A Framework for Dynamic Image Sampling Based on Supervised Learning

G. M. Dilshan P. Godaliyadda, *Student Member, IEEE*, Dong Hye Ye, *Member, IEEE*, Michael D. Uchic, Michael A. Groeber, Gregery T. Buzzard, *Member, IEEE*, and Charles A. Bouman, *Fellow, IEEE*

***Abstract*—Sparse sampling schemes can broadly be classified into two main categories: static sampling where the sampling pattern is predetermined, and dynamic sampling where each new measurement location is selected based on information obtained from previous measurements. Dynamic sampling methods are particularly appropriate for pointwise imaging methods, in which pixels are measured sequentially in arbitrary order. Examples of pointwise imaging schemes include certain implementations of atomic force microscopy, electron back scatter diffraction, and synchrotron X-ray imaging. In these pointwise imaging applications, dynamic sparse sampling methods have the potential to dramatically reduce the number of measurements required to achieve a desired level of fidelity. However, the existing dynamic sampling methods tend to be computationally expensive and are, therefore, too slow for many practical applications. In this paper, we present a framework for dynamic sampling based on machine learning techniques, which we call a supervised learning approach for dynamic sampling (SLADS). In each step of SLADS, the objective is to find the pixel that maximizes the expected reduction in distortion (ERD) given previous measurements. SLADS is fast because we use a simple regression function to compute the ERD, and it is accurate because the regression function is trained using datasets that are representative of the specific application. In addition, we introduce an approximate method to terminate dynamic sampling at a desired level of distortion. We then extend our algorithm to incorporate multiple measurements at each step, which we call groupwise SLADS. Finally, we present results on computationally generated synthetic data and experimentally collected data to demonstrate a dramatic improvement over state-of-the-art static sampling methods.**

***Index Terms*—Dynamic sampling, sparse sampling, electron microscopy, spectroscopy, smart sampling, adaptive sampling.**

Manuscript received March 13, 2017; revised August 16, 2017; accepted November 7, 2017. Date of publication November 24, 2017; date of current version February 8, 2018. This work was supported by the Air Force Office of Scientific Research (Multidisciplinary Research Program of the University Research Initiative—Managing the Mosaic of Microstructure) under Grant FA9550-12-1-0458 and by the Air Force Research Laboratory Materials and Manufacturing directorate under Contract FA8650-10-D-5201-0038. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Laura Waller. *(Corresponding author: G.M. Dilshan Godaliyadda)*

G. M. D. P. Godaliyadda, D. H. Ye, and C. A. Bouman are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: ggodaliy@purdue.edu; yed@purdue.edu; bouman@purdue.edu).

M. D. Uchic and M. A. Groeber are with the Air Force Research Laboratory, Materials and Manufacturing Directorate, Wright-Patterson AFB, OH 45433 USA (e-mail: michael.uchic@us.af.mil; michael.groeber@us.af.mil).

G. T. Buzzard is with the Department of Mathematics, Purdue University, West Lafayette, IN 47907 USA (e-mail: buzzard@purdue.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCI.2017.2777482

I. Introduction

Many important imaging methods are based on the sequential point-wise measurement of pixels in an image. Examples of such point-wise imaging methods include certain forms of atomic force microscopy (AFM) [1], electron back scatter diffraction (EBSD) microscopy [2], X-ray diffraction spectroscopy [3], and scanning Raman spectroscopy [4]. These scientific imaging methods are of great importance in material science, physics, and chemistry.

Sparse sampling offers the potential to dramatically reduce the time required to acquire an image. In sparse sampling, a subset of all available measurements are acquired, and the full resolution image is reconstructed from this set of sparse measurements. By reducing image acquisition time, sparse sampling also reduces the exposure of the object/person being imaged to potentially harmful radiation. This is critically important when imaging biological samples using X-rays, electrons, or even optical photons [5], [6]. Another advantage of sparse sampling is that it reduces the amount of measurement data that must be stored.

However, for a sparse sampling method to be useful, it is critical that the reconstruction made from the sparse set of samples allows for accurate reconstruction of the underlying object. Therefore, the selection of sampling locations is critically important. The methods that researchers have proposed for sparse sampling can broadly be sorted into two primary categories: static and dynamic.

Static sampling refers to any method that collects measurements in a predetermined order. Random sampling strategies such as in [7]–[9], low-discrepancy sampling [10], uniformly spaced sparse sampling methods [8], [11] and other predetermined sampling strategies such as Lissajous trajectories [12] are examples of static sparse sampling schemes. Static sampling methods can also be based on a model of the object being sampled such as in [13], [14]. In these methods knowledge of the object geometry and sparsity are used to predetermine the measurement locations.

Alternatively, dynamic sampling refers to any method that adaptively determines the next measurement location based on information obtained from previous measurements. Dynamic sampling has the potential to produce a high fidelity image with fewer measurements because of the information available from previous measurements. Intuitively, the previous measurements provide a great deal of information about the best location for future measurements.

Over the years, a wide variety of dynamic sampling methods have been proposed for many different applications. We categorize these dynamic sampling methods into three primary categories—dynamic compressive sensing methods where measurements are unconstrained projections, dynamic sampling methods developed for applications that are not point-wise imaging methods, and dynamic sampling methods developed for point-wise imaging methods.

In dynamic compressive sensing methods the objective at each step is to find the measurement that reduces the entropy the most. In these methods the entropy is computed using the previous measurements and a model for the underlying data. Examples of such methods include [15]–[18]. However, in these methods the measurements are projections along unconstrained measurement vectors, and therefore they cannot readily be generalized for point-wise imaging methods. Here, an unconstrained measurement vector is defined as a unit norm vector, where more than one element in the vector can be non-zero.

The next category of dynamic sampling methods in the literature are those developed for specific applications that are not point-wise imaging methods. For example in [19] the authors modify the optimal experimental design [20] framework to incorporate dynamic measurement selection in a biochemical network; and in [21], Seeger *et al.* select optimal K-space spiral and line measurements for magnetic resonance imaging (MRI). Also, in [22], Batenburg et al. present a method for binary computed tomography where each step of the measurement is designed to maximize the information gain.

There are also a few dynamic sampling methods developed specifically for point-wise imaging applications. One example is presented in [23] by Kovačević *et al.* for the application of fluorescence microscopy imaging. In this algorithm, an object is initially measured using a sparse uniformly spaced grid. Then, if the intensity of a pixel is above a certain threshold, the vicinity of that pixel is also measured. However, the threshold here is empirically determined and therefore many not be robust for general applications. In [24], Kovačević *et al.* propose a method for dynamically sampling a time-varying image by tracking features using a particle filter; and in [25], the authors introduce a method where initially different sets of pixels are measured to estimate the image, and further measurements are made where the estimated signal is non-zero. Another point-wise dynamic sampling method was proposed in [26]. In each step of this algorithm, the pixel that reduces the posterior variance the most is selected for measurement. The posterior variance is computed using samples generated from the posterior distribution using the Metropolis-Hastings algorithm [27], [28]. However, Monte-Carlo methods such as the Metropolis-Hastings algorithm can be very slow for cases where the dimensions of the random vector are large [19], [26]. Another shortcoming of this method is that it does not account for the change of conditional variance in the full image due to a new measurement.

In this paper, we present a dynamic sampling algorithm for point-wise imaging methods based on supervised learning techniques that we first presented in [29]. We call this algorithm a supervised learning approach for dynamic sampling (SLADS). In each step of the SLADS algorithm, we select the pixel that greedily maximizes the expected reduction in distortion (ERD) given previous measurements. Importantly, the ERD is computed using a simple regression function applied to features from previous measurements. As a result, we can compute the ERD very rapidly during dynamic sampling.

For the SLADS algorithm to be accurate, the regression function must be trained off-line using reduction in distortion (RD) values from many different typical measurements. However, creating a large training data set with many entries can be computationally challenging because evaluation of each RD entry requires two reconstructions, one before the measurement and one after. In order to solve this problem, we introduce an efficient training scheme and an approximation to the RD, that allows us to extract multiple entries for the training database with just one reconstruction. Then, for each RD entry we extract a corresponding feature vector that captures the uncertainty associated with the unmeasured location, to ensure SLADS is accurate. We then empirically validate the approximation to the RD for small images and describe a method for estimating the required estimation parameter. We also introduce an approximate stopping condition for dynamic sampling, which allows us to stop when a desired distortion level is reached. Finally, we extend our algorithm to incorporate group-wise sampling so that multiple measurements can be selected in each step of the algorithm.

In the results section of this paper, we first empirically validate our approximation to the RD by performing experiments on 64×64 sized computationally generated EBSD images. Then we compare SLADS with state-of-the-art static sampling methods by sampling both simulated EBSD and real SEM images. We observe that with SLADS we can compute a new sample location very quickly (in the range of 1–100 ms), and can achieve the same reconstruction distortion as static sampling methods with dramatically fewer samples. Finally, we evaluate the performance of group-wise SLADS by comparing it to SLADS and to static sampling methods.

II. Dynamic Sampling Framework

Denote the unknown image formed by imaging the entire underlying object as $X \in \mathbb{R}^N$. Then the value of the pixel at location $r \in \Omega$ is denoted by X_r, where Ω is the set of all locations in the image.

In the dynamic sampling framework, we assume that k pixels have already been measured at a set of locations $\mathcal{S} = \{s^{(1)}, \ldots, s^{(k)}\}$. We then represent the measurements and the corresponding locations as a $k \times 2$ matrix

$$Y^{(k)} = \begin{bmatrix} s^{(1)}, X_{s^{(1)}} \\ \vdots \\ s^{(k)}, X_{s^{(k)}} \end{bmatrix}.$$

From these measurements, $Y^{(k)}$, we can compute an estimate of the unknown image X. We denote this best current estimate of the image as $\hat{X}^{(k)}$.

Now we would like to determine the next pixel location $s^{(k+1)}$ to measure. If we select a new pixel location s and measure its value X_s, then we can presumably reconstruct a better estimate of X. We denote this improved estimate as $\hat{X}^{(k;s)}$.

Of course, our goal is to minimize the distortion between X and $\hat{X}^{(k;s)}$, which we denote by the following function

$$D(X, \hat{X}^{(k;s)}) = \sum_{r \in \Omega} D(X_r, \hat{X}_r^{(k;s)}) \, , \tag{1}$$

where $D(X_r, \hat{X}_r^{(k;s)})$ is some scalar measure of distortion between the two pixels X_r and $\hat{X}_r^{(k;s)}$. Here, the function $D(\cdot, \cdot)$ may depend on the specific application or type of image. For example we can let $D(a,b) = |a-b|^l$ where $l \in \mathbb{Z}^+$.

In fact, greedily minimizing this distortion is equivalent to maximizing the reduction in the distortion that occurs when we make a measurement. To do this, we define $R_r^{(k;s)}$ as the reduction-in-distortion at pixel location r resulting from a measurement at location s.

$$R_r^{(k;s)} = D(X_r, \hat{X}_r^{(k)}) - D(X_r, \hat{X}_r^{(k;s)}) \tag{2}$$

It is important to note that a measurement will reduce the distortion in the entire image. Therefore, to represent the total reduction in distortion, we must sum over all pixels $r \in \Omega$.

$$\begin{aligned} R^{(k;s)} &= \sum_{r \in \Omega} R_r^{(k;s)} && (3) \\ &= D(X, \hat{X}^{(k)}) - D(X, \hat{X}^{(k;s)}). && (4) \end{aligned}$$

Notice that $R^{(k;s)}$ will typically be a positive quantity since we expect that the distortion should reduce when we collect additional information with new measurements. However, in certain situations $R^{(k;s)}$ can actually be negative since a particular measurement might inadvertently cause the reconstruction to be less accurate, thereby increasing the distortion.

Importantly, we cannot know the value of $R^{(k;s)}$ during the sampling process because we do not know X. Therefore, our real goal will be to minimize the expected value of $R^{(k;s)}$ given our current measurements. We define the expected reduction-in-distortion (ERD) as

$$\bar{R}^{(k;s)} = \mathbb{E}\left[R^{(k;s)} | Y^{(k)} \right] . \tag{5}$$

Since the ERD is the conditional expectation of $R^{(k;s)}$ given the measurements $Y^{(k)}$, it does not require knowledge of X.

The specific goal of our greedy dynamic sampling algorithm is then to select the pixel s that greedily maximizes the ERD.

$$s^{(k+1)} = \arg\max_{s \in \Omega} \left\{ \bar{R}^{(k;s)} \right\} \tag{6}$$

Intuitively, (6) selects the next pixel to maximize the expected reduction-in-distortion given all the available information $Y^{(k)}$.

Once $s^{(k+1)}$ is determined, we then form a new measurement matrix given by

$$Y^{(k+1)} = \begin{bmatrix} Y^{(k)} \\ s^{(k+1)}, X_{s^{(k+1)}} \end{bmatrix} . \tag{7}$$

We repeat this process recursively until the stopping condition discussed in Section IV is achieved. This stopping condition can be used to set a specific expected quality level for the reconstruction.

```
$k \leftarrow 0$
**repeat**
    $s^{(k+1)} = \arg\max_{s \in \Omega} \left\{ \bar{R}^{(k;s)} \right\}$
    $Y^{(k+1)} = \begin{bmatrix} Y^{(k)} \\ s^{(k+1)}, X_{s^{(k+1)}} \end{bmatrix}$
    $k \leftarrow k+1$
**until** Desired fidelity is achieved
```

Fig. 1. Summary of the greedy dynamic sampling algorithm in pseudocode.

In summary, the greedy dynamic sampling algorithm is given by the iteration shown in Fig. 1.

III. Supervised Learning Approach for Dynamic Sampling (SLADS)

The challenge in implementing this greedy dynamic sampling method is accurately determining the ERD function, $\bar{R}^{(k;s)}$. A key innovation of our proposed SLADS algorithm is to determine this ERD function by using supervised learning techniques with training data.

More specifically, SLADS will use an off-line training approach to learn the relationship between the ERD and the available measurements, $Y^{(k)}$, so that we can efficiently predict the ERD. More specifically, we would like to fit a regression function $f_s^{\theta}(\cdot)$, so that

$$\bar{R}^{(k;s)} = f_s^{\theta}(Y^{(k)}). \tag{8}$$

Here $f_s^{\theta}(\cdot)$ denotes a non-linear regression function determined through supervised learning, and θ is the parameter vector that must be estimated in the learning process.

For the remainder of this section, we have dropped the superscript k in our explanation of the training procedure. We do this because in training we must use different values of k (i.e., different sampling percentages) in order to ensure our estimate for $f_s^{\theta}(\cdot)$ is accurate regardless of the number of measurements k.

Now, to estimate the function $f_s^{\theta}(\cdot)$, we construct a training database containing multiple corresponding pairs of $(R^{(s)}, Y)$. Here, $R^{(s)} = D(X, \hat{X}) - D(X, \hat{X}^{(s)})$ is the RD due to s, where $\hat{X}$ is the best estimate of X computed using the measurements Y, and $\hat{X}^{(s)}$ is the best estimate of X computed using the measurements Y and an additional measurement at location s.

Notice that since $R^{(s)}$ is the reduction-in-distortion, it requires knowledge of the true image X. Since this is an off-line training procedure, X is available, and the regression function, $f_s^{\theta}(Y)$, will compute the required expected reduction in distortion denoted by $\bar{R}^{(s)}$.

In order to accurately estimate the ERD function, we will likely need many corresponding pairs of the reduction in distortion $R^{(s)}$ and the measurements Y. However, to compute $R^{(s)}$ for a single value of s, we must compute two full reconstructions, both $\hat{X}$ and $\hat{X}^{(s)}$. Since reconstruction can be computationally expensive, this means that creating a large database may be very computationally expensive. We will address this problem and propose a solution to it in Section III-A.

For our implementation of SLADS, the regression function $f_s^\theta(Y)$ will be a function of a row vector containing features extracted from Y. More specifically, at each location s, a p-dimensional feature vector V_s will be extracted from Y and used as input to the regression function. Hence, $V_s = g_s(Y)$, where $g_s(\cdot)$ is a non-linear function that maps the measurements Y to a p-dimensional vector V_s. The specific choices of features used in V_s are listed in Section III-D; however, other choices are possible.

From this feature vector, we then compute the ERD using a linear predictor with the form

$$\begin{aligned}\bar{R}^{(s)} &= f_s^\theta(Y) \\ &= g_s(Y)\,\theta \\ &= V_s\,\theta\,, \end{aligned} \tag{9}$$

where $V_s = g_s(Y)$ is a local feature vector extracted from Y. We can estimate the parameter θ by solving the following least-squares regression

$$\hat{\theta} = \arg\min_{\theta\in\mathbb{R}^p} \|\mathbf{R} - \mathbf{V}\theta\|^2\,, \tag{10}$$

where $\mathbf{R}$ is an n-dimensional column vector formed by

$$\mathbf{R} = \begin{bmatrix} R^{(s_1)} \\ \vdots \\ R^{(s_n)} \end{bmatrix}, \tag{11}$$

and $\mathbf{V}$ is given by

$$\mathbf{V} = \begin{bmatrix} V_{s_1} \\ \vdots \\ V_{s_n} \end{bmatrix}. \tag{12}$$

So together $(\mathbf{R}, \mathbf{V})$ consist of n training pairs, $\{(R_{s_i}, V_{s_i})\}_{i=1}^n$, that are extracted from training data during an off-line training procedure. The parameter θ is then given by

$$\hat{\theta} = \left(\mathbf{V}^t\mathbf{V}\right)^{-1}\mathbf{V}^t\mathbf{R}. \tag{13}$$

Once $\hat{\theta}$ is estimated, we can use it to find the ERD for each unmeasured pixel during dynamic sampling. Hence, we find the $k+1$th location to measure by solving

$$s^{(k+1)} = \arg\max_{s\in\Omega}\left(V_s^{(k)}\hat{\theta}\right)\,, \tag{14}$$

where $V_s^{(k)}$ denotes the feature vector extracted from the measurements $Y^{(k)}$ at location s. It is important to note that this computation can be done very fast. The pseudo code for SLADS is shown in Fig. 2.

```
function $Y^{(K)} \leftarrow \text{SLADS}(Y^{(k)}, \hat{\theta}, k)$
    $\mathcal{S} \leftarrow \{s^{(1)}, s^{(2)}, \ldots s^{(k)}\}$
    while Stopping condition not met do
        for $\forall s \in \{\Omega \setminus \mathcal{S}\}$ do
            Extract $V_s^{(k)}$
            $\bar{R}^{(k;s)} \leftarrow V_s^{(k)}\hat{\theta}$
        end for
        $s^{(k+1)} = \arg\max_{s\in\{\Omega\setminus\mathcal{S}\}} \left(\bar{R}^{(k;s)}\right)$
        $Y^{(k+1)} = \begin{bmatrix} Y^{(k)} \\ s^{(k+1)}, X_{s^{(k+1)}} \end{bmatrix}$
        $\mathcal{S} \leftarrow \{\mathcal{S} \cup s^{(k+1)}\}$
        $k \leftarrow k+1$
    end while
    $K \leftarrow k$
end function
```

Fig. 2. SLADS algorithm in pseudocode. The inputs to the function are the initial measurements $Y^{(k)}$, the coefficients needed to compute the ERD, found in training, $\hat{\theta}$, and k, the number of measurements. When the stopping condition is met, the function will output the selected set of measurements $Y^{(K)}$.

A. Training for SLADS

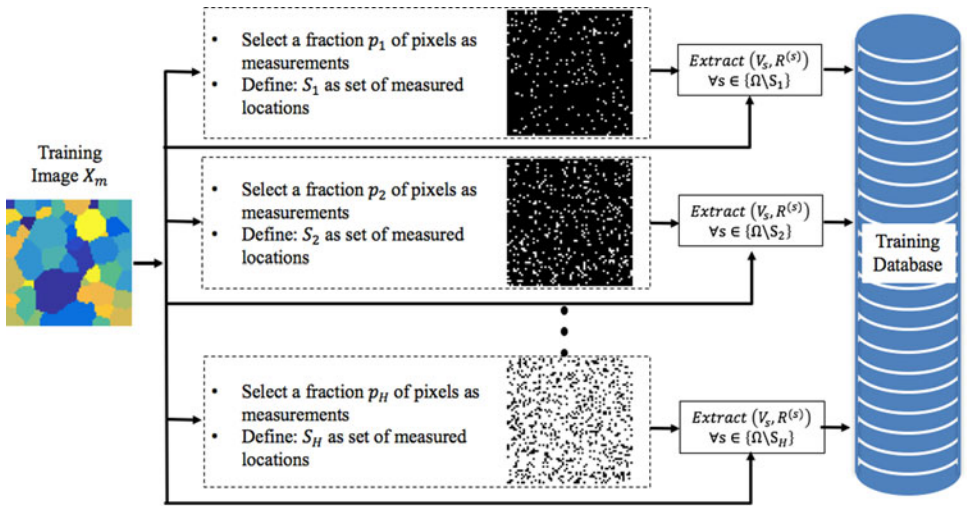
In order to train the SLADS algorithm, we must form a large training database containing corresponding pairs of $R^{(s)}$ and V_s. To do this, we start by selecting M training images denoted by $\{X_1, X_2, \ldots, X_M\}$. We also select a set of sampling densities represented by $p_1, p_2, \ldots, p_H$, where $0 \leq p_k \leq 1$ and $p_i < p_j$ for $i < j$.

For image X_m and each sampling density, p_h, we randomly select a fraction p_h of all pixels in the image to represent the simulated measurement locations. Then for each of the remaining unmeasured locations, s, in the image X_m, we compute the pair $(R^{(s)}, V_s)$ and save this pair to the training database. This process is then repeated for all the sampling densities and all the images to form a complete training database.

Fig. 3 illustrates this procedure. Note that by selecting a set of increasing sampling densities, $p_1, p_2, \ldots, p_H$, the SLADS algorithm can be trained to accurately predict the ERD for a given location regardless of whether the local sampling density is high or low. Intuitively, by sampling a variety of images with a variety of sampling densities, the final training database is constructed to represent all the behaviors that will occur as the sampling density increases when SLADS is used.

B. Approximating the Reduction-in-Distortion

While the training procedure described in Section III-A is possible, it is very computationally expensive because of the need to exactly compute the value of $R^{(s)}$. Since this computation is done in the training phase, we rewrite (4) without



Training Image X_m
Select a fraction p_1 of pixels as measurements
Define: S_1 as set of measured locations
Extract $(V_s, R^{(s)})$ $\forall s \in \{\Omega \setminus S_1\}$
Select a fraction p_2 of pixels as measurements
Define: S_2 as set of measured locations
Extract $(V_s, R^{(s)})$ $\forall s \in \{\Omega \setminus S_2\}$
Select a fraction p_H of pixels as measurements
Define: S_H as set of measured locations
Extract $(V_s, R^{(s)})$ $\forall s \in \{\Omega \setminus S_H\}$
Training Database

Fig. 3. Illustration of how training data are extracted from one image in the training database. We first select a fraction p_1 of the pixels in the image and consider them as measurements Y. Then, for all unmeasured pixel locations ($s \in \{\Omega \setminus \mathcal{S}_1\}$), we extract a feature vector V_s and also compute $R^{(s)}$. We then repeat the process for when fractions of $p_2, p_3, \ldots$ and p_H of all pixels are considered measurements. Here, again Ω is the set of all locations in the training image and $\mathcal{S}_i$ is the set of measured locations when a fraction p_i of pixels are selected as measurements. All these pairs of $\left(V_s, R^{(s)}\right)$ are then stored in the training database.

the dependence on k to be

$$R^{(s)} = D(X, \hat{X}) - D(X, \hat{X}^{(s)}) \,, \tag{15}$$

where X is known in training, $\hat{X}$ is the reconstruction using the selected sample points, and $\hat{X}^{(s)}$ is the reconstruction using the selected sample points along with the value X_s. Notice that $\hat{X}^{(s)}$ must be recomputed for each new pixel s. This requires that a full reconstruction be computed for each entry of the training database.

In other words, a single entry in the training database comprises the features extracted from the measurements, $V_s = g_s(Y)$, along with the reduction in distortion due to an extra measurement at a location s, denoted by $R^{(s)}$. While it is possible to compute this required pair, $(V_s, R^{(s)})$, this typically requires a very large amount of computation.

In order to reduce this computational burden, we introduce a method for approximating the value of $R^{(s)}$ so that only a single reconstruction needs to be performed in order to evaluate $R^{(s)}$ for all pixels s in an image. This dramatically reduces the computation required to build the training database.

In order to express our approximation, we first rewrite the reduction-in-distortion in the form

$$R^{(s)} = \sum_{r \in \Omega} R_r^{(s)},$$

where

$$R_r^{(s)} = D(X_r, \hat{X}_r) - D(X_r, \hat{X}_r^{(s)}).$$

So here $R_r^{(s)}$ is the reduction-in-distortion at pixel r due to making a measurement at pixel s. Using this notation, our approximation is given by

$$R_r^{(s)} \approx \tilde{R}_r^{(s)} = h_{s,r} D\left(X_r, \hat{X}_r\right) \,, \tag{16}$$

where

$$h_{s,r} = \exp\left\{-\frac{1}{2\sigma_s^2}\|r - s\|^2\right\} \tag{17}$$

and σ_s is the distance between the pixel s and the nearest previously measured pixel divided by a user selectable parameter c. More formally, σ_s is given by

$$\sigma_s = \frac{\min_{t \in \mathcal{S}} \|s - t\|}{c}, \tag{18}$$

where $\mathcal{S}$ is the set of measured locations. So this results in the final form for the approximate reduction-in-distortion given by

$$\tilde{R}^{(s)} = \sum_{r \in \Omega} h_{s,r} D\left(X_r, \hat{X}_r\right) \,, \tag{19}$$

where c is a parameter that will be estimated for the specific problem.

In order to understand the key approximation of (16), notice that the reduction-in-distortion is proportional to the product of $h_{s,r}$ and $D\left(X_r, \hat{X}_r\right)$. Intuitively, $h_{s,r}$ represents the weighted distance of r from the location of the new measurement, s; and $D\left(X_r, \hat{X}_r\right)$ is the initial distortion at r before the measurement was made. So for example, when $r = s$ (i.e., when we measure the pixel s), then the RD at s due to measurement s is given by

$$\tilde{R}_s^{(s)} = D\left(X_s, \hat{X}_s\right). \tag{20}$$

Notice that in this case $h_{s,s} = 1$, and the reduction-in-distortion is exactly the initial distortion since the measurement is assumed to be exact with $D\left(X_s, \hat{X}_s\right) = 0$. However, as r becomes more distant from the pixel being measured, s, the reduction-in-distortion will be attenuated by the weight $h_{s,r} < 1$. We weight the impact of a measurement by the inverse Euclidean distance because widely used reconstruction methods [30], [31] weight

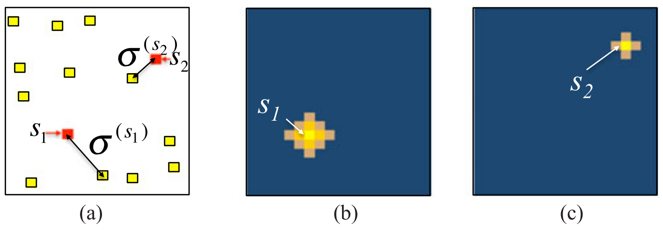


Fig. 4. Images illustrating the shape of the function $h_{s,r}$ as a function of r. (a) Measurement locations, where red squares represent the two new measurement locations s_1 and s_2, and the yellow squares represent the locations of previous measurements. (b) The function $h_{s_1,r}$ resulting from a measurement at location s_1. (c) The function $h_{s_2,r}$ resulting from a measurement at location s_2. Notice that since $\sigma_{s_1} > \sigma_{s_2}$ then the weighting function $h_{s_1,r}$ is wider than the function $h_{s_2,r}$.

the impact of a measurement at a location s on a location r by the inverse Euclidean distance.

Fig. 4(b) and (c) illustrate the shape of $h_{s,r}$ for two different cases. In Fig. 4(b), the pixel s_1 is further from the nearest measured pixel, and in Fig. 4(c), the pixel s_2 is nearer. Notice that as r becomes more distant from the measurement location s, the reduction-in-distortion becomes smaller; however, that rate of attenuation depends on the local sampling density.

C. Estimating the c Parameter

In this section, we present a method for estimating the parameter c used in (18). To do this, we create a training database that contains the approximate reduction-in-distortion for a set of parameter values. More specifically, each entry of the training database has the form

$$\left(\tilde{R}^{(s;c_1)}, \tilde{R}^{(s;c_2)}, \ldots, \tilde{R}^{(s;c_U)}, V_s\right),$$

where $c \in \{c_1, c_2, \ldots c_U\}$ is a set of U possible parameter values, and $\tilde{R}^{(s;c_i)}$ is the approximate reduction-in-distortion computed using the parameter value c_i.

Using this training database, we compute the U associated parameter vectors $\hat{\theta}^{(c_i)}$, and using these parameter vectors, we apply the SLADS algorithm on M images and stop each simulation when K samples are taken. Then for each of these M SLADS simulations, we compute the total distortion as

$$TD_k^{(m,c_i)} = \frac{1}{|\Omega|} D\left(X^{(m)}, \hat{X}^{(k,m,c_i)}\right), \tag{21}$$

where $X^{(m)}$ is the mth actual image, and $\hat{X}^{(m,k,c_i)}$ is the associated image reconstructed using the first k samples and parameter value c_i. Next we compute the average total distortion over the M training images given by

$$\overline{TD}_k^{(c_i)} = \frac{1}{M} \sum_{m=1}^{M} TD_k^{(m,c_i)}. \tag{22}$$

From this, we then compute the area under the $\overline{TD}$ curve as the overall distortion metric for each c_i given by

$$DM^{(c_i)} = \sum_{k=2}^{K} \frac{\overline{TD}_{k-1}^{(c_i)} + \overline{TD}_k^{(c_i)}}{2}, \tag{23}$$

TABLE I
LIST OF DESCRIPTORS USED TO CONSTRUCT THE FEATURE VECTOR V_s

Measures of gradients	
Gradient of the reconstruction in horizontal (x) direction. $Z_{s,1} = D\left(\hat{X}_{s_{x+}}, \hat{X}_{s_{x-}}\right)$ where, s_{x+} and s_{x-} are pixels adjacent to s the horizontal direction	Gradient of the reconstruction in vertical (y) direction. $Z_{s,2} = D\left(\hat{X}_{s_{y+}}, \hat{X}_{s_{y-}}\right)$ where, s_{y+} and s_{y-} are pixels adjacent to s the vertical direction
Measures of standard deviation	
$Z_{s,3} = \sqrt{\frac{1}{L}\sum_{r\in\partial s} D\left(X_r, \hat{X}_s\right)^2}$ Here ∂s is the set containing the indices of the L nearest measurements to s.	$Z_{s,4} = \sum_{r\in\partial s} w_r^{(s)} D\left(X_r, \hat{X}_s\right)$ Here, $w_r^{(s)} = \frac{\frac{1}{\|s-r\|^2}}{\sum_{u\in\partial s}\frac{1}{\|s-u\|^2}}$ and $\|s-r\|$ is the euclidean distance between s and r.
Measures of density of measurements	
$Z_{s,5} = \min_{r\in\partial s}\|s-r\|_2$ The distance from s to the closest measurement.	$Z_{s,6} = \frac{1 + A_{(s;\lambda)}}{1 + A^*_{(s;\lambda)}}$ Here $A_{(s;\lambda)}$ is the area of a circle λ% the size of the image. $A^*_{(s;\lambda)}$ is the measured area inside $A_{(s;\lambda)}$.

There are three main categories of descriptors: measures of gradients, measures of standard deviation, and measures of density of measurements surrounding the pixel s.

where K is the total number of samples taken before stopping. We use the area under the $\overline{TD}$ curve as the deciding factor because it quantifies how fast the error is reduced with each c_i, and therefore can be used to compare the performance under each c_i. The optimal parameter value, c^*, is then selected to minimize the overall distortion metric given by

$$c^* = \arg \min_{c\in\{c_1,c_2,\ldots c_U\}} \left\{DM^{(c)}\right\}. \tag{24}$$

D. Local Descriptors in Feature Vector V_s

In our implementation, the feature vector V_s is formed using terms constructed from the 6 scalar descriptors $Z_{s,1}, Z_{s,2}, \ldots Z_{s,6}$ listed in Table I. More specifically, we take all the unique second-degree combinations formed from these descriptors, and the descriptors themselves to form V_s. The reason we use these first and second-degree combinations in V_s is because we do not know the relationship between the ERD and the descriptors, and therefore our approach is to account for some possible nonlinear relationships between V_s and the ERD in the trained regression function. More generally, one might use other machine learning techniques such as support vector machines [32] or convolutional neural networks [33], [34] to learn this possibly nonlinear relationship.

This gives us a total of 28 elements for the vector V_s.

$$V_s = [1, Z_{s,1}, \dots, Z_{s,6}, Z_{s,1}^2, Z_{s,1}Z_{s,2}, \dots, Z_{s,1}Z_{s,6}, Z_{s,2}^2, Z_{s,2}Z_{s,3}, \dots, Z_{s,2}Z_{s,6}, \dots, Z_{s,6}^2].$$

Intuitively, $Z_{s,1}$ and $Z_{s,2}$ in Table I are the gradients in the horizontal and vertical direction at an unmeasured location s computed from the image reconstructed using previous measurements. $Z_{s,3}$ and $Z_{s,4}$, are measures of the variance for the same unmeasured location s. The value of $Z_{s,3}$ is computed using only the intensities of the neighboring measurements, while $Z_{s,4}$ also incorporates the Euclidean distance to the neighboring measurements. Hence, $Z_{s,1}, Z_{s,2}, Z_{s,3}$ and $Z_{s,4}$ all contain information related to the intensity variation at each unmeasured location. Then the last two descriptors, $Z_{s,5}$ and $Z_{s,6}$, quantify how densely (or sparsely) the region surrounding an unmeasured pixel is measured. In particular, the $Z_{s,5}$ is the distance from the nearest measurement to an unmeasured pixel, and $Z_{s,6}$ is the area fraction that is measured in a circle surrounding an unmeasured pixel.

These specific 6 descriptors were chosen in an ad-hoc manner because we found that they were simple and effective for our application. However, we expect that other more sophisticated descriptors and more sophisticated machine learning techniques could be used in this application or others that could potentially result in a better estimate of the ERD or simply better results in a dynamic sampling application. Moreover, we believe that investigation of better machine learning techniques for SLADS could prove to be a fertile area of future research [34], [35].

IV. Stopping Condition for SLADS

In applications, it is often important to have a stopping criteria for terminating the SLADS sampling process. In order to define such a criteria, we first define the expected total distortion (ETD) at step k by

$$ETD_k = \mathbb{E}\left[\frac{1}{|\Omega|}D\left(X, \hat{X}^{(k)}\right)|Y^{(k)}\right].$$

Notice that the expectation is necessary since the true value of X is unavailable during the sampling process. Then our goal is to stop sampling when the ETD falls below a predetermined threshold, T.

$$ETD_k \leq T \tag{25}$$

It is important to note that this threshold is dependent on the type of image we sample and the specific distortion metric chosen.

Since we cannot compute ETD_k, we instead will compute the function

$$\epsilon^{(k)} = (1-\beta)\epsilon^{(k-1)} + \beta D\left(X_{s^{(k)}}, \hat{X}_{s^{(k)}}^{(k-1)}\right), \tag{26}$$

that we will use in place of the ETD_k for $k > 1$. Here, β is a user selected parameter that determines the amount of temporal smoothing, $X_{s^{(k)}}$ is the measured value of the pixel at step k, $\hat{X}_{s^{(k)}}^{(k-1)}$ is the reconstructed value of the same pixel at step $k-1$ and $\epsilon^{(0)} = 0$.

Intuitively, the value of $\epsilon^{(k)}$ measures the average level of distortion in the measurements. So a large value of $\epsilon^{(k)}$ indicates that more samples need to be taken, and a smaller value indicates that the reconstruction is accurate and the sampling process can be terminated. However, in typical situations, it will be the case that

$$\epsilon^{(k)} > ETD_k$$

because the SLADS algorithm tends to select measurements that are highly uncertain and as a result $D\left(X_{s^{(k)}}, \hat{X}_{s^{(k)}}^{(k-1)}\right)$ in (26) tends to be large.

Therefore, we cannot directly threshold $\epsilon^{(k)}$ with T in (25). Hence, we instead compute a function $\tilde{T}(T)$ using a look-up-table (LUT) and stop sampling when

$$\epsilon^{(k)} \leq \tilde{T}(T).$$

It is important to note here that $\tilde{T}(T)$ is a function of the original threshold T at which we intend to stop SLADS.

The function $\tilde{T}(T)$ is determined using a set of training images, $\{X_1, \cdots, X_M\}$. For each image, we first determine the number of steps, $K_m(T)$, required to achieve the desired distortion.

$$K_m(T) = \min_k \left\{k : \frac{1}{|\Omega|}D\left(X_m, \hat{X}_m^{(k)}\right) \leq T\right\} \tag{27}$$

where, $m \in \{1, 2, \dots M\}$. Then we average the value of $\epsilon_m^{(K_m(T))}$ for each of the M images to determine the adjusted threshold:

$$\tilde{T}(T) = \frac{1}{M}\sum_{m=1}^{M}\epsilon_m^{(K_m(T))}. \tag{28}$$

In practice, we have used the following formula to set β:

$$\beta = \begin{cases} 0.001\left(\frac{\log_2(512^2)-\log_2(|\Omega|)}{2}+1\right) & |\Omega| \leq 512^2 \\ 0.001\left(\frac{\log_2(|\Omega|)-\log_2(512^2)}{2}+1\right)^{-1} & |\Omega| > 512^2 \end{cases}$$

where $|\Omega|$ is the number of pixels in the image. We chose the value of β for a given resolution so that the resulting $\epsilon^{(k)}$ curve was smooth. When the $\epsilon^{(k)}$ curve is smooth the variance of $\epsilon_m^{(K_m(T))}$ in (28) is low, and as a result we get an accurate stopping threshold.

Fig. 5 shows the SLADS algorithm as a flow diagram after the stopping condition in the previous section is incorporated.

V. Groupwise SLADS

In this section, we introduce a group-wise SLADS approach in which B measurements are made at each step of the algorithm. Group-wise SLADS is more appropriate in applications where it is faster to measure a set of predetermined pixels in a single burst when compared to acquiring the same set one at a time. For example, in certain imaging applications moving the probe can be expensive. In such cases by finding b samples all at once,

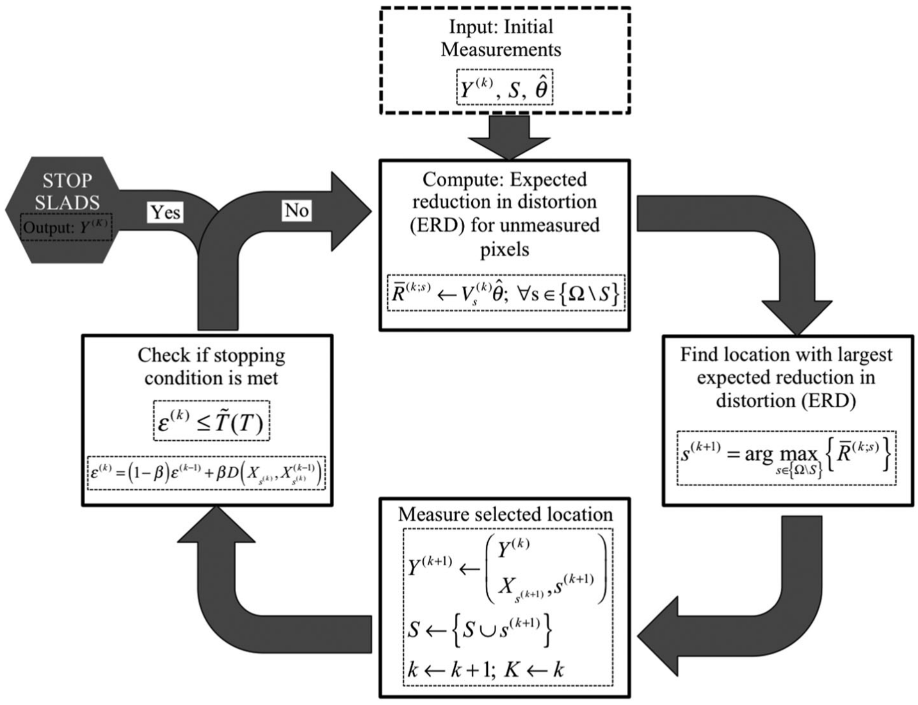

Input: Initial Measurements
$Y^{(k)}, S, \hat{\theta}$
Compute: Expected reduction in distortion (ERD) for unmeasured pixels
$\bar{R}^{(k;s)} \leftarrow V_s^{(k)}\hat{\theta};\ \forall s \in \{\Omega \setminus S\}$
Find location with largest expected reduction in distortion (ERD)
$s^{(k+1)} = \arg\max_{s\in\{\Omega\setminus S\}} \left\{\bar{R}^{(k;s)}\right\}$
Measure selected location
$Y^{(k+1)} \leftarrow \begin{pmatrix} Y^{(k)} \\ X_{s^{(k+1)}}, s^{(k+1)} \end{pmatrix}$
$S \leftarrow \left\{S \cup s^{(k+1)}\right\}$
$k \leftarrow k+1;\ K \leftarrow k$
Check if stopping condition is met
$\varepsilon^{(k)} \leq \tilde{T}(T)$
$\varepsilon^{(k)} = (1-\beta)\varepsilon^{(k-1)} + \beta D\left(X_{s^{(k)}}, X_{s^{(k)}}^{(k-1)}\right)$
Yes
No
STOP SLADS
Output: $Y^{(K)}$

Fig. 5. Flow diagram of SLADS algorithm. The inputs to the algorithm are the initial measurements $Y^{(k)}$, the coefficients needed to compute the ERD, found in training, $\hat{\theta}$, and the set $\mathcal{S}$ containing the indices of the measurements. When the stopping condition is met the function will output the selected set of measurements $Y^{(K)}$.

we can find a shortest path between said locations to reduce the movement of the probe.

So at the kth step, our goal will be to select a group of measurement positions,

$$\mathcal{S}^{(k+1)} = \left\{s_1^{(k+1)}, s_2^{(k+1)}, \dots s_B^{(k+1)}\right\},$$

that will yield the greatest expected reduction-in-distortion.

$$\mathcal{S}^{(k+1)} = \arg \max_{\{s_1, s_2, \dots s_B\} \in \{\Omega \setminus \mathcal{S}\}} \left\{\bar{R}^{(k; s_1, s_2, \dots s_B)}\right\}, \qquad (29)$$

where $\bar{R}^{(k;s_1,s_2,\dots s_B)}$ is the expected reduction-in-distortion due to measurements $s_1^{(k+1)}, s_2^{(k+1)}, \dots s_B^{(k+1)}$. However, solving this problem requires that we consider $\binom{N-|\mathcal{S}|}{B}$ different combinations of measurements.

In order to address this problem, we introduce a method in which we choose the measurements sequentially, just as we do in standard SLADS. Since group-wise SLADS requires that we make measurements in groups, we cannot make the associated measurement after each location is selected. Consequently, we cannot recompute the ERD function after each location is selected, and therefore, we cannot select the best position for the next measurement. Our solution to this problem is to estimate the value at each selected location, and then we use the estimated value as if it were the true measured value.

More specifically, we first determine measurement location $s_1^{(k+1)}$ using (6), and then let $\mathcal{S} \leftarrow \left\{\mathcal{S} \cup s_1^{(k+1)}\right\}$. Now without measuring the pixel at $s_1^{(k+1)}$, we would like to find the location of the next pixel to measure, $s_2^{(k+1)}$. However, since $s_1^{(k+1)}$ has now been chosen, it is important to incorporate this information when choosing the next location $s_2^{(k+1)}$. In our implementation, we temporarily assume that the true value of the pixel, $X_{s_1^{(k+1)}}$, is given by its estimated value, $\hat{X}^{(k)}_{s_1^{(k+1)}}$, computed using all the measurements acquired up until the kth step, which is why we use the superscript k on $\hat{X}^{(k)}_{s_1^{(k+1)}}$. We will refer to $\hat{X}^{(k)}_{s_1^{(k+1)}}$ as a *pseudo*-measurement since it takes the place of a true measurement of the pixel. Now using this *pseudo*-measurement along with all previous real measurements, we estimate a *pseudo*-ERD $\bar{R}^{(k, s_1^{(k+1)}; s)}$ for all $s \in \{\Omega \setminus \mathcal{S}\}$ and from that select the next location to measure. We repeat this procedure to find all B measurements.

So the procedure to find the bth measurement is as follows. We first construct a *pseudo*-measurement vector,

$$Y_b^{(k+1)} = \begin{bmatrix} Y^{(k)} \\ s_1^{(k+1)}, \hat{X}^{(k)}_{s_1^{(k+1)}} \\ s_2^{(k+1)}, \hat{X}^{(k)}_{s_2^{(k+1)}} \\ \cdots \\ s_{b-1}^{(k+1)}, \hat{X}^{(k)}_{s_{b-1}^{(k+1)}} \end{bmatrix}, \qquad (30)$$

where $Y_1^{(k+1)} = Y^{(k)}$. It is important to note that we include the original k measurements $Y^{(k)}$ along with the $b-1$ *pseudo*-measurements in the vector $Y_b^{(k+1)}$. We do this because the

```
function $Y^{(K)} \leftarrow$ GROUP-WISE SLADS($Y^{(k)}, \hat{\theta}, k, B$)
    $\mathcal{S} \leftarrow \{s^{(1)}, s^{(1)}, \dots s^{(k)}\}$
    while Stopping condition not met do
        for $b = 1, \dots B$ do
            Form pseudo-measurement vector $Y_b^{(k+1)}$ as shown in Eq. (30)
            Compute pseudo-ERD $\bar{R}^{(k, s_1^{(k+1)}, s_2^{(k+1)} \dots s_{b-1}^{(k+1)}; s)}$ from $Y_b^{(k+1)}$ $\forall s \in \mathbb{S}$ using Eq. (31)
            $s_b^{(k+1)} = \arg\max_{s \in \{\Omega \setminus \mathcal{S}\}} \left( \bar{R}^{(k, s_1^{(k+1)}, s_2^{(k+1)} \dots s_{b-1}^{(k+1)}; s)} \right)$
            $\mathcal{S} \leftarrow \left\{ \mathcal{S} \cup s_b^{(k+1)} \right\}$
        end for
        $Y^{(k+1)} = \begin{bmatrix} Y^{(k)} \\ s_1^{(k+1)}, \hat{X}^{(k)}_{s_1^{(k+1)}} \\ s_2^{(k+1)}, \hat{X}^{(k)}_{s_2^{(k+1)}} \\ \vdots \\ s_b^{(k+1)}, \hat{X}^{(k)}_{s_b^{(k+1)}} \end{bmatrix}$
        $k \leftarrow k + 1$
    end while
    $K \leftarrow k$
end function
```

Fig. 6. Pseudocode for groupwise SLADS. Here, instead of just selecting one measurement in each step of SLADS, the groupwise SLADS algorithm selects B new measurement locations in each step.

pseudo-measurements, while not equal to the desired true values, still result in much better predictions of the ERD for future measurements.

Then using this *pseudo*-measurement vector, we compute the *pseudo*-ERD for all $s \in \{\Omega \setminus \mathcal{S}\}$

$$\bar{R}^{\left(k, s_1^{(k+1)}, s_2^{(k+1)}, \dots s_{b-1}^{(k+1)}; s\right)} = V_s^{\left(k, s_1^{(k+1)}, s_2^{(k+1)}, \dots s_{b-1}^{(k+1)}\right)} \hat{\theta}. \tag{31}$$

where $V_s^{(k, s_1^{(k+1)}, s_2^{(k+1)}, \dots s_{b-1}^{(k+1)})}$ is the feature vector that corresponds to location s. It is important to note that when $b = 1$ the *pseudo*-ERD is the actual ERD, because when $b = 1$ we know the values of all the sampling locations, and therefore, have no need for *pseudo*-measurements. Now we find the location that maximizes the *pseudo*-ERD by

$$s_b^{(k+1)} = \arg\max_{s \in \{\Omega \setminus \mathcal{S}\}} \left\{ \bar{R}^{(k, s_1^{(k+1)}, s_2^{(k+1)}, \dots s_{b-1}^{(k+1)}; s)} \right\}. \tag{32}$$

Then finally we update the set of measured locations by

$$\mathcal{S} \leftarrow \left\{ S \cup s_b^{(k+1)} \right\}. \tag{33}$$

Fig. 6 shows a detailed illustration of the proposed group-wise SLADS method.

VI. Results

In the following sections, we first validate the approximation to the RD, and we then compare SLADS to alternative sampling approaches based on both real and simulated data. We next evaluate the stopping condition presented in Section IV and finally compare the group-wise SLADS method presented in Section V with SLADS. The distortion metrics and the reconstruction methods we used in these experiments are detailed in Appendices VIII-A and VIII-B. We note that all experiments are started by first acquiring 1% of the image using low-discrepancy sampling [10]. Furthermore, for all the experiments, we set $\lambda = 0.25$ for the descriptor $Z_{s,6}$ in Table I, $|\partial s| = 10$ for all descriptors, and for the training described in Fig. 3, we chose $p_1 = 0.05$, $p_2 = 0.10$, $p_3 = 0.20$, $p_4 = 0.40$, $p_5 = 0.80$ as the fractional sampling densities.

A. Validating the RD Approximation

In this section, we compare the results using the true and approximate RD described in Section III-B in order to validate the efficacy of the approximation. The SLADS algorithm was trained and then tested using the synthetic EBSD images shown in Fig. 7(a) and (b). Both sets of images were generated using the Dream.3D software [36].

The training images were constructed to have a small size of 64×64 so that it would be tractable to compute the true

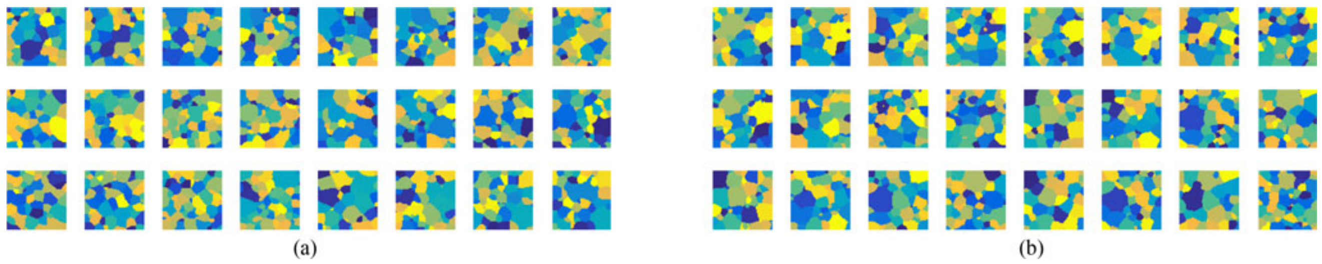


Fig. 7. Images used for (a) training and (b) testing in Section VI-A for the validation of the RD approximation. Each image is a 64×64 synthetic EBSD image generated using the Dream.3D software with the different colors corresponding to different crystal orientation.

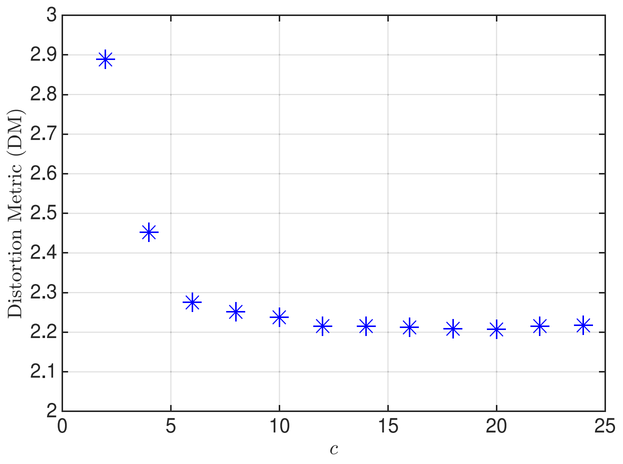


Fig. 8. Plot of overall distortion of (23) versus the parameter c for the experiment of Section VI-A. The optimal value of c^* is chosen to minimize the overall distortion, which in this case is $c^* = 20$.

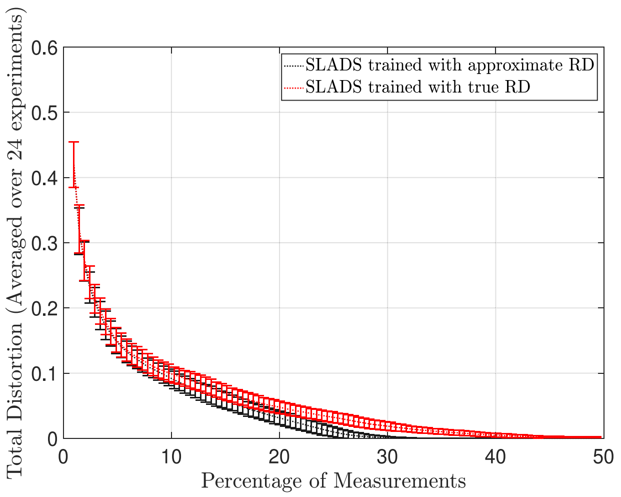


Fig. 9. Plots of the total distortion, $\overline{TD}_k$ versus the percentage of samples taken. The red plot is the result of training with the true RD value, and the blue plot is the result of training with the approximate RD value. Both curves are averaged over 24 experiments with error bars indicating the standard deviation.

reduction-in-distortion from (15) along with the associated true regression parameter vector $\hat{\theta}^{true}$. This allowed us to compute the true ERD for this relatively small problem.

We selected the optimal parameter value, c^*, using the method described in Section III-C from the possible values $c \in \{2, 4, 6, \ldots 24\}$. Fig. 8 shows a plot of the distortion metric, $DM^{(c_i)}$, defined in (23) versus c_i. In this case, the optimal parameter value that minimizes the overall distortion metric is given by $c^* = 20$. However, we also note that the metric is low for a wide range of values.

Fig. 9 shows a plot of the total distortion, $\overline{TD}_k$, versus the percentage of samples for both the true regression parameter vector, $\hat{\theta}^{true}$, and the approximate regression parameter vector, $\hat{\theta}^{(c^*)}$. Both curves include error bars to indicate the standard deviation. While the two curves are close, the approximate reduction-in-distortion results in a lower curve than the true reduction-in-distortion.

While it is perhaps surprising that the approximate RD training results in a lower average distortion, we note that this is not inconsistent with the theory. Since the SLADS algorithm is greedy, the most accurate algorithm for predicting the ERD does not necessarily result in the fastest overall reduction of the total distortion. Moreover, since the value of the parameter c^* is determined by operationally minimizing the total distortion during sampling, the approximate RD training has an advantage over the true RD training.

Fig. 10 provides additional insight into the difference between the results using approximate and true RD training. The figure shows both the measured location and reconstructed images for the approximate and true methods with 10%, 20%, 30% and 40% of the image pixels sampled. From this figure we see that when the approximate RD is used in training, the boundaries are more densely sampled at the beginning. This has the effect of causing the total distortion to decrease more quickly.

B. Results Using Simulated EBSD Images

In this section, we first compare SLADS with two static sampling methods – Random Sampling (RS) [7] and Low-discrepancy Sampling (LS) [10]. Then we evaluate the group-wise SLADS method introduced in Section V and finally we evaluate the stopping method introduced in Section IV. Fig. 11(a) and (b) show the simulated 512×512 EBSD images we used for training and testing, respectively, for all the experiments in this section. All results used the parameter value $c^* = 10$ that was estimated using the method described in Section III-C. The average total distortion, $\overline{TD}_k$, for the experiments was computed over the full test set of images.

Fig. 12 shows a plot of the average total distortion, $\overline{TD}_k$, for each of the three algorithms that were compared, LS, RS and SLADS. Notice that SLADS dramatically reduces error relative

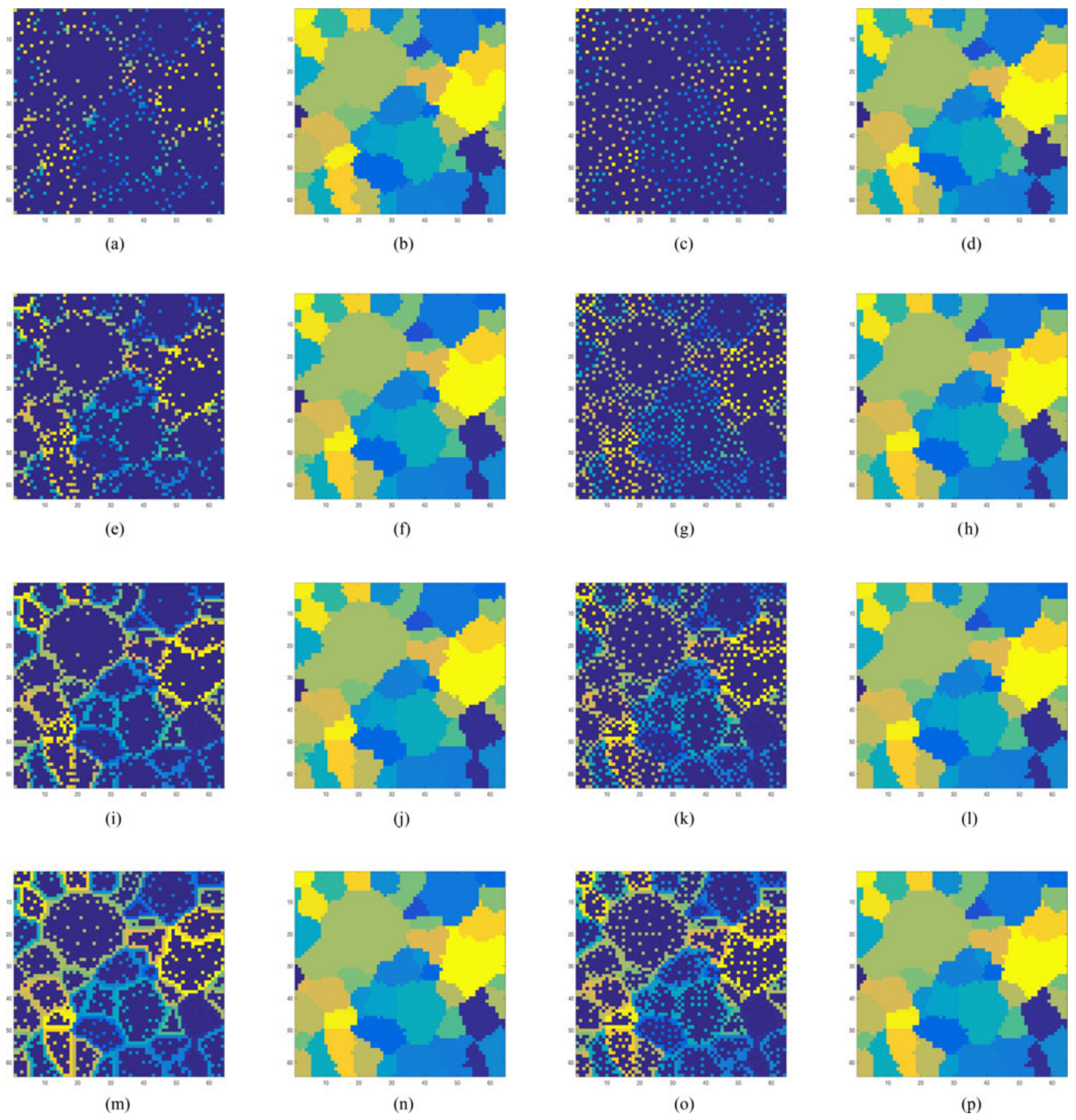


Fig. 10. Images illustrating the sampling locations and reconstructions after 10%, 20%, 30%, and 40% of sampling. The first two columns correspond to the sample points and reconstruction using training with the approximate RD value. The last two columns correspond to the sample points and reconstruction using training with the true RD value. Notice that the approximate training results in greater concentration of the samples near the region boundaries. (a) Measured locations 10%—trained with approximate RD. (b) Reconstructed image 10%—trained with approximate RD. (c) Measured locations 10%—trained with true RD. (d) Reconstructed image 10%—trained with true RD. (e) Measured locations 20%—trained with approximate RD. (f) Reconstructed image 20%—trained with approximate RD. (g) Measured locations 20%—trained with true RD. (h) Reconstructed image 20%—trained with true RD. (i) Measured locations 30%—trained with approximate RD. (j) Reconstructed image 30%—trained with approximate RD. (k) Measured locations 30%—trained with true RD. (l) Reconstructed image 30%—trained with true RD. (m) Measured locations 40%—trained with approximate RD. (n) Reconstructed image 40%—trained with approximate RD. (o) Measured locations 40%—trained with true RD. (p) Reconstructed image 40%—trained with true RD.

to LS or RS at the same percentage of samples, and that it achieves nearly perfect reconstruction after approximately 6% of the samples are measured.

Fig. 13 gives some insight into the methods by showing the sampled pixel locations after 6% of samples have been taken for each of the three methods. Notice that SLADS primarily samples in locations along edges, but also selects some points in uniform regions. This both localizes the edges more precisely while also reducing the possibility of missing a small region or "island" in the center of a uniform region. Alternatively, the LS and RS algorithms select sample locations independent of the measurements; so samples are used less efficiently, and the resulting reconstructions have substantially more errors along boundaries.

To evaluate the group-wise SLADS method we compare it with SLADS and LS. Fig. 14 shows a plot of the average total distortion, $\overline{TD}_k$, for SLADS, LS, group-wise SLADS with the

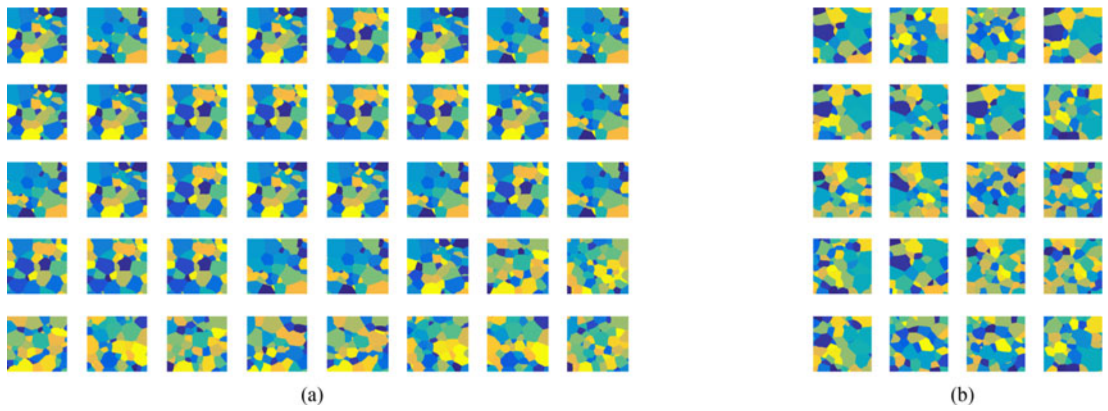


Fig. 11. Images used for (a) training and (b) testing in Section VI-B for the comparison of SLADS with LS and RS. Each image is a 512×512 synthetic EBSD image generated using the Dream.3D software with the different colors corresponding to different crystal orientation.

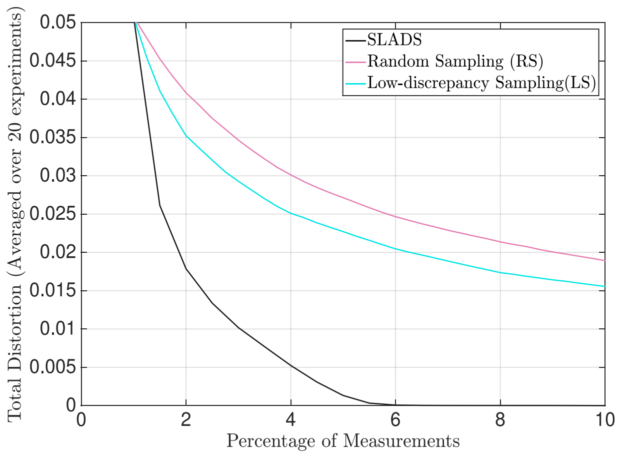


Fig. 12. Plot of the total distortion, $\overline{TD}_k$, for LS, RS, and SLADS, averaged over 20 experiments, versus the percentage of samples for the experiment detailed in Section VI-B.

group sampling rates of $B = 2, 4, 8$ and 16 performed on the images in Fig. 11(b). We see that group-wise SLADS has somewhat higher distortion for the same number of samples as SLADS and that the distortion increases with increasing values of B. This is reasonable since SLADS without group sampling has the advantage of having the most information available when choosing each new sample. However, even when collecting $B = 16$ samples in a group, the distortion is still dramatically reduced relative to LS.

We then evaluate the stopping method by attempting to stop SLADS at different distortion levels. In particular, we will attempt to stop SLADS when $TD_k \leq TD_{\text{desired}}$ for $TD_{\text{desired}} = \{5 \times 10^{-5}, 10 \times 10^{-5}, 15 \times 10^{-5} \ldots 50 \times 10^{-5}\}$. For each TD_{desired} value we found the threshold to place on the stopping function, in (26), by using the method described in Section IV on a subset of the images in Fig. 11(a). Again we used the images shown in Fig. 11(a) and (b) for training and testing, respectively. After each SLADS experiment stopped we computed the true TD value, TD_{true}, and then computed the average true TD value for a given TD_{desired}, $\bar{TD}_{\text{true}}$ (TD_{desired}), by averaging the TD_{true} values over the 20 testing images.

Fig. 15 shows a plot of $\bar{TD}_{\text{true}}$ (TD_{desired}) and TD_{desired}. From this plot we can see that the experiments that in general $TD_{\text{desired}} \geq \bar{TD}_{\text{true}}$ (TD_{desired}). This is the desirable result since we intended to stop when $TD_k \leq TD_{\text{desired}}$. However, from the standard deviation bars we see that in certain experiments the deviation from TD_{desired} is somewhat high and therefore note the need for improvement through future research.

It is also important to mention that the SLADS algorithm (for discrete images) was implemented for protein crystal positioning by Simpson *et al.* in the synchrotron facility at the Argonne National Laboratory [3].

C. Results Using Scanning Electron Microscope Images

In this section we again compare SLADS with LS and RS but now on continuously valued scanning electron microscope (SEM) images. Fig. 17(a) and (b) show the 128×128 SEM images used for training and testing, respectively. Using the methods described in Section III-C, the parameter value $c^* = 2$ was estimated and again the average total distortion, $\overline{TD}_k$ was computed over the full test set of images.

Fig. 18 shows a plot of $\overline{TD}_k$ for each of the three tested algorithms, SLADS, RS, and LS. We again see that SLADS outperforms the static sampling methods, but not as dramatically as in the discrete case.

Fig. 16 shows the results of the three sampling methods after 15% of the samples have been taken along with the resulting sampling locations that were measured. Once more we notice that SLADS primarily samples along edges, and therefore we get better edge resolution. We also notice that some of the smaller dark regions (“islands”) are missed by LS and RS while SLADS is able to resolve almost all of them.

D. Impact of Resolution and Noise on Estimation of c

In this section, we investigate the effect of image resolution on the estimate of the parameter c from (18). For this purpose, we used 10, 512×512 computationally generated EBSD images, and down-sampled them by factors of 1.25, 1.5, 1.75, 2, 4, 6 and 8, to create 7 additional lower resolution training image sets with resolution, $410 \times 410, 342 \times 342, 293 \times 292, 256 \times 256, 128 \times 128, 86 \times 86,$ and 64×64. Then, for each resolution, we estimated the optimal value of c and plotted the down-sapling factor versus the estimated value of c in Fig. 19.

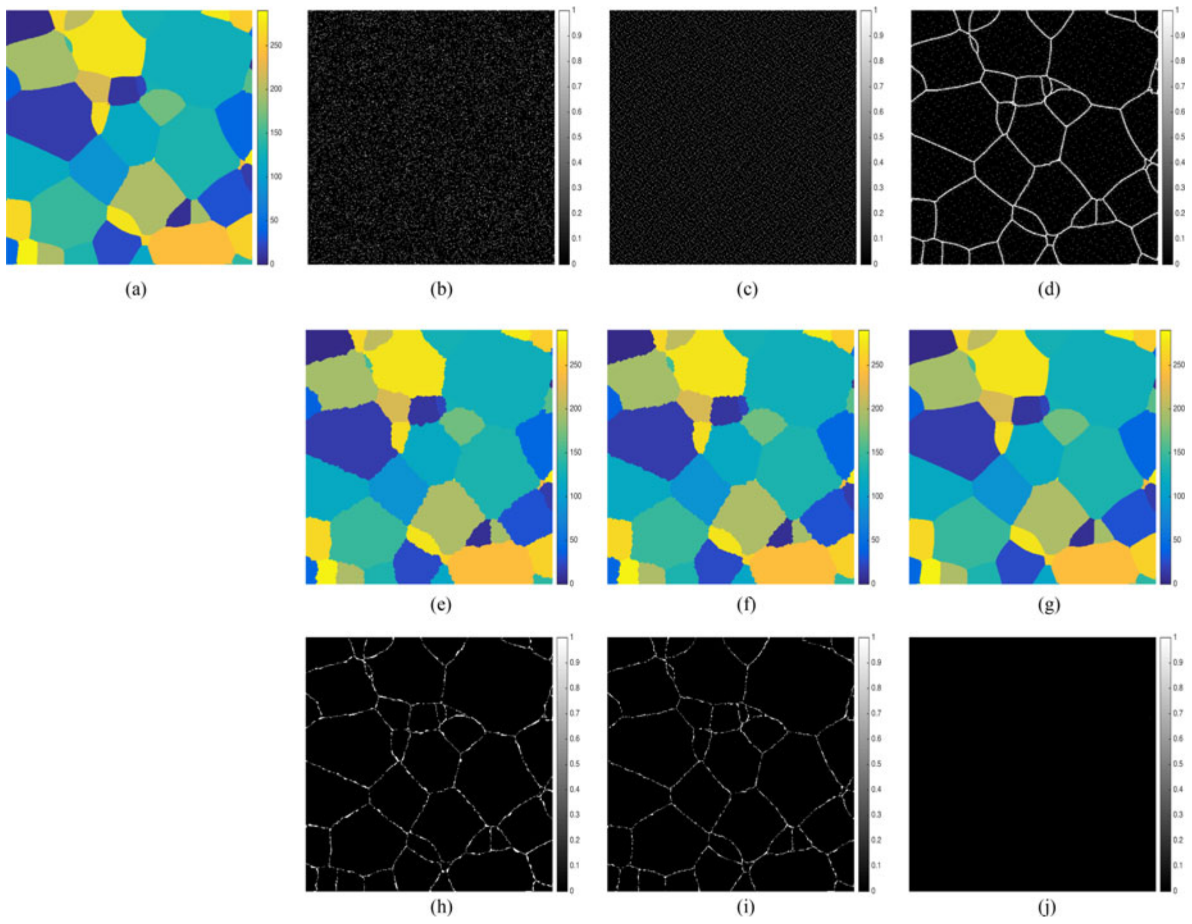


Fig. 13. Images illustrating the sampling locations, reconstructions and distortion images after 6% of the image is sampled. (a) Original image. (b) Random sample (RS) locations. (c) Low discrepancy sample (LS) locations. (d) SLADS sample locations. (e) Reconstruction using RS samples. (f) Reconstruction using LS samples. (g) Reconstruction using SLADS samples. (h) Distortion using RS samples. (i) Distortion using LS samples. (j) Distortion using SLADS samples.

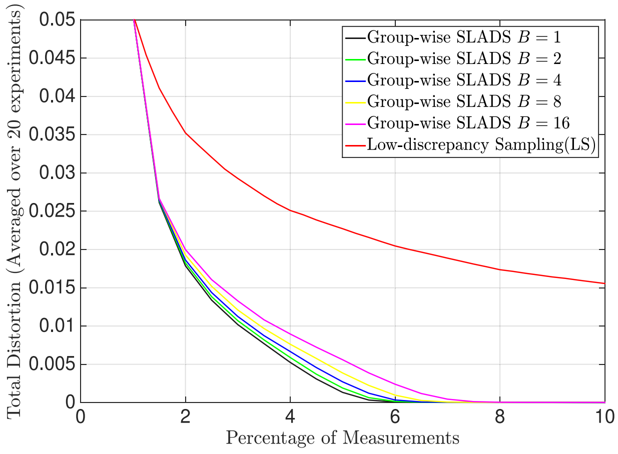


Fig. 14. Plot of the total distortion, $\overline{TD}_k$, versus the percentage of samples for groupwise SLADS with $B = 1, 2, 4, 8, 16$ and low-discrepancy sampling as detailed in Section VI-B. Results are averaged over 20 experiments.

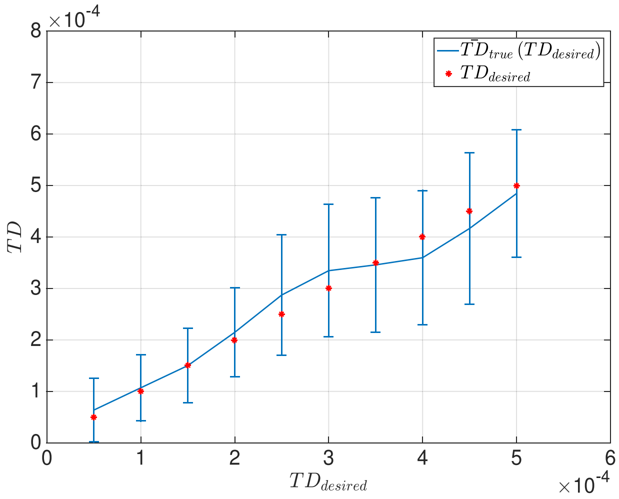


Fig. 15. Plot of the computed total distortion TD, averaged over ten experiments, versus the desired TD for experiment to evaluate stopping condition detailed in Section IV.

The value of c does increase somewhat as the image resolution decreases (i.e., the downsampling factor increases); however, there does not appear to be a strong correlation between c and the image resolution. The images were downsampled using the nearest-neighbor method in Matlab.

In order to investigate the effect of noisy training images in SLADS, we created 10 clean, 64×64 sized training images using the Dream.3D software. Then, we created training sets with different levels of noise by adding noise to these images.

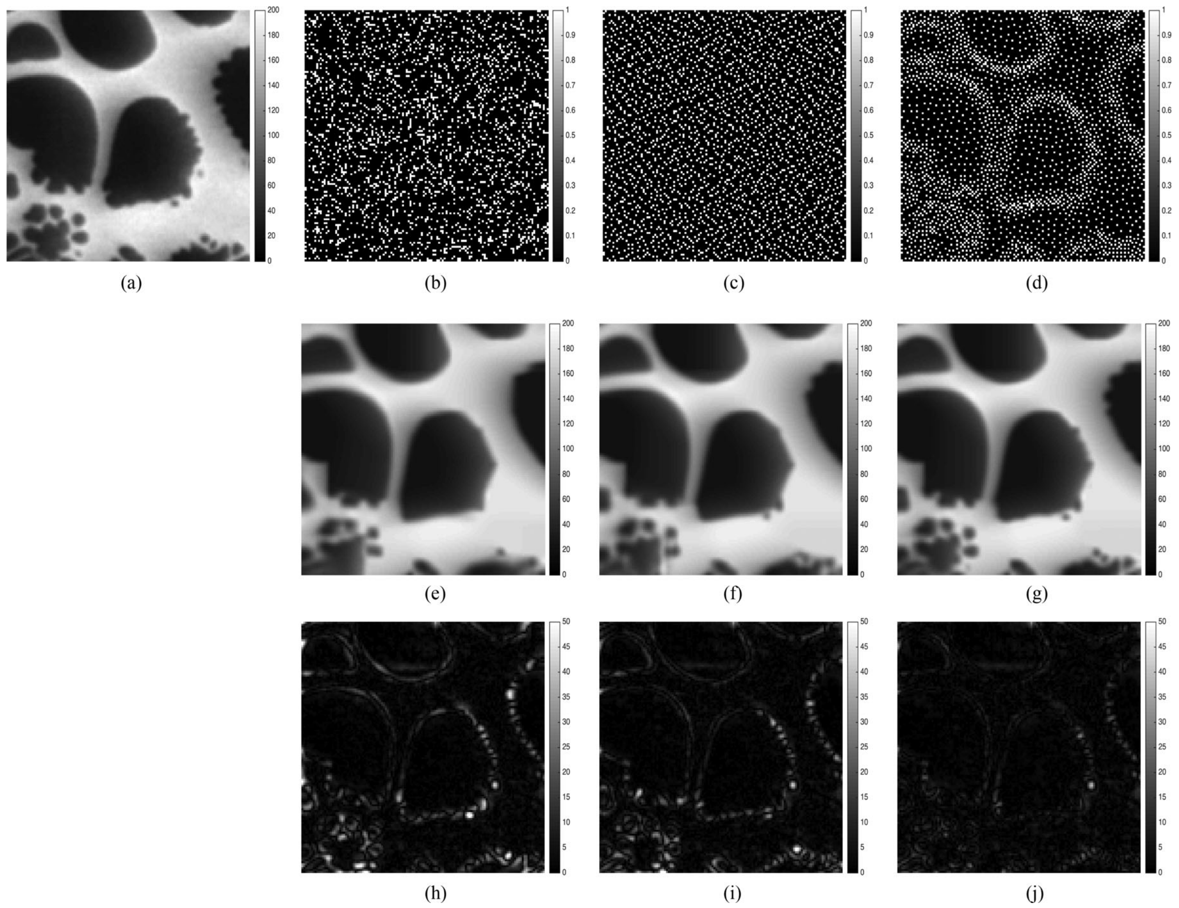


Fig. 16. Images illustrating the sampling locations, reconstructions, and distortion images after 15% of the image in (a) was sampled using RS, LS and SLADS for experiment detailed in Section VI-C. (b)–(d) Sampling locations. (e)–(g) Images reconstructed from measurements. (h)–(j) Distortion images between (a) and (e), (a) and (f), and (a) and (g). (a) Original Image. (b) RS: Sample locations ($\backsim$15%). (c) LS: Sample locations ($\backsim$15%). (d) SLADS: Sample locations ($\backsim$15%). (e) RS: Reconstructed Image. (f) LS: Reconstructed image. (g) SLADS: Reconstructed image. (h) RS: Distortion image ($TD = 3.88$). (i) LS: Distortion image ($TD = 3.44$). (j) SLADS: Distortion image ($TD = 2.63$).

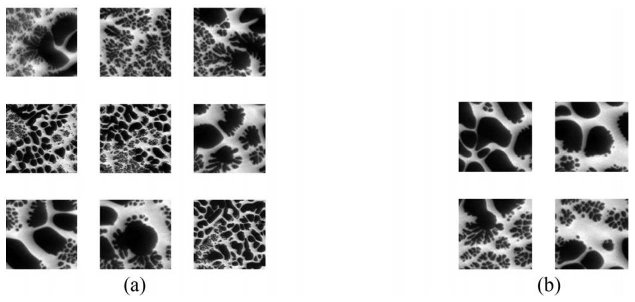


Fig. 17. Images used for (a) training and (b) testing in the experiment detailed in Section VI-C, in which we compared SLADS with LS and RS. These images have 128×128 pixels each, and experimental collected SEM images. These images were collected by Ali Khosravani & Prof. Surya Kalidindi from the Georgia Institute of Technology.

Since these are discretely labeled images, we define the level of noise as the probability of a pixel being mislabeled. In this experiment, the noise levels we chose were, 0.001, 0.005, 0.01, 0.02, 0.04, and 0.08. The resulting values of c are shown in Fig. 20 from which we see that there is no clear relationship between the noise level and the estimate of c.

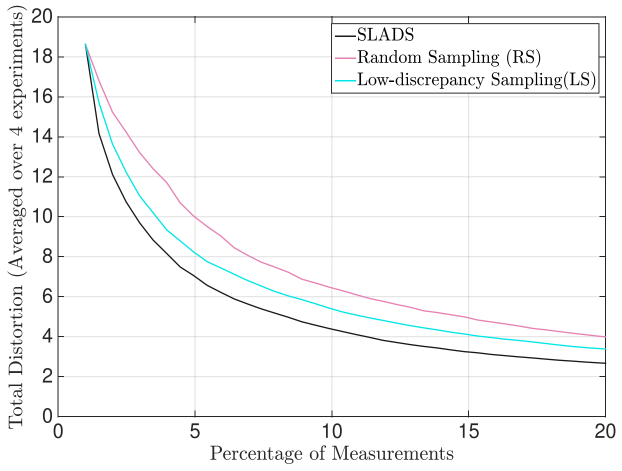


Fig. 18. Plot of the total distortion, $\overline{TD}_k$, for LS, RS, and SLADS, averaged over four experiments, versus the percentage of samples for the experiment detailed in Section VI-C.

VII. Conclusion and Future Directions

In this paper, we presented a framework for dynamic image sampling which we call a supervised learning approach for

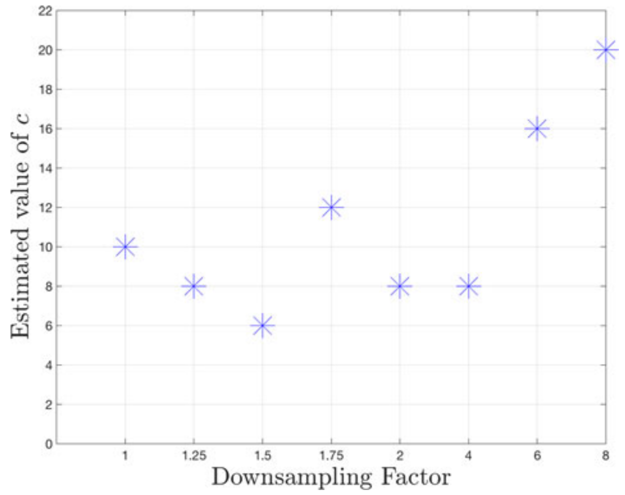


Fig. 19. Estimated value of the parameter c as a function of image resolution. Increased downsampling factor corresponds to lower image resolution.

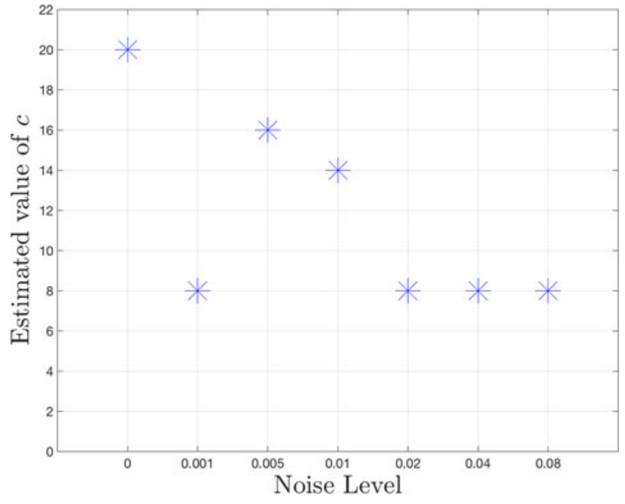


Fig. 20. Estimated value of the parameter c as a function of the level of noise in the training image set.

dynamic sampling (SLADS). The method works by selecting the next measurement location in a manner that maximizes the expected reduction in distortion (ERD) for each new measurement. The SLADS algorithm dramatically reduces the computation required for dynamic sampling by using a supervised learning approach in which a regression algorithm is used to efficiently estimate the ERD for each new measurement. This makes the SLADS algorithm practical for real-time implementation.

Our experiments show that SLADS can dramatically outperform static sampling methods for the measurement of discrete data. For example, SEM analytical methods such as EBSD [2], or synchrotron crystal imaging [3] are just two cases in which sampling of discrete images is important. We also introduced a group-wise SLADS method which allows for sampling of multiple pixels in a group, with only limited loss in performance. Finally, we concluded with simulations on sampling from continuous SEM images in which we demonstrated that SLADS provides modest improvements compared to static sampling.

Finally we note that our proposed dynamic sampling framework uses very simple machine learning techniques, and we believe that more sophisticated machine learning techniques should be able to achieve better performance [34]. Future work may also incorporate the impact of measurement noise in the SLADS model [37]. Finally, we believe that it is possible to extend the SLADS framework to applications such as tomography in which the effects of measurements are not necessarily local.

Appendix

A. *Distortion Metrics for Experiments*

Applications such as EBSD generate images formed by discrete classes. For these images, we use a distortion metric defined between two vectors $A \in \mathbb{R}^N$ and $B \in \mathbb{R}^N$ as

$$D(A,B) = \sum_{i=1}^{N} I(A_i, B_i), \tag{34}$$

where I is an indicator function defined as

$$I(A_i, B_i) = \begin{cases} 0 & A_i = B_i \\ 1 & A_i \neq B_i, \end{cases} \tag{35}$$

where A_i is the ith element of the vector A.

However, for the experiments in Section VI-C we used continuously valued images. Therefore, we defined the distortion $D(A,B)$ between two vectors A and B as

$$D(A,B) = \sum_{i=1}^{N} |A_i - B_i|. \tag{36}$$

B. *Reconstruction Methods for Experiments*

In the experiments with discrete images all the reconstructions were performed using the weighted mode interpolation method. The weighted mode interpolation of a pixel s is $X_{\hat{r}}$ for

$$\hat{r} = \arg\max_{r \in \partial s} \left\{ \sum_{t \in \partial s} \left[(1 - D(X_r, X_t)) \, w_r^{(s)} \right] \right\}, \tag{37}$$

where

$$w_r^{(s)} = \frac{\frac{1}{\|s-r\|^2}}{\sum_{u \in \partial s} \frac{1}{\|s-u\|^2}} \tag{38}$$

and $|\partial s| = 10$.

In the training phase of the experiments on continuously valued data, we performed reconstructions using the Plug & Play algorithm [38], [39] to compute the reduction-in-distortion. However, to compute the reconstructions for descriptors (both in testing and training) we used weighted mean interpolation instead of Plug & Play to minimize the run-time speed of SLADS. We define the weighted mean for a location s by

$$\hat{X}_s = \sum_{r \in \partial s} w_r^{(s)} X_r. \tag{39}$$

Acknowledgment

The authors would like to thank Ali Khosravani & Prof. Surya Kalidindi, Georgia Institute of Technology for providing the images used for dynamic sampling simulations on experimentally-collected images.

References

[1] D. Rugar and P. Hansma, “Atomic force microscopy,” *Phys. Today*, vol. 43, no. 10, pp. 23–30, 1990.

[2] A. J. Schwartz, M. Kumar, B. L. Adams, and D. P. Field, *Electron Backscatter Diffraction in Materials Science*. New York, NY, USA: Springer, 2009.

[3] N. M. Scarborough *et al.*, “Dynamic x-ray diffraction sampling for protein crystal positioning,” *J. Synchrotron Radiation*, vol. 24, no. 1, pp. 188–195, Jan. 2017.

[4] S. Keren, C. Zavaleta, Z. Cheng, A. de La Zerda, O. Gheysens, and S. S. Gambhir, “Noninvasive molecular imaging of small living subjects using raman spectroscopy,” *Proc. Nat. Acad. Sci.*, vol. 105, no. 15, pp. 5844–5849, 2008.

[5] R. Smith-Bindman, J. Lipson, and R. Marcus, “Radiation dose associated with common computed tomography examinations and the associated lifetime attributable risk of cancer,” *Arch. Internal Med.*, vol. 169, no. 22, pp. 2078–2086, 2009.

[6] R. F. Egerton, P. Li, and M. Malac, “Radiation damage in the TEM and SEM,” *Micron*, vol. 35, no. 6, pp. 399–409, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968432804000381

[7] H. S. Anderson, J. Ilic-Helms, B. Rohrer, J. Wheeler, and K. Larson, “Sparse imaging for fast electron microscopy,” *Proc. SPIE*, vol. 8657, 2013, Art. no. 86570C.

[8] K. Hujsak, B. D. Myers, E. Roth, Y. Li, and V. P. Dravid, “Suppressing electron exposure artifacts: An electron scanning paradigm with Bayesian machine learning,” *Microscopy Microanal.*, vol. 22, no. 4, pp. 778–788, 2016.

[9] S. Hwang, C. W. Han, S. Venkatakrishnan, C. A. Bouman, and V. Ortalan, “Towards the low-dose characterization of beam sensitive nanostructures via implementation of sparse image acquisition in scanning transmission electron microscopy,” *Meas. Sci. Technol.*, vol. 28, no. 4, Feb. 2017, Art. no. 045402.

[10] R. Ohbuchi and M. Aono, “Quasi-Monte Carlo rendering with adaptive sampling,” IBM Tokyo Research Laboratory, 1996.

[11] K. A. Mohan *et al.*, “TIMBIR: A method for time-space reconstruction from interlaced views,” *IEEE Trans. Comput. Imag.*, vol. 1, no. 2, pp. 96–111, Jun. 2015.

[12] S. Z. Sullivan *et al.*, “High frame-rate multichannel beam-scanning microscopy based on Lissajous trajectories,” *Opt. Express*, vol. 22, no. 20, pp. 24224–24234, 2014.

[13] K. Mueller, “Selection of optimal views for computed tomography reconstruction,” Patent WO 2 011 011 684, Jan. 28, 2011.

[14] Z. Wang and G. R. Arce, “Variable density compressed image sampling,” *IEEE Trans. Image Process.*, vol. 19, no. 1, pp. 264–270, Jan. 2010.

[15] M. W. Seeger and H. Nickisch, “Compressed sensing and Bayesian experimental design,” in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 912–919.

[16] W. R. Carson, M. Chen, M. R. D. Rodrigues, R. Calderbank, and L. Carin, “Communications-inspired projection design with application to compressive sensing,” *SIAM J. Imag. Sci.*, vol. 5, no. 4, pp. 1185–1212, 2012.

[17] J. Shihao, Y. Xue, and L. Carin, “Bayesian compressive sensing,” *IEEE Trans. Signal Process.*, vol. 56, no. 6, pp. 2346–2356, Jun. 2008.

[18] G. Braun, S. Pokutta, and Y. Xie, “ Info-greedy sequential adaptive compressed sensing,” *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 4, pp. 601–611, Jun. 2015.

[19] J. Vanlier, C. A. Tiemann, P. A. J. Hilbers, and N. A. W. van Riel, “A Bayesian approach to targeted experiment design,” *Bioinformatics*, vol. 28, no. 8, pp. 1136–1142, 2012.

[20] A. C. Atkinson, A. N. Donev, and R. D. Tobias, *Optimum Experimental Designs, with SAS*, vol. 34. Oxford, U.K.: Oxford Univ. Press, 2007.

[21] M. Seeger, H. Nickisch, R. Pohmann, and B. Schölkopf, “Optimization of k-space trajectories for compressed sensing by Bayesian experimental design,” *Magn. Reson. Med.*, vol. 63, no. 1, pp. 116–126, 2010.

[22] B. K. Joost, W. J. Palenstijn, P. Balázs, and J. Sijbers, “Dynamic angle selection in binary tomography,” *Comput. Vision Image Understanding*, vol. 117, pp. 306–318, 2013.

[23] T. Merryman and J. Kovacevic, “An adaptive multirate algorithm for acquisition of fluorescence microscopy data sets,” *IEEE Trans. Image Process.*, vol. 14, no. 9, pp. 1246–1253, Sep. 2005.

[24] C. Jackson, R. F. Murphy, and J. Kovacevic, “Intelligent acquisition and learning of fluorescence microscope data models,” *IEEE Trans. Image Process.*, vol. 18, no. 9, pp. 2071–2084, Sep. 2009.

[25] J. Haupt, R. Baraniuk, R. Castro, and R. Nowak, “Sequentially designed compressed sensing,” in *Proc. IEEE Statist. Signal Process. Workshop*, 2012, pp. 401–404.

[26] G. M. D. Godaliyadda, G. T. Buzzard, and C. A. Bouman, “A model-based framework for fast dynamic image sampling,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 1822–1826.

[27] W. K. Hastings, “Monte carlo sampling methods using Markov chains and their applications,” *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[28] P. H. Peskun, “Optimum Monte-Carlo sampling using Markov chains,” *Biometrika*, vol. 60, no. 3, pp. 607–612, 1973.

[29] G. D. Godaliyadda, D. Hye Ye, M. A. Uchic, M. A. Groeber, G. T. Buzzard, and C. A. Bouman, “A supervised learning approach for dynamic sampling,” in *Proc. Electron. Imag., Comput. Imag. XIV Conf.*, 2016, pp. 1–8.

[30] D. Shepard, “A two-dimensional interpolation function for irregularly-spaced data,” *Proc. 23rd ACM Nat. Conf.*, 1968, pp. 517–524.

[31] S. Lee, W. George, and S. Y. Shin, “Scattered data interpolation with multilevel b-splines,” *IEEE Trans. Vis. Comput. Graph.*, vol. 3, no. 3, pp. 228–244, Jul.–Sep. 1997.

[32] A. J. Smola and S. Chölkopf, “A tutorial on support vector regression,” *Statist. Comput.*, vol. 14, no. 3, pp. 199–222, 2004.

[33] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[34] Y. Zhang, G. M. D. Godaliyadda, N. Ferrier, E. B. Gulsoy, C. A. Bouman, and C. D. Phatak, “SLADS-Net: Supervised learning approach for dynamic sampling using deep neural networks,” *Electron. Imag.*, 2018, to be published.

[35] Y. Zhang, G. M. D. Godaliyadda, Y. S. G. Nashed, N. Ferrier, E. B. Gulsoy, and C. D. Phatak, “Under-sampling and image reconstruction for scanning electron microscopes,” *Microscopy Microanal*, vol. 23, pp. 136–137, 2017.

[36] M. A. Groeber and M. A. Jackson, “Dream.3D: A digital representation environment for the analysis of microstructure in 3D,” *Integrating Mater. Manuf. Innovation*, vol. 19, pp. 1–17, 2014.

[37] Y. Zhang, G. D. Godaliyadda, N. Ferrier, E. B. Gulsoy, C. A. Bouman, and C. Phatak, “Reduced electron exposure for energy-dispersive spectroscopy using dynamic sampling,” *Ultramicroscopy*, vol. 184, pp. 90–97, 2018.

[38] S. Sreehari *et al.*, “Plug-and-play priors for bright field electron tomography and sparse interpolation,” *IEEE Trans. Comput. Imag.*, vol. 2, no. 4, pp. 408–423, Dec. 2016.

[39] S. Sreehari, S. V. Venkatakrishnan, L. F. Drummy, J. P. Simmons, and C. A. Bouman, “Advanced prior modeling for 3D bright field electron tomography,” *Proc. SPIE*, vol. 9401, 2015, Art. no. 940108.

[40] H. Xun and Y. M. Marzouk, “Simulation-based optimal Bayesian experimental design for nonlinear systems,” *J. Comput. Phys.*, vol. 232, no. 1, pp. 288–317, 2013.

[41] C. M. Dettmar, J. A. Newman, S. J. Toth, M. Becker, R. F. Fischetti, and G. J. Simpson, “Imaging local electric fields produced upon synchrotron x-ray exposure,” *Proc. Nat. Acad. Sci.*, vol. 112, no. 3, pp. 696–701, 2015.

[42] W. P. Burmeister, “Structural changes in a cryo-cooled protein crystal owing to radiation damage,” *Acta Crystallographica*, vol. 56.3, pp. 328–341, 2000.

[43] J. M. Holton, “A beginner’s guide to radiation damage,” *J. Synchrotron Radiation*, vol. 16, no. 2, pp. 133–142, 2009.

[44] C. Nave and E. F. Garman, “Towards an understanding of radiation damage in cryocooled macromolecular crystals,” *J. Synchrotron Radiation*, vol. 12, no. 3, pp. 257–260, 2005.

[45] F. Bergeaud and M. Stephane, “Matching pursuit of images,” in *Proc. Int. Conf. Image Process.*, 1995, vol. 1, pp. 607–612.

[46] A. C. Buades and J. M. Morel, “A non-local algorithm for image denoising,” in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2005, vol. 2, pp. 60–65.

[47] K. Dabov, A. Foi, and K. Egiazarian, “Image denoising by sparse 3-D transform-domain collaborative filtering,” *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[48] M. Aharon, M. ELad, and A. Bruckstein, “K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation,” *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[49] S. Sreehari, S. V. Venkatakrishnan, J. Simmons, L. Drummy, and C. A. Bouman, “Model-based super-resolution of SEM images of nano-materials,” *Microscopy Microanal.*, vol. 22, no. S3, pp. 532–533, 2016.

[50] Y. Saad, *Iterative Methods for Sparse Linear Systems*. Philadelphia, PA, USA: SIAM, 2003.

Authors’ photographs and biographies not available at the time of publication.