ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

Efficient Speed-up of Radial Basis Functions Approximation and Interpolation Formula Evaluation*

Michal Smolik[1] [0000−0001−6409−1691] and Vaclav Skala[1] [0000−0001−8886−4281]

Faculty of Applied Sciences, University of West Bohemia,
Plzen, Czech Republic
{smolik,skala}@kiv.zcu.cz

Abstract. This paper presents a method for efficient Radial basis function (RBF) evaluation if compactly supported radial basis functions (CSRBF) are used. Application of CSRBF leads to sparse matrices, due to limited influence of radial basis functions in the data domain and thus non-zero weights (coefficients) are valid only for some areas in the data domain. The presented algorithm uses space subdivision which enables us to use only relevant weights for efficient RBF function evaluation. This approach is applicable for $2D$ and $3D$ case and leads to a significant speed-up. This approach is applicable in cases when the RBF function is evaluated repeatably.

Keywords: Radial basis functions; space subdivision; function evaluation; interpolation; approximation.

1 Introduction

Interpolation and approximation are well-known techniques in many scientific disciplines, i.e. mostly in physical sciences [7]. There are two main approaches dealing with an interpolation, as well as an approximation. The first one is the mesh-based approach and the second one is the mesh-less approach.

The mesh-based approaches need to know the connectivity of the interpolated or approximated dataset. However, the triangulation of the input dataset can be highly time consuming in higher dimensions, as the computational time complexity of Delaunay triangulation [20] is $O(N^{\lceil d/2 \rceil + 1})$, i.e. for $d = 2$ is $O(N^2)$ and for $d = 3$ is $O(N^3)$ [23]. Of course, there are some algorithms [5], [6], [15] that are dealing with decreasing of the time complexity of Delaunay triangulation. However, the computational time to construct the triangulation is still high.

On the other hand, the mesh-less techniques do not require any tessellation, which is a significant advantage over the mesh-based techniques, i.e. one can

* The research was supported by projects Czech Science Foundation (GACR) No. GA17-05534S, by Ministry of Education, Youth, and Sport of Czech Republic - University spec. research - 1311, and partially by SGS 2019-016.

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

directly start to compute the mesh-less approximation or interpolation from the scattered or unsorted input dataset. There are many algorithms for mesh-less approximation of scattered data. Many of them are described in the book [8] or [10]. Other examples of mesh-less approximation are in [1], [2], [3].

The mostly used mesh-less technique for interpolation and approximation is the Radial basis function interpolation and approximation. It uses only the distance between the pairs of centers of radial basis functions and input points. An advantage of the Radial basis function interpolation and approximation is time complexity which is independent of the dimension, which is useful in higher dimensions.

There are many research papers focused on speeding-up the interpolation or approximation process and solving the stability of computation. The paper [12] uses the preconditioned Krylov iteration to compute fast interpolation. The paper [28] uses global radial basis functions for interpolation but use them in a local sense to speed-up the interpolation. The paper [4] provides a summary of fast RBF interpolation and approximation. These methods speed-up the approximation or interpolation, i.e. finding weights of RBFs, whereas we would like to speed-up the evaluation of the final Radial basis function formula. In this paper, we focus on the efficient evaluation of the RBFs, i.e. computation of interpolation or approximation, when coefficients (weights) of the RBF have already been computed. The approach is based on the space subdivision application.

2 Radial Basis Functions

The Radial Basis Function (RBF) interpolation [18] and approximation [8] is a meshless technique which was introduced by Hardy [13]. It is commonly used in many scientific disciplines such as solution of partial differential equations [14], [32], image reconstruction [29], neural networks [31], vector field [27], [24], [26], GIS systems [16], optics [19] etc.

The formula for computing the function value of RBF is the weighted sum of radial basis functions and has the following form

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \lambda_i \varphi\left(\|\boldsymbol{x} - \boldsymbol{\xi}_i\|\right), \qquad (1)$$

where $\boldsymbol{x}$ is the position for which (1) provides the interpolating value, $\varphi(\ldots)$ is the radial basis function, λ_i is the weight of radial basis function, N is the number of radial basis functions and $\boldsymbol{\xi}_i$ is the center of radial basis function.

The radial basis functions used in (1) can be selected from two main groups. The first ones are "global" radial basis functions. These functions influence the

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

whole interval of the data interpolation/approximation domain. The most known examples of "global" radial basis functions are shown below.

$$\begin{aligned}
&\text{Thin Plate Spline} && \varphi(r) = r^2 \log r = \frac{1}{2} r^2 \log r^2 \\
&\text{Gauss function} && \varphi(r) = e^{-(\epsilon r)^2} \\
&\text{Inverse Quadric} && \varphi(r) = \frac{1}{1+(\epsilon r)^2} \\
&\text{Inverse Multiquadric} && \varphi(r) = \frac{1}{\sqrt{1+(\epsilon r)^2}} \\
&\text{Multiquadric} && \varphi(r) = \sqrt{1+(\epsilon r)^2}
\end{aligned} \tag{2}$$

where ϵ is the shape parameter of the radial basis function.

Local RBFs (CSRBF) were introduced by [30]. These radial basis functions have only local influence, i.e. the function value is non-zero only on some limited interval. This is a great advantage when solving the linear system of equations, as the interpolation or approximation matrix is sparse. The CSRBF has the following form

$$\varphi(r) = (1-r)_+^q P(r), \tag{3}$$

where $P(r)$ is some polynomial, q is some non-zero positive number and

$$(1-r)_+ = \begin{cases} (1-r) & (1-r) \geq 0 \\ 0 & (1-r) < 0 \end{cases} \tag{4}$$

The well known examples of CSRBF are given by.

$$\begin{aligned}
\varphi_1(r) &= (1-\hat{r})_+ & \varphi_6(r) &= (1-\hat{r})_+^6(35\hat{r}^2+18\hat{r}+3) \\
\varphi_2(r) &= (1-\hat{r})_+^3(3\hat{r}+1) & \varphi_7(r) &= (1-\hat{r})_+^8(32\hat{r}^3+25\hat{r}^2+8\hat{r}+1) \\
\varphi_3(r) &= (1-\hat{r})_+^5(8\hat{r}^2+5\hat{r}+1) & \varphi_8(r) &= (1-\hat{r})_+^3 \\
\varphi_4(r) &= (1-\hat{r})_+^2 & \varphi_9(r) &= (1-\hat{r})_+^3(5\hat{r}+1) \\
\varphi_5(r) &= (1-\hat{r})_+^4(4\hat{r}+1) & \varphi_{10}(r) &= (1-\hat{r})_+^7(16\hat{r}^2+7\hat{r}+1)
\end{aligned} \tag{5}$$

where $\hat{r} = \epsilon r$ and ϵ is the shape parameter of the radial basis function, see Fig. 1 for a visualization of (5).

2.1 RBF Interpolation

The RBF interpolation was introduced by [13]. It uses the formula (1) and the radial basis functions are placed into the location of input data. The interpolation formula has the following form

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

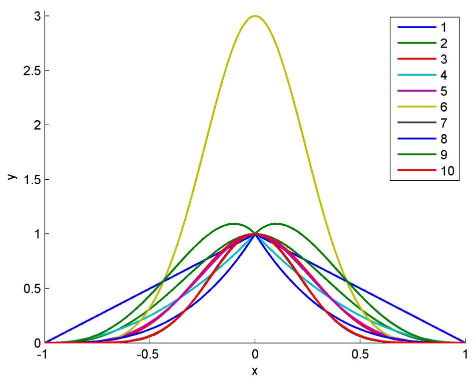


Fig. 1. Examples of CSRBF from (5).

$$h_i = f(\boldsymbol{x}_i) = \sum_{j=1}^{N} \lambda_j \varphi\left(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\right) \quad \text{for } \forall i \in \{1, \dots, N\}, \tag{6}$$

where N is the number of input points and h_i is the function value at point $\boldsymbol{x}_i$. The equation (6) can be rewritten in a matrix form as

$$\boldsymbol{A}\boldsymbol{\lambda} = \boldsymbol{h}. \tag{7}$$

As $\varphi\left(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|\right) = \varphi\left(\|\boldsymbol{x}_j - \boldsymbol{x}_i\|\right)$, the matrix $\boldsymbol{A}$ is symmetric.

2.2 RBF Approximation

The RBF approximation [8], [17], [21] is based on the RBF interpolation. The number of radial basis functions is smaller than the number of input points. It leads to an over-determined system of linear equations

$$h_i = f(\boldsymbol{x}_i) = \sum_{j=1}^{M} \lambda_j \varphi\left(\|\boldsymbol{x}_i - \boldsymbol{\xi}_j\|\right) \quad \text{for } \forall i \in \{1, \dots, N\}, \tag{8}$$

where $\boldsymbol{\xi}_j$ is the center of radial basis function, M is the number of radial basis functions and $M < N$ (mostly $M \ll N$). It is not possible to fulfill all equations at once in (8), so we need to use the least squares errors method (LSE) using the following formula

$$\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{\lambda} = \boldsymbol{A}^T\boldsymbol{h}, \tag{9}$$

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

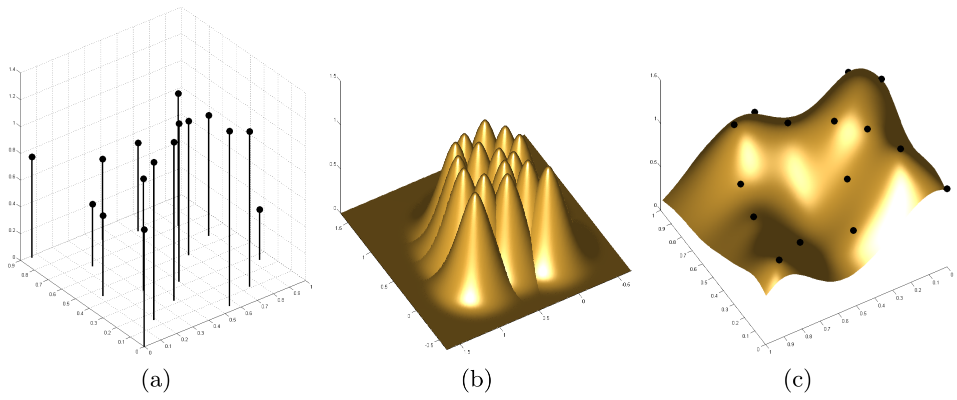


Fig. 2. The visualization of data values (a), the RBF collocation functions (b) and the resulting interpolant (c).

where $\boldsymbol{A}^T\boldsymbol{A}$ is a symmetric matrix.

3 Proposed Approach using CSRBF

The compactly supported radial basis functions (CSRBFs) have limited influence of each radial basis function given by the radius, which is defined as

$$r_{max} = \frac{1}{\epsilon}, \tag{10}$$

where ϵ is the shape parameter of the local radial function.

Due to using CSRBFs, the interpolation or approximation matrix $\boldsymbol{A}$ (in (7) or (9)) is sparse (we assume that the maximum distance of input points is larger than r_{max}). Solving the system of linear equations (7) or (9), the vector of weights $\boldsymbol{\lambda}$ is obtained.

Using the lambda ($\boldsymbol{\lambda}$) coefficients, we can compute the function value of RBF at a location $\boldsymbol{x}$ using the formula

$$f(\boldsymbol{x}) = \sum_{j=1}^{M} \lambda_j \varphi\left(\|\boldsymbol{x} - \boldsymbol{\xi}_j\|\right), \tag{11}$$

where M is the number of radial basis functions and $\boldsymbol{\xi}_j$ is the location of radial basis function, i.e. its center.

The number of radial basis functions can be very high for real-world approximation and interpolation problems. However, the evaluated radial basis functions $\varphi\left(\|\boldsymbol{x} - \boldsymbol{\xi}_j\|\right)$ are mostly equal to zero as the influence of each radial basis function is limited to the maximal radius $r_{max} = 1/\epsilon$. However, evaluation of such zero radial basis functions is wasting computational time and the evaluation of such zero radial basis functions should be omitted.

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

3.1 Space Subdivision

Considering the processing of very large data sets, it is necessary to deal with the efficiency of the evaluation of the final interpolant, especially if the CSRBFs are used.

The proposed approach presents an algorithm for speeding-up the evaluation of the function value of RBF when using the local radial basis functions, i.e. CSRBF (5), see [22] for interpolation of large datasets with CSRBF.

The proposed approach is advisable to combine with the method for fast interpolation using the space subdivision [25], [26]. This method also divides all the input points into a grid, computes the RBF interpolation or approximation of each cell in the grid, and finally blends all the interpolations or approximations together using simple but efficient formula. This approach is suitable for large datasets that cannot be interpolated or approximated using the standard RBF method.

To speed-up the RBF function evaluation, we need to know some relation between the position of a point (where the RBF is to be evaluated) and the location of all radial basis functions. Only the radial basis functions that are closer than $r_{max} = 1/\epsilon$ should be evaluated and used for the RBF evaluation.

Finding all of those functions every time without any special data structure could increase the computational time on the contrary. A much better solution is to use space subdivision to divide all centers of radial basis functions into a rectangular grid.

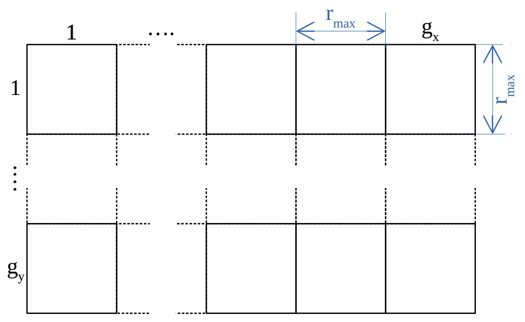


Fig. 3. Visualization of grid used for space subdivision.

Let us split the data domain into a rectangular grid, see Fig. 3. The size of each cell in the grid should be $r_{max} \times r_{max}$, i.e. the size depends on the shape parameter which can be selected as described in [11], [13], [9]. The reason for this size will be clarified later in the following chapter.

The total number of cells is

$$G = g_x g_y \tag{12}$$

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

in the $2D$ case or likewise

$$G = g_x g_y g_z \tag{13}$$

in the $3D$ case of RBF interpolation or approximation. The size g_x is computed as

$$g_x = \left\lceil \frac{x_{max} - x_{min}}{r_{max}} \right\rceil \tag{14}$$

and equations for g_y or g_z are straightforward.

Dividing all radial basis functions according to their centers into the grid is of $O(N)$ time complexity and has no influence on the final computational time of the RBF interpolation or approximation.

The next step is a standard computation of the RBF interpolation or approximation, i.e. solving a system of linear equations. As a result, we obtain the weighting coefficients of radial basis functions, i.e. the lambda coefficients ($\boldsymbol{\lambda}$). Each lambda coefficient is associated with exactly one radial basis function. These coefficients are also divided into the grid and associated with corresponding radial basis functions.

The additional time complexity required by the proposed algorithm to the standard RBF computation can be considered as the preprocessing and is

$$O(N + N) \approx O(N), \tag{15}$$

where the first $O(N)$ is for the division of points into the grid. The second $O(N)$ is for the division of the calculated lambda values (λ_i) into the grid, i.e. associating them with the corresponding radial basis functions.

3.2 RBF Function Value Computation

When the RBF is traditionally evaluated, all the radial basis functions are evaluated for the input point $\boldsymbol{x}$ and multiplied with the weighting lambda coefficients. However, many radial basis functions for the input point $\boldsymbol{x}$ are equal zero because the input point $\boldsymbol{x}$ is more distant than r_{max} ($r_{max} = 1/\epsilon$) from the centers of radial basis functions.

The point $\boldsymbol{x}$ that belongs to one cell of the grid can only affect the value of some radial basis functions, i.e. others will be zero. The visualization of the influence is in Fig. 4. It can be seen, that the point that belongs to one cell can only affect the evaluation of radial basis functions in that cell and all one-neighborhood cells around.

The RBF function value is computed at a point $\boldsymbol{x}$. The first step of the proposed algorithm is the location of the cell in the grid, where this point belongs to. For the RBF function value evaluation, we will use only radial basis functions (with associated computed weights) of the cell where $\boldsymbol{x}$ lies and all the one-neighborhood cells around. All other radial basis functions can be skipped as their value is always zero and their evaluation and multiplication with their weights only increase the computational time while not giving any additional increment for the sum of weighted radial basis functions.

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

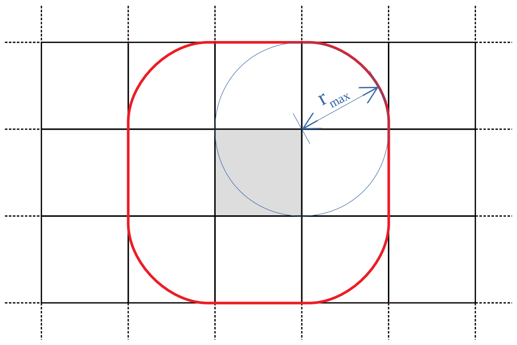


Fig. 4. The area of influence (red color) for evaluation of radial basis functions when the point for function evaluation lies in the marked cell.

The computation of RBF function value using the proposed approach decreases the computational time compared to the standard RBF function evaluation.

3.3 Theoretical Speed-up of RBF Function Value Computation

In this chapter, the expected speed-up of our algorithm for RBF function value computation is analyzed. Let us assume a uniform distribution of points. Then the time complexity of the standard evaluation of a function using RBF is

$$O(M), \tag{16}$$

where M is the number of radial basis functions.

The time complexity of the proposed algorithm for evaluation of Radial basis function is

$$O\left(3^d \frac{M}{G}\right), \tag{17}$$

where d is the dimension, i.e. $d = 2$ for $2D$ and $d = 3$ for $3D$, the G represents the total number of cells in grid.

The expected speed-up of our proposed algorithm to the standard one is computed as

$$v \approx \frac{O(M)}{O\left(3^d \frac{M}{G}\right)}, \tag{18}$$

which is equal

$$v \approx \frac{G}{3^d}. \tag{19}$$

For G higher than 3^d the speed-up is higher than one, i.e. our proposed algorithm is faster.

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

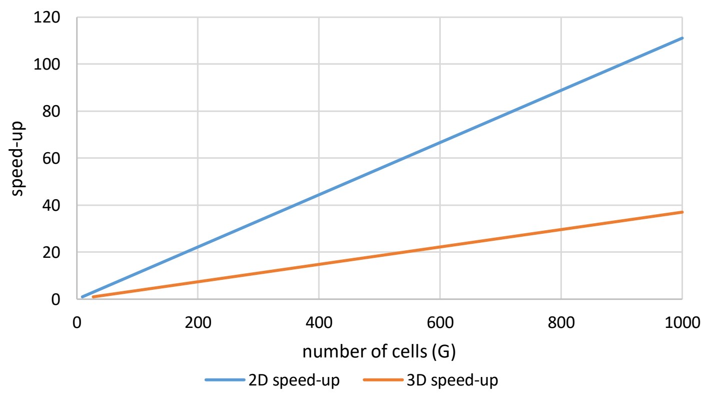


Fig. 5. The theoretical speed-up of evaluation of RBF in $2D$ and $3D$.

The value of G, i.e. the number of cells in the grid, depends on r_{max} and thus on the shape parameter ϵ of radial basis function. Usually r_{max} is much smaller than the range of the input data in each axis, i.e. the grid size will be bigger than 3×3 (in $2D$) or $3 \times 3 \times 3$ (in $3D$). Thus the speed-up will be higher than one, see Fig. 5.

4 Experimental Results

The theoretical speed-up needs to be confirmed experimentally for both $2D$ and $3D$ RBF cases. In our experiments, we created data sets with a different number of randomly distributed input points with a uniform distribution and associated function values. We also tested the proposed approach with real large datasets of LiDAR data[1] (see Fig. 6 for visualization of RBF interpolation). Then the RBF interpolation with the following CSRBF radial basis function was used.

$$\varphi_5(r) = (1 - \epsilon r)_+^4 (4\epsilon r + 1), \tag{20}$$

where this radial basis function is C^2 smooth at the origin, i.e. it is continuous and has the first and the second continuous derivative. The experimental results do not depend on the dataset values, it only depends on the number of cells. The number of cells is determined according to the shape parameter. Due to this, we do not discuss the type of interpolated data as the results are valid for any dataset with the same number of input points and the same number of cells.

In the experiments the radial basis functions were divided into grids of different sizes (see Tab. 1 for $2D$ and Tab. 2 for $3D$).

We evaluated each RBF in 10^9 random points with uniform distribution to obtain a sufficiently large number of measurements. For the evaluation, we

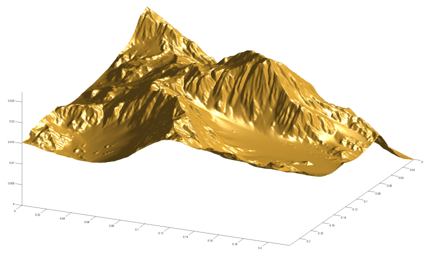
[1] https://liblas.org/

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

used both our proposed approach with space subdivision and also the standard approach without space subdivision. The speed-up of our proposed algorithm compared to the standard one is summarized in Tab. 1 for $2D$ and Tab. 2 for $3D$.

Table 1. The measured speed-up of 2D RBF evaluation using the proposed approach.

	number of radial basis functions	number of cells	speed-up
synthetic data sets	1 000	16	1.80
	5 000	64	6.81
	10 000	100	10.88
	10 000	225	24.53
	100 000	1 089	119.19
	1 000 000	10 000	1 097.78
real data sets	131 044	1 296	141.98
	756 150	7 350	806.46


(a)

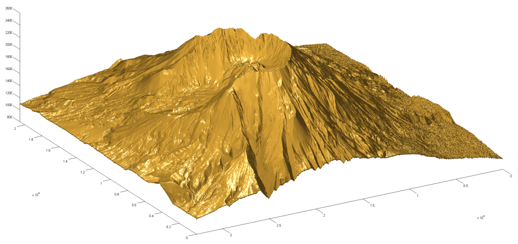

(b)

Fig. 6. RBF interpolation of real datasets. The part of Alps mountain consist of 131 044 points (a) and the mountain of Saint Helens consists of 756 150 points (b).

It can be seen, that even for a small number of radial basis functions and a relatively small number of cells the proposed algorithm is significantly faster. With an increasing number of cells, the speed-up increases as well. For large datasets, the speed-up can be even larger than 100, and thus our proposed algorithm can save a lot of computational time during the evaluation of RBF function. The proposed approach was tested with real datasets as well and proved its ability for high speed-up of RBF function evaluation, see Tab. 1.

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

Table 2. The measured speed-up of 3D RBF evaluation using the proposed approach.

	number of radial basis functions	number of cells	speed-up
	1 000	27	1.01
	5 000	80	2.85
synthetic	10 000	125	4.54
data sets	10 000	216	7.86
	100 000	1 000	36.56
	1 000 000	10 648	390.03

5 Conclusion

In this contribution, we have presented a new approach for speeding up the evaluation of Radial basis functions (RBF). The proposed approach uses the space-subdivision to select only appropriate locations of radial basis functions that are used to evaluate the RBF. This proposed approach is easy to implement and can be used for any dimension.

The proposed approach has significant speed-up compared to the standard one evaluation of RBF. This speed-up was confirmed by experiments that were made for $2D$ as well as for $3D$ RBF using Matlab / Octave.

Acknowledgments

The authors would like to thank their colleagues at the University of West Bohemia, Plzen, for their discussions and suggestions. Also, great thanks belong to our colleague Martin Cervenka for his hints and ideas. The research was supported by projects Czech Science Foundation (GACR) No. GA17-05534S, by the Ministry of Education, Youth, and Sport of Czech Republic - University spec. research - 1311, and partially by SGS 2019-016.

References

1. B. Adams and M. Wicke. Meshless approximation methods and applications in physics based modeling and animation. In *Eurographics (Tutorials)*, pages 213–239, 2009.
2. S. Atluri, H. Liu, and Z. Han. Meshless local petrov-galerkin (MLPG) mixed finite difference method for solid mechanics. *Computer modeling in engineering and sciences*, 15(1):1, 2006.
3. T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. Meshless methods: an overview and recent developments. *Computer methods in applied mechanics and engineering*, 139(1):3–47, 1996.
4. M. E. Biancolini. *Fast radial basis functions for engineering applications.* Springer, 2017.

5. M.-B. Chen. A parallel 3d delaunay triangulation method. In *2011 IEEE Ninth International Symposium on Parallel and Distributed Processing with Applications*, pages 52–56. IEEE, 2011.
6. P. Cignoni, C. Montani, and R. Scopigno. DeWall: A fast divide and conquer delaunay triangulation algorithm in Ed. *Computer-Aided Design*, 30(5):333–341, 1998.
7. P. J. Davis. *Interpolation and approximation.* Courier Corporation, 1975.
8. G. E. Fasshauer. *Meshfree approximation methods with MATLAB*, volume 6. World Scientific, 2007.
9. G. E. Fasshauer and J. G. Zhang. On choosing optimal shape parameters for rbf approximation. *Numerical Algorithms*, 45(1-4):345–368, 2007.
10. A. J. Ferreira, E. J. Kansa, G. E. Fasshauer, and V. Leitão. *Progress on meshless methods.* Springer, 2009.
11. R. Franke. Scattered data interpolation: tests of some methods. *Mathematics of computation*, 38(157):181–200, 1982.
12. N. A. Gumerov and R. Duraiswami. Fast radial basis function interpolation via preconditioned krylov iteration. *SIAM Journal on Scientific Computing*, 29(5):1876–1899, 2007.
13. R. L. Hardy. Multiquadric equations of topography and other irregular surfaces. *Journal of geophysical research*, 76(8):1905–1915, 1971.
14. E. Larsson and B. Fornberg. A numerical study of some radial basis function based solution methods for elliptic PDEs. *Computers & Mathematics with Applications*, 46(5):891–902, 2003.
15. Y. Liu and J. Snoeyink. A comparison of five implementations of 3D delaunay tessellation. *Combinatorial and Computational Geometry*, 52(439-458):56, 2005.
16. Z. Majdisova and V. Skala. Big geo data surface approximation using radial basis functions: A comparative study. *Computers & Geosciences*, 109:51–58, 2017.
17. Z. Majdisova and V. Skala. Radial basis function approximations: Comparison and applications. *Applied Mathematical Modelling*, 51:728–743, 2017.
18. R. Pan and V. Skala. A two-level approach to implicit surface modeling with compactly supported radial basis functions. *Engineering with Computers*, 27(3):299–307, 2011.
19. G. Prakash, M. Kulkarni, and U. Sripati. Using RBF neural networks and Kullback-Leibler distance to classify channel models in free space optics. In *Optical Engineering (ICOE), 2012 International Conference on*, pages 1–6. IEEE, 2012.
20. V. Rajan. Optimality of the delaunay triangulation in R^d. *Discrete & Computational Geometry*, 12(2):189–202, 1994.
21. V. Skala. Fast interpolation and approximation of scattered multidimensional and dynamic data using radial basis functions. 2013.
22. V. Skala. RBF interpolation with CSRBF of large data sets. *Procedia Computer Science*, 108:2433–2437, 2017.
23. M. Smolik and V. Skala. Highly parallel algorithm for large data in–core and out–core triangulation in E2 and E3. *Procedia Computer Science*, 51:2613–2622, 2015.
24. M. Smolik and V. Skala. Spherical RBF vector field interpolation: Experimental study. In *Applied Machine Intelligence and Informatics (SAMI), 2017 IEEE 15th International Symposium on*, pages 431–434. IEEE, 2017.
25. M. Smolik and V. Skala. Large scattered data interpolation with radial basis functions and space subdivision. *Integrated Computer-Aided Engineering*, 25(1):49–62, 2018.

ICCSA 2020 proceedings, part I, pp.165-176, LNCS 12249, 12250, DOI: 10.1007/978-3-030-58799-4_12, Springer, 2020

26. M. Smolik and V. Skala. Efficient simple large scattered 3D vector fields radial basis functions approximation using space subdivision. In *International Conference on Computational Science and Its Applications*, pages 337–350. Springer, 2019.
27. M. Smolik, V. Skala, and Z. Majdisova. Vector field radial basis function approximation. *Advances in Engineering Software*, 123:117–129, 2018.
28. C. E. Torres and L. A. Barba. Fast radial basis function interpolation with gaussians by localization and iteration. *Journal of Computational Physics*, 228(14):4976–4999, 2009.
29. K. Uhlir and V. Skala. Reconstruction of damaged images using radial basis functions. In *Signal Processing Conference, 2005 13th European*, pages 1–4. IEEE, 2005.
30. H. Wendland. Computational aspects of radial basis function approximation. *Studies in Computational Mathematics*, 12:231–256, 2006.
31. L. Yingwei, N. Sundararajan, and P. Saratchandran. Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm. *IEEE Transactions on neural networks*, 9(2):308–318, 1998.
32. X. Zhang, K. Z. Song, M. W. Lu, and X. Liu. Meshless methods based on collocation with radial basis functions. *Computational mechanics*, 26(4):333–343, 2000.