Parallel Numerics '05, 57-66 M. Vajteršic, R. Trobec, P. Zinterhof, A. Uhl (Eds.)
Chapter 3: Differential Equations ISBN 961-6303-67-8

Mesh Free Method Applied to the Diffusion Equation

Marjan Šterk [1,*], Borut Robič [2], and Roman Trobec [1]

[1] Jožef Stefan Institute
Jamova 39, SI-1000 Ljubljana, Slovenia

[2] Faculty of Computer and Information Science,
University of Ljubljana
Tržaška 25, SI-1000 Ljubljana, Slovenia

Partial differential equations have traditionally been solved by finite difference or finite element methods. The new mesh free methods combine the advantages of both, not requiring a mesh of elements while providing better accuracy than finite differences. The accuracy results from using weak formulation of the equation where the solution is approximated by a set of base functions defined on local support domains. In this paper a mesh free method is illustrated on the one dimensional diffusion equation. The methodology of deriving a system of algebraic equations from the weak formulation of the differential equation is given. Parallelization potentials and advantages of mesh free methods are discussed, in particular for applications where the domain shape changes with time.

1 Introduction

Physical phenomena and different engineering problems modeled by partial differential equations (PDE) have traditionally been solved by finite differences (FDM) or finite elements (FEM). Both methods are based on a set of points that are positioned in the problem domain and connected into a mesh. While FDM is based on approximating the derivatives in PDE by Taylor series [1, 2], FEM approximates the solution u by a linear combination of shape functions [3, 2]:

$$u^h(x) = \sum_i u_i \phi_i(x) = \mathbf{u}' \boldsymbol{\phi}(x). \qquad (1)$$

[*]Corresponding author. E-mail: marjan.sterk@ijs.si

The shape functions ϕ_i are defined on local support domains, therefore only the nearest mesh neighbors contribute to the approximate solution in a given point (see Figure 1).

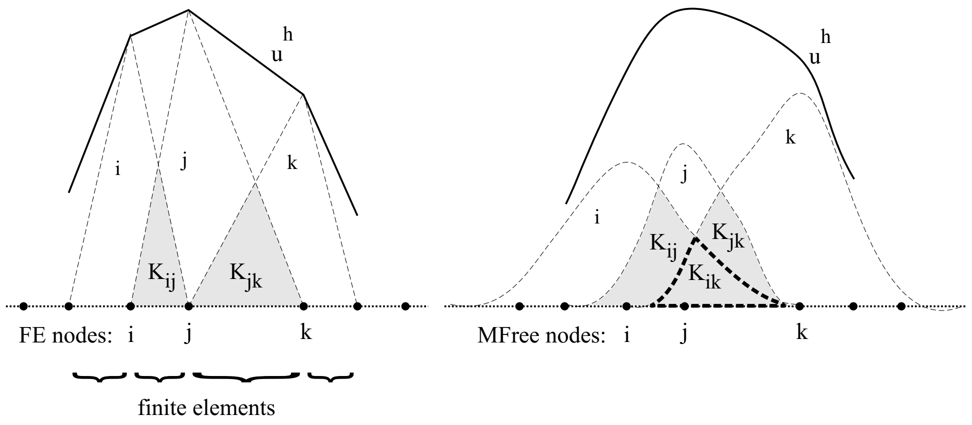

Figure 1: Problem domain Ω (dotted line) discretized by finite elements (left) and by MFM nodes (right). FEM linear shape functions and MFM moving least squares shape functions ϕ are shown (dashed line) for three points of interest i, j, and k. The resulting solution approximation is drawn in bold. The overlapping support domain are shaded and denoted by corresponding elements of global stiffnes matrix K.

The shape functions are known in advance and are the same for all elements of the same type. Solving a PDE corresponds to finding such coefficients u_i of the linear combination that result in minimization of the residual, i.e. the difference between left-hand and right-hand side of the PDE. The well known Galerkin method [4], a special case of weighted residual methods, forces the residual to be minimal. The described methodology yields a sparse system of algebraic equations $K\mathbf{u} = F$ that has to be solved, usually by a parallel iterative solver [2, 1, 5].

It is obvious that in some applications where the problem domain shape and boundaries are changing significantly during the evolution of the solution, initial elements cannot describe well neither the shape nor the boundaries of the problem domain. For example in explosions some parts of the domain will disappear, in structural analysis cracks in material will not follow element boundaries, during heart beating the shape and structure of the heart will change with time [6], etc. The standard solution is re-meshing after any significant distortion in order to retain correspondence between meshes and changes in the problem domain. New difficulties arise now, interpolation between subsequent meshes is needed, accuracy is degraded on critical parts of the problem domain, load balancing can be lost in case of parallel execution, and finally a great amount of human intervention is needed. The mesh free methods (MFM) are a viable alternative that can potentially alleviate mentioned problems.

The roots of MFM are in smooth particle hydrodynamics (SPH) method first published in [7] in 1977. SPH method was applied for the simulation of astronomical explosions with no boundaries and based on the strong form of PDE, but not analyzed in details. A number of approaches based on weak form and implementing boundaries have been published later [8, 9, 10, 11] and mesh free methods are becoming widely used and standardized today. In this paper we focus on an application of the mesh free local Petrov-Galerkin method (MLPG) [12]. As the name indicates, MFM need no predefined global mesh for their implementation. The problem domain and its boundary are represented by a set of scattered nodes. Shape functions contributing to the approximate solution are defined on local subsets of nodes called the support domain instead of on finite elements (see Figure 1). The global stiffness matrix K contains non-zeros only on places corresponding to the shaded integrands in the figure.

In contrast to FEM, the shape functions differ for each point of interest and are constructed during instead of before the numerical analysis. A widely used method for constructing the MFM shape functions is the moving least square (MLS) approximation. After the shape functions are created the MFM methods follow a procedure analogous to FEM in order to generate the final sparse system of algebraic equations.

In the rest of the paper some details about the MLPG are given. In particular, MLS shape functions are introduced and the basic principles of the solution approximation are described. In Section 3 MLPG is applied to the one dimensional diffusion equation. Some more details about the derivation of the system of algebraic equations and time discretization are given. The paper concludes with a short discussion about possible parallelization efficiency of the MFM and their future.

2 Meshless Local Petrov-Galerkin Method

As was described in the Introduction, mesh free methods discretize the domain with nodes scattered throughout the domain and its boundaries. The only requirement is that the nodes be dense enough, particularly in places where the solution has large derivatives. In the one-dimensional case, the domain $[a, b]$ is discretized to nodes $x_1 \dots x_n, x_1 = a, x_n = b$.

In Meshless Local Petrov-Galerkin (MLPG) method, the solution is approximated using moving least squares (MLS) approximation, which is used to formulate the residual. The latter is then used to transform the weak form of the PDE into an algebraic system.

2.1 Moving Least Squares Approximation

The MLS approximation, originated by Lancaster and Salkauskas [13], approximates the solution u as

$$u^h(x) = \sum_{j=1}^{m} p_j(x) a_j(x) = \mathbf{p}^T(x)\mathbf{a}(x), \tag{2}$$

where $\mathbf{p}(x)$ is a vector of m monomials $1, x, \ldots, x^{m-1}$, and $\mathbf{a}(x)$ is a vector of coefficients, which are functions of x. Note that m should be smaller than the minimal number of nodes in the support domain of any point x.

Given a set of nodes $x_I (I = S_{xL} \ldots S_{xR})$ in the support domain[1] of point x and the values u_I that $u^h(x)$ should approximate, we define a weighted residual J of the approximation:

$$J = \sum_{I=S_{xL}}^{S_{xR}} \widehat{W}_I(x) \left[\mathbf{p}^T(x_I)\mathbf{a}(x) - u_I\right], \tag{3}$$

where $\widehat{W}_I$ is a weight function centered at x_I such that $\widehat{W} = 0$ outside the support domain of x_I. $\widehat{W}$ thus ensures local support and gives less weighting to nodes further away. Note that u^h will only approximate the nodal values u_I and in general will not pass through them. u_I is therefore termed nodal parameter as opposed to nodal value.

The minimization condition for J requires

$$\frac{\partial J}{\partial \mathbf{a}} = 0, \tag{4}$$

which results in a $m \times m$ linear system that has to be solved for every point of interest x where u_h is to be evaluated. Given x, the i-th MLS shape function $\phi_i(x)$ that depends only on x and on the locations of nodes in its support domain can be written explicitly.

[1]We use S_{xL} and S_{xR} as the indices of the leftmost and rightmost node in the support domain of point x, respectively. Similarly, S_{iL} and S_{iR} correspond to the support domain of the node x_i.

2.2 Solving time-dependent PDEs with MLPG method

Time-dependent PDEs require that the nodal parameters u_j be functions of time, while the shape functions ϕ_j remain functions of x only. Based on the MLS shape functions, the solution approximation and its derivatives can be written in the general form

$$u^h(x,t) = \sum_{j=S_{xL}}^{S_{xR}} u_j(t)\phi_j(x), \qquad u_t^h = \sum_{j=S_{xL}}^{S_{xR}} u_j'(t)\phi_j(x), \tag{5}$$

$$u_x^h = \sum_{j=S_{xL}}^{S_{xR}} u_j(t)\phi_j'(x), \qquad u_{xx}^h = \sum_{j=S_{xL}}^{S_{xR}} u_j(t)\phi_j''(x), \tag{6}$$

where the subscripts denote derivatives.

Given a PDE, the MLPG method defines a local quadrature domain Ω_{Qi} for each internal node x_i $(i = 2 \ldots n-1)$. In one dimension, the quadrature domain is a continuous interval whose boundaries x_{QiL} and x_{QiR} are not necessarily nodes. The integral of weighted residual over the quadrature domain is then forced to equal zero:

$$\int_{\Omega_{Qi}} r\widehat{W}_i d\Omega_{Qi} = 0, \tag{7}$$

where $\widehat{W}_i$ is a weight function that is non-zero only on Ω_{Qi}. Weight function can be the same as the one used in MLS shape function construction.

Because $u^h(x_j,t)$, in general, is not equal to $u_j(t)$, the boundary conditions must be enforced with two additional equations:

$$u^h(x_1,t) = u_a, \;\; u(x_n,t) = u_b. \tag{8}$$

As will be shown, (7) and (8) can be transformed into a system of n linear equations for n nodal parameters u_i.

3 Applying MLPG to the Diffusion Equation

The one dimensional diffusion equation on interval $[a,b]$ is

$$u_t - cu_{xx} - f = 0, \;\; u(a,t) = u_a, \;\; u(b,t) = u_b, \;\; u(x,0) = u_0, \tag{9}$$

where $u(x,t)$ is the solution, c and f are material properties (i.e. heat conductivity and heat source in case of heat equation), which can also be functions of x and t. Dirichlet boundary conditions u_a, u_b and initial conditions u_0 are also prescribed. Given an approximate solution u^h, the residual is defined as

$$r = u_t^h - cu_{xx}^h - f. \tag{10}$$

3.1 MLPG Procedure

Applying (7) to (10) gives a set of equations numbered from $i = 2$ to $i = N-1$:

$$\int_{\Omega_{Qi}} \left[u_t^h - cu_{xx}^h - f\right] \widehat{W}_i d\Omega_{Qi} = 0. \tag{11}$$

The integral containing the second derivative u_{xx}^h is evaluated by parts:

$$\int_{\Omega_{Qi}} -cu_{xx}^h \widehat{W}_i d\Omega_{Qi} = -cu_x^h \widehat{W}_i \Big|_{x_{QiL}}^{x_{QiR}} + \int_{\Omega_{Qi}} cu_x^h \widehat{W}_{i,x} d\Omega_{Qi}, \tag{12}$$

where $\Omega_{Qi} = [x_{QiL}, x_{QiR}]$. Eq. (12) is substituted into (11):

$$\int_{\Omega_{Qi}} \left[u_t^h - f\right] \widehat{W}_i d\Omega_{Qi} + \int_{\Omega_{Qi}} cu_x^h \widehat{W}_{i,x} d\Omega_{Qi} - cu_x^h \widehat{W}_i \Big|_{x_{QiL}}^{x_{QiR}} = 0. \tag{13}$$

Then, u^h and its derivatives are replaced by the sums introduced in (5-6):

$$\begin{aligned} &\int_{x_{QiL}}^{x_{QiR}} \left[\sum_{j=S_{xL}}^{S_{xR}} u_j'(t)\phi_j(x) - f(x)\right] \widehat{W}_i(x) dx \\ &\quad + c\int_{x_{QiL}}^{x_{QiR}} \sum_{j=S_{xL}}^{S_{xR}} u_j(t)\phi_j'(x)\widehat{W}_i'(x) dx \\ &\quad - c\sum_{j=S_{xL}}^{S_{xR}} u_j(t)\phi_j'(x)\widehat{W}_i(x)\Bigg|_{x_{QiL}}^{x_{QiR}} = 0. \end{aligned} \tag{14}$$

The sums can be moved outside the integrals, e.g.,

$$\int_{x_{QiL}}^{x_{QiR}} \sum_{j=S_{xL}}^{S_{xR}} u_j'(t)\phi_j(x)\widehat{W}_i(x) dx = \sum_{j=S_{QiL}}^{S_{QiR}} \int_{x_{QiL}}^{x_{QiR}} u_j'(t)\phi_j(x)\widehat{W}_i(x) dx. \tag{15}$$

Note that x in the integral can be anywhere in Ω_{Qi}, thus the shape functions centered at all nodes in the support domain of any point inside Ω_{Qi} have to be taken into account, i.e. shape functions at nodes from S_{QiL} and S_{QiR}.

Using (15) on (14) and rearranging the terms, we get

$$\begin{aligned} &\sum_{j=S_{QiL}}^{S_{QiR}} u_j'(t) \int_{x_{QiL}}^{x_{QiR}} \phi_j(x)\widehat{W}_i(x) dx - \int_{x_{QiL}}^{x_{QiR}} f(x)\widehat{W}_i(x) dx \\ &+ c\sum_{j=S_{QiL}}^{S_{QiR}} u_j(t) \left[\int_{x_{QiL}}^{x_{QiR}} \phi_j'(x)\widehat{W}_i'(x) dx - \phi_j'(x)\widehat{W}_i(x)\Big|_{x_{QiL}}^{x_{QiR}}\right] = 0. \end{aligned} \tag{16}$$

In rewriting the system in matrix form, terminology stemming from structural mechanics is traditionally used. The stiffness matrix K contains the terms with spatial derivatives, i.e. terms in square brackets in (16). Its elements corresponding to the contributions of internal nodes i are

$$K_{i,j} = c\left[\int_{x_{QiL}}^{x_{QiR}} \phi_j'(x)\widehat{W}_i'(x)dx - \phi_j'(x)\widehat{W}_i(x)\Big|_{x_{QiL}}^{x_{QiR}}\right], \tag{17}$$

while the boundary conditions are enforced by separate equations. Note that K is sparse because $K_{i,j}$ is zero for j outside the interval $[S_{QiL}, S_{QiR}]$.

The damping matrix C contains the time derivative terms, i.e. the first term in (16):

$$C_{i,j} = \int_{x_{QiL}}^{x_{QiR}} \phi_j(x)\widehat{W}_i(x)dx, \tag{18}$$

and has the same sparseness pattern as K.

The force vector $\mathbf{f}$ contains the remaining second term in (16):

$$\mathbf{f}_i = \int_{x_{QiL}}^{x_{QiR}} f(x)\widehat{W}_i(x)dx. \tag{19}$$

The final form of (16) is:

$$\sum_{j=S_{QiL}}^{S_{QiR}} \left[C_{i,j}u_j'(t)\right] + \sum_{j=S_{QiL}}^{S_{QiR}} \left[K_{i,j}u_j(t)\right] - \mathbf{f}_i = 0 \tag{20}$$

3.2 Time Discretization

Time is best discretized using the Crank-Nicolson scheme, which replaces the time derivative at half-step $u_j'(t + \Delta t/2)$ with the central difference approximation:

$$u_j'(t + \Delta t/2) = \frac{u_j(t + \Delta t) - u_j(t)}{\Delta t} \tag{21}$$

and approximates $u_j(t + \Delta t/2)$ as the average of $u_j(t)$ and $u_j(t + \Delta t)$. The scheme is unconditionally stable and provides second order accuracy in both time and space.

Collecting all the nodal parameters into a vector $\mathbf{u}^{(t)} = (u_1^{(t)}, \ldots, u_n^{(t)})$, the matrix form of the time-discretized linear system (20) is obtained:

$$(2C + \Delta tK)\mathbf{u}^{(t+\Delta t)} = (2C - \Delta tK)\mathbf{u}^{(t)} + 2\Delta t\mathbf{f}. \tag{22}$$

3.3 Enforcing Boundary Conditions

To enforce the Dirichlet boundary conditions, the first and last equation in the linear system (22) are replaced by (8), where u^h is expressed as in (5). The final system that has to be solved at every time-step has the form

$$A\mathbf{u}^{(t+\Delta t)} = B\mathbf{u}^{(t)} + \overset{*}{\mathbf{f}}, \tag{23}$$

where

$$A_{i,j} = \begin{cases} \phi_j(x_i) & x_j \text{ is in the support domain of boundary node } x_i, \\ 2C_{i,j} + \Delta t K_{i,j} & x_j \text{ is in the support domain of any point in the quadrature domain of internal node } x_i, \\ 0 & \text{otherwise,} \end{cases} \tag{24}$$

$$B_{i,j} = \begin{cases} 0 & x_j \text{ is in the support domain of boundary node } x_i, \\ 2C_{i,j} - \Delta t K_{i,j} & x_j \text{ is in the support domain of any point in the quadrature domain of internal node } x_i, \\ 0 & \text{otherwise,} \end{cases} \tag{25}$$

$$\overset{*}{\mathbf{f}}_i = \begin{cases} u_a & i = 1, \\ u_b & i = n, \\ 2\Delta t \mathbf{f}_i & \text{otherwise.} \end{cases} \tag{26}$$

4 Discussion and Conclusions

A simple derivation of MLPG method for the one dimensional diffusion equation was presented in this paper. It becomes clear that the computational complexity of the MFM is significantly higher than that of FEM. On the other hand, FEM requires initial discretization of the problem domain into a mesh and also later remeshing in cases where the domain shape will change significantly. While MFM have not yet reached the final point of their development and are still immature, they have good chances to become fully adaptive and automatic, eliminating any manual work needed.

In the era of ever faster computers, the calculation complexity could be alleviated by parallel computation. The basic properties of mesh free approach (locality and higher computational load) are promising also for parallel implementation, because high speedups can be expected. Our future work will be focused mainly in this direction. Potential opportunities for the parallelization of MFM with their performance analysis will be investigated in detail.

References

[1] G. Golub and J. M. Ortega. *Scientific Computing - An Introduction with Parallel Computing.* Academic Press Inc., Boston, 1993.

[2] M. T. Heath. *Scientific Computing: An Introductory Survey, 2nd Ed.* WCB/McGraw-Hill, 2002.

[3] P. I. Kattan. *MATLAB Guide to Finite Elements: an Interactive Approach.* Springer-Verlag, 2003.

[4] T. Belytschko, Y. Y. Lu, and L. Gu. Element-free galerkin methods. *International Journal of Numerical Methods in Engineering*, 37:229–256, 1994.

[5] P. Wesseling. *An Introduction to Multigrid Methods.* John Wiley and Sons, 1991.

[6] R. Trobec, B. Slivnik, B. Geršak, and T. Gabrijelčič. Computer simulation and spatial modelling in heart surgery. *Computers in Biology and Medicine*, 4:393–403, 1998.

[7] L. B. Lucy. A numerical approach to the testing of the fission hypothesis. *The Astronomical Journal*, 82:1013–1024, 1977.

[8] G. R. Liu. *Mesh Free Methods: Moving beyond the Finite Element Method.* CRC Press, Boca Raton, 2003.

[9] K. T. Danielson, R. A. Uras, M. D. Adley, and S. Li. Large-scale application of some modern csm methodologies by parallel computation. *Advances in Engineering Software*, 31:501–509, 2000.

[10] F. Günther, W. K. Liue, D. Diachin, and M. A. Christon. Multi-scale meshfree parallel computations for viscious, compressible flows. *Comput. Methods Appl. Mech. Engrg.*, 190:279–303, 2000.

[11] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. Meshless methods: An overview and recent developments, 1996.

[12] S. N. Atluri and S. Shen. *The Meshless Local Petrov-Galerkin (MLPG) Method.* Tech Science Press, Encino, 2002.

[13] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Math. Comput.*, 37:141–158, 1981.