Sequential Adaptive Design for Jump Regression Estimation

Abstract

Selecting input variables or design points for statistical models has been of great interest in adaptive design and active learning. Motivated by two scientific examples, this paper presents a strategy of selecting the design points for a regression model when the underlying regression function is discontinuous. The first example we undertook was for the purpose of accelerating imaging speed in a high resolution material imaging, and the second was to use sequential design for mapping a chemical phase diagram. In both examples, the underlying regression functions have discontinuities, and thus many existing design optimization approaches cannot be used because they assume a continuous regression function. Although some existing adaptive design strategies developed from the treed regression models can handle the discontinuities, the related Bayesian model estimation approaches come with computationally expensive Markov Chain Monte Carlo algorithms for posterior inferences and the subsequent design point selections, which may not be applicable for the first motivating example that requires the computation to be faster than the original imaging speed. In addition, the treed models are based on domain partitioning and inefficient in cases when the discontinuities occur at complex sub-domain boundaries. In this paper, we propose a simple and effective adaptive design strategy for regression analysis with discontinuities. After some statistical properties of the estimated regression model are derived in cases with a fixed design, a new criterion for selecting the design points sequentially is suggested. The suggested sequential design selection procedure is then evaluated using a comprehensive simulation study, and demonstrated using the two motivating examples.

Keywords: Active learning, Sequential adaptive design, Adaptive sensing, Discontinuous response surfaces

1 Introduction

Regression analysis is a powerful statistical tool for estimating a regression function that relates explanatory variables to a response variable. In a typical regression analysis, the design points are assumed to be given in advance. When the design points can be selected during a data collection process, optimizing the selection is referred to as optimal design (Chernoff, 1972), active learning (Cohn et al., 1996), or adaptive sensing (Arias-Castro et al., 2013; Malloy and Nowak, 2014). This paper aims to address the problem of selecting the design points for a regression model in cases when the underlying regression function is piecewise continuous, which is motivated by two scientific applications described below.

The first motivating application is the material imaging with scanning transmission electron microscopy (STEM). The STEM technique is an important material characterization tool to image the microstructure of a material specimen at a fine spatial resolution. It uses a focused beam of electrons to probe a material specimen, and the intensity of the beam interacting with the specimen is measured for every focus location. This sequential imaging process will create a rastered image of the material specimen as shown in Fig. 1-(a), where a pixel corresponds to one focus location over the raster, and the corresponding intensity is the pixel intensity. The radius of the focused beam can be below 10^{-10} m, which allows a specimen to be imaged at a very fine spatial resolution; however, this level of detail is also the major reason for a slow imaging speed. The existing approaches to accelerate the imaging speed are based on a partial scan (i.e., scan a material specimen only at selected pixel locations). In the existing approaches (Stevens et al., 2015), the partial set is randomly selected from an uniform distribution over the raster locations, which can cause loss in spatial resolution. Optimizing the pixel locations in the partial set is highly desirable to mitigate the information loss. In this example, the image intensity surface can be regarded as a 2D regression function of the pixel locations. This function would be piecewise continuous because the image intensity could suddenly change at the boundary between different base materials of the imaging specimen, as shown in Fig. 1-(a). Selecting the pixel locations for the partial scan can be formulated as a design optimizion problem for estimating the underlying 2D piecewise continuous regression function.

The second motivating application is to optimize the design of experiments for effectively exploring a chemical phase diagram in chemistry. A phase diagram is a map that relates different experimental conditions to the physical states of materials. The physical state suddenly changes from one state to another around the experimental conditions where phase transitions occur, as illustrated in Fig. 1-(b). Typically the elucidation of a phase diagram requires a large number of experiments to be performed to probe possible physical states that may exist in the experimental phase-space. Optimizing the design of experiments is thus essential for an effective probing process. In particular, we are interested in carbon nanotube growth experiments to study the chemical conditions required for good nanotube growth. The chemical conditions include the reaction temperature and a relative ratio of two chemical ingredients (i.e., a reducing agent and an oxidant). The total nanotube growth changes abruptly around the boundary condition in the relative ratio for a given temperature. Therefore, the total nanotube growth is a piecewise continuous function of the relative ratio and temperature. Optimization the experimental design for relating the two conditions to the nanotube growth can be formulated as the problem of selecting the design points for estimating the piecewise continuous function.

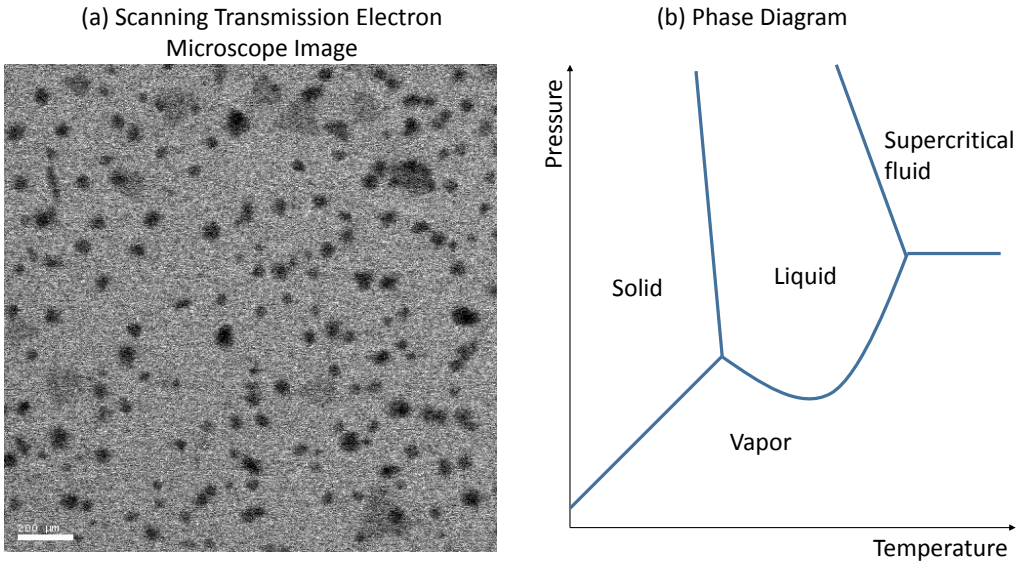

Figure 1: Two motivating examples of design optimization in jump regression analysis.

In both motivating applications described above, the underlying regression functions are piecewise continuous in low-dimensional design spaces. In many existing studies of sequential design and active learning, however, the underlying regression function is often assumed to be a continuous function and many design selection approaches have been developed under that assumption as we will review in Section 2. In this paper, we develop a simple and efficient sequential adaptive design strategy for estimating a piecewise continuous regression function, and present numerical evaluations of this new strategy for regression problems with two or three input dimensions.

The remainder of this paper is organized as follows. In Section 2, we review the existing research in active learning and sequential design, and emphasize the needs for a new adaptive design for our two motivating applications. Section 3 describes an approach of jump regression analysis for estimating a discontinuous regression function and discusses its statistical properties in cases with a fixed design. This approach is then used for developing a novel sequential adaptive design strategy for jump regression analysis. Section 4 presents numerical studies to evaluate the performance of the proposed method. Sections 5 and 6 illustrate the proposed method using the two motivating applications. Finally, Section 7 concludes the article with some summary statements.

2 Related Work

The design optimization problem has been studied in the sequential adaptive design and active learning literature. Sequential adaptive designs and active learning are branched out from different disciplines, so they have slightly different focuses. Sequential adaptive designs are studied in design of experiments, where the designs are often optimized for improving the parameter estimates of statistical models (Sacks et al., 1989). Active learning originates from the machine learning society to optimize data collection for improving the accuracy

of machine learning models (Cohn et al., 1996). Nonetheless, they are similar in the sense that they both aim to optimize data acquistion for improving estimated statistical models.

Many experimental design and active learning strategies have been developed for regression modeling with continuous regression functions. Sequential experimental design allows experiments to be conducted sequentially in order to exploit past experimental outcomes for guiding the design of future experiments. Many sequential design strategies have been developed for parametric regression models (Chaudhuri and Mykland, 1993; Dror and Steinberg, 2008) and nonlinear regression models, using smoothing approaches such as kernel regression (Zhao and Yao, 2012), wavelet regression Bull (2013), and Gaussian process regression (Zhu and Stein, 2006; Zimmerman, 2006). In active learning, selecting design points was studied for many nonparametric regression models, such as the Gaussian process regression models (Krause et al., 2008; Singh et al., 2009; Hoang et al., 2014) and the kernel-based regression models (Paisley et al., 2010). These existing approaches are not designed for estimating regression models with discontinuous regression functions.

There are a few existing adaptive design strategies for estimating piecewise continuous regression functions. By these methods, the input domain is first partitioned into subdomains, and then a simple regression model is fitted in each sub-domain. Depending on the ways of domain partitioning, we can group the existing methods into two groups: Bayesian tree-based approaches and Voronoi tessellation-based approaches. The Bayesian tree-based approaches recursively partition the input domain along one of the axis directions. Malloy and Nowak (2014) and Goetz et al. (2018) studied adaptive learning strategies using such approaches for estimating piecewise constant functions. Bull (2013) discussed an active learning strategy for estimating spatially inhomogeneous regression functions, including piecewise constant functions and functions with sharp bumps. But this method was limited to cases with a single explanatory variable. Gramacy and Lee (2009) discussed a treed Gaussian process model and the corresponding sequential design strategies, involving a treed-partitioning and GP leaves. Its posterior inference involved computationally inefficient reversible-jump methods for MCMC or higher-dimensional particles for sequential inference. Taddy et al. (2011) proposed a dynamic tree model with constant or linear leaves and developed an adaptive design strategy for the model. It does not involve MCMC sampling steps, and thus is computationally more efficient.

There are also methods using the Voronoi tessellation-based domain partitioning for estimating regression models with piecewise continuous regression functions. The method in Kim et al. (2005) first partitions the input domain by the Voronoi tessellation of training inputs, and then models the regression surface in each sub-domain as a local Gaussian process. Pope et al. (2021) discusses an adaptive sampling strategy for learning the piecewise regression model. The major issue with the tessellation-based methods is that the Voronoi tessellation works practically in only spatial domains,and is computationally expensive in 3D. The tessellation in higher dimensions is known to be a NP-hard problem. It is not surprising that these works are only applied to 2D spatial modeling problems. There exist other partitioning-based regression GP models used to model non-stationarity (Cortes et al., 2019; Lee et al., 2021).

Another potentially related approach is the adaptive design for estimating the contour of the underlying regression function $m(x)$ (Ranjan et al., 2008), which sequentially selects

the design points for estimating the contour of the subregion $m(x) < c$, for some constant c. When the discontinuities in $m(x)$ occur at a constant level c, estimating the contour at c would create a partition of the regression domain around the discontinuities and thus allow to use several regression models in sub-domains. The major limitation to use this approach in the two motivating applications is that the level c should be known as priori, and its estimation is not straightforward because the discontinuities do not always occur at a single level c (e.g., material images on uneven background such as MG5 and MG6 in Fig. 9 in the first motivating application).

To sum up, the piecewise regression modeling approaches explicitly partition the input domain to model separate regression functions in sub-domains. The joint learning of the partitioning and subregional model fitting often involves computationally expensive MCMC samplings. Dynamic trees (Taddy et al., 2011) are simple and more computationally feasible, but they are limited to piecewise constant or linear models. A more substantial issue around a piecewise regression model is that the explicit partitioning scheme is not flexible enough to model curvy discontinuities in the regression function efficiently. The tree-based models partition the input domain along one of the axis-aligned directions. The tesselation-based partitioning is applicable only for a spatial domain. As a comparison, the jump regression analysis (Qiu, 2005) provides a simpler modeling approach for a broader class of piecewise continuous regression functions. The major novelty in this paper is that we develop a simple and flexible nonparametric approach for estimating piecewise continuous regression functions, and a sequential adaptive design strategy for jump regression analysis. The new approach is applicable in higher dimensions and more computationally efficient, compared to the existing methods. The proposed modeling approach meets the time constraint and model adequacy of the two motivating applications.

3 Method

Let $\mathcal{X}$ denote a closed subset of $\mathbb{R}^p$ that represents a design space in a regression modeling problem. We consider a general jump regression model that aims to estimate a nonparametric regression function $m : \mathcal{X} \to \mathbb{R}$ from its noisy observations that follow the model

$$Y_i := m(\boldsymbol{x}_i) + \epsilon_i, \tag{1}$$

where $\{Y_i; i = 1, \ldots, n\}$ are noisy observations of the response variable Y at the design points $\{\boldsymbol{x}_i \in \mathcal{X}; i = 1, \ldots, n\}$, and $\{\epsilon_i; i = 1, \ldots, n\}$ are random errors with mean zero and variance σ^2. The underlying regression function m is further assumed to be piecewise continuous such that there exists a partition $\{\mathcal{A}_b; b = 1, \ldots, B\}$ of the design space $\mathcal{X}$ satisfying

(a) Each $\mathcal{A}_b$ is a simple connected (nonempty) subset of $\mathcal{X}$,

(b) $\cup_{b=1}^{B} \mathcal{A}_b = \mathcal{X}$, and $\mathcal{A}_b \cap \mathcal{A}_{b'} = \emptyset$, for any $b \neq b'$,

(c) The function $m(\boldsymbol{x})$ has the expression

$$m(\boldsymbol{x}) = \sum_{b=1}^{B} g_b(\boldsymbol{x}) I_{\mathcal{A}_b}(\boldsymbol{x}), \qquad \text{for } \boldsymbol{x} \in \mathcal{X}.$$

where $g_b(\boldsymbol{x}) \in \mathcal{C}^2(\mathcal{X})$ is a smooth function, for each b. Thus, the regression function $m(\boldsymbol{x})$ is continuous in $\mathcal{A}_b \backslash \partial\mathcal{A}_b$, where $\partial\mathcal{A}_b$ is the boundary set of $\mathcal{A}_b$, and it has jumps over $\mathcal{B} := \cup_{b=1}^{B} \partial\mathcal{A}_b$. For any $\boldsymbol{x}^* \in \mathcal{B}$, there exists b and b' such that $\boldsymbol{x}^* \in \partial\mathcal{A}_b \cap \partial\mathcal{A}_{b'}$ and

$$\lim_{\boldsymbol{x}\to\boldsymbol{x}^*, \boldsymbol{x}\in\mathcal{A}_b} g_b(\boldsymbol{x}) \neq \lim_{\boldsymbol{x}\to\boldsymbol{x}^*, \boldsymbol{x}\in\mathcal{A}_{b'}} g_{b'}(\boldsymbol{x}).$$

The boundary set $\mathcal{B}$ is referred to as the jump location curves (JLCs) of m in the literature (Qiu, 1998).

(d) The boundary is smooth so that a tangent line exists almost everywhere. A point in the boundary set $\mathcal{B}$ is called non-singular when there exists a unique tangent line at the point. Otherwise, it is called singular. We denote a collection of all singular boundary points by $\mathcal{S}$. Then, for any $\boldsymbol{x}^* \in \mathcal{B}\backslash\mathcal{S}$, there exists a unique pair (b, b') such that $\boldsymbol{x}^* \in \partial\mathcal{A}_b \cap \mathcal{A}_{b'}$. Otherwise, its tangent line would not be unique.

(e) The jump size between $\mathcal{A}_b$ and $\mathcal{A}_{b'}$ at $\boldsymbol{x}^* \in \partial\mathcal{A}_b \cap \partial\mathcal{A}_{b'}$ is defined to be

$$\delta_{b,b'}(\boldsymbol{x}^*) = \lim_{\boldsymbol{x}\to\boldsymbol{x}^*, \boldsymbol{x}\in\mathcal{A}_b} g_b(\boldsymbol{x}) - \lim_{\boldsymbol{x}\to\boldsymbol{x}^*, \boldsymbol{x}\in\mathcal{A}_{b'}} g_{b'}(\boldsymbol{x}).$$

It is assumed that $\delta_{b,b'}(\boldsymbol{x}^*) \neq 0$ and they have the same sign, for any $\boldsymbol{x}^* \in \partial\mathcal{A}_b \cap \partial\mathcal{A}_{b'}$.

Estimation of $m(\boldsymbol{x})$ has been studied using two different approaches. By the first approach, the partition $\{\mathcal{A}_b; b = 1, \ldots, B\}$ and the corresponding JLCs are estimated first, and then $m(\boldsymbol{x})$ is estimated using the conventional local smoothing procedures (e.g., kernel smoothing methods) within each sub-region $\mathcal{A}_b$ (Qiu and Yandell, 1997). By the second approach, the regression function $m(\boldsymbol{x})$ is estimated by one-sided kernel smoothing estimate, without explicit estimation of the JLCs (Qiu, 2009). However, optimizing the selection of the design points in a jump regression model has not been studied in these papers. The current paper aims to develop a design selection strategy for jump regression analysis, primarily for applications such as the two motivating applications discussed in Section 1; but it is general enough to be applicable for other similar problems. To describe this design selection strategy, we first discuss estimation of $m(\boldsymbol{x})$ in a fixed design case and then derive the asymptotic bias and variance of the estimator in Section 3.1. The bias and variance of the estimator will be related to the choice of the design points, and this relationship will be exploited to develop a new sequential experimental decision making algorithm that selects the design points to reduce the bias and variance of the estimated response surface in Section 3.2.

3.1 Model estimation in a fixed design case

Given observations $Y_1, \ldots, Y_n$ at the design points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, we discuss nonparametric estimation of $m(\boldsymbol{x})$, based on the one-sided local linear kernel smoothing approach (Qiu, 2009). In this paper, we extend the approach with two modifications for our scientific applications. First, it is assumed that the design points are sparsely located in $\mathcal{X}$, and their locations are non-uniformly distributed over $\mathcal{X}$ as a result of optimizing the choice of design points in sequential design cases and other reasons. To accommodate such non-uniformly distributed design points, we use spatially varying kernel bandwidth, instead of a constant bandwidth used in Qiu (2009). Second, we extend the method from 2D cases (i.e.,

$\boldsymbol{x}$ has a dimension of 2) to cases with two or more dimensions. In the scientific applications discussed in the paper, the dimension is two. We tested the proposed approach in 3D cases using simulated datasets.

The one-sided kernel smoothing approach does not require explicit estimation of $\{\mathcal{A}_b\}$, and it gives a pointwise estimate of the regression function $m(\boldsymbol{x})$ directly with the jumps in $m(\boldsymbol{x})$ being accommodated automatically. At a given location $\boldsymbol{x} \in \mathcal{X}$, consider its neighborhood with the bandwidth h:

$$\mathcal{N}(\boldsymbol{x}) = \{\boldsymbol{x}' \in \mathcal{X} : d(\boldsymbol{x}', \boldsymbol{x}) \leq h\},$$

where $d(\cdot, \cdot)$ is the Euclidean distance. We seek a local estimate of $m(\boldsymbol{x})$ using observed data in the neighborhood $\mathcal{N}(\boldsymbol{x})$. In cases when the design points are uniformly distributed in $\mathcal{X}$, a global bandwidth parameter is typically used as a function of the sample size n. In this paper, we allow the design points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ to be sampled from a non-uniform density $f(\boldsymbol{x})$, due to the design selection procedure that will be discussed in the next section. To be more adaptive to the non-uniform density, we adopt spatially varying bandwidth parameters. Let $h_n(\boldsymbol{x})$ denote the location-dependent bandwidth parameter, which is set to be the Euclidean distance from $\boldsymbol{x}$ to its kth nearest neighbor (k-NN) in $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. The corresponding neighborhood of $\boldsymbol{x}$ is defined to be

$$\mathcal{N}_n(\boldsymbol{x}) := \{\boldsymbol{x}' \in \mathcal{X} : d(\boldsymbol{x}', \boldsymbol{x}) \leq h_n(\boldsymbol{x})\}.$$

Based on the existing literature on the k-NN density estimation (Wasserman, 2006), the k-NN bandwidth selection is asymptotically equivalent to selecting the bandwidth parameter to be inversely proportional to the density of the design points, i.e.,

$$h_n(\boldsymbol{x}) \propto \left(\frac{1}{nf(\boldsymbol{x})}\right)^{1/p}. \tag{2}$$

Based on this asymptotic relationship, k should be chosen such that $k = o(n)$ and $k \to \infty$, as $n \to \infty$ (Mack and Rosenblatt, 1979). Our choice in this paper is $k = \sqrt{n}$.

For the conventional local linear kernel smoother, a local estimate of $m(\boldsymbol{x})$, for any location $\boldsymbol{x} \in \mathcal{X}$, is computed from all available observations in the local neighborhood $\mathcal{N}_n(\boldsymbol{x})$. Different from the conventional local smoothing approaches, the one-sided local linear kernel estimate is obtained using the observations in one of the two halves of $\mathcal{N}_n(\boldsymbol{x})$. The split of $\mathcal{N}_n(\boldsymbol{x})$ into two halves is made so that at least one of them is asymptotically on one side of the JLCs. To proceed, we first describe the conventional local linear kernel estimate and its error for estimating a jump regression function in order to motivate the needs for the one-sided estimate. In the conventional local linear estimation, the function $m(\boldsymbol{x}_*)$ is locally approximated around a test location $\boldsymbol{x}$ by a linear model $\alpha - \boldsymbol{\beta}^T(\boldsymbol{x}_* - \boldsymbol{x})$, where α is the intercept and $\boldsymbol{\beta} \in \mathbb{R}^p$ is the slope. The slope and intercept are estimated so that the locally weighted sum of the residual squares is minimized,

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}}) = \arg\min_{\alpha, \boldsymbol{\beta}} \sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \left[Y_i - \alpha - \boldsymbol{\beta}^T(\boldsymbol{x}_i - \boldsymbol{x})\right]^2 K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n(\boldsymbol{x})}\right), \tag{3}$$

where $K(\boldsymbol{u})$ is an isotropic kernel function with a unit-circle support $\{\boldsymbol{u} \in \mathbb{R}^p : \boldsymbol{u}^T\boldsymbol{u} \leq 1\}$. Then, the local estimate of $m(\boldsymbol{x}_*)$ at the location $\boldsymbol{x}_* = \boldsymbol{x}$ is $\hat{\alpha} - \hat{\boldsymbol{\beta}}^T(\boldsymbol{x} - \boldsymbol{x})$, which is simply $\hat{\alpha}$. Let $\hat{m}_{(0)}(\boldsymbol{x})$ denote the conventional local linear estimate of $m(\boldsymbol{x})$.

The following theorem gives the asymptotic bias and variance of the estimate:

Theorem 3.1. *Assume that $g(\boldsymbol{x}) \in C^2(\mathcal{X})$ has a bounded second-order derivative, the kernel K is a Lipschitz-1 continuous and isotropic density function, and $h_n(\boldsymbol{x})$ follows (2). For a given point $\boldsymbol{x} \in \mathcal{A}_b$, if the projection of the point to the boundary set $\mathcal{B}$ is $\boldsymbol{x}_J$ and it is non-singular, i.e., $\boldsymbol{x}_J \in \mathcal{B}\backslash\mathcal{S}$, then there exists a unique pair of b and b' such that $\boldsymbol{x}_J \in \partial\mathcal{A}_b \cap \partial\mathcal{A}_{b'}$, and*

$$E[\hat{m}_{(0)}(\boldsymbol{x})] - m(\boldsymbol{x}) = o_P\left(\frac{1}{n^{2/p} f(x)^{2/p}}\right) + (c_J + o_P(1)) \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) d\boldsymbol{u}, \tag{4}$$

and

$$Var[\hat{m}_{(0)}(\boldsymbol{x}) | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] = \kappa_1 \sigma^2 (1 + o_P(1)), \tag{5}$$

where $c_J = \delta_{b,b'}(\boldsymbol{x}_J)$ is the jump magnitude at $\boldsymbol{x}_J$, κ_1 is a constant depending on the kernel function, and $Q^{(b')}$ is the part of the kernel support that corresponds to $\mathcal{A}_{b'} \cap \mathcal{N}_n(\boldsymbol{x})$.

The proof of the theorem is provided in the online supplementary material (Appendix A). From (4) and (5), the variance of the estimate is asymptotically a constant. The bias is significantly affected by $d(\boldsymbol{x}, \boldsymbol{x}_J)$, the distance of the test point $\boldsymbol{x}$ to the nearest jump location curve. Please note that if $d(\boldsymbol{x}, \boldsymbol{x}_J) \geq h_n(\boldsymbol{x})$, i.e., the test point is far away from the jump location curve, then $\mathcal{A}_{b'} \cap \mathcal{N}_n(\boldsymbol{x}) = \emptyset$ and consequently $\mathcal{Q}^{(b')}$ is an empty set. In such a case, the bias is simply $o_P\left(\frac{1}{n^{2/p} f(x)^{2/p}}\right)$, which is the same as the bias of the conventional local linear kernel estimate in a continuity region. However, when the distance goes below $h_n(\boldsymbol{x})$, $\mathcal{Q}^{(b')}$ is non-empty, as illustrated in Fig. 2-(a). In such cases, the additional bias, $c_J \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) d\boldsymbol{u}$, is generated. The additional bias is bounded above by

$$c_J \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) d\boldsymbol{u} \leq c_J K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_J}{h_n(\boldsymbol{x})}\right) \mathcal{L}(\mathcal{Q}^{(b')}), \tag{6}$$

where $\mathcal{L}(\cdot)$ is the Lebesgue measure, and $\mathcal{L}(\mathcal{Q}^{(b')})$ is

$$O_p\left(\max\left\{0, 1 - \left(\frac{d(\boldsymbol{x}_J, \boldsymbol{x})}{h_n(\boldsymbol{x})}\right)^p\right\}\right).$$

Therefore, this part of the bias increases as $d(\boldsymbol{x}, \boldsymbol{x}_J)/h_n(\boldsymbol{x})$ decreases, i.e., the test point approaches to the JLC.

To mitigate the bias increment, the local neighborhood $\mathcal{N}_n(\boldsymbol{x})$ is halved into $\mathcal{N}_n^{(1)}(\boldsymbol{x})$ and $\mathcal{N}_n^{(2)}(\boldsymbol{x})$, by a plane passing through $\boldsymbol{x}$ and being perpendicular to $\hat{\boldsymbol{\beta}}_{(0)}$, as illustrated in Fig. 2-(b), where $\hat{\boldsymbol{\beta}}_{(0)}$ is the solution to $\boldsymbol{\beta}$ in the conventional local linear estimation (3). According to Corollary 1 in Qiu (2009), $\hat{\boldsymbol{\beta}}_{(0)}$ is approximately perpendicular to the tangent plane of the jump location curve at $\boldsymbol{x}_J$ with some approximation error. Therefore, the cutting plane is approximately in parallel to the tangent plane of the jump location curve, and either one of the two halves would be approximately on one side of the jump location curve. For example, in Fig. 2-(b), the test point $\boldsymbol{x}$ is in $\mathcal{A}_b$, and $\mathcal{N}_n^{(1)}(\boldsymbol{x})$ mostly belongs to $\mathcal{A}_b$ except for its small portion that corresponds to $\mathcal{Q}^{(1b')}$ in the figure.

In each one-sided neighborhood $\mathcal{N}_n^{(l)}(\boldsymbol{x})$, for $l = 1, 2$, we take the one-sided local linear kernel estimate of m, denoted as $\hat{m}_{(l)}(\boldsymbol{x})$, to be the solution of α to the following optimization

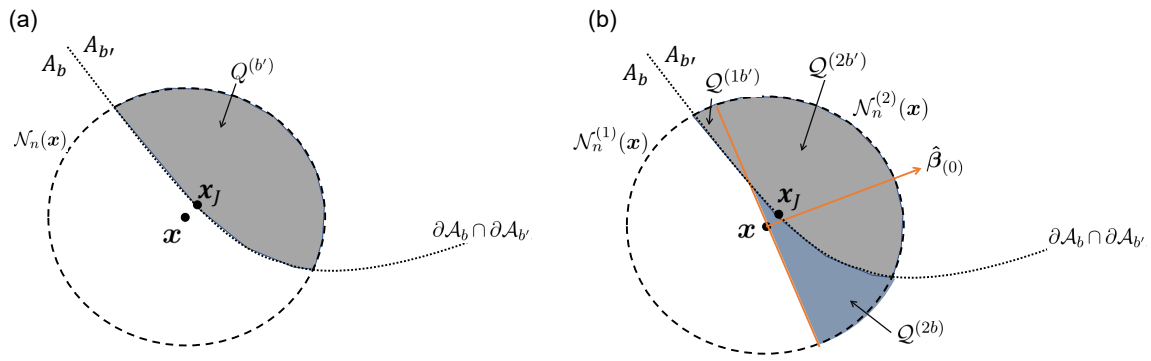

Figure 2: (a) Conventional local linear kernel estimate uses observations in a local neighborhood $\mathcal{N}_n(\boldsymbol{x})$, (b) one-sided local linear kernel estimate uses observations in one of the two halves of $\mathcal{N}_n(\boldsymbol{x})$. In the figure, $Q^{(\ell b)} = A_b \cap N_n^{(\ell)}(\boldsymbol{x})$ and $Q^{(\ell b')} = A_{b'} \cap N_n^{(\ell)}(\boldsymbol{x})$ for $n = 1, 2$.

problem,

$$(\hat{m}_{(l)}(\boldsymbol{x}), \hat{\boldsymbol{\beta}}_{(l)}(\boldsymbol{x})) = \arg\min_{\alpha,\boldsymbol{\beta}} \sum_{\boldsymbol{x}_i \in \mathcal{N}_n^{(l)}(\boldsymbol{x})} \left[Y_i - \alpha - \boldsymbol{\beta}^T(\boldsymbol{x}_i - \boldsymbol{x})\right]^2 K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n(\boldsymbol{x})}\right). \tag{7}$$

The final estimate of $m(\boldsymbol{x})$ is chosen to be one of $\hat{m}_{(1)}(\boldsymbol{x})$ and $\hat{m}_{(2)}(\boldsymbol{x})$, and the choice depends on their estimation errors. The bias and variance of the two one-sided estimates are given in Theorem 3.2. The proof of Theorem 3.2 is similar to that of Theorem 3.1.

Theorem 3.2. *Under the same conditions stated in Theorem 3.1, we have*

$$\begin{aligned} E[\hat{m}_{(1)}(\boldsymbol{x})] - m(\boldsymbol{x}) =& o_P\left(\frac{1}{n^{2/p} f(x)^{2/p}}\right) \\ &+ (2c_J + o_P(1)) \int_{\mathcal{Q}^{(1b')}} K(\boldsymbol{u}) d\boldsymbol{u}, \\ E[\hat{m}_{(2)}(\boldsymbol{x})] - m(\boldsymbol{x}) =& o_P\left(\frac{1}{n^{2/p} f(x)^{2/p}}\right) \\ &+ (-2c_J + o_P(1)) \int_{\mathcal{Q}^{(2b')}} K(\boldsymbol{u}) d\boldsymbol{u}, \end{aligned} \tag{8}$$

and

$$Var[\hat{m}_{(l)}(\boldsymbol{x}) | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] = 2\kappa_1 \sigma^2 (1 + o_P(1)), \tag{9}$$

where $Q^{(\ell b')}$ is the part of the kernel support that corresponds to $\mathcal{A}_{b'} \cap \mathcal{N}_n^{(l)}(\boldsymbol{x})$.

By the above theorem, the variances of the two one-sided estimates are asymptotically the same. Therefore, the mean squared errors of the estimates are largely influenced by their respective bias terms. The major parts of the asymptotic biases are $2c_J \int_{\mathcal{Q}^{(1b')}} K(\boldsymbol{u}) d\boldsymbol{u}$ and

$2c_J \int_{\mathcal{Q}^{(2b')}} K(\boldsymbol{u})d\boldsymbol{u}$. Since $\mathcal{Q}^{(1b')} \cup \mathcal{Q}^{(2b')} = \mathcal{Q}^{(b')}$, the two terms can be written as

$$2c_J \int_{\mathcal{Q}^{(1b')}} K(\boldsymbol{u})d\boldsymbol{u} = a_1 c_J \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u})d\boldsymbol{u} \text{ and}$$
$$2c_J \int_{\mathcal{Q}^{(2b')}} K(\boldsymbol{u})d\boldsymbol{u} = (2-a_1) c_J \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u})d\boldsymbol{u},$$

for a constant $a_1 \in [0,2]$. The smaller value of the two terms is bounded above by

$$\begin{aligned} &2c_J \min\left\{\int_{\mathcal{Q}^{(1b')}} K(\boldsymbol{u})d\boldsymbol{u}, \int_{\mathcal{Q}^{(2b')}} K(\boldsymbol{u})d\boldsymbol{u}\right\} \\ &\quad \leq c_J K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_J}{h_n(\boldsymbol{x})}\right) O_P\left(\max\left\{0, 1-\left(\frac{d(\boldsymbol{x}_J,\boldsymbol{x})}{h_n(\boldsymbol{x})}\right)^p\right\}\right) \min\{a_1, 2-a_1\}. \end{aligned} \tag{10}$$

The last term, $\min\{a_1, 2-a_1\}$, depends only on $\hat{\boldsymbol{\beta}}_{(0)}$. When $\hat{\boldsymbol{\beta}}_{(0)}$ is along the tangent plane at $\boldsymbol{x}_J$, the value of $\min\{a_1, 2-a_1\}$ is approximately at its maximum, one. When the direction of $\hat{\boldsymbol{\beta}}_{(0)}$ is perpendicular to the tangent plane, this value is zero. As mentioned earlier, it has been confirmed that $\hat{\boldsymbol{\beta}}_{(0)}$ is asymptotically perpendicular to the tangent plane (Qiu, 2009). Thus, $\min\{a_1, 2-a_1\}$ is approximately zero.

The bias terms cannot be numerically evaluated since $\mathcal{Q}^{(1b')}$ and $\mathcal{Q}^{(2b')}$ are unknown. To make a choice between $\hat{m}_{(1)}(\boldsymbol{x})$ and $\hat{m}_{(2)}(\boldsymbol{x})$, the following weighted residual mean errors are considered:

$$err^{(l)}(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{x}_i \in \mathcal{N}_n^{(l)}(\boldsymbol{x})} \left[Y_i - \hat{m}_{(l)}(\boldsymbol{x}) - \hat{\boldsymbol{\beta}}_{(l)}(\boldsymbol{x})^T(\boldsymbol{x}_i - \boldsymbol{x})\right]^2 K\left(\frac{\boldsymbol{x}_i-\boldsymbol{x}}{h_n(\boldsymbol{x})}\right)}{\sum_{\boldsymbol{x}_i \in \mathcal{N}_n^{(l)}(\boldsymbol{x})} K\left(\frac{\boldsymbol{x}_i-\boldsymbol{x}}{h_n(\boldsymbol{x})}\right)}.$$

When $err^{(1)}(\boldsymbol{x}) < err^{(2)}(\boldsymbol{x})$, $\hat{m}_{(1)}(\boldsymbol{x})$ is chosen; and $\hat{m}_{(2)}(\boldsymbol{x})$ is chosen otherwise.

3.2 Proposed Method for Sequential Design Selection

In this section, we describe our proposed method for a sequential selection of design points, which selects the design points in multiple stages. The first stage serves as a seed stage, and the design points in the first stage are randomly sampled from an uniform distribution or selected by the Latin hypercube sampling (LHS). In all of our numerical examples, we used LHS in the first stage. Each of the subsequence stages is described as follows. Suppose that there are n design points selected up to the previous stage, and we describe how b additional design points are selected in the current stage. Let f_1 denote the unknown density of the n design points from the previous stages, and let $f_{2|1}$ represent the sampling density used to select the b design points in the current stage. If f was a 'desirable' joint density of the n design points and the b additional design points, then the sampling density for the current stage's design points should be the following conditional density:

$$f_{2|1}(\boldsymbol{x}) = \frac{f(\boldsymbol{x})}{f_1(\boldsymbol{x})}. \tag{11}$$

Intuitively, f should be chosen to minimize the integrated square loss,

$$\int_{\boldsymbol{x}\in\mathcal{X}} E[m(\boldsymbol{x}) - \hat{m}(\boldsymbol{x})]^2 d\boldsymbol{x}.$$

Note that the square loss $E[m(\boldsymbol{x}) - \hat{m}(\boldsymbol{x})]^2$ can be decomposed into the squared-bias and the variance of $\hat{m}(\boldsymbol{x})$. Based on Theorem 3.2, the variance of the jump regression estimate defined in Section 3.1 is approximately a constant, and the square loss is largely influenced by the squared-bias term. We will develop our sequential design strategy to balance the bias and the integrated square loss function. To develop the idea more formally, please note that the bias can be as small as $o_P\left(\frac{1}{n^{2/p}f(x)^{2/p}}\right)$ when the test location is far away from the jump location curve in the sense that $d(\boldsymbol{x}, \boldsymbol{x}_J) \geq h_n(\boldsymbol{x})$. If $d(\boldsymbol{x}, \boldsymbol{x}_J) < h_n(\boldsymbol{x})$, there is an additional bias of the size $2c_J \min\left\{\int_{\mathcal{Q}^{(1b')}} K(\boldsymbol{u})d\boldsymbol{u}, \int_{\mathcal{Q}^{(2b')}} K(\boldsymbol{u})d\boldsymbol{u}\right\}$. By (10), this part of the bias is bounded above by

$$c_J K\left(\frac{d(\boldsymbol{x}_J, \boldsymbol{x})}{h_n(\boldsymbol{x})}\right) O_P\left(\max\left\{0, 1 - \left(\frac{d(\boldsymbol{x}_J, \boldsymbol{x})}{h_n(\boldsymbol{x})}\right)^p\right\}\right) \min\{a_1, 2 - a_1\},$$

which goes to zero as $\frac{d(\boldsymbol{x}_J, \boldsymbol{x})}{h_n \boldsymbol{x}}$ increases or $d(\boldsymbol{x}_J, \boldsymbol{x})f(\boldsymbol{x})$ increases. To balance the bias over $\boldsymbol{x}$ and minimize the integrated square loss function, the desirable sampling density should be

$$f(\boldsymbol{x}) \propto \frac{1}{d(\boldsymbol{x}_J, \boldsymbol{x})}.$$

We hope that collectively the $n + b$ design points have higher densities at places near the jump location curve (i.e., places with small $d(\boldsymbol{x}_J, \boldsymbol{x})$). Certainly, we do not know where the jump location curves are located in practice, so we do not know the distance $d(\boldsymbol{x}_J, \boldsymbol{x})$. But, the distance can be roughly located using the observations of the regression function at the n design points selected in the previous stages. It is easy to show that the following statistic increases as $d(\boldsymbol{x}, \boldsymbol{x}_J)$ decreases,

$$[\hat{m}_{(1)}(\boldsymbol{x}) - \hat{m}_{(2)}(\boldsymbol{x})]^2, \tag{12}$$

so we use it as a jump detection statistic. Based on the jump detection statistic, we propose a desirable joint density f to be

$$f(\boldsymbol{x}) = C \exp\left\{\gamma[\hat{m}_{(1)}(\boldsymbol{x}) - \hat{m}_{(2)}(\boldsymbol{x})]^2\right\}, \boldsymbol{x} \in \mathcal{X}, \tag{13}$$

where $C > 0$ is a normalization constant, and the coefficient γ controls the exploration vs exploitation trade-off. We chose $\gamma = 1/\sigma^2$, where σ^2 is the noise variance. With that choice, the quantity $\gamma(\hat{m}_{(1)} - \hat{m}_{(2)})^2$ is approximately the square of the jump magnitude relative to the noise variance. For a larger σ^2, this sampling function seeks more exploration, and for a smaller σ^2, more exploitation is sought. The noise standard deviation σ is estimated using the median absolute deviation (MAD).

Because $\mathcal{X}$ is bounded and the estimates $\hat{m}_{(l)}$ are bounded, C is well defined. From (11), the sampling density for the b new design points should be

$$f_{2|1}(\boldsymbol{x}) = \frac{f(\boldsymbol{x})}{f_1(\boldsymbol{x})} \approx \frac{C \exp\left\{(\hat{m}_{(1)}(\boldsymbol{x}) - \hat{m}_{(2)}(\boldsymbol{x}))^2\right\}}{\frac{1}{n}\sum_{i=1}^n K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right)}, \tag{14}$$

where the approximation comes from the standard kernel density estimation of $f_1(\boldsymbol{x})$, and h is a non-spatial adaptive kernel bandwidth parameter that depends on the sample size n. Sampling from the complex density (14) can be performed by the Metropolis-Hasting

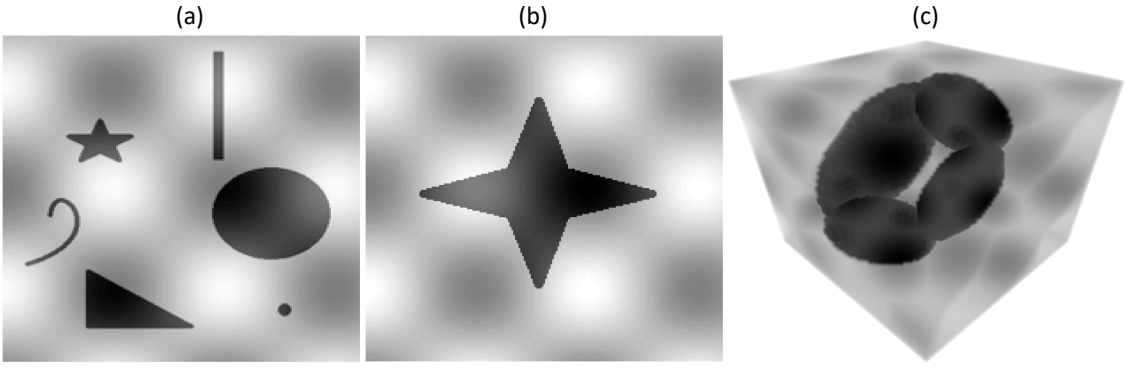

Figure 3: Three synthetic datasets. (a) 2d-others, (b) 2d-star, (c) 3d-donut

Algorithm. For more computational feasibility, we can limit the sampling locations to the ones sampled from the uniform distribution over the regression domain. For each of the possible sampling locations, we can compute $f_{2|1}(\boldsymbol{x})$ up to a normalizing constant. The computed values are normalized so that the summation of all the computed values is equal to one. The normalized values will serve as the probability mass function (pmf) defined on a finite number of the possible sampling locations, and then b samples will be drawn randomly from that pmf as the b new design points for the next stage.

4 Simulation Study: 2D and 3D Domains

For the initial validation of the proposed method, we performed a simulation study with three synthetic datasets. Fig. 3 shows the underlying noise-free regression functions for the first two synthetic datasets defined on the 2D domain $[0, 200]^2$, and the regression function for the third dataset defined on the 3D domain $[0, 50]^3$. The underlying noise-free regression functions are in the mixture form,

$$m(\boldsymbol{x}) = g_0(\boldsymbol{x}) - 0.3I_{\mathcal{A}_b}(\boldsymbol{x}),$$

where g_0 is continuous on $\mathcal{X}$, and $\mathcal{A}_b \subset \mathcal{X}$ represents the subregion with a different intensity level. Then, the regression function is continuous except at the boundary $\partial\mathcal{A}_b \subset \mathcal{X}$. For the first two datasets,

$$g_0(\boldsymbol{x}) = \sin\left(\frac{x_1}{20}\right) \times \cos\left(\frac{x_2}{20}\right),$$

where x_1 and x_2 are the first and second elements of the input vector $\boldsymbol{x}$ respectively. For the third dataset, we used

$$g_0(\boldsymbol{x}) = \sin\left(\frac{x_1}{5}\right) \times \cos\left(\frac{x_2}{5}\right) \times \sin\left(\frac{x_3}{5}\right),$$

where x_1, x_2 and x_3 are the first, second and third elements of $\boldsymbol{x}$ respectively. In Fig. 3, the set $\mathcal{A}_b$ is shown as the dark regions. We then added i.i.d. Gaussian noise from $\mathcal{N}(0, \sigma^2)$ to $m(\boldsymbol{x})$ to obtain observed data.

For each dataset, n design points in total will be selected, using our proposed sequential adaptive approach described in Section 3.2. We varied the number n and the number of

the design points chosen in each stage, denoted by b. Since the domain sizes differ in the three datasets, which are 200 x 200 in the first two datasets and 50 x 50 x 50 in the last dataset, the experimental settings are denoted in terms of the percents of the domain sizes. The number n varies over 2.5%, 3,75%, 5.00%, 6.25%, 7.50%, 8.75% and 10% of the domain size. The number b varies over 0.125%, 0.25%, 0.625%, and 1.25% of the domain size. We also varied the noise level σ over $0.1, 0.2, 0.4, 0.6, 0.8, 1$ to study different signal-to-noise (SNR) cases. In the case of $\sigma = 1$, the noise level is equal to the maximum signal intensity. Combining the three parameter values, we have a total of 168 different simulation scenarios, and each scenario is run for 20 replications. Fig. 4 illustrates the selected design points obtained in the case with $n = 10\%$ and $b = 1.25\%$. The design points selected for the first stage are seed locations, selected by the Latin Hypercube Sampling (LHS), and the design points selected in the subsequent stages are more concentrated around the jump boundaries and some intensity transitioning areas.

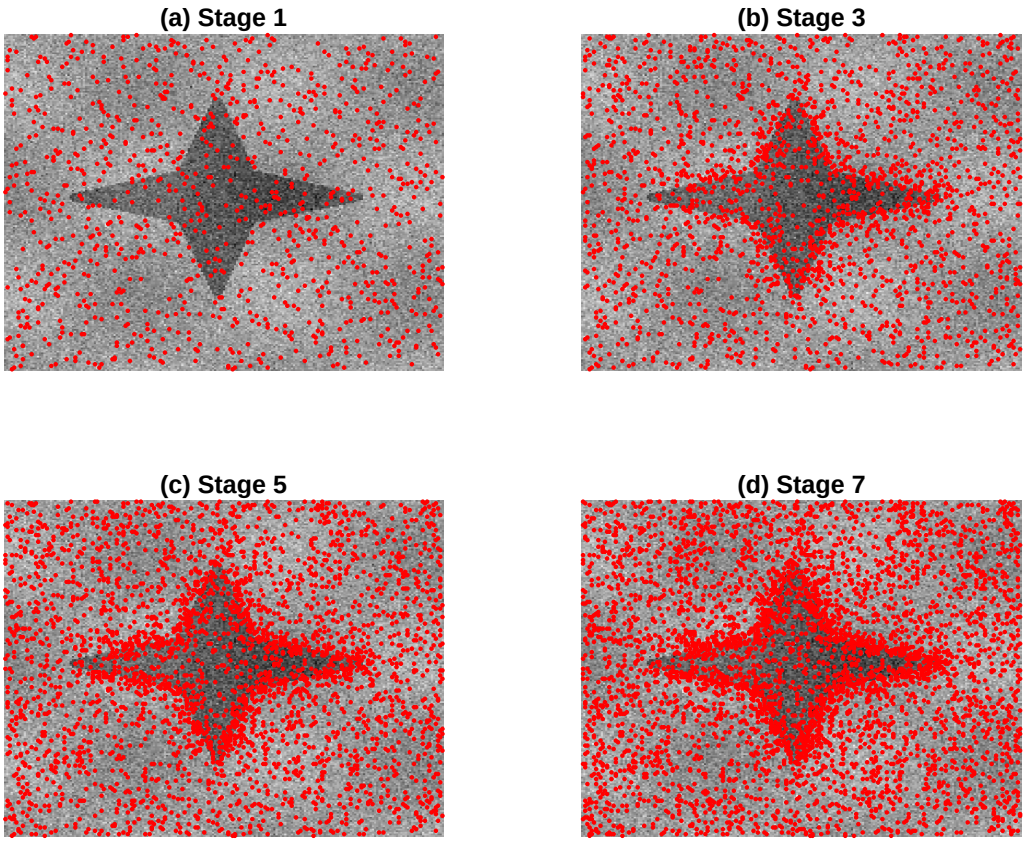

Figure 4: Illustration of the selecting design points in different stages. Each plot shows all the design points selected up to the specified stage.

After all the design points are selected, the noisy observations of the regression function at the design points serve as training data to estimate the regression function. Test locations are randomly sampled from an uniform distribution over the regression domain, excluding the ones overlapped with the training data, and the estimate of the regression function

is computed for each of the the test locations, using the procedure in Section 3.1. The estimates were compared to the corresponding true regression function values (serving as the ground truth) at the test locations to evaluate the mean square errors. We used two mean square error (MSE) metrics: MSE near jump location curves and MSE in the continuity regions, defined to be

$$\text{J-MSE} = \frac{1}{|JB(h)|} \sum_{(x,y)\in JB(h)} (\hat{m}(\boldsymbol{x}) - m(\boldsymbol{x}))^2$$
$$\text{C-MSE} = \frac{1}{|JB(h)^c|} \sum_{(x,y)\in JB(h)^c} (\hat{m}(\boldsymbol{x}) - m(\boldsymbol{x}))^2,$$

where $\hat{m}(\boldsymbol{x})$ is the jump regression estimate, $JB(h)$ is the set of the test locations whose distance from the closest jump location curve is less than or equal to h, and $JB(h)^c$ is the complement of $JB(h)$; h is fixed to be 6, which is about twice of the average distance between two neighboring pixels.

4.1 Effect of the tuning parameters, n, b and σ^2

We first evaluate how the proposed approach performs under various experimental settings. Fig. 5 shows the changes in J-MSE and C-MSE for different settings specified by the total number of the selected design points, denoted by n, and the number of the design points selected per stage, denoted by b. The per-stage selection size b determines the number of stages for a fixed n. According to Fig. 5, the per-stage selection size is not the major factor that affects J-MSE and C-MSE. For the first two test datasets with 2D domains, the per-stage selection size does not make any significant difference in both J-MSE and C-MSE. For the last test dataset with 3D domain, the J-MSE tends to be lower for a smaller b and the C-MSE tends to be lower for a larger b.

Both accuracy measures are more significantly affected by n. Based on the results, we would recommend to set n to meet a required level of accuracy and choose a large b for a computational gain. The number of the stages to get n design points is proportional to n/b. If b is too small, many stages would be needed, and more frequent computations to update the sampling density function are needed. Therefore, the total computation time would increase. For the remainder of our numerical experiments, we will use $b = 1.25\%$, the largest value we tried. We also look at the two performance measures for different noise levels of the observed data. We can see clear downward trends in both J-MSE and C-MSE as the noise level decreases or SNR increases. More details can be found in the online supplementary material (Appendix B).

4.2 Comparison with four benchmarks

We compared the performance of the proposed approach with four benchmark methods. The proposed approach is denoted as JuMp Planner (JMP). The first benchmark is randomly sampling from a uniform density (RAND), the second approach is sampling with Latin Hypercube Sampling (LHS). The third approach is sampling from a density proportional to the weighted residual mean square (WRMS) error of the conventional local linear

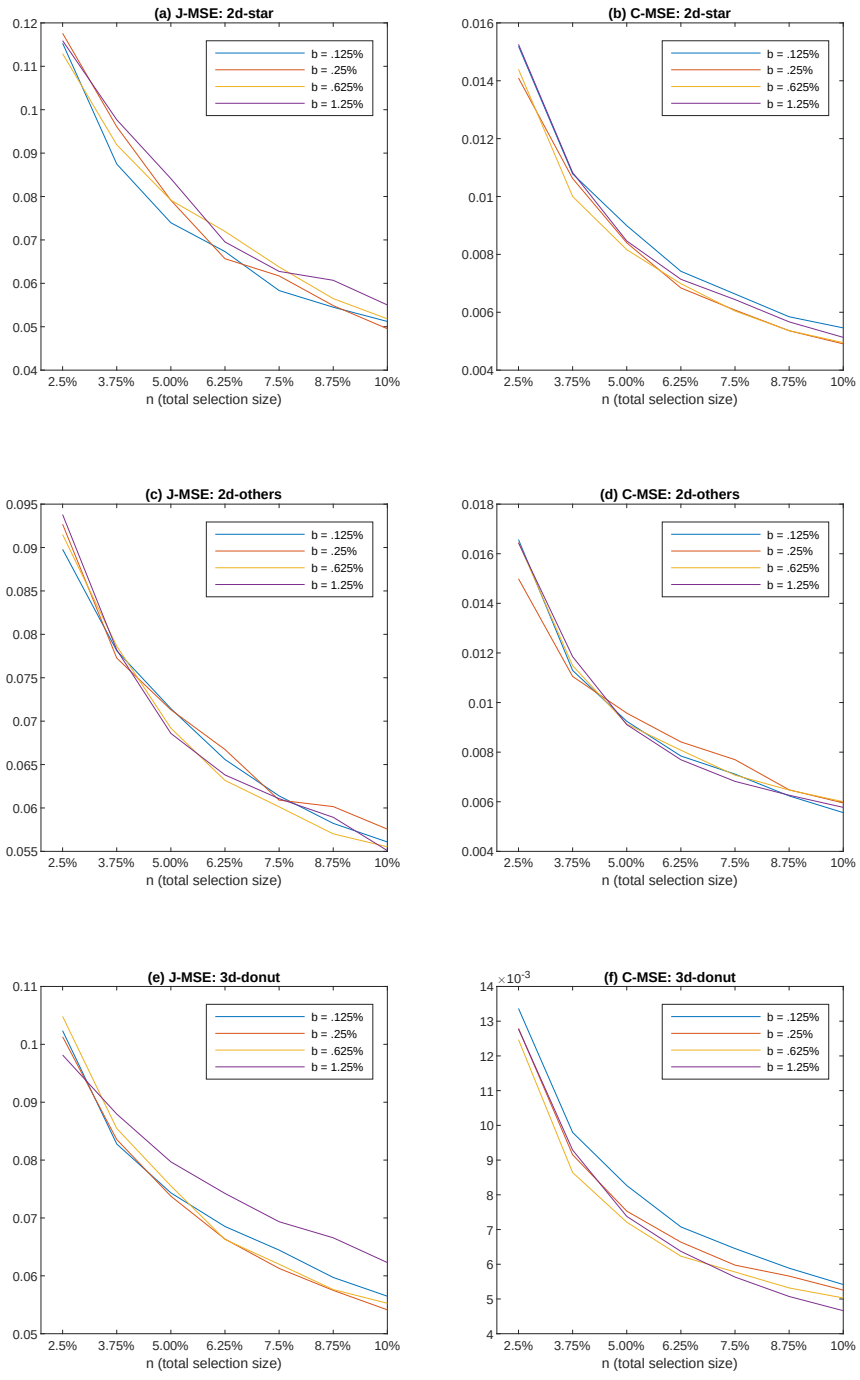

Figure 5: Effect of the total selection size (n) and the per-stage selection size (b). Here we plot the results for $\sigma = 0.4$, because the results for other noise levels follow similar patterns.

kernel smoother,

$$\text{WRMS-C}(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \left[Y_i - \hat{\alpha} - \hat{\boldsymbol{\beta}}^T (\boldsymbol{x}_i - \boldsymbol{x}) \right]^2 K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n(\boldsymbol{x})} \right)}{\sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n(\boldsymbol{x})} \right)},$$

where $\hat{\alpha}$ and $\hat{\boldsymbol{\beta}}$ are the optimal solutions of problem (3). The last benchmark is sampling from a density proportional to the WRMS error of the jump regression model (Qiu, 2004),

$$\text{WRMS-J}(\boldsymbol{x}) = \min\{err^{(1)}(\boldsymbol{x}), err^{(2)}(\boldsymbol{x})\}.$$

In this comparison, we fixed $b = 1.25\%$ and $n = 10\%$, because different choices of b and n did not make much difference of the comparison results. Fig. 6 shows SNR versus the averages of the two MSE metrics over 20 replicated simulation runs. From the figure, it can be seen that the C-MSE values computed over the continuity regions do not dependent on the choice of the design selection method. However, the J-MSE values computed near the jump locations differ significantly among different methods. The major findings regarding J-MSE are summarized below.

- Low Noise Case, $\sigma = 0.1$ or SNR $= 2$: The three error-based methods, JMP, WRMS-J and WRMS-C, significantly outperformed the two random sampling methods, RAND and LHS.
- High Noise Case, $\sigma = 1$ or SNR $= 0$ (i.e. maximum intensity of the regression function is equal to σ): All methods are comparable, which is not surprising. When the noise level is comparable to the maximum signal intensity, the error-guided methods cannot distinguish signals from noise well. In such cases, the three adaptive selection strategies work similarly to the two random sampling methods.
- Medium Noise Cases, $0.1 < \sigma \leq 0.8$: The methods JMP and WRMS-J outperform the method WRMS-C. Namely, the two jump regression based approaches are superior to the conventional local smoothing approach around the jump regions. Furthermore, the method JMP is better than WRMS-J in the first two examples with 2D data, and comparable to WRMS-J in the third example with 3D data.

We also present the variabilities of the two MSE metrics based on 20 replicated simulation runs. Fig. 7shows the standard deviations of J-MSE and C-MSE values at different noise levels. The overall variabilities of the compared methods increase as σ increases or SNR decreases. Among the five methods, the proposed approach has the lowest variability in most cases considered. Thus, a low variability is another advantage of the proposed approach.

4.3 Comparison with the existing adaptive sampling strategies

In this part, we compare the proposed method with two existing adaptive design strategies for treed regression models, including the Taddy's dynamic tree model (Taddy et al., 2011, dynaTree) and the Bayesian Treed GP (Gramacy and Lee, 2009, btgp). For this comparison, we used R-libraries `dynaTree` and `tgp`. The computation for the dynaTree and btgp would be too expensive to afford for a large size problem. For example, when they are applied to

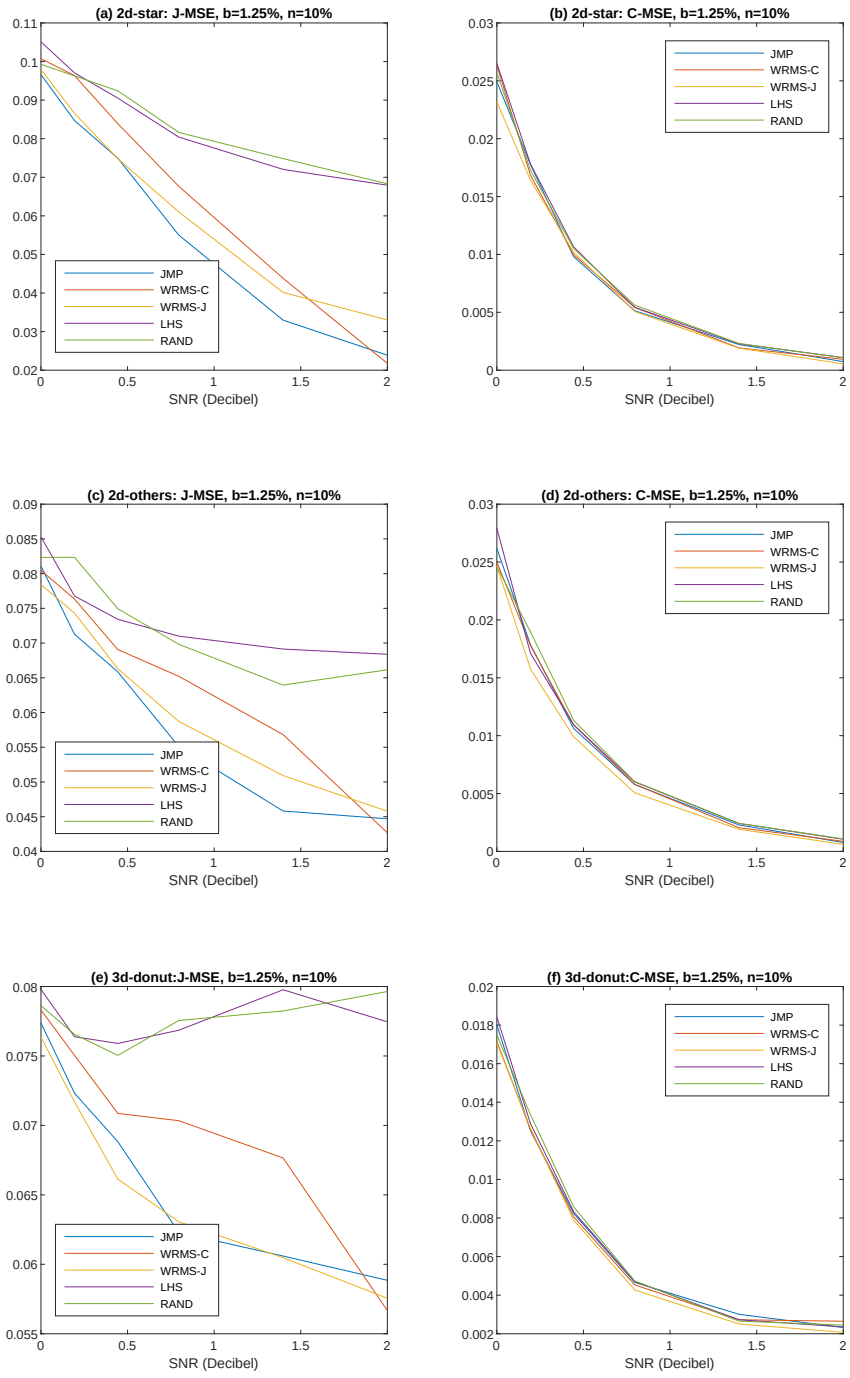

Figure 6: Averaged performance metrics over 20 replicated simulation.

the first simulated scenario considered in the previous section, the dynamic tree model took 433.65 seconds to sample 4,000 design points, and the btgp method took 1,505 seconds for the first two stages of samplings alone.

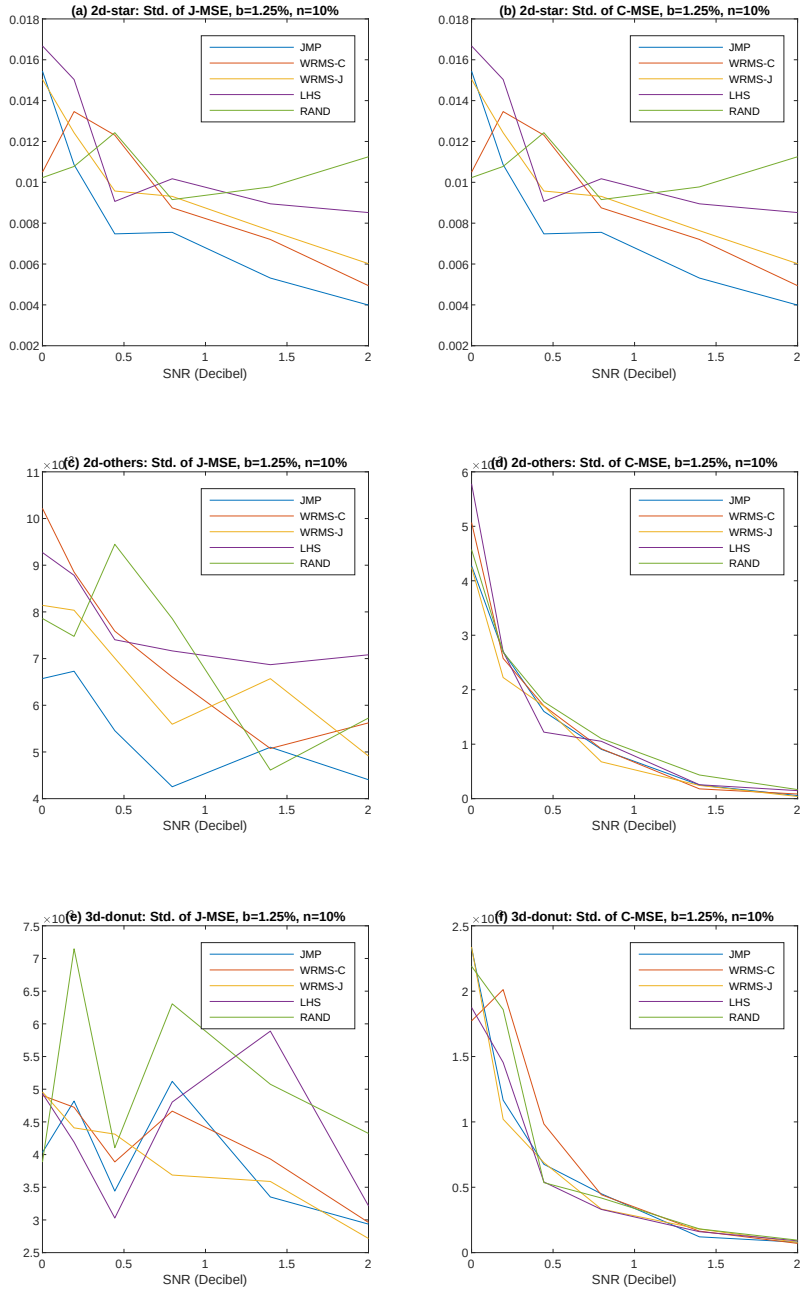

Figure 7: Standard deviations of the two MSE metrics based on 20 replicated simulation runs.

Considering the heavy computation involved in these existing approaches, we consider cases with a small domain size and a small number of sampling size to make a comparison. In the toy example, the true regression function is defined in the small 2D domain $[0, 2]^2$ in the mixture form,

$$m(\boldsymbol{x}) = g_0(\boldsymbol{x}) + 3I_{\mathcal{A}_b}(\boldsymbol{x}),$$

where $g_0(\boldsymbol{x}) = \sin\left(\frac{x_1}{0.4}\right) \times \cos\left(\frac{x_2}{0.4}\right)$ is a continuous function, and $\mathcal{A}_b = \{x_2 >= 1/x_1\}$. To estimate the function, the noisy observations of the function at 125 locations are considered, and the noisy observations are generated by adding Gaussian noise from $\mathcal{N}(0, 0.4^2)$ to $m(\boldsymbol{x})$. The 125 design points are selected by our proposed sampling approach, the ALM sampling function with dynaTree, and the ALM sampling function with btgp, respectively. For all three methods, the 50 initial seed locations are selected by the LHS, and the remaining 75 locations are selected sequentially over 75 sampling stages, one per stage.

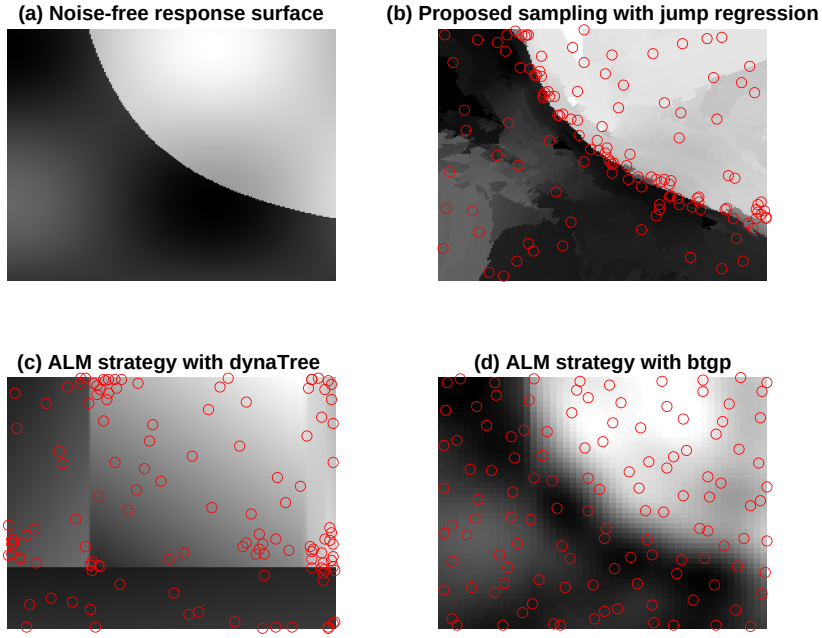

Figure 8: Comparison of the proposed sampling approach with the ALM strategies by the dynaTree (Taddy et al., 2011) and the btgp (Gramacy and Lee, 2009). In panels (b), (c) and (d), the background intensities (grayscale intensities) represent the regression functions estimated by the respective regression approaches with the 125 selected design points (red circles).

Metrics	Proposed Approach	dynaTree with ALM	btgp with ALM
J-MSE	1.0912	2.3457	1.7198
C-MSE	0.1452	0.6426	0.1501
Computing Time (in seconds)	0.40	4.43	18,104.14

Table 1: Quantitative comparison with the two existing adaptive design strategies: the dynamic tree model (dynaTree) and the Bayesian treed regression (btgp).

Figure 8 compares the design points selected by the three adaptive design strategies and also shows the regression functions estimated by the respective regression approaches (i.e., proposed jump regression, dynamic tree model and Bayesian treed GP) with the selected design points. As shown in Figure 8 (b), the proposed approach places more design points around the jump boundary, so the resulting regression estimate is close to the ground truth shown in the panel (a). The dynamic tree model partitions the input domain into five rectangular regions. Please note that the tree model fits a linear leaf model for each sub region. The regression estimate with five linear leaves appears to be too simple to represent the true regression surface with a curvy jump boundary, as shown in Figure 8 (c). The design points selected by this approach are mostly located around the corners of the rectangular sub regions. The btgp model is a more flexible model than the dynamic tree model with GP leaves. However, in this numerical example, the Bayesian tree has only one GP leaf, so one stationary GP model is used for describing the regression function. Therefore, the corresponding regression estimate has quite large bias around the jump boundary, as shown in Figure 8 (d). The design points selected by this approach are similar to the ones selected by LHS.

Table 1 makes a quantitative comparison among the three methods in terms of J-MSE, C-MSE and the computing time. The proposed sampling approach is superior in all three metrics, because it is based on a more flexible regression modeling, obtain better sampling locations, and thus has a better mean square errors in both continuous and discontinuous regions. This study also confirms the computational burden of the tree based approaches, which makes them inappropriate for applications, such as the first motivating example about STEM imaging discussed in Sections 1 and 5.

5 Real Data Study: Adaptive Microscope Imaging For Accelerating the Imaging Speed

Accelerating the STEM imaging speed would open unprecedented opportunities in studying these important material processes. One promising method to accelerate the speed is to scan material specimen in a reduced set of spatial locations. Here we illustrate the use of our proposed sequential adaptive design strategy to select only a targeted, partial set of scan locations. There are two important factors that need to be considered: accuracy of estimating the specimen image at unobserved locations (i.e regression accuracy) and the computation time spent to calculate design selection statistics and sample design points. We do not want the required computation time to surpass the full scan time if one wants to accelerate the total imaging acquisition time. This requirement on computing time makes many Bayesian tree-based sequential design approaches ineligible for this application.

5.1 Application Details

In this application, the total sample size n is set to be 10% of the pixel number of the full scan, considering the accuracy requirement. Based on prior numerical trials, the 10% partial scan can provide a good accuracy for estimating the underlying images. Lowering the sample size could lose many sharp features of the material images, and increasing the sample size would increase the imaging time. In STEM imaging, the time to scan one pixel

is referred to as a pixel dwell time, which is about 10 to 40 microseconds. To achieve good quality pixel measurements, 40 microseconds of the pixel dwell time is applied. The total imaging time is approximately the number of pixels to scan multiplying by the pixel dwell time. For example, scanning a 587×484 imaging area would take $587 \times 484 \times 40\mu s$, equal to 10.9 seconds in total. If we only select 10% of the imaging pixels for a partial scan, then the physical scanning time would be only 1.09 seconds, i.e., 10% of the full scan time.

Selecting design points in multiple stages would require a significant computing time for each stage, because the sampling density needs to be recalculated every stage based on the results in all previous stages. The total computing time increases as the number of stages, M, increases. Therefore, the number M in the sequential design selection scheme should be selected carefully, considering the computing time and the accuracy of the image reconstruction with the selected samples. We first performed some initial experiments to choose the appropriate number for M. Table 2 shows the computing times and reconstruction accuracy for several different values of M when 10% of a 587×484 test image is sub-sampled using the proposed approach:

No. Stages (M)	M=1	M=2	M=3	M=4	M=5	M=10
Computing Time (in seconds)	0.0954	0.9402	1.7832	2.5678	3.4383	7.5855
Reconstruction Error	0.0237	0.0205	0.0197	0.0195	0.0195	0.0194

Table 2: Computing time and reconstruction error versus the number of stages for the first image in Fig. 9 when n = 10% of the full image size.

From the table, it can be seen that the computing time increases and the reconstruction error decreases as M increases. However, the reduction in the reconstruction error is saturated after $M = 4$. We have the similar outcomes for other test images. Therefore, we choose $M = 4$ to balance the accuracy and the computing time in this application. If the number of stages is $M = 4$, the total computing time of the partial scan would include 1.09 seconds of the physical scan time plus 2.5678 seconds of the computing time, which is about 3 times shorter than the time for the full scan. The imaging accelerating factor would be 3 in such a case. We will use $M = 4$ for the remainder of this section, which corresponds to $b = 2.5\%$.

5.2 Applications to STEM Imaging Under Various Different Conditions

To quantitatively evaluate our approach compared to the standard STEM imaging techniques, we first obtained complete imaging scans for eleven different specimens (Fig. 9) to serve as ground truth. These microscope images are characterized by their noise levels. The noise levels of the images are estimated as follows. We first used the jump regression estimates of the images for denoising, and the noise variances are estimated by the mean squared differences of the estimated regression surfaces and the corresponding original images. The eleven images have 587×484, 587×465, 611×474, 592×592, 472×459, 1006×1006, 793×916, 579×579, 505×500, 501×498, and 502×496 pixels, respectively.

In each of the eleven cases, we also achieved the partial scan using our proposed approach.

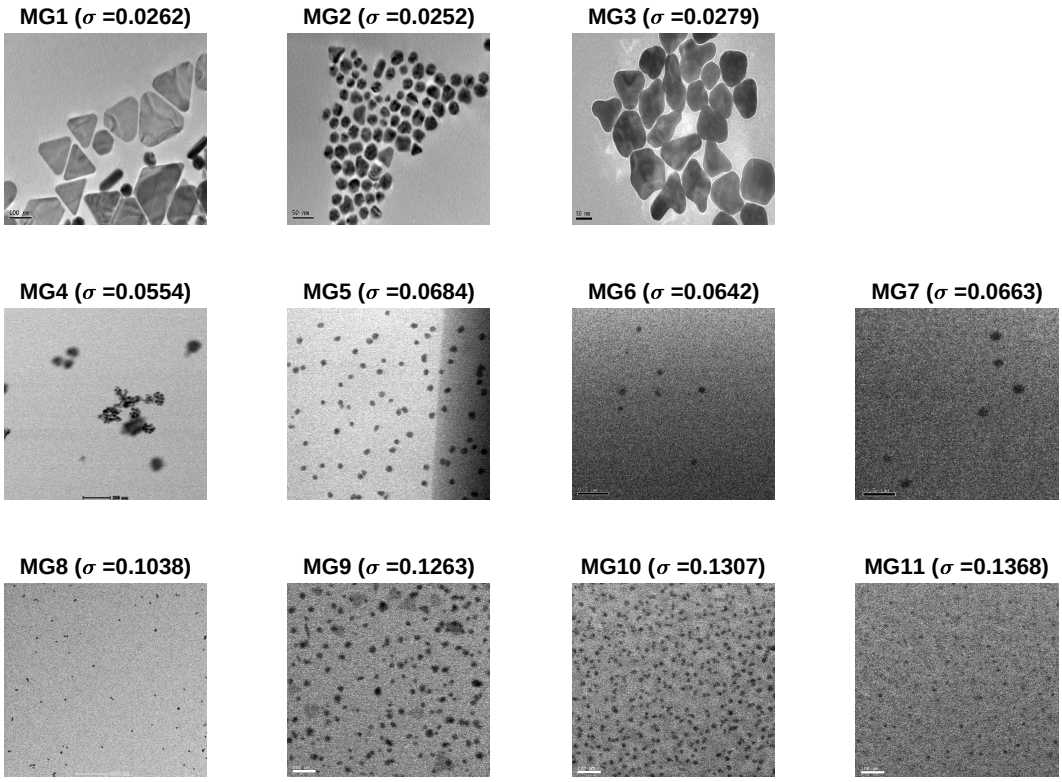

Figure 9: Full raster scanned microscope images. Each image was labeled with image number and σ (in parenthesis), where σ is the noise standard deviation when the image intensity is normalized so that its maximum is 1.

The number n selected is equal to 10% of the total raster location number, and the locations in the subset were selected sequentially in six stages. The subset of the raster locations and the corresponding pixel measurements were used to estimate the pixel measurements at the other unselected raster locations. The estimates were compared to the corresponding values of the full raster image, and the two performance metrics, J-MSE and C-MSE, were computed. The evaluation of J-MSE requires $JB(h)$, which was estimated. We first applied an image segmentation algorithm to identify the outlines of black regions, and the results of the image segmentation algorithm were manually corrected for a better accuracy, and then $JB(h)$ is estimated accordingly. Samplings from the uniform density (RAND), WRMS-C, WRMS-J and LHS were used as benchmarks here.

Fig. 10 shows the comparison of the related methods in terms of the two performance metrics. Fig. 13 shows the design point locations selected by the partial scans as red dots. We used $n = 3\%$ for the illustration because the red dots are too dense to show the results effectively otherwise. A few key findings are summarized below.

- For all test images, the Root C-MSE values for different design selection strategies are comparable and close to the noise level. This is consistent with what we found in the simulation study.

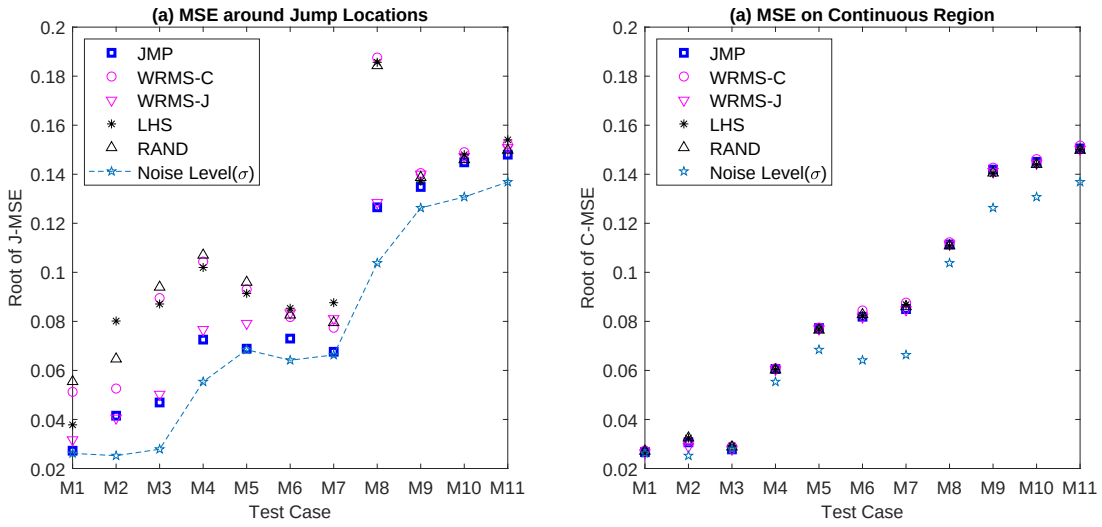

Figure 10: Reconstruction error metrics of different sampling methods for 11 microscope images.

- MG1 through MG4 (Low Noise and High Ratio of Jump Location Boundary Pixels): Sampling from WRMS-J and the proposed design selection strategy are significantly better than the other methods. This is also consistent with the findings from the simulated studies.
- MG5 through MG7 (Medium Noise): The proposed design selection approach is better than all the other methods with significant margins, while sampling from WRMS-C is not much better than Random Sampling.
- MG8 (High Noise and Many Tiny Foregrounds): Sampling from WRMS-J and the proposed design selection strategy are significantly better than sampling from WRMS-C and Random Sampling.
- MG9 through MG11 (Very High Noise): All methods perform similarly. The proposed strategy is based on the jump detection statistic, which is almost uniform when the noise level is comparable to the jump size c_J, so the strategy becomes similar to the uniform sampling strategy as shown in Figure 13 of the online supplementary material (Appendix C).

In summary, the proposed approach is very promising in compressive and adaptive imaging for accelerating the image scan speed in STEM unless the level of image noise is comparable to the intensity jumps at edges.

6 Real Data Study: Experimental Campaign for Predicting Carbon Nanotube Growth

This section presents the application of the proposed approach to another motivating example of this paper, a problem of optimizing an experimental campaign for predicting the

response variable of a chemical experiment under a given experimental condition when the response jumps around certain characteristic boundaries.

6.1 Scientific Background and Significance

We use a chemical experiment of carbon nanotube growth as a motivating example. Carbon nanotubes are tubes made of carbon atoms with nano-scale diameters. The nanotubes exhibit exceptional tensile strengths and great thermal/electricity conductivity, which are being applied for many practical applications. They are chemically synthesized using a chemical vapor deposition (CVD) process. We are interested in understanding how the reaction conditions of the chemical process affect the carbon nanotube growth. The dependent variable of interest (i.e. the reponse variable) is the resultant amount of carbon nanotube growth under a given reaction condition, and the input variables describe the reaction condition. Among many process parameters describing the reaction condition, the reaction temperature and the composition of chemical reactants greatly affect the growth outcomes, which are the two experimental inputs. The first chemical reactant is C_2H_4, which is a catalyst to promote the growth reaction. The second reactant is CO_2, which suppress the growth reaction. When the concentration ratio of the two chemicals is below a certain threshold, the amount of the carbon nanotube growth is flattened to almost a zero level, but the amount suddenly jumps to a certain level right above the threshold; the observed jump behaviors in the closed-loop carbon nanotube (CNT) growth are a direct result of catalyst phase transition, and the underlying physics is discussed in greater detail in our upcoming publication (Carpena-Núñez et al., 2020).

Estimating the response surface embedding jumps would require a significant number of experiments if one uses an uniform design such as a space filling design, mainly due to the presence of sharp jumps in the response surface and locating the sharp jumps precisely is possible only when the design points are uniformly dense over the design space unless a non-uniform design is adopted. From a past experimental campaign of the same kind done at Air Force Research Lab (AFRL), about 70 design points were selected manually by a human operator, and many of the design points were located in not much useful zero-flat growth regions which yielded a rough estimation of the underlying responses. To make the experimental campaign more efficient, we applied our proposed design selection strategy to select an experimental design for estimating the response surface with increased fidelity in locating the jump structures. The accurate estimation of the response surface and the embedded jumps would guide practitioners to design their CVD processes for good carbon nanotube yields.

6.2 Application Details

The sequential process is implemented by AFRL using a research robot, Autonomous Research System (ARES), that performs CNT growth experiments. The detailed description of the growth experiments can be found in our previous works (Nikolaev et al., 2016; Rao et al., 2012; Nikolaev et al., 2014). We limit the design space into practical ranges of the two input variables. The practical range for the concentration ratio of the two chemical reactants is from 0 to 6.7 in the log scale or from 1 to 800. The ratio below zero cannot expect any growth, because there is more growth suppressor (CO_2) than the catalyst. The

ratio 800 is regarded as almost pure catalysts, so the further increase of the ratio would not be more effective. The reaction temperature ranges from 600 to 1100 Celsius. The temperature below 550 is too low to induce the growth of carbon nanotubes, and the temperature above 1200 is difficult to apply given a heat source and the melting temperature of supporting materials. The design space would be $[0, 6.7] \times [600, 1100]$ of the log ratio and temperature. We use the proposed multi-stage sequential design approach for exploring the response surface over the design space. The first stage is the seed experiment, and the experimental design of the first stage is hand-picked by an expert. For running experiments efficiently, five distinct values of the concentration ratio are tried, which are 1, 10, 100, 400 and 680 or 0, 2.3, 4.6, 6.0 and 6.5 in the log scale. For each of the five values, five to seven reaction temperatures are tried, and the reaction temperatures are hand-picked from the temperature range where jumps in growth are expected based on some prior engineering knowledge. In total, 31 design points are selected for the seed stage. In the second stage and thereafter, the design points are chosen by our proposed approach. Given experimental costs, each stage cannot perform too many experiments, so we run 20 experiments per stage. The stages continue until we have a satisfactory outcome. Therefore, the total selection size n is adaptively chosen.

6.3 Results

Fig. 11-(a) shows the response surface estimated with the experimental outcomes at the 31 design points of the first stage, and Fig. 11-(b) shows the jump detection statistics (equation (12)) estimated with the first stage sample. The yellow band with a quite thick bandwidth in the figure is the potential jump region. The thick bandwidth implies that the region of jumps in nanotube growth is not narrowed down, so we need to take more experiments to narrow it down. Based on the statistics and the corresponding sampling density $f_{2|1}$, the second stage samples are taken as shown in Fig. 11-(b). The samples are mostly from the yellow band. After the second stage is completed, the response surface is re-estimated as shown in Fig. 11-(c), and the corresponding jump detection statistic is estimated as shown in Fig. 11-(d). In the figure (d), we observe that there is a narrow region where the jump detection statistic has much higher values than the other region. This narrow region corresponds to where the response surface jumps around. Since the jump region is narrowed down enough, so we decided to stop the design selection. In total, we took 51 design points, which is a very small number compared to more than several hundred design points necessary to narrow down the jump region following the uniform design of experiments.

7 Conclusion

We proposed a novel adaptive design strategy (cf., (14)) for sequential selection of design points in jump regression analysis. The proposed method originated from our asymptotic error analysis of the jump regression estimate based on the one-sided local linear kernel smoothing, which showed that placing more design points around the jump location curves would give a faster decay of the integrated mean square regression error. Therefore, the proposed sampling function has a large density around the jump location curves. The proposed strategy was applied to two materials science applications, the compressive material

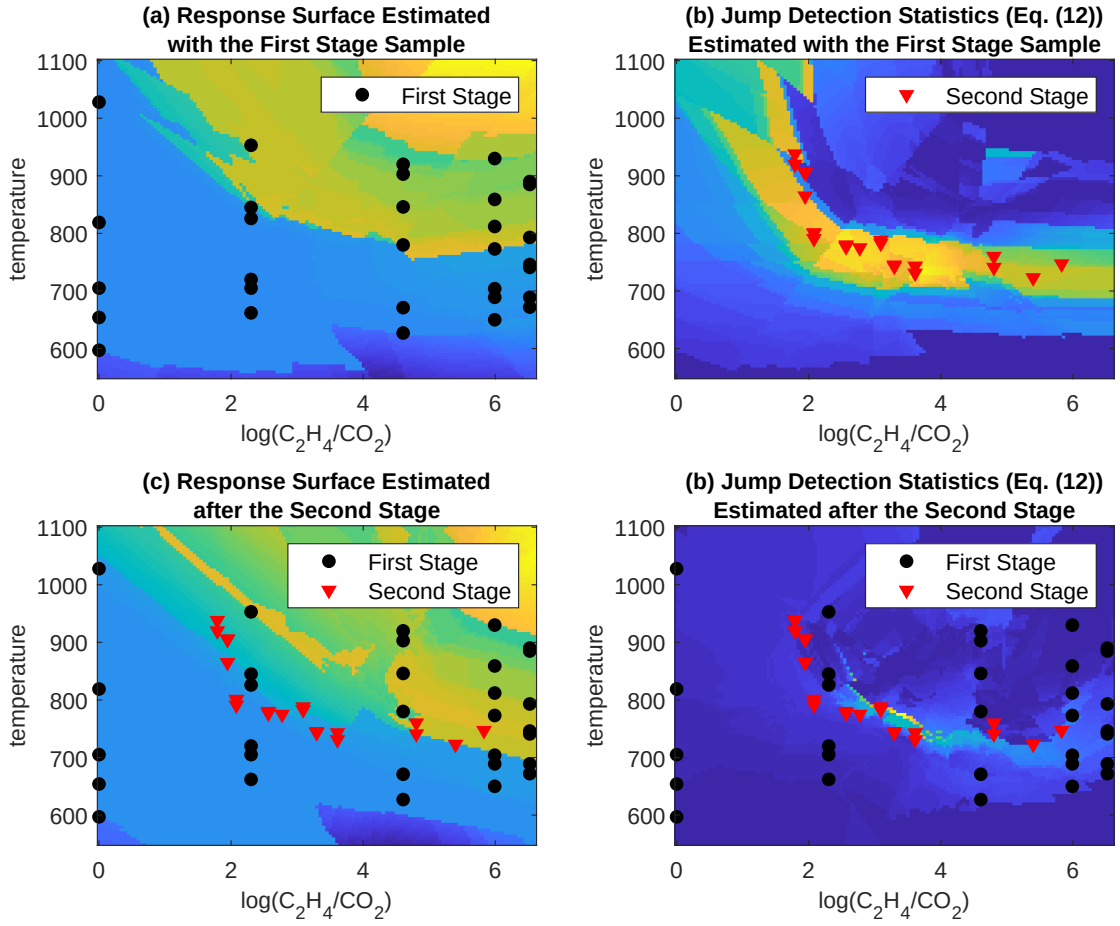

Figure 11: Application of the Proposed Sequential Design Approach. (a) shows the response surface estimate with the first-stage seed experiment. (b) shows the jump detection statistic in equation (12) estimated with the first stage experiment. In (b), red triangules represent the second stage sample from the sampling density $f_{2|1}$ that is calculated using the jump detection statistic. (c) shows the response surface estimate updated with the second stage experiment. (d) shows the jump detection statistic updated with the second stage experiment. In (d), the jump region is narrowed down to a thin layered region, around which there are many design points sampled.

imaging problem in which sub-sampled images are used for reconstructing full images and the design selection for accelerating the materials discovery. The outcomes are promising. We have showed the STEM imaging can be accelerated for at least ten times faster, while sharp image features are preserved, unless the image noise level is comparable to or higher than the image contrast. We also showed from the second example that experimental campaigns for materials discovery in carbon nanotubes can be accelerated by using our proposed approach.

References

Arias-Castro, E., E. J. Candes, and M. A. Davenport (2013). On the fundamental limits of adaptive sensing. *IEEE Transactions on Information Theory 59*(1), 472–481.

Bull, A. D. (2013). Spatially-adaptive sensing in nonparametric regression. *The Annals of Statistics 41*(1), 41–62.

Carpena-Núñez, J., P. Nikolaev, M. Susner, R. Rao, S. Gorsse, C. Park, and B. Maruyama (2020). Mapping carbon nanotube catalyst phase transitions using jump regression. *Unpublished manuscript*.

Chaudhuri, P. and P. A. Mykland (1993). Nonlinear experiments: Optimal design and inference based on likelihood. *Journal of the American Statistical Association 88*(422), 538–546.

Chernoff, H. (1972). *Sequential analysis and optimal design*. SIAM.

Cohn, D. A., Z. Ghahramani, and M. I. Jordan (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research 4*, 129–145.

Cortes, C., G. DeSalvo, C. Gentile, M. Mohri, and N. Zhang (2019). Region-based active learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2801–2809. PMLR.

Dror, H. A. and D. M. Steinberg (2008). Sequential experimental designs for generalized linear models. *Journal of the American Statistical Association 103*(481), 288–298.

Goetz, J., A. Tewari, and P. Zimmerman (2018). Active learning for non-parametric regression using purely random trees. In *Advances in Neural Information Processing Systems*, pp. 2542–2551.

Gramacy, R. B. and H. K. Lee (2009). Adaptive design and analysis of supercomputer experiments. *Technometrics 51*(2), 130–145.

Hoang, T. N., B. K. H. Low, P. Jaillet, and M. Kankanhalli (2014). Nonmyopic ϵ-bayes-optimal active learning of gaussian processes.

Kim, H.-M., B. K. Mallick, and C. C. Holmes (2005). Analyzing nonstationary spatial data using piecewise gaussian processes. *Journal of the American Statistical Association 100*(470), 653–668.

Krause, A., A. Singh, and C. Guestrin (2008). Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research 9*(Feb), 235–284.

Lee, C., K. Wang, J. Wu, W. Cai, and X. Yue (2021). Partitioned active learning for heterogeneous systems. *arXiv preprint arXiv:2105.08547*.

Mack, Y. and M. Rosenblatt (1979). Multivariate k-nearest neighbor density estimates. *Journal of Multivariate Analysis 9*(1), 1–15.

Malloy, M. L. and R. D. Nowak (2014). Near-optimal adaptive compressed sensing. *IEEE Transactions on Information Theory 60*(7), 4001–4012.

Nikolaev, P., D. Hooper, N. Perea-Lopez, M. Terrones, and B. Maruyama (2014). Discovery of wall-selective carbon nanotube growth conditions via automated experimentation. *ACS Nano 8*(10), 10214–10222.

Nikolaev, P., D. Hooper, F. Webber, R. Rao, K. Decker, M. Krein, J. Poleski, R. Barto, and B. Maruyama (2016). Autonomy in materials research: a case study in carbon nanotube growth. *npj Computational Materials 2*, 16031.

Paisley, J., X. Liao, and L. Carin (2010). Active learning and basis selection for kernel-based linear models: A bayesian perspective. *IEEE Transactions on Signal Processing 58*(5), 2686–2700.

Pope, C. A., J. P. Gosling, S. Barber, J. S. Johnson, T. Yamaguchi, G. Feingold, and P. G. Blackwell (2021). Gaussian process modeling of heterogeneity and discontinuities using voronoi tessellations. *Technometrics 63*(1), 53–63.

Qiu, P. (1998). Discontinuous regression surfaces fitting. *The Annals of Statistics 26*(6), 2218–2245.

Qiu, P. (2004). The local piecewisely linear kernel smoothing procedure for fitting jump regression surfaces. *Technometrics 46*(1), 87–98.

Qiu, P. (2005). *Image processing and jump regression analysis*, Volume 599. John Wiley & Sons.

Qiu, P. (2009). Jump-preserving surface reconstruction from noisy data. *Annals of the Institute of Statistical Mathematics 61*(3), 715–751.

Qiu, P. and B. Yandell (1997). Jump detection in regression surfaces. *Journal of Computational and Graphical Statistics 6*(3), 332–354.

Ranjan, P., D. Bingham, and G. Michailidis (2008). Sequential experiment design for contour estimation from complex computer codes. *Technometrics 50*(4), 527–541.

Rao, R., D. Liptak, T. Cherukuri, B. I. Yakobson, and B. Maruyama (2012). In situ evidence for chirality-dependent growth rates of individual carbon nanotubes. *Nature Materials 11*(3), 213.

Ruppert, D. and M. P. Wand (1994). Multivariate locally weighted least squares regression. *The Annals of Statistics*, 1346–1370.

Sacks, J., W. J. Welch, T. J. Mitchell, and H. P. Wynn (1989). Design and analysis of computer experiments. *Statistical Science*, 409–423.

Singh, A., A. Krause, C. Guestrin, and W. J. Kaiser (2009). Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research 34*, 707–755.

Stevens, A., L. Kovarik, P. Abellan, X. Yuan, L. Carin, and N. D. Browning (2015). Applying compressive sensing to tem video: a substantial frame rate increase on any camera. *Advanced Structural and Chemical Imaging 1*(1), 10.

Taddy, M. A., R. B. Gramacy, and N. G. Polson (2011). Dynamic trees for learning and design. *Journal of the American Statistical Association 106*(493), 109–123.

Wasserman, L. (2006). *All of nonparametric statistics*. Springer Science & Business Media.

Zhao, Z. and W. Yao (2012). Sequential design for nonparametric inference. *Canadian Journal of Statistics 40*(2), 362–377.

Zhu, Z. and M. L. Stein (2006). Spatial sampling design for prediction with estimated parameters. *Journal of Agricultural, Biological, and Environmental Statistics 11*(1), 24.

Zimmerman, D. L. (2006). Optimal network design for spatial prediction, covariance parameter estimation, and empirical prediction. *Environmetrics 17*(6), 635–652.

Online Supplementary Materials

Appendix A. Proof of Theorem 3.1

Suppose that $\boldsymbol{x} \in \mathcal{A}_b$ and it is non-singular in that $\mathcal{N}_n(\boldsymbol{x})$ does intersect only with $\mathcal{A}_b$ and one another sub-region, say $\mathcal{A}_{b'}$. The local linear kernel estimate $\hat{m}_{(0)}(\boldsymbol{x})$ can be expressed as

$$\hat{m}_{(0)}(\boldsymbol{x}) = \sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i) Y_i,$$

for a conditional second order kernel ω that satisfies the following conditions:

$$\sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i) = 1 \text{ and } \sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i)(\boldsymbol{x}_i - \boldsymbol{x}) = 0.$$

We use Ruppert and Wand (1994, Theorem 2.1) to get the variance of the estimate to be

$$Var[\hat{m}_{(0)}(\boldsymbol{x})] = \frac{\sigma^2}{n h_n^p} R(K) / f(\boldsymbol{x})(1 + o_P(1)),$$

where $R(K) = \int K^2(\boldsymbol{u}) d\boldsymbol{u}$. Since we choose the spatial varying bandwidth $h_n(\boldsymbol{x}) \propto n^{-1/p} f(\boldsymbol{x})^{-1/p}$, the variance is asymptotically a constant since

$$Var[\hat{m}_{(0)}(\boldsymbol{x})] = \kappa_1 \sigma^2 (1 + o_P(1)),$$

where κ_1 is a fixed constant. The expectation of the estimate is

$$E[\hat{m}_{(0)}(\boldsymbol{x})] = \sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i) m(\boldsymbol{x}_i).$$

To further analyze the term, let $Q_n^{(b)} = \mathcal{N}_n(\boldsymbol{x}) \cap \mathcal{A}_b$ and $Q_n^{(b')} = \mathcal{N}_n(\boldsymbol{x}) \cap \mathcal{A}_{b'}$. The expectation can be split accordingly as follows:

$$\begin{aligned} E[\hat{m}_{(0)}(\boldsymbol{x})] &= \sum_{\boldsymbol{x}_i \in Q_n^{(b)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i) m(\boldsymbol{x}_i) + \sum_{\boldsymbol{x}_i \in Q_n^{(b')}} \omega(\boldsymbol{x}, \boldsymbol{x}_i) m(\boldsymbol{x}_i) \\ &= \sum_{\boldsymbol{x}_i \in Q_n^{(b)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i) g_b(\boldsymbol{x}_i) + \sum_{\boldsymbol{x}_i \in Q_n^{(b')}} \omega(\boldsymbol{x}, \boldsymbol{x}_i) g_{b'}(\boldsymbol{x}_i) \end{aligned} \tag{15}$$

Let $\boldsymbol{x}_J$ denote a boundary point in $\partial\mathcal{A}_b \cap \partial\mathcal{A}_{b'}$ which is closest from $\boldsymbol{x}$, and let $c_J = g_{b'}(\boldsymbol{x}_J) - g_b(\boldsymbol{x}_J)$ denote the intensity jump at the boundary location. We take the second order Taylor expansion of $g_b(\boldsymbol{x}_i)$ at $\boldsymbol{x}_J$ as

$$g_b(\boldsymbol{x}_i) = g_b(\boldsymbol{x}_J) + \boldsymbol{d}_{J,b}^T(\boldsymbol{x}_i - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_J)^T \boldsymbol{Q}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x}_J) + o_P(h_n^2),$$

where $\boldsymbol{d}_{J,b}$ is the first order partial derivative of g_b at $\boldsymbol{x}_J$, $\boldsymbol{H}_{J,b}$ is the Hessian matrix at $\boldsymbol{x}_J$, and the remainder term is bounded by h_n. With the Taylor expansion, the expectation

(15) can be written as

$$
\begin{aligned}
&E[\hat{m}_{(0)}(\boldsymbol{x})] \\
&= \sum_{\boldsymbol{x}_i \in Q_n^{(b)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{g_b(\boldsymbol{x}_J) + \boldsymbol{d}_{J,b}^T(\boldsymbol{x}_i - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_J)^T \boldsymbol{H}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x}_J)\} \\
&+ \sum_{\boldsymbol{x}_i \in Q_n^{(b')}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{g_{b'}(\boldsymbol{x}_J) + \boldsymbol{d}_{J,b'}^T(\boldsymbol{x}_i - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_J)^T \boldsymbol{H}_{J,b'}(\boldsymbol{x}_i - \boldsymbol{x}_J)\} \\
&+ o_P(h_n^2) \\
&= \sum_{\boldsymbol{x}_i \in Q_n^{(b)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{g_b(\boldsymbol{x}_J) + \boldsymbol{d}_{J,b}^T(\boldsymbol{x}_i - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_J)^T \boldsymbol{H}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x}_J) \\
&+ \sum_{\boldsymbol{x}_i \in Q_n^{(b')}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{c_J + g_b(\boldsymbol{x}_J) + \boldsymbol{d}_{J,b'}^T(\boldsymbol{x}_i - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_J)^T \boldsymbol{H}_{J,b'}(\boldsymbol{x}_i - \boldsymbol{x}_J)\} \\
&+ o_P(h_n^2) \\
&= \sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{g_b(\boldsymbol{x}_J) + \boldsymbol{d}_{J,b}^T(\boldsymbol{x}_i - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_J)^T \boldsymbol{H}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x}_J)\} \\
&+ \sum_{\boldsymbol{x}_i \in Q_n^{(b')}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{c_J + \boldsymbol{\delta}_J^T(\boldsymbol{x}_i - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_J)^T \boldsymbol{\Delta}_J(\boldsymbol{x}_i - \boldsymbol{x}_J)\} + o_P(h_n^2),
\end{aligned}
$$

where $\boldsymbol{\delta}_J = \boldsymbol{d}_{J,b'} - \boldsymbol{d}_{J,b}$, and $\boldsymbol{\Delta}_J = \boldsymbol{H}_{J,b'} - \boldsymbol{H}_{J,b}$. With the Taylor expansion of $m(\boldsymbol{x})$ at $\boldsymbol{x}_J$,

$$
m(\boldsymbol{x}) = g_b(\boldsymbol{x}_J) + \boldsymbol{d}_{J,b}^T(\boldsymbol{x} - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_J)^T \boldsymbol{H}_{J,b}(\boldsymbol{x} - \boldsymbol{x}_J) + o_P(h_n^2), \tag{16}
$$

the bias of $\hat{m}_{(0)}(\boldsymbol{x})$ is

$$
\begin{aligned}
&E[\hat{m}_{(0)}(\boldsymbol{x})] - m(\boldsymbol{x}) \\
&= \sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{(\boldsymbol{d}_{J,b} - \boldsymbol{H}_{J,b}\boldsymbol{x}_J + 2\boldsymbol{H}_{J,b}\boldsymbol{x})^T(\boldsymbol{x}_i - \boldsymbol{x}) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{H}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x})\} \\
&\quad + \sum_{\boldsymbol{x}_i \in Q_n^{(2)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{c_J + \boldsymbol{\delta}_J^T(\boldsymbol{x}_i - \boldsymbol{x}_J) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_J)^T \boldsymbol{\Delta}_J(\boldsymbol{x}_i - \boldsymbol{x}_J)\} + o_P(h_n^2) \\
&= \sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{H}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x}) \\
&\quad + \sum_{\boldsymbol{x}_i \in Q_n^{(2)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\{c_J + (\boldsymbol{\delta}_J + \boldsymbol{\Delta}_J \boldsymbol{n}_J)^T(\boldsymbol{x}_i - \boldsymbol{x}) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{\Delta}_J(\boldsymbol{x}_i - \boldsymbol{x})\} \\
&\quad + \frac{1}{2}(\boldsymbol{\Delta}_J \boldsymbol{n}_J + 2\boldsymbol{\delta}_J)^T \boldsymbol{n}_J \sum_{\boldsymbol{x}_i \in Q_n^{(2)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i) \\
&= \sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{H}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x}) \\
&\quad + \sum_{\boldsymbol{x}_i \in Q_n^{(2)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\left\{c_J + c\boldsymbol{\delta}_J^T(\boldsymbol{x}_i - \boldsymbol{x}) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{\Delta}_J(\boldsymbol{x}_i - \boldsymbol{x})\right\}
\end{aligned}
$$

where $\boldsymbol{n}_J = \boldsymbol{x} - \boldsymbol{x}_J$, and $\boldsymbol{H}_\tau \boldsymbol{n}_J + \boldsymbol{d}_\tau = c\boldsymbol{d}_\tau$ for a constant c. In the last equation, the first term is the same as the bias of the estimate in cases when there is no jump around $\boldsymbol{x}$, and the second term is the contribution of the nearby jump to the bias. Using the result for the local linear kernel estimation (Ruppert and Wand, 1994), the first term is

$$
\sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{H}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x}) = \frac{1}{2}\mu_2(K)\left(h_n^2 \sum_{j=1}^{d} \frac{\partial^2 g(\boldsymbol{x})}{\partial x_j^2}\right) + o_P(h_n^2),
$$

where $\mu_2(K)$ is a kernel-dependent constant with $\mu_2(K)\boldsymbol{I} = \int \boldsymbol{u}\boldsymbol{u}^T K(\boldsymbol{u}) d\boldsymbol{u}$. Since g is smooth with a bounded second derivative,

$$
\sum_{\boldsymbol{x}_i \in \mathcal{N}_n(\boldsymbol{x})} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{H}_{J,b}(\boldsymbol{x}_i - \boldsymbol{x}) = o_P(h_n^2) = o_P\left(n^{-2/p} f(x)^{-2/p}\right). \tag{17}
$$

Using Mack and Rosenblatt (1979, Theorem 2.1),

$$
\begin{aligned}
&\sum_{\boldsymbol{x}_i \in Q_n^{(b')}} c_J \omega(\boldsymbol{x}, \boldsymbol{x}_i) = \left(\frac{1}{f(x)} + o_P(1)\right) n^{-1} \sum_{\boldsymbol{x}_i \in \mathcal{Q}_n^{(b')}} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n}\right) c_J \\
&\sum_{\boldsymbol{x}_i \in Q_n^{(b')}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)\boldsymbol{\delta}_J^T(\boldsymbol{x}_i - \boldsymbol{x}) = \left(\frac{1}{f(x)} + o_P(1)\right) n^{-1} \sum_{\boldsymbol{x}_i \in \mathcal{Q}_n^{(b')}} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n}\right) \boldsymbol{\delta}_J^T(\boldsymbol{x}_i - \boldsymbol{x}) \\
&\sum_{\boldsymbol{x}_i \in Q_n^{(b')}} \omega(\boldsymbol{x}, \boldsymbol{x}_i)(\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{\Delta}_J(\boldsymbol{x}_i - \boldsymbol{x}) \\
&\qquad = \left(\frac{1}{f(x)} + o_P(1)\right) n^{-1} \sum_{\boldsymbol{x}_i \in \mathcal{Q}_n^{(b')}} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n}\right) (\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{\Delta}_J(\boldsymbol{x}_i - \boldsymbol{x})
\end{aligned}
$$

and the second term in the bias expression is

$$\sum_{\boldsymbol{x}_i \in Q_n^{(2)}} \omega(\boldsymbol{x}, \boldsymbol{x}_i) \left\{ c_J + c\boldsymbol{\delta}_J^T (\boldsymbol{x}_i - \boldsymbol{x}) + \frac{1}{2} (\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{\Delta}_J (\boldsymbol{x}_i - \boldsymbol{x}) \right\}$$
$$= \left(\frac{1}{f(x)} + o_P(1) \right) n^{-1} \sum_{\boldsymbol{x}_i \in \mathcal{Q}_n^{(b')}} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n} \right) \left\{ c_J + c\boldsymbol{\delta}_J^T (\boldsymbol{x}_i - \boldsymbol{x}) + \frac{1}{2} (\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{\Delta}_J (\boldsymbol{x}_i - \boldsymbol{x}) \right\}$$

where

$$\begin{aligned} n^{-1} \sum_{\boldsymbol{x}_i \in \mathcal{Q}_n^{(b')}} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n} \right) &= \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) f(\boldsymbol{x} + h_n \boldsymbol{u}) d\boldsymbol{u} + o_P(1) \\ &= \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) \{ f(\boldsymbol{x}) + h_n D_f(\boldsymbol{x})^T \boldsymbol{u} + o(h_n) \} d\boldsymbol{u} + o_P(1) \\ &= f(\boldsymbol{x}) \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) d\boldsymbol{u} + h_n D_f(\boldsymbol{x})^T \int_{\mathcal{Q}^{(b')}} \boldsymbol{u} K(\boldsymbol{u}) d\boldsymbol{u} + o_P(1) \\ &= f(\boldsymbol{x}) \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) d\boldsymbol{u} + o_P(1), \end{aligned}$$

$$\begin{aligned} n^{-1} \sum_{\boldsymbol{x}_i \in \mathcal{Q}_n^{(b')}} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n} \right) \boldsymbol{\delta}_J^T (\boldsymbol{x}_i - \boldsymbol{x}) &= \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) h_n \boldsymbol{u} f(\boldsymbol{x} + h_n \boldsymbol{u}) d\boldsymbol{u} + o_P(h_n) \\ &= \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) h_n \boldsymbol{\delta}_J^T \boldsymbol{u} \{ f(\boldsymbol{x}) + h_n D_f(\boldsymbol{x})^T \boldsymbol{u} + o(h_n) \} d\boldsymbol{u} + o_P(h_n) \\ &= h_n f(\boldsymbol{x}) \boldsymbol{\delta}_J^T \int_{\mathcal{Q}^{(b')}} \boldsymbol{u} K(\boldsymbol{u}) d\boldsymbol{u} + h_n^2 \boldsymbol{\delta}_J^T \int_{\mathcal{Q}^{(b')}} \boldsymbol{u}\boldsymbol{u}^T K(\boldsymbol{u}) d\boldsymbol{u} \cdot D_f(\boldsymbol{x}) + o_P(h_n) \\ &= h_n f(\boldsymbol{x}) \boldsymbol{\delta}_J^T \int_{\mathcal{Q}^{(b')}} \boldsymbol{u} K(\boldsymbol{u}) d\boldsymbol{u} + o_P(h_n), \text{ and} \end{aligned}$$

$$\begin{aligned} n^{-1} \sum_{\boldsymbol{x}_i \in \mathcal{Q}_n^{(b')}} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n} \right) (\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{\Delta}_J (\boldsymbol{x}_i - \boldsymbol{x}) &= \int_{\mathcal{Q}^{(b')}} K\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h_n} \right) (\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{\Delta}_J (\boldsymbol{x}_i - \boldsymbol{x}) f(\boldsymbol{x}_i) d\boldsymbol{x}_i \\ &= \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) h_n^2 \boldsymbol{u}^T \boldsymbol{\Delta}_J \boldsymbol{u} f(\boldsymbol{x} + h_n \boldsymbol{u}) d\boldsymbol{u} + o_P(h_n^2) \\ &= \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) h_n^2 \boldsymbol{u}^T \boldsymbol{\Delta}_J \boldsymbol{u} \{ f(\boldsymbol{x}) + h_n D_f(\boldsymbol{x})^T \boldsymbol{u} + o_P(h_n) \} d\boldsymbol{u} + o_P(h_n^2) \\ &= h_n^2 f(\boldsymbol{x}) \int_{\mathcal{Q}^{(b')}} \boldsymbol{u}^T \boldsymbol{\Delta}_J \boldsymbol{u} K(\boldsymbol{u}) d\boldsymbol{u} + h_n^3 \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) \boldsymbol{u}^T \boldsymbol{\Delta}_J \boldsymbol{u}\boldsymbol{u}^T d\boldsymbol{u} D_f(\boldsymbol{x}) + o_P(h_n^2) \\ &= h_n^2 f(\boldsymbol{x}) tr\left(\boldsymbol{\Delta}_J \int_{\mathcal{Q}^{(b')}} \boldsymbol{u}\boldsymbol{u}^T K(\boldsymbol{u}) d\boldsymbol{u} \right) + o_P(h_n^2). \end{aligned}$$

Therefore, the second term in the bias expression is asymptotically to be

$$c_J \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) d\boldsymbol{u} + h_n \boldsymbol{\delta}_J^T \int_{\mathcal{Q}^{(b')}} \boldsymbol{u} K(\boldsymbol{u}) d\boldsymbol{u} + h_n^2 tr\left(\boldsymbol{\Delta}_J \int_{\mathcal{Q}^{(b')}} \boldsymbol{u}\boldsymbol{u}^T K(\boldsymbol{u}) d\boldsymbol{u} \right) + o_P(h_n^2), \tag{18}$$

where $Q^{(b')}$ is the part of the support of K that corresponds to $Q_n^{(b')}$.

Based on the results of (17) and (18), the bias can be described as

$$\begin{aligned} E[\hat{m}_{(0)}(\boldsymbol{x})] - m(\boldsymbol{x}) =& o_P\left(\frac{1}{n^{2/p} f(x)^{2/p}}\right) \\ &+ c_J \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) d\boldsymbol{u} \\ &+ h_n \boldsymbol{\delta}_J^T \int_{\mathcal{Q}^{(b')}} \boldsymbol{u} K(\boldsymbol{u}) d\boldsymbol{u} \\ &+ h_n^2 tr\left(\boldsymbol{\Delta}_J \int_{\mathcal{Q}^{(b')}} \boldsymbol{u}\boldsymbol{u}^T K(\boldsymbol{u}) d\boldsymbol{u}\right) \\ =& o_P\left(\frac{1}{n^{2/p} f(x)^{2/p}}\right) + (c_J + o_P(1)) \int_{\mathcal{Q}^{(b')}} K(\boldsymbol{u}) d\boldsymbol{u}. \end{aligned} \tag{19}$$

Appendix B. J-MSE and C-MSE for different noise levels and different values of n

Fig. 12 shows J-MSE and C-MSE for different noise levels and different values of n. We can see clear downward trends in both J-MSE and C-MSE as the noise level decreases or SNR increases. The C-MSE increases quadratically in σ, and the J-MSE increases linearly, which may be because the estimation error due to jumps dominates the estimation error due to noise around the jump boundaries.

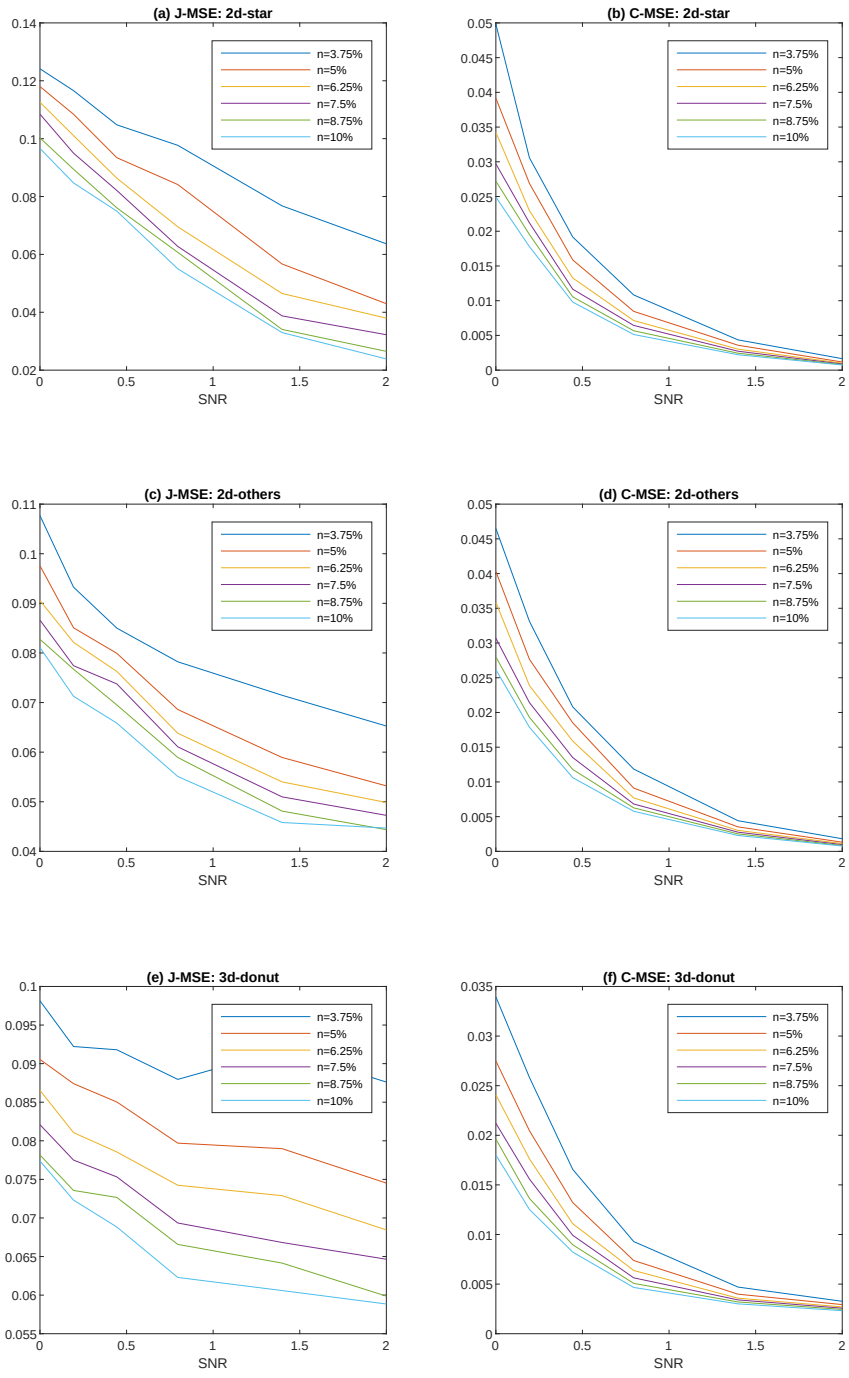

Figure 12: Effect of Noise Level σ. Here we plot the results for $b = 1.25\%$, because the results for other values of b follow similar patterns.

Appendix C. Numerical illustrations of adaptive STEM scan results discussed in Section 5

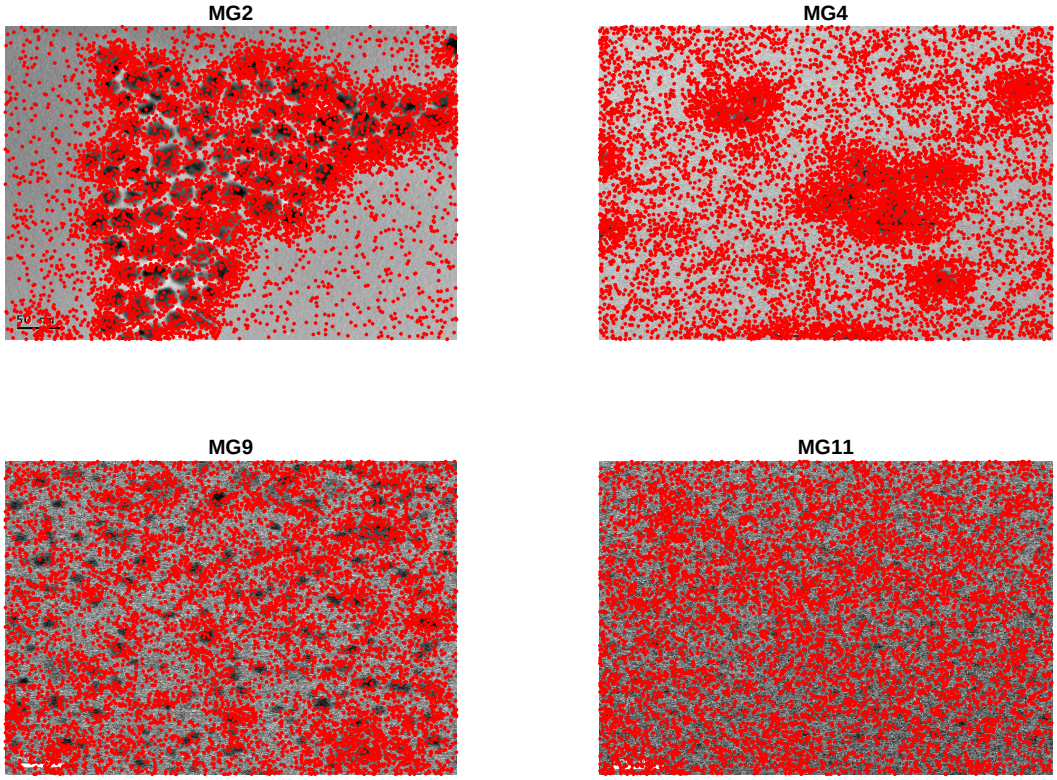

Figure 13: Locations of Partial Scans. Red dots represent the locations for n=3%.