
MFREE: A MATLAB TOOLKIT FOR MESHFREE APPROXIMATION SCHEMES

A PREPRINT

Konstantinos A. Mountris*

Aragón Institute of Engineering Research, IIS Aragón
CIBER-BNN
University of Zaragoza
Spain, Zaragoza, ZGZ 50018
kmountris@unizar.es

Esther Pueyo

Aragón Institute of Engineering Research, IIS Aragón,
CIBER-BBN
University of Zaragoza
Spain, Zaragoza, ZGZ 50018
epueyo@unizar.es

September 9, 2020

1 Summary

Meshfree Methods (MMs) are considered an attractive alternative to mesh-based methods such as the Finite Element Method (FEM) due to their efficiency in solving complex problems (e.g. large deformation, fracture). However, MMs have not yet reached the maturity of FEM and they are a topic of active research. While there is a vast literature in MMs, commercial or open source packages based on MMs are still limited and implementing them from scratch may be challenging.

Here we present MFREE, a MATLAB toolkit providing an implementation of several meshfree approximation schemes for 1D, 2D and 3D problems. We chose MATLAB because its user-friendly syntax facilitates understanding of the implementation procedure, which makes it popular in the engineering community. MFREE implements the Moving Least Squares (MLS), the Radial Point Interpolation (RPI) and the Moving Kriging Interpolation (MKI) meshfree approximation schemes, which are widely used in MMs such as the Element-free Galerkin [1] and Meshless Local Petrov Galerkin [2] methods.

MFREE is designed to be used by both engineering researchers and students in courses on computational mechanics and other topics where MMs can be of interest. It has already been used in a number of scientific publications [3, 4]. This toolkit aims to provide a basis for the design and implementation of meshfree packages that serves to support the integration of MMs in the engineer's artillery to solve challenging problems.

The software is available for download from [GitHub](#) at [KMountris/MFREE](#).

*[mail] kmountris@unizar.es [url] <https://www.mountris.org>

2 Theoretical aspects

For any real-valued function $u(\mathbf{x})$ defined in a domain $\Omega \subset \mathbb{R}^d$, a meshfree approximation $u^h(\mathbf{x})$ at a point $I \in \Omega$ is obtained by:

$$u^h(\mathbf{x}_I) = \sum_{i=1}^m \phi_I^i u(\mathbf{x}_i) \quad (1)$$

where \mathbf{x}_I and \mathbf{x}_i are the coordinates vectors of the point I and the i^{th} point ($i = 1, 2, \dots, m$) in the support domain of point I (being I one of these points). ϕ_I^i is the value of the meshfree approximation function ϕ_I at point i . In the following, we briefly describe the construction of ϕ_I in the Moving Least Squares, Radial Point Interpolation and Moving Kriging Interpolation schemes. The vector ϕ_I contains the values of the meshfree approximation function at all points i in the support domain of point I .

2.1 Moving Least Squares

In the Moving Least Squares (MLS) scheme [5], the meshfree approximation function ϕ_I for point I is given by:

$$\phi_I = \mathbf{p}_I \mathbf{A}^{-1} \mathbf{B}, \quad (2)$$

where the polynomial basis \mathbf{p}_I for point I and matrices \mathbf{A} and \mathbf{B} are defined as follows.

For any point i in the support domain of point I with coordinates $\mathbf{x}_i = (x_i, y_i)$ in 2D (analogously for 1D and 3D), the complete polynomial basis \mathbf{p}_i is given by:

$$\mathbf{p}_i = [1 \ x_i \ y_i \ x_i^2 \ x_i y_i \ y_i^2 \ \dots \ x_i^k \ y_i^k]. \quad (3)$$

where k is the maximal order of the monomials in \mathbf{p}_i . Let's denote the number of monomials in \mathbf{p}_i by n , which depends on k and on the spatial dimension (1D, 2D or 3D). For the linear polynomial basis, i.e. $k = 1$, \mathbf{p}_i is defined as:

$$\begin{aligned} \mathbf{p}_i &= [1 \ x_i], & (n = 2) \text{ in 1D,} \\ \mathbf{p}_i &= [1 \ x_i \ y_i], & (n = 3) \text{ in 2D,} \\ \mathbf{p}_i &= [1 \ x_i \ y_i \ z_i], & (n = 4) \text{ in 3D.} \end{aligned} \quad (4)$$

The matrices \mathbf{A} and \mathbf{B} are given by:

$$\begin{aligned} \mathbf{A} &= \mathbf{P}^T \mathbf{W} \mathbf{P}, \\ \mathbf{B} &= \mathbf{P}^T \mathbf{W}, \end{aligned} \quad (5)$$

where \mathbf{P} is the polynomial basis moment matrix:

$$\mathbf{P} = \begin{bmatrix} 1 & x_1 & y_1 & \dots & p_k(\mathbf{x}_1) \\ 1 & x_2 & y_2 & \dots & p_k(\mathbf{x}_2) \\ \dots & \dots & \dots & \dots & \dots \\ 1 & x_m & y_m & \dots & p_k(\mathbf{x}_m) \end{bmatrix}_{(m \times n)}, \quad (6)$$

\mathbf{W} is the weight function moment matrix:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1m} \\ w_{21} & w_{22} & \dots & w_{2m} \\ \dots & \dots & \dots & \dots \\ w_{m1} & w_{m2} & \dots & w_{mm} \end{bmatrix}_{(m \times m)}. \quad (7)$$

2.2 Radial Point Interpolation

In the Radial Point Interpolation (RPI) scheme [6], the meshfree approximation function ϕ_I for point I is given by:

$$\phi_I = \{\mathbf{r}_I \ \mathbf{p}_I\} \mathbf{G}^{-1}, \quad (8)$$

where \mathbf{p}_I is the polynomial basis as defined from equation (3). For any point i in the support domain of point I , \mathbf{r}_i denotes the radial basis function vector given by:

$$\mathbf{r}_i = [r_{i1} \ r_{i2} \ \dots \ r_{im}], \quad (9)$$

where r_{ij} , $i, j = 1, 2, \dots, m$, is the value of the radial basis function r_i for point i evaluated at point j , i.e. $r_{ij} = r_i(\mathbf{x}_j)$. For common choices of radial basis functions, see [6].

The matrix \mathbf{G} is given by:

$$\mathbf{G} = \begin{bmatrix} \mathbf{R} & \mathbf{P} \\ \mathbf{P}^T & \mathbf{0} \end{bmatrix}_{(m+n \times m+n)}, \quad (10)$$

where \mathbf{P} is the polynomial basis moment matrix defined as in equation (6) and \mathbf{R} is the radial basis function moment matrix:

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1m} \\ r_{21} & r_{22} & \dots & r_{2m} \\ \dots & \dots & \dots & \dots \\ r_{m1} & r_{m2} & \dots & r_{mm} \end{bmatrix}_{(m \times m)}. \quad (11)$$

2.3 Moving Kriging Interpolation

In the Moving Kriging Interpolation (MKI) scheme [7], the meshfree approximation function ϕ at point I is given by:

$$\phi_I = \mathbf{p}_I \mathbf{A} + \mathbf{c}_I \mathbf{B} \quad (12)$$

where \mathbf{p}_I is the polynomial basis as defined from equation (3). For any point i in the support domain of point I , \mathbf{c}_I denotes the correlation function vector given by:

$$\mathbf{c}_i = [c_{i1} \ c_{i2} \ \dots \ c_{im}] \quad (13)$$

where c_{ij} , $i, j = 1, 2, \dots, m$, is the value of the correlation function c_i for point i evaluated at point j , i.e. $c_{ij} = c_i(\mathbf{x}_j)$. For common choices of correlation functions, see [7].

The matrices \mathbf{A} and \mathbf{B} are obtained by:

$$\begin{aligned} \mathbf{A} &= (\mathbf{P}^T \mathbf{C}^{-1} \mathbf{P})^{-1} \mathbf{P}^T \mathbf{C}^{-1} \\ \mathbf{B} &= \mathbf{C}^{-1} (\mathbf{I} - \mathbf{P} \mathbf{A}) \end{aligned} \quad (14)$$

where \mathbf{I} is the identity matrix, \mathbf{P} is the polynomial basis moment matrix defined as in equation (6) and \mathbf{C} is the correlation function moment matrix:

$$\mathbf{C} = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1m} \\ c_{21} & c_{22} & \dots & c_{2m} \\ \dots & \dots & \dots & \dots \\ c_{m1} & c_{m2} & \dots & c_{mm} \end{bmatrix}_{(m \times m)}. \quad (15)$$

3 Program structure

In this section we describe the MFREE toolkit's program structure. MFREE is freely available at <https://www.mountris.org/software/mlab/MFREE> and on GitHub <https://github.com/KMountris/MFREE>. It is released under the MIT license and its current stable release is version 1.0.0.

3.1 aux

The subfolder **aux** contains auxiliary functions. The function *LineMesh.m* constructs a regular mesh of line elements in 1D. *QuadMesh.m* constructs a regular quadrilateral mesh in 2D and *HexMesh.m* constructs a regular hexahedral mesh in 3D. The functions *TriMesh.m* and *TetMesh.m* construct an irregular mesh of simplicial elements in 2D and 3D, respectively. The functions *ShowSolution2D.m* and *ShowSolution3D.m* are used to visualize the solution of a scalar field in either 2 or 3 dimensions.

3.2 mfree-1.0.0

The subfolder **mfree-1.0.0** contains the implementaion of the meshfree approximation functions. *MlsShape.m*, *RpiShape.m*, and *MkiShape.m* are used to compute the MLS, RPI and MKI meshfree approximation functions, respectively. Both the approximation function and its first gradient are computed. The interface function *MfreeShape.m* can be used to compute either the MLS, RPI or MKI approximation, depending on the selected scheme. *MfreeOptions.m* generates a structure containing the default values of the parameters for each meshfree approximation. The default options should be adjusted by the user before use to ensure accurate results. Finally, the functions *ComputeEnrichedRadialBasis* and *ComputeCorrelationBasis.m* are used to compute the corresponding basis functions that are used for the construction of the RPI and MKI approximations, respectively.

3.3 paper

The subfolder **paper** contains source files for the generation of the accompanying article “MFREE: A matlab toolkit for meshfree approximation schemes”.

3.4 support

The subfolder **support** contains the function *SupportRadius.m* which is used to calculate the radius of the support domain for each node of the domain and *SupportNeighs.m* which is used to find the neighbor points i in the local support domain of point I .

3.5 textprogressbar

The subfolder **textprogressbar** contains the program “Paul (2020). Text progress bar (<https://www.mathworks.com/matlabcentral/fileexchange/28067-text-progress-bar>), MATLAB Central File Exchange” for progress reporting during execution.

3.6 weights

The subfolder **weights** contains the implementation of various weight functions, radial basis functions and correlation functions that are used for the computation of W , R and C matrices in the MLS, RPI and MKI schemes, respectively. The available functions are: cubic spline (*CubicSpline.m*), quartic spline (*QuarticSpline.m*), Gaussian function (*Gaussian.m*), multiquadric radial basis function (*Multiquadric.m*), compact-support radial basis function (*CompactRBF.m*), thin plate spline (*ThinPlateSpline.m*).

4 Examples

The programs *test_1D.m*, *test_2D.m*, *test_3D.m* provide examples of how to use the MFREE toolkit for 1D, 2D and 3D problems, respectively. In these programs, the basis functions and the first derivatives of the meshfree function approximations (MLS, RPI, MKI) are compared. An illustration for the 1D case is given in Figure 1. In this example, the MLS approximation is computed using the quartic spline as weight function. The multiquadric radial basis function and the Gaussian function are used as radial basis function and correlation function, respectively, for the computation of the RPI and MKI approximations. All the approximations are computed using the linear polynomial basis ($k = 1$).

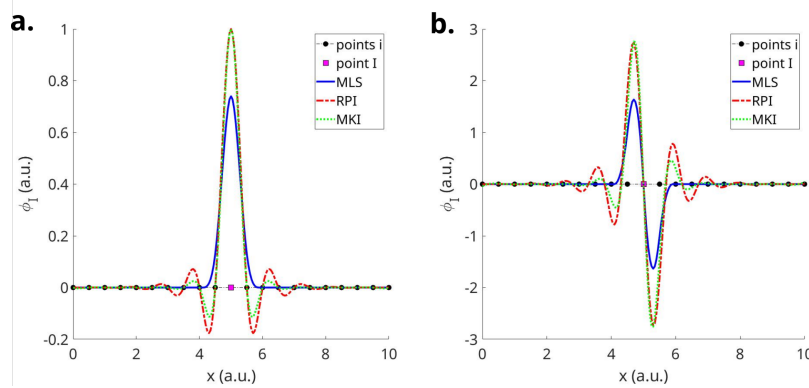


Figure 1: Meshfree approximation functions for point I . The approximation function (panel (a)) and its first order derivative (panel (b)) are shown for the MLS, RPI and MKI meshfree approximation schemes.

Acknowledgments

This work was supported by the European Research Council under grant agreement ERC-StG 638284, by Ministerio de Ciencia e Innovación (Spain) through project PID2019-105674RB-I00 and by European Social Fund (EU) and Aragón Government through BSICoS group (T39_20R) and project LMP124-18. Computations were performed by the ICTS NANBIOSIS (HPC Unit at University of Zaragoza).

References

- [1] T Belytschko et al. “Element-free Galerkin methods for static and dynamic fracture”. In: *International Journal of Solids and Structures* 32.17-18 (1995), pp. 2547–2570.
- [2] SN Atluri and T Zhu. “A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics”. In: *Computational Mechanics* 22.2 (1998), pp. 117–127.
- [3] KA Mountris, C Sanchez, and E Pueyo. “A novel paradigm for in silico simulation of cardiac electrophysiology through the Mixed Collocation Meshless Petrov-Galerkin Method”. In: *2019 Computing in Cardiology (CinC)*. Vol. 46. IEEE. 2019, 10.22489/CinC.2019.382, pp. 1–4.
- [4] KA Mountris and E Pueyo. “The Radial Point Interpolation Mixed Collocation (RPIMC) method for the solution of transient diffusion problems”. In: *arXiv preprint arXiv:2001.01027* (2020).
- [5] P Lancaster and K Salkauskas. “Surfaces generated by moving least squares methods”. In: *Mathematics of Computation* 37.155 (1981), pp. 141–158.
- [6] GR Liu et al. “Point interpolation method based on local residual formulation using radial basis functions”. In: *Structural Engineering and Mechanics* 14.6 (2002), pp. 713–732.
- [7] L Gu. “Moving kriging interpolation and element-free Galerkin method”. In: *International Journal for Numerical Methods in Engineering* 56.1 (2003), pp. 1–11.