Randomization in Statistical Machine Learning

by

Zhishen Huang

B.S., Southeast University, 2015

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Applied Mathematics

2020

Committee Members:

Prof. Stephen Becker

Prof. Claire Monteleoni

Prof. Sergei Kuznetsov

Prof. François Meyer

Prof. William Kleiber

Huang, Zhishen (Ph.D., Applied Mathematics)

Randomization in Statistical Machine Learning

Thesis directed by Prof. Stephen Becker

Supervised learning and reinforcement learning problems are often formulated as optimization problems for training. The optimization algorithms themselves bear interest from the mathematical point of view. This thesis discusses the usage of randomization in optimization, which makes possible what corresponding deterministic algorithms are unable to achieve.

Applying randomization in algorithms has the capability of reducing the time or space complexity at the expense of potential failure to provide a valid solution. Early prominent examples of randomized algorithms include quicksort, Karger's algorithm for min-cut problem and the Bloom filter. In this thesis, randomization is introduced into optimization algorithms and is used for data compression.

This thesis considers first-order optimization algorithms due to their practicality for large-scale application. The first chapter considers the minimization of nonconvex and nonsmooth objectives, where we give probabilistic guarantees for the proximal gradient descent method to converge to local minima [HB20a]. The second chapter varies the randomization format for gradient descent where Gaussian noise is injected at each iteration step. We point out the ergodicity property of such variation, which is not available for a deterministic version of gradient descent.

The third chapter considers using the sketching technique to compress data and evaluate statistics solely based on the sketched dataset [HB20b]. We give theoretical guarantee for the evaluation accuracy of autocorrelation from data sketches and demonstrate numerical performance on molecular dynamic simulation data and synthetic data.

The fourth chapter considers a deterministic algorithm for integer-constrained programming, where we suggest a finer convex relaxation where the primal problem is reformulated by Fenchel-Rockafellar duality, and separated into two subproblems based on pre-selected support.

Dedication

To my family

Acknowledgements

I am grateful to my advisor Prof. Stephen Becker for his guidance, patience and inspiration throughout my PhD career in Boulder.

To Dr. Mohan Sarovar, Prof. Kristina Lerman and Dr. Andrés Abeliuk, Dr. Eric Kightley and Mr. Robert Hipps, I give my thank for their tremendous support in my summer internships.

I thank the friendship of Harry Dudley, Mitchell Krock and Eric Kightley.

My unreserved gratitude goes to my family for their unwavering love, care, patience and encouragement they have given me throughout my life.

The discussion with Prof. Manuel Lladser has benefited penning the reachability section in the second chapter of this dissertation.

This dissertation is partially supported by the grant NSF/DMS 1819251.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Perturbed Proximal Descent to Escape Saddle Points for Non-convex and Non-smooth Objective Functions

1.1 Introduction

We consider the problem of finding approximate local minimizers of the problem

$$\text{minimize}_{\mathbf{x}\in\mathbb{R}^d} \left(\Phi(\mathbf{x}) := f(\mathbf{x}) + g(\mathbf{x})\right) \tag{1.1}$$

where $f(\mathbf{x})$ is not convex but smooth (and with full domain), and $g(\mathbf{x})$ is convex but not smooth. Many optimization problems in engineering, signal processing and machine learning can be cast in this framework, where f is a smooth loss function, and g is a non-smooth regularizer such as a norm. For example, our model captures regularized neural networks [GJP95], where the regularization can induce sparsity as an alternative to dropout. In this chapter, for simplicity we restrict our discussion to $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$, where $\lambda \geq 0$ is a constant, but many of the results apply to more general choices of g. The first-order condition is $0 \in \nabla f(\mathbf{x}) + \partial g(\mathbf{x})$, and any $\mathbf{x}$ satisfying this condition is called a "stationary point" (see [BC17] for background on the subdifferential ∂g). All local minimizers are stationary points, but not vice-versa. We define a "saddle point" to be any stationary point where the Hessian is indefinite (and therefore not a local minimizer). This chapter extends a recent line of work [JGN$^+$17] to analyze when we can expect to find a local minimizer. It has been argued that in many machine learning problems, finding any local minimizer is often enough for good performance, but finding a saddle point is not useful [DPG$^+$14].

The fact that g is non-smooth is crucially important, and it does more than just complicate the analysis, as it also requires a new algorithm. In the smooth case, f is often minimized using gradient descent or an accelerated variant [Nes83] with a fixed stepsize. Naïvely extending gradient descent to apply to (1.1) leads to subgradient descent with fixed-stepsize. Unfortunately, this method fails to converge as the example $d = 1, \lambda = 1$ and $f = 0$ shows [Sho62] since for a generic choice of the initial point, the sequence is not Cauchy.

Instead of gradient descent, we use a perturbed version of proximal gradient descent. For a real-valued convex lower semi-continuous function g, define the "proximity" operator (or "prox" for short) as the map $\operatorname{prox}_g(\mathbf{y}) = \operatorname{argmin}_{\mathbf{x}} g(\mathbf{y}) + \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2$ (throughout this chapter, for vectors we use $\|\cdot\|$ to denote the Euclidean norm). Equivalently, $\operatorname{prox}_g = (I + \partial g)^{-1}$, and thus the first-order condition is equivalent to $\mathbf{x} = \operatorname{prox}_{\eta g}[\mathbf{x} - \eta\nabla f(\mathbf{x})]$ for any $\eta > 0$. Proximal gradient descent is the iteration $\mathbf{x}_{t+1} = \operatorname{prox}_{\eta g}[\mathbf{x}_t - \eta\nabla f(\mathbf{x}_t)]$, so it immediately follows that if the sequence converges, it converges to a stationary point. Convergence of the sequence is known to follow from mild assumptions on f and g, the stepsize η, and boundedness of the sequence $\{\mathbf{x}_t\}$ [ABS11].

We define a *second-order stationary point* to be a first-order stationary point $\mathbf{x}$ that additionally satisfies $\nabla^2 f(\mathbf{x}) \succ 0$, which is a sufficient condition for $\mathbf{x}$ to be a local minimizer. Our main contribution is showing that under suitable assumptions, a perturbed version of proximal gradient descent will generate a sequence that converges to an approximate second-order stationary point. We make assumptions on the second-order behavior of f, similar to assumptions under which it is known that gradient descent will always converge to a second-order stationary point except for adversarially chosen starting points [LSJR16] — in contrast to Newton's method, which is attracted to all stationary points. However, even in the smooth case when the sequence converges, gradient descent converges arbitrarily slowly [DJL$^+$17] in the presence of a saddle point, so perturbation is necessary. In the non-smooth case, perturbation is even more important due to the proximal nature of the algorithm.

A toy example: Gaussian Bump Consider the function $\Phi : \mathbb{R}^2 \to \mathbb{R}, x \mapsto \frac{1}{2}(x^2 - y^2)\mathrm{e}^{-\frac{x^2+y^2}{5}} + \frac{1}{100}h_{100}(\mathbf{x})$ where $h_{100}(\mathbf{x})$ is the Huber function with parameter 100 [Bec17]. The choice of this combination of Huber parameter and the magnitude of Huber function ensures that

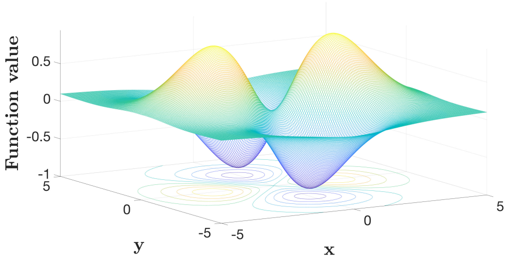


Figure 1.1: Graph of function $\Phi(\mathbf{x})$

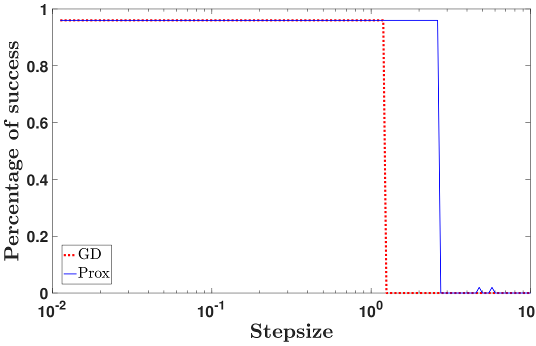


Figure 1.2: The comparison between gradient descent (GD) and proximal gradient descent (Prox) on the percentage of success finding the correct local minima, as a function of the stepsize η

the origin is a saddle point. The Huber function approximates the ℓ_1 norm. The plot is show in Fig. 1.1.

This function has two local minima and a saddle point at $(0,0)$. Because the Huber function is both smooth and it has a known proximity operator, we can treat it as either part of the smooth f component or the non-smooth g component, and therefore run either gradient descent or proximal gradient descent. We experiment with both algorithms, randomly picking initial points at $\mathbf{x}_0 = (0.3, 0.01) + \boldsymbol{\xi}$ where $\boldsymbol{\xi}$ is sampled uniformly from $\mathbb{B}_0(\frac{1}{10}\|\mathbf{x}_0\|)$, and varying the stepsize η, with fixed maximum iteration 1000. Figure 1.2 shows the empirical success rate of finding a local minimizer (as opposed to converging to the saddle point at $(0,0)$).

We observe that the range of stable step size for the proximal descent algorithm is wider than gradient descent, and the success rate of proximal descent is as high as the gradient descent. This example motivates us to adopt proximal descent over gradient descent in real application for better stability and equivalent, if not better, accuracy.

A coincidence In this toy example, the saddle point at $(0,0)$ happens to be a fixed point of proximal operator of $\eta\lambda\|\mathbf{x}\|_1$. Soft thresholding, as the proximal operator of $\lambda\|\mathbf{x}\|_1$ is known [CW05], has an attracting region that sets nearby points to 0. The radius of the attracting region (per dimension) is $\eta\lambda$, thus if $\|\mathbf{x}_{t_0} - \eta\nabla f(\mathbf{x}_{t_0})\|_\infty \leq \eta\lambda$ for some iteration t_0, then $\mathbf{x}_t = 0$

for all $t > t_0$. Proximal gradient descent performs even better when the saddle point is not in the attracting region.

Structure of the chapter 1 Section 1.2 states the algorithm and section 1.3 states the main theorems, followed by section 1.4 where the theoretical guarantee is presented with proof. Section 1.5 shows numerical experiments.

1.1.1 Related literature

Second-order methods for smooth objectives Some recent second order methods, mainly based on either cubic-regularized Newton methods as in or based on trust-region methods, have been shown to converge to ε-approximate local minimizers of smooth non-convex objective functions. Both of these approaches involve evaluation of full Hessian. The central idea for the cubic-regularized Newton method is to locally approximate the objective with cubic polynomial functions and optimize w.r.t. it, which can either be formulated as a one-dimensional optimization problem which involves full Hessian inversion [NP06] with time complexity $\mathcal{O}(\log \varepsilon)$, or can be solved with gradient descent in the stochastic setting with time complexity $\mathcal{O}(\varepsilon^{-3.5})$ [TSJ$^+$18]. The trust-region method uses locally constrained quadratic optimization problem as subroutine for each iteration [CRS17, SQW15], and converges to local optimizers in $\mathcal{O}(\varepsilon^{-1.5})$ iterations as reported in Curtis.

Hybrid approaches which involve evaluation of full Hessian but do not compute the inverse have also been considered. These approaches are referred as *'Hessian-free'* or *Hessian-vector product* approaches. In particular, [CDHS18] points out that the oracle which returns the product of Hessian and vector can suffice to extract the negative curvature of the geometry thus finding the proper descent direction, an example of which is Lanczos method [KW92]. The time complexity to return a local optimizer reported in Carmon is $\mathcal{O}(\varepsilon^{-1.75})$. [RZS$^+$17] describes a generic scheme which alternates between first order oracles and second order oracles to guarantee convergence to local optimizers.

There have been also attempts to apply second-order methods on training neural networks which often involves high-dimensional parameter space and nonconvex objectives. [MG15] suggests that in the supervised learning scenario to train a feed-forward neural network, one can use the Fisher information matrix as Hessian for the loss function, exploit Kronecker structure to construct the approximate inverse of Hessian and use line search to obtain descent direction (aka damped Newton method or natural gradient).

See [AZL18] for a more thorough review of these methods. We do not consider these methods further due to the high-cost of solving for the Newton step in large dimensions and the sophistication of implementation.

First-order methods for smooth objectives We focus on first order methods because each step is cheaper and simpler to implement and these methods are more frequently adopted by the deep learning community. Xu et al. in [XJY18] and Allen-Zhu et al. in [AZL18] develop Negative-Curvature (NC) search algorithms, which find descent direction corresponding to negative eigenvalues of Hessian matrix. The core idea of NC is that a gradient descent subroutine is repeatedly executed at a saddle point where the second order derivative dominates the update, so that the repetition of gd essentially serves as the power method to find the eigenvector corresponding to the smallest eigenvalue of Hessian. The NC search routines avoid using either Hessian or Hessian-vector information directly, and it can be applied in both online and deterministic scenarios.

In the online setting, combining NC search routine with first-order stochastic methods will give algorithms NEON-$\mathcal{A}$ [XJY18] and NEON2+SGD [AZL18] with iteration cost $\mathcal{O}(\frac{d}{\varepsilon^{3.5}})$ and $\mathcal{O}(\varepsilon^{-3.5})$ respectively (the latter still depends on dimension, whose induced complexity is at least $\ln^2(d)$), and these methods generate a sequence that converges to an approximate local minimum with high probability. In the offline setting, Jin et al. in [JGN$^+$17] provide a stochastic first order method that finds an approximate local minimizer with high probability at computational cost $\mathcal{O}(\frac{\ln^4(d)}{\varepsilon^2})$. Combining NEON2 with gradient descent or SVRG, the cost to find an approximate local minimum is $\mathcal{O}(\varepsilon^{-2})$, whose dependence on dimension is not specified but at least $\ln^2(d)$. These methods make Lipschitz continuity assumptions about the gradient and Hessian, so they do not apply to non-smooth optimization.

A recent preprint [LY19] approaches the problem of finding local minima using the forward-backward envelope technique developed in [STP17], where the assumption about the smoothness of objective function is weakened to local smoothness instead of global smoothness.

Non-smooth objectives In the offline settings, Boţ et al. propose a proximal algorithm for minimizing non-convex and non-smooth objective functions in [BCN18]. They show the convergence to KKT points instead of approximate second-order stationary points. Moreau envelope is also an instrument to locally convert the nonsmooth problem to a smooth problem in variational analysis, thus rendering gradient descent applicable for the nonsmooth scenario [STP17, LTP19]. In these work the second-order derivative of envelope function needs to be defined w.r.t. particular direction and are always positive definite under their regularization assumptions, thus unable to characterize the second-order convergence property of the iteration process. Other work [ABS11, BST14] relies on the Kurdya-Lojasiewicz inequality and shows convergence to stationary points in the sense of the limiting sub-differential, which is not the same as a local minimizer or approximate second-order stationary point. In the online setting, Reddi et al. demonstrated in [JRSPS16] that the proximal descent with variance reduction technique (proxSVRG) has linear convergence to a first-order stationary point, but not to a local minimizer.

1.2 Algorithm

The algorithm takes as input a starting vector $\mathbf{x}_0$, the gradient Lipschitz constant L, the Hessian Lipschitz constant ρ, the second-order stationary point tolerance ε, a positive constant c, a failure probability δ, and estimated function value gap Δ_Φ. The key parameter for Algorithm 1 is the constant c. It should be made large enough so that the effect of perturbation will be significant enough for escaping saddle points, and at the same time not too large so that the iteration stepsize is of reasonable magnitude and the iteration will not go wild. The output of the algorithm is an ε-second-order stationary point (see Def. 3).

Algorithm 1 Perturbed Proximal Descent: input$(\mathbf{x}_0, L, \rho, \varepsilon, c, \delta, \Delta_\Phi)$

```
Require: $\chi \leftarrow 3\max\{\ln(\frac{dL\Delta_\Phi}{c\varepsilon^2\delta}), 4\}$, $\eta \leftarrow \frac{c}{L}$, $r \leftarrow \frac{\sqrt{c}}{\chi^2}\cdot\frac{\varepsilon}{L}$, $g_{\text{thres}} \leftarrow \frac{\sqrt{c}}{\chi^2}\cdot\varepsilon$, $\Phi_{\text{thres}} \leftarrow \frac{c}{\chi^3}\cdot\sqrt{\frac{\varepsilon^3}{\rho}}$, $t_{\text{thres}} \leftarrow \frac{\chi}{c^2}\cdot\frac{L}{\sqrt{\rho\varepsilon}}$
t_noise ← −t_thres − 1
for $t = 0, 1, \ldots$ do
    if $\|\mathbf{x} - \text{prox}_{\eta g}[\mathbf{x} - \eta\nabla f(\mathbf{x})]\| < g_{\text{thres}}$ and $t - t_{\text{noise}} > t_{\text{thres}}$ then    ▷ saddle point condition check
        $\tilde{\mathbf{x}}_t \leftarrow \mathbf{x}_t$,   $t_{\text{noise}} \leftarrow t$
        $\mathbf{x}_t \leftarrow \tilde{\mathbf{x}}_t + \boldsymbol{\xi}_t$,   $\boldsymbol{\xi}_t$ uniformly $\sim \mathbb{B}_0(r)$    ▷ add perturbation
    end if
    if $t - t_{\text{noise}} = t_{\text{thres}}$ and $\Phi(\mathbf{x}_t) - \Phi(\tilde{\mathbf{x}}_{t_{\text{noise}}}) > -\Phi_{\text{thres}}$ then    ▷ sufficient function value decrease check
        return $\tilde{\mathbf{x}}_{t_{\text{noise}}}$
    end if
    $\mathbf{x}_{t+1} \leftarrow \text{prox}_{\eta g}[\mathbf{x}_t - \eta\nabla f(\mathbf{x}_t)]$    ▷ PPD step
end for
```

1.3 Main result: escaping saddle points through perturbed proximal descent

The main step in the algorithm is a proximal gradient descent step applied to $f + g$, defined as

$$\begin{aligned}\mathbf{x}_{t+1} &= \underset{\mathbf{y}}{\text{argmin}}\, f(\mathbf{x}_t) + \langle\nabla f(\mathbf{x}_t), \mathbf{y} - \mathbf{x}_t\rangle + \frac{\eta^{-1}}{2}\|\mathbf{y} - \mathbf{x}_t\|^2 + g(\mathbf{y}) \\ &= \text{prox}_{\eta g} \circ (I - \eta\nabla f)(\mathbf{x}_t) \qquad (1.2)\end{aligned}$$

One motivation of preferring proximal descent to gradient descent, as shown in Figure 1.2, is the stability of the algorithm with respect to stepsize change. The proximal step is similar to the implicit/backward Euler scheme, as equation (1.2) can be written as $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\big(\nabla f(\mathbf{x}_t) + \partial g(\mathbf{x}_{t+1})\big)$. From this perspective, we expect that proximal descent will demonstrate at least the same convergence speed as gradient descent and stronger stability with respect to hyperparameter setting.

Definition 1 (Gradient Mapping)**.** *Consider a function* $\Phi(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$*. The gradient mapping is defined as* $G_\eta^{f,g}(\mathbf{x}) := \mathbf{x} - \text{prox}_{\eta g}[\mathbf{x} - \eta\nabla f(\mathbf{x})]$

In the rest of this chapter, the super- and subscript of the gradient mapping are not specified, as it is always clear that f represents the smooth nonconvex part of Φ, g represents $\lambda\|\mathbf{x}\|_1$, and η

is the stepsize used in the algorithm. Observe that the gradient map is just the gradient of f if $g \equiv 0$.

Definition 2 (First order stationary points)**.** *For a function $\Phi(\mathbf{x})$, define first order stationary points as the points which satisfy $G(\mathbf{x}) = 0$.*

Definition 3 (ε-second-order stationary point)**.** *Consider a function $\Phi(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$. A point $\mathbf{x}$ is an ε-second-order stationary point if*

$$\|G(\mathbf{x})\| \leq \varepsilon \quad \textit{and} \quad \lambda\big(\nabla^2 f(\mathbf{x})\big)_{\min} \geq -\sqrt{\rho\varepsilon} \tag{1.3}$$

where $\lambda(\cdot)_{\min}$ is the smallest eigenvalue.

The first Lipschitz assumption below is standard [Bec17], and the assumption on the Hessian was used in [JGN$^+$17] (for example, it is true if f is quadratic).

Assumption 1 (Lipschitz Properties)**.** *∇f is L-Lipschitz continuous and $\nabla^2 f$ is ρ Lipschitz continuous. We write $\mathcal{H}$ as shorthand for $\nabla^2 f(\mathbf{x})$ when $\mathbf{x}$ is clear from context.*

Assumption 2 (Moderate Nonsmooth Term)**.** *The magnitude of $\|\mathbf{x}\|_1$ term, which is denoted by λ, satisfies inequalities* (1.7) *and* (1.9)*.*

Theorem 4 (Main)**.** *There exists an absolute constant $c_{\max}$ such that if $f(\cdot)$ satisfies 1 and 2, then for any $\delta > 0, \varepsilon \leq \frac{L^2}{\rho}, \Delta_\Phi \geq \Phi(\mathbf{x}_0) - \Phi^\star$, and constant $c \leq c_{\max}$, with probability $1-\delta$, the output of PPD$(\mathbf{x}_0, L, \rho, \varepsilon, c, \delta, \Delta_f)$ will be a ε-second order stationary point, and terminate in iterations:*

$$\mathcal{O}\left(\frac{L(\Phi(\mathbf{x}_0) - \Phi^\star)}{\varepsilon^2} \ln^4\left(\frac{dL\Delta_\Phi}{\varepsilon^2\delta}\right)\right)$$

Remark Assuming $\varepsilon \leq \frac{L^2}{\rho}$ does not lead to loss of generality. Recall the second order condition is specified as $\lambda\big(\nabla^2 f(\mathbf{x}^\star)\big)_{\min} \geq -\sqrt{\rho\varepsilon}$, since when $\varepsilon \geq \frac{L^2}{\rho}$, we always have $-\sqrt{\rho\varepsilon} \leq -L \leq \lambda\big(\nabla^2 f(\mathbf{x}^\star)\big)_{\min}$, where the second inequality follows from the fact that the Lipschitz constant is the upper bound for $\lambda(\nabla^2 f(\mathbf{x}))$ in norm. Consequently, when $\varepsilon \geq \frac{L^2}{\rho}$, every ε-second-order stationary point is automatically a first order stationary point.

1.3.1 Sketch of the proof of main theorem

We consider the objective function $\Phi = f + g = f(\mathbf{x}) + \lambda\|\mathbf{x}\|_1$. When the magnitude of the ℓ_1 penalty term is small so that f dominates the geometric landscape of the function Φ, we expect that the characteristics of the objective function should not be too different from the without-ℓ_1 penalty case. At a high-level, we follow the proof of [JGN+17].

The key intuition is that when the iteration arrives in the vicinity of a saddle point, the volume of the trapping region surrounding the saddle point is small. As there is at least one direction for function value to continue decreasing $\big($e.g., the eigenvector corresponding to $\lambda\big(\nabla^2 f(\tilde{\mathbf{x}})\big)_{\min} \leq -\gamma\big)$, a random perturbation $\boldsymbol{\xi}$ added to the current iterate will likely have a component in the escape direction.

We first show that when the iteration arrives in the vicinity of a saddle point $\tilde{\mathbf{x}}$, before achieving sufficient decrease in function value, which partially determines a time threshold T, the proximal descent iteration will remain bounded around the saddle point; i.e., $\|\mathbf{u}_t - \tilde{\mathbf{x}}\| \leq$ const for all $t < T$.

Introducing a perturbation will take the current iteration point $\mathbf{u}_0$ to $\mathbf{w}_0 = \mathbf{u}_0 + \boldsymbol{\xi}$. We track the development of these two iteration sequences $\{\mathbf{u}_t\}$ and $\{\mathbf{w}_t\}$ when proximal descent is applied. We show that when the magnitude of the nonsmooth ℓ_1 term λ is less than a certain constant Λ, these two iteration sequences will stay at least a fixed distance apart at every step; i.e., $\|\mathbf{w}_t - \mathbf{u}_t\| \geq$ const for all $t < T$.

The central idea of proving the perturbed sequence will escape the saddle points is that after the perturbation is introduced, the projection of the iteration sequence in the escaping directions, i.e., on the subspace spanned by eigenvectors of negative eigenvalues of $\nabla^2 f(\tilde{\mathbf{x}})$, will gain more and more weight. To quantify this central idea, an key observation is that when magnitude of the ℓ_1 penalty term is small, the proximal step will preserve the monotonicity relation between increasing weight of iterations on escape-beneficial subspaces and the iteration progress.

Combining the previous two results, we show that at least one of these two iteration sequences will attain sufficient decrease in function value within the given time threshold T, to be followed by the argument that the probability of the chosen perturbation not letting the perturbed iteration sequence to escape the saddle point is small.

The key issue in the final step of the proof is to ensure the returned result will be an ε-second-order stationary point; in other words, we will show that whenever the current point is not an ε-second-order stationary point, the algorithm cannot terminate, combined with the proof that the proposed algorithm 1 will terminate within finitely many steps.

1.4 Technical proofs

For the proof of the main theorem, we introduce some notation and units for the simplicity of proof statement.

For matrices we use $\|\cdot\|$ to denote spectral norm. The operator $\mathcal{P}_{\mathcal{S}}(\cdot)$ denotes projection onto set $\mathcal{S}$. Define the local approximation of the smooth part of the objective function by

$$\tilde{f}_{\mathbf{x}}(\mathbf{y}) := f(\mathbf{x}) + \nabla^T f(\mathbf{x})(\mathbf{y}-\mathbf{x}) + \frac{1}{2}(\mathbf{y}-\mathbf{z})^T \mathcal{H}(\mathbf{y}-\mathbf{z}) \tag{1.4}$$

Units With the conditional number of the Hessian matrix $\kappa := \frac{L}{\gamma} \geq 1$, we define the following units for the convenience of proof statement:

$$\mathscr{F} := \eta L \frac{\gamma^3}{\rho^2} \cdot \ln^{-3}(\frac{d\kappa}{\delta}), \qquad \mathscr{G} := \sqrt{\eta L}\frac{\gamma^2}{\rho} \cdot \ln^{-2}(\frac{d\kappa}{\delta})$$

$$\mathscr{S} := \sqrt{\eta L}\frac{\gamma}{\rho} \cdot \ln^{-1}(\frac{d\kappa}{\delta}), \qquad \mathscr{T} := \frac{\ln(\frac{d\kappa}{\delta})}{\eta\gamma}$$

1.4.1 Lemma: Iterates remain bounded if stuck near a saddle point

Lemma 5. *For any constant $\hat{c} \geq 3$, there exists absolute constant $c_{\max}$: for any $\delta \in (0, \frac{d\kappa}{e}]$, let $f(\cdot), \tilde{\mathbf{x}}$ satisfies the condition in Lemma 10, for any initial point $\mathbf{u}_0$ with $\|\mathbf{u}_0 - \tilde{\mathbf{x}}\| \leq 2\mathscr{S}/(\kappa \cdot \ln(\frac{d\kappa}{\delta}))$, define:*

$$T = \min\left\{ \inf_t \left\{ t \,|\, \tilde{f}_{\mathbf{u}_0}(\mathbf{u}_t) - f(\mathbf{u}_0) + g(\mathbf{u}_t) - g(\mathbf{u}_0) \leq -3\mathscr{F} \right\}, \hat{c}\mathscr{T} \right\}$$

then, for any $\eta \leq c_{\max}/L$, we have for all $t < T$ that $\|\mathbf{u}_t - \tilde{\mathbf{x}}\| \leq 100(\mathscr{S} \cdot \hat{c})$.

Proof. We show if the function value did not decrease, then all the iteration updates must be constrained in a small ball. The proximal descent updates the solution as

$$\begin{aligned}\tilde{\mathbf{u}}_{t+1} &= \mathbf{u}_t - \nabla f(\mathbf{u}_t) = (I - \nabla f)(\mathbf{u}_t)\\ \mathbf{u}_{t+1} &= \operatorname{prox}_{\eta g}\big(\tilde{\mathbf{u}}_{t+1}\big) = \operatorname{prox}_{\eta g} \circ (I - \nabla f)(\mathbf{u}_t)\end{aligned}$$

Without losing of generality, set $\mathbf{u}_0 = 0$ to be the origin. For any $t \in \mathbb{N}$,

$$\|\mathbf{u}_t - \mathbf{u}_0\| = \|\mathbf{u}_t - 0\| = \|\operatorname{prox}_{\eta g}(\tilde{\mathbf{u}}_t) - \operatorname{prox}_{\eta g}(0)\| \leq \|\tilde{\mathbf{u}}_t - 0\| = \|\tilde{\mathbf{u}}_t\|$$

Jin et al. prove in [JGN$^+$17] by induction that if $\|\mathbf{u}_t\| \leq 100(\mathscr{S} \cdot \hat{c})$, then $\|\tilde{\mathbf{u}}_{t+1}\| \leq 100(\mathscr{S} \cdot \hat{c})$. Consequently, $\|\mathbf{u}_{t+1}\| \leq 100(\mathscr{S} \cdot \hat{c})$.

We point out that it is implicitly assumed that $\frac{2\mathscr{S}}{\kappa \cdot \ln(\frac{d\kappa}{\delta})} \ll \hat{c}$, so that for all $t < T$, $\|\tilde{\mathbf{x}}\| \ll \|\mathbf{u}_t\|$, and the relation $\|\mathbf{u}_t - \tilde{\mathbf{x}}\| \leq \|\mathbf{u}_t\| + \|\tilde{\mathbf{x}}\| \leq 100(\mathscr{S} \cdot \hat{c})$ holds. □

1.4.2 Preparation for building pillars

Lemma 6 (Existence of lower bound for the difference sequence $\{\mathbf{v}_t\}_{t=1}^{T}$)**.** *For iteration sequences $\{\mathbf{w}_t\}$ and $\{\mathbf{u}_t\}$ defined in Lemma 8, define the difference sequence as*

$$\mathbf{v}_t = \mathbf{w}_t - \mathbf{u}_t$$

There exists a positive lower bound for $\{\mathbf{v}_t\}$ when $t < \hat{c}\mathscr{T}$.

Proof. To show that the lower bound for iteration difference $\{\mathbf{v}_t\}_{t=1}^{T}$ exists, we consider bounding the iteration sequence $\tilde{\mathbf{v}}_{t+1}$ first. Define the difference between the proximal of l_1 penalty term and its coimage as $\mathcal{D}_g[\mathbf{x}] = \operatorname{prox}_g[\mathbf{x}] - \mathbf{x} = \min\{\lambda\mathbb{1}, |\mathbf{x}|\} \otimes \operatorname{sgn}(-\mathbf{x})$, where $\otimes$ is Hadamard product and the minimum is taken elementwise. We notice that $\|\mathcal{D}_{\eta\lambda\|\cdot\|_1}[\mathbf{x}]\| \leq \eta\lambda\sqrt{d}$. Thus, $\|\mathbf{w}_k - \mathbf{u}_k\| =$

$\|\tilde{\mathbf{w}}_k - \tilde{\mathbf{v}}_k - \lambda(\mathcal{D}_{\eta g}[\tilde{\mathbf{w}}_k] - \mathcal{D}_{\eta g}[\tilde{\mathbf{u}}_k])\| \geq \|\tilde{\mathbf{w}}_k - \tilde{\mathbf{v}}_k\| - 2\eta\lambda\sqrt{d}$.

$$
\begin{aligned}
\|\tilde{\mathbf{v}}_{t+1}\| &= \|\tilde{\mathbf{w}}_{t+1} - \tilde{\mathbf{u}}_{t+1}\| \\
&= \|(I - \eta\nabla f) \circ \operatorname{prox}_{\eta g}(\tilde{\mathbf{w}}_k) - (I - \eta\nabla f) \circ \operatorname{prox}_{\eta g}(\tilde{\mathbf{u}}_k)\| \\
&= \|\mathbf{w}_k - \mathbf{u}_k - \eta(\nabla f(\mathbf{w}_k) - \nabla f(\mathbf{u}_k))\| \\
&\geq \|\mathbf{w}_k - \mathbf{u}_k\| - \eta L\|\mathbf{w}_k - \mathbf{u}_k\| = (1 - \eta L)\|\mathbf{w}_k - \mathbf{u}_k\| \\
&\geq (1 - \eta L)(\|\tilde{\mathbf{w}}_k - \tilde{\mathbf{u}}_k\| - 2\eta\lambda\sqrt{d}) = (1 - \eta L)(\|\tilde{\mathbf{v}}_k\| - 2\eta\lambda\sqrt{d}) \\
&\geq (1 - \eta L)^t\|\tilde{\mathbf{v}}_1\| - 2\eta\lambda\sqrt{d}\sum_{i=1}^{t}(1 - \eta L)^i \\
&= (1 - \eta L)^t\|\tilde{\mathbf{v}}_1\| - 2\lambda\sqrt{d}\frac{(1 - \eta L)\big(1 - (1 - \eta L)^t\big)}{L}
\end{aligned}
$$

As $\tilde{\mathbf{v}}_1 = (I - \eta\nabla f)\mathbf{v}_0 = (I - \eta\nabla f)\mu r \mathrm{e}_1 = \mu r(\mathrm{e}_1 - \eta\nabla^2 f(\boldsymbol{\xi})\theta\mathrm{e}_1) = \mu r(1 + \eta\gamma\theta)\mathrm{e}_1$, where $\theta \in (0, 1)$, we have

$$
\|\tilde{\mathbf{v}}_{t+1}\| \geq (1 - \eta L)^t\mu r(1 + \eta\gamma\theta) - 2\lambda\sqrt{d}\frac{(1 - \eta L)(1 - (1 - \eta L)^t)}{\eta L} \tag{1.5}
$$

To compare $\|\mathbf{v}_t\|$ and $\|\tilde{\mathbf{v}}_t\|$,

$$
\|\mathbf{v}_{t+1}\| \geq \|\tilde{\mathbf{v}}_{t+1}\| - 2\eta\lambda\sqrt{d} \geq (1 - \eta L)^t\mu r(1 + \eta\gamma\theta) - 2\lambda\sqrt{d}\frac{(1 - \eta L)(1 - (1 - \eta L)^t) + \eta L}{L} \tag{1.6}
$$

Therefore, as long as

$$
\lambda < \frac{(1 - \eta L)^{\hat{c}\mathscr{T}}\mu\frac{1}{\kappa(\ln\frac{d\kappa}{\delta})^2}\sqrt{\eta}L^{\frac{3}{2}}\frac{\gamma}{\rho}(1 + \eta\gamma\theta)}{2\sqrt{d}[(1 - \eta L)(1 - (1 - \eta L)^{\hat{c}\mathscr{T}}) + \eta L]} \tag{1.7}
$$

the difference sequence $\{\|\mathbf{v}_t\|\}$ has a positive lower bound on its norm. □

Lemma 7 (Preservation of subspace projection monotonicity after prox of l_1 in rotated coordinate with small λ)**.** *Denote the subspace of $\mathbb{R}^n$ spanned by $\{\mathrm{e}_1\}$ as $\mathbb{E}$, while the complement subspace spanned by $\{\mathrm{e}_2, \cdots, \mathrm{e}_n\}$ as $\mathbb{E}^\perp$. For a given vector $\mathbf{x}$ chosen from a lower bounded set $\mathcal{X}$, i.e. $\forall\, \mathbf{x} \in \mathcal{X}$, $\|\mathbf{x}\| \geq C$ for some constant $C > 0$, assume $\|\mathcal{P}_{\mathbb{E}^\perp}\mathbf{x}\| \leq K\|\mathcal{P}_{\mathbb{E}}\mathbf{x}\|$, where $0 < K \leq 1$ is a constant. If the parameter λ for the l_1 penalty term is small enough, then*

$$
\|\mathcal{P}_{\mathbb{E}^\perp}\operatorname{prox}_{\eta g}(\mathbf{x})\| \leq K\|\mathcal{P}_{\mathbb{E}}\operatorname{prox}_{\eta g}(\mathbf{x})\|
$$

Proof. We want to find a constraint on λ such that when λ is small enough, if the projection in the original coordinate demonstrates the monotonicity relation $\|\mathcal{P}_{\mathbb{E}}\mathbf{x}\| \leq \|\mathcal{P}_{\mathbb{E}^\perp}\mathbf{x}\|$, this monotonicity relation will be preserved after proximal operator of l_1 is applied on the input vector.

Naturally there exists a normal vector, denoted as $\hat{\boldsymbol{n}}_{\text{boundary}} \equiv \hat{\boldsymbol{n}}$, for the boundary hyperplane on which $\|\mathcal{P}_{\mathbb{E}}\mathbf{x}\| = K\|\mathcal{P}_{\mathbb{E}^\perp}\mathbf{x}\|$. By moving along $\hat{\boldsymbol{n}}$, a point approaches the boundary most efficiently. Any vector inside the hyperplane is perpendicular to $\hat{\boldsymbol{n}}$, which we denote as $\hat{\boldsymbol{n}}^\perp$.

Define

$$\hat{\mathbf{v}}_{\text{move}}(\mathbf{x}) = \begin{cases} -\eta\lambda \cdot \text{sgn}(x_i) & \text{if } |x_i| \geq \eta\lambda \\ -x_i & \text{if } |x_i| < \eta\lambda \end{cases} = \min\{|x|, \eta\lambda\mathbb{1}\} \otimes \text{sgn}(-\mathbf{x}) \tag{1.8}$$

where $\otimes$ is the Hadamard product, and the minimum is taken elementwise. Because $\text{prox}_{\eta g}(\mathbf{x}) = \mathbf{x} + \hat{\mathbf{v}}_{\text{move}}$, a sufficient condition to be imposed on λ to guarantee the preservation of projection monotonicity $\|\mathcal{P}_{\mathbb{E}^\perp}\text{prox}_{\eta g}(\mathbf{x})\| \leq K\|\mathcal{P}_{\mathbb{E}}\text{prox}_{\eta g}(\mathbf{x})\|$ is that

$$\lambda < \left\| \frac{\text{Proj}_{\boldsymbol{n}}\mathbf{x}}{\hat{\mathbf{v}}_{\text{move}} \cdot \hat{\boldsymbol{n}}} \right\| = \left\| \frac{\mathbf{x} \cdot \hat{\boldsymbol{n}}}{\hat{\mathbf{v}}_{\text{move}} \cdot \hat{\boldsymbol{n}}} \right\| \leq \frac{\|\mathbf{x}\|}{\|\hat{\mathbf{v}}_{\text{move}} \cdot \hat{\boldsymbol{n}}\|}$$

which means the moving distance caused by applying the l_1 proximal operator (soft shrinkage) projected on the direction of $\hat{\boldsymbol{n}}$ is less that the distance between $\mathbf{x}$ to the boundary hyperplane, hence rendering the vector stay on the same side of the boundary after moving.

Therefore, as long as

$$\lambda < \frac{C}{\|\hat{\mathbf{v}}_{\text{move}} \cdot \hat{\boldsymbol{n}}\|} \tag{1.9}$$

the monotonicity of projection onto subspaces can be preserved.

□

Remark 1 for Lemma 7 As an examples in $\mathbb{R}^2$, set $K = 1$, we visualise the shift caused by proximal operator and the boundary of projection-monotonicity preserving region. Assume $\mathrm{e}_{1,2}$ are orthonormal basis of Cartesian coordinate in the standard position. The directional vector for region division boundary is $\hat{\mathrm{e}}_{\text{boundary}} = \hat{\boldsymbol{n}}^{\perp} = \dfrac{\pm\hat{\mathrm{e}}_1 \pm \hat{\mathrm{e}}_2}{\sqrt{2}}$, and $\hat{\mathrm{e}}^{\perp}_{\text{boundary}} = \hat{\boldsymbol{n}}$ is the corresponding perpendicular directional vector. For l_1 norm, $\hat{\mathbf{v}}_{\text{move}}$ is $(\pm 1, \pm 1)$.

Remark 2 for Lemma 7 We point out that the upper bound for the parameter λ is related to the alignment of the eigenspace of $\mathcal{H}$. If the eigenspace of $\mathcal{H}$ is aligned with canonical orthonormal basis of $\mathbb{R}^d$, then $\lambda \in (0, \infty)$. The most stringent restriction on the upper bound of λ applies when $\hat{\mathbf{v}}_{\text{move}}$ is parallel to $\hat{\boldsymbol{n}}$.

1.4.3 Lemma: perturbed iterates will escape the saddle point

Lemma 8. *There exists absolute constant $c_{\max}, \hat{c}$ such that: for any $\delta \in (0, \frac{d\kappa}{e}]$, let $f(\cdot), \tilde{\mathbf{x}}$ satisfies the condition in Lemma 10, and sequences $\{\mathbf{u}_t\}, \{\mathbf{w}_t\}$ satisfy the conditions in Lemma 10, define:*

$$T = \min\left\{ \inf_t \left\{ t | \tilde{f}_{\mathbf{w}_0}(\mathbf{w}_t) + g(\mathbf{w}_t) - f(\mathbf{w}_0) - g(\mathbf{w}_0) \leq -3\mathscr{F} \right\}, \hat{c}\mathscr{T} \right\}$$

then, for any $\eta \leq c_{\max}/L$, if $\|\mathbf{u}_t - \tilde{\mathbf{x}}\| \leq 100(\mathscr{S} \cdot \hat{c})$ for all $t < T$, we will have $T < \hat{c}\mathscr{T}$.

Proof. We show that if the iterate sequence before time T starting from $\mathbf{u}_0$ does not provide sufficient function value decrease, the other iterate sequence, which starts from $\mathbf{w}_0$, will be able to achieve the function value decrease purpose. Ultimately, we will prove $T < \hat{c}\mathscr{T}$. We establish the inequality about T by considering the difference between $\mathbf{w}_t$ and $\mathbf{u}_t$. Define $\mathbf{v}_t = \mathbf{w}_t - \mathbf{u}_t$. The assumption of the lemma 8, $\mathbf{v}_0 = \mu[\mathscr{S}/(\kappa \cdot \ln(\frac{d\kappa}{\delta}))]\mathrm{e}_1$, $\mu \in [\delta/(2\sqrt{d}), 1]$.

We bound $\|\mathbf{v}_t\|$ from both sides for all $t < T$ to obtain an inequality about T.

Recall that the proximal descent updates the solution as

$$\tilde{\mathbf{u}}_{t+1} = \mathbf{u}_t - \nabla f(\mathbf{u}_t) = (I - \eta \nabla f)(\mathbf{u}_t)$$

$$\mathbf{u}_{t+1} = \mathrm{prox}_{\eta g}\big(\tilde{\mathbf{u}}_{t+1}\big) = \mathrm{prox}_{\eta g} \circ (I - \eta \nabla f)(\mathbf{u}_t)$$

Simple algebraic computation gives

$$\tilde{\mathbf{v}}_{t+1} = (I - \eta\mathcal{H} - \eta\Delta'_t)\mathbf{v}_t \tag{1.10}$$

where $\Delta'_t = \int_0^1 \nabla^2 f(\mathbf{u}_t + \theta \mathbf{v}_t)\, \mathrm{d}\theta - \mathcal{H}$, and $\tilde{\mathbf{v}}_t = \tilde{\mathbf{w}}_t - \tilde{\mathbf{u}}_t$.

Consider $\|\tilde{\mathbf{u}}_t\|$ and $\|\tilde{\mathbf{w}}_t\|$. Because $\mathbf{v}_0 = \tilde{\mathbf{v}}_0$, we have $\|\tilde{\mathbf{w}}_0 - \tilde{\mathbf{x}}\| \leq \|\tilde{\mathbf{u}}_0 - \tilde{\mathbf{x}}\| + \|\tilde{\mathbf{v}}_0\| \leq 2\mathscr{S}/(\kappa \cdot \ln(\frac{d\kappa}{\delta}))$. With same logic in the proof for lemma 5, we see $\|\tilde{\mathbf{u}}_t\| \leq 100(\mathscr{S} \cdot \hat{c})$, and $\|\tilde{\mathbf{w}}_t\| \leq 100(\mathscr{S} \cdot \hat{c})$. (Same relation hold for $\|\mathbf{u}_t\|$ and $\|\mathbf{w}_t\|$ respectively.) As a result, $\|\tilde{\mathbf{v}}_t\| \leq \|\tilde{\mathbf{w}}_t\| + \|\tilde{\mathbf{u}}_t\| \leq 200(\mathscr{S} \cdot \hat{c})$ for all $t < T$. Also,

$$\|\mathbf{v}_t\| \leq 200(\mathscr{S} \cdot \hat{c}) \tag{1.11}$$

Equation (1.11) and Hessian Lipschitz gives for $t < T$, $\|\Delta'_t\| \leq \rho(\|\mathbf{u}_t\| + \|\mathbf{v}_t\| + \|\tilde{\mathbf{x}}\|) \leq \rho\mathscr{S}(300\hat{c} + 1) = \frac{\zeta}{\eta}$, where $\zeta = \eta\rho\mathscr{S}(300\hat{c} + 1)$.

Denote ψ_t be the norm of $\mathbf{v}_t$ projected onto e_1 direction (§), and φ_t be the norm of $\mathbf{v}_t$ projected onto the remaining subspace ($§^c$), while $\tilde{\psi}_t$ be the norm of $\tilde{\mathbf{v}}_t$ projected onto §, and $\tilde{\varphi}_t$ be the norm of $\tilde{\mathbf{v}}_t$ projected onto $§^c$.

Equation (1.10) gives

$$\tilde{\psi}_{t+1} \geq (1 + \gamma\eta)\psi_t - \zeta\sqrt{\psi_t^2 + \varphi_t^2} \tag{1.12}$$

$$\tilde{\varphi}_{t+1} \leq (1 + \gamma\eta)\varphi_t + \zeta\sqrt{\psi_t^2 + \varphi_t^2} \tag{1.13}$$

To obtain the lower bound of $\|\mathbf{v}_t\|$, we prove the following relation as preparation:

$$\text{for all } t < T, \quad \varphi_t \leq 4\zeta t \cdot \psi_t \tag{1.14}$$

By hypothesis of lemma 8, we know $\varphi_0 = 0$, thus the base case of induction holds. Assume equation (1.14) is true for $\tau \leq t$, for $t + 1 \leq T$, we have

$$\begin{aligned} \tilde{\varphi}_{t+1} &\leq 4\zeta t(1 + \gamma\eta)\psi_t + \zeta\sqrt{\psi_t^2 + \varphi_t^2} \\ 4\zeta(t+1)\left[(1 + \gamma\eta)\psi_t - \zeta\sqrt{\psi_t^2 + \varphi_t^2}\right] &\leq 4\zeta(t+1)\tilde{\psi}_{t+1} \end{aligned} \tag{1.15}$$

By choosing $\sqrt{c_{\max}} \leq \frac{1}{300\hat{c}+1} \min\{\frac{1}{2\sqrt{2}}, \frac{1}{4\hat{c}}\}$, and $\eta \leq \frac{c_{\max}}{L}$, we have $4\zeta(t+1) \leq 4\zeta T \leq 4\eta\rho\mathscr{S}(300\hat{c} + 1)\hat{c}\mathscr{T} = 4\sqrt{\eta L}(300\hat{c} + 1)\hat{c} \leq 1$. This gives $4(1 + \gamma\eta)\psi_t \geq 4\psi_t \geq (1 + 1)\sqrt{2\psi_t^2} \geq (1 + 4\zeta(t +$

$1))\sqrt{\psi_t^2+\varphi_t^2}$. i.e.

$$(1+4\zeta(t+1))\sqrt{\psi_t^2+\varphi_t^2} \leq 4\psi_t \tag{1.16}$$

Connecting two parts of equation (1.15), we obtain

$$\tilde{\varphi}_{t+1} \leq 4\zeta(t+1)\tilde{\psi}_{t+1} \tag{1.17}$$

Now we switch our focus to the eigenspace of Hessian $\mathcal{H}$. Assume the orthonormal basis for the eigensapce of $\mathcal{H}$ is $\{\mathrm{e}_1, \mathrm{e}_2, \cdots, \mathrm{e}_d\}$. The order of dimension aligns with the increasing order of the corresponding eigenvalues. This coordinate transformation does not lead to loss of generality, as it is unitary.

By lemma 6, we know the iteration difference sequence $\mathbf{v}_t$ has a positive lower bound in terms of 2-norm. Therefore, by lemma 7, with the virtue of equation (1.17) $\sqrt{\sum_{i=2}^d(\mathrm{e}_i^T\tilde{\mathbf{v}}_{t+1})^2} \leq 4\zeta(t+1)\|\mathrm{e}_1^T\tilde{\mathbf{v}}_{t+1}\|$, we still have the projection monotonicity on the subspace of eigenspace of $\mathcal{H}$, i.e.

$$\varphi_{t+1} = \sqrt{\sum_{i=2}^d(\mathrm{e}_i^T \mathrm{prox}_g(\tilde{\mathbf{v}}_{t+1}))^2} \leq 4\zeta(t+1)\|\mathrm{e}_1^T\mathrm{prox}_g(\tilde{\mathbf{v}}_{t+1})\| = 4\zeta(t+1)\psi_{t+1}$$

Until here we finish the induction.

Recall that $4\zeta(t+1) \leq 1$, we thus have $\varphi_t \leq 4\zeta t\psi_t \leq \psi_t$, which gives

$$\psi_{t+1} \geq (1+\gamma\eta)\psi_t - \sqrt{2}\zeta\psi_t \geq \big(1+\frac{\gamma\eta}{2}\big)\psi_t \tag{1.18}$$

where the last inequality follows from $\zeta = \eta\rho\mathscr{S}(300\hat{c}+1) \leq \sqrt{c_{\max}}(300\hat{c}+1)\gamma\eta \cdot \ln^{-1}(\frac{d\kappa}{\delta}) \leq \frac{\gamma\eta}{2\sqrt{2}}$.

Finally, combining (1.11) and (1.18), we have for all $t < T$:

$$\begin{aligned}200(\mathscr{S}\cdot\hat{c}) \geq \|\mathbf{v}_t\| \geq \psi_t &\geq (1+\frac{\gamma\eta}{2})^t\psi_0 = (1+\frac{\gamma\eta}{2})^t c_0\frac{\mathscr{S}}{\kappa}\ln^{-1}\left(\frac{d\kappa}{\delta}\right) \\ &\geq (1+\frac{\gamma\eta}{2})^t\frac{\delta}{2\sqrt{d}}\frac{\mathscr{S}}{\kappa}\ln^{-1}\left(\frac{d\kappa}{\delta}\right)\end{aligned}$$

This implies

$$T < \frac{1}{2}\frac{\ln[400\frac{\kappa\sqrt{d}}{\delta}\cdot\hat{c}\ln(\frac{d\kappa}{\delta})]}{\ln(1+\frac{\gamma\eta}{2})} \leq \frac{\ln[400\frac{\kappa\sqrt{d}}{\delta}\cdot\hat{c}\ln(\frac{d\kappa}{\delta})]}{\gamma\eta} \leq (2+\ln(400\hat{c}))\mathscr{T}$$

The last inequality is due to $\delta \in (0, \frac{d\kappa}{e}]$, we have $\ln(\frac{d\kappa}{\delta}) \geq 1$. By choosing the constant $\hat{c}$ to be large enough to satisfy $2+\ln(400\hat{c}) \leq \hat{c}$, we will have $T < \hat{c}\mathscr{T}$, which finishes the proof.

□

1.4.4 Combining previous results

Lemma 9. *There exists a universal constant $c_{\max}$, for any $\delta \in (0, \frac{d\kappa}{e}]$, let $f(\cdot), \tilde{\mathbf{x}}$ satisfies the conditions in Lemma 10, and without loss of generality let e_1 be the minimum eigenvector of $\nabla^2 f(\tilde{\mathbf{x}})$. Consider two gradient descent sequences $\{\mathbf{u}_t\}, \{\mathbf{w}_t\}$ with initial points $\mathbf{u}_0, \mathbf{w}_0$ satisfying: (denote radius $r = \mathscr{S}/(\kappa \cdot \ln(\frac{d\kappa}{\delta})))$*

$$\|\mathbf{u}_0 - \tilde{\mathbf{x}}\| \leq r, \quad \mathbf{w}_0 = \mathbf{u}_0 + \mu \cdot r \cdot \mathrm{e}_1, \quad \mu \in [\delta/(2\sqrt{d}), 1]$$

Then, for any stepsize $\eta \leq c_{\max}/L$, and any $T \geq \frac{1}{c_{\max}} \mathscr{T}$, we have:

$$\min\{f(\mathbf{u}_T) + g(\mathbf{u}_T) - f(\mathbf{u}_0) - g(\mathbf{u}_0), f(\mathbf{w}_T) + g(\mathbf{w}_T) - f(\mathbf{w}_0) - g(\mathbf{w}_0)\} \leq -2.7\mathscr{F}$$

Proof. Without losing generality, let $\tilde{\mathbf{x}} = 0$ be the origin. Let $(c^{(2)}_{\max}, \hat{c})$ be the absolute constant so that Lemma 8 holds, also let $c^{(1)}_{\max}$ be the absolute constant to make Lemma 5 holds based on our current choice of $\hat{c}$. We choose $c_{\max} \leq \min\{c^{(1)}_{\max}, c^{(2)}_{\max}\}$ so that our learning rate $\eta \leq c_{\max}/L$ is small enough which make both Lemma 5 and Lemma 8 hold. Let $T^\star := \hat{c}\mathscr{T}$ and define:

$$T' = \inf_t \left\{ t | \tilde{f}_{\mathbf{u}_0}(\mathbf{u}_t) + g(\mathbf{u}_t) - f(\mathbf{u}_0) - g(\mathbf{u}_0) \leq -3\mathscr{F} \right\}$$

Let's consider following two cases:

Case $T' \leq T^\star$: In this case, by Lemma 5, we know $\|\mathbf{u}_{T'-1}\| \leq O(\mathscr{S})$, and therefore

$$\|\mathbf{u}_{T'}\| \leq \|\mathbf{u}_{T'-1}\| + \eta \|\nabla f(\mathbf{u}_{T'-1})\| \leq \|\mathbf{u}_{T'-1}\| + \eta \|\nabla f(\tilde{\mathbf{x}})\| + \eta L \|\mathbf{u}_{T'-1}\| \leq O(\mathscr{S})$$

By choosing $c_{\max}$ small enough and $\eta \leq c_{\max}/L$, this gives:

$$\begin{aligned}
& f(\mathbf{u}_{T'}) + g(\mathbf{u}_{T'}) - f(\mathbf{u}_0) - g(\mathbf{u}_0) \\
\leq & \nabla f(\mathbf{u}_0) \intercal (\mathbf{u}_{T'} - \mathbf{u}_0) + \frac{1}{2} (\mathbf{u}_{T'} - \mathbf{u}_0) \intercal \nabla^2 f(\mathbf{u}_0)(\mathbf{u}_{T'} - \mathbf{u}_0) + \frac{\rho}{6} \|\mathbf{u}_{T'} - \mathbf{u}_0\|^3 + g(\mathbf{u}_{T'}) - g(\mathbf{u}_0) \\
\leq & \tilde{f}_{\mathbf{u}_0}(\mathbf{u}_{T'}) - f(\mathbf{u}_0) + g(\mathbf{u}_{T'}) - g(\mathbf{u}_0) + \frac{\rho}{2} \|\mathbf{u}_0 - \tilde{\mathbf{x}}\| \|\mathbf{u}_{T'} - \mathbf{u}_0\|^2 + \frac{\rho}{6} \|\mathbf{u}_{T'} - \mathbf{u}_0\|^3 \\
\leq & -3\mathscr{F} + O(\rho \mathscr{S}^3) = -3\mathscr{F} + O(\sqrt{\eta L} \cdot \mathscr{F}) \leq -2.7\mathscr{F}
\end{aligned}$$

The first and second inequality exploit Hessian Lipschitz property of smooth function f, and $\|\mathbf{u}_0 - \tilde{\mathbf{x}}\| \leq O(\mathscr{S})$, $\|\mathbf{u}_{T'} - \mathbf{u}_0\| \leq O(\mathscr{S})$. By choose $c_{\max} \leq \min\{1, \frac{1}{\hat{c}}\}$. We know $\eta < \frac{1}{L}$, by *sufficient* *decrease lemma* for proximal descent, we know each proximal descent iteration decreases function value. Therefore, for any $T \geq \frac{1}{c_{\max}}\mathscr{T} \geq \hat{c}\mathscr{T} = T^{\star} \geq T'$, we have:

$$\Phi(\mathbf{u}_T) - \Phi(\mathbf{u}_0) \leq \Phi(\mathbf{u}_{T^\star}) - \Phi(\mathbf{u}_0) \leq \Phi(\mathbf{u}_{T'}) - \Phi(\mathbf{u}_0) \leq -2.7\mathscr{F}$$

Case $T' > T^\star$: In this case, by Lemma 5, we know $\|\mathbf{u}_t\| \leq O(\mathscr{S})$ for all $t \leq T^\star$. Define

$$T'' = \inf_t \left\{ t | \tilde{f}_{\mathbf{w}_0}(\mathbf{w}_t) + g(\mathbf{w}_t) - f(\mathbf{w}_0) - g(\mathbf{w}_0) \leq -3\mathscr{F} \right\}$$

By Lemma 8, we immediately have $T'' \leq T^\star$. Apply same argument as in the case $T' \leq T^\star$, we have for all $T \geq \frac{1}{c_{\max}}\mathscr{T}$ that $f(\mathbf{w}_T) + g(\mathbf{w}_T) - f(\mathbf{w}_0) - g(\mathbf{w}_0) \leq f(\mathbf{w}_{T^\star}) + g(\mathbf{w}_{T^\star}) - f(\mathbf{w}_0) - g(\mathbf{w}_0) \leq -2.7\mathscr{F}$. □

1.4.5 Main lemma

Lemma 10 (Main Lemma). *There exists universal constant $c_{\max}$, for $f(\cdot)$ satisfies 1, for any $\delta \in (0, \frac{d\kappa}{e}]$, suppose we start with point $\tilde{\mathbf{x}}$ satisfying following conditions:*

$$\|G(\tilde{\mathbf{x}})\| = \left\| L\left(\tilde{\mathbf{x}} - \text{prox}_{\frac{1}{L}g}\left(\tilde{\mathbf{x}} - \frac{1}{L}\nabla f(\tilde{\mathbf{x}})\right)\right) \right\| \leq \mathscr{G} \quad \text{and} \quad \lambda_{\min}(\nabla^2 f(\tilde{\mathbf{x}})) \leq -\gamma$$

Let $\mathbf{x}_0 = \tilde{\mathbf{x}} + \boldsymbol{\xi}$ where $\boldsymbol{\xi}$ come from the uniform distribution over ball with radius $\mathscr{S}/(\kappa \cdot \ln(\frac{d\kappa}{\delta}))$, and let $\mathbf{x}_t$ be the iterates of gradient descent from $\mathbf{x}_0$. Then, when stepsize $\eta \leq c_{\max}/L$, with at least probability $1 - \delta$, we have following for any $T \geq \frac{1}{c_{\max}}\mathscr{T}$:

$$f(\mathbf{x}_T) + g(\mathbf{x}_T) - f(\tilde{\mathbf{x}}) - g(\tilde{\mathbf{x}}) \leq -\mathscr{F}$$

Proof. Denote $T_{\frac{l}{L}}(\mathbf{x}) = \text{prox}_{\frac{1}{L}g}\left[\mathbf{x} - \frac{1}{L}\nabla f(\mathbf{x})\right]$. The fisrt order stationary condition is equivalent to $\|\tilde{\mathbf{x}} - T_{\frac{1}{L}}(\tilde{\mathbf{x}})\| = \|\nabla f(\tilde{\mathbf{x}}) + \partial g\big(T_{\frac{1}{L}}(\tilde{\mathbf{x}})\big)\| \leq \mathscr{G}$, where ∂g is the subgradient of the function g.

As $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$ has Lipschitz constant λ, we have

$$f(\mathbf{x}_0) + g(\mathbf{x}_0) \leq f(\tilde{\mathbf{x}}) + \langle \nabla f(\tilde{\mathbf{x}}), \boldsymbol{\xi} \rangle + \frac{L}{2}\|\boldsymbol{\xi}\|^2 + g(\tilde{\mathbf{x}}) + \langle \partial g(\tilde{\mathbf{x}}), \boldsymbol{\xi} \rangle + \frac{\lambda}{2}\|\boldsymbol{\xi}\|^2$$

Notice

$$\begin{aligned}\|\nabla f(\tilde{\mathbf{x}}) + \partial g(\tilde{\mathbf{x}})\| &= \|\nabla f(\tilde{\mathbf{x}}) + \partial g(T_{\frac{l}{L}}(\mathbf{x})) - \big(\partial g(T_{\frac{l}{L}}(\mathbf{x})) - \partial g(\tilde{\mathbf{x}})\big)\| \\ &\leq \mathscr{G} + \lambda\mathscr{G}\end{aligned}$$

By adding perturbation, in worst case we increase function value by:

$$\begin{aligned}
f(\mathbf{x}_0) - f(\tilde{\mathbf{x}}) + g(\mathbf{x}_0) - g(\tilde{\mathbf{x}}) &\leq \|\nabla f(\tilde{\mathbf{x}}) + \partial g(\tilde{\mathbf{x}})\| \|\xi\| + \frac{L+\lambda}{2}\|\xi\|^2 \\
&\leq (1+\lambda)\mathscr{G}(\frac{\mathscr{S}}{\kappa \cdot \ln(\frac{d\kappa}{\delta})}) + \frac{1}{2}(L+\lambda)(\frac{\mathscr{S}}{\kappa \cdot \ln(\frac{d\kappa}{\delta})})^2 \\
&\leq (\frac{3}{2} + \frac{1}{5})\mathscr{F}
\end{aligned}$$

where the last inequality follows from the fact that $\lambda \ll \min\{1, l\}$ per equation (1.7).

On the other hand, let radius $r = \frac{\mathscr{S}}{\kappa \cdot \ln(\frac{d\kappa}{\delta})}$. We know $\mathbf{x}_0$ come from uniform distribution over $\mathbb{B}_{\tilde{\mathbf{x}}}(r)$. Let $\mathcal{X}_{\text{stuck}} \subset \mathbb{B}_{\tilde{\mathbf{x}}}(r)$ denote the set of bad starting points so that if $\mathbf{x}_0 \in \mathcal{X}_{\text{stuck}}$, then $\Phi(\mathbf{x}_T) - \Phi(\mathbf{x}_0) > -2.7\mathscr{F}$ (thus stuck at a saddle point); otherwise if $\mathbf{x}_0 \in B_{\tilde{\mathbf{x}}}(r) - \mathcal{X}_{\text{stuck}}$, we have $\Phi(\mathbf{x}_T) - \Phi(\mathbf{x}_0) \leq -2.7\mathscr{F}$.

By applying Lemma 9, we know for any $\mathbf{x}_0 \in \mathcal{X}_{\text{stuck}}$, it is guaranteed that $(\mathbf{x}_0 \pm \mu r \mathrm{e}_1) \notin \mathcal{X}_{\text{stuck}}$ where $\mu \in [\frac{\delta}{2\sqrt{d}}, 1]$. Denote $I_{\mathcal{X}_{\text{stuck}}}(\cdot)$ be the indicator function of being inside set $\mathcal{X}_{\text{stuck}}$; and vector $\mathbf{x} = (x^{(1)}, \mathbf{x}^{(-1)})$, where $x^{(1)}$ is the component along e_1 direction, and $\mathbf{x}^{(-1)}$ is the remaining $d-1$ dimensional vector. Recall $\mathbb{B}^{(d)}(r)$ be d-dimensional ball with radius r; By calculus, this gives an upper bound on the volume of $\mathcal{X}_{\text{stuck}}$:

$$\begin{aligned}
\text{Vol}(\mathcal{X}_{\text{stuck}}) &= \int_{\mathbb{B}^{(d)}_{\tilde{\mathbf{x}}}(r)} \mathrm{d}\mathbf{x} \cdot I_{\mathcal{X}_{\text{stuck}}}(\mathbf{x}) \\
&= \int_{\mathbb{B}^{(d-1)}_{\tilde{\mathbf{x}}}(r)} \mathrm{d}\mathbf{x}^{(-1)} \int_{\tilde{x}^{(1)} - \sqrt{r^2 - \|\tilde{\mathbf{x}}^{(-1)} - \mathbf{x}^{(-1)}\|^2}}^{\tilde{x}^{(1)} + \sqrt{r^2 - \|\tilde{\mathbf{x}}^{(-1)} - \mathbf{x}^{(-1)}\|^2}} \mathrm{d}x^{(1)} \cdot I_{\mathcal{X}_{\text{stuck}}}(\mathbf{x}) \\
&\leq \int_{\mathbb{B}^{(d-1)}_{\tilde{\mathbf{x}}}(r)} \mathrm{d}\mathbf{x}^{(-1)} \cdot \left(2 \cdot \frac{\delta}{2\sqrt{d}} r\right) = \text{Vol}(\mathbb{B}^{(d-1)}_0(r)) \times \frac{\delta r}{\sqrt{d}}
\end{aligned}$$

Then, we immediately have the ratio:

$$\frac{\text{Vol}(\mathcal{X}_{\text{stuck}})}{\text{Vol}(\mathbb{B}^{(d)}_{\tilde{\mathbf{x}}}(r))} \leq \frac{\frac{\delta r}{\sqrt{d}} \times \text{Vol}(\mathbb{B}^{(d-1)}_0(r))}{\text{Vol}(\mathbb{B}^{(d)}_0(r))} = \frac{\delta}{\sqrt{\pi d}} \frac{\Gamma(\frac{d}{2}+1)}{\Gamma(\frac{d}{2}+\frac{1}{2})} \leq \frac{\delta}{\sqrt{\pi d}} \cdot \sqrt{\frac{d}{2} + \frac{1}{2}} \leq \delta$$

The second last inequality is by the property of Gamma function that $\frac{\Gamma(x+1)}{\Gamma(x+1/2)} < \sqrt{x + \frac{1}{2}}$ as long as $x \geq 0$. Therefore, with at least probability $1-\delta$, $\mathbf{x}_0 \notin \mathcal{X}_{\text{stuck}}$. In this case, we have:

$$\begin{aligned}\Phi(\mathbf{x}_T) - \Phi(\tilde{\mathbf{x}}) &= \Phi(\mathbf{x}_T) - \Phi(\mathbf{x}_0) + \Phi(\mathbf{x}_0) - \Phi(\tilde{\mathbf{x}}) \\ &\leq -2.7\mathscr{F} + 1.7\mathscr{F} \\ &\leq -\mathscr{F}\end{aligned}$$

which finishes the proof.

□

1.4.6 Main theorem, and its proof

Lemma 11 (Sufficient Decrease Lemma for Proximal Descent, [Bec17]). *Assume the function f is real-valued and lower semi-continuous. Then for any $L \in (\frac{L}{2}, \infty)$ where $\eta = \frac{1}{L}$, we have $\Phi(\mathbf{x}_t) - \Phi(\mathbf{x}_{t+1}) \geq \frac{L-\frac{L}{2}}{L^2}\|G_{\frac{1}{L}}(\mathbf{x}_t)\|$.*

1.4.6.1 Proof of the main theorem

Proof. Denote $\tilde{c}_{\max}$ to be the absolute constant allowed in lemma 10 when it is given following parameters $\eta = \frac{c}{L}$, $\gamma = \sqrt{\rho\varepsilon}$, and $\delta = \frac{dL}{\sqrt{\rho\varepsilon}}\mathrm{e}^{-\chi}$. This parameter setting gives g_{thres} to be defined in the following text. In this theorem, we let $c_{\max} = \min\{\tilde{c}_{\max}, 1/2\}$, and choose any constant $c \leq c_{\max}$.

In this proof, we will actually achieve some point satisfying following condition:

$$\|G(\mathbf{x})\| \leq g_{\text{thres}} \equiv \frac{\sqrt{c}}{\chi^2}\cdot\varepsilon, \qquad \lambda_{\min}(\nabla^2 f(\mathbf{x})) \geq -\sqrt{\rho\varepsilon} \tag{1.19}$$

Since $c \leq 1$, $\chi \geq 1$, we have $\frac{\sqrt{c}}{\chi^2} \leq 1$, which implies any $\mathbf{x}$ satisfy Eq.(1.19) is also a ε-second-order stationary point.

Starting from $\mathbf{x}_0$, we know if $\mathbf{x}_0$ does not satisfy Eq.(1.19), there are only two possibilities:

(1) $\|G(\mathbf{x}_0)\| > g_{\text{thres}}$: In this case, Algorithm 1 will not add perturbation. By lemma 11:

$$\Phi(\mathbf{x}_1) - \Phi(\mathbf{x}_0) \leq -\frac{\eta}{2}\cdot g_{\text{thres}}^2 = -\frac{c^2}{2\chi^4}\cdot\frac{\varepsilon^2}{L}$$

(2) $\|G(\mathbf{x}_0)\| \leq g_{\text{thres}}$: In this case, Algorithm 1 will add a perturbation of radius r, and will perform proximal gradient descent (without perturbations) for the next t_{thres} steps. Algorithm 1 will then check termination condition. If the condition is not met, we must have:

$$\Phi(\mathbf{x}_{t_{\text{thres}}}) - \Phi(\mathbf{x}_0) \leq -\Phi_{\text{thres}} = -\frac{c}{\chi^3} \cdot \sqrt{\frac{\varepsilon^3}{\rho}}$$

This means on average every step decreases the function value by

$$\frac{\Phi(\mathbf{x}_{t_{\text{thres}}}) - \Phi(\mathbf{x}_0)}{t_{\text{thres}}} \leq -\frac{c^3}{\chi^4} \cdot \frac{\varepsilon^2}{L}$$

In case 1, we can repeat this argument for $t = 1$ and in case 2, we can repeat this argument for $t = t_{\text{thres}}$. Hence, we can conclude as long as algorithm 1 has not terminated yet, on average, every step decrease function value by at least $\frac{c^3}{\chi^4} \cdot \frac{\varepsilon^2}{L}$. However, we clearly can not decrease function value by more than $\Phi(\mathbf{x}_0) - \Phi^\star$, where $\Phi^\star$ is the function value of global minima. This means algorithm 1 must terminate within the following number of iterations:

$$\frac{\Phi(\mathbf{x}_0) - \Phi^\star}{\frac{c^3}{\chi^4} \cdot \frac{\varepsilon^2}{L}} = \frac{\chi^4}{c^3} \cdot \frac{L(\Phi(\mathbf{x}_0) - \Phi^\star)}{\varepsilon^2} = O\left(\frac{L(\Phi(\mathbf{x}_0) - \Phi^\star)}{\varepsilon^2} \ln^4\left(\frac{dL\Delta_\Phi}{\varepsilon^2\delta}\right)\right)$$

Finally, we would like to ensure when Algorithm 1 terminates, the point it finds is actually an ε-second-order stationary point. The algorithm can only terminate when the gradient mapping is small, and the function value does not decrease after a perturbation and t_{thres} iterations. We shall show every time when we add perturbation to iterate $\tilde{\mathbf{x}}_t$, if $\lambda_{\min}(\nabla^2 f(\tilde{\mathbf{x}}_t)) < -\sqrt{\rho\varepsilon}$, then we will have $\Phi(\mathbf{x}_{t+t_{\text{thres}}}) - \Phi(\tilde{\mathbf{x}}_t) \leq -\Phi_{\text{thres}}$. Thus, whenever the current point is not an ε-second-order stationary point, the algorithm cannot terminate.

According to Algorithm 1, we immediately know $\|G(\tilde{\mathbf{x}}_t)\| \leq g_{\text{thres}}$ (otherwise we will not add perturbation at time t). By lemma 10 (recall the parameter setting stated before), we know this event happens with probability at least $1 - \frac{dL}{\sqrt{\rho\varepsilon}} e^{-\chi}$ each time. On the other hand, during one entire run of Algorithm 1, the number of times we add perturbations is at most:

$$\frac{1}{t_{\text{thres}}} \cdot \frac{\chi^4}{c^3} \cdot \frac{L(\Phi(\mathbf{x}_0) - \Phi^\star)}{\varepsilon^2} = \frac{\chi^3}{c} \frac{\sqrt{\rho\varepsilon}(\Phi(\mathbf{x}_0) - \Phi^\star)}{\varepsilon^2}$$

By the union bound, for all these perturbations, with high probability lemma 10 is satisfied. As a result Algorithm 1 works correctly. The probability of that is at least

$$1-\frac{dL}{\sqrt{\rho\varepsilon}}e^{-\chi}\cdot\frac{\chi^3}{c}\frac{\sqrt{\rho\varepsilon}(\Phi(\mathbf{x}_0)-\Phi^\star)}{\varepsilon^2}=1-\frac{\chi^3e^{-\chi}}{c}\cdot\frac{dL(\Phi(\mathbf{x}_0)-\Phi^\star)}{\varepsilon^2}$$

Recall our choice of $\chi = 3\max\{\ln(\frac{dL\Delta_f}{c\varepsilon^2\delta}), 4\}$, which gives $e^{-\chi/3} = \min\{\frac{c\varepsilon^2\delta}{dl\Delta_f}, \frac{1}{e^4}\}$. Since $\chi \geq 12$, we have $\chi^3e^{-\chi} \leq e^{-\chi/3}$, this gives:

$$\frac{\chi^3e^{-\chi}}{c}\cdot\frac{dL(\Phi(\mathbf{x}_0)-\Phi^\star)}{\varepsilon^2}\leq e^{-\chi/3}\frac{dL(\Phi(\mathbf{x}_0)-\Phi^\star)}{c\varepsilon^2}\leq\delta$$

which finishes the proof.

□

Remarks on large λ We point out that when λ is large enough so that the g term alters the local landscape of the objective function $\Phi(\mathbf{x})$, it is inevitable that new local minima will be introduced to the landscape of the objective function, and potentially change the stability of saddle points. We hypothesize that perturbed proximal descent will still converge to an ε-second-order stationary point regardless of the magnitude of λ.

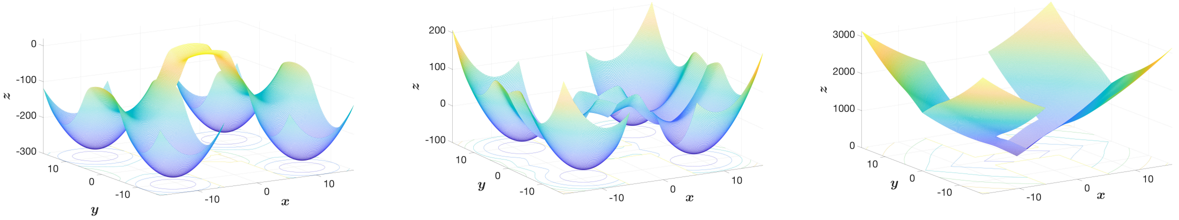
An example for the new local minima introduced by large λ is Fig. 1.3b. We see new wrinkles are introduced to the four legs of the octopus function as λ increases from 1 to 10. If an iteration starts in the neighborhood of creases, it can converge to the bottom of the creases. Fig. 1.3c is an extreme scenario where the original landscape of the octopus function is completely altered to conform to the behavior of ℓ_1 penalty term.

1.4.7 From ε-second-order stationary point to local minimizers

Assumption 3 (Non-degenerate Saddle). *For all stationary points $\mathbf{x}_c$, $\exists\, m > 0$ such that*

$$\min_{i=1,2,\cdots,d}|\lambda_i(\nabla^2 f(\mathbf{x}_c))| > m > 0$$

where λ_i are the eigenvalues (not to be confused with the parameter λ).



(a) $\lambda = 0.01$

(b) $\lambda = 10$

(c) $\lambda = 100$

Figure 1.3: The octopus function with different λ values

With this non-degenerate saddle assumption, the main theorem can be strengthened to the following corollary, whose proof is immediate as one sets the ε value in the main theorem as m^2/ρ and realizes that there is no eigenvalue of $\nabla^2 f$ existing between $-\sqrt{\rho\varepsilon}$ and the first positive eigenvalue.

Corollary 12. *There exists an absolute constant $c_{\max}$ such that if $f(\cdot)$ satisfies assumptions 1, 2 and 3, then for any $\delta > 0, \Delta_\Phi \geq \Phi(\mathbf{x}_0) - \Phi^\star$, constant $c \leq c_{\max}$, and $\varepsilon = \frac{m^2}{\rho}$, with probability $1-\delta$, the output of PPD$(\mathbf{x}_0, L, \rho, \varepsilon, c, \delta, \Delta_f)$ will be a local minimizer of $f + \lambda\|\mathbf{x}\|_1$, and terminate in iterations:*

$$\mathcal{O}\left(\frac{L(\Phi(\mathbf{x}_0) - \Phi^\star)}{\varepsilon^2} \ln^4\left(\frac{dL\Delta_\Phi}{\varepsilon^2\delta}\right)\right)$$

1.5 Numerical experiment

We set f to be the “octopus” function described in [DJL$^+$17] and use perturbed proximal descent to minimize the objective function $\Phi(\mathbf{x}) = f(\mathbf{x}) + \lambda\|\mathbf{x}\|_1$. Plots of octopus function defined in $\mathbb{R}^2$ for various λ are shown in Figure 1.3.

The “octopus” family of functions is parameterized by τ, which controls the width of the “legs,” and M and γ which characterize how sharp each side is surrounding a saddle point, related to the Lipschitz constant. The example illustrated in Fig. 1.3 uses parameters $M = \mathrm{e}, \gamma = 1, \tau = \mathrm{e}$.

We are interested in the octopus family of functions because it can be generalized to any dimension d, and it has $d-1$ saddle points (not counting the origin) which are known to slow down standard gradient descent algorithms. The usual minimization iteration sequence, if starting at the maximum value of the octopus function, will successively go through *each* saddle point before reaching the global minimum, thus rendering the iteration progress easy to track and visualize.

Specifics of Octopus Function

We define octopus function in first quadrant of $\mathbb{R}^d$. And then, by even function reflection, the octopus can be continued to all other quadrants.

Define the *auxiliary gluing functions* as

$$\begin{aligned} \mathcal{G}_1(x_i) &= -\gamma x_i^2 + \frac{-14L+10\gamma}{3\tau}(x_i-\tau)^3 + \frac{5L-3\gamma}{2\tau}(x_i-\tau)^4 \\ \mathcal{G}_2(x_i) &= -\gamma - \frac{10(L+\gamma)}{\tau^3}(x_i-2\tau)^3 - \frac{15(L+\gamma)}{\tau^4}(x_i-2\tau)^4 - \frac{6(L+\gamma)}{\tau^5}(x_i-2\tau)^5 \end{aligned}$$

Define the *gluing function* and *gluing balance constant* respectively as

$$\begin{aligned} \mathcal{G}(x_i, x_{i+1}) = \mathcal{G}_1(x_i) + \mathcal{G}_2(x_i)x_{i+1}^2 \\ \nu = -\mathcal{G}_1(2\tau) + 4L\tau^2 = \frac{26L+2\gamma}{3}\tau^2 + \frac{-5L+3\gamma}{2}\tau^3 \end{aligned}$$

For a given $i = 1, \cdots, d-1$, when $6\tau \geq x_1, \cdots, x_{i-1} \geq 2\tau, \tau \geq x_i \geq 0, \tau \geq x_{i+1}, \cdots, x_d \geq 0$

$$f(\mathbf{x}) = \sum_{j=1}^{i-1} L(x_j - 4\tau)^2 - \gamma x_i^2 + \sum_{j=i+1}^{d} Lx_j^2 - (i-1)\nu \equiv f_{i,1}(\mathbf{x}) \tag{1.20}$$

and if $6\tau \geq x_1, \cdots, x_{i-1} \geq 2\tau, 2\tau \geq x_i \geq \tau, \tau \geq x_{i+1}, \cdots, x_d \geq 0$, we have

$$f(\mathbf{x}) = \sum_{j=1}^{i-1} L(x_j - 4\tau)^2 + \mathcal{G}(x_i, x_{i+1}) + \sum_{j=i+2}^{d} Lx_j^2 - (i-1)\nu \equiv f_{i,2}(\mathbf{x}) \tag{1.21}$$

and for $i = d$, if $6\tau \geq x_1, \cdots, x_{d-1} \geq 2\tau, \tau \geq x_d \geq 0$

$$f(\mathbf{x}) = \sum_{j=1}^{d-1} L(x_j - 4\tau)^2 - \gamma x_d^2 - (d-1)\nu \equiv f_{d,1}(\mathbf{x}) \tag{1.22}$$

and if $6\tau \geq x_1, \cdots, x_{d-1} \geq 2\tau, 2\tau \geq x_d \geq \tau$

$$f(\mathbf{x}) = \sum_{j=1}^{d-1} L(x_j - 4\tau)^2 + \mathcal{G}_1(x_d) - (d-1)\nu \equiv f_{d,2}(\mathbf{x}) \tag{1.23}$$

and if $6\tau \geq x_1, \cdots, x_d \geq 2\tau$,

$$f(\mathbf{x}) = \sum_{j=1}^{d} L(x_j - 4\tau)^2 - d\nu \equiv f_{d+1,1}(\mathbf{x}) \tag{1.24}$$

Remark All saddle points happen at $(\pm 4\tau, \pm 4\tau, \cdots, \pm 4\tau, 0, 0, \cdots, 0)$, and the global minimum is at $(\pm 4\tau, \cdots, \pm 4\tau)$. Regions in the form of $[2\tau, 6\tau] \times \cdots \times [2\tau, 6\tau] \times [\tau, 2\tau] \times [0, \tau] \times \cdots \times [0, \tau]$ are transition zones described by the gluing functions which connect separate pieces to make f a continuous function. The octopus function can be constructed first in the first quadrant, and then using even function reflection to define it in all other quadrants. A typical descent algorithm applied to the octopus generates iterations that take multiple turns like walking down a spiral staircase, each staircase leading to a new dimension.

1.5.1 Results

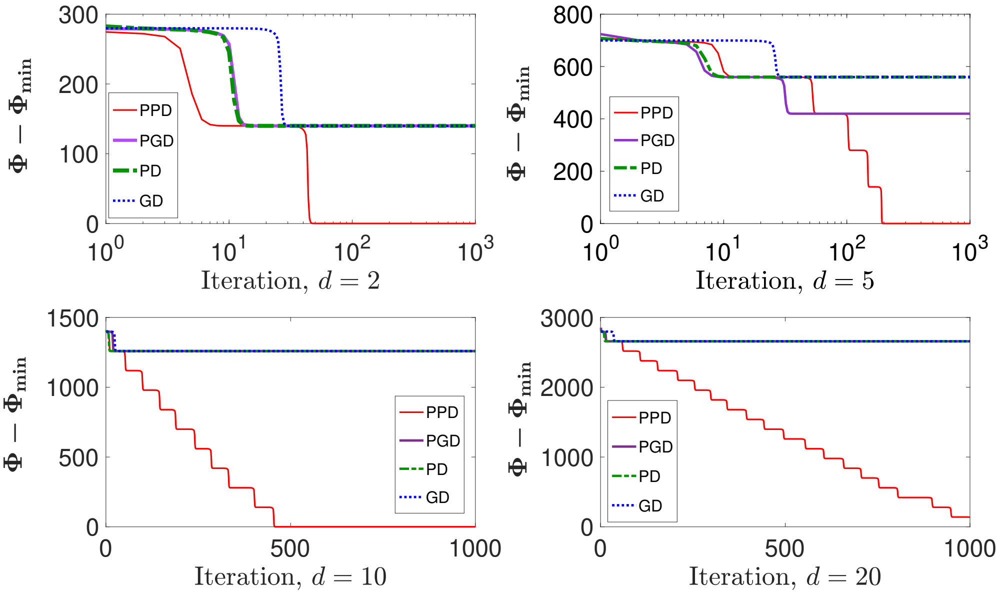


Figure 1.4: Performance of our proposed PPD algorithm on the octopus function with $\lambda = 0.01$

We apply the perturbed proximal descent (PPD) on the octopus function plus $0.01\|\mathbf{x}\|_1$ when the dimension varies between $d = 2, 5, 10, 20$. We set the constant $c = 3$. For comparison, we apply perturbed gradient descent (PGD) as well since $\|\mathbf{x}\|_1$ is differentiable almost everywhere; for both algorithms, the norm of the perturbation $\boldsymbol{\xi}$ is 0.1.

We see that PPD successfully finds the local minimum in the first three cases within 1000 iterations, and in the case of $d = 20$, PPD almost finds the local minimum within 1000 iterations. In contrast, unperturbed proximal descent (PD), gradient descent (GD), and perturbed gradient descent (PGD) sequences are trapped near saddle points.

1.6 Conclusion

This chapter provides an algorithm to minimize a non-convex function plus a ℓ_1 penalty of small magnitude, with a probabilistic guarantee that the returned result is an approximate second-order stationary point, and hence for a large class of functions, a local minimum instead of a saddle point. The complexity is of $\mathcal{O}(\varepsilon^{-2})$ and the result depends on dimension in $\mathcal{O}(\ln^4 d)$.

The deficiency of the result is that the magnitude of ℓ_1 penalty needs to be small to let our theoretical result hold. Meanwhile, we also notice that a large λ will lead to creation of new local minima to the objective function altering the original landscape. Our future work will address the case of large λ in the iteration process.

Chapter 2

Stochastic Gradient Langevin Dynamics with Variance Reduction

2.1 Introduction

In this chapter we consider the optimization algorithm *stochastic gradient descent* (SGD) with variance reduction (VR) and Gaussian noise injected at every iteration step. For historical reasons, the particular randomization format of injecting Gaussian noises bears the name *Langevin dynamics* (LD). Thus, the scheme we consider is referred as *stochastic gradient Langevin dynamics with variance reduction* (SGLD-VR). We point out the ergodicity property of SGLD-VR schemes when used as an optimization algorithm, which the normal SGD method without the additional noise does not have. As the ergodicity property implies the non-trivial probability for the LD process to visit the whole space, the set of global minima may also be traversed during the iteration, thereby revealing the potential of such a scheme for the purposes of global optimization. We provide convergence results of SGLD-VR to local minima in similar style as those in chapter 1.

We apply the SGLD-VR scheme on the empirical risk minimization problem:

$$\text{minimize } f(\boldsymbol{\omega}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{\omega}) \tag{2.1}$$

which is a sampled version of the stochastic optimization problem: $\min\limits_{\boldsymbol{\omega}} F(\boldsymbol{\omega}) = \int f(\boldsymbol{\omega};\mathbf{x})\,\mathrm{d}\mathbf{x}$, where $\mathbf{x}$ is the collection of training data and $\boldsymbol{\omega}$ is the parameter for the model. In the following text we use $\mathbf{x}$ instead of $\boldsymbol{\omega}$ as the input for the objective f. For empirical risk minimization, one may construct the gradient estimator by subsampling terms in the summation of the objective (2.1) for gradient evaluation in order to accelerate the optimization process, which coincides with the SGD framework. Variance reduction is leveraged to accelerate convergence, and LD is further engaged in the SGD scheme to enable ergodicity property and convergence property to local minima.

2.1.1 Prior art

The Langevin dynamical equation describes the trajectory $X(t)$ of the following stochastic differential equation

$$\mathrm{d}X_t = -\nabla U(X_t)\,\mathrm{d}t + \sigma\,\mathrm{d}B_t, \tag{2.2}$$

which is a characterization for the continuous motion of a particle in a potential field U. Using this dynamic as a master equation, through Kramers–Moyal expansion one can derive the Fokker-Planck equation, which gives the spatial distribution of particles at a given time, thus a full characterization of the statistical properties of a particle ensemble [Kra40].

LD and sampling The connection between Langevin dynamics (LD) and the distribution of particle ensemble reveals the potential of applying LD on sampling. Suppose that the distribution of interest is $\pi(\mathbf{x})$ and that there exists a function U such that $\pi(\mathbf{x}) = \frac{\exp(-U(\mathbf{x}))}{\int \exp(-U(\mathbf{x}))\,\mathrm{d}\mathbf{x}}$, then LD equation which contains a Brownian motion term gives a Monte-Carlo-natured simulation based on the distribution $\pi(\mathbf{x})$, and the stationary distribution such a dynamic converges to is $\pi(\mathbf{x})$. To numerically implement LD equation for sampling purposes, one needs to discretize the continuous LD equation. A simple version of the dicretization is the *unadjusted Langevin algorithm* (ULA),

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k \nabla U(\mathbf{x}_k) + \delta_0 \sqrt{\eta_k}\boldsymbol{\epsilon}_k \tag{2.3}$$

where $\boldsymbol{\epsilon}_k \in \mathcal{N}(0, \mathbf{I}_d)$, $\mathbf{x} \in \mathbb{R}^d$. The Gaussian noise term enables the scheme to explore the sample space and the drift term guides the direction of exploration. One common modified scheme is *Metropolis adjusted Langevin algorithm* (MALA), where upon the suggested update by ULA, there is an additional accept/reject step, with the probability of accepting the the update as $1 \wedge \frac{\pi(\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k+1})}{\pi(\mathbf{x}_{k+1})p(\mathbf{x}_{k+1}|\mathbf{x}_k)}$.

Naturally two central questions related to this sample scheme arise: whether or not the distribution of samples generated by LD converges, and if so, to π; and what is the mixing time of

LD (i.e., how long does it takes for the LD to approximately reach equilibrium hence generating valid samples from the distribution π). The first question motivates the importance of MALA: in terms of *total variation* (TV) norm, while ULA can fail to converge for either light-tailed or heavy-tailed target distribution, MALA is guaranteed to converge to any continuous target distribution [MT09]. Regarding the second question about convergence speed, researchers have investigated the sufficient conditions for ULA and MALA respectively to guarantee exponential (geometric) convergence to target distribution. [MT96] shows for distributions over $\mathbb{R}$, the necessary and sufficient condition for MALA to converge to target distribution $\pi(\mathbf{x})$ at geometric speed is that $\pi(\mathbf{x})$ has exponential tails. The sufficiency of this condition is generated to higher dimension in [RT96a]. The seminal work [RT96b] shows that w.r.t. target distributions that are in essence non-localized, or heavy-tailed, MALA cannot converge at geometric speed.

In parallel there have been works to show the convergence of LD for distribution approximation in terms of Wasserstein-2 distance [DK17] and KL-divergence respectively [CB18].

A particular case of interest for the application of LD on sampling is to find the posterior distribution of parameters $\boldsymbol{\theta}$ in the Bayesian setting, where the updates are set as

$$\Delta\boldsymbol{\theta}_k = \eta_k \left(\nabla p(\boldsymbol{\theta}_k) + \sum_{i=1}^{N} \nabla \log p(\mathbf{x}_i|\boldsymbol{\theta}_k) \right) + \sqrt{\eta_k}\boldsymbol{\epsilon}_k \tag{2.4}$$

where $\boldsymbol{\epsilon}_k \sim \mathcal{N}(0, \mathbf{I})$. To maximize the likelihood, [WT11] suggests to use the format of stochastic gradient descent in the derivative term of (2.4). [BM99] shows that this minibatch-styled LD will converge to the correct distribution in terms of KL divergence.

LD and optimization The main focus of this chapter is on optimization. LD offers an exciting opportunity for global optimization due to the exploring nature of the Brownian motion term. Simulating multiple particles to obtain information about the geometric landscape of the objective function—thus locating a global minimum—is often too computationally expensive to be practical. Notice that when one considers the convergence to a distribution, the exploring nature of LD due to continually injected Gaussian noise of constant variance is the key factor, while for the purpose of optimization, one usually exploits noises with diminishing variance since the goal is to converge to a point.

The technique to achieve this point convergence is *annealing*, which means decreasing the variance of the noise as t grows. Formally, let the objective function be U and we construct the probability distribution $p_T(\mathbf{x}) = \frac{1}{Z}\exp\left(-\frac{U(\mathbf{x})}{T}\right)$, where Z is the normalization factor. The key observation is that as the parameter $T \to 0$, the distribution $p_T(\mathbf{x})$ will concentrate on the global minima. This parameter T is usually referred to as *temperature*, alluding to the alloy annealing process where as temperature decreases, the structure of the metal evolves into the most stable one, hence reaching the state with minimum potential energy. To formulate LD for optimization, one essentially takes the usual LD equation but with the variance term σ now a function of time, $\sigma = \sqrt{T}(t)$:

$$\mathrm{d}\mathbf{x}_t = \nabla U(\mathbf{x}_t)\,\mathrm{d}t + \sqrt{T(t)}\,\mathrm{d}B_t$$

The pioneering work by Chiang et al. [CHS87] shows that with the annealing schedule $T(t) \propto (\log t)^{-1}$, then LD will find the global minimum. The work by Chiang et al. does not specify how to simulate the continuous version of Langevin dynamics, thus not providing information on the convergence of discrete approximation (Euler-Maruyama method as an example) for LD. [GM91] fills this gap by proving that with an annealing schedule $\eta_k \propto k^{-1}$ and $T_k \propto (k \log\log k)^{-1}$, then discretized LD will converge to the global minima in probability. More recently, [RRT17] uses optimal transport formalism to study the empirical risk minimization problem. Their proof uses the Wasserstein-2 distance to evaluate distribution discrepancy and consists of two parts: first show that the discretization error of LD from continuous LD accumulates linearly with respect to the error tolerance level, and then show that the continuous LD will converge to the true target distribution exponentially fast.

Variance reduction (VR) and LD In this chapter we aim to apply variance reduction techniques in the setting of LD to acceleration the optimization process and to derive improved time complexity dependence on error tolerance level.

The specific VR technique we use was originally proposed to reduce the variance of the minibatch gradient estimator in stochastic gradient descent for convex objectives by [JZ13] and separately by [DBLJ14]. [RHS+16, AZH16] have respectively generalized the application of VR techniques to nonconvex objectives and provided convergence guarantee to first-order stationary points.

In essence, we use the *control variate* technique to construct a new gradient estimator by adding an additional term to the minibatch gradient estimator in SGD (∇_{SGD}) and this term is correlated with ∇_{SGD}, thus reducing the variance of the gradient estimator as a whole. More specifically, consider the classic gradient estimator of function f in (2.1) at point $\mathbf{x}$: $\nabla_{\text{SGD}} = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \nabla f_i(\mathbf{x})$, where $\mathcal{I} \subseteq [n]$. If there is another r.v. Y whose expectation is known, then we can construct a new unbiased gradient estimator $\widetilde{\nabla}$ of function f at point $\mathbf{x}$ as $\widetilde{\nabla} = \nabla_{\text{SGD}} + \alpha(Y - \mathbb{E}\,Y)$, where α is a constant. If $\alpha = -\frac{\text{Cov}(\nabla_{\text{SGD}},Y)}{\text{Var}[Y]}$, then the variance of new gradient estimator $\widetilde{\nabla}$ is $\text{Var}[\widetilde{\nabla}] = (1 - \rho^2_{\nabla_{\text{SGD}},Y})\text{Var}[\nabla_{\text{SGD}}] \le \text{Var}[\nabla_{\text{SGD}}]$, where ρ is the Pearson correlation coefficient between ∇_{SGD} and Y. Usually one may not be lucky enough to have access to such a r.v. Y whose covariance with ∇_{SGD} is known, therefore the constant α for the control variate needs to be chosen in an empirical manner.

Motivated by applications in physics and neural network, there has been more interest in guarantees for convergence to second-order stationary points by optimization algorithms. A continual line of work has relied on randomized first-order methods, in particular gradient descent [JGN+17] and proximal descent [HB20a] to achieve this purpose. See section 1.1.1 in chapter 1 for a detailed review for this line of work.

The main algorithm we consider in this chapter is stochastic gradient Langevin dynamics (SGLD) with variance reduction, which consists of two sources of randomness: one from stochastic gradients, the other from Gaussian noise injected at each step. Previous work have investigated the SGLD for optimization to find local minimizers [CDT19, ZLC17], and reported results for convergence to approximate second order stationary points. In particular, [XCZG18] show that with constant-variance Gaussian noise injected at each step, SGLD-VR finds an approximate minimizer with time complexity $\mathcal{O}(\frac{d^5}{\varepsilon^4})$. We aim to improve the dependency on ε. We also point out that when variance of Gaussian noise is set as constant in SGLD, the function value or the point distance between optimal point and iterate can never go to zero, but can be bounded by a constant depending on the size of variance.

Finally we want to mention that [DRP$^+$16] introduces the VR technique to Bayesian inference setting where the objective is the posterior and the gradient estimator is constructed with minibatch of training data. However, the given convergence guarantee is in terms of mean squared error of statistics evaluated based on posterior, instead of posterior distribution of the parameter.

Notation Bold symbols indicate vectors, for example a vector $\mathbf{x} \in \mathbb{R}^d$, where d stands for the dimension of Euclidean space. We use o to indicate the starting index of a minibatch. We use $\eta_{a:b}$ as the shorthand for $\sum_{i=a}^{b} \eta_i$.

2.2 Algorithm and main results

The main algorithm is given as algorithm 2.

Algorithm 2 Variance reduced stochastic gradient Langevin dynamics (VRSGLD)

```
Require: initial stepsize $\eta_0 > 0$, stepsize decay order $\nu \geq 1$, initial Gaussian noise magnitude
    parameter $\delta_0$, batch size $B_b$, epoch length $B_\mathrm{e}$
Initialize $\mathbf{x}_0 = 0$, $\widetilde{\mathbf{x}}^{(0)} = \mathbf{x}_0$
for $s = 0, 1, 2, \cdots, \frac{T}{B_\mathrm{e}} - 1$ do
    $\widetilde{\mathbf{w}} = \nabla f(\widetilde{\mathbf{x}}^{(s)})$
    for $l = 0, 1, \cdots, B_\mathrm{e} - 1$ do
        set index $t = sB_\mathrm{e} + l$
        randomly pick a subset $I_t$ from $[n]$ of size $|I_t| = B_b$; randomly draw $\boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{I})$
        construct gradient estimator $\widetilde{\nabla}_t = \frac{1}{B_b} \sum_{i_t \in I_t} \left( \nabla f_{i_t}(\mathbf{x}_t) - \nabla f_{i_t}(\widetilde{\mathbf{x}}^{(s)}) + \widetilde{\mathbf{w}} \right)$
        update $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \widetilde{\nabla}_t + \rho_t \boldsymbol{\epsilon}_t$
    end for
    $\widetilde{\mathbf{x}}^{(s)} = \mathbf{x}_{(s+1)B_\mathrm{e}}$
end for
```

In algorithm 2, the stepsize η_t and ρ_t is set as the following, for some $\nu \geq 1$:

$$\eta_t = \frac{\eta_0}{t^\nu} \text{ and } \rho_t = \frac{\rho_0}{t^{\nu/2}}. \tag{2.5}$$

Ergodicity In this work we show that the discretized variance reduced LD (VRSGLD) has an ergodic property which gives the iteration process the potential of exploring wider space, thus with positive possibility of traversing through the global optimal point.

The argument for ergodicity has two parts: In Lemma 13 we first give an explicit upper bound of the expected time of visiting a given level set for the j-th time ($j \geq 1$); then in Theorem 14 we show that there is a positive possibility such that the LD iteration can visit any fixed point within a level set.

Assumption 4 (Lipschitz Gradient)**.** *f is continuously differentiable, and there exists a positive constant L such that for all $\mathbf{x}$ and $\mathbf{y}$, $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$.*

Assumption 5 (Regularization conditions for the objective function f)**.** *There exist non-negative constants μ_1, μ_2 and ψ_1, ψ_2 such that for all $\mathbf{x} \in \mathbb{R}^d$,*

$$\|\nabla f(\mathbf{x})\|^2 \geq \mu_1 f(\mathbf{x}) - \psi_1 \tag{2.6}$$

$$\|\mathbf{x}\|^2 \leq \mu_2 f(\mathbf{x}) + \psi_2 \tag{2.7}$$

Remark We make the same regularization assumptions as in [CDT19]. A relevant regularization condition commonly used in previous literature is the (m, b)-*dissipative condition* [MSH02, RRT17, XCZG18, ZLC17], which reads that there exist positive constants m and b such that for all $\mathbf{x} \in \mathbb{R}^d$, $\langle \nabla f(\mathbf{x}), \mathbf{x} \rangle \geq m\|\mathbf{x}\|^2 - b$. [DT20] shows that the dissipative condition implies (2.6), which renders the assumption (2.6) weaker. Equation (2.7) implies that f is supercoercive [BC17], and in particular coercive, and thus has bounded level sets.

Lemma 13 (Recurrence)**.** *Let n_0 be the index such that $\eta_{n_0} \leq \delta$, and n_k be the sequence of iteration index $n_{k+1} = \min_s\{s : s > n_k,\, \eta_{n_k:s} \geq \delta\}$.*

Under the regularization conditions in Assumption 5 and under Assumption 4, there exists a constant C_1 such that $\alpha = 1 - 2\exp(-(1 - C_1)\mu\delta)$, $B = 2(\frac{2\eta_0 L^3}{B_{\mathrm{e}}} f(\mathbf{x}_0) + \frac{\delta_0^2 L d}{2\eta_0})$ and $K = \frac{\ln \frac{f(\mathbf{x}_{n_0})}{\delta B}}{(1 - C_1)\mu\delta}$, with the stopping time sequence $\{\tau_k\}$ defined as $\tau_0 = K$ and $\tau_{k+1} = \min\{t : t \geq \tau_k + 1, f(\mathbf{x}_{n_t}) \leq 2\delta B\}$,

$$\mathbb{E}\left[\tau_j\right] \leq \frac{4}{\alpha} + K + j\left(\frac{1}{2\alpha\delta} + 1\right) \tag{2.8}$$

Theorem 14 (Ergodicity)**.** *Under the regularization conditions in Assumption 5 and under Assumption 4, with the same parameter setting as in Lemma 13, for any accuracy $\widetilde{\varepsilon} > 0$, failure probability $p > 0$, and any point $\mathbf{s} \in \mathbb{R}^d$, there is a number*

$$T = \mathcal{O}\left(\frac{1}{p\mu_1\big(4\eta_0 L^3 \frac{\mu_2 f(\mathbf{x}_0)+2\psi_2}{B_\mathrm{e}} + \frac{\delta_0^2}{\eta_0} Ld\big)}\left(1 + \ln f(\mathbf{x}_0) + \frac{(d\|\mathbf{s}\| + \widetilde{\varepsilon})^d}{\big((\frac{4}{\sqrt{2\pi}} - 1)\mathrm{e}^{-1/2}\widetilde{\varepsilon}\big)^d}\right)\right) \tag{2.9}$$

such that

$$\mathbf{Pr}(\|\mathbf{x}_t - \mathbf{s}\| \leq \widetilde{\varepsilon} \textit{ for some } t < T) \geq 1 - p \tag{2.10}$$

Convergence to a first-order stationary point We further compute the time complexity for the LD to converge to a $\varepsilon-$first order stationary point. We define $\mathbf{x}^\star$ to be a ε-first order stationary point if $\|\nabla f(\mathbf{x}^\star)\| \leq \varepsilon$.

Theorem 15. *Under Assumption 4, for any $p \in (0, 1)$, then with probability at least $1 - p$, the time complexity for the LD described in algorithm 2 to converge to an ε-first order stationary point $\mathbf{x}^\star$ is $\mathcal{O}\left(\frac{\Delta_f d}{\varepsilon^2 p}\right)$, where $\Delta_f = f(\mathbf{x}_0) - f(\mathbf{x}^\star)$.*

Convergence to an ε-second-order stationary point An ε-second-order stationary point is a more restrictive type of ε-first order stationary point, and is more likely to be an actual local minimizer.

Definition 16. *Consider a smooth function $f(\mathbf{x})$ with continuous second order derivative. A point $\mathbf{x}$ is an ε-second-order stationary point if*

$$\|\nabla f(\mathbf{x})\| \leq \varepsilon \quad \textit{and} \quad \lambda\big(\nabla^2 f(\mathbf{x})\big)_{\min} \geq -\varepsilon^2 \tag{2.11}$$

where $\lambda(\cdot)_{\min}$ is the smallest eigenvalue.

We make the strict saddle assumption which is common in nonconvex optimization literature [GHJY15, LSJR16, JGN$^+$17, MOJ18, LPP$^+$19, VGFP19, SLQ$^+$19, LY19, Li19, HB20a],: i.e.,

Assumption 6 (strict saddle)**.** *There exists a constant $q > 0$ such that for all first-order stationary points $\mathbf{x}_\mathrm{fsp}$, we have*

$$|\lambda(\nabla^2 f(\mathbf{x}_\mathrm{fsp}))| \geq q > 0.$$

Assumption 7 (Hessian Lipschitz)**.** *f is twice continuously differentiable, and there exists a positive constant L_2 such that for all $\mathbf{x}$ and $\mathbf{y}$, $\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})\| \leq L_2\|\mathbf{x}-\mathbf{y}\|$.*

Theorem 17. *Under Assumptions 4, 6 and 7, setting the stepsize decay parameter $\nu \in [1,2]$ and $\rho_0 = \mathcal{O}(\varepsilon)$, with probability $\mathcal{O}\left(\frac{\varepsilon^{d-1}}{\Gamma(\frac{d-2}{2})L^{d-1}q^{d-1}}\right) \cdot \exp(-\mathcal{O}(\varepsilon d))$, the time complexity for the LD described in algorithm 2 to converge to an ε-second order stationary point $\mathbf{x}^\star$ is $\mathcal{O}\left(\frac{\Delta_f}{\varepsilon^2}\right) + \exp\left(\mathcal{O}(\varepsilon d)\right)$, where $\Delta_f = f(\mathbf{x}_0) - f(\mathbf{x}^\star)$.*

method	w. VR	noise magnitude setting	convergence target	Time complexity m
[RRT17]	no	constant	global min.	$\widetilde{\mathcal{O}}\left(\dfrac{d+1/\delta_0}{\delta_0\varepsilon^4}\right)$
[XCZG18]	yes	constant	global min.	$\widetilde{\mathcal{O}}\left(\dfrac{1}{\varepsilon^{5/2}}\right)\exp(\widetilde{\mathcal{O}}(d))$
[ZLC17]	no	constant	local min.	$\mathcal{O}\left(\dfrac{\Delta_f d^4 L^2}{\varepsilon^4}\right)$
This work	yes	diminishing w. poly. speed	local min.*	$\mathcal{O}\left(\dfrac{\Delta_f}{\varepsilon^2}\right) + \exp\left(\mathcal{O}(\varepsilon d)\right)$

Table 2.1: Comparison between convergence results for variants of LD optimization schemes. * indicates convergence target is actually a ε-second-order stationary point, which coincides with a local minimizer when $\varepsilon < \sqrt{q}$ under assumption 6.

method	Bounded	Grad. Lip.	Hess. Lip.	Regularization	others
[RRT17]	f and $\|\nabla f\|$	yes	no	(m,b)-dissipative	(1) stoch. grad. sub-exp. tails (2) init. pt. sub-Gauss. tails
[XCZG18]	none	yes	no	(m,b)-dissipative	none
[ZLC17]	$\|\nabla f\|$ and $\|\nabla^2 f\|$	yes	yes	$(1,0)$-dissipative	grad. sub-exp. tails
This work	none	yes	yes	Assumption 5	strict saddle

Table 2.2: Comparison between assumptions made for variants of LD optimization schemes. The Hessian Lipschitz assumption is used for second-order convergence property if made.

2.3 First-order stationary point convergence property

In this section we show proof of the result for first-order convergence property (theorem 15) and the needed lemmas as preparation. We first bound the expectation of the square of the gradient norm in a minibatch of using SGLD-VR for minimization. Requiring the gradient norm to be less than pre-designated threshold leads to the first-order stationary point (fsp). To estimate the time needed to converge to a fsp, we exploit the dependence of the gradient norm bound on iteration count t. The quantity that plays a central role in the argument is the Lyapunov function, which is essential in constructing the upper bound for gradient norm and connects the argument between successive minibatches.

Lemma 18 (Bound of variance of SVRG gradient estimator [RHS$^+$16])**.** *In an epoch, the SVRG gradient estimator satisfies*

$$\mathbb{E}\,[\|\widetilde{\nabla}_t\|^2] \leq 2\mathbb{E}\,[\|\nabla f(\mathbf{x}_t)\|^2] + 2\frac{L^2}{B_{\mathrm{e}}}\mathbb{E}\,[\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2]. \tag{2.12}$$

The proof of the above lemma can be found in [RHS$^+$16].

Adapting the framework in [RHS$^+$16] for the LD setting, the following lemma bounds the expectation of the gradient norm for the SGLD-VR iteration sequence in a minibatch:

Lemma 19. *Define the weight sequence $\{c_t\}$ recursively as $c_t = c_{t+1}(1 + \beta_t\eta_t + 2\frac{\eta_t^2 L^2}{B_{\mathrm{e}}}) + \frac{\eta_t^2 L^3}{B_e}$ with $c_{B_{\mathrm{e}}} = 0$, and then define the Lyapunov function $R_t = \mathbb{E}\,[f(\mathbf{x}_t) + c_t\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2]$ for each epoch. Define the normalization sequence $\gamma_t = \eta_t - \frac{c_{t+1}}{\beta_t}\eta_t - \eta_t^2 L - 2c_{t+1}\eta_t^2$ with η_t and $\beta_t > 0$ set to ensure $\gamma_t > 0$. Under Assumption 4, inside an epoch,*

$$\mathbb{E}\,[\|\nabla f(\mathbf{x}_t)\|^2] \leq \frac{R_t - R_{t+1}}{\gamma_t} + \left(\frac{L}{2} + c_{t+1}\right)\frac{d\rho_t^2}{\gamma_t}.$$

Proof. We find upper bounds to the Lyapunov functions R_t in terms of the negative norm of the SVRG gradient estimator, thus proving the lemma. We bound the two terms in the Lyapunov functions respectively. For notational simplicity let $\nabla f(\mathbf{x}_t) = \nabla_t = \mathbb{E}\,_{I_t}[\widetilde{\nabla}_t]$.

For the first term $f(\mathbf{x}_{t+1})$ in the Lyapunov function,

$$\begin{aligned}\mathbb{E}\left[f(\mathbf{x}_{t+1})\right] &\leq \mathbb{E}\left[f(\mathbf{x}_t) + \langle \nabla f(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}_t\rangle + \frac{L}{2}\|\mathbf{x}_{t+1} - \mathbf{x}_t\|_2^2\right] \\ &= \mathbb{E}\left[f(\mathbf{x}_t) - \eta_t\|\nabla_t\|^2 + \frac{L}{2}\left(\eta_t^2\|\widetilde{\nabla}_t\|^2 + \rho_t^2\|\boldsymbol{\epsilon}_t\|^2\right)\right]\end{aligned}$$

where the first inequality uses the L-smooth of function f and the second equality uses the SVRG update in algorithm 2 ($\mathbf{x}_{t+1} - \mathbf{x}_t = -\eta_t\widetilde{\nabla}_t + \rho_t\boldsymbol{\epsilon}_t$) and the unbiasedness of the gradient estimator $\widetilde{\nabla}_t$.

For the second term $\|\mathbf{x}_{t+1} - \widetilde{\mathbf{x}}\|$, as $\langle \nabla_t, \widetilde{\mathbf{x}} - \mathbf{x}_t\rangle \overset{\texttt{CS}}{\leq} \|\nabla_t\|\|\mathbf{x}_t - \widetilde{\mathbf{x}}\| \overset{\texttt{Young}}{\leq} \frac{1}{2\beta_t}\|\nabla_t\|^2 + \frac{\beta_t}{2}\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2$,

$$\begin{aligned}\mathbb{E}\left[\|\mathbf{x}_{t+1} - \widetilde{\mathbf{x}}\|^2\right] = \mathbb{E}\left[\|\mathbf{x}_{t+1} - \mathbf{x}_t + \mathbf{x}_t - \widetilde{\mathbf{x}}\|^2\right] &= \mathbb{E}\left[\|\mathbf{x}_{t+1} - \mathbf{x}_t\|^2 + \|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2 + 2\langle \mathbf{x}_{t+1} - \mathbf{x}_t, \mathbf{x}_t - \widetilde{\mathbf{x}}\rangle\right] \\ &= \mathbb{E}\left[\eta_t^2\|\widetilde{\nabla}_t\|^2 + \rho_t^2\|\boldsymbol{\epsilon}_t\|^2 + \|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2 + 2\eta_t\langle \nabla_t, \widetilde{\mathbf{x}} - \mathbf{x}_t\rangle\right] \\ &\leq \mathbb{E}\left[\eta_t^2\|\widetilde{\nabla}_t\|^2 + \rho_t^2\|\boldsymbol{\epsilon}_t\|^2 + (1 + \eta_t\beta_t)\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2 + \frac{\eta_t}{\beta_t}\|\nabla_t\|^2\right]\end{aligned}$$

Putting these two terms together into R_{t+1}, we have

$$\begin{aligned}R_{t+1} &\leq \mathbb{E}\Bigg[f(\mathbf{x}_t) + \left(\frac{\eta_t c_{t+1}}{\beta_t} - \eta_t\right)\|\nabla_t\|^2 + \left(\frac{L}{2} + c_{t+1}\right)(\eta_t^2\|\widetilde{\nabla}_t\|^2 + \rho_t^2\|\boldsymbol{\epsilon}_t\|^2) \\ &\qquad + (1 + \eta_t\beta_t)c_{t+1}\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2\Bigg] \\ &\overset{(2.12)}{\leq} \mathbb{E}\Bigg[f(\mathbf{x}_t) + \left(\frac{\eta_t c_{t+1}}{\beta_t} - \eta_t + (L + 2c_{t+1})\eta_t^2\right)\|\nabla_t\|^2 + \left(\frac{L}{2} + c_{t+1}\right)\rho_t^2\|\boldsymbol{\epsilon}_t\|^2 \\ &\qquad + \left((1 + \eta_t\beta_t)c_{t+1} + (L + 2c_{t+1})\frac{\eta_t^2 L^2}{B_{\mathrm{e}}}\right)\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2\Bigg] \\ &= \mathbb{E}\left[f(\mathbf{x}_t) - \gamma_t\|\nabla_t\|^2 + \left(\frac{L}{2} + c_{t+1}\right)\rho_t^2\|\boldsymbol{\epsilon}_t\|^2 + c_t\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2\right] \\ &= R_t - \mathbb{E}\left[\gamma_t\|\nabla_t\|^2\right] + \mathbb{E}\left[\left(\frac{L}{2} + c_{t+1}\right)\rho_t^2\|\boldsymbol{\epsilon}_t\|^2\right] \\ &= R_t - \mathbb{E}\left[\gamma_t\|\nabla_t\|^2\right] + \left(\frac{L}{2} + c_{t+1}\right)\rho_t^2 d.\end{aligned}$$

We set (β_t) and η_0 properly (see the remark below this proof) such that $-\gamma_t = \frac{\eta_t c_{t+1}}{\beta_t} - \eta_t + (L + 2c_{t+1})\eta_t^2 \leq 0$ for all $t = 0, 1, \cdots, B_{\mathrm{e}} - 1$. This can always be achieved as c_t is a decreasing sequence and c_t is negatively related to β_t. Then

$$\mathbb{E}\left[\gamma_t\|\nabla_t\|^2\right] \leq -R_{t+1} + R_t + \left(\frac{L}{2} + c_{t+1}\right)\rho_t^2 d. \tag{2.13}$$

□

Remark We show that the η_0 and $\{\beta_t\}$ sequence setting in the end of the proof of lemma 19 always exists. For now we can assume $\beta_t = \widetilde{\beta}$ is a constant. Then we can define an upper bound sequence for $\{c_t\}$ as

$$\widetilde{c}_t = \widetilde{c}_{t+1}(1 + \widetilde{\beta}\eta_0 + 2\frac{\eta_0^2 L^2}{B_\mathrm{e}}) + \frac{\eta_0^2 L^3}{B_\mathrm{e}}$$

with $\widetilde{c}_{B_\mathrm{e}} = 0$. Then, $c_t \leq \widetilde{c}_t$ for $1 \leq t \leq B_\mathrm{e}$. Consequently, for expression simplicity assuming $q = 1 + \widetilde{\beta}\eta_0 + 2\frac{\eta_0^2 L^2}{B_\mathrm{e}}$ and $D = \frac{\frac{\eta_0^2 L^3}{B_\mathrm{e}}}{\widetilde{\beta}\eta_0 + \frac{2\eta_0^2 L^2}{B_\mathrm{e}}} = \frac{\frac{\eta_0 L^3}{B_\mathrm{e}}}{\widetilde{\beta} + \frac{2\eta_0 L^2}{B_\mathrm{e}}}$, we have

$$\frac{\widetilde{c}_t + D}{\widetilde{c}_{t+1} + D} = q.$$

It follows that $\frac{1}{q^{B_\mathrm{e}}}(\widetilde{c}_0 + D) = \widetilde{c}_{B_\mathrm{e}} + D = D$, and $\widetilde{c}_0 = (q^{B_\mathrm{e}} - 1)D$. We need to set $\widetilde{\beta}$ in a way such that $\gamma_t > 0$ for all $1 \leq t \leq B_\mathrm{e}$. As

$$\gamma_t \geq \left(1 - \frac{\widetilde{c}_0}{\widetilde{\beta}} - \eta_0 L - 2\widetilde{c}_0\eta_0\right)\eta_t \overset{\texttt{need}}{>} 0,$$

a sufficient condition to assure the second inequality above is

$$\widetilde{c}_0\left(\frac{1}{\widetilde{\beta}} + 2\eta_0\right) + \eta_0 L < 1 \tag{2.14}$$

Let $\widetilde{\beta}\eta_0$ be small while $\widetilde{\beta} > 1$, then the l.h.s. of (2.14) is of the order $B_\mathrm{e}\eta_0\frac{\frac{\widetilde{\beta}\eta_0 L^3}{B_\mathrm{e}}}{\widetilde{\beta}^2 + 2\frac{\widetilde{\beta}\eta_0 L^2}{B_\mathrm{e}}}$, which can ensure (2.14) to hold.

Now we use the bound of gradient norm within a minibatch to build that for the whole iteration in the following lemma:

Lemma 20. *Let $\bar{\gamma} = \min_{0 \leq t \leq T-1} \gamma_t$ where γ_t is defined in the previous lemma, and $\nu > 0$. Then under Assumption 4,*

$$\mathbb{E}\left[\|\nabla f(\mathbf{x}_a)\|^2\right] \leq \frac{f(\mathbf{x}_0) - f(\mathbf{x}^\star)}{T\bar{\gamma}} + \frac{d}{\bar{\gamma}}\left(\frac{L}{2} + c_0\right)\frac{C_0}{T^\nu}, \tag{2.15}$$

where $\mathbf{x}_a$ is randomly chosen from the entire iterate sequence and C_0 is a universal constant.

Proof. We set $c_{B_{\mathrm{e}}} = 0$ so that $R_0^{(\alpha)} = f(\mathbf{x}_0^{(\alpha)})$ and $R_{B_{\mathrm{e}}}^{(\alpha)} = f(\mathbf{x}_{B_{\mathrm{e}}}^{(\alpha)})$ for the fixed epoch α. Per line 10 in Algorithm 2, the ending point of the previous epoch is the starting point of the next epoch, i.e., $\mathbf{x}_0^{(\alpha)} = \mathbf{x}_{B_{\mathrm{e}}}^{(\alpha-1)}$. Summing up all the iteration steps in each epoch, we have

$$\sum_{\alpha=0}^{\frac{T}{B_{\mathrm{e}}}-1} \sum_{l=0}^{B_{\mathrm{e}}-1} \mathbb{E}\left[\nabla f(x_l^{(\alpha)})\right] \leq \frac{f(\mathbf{x}_0) - f(\mathbf{x}_T)}{\bar{\gamma}} + \frac{\mathbb{E}\left[\|\boldsymbol{\epsilon}\|^2\right]}{\bar{\gamma}} \sum_{t=0}^{T-1} \left(\frac{L}{2} + c_{(t \bmod B_{\mathrm{e}})+1}\right) \rho_t^2$$

When ρ_t is set as $\mathcal{O}(\frac{1}{t^{\nu/2}})$ where $\nu \geq 1$, as c_t is bounded w.r.t. a fixed epoch, $\sum_{t=0}^{T-1} \rho_t^2 = \mathcal{O}(T^{1-\nu})$. (The $\nu = 1$ case leads to logarithmic growth of summation of ρ_t^2, which does not affect the following result.) Then consider the LHS of the inequality as the average over all iterates, then

$$\begin{aligned}
\mathbb{E}\left[\|\nabla f(\mathbf{x}_a)\|^2\right] &\leq \frac{f(\mathbf{x}_0) - f(\mathbf{x}^\star)}{T\bar{\gamma}} + \frac{\mathbb{E}\left[\|\boldsymbol{\epsilon}\|^2\right]}{T\bar{\gamma}} \sum_{t=0}^{T-1} \left(\frac{L}{2} + c_{(t \bmod B_{\mathrm{e}})+1}\right) \rho_t^2 \\
&\leq \frac{f(\mathbf{x}_0) - f(\mathbf{x}^\star)}{T\bar{\gamma}} + \frac{d}{T\bar{\gamma}} \left(\frac{L}{2} + c_0\right) \sum_{t=0}^{T-1} (\frac{L}{2} + c_0) \rho_t^2 \\
&= \frac{f(\mathbf{x}_0) - f(\mathbf{x}^\star)}{T\bar{\gamma}} + \frac{d}{\bar{\gamma}} \left(\frac{L}{2} + c_0\right) \frac{C_0}{T^\nu}. \qquad (2.16)
\end{aligned}$$

$\square$

Proof of Thm. 15. Per (2.16), we see that the time complexity for the LD to converge to an ε-first order stationary point is $\mathcal{O}\left(\frac{\Delta_f d}{\bar{\gamma}\varepsilon^2}\right)$. Another way to phrase the time complexity is through the hitting time of LD to a first-order stationary point (fsp) τ_{fsp}. To estimate the expected time for the iteration sequence to enter a fsp neighborhood,

$$\begin{aligned}
\mathbf{Pr}(\tau_{\text{fsp}} > T) = \mathbf{Pr}(\|\nabla f(\mathbf{x}_t)\| > \varepsilon,\ \forall t \leq T) &\leq \mathbf{Pr}\left(\frac{1}{T}\sum_{t=1}^{T} \|\nabla f(\mathbf{x}_t)\| > \varepsilon\right) \\
&\leq \frac{\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} \|\nabla f(\mathbf{x}_t)\|\right]}{\varepsilon} \\
&= \frac{\mathbb{E}\left[\|\nabla f(\mathbf{x}_a)\|\right]}{\varepsilon} \leq \frac{\sqrt{\mathbb{E}\left[\|\nabla f(\mathbf{x}_a)\|^2\right]}}{\varepsilon},
\end{aligned}$$

where the 2nd inequality is due to Markov's inequality, and the expectation in the final line is taken over choosing a uniformly from $\{1, \ldots, T\}$ in addition to the other random variables, and the final inequality is Jensen's inequality.

Thus, using Lemma 20,

$$\mathbf{Pr}(\tau_{\text{fsp}} > T) \leq \frac{1}{\varepsilon}\sqrt{\frac{\Delta_f}{T\bar{\gamma}} + \frac{d}{\bar{\gamma}}\left(\frac{L}{2} + c_0\right)\frac{C_0}{T^\nu}} \overset{\text{let}}{=} p, \tag{2.17}$$

where p is the failure probability. As $\bar{\gamma}$ is a positive constant independent of d, ε and T, the equation above transforms into $\frac{\Delta_f}{T\bar{\gamma}} + \frac{d}{\bar{\gamma}}\left(\frac{L}{2} + c_0\right)\frac{C_0}{T^\nu} = \varepsilon^2 p$. As $\nu \geq 1$, $T = \mathcal{O}\left(\frac{\Delta_f d}{\bar{\gamma}\varepsilon^2 p}\right)$. □

2.4 Ergodicity property of SGLD

The argument of ergodicity comprises of two parts: recurrence and reachability. The LD term in the optimization scheme, due to its random-walk nature, is the key for the reachability argument. In this section we follow the framework of [CDT19] while giving new specific proofs.

2.4.1 Recurrence

We first show that with Langevin dynamics, the iteration process will visit sublevel sets of interest, for instance the collection of compact neighborhoods of all local minimums, infinitely many times. Lemma 13 is the first pillar to establish the ergodicity result.

In its proof, we first give a more explicit characterization of function value decrease between two successive SGLD-VR updates, as Lyapunov function R_t defined in lemma 19 involves sequences β_t and c_t whose analytical expression is hard to work with. Next, we construct a supermartingale involving the objective function value and iteration count. Through the introduction of a stopping time sequence which records the time of the iteration visiting targeted sublevel sets, one can establish the expectation of any entry in this stopping time sequence, thus proving the lemma.

Lemma 21 (Recurrence, repeat of lemma 13)**.** *Let n_0 be the index such that $\eta_{n_0} \leq \delta$, and n_k be the sequence of iteration index $n_{k+1} = \min_s\{s : s > n_k,\, \eta_{n_k:s} \geq \delta\}$.*

Under the regularization conditions in Assumption 5 and under Assumption 4, there exists a constant C_1 such that $\alpha = 1-2\exp(-(1-C_1)\mu\delta)$, $B = 2(\frac{2\eta_0 L^3}{B_\text{e}} f(\mathbf{x}_0)+\frac{\rho_0^2 Ld}{2\eta_0})$ and $K = \frac{\ln\frac{f(\mathbf{x}_{n_0})}{\delta B}}{(1-C_1)\mu\delta}$, with the stopping time sequence $\{\tau_k\}$ defined as $\tau_0 = K$ and $\tau_{k+1} = \min\{t : t \geq \tau_k + 1, f(\mathbf{x}_{n_t}) \leq 2\delta B\}$,

$$\mathbb{E}\left[\tau_j\right] \leq \frac{4}{\alpha} + K + j\left(\frac{1}{2\alpha\delta} + 1\right) \tag{2.18}$$

Remark As we have assumed that $f \geq 0$ which caters for the non-negativity property of the empirical risk, it is desirable that $f(\mathbf{x}_{n_t})$ goes to 0 as the iteration proceeds. Thus, the choice of δ for analytical purposes would be $\delta \propto \dfrac{\widehat{\varepsilon}}{B}$ for some $\widehat{\varepsilon}$-target level one deems appropriate.

Proof. Recall from proof in FSP, conditioned on $\mathcal{F}_{t-1}$ and $f(\widetilde{\mathbf{x}}) < f(\mathbf{x}_0)$, we have

$$
\begin{aligned}
&\mathbb{E}\left[f(\mathbf{x}_{t+1})\right] \\
&\leq f(\mathbf{x}_t) - \eta_t \|\nabla_t\|^2 + \mathbb{E}\,\frac{\eta_t^2 L}{2}\|\widetilde{\nabla}_t\|^2 + \frac{\rho_t^2 L}{2}\mathbb{E}\,\|\boldsymbol{\epsilon}_t\|^2 \\
&\overset{(2.12)}{\leq} f(\mathbf{x}_t) - \eta_t \|\nabla_t\|^2 + \frac{\eta_t^2 L}{2}(2\|\nabla_t\|^2 + 2\frac{L^2}{B_{\mathrm{e}}}\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2) + \frac{\rho_t^2 L d}{2} \\
&= f(\mathbf{x}_t) - (\eta_t - \eta_t^2 L)\|\nabla_t\|^2 + \frac{\eta_t^2 L^3}{B_{\mathrm{e}}}\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2 + \frac{\rho_t^2 L}{2}\mathbb{E}\,\|\boldsymbol{\epsilon}_t\|^2 \\
&\overset{(2.7)}{\leq} f(\mathbf{x}_t) - (\eta_t - \eta_t^2 L)\|\nabla_t\|^2 + \frac{2\eta_t^2 L^3}{B_{\mathrm{e}}}\big(\mu_2(f(\mathbf{x}_t) + f(\mathbf{x}_0)) + 2\psi_2\big) + \frac{\rho_t^2 L d}{2} \\
&= (1 + \frac{2\eta_t^2 L^3 \mu_2}{B_{\mathrm{e}}})f(\mathbf{x}_t) - (\eta_t - \eta_t^2 L)\|\nabla_t\|^2 + \frac{2\eta_t^2 L^3}{B_{\mathrm{e}}}\big(\mu_2 f(\mathbf{x}_0) + 2\psi_2\big) + \frac{\rho_t^2 L d}{2} \\
&\overset{(2.6)}{\leq} (1 + \frac{2\eta_t^2 L^3 \mu_2}{B_{\mathrm{e}}})f(\mathbf{x}_t) - (\eta_t - \eta_t^2 L)\big(\mu_1 f(\mathbf{x}_t) + \psi_1\big) + \frac{2\eta_t^2 L^3}{B_{\mathrm{e}}}\big(\mu_2 f(\mathbf{x}_0) + 2\psi_2\big) + \frac{\rho_t^2 L d}{2} \\
&= \left(1 - \mu_1 \eta_t + \eta_t^2(\frac{2L^3\mu_2}{B_{\mathrm{e}}} + \mu_1 L)\right) f(\mathbf{x}_t) - (\eta_t - \eta_t^2 L)\psi_1 + \frac{2\eta_t^2 L^3}{B_{\mathrm{e}}}\big(\mu_2 f(\mathbf{x}_0) + 2\psi_2\big) + \frac{\rho_t^2 L d}{2} \\
&\leq \exp\left(-(1 - C_1)\mu_1 \eta_t\right) f(\mathbf{x}_t) + \frac{2\eta_t^2 L^3}{B_{\mathrm{e}}}\big(\mu_2 f(\mathbf{x}_0) + 2\psi_2\big) + \frac{\rho_t^2 L d}{2} \qquad (2.19)
\end{aligned}
$$

Here C_1 is a positive constant such that $\eta_0(\frac{2L^3\mu_2}{\mu_1 B_{\mathrm{e}}} + L) < C_1 < 1$ for small enough η_0.

We introduce index partition to characterize the function value decrease. Let n_0 be the index such that $\eta_{n_0} \leq \delta$, and n_k be the sequence of iteration index $n_{k+1} = \min_s\{s : s > n_k,\, \eta_{n_k:s} \geq \delta\}$. Then $\eta_{n_k:n_{k+1}} \leq 2\delta$.

Before the proof proceeds, we recall the setting of η_t and ρ_t: let $\nu \geq 1$,

$$\eta_t = \frac{\eta_0}{t^\nu} \text{ and } \rho_t = \frac{\rho_0}{t^{\nu/2}}$$

Thus $\rho_t = \rho_0\sqrt{\frac{\eta_t}{\eta_0}}$. Iterating (2.19) m times, we have

$$\begin{aligned}
\mathbb{E}\, f(\mathbf{x}_{t+m}) \leq& \exp\left(-(1-C_1)\mu_1\eta_{t:t+m-1}\right) f(\mathbf{x}_t) + \\
&\sum_{i=t}^{t+m-1} \exp\left(-(1-C_1)\mu_1\eta_{i+1:t+m-1}\right) \eta_i \left(\frac{2\eta_i L^3}{B_{\mathrm{e}}}\left(\mu_2 f(\mathbf{x}_0) + 2\psi_2\right) + \frac{\rho_0^2 L d}{2\eta_0}\right) \\
&\leq \exp\left(-(1-C_1)\mu_1\eta_{t:t+m-1}\right) f(\mathbf{x}_t) + \sum_{i=t}^{t+m-1} \eta_i \left(\frac{2\eta_i L^3}{B_{\mathrm{e}}}\left(\mu_2 f(\mathbf{x}_0) + 2\psi_2\right) + \frac{\rho_0^2 L d}{2\eta_0}\right) \\
&\leq \exp\left(-(1-C_1)\mu_1\eta_{t:t+m-1}\right) f(\mathbf{x}_t) + \eta_{t:t+m-1} \left(\frac{2\eta_t L^3}{B_{\mathrm{e}}}\left(\mu_2 f(\mathbf{x}_0) + 2\psi_2\right) + \frac{\rho_0^2 L d}{2\eta_0}\right)
\end{aligned} \tag{2.20}$$

Setting $t = n_{k-1}$ and $m = n_k - n_{k-1}$, inequality (2.20) takes the form

$$\begin{aligned}
&\mathbb{E}\, f(\mathbf{x}_{n_k}) \\
&\leq \exp\left(-(1-C_1)\mu_1\eta_{n_{k-1}:n_k-1}\right) f(\mathbf{x}_{n_{k-1}}) + \eta_{n_{k-1}:n_k-1} \left(\frac{2\eta_{n_{k-1}} L^3}{B_{\mathrm{e}}}\left(\mu_2 f(\mathbf{x}_0) + 2\psi_2\right) + \frac{\rho_0^2 L d}{2\eta_0}\right) \\
&\leq \exp\left(-(1-C_1)\mu_1\delta\right) f(\mathbf{x}_{n_{k-1}}) + \eta_{n_{k-1}:n_k-1} \left(\frac{2\eta_{n_{k-1}} L^3}{B_{\mathrm{e}}}\left(\mu_2 f(\mathbf{x}_0) + 2\psi_2\right) + \frac{\rho_0^2 L d}{2\eta_0}\right) &(2.21) \\
&\leq \exp\left(-(1-C_1)k\mu_1\delta\right) f(\mathbf{x}_{n_0}) + \delta\, \underbrace{2\left(\frac{2\eta_0 L^3}{B_{\mathrm{e}}}\left(\mu_2 f(\mathbf{x}_0) + 2\psi_2\right) + \frac{\rho_0^2 L d}{2\eta_0}\right)}_{:=B}. &(2.22)
\end{aligned}$$

Consider a function value threshold $M := 2\delta B$. From (2.22), it follows that $\mathbb{E}\, f(\mathbf{x}_{n_k}) \leq M$ when

$$k \geq \frac{\ln \frac{f(\mathbf{x}_{n_0})}{\delta B}}{(1-C_1)\mu_1\delta} := K$$

Now we show that the expected time for the function value to decrease to below this threshold M is upper bounded by a finite number, thus justifying the recurrence of the iteration process to a compact sub-level set. To better exploit the indices partition $\{n_k\}$ of the iteration sequence, define $f(\mathbf{x}_{n_k}) := V_k$, and $\tau = \min\{k : k \geq K, f(\mathbf{x}_{n_k}) \leq M\}$. We claim that

$$V_{\tau\wedge k} + \alpha\delta B \cdot (\tau \wedge k)$$

is a supermartingale with $\alpha = 1 - 2\exp\left(-(1-C_1)\mu_1\delta\right)$, i.e.

$$\mathbb{E}\left[V_{\tau\wedge(k+1)} + \alpha\delta B(\tau \wedge (k+1)) | V_{\tau\wedge k}\right] \leq V_{\tau\wedge k} + \alpha\delta B(\tau \wedge k) \tag{2.23}$$

When $\tau \leq k$, (2.23) holds trivially. When $\tau \geq k+1$, then $V_{k+1} > M$. The relation (2.23) to show in this case takes the form $\alpha\delta B \leq V_k - \mathbb{E}\left[V_{k+1}|V_k\right]$. To let this happen, taking (2.22) into consideration, a sufficient condition is $\mathbb{E}\left[V_{k+1}|V_k\right] \leq \exp(-(1-C_1)\mu_1\delta)V_k + \delta B \leq V_k - \alpha\delta B$, i.e. $(1+\alpha)\delta B \leq \left(1-\exp(-(1-C_1)\mu_1\delta)\right)V_k$. Considering that $\tau > k+1$ implies $V_k > M$, then the previous sufficient condition to show can be further strengthened to $(1+\alpha)\delta B \leq (1-\exp(-(1-C_1)\mu_1\delta)M$, which is catered for per definition of α.

To show that a compact sub-level set is going to be visited by the iteration sequence for infinitely many times, we introduce the stopping time sequence $\{\tau_k\}$ where $\tau_0 = K$ and $\tau_{k+1} = \min\{t : t \geq \tau_k + 1, f(\mathbf{x}_{n_t}) \leq M\}$. Per the same argument as in previous paragraph, $\mathbb{E}\left[V_{\tau_{k+1}} + \alpha\delta B\tau_{k+1}\middle|\tau_k\right] \leq V_{\tau_k+1} + \alpha\delta B(\tau_k+1)$, which gives

$$\alpha\delta B\,\mathbb{E}\left[\tau_{k+1} - \tau_k - 1|\tau_k\right] \leq V_{\tau_k+1} - \mathbb{E}\left[V_{\tau_{k+1}}|\tau_k\right]$$

Taking total expectation, and summing over all k from 0 to j with $\tau_0 = K$, we have

$$\alpha\delta B\left(\mathbb{E}\left[\tau_j\right] - K - j\right) \leq \sum_{k=0}^{j}\mathbb{E}\left[V_{\tau_k+1} - V_{\tau_{k+1}}\right] \tag{2.24}$$

By (2.19), $\mathbb{E}\left[V_{\tau_k+1}\right] \leq \exp\left(-(1-C_1)\mu_1\eta_{\tau_k}\right)V_{\tau_k} + \frac{B}{2} \leq V_{\tau_k} + \frac{B}{2}$, thus

$$\alpha\delta B\left(\mathbb{E}\left[\tau_j\right] - K - j\right) \leq \mathbb{E}\left[V_K - V_{\tau_j}\right] + j\frac{B}{2} \leq 2M + j\frac{B}{2}$$

i.e.

$$\mathbb{E}\left[\tau_j\right] \leq \frac{4}{\alpha} + K + j(\frac{1}{2\alpha\delta} + 1)$$

□

2.4.2 Reachability

We show that when the SGLD iteration sequence starts from a recurrent compact set, there is a positive possibility for the sequence to visit every nearby first-order stationary points. The following Lemma 22 is stated as a fact, whose proof is straightforward computation, and will be needed for bounding the variance of the variance-reduced gradient estimator later in this section.

Lemma 22 (Variance of subset selection)**.** *Consider a dataset $\{\mathbf{a}_i\}_{i=1}^N$ with mean*

$$\bar{\mathbf{a}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{a}_i.$$

Select b elements uniformly ($1 \leq b \leq N$) out of this dataset, and denote the index set of these selected elements as $\mathcal{I}$. The subsampled mean, which is a random variable, is

$$\boldsymbol{\xi} = \frac{1}{b}\sum_{i\in\mathcal{I}} \mathbf{a}_i.$$

The variance of $\boldsymbol{\xi}$ is

$$\begin{aligned}\mathbb{E}_{\mathcal{I}}\|\boldsymbol{\xi} - \bar{\mathbf{a}}\|^2 &= \mathbb{E}_{\mathcal{I}}(\boldsymbol{\xi}^2 - 2\langle\boldsymbol{\xi}, \bar{\mathbf{a}}\rangle + \bar{\mathbf{a}}^2) = \mathbb{E}_{\mathcal{I}}\boldsymbol{\xi}^2 - \bar{\mathbf{a}}^2 \\ &= \frac{N-b}{N^2 b}\mathrm{Var}[\mathbf{a}] = \frac{N-b}{(N-1)b}\left(\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{a}_i - \bar{\mathbf{a}}\|^2\right) \end{aligned} \tag{2.25}$$

Lemma 23 will be used to show that inside a stepsize batch, the reachability property will not be hindered by the gradient descent part in the iteration, thus allowing the Gaussian noise terms to give the desired property.

Lemma 23. *For any sequence $a_k > 0$ such that there is a constant ν and $\sum_{j=1}^n a_j \leq 2\nu$, let $\mathcal{F}_{\mathbf{z}}$ denote the σ-algebra generated by $\mathbf{z}_1, \cdots, \mathbf{z}_n$. Suppose $\boldsymbol{\xi}_k$ is a sequence of random vectors such that*

$$\mathbb{E}\left(\boldsymbol{\xi}_k \,|\, \mathcal{F}_{\mathbf{z}}\right) = 0 \quad \mathbb{E}\left(\|\boldsymbol{\xi}_k\|^2 \,|\, \mathcal{F}_{\mathbf{z}}\right) \leq C_2$$

Let $\mathbf{y}_k = \sum_{j=1}^k a_j\boldsymbol{\xi}_j$, then

$$\mathbf{Pr}(\|\mathbf{y}_k\| \leq 4\nu\sqrt{C_2}) \geq \frac{1}{2} \tag{2.26}$$

Proof. In the proof for this lemma, all expectations are conditioned on $\mathcal{F}_{\mathbf{z}}$. With Jensen's inequality, $(\mathbb{E}\,\|\boldsymbol{\xi}_k\|)^2 \leq \mathbb{E}\left(\|\boldsymbol{\xi}_k\|^2\right) \leq C_2$. By Markov's inequality,

$$\mathbf{Pr}(\|\mathbf{y}_k\| \geq 4\nu\sqrt{C_2}) \leq \frac{\mathbb{E}\,\|\mathbf{y}_k\|}{4\nu\sqrt{C}_2} = \frac{\mathbb{E}\,\|\sum_{j=1}^k a_j\boldsymbol{\xi}_{\boldsymbol{j}}\|}{4\nu\sqrt{C}_2} \leq \frac{\mathbb{E}\,\sum_{j=1}^k a_j\|\boldsymbol{\xi}_j\|}{4\nu\sqrt{C_2}} \leq \frac{1}{2}$$

□

Lemma 24 is the core lemma to establish ergodicity result for the LD optimization scheme. The core idea behind its proof is to leverage the exploratory potential of a *radial* Brownian motion process to show that there is a non-trivial probability for the Gaussian noise accumulation in the LD scheme to visit a pre-designated point in space.

Lemma 24 (Ergodicity due to Brownian motion)**.** *Given any sequence $a_k > 0$, let $\mathbf{z}_k = \sum_{i=1}^{k} \rho_0 \sqrt{a_i} \boldsymbol{\epsilon}_i$ where $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, I_d)$, for any target vector $\mathbf{z}^\star$ and distance r, there exists a non-negative function p_1 such that*

$$\mathbf{Pr}(\|\mathbf{z}_n - \mathbf{z}^\star\| \leq r,\ \|\mathbf{z}_k\| \leq \|\mathbf{z}^\star\| + r\ \forall k = 1, \cdots, n) \geq p_1(r, \rho_0, t_n, \mathbf{z}^\star)$$

such that $p_1(0, \rho_0, t_n, \mathbf{z}^\star) = p_1(r, 0, t_n, \mathbf{z}^\star) = 0$.

Proof. We first give lower bounds to factors $\mathbf{Pr}(\|\mathbf{z}_n - \mathbf{z}^\star\| \leq r)$ and $\mathbf{Pr}(\|\mathbf{z}_k\| \leq \|\mathbf{z}\| + r\ \forall k = 1, \cdots, n)$ respectively, and then conclude the proof with $\mathbf{Pr}(\|\mathbf{z}_n - \mathbf{z}^\star\| \leq r,\ \|\mathbf{z}_k\| \leq \|\mathbf{z}\| + r\ \forall k = 1, \cdots, n) \geq \mathbf{Pr}(\|\mathbf{z}_n - \mathbf{z}^\star\| \leq r) \cdot \mathbf{Pr}(\|\mathbf{z}_k\| \leq \|\mathbf{z}\| + r\ \forall k = 1, \cdots, n)$.

For $\mathbf{Pr}(\|\mathbf{z}_n - \mathbf{z}^\star\|_2 \leq r)$, $\|\mathbf{z}_n - \mathbf{z}^\star\|_2 \leq \|\mathbf{z}_n - \mathbf{z}^\star\|_1 = \sum_{\text{dim}=1}^{d} |(\mathbf{z}_n)_{\text{dim}} - z^\star_{\text{dim}}|$, therefore $\mathbf{Pr}(\|\mathbf{z}_n - \mathbf{z}^\star\|_2 \leq r) \geq \mathbf{Pr}(|(\mathbf{z}_n)_{\text{dim}} - z^\star_{\text{dim}}| \leq \frac{r}{d}\ \forall \text{dim} \in [d]) = \prod_{\text{dim}=1}^{d} \mathbf{Pr}(|(\mathbf{z}_n)_{\text{dim}} - z^\star_{\text{dim}}| \leq \frac{r}{d})$. Notice that $(\mathbf{z}_n)_{\text{dim}}$ has the distribution of the Brownian motion B_{t_n} where $t_k = \rho_0^2 \sum_{i=1}^{k} a_i$, $k = 1, 2, \cdots, n$. By [KT75], $\mathbf{Pr}(|(\mathbf{z}_n)_{\text{dim}} - z^\star_{\text{dim}}| \leq \dfrac{r}{d}) = \displaystyle\int_{\max\{z^\star_{\text{dim}} - \frac{r}{d}, 0\}}^{z^\star_{\text{dim}} + \frac{r}{d}} p_{t_n}(z^\star_{\text{dim}}, y)\, \mathrm{d}y$, where $p_t(x, y) = \sqrt{\frac{2}{\pi t}} \exp(-\frac{x^2 + y^2}{2t}) \cosh(\frac{xy}{t})$. Hence,

$$\mathbf{Pr}(\|\mathbf{z}_n - \mathbf{z}^\star\|_2 \leq r) \geq \left(\min_{\text{dim}} \int_{\max\{z^\star_{\text{dim}} - \frac{r}{d}, 0\}}^{z^\star_{\text{dim}} + \frac{r}{d}} p_{t_n}(z^\star_{\text{dim}}, y)\, \mathrm{d}y \right)^d \tag{2.27}$$

For $\mathbf{Pr}(\|\mathbf{z}_k\| \leq \|\mathbf{z}^\star\| + r,\ \forall k \in [n])$, we have the following lower bound:

$$\begin{aligned}\mathbf{Pr}(\|\mathbf{z}_k\| \leq \|\mathbf{z}^\star\| + r,\ \forall k \in [n]) &\geq \mathbf{Pr}\left(\max_k |(\mathbf{z}_k)_{\text{dim}}| \leq \frac{\|\mathbf{z}^\star\| + r}{\sqrt{d}},\ \forall \text{dim} \in [d] \right) \\ &= \mathbf{Pr}\left(\max_k | \underbrace{(\mathbf{z}_k)_1}_{\text{1D B.M.}} | \leq \frac{\|\mathbf{z}_\star\| + r}{\sqrt{d}} \right)^d \\ &= \left(1 - \mathbf{Pr}\left(\max_k |(\mathbf{z}_k)_1| \geq \frac{\|\mathbf{z}^\star\| + r}{\sqrt{d}} \right) \right)^d\end{aligned}$$

Now notice $\mathbf{Pr}(\max_k |(\mathbf{z}_k)_1| \geq \frac{\|\mathbf{z}^\star\|+r}{\sqrt{d}}) = \mathbf{Pr}\Big(\max_k (\mathbf{z}_k)_1 > \frac{\|\mathbf{z}^\star\|+r}{\sqrt{d}}$ or $\min_k (\mathbf{z}_k)_1 < -\frac{\|\mathbf{z}^\star\|+r}{\sqrt{d}} \Big) \leq \mathbf{Pr}\big(\max_k (\mathbf{z}_k)_1 > \frac{\|\mathbf{z}^\star\|+r}{\sqrt{d}}\big) + \mathbf{Pr}\big(\min_k (\mathbf{z}_k)_1 < -\frac{\|\mathbf{z}^\star\|+r}{\sqrt{d}}\big) = 2\mathbf{Pr}\big(\max_k (\mathbf{z}_k)_1 > \frac{\|\mathbf{z}^\star\|+r}{\sqrt{d}}\big)$. Then, by the reflection principle of Brownian motion,

$$\mathbf{Pr}\left(\max_k (\mathbf{z}_k)_1 > \frac{\|\mathbf{z}^\star\| + r}{\sqrt{d}} \right) = 2\mathbf{Pr}\left((\mathbf{z}_n)_1 \geq \frac{\|\mathbf{z}^\star\| + r}{\sqrt{d}} \right)$$

Therefore,

$$\begin{aligned}
\mathbf{Pr}(\|\mathbf{z}_k\| \leq \|\mathbf{z}^\star\| + r,\, \forall k \in [n]) &\geq \left(1 - 4\mathbf{Pr}\left(\underbrace{(\mathbf{z}_n)_1}_{\sim \mathcal{N}(0,t_n)} \geq \frac{\|\mathbf{z}^\star\| + r}{\sqrt{d}} \right) \right)^d \\
&= \left(1 - 4\mathbf{Pr}\left(\frac{(\mathbf{z}_n)_1}{\sqrt{t_n}} \geq \frac{\|\mathbf{z}^\star\| + r}{\sqrt{dt_n}} \right) \right)^d \\
&= \left(1 - 2\mathbf{Pr}\left(\frac{(\mathbf{z}_n)_1}{\sqrt{t_n}} \geq \frac{\|\mathbf{z}^\star\| + r}{\sqrt{dt_n}} \right) - 2\mathbf{Pr}\left(\frac{(\mathbf{z}_n)_1}{\sqrt{t_n}} \leq -\frac{\|\mathbf{z}^\star\| + r}{\sqrt{dt_n}} \right) \right)^d \\
&= \left(1 - 2\big(1 - \mathbf{Pr}\left(-\frac{\|\mathbf{z}^\star\| + r}{\sqrt{dt_n}} \leq \frac{(\mathbf{z}_n)_1}{\sqrt{t_n}} \leq \frac{\|\mathbf{z}^\star\| + r}{\sqrt{dt_n}} \right) \big) \right)^d \\
&= \left(2 \int_{-\frac{\|\mathbf{z}^\star\|+r}{\sqrt{dt_n}}}^{\frac{\|\mathbf{z}^\star\|+r}{\sqrt{dt_n}}} \frac{\exp(-x^2/2)}{\sqrt{2\pi}} \,\mathrm{d}x - 1 \right)^d \\
&\geq \left(4 \frac{\|\mathbf{z}^\star\| + r}{\sqrt{2\pi dt_n}} \exp\left(-\frac{1}{2} \frac{(\|\mathbf{z}^\star\| + r)^2}{dt_n} \right) - 1 \right)^d \qquad (2.28)
\end{aligned}$$

Let $p_1(r, \rho_0, t_n, \mathbf{z}^\star)$ be the product of two lower bounds (2.27) and (2.28) above, recall that $t_n = \rho_0^2 \sum_{i=1}^n a_i$, we define $p_1(r, \rho_0, t_n, \mathbf{z}^\star)$ as the following,

$$p_1(r, \rho_0, t_n, \mathbf{z}^\star) := \left(\min_{\dim} \int_{\max\{z^\star_{\dim} - \frac{r}{d}, 0\}}^{z^\star_{\dim} + \frac{r}{d}} p_{t_n}(z^\star_{\dim}, y) \,\mathrm{d}y \right)^d \cdot \left(4 \frac{\|\mathbf{z}^\star\| + r}{\sqrt{2\pi dt_n}} \exp\left(-\frac{1}{2} \frac{(\|\mathbf{z}^\star\| + r)^2}{dt_n} \right) - 1 \right)^d \qquad (2.29)$$

where $p_{t_n}(x, y) = \sqrt{\frac{2}{\pi t}} \exp(-\frac{x^2+y^2}{2t_n}) \cosh(\frac{xy}{t_n})$.

To make the dependence of the first factor in p_1 on parameters more explicit, for some

$\xi \in (\max\{z^\star_{\text{dim}} - \frac{r}{d}, 0\}, z^\star_{\text{dim}} + \frac{r}{d})$,

$$\begin{aligned}\int_{\max\{z^\star_{\text{dim}} - \frac{r}{d}, 0\}}^{z^\star_{\text{dim}} + \frac{r}{d}} p_{t_n}(z^\star_{\text{dim}}, y)\,\mathrm{d}y &\geq \sqrt{\frac{2}{\pi t_n}} \exp(-\frac{(z^\star_{\text{dim}})^2 + \xi^2}{2t_n}) \cosh(\frac{z^\star_{\text{dim}}\xi}{t_n})\frac{r}{d} \\ &= \sqrt{\frac{2}{\pi t_n}} \left(\exp(-\frac{(z^\star_{\text{dim}} - \xi)^2}{2t_n}) + \exp(-\frac{(z^\star_{\text{dim}} + \xi)^2}{2t_n})\right)\frac{r}{2d} \\ &\geq \sqrt{\frac{2}{\pi t_n}} 2\sqrt{\exp(-\frac{(z^\star_{\text{dim}} - \xi)^2}{2t_n} - \frac{(z^\star_{\text{dim}} + \xi)^2}{2t_n})}\frac{r}{2d} \\ &= \sqrt{\frac{2}{\pi t_n}} \exp(-\frac{(z^\star_{\text{dim}})^2 + \xi^2}{2t_n})\frac{r}{d} \\ &\geq \sqrt{\frac{2}{\pi t_n}} \exp(-\frac{(z^\star_{\text{dim}})^2 + (z^\star_{\text{dim}} + \frac{r}{d})^2}{2t_n})\frac{r}{d}\end{aligned}$$

We thus redefine p_1 as

$$p_1(r, \rho_0, t_n, \mathbf{z}^\star) := \left(\min_{\text{dim}} \sqrt{\frac{2}{\pi t_n}} \exp(-\frac{(z^\star_{\text{dim}})^2 + (z^\star_{\text{dim}} + \frac{r}{d})^2}{2t_n})\frac{r}{d}\right)^d \cdot \left(4\frac{\|\mathbf{z}^\star\| + r}{\sqrt{2\pi d t_n}} \exp\left(-\frac{1}{2}\frac{(\|\mathbf{z}^\star\| + r)^2}{d t_n}\right) - 1\right)^d \tag{2.30}$$

then we have the lemma 24. □

We leverage lemma 24 to show the reachability of SGLD-VR scheme, which is the second pillar to establish the ergodicity result. The core idea behind the proof of lemma 25 is to balance the influence on the iterates from gradient descent and Gaussian noise accumulation respectively, and show that the exploratory potential behind the Gaussian noise accumulation will fulfill the desired property of reachability.

Lemma 25 (Reachability)**.** *Assume the same stepsize batch setting as in lemma 13 and the gradient Lipschitz condition in assumption 4, with respect to an arbitrary target point* $\mathbf{s}$*, suppose that* $D_F = \frac{1}{2}\rho_0 dL$*, for any* $\varepsilon > 0$,

$$\mathbf{Pr}\left(\|\mathbf{x}_{n_{i+1}} - \mathbf{s}\| \leq \widetilde{\varepsilon}\right) > p_2(\widetilde{\varepsilon}, \rho_0, t_{n_{i+1}}, \mathbf{s}) \tag{2.31}$$

where $\widetilde{\varepsilon} = \varepsilon + 2\delta\sqrt{C_2} + \delta D_F$*, and* $p_2(\varepsilon, \rho_0, t_n, \mathbf{s}) = \frac{1}{2}p_1(\varepsilon, \rho_0, t_n, \mathbf{s})$

Proof. Denote $\mathbf{x}_o = \mathbf{x}_{n_i}$ and $\mathbf{d} = \mathbf{s} - \mathbf{x}_o$. Recall the SGLD scheme in a batch goes as

$$\begin{aligned}
\mathbf{x}_{k+1} &= \mathbf{x}_k - \eta_k \widetilde{\nabla}_k + \rho_k \boldsymbol{\epsilon}_k = \mathbf{x}_o - \sum_{l=0}^{k} \eta_l \widetilde{\nabla}_l + \rho_l \boldsymbol{\epsilon}_l \\
&= \mathbf{x}_o - \sum_{l=0}^{k} \eta_{l+o} \left(\nabla f(\mathbf{x}_{l+o}) + \big(\frac{1}{B}\nabla f_{I_{l+o}}(\mathbf{x}_{l+o}) - \nabla f(\mathbf{x}_l)\big) - \big(\frac{1}{B}\nabla f_{I_{l+o}}(\widetilde{\mathbf{x}}) - \nabla f(\widetilde{\mathbf{x}})\big) \right) \\
&\quad + \sum_{l=0}^{k} \rho_{l+o} \boldsymbol{\epsilon}_{l+o} \\
&= \mathbf{x}_o - \sum_{l=0}^{k} \eta_{l+o} \nabla f(\mathbf{x}_{l+o}) - \mathbf{y}_k + \mathbf{z}_k
\end{aligned} \tag{2.32}$$

where we define $\mathbf{y}_k := \sum_{l=0}^{k} \eta_l\big(\frac{1}{B}\nabla f_{I_l}(\mathbf{x}_l) - \nabla f(\mathbf{x}_l)\big) - \eta_l\big(\frac{1}{B}\nabla f_{I_l}(\widetilde{\mathbf{x}}) - \nabla f(\widetilde{\mathbf{x}})\big)$ and $\mathbf{z}_k := \sum_{l=0}^{k} \rho_0 \sqrt{\frac{\eta_l}{\eta_0}} \boldsymbol{\epsilon}_l$. Note that $\widetilde{\mathbf{x}}$ can change as moving from stepsize batch i to $i+1$ may involve different SVRG batch reference points.

Let $m = n_{i+1} - n_i$. Per law of total probability, Denote the event $\mathcal{E}_{i+1} = \{\|\mathbf{z}_{n_{i+1}-1} - \mathbf{d}\| \leq \varepsilon,\ \|\mathbf{z}_k\| \leq \|\mathbf{d}\| + \varepsilon\ \forall k \in [m] + n_i\}$, then

$$\mathbf{Pr}\left(\|\mathbf{x}_{n_{i+1}} - \mathbf{s}\| \leq \widetilde{\varepsilon}\right) \geq \mathbf{Pr}\left(\|\mathbf{x}_{n_{i+1}} - \mathbf{s}\| \leq \widetilde{\varepsilon} \,\middle|\, \mathcal{E}_{i+1}\right) \cdot \mathbf{Pr}\left(\mathcal{E}_{i+1}\right) \tag{2.33}$$

where $\mathbf{Pr}\left(\mathcal{E}_{i+1}\right)$ is lowered bounded by p_1 from lemma 24. What is left in this proof is to bound the first factor in the above equation.

Now we bound the gradient terms and gradient difference terms in (2.32), thus computing the probability $\mathbf{Pr}\left(\|\mathbf{x}_{n_{i+1}} - \mathbf{s}\| \leq \widetilde{\varepsilon} \,\middle|\, \mathcal{E}_{i+1}\right)$. Recall that the Langevin dynamics has the corresponding continuous form $\mathrm{d}\mathbf{x}(t) = -\eta(t)\nabla f\big(\mathbf{x}(t)\big)\,\mathrm{d}t + \rho_t\,\mathrm{d}\mathbf{W}(t)$ where $\mathbf{W}(t)$ is a d-dimensional Brownian motion.

Consider the target function $v(t, \mathbf{x}_t) = f(\mathbf{x}_t)$. Per Feynman-Kac formula [Pha09], the target function $f(\mathbf{x}_t)$ satisfies the linear parabolic PDE

$$-\|\nabla f(\mathbf{x}_t)\|^2 + \frac{1}{2}\rho_t^2 \mathbf{Tr}[\nabla^2 f(\mathbf{x}_t)] = 0$$

Owing to Lipschitz assumption about gradient, $\mathbf{Tr}[\nabla^2 f] \leq dL$, which gives

$$\|\nabla f\|^2 \leq \frac{1}{2}\rho_t dL \leq \frac{1}{2}\rho_0 dL := D_F \tag{2.34}$$

which gives the boundedness of the gradient in the iteration process. Furthermore, by assumption 5, $\{\mathbf{x}_t\}$ is also bounded within a stepsize batch.

As the gradient is bounded inside a stepsize batch $\{\eta_{n_i} : \eta_{n_{i+1}-1}\}$, by lemma 22, each summation term in $\mathbf{y}_k$ has the following variance upper bound

$$\frac{n-B}{(n-1)B}\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}_l)-\nabla f(\mathbf{x}_l)\|^2 \le \frac{n-B}{(n-1)B}\frac{1}{n}\max_{l\in[m]+o}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}_l)-\nabla f(\mathbf{x}_l)\|^2 := C_2$$

Unbiasedness of SVRG gradient estimator makes lemma 23 applicable to $\mathbf{y}_k$.

$$\begin{aligned}\mathbf{Pr}(\|\mathbf{x}_{n_{i+1}}-\mathbf{s}\| \le \widetilde{\varepsilon}\,|\,\mathcal{E}_{i+1}) &= \mathbf{Pr}(\|\mathbf{x}_o - \sum_{l=0}^{n_{i+1}-1}\eta_{l+o}\nabla f(\mathbf{x}_{l+o}) - \mathbf{y}_{n_{i+1}-1} + \mathbf{z}_{n_{i+1}-1} - \mathbf{s}\| \le \widetilde{\varepsilon}\,|\,\mathcal{E}_{i+1})\\ &= \mathbf{Pr}(\| - \sum_{l=0}^{n_{i+1}-1}\eta_{l+o}\nabla f(\mathbf{x}_{l+o}) - \mathbf{y}_{n_{i+1}-1} + \mathbf{z}_{n_{i+1}-1} - \mathbf{d}\| \le \widetilde{\varepsilon}\,|\,\mathcal{E}_{i+1})\\ &\ge \mathbf{Pr}(\|\sum_{l=0}^{n_{i+1}-1}\eta_{l+o}\nabla f(\mathbf{x}_{l+o})\| \le \delta D_F, \|\mathbf{y}_{n_{i+1}-1}\| \le 4\delta C_2, \,|\,\mathcal{E}_{i+1})\\ &\ge \mathbf{Pr}(\|\mathbf{y}_{n_{i+1}-1}\| \le 4\delta C_2, \,|\,\mathcal{E}_{i+1}) \ge \frac{1}{2}\end{aligned}$$

where the last inequality is due to the fact that the gradient of the objective f is bounded by D_F.

□

Now we are ready to prove the ergodicity result for the SGLD-VR scheme with the recurrence and reachability results above.

Theorem 26 (Repeat of theorem 14)**.** *Under regularization condition 5 and gradient assumption 4, with the same parameter setting as in lemma 13, for any* $\widetilde{\varepsilon} > 0$, $p > 0$ *and a point* $\mathbf{s}$, *there is a*

$$T = \mathcal{O}\left(\frac{1}{p\mu_1\left(4\eta_0 L^3\frac{\mu_2 f(\mathbf{x}_0)+2\psi_2}{B_e} + \frac{\rho_0^2}{\eta_0}Ld\right)}\left(1 + \ln f(\mathbf{x}_0) + \frac{(d\|\mathbf{s}\| + \widetilde{\varepsilon})^d}{\left((\frac{4}{\sqrt{2\pi}} - 1)\mathrm{e}^{-1/2}\widetilde{\varepsilon}\right)^d}\right)\right) \tag{2.35}$$

such that

$$\mathbf{Pr}(\|\mathbf{x}_t - \mathbf{s}\| \le \widetilde{\varepsilon} \textit{ for some } t < T) \ge 1 - p \tag{2.36}$$

Proof. Recall the definition of the stopping time sequence: $\tau_0 = K$, $\tau_{t+1} = \min\{t : t \ge \tau_k + 1, f(\mathbf{x}_{n_t}) \le M\}$. Further define $\tau_* = \min t : t > 0, \|\mathbf{x}_{n_t} - \mathbf{s}\| \le \varepsilon$. We show that $\mathbf{Pr}(\tau_* \ge T) \le \widetilde{p}$

with a proper choice of T. For any J,

$$\begin{aligned}
\mathbf{Pr}(\tau_* \geq T) &= \mathbf{Pr}(\tau_* \geq T, \tau_J > T) + \mathbf{Pr}(\tau_* \geq T, \tau_J < T) \\
&\leq \mathbf{Pr}(\tau_J > T) + \mathbf{Pr}(\|\mathbf{x}_{n_{\tau_k}+1} - \mathbf{s}\| > \varepsilon, \tau_J \leq T,\ \forall k \in [J]) \\
&\leq \mathbf{Pr}(\tau_J > T) + \mathbf{Pr}(\|\mathbf{x}_{n_{\tau_k}+1} - \mathbf{s}\| > \varepsilon,\ \forall k \in [J])
\end{aligned}$$

Lemma 13 gives that $\mathbb{E}\,\tau_J \leq \frac{4}{\alpha} + K + J(\frac{1}{2\alpha\delta} + 1)$, thus by Markov inequality

$$\mathbf{Pr}(\tau_J > T) \leq \frac{\mathbb{E}\,[\tau_J]}{T} \leq \frac{\frac{4}{\alpha} + K + J(\frac{1}{2\alpha\delta} + 1)}{T} \tag{2.37}$$

To ensure the last bound is below the pre-specified threshold $\dfrac{1}{2}p$, we need to take

$$T = \left[\frac{\frac{4}{\alpha} + K + J(\frac{1}{2\alpha\delta} + 1)}{\frac{p}{2}} \right] + 1 \tag{2.38}$$

By lemma 25, there is a $p_2 > 0$ such that

$$\mathbf{Pr}(\|\mathbf{x}_{n_{\tau_k}+1} - \mathbf{s}\| > \widetilde{\varepsilon},\ \forall k \in [J]) = \prod_{k=1}^{J} \mathbf{Pr}(\|\mathbf{x}_{n_{\tau_k}+1} - \mathbf{s}\| > \widetilde{\varepsilon}) \leq (1 - p_2)^J \overset{\text{let}}{\leq} \frac{p}{2} \tag{2.39}$$

To ensure the upper bound to be less than $\frac{p}{2}$, a sufficient condition is that

$$J > \frac{\ln \frac{p}{2}}{\ln(1 - p_2(\widetilde{\varepsilon}, \rho_0, t_{n_{\tau_J}+1}))} > \frac{\ln \frac{2}{p}}{p_2(\widetilde{\varepsilon}, \rho_0, t_{n_{\tau_J}+1})}.$$

We consider the dependence of T on the error tolerance $\widetilde{\varepsilon}$, dimension d and initial perturbation parameter ρ_0. Recall the definition of p_2 and we will upper bound it to rid of the dependence on t_n:

$$\begin{aligned}
&p_2(\widetilde{\varepsilon}, \rho_0, t_n, \mathbf{s}) \\
&= \frac{1}{2}\left(\min_{\text{dim}} \sqrt{\frac{2}{\pi t_n}} \exp(-\frac{(s_{\text{dim}})^2 + (s_{\text{dim}} + \frac{\widetilde{\varepsilon}}{d})^2}{2t_n}) \frac{\widetilde{\varepsilon}}{d}\right)^d \cdot \left(4\frac{\|\mathbf{s}\| + \widetilde{\varepsilon}}{\sqrt{2\pi d t_n}} \exp\left(-\frac{1}{2}\frac{(\|\mathbf{s}\| + \widetilde{\varepsilon})^2}{d t_n}\right) - 1\right)^d \\
&\leq \frac{1}{2}\left(\min_{\text{dim}} \sqrt{\frac{2}{\pi\left((s_{\text{dim}})^2 + (s_{\text{dim}} + \frac{\widetilde{\varepsilon}}{d})^2\right)}} \exp(-\frac{1}{2}) \frac{\widetilde{\varepsilon}}{d}\right)^d \cdot \left(4\frac{1}{\sqrt{2\pi}} - 1\right)^d
\end{aligned}$$

Therefore, a sufficient condition for (2.39) to hold is

$$J > \left(\ln \frac{2}{p}\right) \max_{\text{dim}} \frac{2}{\left(\sqrt{\frac{2}{\pi\left((s_{\text{dim}})^2 + (s_{\text{dim}} + \frac{\widetilde{\varepsilon}}{d})^2\right)}} \exp(-\frac{1}{2}) \frac{\widetilde{\varepsilon}}{d}\right)^d \cdot \left(4\frac{1}{\sqrt{2\pi}} - 1\right)^d} \tag{2.40}$$

Combining (2.38) and (2.40), the total amount of time needed for (2.36) to hold is

$$T = \left\lceil \frac{\frac{4}{\alpha} + K + J(\frac{1}{2\alpha\delta} + 1)}{\frac{p}{2}} \right\rceil + 1$$

Recall from lemma 13 parameter settings $B = 2\left(\frac{2\eta_0 L^3}{B_{\mathrm{e}}}\big(\mu_2 f(\mathbf{x}_0) + 2\psi_2\big) + \frac{\rho_0^2 Ld}{2\eta_0}\right)$, $\alpha = 1 - 2\exp(-(1-C_1)\mu_1\delta)$ and $K = \frac{\ln\frac{f(\mathbf{x}_{n_0})}{\delta B}}{(1-C_1)\mu_1\delta}$. In the light of the remark post the lemma 13, δ is to be set as $\delta \propto B^{-1}$ for minimizing the empirical risk purposes, hence

$$T = \mathcal{O}\left(\frac{1}{p\mu_1\big(4\eta_0 L^3\frac{\mu_2 f(\mathbf{x}_0)+2\psi_2}{B_{\mathrm{e}}} + \frac{\rho_0^2}{\eta_0}Ld\big)}\left(1 + \ln f(\mathbf{x}_0) + \frac{(d\|\mathbf{s}\| + \widetilde{\varepsilon})^d}{\big((\frac{4}{\sqrt{2\pi}} - 1)\mathrm{e}^{-1/2}\widetilde{\varepsilon}\big)^d}\right)\right) \tag{2.41}$$

□

2.5 Second-order stationary point convergence property

By far in the literature there are two common ways to argue the convergence to second-order stationary points (SSP)

- show that $f(\mathbf{x}_T) - f(\mathbf{x}_0) < \Delta_f$ with probabilistic guarantee to ensure the continual function value decrease at saddle point [JGN+17]
- show that $\|\mathbf{x}_T - \mathbf{x}^\star\|$ decreases in the probabilistic sense as T increases [KLY18].

  The time complexity of this approach has the exponential dependency on the inverse of the error tolerance. So in this work we resort to the previous approach.

The argument to show sufficient function value decrease from a FSP uses two iterate sequences to demonstrate the continual function value decrease at saddle point. Now that the noise is injected at every iteration, the geometric intuition that the trapping region is thin plus the probabilistic argument should be able to give a similar proof.

In the LD setting we exploit the property of Brownian motion to show the escape from saddle point, i.e. to characterize the perturbed iterate has high probability in the direction of descent,

$$(\mathbf{x}_t - \mathbf{x}_{\text{fsp}})^\intercal \nabla^2 f(\mathbf{x}_{\text{fsp}})(\mathbf{x}_t - \mathbf{x}_{\text{fsp}}) \leq -\zeta$$

Proof of Thm. 17. **Step 1:** Assume the stepsize decay parameter $\nu \in [1, 2]$ for simplicity. We show that $\Delta_i := \sum_{l=n_i}^{n_{i+1}-1} \sqrt{\eta_i}\boldsymbol{\epsilon}_i$ will lead to saddle point escape, i.e. $\Delta_i^\intercal \nabla^2 f(\mathbf{x}_{\text{fsp}})\Delta_i \leq -\zeta$. Specifically, show that Δ_i has projection on the direction of $\lambda_{\min}$ more than ζ with high probability, which exploits the property of Brownian motion and the idea that the trapping region is thin when faced with LD [HB20a].

At a fixed first-order stationary point $\mathbf{x}_{\text{fsp}}$, due to the spatial homogeneity of Brownian motion, w.l.o.g. assume that $\mathbf{e}_1$ is the unit eigenvector corresponding to the smallest eigenvalue of $\nabla^2 f(\mathbf{x}_{\text{fsp}})$. To let $\Delta_i^\intercal \nabla^2 f(\mathbf{x}_{\text{fsp}})\Delta_i \leq -\zeta$, a sufficient condition is $\lambda_{\min}(\nabla^2 f(\mathbf{x}_{\text{fsp}}))(\Delta_i)_1^2 + L(\|\Delta_i\|^2 - (\Delta_i)_1^2) \leq -\zeta$. Assume for now that $\|\Delta_i\|^2 \leq r^2$, then this condition can be phrased as $\lambda_{\min}(\nabla^2 f(\mathbf{x}_{\text{fsp}}))(\Delta_i)_1^2 + L(r^2 - (\Delta_i)_1^2) \leq -q(\Delta_i)_1^2 + L(r^2 - (\Delta_i)_1^2) \leq -\zeta$, i.e.

$$(\Delta_i)_1^2 \geq \frac{\zeta + Lr^2}{L + q} := Q \tag{2.42}$$

Now we compute the probability for (2.42) to fail within the time $T_i := \sum_{l=n_i}^{n_{i+1}-1} \sqrt{\eta_l}$ for a standard 1D Brownian motion. Define $\tau_Q = \min\{t \,|\, \big((\Delta_i)_1(t)\big)^2 \geq Q\}$. Then

$$\mathbf{Pr}((2.42) \text{ fails to hold within time } T_i) = \mathbf{Pr}(\tau_Q > T_i) \leq \frac{\mathbb{E}\,\tau_Q}{T_i} = \frac{Q}{dT_i}.$$

Here we point out that the failure probability for (2.42) is low due to the large denominator. $T_i = \sum_{l=n_i}^{n_{i+1}-1} \sqrt{\eta_l} \leq \sqrt{(n_{i+1} - n_i)\sum_{l=n_i}^{n_{i+1}-1} \eta_l} \approx \sqrt{(n_{i+1} - n_i)\delta}$. Note that $n_{i+1} - n_i = \mathcal{O}\left(n_i \exp(\delta)\right)$, then $n_i = \mathcal{O}(\exp(i\delta))$, thus

$$T_i = \exp(\mathcal{O}(i\delta)) \tag{2.43}$$

$\Big($**Remark**: consider the case $\nu = 1$ as an example for the preceding claim. As $\sum_{l=n_i}^{n_{i+1}-1} \eta_l \approx \delta$ and $n_0 = 1$, $n_i \approx \exp(i\delta)$ and $n_{i+1} = n_i \exp(\delta)$. The corresponding time in continuous domain $T_i = \sum_{l=n_i}^{n_{i+1}-1} \sqrt{\eta_l} \approx \int_{n_i}^{n_{i+1}-1} \sqrt{\eta_0}\frac{1}{\sqrt{t}}\,\mathrm{d}t = 2\sqrt{\eta_0}(\sqrt{n_{i+1}-1} - \sqrt{n_i}) \approx 2\sqrt{\eta_0}(\sqrt{n_{i+1}} - \sqrt{n_i}) = 2\sqrt{\eta_0 n_i}(\sqrt{\exp(\delta)} - 1) = 2\sqrt{\eta_0 \exp(i\delta)}(\sqrt{\exp(\delta)} - 1).\Big)$

Step 2: We show that when $\mathbf{x} \in \mathcal{U}(\mathbf{x}_{\text{fsp}}, r)$ where $\|\Delta_j\| < r$ for $j = n_i, n_i + 1, \cdots, n_{i+1} - 1$, $\|\nabla f(\mathbf{x})\| < \varepsilon$, thus the first order expansion does not contribute to function value change. Due to gradient Lipschitz, set

$$r = \max\{\frac{\varepsilon}{L}, \sqrt{\frac{3}{Lq}}\varepsilon\}.$$

While projection onto $\mathbf{e}_1$ builds up, we compute the probability that the iteration is still constrained within the ε-neighborhood of $\mathbf{x}_{\text{fsp}}$.

$$\begin{aligned}\mathbf{Pr}(\Delta_i^2 - (\Delta_i)_1^2 \leq r^2 - Q \text{ when } t \leq T_i) &= \mathbf{Pr}(\sum_{i=2}^{d} \hat{x}_i^2 \leq r^2 - Q \text{ when } t \leq T_i) \\ &= \int_0^{\sqrt{r^2-Q}} (T_i)^{-\frac{d-1}{2}} \frac{1}{\Gamma(\frac{d-2}{2})} \exp(-\frac{y^2}{2T_i}) y^{d-2} \,\mathrm{d}y \\ &\approx (T_i)^{-\frac{d-1}{2}} \frac{1}{\Gamma(\frac{d-2}{2})} \int_0^{\sqrt{r^2-Q}} y^{d-2} \,\mathrm{d}y \\ &= (T_i)^{-\frac{d-1}{2}} \frac{1}{\Gamma(\frac{d-2}{2})} (r^2 - Q)^{\frac{d-1}{2}} := P_i \end{aligned} \tag{2.44}$$

As T_i increases exponentially w.r.t. index i, P_i decreases accordingly. I.e. , within a stepsize batch, the probability for the iteration to remain bounded within the vicinity of a FSP is decreasing. Hence, the saddle point escape process can be thought of as a binomial trial with decreasing success probability, and the expected time for the iteration process to escape all saddle points is at least proportional to $\Gamma(\frac{d-2}{2})$.

Step 3: Show that the update $\mathbf{x}' = \mathbf{x} + \Delta_i$ will lead to function value decrease, thus the SGLD algorithm has to terminate, thus converging to SSP.

Denote the event $\mathcal{A}_i = \{\Delta_i^\intercal \nabla^2 f(\mathbf{x}_{\text{fsp}_i})\Delta_i \leq -\zeta \text{ and } \|\Delta_i\| \leq r\}$. From steps 1 and 2, $\mathbf{Pr}(\mathcal{A}_i) \geq (1 - \frac{Q}{dT_i})P_i$. We show that under the assumption that event $\mathcal{A}_i$ happens, function value decrease is guaranteed. Note that within a minibatch,

$$\begin{aligned}
\mathbb{E}\,\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2 &= \mathbb{E}\left\|\sum_{u=o}^{B_{\mathrm{e}}-1} \mathbf{x}_{u+1} - \mathbf{x}_u\right\|^2 = \mathbb{E}\left\|\sum_{u=o}^{B_{\mathrm{e}}-1} \eta_u\big(\nabla f_{i_u}(\mathbf{x}_{u+1}) - \nabla f_{i_u}(\widetilde{\mathbf{x}}) + \nabla f(\widetilde{\mathbf{x}})\big) - \rho_u \boldsymbol{\epsilon}_u\right\|^2 \\
&\le 2\mathbb{E}\,\|\sum_{u=o}^{B_{\mathrm{e}}-1} \eta_u\big(\nabla f_{i_u}(\mathbf{x}_{u+1}) - \nabla f_{i_u}(\widetilde{\mathbf{x}}) + \nabla f(\widetilde{\mathbf{x}})\big)\|^2 + 2\mathbb{E}\,\|\sum_{u=o}^{B_{\mathrm{e}}-1} \rho_u \boldsymbol{\epsilon}_u\|^2 \\
&= 2\mathbb{E}\,\|\sum_{u=o}^{B_{\mathrm{e}}-1} \eta_u\big(\nabla f_{i_u}(\mathbf{x}_{u+1}) - \nabla f_{i_u}(\widetilde{\mathbf{x}}) + \nabla f(\widetilde{\mathbf{x}})\big)\|^2 + 2\mathbb{E}\,\|\sum_{u=o}^{B_{\mathrm{e}}-1} \rho_u \boldsymbol{\epsilon}_u\|^2 \\
&\le 2\mathbb{E}\sum_{u=o}^{B_{\mathrm{e}}-1} \eta_u^2\left(\|\nabla f_{i_u}(\mathbf{x}_{u+1})\|^2 + \|\nabla f(\widetilde{\mathbf{x}}) - \nabla f_{i_u}(\widetilde{\mathbf{x}})\|^2\right) + 2d\sum_{u=o}^{B_{\mathrm{e}}-1} \rho_u^2 \\
&\overset{(2.25)}{\le} 2D_F \sum_{u=o}^{B_{\mathrm{e}}-1} \eta_u^2 (1 + \frac{2}{B_{\mathrm{e}}}) + 2d \sum_{u=o}^{B_{\mathrm{e}}-1} \rho_u^2
\end{aligned} \tag{2.45}$$

For function value decrease in the descent process, we have

$$\begin{aligned}
f(\mathbf{x}_t) - \mathbb{E}\,f(\mathbf{x}_{t+1}) &\ge \mathbb{E}\left[\langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}_{t+1}\rangle - \frac{L}{2}\|\mathbf{x}_t - \mathbf{x}_{t+1}\|^2\right] \\
&= \mathbb{E}\left[\langle \nabla f(\mathbf{x}_t), \eta_k \widetilde{\nabla}_k\rangle - \frac{L}{2}\|\eta_k \widetilde{\nabla}_k - \rho_k \boldsymbol{\epsilon}_k\|^2\right] \\
&= \mathbb{E}\left[\eta_t \|\nabla f(\mathbf{x}_t)\|^2 - \frac{L}{2}(\eta_t^2 \|\widetilde{\nabla}_t\|^2 + \rho_t^2 \|\boldsymbol{\epsilon}_t\|^2)\right] \\
&\ge \mathbb{E}\left[\eta_t \|\nabla f(\mathbf{x}_t)\|^2 - \frac{L}{2}\left(\eta_t^2\big(2[\|\nabla f(\mathbf{x}_t)\|^2] + 2\frac{L^2}{B_{\mathrm{e}}}[\|\mathbf{x}_t - \widetilde{\mathbf{x}}\|^2]\big) + \rho_t^2 \|\boldsymbol{\epsilon}_t\|^2\right)\right] \\
&\overset{(2.45)}{\ge} (\eta_t - \eta_t^2 L)\|\nabla f(\mathbf{x}_t)\|^2 - \underbrace{\frac{L^3}{B_{\mathrm{e}}}\eta_t^2(3D_F \sum_{u=o}^{B_{\mathrm{e}}-1} \eta_u^2 + 2d \sum_{u=o}^{B_{\mathrm{e}}-1} \rho_u^2) - \frac{L}{2}\rho_t^2 d}_{\mathcal{R}} = \mathcal{O}\left(\varepsilon^2\right)
\end{aligned}$$

Here notice that $\sum_{u=o}^{B_{\mathrm{e}}-1} \eta_u^2 = \mathcal{O}(\eta_0 \nu^{-1})$ and $\sum_{u=o}^{B_{\mathrm{e}}-1} \rho_u^2 = \mathcal{O}(\rho_0 \nu^{-1})$, and set $B_{\mathrm{e}} = \max\{\frac{L^3 D_f d}{\varepsilon^2}, 1\}$ and $\delta = \mathcal{O}(r)$ (which consequently gives the order of η_t^2), then $\mathcal{R} = \mathcal{O}(\varepsilon^2)$.

When a saddle point is encountered, within a minibatch with probability $(1-\frac{Q}{dT_i})P_i$, we have

$$
\begin{aligned}
&f(\mathbf{x}_o)-f(\mathbf{x}_o+\Delta_i)=f(\mathbf{x}_o)-f(\mathbf{x}_{\text{fsp}})+f(\mathbf{x}_{\text{fsp}})-f(\mathbf{x}_{\text{fsp}}+\Delta_i)+f(\mathbf{x}_{\text{fsp}}+\Delta_i)-f(\mathbf{x}_0+\Delta_i)\\
&=f(\mathbf{x}_{\text{fsp}})-\left(f(\mathbf{x}_{\text{fsp}})+\frac{1}{2}\Delta_i^{\intercal}\nabla^2 f(\mathbf{x}_{\text{fsp}})\Delta_i+\frac{L_2}{6}\|\Delta_i\|^3\right)+f(\mathbf{x}_o)-f(\mathbf{x}_{\text{fsp}})\\
&\qquad+f(\mathbf{x}_{\text{fsp}}+\Delta_i)-f(\mathbf{x}_0+\Delta_i)\\
&\geq f(\mathbf{x}_o)-f(\mathbf{x}_{\text{fsp}})+f(\mathbf{x}_{\text{fsp}}+\Delta_i)-f(\mathbf{x}_0+\Delta_i)-\frac{1}{2}\Delta_i^{\intercal}\nabla^2 f(\mathbf{x}_{\text{fsp}})\Delta_i+\frac{L_2}{6}\|\Delta_i\|^3\\
&\geq f(\mathbf{x}_o)-f(\mathbf{x}_{\text{fsp}})+f(\mathbf{x}_{\text{fsp}}+\Delta_i)-f(\mathbf{x}_0+\Delta_i)+\frac{\zeta}{2}-\frac{L_2}{6}r^3\\
&\geq \frac{\zeta}{2}-\frac{L_2}{6}r^3-2Lr^2=\mathcal{O}\left(\varepsilon^2\right)
\end{aligned}
$$

Set $\zeta=\frac{5\varepsilon^2}{2L}$, the time complexity to attain sufficient function value decrease before reaching a SSP is $\mathcal{O}\left(\dfrac{f(\mathbf{x}_0)-f_\star}{\varepsilon^2}\right)$.

Step 4: Now we give the description of τ_{SSP} to finish the proof. In the light of setting $\zeta=\frac{5}{2L}\varepsilon^2$, consequently $r^2-Q=\frac{\varepsilon^2}{2Lq(L+q)}$. From (2.44) together with (2.43), the probability for constrained perturbation accumulation within the stepsize batch i is given as $P_i=\mathcal{O}\left(\frac{\varepsilon^{d-1}}{\Gamma(\frac{d-2}{2})L^{d-1}q^{d-1}}\right)\cdot\frac{1}{\exp(\mathcal{O}(i\delta d))}$.

Assume the iteration sequence escapes saddle points in each stepsize batch where a saddle point is encountered, then with probability $\mathcal{O}\left(\frac{\varepsilon^{d-1}}{\Gamma(\frac{d-2}{2})L^{d-1}q^{d-1}}\right)\cdot\frac{1}{\exp(\mathcal{O}(\delta d))}$, the SGLD converges to a local minimum within time

$$\tau_{\text{SSP}}=\mathcal{O}\left(\frac{f(\mathbf{x}_0)-f_\star}{\varepsilon^2}\right)+\exp\left(\mathcal{O}(\varepsilon d)\right), \tag{2.46}$$

where the first term accounts for the step needed for sufficient function value decrease, and the second term accounts for the time needed to escape saddle points as computed in equation (2.43).

□

Chapter 3

Spectral Estimation from Simulations via Sketching

Large-scale computer simulations are a common tool in many disciplines like astrophysics, cosmology, fluid dynamics, computational chemistry, meteorology and oceanography, to name just a few. In many of these fields, a key goal of the simulation is an estimate of the power spectral density (or equivalently autocorrelation) of some dynamic or thermodynamic state variable or derived function.

Computing a full autocorrelation becomes prohibitively expensive for largescale simulations since it requires storing the entire dataset in memory. The textbook strategy to combat this problem is to subsample in time, often with clever logarithmic or multi-level spacing strategies [FS02]. Other simple solutions subsample particles or grid points, or both time and particles/points. Unfortunately, these *ad hoc* methods lack rigorous performance guarantees and can have arbitrarily large error. This article shows how to leverage results from the new field of *randomized linear algebra* to derive subsampling methods that work better in practice and have theoretical guarantees on the accuracy. These new subsampling methods, known as *sketching* methods, essentially exploit the fact that when multiplying by a multivariate Gaussian to do compression there are no worst-case inputs; in comparison, simple subsampling methods do well on some inputs but catastrophically bad on other inputs. Section 3.1 gives a toy example of this, and the rest of the paper shows how this applies to sampling data for spectral estimation.

Throughout the paper, we pay attention to computation and communication costs. In particular, the sketches are linear operators and can be applied to a data stream, so they can be applied during a simulation with negligible memory overhead and in a reasonable time. Our methods are also simple to implement. Indeed, a reason that more sophisticated sampling schemes are not used in practice may be due to the cumbersome book-keeping required for normalizations, but we review a simple trick to deal with this (Remark 32), and other than sampling, our methods do not require any "on-the-fly" computation, as the estimates are formed in post-processing.

Background Spectral estimation arises in molecular dynamic (MD) simulations based on time-dependent density functional theory (TDDFT) [RG84], which is a prominent methodology for electronic structure calculations. Depending on the original variable (position, velocity, dipole-moment, etc.), applications of spectral estimation in TDDFT include calculating vibrational or rotational modes (as used in infrared and Raman spectroscopy) [SR96], optical absorption spectra [YB96], and circular dichroism spectra [VELA$^+$09]. Many of these quantities can be experimentally measured, so one use of the spectrum is to verify that the simulation matches with reality, or to predict properties of novel materials.

Similarly, temporal autocorrelations may be computed during numerical solutions of partial differential equations (PDE). For one example, in fluid dynamics, the autocorrelations computed via direct numerical simulation of the Navier-Stokes equations can be used to validate large-eddy simulation models [RLBPS11]. Another example is oceanography where modern simulation codes rely on multi-scale numerical methods that cannot fully resolve the smallest scales, and so use stochastic models to inform the simulation [GM13, GK19]. The stochastic process can be constrained to conform to a given autocorrelation function.

MD simulations operate on particles, while standard numerical methods for PDE operate on (possibly unstructured) grids and elements. In both cases, the exact sample time-autocorrelation function can be computed provided the data (particles or grid points, at all times) is stored. Due to advances in computing power and algorithm design, it is now feasible to run extremely large simulations. A consequence of this is that many largescale simulations generate more data than can be stored. As an example, running the billion-atom Lennard Jones benchmark on the MD `LAMMPS` software [Pli95] for the equivalent of 1 ns of simulation time on argon atoms [Rap04] takes 4.9 hours on a 288 node GPU computer from 2012 [LAM12], making it a modest largescale computation. Storing the 6 coordinates of position and velocity in double precision for the 10^5 timesteps would require 4.26 PB, well beyond a typical high-end cluster disk quota of 150 TB. Longer simulations, or simulations of molecules, only exacerbate the problem. Standard compression methods for scientific data, like `fpzip` [LI06] and `ZFP` [Lin14], improve this by one or two orders of magnitude at best [SFH+18].

3.1 Sketching

Sketching is used to reduce dimensionality from N dimensions to some $m \ll N$. A family of sketches is a probability distribution on the set of real or complex $m \times N$ matrices such that if $\mathbf{\Omega}$ is drawn from this family, for any fixed vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^N$, then $\|\mathbf{\Omega}\mathbf{v} - \mathbf{\Omega}\mathbf{w}\|_2 \approx \|\mathbf{v} - \mathbf{w}\|_2$ with high probability. Hence the sketch preserves distances, and by the polarisation formula, preserves inner products as well. The core ideas behind sketching have been in place since the 1980s, and were well-known in theoretical computer science literature, but the field has expanded in the past 15 years as many applications in scientific computing were developed. In particular, sketching is often used to efficiently find solutions of large least-square regression problems [CDMI+13, Cla05, MM13, SW11, WZ13, CW12], and to determine the row and column space of large matrices for low-rank matrix decomposition [HMT11, DMM08, MD09].

Formally, a probability distribution on $m \times N$ matrices is a *Johnson-Lindenstrauss Transform* with parameters ε, δ and d if for any fixed set of d vectors $\{\mathbf{v}_i\}_{i=1}^d \subset \mathbb{R}^N$, if $\mathbf{\Omega}$ is drawn from this distribution, then with probability at least $1 - \delta$ it holds that

$$(1-\varepsilon)\|\mathbf{v}_i - \mathbf{v}_j\|_2^2 \leq \|\mathbf{\Omega}\mathbf{v}_i - \mathbf{\Omega}\mathbf{v}_j\|_2^2 \leq (1+\varepsilon)\|\mathbf{v}_i - \mathbf{v}_j\|_2^2$$

for all $i, j \in \{1, \ldots, d\}$. When no confusion arises, it is common to not distinguish between the random variable and the distribution, and write $\mathbf{\Omega} \in \mathrm{JLT}(\varepsilon, \delta, d)$ to encode the notion. The name Johnson-Lindenstrauss Transform honours Johnson and Lindenstrauss' well-known result which shows that such distributions exist for $m = \mathcal{O}(\epsilon^{-2}\log(d))$ [JL84].

Intuition The classic example of a sketch is an appropriately scaled Gaussian matrix with independent entries. To gain insight, consider the case when $\mathbf{\Omega} \in \mathbb{R}^{1\times N}$ is a sketch that compresses $\mathbf{v} \in \mathbb{R}^N$ to a single number, and without loss of generality, let $\|\mathbf{v}\|_2 = 1$. All sketches we consider will be unbiased, meaning $\mathbb{E}\,\mathbf{\Omega}^T\mathbf{\Omega} = I_{N\times N}$ where I is the identity matrix. We wish to preserve norm, so we look at $\|\mathbf{\Omega}\mathbf{v}\|_2^2$, or equivalently $(\mathbf{\Omega}\mathbf{v})^2$ when $m = 1$. Then any unbiased sketch has $\mathbb{E}\,(\mathbf{\Omega}\mathbf{v})^2 = 1$.

Simple subsampling can be written as a sketch by defining $\mathbf{\Omega} = \sqrt{N}\mathbf{e}_i^\intercal$ where $\mathbf{e}_i$ is the i^{th} canonical basis vector in $\mathbb{R}^N$, and i is chosen uniformly from $\{1, \ldots, N\}$; one can easily show this is unbiased. If the input $\mathbf{v}$ has weight evenly distributed over all coordinates, such that $|v_j| = N^{-1/2}$ for all $j = 1, \ldots, N$, then this is a good sketch, since the variance is $\mathbb{V}\text{ar}((\mathbf{\Omega}\mathbf{v})^2) = 0$. However, if the input is $\mathbf{v} = \mathbf{e}_k$ for any fixed k, then an elementary calculation shows that $\mathbb{V}\text{ar}((\mathbf{\Omega}\mathbf{v})^2) = N - 1$, which in high dimensions is too large to be useful.

In contrast, if we define the sketch $\mathbf{\Omega}$ as $1 \times N$ independent standard normal random variables, then $\mathbf{\Omega}$ is also an unbiased sketch, and furthermore $\mathbb{V}\text{ar}((\mathbf{\Omega}\mathbf{v})^2) = 2$ independent of the fixed vector $\mathbf{v}$. The Gaussian sketch is not always more efficient than the subsampling sketch, but it is never much worse, and sometimes it is better by a factor of N.

Types of sketches In this work we consider the following three types of distributions of sketching matrices $\mathbf{\Omega}$ (Matlab code available via [Bec19]; some Python implementations are part of the `random_projection` module of scikit learn):

Gaussian sketch Each entry of $\mathbf{\Omega}$ is independently drawn from the scaled normal distribution $\mathcal{N}(0, \frac{1}{m})$.

Haar sketch Draw $\widetilde{\mathbf{\Omega}}$ as in the Gaussian case and then define the rows of $\mathbf{\Omega}$ to be the output of Gram-Schmidt orthogonalisation applied to the rows of $\widetilde{\mathbf{\Omega}}$, scaled by $\sqrt{\frac{N}{m}}$. This is equivalent to sampling the first m columns of a matrix from the Haar distribution on orthogonal matrices, and can also be computed via the `QR` factorisation algorithm with post-processing [Mez07]. This is essentially the case originally considered by Johnson and

Lindenstrauss.

FJLT The Fast Johnson-Lindenstrauss Transformation (FJLT) as is usually implemented [Woo14] is a structured matrix of the form $\mathbf{\Omega} = \sqrt{\frac{N}{m}}\mathbf{P}^\intercal\mathbf{H}\mathbf{D}$ where $\mathbf{D}$ is a diagonal matrix with Rademacher random variables on the diagonal (i.e., independent, ± 1 with equal probablity), $\mathbf{H}$ is a unitary or orthogonal matrix, and $\mathbf{P}^\intercal$ a simple subsampling matrix such that $\mathbf{P}^\intercal\mathbf{v}$ chooses m of the coordinates from $\mathbf{v}$ uniformly at random (with replacement), so that $\mathbf{P}$ consists of m canonical basis vectors. To be useful, each entry of $\mathbf{H}$ should be as small as possible ($\approx 1/\sqrt{N}$), and $\mathbf{H}$ should be computationally fast to apply to vector. Standard choices for $\mathbf{H}$ are the (Walsh-)Hadamard, discrete Fourier, and discrete Cosine transforms, all of which have fast implementations that take $\mathcal{O}(N\log N)$ flops to apply to a vector. Since applying $\mathbf{D}$ and $\mathbf{P}^\intercal$ take linear and sub-linear time, respectively, the cost of computing $\mathbf{\Omega}\mathbf{v}$ is $\mathcal{O}(N\log N)$, better than the $\mathcal{O}(Nm)$ cost of the Gaussian and Haar sketches. The original FJLT proposed in [AC09] is a slight variant that uses a different sparse matrix $\mathbf{P}$.

There are other types of sketches such as the count-sketch [Cor11], leverage-score based sketches [Mah11], and entry-wise sampling [AM07, AKL13] which can be combined with preconditioning [PAB17]. Some of these sketches are not Johnson-Lindenstrauss transforms but are instead the related notion of subspace embeddings. See [Woo14, Mah11, MT20] for surveys on sketching literature.

Guarantees

Table 3.1 summarizes the required compressed dimension size m for the corresponding sketching matrix to be a JLT(ε, δ, d).

Method	Compressed dimension m
Gaussian [Woo14]	$\mathcal{O}(\varepsilon^{-2}\log(d/\delta))$
Haar [Ver18]	$\mathcal{O}(\varepsilon^{-2}\log(d/\delta))$
FJLT	$\mathcal{O}\Big(\varepsilon^{-2}\log\big(\frac{d}{\delta - N^{-\log^3(N)}}\big)\log^4(N)\Big)$

Table 3.1: Compressed dimension requirement for JLTs.

The result for the FJLT, which holds when $\mathbf{H}$ is a Hadamard, discrete Fourier or discrete Cosine transform, is not explicitly in the literature but follows by combining [KW11, Thm. 3.1] with [FR13, Thm. 12.31]. The constants hidden in the asymptotic notation are not bad. For example, for the Gaussian sketch, with $d = 10^3$ points (in arbitrary dimension), for failure probability $\delta \leq 0.1$ and error $\varepsilon \leq 1/3$, the number of samples required is $m \geq 535$.

3.2 Approximating autocorrelation with sketching

Throughout the article, we think of the data as a signal $x(t, \varphi)$ in time t and space φ, where φ can encode a grid location or a particle number depending on the type of simulation (for space indices in dimension greater than one, we flatten the indices into a large one-dimensional list). Let t have unit spacing $\Delta T = 1$, $t \in \{1, 2, \dots, T\}$, and let space be indexed by $\{\varphi_1, \dots, \varphi_N\}$. We organize the data into a matrix $\mathbf{X} \in \mathbb{R}^{T \times N}$.

In what follows, we consider classical methods for estimating the autocorrelation. There are powerful alternative methods, based on parametric models — most notably, autoregressive-moving-average (ARMA) models [Bro06]. However, these methods excel when T is small, and do not clearly extend to $N > 1$, and are not natively suited to on-the-fly calculations during a simulation as they require significant post-processing and parameter tuning.

Autocorrelation and the Wiener-Khinchin Theorem

For a continuous signal x, the time autocorrelation function of lag τ of signal x is

$$R(\tau) = \mathbb{E}_{\varphi} \lim_{T \to \infty} \frac{1}{2T} \int_{-T}^{T} x(t, \varphi) x(t + \tau, \varphi) \, \mathrm{d}t.$$

For the corresponding discretized signal of length T, the (sample) time autocorrelation of lag τ is defined as

$$\widehat{R}_\tau[\mathbf{X}] = \frac{1}{N}\frac{1}{T-\tau}\sum_{t=1}^{T-\tau}\sum_{i=1}^{N} x(t,\varphi_i)x(t+\tau,\varphi_i). \tag{3.1}$$

As our goal will be to approximate the sample autocorrelation $\widehat{R}_\tau$, we drop the $\widehat{\ }$ notation for clarity and simply write R_τ.

Remark 27 (Cross-terms)**.** *Calculating Eq. 3.1 requires storing $N \times T$ parameters. If one instead computed $\sum_{t=1}^{T-\tau}\left(\sum_{i=1}^{N} x(t,\varphi_i)\right)\left(\sum_{i=1}^{N} x(t+\tau,\varphi_i)\right)$ (with appropriate normalization), then only $\mathcal{O}(T)$ storage is required, but unfortunately this is not equivalent to Eq. 3.1 due to the presence of the cross-terms. One way to view sketching methods is that the sketching adds in suitable randomness so that when using the $\mathcal{O}(T)$ formula, the cross-terms vanish in expectation.*

Letting the shifted, unnormalized (sample) covariance matrix be $\mathbf{\Sigma} = \mathbf{X}\mathbf{X}^\intercal$, our first observation is that R_τ is a linear function of $\mathbf{\Sigma}$, since

$$(\mathbf{\Sigma})_{t,t'} = \sum_{i=1}^{N} x(t,\varphi_i)x(t',\varphi_i)$$

so R_τ is the scaled sum of the τ^{th} diagonal of $\mathbf{\Sigma}$, and hence we use the notation $R_\tau[\mathbf{\Sigma}]$, and also write $\mathbf{R}[\mathbf{\Sigma}] = (R_0[\mathbf{\Sigma}], R_1[\mathbf{\Sigma}], \cdots, R_{T-1}[\mathbf{\Sigma}])^\intercal$ when working with all T possible lags.

The time autocorrelation is often of interest itself, but it can also be used to derive the power spectral density,

$$S(\omega) = \lim_{T\to\infty} \mathbb{E}_\varphi \left|\frac{1}{\sqrt{2T}}\int_{-T}^{T} x(t,\varphi)\mathrm{e}^{-\mathrm{i}\omega t}\,\mathrm{d}t\right|^2.$$

If x is a wide-sense stationary random process, under certain conditions, the Wiener-Khinchin Theorem states that the spectral density is the Fourier transform of $R(\tau)$, and the discrete power spectral density can be estimated by the discrete Fourier transform of $\mathbf{R}$.

Thus both autocorrelation and power spectrum can be reduced to the problem of finding an accurate estimate of $\mathbf{\Sigma}$. Note that $\mathbf{\Sigma}$ is a $T \times T$ matrix and is impractical to store, and is used only for analysis. Our actual software implementation only needs a factored form $\mathbf{\Sigma} = \widehat{\mathbf{X}}\widehat{\mathbf{X}}^\intercal$ for

$\widehat{\mathbf{X}} \in \mathbb{R}^{T\times m}$, and works directly with $\widehat{\mathbf{X}}$. Furthermore, due to linearity, implementations can exploit existing autocorrelation software (which typically use the fast Fourier transform to do convolutions efficiently). Specifically, if the columns of $\widehat{\mathbf{X}}$ are $\mathbf{v}_1, \ldots, \mathbf{v}_m$, then $R_\tau[\mathbf{\Sigma}] = R_\tau[\sum_{i=1}^m \mathbf{v}_i\mathbf{v}_i^\intercal] = \sum_{i=1}^m R_\tau[\mathbf{v}_i\mathbf{v}_i^\intercal]$ and $R_\tau[\mathbf{v}_i\mathbf{v}_i^\intercal]$ is performed implicitly via an efficient autocorrelation implementation.

In the next section, we will use standard results from the sketching literature to create an estimator $\widehat{\mathbf{\Sigma}}$ and bound $\|\mathbf{\Sigma} - \widehat{\mathbf{\Sigma}}\|_F < \varepsilon$, where $\|\cdot\|_F$ denotes the Fröbenius (Hilbert-Schmidt) norm. To use those results, we first show that $\mathbf{R}$ is Lipschitz continuous so that a small ε implies an accurate autocorrelation (and hence an accurate power spectrum).

Lemma 28. *Let $\mathbf{\Sigma}$ and $\widehat{\mathbf{\Sigma}}$ both be symmetric $T\times T$ matrices. Then*

$$\|\mathbf{R}[\widehat{\mathbf{\Sigma}}] - \mathbf{R}[\mathbf{\Sigma}]\|_1 \leq \frac{\sqrt{1+\log T}}{N}\|\mathbf{\Sigma} - \widehat{\mathbf{\Sigma}}\|_F \tag{3.2}$$

$$\|\mathbf{R}[\widehat{\mathbf{\Sigma}}] - \mathbf{R}[\mathbf{\Sigma}]\|_\infty \leq \frac{1}{N}\|\mathbf{\Sigma} - \widehat{\mathbf{\Sigma}}\|_F \tag{3.3}$$

where $\|\mathbf{R}[\widehat{\mathbf{\Sigma}}] - \mathbf{R}[\mathbf{\Sigma}]\|_1 = \sum_{\tau=0}^{T-1} \left|R_\tau[\mathbf{\Sigma}] - R_\tau[\widehat{\mathbf{\Sigma}}]\right|$*, and* $\|\mathbf{R}[\widehat{\mathbf{\Sigma}}] - \mathbf{R}[\mathbf{\Sigma}]\|_\infty = \max_{\tau=0,\ldots,T-1}\left|R_\tau[\mathbf{\Sigma}] - R_\tau[\widehat{\mathbf{\Sigma}}]\right|$.

Proof. Define the difference between true covariance matrix and the estimate as $\mathbf{\Delta} = \mathbf{\Sigma} - \widehat{\mathbf{\Sigma}}$. For the ∞-norm case, using linearity of $\mathbf{R}$,

$$\begin{aligned} \|\mathbf{R}[\mathbf{\Delta}]\|_\infty &= \max_\tau \|R_\tau[\mathbf{\Delta}]\| \\ &= \frac{1}{N}\max_\tau \left|\frac{1}{T-\tau}\sum_{t=1}^{T-\tau}\Delta_{t,t+\tau}\right| \\ &\leq \frac{1}{N}\max_{t,t'}|\Delta_{t,t'}| \leq \frac{1}{N}\|\mathbf{\Delta}\|_F \end{aligned}$$

From this, we immediately have the bound $\|\mathbf{R}[\mathbf{\Delta}]\|_1 \leq \frac{T}{N}\|\mathbf{\Delta}\|_F$, but this is loose, and we show below how to derive a better dependence on T:

$$
\begin{aligned}
\left\|\mathbf{R}[\mathbf{\Sigma}] - \mathbf{R}[\widehat{\mathbf{\Sigma}}]\right\|_1 &= \sum_{\tau=0}^{T-1} \left|R_\tau[\Delta]\right| \\
&\leq \frac{1}{N} \sum_{\tau=0}^{T-1} \frac{1}{T-\tau} \sum_{t=1}^{T-\tau} |\Delta_{t,t+\tau}| \\
&\overset{\textcircled{1}}{\leq} \frac{1}{N} \sum_{\tau=0}^{T-1} \sqrt{\frac{1}{T-\tau} \sum_{t=1}^{T-\tau} |\Delta_{t,t+\tau}|^2} \\
&\overset{\textcircled{2}}{\leq} \frac{1}{N} \sqrt{\sum_{\tau=0}^{T-1} \frac{1}{T-\tau}} \sqrt{\sum_{\tau=0}^{T-1} \sum_{t=1}^{T-\tau} |\Delta_{t,t+\tau}|^2} \\
&= \frac{1}{N} \sqrt{\sum_{\tau=1}^{T} \frac{1}{\tau}} \sqrt{\|\Delta\|_F^2 - \sum_{\substack{\alpha\in\text{lower triang.}\\ \text{off-diag elems}}} \Delta_\alpha^2} \\
&\leq \frac{\sqrt{1+\log T}}{N} \|\Delta\|_F
\end{aligned}
\tag{3.4}
$$

where ① is due to Jensen's inequality, and ② is due to Cauchy-Schwarz. □

3.3 Theoretical guarantees

We give bounds on the error of autocorrelation evaluation due to sketching the rows of $\mathbf{X}$, i.e., $\widehat{\mathbf{X}}^\intercal = \mathbf{\Omega}\mathbf{X}^\intercal$. Each row consists of the data at a given time t, so this can be trivially implemented in a streaming fashion. The overall compression ratio is $\gamma = \frac{m}{N}$, independent of T.

Theorem 29. *For any $\varepsilon > 0$, and for a data matrix $\mathbf{X} \in \mathbb{R}^{T\times N}$, compute $\widehat{\mathbf{X}} = \mathbf{X}\mathbf{\Omega}^\intercal \in \mathbb{R}^{T\times m}$ for $\mathbf{\Omega} \in JLT(\varepsilon, \delta, 2T)$, and define $\mathbf{\Sigma} = \mathbf{X}\mathbf{X}^\intercal$ and $\widehat{\mathbf{\Sigma}} = \widehat{\mathbf{X}}\widehat{\mathbf{X}}^\intercal$. Then with probability at least $1-\delta$, the computed autocorrelation based solely on the data sketch satisfies the following error characterisations:*

$$
\frac{\|\mathbf{R}[\widehat{\mathbf{\Sigma}}] - \mathbf{R}[\mathbf{\Sigma}]\|_1}{\|\mathbf{X}\|_F^2} \leq \frac{\sqrt{1+\log T}}{N}\varepsilon \tag{3.5}
$$

$$
\frac{\|\mathbf{R}[\widehat{\mathbf{\Sigma}}] - \mathbf{R}[\mathbf{\Sigma}]\|_\infty}{\|\mathbf{X}\|_F^2} \leq \frac{1}{N}\varepsilon. \tag{3.6}
$$

In particular, if $\mathbf{\Omega}$ is a Gaussian, Haar or FJLT sketch, then $\mathbf{\Omega} \in JLT(\epsilon, \delta, 2T)$ if m is chosen as in Table 3.1.

Proof. A standard sketching result due to Sarlós [Sar06] gives the error bound for using JLT to estimate matrix products as the following: let $\mathbf{X} \in \mathbb{R}^{T_1 \times N}$ and $\mathbf{Y} \in \mathbb{R}^{N \times T_2}$. If $\mathbf{\Omega}$ is a JLT$(\varepsilon, \delta, T_1 + T_2)$, then

$$\mathbb{P}(\|\mathbf{XY} - \mathbf{X\Omega}^\intercal\mathbf{\Omega Y}\|_F \leq \varepsilon \|\mathbf{X}\|_F \|\mathbf{Y}\|_F) \geq 1 - \delta$$

Applying Lemma 28 with $\mathbf{Y} = \mathbf{X}$ gives the result immediately. □

To quantitatively characterize how the error in autocorrelation evaluation depends on the compression ratio, we have the following corollary which follows immediately using the theorem and Table 3.1.

Corollary 30. *Under the setting of Theorem 29, assuming the data matrix* $\mathbf{X}$ *has bounded entries, then the required compression ratio* $\gamma = m/N$ *to have* $\|\mathbf{R}[\widehat{\mathbf{\Sigma}}] - \mathbf{R}[\mathbf{\Sigma}]\|_1 \leq \varepsilon$ *with probability greater than* $1 - \delta$ *is* $\gamma = \mathcal{O}\big(\frac{T^2 \log T \log(2T/\delta)}{\varepsilon^2 N}\big)$ *for Gaussian or Haar matrix sketches, and* $\gamma = \mathcal{O}\big(\frac{T^2 \log T \log(2T/(\delta - \mathrm{e}^{-\log^4 N}))}{\varepsilon^2 N}\big)$ *for FJLT sketches.*

The corollary suggests that as the simulation time $T \to \infty$, our compression ratio grows, until at some point it is not useful. However, T should be seen as inversely proportional to the lowest desired frequency in the power spectrum, not total simulation time. For longer simulation times T_{long}, the data should be blocked into B matrices $\mathbf{X}_{(1)}, \ldots, \mathbf{X}_{(B)}$, each of size $T = T_{\text{long}}/B$, and then form $\mathbf{\Sigma} = \frac{1}{B}\sum_{b=1}^{B} \mathbf{X}_{(b)}\mathbf{X}_{(b)}^\intercal$, and similarly for $\widehat{\mathbf{\Sigma}}$, with fresh sketches $\mathbf{\Omega}_{(b)}$ drawn for each block. If for some reason one needed arbitrarily low frequencies, and wanted the sample time autocorrelation to converge to the true time autocorrelation, then choose $B \propto \sqrt{T_{\text{long}}}$ [BD87, PM06], but otherwise choose $B \propto T_{\text{long}}$ and hence the block size T is constant.

Thus given a fixed time T, the corollary says that $\gamma \approx \mathcal{O}(1/N)$ and hence as the amount of data increases, the compression savings are great; in fact, the absolute number of measurements m is independent of the spatial size N. For example, this means that if one increases the resolution of a grid or mesh, the amount of data needed to be stored actually stays constant. This holds not just for 1D grids, but 3D or any dimension grids.

We also note that the matrix $\mathbf{\Sigma}$ need not represent all grid points or particles, but could instead represent a subset of grid points or particles, and then the calculations are done independently for each $\mathbf{\Sigma}$ and averaged in the end. This may be beneficial in parallel and distributed computing, where each $\mathbf{\Sigma}$ might represent just the spatial locations stored in local memory.

3.4 Numerical experiments

The pseudo-code for the proposed sketching algorithm is in Algo. 3. It exploits existing fast implementations of sample autocorrelation, e.g., `xcorr` in Matlab or `numpy.correlate` in Python. We use Matlab indexing notation, with $\mathbf{X}(:,j)$ meaning the j^{th} column of $\mathbf{X}$, and $\mathbf{X}(i,:)$ the i^{th} row. For our data, the mean was near zero and was not subtracted explicitly. Bartlett windowing [PM06] was performed to reduce spectral leakage whenever $B > 1$.

Algorithm 3 Sketching for autocorrelation and power density estimation. Requires existing implementation of `autocorr`.

```
Require: Simulation time $T_{\text{long}}$, number of blocks $B$, compression size $m$
$T = T_{\text{long}}/B$
for $b = 0, 1, 2, \ldots, B-1$ do
    Draw $\mathbf{\Omega} \in \mathbb{R}^{m \times N}$                ▷ One of the sketches from §3.1
    Initialize empty array $\widehat{\mathbf{X}} \in \mathbb{R}^{T \times m}$
    for $t = 1, 2, \ldots, T$ do
        Generate data $\mathbf{x}^\intercal \in \mathbb{R}^{1 \times N}$ according to simulation (at time $t + bB$); equivalent to *row* $\mathbf{X}(t,:)$
        Compute and store row $\widehat{\mathbf{X}}(t,:) = (\mathbf{\Omega}\mathbf{x})^\intercal$
        Discard $\mathbf{x}$ from memory
    end for
    Compute $\mathbf{R}_{(b)} = \frac{1}{N}\sum_{i=1}^{m} \texttt{autocorr}(\widehat{\mathbf{X}}(:,i))$
end for
$\mathbf{R} = \frac{1}{B}\sum_{b=0}^{B-1} \mathbf{R}_{(b)}$                ▷ autocorrelation
$S = \texttt{FFT}(\mathbf{R})$                ▷ power spectral density
```

Remark 31. *Conceptually, the algorithm forms* $\widehat{\mathbf{X}} = \mathbf{X}\mathbf{\Omega}$*, though the full-size data matrix* $\mathbf{X}$ *is never actually formed, as* $\widehat{\mathbf{X}}$ *is built up row-by-row (and old rows of* $\mathbf{X}$ *are discarded). Similarly, the estimated covariance matrix* $\widehat{\mathbf{\Sigma}}$*, which is introduced for discussion on theoretical properties of sketching methods, is never explicitly constructed for computation, as discussed in Section 3.2.*

3.4.1 Baseline methods

Many existing algorithms for computing autocorrelation require complete data, such as the utility routines provided with the popular MD simulator `LAMMPS` [Pli95], so we do not compare with these since they work with the full data. Among subsampling approaches, we will compare with the following three types of subsampling (recall the data matrix is structured as $\mathbf{X} \in \mathbb{R}^{T \times N}$, where T is the total length of time and N is the total number of particles or grid size), all of which sample with replacement:

Time dimension compression Given a compression ratio γ, sample time points $\mathcal{I} \subset \{1, \ldots, T\}$ with size $|\mathcal{I}| = \lceil \gamma T \rceil$ (where $\lceil a \rceil$ rounds a up to the nearest integer) by selecting **rows** from the data matrix $\mathbf{X}$. The natural unbiased estimator for the autocorrelation $R_\tau[\mathbf{X}]$ is

$$\frac{1}{N} \frac{1}{z_\tau^{\mathcal{I}}} \sum_{t | t, t+\tau \in \mathcal{I}} \sum_{i=1}^{N} \mathbf{X}(t, i) \mathbf{X}(t+\tau, i) \tag{3.7}$$

where $z_\tau^{\mathcal{I}}$ is a normalization coefficient that is the number of t such that $t \in \mathcal{I}$ and $t + \tau \in \mathcal{I}$ (for full sampling, this is $z_\tau^{\mathcal{I}} = T - \tau$ as in (3.1)). Efficient computation of this autocorrelation estimate is discussed in Remark 32. When the index $\mathcal{I}$ is sufficiently small, not all lags τ will have an estimate, thus making computation of the PSD unclear. In these cases, we interpolate the missing lag values using cubic splines.

There are several common choices for $\mathcal{I}$:

(1) Choosing $\mathcal{I}$ (pseudo-)randomly according to the uniform distribution. This is the method we use in the experiments unless otherwise noted, as it has the best performance among these types of methods.

(2) Choosing $\mathcal{I}$ via a power-series sampling scheme that is common in simulation of polar liquids (where $R_\tau[\mathbf{X}]$ is only needed for short lags τ due to the rapid decorrelation). Given a block length k, let $\mathcal{I}_0 = \{1, 2, 4, 8, \ldots, 2^k\}$, and then the index set $\mathcal{I}$ is divided into blocks $\mathcal{I} = \mathcal{I}_0 \cup \left(2^k + \mathcal{I}_0\right) \cup \left(2^{k+1} + \mathcal{I}_0\right) \cup \ldots$. This scheme is intended to give dense sampling for low lags, and some sampling for higher lags while still allowing for

reasonable book-keeping due to its structured nature. See Fig. 3.1 for a comparison of this scheme with random sampling; it generally underperforms random sampling, so we do not present further comparisons.

(3) Sparse ruler sampling. As shown in Fig. 3.1, the power-series scheme does not generate all possible lags. Sampling schemes that do generate all possible lags (up to some point) are known as *rulers*, and rulers with only a few samples are *sparse rulers*, and are used in signal processing [RL13]. One can modify the power-series scheme so that each block $\mathcal{I}_0$ is a sparse ruler (we used Wichmann Rulers). The scheme still underperforms random sampling; see supplementary information 1.A for more details.

(4) Sampling blocks (Algorithm 8 in [FS02]), which gives good estimates of $R_\tau[\mathbf{X}]$ for small τ, but does not attempt to estimate $R_\tau[\mathbf{X}]$ for τ larger than the block size. This does not perform well and details in left for the supplementary information section 1.A.

(5) Hierarchical sampling schemes (Algorithm 9 in [FS02]), designed to improve on block sampling by giving a small amount of large lag information. This method is exact for some derived quantities (like diffusion coefficients) but *ad-hoc* for estimating the large-lag autocorrelation. This method has high errors (see supplementary information 1.A for details).

These last two methods (4 and 5) are different than all the other baseline methods we discuss as they require “on-the-fly” computation to record the estimate of $R_\tau[\mathbf{X}]$ for a subset of the lags τ, and this estimate is then updated. These methods do not simply sample $\mathbf{X}$ and then postprocess. Both method 4 and 5 do not give accurate estimates for large lags, hence we do not present further simulation results with these methods.

Particle dimension compression Given a compression ratio γ, randomly sample particles (or grid points) to form $\mathcal{I} \subset \{1, \ldots, N\}$ with size $|\mathcal{I}| = \lceil \gamma N \rceil$ by uniformly selecting **columns**

from the data matrix $\mathbf{X}$. The natural unbiased estimator of $R_\tau[\mathbf{X}]$ is then

$$\frac{1}{|\mathcal{I}|}\frac{1}{T-\tau}\sum_{t=1}^{T-\tau}\sum_{i\in\mathcal{I}}\mathbf{X}(t,i)\mathbf{X}(t+\tau,i).$$

Naïve uniform sparsification (both time and particles) Given a compression ratio γ, uniformly sample $\lceil \gamma TN \rceil$ **entries** from $\mathbf{X}$. This approach has the same estimator for autocorrelation of lag τ as the case time dimension compression, except that the sampling set $\mathcal{I}$ and normalization constant now depend on the column i. We refer to this as "naïve" since it uses a uniform distribution, in contrast to complicated weighted sampling schemes like [AKL13] used in the sampling literature. With an appropriate normalization $z^{\mathcal{I}}_{\tau,i}$, the unbiased estimate of $R_\tau[\mathbf{X}]$ is

$$\frac{1}{z^{\mathcal{I}}_{\tau,i}}\sum_{i=1}^{N}\sum_{\substack{t,\text{ such that}\\(t,i),(t+\tau,i)\in\mathcal{I}}}\mathbf{X}(t,i)\mathbf{X}(t+\tau,i).$$

which can be calculated via the above formula or via Remark 32.

One can combine time dimension and particle dimension compression (doing time-then-particle, or particle-then-time), but for a given overall compression level, we did not find that this improved accuracy, and therefore do not include it in the results.

Remark 32. *To efficiently compute the estimate of the autocorrelation for any time dimension compression scheme, i.e., Eq. 3.7, one can use existing fast autocorrelation functions. Specifically, set the non-sampled entries to zero, so they do not contribute to the sum, and put each column of* $\mathbf{X}$ *through a standard autocorrelation function and then average the results. To find the normalization factor* $z^{\mathcal{I}}_{\tau}$*, one can create an indicator vector* $\boldsymbol{\xi}$ *where* $\xi_t = 1$ *if* $t \in \mathcal{I}$ *and* $\xi_t = 0$ *if* $t \notin \mathcal{I}$ *(think of this as a "book-keeping" particle that can be stored as an extra particle or grid-point), and then compute the autocorrelation of* $\boldsymbol{\xi}$ *to get the normalization* $z^{\mathcal{I}}_{\tau}$*. Computing the value by hand is possible but tedious and the programming is error-prone, which may be a reason why simple (non-random) time compression schemes have historically been favoured.*

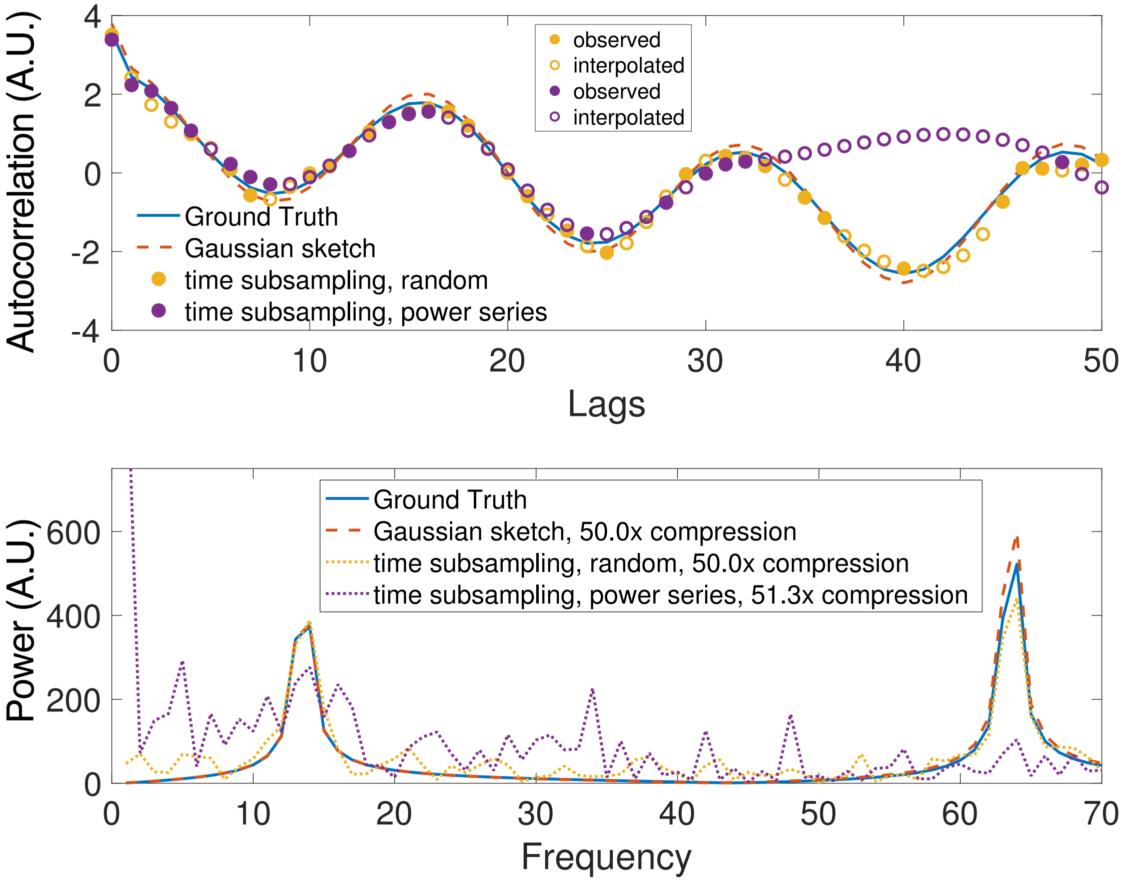


Figure 3.1: Autocorrelation (top) and power spectral density (bottom) for the two frequency simulation.

To illustrate the different types of time dimension compression schemes, we conduct a basic experiment of $N = 10^4$ particles and $T = 2000$ time points with unit spacing, where each particle is randomly assigned one of two possible frequencies (one fast, one slow), and with a random phase; the autocorrelation is the fast sinusoid modulated by the slow sinusoid. The power spectral density ranges up to 500 Hz, of which the first 70 Hz are shown in the bottom of Fig. 3.1. The ground truth would show two delta functions if $T = \infty$ but are spectrally broadened by the finite time sample. Fig. 3.1 shows that, at 50× compression, the time sampling approaches have no observations for some lags and must be interpolated. The random time subsampling is more accurate than the power series approach. The Gaussian sketching method is significantly more accurate than both time compression methods.

3.4.2 Methanol ensemble simulation data

Our dataset is a MD simulation using the `LAMMPS` software [Pli95] for $N = 384$ methanol molecules with time step 1 fs for 10 ps, with potentials between pairs of bonded atoms, between triplets and between quadruplets of atoms set as harmonic, and potential for pairwise interactions set as the hybrid of the “DREIDING” hydrogen bonding Lennard-Jones potential and the Lennard-Jones with cut-off Coulombic potential [PAM$^+$11]. The quantity of interest is the power spectral density of the velocity of the molecules. Except in Fig. 3.5, no blocking was performed, so $B = 1$ and $T = T_{\text{long}} = 10000$. The true sample autocorrelation, up to $\tau = 100$, is shown in Figure 3.2. The actual simulation was run for 20000 time steps (20 ps) but the first 10 ps are ignored as the simulation was equilibrating.

Figure 3.3 shows the corresponding true power spectral density (PSD), as well as the PSD computed via the three proposed sketching methods (with Gaussian, Haar and FJLT sketches), as well as the three benchmark methods, using only about 1% of the data. The three sketching methods faithfully recover the true peaks of the spectrum, while the baseline methods (in blue) either have spurious peaks (time compression and naive uniform compression) or miss/distort peaks (particle compression).

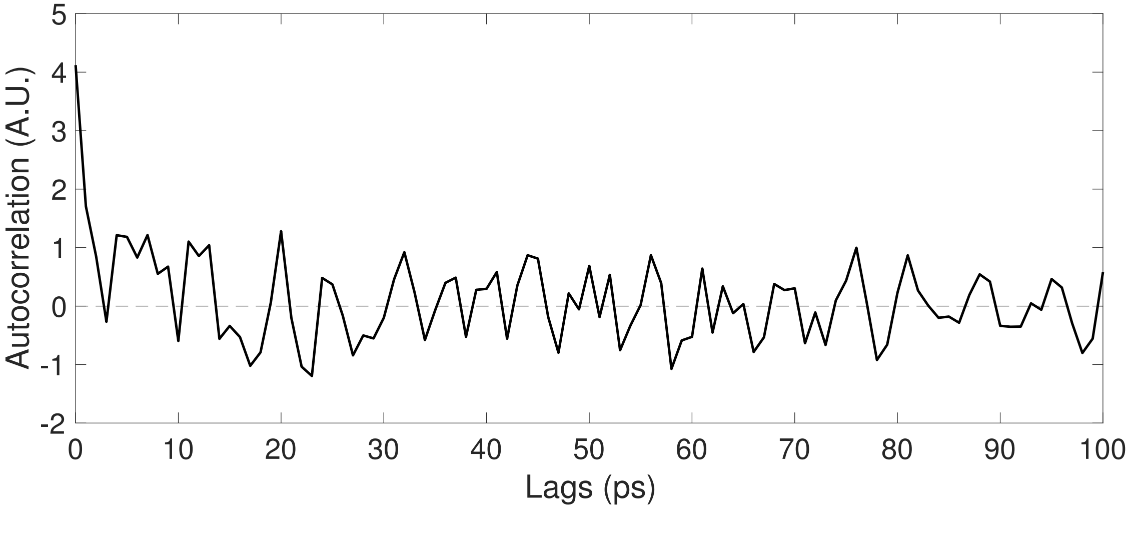


Figure 3.2: Ground truth of autocorrelation of the velocity of methanol molecules up to $\tau = 100$.

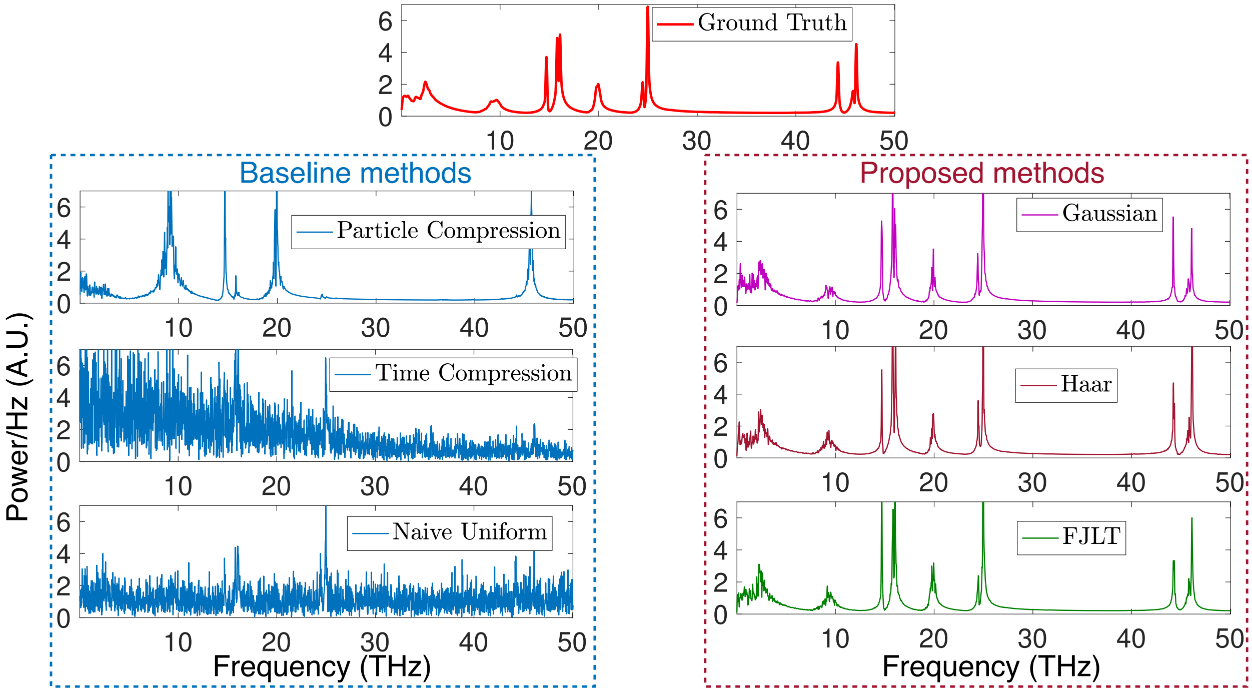


Figure 3.3: Power spectral density for methanol data. The compression ratio is 1% for each method.

For systematic and quantitative comparison, we consider three metrics for evaluating the estimated PSD $\hat{\mathbf{s}} = \widehat{S}(\omega)$ compared to the true PSD $\mathbf{s} = S(\omega)$. First, we use the relative ℓ_2 norm $\|\hat{\mathbf{s}} - \mathbf{s}\|_2/\|\mathbf{s}\|_2$ which also captures the relative ℓ_2 error for the autocorrelation (since the Fourier transform is unitary, i.e., Parseval's identity). Second, we use the relative ℓ_∞ error, which is defined as $\max_{i,s_i\neq 0} \frac{|\hat{s}_i - s_i|}{|s_i|}$. Third, we use a relative ℓ_1 error, defined as $\|\hat{\mathbf{s}} - \mathbf{s}\|_1/\|\mathbf{s}\|_1$, where $\|\mathbf{s}\|_1 = \sum_i |s_i|$.

When computing the compression ratio, a sketching method with $\mathbf{\Omega} \in \mathbb{R}^{m\times N}$ achieves a $\gamma = m/N$ compression ratio, as no meta-data needs to be stored. The time dimension and particle dimension subsampling methods must also save the time or particle/space indices $\mathcal{I}$ as meta-data, though this is typically insignificant, so they achieve approximately $|\mathcal{I}|/T$ and $|\mathcal{I}|/N$ compression ratios, respectively. The naïve uniform sparsification, which samples in both space and time, must save both time and particle/space indices; this is done implicitly by storing the data as a sparse matrix in compressed sparse column format. The overhead of storing these indices can be significant, which is why the compression ratio for "naïve uniform" is slightly worse than the target of $|\mathcal{I}|/(TN)$.

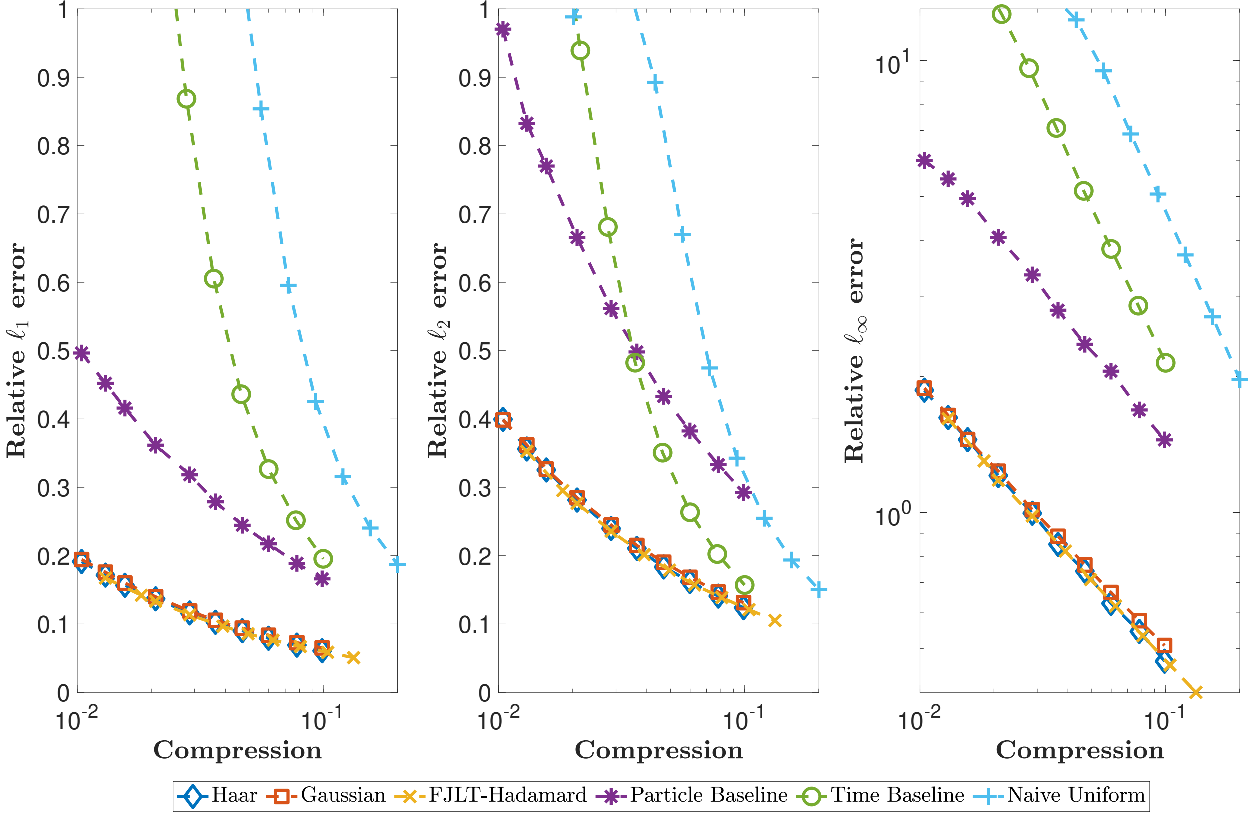


Figure 3.4: The error due to approximating the PSD for the proposed methods (Haar, Gaussian, and FJLT-Hadamard) compared to baselines, on the methanol data. Left: relative ℓ_1 error. Middle: relative ℓ_2 error. Right: relative ℓ_∞ error.

Figure 3.4 shows the error metrics as a function of compression ratio γ in the interesting regime where $\gamma \ll 1$. We see that sketching methods perform better than baseline methods in the ℓ_1, ℓ_2 and ℓ_∞ metrics, and the advantage is most significant when the compression ratio is small.

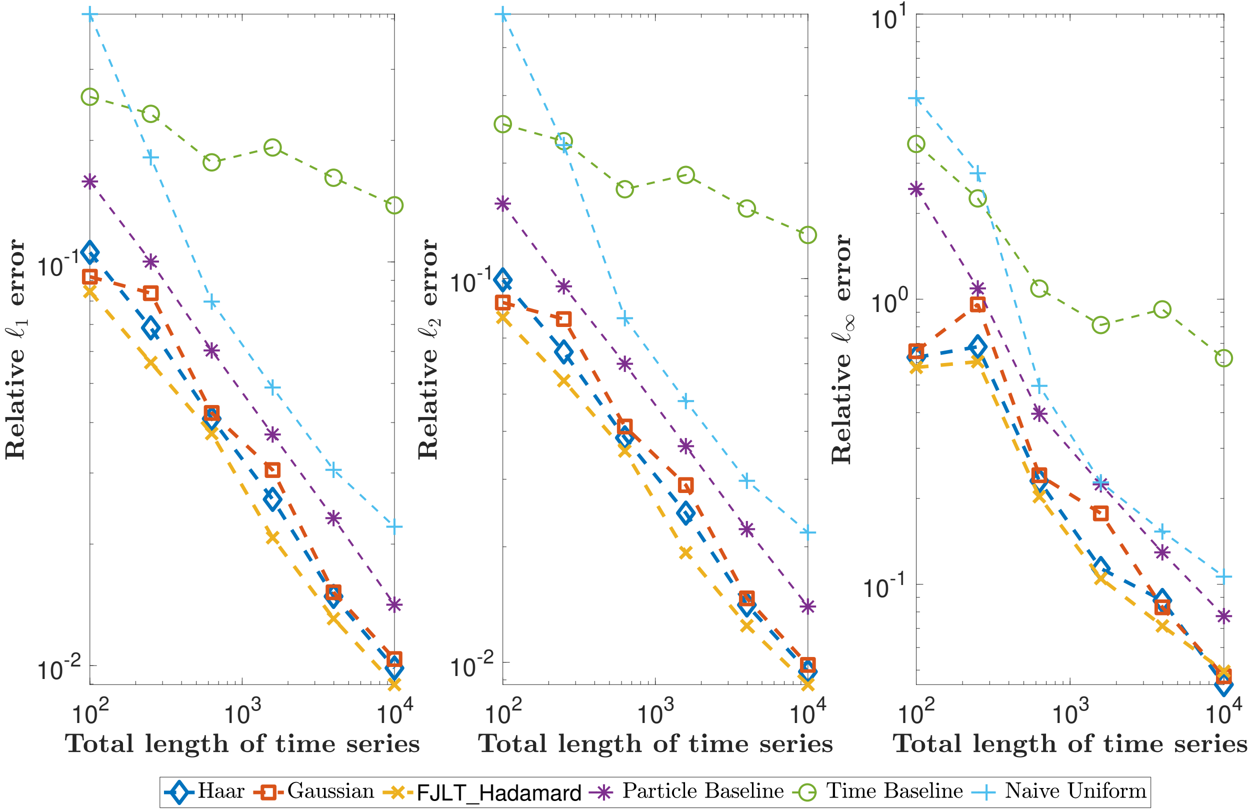


Figure 3.5: Three metrics characterizing the discrepancy between estimated autocorrelation of first 15 lags and the ground truth vs. total length of time signals. The full time signal is divided into $B = \sqrt{T_{\text{long}}}$ blocks, each of which is used to evaluate the first 15 lags of autocorrelation.

Figure 3.5 shows that the ℓ_1, ℓ_2 and ℓ_∞ errors decay to zero as the time series becomes arbitrarily long. Specifically, we take the total simulation time $T_{\text{long}} \to \infty$, and set $B = T = \sqrt{T_{\text{long}}}$ (this is necessary, since the simpler choice of $B = 1$ and $T = T_{\text{long}}$ does not give a consistent estimator even with fully sampled data). The evaluation of the errors of the autocorrelation are with respect to the first 15 lags. The compression ratio of all sketching methods is fixed as 10%. The figure shows that all methods appear to be consistent, with the sketching methods significantly more accurate compared to the *ad hoc* baselines.

Synthetic data The performance of the sketching methods over the classical benchmark methods is significant, but in fact the discrepancy can be arbitrarily large. The supplementary material (1.B) shows a synthetic data set created to be adversarial for the classical methods, for which they perform poorly, whereas the sketching methods do well. The data is created to have a few "special" particles which contribute significantly but are unlikely to be sampled by the particle sampling methods, and to have a few short pulses, so that the relevant time dynamics is likely to be missed by the time sampling methods. The sketching methods are not susceptible to such adversarial examples.

3.5 Conclusions

Since second order statistics like autocorrelation and power density spectral can be computed via the empirical covariance matrix, this means that sketching methods can be used to preserve statistical properties of the data. These sketching methods come with well-understood theory, little extra computational burden, straightforward implementation, and excellent practical performance. For these reasons, we hope they find their place in the numerical simulation toolkit. An interesting future question is whether even more powerful practical estimators of autocorrelation can be achieved by bypassing the estimation of the covariance matrix.

3.6 Further experiments

3.6.1 Alternative baseline methods

We expand on other alternatives for time-dimension compression (beyond the (1) random and (2) power-series sampling), namely

(3) Sparse ruler sampling. The power-series scheme does not generate all possible lags. Sampling schemes that do generate all possible lags (up to some point) are known as *rulers*, and rulers with only a few samples are *sparse rulers*. One can modify the power-series scheme so that each block $\mathcal{I}_0$ is a sparse ruler (we used Wichmann Rulers).

(4) Sampling blocks (Algorithm 8 in [FS02]), which gives good estimates of $R_\tau[\mathbf{X}]$ for small τ, but does not attempt to estimate $R_\tau[\mathbf{X}]$ for τ larger than the block size.

(5) Hierarchical sampling schemes (Algorithm 9 in [FS02]), designed to improve on block sampling by giving a small amount of large lag information. This method is exact for some derived quantities (like diffusion coefficients) but *ad-hoc* for estimating the large-lag autocorrelation. This method has high errors.

Fig. 3.6 compares the sparse ruler sampling and block sampling (Algorithm 8), as well as using the Gaussian sketch. This uses the same $N = 10000$ and $T = 2000$ synthetic data as in Figure 1 in the main text. Both the sparse ruler sampling and block sampling only observe the autocorrelation for short lags. For this reason, the autocorrelation cannot even be interpolated at missing lags, but rather these values must be extrapolated. Rather than do this, the PSD is computed using only the short time lags, but this has the effect of lowering the resolution of the PSD. The bottom part of the figure shows the PSD.

Fig. 3.7 demonstrates the hierarchical sampling scheme on the same data. This scheme samples in blocks (giving a good estimate of short-time autocorrelation lags, much like the block sampling scheme), but then also aggregates blocks to estimate longer lag autocorrelation. For some quantities, such as the diffusion constant when defined as the integral of autocorrelation (e.g., in the discrete case, this is just a sum), this aggregation-by-averaging results in no loss. However, for estimating the autocorrelation itself, the estimate is highly inaccurate. The corresponding PSD is not shown as it is considerably inaccurate.

3.6.2 Synthetic data

The main paper presents realistic data and shows that newly proposed sketching methods outperform classical methods. Here, we show that the difference in performance can be made almost arbitrarily large by choosing adversarial synthetic data. The specific random nature of the sketching methods makes it impossible to create generic adversarial examples, whereas the classical methods which rely on weaker notions of randomness are much more susceptible.

Creation of the data set Consider a collection of $N = 10{,}000$ particles among which 9997 of them share the same eigenfrequency ω while 3 particles have an additional eigenfrequency

ω'. The existence of special particles contributes to the inhomogeneity of the ensemble dynamics. Furthermore, there are 2 pulses in the time range for every particle in the ensemble. Each pulse can be represented by $p_1(t) = p(t - t_1)$, $p_2(t) = p(t - t_2)$ and $p(t) = 10 \sin\left(\frac{\pi}{\delta} t\right) \mathbb{1}(-\frac{\delta}{2} \le t \le \frac{\delta}{2})$, where $\delta \approx 0.6 \cdot \frac{2\pi}{\omega}$ which accounts for more than half of a period of the signal with common eigenfrequency, and $\mathbb{1}$ is the 0-1 indicator function. Each particle has a random phase $\varphi_i \in [0, 2\pi)$. Specifically, 9997 particles have the "common" dynamics

$$(i = 1, \dots, 9997) \quad x_i^{\text{common}}(t) = \sin(\omega t + \varphi_i) + p_1(t) + p_2(t) + \varepsilon_i(t)$$

while 3 "special" particles have one more ingredient in their dynamics

$$(j = 9998, 9999, 10000) \quad x_j^{\text{special}}(t) = \sin(\omega t + \varphi_j) + 80 \sin(\omega' t + \varphi_j') + p_1(t) + p_2(t) + \varepsilon_j(t)$$

so that when taking the expectation the additional frequency component demonstrate significant importance in the overall spectrum, and $\varepsilon(t)$ is white noise. Figure 3.8 shows the signal example of a common particle and a special particle, while the ground truth autocorrelation and power spectral density are shown in Figure 3.9.

Figure 3.10 shows the performance of each sketching method on evaluating the power spectral density of the synthetic data set. The sketching methods perform well, whereas the classical baseline methods perform so poorly as to be unusable. For the sketching methods, even when compression is around 1%, the characteristic peak in the PSD formed by the 3 special particles is still correctly identified, whereas it is completely missed by all 3 classical methods. This is mostly demonstrated by the relative ℓ_∞ error which captures the largest discrepancy in PSD evaluation at any frequency. In fact, all the baseline methods have over 100% relative error on the ℓ_∞ error, regardless of compression.

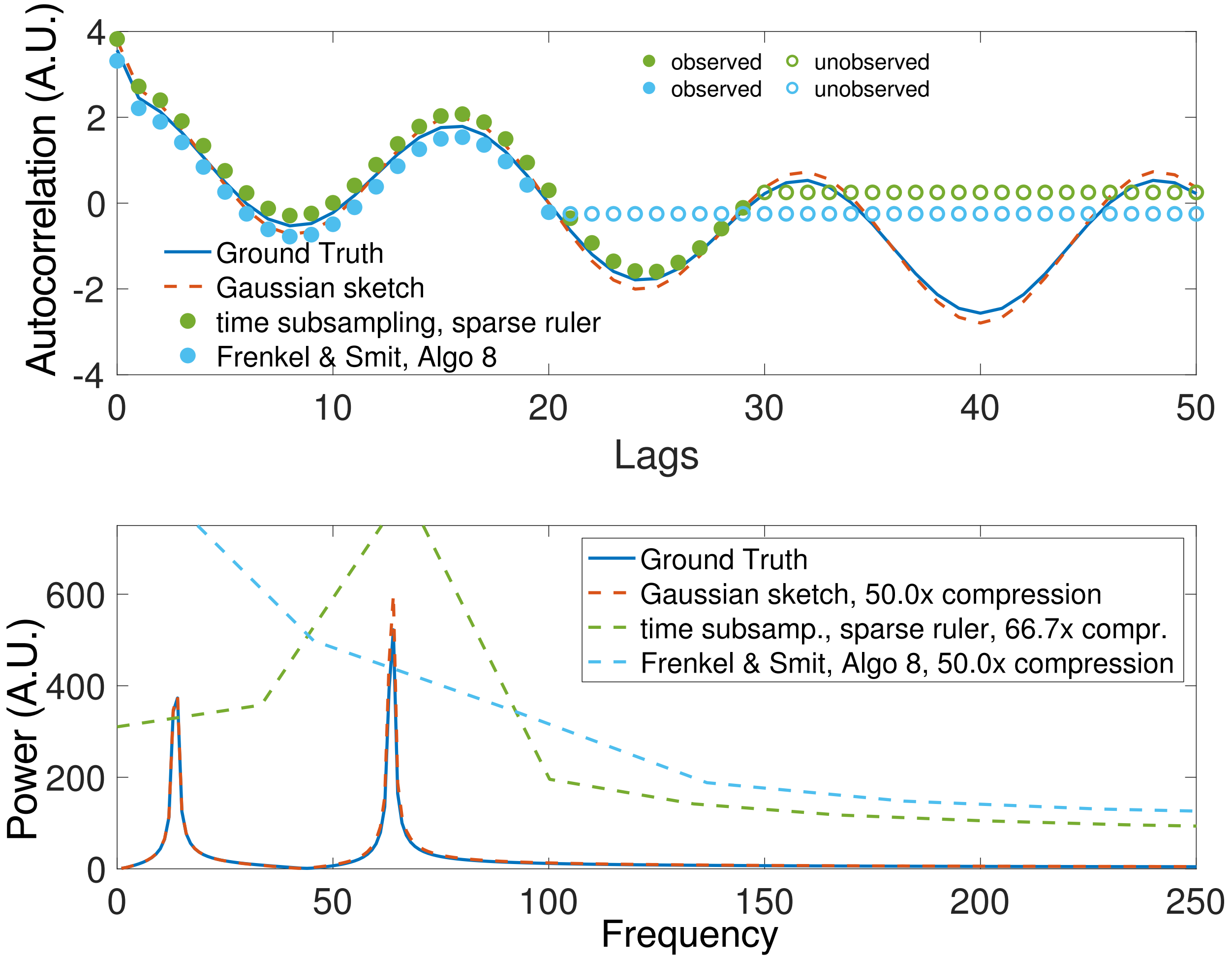


Figure 3.6: Top: autocorrelation, and bottom: Power spectral density (PSD) for a synthetic simulation. The sparse ruler subsampling and the block (Algorithm 8) subsampling miss sampling the autocorrelation at long lags, with the effect of making the PSD estimate have low resolution. Y-axis in arbitrary units for both plots.

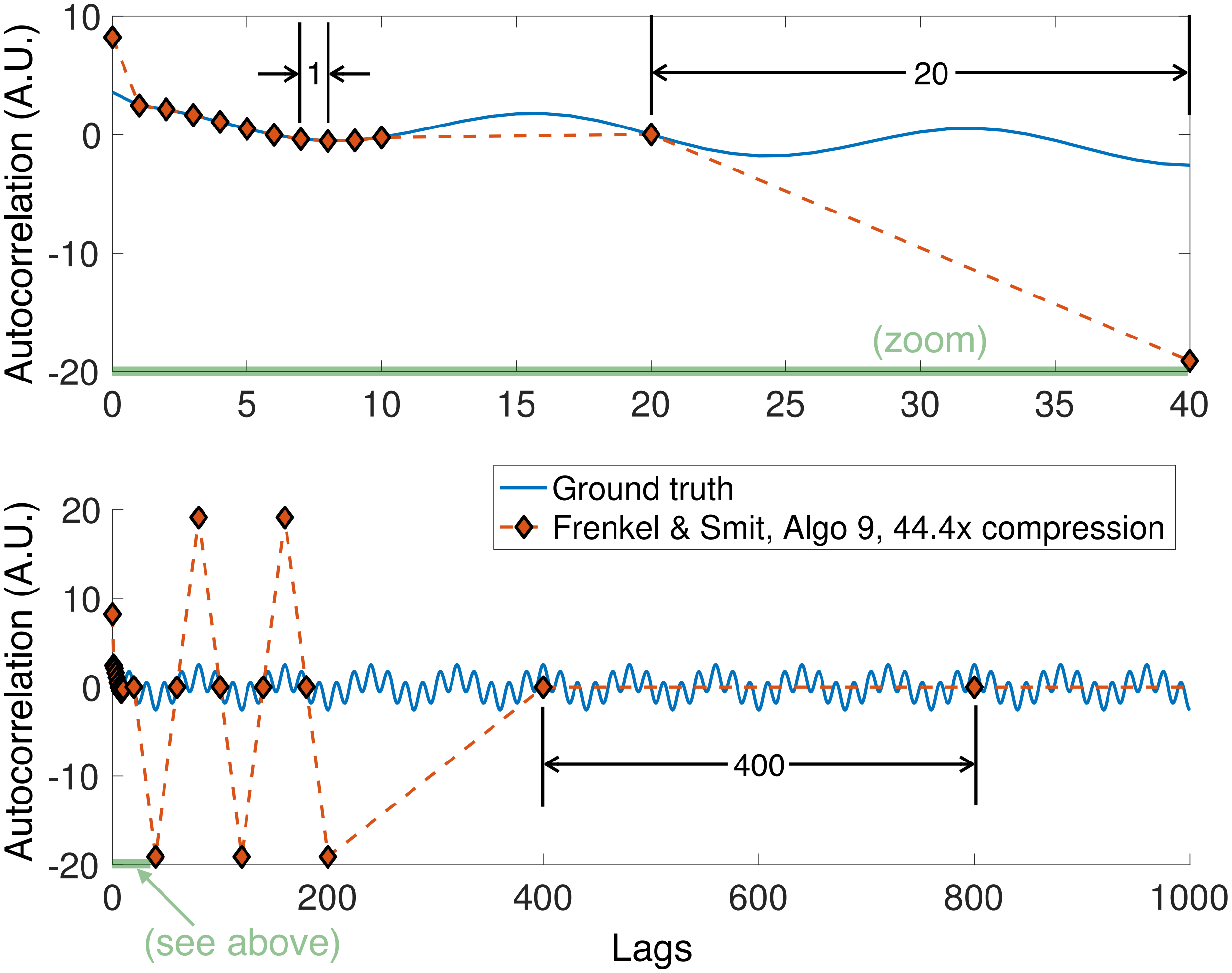


Figure 3.7: Autocorrelation, demonstrating the hierarchical sampling scheme of Algorithm 9. The top plot is a zoomed in version of the bottom plot. The estimate of the autocorrelation at long lags is inaccurate, and the resulting PSD is unusable.

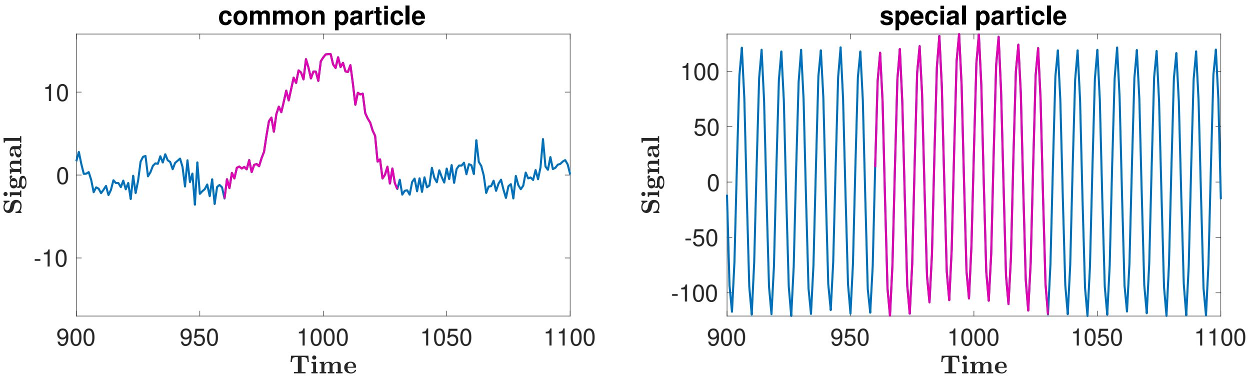


Figure 3.8: Example of particle dynamics in synthetic data. The left subfigures shows the signal of a common particle and the right subfigure shows the signal of a particle with two eigen-frequencies. 2 pulses exist in the synthetic signal and are introduced apart from each other thus not merging their peaks, while we show the zoomed version of one pulse, which is marked in the colour of magenta.

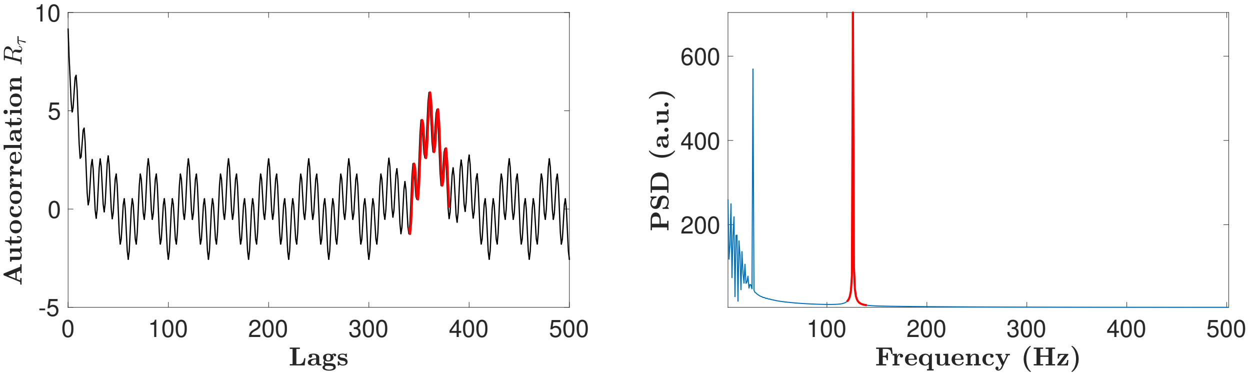


Figure 3.9: Autocorrelation and power spectral density of the synthetic data. The red peak in the power spectral density exists because of special particles, and the red lags in autocorrelation are due to existence of pulses.

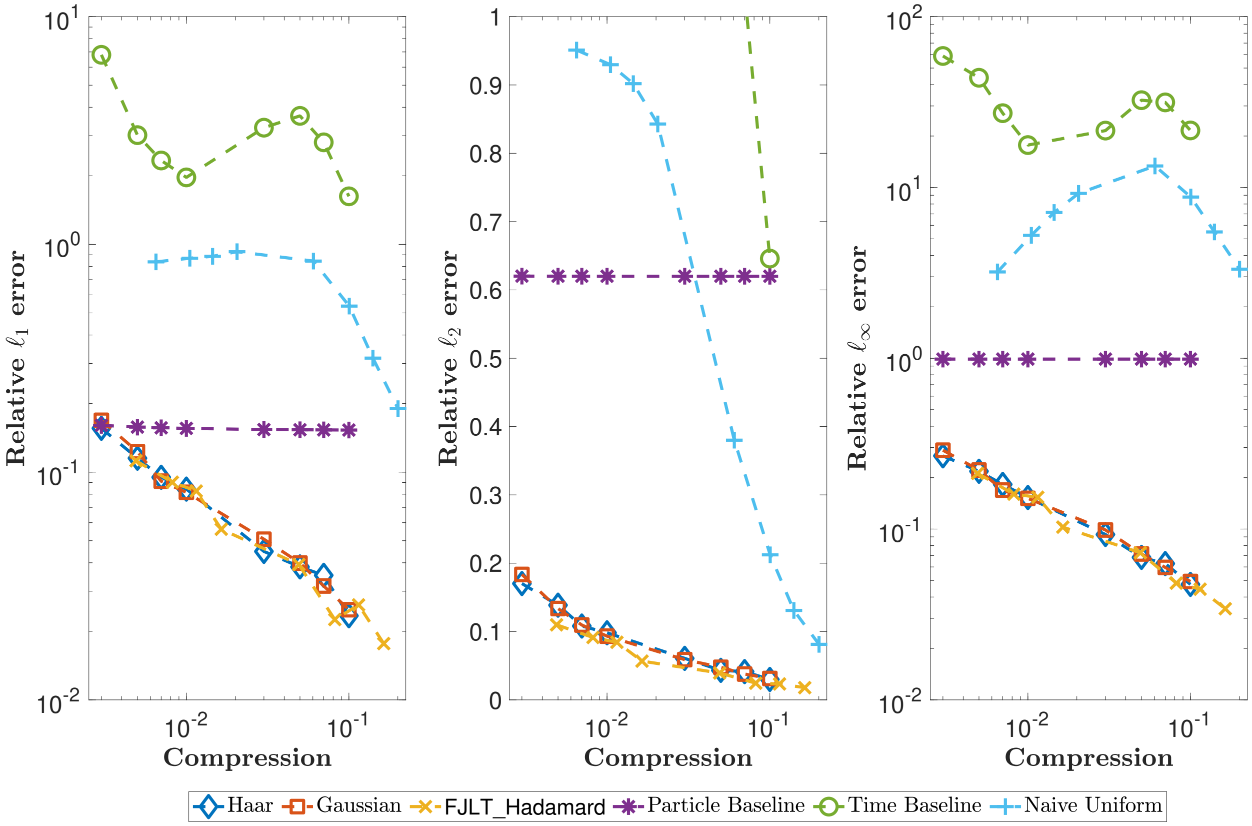


Figure 3.10: Three metrics characterizing accuracy of sketching methods on the PSD in the case of adversarial synthetic data.

Chapter 4

Improved convex relaxations using exact subset selection for ℓ_0 optimization problems

A theme of the previous chapters was finding efficient algorithms to solve a given optimization problem, often using randomization. In this chapter, we examine the optimization problem itself. We propose a new model for sparse recovery in terms of a novel convex optimization problem, and show how one can find the minimizers of this problem.

4.1 Introduction and main idea

Main idea This work considers a novel solver for the sparse signal recovery problem based on a small number of linear measurements. The sparsity of a signal has practical meaning in image compression/restoration [MBP$^+$09], signal denoising [MMB14] and image feature identification identification [MBP14]. The prototype for such sparse recovery problem is formulated through the regression problem with ℓ_0 constraint:

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{x}\|_0 \leq s \tag{4.1}$$

where $\mathbf{A}$ is $n \times p$, usually with $n < p$, and $\|\mathbf{x}\|_0$ is the number of nonzero elements in the vector $\mathbf{x}$. While (4.1) is a combinatorial problem and NP-Hard, if one adds the constraint that $\|\mathbf{x}\|_\infty \leq M$ then [BKM16] shows that this can be solved using mixed-integer quadratic programming (MIQP) techniques with improved performance or to deliver better lower bound certificates. [BKM16] argues that mixed-integer linear programming (MILP) and MIQP solvers, which have combinatorially bad worst-case complexity, have very good heuristics to make them efficient, and that in practice, they can *exactly* solve problems for p around, say, 10^4 (note that n has little effect, as long as we can compute $\mathbf{A}^\top \mathbf{A}$). The argument is compelling: for a moderate amount of data, why solve the least absolute shrinkage and selection operator (lasso) (i.e. $\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2$ such that $\|\mathbf{x}\|_1 \leq s$), which is a convex relaxation of the integer constraint, when you can do exact subset selection for a bit of extra computation? But despite the good solvers, these problems are still hard, and if a state-of-the-art MIQP solver can solve the problem in an hour, it may take weeks to solve a problem with ten times more variables. The central idea behind of the design of the solver proposed in this chapter is to leverage the mature MIQP solver for medium-sized sparse subset selection problems in order to make a better model/relaxation for huge subset selection problems, and meanwhile tackle the rest of the dimension with classic ℓ_1 relaxation.

In this chapter, we consider a similar MILP in the basis pursuit style:

$$\begin{aligned} &\min_{\mathbf{x}} \ \|\mathbf{x}\|_0 \\ &\text{s.t.} \ \ \mathbf{A}\mathbf{x} = \mathbf{b}. \end{aligned} \tag{4.2}$$

To exploit theoretical guarantees in convex analysis, one can derive the Fenchel-Rockafellar dual (FR dual) problem of the primal problem, which is convex and gives a lower bound on the objective, and then apply classic algorithms in convex analysis such as interior-point methods and projected gradient methods to produce a lower bound for the primal problem. Thus instead of solving the primal problem exactly, we propose a hybrid algorithm to solve the FR dual problem of (4.2), which is a convexly relaxed version of the primal MILP problem, and then map back to the primal space to return the solution to the primal problem. Using the MIQP solver allows us to use a tighter relaxation than the usual ℓ_1 relaxation.

We use an ℓ_1-solver (defined in (4.3)) or orthogonal-matching-pursuit solver (OMP) [TG07] to provide an initial starting point $\mathbf{y}_0$ and to pre-select a potential support $T \subset \{1, 2, \ldots, p\}$ for the solution. We will see in the analysis in section 2 that with the introduction of pre-selected support T, the optimization problem can be written as the sum of two parts: one part regarding the support T of solution vector (P1), the other regarding the complement of the support of the solution vector (P2). We use MIQP for P1 to update the gradient for descent purposes, and we can analytically update the gradient for P2. Then the two gradients are combined for descent purposes on the FR dual of the primal problem.

Mathematical details of dual relaxation of ℓ_0 constraint The following formulation

$$\begin{aligned}&\min_{\mathbf{x}} \|\mathbf{x}\|_1\\ &\text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b},\end{aligned} \tag{4.3}$$

known variously as basis pursuit or the lasso, serves as a classic relaxation of the ℓ_0 objective, and we refer to it as the ℓ_1 solver and will use it as a baseline method for comparison. In contrast to the lasso, our formulation of the ℓ_0 to be discussed in the next section is a more specific relaxation of the primal problem. To introduce the Fenchel-Rockafellar dual problem, we first need to define Fenchel-Legendre conjugate function of a function f as

$$f^*(\mathbf{y}) = \sup_{\mathbf{x}} \langle \mathbf{x}, \mathbf{y} \rangle - f(\mathbf{x}) \tag{4.4}$$

We consider the double dual function of ℓ_0 with an additional elementwise upper-boundedness assumption for non-triviality. When f is lower semicontinuous and convex, $f = f^{**}$. We do not assume convexity of the objective, so the double dual serves as a lower bound for the primal objective function [solving the dual and the double-dual problem are equivalent, but the double-dual formulation is often more clear since the variables have the same meaning as in the primal formulation]. The following proposition explains why ℓ_1 is a convex relaxation for ℓ_0 objective. This is included for didactic purposes, as the result is well-known (e.g., it is a special case of the matrix version in [FHB01]).

Proposition 33. *Let* $f(\mathbf{x}) = \|\mathbf{x}\|_0 + \iota_{\{\|\mathbf{x}\|_\infty \leq \lambda\}}(\mathbf{x})$, *then*

$$f^{**}(\mathbf{x}) = \lambda^{-1}\|\mathbf{x}\|_1 + \iota_{\|\mathbf{x}\|_\infty \leq \lambda}(\mathbf{x})$$

where $\lambda > 0$ *is a constant, and for a set* $\mathcal{C}$*, the indicator function* ι *is defined as* $\iota_{\mathcal{C}}(\mathbf{x}) = 0$ *if* $\mathbf{x} \in \mathcal{C}$ *and* $+\infty$ *otherwise.*

Proof. Recall the Fenchel-Legendre dual of $f(\mathbf{x})$ is $f^*(\mathbf{y}) = \max_{\mathbf{x}} \langle \mathbf{y}, \mathbf{x} \rangle - \|\mathbf{x}\|_0 - \iota_{\{\|\mathbf{x}\|_\infty \leq \lambda\}}(\mathbf{x})$. Observe that f is separable in its components, so we can analyze $f^*(\mathbf{y})$ componentwise. For a fixed y_i,

- Case $x_i = 0$, then $f^*(y_i) = 0$
- Case $x_i \neq 0$, $|x_i| \leq \lambda$. Then we consider $x_i y_i - \|x_i\|_0 - \iota_{\{|x_i| \leq \lambda\}}(x)$. If $y_i \geq 0$, then by observation the maximum is achieved at $x_i = \lambda$ with maximum value $\lambda y_i - 1$. Similarly if $y_i \leq 0$, then the maximizer is $x_i = -\lambda$ with maximum value $\lambda |y_i| - 1$.

In conclusion, $f^*(y_i) = \max\big\{0 \,(\text{when } x = 0\,), |\lambda y_i| - 1 \,(\text{when } x_i \neq 0 \text{ and } |x_i| \leq \lambda\,)\big\} = \big[\lambda |y_i| - 1\big]_+$. For a multi-variable function, we have correspondingly

$$f^*(\mathbf{y}) = \big[\lambda |\mathbf{y}| - 1\big]_+. \tag{4.5}$$

Regarding $f^{**}(\mathbf{x})$, for each component, with $\lambda > 0$, we have

$$f^{**}(x_k) = \sup_{y_k} \langle x_k, y_k \rangle - [|\lambda y_k| - 1]_+ = \begin{cases} 0 & \text{if } x_k = 0 \\ |\frac{x}{\lambda}| & \text{if } |x_k| \leq \lambda \\ \infty & \text{if } |x_k| > \lambda \end{cases}$$

Hence, for $f(\mathbf{x}) = \|\mathbf{x}\|_0 + \iota_{\{\|\mathbf{x}\|_\infty \leq \lambda\}}(\mathbf{x})$, we have

$$f^{**}(\mathbf{x}) = \sup_{\mathbf{y}} \langle \mathbf{x}, \mathbf{y} \rangle - \sum_{i=1}^{n} \big[|\lambda y_i| - 1\big]_+ = \lambda^{-1} \|\mathbf{x}\|_1 + \iota_{\{\|\mathbf{x}\|_\infty \leq \lambda\}}(\mathbf{x})$$

□

4.1.1 Prior art

The line of related research starts from ℓ_1 regularization, which is the well-studied lasso formulation known to encourage *sparse* solutions. A variant of lasso which also aims to find sparse representation among a large collection of basis vectors is basis pursuit (BP) [CDS98]: $\min_{\mathbf{x}} \|\mathbf{x}\|_1$ such that $\mathbf{A}\mathbf{x} = \mathbf{b}$. All these problems culminate in the systematic discussion on the relation between underlying composition of the signal and the property of the observation matrix, which is summarized in the topic of *compressed sensing* [CRT06].

We briefly review established algorithms for solving sparsity constrained optimization problems directly without preprocessing by convex relaxation. Iterative methods to approximately solve the sparsity constrained problems include iterative hard thresholding (IHT) [BD09] and greedy methods like *orthogonal matching pursuit*. The iterative methods on a high level intend to extract information from the residual of the current solution, thus update the solution and/or the support and repeat, while the greedy method repeatedly seeks the subspace of the column space of the measurement matrix onto which the projection of the signal is maximized. Other hard thresholding algorithms to assure sparsity constraints include CoSamp [NT09] and conjugate gradient IHT (CGIHT) [BTW15]. The theoretical guarantees for CoSamp and CGIHT requires the measurement matrix A to satisfy the restricted isometry property, which is NP-hard to verify and does not always hold.

4.2 Design of the hybrid solver

4.2.1 Pre-selected support and the separability of the FR-dual problem

First we use OMP-solver to solve for the support T. We assume the prior knowledge of the upper bound of the sparsity of the solution vector $\mathbf{x}$; i.e., $\|\mathbf{x}\|_0 \leq s$.

Recall that $\mathbf{A}$ is $n \times p$ dimensional. Let T be a subset of $1, \ldots, p$ of size n or less, so that $\mathbf{A}_T$ (that is, the matrix formed by choosing the columns if $\mathbf{A}$ where the column belongs to T; in Matlab-notation, `A(:,T)`) is invertible. Let $\mathbf{x}_T$ be the corresponding entries of the p-dimensional vector $\mathbf{x}$. For exposition, assume T is $1, 2, \ldots, |T|$ so that we may write $\mathbf{x} = [\mathbf{x}_T, \mathbf{x}_{T^c}]$ where T^c is the complement of T, i.e., $T^c = \{|T| + 1, \ldots, p\}$.

We reformulate the original optimization problem (4.2) with additional domain constraints into minimizing the objective function

$$\begin{aligned} F(\mathbf{x}) &= \|\mathbf{x}\|_0 + \iota_{\mathcal{B}}(\mathbf{x}) + \iota_{\{\mathbf{Ax}=\mathbf{b}\}}(\mathbf{x}) \\ &= \underbrace{\|\mathbf{x}_T\|_0 + \iota_{\mathcal{B}_T}(\mathbf{x}_T)}_{F_T(\mathbf{x}_T)} + \underbrace{\|\mathbf{x}_{T^c}\|_0 + \iota_{\mathcal{B}_{T^c}}(\mathbf{x}_{T^c})}_{F_{T^c}(\mathbf{x}_{T^c})} + \iota_{\{\mathbf{Ax}=\mathbf{b}\}}(\mathbf{x}) \end{aligned} \tag{4.6}$$

Here we define the set $\mathcal{B} = \mathcal{B}_T \times \mathcal{B}_{T^c}$, where

- $\mathcal{B}_T = \{\mathbf{x}_T \mid \|\mathbf{A}_T\mathbf{x}_T - \mathbf{b}\|_2 \leq \varepsilon \text{ and } \|\mathbf{x}_T\|_\infty \leq \lambda\}$
- $\mathcal{B}_{T^c} = \{\mathbf{x}_{T^c} \mid \|\mathbf{x}_{T^c}\|_\infty \leq \lambda_{\text{leak}}\}$.

Consequently $F(\mathbf{x})$ can be written as $F(\mathbf{x}) = F_T(\mathbf{x}_T) + F_{T^c}(\mathbf{x}_{T^c}) + \iota_{\{\mathbf{Ax}=\mathbf{b}\}}(\mathbf{x})$, where we let $F_T(\mathbf{x}_T) = \|\mathbf{x}_T\|_0 + \iota_{\mathcal{B}_T}(\mathbf{x}_T)$ and $F_{T^c}(\mathbf{x}_{T^c}) = \|\mathbf{x}_{T^c}\|_0 + \iota_{\mathcal{B}_{T^c}}(\mathbf{x}_{T^c})$. This formulation separates the original primal problem into two subproblems that are coupled via the $\mathbf{Ax} = \mathbf{b}$ constraint.

Define $f(\mathbf{x}) = \|\mathbf{x}\|_0 + \iota_{\mathcal{B}}(\mathbf{x})$, and $g(\mathbf{x}) = \iota_{\mathbf{x}=\mathbf{b}}(\mathbf{x})$. Then, the primal problem to solve is $\text{minimize}_{\mathbf{x}}\ F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{Ax})$. Following [BC11], the Fenchel-Rockafellar dual problem of minimizing (4.6) is defined as

$$\text{minimize}_{\mathbf{y}}\ G(\mathbf{y}) \text{ where } G(\mathbf{y}) = f^*(\mathbf{A}^\top\mathbf{y}) + g^*(-\mathbf{y}). \tag{4.7}$$

To solve the convexly-relaxed version of (4.2), we minimize $G(\mathbf{y})$. We denote the minimizer of $G(\mathbf{y})$ as $\mathbf{y}^\star$, which we will use to discover the minimizer $\mathbf{x}^\star$ for (4.6) by choosing $\mathbf{x}^\star$ as (see results in chapter 19 in [BC11] regarding FR-dual framework)

$$\mathbf{x}^\star = \partial_{\mathbf{A}^\top\mathbf{y}} f^*(\mathbf{A}^\top\mathbf{y}^\star). \tag{4.8}$$

where the notation $\partial_{\mathbf{x}} f(\mathbf{x}_0)$ means the subgradient of f w.r.t. $\mathbf{x}$ evaluated at the point $\mathbf{x}_0$. With this above $\mathbf{x}^\star$ in (4.8), if $\mathbf{y}^\star$ is optimal for dual FR problem, then $\mathbf{x}^\star$ is optimal for primal FR problem. $-G(\mathbf{y})$ serves as a lower bound to the primal objective $F(\mathbf{x})$ as $\max_{\mathbf{y}}\ -G(\mathbf{y}) \leq \min_{\mathbf{x}}\ F(\mathbf{x})$, and considering that $G(\mathbf{y})$ is convex due to the convexity of Fenchel-Legendre conjugate function, $-G(\mathbf{y})$ is a convexly relaxed version of $F(\mathbf{x})$, and the dual FR problem is a convexly relaxed problem of the primal FR problem. (see chapter 15 in [BC11])

4.2.2 Subgradient descent algorithm

To compute $f^*(\mathbf{A}^\top\mathbf{y})$, we find

$$f^*(\mathbf{A}^\top\mathbf{y}) = \underbrace{\big[\max_{\mathbf{x}_T}\langle(\mathbf{A}_T)^\top\mathbf{y},\mathbf{x}_T\rangle - \|\mathbf{x}_T\|_0 - \iota_{\{\|\mathbf{A}_T\mathbf{x}_T-\mathbf{b}\|_2\le\varepsilon,\|\mathbf{x}_T\|_\infty\le\lambda\}}(\mathbf{x}_T)\big]}_{f_1^*(\mathbf{A}^\top\mathbf{y})} + \underbrace{\sum_{i\in T^c}\big[\lambda_{\text{leak}}|(\mathbf{A}_{T^c})^\top\mathbf{y}|_i - 1\big]_+}_{f_2^*(\mathbf{A}^\top\mathbf{y})}$$

and by the definition of Fenchel-Legendre conjugate, $g^*(-\mathbf{y}) = -\langle\mathbf{b},\mathbf{y}\rangle$. Hence,

$$G(\mathbf{y}) = \underbrace{\big[\max_{\mathbf{x}_T}\langle(\mathbf{A}_T)^\top\mathbf{y},\mathbf{x}_T\rangle - \|\mathbf{x}_T\|_0 - \iota_{\{\|\mathbf{A}_T\mathbf{x}_T-\mathbf{b}\|_2\le\varepsilon,\|\mathbf{x}_T\|_\infty\le\lambda\}}(\mathbf{x}_T)\big]}_{G_1(\mathbf{y})} + \underbrace{\sum_{i\in T^c}\big[\lambda_{\text{leak}}|(\mathbf{A}_{T^c})^\top\mathbf{y}|_i - 1\big]_+}_{G_2(\mathbf{y})} - \langle\mathbf{b},\mathbf{y}\rangle \tag{4.9}$$

We can see that the optimization problem is separable in $\mathbf{x}_T$ and $\mathbf{x}_{T^c}$, where for the former we use MIQP in the (sub)gradient descent process, and for the latter, it reduces to a closed-form calculation. We will use subgradient descent algorithms (or variants) to minimize $G(\mathbf{y})$, hence we need to determine at least one subgradient vector ∂G of the function G.

We determine $\partial G_1(\mathbf{y})$ through a MIQP solver. Recall that a conjugate function $f^*(\mathbf{y})$ of any function $f(\mathbf{x})$ is defined as $f^*(\mathbf{y}) = \max_{\mathbf{x}}\langle\mathbf{y},\mathbf{x}\rangle - f(\mathbf{x})$, so that if $\mathbf{x}\in\arg\max_{\mathbf{x}}\langle\mathbf{y},\mathbf{x}\rangle - f(\mathbf{x})$, then $\mathbf{x}\in\partial[f^*(\mathbf{y})]$ (see chapter 16 in [BC11]). Similarly, for $f^*(\mathbf{A}\mathbf{y}) = \max_{\mathbf{x}}\langle\mathbf{A}\mathbf{y},\mathbf{x}\rangle - f(\mathbf{x})$, if $\mathbf{x}\in\arg\max_{\mathbf{x}}\langle\mathbf{A}\mathbf{y},\mathbf{x}\rangle - f(\mathbf{x})$, then $\mathbf{A}^\top\mathbf{x}\in\partial_{\mathbf{y}}[f^*(\mathbf{A}\mathbf{y})]$, which contributes to the first part of $\partial G_1(\mathbf{y})$.

Regarding $\partial G_2(\mathbf{y})$, taking the subgradient with respect to y_k, we have

$$\begin{aligned}
\frac{\partial G_2(\mathbf{y})}{\partial y_k} &= \partial_k\Bigg(\sum_{i\in T_c}\big[\lambda_{\text{leak}}|(\mathbf{A}_{T^c})^\top\mathbf{y}|_i - 1\big]_+\Bigg)\\
&= \partial_k\Bigg(\sum_{i\in T_c}\big[\lambda_{\text{leak}}(-1)^{\operatorname{sgn}[((\mathbf{A}_{T^c})^\top\mathbf{y})_i]}((\mathbf{A}_{T^c})^\top\mathbf{y})_i - 1\big]_+\Bigg)\\
&= \partial_k\Bigg(\sum_{i\in T^c}\big[\lambda_{\text{leak}}(-1)^{\operatorname{sgn}[((\mathbf{A}_{T^c})^\top\mathbf{y})_i]}((\mathbf{A}_{T^c})^\top\mathbf{y})_i - 1\big]\chi_i\Bigg)\\
&= \sum_{i\in T^c}\lambda_{\text{leak}}(-1)^{\operatorname{sgn}[(\mathbf{A}_{T^c}^\top\mathbf{y})_i]}\chi_i\cdot((\mathbf{A}_{T^c})^\top)_{ik}
\end{aligned}$$

where we have defined $\chi_i = \begin{cases} 1 & \text{if } (-1)^{\mathrm{sgn}[(\mathbf{A}_{T^c}^\top \mathbf{y})_i]}((\mathbf{A}_{T^c})^\top \mathbf{y})_i \geq \frac{1}{\lambda_{\text{leak}}} \\ 0 & \text{if } (-1)^{\mathrm{sgn}[(\mathbf{A}_{T^c}^\top \mathbf{y})_i]}((\mathbf{A}_{T^c})^\top \mathbf{y})_i < \frac{1}{\lambda_{\text{leak}}} \end{cases}$.

To map the dual variable $\mathbf{y}$ back to the primal space, we define the map $\mathbf{x} = \Phi(\mathbf{y})$ from dual space to primal space as the following. For the support of $\mathbf{x}$ on T, compute $\partial_{\mathbf{A}^\top \mathbf{y}}[f_1^*(\mathbf{A}^\top \mathbf{y})]$ as $\mathbf{x}_T \in \arg\max_{\mathbf{x}_T} \langle (\mathbf{A}_T)^\top \mathbf{y}, \mathbf{x}_T \rangle - \|\mathbf{x}_T\|_0 - \iota_{\{\|\mathbf{A}_T \mathbf{x}_T - \mathbf{b}\|_2 \leq \varepsilon, \|\mathbf{x}_T\|_\infty \leq \lambda\}}(\mathbf{x}_T)$.

For the support on T^c of $\partial_{\mathbf{A}^\top \mathbf{y}}[f^*(\mathbf{A}^\top \mathbf{y})]$, we find

$$\mathbf{x}_{T^c} = \partial_{\mathbf{A}^\top \mathbf{y}}[f_2^*(\mathbf{A}^\top \mathbf{y})] = \begin{cases} \lambda_{\text{leak}} \cdot \mathrm{sgn}((\mathbf{A}_{T^c})^\top \mathbf{y}) & \text{if } \lambda_{\text{leak}} |(\mathbf{A}_{T^c})^\top \mathbf{y}|_i - 1 \geq 0 \\ 0 & \text{if } \lambda_{\text{leak}} |(\mathbf{A}_{T^c})^\top \mathbf{y}|_i - 1 < 0 \end{cases}$$

In summary, we give the following Naive Subgradient Descent Algorithm, Algo. 4. We expect that the weak duality may compromise the accuracy of the solution, while the improved convex relaxation should render the gap narrower than the case for lasso.

Algorithm 4 Naive Subgradient Descent with MIQP

```
Require: $\mathbf{A}$, $\mathbf{b}$, MaxIter, sparsity target $s$
Determine $\mathbf{y}_0$ with $\ell_1$-solver or OMP-solver as iteration starting point for FR-dual problem.
Select indices of largest $s$ components in magnitude from $\mathbf{x}_0$, the primal solution returned by
    the warmup solver, as the support index set $T$;
$k = 0$
while $k <$ MaxIter do
    Use MIQP solver (or method presented in [BKM16]) to solve
        $$\mathbf{d}_T \in \arg\max_{\mathbf{x}_T} \langle (\mathbf{A}_T)^\top \mathbf{y}_k, \mathbf{x}_T \rangle - \|\mathbf{x}_T\|_0 - \iota_{\{\|\mathbf{A}_T \mathbf{x}_T - \mathbf{b}\|_2 \leq \varepsilon, \|\mathbf{x}_T\|_\infty \leq \lambda\}}(\mathbf{x}_T) \quad (4.10)$$
    Let $\boldsymbol{\xi}_k \in \partial G(\mathbf{y}_k) = \mathbf{A}_T \mathbf{d}_T + \partial\left( \sum_i \left[ \lambda_{\text{leak}} |(\mathbf{A}_{T^c})^\top \mathbf{y}_k|_i - 1 \right]_+ \right) - \mathbf{b}$
    Update $\mathbf{y}_{k+1} = \mathbf{y}_k - \eta_k \boldsymbol{\xi}_k$, where $\eta_k$ is the step size used in $k^{\text{th}}$ step
end while
Return the final solution as $\mathbf{x}^\star = \partial_{\mathbf{A}^\top \mathbf{y}} f^*(\mathbf{A}^\top \mathbf{y}^\star) = [\mathbf{x}_T, \mathbf{x}_{T^c}]$ ▷ The order of rows of $\mathbf{x}$ should
    be identical to the partition of $T$ and $T^c$
```

4.2.3 Acceleration with the bundle method, and the algorithm of hybrid solver

We notice that the subgradient descent does not converge fast enough, so we introduce the bundle method into the naive version of the solver in the hope of accelerating the convergence speed of the subgradient descent. The fundamental ideas of bundle methods [BKM14] is that one can leverage all information accumulated from previous iterations, which is stored in the bundle, to provide the most informed update to the solution at current iteration.

Specifically, at each iteration, a subgradient is computed at that iteration point. Therefore, a new supporting hyperplane to the graph of the objective function is generated based on this new subgradient vector; this plane is known as a *cutting plane*. A "bundle" is composed of some of the cutting planes to the objective function through the process of subgradient descent. With this bundle we can construct quadratic programming problems with the help of the local approximation to the objective function at the current iteration point to seek next update. At k-th iteration, the collection of all previous subgradient gives the cutting plane union as the linearized local approximation to the objective $F(\mathbf{x})$:

$$\widetilde{F}_k(\mathbf{x}) = \max_{j \in J_k} \{F(\mathbf{x}_j) + \boldsymbol{\xi}_j^\top (\mathbf{x} - \mathbf{x}_j)\}, \tag{4.11}$$

where $J_k = \{1, 2, \cdots, k-1\}$ (this $\widetilde{F}_k(\mathbf{x})$ is a lower bound for the objective $F(\mathbf{x})$). Therefore, at the iteration point $\mathbf{x}_k$, one can use the following quadratic optimization problem to seek for direction $\mathbf{d}_k$ for next update:

$$\min_{\mathbf{d}_k} \left\{ \widetilde{F}_k(\mathbf{x}_k + \mathbf{d}_k) + \frac{\mu_k}{2} \|\mathbf{d}_k\|^2 \right\}, \tag{4.12}$$

where we set

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}_k.$$

Here the quadratic term with magnitude $\mu_k/2$ is introduced to ensure a minimizer $\mathbf{d}_k$ always exists since $\widetilde{F}_k(\mathbf{x})$ is potentially not lower bounded at k-th iteration when there are not sufficiently many hyperplanes in the bundle. A more sophisticated penalty term can be in the form of $\frac{1}{2}\mathbf{d}_k^\intercal \mathbf{M} \mathbf{d}_k$ for some positive-definite matrix $\mathbf{M}$, which characterizes the variation of curvatures along different dimensions of the geometry of objective F. To leverage the available linear programming solver such as `Gurobi`[1] or `MOSEK`[2] to solve (4.12), it needs to be cast into a smooth quadratic programming

[1] https://www.gurobi.com/products/gurobi-optimizer/

[2] https://www.mosek.com/

problem as the following [Mä02]:

$$\begin{aligned} \min_{v,\mathbf{d}_k} \quad & v + \frac{\mu_k \|\mathbf{d}_k\|^2}{2} \\ \text{s.t.} \quad & \boldsymbol{\xi}_j^\intercal \mathbf{d}_k - \underbrace{\Big(F(\mathbf{x}_k) - F(\mathbf{x}_j) - \boldsymbol{\xi}_j^\intercal (\mathbf{x}_k - \mathbf{x}_j)\Big)}_{\text{linear approx. error for } j\text{-th hyperplane}} \leq v \quad \forall j \in J_k \end{aligned} \tag{4.13}$$

Line 6 in algorithm 4 is executed based on (4.13).

Optional protocol: support exploration To further explore the range of the measurement matrix $\mathbf{A}$, we suggest an optional protocol to update the support index set T in each iteration out of potential missed shots incurred by the warmup solver. When the k-th update is made in each iteration, one can compute $\mathbf{x}_k = \Phi(\mathbf{y}_k)$, and re-select the indices of the largest s components to be combined with previous support T_{k-1} as the new support T_k. This protocol expands the size of support under consideration, which provides more expressiveness for the MIQP subproblem. Similar support refresh protocol appears in the solver CoSamp.

To conclude, we propose a hybrid solver Algo. 5.

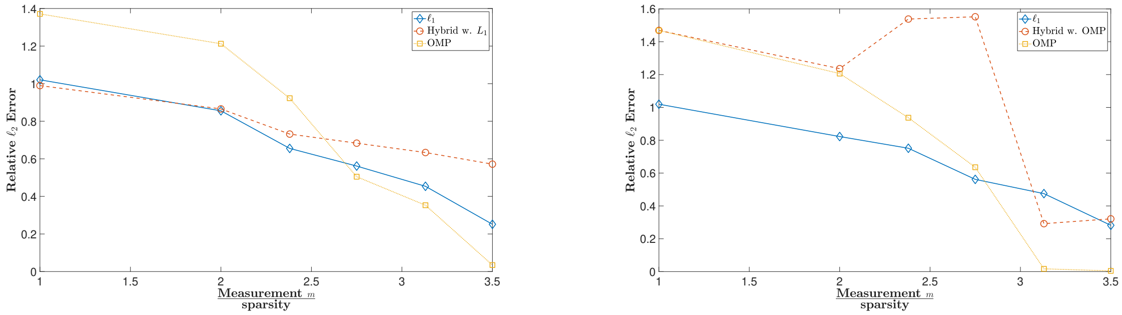
Algorithm 5 Subgradient Descent Accelerated by Bundle Method (BM)

```
Require: $\mathbf{A}$, $\mathbf{b}$, MaxIter, sparsity target $s$
Determine $\mathbf{y}_0$ with L1-solver or OMP-solver as iteration starting point for FR-dual problem;
Select indices of largest $s$ components of the primal solution $\mathbf{x}_0$ returned by the warmup solver as the support index set $T$
while count of iteration < MaxIter do
    Use MIQP solver (or method presented in [BKM16]) to solve
        $\max_{\mathbf{x}_T} \langle (\mathbf{A}_T)^\top \mathbf{y}_k, \mathbf{x}_T \rangle - \|\mathbf{x}_T\|_0 - \iota_{\|\mathbf{A}_T\mathbf{x}_T - \mathbf{b}\|_2 \le \varepsilon, \|\mathbf{x}_T\|_\infty \le \lambda}(\mathbf{x}_T)$    (4.14)
                                                        ▷ Denote the maximizer of (4.14) as $\mathbf{d}_T$
    Let $\boldsymbol{\xi}_k \in \partial G(\mathbf{y}_k) = \mathbf{A}_T\mathbf{d}_T + \partial\left(\sum_i \left[\lambda_{\text{leak}} |(\mathbf{A}_{T^c})^\top \mathbf{y}_k|_i - 1\right]_+\right) - \mathbf{b}$    ▷ BM Step 1: expand the subgradient bundle
    Let $\mathbf{y}_{\text{new}} = \arg\min_{\mathbf{y}} \left\{ \max_{j \in J_k} \left\{ G(\mathbf{y}_k) + \boldsymbol{\xi}_j^\top (\mathbf{y} - \mathbf{y}_k) \right\} + \frac{\mu_k}{2} (\mathbf{y} - \mathbf{y}_k)^2 \right\}$
    Define $G_k(\mathbf{y}) := \max_{j \in J_k} \left\{ G(\mathbf{y}_k) + \boldsymbol{\xi}_j^\top (\mathbf{y} - \mathbf{y}_k) \right\}$
    if $\exists\, m \in (0, \frac{1}{2})$ such that $G(\mathbf{y}_{\text{new}}) - G(\mathbf{y}_k) \le m\big(G_k(\mathbf{y}_{\text{new}}) - G(\mathbf{y}_k)\big)$ then
        let $\mathbf{y}_{k+1} = \mathbf{y}_{\text{new}}$;    ▷ BM Step 2: descent step (executed if the update is wise)
    else
        $\mathbf{y}_{k+1} = \mathbf{y}_k$
    end if
end while
Return the final solution as $\mathbf{x}^\star = \partial_{\mathbf{A}^\top \mathbf{y}} f^*(\mathbf{A}^\top \mathbf{y}^\star) = [\mathbf{x}_T, \mathbf{x}_{T^c}]$ ▷ The order of rows of $\mathbf{x}$ should be identical to the partition of $T$ and $T^c$
```

4.3 Numerical results

Baseline methods We use an ℓ_1 solver and Orthogonal Matching Pursuit (OMP) as baselines to compare the hybrid solver with. Recall that the ℓ_1 model is $\min_{\mathbf{x}} \|\mathbf{x}\|_1$ subject to $\mathbf{A}\mathbf{x} = \mathbf{b}$, which is a linear program, and the `Gurobi` solver is used to solve it. OMP method is implemented by the `wmpalg` package in Matlab[3] .

Results Figure 4.1 shows the performance comparison between three solvers. We test three solvers on synthetic datasets where, in Matlab notation, $\mathbf{A} = \texttt{randn(m,2000)}$, $\mathbf{x}_{\text{ground truth}} =$ a random sparse vector in $\mathbb{R}^{2000}$ with sparsity 100 and $\mathbf{b} = \mathbf{A}\mathbf{x}_{\text{ground truth}}$. The test is repeated for various measurement setting m.



(a) The measurement matrix $A \in \mathbb{R}^{m\times 2000}$, the proposed hybrid solver uses the ℓ_1 solver to compute T; in relative ℓ_2 error

(b) The measurement matrix $A \in \mathbb{R}^{m\times 2000}$, the proposed hybrid solver uses OMP to compute T; in relative ℓ_2 error

Figure 4.1: Comparison between three solvers in the moderate measurement-sparsity ratio regime. The sparsity is 100.

Although unlikely to fundamentally renovate the performance of the hybrid solver with a more informed hyperparameter setting, we report the following combination which generates the best performances among all combinations we have tested. The following are the hyperparameters involved in the implementation of bundle methods to create Figure 4.1:

- Infinity norm constraint bound λ and λ_{leak}: $\lambda = 3\|\mathbf{y}_0\|$ and $\lambda_{\text{leak}} = 0.1\lambda$
- Bundle method regularization parameter for the quadratic term μ_k (like an inverse stepsize): $\mu_{k+1} = (k-1)^3\mu_k$ when line 10 in algorithm 5 is reached; $\mu_{k+1} = \frac{10}{k^3}\mu_k$ when line 8 is executed instead.

[3] https://www.mathworks.com/help/wavelet/ref/wmpalg.html

- Tolerance parameter ε for MIQP subproblem: $\varepsilon = 2\max\{\|\mathbf{A}\mathbf{x}_0 - \mathbf{b}\|, 0.01\}$, where $\mathbf{x}_0$ is the primal solution returned by the ℓ_1 solver
- Support T: we have tried two ways dealing with support. The first is to combine the support of largest s-entries of solutions returned by ℓ_1 solver and OMP solver, which has maximally $2s$ nonzero entries, and hard-threshold the returned solution when exitting algorithm 5. The second is to use the optional support exploration protocol.

4.3.1 Conclusion

We do not observe improvement in evaluation accuracy by implementing the hybrid solver in terms of relative ℓ_2 error or relative ℓ_∞ error when the measurement-sparsity ratio is greater than 1, regardless of the choice of warmup solvers.

The stability of the hybrid solver with OMP solver as warmup is poor, demonstrated by frequent breakdown of the bundle method with the pre-selected support returned by OMP solver. It appears that the breakdown is due to the failure of solving the quadratic programming problem (4.13) for new descent direction vector. As there is not an obvious starting dual point with OMP used as warmup solver, the heuristic of using the dual point returned by ℓ_1 solver as the iteration start point coupled with pre-determined support returned by OMP solver leads to this instability. Also, the support expansion protocol does not mitigate this issue.

In the case of using the ℓ_1 solver for warmup, the slow convergence of subgradient descent may be a factor for the low accuracy improvement of the hybrid solver. The convergence in dual space does not necessarily correspond to the convergence in the primal space due to the dual-primal gap. The computation complexity is high, due to repeatedly solving the MIQP and QP within the hybrid solver, which makes it expensive to run many iterations.

We point out that the comparison result between these three solvers still holds in the same measurement-sparsity ratio domain when the measurement outcome $\mathbf{b}$ is corrupted with Gaussian noise $\boldsymbol{\epsilon}$ as $\mathbf{b} = \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon}$.

In summary, this chapter is a first attempt at using a finer convex relaxation to improve the solution accuracy of sparse recovery problems, compared to established solvers like ℓ_1 and OMP. Our numerical results are not promising, and this specific convex relaxation does not appear to help. However, we have opened up a new line of research: how can one leverage powerful MIQP solvers that can exactly solve the combinatorial problem in medium dimensions, in order to better solve sparse recovery problems in very high dimensions?

Bibliography

[ABS11] H. Attouch, J. Bolte, and B.F. Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized Gauss-Seidel methods. Mathematical Programming, pages 1–39, 2011.

[AC09] N. Ailon and B. Chazelle. The fast Johnson-Lindenstrauss transformation and approximate nearest neighbors. SIAM J. Computing, 39(1):302–322, 2009.

[AKL13] Dimitris Achlioptas, Zohar Karnin, and Edo Liberty. Near-optimal entrywise sampling for data matrices. In Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1, NIPS'13, pages 1565–1573, USA, 2013. Curran Associates Inc.

[AM07] D. Achlioptas and F. Mcsherry. Fast computation of low-rank matrix approximations. J. ACM, 54(2), April 2007.

[AZH16] Z. Allen-Zhu and E. Hazan. Variance reduction for faster non-convex optimization. In International Conference on Machine Learning, pages 699–707, 2016.

[AZL18] Zeyuan Allen-Zhu and Yuanzhi Li. Neon2: Finding local minima via first-order oracles. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, Advances in Neural Information Processing Systems 31, pages 3716–3726. Curran Associates, Inc., 2018.

[BC11] H. H. Bauschke and P. L. Combettes. Convex Analysis and Monotone Operator Theory in Hilbert Spaces. Springer-Verlag, New York, 2011.

[BC17] H. H. Bauschke and P. L. Combettes. Convex Analysis and Monotone Operator Theory in Hilbert Spaces. Springer-Verlag, New York, 2 edition, 2017.

[BCN18] R.I. Bot, E.R.. Csetnek, and D-K Nguyen. A proximal minimization algorithm for structured nonconvex and nonsmooth problems. arXiv preprint arXiv:1805.11056v1[math.OC], 2018.

[BD87] P. J. Brockwell and R. A. Davis. Time Series: Theory and Methods. Springer, 1987.

[BD09] T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. Harm. Anal., 27(3):265–274, 2009.

[Bec17] A. Beck. First-Order Methods in Optimization. MOS-SIAM Series on Optimization, 2017.

[Bec19] Stephen Becker. Matlab code for sketching. `https://github.com/stephenbeckr/randomized-algorithm-class/blob/master/Code/sketch.m`, 2019.

[BKM14] Adil Bagirov, Napsu Karmitsa, and Marko M. Mäkelä. Bundle Methods, pages 305–310. Springer International Publishing, Cham, 2014.

[BKM16] Dimitris Bertsimas, Angela King, and Rahul Mazumder. Best subset selection via a modern optimization lens. Ann. Statist., 44(2):813–852, 04 2016.

[BM99] V.s Borkar and Sanjoy Mitter. A strong approximation theorem for stochastic recursive algorithms. Journal of Optimization Theory and Applications, 100:499–513, 03 1999.

[Bro06] Petrus Broersen. Automatic autocorrelation and spectral analysis. Springer Science & Business Media, 2006.

[BST14] J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. Math. Prog., 146(1-2):459–494, 2014.

[BTW15] Jeffrey D. Blanchard, Jared Tanner, and Ke Wei. CGIHT: conjugate gradient iterative hard thresholding for compressed sensing and matrix completion. Information and Inference: A Journal of the IMA, 4(4):289–327, 11 2015.

[CB18] Xiang Cheng and Peter Bartlett. Convergence of langevin mcmc in kl-divergence. In Firdaus Janoos, Mehryar Mohri, and Karthik Sridharan, editors, Proceedings of Algorithmic Learning Theory, volume 83 of Proceedings of Machine Learning Research, pages 186–211. PMLR, 07–09 Apr 2018.

[CDHS18] Y. Carmon, J. Duchi, O. Hinder, and A. Sidford. Accelerated methods for nonconvex optimization. SIAM Journal on Optimization, 28(2):1751–1772, 2018.

[CDMI$^+$13] Kenneth L. Clarkson, Petros Drineas, Malik Magdon-Ismail, Michael W. Mahoney, Xiangrui Meng, and David P. Woodruff. The fast Cauchy transform and faster robust linear regression. In Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '13, pages 466–477, Philadelphia, PA, USA, 2013. Society for Industrial and Applied Mathematics.

[CDS98] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. SIAM J. Sci. Comput., 20:33–61, 1998.

[CDT19] Xi Chen, Simon S. Du, and Xin T. Tong. On Stationary-Point Hitting Time and Ergodicity of Stochastic Gradient Langevin Dynamics. arXiv e-prints, page arXiv:1904.13016, April 2019.

[CHS87] Tzuu-Shuh Chiang, Chii-Ruey Hwang, and Shuenn Jyi Sheu. Diffusion for global optimization in $\mathbb{R}^n$. SIAM Journal on Control and Optimization, 25(3):737–753, 1987.

[Cla05] Kenneth L. Clarkson. Subgradient and sampling algorithms for l1 regression. In Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '05, pages 257–266, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

[Cor11] Graham Cormode. Sketch techniques for approximate query processing. In Synposes for Approximate Query Processing: Samples, Histograms, Wavelets and Sketches, Foundations and Trends in Databases. NOW publishers, 2011.

[CRS17] F.E. Curtis, D.P. Robinson, and M. Samadi. A trust region algorithm with a worst-case iteration complexity of $\mathcal{O}(\epsilon^{\frac{3}{2}})$ for nonconvex optimization. Mathematical Programming, 162(1):1–32, Mar 2017.

[CRT06] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inform. Theory, 52(2):489–509, 2006.

[CW05] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. SIAM Multiscale Model. Simul., 4(4):1168–1200, 2005.

[CW12] Kenneth L. Clarkson and David P. Woodruff. Low Rank Approximation and Regression in Input Sparsity Time. Journal of the ACM, 63(6):1–45, 2012.

[DBLJ14] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 27, pages 1646–1654. Curran Associates, Inc., 2014.

[DJL^{+}17] Simon S Du, Chi Jin, Jason D Lee, Michael I Jordan, Aarti Singh, and Barnabas Poczos. Gradient descent can take exponential time to escape saddle points. In Advances in Neural Information Processing Systems, pages 1067–1077, 2017.

[DK17] Arnak S. Dalalyan and Avetik G. Karagulyan. User-friendly guarantees for the langevin monte carlo with inaccurate gradient, 2017.

[DMM08] Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Relative-error cur matrix decompositions. SIAM J. Matrix Anal. Appl., 30(2):844–881, September 2008.

[DPG^{+}14] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In Advances in Neural Information Processing Systems, pages 2933–2941, 2014.

[DRP^{+}16] Avinava Dubey, Sashank J. Reddi, Barnabás Póczos, Alexander J. Smola, Eric P. Xing, and Sinead A. Williamson. Variance reduction in stochastic gradient langevin dynamics. In Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS'16, page 1162–1170, Red Hook, NY, USA, 2016. Curran Associates Inc.

[DT20] Jing Dong and Xin T. Tong. Replica exchange for non-convex optimization, 2020.

[FHB01] M. Fazel, H. Hindi, and S. P. Boyd. A rank minimization heuristic with application to minimum order system approximation. In Proceedings of American Control Conference, volume 6, pages 4734–4739. IEEE, 2001.

[FR13] Simon Foucart and Holger Rauhut. A Mathematical Introduction to Compressive Sensing. Birkhäuser, New York, NY, 2013.

[FS02] Daan Frenkel and Berend Smit. Chapter 4 - molecular dynamics simulations. In Daan Frenkel and Berend Smit, editors, Understanding Molecular Simulation (Second Edition), pages 63 – 107. Academic Press, San Diego, second edition, 2002.

[GHJY15] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points — online stochastic gradient for tensor decomposition. In Peter Grünwald, Elad Hazan, and Satyen Kale, editors, Proceedings of The 28th Conference on Learning Theory, volume 40 of Proceedings of Machine Learning Research, pages 797–842, Paris, France, 03–06 Jul 2015. PMLR.

[GJP95] F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. Neural computation, 7(2):219–269, 1995.

[GK19] Ian Grooms and William Kleiber. Diagnosing, modeling, and testing a multiplicative stochastic gent-mcwilliams parameterization. Ocean Modelling, 133:1–10, 2019.

[GM91] Saul B. Gelfand and Sanjoy K. Mitter. Recursive stochastic algorithms for global optimization in $\mathbb{R}^d$. SIAM Journal on Control and Optimization, 29(5):999–1018, 1991.

[GM13] Ian Grooms and Andrew J Majda. Efficient stochastic superparameterization for geophysical turbulence. Proceedings of the National Academy of Sciences, 110(12):4464–4469, 2013.

[HB20a] Zhishen Huang and Stephen Becker. Perturbed proximal descent to escape saddle points for non-convex and non-smooth objective functions. In Luca Oneto, Nicolò Navarin, Alessandro Sperduti, and Davide Anguita, editors, Recent Advances in Big Data and Deep Learning, pages 58–77, Cham, 2020. Springer International Publishing.

[HB20b] Zhishen Huang and Stephen Becker. Spectral estimation from simulations via sketching, 2020.

[HMT11] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM Review, 53(2):217–288, 2011.

[JGN$^+$17] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M. Kakade, and Michael I. Jordan. How to escape saddle points efficiently. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pages 1724–1732, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[JL84] William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. Contemporary mathematics, 26(189-206):1, 1984.

[JRSPS16] S. J. Reddi, S. Sra, B. Poczos, and A.J. Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, Advances in Neural Information Processing Systems 29, pages 1145–1153. Curran Associates, Inc., 2016.

[JZ13] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 26, pages 315–323. Curran Associates, Inc., 2013.

[KLY18] Bobby Kleinberg, Yuanzhi Li, and Yang Yuan. An alternative view: When does SGD escape local minima? In Jennifer Dy and Andreas Krause, editors, Proceedings of the 35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research, pages 2698–2707, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[Kra40] H.A. Kramers. Brownian motion in a field of force and the diffusion model of chemical reactions. Physica, 7(4):284 – 304, 1940.

[KT75] Samuel Karlin and Howard M. Taylor. Chapter 7 - brownian motion. In Samuel Karlin and Howard M. Taylor, editors, A First Course in Stochastic Processes (Second Edition), pages 340 – 391. Academic Press, Boston, second edition edition, 1975.

[KW92] J. Kuczyński and H. Woźniakowski. Estimating the largest eigenvalue by the power and lanczos algorithms with a random start. SIAM Journal on Matrix Analysis and Applications, 13(4):1094–1122, 1992.

[KW11] Felix. Krahmer and Rachel. Ward. New and improved Johnson–Lindenstrauss embeddings via the restricted isometry property. SIAM Journal on Mathematical Analysis, 43(3):1269–1281, 2011.

[LAM12] LAMMPS benchmarks. `https://lammps.sandia.gov/bench.html#billion`, 2012. Accessed: 2020-02-13.

[LI06] Peter Lindstrom and Martin Isenburg. Fast and efficient compression of floating-point data. IEEE transactions on visualization and computer graphics, 12(5):1245–1250, 2006.

[Li19] Zhize Li. Ssrgd: Simple stochastic recursive gradient descent for escaping saddle points. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 1523–1533. Curran Associates, Inc., 2019.

[Lin14] Peter Lindstrom. Fixed-rate compressed floating-point arrays. IEEE transactions on visualization and computer graphics, 20(12):2674–2683, 2014.

[LPP$^+$19] Jason D. Lee, Ioannis Panageas, Georgios Piliouras, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. First-order methods almost always avoid strict saddle points. Math. Program., 176(1–2):311–337, July 2019.

[LSJR16] Jason D. Lee, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, 29th Annual Conference on Learning Theory, volume 49 of Proceedings of Machine Learning Research, pages 1246–1257, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR.

[LTP19] Puya Latafat, Andreas Themelis, and Panagiotis Patrinos. Block-coordinate and incremental aggregated proximal gradient methods for nonsmooth nonconvex problems, 2019.

[LY19] Yanli Liu and Wotao Yin. An envelope for davis—yin splitting and strict saddle-point avoidance. J. Optim. Theory Appl., 181(2):567–587, May 2019.

[Mah11] M. Mahoney. Randomized algorithms for matrices and data. Found. Trends Machine Learning, 3(2):123–224, 2011.

[MBP$^+$09] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In 2009 IEEE 12th International Conference on Computer Vision, pages 2272–2279, Sep. 2009.

[MBP14] J. Mairal, F. Bach, and J. Ponce. Sparse Modeling for Image and Vision Processing. now, 2014.

[MD09] Michael W. Mahoney and Petros Drineas. CUR matrix decompositions for improved data analysis. Proceedings of the National Academy of Sciences, 106(3):697–702, 2009.

[Mez07] F. Mezzadri. How to generate random matrices from the classical compact groups. Notices of the American Mathematical Society, 54(5):592–604, 2007.

[MG15] James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15, page 2408–2417. JMLR.org, 2015.

[MM13] Xiangrui Meng and Michael W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing, STOC '13, pages 91–100, New York, NY, USA, 2013. ACM.

[MMB14] Christopher A. Metzler, Arian Maleki, and Richard G. Baraniuk. From denoising to compressed sensing. IEEE Transactions on Information Theory, 62:5117–5144, 2014.

[MOJ18] Aryan Mokhtari, Asuman Ozdaglar, and Ali Jadbabaie. Escaping saddle points in constrained optimization. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, page 3633–3643, Red Hook, NY, USA, 2018. Curran Associates Inc.

[MSH02] J.C. Mattingly, A.M. Stuart, and D.J. Higham. Ergodicity for sdes and approximations: locally lipschitz vector fields and degenerate noise. Stochastic Processes and their Applications, 101(2):185 – 232, 2002.

[MT96] K. L. Mengersen and R. L. Tweedie. Rates of convergence of the hastings and metropolis algorithms. ANNALS OF STATISTICS, 24:101–121, 1996.

[MT09] Sean Meyn and Richard L. Tweedie. Markov Chains and Stochastic Stability. Cambridge University Press, USA, 2nd edition, 2009.

[MT20] Per-Gunnar Martinsson and Joel Tropp. Randomized numerical linear algebra: Foundations & algorithms. arXiv preprint arXiv:2002.01387, 2020.

[Mä02] Marko Mäkelä. Survey of bundle methods for nonsmooth optimization. Optimization Methods and Software, 17(1):1–29, 2002.

[Nes83] Y. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $\mathcal{O}(1/k^2)$. Doklady AN SSSR, translated as Soviet Math. Docl., 269:543–547, 1983.

[NP06] Yurii Nesterov and Boris T. Polyak. Cubic regularization of newton method and its global performance. Math. Program., 108:177–205, 2006.

[NT09] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. Appl. Comput. Harmon. Anal, 26:301–321, 2009.

[PAB17] F. Pourkamali-Anaraki and S. Becker. Preconditioned data sparsification for big data with applications to PCA and K-means. IEEE Trans. Info. Theory, 63(5):2954–2974, 2017.

[PAM+11] S. Plimpton, A.Thompson, S. Moore, A. Kohlmeyer, and R. Berger. Lammps dreiding example. `https://github.com/lammps/lammps/tree/master/examples/dreiding`, 2011.

[Pha09] Huyn Pham. Continuous-time Stochastic Control and Optimization with Financial Applications. Springer Publishing Company, Incorporated, 1st edition, 2009.

[Pli95] S. Plimpton. Fast parallel algorithms for short-range molecular dynamics. J. Comp. Phys., 117:1–19, 1995. `http://lammps.sandia.gov`.

[PM06] John G. Proakis and Dimitris K. Manolakis. Digital Signal Processing (4th Edition). Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006.

[Rap04] Dennis Rapaport. The art of molecular dynamics simulation. Cambridge university press, 2004.

[RG84] E. Runge and E. K. U. Gross. Density-functional theory for time-dependent systems. Phys. Rev. Lett., 52:997–1000, Mar 1984.

[RHS+16] Sashank J. Reddi, Ahmed Hefny, Suvrit Sra, Barnabás Póczós, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, page 314–323. JMLR.org, 2016.

[RL13] D. Romero and G. Leus. Compressive covariance sampling. In 2013 Information Theory and Applications Workshop (ITA), pages 1–8, 2013.

[RLBPS11] I. Rodríguez, O. Lehmkuhl, R. Borrell, and C.D. Pérez-Segarra. On DNS and LES of natural convection of wall-confined flows: Rayleigh-Bénard convection. In H. Kuerten, B. Geurts, V. Armenio, and J. Fr ohlich, editors, Direct and Large-Eddy Simulation VIII, volume 15 of ERCOFTAC Series, pages 389–394. Springer, 2011.

[RRT17] Maxim Raginsky, Alexander Rakhlin, and Matus Telgarsky. Non-convex learning via stochastic gradient langevin dynamics: a nonasymptotic analysis. In Satyen Kale and Ohad Shamir, editors, Proceedings of the 2017 Conference on Learning Theory, volume 65 of Proceedings of Machine Learning Research, pages 1674–1703, Amsterdam, Netherlands, 07–10 Jul 2017. PMLR.

[RT96a] G. O. ROBERTS and R. L. TWEEDIE. Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms. Biometrika, 83(1):95–110, 03 1996.

[RT96b] Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of langevin distributions and their discrete approximations. Bernoulli, 2(4):341–363, 1996.

[RZS$^+$17] Sashank J Reddi, Manzil Zaheer, Suvrit Sra, Barnabas Poczos, Francis Bach, Ruslan Salakhutdinov, and Alexander J Smola. A Generic Approach for Escaping Saddle points. arXiv e-prints, page arXiv:1709.01434, Sep 2017.

[Sar06] T. Sarlos. Improved approximation algorithms for large matrices via random projections. In 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), pages 143–152, Oct 2006.

[SFH$^+$18] Maher Salloum, Nathan D Fabian, David M Hensinger, Jina Lee, Elizabeth M Allendorf, Ankit Bhagatwala, Myra L Blaylock, Jacqueline H Chen, and Irina Templeton, Jeremy Aand Tezaur. Optimal compressed sensing and reconstruction of unstructured mesh datasets. Data Science and Engineering, 3(1):1–23, 2018.

[Sho62] N. Z. Shor. An application of the method of gradient descent to the solution of the network transportation problem. Materialy Naucnovo Seminara po Teoret i Priklad. Voprosam Kibernet. i Issted. Operacii, Nucnyi Sov. po Kibernet, Akad. Nauk Ukrain. SSSR, vyp, 1:9–17, 1962.

[SLQ$^+$19] Tao Sun, Dongsheng Li, Zhe Quan, Hao Jiang, Shengguo Li, and Yong Dou. Heavy-ball algorithms always escape saddle points. pages 3520–3526, 08 2019.

[SQW15] Ju Sun, Qing Qu, and John Wright. When are nonconvex problems not scary?, 2015.

[SR96] A. P. Scott and L. Radom. Harmonic vibrational frequencies: an evaluation of hartree-fock, møller- plesset, quadratic configuration interaction, density functional theory, and semiempirical scale factors. J. Phys. Chem., 100(41):16502–16513, 1996.

[STP17] L. Stella, A. Themelis, and P. Patrinos. Forward-backward quasi-Newton methods for nonsmooth optimization problems. Computational Optimization and Applications, 67(3):443–487, 2017.

[SW11] Christian Sohler and David P. Woodruff. Subspace embeddings for the l1-norm with applications. In Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing, STOC '11, pages 755–764, New York, NY, USA, 2011. ACM.

[TG07] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. IEEE Tran. Info. Theory, 53(12), 2007.

[TSJ+18] Nilesh Tripuraneni, Mitchell Stern, Chi Jin, Jeffrey Regier, and Michael I. Jordan. Stochastic cubic regularization for fast nonconvex optimization. In Proceedings of the 32Nd International Conference on Neural Information Processing Systems, NIPS'18, pages 2904–2913, USA, 2018. Curran Associates Inc.

[VELA+09] D. Varsano, D. A. Espinosa-Leal, X. Andrade, M. A. L. Marques, R. di Felice, and A. Rubio. Towards a gauge invariant method for molecular chiroptical properties in tddft. Phys. Chem. Chem. Phys., 11:4481–4489, 2009.

[Ver18] Roman Vershynin. High-Dimensional Probability: An Introduction with Applications in Data Science. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018.

[VGFP19] Emmanouil-Vasileios Vlatakis-Gkaragkounis, Lampros Flokas, and Georgios Piliouras. Efficiently avoiding saddle points with zero order methods: No gradients required. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 10066–10077. Curran Associates, Inc., 2019.

[Woo14] D. P. Woodruff. Sketching as a tool for numerical linear algebra. Foundations and Trends® in Theoretical Computer Science, 10(1–2):1–157, 2014.

[WT11] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML'11, page 681–688, Madison, WI, USA, 2011. Omnipress.

[WZ13] David P. Woodruff and Qin Zhang. Subspace Embeddings and ℓ_p-Regression Using Exponential Random Variables. In Shai Shalev-Shwartz and Ingo Steinwart, editors, COLT - The 26th Annual Conference on Learning Theory, volume 30 of JMLR Workshop and Conference Proceedings, pages 546–567, Princeton University, NJ, USA, June 2013. JMLR.org.

[XCZG18] Pan Xu, Jinghui Chen, Difan Zou, and Quanquan Gu. Global convergence of langevin dynamics based algorithms for nonconvex optimization. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, page 3126–3137, Red Hook, NY, USA, 2018. Curran Associates Inc.

[XJY18] Y. Xu, R. Jin, and T. Yang. First-order stochastic algorithnms for escaping from saddle points in almost linear time. arXiv preprint, 2018. arXiv:1711.01944v3 [math.OC].

[YB96] K. Yabana and G. F. Bertsch. Time-dependent local-density approximation in real time. Phys. Rev. B, 54:4484–4487, Aug 1996.

[ZLC17] Yuchen Zhang, Percy Liang, and Moses Charikar. A Hitting Time Analysis of Stochastic Gradient Langevin Dynamics. arXiv e-prints, page arXiv:1702.05575, February 2017.