Meshless Methods and Kohonen Neural Network for Electromagnetics design

by

Rajeev Das

A dissertation submitted to McGill University in partial fulfillment of the requirements of the degree of Master of Engineering in

August 2009

Abstract

The purpose of this research work is to implement a method, namely the meshless method, to represent Electromagnetic field problems. Meshless methods are currently being widely researched and are in a rapid development stage, this work verifies whether it is viable to use this method for low frequency problems. The benefit that this method will demonstrate is that it will bypass the mesh generation process involved in a well established technique called the finite element method presently being used in most of the computer aided design softwares available in the market, thereby reducing the computation cost. The other objective of this work is when there is a solution from the meshless method in place; it then incorporates a routine that re-arranges the nodes involved in the meshless process to minimize the error in the solution. This work investigates the ideas from a self-organizing feature map, specifically the Kohonen network, to determine whether such an approach is feasible enough to drive the nodes involved in the meshless approach to the regions where they will assist in obtaining the most accurate solution.

Résumé

Le but de ce travail de recherche est d'exécuter la méthode sans maille afin de représenter des problèmes de Champ électromagnétique. Les méthodes sans maille sont presentement examinées en détail et sont dans un étape de développement rapide. Ce travail vérifie s'il est réalisable d'utiliser cette méthode pour les problèmes de basse fréquence. L'avantage que cette méthode démontrera est qu'elle évitera le processus de génération de maille, un processus impliqué dans une technique bien établie appelée la méthode des éléments finies étant à présent utilisé dans la plupart des logiciels disponibles dans le marché de conception assistée par ordinateur, en réduisant ainsi le prix de compte. L'autre objectif de ce travail consiste en ce quand il y a une solution de la méthode sans maille dans l'endroit; elle incorpore alors une routine qui réarrange les noeuds impliqués dans la méthode sans maille de minimiser l'erreur dans la solution. Ce travail enquête sur les idées d'un plan de caractéristique auto-organisée, spécialement le réseau Kohonen, afin de déterminer si une telle approche est assez réalisable pour conduire les noeuds impliqués dans l'approche de la méthode sans maille aux régions où ils aideront à l'obtention de la solution la plus exacte.

Acknowledgements

I would like to thank my advisor Professor David A. Lowther without whose guidance and patience this work would not have been possible. He introduced me to the world of meshless methods and his encouragement through out this work provided me a rewarding learning experience.

I express my gratitude to my father and my aunts in believing me and for their affection and support. Without their help I would not have achieved all that I have done so far.

Thanks to my fellow graduate students from the CAD Lab for their support in these two years. Thanks in particular to Mr. Prakash Paul, Miss Maryam Golshayan, Mr. Hussam Maleh, Mr. Jun Ouyang and Mr. Adrain Ngoly. I would also like to thank all my friends for being there when I needed them. Special thanks go to my friend Miss Malika Meghjani for being a source of inspiration and ensuring that I had a life apart from academia.

I dedicate this work to the memory of my mother; you were and will be missed forever and to my uncle Mr. Dilip Das who passed away last year.

Table of Contents

CHAPTER 1

1 Introduction

1.1 Motivation and Overview

In today's world, where devices like television, radio, internet, microwave ovens, mobile telephones, electrical generators, computers and many more have become a necessity, life without them would be unthinkable for the present generation. Each of these examples is some form of electromagnetic system and is used in a broad range of situations. We can say without any qualms that we are heavily dependent on electromagnetic devices for a comfortable life. Because of this profound role, the study of electromagnetics becomes imperative.

The understanding of electromagnetic phenomena is derived from the electromagnetic field theory that describes the interactions between electric charges at rest and in motion. The theory explains such interactions through Maxwell's equations, which are a system of coupled partial differential equations that relate sources to the electromagnetic fields and fluxes. The possible ways of obtaining knowledge on the properties of electromagnetic systems can be classified into: a) theoretical methods and, b) experimental investigations. Theoretical methods can further be categorized into: i) analytical approaches and ii) numerical approaches.

In case of analytical techniques the equations, which have to be considered for a realistic description of the processes, are not often solvable for practical problems. In order to use an analytical approach one would have to simplify the equations and this often leads to inaccurate or inapplicable results. In terms of experimental investigations, the aim is to acquire system information by means of tests and this presents a lot of constraints such as the measurement of real objects becoming difficult in scenarios where the dimensions are either too small or too large; and such investigations require a considerable amount of time. The remaining approach, i.e. the numerical solutions, provides us with a fair degree of accurate approximation that overcomes the hurdles presented in the scenarios mentioned above. Using this approach coupled with the advent of computers, the ability to solve Maxwell's equations has changed profoundly and the advantages are manifold e.g., results can be obtained faster compared to the times when engineers had to prepare data elements on a piece by piece basis and parameter variations are easily realizable. Numerical simulation techniques have become an established self-contained scientific discipline and the solution of electromagnetic field problems using them has been a subject of research for almost half a century.

As stated earlier, determining device performance primarily involves solving the complex partial differential equations (PDEs) that govern the electromagnetic phenomena over the region of interest. One of the most prominent techniques that has been developed over the years and is now widely applied in this field is the finite element method.

It provides the advantage of being versatile enough to solve complex geometrical problems along with the capability of solving linear and non-linear problems with a great degree of accuracy. However, the finite element method does have a few shortcomings; the predominant one being the high cost of creating the meshes required for solving. Further, when adaptation is included in a system and is implemented by modifying the mesh to add new degrees of freedom, the computation cost increases significantly. For this reason, attention has been paid in recent years towards alternate methods where one can avoid the mesh generation completely.

An alternative methodology, with significant potential in terms of reducing the computation cost, that is currently being widely researched and is in a rapid development stage is the Meshless method (MM) [1], [2], [3], [4]. As the name suggests it does not require generating a mesh and thus bypasses the major drawback of the finite element system. Ideally no predefined mesh is necessary (some versions, may require a mesh to perform integration, refer to section 2.2.3) at all throughout the process of solving a problem involving an arbitrary geometry governed by the Maxwell's equations subject to a variety of boundary conditions. These methods originated over thirty years ago and began with the smooth particle hydrodynamics (SPH) method developed by Leon Lucy, Bob Gingold and Joe Monaghan who used it for modeling astrophysical phenomena such as exploding stars and dust clouds[5].

Instead of mesh generation, as done in FEM, the approach involves distributing nodes, referred to as field nodes, within the problem domain as well as on its boundaries to represent the problem and its boundaries. The later stages, involving the creation of shape

functions, the generation of a discretized set of equations and then solving for the field variables will be discussed in more detail in Chapter 2.

Continuing with the discussion on FEM, certain aspects of it should be highlighted in order to appreciate their relevance in the meshless approach later. The usual finite element analysis would proceed from the generation of a mesh and basis functions to the creation of a solution. The solution is dependent on the manner in which the analysis of the problem is done for instance the way boundary conditions are applied and the error estimation, generally the latter is used as means of feedback to obtain a result with greater accuracy. Error estimation typically requires the generation of a second solution on a finer mesh or a different approach such as the use of higher degree of shape functions (p-refinement) and a comparison of the two solutions. With the increase in speed and storage capabilities of computers nowadays, many problems are solvable with great accuracy by simple local refinement (without going for re-meshing over the entire domain) or relocating a mesh (r-refinement). The motive is to use low degrees of freedom in the problem without compromising the accuracy of the approximated solution. For this reason, the concept of "a posteriori error" estimation and adaptive mesh refinement has been developed in FEM. A typical adaptive algorithm will include the steps of solving the problem, a measurement of the error and then re-meshing and solving again until the error is below a prescribed quantity. An "a posteriori" error estimate provides the accuracy feedback that is necessary to terminate the adaptive procedure. One of the most popular methods in determining this is with an enrichment indicator. In this approach, the

basic assumption is that large errors come from regions where the local error estimate is large and this is where the mesh should get concentrated. Correspondingly, the mesh is coarsened where the error estimate is small. These ideas can be extended to MMs, except now, instead of meshes, we will have nodes to play with in order to ensure results of desired accuracy.

The present work proposes to have an adaptive system in a meshless model. The approach that has been incorporated for obtaining the optimal solution draws its ideas from the Kohonen Self Organizing Feature Map – a form of neural network. The Kohonen networks have the ability to learn autonomously by extracting statistical features (without supervision) from the input pattern population. The input pattern will be developed on similar lines to the enrichment indicator stated earlier in order to have an inclusive model in place that will comprise a functioning Kohonen model working synchronously with the meshless model. It can be viewed as an approach for optimization that will not involve any increase in the degrees of freedom by the introduction of additional nodes in the system but will involve positioning, or arranging, a fixed set of nodes using the Kohonen network in such a manner that it will minimize the global and local errors.

One of the main objectives of this work is to develop a flexible method for solving electromagnetic field problems that will bypass the rigidities involved in generating meshes presented by FEM. On obtaining results from such a method, the next objective will be to investigate whether the ideas of a self-organizing feature map particularly that

of Kohonen network can be used to obtain the best solution with the available number of nodes.

To summarize the outline of the structure of this dissertation, the next chapter i.e., chapter 2 section 2.1 will discuss the preliminaries of radial basis functions (RBFs) which will be used as the basis function in the meshless approximation technique to be considered, this chapter will also provide an introduction and a discussion on the theoretical as well as the mathematical aspects involved in meshless methods. Chapter 3 will have details regarding neural networks and aspects concerning the Kohonen neural network. Chapter 4 illustrates the approach involved in designing the meshless and Kohonen models in detail and then presents the algorithm being proposed in this work to integrate these two models and the results. Chapter 5 presents the conclusion of this work and a discussion about future work.

I have used Matlab version R2007b along with a system specification of Intel Pentium M processor 1.60 GHz, RAM of 512 MB and Windows XP operating system SP 2 for designing the models and computation tasks involved in this work.

CHAPTER 2

2.1 Radial Basis Function

2.1.1 Preliminaries on radial basis function

The meshless method is primarily a numerical technique for approximating functions. Numerical methods such as the finite difference method (FDM), FEM or the finite volume method (FVM) first involve discretization wherein the creation of some sort of mesh is required for computation and, finally, all of them approximate the unknown function. There are a number of approximation techniques available which can broadly be classified into - 1) Least square based approximation 2) Quasi interpolation for approximation 3) Interpolation using polynomials or piecewise polynomials, or radial basis functions for approximation [6], [7]. Radial basis functions when used for interpolation as explained later in the text, involve translates of the basis function, for the construction of a multivariate function from the data points. As we will find later, the MM, when using radial basis function, requires reconstruction of a multivariate function from a scattered data set which can be either be structured or unstructured data. This aspect provides us with a reason to highlight the properties of multivariate scattered data interpolation.

Scattered data interpolation involves finding a continuous function f such that for given a data (x_j, y_j), $j = 1, \ldots, N$, with $x_j \in \Re^n$, $y_j \in \Re$, $f(x_j) = y_j$. A convenient approach to

represent such a function is by assuming that the function f is a linear combination of certain basis functions B_k,

$$f(X) = \sum_{k=1}^{N} ck\, B_k(X), \qquad X \in \Re^n \tag{1}$$

where ck denotes the coefficient. The above interpolation problem will lead to a system of linear equations of the form Ax = D where A denotes the interpolation matrix, x the vector of unknowns and D is a vector of known values or constants. The solution to the linear equations will exist if and only if the matrix A is non-singular. In a univariate environment, one can interpolate to arbitrary data at N distinct data sites using a polynomial of degree N-1, but in the case of a multivariate condition the system will fail as evident from the Mairhuber – Curtis theorem (refer to appendix A). The theorem highlights the fact that if we want to have a well posed multivariate scattered data interpolation problem then the basis should depend on the data locations.

In order to construct a function, f, that interpolates a given set of uniformly scattered data sites, for instance in a unit cube, one would use piecewise polynomials to avoid any oscillations, the simplest approach in this scenario being through the use of continuous piecewise linear splines (spline functions of order 2). Such an approach involves shifting of the absolute value function to data sites, the function, f , will be of the following form

$$f(X) = \sum_{k=1}^{N} c_k \|X - X_k\|, \qquad X \in [0,1] \tag{2}$$

where ||.|| denotes some norm. The above equation clearly indicates that the basis functions $B_k = |. - X_k|$ are dependent on the data sites. The points X_k to which the functions are shifted are usually referred to as centers and, generally, they are picked to coincide with the data sites to simplify the analysis. The functions B_k are radially symmetric about their respective centers thereby constituting an example of a radial basis function. While dealing with field problems later in this work, the field values at the nodes or the data points involved in the MM will not be known, infact MM will use interpolation technique and the values at the boundaries of the problem domain to estimate the field values at the data points.

The above discussion provides us with a background that highlights the intricacies involved while dealing with multivariate scattered data interpolation and they in fact guide the basic criteria that result in the use of RBFs for approximation, which are 1) when the function to be approximated (the approximand) depends on many variables or parameters; 2) the data are scattered in their domain. The manner in which RBFs and meshless methods achieve this is by composing a univariate basic function with a Euclidean or P-norm and thereby converting a problem involving many dimensions into one that we can say is similar to a one-dimensional problem.

Definition 1**:** A function $\phi: \Re^n \rightarrow \Re$ is called a radial provided there exists a univariate function $\Psi : [0, \infty) \rightarrow \Re$ such that

$$\phi(x) = \Psi(r) \text{ where } r = ||x|| \tag{3}$$

where ||.|| is a norm on $\Re^n$. As defined above, Radial Basis Functions (RBFs) are a good means to approximate multivariate functions. Given data in **n** dimensions that consist of data sites $\xi \in \Re^n$ and function values $\mathbf{f}_\xi = f(\xi) \in \Re$, we seek an approximant **s:** $\Re^n \rightarrow \Re$ to the function (the approximand) **f:** $\Re^n \rightarrow \Re$ from which the data are assumed to originate. Here **n** > 0 is the dimension of underlying space. One can also be restricted to a domain $D \subset \Re^n$ and if this D is prescribed one seeks an approximation **s:** $D \rightarrow \Re$ only. We can consider $f(\xi)$ as the explicit function values we know of **f,** which itself is unknown, or at least unavailable, for arbitrarily large numbers of evaluations. A good approximate of $f(\xi)$ can be obtained by distributing large numbers of nodes (data points) in the required and precise areas of the problem domain thereby obtaining a solution close to the "true" solution. An estimation of the accuracy of the approximant can be obtained by formulating a four or five point stencil approach, a technique that is similar to what is done in a FDM solution by taking the average of the unknown values around a particular point and then comparing it with the value computed at that point. One has to postulate the existence of $f(\xi)$ so that **s** and **f** can be compared and the quality of the approximation be estimated. In an interpolation approach to approximation we explicitly require $\mathbf{s}|_\Xi \cong \mathbf{f}|_\Xi$ where $\Xi \subset \Re^n$ is the discrete set of data sites (nodes). It is desirable to be able to perform the interpolation without any further assumptions on the shape of Ξ so that the nodes can be scattered and these ideas can easily be applied when the node distribution is uniform and it will be easier to handle the gridded data where we will have equal spacing or same step size. However we at times do consider $\Xi = (h\mathbf{Z})^n$; h being a positive step size and **Z** being integers thereby we have step sizes

with some integral multiple attached to them which help us in analyzing properties of approximation easily.

The RBF approximant **s** is usually composed of finite linear combinations of translates of radially symmetric basis functions say $\Psi(||.||)$. Radial symmetry implies that the value of the function only depends on the distance of the argument from the origin and any rotation thereof makes no difference to the function value. The translates are to the point $\xi \in \Xi$, from where we consider linear combinations of $\Psi(||.- \xi||)$. ξ are called the centers and the space **S** is dependent on the set Ξ. this observation again becomes clearer through the Mairhuber – Curtis theorem (refer to appendix A). The simplest example is given by

$$\mathbf{S} = \left\{ \sum_{\xi \in \Xi} \lambda_\xi ||. - \xi|| \;\middle|\; \lambda_\xi \in \Re \right\} \tag{4}$$

The radial basis function $\Psi(r) = r$ (linear radial basis function) where the radial symmetry stems from the norm $||.||$ and it is shifted by the centers ξ. RBF spaces are spanned by translates

$$\Psi(||. - \xi||), \qquad \xi \in \Xi$$

The general form of the radial basis approximants is.

$$s(x) = \sum_{\xi \in \Xi} \lambda_\xi \Psi(||. - \xi||), \; x \in \Re^n \tag{5}$$

with real coefficients λ_ξ

Examples of radial basis functions $\Psi(r)$ are

Multiquadrics $\Psi(r) = \sqrt{r^2 + c^2}$ c a positive parameter. (6)

where c is a shape (user-defined) parameter.

Thin plate splines $\Psi(r) = r^2 \log r$ (7)

Linear radial basis $\Psi(r) = r$ (8)

Gaussian $\Psi(r) = e^{-(\alpha r)_2}$ (9)

α is a positive parameter which controls the smoothness properties of the interpolating function.

For the thin-plate spline, and several other radial basis functions, a linear (generally low-order) polynomial has to be added to s with side conditions

$$\sum_{j=1}^{m} \lambda_{\xi j} = \sum_{j=1}^{m} \lambda_{\xi j} \xi_j = 0 \quad (10)$$

in order to be able to solve the interpolation equations uniquely; where m is the number of center points. In that case, the centers must not lie on a straight line, but may otherwise be arbitrarily distributed. For multi-quadrics, Gaussian and linear radial functions, among others, the extra geometric condition is not needed.

Before concluding the section on RBFs, we need to highlight two important aspects that affect the accuracy of the solution and the efficiency of solving problems involving RBF

interpolation and they are 1) conditionality and 2) convergence. A detailed discussion is beyond the scope of this presentation. The accuracy of the solution is dependant upon the condition number of the coefficient matrices that are generated using the RBF interpolants. The condition number can be adjusted through variable shape parameters but it comes with a catch, as there is a conflict between the numerical stability and the accuracy of the solution. In case of Multiquadrics or Gaussian, convergence can only be achieved at the cost of instability [8], [9], [10]. To study convergence and its related issues in RBFs, the works of J Duchon and Z. M Wu can be considered as a good starting point; it is difficult to generalize the convergence phenomena when we deal with different types of RBFs under the stationary and the non-stationary conditions (we deal with these conditions later in this thesis) as they behave in a slightly different manner. However, its study also brought to light the fact that as the spatial dimension of the problem increases, the convergence order also increases, and hence when the RBF collocation (which involves interpolation using RBFs) approach is used, fewer collocation points are required to maintain the same accuracy when compared with FDM and FEM to solve similar problems, in this scenario, the convergence implies the approach of the solution to a physical problem towards the “true” solution, which is dependent on the position and the number of nodes.

2.2 Meshless Methods

2.2.1 Introduction

As mentioned earlier, meshless methods are used to establish a system of algebraic equations for the whole problem domain without the use of a predefined mesh for solving partial differential equations (PDEs). Like other numerical techniques, they too require mathematical models that are generally expressed in terms of the field variables in the governing equations with proper boundary conditions (BCs) and/or initial conditions (ICs). The governing equations are usually a set of (PDEs) or integral equations (IEs) keeping in mind the latter are treated differently from the former. For dealing with problems involving PDEs, the methods use a set of nodes known as field nodes, distributed within the problem domain, as well as sets of field nodes on the boundaries, to represent the problem. This set of scattered nodes does not form a mesh, which means that no information on the relationship between the nodes is required, at least for field variable interpolation and when we are dealing only with linear problems. There are various flavors of meshless methods that have been developed over the years such as: smooth particle hydrodynamics (SPH) by Gingold and Monagham in 1977, the diffuse element method (DEM) by Nayroles in the year 1992, the element-free Galerkin (EFG) method proposed by Belytschko in 1994, the reproducing kernel particle method (RKPM) by Liu in 1995, the hp-clouds method in 1996 by Duarte and Oden and many more.[2],[5] They can broadly be classified into three categories: a) those based on collocation methods; they are truly mesh-free and do not require either a mesh structure or an integration procedure and are relatively easy to implement. However, they are less stable

and less accurate; b) those derived from the weak form, these are not considered truly mesh-free due to the fact that they require a background mesh to implement the integration; and c) thirdly those based on a local weak form, here the concept of local domains is used to formulate the weak form locally and to define the integration points [1], [2], [11]. To elaborate a bit more on the nature of the algebraic system involved in a meshless environment, i) as stated above, we have the strong form (SF) which is represented as a system of ordinary or partial differential equations in space, complemented by appropriate boundary conditions. This may be presented in integro-differential form reduced to algebraic equations, an example being the collocation method;

ii) the weak form (WF) wherein the governing PDEs with derivative boundary conditions are first transformed to a weighted integral equation that "relaxes" the strong form into a domain-averaging statement. The weak-forms are used to derive a set of algebraic system equations through a numerical integration process using sets of background cells that may be constructed globally or locally in the problem domain, the element-free Galerkin (EFG) method is one such approach;

iii) the variational form (VF) is presented as a functional whose stationary conditions generate the weak and strong forms [2]. Examples include the Petrov-Galerkin approach wherein we find variational statements.

The choice or approach of mathematical model will ultimately depend upon the given conditions for the system and the ability of a particular approach to handle them well and provide a good solution.

2.2.2 Procedures of Meshless methods

This section will present the details that are involved in the use of meshless methods. A typical meshless system will involve the following steps (Fig 1).

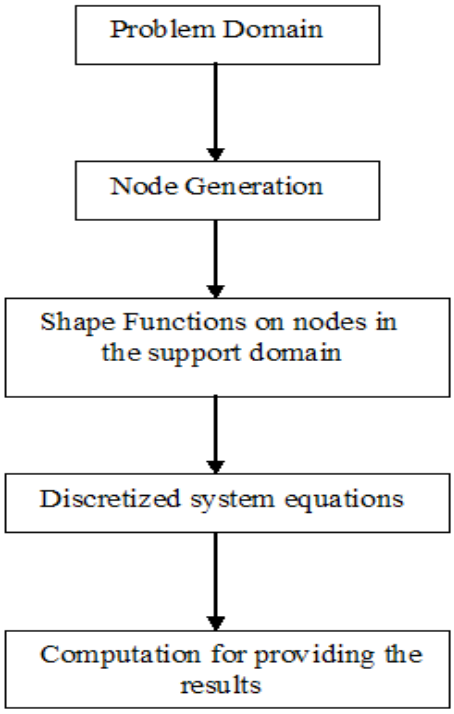


Fig 1

The basic steps involved in solving a problem using the meshless approach.

Node Generation: Node distribution can be performed arbitrarily over the problem domain provided the nodes are placed in the regions where the solution is required. In certain scenarios where it is convenient to have basis functions that can automatically adapt to changes in the nodal distribution such as in the case of fluid dynamics or of anisotropic supports , the nodes can be generated in a problem domain using triangulation

algorithms that are routinely available for 2D and 3D domains. The significance here is that the process of node generation can be fully automated and the analysis can be performed in a fully adaptive manner; such an approach is prevalent in a well established meshless technique known as the Natural Element Method (NEM) [12]. In terms of implementation for the present work, the uniform distribution was created easily by defining a fixed step size (spacing among the nodes). On identifying the boundaries and with the available number of nodes an iterative approach has been used to distribute them in the problem domain. In the case of a random node distribution we have considered the Halton sequence, (refer to appendix B).

Shape Functions: The approximation function is an essential feature of the meshless method. In this, the field solution is approximated by local shape functions, often but not always, using a form of radial basis functions.

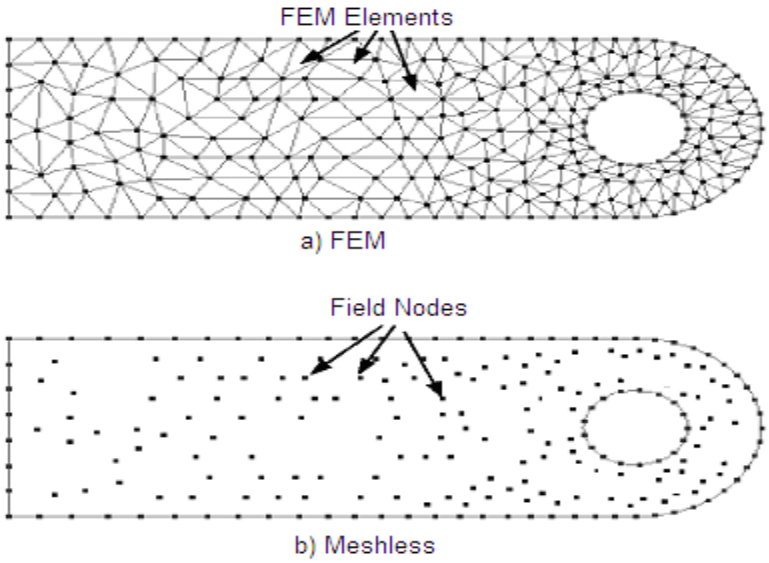


Fig 2

Domain representation using a) the FEM approach with its triangular elements in place b) random distribution of field nodes using the meshless approach with the same problem domain [2].

Each node, through its shape function, is considered to have a domain of influence, which could cover the entire problem space or might be limited to a local neighborhood. The field value at any point in space is then given by summing the contributions of all the nodes within whose domain of influence the point lies. The weight function, or basis function, plays an important role in the performance of the methods and is used in all varieties of meshless methods. In the construction of the shape functions, one of the concepts that we come across is the influence domains and the support domains. One would tend to interpret both of them as the same but there is a slight nuance.

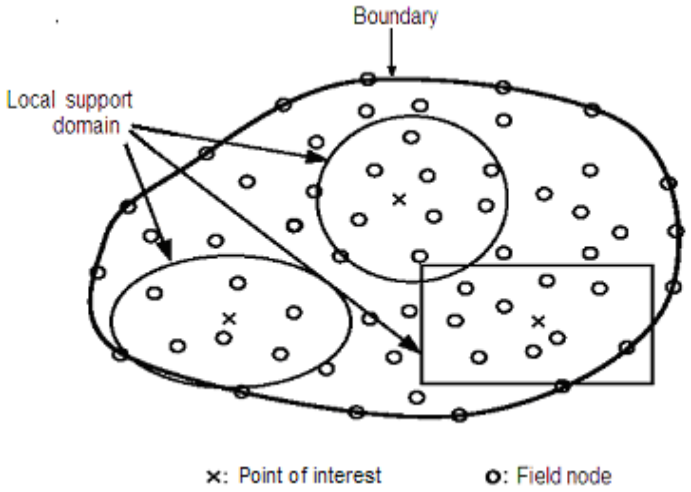


Fig 3

A meshless discretization of an arbitrary domain with the support domains [2].

The support domain defines which nodes in the problem domain are needed for the interpolation at a particular point x = (x, y). Nodes closer to x can be given more importance in the interpolation calculation than the nodes further away, in which case the support domain is said to be weighted. However, this is not always done. Different points can have support domains of different shapes and sizes, although typically the shape is

circular or rectangular. The influence domain is a similar concept except that in this the point (or the reference point) is not merely a point but is a node considered from the ones distributed in the domain. The influence domain is just how far a node (where the node is the center of a RBF) extends its influence, and it can be different for each node. The concept of a support domain works well when the nodes are distributed evenly whereas the influence domain works well even if the nodes are distributed very unevenly; but the latter is slightly more difficult to construct.

Disctretized system equations and the field variable solution: The discrete equations involved in a meshless method are formulated using the shape functions and the strong or weak form of the system equations. These equations are often written in a nodal matrix form and are assembled into the global system matrices for the entire problem domain. The solution is obtained using the available established numerical techniques used for solving systems of algebraic equations, including direct or iterative methods such as the Gauss elimination or the QR method etc.

2.2.3 Numerical implementations

In Chapter 1 some background concerning approximation using interpolation with radial basis functions were presented since this approach has been used in the present work. Before additional details are provided in terms of the manner in which the implementation has been handled, this section will briefly highlight three main numerical techniques that are widely used in the meshless world and also extensively applied in the field of electromagnetics, including the one that has been used in this work.

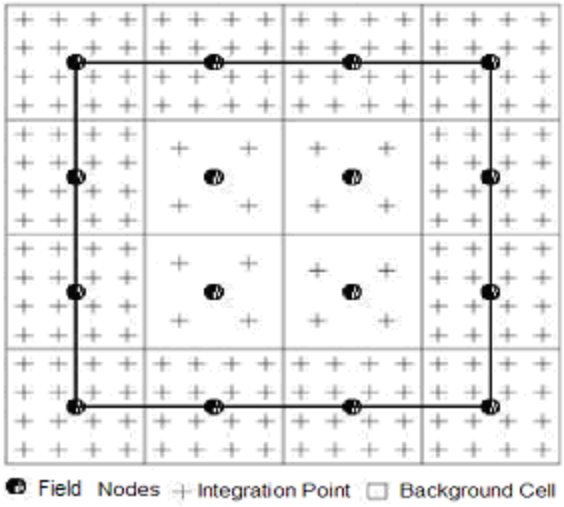


Fig 4

The Background Cell for the Galerkin weak-form using Quadrature Integration Scheme approach. Field nodes represent the problem domain. The background cell structure is used to evaluate the integrations in the weak form.

Element Free Galerkin (EFG) method: Meshless methods based on the global weak form usually use this approach. In the EFG method, the problem domain is represented by a set of distributed nodes. The moving least square (MLS) approximation is used to construct shape functions that use only on a group of arbitrarily distributed nodes in a local domain, which is a part of the global domain. As shown in Fig 4, the domain is divided into a latticed domain known as the background cells consisting of nodes. A set of background cells is used to evaluate the integrals that result from the use of the Galerkin weak form (the Quadrature Integration Scheme approach). Each cell becomes a unit of integration. Usually Gauss integration is applied in all cells. To have a better idea, the figure (Fig 4) is a representation of such an arrangement when the number of integration points is four at an inner cell, and sixteen at a boundary cell applied to a square domain. There are certain difficulties associated with this approach particularly in enforcing the boundary conditions [13],[14], besides this it also requires global numerical integrations for which

a global background cell structure has to be used, which adds more complexity. An alternative approach is the spatial integration of the Galerkin method, which was achieved by evaluating the integrals of the weak form only at the nodes suggested by Beissel and Belytschko in 1996. There is no need of a cell structure or background mesh in this approach and it is a truly meshless method. However, in certain cases it results in a spatial instability because of underintegration causing spurious modes of energy (modes with no physical basis that appear due to lower order of numerical integration) and providing a singular assembled stiffness matrix; this problem arises because the integration scheme at times leads to rank deficiency [15].

Petrov–Galerkin Method and Local Boundary Integral: Zhu in 1999, and Atluri and Zhu in 2000 proposed two kinds of meshless methods: the meshless local boundary integral equation (MLBIE) method and the meshless Petrov–Galerkin (MLPG) method. Both these methods use the MLS approximation to interpolate the solution variables, the MLBIE method uses a local boundary integral equation formulation, and the MLPG employs a local symmetric weak form. Here a local weak form over a local subdomain, which is located entirely inside the global domain, is used. Integrals in both methods are evaluated over regularly shaped domains and their boundaries. There is no need for a background mesh, and they are truly meshless methods. Here, too, the nodal shape functions from meshless interpolations, such as MLS, are highly complex in nature; and this makes an accurate numerical integration of the weak form difficult [4], [16].

Collocation Method: This technique has been in place for a long time. There are various forms of it in place such as the vortex method proposed by Chorin in 1973, the finite point method (FPM) by Oñate and others in the years 1996 and 1998, the hp-meshless cloud method by Liszka in 1996, the meshfree collocation method by Kansa in the year 1990 and other variants of collocation technique by Wu in 1992; Xu in the year 1999 and many more [2],[17]. The present work implements an approach similar to the Kansa method. In collocation techniques, the discretized equations can be obtained directly from the strong forms of PDEs governing the problem. The PDEs are discretized directly without using weak forms, and hence no numerical integration is required thereby making the process simple to implement, which is one of its major advantages.

However, two major problems are common to this technique. One being the issue of singularity of the moment matrix (coefficient matrix) arising in the process of function approximation. This can be avoided by using a matrix triangularization algorithm for a point interpolation method (PIM) that uses polynomial basis as suggested by GR Liu and Gu in 2001 and 2003 or by using the weighted least squares method as proposed by Krok and Orkisz in the year 1989, or as implemented in the Kansa approach by using RBFs. The Kansa method is a global collocation method that uses all the points in the problem domain, which leads to a fully populated system matrix [17]. As RBFs are used, the moment matrix is, in general, not singular.

The second issue arises due to the boundary conditions. Strong-form methods can produce accurate results for PDEs, when the boundary conditions are all of the Dirichlet type. The accuracy of the solution deteriorates drastically when there is any derivative

boundary condition, and the solution can end up being very unstable: small changes in the setup of the problem can lead to a large change in the solution. The discretized system equation behaves like an ill-posed problem in which errors introduced into the system are magnified in the output. Some of the common ways of rectifying such issues are a) using a set of fictitious points (FPs) that is added outside the problem domain along the derivative boundary, which leads to two sets of equations one for the derivative boundary condition and the other for the governing equation;

b) a Hermite-type collocation (HC) method in which additional derivative variables for the derivative boundary nodes are used to enforce the derivative boundary conditions;

c) a Direct collocation (DC) method where the derivative boundary conditions are discretized by collocation to obtain a set of separate equations that are different from the governing system equations. This approach has been applied in the present work.

2.2.4 Collocation Method

This section will restrict itself to presenting the collocation approach using the Gaussian form of RBF, as it has been used in the present work. Before presenting the mathematical aspects involved in it, a few properties of the Gaussian form of RBF will be provided in the following paragraphs.

Consider equation (9), $\Psi(r) = e^{-(\alpha r)_2}$, where α is a positive parameter and is known as the shape parameter. It is related to the variance σ of the normal distribution function by α^2

= $1/(2\sigma^2)$. The Gaussian with Euclidean distance function $\|.\|_2$ for any fixed center $\mathbf{x_k} \in \Re^n$, can be defined as

$$\phi(x) = e^{-(\alpha)_2 \| x - x_k \|^2_2}, \qquad \mathbf{x} \in \Re^n \tag{11}$$

or, we can write using equations (9) and (11)

$$\phi_k(x) = \Psi(\|x - x_k\|_2) \tag{12}$$

a smaller value of α (greater variance) will result in flat distribution whereas for increased α we have a peaked RBF, α plays an important role as highlighted in section 2.1.1 in terms of the accuracy and stability of the solution. There are two ways to set the shape parameter either by fixing the value of α for all the experiments with the Gaussians, known as a non-stationary approximation, or by scaling the shape parameter according to the spacing among the nodes so that we have peaked basis functions for densely spaced data and flat basis functions for coarsely spaced data. The selection of a good shape or optimal shape parameter can be performed in number of ways such as using a power function as the indicator or cross validation but the approach that is most widely used by practitioners is through trial and error.

In interpolation with data $\{Z_i, f_i\}$ where i = 1,…., N (the nodal points), $\mathbf{Z_i} \in \Re^n$ the values of f_i can be considered being sampled from a function *f:* $\Re^n \rightarrow \Re$ which is a linear

combination of certain basis functions, here the functions are RBFs $\Psi(r)$. Referring to equations (4) and (5) we will have an equation of the following nature

$$P_f(Z) = \sum_{k=1}^{N} x_k \, \Psi(||Z - Z_k||_2), \qquad \mathbf{Z} \in \Re^n \tag{13}$$

such that $P_f(Z_i) = f_i$ where i = 1,…., N which leads to a linear form of the system equation of the nature $Ax = f$, interchanging the L.H.S and R.H.S in (13) the matrix form is represented as:

$$\begin{pmatrix} \Psi(||Z_1 - Z_1||_2) & \Psi(||Z_1 - Z_2||_2) \ldots \Psi(||Z_1 - Z_N||_2) \\ \Psi(||Z_2 - Z_1||_2) & \Psi(||Z_2 - Z_1||_2) \ldots \Psi(||Z_2 - Z_N||_2) \\ \vdots & \\ \Psi(||Z_N - Z_1||_2) & \Psi(||Z_N - Z_1||_2) \ldots \Psi(||Z_N - Z_N||_2) \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix} = \begin{pmatrix} f(Z_1) \\ f(Z_2) \\ \vdots \\ f(Z_N) \end{pmatrix} \tag{14}$$

Where k varies from 1 to N indicating the total number of data points or nodes. x_k being the notation for the unknowns. Equation (14) shows the basic structure of the matrix that will be used in most of the scenarios later. In the collocation method, matrices of such form are altered or modified depending upon the boundary conditions and the governing equations that completely define the system.

The idea employed behind the collocation approach is to satisfy the governing PDEs at each of the nodes in the domain. If a Dirichlet or a Neumann boundary condition is imposed on a node that is on the boundary, then an equation that satisfies the boundary condition is developed for the boundary node instead of satisfying the governing partial

differential equation. The Kansa's collocation method approximates the solution of $\boldsymbol{u}$ (the unknown) by a radial basis function (multiquadric) expansion.

$$\boldsymbol{u}(x) = \sum_{j=1}^{N} c_j \Psi(\|x - \xi_j\|_2), \qquad \mathbf{x} \in \Re^n \tag{15}$$

The set of centers is represented as Ξ ; $\Xi = \{\xi_1, \xi_2 \ldots\ldots \xi_N\}$, the collocation points and the centers often coincide. Equation (15) can easily be interpreted from the discussions presented for equations (3), (5) and (14). Assuming $\mathbf{N_d}$ to be the number of nodes carrying a Dirichlet boundary condition, $\mathbf{N_n}$ to be the number of nodes carrying a Neumann boundary condition and $\mathbf{N_i}$ be the remaining nodes or the interior ones. The total number of nodes ($\mathbf{N_t}$) covering the domain is given by $\mathbf{N_t} = \mathbf{N_d} + \mathbf{N_n} + \mathbf{N_i}$. Any PDE problem can then be described in the following manner.

$$\mathcal{L}\boldsymbol{u} = \boldsymbol{f} \qquad \text{(for all } \mathbf{N_i}) \tag{16}$$

$$\boldsymbol{u} = d \quad \text{on} \quad \Gamma_d \qquad \text{(Dirichlet's conditions, for all } \mathbf{N_d}) \tag{17}$$

$$\partial\boldsymbol{u}/\partial n = n \quad \text{on} \quad \Gamma_n \qquad \text{(Neumann's conditions, for all } \mathbf{N_n}) \tag{18}$$

$\mathcal{L}$ signifies the differential; Γ_d and Γ_n are the Dirichlet and Neumann boundaries respectively, d and n signify the values present on such boundaries, whereas $\boldsymbol{f}$ signifies the right hand condition that exists in the governing PDE it can be 0 or some constant

value or function of variables depending upon the nature of the equation. On applying the set of conditions to the respective nodes, it will lead us to a matrix problem of the following form

$$\begin{pmatrix} \boldsymbol{u}_{\mathcal{L}} \\ \boldsymbol{u} \end{pmatrix} \begin{pmatrix} C \end{pmatrix} = \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} \tag{19}$$

where the vector **a** collects values for $\mathbf{N_i}$ nodes and the vector **b** collects values for $\mathbf{N_d} + \mathbf{N_n}$ nodes. Where $\boldsymbol{u}_{\mathcal{L}} = \mathcal{L}\Psi\,(\|x - \xi_j\|_2)\,|_{x = xi};\ _{xi} \in \mathbf{N_i}$; $\xi_j \in \Xi$ and $\boldsymbol{u} = \Psi\,(\|x_i - \xi_j\|)$; $_{xi} \in \mathbf{N_d} + \mathbf{N_n}$; $\xi_j \in \Xi$.

The matrix $\begin{pmatrix} \boldsymbol{u}_{\mathcal{L}} \\ \boldsymbol{u} \end{pmatrix} \in \Re^{NT \times NT}$; $C \in \Re^{NT \times 1}$ and the RHS vector $\in \Re^{NT \times 1}$.

The above discussion provides the mathematical aspects of the Kansa approach also known as the non-symmetrical collocation method due to the nature of the collocation matrix (refer to appendix C) in the simpler form that has been used in the present work.

2.2.5 Summary

This concludes the present chapter which has provided an introduction along with an insight into the theories involved in meshless methods particularly the collocation approach and the RBFs that are used for interpolation. The next chapter presents an overview of the ideas involved in Artificial Neural Networks (ANNs). Section 3.4 in the next chapter is entirely devoted in highlighting the facts involved in the Kohonen Networks since it has been applied in the present work. As mentioned earlier, the purpose

of using self-organizing feature map is to move the nodes to positions that will provide the most accurate solution with the given number of degrees of freedom.

CHAPTER 3

3 Neural Networks

3.1 Introduction

The discipline of neural networks originated from a understanding of the human brain. It had its origins when McCulloch and Pitts in the 1940s found that neurons could be modeled as simple threshold devices which perform logic functions. Later Hebb, in the late 1940's, proposed the Hebbian rule to describe how learning affects the synaptics between two neurons [18], [19]. These fundamental discoveries laid the foundations for future research in this area and over the years many models were put forward; these include the perceptron model by Rosenblatt proposed in 1957 and in 1960 Widrow and Hoff proposed the adaline (adaptive linear element) model [19], [20]. Some pioneering works were conducted on competitive learning and self-organization by von der Malsburg and Stephen Grossberg and in 1973 Stephen Grossberg, based on the connection patterns found in the visual cortex, proposed the concept of self-organization maps (SOMs) and one of the well-known network models based upon such maps is the Kohonen network [21], [22]. The modern era of neural-network research is commonly deemed to have started with the publication of works related to the Hopfield network in 1982. This was followed by the advent of some additional models such as the Hamming network proposed by Lippman in the mid-1980s, the multilayer perceptron (MLP) model trained with the back propagation (BP) published in 1986 by Rumelhart

and many more. Biological neurons are the basic cells that make up the nervous system and the brain; small electrical gap junctions called synapses interconnect them. The electrical signals of the brain are processed in the neuron's cell body and sent to the synapses through the axons and dendrites (neuron endings) and the synapses distribute this signal to the neighboring neurons. It is estimated that an average human brain consists of 100 billion neurons of various types, with each neuron connected to up to 104 synapses. This massive, as well as complex, network of neurons processes signals separately and simultaneously thereby demonstrating a parallel distributed processing system. This parallel aspect is one of the main features that various artificial neural networks (ANNs) try to emulate. Like its counterpart in nature, the basic functional unit of an ANN is also known as a neuron. Such a node (neuron) processes all fan-in from other nodes and generates an output according to a transfer function called the "activation function", which represents a linear or non-linear mapping from the input to the output and is denoted by ϕ **(.)** (Fig 5)**.**

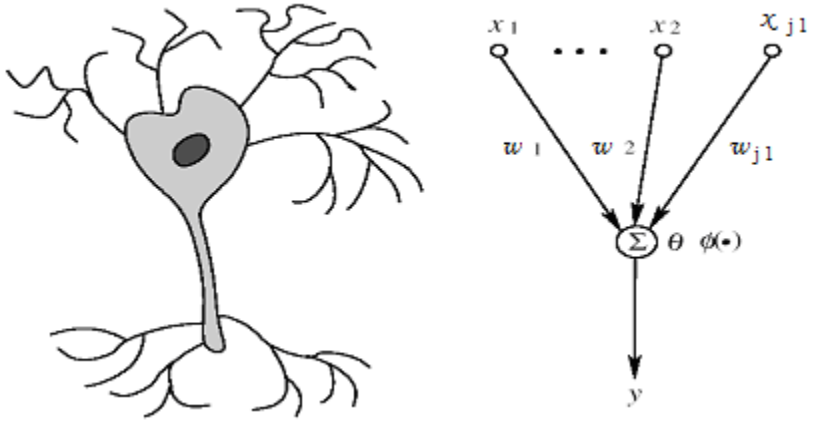


Fig 5

On the left a biological neuron, on the right a mathematical model representation found in ANNs – a McCulloch-Pitts neural model [19].

$$\text{net} = \sum_{i=1}^{j_1} w_i x_i - \theta = w^T X - \theta \qquad (20)$$

$$y = \phi(\text{net}) \qquad (21)$$

where x_i is the **i**th input, w_i is the link weight from the ith input, the vector $\mathbf{w} = (w_i, \ldots\ldots, w_j)^T$, $X = (x_i, \ldots\ldots, x_j)^T$, θ is a threshold or bias and j_1 is the number of inputs. The activation function ϕ, equation 21), is usually some continuous or discontinuous function mapping the real numbers into the interval (-1, 1) or (0, 1). The following are examples of some of the activation functions used.

Hard Limiter threshold (a step function)

$$\phi(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases} \qquad (22)$$

Logistic function (sigmoid function)

$$\phi(x) = 1/(1 + e^{-\beta x}) \qquad (23)$$

Semi linear function

$$\phi(x) = \begin{cases} 1 & x > a \\ x/2a + 1/a & -a \leq x \leq a \\ 0 & x < -a \end{cases} \qquad (24)$$

In practical implementations, all the neurons are typically assumed to have the same activation functions. The McCulloch–Pitts neuron model employs the sigmoidal

activation function and has been used in most neural-network models, including multilayer perceptrons (MLPs) and Hopfield networks. Many other neural networks are also based on the McCulloch–Pitts neuron model, but use other activation functions e.g., in the adaline and Kohonen's self-organizing maps, (SOMs), linear activation functions are used. The sigmoidal activation function is suitable for a training algorithm due to its monotonic character and derivability but from an approximation viewpoint, the function is not always an optimal choice and, as a result, we find the use of other activation functions in ANNs.

Neural network models are specified by the net topology, node characteristics, and training rules. The training rules specify an initial set of weights and indicate how weights should be adapted during use to improve performance. A detailed discussion on these aspects of ANNs has been provided in the following sections.

The ANNs have the ability to detect trends or patterns from complicated sets of data. The trends in such cases are difficult for humans to identify and it is in such scenarios that ANNs play a significant role. After being fully trained, the ANNs gain the ability to provide projections when new situations are presented to them. The neural network models are used to address certain types of problems such as a) optimization, b) classification, and c) prediction. In the electromagnetics community they have been put to use in number of ways e.g., optimization of magnetic devices [23], in FEM solvers for approximating mesh density [24], magnetic performance prediction [25] and many more [26], [27]. This work uses an ANN to optimize a meshless model applied in the field of electromagnetics; as mentioned earlier, in this scenario we will look for an optimal nodal

arrangement in a given problem domain that will provide us with the accurate field solution.

3.2 Neural Network Architecture

The weight matrix $\mathbf{w} = [w_{ij}]$, where w_{ij} denotes the connection weight from node i to node j, is used to describe the network architecture. When $w_{ij} = 0$ there is no connection from node i to node j. By setting the connection weights between nodes, one can realize different network topologies. The basic structure of a neural network consists of one or more layers of neurons. The input layer distributes the network inputs to the subsequent layers. One or more hidden layers follow it. The functioning of the hidden layer is invisible to the user and hence the name. The hidden layers usually do the computation and the result is passed on to the output layer. In terms of architecture, neural networks can broadly be classified into a) feed forward neural networks (FNNs) b) recurrent neural networks (RNNs) and the combination of two. Some popular network topologies include fully connected layered FNNs, RNNs, lattice networks and layered FNNs with lateral connections (Fig 6).

FNNs are the most common type of neural networks. As implied by the name, the signal propagation is only in the forward direction, and there is no physical feedback. The network is usually arranged in the form of layers. FNNs exhibit no dynamic properties - the networks are simply a nonlinear mapping and are often referred to as universal approximators [19], [28]. It has been proved that with an appropriate number of hidden layers, feed forward networks can be used to approximate any function. Examples include MLPs and RBFNs, which are fully connected, and layered FNNs. In RNNs,

there is at least one feedback connection that corresponds to an integration operation or unit delay. A RNN actually represents a nonlinear dynamic system. The Hopfield models and the Boltzmann machines are examples of RNNs. Lattice networks are comprised of one-, two- or higher-dimensional arrays of neurons (Fig 6 c). Each array has a corresponding set of input nodes. The Kohonen network uses a one- or two-dimensional lattice architecture [22] [29], [30].

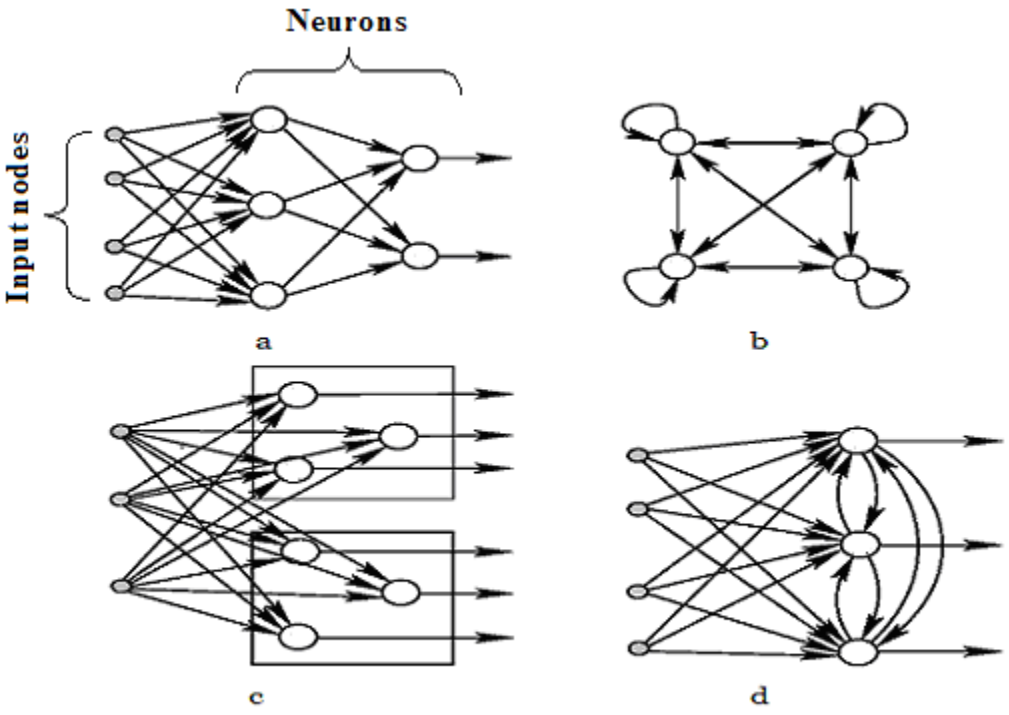


Fig 6

(a) Fully connected layered FNN. (b) RNN. (c) Two-dimensional lattice network. (d) Layered FNN with lateral connections. The large circles denote neurons, and the small circles denote input nodes.

3.3 Learning in Neural Networks

Learning can be viewed as the task of searching through a large space of hypotheses; the goal of this process is to find a result or representation that best fits the training scenarios. A learning algorithm can adapt the nonzero elements of **w** or weights. Learning in the context of neural networks can be defined as a process by which the free parameters of a neural network adapt through a process of stimulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the parameter changes take place and can be viewed as a nonlinear optimization problem in which the goal is to find a set of network parameters minimizing the cost function for a given set of examples. This kind of parameter estimation is also called a *training algorithm*. They are usually trained by epoch, which is a complete run when all the training examples are presented to the network and are processed using the learning algorithm only once. After learning, a neural network represents a complex relationship, and possesses the ability for generalization. When a new input is presented to the trained neural network, a reasonable output is produced, provided it was trained with correct input data. The following are a few of more common learning algorithms (models) that are used in the various network models [18], [19]. The purpose of this discussion is to highlight the manner in which the learning models update weights and this is how they are different from one another, most of the network architectures use these update rules through an iterative process.

Error Correction learning: Consider a network with an input vector x(n) , the output being $y_k(n)$, the desired response being denoted as $d_k(n)$ and the error signal as $e_k(n)$. k

signifies the neurons in the output layer and n is the current time step. Based upon the error signal $e_k(n)$ a sequence of corrective adjustments to the weights of neuron k is applied. The purpose is to make the output signal $y_k(n)$ move closer to the desired response $d_k(n)$ in a step by step manner. This objective is achieved by minimizing a cost function, $\xi(n)$, defined in terms of the error signal $e_k(n)$ as

$$\xi(n) = \tfrac{1}{2} (e_k(n))^2 \tag{25}$$

The minimization of the cost function $\xi(n)$ defines a learning rule commonly referred to as the "delta rule" or the "Widrow – Hoff rule" an approach applied in single and multi layer feed forward networks. According to this $w_{kj}(n)$ the synaptic weight for neuron k, is excited by element $x_j(n)$ of signal vector x(n) and is defined by

$$\nabla w_{kj}(n) = \eta e_k(n)\, x_j(n) \tag{26}$$

where η is a positive constant that determines the rate of learning as we proceed from one step in the learning process to another. Having computed the weight adjustment $\nabla w_{kj}(n)$ the updated value of synaptic weight w_{kj} is determined by

$$w_{kj}(n + 1) = w_{kj}(n) + \nabla w_{kj}(n) \tag{27}$$

Memory-Based (Instance based) Learning: Here the past experiences are explicitly stored in a large memory of correctly classified input – output examples: (χ_i, d_i) where i ranges from 1 to N, χ_i is the input vector, d_i denotes the desired response and N is the number of data points. In classifying a test vector χ_{test} this algorithm responds by retrieving and analyzing the training data in a local neighborhood of χ_{test}. One of the important aspects involved here is the manner in which the local neighborhood of the test vector χ_{test} is defined; i.e. by defining a function of the form.

$$\min d(x_i, \chi_{test}) = d(x'_N, \chi_{test}) \tag{28}$$

where $d(x'_N, \chi_{test})$ is the Euclidean distance between the vectors, x_i and χ_{test}. The class associated with the minimum distance, which is the vector x'_N, is reported as the classification of χ_{test}. Apart from this, there are three other properties that are defined in this technique – a) the number of neighbors; b) a weighing function such that the nearby points are weighted strongly compared to far points (but this is optional in some cases); and c) fitting the local points .

Hebbian Learning: This is the oldest among the learning rules and states that if two neurons on either side of a connection are activated synchronously then the strength (weight) of that connection is selectively increased or if on either side they are activated asynchronously, then the connection is selectively weakened or eliminated. This approach is known to have linear dependency on the size of the network, where saturation (a state when the output of a node is near its extreme value) leads to

"catastrophic forgetting". The outputs from adjacent nodes are locally available and are used to produce a local modification that is input specific, implying that it doesn't take into account the overall system input-output characteristic. The manner in which the weights are adjusted can be represented in the following manner

$$w_{kj}(n + 1) = w_{kj}(n) + \eta x_k(n) x_j(n) \qquad (29)$$

where η is a positive constant that determines the rate of learning; the x's are the outputs of kth and jth elements. We find such rules in recurrent networks, such as the Hopfield neural network.

Boltzmann Learning: This is a stochastic learning algorithm derived from the ideas in statistical mechanics and is based upon the Boltzmann machine in which the neurons constitute a recurrent structure and operate in a binary manner. For example, they are either in an "on" state denoted by +1 or in an "off" state denoted by -1. Boltzmann Learning is characterized by states of individual neurons as shown by

$$E = - \tfrac{1}{2} \Sigma \Sigma w_{kj} x_k x_j \qquad j \neq k \qquad (30)$$

where w_{kj} is the weight connecting neuron j to k and x_j is the state of neuron j. It is similar to the error-correction learning rule but instead of a direct difference between the result value and the desired value, the difference between the probability distributions of the system is considered. Hence in effect, the entire system output is taken into account.

The probability distribution P during the flipping state of neuron k from x_k to state - x_k at some state T is given as

$$P(x_k \rightarrow - x_k) = 1/ 1+ \exp (- \Delta E_k / T) \quad (31)$$

Competitive Learning: Unlike the Hebbian based ANN where several outputs may be active simultaneously, there is only one in this case. The neurons compete among themselves to become active (fired). This feature makes it highly suitable for discovering features statistically that can be used to classify a set of input patterns. The Kohonen network uses this learning rule and a detailed discussion on this approach is included in the next section.

So far in this chapter, the rules involving the training of ANNs have been described. There is a certain set of standard methodologies that use these rules to perform the task of training, known as learning methods, and they are classified as supervised, unsupervised, reinforcement, and evolutionary learning. The present discussion will highlight only the first two forms since they are more relevant to the work being discussed here. Supervised learning involves comparison between the actual network output and the desired output. Network parameters are adjusted by a combination of training patterns and the corresponding errors between the desired output and the actual network response. Supervised learning is a closed-loop feedback system, where the error is the feedback signal. Error correction learning and Boltzmann learning rules are implemented using this method. In the case of the unsupervised method, there are no target values and it does not

require any teacher. In this scenario, the training data is unlabeled implying there is no "a priori" knowledge of a distribution class that may exists in the data source. In terms of a statistical approach, the models using an unsupervised method often use clustering algorithms or probability density approximations. A criterion is needed to terminate the learning process or else the learning process continues even when a pattern, which does not belong to the training patterns set is presented to the network. The Hebbian learning and the competitive learning schemes are the ones that are involved with such an approach.

3.4 Kohonen Networks

Stephen Grossberg, Von der Malsburg and Fukushima conducted pioneering work on competitive learning and self-organization maps, based on the connection patterns found in the visual cortex. Upon those concepts professor Teuvo Kohonen proposed his self-organization map (SOM) which is widely known as the Kohonen Network [22].

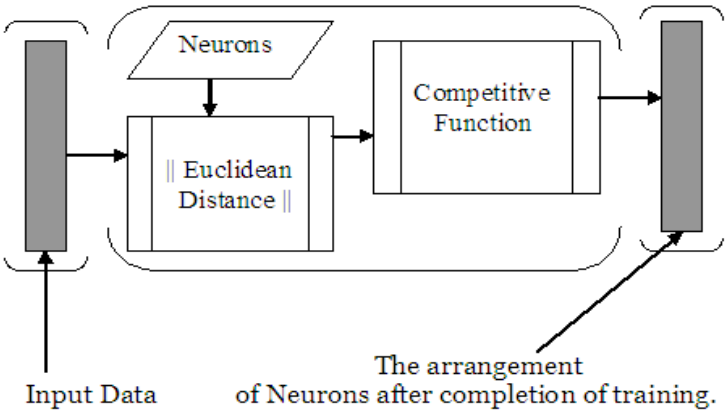


Fig 7

The Kohonen Model

The Kohonen network (Fig 7) is a feed forward structure with fully interconnected processing units that compete among themselves (competitive learning) to be activated or fired. The output layer is called the Kohonen layer. The input nodes are fully connected to the output neurons with their associated weights. Lateral connections between neurons are used as a form of feedback whose magnitude is dependent on the lateral distance from a specific neuron, which is characterized by a neighborhood parameter (Fig 8).

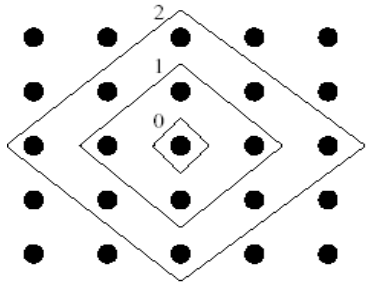


Fig 8

Neighborhoods (0, 1 and 2) of the centermost unit with a rectangular lattice The innermost polygon corresponds to 0-, next to the 1- and the outmost to the 2-neighborhood.

A SOM can be thought of as a net which is spread over the data cloud (the vector carrying tagged data in the input space). In the present work, this data cloud will be an error grid (in vector form) spread over the input space; the error being the measure of the accuracy of the approximate solution obtained using the Kohonen neurons (nodes) in conjunction with the meshless model. The SOM training algorithm moves the weight vectors towards the element of the data cloud based on certain criteria (here it is the highest error criteria), so that all the neurons span the data cloud and the network of neurons is organized; this is done for all the neurons and while one neuron is being moved the neighboring neurons also get adjusted accordingly. The highest error criteria

involves a process in which a neuron is moved towards a location where the error is the highest from a set of error points that are within its domain of influence. The SOM is trained iteratively, in each training step one vector **x** from the input data set is chosen randomly and the distances between it and all the weight vectors of the SOM (Kohonen nodes) are calculated using some distance measure. Here we are using the Euclidean distance. The neuron whose weight vector is the closest to the input vector **x** is called the Best Matching Unit (BMU) denoted here by **b**

$$\| x - m_b \| = \min_{i \in N} \{\| x - m_i \|\} \tag{32}$$

N represents the set of Kohonen nodes and ||.|| the Euclidean distance. Each variable has an associated weighting factor; the greater this value, the bigger the component's (neuron's) effect on map organization will be.

After finding the BMU, the weight vectors of SOM are updated so that the BMU is moved closer to the input vector in the input space, the topological neighbors of the BMU are treated in a similar manner (Fig 9).

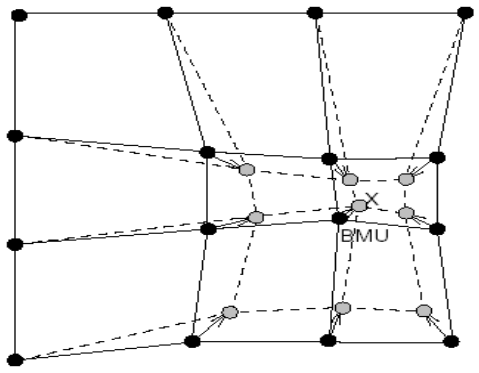


Fig 9

The solid and the dashed lines correspond to situation before and after updating respectively.

The Kohonen update rule can be described in the following manner:

$$m_i(s+1) = m_i(s) + \alpha(s)h_{bi}(s)[\mathbf{x}(s) - m_i(s)] \quad (33)$$

where s denotes time or iteration steps, x is an input vector drawn from the input data set during s, $h_{bi}(s)$ is the excitation response also known as the neighborhood function or the kernel function around the winning neuron b and $\alpha(s)$ is the learning rate during s. The neighborhood function is a non-increasing function (decay) of iteration steps and the distance of neuron, **i** from the winning neuron **b:** (h_{bi}) is typically selected as

$$h_{bi}(s) = h_0 e^{(-\|m_b - m_i\|^2/\sigma^2(s))} \quad (34)$$

h_0 here is a positive constant. The topological neighborhood also known as the neighborhood size, is a decreasing function of *s* and one popular choice for the function decay is to use an exponential decay as shown above. The training can be stopped when the map achieves equilibrium with a given accuracy or when a specified number of iterations is reached.

The role that will be played by the Kohonen network in the present work is to identify or to locate the regions in a given problem space, where a given number of nodes can be re-arranged to obtain the best possible solution using the meshless model.

3.5 Summary

This concludes our discussion pertaining to ANNs. It provided in depth details concerning certain important aspects of ANNs such as learning and the manner in which learning is handled in various models of ANNs; emphasis was placed on presenting the Kohonen model as it has been implemented in the present work. The following chapter will present the ideas behind designing the meshless and the Kohonen models and various test scenarios that were used during the design phase.

CHAPTER 4

4 Model Design and Simulation

This section will discuss in detail the approach involved in designing the meshless and the Kohonen models and then present the manner in which the two have been integrated. It will also present some mathematical aspects involved in them during the course of the discussion and the simulation results.

4.1 Meshless Model

We will present number of scenarios for solving elliptic partial differential equations using RBF collocation. The approach will be similar to that used by Kansa [34] but instead of multiquadrics, the present work uses Gaussian RBFs in most of the scenarios. Before we proceed further, let us familiarize ourselves with certain aspects of field problems with boundary conditions, which will help us to appreciate the manner in which the technique is applied. Generally, a field problem is represented by a differential equation, which is similar to those used in equations (19), (20) and (21) to describe the attributes of approximation, and is of the form.

$$\mathcal{L}\,\boldsymbol{u} = \boldsymbol{f} \tag{35}$$

$\mathcal{L}$ is the differential operator, $\boldsymbol{u}$ is the unknown function to be determined, $\boldsymbol{f}$ is the forcing function, and together with boundary conditions, they describe the field problem for the domain under analysis. In electromagnetic problems, equation (38) is given by the Poisson, Laplace or Helmholtz equation, in which $\boldsymbol{u}$ is a scalar or a vector field. For instance, in an example of an electric field problem to be discussed later, $\boldsymbol{u}$, the unknown, represents the electric potential V and with a uniform charge density distributed over the space in a homogenous medium the differential operator acting on the potential in space is described by the Laplace equation and is of the following form

$$\nabla^2 V = 0 \tag{36}$$

The conditions that represent the behavior of $\boldsymbol{u}$ on the edges Γof the domain are called the "boundary conditions" and they constraint the fields along Γ of the domain under analysis. Among these conditions, one can assign a Dirichlet's condition, which occurs when a given value of $\boldsymbol{u}$ is assigned on the boundary Γ, or a Neumann's condition, which is assigned when the derivative of $\boldsymbol{u}$ is normal to Γ. In a homogenous environment, the Dirichlet's and Neumann's conditions are represented as $\boldsymbol{u} = 0$ and $\partial \boldsymbol{u}/\partial n = 0$ respectively [31].

We begin our discussion with a simple parallel plate capacitor arrangement where there are two plates held at different potentials with an air gap in between them (Fig 10 a). This example will be used extensively in order to demonstrate the fundamentals that are involved in this method.

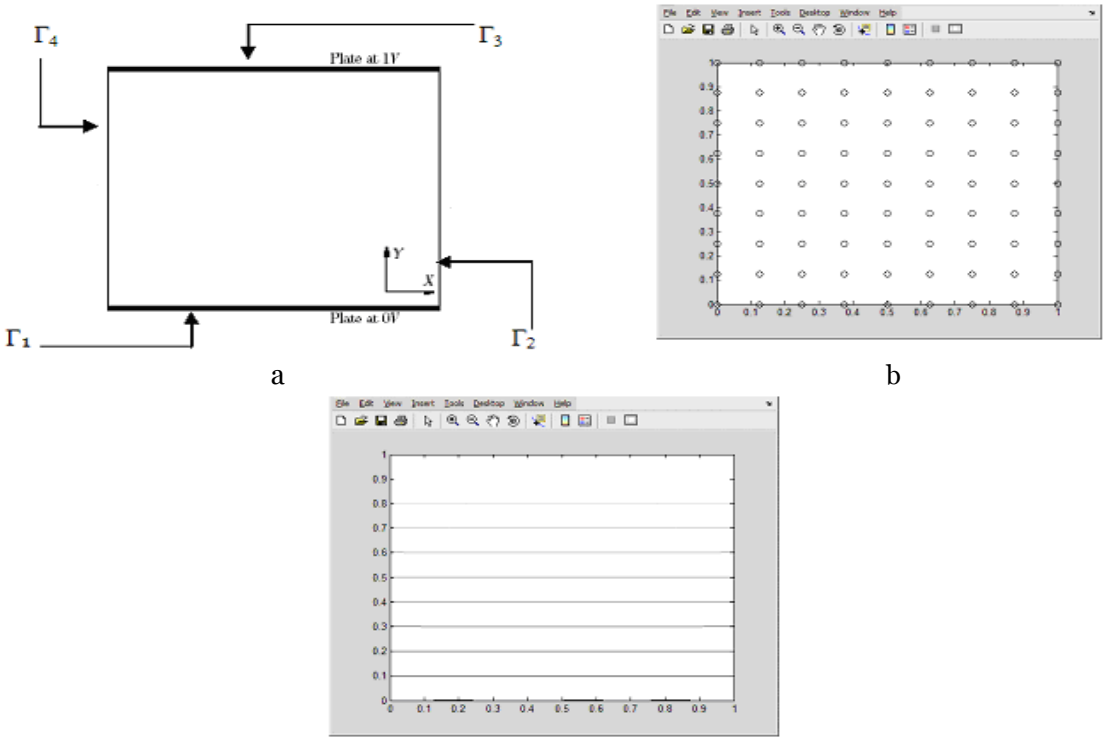


Fig 10

a) Set up for the Laplace equation $\nabla^2 V = 0$ along with boundary conditions $\partial V/\partial n = 0$ for $y = 0$, $\partial V/\partial n = 0$ for $y = 1$, $V = 1$ for $x = 0$ and $V = 0$ for $x = 0$ b) 81 nodes uniformly distributed within a region of $[0, 1]^2$ c) The potential distribution.

In the example above, the bottom and the top boundaries were set to potential values of 0 and 1 respectively whereas the left and right boundaries were set to a homogenous Neumann's condition of $\partial V/\partial n = 0$ (V denotes the unknown field). 81 nodes were uniformly distributed over the problem domain of which 32 nodes were present on the boundaries and the remaining 49 were inside. We carried out most of our simulations with node numbers ranging from 16 to 450 nodes, the count was kept small in order to save computation time. The shape parameter, α, in all the test scenarios presented in the following sections unless specified, was set within a range of 3 to 5. The value of α was

decided by carrying out a basic experimentation involving stationary approximation (refer to section 2.2.4) where we began with a large value of α and gradually decreasing the value until there was a warning from Matlab of the matrix being ill conditioned, a value of α prior to reaching such a state was considered good enough as it was also noticed that the accuracy increased with the decreasing value of α (the trade off mentioned in section 2.1.1). An extreme test case was also performed where we uniformly distributed 2000 nodes in a similar [0, 1] X [0, 0.5] domain, here α was altered to a value of 19 to obtain good results (R.M.S error being of the order 1.10E-03). In the present problem the first assumption that was made, as was mentioned earlier, is that the centers of the basis functions associated with the interior points are considered to be the same as the collocation points. We then had to fit the collocation points and the centers with the boundary conditions. The manner in which the collocation matrix is derived has been discussed in section 2.2.4, where the governing equation is applied to the interior nodes whereas the Dirichlet's conditions and Neumann's conditions, the latter involve handling of derivatives, are applied to the boundary nodes (for further details, refer to appendix C). This approach involving RBFs resulted in the largest errors near the boundaries. To avoid this, the collocation points on the boundaries were made to satisfy the PDE as suggested by Fedoseyev [32].

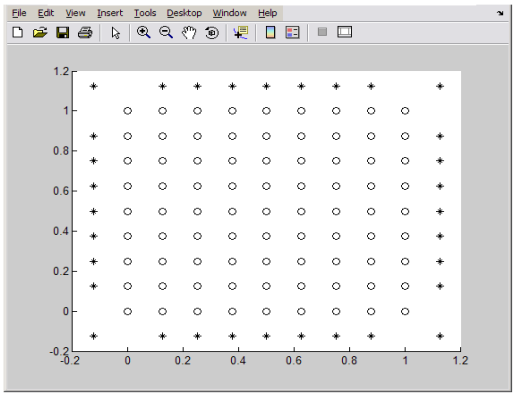
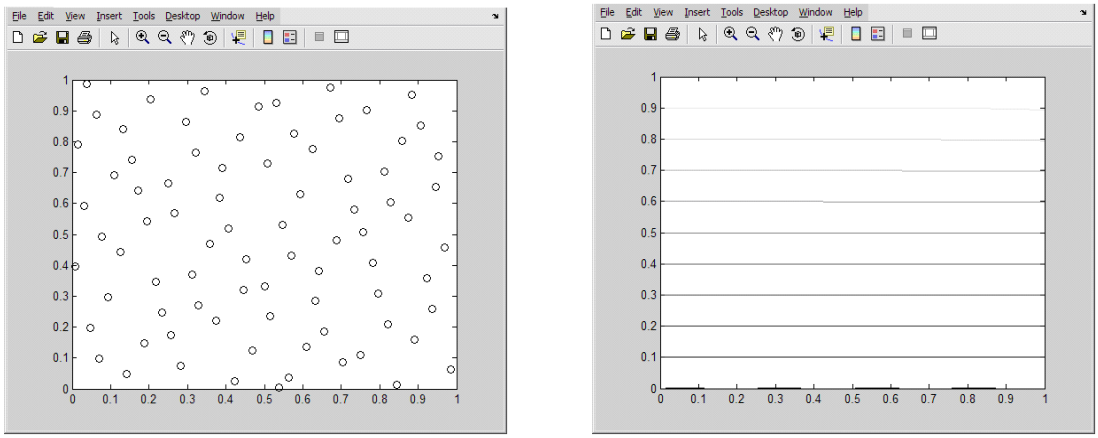


Fig 11

The additional collocation points represented as '*' that were used to obtain accurate results on the boundaries

To meet the guideline as stated earlier, additional collocation points were created outside the boundaries (Fig 11, denoted with asterisk) which were equally spaced. The next task involved a test scenario as performed in the previous setup but with a random distribution of nodes (Fig 12). We retained the same boundary conditions as earlier and the random distribution of 81 nodes was achieved using the Halton sequence (refer to appendix B).



a b

Fig 12

a) The random distribution of 81 nodes within a region of $[0, 1]^2$. b) The potential distribution.

The R.M.S errors were 1.22e-003 and 1.13e-003 for the uniform and random distributions respectively. With a similar and simple geometry and with the same number of nodes in place, there were no substantial differences in the results.

Our next endeavor was to verify whether such an approach (collocation RBFs) has the ability to handle the interface conditions. To study this we considered a problem where we had two domains Ω_1 [0 1; 0.5 1] and Ω_2 [0 1; 0 0.5] with permeabilities μ_1 and μ_2 respectively (Fig 13) the condition being $\mu_1 = 10 * \mu_2$ (i.e., the upper half had 10 times the permeability of the lower half).

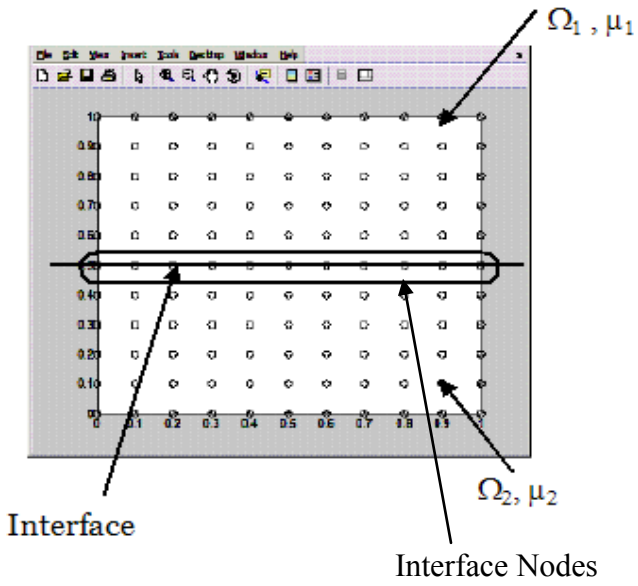


Fig 13

The two domains with different permeabilities each having 66 uniformly distributed nodes.

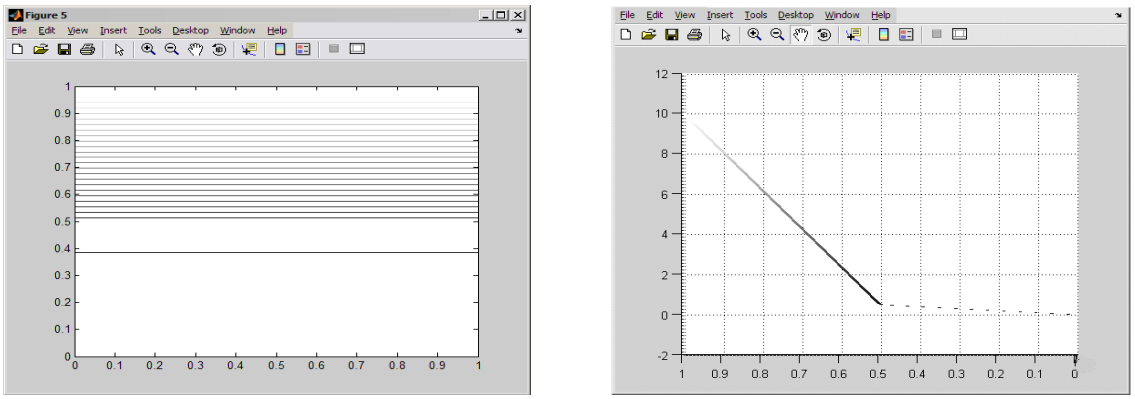
A homogenous Neumann boundary condition was applied to - the left and right hand edges for both domains i.e., $\partial V/\partial n = 0$. In the case of Ω_2 the essential boundary condition at the bottom edge was set to $V_2 = 0$, whereas for Ω_1 the top edge was set to $V_1 = 1$. 66

nodes were uniformly distributed in each domain. As can be seen from the figure above, we had 11 nodes at the interface; these were the set of nodes that required a slightly different treatment for solving the problem. A particular RBF solution was defined for each of the two regions. The set of nodes $\mathcal{N}$ (here it was 11) distributed at the region of interface Γ_v simultaneously belonged to both the regions and we used the following two sets of equations for coupling the solutions at the interface.

$$V_1(r_v) = V_2(r_v) \tag{37}$$

$$\mu_1.\nabla V_1(r_v) = \mu_2.\nabla V_2(r_v) \tag{38}$$

where r_v denotes the point at the interface. The equation (37) forces the continuity of the solution at the interface and the equation (38) is established by the provided interface conditions [33],[34],[35],[36]. Using this approach, we obtained the results exhibited shown in Fig 14.



a b

Fig 14

The solution to the interface problem a) the contour plot displaying the contour lines getting accumulated at the upper half b) the side view of the surf plot fort the same solution showing the change in the potential over the entire domain.

Until now, the discussions have laid emphasis on applying the theoretical aspects to some simple problems. Our next attempt was to extend the ideas developed so far and apply them to geometries that are more complex and more relevant from an application point of view.

The first scenario involved a semicircular hole present on the upper edge of a square domain as represented in Fig 15 a. The left and the right hand edges were set to a homogenous Neumann boundary condition of $\partial V/\partial n = 0$, whereas the bottom edge and the top edge had the potentials of V = 0 and V = 1 respectively.

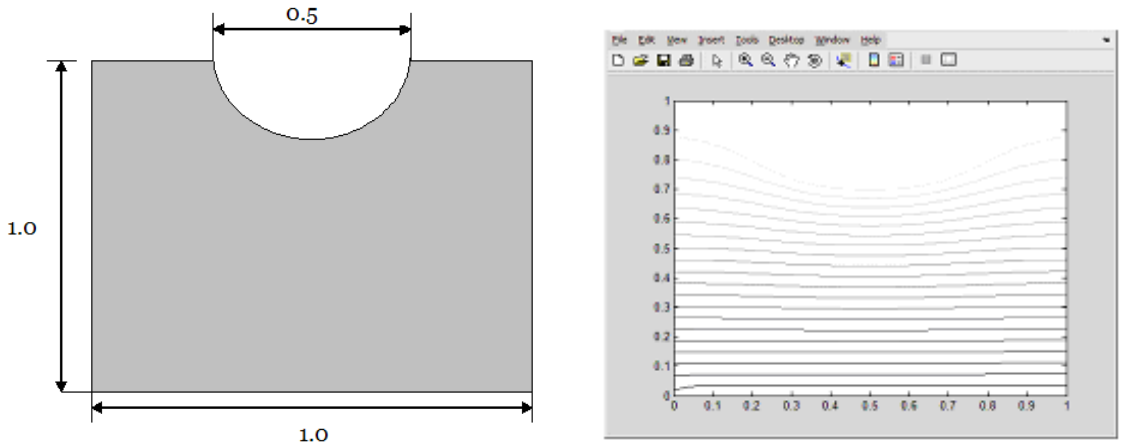
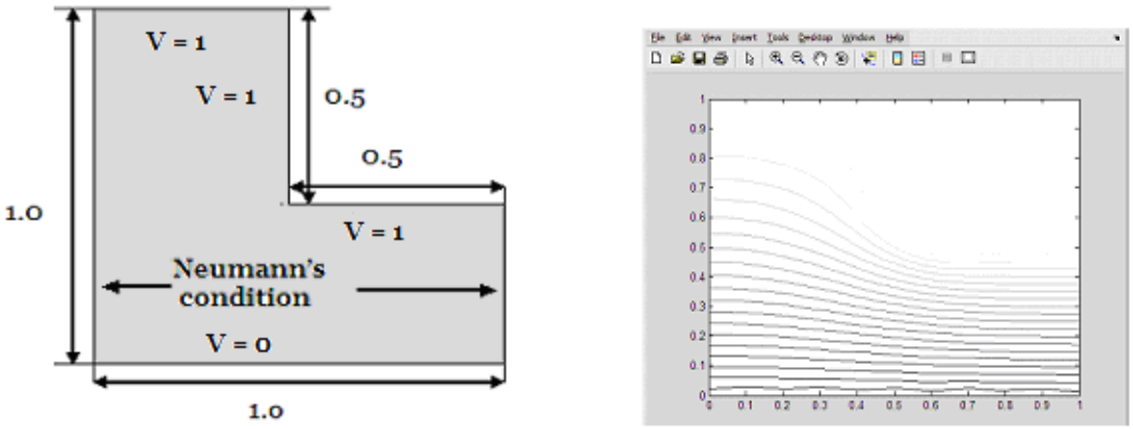


Fig 15

a) Problem domain with a semicircular hole on top. b) The contour plot displaying the potential distribution inside the problem domain.

The governing equation considered was the Laplace equation, $\nabla^2 V = 0$ equation (36), 410 nodes distributed uniformly were used to solve the problem and the result obtained is shown in Fig 15 b.

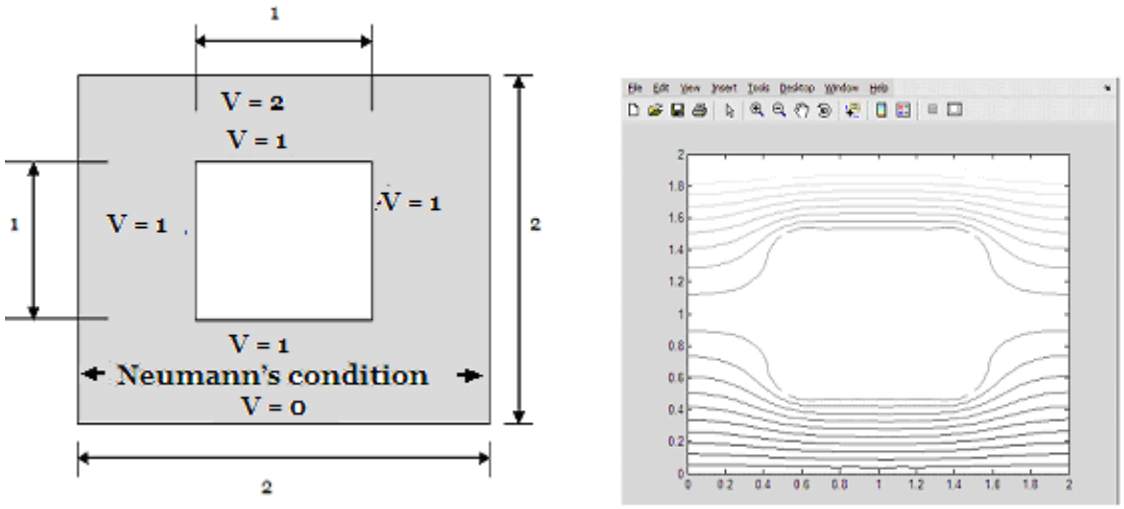


a b

Fig 16

a) The "L" problem domain. b) The contour plot displaying the potential distribution inside the problem domain.

We further tested our approach for solving the Laplace equation by investigating the "L" domain problem Fig 16 a, and a problem domain that had a square hole inside, Fig 17 a. In both cases we had the homogenous Neumann condition.



a b

Fig 17

a) The problem domain with a square cavity. b) The contour plot displaying the potential distribution inside the problem domain.

The results from both the instances were according to what was anticipated as shown (Fig 16 b and Fig 17 b).

At this point, we have demonstrated the use of MMs in function approximation for simple static field scenarios. The examples so far presented establish that such methods are capable of solving cases where we have defined boundary conditions and problems that involve the Laplace equation. The ideas developed will be carried forward while dealing with more realistic problems in electromagnetics that are presented below but before we proceed further to study the use of MMs in electromagnetics [37], [38], [39] we deviate slightly to review some vector algebra and properties of electromagnetics that will highlight the manner in which problems concerning them could be handled.

The static magnetic field is completely defined by the curl and the divergence in following equations (39) (40) as per the Helmholtz theorem.

$$\nabla \text{ X } H = J \tag{39}$$

$$\nabla.B = 0 \tag{40}$$

Where B is the magnetic flux density, H is the magnetic field intensity and J is the current density. It is possible to define a given problem in a manner, which will minimize the number of unknowns, and while dealing with approximation problems such as the ones

demonstrated earlier, one would look for functions (which can be scalars or vectors) that can be used in conjunction with magnetic fields to simplify the solution process [33],[40].

Although a scalar function may be used under certain conditions, the vector function is generally used to describe the magnetic field. The scalar function will be discussed later in this section but for the vector function, we begin the discussion with one of the vector identities. We know that the vector identity $\nabla . (\nabla \times A) \equiv 0$; i.e., the divergence of the curl of any vector is identically 0. Now, if the divergence of a vector field is 0 then this vector can be defined as the curl of another vector. In order to relate this aspect with magnetic flux density (B), the vector B can always be written as the curl of another vector A, as shown in equation (41) since the divergence of the magnetic flux density is always 0.

$$B = \nabla \times A \tag{41}$$

If we substitute equation (41) in the vector identity mentioned earlier it would satisfy the identity's stated condition. The vector A is known as the magnetic vector potential. After A is calculated, it can be used to directly evaluate other quantities or to calculate B. The choice should be guided by the ease of use and the intention of highlighting the facts above was to emphasize that the vector potential could work as a substitute for B and this approach has been applied in the examples.

We now define Ampere's law and magnetic flux in terms of the magnetic vector potential (A). In free space where the magnetic permeability μ_0 is constant, we will have

$$\nabla \text{ X } H = J \quad \rightarrow \quad \nabla \text{ X } B = \mu_0 J \tag{42}$$

On substituting the definition of B we get

$$\nabla \text{ X } (\nabla \text{ X } A) = \mu_0 J \rightarrow \nabla (\nabla . A) - \nabla^2 A = \mu_0 J \tag{43}$$

Considering $\nabla . A = 0$, this condition, known as the Coulomb gauge, is the best choice when we are dealing with static fields with no changes in the properties of the magnetic field. This will then lead us to the vector Poisson equation as shown below that will serve our purpose of approximating field values for electromagnetic problems involving current densities as demonstrated in the next set of scenarios.

$$\nabla^2 A = - \mu_0 J \tag{44}$$

In the case of a scalar function, we consider that a field should be conservative in nature, thereby making it curl free. Any vector field F that satisfies the curl free condition $\nabla \text{ X } F = 0$ can be used to describe the gradient of a scalar function φ given by

$$F = -\nabla \varphi \tag{45}$$

on substitution with the curl in place, we will have $\nabla \times F = \nabla \times (-\nabla \varphi) \rightarrow \nabla^2 \varphi = 0$. From an electromagnetics perspective, though the magnetic field intensity H is not, in general, curl free there are important applications where a magnetic field exists but there are no current densities involved (J = 0), such as permanent magnets, and these scenarios are best described using the following equation

$$\nabla \times H = 0 \rightarrow H = -\nabla \varphi \tag{46}$$

Where φ is the scalar function and is known as the magnetic scalar potential. The magnetic scalar potential satisfies the Laplace equation i.e., $\nabla^2 \varphi = 0$ similar to the electric potential [33].

Returning to the discussion on the meshless approach, we now present the scenarios that were used to verify the solutions of Poisson's equation. In the first scenario (Fig 18), we considered a problem $\nabla^2 A = \pm 2.0E+06$ with the boundary condition of A = 0 along all the edges. The problem was solved using 216 randomly distributed nodes and the result obtained had contours, which were elliptical in nature that represented the field inside the problem domain the flux plot for the solution is shown in Fig 18. The next scenarios involved trying the Poisson problem with two different media inside the problem domain, they represented some typical cases for instance a core with coils wound around it (2-D representation Fig 19 a) and a C core sliced in half (2-D representation Fig 19 b). The product of current density (J) and permeability (μ) was 2.0E+06 A m^{-2}.

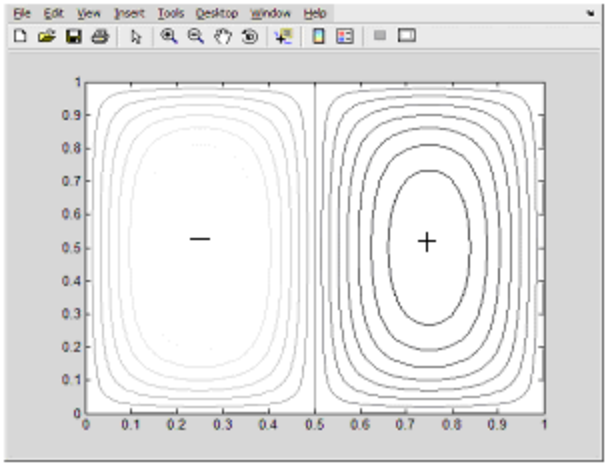


Fig 18

The problem involved two current sources with boundary condition of A = 0 along all the edges.

The top edge in both the cases had the Neumann's condition of $\partial A/\partial n = 0$ whereas the remaining edges had the boundary condition set to A = 0. For the first scenario, 216

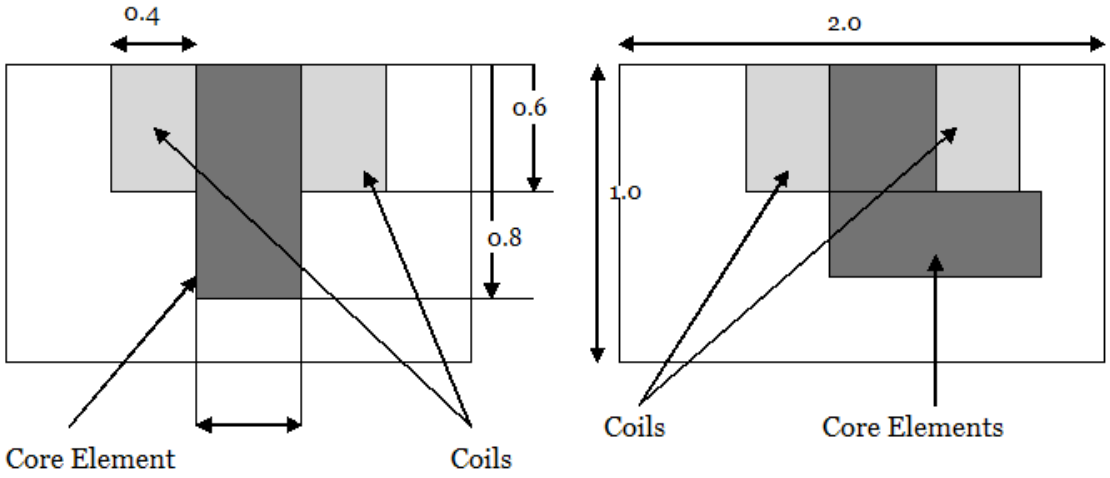
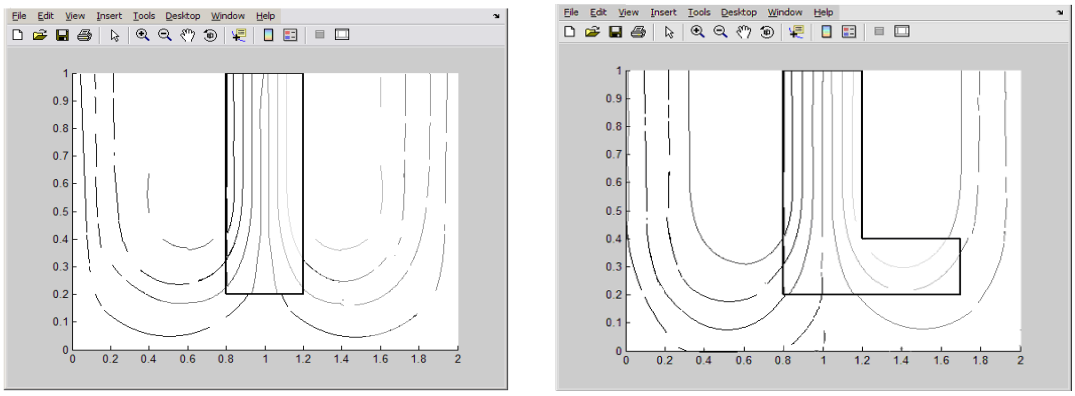


Fig 19

a) The problem involved a rectangular material (limb) with coils adjacent to the limb, the latter having a permeability of 3 times that of the surrounding medium. b) The problem had a slightly different limb arrangement the other properties such as the coil arrangement and the permeability remaining the same as the previous scenario. This problem represented a c core that had been sliced from the middle.

randomly distributed nodes were used whereas for the second case 216 uniformly distributed nodes were used to solve the respective problems. Fig 20 a represents the

result for the scenario 19 a and figure 20 b presents the solution for the scenario 19 b respectively.



a b

Fig 20

Field solutions to the problems presented in figures 19 a and 19 b.

In this section, we presented the ideas that were used in implementing the meshless model and the outcomes when we tested them for solving problems that involved the Laplace and Poisson equations along with geometries having minor complexities. The next section will discuss briefly the manner in which the dense system matrices present in the above technique could be made sparse by using a simple approach involving a check on the radial distance and its effects on the solution.

4.2 Computational Efficiency

In section, 2.2.3 it was pointed out that the Kansa approach has an issue in terms of matrix density as it leads to a fully populated system matrix and with a computational complexity of $O(n^3)$ for the solver, the efficiency is not that considerable. To address this

issue an approach leading to a sparse system matrix with a small bandwidth is needed. To achieve this, we considered a simple problem domain similar to that presented in the first meshless test scenario (Fig 10) , in the present case 16 nodes were uniformly distributed over the problem domain and all the remaining conditions such as the governing equation and the boundary conditions remained the same as for Fig 10. A check was implemented based on radial distance while forming the system matrix, such that if the distance was beyond a certain limit from the center of the node under consideration then the radial value contributing to the formation of the matrix was considered as 0. It was also ensured here that no region of the problem space was left uncovered or in other words, all the regions in the problem domain were covered by at least one RBF. The next example (Fig 21a and 21b) shows the output from one of the experiments that were carried out.

The limiting radial distance that was considered in this scenario was 0.5 implying that any distance value greater than this contributed 0 to the matrix. A stable solution was obtained but we had less accuracy (the R.M.S error being 5.54E-01). As evident, the error was high and it was also noticed that as the limiting condition was reduced further the error value increased. For the same problem domain we will have better accuracy if we use a larger number of nodes as shown in Fig 21 c. The increase in the number of computation points will again deteriorate the computation efficiency, hence to have a sparse matrix, using the above approach wherein we are removing a part of the RBFs contributions to the matrix entries will require substantial human judgment to have an accurate result.

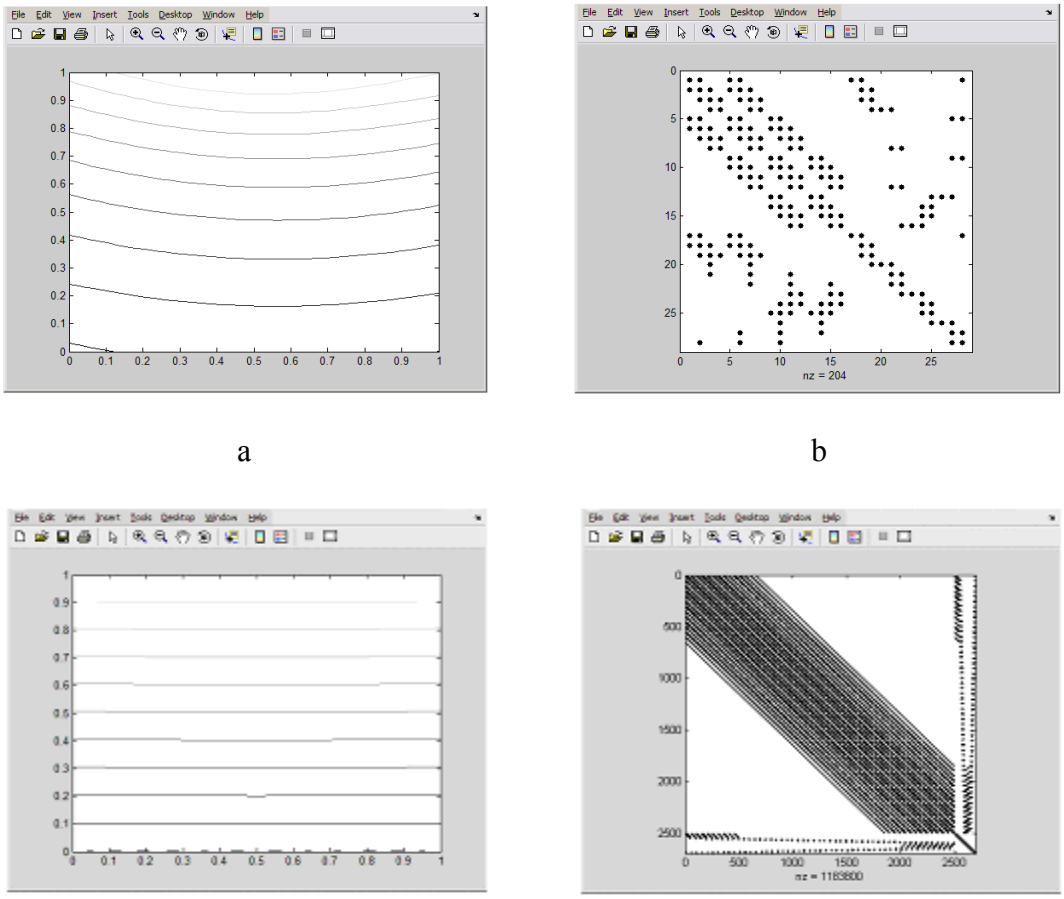


Fig 21

a) The solution with the limiting condition in place. b) The matrix sparsity nz = 204. c) The solution for the same problem using 2500 nodes. d) The matrix sparsity for second scenario nz = 1183800.

As noted earlier, the RBF collocation approach has issues at the boundaries and to address them we use additional collocation points (section 4.1, page 54). If we try to use this approach in such scenarios for sparse matrix generation we will obtain inaccurate results because we lose valuable information specifically at the boundaries that are essential for providing an accurate solution.

This was observed when we carried out experiments on similar lines as discussed above and it also becomes difficult to have a generalized approach in place as it may require the boundary nodes to be treated differently to the interior ones to have good results. Moreover, if we have complex geometries involving curves or abrupt changes in boundary conditions etc., then it will further restrict us in having a complete automated system and will demand human decision making for good solutions. In order to solve problems with a large number of nodes along with a sparse matrix in place as shown in Fig 21 c and Fig 21 d we used another approach involving the scaling of the shape parameter α which involved limiting the span or spread of the RBF function and not removal or "chopping" of the function. In the present scenario it was set to a value of 17 and the R.M.S Error associated with the result was 3.63E-03. The value of α was reached in the manner as discussed in section 4.1.

The above techniques demonstrate an attempt to have some localization in place but there exists a more elegant approach using a different radial basis function known as the compactly supported RBFs (CSRBFs), the most popular being the one introduced by Wendland [10], [41]. It allows multilevel iteration, which involves scaling the size of the basis functions according to nodal distance (the distance that separates one node from the other in the problem space) and progressively interpolate on refined sets of centers (computational grids) based upon residuals [42]. Such a multilevel approach provides a computational complexity of O(n) (n being the interpolation points in the final level), can be put to use in a similar collocation environment and is well conditioned but has a few convergence issues in certain scenarios.

Another alternative approach to achieve computational efficiency is through the use of adaptive iteration such as a Greedy adaptive algorithm [43]. All these ideas have theoretically been proven effective to handle the computational efficiency problem involving RBFs and there are examples of the use of CSRBFs in the field of mechanics. The use of such approaches is still at a very nascent stage in the field of electromagnetics and the present work has not implemented any of these ideas.

The above two sections provided us with an in depth detail concerning the design of a meshless model, along with all the test scenarios. The next section will provide the details concerning the design of a Kohonen network and the technique that was used to integrate the meshless model and the Kohonen model.

4.3 The ANN model and its integration with the Meshless model

We will first provide the details concerning the design of the Kohonen model and later will present the approach of integrating the two models.

In chapter 3, under section 3.4, we discussed the theoretical aspects involved in the Kohonen network. We consider equation (33) i.e., the update rule wherein the weight vectors (neurons) are adjusted in each iteration. We will concentrate on the two parameters present in this rule i.e., the neighborhood function ($h_{bi}(s)$) and the learning rate ($\alpha(s)$) and investigate first as to how each one of them affects the training process. The neighborhood function involves an exponential decay component (refer to equation (34)) that includes two parameters apart from the distance measure namely, h_0 which was

kept at 1 and the neighborhood size. The neighborhood size determines the reach of influence of individual neurons in the network system implying that, if a neuron is moved towards an element of a data cloud, then some of the neighboring neurons should be adjusted (or moved) to implement an effective learning scheme. The neighborhood size is initialized to a large value; in the present case it was set to 10 cm or more. This size is determined by studying the size of the input space and, through trial and error, and is gradually shrunk down to 0 at the end of learning; the gradual shrinkage helps the network in finding a good ordering pattern. To cause this decrease in the influence domain we may use a parameter, as has been done here, known as the "collapse rate", which is slightly less than the value of 1 (here it was set to 0.97), so as the learning iteration proceeds, the neighborhood size gets updated by multiplying the "collapse rate" (a constant) with that of the initial neighborhood size. The idea behind such an approach is to allow every neuron to adapt at the beginning of the learning using the neighborhood function and gradually it is more confined. A large value of neighborhood size may not provide a convergence or it may take a considerable amount of time, whereas a small value of size may lead to a wrong ordering of the neurons. The second parameter i.e., the learning rate, decides the amount by which each weight changes with every learning iteration. The learning rate is a constant and must be a positive number less than 1 and is generally set in the range of 0.1 – 0.5 (here most of the scenarios were tried with 0.1). A large value of learning rate will cause the training to progress faster but here too a very large value can result in no convergence. This is because the oscillations of the weight vector will be too large for any classification.

In practice, a vector represents the neurons in the network, which is a matrix of nature M X N where M denotes the number of neurons and N the dimension of the input data. The neighborhood relation, as described earlier, connects each neuron to the adjacent neurons. In each training set, there will be one neuron that wins (as it involves competitive learning) and the weight of this winning neuron is adjusted in such a way so that it will react more strongly when a similar input is provided next time (refer to section 3.4 for weight adjustments). In all the examples that have been presented below with the above set of parameters in place we found 80 to 100 iterations were adequate to train the networks under consideration. It must be pointed out that this number would vary as we try to train them in a different input size (or domain size) and with a different number of neurons in place. The main constraint involved in the Kohonen network is that the network weights should be normalized and each of the inputs should fully use the range (the limits of the problem space). In all the examples we tested, the maximum limiting range of the input space was [0,2] X [0,2], therefore the neurons were normalized within this limit. In practice along with this normalization factor, we also ensured that the weights initially had small and random values; this was done to avoid any condition of non convergence or slow training cycles, as a wide initial random spread may lead to input vectors falling into clusters over a limited region.

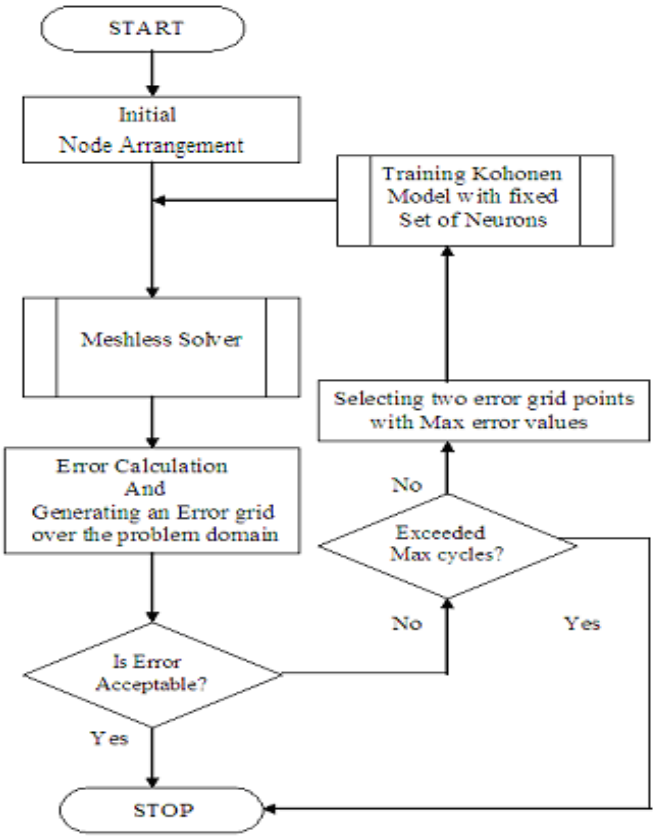


Fig 22

Flow chart of the proposed algorithm.

We now explain the manner in which the Kohonen and the meshless models were integrated. The thesis to this point has shown the manner in which we proceeded with this work. The first goal was to have a working meshless model in place. The objective when a problem domain with defined boundary conditions was presented was to distribute nodes over the domain and solve the problem; at the end of this process, we will have a solution vector of field values (with reasonable accuracy). We will then calculate the error distribution over the domain, which will form the data set (referred as the data cloud) on which the Kohonen network will work as shown in Fig 22.

For the error distribution, we used a simple approach in which the error at a particular node was estimated by considering the average value of potential of the four adjacent nodes (we may even use a different stencil here provided the averaging points are close to the point of estimation), and then we compared the average value with the potential computed at the reference node (Fig 23). Referring to Fig 22, after the error computation the algorithm has an accuracy check in place (represented by the decision block) set to a tolerance value of 1×10^{-8}, a root mean square (R.M.S) estimate of the error

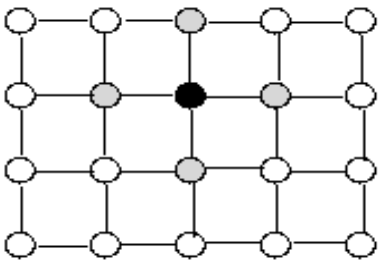


Fig 23

The error map. The node shaded dark is the reference node, to estimate the error at this coordinate the average potential value of the adjacent nodes shaded in grey is considered and then compared with the value at the reference node.

values is used to compare with the tolerance value. Next there is a check based on the number of iterations or cycles to allow termination of the process in the event of non convergence. If the output of the decision is no, then the algorithm identifies the points of maximum error and these coordinates are used to "train" the Kohonen network and, in the process, the nodes are moved towards the high error positions. On completion of the training, the new arrangement of neurons (nodes) is fed to the meshless solver and the same cycle as described above is repeated till there is a convergence.

4.4 Experiments and Results

We now present the results from the experiments that we carried out using this algorithm with the meshless models described in the preceding section.

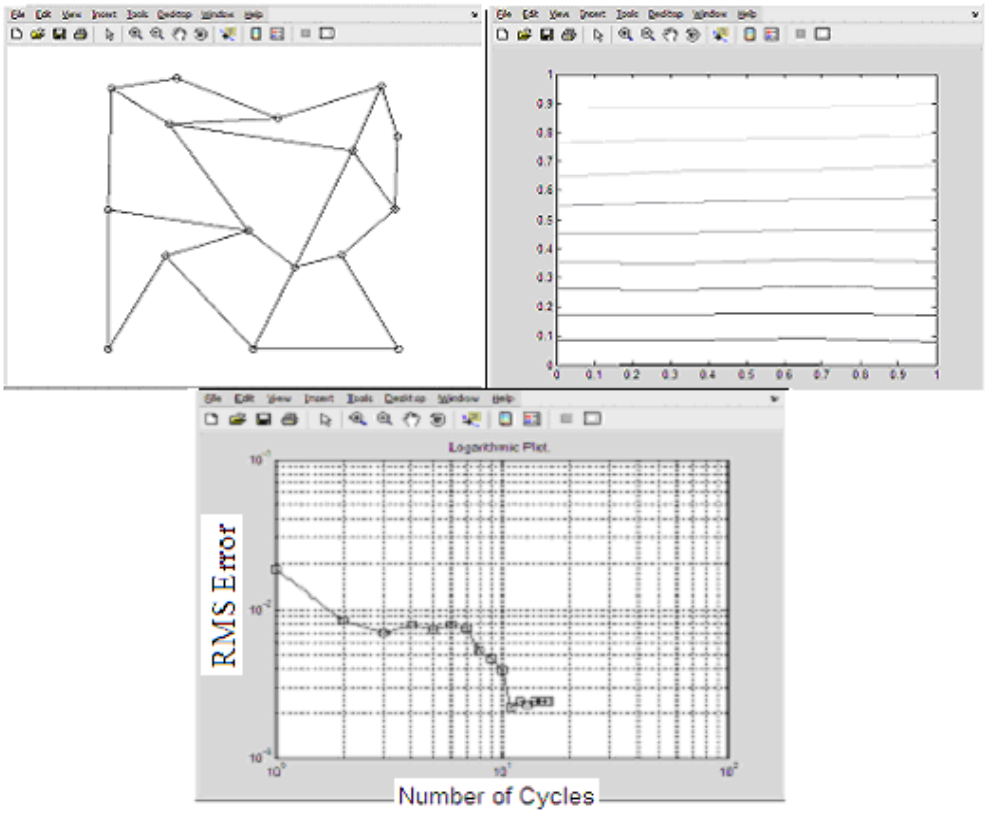


Fig 24

Final positions of the 16 nodes and the field solution along with the convergence plot at the bottom.

Fig 24 represents the result when we tried the algorithm with a meshless model for the problem presented in Fig 10. We can find a convergence and the system optimizes the positions of the nodes with the R.M.S error being 2.40E-03.

In the next example we tried the algorithm with the model presented in figure 16 (Fig 16), the “L” domain problem. The R.M.S error at the end of the training was 3.91E-004.

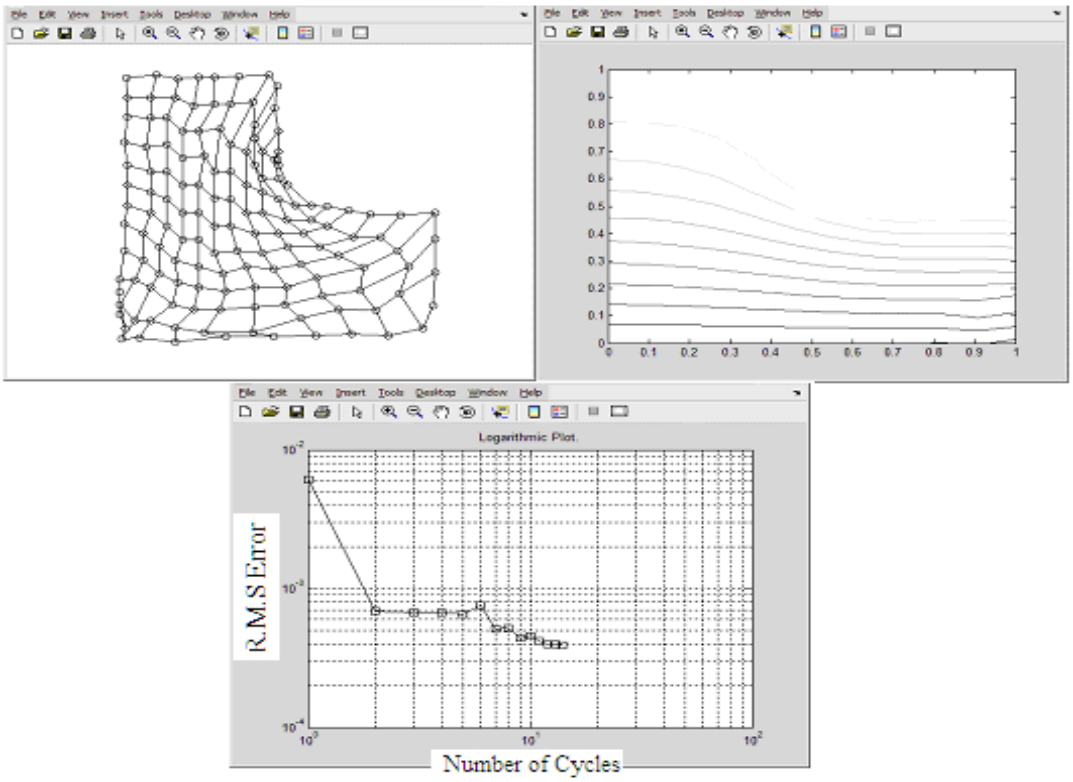


Fig 25

Final positions of the 144 nodes and the field solution along with the convergence plot at the bottom.

4.5 Summary

This chapter clarified the procedures involved in designing meshless and Kohonen models and presented the algorithm that was used to make them work together. To demonstrate that the two models are effective in handling problems in electromagnetics a number of cases were presented but a more rigorous investigation using the ideas introduced so far is required in order to establish it as a reliable and a robust technique. The Appendix D provides a comparative detail pertaining to a simple scenario between

FEM and MM approaches. This work has presented examples that involved the use of collocation technique for solving the field problems, however as mentioned earlier, there are number of methods that are available, such as the ones based on integral approaches (refer to sections 2.2.1 and 2.2.3) which could have been developed to ascertain the best possible meshless solver for the above algorithm, but because of time constraints they were not investigated.

CHAPTER 5

5 Conclusion and Future work

The main objectives of this thesis work included 1) finding a more flexible method of representing the electromagnetic field problem that avoids the rigidity imposed by FEM – will meshless systems give us this? If yes, then to what extent? 2) once we have a solution, can the idea of a self organizing feature map (Kohonen network) allow us to move the nodes of a meshless system to where they can produce the most accurate solution, i.e., to provide us with the optimal nodal arrangement.

This thesis work introduced an algorithm that successfully uses meshless and Kohonen network systems, which can work in tandem to solve electromagnetic field problems effectively. The work addresses most of the objectives stated previously; it was demonstrated that the meshless approach is free from the issues and problems of re-meshing, which is prevalent in FEM, and the approach can provide results with good accuracy when solving electromagnetic field problems. In addition to this, it was also shown that Kohonen networks have the ability to move nodes of meshless systems, and thereby the network can provide an optimal nodal arrangement. The Appendix D provides some comparison for a simple scenario between the MM and the FEM approaches but a more detailed comparative study between the two methods is required; such a study could have involved verifying the efficiency (computational cost) between them. Another area for investigation could have been a comparison between the number of nodes required in MMs and the number of elements required in FEM to solve a

problem with the same level of accuracy, one can gain some idea from Appendix D. These studies would have helped us in clarifying to a greater extent the effectiveness of meshless method when compared to FEM for solving electromagnetic field problems.

From an overall perspective, the computational complexity as explained in sections 2.2.3 and 4.2 is the main shortcoming, this factor assumes significance as we have an adaptive system in place where we need to solve the system matrix repeatedly. A simple approach to generate a sparse matrix was presented in the work (section 4.2) but it was shown that accuracy levels were quite low when we tried to remove portions of RBF values from the system matrix. Another technique that involved scaling of the shape parameter provided us with good accuracy and also a sparse system matrix when tried on a problem with simple geometry; further investigations are required here to ascertain its viability in complex geometrical scenarios. The approach involving Kohonen network and the techniques that generate sparse matrices were not explored fully as it was thought it is wise to build a system first that tries to optimize a solution of high accuracy (involving a dense matrix) rather than one that tries to optimize a solution with less accuracy. To deal with real world scenarios computational efficiency has to be taken under consideration and for this we need to study the techniques for generating sparse matrices, one such approach that involved scaling of shape parameter has been discussed here and there are certain new techniques such as the CSRBFs [10], [43] that hold a lot of promise but they were not examined. Apart from addressing the issue of computational complexity, the major future work will include studies involving the coupling of FEM and meshless techniques for better accuracies and handling of non-linear conditions as highlighted

below. It has been discussed earlier in this work that in order to have good results at the boundaries involving the collocation approach we need to construct additional collocation points, such an approach can get very complex when we have moving objects, as in case of motors, or objects that change their geometrical shapes during computation. To avoid the issues at the boundaries while using meshless techniques researchers have suggested the use of a hybrid approach [44] wherein to implement the boundary conditions we use FEM and for approximation in regions associated with motion we use meshless techniques. Here we need to determine whether the present collocation approach (strong formulation) can be coupled with FEM to address problems of similar nature. This thesis work has dealt with scenarios that involve only linear conditions; we can also investigate problems involving non linear conditions which may involve careful study of RBFs in order to incorporate the material effects into the basis functions for precise approximation. This may involve basis functions based on material coordinates, or some form of tessellation where approximation is performed at a sub domain level rather than on a global scale. The use of the meshless methods to deal with nonlinear problems is still at a very nascent stage, and most of the published work has concentrated on the implementation of integral approaches, in particular the nodal integration approach for such scenarios [45], [46]. However, the FEM has a vast literature that concentrates on nonlinear problems; future work may involve the use of methods developed in FEM with some modifications, and use them in the meshless methods. We can also explore as to how we can use the meshless solver to handle 3 D geometries, which would be closer to the real world. Coming back to the proposed algorithm in the present work, the main issue that needs to be investigated is that of computational complexity and as stated

earlier there are few new techniques in the meshless world that use RBFs, which do hold some promise, we need to find whether they are good enough for solving electromagnetic field problems. If they are found effective, then we can try to integrate them with the existing Kohonen model developed in this work.

The techniques developed in this work provided a working concept that integrates the meshless method (collocation approach) and the Kohonen model to solve electromagnetic field problems. Certain issues, concerning the computational efficiency were also highlighted. Further development of this work will be to address the issue stated above in order to have a viable software system in place.

References

[1] G.R. Lui; "Meshfree Methods Moving beyond the Finite Element Method", CRC Press, pp 15, 19-24, 67-107, 2003.

[2] G.R. Lui and Y.T Gu; "An Introduction to Meshfree Methods and Their Programming". Springer, Chapters 2, 3, 4 and 6, 2005.

[3] Michael Griebel and Marc A Schweitzer; "Meshfree Methods for Partial Differential Equations I"., Springer., pp 1-20, 75-86,143-192, 2003.

[4] Satya N. Atluri & Shengping Shen; "The Meshless Local Petrov-Galerkin (MLPG) Method: A Simple & Less-costly Alternative to the Finite Element and Boundary Element Methods"., Tech Science Press CMES, vol.3, no.1, pp.11-51, 2002.

[5] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming and P Krysl; "Meshless methods: An overview and recent developments"., Elsevier. Comput. Methods Appl. Mech. Engrg., vol. 139, pp 3- 47, 1996.

[6] Martin D Bhuman; "Radial Basis Functions"., Cambridge University Press., pp 1-5, 48-65, 99-108, 2003.

[7] M. J. D. Powell; “Approximation Theory and Methods ”.,Cambridge University Press, Chapters 1-5,11,14,16,18,19, 20 and 24, 1982

[8] Robert Schaback “Error estimates and condition numbers for radial basis function interpolation”., Advances in Computational Mathematics 3, pp 251-264, 1995.

[9] I R H Jackson; “An Order of Convergence of Some Radial Basis Functions”., I M A Journal of Numerical Analysis, vol. 9, pp 567-587, 1989.

[10] Holger Wendland; “Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree”., Advances in Computational Mathematics, vol. 4, pp 389-396, 1995.

[11] G. R. Liu, Y. T. Gu; “A meshfree method: meshfree weak–strong (MWS) form method, for 2-D solids”., Springer-Verlag, Computational Mechanics, vol. 33. pp 2–14, 2003.

[12] M.A Martinez, E Cueto, M Doblare, F Chinesta; “Natural element meshless simulation of flows involving short fiber suspensions”., Elsevier., J. Non-Newtonian Fluid Mech., vol. 115, pp 51–78, 2003.

[13] C. Hkrault and Y. Markchal; "Boundary and Interface Conditions In Meshless Methods"., IEEE Transactions on Magnetics, vol. 35, No. 3, pp 1450-1453, May 1999.

[14] Y. X. Mukherjee, S. Mukherjee; "On boundary conditions in the element-free Galerkin method "., Springer- Verlag, Computational Mechanics, vol. 19, pp 264–270,1997.

[15] Michael Griebel and Marc A Schweitzer "Meshfree Methods for Partial Differential Equations III"., Springer., pp 57-76, 2003.

[16] Soichiro Ikuno, Katsuyuki Takakura, and Atsushi Kamitani; "Influence of Method for Imposing Essential Boundary Condition on Meshless Galerkin /Petrov– Galerkin Approaches"., IEEE Transactions on Magnetics, vol. 43, No. 4, pp 1501 -1504, April 2007.

[17] Maithili Sharan, Suman Gupta, E. J. Kansa; "Application of the Multiquadratic Method for Numerical Solution of Elliptical Partial Differential Equations"., Elsevier, Applied Mathematics and Computation, vol.84, pp 275-302, 1997.

[18] Simon Haykin; "Neural Networks a Comprehensive Foundation", Prentice Hall (second edition)., pp. 58 - 104, 443- 476, 1999.

[19] K.L Du and M.N.S; “Swamy Neural Network in Softcomputing Framework”. Springer., Chapters 1,2 and 5, 2006.

[20] James A. Freeman, David M. Skapura; “Neural Networks: Algorithms, Applications, and Programming Techniques”., Addison-Wesley., Chapters 2, 3, 5, and 7., 1991

[21] Christopher M Bishop; “Neural Networks for pattern recognition”, Oxford University Press.,pp 1-17,116-126, 1995.

[22] Teuvo Kohonen;“Self organization and associative memory”.,Springer-Verilag (second edition), 1988.

[23] Christos Christodoulou and Michael Georgiopoulos; “Applications of Neural Networks in Electromagnetics”. Artech House., Chapters 1, 3 and 10, 2001.

[24] Derek Dyck; “Determining Finite Element Mesh Density from Problem Specification using Neural Networks”., M.Eng dissertation McGill University, 1990.

[25] G.K. Miti, A.J. Moses, N. Derebasi, D. Fox; “A neural network based tool for magnetic performance of toroidal cores”., Elsevier., Journal of Magnetism and Magnetic Materials, pp 262-264, 2003.

[26] A. A. Adly, S. K. Abd-El-Hafiz;"Automated Two-Dimensional Field Computation in Nonlinear Magnetic Media Using Hopfield Neural Networks"., IEEE Transactions on Magnetics, vol. 38, No. 5, pp 2364 – 2366, September 2002.

[27] J. Seguin, F. Dandurand, J.K. Sykulski and D.A. Lowther; "The optimisation of electromagnetic devices using a combined finite element/ neural network approach with on-line training"., COMPEL, vol. 18, No. 3, pp. 266-274, 1999.

[28] Bernhard Lang; "Artificial Neural Networks: Formal Models and Their Applications - ICANN 2005"., 15th International Conference Proceedings, Part II. Springer., pp 325-330., 2005.

[29] Robert Schaback and Holger Wendland; "Kernel techniques: From machine learning to meshless methods"., Cambridge University Press, Acta Numerica, pp 1-97 ., 2006.

[30] Miklós Hoffmann; "Numerical control of kohonen neural network for scattered data approximation"., Springer. Numerical Algorithms, vol. 39**,** pp 175–186, 2005.

[31] Nicola Bianchi; "Electrical Machine Analysis using Finite Elements."., CRC Press., Chapters 2, 3 and 4., 2005.

[32] A. I. Fedoseyev; "Improved Multiquadratic Method for Elliptical Partial Differential Equations via PDE collocation on the boundary"., Pergamon. Computers and Mathematics with Applications, vol.43, pp 439-455, 2002.

[33] Nathan Ida; "Engineering Electromagnetics"., Springer. Chapters 1, 2, 5,6, 8,9 and 11., 2003.

[34] Frederico G. Guimarães, Rodney R. Saldanha, Renato C. Mesquita, David A. Lowther, and Jaime A. Ramírez; "A Meshless Method for Electromagnetic Field Computation Based on the Multiquadric Technique"., IEEE Transactions on Magnetics, vol. 43, No. 4, pp 1281-1284, April 2007.

[35] Kok-Meng Lee, Qiang Li, and Hungson Sun; "Effects of Numerical formulation on Magnetic Field Computation Using Meshless Methods"., IEEE Transactions on Magnetics, vol. 42, No. 9, pp 2164-2171, September 2006.

[36] Qiang Li; "Effects of Adaptive Discretization on Numerical Computation using Meshless Method with Live-object Handling Applications" Phd dissertation Georgia Institute of Technology 2007.

[37] S. L. Ho, S. Yang, J. M. Machado, and H. C. Wong; "Application of a Meshless Method in Electromagnetics"., IEEE Transaction on Magnetics, vol. 37, No. 5, pp 3198 -3202, September 2001.

[38] S. A. Viana, D. Rodger, and H. C. Lai; "Meshless local Petrov-Galerkin method with radial basis functions applied to electromagnetics"., IEE Proc. Sci. Meas. Technol., vol. 151, No 6, pp 449 -451, November 2004.

[39] S. A. Viana, D. Rodger, and H. C. Lai; "Application of the Local Radial Point Interpolation Method to Solve Eddy-Current Problems"., IEEE Transactions on Magnetics, vol. 42, No. 4, pp 591 -594, April 2006.

[40] Nannapaneni Narayana Rao; "Elements of Engineering Electromagnetics"., Prentice Hall (fourth Edition)., Chapters 2,3 and 4., 1994.

[41] Holger Wendland; "Meshless Galerkin Methods using Radial Basis Functions"., Mathematics of Computation, vol. 68, pp 1521-1531, 1999.

[42] C S Chen, M. Ganesh, M. A Golberg, A.H. D Cheng; "Multilevel Compact Radial Functions Based Computational Schemes for some Elliptical Problems"., Pergamon, Computers and Mathematics with Applications, vol. 43,pp 359- 378, 2002.

[43] Leevan Ling, Roland Opfer, Robert Schaback; "Results on meshless collocation techniques"., Elsevier., Engineering Analysis with Boundary Elements, vol. 30, pp 247-253, 2006.

[44] Vlatko Cingoski, Naoki Miyamoto, and Hideo Yamashita; "Hybrid Element-Free Galerkin—Finite Element Method for Electromagnetic Field Computations"., IEEE Transactions on Magnetics, vol. 36, No. 4, pp1543-1547, July 2000.

[45] Jiun-Shyan Chen, Sangpil Yoon, and Cheng-Tang Wu; “Non-linear version of stabilized conforming nodal integration for Galerkin mesh-free methods”., International Journal for Numerical Methods in Engineering, vol. 53, pp 2587-2615, 2002.

[46] L. Kucherov, E. B. Tadmor, and R. E. Miller; “Umbrella spherical integration: A stable meshless method for non-linear solids”., International Journal for Numerical Methods in Engineering, vol. 69, pp 2807-2847, 2007

Appendix

Appendix A

Theorem: (Mairhuber – Curtis). If $\Omega \subset \Re^n$, $n \geq 2$ contains an interior point, then there exist no Haar spaces of continuous functions except for one-dimensional ones.

Definition: Let the finite-dimensional function space $\mathcal{B} \subseteq C(\Omega)$ have a basis $\{B_1, \ldots, B_N\}$. Then $\mathcal{B}$ is a Haar space on Ω if

$$\mathbf{det}\,(B_k(\chi_j)) \neq 0 \qquad (1)$$

for set of distinct $\chi_1, \ldots, \chi_N$ in Ω. Note that Haar space guarantees invertibility of an interpolation matrix.

Proof: Let $n \geq 2$ and consider $\mathcal{B}$ is a Haar space with $\{B_1, \ldots, B_N\}$ with $N \geq 2$. Then, by the definition of a Haar space.

$$\mathbf{det}\,(B_k(\chi_j)) \neq 0$$

for set of distinct $\chi_1, \ldots, \chi_N$.

Now consider a closed path P inside Ω. connecting only χ_1 and χ_2. We can exchange the positions of χ_1 and χ_2 by moving them continuously along the path P without interfering with any of the other χ_j. This means, however, that rows 1 and 2 of the above

determinant have been exchanged and the determinant has changed sign. Since the determinant is a continuous function of χ_1 and χ_2 we must have had at some point **det** = 0 along P (two identical rows). This is a contradiction.

Appendix B

Halton points are created from van der Corput sequences that are frequently used in quasi – Monte Carlo methods for multidimensional integration application. For constructing van der Corput sequences, every non-negative integer, n, is written uniquely using a prime base p i.e.

$$n = \sum_{j=0}^{k} a_j p^j \qquad (1)$$

where the coefficient a_j , is an integer such that $0 \leq a_j < p$.

The following examples will clarify the manner in which the van der Corput sequences are generated. We consider two prime bases i.e., p = 2 and 3 and demonstrate how any integer can be represented using them, the coefficients that we obtain from here will play an important role in constructing the Halton sequences as explained later.

With p = 3, the integer 10 can be represented as:

$$10 = 1.3^0 + 0.3^1 + 1.3^2$$

So the coefficients are $a_0 = a_2 = 1$ and $a_1 = 0$ with k = 2. For integers 9 and 8, they are represented as:

$$9 = 0.3^0 + 0.3^1 + 1.3^2$$

The coefficients are $a_0 = a_1 = 0$ and $a_2 = 1$ and again we have k = 2,

and $$8 = 2.3^0 + 2.3^1 + 0.3^2$$

where the coefficients are $a_0 = a_1 = 2$ and $a_2 = 0$ and the parameter k is 2.

Now we tryout with a different prime base i.e., p = 2 with the same set of above integers i.e., 10, 9 and 8

$$10 = 0.2^0 + 1.2^1 + 0.2^2 + 1.2^3$$

Here the coefficients are $a_0 = a_2 = 0$ and $a_1 = a_3 = 1$ and the parameter k is 3

$$9 = 1.2^0 + 0.2^1 + 0.2^2 + 1.2^3$$

for 9, the coefficients are $a_1 = a_2 = 0$ and $a_0 = a_3 = 1$ and the parameter k is 3

$$8 = 0.2^0 + 0.2^1 + 0.2^2 + 1.2^3$$

and the coefficients are $a_0 = a1 = a_2 = 0$ and $a_3 = 1$ and the parameter k is 3

We then proceed to define a function h_p (p being the prime base) that maps the non negative integers to the interval [0, 1) using

$$h_p(n) = \sum_{j=0}^{k} a_j / p^{j+1} \qquad (2)$$

where a_j are the coefficients obtained from the van der Corput sequences above

e.g., $h_3(10) = 1/3 + 1/3^3 = 10/27$ and $h_2(10) = 1/2^2 + 1/2^4 = 5/16$.

The first eleven elements for the prime bases 3 and 2 are:

$h_3(n) = \{0,1/3, 2/3, 1/9, 4/9, 7/9, 2/9, 5/9, 8/9, 1/27,10/27 \}$

$h_2(n) = \{0, 1/2, 1/4, 3/4, 1/8, 5/8, 3/8, 7/8, 1/16, 9/16, 5/16\}$

Where n = 0, 1, …. , 10. The above two sets provide us with the points for a two dimensional Halton sequence and for a unit square i.e., $[0,1]^2$ we just pair them up like (1/2, 1/3), (1/4, 2/3), (3/4, 1/9), (1/8, 4/9) and so on.

Thus, in order to generate the Halton point set in m dimensional space we consider m (usually distinct) primes $p_{1,\ldots,} p_m$ and use the resulting van der Corput sequences as the coordinate of m dimensional Halton points. The resulting set provides us with the Halton points as.

$$H_{m, N} = \{ (h_{p1}(n) , ... , h_{pm}(n)): n = 0, 1, …. ,N \} \quad (3)$$

Appendix C

To demonstrate the Kansa collocation approach we present a simple scenario similar to the problem presented in Fig 10

Basic function and the derivatives:

Gaussian RBF $\Psi(r) = e^{-(\alpha r)_2}$ (1)

$$\partial \Psi(r) / \partial r = -2 \alpha r e^{-(\alpha r)_2} . \quad (2)$$

In terms of x and y coordinates the above function and derivative can be written as

$$\Psi(x, y) = e^{-\alpha_2 (\{x - x_i\}^2 + \{y - y_i\}^2)} \quad (3)$$

$$\partial \Psi(x, y) / \partial x = -2 \alpha \Psi(x, y) e^{-(\alpha r)_2} (x - x_i). \quad (4)$$

$$\partial \Psi(x, y) / \partial y = -2 \alpha \Psi(x, y) e^{-(\alpha r)_2} (y - y_i). \quad (5)$$

The second order PDE for the equation (1) will be of the nature

$$\partial^2 \Psi(r) / \partial r^2 = 2 \alpha^2 e^{-(\alpha r)_2} (2(\alpha r)^2 - 1) \quad (6)$$

On applying the chain rule to equation (3) we can obtain the second order derivatives in terms of x and y.

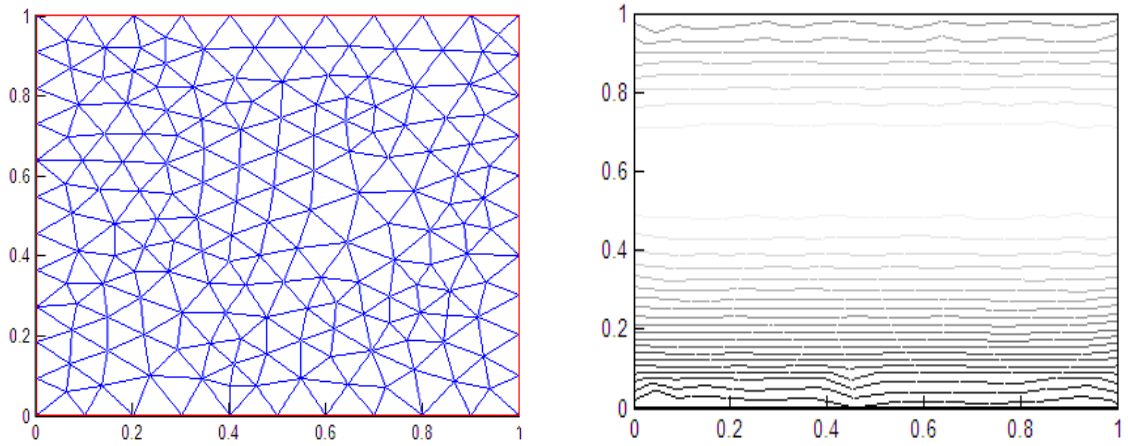
Most of the problems that have been presented in this work involve solving equations of the form $\nabla^2 \boldsymbol{u} = \boldsymbol{f}$, referred to as the governing equations.

In terms of collocation approach, (refer to section 2.2.4) we construct a linear equation of the form Ax = f. The components of such an equation i.e., A x and f are in matrix form where A is known as the collocation matrix, x carries the unknowns or the coefficients and f is known as the right hand side matrix. As mentioned in section 2.2.4 the collocation approach involves carefully constructing the collocation and the right hand side matrices. The collocation matrix is made by matching the governing equation (PDEs) and the boundary conditions. Considering the problem in figure 10, where we have a square problem domain with the governing equation $\nabla^2 \boldsymbol{u} = 0$ defining the interiors of the domain, the left and the right hand edges are defined by the homogenous Neumann condition of $\partial \boldsymbol{u} / \partial n = 0$ and the top and bottom edges by the Dirichlet condition where the value of u is known, in this instance the bottom edge has a value of 0 and the top has a value of 1 respectively. For constructing the matrices, we first consider the governing equation, on the left hand side since the equation involves the second order derivative we consider the equation (6). This equation (6) is applied to all the interior nodes. We then define a block on the right hand side matrix, which will apply the right hand condition of the governing equation, as we have a 0 value in the present scenario so all the entries corresponding to the interior nodes are 0. Then we approach systematically in defining the blocks for the boundary nodes in the collocation and the right hand side matrices, implying we deal with them in counter clockwise or clockwise order but not in some random order i.e., if the first boundary considered is the bottom edge then and it is a counter clockwise approach the next should be the right hand edge and then the top edge and so on. In the present case as the bottom edge involves the Dirichlet's condition of 0 we consider all the boundary nodes that are present at the bottom edge and apply

equation (1). This has been done specifically as per the Fedoseyev's guideline [32] in order to minimize errors on the boundary. Such a sub block thus formed is placed inside the collocation matrix just after the sub block formed by the use of the governing equation. In the right hand side matrix since the value at the bottom edge was 0, so all the entries corresponding to the nodes present at the bottom edge are set to 0. Then we treat in a similar way the right hand edge but this time as we have a Neumann condition in place, we use an equation involving the first order derivative such as equation (2) and place the sub block thus obtained just after the previous sub block. We repeat the procedure as explained above till we have defined all the boundaries in the system matrices. The coefficient matrix (x) can then be obtained by using QR decomposition involving the collocation matrix A and the right hand side matrix f. The manner in which the collocation matrix is formed does not always lead to a symmetric arrangement because of this the Kansa collocation approach is known as non–symmetric collocation technique.

Appendix D

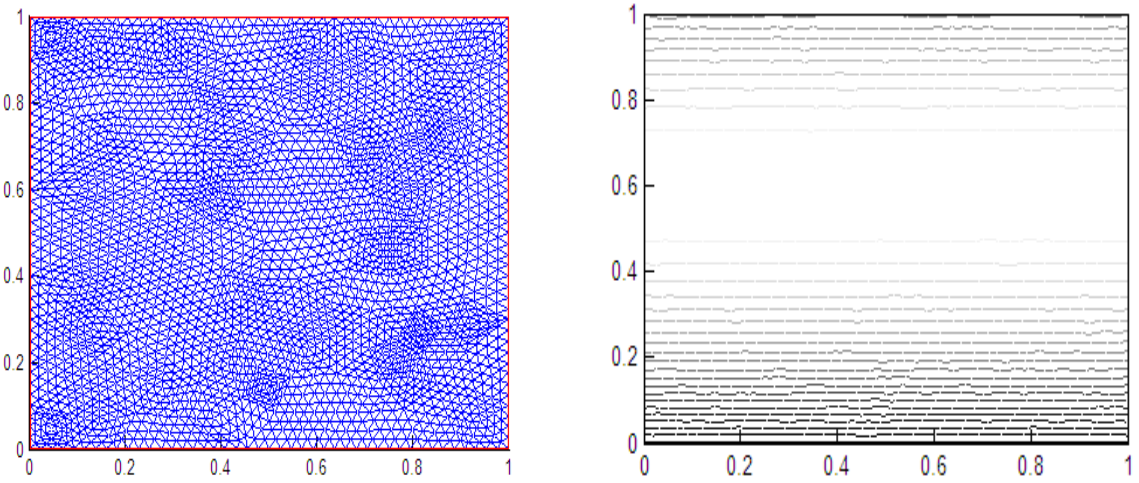
Following are the simulation results, when a scenario same as that presented in Fig 10 was tested using the FEM approach. The simulation was carried out using MATLAB's PDE toolbox.



a b

Fig 26

a) FEM mesh with nodes 191 nodes and 338 triangular elements b) Output from the FEM solver.



a b

Fig 27

a) FEM mesh with nodes 2789 nodes and 5408 triangular elements b) Output from the FEM solver.

Coordinates (x,y)	MM - 81	FEM - 191 nodes	FEM - 2789 nodes
0.5,0.5	0.53	1.12	0.81
0.5,1	1	1	1
0.2,0	0	0	0
0.25,0.85	0.84	0.91	0.85

Fig 28

Comparative chart between the MM approach and the FEM approach for the same field problem as in Fig 10.