Dynamical, Symplectic and Stochastic Perspectives on Gradient-Based Optimization

Michael I. Jordan
University of California, Berkeley

March 3, 2018

Our topic is the relationship between dynamical systems and optimization. This is a venerable, vast area in mathematics, counting among its many historical threads the study of gradient flow and the variational perspective on quantities such as risk, the number of data points and the model complexity, while also taking into account computational resource constraints such as number of processors, the communication bandwidth and the degree of asynchrony. Fundamental understanding of such tradeoffs seems likely to emerge through the development of lower bounds—by establishing notions of “best” one can strip away inessentials and reveal essential relationships. Here too optimization theory has been important. In a seminal line of research beginning in the 1970’s, Nemirovskii, Nesterov and others developed a complexity theory of optimization, establishing lower bounds on rates of convergence and discovering algorithms that achieve those lower bounds (Nemirovskii and Yudin, 1983; Nesterov, 1998). Moreover, the model of complexity was a relative one—an “oracle” is specified and algorithms can only use information that is available to the oracle. For example, it is possible to consider oracles that have access only to function values and gradients. Thus the dictum of practical computational efficiency can be imposed in a natural way in the theory.

Our focus is the class of optimization algorithms known as “accelerated algorithms” (Nesterov, 1998). These algorithms often attain the oracle lower-bound rates, although it is something of a mystery why they do so. We will argue that some of mystery is due to the historical focus in optimization on discrete-time algorithms and analyses. In optimization, the distinction between “continuous optimization” and “discrete optimization” refers to the configuration (“spatial”) variables. By way of contrast, our discussion will focus on continuous time. In continuous time we can give acceleration a mathematical meaning as a differential concept, as a change of speed along a curve. And we can pose the question of “what is the fastest rate?” as a problem in variational analysis; in essence treating the problem of finding the “optimal way to optimize” for a given oracle itself as a formal problem of optimization. Such a variational perspective also has the advantage of being generative—we can derive algorithms that achieve fast rates rather than requiring an analysis to establish a fast rate for a specific algorithm that is derived in an adhoc manner.

Working in continuous time forces us to face the problem of discretizing a continuous-time dynamical system, so as to derive an algorithm that can be implemented on a digital computer. Interestingly, we will find that symplectic integrators, which are widely used for integrating dynamics obtained from variational or Hamiltonian perspectives, are relevant in the optimization setting. Symplectic integration preserves the continuous symmetries of the underlying dynamical system, and this stabilizes the dynamics, allowing step sizes to be larger. Thus algorithms obtained from symplectic integration can move more quickly through a configuration space; this gives a geometric meaning to “acceleration.”

It is also of interest to consider continuous-time stochastic dynamics that are in some sense “accelerated.” The simplest form of gradient-based stochastic differential equation is the Langevin diffusion. The particular variant that has been studied in the literature is an *overdamped* diffusion that is an analog of gradient descent. We will see that by considering instead an *underdamped* Langevin diffusion, we will obtain a method that is more akin to accelerated gradient descent, and which in fact provably yields a faster rate than the overdamped diffusion.

The presentation here is based on joint work with co-authors Andre Wibisono, Ashia Wilson, Michael Betancourt, Chi Jin, Praneeth Netrapalli, Rong Ge, Sham Kakade, Niladri Chatterji and Xiang Cheng, as well as other co-authors who will be acknowledged in specific sections.

1 Lagrangian and Hamiltonian Formulations of Accelerated Gradient Descent

Given a continuously differentiable function f on an open Euclidean domain $\mathcal{X}$, and given an initial point $\mathbf{x}_0 \in \mathcal{X}$, gradient descent is defined as the following discrete dynamical system:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \nabla f(\mathbf{x}_k), \tag{1}$$

where $\eta > 0$ is a *step size* parameter.

When f is convex, it is known that gradient descent converges to the global optimum $\mathbf{x}^\star$, assumed unique for simplicity, at a rate of $O(1/k)$ (Nesterov, 1998). This means that after k iterations, the function value $f(\mathbf{x}_k)$ is guaranteed to be within a constant times $1/k$ of the optimum value $f^\star = f(\mathbf{x}^\star)$. This is a worst-case rate, meaning that gradient descent converges as least as fast as $O(1/k)$ across the function class of convex functions. The constant hidden in the $O(\cdot)$ notation is an explicit function of a complexity measure such as a Lipschitz constant for the gradient.

In the 1980's, a complexity theory of optimization was developed in which rates such as $O(1/k)$ could be compared to lower bounds for particular problem classes (Nemirovskii and Yudin, 1983). For example, an oracle model appropriate for gradient-based optimization might consider all algorithms that have access to sequences of gradients of a function, and whose iterates must lie in the linear span of the current gradient and all previous gradients. This model encompasses, for example, gradient descent, but other algorithms are allowed as well. Nemirovskii and Yudin (1983) were able to prove, by construction of a worst-case function, that no algorithm in this class can converge at a rate faster than $O(1/k^2)$. This lower bound is better than gradient descent, and it holds open the promise that some gradient-based algorithm can beat gradient descent across the family of convex functions. That promise was realized by Nesterov (1983), who presented the following algorithm, known as *accelerated gradient descent*:

$$\begin{aligned} \mathbf{y}_{k+1} &= \mathbf{x}_k - \eta \nabla f(\mathbf{x}_k) \\ \mathbf{x}_{k+1} &= (1 + \lambda_k)\mathbf{y}_{k+1} - \lambda_k \mathbf{y}_k, \end{aligned} \tag{2}$$

and proved that the algorithm converges at rate $O(1/k^2)$ for convex functions f. Here λ_k is an explicit function of the other problem parameters. We see that the acceleration involves two successive gradients, and the resulting dynamics are richer than those of gradient descent. In particular, accelerated gradient descent is not a descent algorithm—the function values can oscillate.

Nesterov's basic algorithm can be presented in other ways; in particular, we will also use a three-sequence version:

$$\begin{aligned}\mathbf{x}_k &= y_k + \lambda_k \mathbf{v}_k \\ \mathbf{y}_{k+1} &= \mathbf{x}_k - \eta \nabla f(\mathbf{x}_k) \\ \mathbf{v}_{k+1} &= \mathbf{y}_{k+1} - y_k. \end{aligned} \tag{3}$$

We note in passing that in both this version and the two-sequence version, the parameter λ_k is time-varying for some problem formulations and constant for others.

After Nesterov's seminal paper in 1983, the subsequent three decades have seen the development of a variety of accelerated algorithms in a wide variety of other problem settings. These include mirror descent, composite objective functions, non-Euclidean geometries, stochastic variants and higher-order gradient descent. Rates of convergence have been obtained for these algorithms, and these rates often achieve oracle lower bounds. Overall, acceleration has been one of the most productive ideas in modern optimization theory. See Nesterov (1998) for a basic introduction, and Bubeck et al. (2015); Allen-Zhu and Orecchia (2014) for examples of recent progress.

And yet the basic acceleration phenomenon has remained somewhat of a mystery. Its derivation and its analysis are often obtained only after lengthy algebraic manipulations, with clever but somewhat opaque upper bounds needed at critical junctures.

In Wibisono et al. (2016), we argue that this mystery is due in part to the discrete-time formalism that is generally used to derive and study gradient-based optimization algorithms. Indeed, the notion of "acceleration" seems ill-defined in a discrete-time framework; what does it mean to move more quickly along a sequence of discrete points? Such a notion seems to require an embedding in an underlying flow of time, such that acceleration can be viewed as a diffeomorphism. Moreover, if accelerated optimization algorithms are in some sense optimal, there must be something special about the curve that they follow in the configuration space, not merely the speed at which they move. Such a separate characterization of curve and speed also seem to require continuous time.

Wibisono et al. (2016) address these issues via a variational framework that aims to capture the phenomenon of acceleration in some generality. We review this framework in the remainder of this section, discussing the Lagrangian formulation that captures acceleration in continuous time, showing how this formulation gives rise to a family of differential equations whose convergence rates are the continuous-time counterpart of the discrete-time oracle rates. We highlight the problem of the numerical integration of these differential equations, setting up the symplectic integration approach that we discuss in Section 2.

We consider the general non-Euclidean setting in which the space $\mathcal{X}$ is endowed with a distance-generating function $h : \mathcal{X} \to \mathbb{R}$ that is convex and essentially smooth (i.e., h is continuously differentiable in $\mathcal{X}$, and $\|\nabla h(\mathbf{x})\|_* \to \infty$ as $\|\mathbf{x}\| \to \infty$). The function h can be used to define a measure of distance in $\mathcal{X}$ via its *Bregman divergence*:

$$D_h(\mathbf{y}, \mathbf{x}) = h(\mathbf{y}) - h(\mathbf{x}) - \langle \nabla h(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle. \tag{4}$$

The Euclidean setting is obtained when $h(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|^2$.

We use the Bregman divergence to construct a *Bregman kinetic energy* for a dynamical system. We do this by taking the Bregman divergence between a point $\mathbf{x}$ and its translation in the direction of the velocity $\mathbf{v}$ by a time-varying magnitude, $e^{-\alpha_t}$:

$$K(\mathbf{x}, \mathbf{v}, t) := D_h(\mathbf{x} + e^{-\alpha_t}\mathbf{v}, \mathbf{x}). \tag{5}$$

Using the definition in Eq. (4), we see that this kinetic energy can be interpreted as comparison between the amount that h changes under a finite translation, $h(\mathbf{x} + e^{-\alpha_t}\mathbf{v}) - h(\mathbf{x})$, versus an infinitesimal translation, $e^{-\alpha_t}\langle \nabla h(\mathbf{x}), \mathbf{v} \rangle$.

We now define a time-dependent potential energy, $U(\mathbf{x})$:

$$U(\mathbf{x}, t) := e^{\beta_t} f(\mathbf{x}), \tag{6}$$

and we subtract the potential energy from the kinetic energy to obtain the *Bregman Lagrangian*:

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \mathbf{v}, t) &:= e^{\alpha_t + \gamma_t}(K(\mathbf{x}, \mathbf{v}, t) - U(\mathbf{x}, t)) \\ &= e^{\alpha_t + \gamma_t}(D_h(\mathbf{x} + e^{-\alpha_t}\mathbf{v}, \mathbf{x}) - e^{\beta_t} f(\mathbf{x})). \end{aligned} \tag{7}$$

In this equation, the time-dependent factors α_t, β_t and γ_t are algorithmic degrees of freedom that allow the Bregman-Lagrangian framework to encompass a range of different algorithms.

Although α_t, β_t and γ_t can be set independently in principle, we define a set of *ideal scaling conditions* that reduce these three degrees of freedom to a single functional degree of freedom:

$$\begin{aligned} \dot{\beta}_t &\le e^{\alpha_t} \\ \dot{\gamma}_t &= e^{\alpha_t}. \end{aligned} \tag{8}$$

Wibisono et al. (2016) show that these conditions are needed to obtain differential equations whose rates of convergence are the optimal rates; see Theorem 1 below.

Given the Bregman Lagrangian, we use standard calculus of variations to obtain a differential equation whose solution is the path that optimizes the time-integrated Bregman Lagrangian. In particular, we form the Euler-Lagrange equations:

$$\frac{d}{dt}\left\{\frac{\partial \mathcal{L}}{\partial \mathbf{v}}(\mathbf{x}_t, \dot{\mathbf{x}}_t, t)\right\} - \frac{\partial \mathcal{L}}{\partial \mathbf{x}}(\mathbf{x}_t, \dot{\mathbf{x}}_t, t) = 0, \tag{9}$$

a computation which is easily done based on Eq. (7). Using the ideal scaling conditions in Eq. (8), the result simplifies to the following master differential equation:

$$\ddot{x}_t + (e^{\alpha_t} - \dot{\alpha}_t)\dot{x}_t + e^{2\alpha_t+\beta_t}\Big[\nabla^2 h(x_t + e^{-\alpha_t}\dot{x}_t)\Big]^{-1}\nabla f(x_t) = 0. \tag{10}$$

We see that the equation is second order and non-homogeneous. Moreover, the gradient $\nabla f(x_t)$ appears as a force, modified by geometric terms associated with the Bregman distance-generating function h. As we will discuss below, this equation is a general form of Nesterov acceleration in continuous time.

It is straightforward to obtain a convergence rate for the master differential equation. We define the following Lyapunov function:

$$\mathcal{E}_t \;=\; D_h\left(\mathbf{x}^\star,\, \mathbf{x}_t + e^{-\alpha_t}\dot{\mathbf{x}}_t\right) + e^{\beta_t}(f(\mathbf{x}_t) - f^\star)). \tag{11}$$

Taking a first derivative with respect to time, and asking that this derivative be less than or equal to zero, we immediately obtain a convergence rate, as documented in the following theorem, whose proof can be found in Wibisono et al. (2016).

Theorem 1. *If the ideal scaling in Eq. (8) holds, then solutions to the Euler-Lagrange equation Eq. (10) satisfy*

$$f(\mathbf{x}_t) - f^\star \leq O(e^{-\beta_t}).$$

For further explorations of Lyapunov-based analysis of accelerated gradient methods, see Wilson et al. (2016).

Wibisono et al. (2016) studied a subfamily of Bregman Lagrangians with the following choice of parameters, indexed by a parameter $p > 0$:

$$\begin{aligned}\alpha_t &= \log p - \log t\\ \beta_t &= p\log t + \log C\\ \gamma_t &= p\log t,\end{aligned} \tag{12}$$

where $C > 0$ is a constant. This choice of parameters satisfies the ideal scaling condition in Eq. (8). The Euler-Lagrange equation, Eq. (10), reduces in this case to:

$$\ddot{x}_t + \frac{p+1}{t}\dot{x}_t + Cp^2t^{p-2}\left[\nabla^2 h\left(x_t + \frac{t}{p}\dot{x}_t\right)\right]^{-1}\nabla f(x_t) = 0, \tag{13}$$

and, by Theorem 1, it has an $O(1/t^p)$ rate of convergence.

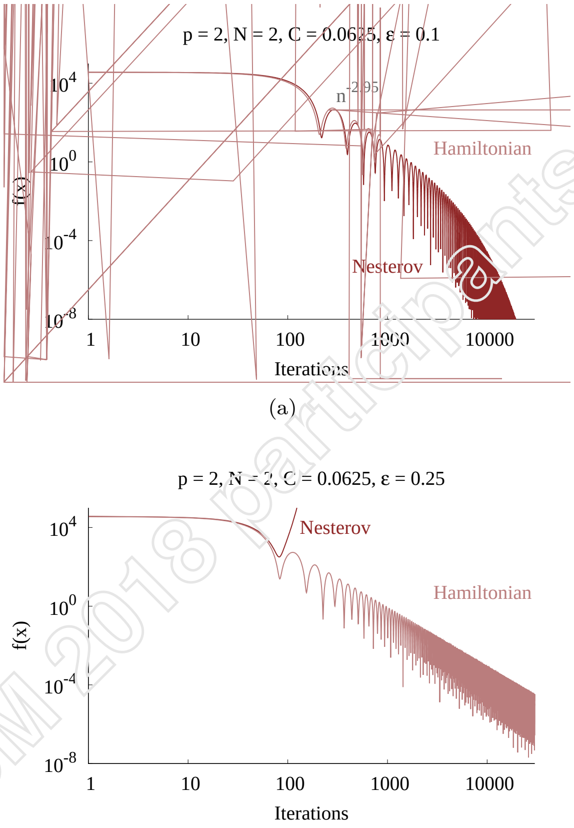
The case $p = 2$ of the Eq. (13) is the continuous-time limit of Nesterov's accelerated mirror descent (Krichene et al., 2015), the case $p = 3$ is the continuous-time limit of Nesterov's accelerated cubic-regularized Newton's method (Nesterov and Polyak, 2006). In the Euclidean case, when the Hessian $\nabla^2 h$ is the identity matrix, we recover the following differential equation:

$$\ddot{x}_t + \frac{3}{t}\dot{x}_t + \nabla f(x_t) = 0, \tag{14}$$

which was first derived by Su et al. (2016) as the continuous-time limit of the basic Nesterov accelerated gradient descent algorithm in Eq. (2).

The Bregman Lagrangian has several mathematical properties that give significant insight into aspects of the acceleration phenomenon. For example, the Bregman Lagrangian is closed under time dilation. This means that if we take an Euler-Lagrange curve of a Bregman Lagrangian and reparameterize time so that we travel the curve at a different speed, then the resulting curve is also the Euler-Lagrange curve of another Bregman Lagrangian, with appropriately modified parameters. Thus, the entire family of accelerated methods correspond to a single curve in spacetime and can be obtained by speeding up (or slowing down) any single curve. As suggested e iu Lagraes d p wing pr

`B H `€ °



(a)

(b)

Figure 1: (a) When appropriately tuned, both the leapfrog integrator and the three-sequence Nesterov algorithm simulate the same latent Bregman dynamics and hence achieve similar convergence rates, here approximately $\mathcal{O}(k^{-2.95})$. (b) Given a larger step size, the symplectic integrator remains stable and thus converges more quickly, whereas the three-sequence Nesterov algorithm becomes unstable.

more stable than other integration schemes, such that it is possible to use larger step sizes in the discrete-time system. It is this latter fact that suggests a role for symplectic integrators in the integration of the differential equations associated with accelerated optimization methods. This idea has been pursued in recent work by Betancourt et al. (2018), whose results we review in this section.

While symplectic integrators can be obtained from a Lagrangian framework, they are most naturally obtained from a Hamiltonian framework. We thus begin by transforming the Bregman Lagrangian into a Bregman Hamiltonian. This is readily done via a Legendre transform, as detailed in Wibisono et al. (2016) and Betancourt et al. (2018). The resulting Hamiltonian is as follows:

$$H(\mathbf{x},\mathbf{r},t) = e^{\alpha(t)+\gamma(t)}\left(D_{h^*}(e^{-\gamma(t)}\mathbf{r} + \frac{\partial h}{\partial \mathbf{x}}(\mathbf{r}), \frac{\partial h}{\partial \mathbf{x}}(\mathbf{x})) + e^{\beta(t)}f(\mathbf{x})\right), \qquad (16)$$

where

$$D_{h^*}(\mathbf{r},\mathbf{s}) = h^*(\mathbf{r}) - h^*(\mathbf{s}) - \frac{\partial h^*}{\partial \mathbf{r}}(\mathbf{s}) \cdot (\mathbf{r} - \mathbf{s}),$$

and where h^* is the Fenchel conjugate:

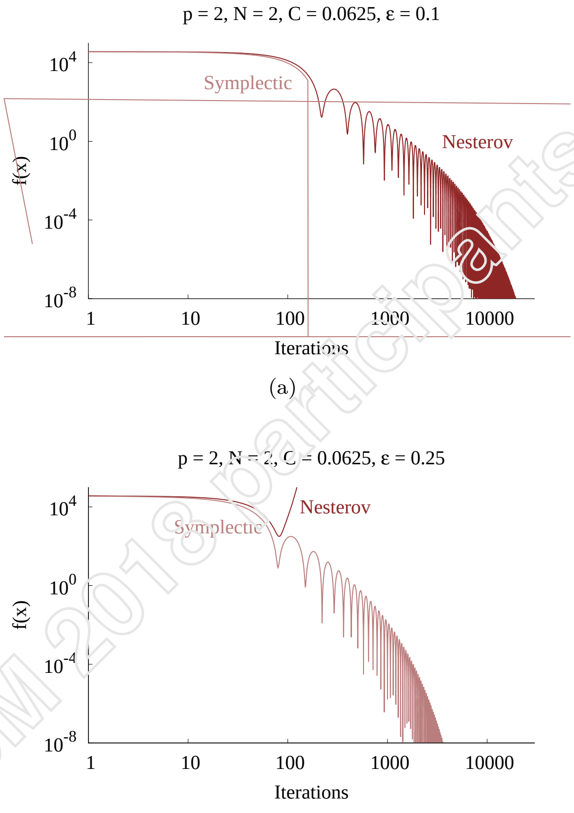
$$h^*(r) \underset{\mathbf{v}\in T\mathcal{X}}{=} \sup\left(\langle \mathbf{r}, \mathbf{v}\rangle - h(v)\right).$$

Given the Bregman Hamiltonian in Eq. (16), Betancourt et al. (2018) follow a standard sequence of steps to obtain a symplectic integrator. First, the Bregman Hamiltonian is time-varying, and it is thus lifted into a time-invariant Hamiltonian on an augmented configuration space that includes time as an explicit variable and includes a conjugate energy variable in the phase space. Second, the Hamiltonian is split into a set of component Hamiltonians, each of which can be solved analytically (or nearly so via simple numerical methods). Third, the component dynamics are composed symmetrically to form the full dynamics. In particular, Betancourt et al. (2018) illustrate how to form a symmetric leapfrog integrator (a particular kind of symplectic integrator) for the Bregman Hamiltonian. They prove that the error between this integrator and the true dynamics is of order $O(\epsilon^2)$, where ϵ is the step size in the discretization.

Betancourt et al. (2018) also present empirical results for a quadratic objective function, $f(x) = \langle \Sigma^{-1}x, x\rangle$, on a 50-dimensional Euclidean space, where

$$\Sigma_{ij} = \rho^{|i-j|},$$

and $\rho = 0.9$. This experiment was carried out in the setting of Eq. (12), for various choices of p, C and N. Representative results are shown in Figure 1(a), which compare the leapfrog integrator with the three-sequence version of Nesterov acceleration from Eq. (15). Here we see that both approaches yield stable, oscillatory dynamics whose asymptotic convergence rate is approximately $\mathcal{O}(k^{-2.95})$. Moreover, as shown in Figure 1(b), the symplectic integrator remains stable when a larger step size is chosen, whereas the three-sequence Nesterov algorithm becomes unstable.



(a)

(b)

Figure 2: (a) By incorporating gradient flow into the leapfrog integration of the Bregman Hamiltonian dynamics we recover the same asymptotic exponential convergence near the minimum of the objective exhibited by the dynamical Nesterov algorithm. (b) These modified Hamiltonian dynamics remain stable even as we increase the step size, allowing for more efficient computation and the advantageous asymptotic behavior.

Interestingly, however, while the initial rate is the same, the three-sequence algorithm exhibits exponential convergence near the optimum. This behavior does not hold in general (e.g., Betancourt et al. (2018) show that it does not hold for quartic functions), but it is nonetheless an interesting feature of the three-sequence method in the case of quadratic objectives. Betancourt et al. (2018) show that it arises from an implicit gradient flow that is a side effect of the three-sequence discretization. Moreover, they note that it is possible to mimic this flow within the symplectic integrator. When this is done the results for the quadratic objective are as shown in Figure 2(a). We see that a symplectic integrator that incorporates gradient flow and the three-sequence integrator yield convergence profiles that are essentially equivalent in this case.

But it is also true that the symplectic approach is topologically more stable than the three-sequence method, a fact which is revealed if one chooses a more aggressive step size. This is exhibited in Figure 2(b), where the symplectic integrator converges while the three-sequence Nesterov method diverges when the step size is increased.

In summary, this section has exhibited a connection between symplectic integration and the acceleration phenomenon in optimization. When the latter is construed as a continuous-time phenomenon, symplectic integration appears to provide an effective and flexible way to obtain discrete-time approximations. Much remains to be done, however, to tighten this link. In particular, we would like to obtain necessary and sufficient conditions for stable integrators that achieve oracle rates, and it is not yet clear what role symplectic geometry will play in uncovering those conditions.

3 Acceleration and the Escape from Saddle Points in Nonconvex Optimization

In this section we turn to *nonconvex* optimization. Although the general nonconvex setting harbors many intractable problems about which little can be said regarding computational or statistical efficiency, it turns out that for a wide range of problems in statistical learning, there is sufficient mathematical structure present in the nonconvex setting that useful mathematical results can be obtained. Indeed, in many cases the ideas and algorithms from convex optimization—suitably modified—can be carried over to the nonconvex setting. In particular, for gradient-based optimization, the same algorithms that perform well in the convex setting also tend to yield favorable performance in the nonconvex setting. In this sense, convex optimization has served as a laboratory for nonconvex optimization, in addition to having many natural applications of its own.

A useful first foothold on nonconvex optimization is obtained by considering the criterion of *first-order stationarity.* Given a differentiable function $f : \mathcal{X} \to \mathbb{R}$, on some well-behaved Euclidean domain $\mathcal{X}$ of dimension d, we define first-order stationary points to be those points $\mathbf{x} \in \mathcal{X}$ where the gradient vanishes: $\|\nabla f(\mathbf{x})\| = 0$. Although first-order stationary points can in general be associated with many kinds of topological singularity, for many statistical learning problems it suffices to consider the categorization into points that are global minima, local minima, local maxima and saddle points. Of these, local maxima are rarely viewed as problematic—simple modifications of gradient descent, such as stochastic perturbation, can suffice to ensure that algorithms do not get stuck at local maxima. Local minima have long been viewed as the core concern in nonconvex optimization for statistical learning problems. Recent work has shown, however, that in a wide range of nonconvex statistical learning problems, local minima are provably absent, or, in empirical studies, even when local minima are present they do not appear to be discovered by gradient-based algorithms. Such results have been obtained for smooth semidefinite programs (Boumal et al., 2016), matrix completion (Ge et al., 2016), synchronization and MaxCut (Bandeira et al., 2016; Mei et al., 2017), multi-layer neural networks (Choromanska et al., 2014; Kawaguchi, 2016), matrix sensing (Bhojanapalli et al., 2016) and robust principal components analysis (Ge et al., 2017).

As for global minima, while they are unambiguously the desirable end states for optimization algorithms, when there are multiple global minima it will generally be necessary to impose additional criteria (e.g., statistical) to single out preferable global minima, and to ask that an optimization algorithm respect this preference. We will not discuss these additional criteria here.

It remains to consider saddle points. Naively one might view these as akin to local maxima, in the sense that it is plausible that a simple perturbation could suffice for a gradient-based algorithm to roll down a direction of negative curvature. Such an argument has support from a recent theoretical result: Lee et al. (2016) have shown that under regularity conditions gradient descent will converge asymptotically and almost surely to a (local) minimum and thus avoid saddle points. In particular, a gradient-based algorithm that is initialized at a random point in $\mathcal{X}$ will avoid any and all saddle points in the asymptotic limit. While this result helps to emphasize the strength of gradient descent, it is of limited practical in that it is asymptotic (providing no rate of convergence); moreover, critically, it does not provide any insight into the rate of escape of saddle points as a function of dimension. While under suitable regularity all directions are escape directions for local maxima, it could be that only one direction is an escape direction for a saddle point. The computational burden of finding that direction could be significant; perhaps exponential in dimension. Given that modern statistical learning problems can involve many hundreds of thousands or millions of dimensions, such a burden would be fatal.

We thus focus our discussion on saddle points. To tie the discussion here to the discussion of the previous section, we take a dynamical systems perspective and study the extent to which acceleration (second-order dynamics) is able to improve the rate of escape of saddle points in gradient-based optimization. Intuitively, there is a narrow region around a saddle point in which the flow is principally directed towards the saddle point, and it seems plausible that an accelerated algorithm is able to bypass such a region more effectively than a non-accelerated algorithm. Whether this is actually true has been an open

22 i3;2+ r2

aexclude;

question in the literature.

To understand how dynamics and geometry interact in the neighborhood of saddle points, it is necessary to go beyond first-order stationarity, which lumps saddle points together with local minima, and to impose a condition that excludes saddle points. We define a *second-order stationary point* to be a point $\mathbf{x} \in \mathcal{X}$ such that $\|\nabla f(\mathbf{x})\| = 0$ and $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \geq 0$. This definition includes local minima, but it also allows degenerate saddle points, in which the smallest eigenvalue of the Hessian is zero, and so we also define a *strict saddle point* to be a point $\mathbf{x} \in \mathcal{X}$ for which $\lambda_{\min}(\nabla^2 f(\mathbf{x})) < 0$. These two definitions jointly allow us to separate local minima from most saddle points. In particular, if all saddle points are strict, then an algorithm that converges to a second-order stationary point necessarily converges to a local minimum. (See Ge et al. (2015) for further discussion.)

sr 5 6th saBob eometry

It turns out that these requirements are reasonable in practical applications. Indeed, it it has been shown theoretically that all saddle points are strict in many of the nonconvex problems mentioned earlier, including tensor decomposition, phase retrieval, dictionary learning and matrix completion (Ge et al., 2015; Sun et al., 2016a,b; Bhojanapalli et al., 2016; Ge et al., 2016) Coupled with the fact (mentioned above) that there is a single global minimum in such problems, we see that an algorithm that converges to a second-order stationary point will actually converge to a global minimum.

To obtain rates of convergence, we need to weaken the definitions of stationarity to allow an algorithm to arrive in a ball of size $\epsilon > 0$ around a stationary point, for varying ϵ. We define an *ϵ-first-order stationary point* as a point $\mathbf{x} \in \mathcal{X}$ such that $\|\nabla f(\mathbf{x})\| \leq \epsilon$. Similarly we define an *ϵ-second-order stationary point* as a point $\mathbf{x} \in \mathcal{X}$ for which $\lambda_{\min}(\nabla^2 f(\mathbf{x})) \geq -\sqrt{}$

Algorithm 1 Perturbed Gradient Descent (PGD)

```
for $k = 0, 1, \dots,$ do
    if $\|\nabla f(\mathbf{x}_k)\| < g$ and no perturbation in last $\mathcal{T}$ steps then
        $\mathbf{x}_k \leftarrow \mathbf{x}_k + \xi_k \qquad \xi_k \sim \mathrm{Unif}(\mathbb{B}_0(r))$
    $\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \nabla f(\mathbf{x}_k)$
```

where $\mathbf{x}_0$ is the point at which the algorithm is initialized. While the rate here is less favorable than in the case of smooth convex functions—where it is $O(\epsilon)^1$—the rate retains the essential feature from the convex setting that it is independent of dimension. Recall, however, that saddle points are ϵ-first-order stationary points, and thus this result describes (inter alia) the rate of *approach* to a saddle point. The question that we now turn to is the characterization of the rate of *escape* from a saddle point, where we expect that dimensionality will rear its head.

Turning to algorithmic considerations, we first note that—in contradistinction to the convex case—pure gradient descent will *not* suffice for convergence to a local minimum. Indeed, in the presence of saddle points, the rate of convergence of gradient descent can depend exponentially on dimension (Du et al., 2018). Thus we need to move beyond gradient descent to have a hope of *efficient* escape from saddle points. We could avail ourselves of Hessians, in which case it would be relatively easy to identify directions of escape (as eigenvectors of the Hessian), but as discussed earlier we wish to avoid the use of Hessians on computational grounds. Instead, we focus on gradient descent that is augmented with a stochastic perturbation. Ge et al. (2015) and Jin et al. (2017a) studied such an augmentation in which a homogeneous stochastic perturbation (uniform noise in a ball) is added sporadically to the current iterate. Specifically, noise is added when: (1) the norm of the gradient at the current iterate is small, and (2) the most recent such perturbation is at least $\mathcal{T}$ steps in the past, where $\mathcal{T}$ is an algorithmic hyperparameter. We refer to this algorithm as "perturbed gradient descent" (PGD); see Algorithm 1.

We can now state h oJ

be an ϵ-second order stationary point after the following number of iterations k:

$$O\left(\frac{\ell(f(\mathbf{x}_0)-f^\star)}{\epsilon^2}\log^4\left(\frac{d}{\epsilon^2}\right)\right).$$

We see that the first factor is exactly the same as for ϵ-first-order stationary points with pure gradient descent. The penalty incurred by incorporating the perturbation—and thereby avoiding saddle points—is quite modest—it is only polylogarithmic in the dimension d.

With this convergence result as background, we turn to the main question of this section: Does acceleration aid in the escape from saddle points? In particular, can we improve on the rate in Theorem 2 by incorporating acceleration into the perturbed gradient descent algorithm? We will see that the answer is "yes"; moreover, we will see that a continuous-time dynamical-systems perspective will play a key role in establishing the result.

A major challenge in analyzing accelerated algorithms is that the objective function does not decrease monotonically as is the case for gradient descent. In the convex case, we met this challenge by explciting the Lagrangian/Hamiltonian formulation to design a Lyapunov function, see Eq. (11). These Lyapunov functions, however, involve the global minimum $\mathbf{x}^\star$, which is unknown to the algorithm. This is not problematic in the convex setting, as terms involving $\mathbf{x}^\star$ can be bounded using convexity; in the nonconvex setting, however, it is fatal.

To overcome this problem in the nonconvex setting, Jin et al. (2017b) developed a Hamiltonian that is appropriate for the analysis of nonconvex problems. Specializing to Euclidean geometry, the function takes the following form:

$$E_t := \frac{1}{2\eta}\|\mathbf{v}_t\|^2 + f(\mathbf{x}_t); \tag{19}$$

a sum of kinetic energy and potential energy terms. Let us consider using this Hamiltonian to analyze the following second-order differential equation:

$$\ddot{\mathbf{x}} + \tilde{\theta}\dot{\mathbf{x}} + \nabla f(\mathbf{x}) = 0. \tag{20}$$

Integrating both sides, we obtain:

$$f(\mathbf{x}(t_2)) + \frac{1}{2}\dot{\mathbf{x}}(t_2)^2 = f(\mathbf{x}(t_1)) + \frac{1}{2}\dot{\mathbf{x}}(t_1)^2 - \tilde{\theta}\int_{t_1}^{t_2}\dot{\mathbf{x}}(t)^2\mathrm{d}t. \tag{21}$$

The integral shows that the Hamiltonian decreases monotonically with time t, and the decrease is given by the *dissipation* term $\tilde{\theta}\int_{t_1}^{t_2}\dot{\mathbf{x}}(t)^2\mathrm{d}t$. Note that Eq. (21) holds regardless of the convexity of $f(\cdot)$.

Although the Hamiltonian decreases monotonically in continuous time, Jin et al. (2017b) show that this is *not* the case in discrete time (in the nonconvex setting). Thus, once again, the complexity associated with acceleration manifests itself principally in the transition to discrete time. Jin et al. (2017b) were able to resolve this problem by isolating a condition under which the change of the Hamiltonian is indeterminate (it may decrease or increase), and show that

Algorithm 2 Perturbed Accelerated Gradient Descent (PAGD)

```
1: $\mathbf{v}_0 = 0$
2: for $k = 0, 1, \ldots,$ do
3:     if $\|\nabla f(\mathbf{x}_k)\| \le \epsilon$ and no perturbation in last $\mathcal{T}$ steps then
4:         $\mathbf{x}_k \leftarrow \mathbf{x}_k + \xi_k \qquad \xi_k \sim \mathrm{Unif}(\mathbb{B}_0(r))$    } Perturbation
5:     $\mathbf{y}_k = \mathbf{x}_k + \lambda_k \mathbf{v}_k$                              }
6:     $\mathbf{x}_{k+1} = \mathbf{y}_k - \eta \nabla f(\mathbf{y}_k)$                     } AGD
7:     $\mathbf{v}_{k+1} = \mathbf{x}_{k+1} - \mathbf{x}_k$                                }
8:     if $f(\mathbf{x}_k) \le f(\mathbf{y}_k) + \langle \nabla f(\mathbf{y}_k), \mathbf{x}_k - \mathbf{y}_k \rangle - \frac{\gamma}{2}\|\mathbf{x}_k - \mathbf{y}_k\|^2$ then
9:         $(\mathbf{x}_{k+1}, \mathbf{v}_{k+1}) \leftarrow$ Negative-Curvature-Exploitation$(\mathbf{x}_k, \mathbf{v}_k)$    } Negative curvature exploitation
```

under the complement of this condition, the Hamiltonian necessarily decreases. Roughly speaking, the condition arises when the function is "too nonconvex." Moreover, this condition can be assayed algorithmically, and the algorithm can be modified to ensure decrease of the Hamiltonian when the condition arises.

The overall algorithm, Perturbed Accelerated Gradient Descent (PAGD), is presented in Algorithm 2. Steps 3 and 4 are identical to the PGD algorithm. Steps 5, 6 and 7 replace gradient descent in the latter algorithm with accelerated gradient descent (cf. Eq. (3)). Step 8 measures the "amount of nonconvexity," and if it is too large, makes a call to a function called "Negative-Curvature-Exploitation." This function does one of two things: (1) if the momentum is large, it zeros out the momentum; (2) if the momentum is small, it conducts a local line search along the direction of the momentum. Jin et al. (2017b) prove that this overall algorithm yields monotone decrease in the Hamiltonian. Moreover, they use this result to prove the following theorem regarding the convergence rate of PAGD. (As in the case of PGD, we are not specifying the settings of the various algorithm hyperparameters; we refer to Jin et al. (2017b) for these settings. We note, however, that all hyperparameters are functions of the Lipschitz constants ℓ and ρ, with the exception of a constant quantifying the probability statement in the theorem, and a universal scaling constant.)

Theorem 3. *Assume that the function f is ℓ-smooth and ρ-Hessian Lipschitz. Then, with high probability, at least one of iterates, $\mathbf{x}_k$, of PAGD (Algorithm 2) will be an ϵ-second order stationary point after the following number of iterations:*

$$O\left(\frac{\ell^{1/2}\rho^{1/4}(f(\mathbf{x}_0) - f^*)}{\epsilon^{7/4}} \log^6\left(\frac{d}{\epsilon}\right)\right).$$

Comparing this result to Theorem 2, we see that the rate has improved from $1/\epsilon^2$ to $1/\epsilon^{7/4}$. Thus we see that acceleration provably improves upon gradient descent in the nonconvex setting—acceleration hastens the escape from saddle points. We also see that, once again, there is a mild—polylogarithmic—dependence on the dimension d.

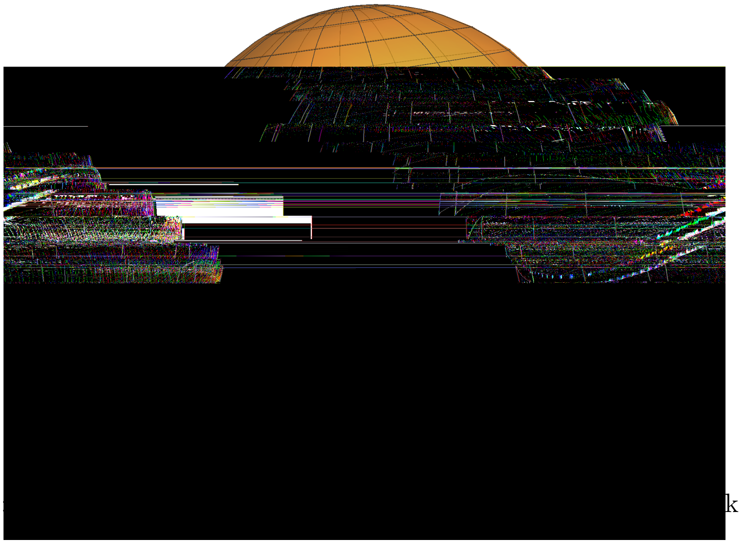


Figure 3 [illegible] k region.

The core of the proofs of Theorems 2 and 3 revolves around the study of the local geometry around saddlepoints and its interaction with gradient-descent dynamics. As depicted in Figures 3 and 4, there is a slab-like region in the neighborhood of a saddle point in which gradient descent will be "stuck"—taking an exponential amount of time to escape. This region is not flat, but instead varies due to the variation of the Hessian in this neighborhood. The Lipschitz assumption gives us control over this variation. To analyze the width of the stuck region, and thus its volume as a fraction of the perturbation ball, Jin et al. (2017b) study the rate of escape of a pair of gradient-descent (or accelerated-gradient-descent) sequences that start on the sides of the stuck region. These initial points are a distance r apart along the direction given by the minimum eigenvector of the Hessian, $\nabla^2 f(\mathbf{x})$, at the saddle point. The critical value r for which at least one of the two sequences escapes the stuck region quickly can be computed, and this provides an estimate of the volume of the stuck region. The overall result is that this volume is small compared to that of the perturbation ball, and thus the perturbation is highly likely to cause the optimization algorithm to leave the stuck region (and not return).

4 Underdamped Langevin Diffusion

Our focus thus far has been on dynamical systems which are deterministic, with stochastic

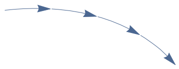


Figure 4: Gradient flow and "narrow band" stuck region in two dimensions.

uniform perturbation in a ball—is sufficient for fast escape, but it is not necessary. Given the success of this simple choice, however, we are motivated to

"Langevin MCMC."

Asymptotic guarantees for overdamped Langevin MCMC were established in Gelfand and Mitter (1991). The first explicit proof of non-asymptotic convergence of overdamped Langevin MCMC for log-smooth and strongly log-concave distributions was given by Dalalyan (2017), who showed that discrete, overdamped Langevin diffusion achieves ϵ error, in total variation distance, in $O(d/\epsilon^2)$ steps. Following this, Durmus and Moulines (2016) proved that the same algorithm achieves ϵ error, in 2-Wasserstein distance, in $O(d/\epsilon^2)$ steps.[2]

Our second-order perspective motivates us to consider *underdamped Langevin diffusion*:

$$\begin{aligned} d\mathbf{v}_t &= -\gamma \mathbf{v}_t dt - u\nabla f(\mathbf{x}_t)dt + \sqrt{2\gamma u}dB_t \\ d\mathbf{x}_t &= \mathbf{v}_t dt, \end{aligned} \tag{23}$$

where u and γ are parameters. It can be shown that the invariant distribution of this continuous-time process is proportional to $\exp(-(f(\mathbf{x}) + \|\mathbf{v}\|_2^2/2u))$. Thus the marginal distribution of $\mathbf{x}$ is proportional to $\exp(-f(\mathbf{x}))$ and samples from $p^\star(\mathbf{x}) \propto \exp(-f(\mathbf{x}))$ can be obtained by solving the underdamped Langevin diffusion numerically and ignoring the momentum component.

Underdamped Langevin diffusion is interesting because it is analogous to accelerated gradient descent; both are second-order dynamical systems. Moreover, it contains a Hamiltonian component, and its discretization can be viewed as a form of Hamiltonian MCMC. Hamiltonian MCMC has been empirically observed to converge faster to the invariant distribution compared to standard Langevin MCMC (Betancourt et al., 2017).

In recent work, Cheng et al. (2017) have analyzed underdamped Langevin diffusion. They have shown that—in the same setting analyzed by Durmus and Moulines (2016) for overdamped Langevin diffusion—that the underdamped algorithm achieves a convergence rate of $O(\sqrt{d}/\epsilon)$ in 2-Wasserstein distance. This is a significant improvement over the $O(d/\epsilon^2)$ rate of overdamped Langevin diffusion, both in terms of the accuracy parameter ϵ and the dimension d.

5 Discussion

The general topic of gradient-based optimization, and its application to large-scale statistical inference problems, is currently very active. Let us highlight one particular set of questions that appear likely to attract ongoing attention in coming years. Note that optimization methods are used classically in the statistical setting to solve point estimation problems, where the core problem

[2] Recall that the Wasserstein distance, $W_2(\mu, \eta)$, between probability measures μ and η is defined as follows:

$$W_2(\mu, \eta) := \left(\inf_{\zeta \in \Gamma(\mu,\eta)} \int \|\mathbf{x} - \mathbf{y}\|_2^2 d\zeta(\mathbf{x}, \mathbf{y}) \right)^{1/2}, \tag{22}$$

where ζ ranges over the set of transference plans $\Gamma(\mu, \eta)$.

is to output a single point in the configuration space that has desirable statistical properties. But the broader problem is to provide in addition an indication of the uncertainty associated with that output, in the form of some summary of a probability distribution. Optimization ideas can be relevant here as well, by considering a configuration space that is a space of probability distributions. Relatedly, one can ask to converge not to a single point, but to a distribution over points. The Hamiltonian approach naturally yields oscillatory solutions, and, as we have seen, some work is required to obtain algorithms that converge to a point. This suggests that the Hamiltonian approach may in fact be easier to employ in the setting of distributional convergence than in the point estimation setting, and thereby provide an algorithmic bridge between point estimation and broader inference problems. Indeed, in Bayesian inference, Hamiltonian formulations (and symplectic integration, in the form of leapfrog integrators) have been successfully employed in the setting of Markov chain Monte Carlo algorithms, where the momentum component of the Hamiltonian (empirically) provides faster mixing. Deeper connections between acceleration and computationally-efficient inference are clearly worth pursuing.

References

Allen-Zhu, Z. and Orecchia, L. (2014). Linear coupling: An ultimate unification of gradient and mirror descent. *arXiv preprint arXiv:1407.1537*.

Bandeira, A. S., Boumal, N., and Voroninski, V. (2016). On the low-rank approach for semidefinite programs arising in synchronization and community detection. In *Conference on Computational Learning Theory (COLT)*, pages 361–382.

Betancourt, M., Byrne, S., Livingstone, S., and Girolami, M. (2017). The geometric foundations of Hamiltonian Monte Carlo. *Bernoulli*, 23:2257–2298.

Betancourt, M., Jordan, M. I., and Wilson, A. (2018). On symplectic optimization. *arXiv preprint arXiv:1802.03653*.

Bhojanapalli, S., Neyshabur, B., and Srebro, N. (2016). Global optimality of local search for low rank matrix recovery. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3873–3881.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

Boumal, N., Voroninski, V., and Bandeira, A. (2016). The non-convex Burer-Monteiro approach works on smooth semidefinite programs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2757–2765.

Bubeck, S., Lee, Y. T., and Singh, M. (2015). A geometric alternative to Nesterov's accelerated gradient descent. *arXiv preprint arXiv:1506.08187*.

Cheng, X., Chatterji, N., Bartlett, P., and Jordan, M. I. (2017). Underdamped Langevin MCMC: A non-asymptotic analysis. *arXiv preprint arXiv:1707.03663*.

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., and LeCun, Y. (2014). The loss surface of multilayer networks. *arXiv:1412.0233*.

Dalalyan, A. S. (2017). Theoretical guarantees for approximate sampling from smooth and log-concave densities. *Journal of the Royal Statistical Society: Series B*, 79:651–67.

Du, S., Jin, C., Lee, J., Jordan, M. I., Poczos, B., and Singh, A. (2018). Gradient descent can take exponential time to escape saddle points. In *Advances in Neural Information Processing Systems (NIPS) 30*.

Durmus, A. and Moulines, E. (2016). Sampling from strongly log-concave distributions with the Unadjusted Langevin Algorithm. *arXiv preprint arXiv:1605.01559*.

Ge, R., Huang, F., Jin, C., and Yuan, Y. (2015). Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on Computational Learning Theory (COLT)*.

Ge, R., Jin, C., and Zheng, Y. (2017). No spurious local minima in nonconvex low rank problems: A unified geometric analysis. *arXiv preprint arXiv:1704.00708*.

Ge, R., Lee, J. D., and Ma, T. (2016). Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2973–2981.

Gelfand, S. B. and Mitter, S. (1991). Recursive stochastic algorithms for global optimization in $\mathbb{R}^d$. *SIAM Journal on Control and Optimization*, 29:999–1018.

Hairer, E., Lubich, C., and G, W. (2006). *Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations.* Springer, New York.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. (2017a). How to escape saddle points efficiently. In *International Conference on Machine Learning (ICML)*.

Jin, C., Netrapalli, P., and Jordan, M. I. (2017b). Accelerated gradient descent escapes saddle points faster than gradient descent. *arXiv preprint arXiv:1711.10456*.

Kawaguchi, K. (2016). Deep learning without poor local minima. In *Advances In Neural Information Processing Systems (NIPS)*, pages 586–594.

Krichene, W., Bayen, A., and Bartlett, P. (2015). Accelerated mirror descent in continuous and discrete time. In *Advances in Neural Information Processing Systems (NIPS) 27*.

Lee, J. D., Simchowitz, M., Jordan, M. I., and Recht, B. (2016). Gradient descent only converges to minimizers. In *Conference on Computational Learning Theory (COLT)*, pages 1246–1257.

Mei, S., Misiakiewicz, T., Montanari, A., and Oliveira, R. I. (2017). Solving SDPs for synchronization and maxcut problems via the Grothendieck inequality. In *Conference on Computational Learning Theory (COLT)*, pages 1476–1515.

Nemirovskii, A. and Yudin, D. (1983). *Problem Complexity and Method Efficiency in Optimization*. John Wiley, New York.

Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate o(1/k2). *Soviet Mathematics Doklady*, 27:372–376.

Nesterov, Y. (1998). *Introductory Lectures on Convex Programming*. Springer, New York.

Nesterov, Y. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal of Optimization*, 22:341–362.

Nesterov, Y. and Polyak, B. T. (2006). Cubic regularization of Newton method and its global performance. *Mathematical Programming*, 108(1):177–205.

Su, W., Boyd, S., and Candes, E. J. (2016). A differential equation for modeling Nesterov's accelerated gradient method: theory and insights. *Journal of Machine Learning Research*, 17(153):1–43.

Sun, J., Qu, Q., and Wright, J. (2016a). Complete dictionary recovery over the sphere I: Overview and the geometric picture. *IEEE Transactions on Information Theory*, 63:853–884.

Sun, J., Qu, Q., and Wright, J. (2016b). A geometric analysis of phase retrieval. In *IEEE International Symposium on Information Theory (ISIT)*, pages 2379–2383. IEEE.

Wibisono, A., Wilson, A. C., and Jordan, M. I. (2016). A variational perspective on accelerated methods in optimization. *Proceedings of the National Academy of Sciences*, 133:E7351–E7358.

Wilson, A. C., Recht, B., and Jordan, M. I. (2016). A Lyapunov analysis of momentum methods in optimization. *arXiv preprint arXiv:1611.02635*.