ISSN: 1392-6292/ eISSN: 1648-3510

VILNIUS TECH
Vilnius Gediminas Technical University

MATHEMATICAL MODELLING and ANALYSIS

2026
Volume 31
Issue 1
Pages 130–148
https://doi.org/10.3846/mma.2026.22788

# Analytical formulas for polynomial coefficients in radial basis function interpolation

Vaclav Skala ✉

*Department of Computer Science and Engineering, Faculty of Applied Sciences, University of West Bohemia in Pilsen, Pilsen, Czech Republic* ROR

**Article History:**

- received November 28, 2024
- revised November 5, 2025
- accepted November 19, 2025

**Abstract.** Radial basis functions (RBF) are used in many areas, including interpolation and approximation, solution of partial differential equations, neural networks, and machine learning. RBFs are based on the sum of weighted kernel functions. Additional orthogonal polynomials are added for robustness, numerical stability, and computational efficiency improvement.

This contribution gives a new analytical formula specifying values of the polynomial coefficients used in RBF interpolation. The zero-degree polynomial coefficient is related to the sigmoid function used in RBF-neural networks (RBF-NN).

Unlike prior works where polynomial augmentation is only used to guarantee solvability, this paper provides explicit closed-form formulae for polynomial coefficients (with special focus on the zero-degree case). This new analytical treatment clarifies their role as global bias terms in both interpolation and RBF neural networks.

Expected applicability is in data interpolation and approximation, RBF-neural networks, scientific computing and PDE solutions, geostatistics & spatial interpolation, machine learning, and data fitting and signal processing.

**Keywords:** Radial basis functions; RBF; meshless methods; RBF interpolation; polynomial; RBF neural networks; RBF-NN; invariance; scattered spatio-temporal data.

**AMS Subject Classification:** 41A05; 65D10; 68U05; 68T07.

✉ Corresponding author. E-mail: skala@kiv.zcu.cz

## 1 Introduction

Interpolation and approximation are among the most commonly used operations in computational techniques. Various methods have been developed for data interpolation, typically requiring some form of data "ordering", such as structured, rectangular, or unstructured meshes. A typical example is the numerical solution of partial differential equations (PDEs), where derivatives are approximated by finite differences, predominantly using rectangular or hexagonal meshes.

However, in many engineering applications, data are not structured but generally scattered within a $d$-dimensional space-time space. In technical applica-

tions, scattered data is often handled through tessellation using $d$-dimensional Delaunay triangulation. However, for $d$-dimensional data interpolation, this approach is computationally expensive and often impractical.

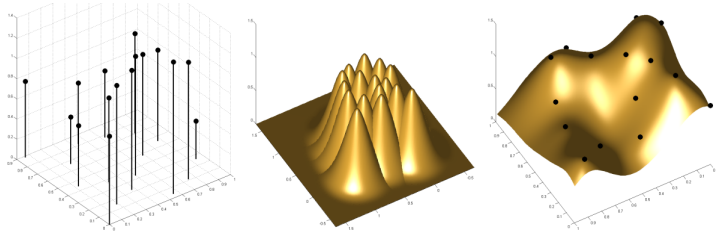


**Figure 1.** Given data values, collocation functions, final RBF interpolation; courtesy [23].

RBF interpolation is based on the *partition of unity* idea[1] used to construct a global interpolant by blending multiple local interpolants while ensuring smooth transitions between them. The key idea is to use a set of weighting functions that sum to one across the domain, allowing localized interpolation while maintaining global consistency (Figure 1). RBF interpolation is defined as:

$$h(\mathbf{x}) = \sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x} - \mathbf{x}_j\|) = \sum_{j=1}^{n} \lambda_j \varphi(r_j) \ , \ \mathbf{x} \in \mathbb{R}^d \ , \ \lambda_j \in \mathbb{R}^1, \tag{1.1}$$

where: $r_j$ represents the distance from a point $\mathbf{x}$ to the point $\mathbf{x}_j$, $\lambda_j \in \mathbb{R}^1$ represents the weight and $\varphi(\cdot)$ is the RBF kernel function used[2]. A polynomial $P_k(\mathbf{x})$ of degree $k$ is added to radial basis function (RBF) interpolation to ensure solvability and uniqueness of the interpolation system, especially when the RBF is conditionally positive definite (e.g., thin-plate splines). The polynomial compensates for the null space of the RBF kernel and enforces constraints that eliminate undesired affine components in the solution [17, 25]. It also improves approximation quality for low-frequency trends in the data.

$$h(\mathbf{x}) = \sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x} - \mathbf{x}_j\|) + P_k(\mathbf{x}),$$

where $P_k(\mathbf{x})$ is a polynomial of degree $k$. There are the main advantages:

- No data domain tessellation is needed.
- The formulation leads to a system of linear equations $\mathbf{Ax} = \mathbf{b}$.
- Scattered data is handled efficiently.
- Ensures the smoothness of the final interpolation.
- Simple use in radial basis function (RBF) interpolation, finite element methods (FEM), and meshfree methods.

[1] There is a special class of RBF-PUM with a strictly Partition of Unity property [4].
[2] In some cases, different scaling on axes can be used, leading to anisometric RBF kernels.

- If Compactly Supported RBF (CS-RBF) is used, it reduces computational cost compared to global interpolation methods.

RBF interpolation avoids a tessellation of the data domain and converts the problem to a solution of linear systems of equations [6].

In the case of RBF use, the dataset $\left\{(\mathbf{x}_i, h_i)_{i=1}^n\right\}$ establishes $n$ linear constraints, which formally results in a system of linear equations $\mathbf{Ax} = \mathbf{b}$:

$$h(\mathbf{x}_i) = \sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) = \sum_{j=1}^{n} \lambda_j \varphi(r_{ij}), \quad i = 1, \ldots, n, \tag{1.2}$$

where $\lambda_j$ are weights to be determined and $\varphi(*)$ is an RBF kernel function.[3] In some cases, the RBF-based methods are also sensitive to properly selecting the shape parameter $\alpha$. As the RBF matrices tend to be ill-conditioned in some cases, additional orthogonal polynomials are added to improve the robustness and precision of computation. However, additional polynomials lead to a loss of rotational and translational invariance.[4]

Radial basis functions (RBFs) are used in many areas, e.g., interpolation and approximation, machine learning and neural networks (RBF-NN), meshless methods in computational mathematics, image processing (registration, inpainting, and image registration), financial modeling, pattern recognition, and image analysis.

## 2 Radial Basis Functions

Many radial basis functions (RBFs) have been introduced recently [2, 6, 16]. RBF interpolation of scattered data involves solving a linear system of equations $\mathbf{Ax} = \mathbf{b}$ and consists of two significant steps:

1. The weights $\boldsymbol{\lambda}$ computation (Equation (1.2)).
2. Computation of the interpolated function $h(\mathbf{x})$ for the value $\mathbf{x}$ (Equation (1.1)).

### 2.1 RBF kernel functions

RBF interpolation is based on the weighted sum of RBF kernel functions, which can be categorized into two major groups:

- "Global" RBFs: These RBFs have a global impact and include functions:
  - Polyharmonic spline (PHS):
    $\varphi(r) = r^{2k-1}, \quad k = 1, 2, \ldots; \quad \varphi(r) = r^k \ln r, \quad k = 1, 2, \ldots;$
  - Thin-plate spline (TPS)[5] $\varphi(r) = r^{2k} \ln r, \quad k = 1, 2, \ldots;$

[3] Some RBFs are parameterized by a shape parameter $\alpha > 0$.

[4] In some cases, especially if the polynomial degree is higher and the data domain range is high, it might cause severe computational problems.

[5] In an implementation $\varphi(r) = r^2 \ln r^2$ should be used instead as $r^2 \ln r = \frac{1}{2} r^2 \ln r^2$ eliminates $\sqrt{*}$ operation and $\lambda$ coefficients will be doubled, only.

- Gaussian: $\varphi(r) = e^{-\alpha r^2}$;
- Multiquadric: $\varphi(r) = \sqrt{1+\alpha r^2}$;
- Inverse quadratic: $\varphi(r) = \frac{1}{1+\alpha r^2}$;
- Inverse multiquadratic: $\varphi(r) = \frac{1}{\sqrt{1+\alpha r^2}}$;

where $r \in (0, \infty)$; a shape parameter $\alpha > 0$ is used, i.e., $\alpha r$ instead of $r$, to modify the RBF kernel function influence. Global RBFs lead to a dense (full) matrix $\mathbf{A}$, sometimes ill-conditioned [14].

- "Local" RBFs (Compactly Supported RBFs or CS-RBFs): These RBFs have non-zero positive values only on the interval $(0, 1)$. CS-RBFs usually lead to a sparse matrix $\mathbf{A}$ and depend on the shape parameter $\alpha$.[6] CS-RBF examples are listed in Table 1.
- Other more complex functions have been defined (RBF-PUM), e.g. [4,16].

**Table 1.** Wendland's Compactly Supported RBF (CS-RBF) $(\cdot)_+$ indicates that the value of the expression is zero for $r \geq 1$; $\alpha r$ instead of $r$ is used to modify the RBF kernel function influence, $\alpha > 0$.

| ID | RBF | Function | ID | RBF | Function |
|---|---|---|---|---|---|
| 1 | $\varphi_{1,0}$ | $(1-r)_+$ | 2 | $\varphi_{1,1}$ | $(1-r)_+^3(3r+1)$ |
| 3 | $\varphi_{1,2}$ | $(1-r)_+^5(8r^2+5r+1)$ | 4 | $\varphi_{3,0}$ | $(1-r)_+^2$ |
| 5 | $\varphi_{3,1}$ | $(1-r)_+^4(4r+1)$ | 6 | $\varphi_{3,2}$ | $(1-r)_+^6(35r^2+18r+3)$ |
| 7 | $\varphi_{3,3}$ | $(1-r)_+^8(32r^3+25r^2+8r+3)$ | 8 | $\varphi_{5,0}$ | $(1-r)_+^3$ |
| 9 | $\varphi_{5,1}$ | $(1-r)_+^3(5r+1)$ | 10 | $\varphi_{5,2}$ | $(1-r)_+^7(16r^2+7r+1)$ |

### 2.2 Standard testing functions

Several testing functions are commonly used to evaluate interpolation and approximation precision [10]. The most commonly used functions $F_i(x, y)$ are listed in Table 2.

In some cases [5], an improper choice of the shape parameter $\alpha$ of the kernel function $\varphi(r)$ or polynomial degree significantly influences precision and RBF matrix conditionality [22]. Figure 2 presents the conditionality of the RBF matrix dependency on a shape parameter value and the number of points $n$.

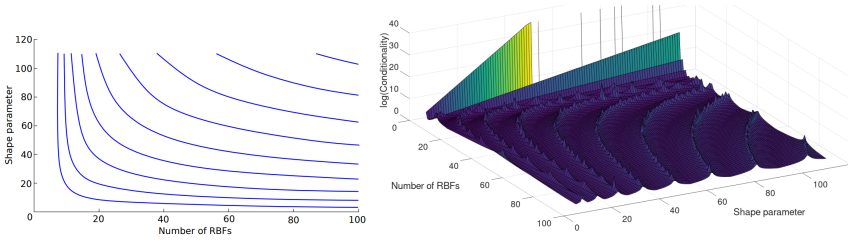
## 3 RBF Interpolation

Radial basis functions (RBF) are used in interpolation and approximation in the solution of partial differential equations and are widely used in various applications [1, 19]. RBF interpolation is based on the mutual distances between

[6] The shape parameter $\alpha$ value may be critical to interpolation precision, as it influences the interpolation errors, see [9, 13]; a deeper search for optimal shape parameter choice was analyzed in [3, 19].

**Table 2.** Typical testing functions.

$$
\begin{aligned}
F_1(x,y) &= \tfrac{3}{4}\, e^{-\frac{1}{4}((9x-2)^2+(9y-2)^2)} + \tfrac{3}{4}\, e^{-\frac{1}{49}(9x+1)^2-\frac{1}{10}(9y+1)^2},\\
&\quad +\tfrac{1}{2}\, e^{-\frac{1}{4}((9x-7)^2+(9y-3)^2)} - \tfrac{1}{5}\, e^{-(9x-4)^2-(9y-7)^2},\\
F_2(x,y) &= \tfrac{1}{9}\left(\tanh(9y-9x)+1\right), \quad F_3(x,y) = \tfrac{1}{6}\tfrac{1.25+\cos(4.5y)}{1+(3x-1)^2}\\
F_4(x,y) &= \tfrac{1}{3}e^{-\frac{81}{16}\left((x-\frac{1}{2})^2+(y-\frac{1}{2})^2\right)}, \quad F_5(x,y) = \tfrac{1}{3}e^{-\frac{81}{4}\left((x-\frac{1}{2})^2+(y-\frac{1}{2})^2\right)},\\
F_6(x,y) &= \tfrac{1}{9}\sqrt{64-81\left((x-\tfrac{1}{2})^2+(y-\tfrac{1}{2})^2\right)}-0.5\\
F_7(x,y) &= \sqrt{x^2+y^2}+0.2\\
H_{xx}(x,y) &= \tfrac{1}{3}e^{-\frac{81}{16}\left((x-\frac{1}{2})^2+(y-\frac{1}{2})^2+(z-\frac{1}{2})^2\right)}
\end{aligned}
$$




(a) Worst conditionality for Gauss RBF (b) Gaussian RGB matrix conditionality

**Figure 2.** Gaussian RGB matrix conditionality dependency on a shape parameter $\alpha$ and number of RBFs; courtesy [5].

points within the data domain $\Omega$ [6,7,10,11,20,24,25,26]. Due to the meshless representation, RBF can be used for meshless interpolation in $d$-dimensional space and approximation of time-varying scattered data, where a domain tessellation cannot be used.
RBF interpolation is defined as:

$$h(\mathbf{x}) = \sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x}-\mathbf{x}_j\|) = \sum_{j=1}^{n} \lambda_j \varphi(r_j)\ ,\ \mathbf{x} \in \mathbb{R}^d\ ,\ \lambda_j, h \in \mathbb{R}^1, \tag{3.1}$$

where: $r_j$ represents the distance from a point $\mathbf{x}$ to the point $\mathbf{x}_j$, $\lambda_j$ represents the weight and $\varphi(\cdot)$ is the RBF kernel function used. Since the parameter $r$ of the function $\varphi(r)$ is a distance between two points in $d$-dimensional space, the RBF interpolation is independent of dimension.

Let us consider the given dataset $\left\{(\mathbf{x}_i, h_i)_{i=1}^n\right\}$. Then, for each point $\mathbf{x}_i$, the interpolating function must take the value $h_i$. It leads to a system of linear equations:

$$h_i = h(\mathbf{x}_i) = \sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) = \sum_{j=1}^{n} \lambda_j \varphi(r_{ij}), \quad i = 1, \dots, n, \tag{3.2}$$

where: $\lambda_j$ represents unknown weights for each radial basis function, $n$ is the number of given points, and $\varphi(*)$ is the radial basis kernel function.
Equation (3.2) can be rewritten in the matrix form using $\varphi_{ij} = \varphi(r_{ij}) = \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|)$ as:

$$\begin{bmatrix} \varphi_{11} & \cdots & \varphi_{1j} & \cdots & \varphi_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \varphi_{i1} & \cdots & \varphi_{ij} & \cdots & \varphi_{in} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \varphi_{n1} & \cdots & \varphi_{nj} & \cdots & \varphi_{nn} \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_i \\ \vdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_i \\ \vdots \\ h_n \end{bmatrix}. \tag{3.3}$$

Or in a more compact form as:

$$\mathbf{A}\boldsymbol{\lambda} = \mathbf{h}\ ,\ \mathbf{A} \in \mathbb{R}^{n\times n},\ \mathbf{h}, \boldsymbol{\lambda} \in \mathbb{R}^n.$$

Then, the interpolated value $h(\mathbf{x})$ at any point $\mathbf{x}$ is computed using Equation (3.1) as described above.

However, this simple formulation can lead to the ill-conditionality of the linear system of equations in some cases [5, 6], especially with a high number of points $n$ and a large range of data [22], special solutions of the linear system have to be used, e.g., [18].

Therefore, additional orthogonal functions are to be added to ensure better numerical properties and positive definiteness of the system [6, 8, 22].

In the case of an additional polynomial $P_k(\mathbf{x}_i)$ of degree $k$ is used, Equation (3.1) is modified to:

$$h(\mathbf{x}_i) = \sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) + P_k(\mathbf{x}_i), \quad i = 1, \dots, n,$$

and additional orthogonal conditions have to be added, e.g., in the case of a bilinear polynomial of degree one $P_1(x, y)$:

$$P_1(x, y) = a_0 + a_1 x + a_2 y + a_3 xy$$

and the following additional orthogonal conditions are added:[7]

$$\sum_{j=1}^{n} \lambda_j = 0, \quad \sum_{j=1}^{n} \lambda_j x_j = 0, \quad \sum_{j=1}^{n} \lambda_j y_j = 0, \quad \sum_{j=1}^{n} \lambda_j x_j y_j = 0. \tag{3.4}$$

[7] Coefficients $a_0, a_1, a_2$ represent an approximation by a plane, while the additional term $a_3 xy$ enables a bilinear approximation of $h(\mathbf{x}) = f(x, y)$.

The coefficients $a_0, a_1, \ldots$ act as *Lagrange multipliers* that enforce the orthogonality condition:

$$\sum_{j=1}^{n} \lambda_j p(\mathbf{x}_j) = 0 \quad \text{for all } p \in \Pi_k,$$

where $\Pi_k$ is the space of polynomials of total degree less than or equal to $k$. This ensures that the RBF interpolant is uniquely defined and orthogonal to the null space associated with the kernel function.

It results in a system of linear equations:

$$\begin{bmatrix} \varphi_{11} & \ldots & \varphi_{1n} & 1 & x_1 & y_1 & x_1y_1 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varphi_{n1} & \ldots & \varphi_{nn} & 1 & x_n & y_n & x_ny_n \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ x_1 & \ldots & x_n & 0 & 0 & 0 & 0 \\ y_1 & \ldots & y_n & 0 & 0 & 0 & 0 \\ x_1y_1 & \ldots & x_ny_n & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} h_1 \\ \vdots \\ h_n \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

The conditions in Equation (3.4) lead to a compact formulation of RBF interpolation:

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{h} \\ \mathbf{0} \end{bmatrix} \ , \ \mathbf{P} \in \mathbb{R}^{n \times 4} \ , \ \boldsymbol{\lambda} \in \mathbb{R}^n \ , \ \mathbf{a} \in \mathbb{R}^4,$$

where the matrix $\mathbf{P}$ represents the polynomial, $\boldsymbol{\lambda}$ is the vector of the RBF weights, the vector $\mathbf{a}$ contains the resulting polynomial coefficients, and $\mathbf{h}$ are the given values $h_i$ at the given points $\mathbf{x}_i \in \Omega$. The matrix $\mathbf{P}^T$ represents the additional orthogonal conditions defined in Equation (3.4).

$$\mathbf{P} = \begin{bmatrix} 1 & x_1 & y_1 & x_1y_1 \\ 1 & x_2 & y_2 & x_2y_2 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & y_n & x_ny_n \end{bmatrix}.$$

The bottom block row imposes the constraint:

$$\mathbf{P}^T\boldsymbol{\lambda} = \mathbf{0},$$

which ensures the uniqueness of the interpolant by orthogonality to the polynomial space.

It is worth noting that the matrix size is nearly independent of the data domain dimension and has the size of $(p \times p)$, where $p = n + k + 1$ in the case of the $k$-degree polynomial.
However, when using the polynomial $P_k(\mathbf{x})$:

- RBF interpolation is not invariant to rotation and translation except when a zero-degree polynomial $P_0(\mathbf{x}) = a_0$ is used.

- Interpolation, represented by $\boldsymbol{\lambda}$ and $\mathbf{a}$, depends on the physical units used for the vector $\mathbf{x}$, i.e. the points' coordinates.
- Use of a $k$-degree polynomial, $k > 0$, might be counter-productive in cases of an extensive range of the domain data [22].

The RBF interpolation results in a linear system of equations $\mathbf{Ax} = \mathbf{b}$. It is a significant advantage of the RBF methods that sophisticated numerical methods have been developed, and no tessellation is required.[8]

### 3.1 Normalized RBF

Normalized RBF (N-RBF) is a modification of the standard RBF used in RBF neural network-related applications [21]. It is given as:

$$h(\mathbf{x}) = \frac{\sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x} - \mathbf{x}_j\|)}{\sum_{j=1}^{n} \varphi(\|\mathbf{x} - \mathbf{x}_j\|)} = \frac{\sum_{j=1}^{n} \lambda_j \varphi(r_j)}{\sum_{j=1}^{n} \varphi(r_j)}, \tag{3.5}$$

where: $r_j$ is the distance from a point $\mathbf{x}$ to the point $\mathbf{x}_j$.
It can be seen that denominator values in Equation (3.5) can be zero, or close to zero, i.e., $\sum_{j=1}^{n} \varphi(\|\mathbf{x} - \mathbf{x}_j\|) \to 0$, for some values $\mathbf{x}$, which would lead to instability as $h(\mathbf{x}) \to \infty$. However, the N-RBFs are used in the RBF neural network applications [21] as projective linearity is kept.

### 3.2 Squared normalized RBF

Some RBF functions $\varphi(r)$ used for interpolation and approximation are not strictly positive, such as $r^2 \ln(r)$(Thin-Plate Spline-TPS), which is negative on the interval $(0, 1)$. The Euclidean norm should be used for more robust computation in such cases.[9]

The Squared Normalized RBF (SN-RBF) is defined as follows:

$$h(\mathbf{x}) = \frac{\sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x} - \mathbf{x}_j\|)}{\sqrt{\sum_{j=1}^{n} \varphi^2(\|\mathbf{x} - \mathbf{x}_j\|)}} = \frac{\sum_{j=1}^{n} \lambda_j \varphi(r_j)}{\sqrt{\sum_{j=1}^{n} \varphi^2(r_j)}}, \tag{3.6}$$

where, $r_j = \|\mathbf{x} - \mathbf{x}_j\|$ is the distance from a point $\mathbf{x}$ to the point $\mathbf{x}_j$.

It is essentially the Euclidean normalization of each row of the matrix $\mathbf{A}$ in Equation (3.3), which leads to slightly better numerical conditionality of the linear system of equations [5]. In the case of interpolation, Equation (3.6) must be valid for all given points $\mathbf{x}_i \in \Omega$:

$$h(\mathbf{x}_i) = \frac{\sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|)}{\sqrt{\sum_{j=1}^{n} \varphi^2(\|\mathbf{x}_i - \mathbf{x}_j\|)}} = \frac{\sum_{j=1}^{n} \lambda_j \varphi(r_{ij})}{\sqrt{\sum_{j=1}^{n} \varphi^2(r_{ij})}}, \quad i = 1, \ldots, n \tag{3.7}$$

[8] Note that in the case of spatio-temporal varying scattered data also in time, data tessellation is not possible.

[9] It should be noted that $r^2 \log(r) = \frac{1}{2} r^2 \log(r^2)$ is to be used as no $\sqrt{r^2}$ computation is needed; only the values of the weights $\lambda_j$ are doubled.

and it leads to:

$$\sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) = h(\mathbf{x}_i) \sqrt{\sum_{j=1}^{n} \varphi^2(\|\mathbf{x}_i - \mathbf{x}_j\|)}, \quad i = 1, \ldots, n. \tag{3.8}$$

In the case of the RBF interpolation with an added polynomial $P_k(\mathbf{x})$ of degree $k$, Equation (3.7) becomes:

$$h(\mathbf{x}) = \frac{\sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x} - \mathbf{x}_j\|)}{\sqrt{\sum_{j=1}^{n} \varphi^2(\|\mathbf{x} - \mathbf{x}_j\|)}} + P_k(\mathbf{x}). \tag{3.9}$$

The additional orthogonal conditions will be added, as discussed in Equation (3.4). The polynomial $P_k(\mathbf{x})$ improves the conditionality of the RBF matrix and provides a rough approximation of the given data.

The SN-RBF in Equation (3.9) can be rewritten as:

$$h(\mathbf{x}_i) = \frac{\sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|)}{\sqrt{\sum_{j=1}^{n} \varphi^2(\|\mathbf{x}_i - \mathbf{x}_j\|)}} + P_k(\mathbf{x}_i) = \frac{\sum_{j=1}^{n} \lambda_j \varphi(r_{ij})}{\sqrt{\sum_{j=1}^{n} \varphi^2(r_{ij})}} + P_k(\mathbf{x}_i), \; i = 1, \ldots, n, \tag{3.10}$$

where: $r_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$. It is important to note that this approach replaces $O(n^2)$ division operations with $O(n)$ multiplications. The given values $h(\mathbf{x}_i)$ are just multiplied by the values $q_i = \sqrt{\sum_{j=1}^{n} \varphi^2(r_{ij})}$.

Now, Equation (3.10) can be modified similarly as to Equation (3.8) to:

$$\sum_{j=1}^{n} \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{x}_j\|) + P_k(\mathbf{x}_i) \left( \sum_{j=1}^{n} \varphi^2(\|\mathbf{x}_i - \mathbf{x}_j\|) \right)^{1/2}$$
$$= h(\mathbf{x}_i) \sqrt{\sum_{j=1}^{n} \varphi^2(\|\mathbf{x}_i - \mathbf{x}_j\|)} \quad i = 1, \ldots, n. \tag{3.11}$$

As $q_i$ is a constant for the $i^{th}$ row ($i = 1, \ldots, n$), Equation (3.11) is simplified to:

$$\sum_{j=1}^{n} \lambda_j \varphi \|\mathbf{x}_i - \mathbf{x}_j\|) + q_i \; P_k(\mathbf{x}_i) = q_i \; h(\mathbf{x}_i) \quad i = 1, \ldots, n, \tag{3.12}$$

where: $q_i = \sqrt{\sum_{j=1}^{n} \varphi^2(\|\mathbf{x}_i - \mathbf{x}_j\|)}$.

It leads to the system of linear equations for the SN-RBF interpolation:

$$\begin{bmatrix} \varphi_{11} & \ldots & \varphi_{1n} & q_1 & q_1 x_1 & q_1 y_1 & q_1 x_1 y_1 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varphi_{n1} & \ldots & \varphi_{nn} & q_n & q_n x_n & q_n y_n & q_n x_n y_n \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ x_1 & \ldots & x_n & 0 & 0 & 0 & 0 \\ y_1 & \ldots & y_n & 0 & 0 & 0 & 0 \\ x_1 y_1 & \ldots & x_n y_n & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \\ a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} q_1 h_1 \\ \vdots \\ q_n h_n \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

or more compactly as:

$$\begin{bmatrix} \mathbf{A} & \mathbf{QP} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{Qh} \\ \mathbf{0} \end{bmatrix},$$

where: $\mathbf{Q} = diag[q_1, \ldots, q_n]$ is a diagonal matrix and $q_i > 0$,[10] $i = 1, \ldots, n$ in the case of SN-RBFs.

It should be noted that in the approximation case, i.e., the matrix $\mathbf{A}$ is $(n \times m), n > m$, the polynomial part has to be handled differently, and the Least Squares Error method (LSE) cannot be used directly [15].

The polynomial $P_k(\mathbf{x})$ of degree $k$ used in RBF generally leads to better conditionality and computation robustness.

In the following sections, explicit formulae of additional polynomial coefficients will be derived, including the special case of a zero-degree polynomial related to the "bias" term in RBF-NN.

## 4 RBF polynomial coefficients

In RBF interpolation, the polynomial term can be viewed as capturing the broad, low-frequency trend of the data (a "coarse" approximation). At the same time, the radial basis functions provide localized corrections or "fine-tuning" to achieve an accurate overall interpolation.

The coefficients of the polynomial $P_k(\mathbf{x})$ **do not** represent statistical summaries of the data, such as the mean or the median. Instead, they arise as part of the solution to the saddle-point (or Karush-Kuhn-Tucker) system that ensures the solvability and uniqueness of the interpolation problem, especially when $\varphi(*)$ is a *conditionally positive definite* function. They are Lagrange multipliers that arise from enforcing the solvability conditions of the RBF system, particularly when using conditionally positive definite kernels (like polyharmonic splines).

- *Algebraically*, these coefficients enforce that the RBF part lies in the orthogonal complement of the space of polynomials of degree $\leq k$, ensuring a unique solution.
- *Geometrically*, they determine the global trend or bias in the interpolation, e.g., $a_0$ can be interpreted as the vertical shift (offset). In contrast, the other coefficients model linear or higher-order trends, i.e., represent tilt or slope, and higher-order terms capture curvature, etc.

In the following, the dependency of the coefficients $\mathbf{a}$ of the polynomial $P_k(\mathbf{x})$ on the distribution of points $\mathbf{x}$ in the data domain $\Omega$ and given interpolated values $\mathbf{h}$ will be analyzed.

[10] It is expected that $\mathbf{x}_i \neq \mathbf{x}_j, \ \forall i \neq j, \quad i, j = 1, \ldots n.$

### 4.1 RBF polynomial - general case

Let us consider RBF with a polynomial defined by the block matrix system:

$$\begin{bmatrix} \mathbf{A} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{h} \\ \mathbf{0} \end{bmatrix} \ , \ \mathbf{M}\,\boldsymbol{\xi} = \boldsymbol{\eta}, \tag{4.1}$$

where $\mathbf{A}$ and $\mathbf{M}$ are symmetrical matrices and the matrix $\mathbf{A}$ is non-singular, i.e., $\mathtt{DET}(\mathbf{A}) \neq 0$.[11] As Equation (4.1) gives the two equations:

$$\mathbf{A}\boldsymbol{\lambda} + \mathbf{P}\mathbf{a} = \mathbf{h}, \tag{4.2}$$

$$\mathbf{P}^T\boldsymbol{\lambda} = \mathbf{0}, \tag{4.3}$$

then the $\boldsymbol{\lambda}$ coefficients can be expressed, using Equation (4.2), as:

$$\boldsymbol{\lambda} = \mathbf{A}^{-1}(\mathbf{h} - \mathbf{P}\mathbf{a}). \tag{4.4}$$

Substituting Equation (4.4) into Equation (4.3):

$$\mathbf{P}^T\boldsymbol{\lambda} = \mathbf{P}^T\mathbf{A}^{-1}(\mathbf{h} - \mathbf{P}\mathbf{a}) = \mathbf{0}, \text{ i.e., } \mathbf{P}^T\mathbf{A}^{-1}\mathbf{h} - \mathbf{P}^T\mathbf{A}^{-1}\mathbf{P}\mathbf{a} = \mathbf{0},$$

the coefficients $\mathbf{a}$ of the polynomial $P_k(\mathbf{x})$ are given as:

$$\mathbf{a} = \left(\mathbf{P}^T\mathbf{A}^{-1}\mathbf{P}\right)^{-1}\mathbf{P}^T\mathbf{A}^{-1}\mathbf{h}, \tag{4.5}$$

where $\mathbf{P}^T\mathbf{A}^{-1}\mathbf{P}$ is the Schur's complement $\mathbf{M}\backslash\mathbf{0}$ of the block sub-matrix $\mathbf{A}$ of the matrix $\mathbf{M}$. The formula Equation (4.5) expresses the polynomial coefficient $a_0$ directly from the known matrices $\mathbf{A}$, $\mathbf{P}$, and the vector of the given values $\mathbf{h}$.

### 4.2 SN-RBF polynomial - general case

The Squared-normalized RBF (SN-RBF) is described in the block matrix form as:

$$\begin{bmatrix} \mathbf{A} & \mathbf{Q}\mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{Q}\mathbf{h} \\ \mathbf{0} \end{bmatrix}, \tag{4.6}$$

where $\mathbf{Q} = diag[q_1, \ldots, q_n]$ is a diagonal matrix, $q_i = \sqrt{\sum_{j=1}^{n} \varphi^2(\|\mathbf{x}_i - \mathbf{x}_j\|)}$, see Equation (3.12). Then Equation (4.6) can be rewritten as:

$$\mathbf{A}\boldsymbol{\lambda} + \mathbf{Q}\mathbf{P}\mathbf{a} = \mathbf{Q}\mathbf{h}, \tag{4.7}$$

$$\mathbf{P}^T\boldsymbol{\lambda} = \mathbf{0}. \tag{4.8}$$

Then $\boldsymbol{\lambda}$ can be expressed from Equation (4.7) as:

$$\boldsymbol{\lambda} = \mathbf{A}^{-1}\left(\mathbf{Q}\mathbf{h} - \mathbf{Q}\mathbf{P}\mathbf{a}\right) = \mathbf{A}^{-1}\mathbf{Q}(\mathbf{h} - \mathbf{P}\mathbf{a}).$$

[11] For some kernel functions $\varphi(*)$, the matrix $\mathbf{A}$ is a positive definite matrix.

Substituting into Equation (4.8):

$$\mathbf{P}^T\boldsymbol{\lambda} = \mathbf{P}^T\mathbf{A}^{-1}\mathbf{Q}(\mathbf{h} - \mathbf{P}\mathbf{a}) = \mathbf{0}.$$

Rewriting:

$$\mathbf{P}^T\mathbf{A}^{-1}\mathbf{Q}\mathbf{h} = \mathbf{P}^T\mathbf{A}^{-1}\mathbf{Q}\mathbf{P}\mathbf{a}.$$

Therefore, the polynomial coefficients $\mathbf{a}$ can be expressed as:

$$\mathbf{a} = \left(\mathbf{P}^T\mathbf{A}^{-1}\mathbf{Q}\mathbf{P}\right)^{-1}\mathbf{P}^T\mathbf{A}^{-1}\mathbf{Q}\mathbf{h}. \tag{4.9}$$

### 4.3 Zero-order polynomial

Using a polynomial $P_k(\mathbf{x})$ of degree $k$ leads to a loss of invariance against rotation and translation, which might not be acceptable in some applications. However, using a zero-degree polynomial $P_0(\mathbf{x}) = a_0$ can be interpreted as the vertical shift (offset), while the other coefficients model linear or higher-order trends.

Now, the above-derived formulae for a general polynomial $P_k(\mathbf{x})$ can be specified for the zero-degree polynomial.

#### 4.3.1 RBF interpolation

In the case of the zero-order polynomial, i.e., $P_0(\mathbf{x}) = a_0$, the coefficient $a_0$ can be asily expressed as:

$$a_0 = \left(\mathbf{1}^T\mathbf{A}^{-1}\mathbf{1}\right)^{-1}\mathbf{1}^T\mathbf{A}^{-1}\mathbf{h} = \frac{\mathbf{1}^T\mathbf{A}^{-1}\mathbf{h}}{\mathbf{1}^T\mathbf{A}^{-1}\mathbf{1}}, \tag{4.10}$$

where $\mathbf{1}^T = [1, \ldots, 1]^T$ is a column vector of "1" and Equation (4.10) can be rewritten as

$$a_0 = \frac{\sum_{i,j=1}^{n}(\mathbf{A}^{-1})_{ij}h_j}{\sum_{i,j=1}^{n}(\mathbf{A}^{-1})_{ij}} = \frac{\sum_{j=1}^{n}h_j\sum_{i=1}^{n}\alpha_{ij}}{\sum_{j=1}^{n}\sum_{i=1}^{n}\alpha_{ij}},$$

where $\alpha_{ij}$ are elements of the inverse matrix $\mathbf{A}^{-1}$.

#### 4.3.2 SN-RBF interpolation

It is easy to prove that in the case of the SN-RBF, see Equations (4.9) and (3.10), the formula for the zero-degree polynomial coefficient $a_0$ has the form:

$$a_0 = \left(\mathbf{1}^T\mathbf{A}^{-1}\mathbf{Q}\ \mathbf{1}\right)^{-1}\mathbf{1}^T\mathbf{A}^{-1}\mathbf{Q}\mathbf{h} = \frac{\mathbf{1}^T\mathbf{A}^{-1}\mathbf{Q}\mathbf{h}}{\mathbf{1}^T\mathbf{A}^{-1}\mathbf{Q}\ \mathbf{1}},$$

$$a_0 = \frac{\sum_{j=1}^{n}h_jq_j\sum_{i=1}^{n}\alpha_{ij}}{\sum_{j=1}^{n}(q_j\sum_{i=1}^{n}\alpha_{ij})},$$

where $\mathbf{Q} = diag[q_1, \ldots, q_n]$ and $q_i = \sqrt{\sum_{j=1}^{n}\varphi^2(\|\mathbf{x}_i - \mathbf{x}_j\|)}$.

### 4.4 Optimized computation of the polynomial coefficients

Let us consider the SN-RBF system:

$$\begin{bmatrix} \mathbf{A} & \mathbf{QP} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{Qh} \\ \mathbf{0} \end{bmatrix}.$$

Let us define

$$\mathbf{z} := \mathbf{Qh} \ , \ \mathbf{B} := \mathbf{QP}.$$

To compute the polynomial coefficients $\mathbf{a}$ efficiently, the following steps should be used:

1. Solve the linear system $\mathbf{AY} = \mathbf{B}$, i.e., compute $\mathbf{Y} := \mathbf{A}^{-1}\mathbf{B}$.
2. Solve the linear system $\mathbf{Ay} = \mathbf{z}$, i.e., compute $\mathbf{y} := \mathbf{A}^{-1}\mathbf{z}$.
3. Then, compute: $\mathbf{a} = \left(\mathbf{P}^T\mathbf{Y}\right)^{-1}\left(\mathbf{P}^T\mathbf{y}\right)$.

This formulation avoids explicit matrix inversion and leverages the system's structure for improved numerical stability and performance.

## 5 RBF approximation with polynomial term

Let us consider the problem of approximating data values $h_i$ given at $n$ data points $\mathbf{x}_i \in \mathbb{R}^d$ using a radial basis function (RBF) interpolant augmented with a polynomial of degree $k$. The approximant takes the form:

$$s(\mathbf{x}) = \sum_{j=1}^{m} \lambda_j \varphi(\|\mathbf{x} - \boldsymbol{\xi}_j\|) + P_k(\mathbf{x}),$$

where: $\boldsymbol{\xi}_j$ are $m$ chosen center points ($knots$)[12] in $\mathbb{R}^d$, $\varphi$ is a radial basis function (e.g., Gaussian, multiquadric, etc.), $P_k(\mathbf{x})$ is a polynomial of total degree $k$.

### 5.1 Formulation

Approximation leads to an **overdetermined system** with $n$ equations and $m + L$ unknowns, where $L = \dim \Pi_k$, the number of monomials in $P_k$.

Then the approximation system becomes:

$$\mathbf{A}\boldsymbol{\lambda} + \mathbf{Pa} = \mathbf{h}, \quad \mathbf{h} \in \mathbb{R}^n,$$

where: $\mathbf{A} \in \mathbb{R}^{n\times m}$: matrix with entries $A_{ij} = \varphi(\|\mathbf{x}_i - \boldsymbol{\xi}_j\|)$, $\mathbf{P} \in \mathbb{R}^{n\times L}$: matrix with rows containing evaluations of all monomials of degree $\leq k$ at $\mathbf{x}_i$, $\boldsymbol{\lambda} \in \mathbb{R}^m$: RBF coefficients, $\mathbf{a} \in \mathbb{R}^L$: polynomial coefficients.

[12] Note that the *knots* points might differ from the given points $\mathbf{x}$; sometimes called points of importance, e.g., extreme points, inflection points, etc.

This can be solved in the *least-squares sense*. Defining the residual:

$$\mathbf{r} = \mathbf{A}\boldsymbol{\lambda} + \mathbf{P}\mathbf{a} - \mathbf{h}.$$

We want to minimize the squared error:

$$\|\mathbf{r}\|_2^2 = \|\mathbf{A}\boldsymbol{\lambda} + \mathbf{P}\mathbf{a} - \mathbf{h}\|_2^2.$$

To solve this constrained least-squares problem, the **normal equations** are to be set:

$$\begin{bmatrix} \mathbf{A}^T \\ \mathbf{P}^T \end{bmatrix} (\mathbf{A}\boldsymbol{\lambda} + \mathbf{P}\mathbf{a} - \mathbf{h}) = \mathbf{0},$$

which leads to the system:

$$\begin{bmatrix} \mathbf{A}^T\mathbf{A} & \mathbf{A}^T\mathbf{P} \\ \mathbf{P}^T\mathbf{A} & \mathbf{P}^T\mathbf{P} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{A}^T\mathbf{h} \\ \mathbf{P}^T\mathbf{h} \end{bmatrix}.$$

This $(m+L)\times(m+L)$ symmetric system can be solved using standard methods (e.g., Cholesky, if positive definite).

**Remarks:**

- Using a low-degree polynomial $P_k$ helps control global behavior and avoids ill-conditioning for certain RBFs.
- Choosing knot points $\boldsymbol{\xi}_j$ carefully (e.g., using clustering or low-discrepancy sampling) improves accuracy.

### 5.2 Explicit evaluation of polynomial coefficients

The approximation of given $n$ data points $\{\mathbf{x}_i, h_i\}_{i=1}^n$ and $m$ center points (*knots*) $\{\mathbf{c}_j\}_{j=1}^m$ leads to:

$$h(\mathbf{x}_i) \approx \sum_{j=1}^{m} \lambda_j \varphi(\|\mathbf{x}_i - \mathbf{c}_j\|) + \sum_{k=0}^{L-1} a_k p_{ik}(\mathbf{x}_i),$$

where: $\mathbf{A} \in \mathbb{R}^{n\times m}$, where $\mathbf{A}_{ij} = \varphi(\|\mathbf{x}_i - \mathbf{c}_j\|)$, $\mathbf{P} \in \mathbb{R}^{n\times L}$, where $\mathbf{P}_{ik} = p_k(\mathbf{x}_i)$, $\boldsymbol{\lambda} \in \mathbb{R}^m$: RBF weights, $\mathbf{a} \in \mathbb{R}^L$: polynomial coefficients, $\mathbf{h} \in \mathbb{R}^n$: given data vector.

Then the least squares error is minimized by solving:

$$\min_{\boldsymbol{\lambda},\mathbf{a}} \|\mathbf{A}\boldsymbol{\lambda} + \mathbf{P}\mathbf{a} - \mathbf{h}\|^2.$$

This leads to the standard equations:

$$\begin{bmatrix} \mathbf{A}^T\mathbf{A} & \mathbf{A}^T\mathbf{P} \\ \mathbf{P}^T\mathbf{A} & \mathbf{P}^T\mathbf{P} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} \mathbf{A}^T\mathbf{h} \\ \mathbf{P}^T\mathbf{h} \end{bmatrix}.$$

Assuming $\mathbf{A}^T\mathbf{A}$ is invertible, the first block row can be solved:

$$\boldsymbol{\lambda} = (\mathbf{A}^T\mathbf{A})^{-1}\left(\mathbf{A}^T\mathbf{h} - \mathbf{A}^T\mathbf{P}\mathbf{a}\right).$$

Substitute into the second block row:

$$\begin{aligned}&\mathbf{P}^T\mathbf{A}\boldsymbol{\lambda} + \mathbf{P}^T\mathbf{P}\mathbf{a} = \mathbf{P}^T\mathbf{h},\\ &\mathbf{P}^T\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\left(\mathbf{A}^T\mathbf{h} - \mathbf{A}^T\mathbf{P}\mathbf{a}\right) + \mathbf{P}^T\mathbf{P}\mathbf{a} = \mathbf{P}^T\mathbf{h}.\end{aligned}$$

Distribute terms:

$$\mathbf{P}^T\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{h} - \mathbf{P}^T\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{P}\mathbf{a} + \mathbf{P}^T\mathbf{P}\mathbf{a} = \mathbf{P}^T\mathbf{h}$$

and rearranging gives:

$$\left(\mathbf{P}^T\mathbf{P} - \mathbf{P}^T\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{P}\right)\mathbf{a} = \mathbf{P}^T\mathbf{h} - \mathbf{P}^T\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{h}.$$

Finally, the explicit formula for $\mathbf{a}$ is given as:

$$\mathbf{a} = \left(\mathbf{P}^T\mathbf{P} - \mathbf{P}^T\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{P}\right)^{-1}\left(\mathbf{P}^T\mathbf{h} - \mathbf{P}^T\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{h}\right).$$

Observing that the matrix:

$$\mathbf{H}_A = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$$

is the orthogonal projector onto the column space of $\mathbf{A}$.

Therefore, the expression can be simplified as:

$$\mathbf{P}^T\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{P} = \mathbf{P}^T\mathbf{H}_A\mathbf{P}.$$

This interpretation emphasizes that the term measures how much of the polynomial basis $\mathbf{P}$ lies in the column space of the RBF basis $\mathbf{A}$.

If the polynomial $P_k(\mathbf{x})$ used in RBF approximation is of degree zero, then $\mathbf{P} = \mathbf{1}$ is a column vector of ones. The general formula for the polynomial coefficients simplifies to:

$$a_0 = \left(\mathbf{1}^T(\mathbf{I} - \mathbf{H}_A)\mathbf{1}\right)^{-1}\mathbf{1}^T(\mathbf{I} - \mathbf{H}_A)\mathbf{h}$$

or equivalently:

$$a_0 = \frac{\mathbf{1}^T(\mathbf{I} - \mathbf{H}_A)\mathbf{h}}{\mathbf{1}^T(\mathbf{I} - \mathbf{H}_A)\mathbf{1}},$$

which is analogous to Equation (4.10); similarly, in the SN-RBF approximation case. This expression gives the least-squares optimal constant term that complements the RBF approximation orthogonal to the RBF basis.

### 5.3 Characterization of the coefficient $a_0$

In the case of RBF approximation with a degree-zero polynomial, the coefficient $a_0$ plays a specific role that can be understood from various perspectives:

- *Geometrical Point of View*: Geometrically, $a_0$ represents the vertical shift (translation) of the entire RBF approximation surface in the function space. It adjusts the baseline level of the approximation so that the residuals $\mathbf{h} - \mathbf{A}\boldsymbol{\lambda}$ are optimally aligned (in the least squares sense) with the constant function. Thus, $a_0$ positions the RBF surface to best fit the data on average, after accounting for the contribution of the RBF part.
- *Algebraic Point of View*: Algebraically, $a_0$ is the projection of the residual vector $(\mathbf{h} - \mathbf{A}\boldsymbol{\lambda})$ onto the subspace spanned by the constant function, orthogonal to the RBF span. Explicitly, it is given by:

$$a_0 = \frac{\mathbf{1}^T(\mathbf{I} - \mathbf{H}_A)\mathbf{h}}{\mathbf{1}^T(\mathbf{I} - \mathbf{H}_A)\mathbf{1}},$$

  where $\mathbf{H}_A = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ is the projection matrix onto the RBF subspace. This formula ensures that $a_0$ is the optimal scalar minimizing the residual norm in the orthogonal complement of the RBF space.
- *RBF-Neural Network Interpretation*: In the framework of RBF neural networks, the term $a_0$ can be interpreted as a bias node or global bias term. While the radial basis functions (neurons) capture localized features of the data, $a_0$ adjusts the global activation level of the output layer. It ensures that the output neurons (linear combination of RBF activations) are centered correctly with respect to the target values.

## 6 Conclusions

This contribution presents **new analytical formulae** expressing the influence of the additional polynomial $P_k(\mathbf{x})$ of degree $k$. In this case, special formulae are presented for the zero-degree polynomial, as RBF and SN-RBF are invariant to rotation and translation. The zero-degree case directly impacts RBF-Neural Networks by providing an exact value for the sigmoid threshold.

The coefficient $a_0$ of the zero-degree polynomial $P_0(\mathbf{x})$ is not an average of the given values $\mathbf{h}$, nor the median, etc. It depends on the distribution of points $\mathbf{x}_i$ in the data domain $\Omega$ and given interpolated values $\mathbf{h}$.

The following are the main impacts of the analytical formula for $a_0$:

- *RBF Interpolation*: Ensuring solvability when working with conditionally positive definite kernels; interpreting $a_0$ as a global bias or base level in the interpolant, supporting bias correction and hierarchical decomposition.
- *RBF Neural Networks*: Direct computation instead of iterative learning, improved initialization and hybrid analytical-learning strategies, interpretation of low-frequency vs high-frequency components in learned functions.

- RBF methods like the Kansa method [12] are widely used for solving PDEs. The inclusion of a polynomial term (especially constant terms) in RBF-PDE methods is known to enhance numerical stability and matrix conditioning, provide analytical error bounds, and rigorously assist in satisfying boundary conditions [8, 9].
  An explicit formula for $a_0$ can aid in the development of robust preconditioners and adaptive meshless solvers.

The derivation of the formulae above establishes a connection between the inverse of the RBF kernel matrix and the polynomial subspace, thereby providing a bridge to reproducing kernel Hilbert spaces (RKHS), projection theory, and functional approximation.

The closed-form expression enables direct computation, bypassing iterative procedures and reducing the computational cost, which is especially beneficial for online systems and streaming data scenarios.

Also, a modified formulation for the approximation case has been briefly described, and relevant formulae have been derived.

**Acknowledgements**

The author thanks to students and colleagues at the University of West Bohemia in Pilsen for their critical comments and discussions. Thanks belong also to the anonymous reviewers, as their comments and hints helped to improve this paper significantly.

## References

[1] M.E. Biancolini. *Fast Radial Basis Functions for Engineering Applications*. Springer, first edition, 2017. https://doi.org/10.1007/978-3-319-75011-8.

[2] M.D. Buhmann. On quasi-interpolation with radial basis functions. *Journal of Approximation Theory*, **72**(1):103–130, 1993. https://doi.org/10.1006/jath.1993.1009.

[3] R. Cavoretto. Optimizing the shape parameter in rational RBF partition of unity interpolation. *Applied Mathematics Letters*, **173**, 2026. https://doi.org/10.1016/j.aml.2025.109766.

[4] R. Cavoretto, S. De Marchi, A. De Rossi, E. Perracchione and G. Santin. Partition of unity interpolation using stable kernel-based techniques. *Applied Numerical Mathematics*, **116**:95–107, 2017. https://doi.org/10.1016/j.apnum.2016.07.005.

[5] M. Cervenka and V. Skala. Conditionality analysis of the radial basis function matrix. *LNCS*, **12250 LNCS**:30–43, 2020. https://doi.org/10.1007/978-3-030-58802-1_3.

[6] G.E. Fasshauer. *Meshfree Approximation Methods with Matlab*. World Scientific, first edition, 2007. https://doi.org/10.1142/6437.

[7] M.S. Floater and A. Iske. Multistep scattered data interpolation using compactly supported radial basis functions. *Journal of Computational and Applied Mathematics*, **73**(1-2):65–78, 1996. https://doi.org/10.1016/0377-0427(96)00035-0.

[8] N. Flyer, B. Fornberg, V. Bayona and G.A. Barnett. On the role of polynomials in RBF-FD approximations: I. interpolation and accuracy. *Journal of Computational Physics*, **321**:21–38, 2016. https://doi.org/10.1016/j.jcp.2016.05.026.

[9] B. Fornberg and G. Wright. Stable computation of multiquadric interpolants for all values of the shape parameter. *Computers and Mathematics with Applications*, **48**(5-6):853–867, 2004. https://doi.org/10.1016/j.camwa.2003.08.010.

[10] R. Franke. A critical comparison of some methods for interpolation of scattered data. Technical report, Naval Postgraduate School Monterey CA, 1979. Available on Internet: `https://apps.dtic.mil/sti/pdfs/ADA081688.pdf`

[11] R.L. Hardy. Theory and applications of the multiquadric-biharmonic method 20 years of discovery 1968-1988. *Computers & Mathematics with Applications*, **19**(8-9):163–208, 1990. https://doi.org/10.1016/0898-1221(90)90272-L.

[12] E.J. Kansa, R.C. Aldredge and L. Ling. Numerical simulation of two-dimensional combustion using mesh-free methods. *Engineering Analysis with Boundary Elements*, **33**(7):940—950, 2009. https://doi.org/10.1016/j.enganabound.2009.02.008.

[13] E. Larsson and R. Schaback. Scaling of radial basis functions. *IMA Journal of Numerical Analysis*, **44**(2):1130–1152, 2024. https://doi.org/10.1093/imanum/drad035.

[14] Z. Majdisova and V. Skala. Big geo data surface approximation using radial basis functions: A comparative study. *Computers and Geosciences*, **109**:51–58, 2017. https://doi.org/10.1016/j.cageo.2017.08.007.

[15] Z. Majdisova and V. Skala. Radial basis function approximations: comparison and applications. *Applied Mathematical Modelling*, **51**:728–743, 2017. https://doi.org/10.1016/j.apm.2017.07.033.

[16] F.C.M. Menandro. Two new classes of compactly supported radial basis functions for approximation of discrete and continuous data. *Engineering Reports*, **2019**:1–30, 2019. https://doi.org/10.1002/eng2.12028.

[17] Ch.A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive Approximation*, **2**(1):11–22, 1986. https://doi.org/10.1007/BF01893414.

[18] A. Noorizadegan, D. Chen, Ch.S. Shan, R. Cavoretto and A. De Rossi. Efficient truncated randomized SVD for mesh-free kernel methods. *Computers and Mathematics with Applications*, **164**:12–20, 2024. https://doi.org/10.1016/j.camwa.2024.03.021.

[19] S.A. Sarra and D. Sturgill. A random variable shape parameter strategy for radial basis function approximation methods. *Engineering Analysis with Boundary Elements*, **33**(11):1239–1245, 2009. https://doi.org/10.1016/j.enganabound.2009.07.003.

[20] R. Schaback. Optimal geometric Hermite interpolation of curves, mathematical methods for curves and surfaces. ii. *Innov. Appl. Math*, pp. 417–428, 1998. Available on Internet: `http://num.math.uni-goettingen.de/schaback/research/papers/OGHIoC.pdf`

[21] F. Schwenker, H.A. Kestler and G. Palm. Three learning phases for radial-basis-function networks. *Neural Networks*, **14**(4-5):439–458, 2001. https://doi.org/10.1016/S0893-6080(01)00027-2.

[22] V. Skala. RBF interpolation with CSRBF of large data sets. In *ICCS 2017, Proceedia Computer Science*, volume 108, pp. 2433–2437. Elsevier, 2017. https://doi.org/10.1016/j.procs.2017.05.081.

[23] M. Smolik and V. Skala. Large scattered data interpolation with radial basis functions and space subdivision. *Integrated Computer Aided Engineering*, **25**:49–62, 2018. https://doi.org/10.3233/ICA-170556.

[24] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in computational Mathematics*, **4**(1):389–396, 1995. https://doi.org/10.1007/BF02123482.

[25] H. Wendland. *Scattered data approximation*, volume 17. Cambridge University Press, 2004. https://doi.org/10.1017/CBO9780511617539.

[26] Z. Wu. Compactly supported positive definite radial functions. *Advances in computational mathematics*, **4**(1):283–292, 1995. https://doi.org/10.1007/BF03177517.