Journal of Computational Mathematics
Vol.39, No.6, 2021, 865–879.
http://www.global-sci.org/jcm
doi:10.4208/jcm.2104-m2020-0309

CONVERGENCE OF THE WEIGHTED NONLOCAL LAPLACIAN ON RANDOM POINT CLOUD *

Zuoqiang Shi[1)]
Department of Mathematical Sciences, Tsinghua University, Beijing 100084, China
Email: zqshi@tsinghua.edu.cn
Bao Wang
Department of Mathematics, University of Utah, Salt Lake City, UT 84112, USA
Email: wangbaonj@gmail.com

Abstract

We analyze the convergence of the weighted nonlocal Laplacian (WNLL) on the high dimensional randomly distributed point cloud. Our analysis reveals the importance of the scaling weight, $\mu \sim |P|/|S|$ with $|P|$ and $|S|$ being the number of entire and labeled data, respectively, in WNLL. The established result gives a theoretical foundation of the WNLL for high dimensional data interpolation.

Mathematics subject classification: 65D05, 65D25, 41A05.
Key words: Weighted nonlocal Laplacian, Laplace-Beltrami operator, Point cloud; High-dimensional interpolation.

1. Introduction

In this paper, we consider the convergence of the weighted nonlocal Laplacian (WNLL) on high dimensional randomly distributed data. WNLL is proposed in [11] for high dimensional point cloud interpolation, which successfully resolves the curse of dimensionality issue in the classical basis function-based approaches. High dimensional point cloud interpolation is a fundamental problem in machine learning, which can be mathematically formulated as follows: Let $P = \{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n\}$ and $S = \{\boldsymbol{s}_1, \cdots, \boldsymbol{s}_m\}$ be two sets of points in $\mathbb{R}^d$. Suppose u is a function defined on the point cloud $\bar{P} = P \cup S$, which is known only over the set S, and we denote the function u as $b(\boldsymbol{s})$ for any $\boldsymbol{s} \in S$. We use interpolation methods, e.g. WNLL, to compute u over the whole point cloud $\bar{P}$ leveraging the given values over S.

Nonlocal Laplacian is widely used in nonlocal methods for image processing [2,3,6,7], and in nonlocal Laplacian, the interpolating function is obtained by minimizing the following energy functional

$$\mathcal{J}(u) = \frac{1}{2} \sum_{\boldsymbol{x},\boldsymbol{y} \in \bar{P}} w(\boldsymbol{x},\boldsymbol{y})(u(\boldsymbol{x}) - u(\boldsymbol{y}))^2, \tag{1.1}$$

with the constraint

$$u(\boldsymbol{x}) = b(\boldsymbol{x}), \qquad \boldsymbol{x} \in S. \tag{1.2}$$

Here, $w(\boldsymbol{x},\boldsymbol{y})$ is a given weight function, typically chosen to be Gaussian, i.e. $w(\boldsymbol{x},\boldsymbol{y}) = \exp(-\|\boldsymbol{x}-\boldsymbol{y}\|^2/\sigma^2)$ with $\sigma > 0$ being a hyperparameter, and $\|\cdot\|$ is the Euclidean norm in

* Received December 13, 2020 / Revised version received April 17, 2021 / Accepted April 29, 2021 / Published online September 13, 2021 /
1) Correspondence author

$\mathbb{R}^d$. In graph theory and machine learning literature, nonlocal Laplacian is also called graph Laplacian [4,16].

Graph Laplacian works very well with a high labeling rate, i.e., there is a large portion of labeled data. However, when the labeling rate is low, i.e. $|S|/|\bar{P}| \ll 1$, the solution of the graph Laplacian is found to be discontinuous at the labeled points [11,12]. WNLL is devised to fix the issues related to the low-labeling rate, and in WNLL, the energy functional in (1.1) is modified by adding the weight, $\frac{|\bar{P}|}{|S|}$, to balance the labeled and unlabeled terms, which resulting in

$$\min_u \sum_{\boldsymbol{x}\in P}\left(\sum_{\boldsymbol{y}\in\bar{P}} w(\boldsymbol{x},\boldsymbol{y})(u(\boldsymbol{x})-u(\boldsymbol{y}))^2\right) + \frac{|\bar{P}|}{|S|}\sum_{\boldsymbol{x}\in S}\left(\sum_{\boldsymbol{y}\in\bar{P}} w(\boldsymbol{x},\boldsymbol{y})(u(\boldsymbol{x})-u(\boldsymbol{y}))^2\right), \tag{1.3}$$

with the constraint

$$u(\boldsymbol{x}) = b(\boldsymbol{x}), \qquad \boldsymbol{x}\in S.$$

When the labeling rate is high, WNLL is close to graph Laplacian. However, when the labeling rate is low, the specially designed weight forces the solution to be close to the given values near the labeled points, such that the discontinuities are removed. Furthermore, The optimization problem (1.3) is easy to solve numerically. With a symmetric weight function, i.e. $w(\boldsymbol{x},\boldsymbol{y}) = w(\boldsymbol{y},\boldsymbol{x})$, the corresponding Euler-Lagrange equation of (1.3) is a simple linear system

$$\begin{aligned} &2\sum_{\boldsymbol{y}\in P} w(\boldsymbol{x},\boldsymbol{y})\,(u(\boldsymbol{x})-u(\boldsymbol{y})) + \left(\frac{|P|}{|S|}+2\right)\sum_{\boldsymbol{y}\in S} w(\boldsymbol{y},\boldsymbol{x})(u(\boldsymbol{x})-b(\boldsymbol{y})) = 0, \quad && \boldsymbol{x}\in P,\\ &u(\boldsymbol{x}) = b(\boldsymbol{x}), && \boldsymbol{x}\in S. \end{aligned}$$

This linear system can be solved efficiently by the conjugate gradient iteration. The advantages of the WNLL over the graph Laplacian have been shown evidently in image inpainting [11,12], scientific data interpolation [15], and more recently deep learning [13].

1.1. Main Result

We consider the error of the WNLL in a model problem, where the whole computational domain is set to be a k-dimensional closed manifold $\mathcal{M}$ embedded in $\mathbb{R}^d$. The point cloud P, uniformly distributed on $\mathcal{M}$, gives a discrete representation of $\mathcal{M}$. Let $\mathcal{D}\subset\mathcal{M}$ be a labeled subset of $\mathcal{M}$, and S is a uniform sample of $\mathcal{D}$. In S, we have $u(\boldsymbol{x}) = b(\boldsymbol{x})$. An illustration of the computational domain and the point cloud is shown in Fig. 1.1.

In WNLL, we solve the following linear system, (1.4), to extend the label function u to the entire domain P.

$$\sum_{\boldsymbol{y}\in P} R_\delta(\boldsymbol{x},\boldsymbol{y})\,(u_\delta(\boldsymbol{x})-u_\delta(\boldsymbol{y})) + \mu\sum_{\boldsymbol{y}\in S} R_\delta(\boldsymbol{x},\boldsymbol{y})(u_\delta(\boldsymbol{x})-b(\boldsymbol{y})) = 0, \quad \boldsymbol{x}\in P, \tag{1.4a}$$

$$u_\delta(\boldsymbol{x}) = b(\boldsymbol{x}), \qquad \boldsymbol{x}\in S, \tag{1.4b}$$

where $R_\delta(\boldsymbol{x},\boldsymbol{y})$ is kernel function given as

$$R_\delta(\boldsymbol{x},\boldsymbol{y}) = C_\delta R\left(\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{4\delta^2}\right), \tag{1.5}$$

where $C_\delta = \frac{1}{\omega_k\delta^k}$ with ω_k is the volume of the unit ball in $\mathbb{R}^k$. $R:[0,+\infty)\to\mathbb{R}$ is a kernel functions satisfying the conditions listed in Assumption 1.1.