2011

Minimal and Symmetric Global Partition Polynomials for Reproducing Kernel Elements

Mario Jesus Juha
University of South Florida, mariojesusjuha@gmail.com

Minimal and Symmetric Global Partition Polynomials for Reproducing Kernel Elements

by

Mario J. Juha

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Civil and Environmental Engineering
College of Engineering
University of South Florida

Date of Approval:
July 8, 2011

Keywords: Numerical analysis, interpolation, weak form, thin plates, continuity

Dedication

To God and His beloved son Jesus. To my mother who has been very supportive through all my life. To Paola and our baby, who have given me love and courage to complete this journey. To my family for trusting in me no matter what happens. To Professor Daniel Simkins who was a bright light in my moments of darkness.

Acknowledgments

I express my most sincere gratitude to the members of my committee. First of all to Professor Daniel Simkins for believing in me. To Professors Nathan Crane and Ryan Toomey for giving me support in difficult times. To Professor Andres Tejada-Martinez for all the useful conversation about many topics and for his wonderful teaching. To Professor Autar Kaw who has inspired me to become a better professional. Finally, I would like to give thanks to Professor David Rabson for his valuable comments.

Table of Contents

List of Tables

List of Figures

List of Symbols

$\underset{e\in\Lambda_E}{\mathbf{A}}$	Assembly operator
$\mathbf{x}_g$	Gauss point
δ	Kronecker delta
$\kappa_\rho\left(\bullet\,;\,\bullet\right)$	Kernel function
Λ_E	Set of all the elements in the domain
Λ_e	Set of all the vertex in an element
Λ_g	Set of Gauss points in an element
Λ_N	Set of all the nodes in the domain
(e,i)	Local index pair. Element and node number
$\mathscr{I}$	Interpolation/approximation operator
Ω	Problem domain
Ψ	Reproducing Kernel Element shape function
ψ	Global partition polynomial
$\Re$	Set of real numbers
ρ	Radius of support
ζ	Parent domain
C^k	k-times continuously differentiable function
d	Space dimension
n_{el}	Number of elements in the domain

n_{np}	Number of nodes per element
W_g	Gauss point weight, including the determinant of the Jacobian

Abstract

The Reproducing Kernel Element Method is a numerical technique that combines finite element and meshless methods to construct shape functions of arbitrary order and continuity, yet retains the Kronecker-δ property. Central to constructing these shape functions is the construction of global partition polynomials on an element. This dissertation shows that asymmetric interpolations may arise due to such things as changes in the local to global node numbering and that may adversely affect the interpolation capability of the method. This issue arises due to the use in previous formulations of incomplete polynomials that are subsequently non-affine invariant. This dissertation lays out the new framework for generating general, symmetric, truly minimal and complete affine invariant global partition polynomials for triangular and tetrahedral elements. It is shown that this new class of reproducing kernel element solves the asymmetry issue that affected previous developed elements. The interpolation capabilities of this new class of reproducing kernel elements is studied in problems of surface representations and in solving problems of bending of thin plates using a Galerkin approach. Optimal convergence rates were observed in the solution of Kirchhoff plate problems with rectangular domains. Furthermore, it is shown that the new proposed two-dimensional elements out perform the previous elements with the addition of only a few internal degrees of freedom.

Chapter 1

Introduction

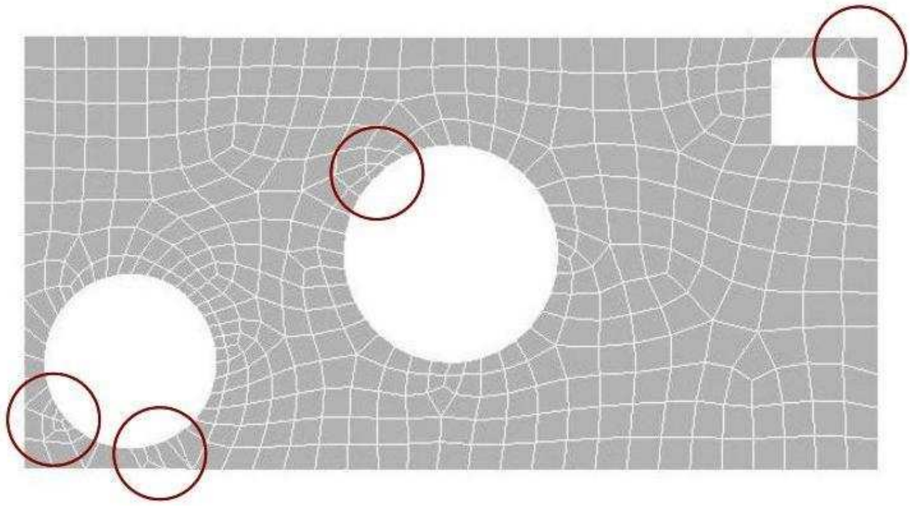
1.1 Motivation

The construction of functions k-times continuously differentiable and globally compatible in multiple dimensions was the main goal during the initial development of the Finite Element Method in the 1960's and 1970's before the idea was abandoned for its extreme difficulty [26]. Then, a question arose, *why do we want k-times continuously differentiable and globally compatible functions?* One answer is because we want to solve fourth-order differential equations, as in thin plate problems [54], using a Galerkin approach, and it will lead to an irreducible form that requires, at least, functions with continuous first derivatives [59]. Also, higher order interpolations associated with the space of functions k-times continuously differentiable, where $k \geq 1$, are believed to have better computational performance than the standard space of continuous functions [27]. The Finite Element Method (FEM) has been the *de facto* standard to solve the partial differential equations that appears in important problems in engineering and science. However, there are important problems where the Finite Element Method does not perform well due to the difficulty in the construction of globally and compatible k-times continuously differentiable space of functions. One of these areas is the formulation of effective and reliable plate elements, that require globally compatible functions with continuous first derivatives. Several attempts were carried out to develop those elements, the most important are cited in [2, 11, 13, 14, 24, 28]. The meshless community has attempted to develop interpolation schemes to solve the problem of bending of thin plates. Some of those schemes are cited in [31, 32, 34, 40, 56–58]. Their most important drawback is the difficulty in the imposition of Dirichlet boundary conditions on nodes. An excellent account of the problem is presented in [23]. The Reproducing Kernel Element Method (RKEM) [29, 33, 38, 42, 50], is the first method developed that has k-times continuously differentiable and globally compatible functions in multiple dimensions, yet retains the Kronecker-δ property on the nodes. Table 1.1 presents a comparison of the finite element method with the reproducing kernel element method. From the table aforementioned, we notice that both methods require an element mesh for the shape functions creation. Figure 1.1(a) shows a simple example of how meshing a domain with quadrilaterals elements, using an automatic mesh generator routine, introduces excessive aspect ratios (see Appendix B) and geometric distortions of the elements. The latter inevitably degrade the interpolation capability of the method. It can be explained mathematically in the context of the finite element method using the expression for the interpolation error [9], as follows,

$$\|\mathbf{u}-\mathbf{u}_h\|_1^2 \leq c \sum_m h_m^{2k} \|\mathbf{u}\|_{k+1,m}^2 \tag{1.1}$$

where m denotes an individual element, h_m is a measure of the size of the element, c is a constant independent of h_m, $\mathbf{u}$ represent the exact solution, $\mathbf{u}_h$ represent the interpolated function, k represent the order of the polynomial space used for interpolation and $\| \bullet \|_k$ are norms (see Appendix A). Hence, the total interpolation error is estimated summing the local contribution from each element. Therefore, regions with geometrically distorted quadrilateral elements will introduce considerable errors in the interpolated solution. On the other hand, the generation of a good quality quadrilaterals mesh (mapped mesh) will require the geometry to be built as a series of fairly regular volumes and/or areas that can accept a mapped mesh [1], undeniably, a time consuming task that will involve significant human interaction with the software. Figure 1.1(b) shows the same domain as before, but meshed using triangular elements, it can be seen that the automatic meshing generator routines provides triangular elements with good aspect ratio and small geometric distortions. Then, *why linear triangular elements are not recommended in finite element?* Because, it is well known that linear triangular elements have poor performance solving important engineering problems, for example, they suffer from "locking" phenomena in bending dominated problems [59], and therefore some solution variables are under predicted, unlike quadrilaterals elements. Whereas, the Reproducing Kernel Element Method based on triangular elements is not affected by the same finite element issue just mentioned. Although, a good aspect ratio of elements in the mesh is preferred in the RKEM.

Another field where RKEM was explored, was geometry representation, modification and iterative design [18, 49]. In fact, an asymmetry issue with the geometry representation of the circle was detected in [49] and it motivated the development of a new class of RKEM elements with symmetric behavior and truly minimal degrees of freedom.


(a) Quadrilateral elements. Red circles shows region with poor element quality.

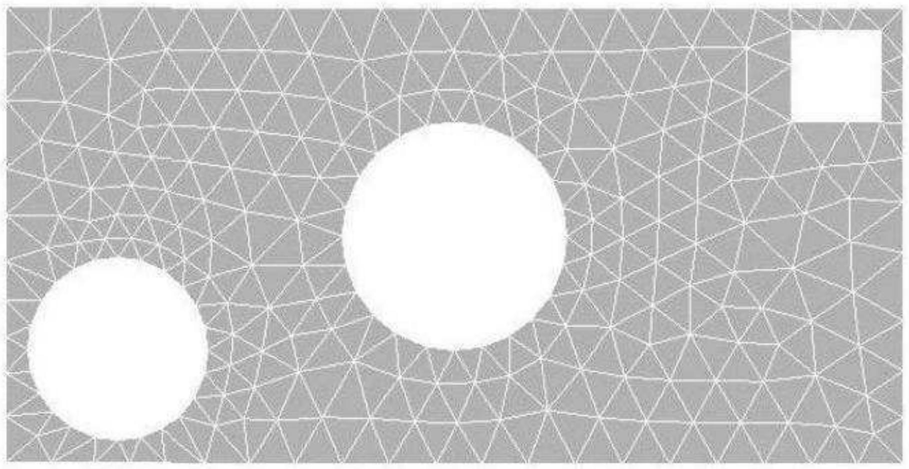

(b) Triangular elements.

Figure 1.1 Domain meshed using an automatic meshing generator program.

Table 1.1 Comparison between Finite Element and Reproducing Kernel Element (adapted from [36]).

	Items	**Finite Element**	**Reproducing Kernel Element**
1.	Element mesh	yes	yes
2.	Mesh creation and automatization	Difficult for quadrilaterals and hexahedral elements	Relatively easy for triangular and tetrahedral elements
3.	Shape function creation	Element based	Nodal and element based
4.	Shape function property	Satisfy Kronecker-δ condition. C^0 functions.	Satisfy Kronecker-δ condition. C^k functions; $k \geq 0$.
5.	Stiffness matrix	Symmetrical	Symmetrical
6.	Imposition of essential boundary conditions	Easy and standard	Easy but weakly enforced
7.	Post-processing of derived variables	Special technique required	Direct interpolation and differentiation
8.	Stage of development	Mature	Infancy
9.	Commercially available software	Many	None

1.2 Objective

The goal of research in computational mechanics is to develop methods to assist in the engineering design process, using computational approaches, to characterize, predict, and simulate physical events in engineering systems. An important ingredient of the computational approach is the generation of a space of basis functions that are capable of representing solution variables in a Galerkin weak formulation. One of these methods is the well known finite element method. However, the finite element method has its limitations in solving important practical problems in engineering and therefore, other methods have evolved to address its weaknesses. One of these weaknesses is the construction of a space of basis functions k-times continuously differentiable and globally compatible in multiple dimensions. The Reproducing Kernel Element Method is a relatively new method that allows us the construction of such a space of basis functions by combining finite element and meshless methods. An important ingredient of the Reproducing Kernel Element Method is the construction of the so-called global partition polynomial, therefore, a robust framework for constructing the global partition polynomials is crucial for the success of the method.

The goal of this dissertation is to lay out the framework for generating general, symmetric and truly minimal global partition polynomials necessary to construct a space of of basis functions k-times continuously differentiable and globally compatible in multiple dimensions and evaluate their performance in interpolation and in solving bi-harmonic differential equations that are difficult or even impossible to solve using the well established finite element method.

Chapter 2

Background

2.1 Interpolation Using Reproducing Kernel Element Shape Functions

An important application area for RKEM is the interpolation of a set of points. According to [55], we can distinguish two types of fitting, *interpolation and approximation*. The difference is interpolation goes through the points, and approximation does not. Methods like finite element and finite differences use the concept of interpolation, on the other hand, popular meshless methods use the concept of approximation. This difference has an enormous impact on the application of the methods. In this chapter we introduce the interpolation capability of the reproducing kernel element method.

2.2 Concept of Shape Function in Reproducing Kernel Element Method

A shape function is the name given to a collection of functions used to interpolate or approximate a data set. The success of an interpolation/approximation depends on the shape functions chosen. In general we want a method capable of representing a function at some point just using the information in its vicinity. A number of ways to construct shape functions have been proposed and we can find a considerable literature about them. Liu [36] classifies these methods in three major categories:

1. Finite integral representation methods where the function is represented using its information in a local domain via an integral form,

$$f(x) \approx \int_{x_1}^{x_2} f(\xi)\, \kappa_\rho (x-\xi;x)\, d\xi \tag{2.1}$$

   where $\kappa_\rho (x-\xi;x)$ is known as the kernel and ρ as the support size. Figure 2.1 is a pictorial definition of the support size for a two-dimensional space with a circular support. The solid circles represents nodes location and $\mathbf{x}_Q$ is an evaluation point. The kernel is evaluated for the nodes that are inside the circle of radius ρ. A mathematical definition of the support size is given in §2.2.3.2.

2. Finite series representation methods where the function is interpolated using a basis of functions, as follows,

$$f(x) \approx a_0 + a_1 p_1(x) + a_2 p_2(x) + \cdots \tag{2.2}$$

   where $p_1(x), p_2(x), \ldots$ are functions that depend on the nodes position but not on the constants $a_0, a_1, \ldots$ Usually the functions are constructed using a Lagrange interpolation, but it is not limited to them.

3. Finite differential representation methods where the function is approximated as a finite sum of terms that are calculated from the values of the function's derivatives at

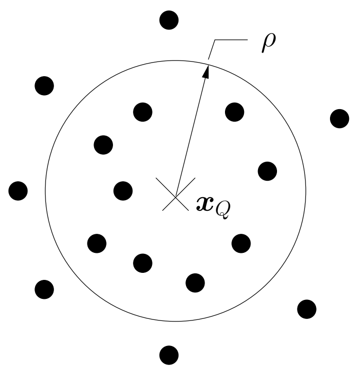


Figure 2.1 Support size.

a single point,

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2!} f''(x_0)(x - x_0)^2 + \cdots \tag{2.3}$$

where x_0 is an arbitrary point inside the interval of interpolation, and $f'(\bullet), f''(\bullet), \cdots$ are the derivatives of the function to represent.

Mathematically the finite differential representation method can be seen as a particular case of the finite representation method. However, conceptually they are differents, the former can be seen as a Lagrange interpolation and the latter is an application of the Taylor's theorem [30]. An important property when we deal with interpolation is the Kronecker-δ property. It definition is given below,

Definition 2.2.1 (Kronecker-δ property) *Let $\phi_I(\mathbf{x})$ denote a function evaluated at point $\mathbf{x} \in \Re^d$, for the node I, where, $I = 1, 2, \ldots, N$, being N the number of nodes in the domain. Let $\mathbf{x}_I \in \Re^d$ denote the location of the node I. Then, the Kronecker-δ property means,*

$$\phi_I(\mathbf{x}_J) = \delta_{IJ} \qquad I, J = 1, 2, \ldots, N \tag{2.4}$$

where,

$$\delta_{IJ} = \begin{cases} 1 & \textit{if} \quad I = J \\ \\ 0 & \textit{if} \quad I \neq J \end{cases} \tag{2.5}$$

The reproducing kernel element shape function is closely related to the *finite integral representation* and furthermore it has some distinguishing features [38]:

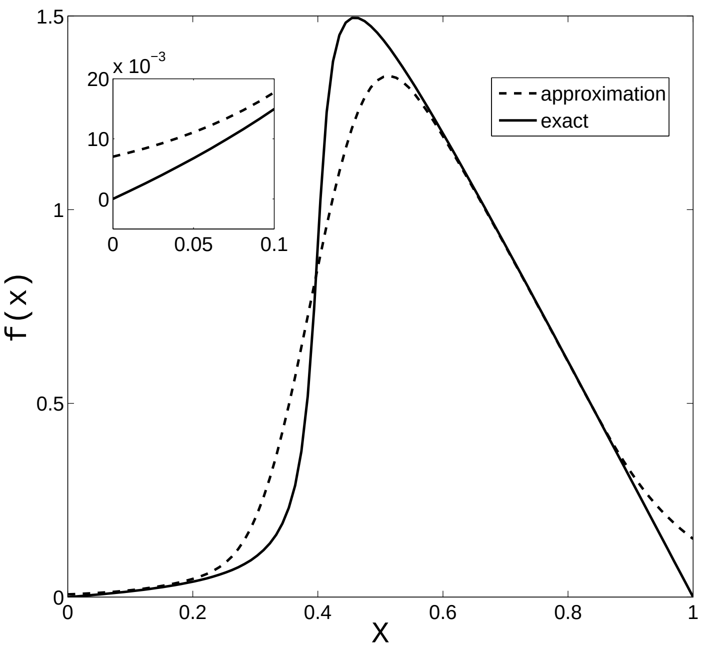
1. The smoothness of the global basis functions are solely determined by that of the kernel function. Smoothness in this dissertation means functions that are continuous and at least one derivatives exist and is continuous [26].

2. The global basis functions of RKEM have the Kronecker delta property at the associated nodes, provided that some conditions on the support size of the kernel function are met [17].

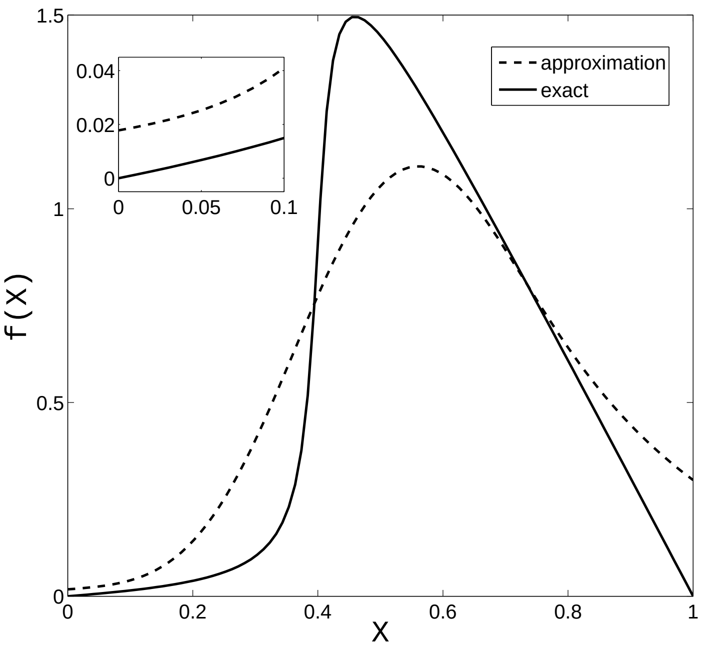
To understand the finite integral representation and its associated problem, we show an example in Fig. 2.2. The exact function we want to represent is given by

$$f(x) = (1 - x)\left[\arctan\left(\alpha\left(x - \bar{x}\right)\right) + \arctan\left(\alpha\bar{x}\right)\right] \tag{2.6}$$

where $\alpha = 50.0$ and $\bar{x} = 0.40$. From Fig. 2.2 it can be seen that the approximation is continuous and smooth. At the same time, the support size has a direct impact on the approximation function accuracy. The function shown in Eq. (2.6) is zero when $x = 0$ and $x = 1$, that is, $f(0) = f(1) = 0$. We have placed nodes at both ends, and we expect that if the approximation has the Kronecker-δ property, then when the approximation formula



(a) Kernel with $\rho = 0.2$



(b) Kernel with $\rho = 0.6$

Figure 2.2 Comparison between exact and approximate function for two different kernels using the finite integral approximation.

is evaluated at those nodes, its value must be zero. But that is not the case. Most of the meshless method are based, in one way or another, on a finite integral representation, therefore they will inherit the lack of Kronecker-δ property. Whereas, the reproducing kernel element method has the Kronecker-δ on the desired nodes, as it will be explain in §2.2.3.3.

To clarify the subsequent exposition of the RKEM interpolant some definitions about continuity, global partition polynomials and multi-index notations are essential.

Definition 2.2.2 (Continuity) *(Hughes [26]) A function $f : \Omega \to \Re$ is said to be k-times continuously differentiable, or class $C^k = C^k(\Omega)$, if its derivatives of order j, where $0 \leq j \leq k$, exist and are continuous functions.*

Definition 2.2.3 (Global Partition Polynomials) *(Li and Liu [35]) Consider a finite element discretization, $\Omega_e, e \in \Lambda_E := \{1,2,3,\ldots,n_{el}\}$ where n_{el} is the total number of elements. We assume that each element, Ω_e, has n_{np} number of vertexes, or nodes. We further assume that there are linearly independent functions $\{\psi_{e,i}\}, i \in \Lambda_e := \{1,2,3,\ldots,n_{np}\}$ and such that the following reproducing property of order k holds:*

$$\sum_{i \in \Lambda_e} \psi_{e,i}(x) x_{e,i}^{\gamma} = x^{\gamma} \qquad \forall \gamma : |\gamma| \leq k,\ \forall x \in \bar{\Omega} \tag{2.7}$$

Where, $\psi_{e,i}$ are globally defined polynomial functions, we call them global partition polynomials.

The global partition polynomial could be seen as finite element shape functions that are extended beyond the element domain, and therefore the adjective "global" was added to the name. From definition 2.2.3, it is clear that the global partition polynomial should form a partition of unity in Λ_e if the reproducing property of order zero is enforced. The proof is self evident, because

$$\sum_{i\in\Lambda_e} \psi_{e,i}(x) x_{e,i}^0 = x^0, \quad \forall x \in \bar{\Omega} \quad \Longrightarrow \quad \sum_{i\in\Lambda_e} \psi_{e,i}(x) = 1, \quad \forall x \in \bar{\Omega} \tag{2.8}$$

Definition 2.2.4 (Multi-Index Notation) *(Li and Liu [35]) Let $\mathbf{Z}^d$ denote the set of all ordered d-tuples of non-negative integers. A multi-index is an ordered collection (d-tuples) of d non-negative integers, $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_d)$, and its length is defined as*

$$|\alpha| = \sum_{i=1}^{d} \alpha_i \tag{2.9}$$

we write $\alpha! = \alpha_1!\alpha_2!\cdots\alpha_d!$ and $\mathbf{x}^\alpha = x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_d^{\alpha_d}$. For a differentiable function $u(\mathbf{x})$ and any α with $|\alpha| \le p$,

$$D^\alpha u(\mathbf{x}) = \frac{\partial^{|\alpha|} u(\mathbf{x})}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \tag{2.10}$$

is the αth order partial derivative. As usual, $D^0 u(\mathbf{x}) = u(\mathbf{x})$.

In addition to the reproducing condition (2.7), the global partition polynomial has the Kronecker-δ property, as has been proved in [33]

$$D^\alpha \psi_{e,i}^{(\beta)} \Big|_{\mathbf{x}=\mathbf{x}_j} = \delta_{ij}\delta_{\alpha\beta}, \quad \mathbf{x}_i,\, \mathbf{x}_j \in \Omega_e, \quad |\alpha|, |\beta| \le m \tag{2.11}$$

2.2.1 The Reproducing Kernel Element Interpolant

The reproducing kernel element interpolant is a hybrid between the finite integral representation and the global partition polynomial that produces globally compatible shape functions of arbitrary degree. The RKEM shape functions could be used for interpolation, approximation or even approximation/interpolation at the same time, provided some conditions on the kernel function are met [17]. This interpolation/approximation capability is because we are free to assign the Kronecker-δ property to the desired nodes in the domain, as it is defined in [17]. In [38] the Reproducing Kernel Element interpolant, $\mathscr{I}$, is defined as the operator acting on a continuous function $f \in C(\Omega)$ such that

$$\mathscr{I} f(\mathbf{x}) = \sum_{n=1}^{n_{el}} \left[\int_{\Omega_n} \kappa_\rho (\mathbf{y} - \mathbf{x}; \mathbf{x})\, d\mathbf{y} \sum_{i=1}^{n_{np}} \psi_{n,i}(\mathbf{x}) f(x_{n,i}) \right] \tag{2.12}$$

A more familiar way to look at Eq. (2.12) is obtained by expressing the interpolation as a linear combination of shape functions and nodal weights,

$$\mathscr{I} f(\mathbf{x}) = \sum_{I \in \Lambda_N} \Psi_I(\mathbf{x}) f_I \tag{2.13}$$

where Λ_N is the set of all the nodes in the domain, f_I is the nodal weight corresponding to node I and $\Psi_I(\mathbf{x})$ is the reproducing kernel element shape function at node I, and that can be expressed as

$$\Psi_I(\mathbf{x}) = \sum_{k \in \Lambda_E} \left[\left(\int_{\Omega_k} \kappa_\rho (\mathbf{y} - \mathbf{x}; \mathbf{x})\, d\mathbf{y} \right) \psi_{e_k, i_k}(\mathbf{x}) \right] \tag{2.14}$$

where, $(e_1, i_1), \cdots, (e_k, i_k) \rightarrow I$, is the local index pair that define a connectivity map. Note

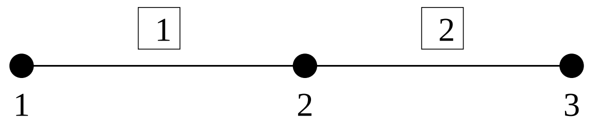


Figure 2.3 Spatial discretization with two elements and three nodes.

that we have specified in Eq. (2.14) a summation over all the elements in the domain. However, in practice, it is not required because the function, Eq. (2.14), is compactly supported; but special care should be exercised in order to precompute the set of elements that participate in the summation, as is shown in [17]. To understand better Eq. (2.13) the following one-dimensional example is presented. Considering the one-dimensional mesh in Fig. 2.3 and using Eq. (2.12) to interpolate the generic function $u(x)$, we arrive at

$$u(x) = \left[\int_{\Omega_1} \kappa_\rho (y-x;x)\, dy\right] \left[\psi_{1,1}(x)\,\hat{u}_1 + \psi_{1,2}(x)\,\hat{u}_2\right] + \left[\int_{\Omega_2} \kappa_\rho (y-x;x)\, dy\right] \left[\psi_{2,1}(x)\,\hat{u}_2 + \psi_{2,2}(x)\,\hat{u}_3\right] \quad (2.15)$$

where $\hat{u}_i$ corresponds to the value of the function to interpolate evaluated at the node i. Doing some algebraic manipulations we get the desired representation.

$$u(x) = \sum_{I=1}^{3} \Psi_I(x)\,\hat{u}_I \quad (2.16)$$

where,

$$\Psi_1(x) = \left[\int_{\Omega_1} \kappa_\rho (y-x;x)\, dy\right] \psi_{1,1}(x) \quad (2.17)$$

$$\Psi_2(x) = \left[\int_{\Omega_1} \kappa_\rho (y-x;x)\, dy\right] \psi_{1,2}(x) + \left[\int_{\Omega_2} \kappa_\rho (y-x;x)\, dy\right] \psi_{2,1}(x) \quad (2.18)$$

$$\Psi_3(x) = \left[\int_{\Omega_2} \kappa_\rho(y-x;x)\,dy\right] \psi_{2,2}(x) \tag{2.19}$$

Notice that the structure of the previous shape functions are also common in the finite element method. The main difference is the replacement of the integration of the kernel by a Heaviside step function,

$$\chi_e(x) = \begin{cases} 1 & x \in \Omega_e \\ \\ 0 & x \notin \Omega_e \end{cases} \tag{2.20}$$

The purpose of the Heaviside step function is to truncate the polynomial basis such that the approximation has a compact support, its length is determined by the element size, and to enforce compatibility. The requirement of compatibility means that the function within the elements and across the element boundaries must be continuous [9]. The kernel achieves similar effect as the Heaside step function by the specification of a support size, such that, the kernel is different from zero inside the specified support, and zero otherwise. The differentiability of the RKEM shape function is dictated by the differentiability of the kernel [38, 51], since the global partition polynomials are infinitely smooth and therefore they posseses continuous derivatives of any order.

2.2.2 Globally Conforming C^n Reproducing Kernel Element Shape Functions

In §2.2.1 we introduced reproducing kernel element shape functions whose differentiability is solely determined by the continuity of the kernel. It is because the global partition polynomials are C^∞ functions, therefore, the limiting condition for the differentiability of the RKEM shape functions is given by the maximum derivative that exist and is continuous in the kernel. The existence of higher order derivatives does not mean that the functions constructed so far are globally compatible. In this section we will show how to construct globally conforming C^n reproducing kernel element shape functions using what was called in [33] globally conforming I^m/C^n hierarchy II. According to Li [33], we assume that there exist a set of Hermite type global polynomials, $\left\{\psi_{e,i}^{(0)}, \psi_{e,i}^{(1)}, \ldots, \psi_{e,i}^{(m)}\right\}$, such that within the element, e, they can reproduce λ^{th} order polynomials,

$$\sum_{i\in e}\left\{\psi_{e,i}^{(0)}(\mathbf{x})\,\mathbf{x}_i^{\lambda}+\lambda\,\psi_{e,i}^{(1)}(\mathbf{x})\,\mathbf{x}_i^{\lambda-1}+\cdots+\frac{\lambda!}{(\lambda-m)!}\psi_{e,i}^{(m)}(\mathbf{x})\,\mathbf{x}_i^{\lambda-m}\right\}=\mathbf{x}^{\lambda}, \quad |\lambda|\le k \tag{2.21}$$

A set of global Reproducing Kernel Element shape functions are constructed as follows,

$$\Psi_I^{(m)}(\mathbf{x})=\sum_{k\in\Lambda_I}\left[\left(\int_{\Omega_k}\kappa_\rho(\mathbf{y}-\mathbf{x};\mathbf{x})\,d\mathbf{y}\right)\psi_{e_k,i_k}^{(m)}(\mathbf{x})\right] \tag{2.22}$$

Using the previous framework, Eq. (2.13) transforms into,

$$\mathscr{I}f^{(m)}(\mathbf{x})=\sum_{I\in\Lambda_N}\left(\Psi_I^{(0)}(\mathbf{x})\,f_I+\Psi_I^{(1)}(\mathbf{x})\,Df\Big|_I+\cdots+\Psi_I^{(m)}(\mathbf{x})\,D^m f\Big|_I\right) \tag{2.23}$$

where $D^m f$ refers to the nodal value corresponding to the m-derivative of the function to interpolate. The main feature of this construction is that the continuity of the globally conforming RKEM shape function is given by the order of the global partition polynomials. This does not contradict what we have said at the beginning of this section, as long as, we understand that in the former case we construct non-conforming shape functions, but still differentiable functions, and in the latter, we have globally conforming shape functions. An excellent explanation of conforming and non conforming shape functions is given in [59] in the context of variational formulations.

2.2.3 Kernel

In this section we will discuss the basic properties of the kernel used in this dissertation. The concept of the kernel is related to the finite integral representation of a function and the Dirac delta function. The basic concept is as follows,

$$f(\mathbf{x}) = \int_{\Omega} \delta(\bar{\mathbf{x}} - \mathbf{x}) f(\bar{\mathbf{x}}) \, d\bar{\mathbf{x}} \tag{2.24}$$

where $\delta(\mathbf{x})$ is the Dirac delta function and has the following two properties

$$\int_{-\infty}^{+\infty} \delta(\zeta - x) f(\zeta) \, d\zeta = f(x) \tag{2.25}$$

$$\int_{-\infty}^{+\infty} \delta(x) \, dx = 1 \tag{2.26}$$

From a computational point of view, the Dirac delta function is not attractive because it is not a “function” in a strict sense. As a result, a modification of Eq. (2.24) is required in order to be used for practical computation. The idea is to mimic the properties of the Dirac delta function using what is called a kernel. Therefore, the desired properties for the kernel are

1. $\int_{\Re^d} \kappa_\rho(\mathbf{x})\ d\Omega = 1$

2. $\kappa_\rho(\mathbf{x}) \to \delta(\mathbf{x}), \quad \rho \to 0$

To achieve the two previous properties, a special form for the kernel was used

$$\kappa_\rho(\mathbf{z};\mathbf{x}) := \frac{1}{\rho^d} w\left(\frac{\|\mathbf{z}\|}{\rho}\right) b(\mathbf{x}) \tag{2.27}$$

where $\| \bullet \|$ is the Euclidean norm, $\mathbf{z}$ is the difference between the point $\mathbf{x}$, at which the kernel is centered and an arbitrary evaluation point. The function w is a compactly-supported smooth window function (§ 2.2.3.2), and $b(\mathbf{x})$ is a normalization factor that properly adjusts the integration of the kernel to one when other than exact integration is used. The evaluation of the normalizer follows from the requirement that the integral of the kernel over the domain must be equal to one. Therefore,

$$\left\{\int_\Omega \left[\frac{1}{\rho^d} w\left(\frac{\|\mathbf{z}\|}{\rho}\right) b(\mathbf{x})\right]\ d\mathbf{z}\right\} = 1 \tag{2.28}$$

$$b(\mathbf{x}) := \left\{\int_\Omega \left[\frac{1}{\rho^d} w\left(\frac{\|\mathbf{z}\|}{\rho}\right)\ d\mathbf{z}\right]\right\}^{-1} \tag{2.29}$$

If exact integration could be done in Eq. (2.29), then $b(\mathbf{x})$ should be equal to one. Nevertheless, in practice, it is difficult to use exact integration in Eq. (2.29), because the function to integrate is not available in simple form. For this reason, numerical integration should be used, as is shown in § 2.2.3.1, thus the requirement of the normalizer.

2.2.3.1 The Role of Nodal Integration

There are several ways to integrate numerically Eq. (2.29), the simplest one being, nodal integration. In fact, nodal integration was used in this dissertation to integrate the normalizer and kernel.

$$b(\mathbf{x}) := \left\{ \underset{e \in \Lambda_E}{\mathbf{A}} \left[\sum_{j \in \Lambda_e} \frac{1}{\rho^d} w \left(\frac{\left\| \mathbf{x} - \mathbf{x}_{e,j} \right\|}{\rho_{e,j}} \right) \Delta V_{e,j} \right] \right\}^{-1} \tag{2.30}$$

where $\underset{e \in \Lambda_E}{\mathbf{A}}$ is the assembly operator and denotes the summation over all the mesh [26], $\Delta V_{e,j}$ is an integration weight whose value is equal to the volume/area of the element divided by the number of nodes in the element, and $\mathbf{x}_{e,j}$ is the corresponding nodal point location.

2.2.3.2 Window Function

The concept of window function used in the kernel is inherited from the meshless community [35, 36]. In general, it satisfies the following criteria [51]

$$\begin{cases} w(\mathbf{x}) \in C^k\left(\Re^d\right), & k \geq 1 \\ supp(w) = B_\rho \\ w(\mathbf{x}) > 0, & \text{for } \|\mathbf{x}\| < \rho \\ \int_{B_\rho} w(\mathbf{x})\; d\Omega = 1 \end{cases} \tag{2.31}$$

where, for $\bar{\mathbf{x}} \in \Re^d$ and $\rho > 0$, a spherical ball with radius ρ is defined as the domain of influence of $\bar{\mathbf{x}}$,

$$B_\rho(\bar{\mathbf{x}}) = \left\{ \mathbf{x} \,\middle|\, \|\mathbf{x} - \bar{\mathbf{x}}\| \leq \rho, \quad \mathbf{x} \in \Re^d \right\} \tag{2.32}$$

A conical window function in one-dimension was selected for computation. The conical window function has the following simple form [7],

$$w(x) = \begin{cases} \left[1 - \left(\frac{x}{\rho}\right)^2\right]^m, & |x| \leq \rho \\ 0 & |x| > \rho \end{cases} \tag{2.33}$$

The proper choice of the natural number m depends on the continuity of the shape function. Without loss of generality, we will show it for a one-dimensional problem. Imagine that we are solving a one-dimensional linear elastic strain gradient problem [6]. As a result, the continuity of the second derivative of the reproducing kernel element shape function is required in a weak form of the problem. For this reason, the continuity of the second derivative of the window function is also required. If we want the window function to be continuous up to the m-derivative, then the derivative when we approach to a point from the left should be equal to the derivative when we approach from the right side, mathematically,

$$\frac{d^{(m+1)}}{dx^{(m+1)}} w(x) \bigg|_{x=z^+} = \frac{d^{(m+1)}}{dx^{(m+1)}} w(x) \bigg|_{x=z^-} \tag{2.34}$$

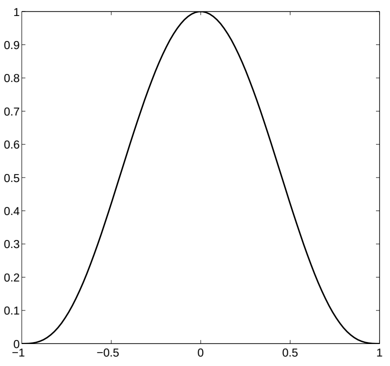
Equation. (2.34) is no more than an extension of the concept of the continuity of the first derivatives that are taught in basic calculus [3]. Going back to the specific case where the continuity of the second derivative is required and using Eq. (2.34) we arrive at

$$\frac{d^3}{dx^3}w(x) = \frac{12\,(m-1)\,mx\left(1-\frac{x^2}{\rho^2}\right)^{m-2}}{\rho^4} - \frac{8\,(m-2)\,(m-1)\,mx^3\left(1-\frac{x^2}{\rho^2}\right)^{m-3}}{\rho^6}$$

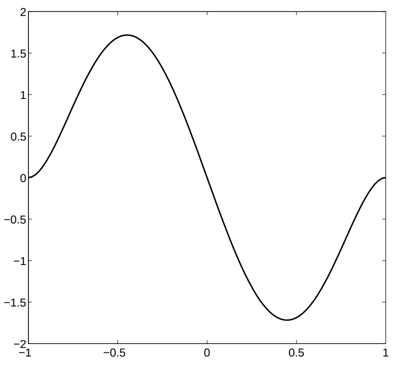
Note that the continuity of the second derivative of w is satisfied if $m > 3$. In general, we have found that the order of the conical window function must satisfy the condition $m > n+1$, where n is the desired continuity order in the weak form. In Fig. 2.4 we appreciate that if $m = 3$, then the continuity of the second derivative is lost at both end points, because a kink is formed at both extremes (Fig. 2.4(c)), unlike Fig 2.5 where the continuity in the second derivative is guaranteed for $m = 4$.

2.2.3.3 Geometry of the Support Domain

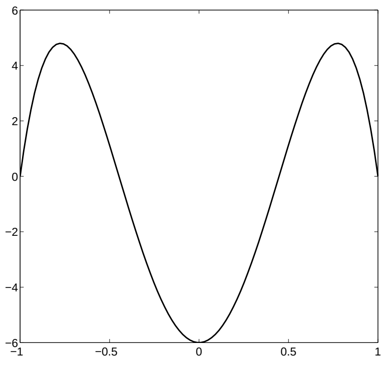
The geometry of the support domain plays an important role in the computation and integration of the shape functions [19–22, 34]. In the meshfree literature, the rectangular shape is dominant because it covers more area and the intersection of the domains is more likely to align with the Gauss points as a consequence of the background integration cells [22]. In this dissertation, we have chosen a circular shape for the geometry of the support domain, because of its symmetry and because our background cells are triangles anyway. We say that a node is *isolated* if it is not in the support of any other node [17]. The latter guarantee the Kronecker-δ property on the node. Figure 2.6 presents two possible geometries of the support domain; rectangular and circular. The mesh shown in Fig 2.6(a) has the Kronecker-δ property for the center node (for both shapes of the support domain),



(a) Window function

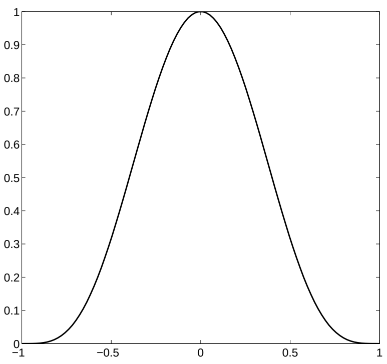


(b) First derivative

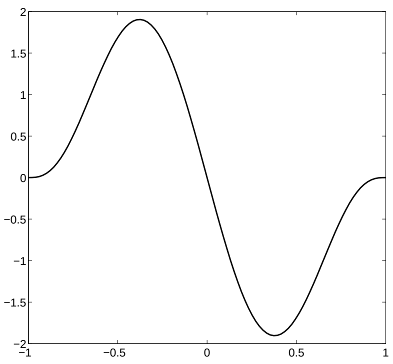


(c) Second derivative

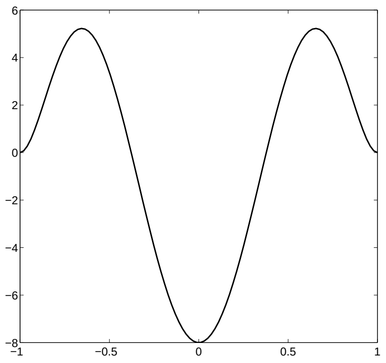
Figure 2.4 Window function and its derivatives for $m = 3$.



(a) Window function



(b) First derivative



(c) Second derivative

Figure 2.5 Window function and its derivatives for $m = 4$.

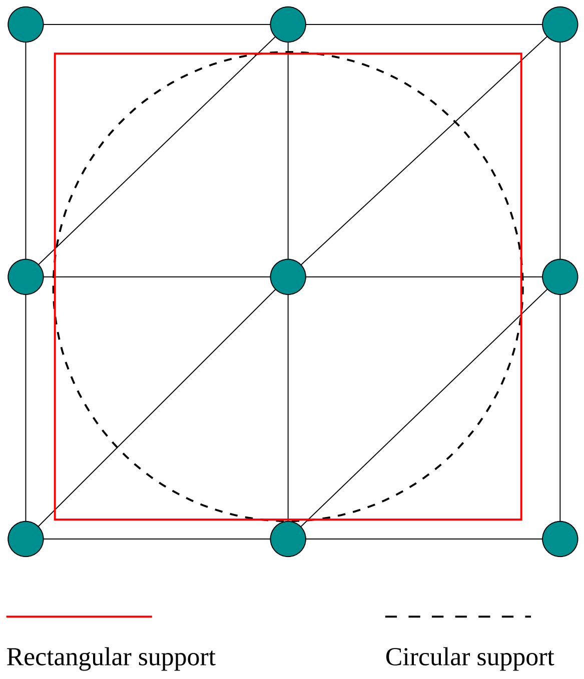
but in Fig 2.6(b) the rectangular shape fails maintaining the Kronecker-δ property for the nodes that are inside the rectangular support of the center node. Those nodes violates the definition of isolated node.

2.2.3.4 Numerical Integration

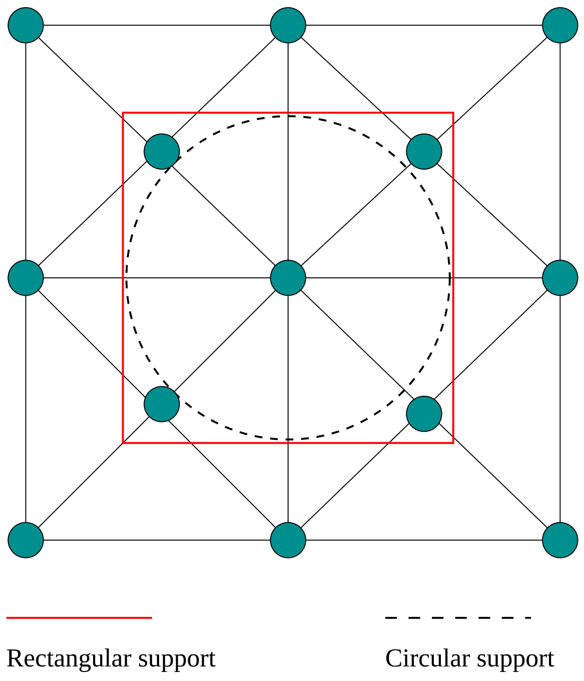
To integrate the weak form, we have chosen to use a higher order cubature rule for triangular domains [52]. In general, these cubature rules are not optimal, but there are no other high-order cubature rules available for triangular domains. We have found that to satisfy the patch test with reproducing kernel element interpolations, accurate evaluations of the derivatives and integrals in the weak form are essential. The same issues have been found in [10, 12, 15, 39, 44] for the evaluation of integrals in the Element Free Galerkin method (EFG). In addition, we face another important issue related to the integration of the stiffness matrix and force vector that appear in the weak form; that is, our domain of integration is not aligned with the intersection of the support domain of two nodes. To illustrate the last statement, consider a typical entry of the stiffness matrix for a two-dimensional problem as shown in Eq. (2.35).

$$K_{IJ} = \int_{\Omega} \Psi^{(00)}_{I,x}(\mathbf{x})\ \Psi^{(00)}_{J,x}(\mathbf{x})\ d\Omega \tag{2.35}$$

In Fig. 2.7 the shaded region represents a domain where the derivative of the shape function is different from zero and the red circle corresponds to Gauss point location. We could summarize the source of errors during the integration process:



(a) mesh A and support for center node.



(b) mesh B and support for center node.

Figure 2.6 Two square meshes with different geometry support domain.

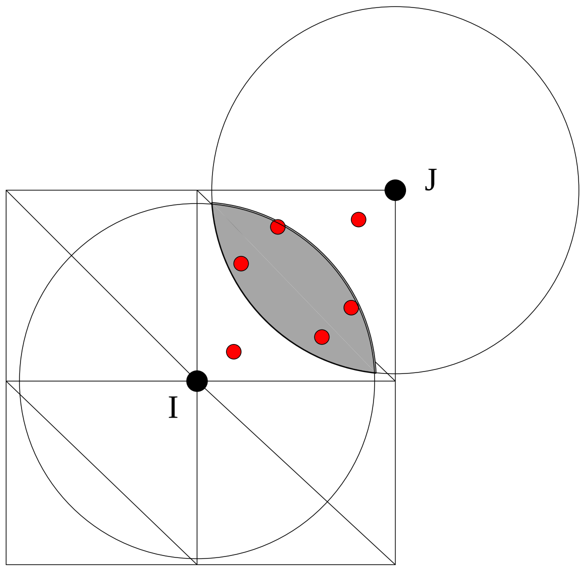


Figure 2.7 Integration issue in Reproducing Kernel Element.

1. The reproducing kernel element shape functions are not polynomial and they may have strong oscillations.

2. High-order cubature rules for triangular domains are not optimal.

3. The domain of integration is not aligned with the background cell and this affect negatively the accuracy of the Gauss point rule, in the sense that, we are *truncating* the Gauss point rule and therefore, changing its original degree of precision.

Previous drawbacks could be potentially eliminated by the implementation of exact integration formulas for the RKEM shape functions, as presented in [8], for the integration of Reproducing Kernel Method (RKM) shape functions. Still it is an open question that should be addressed in future research.

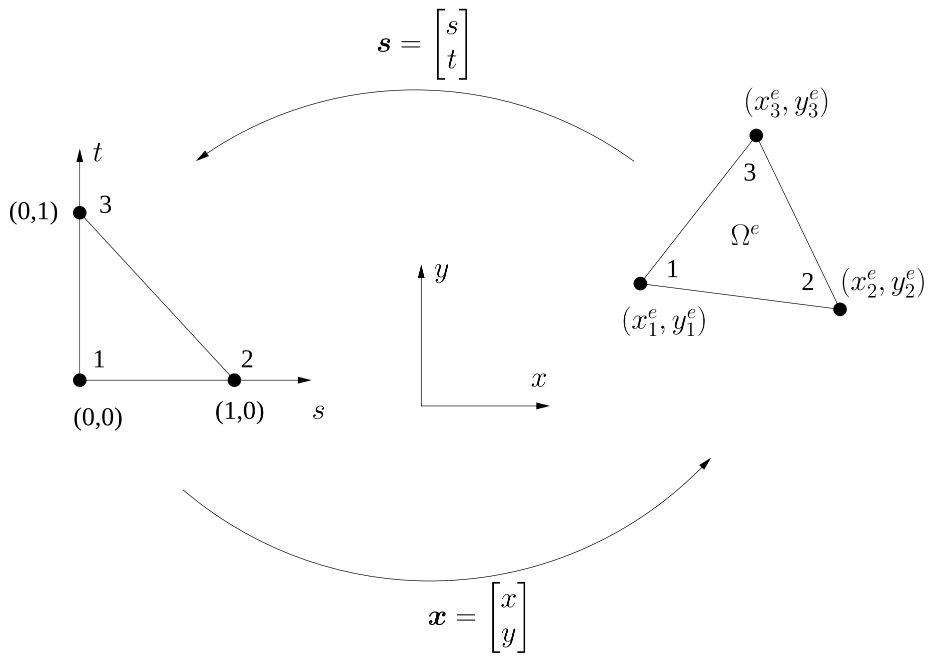
2.3 Review of Construction of Reproducing Kernel Element Global Partition Polynomials

This section deals with the construction of the new proposed symmetric global partition polynomials. The basic idea will be explained for a triangular element; due to its simplicity and elegance. Similiar procedures are used for the construction of tetrahedral elements in a three dimensional space. The construction of general quadrilaterals and hexahedral elements requires additional care, as will be discussed in § 3.5. The main reason to use reproducing kernel element is its ability to reproduce polynomials of the desired order. However it retains the Kronecker-δ property and the ability to have global smooth functions, C^n; $n \geq 0$. Contrary to finite element method, the additional degrees of freedom required for higher-order interpolation come from interpolating the primary variable and its derivatives instead of adding mid-side nodes on the edges/faces of the elements. The continuity of the globally conforming RKEM shape function is given by the order of the global partition polynomials, as was explained in §2.2.2.

The global partition polynomials for an element can be formulated using a direct or parametric approach [50]. In this dissertation, we have chosen the parametric approach to construct the global partition polynomials.

2.4 Parametric Approach

In this methodology, we want to transform the geometric domain into a parent domain using a predefined mapping as shown in Fig. 2.8. It will give us the advantage of performing



$\boldsymbol{s} = \begin{bmatrix} s \\ t \end{bmatrix}$
t
(0,1)
3
1
2
(0,0)
(1,0)
s
y
x
(x_3^e, y_3^e)
3
Ω^e
1
2
(x_2^e, y_2^e)
(x_1^e, y_1^e)
$\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

Figure 2.8 Linear triangular element domain and local node ordering.

all the calculations in a predefined domain, provided the mapping between the geometric and parametric space exists. Let ζ denote a parent domain in the parametric space,

Definition 2.4.1 (Mapping) *(Hughes [26]) Let* $\mathbf{x} : \zeta \rightarrow \overline{\Omega}^e$ *be of the form*

$$\mathbf{x}(\mathbf{s}) = \sum_{a=1}^{n_{np}} N_a(\mathbf{s})\,\mathbf{x}_a^e \tag{2.36}$$

where N_a *are shape functions and* $\mathbf{x}_a^e$ *are nodal coordinates. Eq.* (2.36) *define a mapping such that* $\mathbf{x} : \zeta \rightarrow \overline{\Omega}^e \subset \Re^d$ *is said to be* ***one-to-one*** *if for each pair of points* $\mathbf{s}^{(1)}, \mathbf{s}^{(2)} \in \zeta$ *such that* $\mathbf{s}^{(1)} \neq \mathbf{s}^{(2)}$*, then* $\mathbf{x}(\mathbf{s}^{(1)}) \neq \mathbf{x}(\mathbf{s}^{(2)})$.

The triangle shown in Fig. 2.8 is different from the triangle chosen in [50] to construct the global partition polynomials. It is a difference between our concept and the one presented in the aforementioned reference. The shape functions for this triangle are the standard finite element shape functions for a three-node triangular element:

$$N_1(s,t) = 1 - s - t \tag{2.37a}$$

$$N_2(s,t) = s \tag{2.37b}$$

$$N_3(s,t) = t \tag{2.37c}$$

In §2.2.2 we assigned to each node a set of global partition polynomials that represent generalized Hermite polynomials in order to construct a RKEM representation capable of interpolating up to the m-derivative and being of class C^n. Using a compact notation, we represent the global partition polynomial for each element as

$$\begin{gathered}
\tilde{\psi}_{e,1}(\mathbf{s}) = \begin{bmatrix} \tilde{\psi}_1^{(00)} \\ \tilde{\psi}_1^{(10)} \\ \tilde{\psi}_1^{(01)} \\ \vdots \end{bmatrix} \quad \tilde{\psi}_{e,2}(\mathbf{s}) = \begin{bmatrix} \tilde{\psi}_2^{(00)} \\ \tilde{\psi}_2^{(10)} \\ \tilde{\psi}_2^{(01)} \\ \vdots \end{bmatrix} \\
\tilde{\psi}_{e,3}(\mathbf{s}) = \begin{bmatrix} \tilde{\psi}_3^{(00)} \\ \tilde{\psi}_3^{(10)} \\ \tilde{\psi}_3^{(01)} \\ \vdots \end{bmatrix} \quad \tilde{\psi}_{e,4}(\mathbf{s}) = \begin{bmatrix} \tilde{\psi}_3^{(00)} \\ \tilde{\psi}_3^{(10)} \\ \tilde{\psi}_3^{(01)} \\ \vdots \end{bmatrix}
\end{gathered} \tag{2.38}$$

where e denotes an element in the mesh and 1, 2, 3, 4 represents the nodes in a triangle, the last one being an interior node (if required), such that the primary scalar variable can be interpolated by the formula

$$\mathscr{I}\tilde{w}(s,t) = \begin{bmatrix} \tilde{\psi}_{e,1}^T & \tilde{\psi}_{e,2}^T & \tilde{\psi}_{e,3}^T & \tilde{\psi}_{e,4}^T \end{bmatrix} \tilde{\mathbf{w}}_I, \ \ \forall \mathbf{s} \in \Omega_e \tag{2.39}$$

where $\mathscr{I}$ is the interpolation operator, the superscript T denotes transpose, and $\tilde{\mathbf{w}}_I$ is a vector of all the nodal unknowns on the element:

$$\tilde{\mathbf{w}}_I := \begin{bmatrix} \tilde{\mathbf{w}}_{e,1} \\ \tilde{\mathbf{w}}_{e,2} \\ \tilde{\mathbf{w}}_{e,3} \\ \tilde{\mathbf{w}}_{e,4} \end{bmatrix} \tag{2.40}$$

$$\begin{aligned} \tilde{\mathbf{w}}_{e,1}(\mathbf{s}) &= \begin{bmatrix} \tilde{w}(s_{e,1},t_{e,1}) \\ \tilde{w}_{,s}(s_{e,1},t_{e,1}) \\ \tilde{w}_{,t}(s_{e,1},t_{e,1}) \\ \vdots \end{bmatrix} \quad \tilde{\mathbf{w}}_{e,2}(\mathbf{s}) = \begin{bmatrix} \tilde{w}(s_{e,2},t_{e,2}) \\ \tilde{w}_{,s}(s_{e,2},t_{e,2}) \\ \tilde{w}_{,t}(s_{e,2},t_{e,2}) \\ \vdots \end{bmatrix} \\ \tilde{\mathbf{w}}_{e,3}(\mathbf{s}) &= \begin{bmatrix} \tilde{w}(s_{e,3},t_{e,3}) \\ \tilde{w}_{,s}(s_{e,3},t_{e,3}) \\ \tilde{w}_{,t}(s_{e,3},t_{e,3}) \\ \vdots \end{bmatrix} \quad \tilde{\mathbf{w}}_{e,4}(\mathbf{s}) = \begin{bmatrix} \tilde{w}(s_{e,4},t_{e,4}) \\ \tilde{w}_{,s}(s_{e,4},t_{e,4}) \\ \tilde{w}_{,t}(s_{e,4},t_{e,4}) \\ \vdots \end{bmatrix} \end{aligned} \tag{2.41}$$

The tilde was added to some variables to emphasize that the interpolation is done in the parent domain. The element interpolation provided by the global partition polynomials can

Order								*#Terms*
0				1				1
1			s		t			2
2		s^2		st		t^2		3
3	s^3		s^2t		st^2		t^3	4

Figure 2.9 Pascal's triangle for polynomials in two variables.

be written as a vector equation

$$\tilde{w}(s,t) = \Phi^T(s,t)\mathbf{c} \tag{2.42}$$

where

$$\mathbf{c}^T := \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & \dots \end{bmatrix}^{1\times n_{dof}}$$

$$\Phi^T(s,t) := \begin{bmatrix} 1 & s & t & s^2 & st & t^2 & \dots \end{bmatrix}^{1\times n_{dof}} \tag{2.43}$$

and n_{dof} is the total number of degrees of freedom in an element, and they are determined by the order of the complete polynomial space that is desired. The Pascal's triangle shown in Fig. 2.9 is used to construct Φ in a two-dimensional space. One can determine the coefficients $\mathbf{c}$ in terms of the nodal unknowns $\tilde{\mathbf{w}}_I$, by solving a set of linear equations

$$\begin{aligned}
\tilde{w}(s_{e,1},t_{e,1}) =& \quad \Phi^T(s_{e,1},t_{e,1})\mathbf{c} \\
\tilde{w}_{,s}(s_{e,1},t_{e,1}) =& \quad \Phi^T_{,s}(s_{e,1},t_{e,1})\mathbf{c} \\
\tilde{w}_{,t}(s_{e,1},t_{e,1}) =& \quad \Phi^T_{,t}(s_{e,1},t_{e,1})\mathbf{c} \\
&\vdots \\
\tilde{w}(s_{e,2},t_{e,2}) =& \quad \Phi^T(s_{e,2},t_{e,2})\mathbf{c} \\
\tilde{w}_{,s}(s_{e,2},t_{e,2}) =& \quad \Phi^T_{,s}(s_{e,2},t_{e,2})\mathbf{c}
\end{aligned}$$

$$\tilde{w}_{,t}(s_{e,2},t_{e,2}) = \Phi_{,t}^T(s_{e,2},t_{e,2})\mathbf{c}$$

$$\vdots$$

Denote the resulting coefficient matrix **A**, then

$$\mathbf{c} = \mathbf{A}^{-1}\tilde{\mathbf{w}}_I \tag{2.44}$$

and

$$\tilde{w}(s,t) = \mathbf{\Phi}^T\mathbf{A}^{-1}\tilde{\mathbf{w}}_I \tag{2.45}$$

where

$$\begin{aligned}\mathbf{A}^T = [A_{ij}^T]^{n_{dof}\times n_{dof}} &= [\mathbf{\Phi}(s_{e,1},t_{e,1}),\mathbf{\Phi}_{,x}(s_{e,1},t_{e,1}),\mathbf{\Phi}_{,y}(s_{e,1},t_{e,1}),\cdots,\\ &\quad \mathbf{\Phi}(s_{e,2},t_{e,2}),\mathbf{\Phi}_{,x}(s_{e,2},t_{e,2}),\mathbf{\Phi}_{,y}(s_{e,2},t_{e,2}),\cdots,\\ &\quad \mathbf{\Phi}(s_{e,3},t_{e,3}),\mathbf{\Phi}_{,x}(s_{e,3},t_{e,3}),\mathbf{\Phi}_{,y}(s_{e,3},t_{e,3})\cdots]\\ &\quad \mathbf{\Phi}(s_{e,4},t_{e,4}),\mathbf{\Phi}_{,x}(s_{e,4},t_{e,4}),\mathbf{\Phi}_{,y}(s_{e,4},t_{e,4})\cdots]\ .\end{aligned} \tag{2.46}$$

The desired global partition polynomials in the parametric space can now be determined by comparing Eq. (2.39) and Eq. (2.45). Therefore,

$$\begin{bmatrix} \tilde{\psi}_{e,1}^T & \tilde{\psi}_{e,2}^T & \tilde{\psi}_{e,3}^T & \tilde{\psi}_{e,4}^T \end{bmatrix} = \mathbf{\Phi}^T\mathbf{A}^{-1} \tag{2.47}$$

We can explicitly generate the global partition polynomials in the parametric space given the inversion of the matrix **A**. The inversion of the matrix **A** can be done in exact arithmetic using a computer algebra system. In fact, we have used the open-source program *Maxima* [45] for the inversion of all the **A** matrices used in this dissertation. It is worth mentioning that the inversion is done only once. Examples of global partition polynomials generated by this methodology will be given in Ch. 4.

2.4.1 From the Geometric to Parametric Space

The use of a parametric space to construct and compute the global partition polynomials necessitate the introduction of a transformation, such a transformation is given by Eq. (2.36). Because the global partition polynomials are generalized Hermite polynomials, we need to transform the nodal derivative values. As an example we will show how to transform nodal values associated with the first derivative for the node I,

$$\left\{\begin{matrix} w_{,x} \\ \\ w_{,y} \end{matrix}\right\}_I = \begin{bmatrix} s_{,x} & t_{,x} \\ \\ s_{,y} & t_{,y} \end{bmatrix} \left\{\begin{matrix} \tilde{w}_{,s} \\ \\ \tilde{w}_{,t} \end{matrix}\right\}_I \tag{2.48}$$

Equation (2.36) enables us to calculate the matrix,

$$\mathbf{x}_{,\mathbf{s}} = \begin{bmatrix} x_{,s} & x_{,t} \\ \\ y_{,s} & y_{,t} \end{bmatrix} \tag{2.49}$$

whose components are,

$$x_{,s} = \sum_{a=1}^{np} N_{a,s}(s,t)\, x_a^e \qquad\qquad x_{,t} = \sum_{a=1}^{np} N_{a,t}(s,t)\, x_a^e \tag{2.50}$$

$$y_{,s} = \sum_{a=1}^{np} N_{a,s}(s,t)\, y_a^e \qquad\qquad y_{,t} = \sum_{a=1}^{np} N_{a,t}(s,t)\, y_a^e \tag{2.51}$$

Using Eq. (2.36) with Eq. (2.37) and the information given in Fig. 2.8, we write it out explicitly as,

$$x_{,s} = x_2 - x_1 \qquad\qquad x_{,t} = x_3 - x_1 \tag{2.52a}$$

$$y_{,s} = y_2 - y_1 \qquad\qquad y_{,t} = y_3 - y_1 \tag{2.52b}$$

Performing the inversion of matrix (2.49), we have the desired transformation,

$$\mathbf{s}_{,\mathbf{x}} = \begin{bmatrix} s_{,x} & s_{,y} \\ t_{,x} & t_{,y} \end{bmatrix} = \left(\mathbf{x}_{,\mathbf{s}}\right)^{-1} = \frac{1}{x_{,s}y_{,t} - x_{,t}y_{,s}} \begin{bmatrix} y_{,t} & -x_{,t} \\ -y_{,s} & x_{,s} \end{bmatrix} \tag{2.53}$$

2.5 Interlude: Element Nomenclature

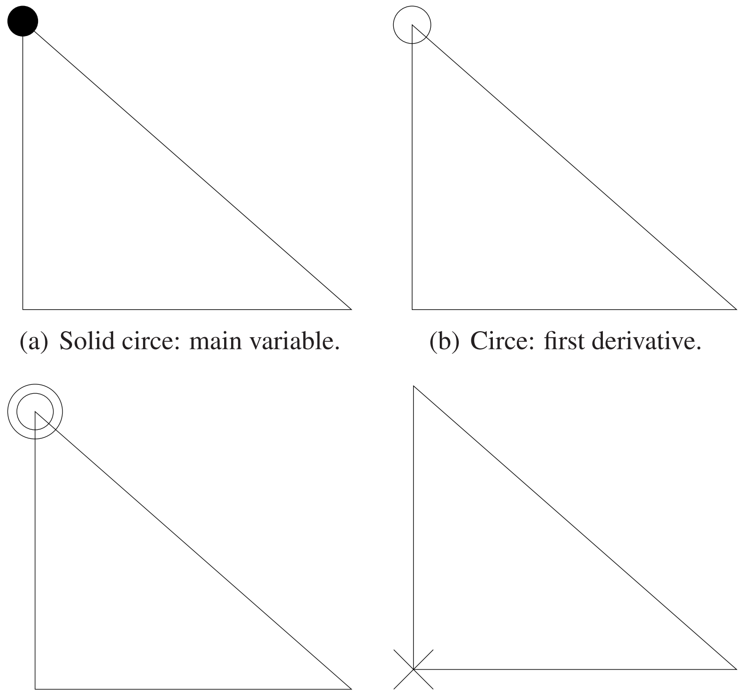
It may be useful to elaborate on the nomenclature used for specifying Reproducing Kernel Elements. We use the notation introduced in [51] where the general structure of a Reproducing Kernel Element element name takes the form:

SmPnIo

where

- The letter S denotes the shape of the element.
- The letter m is replaced by a number that is the total number of degrees of freedom for the element.
- The letter P denotes polynomial.
- The letter n is replaced by an integer that is the highest degree of globally reproduced complete polynomial.
- The letter I denotes interpolation.
- The letter o is replaced by a number, perhaps rational, that is the number of derivatives interpolated at the vertex degrees of freedom.

This nomenclature was developed for reproducing kernel elements that only had vertex nodes [33], but it contained redundant information. With a slight re-interpretation of the components of the name, we believe it is sufficient to describe the new elements derived in this dissertation. The previous redundancy was in the fact that a given shape already determines the number of vertex nodes, and combined with the order of interpolated degree-of-freedom's (primary variable, first derivatives, etc.), the total number of degrees of freedom was redundant. Figure 2.10 shows a pictorial representation of the DOF's (degree of freedom) on nodes. A solid circle means that a function value is interpolated at that point, a circle about a point means that both first derivatives are to be interpolated at that point, a double circle about a point means that both second derivatives are to be interpolated at that point and a cross means that cross derivative is to be interpolated at that point. An example



(a) Solid circe: main variable.

(b) Circe: first derivative.

(c) Double circe: Second derivative.

(d) Cross : partial derivative.

Figure 2.10 Pictorial nomenclature for degrees of freedom on nodes.

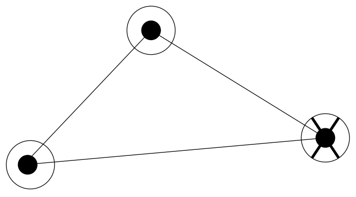
may be helpful. In previous work, the T9P2I1 triangle interpolated the primary variable and the first partial derivatives (3 degree-of-freedom/node). Since a triangle always has three vertices, the total number of degrees available for that element are (3 nodes) x (3 degree-of-freedom/node) = 9 degree-of-freedom. Hence the '9' in the element name is redundant. In §4.1.2, we will develop the T10P3I1 element. In this case, the vertex degree-of-freedom are the same, but the total number of degree-of-freedom is 10, indicating that there must be an interior node with one degree of freedom. One cannot tell from the name precisely which variables, primary or its derivatives, are interpolated at the interior node. We elected not to enhance the element name to reflect this additional information, as we feared the nomenclature would become overly cumbersome.

Chapter 3

Minimal Affine Invariant Global Partition Polynomials

3.1 Difficulties with Previous Formulation

The construction of global partition polynomials in previous formulations of RKEM have presented some problems. First, in the desire to restrict the degrees of freedom to the vertex nodes, the number of available DOF's may not exactly match the number required to represent a complete polynomial of a given order. In [42], this was fixed by using different DOF's at the vertices to ensure a match with those required. However, this complicates the problem of generating meshes, since adjacent elements sharing that vertex are now constrained to having the same DOF's. In Figure 3.1 is shown a pictorial of a mesh with different DOF's specified on nodes. In that case, the specification of the global to local numbering indexing require extra effort during the meshing process, because it must verify that a vertex that belong to more than one element has the correct DOF's. On the other hand, it is very desirable, from the meshing point of view, to restrict all vertex nodes to have the same DOF's, consequently the global to local numbering indexing do not require any special algorithm. In [50], [51] and [49], the DOF's mismatch problem was solved by selecting interpolations that provided equal or greater total DOF's required for the desired degree polynomial through additional derivative degrees of freedom. Use of derivative


(a) Element with different DOF's on nodes.

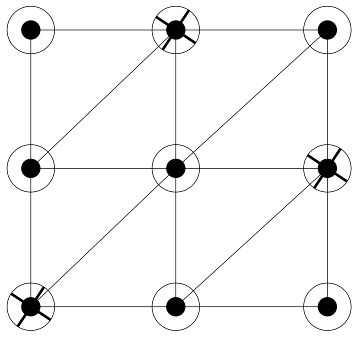

(b) Mesh

Figure 3.1 Mesh with different DOF's on nodes.

degrees of freedom leads to an added restriction for the successful use of a parametric formulation using a parent element. In order for the mapping between geometric and parent elements to be complete, all derivatives of a given order must be present. For example, in two dimensions, if one would like to interpolate the second derivatives $u_{,ss}$ and $u_{,tt}$, one must also interpolate $u_{,st}$. Generally, then, more DOF's are available than required. In the case of the T9P2I1 element, 9 vertex DOF's are available, consisting of the primary variable and its first partial derivatives (see Fig. 3.2), but only a quadratic polynomial is exactly reproduced. A quadratic only requires 6 DOF's so to account for the extra degree-of-freedom, the next higher polynomial, a cubic, is used. The cubic requires a total of 10 degree-of-freedom, so the previous formulation combined two higher order terms, s^2t+st^2 to match the 9 available DOF's. According to Eq. (2.42), for the T9P2I1 we have,

$$\mathbf{c}^{\mathbf{T}} := \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 \end{bmatrix} \tag{3.1a}$$

$$\Phi^T(s,t) := \begin{bmatrix} 1 & s & t & s^2 & st & t^2 & s^3 & s^2t+st^2 & t^3 \end{bmatrix} \tag{3.1b}$$

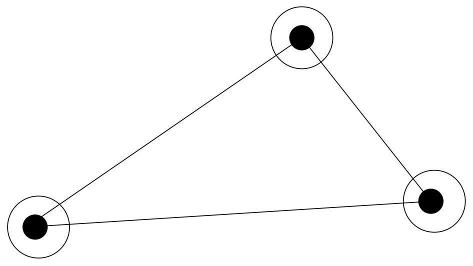


Figure 3.2 T9P2I1 element.

While this process achieved the desired reduction, a full quadratic field is interpolated and only a partial cubic field.

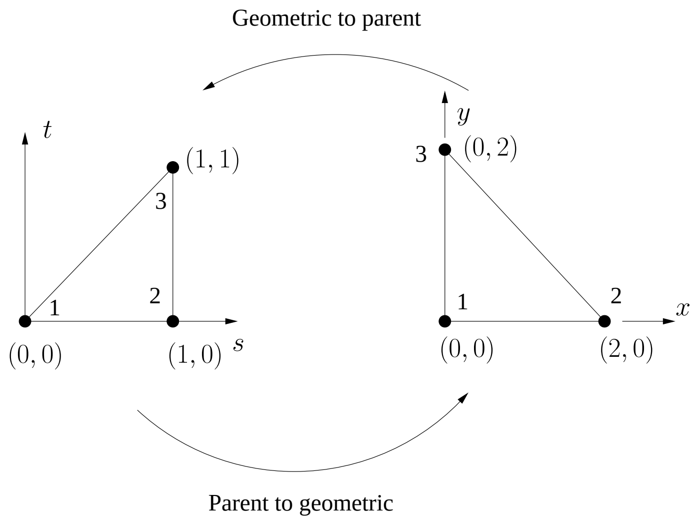
Definition 3.1.1 (Linearly Dependent Functions) *(Sansone [47]) The n functions $f_1(\mathbf{x})$, $f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})$ are linearly dependent if, for some $c_1, c_2, \ldots, c_n \in \Re$ not all zero,*

$$\sum_{i=1}^{n} c_i f_i(\mathbf{x}) = 0 \tag{3.2}$$

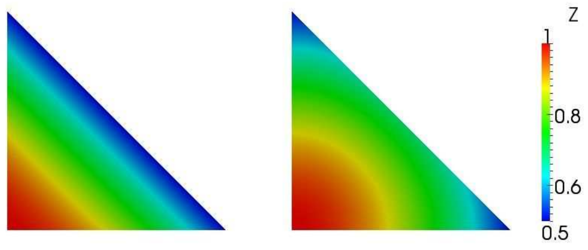
for all $\mathbf{x} \in \Re^d$ in some interval. If the functions are not linearly dependent, they are said to be linearly independent.

The set of polynomials defined in Eq. (3.1b) are linearly dependent and therefore they do not form an *affine-transformation*. An account of the theory of affine-transformations is given in [46] and presented here using same notation as in the reference aforementioned.

Definition 3.1.2 (Affine transformation) *(Oden [46]) Let $\{\Gamma_\Delta(\mathbf{x})\}$ be another set of G linearly independent functions. If this set also belongs to a subspace Φ of finite dimension G, it is always possible to find a matrix A^Γ_Δ which describes an affine trasformation of the set $\{\Gamma_\Delta\}$ into $\{\Phi_\Delta\}$.*



(a) Geometric to Parent.



(b) Global partition polynomial for geometric node 1 and two different local to global node numbering.

Figure 3.3 Asymmetry in global partition polynomials in the T9P2I1 element.

Using Eq. (3.1b) and Eq. (3.2) we arrive at,

$$c_1 + c_2 t + c_3 s + c_4 s^2 + c_5 st + c_6 t^2 + c_7 s^3 + c_8 \left(s^2 t + st^2\right) + c_9 t^3 = 0 \tag{3.3}$$

If only $c_8 \neq 0$ in Eq. (3.3), then $s = -t$ for $s,t \neq 0$. In other words, there exist a $c_8 \neq 0$ that makes Eq. (3.2) equal to zero for all the s,t lying on the line $s+t=0$, but excluding the origin. The latter means that the space of polynomials defined in Eq. (3.1b) is linearly dependent, hence not affine-invariant. The reduction process (i.e., blending two higher order terms, $s^2t + st^2$, to match required DOF's) destroys the affine invariance of the restricted polynomial space. This leads to asymmetric interpolations even for simple changes in the local to global node numbering. Figure 3.3 demonstrates the asymmetry in the global partition polynomials for the T9P2I1 element. The asymmetry is due to re-ordering of the local to global node numbering only. Figure 3.3(b) plots the global partition polynomial associated with the global node number 1. Note how the resulting global partition polynomial is different, even for a simple change of the local to global nodal numbering. Further, the restricted polynomials could lead to a singularity in the coefficient equation, Eq. (2.46). To side step the singularity issue, special choices of parent elements were made to guarantee invertibility, [50, 51]. The asymmetry due to re-ordering the local to global node numbering can be seen in the following example for the reproducing kernel element interpolation. We used a simple two-element mesh shown in Fig. 3.4(a) and used the T9P2I1 element to interpolate the function

$$f(x,y) = \sin(x \cdot y) \quad \text{on} \quad (x,y) \in [0,2] \times [0,2]. \tag{3.4}$$

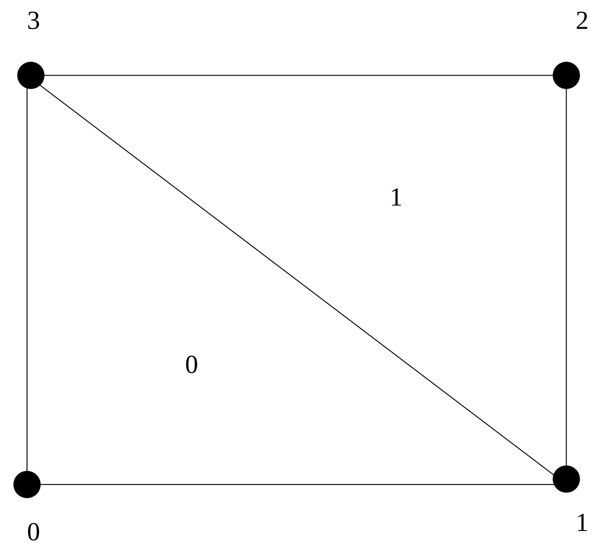
Table 3.1 Local to global node numbering for the mesh in Fig. 3.4(a) used to demonstrate asymmetric interpolation of the T9P2I1 element.

Element	Local Node 1	Local node 2	Local node 3
0	3	0	1
1	1	2	3
0	1	3	0
1	3	1	2

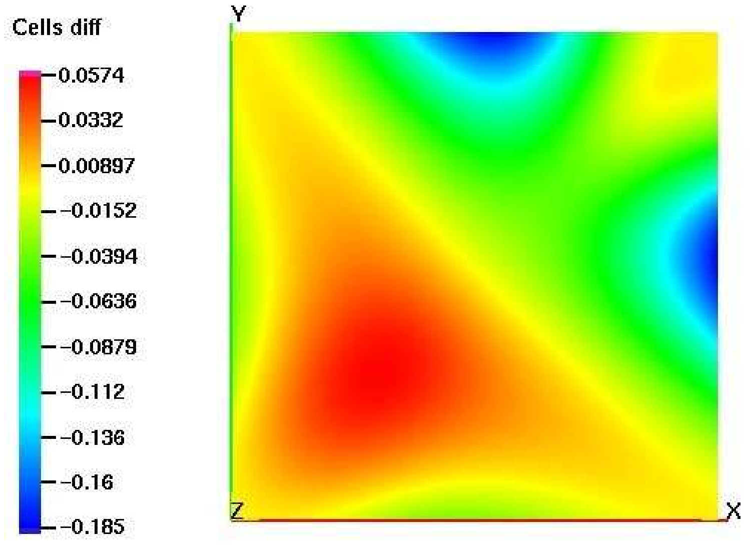
We then changed the local to global node numbering for the elements, as shown in Table 3.1, and compared the difference between the two interpolations, as shown in Fig. 3.4(b). In Table 3.1, the top two rows are the connectivity for the first interpolation, the second two rows for the second interpolation. The difference between the two interpolations, due solely to changes in the connectivity, ranges from -0.185 to 0.057.

3.2 A New Formulation of Symmetric Global Partition Polynomials

We have formulated truly minimal affine-invariant global partition polynomials for parametric reproducing kernel elements. Further, our formulation retains the same simplicity of meshing as finite element triangular and tetrahedral meshes [25, 48] but with higher order continuity. The latter being an important practical issue for the meshing of complex domains and adaptive meshing analysis [37]. The basic concept is to automatically generate a single interior node at the centroid of the element (see Fig. 3.5). This node then carries enough degrees of freedom to complete the necessary degrees of freedom for an exact polynomial of given degree. To avoid meshing complications, we require that, for any element, the degree-of-freedom interpolated at each vertex is exactly the same. In finite elements, an interior node would generally lead to either compatibility issues along



(a) mesh



(b) Difference between interpolations

Figure 3.4 Asymmetric interpolation arising from changes in node numbering.

the boundary of the element. Since reproducing kernel elements uses the meshfree kernel to enforce compatibility, we are free to add extra nodes without compatibility concerns. Note that we could add additional interior nodes, but we feel that is an extra level of complication, and we have not found it necessary to do so for triangular and tetrahedral elements. Also, mid-side nodes could be added without compatibility concerns, but that could still lead to the same meshing difficulties encountered in FEM for elements with mid-side nodes. Since our goal is the simplest, complete, affine-invariant element formulation, we chose to use a single additional node at the centroid of the element. As an interior node, our implementation automatically creates the node during the initialization process. This allows us to use FEM triangular and tetrahedral mesh generators - all the extra nodes are created completely within the reproducing kernel element code. For the generation of the interior node in the geometrical space, we only need to know the coordinate location of the vertex nodes of the element, then the interior node is created easily. For triangular elements, we use Eq. (2.37) as follows

$$\mathbf{x}_4 = N_1\left(\frac{1}{3},\frac{1}{3}\right)\mathbf{x}_1 + N_2\left(\frac{1}{3},\frac{1}{3}\right)\mathbf{x}_2 + N_3\left(\frac{1}{3},\frac{1}{3}\right)\mathbf{x}_3 \tag{3.5}$$

With the goal of truly practical, robust implementations, we use the following guide lines in our constructions:

a) All nodes on the boundary of an element lie at the vertices.

b) All vertices have exactly the same degrees of freedom.

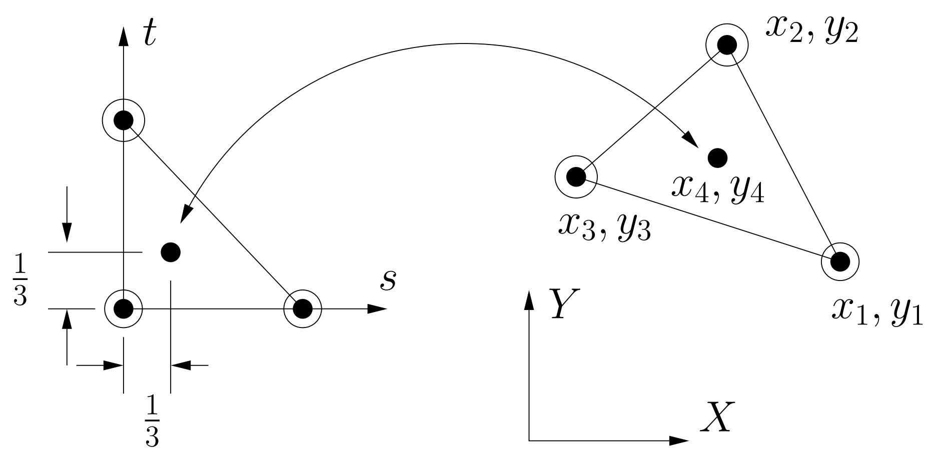


Figure 3.5 Location of interior node.

c) If any DOF corresponds to a derivative, then all terms of that order derivative must also be degrees of freedom. In order for the mapping between geometric and parent elements to be complete, all derivatives of a given order must be present.

3.2.1 Tutorial: Construction of Global Partition Polynomials for the T6P2I0 Element

In this section we present a tutorial on how to construct global partition polynomials for the T6P2I0 element. The element is capable of reproducing quadratic polynomials, but it is not capable of interpolating derivatives other than for quadratic polynomials. The latter, because we do not interpolate the first partial derivatives at the vertex nodes. According to Eq. (2.42), for the T6P2I0 (see Fig 3.6.) we have,

$$\mathbf{c}^{\mathbf{T}} := \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{bmatrix} \tag{3.6a}$$

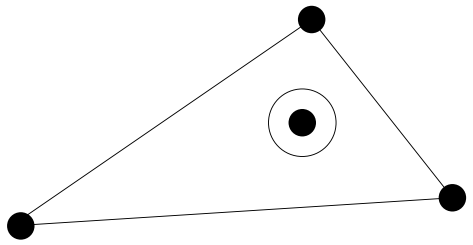


Figure 3.6 T6P2I0 element.

$$\Phi^T(s,t) := \begin{bmatrix} 1 & s & t & s^2 & st & t^2 \end{bmatrix} \tag{3.6b}$$

The requirement of six DOF's is because to represent a complete quadratic polynomials, six monomial terms should be presents (see Pascal's triangle in Fig. 2.9). Before using Eq. (2.46) to evaluate $\mathbf{A}$, we compute the requires derivatives of $\Phi^T(s,t)$

$$\Phi^T_{,s}(s,t) = \begin{bmatrix} 0 & 1 & 0 & 2s & t & 0 \end{bmatrix} \tag{3.7a}$$

$$\Phi^T_{,t}(s,t) = \begin{bmatrix} 0 & 0 & 1 & 0 & s & 2t \end{bmatrix} \tag{3.7b}$$

Using the coordinates of the nodes in the parent element, $(0,0)$, $(1,0)$, $(0,1)$, $(1/3,1/3)$, Eq. (3.6b) and Eq. (3.7), we arrive at Eq. (3.8). The global partition polynomials are obtained using Eq. (2.47) and Eq. (3.8). The resulting global partition polynomials are symmetric as will be shown in next chapter. To conclude this tutorial, it is important to mention that we are not interpolating derivatives in the vertex nodes, then the global partition polynomials for the T6P2I0 element do not have the Kronecker-δ property in the derivatives (Eq. (2.11)) evaluated at vertex nodes, furthermore that is independent of the size of the radius of support for the interior node.

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & \frac{1}{3} & \frac{1}{3} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ 0 & 1 & 0 & \frac{2}{3} & \frac{1}{3} & 0 \\ 0 & 0 & 1 & 0 & \frac{1}{3} & \frac{2}{3} \end{bmatrix}, \quad \mathbf{A}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -3 & -1 & 1 & 3 & 1 & -2 \\ -3 & 1 & -1 & 3 & -2 & 1 \\ 2 & 2 & -1 & -3 & -1 & 2 \\ 5 & -1 & -1 & -3 & 2 & 2 \\ 2 & -1 & 2 & -3 & 2 & -1 \end{bmatrix} \tag{3.8}$$

3.3 Summary of Triangular and Tetrahedral Elements

We summarize the successful triangular and tetrahedral Reproducing Kernel Elements, their DOF, and the locations of the DOF in Table 3.2. In Table 3.2 the "*" in the Tet10P2I0 is used to denote that this element could violate the guide lines of construction of symmetric global partition polynomials, as we will discuss in next chapter. Also, it is important to say that the Kronecker-δ property may or may not be imposed on the interior node. In this dissertation, we have chosen to relax the Kronecker-δ property for those nodes, in a attempt to smooth the RKEM shape functions associated with interior nodes.

3.4 Symmetry of the Interpolation

The T6P2I0 and T10P3I1 (§ 4.1.2) elements do provide symmetric interpolations, as shown in Fig. 3.7 and Fig. 3.8, unlike the T9P2I1 element. Figure 3.7 plots the global partition polynomial associated with the global node number 2 for the T6P2I0 and T10P3I1.

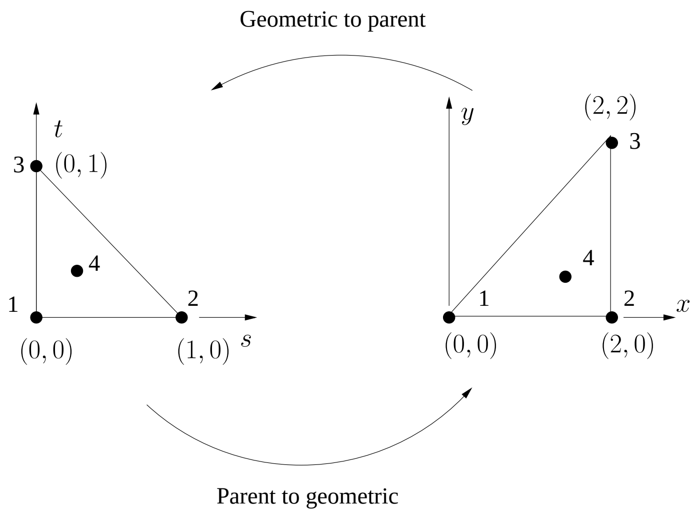
Table 3.2 Minimal triangular/tetrahedral elements and their vertex and interior degrees of freedom.

Element	# nodes (exterior / interior)	Vertex DOF	Interior node DOF
T3P1I0	3 / 0	u	-
T6P2I0	3 / 1	u	u; $u_{,x}$; $u_{,y}$
T10P3I1	3 / 1	u; $u_{,x}$; $u_{,y}$	u
Tet4P1I0	4 / 0	u	-
Tet10P2I0	4 / 1	u	$u_{,xx}$; $u_{,yy}$; $u_{,zz}$; $u_{,xy}$; $u_{,yz}$; $u_{,zx}$
Tet10P2I0*	4 / 1	u	$u_{,x}$; $u_{,y}$; $u_{,z}$; $u_{,xy}$; $u_{,yz}$; $u_{,zx}$
Tet20P3I1	4 / 1	u; $u_{,x}$; $u_{,y}$; $u_{,z}$	u; $u_{,x}$; $u_{,y}$; $u_{,z}$

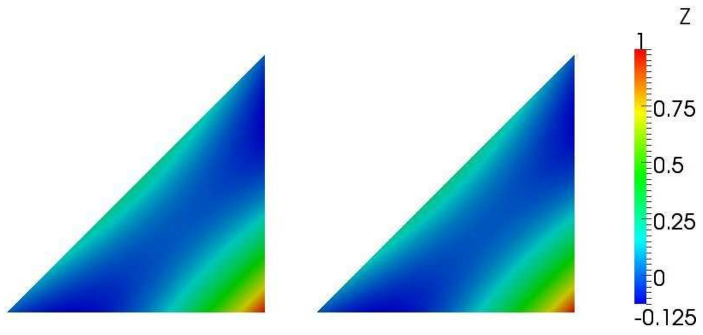
Within numerical precision, the global partition polynomial with different local global numbering is the same. Considering the sample RKEM interpolation in Fig. 3.4, the difference plots for the T6P2I0 and the T10P3I1 are shown in Fig 3.8. Within numerical precision, the interpolation with different local global numbering is the same.

3.5 Quadrilateral and Hexahedral Elements

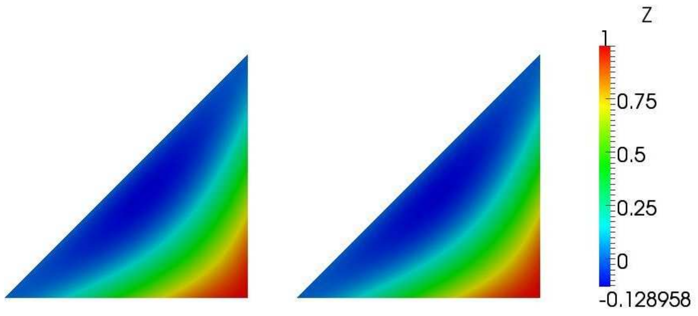
We finish this chapter with a comment on constructing quadrilateral and hexahedral elements using this methodology. The main impetus for the use of quadrilateral and hexahedral elements in finite elements was to increase the interpolatory power of the shape functions. The strictly linear triangle element was seen to demonstrate locking in simple, but important problems, such as beam bending. The additional degree-of-freedoms afforded by the extra node reduced this phenomenon. While this solved one problem, it introduced another, namely in meshing. Every point set in two-dimensions may be triangularized, but they may not always be quadrialteralized. Thus, quadrilateral meshing is more difficult than triangular meshing. A similar situation exists in three dimensions, though



(a) Geometric to parent.

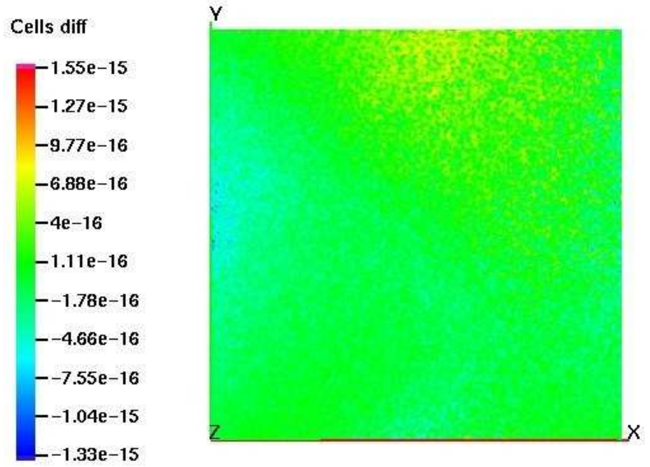


(b) Global partition polynomial for geometric node 2 and two different local to global node numbering for the T6P2I0 element.

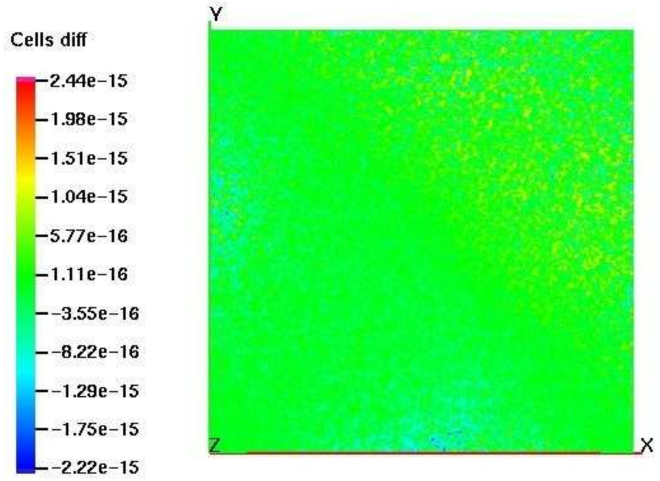


(c) Global partition polynomial for geometric node 2 and two different local to global node numbering for the T10P2I1 element.

Figure 3.7 Symmetric global partition polynomials in the T6P2I0 and T10P3I1 elements.



(a) T6P2I0



(b) T10P3I1

Figure 3.8 Difference plots for symmetric interpolations.

not every point set admits even a tetrahedralization, construction of tetrahedral meshes is still simpler than hexahedral meshes. Higher-order triangular finite elements exist by introducing nodes along their edges. Compatibility requires these nodes to also exist in the shared neighbor, and thus complicates the meshing problem. With the advent of higher-order triangular and tetrahedral elements that only share vertex nodes with their neighbors, we do not believe there is much motivation to use reproducing kernel element quadrilateral or hexahedral elements, and as such, have not devoted much effort to their development. However, if one already possesses a quadrilateral or hexahedral mesh, it may be desirable to construct RKEM shape functions directly on that mesh, and RKEM quadrilateral and hexahedral elements may have a place.

We attempted to construct truly minimal parametric quadrilateral and hexahedral elements as before, however, we were unable to construct these elements. The approach we adopted goes by the name *Birkhoff* or *Hermite-Birkhoff* interpolation in the mathematics literature, [4, 41]. Matrices of the form Eq. (2.46) are called Vandermonde matrices. Under certain conditions, the Vandermonde matrix may be singular, and in the case of the elements that we attempted in Table 3.4, it is singular. Since we are not particularly interested in quadrilateral and hexahedral elements, we did not pursue the issue further. Nevertheless, we present the details of our attempt, and a guide to resolving the issues that arose. In Table 3.3, we list the number of degrees of freedom available at the vertexes depending, consistent with the guidelines of construction of global partition polynomials. Recalling that quadratic polynomials in two- and three-dimensions require 6 and 10 degree-of-freedom, respectively, and cubic polynomials require 10 and 20, respectively, it is easy to see from

Table 3.3 Vertex degree-of-freedom available for quadrilateral/hexahedral elements for 0^{th}- and 1^{st}-order Hermite interpolation.

Element	# nodes	0^{th}-order	1^{st}-order
Quadrilateral	4	4	12
Hexahedral	8	8	32

Table 3.3 that interpolating any derivative degrees of freedom at the vertexes provides too many degree-of-freedom. Thus, quadrilateral and hexahedral elements consistent with our formulation only interpolate the primary variable at the vertexes. A quadratic quadrilateral can easily be formulated meeting our rules of construction of global partition polynomials by adding a single node at the centroid of the quadrilateral and interpolating the two first partial derivatives. This element is the Q6P2I0 element. A cubic quadrilateral requires 6 degree-of-freedom at the center node, which can be provided by interpolating the primary variable, and the first and second partial derivatives. A quadratic hexahedral element requires two additional interior degrees-of-freedom, but no combination of primary variable or derivative degrees of freedom consistent with it exist. Cubic hexahedral elements require an additional 12 degrees-of-freedom, and again, no combination of degrees-of-freedom's consistent with our rules exist. We are then faced with using more than one internal node. From an aesthetic point of view, we believe it is desirable to have each internal node carry the same degrees of freedom. Quadratic and cubic hexahedral elements can be constructed using two internal nodes; the interior nodes interpolate the primary variable in the quadratic case, and interpolate the second partial derivatives in the cubic case. The placement of the two interior nodes is arbitrary. These elements are summarized in Table 3.4.

Table 3.4 Failed quadrilateral/hexahedral elements and their vertex and interior degrees of freedom.

Element	# nodes (exterior / interior)	Vertex DOF	Interior node DOF
Q6P2I0	4 / 1	u	$u_{,x}$; $u_{,y}$
Q10P3I0	4 / 1	u	u; $u_{,x}$; $u_{,y}$; $u_{,xx}$; $u_{,yy}$; $u_{,xy}$
Hex10P2I0	8 / 2	u	u
Hex20P3I0	8 / 2	u	$u_{,xx}$; $u_{,yy}$; $u_{,zz}$; $u_{,xy}$; $u_{,yz}$; $u_{,zx}$

Chapter 4

Element Library

4.1 Triangular Elements

4.1.1 The T6P2I0 Triangle

A quadratic polynomial requires six degrees of freedom. We specify primary variable DOF, u, at each vertex and primary variable plus its first partial derivatives, $\{u; u_{,x}; u_{,y}\}$, at the interior node. The global partition polynomials are:

$$\tilde{\psi}_1^{(00)} = 2t^2 + 5st - 3t + 2s^2 - 3s + 1 \tag{4.1a}$$

$$\tilde{\psi}_2^{(00)} = -t^2 - st + t + 2s^2 - s \tag{4.1b}$$

$$\tilde{\psi}_3^{(00)} = 2t^2 - st - t - s^2 + s \tag{4.1c}$$

$$\tilde{\psi}_4^{(00)} = -3t^2 - 3st + 3t - 3s^2 + 3s \tag{4.1d}$$

$$\tilde{\psi}_4^{(10)} = 2t^2 + 2st - 2t - s^2 + s \tag{4.1e}$$

$$\tilde{\psi}_4^{(01)} = -t^2 + 2st + t + 2s^2 - 2s \tag{4.1f}$$

Figure 4.1 shows the corresponding global partition polynomials for a mesh composed of two triangular elements to emphasizes that the global partition polynomials expand beyond the elements where they are defined. The global partition polynomials are defined in the whole domain, and they are smooth polynomials. Figure 4.2 shows the RKEM shape functions for the mesh center node and a symmetric mesh and Figure 4.3 shows same but for an asymmetric mesh. In both cases, are appreciates the benefit of the relaxation of the Kronecker-δ property on those nodes that are not required to have it.

4.1.2 The T10P3I1 Triangle

A cubic polynomial requires ten degrees of freedom. We specify primary variable and first partial derivative degree-of-freedoms, $\{u; u_{,x}; u_{,y}\}$, at each vertex and the primary variable u, at the interior node. The global partition polynomials are:

$$\tilde{\psi}_1^{(00)} = 2t^3 + 13st^2 - 3t^2 + 13s^2t - 13st + 2s^3 - 3s^2 + 1 \tag{4.2a}$$

$$\tilde{\psi}_1^{(10)} = 2st^2 + 3s^2t - 3st + s^3 - 2s^2 + s \tag{4.2b}$$

$$\tilde{\psi}_1^{(01)} = t^3 + 3st^2 - 2t^2 + 2s^2t - 3st + t \tag{4.2c}$$

$$\tilde{\psi}_2^{(00)} = 7st^2 + 7s^2t - 7st - 2s^3 + 3s^2 \tag{4.2d}$$

$$\tilde{\psi}_2^{(10)} = -2st^2 - 2s^2t + 2st + s^3 - s^2 \tag{4.2e}$$

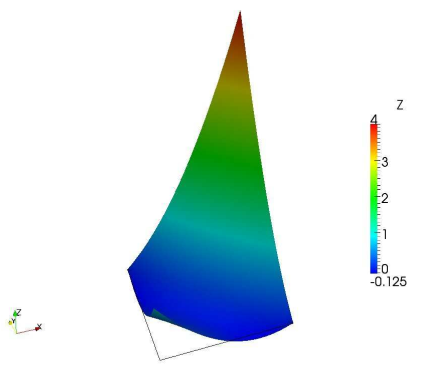
$$\tilde{\psi}_2^{(01)} = st^2 + 2s^2t - st \tag{4.2f}$$

$$\tilde{\psi}_3^{(00)} = -2t^3 + 7st^2 + 3t^2 + 7s^2t - 7st \tag{4.2g}$$

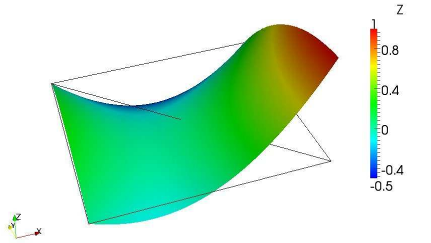
$$\tilde{\psi}_3^{(10)} = 2st^2 + s^2t - st \tag{4.2h}$$

$$\tilde{\psi}_3^{(01)} = t^3 - 2st^2 - t^2 - 2s^2t + 2st \tag{4.2i}$$

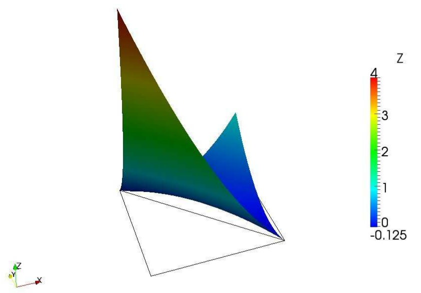
$$\tilde{\psi}_4^{(00)} = -27st^2 - 27s^2t + 27st \tag{4.2j}$$



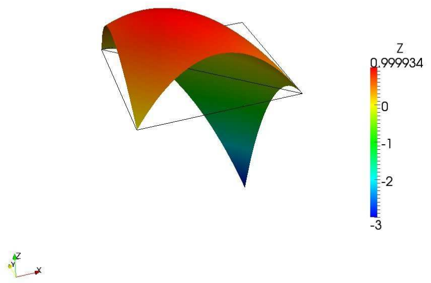
(a) $\tilde{\psi}_1^{(00)}$.



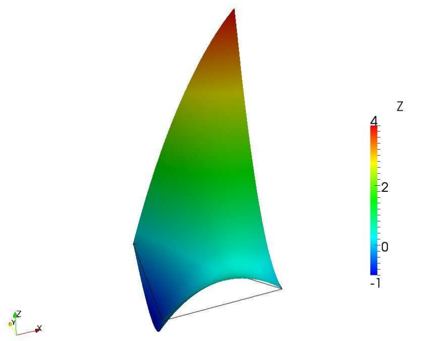
(b) $\tilde{\psi}_2^{(00)}$.



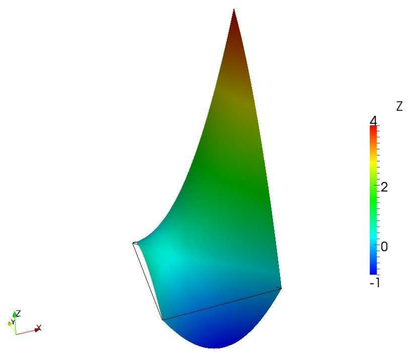
(c) $\tilde{\psi}_3^{(00)}$.



(d) $\tilde{\psi}_4^{(00)}$.

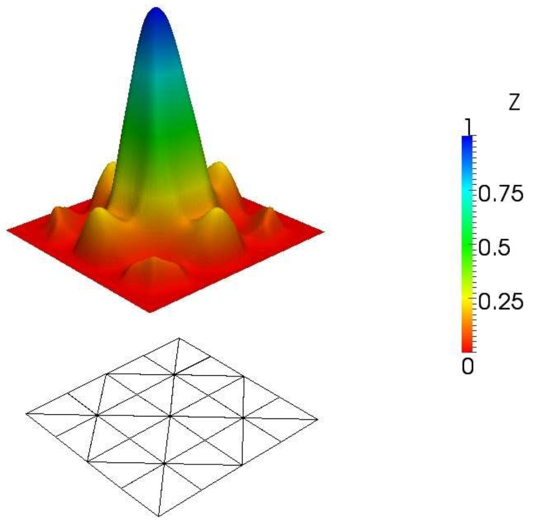


(e) $\tilde{\psi}_4^{(10)}$.

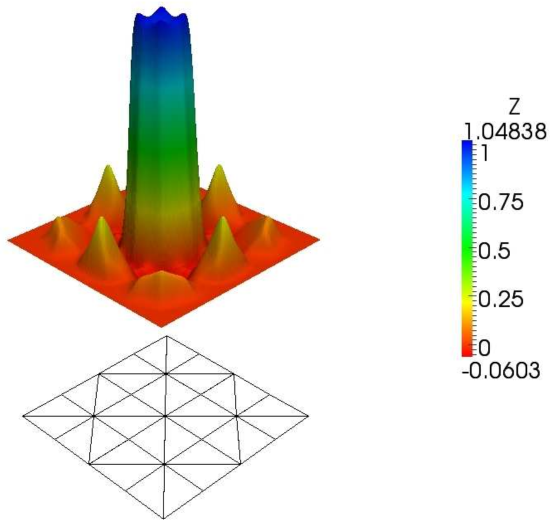


(f) $\tilde{\psi}_4^{(01)}$.

Figure 4.1 Global partition polynomial for the T6P2I0 element.

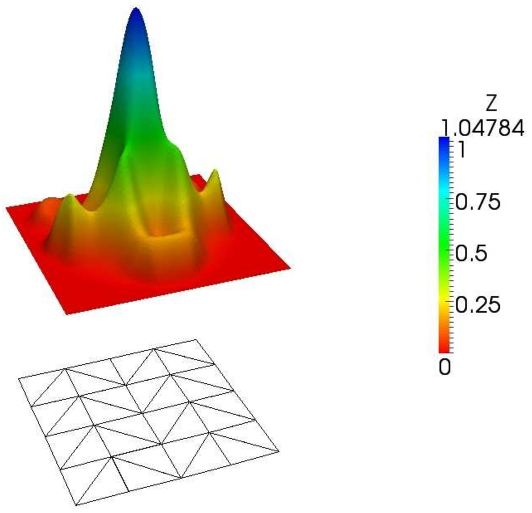


(a) Kronecker-δ property enforced only on boundary nodes.

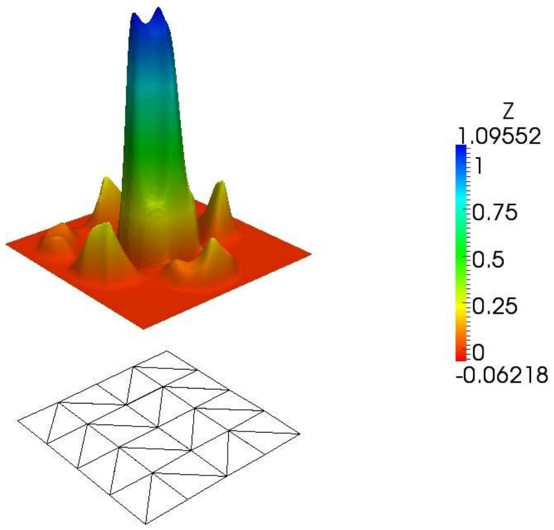


(b) Kronecker-δ property enforced on all the vertex nodes.

Figure 4.2 The global shape function of T6P2I0 element and symmetric mesh:$\Psi_I^{(00)}$.



(a) Kronecker-δ property enforced only on boundary nodes.



(b) Kronecker-δ property enforced on all the vertex nodes.

Figure 4.3 The global shape function of T6P2I0 element and asymmetric mesh: $\Psi_I^{(00)}$.

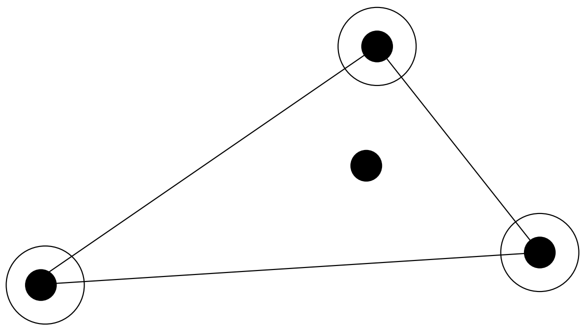


Figure 4.4 T10P3I1 element.

The element is capable of reproducing cubic polynomials and interpolate first derivatives at the corner nodes. The latter, because we specify the first partial derivatives at the vertex nodes. According to Eq. (2.42), for the T10P3I1 (see Fig. 4.4) we have,

$$\mathbf{c^T} := \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} \end{bmatrix} \tag{4.3a}$$

$$\Phi^T(s,t) := \begin{bmatrix} 1 & s & t & s^2 & st & t^2 & s^3 & s^2t & st^2 & t^3 \end{bmatrix} \tag{4.3b}$$

Figure 4.5 and Fig 4.6 shows the corresponding global partition polynomials for a mesh composed of two triangular elements to emphasizes that the global partition polynomials expand beyond the elements where they are defined. The global partition polynomials are defined in the whole domain, and they are smooth polynomials. Figure 4.7 shows the RKEM shape functions for the mesh center node and a symmetric mesh and Figure 4.10 shows same but for an asymmetric mesh. In both cases, it is appreciated the benefit of the relaxation of the Kronecker-δ property on those nodes that are not required to have it. In §5.1 is presented an application example where we have relaxed the Kronecker-

δ property for some nodes. Unlike the T6P2I0 global partition polynomials and RKEM shape functions, the T10P3I1 global partition polynomials and RKEM shape functions are are more complex. It could be related with the large values of the global partition polynomials for the T10P3I1; specially at the interior node ($\tilde{\psi}_4^{(00)}$), besides in general, cubic polynomials has a more complex behavior that quadratic polynomials.

4.2 Tetrahedral Elements

4.2.1 The Tet10P2I0 Tetrahedra

A quadratic polynomial in three variables contains ten terms. The Tet10P2I0 element interpolates the primary variable, u, at each vertex. The remaining six degrees of freedom at the interior node are chosen to be the six second partial derivatives, $\{u_{,xx}; u_{,yy}; u_{,zz}; u_{,xy}; u_{,yz}; u_{,zx}\}$. It is interesting to note the simplicity of the global partition polynomials for this case, and that the vertex global partition polynomials are the same as the C^0 finite element shape functions for the same tetrahedron. The global partition polynomials are:

$$\tilde{\psi}_1^{(000)} = 1 - w - t - s \tag{4.4a}$$

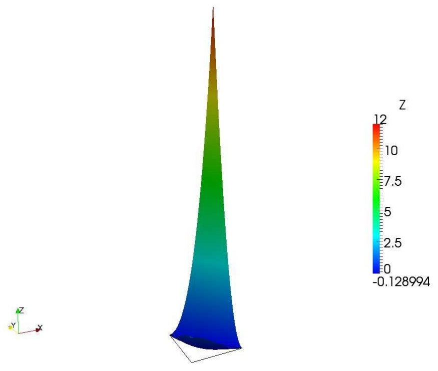
$$\tilde{\psi}_2^{(000)} = s \tag{4.4b}$$

$$\tilde{\psi}_3^{(000)} = t \tag{4.4c}$$

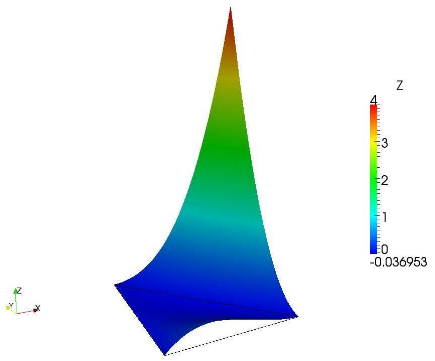
$$\tilde{\psi}_4^{(000)} = w \tag{4.4d}$$

$$\tilde{\psi}_5^{(200)} = \frac{s^2}{2} - \frac{s}{2} \tag{4.4e}$$

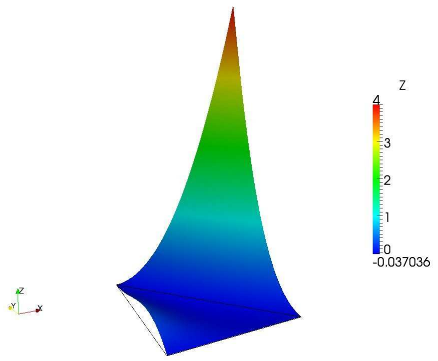
$$\tilde{\psi}_5^{(200)} = \frac{t^2}{2} - \frac{t}{2} \tag{4.4f}$$



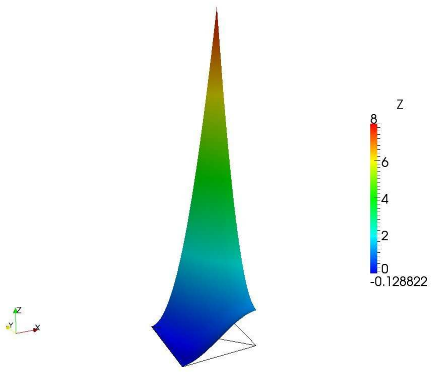
(a) $\tilde{\psi}_1^{(00)}$.



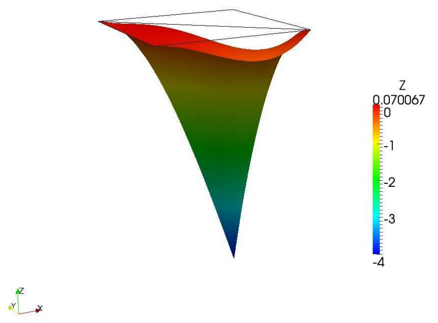
(b) $\tilde{\psi}_1^{(10)}$.



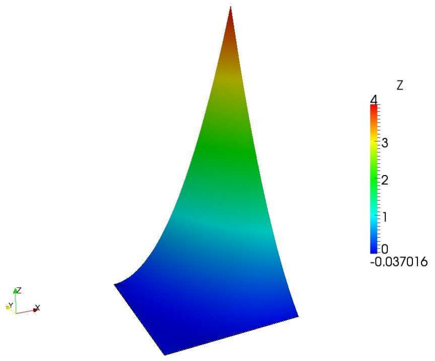
(c) $\tilde{\psi}_1^{(01)}$.



(d) $\tilde{\psi}_2^{(00)}$.

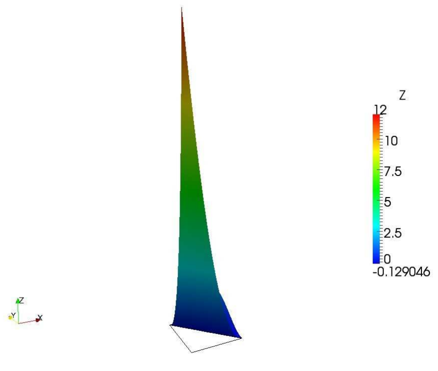


(e) $\tilde{\psi}_2^{(10)}$.

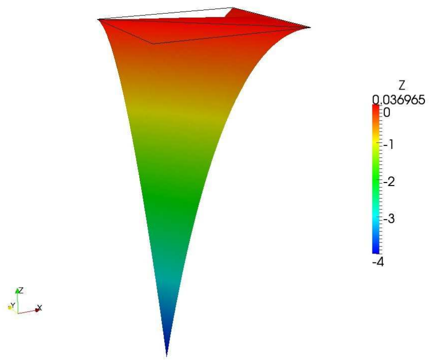


(f) $\tilde{\psi}_2^{(01)}$.

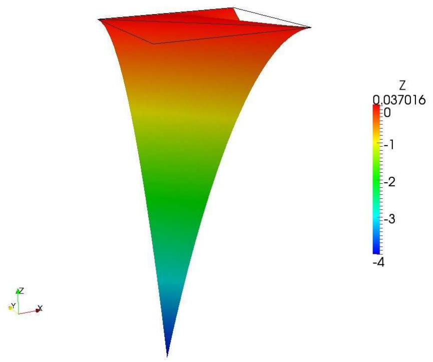
Figure 4.5 Global partition polynomial for the T10P3I1 element: node 1 to 2.



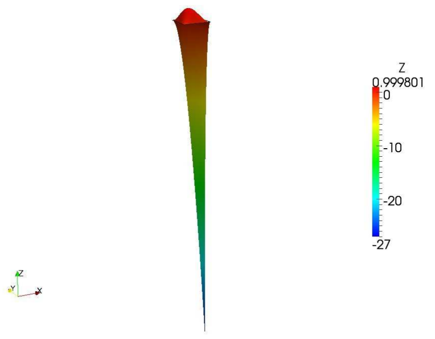
(a) $\tilde{\psi}_3^{(00)}$.



(b) $\tilde{\psi}_3^{(10)}$.

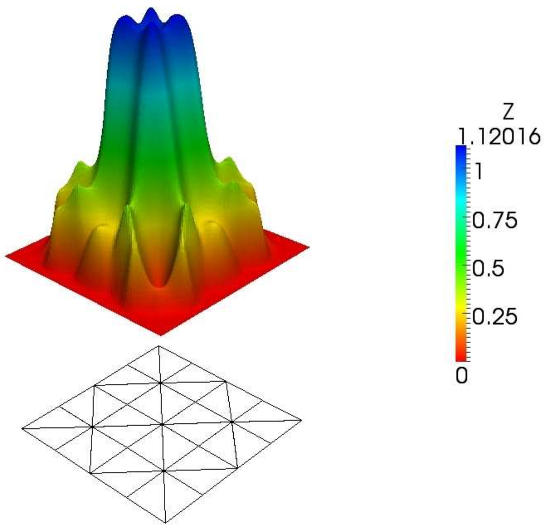


(c) $\tilde{\psi}_3^{(01)}$.

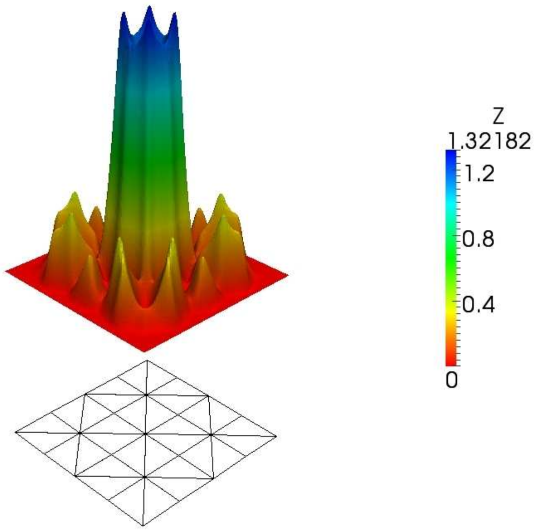


(d) $\tilde{\psi}_4^{(00)}$.

Figure 4.6 Global partition polynomial for the T10P3I1 element: node 3 to 4.

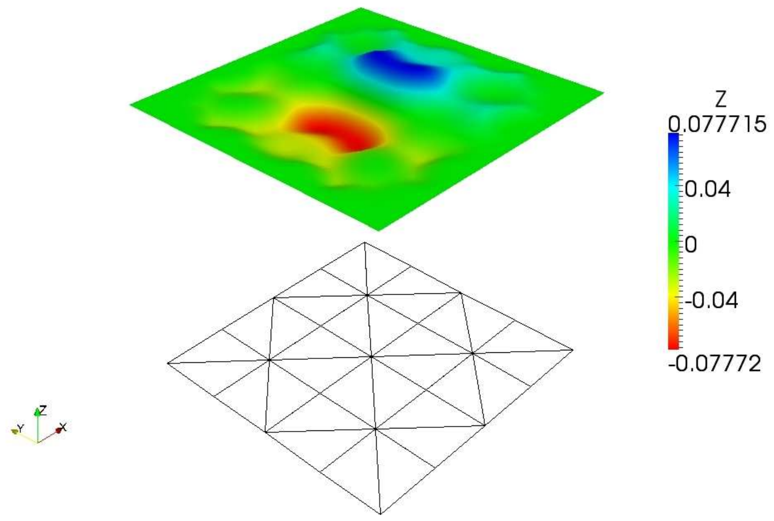


(a) Kronecker-δ property enforced only on boundary nodes.

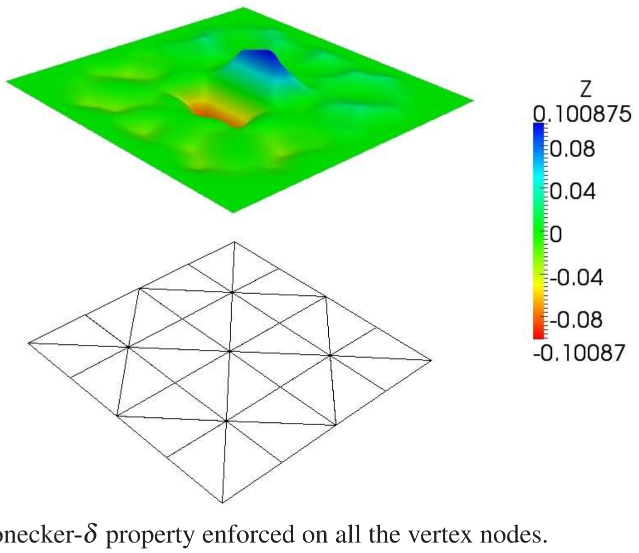


(b) Kronecker-δ property enforced on all the vertex nodes.

Figure 4.7 The global shape function of T10P3I1 element and symmetric mesh: $\Psi_I^{(00)}$.

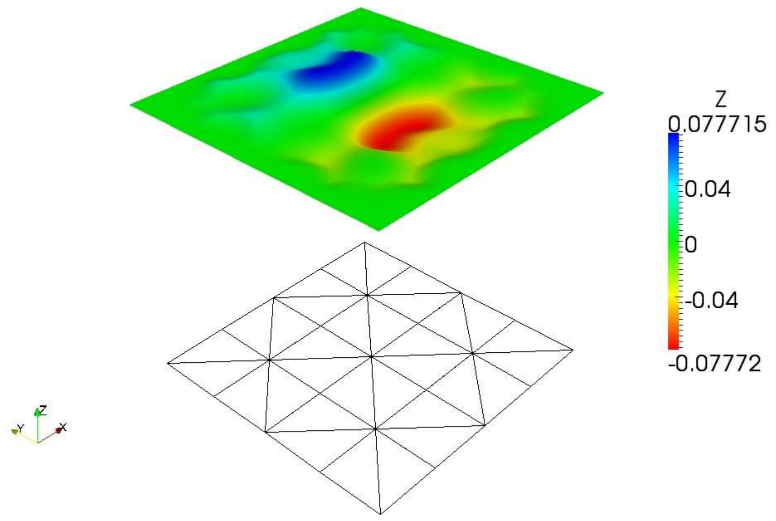


(a) Kronecker-δ property enforced only on boundary nodes.

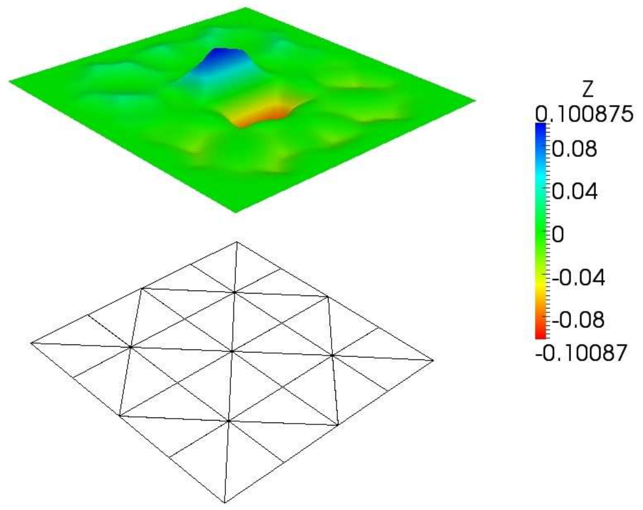


(b) Kronecker-δ property enforced on all the vertex nodes.

Figure 4.8 The global shape function of T10P3I1 element and symmetric mesh: $\Psi_I^{(10)}$.

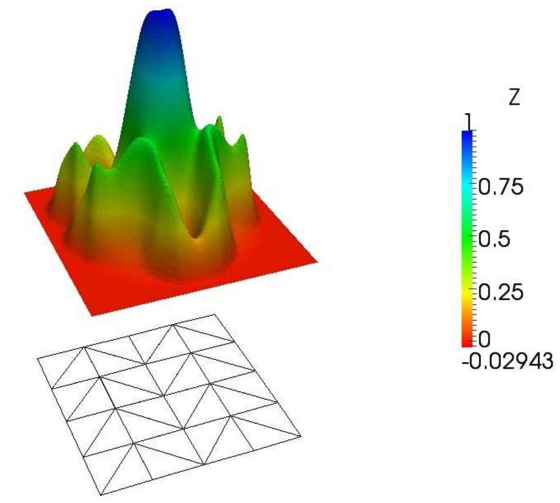


(a) Kronecker-δ property enforced only on boundary nodes.

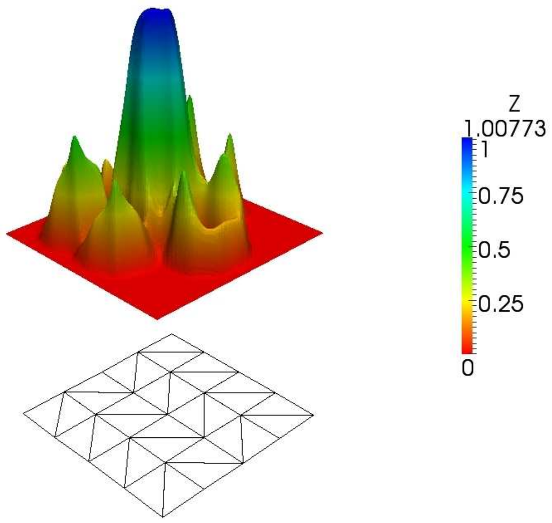


(b) Kronecker-δ property enforced on all the vertex nodes.

Figure 4.9 The global shape function of T10P3I1 element and symmetric mesh: $\Psi_I^{(01)}$.

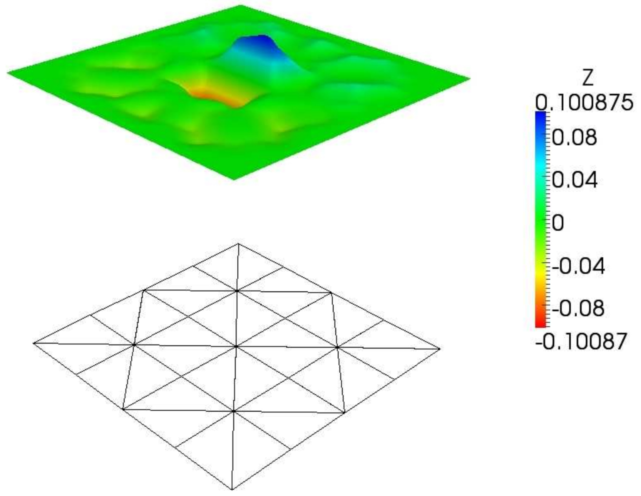


(a) Kronecker-δ property enforced only on boundary nodes.

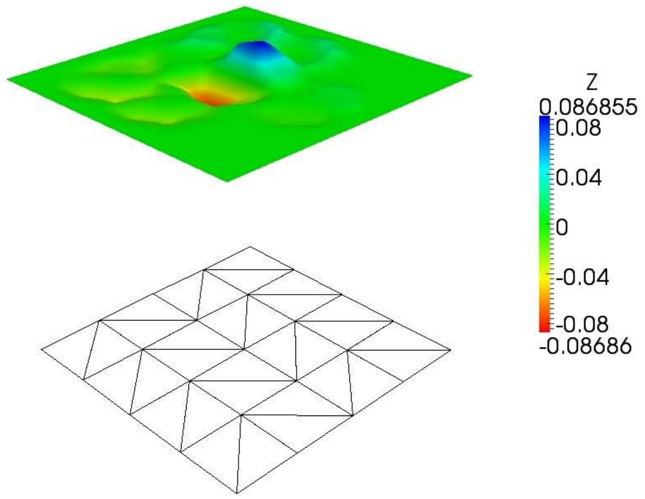


(b) Kronecker-δ property enforced on all the vertex nodes.

Figure 4.10 The global shape function of T10P3I1 element and asymmetric mesh: $\Psi_I^{(00)}$.

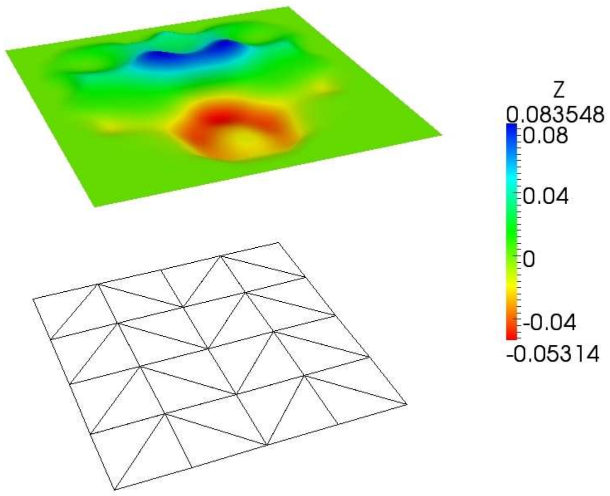


(a) Kronecker-δ property enforced only on boundary nodes.

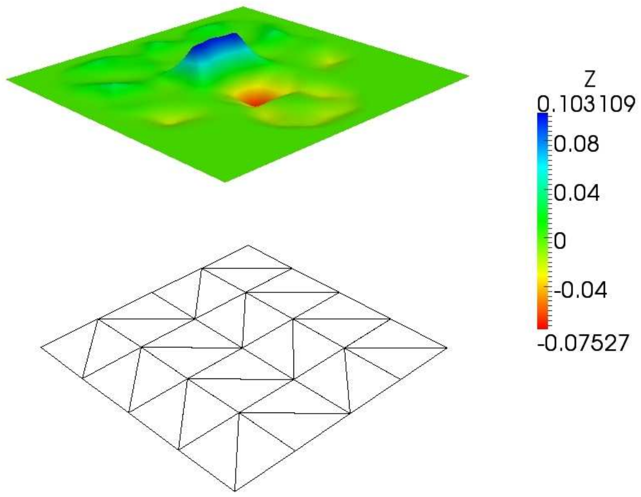


(b) Kronecker-δ property enforced on all the vertex nodes.

Figure 4.11 The global shape function of T10P3I1 element and asymmetric mesh: $\Psi_I^{(10)}$.



(a) Kronecker-δ property enforced only on boundary nodes.



(b) Kronecker-δ property enforced on all the vertex nodes.

Figure 4.12 The global shape function of T10P3I1 element and asymmetric mesh: $\Psi_I^{(01)}$.

$$\tilde{\psi}_5^{(200)} = \frac{w^2}{2} - \frac{w}{2} \tag{4.4g}$$

$$\tilde{\psi}_5^{(110)} = s\,t \tag{4.4h}$$

$$\tilde{\psi}_5^{(011)} = t\,w \tag{4.4i}$$

$$\tilde{\psi}_5^{(101)} = s\,w \tag{4.4j}$$

A different possible quadratic element can be constructed that violates the rule for construction of global partition polynomials by interpolating the three first partial derivatives, $\{u_{,x};\ u_{,y};\ u_{,z}\}$ and the three mixed second partial derivatives,$\{u_{,xy};\ u_{,yz};\ u_{,zx}\}$. Violating this rule does not actually preclude this element, but what it means is that the degrees of freedom associated with the second mixed partial derivatives will not transform from the parent triangle to the geometric triangle in such a way that corresponds to the mixed partials with respect to the geometric coordinates. As such, they can be treated as arbitrary *internal* degrees of freedom. We denote this element as Tet10P2I0*. One could similarly define an element where the interior degree-of-freedom are the first partials, and $\{u_{,xx};\ u_{,yy};\ u_{,zz}\}$, which we omit here. The global partition polynomials are:

$$\tilde{\psi}_1^{(000)} = -2w^2 + w - 2t^2 + t - 2s^2 + s + 1 \tag{4.5a}$$

$$\tilde{\psi}_2^{(000)} = 2s^2 - s \tag{4.5b}$$

$$\tilde{\psi}_3^{(000)} = 2t^2 - t \tag{4.5c}$$

$$\tilde{\psi}_4^{(000)} = 2w^2 - w \tag{4.5d}$$

$$\tilde{\psi}_5^{(100)} = 2s - 2s^2 \tag{4.5e}$$

$$\tilde{\psi}_5^{(010)} = 2t - 2t^2 \tag{4.5f}$$

$$\tilde{\psi}_5^{(001)} = 2w - 2w^2 \tag{4.5g}$$

$$\tilde{\psi}_5^{(110)} = \frac{t^2}{2} + st - \frac{t}{2} + \frac{s^2}{2} - \frac{s}{2} \tag{4.5h}$$

$$\tilde{\psi}_5^{(011)} = \frac{w^2}{2} + sw - \frac{w}{2} + \frac{s^2}{2} - \frac{s}{2} \tag{4.5i}$$

$$\tilde{\psi}_5^{(101)} = \frac{w^2}{2} + tw - \frac{w}{2} + \frac{t^2}{2} - \frac{t}{2} \tag{4.5j}$$

4.2.2 The Tet20P3I1 Tetrahedral

A cubic tetrahedral element, the Tet20P3I1 interpolates the primary variable and its first partial derivatives at each vertex and at an interior node. The global partition polynomials are:

$$\begin{aligned}\tilde{\psi}_1^{(000)} = {} & 2w^3 + 13tw^2 + 13sw^2 - 3w^2 + 13t^2w + 44stw - 13tw + 13s^2w \\ & - 13sw + 2t^3 + 13st^2 - 3t^2 + 13s^2t - 13st + 2s^3 - 3s^2 + 1\end{aligned} \tag{4.6a}$$

$$\tilde{\psi}_1^{(100)} = 2sw^2 + 5stw + 3s^2w - 3sw + 2st^2 + 3s^2t - 3st + s^3 - 2s^2 + s \tag{4.6b}$$

$$\tilde{\psi}_1^{(010)} = 2tw^2 + 3t^2w + 5stw - 3tw + t^3 + 3st^2 - 2t^2 + 2s^2t - 3st + t \tag{4.6c}$$

$$\tilde{\psi}_1^{(001)} = w^3 + 3tw^2 + 3sw^2 - 2w^2 + 2t^2w + 5stw - 3tw + 2s^2w - 3sw + w \tag{4.6d}$$

$$\begin{aligned}\tilde{\psi}_2^{(000)} = {} & -11tw^2 + 7sw^2 - 11t^2w - 4stw + 11tw + 7s^2w - 7sw + 7st^2 \\ & + 7s^2t - 7st - 2s^3 + 3s^2\end{aligned} \tag{4.6e}$$

$$\begin{aligned}\tilde{\psi}_2^{(100)} = {} & 3tw^2 - 2sw^2 + 3t^2w + stw - 3tw - 2s^2w + 2sw - 2st^2 - 2s^2t \\ & + 2st + s^3 - s^2\end{aligned} \tag{4.6f}$$

$$\tilde{\psi}_2^{(010)} = -tw^2 - t^2w - stw + tw + st^2 + 2s^2t - st \tag{4.6g}$$

$$\tilde{\psi}_2^{(001)} = -tw^2 + sw^2 - t^2w - stw + tw + 2s^2w - sw \tag{4.6h}$$

$$\tilde{\psi}_3^{(000)} = 7tw^2 - 11sw^2 + 7t^2w - 4stw - 7tw - 11s^2w + 11sw - 2t^3 + 7st^2$$
$$+ 3t^2 + 7s^2t - 7st \tag{4.6i}$$

$$\tilde{\psi}_3^{(100)} = -sw^2 - stw - s^2w + sw + 2st^2 + s^2t - st \tag{4.6j}$$

$$\tilde{\psi}_3^{(010)} = -2tw^2 + 3sw^2 - 2t^2w + stw + 2tw + 3s^2w - 3sw + t^3 - 2st^2$$
$$- t^2 - 2s^2t + 2st \tag{4.6k}$$

$$\tilde{\psi}_3^{(001)} = tw^2 - sw^2 + 2t^2w - stw - tw - s^2w + sw \tag{4.6l}$$

$$\tilde{\psi}_4^{(000)} = -2w^3 + 7tw^2 + 7sw^2 + 3w^2 + 7t^2w - 4stw - 7tw + 7s^2w$$
$$- 7sw - 11st^2 - 11s^2t + 11st \tag{4.6m}$$

$$\tilde{\psi}_4^{(100)} = 2sw^2 - stw + s^2w - sw - st^2 - s^2t + st \tag{4.6n}$$

$$\tilde{\psi}_4^{(010)} = 2tw^2 + t^2w - stw - tw - st^2 - s^2t + st \tag{4.6o}$$

$$\tilde{\psi}_4^{(001)} = w^3 - 2tw^2 - 2sw^2 - w^2 - 2t^2w + stw + 2tw - 2s^2w + 2sw + 3st^2$$
$$+ 3s^2t - 3st \tag{4.6p}$$

$$\tilde{\psi}_5^{(000)} = -16tw^2 - 16sw^2 - 16t^2w - 32stw + 16tw - 16s^2w + 16sw$$
$$- 16st^2 - 16s^2t + 16st \tag{4.6q}$$

$$\tilde{\psi}_5^{(100)} = 12tw^2 - 4sw^2 + 12t^2w + 8stw - 12tw - 4s^2w + 4sw$$
$$- 4st^2 - 4s^2t + 4st \tag{4.6r}$$

$$\tilde{\psi}_5^{(010)} = -4tw^2 + 12sw^2 - 4t^2w + 8stw + 4tw + 12s^2w - 12sw$$

$$-4st^2 - 4s^2t + 4st \tag{4.6s}$$

$$\tilde{\psi}_5^{(001)} = -4tw^2 - 4sw^2 - 4t^2w + 8stw + 4tw - 4s^2w + 4sw$$

$$+12st^2 + 12s^2t - 12st \tag{4.6t}$$

Chapter 5

Examples

In this chapter, three ways of using the RKEM interpolant are shown, one is used in a Galerkin weak form, the other is used to represent geometry and finally, for point interpolation. Some convergence results are presented for the newly developed T10P3I1 and T6P2I0 elements and also a comparison between convergence rates with the T9P2I1 element is presented. A Kirchhoff plate theory is used to show the performance of the new element in solving Galerkin weak forms. This problem was chosen, because of its importance in structural applications and because it involves interpolation of second derivatives. Three different problems were considered: clamped square plate, simply supported square plate and clamped circular plate. Same problems were used in [50] to test the T18P4I2 element. In the following, L_2, H_1 and H_2 are the standard Sobolev norms. Contrary to the C^0 elements used in finite element analysis of plates, where a mixed formulation is implemented to solve the problem, the T10P3I1 element could be used directly into a displacement based formulation and obtain accurate solution and optimal convergence rates. Furthermore, the RKEM interpolants are globally smooth functions and therefore, there is no need to apply smoothing techniques to the solution. In all the examples, we have enforced the Kronecker-δ property only on the boundary nodes, using the concept of nodal isolation presented in [17]. Furthermore, a fourth-order conical window function with a radial support has been used to calculate the kernel and 36 Gauss points per element were used to integrate the weak form, in contrast to the 576 Gauss points per element used before.

5.1 Plate Bending Approximation: Thin (Kirchhoff) Plates

The differential equation that governs the behavior of thin plates is

$$\frac{\partial^4 w}{\partial x^4} + 2\frac{\partial^2 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^4} = \frac{p}{D} \qquad \forall\,(x,y) \in \Omega \tag{5.1}$$

where Ω is the domain of the problem, D is the bending stiffness and p the transverse distributed load. In all the examples chosen, $p/D = 1$. To solve this equation using a Galerkin method, we use the weak form. The weak form specific to a simply supported and clamped plate then becomes,

$$\int \left(\nabla^2 w\right)\left(\nabla^2 \delta w\right) d\Omega = \int \frac{p}{D} \delta w \, d\Omega \tag{5.2}$$

For a clamped plate, the essential boundary conditions are

$$\begin{aligned} w &= 0 \qquad \forall\,(x,y) \in \delta\Omega \\ w_{,n} &= 0 \qquad \forall\,(x,y) \in \delta\Omega \\ w_{,s} &= 0 \qquad \forall\,(x,y) \in \delta\Omega \end{aligned}$$

For a Galerkin solution, these boundary conditions require the arbitrary variation δw to also satisfy

$$\delta w = 0 \qquad \forall (x,y) \in \delta\Omega$$

$$\delta w_{,n} = 0 \qquad \forall (x,y) \in \delta\Omega$$

$$\delta w_{,s} = 0 \qquad \forall (x,y) \in \delta\Omega$$

For a simply supported plate, the essential boundary conditions are

$$w = 0 \qquad \forall (x,y) \in \delta\Omega$$

For a Galerkin solution, these boundary conditions require the arbitrary variation δw to also satisfy

$$\delta w = 0 \qquad \forall (x,y) \in \delta\Omega$$

5.1.1 Clamped Square Plate

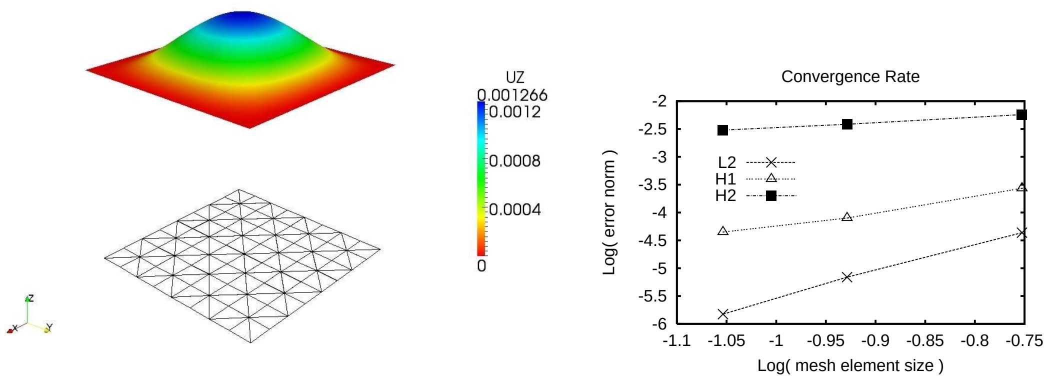
A unit clamped square plate subjected to a uniform transverse load is analyzed. The problem was solved using the T10P3I1 element. The boundary conditions were enforced by setting $w = 0$ and $w_{,x} = w_{,y} = 0$ at the boundary nodes. A solution for a plate of dimensions $a \times b$ is given in [53] as:

$$w(x,y) = \sum_{m=1}^{N} \sum_{n=1}^{N} w_{mn} \left(1 - \cos\frac{2m\pi x}{a}\right)\left(1 - \cos\frac{2m\pi y}{b}\right) \tag{5.3}$$

Table 5.1 Rates of convergence for the clamped square plate.

	L_2	H_1	H_2
T10P3I1	4.838 ± 0.2167	2.641 ± 0.2846	0.931 ± 0.0388
T9P2I1	2.128 ± 0.1236	$1.966 \pm 0.0.0959$	0.854 ± 0.1076

where the coefficients w_{mn} are computed using the method in [53]. 1000×1000 terms were used in Eq. (5.3) to compute the exact solution, and the maximum displacement at the middle point is given as $1.265319087 \times 10^{-3}$. The displacement profile of the plate and the convergence rates for the Galerkin solution are depicted in Fig. 5.1 and Table 5.1. A High convergence rate in L_2 error norm can be observed in the Galerkin solution. Also, the relative error in the maximum displacement at the middle point is less than 0.08%.



(a) Displacement profile (b) Convergence rate

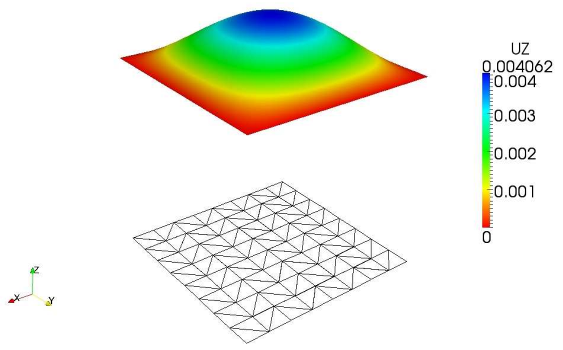
Figure 5.1 Convergence for the clamped square plate and the T10P3I1 element.

5.1.2 Simply Supported Square Plate

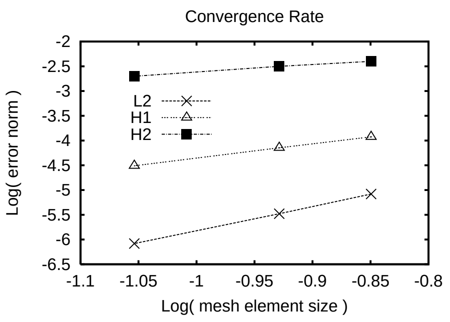
A unit simply supported square plate subjected to a uniform transverse load is analyzed. Again, the T10P3I1 element was used. The boundary conditions were enforced by setting $w = 0$ at the boundary nodes. A solution for a plate of dimensions $a \times b$ is given in [54] as:

$$w(x,y) = \frac{16p}{\pi^6 D} \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \frac{\sin \frac{m\pi x}{a} \sin \frac{n\pi y}{b}}{mn\left[(m/a)^2 + (n/b)^2\right]^2} \qquad (m,n = 1,3,5,\ldots) \tag{5.4}$$

The maximum displacement at the midpoint is given as $4.06234741 \times 10^{-3}$. The displacement profile of the plate and the convergence rates for the Galerkin solution are depicted in Fig. 5.2 and Table 5.2. Approximately same convergence rate in L_2 error norm can be observed in the Galerkin solution for this case and previous one. The relative error in the maximum displacement at the midpoint is exact up to four significant digits.



(a) Displacement profile



(b) Convergence rate

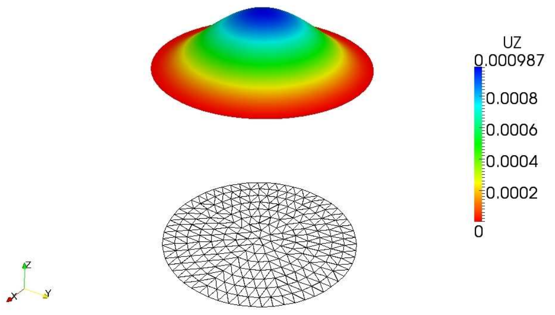
Figure 5.2 Convergence for the simply square plate and the T10P3I1 element.

Table 5.2 Rates of convergence for the simply square plate.

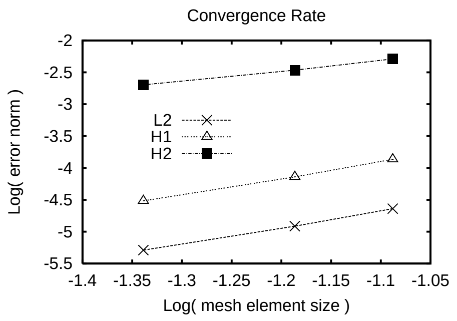
	L_2	H_1	H_2
T10P3I1	4.898 ± 0.0531	2.872 ± 0.0280	1.488 ± 0.0899
T9P2I1	3.044 ± 0.0418	2.711 ± 0.0583	1.116 ± 0.0175

5.1.3 Clamped Circular Plate

A unit clamped circular plate subjected to a uniform transverse load is analyzed. Again, the T10P3I1 element was used. The boundary conditions were enforced by setting $w = 0$,



(a) Displacement profile



(b) Convergence rate

Figure 5.3 Convergence for the circle clamped plate and the T10P3I1 element.

$w_{,x} = w_{,y} = 0$ at the boundary nodes. A solution for a circular plate of radius a is given in [54] as:

$$w(x,y) = \frac{p}{64D}\left(a^2 - x^2 - y^2\right)^2 \tag{5.5}$$

The maximum displacement at the midpoint is given as 9.76562×10^{-4}. The displacement profile of the plate and the convergence rates for the Galerkin solution are depicted in Fig. 5.3 and Table 5.3. The relative error in the maximum displacement at the midpoint is less than 1.02%. The convergence order for this problem is almost half the value of previous examples. It is related to the *Babuska paradox*, which states that no convergent approximation may be found, in clamped circular plate, if the curved boundary is replaced by a polygonal domain [16].

Table 5.3 Rates of convergence for the clamped circular plate.

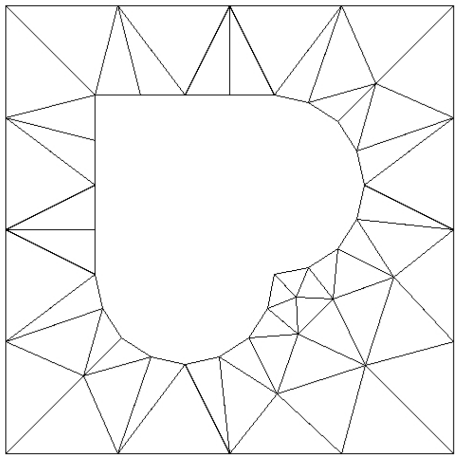
	L_2	H_1	H_2
T10P3I1	2.584 ± 0.090	2.585 ± 0.0947	1.613 ± 0.0727
T9P2I1	0.980 ± 0.2929	1.157 ± 0.2317	0.647 ± 0.2362

5.2 Surface Fitting Using Reproducing Kernel Element Shape Functions

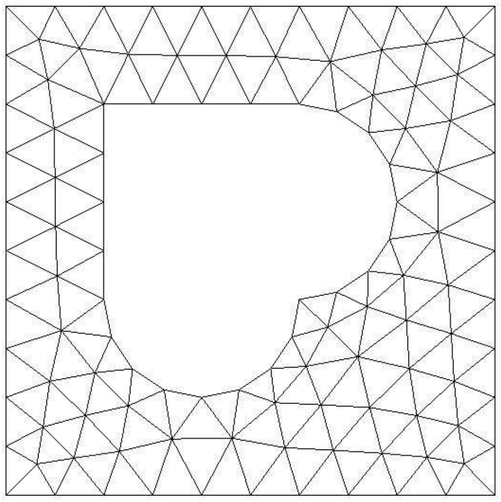
In this section we present the results of the interpolation of a surface using the reproducing kernel element shape functions associated with the new global partition polynomials developed in §4.1.1 and §4.1.2. The function to interpolate is defined in the 2D domain shown in Fig. 5.4 and is given by

$$f_1(x,y) = \sin(x)\cos(y) \tag{5.6}$$

The domain of the problem is a complex shape, but it is easy to mesh using triangular elements. There is no effort, from a user point of view, in the generation of the mesh, unlike meshes with quadrilaterals. Analyzing the results of Fig. 5.5 to Fig 5.8, it is clear that the RKEM functions are able to represent the solution better than using a simple linear interpolation scheme. Even for a coarse mesh the approximation is good. With a small refinement of the mesh we obtained with good accuracy an excellent representation of the exact shape. Figure 5.9 presents the convergence rate in L2 error norm of the interpolation for a linear triangular finite element, T6P2I0 and T10P3I1 reproducing kernel elements. The interpolation is performed in the irregular domain of Fig 5.4. As expected, the T10P3I1 has the better approximation property with respect to the L2 error norm. In Table 5.4 shows the comparison of the total number of degrees of freedom used for interpolation and the corresponding L2 error norm for different elements. It is important to note that with only 521 degrees of freedom, the T10P3I1 reproducing kernel Element has approximately the



(a) Coarse mesh



(b) Finer mesh

Figure 5.4 Background meshes used for interpolation.

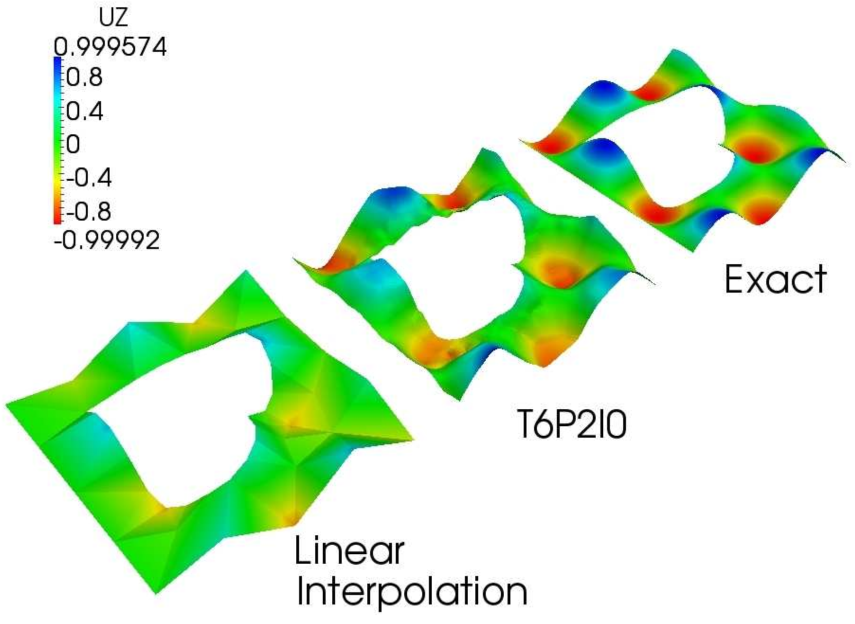


Figure 5.5 Interpolations for coarse mesh and exact solution, T6P2I0 element.

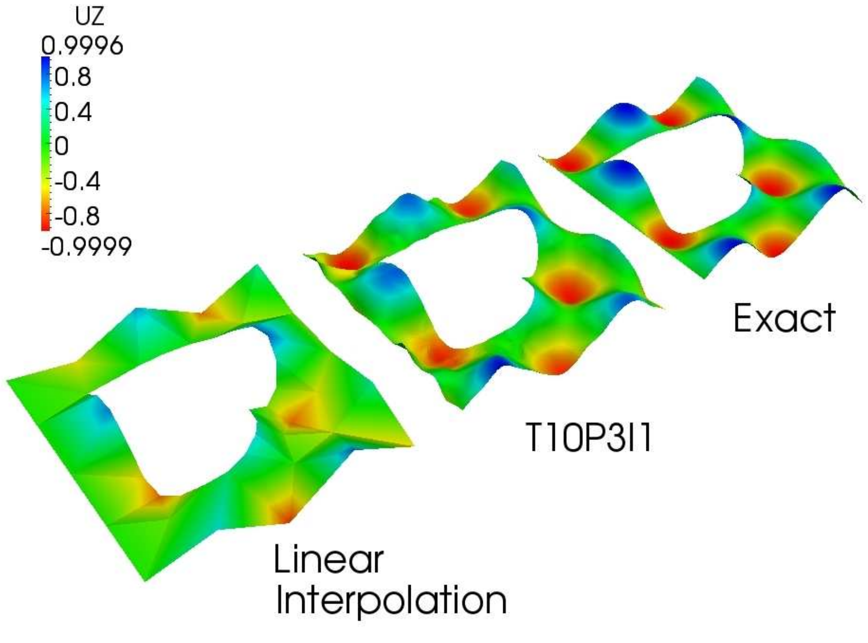


Figure 5.6 Interpolations for coarse mesh and exact solution, T10P3I1 element.

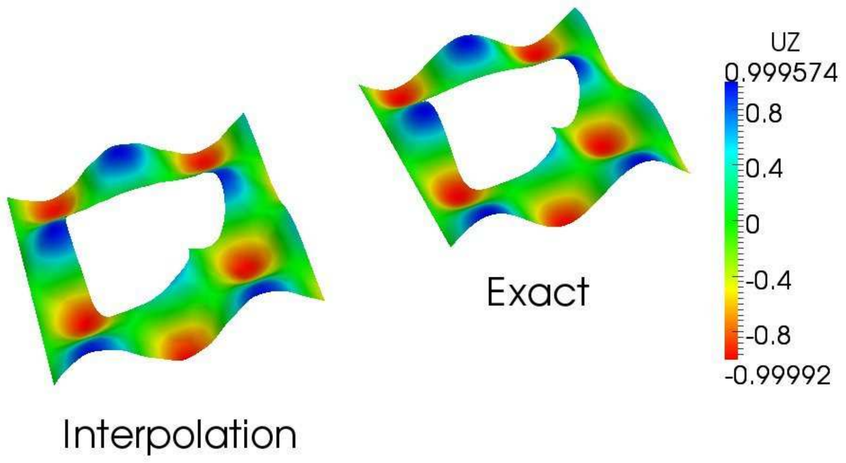


Figure 5.7 Interpolated and exact surface for the T6P2I0 element for finer mesh.

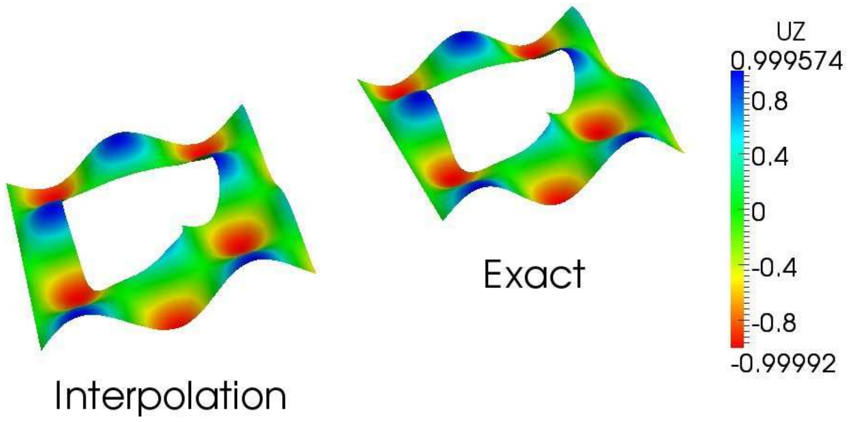


Figure 5.8 Interpolated and exact surface for the T10P3I1 element for finer mesh.

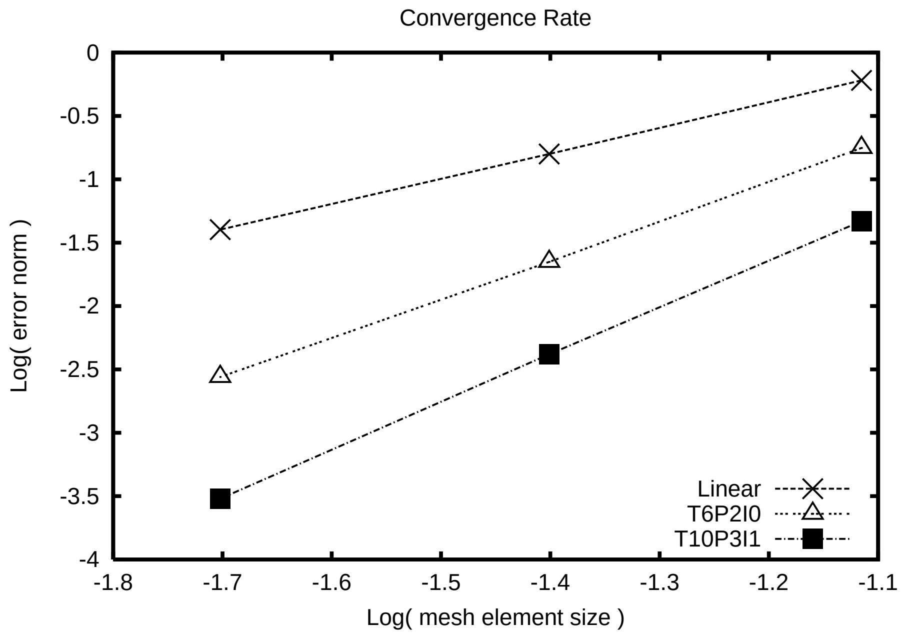


Figure 5.9 Convergence rates in L2 error norm of interpolation for the function $f_1(x,y) = \sin(x)\cos(y)$ in an irregular domain.

same error in the L2 norm as the linear triangular finite element with 1390 degrees of freedom. Additionally, the T10P3I1 element produces continuous interpolation, up to the first derivative, contrary to the linear triangular finite element.

As a second example of interpolation, we used a more complicated function that has a rough derivative. We will show that it will be a difficult test for the interpolation of the

Table 5.4 Comparison of the total number of degrees of freedom used for interpolation of $f_1(x,y)$ and the corresponding L2 error norm for different elements.

	L2 error norm	Degrees of freedom	# elements
Linear	0.04008	1390	2536
T10P3I1	0.04678	521	170

function using generalized Hermite type polynomials. The function is as follows,

$$f_2(x,y) = g(x) \cdot h(y) \tag{5.7a}$$

$$g(x) = (1-x)\left[\arctan\left(\alpha\left(x-\bar{x}\right)\right) + \arctan\left(\alpha\bar{x}\right)\right] \tag{5.7b}$$

$$h(y) = (1-y)\left[\arctan\left(\alpha\left(y-\bar{y}\right)\right) + \arctan\left(\alpha\bar{y}\right)\right] \tag{5.7c}$$

The functions given in Eq. (5.7b) and Eq. (5.7c) changes its roughness as the parameter α varies [38]. It becomes smoother as the parameter α gets smaller, and the graph of the function has a sharp knee at $x = \bar{x}$ (i.e., Fig. 2.2). In this example we have chosen $\alpha = 50.0$ and $\bar{x} = 0.40$. The domain of the problem corresponds to the bi-unit square, $\Omega = (0,1) \times (0,1)$. Contrary to the previous example, where the function and its derivatives are smooth, here, the number of degrees of freedom needed to have approximately the same error as the linear triangular finite element is bigger, as presented in Table 5.5. The DOF presented in Table 5.5 were computed for different meshes. For the linear finite element triangular element a finer mesh was used (3872 elements), for the T6P2I0 a mesh with 2312 element was adopted and for the T10P3I1 a rough mesh was utilized (1152 elements). We presume that the increase of DOF is because with the T10P3I1 we are interpolating the primary variable and the derivative. Therefore, to capture the roughness of the derivative, as shown in Fig 5.10, we will need more degrees of freedom than in the case where derivatives are smooth. As a consequence, if the derivative needs to be interpolated, then the T10P3I1 element will have a better behavior that the linear triangular finite element. The same effect of the T10P3I1 is seen in the T6P2I0 element. The convergence rate and the graph of the interpolation are shown in Fig 5.11 and Fig 5.12, respectively. In Fig. 5.13 the difference between the interpolation and the exact function for the T10P3I1 and linear triangular finite elements for a finer mesh is plotted. To explain the increase of DOF's, imagine that we want to interpolate a function in two dimensions using the T6P2I0 element, then

$$u = \sum_{I}^{M} \Psi_I^{(00)} u^I + \sum_{J}^{N} \left[\Psi_J^{(00)} u^J + \Psi_J^{(10)} u^J_{,x} + \Psi_J^{(01)} u^J_{,y} \right] \tag{5.8}$$

where we have omitted the arguments of the functions and the first summation is over the set of vertex nodes, and the second summation is over the set of interior nodes. If the nodal weights associated with the derivatives change abruptly and they have a value much bigger than the nodal weight associated with the main variable, then the interpolation will have a strong gradient. In Fig. 5.14 the case is presented where we are interpolating the function $f_2(x,y)$ for two different values of α using same mesh. If we use same mesh for interpolation, then the RKEM shape functions will be the same for both interpolations, hence the only difference will be in the nodal weights. For the case where $\alpha = 5.0$ we have a smooth representation, contrary to the case where $\alpha = 50.0$. In Fig. 5.14(b) the jagged surface is a result of the RKEM interpolation and it is not the actual geometry. This example shows the deleterious effect when the function has a strong gradient in the derivative and a relatively corse mesh is used for interpolation. Under refinement, the "abrupt" changes are reduced; and therefore a better approximation is obtained.

Table 5.5 Comparison of the total number of degrees of freedom used for interpolation of $f_2(x,y)$ and the corresponding L2 error norm for different elements.

	L2 error norm	Degrees of freedom	# elements
Linear	0.008014	2025	3872
T6P2I0	0.006887	8161	2312
T10P3I1	0.008315	3027	1152

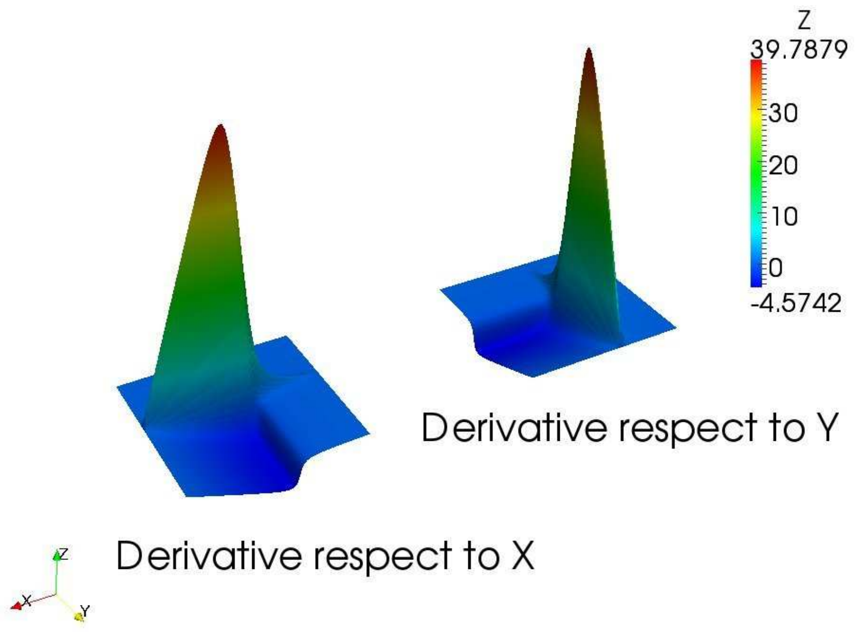


Figure 5.10 Derivatives of the function $f_2(x,y) = g(x) \cdot h(y)$ for $\alpha = 50.0$; and height was scaled by 0.04 for illustration purpose.

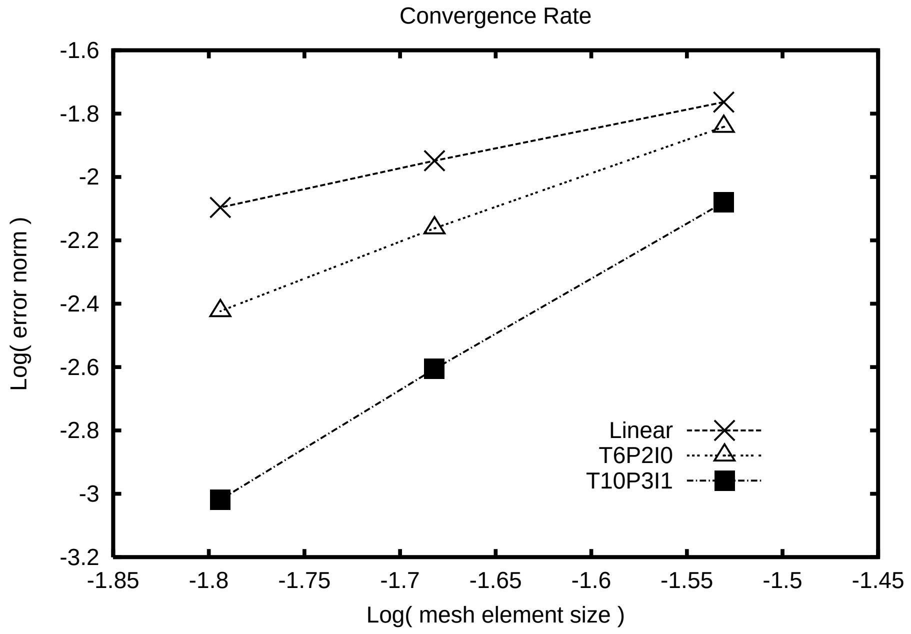


Figure 5.11 Convergence rates in L2 error norm of interpolation for the function $f_2(x,y) = g(x) \cdot h(y)$ in a regular domain.

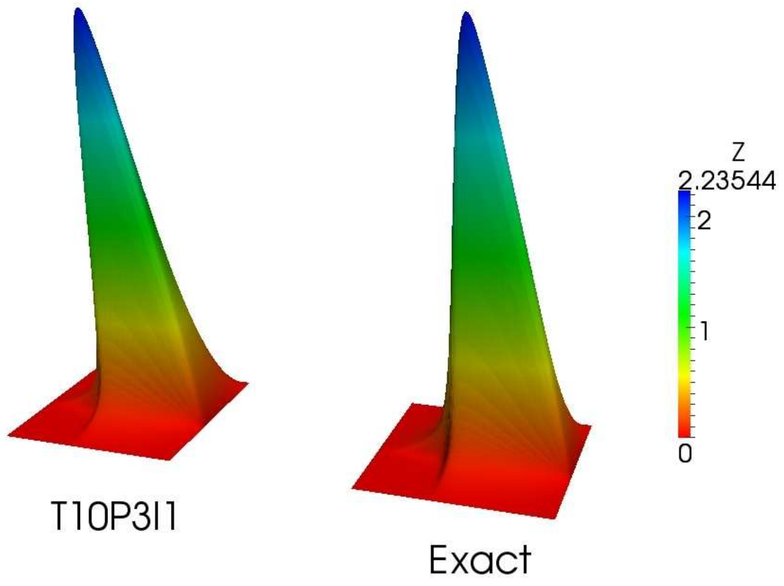
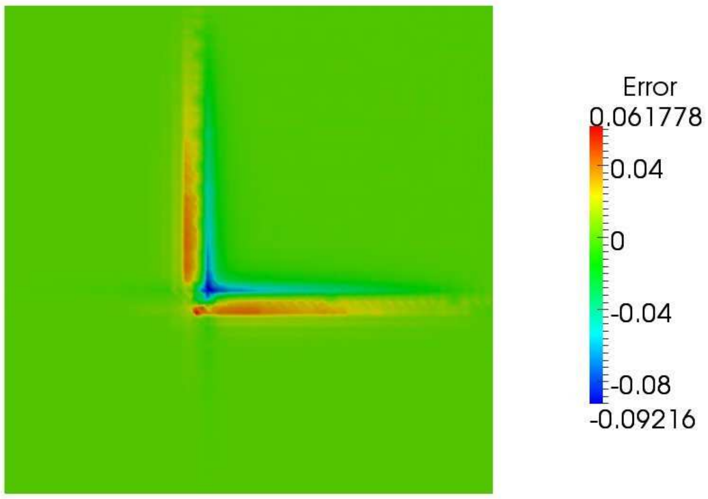
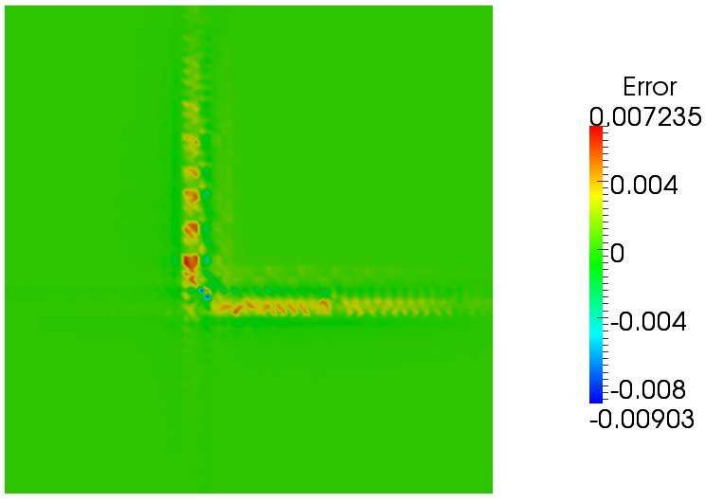


Figure 5.12 Interpolation of the function $f_2(x,y) = g(x) \cdot h(y)$ in a finer mesh for the T10P3I1 element and comparison with exact function for $\alpha = 50$.

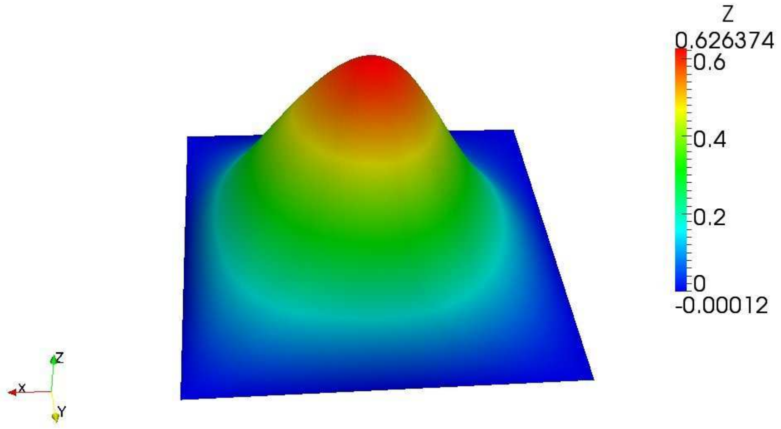


(a) Interpolation using linear finite element triangles

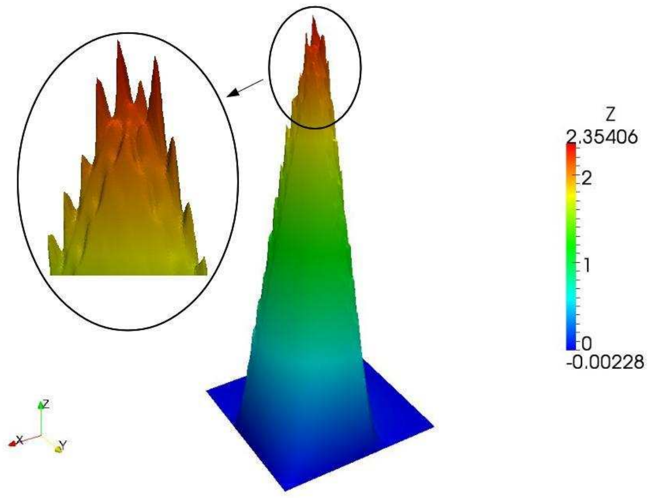


(b) Interpolation using T10P3I1 elements triangles

Figure 5.13 Absolute error of the interpolation of the function $f_2(x,y) = g(x) \cdot h(y)$ in a finer mesh (3872 elements) for the T10P3I1 element and the linear finite element triangle.



(a) $\alpha = 5.0$



(b) $\alpha = 50.0$.

Figure 5.14 Interpolation of the function $f_2(x,y) = g(x) \cdot h(y)$ in same mesh for the T6P2I0 element and different values of α.

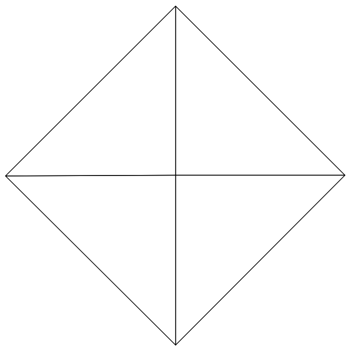
5.3 Volume Representation

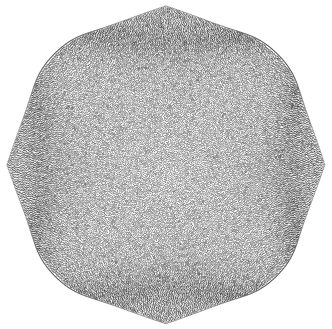
We now present an example in two dimensions of volume representation of a circle using RKEM. The theory behind volume representation is found in [49] and it is explained briefly here. Given a boundary point set, a perpendicular projection of the point onto the nearest RKEM boundary edge is found. We called these points a boundary auxiliary point set. Next, we must also determine a set of auxiliary points for the interior. Our procedure is to take the RKEM mesh points from the boundary auxiliary point list and use a mesh generator to generate the interior. This gives us a set of auxiliary points, u_i, for the entire body. At this point, we only know the associated body points for the boundary. To determine the body points, x_i, for the interior, we use a linear finite element solution on the volume mesh using all essential boundary conditions on the known boundary auxiliary points. This generates a displacement field for the interior points that we use to populate the interior auxiliary point pairs. Given the auxiliary points, we now solve a set of linear algebraic equations like Eq (5.9) and solve in a minimum least-square sense for the nodal data.

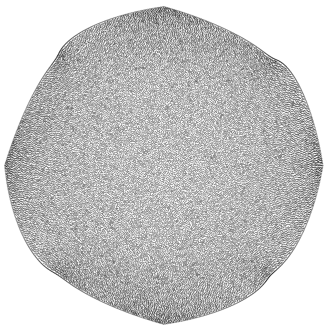
$$x_i = \tilde{x}(u_i); \quad i = 1 \ldots M \tag{5.9}$$

where M is the number of body points and $\tilde{x}(u_i)$ is the RKEM interpolation evaluated at the auxiliary point u_i.

In Fig 5.15 we show the results of the volume representation of a circle, using three different RKEM elements. A total of 36 equally spaced geometric boundary points, representing a circle, were projected onto the RKEM boundary edges. For all the cases a four element mesh as shown in Fig 5.15(a) were used. Some aspect should be pointed out. We are trying to represent a circle using few points on the boundary and a coarse RKEM mesh, therefore, it represents an extreme case. The presence of the "cusps" in the representation was discussed for the first time in [18] for boundary and volume representation using the T9P2I1 and Tet16P2I1 elements. It was concluded that the presence of the cusps are due to the fact that we are projecting data that is fundamentally smooth onto boundary edges that are not smooth. In this particular example, for the four corner nodes, an inconsistency arises because we cannot automatically have there the necessary conditions for continuous functions for arbitrary parameters of the nodal data. The effect of the cusp is more evident in the T6P2I0 element because this element does not interpolate derivatives at the corner nodes. The lack of symmetry of the global partition polynomials for the T9P2I1 element has an impact in the volume representation, at least, when a coarse mesh is used. On the contrary, the T10P3I1 can represent the circle almost exactly and does not suffer from symmetry issues.


(a) RKEM mesh


(b) T6P2I0


(c) T9P2I1

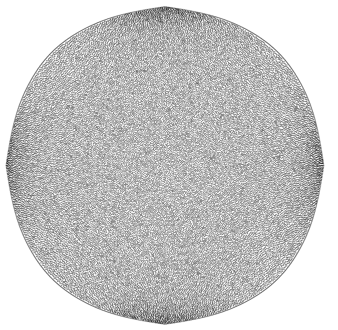

(d) T10P3I1

Figure 5.15 Volume representation of a circle for different RKEM elements.

Chapter 6

Conclusions

We have proposed a new methodology to construct truly minimal and symmetric global partition polynomials that have the desired Kronecker-δ property at the nodes. We have shown that the previous two-dimensional RKEM triangular elements suffer from an asymmetry problem and it is because they are not *affine-invariant*. Consequently, previous formulations for construction of RKEM elements should be abandoned, in favor of the new formulation proposed in this dissertation. A key point in this new formulation is the inclusion of an interior node. Unlike other methods, where adding more nodes complicates the meshing process and even destroys the compatibility of the elements, in our methodology the inclusion of the interior node does not involve any extra effort from a meshing point of view and still we have globally compatible elements. Indeed, it is done automatically when an element is created. It means that we can use with absolutely no modifications the same mesh for all the two-dimensional RKEM triangular elements. The new T10P3I1 element was used in a Galerkin weak form in order to solve some Kirchhoff plate bending problems to test numerically the convergence rate, and also, the same problems were solved using the previously developed T9P2I1 element for comparison purpose. The performance of the T10P3I1 element proved to be superior to the T9P2I1 element. The imposition of the Kronecker-δ property only on the nodes where a Dirichlet boundary condition needs to be enforced allowed us to reduce the number of Gauss points required to integrate the weak form. From the original 576 Gauss points per element used in [50] we moved to 36 Gauss points per element. The relaxation of the Kronecker-δ property has the effect of smoothing the RKEM shape functions. Further research in this area should be conducted in order to find an "optimal" quadrature rule. The use of the T10P3I1 RKEM interpolant for volume representation showed an improvement over the T6P2I0 and T9P2I1 elements.

6.1 Future Works

This dissertation explains the new way of construction of minimal and symmetric global partition polynomials needed in the formulation of RKEM elements. Still there are some future work to explore. A number of these are summarized below,

a) A rigorous mathematical error analysis for the new elements should be performed in order to evaluate, theoretically, the approximation property of the RKEM space of functions. This is a necessary step in order to lay out a firm mathematical foundation of the method. Although an error analysis was done in [38] for the construction of an interpolation error for RKEM elements with linear reproducing property.

b) The reproducing kernel element method requires a great effort to compute accurately the integrals appearing in a weak formulation by using a conventional numerical integration scheme such as the Gaussian quadrature rule. The previous drawback could be potentially eliminated by the implementation of exact integration formulas for the reproducing kernel element shape functions, as present in [8].

c) The reproducing kernel element method produces global and smooth higher-order shape functions, as a result, they are continuously differentiable, or $C^k(\Omega)$ type, where often, $k \geq 1$. Consequently, post-processing of derived variables (i.e. strain and stress) is performed using a direct interpolation and differentiation. However, this leads to a difficulty when there is an imposed discontinuity in the derivatives, such as the discontinuity in strains when material discontinuities are present [6]. Hence, it is important to have a method to impose this physical discontinuity. In [43] a first attempt to introduce discontinuities in the reproducing kernel element approximation is presented. As part of future research, an extension of the methodology presented in [43] should be done for the new elements proposed in this dissertation.

d) Only plate elements were implemented in this dissertation, but the use of RKEM shell elements in linear and non-linear analyzes should be explored. The analysis of shell structures is necessary in engineering calculations of stresses and strains of flat and curved surfaces (i.e., stresses in the fuselage of an airplane).

e) While having a higher-order interpolation capability could be beneficial respect to the solution accuracy viewpoint, it is in general more expensive to compute that lower-order interpolations. Therefore, coupling lower-order Finite Elements with RKEM elements would be advantageous. The blending of Finite Elements and RKEM elements could speed up the solution time process if only the higher-order RKEM elements are needed in a small sub-domain of the problem.

List of References

[1] Ansys Inc. *Ansys 13.0 Reference Guide*, November 2010.

[2] H. Antes. Bicubic fundamental splines in plate bending. *International Journal for Numerical Methods in Engineering*, 8(3):503–511, 1974.

[3] Tom Apostol. *Calculus*, volume 1. Wiley, 1967.

[4] K. Atkinson and A. Sharma. A partial characterization of poised Hermite-Birkhoff interpolation problems. *SIAM Journal on Numerical Analysis*, 6(2):230–235, June 1969.

[5] K.E. Atkinson and W. Han. *Theoretical numerical analysis: a functional analysis framework*. Texts in applied mathematics. Springer, 2001.

[6] Satya N. Atluri. *The Meshless Method (MLPG) for Domain & BIE Discretizations*. Tech Science Press, USA, 2004.

[7] Ivo Babuska, Uday Banerjee, and John E. Osborn. Survey of meshless and generalized finite element methods: A unified approach. *Acta Numerica*, 12:1–125, 2003.

[8] E. Barbieri and M. Meo. Evaluation of the integral terms in reproducing kernel methods. *Computer Methods in Applied Mechanics and Engineering*, 198(33-36):2485 – 2507, 2009.

[9] Klaus-Jurgen Bathe. *Finite Element Procedures*. Prentice Hall Inc., 1996.

[10] Stephen Beissel and Ted Belytschko. Nodal integration of the element-free galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 139(1-4):49 – 74, 1996.

[11] K. Bell. A refined triangular plate bending finite element. *International Journal for Numerical Methods in Engineering*, 1:101–122, 1969.

[12] Ted Belytschko, Yury Krongauz, Mark Fleming, Daniel Organ, and Wing Kam Snm Liu. Smoothing and accelerated computations in the element free galerkin method. *Journal of Computational and Applied Mathematics*, 74(1-2):111 – 126, 1996.

[13] Garrett Birkhoff and Lois Mansfield. Compatible triangular finite elements. *Journal of Mathematical Analysis and Applications*, 47:531–553, 1974.

[14] C. Caramanlian, K. A. Selby, and G. T. Will. A quintic conforming plate bending triangle. *International Journal for Numerical Methods in Engineering*, 12(7):1109–1130, 1978.

[15] Jiun-Shyan Chen, Cheng-Tang Wu, Sangpil Yoon, and Yang You. A stabilized conforming nodal integration for galerkin mesh-free methods. *International Journal for Numerical Methods in Engineering*, 50(2):435–466, 2001.

[16] P.G. Ciarlet. *The finite element method for elliptic problems*. Studies in mathematics and its applications. North-Holland Pub. Co., 1978.

[17] N. Collier and D. C. Simkins, Jr. The quasi-uniformity condition for reproducing kernel element method meshes. *Computational Mechanics*, 44(3):333, 2009.

[18] Nathaniel O. Collier. *The Quasi-Uniformity Condition and Three-Dimensional Geometry Representation as it Applies to the Reproducing Kernel Element Method*. PhD thesis, University of South Florida, Tampa, FL, March 2009.

[19] S. De and K. J. Bathe. The method of finite spheres. *Computational Mechanics*, 25:329–345, 2000. 10.1007/s004660050481.

[20] S. De, J.-W. Hong, and K. J. Bathe. On the method of finite spheres in applications: towards the use with adina and in a surgical simulator. *Computational Mechanics*, 31:27–37, 2003. 10.1007/s00466-002-0390-3.

[21] Suvranu De and Klaus-Jurgen Bathe. The method of finite spheres with improved numerical integration. *Computers & Structures*, 79(22-25):2183 – 2196, 2001.

[22] John Dolbow and Ted Belytschko. Numerical integration of the galerkin weak form in meshfree methods. *Computational Mechanics*, 23:219–230, 1999. 10.1007/s004660050403.

[23] Sonia Fernandez-Mendez and Antonio Huerta. Imposing essential boundary conditions in mesh-free methods. *Computer Methods in Applied Mechanics and Engineering*, 193(12-14):1257 – 1275, 2004. Meshfree Methods: Recent Advances and New Applications.

[24] B. Fraeijs de Veubeke. A conforming finite element for plate bending. *International Journal of Solids and Structures*, 4:95–108, 1968.

[25] Christophe Geuzaine and Jean-Francois Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[26] T.J.R. Hughes. *The Finite Element Method*. Dover, Mineola, New York, 2000.

[27] T.J.R Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[28] Bruce M. Irons. A conforming quartic triangular element for plate bending. *International Journal for Numerical Methods in Engineering*, 1:29–45, 1969.

[29] Daniel C. Simkins, Jr., Nathan Collier, Mario Juha, and Lisa B. Whitenack. A framework for studying the rkem representation of discrete point sets. In Michael Griebel and Marc Alexander Schweitzer, editors, *Meshfree Methods for Partial Differential Equations IV*, volume 65 of *Lecture Notes in Computational Science and Engineering*, pages 301–314, Berlin, 2008. Springer-Verlag.

[30] D.R. Kincaid and E.W. Cheney. *Numerical analysis: mathematics of scientific computing*. Brooks/Cole series in advanced mathematics. Brooks/Cole, 2002.

[31] P. Krysl and T. Belytschko. Analysis of thin plates by the element-free galerkin method. *Computational Mechanics*, 17:26–35, 1995. 10.1007/BF00356476.

[32] Vitor M. A. Leitao. A meshless method for kirchhoff plate bending problems. *International Journal for Numerical Methods in Engineering*, 52(10):1107–1130, 2001.

[33] S. Li, H. Lu, W. Han, W. K. Liu, and D. C. Simkins, Jr. Reproducing kernel element method, Part II. Global conforming I^m/C^n hierarchy. *Computer Methods in Applied Mechanics and Engineering*, 193:953–987, 2004.

[34] Shaofan Li and Wing Kam Liu. Meshfree and particle methods and their applications. *Applied Mechanics Reviews*, 55(1):1–34, 2002.

[35] Shaofan Li and Wing Kam Liu. *Meshfree Particle Methods*. Springer, Berlin, 2004.

[36] G.R. Liu. *Mesh free methods: Moving beyond the finite element method*. CRC Press LLC, United States of America, 2003.

[37] G.R. Liu and Nguyen Thoi Trung. *Smoothed Finite Element Methods*. CRC Press, first edition, June 2010.

[38] W. K. Liu, W. Han, H. Lu, S. Li, and J. Cao. Reproducing kernel element method: Part I. Theoretical formulation. *Computer Methods in Applied Mechanics and Engineering*, 193:933–951, 2004.

[39] Yan Liu and Ted Belytschko. A new support integration scheme for the weakform in mesh-free methods. *International Journal for Numerical Methods in Engineering*, 82(6):699–715, 2010.

[40] Shuyao Long and S. N. Atluri. A meshless local petrov-galerkin method for solving the bending problem of a thin plate. *CMES*, 3(1):55–63, 2002.

[41] Rudolph A. Lorentz. *Multivariate Birkhoff Interpolation*. Number 1516 in Lecture Notes in Mathematics. Springer-Verlag, Berlin, 1992.

[42] H. Lu, S. Li, D. C. Simkins, Jr., W. K. Liu, and J. Cao. Reproducing kernel element method Part III. Generalized enrichment and applications. *Computer Methods in Applied Mechanics and Engineering*, 193:989–1011, 2004.

[43] Hongsheng Lu, Do Wan Kim, and Wing Kam Liu. Treatment of discontinuity in the reproducing kernel element method. *International Journal for Numerical Methods in Engineering*, 63(2):241–255, 2005.

[44] Y.Y. Lu, T. Belytschko, and L. Gu. A new implementation of the element free galerkin method. *Computer Methods in Applied Mechanics and Engineering*, 113(3-4):397 – 414, 1994.

[45] Maxima.sourceforge.net. *Maxima, a Computer Algebra System.* Maxima. sourceforge. net, 2009.

[46] J.T. Oden. *Finite elements of nonlinear continua*. Dover Publications, 2006.

[47] G. Sansone. *Orthogonal functions*. Pure and applied mathematics. Interscience Publishers, 1959.

[48] Jonathan Richard Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry*, 22(1-3):21 – 74, 2002.

[49] D. C. Simkins, Jr., A. Kumar, N. Collier, and L.B. Whitenack. Geometry representation, modification and iterative design using RKEM. *Computer Methods in Applied Mechanics and Engineering*, 196:4304–4320, 2007.

[50] D. C. Simkins, Jr., S. Li, H. Lu, and W. K. Liu. Reproducing kernel element method Part IV. Globally compatible $C^n(n \geq 1)$ triangular hierarchy. *Computer Methods in Applied Mechanics and Engineering*, 193:1013–1034, 2004.

[51] Daniel C. Simkins, Jr. *General Reproducing Kernel Element Hierarchies*. PhD thesis, University of California, Berkeley, CA, May 2004.

[52] A.H. Stroud and Don Secrest. *Gaussian Quadrature Formulas*. Prentice Hall Inc., Englewood Cliffs, New Jersey, 1966.

[53] R. L. Taylor and S. Govindjee. Solution of clamped rectangular plate problems. *Communications in Numerical Methods in Engineering*, 20:757–765, 2004.

[54] A. C. Ugural. *Stresses In Plates And Shells*. McGraw-Hill, Boston, Massachusetts, second edition, 1999.

[55] L. Piegl, W. Tiller. *The NURBS book*. Monographs in visual comunication. Springer, Germany, 1997.

[56] Dongdong Wang and Jiun-Shyan Chen. A hermite reproducing kernel approximation for thin-plate analysis with sub-domain stabilized conforming integration. *International Journal for Numerical Methods in Engineering*, 74(3):368–390, 2008.

[57] Dongdong Wang and Zhenting Lin. Free vibration analysis of thin plates using hermite reproducing kernel galerkin meshfree method with sub-domain stabilized conforming integration. *Computational Mechanics*, 46:703–719, 2010.

[58] Dongdong Wang and Zhenting Lin. Dispersion and transient analyses of hermite reproducing kernel galerkin meshfree method with sub-domain stabilized conforming integration for thin beam and plate structures. *Computational Mechanics*, pages 1–17, 2011. 10.1007/s00466-011-0580-y.

[59] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method*, volume 1. Butterworth-Heinemann, Oxford, 5 edition, 2000.

Appendices

Appendix A: Sobolev Norms

In numerical analysis the concept of Sobolev norms is used to evaluate the convergence rate of different approximations [5]. One important motivation for the adoption of these measure norms is its relations with the weak form of a boundary value problem. In this appendix we will describe three Sobolev norms used in this dissertation. No new information is described in this appendix. According to [26], consider $\Omega \subset \Re^n$, $n \geq 1$, and let $u, v : \Omega \rightarrow \Re$.

The $L_2(\Omega)$ inner product and norm are defined by

$$(u,v) = \int_\Omega uv\, d\Omega \tag{A.1}$$

and

$$\|u\|_0 = (u,v)^{1/2} \tag{A.2}$$

respectively. For the computation of the L_2 error norm used in this dissertation, Gaussian quadrature was employed to evaluate the integral, as follows,

$$\|e\|_0 = \left\{ \sum_{\Lambda_E} \sum_{\Lambda_g} \left[\mathscr{I} f(\mathbf{x}_g) - f(\mathbf{x}_g) \right]^2 W_g \right\}^{1/2} \tag{A.3}$$

where, Λ_g is the set of Gauss points in an element, $\mathbf{x}_g$ is a Gauss point and W_g is the Gauss point weigth multiplied by the determinant of the Jacobian. This norm is used to evaluate the interpolation error of main variables, for example, displacements.

The $H_1(\Omega)$ inner product and norm are defined by

$$(u,v)_1 = \int_\Omega (uv + u_{,i}v_{,i})\, d\Omega \qquad (\text{sum, } 1 \leq i \leq n) \tag{A.4}$$

and

$$\|u\|_1 = (u,v)_1^{1/2} \tag{A.5}$$

respectively. The computation of the H_1 error norm was performed in the same way as the $L_2(\Omega)$ error norm, but including the first derivatives. This norm is used to evaluate the interpolation error of main variables and its first derivative, for example, displacements and rotations in the classical Kirchhoff theory of plates.

The $H_2(\Omega)$ inner product and norm are defined by

$$(u,v)_2 = \int_\Omega \left(uv + u_{,i}v_{,i} + u_{,ij}v_{,ij}\right) d\Omega \qquad (\text{sum, } 1 \leq i,j \leq n) \tag{A.6}$$

and

$$\|u\|_2 = (u,v)_2^{1/2} \tag{A.7}$$

repectively. The computation of the $H_2(\Omega)$ error norm was done using Gaussian quadrature, including the second derivatives. This norm is used to evaluate the interpolation error of main variables, its first and second derivatives, for example, displacements, rotations and moments in the analysis of bending of thin plates. The $H_m(\Omega)$ inner product and norm are defined by

$$(u,v)_m = \int_\Omega \left(uv + u_{,i} v_{,i} + u_{,ij} v_{,ij} + \cdots + u_{\underbrace{,ij\cdots k}_{m \text{ indices}}} v_{\underbrace{,ij\cdots k}_{m \text{ indices}}} \right) d\Omega \qquad (\text{sum, } 1 \le i,j \le n) \tag{A.8}$$

and

$$\|u\|_m = (u,v)_m^{1/2} \tag{A.9}$$

Appendix B: Aspect Ratio

A convenient way to test is a mesh is well graded is to use the concept of aspect ratio, that is defined as follow.

According to [26], consider a domain $\Omega \subset \Re^d$ and $\bar{\Omega} = \cup_{e=1}^{n_{el}} \bar{\Omega}^e$. Let h^e be the diameter of smallest circle that contain the element Ω^e, then the aspect ratio is given by

$$\sigma = \frac{h}{\rho} \tag{B.1}$$

where,

$$h^e = \text{diameter } \Omega^e \tag{B.2}$$

$$\rho^e = \text{diameter of largest sphere contained in } \Omega^e \tag{B.3}$$

$$h = \max_{1 \leq e \leq n_{el}} (h^e) \tag{B.4}$$

$$\rho = \min_{1 \leq e \leq n_{el}} (\rho^e) \tag{B.5}$$

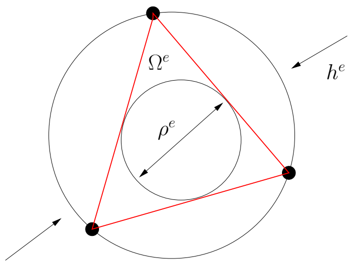


Figure B.1 Schematic of the smallest circle that contains the element Ω^e.