# Variable Shape Parameters in Radial Basis Function Interpolation

Julian Myers

December 9, 2024

# Contents

# 1 Introduction

Interpolation methods are a fundamental and particularly useful subject in the field of numerical analysis. Given a set of points, interpolation methods can be used to find an approximate function which could potentially predict new points[2] based on the few that points are given. There exist many different interpolation methods including polynomial, linear, and spline interpolation. One particularly interesting method is Radial Basis Function Interpolation (RBF). RBF methods stand out for their flexibility and extensibility to higher dimensions. Hence, the application of RBF methods are seen in a wide variety of disciplines, such as machine, topography, fluid dynamics, and many others[24][8].

A significant component of the RBF is its shape parameter. A constant shape parameter is necessary for allowing the theory behind the RBF to be more digestible and understandable. Thus, initial implementation and theory were conducted by using a constant shape parameter[24]. The shape parameter is very consequential in determining the efficacy of RBF methods, as in choosing a large shape parameter makes for a more well conditioned matrix but results in a less accurate RBF approximation [24]. Choosing a small shape parameters leads to a more accurate interpolation, but causes the matrix to be ill-conditioned. This phenomenon is known as the uncertainty principle [22][24][19].

There exists the possibility of using a variable shape parameter in order to remedy some of the conditioning problems that arise with RBF interpolation[21]. In this paper, we explore some of the different strategies for using a variable shape parameter RBF methods.

# 2 Background on RBF Interpolation

## 2.1 Motivation for RBF Interpolation

One of the primary motivations for using RBF interpolation is its flexibility in handling scattered data in arbitrary dimensions. Unlike polynomial or spline interpolation, which are generally constrained to structured grids or specific dimensions, RBF methods can efficiently interpolate irregular datasets without requiring additional grid construction. This property makes RBF interpolation highly suitable for applications in fields such as geostatistics, image reconstruction, and machine learning. For example, spline-based methods are effective for one-dimensional or regularly gridded two-dimensional data but become computationally expensive and less effective as dimensionality increases. RBFs, by contrast, provide a straightforward extension to higher dimensions without significant loss in computational feasibility.

RBF interpolation is inherently mesh-free[16], meaning it does not require a predefined connectivity structure among data points. This contrasts with methods like finite element or finite difference interpolation, which depend on constructing and maintaining complex mesh structures[4]. The mesh-free property simplifies the implementation and is particularly advantageous in problems involving moving boundaries or dynamically changing datasets, such as those encountered in fluid dynamics, computer graphics, and more generally partial differential equations [28].

RBFs offer a high degree of accuracy due to their globally or locally defined basis functions, which are capable of capturing complex relationships within data. This makes them competitive with or superior to other methods, especially when dealing with smooth functions or functions with varying behavior across the domain. Polynomial methods, for instance, suffer from Runge's phenomenon in high-degree approximations, whereas RBF methods do not encounter such issues because of their localized nature. An added advantage of RBF methods is their adaptability through the selection of shape parameters, allowing for the fine-tuning of interpolation performance. Traditional methods like polynomials or splines lack similar adaptability and often require manual adjustment of degree or knot placement to achieve comparable results. The shape parameter in RBF interpolation enables users to balance accuracy and stability, particularly in regions with varying data density. Unlike methods such as splines, which require different formulations for one-dimensional, two-dimensional, or three-dimensional data, RBF interpolation employs a unified mathematical framework that can be directly applied to problems in any dimension. This consistency reduces complexity when extending algorithms to higher-dimensional applications, a task that often requires significant reformulation in other interpolation methods.

RBF interpolation is also valued for its robustness in solving numerical problems, including partial differential equations (PDE's) and data reconstruction. Polynomial methods can exhibit instability in the presence of noise or uneven data distribution, while RBF interpolation can accommodate these challenges with greater stability. This robustness has contributed to its adoption in engineering, scientific computing, and signal processing Motivations for choosing RBF interpolation over traditional methods stem from its ability to handle scattered and multidimensional data, its mesh-free nature, high approximation accuracy, adaptability through shape parameters, and unified framework across dimensions. These advantages position RBFs as a versatile and powerful tool for a wide range of interpolation tasks, driving their widespread use in both theoretical and applied research domains.

## 2.2 RBF Interpolation of Scattered Data

Given a set of $N$ distinct centers $x_1^c, x_2^c, \ldots, x_N^c \in \Omega \subset \mathbb{R}^d$, the RBF interpolant is

$$I_N f(x) = \sum_{i=1}^{N} a_i \phi(||x - x_i^c||_2, \epsilon_i) \tag{1}$$

where $a_i$ is the expansion coefficients, $\phi$ is the RBF, and $\epsilon_i$ is the shape parameter, and

$$I_N f(x_i^c) = f(x_i^c), \qquad i = 1, 2, \ldots, N \tag{2}$$

is the interpolation conditions.

There a four widely used choices for an RBF $\phi(r)$,

$$\text{Gaussian(GA): } \phi(r) = e^{-(\epsilon r)^2} \tag{3}$$

$$\text{Multiquadric(MQ): } \phi(r) = \sqrt{1 + (\epsilon r)^2} \tag{4}$$

$$\text{Inverse Quadratic(IQ): } \phi(r) = \frac{1}{1 + (\epsilon r)^2} \tag{5}$$

$$\text{Inverse Multiquadric(IMQ): } \phi(r) = \frac{1}{\sqrt{1 + (\epsilon r)^2}} \tag{6}$$

Using the interpolation conditions, we get a linear equation

$$Ba = f \tag{7}$$

where $B$ is an $N \times N$ system matrix with elements

$$b_{ij} = \phi(||x_i^c - x_j^c||_2, \epsilon_i), \qquad i, j = 1, 2, \ldots, N. \tag{8}$$

Solving (7) for $a$ gives us the expansion coefficients which are then used in the interpolant function (1) to solve

$$f_a = Ha. \tag{9}$$

The $M \times N$ evaluation matrix $H$ has elements

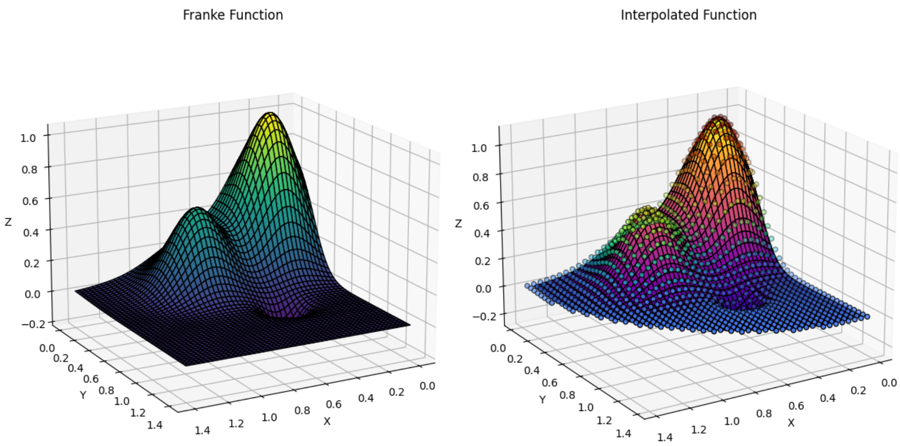
$$h_{ij} = \phi(||x_i - x_j^c||_2, \epsilon_j), \qquad i = 1, 2, \ldots, M \text{ and } j = 1, 2, \ldots, N. \tag{10}$$

Thus, satisfying the information required by the interpolant (1).

## 2.3 The Role of the Shape Parameter

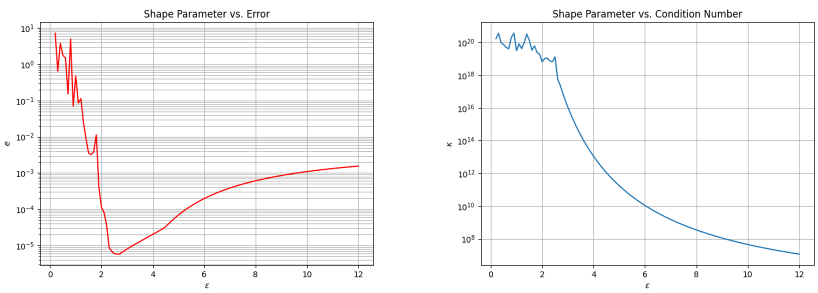
Using a python program, the Franke function [4] was interpolated using the Mulitquadric RBF kernel which produced the following.

Figure 1: Left: Franke Function, right:Interpolation




To show the significance of the shape parameter in RBF interpolation, the program iterates through an array of shape parameters begin from $\epsilon = 0.2$ to $\epsilon = 12.0$ which gave the following plots.

Figure 2: Left:Error vs. $\epsilon$, right: $\kappa$ vs. $\epsilon$




As we can see, beyond $\epsilon \approx 2.2$, the condition number decreases as the error increases following the uncertainty principle.

## 2.4 Challenges of Fixed Shape Parameters

While constant shape parameter RBF methods have been widely used for interpolation tasks due to their relative simplicity, they exhibit notable limitations that can adversely affect their performance in various scenarios.

One of the primary drawbacks of constant shape parameter methods is their inability to adapt to variations in data density or local behavior. A single shape parameter applied uniformly across the entire domain may lead to suboptimal interpolation accuracy in regions where data points are sparsely distributed or where the underlying function exhibits rapid changes. For instance, flatter basis functions may over-smooth data in dense regions, while narrower functions may fail to capture significant variations in sparse regions

The choice of a constant shape parameter involves a compromise between numerical accuracy and stability. Small shape parameters yield more accurate results for smooth functions but can lead to ill-conditioned interpolation matrices, especially for large datasets. Conversely, larger shape parameters improve matrix conditioning but reduce interpolation accuracy, particularly in capturing finer details of the target function.

Selecting an appropriate constant shape parameter is non-trivial and problem-specific, often requiring extensive trial and error or heuristic methods. An unsuitable choice can drastically reduce the method's effectiveness, making it difficult to apply constant shape parameter methods universally without prior knowledge of the dataset or function.

# 3 Variable Shape Parameter Strategies

There are several methods for choosing or defining a variable shape parameter. A few of such strategies include analytical approaches, machine learning algorithms and statistical techniques, or random shape parameters. [15][21][24].

## 3.1 Analytical Approaches

Early approaches to incorporate a variable shape parameter to find a balance between the condition number and accuracy came in the form of defining the shape parameter as a mathematical expression. One such method, in [9], is

### 3.1.1 Exponentially Varying Shape Parameter

$$\epsilon_j = \left[\epsilon_{\text{min}}^2 \left(\frac{\epsilon_{\text{max}}^2}{\epsilon_{\text{min}}^2}\right)^{\frac{j-1}{N-1}}\right]^{\frac{1}{2}}, \qquad j = 1, \dots, N. \tag{11}$$

This strategy, called the Exponentially Varying Shape Parameter (ESP) was shown in [10] to be very successful given certain conditions of $\epsilon_{\text{min}}^2$ and $\epsilon_{\text{max}}^2$. Namely, when the values of $\epsilon_{\text{min}}^2$ and $\epsilon_{\text{max}}^2$ had varied by several orders of magnitude.

### 3.1.2 Linearly Varying Shape Parameter

Another strategy shown in [19] is

$$\epsilon_j = \epsilon_{\text{min}} + \left(\frac{\epsilon_{\text{max}} - \epsilon_{\text{min}}}{N-1}\right)j, \qquad j = 0, 1, \dots, N-1, \tag{12}$$

called linearly varying shape parameter (VSP). In [19], numerical experiments showed that (11) outperforms (12) in accuracy for two numerical experiments where each were used to interpolate a function and then solve an elliptical PDE.

### 3.1.3 Random Shape Parameter

Introduced in [21], the Random Shape Parameter (RSP) in numerical experiments had showed great accuracy in terms of results.

$$\epsilon_j = \epsilon_{min} + (\epsilon_{max} - \epsilon_{min}) \times rand(1, N) \tag{13}$$

Note that *rand* here refers to Matlab's random function. For the numerical experiments performed later on, [14] uses Numpy's random module to replicate it.

### 3.1.4 Trigonometric Shape Parameter

From [29], the author introduces the strategy

$$\epsilon_{\text{min}} + (\epsilon_{\text{max}} + \epsilon_{\text{min}}) \sin(j) \tag{14}$$

called the Trigonometric Shape Parameter (TSP). Numerical experiments have shown great success, however, this strategy causes negative shape parameter values [6], [7].

### 3.1.5 Sinusoidal Shape Parameter

As a solution to the negative side effects of using the TSP, in [6], the author proposed the Sinusoidal Shape Parameter:

$$\epsilon_j = \epsilon_{\text{min}} + (\epsilon_{\text{max}} - \epsilon_{\text{min}}) \sin\left( (j-1) \frac{\pi}{2(N-1)} \right) \tag{15}$$

### 3.1.6 Decreasing Linear Shape Parameter

In [7], the authors also mention the Decreasing Linear Shape Parameter (DSLP), which is similar to LSP, but decreases instead.

$$\epsilon_j = \epsilon_{\text{max}} + \left( \frac{\epsilon_{\text{min}} - \epsilon_{\text{max}}}{N-1} \right) j, \qquad j = 0, 1, \ldots, N-1, \tag{16}$$

### 3.1.7 Hybrid Shape Parameter

This one is also introduced in [6]. It makes use of three different variable shape parameter (SSP, DLSP, and ESP), strategies which prevents monotonicity and randomness.

$$\epsilon_j = \begin{cases} \text{SSP}_j & j = 3k+1 \\ \text{DLSP}_j & j = 3k+2 \\ \text{ESP}_j & j = 3k+3 \end{cases} \tag{17}$$

### 3.1.8 Binary Shape Parameter

The last variable shape parameter we will observe, is the Binary Shape Parameter (BSP), proposed in [7].

$$\epsilon_j = \begin{cases} \epsilon_{\min} & j = 2k+1 \\ \epsilon_{\max} & j = 2k \end{cases} \tag{18}$$

The BSP is notably simple while still being non-monotonic.

## 3.2 Machine Learning

More recently, strategies of training Neural Networks (NN) to predict an optimal shape parameter for each center have been proposed in [15], which show promise with respect to a constant shape parameter strategy and more traditional ways of solving numerical PDE's. However, in many tests, the NN failed to outperform even the constant shape parameter strategy. Additionally, while the results show that the NN strategy has similar computation times compared to other methods of solving PDE's, this notably did not account for the time it took to train the NN.

# 4 Numerical Experiments

For the following numerical experiments, the previous 8 analytic variable shape parameter strategies were compared to the performance of a constant shape parameter (CSP) in interpolating a one-dimensional function. The experiments were conducted using python scripts that can be found here [14].

The first test interpolates the function

$$f(x) = e^{\sin(\pi x)} \tag{19}$$

on the interval $[0, 1]$.

The second test interpolates the function

$$f(x) = x^3 + 3x^2 + 12x + 6 \tag{20}$$

on the interval $[0, 1]$.

The third test interpolates the function

$$f(x) = \frac{1}{x^2 + 1} \tag{21}$$

(i.e. Runge's Function) on the interval $[0, 1]$

For the sake of an accurate comparison, it is crucial that the condition numbers for each system matrix be as similar as possible for each VSP. To accomplish this, a search method using the function *scipy.optimize.dual_annealing* was created which would search in ranges and find optimal $\epsilon_{\min}$'s and $\epsilon_{\max}$'s such that the condition numbers were as close to the condition number of the CSP strategy for each VSP tested.

Centers were selected by using the function *R1Points* from the the Python Radial Basis Function Toolkit [20] based on [26]. Each test interpolates with $N = 200$ centers and $M = 250$ evaluation points.

## 4.1 Results of Test 1

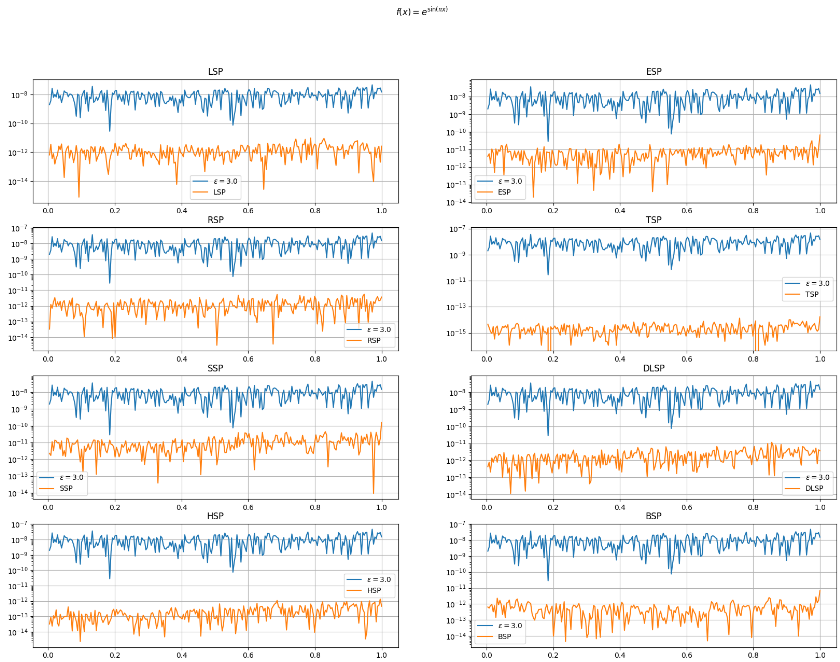



Figure 3: Point Wise Errors for each VSP

| $\epsilon$ | Max Error | $\kappa$ | Average Error | $\epsilon_{\text{min}}$ | $\epsilon_{\text{max}}$ |
|---|---|---|---|---|---|
| CSP | 4.61210e-08 | 6.86613e+18 | 1.00880e-08 | 3.00000 | 3.00000 |
| LSP | 9.59555e-12 | 6.87559e+18 | 1.64175e-12 | 1.76570 | 8.47396 |
| ESP | 6.45741e-11 | 6.87380e+18 | 6.67425e-12 | 2.18705 | 14.82776 |
| RSP | 5.49538e-12 | 6.86687e+18 | 1.36323e-12 | 2.67358 | 3.13833 |
| TSP | 1.68754e-14 | 6.81102e+18 | 2.58904e-15 | 2.29680 | 8.19247 |
| SSP | 1.60071e-10 | 6.85672e+18 | 1.10625e-11 | 8.12386 | 14.15324 |
| DLSP | 1.14281e-11 | 6.86541e+18 | 2.06291e-12 | 0.89994 | 4.36451 |
| HSP | 1.33982e-12 | 6.84866e+18 | 2.25931e-13 | 5.25059 | 12.46560 |
| BSP | 6.76259e-12 | 6.83692e+18 | 5.80355e-13 | 7.37997 | 14.86867 |

## 4.2 Results of Test 2

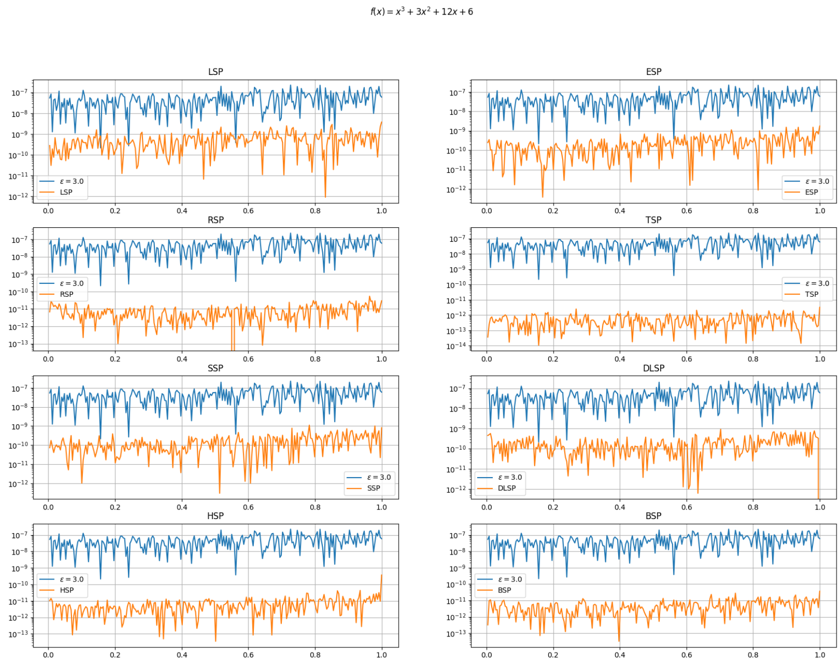



Figure 4: Point Wise Errors for each VSP

| $\epsilon$ | Max Error | $\kappa$ | Average Error | $\epsilon_{\min}$ | $\epsilon_{\max}$ |
|---|---|---|---|---|---|
| CSP | 2.26179e-07 | 6.86613e+18 | 5.39897e-08 | 3.00000 | 3.00000 |
| LSP | 3.72529e-09 | 6.85868e+18 | 5.98632e-10 | 8.17541 | 8.70833 |
| ESP | 1.74623e-09 | 6.86921e+18 | 2.62611e-10 | 8.77465 | 16.79031 |
| RSP | 5.37241e-11 | 6.85518e+18 | 8.14270e-12 | 8.50702 | 10.59205 |
| TSP | 3.18323e-12 | 6.81102e+18 | 5.53090e-13 | 2.29680 | 8.19247 |
| SSP | 1.11401e-09 | 6.88995e+18 | 1.66596e-10 | 5.66976 | 8.30512 |
| DLSP | 9.11545e-10 | 6.86001e+18 | 1.91953e-10 | 5.96134 | 6.82129 |
| HSP | 3.71074e-10 | 6.86586e+18 | 8.23707e-12 | 7.15324 | 16.94056 |
| BSP | 3.63798e-11 | 6.86875e+18 | 5.25165e-12 | 8.87950 | 10.44328 |

## 4.3 Results of Test 3

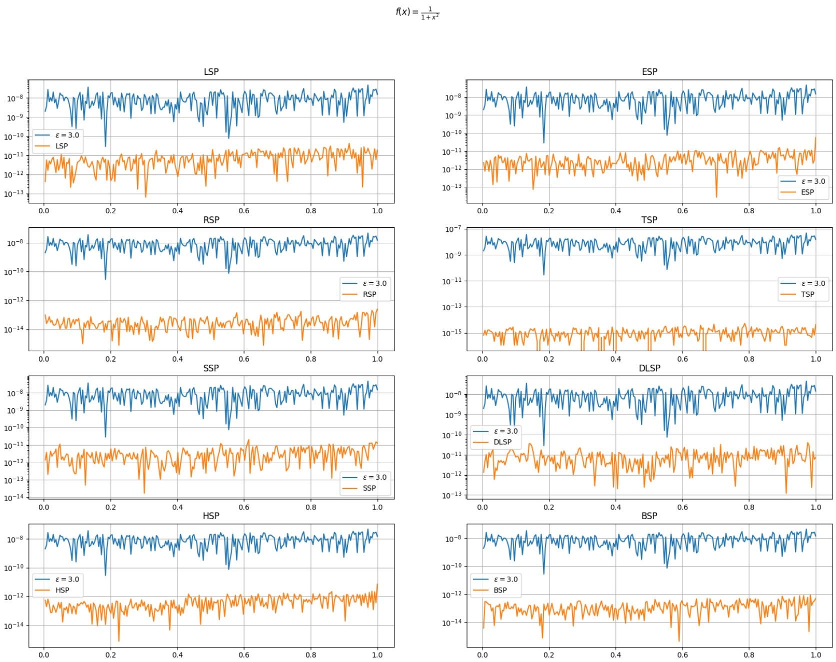



Figure 5: Point Wise Errors for each VSP

| $\epsilon$ | Max Error | $\kappa$ | Average Error | $\epsilon_{\text{min}}$ | $\epsilon_{\text{max}}$ |
|---|---|---|---|---|---|
| CSP | 4.61210e-08 | 6.86613e+18 | 1.00880e-08 | 3.00000 | 3.00000 |
| LSP | 4.09452e-11 | 6.87625e+18 | 8.28910e-12 | 8.87232 | 16.90561 |
| ESP | 5.63887e-11 | 6.86407e+18 | 3.92948e-12 | 2.79842 | 14.11317 |
| RSP | 2.43361e-13 | 6.85967e+18 | 3.85683e-14 | 8.62992 | 18.21255 |
| TSP | 5.21805e-15 | 6.90454e+18 | 1.20393e-15 | 4.23908 | 7.99555 |
| SSP | 2.03593e-11 | 6.88995e+18 | 3.53605e-12 | 5.66976 | 8.30512 |
| DLSP | 4.00269e-11 | 6.86001e+18 | 8.77729e-12 | 5.96134 | 6.82129 |
| HSP | 7.04858e-12 | 6.86586e+18 | 5.13644e-13 | 7.15324 | 16.94056 |
| BSP | 8.93619e-13 | 6.86875e+18 | 1.64494e-13 | 8.87950 | 10.44328 |

# 5 Conclusion

Each test shows that for all eight VSP's, their accuracy was at least 2 orders of magnitude better than that of the CSP as expected. From the numerical experiments, the tests show two notably effective strategies in terms of accuracy compared to the rest: TSP and RSP. Unfortunately each strategy comes with drawbacks, RSP being random and TSP producing potentially negative shape parameters. The two best performing strategies that aren't either of those two are the two strategies proposed in [7], with BSP showing slightly better results in two out of the three experiments. Considering that it is non-monotonic and non-random, for 1-dimensional interpolation, the BSP makes a good choice for VSP.

## 5.1 Further Research

Using similar methods in the future, numerical experiments ought to be conducted for higher dimension interpolation problems and for solving PDE's as well. Similarly, more numerical experiments on functions who's graphical behavior varies more than the three tested here should be considered.

# References

[1] M. Bozzini, L. Lenarduzzi, Milvia Rossini, and Robert Schaback. Interpolation by basis functions of different scales and shapes. *Calcolo*, 41, 11 2000.

[2] Richard L. Burden and J. Douglas Faires. *Numerical Analysis*. Brooks Cole, 511 Forest Lodge Road, Pacific Grove, CA 93950, 7th edition edition, 2004.

[3] Roberto Cavoretto. Adaptive radial basis function partition of unity interpolation: A bivariate algorithm for unstructured data. *Journal of Scientific Computing*, 87(41):0–1, January 2024.

[4] Richard Franke. Scattered data interpolation: Tests of some methods. *Mathematics of Computation*, 38(157):181–200, January 1982.

[5] Shabnam Sadat Seyed Ghalichi, Majid Amirfakhrian, and Tofigh Allahviranloo. An algorithm for choosing a good shape parameter for radial basis functions method with a case study in image processing. *Results in Applied Mathematics*, 16:10337, 11 2022.

[6] A. Golbabai and H. Rabiei. Hybrid shape parameter strategy for the rbf approximation of vibrating systems. *International Journal of Computer Mathematics*, 89(17):2410–2427, 2012.

[7] Ahmad Golbabai, Ehsan Mohebianfar, and Hamed Rabiei. On the new variable shape parameter strategies for radial basis functions. *Computational and Applied Mathematics*, 34, 01 2014.

[8] Rolland L. Hardy. Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, 76(8):1905–1915, 3 1971.

[9] Edward Kansa. Multiquadrics-a scattered data approximation scheme with applications to computational fluid-dynamics–ii solutions to

parabolic, hyperbolic and elliptical partial differential equations. *Computers and Mathematics*, 19(8-9):147–161, September 2002.

[10] E.J. Kansa and R.E. Carlson. Improved accuracy of multiquadric interpolation using variable shape parameters. *Computers and Mathematics with Applications*, 24(12):99–120, September 2002.

[11] Kai Li, Dongxiang Jiang, Kai Xiong, and Yongshan Ding. Application of rbf and sofm neural networks on vibration fault diagnosis for aero-engines. In *Advances In Neural Networks*, Lecture Notes in Computer Science, pages 414–419, Heidelberg, Berlin, 2006. Springer.

[12] Shin-Ruei Lin, D.L. Young, and Chuin-Shan Chen. Ghost-point based radial basis function collocation methods with variable shape parameters. *Engineering Analysis with Boundary Elements*, 130:40–48, 2021.

[13] Maans Magnusson, Michael Andersen, Johan Jonasson, and Aki Vehtari. Bayesian leave-one-out cross-validation for large data. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4244–4253. PMLR, 09–15 Jun 2019.

[14] Julian Myers. julian-myers/mth443-research. https://github.com/julian-myers/mth443-research. Accessed: 2024-12-08.

[15] Fatemeh Nassajian Mojarrad, Maria Han Veiga, Jan S. Hesthaven, and Philipp Öffner. A new variable shape parameter strategy for rbf approximation using neural networks. *Computers and Mathematics with Applications*, 143:151–168, 2023.

[16] Hananeh Nojavan, Saeid Abbasbandy, and Tofigh Allahviranloo. Variable shape parameter strategy in local radial basis functions collocation method for solving the 2d nonlinear coupled burgers' equations. *Mathematics*, 5(3), 2017.

[17] Aitong Huang Sanpend Zheng, Renzhong Feng. The optimal shape parameter for the least squares approximation based on the radial basis function. *Mathematics*, 8(11), 11 2020.

[18] Scott Sarra. Local radial basis function methods: Comparison improvements, and implementation. *Applied Mathematics and Physics*, 12(7):3867–3886, December 2023.

[19] Scott Sarra and Edward Kansa. Multiquadric radial basis function approximation methods for the numerical solution of partial differential equations. *Advances in Computational Mechanics*, 2, 01 2009.

[20] Scott A. Sarra. The python radial basis function toolkit. http://scottsarra.org/rbf/rbf.html.

[21] Scott A. Sarra. A random variable shape parameter strategy for radial basis function approximation methods. *Elsevier*, pages 1–18, June 2009.

[22] Robert Schaback. Error estimates and condition numbers for radial basis function interpolation. *Advances in Computational Mathematics*, 3, 03 1997.

[23] Y. Bai Scott A. Sarra. A rational radial basis function method for accurately resolving discontinuities and steep gradients. *Applied Numerical Mathematics*, (130):192–208, April 2018.

[24] Derek Sturgill. Variable shape parameter strategies in radial basis function methods. Master's thesis, Marshall University, 2009.

[25] Jian Sun, Ling Wang, and Dianxuan Gong. Model for choosing the shape parameter in the multiquadratic radial basis function interpolation of an arbitrary sine wave and its application. *Mathematics*, 11(8), 2023.

[26] Unknown. The unreasonable effectiveness of quasirandom sequences. Accessed: 2024-12-07.

[27] Aki Vehtari, Tommi Mononen, Ville Tolvanen, Tuomas Sivula, and Ole Winther. Bayesian leave-one-out cross-validation approximations for gaussian latent variable models. *Journal of Machine Learning Research*, 17(103):1–38, 2016.

[28] J. Wertz, E.J. Kansa, and L. Ling. The role of the multiquadric shape parameters in solving elliptic partial differential equations. *Computers and Mathematics with Applications*, 51(8):1335–1348, 2006. Radial Basis Functions and Related Multivariate Meshfree Approximation Methods: Theory and Applications.

[29] Song Xiang, Ke ming Wang, Yan ting Ai, Yun dong Sha, and Hong Shi. Trigonometric variable shape parameter and exponent strategy for generalized multiquadric radial basis function approximation. *Applied Mathematical Modelling*, 36(5):1931–1938, 2012.