Journal of Machine Learning Research 26 (2025) 1-25 Submitted 8/21; Revised 10/25; Published 10/25

Hierarchical and Stochastic Crystallization Learning: Geometrically Leveraged Nonparametric Regression with Delaunay Triangulation

Jiaqi Gu JIAQIGU@USF.EDU
Department of Mathematics and Statistics
University of South Florida
Tampa, FL 33620, USA

Guosheng Yin* GYIN@HKU.HK
Department of Statistics and Actuarial Science
School of Computing and Data Science
University of Hong Kong
Hong Kong SAR, China

Editor: Risi Kondor

Abstract

High-dimensionality is known to be the bottleneck for both nonparametric regression and the Delaunay triangulation. To efficiently exploit the advantage of the Delaunay triangulation in utilizing geometry information for nonparametric regression without conducting the Delaunay triangulation for the entire feature space, we develop the crystallization search for the neighbor Delaunay simplices of the target point similar to crystal growth and estimate the conditional expectation function by fitting a local linear model to the data points of the constructed Delaunay simplices. Because the shapes and volumes of Delaunay simplices are adaptive to the density of feature data points, our method selects neighbor data points more uniformly in all directions in comparison with Euclidean distance based methods and thus it is more robust to the local geometric structure of the data. We further develop the stochastic approach to hyperparameter selection and the hierarchical crystallization learning under multimodal feature data densities, where an approximate global Delaunay triangulation is obtained by first triangulating the local centers and then constructing local Delaunay triangulations in parallel. We study the asymptotic properties of our method and conduct numerical experiments on both synthetic and real data to demonstrate the advantages of our method over the existing ones.

Keywords: Crystallization, Delaunay Triangulation, High dimensionality, Nonparametric Regression, Parallel triangulation.

1. Introduction

Nonparametric regression methods are popular in statistics and machine learning due to their robustness (or model-free) property. In machine learning applications, decision trees, random forests, and nearest neighbor approaches are well-known model-free methods. Nevertheless, tree-based methods are sequential in the tree growth in the sense that one layer

*. Guosheng Yin is the corresponding author.

of nodes depends on the previous layer. Regarding nearest neighbor methods, nearest neighbors are often selected based on the Euclidean distance from the target data point. Although different distance metrics or kernelization can incorporate the global geometric information to nearest neighbor methods, imposing a common metric or kernel on all target points still ignores the local geometric information in the data and leads to large estimation errors, particularly when the data distribution is highly skewed or contains sudden jumps.

Our goal is to estimate the conditional mean in a nonparametric regression model while incorporating the local geometric structure of the data. More specifically, let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be independent and identically distributed (i.i.d.) feature points from a density $f(\mathbf{x})$ in the d-dimensional Euclidean space $\mathscr{R}^d$ ($n > d$). Let $y_1, \ldots, y_n$ be the corresponding observed responses, and we consider

$$y_i = \mu(\mathbf{x}_i) + \epsilon_i, \quad i = 1, \ldots, n, \tag{1}$$

where $\mu(\cdot) = E(Y|\cdot)$ is the conditional expectation function of the response Y and $\epsilon_1, \ldots, \epsilon_n \in \mathscr{R}$ are i.i.d. errors with $E(\epsilon_i) = 0$ and $E(\epsilon_i^2) < \infty$. For estimating $\mu(\cdot)$ without any rigid assumptions on its shape, various nonparametric regression methods have been developed in recent decades, including kernel smoothing (Nadaraya, 1964; Watson, 1964; Priestley and Chao, 1972; Härdle and Gasser, 1984; Hein, 2009), nearest-neighbor methods (Cover and Hart, 1967; Benedetti, 1977; Stone, 1977; Altman, 1992), local linear regression (Cleveland, 1979; Cleveland and Devlin, 1988; Fan and Gijbels, 1996). All the existing methods follow a common philosophy that y_i is more informative of the conditional expectation $\mu(\mathbf{z})$ if the corresponding feature point $\mathbf{x}_i$ is more similar or closer to the target point $\mathbf{z}$. As a result, $\mu(\mathbf{z})$ can be estimated by fitting a local model under which only the selected neighbor data points would be used or those data points similar to the target would be assigned higher weights.

Although the consistency of the aforementioned methods has been established under mild conditions, their finite-sample performances are sensitive to the local geometric structure of the observed feature points. Because only the distances from the target point $\mathbf{z}$ to the observed feature points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are considered in identifying the neighbor data points or assigning weights, it is possible that the directions from $\mathbf{z}$ to its neighbors (or angles between $\mathbf{z}$ and the neighbor $\mathbf{x}_i$'s) are not uniformly distributed, especially when $\mathbf{z}$ is close to the boundary of the convex hull of observed feature points or jump points of the feature data density $f(\mathbf{x})$. As a result, the (weighted) mean of y_i's corresponding to the neighbor data points may be far from that of the target point $\mathbf{z}$, leading to large bias in estimating the conditional expectation $\mu(\mathbf{z})$. To take the local geometric structure into account, the Delaunay interpolation (also known as Delaunay triangulation learning), has been developed to incorporate the Delaunay triangulation (Delaunay, 1934) into the framework of nonparametric regression (Liu and Yin, 2020).

However, the Delaunay triangulation cannot overcome the curse of dimensionality for the whole space. By focusing on the local neighborhood of the target point, Gu and Yin (2021) developed the crystallization search, which mimics the crystallization process in thermodynamics and obtains the Delaunay simplices closest to the target. The contributions of our work are three-fold: (i) We take a hierarchical approach to first identifying the local key centers and then grow the crystals around these centers in parallel, which can greatly expedite the Delaunay triangulation over the entire space and thus overcome the difficulties

caused by high dimensionality. (ii) We propose the stochastic crystallization learning to cope with the discontinuity in hyperparameter selection. (iii) Through hierarchical triangulation, we can avoid generating sharp-shaped Delaunay simplicies under multimodal feature data density $f(\mathbf{x})$. As a result, a new nonparametric regression method named the hierarchical crystallization learning is proposed to estimate $\mu(\mathbf{z})$ by fitting a local linear model to the data points of all the neighbor Delaunay simplices obtained by the crystallization search.

The rest of this article is organized as follows. In Section 2, we introduce the Delaunay triangulation and interpolation. Section 3 presents crystallization search to grow the neighbor simplices, local estimation procedure, and hyperparameter selection under the deterministic manner. In Section 4, we propose the stochastic approach to searching for the topological distance parameter by introducing an energy distribution function. We further develop a hierarchical crystallization learning procedure in Section 5, which can overcome the curse-of-dimensionality through first selecting centers and then conduct Delaunay triangulation around those centers in parallel. We present the asymptotic properties of the hierarchical crystallization learning in Section 6. Experiments on synthetic and real data are conducted in Section 7 to compare the our proposal with existing methods in terms of estimation and prediction accuracy. Section 9 concludes with a discussion.

2. Delaunay Interpolation

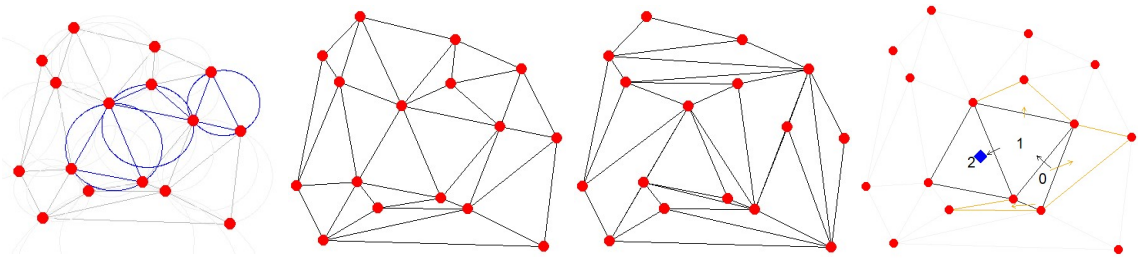
Let $\mathbb{X}$ be a set of n feature points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in the Euclidean space $\mathscr{R}^d$ $(n > d)$. A d-dimensional triangulation of $\mathbb{X}$, denoted as $\mathcal{T}(\mathbb{X})$, is a mesh of d-simplices $\{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$ satisfying:

1. For $j = 1, \ldots, m$, the set of $d+1$ vertices of simplex $\mathcal{S}_j$, denoted as $\mathbb{V}(\mathcal{S}_j)$, is a subset of $\mathbb{X}$ and does not lie in any affine hyperplane of $\mathscr{R}^d$.

2. For any $j \neq k$, simplices $\mathcal{S}_j$ and $\mathcal{S}_k$ are disjoint except on their shared boundaries $\mathcal{S}_j \cap \mathcal{S}_k$.

3. The union $\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_m$ is the convex hull of $\mathbb{X}$, denoted by $\mathcal{H}(\mathbb{X})$.

As the d-simplices $\mathcal{S}_1, \ldots, \mathcal{S}_m$ of the triangulation $\mathcal{T}(\mathbb{X})$ fully cover the convex hull $\mathcal{H}(\mathbb{X})$, for each internal point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$, there exists a simplex $\mathcal{S}(\mathbf{z}) \in \mathcal{T}(\mathbb{X})$ such that $\mathbf{z} \in \mathcal{S}(\mathbf{z})$. Let $i_1(\mathbf{z}), \ldots, i_{d+1}(\mathbf{z})$ denote the indices corresponding to the data points of $\mathcal{S}(\mathbf{z})$, and then there exist $d+1$ values of $\gamma_1, \ldots, \gamma_{d+1} \in [0, 1]$ such that $\sum_{k=1}^{d+1} \gamma_k \mathbf{x}_{i_k(\mathbf{z})} = \mathbf{z}$ and $\sum_{k=1}^{d+1} \gamma_k = 1$. Among all triangulations, the Delaunay triangulation is the most widely used for multivariate interpolation (de Berg et al., 2008) due to its smoothness property. Specifically, let $\mathcal{B}_j$ be the open ball whose boundary is the circumscribed $(d-1)$-sphere of $\mathcal{S}_j$. The Delaunay triangulation of $\mathbb{X}$, denoted as $\mathcal{DT}(\mathbb{X})$, is a triangulation of $\mathbb{X}$ such that $\mathcal{B}_j \cap \mathbb{X} = \varnothing$ for $j = 1, \ldots, m$, which is known as the empty-ball property, as shown in Figure 1 (a). As the geometric dual of the Voronoi diagram under the L_2 norm, the Delaunay triangulation generates a mesh of simplices that are most regularized in shape. In a two-dimensional space, $\mathbb{X} \subset \mathscr{R}^2$, the Delaunay triangulation $\mathcal{DT}(\mathbb{X})$ maximizes the minimum angle in all the triangles (2-simplices) $\mathcal{S}_1, \ldots, \mathcal{S}_m$ over all possible triangulations (Sibson, 1978), as shown in Figure 1 (b) and (c). The Delaunay triangulation $\mathcal{DT}(\mathbb{X})$ is unique under the assumption that $\mathbb{X}$ is in general position (Delaunay, 1934).



(a) (b) (c) (d)

Figure 1: (a) Graphical illustration of the empty-ball property of the Delaunay triangulation; (b) the Delaunay triangulation; (c) a random triangulation; (d) graphical illustration of the breadth first search in the DELAUNAYSPARSE algorithm.

Considering the data $\{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$ from the regression model (1), the Delaunay interpolation aims to estimate the conditional expectation $\mu(\mathbf{z})$ for all $\mathbf{z} \in \mathcal{H}(\mathbb{X})$. Generally, there are three steps in the Delaunay interpolation: (i) construct the Delaunay triangulation $\mathcal{DT}(\mathbb{X})$; (ii) find the simplex $\mathcal{S}(\mathbf{z}) \in \mathcal{DT}(\mathbb{X})$; and (iii) obtain the estimated function $\hat{\mu}(\cdot)$ by optimizing a specific target function and compute $\hat{\mu}(\mathbf{z})$. For example, with $\gamma_1, \ldots, \gamma_{d+1} \in [0, 1]$ such that $\sum_{k=1}^{d+1} \gamma_k \mathbf{x}_{i_k(\mathbf{z})} = \mathbf{z}$ and $\sum_{k=1}^{d+1} \gamma_k = 1$, the estimator of de Berg et al. (2008) is

$$\hat{\mu}(\mathbf{z}) = \sum_{k=1}^{d+1} \gamma_k y_{i_k(\mathbf{z})}, \tag{2}$$

where $\hat{\mu}(\cdot)$ is the minimizer of the squared loss function $\sum_{i=1}^{n} (y_i - g(\mathbf{x}_i))^2$ among all continuous piecewise linear functions $g(\mathbf{x}) = \sum_{j=1}^{m} 1_{\{\mathbf{x} \in \mathcal{S}_j\}} (\alpha_j + \boldsymbol{\beta}_j^{\mathsf{T}} \mathbf{x})$. Liu and Yin (2020) introduced a regularization function to balance the model fitting and smoothness of the estimator $\hat{\mu}(\cdot)$. However, all the aforementioned methods require a complete construction of $\mathcal{DT}(\mathbb{X})$ for the entire feature space, whose size (i.e., m) grows exponentially with the dimension d. As a result, no existing algorithm is feasible when $d > 7$ due to the limitations of computation time/power and memory space (Chang et al., 2020).

Alternatively, several methods have been proposed to circumvent the curse-of-dimensionality issue for medium- to high-dimensional Delaunay interpolation (Chang et al., 2018a,b, 2020). Instead of obtaining the complete $\mathcal{DT}(\mathbb{X})$, these methods only locally construct the Delaunay simplex $\mathcal{S}(\mathbf{z})$ at each target point $\mathbf{z}$ and thus $\mu(\mathbf{z})$ can be estimated at a polynomial cost. For any point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$, the DELAUNAYSPARSE algorithm (Chang et al., 2020) first computes a seed Delaunay simplex $\mathcal{S}_{\text{seed}}$ close to $\mathbf{z}$. Specifically, to generate $\mathcal{S}_{\text{seed}}$, the nearest neighbor $\mathbf{x}_{\text{seed}_0}$ of $\mathbf{z}$ is found (step 2) and then $\mathbf{x}_{\text{seed}_1}, \ldots, \mathbf{x}_{\text{seed}_d}$ are recursively found by minimizing the diameter of the circumscribed sphere of the simplex $\{\mathbf{x}_{\text{seed}_0}, \ldots, \mathbf{x}_{\text{seed}_k}\}$ in a greedy manner (steps 3-8). As a result, for $k = 1, \ldots, d$, the open ball whose boundary is the circumscribed sphere of the simplex $\{\mathbf{x}_{\text{seed}_0}, \ldots, \mathbf{x}_{\text{seed}_k}\}$ contains no points of $\mathbb{X}$, guaranteeing the empty-ball property of $\mathcal{S}_{\text{seed}}$. Based on $\mathcal{S}_{\text{seed}}$, it finds $\mathcal{S}(\mathbf{z})$ via the breadth first search as presented in Algorithm 1 and Figure 1 (d) and further computes the estimator $\hat{\mu}(\mathbf{z})$ via (2). Although such an approach is computationally efficient because $\gamma_1, \ldots, \gamma_{d+1}$ are simultaneously calculated in searching $\mathcal{S}(\mathbf{z})$, it only utilizes the information of $d + 1$ data points $\{(\mathbf{x}_{i_k(\mathbf{z})}, y_{i_k(\mathbf{z})}) : k = 1, \ldots, d + 1\}$ in the estimation. This may lead

Algorithm 1 DELAUNAYSPARSE (Chang et al., 2020)

Input: Feature points $\mathbb{X}$, target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$.
Initialize: The seed Delaunay simplex $\mathcal{S}_{\text{seed}} = \{\mathbf{x}_{\text{seed}_0}\}$ where $\text{seed}_0 = \arg\min_i \|\mathbf{x}_i - \mathbf{z}\|_2$.
for $k = 1, \ldots, d$ **do**
  for $i \in \{1, \ldots, n\} \setminus \mathcal{S}_{\text{seed}}$ **do**
    Compute r_{ik} as the the diameter of the circumscribing sphere of the simplex $\mathcal{S}_{\text{seed}} \cup \{\mathbf{x}_i\}$.
  end for
  Let $\mathcal{S}_{\text{seed}} \leftarrow \mathcal{S}_{\text{seed}} \cup \{\mathbf{x}_{\text{seed}_k}\}$ where $\text{seed}_k = \arg\min_i r_{ik}$.
end for
Let $\mathcal{S}_{\text{current}} = \mathcal{S}_{\text{seed}}$, $\mathbb{A}_{\text{Frontier}} = \{\mathcal{S}_{\text{seed}}\}$, $\mathbb{A}_{\text{Explored}} = \varnothing$.
while $\mathbf{z} \notin \mathcal{S}_{\text{current}}$ **do**
  Compute $\mathbb{F}_{\mathbf{z}}(\mathcal{S}_{\text{current}})$, the set of facets of $\mathcal{S}_{\text{current}}$ which is visible to $\mathbf{z}$. A facet $\mathcal{F}$ of the simplex $\mathcal{S}_{\text{current}}$ is visible to $\mathbf{z}$ if there exists an internal point $\mathbf{z}'$ of $\mathcal{S}_{\text{current}}$ such that the linear segment from $\mathbf{z}$ to $\mathbf{z}'$ intersects $\mathcal{F}$.
  for each facet $\mathcal{F} \in \mathbb{F}_{\mathbf{z}}(\mathcal{S}_{\text{current}})$ **do**
    Grow a new Delaunay simplex $\mathcal{S}_{\text{new}} \neq \mathcal{S}_{\text{current}}$ on the facet $\mathcal{F}$ if it exists.
    $\mathbb{A}_{\text{Frontier}} \leftarrow \mathbb{A}_{\text{Frontier}} \cup \{\mathcal{S}_{\text{new}}\}$ if $\mathcal{S}_{\text{new}}$ exists and $\mathcal{S}_{\text{new}} \notin \mathbb{A}_{\text{Explored}} \cup \mathbb{A}_{\text{Frontier}}$.
  end for
  $\mathbb{A}_{\text{Explored}} \leftarrow \mathbb{A}_{\text{Explored}} \cup \{\mathcal{S}_{\text{current}}\}$.
  $\mathbb{A}_{\text{Frontier}} \leftarrow \mathbb{A}_{\text{Frontier}} \setminus \{\mathcal{S}_{\text{current}}\}$.
  $\mathcal{S}_{\text{current}} \leftarrow$ the first simplex in $\mathbb{A}_{\text{Frontier}}$.
end while
Output: Simplex $\mathcal{S}_{\text{current}}$.

to overfitting and poor estimation when the simplex $\mathcal{S}(\mathbf{z})$ has a small volume or a poorly regularized shape.

3. Crystallization Learning

3.1 Crystallization Search for Delaunay Simplices

As one component of $\mathcal{DT}(\mathbb{X})$, $\mathcal{S}(\mathbf{z})$ has $d+1$ facets $\mathcal{F}_1, \ldots, \mathcal{F}_{d+1}$, each of which is either a facet of the boundary of $\mathcal{H}(\mathbb{X})$ or the shared boundary of $\mathcal{S}(\mathbf{z})$ and one of its neighbor Delaunay simplicies.

Definition 1 ***Neighbor Delaunay simplices:*** *Given the Delaunay triangulation $\mathcal{DT}(\mathbb{X}) = \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$ of $\mathbb{X} \in \mathscr{R}^d$, Delaunay simplices $\mathcal{S}_j$ and $\mathcal{S}_k$ are neighbors if and only if the intersection $\mathcal{S}_j \cap \mathcal{S}_k$ is a shared facet of $\mathcal{S}_j$ and $\mathcal{S}_k$.*

On the basis of Algorithm 1 (Chang et al., 2020) which searches $\mathcal{S}(\mathbf{z})$ by recursively growing neighbor Delaunay simplices on the facets of the explored ones, we develop the crystallization search (Algorithm 2) to construct all the Delaunay simplices within the topological distance L to $\mathcal{S}(\mathbf{z})$, denoted as $\mathcal{N}_L(\mathbf{z})$. When $L = 0$, only the simplex $\mathcal{S}(\mathbf{z})$ is

Algorithm 2 Crystallization search

1: **Input:** Feature points $\mathbb{X}$, target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$ and the maximal topological distance L.
2: Compute $\mathcal{S}(\mathbf{z})$ via Algorithm 1.
3: Let $\mathbb{A}_{\text{Frontier}} = \{(\mathcal{S}(\mathbf{z}), 0)\}$ and $\mathcal{N}_L(\mathbf{z}) = \varnothing$.
4: **while** $\mathbb{A}_{\text{Frontier}} \neq \varnothing$ **do**
5: $(\mathcal{S}_{\text{current}}, L_{\text{current}}) \leftarrow$ the first element in $\mathbb{A}_{\text{Frontier}}$.
6: **if** $L_{\text{current}} < L$ **then**
7: Compute all the facets of $\mathcal{S}_{\text{current}}$, denoted as $\mathcal{F}_1, \ldots, \mathcal{F}_{d+1}$.
8: **for** $j = 1, \ldots, d+1$ **do**
9: Grow a new Delaunay simplex $\mathcal{S}_{\text{new}} \neq \mathcal{S}_{\text{current}}$ on the facet $\mathcal{F}_j$ if it exists.
10: $\mathbb{A}_{\text{Frontier}} \leftarrow \mathbb{A}_{\text{Frontier}} \cup \{(\mathcal{S}_{\text{new}}, L_{\text{current}} + 1)\}$ if $\mathcal{S}_{\text{new}}$ exists and $\mathcal{S}_{\text{new}} \notin \mathcal{N}_L(\mathbf{z}) \cup \mathbb{A}_{\text{Frontier}}$.
11: **end for**
12: **end if**
13: $\mathcal{N}_L(\mathbf{z}) \leftarrow \mathcal{N}_L(\mathbf{z}) \cup \{\mathcal{S}_{\text{current}}\}$.
14: $\mathbb{A}_{\text{Frontier}} \leftarrow \mathbb{A}_{\text{Frontier}} \setminus \{(\mathcal{S}_{\text{current}}, L_{\text{current}})\}$.
15: **end while**
16: **Output:** The set of Delaunay simplices $\mathcal{N}_L(\mathbf{z})$.

Table 1: Average runtime (in seconds) in computing $\mathcal{N}_L(\mathbf{z})$ under different values of the maximal topological distance L, sample size n, and dimension d.

L	$n = 500$			$n = 1000$			$n = 2000$		
	$d = 6$	$d = 8$	$d = 10$	$d = 6$	$d = 8$	$d = 10$	$d = 6$	$d = 8$	$d = 10$
2	0.05	0.09	0.14	0.06	0.11	0.18	0.07	0.14	0.23
3	0.23	0.51	0.98	0.28	0.63	1.22	0.34	0.80	1.56
4	0.82	2.26	5.20	1.02	2.80	6.51	1.21	3.55	8.27

constructed. As L increases, Delaunay simplices are constructed in a sequential way that new simplices grow on the facets of the explored ones. The whole process of crystallization search is analogous to the crystallization process in thermodynamics, where the search of $\mathcal{S}(\mathbf{z})$ indexed by line 2 in Algorithm 2 plays the role of nucleation and the remaining steps correspond to crystal growth.

As shown by Chang et al. (2020), the average computational complexity of Algorithm 1 is $\mathcal{O}(d^2 n)$. However, as Algorithm 1 is only implemented once in the crystallization search, the dominant cost of Algorithm 2 lies in the process of growing simplices (lines 4—15). Because each d-simplex has $d+1$ facets, there are at most $d+1$ neighbor simplices for each Delaunay simplices and thus the number of generated Delaunay simplices in Algorithm 2 is $\mathcal{O}(d^L)$. With the rank-1 update suggested by Chang et al. (2020), the average computational complexity of growing a new Delaunay simplex on the facet of an existing Delaunay simplex is $\mathcal{O}(n)$. Thus, the average computational complexity of Algorithm 2 is $\mathcal{O}(d^L n)$. Table 1 shows the average runtime in computing $\mathcal{N}_L(\mathbf{z})$ under different configurations.

3.2 Estimation with Weighted Least Squares

Let $\mathbb{V}_{\mathbf{z},L} = \cup_{\mathcal{S}\in\mathcal{N}_L(\mathbf{z})}\mathbb{V}(\mathcal{S})$ denote the set of all the data points of the simplices in $\mathcal{N}_L(\mathbf{z})$, where $\mathbb{V}(\mathcal{S})$ represents all the vertices of simplex $\mathcal{S}$. Based on the set $\mathcal{N}_L(\mathbf{z})$, we propose the crystallization learning to estimate $\mu(\mathbf{z})$ by fitting a local linear model,

$$\mu(\mathbf{z}) = \alpha + \boldsymbol{\beta}^{\mathsf{T}}\mathbf{z},$$

to all the data points in $\mathbb{V}_{\mathbf{z},L}$ instead of using only the $d+1$ data points of $\mathcal{S}(\mathbf{z})$. If a vertex shared by more Delaunay simplices, it typically has a larger degree in the network formed by Delaunay edges and thus provides more informative on the geometric structure of $\mathcal{N}_L(\mathbf{z})$. By assigning more weights to such vertices, we estimate α and $\boldsymbol{\beta}$ via the weighted least squares approach,

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}}) = \arg\min_{\alpha,\boldsymbol{\beta}} \sum_{\mathbf{x}_i\in\mathbb{V}_{\mathbf{z},L}} \mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)(y_i - \alpha - \boldsymbol{\beta}^{\mathsf{T}}\mathbf{x}_i)^2, \tag{3}$$

where the weight function is given by

$$\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i) = \left(\sum_{\mathcal{S}\in\mathcal{N}_L(\mathbf{z})} 1_{\{\mathbf{x}_i\in\mathbb{V}(\mathcal{S})\}}\right)\exp\left(-\frac{\|\mathbf{x}_i-\mathbf{z}\|_2^2}{m_L(\mathbf{z})}\right),$$
$$\text{with } m_L(\mathbf{z}) = \left(\sum_{\mathbf{x}_i\in\mathbb{V}_{\mathbf{z},L}} \|\mathbf{x}_i-\mathbf{z}\|_2^2\right)\Big/\left(\sum_{i=1}^{n} 1_{\{\mathbf{x}_i\in\mathbb{V}_{\mathbf{z},L}\}}\right).$$

Once we obtain the estimators $\hat{\alpha}$ and $\hat{\boldsymbol{\beta}}$, we can predict the outcome at the target point, $\hat{\mu}(\mathbf{z}) = \hat{\alpha} + \hat{\boldsymbol{\beta}}^{\mathsf{T}}\mathbf{z}$. Similar to the work of Nadaraya (1964) and Watson (1964), our weight function places more weights on the data points closer to $\mathbf{z}$ as well as those shared by more simplices in $\mathcal{N}_L(\mathbf{z})$. For all $\mathbf{x}_i \notin \mathbb{V}_{\mathbf{z},L}$, the weights are set to be zero. In addition, our weight function is scale-invariant due to the normalization term $m_L(\mathbf{z})$, i.e., multiplying any constant to features would not change the weights. As a result, the estimated conditional expectation function, $\hat{\mu}(\cdot)$, is piecewise smooth but not piecewise linear in $\mathcal{H}(\mathbb{X})$, as demonstrated by Theorem 1 with the proof given in the Appendix A.

Theorem 1 *Let $\mathbb{X}$ be a set of n feature points $\mathbf{x}_1,\ldots,\mathbf{x}_n$ in general position and responses $y_1,\ldots,y_n$ are generated from model (1). The estimated conditional expectation function under the crystallization learning, $\hat{\mu}(\cdot)$, is smooth in $\mathcal{S}_k$, for $k = 1,\ldots,m$.*

3.3 Selection of Topological Distance L

The statistical complexity and estimation performance of the crystallization learning is controlled by the hyperparameter L, the maximal topological distance from the generated neighbor Delaunay simplices to $\mathcal{S}(\mathbf{z})$. Because a small L leads to overfitting and a large L makes $\hat{\mu}(\cdot)$ overly smooth, we propose the leave-one-out cross validation (LOO-CV) to select L with respect to the target point $\mathbf{z}$.

1. Compute the Delaunay simplex $\mathcal{S}(\mathbf{z})$ containing $\mathbf{z}$ and the values of $\gamma_1,\ldots,\gamma_{d+1} \in [0,1]$ such that $\sum_{k=1}^{d+1}\gamma_k\mathbf{x}_{i_k(\mathbf{z})} = \mathbf{z}$ and $\sum_{k=1}^{d+1}\gamma_k = 1$ via Algorithm 1, where $\mathbf{x}_{i_1(\mathbf{z})},\ldots,\mathbf{x}_{i_{d+1}(\mathbf{z})}$ are the $d+1$ data points of $\mathcal{S}(\mathbf{z})$.

2. For each $\mathbf{x}_{i_k(\mathbf{z})} \in \mathcal{S}(\mathbf{z})$, apply the crystallization learning with different candidate values of L on the leave-one-out data excluding $(\mathbf{x}_{i_k(\mathbf{z})}, y_{i_k(\mathbf{z})})$ to estimate $\mu(\mathbf{x}_{i_k(\mathbf{z})})$. Let $\hat{\mu}(\mathbf{x}_{i_k(\mathbf{z})}; L)$ be the estimator corresponding to the observation $(\mathbf{x}_{i_k(\mathbf{z})}, y_{i_k(\mathbf{z})})$ and candidate value L.

3. Select the optimal $\widetilde{L}$ as

$$\widetilde{L} = \arg\min_{L} \sum_{k=1}^{d+1} \gamma_k \log\{\hat{\mu}(\mathbf{x}_{i_k(\mathbf{z})}; L) - y_{i_k(\mathbf{z})}\}^2. \tag{4}$$

4. Stochastic Crystallization Learning

As the crystallization search (Algorithm 2) grows the Delaunay simplicies $\mathcal{N}_L(\mathbf{z})$ in a deterministic way, the hyperparameter L can only take integer values. To obtain a continuous domain for the hyperparameter, we develop the stochastic counterpart of crystallization learning where the neighbor Delaunay simplicies can grow with randomness. This is analogous to considering a saturated solution where the solubility of the solute decreases as the temperature decreases. The cooling crystallization process is subject to a thermodynamic rule that the size of the crystal increases as the solution loses more thermal energy while the shape is random. To take the shape randomness into account as in the physical mechanism, we develop the stochastic crystallization search (Algorithm 3), which computes the neighbor Delaunay simplicies of $\mathcal{S}(\mathbf{z})$ in a random and recursive manner similar to the Mondrian process (Roy and Teh, 2009). Beginning with $\mathcal{S}(\mathbf{z})$, the growth of each new simplex is assigned with a random value λ which measures the energy loss of crystal growth. The whole process of Delaunay simplicies construction would not be terminated until the cumulative energy loss exceeds a pre-specified maximal energy loss Λ. Figure 2 summarizes 100 randomly generated $\mathcal{N}_\Lambda(\mathbf{z})$ ($\Lambda = 1, \ldots, 6$) computed by Algorithm 3 with respect to a target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$ in $\mathscr{R}^2$ and the energy distribution

$$h_{\Delta_{\mathcal{S}_{\text{new}},\mathbf{z}}}(\lambda) = \frac{1}{\Delta_{\mathcal{S}_{\text{new}},\mathcal{S}(\mathbf{z})}} \exp\left(-\frac{\lambda}{\Delta_{\mathcal{S}_{\text{new}},\mathcal{S}(\mathbf{z})}}\right), \tag{5}$$

where $\Delta_{\mathcal{S}_{\text{new}},\mathcal{S}(\mathbf{z})}$ is the topological distance between $\mathcal{S}_{\text{new}}$ and $\mathcal{S}(\mathbf{z})$. Although the energy distribution can take arbitrary form, we choose the exponential distribution here to mimic the memoryless property of energy loss in the cooling crystallization process that the amount of energy loss in new crystal formation is independent to the amount of energy loss for all existing crystals.

Let $\mathcal{N}_\Lambda^{(1)}(\mathbf{z}), \ldots, \mathcal{N}_\Lambda^{(B)}(\mathbf{z})$ be B sets of stochastic simplicies generated by Algorithm 3 and $\mathbb{V}_{\mathbf{z},\Lambda} = \cup_{b=1}^{B} \mathbb{V}_{\mathbf{z},\Lambda}^{(b)}$ with $\mathbb{V}_{\mathbf{z},\Lambda}^{(b)} = \cup_{\mathcal{S} \in \mathcal{N}_\Lambda^{(b)}(\mathbf{z})} \mathbb{V}(\mathcal{S})$. Similar to Section 3.2, $\mu(\mathbf{z})$ is estimated by fitting a local linear model to all the data points in $\mathbb{V}_{\mathbf{z},\Lambda}$ via the weighted least squares approach,

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}}) = \arg\min_{\alpha, \boldsymbol{\beta}} \sum_{\mathbf{x}_i \in \mathbb{V}_{\mathbf{z},\Lambda}} \mathrm{w}_{\mathbf{z},\Lambda}(\mathbf{x}_i)(y_i - \alpha - \boldsymbol{\beta}^{\mathsf{T}} \mathbf{x}_i)^2,$$

Algorithm 3 Stochastic crystallization search

Input: Feature points $\mathbb{X}$, target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$ and the maximal energy loss Λ.
Compute $\mathcal{S}(\mathbf{z})$ via Algorithm 1.
Let $\mathbb{A}_{\text{Frontier}} = \{(\mathcal{S}(\mathbf{z}), 0)\}$ and $\mathcal{N}_\Lambda(\mathbf{z}) = \varnothing$.
while $\mathbb{A}_{\text{Frontier}} \neq \varnothing$ **do**
  $(\mathcal{S}_{\text{current}}, \Lambda_{\text{current}}) \leftarrow$ the first element in $\mathbb{A}_{\text{Frontier}}$.
  Compute all the facets of $\mathcal{S}_{\text{current}}$, denoted as $\mathcal{F}_1, \ldots, \mathcal{F}_{d+1}$.
  Sample a permutation $\boldsymbol{\varphi} = (\varphi_1, \ldots, \varphi_{d+1})^\mathsf{T}$ of $(1, \ldots, d+1)^\mathsf{T}$.
  for $j = \varphi_1, \ldots, \varphi_{d+1}$ **do**
    Grow a new Delaunay simplex $\mathcal{S}_{\text{new}} \neq \mathcal{S}_{\text{current}}$ on the facet $\mathcal{F}_j$ if it exists.
    if $\mathcal{S}_{\text{new}}$ exists and $\mathcal{S}_{\text{new}} \notin \mathcal{N}_\Lambda(\mathbf{z}) \cup \mathbb{A}_{\text{Frontier}}$ **then**
      Draw energy loss λ from the energy density $h_{\Delta_{\mathcal{S}_{\text{new}},\mathcal{S}(\mathbf{z})}}(\lambda)$ where $\Delta_{\mathcal{S}_{\text{new}},\mathcal{S}(\mathbf{z})}$ is the topological distance between $\mathcal{S}_{\text{current}}$ and $\mathcal{S}(\mathbf{z})$.
      $\mathbb{A}_{\text{Frontier}} \leftarrow \mathbb{A}_{\text{Frontier}} \cup \{(\mathcal{S}_{\text{new}}, \Lambda_{\text{current}} + \lambda)\}$ if $\Lambda_{\text{current}} + \lambda \leq \Lambda$.
    end if
  end for
  $\mathcal{N}_\Lambda(\mathbf{z}) \leftarrow \mathcal{N}_\Lambda(\mathbf{z}) \cup \{\mathcal{S}_{\text{current}}\}$.
  $\mathbb{A}_{\text{Frontier}} \leftarrow \mathbb{A}_{\text{Frontier}} \setminus \{(\mathcal{S}_{\text{current}}, \Lambda_{\text{current}})\}$.
end while
Output: The set of Delaunay simplices $\mathcal{N}_\Lambda(\mathbf{z})$.

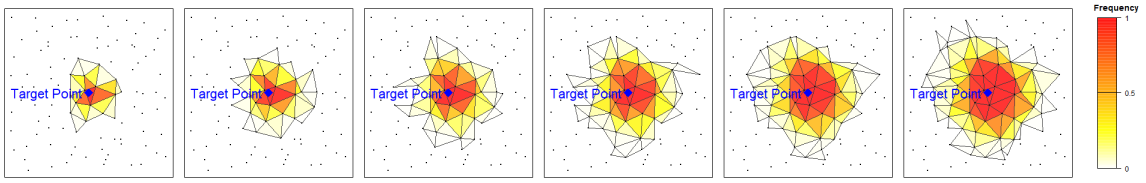
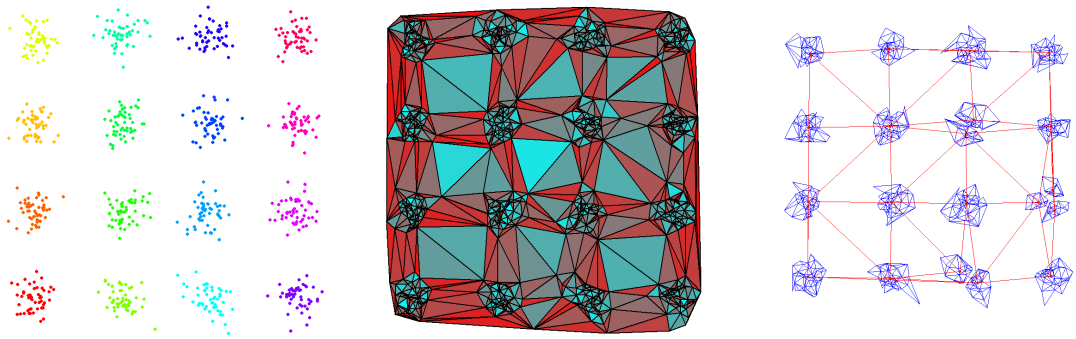


Figure 2: Stochastic crystallization search of $\mathcal{N}_\Lambda(\mathbf{z})$ for $\Lambda = 1, \ldots, 6$ under the energy distribution (5) with respect to a target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$ in $\mathscr{R}^2$, where Delaunay simplicies with higher frequencies falling in $\mathcal{N}_\Lambda(\mathbf{z})$ are filled with more intense color.

where the weight function is

$$\begin{aligned} \mathrm{w}_{\mathbf{z},\Lambda}(\mathbf{x}_i) &= \frac{1}{B}\left(\sum_{b=1}^{B} \sum_{\mathcal{S} \in \mathcal{N}_\Lambda^{(b)}(\mathbf{z})} 1_{\{\mathbf{x}_i \in \mathbb{V}(\mathcal{S})\}}\right) \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{z}\|_2^2}{m_\Lambda(\mathbf{z})}\right), \\ \text{with } m_\Lambda(\mathbf{z}) &= \left(\sum_{b=1}^{B} \sum_{\mathbf{x}_i \in \mathbb{V}_{\mathbf{z},\Lambda}^{(b)}} \|\mathbf{x}_i - \mathbf{z}\|_2^2\right) \Big/ \left(\sum_{b=1}^{B} \sum_{i=1}^{n} 1_{\{\mathbf{x}_i \in \mathbb{V}_{\mathbf{z},\Lambda}^{(b)}\}}\right). \end{aligned} \tag{6}$$

Note that Algorithm 2 is a special case of Algorithm 3 by setting $\Lambda = L$ and $h_{\Delta_{\mathcal{S}_{\text{new}},\mathbf{z}}}(\lambda)$ to be the Dirac distribution at point $\lambda = 1$.



(a) (b) (c)

Figure 3: (a) Feature points $\mathbb{X} \subset \mathscr{R}^2$ generated from a Gaussian mixture density (7); (b) many skinny 2-simplicies (visualized by red triangles) exist in the Delaunay triangulation $\mathcal{DT}(\mathbb{X})$; and (c) the Delaunay triangulation $\mathcal{DT}(\mathbb{X}^*)$ of representative points $\mathbb{X}^*$.

5. Hierarchical Crystallization Learning

Although the shapes of simplicies in the Delaunay triangulation are the most regularized compared to any other triangulations, it is possible that sharp-shaped simplicies exist in $\mathcal{DT}(\mathbb{X})$, especially when the feature data density $f(\mathbf{x})$ is multimodal. Consider an example with feature points $\mathbf{x}_1, \ldots, \mathbf{x}_{800} \in \mathscr{R}^2$ generated from a Gaussian mixture density,

$$f(\mathbf{x}) \propto \sum_{c_1=1}^{4} \sum_{c_2=1}^{4} \exp\{-2(x_1 - 4c_1)^2 - 2(x_2 - 4c_2)^2\} \tag{7}$$

as exhibited in Figure 3 (a). It is clear in Figure 3 (b) that the corresponding Delaunay triangulation $\mathcal{DT}(\mathbb{X})$ consists of many skinny simplicies. As a result, many simplicies in $\mathcal{N}_L(\mathbf{z})$ would be sharp-shaped if the target point $\mathbf{z}$ falls in the sparse region of the density $f(\mathbf{x})$, leading to poor performance in estimating $\mu(\mathbf{z})$.

To handle the multimodality issue, we develop the hierarchical crystallization learning by incorporating representative point selection (Stampfer and Stadlober, 2002; Daszykowski et al., 2002; Qi et al., 2017; Zhu et al., 2019) as follows. Given a set of feature points $\mathbb{X}$, we apply a cluster-based representative point selection method to obtain a set of n^* representative points $\mathbb{X}^* = \{\mathbf{x}_1^*, \ldots, \mathbf{x}_{n^*}^*\}$ and then partition $\mathbb{X}$ into $\{\mathbb{X}_1, \ldots, \mathbb{X}_{n^*}\}$, where $\mathbb{X}_{i^*}$ contains all the feature data points surrounding the representative point $\mathbf{x}_{i^*}^*$. In particular, we use the k-means algorithm to obtain n^* cluster centers as representative points $\mathbf{x}_1^*, \ldots, \mathbf{x}_{n^*}^*$ and construct $\mathbb{X}_{i^*} = \{\mathbf{x}_i : \|\mathbf{x}_i - \mathbf{x}_{i^*}^*\| \leq \|\mathbf{x}_i - \mathbf{x}_{j^*}^*\|, \forall j^* \neq i^*\}$. We then estimate $\mu(\mathbf{z})$ according to the property of the simplex $\mathcal{S}(\mathbf{z})$ computed by Algorithm 1 as follows:

(i) If all the vertices of $\mathcal{S}(\mathbf{z})$ belong to the same $\mathbb{X}_{i^*}$, the estimator $\hat{\mu}(\mathbf{z})$ is obtained via the procedures proposed in Section 3.

(ii) If the vertices of $\mathcal{S}(\mathbf{z})$ belong to more than one $\mathbb{X}_{i^*}$, we construct $\mathcal{N}_L^*(\mathbf{z})$ on representative points $\mathbb{X}^*$ by Algorithm 2. Let $\mathbb{V}_{\mathbf{z},L}^* = \cup_{\mathcal{S} \in \mathcal{N}_L^*(\mathbf{z})} \mathbb{V}^*(\mathcal{S})$ denote the set of all the

representative points of the simplices in $\mathcal{N}_L^*(\mathbf{z})$, where $\mathbb{V}^*(\mathcal{S})$ are all the representative vertices of simplex $\mathcal{S}$. The estimator $\hat{\mu}(\mathbf{z})$ is obtained by fitting a local linear model to all the data points in $\cup_{\mathbf{x}_{i^*}^* \in \mathbb{V}_{\mathbf{z},L}^*} \mathbb{X}_{i^*}$ via the weighted least squares approach,

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}}) = \arg\min_{\alpha,\boldsymbol{\beta}} \sum_{\mathbf{x}_{i^*}^* \in \mathbb{V}_{\mathbf{z},L}^*} \mathrm{w}_{\mathbf{z},L}^*(\mathbf{x}_{i^*}^*) \Big\{ \sum_{\mathbf{x}_i \in \mathbb{X}_{i^*}} (y_i - \alpha - \boldsymbol{\beta}^\mathsf{T} \mathbf{x}_i)^2 \Big\},$$

where the weight function is

$$\mathrm{w}_{\mathbf{z},L}^*(\mathbf{x}_{i^*}^*) = \Big(\sum_{\mathcal{S} \in \mathcal{N}_L^*(\mathbf{z})} \frac{1_{\{\mathbf{x}_{i^*}^* \in \mathbb{V}^*(\mathcal{S})\}}}{|\mathbb{X}_{i^*}|} \Big) \exp\Big(- \frac{\|\mathbf{x}_{i^*}^* - \mathbf{z}\|_2^2}{m_L^*(\mathbf{z})} \Big),$$

$$\text{with } m_L^*(\mathbf{z}) = \Big(\sum_{\mathbf{x}_{i^*}^* \in \mathbb{V}_{\mathbf{z},L}^*} \|\mathbf{x}_{i^*}^* - \mathbf{z}\|_2^2 \Big) \Big/ \Big(\sum_{i^*=1}^{n^*} 1_{\{\mathbf{x}_{i^*}^* \in \mathbb{V}_{\mathbf{z},L}^*\}} \Big),$$

and $|\mathbb{X}_{i^*}|$ is the number of data points in $\mathbb{X}_{i^*}$.

In the hierarchical crystallization learning, we can construct a two-layer structure of Deualnay triangulation as shown by Figure 3 (c), where the Delaunay triangulation on the representative points $\mathcal{DT}(\mathbb{X}^*)$ covers the low-density or sparse regions while $\mathcal{DT}(\mathbb{X}_{i^*})$ focus on the dense regions surrounding the representative points $\mathbf{x}_{i^*}^*$ $(i^* = 1, \ldots, n^*)$.

6. Asymptotic Theory

We first study the asymptotic geometric properties of the Delaunay triangulation $\mathcal{DT}(\mathbb{X})$ under the general distribution of feature points and then prove the consistency of the crystallization learning in estimating $\mu(\cdot)$. Let $\mathbb{X}$ be a set of n i.i.d. feature data points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathscr{R}^d$ from a density $f(\mathbf{x})$, which is strictly positive and bounded away from infinity on $\mathscr{R}^d$.

Lemma 1 *For any target point $\mathbf{z} \in \mathscr{R}^d$, we have that $\mathbb{P}(\mathbf{z} \in \mathcal{H}(\mathbb{X})) \to 1$, as $n \to \infty$.*

By Lemma 1, the target point $\mathbf{z}$ falls inside $\mathcal{H}(\mathbb{X})$ with asymptotic probability one, and thus we only need to consider the inside-hull case.

Theorem 2 *For any target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$ and any $\rho \in (0, 1)$, we have*

$$T(\mathbf{z}) = O_p(n^{-\rho/d}),$$

where $T(\mathbf{z}) = \max\{\|\mathbf{x}_i - \mathbf{z}\|_2; \mathbf{x}_i \in \mathbb{V}_{\mathbf{z},L}\}$ is the maximal L_2 norm between $\mathbf{z}$ and its neighbor feature data points.

Theorem 2 implies that all the feature points of $\mathbb{V}_{\mathbf{z},L}$ converge to $\mathbf{z}$ in probability.

Theorem 3 *Assume the data $\{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$ are generated from model (1), where the conditional expectation function $\mu(\cdot)$ is differentiable on $\mathscr{R}^d$. The estimated conditional expectation function under crystallization learning, $\hat{\mu}(\cdot)$, satisfies*

$$E\big\{\hat{\mu}(\mathbf{z}) - \mu(\mathbf{z})\big\}^2 \to 0, \quad \text{as } n \to \infty,$$

for all $\mathbf{z} \in \mathcal{H}(\mathbb{X})$ if $L \to \infty$ as $n \to \infty$.

All proofs of the lemma and theorems are given in Appendices B–D.

7. Experimental Studies

7.1 Deterministic Crystallization Learning

We conduct experiments on synthetic data under two different scenarios: (i) to illustrate the effectiveness of the crystallization learning in estimating the conditional expectation function $\mu(\cdot)$; (ii) to evaluate the estimation accuracy of the crystallization learning in comparison with existing nonparametric regression methods, including the k-NN regression using Euclidean distance, local linear regression using Gaussian kernel, multivariate kernel regression using Gaussian kernel (Hein, 2009) and Gaussian process models; and (iii) to validate the proposed data-driven procedure for selection of the hyperparameter L.

7.2 Estimation Accuracy

We consider two scenarios to investigate the estimation performance of our crystallization learning: (i) general internal points of $\mathcal{H}(\mathbb{X})$, and (ii) jump points of the feature data density. For each scenario, we simulate 100 training data sets $\{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$ under different values of sample size n and dimension d. For each training data set, we evaluate the prediction performance of our method on 100 randomly generated target points $\mathbf{z}_1, \ldots, \mathbf{z}_{100}$. We use the mean squared error (MSE) under $\mathcal{M}$ (a generic symbol for a method),

$$\text{MSE}_{\mathcal{M}} = \frac{1}{100}\sum_{k=1}^{100}\{\hat{\mu}_{\mathcal{M}}(\mathbf{z}_k) - \mu(\mathbf{z}_k)\}^2,$$

to evaluate the accuracy of the estimator $\hat{\mu}_{\mathcal{M}}(\cdot)$ at the target points $\mathbf{z}_1, \ldots, \mathbf{z}_{100} \in \mathcal{H}(\mathbb{X})$.

Scenario 1 (General internal points): For each data set, $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are independently sampled from the multivariate normal distribution $\text{MVN}(\mathbf{0}, \mathbf{I}_d)$ with an identity covariance matrix $\mathbf{I}_d$. The responses $y_1, \ldots, y_n$ are generated from an additive model,

$$Y|\mathbf{x} \sim N\bigg(\sum_{j=1}^{d} c_j g_j(x_j), 1\bigg), \tag{8}$$

where $\mathbf{x} = (x_1, \ldots, x_d)^\intercal$, $c_1, \ldots, c_d \sim N(0,1)$, $g_j(\cdot) = \sum_{l=1}^{10} b_{jl}\phi(\cdot; \nu_{jl}, \sigma_{jl}^2)$, $b_{jl} \sim N(0,1)$, $\nu_{jl} \sim N(0,1)$, $\sigma_{jl}^2 \sim \text{Gamma}(1,1)$, and $\phi(\cdot; \nu_{jl}, \sigma_{jl}^2)$ is the density of a normal distribution $N(\nu_{jl}, \sigma_{jl}^2)$, for $j = 1, \ldots, d; l = 1, \ldots, 10$. For $k = 1, \ldots, 100$, the target point $\mathbf{z}_k$ is generated as $\mathbf{z}_k = \sum_{i=1}^{n} \omega_{ik}\mathbf{x}_i$, with $(\omega_{1k}, \ldots, \omega_{nk}) \sim \text{Dirichlet}(1, \ldots, 1)$.

Scenario 2 (Jump points of the feature data density): For each data set, $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are sampled from the density,

$$f(\mathbf{x}) = 2^{-d}\prod_{j=1}^{d}(1 + 0.4 \cdot \text{sign}(x_j))\exp(-|x_j|),$$

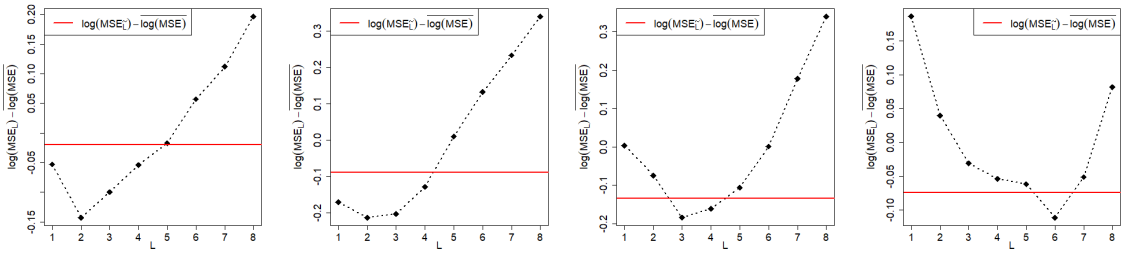
which has jumps at the set of points $\{\mathbf{x} \in \mathscr{R}^d : \prod_{j=1}^{d} x_j = 0\}$. Responses $y_1, \ldots, y_n$ are generated from the same additive model in (8). For $k = 1, \ldots, 100$, the target point $\mathbf{z}_k$ is generated as $\mathbf{z}_k = \sum_{i=1}^{n} \omega_{ik}\mathbf{x}_i \otimes \mathbf{s}_k$, where $(\omega_{1k}, \ldots, \omega_{nk}) \sim \text{Dirichlet}(1, \ldots, 1)$, $\otimes$ is the elementwise multiplication operator, $\mathbf{s}_k = (s_{k1}, \ldots, s_{kd})^\intercal$ and $s_{k1}, \ldots, s_{kd} \sim \text{Bernoulli}(0.7)$.

Table 2: Averaged values of log(MSE) and standard deviations in parentheses using crystallization learning (CL) in comparison with k-NN ($k = 5, 10, k^*$, where k^* equals the size of $\mathbb{V}_{\mathbf{z},L}$), local linear (LL) regression, kernel regression (KR) and Gaussian process (GP) in estimating $\mu(\cdot)$ under the two scenarios, with different sample sizes (n) and different dimensions of the feature space (d). The best results are highlighted in boldface.

d	n	$\log(\text{MSE}_{\text{CL}})$	$\log\left(\frac{\text{MSE}_{\text{5-NN}}}{\text{MSE}_{\text{CL}}}\right)$	$\log\left(\frac{\text{MSE}_{\text{10-NN}}}{\text{MSE}_{\text{CL}}}\right)$	$\log\left(\frac{\text{MSE}_{k^*\text{-NN}}}{\text{MSE}_{\text{CL}}}\right)$	$\log\left(\frac{\text{MSE}_{\text{LL}}}{\text{MSE}_{\text{CL}}}\right)$	$\log\left(\frac{\text{MSE}_{\text{KR}}}{\text{MSE}_{\text{CL}}}\right)$	$\log\left(\frac{\text{MSE}_{\text{GP}}}{\text{MSE}_{\text{CL}}}\right)$
Scenario 1: General internal points								
5	200	**-1.11 (0.21)**	0.23 (0.09)	0.12 (0.09)	0.33 (0.11)	0.56 (0.11)	0.57 (0.11)	0.24 (0.18)
	500	**-2.13 (0.18)**	0.55 (0.13)	0.37 (0.11)	0.45 (0.13)	0.91 (0.17)	0.94 (0.17)	0.76 (0.18)
	1000	**-2.04 (0.18)**	0.53 (0.13)	0.42 (0.13)	0.62 (0.12)	1.18 (0.19)	1.22 (0.19)	0.41 (0.20)
	2000	**-2.21 (0.20)**	0.48 (0.14)	0.38 (0.14)	0.59 (0.16)	1.06 (0.22)	1.08 (0.21)	0.81 (0.17)
10	200	-0.03 (0.16)	0.28 (0.09)	0.13 (0.07)	0.14 (0.08)	0.10 (0.07)	0.12 (0.07)	**-0.08 (0.14)**
	500	0.01 (0.21)	0.43 (0.13)	0.31 (0.10)	0.29 (0.11)	0.47 (0.12)	0.47 (0.12)	**-0.01 (0.17)**
	1000	-0.50 (0.22)	0.37 (0.14)	0.30 (0.12)	0.43 (0.10)	0.54 (0.12)	0.53 (0.12)	**-0.09 (0.21)**
	2000	**-0.67 (0.20)**	0.42 (0.13)	0.33 (0.12)	0.51 (0.11)	0.59 (0.16)	0.60 (0.16)	0.10 (0.14)
20	200	1.46 (0.14)	0.14 (0.08)	-0.02 (0.06)	-0.01 (0.06)	-0.02 (0.03)	**-0.04 (0.06)**	0.17 (0.15)
	500	1.09 (0.15)	0.25 (0.10)	0.11 (0.07)	-0.01 (0.07)	-0.07 (0.06)	-0.03 (0.06)	**-0.18 (0.16)**
	1000	0.92 (0.18)	0.48 (0.11)	0.36 (0.10)	0.00 (0.11)	**-0.10 (0.08)**	-0.02 (0.08)	0.22 (0.18)
	2000	**0.73 (0.22)**	0.24 (0.15)	0.24 (0.12)	0.06 (0.11)	0.18 (0.11)	0.14 (0.11)	0.15 (0.19)
50	500	2.47 (0.14)	0.08 (0.09)	-0.02 (0.07)	0.02 (0.05)	-0.01 (0.03)	**-0.08 (0.11)**	0.06 (0.19)
	1000	2.32 (0.17)	0.08 (0.12)	-0.02 (0.10)	0.04 (0.06)	-0.03 (0.03)	-0.13 (0.12)	**-0.22 (0.18)**
	2000	2.12 (0.17)	0.17 (0.13)	0.18 (0.10)	-0.01 (0.06)	0.02 (0.04)	0.00 (0.11)	**-0.08 (0.19)**
Scenario 2: Jump points of the feature data density								
5	200	**-0.72 (0.17)**	0.34 (0.05)	0.33 (0.04)	0.51 (0.06)	0.60 (0.07)	0.70 (0.07)	0.32 (0.10)
	500	**-1.46 (0.15)**	0.42 (0.05)	0.31 (0.05)	0.44 (0.06)	0.92 (0.09)	1.03 (0.09)	0.59 (0.11)
	1000	**-1.94 (0.13)**	0.48 (0.06)	0.21 (0.05)	0.33 (0.07)	0.99 (0.10)	1.11 (0.10)	0.92 (0.11)
	2000	**-1.87 (0.17)**	0.46 (0.05)	0.26 (0.05)	0.33 (0.06)	1.43 (0.11)	1.53 (0.11)	1.10 (0.11)
10	200	**0.59 (0.12)**	0.08 (0.05)	0.03 (0.04)	0.17 (0.04)	0.09 (0.03)	0.13 (0.03)	0.14 (0.09)
	500	0.44 (0.14)	0.18 (0.04)	0.08 (0.04)	0.05 (0.04)	0.09 (0.04)	0.15 (0.04)	**-0.07 (0.08)**
	1000	0.27 (0.11)	0.18 (0.05)	0.11 (0.04)	0.18 (0.04)	0.29 (0.05)	0.38 (0.05)	**-0.11 (0.07)**
	2000	0.02 (0.13)	0.23 (0.04)	0.11 (0.04)	0.17 (0.04)	0.43 (0.05)	0.49 (0.05)	**-0.12 (0.07)**
20	200	1.92 (0.12)	0.08 (0.04)	0.03 (0.03)	0.02 (0.02)	-0.01 (0.01)	**-0.04 (0.03)**	0.04 (0.07)
	500	1.77 (0.10)	0.14 (0.05)	0.01 (0.03)	-0.02 (0.03)	-0.01 (0.04)	-0.02 (0.02)	**-0.07 (0.07)**
	1000	1.68 (0.13)	0.08 (0.05)	0.02 (0.03)	-0.05 (0.03)	-0.04 (0.02)	-0.03 (0.03)	**-0.09 (0.06)**
	2000	1.50 (0.12)	0.11 (0.05)	0.06 (0.03)	0.08 (0.03)	0.02 (0.02)	0.09 (0.03)	**-0.11 (0.07)**
50	500	2.85 (0.09)	0.16 (0.06)	0.05 (0.04)	-0.01 (0.04)	0.09 (0.03)	0.14 (0.06)	**-0.04 (0.08)**
	1000	2.90 (0.09)	0.20 (0.05)	0.08 (0.04)	-0.03 (0.02)	0.03 (0.02)	0.19 (0.06)	**-0.10 (0.07)**
	2000	2.82 (0.10)	0.15 (0.04)	0.08 (0.03)	-0.01 (0.01)	-0.01 (0.01)	0.10 (0.04)	**-0.12 (0.07)**

In both scenarios, we apply the crystallization learning and existing methods to estimate $\mu(\mathbf{z}) = \sum_{j=1}^{d} c_j g_j(z_j)$ at the target points $\mathbf{z}_1, \ldots, \mathbf{z}_{100}$. We implement the crystallization learning with $L = 3$ for $d = 5, 10$ and $L = 2$ for $d = 20, 50$, and obtain $\hat{\mu}(\mathbf{z}_1), \ldots, \hat{\mu}(\mathbf{z}_{100})$. We implement the k-NN regression with $k = 5, 10, k^*$, where k^* is chosen to be equal to the size of $\mathbb{V}_{\mathbf{z},L}$, and the local linear regression and kernel regression with bandwidth $h = 1$.

Table 2 presents the estimation results averaged over 100 simulations under the two scenarios with different values of n and d. For each d, the estimation accuracy of the crystallization learning improves as the sample size increases, indicating its consistency in



(a) $n = 200$ (b) $n = 500$ (c) $n = 1000$ (d) $n = 2000$

Figure 4: Averaged values of $\log(\text{MSE}_L) - \overline{\log(\text{MSE})}$ $(L = 1, \ldots, 8)$ and $\log(\text{MSE}_{\widetilde{L}}) - \overline{\log(\text{MSE})}$ under different sample sizes (n), where MSE_L is the MSE using the hyperparameter L and $\overline{\log(\text{MSE})} = \sum_{L=1}^{8} \log(\text{MSE}_L)/8$.

estimating $\mu(\cdot)$ in the convex hull $\mathcal{H}(\mathbb{X})$. For lower dimensional cases $(d = 5, 10)$, the crystallization learning generally outperforms the existing methods, demonstrating that our approach is more efficient. For the higher dimensional cases $(d = 20, 50)$, although our method cannot completely dominate the existing ones, the performances of different approaches are comparable. The results under Scenario 2 suggest the robustness of our method to the variations or sudden changes in the feature data density. Overall, the crystallization learning performs well and is stable in estimating $\mu(\cdot)$ at general internal points of $\mathcal{H}(\mathbb{X})$ and jump points of the feature data density.

7.3 Hyperparameter Selection

To examine the data-driven selection procedure for L proposed in Section 3.3, we conduct experiments under Scenario 1. With candidate values $L = 1, \ldots, 8$ and $d = 5$, we simulate 100 training data sets with sample sizes $n = 200, 500, 1000, 2000$ respectively and generate the corresponding sets of target points.

Figure 4 shows the estimation results of our method averaged over 100 simulations under different sample sizes, when using different candidate values of L and the optimally selected $\widetilde{L}$ in (4). It is clear that as the sample size n increases, the optimal value of L, which results in the smallest averaged value of $\log(\text{MSE}_L)$, increases. This is reasonable because a larger n would lead to smaller volumes of simplices in $\mathcal{N}_L(\mathbf{z})$ and thus a larger L is needed for more accurate estimation. In addition, the averaged value of $\log(\text{MSE}_{\widetilde{L}})$ is closer to the smallest averaged value of $\log(\text{MSE}_L)$ when n is larger, corroborating the effectiveness of our LOO-CV procedure.

7.4 Stochastic vs. Deterministic Crystallization Learning

As discussed in Section 4 that the stochastic crystallization learning is a generalization of the deterministic version, we consider both scenarios in Section 7.2 with $n = 200$ and $d = 5$ to compare the stochastic and deterministic crystallization learning. For $k = 1, \ldots, 100$, we implement the stochastic crystallization learning to estimate $\mu(\mathbf{z}_k)$ with $B = 100$ randomly generated sets of simplicies under the energy distribution (5) with the maximal energy loss $\Lambda = 0, 0.1, \ldots, 3.0$. We compare the mean squared error of the stochastic and the deter-

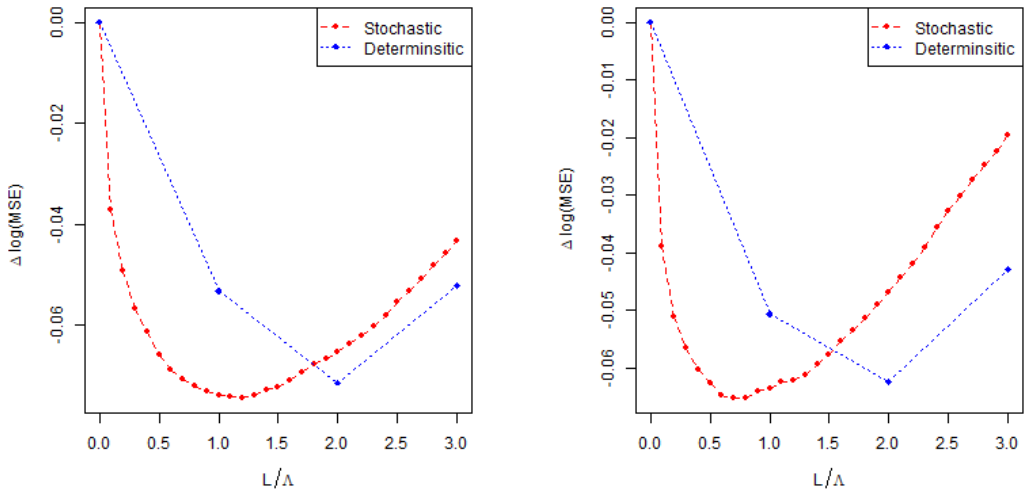


Figure 5: Averaged values of $\Delta\log(\text{MSE}) = \log(\text{MSE}_\Lambda) - \log(\text{MSE}_0)$ of the stochastic crystallization learning ($\Lambda = 0, 0.1, \ldots, 3.0$) and $\Delta\log(\text{MSE}) = \log(\text{MSE}_L) - \log(\text{MSE}_0)$ of the deterministic crystallization learning ($L = 0, 1, 2, 3$) at general internal points of $\mathcal{H}(\mathbb{X})$ (left) and jump points of the feature data density (right).

ministic crystallization learning ($L = 0, 1, 2, 3$) in Figure 5. The stochastic crystallization learning can continually search for the optimal tuning parameter based on the energy distribution while the deterministic one can only choose among the discrete values of L. It is clear that the optimal estimation performance (corresponding to the minimum point) of the stochastic crystallization learning is better than that of the deterministic version.

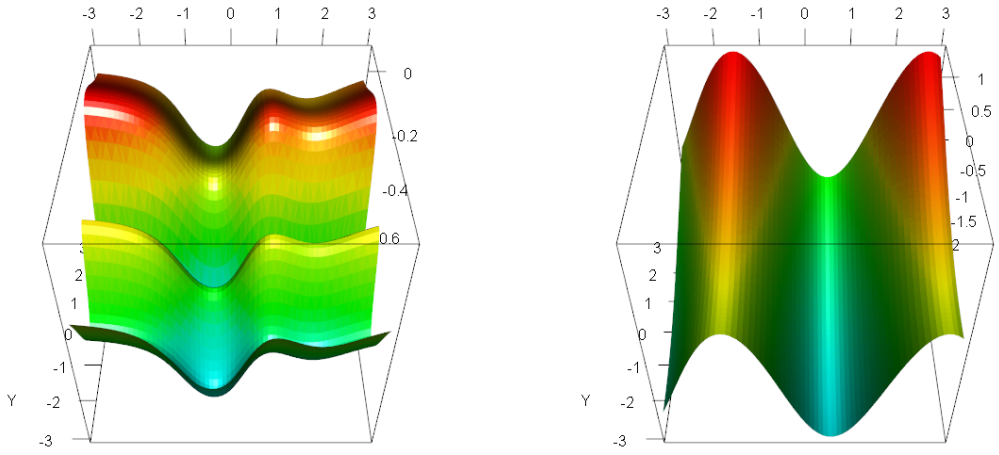
7.5 Hierarchical Crystallization Learning with Multimodal Densities

We conduct experiments on synthetic data sets to evaluate the performance of the hierarchical crystallization learning (HCL) when the feature data points come from a multimodal density $f(\mathbf{x})$. In each data set, feature data points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are independently generated from a Gaussian mixture distribution, $1/C \sum_{c=1}^{C} \text{MVN}(\boldsymbol{\theta}_c, \mathbf{I}_d)$, where C is the number of distributions in the mixture and $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_C$ (representing centers) are i.i.d. random variables from a multivariate normal distribution, $\text{MVN}(\mathbf{0}, 5\mathbf{I}_d)$. We take the sample size of the training data n to be a multiple of C ($n = Cn_C$). Responses $y_1, \ldots, y_n$ are generated from the Gaussian density additive model (8) or the Sine additive model,

$$Y|\mathbf{x} \sim N\Big(\sum_{j=1}^{d} c_j \sin\{b_j(x_j - \eta_j)\}, 1\Big), \tag{9}$$

where $\mathbf{x} = (x_1, \ldots, x_d)^\mathsf{T}$, $c_1, \ldots, c_d \sim N(0,1)$, $b_1, \ldots, b_d \sim N(0,1)$ and $\eta_1, \ldots, \eta_d \sim U[-\pi, \pi]$. For $k = 1, \ldots, 100$, the target point $\mathbf{z}_k$ is generated as $\mathbf{z}_k = \sum_{c=1}^{C} \omega_{ck}\boldsymbol{\theta}_c$, with $(\omega_{1k}, \ldots, \omega_{Ck}) \sim \text{Dirichlet}(1, \ldots, 1)$.

We simulate 100 training data sets and the corresponding sets of target points under different dimensions (d), numbers of mixture components C, average numbers of observations



(a) (b)

Figure 6: Curves of the conditional expectation function $\mu(\cdot)$ when $d = 2$ under (a) the Gaussian density additive model and (b) the Sine additive model.

per component n_C and different additive models. We implement the two-layer hierarchical crystallization learning with $n^* = 2C$ representative points and $L = 2$ to compare its estimation accuracy with the generalized additive model (GAM, Hastie and Tibshirani (1990)). Table 3 summarizes the estimation performance averaged over 100 simulations under different configurations. It is clear that the two-layer hierarchical crystallization learning yields comparable estimation performance to GAM under the Sine additive model (9) and outperforms GAM under the Gaussian density additive model (8) where the conditional expectation function $\mu(\cdot)$ fluctuates greatly as shown in Figure 6. This implies that the hierarchical crystallization learning is able to estimate $\mu(\cdot)$ with larger roughness. Under the same additive model and the same dimension, the advantage of our method over GAM amplifies as the sample size $n = Cn_C$ increases, especially when the number of mixture components in the feature data density $f(\mathbf{x})$ increases. As a result, the hierarchical crystallization learning performs well under the multimodal density $f(\mathbf{x})$.

8. Real Data Analysis

For illustration, we apply the deterministic crystallization learning to several real data sets from the UCI repository. The critical assessment of protein structure prediction (CASP) data set (Betancourt and Skolnick, 2001) contains experimental records on protein structure prediction. The CASP data set includes 45730 records of 9 features, where the response is the root mean squared deviation (RMSD) of the residues. The Concrete data set (Yeh, 1998) consists of 1030 experimental records of concrete compressive strength measurement. We use the content of 7 concrete ingredients and the age of a concrete sample to predict its compressive strength. Parkinson's telemonitoring data set is composed of 5875 voice recordings of 16 biomedical voice measures from 42 patients with early-stage Parkinson's

Table 3: Averaged values of $\log(\text{MSE}_{\text{GAM}}/\text{MSE}_{\text{HCL}})$ and standard deviations in parentheses under different dimensions (d), numbers of mixture components (C), average numbers of observations per component (n_C) and different additive models.

Data Generation Mechanism	$d=5$			$d=10$		
	C	n_C	$\log\left(\frac{\text{MSE}_{\text{GAM}}}{\text{MSE}_{\text{HCL}}}\right)$	C	n_C	$\log\left(\frac{\text{MSE}_{\text{GAM}}}{\text{MSE}_{\text{HCL}}}\right)$
Gaussian	500	50	0.65 (0.37)	500	50	-0.01 (0.18)
Density	500	100	0.71 (0.38)	500	100	0.10 (0.14)
Additive	500	200	0.71 (0.39)	500	200	0.11 (0.17)
Model	1000	100	0.85 (0.47)	1000	100	0.16 (0.16)
	2000	100	1.07 (0.47)	2000	100	0.22 (0.14)
	500	50	0.06 (0.19)	500	50	-0.01 (0.04)
Sine	500	100	0.03 (0.13)	500	100	0.00 (0.04)
Additive	500	200	0.03 (0.10)	500	200	0.00 (0.04)
Model	1000	100	0.07 (0.17)	1000	100	0.01 (0.06)
	2000	100	0.09 (0.18)	2000	100	0.02 (0.05)

disease in a six-month trial. We use these 16 biomedical voice measures to predict the motor and total UPDRS (unified Parkinson's disease rating scale) scores.

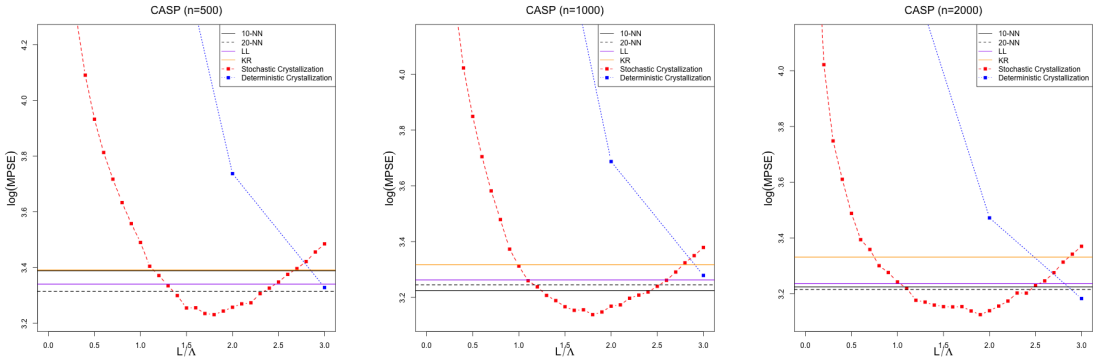
For each data set, we take 100 bootstrap samples without replacement of size n (n = 200, 500, 1000 or 2000) for training and 100 bootstrap samples of size 100 for testing. To eliminate the impact of feature correlations and scales, we standardize the principal components of features in the training set and take them as the feature points $\mathbf{x}_1, \ldots, \mathbf{x}_n$. The same transformation is applied to the features in the testing set to obtain the target points $\mathbf{z}_1, \ldots, \mathbf{z}_{100}$. We take $L = 0, 1, 2, 3$ for deterministic crystallization learning and $\Lambda = 0, 0.1, \ldots, 3$ for stochastic crystallization learning, and implement the k-NN regression with $k = 10, 20$, and the local linear regression and kernel regression with bandwidth $h = 1$. Based on the testing set, we quantify the performance of the method $\mathcal{M}$ by the mean predictive squared error (MPSE),

$$\text{MPSE}_{\mathcal{M}} = \frac{1}{100}\sum_{k=1}^{100}\{\hat{\mu}_{\mathcal{M}}(\mathbf{z}_k) - y_k\}^2,$$

where y_k's are the responses corresponding to $\mathbf{z}_k$'s.

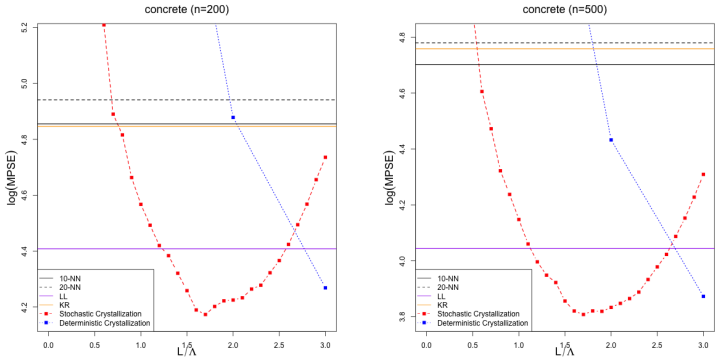
Figures 7-9 present the comparison results averaged over 100 bootstrap samples between our methods and existing ones under different data sets and sizes of the training set (n). Consistent with results in Sections 7.3 and 7.4, the prediction accuracy of stochastic crystallization learning would increase and then decrease as the hyperparameter Λ grows. In most of scenarios, the lowest value of log(MPSE) for stochastic crystallization learning outperforms all existing methods (including its deterministic counterpart), suggesting its superiority and the necessity of hyperparameter selection.

We also apply the hierarchical crystallization learning to the YearPredictionMSD data set from the UCI repository. This data set includes 12 audio features of 515,345 songs, whose release years range from 1922 to 2011. To evaluate the performance of the hierarchical



(a) $n = 500$. (b) $n = 1000$. (c) $n = 2000$.

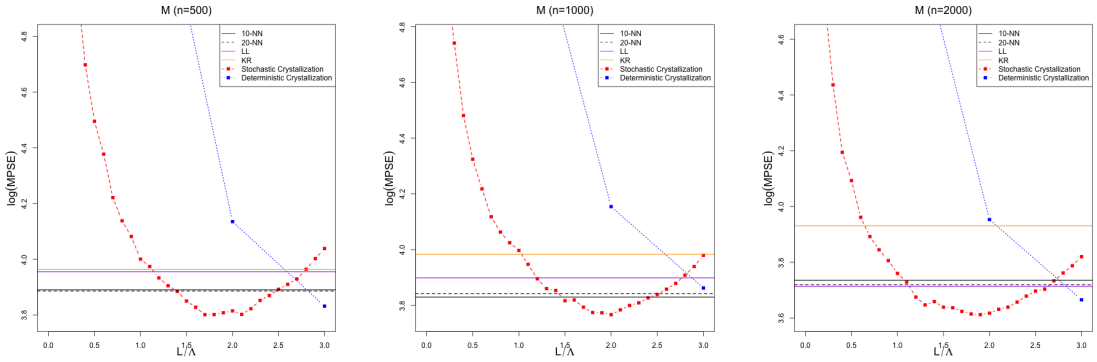
Figure 7: Average values of log(MPSE) of k-NN ($k = 10, 20$), local linear (LL) regression, kernel regression (KR), deterministic crystallization learning and stochastic crystallization learning in estimating $\mu(\cdot)$ under CASP data sets and different sizes of the training set (n).



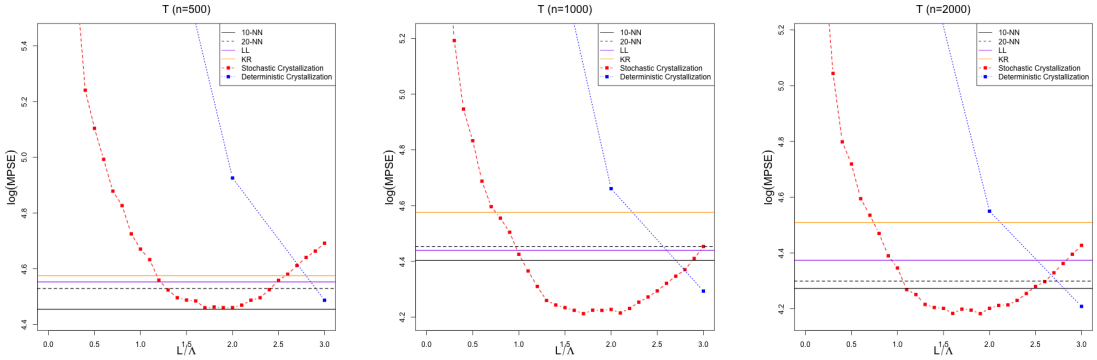
(a) $n = 200$. (b) $n = 200$.

Figure 8: Average values of log(MPSE) of k-NN ($k = 10, 20$), local linear (LL) regression, kernel regression (KR), deterministic crystallization learning and stochastic crystallization learning in estimating $\mu(\cdot)$ under concrete data sets and different sizes of the training set (n).

crystallization learning and GAM in predicting the release year via the audio features of a song, we take 100 bootstrap samples without replacement of size n ($n = n^* \times n_C$ where n^* is the number of representative points and n_C is the average number of observations represented by one representative point) for training and 100 bootstrap samples of size 500 for testing. Table 4 presents the comparison results averaged over 100 bootstrap samples between our method and GAM under different configurations of n^* and n_C. It is clear that our method also outperforms GAM in prediction as the sample size $n = n^* \times n_C$ increases. The advantage of the two-layer crystallization learning amplifies greatly when n^* increases, which is consistent with the simulation results in Section 7.5. Overall, the crystallization learning dominates all the existing methods in most of the cases.



(a) Motor UPDRS and $n = 500$. (b) Motor UPDRS and $n = 1000$. (c) Motor UPDRS and $n = 2000$.



(d) Total UPDRS and $n = 500$. (e) Total UPDRS and $n = 1000$. (f) Total UPDRS and $n = 2000$.

Figure 9: Average values of log(MPSE) of k-NN ($k = 10, 20$), local linear (LL) regression, kernel regression (KR), deterministic crystallization learning and stochastic crystallization learning in estimating $\mu(\cdot)$ under Parkinson's data sets with different responses and different sizes of the training set (n).

Table 4: Averaged values of $\log(\text{MPSE}_{\text{GAM}}/\text{MPSE}_{\text{HCL}})$ and standard deviations in parentheses under different values of the number of representative points (n^*) and the average number of observations represented by one representative point (n_C).

n^*	n_C	$\log\left(\frac{\text{MPSE}_{\text{GAM}}}{\text{MPSE}_{\text{HCL}}}\right)$
500	50	0.08(0.05)
500	100	0.09(0.06)
500	200	0.10(0.04)
1000	100	0.12(0.05)
2000	100	0.20(0.04)

9. Conclusions

The Delaunay triangulation is a powerful tool to partition the feature space in a data-driven way, which generates a mesh of simplicies with the most regularized shapes on the observed feature data points. We incorporate the Delaunay triangulation into the framework of nonparametric regression and develop the crystallization learning procedure. Without the need to triangulate the entire feature space which becomes infeasible for medium- to high-dimensional cases, our method conducts the Delaunay triangulation locally at each specific target point like crystal growth. The conditional expectation $\mu(\mathbf{z})$ at the target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$ is estimated by fitting a local linear model to the data points of the Delaunay simplices identified by the crystallization search. We develop the stochastic crystallization learning that can continually search for the depth of the Delaunay triangulation from the target point. We further propose hierarchical crystallization learning to expedite the Delaunay triangulation process which allows for parallel triangulations at multiple representative centers. Compared to existing nonparametric regression methods, our method is more adaptive to the local geometric structure of the data, which selects the neighbor data points uniformly in all directions and their weighted mean is closer to the target point $\mathbf{z}$. We conduct numerical experiments on synthetic data sets and real data to show that the crystallization learning generally outperforms all the existing methods in both estimation and prediction accuracy, no matter whether the feature data density is unimodal or multimodal.

Acknowledgments

We thank the Action Editor and two referees for their careful reviews that greatly improved our work. Yin's research was supported by the Research Grants Council of Hong Kong (17308321) and the Patrick SC Poon Endowment Fund.

Appendix A. Proof of Theorem 1

For any target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$, the estimated conditional expectation is given by

$$\hat{\mu}(\mathbf{z}) = \hat{\alpha} + \hat{\boldsymbol{\beta}}^{\mathsf{T}}\mathbf{z},$$

where

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}}) = \arg\min_{(\alpha,\boldsymbol{\beta})} \sum_{i=1}^{n} \mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)(y_i - \alpha - \boldsymbol{\beta}^{\mathsf{T}}\mathbf{x}_i)^2,$$

with the weight $\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)$ set to be 0 for all $\mathbf{x}_i \notin \mathbb{V}_{\mathbf{z},L}$. Define

$$\mathbf{H}_{\mathbf{z},L} = \sum_{i=1}^{n} \mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i) \begin{pmatrix} 1 & \mathbf{x}_i^{\mathsf{T}} \\ \mathbf{x}_i & \mathbf{x}_i\mathbf{x}_i^{\mathsf{T}} \end{pmatrix},$$

$$\mathbf{G}_{\mathbf{z},L} = \sum_{i=1}^{n} \mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i) \begin{pmatrix} y_i \\ \mathbf{x}_i y_i \end{pmatrix},$$

and then we have

$$\begin{pmatrix}\hat{\alpha}\\ \hat{\boldsymbol{\beta}}\end{pmatrix} = \mathbf{H}_{\mathbf{z},L}^{-1}\mathbf{G}_{\mathbf{z},L}.$$

As the set of Delaunay simplices $\mathcal{N}_L(\mathbf{z})$ remains unchanged for all $\mathbf{z} \in \mathcal{S}_k$, $\sum_{\mathcal{S}\in\mathcal{N}_L(\mathbf{z})} 1_{\{\mathbf{x}_i\in\mathbb{V}(\mathcal{S})\}}$ and $\sum_{i=1}^n 1_{\{\mathbf{x}_i\in\mathbb{V}_{\mathbf{z},L}\}}$ are fixed. As a result, the normalization term $m_L(\mathbf{z})$ and weight functions $\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)$ $(i = 1,\ldots,n)$ are smooth with respect to $\mathbf{z} \in \mathcal{S}_k$. Given that $\mathbb{X}$ is in general position, matrix $\mathbf{H}_{\mathbf{z},L}$ is full-rank and hence the estimator

$$\hat{\mu}(\mathbf{z}) = \hat{\alpha} + \hat{\boldsymbol{\beta}}^{\mathsf{T}}\mathbf{z} = \mathbf{G}_{\mathbf{z},L}^{\mathsf{T}}\mathbf{H}_{\mathbf{z},L}^{-1}\begin{pmatrix}1\\ \mathbf{z}\end{pmatrix}$$

is smooth with respect to $\mathbf{z} \in \mathcal{S}_k$ $(k = 1,\ldots,m)$. This implies that $\hat{\mu}(\cdot)$ is piecewise smooth in the convex hull $\mathcal{H}(\mathbb{X})$.

Appendix B. Proof of Lemma 1

Let $M = \|\mathbf{z}\|_2$, and let $\mathcal{U}_j(M)$, $j = 1,\ldots,2^d$, denote a series of d-boxes of $\{(-\infty,-M)\cup(M,\infty)\}^d$. Define the event $H_n = \cap_{j=1}^{2^d}\{\mathcal{U}_j(M)\cap\mathbb{X}\neq\varnothing\}$. Because the complement of H_n satisfies

$$\begin{aligned}
\mathbb{P}(H_n^c) &= \mathbb{P}\big(\cup_{j=1}^{2^d}\{\mathcal{U}_j(M)\cap\mathbb{X}=\varnothing\}\big)\\
&\le \sum_{j=1}^{2^d}\mathbb{P}\big(\{\mathcal{U}_j(M)\cap\mathbb{X}=\varnothing\}\big)\\
&= \sum_{j=1}^{2^d}\Big(1-\int_{\mathcal{U}_j(M)} f(\mathbf{x})d\mathbf{x}\Big)^n\\
&\le 2^d\Big(1-\min_{j=1,\ldots,2^d}\int_{\mathcal{U}_j(M)} f(\mathbf{x})d\mathbf{x}\Big)^n\\
&\to 0, \quad \text{as } n\to\infty,
\end{aligned}$$

we have $\mathbb{P}(H_n)\to 1$ as $n\to\infty$. Since H_n implies that $[-M,M]^d\subset\mathcal{H}(\mathbb{X})$, we have

$$\mathbb{P}(\mathbf{z}\in\mathcal{H}(\mathbb{X})) \ge \mathbb{P}([-M,M]^d\subset\mathcal{H}(\mathbb{X})) \ge \mathbb{P}(H_n),$$

and thus $\mathbb{P}(\mathbf{z}\in\mathcal{H}(\mathbb{X}))\to 1$ as $n\to\infty$.

Appendix C. Proof of Theorem 2

Let $\mathcal{B}(\mathbf{z},r)$ be a d-ball with center $\mathbf{z}$ and radius r. Let V_d be the volume of $\mathcal{B}(\mathbf{z},1)$ and let K be a positive constant. Consider the density

$$f_{K,n}(\mathbf{x}) = \begin{cases} c_{1,n}, & \text{for } \mathbf{x}\in\mathcal{B}(\mathbf{z},Kn^{-\rho/d}),\\ f(\mathbf{x})\big(\frac{1-c_{1,n}V_dK^dn^{-\rho}}{1-c_{2,n}}\big), & \text{otherwise,}\end{cases}$$

where

$$c_{1,n} = \min\{f(\mathbf{x});\mathbf{x}\in\mathcal{B}(\mathbf{z},Kn^{-\rho/d})\}, \quad c_{2,n} = \int_{\mathcal{B}(\mathbf{z},Kn^{-\rho/d})} f(\mathbf{x})d\mathbf{x}.$$

By Doeblin's coupling method (Theorem 5.2, Chapter 1 of Lindvall (2002)), there exist random variables $\mathbf{x} \sim f$ and $\mathbf{v} \sim f_{K,n}$ such that

$$\begin{aligned}\mathbb{P}(\mathbf{x} \neq \mathbf{v}) =& \frac{1}{2}\int |f(\mathbf{x}) - f_{K,n}(\mathbf{x})| d\mathbf{x} \\ =& \int_{\mathcal{B}(\mathbf{z}, Kn^{-\rho/d})} \{f(\mathbf{x}) - c_{1,n}\} d\mathbf{x} \\ =& o(n^{-\rho}).\end{aligned} \tag{10}$$

Consider i.i.d. feature points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ from the density f and $\mathbf{v}_1, \ldots, \mathbf{v}_n$ from the density $f_{K,n}$ where $\mathbf{x}_i$ and $\mathbf{v}_i$ satisfy (10) for $i = 1, \ldots, n$. By the definition, $c_{1,n}$ is non-decreasing as $n \to \infty$. As a result, the expected number of feature points $\mathbf{v}_1, \ldots, \mathbf{v}_n$ in the d-ball $\mathcal{B}(\mathbf{z}, Kn^{-\rho/d})$ is

$$n\mathbb{P}(\mathbf{v} \in \mathcal{B}(\mathbf{z}, Kn^{-\rho/d})) = c_{1,n} V_d K^d n^{1-\rho} \to \infty$$

as $n \to \infty$. By (10), the expected number of feature points $\mathbf{x}_1, \ldots, \mathbf{x}_n$ in the d-ball $\mathcal{B}(\mathbf{z}, Kn^{-\rho/d})$ is

$$n\mathbb{P}(\mathbf{x} \in \mathcal{B}(\mathbf{z}, Kn^{-\rho/d})) = n\mathbb{P}(\mathbf{v} \in \mathcal{B}(\mathbf{z}, Kn^{-\rho/d})) + o(n^{1-\rho}),$$

which also goes to infinity as $n \to \infty$ because the first term dominates the right-hand side. Therefore, we have $\mathbb{P}(\mathbb{V}_{\mathbf{z},L} \subset \mathcal{B}(\mathbf{z}, Kn^{-\rho/d})) \to 1$ and $T(\mathbf{z}) = O_p(n^{-\rho/d})$.

Appendix D. Proof of Theorem 3

Because each d-simplex has $d+1$ facets, there are at most $d+1$ neighbor simplices in $\mathcal{DT}(\mathbb{X})$ for each $\mathcal{S}_j$. As a result, there are at most $1+(d^L-1)(d+1)/(d-1)$ simplices in $\mathcal{N}_L(\mathbf{z})$ for any target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$. Note that each pair of neighbor simplices share d vertices, and there are at most $C(d,L) = d+1+(d^L-1)(d+1)/(d-1)$ feature points in $\mathbb{V}_{\mathbf{z},L}$. For convenience, we reindex the data points $\{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$ so that $\mathbb{V}_{\mathbf{z},L} \subset \{\mathbf{x}_1, \ldots, \mathbf{x}_{C(d,L)}\}$.

For any target point $\mathbf{z} \in \mathcal{H}(\mathbb{X})$, the estimated conditional expectation is $\hat{\mu}(\mathbf{z}) = \hat{\alpha} + \hat{\boldsymbol{\beta}}^{\mathsf{T}}\mathbf{z}$, where $\hat{\alpha}$ and $\hat{\boldsymbol{\beta}}$ are obtained via the weighted least squares approach,

$$\begin{aligned}(\hat{\alpha}, \hat{\boldsymbol{\beta}}) =& \arg\min_{(\alpha,\boldsymbol{\beta})} \sum_{\mathbf{x}_i \in \mathbb{V}_{\mathbf{z},L}} \mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)(y_i - \alpha - \boldsymbol{\beta}^{\mathsf{T}}\mathbf{x}_i)^2 \\ =& \arg\min_{(\alpha,\boldsymbol{\beta})} \sum_{i=1}^{C(d,L)} \mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)(y_i - \alpha - \boldsymbol{\beta}^{\mathsf{T}}\mathbf{x}_i)^2,\end{aligned}$$

with $\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i) = 0$ for all $\mathbf{x}_i \notin \mathbb{V}_{\mathbf{z},L}$. Through reparametrization $\mathbf{x}_i^* = \mathbf{x}_i - \mathbf{z}$, $\mathbf{z}^* = \mathbf{0}$ and $\alpha^* = \alpha + \beta^{\mathsf{T}}\mathbf{z}$, it is clear that $\hat{\mu}(\mathbf{z}) = \hat{\alpha}^*$, where

$$(\hat{\alpha}^*, \hat{\boldsymbol{\beta}}) = \arg\min_{(\alpha^*,\boldsymbol{\beta})} \sum_{i=1}^{C(d,L)} \mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)(y_i - \alpha^* - \boldsymbol{\beta}^{\mathsf{T}}\mathbf{x}_i^*)^2, \tag{11}$$

by the invariance of a linear model to linear transformation. Define

$$(\tilde{\alpha}^*, \tilde{\boldsymbol{\beta}}) = \arg\min_{(\alpha^*,\boldsymbol{\beta})} \sum_{i=1}^{C(d,L)} \mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)\{\mu(\mathbf{x}_i) - \alpha^* - \boldsymbol{\beta}^{\mathsf{T}}\mathbf{x}_i^*\}^2. \tag{12}$$

Thus, we have

$$\begin{aligned}\mathbb{E}\big\{\hat{\mu}(\mathbf{z})-\mu(\mathbf{z})\big\}^2 =&\mathbb{E}\big\{\hat{\alpha}^*-\mu(\mathbf{z})\big\}^2\\ =&\mathbb{E}\big\{\hat{\alpha}^*-\tilde{\alpha}^*\big\}^2+\mathbb{E}\big\{\tilde{\alpha}^*-\mu(\mathbf{z})\big\}^2+2\mathbb{E}\big\{\hat{\alpha}^*-\tilde{\alpha}^*\big\}\big\{\tilde{\alpha}^*-\mu(\mathbf{z})\big\}.\end{aligned} \tag{13}$$

From Theorem 2, we have $\mathrm{w}^q_{\mathbf{z},L}(\mathbf{x}_i)\|\mathbf{x}_i^*\|_2^\rho \to 0$ for $i=1,\ldots,C(d,L)$, $\rho>0$ and $q\geq 0$. By the differentiability of the conditional expectation function $\mu(\cdot)$, we have $(\tilde{\alpha}^*,\tilde{\boldsymbol{\beta}})\to(\mu(\mathbf{z}),\nabla\mu(\mathbf{z}))$ and thus the second term of (13) goes to zero as $n\to\infty$. By the Cauchy—Schwarz inequality, the third term of (13) satisfies

$$\Big|\mathbb{E}\big\{\hat{\alpha}^*-\tilde{\alpha}^*\big\}\big\{\tilde{\alpha}^*-\mu(\mathbf{z})\big\}\Big|\leq\sqrt{\mathbb{E}\big\{\hat{\alpha}^*-\tilde{\alpha}^*\big\}^2}\sqrt{\mathbb{E}\big\{\tilde{\alpha}^*-\mu(\mathbf{z})\big\}^2}\to 0$$

as $n\to\infty$. Define the random matrix

$$\mathbf{A}_n=\left\{\sum_{i=1}^{C(d,L)}\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)\begin{pmatrix}1 & [\mathbf{x}_i^*]^\intercal\\ \mathbf{x}_i^* & \mathbf{x}_i^*[\mathbf{x}_i^*]^\intercal\end{pmatrix}\right\}^{-1}=\begin{pmatrix}a_n^{(1,1)} & [\mathbf{a}_n^{(1,-1)}]^\intercal\\ \mathbf{a}_n^{(1,-1)} & \mathbf{A}_n^{(-1,-1)}\end{pmatrix}.$$

Under model (1), we have

$$\begin{aligned}\begin{pmatrix}\hat{\alpha}^*\\ \hat{\boldsymbol{\beta}}\end{pmatrix}&=\sum_{i=1}^{C(d,L)}y_i\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)\mathbf{A}_n\begin{pmatrix}1\\ \mathbf{x}_i^*\end{pmatrix}\\ &=\sum_{i=1}^{C(d,L)}\mu(\mathbf{x}_i)\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)\mathbf{A}_n\begin{pmatrix}1\\ \mathbf{x}_i^*\end{pmatrix}+\sum_{i=1}^{C(d,L)}\epsilon_i\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)\mathbf{A}_n\begin{pmatrix}1\\ \mathbf{x}_i^*\end{pmatrix}\\ &=\begin{pmatrix}\tilde{\alpha}^*\\ \tilde{\boldsymbol{\beta}}\end{pmatrix}+\sum_{i=1}^{C(d,L)}\epsilon_i\mathrm{w}_{\mathbf{z},L}(\mathbf{x}_i)\mathbf{A}_n\begin{pmatrix}1\\ \mathbf{x}_i^*\end{pmatrix}.\end{aligned}$$

Let $E(\epsilon_i^2)=\sigma^2<\infty$, and then we have

$$\begin{aligned}\mathbb{E}\big(\hat{\alpha}^*-\tilde{\alpha}^*\big)^2&=\sigma^2\mathbb{E}\left\{\sum_{i=1}^{C(d,L)}\mathrm{w}^2_{\mathbf{z},L}(\mathbf{x}_i)(a_n^{(1,1)}+[\mathbf{a}_n^{(1,-1)}]^\intercal\mathbf{x}_i^*)^2\right\}\\ &=\sigma^2\times\{1/C(d,L)+o(1/C(d,L))\}\\ &\to 0,\quad \text{as } n\to\infty \text{ and } L\to\infty.\end{aligned}$$

As a result, (13) goes to 0 and the consistency is shown.

References

N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

Jacqueline K. Benedetti. On the nonparametric estimation of regression functions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(2):248–253, 1977.

Marcos R. Betancourt and Jeffrey Skolnick. Universal similarity measure for comparing protein structures. *Biopolymers*, 59(5):305–309, 2001.

Tyler H. Chang, Layne T. Watson, Thomas C. H. Lux, Jon Bernard, Bo Li, Li Xu, Godmar Back, Ali R. Butt, Kirk W. Cameron, and Yili Hong. Predicting system performance by interpolation using a high-dimensional Delaunay triangulation. In *Proceedings of the High Performance Computing Symposium*, HPC'18, 2018a.

Tyler H. Chang, Layne T. Watson, Thomas C. H. Lux, Bo Li, Li Xu, Ali R. Butt, Kirk W. Cameron, and Yili Hong. A polynomial time algorithm for multivariate interpolation in arbitrary dimension via the Delaunay triangulation. In *Proceedings of the ACMSE 2018 Conference on - ACMSE 18*. ACM Press, 2018b.

Tyler H. Chang, Layne T. Watson, Thomas C. H. Lux, Ali R. Butt, Kirk W. Cameron, and Yili Hong. Algorithm 1012: DELAUNAYSPARSE: Interpolation via a sparse subset of the Delaunay triangulation in medium to high dimensions. *ACM Transactions on Mathematical Software*, 46(4):1–20, 2020.

William S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74(368):829–836, 1979.

William S. Cleveland and Susan J. Devlin. Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83 (403):596–610, 1988.

T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

M. Daszykowski, B. Walczak, and D.L. Massart. Representative subset selection. *Analytica Chimica Acta*, 468(1):91–103, 2002.

Mark de Berg, Otfried Cheong, Mark van Kreveld, and Mark Overmars. Delaunay triangulations. In *Computational Geometry*, pages 191–218. Springer Science & Business Media, 2008.

B. Delaunay. Sur la sphère vide. *Bulletin de l'Académie des Sciences de l'URSS. Classe des sciences mathématiques et na*, 6:793–800, 1934.

J. Fan and I. Gijbels. *Local Polynomial Modelling and Its Applications*. Boca Raton: Chapman and Hall, 1996.

Jiaqi Gu and Guosheng Yin. Crystallization learning with the Delaunay triangulation. In *The 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3854–3863. PMLR, 2021.

W. Härdle and T. Gasser. Robust non-parametric function fitting. *Journal of the Royal Statistical Society. Series B (Methodological)*, 46(1):42–51, 1984.

Trevor Hastie and Robert Tibshirani. *Generalized Additive Models*. Wiley Online Library, 1990.

Matthias Hein. Robust nonparametric regression with metric-space valued output. In *Advances in Neural Information Processing Systems*, volume 22, 2009.

Torgny Lindvall. *Lectures on the coupling method*. Dover Books on Mathematics. Dover, Mineola, NY, 2002.

Yehong Liu and Guosheng Yin. The Delaunay triangulation learner and its ensembles. *Computational Statistics & Data Analysis*, 152:107030, 2020.

E. A. Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1): 141–142, 1964.

M. B. Priestley and M. T. Chao. Non-parametric function fitting. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(3):385–392, 1972.

Zong-Feng Qi, Yong-Dao Zhou, and Kai-Tai Fang. Representative points for location-biased datasets. *Communications in Statistics - Simulation and Computation*, 48(2):458–471, 2017.

Daniel M Roy and Yee Whye Teh. The mondrian process. In *Advances in Neural Information Processing Systems*, volume 21, 2009.

R. Sibson. Locally equiangular triangulations. *The Computer Journal*, 21(3):243–245, 1978.

Erwin Stampfer and Ernst Stadlober. Methods for estimating principal points. *Communications in Statistics - Simulation and Computation*, 31(2):261–277, 2002.

Charles J. Stone. Consistent nonparametric regression. *The Annals of Statistics*, 5(4): 595–620, 1977.

Geoffrey S. Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, 26(4):359–372, 1964.

I.-C. Yeh. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research*, 28(12):1797–1808, 1998.

Hao Zhu, Bin Guo, Ke Zou, Yongfu Li, Ka-Veng Yuen, Lyudmila Mihaylova, and Henry Leung. A review of point set registration: From pairwise registration to groupwise registration. *Sensors*, 19(5):1191, 2019.