Spherical interpolation using the partition of unity method: an efficient and flexible algorithm

Spherical interpolation using the partition of unity method: an efficient and flexible algorithm

Roberto Cavoretto*, Alessandra De Rossi

Department of Mathematics "G. Peano", University of Torino, via Carlo Alberto 10, I–10123 Torino, Italy

Abstract

An efficient and flexible algorithm for the spherical interpolation of large scattered data sets is proposed. It is based on a partition of unity method on the sphere and uses spherical radial basis functions as local approximants. This technique exploits a suitable partition of the sphere in a number of spherical zones, the construction of a certain number of cells such that the sphere is contained in the union of the cells, with some mild overlap among the cells, and finally the employment of an optimized spherical zone searching procedure. Some numerical experiments show the good accuracy of the spherical partition of unity method and the high efficiency of the algorithm.

Key words: partition of unity method, spherical radial basis functions, fast interpolation algorithms, scattered data interpolation.
2010 MSC: 65D05, 65D15, 65D17.

1. Introduction

In [4] we presented a fast and accurate algorithm for the spherical interpolation of large scattered data sets. It is based on a modified spherical Shepard's method and involves spherical radial basis functions (SRBFs) as local approximants (see [5]), but the main novelty of the paper is the employment of a partition of the unit sphere in a suitable number of parallel spherical zones and the construction of local neighbourhoods. The combination of these two processes gives rise to an optimized spherical zone searching procedure used in the nearest neighbour node searching, thus producing a good accuracy and a high efficiency (see also [3]).

The aim of this paper is to generalize the interpolation algorithm, extending the idea proposed in [4] to the more general *Partition of Unity Method* (PUM) on the sphere. The partition of unity method was suggested in [1, 13] in the mid 1990s in the context of meshfree Galerkin methods for the solution of partial differential equations (PDEs), but now it is also an effective method for fast computation in the field of approximation theory (see, e.g, [7, 15, 16]). This approach is here combined with spherical radial basis functions. The final result is an efficient and flexible algorithm for scattered data interpolation on the sphere.

The paper is organized as follows. Section 2 is devoted to a general description of the partition of unity method on the sphere. In Section 3 we present in detail the spherical interpolation algorithm which is based on a partition of unity. In Section 4 we show some numerical results concerning accuracy and effectiveness of the spherical PUM algorithm. Finally, Section 5 contains conclusions and future work.

2. Partition of unity method on the sphere

The basic idea of the spherical partition of unity method is to consider a partition of the unit sphere $\Omega = \mathbb{S}^2 \subset \mathbb{R}^3$ into d cells Ω_j such that $\Omega = \bigcup_{j=1}^{d} \Omega_j$ with some mild overlap among the cells. At first, we choose a partition of

*Corresponding author.
Email addresses: roberto.cavoretto@unito.it (Roberto Cavoretto), alessandra.derossi@unito.it (Alessandra De Rossi)

unity, i.e. a family of compactly supported, non-negative, continuous functions W_j with $\mathrm{supp}(W_j) \subseteq \Omega_j$ such that $\sum_{j=1}^{d} W_j(x) = 1$, for all $x \in \Omega$. Then, for each cell Ω_j we consider a spherical radial basis function Z_j as local approximant and form the global approximant given by

$$\mathcal{I}(x) = \sum_{j=1}^{d} Z_j(x) W_j(x), \qquad x \in \Omega. \tag{1}$$

Note that if the local approximants satisfy the interpolation conditions at data point x_i, i.e. $Z_j(x_i) = f(x_i)$, then the global approximant also interpolates at this node, i.e. $\mathcal{I}(x_i) = f(x_i)$, for $i = 1, 2, \ldots, n$.

In accordance with the statements in [15] we require some additional assumptions on the *covering* $\{\Omega_j\}_{j=1}^{d}$.

Definition 2.1. *Suppose that $\Omega = \mathbb{S}^2 = \bigcup_{j=1}^{d} \Omega_j \subset \mathbb{R}^3$ and $\mathcal{X}_n = \{x_i, i = 1, 2, \ldots, n\} \subseteq \mathbb{S}^2$ are given. An open covering $\{\Omega_j\}_{j=1}^{d}$ is called regular for $(\Omega, \mathcal{X}_n)$ if the following properties are satisfied:*

(a) for each $x \in \Omega$, the number of cells Ω_j with $x \in \Omega_j$ is bounded by a global constant K;

(b) the local fill distances $h_{\mathcal{X}_j,\Omega_j}$ are uniformly bounded by the global fill distance $h_{\mathcal{X}_n,\Omega}$, where $\mathcal{X}_j = \mathcal{X}_n \cap \Omega_j$.

Moreover, we consider the following theorem, which yields the polynomial precision and controls the growth of error estimates. For this reason, we denote by $\pi_s(\mathbb{S}^2)$ the set of spherical polynomials of degree s (see, e.g., [10, 16]).

Theorem 2.1. *Suppose that $\Omega = \mathbb{S}^2 = \bigcup_{j=1}^{d} \Omega_j \subset \mathbb{R}^3$ and Ω_j is a spherical cap (cell) with centre at $x \in \mathbb{S}^2$ and radius r, where $0 < r \leq \pi/2$. Let $s \in \mathbb{N}$ be fixed and there exists a constant $h_{r,s} = \tan(r/4)/\{4[1 + 2/(\sqrt{3}\cos(r/2))]s^2\}$, such that $h_{\mathcal{X}_j,\Omega_j} \leq h_{r,s}$. Then, for all $\mathcal{X}_j = \{x_i, i = 1, 2, \ldots, n_j\} \subseteq \Omega_j$, $\mathcal{X}_j$ is a $\pi_s(\mathbb{S}^2)$-unisolvent set and there exist functions $u_k : \Omega_j \to \mathbb{R}$, $k = 1, 2, \ldots, n_j$, $j = 1, 2, \ldots, d$, such that*

(1) $\sum_{k=1}^{n_j} u_k(x) p(x_k) = p(x)$, for all $p \in \pi_s(\mathbb{S}^2)$ and all $x \in \Omega_j$;

(2) $\sum_{k=1}^{n_j} |u_k(x)| \leq 2$, for all $x \in \Omega_j$.

Although the theory of SRBFs is here mentioned, for brevity we do not report basic definitions and theorems, referring to the papers [2, 11] for a more detailed analysis. However, in order to formulate the following theorem, we say that a strictly conditionally positive definite function ψ of order m on $\mathbb{S}^2$ has *α-Fourier decay* if there exist positive constants c_1, c_2, such that

$$c_1(1+k)^{-(2+\alpha)} \leq a_k \leq c_2(1+k)^{-(2+\alpha)}, \qquad \alpha > 0,\ k \geq m,$$

with a_k spherical Fourier coefficients which are related, via the additional theorem, to Legendre coefficients of ψ (see [11]). Moreover, we define the *normed* native Hilbert space of ψ by

$$H_\psi = \{f \in L_2(\mathbb{S}^2) : \|f\|_\psi < \infty\},$$

where $\|\cdot\|_\psi$ is the norm induced by the inner product.

Therefore, we consider the following convergence result (see, e.g., [9, 10, 11, 16] and references therein).

Theorem 2.2. *Suppose that $\Omega = \mathbb{S}^2 = \bigcup_{j=1}^{d} \Omega_j$ and the assumptions of Theorem 2.1 hold. Let $\mathcal{X}_n = \{x_i, i = 1, 2, \ldots, n\} \subseteq \Omega$. Let ψ be strictly conditionally positive definite function of order m on $\mathbb{S}^2$. Let $\{\Omega_j\}_{j=1}^{d}$ be a regular covering for $(\Omega, \mathcal{X}_n)$ and let $\{W_j\}_{j=1}^{d}$ be a family of compactly supported, non-negative, continuous functions for $\{\Omega_j\}_{j=1}^{d}$. Then the error beetween $f \in H_\psi(\Omega)$ satisfying the α-Fourier decay and its partition of unity interpolant (1) can be bounded by*

$$|f(x) - \mathcal{I}(x)| \leq C h_{\mathcal{X}_n,\Omega}^{\alpha/4} \|f - \mathcal{I}\|_{H_\psi},$$

where C is a positive constant independent of $h_{\mathcal{X}_n,\Omega}$.

Note that the partition of unity preserves the local approximation order for the global fit. Hence, we can efficiently compute large SRBF interpolants by solving small SRBF interpolation problems (in parallel as well) and then combine them together with the global partition of unity $\{W_j\}_{j=1}^{d}$. This approach enables to decompose a large problem into many small problems, and at the same time ensures that the accuracy obtained for the local fits is carried over to the global fit. In particular, the spherical PUM can be thought as a spherical Shepard's method with higher-order data, since local approximations Z_j instead of data values f_j are used (see [12]).

3. Spherical zone algorithm based on partition of unity

In this section we propose a new fast algorithm for spherical interpolation of large scattered data sets lying on $\mathbb{S}^2 \subset \mathbb{R}^3$. This technique is based on the partition of the unit sphere in a suitable number of spherical zones, the construction of a number of d cells Ω_j such that the domain $\Omega = \mathbb{S}^2 = \bigcup_{j=1}^{d} \Omega_j$ with some mild overlap among the subdomains (cells), and then the employment of an optimized spherical zone searching procedure. Finally, the partition of unity method for spherical interpolation is combined with spherical radial basis functions.

This process can be briefly described as follows: (i) partition the sphere into a suitable number of parallel spherical zones; (ii) consider a *spherical zone searching procedure* establishing the minimal number of zones to be examined, in order to localize the set of nodes for each cell; (iii) apply the spherical PUM method which uses spherical radial basis functions as nodal functions. These three steps correspond to data distribution, localization and evaluation phases, respectively. We remark that only one spherical zone structure is used for both localization and evaluation phases.

Let us now consider in detail the algorithm for scattered data interpolation on the sphere.

INPUT: n, number of data; $\mathcal{X}_n = \{(x_i, y_i, z_i), i = 1, 2, \ldots, n\}$, set of data points; $\mathcal{F}_n = \{f_i, i = 1, 2, \ldots, n\}$, set of data values; d, number of cells; $C_d = \{(\bar{x}_i, \bar{y}_i, \bar{z}_i), i = 1, 2, \ldots, d\}$, set of cell points (centres of PU cells); s, number of evaluation points; $\mathcal{E}_s = \{(x_i, y_i, z_i), i = 1, 2, \ldots, s\}$, set of evaluation points.

OUTPUT: $\mathcal{A}_s = \{\mathcal{I}_i \equiv \mathcal{I}(x_i, y_i, z_i), i = 1, 2, \ldots, s\}$, set of approximated values.

`Stage 1.` The set $\mathcal{X}_n$ of nodes and the set $\mathcal{E}_s$ of evaluation points are ordered with respect to a common direction (e.g. the z-axis), by applying a *quicksort$_z$ procedure*.

`Stage 2.` For each cell point $\bar{x}_i$, $i = 1, 2, \ldots, d$, a local circular cell (a spherical cap) is constructed, whose half-size (the spherical radius) depends on the cell number d, that is

$$\delta_{cell} = \frac{3.809}{\sqrt{d}}. \tag{2}$$

This value is suitably chosen by considering the result in [14].

`Stage 3.` After the number of spherical zones to be considered is found taking

$$q = \left\lceil \frac{\pi}{\delta_{cell}} \right\rceil,$$

the spherical zones are numbered from 1 to q. Such a choice follows directly from the length, that here holds π, of the (shorter) great circle arc joining north pole and south pole, and the cell spherical radius δ_{cell}.

`Stage 4.` A suitable family of q spherical zones of equal width $\delta_{zone} \equiv \delta_{cell}$ (with possible exception of one of them) on the sphere and parallel to the xy-plane is constructed. Then, the following two structures are considered:

- the set $\mathcal{X}_n$ of nodes is partitioned by the spherical zone structure into q subsets $\mathcal{X}_{n_k}$, whose elements are $(x_{k1}, y_{k1}, z_{k1}), (x_{k2}, y_{k2}, z_{k2}), \ldots, (x_{kn_k}, y_{kn_k}, z_{kn_k})$, $k = 1, 2, \ldots, q$. These are counted and the number of nodes in the k-th spherical zone is stored in n_k;
- the set C_d of cell points is partitioned by the spherical zone structure into q subsets C_{d_k}, whose elements are $(\bar{x}_{k1}, \bar{y}_{k1}, \bar{z}_{k1}), (\bar{x}_{k2}, \bar{y}_{k2}, \bar{z}_{k2}), \ldots, (\bar{x}_{kd_k}, \bar{y}_{kd_k}, \bar{z}_{kd_k})$, $k = 1, 2, \ldots, q$. These are counted and the number of cell points in the k-th spherical zone is stored in d_k;
- the set $\mathcal{E}_s$ is partitioned into q subsets $\mathcal{E}_{p_k}$, $k = 1, 2, \ldots, q$, so that the evaluation points of $\mathcal{E}_{p_k}$ belong to the k-th zone. These are counted and the number of evaluation points are stored in p_k.

This stage can be summarized as follows:

Set $count_1, count_2, count_3 = 0$;
  $z_s = -1$;
for $(k = 1, 2, \ldots, q)$ **do**
  Set $n_k, d_k, p_k = 0$;

$i = count_1 + 1$;
$j = count_2 + 1$;
$l = count_3 + 1$;
$u_k = k \cdot \delta_{zone}$;
$v_k = z_s + (1 - \cos(u_k))$;
while $(z_i \leq v_k \wedge i \leq n)$ **do**
Set $n_k = n_k + 1$;
$count_1 = count_1 + 1$;
$i = i + 1$;
end while
Set $bsz_k = count_1 - n_k + 1$;
$esz_k = count_1$;
while $(\bar{z}_j \leq v_k \wedge j \leq d)$ **do**
Set $d_k = d_k + 1$;
$count_2 = count_2 + 1$;
$j = j + 1$;
end while
Set $cell\_bsz_k = count_2 - d_k + 1$;
$cell\_esz_k = count_2$;
while $(z_i \leq v_k \wedge i \leq s)$ **do**
Set $p_k = p_k + 1$;
$count_3 = count_3 + 1$;
$l = l + 1$;
end while
Set $eval\_bsz_k = count_3 - p_k + 1$;
$eval\_esz_k = count_3$;
return $(n_k; \mathcal{X}_{n_k}) \wedge (d_k; \mathcal{C}_{d_k}) \wedge (p_k; \mathcal{E}_{p_k})$;
end for

`Stage 5.` In order to identify the zones to be examined in the searching procedure, we adopt the following rule which is composed of three steps: (i) the width δ_{zone} of a spherical zone is chosen equal to the cell spherical radius δ_{cell}, i.e. $\delta_{zone} \equiv \delta_{cell}$, and the ratio between these quantities is denoted by $i^* = \delta_{cell}/\delta_{zone}$; (ii) the value i^* provides the number j^* of spherical zones to be examined for each node by the rule $j^* = 2i^* + 1$, which obviously here gives $j^* = 3$. In practice, this means that the search of the nearby nodes (or cell points) is limited at most to three spherical zones: the zone on which the considered node lies, the previous and the next zones; (iii) for each zone k, $k = 1, 2, \ldots, q$, a spherical zone searching procedure is considered, examining the nodes from zone $k - i^*$ to zone $k + i^*$. Note that for the nodes of the first and last spherical zones, in general, we need to reduce the total number of spherical zones to be examined, because if $k - i^* < 1$ or $k + i^* > q$ it will assign $k - i^* = 1$ and $k + i^* = q$, respectively.

After defining which and how many spherical zones are to be examined, a spherical zone searching procedure is applied:

- for each cell point of $\mathcal{C}_{d_k}$, $k = 1, 2, \ldots, q$, to determine all nodes belonging to a cell. The number of nodes of the cell centred at $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$ is counted and stored in m_i, $i = 1, 2, \ldots, d$;
- for each evaluation point of $\mathcal{E}_{p_k}$, $k = 1, 2, \ldots, q$, in order to find all those belonging to a cell of centre $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$ and geodesic radius δ_{cell}. The number of cells containing the i-th evaluation point is counted and stored in r_i, $i = 1, 2, \ldots, s$.

This stage can be summarized as follows:

for $(k = 1, 2, \ldots, q)$ **do**
Set $first = k - i^*$;
$last = k + i^*$;
if $(first < 1)$ **then**

```
      Set first = 1;
   end if
   if (last > q) then
      Set last = q;
   end if
   for (h = cell_bsz_k, ..., cell_esz_k) do
      Set m_h = 0;
      for (i = first, ..., last) do
         for (j = bsz_i, ..., esz_i) do
            if ((x_j, y_j, z_j) ∈ I_h((x̄, ȳ, z̄); δ_cell)) then
               Set m_h = m_h + 1;
                  STORE_{h,m_h}(x_j, y_j, z_j, f_j);
            end if
         end for
      end for
      return (x, y, z) ∈ I_h((x̄, ȳ, z̄); δ_cell);
   end for
   for (h = eval_bsz_k, ..., eval_esz_k) do
      Set r_h = 0;
      for (i = first, ..., last) do
         for (j = cell_bsz_i, ..., cell_esz_i) do
            if ((x_j, y_j, z_j) ∈ I_h((x̄, ȳ, z̄); δ_cell)) then
               Set r_h = r_h + 1;
                  STORE_{h,r_h}(x_j, y_j, z_j);
            end if
         end for
      end for
      return (x, y, z) ∈ I_h((x̄, ȳ, z̄); δ_cell);
   end for
end for
```

`Stage 6.` A local interpolant Z_j, $j = 1, 2, \ldots, d$, is found for each cell point.
`Stage 7.` A local approximant $Z_j(x, y, z)$ and a weight function $W_j(x, y, z)$, $j = 1, 2, \ldots, d$, is found for each evaluation point.

`Stage 8.` Applying the global fit (1), the surface can be approximated at any evaluation point $(x, y, z) \in \mathcal{E}_s$.

Computational complexity and storage requirements. The spherical PUM algorithm involves the use of the standard sorting routine quicksort, which requires on average a time complexity $O(n \log n)$ and $O(s \log s)$ in `Stage 1`. In order to compute the local SRBF interpolants, we solve d linear systems of small dimensions, thus requiring a computational cost of order $O(m_i^3)$, $i = 1, 2, \ldots, d$, for each cell, where m_i is the number of nodes in the i-th cell (see `Stage 6`). Moreover, in `Stage 5`, `7` and `8` we also need of a cost of $r_k \cdot O(m_i)$, $i = 1, 2, \ldots, d$, $k = 1, 2, \ldots, s$, for the k-th evaluation point of $\mathcal{E}_s$. Finally, the algorithm requires $4n$, $4d$ and $4s$ storage requirements for the data, and m_i, $i = 1, 2, \ldots, d$, locations for the coefficients of each local SRBF interpolant.

4. Numerical experiments

In this section we present some tests to verify performance and effectiveness of the spherical PUM algorithm on scattered data sets. The code is implemented in C/C++ language, while numerical results are carried out on a Pentium IV computer (2.1 GHz). In the experiments we consider two different node distributions containing $n = (2^k + 1)^2$, $k = 7, 8, 9$, nodes: (a) near-uniform distribution generated by the spiral method of Saff and Kuijlaars (SK) [14] and (b) uniformly random distribution obtained by the MATLAB (M) statements given in [8].

The spherical PUM method/algorithm is running by considering d cell points and $s = 600$ evaluation points, which are generated by the SK's method on $\mathbb{S}^2$. Here, for the global interpolant (1) we use the spherical Shepard's weight.

Moreover, in Table 1 we compare the spherical interpolation algorithm implemented by combining the PUM with the spherical zone structure and the algorithm where the sphere is not partitioned in spherical zones. It emphasizes that the use of zonal structure gives a considerable saving of time. Finally, in Tables 2 - 3 we report the Root Mean Square Errors (RMSEs), which are computed on the following two test functions

$$f_1(x, y, z) = \frac{9x^3 - 2x^2y + 3xy^2 - 4y^3 + 2z^3 - xyz}{10}, \qquad f_2(x, y, z) = \sin x \sin y \sin z,$$

using the spherical inverse multiquadric ψ [6] as local SRBF approximant

$$\psi(t) = \left(1 + \gamma^2 - 2\gamma c\right)^{-1/2}, \qquad \gamma \in (0, 1),$$

where $c = \cos(t)$ and t measures geodesic distance on the unit sphere. Observing the errors in Tables 2 - 3, we note that the method reaches a good level of accuracy.

n	d	zone sph. PUM	no-zone sph. PUM
16641	4096	1.188	13.454
66049	16384	7.953	205.062
263169	65536	60.735	3260.375

Table 1: CPU times (in seconds) obtained by running spherical PUM algorithms.

n	16641	66049	263169
SK	9.7840E − 7	6.3241E − 8	4.2746E − 8
M	1.6272E − 5	5.1237E − 6	9.2968E − 7

Table 2: RMSEs obtained by using ψ with $\gamma = 0.5$ for f_1.

n	16641	66049	263169
SK	4.0711E − 7	2.3664E − 8	1.4662E − 8
M	4.7026E − 6	2.3114E − 6	6.8838E − 7

Table 3: RMSEs obtained by using ψ with $\gamma = 0.5$ for f_2.

5. Conclusions

In this work we present a new local algorithm for scattered data interpolation on the sphere, which works well and fastly also when the amount of data is very large. In particular, this optimized implementation of the PUM on the sphere is obtained by applying a spherical zone nearest neighbour searching procedure. Moreover, the proposed algorithm is flexible since different choices of local approximants are allowable, is easily parallelizable, and completely automatic. However, as research and future work we expect to refine the spherical zone algorithm based on PUM adopting suitable data structures like kd-trees and range trees.

References

[1] I. Babuška, J. M. Melenk, The partition of unity method, Internat. J. Numer. Methods. Engrg. 40 (1997) 727–758.

[2] B.J.C. Baxter, S. Hubbert, Radial basis function on the sphere, in: Recent Progress in Multivariate Approximation, Internat. Ser. Numer. Math., vol. 137, Birkhäuser, Basel, Switzerland, 2001, pp. 33–47.

[3] R. Cavoretto, Meshfree Approximation Methods, Algorithms and Applications, PhD Thesis, University of Turin, 2010.

[4] R. Cavoretto, A. De Rossi, Fast and accurate interpolation of large scattered data sets on the sphere, J. Comput. Appl. Math. 234 (2010) 1505–1521.

[5] A. De Rossi, Spherical interpolation of large scattered data sets using zonal basis functions, in: M. Dæhlen, K. Mørken, L.L. Schumaker (Eds.), Mathematical Methods for Curves and Surfaces, Nashboro Press, Brentwood, TN, 2005, pp. 125–134.

[6] G.E. Fasshauer, L.L. Schumaker, Scattered data fitting on the sphere, in: M. Dæhlen, T. Lyche, L.L. Schumaker (Eds.), Mathematical Methods for Curves and Surfaces, Vanderbilt Univ. Press, Nashville, TN, 1998, pp. 117–166.

[7] G.E. Fasshauer, Meshfree Approximation Methods with MATLAB, World Scientific Publishers, Singapore, 2007.

[8] B. Fornberg, C. Piret, A stable algorithm for flat radial basis functions on a sphere, SIAM J. Sci. Comput. 30 (2007/08), 60–80.

[9] W. Freeden, T. Gervens, M. Schreiner, Constructive Approximation on the Sphere with Applications to Geomathematics, Clarendon Press, Oxford, 1998.

[10] K. Hesse, Q.T. Le Gia, Local radial basis function approximation on the sphere, Bull. Aust. Math. Soc. 77 (2008) 197–224.

[11] S. Hubbert, T. Morton, L_p-error estimates for radial basis function interpolation on the sphere, J. Approx. Theory 129 (2004) 58–77.

[12] R. Franke, Locally determined smooth interpolation at irregularly spaced points in several variables, J. Inst. Math. Appl. 19 (1977) 471–482.

[13] J.M. Melenk, I. Babuška, The partition of unity finite element method: basic theory and applications, Comput. Methods. Appl. Mech. Engrg. 139 (1996) 289–314.

[14] E. Saff, A.B.J. Kuijlaars, Distributing many points on a sphere, Math. Intelligencer 19 (1997) 5–11.

[15] H. Wendland, Fast evaluation of radial basis functions: Methods based on partition of unity, in: C.K. Chui, L.L. Schumaker, J. Stöckler (Eds.), Approximation Theory X: Wavelets, Splines, and Applications, Vanderbilt Univ. Press, Nashville, TN, 2002, pp. 473–483.

[16] H. Wendland, Scattered Data Approximation, Cambridge Monogr. Appl. Comput. Math., vol. 17, Cambridge Univ. Press, Cambridge, 2005.