VII International Conference on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2017
M. Papadrakakis, E. Oñate and B. Schrefler (Eds)

RADIAL BASIS FUNCTION INTERPOLATION FOR BLACK-BOX MULTI-PHYSICS SIMULATIONS

FLORIAN LINDNER*, MIRIAM MEHL* AND BENJAMIN UEKERMANN†

*Institute for Parallel and Distributed Systemes
University of Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany
e-mail: {florian.lindner, miriam.mehl}@ipvs.uni-stuttgart.de, web page: https://www.ipvs.uni-stuttgart.de

† Scientific Computing in Computer Science, Technical University of Munich
Boltzmannstraße 3, 85748 Garching b. München, Germany
e-mail: uekerman@in.tum.de, web page: http://www5.in.tum.de

Key words: Interpolation, Coupled Problems, Multiphysics Problems, Applications, Computing Methods, Radial Basis Functions

Abstract. Interpolation based on radial basis functions (RBF) is a standard data mapping method used in multi-physics coupling. It works on scattered data without requiring additional mesh topology or neighborhood information of support points. However, system matrices of the equations for the coefficients tend to be ill-conditioned. In this work, we illustrate the problem by a simple example and discuss possible remedies. Furthermore, we investigate the numerical performance of this method on uniform and non-uniform meshes with a particular focus on the coupling of black-box components where typically no information about the underlying discretization can be extracted. Radial basis function interpolation usually uses an enhancement of the radial basis functions by a global polynomial in order to properly capture constant components and linear trends in the given data. We present a method that determines this polynomial independent from the radial basis function ansatz, which substantially improves the condition number of the remaining RBF system. Furthermore, we show that a rescaling approach can be used to either increase the accuracy or improve the condition number even further by choosing radial basis functions with a smaller support radius. The results represent an intermediate state with the aim to be integrated into the multi-physics coupling library preCICE.

1 INTRODUCTION

While multi-physics simulations are an emerging field in science and industry, most groups concentrate their work on a single aspect of the simulation. Adding additional physical aspects to their simulation requires work on additional solvers, which hinders fast prototyping of coupled simulation environments.

preCICE[1] [3, 9] is a black-box, partitioned coupling library. It is developed to foster quick integration of new codes into the overall coupled model. The black-box approach enables integration of solvers with only minimal knowledge about the numerical details of the coupled codes. The software is free and open source under the LGPL 3 license. One of the crucial parts of a partitioned multi-physics coupling is data interpolation between non-matching meshes. Due to the the black-box approach, the mesh is represented as a point cloud. While solvers can optionally provide topological information using basic primitives such as edges, triangles and quads, the coupling library cannot rely on it. Also the missing information about the solvers discretization approach and mesh topology restricts the choice of applicable data mapping algorithms or the use of adaptive methods.

Several interpolation methods that work on point cloud data exist, of which interpolation by radial basis functions is among the most used. Radial basis function interpolation is well known to produce ill-conditioned systems, which harms the solution accuracy and the time-to-solution, in particular for large systems. Methods to cope with this are reviewed in this work. These methods include preconditioning [6], rescaling of the interpolant [4] and a different numerical treatment of the additional polynomial that is used to properly capture global constant or linear components of the data.

2 DATA MAPPING USING RADIAL-BASIS FUNCTIONS

Consider a function $f : \mathbb{R}^d \to \mathbb{R}$ of d variables, that is to be approximated by $S : \mathbb{R}^d \to \mathbb{R}$ based on the given values $\{f(x_i) : i = 1, \dots, n\}$. Using radial basis function interpolation, the resulting interpolant S of the function f reads

$$S(x) = \sum_{i=1}^{n} \gamma_i \, \varphi(||x - x_i||) + \beta_0 + \beta^T x, \tag{1}$$

where $\varphi : \mathbb{R}_+ \to \mathbb{R}$ is a radial basis function, $\beta \in \mathbb{R}^d$ and $|| \cdot ||$ denotes the Euclidean norm.

The interpolation conditions are $S(x_k) = f(x_k), k = 1, \dots, n$. To guarantee that the interpolation problem has a unique solution [10], we enhance the interpolation conditions with $d + 1$ equations for the polynomial:

$$\sum_{i=1}^{n} \gamma_i = 0 \text{ and } \sum_{i=1}^{n} \gamma_i \, x_i^{(j)} = 0 \quad \forall j = 1, \dots, d \tag{2}$$

[1] `www.precice.org`

One of the most common choices for the basis function φ are Gaussians

$$\varphi(x) = \exp\left(-\left(s||x||\right)^2\right) \tag{3}$$

We restrict to this basis function type in the remainder of this paper. Other common choices include Thin Plate Splines and Multi Quadrics, of which the latter two were also investigated, but not further detailed in this paper.

The interpolant S can yield a matrix $\bar{S}$ defining a linear mapping of the input values $f(x_k)$ from the input mesh to a vector δ of values in the output mesh:

$$\delta = \bar{S}\,\omega \;\text{ with }\; \omega = \begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{pmatrix} \tag{4}$$

When the row-sum of $\bar{S}$ is equal to one, the interpolation is consistent, which means that constant values are interpolated exactly. On the other hand, when the column-sum of $\bar{S}$ equals to one, the interpolation is conservative [1]. The operator can be changed from consistent to conservative and vice versa by transposing it. While obviously this operator can be both consistent and conservative, ensuring this requires acccess to the underlying solver's basis functions and is thus not suited for black-box coupling [7]. Conservative mapping preserves integral values and is, therefore, usually applied to integral quantities such as forces. Consistent mapping is usually applied to point values such as velocities or displacements. Equations (1) and (2) define the consistent formulation of the radial basis function interpolation. We do not further consider the conservative variant throughout this paper, though both variants are implemented in `preCICE`.

In order to ensure that $\bar{S}$ is a sparse matrix, we use a cut-off radius r for the basis functions φ and shift them to achieve continuous functions:

$$\varphi_{\text{local}}(x) = \begin{cases} \varphi_{\text{global}}(x) - \varphi_{\text{global}}(r) & \text{for } x < r, \\ 0 & \text{for } x \geq r. \end{cases} \tag{5}$$

2.1 Numerical Challenges & Solution Approaches

It is well known that radial basis function interpolation suffers from ill-conditioning of the resulting linear system [6]. This conditioning can be controlled by the shape parameter s of φ, i.e., by controlling the width of the basis functions. This results in a trade-off between accuracy and conditioning of the interpolation which is illustrated in Sect. 3.

A suitable choice for the shape parameter s is therefore of tremendous importance for the resulting approximation quality and convergence properties. This can be illustrated by $s \to 0$, an infinitely flat basis function. The resulting matrix would only consist of ones and thus be singular, which corresponds to condition number ∞. On the other

hand, $s \to \infty$ yields singular peaks as basis functions resulting a diagonal system matrix but very poor interpolation accuracy. In our approach, we use a cut-off value of 10^{-9}, i.e., choose the radius r such that $\varphi(r) = 10^{-9}$. Together with the definition of a basis function, a shape parameter, resulting in a carrier that includes exactly a given number of vertices m based on the maximal mesh width of the discretization can be determined. If $h_{\max}$ is the maximum mesh width of the input mesh, the corresponding shape parameter s for Gaussian basis functions is given by

$$s = \frac{\sqrt{-\ln 10^{-9}}}{m \cdot h_{\max}} \tag{6}$$

This ensures that the basis function decays to 10^{-9} within the m neighboring vertices from its center if the distance between data points is $h_{\max}$. It is common practice to include at least the nearest neighbor [4] in both the input and the output mesh [2]. For a non-uniform mesh, the largest distance should be considered [1]. While these statements are based on experiences, [5] gives a systematic approach to this issue, based on *leave-one-out cross validation* (LOOCV). Whereas naive LOOCV requires to create one interpolant for each vertex followed by an optimization step, [8] was able to simplify the algorithm such that the interpolant needs to be created only once for all vertices. Because it is still followed by the optimization step and needs to explicitely create the inverse of the interpolation matrix, it significantly increases the computational cost. The algorithm also optimises only with respect to accuracy and does not take the harming effect of the ill-conditioning into account. We, therefore, use a heuristic approach to determine the best value for m.

The ill-conditioning of the interpolation problem is aggravated by non-uniform meshes, which are used, e.g., in discontinous-Galerkin based solvers. A common representation uses a Gauss-Chebyshev distribution of nodes, which is defined by

$$x_k = \cos\left(\frac{2k-1}{2n}\pi\right), \quad k = 1, \ldots, n \tag{7}$$

for polynomial degree n in the one-dimensional interval $(-1, 1)$. Non-uniform point distribution greatly complicates finding an appropriate shape parameter and thus to control accuracy and conditioning.

An obvious remedy to the problem of the fixed shape parameter of the basis function on non-uniform meshes are adaptive methods. Techniques that use basis functions with a variable shape parameter, computed based on the local mesh density are known to reduce the condition number by four orders of magnitude, see section 3. More sophisticated methods also include adaptive refinement of the input mesh. This can result in a reduced problem size given the input data are sufficiently smooth. [10] uses a local error indicator on a coarse mesh to determine areas to refine. Both methods, however, require a definition of mesh density, e.g., based on the distance to the nearest neighbor for each data point, which would potentially compromise one of the main features of RBFs to work on scattered data.

The problem of the conditioning can also be tackled by preconditioning techniques, such as preconditioning by accelerated iterated approximate moving least squares as presented in [6]. The technique has shown good first results reducing the condition number on uniform meshes. However, achieving stability on non-uniform meshes remains problematic, as the method requires control of the distance of the interpolation system matrix to the identity matrix. If this distance is not bounded by one, convergence can not be reached. Thus, the method is not fully applicable to coupled problems with black-box solvers where the eigenvalues of the interpolation matrix are unknown.

The rescaling procedure proposed in [4] does not try to address the ill-conditioning by using a preconditioner, but improves interpolation quality, thus making it possible to use skinnier smaller basis functions, i.e., smaller values for m which, in turn, improves the condition number of the interpolation problem. The method showed very promising results during first investigations. Basically, it works by constructing a rescaled interpolant $\bar{S}$ by

$$S_r(x) = \frac{S_f(x)}{S_1(x)} \tag{8}$$

where $S_f(x)$ denotes the interpolant constructed from the given values as shown in Eq. (1) and $S_1(x)$ is the interpolant of the constant function $g(x) = 1$. Substantial improvements can be observed in particular for small m without negatively affecting the conditoning. The additional computational cost remains moderate as it only involves the computation of one additional interpolant for each set of data positions to be mapped. We present results in Sect. 3.

2.2 Our Algorithm

In this section, we show the structure of the matrices and describe the algorithm we use to compute the interpolation result. We present two variants to treat the polynomial with the second variant leading to a significant improvement in conditioning and equal accuracy.

Integrated Polynomial We construct the system matrix C that defines the linear system for the coefficients $\gamma_i, i = 1, \ldots, n$ and $\beta_i, i = 0, \ldots, d$ of interpolant S from Eq. (1) including the polynomial equations:

$$\underbrace{\begin{pmatrix} 0 & Q^T \\ Q & P \end{pmatrix}}_{C} \underbrace{\begin{pmatrix} \beta \\ \gamma \end{pmatrix}}_{p} = \begin{pmatrix} 0 \\ \omega \end{pmatrix} \tag{9}$$

where $P \in \mathbb{R}^{n \times n}, P_{i,j} = \varphi\left(||x_i - x_j||_2\right), i, j = 1, \ldots, n$ and where the i th row of $Q \in \mathbb{R}^{n \times d+1}$ is $(1\, x_i^{(1)} \ldots\, x_i^{(d)})$, i.e., holding the coordinates of the input data.

The evaluation matrix A is smiliarly constructed:

$$A = \left(\begin{array}{c|c} V & P' \end{array} \right) \tag{10}$$

with $P'_{i,j} = \varphi\left(||y_i - x_j||_2\right)$, $i = 1, \ldots, \tilde{n}$, $j = 1, \ldots, n$ and $V_{i,\cdot} = (1\, y_i^{(1)} \ldots y_i^{(d)})$. y_i, $i = 1, \ldots, \tilde{n}$ are the evaluation points in the output mesh. While P and P' account for the basis functions evaluated at the input resp. output mesh, Q and V hold the coordinates of the input or output vertices and account for the polynomial part of Eq. (1).

The output values δ are now computed by

$$\delta = A \cdot C^{-1} \omega \tag{11}$$

Separated Polynomial In the formulation above, the polynomial is embedded into the system and solved together with it. Another approach is to treat the polynomial coefficients separately by first solving a least squares problem

$$\left[Q \right] \cdot \left[\beta \right] \approx \left[\omega \right] \quad \Rightarrow \quad \beta = Q^{\dagger} \cdot \omega \tag{12}$$

for polynomial coefficients β by QR-decomposition. $Q^{\dagger}$ denotes the pseudo-inverse of Q. In the next step, the polynomial values have to be subtracted from the input values ω before the basis function coefficients γ can be computed from

$$P \cdot \gamma = \omega - Q \cdot \beta \quad \Rightarrow \quad \gamma = P^{-1} \left(\omega - Q \cdot \beta \right) \tag{13}$$

Evaluation involves adding the polynomial values computed at the output positions V:

$$\delta = P' \cdot \gamma + V \cdot \beta = P'\, P^{-1} \left(\omega - Q\, Q^{\dagger} \cdot \omega \right) + V\, Q^{\dagger} \cdot \omega. \tag{14}$$

Parallelization Using basis functions with a compact support yields sparse matrices P and P'. However, Q and V that represent the polynomial are dense. Using the integrated polynomial therefore adds global communication overhead in the process of solving the ill-conditinoned system. The separated polynomial, on the other hand, solves a smaller, better conditioned system, resulting in less global communication.

As an effect of the black-box approach to coupling, we reuse the surface partitioning of the respective solver and therefore cannot guarantee optimal load balancing. In practice, however, the partitioning already provides a good clustering of the vertices. As a result, the matrix P and P' have are band-matrices and, thus, the corresponding systems are easy to solve in parallel.

3 RESULTS

In the remainder of this work, we present results that illustrate the trade-off between conditioning and accuracy for uniform and non-uniform meshes. We compare techniques such as adaptive basis functions and rescaled interpolants as well as integrated and separated computation of the polynomial.

The test function to be interpolated is a rescaled Gaussian

$$f(x) = \exp(-|x-3|^2) + 2 \tag{15}$$

Albeit this test function appears simplicistic, it is able to reproduce properties of the interpolation we experienced on real-world FSI simulations.

Figs. 1 and 2 show the resulting root mean squared error (RMSE) and the condition of the interpolation matrix C or P, respectively, using a Gaussian basis function. The shape parameter is set according to Eq. (6). While the interpolation using uniform meshes has a condition number of 8.50×10^{15} when reaching an error of 10×10^{-6} ($m = 17.71$), the non-uniform mesh results in a condition number of 2.57×10^{18} reaching the same error ($m = 13.73$). Using an adaptive basis function, the condition number is significantly lower, resulting in 7×10^{14} for the same level of accuracy as illustrated in Fig. 3.

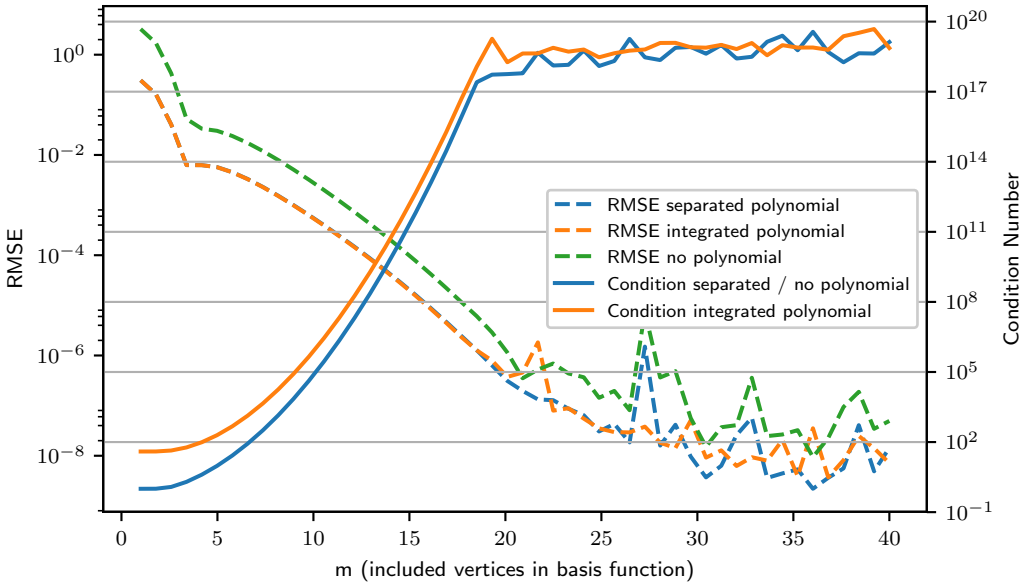

Figure 1: Condition number and root mean square error (RSME) for a uniform point distribution with 192 points.

Figs. 4 and 5 show accuracy and condition number for increasing mesh sizes. Since the shape parameter is computed based on the maximum mesh width, in most parts

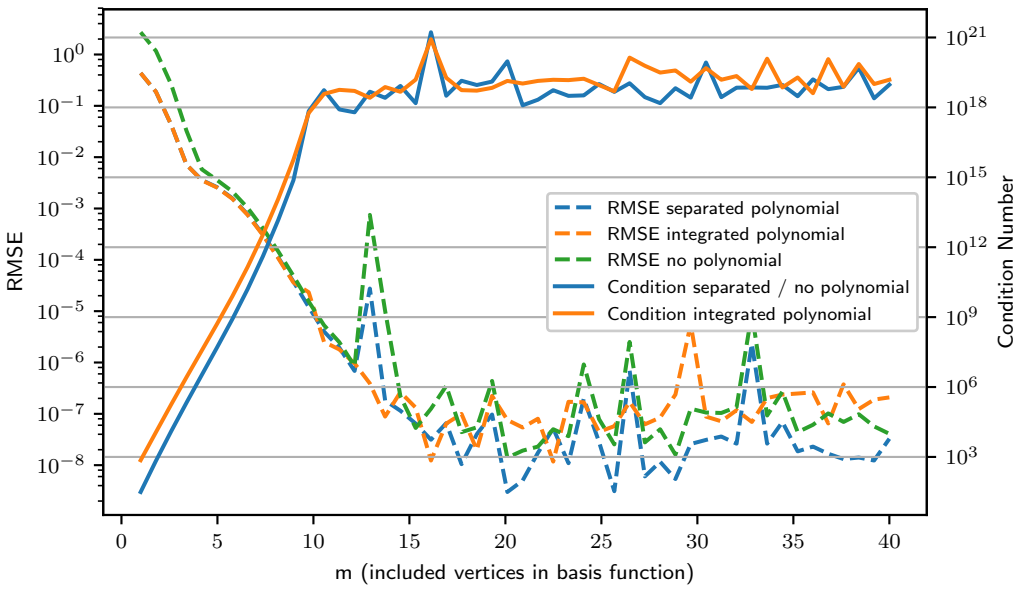

Figure 2: Condition number and root mean square error (RSME) for a Gauss-Chebyshev non-uniform point distribution, using 16 elements of 12th order, resulting in 192 points.

of the interval, the actual m is larger than the prescribed one for the Gauss-Chebyshev distribution, which explains the better accuracy. The Gauss-Chebyshev mesh yields a condition of 1.3×10^{18} to reach an error below 1×10^{-4} on a mesh of size 848. The uniform mesh, on the other hand, requires a higher value for m to reach a comparable accuracy, but the condition remained below 5×10^{9}. In contrast to the non-uniform mesh, condition number and accuracy curves flatten out, making m a suitable parameter to control accuracy and condition for uniform meshes of different sizes. Non-uniform meshes show an unstable behavior for larger mesh sizes, resulting in an smaller, but heavily oscillating error.

The rescaled basis functions method does not change the interpolation matrices themselves and, therefore, obviously results in identical condition numbers. With regard to accuracy, it performs about two orders of magnitude better. This gain in accuracy makes it possible to use a smaller value for m, which in turn improves the conditioning. Whereas, without rescaling, the interpolant supplemented by the polynomial performed much better than without polynomial, both methods are comparable when combined with rescaling.

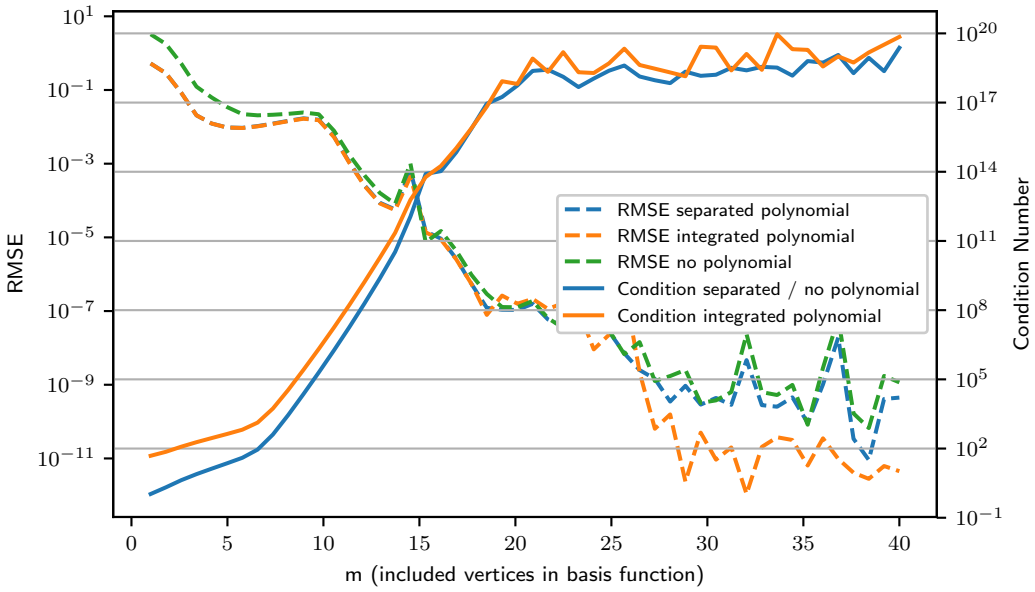

Figure 3: Condition number and root mean square error (RSME) for Gauss-Chebyshev non-uniform point distribution, using 16 elements of 12th order, resulting in 192 points using shape parameters adapted to the local mesh width.

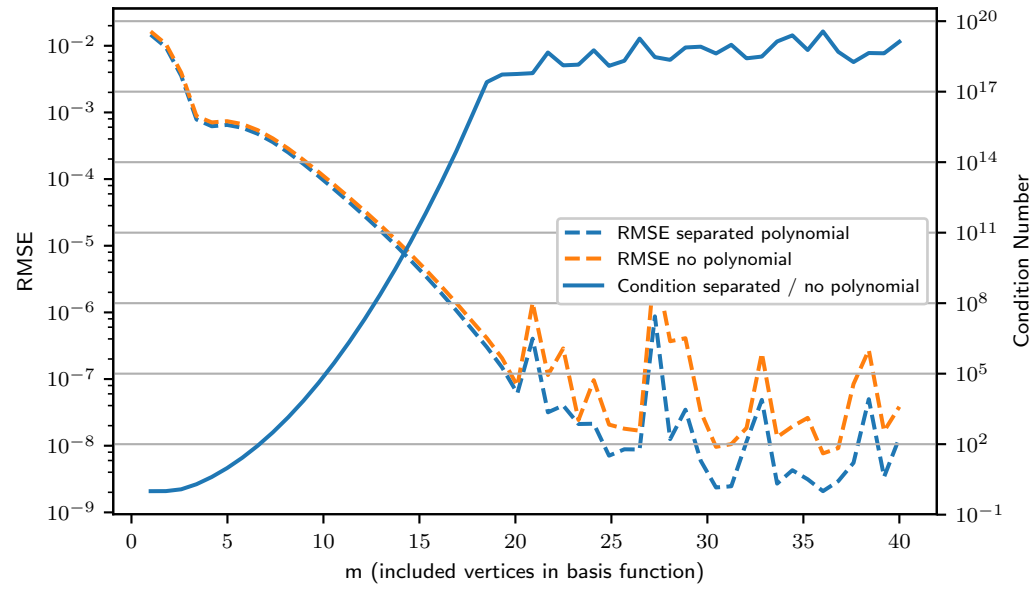

Figure 6: Condition number and root mean square error (RSME) for the rescaled interpolator and uniform point distribution using 192 points.

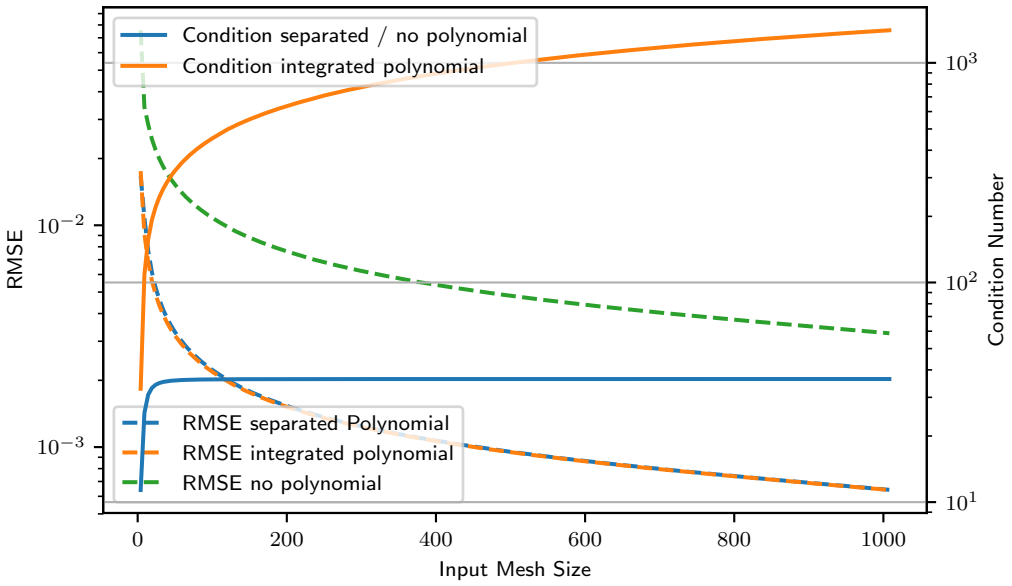

Figure 4: Condition number and root mean square error (RSME) for uniform point distribution, using Gaussian basis functions with width $m = 6$.

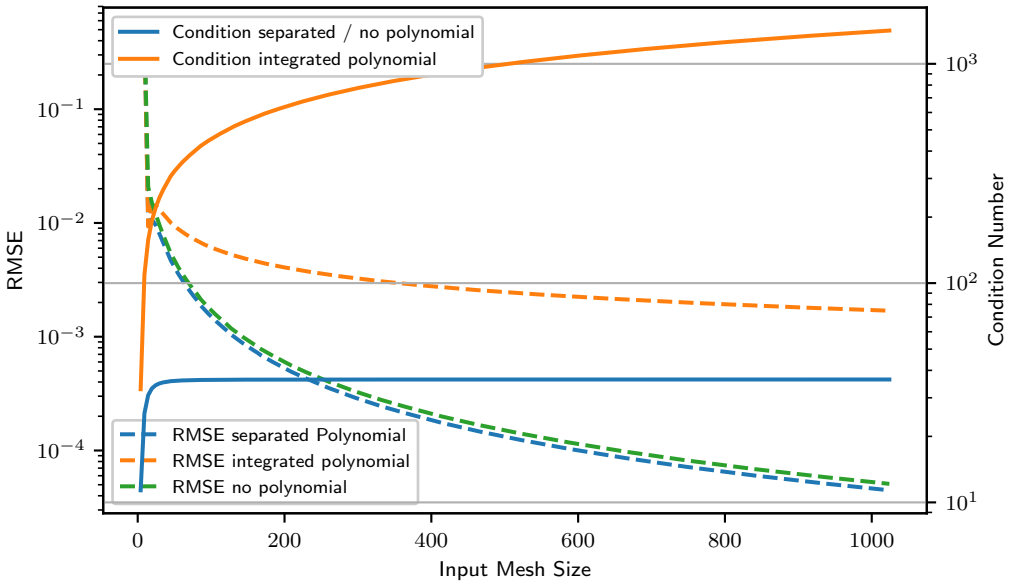

Figure 7: Condition number and root mean square error (RSME) for the rescaled interpolator and uniform point distribution, using a Gaussian basis function whith $m = 6$.

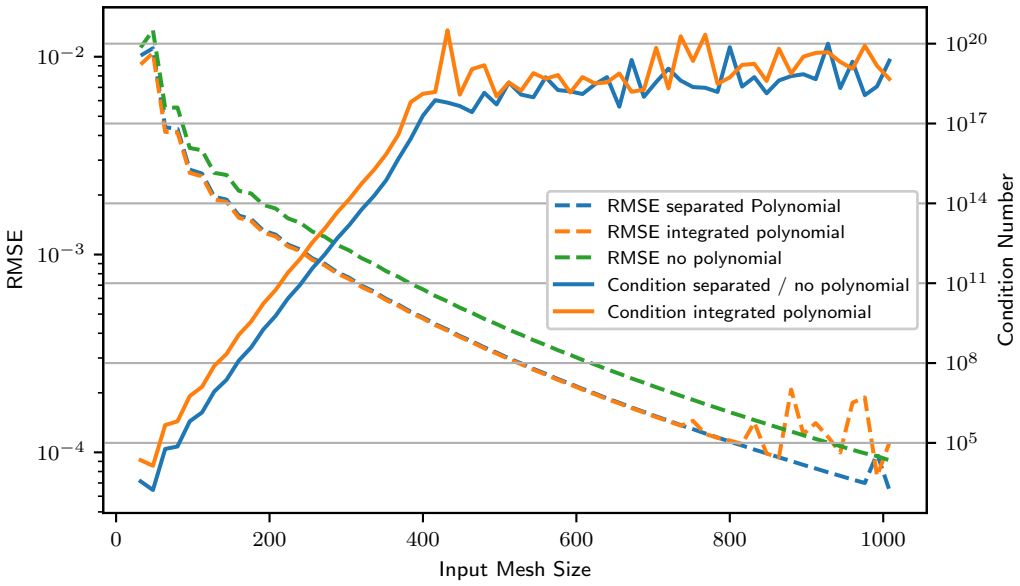

Figure 5: Condition number and root mean square error (RSME) for Gauss-Chebyshey non-uniform point distribution, using a fixed element size of 0.25 with varying degrees from 2 to 64 for basis functions with width $m = 6$ neighboring vertices (based on maximum mesh width).

4 CONCLUSIONS

We evaluated the conditioning of the interpolation system matrix, as well as the accuracy of the interpolation using radial basis functions on uniform and non-uniform meshes. For this paper, we focused on consistent interpolation using a Gaussian basis function. Multi Quadrics and Thin Plate Splines are other often mentioned basis function for FSI, but require a global support and therefore are detrimental to parallelization. Results show that interpolation by RBF is powerful, easy to implement and works well on uniformely scattered data. On uniform meshes, the width m of the basis functions is an effective parameter to control the accuracy-condition trade-off and is applicable to meshes of different sizes. This trade-off also appears when other basis functions, such as Multi-Quadrics or Thin Plate Splines are used. However, the interpolation performs less well on non-uniform meshes. The parameter m needs to be reduced on larger meshes to keep the condition reasonably low and to produce reliable numerical results. Adaptivity can ease the bad conditioning but requires the knowledge of local mesh density, which needs to be either supplied by the solver providing the mesh or computed by the interpolation algorithm. In addition to the computational overhead, this would require significant changes to the software stack to efficiently provide and use this information. Another problematic feature of RBF interpolation is oszillations at the edges of the interpolation interval, similiar to Runge's phenomenon for polynomial interpolation. This can in particular harm solvers

that - following the black-box coupling - rely on the interpolation results close to the edge of the mesh or even outside the source mesh (extrapolation). The rescaling approach also solves this issue in large parts.

The investigation in this paper were performed on one-dimensional data sets. Though experiences show similiar behavior for three-dimensional sets, it warrants further investigation with particular focus on the rescaled interpolation approach. At the current state of RBF interpolation, preCICE provides a powerful and efficient implementation, which still needs significant user experience to control the results. Future work will introduce automated methods to remove this burden from the user. Next steps include the implementation of the rescaling into the coupling software preCICE and tests on real-world multi-physics problem as well as performance optimization.

References

[1] Aukje de Boer, Alexander van Zuijlen, and Hester Bijl. "Comparison of conservative and consistent approaches for the coupling of non-matching meshes". In: *Computer Methods in Applied Mechanics and Engineering* 197.49-50 (Sept. 2008), pp. 4284–4297.

[2] Aukje de Boer, Alexander van Zuijlen, and Hester Bijl. "Review of coupling methods for non-matching meshes". In: *Computer Methods in Applied Mechanics and Engineering* 196.8 (Jan. 2007), pp. 1515–1525.

[3] Hans-Joachim Bungartz et al. "preCICE - A Fully Parallel Library for Multi-Physics Surface Coupling". In: *Computers and Fluids* (2016), pp. 1–9.

[4] Simone Deparis, Davide Forti, and Alfio Quarteroni. "A Rescaled Localized Radial Basis Function Interpolation on Non-Cartesian and Nonconforming Grids". In: *SIAM J. Sci. Comput.* 36.6 (2014), A2745–A2762.

[5] Gregory E. Fasshauer and Jack G. Zhang. "On choosing "optimal" shape parameters for RBF approximation". In: *Numerical Algorithms* 45.1-4 (2007), pp. 345–368.

[6] Gregory E. Fasshauer and Jack G. Zhang. "Preconditioning of Radial Basis Function Interpolation Systems via Accelerated Iterated Approximate Moving Least Squares Approximation". In: *Progress on Meshless Methods*. Ed. by A. J. M. Ferreira et al. Dordrecht: Springer Netherlands, 2009, pp. 57–75.

[7] Bernhard Gatzhammer. "Efficient and Flexible Partitioned Simulation of Fluid-Structure Interactions". PhD Thesis. Technische Universität München, 2014, p. 261.

[8] Shmuel Rippa. "An algorithm for selecting a good value for the parameter c in radial basis function interpolation". In: *Advances in Computational Mathematics* 11 (1999), pp. 193–210.

[9] Benjamin Uekermann. "Partitioned Fluid-Structure Interaction on Massively Parallel Systems". PhD thesis. Technical University of Munich, 2016.

[10] Qi Zhang, Yangzhang Zhao, and Jeremy Levesley. "Adaptive radial basis function interpolation using an error indicator". In: *Numerical Algorithms* (Jan. 2017).