Sampling Sparse Representations with Randomized Measurement Langevin Dynamics

KAFENG WANG, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, and University of Chinese Academy of Sciences
HAOYI XIONG, Baidu Inc.
JIANG BIAN, University of Central Florida
ZHANXING ZHU, Peking University
QIAN GAO, Baidu Inc.
ZHISHAN GUO, University of Central Florida
CHENG-ZHONG XU, University of Macau
JUN HUAN, StylingAI Inc.
DEJING DOU, Baidu Inc.

Stochastic Gradient Langevin Dynamics (SGLD) have been widely used for Bayesian sampling from certain probability distributions, incorporating derivatives of the log-posterior. With the derivative evaluation of the log-posterior distribution, SGLD methods generate samples from the distribution through performing as a thermostats dynamics that traverses over gradient flows of the log-posterior with certainly controllable perturbation. Even when the density is not known, existing solutions still can first learn the kernel density models from the given datasets, then produce new samples using the SGLD over the kernel density derivatives. In this work, instead of exploring new samples from kernel spaces, a novel SGLD sampler, namely, *Randomized Measurement Langevin Dynamics* (RMLD) is proposed to sample the high-dimensional sparse representations from the spectral domain of a given dataset.

An earlier version of this work has been accepted for publication as "SpHMC: Spectral Hamiltonian Monte Carlo" [57] in the Proceedings of The 33rd AAAI Conference on Artificial Intelligence (AAAI-19). The work was done when Kafeng Wang was a research intern at Baidu Inc. Correspondence should be directed to Dr. Haoyi Xiong by writing to xionghaoyi@baidu.com.
This work is supported in part by National Key R&D Program of China (Grant No. 2019YFB2102100 to Cheng-Zhong Xu and Grant No. 2018YFB1402600 to Haoyi Xiong), Science and Technology Development Fund of Macao S.A.R (FDCT) under Grant No. 0015/2019/AKP, the National Natural Science Foundation of China (No. 61806192), and Shenzhen Discipline Construction Project for Urban Computing and Data Intelligence.
Authors' addresses: K. Wang, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, and University of Chinese Academy of Sciences, Shenzhen, Guangdong, 518055, China; email: kf.wang@siat.ac.cn; H. Xiong, Big Data Lab, Baidu Inc., Haidian, Beijing, 100193, China; email: haoyi.xiong.fr@ieee.org; J. Bian, Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL, 32816; email: bjbj11111@knights.ucf.edu; Z. Zhu, School of Mathematical Sciences, Peking University, Haidian, Beijing, 100871, China; email: zhanxing.zhu@pku.edu.cn; Q. Gao, Business Group of Baidu Search, Baidu Inc., Haidian, Beijing, 100193, China; email: gaoqian05@baidu.com; Z. Guo, Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL, 32816; email: zsguo@ucf.edu; C.-Z. Xu, State Key Laboratory of Internet of Things for Smart City, Faculty of Science and Technology, University of Macau, Taipa, Macau, China; email: czxu@um.edu.mo; J. Huan, StylingAI Inc., Haidian, Beijing, 100193, China; email: lukehuan@shenshangtech.com; D. Dou, Big Data Lab, Baidu Inc., Haidian, Beijing, 100193, China; email: doudejing@baidu.com.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
1556-4681/2021/02-ART21 $15.00
https://doi.org/10.1145/3427585

Specifically, given a random measurement matrix for sparse coding, RMLD first derives a novel likelihood evaluator of the probability distribution from the loss function of LASSO, then samples from the high-dimensional distribution using stochastic Langevin dynamics with derivatives of the logarithm likelihood and Metropolis–Hastings sampling. In addition, new samples in low-dimensional measuring spaces can be regenerated using the sampled high-dimensional vectors and the measurement matrix. The algorithm analysis shows that RMLD indeed projects a given dataset into a high-dimensional Gaussian distribution with Laplacian prior, then draw new sparse representation from the dataset through performing SGLD over the distribution. Extensive experiments have been conducted to evaluate the proposed algorithm using real-world datasets. The performance comparisons on three real-world applications demonstrate the superior performance of RMLD beyond baseline methods.

CCS Concepts: • **Computing methodologies** → **Spectral methods**; *Learning linear models*; • **Mathematics of computing** → Exploratory data analysis;

Additional Key Words and Phrases: Hamiltonian Monte Carlo, compressive sensing, LASSO

ACM Reference format:
Kafeng Wang, Haoyi Xiong, Jiang Bian, Zhanxing Zhu, Qian Gao, Zhishan Guo, Cheng-Zhong Xu, Jun Huan, and Dejing Dou. 2021. Sampling Sparse Representations with Randomized Measurement Langevin Dynamics. *ACM Trans. Knowl. Discov. Data* 15, 2, Article 21 (February 2021), 21 pages.
https://doi.org/10.1145/3427585

1 INTRODUCTION

Statistical learning algorithms [50] can be improved, with better generalization performance, via Bayesian sampling from the posterior Oracles. For examples, authors in [28, 59, 61] proposed to improve the generalization error bounds of linear classifiers, such as Fisher's discriminant analysis [55, 56, 58, 61], Support Vector Machines [59] and regularized linear regression models [28], through re-sampling from a Markov-chain, which characterizes the distribution of multivariate datasets for training and testing. Furthermore, [53] extent the effects of such Bayesian sampling from linear models to the functional regression tasks based on Gaussian processes [36]. More recently, Du et al. [17] studied a neural network based approach to facilitate Bayesian sampling steps for the (approximate) inference of Deep Generative Models. All above research studies [17, 28, 59, 61] study the relevance of Bayesian sampling to the generalization of learning systems.

1.1 Backgrounds

Among a wide range of sampling algorithms, Markov Chain Monte Carlo (MCMC) techniques [3, 51] such as Hamiltonian Monte Carlo (HMC) via Stochastic Gradient Langevin Dynamics (SGLD) [11, 17] provide the supports or solutions to random sample generation for a wide range of machine learning tasks. For example, SLGD can provide extra data/parameters sampling from specific prior distribution for data augmentation task [4, 9] and Bayesian learning/inference [3, 52], where the new data improve the performance of the tasks. Specifically, given the gradient/derivative estimation of the log-density function, SGLD samples a sequence of new data from the target distribution through discretizing a second-order Langevin dynamics [52]. SGLD can draw samples from arbitrary probabilistic distributions with controllable biases using the derivative/gradient oracles of the distribution [30].

Estimating the gradient/derivative of the density function is a key procedure in SGLD method and in many applications the density function unknown or not differentiable. Thus, before SGLD can generate new samples and improve the learning performance on a specific dataset, it is necessary to first well-build a density or approximate the corresponding derivative based on the training datasets. A typical way to handle the case is using Kernel Density Estimation [10], a non-parametric method to estimate the probability density function. Such kind of solutions namely *Kernel Sequential Monte Carlo* [42] or *Kernel HMC* [45] can be used to learn the distribution form the target datasets, with the selection of appropriate Kernels, such as Gaussian Kernel. Then SGLD is able to draw new instances form this modeled distribution.

1.2 Technical Challenges

Although such kernel density estimation can provide close approximations of the target probability density, a couple of factors may lead to obstacles when using kernel density estimation in practice:

(1) *Kernel Distance Measurement.* One of the traditional methods to evaluate the effectiveness of the modeled density is to calculate the geometric distance between the modeled density and the oracle one. However, it is inapplicable for some data distributions with uncommon structures, such as high-resolution image data. To this point, without a carefully hand-chosen and data-specific kernel function, such geometric distance-based evaluation often leads to unstable or intractable kernel density functions. We also need to carefully choose the kernel according to the datasets, which makes the kernel density estimation unstable and intractable.
(2) *Curse of Dimensionality.* The high-dimensional data distribution is hard to approximate by the kernel density estimation [6], and the inaccurate modeled distribution cannot give the proper guide of SGLD when generating the sequence of new samples. Scott et al. [43] demonstrated a progressive deterioration of multivariate kernel density estimation as the dimension increases by showing that an increase in sample size is required to attain an equivalent amount of accuracy.
(3) *Computation Complexity.* Not only the accuracy suffers from the high dimensions, but the computation cost also increases significantly [40, 41]. In addition, it is quite time consuming to select an optimal bandwidth for the kernel when the datasets are of high dimension with tremendous instances [37, 49]. For example, the images in the MNIST dataset are with $28 \times 28 = 784$ dimensions and the dataset contains 60,000 images. Setting up an optimal kernel for SGLD is time-consuming. Besides the bandwidth selection, it is also suffering the heavy load for calculating the derivatives of the kernel density functions especially in high-dimension space, i.e. mixing computation for the first derivatives of LogSumExp functions and extra large matrix/vector multiplications.

There is a pressing need for methods to replace kernel density estimation and to provide accurate derivative evaluation of probability density. To lower the computational complexity to compute the derivative of kernel density functions, Sasaki et al. [40, 41] proposed the direct density derivative estimator that learns a regression model through sampling from fitted kernels, then predicts the derivatives using the regression model. With such methods, one can draw samples from the distribution with low computational complexity. However, such methods still cannot handle the dimensionality curse of kernel methods, which may cause significant performance degradation [44] due to the divergence between fitted and ground truth distributions.

1.3 Summary of Contributions

From the previous discussions, it is clear that a new probabilistic model is required to define (i) the prior probability of samples and characterize (ii) the likelihood from the given dataset. In this work, we propose a novel SGLD sampler, namely *Randomized Measurement Langevin Dynamics* (`RMLD`). Instead of exploring in the kernel spaces, `RMLD` samples the high-dimensional sparse representations from the given dataset. Inspired by compressed sensing [14, 29], `RMLD` assumes all samples in the datasets are the low-dimensional measurements of certain high-dimensional sparse

Table 1. Notations for RMLD

Notations	Description
$\mathbf{A}$	Measurement matrix
$\mathbf{Y}$	An (Low-dimensional) observation
$\mathbf{X}$	The sparse representation that potentially exists
$\mathbb{P}(\mathbf{X})$	Prior probability distribution
$\mathbb{P}(x\|\mathbf{X})$	Likelihood of x based on the parameter $\mathbf{X}$
$\mathcal{D}$	The set of original data samples
T	Total number of iterations for SGLD exploration
K	The number of generated samples required
m	Size of mini-batch ($m \geq 1$)
η	Step-size for SGLD dynamics discretization
λ	A tuning parameter for sparsity induction
`discount`	Factor controls the "quality" of sample generation.
$\mathbf{X}^K$	The sequence of K generated samples

vectors, based on a certain dictionary or namely measurement matrix [46]. With the distribution of the given dataset, a continuous probability distribution exists in such a high-dimensional space. Thus, each sample drawn from the high-dimensional distribution corresponds to a potential low-dimensional measurement via the same measurement matrix (and reverse).

Specifically, with a measurement matrix for compressed sensing, e.g., a random Gaussian matrix [8], RMLD first proposes a novel log-density evaluator of the probability distribution that characterizes the given datasets. Such log-density evaluator is derived from the loss function of Least Absolute Shrinkage and Selection Operator (LASSO) [29, 47], with low complexity solutions for exact derivative computation. Then, given the log-density (derivative) evaluator, RMLD can draw high-dimensional samples from the distribution using a second-order stochastic Langevin dynamics with a Metropolis–Hastings (MH) sampler.

Finally, with the high-dimensional samples drawn, two types of applications can be supported as follows. (1) Using the random measurement matrix, one can project the high-dimensional samples to their low-dimensional samples. In this way, RMLD performs as a generator that reproduces new datums from the datasets with intractable likelihoods. (2) In addition, one can directly use generated the high-dimensional samples to augment the machine learning algorithms with sparse coding. For example, some image classification tasks can be improved using compressed sensing, while they can further be improved by incorporating the high-dimensional data generation based on the same measurement matrix. Extensive experiments are conducted to evaluate the proposed algorithms with the two applications using four real-world datasets including MNIST, Fashion MNIST, Quick Draw and EMNIST. The performance comparisons on the two applications demonstrate the excellence of RMLD beyond baseline methods.

2 RELATED WORKS: BAYESIAN SAMPLING WITH SLGD

In this section, we first introduce the related works of this work, then review the most relevant work. The notations used in this article have been listed in Table 1.

2.1 SGLD for Bayesian Sampling

For a great number of machine learning tasks, sampling from a given distribution is frequently required. Among a wide range of samplers, MCMC [3] comprises a class of general-purpose sampling algorithms with fast mixing properties. Through incorporating a Markov chain derived from the distribution, MCMC can sample from the distribution by traversing the Markov chain via a number of steps. To construct the Markov chain from a distribution, the stochastic gradient Hamiltonian Monte Carlo (SGHMC) [11, 30] has been proposed to use the discrete-time Langevin dynamics [52] coupled by the gradient flow of logarithm density of the probability distribution [5]. The random jumps with the Langevin dynamics help SGHMC traverse the Markov chain and obtain samples.

2.1.1 SGLD Continuous-time Process. Specifically, given the posterior density model $\mathrm{P}(\mathbf{X}) \propto \exp(-U(\mathbf{X}))$ of the desired distribution, SGLD draws a sequence of samples, e.g., $\mathbf{X}^1, \mathbf{X}^2, \ldots,$ from the distribution $\mathrm{P}(\mathbf{X})$ through discretizing the Langevin dynamics as follows:

$$\begin{cases} \mathrm{d}\mathbf{X} = \mathbf{r}\,\mathrm{d}t \\ \mathrm{d}\mathbf{r} = -\nabla U(\mathbf{X})\mathrm{d}t - \mathbf{B}\mathbf{r}\mathrm{d}t + \mathcal{N}(\mathbf{0}, 2\mathbf{B}\mathrm{d}t), \end{cases} \tag{1}$$

where r is a vector referring the momentum of the dynamics, $\mathbf{B}$ refers to a constant matrix that controls the influence of noise. The noise term $\mathcal{N}(0, 2\mathbf{B}\mathrm{d}t)$ generates a Gaussian noise i.i.d from $\mathcal{N}(\mathbf{0}, 2\mathbf{B})$ over the time t, so as to incorporate randomness during the sampling procedure. To obtain a sample, one can theoretically set the initial state of dynamics as the white noise, i.e., $\mathbf{X}(0) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $r(0) = \mathbf{0}$.

2.1.2 SGLD Algorithms. To implement the SGLD for as a Bayesian sampler, certain discretization form of this dynamics can be further developed through Euler-Maruyama scheme. The algorithm can be designed as an iterative process on discrete time t, such that

$$\begin{cases} \mathbf{X}_{t+1} = \mathbf{v}_t + \eta\mathbf{v}_t \\ \mathbf{v}_{t+1} = \mathbf{v}_t - \gamma\eta\mathbf{v}_t - \alpha\nabla U(\mathbf{X}_t)\mathrm{d}t + \sqrt{2\gamma\alpha}\epsilon_t, \end{cases} \tag{2}$$

with the changing rate of momentum η and the random Gaussian white noise ϵ_t at time t. For every iteration (or every certain number of iterations), the algorithm draws a sample from the distribution. In this way, the SGLD can iteratively sample a sequence of new instances from the target distribution. From time $t = 0$, we can consider the output $\mathbf{X}_0,\ \mathbf{X}_1,\ \mathbf{X}_2 \ldots,$ as a sequence of samples drawn from a posterior distribution with log-density $\log P(\mathbf{X}) \propto U(\mathbf{X})$.

For initialization prior to the sampling process, one can set the value of the noise as $\mathbf{X}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and momentum as $\mathbf{v}_0 = \mathbf{0}$. The trajectories traversed by the above algorithm is considered as a discrete-time approximation to SGLD which would asymptotically converge to the desired distribution [30].

2.2 Variants of SGLD for Improved Sampling

Given the procedure of SLGD, a number of pioneering studies have been done to improve the method. We sort them into four categories:

2.2.1 SGLD Dynamics and Averaging. Instead of using the second-order Langevin dynamics, the first-order dynamics, such as the one that stochastic gradient descent (SGD) algorithm behaves as, has been studied for approximately variational inference [32] with a sampling-based procedure. The purpose of these dynamics is to minimize the Kullback–Leibler divergence between the stationary distribution of its continuous-time process and the posterior, while the sampled trajectories can be viewed as an approximation to the inferences. Compared to SGD, SGLD leverages a momentum term to expand the regions that the process explores. In addition to using alternative dynamics, averaging schemes are frequently used to further accelerate the search. Polyak and Juditysky [35] first proved the optimality of averaging for SGD-based inference. While Polyak average requires the storage of the whole trajectory traversed by SGD, the average based on sliding windows helps to lower the space complexity while also ensuring good performance [33]. Furthermore [1] proposed to use a coupled dynamics for Bayesian inference of matrix factorization. Note that averaging has been frequently used to accelerate Bayesian inference (optimization) but rarely for data generation.

2.2.2 Conditioning Gradients. The (vanilla) SGLD listed in Equation (1) might be significantly influenced by the noise terms incorporated. To control the influence of Noise, Stochastic Gradient Fisher Scoring (SGFS) [2] has been firstly proposed to use a positive-definite matrix **H** to precondition the dynamics as follow:

$$\begin{cases} \mathrm{d}\mathbf{X} = \mathbf{H}\mathbf{r}\mathrm{d}t \\ \mathrm{d}\mathbf{r} = -\nabla U(\mathbf{X})\mathrm{d}t - \mathbf{B}\mathbf{H}\mathbf{r}\mathrm{d}t + \mathcal{N}(\mathbf{0}, 2\mathbf{B}\mathrm{d}t). \end{cases} \tag{3}$$

The traditional SGLD can be viewed as a special case of SGFS with $\mathbf{H} = \mathbf{I}$. Please refer to Section 5.1 of [33] for the analysis. Further [27, 34] preconditioned SGLD for Bayesian inference of deep networks.

2.2.3 Stochastic Gradient Approximation. As was mentioned, the derivative evaluator of the log-posterior density is indispensable for gradient-based sampling, while the gradient computation for large high-dimensional datasets is quite time consuming. First of all, to lower the complexity with increasing size of datasets, noisy gradient estimation with mini-batch of samples has been widely used in [11, 25, 30, 44]. In addition to the direct estimation, some regression-based methods have been studied that can learn to predict the gradient [40, 41]. Furthermore [19] studied to use a conjugate gradient for sampling from a Gaussian process with an unbiased solver. Most recent work replaces the common gradients with Fractional-order derivatives [60], so as to accelerate the Bayesian inference with a log-concave density.

2.2.4 MH Correction. The sequence sampled by MCMC can be corrected using MH algorithm [23], which rejects the newly generated sample, if the new sample is not "with respect to the current sample, according to the distribution." Tons of work have been done to use MH or rejection-based method to correct the sampling sequence [3, 31]. The most recent work [18] proved that bias caused by the step size can be appropriately corrected by using MH rejection with strong theoretical consequences under the log-concave assumption.

A comprehensive survey has been made in [33]. While most of the above work intent to enable Bayesian inference with samplers, this article aims at using gradient-based MCMC to generate datums through sampling from distributions.

2.3 Comparison to Previous Works

As stated in the introduction section, our work studies a new problem that intends to draw samples of sparse representation from the distribution of the given dataset using SGLD. In terms of a research problem, our work follows the sparsity settings of compressive sensing [16] (please see also in Section 3), while especially focusing on generating new samples in the high-dimensional space, rather than reconstructing the exact sparse representations for each existing datum in the dataset.

In terms of methodologies, the most relevant work to us is Kernel HMC [44, 45], which studies a similar research problem and leverages SGLD as well. Compared to Kernel HMC, which uses Kernels to model to the distribution of the given dataset, our work adopts a probabilistic model derived from Bayesian compressive sensing [8, 24] to model the distribution. In this way, `RMLD` defacto models the posterior distribution using a Gaussian likelihood with Laplacian prior. Note that compared to the derivatives of Bayesian compressive sensing, estimating the gradients of

Kernel is quite time consuming (please see also in introduction section for detailed discussion and references). In this way, RMLD lowers the computational complexity of the gradient estimation while making sense in terms of the probability model.

Please note that this article is derived from our conference paper [57]. On top of our conference version, we have made tremendous efforts in making the journal version standout as a complete and comprehensive work. In this article, we have included new contributions as follow. We revise the introductions section so as to include more discussion on our motivations, intuitions, and possible intellect merit of this work. In Section 2, we reorganize the introduction to the related work, with more comprehensive details presented. In Section 3, we included additional technical backgrounds with preliminary models of our research, where a clear problem formulation is given based on the aforementioned models, assumptions and settings. In terms of methodology, we extend our algorithm to incorporate an additional tuning parameter, namely, discount factor, which controls the quality of generated data. Experiments have been done with one additional dataset Quick Draw [22], which is artificially synthesized as vector plots by Recurrent Neural Networks (RNNs). More discussion on experimental results and the overall article has been made by the end of this version. The additional materials provided in the journal version yield audiences possibility to further understand the theoretical advantage of the proposed algorithms.

3 BACKGROUNDS MODELS AND PROBLEMS

In this section, we present the backgrounds with preliminary models of our work, then formulate the research problem.

3.1 Compressive Sensing

Compressive sensing or compressed sensing originally refers to a signal reconstruction technique that intends to reconstruct the sparse representation of a signal from its low-dimensional measurement, through solving an undetermined linear system [14, 15].

Given a $d \times p$ measurement matrix $\mathbf{A}$ (where $p \gg d$) and a p-dimensional unknown sparse signal $\mathbf{X}^*$, one is assumed to obtain a noisy measurement of the signal such that

$$\mathbf{Y} = \mathbf{A}\mathbf{X}^* + \epsilon, \tag{4}$$

where ϵ refers to a noise following certain structural assumption [7, 8, 46, 48]. Compressive sensing makes it possible to exactly reconstruct the sparse signal $\mathbf{X}^*$ from $\mathbf{Y}$ and $\mathbf{A}$, under so-called k-sparsity and restricted isometric property assumptions [8].

3.2 ℓ_1-Penalized Least Square Estimation for Compressive Sensing

To obtain the sparse reconstruction, compressive sensing techniques offered several effective estimators to invert the process listed in Equation (4) via solving the undetermined linear system with the "fat" matrix $\mathbf{A}$. All in all, under mild conditions on ϵ, the sparse signal reconstruction problem can be reformulated as finding the ℓ_1-norm sparsest solution to the linear system [16], such tat

$$\mathbf{X}^{\ell_1} \leftarrow \underset{\mathbf{X}\in\mathbb{R}^p}{\operatorname{argmin}} \|\mathbf{X}\|_1 \text{ s.t. } \|\mathbf{A}\mathbf{X} - \mathbf{Y}\|_2 \leq \varepsilon, \tag{5}$$

where ε refers to the tolerable error of reconstruction. To compute the solution, the ℓ_1-norm regularized least square estimator (or namely, LASSO, in high–dimensional statistics) is frequently used under K.K.T condition, such that

$$\hat{\mathbf{X}}^{\ell_1} \leftarrow \underset{\mathbf{X}\in\mathbb{R}^p}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{A}\mathbf{X} - \mathbf{Y}\|_2 + \lambda\|\mathbf{X}\|_1, \tag{6}$$

where λ refers to a tuning parameter with respect to the potential ε.

3.3 Bayesian Compressive Sensing

Bayesian Compressive Sensing [24, 29] interpreted the estimator in Equation (6) as a Bayesian estimation on the potential structure of such sparse signal via maximum a posteriori estimation, where the probability of sparse signal $\mathbf{X}^* = \mathbf{X}$ with the noisy measurement $\mathbf{Y}$ and the given measurement matrix $\mathbf{A}$ is modeled as follows

$$\log P(\mathbf{X}^* = \mathbf{X}|\mathbf{Y}, \mathbf{A}) \propto \frac{1}{2}\|\mathbf{AX} - \mathbf{Y}\|_2 + \lambda\|\mathbf{X}\|_1. \tag{7}$$

Specifically, $\frac{1}{2}\|\mathbf{AX} - \mathbf{Y}\|_2$ is proportional to the log-likelihood term corresponding a Gaussian reconstruction error based on the measurements and $\lambda\|\mathbf{X}\|_1$ refers to the log-prior probability term based on a $\frac{1}{\lambda}$-width Laplacian distribution [29].

3.4 Problem Formulation

As was stated in introduction section, the problem of our study is to generate samples of high-dimensional sparse representations from a given dataset, so as to augment the machine learning tasks on top of the sparse representation (with more data). In this way, we summarize the problem setting of our research as following assumptions.

Assumption 1 (Compressed Realization). *Given an (unknown) p-dimensional probability distribution $\mathcal{P}$, one first draws n random i.i.d samples $y_1, y_2, \ldots, y_n$ from $\mathcal{P}$. Suppose there exists a $d \times p$ measurement matrix $\mathbf{A}$ and $d \ll p$. With the measurement matrix $\mathbf{A}$ and $y_1, y_2, \ldots, y_n$, one can obtain n d-dimensional (noisy) observations, such that $\forall y_i$*

$$x_i = \mathbf{A}y_i + \varepsilon_i \ \ \textit{and} \ \ \varepsilon_i \sim \mathcal{N}(\mathbf{0}, \delta^2\mathbf{I}), \tag{8}$$

where δ controls the variance of white noise.

Problem. Given the matrix $\mathbf{A}$ and the d-dimensional observation set $\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$, our problem is to generate a sequence of *identical* and *independent* p-dimensional *random* vectors $\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^K$ that maximize the likelihoods on $\mathcal{P}$, such that

$$\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^K \leftarrow \underset{x^1, x^2, \ldots, x^K}{\mathbf{argmax}} \prod_{t=1}^{K} \mathcal{P}(x^t), \tag{9}$$

where $\mathcal{P}(x^t)$ refers to the probability of the t^{th} generated sample based on the distribution $\mathcal{P}$. In following sections, we introduce our proposed algorithm to solve above problem.

4 RMLD: SPECTRAL SAMPLER USING RANDOMIZED MEASUREMENT LANGEVIN DYNAMICS

In this work, we propose RMLD—*the Spectral Sampler using Randomized Measurement Langevin Dynamics* for HMC. In this section, we first present the overall algorithm design of RMLD. Then, we introduce the detailed implementation of the algorithms.

4.1 The Algorithmic Framework

Given a dataset and a random measurement matrix for sparse reconstruction, RMLD first models the posterior distribution of data spectrum using the dictionary-based Gaussian likelihoods with a Laplacian prior, then adopt SGHMC to draw samples from the distribution of spectrum.

4.1.1 Data Input. Algorithm 1 shows the overall design of RMLD, where the input of algorithms include (i) $\mathcal{D}$–the set of original data samples, (ii) $\mathbf{A}$ the randomized measurement matrix for sparse reconstruction in spectral spaces, (iii) T total number of iterations for SGLD exploration, (iv) K the number of generated samples required, (v) $m \geq 1$ the size of mini-batch, (vi) $\eta > 0$ the step-size for SGLD dynamics discretization, (vii) $\lambda > 0$ the parameter for ℓ_1 regularization for posterior modeling, and (viii) discount > 0 a factor that control the "quality" of sample generation. The algorithm outputs $\mathbf{X}^1,\ \mathbf{X}^2, \ldots, \mathbf{X}^K$–the sequence of K generated samples in the spectral space of $\mathcal{D}$. The overall complexity of this algorithm is $O(Km)$, while some samples may repeat in the generated sequence (i.e., # of unique samples $\leq K$). The algorithm is designed as follows.

4.1.2 Initialization. In line 1 of Algorithm 1, RMLD initializes the whole sampling procedure by defining $\mathbf{X}^0$. A simple way to initialize is using the white noise i.i.d drawn from $\mathcal{N}(0, I)$. Yet another method would first draw an i.i.d sample from the dataset i.e., $z \overset{i.i.d}{\sim} \mathcal{D}$, then performs LASSO to recover its high-dimensional sparse representation using z and the dictionary $\mathbf{A}$, such that

$$\mathbf{X}^0 \leftarrow \underset{x}{\operatorname{argmin}} \frac{1}{2}\|z - \mathbf{A}x\|_2^2 + \lambda\|x\|_1 \ . \tag{10}$$

In this way, we can start the sequence of sampling from the sparse representation of a (known) random sample.

ALGORITHM 1: RMLD: Spectral Sampler using Randomized Measurement Langevin Dynamics

```
1:  procedure RMLD($\mathcal{D}$, A, $T$, $K$, $m$, $\eta$, $\lambda$, discount)
2:      Initialization: $\mathbf{X}^0$
3:      /*Sequence Sampling*/
4:      for all $k = 1, 2, 3, \ldots, K$ do
5:          $\mathbf{r}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
6:          $\mathbf{X}_0 \leftarrow \mathbf{X}^{k-1}$
7:          /* Gradient-based sampling with $T$ Steps*/
8:          for all $t = 1, 2, 3, \ldots, T$ do
9:              $\mathbf{g}_t \leftarrow \mathrm{GradEst}(\mathbf{X}_{t-1}, \mathcal{D}, \mathbf{A}, m, \lambda)$
10:             $\mathbf{r}_t \leftarrow \mathbf{r}_{t-1} - \eta\mathbf{g}_t - \eta\mathbf{r}_{t-1}$
11:             $\mathbf{X}_t \leftarrow \mathbf{X}_{t-1} + \eta\mathbf{r}_t$
12:         end for
13:         /* Discounted Metropolis-Hastings Correction*/
14:         $\mathbf{X}^k \leftarrow \mathrm{MH}(\mathbf{X}^{k-1}, \mathbf{X}_T, T, \mathcal{D}, \lambda, \eta,$ discount)
15:     end for
16:     return $\mathbf{X}^1, \mathbf{X}^2, \mathbf{X}^3, \ldots, \mathbf{X}^K$, discount
17: end procedure
```

4.1.3 Sampling Sparse Representation from Spectral Distributions. In Lines 4–15 of Algorithm 1, RMLD leverages a loop of K iterations to generate a sequence of K samples, where each iteration performs an iterative process of T steps to obtain the next sample generated in the sequence. Specifically, in each iteration (e.g., the k^{th} iteration), RMLD first setups $\mathbf{r}_0$ using the white noise and $\mathbf{X}_0$ with the previous sampling result $\mathbf{X}^{k-1}$ in the sequence, so as to initialize $\mathbf{r}$ and $\mathbf{X}$ of the SGLD dynamics. Then, RMLD walks T steps of discrete-time SGLD with noisy gradient estimator (in line 9) to reach the potentially next sample $\mathbf{X}_T$. Further, a rejection-based MH correction is given to decide whether (1) to accept $\mathbf{X}_T$ as $\mathbf{X}^k$ or (2) to reject $\mathbf{X}_T$ while reusing $\mathbf{X}^{k-1}$ as $\mathbf{X}^k$. Note that a discount factor has been used here to adjust the bar of rejection. The design and implementation of the noisy gradient estimator and MH correction would be introduced in the next sections.

Finally, as shown in line 16 of the Algorithm 1, RMLD outputs the sequence as the sampling results. Note that such sampling procedure depends on the initial status such as $\mathbf{X}^0$. Then it is better to repeat the whole algorithm multiple times while setting a relevant small K for each run, to balance the bias and the time complexity.

ALGORITHM 2: Noisy Gradient Estimation for ℓ_1-Penalized Log-Likelihood with Random Mini-Batch

```
1: procedure GradEst($\mathbf{X}$, $\mathcal{D}$, $\mathbf{A}$, $m$, $\lambda$)
2:     /*Noisy gradient estimation*/
3:     $M \leftarrow$ draw $m$ i.i.d samples from $\mathcal{D}$
4:     $\mathbf{g}_m \leftarrow \mathbf{A}^\top\mathbf{A}\mathbf{X} - \frac{1}{m}\sum_{\forall x \in M}\mathbf{A}^\top x + \lambda \cdot \mathrm{sign}(\mathbf{X})$
5:     return $\mathbf{g}_m$
6: end procedure
```

4.1.4 Data Reconstruction via Compression. Note that the output of Algorithm 1 $\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^K$ are a sequence of sparse representations based on the dictionary $\mathbf{A}$. To obtain the generated datums, which can be observed in low dimension, one uses $\mathbf{A}$ to compress the sampling results, such that for $\forall\ \mathbf{X}^k$ for $1 \leq k \leq K$ one can compute as the k^{th} generated sample of low-dimensional observations. Note that some simple soft-thresholding strategies can be applied here to denoise the data (e.g., images) and improve the reconstruction.

4.2 Gradient Estimation for Negative Log-Density

Algorithm 2 demonstrates the design of (noisy) gradient estimator used in the SGLD dynamics and sampling. Specifically, we define the posterior probability distribution of the high-dimensional sparse representation [29] of the dataset $\mathcal{D}$ using Bayes' theorem as follow:

$$\mathbb{P}(\mathbf{X}|\mathcal{D}) \propto \mathbb{P}(\mathbf{X})\mathbb{P}(\mathcal{D}|\mathbf{X}) = \mathbb{P}(\mathbf{X}) \prod_{\forall x \in \mathcal{D}} \mathbb{P}(x|\mathbf{X}), \tag{11}$$

where $\mathbb{P}(\mathbf{X})$ refers to the prior distribution and $\mathbb{P}(x_i|\mathbf{X})$ is the likelihood of $x \in \mathcal{D}$ given the high-dimensional sparse representation. Then, we define the prior distribution of $\mathbf{X}$ as the Laplacian distribution such that

$$\mathbb{P}(\mathbf{X}) \propto \exp(-\lambda\|\mathbf{X}\|_1). \tag{12}$$

Further, the likelihood of low-dimensional observation $\forall x \in \mathcal{D}$ given the high-dimensional sparse representation $\mathbf{X}$ can be modeled using Gaussian distribution, such that

$$\mathbb{P}(x|\mathbf{X}) \propto \exp\left(-\frac{1}{|\mathcal{D}|}\|x - \mathbf{A}\mathbf{X}\|_2^2\right). \tag{13}$$

In this way the negative log-density can be modeled as

$$-\log\ \mathbb{P}(\mathbf{X}|\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{\forall x \in \mathcal{D}} \|x - \mathbf{A}\mathbf{X}\|_2^2 + \lambda\|\mathbf{X}\|_1 + c, \tag{14}$$

where c is a constant. We find above function above actually averages the LASSO losses using all samples in the dataset $\mathcal{D}$. The gradient of the negative log-density can be written as

$$\nabla\left(-\log\ \mathbb{P}(\mathbf{X}|\mathcal{D})\right) = \frac{1}{|\mathcal{D}|}\sum_{\forall x \in \mathcal{D}} \mathbf{A}^\top(\mathbf{A}\mathbf{X} - x) + \lambda \cdot \mathrm{sign}(\mathbf{X}), \tag{15}$$

where $\mathrm{sign}(\cdot) \rightarrow \{\pm 1\}$ is the signal function. According to Lines 3–4 of Algorithm 2, the vector $\mathbf{g}_m$ indeed is a noisy estimation of the gradient with a mini-batch of m random samples drawn from $\mathcal{D}$, where the noise in the gradient estimation can be well controlled by m.

4.3 Discounted MH Correction

Algorithm 3 presents the design of the MH correction mechanism used in Line 14 of Algorithm 1. The major input of this algorithm includes (i) the previous generated sample $\mathbf{X}$ (i.e., $\mathbf{X}^k$ in Algorithm 1), (ii) the current approaching sample $\mathbf{Z}$ (i.e., $\mathbf{X}_T$ in Algorithm 1), (iii) the number of iterations required T, and (iv) the step size of SGLD η. RMLD aims at correcting the bias caused by the discretization of dynamics with step size η. The output of RMLD would assign to the newly generated sample $\mathbf{X}^k$.

ALGORITHM 3: Discounted Metropolis–Hastings Correction

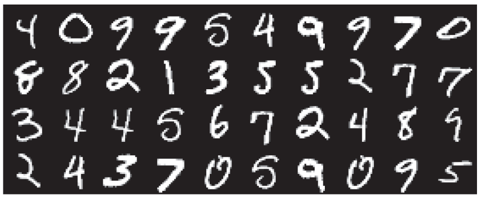
```
1: procedure MH(X, Z, T, $\mathcal{D}$, $\lambda$, $\eta$, discount)
2:     /*Function and Derivative Evaluation*/
3:     $f_x \leftarrow \frac{1}{2|\mathcal{D}|}\sum_{\forall x\in\mathcal{D}} \|\mathbf{AX}-x\|_2^2 + \lambda\|\mathbf{AX}\|_1$
4:     $f_z \leftarrow \frac{1}{2|\mathcal{D}|}\sum_{\forall x\in\mathcal{D}} \|\mathbf{AZ}-x\|_2^2 + \lambda\|\mathbf{AZ}\|_1$
5:     $\nabla f_x \leftarrow \frac{1}{|\mathcal{D}|}\sum_{\forall x\in\mathcal{D}} \mathbf{A}^\top(\mathbf{AX}-x) + \lambda\cdot\text{sign}(\mathbf{X})$
6:     $\nabla f_z \leftarrow \frac{1}{|\mathcal{D}|}\sum_{\forall x\in\mathcal{D}} \mathbf{A}^\top(\mathbf{AZ}-x) + \lambda\cdot\text{sign}(\mathbf{Z})$
7:     $\Delta \leftarrow \eta T$
8:     /*Transition Probability Estimation*/
9:     $\alpha = \min.\left\{1, \frac{\exp(-f_z - \|\mathbf{X}-\mathbf{Z}+\Delta\cdot\nabla f_z\|_2^2/4\Delta)}{\exp(-f_x - \|\mathbf{Z}-\mathbf{X}+\Delta\cdot\nabla f_x\|_2^2/4\Delta)}\right\}$
10:    $\gamma \overset{i.i.d}{\sim} \text{Uniform}[0,1]$
11:    if $\alpha >$ discount $\cdot\,\gamma$ then
12:        return Z /* Accept Z*/
13:    else
14:        return X /* Reject Z*/
15:    end if
16: end procedure
```

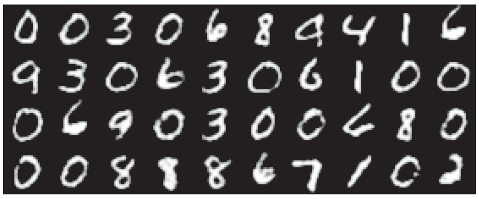
This algorithm first estimates the *transition probability* from $\mathbf{X}$ to $\mathbf{Z}$ and reverse. Then, the algorithm outputs $\mathbf{X}$ or $\mathbf{Z}$ depending on whether $\mathbf{X}$ would be accepted or rejected respectively. In Lines 2–7 of Algorithm 3, RMLD evaluates the negative log-density and the derivatives on $\mathbf{X}$ and $\mathbf{Z}$ based on the whole dataset $\mathcal{D}$ respectively. Then it estimates the ratio α between the transition probability from $\mathbf{X}$ to $\mathbf{Z}$ and the transition probability from $\mathbf{Z}$ to $\mathbf{X}$ (in line 9). Such ratio is upper-bounded by 1. Then, like other MH algorithms, RMLD randomly draws $\gamma \sim \text{Uniform}[0,1]$ and compares γ to α to make the decision for acceptance/rejection. Note that the newly approaching sample $\mathbf{Z}$ would be rejected when the ratio α, between the transition probabilities from $\mathbf{X}$ to $\mathbf{Z}$ and reverse, is not greater than the discounted random number $\texttt{discount}\cdot\gamma$.

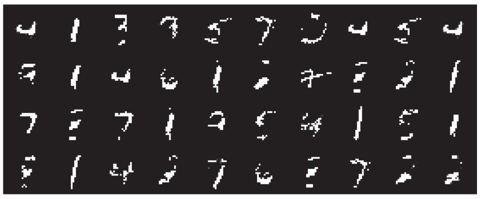
Discounting trade-off. As was stated above, a discount factor discount has been used to control the quality of generated data. Given a discount factor $\texttt{discount} = 1$, the algorithm would accept or reject a generated sample with a standard bar (as other MH correction methods). When $\texttt{discount} \in (0,1]$, RMLD would lower the bar to accept a generated sample, then the algorithm enjoys a higher rate of acceptance. On the other hand, given a discount factor $\texttt{discount} > 1$, the algorithm makes it "more difficult" for acceptance (with higher transition probability required).

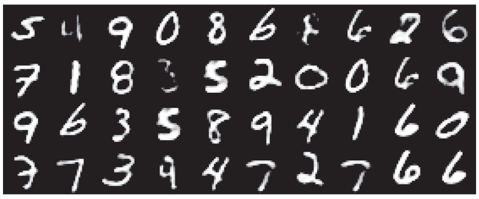
To lower the complexity of estimation, the transition probability estimation is linearized, e.g., $\Delta\cdot\nabla f_x$ or $\Delta\cdot\nabla f_z$, using the aggregated step size $\Delta = \eta T$ without considering the second-order dynamics. In this way, we don't need to calculate the aggregation of the T steps. On the other hand, When $T = 1$, the proposed algorithm would perform exactly the same as [18] with a strong theoretical guarantee to eliminate bias caused by the finite step-size η.

(a) Original


(b) Compression


(c) RMLD


(d) RMLD w/o MH


(e) Kernel HMC

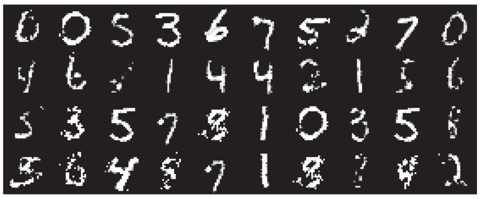

(f) Vanilla GAN

Fig. 1. Examples of data synthesis and generation based on MNIST.

5 EXPERIMENTS AND EMPIRICAL VALIDATION

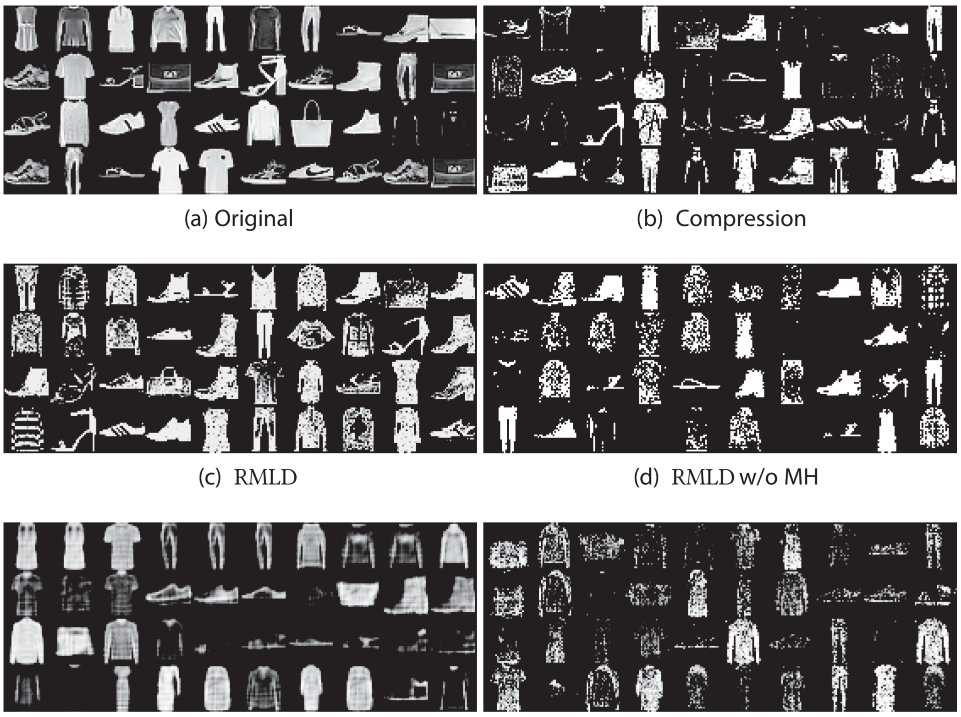
We propose to evaluate `RMLD` using four real-world benchmark datasets including MNIST, Fashion MNIST, Quick Draw, and EMNIST for two applications such as data synthesis and regeneration, spectral data augmentation for supervised machine learning and image classification.

5.1 Image Dataset Synthesis and Regeneration

To better understand the performance of `RMLD` for image data generation, we evaluate the proposed algorithms with baselines using the visionary datasets. All these datasets consist of images with 28×28 pixels at gray-scale. In our research, we consider each image as a vector of 784 dimensions, where each dimension is scaled from 0 to 255.

5.1.1 Baselines and Setup. We include the following methods as the baseline algorithms for comparison. The work that is most relevant to our work is Kernel HMC [44, 45], which enables Bayesian sampling without the derivative evaluator of the (log) posterior density. In addition to Kernel HMC, we also want to measure the effect the MH correction to the performance of sampling. Thus, we also include the option to disable/enable the MH correction during the sampling procedure. With MH correction disabled, all samples that are traversed by the dynamics will be accepted as the data generation. In this way, we provide three key baseline algorithms:

(1) `RMLD` *without MH correction:* An algorithm derived from `RMLD` without incorporating the discounted MH rejection;



(a) Original (b) Compression

(c) RMLD (d) RMLD w/o MH

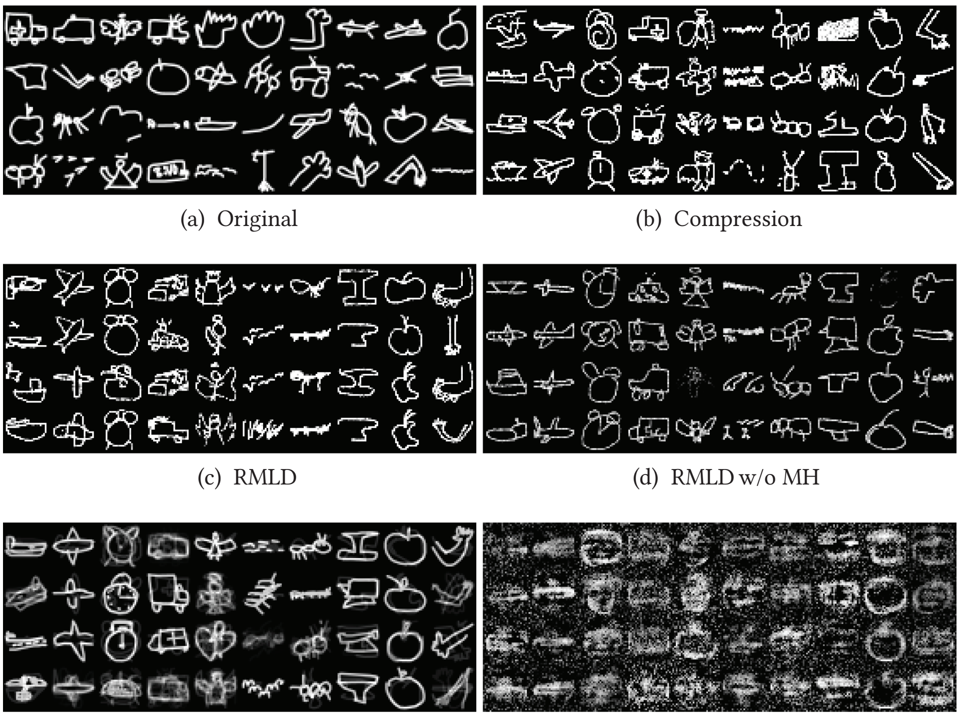
(e) Kernel HMC (f) Vanilla GAN

Fig. 2. Examples of data synthesis and generation based on Fashion MNIST.

(2) *Kernel HMC:* The algorithm [44, 45] based on common SGLD, where Gaussian kernels are used to model the posterior probability distribution of the given dataset for data generation;
(3) *Vanilla GAN:* the most principled Generative Adversarial Network (GAN) [20] that generates images from random noise. All Vanilla GANs are trained using the training datasets with 20,000 iterations. Note that the training process of vanilla GAN are usually collapsed, due to the complexity of datasets, while some tricks are available to fix these issues [12]. The motivation to use Vanilla GAN is to demonstrate the capacity to learn distribution from datasets through adversarial training over generator-discriminator structure [38].

The comparison between RMLD to its variant without MH correction demonstrates the improvement made through rejecting low probability samples for data generation, while the comparison to *Kernel HMC* illustrates the effectiveness of our fantastic intuition that leverages sparse coding/reconstruction in Monte Carlo settings (rather than compressed sensing).

Further, we setup the proposed algorithm RMLD (and RMLD without MH correction) with a $784 \times 3{,}136$ Gaussian random noise matrix as the measurement $\mathbf{A}$. With such measurement matrix, RMLD can reconstruct a higher dimensional (i.e., four times of original dimensions) probability distribution that characterizes the given datasets as its low dimensional observations. Note that the performance of RMLD can be further improved with a fine-tuned measurement matrix $\mathbf{A}$ given through dictionary learning, though a Gaussian matrix is capable of providing a certain theoretical consequence. All other parameters such as regularization term λ, step size η and batch size m for



(a) Original (b) Compression

(c) RMLD (d) RMLD w/o MH

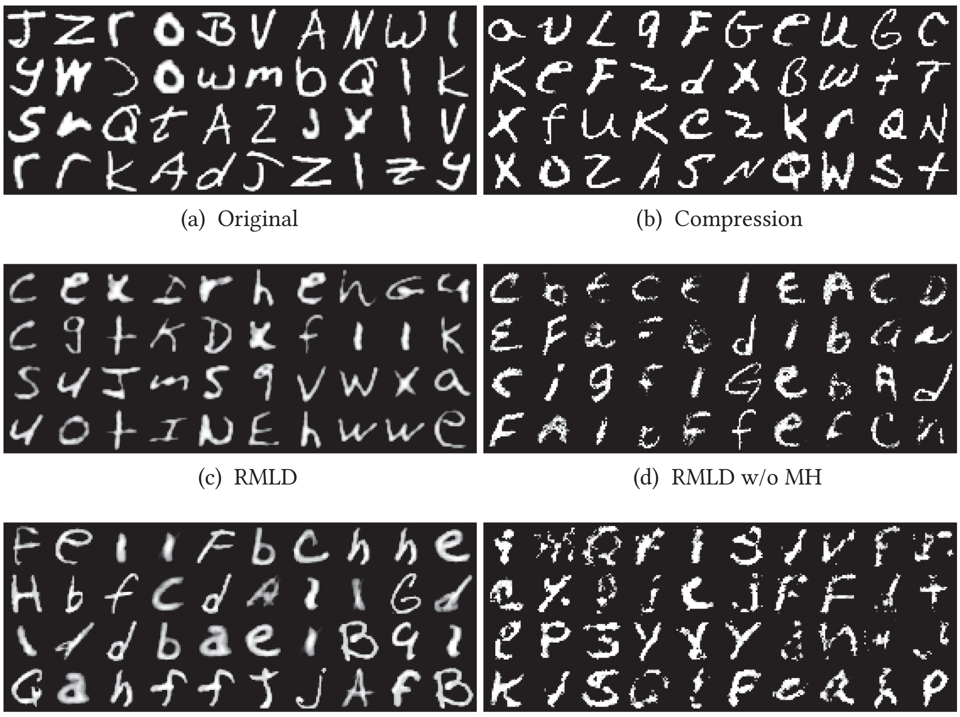
(e) Kernel HMC (f) Vanilla GAN (collapsed)

Fig. 3. Examples of data synthesis and generation based on QuickDraw.

both RMLD and baselines are all tuned best with repeated trials. All generated images (except GAN) are filtered by the same Gaussian smoother to rescale each pixel at 0–255 range.

5.1.2 Results. Figures 1–4 demonstrate the original images and generated images by various algorithms. Specifically, Figures 1 refers to the images randomly drawn from the MNIST dataset, while Figure 1(b) demonstrates the results based on *Compression*, where for each image, we first draw a random image from the original dataset, then we use LASSO and measurement matrix $\mathbf{A}$ to reconstruct its high-dimensional representation, further we use $\mathbf{A}$ to obtain its low-dimensional observation. We expect to observe the effect on the reconstruction and compression of the original image. The comparison between compression to the original shows that the shape of letters and digits for EMNIST and MNIST datasets can be well preserved through the reconstruction-and-compression procedure, while the image quality for Fashion MNIST image reconstruction is poor.

Figures 1 (c)–(h) present the generated images of the MNIST dataset using RMLD, RMLD without MH correction, Kernel HMC, and GAN, respectively. All these models have successfully generated images that are visible and human understandable. It is quite subjective to judge or compare the quality of these images. We observe that the original images indeed incorporate some local patterns, such as shadows and textures of materials. Most of the sampling methods can well reconstruct the shapes while missing the supports to the local patterns, RMLD seems working well for both shape and pattern recovery. Similar observations can be made from the images generated for all other datasets including Fashion MNIST, EMNIST and QuickDraw demonstrated in Figures 2–4. We also find it is quite difficult for convolutional GAN to learn Quick Draw dataset [22]. We believe it is due to that the rest image datasets were all gray-scaled from photographs of real-world objects, while Quick Draw consisted of images plotted or created by a computer program (RNN).



(a) Original (b) Compression

(c) RMLD (d) RMLD w/o MH

(e) Kernel HMC (f) Vanilla GAN

Fig. 4. Examples of data synthesis and generation based on EMNIST.

5.1.3 Discussion on Image Generation. Comparing RMLD to RMLD w/o MH (the RMLD wit MH Correction disabled), the images procedure by RMLD can be reviewed as a subset of images by RMLD w/o MH, which has been carefully selected by the transition probabilities. The images sampled by RMLD are rendered with higher sharpness, while the edges of RMLD w/o MH are usually blurred. Kernel HMC is with the similar drawback, where the produced images are lack of detailed local patterns and with blurred edges. The images produced by Vanilla GAN are generally with good quality and local patterns reconstructed. From a Bayesian sampling perspective, we doubt whether GAN or the general Generative Adversarial Net can sample from the original distribution of images consistently. Recent study [38] shows that Vanilla GAN failed to capture/cover the whole distribution from sample generation with significant biases. Note that using "compression," one can also obtain good plots of images through a procedure based on compressing sensing and the measurement matrix based compression. However, compression generates image one-by-one based on original images and it can not generate any new images in these settings.

In summary, we conclude that RMLD is capable of generating images. Actually. it samples from the distribution of sparse representation from the datasets, while one can easily convert these sparse representations back to the images using the dictionary (i.e., measurement matrix). Compared to the images produced Kernel HMC and GAN, we cannot observe the significant drawback of RMLD. We subjectively the images generated by RMLD look better, as they preserve more local patterns and textures.

5.1.4 Discussion on the Comparison with the Vanilla GAN. In our work, we compare RMLD with the data generated by Vanilla GAN [20], where tricks to improve the image quality of GAN are

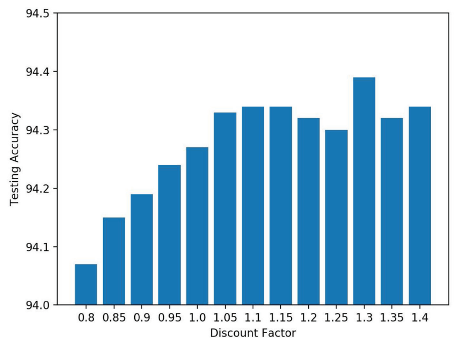
Table 2. Error Comparison Using RMLD on MNIST, Fashion MNIST, Quick Draw, and EMNIST Datasets

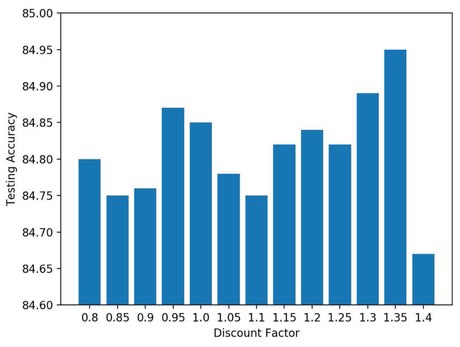
MNIST [26] (n = 60,000, d = 784)						
	SVM	ℓ_2-SVM	ElasticNet	ℓ_1-Log. Reg.	ℓ_2-Log. Reg.	ℓ_2-Percep.
RMLD	**3.59 ± 0.00**	**5.74 ± 0.06**	**8.03 ± 0.08**	**5.49 ± 0.01**	**5.23 ± 0.08**	**12.54 ± 0.49**
$\text{RMLD}_{\text{w/o MH}}$	3.61 ± 0.03	5.87 ± 0.19	8.02 ± 0.08	5.50 ± 0.09	5.38 ± 0.06	18.84 ± 2.60
Original	5.96 ± 0.00	11.11 ± 1.29	9.55 ± 0.04	8.06 ± 0.01	7.99 ± 0.00	16.62 ± 0.00
Compression	3.61 ± 0.00	5.84 ± 0.18	7.89 ± 0.12	5.50 ± 0.00	5.32 ± 0.00	13.22 ± 0.00
Kernel HMC	6.14 ± 0.11	15.44 ± 2.40	17.09 ± 2.50	34.14 ± 7.11	13.46 ± 0.79	29.42 ± 4.98
Vanilla GAN	5.99 ± 0.09	11.33 ± 0.45	10.60 ± 0.59	8.11 ± 0.14	8.15 ± 0.89	15.05 ± 1.27
Fashion MNIST [54] (n = 60,000, d = 784)						
	SVM	ℓ_2-SVM	ElasticNet	ℓ_1-Log. Reg.	ℓ_2-Log. Reg.	ℓ_2-Percep.
RMLD	**13.09 ± 0.08**	**15.19 ± 0.27**	**17.54 ± 0.27**	**14.33 ± 0.00**	**14.54 ± 0.08**	**19.82 ± 0.69**
$\text{RMLD}_{\text{w/o MH}}$	13.54 ± 0.08	15.42 ± 0.35	17.77 ± 0.15	14.46 ± 0.08	14.69 ± 0.09	22.18 ± 2.62
Original	15.36 ± 0.00	19.88 ± 2.91	20.35 ± 2.92	15.81 ± 0.03	15.89 ± 0.00	30.84 ± 0.00
Compression	13.20 ± 0.00	15.32 ± 0.26	17.61 ± 0.10	14.34 ± 0.01	14.55 ± 0.00	25.09 ± 0.00
Kernel HMC	15.73 ± 0.18	27.07 ± 2.36	25.64 ± 3.14	25.70 ± 0.78	24.99 ± 0.44	37.30 ± 7.73
Vanilla GAN	15.39 ± 0.10	19.29 ± 1.26	19.39 ± 1.78	15.99 ± 0.09	16.09 ± 0.18	34.17 ± 2.69
Quick Draw [22] (n = 60,000, d = 784)						
	SVM	ℓ_2-SVM	ElasticNet	ℓ_1-Log. Reg.	ℓ_2-Log. Reg.	ℓ_2-Percep.
RMLD	**29.39 ± 0.00**	**33.98 ± 0.59**	**34.78 ± 0.13**	**32.42 ± 0.01**	**31.33 ± 0.00**	**52.47 ± 0.07**
$\text{RMLD}_{\text{w/o MH}}$	29.72 ± 0.08	34.07 ± 0.18	35.18 ± 0.45	32.64 ± 0.00	31.62 ± 0.03	56.55 ± 0.08
Original	35.22 ± 0.00	47.39 ± 1.80	43.06 ± 1.60	37.47 ± 0.02	37.6 ± 0.00	58.49 ± 0.00
Compression	29.43 ± 0.00	33.99 ± 0.80	35.05 ± 0.90	32.62 ± 0.00	31.34 ± 0.00	64.35 ± 0.00
Kernel HMC	35.54 ± 0.18	46.32 ± 1.14	42.36 ± 0.72	37.80 ± 0.14	37.88 ± 0.19	53.43 ± 1.37
Vanilla GAN	35.27 ± 0.20	46.39 ± 1.81	43.94 ± 1.92	37.63 ± 0.14	37.76 ± 0.15	53.70 ± 2.27
EMNIST [13] (n = 124,800, d = 784)						
	SVM	ℓ_2-SVM	ElasticNet	ℓ_1-Log. Reg.	ℓ_2-Log. Reg.	ℓ_2-Percep.
RMLD	**13.14 ± 0.09**	**15.38 ± 0.11**	**28.80 ± 0.11**	**17.64 ± 0.11**	**14.41 ± 0.07**	**19.83 ± 0.69**
$\text{RMLD}_{\text{w/o MH}}$	13.32 ± 0.06	22.71 ± 0.10	29.32 ± 0.15	18.37 ± 0.13	18.43 ± 0.07	49.36 ± 2.48
Original	21.21 ± 0.00	37.33 ± 0.80	32.41 ± 0.50	28.82 ± 0.00	29.10 ± 0.00	50.62 ± 0.00
Compression	13.21 ± 0.00	22.60 ± 0.17	28.93 ± 0.05	18.33 ± 0.01	18.36 ± 0.00	49.43 ± 0.00
Kernel HMC	25.18 ± 0.19	42.54 ± 2.26	34.31 ± 0.68	31.59 ± 0.09	31.99 ± 0.07	56.55 ± 2.80
Vanilla GAN	22.17 ± 0.14	44.26 ± 1.70	38.98 ± 1.01	33.16 ± 0.29	33.21 ± 0.40	57.83 ± 2.35

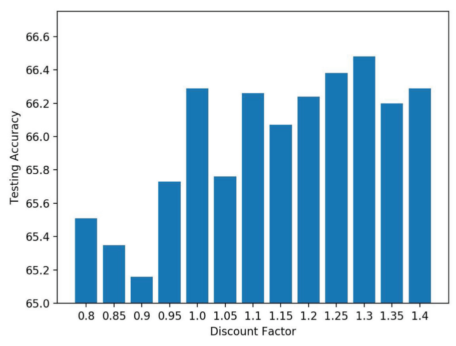
all disabled [12, 21, 39]. To ensure the fair comparison, we trained the Vanilla GAN using the comparable computational resources (CPU/GPU hours) consumed by other methods. Under above settings, it sometimes leads to model collapse and often generates low quality images, compared to RMLD and Monte Carlo-based solutions. Even though certain image quality improvement tricks were used with additional computational cost, we still doubt that GAN could perform well in terms of capturing the whole distribution of datasets. Some observation has been made from the comparison between probabilistic models and GAN [38]. The next experiments based on spectral data augmentation would further validate this observation, where the data generated by RMLD could help to improve the generalization performance of common statistical machines with higher testing accuracy, as RMLD demonstrates great potentials to capture the distribution.

5.2 Spectral Data Augmentation

To further evaluate the quality of samples drawn, we evaluate RMLD, as a tool for data augmentation, using a wide range of linear classifiers including SVM, ℓ_2-SVM, ℓ_1-regularized Logistic


(a) MNIST ℓ_2-SVM


(b) Fashion MNIST ℓ_2-SVM


(c) Quick Draw ℓ_2-SVM

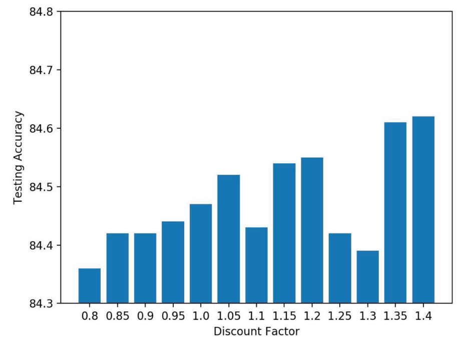
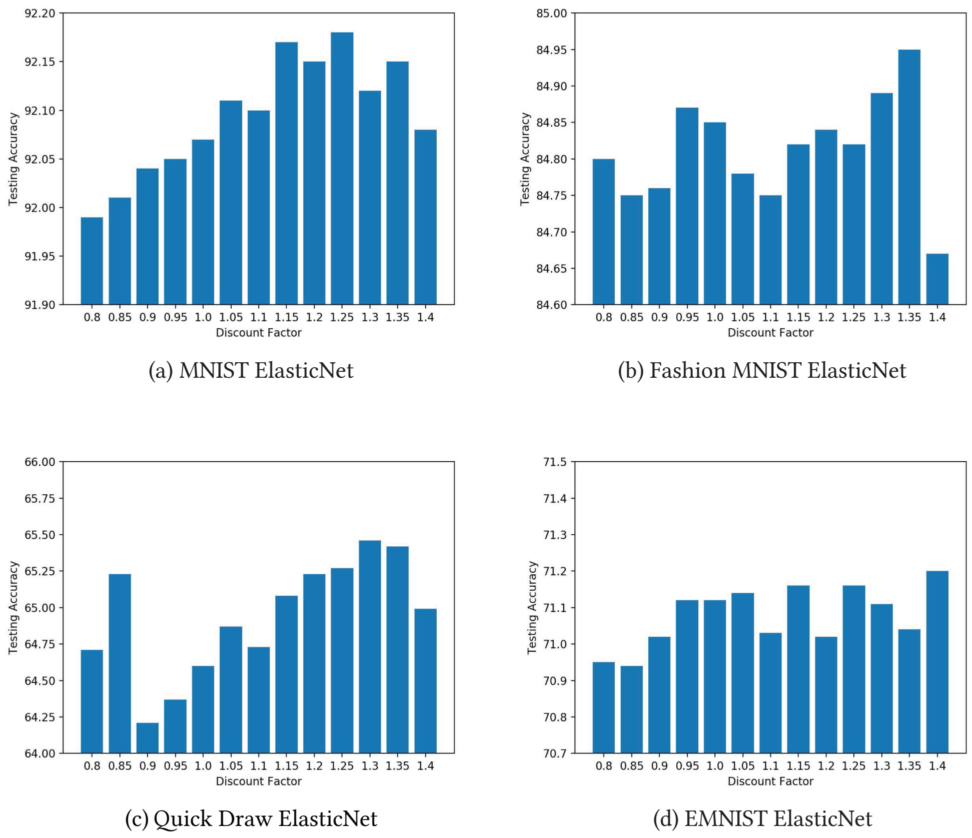

(d) EMNIST ℓ_2-SVM

Fig. 5. Testing accuracy vs. discount factor on SVM classifiers.

Regression (entitled ℓ_1-Log. Reg.), ℓ_2-regularized Logistic Regression (entitled ℓ_2-Log. Reg.), and ℓ_1-regularized Perceptron (entitled ℓ_1-Perceptron). We do not intend to perform such comparison on top of neural networks, as these methods already incorporate data augmentation in their deep architectures.

In this experiment, we shuffle the original datasets with the images generated in the ratio of 6:1. Note that for the experiment based on RMLD and compression, we use the spectral data (generated spectral samples by RMLD and/or the one obtained by LASSO using the same dictionary) for training and testing rather than the images. All algorithms are tuned with the best hyper-parameters through 10 folder cross-validation on the training set. We repeat the experiments five times to estimate the accuracy with intervals.

5.2.1 Results and Comparison. Table 2 presents the testing error comparison of the six classifiers on the three datasets with various augmentations. RMLD outperforms all baseline methods with higher accuracy, while it marginally improves the results of compression (which consists of the spectral representation of original data). It is obvious that, in the most cases, the upper interval of the error of RMLD (indicating the worst case accuracy) is still lower than the lower interval of baselines (indicating the best case accuracy). The comparison shows that RMLD significantly outperform the one based on the original dataset, Kernel HMC and GAN with clearly higher accuracy. The comparison between RMLD and Kernel HMC demonstrate the power of sampling sparse representation from space, rather than the spatial-temporal information of images. The comparison



(a) MNIST ElasticNet

(b) Fashion MNIST ElasticNet

(c) Quick Draw ElasticNet

(d) EMNIST ElasticNet

Fig. 6. Testing accuracy vs discount factor on ElasticNet.

between RMLD and RMLD without MH correction shows the power of rejecting low probability samples to well augment the training set. The compression between RMLD and Compression shows the effectiveness of Bayesian sampling for generating new spectral samples. We consider the advantage of RMLD is due to the effectiveness combining sparse coding and Bayesian sampling.

5.2.2 Case Studies. Figures 5 and 6 demonstrated the accuracy of ℓ_2-SVM and ElasticNet Logistic Regression classifiers on varying `discount` factor. In overall, RMLD works very well with excellent classification accuracy on top of a wide range of `discount` selection. Further, we observed that both models would deliver a lower accuracy on both datasets when `discount` factor is relatively small. It would enjoy a higher accuracy when increasing the `discount` factor and achieve its best accuracy when the `discount` factor is set to around 1.15~1.35. As was discussed in Section 4.3, RMLD would enjoy its best performance for data augmentation when we slightly increase the bar for generated image acceptance. The accuracy will fall down when `discount` further increases, as the randomness given to the generated images would be relatively smaller in such case.

6 DISCUSSION AND CONCLUSION

In this article, we formulate and study the problem of sparse representations sampling under several key assumptions. The problem intends to draw new samples from an unknown sparse representation domain using a given dataset, so as to generate a sequence of random, identical and independent samples (sparse representations). We then propose RMLD, which aims at sampling the sparse representations from the given dataset. Incorporating a random measurement matrix (or namely random dictionary) for sparse coding, RMLD leverages a SLGD to traverse on a log-posterior density model derived from compressed sensing, where a sequence of samples can be generated via the trajectory sampled by the dynamics. Due to the noise incorporated by mini-batch re-sampling, this method can provide controllable randomness to the generated samples. On the other hand, the perturbed gradient flow used in Langevin dynamics cannot ensure independence between generated samples in a non-asymptotic setting [30].

The empirical validation, based on four benchmark image datasets, shows RMLD can draw the possible sparse representation from the spectral space of the images, while the sparse representations drawn can be used to generate new images. We compare the images generated by RMLD with those based on Kernel HMC and GAN. While all generated images are visible and human understandable, the images produced by RMLD seem to preserve more local patterns and textures. Moreover, our experiments indicate that the generated data could help to augment the original datasets for supervised learning tasks, i.e., image classification. The comparison shows RMLD significantly outperforms the one augmented by Kernel HMC (SGLD based on Gaussian Kernel) and GAN with higher classification accuracy.

REFERENCES

[1] Sungjin Ahn, Anoop Korattikara, Nathan Liu, Suju Rajan, and Max Welling. 2015. Large-scale distributed Bayesian matrix factorization using stochastic gradient MCMC. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 9–18.

[2] Sungjin Ahn, Anoop Korattikara, and Max Welling. 2012. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *Proceedings of the International Conference on Machine Learning (ICML'12)*. 1591–1598.

[3] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I. Jordan. 2003. An introduction to MCMC for machine learning. *Machine Learning* 50, 1–2 (2003), 5–43.

[4] Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340* (2017).

[5] Mark Bagnoli and Ted Bergstrom. 2005. Log-concave probability and its applications. *Economic Theory* 26, 2 (2005), 445–469.

[6] Yoshua Bengio, Olivier Delalleau, and Nicolas L. Roux. 2006. The curse of highly variable functions for local kernel machines. In *Proceedings of the Advances in Neural Information Processing Systems*. 107–114.

[7] T. Tony Cai and Lie Wang. 2011. Orthogonal matching pursuit for sparse signal recovery with noise. *IEEE Transactions on Information theory* 57, 7 (2011), 4680–4688.

[8] Emmanuel J. Candès, Justin Romberg, and Terence Tao. 2006. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information theory* 52, 2 (2006), 489–509.

[9] Bing Cao, Nannan Wang, Jie Li, and Xinbo Gao. 2018. Data augmentation-based joint learning for heterogeneous face recognition. *IEEE Transactions on Neural Networks and Learning Systems* 30, 6 (2018), 1731–1743.

[10] Yuan Cao, Haibo He, and Hong Man. 2012. SOMKE: Kernel density estimation over data streams by sequences of self-organizing maps. *IEEE Transactions on Neural Networks and Learning Systems* 23, 8 (2012), 1254–1268.

[11] Tianqi Chen, Emily Fox, and Carlos Guestrin. 2014. Stochastic gradient hamiltonian monte carlo. In *Proceedings of the International Conference on Machine Learning*. 1683–1691.

[12] Soumith Chintala, Emily Denton, Martin Arjovsky, and Michael Mathieu. [n.d.]. How to train a GAN? Tips and tricks to make GANs work (2017). Retrieved from https://github.com/soumith/ganhacks.

[13] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. 2017. EMNIST: An extension of MNIST to handwritten letters. In *Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN'17)*. IEEE, 2921–2926.

[14] David L. Donoho. 2006. Compressed sensing. *IEEE Transactions on Information Theory* 52, 4 (2006), 1289–1306.

[15] David L. Donoho. 2006. For most large underdetermined systems of linear equations the minimal ℓ_1−norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 59, 6 (2006), 797–829.

[16] David L. Donoho, Michael Elad, and Vladimir N. Temlyakov. 2006. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on Information Theory* 52, 1 (2006), 6–18.

[17] Chao Du, Jun Zhu, and Bo Zhang. 2018. Learning deep generative models with doubly stochastic gradient MCMC. *IEEE Transactions on Neural Networks and Learning Systems* 29, 7 (2018), 3084–3096.

[18] Raaz Dwivedi, Yuansi Chen, Martin J. Wainwright, and Bin Yu. 2018. Log-concave sampling: Metropolis-Hastings algorithms are fast! In *Proceedings of the Conference on Learning Theory*. PMLR, 793–797.

[19] Maurizio Filippone and Raphael Engler. 2015. Enabling scalable stochastic gradient-based inference for Gaussian processes by employing the Unbiased LInear System SolvEr (ULISSE). In *International Conference on Machine Learning*. 1015–1024.

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of the Advances in Neural Information Processing Systems*. 2672–2680.

[21] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C. Courville. 2017. Improved training of wasserstein gans. In *Proceedings of the Advances in Neural Information Processing Systems*. 5767–5777.

[22] David Ha and Douglas Eck. 2018. A neural representation of sketch drawings. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Hy6GHpkCW.

[23] W. Keith Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57, 1 (1970), 97–109.

[24] Shihao Ji, Ya Xue, and L. Carin. 2008. Bayesian compressive sensing. *IEEE Transactions on Signal Processing* 56, 6 (2008), 2346–2356.

[25] Xiao-Bo Jin, Xu-Yao Zhang, Kaizhu Huang, and Guang-Gang Geng. 2018. Stochastic conjugate gradient algorithm with variance reduction. *IEEE Transactions on Neural Networks and Learning Systems* 30, 5 (2018), 1360–1369.

[26] Yann LeCun, Corinna Cortes, and C. J. Burges. 2010. MNIST handwritten digit database. *AT&T Labs*. Retrieved from http://yann.lecun.com/exdb/mnist. 2 (2010).

[27] Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. 2016. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *Proceedings of the 13th AAAI Conference on Artificial Intelligence*. AAAI Press, 1788–1794.

[28] Luoqing Li, Weifu Li, Bin Zou, Yulong Wang, Yuan Yan Tang, and Hua Han. 2018. Learning with coefficient-based regularized regression on Markov resampling. *IEEE Transactions on Neural Networks and Learning Systems* 29, 9 (2018), 4166–4176.

[29] Xiaoqiang Lu, Yulong Wang, and Yuan Yuan. 2013. Sparse coding from a Bayesian perspective. *IEEE Transactions on Neural Networks and Learning Systems* 24, 6 (2013), 929–939.

[30] Yi-An Ma, Tianqi Chen, and Emily Fox. 2015. A complete recipe for stochastic gradient MCMC. In *Proceedings of the Advances in Neural Information Processing Systems*. 2917–2925.

[31] Dougal Maclaurin and Ryan P. Adams. 2014. Firefly Monte Carlo: Exact MCMC with subsets of data. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*. 543–552.

[32] Stephan Mandt, Matthew Hoffman, and David Blei. 2016. A variational analysis of stochastic gradient algorithms. In *Proceedings of the International Conference on Machine Learning*. 354–363.

[33] Stephan Mandt, Matthew D. Hoffman, and David M. Blei. 2017. Stochastic gradient descent as approximate Bayesian inference. *The Journal of Machine Learning Research* 18, 1 (2017), 4873–4907.

[34] Gaétan Marceau-Caron and Yann Ollivier. 2017. Natural Langevin dynamics for neural networks. In *Proceedings of the International Conference on Geometric Science of Information*. Springer, 451–459.

[35] Boris T. Polyak and Anatoli B. Juditsky. 1992. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization* 30, 4 (1992), 838–855.

[36] Carl Edward Rasmussen. 2004. Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning*. Springer, 63–71.

[37] Vikas Chandrakant Raykar and Ramani Duraiswami. 2006. Fast optimal bandwidth selection for kernel density estimation. In *Proceedings of the 2006 SIAM International Conference on Data Mining*. SIAM, 524–528.

[38] Eitan Richardson and Yair Weiss. 2018. On GANs and GMMs. *Advances in Neural Information Processing Systems* 31 (2018), 5847–5858.

[39] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. 2016. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems 29*. D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Eds.). Curran Associates, Inc., 2234–2242. Retrieved from http://papers.nips.cc/paper/6125-improved-techniques-for-training-gans.pdf.

[40] Hiroaki Sasaki, Yung-Kyun Noh, Gang Niu, and Masashi Sugiyama. 2016. Direct density derivative estimation. *Neural Computation* 28, 6 (2016), 1101–1140.

[41] Hiroaki Sasaki, Yung-Kyun Noh, and Masashi Sugiyama. 2015. Direct density-derivative estimation and its application in KL-divergence approximation. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*. 809–818.

[42] Ingmar Schuster, Heiko Strathmann, Brooks Paige, and Dino Sejdinovic. 2017. Kernel sequential Monte Carlo. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases.* Springer, 390–409.

[43] David W. Scott. 2015. *Multivariate Density Estimation: Theory, Practice, and Visualization.* John Wiley & Sons.

[44] Heiko Strathmann. 2018. *Kernel Methods for Monte Carlo.* Ph.D. Dissertation. UCL (University College London).

[45] Heiko Strathmann, Dino Sejdinovic, Samuel Livingstone, Zoltan Szabo, and Arthur Gretton. 2015. Gradient-free Hamiltonian Monte Carlo with efficient kernel exponential families. In *Proceedings of the Advances in Neural Information Processing Systems.* 955–963.

[46] Jayaraman J. Thiagarajan, Karthikeyan Natesan Ramamurthy, and Andreas Spanias. 2015. Learning stable multilevel dictionaries for sparse representations. *IEEE Transactions on Neural Networks and Learning Systems* 26, 9 (2015), 1913–1926.

[47] Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 58, 1 (1996), 267–288.

[48] Joel A. Tropp and Anna C. Gilbert. 2007. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory* 53, 12 (2007), 4655–4666.

[49] Berwin A. Turlach. 1993. Bandwidth selection in kernel density estimation: A review. In *CORE and Institut de Statistique.* Citeseer.

[50] Vladimir Naumovich Vapnik. 1999. An overview of statistical learning theory. *IEEE Transactions on Neural Networks* 10, 5 (1999), 988–999.

[51] Ruosi Wan, Mingjun Zhong, Haoyi Xiong, and Zhanxing Zhu. 2020. Neural control variates for Monte Carlo variance reduction. In *Machine Learning and Knowledge Discovery in Databases*, Ulf Brefeld, Elisa Fromont, Andreas Hotho, Arno Knobbe, Marloes Maathuis, and Céline Robardet (Eds.). Springer International Publishing, Cham, 533–547.

[52] Max Welling and Yee W. Teh. 2011. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11).* 681–688.

[53] Di Wu and Jinwen Ma. 2018. A two-layer mixture model of Gaussian process functional regressions and its MCMC EM algorithm. *IEEE Transactions on Neural Networks and Learning Systems* 29, 10 (2018), 4894–4904.

[54] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).

[55] Haoyi Xiong, Wei Cheng, Jiang Bian, Wenqing Hu, and Zhishan Guo. 2017. AWDA: Adaptive wishart discriminant analysis. In *Proceedings of the 17th IEEE International Conference on Data Mining (ICDM'17).* IEEE.

[56] Haoyi Xiong, Wei Cheng, Yanjie Fu, Wenqing Hu, Jiang Bian, and Zhishan Guo. 2018. De-biasing covariance-regularized discriminant analysis. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence.* 2889–2897.

[57] Haoyi Xiong, Kafeng Wang, Jiang Bian, Zhanxing Zhu, Cheng-zhong Xu, Zhishan Guo, and Jun Huan. [n.d.]. SpHMC: Spectral Hamiltonian Monte Carlo. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence.*

[58] Haoyi Xiong, Jinghe Zhang, Yu Huang, Kevin Leach, and Laura E. Barnes. 2017. Daehr: A discriminant analysis framework for electronic health record data and an application to early detection of mental health disorders. *ACM Transactions on Intelligent Systems and Technology* 8, 3 (2017), 47.

[59] Jie Xu, Yuan Yan Tang, Bin Zou, Zongben Xu, Luoqing Li, and Yang Lu. 2015. The generalization ability of online SVM classification based on Markov sampling. *IEEE Transactions on Neural Networks and Learning Systems* 26, 3 (2015), 628–639.

[60] Nanyang Ye and Zhanxing Zhu. 2018. Stochastic fractional Hamiltonian Monte Carlo. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18).*

[61] Bin Zou, Luoqing Li, Zongben Xu, Tao Luo, and Yuan Yan Tang. 2013. Generalization performance of Fisher linear discriminant based on Markov sampling. *IEEE Transactions on Neural Networks and Learning Systems* 24, 2 (2013), 288–300.

Received October 2019; accepted September 2020