2015

Adaptivity For Meshfree Point Collocation Methods In Linear Elastic Solid Mechanics

Joshua Wayne Derrick
University of South Carolina

ADAPTIVITY FOR MESHFREE POINT COLLOCATION METHODS IN LINEAR ELASTIC SOLID MECHANICS

by

Joshua Wayne Derrick

Bachelor of Science in Engineering
University of South Carolina, 2013

Submitted in Partial Fulfillment of the Requirements

For the Degree of Master of Science in

Civil Engineering

College of Engineering and Computing

University of South Carolina

2015

Accepted by:

Robert L. Mullen, Director of Thesis

Juan Caicedo, Reader

Joseph Flora, Reader

Shamia Hoque, Reader

Lacy Ford, Senior Vice Provost and Dean of Graduate Studies

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and gratitude to my academic advisor, Dr. Robert L. Mullen, for accepting the task of advising me the remainder of my graduate studies after the departure of my research advisor. Dr. Mullen's invaluable guidance and help was more than I could have asked for and I appreciate all of the energy he spent helping me to complete this research.

ABSTRACT

This study presents the behavior of local node adaptivity for point collocation meshfree numerical methods and its application to linear elastic solid mechanics. A bisection approach to meshfree adaptivity will be presented with numerical examples showing that convergence is achieved. A threshold value is required from the user and this value is an acceptable change in gradient for the problem. If the change in gradient between 2 nodes exceed this limit, a node is added at the mid-point between the two nodes. This process applies to all nodes in the model. Once the refinement is complete, the model is processed again and this procedure is repeated until the program reaches a limit of iterations or the acceptable error is achieved.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Computer simulations are powerful tools for practicing and research engineers due to the increasing power in computer technology and ability to simulate full scale engineered systems without the need to run expensive and potentially dangerous laboratory experiments. Computer simulation software is driven primarily by the numerical methods developed to approximate complex equations that do not have an obtainable analytical solution. These complex equations are often Partial Differential Equations (PDE) that describe the physics of a system, but cannot be easily solved analytically.

Emergence of numerical methods for PDE's have been targeted at one of two forms taken by PDE's; the strong form and the weak form. The strong form is a phrase that refers to the governing differential equation written at a point in the domain along with its associated boundary conditions. The weak form is an integral form of the governing equation. Boris Galerkin devised a transformation scheme to convert a continuous PDE to an approximate-discrete integral form for the purpose of making it easier to obtain a solution to these complex equations. This transformation is an effective approach because the integral form has weaker requirements on space of functions satisfying the continuity conditions for the governing equation making it easier to numerically approximate the solution to a given PDE.

Meshed Methods and Meshfree Methods.

Numerical methods used to approximate the solution for PDE's can be divided into two categories: meshed methods and meshfree methods.

Meshed methods are numerical methods that require a mesh for the purposes of approximating the integral form of a PDE. Mesh is a geometric representation of a model constructed from an arrangement of elements and nodes where elements are lines (1D models), triangles, quadrilaterals (2D models), tetrahedrons, pentahedrons and hexahedrons (3D models) and nodes are the vertices of each on an element. Typically, elements have assigned interpolation schemes to convert the nodal approximations to a functional representation over the domain of that element. This allows the model to have a continuous function to determine the solution values at any point in the model based on the nodal solutions only. One purpose of having mesh is to use the nodal solutions and element interpolants to obtain the functions over that element. When the union of all elements form the problem domain, a function can be constructed for the entire model allowing access to the solution values at any point in that model.

Meshfree methods, on the other hand, are numerical methods that use only nodes. They do not approach the approximation by local interpolation functions inside an element; instead, they use a nodal derivative approximation scheme at a point. A function for a model can be obtained with meshfree analysis but not from functional with the local support, but by computing the nodal derivatives at a point and expanding the function over a decreasing domain of influence. The most common approach to derivative approximation is the Least Squares Method (LSM). The LSM is a derivative

approximation scheme in which a weighted function is expanded about each node to form a derivative approximation based on the relative location of the surrounding nodes. The detail of the LSM used in this research will be expanded more in chapter 3.

Using meshed based numerical methods is sometimes cumbersome due to the complexity of the models' geometry. Mesh generation can be difficult for models with complex geometries and ill-conditioned solutions can result from certain mesh constructions, for example when the mesh is highly distorted. The interest for meshfree technology is also motivated by the absence of mesh and eliminating the need for complex computational geometry programs. Computational geometry programs can be very time consuming and computationally expensive depending on the fineness of the model and the degree of accuracy prescribed. In meshed methods, nodal data and element data is required for the operations, where in meshfree methods, only nodal data is needed for the approximation schemes.

Numerical Methods in Solid Mechanics.

In the field of solid mechanics, numerical methods are typically used for stress analysis, meaning, as forces are applied to a non-rigid body, the body deforms and internal energy is developed which leads to the development of stress in that body. Some of the most common methods are the Finite Difference Method (FDM), Boundary Element Method (BEM) and Finite Element Method (FEM).

The Finite Difference Method (FDM) requires a structured mesh of nodes but is a derivative approximation scheme used to compute nodal solutions of the strong form based on finite difference equations. These finite difference equations are nodal

dependent, in other words, the solution of a node depends on the solution of the surrounding nodes. The grid needed for most FDM's can introduce more difficulties compared to general mesh based methods, as the grid is needed to be well structured. The FDM can be a less desirable choice for modeling due to the node arrangement restriction making it difficult to model problems with curved boundaries or inclusions, also, it is not well suited for modeling discontinuities.

The Boundary Element Method (BEM) is a meshed numerical method that discretizes the boundary of a problem into elements and nodes and uses a boundary integral formulation to approximate the weak form. The BEM uses polynomial interpolants to approximate the solutions in the domain between known solutions on the boundary. The BEM requires that a fundamental solution be known for the method to work, however, the fundamental solution are known for only a limited number of linear PDE's.

The Finite Element Method (FEM) is one of the most common tools used in stress analysis due to its robust architecture. The FEM is a meshed method used to approximate the weak form by discretizing the model into finite elements and integrating over each element to obtain nodal solutions. The integration is often done by numerical integration such as Gauss Quadrature. Gauss Quadrature maps the physical element to a simple-shaped element and uses the sum of the function values multiplied by weight values at so-called Gauss points to approximate the integration. Gauss Points are specific points that lie in the mapped space the provide optimal solutions to and the recommended number of Gauss Points used are dependent on the order of polynomial prescribed in the model. Once the integration is complete, the nodal solutions are then used to interpolate a polynomial over each element producing a solution over the entire domain. The FEM can

be problematic to work with when modeling complex problems due to the computational requirements needed to process the data such as complex 3D geometries, multi-physics modeling, discontinuities, etc.

The Element-Free Galerkin Method (EFGM) [1] is a meshfree numerical method that uses nodes only and incorporates the Moving Least Squares Method (MLSM) to approximate the derivatives of the weight functions for the weak form of a differential equation. The EFGM does, however, use a background mesh to generate a nodal discretization. Once the nodal solutions are known, Gauss Integration of each cell is used to compute the integral form.

Developments in Meshfree Methods.

Meshfree numerical methods have become a more popular choice for modeling since they do not require the computational power and resources like meshed methods. Meshfree methods are very popular in studies regarding fracture mechanics and crack propagation because the mesh refinement process is not required for approximations as cracks propagate through a body. Several methods have been introduced into modeling fracture such as that in [1, 2, 3] where the meshfree method is used with enriched weight functions for modeling fatigue crack growth. Another meshfree method was presented for modeling elastic cracks in [2] where a point collocation scheme based on diffuse derivatives was presented.

Concept of Adaptivity.

Adaptivity in numerical methods is the process of changing the grid, mesh or nodes to improve the approximate solution. For meshed numerical methods, adaptivity is practiced

with two difference approaches: h-adaptivity and p-adaptivity. H-adaptivity is the idea of using smaller elements in regions with a high gradient in the solution, where p-adaptivity is the process of using a higher order interpolant for smoothing out a solution where the current interpolant is not capturing all of the effects of the nodal solution. Another approach to adaptivity is L-adaptivity in which the discretizations are moved based on the solution. In the FEM, L-adaptivity is moving existing nodes closer to the regions with high gradients so that refinement is completed without the need to add elements. In meshfree methods, h-adaptivity is the most commonly form of adaption practiced. Meshfree adaptivity is a popular practice due to the simplicity of refinement for meshfree methods. To adapt a model that only contains points simply means adding more points in strategically computed locations.

Summary of Study.

In this study, adaptive techniques are developed and applied to meshfree point collocation methods. Specifically bisection adaptivity has been studied with other meshfree methods containing structured node discretization patterns and its behavior with the point collocation method will be addressed. The normalized interpolation method is another adaptive approach where an acceptable change in the gradient is defined by the user and a calculated number of nodes will be added between two existing nodes with excessive gradients to normalize the change in gradients between the two existing nodes. The meshfree numerical method comes from Yoon and Song [2] where their point collocation scheme is based on a minimized residual functional and the weight function for neighboring nodes are zero under the node of interest. Several numerical examples

will be presented to show the behavior of the error and convergence for this meshfree numerical method combined with these adaptivity schemes.

CHAPTER 2

LITERATURE REVIEW

The use of adaptive methods in numerical analysis is quiet extensive. Here, the focus is on adaptivity in meshfree methods. Adaptivity for meshfree numerical methods are motivated by the reduction in numerical modeling by local refinement strategies rather than defining an extremely fine model that could be extremely computationally expensive or time consuming.

Adaptivity in meshed based methods such as the FEM and BEM has been an efficient alternative to refining the entire domain into an extremely fine mesh that would have extremely high computational costs and require computational resources that may not be available. The criteria for self-automated adaptivity programs come from error estimators that decide if the error in a particular region is too high and requires refined mesh for the reduction of error. In meshed methods, the typical error computation is known as the energy norm given in [3]. The energy norm is integrating the product of the elements' stress and strain and summing all the element energy up over the domain. The use of the energy norm requires elements or background mesh for the purposes of integration.

Adaptivity for meshfree numerical methods is becoming a more popular modeling choice because the computational efforts of refining mesh is eliminated. H-Adaptivity in meshfree methods is the process of adding nodes to a region that meet the criteria for local refinement. It is popular because it is computationally efficient to model complex non-linear physics problems such as fracture mechanics and crack propagation.

Adaptivity has been implemented primarily for problems involving complex physics such as fracture mechanics or crack propagation. Belytschko and Rabczuk [4] implemented the algorithm they proposed in [5] to model arbitrary evolving cracks and large deformation by including enrichment functions to mathematically model the discontinuity in the domain.

A common approach is using a structured grid of nodes and defining cells that connect the nodes. Ling [6] proposed an adaptive method in which the model was discretized into a structured grid of nodes and square or cubic cells were defined as a background node connectivity mesh depending on the dimension of the problem. The algorithm proposed refined cells based on a cell principal value and if the cell met the refinement criteria, the cell was subdivided by quadrisection which is dividing 1 cell into 4 cells. Belytschko and Rabczuk [5] also proposed an adaptivity algorithm based on background cells in which each cell was quadrisected into 4 cells based on the energy in that cell. The difference between the work developed by Ling and Belytschko is that Ling does not use cell-wise integration to determine which regions will be adapted with more nodes where Rabczuk and Belytschko use background cells to determine the error based on the energy norm.

In a slightly different direction, Kee, Liu and Lu [7] use a triangular background mesh to compute the residual of the governing equation at the center of the triangular cell to determine if refinement is necessary for that cell or not. If refinement was necessary, their adaption process was adding a node at the center of the triangular cell that met the refinement process. The problem with this method is that long and skinny triangular cells could force some adapted nodes to be placed very close to each other which could lead to highly irregular node patterns with non-smooth refinement maps over the domain. In

other words, meshfree methods are highly sensitive to node distributions and triangulated cells with centroidal adaption could add more nodes to one side of a triangular cell and result in too many nodes in a localized region.

In this study, the adaptivity is truly meshfree in the since there are no background cells or background elements for integration. The error estimator is the L_2 norm of the displacement vector because all other quantities such as stress and strain and functions of the displacement. The adaptivity is based on the change in the gradient of the displacement field which is the second derivative of the displacement field. The adaptivity proposed is based on a bisection method in the sense that if the change in gradient between two nodes exceed the defined limit, a node is added at the midpoint between those two nodes. Traditional adaptivity methods implemented quadrisection of background cells but that only applies for structured node patterns with background cells.

CHAPTER 3

FORMULATION OF THE PARTICLE DIFFERENCE METHOD

In this section, the Particle Derivative Approximation (PDA) is derived as presented in [2]. This meshfree numerical method is a point collocation numerical method used to directly approximate the strong form where methods such as the EFGM is used to approximate the weak form. The PDA was constructed in the framework of the Moving Least Squares Method (MLSM) which is a numerical differentiation scheme used to compute the nodal derivatives based on weighting functions and spatial discretization of the nodes. Once the nodal derivatives and nodal solutions are computed, the Taylor Series Expansion is used to convert the discrete nodal solutions to a continuous function.

For the PDA, the continuous function, u^{cont}, will be expanded with Taylor Series Expansion about the reference coordinate, $\bar{\mathbf{x}}$, given by

$$\mathbf{u}^{cont}(\mathbf{x}, \bar{\mathbf{x}}) = \sum_{r}^{m} \frac{(\mathbf{x} - \bar{\mathbf{x}})^r}{r!} D^r \, \mathbf{u}(\mathbf{x} - \bar{\mathbf{x}}) \tag{3.1}$$

Where $\mathbf{x}$ is the spatial coordinate, $\bar{\mathbf{x}}$ is the local center, r is the order of the term, D^r is the r^{th} derivative, m is the highest order for the polynomial and $\mathbf{u}(\mathbf{x} - \bar{\mathbf{x}})$ is the nodal solution at $\mathbf{x}$ with respect to $\bar{\mathbf{x}}$). This continuous solution is summed up to the highest order term which is defined by the user. Terms can be separated out from equation (1) and rewritten to give

$$\mathbf{u}^{cont}(\mathbf{x},\bar{\mathbf{x}}) = \sum_{r}^{m} \frac{(\mathbf{x}-\bar{\mathbf{x}})^r}{r!} D^r\, \mathbf{u}(\mathbf{x}-\bar{\mathbf{x}}) = \mathbf{p}_m^T(\mathbf{x};\, \bar{\mathbf{x}})\mathbf{a}(\bar{\mathbf{x}}) \tag{3.2}$$

where $\mathbf{p}_m^T$ is the polynomial vector of length $K = \frac{(m+N)!}{N!m!}$ and N is the number of spatial dimensions (N=2 for 2D or N=3 for 3D). The polynomial vector, $\mathbf{p}_m^T$, is defined to be

$$\mathbf{p}_m^T(\mathbf{x};\, \bar{\mathbf{x}}) = \left(\frac{(\mathbf{x}-\bar{\mathbf{x}})^0}{0!}, \ldots, \frac{(\mathbf{x}-\bar{\mathbf{x}})^m}{m!}\right) \tag{3.3}$$

and the derivative coefficient vector is defined to be

$$\mathbf{a}(\bar{\mathbf{x}}) = \begin{pmatrix} D^{r_1}u(\bar{x}) \\ \vdots \\ D^{r_K}u(\bar{x}) \end{pmatrix} \tag{3.4}$$

The MLSM will be used to approximate the $\mathbf{a}(\bar{\mathbf{x}})$ terms since it is a numerical differentiation scheme and $\mathbf{a}(\bar{\mathbf{x}})$is the derivative coefficient. The residual of the approximation is given by

$$J = \sum_{i=1}^{n} w\left(\frac{\mathbf{x}_i - \bar{\mathbf{x}}}{\rho_i}\right)(\mathbf{p}_m^T(\mathbf{x};\bar{\mathbf{x}})\mathbf{a}(\bar{\mathbf{x}}) - u_i)^2 \tag{3.5}$$

where w is the weight function of node i, ρ_i is the size of the local influence domain (radius of a circle for 2D and radius of a sphere for 3D) and u_i is the discrete nodal approximation at $\mathbf{x}$ with respect to $\bar{\mathbf{x}}$. J is the residual computed between the nodal solution, u_i, and the continuous solution.

The weight function is an interesting feature of the PDA because any weight function does not have to be differentiable where in most other meshfree methods the weight function is differentiated as part of the numerical scheme used. Figure (3.1) shows how the ratio of

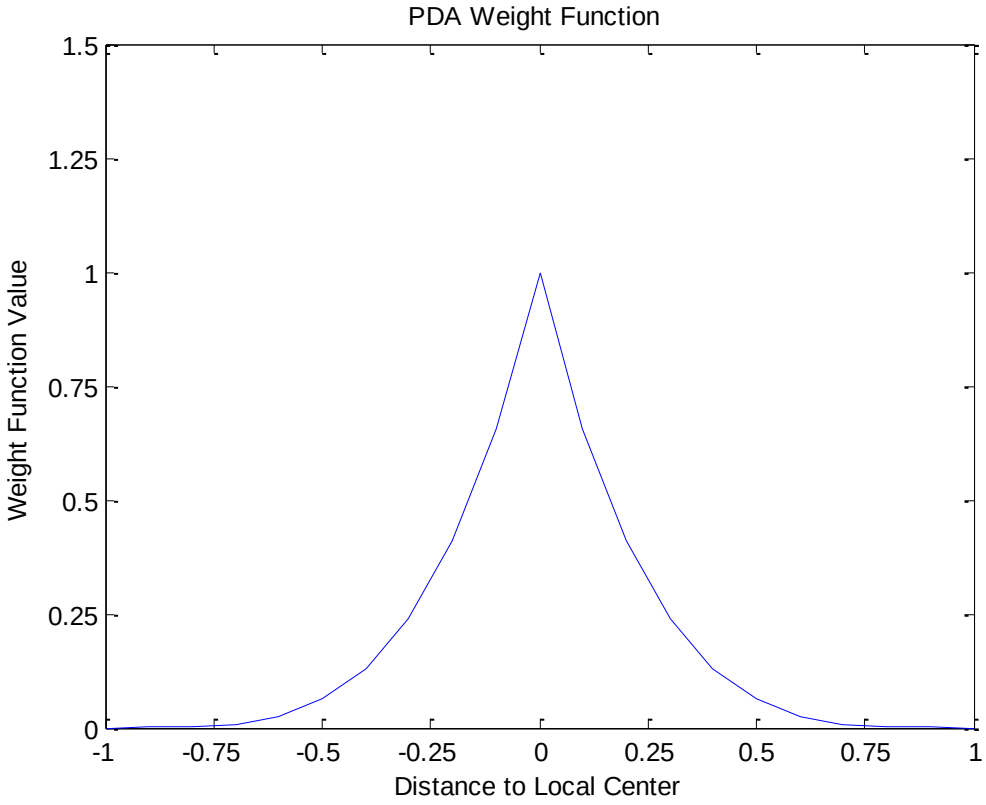


Figure 3.1: Weight function of the reference node with respect to another node where the Distance to Local Center axis is the ratio of the spatial difference over the size of the influence parameter.

the spatial difference to the influence domain size. This figure implies that all nodes that fall within the local influence domain will have a weighting value not equal to zero; where all nodes that fall outside the local influence domain will have a weight value of zero because the radius between the outside nodes and the target node is larger than the influence parameter, therefore making the ratio greater than 1. The weight function used in the PDA is given by

$$w\left(\frac{\mathbf{x}-\bar{\mathbf{x}}}{\rho_i}\right)=\left(1-\frac{\|\mathbf{x}-\bar{\mathbf{x}}\|}{\rho_i}\right)^4 \tag{3.6}$$

The size of the local influence domain is also important to discuss. This topic will be expanded in chapter 3 but an overview will be addressed in this section. The size of the local influence domain, more formally known as the *influence parameter*, is a value that determines the node density with respect to the reference node.

In traditional MLSM, the residual is minimized by setting the derivative equal to zero $\left(\frac{\partial J}{\partial \bar{\mathbf{x}}}=0\right)$ to yield the following equation

$$\mathbf{M}(\bar{\mathbf{x}})\mathbf{a}(\bar{\mathbf{x}})=\mathbf{B}(\bar{\mathbf{x}})\{u\} \tag{3.7}$$

Where M is the moment matrix and the B matrix is an assembly of p and weight functions defined as

$$\mathbf{M}(\bar{\mathbf{x}})=\sum_{i=1}^{n}\left(\mathbf{p}_m(\mathbf{x}_{\boldsymbol{i}};\bar{\mathbf{x}})\cdot w\left(\frac{\mathbf{x}-\bar{\mathbf{x}}}{\rho_i}\right)\cdot\mathbf{p}_m^T(\mathbf{x}_{\boldsymbol{i}};\bar{\mathbf{x}})\right) \tag{3.8}$$

$$\mathbf{B}(\bar{\mathbf{x}})=\left(w\left(\frac{\mathbf{x}_1-\bar{\mathbf{x}}}{\rho_1}\right)\cdot\mathbf{p}_m(\mathbf{x}_1;\bar{\mathbf{x}});\ldots;\ w\left(\frac{\mathbf{x}_{\boldsymbol{n}}-\bar{\mathbf{x}}}{\rho_n}\right)\cdot\mathbf{p}_m(\mathbf{x}_n;\bar{\mathbf{x}})\right) \tag{3.9}$$

The M matrix is size $K\times K$ and is summed over all nodes with respect to the reference node. The B matrix is size $K\times n$ where each column of the B matrix is the product of the weight function and the $\mathbf{p}_m$ vector of the reference node with respect to node i. To

obtain the derivative coefficient matrix, the inverse of the moment matrix must be computed to yield

$$\mathbf{a}(\bar{\mathbf{x}}) = \mathbf{M^{-1}}(\bar{\mathbf{x}})\mathbf{B}(\bar{\mathbf{x}})\{u\} \tag{3.10}$$

The invertibility of the moment matrix must be invertible which is only possibly by having a high enough node density around the reference node. The nodal density only depends on one parameter and that is the influence parameter. The larger the influence domain is, the higher the node density will be. By substitution, equation (10) can be rewritten as

$$\begin{pmatrix} D^{r_0}u(\bar{x}) \\ \vdots \\ D^{r_m}u(\bar{x}) \end{pmatrix} = \mathbf{M^{-1}}(\bar{\mathbf{x}})\mathbf{B}(\bar{\mathbf{x}}) \cdot \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \tag{3.11}$$

$$\begin{pmatrix} D^{r_0} \\ \vdots \\ D^{r_m} \end{pmatrix} = \mathbf{M^{-1}}(\bar{\mathbf{x}})\mathbf{B}(\bar{\mathbf{x}}) \tag{3.12}$$

Equation (12) implies that the inverse of the product of the moment matrix and the B matrix are the actual derivative approximation values. The first row is the 0^{th} derivative approximation and the second row is the first derivative, so forth and so on. These values are substituted directly into the governing differential equation of the reference node to be superimposed into the global shape function matrix. The global shape function matrix can be re-written as

$$\begin{pmatrix} D^{r_0}u(\bar{x}) \\ \vdots \\ D^{r_m}u(\bar{x}) \end{pmatrix} = \begin{bmatrix} \Phi_1^{r_0} & \cdots & \Phi_n^{r_0} \\ \vdots & \ddots & \vdots \\ \Phi_1^{r_m} & \cdots & \Phi_n^{r_m} \end{bmatrix} \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \tag{3.13}$$

Where $\Phi_k^{r_\alpha}$defines the r^{th} derivative coefficient of node *k*. Equation (13) is the form for just 1 node, so to obtain a global shape function matrix, this process should be repeated for all nodes in the domain and each of these local shape function matrices should be

superimposed into a global matrix so that there are an equal number of equations and unknowns.

Application to Solid Mechanics Problems

In the field of solid mechanics, the idea is to obtain the stress, strain and displacement fields of a problem. Extensive research in numerical methods to handle complex problems in solid mechanics have been explored such as modeling permanent deformation, cracks, contact (friction), impact and interface physics. In any solid mechanics problems, each problem is subjected to 3 boundary conditions; the essential boundary condition, the natural boundary condition and the inner domain. The essential boundary condition is a displacement boundary condition in the sense that the displacements are known. The natural boundary condition is the region on the surface of a model where tractions are prescribed. When modeling solid mechanics problems with meshfree numerical methods, each degree of freedom must be subjected to a boundary condition.

The governing equation in linear elastic solid mechanics is given by

$$\boldsymbol{\nabla \cdot \sigma + b = 0} \tag{3.14a}$$

Indicial notation (I.N.) will be used for the remaining equations. Equation (3.14a) written in I.N. and given by

$$\nabla_j \sigma_{ij} + b_i = 0 \tag{3.14b}$$

where σ is the stress tensor, and b is the body force. For the 1D case, equation (3.14b) is expanded to

$$\frac{\partial \sigma_{xx}}{\partial x} + \ b_x = 0 \tag{3.15}$$

For the 2D case, the equation is expanded to

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + b_x = 0 \tag{3.16}$$

$$\frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + b_y = 0 \tag{3.17}$$

Stress for linear elastic, homogeneous material is given by

$$\sigma_{ij} = C_{ijkl}\varepsilon_{kl} \tag{3.18}$$

The C tensor is more formally known as the elastic coefficient matrix, which is a 4th order tensor given by the following equation

$$C_{ijkl} = \lambda\delta_{ij}\delta_{kl} + \mu\left(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}\right) \tag{3.19}$$

And substituting equation (3.19) into equation (3.18) and simplifying yields

$$\sigma_{ij} = \lambda\delta_{ij}\varepsilon_{kk} + 2\mu\varepsilon_{ij} \tag{3.20}$$

Where λ and μ are material constants known as Lame parameters. The delta is more formally known as the Kronecker Delta property. The kronecker delta is given by

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{3.21}$$

Equation (3.14) is used for stress analysis for any point within the domain of a problem. The surface tractions are given by the following equation

$$t_i = \sigma_{ij}n_j \tag{3.22}$$

Where *t* is the tractions at a point and *n* is the unit normal to a point on the surface. Stress is a function of strain and strain is defined as

$$\varepsilon_{ij} = \frac{1}{2}\left(u_{i,j} + u_{j,i}\right) \tag{3.23}$$

Where the u denotes displacement at a point. By substituting equation (3.23) into equation (3.20) and substituting equation (3.20) into equation (3.14), the resulting differential equation, also known as the Navier-Lamé Equation, becomes

$$(\lambda + \mu)u_{j,ij} + (\mu)u_{i,jj} + b_i = 0 \tag{3.24}$$

This is the final form in which the differential equation will be approximate by the PDA for all nodes inside the domain. For nodes on the natural boundary subjected to applied tractions and traction-free regions, equation (3.23) will be substituted into equation (3.20) and then equation (3.20) will be substituted into equation (3.22) to obtain a differential form for the traction boundary condition nodes. The final equation will yield

$$t_i = \lambda u_{k,k} n_j + \mu(u_{i,j} + u_{j,i})n_j \tag{3.25}$$

Equation (3.24) and (3.25) are superimposed into the global shape function matrix that will be used to determine the nodal solutions, and in these sets of equations, the nodal solutions are the nodal displacements. Once the displacements are known, the first derivatives are also known for each node so using equation (3.23) the strains can be computed for each node and finally equation (3.20) will yield the stress at each node.

The PDA scheme is applied to every node in the model. In other words, the first node in the model is set to be the reference node. All the remaining equations from the PDA derivation will be applied with respect to that reference node. Once the derivative coefficients are obtained and placed in the global matrix, the second node in the model becomes the reference node and the entire process is repeated until all nodes have been the reference node for the method.

CHAPTER 4

COMPUTER IMPLEMENTATION

In this section, computer implementation will be discussed. In chapter 3, the PDA was derived and the corresponding equations was presented, however the equations and concepts can be overwhelming and tedious to decipher.

Implementation Concepts.

One of the most important concepts in using this meshfree numerical method with adaptivity is understanding that each node has *N* degrees of freedom. To be able to solve for these unknown nodal solutions, a system of equations is needed that is *n* x *n* where *n* is the total degrees of freedom in the overall system. To obtain such as matrix, the PDA will be applied to every node in the system and the matrix will come from the derivative approximations of all nodes with respect to each node.

General Process.

Below are descriptions of how to construct the PDA for linear elastic problems. Each section is listed in order of chronology.

First, the model with all prescribed boundary conditions must be defined. This model should include the dimensions, applied tractions, displacement boundary conditions, material parameters and node distribution. Each node should consist of nodal coordinates, material parameters, nodal tractions (if on the natural boundary), prescribed

displacements (if on the essential boundary), and surface unit normal (if node is on the surface). The best practice for implementation is the use of data structures for the nodes.

Once the nodal data is obtained, the nodal influence parameters should be computed. This is not part of the discretization process, but should be part of the pre-processing step because these parameters are based on the mathematical modeling data prescribed by the user. There are several methods to compute the influence parameters, but at this point each node needs an influence parameter.

At this point, the pre-processing is complete. This step is the assembly of the shape function matrix where the PDA starts. The PDA process is repeated for each node in the model, in other words, the first node takes the role as the reference point. The reference node, for the purposes of explanation will be denoted as node-i. The M and B matrix need to be computed first by iterating through all nodes in the model, this will be referred to as node-j. To construct the M matrix, the p-vector needs to be assembled. The p-vector follows the Taylor Series Expansion coefficients, for example, 2D quadratic is given by $p^T = [1\ dx\ dy\ dx^2\ dxdy\ \ dy^2]$. Where $dx = x - \bar{x}$ and $dy = y - \ \bar{y}$. The x and y are the coordinates of node-j where $\bar{x}$ and $\bar{y}$ are the coordinates of the reference node, node-i. The weight value is computed as the radial difference between node-i and node-j and if the radial distance is within the influence domain, the weight value will be non-zero. The p-vector and product of the weight value will be placed in the j^{th} column of the B matrix.

Once the M and B matrix is constructed, the M matrix needs to be inverted to obtain the derivative coefficients. The derivative coefficients should be multiplied by the Taylor Series Expansion factorial. For example, to obtain the 0^{th} derivative coefficient of node-i

with respect to all other nodes, the equation is $D_i^0 = [1\ 0\ 0\ 0\ 0\ 0]M^{-1}B$ for a 2D quadratic system, where $D_i^{x2} = [0\ 0\ 0\ 2\ 0\ 0]M^{-1}B$would be the second derivative with respect to x. Once this coefficient array is obtained, each coefficient should be substituted into the governing equation for that node and iterated through the rest of the model nodes and superimposed into the global shape function matrix. For example, consider a 1D derivative term, $\frac{d^2y}{dx^2}$, the coefficient would be $D_i^2 = [0\ 0\ 2]M^{-1}B$ and superimposed into the global shape function matrix in the row corresponding to that node. These derivative coefficient arrays are the actual values of the derivative terms without the value of the unknown function.

Once this process is over, the displacements can be solved. The differential equation coefficients are in the shape function matrix and the RHS of the differential equations consist of the body force terms, tractions and prescribed displacements for the nodes.

At this point the nodal displacements are computed so using the PDA (a derivative approximation method), the nodal strains can be computed because the strains are the derivative of the displacements. Once the strains are known, the stresses can be computed by substituting into the stress equation given back in chapter 3.

Given that this is a meshfree method, it is typical to use certain plots for visualization purposes such as triangular shaded surfaces that appear to be elements, but this does not change the fact that the PDA does not use elements or mesh to compute the solutions.

CHAPTER 5

NUMERICAL EXAMPLES

In this section, several numerical examples will be presented. The first two examples are classic examples from solid mechanics will then be presented to show that the PDA is a self-converging system in the sense that globally refining the model will converge the approximate solution to the analytical solution. Then error results for the same two examples will be presented with

Example – Cantilever Beam with Global Refinement

In this section, the PDA will be applied to a 2D cantilever beam with global refinement. The numerical solution will then be compared to the analytical solution given by Goodier and Deeks [8] which is represented by equations (5.4, 5.5). This particular problem was chosen because the analytical solution to the displacement, strain and stress fields are known. The solution for this problem is interesting in the fact that the solution was derived based on a fixed point at the origin of the coordinate system as shown in figure (5.1). It is not possible to model this problem with the traction boundary conditions only because there is instability in the physics due to an unrestrained rigidbody rotation. To model this problem, the tractions will be applied in a parabolic form and two rollers will be used to restrain the Rigidbody rotation mode at the fixed end as shown in figure (5.2). The parameters for this problem are given by table 5.1 with variable names as shown in the program. For his problem, the material model is plane-stress and not plane-strain.

Plane-stress is used because plane-strain is used to model structures that are 1-dimensional in the sense that is has a relatively large dimension in one direction. The assumption for plane stress is that the stress normal to the plane of the domain is 0. Since the domain of the model is given in x and y coordinates, the assumptions associated with plane-stress is that $\sigma_{zz}, \sigma_{xz}, \sigma_{yz}$ are all zero.

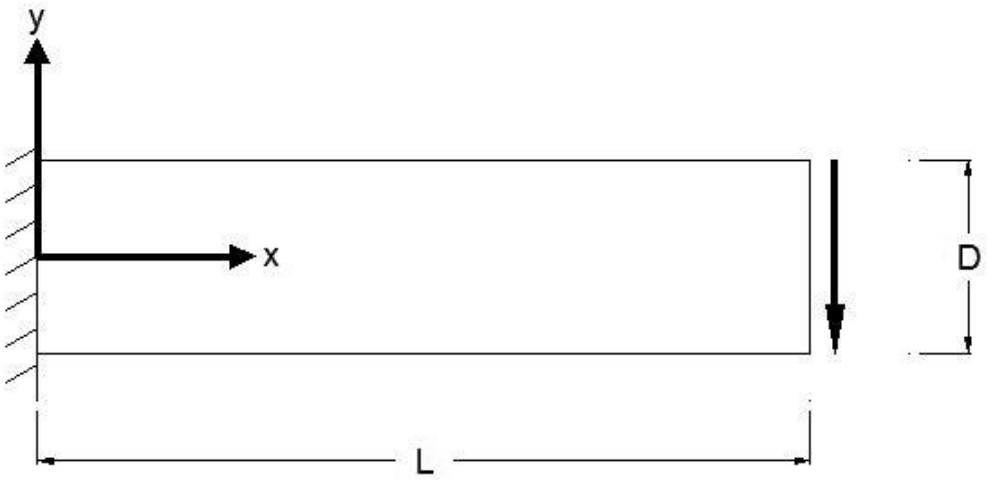


Figure 5.1 – Cantilever beam problem with applied traction at end.

Figure 5.2 – Cantilever model with parabolic applied traction at end.

Table 5.1 – Parameters for a 2D cantilever beam

Physical Parameter	**Variable Name**	**Variable Quantity**
Length	L	64 inches
Depth	D	16 inches
Applied Traction	P	10 ksi
Modulus of Elasticity	E	2000 ksi
Poisson Ratio	v	0.33

The node discretizations are given by figure 5.3 and the associated error is given by figure 5.4. The error estimate was based on the L_2 norm for the nodal displacements and the equation for the L_2 norm is given by

$$L_2 = \|\mathbf{u}\|_2 = \left(\sum u_i^2\right)^{1/2} \tag{5.2}$$

And the error estimator used to compute the error of the approximate solution with respect to the analytical solutions is given by

$$error = \frac{\left\|u_{exact} - u_{approximate}\right\|_2}{\|u_{exact}\|_2} \times 100\% \tag{5.3}$$

$$u_x(x,y) = -\frac{Py}{6EI}\left[(6L - 3x)x + (2 + v)\left(y^2 - \frac{D^2}{4}\right)\right] \tag{5.4}$$

$$u_y = -\frac{P}{6EI}\left[3vy^2(L - x) + (4 + 5v)\frac{D^2 x}{4} + (3L - x)x^2\right] \tag{5.5}$$

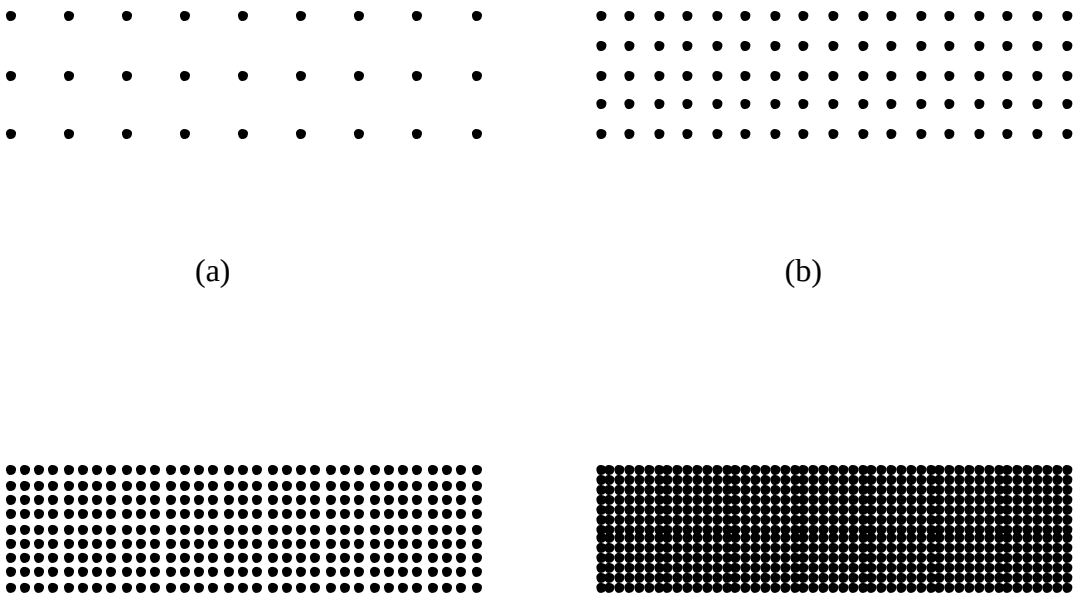


Figure 5.3. Global refinement for 4 models where (a) is 27 nodes, (b) is 85 nodes, (c) is 297 nodes and (d) is 637 nodes.

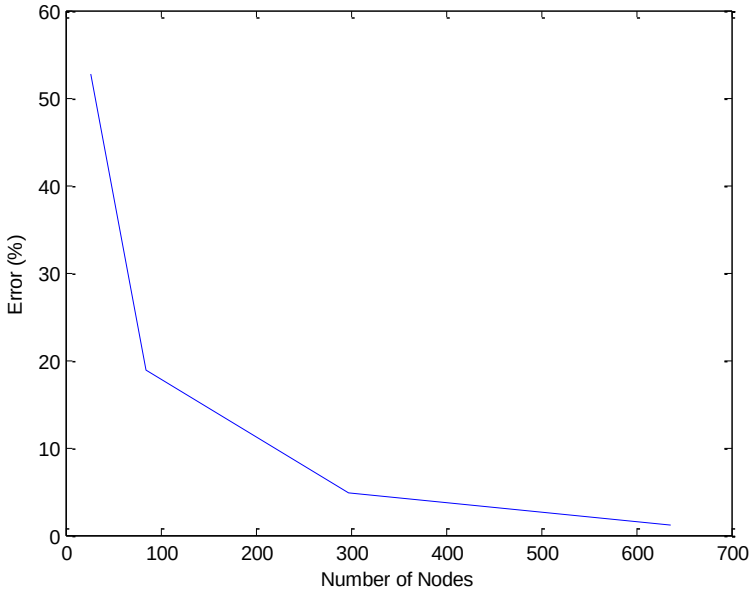


Figure 5.4. Cantilever beam relative error of approximate solution with respect to the analytical solution with L_2 norm.

Example – Plate with Hole with Global Refinement

In this example, a plate with a hole at the center will be studied as shown in figure 5.5. The model will only be a quarter-plate with the assumption that each quadrant of the model is equal and opposite as shown in figure 5.6. In the Theory of Elasticity, this class of problem is defined as the infinite plate problem and the analytical solution is given by Timoshenko's Theory of Elasticity [9] and is represented by equation (5.6, 5.7). It is not possible to model the entire problem due to an unrestrained rigidbody translation. In the model, rollers are used to enforce the condition that the y-axis is non-translational in the x-direction due to symmetry and the x-axis is non-translational in the y-direction as shown in figure 5.7. Table 5.2 gives the parameters of the model and variables as shown in the program. Figure 5.8 shows the relative error with respect to the exact solution and the convergence rate with global refinement.

Table 5.2 – Parameters for a 2D Plate with Hole

Physical Parameter	**Variable Name**	**Variable Quantity**
Length	L	5 inches
Applied Traction	P	1 ksi
Modulus of Elasticity	E	2000 ksi
Poisson Ratio	v	0.33
Radius of Hole	a	1 inch

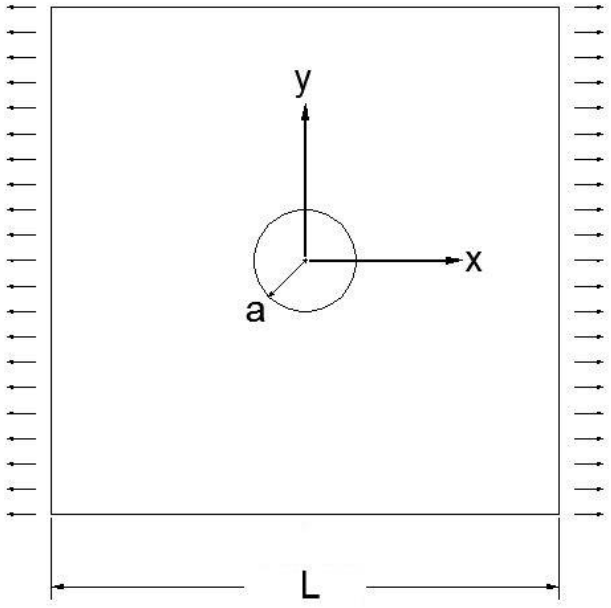


Figure 5.5. A plate with a hole at the center. L is the length of the plate and *a* is the radius of the hole.

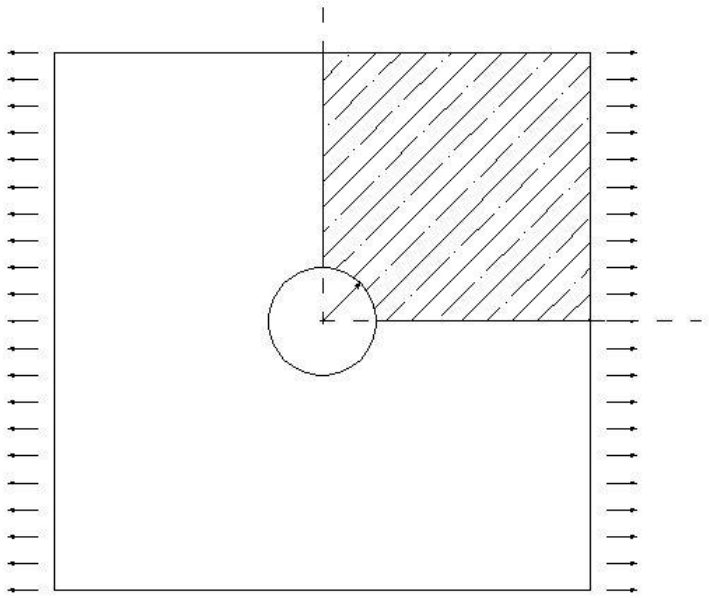


Figure 5.6. The modeling region from the original problem.

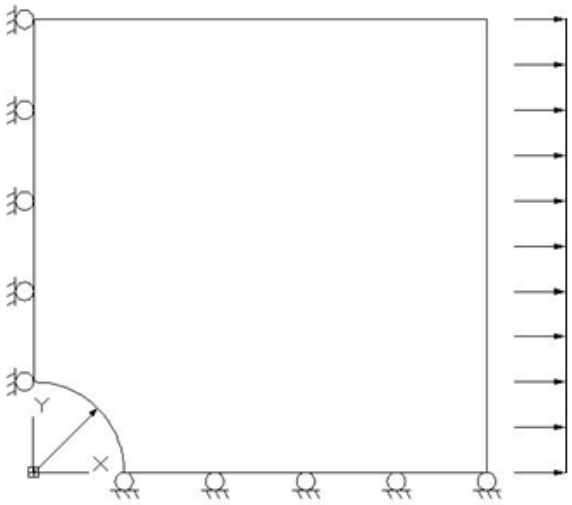
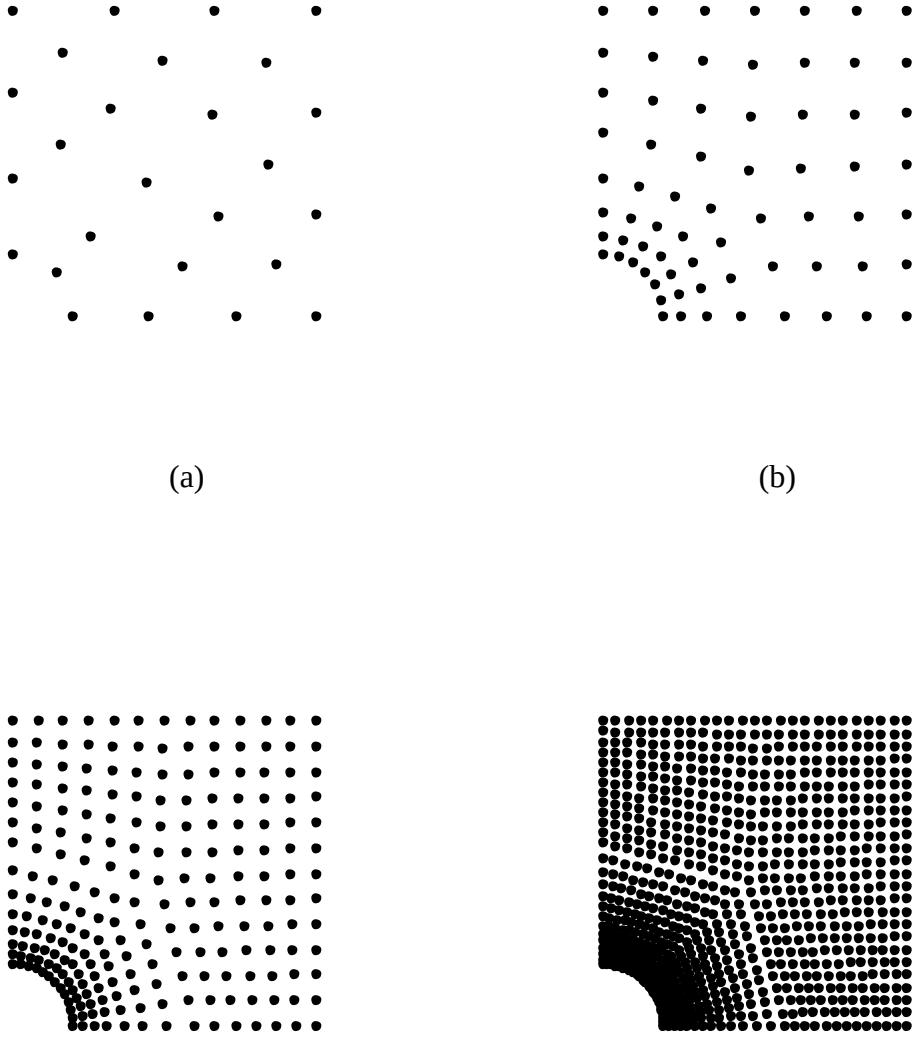


Figure 5.7. Model definition with appropriate boundary conditions to satisfy initial assumptions of quadrant symmetry.



(a) (b)

(c) (d)

Figure 5.8. Global refinement for 4 models where (a) is 26 nodes, (b) is 68 nodes, (c) is 224 nodes and (d) is 806 nodes.

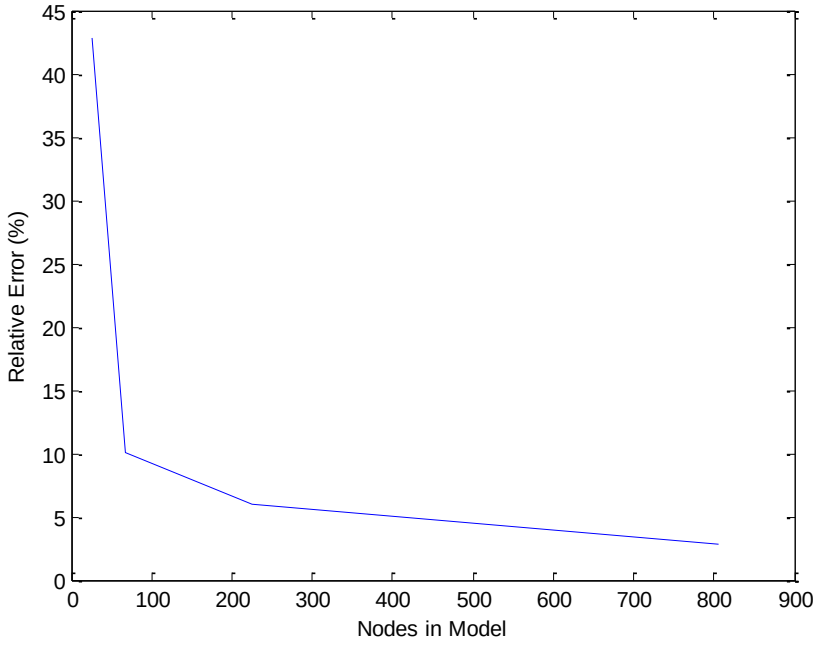


Figure 5.9. Plate with hole relative error of approximate solution with respect to the analytical solution with L_2 norm.

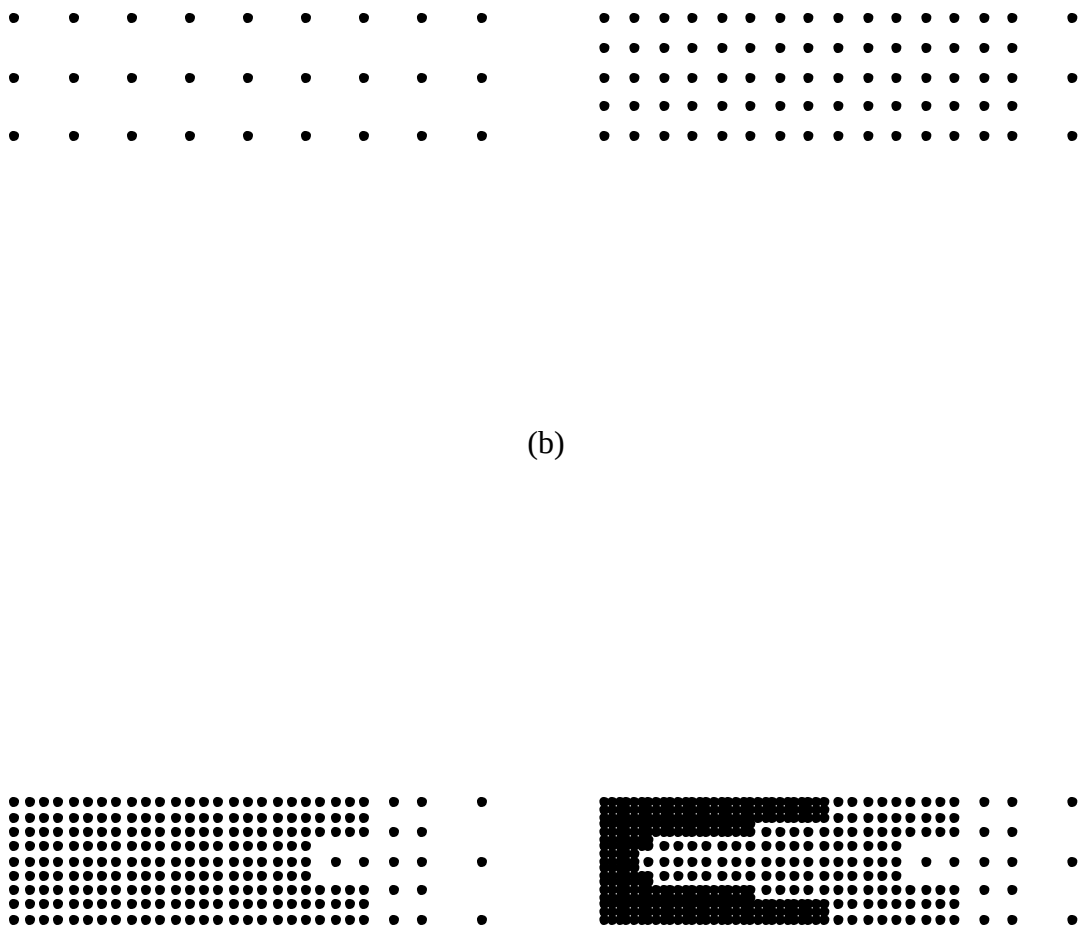
The analytic solutions to the plate with the hole is given by

$$u_r = r\left(\frac{K-1}{2} + \cos(2\theta)\right) + \frac{a^2}{r}(1 + (1+K)\cos(2\theta)) - \frac{a^4}{r^3 4G}\cos(2\theta) \quad (5.6)$$

$$u_\theta = \frac{1}{4G}\left(\frac{(1-K)}{r}a^2 - r - \frac{a^4}{r^3}\right)\sin(2\theta) \quad (5.7)$$

Example – 2D Cantilever Beam with Local Adaptivity

In this section, the 2D cantilever beam problem will be presented with the adaptive approach to show solution convergence without the need of global refinement. All the same parameters will be used, and the adaptivity will be based on the solution of the initial node discretization model containing 27 nodes.



(a) (b)

(c) (d)

Figure 5.10 – Node discretization for the adaptivity model where (a) is 27 nodes, (b) is 78 nodes, (c) is 228 nodes and (d) is 444 nodes.

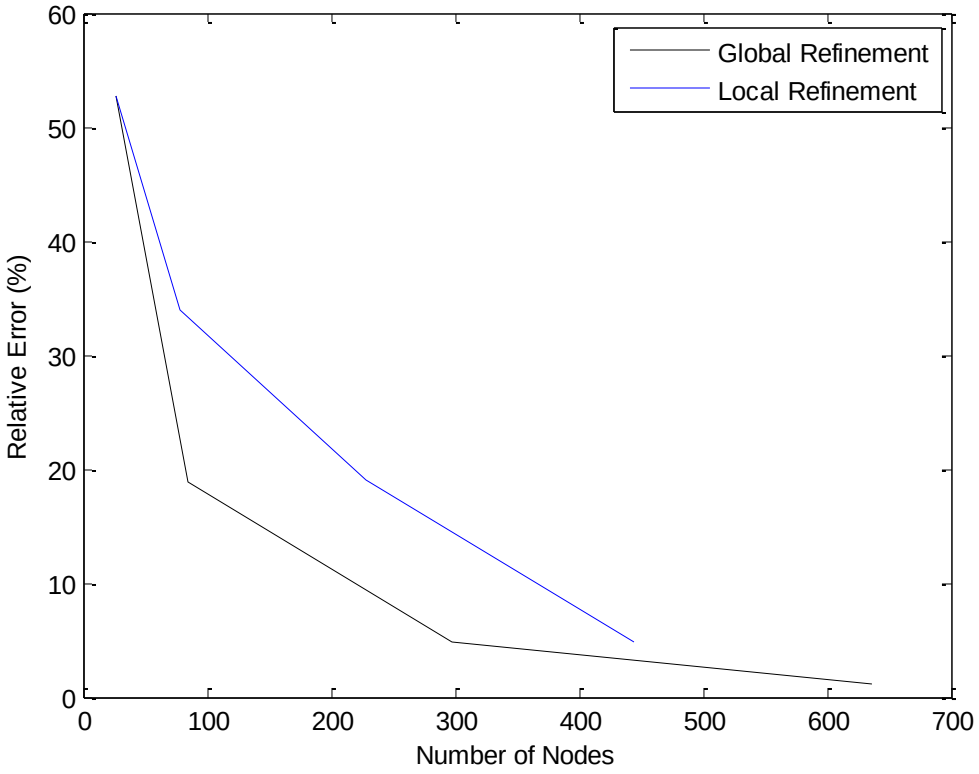
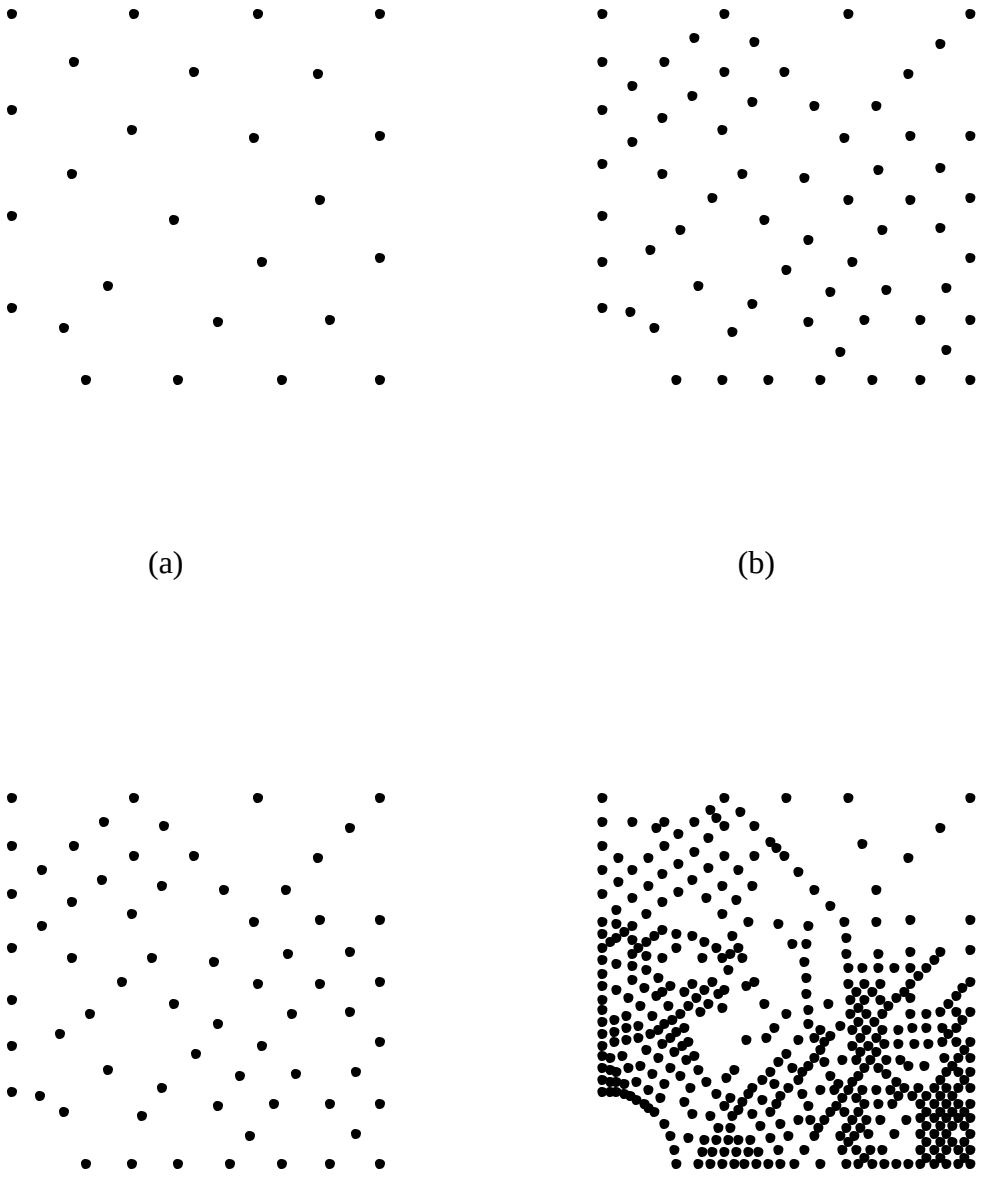


Figure 5.11 – Relative error of cantilever beam problem with local node adaptivity.

Example – 2D Plate with Hole with Local Adaptivity

In this section, the 2D plate with hole problem will be presented with the adaptive approach to show solution convergence without the need of global refinement. All the same parameters will be used from before and the adaptivity will be based on the solution of the initial node model consisting of 26 nodes.



(a) (b)

(c) (d)

Figure 5.12 – Adapted models where (a) is 26 nodes, (b) is 67 nodes, (c) is 188 nodes and (d) is 447 nodes.

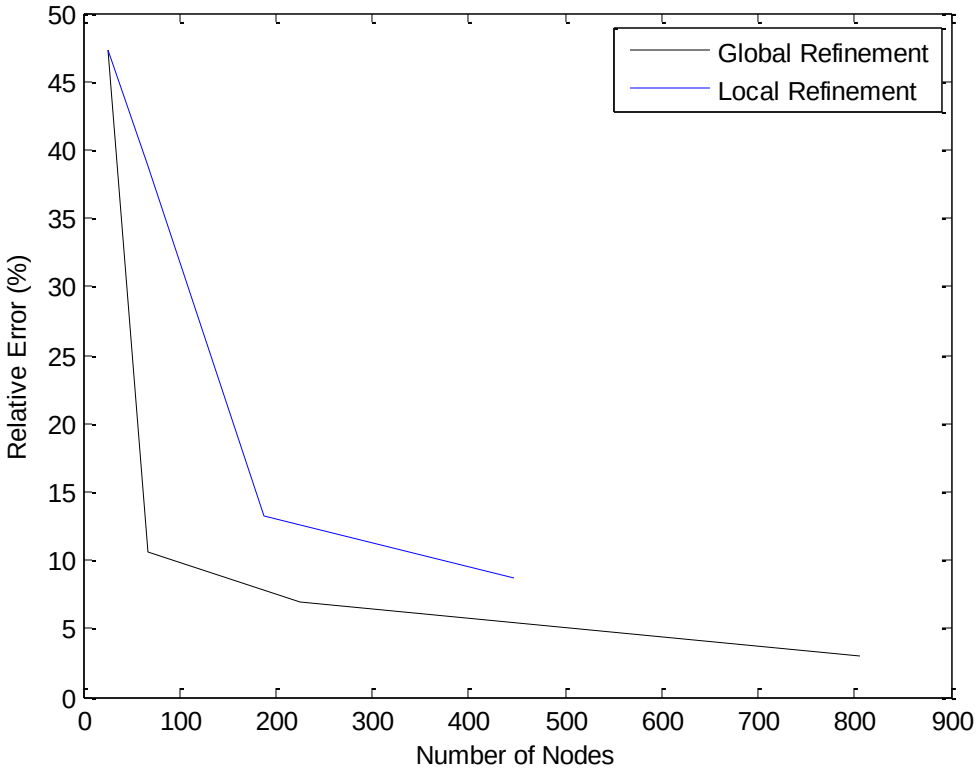


Figure 5.13 – Plate with hole relative error based on adaptive process.

CHAPTER 6

DISCUSSION ON THE LOCAL INFLUENCE PARAMETER

In this section, the influence parameter will be discussed. As mentioned in chapter 2, each node possesses a feature known as the influence parameter. The influence parameter is a radius in which is defines a local influence domain. For the 2D case, the influence domain is a circle and for the 3D case the influence domain is a sphere. The objective of the influence parameter is to be large enough to capture the effects of surrounding nodes to ensure the invertibility of the moment matrix but not too large or the effects of relatively distant nodes could cause the solution to diverge. Kim and Kim [9] devised a dilation function scheme for point collocation meshfree methods, however, their algorithm tends to be unstable for highly irregular node distribution patterns. When implementing their method in an adaptive algorithm, the solution becomes highly sensitive to the 3 input parameters and produces large oscillating errors in a refinement process. In other meshfree methods that implement the MLSM, others have defined the influence parameter to be a constant value, but large enough to satisfy the invertibility criteria such as that in [1].

As a part of this study, an algorithm was developed to compute the influence parameter for each node to ensure the invertibility of the moment matrix. The algorithm is a node counting algorithm in which the influence parameter must satisfy one criteria and that is the number of required nodes which is given by the following.

$$R = C \cdot \frac{(m+N)!}{m!\,N!} \tag{1}$$

Where R is the number of required nodes, *C* is a density coefficient, m is the highest order polynomial and N is the number of spatial dimensions. The algorithm is an iterated process where the initial influence parameter value is the distance to the closets node then an iteration process starts to increase the influence value process until the required number of nodes within the influence parameter is reached.

CHAPTER 7

SUMMARY AND CONCLUSIONS

Meshfree adaptivity methods for point collocation methods are studied. Also a nodal influence parameter algorithm is presented and compared with the dilation parameter function by Kim and Kim [9]. The PDA is different than other point collocation methods because the MLSM derivative approximation is based on a minimized residual functional. Two methods for nodal adaption was presented. The first method was based on a bisection approach in which a node was added between two nodes that resulted in a high change in the gradients. This method is applicable for structured nodal discretizations and rectangular node patterns. The second method is an interpolation approach in which the user defines an acceptable change in gradient and the algorithm adds an appropriate amount of nodes between two high gradient nodes so that the change in gradients are less than or equal to the defined changed in gradient. Several examples were presented ranging from scalar field problems to vector field problems with mixed boundary conditions to show that the proposed adaptive methods allow the approximate solution converge to the analytical solutions, exponentially.

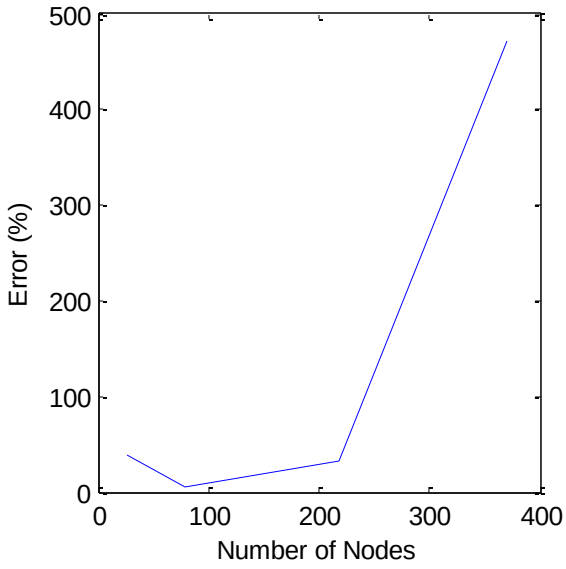
Further Studies

Adaptivity for meshfree methods pose a great potential for modeling large scale problems with limited computational resources. Adaptivity for meshfree methods require more investigation and development for the local influence parameter. In this study, several

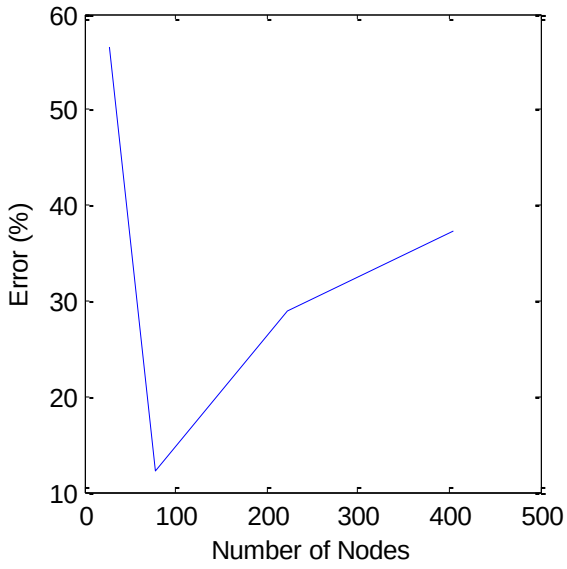
comparisons have been performed to show that adaptivity can be sensitive, and in some cases, highly unstable based on the influence parameter. Figure 9.1 shows the relative error as the number of nodes increase. Recall the proposed influence parameter algorithm requires the user define a radial step size for the algorithm to iteratively increase the radius until the node density is satisfied. Each error plot in figure 7.1 shows the effects of variable radial step sizes. In 7.1 (a), the step size is a constant 2 for each refined model. In 7.1 (b), the step size is a constant 1 for each refined model. In 7.1 (c), the step size is initially 0.5 and decreases linearly by 0.1 until the end of the refinement process. In figure 7.1 (d) the radial step size is the square root of 1 to account for nodes along a square diagonal. These are just a few examples of how significant the effects of the radial step size are in the refinement process.

Other mathematical approaches can be used for the adaptivity process. In meshed methods, the 3 types of adaptivity approaches are H-adaptivity, P-adaptivity and L-adaptivity. For meshfree methods, H- and P-adaptivity is generally accepted, however the L-adaptivity is not. Recall L-adaptivity for meshfree methods is the process of moving nodes to have better coverage in regions of high gradients, but this is not acceptable in meshfree methods because the initial points are typically defined at locations where information is critical such as places where boundary conditions exist or tractions are applied. Certain points cannot be moved also due to the node density criteria and having irregular influence parameters that could cause instability in the adaptive methods.

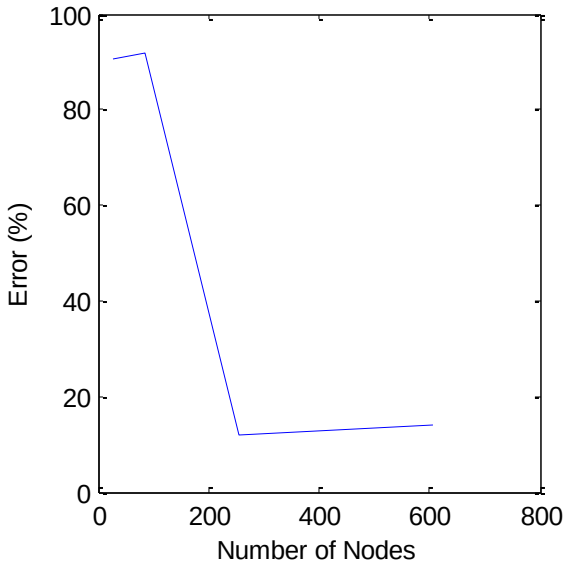
Adaptivity for meshfree numerical methods are promising for largescale computing and modeling. Further applications that are being investigated are moving boundary problems, phase transition problems and discontinuity problems.



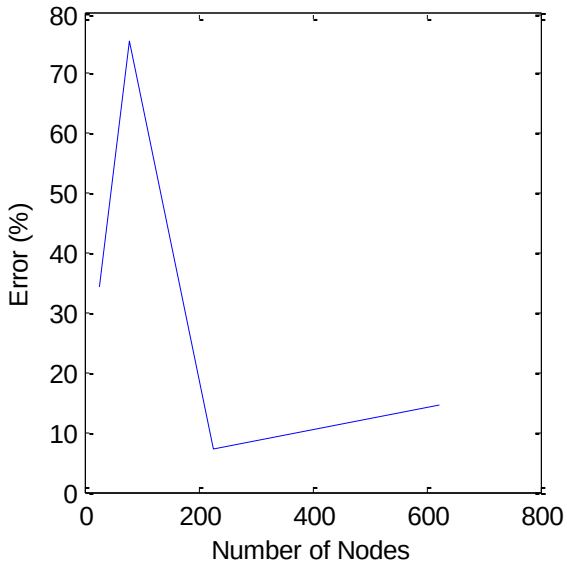
(a)



(b)



(c)



(d)

Figure 7.1 – Relative error as a function of node count based on different influence parameters. (a) is the solution based on the proposed dilation function by [9], (b-d) is the proposed influence parameter based on variable K values.

REFERENCES

1. T. Belytschko, Y. Lu, L. Gu, “Element-Free Galerkin Method”, International Journal for Numerical methods in Engineering, 37:229-256 (1994).
2. Y. Yoon, J-H Song, “Extended Particle Difference Method for Weak and Strong Discontinuity Problems Part I.”, Computer Methods in Mechanics, 53:1087 – 1103 (2014).
3. J. Oden, S. Prudhomme, “Goal-Oriented Error Estimation and Adaptivity for the Finite Element Method”, Computers and Mathematics with Applications, 41:735-756 (2001).
4. T. Rabczuk, T. Belytschko, “A Three-Dimensional Large Deformation Meshfree Method for Arbitrary Evolving Cracks”, Computer Methods in Applied Mechanics and Engineering, 196:2777-2799 (2007).
5. T. Rabczuk, T. Belytschko, “Adaptivity for Structured Meshfree Particle Methods in 2D and 3D”, International Journal for Numerical Methods in Engineering, 63:1559-1582 (2005).
6. L. Ling, “An Adaptive-Hybrid Meshfree Approximation Method”, International Journal for Numerical Methods in Engineering, 89:637-657 (2012).
7. B. Kee, G. Lui, C. Lu, “A Least-Square Radial Point Collocation Method for Adaptive Analysis in Linear Elasticity”, Engineering Analysis with Boundary Elements, 32:440-460 (2008).

8. C. Augarde, A. Deeks, “The Use of Timoshenko’s Exact Solution for a Cantilever Beam in Adaptive Analysis”, Finite Elements in Analysis and Design, 44:595-601 (2008).
9. D. Kim, H. Kim, “Point Collocation Method Based on the FMLSRK Approximation for Electromagnetic Field Analysis” IEEE Transactions on Magnetics, 40 (2004).
10. Timosheko, S., and J. Goodier. “Theory of Elasticity”, McGraw-Hill, 1970. Print.