Batch-less stochastic gradient descent for compressive learning of deep regularization for image denoising

Hui Shi, Yann Traonmilin, J-F Aujol

Batch-less stochastic gradient descent for compressive learning of deep regularization for image denoising

Hui Shi[1], Yann Traonmilin[1*] and Jean-François Aujol[1]

[1]Univ. Bordeaux, Bordeaux INP, CNRS, IMB, UMR 5251, F-33400, Talence, France.

*Corresponding author(s). E-mail(s): yann.traonmilin@math.u-bordeaux.fr;
Contributing authors: hui.shi@u-bordeaux.fr; jean-francois.aujol@math.u-bordeaux.fr;

Abstract

We consider the problem of denoising with the help of prior information taken from a database of clean signals or images. Denoising with variational methods is very efficient if a regularizer well adapted to the nature of the data is available. Thanks to the maximum a posteriori Bayesian framework, such regularizer can be systematically linked with the distribution of the data. With deep neural networks (DNN), complex distributions can be recovered from a large training database. To reduce the computational burden of this task, we adapt the compressive learning framework to the learning of regularizers parametrized by DNN. We propose two variants of stochastic gradient descent (SGD) for the recovery of deep regularization parameters from a heavily compressed database. These algorithms outperform the initially proposed method that was limited to low-dimensional signals, each iteration using information from the *whole database*. They also benefit from classical SGD convergence guarantees. Thanks to these improvements we show that this method can be applied for patch based image denoising.

1 Introduction

At the heart of imaging inverse problems, having a precise prior information on the distribution of the unknown image is crucial for efficient recovery of said image. In particular, consider the denoising problem, i.e. finding an accurate estimate $u^\star$ of the original image $u_0 \in \mathbb{R}^d$ from the observed noisy image $v \in \mathbb{R}^d$:

$$v = u_0 + \epsilon, \tag{1}$$

where the noise ϵ (assumed to be additive white Gaussian noise of standard deviation σ) is independent of u_0. Recovering u_0 from its degraded version v is an ill-posed problem and we need to use additional (prior) information about the unknown image u_0 to obtain meaningful solutions. A common strategy [2] for solving inverse problems is to define an estimator $u^\star$ which is the minimizer of a functional:

$$u^\star \in \arg\min_u F(u) + \lambda R(u), \tag{2}$$

where F is the data fidelity term making the solution consistent with the observation y and R is the regularization term that incorporates the prior information, weighted by the regularization parameter $\lambda > 0$. The choice of R depends on the statistics of the signal of interest which is not always available in real-life applications.

The maximum a posteriori (MAP) Bayesian framework provides a useful tool to interpret such methods. The MAP estimator is given by:

$$u^\star_{\text{MAP}} \in \arg\min_u \|v - u\|_2^2 - \lambda \log(\mu(u)) \tag{3}$$

where μ denotes a prior probability law (of density $\mu(\cdot)$) of the unknown data u. In this context, the regularizer is related to the prior distribution of the data, i.e., $R(u) = -log(\mu(u))$.

Defining an accurate prior model in the form of a regularizer or distribution of the images of interest is one of the main difficulties for designing efficient estimation methods. Classical Bayesian approaches, e.g. in image processing, rely on explicit priors such as total variation or Gaussian mixture models (GMM) [21] trained on a database of image patches. Recently, researchers proposed to use DNN to design the regularizer. Methods such as the total deep variation [11], adversarial regularizers [12, 15], as well as the Plug & Play approach and its extensions [20, 9] deliver remarkably accurate results. However, such models are typically learned from large datasets. Estimating their parameters from such a large-scale dataset is a serious computational challenge.

Compressive learning

One possibility to reduce the computational resources of learning consists in using the compressive learning (CL) framework [4, 5, 7]. The main idea of CL, coined as sketching, is to compress the whole data collection into a fixed-size representation, a so-called *sketch* of data, such that enough information relevant to the considered learning task is captured. Then the learned parameters are estimated by minimizing a nonlinear least-square problem built with the sketch. The size of the sketch m is chosen proportional to the intrinsic complexity of the learning task. Consequently, the cost of inferring the parameters of interest from the sketch does not depend on the size of the training database but on the number of parameters we want to estimate. Hence, it is possible to exploit arbitrarily large datasets in the sketching framework without demanding more computational resources.

During the sketching phase, a huge collection of n d-dimensional data vectors $X = \{x_i\}_{i=1}^n$ is summarized into a single m-dimensional ($m \ll n$) vector $\hat{z}$ with:

$$\hat{z} = \frac{1}{n}\sum_{i=1}^{n} \Phi(x_i) = \mathcal{S}(\hat{\mu}_n), \tag{4}$$

where $\hat{\mu}_n := \frac{1}{n}\sum_{i=1}^n \delta_{x_i}$ is the empirical probability distribution of the data, δ_{x_i} is the Dirac measure at x_i and the function $\Phi : \mathbb{R}^d \to \mathbb{C}^m$ is called the feature map (typically random Fourier moments). The sketch $\hat{z}$ is the mean of the feature map over the whole database.

Sketching can be interpreted as a linear operation $\mathcal{S}$ on measures μ defined by $\mathcal{S}\mu := \mathbb{E}_{X\sim\mu}\Phi(X)$. An estimate of a distribution μ_θ (or of distributional parameters θ of interest) can be calculated from the sketch by solving:

$$\mu_\theta^\star \in \arg\min_{\mu_\theta} G(\theta) = \arg\min_{\mu_\theta} \|\hat{z} - \mathcal{S}\mu_\theta\|_2^2. \tag{5}$$

In practice, this "sketch matching" problem can be solved by greedy compressive learning Orthogonal Matching Pursuit (OMP) algorithm and its extension Compressive Learning-OMP with replacement [10] when μ_θ is a mixture of elementary distribution (i.e. a GMM). When the distribution μ is a GMM in high-dimension with flat tail covariances, the problem can also be solved by the Low-Rank OMP algorithm (LR-OMP). It was shown that the prior model learned with LR-OMP can be used to perform image denoising [18] with no loss of performance compared to the non compressive approach, and with faster training time.

These greedy algorithms are suitable for any sketching operator $\mathcal{S}$ and any distribution density μ, as long as the sketch $\mathcal{S}\mu$ and its gradient $\nabla_\theta\mathcal{S}\mu$ with respect to the distributional parameters θ of interest have a closed-form expression: the core of these OMP-based algorithms is computing the expression of $\mathcal{S}\mu$ and $\nabla_\theta\mathcal{S}\mu$. However, for some more general distributions, the sketching feature map may not have a closed form. This limits the use of the sketching framework.

In this paper, our goal is to recover a good approximation of the probability distribution of any unknown data from its sketch (i.e. beyond GMM). As neural networks (NN) have great expressive power [8, 14], we propose to tackle the problem by adapting the sketching to NN. More precisely, we propose to define the regularizer R_θ parametrized by a DNN f_θ (precisely a ReLU network) as

$$R_\theta(\cdot) = \|f_\theta(\cdot)\|_2^2. \tag{6}$$

Such a regularization corresponds to the parametric distribution density $\mu_\theta \propto e^{-\|f_\theta(\cdot)\|_2^2}$. Thus it

can be viewed as a generalized Gaussian distribution, where the bilinear form induced by the covariance matrix is replaced by a network. Due to the fact that NN have good generalization properties, the proposed regularization should be capable of encoding complex probability distributions. Unfortunately, a direct practical application of existing tools is not possible as closed-form expressions of $\mathcal{S}\mu$ are not available for sketching operator $\mathcal{S}$ based on random Fourier features. In [19], it was shown on low-dimensional (2D and 3D) data that it was possible to estimate a deep regularizer by approximating $\mathcal{S}$ by a sampled version on a regular grid $\mathcal{S}_d$. Unfortunately the use of a grid limits the extension of this method to data in higher dimension such as image patches.

Contributions and outline

In this work, we propose novel approaches to learn deep regularizers from sketches beyond our initial method from [19]. Instead of relying on a grid-based discretization of the sketching operator $\mathcal{S}$, we propose an adaptation of the stochastic gradient descent method (that we call compressive learning stochastic gradient descent, CL-SGD), dynamically generating descent directions from the *whole training dataset* with the help of a random discretization of the sketching operator performed at each iteration. This strategy not only makes the approach suitable for higher-dimensional problems but also substantially enhances the efficiency and flexibility of the sketching process, requiring far fewer grid points and delivering considerably faster results compared to its predecessor. Once the neural network is trained denoising can be performed using classical variational methods. The method is tested using both synthetic and real audio and image data, demonstrating that a deep prior can be learned to perform patch based denoising. Moreover we provide a theoretical analysis ensuring the convergence of our compressive learning stochastic gradient method.

The rest of this article is organized as follows. We start by introducing the sketching framework, ReLU networks and some related works in section 2. In section 3, we describe the proposed framework: the adaptation of the compressive learning framework to the learning of regularizers parametrized by ReLU networks. Section 4 illustrates the performance of the proposed methods on both synthetic data and real-life data. Finally, conclusions are drawn in section 5.

2 Background, related works

We suppose that data samples x_i are modeled as independent and identically distributed random vectors having an unknown probability distribution with density $\mu \in \mathcal{M}(\mathcal{D})$ (where $\mathcal{M}(\mathcal{D})$ is the set of measures having a (Gateaux)-differentiable density supported on a domain $\mathcal{D} \subset \mathbb{R}^d$). For simplicity we identify μ with its density $\mu : \mathbb{R}^d \to \mathbb{R}^+$. Hence we can evaluate μ on any point $p_i \in \mathbb{R}^d$ with $\mu(p_i)$. For a collection of points $\mathbf{p} = (p_i)_{i=1}^P$, we write $\mu(\mathbf{p}) = (\mu(p_i))_{i=1}^P \in \mathbb{R}^P$. We define the linear sketching operator $\mathcal{S}$ that maps μ to the m-dimensional sketch vector z:

$$\begin{aligned} \mathcal{S} : \mathcal{M}(\mathcal{D}) &\to \mathbb{C}^m \\ z = \mathcal{S}\mu &:= \int_{\mathbb{R}^d} \mu(x)\Phi(x)dx. \end{aligned} \tag{7}$$

When the transformation (sketching feature map) $\Phi(\cdot)$ is built with random frequencies of the Fourier transform, the l-th component of the sketch is

$$z_l = \int_{\mathbb{R}^d} e^{-j\langle \omega_l, x\rangle}\mu(x)dx, \quad \text{for} \quad l = 1, \dots, m, \tag{8}$$

where $\{\omega_l\}_{l=1}^m \in \mathbb{R}^d$ are frequencies drawn at random. Taking a statistical perspective, the components z_l can be seen as samples of the characteristic function of $\mu_{\mathcal{X}}$. Accordingly, given a dataset $X = \{x_i\}_{i=1}^n$, the empirical sketch $\hat{z}$ can be computed from the samples of the database as

$$\hat{z}_l = \frac{1}{n}\sum_{i=1}^{n} e^{-j\langle \omega_l, x_i\rangle}, \quad \text{for} \quad l = 1, \dots, m. \tag{9}$$

The compression ratio r is m/nd. It was shown [10, 6, 5] that when the probability distribution μ has a low dimension structure, e.g. a GMM, one can recover it (with high probability) from enough randomly chosen samples of its Fourier transform. The required size of the sketch is typically of the order of the number of parameters we need to estimate.

ReLU network

A ReLU network, denoted by f_θ, is defined as a fully connected, feed-forward network (multi-layer perceptrons) with rectified linear unit (ReLU) activations. This activation has grown in popularity in feed-forward networks due to the success of first-order gradient based heuristic algorithms and the improvement in convergence to the approximated function for training [13].

Related works

The sketching framework has been successfully applied to parametric models including GMMs [10, 5, 18], K-means clustering [10, 5] and classification [16]. These methods are limited to the models for which the sketch function has a closed form. In our work, we apply the sketching to neural networks to encode more complex and high-dimensional probability distributions. Sketching techniques have found applications in neural networks, as evidenced by their use in [17] and [19]. In [17], the integration of sketching with generative networks enables the generation of data samples, while [19] pursues the goal of developing a deep regularizer for solving inverse problems. Notably, in [17], the authors suggest an approximation of the sketching map using Monte-Carlo sampling, whereas in [19], we chose for a discrete sketching operator for the approximation. It is worth highlighting that the sketching framework outlined earlier emphasizes data-independent approximation, specifically obtaining sketches through the averaging of random features.

3 Proposed method

In this section, we explain how we adapt the sketching framework to estimate regularizations by DNN. We then provide a theoretical analysis of our method and we detail the implementation of our algorithms.

3.1 Previous work

We start by explaining why there are no explicit closed-form expressions of the sketching function available in the context of prior parametrized by DNN. Intuitively, since ReLU networks define piecewise affine functions, we can indeed express a ReLU network f_θ as:

$$f_\theta(x) = \sum_{\gamma=1}^{N_R} \mathbf{1}_{R_\gamma}(x)(W_\gamma x + b_\gamma), \tag{10}$$

where $\mathbf{1}_{R_\gamma}$ is the indicator function of each of the N_R affine regions R_γ, with parameters (W_γ, b_γ).

Given a dataset X, we aim at learning, from only the sketch $\hat{z}$, an approximation μ_{θ^*} of the probability distribution μ generating X. We consider a regularizer of the form $R_\theta(\cdot) = \|f_\theta(\cdot)\|_2^2$ which corresponds to parametric densities of the form $\mu_\theta(\cdot) \propto e^{-R_\theta(\cdot)}$. Ideally, with the definition (8), the sketch would have to be calculated as

$$\begin{aligned} z_l &= \int_{\mathbb{R}^d} e^{-j\langle\omega,x\rangle} e^{-\|f_\theta(x)\|_2^2} dx \\ &= \int_{\mathbb{R}^d} e^{-j\langle\omega,x\rangle} e^{-\sum_{p=1}^{d}(\sum_\gamma^{N_R} \mathbf{1}_{R_\gamma}(x)((W_\gamma x)+b_\gamma))^2} dx. \end{aligned} \tag{11}$$

This would require Fourier Transforms on the individual regions R_γ. However, to the best of our knowledge, there is no analytic expression of such Fourier transform (Fourier transform on polygons).

In our previous work [19], it was proposed to perform the following minimization

$$\theta^* \in \arg\min_{\theta\in\Theta} \|\mathcal{S}_\mathbf{p}\mu_\theta - \hat{z}\|_2^2, \tag{12}$$

where $\mathcal{S}_\mathbf{p}$ is a discretization on a regular grid $\mathbf{p} = (p_i)_{i=1}^P \subset \mathbb{R}^d$ over the domain of the data, i.e.

$$(\mathcal{S}_\mathbf{p}\mu_\theta)_l \propto \sum_{p_i=1}^{P} e^{-j\langle\omega_l,p_i\rangle}\mu_\theta(p_i) \tag{13}$$

This optimization problem is then solved through gradient descent based methods. Considering the data at iteration t as θ^t, the update step can be expressed as follows:

$$\theta^t = \theta^{t-1} - \eta 2\mathcal{R}e\left((\nabla\mathcal{S}_\mathbf{p}\mu_\theta)^*(\mathcal{S}_\mathbf{p}\mu_\theta - \hat{z})\right) \tag{14}$$

where $\eta > 0$ represents the learning rate.

With the discretization, if μ_θ is differentiable at point p_i, the gradient of $\mathcal{S}_p\mu_\theta$ with respect to the parameters θ can be computed easily by using

the automatic differentiation. Note that the discretization is used only in the estimation of the regularizer from the sketch. It thus only impacts the calculation time and memory requirement of the estimation of the regularizer and not the size of the compressed dataset itself. Of course, the major pitfall of this approximation is the limitation for applications in high dimension as the number of points is exponential with respect to the dimension d. The required boundedness (or approximate boundedness such as in the Gaussian case) of the data is a valid assumption in many practical applications in signal and image processing.

3.2 Compressive learning stochastic gradient descent (CL-SGD)

Instead of discretizing the forward operator $\mathcal{S}$ once on a grid, we propose to perform a stochastic gradient descent where descent directions are generated with the help of a different random uniform discretization of $\mathcal{S}$ at each step. We first introduce a naïve discretization which we then adadapt to a CL-SGD method with theoretical convergence guarantees.

As an approximation of the minimization of G, first consider the discretized operator on a random grid $\mathbf{p}$ where $p_i \sim \mathcal{U}(\mathcal{D})$ ($\mathcal{U}(\mathcal{D})$ is the uniform distribution on $\mathcal{D}$) and the corresponding function

$$G_{\mathbf{p}}(\theta) = \|\mathcal{S}_{\mathbf{p}}\mu_\theta - \hat{z}\|_2^2 . \tag{15}$$

We remark that $\mathcal{S}_{\mathbf{p}}\mu_\theta$ can be written as

$$\mathcal{S}_{\mathbf{p}}\mu_\theta = B_{\mathbf{p}}\mu_\theta(\mathbf{p}) \tag{16}$$

where $\mu_\theta(\mathbf{p})$ is the density of μ_θ evaluated on the grid $\mathbf{p}$ and where $B_{\mathbf{p},l,i} = \frac{e^{-j\langle \omega_l, p_i\rangle}}{P}$. It is shown in Section 3.4 that $\mathbb{E}B_{\mathbf{p}}(\mu_\theta(\mathbf{p})) = \mathcal{S}\mu_\theta$, i.e. this random discretization is consistent with the sketch in expectation. Consider the minimization of $G_{\mathbf{p}}$

$$\min_{\theta\in\Theta} \|B_{\mathbf{p}}\mu_\theta(\mathbf{p}) - \hat{z}\|_2^2 . \tag{17}$$

where $\theta \subset \mathbb{R}^{d_0}$ is the set where the parameters (weights and bias) of the DNN live. This is a simple non-linear least square problem. We calculate the directional derivative of this functional with respect to θ in a direction h (the gradient is the evaluation of these derivatives in the directions formed by the canonical basis of $\mathbb{R}^{d_0}$):

$$\frac{1}{2}\partial_h G_{\mathbf{p}}(\theta) = \mathcal{R}e\langle B_{\mathbf{p}}\partial_h\mu_\theta(\mathbf{p}), B_{\mathbf{p}}\mu_\theta(\mathbf{p}) - \hat{z}\rangle \tag{18}$$

Recall that the directional derivatives of G are given by

$$\frac{1}{2}\partial_h G(\theta) = \mathcal{R}e\langle \mathcal{S}\partial_h\mu_\theta, \mathcal{S}\mu_\theta - \hat{z}\rangle. \tag{19}$$

Recall that we supposed that μ_θ is differentiable with respect to θ in any direction h (usually referred as Gateaux differentiability). We link the expectation of these derivatives with the directional derivatives of the original sketch matching functional G with the following Lemma.

Lemma 3.1. *Consider $\mathcal{S}$ constructed with frequencies $(\omega_l)_{l=1}^m$. Let $\mathbf{p} = (p_i)_{i=1}^P$ with $p_i \in \mathcal{U}(\mathcal{D})$, $\mu_\theta \in \mathcal{M}(\mathcal{D})$ and $h \in \mathbb{R}^{d_0}$ such that $\|h\|_2 = 1$. Then,*

$$\begin{aligned}\mathbb{E}_{\mathbf{p}}\partial_h G_{\mathbf{p}}(\theta) &= \partial_h G(\theta) \\ &\quad + \frac{2}{P}(m\langle \partial_h\mu_\theta, \mu_\theta\rangle_{L^2(\mathcal{D})} - \langle \mathcal{S}\partial_h\mu_\theta, \mathcal{S}\mu_\theta\rangle).\end{aligned} \tag{20}$$

Proof Thanks to Lemma 3.5, we calculate the expectation.

$$\begin{aligned}&\frac{1}{2}\mathbb{E}_{\mathbf{p}}\partial_h H(\theta) = \mathbb{E}_{\mathbf{p}}\langle B_{\mathbf{p}}\partial_h\mu_\theta, B_{\mathbf{p}}\mu_\theta - \hat{z}\rangle \\ &= \mathbb{E}\langle B_{\mathbf{p}}\partial_h\mu_\theta, B_{\mathbf{p}}\mu_\theta\rangle - \langle \mathcal{S}\partial_h\mu_\theta, \hat{z}\rangle \\ &= \frac{m}{P}\langle \mu_1, \mu_2\rangle_{L^2(\mathcal{D})} + \frac{P-1}{P}\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2\rangle \\ &\quad - \langle \mathcal{S}\partial_h\mu_\theta, \hat{z}\rangle \\ &= \langle \mathcal{S}\partial_h\mu_\theta, \mathcal{S}\mu_\theta - \hat{z}\rangle \\ &\quad + \frac{1}{P}(m\langle \partial_h\mu_{\theta_n}, \mu_\theta\rangle_{L^2(\mathcal{D})} - \langle \mathcal{S}\partial_h\mu_\theta, \mathcal{S}\mu_\theta\rangle).\end{aligned} \tag{21}$$

□

Remark that, unfortunately, there is a bias term that converges to 0 when $P \to \infty$. Hence this method will permit to obtain an approximation of the true gradients of G when P is large enough (note that the quality of the approximation does not depend on the size of the training database). The corresponding theoretical naïve SGD algorithm is given in Algorithm 1.

Algorithm 1: Naive CL-SGD algorithm

Data: Sketch $\hat{z}$, K iterations, γ step size

$z_0 \leftarrow \hat{z}$;
θ_0 randomly initialized;
for $k = 0, ..., K$ **do**
  Generate a grid $\mathbf{p}$ of P points $p_i \sim \mathcal{U}(\mathcal{D})$;
  $\theta_{k+1} = \theta_k - \gamma \nabla G_{\mathbf{p}}(\theta_k, z_k, p)$;
end

Looking at the origin of the bias term in the proof of Lemma 3.1, we are able to propose another approximation of the directional derivative, which is unbiased. Using two i.i.d random grids $\mathbf{p}$ and $\mathbf{q}$, we define the directions

$$d_{h,\mathbf{p},\mathbf{q}} = 2\langle B_{\mathbf{p}} \partial_h \mu_\theta(\mathbf{p}), B_{\mathbf{q}} \mu_\theta(\mathbf{q}) - \hat{z} \rangle \tag{22}$$

Its expectation is exactly the gradient of the sketch matching problem.

Lemma 3.2. *Consider $\mathcal{S}$ constructed with frequencies $(\omega_l)_{l=1}^m$. Let $\mathbf{p} = (p_i)_{i=1}^P, \mathbf{q} = (q_i)_{i=1}^P$ with $p_i, q_i \in \mathcal{U}(\mathcal{D})$ i.i.d., $\mu_\theta \in \mathcal{M}(\mathcal{D})$ and $h \in \mathbb{R}^{d_0}$ such that $\|h\|_2 = 1$, then*

$$\mathbb{E}_{\mathbf{p},\mathbf{q}} \left(d_{h,\mathbf{p},\mathbf{q}} \right) = \partial_h G(\theta). \tag{23}$$

Proof We calculate the expectation.

$$\mathbb{E}_{\mathbf{p},\mathbf{q}} (d_{h,\mathbf{p},\mathbf{q}}) = \mathbb{E}_{\mathbf{p},\mathbf{q}} \langle B_{\mathbf{p}} \partial_h \mu_\theta(\mathbf{p}), B_{\mathbf{q}} \mu_\theta(\mathbf{q}) - \hat{z} \rangle \tag{24}$$

As $\mathbf{p}, \mathbf{q}$ are i.i.d we have

$$\mathbb{E}_{\mathbf{p},\mathbf{q}} d_{h,p,\mathbf{q}} = \langle \mathbb{E}_{\mathbf{p}} B_{\mathbf{p}} \partial_h \mu_\theta(\mathbf{p}), \mathbb{E}_{\mathbf{q}} B_{\mathbf{q}} \mu_\theta(\mathbf{q}) - \hat{z} \rangle \tag{25}$$

Using Lemma 3.4 we have $\mathbb{E}_{\mathbf{p}} \left(B_{\mathbf{p}} \partial_h \mu_\theta(\mathbf{p}) \right) = \mathcal{S} \partial_h \mu_\theta$ and $\mathbb{E}_{\mathbf{q}} \left(B_{\mathbf{q}} \mu_\theta(\mathbf{q}) \right) = \mathcal{S} \mu_\theta$. This gives

$$\mathbb{E}_{\mathbf{p},\mathbf{q}} d_{h,\mathbf{p},\mathbf{q}} = \langle \mathcal{S} \partial_h \mu_\theta, \mathcal{S} \mu_\theta - \hat{z} \rangle. \tag{26}$$

□

With this expression we can build a stochastic descent direction with expectation being exactly the gradient of G:

$$D_{\mathbf{p},\mathbf{q}} = (d_{e_i,\mathbf{p},\mathbf{q}})_{i=1}^{d_0} \tag{27}$$

where e_i are the elements of the canonical basis of $\mathbb{R}^{d_0} \supset \Theta$. We have

Algorithm 2: Unbiased CL-SGD algorithm

Data: Sketch $\hat{z}$, K iterations, τ step size

$z_0 \leftarrow \hat{z}$;
θ_0 randomly initialized;
for $k = 0, ..., K$ **do**
  Generate a grid $\mathbf{p}$ of P points $p_i \sim \mathcal{U}(\mathcal{D})$;
  Generate a grid $\mathbf{q}$ of P points $q_i \sim \mathcal{U}(\mathcal{D})$;
  $\theta_{k+1} = \theta_k - \tau D_{\mathbf{p},\mathbf{q}}$;
end

Lemma 3.3. *Under the hypotheses of Lemma 3.2, we have*

$$\mathbb{E}_{\mathbf{p},\mathbf{q}} \left(D_{\mathbf{p},\mathbf{q}} \right) = \nabla G(\theta). \tag{28}$$

Proof Use Lemma 3.2, for $h = e_i$ where the e_i form the canonical basis of $\mathbb{R}^{d_0}$. □

We also have that a direct application of Lemma 3.6 permits to bound the variance of this estimator of $\nabla G(\theta)$:

$$\mathbb{E}_{\mathbf{p},\mathbf{q}} \| D_{\mathbf{p},\mathbf{q}} - \nabla G(\theta) \|_2^2 = O\left(\frac{1}{P} \right) \tag{29}$$

This leads to the second theoretical method Algorithm 2.

3.3 Practical implementation of CL-SGD

In practice, we implement the theoretical Algorithms 1 and 2 wih slight differences.

For Algorithm 1,we estimate the normalization constant of μ_θ that best matches μ_θ to the sketch in the least square sense at the current iteration i.e. we minimize for α_p the objective $\| B_{\mathbf{p}} \alpha \mu_\theta - \hat{z} \|_2^2$. This gives the update

$$\alpha_{\mathbf{p}} = \frac{|\langle B_{\mathbf{p}} \mu_\theta, \hat{z} \rangle|}{\| B_{\mathbf{p}} \mu_\theta \|_2^2}. \tag{30}$$

This yields the practical implementation of Algorithm 1 described in Algorithm 3.

For Algorithm 2, we remark that at a given step the important fact is that we have two independent random grids. Hence we just generate one grid at each iteration and use the grid from

Algorithm 3: Practical implementation of naive CL-SGD Algorithm 1

Data: Sketch $\hat{z}$, K iterations, τ step size

$z_0 \leftarrow \hat{z}$;
θ_0 randomly initialized;
for $k = 0, ..., K$ **do**
  Generate a grid $\mathbf{p}$ of P points $p_i \sim \mathcal{U}(\mathcal{D})$;
  $\alpha_{\mathbf{p}} = \frac{|\langle B_{\mathbf{p}}\mu_{\theta_k}, \hat{z}\rangle|}{\|\langle B_{\mathbf{p}}\mu_{\theta_k}\|_2^2}$;
  Set $H_1(\theta) = \|B_{\mathbf{p}}\alpha_{\mathbf{p}}\mu_\theta - \hat{z}\|$;
  $\theta_{k+1} = \theta_k - \tau\nabla H_1(\theta)$;
end

the previous iteration to generate our descent direction. Morevover the descent direction is easily implemented with automatic differentiation by remarking that

$$D_{\mathbf{p},\mathbf{q}} = \nabla_\theta \langle B_{\mathbf{p}}\mu_\theta(\mathbf{p}), l_{\mathbf{q}}\rangle \tag{31}$$

where $l_{\mathbf{q}} = B_{\mathbf{q}}\alpha_p\mu_\theta(\mathbf{q}) - \hat{z}$ is fixed. This leads to Algorithm 4 (which is the practical implementation of Algorithm 2). For this algorithm the previous automatic normalization of the gradient has a tendency to fall in a local minimum (clipping effect due to the fact that $\mu_\theta(p)$ is bounded in $[0, 1]$), however manually setting this parameter yields good results.

Algorithm 4: Practical implementation of Unbiased CL-SGD Algorithm 2

Data: Sketch $\hat{z}$, K iterations, τ step size, normalization α

$z_0 \leftarrow \hat{z}$;
θ_0 randomly initialized;
Generate a grid $\mathbf{q}$ of P points $q_i \sim \mathcal{U}(\mathcal{D})$;
for $k = 0, ..., K$ **do**
  Generate a grid $\mathbf{p}$ of P points $p_i \sim \mathcal{U}(\mathcal{D})$;
  $l_{\mathbf{q}} = B_{\mathbf{q}}\alpha\mu_\theta(\mathbf{q}) - \hat{z}$;
  Set $H_2(\theta) = \mathcal{R}e\langle B_{\mathbf{p}}\alpha\mu_\theta(\mathbf{p}), l_{\mathbf{q}}\rangle$;
  $\theta_{k+1} = \theta_k - \tau\nabla H_2(\theta)$;
  $\mathbf{q} = \mathbf{p}$;
end

Both algorithms update the DNN with information synthetized from the *whole database at each iteration*. The computational cost of each iteration is similar to the cost of computing the gradient of a typical ℓ^2 loss with a dataset of size m (size of the sketch instead of size of the dataset). The advantage of this method is that the number of iterations required to converge to the sketch matching problem does not depend on the size of the original database compared to traditional learning with batches where passes (epochs) through the whole database are required.

3.4 Consistency of CL-SGD with the sketch matching problem

In this section, we give the necessary lemmas to link our stochastic descent directions with the original sketch matching problem (summarized by the Lemmas of Section 3.2). The central idea of our method is that we can generally approximate $S\mu_\theta$ with $B_{\mathbf{p}}\mu_\theta(\mathbf{p})$, which will translate to the chosen stochastic gradients.

Lemma 3.4. *Consider $\mathcal{S}$ constructed with frequencies $(\omega_l)_{l=1}^m$. Let $B_{\mathbf{p}} \in \mathbb{C}^{m\times P}$ with general term $B_{\mathbf{p},l,i} = \frac{e^{-j\langle\omega_l,p_i\rangle}}{P}$ and $\mu \in \mathcal{M}(\mathcal{D})$. Then*

$$\mathbb{E}_{\mathbf{p}}\left(B_{\mathbf{p}}\mu(\mathbf{p})\right) = S\mu. \tag{32}$$

Proof The expectation yields for $l \in \{1, \dots, m\}$

$$[\mathbb{E}_{\mathbf{p}}(B_{\mathbf{p}}\mu(\mathbf{p}))]_l = \mathbb{E}_{\mathbf{p}} \sum_{r=1}^{P} \frac{e^{-j\langle\omega_l,p_r\rangle}\mu(p_r)}{P}. \tag{33}$$

As the p_r are i.i.d, we have

$$\begin{aligned}[\mathbb{E}_{\mathbf{p}}B_{\mathbf{p}}\mu(\mathbf{p})]_l &= P\mathbb{E}_{\mathbf{p}}\frac{e^{-j\langle\omega_l,p_1\rangle}\mu(p_1)}{P} \\ &= \int_{p_1\in\mathcal{D}} e^{-j\langle\omega_l,p_1\rangle}\mu(p_1)dp_1 = [S\mu]_l.\end{aligned} \tag{34}$$

□

This shows that on average random discretization of the data domain for the forward sketching operator is consistent with the original sketch.

To calculate the expectation of our stochastic gradients, we provide the following Lemma which gives the expectation of the discretized cross product between two measures. We write $\langle\mu_1, \mu_2\rangle_{L^2(\mathcal{D})} := \int_{\mathcal{D}} \mu_1(x)\mu_2(x)dx$ the cross product between two densities μ_1 and μ_2.

Lemma 3.5. *Consider $\mathcal{S}$ constructed with frequencies $(\omega_l)_{l=1}^m$. Let $B_{\mathbf{p}} \in \mathbb{C}^{m\times P}$ with general term $B_{\mathbf{p},l,i} = \frac{e^{-j\langle\omega_l,p_i\rangle}}{P}$ and $\mu_1,\mu_2 \in \mathcal{M}(\mathcal{D})$. We have*

$$\mathbb{E}_{\mathbf{p}}\left(\langle B_{\mathbf{p}}\mu_1(\mathbf{p}), B_{\mathbf{p}}\mu_2(\mathbf{p})\rangle\right) = \frac{m}{P}\langle\mu_1,\mu_2\rangle_{L^2(\mathcal{D})} + \frac{P-1}{P}\langle\mathcal{S}\mu_1,\mathcal{S}\mu_2\rangle. \tag{35}$$

Proof We have

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p}}\langle B_{\mathbf{p}}\mu_1(\mathbf{p}), B_{\mathbf{p}}\mu_2(\mathbf{p})\rangle \\
&= \mathbb{E}_{\mathbf{p}}(\mu_2(\mathbf{p})^T B_{\mathbf{p}}^* B_{\mathbf{p}}\mu_1(\mathbf{p})) \\
&= \frac{1}{P^2}\mathbb{E}_{\mathbf{p}}\sum_{t=1}^P \mu_2(p_t)\sum_{g=1}^m e^{j\langle\omega_g,p_t\rangle}\sum_{r=1}^P e^{-j\langle\omega_g,p_r\rangle}\mu_1(p_r) \\
&= \frac{1}{P^2}\sum_{t=1}^P\sum_{g=1}^m\sum_{r=1}^P \mathbb{E}_{\mathbf{p}} e^{j\langle\omega_g,p_t-p_r\rangle}\mu_2(p_t)\mu_1(p_r).
\end{aligned} \tag{36}$$

The diagonal terms in the sum $p_t = p_r$ are

$$D = \frac{1}{P^2}\sum_{t=1}^P\sum_{g=1}^m \mathbb{E}_{\mathbf{p}}\mu_2(p_t)\mu_1(p_t) = \frac{m}{P}\langle\mu_1,\mu_2\rangle_{L^2(\mathcal{D})}. \tag{37}$$

The non diagonal terms $p_t \neq p_r$ give (with the fact that the p_i are i.i.d.):

$$\begin{aligned}
N &= \frac{1}{P^2}\sum_{t=1}^P\sum_{g=1}^m\sum_{r=1,r\neq t}^P \mathbb{E}_{\mathbf{p}} e^{j\langle\omega_g,p_t-p_r\rangle}\mu_2(p_t)\mu_1(p_r) \\
&= \frac{P-1}{P}\sum_{g=1}^m\left(\mathbb{E}_{\mathbf{p}} e^{j\langle\omega_g,p_1\rangle}\mu_2(p_1)\right)\left(\mathbb{E}_{\mathbf{p}} e^{-j\langle\omega_g,p_1\rangle}\mu_1(p_1)\right) \\
&= \frac{P-1}{P}\sum_{g=1}^m(\mathcal{S}\mu_2)_g^*(\mathcal{S}\mu_1)_g \\
&= \frac{P-1}{P}\langle\mathcal{S}\mu_1,\mathcal{S}\mu_2\rangle.
\end{aligned} \tag{38}$$

□

We also calculate the variance of the unbiased estimator of the gradient of G thanks to the following Lemma.

Lemma 3.6. *Consider $\mathcal{S}$ constructed with frequencies $(\omega_l)_{l=1}^m$. Let $B_{\mathbf{p}} \in \mathbb{C}^{m\times P}$ with general term $B_{p,l,i} = \frac{e^{-j\langle\omega_l,p_i\rangle}}{P}$ and $\mu_1,\mu_2 \in \mathcal{M}(\mathcal{D})$. We have*

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p},\mathbf{q}}|\langle B_{\mathbf{p}}\mu_1(\mathbf{p}), B_{\mathbf{q}}\mu_2(\mathbf{q}) - z\rangle|^2 \\
&\qquad - |\mathbb{E}_{\mathbf{p},\mathbf{q}}\langle B_{\mathbf{p}}\mu_1(\mathbf{p}), B_{\mathbf{q}}\mu_2(\mathbf{q}) - z\rangle|^2 \\
&= \frac{1}{P^2}\langle|\mu_1|^2,|\mu_2|^2\rangle_{L^2,|\mathcal{S}^*\mathbf{1}|^2} + \frac{1}{P}C(\mu_1,\mu_2,z)
\end{aligned} \tag{39}$$

where we define $\mathcal{S}^*z : p \to \sum_g z_g e^{j\langle\omega_g,p\rangle}$, and for a kernel K (a function from $\mathcal{D}$ to $\mathbb{R}$), $\langle\nu_1,\nu_2\rangle_{L^2(\mathcal{D}),K} := \int_{x,y}\nu_1(x)\nu_2(y)h(x-y)dxdy$. and where

$$\begin{aligned}
&C(\mu_1,\mu_2,z) \\
&:= \frac{P-1}{P}(\langle|\mathcal{S}^*\mathcal{S}\mu_1|^2,|\mu_2|^2\rangle_{L^2} + \langle|\mathcal{S}^*\mathcal{S}\mu_2|^2,|\mu_1|^2\rangle_{L^2}) \\
&+ \mathcal{R}e\langle|\mathcal{S}^*z|^2 - 2\mathcal{S}^*z(\mathcal{S}^*\mathcal{S}\mu_2)^*,|\mu_1|^2\rangle_{L^2} \\
&+ 2\mathcal{R}e\left(\langle\mathcal{S}\mu_1,z\rangle\langle\mathcal{S}\mu_1,\mathcal{S}\mu_2\rangle^*\right) - |\langle\mathcal{S}\mu_1,z\rangle|^2 \\
&- \frac{2P-1}{P}|\langle\mathcal{S}\mu_1,\mathcal{S}\mu_2\rangle|^2
\end{aligned} \tag{40}$$

Proof We need to calculate a few terms separately. For $y \in \mathbb{C}^m$, using the fact that

$$\langle B_{\mathbf{p}}\mu_1(\mathbf{p}), z\rangle = \sum_{g=1}^m\sum_{t=1}^P e^{j\langle\omega_g,p_t\rangle}\mu_1(p_t)z_g,$$

we have

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p}}\langle B_{\mathbf{p}}\mu_1(\mathbf{p}), z\rangle\langle B_{\mathbf{p}}\mu_1(\mathbf{p}), y\rangle^* \\
&= \frac{1}{P^2}\sum_{g,t}\sum_{\tilde{g},\tilde{t}}\mathbb{E}_{\mathbf{p}}\left(e^{j\langle\omega_g,p_t\rangle - j\langle\omega_{\tilde{g}},p_{\tilde{t}}\rangle}\mu_1(p_t)\mu_1(p_{\tilde{t}})z_g y_{\tilde{g}}^*\right) \\
&= \frac{1}{P^2}\sum_{g,\tilde{g}} z_g y_{\tilde{g}}^*\sum_{t,\tilde{t}}\mathbb{E}_{\mathbf{p}}\left(e^{j\langle\omega_g,p_t\rangle - j\langle\omega_{\tilde{g}},p_{\tilde{t}}\rangle}\mu_1(p_t)\mu_1(p_{\tilde{t}})\right).
\end{aligned} \tag{41}$$

As the p_t are i.i.d., we have

$$\begin{aligned}
&\sum_{t,\tilde{t}}\mathbb{E}_{\mathbf{p}} e^{j\langle\omega_g,p_t\rangle - j\langle\omega_{\tilde{g}},p_{\tilde{t}}\rangle}\mu_1(p_t)\mu_1(p_{\tilde{t}}) \\
&\qquad = P\mathbb{E}_{\mathbf{p}}[e^{-j\langle\omega_{\tilde{g}}-\omega_g,p_1\rangle}|\mu_1(p_1)|^2] \\
&\qquad + P(P-1)(\mathcal{S}\mu_1)_g^*(\mathcal{S}\mu_1)_{\tilde{g}}.
\end{aligned} \tag{42}$$

We obtain

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p}}\langle B_{\mathbf{p}}\mu_1(\mathbf{p}),z\rangle\langle B_{\mathbf{p}}\mu_1(\mathbf{p}),y\rangle^*\\
&=\frac{1}{P^2}\sum_{g,\tilde{g}}\Big(z_g y_{\tilde{g}}^* P\mathbb{E}_{\mathbf{p}}[e^{-j\langle\omega_{\tilde{g}}-\omega_g,p_1\rangle}|\mu_1(p_1)|^2]\\
&+P(P-1)(\mathcal{S}\mu_1)_g^*(\mathcal{S}\mu_1)_{\tilde{g}}z_g y_{\tilde{g}}^*\Big)\\
&=\frac{1}{P}\mathbb{E}_{\mathbf{p}}[B_{\mathbf{p}}^*z(B_{\mathbf{p}}^*y)^*](p_1)|^2|\mu_1(p_1)|^2]\\
&+\frac{P-1}{P}\langle\mathcal{S}\mu_1,z\rangle\langle\mathcal{S}\mu_1,y\rangle^*\\
&=\frac{1}{P}\langle\mathcal{S}^*z(\mathcal{S}^*y)^*,|\mu_1|^2\rangle_{L^2}+\frac{P-1}{P}\langle\mathcal{S}\mu_1,z\rangle\langle\mathcal{S}\mu_1,y\rangle^*
\end{aligned}\tag{43}$$

where $\mathcal{S}^*z$ is defined in the hypotheses of the Lemma. For $y=z$ we obtain

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p}}|\langle B_{\mathbf{p}}\mu_1(\mathbf{p}),z\rangle|^2\\
&=\frac{1}{P}\langle|\mathcal{S}^*z|^2,|\mu_1|^2\rangle_{L^2}+\frac{P-1}{P}|\langle\mathcal{S}\mu_1,z\rangle|^2
\end{aligned}\tag{44}$$

We now calculate the following expectation:

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p},\mathbf{q}}|\langle B_{\mathbf{p}}\mu_1(\mathbf{p}),B_{\mathbf{q}}\mu_2(\mathbf{q})\rangle|^2\\
&=\frac{1}{P^4}\mathbb{E}_{\mathbf{p},\mathbf{q}}|\sum_{g=1}^{m}\sum_{t=1}\sum_{r=1}e^{j\langle\omega_g,p_t-q_r\rangle}\mu_1(p_t)\mu_2(q_r)|^2\\
&=\frac{1}{P^4}\sum_{g,t,r}\sum_{\tilde{g},\tilde{t},\tilde{r}}\mathbb{E}_{\mathbf{p},\mathbf{q}}\Big(e^{j\langle\omega_g,p_t-q_r\rangle}e^{-j\langle\omega_{\tilde{g}},p_{\tilde{t}}-q_{\tilde{r}}\rangle}\\
&\mu_1(p_t)\mu_2(q_r)\mu_1(p_{\tilde{t}})\mu_2(q_{\tilde{r}})\Big)
\end{aligned}\tag{45}$$

As $\mathbf{p}$ and $\mathbf{q}$ are i.i.d.,

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p},\mathbf{q}}|\langle B_{\mathbf{p}}\mu_1(\mathbf{p}),B_{\mathbf{q}}\mu_2(\mathbf{q})\rangle|^2\\
&=\frac{1}{P^4}\sum_{g,t,r}\sum_{\tilde{g},\tilde{t},\tilde{r}}\Big(\mathbb{E}_{\mathbf{p}}e^{j\langle\omega_g,p_t\rangle-j\langle\omega_{\tilde{g}},p_{\tilde{t}}\rangle}\mu_1(p_t)\mu_1(p_{\tilde{t}})\Big)\\
&\mathbb{E}_{\mathbf{q}}\Big(e^{-j\langle\omega_g,q_r\rangle+j\langle\omega_{\tilde{g}},q_{\tilde{r}}\rangle}\mu_2(q_r)\mu_2(q_{\tilde{r}})\Big)\\
&=\frac{1}{P^4}\sum_{g,\tilde{g}}\sum_{t,\tilde{t}}\mathbb{E}_{\mathbf{p}}e^{j\langle\omega_g,p_t\rangle-j\langle\omega_{\tilde{g}},p_{\tilde{t}}\rangle}\mu_1(p_t)\mu_1(p_{\tilde{t}})\\
&\sum_{r,\tilde{r}}\mathbb{E}_{\mathbf{q}}e^{-j\langle\omega_g,q_r\rangle+j\langle\omega_{\tilde{g}},q_{\tilde{r}}\rangle}\mu_2(q_r)\mu_2(q_{\tilde{r}})\\
&=\frac{1}{P^4}\sum_{g,\tilde{g}}A_{1,g,\tilde{g}}A_{2,g,\tilde{g}}^*
\end{aligned}\tag{46}$$

where, with the decomposition of the sum into diagonal and off-diagonal terms,

$$\begin{aligned}
A_{i,g,\tilde{g}}&=\sum_{t,\tilde{t}}\mathbb{E}_{\mathbf{p}}e^{j\langle\omega_g,p_t\rangle-j\langle\omega_{\tilde{g}},p_{\tilde{t}}\rangle}\mu_i(p_t)\mu_i(p_{\tilde{t}})\\
&=P\mathbb{E}_{\mathbf{p}}e^{-j\langle\omega_{\tilde{g}}-\omega_g,p_t\rangle}|\mu_i(p_t)|^2\\
&\quad+P(P-1)(\mathcal{S}\mu_i)_g^*(\mathcal{S}\mu_i)_{\tilde{g}}.
\end{aligned}\tag{47}$$

We obtain

$$\begin{aligned}
&P^2\mathbb{E}_{\mathbf{p},\mathbf{q}}|\langle B_{\mathbf{p}}\mu_1(\mathbf{p}),B_{\mathbf{q}}\mu_2(\mathbf{q})\rangle|^2\\
&=\sum_{g,\tilde{g}}(\mathbb{E}_{\mathbf{p}}e^{-j\langle\omega_{\tilde{g}}-\omega_g,p_t\rangle}|\mu_1(p_t)|^2\mathbb{E}_{\mathbf{q}}e^{j\langle\omega_{\tilde{g}}-\omega_g,q_r\rangle}|\mu_2(q_r)|^2\\
&+(P-1)(\mathcal{S}\mu_1)_g^*(\mathcal{S}\mu_1)_{\tilde{g}}\mathbb{E}_{\mathbf{q}}e^{j\langle\omega_{\tilde{g}}-\omega_g,q_r\rangle}|\mu_2(q_r)|^2\\
&+(P-1)(\mathcal{S}\mu_2)_g(\mathcal{S}\mu_2)_{\tilde{g}}^*\mathbb{E}_{\mathbf{p}}e^{-j\langle\omega_{\tilde{g}}-\omega_g,p_t\rangle}|\mu_1(p_t)|^2\\
&+(P-1)^2(\mathcal{S}\mu_1)_g^*(\mathcal{S}\mu_1)_{\tilde{g}}(\mathcal{S}\mu_2)_g(\mathcal{S}\mu_2)_{\tilde{g}}^*
\end{aligned}\tag{48}$$

with

$$\begin{aligned}
&\sum_{g,\tilde{g}}(\mathbb{E}_{\mathbf{p}}e^{-j\langle\omega_{\tilde{g}}-\omega_g,p_t\rangle}|\mu_1(p_t)|^2\mathbb{E}_{\mathbf{q}}e^{j\langle\omega_{\tilde{g}}-\omega_g,q_r\rangle}|\mu_2(q_r)|^2\\
&=\mathbb{E}_{\mathbf{p}}\mathbb{E}_{\mathbf{q}}\left(\sum_{g,\tilde{g}}e^{-j\langle\omega_{\tilde{g}}-\omega_g,p_t\rangle}e^{j\langle\omega_{\tilde{g}}-\omega_g,q_r\rangle}\right)|\mu_1(p_t)|^2|\mu_2(q_r)|^2
\end{aligned}\tag{49}$$

We have inside the expectation,

$$\begin{aligned}
&\left(\sum_{g,\tilde{g}}e^{j\langle\omega_{\tilde{g}},q_r-p_t\rangle}e^{-j\langle\omega_g,q_r-p_t\rangle}\right)|\mu_1(p_t)|^2|\mu_2(q_r)|^2\\
&=\left(\sum_{\tilde{g}}e^{j\langle\omega_{\tilde{g}},q_r-p_t\rangle}\sum_{g}e^{-j\langle\omega_g,q_r-p_t\rangle}\right)|\mu_1(p_t)|^2|\mu_2(q_r)|^2\\
&=\left(\sum_{\tilde{g}}e^{j\langle\omega_{\tilde{g}},q_r-p_t\rangle}\sum_{g}e^{-j\langle\omega_g,q_r-p_t\rangle}\right)|\mu_1(p_t)|^2|\mu_2(q_r)|^2
\end{aligned}\tag{50}$$

This gives

$$\begin{aligned}
&P^2\mathbb{E}_{\mathbf{p},\mathbf{q}}|\langle B_{\mathbf{p}}\mu_1(\mathbf{p}),B_{\mathbf{q}}\mu_2(\mathbf{q})\rangle|^2\\
&=\mathbb{E}_{\mathbf{p},\mathbf{q}}|[\mathcal{S}^*\mathbf{1}](q_r-p_t)|^2|\mu_1(p_t)|^2|\mu_2(q_r)|^2\\
&=\langle|\mu_1|^2,|\mu_2|^2\rangle_{L^2(\mathcal{D}),|\mathcal{S}^*\mathbf{1}|^2}.
\end{aligned}\tag{51}$$

where we define for a kernel K (a function from $\mathcal{D}$ to $\mathbb{R}$), $\langle\nu_1,\nu_2\rangle_{L^2(\mathcal{D}),K}=\int_{x,y}\nu_1(x)\nu_2(y)h(x-y)dxdy$.

We calculate the second term (and similarly the third term) of the right hand side of (48).

$$\begin{aligned}
&\sum_{g,\tilde{g}}(\mathcal{S}\mu_1)_g^*(\mathcal{S}\mu_1)_{\tilde{g}}\mathbb{E}_{\mathbf{q}}e^{j\langle\omega_{\tilde{g}}-\omega_g,q_r\rangle}|\mu_2(q_r)|^2\\
&\mathbb{E}_{\mathbf{q}}\sum_{g,\tilde{g}}e^{-j\langle\omega_g,q_r\rangle}(\mathcal{S}\mu_1)_g^*e^{j\langle\omega_{\tilde{g}},q_r\rangle}(\mathcal{S}\mu_1)_{\tilde{g}}|\mu_2(q_r)|^2\\
&=\mathbb{E}_{\mathbf{q}}\left(|\mathcal{S}^*\mathcal{S}\mu_1|_{q_r}^2|\mu_2(q_r)|^2\right)=\langle|\mathcal{S}^*\mathcal{S}\mu_1|^2,|\mu_2|^2\rangle_{L^2(\mathcal{D})}.
\end{aligned}\tag{52}$$

The fourth term of the right hand side of (48) yields

$$\sum_{g,\tilde{g}} (\mathcal{S}\mu_1)^*_g (\mathcal{S}\mu_1)_{\tilde{g}} (\mathcal{S}\mu_2)_g (\mathcal{S}\mu_2)^*_{\tilde{g}}$$
$$= \sum_{g} (\mathcal{S}\mu_1)^*_g (\mathcal{S}\mu_2)_g \langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle = |\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle|^2. \tag{53}$$

Going back to (46), we have using the expressions (51), (52) and (53) in (48)

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) \rangle|^2 \\
&= \frac{1}{P^2} \langle |\mu_1|^2, |\mu_2|^2 \rangle_{L^2(\mathcal{D}), |\mathcal{S}^* \mathbf{1}|^2} \\
&+ \frac{P-1}{P^2} \langle |\mathcal{S}^* \mathcal{S}\mu_1|^2, |\mu_2|^2 \rangle_{L^2(\mathcal{D})} \\
&+ \frac{P-1}{P^2} \langle |\mathcal{S}^* \mathcal{S}\mu_2|^2, |\mu_1|^2 \rangle_{L^2(\mathcal{D})} \\
&+ \frac{(P-1)^2}{P^2} |\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle|^2.
\end{aligned} \tag{54}$$

By developing expressions, we have

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) - z \rangle \\
&\qquad - \mathbb{E}_{\mathbf{p},\mathbf{q}} \langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) - z \rangle|^2 \\
&= \mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) - z \rangle|^2 - |\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 - z \rangle|^2 \\
&= \mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) \rangle|^2 \\
&- 2\mathbb{E}_{\mathbf{p},\mathbf{q}} \mathcal{R}e \left(\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), z \rangle \langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) \rangle^* \right) \\
&+ \mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), z \rangle|^2 - |\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 - z \rangle|^2 \\
&= \mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) \rangle|^2 \\
&- 2\mathbb{E}_{\mathbf{p},\mathbf{q}} \mathcal{R}e \left(\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), z \rangle \langle B_{\mathbf{p}} \mu_1(\mathbf{p}), \mathcal{S}\mu_2 \rangle^* \right) \\
&+ \mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), z \rangle|^2 - |\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 - z \rangle|^2
\end{aligned} \tag{55}$$

Using equation (54) and the fact that $\mathbb{E}_{\mathbf{p},\mathbf{q}} \mathcal{R}e \langle B_{\mathbf{p}} \mu_1(\mathbf{p}), z \rangle = \mathcal{R}e \langle \mathbb{E}_{\mathbf{p},\mathbf{q}} B_{\mathbf{p}} \mu_1(\mathbf{p}), z \rangle = \mathcal{R}e \langle \mathcal{S}\mu_1, z \rangle$ with Lemma 3.4, equation (43), we have

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) - z \rangle \\
&\qquad - \mathbb{E}_{\mathbf{p},\mathbf{q}} \langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) - z \rangle|^2 \\
&= \frac{1}{P^2} \langle |\mu_1|^2, |\mu_2|^2 \rangle_{L^2(\mathcal{D}), |\mathcal{S}^* \mathbf{1}|^2} \\
&+ \frac{P-1}{P^2} \langle |\mathcal{S}^* \mathcal{S}\mu_1|^2, |\mu_2|^2 \rangle_{L^2(\mathcal{D})} \\
&+ \frac{P-1}{P^2} \langle |\mathcal{S}^* \mathcal{S}\mu_2|^2, |\mu_1|^2 \rangle_{L^2(\mathcal{D})} \\
&+ \frac{(P-1)^2}{P^2} |\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle|^2 \\
&- 2\frac{1}{P} \mathcal{R}e \langle (\mathcal{S}^* z)(\mathcal{S}^* \mathcal{S}\mu_2)^*, |\mu_1|^2 \rangle_{L^2} \\
&- 2\frac{P-1}{P} \mathcal{R}e \left(\langle \mathcal{S}\mu_1, z \rangle \langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle^* \right) \\
&+ \frac{1}{P} \langle |\mathcal{S}^* z|^2, |\mu_1|^2 \rangle_{L^2(\mathcal{D})} \\
&+ \frac{P-1}{P} |\langle \mathcal{S}\mu_1, z \rangle|^2 \\
&- |\langle \mathcal{S}\mu_1, z \rangle|^2 \\
&+ 2\mathcal{R}e \left(\langle \mathcal{S}\mu_1, z \rangle \langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle^* \right) \\
&- |\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle|^2
\end{aligned} \tag{56}$$

Regrouping terms yields

$$\begin{aligned}
&\mathbb{E}_{\mathbf{p},\mathbf{q}} |\langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) - z \rangle \\
&\qquad - \mathbb{E}_{\mathbf{p},\mathbf{q}} \langle B_{\mathbf{p}} \mu_1(\mathbf{p}), B_{\mathbf{q}} \mu_2(\mathbf{q}) - z \rangle|^2 \\
&= \frac{1}{P^2} \langle |\mu_1|^2, |\mu_2|^2 \rangle_{L^2, |\mathcal{S}^* \mathbf{1}|^2} \\
&+ \frac{P-1}{P^2} \left(\langle |\mathcal{S}^* \mathcal{S}\mu_1|^2, |\mu_2|^2 \rangle_{L^2} + \langle |\mathcal{S}^* \mathcal{S}\mu_2|^2, |\mu_1|^2 \rangle_{L^2} \right) \\
&+ \frac{1}{P} \mathcal{R}e \langle |\mathcal{S}^* z|^2 - 2\mathcal{S}^* z (\mathcal{S}^* \mathcal{S}\mu_2)^*, |\mu_1|^2 \rangle_{L^2} \\
&+ \frac{2}{P} \mathcal{R}e \left(\langle \mathcal{S}\mu_1, z \rangle \langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle^* \right) \\
&- \frac{1}{P} |\langle \mathcal{S}\mu_1, z \rangle|^2 \\
&- \frac{2P-1}{P^2} |\langle \mathcal{S}\mu_1, \mathcal{S}\mu_2 \rangle|^2
\end{aligned} \tag{57}$$

□

We have that the variance converges to 0 at the typical rate $1/P$.

3.5 Convergence analysis

The advantage of Lemma 3.3 showing the consistency in expectation of our CL-SGD update with the gradient of the ideal sketch matching problem is that we can use out of the box results of the convergence of SGD. Under some hypotheses on

the stochastic descent direction, it is possible de to show that

$$\lim_{k\to\infty} \inf \mathbb{E}\|\nabla G(\theta_k)\|_2^2 = 0 \tag{58}$$

i.e., in expectation, our estimate converges to a critical point of G (a point θ such that $\nabla G(\theta) = 0$) which is the best we can hope for in this non-convex setting (with minimal hypotheses).

Specifically, we can use results in the non-convex setting from the review article [1, Theorem 4.9]. We suppose that G has Lipschitz gradient, G is lower bounded, that the expectation of our descent direction is exactly the gradient of G and the variance is bounded by $M_1 + M_2\|\nabla G(\theta)\|_2^2$ (which is verified with $M_2 = 0$ as long as the μ_θ are bounded in L^2 norm thanks to Lemma 3.6). Then with a sequence of diminishing step sizes $\tau_k > 0$ such that $\|\sum_k \tau_k\| = \infty$ and $\|\sum_k \tau_k^2\| < \infty$ (tipically one may choose $\tau_k = \frac{1}{k}$), we have that the equation (58) is verified. With fixed step sizes the convergence results include the variance.

4 Experimental results

4.1 Synthetic data

We first test our proposed approaches with 2-D synthetic data. The used training dataset is made of $n = 10^6$ samples which are generated from a spiral with a radius of circular curve from 0.3 to 1 and spiral length 2π (shown as (a) in Fig. 1). The ReLU network f_θ with 3 hidden layers, each layer contains 64, 64, 128 neurons respectively. The dataset is compressed into a sketch of size $m = 500$, i.e. with a compression ratio $r = 2000$.

Figure 1 (b) shows the prior model learned using the initial method, i.e. the method from [19] described in equation (14), with a sketch of size $m = 500$, $P = 900$ points uniformly generated on the grid of the data domain. Figure 1 (c) and (d) shows the prior model learned with Algorithm 3 and 4. We use $P = 1600$ points randomly generated on the data domain for algorithm 3. For algorithm 4, we use $P = 1000$ points and fix the value of $\alpha = 50$. The proposed algorithms are capable of recovering good approximations of the probability distribution of sample data while taking about half the training time (about 6 minutes) of the initial method.

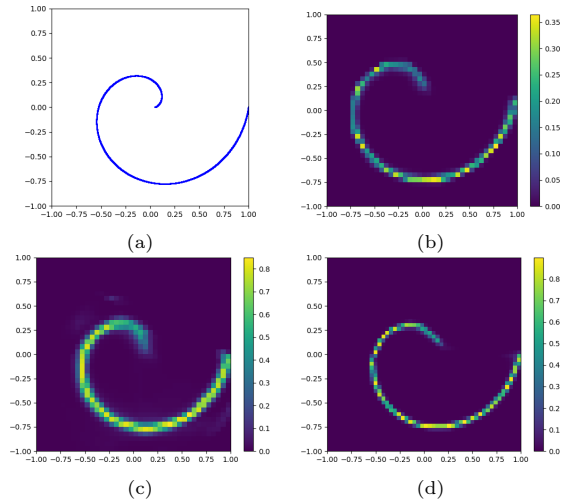

Fig. 1 Results for the learning of densities. (a) The training data. The prior model learned with (b) regular discretization (previous work [19]), (c) Naïve CL-SGD Algorithm 3 and (d) Unbiased SL-SGD Algorithm 4.

Denoising results

Similar to [19], we apply the learned regularization term to solve the variational problem defined in equation (3). This results in the minimization of:

$$G(u) = \|u - v\|_2^2 + \lambda\|f_\theta(u)\|_2^2. \tag{59}$$

Furthermore, we can conveniently compute the gradient using automatic differentiation. It is important to mention that this denoising approach can be seamlessly extended to address diverse linear inverse problems (beyond the scope of this article). It is indeed now well known with plug and play approach that denoisers capture enough information on the data for the solving of inverse problems.

The effectiveness of the learned regularizers is evaluated within the context of denoising white Gaussian noise. Specifically, the noisy dataset consists of 500 samples generated with a noise level of $\sigma^2 = 0.15$. We manually select the optimal hyperparameter values, including the gradient step size and the regularization parameter λ, for each individual model.

Figure 2 visually illustrates the 2-D denoising results with different noise levels $\sigma^2 = 0.15, 0.2$. From top to bottom, the figure shows the denoising results using regularizers learned from the compressed dataset 2000 times smaller with the

initial method [19], Algorithm 3 and 4 respectively. Table 1 shows the average gain on SNR (Signal to Noise Ratio) of denoising results using different models. We observe unbiased CL-SGD yields the best performance. However it must be noted that its parametrization is much harder than Naïve SGD in practive.

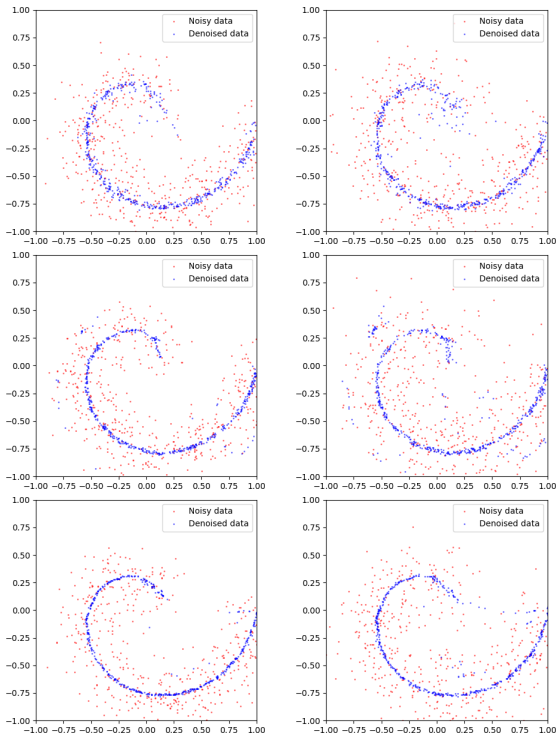

Fig. 2 Denoising results with noise level $\sigma = 0.15$ (left) and $\sigma = 0.2$ (right). Regularizers learned from 2000 times compressed dataset ($m = 500$) with initial method [19] (1st row), Algorithm 3 (2nd row) and Algorithm 4 (3rd row).

Table 1 Average gain on SNR

Gain SNR	Inital method	Algo 3	Algo 4
$\sigma^2 = 0.15$	+2.065	+1.954	+2.89
$\sigma^2 = 0.2$	+1.613	+1.583	+2.45

4.2 Audio denoising

To illustrate the advantages of the proposed methods in the same setting as [19] on real data (as we were unable to deal with dimensions larger than 3 with the regular discretization of $\mathcal{S}$), we perform experiments on recorded musical notes (monophonic 16kHz audio snippets) from the NSynth dataset [3]. To compare, we use the same compressed dataset as in the previous work. That is, the training dataset comprises 0.125 seconds of audio extracted from an acoustic guitar. After filtering the normalized audio data s by two 4th-order Butterworth low-pass filters h_1 and h_2 with a cutoff frequency of 1.5kHz and 3.75kHz, three frequency responses are constructed with $s_1 = h_1 * s$, $s_2 = h2 * (s - s_1)$, and $s_3 = s - s1 - s2$. Then the frequency responses are concatenated, hence the training set is of dimension 2000×3; i.e. 2000 samples in dimension 3. The regularizer is learned from a sketch of size $m = 200$, *i.e.* the dataset is compressed by a factor of 30. Once the regularizer is learned, it is applied to denoise the audio that has been corrupted by Gaussian white noise at noise level $\sigma^2 = 0.1$.

Figure 3 and Figure 4 demonstrate the audio denoising results. We gain 1.49dB on SNR with Algorithm 3 and 1.97dB with Algorithm 4 in the case of small noise ($\sigma^2 = 0.1$). Worse denoising results (gain 1.36dB) are achieved by using the initial approach (14) in the previous work [19]. Compared to the initial method, which use $P = 8000$ points to discretize the sketching operator, the results displayed in the figure is achieved with a model learned with $P = 4000$ points. Consequently, the new proposed algorithms exhibits enhanced efficiency (4 times faster) compared to the initial approach.

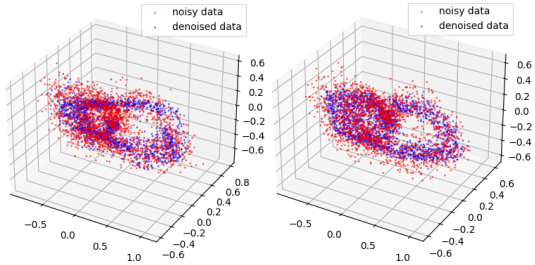

Fig. 3 Audio denoising results with regularizers learned from a 30 times compressed dataset with Algorithm 3 (left) and 4 (right).

4.3 Image denoising results

We show that our proposed method can be applied in the higher dimensional context of patch-based image denoising. The initial training data comprises $n = 4 \times 10^6$ patches of size 3×3 (i.e.

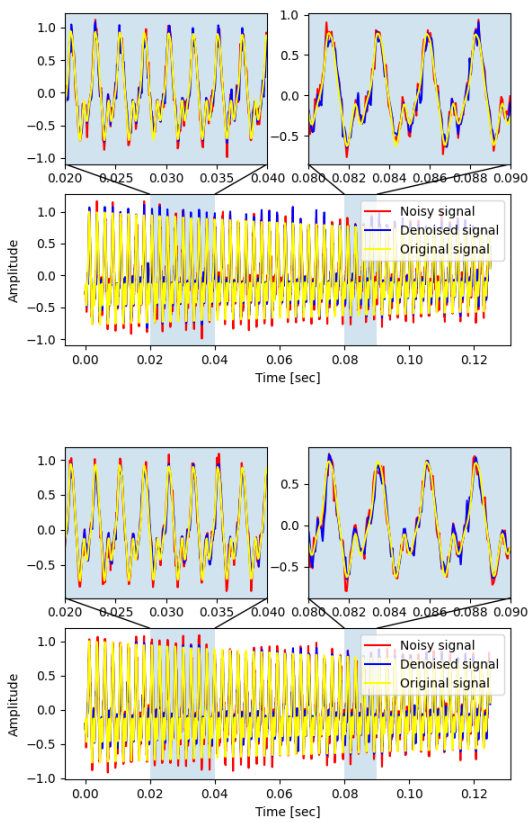

Fig. 4 Audio results with regularizers learned from a 30 times compressed dataset with Algorithm 3 (top) and 4 (bottom).

$d = 9$). These patches are compressed into a sketch of size $m = 10^4$. Figure 5 shows test images (1st row) and their noisy version (2nd row) used in the experiment (with a noise level of $\sigma = 0.07$.

It also shows the denoising result achieved by using a regularizer learned via Algorithm 3 (4th row). The regularizer is trained using a ReLU network consisting of 5 hidden layers, with respectively 64, 64, 128, 196, 196 neurons in each layer. We use $P = 8 \times 10^4$ random points on the data domain and the Adam optimizer with a learning rate of 10^{-3}. The learning process takes 5.7 hours on a machine with 2 * AMD EPYC 7452 32-Core Processor and 256 GB RAM.

We compare the denoising results with that achieved with a GMM regularizer learned from the same sketch using the method from [18] (shown in the 3rd row in Figure 5). Note that only one iteration of the denoising method is used with an optimal choice of regularization parameter for all methods. Hence we observe directly the denoising effect of the two different regularizers. This comparison demonstrates that Algorithm 3 yields similar result as the GMM method, thus showing that CL-SGD can be used successfully for the learning of a deep prior from a sketch in high dimension. We attribute the fact that the denoising performance is not increased to the limited denoising possibilites on independent 3×3 patches. Further work on accelerating our algorithms to manage bigger patches should show the improved representation capabilities of DNN.

We present denoising results with Unbiased CL-SGD in Figure 6. While Algorithm 4 performs a strong denoising visually, we observe a lack of preservation of the contrast of the original image (hence the PSNR metric is not improved by this method). Unfortunately, the difficulty of parametrization that we observe in the 2D synthetic context becomes even more harder in dimension 9 (3×3 patches). This makes the study of acceleration method fo unbiased CL-SGD even more important for an easy use in the context of image denoising. Moreover forcing some knowledge on the regularizer (e.g. $R_\theta(u) = 0$ if u is a constant patch) might facilitate its training process.

5 Conclusions

In this work, we adapt the sketching framework to the learning of a regularizer parametrized by a DNN. This is achieved with a new stochastic gradient descent algorithm, CL-SGD, with convergence guarantees where randomness is used for the discretization of the sketching operator. Our method outperforms the previous work based on the deterministic discretization of the forward operator $\mathcal{S}$ on a regular grid. Moreover, it must be noted that only a database of clean images suffices to learn the deep prior. Experimental results obtained from synthetic 2-D data, audio denoising (data in 3D), and real images validates CL-SGD both in terms of computational efficiency and quality of the trained regularizer.

Many questions arise from this work. While the design of the sketching operator and its theoretical justification was heavily influenced by the prior model (e.g. GMM), this design should be revisited and generalized to DNN based priors. From a practical perspective, it would be

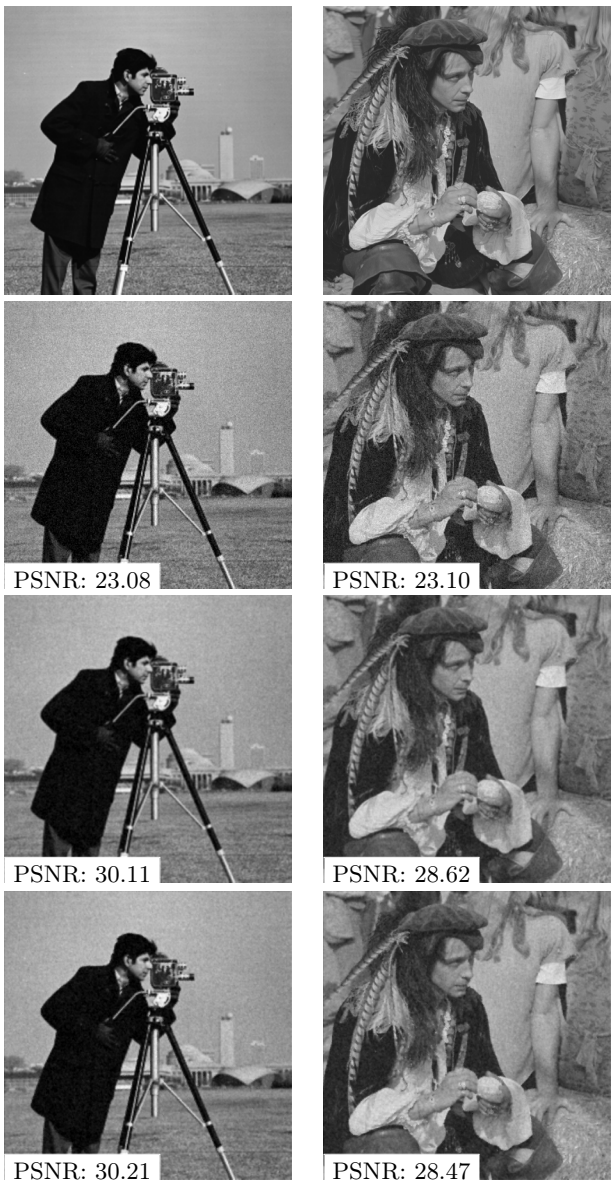

Fig. 5 Original test images (1st row). Noisy test images (2nd row) with noise level $\sigma = 0.07$. Denoised images with 10-GMM learned from the sketch (3rd row). Denoised images with regularizers learned with Naive CL-SGD (Algorithm 3) (4th row).

interesting to compare denoising results with our compressive approach on huge image databases with plug and play approaches where a training process is performed on pairs of noisy/clean images. Also acceleration methods (with inertia) should be investigated to improve the learning time of the deep prior.

Fig. 6 Denoised images with regularizers learned with Unbiased CL-SGD (Algorithm 4).

References

[1] Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning. SIAM review **60**(2), 223–311 (2018)

[2] Demoment, G.: Image reconstruction and restoration: Overview of common estimation structures and problems. IEEE Transactions on Acoustics, Speech, and Signal Processing **37**(12), 2024–2036 (1989)

[3] Engel, J., Resnick, C., Roberts, A., Dieleman, S., Norouzi, M., Eck, D., Simonyan, K.: Neural audio synthesis of musical notes with wavenet autoencoders. In: Int. Conf. on Machine Learning. pp. 1068–1077. PMLR (2017)

[4] Gribonval, R., Blanchard, G., Keriven, N., Traonmilin, Y.: Compressive statistical learning with random feature moments. Mathematical Statistics and Learning **3**(2), 113–164 (2021)

[5] Gribonval, R., Blanchard, G., Keriven, N., Traonmilin, Y.: Statistical learning guarantees for compressive clustering and compressive mixture modeling. Mathematical Statistics and Learning **3**(2), 165–257 (2021)

[6] Gribonval, R., Chatalic, A., Keriven, N., Schellekens, V., Jacques, L., Schniter, P.: Sketching datasets for large-scale learning (long version). arXiv preprint arXiv:2008.01839 (2020)

[7] Gribonval, R., Chatalic, A., Keriven, N., Schellekens, V., Jacques, L., Schniter, P.: Sketching data sets for large-scale learning: Keeping only what you need. IEEE Signal Processing Magazine **38**(5), 12–36 (2021)

[8] Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks **2**(5),

359–366 (1989)
[9] Hurault, S., Leclaire, A., Papadakis, N.: Gradient step denoiser for convergent plug-and-play. In: International Conference on Learning Representations (2022)
[10] Keriven, N., Bourrier, A., Gribonval, R., Pérez, P.: Sketching for large-scale learning of mixture models. Information and Inference **7**(3), 447–508 (2018)
[11] Kobler, E., Effland, A., Kunisch, K., Pock, T.: Total deep variation: A stable regularization method for inverse problems. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–1 (2021)
[12] Lunz, S., Öktem, O., Schönlieb, C.B.: Adversarial regularizers in inverse problems. Advances in Neural Information Processing systems **31** (2018)
[13] Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Int. Conf. on Machine Learning (2010)
[14] Pan, X., Srikumar, V.: Expressiveness of rectifier networks. In: Int. Conf. on Machine Learning. pp. 2427–2435. PMLR (2016)
[15] Prost, J., Houdard, A., Almansa, A., Papadakis, N.: Learning local regularization for variational image restoration. In: Int. Conf. on Scale Space and Variational Methods in Computer Vision. pp. 358–370. Springer (2021)
[16] Schellekens, V., Jacques, L.: Compressive classification (machine learning without learning). arXiv preprint arXiv:1812.01410 (2018)
[17] Schellekens, V., Jacques, L.: Compressive learning of generative networks. arXiv preprint arXiv:2002.05095 (2020)
[18] Shi, H., Traonmilin, Y., Aujol, J.F.: Compressive learning for patch-based image denoising. SIAM Journal on Imaging Sciences **15**(3), 1184–1212 (2022)
[19] Shi, H., Traonmilin, Y., Aujol, J.F.: Compressive learning of deep regularization for denoising. In: International Conference on Scale Space and Variational Methods in Computer Vision. pp. 162–174. Springer (2023)
[20] Venkatakrishnan, S.V., Bouman, C.A., Wohlberg, B.: Plug-and-play priors for model based reconstruction. In: IEEE Global Conf. on Signal and Information Processing. pp. 945–948. IEEE (2013)
[21] Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. In: Int. Conf. on Computer Vision. pp. 479–486. IEEE (2011)