Figure 2: (a) Random quadratic function (38) where we perform 50 trials, sampling A, and $q_0 = (1, \dots, 1)^T$, $p_0 = 0$ is the initial state. In each case we tune every algorithm. Solid lines are the mean and shaded areas $\pm$ standard deviation. (b) Correlated quadratic (39) over 200 experiments where $-1 \leq q_{0,i} \leq 1$ is chosen uniformly at random. In both cases we see a significant faster convergence of RGD.

7 Numerical Experiments

We now compare RGD against the well-known CM and NAG algorithms on some test problems. Some physical intuition on parameter tuning is discussed in Appendix A. The reader can also find details about the tuning procedure for each experiment in Appendix B. In all cases, we set the initial momentum to $p_0 = 0$. We note that each algorithm was tuned independently through an exhaustive random search on its parameter space.

Quadratic Functions Let us start with a simple quadratic function

$$f(q) = \tfrac{1}{2} q^T A\, q, \qquad \lambda(A) \overset{iid}{\sim} \mathcal{U}\big(10^{-3}, 1\big), \tag{38}$$

where $A \in \mathbb{R}^{500\times 500}$ is a positive definite random matrix with eigenvalues uniformly distributed on the range $[10^{-3}, 1]$. In Fig. 2a we show the convergence rate of each algorithm when minimizing such a function over 50 Monte Carlo runs.

Next, we consider the correlated quadratic function

$$f(q) = \tfrac{1}{2} q^T A\, q, \qquad A_{ij} = \frac{\sqrt{ij}}{2^{|i-j|}}, \tag{39}$$

for $i, j = 1, \dots, 50$. We perform 200 Monte Carlo runs where on each trial we sample the initial position uniformly at random in the range $-1 \leq q_{0,i} \leq +1$. The results are in Fig. 2b.

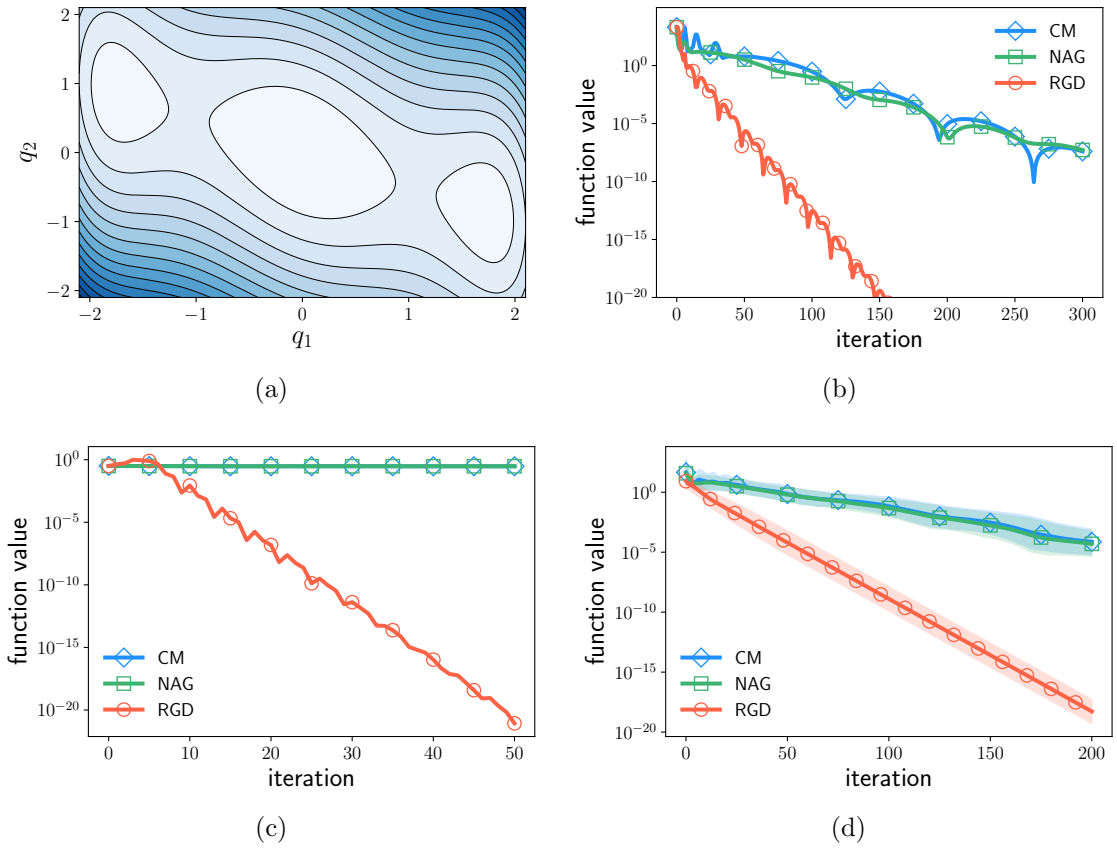


Figure 3: (a) Contour plot of the Camelback function (40). Note the global minimum at the center and the two other local minima. (b) Convergence rate starting at $q_0 = (5, 5)^T$. (c) We initialize close to a local minima, $q_0 = (1.8, -0.9)^T$. (d) We perform 500 experiments where $-5 \leq q_{0,i} \leq 5$ is chosen uniformly at random. We show convergence to the global minimum where solid lines are the mean and shaded area $\pm$ standard deviation.

Camelback Function We now consider the nonconvex Camelback function with three humps [45]:

$$f(q) \equiv 2q_1^2 - 1.05q_1^4 + \tfrac{1}{6}q_1^6 + q_1 q_2 + q_2^2. \tag{40}$$

A contour plot is shown in Fig. 3a. The global minimum is $f(0) = 0$ and there are two local minima at $x \approx \pm(-1.75, 0.87)^T$ where $f \approx 0.30$. In Fig. 3b we minimize (40) with the initial state $q_0 = (5, 5)^T$ and $p_0 = 0$. Note that RGD has a much faster convergence. It is interesting that in this example RGD uses a much smaller momentum factor, e.g. $\mu \approx 0.4$ whereas for CM and NAG $\mu \approx 0.92$.[10] In Fig. 3c we repeat the same experiment but

[10] The improvement of RGD by reducing μ was noticeable in this example, but did not always happen for other problems. In this example we verified that CM and NAG cannot improve their performance with such a small μ. Thus, RGD allows for a wider range of μ compared to CM and NAG.

initializing very close to one of the local minimizers. CM and NAG were unable to escape the local minimum, as opposed to RGD. In Fig. 3d we perform 500 experiments where the initial position is sampled uniformly in the range $-5 \leq q_{0,i} \leq 5$. Every algorithm converged sometimes to one of the local minima (not shown) and other times to the global minimum (shown in the plot). In all these cases, the convergence of RGD was drastically superior. Before runing these 500 trials, we tuned each algorithm over this region (see the Appendix for details).

Rosenbrock Function To consider a challenging problem in higher dimensions we minimize the nonconvex Rosenbrock function [46, 47]

$$f(q) \equiv \sum_{i=1}^{n-1} \Big(100 \big(q_{i+1} - q_i^2 \big)^2 + \big(1 - q_i \big)^2 \Big) \,. \tag{41}$$

We consider the $n = 100$ dimensional case since this was already studied in detail [48]. An illustration of this function is provided in Fig. 4a. Its landscape can be quite involved, for instance there are only two local minimizers, one global at $q^\star = (1, \ldots, 1)^T$ where $f(q^\star) = 0$, and one local near $q \approx (-1, 1, \ldots, 1)^T$ where $f \approx 3.99$. Moreover, there are (exponentially) many saddle points [48]. However, only two of these saddle points are hard to escape since they have a single negative eigenvalue of the Hessian with a magnitude quite small compared to the positive eigenvalues. These four stationary points account for 99.9% of the solutions found by Newton's method with random initializations [48]. Note that both minimizers lie on a flat, deep and narrow valley, which makes the optimization problem challenging. In Fig. 4b we show the convergence rate of the objective function when initializing close to the local minimum, which is already in the flat valley. This test was suggested by [49]. In Fig. 4c we initialize far away from this valley. In this case the improvement of RGD is even more prominent. In Fig. 4d we perform 200 Monte Carlo runs where the initial position is sampled uniformly at random in the range $-2.048 \leq q_{0,i} \leq +2.048$, which is the standard region where (41) is studied. We show the mean ($\pm$ standard deviation) of the states that converged to the global minimum. In other fewer instantiations the algorithms also converged to the local minimum (not shown). We note that NAG could be slightly faster than CM, but it proved to be more unstable. We tuned each algorithm over the entire region before running these experiments. In all cases, RGD always had a much faster convergence rate than the alternatives.

Deep Learning We now explore RGD on a nonconvex and high dimensional problem, namely an image classification problem with the MNIST dataset. We employ a standard LeNET-style convolutional neural network with 3 layers. Due to the large amount of training samples, an online optimization approach is used, essentially applying RGD to mini-batches

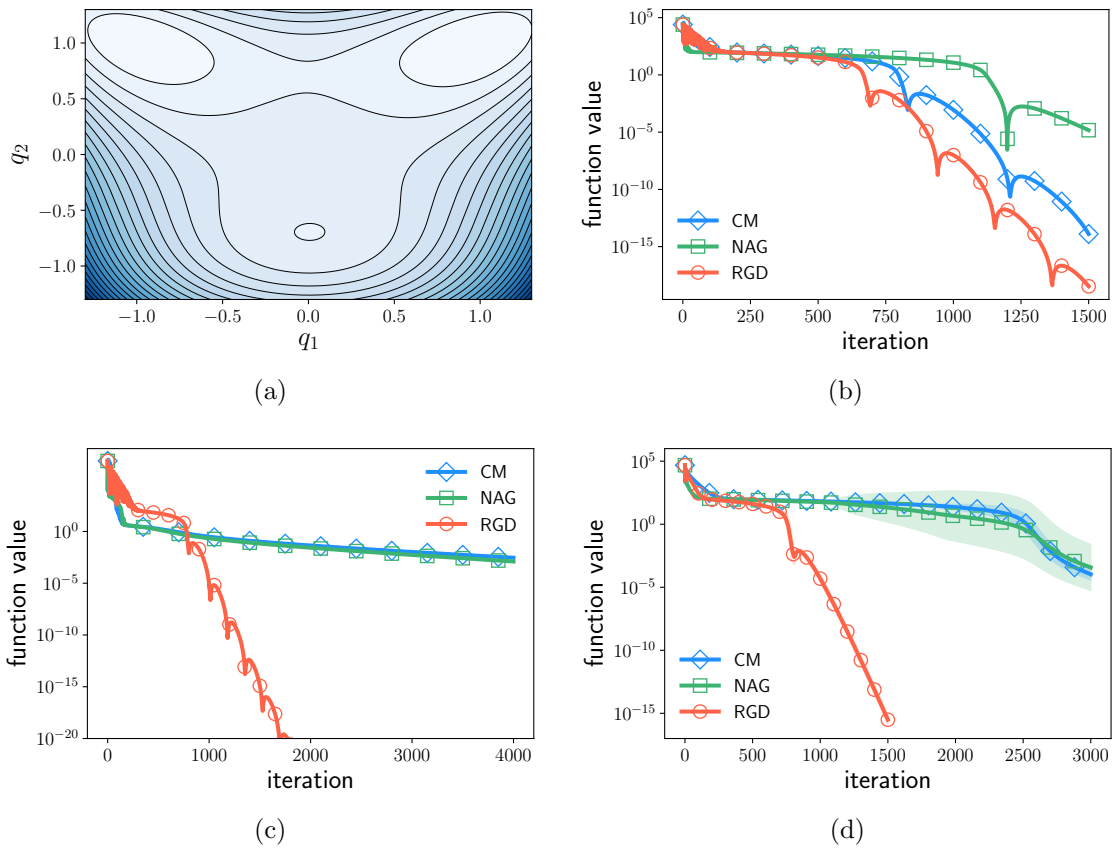


Figure 4: Rosenbrock function (41) in $\mathbb{R}^{100}$. (a) Contour plot where $q = (q_1, q_2, 1, \ldots, 1)^T$. Note the local and global minima (see text). (b) Minimization where $q_{0,2i} = 1$ and $q_{0,2i-1} = -1.2$, which is close to the local minimum. (c) Initial condition far from the minima, $q_{0,2i} = 5$ and $q_{0,2i-1} = -5$. (d) 200 experiments where $-2.048 \leq q_{0,i} \leq 2.048$ is chosen uniformly at random. Solid line is the mean, shaded region $\pm$ standard deviation.

and turning gradients into their stochastic counterparts. In this case, we refer to RGD as SRGD, which amounts to replacing the gradient in (37) by a stochastic gradient. In light of the adaptive normalization provided by popular methods in deep learning, we wish to verify whether SRGD has a similar behaviour. We thus compare SRGD to Adam [5] as a representative candidate. The parameters for all methods were tuned with a grid search (see Appendix B). Note that in this case we consider a fixed $\mu = 0.9$ for all methods. Also, due to the expensive computational time, we were only able to run these methods a couple of times. For this reason, it may be the case that the tuning procedure for SRGD is suboptimal since it has two extra parameters compared to the other methods. The results are shown in Fig. 5. Here momentum SGD is really the stochastic version of CM. We also used the stochastic counterpart of NAG, however its performance was nearly the same as SGD and

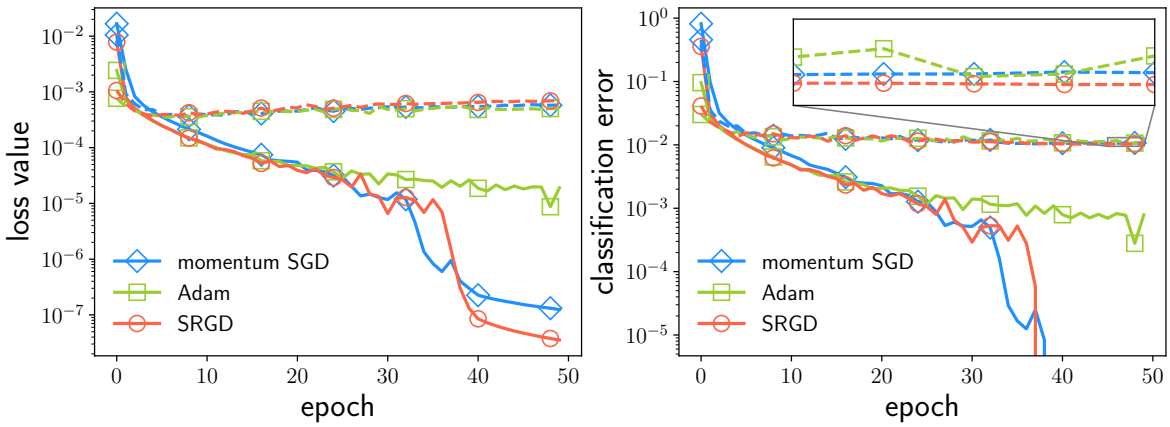


Figure 5: Feed forward CNN trained on MNIST. We use SGD with momentum, which the stochastic counterpart of (1). We also employed NAG, given by (2), but its performance was nearly the same as SGD and SRGD thus we omit these results. We mostly want to compare SRGD, the stochastic counterpart of (37), with Adam [5]. The plots show the mean over 6 random initializations (the standard deviation is very small thus we omit for a better visualization). The solid lines indicate results for the training set, while dashed lines are for the testing set. Note that SRGD starts as fast as Adam (faster than SGD in the beginning), however it does converge nearly to the same point as SGD towards the end.

SRGD, therefore we omit these results. It is important to note that in this example SRGD has nearly the same performance as SGD with momentum, and both outperform Adam. Recently, it was observed that adaptive methods such as Adam or RMSProp often find very different solutions compared with SGD, and with worse generalization error [7]. This interesting observation might point to the fact that RGD, being a natural generalization of CM, may not suffer from such limitations of ad-hoc adaptive gradient methods such as Adam, RMSProp and AdaGrad. One should keep in mind that in the worst case, RGD can always recover CM.

8 Discussion

We showed that well-known, and more importantly new, accelerated optimization methods can be obtained by appropriate discretizations of dissipative physical systems. Our approach focus on discretizations that preserve the intrinsic symplectic geometry of Hamiltonian systems, which is their most important property. We considered the classical mechanical system (6) whereby two interesting results were revealed. First, the classical momentum method (CM) is conformal symplectic (Corollary 4). Second, Nesterov's accelerated gradient (NAG) is not conformal symplectic (Theorem 5). These are interesting results that provide

a deeper understanding about well-known optimization methods from the perspective of continuous dynamical systems. We gave strong and general arguments that conformal symplectic integrators can preserve the rates of the associated continuous dynamical system up to a negligible error (see Section 5). Moreover, by considering a *relativistic extension* of the classical system we obtained a principled generalization of CM, that we call relativistic gradient descent (RGD) and is given by the simple updates (37). RGD automatically includes normalization of the momentum, is conformal symplectic, and may provide improvements regarding convergence rates and stability, as illustrated in our numerical experiments.

Being a simulation of a relativistic system, RGD operates on a different space compared to its classical predecessor (Fig. 1) where there exists a maximum speed limit c. In practice, it is important to tune the mass m and c to be able to incorporate relativistic effects into the dynamics; otherwise RGD works in a classical regime and is essentially equivalent to CM. We expect RGD to improve over CM on difficult optimization problems having regions with high curvature, so that instabilities can be controlled by tuning c while allowing m to decrease. A more complete study about its convergence rates and stability is a challenging, and interesting, open problem that we leave for future work.

Acknowledgments

This work was supported by grants ARO MURI W911NF-17-1-0304 and NSF 1447822.

A Physical Intuition on Parameter Tuning

Here we provide some intuition on parameter tuning for CM (1), NAG (2) and RGD (37). The following observations are based on the connections between CM and NAG with the classical system (6). Moreover, RGD is related to to the relativistic system (36) which is a generalization of this classical system. We thus make the following observations:

- The parameter γ controls the amount of dissipation. Note that we automatically have the momentum factor $\mu = e^{-\gamma h} \in (0, 1)$. Large γ, i.e. small μ, brings more friction which tends to make the system slower. Thus, if the objective function f is "flat" in some given region, a small γ (large μ) is desirable. However, in a region where f has high curvature the system can exhibit strong oscillations, which can be controlled by a larger γ and improve convergence. Thus, in this case a large γ (small μ) is advantageous. A good empirical guideline for the classical methods, i.e. CM and NAG, is to look for $\mu \in [0.9, 1)$. In most examples we observed that this also holds

for RGD, however we did find cases where RGD had a considerable improvement with a much smaller μ, e.g. $\mu \approx 0.5$. Thus, RGD may benefit from overdamping in some problems.

- The mass m controls the inertia of the system, i.e. how sensitive the system responds to to the external force $F = -\nabla f$. Light particles tend to gain more acceleration and travel faster. Thus, in general we expect that small m will speedup the algorithm. One should be careful, however, since when m is too small, updates such as (24b) may become unstable and diverge. Also, in special relativity only massless particles can travel at the speed c. Thus, it is desirable to make m as small as possible in RGD (37).

- Naturally, by increasing the discretization stepsize $h > 0$ the algorithm gives "larger strides", i.e. it simulates the continuous time $t_k = kh$ with fewer iterations. However, the stability of the algorithm strongly rely on h and it may become unstable, especially in regions of high curvature of the function f, so h must be small in these cases. On the other hand, when f is nearly flat, a large h might work fine and speedup the algorithm.

- Note that the above physical intuition is consistent with the formulas (32) and (33).

- Finally, since RGD is an extension of CM, where the latter is recovered in the limit $c \to \infty$, we expect improvements only when relativistic effects are into play. Therefore, after tuning CM (which has $m = h$) to gain improvements with RGD one should look for smaller values of m and potentially larger values of the stepsize h. Under these conditions, the classical algorithm would break down, however with RGD such instabilities may be controlled by tuning c. Thus, a good rule of thumb is to first tune CM to obtain intuition on μ and h, and then starts RGD with these parameters and a large enough c. Then one progressively decreases $m < h$ and also c. Ideally, all four parameters must be tuned together because the value of the momentum μ might also be quite different for RGD in comparison to CM and NAG.

B Numerical Details

Except for the deep learning experiment of Fig. 5, in the other experiments we use an exhaustive random search on parameter space. We use the same number of Monte Carlo runs for each algorithm, even though RGD has two extra parameters compared to CM and NAG. Also, we always set the initial momentum to zero, $p_0 = 0$.

Quadratic Functions For the experiment in Fig. 2a we performed 50 Monte Carlo runs, where for each sample of A we tune each algorithm through a random search on parameter

algorithm	stepsize	momentum μ	mass m	speed of light c
CM	$[10^{-2}, 0.8]$	$[0.8, 0.999]$		
NAG	$[10^{-3}, 0.5]$	$[0.8, 0.999]$		
RGD	$[10^{-3}, 0.5]$	$[0.6, 0.95]$	$[10^{-4}, 10^{-2}]$	$[10^{3}, 10^{6}]$

Table 1: Range for parameters search in the experiment of Fig. 2a.

algorithm	stepsize	momentum μ	mass m	speed of light c
CM	$[10^{-4}, 10^{-2}]$	$[0.6, 0.95]$		
NAG	$[10^{-4}, 10^{-2}]$	$[0.6, 0.95]$		
RGD	$[10^{-4}, 8 \cdot 10^{-3}]$	$[0.6, 0.8]$	$[10^{-6}, 10^{-4}]$	$[10^{3}, 10^{5}]$

Table 2: Range for parameters search in the experiment of Fig. 2b.

space (uniformly). The ranges are shown in Table 1. We run each algorithm 150 times and for 200 iterations, and choose the parameters which give the lowest objective function value. For the experiments in Fig. 2b we tune the algorithms over the whole region. We do this by sampling 50 points over this region, then we tune each algorithm for each of these points. In this case, the ranges are indicated in Table 2, where again we perform a random search (uniformly). We run each algorithm 200 times and for 150 iterations. Then, for each algorithm, we choose the mean value of each parameter. Once the parameters for each algorithm are fixed, we perform 200 experiments to display Fig. 2b.

Camelback Function For the experiment in Fig. 3b we use a random search on the ranges indicated in Table 3. We perform 1500 Monte Carlo runs for each method. We follow the same procedure for Fig. 3c. For the experiment in Fig. 3d we tune each algorithm for the entire region. We do this by tuning each algorithm over 100 sampled points q_0. We perform 100 trials for each algorithm with 100 iterations. The final parameters are the mean of these parameters. We then perform 500 Monte Carlo runs to obtain the results of Fig. 3d.

algorithm	stepsize	momentum μ	mass m	speed of light c
CM	$[10^{-5}, 10^{-3}]$	$[0.8, 0.999]$		
NAG	$[10^{-5}, 10^{-3}]$	$[0.8, 0.999]$		
RGD	$[10^{-5}, 8 \cdot 10^{-3}]$	$[0.3, 0.8]$	$[10^{-6}, 10^{-4}]$	$[10^{3}, 10^{5}]$

Table 3: Range for parameters search in the experiments of Fig. 3b and Fig. 3c.

Rosenbrock Function For the experiments in Figs. 4b/4c we use mostly the same range as in Table 3, except for the momentum factor which is searched in the range $\mu \in [0.9, 0.98]$. We perform 500 Monte Carlo runs with each algorithm runing for 1200 iterations. To tune over the region of Fig. 4d we perform 20 trials, sampling the initial state uniformly in the range $-2.048 \leq q_{0,i} \leq +2.048$, and tuning each algorithm where we look for parameters in the range

$$\epsilon \in [2 \times 10^{-4}, 4 \times 10^{-4}], \qquad \mu \in [0.94, 0.98], \tag{42}$$

for CM and NAG, while

$$h \in [10^{-5}, 10^{-4}], \quad \mu \in [0.93, 0.97], \quad m \in [4 \times 10^{-7}, 10^{-6}], \quad c \in [1 \times 10^{4}, 9 \times 10^{4}], \tag{43}$$

for RGD. After tuning over this region we fix the parameters to be the mean of the obtained results. Then, with these final parameters, we run 200 experiments with different initial states to obtain the results in Fig. 4d.

Deep Learning For the experiment of Fig. 5 we consider the standard stochastic gradient descent (SGD) with momentum, which is the stochastic counterpart of CM (1). Analogously, we consider a stochastic version of NAG (2), referred to as SNAG. For SRGD, the stochastic version of (37), we optimize over h, m and c. The momentum factor $\mu = 0.9$ is kept fixed for all methods. We tune the parameters through a grid search. The results are shown in Table 4.

algorithm	stepsize	momentum μ	mass m	speed of light c
Momentum SGD	8×10^{-2}	0.9		
SNAG	1.53×10^{-2}	0.9		
SRGD	2×10^{-3}	0.9	1.170×10^{-1}	10^{9}

Table 4: Parameters for the deep learning experiment of Fig. 5.

For Adam [5] we use the following parameters:

$$\epsilon = 1.6 \times 10^{-3}, \quad \beta_1 = 0.90, \quad \beta_2 = 0.999. \tag{44}$$

In this experiment, we use a standard LeNET convolutional neural network with 3 layers.

References

[1] B. T. Polyak. Some Methods of Speeding Up the Convergence of Iteration Methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[2] Y. Nesterov. A Method of Solving a Convex Programming Problem with Convergence Rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[3] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the Importance of Initialization and Momentum in Deep Learning. In *Int. Conf. Machine Learning*, 2013.

[4] 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, 2012.

[5] D. P. Kingma and J. L. Ba. Adam: A Method for Stochastic Optimization. In *Int. Conf. Learning Representations*, 2015.

[6] J. Duchi, E. Hazan, and Y. Singer. Adaptive Subgradient Methods of Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2017.

[7] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The Marginal Value of Adaptive Gradient Methods in Machine Learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[8] W. Su, S. Boyd, and E. J. Candès. A Differential Equation for Modeling Nesterov's Accelerated Gradient Method: Theory and Insights. *Journal of Machine Learning Research*, 17(153):1–43, 2016.

[9] A. Wibisono, A. C. Wilson, and M. I. Jordan. A Variational Perspective on Accelerated Methods in Optimization. *Proceedings of the National Academy of Sciences*, 113(47):E7351–E7358, 2016.

[10] W. Krichene, A. Bayen, and P. L. Bartlett. Accelerated Mirror Descent in Continuous and Discrete Time. *Advances in Neural Information Processing Systems*, 28, 2015.

[11] H. Attouch, Z. Chbani, J. Peypouquet, and P. Redont. Fast Convergence of Inertial Dynamics and Algorithms with Asymptotic Vanishing Viscosity. *Mathematical Programming*, pages 1–53, 2016.

[12] H. Attouch, A. Cabot, and M-O. Czarnecki. Asymptotic Behaviour of Nonautonomous Monotone and Subgradient Evolution Equations. *Trans. Amer. Math. Soc.*, pages 755–790, 2017.

[13] H. Attouch and A. Cabot. Convergence of Damped Inertial Dynamics Governed by Regularized Maximally Monotone Operators. *J. Differential Equations*, 264:7138–7182, 2018.

[14] J. Zhang, A. Mokhtari, S. Sra, and A. Jadbabaie. Direct Runge-Kutta Discretization Achieves Acceleration. *Advances in Neural Information Processing Systems*, 31, 2018.

[15] B. Shi, S. S. Du, M. I. Jordan, and W. J. Su. Understanding the Acceleration Phenomenon via High-Resolution Differential Equations. arXiv:1810.08907 [math.OC], 2018.

[16] L. F. Yang, R. Arora, V. Braverman, and T. Zhao. The Physical Systems Behind Optimization Algorithms. *32nd Conference on Neural Information Processing Systems*, 2018.

[17] G. França, D. P. Robinson, and R. Vidal. ADMM and Accelerated ADMM as Continuous Dynamical Systems. *International Conference on Machine Learning*, 2018.

[18] G. França, D. P. Robinson, and R. Vidal. A Dynamical Systems Perspective on Nonsmooth Constrained Optimization. arXiv:1808.04048 [math.OC], 2018.

[19] R. I. McLachlan and G. R. W. Quispel. Geometric Integrators for ODEs. *J. Phys. A: Math. Gen.*, 39:5251–5285, 2006.

[20] R. Quispel and R. McLachlan. Geometric Numerical Integration of Differential Equations. *J. Phys. A: Math. Gen.*, 39, 2006.

[21] R. de Vogelaere. Methods of Integration which Preserve the Contact Transformation Property of the Hamiltonian Equations. Tech. Rep. 4, Department of Mathematics, University of Notre Dame, 1956.

[22] R. D. Ruth. A Canonical Integration Technique. *IEEE Trans. on Nuclear Science*, NS-30(4), 1983.

[23] P. J. Channell. Symplectic Integration Algorithms. Internal report AT-6:ATN-83-9, Los Alamos National Laboratory, 1983.

[24] P. J. Channell. Initial Value and Eigenvalue Problems for Field Equations Using Symplectic Integration Algorithms. Internal report AT-6:ATN-83-18, Los Alamos National Laboratory, 1983.

[25] F. Kang. Difference Schemes for Hamiltonian Formalism and Symplectic Geometry. *J. Comput. Math.*, 4(279), 1986.

[26] F. Neri. Lie Algebras and Canonical Integration. Technical Report, University of Maryland, Department of Physics, 1987.

[27] C. R. Menyuk. Some Properties of the Discrete Hamiltonian Methods. *Physica D*, 11D(109), 1984.

[28] E. Forest. Geometric Integration for Particle Accelerators. *J. Phys. A: Math. Gen.*, 39:5321–5377, 2006.

[29] P. J. Channell and C. Scovel. Symplectic Integration of Hamiltonian Systems. *Nonlinearity*, 3:231–259, 1990.

[30] M. Betancourt, M. I. Jordan, and A. C. Wilson. On Symplectic Optimization. arXiv:1802.03653 [stat.CO], 2018.

[31] R. McLachlan and M. Perlmutter. Conformal Hamiltonian Systems. *Journal of Geometry and Physics*, 39:276–300, 2001.

[32] A. Bhatt, D. Floyd, and B. E. Moore. Second Order Conformal Symplectic Schemes for Damped Hamiltonian Systems. *Journal of Scientific Computing*, 66:1234–1259, 2016.

[33] X. Lu, V. Perrone, L. Hasenclever, Y. W. Teh, and S. J. Vollmer. Relativistic monte carlo. *20th International Conference on Articial Intelligence and Statistics (AISTATS)*, 2017.

[34] C. J. Maddison, D. Paulin, Y. W. Teh, B. O'Donoghue, and A. Doucet. Hamiltonian Descent Methods. arXiv:1809.05042 [math.OC], 2018.

[35] S. Livingstone, M. F. Faulkner, and G. O. Roberts. Kinetic Energy choice in Hamiltonian/Hybrid Monte Carlo. arXiv:1706.02649 [stat.CO], 2017.

[36] R. I. McLachlan and G. R. W. Quispel. What Kinds of Dynamics are There? Lie Pseudogroups, Dynamical Systems, and Geometric Integration. *Nonlinearity*, 14:1689–1706, 2001.

[37] U. Dessler. Symmetry Property of the Lyapunov Spectra of a Class of Dissipative Dynamical Systems with Viscous Damping. *Phys. Rev. A*, 38:2103, 1988.

[38] S. Marò and A. Sorrentino. Aubry-Mather Theory for Conformally Symplectic Systems. *Commun. Math. Phys.*, 354:775–808, 2017.

[39] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration*. Springer, 2006.

[40] E. Hairer and G. Wanner. *Euler Methods, Explicit, Implicit, Symplectic*, pages 451–455. Springer Berlin Heidelberg, 2015.

[41] B. Leimkuhler and S. Reich. *Simulating Hamiltonian Dynamics*. Cambridge University Press, 2004.

[42] E. Hairer and Ch. Lubich. The Life-Span of Backward Error Analysis for Numerical Integrators. *Numer. Math.*, 76:441–462, 1997.

[43] G. Benettin and A. Giorgilli. On the Hamiltonian Interpolation of Near-to-the-Identity Symplectic Mappings with Application to Symplectic Integration Algorithms. *Journal of Statistical Physics*, 74:1117–1143, 1994.

[44] L. D. Landau and E. M. Lifshitz. *The Classical Theory of Fields*. Butterworth-Heinemann, 1976.

[45] F. H. Branin. Widely Convergent Method for Finding Multiple Solutions of Simultaneous Nonlinear Equations. *IBM Journal of Research and Development*, 16(5):504–522, 1972.

[46] H. H. Rosenbrock. An Automatic Method for Finding the Greatest or Least Value of a Function. *The Computer Journal*, 3(3):175–184, 1960.

[47] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[48] S. Kok and C. Sandrock. Locating and Characterizing the Stationary Points of the Extended Rosenbrock Function. *Evolutionary Computation* , 17(3):437–453, 2009.

[49] J. J. Moré, B. S. Garbow, and K. E. Hillstrom. Testing Unconstrained Optimization Software. *ACM Trans. Math. Software*, 7(1):17–41, 1981.