Partition of Unity Methods for Signal Processing on Graphs

Roberto Cavoretto\textsuperscript{1} · Alessandra De Rossi\textsuperscript{1} · Wolfgang Erb\textsuperscript{2,3}

Received: 19 December 2020 / Revised: 2 May 2021 / Accepted: 9 June 2021
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract
Partition of unity methods (PUMs) on graphs are simple and highly adaptive auxiliary tools for graph signal processing. Based on a greedy-type metric clustering and augmentation scheme, we show how a partition of unity can be generated in an efficient way on graphs. We investigate how PUMs can be combined with a local graph basis function (GBF) approximation method in order to obtain low-cost global interpolation or classification schemes. From a theoretical point of view, we study necessary prerequisites for the partition of unity such that global error estimates of the PUM follow from corresponding local ones. Finally, properties of the PUM as cost-efficiency and approximation accuracy are investigated numerically.

Keywords
Partition of unity method (PUM) · Graph signal processing · Graph basis functions (GBFs) · Kernel-based approximation and interpolation · Spectral graph theory

1 Introduction
Graph signal processing is a cutting-edge research field for the study of graph signals in which mathematical processing tools as filtering, compression, noise removal, sampling, or decomposition methods are investigated \cite{6,11,22–24,26,33,35}. Graph structures appear naturally in a multitude of modern applications, as in social net-
works, traffic maps or biological networks. In general, these networks exhibit a large number of vertices and a highly irregular edge structure. In order to be able to deal with signals on such irregular graphs, efficient and fast processing tools are necessary.

In this work, we introduce and analyze partition of unity methods (PUMs) on graphs as flexible and efficient auxiliary tools to individualize and accelerate signal processing steps. Many algorithms in graph signal processing as, for instance, the calculation of the graph Fourier transform get computationally infeasible if the size of the graph is too large or the topological structure of the network is not sparse. PUMs allow, in an efficient way, to perform operations as signal reconstruction from samples, classification of nodes, or signal filtering locally on smaller portions of the graph, and, then, to rebuild the global signal from the local ones (see Fig. 1 for a simple sketch). This makes a PUM to an ideal auxiliary tool also if more adaptivity is required and processing steps have to be individualized to local spatial prerequisites on the graph.

In recent years, PUMs have been successfully combined with a multitude of different computational methods. In [15,21], for instance, PUMs have been considered in the context of meshfree Galerkin methods to obtain more adaptive and robust solvers for differential equations. In the approximation with radial basis functions (RBFs), the combination of RBFs with PUMs yields significantly sparser system matrices in collocation or interpolation problems, and, therefore, a considerable speed-up of calculations [2–4], [12, Chapter 29], [20,39], [40, Section 15.4].

On graphs, PUMs combined with kernel-based interpolation or approximation techniques have similar benefits: the computations can be reduced, in a highly-flexible way, to local calculations on smaller portions of the graph. In this way, expensive global costs can be avoided and parameters of the approximating scheme can be adjusted to local prerequisites. In this work, we will focus on approximation methods based on generalized translates of a graph basis function (GBF) (see [8,9] for an overview). This kernel-based method is an analog of RBF approximation in the Euclidean space and contains diffusion kernels [19] and variational splines [25,38] on graphs as special cases.

For the construction of partition of unities, the discrete graph structure allows to take advantage of a multitude of clustering algorithms to generate, in an automatic way, a cover of the vertex domain by basic subdomains. The spectral structure induced by the graph Laplacian offers one possibility for such a clustering [36]. In this article, we pursue a second possibility that uses a discrete metric distance on the graph to determine $J$ center nodes for the clusters [14,17]. This metric $J$-center clustering has the advantage that, with a slight modification of the algorithm, the selection of the centers can be restricted to nodes that are relevant for further computations. Once the clusters
are determined, an augmentation procedure can be applied to the clusters in order to obtain an overlapping cover of subdomains. As soon as this cover is determined, the partition of unity itself can be obtained by the use of Shepard weight functions [32].

As for PUMs in Euclidean settings, overlapping subdomains are generally desirable also on graph domains. This allows to incorporate the global geometric information of the graph into the calculations. Further, overlapping domains allow to construct partition of unities with coherent boundary conditions. In Theorem 2, the main theoretical finding of this article, we will show that global approximants based on the PUM inherit smoothness properties of the local approximants on the subdomains once the partition of unity satisfies suitable boundary conditions.

Outline and main contributions of this article
1. In Sect. 2, we give a brief overview on spectral graph theory, the graph Fourier transform and approximation with GBFs.
2. An efficient construction of a partition of unity on metric graphs is provided in Sect. 3. In order to obtain a basic splitting of a graph in \( J \) parts we will use an adaption of metric \( J \)-center clustering. This adaption will guarantee that every cluster contains at least one node with sampling information. In a second step, we augment the \( J \) clusters with neighboring nodes to generate a cover of \( J \) overlapping subdomains. Subordinate to this cover, we show how to generate a partition of unity on the graph.
3. In Algorithm 3 (given in Sect. 4), we combine the partition of unity constructed in Sect. 3 with the GBF approximation scheme outlined in Sect. 2.3. In this way, we are able to merge local GBF approximations on subgraphs to a global GBF-PUM approximation scheme on the graph.
4. In Sect. 5, we prove that, under particular assumptions on the partition of unity, smoothness of the global PUM approximants is inherited from local smoothness. The corresponding main result is stated as a global error estimate for PUM approximation in Theorem 2.
5. In Sect. 6, we conclude this article with some numerical experiments on the cost-efficiency and the accuracy of the PUM.

2 Spectral Graph Theory and Approximation with GBFs

2.1 Preliminaries on Graph Theory

In this article, we will consider simple and connected graphs equipped with a graph Laplacian (providing a harmonic structure on the graph), and a metric distance. More precisely, we define a graph \( G \) as a quadruplet \( G = (V, E, L, d) \) with the following components:

- A vertex set \( V = \{v_1, \ldots, v_n\} \) consisting of \( n \) graph nodes.
- A set \( E \subseteq V \times V \) containing all (directed) vertices \( e_{i,i'} = (v_i, v_{i'}) \), \( i \neq i' \), of the graph. We assume that \( e_{i,i'} \in E \iff e_{i',i} \in E \), i.e., that the graph is undirected.
A general symmetric graph Laplacian \( L \in \mathbb{R}^{n \times n} \) of the form (as, for instance, described in [13, Section 13.9])

\[
L_{i,i'} = \begin{cases} 
0 & \text{if } i \neq i' \text{ and } v_i, v_{i'} \text{ are not connected}, \\
-L_{i,i'} & \text{if } i \neq i' \text{ and } v_i, v_{i'} \text{ are connected, i.e., } e_{i,i'} \in E,
\end{cases}
\]

for \( i \in \{1, \ldots, n\} \).

A symmetric distance metric \( d \) on the vertex set \( V \), satisfying a triangle inequality. A typical example of such a metric is given by the graph geodesic, i.e., the length of the shortest path connecting two graph nodes. We assume that \( G \) is a connected graph and, thus, that the distance \( d \) between two arbitrary nodes is finite.

The negative non-diagonal elements \( L_{i,i'} \), \( i \neq i' \), of the Laplacian \( L \) contain the connection weights of the edges \( e_{i,i'} \in E \). These are usually collected as positive weights in the symmetric adjacency matrix \( A \):

\[
A_{i,i'} := \begin{cases} 
-L_{i,i'} & \text{if } e_{i,i'} \in E, \\
0 & \text{otherwise}.
\end{cases}
\]

In this article, we set no restrictions on the diagonal entries of the graph Laplacian \( L \). There are however some well-known examples in which the diagonal entries of the graph Laplacian are fixed. One example is the negative adjacency matrix \( L_A = -A \) in which all diagonal entries of \( L_A \) vanish. Another more prominent example is the standard graph Laplacian

\[
L_S = D - A 
\]

in which the diagonal is determined by the degree matrix \( D \) given by

\[
D_{i,i'} := \begin{cases} 
\sum_{j=1}^{n} A_{i,j} & \text{if } i = i', \\
0 & \text{otherwise}.
\end{cases}
\]

It is well-known that the standard Laplacian \( L_S \) in (2) is a positive semi-definite matrix and the multiplicity of the eigenvalue \( \lambda_1 = 0 \) corresponds to the number of connected components of the graph \( G \). A further important example is the normalized graph Laplacian \( L_N \) defined by \( L_N = D^{-1/2} L_S D^{-1/2} \) for which all diagonal entries are equal to 1. A more profound introduction to combinatorial graph theory and the properties of different graph Laplacian can be found in [7,13].

In graph signal processing, the main research objects are graph signals. A graph signal \( x : V \to \mathbb{R} \) is a function on the vertices \( V \) of \( G \). We denote the \( n \)-dimensional vector space of all graph signals by \( \mathcal{L}(G) \). As the vertices in \( V \) are ordered, we can represent every signal \( x \) also as a vector \( x = (x(v_1), \ldots, x(v_n))^T \in \mathbb{R}^n \). On the space \( \mathcal{L}(G) \), we have a natural inner product of the form

\[
y^T x := \sum_{i=1}^{n} x(v_i) y(v_i).
\]
The system \{\delta v_1, \ldots, \delta v_n\} of unit vectors forms a canonical orthonormal basis of \(\mathcal{L}(G)\), where the unit vectors \(\delta_{v_i}\) are defined as \(\delta_{v_i}(v_i) = \delta_{i,i'}\) for \(i, i' \in \{1, \ldots, n\}\). In this article, we will also consider functions on the set of edges \(E\) of the graph \(G\). We denote the corresponding space of functions by \(\mathcal{L}(E)\).

### 2.2 The Graph Fourier Transform

An important tool for denoising, filtering and decomposition of graph signals is the graph Fourier transform. On graphs, this transform is defined via the eigendecomposition of the graph Laplacian \(L\):

\[ L = UM_\lambda U^T. \]

Here, \(M_\lambda = \text{diag}(\lambda) = \text{diag}(\lambda_1, \ldots, \lambda_n)\) is the diagonal matrix containing the increasingly ordered eigenvalues \(\lambda_i, \ i \in \{1, \ldots, n\}\), of \(L\) as diagonal entries. The column vectors \(\hat{G} = \{u_1, \ldots, u_n\}\) of the orthonormal matrix \(U\) form an orthonormal basis of eigenvectors for the space \(\mathcal{L}(G)\) with respect to the eigenvalues \(\lambda_1, \ldots, \lambda_n\). We call \(\hat{G}\) the spectrum or the Fourier basis of the graph \(G\). Given the spectrum \(\hat{G}\), the graph Fourier transform of a graph signal \(x\), and the inverse Fourier transform are given as

\[
\hat{x} := U^T x = (u_1^T x, \ldots, u_n^T x)^T, \quad \text{and} \quad x = U \hat{x},
\]

respectively. The entries \(\hat{x}_k = u_k^T x, k \in \{1, \ldots, n\}\), of the Fourier transform \(\hat{x}\) describe the frequency components of the signal \(x\) with respect to the basis functions \(u_k\). As in classical settings like the Euclidean space, the Fourier transform on graphs is a crucial tool for the processing and manipulation of graph signals. Overviews about possible uses of the graph Fourier transform, in particular as analysis or decomposition tool for graph signals, can, for instance, be found in [10,22,24,33,35].

### 2.3 Positive Definite GBFs for Signal Approximation on Graphs

The usage of positive definite GBFs provides a simple and efficient tool for the interpolation and approximation of graph signals if only a few samples \(x(w_i)\) of a graph signal \(x\) are known on a fixed sampling set \(W = \{w_1, \ldots, w_N\} \subset V\), see [8,9]. The usage of GBFs fits into the more general framework of kernel-based approximation techniques for signals on graphs [28], and follows a corresponding theory on scattered data approximation with positive definite RBFs in the Euclidean space [30,40]. In this work, we will exploit GBFs on graphs as local approximation tools for the PUM. In the following, we give a brief introduction into the terminology and the usage of GBFs.

In this method, the approximation spaces are built upon generalized shifts of a GBF \(f\). These are defined in terms of a convolution operator \(C_y\) acting on a signal \(x\) as

\[ C_y x = U M_\lambda U^T x. \]
A function $f \in \mathcal{L}(G)$ is called a positive definite GBF if the matrix

$$
K_f = \begin{pmatrix}
C_{\delta v_1} f(v_1) & C_{\delta v_2} f(v_1) & \ldots & C_{\delta v_n} f(v_1) \\
C_{\delta v_1} f(v_2) & C_{\delta v_2} f(v_2) & \ldots & C_{\delta v_n} f(v_2) \\
\vdots & \vdots & \ddots & \vdots \\
C_{\delta v_1} f(v_n) & C_{\delta v_2} f(v_n) & \ldots & C_{\delta v_n} f(v_n)
\end{pmatrix}
$$

is symmetric and positive definite. Here $\{\delta v_1, \ldots, \delta v_n\}$ denotes the standard unit basis in $\mathcal{L}(G)$.

The signals $C_{\delta v_i} f$ can be interpreted as generalized translates of the basis function $f$ on the graph $G$. In fact, if $G$ has a group structure and the spectrum $\hat{G}$ consists of properly scaled characters of $G$, then the basis functions $C_{\delta v_i} f$ are shifts of the signal $f$ by the group element $v_i$.

$\text{AG B F}$ $f$ is positive definite if and only if $\hat{f}_k > 0$ for all $k \in \{1, \ldots, n\}$ (for the derivations, see [8]). Further, the kernel $K_f(v, w) := C_{\delta w} f(v)$ has the Mercer decomposition

$$
K_f(v, w) = C_{\delta w} f(v) = \sum_{k=1}^{n} \hat{f}_k u_k(v) u_k(w),
$$

and the respective operator matrix $K_f$ can be compactly written as

$$
K_f = \mathbf{U} \hat{M}_f \mathbf{U}^\top.
$$

In this way, a positive definite GBF $f$ induces, in a natural way, an inner product $\langle x, y \rangle_{K_f}$ and a norm $\|x\|_{K_f}$ as

$$
\langle x, y \rangle_{K_f} = \sum_{k=1}^{n} \hat{x}_k \hat{y}_k \frac{\hat{f}_k}{\hat{f}_k} = \hat{y}^\top \frac{1}{\hat{f}} \hat{I} \frac{1}{\mathbf{f}} \frac{1}{\hat{f}} \hat{x} \text{ and } \|x\|_{K_f} = \sqrt{\sum_{k=1}^{n} \hat{x}_k^2 \hat{f}_k}.
$$

The space $\mathcal{L}(G)$ of signals endowed with this inner product is a reproducing kernel Hilbert space $\mathcal{N}_{K_f}$ with reproducing kernel $K_f$ (a systematic study of such Hilbert spaces is given in [1]).

To obtain a GBF approximation $x_*$ of a signal $x$ for a few known samples $x(w_i)$, $i \in \{1, \ldots, N\}$, we consider the solution of the following regularized least squares (RLS) problem (more details are given in [9,27,34])

$$
x_* = \arg\min_{y \in \mathcal{N}_{K_f}} \left( \frac{1}{N} \sum_{i=1}^{N} |x(w_i) - y(w_i)|^2 + \gamma \|y\|_{K_f}^2 \right), \quad \gamma > 0.
$$

(3)

The first term in this RLS functional is referred to as data fidelity term that ensures that the values $x_*(w_i)$ are close to the values $x(w_i) \in \mathbb{R}$, on the sampling set $W = \{w_1, \ldots, w_N\} \subset V$. The second term with the regularization parameter $\gamma > 0$ is called
regularization term. It forces the optimal solution \( x_* \) to have a small Hilbert space norm \( \|x_*\|_{K_f} \). The representer theorem [31, Theorem 4.2] implies that the minimizer \( x_* \) of the RLS functional (3) can be uniquely written as

\[
x_*(v) = \sum_{i=1}^{N} c_i \mathbf{C}_{\delta w_i} f(v),
\]

and is therefore an element of the subspace

\[
\mathcal{N}_{K_f, W} = \left\{ x \in L(G) \mid x = \sum_{i=1}^{N} c_i \mathbf{C}_{\delta w_i} f, \ c_i \in \mathbb{R} \right\}.
\]

Furthermore, the coefficients \((c_1, \ldots, c_N)^T\) in (4) can be calculated as the solution of the linear system (cf. [27], [37, Theorem 1.3.1])

\[
\begin{pmatrix}
\mathbf{C}_{\delta w_1} f(w_1) & \cdots & \mathbf{C}_{\delta w_N} f(w_1) \\
\vdots & \ddots & \vdots \\
\mathbf{C}_{\delta w_1} f(w_N) & \cdots & \mathbf{C}_{\delta w_N} f(w_N)
\end{pmatrix} + \gamma N \mathbf{I}_N \begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} x(w_1) \\ \vdots \\ x(w_N) \end{pmatrix}.
\]

The principal submatrix \( \mathbf{K}_{f, W} \) of the positive definite matrix \( \mathbf{K}_f \) is positive definite by the inclusion principle [18, Theorem 4.3.15]. The linear system (5) has therefore a unique solution and provides a unique GBF approximation \( x_* \).

For a vanishing regularization parameter \( \gamma \to 0 \), the limit \( x_o = \lim_{\gamma \to 0} x_* \) is uniquely determined by the condition (4) and the unique coefficients calculated in (5) with \( \gamma = 0 \). The resulting signal \( x_o \) interpolates the data \((w_i, x(w_i))\), i.e. we have \( x_o(w_i) = x(w_i) \) for all \( i \in \{1, \ldots, N\} \).

**Example 1** Two main examples of positive definite GBFs are the following:

(i) The diffusion kernel (or also heat kernel) on a graph [19] is defined by the Mercer decomposition

\[
\mathbf{K}_{f_{e^{-t L}}} = e^{-t \mathbf{L}} = \sum_{k=1}^{n} e^{-t \lambda_k} u_k u_k^T,
\]

where \( \lambda_k, k \in \{1, \ldots, n\} \), denote the eigenvalues of the graph Laplacian \( \mathbf{L} \). This kernel is positive definite for all \( t \in \mathbb{R} \). The corresponding GBF \( f_{e^{-t L}} \) is uniquely determined by the Fourier transform

\[
\hat{f}_{e^{-t L}} = (e^{-t \lambda_1}, \ldots, e^{-t \lambda_n}).
\]
(ii) Variational or polyharmonic splines on graphs are based on the kernel

\[ K_{f_\epsilon} = (\epsilon I_n + L)^{-s} = \sum_{k=1}^n \frac{1}{(\epsilon + \lambda_k)^s} u_k u_k^T. \]

This kernel is positive definite for \( \epsilon > -\lambda_1 \) and \( s > 0 \). Variational splines are studied in [25,38] as interpolants \( x_0 \) that minimize the energy functional \( x^T (\epsilon I_n + L)^s x \). They can be regarded as GBF interpolants based on the GBF with the Fourier transform

\[ \hat{f}(\epsilon I_n + L)^{-s} = \left( \frac{1}{(\epsilon + \lambda_1)^s}, \ldots, \frac{1}{(\epsilon + \lambda_n)^s} \right). \]

3 How to Obtain a Partition of Unity on Graphs

3.1 Graph Partitioning by Metric \( J \)-center Clustering

A simple way to split a connected graph \( G \) in \( J \) disjoint clusters \( C_j \) such that

\[ V = \bigcup_{j=1}^J C_j, \quad C_j \cap C_{j'} = \emptyset, \quad \text{for } j' \neq j \in \{1, \ldots, J\}, \]

is by minimizing an objective functional that measures the distances of the nodes inside the partitions. For this, a common approach is to consider a metric \( d \) on the graph \( G \) and \( J \) center nodes \( q_j \in C_j \) as reference nodes for the clusters \( C_j \). In metric \( J \)-center clustering (in the literature usually a \( k \) is used instead of the \( J \)), the centers \( Q_J = \{q_1, \ldots, q_J\} \) are determined such that the objective functional

\[ h(Q_J) = \max_{i \in \{1, \ldots, n\}} \min_{j \in \{1, \ldots, J\}} d(q_j, v_i) \]

is minimized. We refer the functional \( h(Q_J) \) to as the fill distance of the node set \( Q_J \). The fill distance \( h(Q_J) \) can be regarded as the smallest radius such that every node \( v \in V \) is contained in at least one ball of radius \( h(Q_J) \) centered at a node in \( Q_J \). Finding the set \( Q_J^{opt} \) that minimizes the fill distance \( h \) is in general a NP-hard problem [17]. A feasible alternative to the calculation of the global minimum \( Q_J^{opt} \) is to use a greedy procedure for the preallocation of the centers [14].

In every iteration \( j \) of this greedy procedure a new node \( q_j \) is added to an already computed set \( Q_{j-1} \subset \{q_1, \ldots, q_{j-1}\} \) such that the new \( q_j \) is farthest away from \( Q_{j-1} \). In view of our application, we also want to guarantee that every cluster \( C_j \) contains at least one sampling node in \( W \). To achieve this, we will only consider center nodes \( q_j \).
that lie in the node set $W$. The corresponding restricted greedy $J$-center clustering is summarized in Algorithm 1.

**Algorithm 1: Restricted greedy $J$-center clustering based on interpolation nodes**

**Input**: The interpolation nodes $W = \{w_1, \ldots, w_N\}$, a starting center $q_1 \in W$ and the number $J$ of partitions.

**for** $j = 2$ to $J$ **do**

- select a center $q_j = \arg\max_{w \in W} \min_{q \in \{q_1, \ldots, q_{j-1}\}} d(q, w)$ farthest away from $Q_{j-1} = \{q_1, \ldots, q_{j-1}\}$.

**Calculate** the $J$ clusters

$$C_j = \{v \in V : d(q_j, v) = \min_{q \in \{q_1, \ldots, q_J\}} d(q, v)\}, \quad j \in \{1, \ldots, J\}.$$ 

**Return** centers $Q_J = \{q_1, \ldots, q_J\}$ and clusters $\{C_1, \ldots, C_J\}$.

**Remark 1** The greedy selection of the node $q_j$ in Algorithm 1 is in general not unique. In case of non-uniqueness, a simple selection rule to obtain $q_j$ out of the maximizers is the lowest-index rule. In this rule, we choose $q_j$ as the admissible node $w_k \in \{w_1, \ldots, w_N\}$ with the lowest index $k$.

If the distribution of the interpolation nodes $W$ inside the graph nodes $V$ is fairly reasonable, the restricted greedy method in Algorithm 1 provides a graph clustering which is close to the optimal solution. This is specified in the following result.

**Theorem 1** Algorithm 1 generates a center set $Q_J$ with $J$ centers and clusters $\{C_1, \ldots, C_J\}$ such that each cluster $C_j$ contains at least one interpolation node of $W$ and the fill distance $h(Q_J)$ is bounded by

$$h(Q_J) \leq 2h(Q_J^{\text{opt}}) + h(W).$$

In particular, if $W$ satisfies $h(W) \leq \epsilon h(Q_J^{\text{opt}})$ for some $\epsilon > 0$ then the greedy method in Algorithm 1 yields a solution $Q_J$ with an objective function $h(Q_J)$ that is within a factor $(2 + \epsilon)$ of the minimal value $h(Q_J^{\text{opt}})$. An algorithm with such a property is also referred to as $(2 + \epsilon)$-approximation algorithm.

**Proof** We proceed similar as in the proof of the original result in which no reduction to a subset is performed, see [16, Theorem 4.3.]. We divide the proof into two disjoint cases:

We assume first that every cluster $C_j^{\text{opt}}, j \in \{1, \ldots, J\}$, based on the optimal centers $Q_j^{\text{opt}}$ contains exactly one center from the set $Q_J$. For an arbitrary node $v \in V$, let $q_j^{\text{opt}}$ be the center of the cluster $C_j^{\text{opt}}$ in which $v$ is contained and let $q_j$ be the unique node in $Q_J$ which is also contained in $C_j^{\text{opt}}$. Then, the triangle inequality gives

$$d(v, q_j) \leq d(v, q_j^{\text{opt}}) + d(q_j^{\text{opt}}, q_j) \leq 2h(Q_J^{\text{opt}}).$$
In the second case, there exists a cluster $C^\text{opt}_j$ (with center $q^\text{opt}_j$) that contains two distinct elements $q_j$ and $q_k$ of the set $Q_J$. Without loss of generality we assume that $k < j$, and that $q_j$ is added later to the center set $Q_j$ in the $j$-th iteration of the greedy algorithm. The greedy selection in Algorithm 1 picks a node $q_j$ in the interpolation set $W$ that is farthest away from $Q_j-1$. Let $v^\text{opt}_j \in V$ be an element that maximizes

$$h(Q_j-1) = \max_{v \in V} \min_{l \in \{1, \ldots, j-1\}} d(q_l, v).$$

Further, let $w_j$ be the element in $W$ that is closest to $v^\text{opt}_j$. Then

$$h(Q_J) \leq h(Q_j-1) = \min_{l \in \{1, \ldots, j-1\}} d(v^\text{opt}_j, q_l) \leq \min_{l \in \{1, \ldots, j-1\}} d(w_j, q_l) + d(v^\text{opt}_j, w_j) \leq d(q_j, q_k) + d(v^\text{opt}_j, w_j) \leq 2h(Q^\text{opt}_J) + h(W).$$

\[ \square \]

3.2 Augmentation of Graph Clusters

Although it is possible to use the disjoint clusters $C_j$, $j \in \{1, \ldots, J\}$, directly as subdomains for a partition of unity on the vertex set $V$, this is not recommendable, as the topological information of the graph $G$ gets lost. Instead, we will enlarge the clusters $C_j$ to overlapping subdomains $V_j$ such that the partition of unity is subsequently generated subordinate to the covering given by the subdomains $V_j$, $j \in \{1, \ldots, J\}$. The relevance of this domain augmentation step will be visible in the numerical experiments performed in Sect. 6, as well as in the theoretical error estimates of Theorem 2.

The enlargement of the clusters $C_j$ can be performed in several ways. In the method described in Algorithm 2, we enlarge the clusters $C_j$ by all vertices of $V$ that have a graph distance to some element of $C_j$ less or equal to $r \geq 0$. Note that the choice of the augmentation distance $r$ has an impact on the performance of the PUM. Our numerical tests in Sect. 6 indicate that the parameter $r$ must be chosen large enough to guarantee a proper approximation power of the method. On the other hand, it is not desirable to enlarge the clusters $C_j$ too much, as the overall costs of the computations on the subdomains $V_j$ increase with increasing $r$. 

\[ \Diamond \]
Algorithm 2: Augmentation of clusters with neighbors of distance \( r \)

**Input:** A cluster \( C_j \) and an enlargement distance \( r \geq 0 \).

**Add** all nodes \( v \in V \) with a graph distance less than \( r \) to an element of \( C_j \), i.e.,

\[
V_j = C_j \cup \{ v \in V \mid d(v, w) \leq r, w \in C_j \}.
\]

**Return** subdomain \( V_j \).

**Corollary 1** The cover \( \{ V_j \}_{j=1}^J \) of the node set \( V \) generated by Algorithms 1 and 2 has the following properties:

(i) Each subdomain \( V_j \) contains a center node \( q_j \) which is also an element of the sampling set \( W \).

(ii) The distance of each node \( v \in V_j \) to the center node \( q_j \) is at most \( h(Q_J) + r \). In particular, the diameter of each subdomain \( V_j \) is bounded by \( 2h(Q_J) + 2r \).

### 3.3 Generating the Partition of Unity

Once a cover \( \{ V_j \}_{j=1}^J \) of the vertex set \( V \) is determined, our next objective is to construct a partition of unity \( \{ \varphi(j) \}_{j=1}^J \) subordinate to this cover, i.e., a set of functions \( \varphi(j) \in \mathcal{L}(G), j \in \{1, \ldots, J\} \), such that

\[
\text{supp}(\varphi(j)) \subseteq V_j, \quad \varphi(j) \geq 0, \quad \text{and} \quad \sum_{j=1}^J \varphi(j)(v) = 1 \quad \text{for all } v \in V.
\]

A simple construction principle to generate such a set of functions is based on Shepard weights [32]: if \( \psi(j), j \in \{1, \ldots, J\} \), denote non-negative weight functions on \( V \) with \( \text{supp}(\psi(j)) \subseteq V_j \) and \( \sum_{j=1}^J \psi(j) > 0 \), then the functions

\[
\varphi(j)(v) = \frac{\psi(j)(v)}{\sum_{j=1}^J \psi(j)(v)}, \quad v \in V,
\]

form a partition of unity subordinate to \( \{ V_j \}_{j=1}^J \).

**Example 2** (i) Choosing \( \psi(j)(v) = \mathbf{1}_{V_j}(v) \), we get the partition of unity

\[
\varphi(j)(v) = \frac{\mathbf{1}_{V_j}(v)}{\# \{j \in \{1, \ldots, J\} \mid v \in V_j \}}.
\]

(ii) We consider the disjoint clusters \( C_j, j \in \{1, \ldots, J\} \), from Algorithm 1 as subsets of the augmented domains \( V_j \). Then, choosing \( \psi(j)(v) = \mathbf{1}_{C_j}(v) \), the corresponding partition of unity is given as

\[
\varphi(j)(v) = \psi(j)(v) = \mathbf{1}_{C_j}(v).
\]
4 A Partition of Unity Method for GBF Approximation on Graphs

4.1 The Algorithm

In this section, we show how a PUM can be implemented in order to obtain a global approximation on the graph from local GBF approximations on subgraphs $G_j$. We will mainly focus on the construction and the computational steps to obtain the global approximation, the local approximation procedure is already outlined in Sect. 2.3. The main computational steps, elaborated formally in Algorithm 3, are the following:

1. In a preliminary step, a cover $\{V_j\}_{j=1}^J$ of the node set $V$ and a partition of unity $\{\varphi(j)\}_{j=1}^J$ subordinate to $\{V_j\}_{j=1}^J$ is constructed. In particular, the following operations are performed:

   (a) Algorithm 1 is applied to decompose $V$ into $J$ disjoint clusters $C_j$.
   (b) Algorithm 2 is applied to augment the clusters $C_j$ with neighboring nodes and obtain a cover of overlapping subdomains $V_j$, $j \in \{1, \ldots, J\}$.
   (c) As described in Sect. 3.3, Shepard weight functions are introduced on the cover $\{V_j\}_{j=1}^J$ to obtain a desired partition of unity $\{\varphi(j)\}_{j=1}^J$.

2. Once the cover $\{V_j\}_{j=1}^J$ is determined, we extract in a second preliminary step the subgraphs $G_j = (V_j, E_j, L^{(j)}, d)$, where $V_j$ is the set of vertices, $E_j = \{e_{ij} \in E \mid v_i, v_j \in V_j\}$ denotes the set of edges connecting two nodes in $V_j$, and the local Laplacian $L^{(j)}$ is the principal submatrix of $L$ corresponding to the nodes of $V_j$. Further, the metric $d$ on $G_j$ is the induced metric. The $J$-center clustering in Algorithm 1 and the augmentation procedure in Algorithm 2 ensure that the subgraphs $G_j$ are connected.

3. The main computational step consists in the calculation of the local approximations $x^{(j)}_s$ on the subgraphs $G_j$. As outlined in Sect. 2.3, this can be done by using a GBF approximation scheme. To this end, on each subgraph a positive definite GBF $f^{(j)}$ has to be chosen. As sampling information for the calculation of the approximation, the samples of the signal $x$ on the subset $W_j = W \cap V_j$ are taken. The modified $J$-center clustering in Algorithm 1 ensures that $W_j \neq \emptyset$.

4. Once the local approximations $x^{(j)}_s$ are computed, the global GBF-PUM approximation on $G$ is determined by

$$x_s(v) = \sum_{j=1}^J \varphi^{(j)}(v)x_s^{(j)}(v).$$

In a similar way, a global interpolating function $x_s(v)$ can be calculated by using local interpolating functions $x_s^{(j)}$ instead of the approximants $x^{(j)}_s$. Finally, as described in [9], the approximant $x_s$ can be processed further by selection functions in order to obtain, for instance, a global classification algorithm for the nodes of the graph $G$. 
Algorithm 3: GBF-PUM approximation on graphs

Input: (i) Samples \( x(w_1), \ldots, x(w_N) \in \mathbb{R} \) at the interpolation nodes \( W = \{w_1, \ldots, w_N\} \), (ii) Number \( J \) of subdomains.

1.a) Metric \( J \)-center clustering on \( G \)
Apply Algorithm 1 to decompose \( V \) into \( J \) disjoint clusters \( C_j \).

1.b) Create \( J \) subdomains \( V_j \)
Apply Algorithm 2 to augment the clusters \( C_j \) into \( J \) overlapping subdomains \( V_j = \{v_{j1}, \ldots, v_{jn_j}\} \) with \( n_j \) elements.

1.c) Generate Partition of Unity
Subordinate to the cover \( \{V_j\}_{j=1}^J \) of \( V \) construct a partition of unity \( \{\phi(j)\}_{j=1}^J \) such that
\[
\text{supp}(\phi(j)) \subseteq V_j, \quad \phi(j) \geq 0, \quad \text{and} \quad \sum_{j=1}^J \phi(j)(v) = 1 \text{ for all } v \in V.
\]

2. Extract local graphs and local Laplacians \( L^{(j)} \)
For all subdomains \( V_j, j \in \{1, \ldots, J\} \), generate the subgraphs \( G_j = (V_j, E_j, L^{(j)}, d) \), with the node sets \( V_j \), the edges \( E_j = \{(v, w) \in E \mid v, w \in V_j\} \) and the local Laplacian
\[
L^{(j)} = \begin{pmatrix}
L_{1,1}^{(j)} & \cdots & L_{1,n_j}^{(j)} \\
\vdots & \ddots & \vdots \\
L_{n_j,1}^{(j)} & \cdots & L_{n_j,n_j}^{(j)}
\end{pmatrix}.
\]

If required, calculate the spectral decomposition \( L^{(j)} = U^{(j)} M^{(j)} U^{(j)\top} \) to define a local graph Fourier transform on the subgraphs \( G_j \).

3.a) Construct the local GBF kernel \( K^{(j)}_{f^{(j)}}(v, w) = C^{(j)}_{f^{(j)}}(v) \) on the subgraph \( G_j \). One possibility is to use the variational spline kernel
\[
K^{(j)}_{f^{(j)}} = (\epsilon I_n + L^{(j)})^{-s} = \sum_{k=1}^{n_j} \frac{1}{(\epsilon + \lambda^{(j)}_k)^s} u^{(j)}_k u^{(j)\top}_k, \quad \epsilon > -\lambda^{(j)}_1, \quad s > 0.
\]

3.b) For the sampling sets \( W_j = W \cap V_j \) with \( N_j \geq 1 \) elements, and \( \gamma \geq 0 \), solve
\[
(K^{(j)}_{f^{(j)}} v^{(j)}_{w_j} + \gamma N_j I_{N_j}) \begin{pmatrix}
\psi^{(j)}_{1} \\
\vdots \\
\psi^{(j)}_{N_j}
\end{pmatrix} = \begin{pmatrix}
x(w_{j1}) \\
\vdots \\
x(w_{jN_j})
\end{pmatrix}.
\]

3.c) Calculate the local GBF approximation \( x^{(j)}_{w_j} \) on \( G_j \):
\[
x^{(j)}_{w_j}(v) = \sum_{i=1}^{N_j} c^{(j)}_{w_j} f^{(j)}(v).
\]

4. A global GBF-PUM approximation on \( G \) is then given as
\[
x^{(j)}(v) = \sum_{j=1}^J \phi(j)(v)x^{(j)}_{w_j}(v).
\]
4.2 Complexity of GBF-PUM Approximation on Graphs

We analyze the principal computational cost of the GBF-PUM approximation procedure described in Algorithm 3 and compare it to the complexity of a global GBF approximation, as for instance used in [8].

The essential costs in both approximation schemes are generated by the steps described in 3.a) and 3.b) of Algorithm 3, i.e. by the calculation of the basis functions $C_{\delta w f}$ and $C_{\delta w f}(j)$, respectively, and the solution of the linear system (5) for the expansion coefficients in this basis. While for a global GBF approximation scheme these two steps have to be executed on the entire graph, in the partition of unity method these calculations can be reduced to the $J$ subgraphs $G_j$.

**Calculation of the basis functions** The computational complexity of this step depends on the algorithmic procedure to determine the kernels $K_f$ and $K_f(j)$, respectively. In the worst case, the entire spectral decomposition of the graph Laplacian $L$ (respectively) is required. For the calculation on the entire graph $G$ this means $O(n^3)$ arithmetic operations for the decomposition of the graph Laplacian $L$, while in step 3.a) of Algorithm 3 we would require $\sum_{j=1}^{J} O(n_j^3) \leq J \cdot O(\max_j(n_j)^3)$ for the PUM. If all the subdomains $V_j$ are of the same size $n_j \sim n/J$, the PUM leads to a significant reduction of the computational costs in this case.

**Calculation of the expansion coefficients** Regarding the complexity of the calculation of the coefficients $(c_1, \ldots, c_N)^T$, we observe that the linear system (5) for the expansion coefficients can be computed in $O(N^3)$ arithmetic operations in the global setting, being $N$ the number of interpolation nodes. In the local settings provided by the PUM the calculations are less expensive. In fact, in step 3.a) of Algorithm 3 we have to solve $J$ linear systems of equations with dimension equal to $N_j$. In this case $J$ solutions must be computed, each with a cost of $O(N_j^3)$ arithmetic operations. As all matrices $K_f(j), W_j$ are positive definite, a straightforward option is to apply the Cholesky factorization to obtain the solutions of these systems.

5 From Local to Global Error Estimates in PUM Approximation

A partition of unity induces naturally the following approximation spaces:

**Definition 1** Let $\{V_j\}_{j=1}^{J}$ be a cover of the set $V$ and $\{\varphi(j)\}_{j=1}^{J}$ a subordinate partition of unity. Let $G_j = (V_j, E_j, L^{(j)}, d)$ be the subgraphs of $G$ based on the node sets $V_j$, the edges $E_j = \{(v, w) \in E \mid v, w \in V_j\}$ and the local Laplacian $L^{(j)}$. Further, for the subgraphs $G_j$, let $\mathcal{N}^{(j)} \subset \mathcal{L}(G_j)$ be given local approximation spaces. Then, we define the global partition of unity approximation space $\mathcal{N}^{PUM}$ as

$$\mathcal{N}^{PUM} = \left\{ x \in \mathcal{L}(G) \mid x = \sum_{j=1}^{J} \varphi(j)x^{(j)} , x^{(j)} \in \mathcal{N}^{(j)} \right\}.$$

In the following, our goal is to derive approximation properties of the spaces $\mathcal{N}^{PUM}$ from given local estimates on the subgraphs $G_j$. As a measure for the smoothness of
a function \( x \in \mathcal{L}(G) \), we will consider norms of the Laplacian \( \mathbf{L} \) applied to \( x \) or, alternatively, the gradient \( \nabla \mathbf{L}x \). The weighted gradient \( \nabla \mathbf{L}x \) is defined as a function on the (directed) edges \( e_{i,i'} = (v_i, v_{i'}) \in E \) of the graph \( G \) as
\[
\nabla \mathbf{L}x(e_{i,i'}) = \sqrt{A_{i,i'}(x(v_i) - x(v_{i'}))).
\]  

Based on this definition of the gradient, we can decompose the standard graph Laplacian \( \mathbf{L}_S \) as
\[
x^T \mathbf{L}_S x = \frac{1}{2} \sum_{i=1}^{n} \sum_{i':e_{i,i'} \in E} A_{i,i'} \left( x(v_i) - x(v_{i'}) \right)^2 = (\nabla \mathbf{L}x)^T \nabla \mathbf{L}x,
\]
showing, in particular, that the standard Laplacian \( \mathbf{L}_S \) is positive semi-definite. As possible norms for functions \( x \in \mathcal{L}(G) \) on the graph \( G \) and for functions \( z \in \mathcal{L}(E) \) on the edges of \( G \), we consider for \( 1 \leq p \leq \infty \), the usual \( \mathcal{L}^p \)-norms given by
\[
\|x\|_{\mathcal{L}^p(G)} := \begin{cases} 
\left( \frac{1}{n} \sum_{i=1}^{n} |x(v_i)|^p \right)^{1/p} & \text{for } 1 \leq p < \infty, \\
\max_{i \in \{1,\ldots,n\}} |x(v_i)| & \text{for } p = \infty,
\end{cases}
\]

\[
\|z\|_{\mathcal{L}^p(E)} := \begin{cases} 
\left( \frac{1}{|E|} \sum_{i,i':e_{i,i'} \in E} |z(e_{i,i'})|^p \right)^{1/p} & \text{for } 1 \leq p < \infty, \\
\max_{i,i':e_{i,i'} \in E} |z(e_{i,i'})| & \text{for } p = \infty.
\end{cases}
\]

In addition, we require for functions \( z \in \mathcal{L}(E) \) and \( 1 \leq p < \infty \) also the hybrid norms \( \|z\|_{\mathcal{L}^p(E)} \) given by
\[
\|z\|_{\mathcal{L}^p(E)} := \max_{i \in \{1,\ldots,n\}} \left( \frac{1}{|E|} \sum_{i,i':e_{i,i'} \in E} |z(e_{i,i'})|^p \right)^{1/p}.
\]

For \( p = \infty \), a corresponding definition of this hybrid norm reduces to the already introduced \( \mathcal{L}^\infty \)-norm \( \|z\|_{\mathcal{L}^\infty(E)} \).

Our main result reads as follows:

**Theorem 2** Let \( \{V_j\}_{j=1}^J \) be a cover of \( V \) and \( \{\phi^{(j)}\}_{j=1}^J \) a subordinate partition of unity. For a signal \( x \in \mathcal{L}(G) \) we assume that there exist local approximations \( x^{(j)}_* \in \mathcal{L}(G_j) \) such that, for given \( 1 \leq p \leq \infty \), either the bound (i), the bounds (i), (ii), or the bounds (i), (ii), (iii) are satisfied, where

(i) \( \|x - x^{(j)}_*\|_{\mathcal{L}^p(G_j)} \leq \epsilon^{(j)}_p \),

(ii) \( \|\nabla \mathbf{L}^{(j)}x - \nabla \mathbf{L}^{(j)}x^{(j)}_*\|_{\mathcal{L}^p(E)} \leq \epsilon^{(j)}_p \),

(iii) \( \|\mathbf{L}^{(j)}x - \mathbf{L}^{(j)}x^{(j)}_*\|_{\mathcal{L}^p(G_j)} \leq \epsilon^{(j)}_p \).
If (ii) or (iii) are satisfied, we further assume that the partition of unity \( \{ \varphi^{(j)} \}_{j=1}^{J} \) fulfills the boundary conditions

\[
(BC) \quad \nabla_{L} \varphi^{(j)}(e_{i,i'}) = 0 \quad \text{for all edges } e_{i,i'} \in E \text{ with } v_{i} \in V_{j} \text{ and } v_{i'} \notin V_{j}.
\]

Then, we have the following error bounds for

\[
x^{*} = \sum_{j=1}^{J} \varphi^{(j)} x^{(j)} \in \mathcal{N}^{PUM};
\]

\[\begin{align*}
(i)' \quad &\| x - x^{*} \|_{L^{p}(G)} \leq \sum_{j=1}^{J} \mathcal{E}^{(j)}_{p,0} \quad \text{if (i) holds true,} \\
(ii)' \quad &\| \nabla_{L} x - \nabla_{L} x^{*} \|_{L^{p}(E)} \leq \sum_{j=1}^{J} \left( \mathcal{E}^{(j)}_{p,1} + \| \nabla_{L} \varphi^{(j)} \|_{L^{\infty}(E)} \mathcal{E}^{(j)}_{p,0} \right) \quad \text{if (i), (ii) and (BC) hold true,} \\
(iii)' \quad &\| L x - L x^{*} \|_{L^{p}(G)} \leq \sum_{j=1}^{J} \left( \mathcal{E}^{(j)}_{p,2} + \| \nabla_{L} \varphi^{(j)} \|_{L^{\infty}(E)} \mathcal{E}^{(j)}_{p,1} + \| L S \varphi^{(j)} \|_{L^{\infty}(G)} \mathcal{E}^{(j)}_{p,0} \right) \quad \text{if (i), (ii), (iii) and (BC) hold true.}
\end{align*}\]

In (iii)', the number \( 1 \leq q \leq \infty \) is dual to \( p \) and determined by \( \frac{1}{p} + \frac{1}{q} = 1 \).

**Remark 2** According to Theorem 2, we can expect small global errors if the local errors are small and if the partition of unity is properly constructed. The latter includes that the functions \( \varphi^{(j)} \) satisfy the boundary conditions (BC), and that the norms \( \| \nabla_{L} \varphi^{(j)} \|_{L^{\infty}(E)} \) and \( \| L S \varphi^{(j)} \|_{L^{\infty}(G)} \) are small. In analogy to continuous frameworks, this indicates that the functions \( \varphi^{(j)} \) of the partition should decay smoothly and vanish at the boundary of the subdomains \( V_{j} \) in order to guarantee small global errors.

**Remark 3** (Implications of (BC)) Assuming that we have an edge from \( v_{i} \) to \( v_{i'} \) with \( v_{i} \in V_{j}, v_{i'} \notin V_{j} \), then the property \( \text{supp}(\varphi^{(j)}) = \{ v \in G : \varphi^{(j)}(v) \neq 0 \} \subseteq V_{j} \) implies that \( \varphi^{(j)}(v_{i'}) = 0 \). Hence, the boundary condition (BC) implies that also \( \varphi^{(j)}(v_{i}) = 0 \).

Therefore, in order to guarantee that (BC) is satisfied and that \( \{ \varphi^{(j)} \}_{j=1}^{J} \) is a subordinate partition of unity, the subdomains \( V_{j} \) of the cover have to overlap to some extend. A partition \( \{ \varphi^{(j)} \}_{j=1}^{J} \) satisfying (BC) is, for instance, provided in Example 2 (ii) as soon as the enlargement distance \( r \) in Algorithm 2 is chosen large enough. The relevance of this vanishing boundary condition and the enlargement parameter \( r \) for the global error is further illustrated in the numerical experiments of Sect. 6.

### 5.1 Examples of Error Estimates for Bandlimited Signals

So far, we have not considered the local error terms \( \mathcal{E}^{(j)}_{p,0}, \mathcal{E}^{(j)}_{p,1} \) and \( \mathcal{E}^{(j)}_{p,2} \) in Theorem 2. They depend on the chosen local approximation schemes and on the function class they are approximating. In the following, we give two examples on how a particular choice of the GBF and the corresponding local approximation space lead to global error estimates. As smooth test functions we will consider the following two signal

\[
\phi
\]
spaces:

\[ B_M := \left\{ x \in \mathcal{L}(G) \left| x = \sum_{k=1}^{M} \alpha_k u_k, \alpha_k \in \mathbb{R} \right. \right\}, \]

\[ B_{M}^{\text{PUM}} := \left\{ x \in \mathcal{L}(G) \left| x = \sum_{j=1}^{J} \varphi^{(j)} x^{(j)}, x^{(j)} \in B_{M}^{(j)} \right. \right\}. \]

The first space \( B_M \) is spanned by the eigenvectors \( u_k, 1 \leq k \leq M \), of the graph Laplacian \( L \) and is usually referred to as space of bandlimited functions on \( G \). The second space is a PUM variant of the previous space in which local bandlimited functions in \( B_{M}^{(j)} \) are combined to a global signal via the partition of unity. For the error estimates of the local approximation, we will consider the variational spline kernel given in Example 1 (ii). Error bounds for these variational splines have been proven in [25]. To apply the bounds from [25], we will use the standard graph Laplacian \( L = L_S \) as well as the following notion of \( \Lambda \)-set: for \( \Lambda > 0 \) a subset \( W \subset V \) is called a \( \Lambda \)-set if for any \( x \in \mathcal{L}(G) \) with \( \text{supp}(x) \subset W \) the following Poincaré inequality is satisfied:

\[ \| x \|_2 \leq \Lambda \| Lx \|_2. \]

**Error estimates for signals in** \( B_{M}^{\text{PUM}} \). Using the variational spline GBF \( f_{\epsilon I_n+L^{(j)}-s} \) as basis for the interpolation \( x_{\epsilon}^{(j)} \) of a single part \( x^{(j)} \in \mathcal{L}(G_j) \) of the signal \( x \in B_{M}^{\text{PUM}} \), we get the following \( \mathcal{L}^2 \)-error estimates.

**Corollary 2** Let \( \{ V_j \}_{j=1}^{J} \) be a cover of \( V \) and \( \{ \varphi^{(j)} \}_{j=1}^{J} \) a subordinate partition of unity. Let \( W \) be a set of interpolation nodes such that the complements \( V_j \setminus W_j \) are \( \Lambda \)-sets on the subgraphs \( G_j \), respectively. Further, let \( s = 2^l, l \in \mathbb{N}, 0 < \epsilon < \frac{1}{\Lambda} \), and assume that for \( M \in \mathbb{N} \) we have

\[ \lambda^{(j)}_M + \epsilon < \frac{1}{\Lambda}, \quad j \in \{1, \ldots, J\}, \]

where \( \lambda^{(j)}_M \) denotes the \( M \)-th eigenvector of the Laplacian \( L^{(j)} \). For a signal \( x \in B_{M}^{\text{PUM}} \) we denote by \( x_{\epsilon}^{(j)} \in \mathcal{N}_{K_{j}}^{(j),W_j} \) the GBF-interpolant of \( x^{(j)} \) in \( \mathcal{L}(G_j) \) generated by the local variational spline GBF \( f^{(j)} = f_{\epsilon I_n+L^{(j)}-s} \). Then, the \( \mathcal{L}^2 \)-error between \( x \) and the global interpolant \( x_{\epsilon} = \sum_{j} \varphi^{(j)} x^{(j)} \) is bounded by

\[ \| x - x_{\epsilon} \|_{\mathcal{L}(G)} \leq 2 J \gamma^s \| x \|_{\mathcal{L}(G)}, \quad \text{where} \quad \gamma = \Lambda (\max_j \lambda^{(j)}_M + \epsilon) < 1, \ s = 2^l. \]

**Proof** The prerequisites in Corollary 2 guarantee that [25, Theorem 1.1] can be applied in order to get the local estimates.
\[ \| x^{(j)} - x_0^{(j)} \|_{L^2(G_j)} < 2\gamma^s \| x^{(j)} \|_{L^2(G_j)}. \]

Now, Theorem 2 (i)' provides the aimed-at result. \(\square\)

**Error estimates for signals in** \(B_M\). In combination with variational splines, the PUM performs well in the approximation of bandlimited signals \(x \in B_M\). We explore this further in the numerical experiments of Sect. 6, but we will also give a theoretical indication right now based on a second error estimate for signals in \(B_M\). Compared to Algorithm 3 and the setting of Corollary 2, we will use a slightly altered way to calculate the local interpolants \(x^{(j)}_\circ\): instead of using local basis functions \(f^{(j)}\) for the approximation, we apply a global GBF \(f\) in order to compute the local interpolation on the subgraph \(G_j\). This leads to a larger computational cost for the PUM, gives us however a suitable basis for approximating signals in \(B_M\).

**Corollary 3** Let \(\{V_j\}_{j=1}^J\) be a cover of \(V\) and \(\{\varphi^{(j)}\}_{j=1}^J\) a respective subordinate partition of unity. Let \(W\) be a set of interpolation nodes such that all complements \(V \setminus W_j = V \setminus (W \cap V_j)\) are \(\Lambda\)-sets on the graph \(G\). Further, let \(s = 2^l\), \(l \in \mathbb{N}\), \(0 < \epsilon < \frac{1}{\Lambda}\) and assume that for \(M \in \mathbb{N}\) we have

\[ \lambda_M + \epsilon < \frac{1}{\Lambda}, \]

where \(\lambda_M\) denotes the \(M\)-th. eigenvector of the Laplacian \(L\). For a signal \(x \in B_M\) we denote by \(x^{(j)}_\circ \in N_{K_f,W_j}\) the GBF- interpolant that interpolates \(x\) on the subset \(W_j\) based on the variational spline GBF \(f = f(\epsilon L_n + L)^{-s}\) given in Example 1 (ii). Then, the \(L^2\)-error between \(x\) and the corresponding global PUM interpolant \(x_\circ\) is bounded by

\[ \| x - x_\circ \|_{L^2(G)} \leq 2J\gamma^s \| x \|_{L^2(G)}, \quad \text{with} \quad \gamma = \Lambda(\lambda_M + \epsilon) < 1. \]

**Proof** The prerequisites of Corollary 3 guarantee also in this case that [25, Theorem 1.1] can be applied. In this way, we obtain the local error estimates

\[ \| x^{(j)} - x_\circ^{(j)} \|_{L^2(G_j)} < 2\gamma^s \| x^{(j)} \|_{L^2(G_j)}. \]

Again, Theorem 2 (i)' provides the desired result. \(\square\)

Compared to the global error estimate for variational spline interpolation in [25, Theorem 1.1], the PUM error estimates in Corollaries 2 and 3 have an additional factor \(J\). On the other hand, in Sect. 4.2 we saw that a PUM in general requires a lower computational effort. This is a trade-off that we will also encounter in the numerical experiments of the next section. We finally note that further error estimates for GBF interpolation can be found in [8]. Similar as in Corollaries 2 and 3, also these can be applied to obtain global error estimates for the PUM approximation.
5.2 Proof of Theorem 2

In order to prove Theorem 2, we require a product rule for the graph Laplacian \( L \) and the gradient \( \nabla_L \) when these are applied to the pointwise product \( xy \) of two graph signals.

**Lemma 1** For two signals \( x, y \in L(G)\), we have the following product rules for the gradient \( \nabla_L \) and the graph Laplacian \( L \):

(i) \[
\nabla_L(xy)(e_{i,i'}) = \nabla_L x(e_{i,i'})y(v_i) + x(v_{i'})\nabla_L y(e_{i,i'})
\]

(ii) \[
L(xy)(v_i) = Lx(v_i)y(v_i) + x(v_{i'})L_S y(v_i) + \sum_{i': e_{i,i'} \in E} L_{i,i'}(x(v_i) - x(v_{i'}))(y(v_i) - y(v_{i'})).
\]

**Proof** (i) The gradient \( \nabla_L \) defined in (6) and applied to the product \( xy \) yields

\[
\nabla_L(x y)(e_{i,i'}) = \left(A_{i,i'}\right)^{\frac{1}{2}}(x(v_i)y(v_i) - x(v_{i'})y(v_{i'}))
\]

\[
= \left(A_{i,i'}\right)^{\frac{1}{2}}(x(v_i)y(v_i) - x(v_{i'})y(v_{i'}) + x(v_{i'})y(v_i) - x(v_{i})y(v_{i}))
\]

\[
= \nabla_L x(e_{i,i'})y(v_i) + x(v_{i'})\nabla_L y(e_{i,i'}).\n\]

(ii) Similarly, the definition of the graph Laplacian \( L \) applied to the product \( xy \) of the signal \( x \) and \( y \) gives

\[
L(xy)(v_i) = L_{i,i}x(v_i)y(v_i) + \sum_{i': e_{i,i'} \in E} L_{i,i'}x(v_{i'})y(v_{i'})
\]

\[
= Lx(v_i)y(v_i) + \sum_{i': e_{i,i'} \in E} L_{i,i'}x(v_{i'})y(v_{i')}. \tag{7}
\]

As the entries of the standard graph Laplacian \( L_S \) are given as

\[
(L_S)_{i,i'} = \begin{cases} 
\sum_{j=1}^n A_{i,j}, & \text{if } i = i', \\
L_{i,i'} = -A_{i,i'}, & \text{otherwise},
\end{cases}
\]

we further have the identity

\[
x(v_i)L_S y(v_i) = \sum_{i': e_{i,i'} \in E} L_{i,i'}x(v_i)(y(v_{i'}) - y(v_{i})).
\]

Combining this identity with the formula in (7), we obtain

\[
L(xy)(v_i) = Lx(v_i)y(v_i) + x(v_i)L_S y(v_i)
\]
The product rule in Lemma 1 implies for Lemma 2 and, thus, the second statement.

\[ \text{Lemma 2 } \]

The product rule in Lemma 1 implies for \( 1 \leq p, q \leq \infty, \frac{1}{p} + \frac{1}{q} = 1, \) the following \( \mathcal{L}^p \)-norm estimates:

(i) \( \| \nabla L(x, y) \|_{\mathcal{L}^p(E)} \leq \| \nabla Lx \|_{\mathcal{L}^p(G)} \| y \|_{L^\infty(G)} + \| x \|_{\mathcal{L}^p(G)} \| \nabla Ly \|_{L^\infty(E)} \).

(ii) \( \| L(x, y) \|_{\mathcal{L}^p(G)} \leq \| Lx \|_{\mathcal{L}^p(G)} \| y \|_{L^\infty(G)} + \| x \|_{\mathcal{L}^p(G)} \| Ls y \|_{L^\infty(G)} + \| \nabla Lx \|_{\mathcal{L}^p(E)} \| \nabla L y \|_{L^\infty(E)} \).

**Proof** In general lines, the two statements follow from the product formulas in Lemma 1 in combination with Hölder’s inequality and the triangle inequality for the \( \mathcal{L}^p \)-norm. More precisely, we get for \( 1 \leq p < \infty \) (for \( p = \infty \) the argumentation line is analog) the estimates

\[
\| \nabla L(x, y) \|_{\mathcal{L}^p(E)} = \left( \sum_{i, i': e_{i, i'} \in E} \left| \nabla Lx(e_{i, i'}) y(v_i) + x(v_{i'}) \nabla Ly(e_{i, i'}) \right|^p \right)^{1/p} 
\leq \| \nabla Lx \|_{\mathcal{L}^p(E)} \| y \|_{L^\infty(G)} + \left( \sum_{i' = 1}^n |x(v_{i'})|^p \right)^{1/p} \left( \sum_{i: e_{i, i'} \in E} \left| \nabla Ly(e_{i, i'}) \right|^p \right)^{1/p} 
\leq \| \nabla Lx \|_{\mathcal{L}^p(E)} \| y \|_{L^\infty(G)} + \| x \|_{\mathcal{L}^p(G)} \| \nabla L y \|_{L^\infty(E)},
\]

and

\[
\| L(x, y) \|_{\mathcal{L}^p(G)} \leq \| Lx \|_{\mathcal{L}^p(G)} + \| x \|_{\mathcal{L}^p(G)} \| \nabla L \|_{\mathcal{L}^p(E)} + \left( \sum_{i = 1}^n \left( \sum_{i': e_{i, i'} \in E} L_{i, i'} \left( x(v_{i'}) - x(v_i) \right) \left( y(v_{i'}) - y(v_i) \right) \right)^p \right)^{1/p} 
\leq \| Lx \|_{\mathcal{L}^p(G)} + \| x \|_{\mathcal{L}^p(G)} \| Ls y \|_{\mathcal{L}^p(G)} + \left( \sum_{i = 1}^n \left( \sum_{i': e_{i, i'} \in E} | \nabla Lx(e_{i, i'}) |^p \left( \sum_{i': e_{i, i'} \in E} \left| \nabla Ly(e_{i, i'}) \right|^q \right) \right)^{\frac{p}{q}} \right)^{\frac{1}{p}} 
\leq \| Lx \|_{\mathcal{L}^p(G)} \| y \|_{L^\infty(G)} + \| x \|_{\mathcal{L}^p(G)} \| Ls y \|_{\mathcal{L}^p(G)} + \| \nabla Lx \|_{\mathcal{L}^p(E)} \| \nabla L y \|_{L^\infty(E)}.
\]

**Remark 4** The identity in Lemma 1 (ii) and the estimates in Lemma 2 are not entirely symmetric in terms of the operator \( L \) and the adopted \( \mathcal{L}^p \)-norms. In Lemma 1 (ii) and Lemma 2 (ii) more symmetry is obtained if the standard Laplacian \( L = Ls \) is used. The estimate in Lemma 2 (i) gets symmetric in terms of the norm if the \( \mathcal{L}^\infty \)-norm is used.

**Proof of Theorem 2** (i) \( \Rightarrow \) (i')': The approximant \( x_* \in \mathcal{N}^{PUM} \) is a linear combination \( x_* = \sum_{j=1}^J \varphi^{(j)} \chi^{(j)} \) of local approximations, where \( (\varphi^{(j)})_{j=1}^J \) is a partition of
unity subordinate to the cover \( \{ V_j\}_{j=1}^J \), satisfying \( \text{supp}(\varphi^{(j)}) \subseteq V_j \), \( \varphi^{(j)} \geq 0 \), and \( \sum_{j=1}^J \varphi^{(j)} = 1 \). Therefore, the triangle inequality for the \( L^p \)-norm and the local bounds in (i) provide the estimates

\[
\| x - x_* \|_{L^p(G)} = \left\| \sum_{j=1}^J \varphi^{(j)}(x - x_*) \right\|_{L^p(G)} \leq \sum_{j=1}^J \| \varphi^{(j)}(x - x_*) \|_{L^p(G)} \leq \sum_{j=1}^J \| \varphi^{(j)} \|_{L^\infty(G_j)} \| x - x_* \|_{L^p(G)} \leq J \mathcal{E}_p^{(j)}.
\]

(i), (ii) \( \Rightarrow \) (i)', (ii)': As the functions \( \varphi^{(j)} \) satisfy \( \text{supp}(\varphi^{(j)}) \subseteq V_j \) and, in addition, the boundary condition (BC), we get

\[
\text{supp}(\nabla_L \varphi^{(j)}) \subseteq E_j \quad \text{and} \quad \text{supp}(\nabla_L (\varphi^{(j)}(x - x_*^{(j)}))) \subseteq E_j.
\]

Thus, also \( \nabla_L \varphi^{(j)} = \nabla_L (\varphi^{(j)}) \) and \( \nabla_L (\varphi^{(j)}(x - x_*^{(j)})) = \nabla_L (\varphi^{(j)}(x - x_*^{(j)})) \) on the subset \( E_j \). This fact, together with the triangle inequality, gives

\[
\| \nabla_L x - \nabla_L x_* \|_{L^p(E_j)} = \left\| \sum_{j=1}^J \nabla_L (\varphi^{(j)}(x - x_*)) \right\|_{L^p(E_j)} \leq \sum_{j=1}^J \| \nabla_L (\varphi^{(j)}(x - x_*)) \|_{L^p(E_j)}.
\]

Now, using Lemma 2 (i) together with the assumptions (i), (ii) of the theorem, we can conclude that

\[
\| \nabla_L x - \nabla_L x_* \|_{L^p(E_j)} \leq \sum_{j=1}^J \left( \| \varphi^{(j)} \|_{L^\infty(G_j)} \| \nabla_L (\varphi^{(j)}(x - x_*^{(j)})) \|_{L^p(E_j)} + \| \nabla_L \varphi^{(j)} \|_{L^\infty(E_j)} \| x - x_*^{(j)} \|_{L^p(G_j)} \right)
\]

\[
\leq \sum_{j=1}^J \left( \mathcal{E}_p^{(j)} + \| \nabla_L \varphi^{(j)} \|_{L^\infty(E_j)} \mathcal{E}_p^{(j)} \right).
\]

(i), (ii), (iii) \( \Rightarrow \) (i)', (ii)', (iii)': Since \( \text{supp}(\varphi^{(j)}) \subseteq V_j \) and the condition (BC) is satisfied, we get, similarly as before, the inclusions

\[
\text{supp}(L \varphi^{(j)}) \subseteq V_j \quad \text{and} \quad \text{supp}(L (\varphi^{(j)}(x - x_*^{(j)}))) \subseteq V_j.
\]

This implies that \( L \varphi^{(j)} = L^{(j)} \varphi^{(j)} \) and \( L (\varphi^{(j)}(x - x_*^{(j)})) = L^{(j)} (\varphi^{(j)}(x - x_*^{(j)})) \) on the subset \( V_j \), and thus

\[
\| Lx - Lx_* \|_{L^p(G)} = \left\| \sum_{j=1}^J L (\varphi^{(j)}(x - x_*)) \right\|_{L^p(G)} \leq \sum_{j=1}^J \| L^{(j)} (\varphi^{(j)}(x - x_*)) \|_{L^p(G_j)}.
\]
Fig. 2  Partition of the Minnesota graph in $J = 8$ subdomains via reduced $J$-center clustering (dark domains) and domain augmentation (red nodes, $r = 8$). The ringed nodes indicate the centers of the subdomains.

Now, the statement $(iii)'$ follows from Lemma 2 (ii) and the assumptions (i), (ii), and (iii) of the theorem. 

\[
\square
\]

6 Numerical Experiments

In this section, we perform a numerical case study in order to illustrate several properties of the PUM on graphs. In particular, we show that the construction of the partition of unity via graph partitioning and domain augmentation, as described in Algorithm 1 and Algorithm 2, is computationally efficient, and that the GBF-PUM approximation scheme in Algorithm 3 provides accurate global interpolation results. All codes are implemented in a Matlab environment, while the experiments have been carried out on a laptop with an Intel(R) Core i7 6500U CPU 2.50GHz processor and 8.00GB RAM.

The test graph $G$ consists of a road network of the state of Minnesota, referred to as Minnesota graph and has been retrieved from [29]. It consists of $n = 2642$ vertices and is characterized by 3304 edges. As a distance metric $d$ on $G$, we use the shortest-path distance. As a Laplacian, we use the normalized graph Laplacian $L_N = D^{-1/2}L_SD^{-1/2}$. An exemplary partition of the entire graph $G$ in $J = 8$ overlapping subdomains obtained by a combination of Algorithms 1 and 2 (discussed in Sect. 3) is displayed in Fig. 2.

Regarding the interpolation nodes for the tests, we recursively generate a sequence $W_N$ of sampling sets in $V$ such that $\#W_N = N$, $W_{N-1}$ is contained in $W_N$ and the new node $w_N \in W_N$ is chosen randomly from $V \setminus W_{N-1}$. As a test function, we use the bandlimited signal $x_B = \sum_{k=1}^{10} u_k$, consisting of the first 10 eigenvectors of the normalized Laplacian $L_N$ of $G$. Moreover, as a GBF-kernel for interpolation on the subgraphs, we mainly employ the variational spline kernel defined in Example 1 (ii). The chosen parameters for graph interpolation (i.e., $\gamma = 0$) are $\epsilon = 0.001$ and $s = 2$. 
Fig. 3 Local GBF-interpolation using the variational spline kernel \((\epsilon = 0.001, s = 2)\) of the test function \(x_B\) on \(J = 3\) subdomains of the Minnesota graph based on the respective sampling subsets of \(W_{200}\). The black ringed nodes indicate the sampling nodes, while the red ringed nodes denote the centers \(q_j\) of the subdomains.

Fig. 4 GBF-PUM method for the Minnesota graph. Left: Global GBF-PUM interpolant \(x_{B,\circ}\) of the test function \(x_B\) for given samples on the nodes \(W_{200}\). Right: Absolute error \(|x_B - x_{B,\circ}|\) with respect to the original signal. The ringed nodes indicate the sampling nodes. Both graphs are obtained by using the variational spline kernel with \(\epsilon = 0.001\) and \(s = 2\).

Furthermore, as a comparison at the end of this section we also provide some results for the diffusion kernel given in Example 1 (i), using the parameter \(t = 20\).

Once the cover \(\{V_j\}_{j=1}^J\) is computed, the partition of unity \(\{\varphi_j\}_{j=1}^J\) is generated by the Shepard weights given in Example 2 (ii). Using Algorithm 3 with \(J = 3\) domains and the augmentation parameter \(r = 8\), the local interpolation results for the test function \(x_B\) on the 3 subdomains are shown in Fig. 3. The corresponding composed GBF-PUM interpolant \(x_{B,\circ}\) together with an illustration of the absolute error \(|x_B - x_{B,\circ}|\) is given in Fig. 4.

**Global interpolation compared to PUM interpolation** To analyze the behavior of the GBF-PUM interpolant in Algorithm 3 compared to the global GBF method considered in [8], we compute the relative root mean square error (RRMSE) and the CPU time expressed in seconds. In Table 1, we report RRMSEs and CPU times for several sets of vertices \(W_N\). By doing so, we compare the results achieved for PUMs with smaller and larger overlappings of the subdomains in the cover. In particular, Table
Table 1  Interpolation errors and CPU times (in seconds) computed by applying GBF-PUM with $J = 8$ overlapping subdomains and using a variational spline kernel with $\epsilon = 0.001$ and $s = 2$

| $N$ | Smaller overlapping $(r = 8)$ | Larger overlapping $(r = 12)$ |
|-----|--------------------------------|--------------------------------|
|     | RRMSE  | Time   | RRMSE  | Time   |
| 132 | 7.27e−2 | 0.23   | 6.16e−2 | 0.37   |
| 264 | 2.51e−2 | 0.26   | 7.95e−3 | 0.41   |
| 528 | 6.88e−3 | 0.27   | 1.69e−3 | 0.48   |
| 1056| 8.89e−4 | 0.35   | 8.50e−5 | 0.65   |
| 2112| 5.55e−6 | 0.62   | 5.55e−6 | 1.06   |

Fig. 5  RRMSEs and CPU times (in seconds) obtained for the GBF-PUM (with $J = 8$ and $r = 8$) and the global GBF scheme in terms of $N$ interpolation nodes. Tests have been carried out by using the variational spline kernel with $\epsilon = 0.001$ and $s = 2$.

Fig. 5 highlights that, for enlarged subgraph sizes, a general improvement in terms of accuracy is obtained, but also a larger computational cost is necessary. In Fig. 5 and Fig. 6, we represent the same above-mentioned quantities for increasing values of $N$ from 200 to 2600. In these two figures, we compare the GBF-PUM interpolation results also with the global GBF method introduced in [8]. From this study, we can observe that with growing number $N$ of sampling nodes the interpolation errors of the GBF-PUM scheme improve, as predicted by Theorem 2 (since the local errors decrease). On the other hand, the CPU times show that the use of the PUM for graph signal interpolation is particularly advantageous if the number of interpolation nodes gets large.

The role of the smoothness parameter $s$. By the convergence result given in [25, Theorem 1.1] and our derivations in Corollaries 2 and 3, one would expect that, for $s \to \infty$ ($k \to \infty$ in [25]), it is possible to reconstruct the bandlimited signal $x_B$. Our numerical test showed that, for increasing $s$, the approximation error decreases, but, starting from a certain value of $s$, the ill-conditioning of the interpolation matrix heavily compromises the accuracy (see Figs. 7 and 8, for $J = 8$, $r = 8$ and $J = 8$, $r = 12$, respectively). Moreover, the same phenomenon is also visible in the global GBF scheme. In further experiments (not reported here for shortness) we observed that with increasing $s$ in the variational spline kernel the condition numbers grow quickly in both cases.
Fig. 6 RRMSEs and CPU times (in seconds) obtained for the GBF-PUM (with $J = 8$ and $r = 12$) and the global GBF scheme in terms of $N$ interpolation nodes. Tests have been carried out by using the variational spline kernel with $\epsilon = 0.001$ and $s = 2$.

Fig. 7 Analysis of the role and impact of the parameter $s$ on errors (left) and times (right) obtained for the GBF-PUM (with $J = 8$ and $r = 8$) in terms of $N$ interpolation nodes. Tests have been carried out by using the variational spline kernel with $\epsilon = 0.001$ and $s = 1, 2, 3, 4$.

Fig. 8 Analysis of the role and impact of the parameter $s$ on errors (left) and times (right) obtained for the GBF-PUM (with $J = 8$ and $r = 12$) in terms of $N$ interpolation nodes. Tests have been carried out by using the variational spline kernel with $\epsilon = 0.001$ and $s = 1, 2, 3, 4$. 
The role of the augmentation parameter $r$. These first numerical experiments indicate that the parameters $J$ (number of clusters) and $r$ (augmentation distance for the subdomains) are two main control parameters for the accuracy and the computational cost of the PUM on graphs. After testing the PUM for a range of different values $J$ and $r$, we made the following observations: (a) the experiments suggest that a good compromise between $J$ and $r$ must be found in order to guarantee both, accuracy and efficiency; (b) improved accuracy results can be obtained if $J$ increases and, simultaneously, $r$ is enlarged too; (c) increasing the parameter $r$ seems to improve accuracy more than increasing $J$. As an enlarged parameter $r$ leads also to a larger computational cost, further investigations were necessary to find an optimal balancing of the different parameters. In Fig. 9 we thus illustrate the role and impact of the augmentation parameter $r$ in the case of overlapping subdomains. More precisely, in this study we analyze the behavior of RRMSEs and CPU times associated with the GBF-PUM scheme for $N = 300, 600, 1200, 2400$ interpolation nodes. By fixing $J = 8$ and increasing $r$ from 0 to 20, we observe that the computational error decreases up to stabilizing at a certain value which depends on the total number $N$ of vertices. On the other hand, when the value of $r$ increases (and so the number of vertices within each subdomain), the entire computational cost of the GBF-PUM scheme grows accordingly.

Experimental results for the diffusion kernel Finally, in order to have a comparison between the two kernels (i) and (ii) from Example 1, in Table 2 we also report interpolation errors and execution times computed by applying GBF-PUM with $J = 8$ overlapping subdomains and using the diffusion kernel as basis functions. Similar as in the context of Corollary 3, we used a global diffusion kernel to interpolate the signal $x_B$ locally on the subdomains. As shown by Table 1, the approximation errors are in this case quite similar to those obtained with the variational spline kernel, while CPU times are roughly doubled. This last fact is essentially due to the calculation of the kernel.
Table 2  Interpolation errors and CPU times (in seconds) computed by applying GBF-PUM with $J = 8$ overlapping subdomains and using a diffusion kernel with $t = 20$

| $N$   | Smaller overlapping ($r = 8$) | Larger overlapping ($r = 12$) |
|-------|-------------------------------|-------------------------------|
|       | RRMSE | Time | RRMSE | Time |
| 132   | 3.02e−1 | 0.55 | 3.02e−1 | 0.49 |
| 264   | 1.23e−1 | 0.56 | 1.23e−1 | 0.55 |
| 528   | 3.29e−2 | 0.61 | 3.24e−2 | 0.67 |
| 1056  | 1.05e−3 | 0.82 | 9.96e−4 | 0.82 |
| 2112  | 1.44e−7 | 1.16 | 2.32e−7 | 1.28 |

7 Conclusions and Future Work

In this paper, we proposed an efficient way to generate partitions of unity on graphs via greedy-type $J$-center clustering and a domain augmentation procedure. We investigated how the PUM can be used to reconstruct global signals from local GBF approximations on subdomains, and showed that under suitable restrictions on the PUM global error estimates are inherited from local ones. Moreover, some numerical results were presented, showing the cost-efficiency and the accuracy of the method. Work in progress concerns the optimal selection of the parameters in the PUM, namely the number $J$ of the subdomains, the augmentation factor $r$ for the subdomains, more sophisticated Shepard weights for the partition of unity, and the shape parameters associated with the kernels (in analogy to the RBF case [5]). The first experiments presented here showed that a proper choice of these parameters highly influences accuracy and efficiency of the computational procedure. A further development step consists in providing more adaptive techniques for the selection of the partitions on the graph in order to tailor the PUM best possibly to the underlying graph topology.

Acknowledgements  The authors sincerely thank the editors and the anonymous referees for insightful comments and suggestions, which gave the chance to significantly improve the quality of the paper. The first two authors acknowledge support from the Department of Mathematics “Giuseppe Peano” of the University of Torino via projects 2020 “Models and numerical methods in approximation, in applied sciences and in life sciences” and “Mathematical methods in computational sciences”. This research has been accomplished within the RITA “Research Italian network on Approximation” and the UMI Group TAA “Approximation Theory and Applications”. The work was partially supported by INdAM-GNCS, and all the authors are members of the INdAM Research group GNCS.

References

1. Aronszajn, N.: Theory of reproducing kernels. Trans. Am. Math. Soc. 68, 337–404 (1950)
2. Cavoretto, R., De Rossi, A.: Adaptive meshless refinement schemes for RBF-PUM collocation. Appl. Math. Lett. 90, 131–138 (2019)
3. Cavoretto, R., De Rossi, A.: Error indicators and refinement strategies for solving Poisson problems through a RBF partition of unity collocation scheme. Appl. Math. Comput. 369, 124824 (2020)
4. Cavoretto, R., De Rossi, A., Perracchione, E.: Optimal selection of local approximants in RBF-PU interpolation. J. Sci. Comput. 74, 1–27 (2018)
5. Cavoretto, R., De Rossi, A., Mukhametzhanov, M.S., Sergeyev, Y.D.: On the search of the shape parameter in radial basis functions using univariate global optimization methods. J. Global Optim. 79, 305–327 (2021)
6. Cheng, C., Jiang, Y., Sun, Q.: Spatially distributed sampling and reconstruction. Appl. Comput. Harm. Anal. 47(1), 109–148 (2019)
7. Chung, F.R.K.: Spectral Graph Theory. American Mathematical Society, Providence (1997)
8. Erb, W.: Graph Signal Interpolation with Positive Definite Graph Basis Functions. arXiv:1912.02069 (2019)
9. Erb, W.: Semi-Supervised Learning on Graphs with Feature-Augmented Graph Basis Functions. arXiv:2003.07646 (2020)
10. Erb, W.: Shapes of uncertainty in spectral graph theory. IEEE Trans. Inf. Theory 67(2), 1291–1307 (2021)
11. Fang, Q., Shin, C.E., Sun Q.: Polynomial control on weighted stability bounds and inversion norms of localized matrices on simple graph. To appear in J. Fourier Anal. Appl. (2021), https://doi.org/10.1007/s00041-021-09864-9.
12. Fasshauer, G.E.: Meshfree Approximation Methods with Matlab. World Scientific, Singapore (2007)
13. Godsil, C., Royle, G.: Algebraic Graph Theory. Springer, New York (2001)
14. Gonzalez, T.F.: Clustering to minimize the maximum intercluster distance. Theor. Comput. Sci. 38, 293–306 (1985)
15. Griebel, M., Schweitzer, M.A.: A particle-partition of unity method for the solution of elliptic, parabolic and hyperbolic PDE. SIAM J. Sci. Comp. 22(3), 853–890 (2000)
16. Har-Peled, S.: Geometric Approximation Algorithms. American Mathematical Society, Boston (2011)
17. Hochbaum, D.S., Shmoys, D.B.: A best possible heuristic for the k-center problem. Math. Oper. Res. 10(2), 180–184 (1985)
18. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1985)
19. Kondor, R.I., Lafferty, J.: Diffusion kernels on graphs and other discrete input spaces. in Proc. of the 19th. Intern. Conf. on Machine Learning ICML02, pp. 315–322 (2002)
20. Larsson, E., Shcherbakov, V., Heryudo, A.: A least squares radial basis function partition of unity method for solving PDEs. SIAM J. Sci. Comput. 39(6), A2538–A2563 (2017)
21. Melenk, J.M., Babuška, I.: The partition of unity finite element method: Basic theory and applications. Comput. Methods Appl. Mech. Eng. 139(1–4), 289–314 (1996)
22. Ortega, A., Frossard, P., Kovačević, J., Moura, J.M.F., Vandergheynst, P.: Graph signal processing: overview, challenges, and applications. Proce. IEEE 106(5), 808–828 (2018)
23. Pesenson, I.Z.: Sampling by averages and average splines on Dirichlet spaces and on combinatorial graphs. To appear in: Hirn M., et al. editors: Excursions in Harmonic Analysis, Volume 6, book series Applied and Numerical Harmonic Analysis, Birkhäuser-Springer (2021)
24. Pesenson, I.Z.: Sampling in Paley-Wiener spaces on combinatorial graphs. Trans. Am. Math. Soc. 360(10), 5603–5627 (2008)
25. Pesenson, I.Z.: Variational splines and Paley–Wiener spaces on combinatorial graphs. Constr. Approx. 29(1), 1–21 (2009)
26. Pesenson, I.Z., Pesenson, M.Z.: Graph signal sampling and interpolation based on clusters and averages. J. Fourier Anal. Appl. 27, 39 (2021). https://doi.org/10.1007/s00041-021-09828-z
27. Rifkin, R., Yeo, G., Poggio, T.: Regularized least-squares classification. Nato Science Series Sub Series III. Comput. Syst. Sci. 190, 131–154 (2003)
28. Romero, D., Ma, M., Giannakis, G.B.: Kernel-based reconstruction of graph signals. IEEE Trans. Signal Process. 65(3), 764–778 (2017)
29. Rossi, R.A., Ahmed, N.K.: The Network Data Repository with Interactive Graph Analytics and Visualization. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (2015), http://networkrepository.com
30. Schaback, R., Wendland, H.: Approximation by Positive Definite Kernels. In Advanced Problems in Constructive Approximation, Birkhäuser Verlag, Basel (2003), 203–222
31. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press, Cambridge (2002)
32. Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data. in Proceedings of the 1968 23rd ACM National Conference, ACM ’68, New York, NY, USA (1968), 517–524
33. Shuman, D.I., Ricaud, B., Vandergheynst, P.: Vertex-frequency analysis on graphs. Appl. Comput. Harm. Anal. 40(2), 260–291 (2016)
34. Smola, A., Kondor. R.: Kernels and regularization on graphs. In Learning Theory and Kernel Machines. Springer, Berlin Heidelberg, pp. 144–158 (2003)
35. Stanković, L., Đaković, L., Sejdić, E.: Introduction to Graph Signal Processing, pp. 3–108. Springer, In Vertex-Frequency Analysis of Graph Signals (2019)
36. von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. 17, 395–416 (2007)
37. Wahba, G.: Spline Models for Observational Data. CBMS-NSF Regional Conference Series in Applied Mathematics, volume 59, SIAM, Philadelphia (1990)
38. Ward, J.P., Narcowich, F.J., Ward, J.D.: Interpolating splines on graphs for data science applications. Appl. Comput. Harm. Anal. 49(2), 540–557 (2020)
39. Wendland, H.: Fast evaluation of radial basis functions: Methods based on partition of unity. In: C.K. Chui et al. (Eds.), Approximation Theory X: Wavelets, Splines, and Applications (2002), 473–483
40. Wendland, H.: Scattered Data Approximation. Cambridge University Press, Cambridge (2005)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.