An introduction of the GE/BC embedded meshless method by using an ODE problem as example

Nan-Jing Wu*, Yen-Ting Lin
Department of Civil and Water Resources Engineering, National Chiayi University, Chiayi City, Taiwan
email: njwu@mail.ncku.edu.tw

Abstract. In this paper, the GE/BC embedded meshless method is illustrated by using a 1D boundary value problem as an example. The formulation for solving one dimensional ordinary differential equation is listed. Comparison with conventional collocation method is also demonstrated. The GE/BC embedded meshless method performs much better, especially in the computation of the derivative at boundary nodes.

1. Introduction
In many engineering applications, physical phenomena are described by using differential equations. So, mathematics is a very important curriculum in engineering programs in universities. In most practical cases, only numerical solutions could be obtained because of the complexity of the problem. Conventional numerical methods such as finite difference method (FDM), finite element method (FEM), or boundary element method (BEM) need grid/mesh for seeking the solution. Mesh or grid which denotes the connectivity among the adjacent nodes plays a very important role. The accuracy of the numerical results depends on the shapes, the sizes and the uniformity of the mesh panels/elements. Generating an adequate mesh is a quite professional task.

In recent decades, meshless/mesh-free methods have drawn more and more attentions due to the capability for problems with complex geometry and moving boundaries. General speaking, there are two types of meshless methods, the strong form and the weak form. Strong form methods are more straightforward and easy learning/coding, but they are usually unstable [1, 2]. Introduction of the development of meshless methods and the comparisons among various meshless methods could be found in references [1-3].

Mainly there are two kinds of strong form methods. One is based on RBF (radial basis functions) and the other is based on local polynomial approximation with the WLS (weighted least-squares) or MLS (moving least-squares). RBF collocation methods are mostly global type while local polynomial collocation methods must belong to the local type. The local type indicates that only some neighboring nodes within a supporting range are used when constructing the local approximation. The well-known finite point method (FPM) [4] and the general finite difference method are typical representatives of this family.
Theoretically, the governing equations should be satisfied on boundaries. However, strong form meshless methods only use the boundary conditions when implementing the collocation. Consequently, accurate derivatives at boundary nodes are rarely obtained. This is the reason why strong form methods are usually unstable. In Ref. [5, 6], strong methods that make the governing equations satisfied as well as the boundary conditions at boundary nodes were proposed. Inspired by Ref. [5,6], Wu and Tsay [7] modified the FPM [4] by using the penalty weighting to make the governing equations satisfied at boundary nodes as well. That method was firstly denominated as a “Robust” collocation method, and also the “modified FPM”. However, since many other meshless methods are also robust, and there are many other modifications of the FPM, Chiou and Wu [8] borrow the phrase “embedded” to rename the method of Wu and Tsay [7].

Though the method of Wu and Tsay [7] is much easier and robust, it is not widely used yet. In this paper, we demonstrate it with a simple ODE problem as an example in the hope that this numerical method will be more and more widely known.

2. The local polynomial approximation with WLS
Mathematical expressions for 2D and 3D problems of this method could be found in Refs. [7-11]. The following expressions are for 1D. In the domain of \( x_L \leq x \leq x_R \), function \( y(x) \) satisfies the governing equation and the boundary conditions.

\[
\begin{align*}
y'' + p(x)y' + q(x)y &= r(x) , \quad x_L \leq x \leq x_R \\
y' + s(x)y &= t(x) , \quad x = x_L \text{ and } x = x_R
\end{align*}
\]  

(1)

The domain is discretized as \( N \) nodes and our goal is to find the approximations of \( y \) at \( x_1, x_2, ..., x_N \). At the \( j \)-th node, function \( y \) is approximated as

\[
y|_{x_j} = \alpha_{j1} + \alpha_{j2}X + \alpha_{j3}X^2 / 2 + \alpha_{j4}X^3 / 6
\]  

(2)

in which \( X = x - x_j \) denotes the local coordinate, and \( \alpha_{j1}, \alpha_{j2}, \alpha_{j3}, \alpha_{j4} \) are coefficients to be determined. These coefficients also represent the approximations of function \( y \) and its derivatives of the first, second and third order. If we let \( \alpha_{j1} \) exact to \( y_j \), the above formulations can reduce to the general finite difference method.

A more general form of Eq. 2 is

\[
y|_{x_j} = \sum_{i=1}^{m} \alpha_{ji} f_i(X)
\]  

(3)

As we are considering an 1D problem and the third degree polynomial is employed, we get \( m = 4 \). We need more than \( m \) nodes to construction the local approximation.

\[
A\alpha_j = Wy
\]  

(4)

in which

\[
A = \begin{bmatrix} a_{11} & a_{12} & L & L & a_{1m} \\ a_{21} & a_{22} & a_{2m} \\ M & a_{3i} & M \\ M & O & M \\ a_{ni,1} & a_{ni,2} & L & L & a_{ni,m} \end{bmatrix}
\]  

(5)

\[
a_{ij} = w_{j,i}(x_i - x_j)
\]  

(6)
in which \( y_\ell = y(x_\ell) \) and \( k \) is the local index of node \( x_\ell \) in the \( j \)-th local domain. The underline is to emphasize it is a local index. The \( l \)-th node is locally numbered as the \( k \)-th node if it is inside the \( j \)-th supporting range. The symbol \( n_{loc} \) denotes the number of nodes enclosed in the range of \( \left| x - x_j \right| < \rho_j \). The symbol \( W_\rho \) represents the weighting factor whose value is between 0 and 1 and is dependent on the distant from \( x_j \) to \( x_l \) (i.e. \( r_\rho = \left| x_j - x_l \right| \)). There are many functions for determining the weighting factor. In this paper, we use the normalized Gaussian function.

\[
W_\rho = \begin{cases} 
\frac{\exp(-\epsilon(r_\rho / \rho_j)_2^2) - \exp(-\epsilon)}{1 - \exp(-\epsilon)}, & r_\rho < \rho_j \\
0, & r_\rho \geq \rho_j 
\end{cases}
\]

In which \( \epsilon \) is the shape parameter. There is no exact solution for Eq. 4. The best solution we can obtain is the least-squares solution.

\[
\alpha = \begin{bmatrix} \alpha_{j1} & \alpha_{j2} & \alpha_{j3} \end{bmatrix}^T
\]

\[
1
T
j
\begin{bmatrix} \alpha & A & A & W \end{bmatrix}^T
\begin{bmatrix} y \end{bmatrix}
\]

3. The conventional collocation method

The following boundary value problem is chosen as the example.

\[
y'' - 10y' = 0, \quad y(0) = 1, \quad y(1) = 2
\]

Discretizing the domain with 6 nodes uniformly, we have \( x_1 = 0, \quad x_2 = 0.2, \quad x_3 = 0.4, \quad x_4 = 0.6, \quad x_5 = 0.8 \), and \( x_6 = 1 \). The size of the supporting range is set to be 1.01 times of the minimal distance that can enclose at least 5 nodes. Therefore, we have \( \rho_1 = \rho_6 = 0.808, \quad \rho_2 = \rho_5 = 0.606 \), and \( \rho_3 = \rho_4 = 0.404 \). For conciseness, only 6 decimal places are shown in the following expressions. It should be kept in mind that all computations are in 14 significant digits.

At the first node, one can obtain the following equation by Eq. 4
And by using Eq. 12, we get
\[
\begin{bmatrix}
1.000000 & 0.000000 & 0.000000 & 0.000000 \\
0.692384 & 0.138477 & 0.013848 & 0.000923 \\
0.229811 & 0.091924 & 0.018385 & 0.002451 \\
0.036487 & 0.021892 & 0.006568 & 0.001314 \\
0.001280 & 0.001024 & 0.000410 & 0.000109
\end{bmatrix}
\begin{bmatrix}
\alpha_{i1} \\
\alpha_{i2} \\
\alpha_{i3} \\
\alpha_{i4}
\end{bmatrix} =
\begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4
\end{bmatrix}
\] (14)

Similarly, at the second node we have
\[
\begin{bmatrix}
0.999998 & 0.000006 & -0.000010 & 0.000006 & -0.000002 \\
-9.157044 & 14.961509 & -7.442263 & 1.628175 & 0.009623 \\
49.859417 & -124.437666 & 99.156499 & -24.437666 & -0.140584 \\
-124.323431 & 372.293724 & -370.940585 & 122.293724 & 0.676569
\end{bmatrix}
\begin{bmatrix}
\alpha_{i1} \\
\alpha_{i2} \\
\alpha_{i3} \\
\alpha_{i4}
\end{bmatrix} =
\begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4
\end{bmatrix}
\] (15)

At other nodes we can get analogous equations for the local approximation. They are skipped for saving the page space.

By deeming approximations as exact values to apply them in the governing equation
\[
\begin{cases}
y' = \alpha_{j2} \\
y'' = \alpha_{j3}
\end{cases} \Rightarrow \begin{cases}
y' = y'' = 0 \\
\alpha_{j3} - 10\alpha_{j2} = 0
\end{cases}
\] (18)

and applying the boundary conditions \( y_1 = 1 \) and \( y_6 = 2 \), the global matrix system is obtained.
The numerical solution is then obtained.

\[ \begin{bmatrix}
1.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 \\
41.677303 & -25.042543 & -24.936185 & 8.290790 & 0.010636 & 0.000000 \\
-4.163319 & 58.319944 & 49.979916 & 8.346723 & 4.170014 & 0.000000 \\
0.000000 & -4.163319 & 58.319944 & -8.346723 & -8.4979916 & 4.170014 \\
0.000000 & -0.013397 & 8.279746 & 74.919618 & 74.946412 & 8.319936 \\
1.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & 1.000000
\end{bmatrix} \begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4 \\
y_5 \\
y_6
\end{bmatrix} = \begin{bmatrix}
1 \\
0 \\
0 \\
0 \\
0 \\
2
\end{bmatrix} \quad (19)
\]

This is how the collocation is employed in meshless methods. Making the governing equation satisfied at boundary nodes is not implemented.

4. Embedding the GE and BC to the local approximation

According to the definition of the local approximation, the governing equation in Eq. 1 can be written as

\[ \alpha_{j3} + p_j \alpha_{j2} + q_j \alpha_{j1} = r_j \]

in which \( p_j = p(x_j), \ q_j = q(x_j), \) and \( r_j = r(x_j). \) At a boundary node, the boundary condition should also be satisfied. The boundary condition could also be expressed with the local approximation.

\[ y'(x) + s(x)y(x) = t(x) \] at \( x = x_j \)

\[ \Rightarrow \alpha_{j2} + s_j \alpha_{j1} = t_j \]

in which \( s_j = s(x_j), \ t_j = t(x_j). \) Eq. 4 is modified when the GE and BC are embedded to the local approximation.

\[ \begin{bmatrix}
A' \\
A
\end{bmatrix} \begin{bmatrix}
\alpha_j \\
\mathbf{A}_{ij}
\end{bmatrix} = \begin{bmatrix}
\mathbf{Wy} \\
\mathbf{Wy'}
\end{bmatrix} \quad (23)
\]

in which

\[ A' = \begin{bmatrix}
w'q_j & w'p_j & w'q_j & w'p_j & 0 & 0 \\
w'q_j & w'p_j & w'q_j & w'p_j & 0 & 0 \\
w's_j & w' & w's_j & w' & 0 & 0
\end{bmatrix}, \text{ if } x_j \text{ is an inner node} \]

\[ \mathbf{A} = \begin{bmatrix}
w^r_j & w^r_j & w^r_j & w^r_j & w^r_j \\
w^r_j & w^r_j & w^r_j & w^r_j & w^r_j
\end{bmatrix}, \text{ if } x_j \text{ is a boundary node} \]

\[ \mathbf{B}' = \begin{bmatrix}
w^r_j \\
w^r_j \\
w^r_j \\
w^r_j
\end{bmatrix}, \text{ if } x_j \text{ is an inner node} \]

\[ \mathbf{B} = \begin{bmatrix}
w^r_j \\
w^r_j \\
w^r_j \\
w^r_j \\
w^r_j
\end{bmatrix}, \text{ if } x_j \text{ is a boundary node} \]

where \( w' \) is the penalty weighting whose value is much greater than 1. Its purpose is to diminish the error in the governing equation and the boundary condition. There is no exact solution for Eq. 23. The best solution we can obtain is the least-squares solution.
\[ \alpha_j = \left[ \begin{array}{c} A' \\ A \end{array} \right] \left[ \begin{array}{c} A' \\ A \end{array} \right]^T \left[ \begin{array}{c} A' \\ A \end{array} \right] W_y \beta' \]  

(26)

5. Collocation with the GE/BC embedded method

The boundary value problem expressed in Eq. 3 is demonstrated as the example. We choose \( w' = 100 \). At the first node, one can obtain the following equation by Eq. 23

\[
\begin{bmatrix}
1.000000 & 0.000000 & 0.000000 & 0.000000 \\
0.692384 & 0.138477 & 0.013848 & 0.000923 \\
0.229811 & 0.091924 & 0.013845 & 0.000923 \\
0.036487 & 0.021892 & 0.006568 & 0.001314 \\
0.001280 & 0.001024 & 0.000410 & 0.000109 \\
0.000000 & -1000.000000 & 100.000000 & 0.000000 \\
100.000000 & 0.000000 & 0.000000 & 0.000000
\end{bmatrix}
\begin{bmatrix}
\alpha_{11} \\
\alpha_{12} \\
\alpha_{13} \\
\alpha_{14}
\end{bmatrix} = \begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4 \\
y_5 \\
y_6 \\
y_7 \\
y_8
\end{bmatrix}
\]  

(27)

It should be noted that the difference between Eq. 27 and Eq. 14 is the last two rows in Eq. 27 which represent the additional constraints of the governing equation and the boundary conditions. The least square solution of Eq. 27 is

\[
\begin{bmatrix}
\alpha_{11} \\
\alpha_{12} \\
\alpha_{13} \\
\alpha_{14}
\end{bmatrix} = \begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4 \\
y_5 \\
y_6 \\
y_7 \\
y_8
\end{bmatrix}
\]  

(28)

At the second node, we can obtain
\[
\begin{bmatrix}
0.520202 & -0.104040 & 0.010404 & -0.000694 \\
1.000000 & 0.000000 & 0.000000 & 0.000000 \\
0.520202 & 0.104040 & 0.010404 & 0.000694 \\
0.073190 & 0.029276 & 0.005855 & 0.000781 \\
0.001280 & 0.000768 & 0.000230 & 0.000046 \\
0.000000 & 1000.000000 & 100.000000 & 0.000000
\end{bmatrix}
\begin{bmatrix}
\alpha_{21} \\
\alpha_{22} \\
\alpha_{23} \\
\alpha_{24}
\end{bmatrix}
= 
\begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4 \\
y_5 \\
y_6
\end{bmatrix}
\tag{29}
\]

And then
\[
\begin{bmatrix}
\alpha_{21} \\
\alpha_{22} \\
\alpha_{23} \\
\alpha_{24}
\end{bmatrix}
= 
\begin{bmatrix}
0.046926 & 0.971781 & -0.028034 & 0.009308 & 0.000018 \\
1.975491 & -4.684370 & 2.812911 & -0.103760 & -0.000272 \\
19.754909 & -46.843700 & 28.129109 & -1.037599 & -0.002719 \\
-498.382621 & -347.410951 & 598.779261 & 50.014276 & 0.073620
\end{bmatrix}
\begin{bmatrix}
y_1 \\
y_2 \\
y_3 \\
y_4 \\
y_5 \\
y_6
\end{bmatrix}
\tag{30}
\]

At other nodes we can get analogous equations for the local approximations. They are skipped for saving the page space. Using the first row in Eq. 28 and deeming the approximations as exact values, we can obtain

\[
y_i = \begin{bmatrix}
0.000100 & 0.000002 & -0.000001 & 0.000000 & 0.000000 \\
y_2 & y_3 & y_4 & y_5 & + 0.999899
\end{bmatrix}
\Rightarrow \begin{bmatrix}
-0.999900 & 0.000002 & -0.000001 & 0.000000 & 0.000000 \\
y_3 & y_4 & y_5 & y_6 & = -0.999899
\end{bmatrix}
\tag{31}
\]

The second to the sixth rows in the global matrix system can be obtained in the same way. Finally, the global matrix system is obtained.
and the numerical solution is obtained.

\[
\begin{bmatrix}
-0.999900 & 0.000002 & -0.000001 & 0.000000 & 0.000000 & 0.000000 \\
0.046926 & -0.028219 & -0.028034 & 0.009308 & 0.000018 & 0.000000 \\
-0.000015 & 0.000157 & 0.000147 & 0.000000 & 0.000005 & 0.000000 \\
0.000000 & -0.000015 & 0.000157 & -0.000147 & 0.000000 & 0.000005 \\
0.000000 & -0.000019 & -0.015727 & 0.142112 & 0.142150 & 0.015784 \\
0.000000 & 0.000000 & 0.000000 & 0.000000 & 0.000000 & -0.999900 \\
\end{bmatrix}
= \begin{bmatrix}
-0.999899 \\
0.000000 \\
0.000000 \\
0.000000 \\
0.000000 \\
1.999795 \\
\end{bmatrix}
\]  

(32)

6. The comparison and the conclusion

The exact solution of this benchmark problem is \( y_1 = e^{-10(1-x)} / (1-e^{-10}) \). One could use the finite difference method with the second order central difference scheme to obtain \( y_1 = L = y_3 = 1 \), and \( y_6 = 2 \). Both methods illustrated in this paper perform much better than the FDM.

The root mean square error of the conventional collocation method is 0.027896. The root mean square error of the numerical solution of the BC/GE embedded collocation method is 0.013149. The improvement is perspicuous. We can also check the accuracy of the first order derivative at the boundary. Applying Eq. 20 to Eq. 15, one gets \( y'_1 = \alpha_{12} = 0.146279 \). Applying Eq. 33 to Eq. 28, one gets \( y'_1 = \alpha_{12} = 0.010612 \). The result of the BC/GE embedded collocation method is much closer to the exact value which is zero.

In the example illustrated in this paper, the benefit of the embedding the governing equation and the boundary condition to the local approximation is shown. The accuracy of the derivative at the boundary is greatly improved.

References

[1] Liu, G.R. and Gu, Y.T., An Introduction to Meshfree Methods and Their Programming, Springer: The Netherlands, 2005.
[2] Nguyen, V.P., Rabczuk, T., Bordas, S., and Duflot, M., Meshless methods: a review and computer implementation aspects”, Mathematics and Computers in Simulation, 79, 763–813, 2008.
[3] Chen, J.-S., Hillman, M., Chi, S.-W., Meshfree Methods: Progress Made after 20 Years, Journal of Engineering Mechanics, 143, 04017001, 2017.
[4] Oñate, E., Idelsohn, S., Zienkiewicz, O.C., and Taylor, R.L., A finite point method in computational mechanics. Applications to convective transport and fluid flow, International Journal for Numerical Methods in Engineering, 39, 3839-3866, 1996.
[5] Jin, X.Z., Li, G., and Aluru, N.R., New approximations and collocation schemes in the finite cloud method, Computers and Structures, 83, 1366–1385, 2005.
[6] Wu, N.-J. and Chang, K.-A., Simulation of free-surface waves in liquid sloshing using a domain
type-meshless method", *International Journal for Numerical Methods in Fluids*, 67, 269-288, 2011.

[7] Wu, N.-J. and Tsay, T.-K., A robust local polynomial collocation method", *International Journal for Numerical Methods in Engineering*, 93, 355-375, 2013.

[8] Chiou, Y.-C., Wu, N.-J., A GE/BC imbedded local polynomial collocation method for two dimensional multivariable problems, *Engineering Analysis with Boundary Elements*, 100, 185-194, 2019.

[9] Wu, N.-J., Tsay, T.-K., and Chen, Y.-Y., Generation of stable solitary waves by a piston-type wave maker", *Wave Motion*, 51, 240-255, 2014.

[10] Wu, N.-J., Hsiao, S.-C., and Wu H.-L., Mesh-free simulation of liquid sloshing subjected to harmonic excitations", *Engineering Analysis with Boundary Elements*, 64, 90-100, 2016.

[11] Hsiao, S.-C., Shih, M.-Y., Wu, N.-J., Simulation of Propagation and Run-Up of ThreeDimensional Landslide-InducedWaves Using aMeshless Method, *Water*, 10(5), 552, 2018.