SOLVING PARTIAL DIFFERENTIAL EQUATIONS ON POINT CLOUDS*

JIAN LIANG† AND HONGKAI ZHAO‡

Abstract. In this paper we present a general framework for solving partial differential equations on manifolds represented by meshless points, i.e., point clouds, without parametrization or connection information. Our method is based on a local approximation of the manifold, such as using least squares, in a local intrinsic coordinate system constructed by local principal component analysis (PCA) using K-nearest neighbors (KNN). Once the local reconstruction is available, differential operators on the manifold can be approximated discretely. The framework extends to manifolds of any dimension. The complexity of our method scales well with the total number of points and the true dimension of the manifold (not the embedded dimension). The numerical algorithms, error analysis, and test examples are presented.

Key words. Manifold, gradient, Laplace-Beltrami operator, principle component analysis, moving least squares, constrained quadratic optimization, time dependent PDE, upwind scheme, semi-Lagrangian method, eigenvalue problem

AMS subject classifications.

1. Introduction.

Point cloud data is defined simply as a set of points with no specific ordering and connection. In 2D or 3D, points are defined by their X, Y and X, Y, Z coordinates respectively. Point cloud is the most basic and intrinsic way for sampling and representation of geometric objects or information in high dimensions. For examples, 3D point cloud can be easily obtained through scanner for shape modeling, images can be thought as points in high dimensions, etc. In this work we present a general framework for solving partial differential equations (PDE) on manifolds represented by point clouds. The motivation comes from many problems in science and engineering such as surfactant distribution along a moving interface in fluids [1], surface diffusion in sintering [2], in biology [3, 4], in image processing [5, 6, 7, 8] and etc. Another important application is in data science, where the task of visualizing, extracting information, analyzing and inferring underlying structure from data samples is ubiquitous. In many cases, point cloud data resides or is believed to reside on or near a low-dimensional manifold in a much higher dimensional ambient space. Although there are useful tools, such as the principal component analysis (PCA), to provide local dimension and linear structure approximation, it is very challenging to extract global information and structure in general. Mathematically and computationally one can obtain a lot of intrinsic information, such as manifold learning, by studying the behavior of differential equations, such as heat equation, or eigenvalue problem for differential operators, such as Laplace-Beltrami operator, on manifolds [9, 10, 11, 12, 13, 14, 15, 16, 17].

There are different approaches to solve PDEs on manifold depending on how the manifold is represented. For a nicely parametrized manifold the natural way is to express differential operators in the parameter space and then discretize the resulting equations. [18] gives a decent tutorial and survey of methods for parameterizing surfaces. However, it can be very difficult to construct a global parameterization for

*This work was partially supported by grants ONR grant N00014-11-1-0602, ARO/MURI W911NF-07-1-0185, and NGA NURI HM1582-10-1-0012.

†Department of Mathematics, University of California, Irvine, CA, US (jianl@uci.edu).

‡Department of Mathematics, University of California, Irvine, CA, US (zhao@math.uci.edu).

complicated surfaces and especially for high dimensional manifolds. For a nicely triangulated manifold, one can discretize a PDE directly on the tirangulation, which can be effective for certain classes of equations, such as for elliptic equations using standard finite element method. However, this approach could also have a few difficulties. First, to get a nice triangulation can be difficult if not impossible when the dimension of the manifold is three or higher. Second, it is difficult to define high order geometric quantities, such as normal and curvature, accurately based on piecewise linear approximation. These are discussed in [19, 20]. To avoid the difficulty of parametrization or triangulation, an alternative is to use implicit representation, e.g., using level set representation, which embeds the manifold as well as the differential equation defined on the manifold into the ambient space. Then discretize the extended differential equation in the ambient space using a Cartesian grid [20, 21, 22]. Similar in nature, Ruuth et al. [23] proposed closest point method to solve PDEs on surfaces, which uses a closest point representation of the underlying surface and embeds the surface differential equations to the ambient space and then solve it using finite difference method on a uniform Cartesian grid in a narrow tube around the manifold in the ambient space. First, laying down a grid in a high dimensional ambient space and performing computation on it can be very expensive, even though the true dimension of the manifold may be low, e.g. manifold with high co-dimension. Also it is difficult to incorporate adaptivity using Cartesian grid in general.

Since point cloud is the simplest and intrinsic way for sampling and representation of manifold in practice, we propose a framework of solving PDEs directly on point clouds without using parametrization, triangulation or grid, which can be difficult to construct and may introduce artifacts. The key idea is that one can define differential operators on manifold by local construction of the manifold, which is first proposed in [24]. In another word, once we can construct a function as well as the manifold locally in a common reference coordinate, we can differentiate the function with respect to the metric of the underlying manifold simply using chain rule. So in our method, we only need to use the K-nearest neighbor (KNN) points to define a local intrinsic coordinate system using PCA and to construct the manifold and function locally using least squares. Our method can handle manifold of any dimensions or co-dimensions in the same way and the complexity scales well with the total number of sample points and the real dimension of the manifold.

The paper is organized as follows. Section 2 gives some brief mathematical formulations about differentiation on manifolds and moving least squares (MLS) method, which will be used throughout the paper. In Section 3, we describe our approach to approximate differential operators. Briefly, our approach consists of three main parts, construction of local coordinate system, local approximation of surface and local approximation of function. In Section 4, we use our approximated differential operators to solve PDEs directly on point clouds. How to handle boundary conditions for open surfaces is also discussed. Numerical experiments in 3D and higher dimension spaces are presented in Section 5. In Section 6, we give a brief summary. Finally, Appendix A gives error estimates for MLS and Appendix B gives the connection between MLS and a constrained quadratic optimization problem.

2. Mathematical formulations. Before explaining our idea of solving PDEs on point clouds based on local approximation, we first briefly introduce some mathematical background and notation of differential geometry about derivatives on manifolds and the MLS problem. They will be used throughout the rest of the paper.

2.1. Derivatives on manifolds. For simplicity, we only consider two-dimensional manifold in $\mathbb{R}^3$ and only briefly introduce definition of gradient and Laplace-Beltrami (surface Laplacian). We refer [25] to readers for more details and definitions of the derivatives for other dimensional manifolds.

Let $\mathcal{M} \subset \mathbb{R}^3$ be a two-dimensional manifold and suppose it is parameterized by (x_1, x_2). We can write the manifold as $\mathbf{\Gamma}(x_1, x_2) \doteq (X(x_1, x_2), Y(x_1, x_2), Z(x_1, x_2))$, the metric tensor $G = [g_{ij}]$ is given by $g_{ij} = \langle \mathbf{\Gamma}_{x_i}, \mathbf{\Gamma}_{x_j} \rangle$, where $\mathbf{\Gamma}_{x_1} = (X_{x_1}, Y_{x_1}, Z_{x_1})$, and $\mathbf{\Gamma}_{x_2} = (X_{x_2}, Y_{x_2}, Z_{x_2})$. The tangent space $T_{\mathbf{x}}\mathcal{M}$ at $\mathbf{x} \in \mathcal{M}$ is spanned by $\mathbf{\Gamma}_{x_1}(\mathbf{x})$ and $\mathbf{\Gamma}_{x_2}(\mathbf{x})$.

Let $f \in C^2(\mathcal{M})$. Under this parameterization, one has the gradient operator, given by (see [25] page 102)

$$\nabla_{\mathcal{M}} f = [\mathbf{\Gamma}_{x_1}, \mathbf{\Gamma}_{x_2}]\, G^{-1} \nabla f = \left(g^{11} \frac{\partial f}{\partial x_1} + g^{12} \frac{\partial f}{\partial x_2} \right) \mathbf{\Gamma}_{x_1} + \left(g^{21} \frac{\partial f}{\partial x_1} + g^{22} \frac{\partial f}{\partial x_2} \right) \mathbf{\Gamma}_{x_2} \tag{2.1}$$

where g^{ij} are the components of G^{-1}, the inverse of the metric tensor G. And the gradient $\nabla_{\mathcal{M}} f(\mathbf{x})$ is a vector in the tangent space $T_{\mathbf{x}}\mathcal{M}$.

The Laplace-Beltrami operator (surface Laplace) can be written as

$$\Delta_{\mathcal{M}} f = \sum_{i,j=1}^{2} \frac{1}{\sqrt{g}} \frac{\partial}{\partial x_i} \left(\sqrt{g} g^{ij} \frac{\partial f}{\partial x_j} \right) \tag{2.2}$$

where $g = \det(G)$.

Both the gradient $\nabla_{\mathcal{M}} f$ and Laplace-Beltrami operator $\Delta_{\mathcal{M}} f$ are geometric intrinsic, though the expression (2.1) and (2.2) depend on a local surface parameterization. Keep this in mind, we will use this important property to introduce local parameterization while the computation based on local parameterization is still geometric intrinsic.

2.2. Moving least squares. Moving least squares (MLS) is a method of approximating functions by linear combination of certain basis functions, such as polynomials, from a set of point samples using (weighted) least square formulation with the origin positioned at a location depending on the point samples (moving). MLS is a powerful tool for function approximation from scattered points. Compared to standard interpolation, which can be viewed as a special case of least square approximation where the degree of freedom matches the number of constraint from data, although using more data points, the key advantages of MLS is its robustness with respect to perturbations and extra degree of freedom that may be utilized to incorporate other desired structures. We briefly introduce MLS problem and its solution here. A short introduction to MLS method can be found in [26]. We point out the error estimates for MLS at the end of this sub-section. We refer the readers to [27] and [28] for the approximation power of MLS. Error analysis is provided in Appendix A. Also we recast the MLS problem as a constrained quadratic optimization problem in Appendix B, from which we can add additional constraint to incorporate desired properties of the continuous operator during discretization, such as requiring diagonal dominant which corresponds to maximal principle, in the resulting discretized Laplace-Beltrami operator matrix.

We use polynomials for our MLS, like explained in [26]. Given K points located at positions $\mathbf{x}_k$ around point $\overline{\mathbf{x}}$ in $\mathbb{R}^d$ where $k \in [1, \cdots, K]$. We wish to obtain a local d-dimensional degree m polynomial $f_{\overline{\mathbf{x}}}(\mathbf{x})$ that approximates the given scaler values

f_k at points $\mathbf{x}_k$. We can compute such $f_{\overline{\mathbf{x}}}(\mathbf{x})$ by minimizing the following weighted sum:

$$\min_{f_{\overline{\mathbf{x}}} \in \Pi_m^d} \sum_{k=1}^{K} w(\|\mathbf{x}_k - \overline{\mathbf{x}}\|) \|f_{\overline{\mathbf{x}}}(\mathbf{x}_k) - f_k\|^2 \tag{2.3}$$

where Π_m^d is the space of polynomials of total degree m in d-dimensions and $w(\cdot)$ is some positive weight function. The above sum is weighted by $w(d_k)$ where $d_k = \|\mathbf{x}_k - \overline{\mathbf{x}}\|$ are the Euclidian distances between $\overline{\mathbf{x}}$ and the position of data point $\mathbf{x}_k$.

Since Π_m^d is shift invariant, we suggest taking the basis functions to be the monomials shifted to $\overline{\mathbf{x}}$, which makes computations easy and clear. $f_{\overline{\mathbf{x}}}$ can then be written as

$$f_{\overline{\mathbf{x}}}(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \mathbf{c}(\overline{\mathbf{x}}) = \mathbf{b}(\mathbf{x}) \cdot \mathbf{c}(\overline{\mathbf{x}}) \tag{2.4}$$

where $\mathbf{b}(\mathbf{x})$ is the polynomial basis vector, and $\mathbf{c}(\overline{\mathbf{x}}) = [c_1, c_2, \cdots, c_I]^T$ is the coefficient vector to be determined by (2.3), $I = \frac{(d+m)!}{d!m!}$ is the number of basis in Π_m^d. For example, $\mathbf{b}(\mathbf{x}) = \left[1, x_1 - \overline{x}_1, x_2 - \overline{x}_2, (x_1 - \overline{x}_1)^2, (x_1 - \overline{x}_1)(x_2 - \overline{x}_2), (x_2 - \overline{x}_2)^2\right]^T$ for Π_2^2, bivariate ($d = 2$) degree 2 polynomial space. By taking partial derivatives with respect to the unknown coefficients $c_1, c_2, \cdots, c_I$, we obtain a linear system of equations and we can compute $\mathbf{c}(\overline{\mathbf{x}})$ as

$$\mathbf{c}(\overline{\mathbf{x}}) = \left[\sum_{k=1}^{K} w_k \mathbf{b}(\mathbf{x}_k)\mathbf{b}(\mathbf{x}_k)^T\right]^{-1} \sum_{k=1}^{K} w_k \mathbf{b}(\mathbf{x}_k) f_k \tag{2.5}$$

where $w_k = w(\|\mathbf{x}_k - \overline{\mathbf{x}}\|)$ (details can be found in [26]). Suppose $b_i = (\mathbf{x} - \overline{\mathbf{x}})^{\alpha_i}$ with multi-index $\alpha_i = (a_1, a_2, \cdots, a_d)$, that is $b_i = (x_1 - \overline{x}_1)^{a_1} \cdots (x_d - \overline{x}_d)^{a_d}$, then we can use $\alpha_i! c_i$ to approximate $D^{\alpha_i} f(\overline{\mathbf{x}})$. A key point is that all the c_i and hence the approximation of $D^{\alpha_i} f(\overline{\mathbf{x}})$ is a linear operation on f_k if the neighboring points $\mathbf{x}_k$ are given. The local approximation error is of order $O(h^{m+1-|\alpha_i|})$, where $h = \max_k \|\mathbf{x}_k - \overline{\mathbf{x}}\|$. Actually one order higher super-convergence can be observed often in practice due to error cancellation when points distribution has some symmetry. Both error estimates and super-convergence are presented Appendix A.

3. Approximation of differential operators on point clouds. We explain our approach of using local construction to approximate differential operators on manifold, such as gradient $\nabla_{\mathcal{M}} f$ and Laplace-Beltrami operator $\Delta_{\mathcal{M}} f$, numerically for point clouds in this section. The main issue is how to compute derivatives with respect to local metric. The key idea is simply based on chain rule, i.e., computing the metric of the manifold and the derivatives of a function with respect to a reference coordinates locally. Our method consists of three main parts. First we use PCA on KNN to estimate local dimensions and construct a local coordinate system for both tangent and normal spaces. In this coordinate system, the manifold can be parameterized in terms of the tangent space. We use MLS to construct the manifold and compute the metric tensor $g_{ij} = \langle \mathbf{\Gamma}_{x_i}, \mathbf{\Gamma}_{x_j} \rangle$ in this coordinate system. Finally, we can approximate differential operators on the manifold by using MLS approximation of a function and its derivatives in the same coordinate system and then using formulas like (2.1), (2.2). Again the key point is that the operation is linear in terms of the values of the function at the neighboring points. Hence we construct a finite difference scheme directly on point clouds. Actually we use the example of Laplace-Beltrami

operator to show that we can design the finite difference scheme either based on MLS or using a constrained quadratic optimization approach which guarantees both the accuracy and that the resulting discretized Laplace-Beltrami operator satisfies the discrete maximal principle. Also we will discuss how to incorporate boundary conditions (both Dirichlet and Neumann) for open manifolds in Section 4. For simplicity, we use two-dimensional manifold in $\mathbb{R}^3$ to illustrate our approach, it is straightforward to generalize the approach to higher dimensions.

3.1. Local coordinate system. First we use local PCA to construct a local coordinate system which is a direct sum of tangent space and normal space at each point [29, 30]. Then we parametrize the manifold locally on the tangent space and then use equations such as (2.1) and (2.2) to compute differential operators on manifold. For example for a point cloud $P = \{\mathbf{p}_i | i = 1, 2, \cdots, N\}$ sampled from a smooth two-dimensional manifold $\mathcal{M}$ in $\mathbb{R}^3$, denote the indices set of the K-nearest-neighbors (KNN) of each point $\mathbf{p}_i \in P$ by $N(i)$. Using the covariance matrix P_i of $N(i)$, defined by:

$$P_i = \sum_{k \in N(i)} (\mathbf{p}_k - \mathbf{c}_i)^T (\mathbf{p}_k - \mathbf{c}_i) \tag{3.1}$$

we can estimate some local geometric information near $\mathbf{p}_i$. Here, $\mathbf{c}_i$ is the local barycenter $\mathbf{c}_i = \frac{1}{K} \sum_{k \in N(i)} \mathbf{p}_k$. The eigenvectors $(\mathbf{e}_1^i, \mathbf{e}_2^i, \mathbf{e}_3^i)$ of P_i form an orthogonal frame associated with eigenvalues $(\lambda_1^i, \lambda_2^i, \lambda_3^i)$ with $\lambda_1^i \geq \lambda_2^i \geq \lambda_3^i \geq 0$. The relative size of the eigenvalues can reveal the true dimension of the manifold locally. The eigenvectors corresponding to the small eigenvalues form the basis of the normal space. In our example, for point cloud sampled from a smooth two-dimensional manifold $\mathcal{M}$ in $\mathbb{R}^3$ and if the sampling rate is fine enough to resolve the local features, we have $\lambda_1^i \geq \lambda_2^i \gg \lambda_3^i \geq 0$. Hence $(\mathbf{e}_1^i, \mathbf{e}_2^i)$ form the basis of the local tangent plane. In another word, the plane that goes through $\mathbf{c}_i$ and orthogonal to $\mathbf{e}_3^i$ fits the KNN best in terms of least squares. In our computation, $\mathbf{p}_i$ is always taken as the origin of the local coordinate system. In this way, we have defined a local coordinate system $\langle \mathbf{p}_i; \mathbf{e}_1^i, \mathbf{e}_2^i, \mathbf{e}_3^i \rangle$ at each point in P. KNN of $\mathbf{p}_i$ have local coordinates (x_k^i, y_k^i, z_k^i), which will be used for surface and function approximations.

3.2. Local approximation of manifold and the metric tensor. To compute the differentiation on manifold one needs the metric tensor. Here we use the MLS method (Section 2.2) to approximate the manifold in the local coordinate system constructed above and then compute the local metric tensor at each point. Other approximation methods can also be used in our approach.

In principle, one can use MLS to construct polynomials of any degree as long as enough KNN are used. To compute the Laplace-Beltrami operator, which is a second order differential operator, it suffices to construct quadratic polynomial through the KNN at each point. Again, assuming the surface is dimension two, once a local coordinate system for a point $\mathbf{p}_i$ is constructed, a local degree two bivariate polynomial $z_i(x, y)$ is approximated by minimizing the following weighted sum:

$$\sum_{k \in N(i)} w(\|\mathbf{p}_k - \mathbf{p}_i\|) \left(z_i(x_k^i, y_k^i) - z_k^i \right)^2 \tag{3.2}$$

where (x_k^i, y_k^i, z_k^i) are local coordinates of point $\mathbf{p}_k$ in the KNN of $\mathbf{p}_i$ and $w(\cdot)$ is some positive weight function. $\mathbf{\Gamma}_i = (x, y, z_i(x, y))$ is thus a smooth representation of the surface near the point $\mathbf{p}_i$ under local coordinate system $\langle \mathbf{p}_i; \mathbf{e}_1^i, \mathbf{e}_2^i, \mathbf{e}_3^i \rangle$.

Assume $z_i(x,y) = a_1 + a_2x + a_3y + a_4x^2 + a_5xy + a_6y^2$. The two tangent vector basis are given by $\mathbf{\Gamma}_x(\mathbf{p}_i) = (1, 0, \frac{\partial z_i}{\partial x}(0,0)) = (1, 0, a_2)$ and $\mathbf{\Gamma}_y(\mathbf{p}_i) = (0, 1, \frac{\partial z_i}{\partial y}(0,0)) = (0, 1, a_3)$, notice that we take $\mathbf{p}_i$ as origin of the local coordinate system. Under such parameterization, the metric tensor $G(\mathbf{x})$ is some function of coefficients of z_i, so is its inverse $G^{-1}(\mathbf{x})$. For instance, $g^{11}(\mathbf{x}) = \frac{1+a^2}{1+a^2+b^2}$, where $a = a_2 + 2a_4x + a_5y$, $b = a_3 + a_5x + 2a_6y$ and (x, y, z) are local coordinates of $\mathbf{x}$ and $g^{11}(\mathbf{p}_i) = \frac{1+a_3^2}{1+a_2^2+a_3^2}$.

The gradient in local coordinate system is

$$\nabla_{\mathcal{M}} u_s(\mathbf{p}_i) = (g^{11}\frac{\partial u_s}{\partial x}(\mathbf{p}_i) + g^{12}\frac{\partial u_s}{\partial y}(\mathbf{p}_i))\mathbf{\Gamma}_x(\mathbf{p}_i) + (g^{21}\frac{\partial u_s}{\partial x}(\mathbf{p}_i) + g^{22}\frac{\partial u_s}{\partial y}(\mathbf{p}_i))\mathbf{\Gamma}_y(\mathbf{p}_i). \tag{3.3}$$

To simplify notation, we use g^{ij} instead of $g^{ij}(\mathbf{p}_i)$. For Laplace-Beltrami operator, we can write $\Delta_{\mathcal{M}} u_s(\mathbf{p}_i)$ as

$$\Delta_{\mathcal{M}} u_s(\mathbf{p}_i) = A_1\frac{\partial u_s}{\partial x}(\mathbf{p}_i) + A_2\frac{\partial u_s}{\partial y}(\mathbf{p}_i) + A_3\frac{\partial^2 u_s}{\partial x^2}(\mathbf{p}_i) + A_4\frac{\partial^2 u_s}{\partial x\partial y}(\mathbf{p}_i) + A_5\frac{\partial^2 u_s}{\partial y^2}(\mathbf{p}_i) \tag{3.4}$$

where A_j's are obtained by expanding and simplifying equation (2.2) and they only depend on coefficients of local surface approximation $a_2, a_3, \cdots, a_6$, which only depend on relative locations of the KNN. Notice that terms in (3.3) g^{ij}, $\mathbf{\Gamma}_x$ and $\mathbf{\Gamma}_y$ and terms in (3.4) A_j's are computed based on local approximation of the surface/manifold and they all are independent of the involved function u_s.

3.3. Local approximation of function and its derivatives on manifold. Now that the manifold is locally approximated, which can be viewed as a local parametrization in tangent space, one can use MLS to locally approximate a function u_s, which is defined on the manifold, and its derivatives in the local coordinate system. Again we locally approximate u_s in the local coordinate system $\langle \mathbf{p}_i; \mathbf{e}_1^i, \mathbf{e}_2^i, \mathbf{e}_3^i\rangle$ using a degree two bivariate polynomial $u_i(x,y)$ near point $\mathbf{p}_i$ by minimizing the following weighted least squares sum:

$$\sum_{k\in N(i)} w(\|\mathbf{p}_k - \mathbf{p}_i\|)\left(u_i(x_k^i, y_k^i) - u_s(\mathbf{p}_k)\right)^2 \tag{3.5}$$

Assume $u_i(x,y) = b_1 + b_2x + b_3y + b_4x^2 + b_5xy + b_6y^2$, the surface gradient in local coordinate system (3.3) becomes

$$\nabla_{\mathcal{M}} u_s(\mathbf{p}_i) = (g^{11}b_2 + g^{12}b_3)\mathbf{\Gamma}_x(\mathbf{p}_i) + (g^{21}b_2 + g^{22}b_3)\mathbf{\Gamma}_y(\mathbf{p}_i) \tag{3.6}$$

which is equivalent to

$$\nabla_{\mathcal{M}} u_s(\mathbf{p}_i) = (g^{11}b_2 + g^{12}b_3)(\mathbf{e}_1^i + a_2\mathbf{e}_3^i) + (g^{21}b_2 + g^{22}b_3)(\mathbf{e}_2^i + a_3\mathbf{e}_3^i) \tag{3.7}$$

Similarly for Laplace-Beltrami operator, we can write (3.4) as

$$\Delta_{\mathcal{M}} u_s(\mathbf{p}_i) = A_1b_2 + A_2b_3 + A_3(2b_4) + A_4b_5 + A_5(2b_6) \tag{3.8}$$

where A_i's are obtained by expanding and simplifying equation (2.2) and they only depend on coefficients of local surface approximate $a_2, a_3, \cdots, a_6$. Using Lemma A.2 in Appendix A, one can easily see the following Lemma.

LEMMA 3.1. *Let h denote the smallest radius such that the K-nearest neighbors lie in a ball of such radius and centered at $\overline{\mathbf{x}}$, i.e., $h = \max_k \|\mathbf{x}_k - \overline{\mathbf{x}}\|$ then the approximation (3.7) and (3.8), which are based on MLS using polynomials of degree m,*

are of order $O(h^m)$ and $O(h^{m-1})$ respectively.

We point out that super-convergence can be achieved by MLS in certain situations. More details can be found in Section 5 and Appendix A.

In general, given a point cloud $P = \{\mathbf{p}_i | i = 1, 2, \cdots, N\}$ with N points sampled from a two-dimensional manifold in $\mathbb{R}^3$. We can represent a function u_s defined on the manifold as a N-dimensional vector $U = [u_1, u_2, \cdots, u_N]^T$ with $u_i = u_s(\mathbf{p}_i)$. All the g^{ij}, $\mathbf{e}_j^i$ and A_j are computed from local manifold approximation $\mathbf{\Gamma}_i$. For b_j, when using (2.5) to solve (3.5), we can write b_j computed from (3.5) as a linear function of U as shown in (2.5), i.e., $b_j = B_j U$ where B_j is some N-dimensional row vector, and only points in the KNN of $\mathbf{p}_i$ have nonzero coefficients. After all, we can discretize surface gradient and Laplace-Beltrami operator as

$$\nabla_{\mathcal{M}} u_s = [V_1 \ V_2 \ V_3] U \tag{3.9}$$

and

$$\Delta_{\mathcal{M}} u_s = MU \tag{3.10}$$

where V_1, V_2, V_3 and M are $N \times N$ matrices and these matrices are sparse.

REMARK 1. *Although the weight function in MLS does not affect the approximation order as shown in Lemma A.2 in Appendix A, however the stability can be affected. For example, two popular weight functions used in MLS are Wendland function defined as*

$$w(d) = \left(1 - \frac{d}{D}\right)^4 \left(\frac{4d}{D} + 1\right), \tag{3.11}$$

which is well defined on the interval $d \in [0, D]$ and $w(0) = 1$, $w(D) = 0$, $w'(D) = 0$ and $w''(D) = 0$, and inverse of squared distance function $1/(d^2 + \varepsilon^2)$. These two weight functions work fine for pretty uniform point clouds but may be unstable for non-uniform ones. On the other hand, the special weight function

$$w(d) = \begin{cases} 1 & \text{if } d = 0 \\ 1/K & \text{if } d \neq 0 \end{cases} \tag{3.12}$$

introduced in [17] works for more general data set.

REMARK 2. *All the above formulations extend naturally to embedded manifolds with high-codimensions. Using local PCA one can find the true dimension of the manifold as well as the tangent and normal spaces. Local approximation of the manifold and function defined on the manifold can be parametrized by the tangent space. For example, assume a manifold of n dimension is embedded in $\mathbb{R}^d$, $n < d$. Local PCA provides a local coordinate system $(x_1, \ldots, x_n, x_{n+1}, \ldots, x_d)$, where $x_1, \ldots, x_n$ belong to the tangent space and $x_{n+1}, \ldots, x_d$ belong to the normal space. One can construct a local approximation of the manifold $(x_1, \ldots, x_n, x_{n+1}(x_1, \ldots, x_n), \ldots, x_d(x_1, \ldots, x_n))$ as well as a local approximation of a function f defined on the manifold as $f(x_1, \ldots, x_n)$ in the same coordinate system using MLS. Local metric $g_{ij} = \langle \mathbf{\Gamma}_{x_i}, \mathbf{\Gamma}_{x_j} \rangle, i, j = 1, \ldots, n$ as well as differentiation of functions on the manifold can be approximated as before. The complexity of our method scales well with the true dimension of the manifold rather than the embedded dimension. We will show examples of solving PDEs on high co-dimensional manifold in Section 5.*

3.4. Discretizations preserving structures of the PDE. One issue of using MLS to approximate a function and its derivatives is that it only takes into account local approximation error. When solving a PDE, one also needs to preserve certain property of the differential operator after discretization. For examples, for an elliptic or a parabolic PDE, such as Laplace equation or heat equation, maximum principle is a very important property that should be preserved after discretization, while for hyperbolic PDE, information propagates along characteristics and hence upwind scheme should be used. Below we give two concrete examples for designing such discretizations.

3.4.1. Maximum principle preserving discretized Laplace-Beltrami operator. Here we present a constrained optimization approach to approximate a function and its derivatives which can utilize the flexibility that there are more degrees of freedom than constraints to enforce desired property in the discretization. We use Laplace-Beltrami operator as an example to design a discretization that preserves the maximal principle using constrained optimization. As shown in Appendix B, estimates of partial derivatives of u_s can also be obtained from the following constrained quadratic optimization problem: finding coefficients vector $\mathbf{a}_l \in \mathbb{R}^K$ for the approximation $\widehat{b}_l = \mathbf{a}_l^T U_N$ for $l = 2, 3, \cdots, 6$ where $\widehat{b}_2$, $\widehat{b}_3$, $\widehat{b}_4$, $\widehat{b}_5$, $\widehat{b}_6$ are estimates of $\frac{\partial u_s}{\partial x}(\mathbf{p}_i)$, $\frac{\partial u_s}{\partial y}(\mathbf{p}_i)$, $\frac{\partial^2 u_s}{\partial x^2}(\mathbf{p}_i)$, $\frac{\partial^2 u_s}{\partial x \partial y}(\mathbf{p}_i)$, $\frac{\partial^2 u_s}{\partial y^2}(\mathbf{p}_i)$ respectively and $U_N = [u_s(\mathbf{x}_k)]^T, k \in N(i)$ by minimizing the quadratic form

$$Q = \frac{1}{2} \sum_{k \in N(i)} \frac{\left(a_k^{(l)}\right)^2}{w(\|\mathbf{p}_k - \mathbf{p}_i\|)} \tag{3.13}$$

subject to the linear constraints

$$\sum_{k \in N(i)} a_k^{(l)} b_j(\mathbf{x}_k) = r_j^{(l)} \qquad \text{for } j = 1, 2, \cdots, I \tag{3.14}$$

where $\mathbf{r}_l = \left[r_1^{(l)}, r_2^{(l)}, \cdots, r_I^{(l)}\right]^T = \alpha_l! \mathbf{e}_l$ and $\mathbf{e}_l$ is the l-th standard basis for $\mathbb{R}^I$. For degree two bivariate polynomial, $I = 6$, $\mathbf{b}(\mathbf{x}) = \left[1, x, y, x^2, xy, y^2\right]^T$, $\alpha_1 = (0, 0)$, $\alpha_2 = (1, 0)$ $\alpha_3 = (0, 1)$, $\alpha_4 = (2, 0)$, $\alpha_5 = (1, 1)$ and $\alpha_6 = (0, 2)$. We call (3.14) as "consistency constraint" (more details can be found in Appendix B).

To obtain estimates of $\frac{\partial u_s}{\partial x}(\mathbf{p}_i)$, $\frac{\partial u_s}{\partial y}(\mathbf{p}_i)$, $\frac{\partial^2 u_s}{\partial x^2}(\mathbf{p}_i)$, $\frac{\partial^2 u_s}{\partial x \partial y}(\mathbf{p}_i)$ and $\frac{\partial^2 u_s}{\partial y^2}(\mathbf{p}_i)$, we can either solve the MLS problem or solve the above constrained quadratic optimization problem for $l = 2, 3, \cdots, 6$. And the estimates from MLS are the same as those from the above constrained quadratic optimization (more details in Appendix B). However, the constrained quadratic optimization formulation allows us to add additional constraint to enforce extra property in the discretization. For example, one desirable property for the discretized Laplace-Beltrami operator matrix M in (3.10) is diagonal dominant, which is the discretized version of the maximum principle for Laplace-Beltrami operator. Following [31] and section 6.5 of [32], we impose a sign restriction. Using $\widehat{b}_l = \mathbf{a}_l^T U_N$, expression (3.8) becomes

$$\Delta_M u_s(\mathbf{p}_i) = (A_1 \mathbf{a}_2 + A_2 \mathbf{a}_3 + A_3 \mathbf{a}_4 + A_4 \mathbf{a}_5 + A_5 \mathbf{a}_6)^T U_N \tag{3.15}$$

Suppose $N(i) = \{i_1, i_2, \cdots, i_K\}$ and without loss of generality $i = i_1$, that is $\mathbf{p}_i = \mathbf{p}_{i_1}$. Compare with expression (3.10), we know only K entries in each row of M are non-

zero. And they are

$$M_{ii_k} = A_1 a_k^{(2)} + A_2 a_k^{(3)} + A_3 a_k^{(4)} + A_4 a_k^{(5)} + A_5 a_k^{(6)} \tag{3.16}$$

We impose the following sign restriction constraint.

$$M_{ii_k} < 0 \quad \text{if } i = i_k, \qquad M_{ii_k} \geq 0 \quad \text{if } i \neq i_k \tag{3.17}$$

Notice that $b_1(\mathbf{x}) = 1$ implies $\sum_k a_k^{(l)} = 0$, so $\sum_k M_{ii_k} = 0$. The resulting matrix M will then be diagonal dominant and hence positive semi-definite. We call constraint (3.17) "diagonal dominant constraint".

By solving the quadratic minimization problem (3.13) with "consistency constraint" (3.14) and "diagonal dominant constraint" (3.17), we obtain a discretized Laplace-Beltrami operator matrix M that is positive semi-definite and satisfies the discrete maximum principle. In general, (3.13)+(3.14)+(3.17) does not have a closed form solution. One can use "*quadprog*" function in Matlab to solve the quadratic optimization problem with the 2 systems of constraints.

REMARK 3. *Notice that the starting point of MLS approach is to provide a local function approximation, while that of the constrained quadratic optimization approach is to provide estimates of a function and its derivatives evaluated at a point. On the other hand both methods produce the local Taylor expansion of the function. We give an exact relation between these two constructions in Appendix B. Although two methods can achieve the same accuracy, the constrained quadratic optimization approach is more computationally expensive. However it preserves maximum principle in discrete solution and is more robust for more challenging examples, such as when the data points are non-uniform.*

3.4.2. Semi-Lagrangian scheme for hyperbolic PDE. We design an upwind scheme for hyperbolic PDE on point clouds using the semi-Lagrangian approach. To compute the solution at a given point $\mathbf{p}$ at time t^{n+1}, one can first find the characteristic along the manifold that starts at $\mathbf{p}$ and goes backward by Δt to $\widehat{\mathbf{p}}$, and then solve an ODE along the characteristic from $\widehat{\mathbf{p}}$ to $\mathbf{p}$ with initial value given at $\widehat{\mathbf{p}}$. To find the characteristic along the manifold may not be easy itself. One can use a local first order linear approximation and then project $\widehat{\mathbf{p}}$ back to the manifold. To get the value at $\widehat{\mathbf{p}}$ at t^n, we first find the KNN of $\widehat{\mathbf{p}}$, and then use local PCA through the KNN to define a local coordinate system. The value at $\widehat{\mathbf{p}}$ at t^n is then interpolated by the values at the KNN at t^n using MLS in this local coordinate system. For a concrete example and details, please see advection equation on a torus in Section 5.1.2.

4. Solving PDEs on point clouds. Once we know how to discretize differential operators on point clouds, we can solve PDEs on point clouds. Here we use two types of PDEs as examples. One is the time dependent PDE of the following form

$$\frac{\partial u_s}{\partial t} = F(\mathbf{x}, u_s, \nabla_{\mathcal{M}} u_s, \Delta_{\mathcal{M}} u_s) \tag{4.1}$$

the other is the eigenvalue problem for the Laplace-Beltrami operator,

$$-\Delta_{\mathcal{M}} \phi = \lambda \phi \tag{4.2}$$

For the time dependent problem (4.1), we use simple time discretization, such as forward Euler,

$$U^{n+1} = U^n + \Delta t \cdot F(\mathbf{x}, U^n, [V_1\ V_2\ V_3]U^n, MU^n) \tag{4.3}$$

More sophisticated time discretization to relax time step constraint, such as Crank-Nicholson scheme, was designed in [24], which will not be discussed in this paper. Solving (4.2) is equivalent to solving the eigenvalue problem for the discretized matrix operator. A few examples based on MLS and some applications in computer vision were reported in [17]. We will show a few more examples, especially using the discretization based on the constrained optimization approach and for manifolds with boundary.

4.1. Boundary condition. For closed manifold, there is no boundary condition involved. For open manifold one has to deal with boundary conditions at the manifold boundary. A reasonable assumption is that we know what are boundary points and interior points in the point cloud. We use the time dependent PDE (4.1) as an example to demonstrate our approach. Without loss of generality, we assume $\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_L$ are boundary points, that is we have L boundary points and $N - L$ interior points. And we can write U as $U = [U_1\ U_2]^T$ where $U_1 = [u_1, u_2, \cdots, u_L]^T$ and $U_2 = [u_{L+1}, u_{L+2}, \cdots, u_N]$.

4.1.1. Dirichlet boundary condition. Dirichlet boundary condition is easy to implement. At interior points we simply discretize the PDE using KNN as before. Whenever boundary points are involved in the KNN, their prescribed values are used. As an example, the time dependent PDE (4.1) with Dirichlet boundary condition has the following form

$$\begin{cases} \frac{\partial u_s}{\partial t} = F(\mathbf{x}, u_s, \nabla_{\mathcal{M}} u_s, \Delta_{\mathcal{M}} u_s) & \mathbf{x} \in \Omega \\ u_s(\mathbf{x}, t) = g(\mathbf{x}, t) & \mathbf{x} \in \partial\Omega \end{cases} \tag{4.4}$$

where g is a given function. We can discretize $g(\mathbf{x}, t)$ as a L-dimensional vector $G(\mathbf{x}, t)$, since we know $\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_L$ are boundary points. Using (4.3), we have the following solver

$$\begin{cases} U_1^{n+1} = U_1^n + \Delta t \cdot F(\mathbf{x}, U^n, [V_1\ V_2\ V_3]U^n, MU^n) \\ U_2^{n+1} = G(\mathbf{x}, t^{n+1}) \end{cases} \tag{4.5}$$

4.1.2. Neumann boundary condition. Implementation of Neumann boundary condition is a little bit more complicated since geometric information of the boundary, i.e., the normal, is involved. However, we can use the same procedure as before to construct the boundary and approximate its normal in a local coordinate system and then set up a discretized equation at a boundary point using the boundary condition. Again we use the PDE (4.1) with Neumann boundary condition on two-dimensional manifold in $\mathbb{R}^3$ as an example,

$$\begin{cases} \frac{\partial u_s}{\partial t} = F(\mathbf{x}, u_s, \nabla_{\mathcal{M}} u_s, \Delta_{\mathcal{M}} u_s) & \mathbf{x} \in \Omega \\ \frac{\partial u_s}{\partial \mathbf{n}} = g(\mathbf{x}, t) & \mathbf{x} \in \partial\Omega \end{cases} \tag{4.6}$$

where $\mathbf{n}$ denotes the normal to the boundary $\partial\Omega$ in the tangent plane of Ω and g is a given function.

At interior points we simply discretize the PDE using KNN as before. At the boundary, which is a one-dimensional curve for a two-dimensional manifold, we need to approximate $\frac{\partial u_s}{\partial \mathbf{n}} = \mathbf{n} \cdot \nabla_{\mathcal{M}} u_s$. At each boundary point, we first find its KNN from the point cloud and approximate $\nabla_{\mathcal{M}} u_s$ as before. Next we approximate the normal to the boundary in the tangent plane, $\mathbf{n}$. Since we already have the normal to the surface in the first step when we approximate $\nabla_{\mathcal{M}} u_s$, we only need to approximate the

tangent direction of the boundary curve. To do this we have to construct the boundary curve in a similar fashion as we do for the surface. We find KNN of a boundary point $\mathbf{p}_i \in \{\mathbf{p}_i | i = 1, 2, \cdots, L\}$ where all these KNN points belong to boundary points and this K can be different from that for surface approximation. Then we construct a local coordinate system $\langle \mathbf{p}_i; \widetilde{\mathbf{e}}_1^i, \widetilde{\mathbf{e}}_2^i, \widetilde{\mathbf{e}}_3^i \rangle$ using PCA from these boundary KNN. MLS is used to approximate the boundary curve near $\mathbf{p}_i$ as $\mathbf{r}_i(x) = (x, y_i(x), z_i(x))$ in this local coordinate system $\langle \mathbf{p}_i; \widetilde{\mathbf{e}}_1^i, \widetilde{\mathbf{e}}_2^i, \widetilde{\mathbf{e}}_3^i \rangle$. From the MLS construction we can compute the tangent direction of the boundary curve. Finally the normal $\mathbf{n}$ is defined as the direction orthogonal to both the normal of the surface and the tangent of the boundary. Use the above procedure, Neumann boundary condition can be disretized as $AU = G(\mathbf{x}, t)$ where A is a $L \times N$ matrix and $G(\mathbf{x}, t)$ is a L-dimensional vector function. We can write A as $[A_1\ A_2]$ where A_1 and A_2 are $L \times L$ and $L \times (N - L)$ matrices. For the time dependent problem (4.6), we use the following scheme

$$\begin{cases} U_1^{n+1} = U_1^n + \Delta t \cdot F(\mathbf{x}, U^n, [V_1\ V_2\ V_3]U^n, MU^n) \\ A_2 U_2^{n+1} = G(\mathbf{x}, t^{n+1}) - A_1 U_1^{n+1} \end{cases} \tag{4.7}$$

5. Numerical experiments. In this section we present numerical examples in three and higher dimensions. In particular, convergence studies are carried out for examples where exact solutions are known. Some of the examples used here come from [22], [23] and [16]. For simplicity, we use degree 2 polynomial for all examples. For PDEs that contain Laplace-Beltrami operator, we use both MLS and quadratic optimization approach with "consistency constraint" and "diagonal dominant constraint" to discretize the PDEs. At the end of this section, an experiment of a benchmark point cloud, the Stanford bunny, is presented.

5.1. Time dependent PDEs.

5.1.1. Diffusion equation on sphere. Consider first diffusion $\frac{\partial u_s}{\partial t} = \Delta_{\mathcal{M}} u_s$ on the unit sphere. The unit sphere is parameterized as $\mathbf{\Gamma} = (\cos\theta \sin\phi, \sin\theta \sin\phi, \cos\phi)$, the Laplace-Beltrami operator can be written as $\Delta_{\mathcal{M}} = \frac{1}{\sin^2\phi}\frac{\partial^2}{\partial\theta^2} + \frac{\cos\phi}{\sin\phi}\frac{\partial}{\partial\phi} + \frac{\partial^2}{\partial\phi^2}$. With initial condition

$$u_s(\theta, \phi, 0) = \cos\phi,$$

the solution at any time t is given by

$$u_s(\theta, \phi, t) = e^{-2t}\cos\phi.$$

We apply our method to discretize the diffusion operator directly on point clouds and use forward Euler (4.3) with time step $\Delta t = 0.1\Delta x^2$ where $\Delta x = \min_{i,j} \|\mathbf{p}_i - \mathbf{p}_j\|$ and constant $K = 15$ (number of KNN to be used). We calculate the max-norm relative errors of the numerical solution at the final time $t = 1$ for several uniformly distributed point clouds with different sample sizes. We calculate numerical solutions for both MLS approach and constrained quadratic optimization approach, also we use 2 popular weight functions, one is the Wendland function (3.11), the other is the inverse of squared distance. For simplicity, we use $D = 1.1 \max_k d_k$ for all experiments and $\varepsilon = 10^{-3}$ for $1/(d^2 + \varepsilon^2)$. These results are reported in Table 5.1.

This convergence test indicates a first-order convergence in the value of u_s with respect to sample size (N) and second-order with respect to space (h) for both MLS approach and constrained quadratic optimization approach. Also, we observe the same convergence orders for both Wendland and inverse of squared distance weight

sample size	MLS [Wendland]			MLS $[1/(d^2+\varepsilon^2)]$		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
1002	1.54e-02			2.35e-02		
1962	7.88e-03	1.00	1.99	1.19e-02	1.01	2.03
4002	3.90e-03	0.99	1.97	5.83e-03	1.00	2.00
7842	2.01e-03	0.99	1.97	2.96e-03	1.01	2.02
16002	9.97e-04	0.98	1.97	1.45e-03	1.00	2.00
sample size	Constraint [Wendland]			Constraint $[1/(d^2+\varepsilon^2)]$		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
1002	1.54e-02			2.35e-02		
1962	7.88e-03	1.00	1.99	1.19e-02	1.01	2.03
4002	3.90e-03	0.99	1.97	5.83e-03	1.00	2.00
7842	2.01e-03	0.99	1.97	2.96e-03	1.01	2.02
16002	9.97e-04	0.98	1.97	1.45e-03	1.00	2.00

TABLE 5.1
Max-norm relative errors for the diffusion equation on a unit sphere.

functions, which is also observed for the rest of the examples. For simplicity, we only present Wendlend weight function results in the rest experiments. Notice that the convergence is about one order higher than we expect in Lemma 3.1 due to symmetry of the sphere and uniform sampling of the point clouds, which agrees with the super-convergence result (Lemma A.3) in Appendix A. We will see similar results in later experiments. An interesting observation is that the difference between MLS approach and constrained quadratic optimization approach is very little for this simple shape with uniform sampling. In other words, the discretized Laplace-Beltrami matrix from MLS approach is close to be diagonal dominant in this case.

5.1.2. Advection equation on a torus. We next solve an advection equation

$$\frac{\partial u_s}{\partial t} + \frac{\partial u_s}{\partial \theta} + 2\frac{\partial u_s}{\partial \phi} = 0 \tag{5.1}$$

on a torus given by $\mathbf{\Gamma} = ((R + r\cos\phi)\cos\theta, (R + r\cos\phi)\sin\theta, r\sin\phi)$ with $R = 1$ and $r = 0.5$. The surface gradient can be written as $\nabla_{\mathcal{M}} = \frac{1}{(R+r\cos\phi)^2}\mathbf{\Gamma}_\theta \frac{\partial}{\partial\theta} + \frac{1}{r^2}\mathbf{\Gamma}_\phi \frac{\partial}{\partial\phi}$ and (5.1) can be written as

$$\frac{\partial u_s}{\partial t} + \mathbf{v}\cdot\nabla_{\mathcal{M}} u_s = 0 \qquad \text{with } \mathbf{v} = \mathbf{\Gamma}_\theta + 2\mathbf{\Gamma}_\phi \tag{5.2}$$

We consider the initial profile

$$u_s(\theta, \phi, 0) = \cos\theta + \sin\phi$$

Our computation measures the max-norm of the difference between our computed solution and the exact analytical solution

$$u_s(\theta, \phi, t) = \cos(\theta - t) + \sin(\phi - 2t)$$

Due to periodicity and smoothness of the solution, we can use MLS approximation using all KNN for the discretization for this hyperbolic problem. Time-stepping is carried out using forward Euler with 2 different time step-sizes $\Delta t = 0.01\Delta x$ and $\Delta t = 0.1\Delta x$ where $\Delta x = \min_{i,j} \|\mathbf{p}_i - \mathbf{p}_j\|$ and constant $K = 15$ (number of KNN to be used). Relative errors in the result at the final time $t = 1$ are computed on the torus using the max-norm for a variety of sample sizes. These results are reported in

sample size	$\Delta t = 0.01\Delta x$ MLS [Wendland]			$\Delta t = 0.1\Delta x$ MLS [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
1035	9.54e-03			1.60e-02		
1800	4.96e-03	1.18	2.36	1.10e-02	0.68	1.35
4050	2.07e-03	1.08	2.16	7.12e-03	0.54	1.07
7200	1.15e-03	1.02	2.04	5.27e-03	0.52	1.05
16200	5.49e-04	0.91	1.82	3.48e-03	0.51	1.02

TABLE 5.2
Max-norm relative errors for the advection equation on a torus.

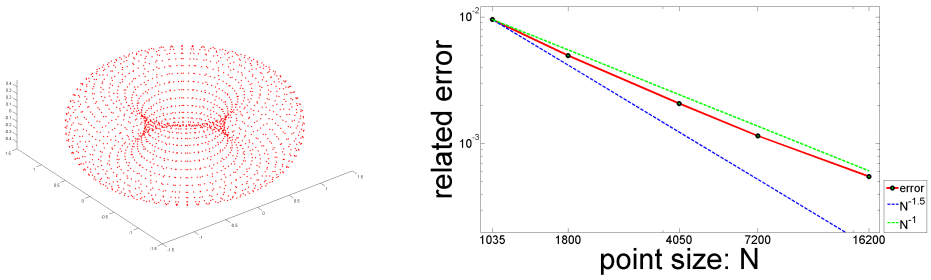

FIG. 5.1. *Point cloud sampled from torus with 1800 points. Convergence plot of max-norm relative errors (for $\Delta t = 0.01\Delta x$) using MLS approach with Wendland function as weight function for the advection equation on a torus.*

Table 5.2 and a convergence plot of the relative errors (for $\Delta t = 0.01\Delta x$) is shown in Figure 5.1.

Since we use forward Euler, a first order discretization in time, if Δt is small enough ($0.01\Delta x$), first-order convergence in the value of u_s with respect to sample size (N) and second-order in space (h) are observed for this first order PDE, which agrees with Lemma 3.1; if Δt is large ($0.1\Delta x$), the error from time discretization will dominate. As a result only first-order convergence in space (h) is observed. Standard high order discretization in time, which will not be discussed in this paper, can be used.

Now we apply the semi-Lagrangian method based on upwind scheme explained in Section 3.4. Given the discretized time step Δt, from the PDE we can approximate $u^{n+1}(\mathbf{p}_i)$ by $u^n(\widehat{\mathbf{p}}_i)$ where $\widehat{\mathbf{p}}_i = \mathbf{p}_i - \Delta t \cdot \mathbf{v}(\mathbf{p}_i)$. We then use MLS as explained in Section 3.4.2 to approximate $u^n(\widehat{\mathbf{p}}_i)$. We first find KNN of $\widehat{\mathbf{p}}_i$ in the point cloud, then use local PCA through the KNN to compute a local coordinate system. Under this coordinate system, we locally approximate the surface and the function u^n. Notice that $\widehat{\mathbf{p}}_i$ may not lie on the surface. Suppose $\widehat{\mathbf{p}}_i$ has coordinate (x_0, y_0, z_0) in the local coordinate system. Since the local MLS approximation $\widetilde{u}^n$ of u^n only depends on the first two local coordinates, so we use $\widetilde{u}^n(x0, y0)$ as the estimate of $u^n(\widehat{\mathbf{p}}_i)$ and $u^{n+1}(\mathbf{p}_i) \approx \widetilde{u}^n(x0, y0)$. We test our upwind scheme using the same data and same parameters. These results are reported in Table 5.3.

The tests show better numerical results than using MLS approximation. Upwind scheme will be more stable in general. The accuracy can be further improved if characteristics can be computed more accurately on the manifold to get $\widehat{\mathbf{p}}_i$.

5.1.3. Diffusion on a filament in 3D. We consider the diffusion equation on an open helical curve in $\mathbb{R}^3$ parametrized by

$$(x, y, z) = (\sin(2\pi s), \cos(2\pi s), 2s - 1),$$

sample size	$\Delta t = 0.01\Delta x$ upwind			$\Delta t = 0.1\Delta x$ upwind		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
1035	2.36e-02			1.12e-02		
1800	1.06e-02	1.45	2.89	8.10e-03	0.59	1.17
4050	3.25e-03	1.46	2.92	5.28e-03	0.53	1.06
7200	1.46e-03	1.39	2.78	3.95e-03	0.50	1.01
16200	5.28e-04	1.25	2.51	2.64e-03	0.50	0.99

TABLE 5.3
Max-norm relative errors for the advection equation on a torus using upwind scheme.

where $0 \leq s \leq 1$ and homogeneous Neumann condition is imposed at endpoint $s = 0$ and homogeneous Dirichlet condition is imposed at endpoint $s = 1$.

The initial condition is given by

$$u_s(s, 0) = \cos(0.5\pi s).$$

We measure the max-norm of the difference between our computed solution and the analytical solution

$$u_s(s, t) = \exp\left(-(\frac{\pi}{2L})^2 t\right) \cos(0.5\pi s)$$

where $L = 2\sqrt{1 + \pi^2}$ is the length of the helix.

We apply our approach and time-stepping is carried out using forward Euler with time step-size $\Delta t = 0.1\Delta x^2$ where $\Delta x = \min_{i,j} \|\mathbf{p}_i - \mathbf{p}_j\|$ and constant $K = 15$ (number of KNN to be used). We use both MLS approach and constrained quadratic optimization approach. Relative errors in the results at the final time $t = 1$ are computed on the filament using the max-norm for a variety of sample sizes. These results are reported in Table 5.4 and a convergence plot of the relative errors for MLS approach with Wendland function as weight function is shown in Figure 5.2.

sample size	MLS [Wendland]			Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
90	1.00e-02			2.21e-03		
180	2.06e-03	2.28	2.28	5.06e-04	2.13	2.13
360	3.47e-04	2.57	2.57	1.21e-04	2.06	2.06
720	5.21e-05	2.74	2.74	2.96e-05	2.03	2.03
1440	6.85e-06	2.93	2.93	7.31e-06	2.02	2.02

TABLE 5.4
Max-norm relative errors for the diffusion equation on a helix with boundary conditions.

This convergence test indicates at least a second-order (super-)convergence with respect to both sample size (N) and space (h) for both MLS approach and constrained quadratic optimization approach for this 1D curve.

5.1.4. Reaction diffusion system on sphere. We solve a reaction diffusion system to get a spiral wave evolving on the point cloud of a unit sphere. The simulated system in this experiment is the well-known Fitzhugh-Nagumo equations [33]

$$\frac{\partial u_s}{\partial t} = (a - u_s)(u_s - 1)u_s - v_s + \nu\Delta_M u_s \tag{5.3}$$

$$\frac{\partial v_s}{\partial t} = \epsilon(\beta u_s - v_s) \tag{5.4}$$

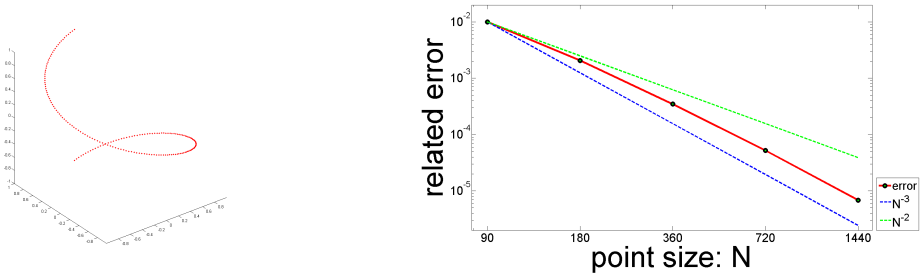

FIG. 5.2. *Point cloud sampled from a filament with 180 points. Convergence plot of max-norm relative errors using MLS approach with Wendland function as weight function for the diffusion equation on a helix with boundary conditions.*

where u_s is the excitation variable, $\epsilon = 0.01$, $a = 0.1$, $\beta = 0.5$ and $\nu = 0.0001$. We set our initial conditions according to

$$(u_s, v_s) = \begin{cases} (1,0) & \text{if } x > 0, y > 0, z > 0, \\ (0,1) & \text{if } x < 0, y > 0, z > 0, \\ (0,0) & \text{otherwise.} \end{cases} \tag{5.5}$$

to obtain an attractive spiral wave. This simulation uses forward Euler with $\Delta t = 0.01$ and $K = 15$ (number of KNN to be used). In this example, we use 16002 points sampled from unit sphere and $\Delta x = 0.0227$. The results of excitation variable u_s are displayed at time t = 400, 450, 500 and 550 in Figure 5.3. The solution is displayed on a triangulated mesh for better visual effect.

5.1.5. Diffusion equation on a flat 2-tours in 4D. We consider diffusion on a flat 2-tours T^2, which is a two dimensional manifold embedded in $\mathbb{R}^4$ parameterized as $\mathbf{\Gamma} = (\cos\alpha, \sin\alpha, \cos\beta, \sin\beta)$, with $\alpha, \beta \in [0, 2\pi]$. The Laplace-Beltrami operator can be written as $\Delta_{\mathcal{M}} = \frac{\partial^2}{\partial\alpha^2} + \frac{\partial^2}{\partial\beta^2}$.

For an initial profile

$$u_s(\alpha, \beta, 0) = \sin(\alpha) + \sin(2\beta)$$

the solution at any time t is given by

$$u_s(\alpha, \beta, t) = e^{-t}\sin(\alpha) + e^{-4t}\sin(2\beta).$$

We apply our method and use forward Euler with $\Delta t = 0.1\Delta x^2$ where $\Delta x = \min_{i,j} \|\mathbf{p}_i - \mathbf{p}_j\|$ and constant $K = 15$ (number of KNN to be used). We calculate numerical solutions at the final time $t = 1$ for several point clouds with different sample sizes. We use both MLS approach and constrained quadratic optimization approach when discretizing the diffusion operator. Relative errors in the results are computed on a flat 2-tours T^2 using the max-norm. These results are reported in Table 5.5 and a convergence plot of the relative errors for MLS approach with Wendland function as weight function is shown in Figure 5.4.

This test indicates a first-order convergence in the value of u_s with respect to sample size (N) and second-order in space (h) (since the true dimension of the manifold is 2) for both MLS approach and constrained quadratic optimization approach.

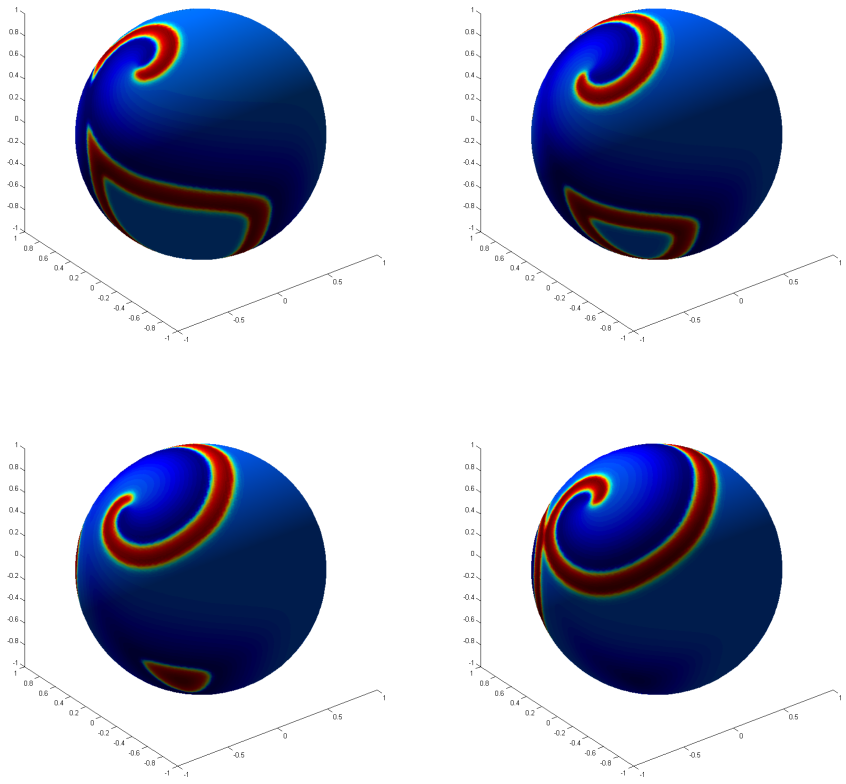

FIG. 5.3. *Fitzhugh-Nagumo equation evolving on a sphere. The excitation variable u_s is displayed at time $t = 400$, 450, 500 and 550. We use triangular mesh for better visualization, although our solution is computed based on point cloud.*

sample size	MLS [Wendland]			Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
1600	1.37e-02			1.37e-02		
3600	6.16e-03	0.99	1.97	6.16e-03	0.99	1.97
6400	3.46e-03	1.00	2.01	3.46e-03	1.00	2.01
14400	1.55e-03	0.99	1.98	1.55e-03	0.99	1.98
25600	8.67e-04	1.01	2.02	8.67e-04	1.01	2.02

TABLE 5.5
Max-norm relative errors for the diffusion equation on a flat 2-tours T^2.

5.2. Eigenvalue problems. Here we show a few examples of eigenvalue problem for Laplace-Beltrami operator on point clouds. We mainly test our method for open manifolds with Dirichlet or Neumann boundary conditions. More examples and applications in computer vision can be found in [17].

5.2.1. Hemisphere in 3D with Dirichlet boundary condition. Consider first eigenvalue problem for unit hemisphere in 3D. With homogeneous Dirichlet

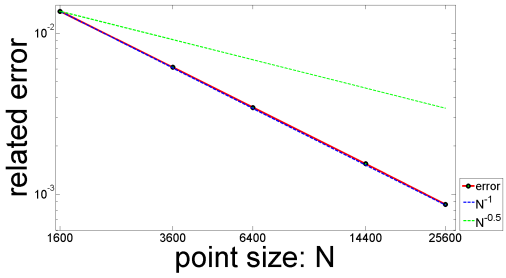

FIG. 5.4. *Convergence plot of max-norm relative errors using MLS approach with Wendland function as weight function for the diffusion equation on a flat 2-tours T^2.*

boundary condition, the problem becomes

$$\begin{cases} -\Delta_{\mathcal{M}}\phi = \lambda\phi & \mathbf{x} \in \Omega \\ \phi(\mathbf{x}) = 0 & \mathbf{x} \in \partial\Omega \end{cases} \tag{5.6}$$

The exact value of the n-th eigenvalue is given by $\lambda_n = n(n+1)$, with multiplicity n for $n = 1, 2, \cdots$. To measure the error of our approach in computing eigenvalues, we compute the normalized error $E_{\max,n} = \max(\frac{|\tilde{\lambda}_{n,i} - \lambda_n|}{\lambda_n})$, where $\tilde{\lambda}_{n,i}$'s are the eigenvalues computed from our approach for eigenvalue λ_n, and i runs over each multiplicity. $E_{\max,n}$ represents the worst possible error in computing λ_n. We use constant $K = 15$ (number of KNN to be used) and calculate numerical solutions for both MLS approach and constraint quadratic optimization approach. We show $E_{\max,n}$ for $n = 5$ and 13 for several point clouds with different sample sizes to illustrate convergence of our approach. These results are reported in Table 5.6 and a convergence plot of the relative errors for MLS approach with Wendland function as weight function is shown in Figure 5.5.

sample size	$\lambda = 30$ MLS [Wendland]			$\lambda = 30$ Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
521	3.98e-02			4.03e-02		
1009	2.22e-02	0.88	1.77	2.25e-02	0.88	1.76
2041	1.18e-02	0.90	1.79	1.19e-02	0.90	1.81
3977	6.39e-03	0.92	1.84	6.43e-03	0.92	1.85
8081	3.31e-03	0.93	1.86	3.32e-03	0.93	1.86
sample size	$\lambda = 182$ MLS [Wendland]			$\lambda = 182$ Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
521	3.14e-01			2.44e-01		
1009	1.31e-01	1.32	2.65	1.33e-01	0.92	1.84
2041	6.96e-02	0.90	1.80	7.01e-02	0.91	1.82
3977	3.76e-02	0.92	1.85	3.79e-02	0.92	1.84
8081	1.91e-02	0.96	1.91	1.93e-02	0.95	1.90

TABLE 5.6
Max-norm relative errors $E_{\max,n}$ for eigenvalues 30 and 182.

This test indicates a first-order convergence in the value of eigenvalue λ_n with respect to sample size (N) and second-order with respect to space (h) for both MLS approach and constrained quadratic optimization approach.

5.2.2. Hemisphere in 3D with Neumann boundary condtion. Next we consider eigenvalue problem for unit hemisphere in 3D with homogeneous Neumann

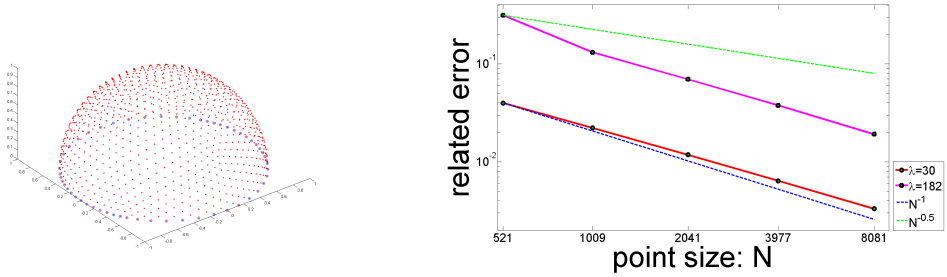

FIG. 5.5. *Point cloud sampled from hemisphere with 1009 points, and points with blue circle are boundary points. Convergence plot of max-norm relative errors using MLS approach with Wendland function as weight function for eigenvalue problem on unit hemisphere with Dirichlet boundary condition.*

boundary condition. The problem becomes

$$\begin{cases} -\Delta_{\mathcal{M}}\phi = \lambda\phi & \mathbf{x} \in \Omega \\ \frac{\partial \phi}{\partial \mathbf{n}} = 0 & \mathbf{x} \in \partial\Omega \end{cases} \tag{5.7}$$

The exact value of the n-th eigenvalue is given by $\lambda_n = (n-1)n$, with multiplicity n for $n = 1, 2, \cdots$. Again we compute $E_{\max,n}$ for $n = 6$ and 14 for several point clouds with different sample sizes to illustrate convergence of our approach. We use constant $K = 15$ (number of KNN to be used) and calculate numerical solutions for both MLS approach and constraint quadratic optimization approach. These results are reported in Table 5.7 and a convergence plot of the relative errors for MLS approach with Wendland function as weight function is shown in Figure 5.6.

sample size	$\lambda = 30$ MLS [Wendland]			$\lambda = 30$ Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
521	6.42e-02			4.92e-02		
1009	3.11e-02	1.10	2.19	2.60e-02	0.96	1.93
2041	1.47e-02	1.06	2.13	1.31e-02	0.97	1.95
3977	7.46e-03	1.02	2.03	6.89e-03	0.96	1.93
8081	3.67e-03	1.00	2.00	3.48e-03	0.96	1.93
sample size	$\lambda = 182$ MLS [Wendland]			$\lambda = 182$ Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
521	3.13e-01			2.59e-01		
1009	1.97e-01	0.70	1.40	1.49e-01	0.84	1.67
2041	9.94e-02	0.97	1.94	7.82e-02	0.92	1.83
3977	4.86e-02	1.07	2.15	4.12e-02	0.96	1.92
8081	2.29e-02	1.06	2.12	2.05e-02	0.98	1.97

TABLE 5.7
Max-norm relative errors $E_{\max,n}$ for eigenvalues 30 and 182.

This convergence test indicates a first-order convergence in the value of eigenvalue λ_n with respect to sample size (N) and second-order with respect to space (h) for both MLS approach and constrained quadratic optimization approach.

5.2.3. Flat 3-tours T^3 in 6D. Consider eigenvalue problem for flat 3-torus T^3, a three dimensional manifold embedded in $\mathbb{R}^6$ paramterized as

$$\mathbf{\Gamma} = (\cos\alpha, \sin\alpha, \cos\beta, \sin\beta, \cos\theta, \sin\theta)$$

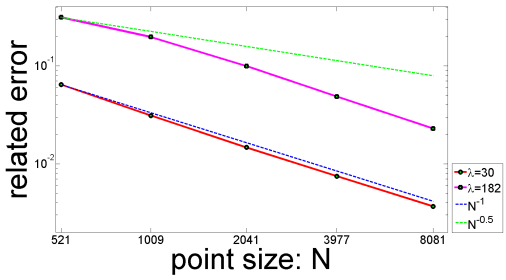

FIG. 5.6. *Convergence plot of max-norm relative errors using MLS approach with Wendland function as weight function for eigenvalue problem on unit hemisphere with Neumann boundary condition.*

with $\alpha, \beta, \theta \in [0, 2\pi]$. Using our approach, the manifold is locally approximated as

$$\mathbf{\Gamma} = (x_1, x_2, x_3, y_1(x_1, x_2, x_3), y_2(x_1, x_2, x_3), y_3(x_1, x_2, x_3))$$

and the eigenfunction ϕ is locally approximated as $\Phi(x_1, x_2, x_3)$, where y_1, y_2, y_3 and Φ are 3-dimensional degree 2 polynomials. Our method apply to this 3D manifold in $\mathbb{R}^6$ in a straight forward way. The "ground truth" eigenvalues and their multiplicities are not available, instead we use some test functions to measure the error. We compute the L_∞ error for $\Delta_{\mathcal{M}} f$ for the functions $f = x, x^2, e^x$ on T^3, where x is the first coordinate in $\mathbb{R}^6$, and the closed forms of their surface Laplacian are known. The L_∞ error is defined as $E_\infty = \frac{||MF-U||_\infty}{||U||_\infty}$, where M is the discretized Lapalce-Beltrami operator defined in (3.10), F is the N-dimensional vector for f evaluated on data points and U is the N-dimensional vector for the known values of $\Delta_{\mathcal{M}} f$ in its closed form.

We apply our approach and calculate the E_∞ relative errors of the numerical solutions for several point clouds with different sample sizes. We use constant $K = 20$ (number of KNN to be used) and both MLS approach and constrained quadratic optimization approach. These results are reported in Table 5.8 and a convergence plot of the relative errors for MLS approach with Wendland function as weight function for test function $f = x$ is shown in Figure 5.7.

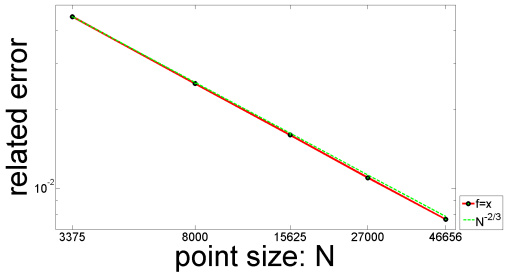

FIG. 5.7. *Convergence plot of E_∞ relative errors using MLS approach with Wendland function as weight function for test function $f = x$ on T^3.*

Again, second-order super-convergence with respect to space (h) for both MLS approach and constrained quadratic optimization approach is observed.

sample size	$f = x$ MLS [Wendland]			$f = x$ Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
3375	4.51e-02			4.51e-02		
8000	2.51e-02	0.68	2.04	2.51e-02	0.68	2.04
15625	1.60e-02	0.67	2.02	1.60e-02	0.67	2.02
27000	1.10e-02	0.69	2.06	1.10e-02	0.69	2.06
46656	7.64e-03	0.67	2.00	7.64e-03	0.67	2.00
sample size	$f = x^2$ MLS [Wendland]			$f = x^2$ Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
3375	6.43e-04			6.43e-02		
8000	2.65e-04	1.03	3.08	2.65e-04	1.03	3.08
15625	1.41e-04	0.94	2.83	1.41e-04	0.94	2.83
27000	9.21e-05	0.78	2.34	9.21e-05	0.78	2.34
46656	5.21e-05	1.04	3.12	5.21e-05	1.04	3.12
sample size	$f = e^x$ MLS [Wendland]			$f = e^x$ Constraint [Wendland]		
N	Error	Conv. in N	Conv. in h	Error	Conv. in N	Conv. in h
3375	2.65e-02			2.65e-02		
8000	1.45e-02	0.70	2.10	1.45e-02	0.70	2.10
15625	9.62e-03	0.61	1.84	9.62e-03	0.61	1.84
27000	6.85e-02	0.62	1.86	6.85e-02	0.62	1.86
46656	4.60e-03	0.73	2.18	4.60e-03	0.73	2.18

TABLE 5.8
E_∞ relative errors for test functions $f = x, x^2, e^x$ on T^3.

5.3. Diffusion on Stanford bunny. We conclude our numerical experiments by the heat equation on the point cloud for Stanford bunny with a point source in the left ear. We normalize the point cloud so that it is bounded by unit box. The total number of points is 35296. The minimal distance between data points Δx may not be a good indicator of h for complicated shape. Instead, we report the mean of the distance between data point and its closest neighbor $\widetilde{\Delta x} = \text{mean}\Delta x_i$ where $\Delta x_i = \min_j \|\mathbf{p}_i - \mathbf{p}_j\|$. For our normalized bunny, we have $\widetilde{\Delta x} = 0.0047$. In this simulation forward Euler is used with $\Delta t = 1e-5$ and $K = 20$ (number of KNN to be used). The results of numerical solution u_s are displayed at time t = 0.1, 0.2, 0.4 and 0.8 in Figure 5.8. Again, the solution is displayed on a triangulated mesh for better visual effect.

The above simulation is based on the constraint quadratic optimization approach with the Wendlend weight function. We want to point out that using MLS approach with standard Wendland or inverse of squared distance weight function works well for pretty uniformly distributed point clouds. However, in more challenging situation, such as for non-uniform point clouds, one may need to use the constrained quadratic optimization approach which is more robust and works independent of the choice of weight function. The MLS approach with the Wendlend weight function produces negative values for this example. Similarly, using MLS with the Wendlend weight function to solve eigenvalue problem for Laplace-Beltrami operator on this point cloud produces negative eigenvalues, while using the constrained quadratic optimization approach does not. Another interesting fact is that the special weight function (3.12) for MLS seems more robust and works well for non-uniform point clouds like this example.

6. Conclusion. In this work, we present a general approach to solve PDEs on manifolds represented by point clouds. The key idea is to approximate differential operators on the manifold by constructing the function, the manifold and hence the metric in a local coordinate system at each point. In this way a global parametrization

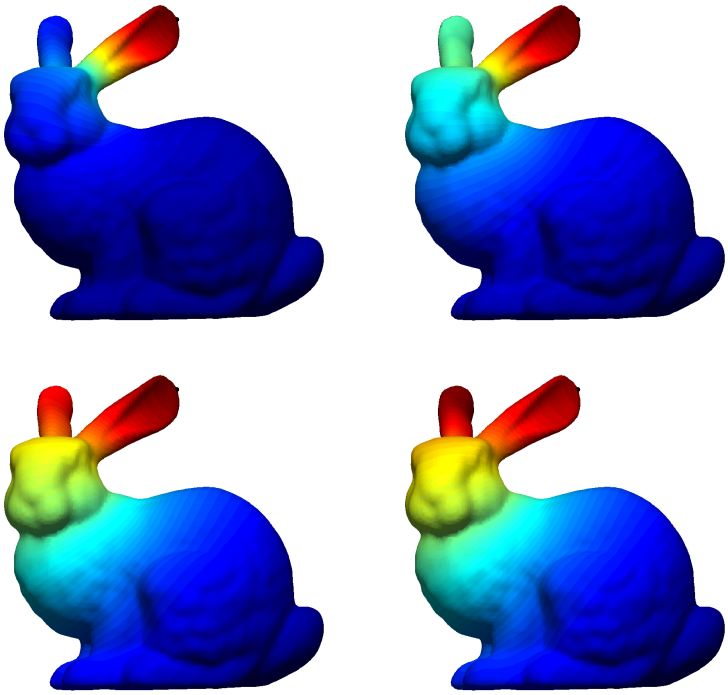

FIG. 5.8. *Solve heat equation on Stanford bunny. The numerical solution u_s is displayed at time $t = 0.1$, 0.2, 0.4 and 0.8. The black dot is the heat source. Again, we use triangular mesh for better visualization, although our solution is computed directly on point cloud.*

or mesh can be avoided, which allows this approach to handle manifolds with arbitrary dimensions and co-dimensions. Moreover, the complexity of the methods scales well with the total number of points and the true dimension of the manifold. Different least square approximations, treatment of boundary conditions, approximation error analysis, and numerical tests are presented.

Appendix A. Approximation error for MLS. In this appendix, we rewrite some results in [28] to provide a proof of the approximation error for the MLS problem (2.3).

First, we introduce some notations. To expedite the presentation, multi-index notation is used. If $\alpha := (a_1, a_2, \cdots, a_d)$ is a d-tuple of nonnegative integers a_i and its length is defined as

$$|\alpha| := \sum_{i=1}^{d} a_i \tag{A.1}$$

We then denote

$$\mathbf{x}^\alpha := x_1^{a_1} x_2^{a_2} \cdots x_d^{a_d} \tag{A.2}$$

and the α-th derivative of function f as

$$D^\alpha f := \frac{\partial^{a_1}}{\partial x_1^{a_1}} \frac{\partial^{a_2}}{\partial x_2^{a_2}} \cdots \frac{\partial^{a_d}}{\partial x_d^{a_d}} f \tag{A.3}$$

Use the same notations in Section 2.2, Π_m^d is the space of polynomials of total degree m in d-dimensions and $\mathbf{b}(\mathbf{x}) = [(\mathbf{x}-\overline{\mathbf{x}})^{\alpha_1}, (\mathbf{x}-\overline{\mathbf{x}})^{\alpha_2}, \cdots, (\mathbf{x}-\overline{\mathbf{x}})^{\alpha_I}]^T$ is the polynomial basis vector where $I = \frac{(d+m)!}{d!m!}$ is the number of basis in Π_m^d. Denote E as a $K \times I$ matrix with $E_{ij} = b_j(\mathbf{x}_i)$, $W = \mathrm{Diag}\{w_1, w_2, \cdots, w_K\}$ with $w_k = w(\|\mathbf{x}_k - \overline{\mathbf{x}}\|)$ and $F = [f(\mathbf{x}_1), f(\mathbf{x}_2), \cdots, f(\mathbf{x}_K)]^T$. Assume $I \le K$, $\mathrm{Rank}(E) = I$ and $w(\cdot) > 0$, the solution (2.5) of MLS problem can be written as

$$\mathbf{c}(\overline{\mathbf{x}}) = \left(E^T W E\right)^{-1} E^T W F \tag{A.4}$$

The following completeness condition holds.

LEMMA A.1. *Assume $I \le K$, Rank$(E) = I$ and $w(\cdot) > 0$. The moving least squares (MLS) approximation function $f_{\overline{\mathbf{x}}}(\mathbf{x})$ (2.4) can reproduce any polynomials $f(\mathbf{x}) \in \Pi_m^d$ exactly by using the sample values, viz.*

$$f_{\overline{\mathbf{x}}}(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \mathbf{c}(\overline{\mathbf{x}}) = f(\mathbf{x}) \tag{A.5}$$

where $\mathbf{c}(\overline{\mathbf{x}})$ is given by (2.5) or (A.4) and the result holds for any weight function $w(\cdot) > 0$.

Proof. It suffices to show that for $0 < i < I$, if $f(\mathbf{x}) = b_i(\mathbf{x}) = (\mathbf{x}-\overline{\mathbf{x}})^{\alpha_i}$, then $\mathbf{c}(\overline{\mathbf{x}}) = \mathbf{e}_i$ where $\mathbf{e}_i$ is the i-th standard basis for $\mathbb{R}^I$.

Assume $f(\mathbf{x}) = b_i(\mathbf{x})$, then

$$\begin{aligned}
\mathbf{c}(\overline{\mathbf{x}}) &= \left[\sum_k w_k \mathbf{b}(\mathbf{x}_k)\mathbf{b}(\mathbf{x}_k)^T\right]^{-1} \sum_k w_k \mathbf{b}(\mathbf{x}_k) b_i(\mathbf{x}_k) \\
&= \begin{bmatrix} \sum w_k b_1(\mathbf{x}_k) b_1(\mathbf{x}_k) & \cdots & \sum w_k b_1(\mathbf{x}_k) b_I(\mathbf{x}_k) \\ \vdots & \ddots & \vdots \\ \sum w_k b_1(\mathbf{x}_k) b_I(\mathbf{x}_k) & \cdots & \sum w_k b_I(\mathbf{x}_k) b_I(\mathbf{x}_k) \end{bmatrix}^{-1} \begin{bmatrix} \sum w_k b_1(\mathbf{x}_k) b_i(\mathbf{x}_k) \\ \vdots \\ \sum w_k b_I(\mathbf{x}_k) b_i(\mathbf{x}_k) \end{bmatrix} \\
&= \mathbf{e}_i
\end{aligned}$$

where *Laplace* theorem is used in the last step. ∎

Next we give the following lemma for approximation error of MLS method.

LEMMA A.2. *Assume $f \in C^{m+1}(\mathbb{R}^d)$, $I \le K$, Rank$(E) = I$ and $w(\cdot) > 0$. The solution (2.5) of MLS method can be used to approximate a function f and its derivatives. More precisely, denote h as the smallest radius such that the K given points lie in a ball of such radius and centered at $\overline{\mathbf{x}}$, i.e., $h = \max_k \|\mathbf{x}_k - \overline{\mathbf{x}}\|$ then*

$$\left| c_i - \frac{1}{\alpha_i!} D^{\alpha_i} f(\overline{\mathbf{x}}) \right| = C(w, f, \alpha_i) h^{m+1-|\alpha_i|} \tag{A.6}$$

where C is a constant depends on w, f and α_i.

Proof. By Taylor expansion

$$f(\mathbf{x}_k) = \sum_{i=1}^{I} \frac{1}{\alpha_i!} D^{\alpha_i} f(\overline{\mathbf{x}}) b_i(\mathbf{x}_k) + \sum_{|\alpha|=m+1} \frac{1}{\alpha!} D^{\alpha} f(\overline{\mathbf{x}} + \theta(\mathbf{x}_k - \overline{\mathbf{x}}))(\mathbf{x}_k - \overline{\mathbf{x}})^{\alpha} \tag{A.7}$$

Denote $A = \left[\sum_{k=1}^K w_k \mathbf{b}(\mathbf{x}_k)\mathbf{b}(\mathbf{x}_k)^T\right]$, the first term in (A.7) as I_1 and the second

term as I_2, use Lemma A.1, we have

$$\begin{aligned}\mathbf{c}(\overline{\mathbf{x}}) &= A^{-1}\sum_{k=1}^{K} w_k\mathbf{b}(\mathbf{x}_k)(I_1+I_2)\\ &= \sum_{i=1}^{I}\frac{1}{\alpha_i!}D^{\alpha_i}f(\overline{\mathbf{x}})A^{-1}\sum_{k=1}^{K} w_k\mathbf{b}(\mathbf{x}_k)b_i(\mathbf{x}_k) + A^{-1}\sum_{k=1}^{K} w_k\mathbf{b}(\mathbf{x}_k)I_2\\ &= \sum_{i=1}^{I}\frac{1}{\alpha_i!}D^{\alpha_i}f(\overline{\mathbf{x}})\mathbf{e}_i + A^{-1}\sum_{k=1}^{K} w_k\mathbf{b}(\mathbf{x}_k)I_2\end{aligned}$$

Denote $A=[A_{ij}]$ and $A^{-1}=\left[A^{ij}\right]$, then we have

$$\begin{aligned}\left|c_i - \frac{1}{\alpha_i!}D^{\alpha_i}f(\overline{\mathbf{x}})\right| &= \left|\sum_j A^{ij}\sum_k w_k b_j(\mathbf{x}_k)I_2\right|\\ &\le \sum_k w_k\sum_j |A^{ij}|\cdot|b_j(\mathbf{x}_k)|\cdot|I_2|\end{aligned}$$

Notice that $A_{ij}\sim O(h^{|\alpha_i|+|\alpha_j|})$, $A^{ij}\sim O(h^{-|\alpha_i|-|\alpha_j|})$, $b_j(\mathbf{x}_k)\sim O(h^{|\alpha_j|})$ and $I_2\sim O(h^{m+1})$, so each term in the above sum is with $O(h^{m+1-|\alpha_i|})$, hence

$$\left|c_i - \frac{1}{\alpha_i!}D^{\alpha_i}f(\overline{\mathbf{x}})\right| = C(w,f,\alpha_i)h^{m+1-|\alpha_i|} \tag{A.8}$$

for $i=1,2,\cdots,I$. ∎

Finally, we show super-convergence for MLS approximation due to cancellation of errors. We first present a precise statement in 1D .

LEMMA A.3. *For $f\in C^{m+2}(\mathbb{R})$, $I=m+1\le K$, Rank$(E)=m+1$ and $w(\cdot)>0$., if $P=\{x_k|k=1,2,\cdots,K\}$ are symmetrically distributed around $\overline{x}$, that is if $x_k\in P$ then $2\overline{x}-x_k\in P$. We have the following*

$$\left|c_{i+1} - \frac{1}{i!}f^{(i)}(\overline{x})\right| = C(w,f,i)h^{m+2-i} \tag{A.9}$$

for i ($\in\{0,1,\cdots,m\}$) with the same parity of m, where C is a constant depends on w, f and i.

Proof. By Taylor expansion

$$\begin{aligned}f(x_k) = \;& \textstyle\sum_{i=0}^{m}\frac{1}{i!}f^{(i)}(\overline{x})b_i(x_k) + \frac{1}{(m+1)!}f^{(m+1)}(\overline{x})b_{m+1}(x_k)\\ & + \textstyle\frac{1}{(m+2)!}f^{(m+2)}(\overline{x}+\theta_k(x_k-\overline{x}))b_{m+2}(x_k)\end{aligned} \tag{A.10}$$

where $b_i(x)=(x-\overline{x})^i$. Denote $A=\left[\sum_{k=1}^{K} w_k\mathbf{b}(x_k)\mathbf{b}(x_k)^T\right]$, the first term in (A.10) as I_1, the second term as I_2 and the third term as I_3, use computations in Lemma A.2, we have

$$\mathbf{c}(\overline{x}) = \sum_{i=0}^{m}\frac{1}{i!}f^{(i)}(\overline{x})\mathbf{e}_{i+1} + A^{-1}\sum_{k=1}^{K} w_k\mathbf{b}(x_k)(I_2+I_3)$$

and

$$\left| c_{i+1} - \frac{1}{i!} f^{(i)}(\overline{x}) \right| \leq \left| \sum_{j=1}^{m+1} A^{i+1j} \sum_{k=1}^{K} w_k b_{j-1}(x_k) I_2 \right| + O(h^{m+2-i})$$

Denote $\widehat{I_2} = \sum_j A^{i+1j} \sum_k w_k b_{j-1}(x_k) I_2$ and $A^{i+1j} = a^{i+1j}/\det(A)$, where $\det(A) = O(h^{2s})$ with $s = m(m+1)/2$ and $a^{i+1j} = \sum_\mu \prod_{\nu=1}^{m} \sum_k w_k p_\nu^{(\mu)}(x_k)$ where $p_\nu^{(\mu)}(x)$ is some polynomial and summation of the degree of $p_\nu^{(\mu)}(x)$, $\sum_\nu \text{degree}(p_\nu^{(\mu)}(x))$ is $2s-(i+1)-j$ for each μ. (The notations here are a little complicated, please see the following example for better understanding.)

$$\widehat{I_2} = \frac{1}{\det(A)} \frac{1}{(m+1)!} f^{(m+1)}(\overline{x}) \sum_j \left(a^{i+1j} \left(\sum_k w_k b_{j-1}(x_k)(x_k - \overline{x})^{m+1} \right) \right)$$

Now that $w_k = w(|x_k - \overline{x}|)$, $\{x_k | k = 1, 2, \cdots, K\}$ are symmetrically distributed around $\overline{x}$ and i has the same parity as m. We can divide the above summation into two subgroup summations, one is for j that has the same parity as i, the other different. For j that has the same parity as i, now $2s-(i+1)-j$ is odd, there exists some $\widehat{\nu}$ (depends on μ) for each μ in the summation for a^{i+1j} that $p_{\widehat{\nu}}^{(\mu)}(x)$ is an odd degree polynomial and $\sum_k w_k p_{\widehat{\nu}}^{(\mu)}(x_k) = 0$, hence $a^{i+1j} = 0$. On the other hand, for j that has different parity as i (also m), $b_{j-1}(x)(x-\overline{x})^{m+1} = (x-\overline{x})^{j+m}$ is an odd degree polynomial and $\sum_k w_k b_{j-1}(x_k)(x_k - \overline{x})^{m+1} = 0$. After all, we have $\widehat{I_2} = 0$. □

We give a concrete example for better understanding of Lemma A.3. Consider MLS problem in 1D with symmetric distributed sampling around $\overline{x}$. We construct quadratic polynomial (m=2) to estimate $f''(\overline{x})$. Now $\widehat{I_2}$ can be written as

$$\widehat{I_2} = \frac{1}{2\det(A)} f^{(3)}(\overline{x}) \sum_j \left(a^{3j} \left(\sum_k w_k b_{j-1}(x_k)(x_k - \overline{x})^3 \right) \right)$$

For $j = 1$ and 3, $\sum_k w_k b_{j-1}(x_k)(x_k - \overline{x})^3 = \sum_k w_k (x_k - \overline{x})^{j+2} = 0$. For $j = 2$, $a^{32} = \sum w_k (x_k - \overline{x}) \sum w_k (x_k - \overline{x})^2 - \sum w_k \sum w_k (x_k - \overline{x})^3$. For $\mu = 1$, we have $p_1^{(1)} = x - \overline{x}$ and $\sum w_k p_1^{(1)}(x_k) = 0$; for $\mu = 2$, we have $p_2^{(2)} = (x-\overline{x})^3$ and $\sum w_k p_2^{(2)}(x_k) = 0$. After all, we have $\widehat{I_2} = 0$.

For high dimensional MLS, super-convergence result still holds for $|\alpha_i|$ with the same parity as m. Below is a brief argument with more complicated computations and notations. Again using Taylor expansion we have

$$\begin{aligned} f(\mathbf{x}_k) = \quad & \sum_{i=1}^{I} \tfrac{1}{\alpha_i!} D^{\alpha_i} f(\overline{\mathbf{x}}) b_i(\mathbf{x}_k) + \sum_{|\alpha|=m+1} \tfrac{1}{\alpha!} D^\alpha f(\overline{\mathbf{x}})(\mathbf{x}_k - \overline{\mathbf{x}})^\alpha \\ & + \sum_{|\alpha|=m+2} \tfrac{1}{\alpha!} D^\alpha f(\overline{\mathbf{x}} + \theta_k(\mathbf{x}_k - \overline{\mathbf{x}}))(\mathbf{x}_k - \overline{\mathbf{x}})^\alpha \end{aligned} \tag{A.11}$$

Denote $A = \left[\sum_{k=1}^{K} w_k \mathbf{b}(\mathbf{x}_k) \mathbf{b}(\mathbf{x}_k)^T \right]$, the first term in (A.11) as I_1, the second term as I_2 and the third term as I_3, use computations in Lemma A.2, we have

$$\mathbf{c}(\overline{\mathbf{x}}) = \sum_{i=1}^{I} \frac{1}{\alpha_i!} D^{\alpha_i} f(\overline{\mathbf{x}}) \mathbf{e}_i + A^{-1} \sum_{k=1}^{K} w_k \mathbf{b}(\mathbf{x}_k)(I_2 + I_3)$$

and

$$\left| c_i - \frac{1}{\alpha_i!} D^{\alpha_i} f(\overline{\mathbf{x}}) \right| \leq \left| \sum_j A^{ij} \sum_k w_k b_j(\mathbf{x}_k) I_2 \right| + O(h^{m+2-|\alpha_i|})$$

Denote $\widehat{I_2} = \sum_j A^{ij} \sum_k w_k b_j(\mathbf{x}_k) I_2$ and $A^{ij} = a^{ij}/\det(A)$, where $\det(A) = O(h^{2s})$ with $s = \sum_{i=1}^{I} |\alpha_i|$ and $a^{ij} = \sum_\mu \prod_{\nu=1}^{I-1} \sum_k w_k p_\nu^{(\mu)}(\mathbf{x}_k)$ where $p_\nu^{(\mu)}(\mathbf{x})$ is some d-dimensional polynomial and summation of the degree of $p_\nu^{(\mu)}(\mathbf{x})$, $\sum_\nu \text{degree}(p_\nu^{(\mu)}(\mathbf{x}))$ is $2s - |\alpha_i| - |\alpha_j|$ for each μ.

$$\widehat{I_2} = \frac{1}{\det(A)} \sum_{|\alpha|=m+1} \frac{1}{\alpha!} D^\alpha f(\overline{\mathbf{x}}) \sum_j a^{ij} \sum_k w_k b_j(\mathbf{x}_k)(\mathbf{x}_k - \overline{\mathbf{x}})^\alpha$$

Again $\widehat{I_2}$ is the leading term in the error and cancellation has to occur for super-convergence. Assume there are some symmetry in the distribution of given points $\{\mathbf{x}_k | k = 1, 2, \cdots, K\}$ around $\overline{\mathbf{x}}$. For j such that $|\alpha_j|$ has the same parity as m, $b_j(\mathbf{x})(\mathbf{x} - \overline{\mathbf{x}})^\alpha$ is an odd degree polynomial since b_j is a d-dimensional polynomial of degree $|\alpha_j|$. Error cancellation will occur in $\sum_k w_k b_j(\mathbf{x}_k)(\mathbf{x}_k - \overline{\mathbf{x}})^\alpha$. For j such that $|\alpha_j|$ and m (also $|\alpha_i|$) have different parities, $2s - |\alpha_i| - |\alpha_j|$ is odd, there exists some $\widehat{\nu}$ (depends on μ) for each μ in the summation for a^{ij} that $p_{\widehat{\nu}}^{(\mu)}(\mathbf{x})$ is a odd degree polynomial. Again cancellation will occur in $\sum_k w_k p_{\widehat{\nu}}^{(\mu)}(\mathbf{x}_k)$. Error cancellation in the summation over k of these odd-degree polynomials will give super-convergence. Although strict symmetry condition is difficult to satisfy in high dimensions, perfect cancellation is not necessary. Often one observes super-convergence up to one order due to cancellation to some extent. For most of the tests in Section 5, the term $\widehat{I_2}$ can achieve one order higher and we do observe super-convergence when $|\alpha_i|$ has the same parity as m.

REMARK 4. *The above super-convergence result agrees with the study of recovery method in [34].*

Appendix B. Connection between MLS and constrained quadratic optimization problem. We show how we use approach in [27] to recast the MLS problem (2.3) as a constrained quadratic optimization problem, from which we can add additional constraint. Based on equation (2.4) and (A.4), $f(\mathbf{x})$ can be approximated by $f_{\overline{\mathbf{x}}}(\mathbf{x})$ where $f_{\overline{\mathbf{x}}}$ is the MLS approximation computed around $\overline{\mathbf{x}}$ and it can be written as

$$f_{\overline{\mathbf{x}}}(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \mathbf{c}(\overline{\mathbf{x}}) \tag{B.1}$$

And $D^{\alpha_i} f_{\overline{\mathbf{x}}}(\overline{\mathbf{x}}) = \left[D^{\alpha_i} \mathbf{b}(\overline{\mathbf{x}})\right]^T \mathbf{c}(\overline{\mathbf{x}}) = \alpha_i! \mathbf{e}_i^T \mathbf{c}(\overline{\mathbf{x}}) = \alpha_i! c_i$ is an estimate of the true derivative $D^{\alpha_i} f(\overline{\mathbf{x}})$. Such approximation is a linear combination of $f_i = f(\mathbf{x}_i)$, since $\mathbf{c}(\overline{\mathbf{x}})$ is a linear combination of f_i (see (2.5) or (A.4)). Let us consider the following problem: finding coefficients vector $\mathbf{a}_i \in \mathbb{R}^K$ for the approximation $\widehat{f_i} = \mathbf{a}_i^T F$ of $D^{\alpha_i} f(\overline{\mathbf{x}})$ by minimizing the quadratic form

$$Q = \frac{1}{2} \sum_{k=1}^{K} \frac{\left(a_k^{(i)}\right)^2}{w_k} = \frac{1}{2} \mathbf{a}_i^T W^{-1} \mathbf{a}_i \tag{B.2}$$

subject to the linear constraints

$$\sum_{k=1}^{K} a_k^{(i)} b_j(\mathbf{x}_k) = r_j^{(i)} \qquad \text{for } j = 1, 2, \cdots, I \tag{B.3}$$

where $\mathbf{r}_i = \left[r_1^{(i)}, r_2^{(i)}, \cdots, r_I^{(i)}\right]^T = \alpha_i! \mathbf{e}_i$.

Now we prove that MLS estimates evaluated at $\overline{\mathbf{x}}$ are equivalent to approximations from the above minimization problem.

LEMMA B.1. *The estimate $D^{\alpha_i} f_{\overline{\mathbf{x}}}(\overline{\mathbf{x}})$ (from MLS) of the true derivative $D^{\alpha_i} f(\overline{\mathbf{x}})$ is equivalent to the estimate $\widehat{f_i}$ from the above constrained quadratic optimization problem for all $i = 1, 2, \cdots, I$, assuming $I \leq K$, Rank$(E) = I$ and $w(\cdot) > 0$.*

Proof. The constraint minimization problem (B.2)-(B.3) is transformed, using Lagrange multipliers $z_1, z_2, \cdots, z_I$, into the linear system

$$W^{-1}\mathbf{a}_i + E\mathbf{z} = \mathbf{0} \text{ and } E^T \mathbf{a}_i = \mathbf{r}_i \tag{B.4}$$

Notice that the matrix of the system,

$$\begin{bmatrix} W^{-1} & E \\ E^T & 0 \end{bmatrix} \tag{B.5}$$

is non-singular, and the solution is given by

$$\mathbf{a}_i = WE(E^T W E)^{-1}\mathbf{r}_i \quad \text{and} \quad \mathbf{z} = -(E^T W E)^{-1}\mathbf{r}_i$$

Hence, $\widehat{f_i} = \mathbf{a}_i^T F = \mathbf{r}_i^T (E^T W E)^{-1} E^T W F$. Using (A.4), we have $\widehat{f_i} = \mathbf{r}_i^T \mathbf{c}(\overline{\mathbf{x}}) = \alpha_i! \mathbf{e}_i^T \mathbf{c}(\overline{\mathbf{x}}) = \alpha_i! c_i = D^{\alpha_i} f_{\overline{\mathbf{x}}}(\overline{\mathbf{x}})$ for all $i = 1, 2, \cdots, I$. ∎

REMARK 5. *The proof of Lemma B.1 is almost the same as that of the proof for Proposition 1 in [27], we modify it for our problem.*

REMARK 6. *We call constraint (B.3) "consistency constraint", since it reproduce the MLS estimates and hence the solution from the above quadratic optimization problem shares the same accuracy with MLS approach.*

REFERENCES

[1] R. DEFAY AND I. PRIOGINE, *Surface Tension and Adsorption*, John Wiley & Sons, New York, 1966.

[2] C. HERRING, *Surface tension as a motivation for sintering*, in The Physics of Powder Metallurgy, W. E. Kingston, ed., McGraw-Hill, New York, 1951, pp. 143-179.

[3] M. HOFER AND H. POTTMANN, *Energy-minimizing splines in manifolds*, ACM Trans. Graph., 23(2004), pp. 284-293.

[4] F. MÉMOLI, G. SAPIRO AND P. THOMPSON, *Implicit brain imaging*, NeuroImage, 23(2004), pp. 179-188.

[5] U. DIEWALD, T. PREUSSER AND M. RUMPF, *Anisotropic diffusion in vector field visualization on Euclidean domains and surfaces*, IEEE Trans. Visualization Computer Graphics, 6(2000), pp. 139-149.

[6] J. DORSEY AND P. HANRAHAN, *Digital materials and virtual weathering*, Sci. Am., 282(2000), pp. 282-289.

[7] G. TURK, *Generating textures on arbitrary surfaces using reaction-diffusion*, ACM SIGGRAPH Comput. Graph., 25(1991), pp. 289-298.

[8] A. WITKIN AND M. KASS, *Reaction-diffusion textures*, ACM SIGGRAPH Comput. Graph., 25(1991), pp. 299-308.

[9] A. M. BRONSTEIN, M. M. BRONSTEIN, R. KIMMEL, M. MAHMOUDI AND G. SAPIRO, *A gromov-hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching*, Int. J. Comput. Vis., 89(2010), pp. 266-286.

[10] R. Lai, Y. Shi, K. Scheibel, S. Fears, R. Woods, A. W. Toga and T. F. Chan, *Metric-induced optimal embedding for intrinsic 3D shape analysis*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

[11] D. Raviv, A . M. Bronstein, M. M. Bronstein, R. Kimmel and N. Sochen, *Affine-invariant diffusion geometry for the analysis of deformable 3D shapes*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011.

[12] B. Levy, *Laplace-Beltrami eigenfunctions towards an algorithm that "understands" geometry*, IEEE Conference on Shape Modeling and Applications, 2006.

[13] J. Sun, M. Ovsjanjkov and L. Guibas, *A concise and provably informative multi-scale signature based on heat diffusion*, Proc. SGP, 2009.

[14] M. Belkin and P. Niyogi, *Laplacian eigenmaps for dimensionality reduction and data representation*, Neural Comput., 15(2002), pp. 1373-1396.

[15] R. R. Coifman and S. Lafon, *Diffusion maps*, Appl. Comput. Harmon. Anal., 21(2006), pp. 5-30.

[16] M. Belkin, J. Sun and Y. Wang, *Constructing Laplace operator from point clouds in* $\mathbb{R}^d$, Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, New York, New York, 2009, pp 1031-1040.

[17] J. Liang, R. Lai, T. W. Wong and H. Zhao, *Geometric understanding of point clouds using Laplace-Beltrami operator*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, (accepted).

[18] M.S. Floater and K. Hormann, *Surface parameterization: a tutorial and survey*, in Advances in Multiresolution for Geometric Modelling, N.A. Dodgson, M.S. Floater and M.A. Sabin, ed., Springer Verlag, 2005, pp. 157-186.

[19] M. Bertalmío, L.T. Cheng, S. Osher and G. Sapiro, *Variational problems and partial differential equations on implicit surfaces*, J. Comput. Phys., 174(2001), pp. 759-780.

[20] M. Bertalmío, F. Mémoli, L.T. Cheng, G. Sapiro and S. Osher, *Variational problems and partial differential equations on implicit surfaces: bye bye triangulated surfaces?*, in: Geometric Level Set Methods in Imaging, Vision, and Graphics, S. Osher and N. Paragios, ed., Springer, New York, 2003, pp. 381-398.

[21] J. Xu and H. Zhao, *An Eulerian formulation for solving partial differential equations along a moving interface*, J. Sci. Comput., 19(2003), pp. 573-594.

[22] J. Greer, *An improvement of a recent eulerian method for solving PDEs on general geometries*, J. Sci. Comput., 29(2006), pp. 321-352.

[23] S. J. Ruuth and B. Merriman, *A simple embedding method for solving partial differential equations on surfaces*, J. Comput. Phys., 227(2008), pp. 1943-1961.

[24] S. Leung, J. Lowengrub, and H. Zhao, *A grid based particle method for solving partial differential equations on evolving surfaces and modeling high order geometrical motion*, J. Comput. Phys., 230(2011), pp. 2540-2561.

[25] M. P. do Carmo, *Differential Geometry of Curves and Surfaces*, Prentice Hall, Englewood Cliffs, New Jersey, 1976.

[26] A. Nealen, *An as-short-as-possible introduction to the least squares, weighted least squares and moving least squares methods for scattered Data approximation and interpolation*, Tech. Rep., TU Darmstadt, 2004.

[27] D. Levin, *The approximation power of moving least-squares*, Math. Comp., 67(1998), pp. 1517-1531.

[28] W.K. Liu, S. Li and T. Belytschko, *Moving least-square reproducing kernel methods (I) methodology and convergence*, Comput. Methods Appl. Mech. Engrg., 143(1997), pp. 113-154.

[29] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, *Surface reconstruction from unorganized points*, in Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques, New York, NY, 1992, pp. 71-78.

[30] E. Castillo and H. Zhao, *Point cloud segmentation via constrained nonlinear least squares surface normal estimates*, UCLA CAM Reports, 104(2009).

[31] Z. Li and K. Ito, *Maximum principle preserving schemes for interface problems with discontinuous coefficients*, SIAM J. Sci. Comput., 23(2001), pp. 339-361.

[32] K.W. Morton and D.F. Mayers, *Numerical Solution of Partial Differential Equations*, Cambridge Press, Cambridge, UK, 1995.

[33] R. FitzHugh, *FitzHugh-Nagumo simplified cardiac action potential model*, Biophys. J., 1(1961), pp. 445-466.

[34] Z. Zhang and A. Naga, *A new finite element gradient recovery method: super-convergence property*, SIAM J. Sci. Comput., 26(2005), pp. 1192-1213.