2-1-2010

Application of the b-spline collocation method to a geometrically non-linear beam problem

Jason Magoon

Application of the B-spline Collocation Method to a Geometrically Non-Linear Beam Problem

By

Jason Magoon

A Thesis Submitted in Partial Fulfillment of the Requirement
For Master of Science Degree in Mechanical Engineering

Approved by:

Dr. Hany Ghoneim - *Thesis Adviser* _____
Department of Mechanical Engineering

Dr. Steven Weinstein _____
Department Head of Chemical Engineering

Dr. P.Venkataraman _____
Department of Mechanical Engineering

Dr. Alan Nye _____
Associate Department Head of Mechanical Engineering

Department of Mechanical Engineering
Rochester Institute of Technology
Rochester, NY 14623
February 2010

Permission to Reproduce the Thesis

Application of the B-spline Collocation Method to a Geometrically Non-Linear Beam Problem

I, JASON MAGOON, hereby grant permission to the Wallace Memorial Library of Rochester Institute of Technology to reproduce my thesis in the whole or part. Any reproduction will not be for commercial use or profit.

Date: _____ Signature: _____

February 2010

Abstract

Engineers are researching solutions to resolve many of today's technical challenges. Numerical techniques are used to solve the mathematical models in engineering problems. Many of the mathematical models of engineering problems are expressed in terms of Boundary Value Problems, which are partial differential equations with boundary conditions. Two of the most popular techniques for solving partial differential equations are the Finite Element Method and the Finite Difference Method. In the last few decades another numerical technique has been increasingly used to solve mathematical models in engineering research, the B-spline Collocation Method. A collocation method involves satisfying a differential equation to some tolerance at a finite number of points, called collocation points. The B-spline Collocation Method does have a few distinct advantages over the Finite Element and Finite Difference Methods. The advantage over the Finite Difference Method is that the B-spline Collocation Method efficiently provides a piecewise-continuous, closed form solution. An advantage over the Finite Element Method is that the B-spline Collocation Method procedure is simpler and it is easy to apply to many problems involving partial differential equations. Although there are some advantages to using the B-spline Collocation Method, there are also disadvantages. The main disadvantage of the B-spline Collocation Method compared to the Finite Element Method is that the Finite Element Method is better for computation where complex geometries are involved. The B-spline Collocation Method is suitable for use with standard geometries, like rectangles. The B-spline Collocation Method has been used in fluid flow problems with a great deal of success, but has not been used to solve Mechanics of Materials type problems.

The current research involves developing, and extensively documenting, a comprehensive, step-by-step procedure for applying the B-spline Collocation Method to the solution of Boundary Value problems. The simplicity of this approximation technique makes it an ideal candidate for computer implementation. Therefore, a symbolic Matlab code was developed, that calculates and plots everything necessary to apply this technique to a wide variety of boundary value problems. In addition, the current research involves applying the B-spline Collocation Method to solve the mathematical model that arises in the deflection of a geometrically nonlinear, cantilevered beam. The solution is then compared to a known solution found in the literature.

ACKNOWLEDGEMENTS

Above all, I would like to thank my wife, Jill, for her unwavering love, support and patience over the past 10 years of part time study. Without her encouragement and moral support, I would definitely not be a Masters of Science degree candidate.

I also would like to thank my grandmother, Harriet Schilling, for believing in me when nobody else in my life did. As a 16 year old high school drop out, I had very few options. She took me in and showed me that with hard work and determination, any goal is attainable.

I would sincerely like to thank Dr. Hany Ghoneim for his support, knowledge and extremely positive attitude. I am very grateful that he accepted my request and became my adviser for this research. Without his guidance and vision, this research project would not have turned out as well as it has.

I would also like to thank the Mechanical Engineering Department at the Rochester Institute of Technology, specifically the Department Head, Dr. Edward Hensel and the Student Information Specialist, Diane Selleck. Their support and guidance helped direct me through the seemingly endless maze of part time study.

Table of Contents

List of Figures

List of Tables

1 Introduction

Engineers are researching solutions to resolve many of today's technical challenges. Numerical techniques are used to solve the mathematical models in engineering problems. Many of the mathematical models of engineering problems are expressed in terms of partial differential equations. Two of the most popular techniques for solving partial differential equations are the Finite Element Method and the Finite Difference Method. The Finite Element Method involves dividing the domains into a finite number of sub domains called elements and placing nodes at predetermined locations around the elements boundary. The Finite Element Method finds the solution at each of the nodes very accurately. The elements, along with the nodes, form the mesh, which can be refined to provide any level of accuracy desired. An advantage of the Finite Element Method is that each element can have its own distinct geometry of varying complexities. This is extremely useful in solving complicated problems with unusual geometrical shapes or boundaries. Another one of the computational techniques used to solve mathematical models today is the Finite Difference Method. In the Finite Difference Method, a solution is derived at a finite number of points by approximating the derivatives at each of them. The accuracy of this method is based on the refinement level of the grid points where the solution is being evaluated.

In the last few years another numerical technique has been increasingly used to solve mathematical models in engineering research, the B-spline Collocation Method. A collocation method involves satisfying a differential equation to some tolerance at a finite number of points, called collocation points. The B-spline Collocation Method has a few distinct advantages over the Finite Element and Finite Difference Methods. The advantage over the Finite Difference Method is that the B-spline Collocation Method provides a piecewise-continuous, closed form solution. An advantage over the Finite Element Method is

that the B-spline Collocation Method procedure is simpler and easy to apply to many problems involving differential equations. This technique has been used in fluid flow problems with a great deal of success and more recently, some research has been conducted using the technique on computational aero acoustic [47] and biology [30] problems. However, for problems with complex geometries with curved boundaries, the Finite Element Method is still the computational technique of choice for most researchers.

This thesis explores the B-spline Collocation Method procedure in depth and lays the foundation for future research necessary to apply it to computational mechanics problems. In order to show the simplicity and accuracy of the B-spline Collocation Method, the deflection of a geometrically non-linear, cantilevered beam was calculated using the B-spline Collocation Method and compared to a known solution [10] found in the literature. To get to that point, a few steps needed to be completed.

1. The properties of B-spline curves and surfaces were thoroughly researched.
2. A thorough review of the research literature available for the B-spline Collocation Method, as well as, Geometrically Non-linear problems was conducted.
3. A comprehensive, step-by-step procedure was developed and extensively documented for applying the B-spline Collocation Method to the solution of Boundary Value problems.
4. A symbolic Matlab code was developed that can be easily extended to solve many Boundary Value problems.

The first step was to research the properties of B-spline curves and surfaces. Unfortunately, the research papers that used the B-spline Collocation Method to solve different types of problems left out many, if not all, of the details required to understand B-splines. A great

deal of information on the properties of B-spline curves and surfaces was found in the Rogers [40], deBoor [6], Farin [9] and Prautzsch [35] books.

As stated previously, a great deal of research has been done in the past decade on applying the B-spline Collocation Method to fluid flow problems. A collocation method is based on evaluating the accuracy of a differential equation at a finite set of collocation points. This fluid flow research has yielded multiple methods for the determination of the collocation points. Fairweather and Meade [8] present a summary of the spline collocation methods for boundary value problems with an outstanding compilation of references. Their summary states that there are four types of spline collocation methods used in engineering research. They are:

1. Nodal which satisfies the differential equation at each distinct knot location (nodes) on which the spline is defined.
2. Orthogonal which satisfies the differential equation by collocating at the roots of orthogonal polynomials in each subinterval of the partition for which the spline is defined.
3. Extrapolated/Modified is a modified nodal method that achieves an optimal rate of convergence on a uniform partition.
4. Collocation-Galerkin combines the advantages of spline collocation with the advantages of the Galerkin finite element analysis methods [8].

Research has been done on all four types of spline collocation methods and many variations on each of them have been developed. For instance, Botella [3] used a nodal technique that took the maximum of the B-spline basis functions as collocation points for solving the Navier-Stokes equation, but he could not account for the pressure terms. DeBoor and Swartz [7] used an orthogonal collocation approach that used the Gaussian points as collocation

points. The Gaussian points are the zeroes of the appropriate degree Legendre polynomials over the normalized knot interval. Johnson [18] compared using nodal and orthogonal collocation with nodal collocation at the Greville Abscissa points. He determined that using the Greville Abscissa was more convenient and did not result in a loss of accuracy. He also devised a method, called the Boundary Residual method [19], for including the pressure terms in the Navier-Stokes equation. Mazzia, Sestini and Trigiante [33] analyzed the convergence of multiple methods that have the collocation points coinciding with the knot points. This approach was validated by Saka and Dag [41], when they compared the B-spline results with the computed results of the Burgers' equation and modified Burgers' equation. Ramadan, El-Danaf and Alaal [36] use seventh order B-splines with collocation points uniformly distributed to solve the nonlinear Burgers Equation. Jator and Sinkala [17] devised a method where the collocation points form a strictly increasing sequence of points in the domain. Khattak and ul-Islam [26] compared the B-spline basis functions with a Radial basis function and had very similar results.

Another area where a much research has been done is mesh creation and refinement for two dimensional problems. Guzman and Morillo [13] use B-splines to define the domain of an unstructured mesh. Karim and Moser [43] defined an algorithm for two dimensional B-splines that allows the local refinement of the mesh in areas where it is required. Mesh refinement helps approximate the boundary layer of fluid flow problems by allowing an increase, or decrease if necessary, of node points in the region.

Once mesh refinement was possible, much of the work done was approximating the solution of singularly perturbed fluid flow problems. Most perturbation problems are solved by setting the small parameter to zero and converging on a solution. A singular perturbed problem cannot be solved by setting the small parameter to zero because it multiplies its

highest order differentials. This is a problem that arises in the boundary layer and therefore the boundary conditions cannot be satisfied in the governing differential equation. One of the more recent researchers, Kadalbajoo, et al, uses piecewise-uniform mesh points as collocation points. Kadalbajoo with Yadaw [20] used that collocation method to approximate a solution for a two parameter, singularly perturbed, convection-diffusion boundary value problem. With Kumar [21], Kadalbajoo used the same collocation strategy on a problem with an additional small delay factor. On another singularly perturbed convection-diffusion problem, Kadalbajoo, with coauthor Gupta [22] used that same piecewise-uniform mesh collocation strategy. Kadalbajoo and Arora [23] approximate the solution to a singularly perturbed problem by redefining the problem to include artificial viscosity, which is determined later. In order to solve a time dependant problem, Kadalbajoo, Awasthi and Gupta [24] use the piecewise-uniform collocation strategy on a singularly perturbed, linear convection-diffusion problem. Kadalbajoo and Aggaewal [25] approximate the solution of a self adjoint, singularly perturbed boundary value problem. To solve a problem that arises in biology, Lin, Li and Cheng [30] use B-splines to solve a singularly perturbed boundary value problem, with the use of wavelet functions. Rao and Kumar [37] used an optimization technique to select collocation points on a self adjoint, singularly perturbed boundary value problem. They compared a third order with two intermediate points with a solution from a paper using a seventh order smooth B-spline. Using a nodal collocation scheme, Rao and Kumar [38] also approximated the solution of a self adjoint, singularly perturbed boundary value problem in the space of an exponential B-spline.

Fluid flow problems are not the only areas where research is being conducted. For instance, Widjaja, Ooi, Chen and Manasseh [47] apply the B-spline collocation method to spinning co-rotating vortices. Their B-spline collocation method allows mesh points to be

placed arbitrarily, which allows them to avoid using mapping functions currently used in Computational Aero-Acoustics. Ghoneim [11] applies the B-spline collocation method with the smoothest open-uniform knot vector, orthogonal collocation points and Greville Abscissa for the control points to get the solution for the axial stretching of a composite hyperbolic structure. Ghoneim and Santos [12] applies the same method to approximate the axial stretching of a composite barrel shaped shell structure. Hou, Yin and Wang [14] applied the B-spline collocation method, using equal distance nodes as collocation points, to a variable thickness conical shell. Wu, Chung and Huang [49] use a Radial Spline Collocation method that does not require the collocation points to be equally spaced. This increases the accuracy when there are discontinuities in the load distribution, geometry or material properties of the beam. Sun [44] uses orthogonal, cubic, B-spline collocation for solving the equations of linear elasticity

Most of the work that had been done on solving non-linear mechanics problems was conducted earlier last century. Von Karman [46] provides a nice summary of the techniques applied to solve large deflection problems before 1940. Bisshopp and Drucker [2] solve the large deflection cantilevered beam problem with elliptical integrals. It has become an increasingly popular area of research recently with alternative approaches to solving these problems being developed. Howell and Midha [15] develop a parametric approximation for the large deflection of a cantilevered beam by constructing a pseudo-rigid body that assumes the beam will follow a near circular path. Luo [31] developed an approximation theory for geometrically non-linear thin plates by using the physical strains and equilibrium equations of the plate and measuring the exact geometry of the deformed middle surface. This, according to them, eliminated the requirements of the constitutive laws, because the physical strains were used.

One of the contributions of this thesis was to thoroughly document the B-spline Collocation Method in a very simple and easy to follow format. The example problems were solved in a step by step manner that any individual studying engineering or mathematics can follow and use the same technique to solve additional problems. The other contribution of this thesis was to apply the B-spline Collocation Method to solve a geometrically nonlinear beam problem. This will eliminate the need for the use of the elliptical integrals, which can be difficult to work with, in the solution of the large deflection problem. The work of Johnson [18], utilizing the Greville Abscissa as collocation points, is expanded upon and shown to be a far simpler approach than the elliptical integrals approach.

In Chapter 2 of this thesis, the properties and components that make up the B-spline functions, as well as their derivatives, is discussed in great detail. In Chapter 3, a comprehensive, step-by-step procedure is developed and applied to the solutions of one dimensional, under damped and over damped boundary value problems with Dirichlet Boundary Conditions. These problems are worked out with intermediate knot points inserted, as well as, the smoothest, higher order knot vectors. In addition, a fourth order, linear Euler-Bernoulli beam problem is solved with both Dirichlet and Neumann boundary conditions and compared to the known solution. This example is very important because it shows, step-by-step, how to solve a higher order boundary value problem, with multiple types of boundary conditions. At the end of Chapter 3 is an in-depth discussion on the accuracy of this B-spline Collocation Method. In Chapter 4, the step-by-step procedure for the one dimensional B-spline function is applied to two dimensional heat transfer problems. Once again, problems with both Dirichlet and Neumann boundary conditions are solved and compared to solutions from ANSYS Finite Element Analysis software. Finally, the geometrically nonlinear beam problem is solved in Chapter 5. The solution is compared to a previously documented

solution by Frisch-Fay [10], in Chapter 2 of his book ‘Flexible Bars’. In his solution, Frisch-Fay goes through a very complex process to reach his solution. The B-spline solution developed here is found to be very accurate and far simpler to solve.

The simplicity of this approximation technique makes it an ideal candidate for computer implementation. In the Appendix of this thesis is a symbolic Matlab code for every example problem shown. The Matlab code, with a few modifications and inputs, calculates and plots everything necessary to implement this technique on a wide variety of boundary value problems. With the research documented on the B-spline Collocation Method and the development of the symbolic Matlab code, a solid foundation for future research has been laid.

2 B-spline Background

2.1 Introduction

A spline is a continuous piecewise curve used to approximate a solution to a mathematical problem. A spline curve is dependant upon a relationship between the basis function and the vertices of a defining polygon. The B-spline curve has its own type of basis function, known as the B-spline basis, to establish the relationship with the defining polygon. That relationship for the B-spline curve, P (t), is defined by the equation $P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$, where B_i are the position vectors of the n + 1 defining polygon vertices and $N_{i,k}(t)$ are the normalized basis functions. The order 'k' of the basis function results in a polynomial of degree, n, where n = k - 1. This equation will be discussed in detail later in the chapter. One property of the B-spline curve is that it must lie entirely within the convex hull of the defining polygon, as shown in Figure 1.

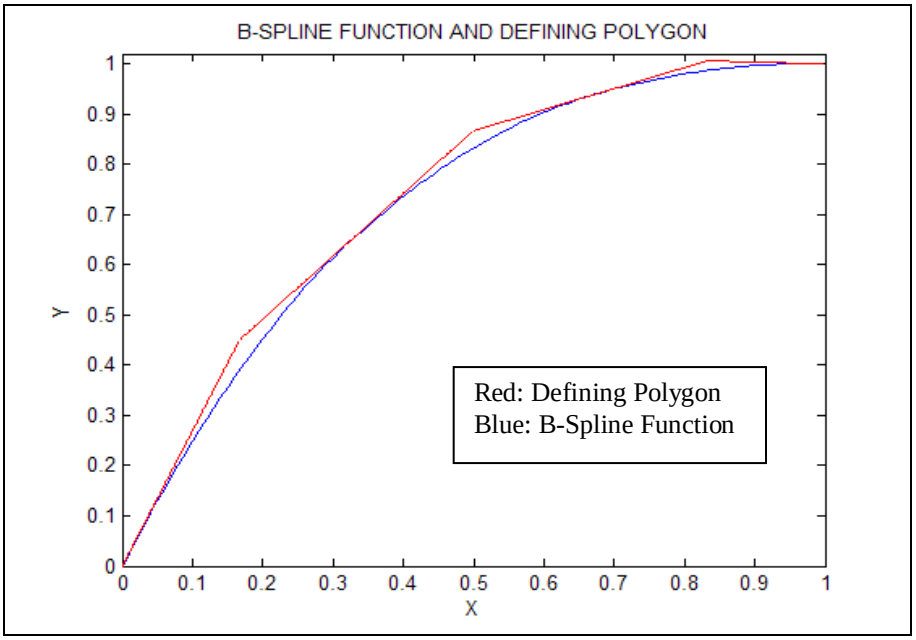

Figure 1 Defining Polygon with Five Vertices

Other types of splines are used in the mathematical and engineering fields. Another popular type of spline, the Bezier spline, uses the Bernstein basis function to establish the relationship. However, the B-spline basis function has a few advantages over the Bernstein basis function. One advantage is that the order (k) of the basis function can be reduced, resulting in a curve with a lower degree (k-1), without affecting the approximations accuracy. This can be accomplished by changing the number of vertices in the defining polygon with the addition of intermediate points to the knot vector. Another advantage is that the B-spline curves behave in a non-global manner, with each vertex, B_i, associated to a unique basis function. This means that each vertex affects only the shape of the curve over a range of values where the basis function has a value greater than zero. That behavior is helpful when a local refinement to the function is necessary.

Some very important properties of the B-spline curve that will help solve and troubleshoot problems are:

- P(t) is a polynomial of degree $k - 1$ on each interval $x_i \le t < x_{i+1}$
- P(t) and its derivatives of order 1,2 ,…,k – 2 are all continuous over the entire curve
- The sum of the B-spline basis functions for any parameter value 't' can be shown to be $\sum_{i=1}^{n+1} N_{i,k}(t) \equiv 1$
- Each basis function is positive or zero for all parameter values
- Except for k=1 each basis function has one maximum value
- The maximum order in the smoothest of the curves is equal to the number of defining polygon vertices
- The curve does not oscillate about any straight line more often than its defining polygon

- The curve generally follows the shape of the defining polygon
- The curve is transformed by transforming the defining polygon vertices
- The curve lies within the convex hull of its defining polygon
- C^{k-2} continuous at each transition point, called knot points, and C^{k-1} continuous elsewhere

2.2 B-spline Curve Components

In order to calculate B-spline curves three things are required:

1. Knot Vectors.
2. Basis Functions.
3. Control Points.

Once the knot vector is selected and the basis functions and control points are calculated, the B-spline curve can be derived. The resulting B-spline curve, P(t) approximates the solution to a predefined boundary value problem that can be locally controlled by the location of the vertices of the defining polygon. The B-spline curve is given by the equation:

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t) \qquad t_{\min} \le t < t_{\max}, \ 2 \le k \le n+1 \tag{1}$$

where B_i are the position vectors of the n + 1 defining polygon vertices. In the B-spline curve calculation, the basis functions that are used are only the ones that represent the highest order, k. A typical expansion of a third order, (k = 3) B-spline curve would look like this:

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$$

$$P(t) = B_1 N_{1,3}(t) + B_2 N_{2,3}(t) + B_3 N_{3,3}(t)$$

2.2.1 Knot Vectors

The selection of the knot vectors is very important to the creation of the B-spline. Their selection determines the order of the resulting polynomial and the size of intervals that each B-spline basis function represents. In addition, the smoothness of the curve is determined by the choice of the knot vector. The knot vector chosen must provide sufficient resolution to approximate the solution of the mathematical problem.

There are 3 different types of knot vectors used. They are:

1. Uniform: Individual knot values are evenly spaced.
2. Open-Uniform: Multiplicity of knot values at ends equal to the order of the B-spline curve and internal knot values are evenly spaced.
3. Non-Uniform: Unequally spaced and/or multiple internal knot values.

In this thesis, open-uniform knot vectors are used exclusively to calculate B-spline curves and basis functions. The number of control points and basis functions when the open-uniform knot vector is used is equal to (k + m), where ‘k’ is the order and ‘m’ is the number of internal points in the knot vector. When there are no internal knot points, the B-spline basis reduces to the Bernstein basis, rendering the smoothest curve. In this special case, the resulting B-spline curve is a Bezier curve. Some examples with the corresponding order (k) and number of internal points in the knot vector (m) are;

[0 0 0 1 1 1]	k = 3, m = 0
[0 0 0 1 2 2 2]	k = 3, m = 1
[0 0 0 0 1 1 1 1]	k = 4, m = 0
[0 0 0 0 1 2 3 3 3 3]	k = 4, m = 2

In order to make the algorithm simpler and faster to solve, they also can be normalized, prior to the calculation of the basis functions. To do this, divide all the

components of the vector by the largest value ‘h’ in the knot vector. However, note that this normalization will have to be accounted for in the collocation process.

Example: [0 0 0 1 2 2 2] h = 2, k = 3

[0 0 0 ½ 1 1 1] Normalized

2.2.2 Basis Functions

Once the knot vector has been selected, the B-spline basis function can be calculated. The B-spline basis function $N_{i,k}(t)$ is defined by a recursive relationship that is known as the Cox-deBoor recursion formula.

$$N_{i,1}(t) = \begin{cases} 1 \; x_i \le t < x_{i+1} \\ 0 \;\; otherwise \end{cases} \tag{2}$$

$$N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}} \tag{3}$$

Where ‘i’ is a counter and ‘k’ is the order of the basis function. The values of x_i are elements of the predetermined knot vector that satisfies the relationship $x_i \le x_{i+1}$. The basis functions and the knot vector are defined in terms of the parameter ‘t’, which is a coordinate in parametric space and may or may not coincide with any of the Cartesian coordinates. The parameter *t* will vary from t_{min} to t_{max} along the B-spline curve. Also during calculation, the convention of 0/0 = 0 must be adopted for the recursion to work. Their calculation is completely driven by the predetermined knot vector. The order (k) of the basis functions corresponds to an approximating B-spline polynomial of (k-1) degrees.

Here is an example for the following knot vector [0 0 0 1 1 1], with k = 3, using the Cox-deBoor recursion relationship in equations (1) and (2):

$$N_{i,1}(t) = \begin{cases} 1 \; x_i \le t < x_{i+1} \\ 0 \;\; otherwise \end{cases} \qquad N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

$N_{1,1}(t) = 0;$ $N_{2,1}(t) = 0;$ $N_{3,1}(t) = 1;$ $N_{4,1}(t) = 0;$

$N_{1,2}(t) = 0;$ $N_{2,2}(t) = 1 - t;$ $N_{3,2}(t) = t;$ $N_{4,2}(t) = 0;$

$N_{1,3}(t) = (1-t)^2;$ $N_{2,3}(t) = 2t(1-t);$ $N_{3,3}(t) = t^2;$ $N_{4,3}(t) = 0$

By plotting all three of the k = 3 basis functions, which are shown in Figure 2, a great deal of information can be derived. Notice that at any value of t, the sum of all of the basis functions is equal to a value of one. Also notice the degree of the basis function curves. The curves of these basis functions are 2nd degree, which corresponds to the 3rd order B-spline. Another property to observe is that the first basis function, $N_{1,k}(t)$, always starts at a value of one and ends at a value of zero, for the functions with no intermediate points in the knot vector. Conversely, the last basis function, $N_{i,k}(t)$, starts at a value of zero and ends at one. The intermediate basis function starts and ends at values of 0, respectively.

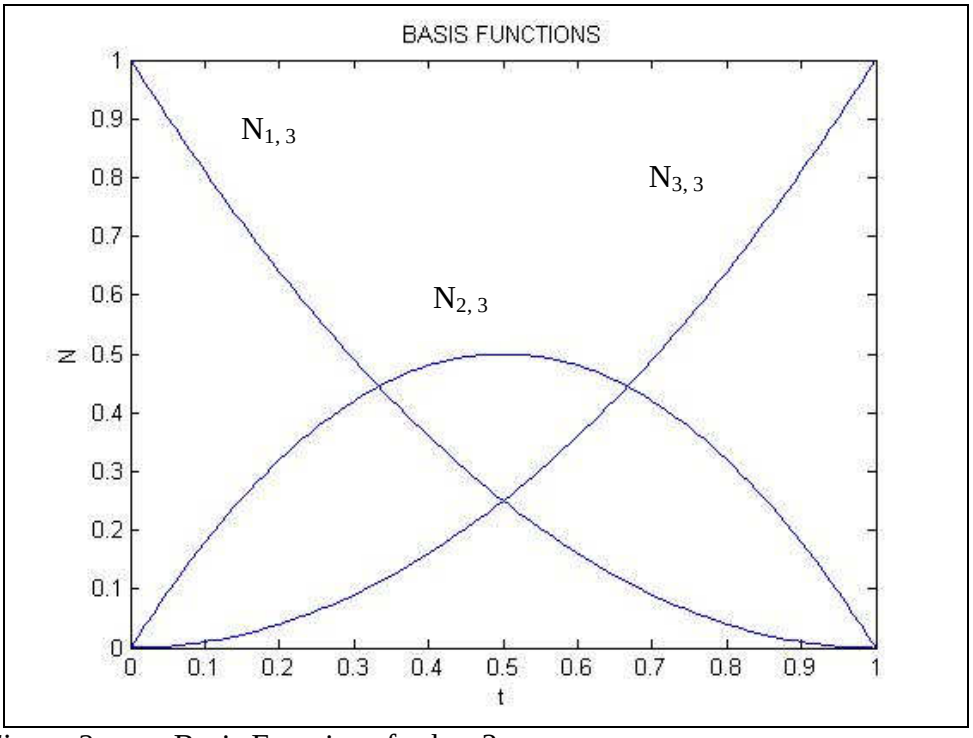

Figure 2 Basis Functions for k = 3

Here is an example for the following knot vector [0 0 0 0 1 1 1 1] , with k = 4, using the Cox-deBoor recursion relationship in equations (1) and (2):

$N_{1,1}(t) = 0;\quad N_{2,1}(t) = 0;\quad N_{3,1}(t) = 0;\quad N_{4,1}(t) = 1;\quad N_{5,1}(t) = 0;$

$N_{1,2}(t) = 0;\quad N_{2,2}(t) = 0;\quad N_{3,2}(t) = 1-t;\quad N_{4,2}(t) = t;\quad N_{5,2}(t) = 0;$

$N_{1,3}(t) = 0;\quad N_{2,3}(t) = (1-t)^2;\quad N_{3,3}(t) = -2t(t-1);\quad N_{4,3}(t) = t^2;\quad N_{5,3}(t) = 0;$

$N_{1,4}(t) = (1-t)^3;\quad N_{2,4}(t) = 3t(t-1)^2;\quad N_{3,4}(t) = -3t^2(t-1);\quad N_{4,4}(t) = t^3;\quad N_{5,4}(t) = 0$

By plotting all four of the k = 4 basis functions, which are shown in Figure 3, all of the same information and properties from the previous k = 3 example still apply. Notice now that both of the intermediate functions start and end at values of 0. Also notice the curves of these basis functions are 3rd degree, which corresponds to the 4th order B-spline.

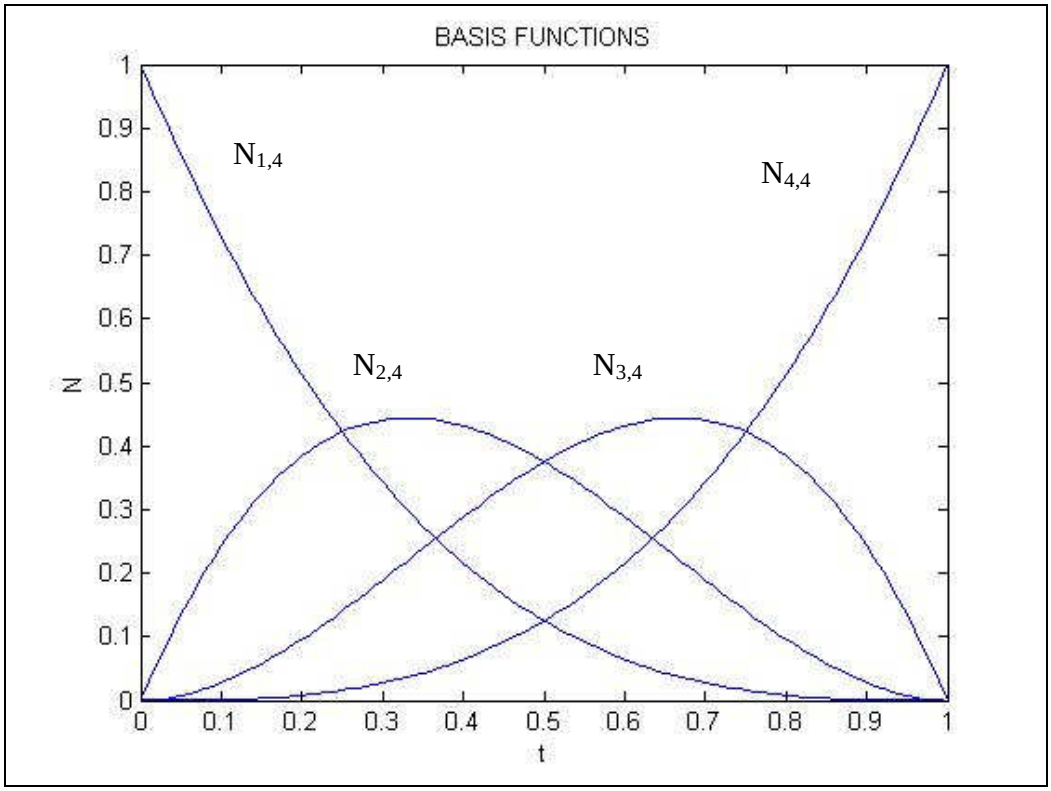

Figure 3 Basis Functions for k = 4

In order to show the differences of the basis functions when intermediate points are put into the knot vector, here is an example calculating the basis functions for the following knot vector [0 0 0 ½ 1 1 1], with k = 3. The basis functions are broken up into two equal intervals because it is an open-uniform knot vector. The Cox-deBoor recursion relationship in equations (1) and (2), now should be calculated twice, once for each interval.

For the first interval, $0 \le t < \frac{1}{2}$

$N_{1,1}(t) = 0;$ $N_{2,1}(t) = 0;$ $N_{3,1}(t) = 1;$ $N_{4,1}(t) = 0;$

$N_{1,2}(t) = 0;$ $N_{2,2}(t) = 1 - 2t;$ $N_{3,2}(t) = 2t;$ $N_{4,2}(t) = 0;$

$N_{1,3}(t) = (1-2t)^2;$ $N_{2,3}(t) = 2t(2-3t);$ $N_{3,3}(t) = 2t^2;$ $N_{4,3}(t) = 0$

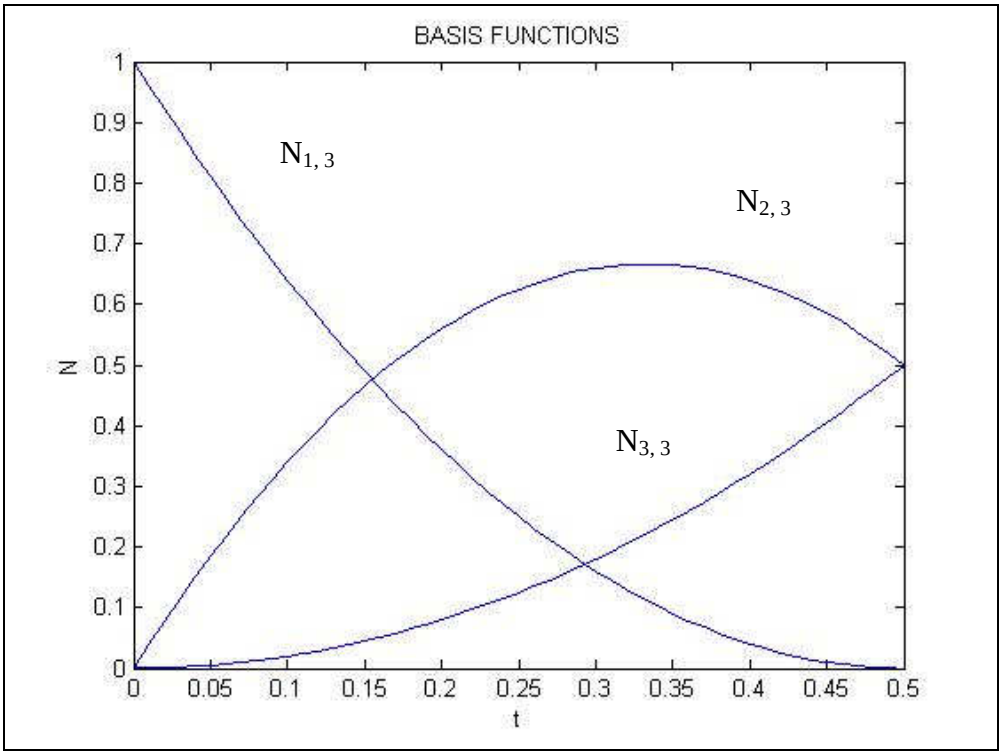

Figure 4 Basis Functions for the Interval: $0 \le t < \frac{1}{2}$

The plots of the first interval basis functions, in Figure 4, show some different characteristics from the plots with no intermediate points. Notice now that the second and third functions end at the intermediate knot vector point value.

For the second interval, $\frac{1}{2} \le t < 1$

$N_{1,1}(t) = 0$; $N_{2,1}(t) = 0$; $N_{3,1}(t) = 0$; $N_{4,1}(t) = 1$; $N_{5,1}(t) = 0$;

$N_{1,2}(t) = 0$; $N_{2,2}(t) = 0$; $N_{3,2}(t) = 2 - 2t$; $N_{4,2}(t) = 2t - 1$; $N_{5,2}(t) = 0$;

$N_{1,3}(t) = 0$; $N_{2,3}(t) = 2(t-1)^2$; $N_{3,3}(t) = -6t^2 + 8t - 2$; $N_{4,3}(t) = (1-2t)^2$; $N_{5,3}(t) = 0$

The plots of the second interval basis functions are shown in Figure 5. Notice now that the second and third basis functions start at the intermediate knot vector point value and end at a value of zero. Also notice that the first basis function is gone in this interval and a fourth one was added. As was the case earlier for the last basis functions, its values start at zero and end at one.

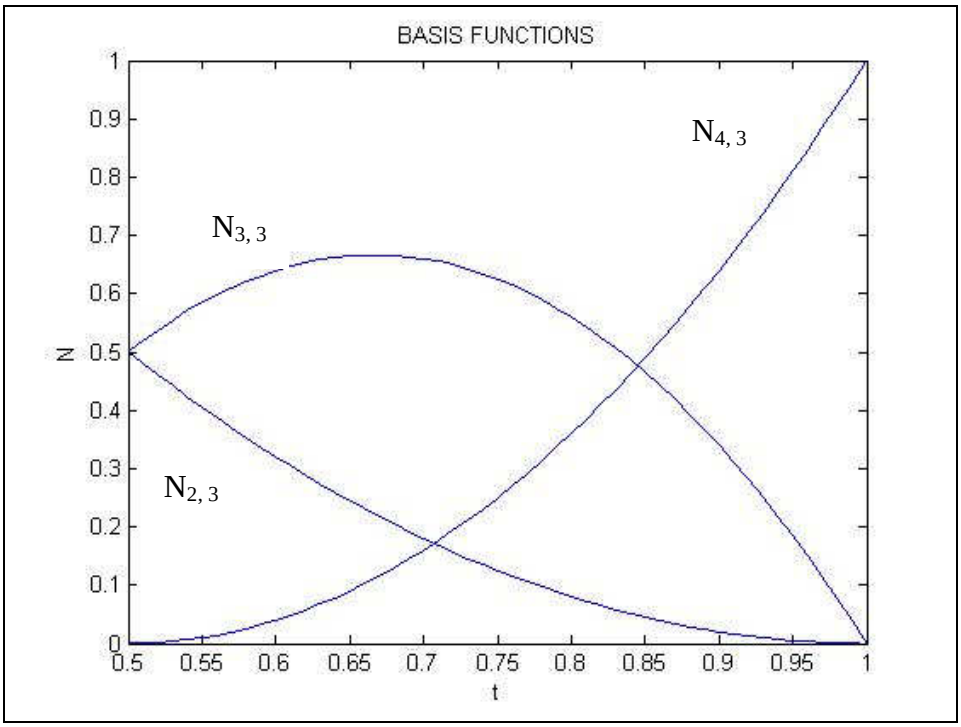

Figure 5 Basis Functions for the Interval: $\frac{1}{2} \le t < 1$

By combining the graphs of the two intervals, as seen in Figure 6, the main advantage of using intermediate points in the knot vector can be seen. All of the second degree basis functions join together at the interval line and form pseudo third degree basis functions. The

joined together functions increase the overall accuracy, an advantage typical of a higher order function, while not increasing the order of the basis functions.

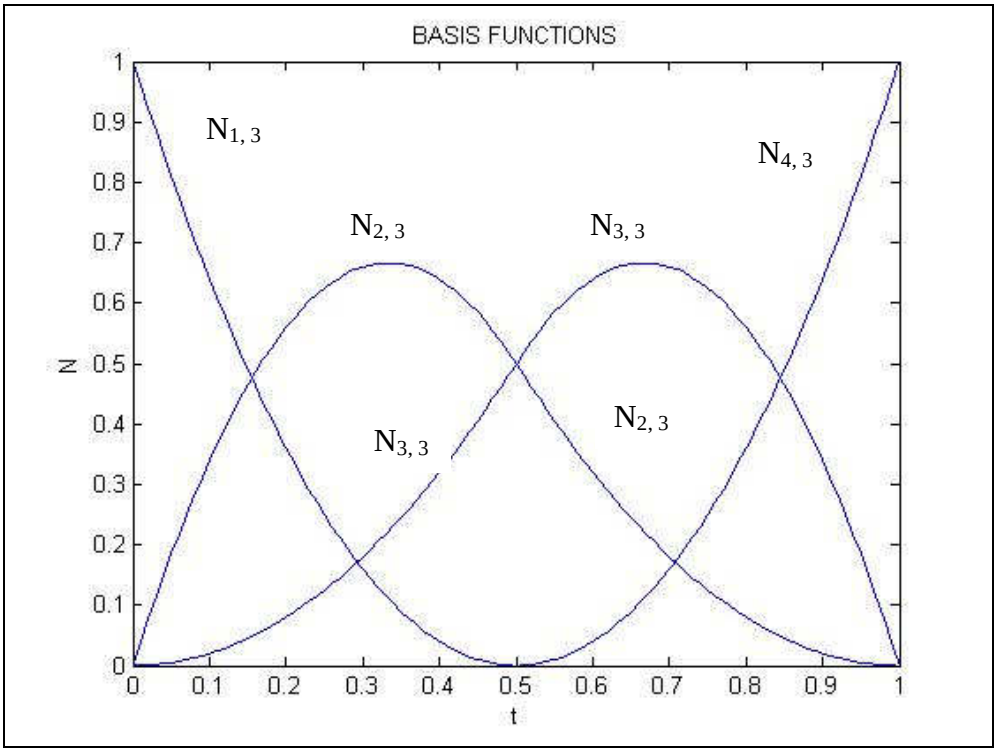

Figure 6 Basis Functions for the Interval: $0 \le t < 1$

Generally, the higher the order of the basis functions, the longer the computation time associated with the problem. In addition to the increased time, the higher order also makes the solution of under damped problems more susceptible to spurious oscillations. Therefore, the addition of intermediate points in the knot vector can help by dividing the basis functions into smaller piecewise segments. The smaller piecewise segments have the beneficial effect of increasing the resolution of the approximation without increasing the order of the functions.

2.2.3 Control Points

Once the knot vectors and basis functions have been determined, the last remaining element needed to create the B-spline curve is the position vector, B_i. The coordinates that make up these vectors are referred to as control points and they locate the vertices of the

defining polygon. The control points are selected in order to yield an accurate approximation to the desired curve.

The method used in this thesis to locate the abscissa control points is the Greville Abscissa approach. If the abscissa values of the control points are restricted to lie at the Greville Abscissa for the B-spline curve, the parametric coordinate 't' is constrained to the 'x' Cartesian coordinate. The formula for calculating these coordinates is:

$$x_i = \frac{1}{n}(t_i + t_{i+1} + ... + t_{i+n-1}) \qquad \text{For i=0, 1,…,g-n} \qquad (4)$$

where 'n' is the degree of the basis functions, or k – 1 and 'g' is the total number of knots in the knot vector.

Example: [0 0 0 1 1 1] k = 3; n = 2; m = 6

$$x_0 = \frac{1}{2}(t_0 + t_1) = \frac{1}{2}(0+0) = 0$$

$$x_1 = \frac{1}{2}(t_1 + t_2) = \frac{1}{2}(0+0) = 0$$

$$x_2 = \frac{1}{2}(t_2 + t_3) = \frac{1}{2}(0+1) = \frac{1}{2}$$

$$x_3 = \frac{1}{2}(t_3 + t_4) = \frac{1}{2}(1+1) = 1$$

$$x_4 = \frac{1}{2}(t_4 + t_5) = \frac{1}{2}(1+1) = 1$$

If calculated properly, there will always be two repeating values at the ends, one of which can be dropped for the purposes of the B-spline curve derivation. In this example, x_0 and x_4 will be dropped and x_1, x_2 and x_3 will be used to calculate the B-spline curve.

2.3 Derivatives

Once the computation of the knot vectors, basis functions and control points is completed, the B-spline curves can be calculated. Since computing the B-spline curve to approximate the solution of a boundary value problem is the goal of this technique, calculating the derivatives of the B-spline curve is necessary. The equations for the first two derivatives, with respect to the parameter 't' are:

$$P'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t) \tag{5}$$

$$P''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t) \tag{6}$$

Also notice once again, that the basis functions that are used are only the ones that represent the highest order, k. When using B-spline curves and their derivatives to approximate the solution to Boundary Value Problems, typically the boundary conditions are with respect to Cartesian coordinates. Since the B-spline curves are functions of the parametric coordinate system and B-spline derivatives are with respect to parametric coordinates, they must be converted to the Cartesian coordinate system. In addition, the normalizing of the knot vector done earlier must also be accounted for. For the first order, the derivative of the basis function with respect to 'x' is:

$$\frac{\partial N_{i,k}(t)}{\partial x} = \frac{\partial N_{i,k}(t)}{\partial t} * \left[\frac{\partial t}{\partial x}\right] \tag{7}$$

It was stated earlier that when the Greville Abscissa are used to compute the control points the Cartesian coordinate 'x' is constrained to the parametric coordinate 't'. That is, x = h * t, or $\left[\frac{\partial t}{\partial x}\right] = \left[\frac{1}{h}\right]$, where 'h' is the largest value of the original knot vector. By substituting this

into equation (7), the expression for the first derivative of the basis function with respect to the Cartesian coordinate system becomes:

$$\dot{N}_{i,k} = \frac{\partial N_{i,k}}{\partial x} = \frac{\partial N_{i,k}(t)}{\partial t} * \left[\frac{1}{h}\right] \tag{8}$$

Consequently the first derivative of the B-spline function with respect to the Cartesian coordinate system then becomes:

$$\dot{P}(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t) \left[\frac{1}{h}\right] \tag{9}$$

Similarly, the second derivative of the basis function can be shown to be:

$$\ddot{N}_{i,k} = \left(\frac{\partial^2 N_{i,k}(t)}{\partial t^2}\right)\left(\frac{1}{h}\right)^2 \tag{10}$$

The second derivative of the B-spline function with respect to the Cartesian coordinate system then becomes:

$$\ddot{P}(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t) \left[\frac{1}{h}\right]^2 \tag{11}$$

In general, it can be shown that:

$$\overset{(a)}{N}_{i,k} = \left(\frac{\partial^a N_{i,k}(t)}{\partial t^a}\right)\left(\frac{1}{h}\right)^a \tag{12}$$

$$\ddot{P}(t) = \sum_{i=1}^{n+1} B_i N^a_{i,k}(t) \left[\frac{1}{h}\right]^a \tag{13}$$

where $\overset{(a)}{N}_{i,k}$ is the 'a'th derivative with respect to Cartesian coordinate'x' and $N^a_{i,k}(t)$ is the 'a'th derivative with respect to the parameter't'.

3 One Dimensional B-spline Collocation

The B-spline collocation method involves the determination of knot vectors and control points such that the differential equation, or boundary value problem, is satisfied to some tolerance at a finite set of collocation points. These collocation points reside on the Cartesian abscissa axis. The selection of these points appears to be arbitrary [18] and throughout the literature there are many different ways used to select these collocation points. As discussed previously, the four main types of spline collocation are the Nodal, Orthogonal, Extrapolated/Modified and Collocation-Galerkin approaches. There have been many variations of these four types used in the literature to solve mathematical models.

For this thesis, a variation on the Nodal type, called the Greville Abscissa approach [18-19], has been selected and the collocation points are located at the Greville Abscissa points along the Cartesian abscissa axis. Since these points have already been calculated and the selection of the collocation points is arbitrary, the Greville Abscissa points are a logical selection for the collocation points. Additional benefits of using this approach were previously discussed when calculating the derivatives of the B-spline curves.

Once the collocation method has been selected, the one dimensional B-spline collocation method to approximate a differential equation or boundary value problem solution involves following a few steps.

1. Select the appropriate knot vector
2. Calculate all required basis functions, using equation (2) and (3)
3. Calculate abscissa coordinates for the required control points, using the Greville Abscissa equation (4)
4. Calculate the B-spline curve equations and the required B-spline curve derivatives

5. If necessary, use the boundary conditions to solve for end ordinate values of control points
6. Substitute B-spline curves and the derivatives into differential equation or boundary value problem
7. Calculate the remaining interior ordinate coordinates of the control points by evaluation of the differential equation at the corresponding abscissa of the collocation points

In order to quantify the approximation results, the error has been calculated at each collocation point using the relative error formula:

$$ERR\% = \left[1 - \left|\frac{BSpline}{Exact}\right|\right] * 100 \tag{14}$$

A statistical sampling method has been employed in order to quantify a baseline value for the overall approximation procedure. The statistical method chosen for use is the Root Mean Squares method. This technique is used throughout industry for tolerance analysis. The formula for the Root Mean Squares method is:

$$RMS = \sqrt{\frac{1}{N} * \sum_{i=1}^{N} \left(ERR\%\right)_i^2} \tag{15}$$

A baseline value for a successful approximation has been set at RMS < 0.5% for the entirety of this research. One of the objectives of this research is to compare the speed and accuracy of the B-spline Collocation Method with two different techniques. The first is by increasing the number of internal points in the knot vector and the second is by increasing the approximation to a higher order. In order to accomplish this, the B-spline Collocation Method solution is compared to analytical solutions that can be easily computed. If an analytical situation is not available, selecting convergence criteria would be more applicable when calculating the solution at the collocation points. To ensure an even higher level of accuracy for the entire approximation, the residuals can be calculated and compared until

their values converge. The residuals are the solution errors at points in addition to the collocation points. If an even higher level of accuracy is required, increase the density of the evaluating points that are calculated.

Three types of one-dimensional problems were evaluated for this thesis in order to calculate the effectiveness of the B-Spline Collocation Method. The approximations of both over-damped and un-damped, second order boundary value problems, as well as, a fourth order, Euler-Bernoulli beam problem have been evaluated. Please note that to get to the solution of each problem, one must go through the same 7 steps mentioned previously.

3.1 Example 1A

Over Damped, Second Order ODE Approximated with a Third Order B-spline, with No Intermediate Knot Vector Points

P'' + 2P' + P = 0 For $0 < x < 1$ Where P' = $\frac{\partial}{\partial x}$

Boundary Conditions: P(0) = 0, P(1) = 1

1. The first step in the B-spline collocation method is to select the appropriate knot vector that provides the resolution required for the problem. A third order, open-uniform knot vector has been selected.

   [0 0 0 1 1 1]

2. The second step is to calculate all required basis functions using equations (2) and(3).

$$N_{i,1}(t) = \begin{cases} 1 & x_i \le t < x_{i+1} \\ 0 & otherwise \end{cases} \qquad N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

$$N_{1,1}(t) = 0; \quad N_{2,1}(t) = 0; \quad N_{3,1}(t) = 1; \quad N_{4,1}(t) = 0;$$

$$N_{1,2}(t) = 0; \quad N_{2,2}(t) = 1 - t; \quad N_{3,2}(t) = t; \quad N_{4,2}(t) = 0;$$

$$N_{1,3}(t) = (1-t)^2; \quad N_{2,3}(t) = 2t(1-t); \quad N_{3,3}(t) = t^2; \quad N_{4,3}(t) = 0$$

By plotting the k = 3 basis functions, the generated curves can be used to verify that the calculated basis functions comply with the criteria discussed in the previous chapter. In Figure 7, it can be seen that indeed they do meet the required criteria and it is safe to move on to the next step.

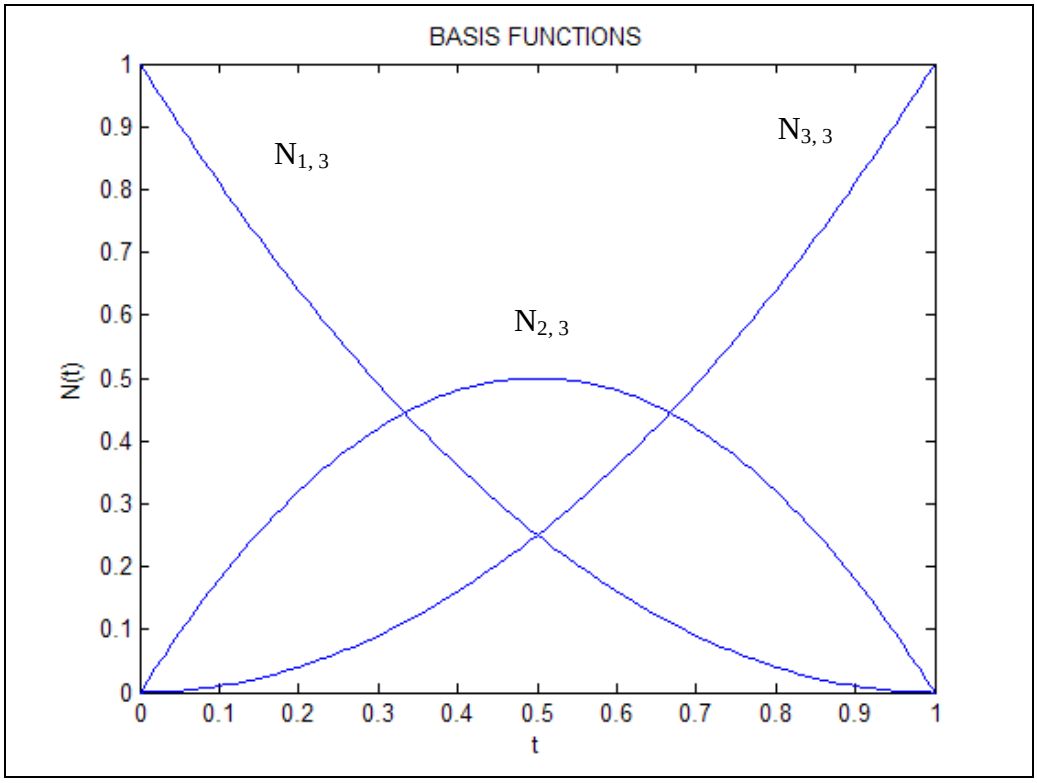

Figure 7 Basis functions for third order B-spline approximation

3. The next step is to calculate the abscissa coordinates for the control points. Since the Greville Abscissa approach has been chosen, use equation (4) to calculate the Greville Abscissa for required control points.

$$x_i = \frac{1}{n}(t_i + t_{i+1} + ... + t_{i+n-1}) \qquad \text{For i=0, 1,…,m-n}$$

$$x_0 = \frac{1}{2}(t_0 + t_1) = \frac{1}{2}(0+0) = 0$$

$$x_1 = \frac{1}{2}(t_1 + t_2) = \frac{1}{2}(0+0) = 0$$

$$x_2 = \frac{1}{2}(t_2 + t_3) = \frac{1}{2}(0+1) = \frac{1}{2}$$

$$x_3 = \frac{1}{2}(t_3 + t_4) = \frac{1}{2}(1+1) = 1$$

$$x_4 = \frac{1}{2}(t_4 + t_5) = \frac{1}{2}(1+1) = 1$$

As discussed in the previous chapter, the repeating values of $x_0 = 0$ and $x_4 = 1$ should be dropped. That leaves $x_1 = 0$, $x_2 = \frac{1}{2}$ and $x_3 = 1$ as the abscissa coordinates for the control points.

4. The fourth step is to calculate the appropriate B-spline curve equations and the required derivatives for the boundary conditions that need to be met. Since the third term of the differential equation and both boundary conditions involve the B-spline curve, it must be calculated. The B-spline curve must be calculated, by using equation (1).

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$$

$$P(t) = B_1 N_{1,3}(t) + B_2 N_{2,3}(t) + B_3 N_{3,3}(t)$$

$$\boxed{P(t) = B_1\left[(1-t)^2\right] + B_2\left[2t(1-t)\right] + B_3\left[t^2\right]}$$

Since the second term of the differential equation involves the first derivative, the first derivative of the B-spline equation must be calculated, using equation (5).

$$P'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t)$$

$$P'(t) = B_1 N'_{1,3}(t) + B_2 N'_{2,3}(t) + B_3 N'_{3,3}(t)$$

$$\boxed{P'(t) = B_1\left[2(t-1)\right] + B_2\left[(2-4t)\right] + B_3\left[2t\right]}$$

The first term of the differential equation involves the second derivative. The second derivative of the B-spline curve can be calculated using equation (6).

$$P''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t)$$

$$P''(t) = B_1 N''_{1,3}(t) + B_2 N''_{2,3}(t) + B_3 N''_{3,3}(t)$$

$$\boxed{P''(t) = B_1[2] + B_2[(-4)] + B_3[2]}$$

5. The next step is to use the boundary conditions to solve for end ordinate values of control points. Since the Greville Abscissa were used as control points, the parametric coordinate 't' is constrained to the Cartesian coordinate 'x' and it becomes a direct substitution. The boundary conditions were P(0) = 0 and P(1) = 1. When the B-spline curve, $P(t) = B_1[(1-t)^2] + B_2[2t(1-t)] + B_3[t^2]$, is evaluated at the boundaries, the following external ordinate values are calculated to be:

   P(0) = 0➔B_1 = 0➔ $y_1 = 0$

   P(1) = 1➔ B_3 = 1➔ $y_3 = 1$

6. The sixth step is to substitute B-spline curve and the derivatives into the differential equation, P'' + 2P' + P = 0, we get

$$\{B_1[2] + B_2[(-4)] + B_3[2]\} + 2\{B_1[2(t-1)] + B_2[(2-4t)] + B_3[2t]\} + \{B_1[(1-t)^2] + B_2[2t(1-t)] + B_3[t^2]\} = 0$$

   Now using the ordinate values from the previous step, the differential equation reduces to: $\{B_2[(-4)] + [2]\} + 2\{B_2[(2-4t)] + [2t]\} + \{B_2[2t(1-t)] + [t^2]\} = 0$

7. The seventh step is to calculate the remaining internal ordinate coordinates of the control points by evaluation of the previously reduced differential equation, $\{B_2[(-4)] + [2]\} + 2\{B_2[(2-4t)] + [2t]\} + \{B_2[2t(1-t)] + [t^2]\} = 0$, at the corresponding abscissa coordinates. At't' = ½, the remaining internal ordinate coordinate is calculated to be:

   $B_2 = y_2 = 0.82353$

Now that all of the coordinates for the control points have been calculated, they can be substituted into the B-spline curve equation, $P(t) = B_1\left[(1-t)^2\right] + B_2\left[2t(1-t)\right] + B_3\left[t^2\right]$ to give the final equation for the B-spline approximation. Since the B-spline curve is in terms of the parametric coordinate system and the desired result is an equation in the Cartesian coordinate system, a bit of manipulation is required. As stated earlier, using the Greville Abscissa allows a direct substitution of the 'x' and't' parameters. The B-spline function approximation equation then becomes $\boxed{P(x) = 0.82353\left[2x(1-x)\right] + \left[x^2\right]}$.

Now a comparison of the B-spline function approximation and the exact solution, P(x) = $2.718 * x * e^{-x}$, must be done to determine the accuracy of the collocation method. The first step to determining the accuracy is to evaluate both the B-spline function approximation and the exact solution at the collocation points. Next, calculate the relative error percentage, by using equation (14) for each collocation point. These values have been listed in Table 1.

Collocation Point Evaluation			
Point	Approximate	Exact	%Error
0	0	0	0
0.5	0.85714	0.82428	3.99
1	1	0.9999	0.001

Table 1 Table of collocation points for Example 1a

Once all of the relative error percentage calculations have been done, the Root Mean Squared Error must be computed, by using equation (15).

RMS Error = 2.3%

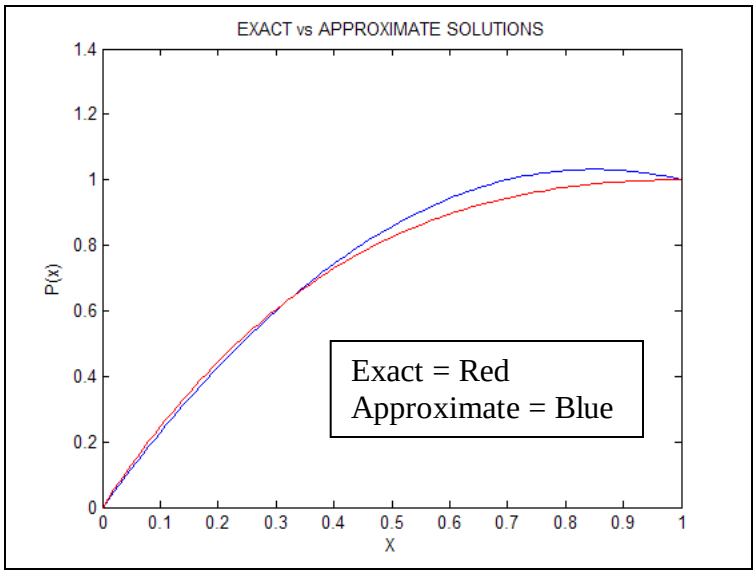

Figure 8 B-spline solution plotted with the exact solution

The RMS Error of 2.3% is greater than the 0.5% baseline, previously established. By plotting the two functions for the entire range, as seen in Figure 8, the areas where refinement is necessary can be determined. One method of refinement is to increase the order of the B-spline Basis Functions. Another method is to add a collocation point within the range where the highest error percentage is. By a visual inspection of Figure 8, it can be seen that the largest difference between the two functions is within the range of $0.6 < x < 0.8$. The next example (1b) shows what will happen when two additional collocation points are added. In order to add two collocation points, two intermediate points are added to the knot vector.

3.2 Example 1B

Over Damped, Second Order ODE Approximated with a Third Order B-spline, with Two Intermediate Knot Vector Points

$P'' + 2P' + P = 0$ For $0 < x < 1$

Boundary Conditions: $P(0) = 0$, $P(1) = 1$

1. Once again, the first step is to select the appropriate knot vector. A third order, open-uniform knot vector with two intermediate points has been selected.

   [0 0 0 1/3 2/3 1 1 1]

2. The next step is to calculate all of the required basis functions for each range of values of the parameter 't' using equations (2) and (3).

$$N_{i,1}(t) = \begin{cases} 1 \; x_i \le t < x_{i+1} \\ 0 \;\; otherwise \end{cases} \qquad N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

$0 \le t < \frac{1}{3}$

$N_{1,1}(t) = 0;$ $N_{2,1}(t) = 0;$ $N_{3,1}(t) = 1;$ $N_{4,1}(t) = 0;$

$N_{1,2}(t) = 0;$ $N_{2,2}(t) = 1 - 3t;$ $N_{3,2}(t) = 3t;$ $N_{4,2}(t) = 0;$

$N_{1,3}(t) = (1 - 3t)^2;$ $N_{2,3}(t) = \frac{3t}{2}(4 - 9t);$ $N_{3,3}(t) = \frac{9t^2}{2};$ $N_{4,3}(t) = 0$

$\frac{1}{3} \le t < \frac{2}{3}$

$N_{2,1}(t) = 0;$ $N_{3,1}(t) = 0;$ $N_{4,1}(t) = 1;$ $N_{5,1}(t) = 0;$

$N_{2,2}(t) = 0;$ $N_{3,2}(t) = 2 - 3t;$ $N_{4,2}(t) = 3t - 1;$ $N_{5,2}(t) = 0;$

$N_{2,3}(t) = \frac{(2 - 3t)^2}{2};$ $N_{3,3}(t) = \frac{-3}{2}(6t^2 - 6t + 1);$ $N_{4,3}(t) = \frac{(1 - 3t)^2}{2};$ $N_{5,3}(t) = 0$

$\frac{2}{3} \le t < 1$

$N_{2,1}(t) = 0;$ $N_{3,1}(t) = 0;$ $N_{4,1}(t) = 0;$ $N_{5,1}(t) = 1;$

$N_{2,2}(t) = 0;$ $N_{3,2}(t) = 0;$ $N_{4,2}(t) = 3 - 3t;$ $N_{5,2}(t) = 3t - 2;$

$N_{2,3}(t) = 0;$ $N_{3,3}(t) = \frac{9(t - 1)^2}{2};$ $N_{4,3}(t) = \frac{-3}{2}(9t^2 - 14t + 5);$ $N_{5,3}(t) = (2 - 3t)^2$

By plotting the k = 3 basis functions for each interval together, as shown in Figure 9, it can be seen that all the required criteria is met and it is safe to move on to the next step.

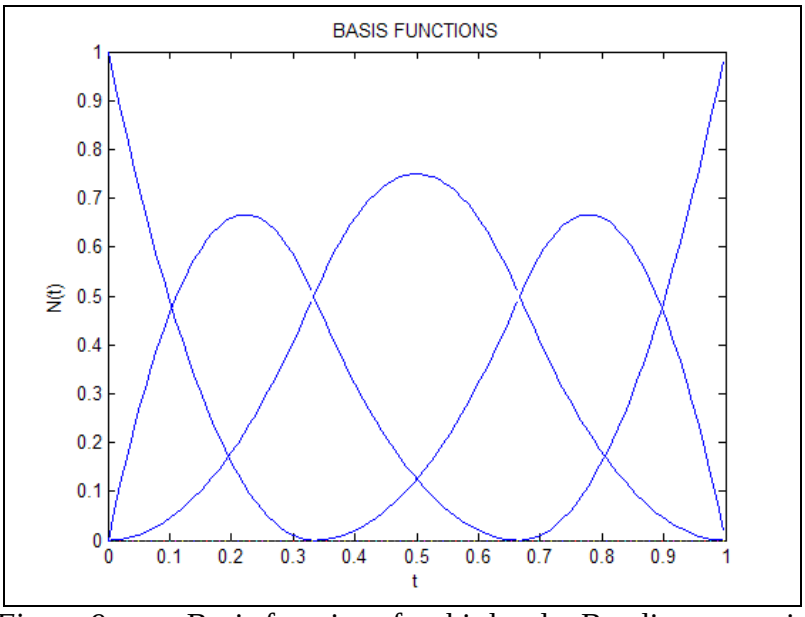

Figure 9 Basis functions for third order B-spline approximation with two intermediate Knot Vector points

3. The third step is to calculate all of the abscissa coordinates values for the required control points, using the Greville Abscissa formula, equation (4).

$$x_i = \frac{1}{n}(t_i + t_{i+1} + \ldots + t_{i+n-1}) \qquad \text{For i=0, 1,…,m-n}$$

$$x_0 = \frac{1}{2}(t_0 + t_1) = \frac{1}{2}(0+0) = 0$$

$$x_1 = \frac{1}{2}(t_1 + t_2) = \frac{1}{2}(0+0) = 0$$

$$x_2 = \frac{1}{2}(t_2 + t_3) = \frac{1}{2}(0+\frac{1}{3}) = \frac{1}{6}$$

$$x_3 = \frac{1}{2}(t_3 + t_4) = \frac{1}{2}(\frac{1}{3}+\frac{2}{3}) = \frac{1}{2}$$

$$x_4 = \frac{1}{2}(t_4 + t_5) = \frac{1}{2}(\frac{2}{3}+1) = \frac{5}{6}$$

$$x_5 = \frac{1}{2}(t_5 + t_6) = \frac{1}{2}(1+1) = 1$$

$$x_6 = \frac{1}{2}(t_6 + t_7) = \frac{1}{2}(1+1) = 1$$

Once again, the two repeating values at the ends should be dropped and that will leave $x_1 = 0$, $x_2 = \frac{1}{6}$, $x_3 = \frac{1}{2}$, $x_4 = \frac{5}{6}$ and $x_5 = 1$ as the abscissa coordinates for the control points.

4. The fourth step is to calculate the appropriate B-spline curve equations and the required derivatives for the boundary conditions that need to be met. Since the third term of the differential equation and both boundary conditions involve the B-spline curve, it must be calculated for each range of the parameter‘t’. The B-spline curve must be calculated, by using equation (1).

For $0 \le t < \frac{1}{3}$ $\quad P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$

$$P(t) = B_1 N_{1,3}(t) + B_2 N_{2,3}(t) + B_3 N_{3,3}(t)$$

$$\boxed{P(t) = B_1\left[(1-3t)^2\right] + B_2\left[\frac{3t}{2}(4-9t)\right] + B_3\left[\frac{9t^2}{2}\right]}$$

For $\frac{1}{3} \le t < \frac{2}{3}$ $\quad P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$

$$P(t) = B_2 N_{2,3}(t) + B_3 N_{3,3}(t) + B_4 N_{4,3}(t)$$

$$\boxed{P(t) = B_2\left[\frac{(2-3t)^2}{2}\right] + B_3\left[\frac{-3}{2}(6t^2 - 6t + 1)\right] + B_4\left[\frac{(1-3t)^2}{2}\right]}$$

For $\frac{2}{3} \le t < 1$ $\quad P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$

$$P(t) = B_3 N_{3,3}(t) + B_4 N_{4,3}(t) + B_5 N_{5,3}(t)$$

$$\boxed{P(t) = B_3\left[\frac{9(t-1)^2}{2}\right] + B_4\left[\frac{-3}{2}(9t^2 - 14t + 5)\right] + B_5\left[(2-3t)^2\right]}$$

Since the second term of the differential equation involves the first derivative, the first derivative of the B-spline equation must be calculated for each range of the parameter‘t’, using equation (5).

For $0 \le t < \frac{1}{3}$ $\quad P'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t)$

$$P'(t) = B_1 N'_{1,3}(t) + B_2 N'_{2,3}(t) + B_3 N'_{3,3}(t)$$

$$\boxed{P'(t) = B_1[18t - 6] + B_2[6 - 27t] + B_3[9t]}$$

For $\frac{1}{3} \le t < \frac{2}{3}$ $\quad P'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t)$

$$P'(t) = B_2 N'_{2,3}(t) + B_3 N'_{3,3}(t) + B_4 N'_{4,3}(t)$$

$$\boxed{P'(t) = B_2[9t - 6] + B_3[(9 - 18t)] + B_4[9t - 3]}$$

For $\frac{2}{3} \le t < 1$ $\quad P'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t)$

$$P'(t) = B_3 N'_{3,3}(t) + B_4 N'_{4,3}(t) + B_5 N'_{5,3}(t)$$

$$\boxed{P'(t) = B_3[9(t-1)] + B_4[21 - 27t] + B_5[6(3t - 2)]}$$

The first term of the differential equation involves the second derivative. The second derivative of the B-spline curve can be calculated for each range of the parameter‘t’, using equation (6).

For $0 \le t < \frac{1}{3}$ $\quad P''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t)$

$$P''(t) = B_1 N''_{1,3}(t) + B_2 N''_{2,3}(t) + B_3 N''_{3,3}(t)$$

$$\boxed{P''(t) = B_1[18] + B_2[(-27)] + B_3[9]}$$

For $\frac{1}{3} \le t < \frac{2}{3}$ $\quad P''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t)$

$$P''(t) = B_2 N''_{2,3}(t) + B_3 N''_{3,3}(t) + B_4 N''_{4,3}(t)$$

$$\boxed{P''(t) = B_2[9] + B_3[(-18)] + B_4[9]}$$

For $\frac{2}{3} \le t < 1$ $\quad P''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t)$

$$P''(t) = B_3 N''_{3,3}(t) + B_4 N''_{4,3}(t) + B_5 N''_{5,3}(t)$$

$$\boxed{P''(t) = B_3[9] + B_4[(-27)] + B_5[18]}$$

5. The next step is to use the boundary conditions to solve for end ordinate values of control points. Since the Greville Abscissa were used as control points, the parametric coordinate 't' is constrained to the Cartesian coordinate 'x' and it becomes a direct substitution. The boundary conditions were P(0) = 0 and P(1) = 1. Since the B-spline curve is a function over multiple ranges, the appropriate range must be used when calculating the external ordinate values.

For $0 \le t < \frac{1}{3}$ $\quad P(t) = B_1[(1-3t)^2] + B_2\left[\frac{3t}{2}(4-9t)\right] + B_3\left[\frac{9t^2}{2}\right]$

P(0) = 0➔$B_1 = 0$➔ $y_1 = 0$

For $\frac{2}{3} \le t < 1$ $\quad P(t) = B_3\left[\frac{9(t-1)^2}{2}\right] + B_4\left[\frac{-3}{2}(9t^2 - 14t + 5)\right] + B_5[(2-3t)^2]$

P(1) = 1➔$B_5 = 1$➔ $y_5 = 1$

6. The sixth step is to substitute B-spline curves and the derivatives into original differential equation, P'' + 2P' + P = 0. We get an equation for each range of the parameter 't'.

For $0 \le t < \frac{1}{3}$

$$\{B_1[18] + B_2[(-27)] + B_3[9]\} + 2\{B_1[18t-6] + B_2[6-27t] + B_3[9t]\} +$$

$$\left\{B_1[(1-3t)^2] + B_2\left[\frac{3t}{2}(4-9t)\right] + B_3\left[\frac{9t^2}{2}\right]\right\} = 0$$

Now using the ordinate values from the previous step, the differential equation for this range reduces to:

$$\{B_2[(-27)] + B_3[9]\} + 2\{B_2[6-27t] + B_3[9t]\} + \left\{B_2\left[\frac{3t}{2}(4-9t)\right] + B_3\left[\frac{9t^2}{2}\right]\right\} = 0$$

For $\frac{1}{3} \le t < \frac{2}{3}$

$$\{B_2[9] + B_3[(-18)] + B_4[9]\} + 2\{B_2[9t-6] + B_3[(9-18t)] + B_4[9t-3]\} +$$

$$\left\{B_2\left[\frac{(2-3t)^2}{2}\right] + B_3\left[\frac{-3}{2}(6t^2-6t+1)\right] + B_4\left[\frac{(1-3t)^2}{2}\right]\right\} = 0$$

For $\frac{2}{3} \le t < 1$

$$\{B_3[9] + B_4[(-27)] + B_5[18]\} + 2\{B_3[9(t-1)] + B_4[21-27t] + B_5[6(3t-2)]\} +$$

$$\left\{B_3\left[\frac{9(t-1)^2}{2}\right] + B_4\left[\frac{-3}{2}(9t^2-14t+5)\right] + B_5[(2-3t)^2]\right\} = 0$$

Once again using the ordinate values from the previous step, the differential equation for this range reduces to:

$$\{B_3[9] + B_4[(-27)] + [18]\} + 2\{B_3[9(t-1)] + B_4[21-27t] + [6(3t-2)]\} +$$

$$\left\{B_3\left[\frac{9(t-1)^2}{2}\right] + B_4\left[\frac{-3}{2}(9t^2-14t+5)\right] + [(2-3t)^2]\right\} = 0$$

7. The seventh step is to calculate the remaining internal ordinate coordinates of the control points by evaluation of the previously reduced differential equation, using the appropriate range. At 't' = 0.1667, using the equation for the range $0 \le t < \frac{1}{3}$,

$$\{B_2[(-27)] + B_3[9]\} + 2\{B_2[6-27t] + B_3[9t]\} + \left\{B_2\left[\frac{3t}{2}(4-9t)\right] + B_3\left[\frac{9t^2}{2}\right]\right\} = 0,$$

the equation reduces to: $\boxed{-23.375B_2 + 12.125\ B_3 = 0.}$

At 't' = 0.5000, using the equation for the range $\frac{1}{3} \le t < \frac{2}{3}$,

$$\{B_2[9] + B_3[(-18)] + B_4[9]\} + 2\{B_2[9t-6] + B_3[(9-18t)] + B_4[9t-3]\} +$$

$$\left\{B_2\left[\frac{(2-3t)^2}{2}\right] + B_3\left[\frac{-3}{2}(6t^2 - 6t + 1)\right] + B_4\left[\frac{(1-3t)^2}{2}\right]\right\} = 0,$$

the equation reduces to: $\boxed{6.125B_2 - 17.25\ B_3 + 12.125B_4 = 0.}$

At 't' = 0.8333, using the equation for the range $\frac{2}{3} \le t < 1$,

$$\{B_3[9] + B_4[(-27)] + [18]\} + 2\{B_3[9(t-1)] + B_4[21-27t] + [6(3t-2)]\} +$$

$$\left\{B_3\left[\frac{9(t-1)^2}{2}\right] + B_4\left[\frac{-3}{2}(9t^2 - 14t + 5)\right] + [(2-3t)^2]\right\} = 0,$$

the equation reduces to: $\boxed{6.125B_3 - 29.375\ B_4 = -24.25.}$

With three equations and three unknowns, use Linear Algebra to solve for the remaining values for the internal ordinate coordinates:

$$\begin{bmatrix} -23.375 & 12.125 & 0 \\ 6.125 & -17.25 & 12.125 \\ 0 & 6.125 & -29.375 \end{bmatrix} \begin{pmatrix} B_2 \\ B_3 \\ B_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -24.25 \end{pmatrix}$$

The remaining internal ordinate coordinates are calculated to be:

$B_2 = y_2 = 0.4497$ $B_3 = y_3 = 0.8670$

$B_4 = y_4 = 1.0063$

Now that all of the coordinates for the control points have been calculated, they can be substituted into the B-spline function equation, to give the final equation for the B-spline approximation for each range.

For $0 \le x < \frac{1}{3}$ $P(x) = (2.6982 - 2.16932x)x$

For $\frac{1}{3} \le x < \frac{2}{3}$ $P(x) = -1.2512(x - 1.715)(x + 0.0476)$

For $\frac{2}{3} \le x < 1$ $P(x) = -0.6837(x - 2.1820)(x + 0.2375)$

Now, once again, a comparison of the B-spline function approximation and the exact solution, P(x) = $2.718 * x * e^{-x}$, must be done to determine the accuracy of the collocation method. The first step to determining the accuracy is to evaluate both the B-spline function approximation and the exact solution at the collocation points. Next, calculate the relative error percentage, by using equation (14) for each collocation point. These values have been listed in Table 2.

Collocation Point Evaluation			
Point	Approximate	Exact	%Error
0	0	0	0
0.1667	0.38947	0.38346	1.568
0.5	0.83228	0.8243	0.971
0.8333	0.9873	0.9844	0.301
1	1	0.9999	0.001

Table 2 Table of collocation points for Example 1b

Once all of the relative error percentage calculations have been done, the Root Mean Squared Error must be computed, by using equation (15).

RMS Error = 0.84%

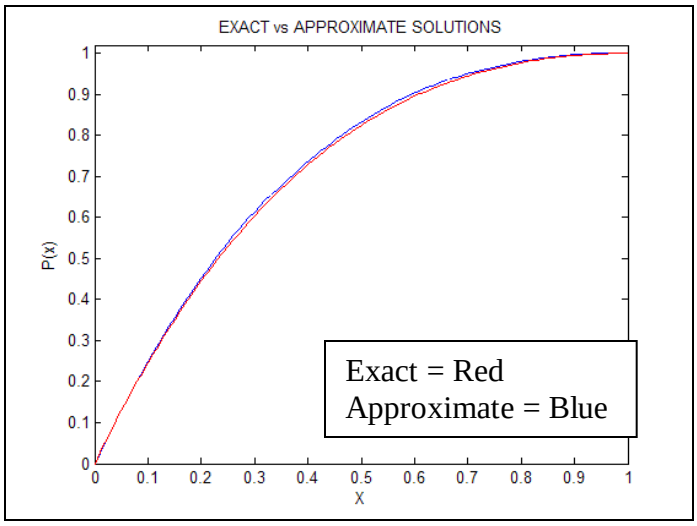

Figure 10 B-spline solution plotted with the exact solution

The RMS Error of 0.84% is still greater than the 0.5% baseline that was previously established, but substantially less than the 2.3% from the previous example. In fact, by plotting the two functions for the entire range, as seen in Figure 10, it can be shown that the B-spline approximation is much closer to the exact solution. This increase in the precision of the approximation is achieved by just inserting two intermediate points in the knot vector. As previously discussed, another way to increase the accuracy is to increase the order of the approximation. Figure 11, shows the solution plotted for the knot vector [0 0 0 0 0 1 1 1 1 1]. The RMS error for this solution is calculated to be 0.036%, well below the 0.5% threshold.

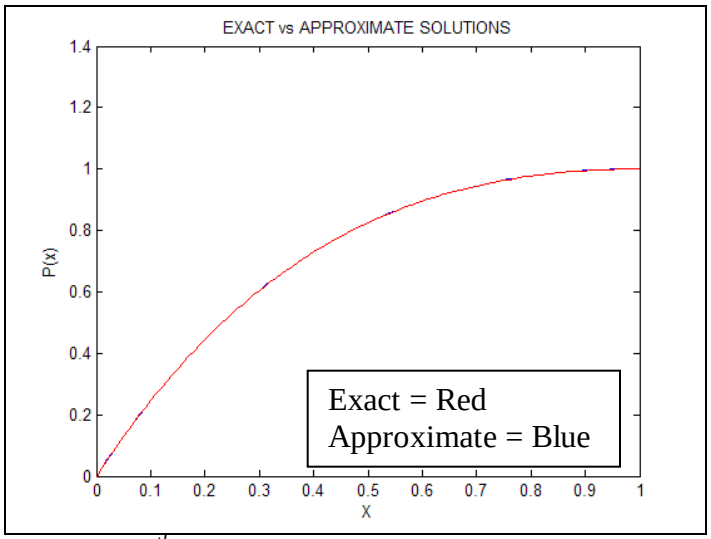

Figure 11 5^{th} Order B-spline solution plotted with the exact solution

It can be seen from the previous examples that the B-spline Collocation Method is extremely efficient at approximating the solutions of over damped boundary value problems. The next example will take the B-spline Collocation Method, using exactly the same steps, and apply it to a simple un-damped problem.

3.3 Example 2

Un-Damped, Second Order ODE Approximated with a Third Order B-spline, with No Intermediate Knot Vector Points

$P''+\lambda^2 P = 0$ For $0 < x < 1$ Where $\lambda = 4$

Boundary Conditions: P(0) = 0, P(1) = 1

1. The first step in the B-spline collocation method is to select the appropriate knot vector that provides the resolution required for the problem. A third order, open-uniform knot vector has been selected.

   [0 0 0 1 1 1]

2. The second step is to calculate all required basis functions using equations (2) and (3). The basis functions for this knot vector were calculated in Example 1a and are:

   $N_{1,3}(t) = (1-t)^2$; $N_{2,3}(t) = 2t(1-t)$; $N_{3,3}(t) = t^2$; $N_{4,3}(t) = 0$.

3. The next step is to calculate the abscissa coordinates for the control points. Since the Greville Abscissa approach has been chosen, use equation (4) to calculate the Greville Abscissa for required control points. These also were calculated in Example 1a. Once again, the repeating values of $x_0 = 0$ and $x_4 = 1$ should be dropped. That leaves $x_1 = 0$, $x_2 = \frac{1}{2}$ and $x_3 = 1$ as the abscissa coordinates for the control points.

4. The fourth step is to calculate the appropriate B-spline curve equations and the required derivatives for the boundary conditions that need to be met. Since the third term of the

differential equation and both boundary conditions involve the B-spline curve, it must be calculated. The B-spline curve must be calculated, by using equation (1).

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$$

$$P(t) = B_1 N_{1,3}(t) + B_2 N_{2,3}(t) + B_3 N_{3,3}(t)$$

$$\boxed{P(t) = B_1\left[(1-t)^2\right] + B_2\left[2t(1-t)\right] + B_3\left[t^2\right]}$$

Since the second term of the differential equation involves the first derivative, the first derivative of the B-spline curve must be calculated, using equation (5).

$$P'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t)$$

$$P'(t) = B_1 N'_{1,3}(t) + B_2 N'_{2,3}(t) + B_3 N'_{3,3}(t)$$

$$\boxed{P'(t) = B_1\left[2(t-1)\right] + B_2\left[(2-4t)\right] + B_3\left[2t\right]}$$

The first term of the differential equation involves the second derivative. The second derivative of the B-spline curve can be calculated using equation (6).

$$P''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t)$$

$$P''(t) = B_1 N''_{1,3}(t) + B_2 N''_{2,3}(t) + B_3 N''_{3,3}(t)$$

$$\boxed{P''(t) = B_1\left[2\right] + B_2\left[(-4)\right] + B_3\left[2\right]}$$

5. The next step is to use the boundary conditions to solve for end ordinate values of control points. Since the Greville Abscissa were used as control points, the parametric coordinate ‘t’ is constrained to the Cartesian coordinate ’x’ and it becomes a direct substitution. The boundary conditions were P(0) = 0 and P(1) = 1. When the B-spline curve, $P(t) = B_1\left[(1-t)^2\right] + B_2\left[2t(1-t)\right] + B_3\left[t^2\right]$, is evaluated at the boundaries, the following external ordinate values are calculated to be:

P(0) = 0➔B_1 = 0➔ $y_1 = 0$

P(1) = 1➔ B_3 = 1➔ $y_3 = 1$

6. The sixth step is to substitute B-spline curve and the derivatives into the differential equation, $P''+\lambda^2 P = 0$, where $\lambda = 4$, and we get:

   $$\{B_1[2]+B_2[(-4)]+B_3[2]\}+16\{B_1[(1-t)^2]+B_2[2t(1-t)]+B_3[t^2]\}=0.$$

   Now using the ordinate values from the previous step, the differential equation reduces to: $\{B_2[(-4)]+[2]\}+16\{B_2[2t(1-t)]+[t^2]\}=0$.

7. The seventh step is to calculate the remaining internal ordinate coordinates of the control points by evaluation of the previously reduced differential equation, $\{B_2[(-4)]+[2]\}+2\{B_2[(2-4t)]+[2t]\}+\{B_2[2t(1-t)]+[t^2]\}=0$, at the corresponding abscissa coordinates. At't' = ½, the remaining internal ordinate coordinate is calculated to be:

   $B_2 = y_2 =$ -1.5000

Now that all of the coordinates for the control points have been calculated, they can be substituted into the B-spline curve equation, $P(t) = B_1[(1-t)^2]+B_2[2t(1-t)]+B_3[t^2]$ to give the final equation for the B-spline approximation. Since the B-spline curve is in terms of the parametric coordinate system and the desired result is an equation in the Cartesian coordinate system, a bit of manipulation is required. As stated earlier, using the Greville Abscissa allows a direct substitution of the 'x' and't' parameters. The B-spline function approximation equation then becomes $\boxed{P(x) = [x^2]-[3x(1-x)]}$.

Now a comparison of the B-spline function approximation and the exact solution, $P(x) = \frac{1}{Sin[\lambda]} * Sin[\lambda x]$, must be done to determine the accuracy of the collocation method. The first step to determining the accuracy is to evaluate both the B-spline function approximation and the exact solution at the collocation points. Next, calculate the relative

error percentage, by using equation (14) for each collocation point. These values have been listed in Table 3.

Collocation Point Evaluation			
Point	Approximate	Exact	%Error
0	0	0	0
0.5	-0.5	-1.2015	58.385
1	1	0.9999	0.001

Table 3 Table of collocation points for Example 2

Once all of the relative error percentage calculations have been done, the Root Mean Squared Error must be computed, by using equation (15).

RMS Error = 33.7%

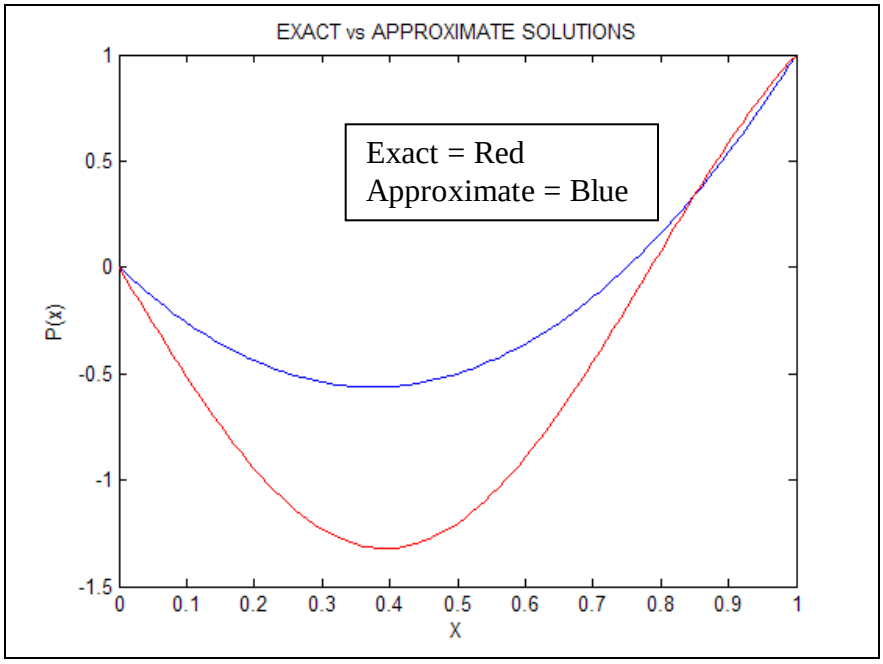

Figure 12 B-spline solution plotted with the exact solution

The RMS Error of 33.7% is way greater than the 0.5% baseline, previously established. By plotting the two functions for the entire range, as seen in Figure 12 the areas where refinement is necessary can be determined. As previously discussed, one method of refinement is to increase the order of the B-spline Basis Functions. Another method is to add a collocation point within the range where the highest error percentage is. By a visual inspection of Figure 12 it can be seen that the largest difference between the two functions is

within the range of $0.1 < x < 0.7$. The large amount of error, over a long range, suggests an increase in order is necessary. Figure 13, shows the 7^{th} order solution plotted for the knot vector [0 0 0 0 0 0 0 1 1 1 1 1 1 1]. The RMS error for this solution is calculated to be 0.31%, which is below the 0.5% threshold.

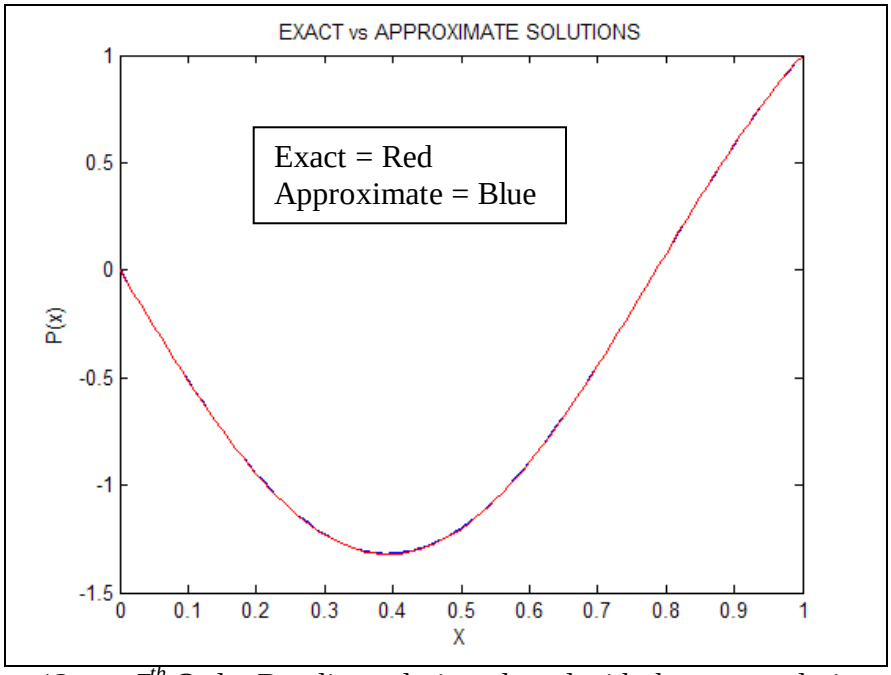

Figure 13 7^{th} Order B-spline solution plotted with the exact solution

The B-spline Collocation Method can also be used to approximate the solutions to higher order problems. The next example is the approximation of the fourth order Euler-Bernoulli Beam Equation, cantilevered with a distributed load over the entire beam length, depicted in Figure 14. This example will also show the normalization process of the B-spline Collocation Method when the range of the problem solution is greater than one.

3.4 Example 3

Fourth Order Euler-Bernoulli Beam, Approximated with a Sixth Order B-spline, with No Intermediate Knot Vector Points

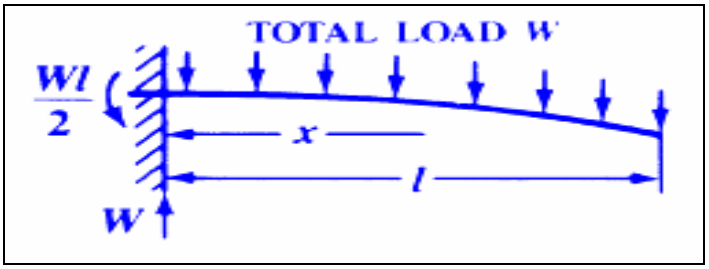

Figure 14 *E*uler-Bernoulli Beam Example Diagram

The calculation of deflection of linear beam problems is governed by the Euler-Bernoulli Beam Equation, $EI\frac{d^4u}{dx^4} = -w$, where E is the Modulus of Elasticity, I is the Moment of Inertia of the cross section about its neutral axis and w is the distributed load.

Boundary Conditions: $u(0) = 0, \quad \frac{du}{dx}(0) = 0, \quad \frac{d^2u}{dx^2}(L) = 0, \quad \frac{d^3u}{dx^3}(L) = 0$

$E = 29000000 \ \frac{lbs}{in^2}$ (ASTM-A36 STEEL)

$I = 0.6 \ in^4$ (2” x 2” x 1/8” WALL)

$w = 10 \ \frac{lbs}{in}$

L = 10 in

1. The first step in the B-spline collocation method is to select the appropriate knot vector that provides the resolution required for the problem. A sixth order, open-uniform knot vector with no intermediate points has been selected.

   [0 0 0 0 0 0 1 1 1 1 1 1]

2. The second step is to calculate all required basis functions using equations (2) and (3). Here only the basis functions with k = 6 have been listed.

$$N_{i,1}(t) = \begin{cases} 1 \; x_i \le t < x_{i+1} \\ 0 \;\; otherwise \end{cases} \qquad N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

$$N_{1,6}(t) = (1-t)^5; \qquad N_{2,6}(t) = 5t(1-t)^4; \qquad N_{3,6}(t) = 10t^2(1-t)^3;$$

$$N_{4,6}(t) = 10t^3(1-t)^2; \qquad N_{5,6}(t) = 5t^4(1-t); \qquad N_{6,6}(t) = t^5$$

By plotting the k = 6 basis functions, the generated curves can be used to verify that the calculated basis functions comply with the criteria discussed in the previous chapter. In Figure 15, it can be seen that indeed they do meet the required criteria and it is safe to move on to the next step.

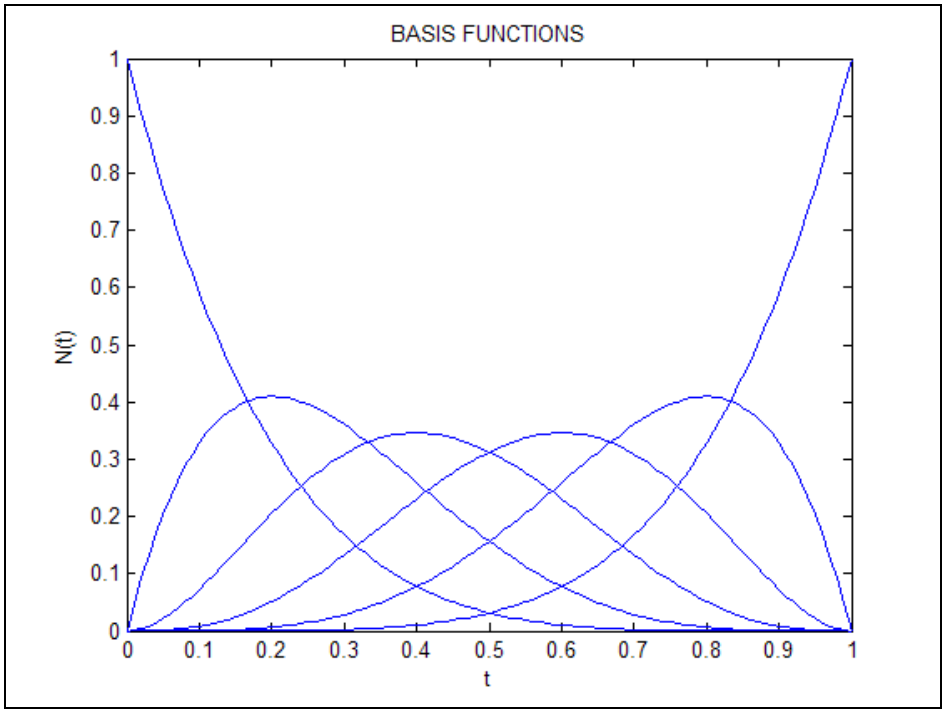

Figure 15 Basis functions for sixth order B-spline approximation

3. The next step is to calculate the abscissa coordinates for the control points. Since the Greville Abscissa approach has been chosen, use equation (4) to calculate the Greville Abscissa for required control points.

$x_i = \frac{1}{n}(t_i + t_{i+1} + ... + t_{i+n-1})$ For i=0, 1,…,m-n

$$x_0 = \frac{1}{5}(t_0 + t_1 + t_2 + t_3 + t_4) = \frac{1}{5}(0+0+0+0+0) = 0$$

$$x_1 = \frac{1}{5}(t_1 + t_2 + t_3 + t_4 + t_5) = \frac{1}{5}(0+0+0+0+0) = 0$$

$$x_2 = \frac{1}{5}(t_2 + t_3 + t_4 + t_5 + t_6) = \frac{1}{5}(0+0+0+0+1) = \frac{1}{5}$$

$$x_3 = \frac{1}{5}(t_3 + t_4 + t_5 + t_6 + t_7) = \frac{1}{5}(0+0+0+1+1) = \frac{2}{5}$$

$$x_4 = \frac{1}{5}(t_4 + t_5 + t_6 + t_7 + t_8) = \frac{1}{5}(0+0+1+1+1) = \frac{3}{5}$$

$$x_5 = \frac{1}{5}(t_5 + t_6 + t_7 + t_8 + t_9) = \frac{1}{5}(0+1+1+1+1) = \frac{4}{5}$$

$$x_6 = \frac{1}{5}(t_6 + t_7 + t_8 + t_9 + t_{10}) = \frac{1}{5}(1+1+1+1+1) = 1$$

$$x_7 = \frac{1}{5}(t_7 + t_8 + t_9 + t_{10} + t_{11}) = \frac{1}{5}(1+1+1+1+1) = 1$$

Once again, the two repeating values at the ends should be dropped and that will leave $x_1 = 0$, $x_2 = \frac{1}{5}$, $x_3 = \frac{2}{5}$, $x_4 = \frac{3}{5}$, $x_5 = \frac{4}{5}$ and $x_6 = 1$ as the abscissa coordinates for the control points.

4. The fourth step is to calculate the appropriate B-spline curve equations and the required derivatives for the boundary conditions that need to be met. Since the first boundary condition involve the B-spline curve. The B-spline curve must be calculated, by using equation (1).

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$$

$$P(t) = B_1\left[(1-t)^5\right] + B_2\left[5t(1-t)^4\right] + B_3\left[10t^2(1-t)^3\right] + B_4\left[10t^3(1-t)^2\right] + B_5\left[5t^4(1-t))\right] + B_6\left[t^5\right]$$

Since the second boundary condition involves the first derivative, the first derivative of the B-spline equation must be calculated, using equation (5).

$$P'(t) = \sum_{i=1}^{n+1} B_i N_{i,k}'(t)$$

$$P'(t) = B_1\left[-5(1-t)^4\right] + B_2\left[5(1-t)^4 - 20t(1-t)^3\right] + B_3\left[20t(1-t)^3 - 30t^2(1-t)^2\right] + B_4\left[30t^2(1-t)^2 - 20t^3(1-t)\right] + B_5\left[20t^3(1-t) - 5t^4\right] + B_6\left[5t^4\right]$$

The third boundary condition involves the second derivative; therefore the second derivative of the B-spline equation must be calculated, using equation (6).

$$P''(t) = \sum_{i=1}^{n+1} B_i N_{i,k}''(t)$$

$$P''(t) = B_1\left[20(1-t)^3\right] + B_2\left[60t(1-t)^2 - 40(1-t)^3\right] + B_3\left[20(1-t)^3 - 120t(1-t)^2 + 60t^2(1-t)\right] + B_4\left[60t(1-t)^2 - 120t^2(1-t) + 20t^3\right] + B_5\left[60t^2(1-t) - 40t^3\right] + B_6\left[20t^3\right]$$

Additionally, the last boundary condition involves the third derivative:

$$P'''(t) = \sum_{i=1}^{n+1} B_i N_{i,k}'''(t)$$

$$P'''(t) = B_1\left[-60(1-t)^2\right] + B_2\left[180(1-t)^2 - 120t(1-t)\right] + B_3\left[-180(1-t)^2 + 360t(1-t) - 60t^2\right] + B_4\left[60(1-t)^2 - 360t(1-t) + 180t^2\right] + B_5\left[120t(1-t) - 180t^2\right] + B_6\left[60t^2\right]$$

Finally, the first term of the differential equation is fourth order:

$$P''''(t) = \sum_{i=1}^{n+1} B_i N_{i,k}''''(t)$$

$$P''''(t) = B_1\left[120 - 120t\right] + B_2\left[-480 + 600t\right] + B_3\left[720 - 1200t\right] + B_4\left[-480 + 1200t\right] + B_5\left[120 - 600t\right] + B_6\left[120t\right]$$

5. The next step is to use the boundary conditions to solve for end ordinate values of control points. Since the Greville Abscissa were used as control points, the parametric coordinate 't' is constrained to the Cartesian coordinate 'x' and it becomes a direct substitution.

$$P(0) = 0 \rightarrow B_1 = 0$$

$$\frac{dP}{dx}(0) = 0 \rightarrow B_2 = 0$$

$$\frac{d^2u}{dx^2}(L) = 0 \rightarrow \boxed{-165780B_3 + 176600B_4 - 94000B_5 + 20000B_6 = 0}$$

$$\frac{d^3u}{dx^3}(L) = 0 \rightarrow \boxed{-52980B_3 + 55260B_4 - 28800B_5 + 6000B_6 = 0}$$

6. The sixth step is to substitute B-spline derivatives into the differential equation, $EI\frac{d^4u}{dx^4} = -w$, along with the values from the previous step, we get:

$$B_3[720 - 1200t] + B_4[-480 + 1200t] + B_5[120 - 600t] + B_6[120t] = \frac{-wL^4}{EI}$$

7. The seventh step is to calculate the remaining internal ordinate coordinates of the control points by evaluation of the previously reduced differential equation, $B_3[720 - 1200t] + B_4[-480 + 1200t] + B_5[120 - 600t] + B_6[120t] = \frac{-wL^4}{EI}$, at the corresponding abscissa coordinates.

   At't' = 0.4, the differential equation reduces to be:

$$\boxed{B_3[240] + B_5[-120] + B_6[48] = \frac{-wL^4}{EI}}$$

   At't' = 0.6, the differential equation reduces to be:

$$\boxed{B_4[240] + B_5[-240] + B_6[72] = \frac{-wL^4}{EI}}$$

With four equations and four unknowns, , use Linear Algebra to solve for the remaining values for the internal ordinate coordinates:

$$\begin{bmatrix} -165780 & 176600 & -94000 & 20000 \\ -52980 & 55260 & -28800 & 6000 \\ 240 & 0 & -120 & 48 \\ 0 & 240 & -240 & 72 \end{bmatrix} \begin{pmatrix} B_3 \\ B_4 \\ B_5 \\ B_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \frac{-wL^4}{EI} \\ \frac{-wL^4}{EI} \end{pmatrix}$$

The remaining internal ordinate coordinates are calculated to be:

$B_3 = -0.0144$

$B_4 = -0.0421$

$B_5 = -0.0824$

$B_6 = -0.1343$

Now that all of the coordinates for the control points have been calculated, they can be substituted into the B-spline curve equation,

$$P(t) = B_1\left[(1-t)^5\right] + B_2\left[5t(1-t)^4\right] + B_3\left[10t^2(1-t)^3\right] + B_4\left[10t^3(1-t)^2\right] + B_5\left[5t^4(1-t))\right] + B_6\left[t^5\right]$$

to give the final equation for the B-spline approximation. Since the B-spline curve is in terms of the parametric coordinate system and the desired result is an equation in the Cartesian coordinate system, a bit of manipulation is required. As stated earlier, using the Greville Abscissa allows a direct substitution of the 'x' and't' parameters. The B-spline function approximation equation then becomes:

$$\boxed{P(x) = 1.437 * 10^{-5} x^2(1-x)^3 - 0.4214x^3(1-x)^2 - .4121x^4(1-x) - .1343x^5}.$$

Now a comparison of the B-spline function approximation and the exact solution,

P(x) = $\frac{x^2 w}{24EI}\left[4Lx - x^2 - 6L^2\right]$, must be done to determine the accuracy of the collocation method. The first step to determining the accuracy is to evaluate both the B-spline function

approximation and the exact solution at the collocation points. Next, calculate the relative error percentage, by using equation (14) for each collocation point. These values have been listed in Table 4.

Collocation Point Evaluation			
Point	Approximate	Exact	%Error
0	0	0	0
0.2	-5.67E-08	-5.67E-08	3.63E-09
0.4	-2.24E-07	-2.24E-07	3.65E-09
0.6	-4.97E-06	-4.97E-06	3.66E-09
0.8	-8.72E-06	-8.72E-06	3.68E-09
1	-1.34E-05	-1.34E-05	3.68E-09

Table 4 Table of collocation points for Example 3

Once all of the relative error percentage calculations have been done, the Root Mean Squared Error must be computed, by using equation (15).

RMS Error = 3.35e-9%

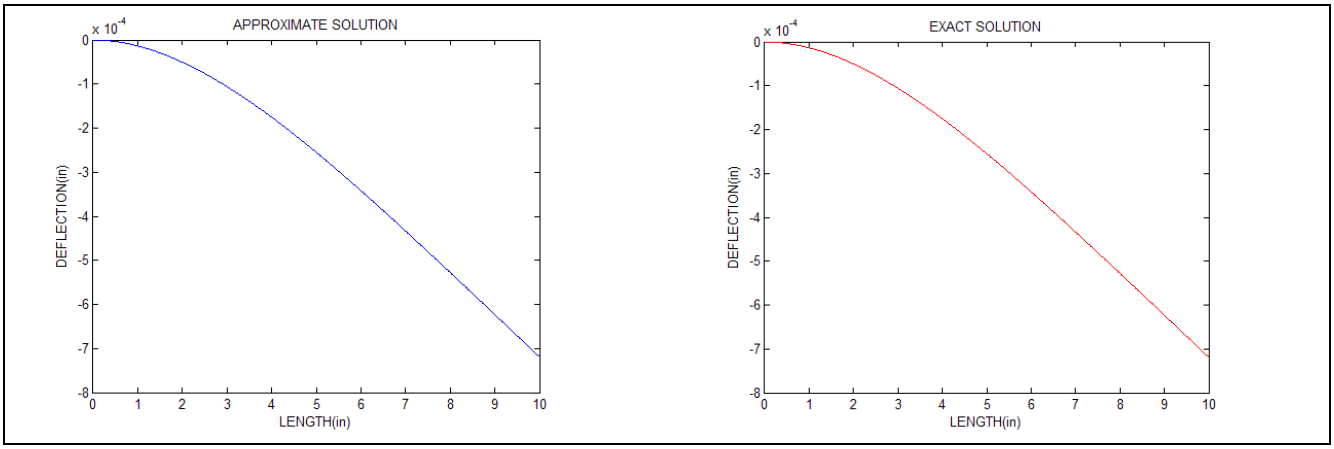

Figure 16 B-spline solution plotted with the exact solution

The RMS Error of 3.35e-9% is way less than the 0.5% baseline, previously established. By plotting the two functions for the entire range, as seen in Figure 16, it can be seen that no refinement is necessary.

3.5 Discussion of the Accuracy of the B-Spline Approximation

In order to determine if the B-spline used in the approximation should be a higher order, or a lower order with intermediate points, the error for each should be calculated. Also, the time to compute the solution should be measured. As stated previously, the benchmark value for RMS error percentage in the B-Spline Collocation Method approximation technique is set at < 0.5%. Table 5 shows the calculated RMS Error percentage for the over damped solution approximation of Examples 1a and 1b for different combinations of control points.

Order	# of Internal Knot Points	RMS Error %
3	0	2.3
4	0	0.631
5	0	0.036
3	1	1.573
3	2	0.836
3	3	0.515
3	4	0.349
4	1	0.477

Table 5 Over Damped Solution Comparison

With no internal points in the knot vector, Table 5 shows that it will take a fifth order approximation to get the RMS error below 0.5 % for the third order over damped solution problem. In Table 5, it can also be seen that it will take four intermediate points for a third order approximation, and one intermediate point for a fourth order approximation to reach the specified target RMS error value.

The most efficient way to approximate the over damped solution is to find the right balance of the approximation order, number of internal knot points and computation time. Table 6 shows the computation times for the approximations that meet the accuracy goals.

Order	# of Internal Knot Points	CPU Time (sec)
3	4	2.93
4	1	1.95
5	0	1.45

Table 6 Over Damped Solution Time Comparison

For this simple problem, the best accuracy and computation times both belong to the fifth order approximation, with no internal knot points.

With the under damped solution problem, in Example 2, the efficiency of the B-Spline Collocation Method decreases as the frequency of the problem increases. By evaluating the results in Tables 7, 8 and 10, notice that the increase in frequency, or Lambda (λ), results in a need to increase the order of the approximation to reach the desired accuracy. The RMS error percentage calculated in Table 7, where $\lambda = 2$, shows that the B-Spline Collocation Method behaves very similarly to the over damped solution, due to the low frequency. The calculated error results show that it will take a fifth order approximation with no intermediate knot points, a third order approximation with 4 internal knot points or a fourth order approximation with 6 internal knot points to get the RMS error percentage below 0.5 % .

Lambda = 2		
Order	# of Internal Knot Points	RMS Error %
3	0	4.65
4	0	2.18
5	0	0.15
3	1	1.33
3	2	0.87
3	3	0.56
3	4	0.39
4	1	1.4
4	2	1.5
4	3	1.04
4	4	0.74
4	5	0.54
4	6	0.42

Table 7 Under Damped Solution Comparison for $\lambda = 2$

For the $\lambda = 4$ problem, it can be seen in Table 8, that it takes a seventh order approximation, with no internal knot points, to get the RMS error below 0.5 % for a third order under damped solution. In addition, it takes a thirty one internal knot points for the

third order approximation and forty three internal knot points for a fourth order approximation to reach the RMS benchmark.

Lambda = 4		
Order	# of Internal Knot Points	RMS Error %
3	0	33.7
4	0	32
5	0	6.05
6	0	11.1
7	0	0.31
3	25	2.39
3	26	0.696
3	27	0.537
3	28	0.678
3	29	2.52
3	30	0.987
3	31	0.477
4	37	1.43
4	38	0.7
4	39	0.655
4	40	1.04
4	41	14
4	42	0.85
4	43	0.5
5	1	0.69
5	2	0.42

Table 8 Under Damped Solution Comparison for $\lambda = 4$

As stated previously, the most efficient way to approximate the solution is to find the right balance of the approximation order, number of internal knot points and computation time. Table 9 shows the computation times for the approximations that meet the accuracy goals. Notice the large increase in computation time when a large number of internal knot points are needed to get the desired accuracy.

Lambda = 4		
Order	# of Internal Knot Points	CPU Time (sec)
7	0	2.1
3	31	71.3
4	43	270.4
5	2	5.36

Table 9 Under Damped Solution Time Comparison for $\lambda = 4$

For this frequency, the options are to use either a fifth order approximation with two internal knot points or a seventh order with no internal knot points. Since the accuracies are similar

and both computation times are minimal, either solution would be acceptable. If spurious oscillations are a concern, the fifth order with the two internal knot points is the most balanced solution.

When the frequency increases substantially, as in the $\lambda = 16$ problem shown in Table 10, the approximation will require a very high order B-spline to get the needed accuracy. Remember that this data was taken from a Second Order ODE and it will take a Twenty First Order B-spline to approximate the solution, if no internal knot points are used. As discussed previously, this will make the approximation very susceptible to spurious oscillations that could dramatically affect the results.

Lambda = 16		
Order	# of Internal Knot Points	RMS Error %
14	0	156.6
15	0	198
16	0	30.4
17	0	10.5
18	0	20.6
19	0	1.02
20	0	0.62
21	0	0.064

Table 10 Under Damped Solution Comparison for $\lambda = 16$

Table 11 shows the computation times for some approximations that meet the accuracy goals.

Lambda = 16			
Order	# of Internal Knot Points	RMS Error %	CPU Time (sec)
6	43	0.478	1000.95
21	0	0.064	17.6

Table 11 Under Damped Solution Time Comparison for $\lambda = 16$

If spurious oscillations are a concern, neither of these solutions is very attractive. The insertion of intermediate knot points will reduce the order of the approximation, but in this example they dramatically increase the computation time. Therefore it is recommended that enough intermediate points are used in the knot vector to decrease order of the approximation, while computing the solution in a reasonable amount of time.

4 Two-Dimensional B-spline Collocation

The two dimensional B-spline collocation method involves the determination of knot vectors in two directions and control points such that the differential equation is satisfied to some tolerance at a finite set of collocation points along both axis. These collocation points will reside on the Cartesian abscissa and ordinate axis. For this thesis, the collocation points have been selected to be located at the Greville Abscissa point along the abscissa and ordinate axis. In order to approximate the solution to two dimensional problems, a few B-spline curve components need to be made multi-directional. They are:

1. Directional Parameters
2. Knot Vectors
3. Basis Functions
4. Derivatives

First, now that there are multiple directions, there needs to be multiple directional parameters. In the one dimensional problems, the parameter‘t’ is used and can be extended to any vectored direction, usually Cartesian ‘x’. In the two dimensional problems, the parameters ‘u’ and ‘w’ are introduced and like the one dimensional problems they can be extended in any vectored direction. The only constraint is that the two vectored directions must be orthogonal. For use in this thesis the parameter ‘u’ will be assigned to the Cartesian abscissa axis and the parameter ‘w’ will be assigned to the Cartesian ordinate axis. With these parameters, the formula for a B-spline surface becomes:

$$Q(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j} N_{i,k}(u) M_{j,l}(w) \qquad (16)$$

where both $N_{i,k}(u)$ and $M_{j,l}(w)$ are now the basis functions and $B_{i,j}$ is a three dimensional position vector for the defining polygon.

Second, each direction will have its own knot vector. The basis functions for each direction can be any desired order and are not tied to the other direction in any way. Similar to the one-dimensional B-spline curves, the knot vector chosen must provide sufficient resolution to approximate the solution of the mathematical problem. As previously discussed, open-uniform knot vectors are used exclusively to calculate B-spline surfaces and basis functions in this thesis.

Next, the multiple directions required to approximate a two dimensional solution requires multiple basis functions. The Cox-deBoor recursion formulas defining the basis functions for the B-spline surface become:

$$N_{i,1}(u) = \begin{cases} 1 & x_i \le u < x_{i+1} \\ 0 & otherwise \end{cases} \tag{17}$$

$$N_{i,k}(u) = \frac{(u - x_i)N_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u)N_{i+1,k-1}(u)}{x_{i+k} - x_{i+1}} \tag{18}$$

$$M_{j,1}(w) = \begin{cases} 1 & y_j \le w < y_{j+1} \\ 0 & otherwise \end{cases} \tag{19}$$

$$M_{j,l}(w) = \frac{(w - y_j)M_{j,l-1}(w)}{y_{j+l-1} - y_j} + \frac{(y_{j+l} - w)M_{j+1,l-1}(w)}{y_{j+l} - y_{j+1}} \tag{20}$$

Finally, similar to B-spline curves, the derivatives of the B-spline surfaces must be calculated and substituted for the equivalent order derivatives in the differential equation. For instance, the first derivative of the B-spline surface with respect to the parameter 'u' is:

$$Q_u(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j} N'_{i,k}(u) M_{j,l}(w) \tag{21}$$

To illustrate the process, the first derivative with respect to the parameter 'u' for a k = 3 and l = 3 basis function will be summed. Once all of the terms are accounted for, the derivative with respect to the parameter 'u' will become:

$$Q_u(u,w) = N'_{1,3}[B_{1,1}M_{1,3} + B_{1,2}M_{2,3} + B_{1,3}M_{3,3}] + N'_{2,3}[B_{2,1}M_{1,3} + B_{2,2}M_{2,3} + B_{2,3}M_{3,3}] +$$

$$N'_{3,3}[B_{3,1}M_{1,3} + B_{3,2}M_{2,3} + B_{3,3}M_{3,3}]$$

Similarly, the first derivative of the B-spline surface with respect to the parameter 'w' is:

$$Q_w(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j}N_{i,k}(u)M`_{j,l}(w) \tag{22}$$

If the calculation of the first derivative with respect to both parameters 'u' and 'w' is necessary, the differentiation results in:

$$Q_{uw}(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j}N'_{i,k}(u)M`_{j,l}(w) \tag{23}$$

For higher order problems, the second derivatives can be calculated to show that the derivatives with respect to 'u' and 'w' respectively are:

$$Q_{uu}(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j}N''_{i,k}(u)M_{j,l}(w) \tag{24}$$

$$Q_{ww}(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j}N_{i,k}(u)M``_{j,l}(w) \tag{25}$$

The application of the two dimensional B-spline collocation method towards solving a boundary value problem or differential equation has a few very distinct steps. They are:

1. Select the appropriate knot vectors for each direction
2. Calculate all required basis functions, using equation (2) and (3)
3. Calculate abscissa and ordinate coordinates for the required control points, using the Greville Abscissa equation (4)
4. Calculate the B-spline surface equations and the required B-spline surface derivatives
5. If necessary, use the boundary conditions to solve for boundary applicate, or 'z' coordinate, values of the control points

6. Substitute B-spline surfaces and the derivatives into differential equation or boundary value problem
7. Calculate the remaining interior applicate coordinates of the control points by evaluation of the differential equation at the corresponding abscissa and ordinates of the collocation points

4.1 Example 4

Second Order ODE with Temperature Boundary Conditions, Approximated with a Third Order B-spline, in Two Directions, with No Intermediate Knot Vector Points

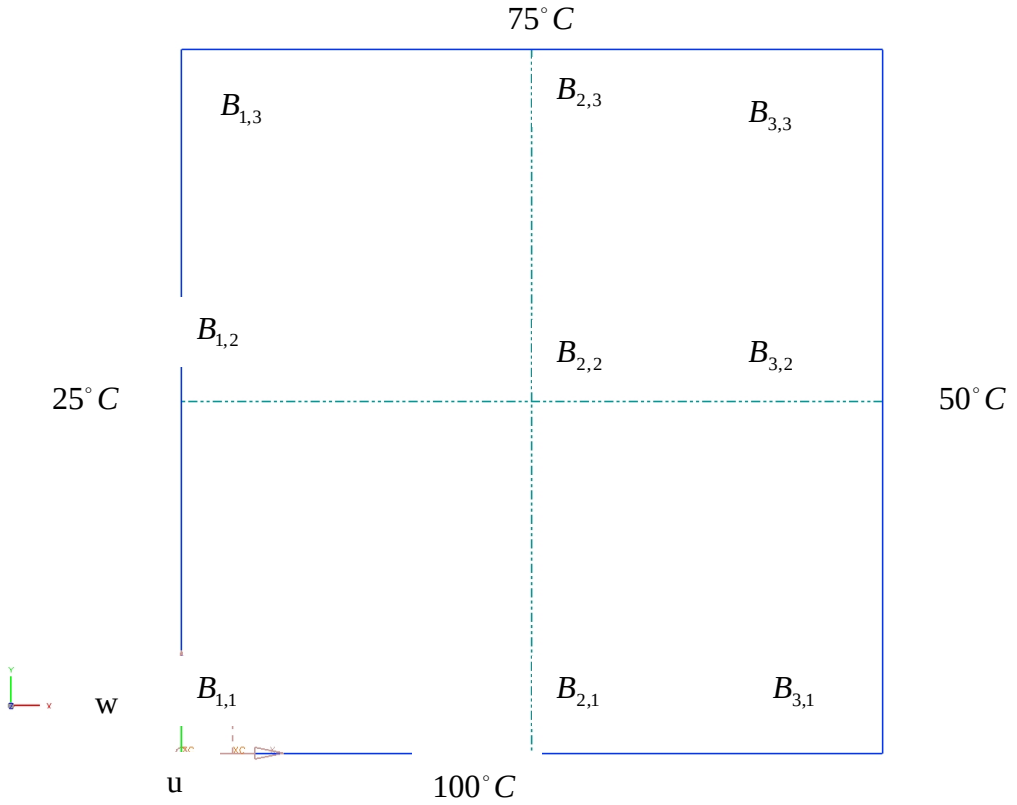

Figure 17 Plane Heat Transfer Problem

The governing equation for a two dimensional heat transfer problem with no heat generation is the Laplace Equation, $\frac{\partial^2 Q}{\partial x^2} + \frac{\partial^2 Q}{\partial y^2} = 0$, where Q is the temperature change in the indicated direction.

Boundary Conditions: $Q(0,w) = 25°C$; $Q(u,0) = 100°C$;

$Q(1,w) = 50°C$; $Q(u,1) = 75°C$

1. The first step in the B-spline collocation method is to select the appropriate knot vectors that provides the resolution required for the problem. A third order, open-uniform knot vector has been selected for each direction.

   X = [0 0 0 1 1 1]

   Y = [0 0 0 1 1 1]

2. The second step is to calculate all required basis functions using equations (17), (18), (19) and (20).

$$N_{i,1}(u) = \begin{cases} 1 & x_i \le u < x_{i+1} \\ 0 & otherwise \end{cases} \qquad N_{i,k}(u) = \frac{(u - x_i)N_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u)N_{i+1,k-1}(u)}{x_{i+k} - x_{i+1}}$$

$$M_{j,1}(w) = \begin{cases} 1 & y_j \le w < y_{j+1} \\ 0 & otherwise \end{cases} \qquad M_{j,l}(w) = \frac{(w - y_j)M_{j,l-1}(w)}{y_{j+l-1} - y_j} + \frac{(y_{j+l} - w)M_{j+1,l-1}(w)}{y_{j+l} - y_{j+1}}$$

$N_{1,1}(u) = 0$; $N_{2,1}(u) = 0$; $N_{3,1}(u) = 1$; $N_{4,1}(u) = 0$;

$N_{1,2}(u) = 0$; $N_{2,2}(u) = 1 - u$; $N_{3,2}(u) = u$; $N_{4,2}(u) = 0$;

$N_{1,3}(u) = (1-u)^2$; $N_{2,3}(u) = 2u(1-u)$; $N_{3,3}(u) = u^2$; $N_{4,3}(u) = 0$

$M_{1,1}(w) = 0$; $M_{2,1}(w) = 0$; $M_{3,1}(w) = 1$; $M_{4,1}(w) = 0$;

$M_{1,2}(w) = 0$; $M_{2,2}(w) = 1 - w$; $M_{3,2}(w) = w$; $M_{4,2}(w) = 0$;

$M_{1,3}(w) = (1-w)^2$; $M_{2,3}(w) = 2w(1-w)$; $M_{3,3}(w) = w^2$; $M_{4,3}(w) = 0$

By plotting the k = 3 basis functions, shown in Figure 18, the generated curves can be used to verify that the calculated basis functions comply with the criteria discussed in the previous chapters.

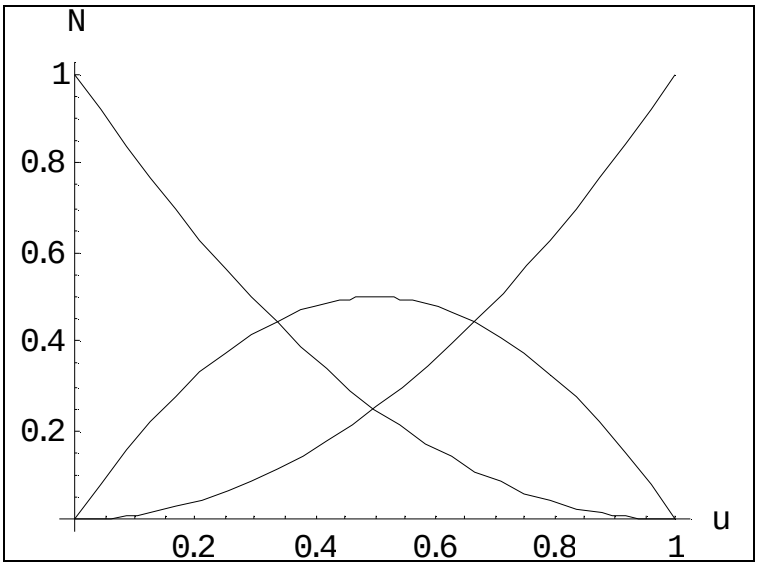

Figure 18 Basis Functions for the 'u' Parameter

Likewise, by plotting the l = 3 basis functions, shown in Figure 19, the generated curves can be used to verify that the calculated basis functions comply with the criteria previously discussed in Chapter 2.

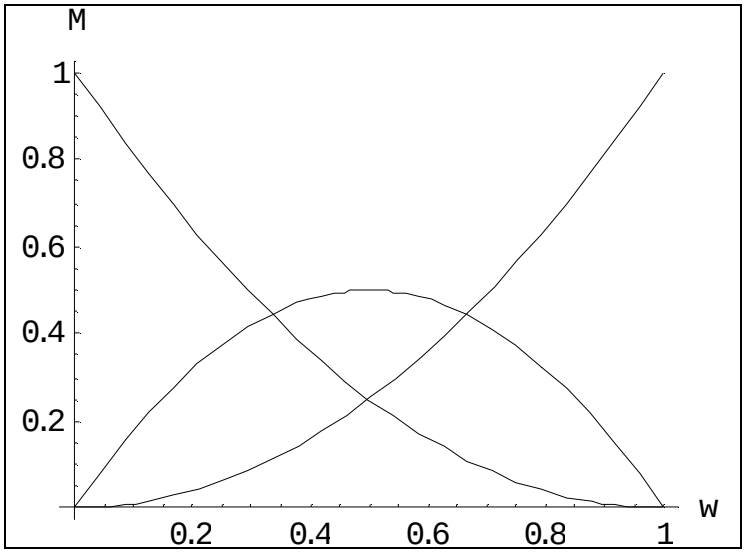

Figure 19 Basis Functions for the 'w' Parameter

In Figures 18 and 19, it can be seen that indeed they do meet the required criteria and it is safe to move on to the next step.

3. The next step is to calculate the abscissa and ordinate coordinates for the control points. Since the Greville Abscissa approach has been chosen, use equation (4) to calculate the Greville Abscissa for required control points.

$$x_i = \frac{1}{n}(t_i + t_{i+1} + ... + t_{i+n-1}) \qquad \text{For i=0, 1,…,m-n}$$

$$x_0 = \frac{1}{2}(t_0 + t_1) = \frac{1}{2}(0+0) = 0$$

$$x_1 = \frac{1}{2}(t_1 + t_2) = \frac{1}{2}(0+0) = 0$$

$$x_2 = \frac{1}{2}(t_2 + t_3) = \frac{1}{2}(0+1) = \frac{1}{2}$$

$$x_3 = \frac{1}{2}(t_3 + t_4) = \frac{1}{2}(1+1) = 1$$

$$x_4 = \frac{1}{2}(t_4 + t_5) = \frac{1}{2}(1+1) = 1$$

Now calculate all ordinate values for required control points, using equation (4).

$$y_i = \frac{1}{n}(t_i + t_{i+1} + ... + t_{i+n-1}) \qquad \text{For i=0, 1,…,m-n}$$

$$y_0 = \frac{1}{2}(t_0 + t_1) = \frac{1}{2}(0+0) = 0$$

$$y_1 = \frac{1}{2}(t_1 + t_2) = \frac{1}{2}(0+0) = 0$$

$$y_2 = \frac{1}{2}(t_2 + t_3) = \frac{1}{2}(0+1) = \frac{1}{2}$$

$$y_3 = \frac{1}{2}(t_3 + t_4) = \frac{1}{2}(1+1) = 1$$

$$y_4 = \frac{1}{2}(t_4 + t_5) = \frac{1}{2}(1+1) = 1$$

As discussed in the previous chapters, the repeating values of $x_0 = 0$ and $x_4 = 1$ should be dropped. That leaves $x_1 = 0$, $x_2 = \frac{1}{2}$ and $x_3 = 1$ as the abscissa coordinates for the

control points. Likewise, the repeating values of $y_0 = 0$ and $y_4 = 1$ should be dropped. That leaves $y_1 = 0$, $y_2 = \frac{1}{2}$ and $y_3 = 1$ as the ordinate coordinates for the control points.

4. The fourth step is to calculate the appropriate B-spline surface equations and the required derivatives for the boundary conditions that need to be met. Since the boundary conditions involve the B-spline surface, it must be calculated. The B-spline surface must be calculated, by using equation (16).

$$Q(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j} N_{i,k}(u) M_{j,l}(w)$$

$$Q(u,w) = N_{1,3}(u)[B_{1,1}M_{1,3}(w) + B_{1,2}M_{2,3}(w) + B_{1,3}M_{3,3}(w)] +$$

$$N_{2,3}(u)[B_{2,1}M_{1,3}(w) + B_{2,2}M_{2,3}(w) + B_{2,3}M_{3,3}(w)] +$$

$$N_{3,3}(u)[B_{3,1}M_{1,3}(w) + B_{3,2}M_{2,3}(w) + B_{3,3}M_{3,3}(w)]$$

$$\begin{aligned} Q(u,w) = {} & (1-u)^2[B_{1,1} * (1-w)^2 + B_{1,2} * 2w(1-w) + B_{1,3} * w^2] + \\ & 2u(1-u)[B_{2,1} * (1-w)^2 + B_{2,2} * 2w(1-w) + B_{2,3} * w^2] + \\ & u^2[B_{3,1} * (1-w)^2 + B_{3,2} * 2w(1-w) + B_{3,3} * w^2] \end{aligned}$$

The first term of the differential equation involves the second derivative in the parameter ‘u’ direction. The second derivative of the B-spline surface, with respect to the parameter ‘u’ can be calculated using equation (24).

$$Q_{uu}(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j} N''_{i,k}(u) M_{j,l}(w)$$

$$Q_{uu}(u,w) = N''_{1,3}(u)[B_{1,1}M_{1,3}(w) + B_{1,2}M_{2,3}(w) + B_{1,3}M_{3,3}(w)] +$$

$$N''_{2,3}(u)[B_{2,1}M_{1,3}(w) + B_{2,2}M_{2,3}(w) + B_{2,3}M_{3,3}(w)] +$$

$$N''_{3,3}(u)[B_{3,1}M_{1,3}(w) + B_{3,2}M_{2,3}(w) + B_{3,3}M_{3,3}(w)]$$

$$Q_{uu}(u,w) = 2[B_{1,1} * (1-w)^2 + B_{1,2} * 2w(1-w) + B_{1,3} * w^2] -$$

$$4[B_{2,1} * (1-w)^2 + B_{2,2} * 2w(1-w) + B_{2,3} * w^2] +$$

$$2[B_{3,1} * (1-w)^2 + B_{3,2} * 2w(1-w) + B_{3,3} * w^2]$$

The second term of the differential equation involves the second derivative in the parameter 'w' direction. The second derivative of the B-spline surfae, with respect to the parameter 'w' can be calculated using equation (25).

$$Q_{ww}(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j} N_{i,k}(u) M``_{j,l}(w)$$

$$Q_{ww}(u,w) = N_{1,3}(u)[B_{1,1}M``_{1,3}(w) + B_{1,2}M``_{2,3}(w) + B_{1,3}M``_{3,3}(w)] +$$

$$N_{2,3}(u)[B_{2,1}M``_{1,3}(w) + B_{2,2}M``_{2,3}(w) + B_{2,3}M``_{3,3}(w)] +$$

$$N_{3,3}(u)[B_{3,1}M``_{1,3}(w) + B_{3,2}M``_{2,3}(w) + B_{3,3}M``_{3,3}(w)]$$

$$Q_{ww}(u,w) = (1-u)^2[B_{1,1} * 2 - B_{1,2} * 4 + B_{1,3} * 2] +$$

$$2u(1-u)[B_{2,1} * 2 - B_{2,2} * 4 + B_{2,3} * 2] +$$

$$u^2[B_{3,1} * 2 - B_{3,2} * 4 + B_{3,3} * 2]$$

5. The next step is to use the boundary conditions to solve for boundary applicate values of control points. Since the Greville Abscissa were used as control points, the parametric coordinate 'u' is constrained to the Cartesian coordinate 'x' and the parametric coordinate 'w' is constrained to the Cartesian coordinate 'y'. The Boundary Conditions are $Q(0,w) = 25°C$, $Q(u,0) = 100°C$, $Q(1,w) = 50°C$ and $Q(u,1) = 75°C$. When the B-spline surface,

$$Q(u,w) = (1-u)^2[B_{1,1} * (1-w)^2 + B_{1,2} * 2w(1-w) + B_{1,3} * w^2] +$$

$$2u(1-u)[B_{2,1} * (1-w)^2 + B_{2,2} * 2w(1-w) + B_{2,3} * w^2] +$$

$$u^2[B_{3,1} * (1-w)^2 + B_{3,2} * 2w(1-w) + B_{3,3} * w^2],$$

is evaluated at the boundaries, the following external applicate values are calculated to be:

At $[0,0,B_{1,1}] = 62.5°$ ➔ $B_{1,1} = z_1 = 62.5$

At $[1/2,0,B_{2,1}] = 100°$ ➔ $\frac{B_{1,1}}{4} + \frac{B_{2,1}}{2} + \frac{B_{3,1}}{4} = 100$

At $[1,0,B_{3,1}] = 75°$ ➔ $B_{3,1} = z_3 = 75$

At $[0,1/2,B_{1,2}] = 25°$ ➔ $\frac{B_{1,1}}{4} + \frac{B_{1,2}}{2} + \frac{B_{1,3}}{4} = 25$

At $[1,1/2,B_{3,2}] = 50°$ ➔ $\frac{B_{3,1}}{4} + \frac{B_{3,2}}{2} + \frac{B_{3,3}}{4} = 50$

At $[0,1,B_{1,3}] = 50°$ ➔ $B_{1,3} = z_7 = 50$

At $[1/2,1,B_{2,3}] = 75°$ ➔ $\frac{B_{1,3}}{4} + \frac{B_{2,3}}{2} + \frac{B_{3,3}}{4} = 75$

At $[1,1,B_{3,3}] = 62.5°$ ➔ $B_{3,3} = z_9 = 62.5$

$B_{1,1} = 62.5$;	$B_{2,1} = 131.25$;	$B_{3,1} = 75$;	$B_{1,2} = -6.25$;
$B_{3,2} = 31.25$;	$B_{1,3} = 50$;	$B_{2,3} = 93.75$;	$B_{3,3} = 62.5$

6. The sixth step is to substitute B-spline surface and the derivatives into the differential equation, $\frac{\partial^2 Q}{\partial x^2} + \frac{\partial^2 Q}{\partial y^2} = 0$ ➔ $Q_{uu}(u,w) + Q_{ww}(u,w) = 0$

$$2[B_{1,1}*(1-w)^2 + B_{1,2}*2w(1-w) + B_{1,3}*w^2] -$$

$$4[B_{2,1}*(1-w)^2 + B_{2,2}*2w(1-w) + B_{2,3}*w^2] +$$

$$2[B_{3,1}*(1-w)^2 + B_{3,2}*2w(1-w) + B_{3,3}*w^2] + (1-u)^2[B_{1,1}*2 - B_{1,2}*4 + B_{1,3}*2] +$$

$$2u(1-u)[B_{2,1}*2 - B_{2,2}*4 + B_{2,3}*2] + u^2[B_{3,1}*2 - B_{3,2}*4 + B_{3,3}*2] = 0$$

7. The seventh step is to calculate the remaining internal applicate coordinate of the control point by evaluation of the differential equation

$$\text{At } [1/2, 1/2, B_{2,2}] \quad \rightarrow \quad B_{2,2} = z_5 = 109.416$$

Now that all of the coordinates for the control points have been calculated, they can be substituted into the B-spline surface equation, to give the final equation for the B-spline approximation. The B-spline surface approximation equation then becomes:

$$Q(u,w) = (1-u)^2[62.5*(1-w)^2 - 6.25*2w(1-w) + 50*w^2] +$$

$$2u(1-u)[131.25*(1-w)^2 + 109.416*2w(1-w) + 93.75*w^2] +$$

$$u^2[75*(1-w)^2 + 31.25*2w(1-w) + 62.5*w^2].$$

As stated earlier, using the Greville Abscissa allows a direct substitution of the ‘x’ and ’y’ with the ‘u’ and ‘w’ parameters. The B-spline function approximation equation, with respect to the Cartesian coordinate system then becomes:

$$Q(x,y) = (1-x)^2[62.5*(1-y)^2 - 6.25*2y(1-y) + 50*y^2] +$$

$$2x(1-x)[131.25*(1-y)^2 + 109.416*2y(1-y) + 93.75*y^2] +$$

$$x^2[75*(1-y)^2 + 31.25*2y(1-y) + 62.5*y^2].$$

One way of comparing the B-spline approximation results with the exact solution for a two dimensional problem is to contour plot the B-spline solution, as seen in Figure 20, and compare it to an ANSYS solution, shown in Figure 21.

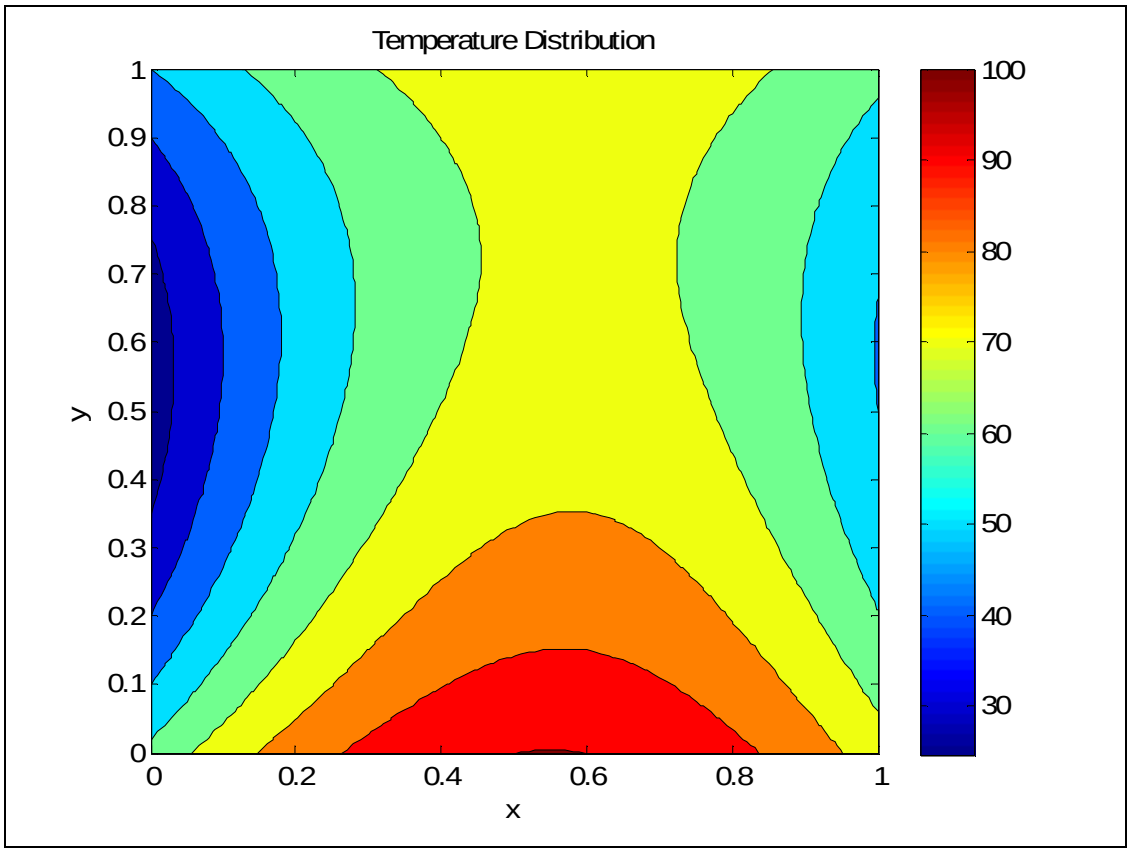

Figure 20 *2-D* B-spline Solution Contour Plot for 3rd Order Approximation

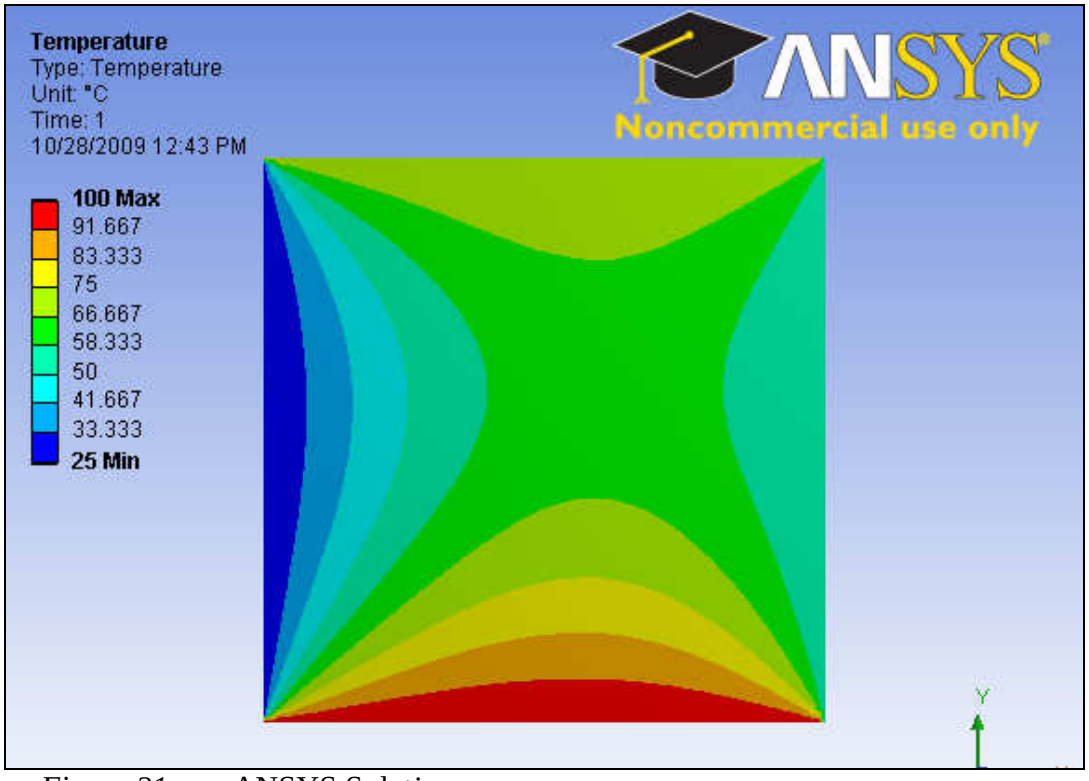

Figure 21 ANSYS Solution

By comparing the two plots in Figures 20 and 21, it can be seen that the third order approximation is not a good solution. The main difference is along the boundaries, where the ANSYS plot has the boundary temperature extending the entire length and the B-spline approximation does not. This is due to the B-spline approximation only satisfying the boundary conditions at one point in the center of the boundary. The ANSYS plot, in Figure 21, uses element type PLANE55 and has an element size of 0.001 mm square. Similarly to the one-dimensional B-spline approximation there are two ways to increase the overall accuracy, increasing the order and increasing the internal knot points.

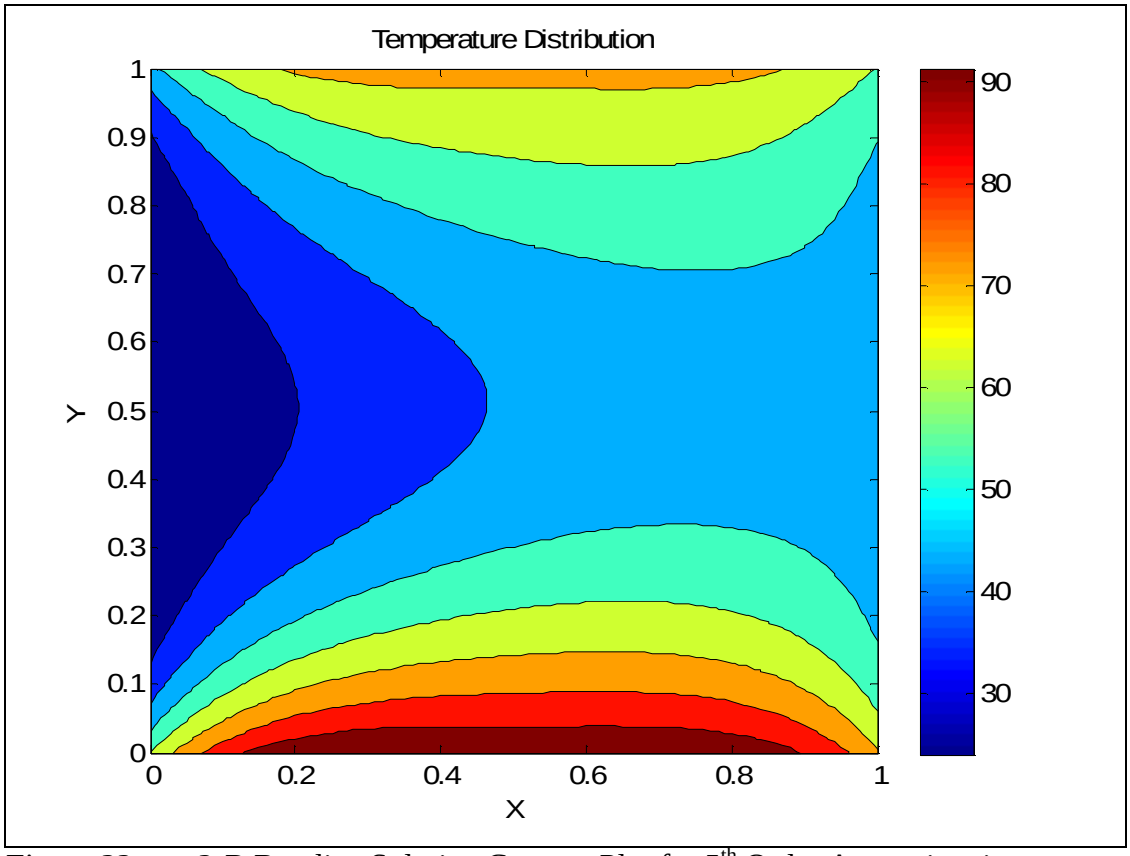

Figure 22 *2-D* B-spline Solution Contour Plot for 5th Order Approximation

Figure 22, shows the 5th order solution plotted for the knot vector [0 0 0 0 0 1 1 1 1 1] in both directions. This contour plot shows the boundary conditions being met at a larger percentage of the boundary lengths, so the approximation is more accurate, as expected.

4.2 Example 5

Second Order ODE with Two Derivative Boundary Conditions, Approximated with a Third Order B-spline, in Two Directions, with No Intermediate Knot Vector Points

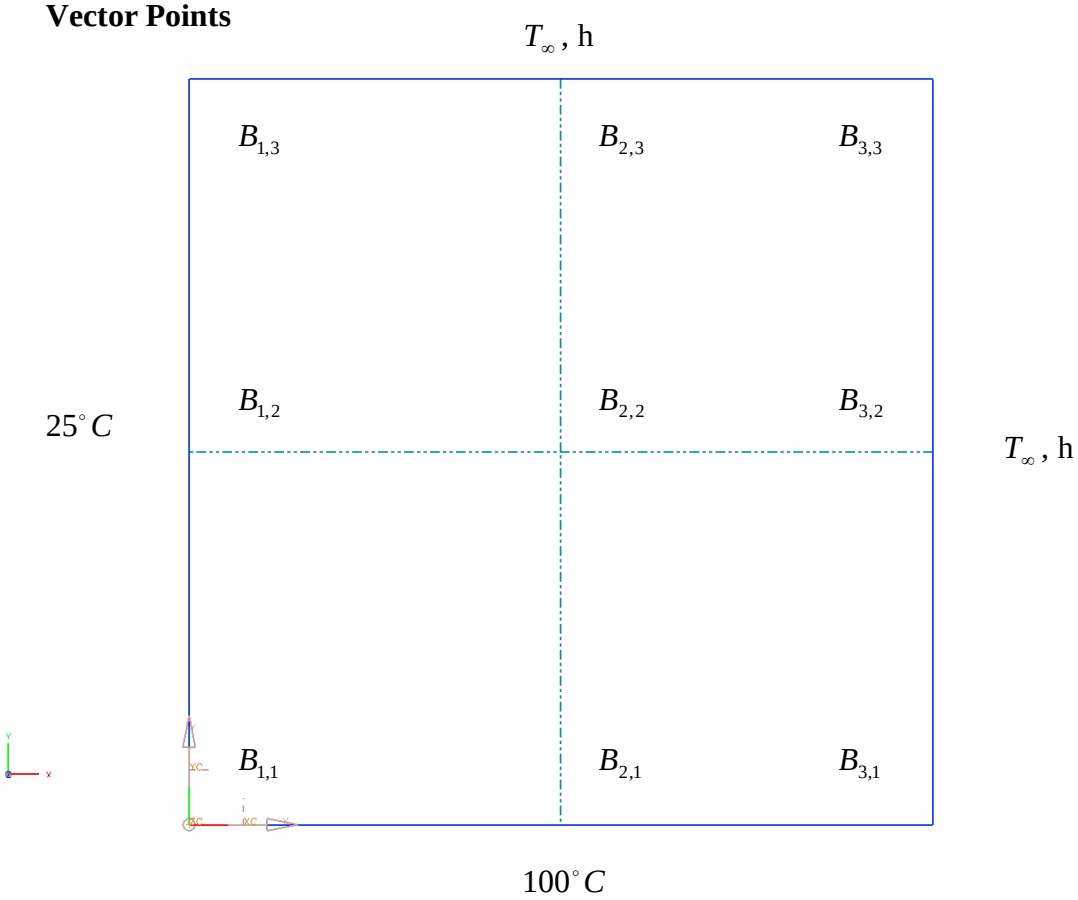

Figure 23 Plane Heat Transfer Problem

As discussed in Example 4, the governing equation for a two dimensional heat transfer problem with no heat generation is the Laplace Equation, $\frac{\partial^2 Q}{\partial x^2} + \frac{\partial^2 Q}{\partial y^2} = 0$.

Boundary Conditions:

$T1 = 25°C$; $T2 = 100°C$; $T_\infty = 200°C$; $h = 1$; $k = 1$

$$-k\frac{\partial Q}{\partial y}\bigg|_{y=L} = h*\left(T(L,t) - T_\infty\right) \qquad -k\frac{\partial Q}{\partial x}\bigg|_{x=L} = h*\left(T(L,t) - T_\infty\right)$$

1. The first step in the B-spline collocation method is to select the appropriate knot vectors that provides the resolution required for the problem. Once again a third order, open-uniform knot vector has been selected for each direction.

   X = [0 0 0 1 1 1]

   Y = [0 0 0 1 1 1]

2. The second step is to calculate all required basis functions using equations (17), (18), (19) and (20). The basis functions for this knot vector were calculated in Example 4 and are:

$$N_{i,1}(u) = \begin{cases} 1 & x_i \le u < x_{i+1} \\ 0 & otherwise \end{cases} \qquad N_{i,k}(u) = \frac{(u - x_i)N_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u)N_{i+1,k-1}(u)}{x_{i+k} - x_{i+1}}$$

$$N_{1,3}(u) = (1-u)^2; \qquad N_{2,3}(u) = 2u(1-u); \quad N_{3,3}(u) = u^2; \qquad N_{4,3}(u) = 0$$

$$M_{j,1}(w) = \begin{cases} 1 & y_j \le w < y_{j+1} \\ 0 & otherwise \end{cases} \qquad M_{j,l}(w) = \frac{(w - y_j)M_{j,l-1}(w)}{y_{j+l-1} - y_j} + \frac{(y_{j+l} - w)M_{j+1,l-1}(w)}{y_{j+l} - y_{j+1}}$$

$$M_{1,3}(w) = (1-w)^2; \quad M_{2,3}(w) = 2w(1-w); \; M_{3,3}(w) = w^2; \qquad M_{4,3}(w) = 0$$

3. The next step is to calculate the abscissa and ordinate coordinates for the control points. Since the Greville Abscissa approach has been chosen, use equation (4) to calculate the Greville Abscissa for required control points. These also were calculated in Example 4. The abscissa coordinates for the control points are; $x_1 = 0$, $x_2 = \frac{1}{2}$ and $x_3 = 1$. The ordinate coordinates for the control points are; $y_1 = 0$, $y_2 = \frac{1}{2}$ and $y_3 = 1$.

4. The fourth step is to calculate the appropriate B-spline surface equations and the required derivatives for the boundary conditions that need to be met. Since the boundary conditions involve the B-spline surface, it must be calculated. The B-spline surface must be calculated, by using equation (16).

$$Q(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j} N_{i,k}(u) M_{j,l}(w)$$

$$Q(u,w) = N_{1,3}(u)[B_{1,1}M_{1,3}(w) + B_{1,2}M_{2,3}(w) + B_{1,3}M_{3,3}(w)] +$$

$$N_{2,3}(u)[B_{2,1}M_{1,3}(w) + B_{2,2}M_{2,3}(w) + B_{2,3}M_{3,3}(w)] +$$

$$N_{3,3}(u)[B_{3,1}M_{1,3}(w) + B_{3,2}M_{2,3}(w) + B_{3,3}M_{3,3}(w)]$$

$$Q(u,w) = (1-u)^2[B_{1,1} * (1-w)^2 + B_{1,2} * 2w(1-w) + B_{1,3} * w^2] +$$

$$2u(1-u)[B_{2,1} * (1-w)^2 + B_{2,2} * 2w(1-w) + B_{2,3} * w^2] +$$

$$u^2[B_{3,1} * (1-w)^2 + B_{3,2} * 2w(1-w) + B_{3,3} * w^2]$$

Since the boundary conditions of the differential equation involve the first derivative with respect to the parameter 'u', the first derivative of the B-spline surface must be calculated, using equation (21). $Q_u(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j} N'_{i,k}(u) M_{j,l}(w)$

$$Q_u(u,w) = N'_{1,3}(u)[B_{1,1}M_{1,3}(w) + B_{1,2}M_{2,3}(w) + B_{1,3}M_{3,3}(w)] +$$

$$N'_{2,3}(u)[B_{2,1}M_{1,3}(w) + B_{2,2}M_{2,3}(w) + B_{2,3}M_{3,3}(w)] +$$

$$N'_{3,3}(u)[B_{3,1}M_{1,3}(w) + B_{3,2}M_{2,3}(w) + B_{3,3}M_{3,3}(w)]$$

$$Q_u(u,w) = 2(u-1)[B_{1,1} * (1-w)^2 + B_{1,2} * 2w(1-w) + B_{1,3} * w^2] +$$

$$(2-4u)[B_{2,1} * (1-w)^2 + B_{2,2} * 2w(1-w) + B_{2,3} * w^2] +$$

$$(2u)[B_{3,1} * (1-w)^2 + B_{3,2} * 2w(1-w) + B_{3,3} * w^2]$$

Since other boundary conditions of the differential equation involve the first derivative with respect to the parameter 'w', the first derivative of the B-spline surface must be calculated, using equation (21).

$$Q_w(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j} N_{i,k}(u) M`_{j,l}(w)$$

$$Q_w(u,w) = N_{1,3}(u)[B_{1,1}M`_{1,3}(w) + B_{1,2}M`_{2,3}(w) + B_{1,3}M`_{3,3}(w)] +$$

$$N_{2,3}(u)[B_{2,1}M`_{1,3}(w) + B_{2,2}M`_{2,3}(w) + B_{2,3}M`_{3,3}(w)] +$$

$$N_{3,3}(u)[B_{3,1}M`_{1,3}(w) + B_{3,2}M`_{2,3}(w) + B_{3,3}M`_{3,3}(w)]$$

$$Q_w(u,w) = (1-u)^2[B_{1,1}(2(w-1)) + B_{1,2}(2-4w) + B_{1,3}(2w)] +$$

$$2u(1-u)[B_{2,1}(2(w-1)) + B_{2,2}(2-4w) + B_{2,3}(2w)] +$$

$$(u^2)[B_{3,1}(2(w-1)) + B_{3,2}(2-4w) + B_{3,3}(2w)]$$

The first term of the differential equation involves the second derivative in the parameter 'u' direction. The second derivative of the B-spline surface, with respect to the parameter 'u' can be calculated using equation (24).

$$Q_{uu}(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j}N^{"}_{i,k}(u)M_{j,l}(w)$$

$$Q_{uu}(u,w) = N^{"}_{1,3}(u)[B_{1,1}M_{1,3}(w) + B_{1,2}M_{2,3}(w) + B_{1,3}M_{3,3}(w)] +$$

$$N^{"}_{2,3}(u)[B_{2,1}M_{1,3}(w) + B_{2,2}M_{2,3}(w) + B_{2,3}M_{3,3}(w)] +$$

$$N^{"}_{3,3}(u)[B_{3,1}M_{1,3}(w) + B_{3,2}M_{2,3}(w) + B_{3,3}M_{3,3}(w)]$$

$$Q_{uu}(u,w) = 2[B_{1,1} * (1-w)^2 + B_{1,2} * 2w(1-w) + B_{1,3} * w^2] -$$

$$4[B_{2,1} * (1-w)^2 + B_{2,2} * 2w(1-w) + B_{2,3} * w^2] +$$

$$2[B_{3,1} * (1-w)^2 + B_{3,2} * 2w(1-w) + B_{3,3} * w^2]$$

The second term of the differential equation involves the second derivative in the parameter 'w' direction. The second derivative of the B-spline surface, with respect to the parameter 'w' can be calculated using equation (25).

$$Q_{ww}(u,w) = \sum_{i=1}^{n+1}\sum_{j=1}^{m+1} B_{i,j}N_{i,k}(u)M``_{j,l}(w)$$

$$Q_{ww}(u,w) = N_{1,3}(u)[B_{1,1}M``_{1,3}(w) + B_{1,2}M``_{2,3}(w) + B_{1,3}M``_{3,3}(w)] +$$

$$N_{2,3}(u)[B_{2,1}M``_{1,3}(w) + B_{2,2}M``_{2,3}(w) + B_{2,3}M``_{3,3}(w)] +$$

$$N_{3,3}(u)[B_{3,1}M``_{1,3}(w) + B_{3,2}M``_{2,3}(w) + B_{3,3}M``_{3,3}(w)]$$

$$Q_{ww}(u,w) = (1-u)^2[B_{1,1} * 2 - B_{1,2} * 4 + B_{1,3} * 2] +$$

$$2u(1-u)[B_{2,1} * 2 - B_{2,2} * 4 + B_{2,3} * 2] +$$

$$u^2[B_{3,1} * 2 - B_{3,2} * 4 + B_{3,3} * 2]$$

5. The next step is to use the boundary conditions to solve for boundary applicate values of control points. Since the Greville Abscissa were used as control points, the parametric coordinate ‘u’ is constrained to the Cartesian coordinate ’x’ and the parametric coordinate ‘w’ is constrained to the Cartesian coordinate ’y’.

$$Q(u,w) = (1-u)^2[B_{1,1} * (1-w)^2 + B_{1,2} * 2w(1-w) + B_{1,3} * w^2] +$$

$$2u(1-u)[B_{2,1} * (1-w)^2 + B_{2,2} * 2w(1-w) + B_{2,3} * w^2] +$$

$$u^2[B_{3,1} * (1-w)^2 + B_{3,2} * 2w(1-w) + B_{3,3} * w^2]$$

- At $[0,0,B_{1,1}] = 62.5°$ ➔ $B_{1,1} = 62.5$
- At $[1/2,0,B_{2,1}] = 100°$ ➔ $\frac{B_{1,1}}{4} + \frac{B_{2,1}}{2} + \frac{B_{3,1}}{4} = 100$ ➔ $\frac{B_{2,1}}{2} + \frac{B_{3,1}}{4} = 84.375$
- At $[0,1/2,B_{1,2}] = 25°$ ➔ $\frac{B_{1,1}}{4} + \frac{B_{1,2}}{2} + \frac{B_{1,3}}{4} = 25$ ➔ $\frac{B_{1,2}}{2} + \frac{B_{1,3}}{4} = 9.375$
- At $[0,1,B_{1,3}] = 25°$ ➔ $B_{1,3} = 25$
- At $[1,0,B_{3,1}] = 100°$ ➔ $B_{3,1} = 100$

$B_{1,1} = 62.5$; $B_{2,1} = 118.75$; $B_{3,1} = 100$; $B_{1,2} = 6.75$; $B_{1,3} = 25$

- At $[1,1/2,B_{3,2}]$ $\quad -k\left.\frac{\partial Q}{\partial u}\right|_{u=1,w=1/2} = h*\left(Q(1,1/2)-T_\infty\right)$

$$T_\infty - \left.\frac{\partial Q}{\partial u}\right|_{u=1,w=1/2} = Q(1,1/2)$$

$$T_\infty - [2(u-1)[B_{1,1}*(1-w)^2 + B_{1,2}*2w(1-w) + B_{1,3}*w^2] +$$

$$(2-4u)[B_{2,1}*(1-w)^2 + B_{2,2}*2w(1-w) + B_{2,3}*w^2] +$$

$$(2u)[B_{3,1}*(1-w)^2 + B_{3,2}*2w(1-w) + B_{3,3}*w^2]] =$$

$$(1-u)^2[B_{1,1}*(1-w)^2 + B_{1,2}*2w(1-w) + B_{1,3}*w^2] +$$

$$2u(1-u)[B_{2,1}*(1-w)^2 + B_{2,2}*2w(1-w) + B_{2,3}*w^2] +$$

$$u^2[B_{3,1}*(1-w)^2 + B_{3,2}*2w(1-w) + B_{3,3}*w^2]$$

$$T_\infty - (-2)*\left[\frac{B_{2,1}}{4}+\frac{B_{2,2}}{2}+\frac{B_{2,3}}{4}\right] + 2*\left[\frac{B_{3,1}}{4}+\frac{B_{3,2}}{2}+\frac{B_{3,3}}{4}\right] = \left(\frac{B_{3,1}}{4}+\frac{B_{3,2}}{2}+\frac{B_{3,3}}{4}\right)$$

$$\boxed{-B_{2,2}+\frac{3*B_{3,2}}{2}-\frac{B_{2,3}}{2}+\frac{3*B_{3,3}}{4}=184.375}$$

- At $[1/2,1,B_{2,3}]$ $\quad -k\left.\frac{\partial Q}{\partial w}\right|_{w=1,u=1/2} = h*\left(Q(1/2,1)-T_\infty\right)$

$$T_\infty - \left.\frac{\partial Q}{\partial w}\right|_{w=1,u=1/2} = Q(1/2,1)$$

$$T_\infty - [(1-u)^2[B_{1,1}(2(w-1)) + B_{1,2}(2-4w) + B_{1,3}(2w)] +$$

$$2u(1-u)[B_{2,1}(2(w-1)) + B_{2,2}(2-4w) + B_{2,3}(2w)] +$$

$$(u^2)[B_{3,1}(2(w-1)) + B_{3,2}(2-4w) + B_{3,3}(2w)]] =$$

$$(1-u)^2[B_{1,1}*(1-w)^2 + B_{1,2}*2w(1-w) + B_{1,3}*w^2] +$$

$$2u(1-u)[B_{2,1}*(1-w)^2 + B_{2,2}*2w(1-w) + B_{2,3}*w^2] +$$

$$u^2[B_{3,1}*(1-w)^2 + B_{3,2}*2w(1-w) + B_{3,3}*w^2]$$

$$T_\infty - \left[\frac{1}{4}\left(-2B_{1,2}+2B_{1,3}\right)+\frac{1}{2}\left(-2B_{2,2}+2B_{2,3}\right)+\frac{1}{4}\left(-2B_{3,2}+2B_{3,3}\right)\right]=\left(\frac{B_{1,3}}{4}+\frac{B_{2,3}}{2}+\frac{B_{3,3}}{4}\right)$$

$$\boxed{-B_{2,2}-\frac{B_{3,2}}{2}+\frac{3*B_{2,3}}{2}+\frac{3*B_{3,3}}{4}=184.625}$$

- At $[1,1,B_{3,3}]$ $\quad -k\left.\frac{\partial Q}{\partial w}\right|_{w=1,u=1} - k\left.\frac{\partial Q}{\partial u}\right|_{u=1,w=1} = h*\left(Q(1,1)-T_\infty\right)$

$$T_\infty - \left[\left.\frac{\partial Q}{\partial w}\right|_{w=1,u=1} + \left.\frac{\partial Q}{\partial u}\right|_{u=1,w=1}\right] = Q(1,1)$$

$$T_\infty - [2(u-1)[B_{1,1}*(1-w)^2 + B_{1,2}*2w(1-w) + B_{1,3}*w^2] +$$

$$(2-4u)[B_{2,1}*(1-w)^2 + B_{2,2}*2w(1-w) + B_{2,3}*w^2] +$$

$$(2u)[B_{3,1}*(1-w)^2 + B_{3,2}*2w(1-w) + B_{3,3}*w^2] +$$

$$[(1-u)^2[B_{1,1}(2(w-1)) + B_{1,2}(2-4w) + B_{1,3}(2w)] +$$

$$2u(1-u)[B_{2,1}(2(w-1)) + B_{2,2}(2-4w) + B_{2,3}(2w)] +$$

$$(u^2)[B_{3,1}(2(w-1)) + B_{3,2}(2-4w) + B_{3,3}(2w)]] =$$

$$(1-u)^2[B_{1,1}*(1-w)^2 + B_{1,2}*2w(1-w) + B_{1,3}*w^2] +$$

$$2u(1-u)[B_{2,1}*(1-w)^2 + B_{2,2}*2w(1-w) + B_{2,3}*w^2] +$$

$$u^2[B_{3,1}*(1-w)^2 + B_{3,2}*2w(1-w) + B_{3,3}*w^2]$$

$$\boxed{-2B_{3,2}-2B_{2,3}+5B_{3,3}=200}$$

6. Substitute B-spline curves and the derivatives into original differential equation

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}=0 \qquad \rightarrow \qquad Q_{uu}(u,w)+Q_{ww}(u,w)=0$$

$$2[B_{1,1}*(1-w)^2 + B_{1,2}*2w(1-w) + B_{1,3}*w^2] -$$

$$4[B_{2,1}*(1-w)^2 + B_{2,2}*2w(1-w) + B_{2,3}*w^2] +$$

$$2[B_{3,1}*(1-w)^2 + B_{3,2}*2w(1-w) + B_{3,3}*w^2] + (1-u)^2[B_{1,1}*2 - B_{1,2}*4 + B_{1,3}*2] +$$

$$2u(1-u)[B_{2,1}*2 - B_{2,2}*4 + B_{2,3}*2] + u^2[B_{3,1}*2 - B_{3,2}*4 + B_{3,3}*2] = 0$$

7. Calculate the remaining applicate coordinates of the control points by evaluation of the differential equation at the corresponding 'x' and 'y' coordinates

- At $[1/2, 1/2, B_{2,2}]$

$$2\left[\frac{B_{1,1}}{4} + \frac{B_{1,2}}{2} + \frac{B_{1,3}}{4}\right] - 4\left[\frac{B_{2,1}}{4} + \frac{B_{2,2}}{2} + \frac{B_{2,3}}{4}\right] + 2\left[\frac{B_{3,1}}{4} + \frac{B_{3,2}}{2} + \frac{B_{3,3}}{4}\right] +$$

$$\frac{1}{4}[B_{1,1}*2 - B_{1,2}*4 + B_{1,3}*2] + \frac{1}{2}[B_{2,1}*2 - B_{2,2}*4 + B_{2,3}*2] + \frac{1}{4}[B_{3,1}*2 - B_{3,2}*4 + B_{3,3}*2] = 0$$

$$\boxed{-4B_{2,2} + B_{3,3} = -187.5}$$

With four equations and four unknowns, use Linear Algebra to solve for the remaining values for the internal applicate coordinates:

$$\begin{bmatrix} -1 & 1.5 & -0.5 & .75 \\ -1 & -0.5 & 1.5 & .75 \\ 0 & -2 & -2 & 5 \\ -4 & 0 & 0 & 1 \end{bmatrix} * \begin{pmatrix} B_{2,2} \\ B_{3,2} \\ B_{2,3} \\ B_{3,3} \end{pmatrix} = \begin{bmatrix} 184.375 \\ 184.625 \\ 200 \\ -187.5 \end{bmatrix}$$

The remaining internal applicate coordinates are calculated to be:

$B_{2,2} = z_5 = 87.0714$ $\qquad$ $B_{3,2} = z_6 = 150.9196$

$B_{2,3} = z_8 = 151.0446$ $\qquad$ $B_{3,3} = z_9 = 160.7857$

Now that all of the coordinates for the control points have been calculated, they can be substituted into the B-spline surface equation, to give the final equation for the B-spline approximation.

$$\boxed{Q(u,w) = 62.5 - 111.5w + 74w^2 + u^2(-75 + 117.05w - 158.36w^2) + u(112.5 + 96.29w + 43.30w^2)}$$

As stated earlier, using the Greville Abscissa allows a direct substitution of the ‘x’ and ’y’ with the ‘u’ and ‘w’ parameters. The B-spline function approximation equation, with respect to the Cartesian coordinate system then becomes:

$$Q(x,y) = 62.5 - 111.5y + 74y^2 + x^2(-75 + 117.05y - 158.36y^2) + x(112.5 + 96.29y + 43.30y^2)$$

Similarly to Example 4, one way of comparing the B-spline approximation results with the exact solution for a two dimensional problem is to contour plot the B-spline solution, as seen in Figure 24, and compare it to an ANSYS solution, shown in Figure 25.

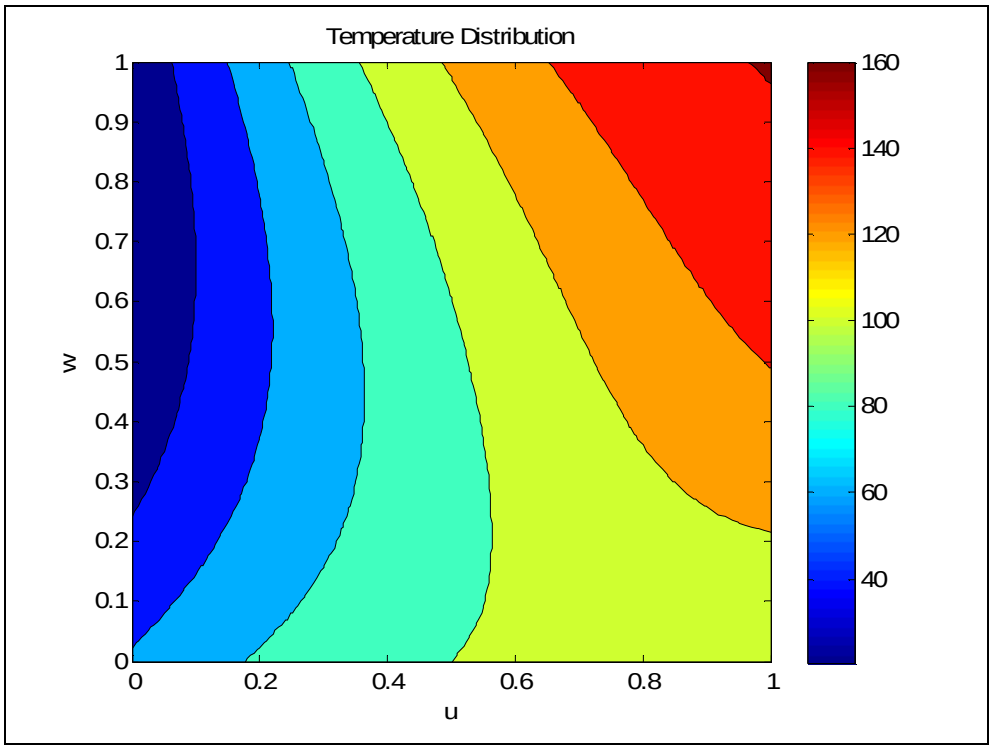

Figure 24 *2-D* B-spline Solution Contour Plot

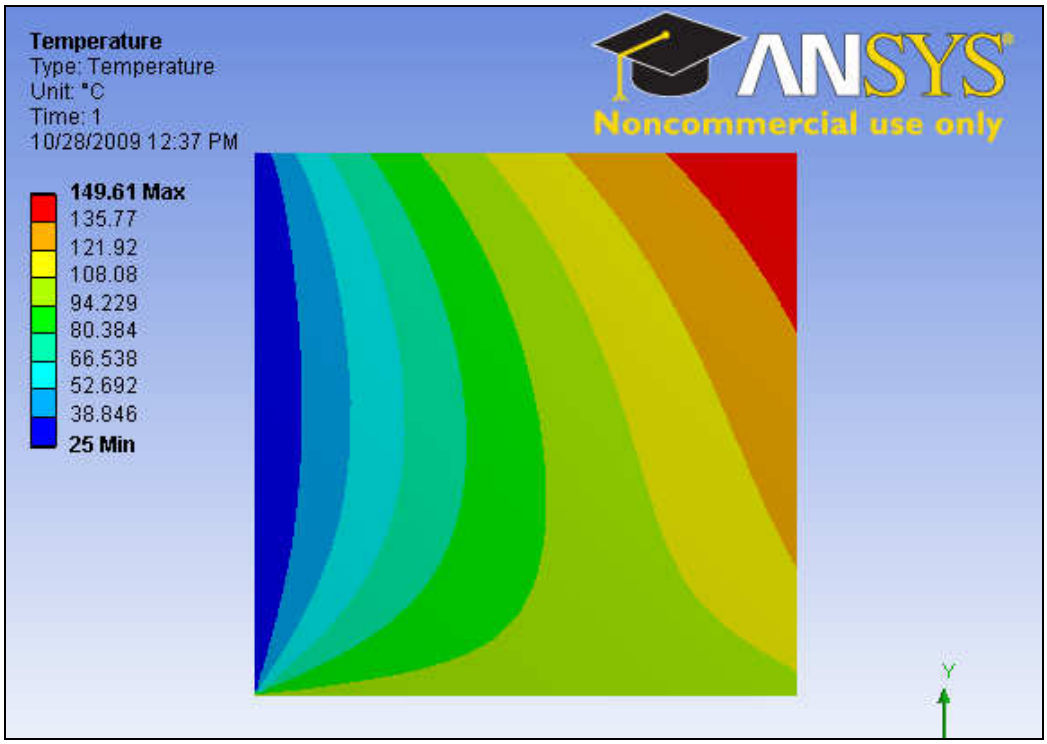

Figure 25 ANSYS Solution

Similarly to Example 4, by comparing the two plots in Figures 24 and 25, it can be seen that the third order approximation is not a good solution. The main difference, once again, is along the boundaries, where the ANSYS plot has the boundary temperature extending the entire length for the constant temperature boundary conditions and the B-spline approximation does not. This is due to the B-spline approximation only satisfying the boundary conditions at one point in the center of the boundary. The ANSYS plot, in Figure 25, uses element type PLANE55 and has an element size of 0.001 mm square. There are two ways to increase the overall accuracy, increasing the order and increasing the internal knot points. Either of these methods will provide an increase in accuracy because the boundary conditions will be satisfied at more points.

5 The Non-Linear Beam Problem

In the field of Mechanical Engineering, a very common type of problem encountered is a beam problem. As discussed in Chapter 3, the governing equation for beam problems is the Euler-Bernoulli beam equation. That is,

$$\frac{d^2y}{dx^2} = \frac{M(x)}{EI}\left(1+\left(\frac{dy}{dx}\right)^2\right)^{3/2} \tag{26}$$

where M(x) is the bending moment as a function of x, E is the Modulus of Elasticity and I is the Moment of Inertia of the cross sectional area about its neutral axis. The term at the end of the Euler-Bernoulli beam equation, $\left(1+\left(\frac{dy}{dx}\right)^2\right)^{3/2}$, makes this Equation (26) non-linear. In most of the beam problems encountered, the deflection of the beam is very small relative to its length and cross section. If the deflection is small, it is assumed that the slope, $\frac{dy}{dx}$ is also very small and its square is negligible compared to unity. This reduces the Euler-Bernoulli equation (26) to:

$$\frac{d^2y}{dx^2} = \frac{M(x)}{EI} \tag{27}$$

The equation is reduced to a linear second order differential equation, and is regarded as the governing differential equation for the elastic curve. With the appropriate boundary and loading conditions, equation (27) becomes the linear beam deflection equations in the back of every Mechanics of Materials textbook.

However, not every beam problem is a linear one. There are three causes for non-linear solutions to beam problems. They are:

1. Material Non-linearity, where the constitutive equation is non-linear

2. Geometric Non-linearity, where the deflection is large enough that the slope assumption previously discussed is invalid
3. Contact or Status change

The type of the non-linearity in the beam problem solved for this thesis is Geometric Non-linearity with large rotation and small strain. Throughout the literature there are several methods used to solve the large rotation, non-linear beam problems. The most prevalent method is by using elliptical integrals to solve the second order non-linear differential equation. This technique will provide a closed form solution, but the derivations are cumbersome and solutions only exist for certain geometries and loadings. Another method of solution is numerical methods, such as the finite element method. Numerical methods can be applied to more general problems and appears to be the preferred method of solution for the most recent papers.

The study of beams in bending dates back to the 13th century and includes some of the most famous names in science and engineering. The likes of Galileo, James and Daniel Bernoulli, Leonhard Euler and Robert Hooke have all worked to solve the elasticity problem, or the Elastica, and lay the foundation for the present day theory. James Bernoulli solved the problem in 1694 by equating the bending moment of the beam to the curvature of a line. His premise was that if the force remained vertical, the moment at any point along the curve is proportional to the distance from the line of force. This work was improved upon by his nephew, Daniel Bernoulli and Leonhard Euler which developed the Euler-Bernoulli Law. This law states that the bending moment at any point is proportional to the change in the curvature caused by the action of the load. Additionally, a series of assumptions are made:

1. Material is linearly elastic
2. Beam is rigid in shear, that is, the cross section does not deform due to shearing stresses

3. Beam has a constant cross section, that is always normal to the beam centerline

The basic formula with respect to curvature is:

$$\frac{1}{r} = \frac{M}{EI} = \left(\frac{\partial \theta}{\partial s}\right) \tag{28}$$

Where θ is the slope and‘s’ is the arc length. For the cantilevered beam with a vertical load at the end, the bending moment is:

$$M = P(L - x) \tag{29}$$

Plugging this moment into the Euler-Bernoulli equation (28) yields the equation:

$$\frac{\partial \theta}{\partial s} = \frac{P}{EI}(L - x) \tag{30}$$

Differentiating equation (30) with respect to the arc length‘s’:

$$\frac{\partial^2 \theta}{\partial s^2} = \frac{P}{EI}\left(\frac{-\partial x}{\partial s}\right) \tag{31}$$

Since $\frac{\partial x}{\partial s} = \cos(\theta)$ ➔ $\boxed{\frac{\partial^2 \theta}{\partial s^2} + \frac{P}{EI}\cos(\theta) = 0}$ (32)

At this point, Frisch-Fay [9] goes through a complex series of differentiation, integration and elliptical integrals to arrive at his solution for the slope at the end of the beam. The B-spline collocation method can be used to simplify the solution to this problem and is worked through, in detail, in Section 5.1, using the same steps previously defined. An additional assumption must be made that the beam geometry is inextensible, that is, the length does not change. In addition, the force at the end remains vertical throughout the bending process, as shown in Figure 26.

5.1 Geometrically Non-Linear Beam Problem

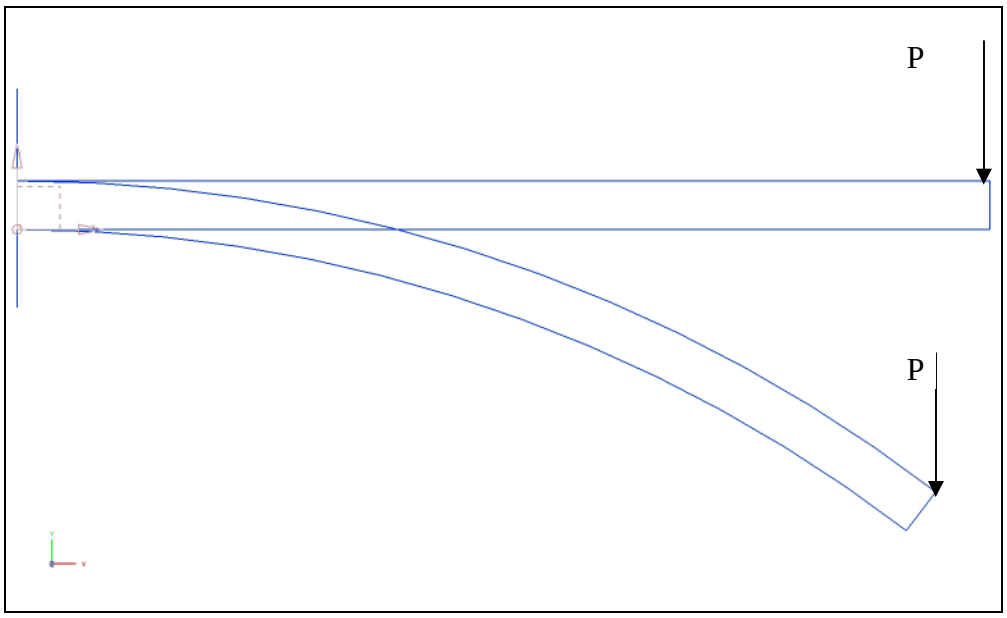

Figure 26 Cantilevered Beam with vertical end loading

The governing equation for the non-linear, cantilevered, end loaded beam is $\frac{\partial^2\theta}{\partial s^2}+\frac{P}{E*I}\cos(\theta)=0$, where E is the Modulus of Elasticity, I is the Moment of Inertia of the cross section about its neutral axis, L is the beam length and P is the end load.

Boundary Conditions: $\theta(0)=0$; $\left.\frac{\partial\theta}{\partial s}\right|_{s=L}=0$

The values given for the variables in the Frisch-Fay solution are:

L = 100 in

E*I = 1000 lb in^2

For the end load, P, there are five different load cases. The load cases are:

Load case 1: P = 0.073 lbs

Load case 2: P = 0.169 lbs

Load case 3: P = 0.3405 lbs

Load case 4: P = 0.8678 lbs

Load case 5: P = 1.00 lbs

The resulting beam slope from each load case must be verified by the B-spline approximation technique to calculate a solution with an error tolerance of < 0.5%. The same procedure for one-dimensional B-spline Collocation outlined in Chapter 3 will be followed.

1. The first step is to select the appropriate knot vector. A fourth order, open-uniform knot vector with one intermediate point has been selected.

   [0 0 0 0 ½ 1 1 1 1]

2. The next step is to calculate all of the required basis functions for each range of values of the parameter 't' using equations (2) and (3).

$$N_{i,1}(t) = \begin{cases} 1 \; x_i \le t < x_{i+1} \\ 0 \;\; otherwise \end{cases} \qquad N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}$$

For $0 \le t < \frac{1}{2}$

$N_{1,1}(t) = 0$; $N_{2,1}(t) = 0$; $N_{3,1}(t) = 0$; $N_{4,1}(t) = 1$;

$N_{1,2}(t) = 0$; $N_{2,2}(t) = 0$; $N_{3,2}(t) = 1 - 2t$; $N_{4,2}(t) = 2t$;

$N_{1,3}(t) = 0$; $N_{2,3}(t) = (1 - 2t)^2$; $N_{3,3}(t) = 2t(2 - 3t)$; $N_{4,3}(t) = 2t^2$

$N_{1,4}(t) = (1 - 2t)^3$; $N_{2,4}(t) = 2t(7t^2 - 9t + 3)$; $N_{3,4}(t) = 2t^2(3 - 4t)$; $N_{4,4}(t) = 2t^3$

For $\frac{1}{2} \le t < 1$

$N_{1,1}(t) = 0$; $N_{2,1}(t) = 0$; $N_{3,1}(t) = 0$; $N_{4,1}(t) = 0$; $N_{5,1}(t) = 1$;

$N_{1,2}(t) = 0$; $N_{2,2}(t) = 0$; $N_{3,2}(t) = 0$; $N_{4,2}(t) = 2 - 2t$; $N_{5,2}(t) = 2t - 1$;

$N_{1,3}(t) = 0$; $N_{2,3}(t) = 0$; $N_{3,3}(t) = 2(t - 1)^2$; $N_{4,3}(t) = -6t^2 + 8t - 2$; $N_{5,3}(t) = (1 - 2t)^2$

$N_{1,4}(t) = 0$; $N_{2,4}(t) = -2(t - 1)^2$; $N_{3,4}(t) = 2(t - 1)^2(4t - 1)$;

$N_{4,4}(t) = -14t^3 + 24t^2 - 12t + 2$; $N_{5,4}(t) = (2t - 1)^3$

By plotting the k = 4 basis functions for each interval together, as shown in Figure 27, it can be seen that all the required criteria is met and it is safe to move on to the next step.

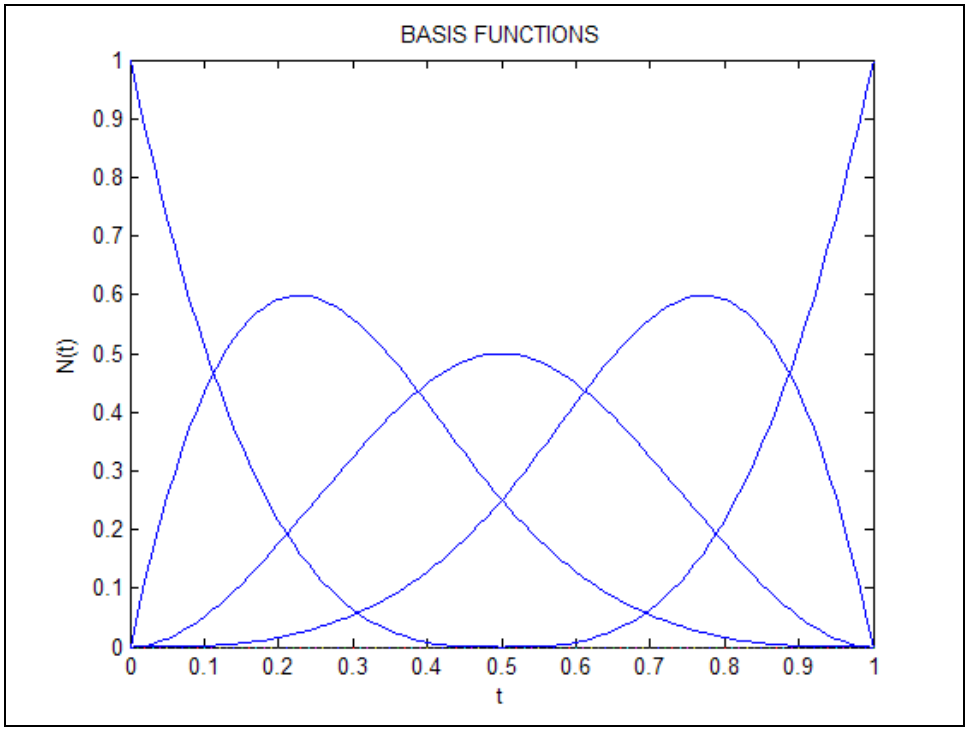

Figure 27 Basis functions for fourth order B-spline approximation with one intermediate Knot Vector points

3. The third step is to calculate all of the abscissa coordinates values for the required control points, using the Greville Abscissa formula, equation (4).

$$x_i = \frac{1}{n}(t_i + t_{i+1} + ... + t_{i+n-1}) \qquad \text{For i=0, 1,…,g-n}$$

where ‘n’ is the degree of the basis functions, or k – 1 and ‘g’ is the total number of knots in the knot vector.

$$x_0 = \frac{1}{3}(t_0 + t_1 + t_2) = \frac{1}{3}(0 + 0 + 0) = 0$$

$$x_1 = \frac{1}{3}(t_1 + t_2 + t_3) = \frac{1}{3}(0 + 0 + 0) = 0$$

$$x_2 = \frac{1}{3}(t_2 + t_3 + t_4) = \frac{1}{3}(0 + 0 + \frac{1}{2}) = \frac{1}{6}$$

$$x_3 = \frac{1}{3}(t_3 + t_4 + t_5) = \frac{1}{3}(0 + \frac{1}{2} + 1) = \frac{1}{2}$$

$$x_4 = \frac{1}{3}(t_4 + t_5 + t_6) = \frac{1}{3}(\frac{1}{2} + 1 + 1) = \frac{5}{6}$$

$$x_5 = \frac{1}{3}(t_5 + t_6 + t_7) = \frac{1}{3}(1 + 1 + 1) = 1$$

$$x_6 = \frac{1}{3}(t_6 + t_7 + t_8) = \frac{1}{3}(1 + 1 + 1) = 1$$

Once again, the two repeating values at the ends should be dropped and that will leave $x_1 = 0$, $x_2 = \frac{1}{6}$, $x_3 = \frac{1}{2}$, $x_4 = \frac{5}{6}$ and $x_5 = 1$ as the abscissa coordinates for the control points.

4. The fourth step is to calculate the appropriate B-spline curve equations and the required derivatives for the boundary conditions that need to be met. Since the second term of the differential equation and one of the boundary conditions involve the B-spline function, it must be calculated for each range of the parameter‘t’. The B-spline curve must be calculated, by using equation (1).

For $0 \le t < \frac{1}{2}$ $\qquad \theta(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$

$$\theta(t) = B_1 N_{1,4}(t) + B_2 N_{2,4}(t) + B_3 N_{3,4}(t) + B_4 N_{4,4}(t)$$

$$\boxed{\theta(t) = B_1\left[(1-2t)^3\right] + B_2\left[2t(7t^2 - 9t + 3)\right] + B_3\left[2t^2(3-4t)\right] + B_4\left[2t^3\right]}$$

For $\frac{1}{2} \le t < 1$ $\qquad \theta(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t)$

$$\theta(t) = B_1 N_{1,4}(t) + B_2 N_{2,4}(t) + B_3 N_{3,4}(t) + B_4 N_{4,4}(t)$$

$$\boxed{\theta(t) = B_2\left[-2(t-1)^3\right] + B_3\left[2t(t-1)^2(4t-1)\right] + B_4\left[-14t^3 + 24t^2 - 12t + 2\right] + B_5\left[(2t-1)^3\right]}$$

Since the second boundary condition involves the first derivative, the first derivative of the B-spline curve must be calculated for each range of the parameter‘t’, using equation (5).

For $0 \le t < \frac{1}{2}$ $\qquad \theta'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t)$

$$\theta'(t) = B_1 N'_{1,4}(t) + B_2 N'_{2,4}(t) + B_3 N'_{3,4}(t) + B_4 N'_{4,4}(t)$$

$$\boxed{\theta'(t) = B_1\left[-6(1-2t)^2\right] + B_2\left[6(7t^2 - 6t + 1)\right] + B_3\left[12t(1-2t)\right] + B_4\left[6t^2\right]}$$

For $\frac{1}{2} \le t < 1$ $\qquad \theta'(t) = \sum_{i=1}^{n+1} B_i N'_{i,k}(t)$

$$\theta'(t) = B_1 N'_{1,4}(t) + B_2 N'_{2,4}(t) + B_3 N'_{3,4}(t) + B_4 N'_{4,4}(t)$$

$$\boxed{\theta'(t) = B_2\left[-6(t-1)^2\right] + B_3\left[12(2t^2 - 3t + 1)\right] + B_4\left[-6(7t^2 - 8t + 2)\right] + B_5\left[6(1-2t)^2\right]}$$

The first term of the differential equation involves the second derivative. The second derivative of the B-spline curve can be calculated for each range of the parameter‘t’, using equation (6).

For $0 \le t < \frac{1}{2}$ $\qquad \theta''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t)$

$$\theta''(t) = B_1 N''_{1,4}(t) + B_2 N''_{2,4}(t) + B_3 N''_{3,4}(t) + B_4 N''_{4,4}(t)$$

$$\boxed{\theta''(t) = B_1\left[24 - 48t\right] + B_2\left[12(7t - 3)\right] + B_3\left[12 - 48t\right] + B_4\left[12t\right]}$$

For $\frac{1}{2} \le t < 1$ $\qquad \theta''(t) = \sum_{i=1}^{n+1} B_i N''_{i,k}(t)$

$$\theta''(t) = B_1 N''_{1,4}(t) + B_2 N''_{2,4}(t) + B_3 N''_{3,4}(t) + B_4 N''_{4,4}(t)$$

$$\boxed{\theta''(t) = B_2\left[-12(t-1)\right] + B_3\left[48t - 36\right] + B_4\left[48 - 84t\right] + B_5\left[48t - 24\right]}$$

5. The next step is to use the boundary conditions to solve for end ordinate values of control points. Since the Greville Abscissa were used as control points, the parametric coordinate ‘t’ is constrained to the Cartesian coordinate ’x’ and it becomes a direct substitution. The

boundary conditions were $\theta(0)=0$ and $\left.\frac{\partial\theta}{\partial s}\right|_{s=L}=0$. Since the B-spline curve is a function over multiple ranges, the appropriate range must be used when calculating the external ordinate values.

For $0 \le t < \frac{1}{2}$ $\qquad \theta(t) = B_1\left[(1-2t)^3\right] + B_2\left[2t(7t^2-9t+3)\right] + B_3\left[2t^2(3-4t)\right] + B_4\left[2t^3\right]$

$\theta(0)=0 \rightarrow \boxed{B_1 = 0 \rightarrow y_1 = 0}$

For $\frac{1}{2} \le t < 1$

$$\theta'(t) = B_2\left[-6(t-1)^2\right] + B_3\left[12(2t^2-3t+1)\right] + B_4\left[-6(7t^2-8t+2)\right] + B_5\left[6(1-2t)^2\right]$$

$\left.\frac{\partial\theta}{\partial s}\right|_{s=L} = 0 \rightarrow \boxed{B_4 = B_5 \rightarrow y_4 = y_5}$

6. The sixth step is to substitute B-spline derivatives into the differential equation, $\frac{\partial^2\theta}{\partial s^2} + \frac{P}{E*I}\cos(\theta) = 0$. We get an equation for each range of the parameter 't'.

For $0 \le t < \frac{1}{2}$

$$\frac{B_2\left[12(7t-3)\right] + B_3\left[12-48t\right] + B_4\left[12t\right]}{L^2} + \frac{P}{E*I}\cos\left(B_2\left[2t(7t^2-9t+3)\right] + B_3\left[2t^2(3-4t)\right] + B_4\left[2t^3\right]\right) = 0$$

For $\frac{1}{2} \le t < 1$

$$\frac{B_2\left[-12(t-1)\right] + B_3\left[48t-36\right] + B_4\left[24-36t\right]}{L^2} +$$

$$\frac{P}{E*I}\cos\left(B_2\left[-2(t-1)^3\right] + B_3\left[2t(t-1)^2(4t-1)\right] + B_4\left[-14t^3+24t^2-12t+2+(2t-1)^3\right]\right) = 0$$

7. The seventh step is to calculate the remaining internal ordinate coordinates of the control points by evaluation of the previously reduced differential equation, using the appropriate

range. Since the length was normalized to provide a Knot Vector with a maximum value of one, the second derivative portion of the approximation must be divided by L^2.

At 't' = $\frac{1}{6}$, using the equation for the range $0 \le t < \frac{1}{2}$,

$$\boxed{\frac{B_2[-22]+B_3[4]+B_4[2]}{L^2}+\frac{P}{E*I}\cos\left(B_2[0.5648]+B_3[0.12963]+B_4[0.009259]\right)=0}$$

At 't' = $\frac{1}{2}$, using the equation for the range $\frac{1}{2} \le t < 1$,

$$\boxed{\frac{B_2[6]+B_3[-12]+B_4[6]}{L^2}+\frac{P}{E*I}\cos\left(B_2[0.25]+B_3[0.50]+B_4[0.25]\right)=0}$$

At 't' = $\frac{5}{6}$, using the equation for the range $\frac{1}{2} \le t < 1$,

$$\boxed{\frac{B_2[2]+B_3[4]+B_4[-6]}{L^2}+\frac{P}{E*I}\cos\left(B_2[0.009259]+B_3[0.12963]+B_4[0.8611]\right)=0}$$

With three non-linear equations and three unknowns, the built in non-linear equation solver in Matlab, fsolve, is used to solve for the remaining ordinate coordinates for each of the five load cases specified. The results are listed here in Table 12.

LOAD CASE	LOAD (lbs)	B2	B3	B4	B5
1	0.073	0.1178	0.2919	0.3486	0.3486
2	0.169	0.2469	0.5948	0.6993	0.6993
3	0.3405	0.4113	0.9220	1.0467	1.0467
4	0.8678	0.6968	1.3187	1.3917	1.3917
5	1	0.8049	1.4119	1.4418	1.4418

Table 12 Ordinate Coordinates Calculated with Matlab 'fsolve' Command

For load case 1, P = 0.073 lbs, the resulting B-spline approximate solution for each range is:

For $0 \le s < \frac{L}{2}$ $\quad\boxed{\theta(s)=\frac{s(14s^2-46125s+8835000)}{1250000000}}$

For $\frac{L}{2} \le s < L$ $\quad\boxed{\theta(s)=\frac{2s^3-9105s^2+1761000s+100000}{250000000}}$

For load case 2, P = 0.169 lbs, the resulting B-spline approximate solution for each range is:

$$\text{For } 0 \le s < \frac{L}{2} \qquad \boxed{\theta(s) = \frac{s(121s^2 - 109425s + 18517500)}{1250000000}}$$

$$\text{For } \frac{L}{2} \le s < L \qquad \boxed{\theta(s) = \frac{86s^3 - 104175s^2 + 18255000s + 4375000}{1250000000}}$$

For load case 3, P = 0.3405 lbs, the resulting B-spline approximate solution for each range is:

$$\text{For } 0 \le s < \frac{L}{2} \qquad \boxed{\theta(s) = \frac{s(1189s^2 - 467850s + 61695000)}{2500000000}}$$

$$\text{For } \frac{L}{2} \le s < L \qquad \boxed{\theta(s) = \frac{683s^3 - 391950s^2 + 57900000s + 63250000}{2500000000}}$$

For load case 4, P = 0.8678 lbs, the resulting B-spline approximate solution for each range is:

$$\text{For } 0 \le s < \frac{L}{2} \qquad \boxed{\theta(s) = \frac{3s(663s^2 - 154340s + 13936000)}{1000000000}}$$

$$\text{For } \frac{L}{2} \le s < L \qquad \boxed{\theta(s) = \frac{3(1343s^3 - 475900s^2 + 54890000s + 246500000)}{5000000000}}$$

For load case 5, P = 1.00 lbs, the resulting B-spline approximate solution for each range is:

$$\text{For } 0 \le s < \frac{L}{2} \qquad \boxed{\theta(s) = \frac{s(2857s^2 - 601680s + 48294000)}{1000000000}}$$

$$\text{For } \frac{L}{2} \le s < L \qquad \boxed{\theta(s) = \frac{5173s^3 - 1641600s^2 + 173130000s + 1139000000}{5000000000}}$$

The slopes at the end of the beam for all five load cases are listed in Table 13. The results listed in Table 13, show that a fourth order, open-uniform knot vector with one intermediate point has the resolution to approximate the geometrically non-linear beam solution for the first four load cases. The fifth load case has an error slightly above the threshold previously set and should be enhanced by adding an intermediate knot vector point or by increasing the B-spline function to a fifth the order approximation.

		SLOPE AT END (rads)		
LOAD CASE	LOAD (lbs)	Frisch-Fay	B-spline Collocation	%ERROR
1	0.073	0.34907	0.3486	1.33E-01
2	0.169	0.69813	0.6993	1.67E-01
3	0.3405	1.0472	1.0467	4.75E-02
4	0.8678	1.39626	1.3917	3.27E-01
5	1	1.43117	1.4418	7.43E-01

Table 13 Comparison of the B-spline solutions versus the Frisch-Fay solutions

In order to get the B-spline Collocation Method solution for Load Case 5 to be below the 0.5% error threshold, a fifth order approximation was calculated. The values for the ordinate coordinates are listed in Table 14.

LOAD CASE	LOAD (lbs)	B2	B3	B4	B5
5	1	0.8079	1.4125	1.4315	1.4315

Table 14 Ordinates Coordinates for Fifth Order B-spline

For load case 5, P = 1.00 lbs, the final resulting B-spline approximation solution is:

$$\theta(s) = \frac{4749}{5e11}s^4 - \frac{1913}{125e7}s^3 - \frac{6099}{5e7}s^2 + \frac{8079}{25e4}s$$

The results listed in Table 15, show that a fifth order, open-uniform knot vector with no intermediate points has the resolution to approximate the geometrically non-linear beam solution for the fifth load case.

		SLOPE AT END (rads)		
LOAD CASE	LOAD (lbs)	Frisch-Fay	B-spline Collocation	%ERROR
5	1	1.43117	1.4312	2.10E-02

Table 15 Comparison of the 5^{th} Order B-spline solution vs. the Frisch-Fay solutions

In Figure 28, all of the B-spline approximation solutions are plotted for each of the load cases given. As expected, the load increases resulted in an increase of the slope of the beam all along its arc length.

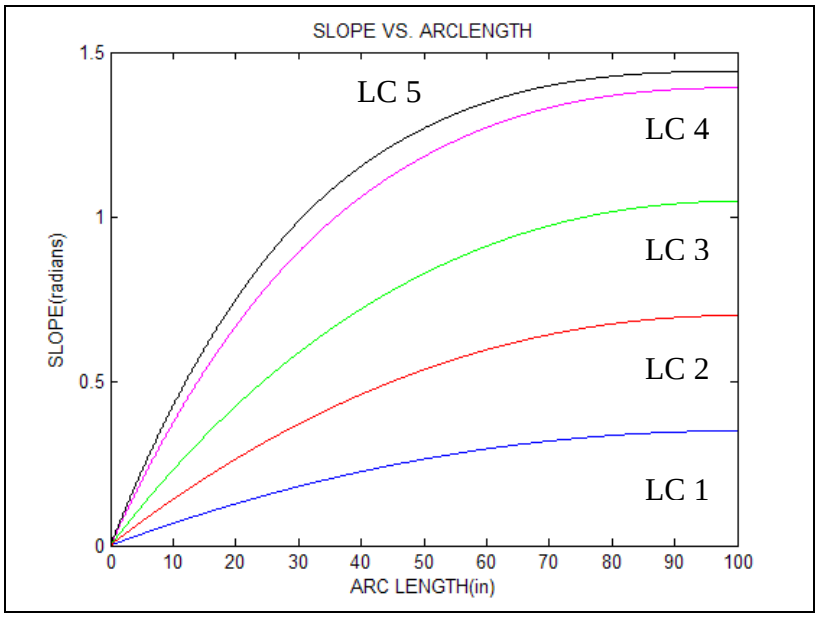

Figure 28 Plots of all of the B-spline solution equations

Once the slope equation with respect to arc length is generated, the deflection in both directions can be calculated without any difficulty. Since $dy = \sin(\theta(s))ds$, the Cartesian ordinate location for each load case can be calculated by integrating over the entire arc length.

$$y = \int_0^L \sin(\theta(s))ds \tag{33}$$

Also, since $dx = \cos(\theta(s))ds$, the $dy = \sin(\theta(s))ds$, the Cartesian abscissa location for each load case can be calculated by integrating over the entire arc length.

$$x = \int_0^L \cos(\theta(s))ds \tag{34}$$

		BEAM END COORDINATES	
LOAD CASE	LOAD (lbs)	X	Y
1	0.073	96.7690	22.9920
2	0.169	87.0995	44.6481
3	0.3405	71.1421	63.6128
4	0.8678	47.2268	79.7426
5	1	41.3610	82.7125

Table 16 Calculated Cartesian coordinate values for all load cases

The abscissa and ordinate coordinates of the end of the beam during the bending process are calculated and listed in Table 16. The end of the beam behaves as expected by moving down and towards the cantilever supported end.

6 Conclusions and Discussion

As discussed in the introduction, there were two objectives of this research. The first was to thoroughly document the B-spline Collocation Method in a very simple and easy to follow format. The second objective was to apply the B-spline Collocation Method to solve a geometrically nonlinear beam problem. The research conducted met both objectives and laid a solid foundation for future work to be done in applying the B-spline Collocation Method to mathematical models.

To meet the first objective, the B-spline Collocation Method process was documented in a manner that any individual with a background in mathematics or engineering can follow. First, the properties, components and derivatives of B-spline curves was discussed in great detail and documented thoroughly. Then, a step-by-step procedure was developed, documented and applied to multiple problems, with different types of boundary conditions. Next, the simplicity and accuracy of the B-spline Collocation Method was discussed and analyzed in great detail. This included a discussion on the benefits of additional internal knot vector points and a comparison of them with the smoothest type. Once the one-dimensional B-spline Collocation Method is documented, it was expanded to two-dimensions and the foundation was laid for future research in this area. Finally, a symbolic Matlab code was developed to help solve boundary value problems with the B-spline Collocation Method.

The second objective was to apply the B-spline Collocation Method to solve a geometrically nonlinear beam problem. A cantilevered beam with a vertical end force was chosen from Frisch-Fay [10], in Chapter 2 of his book ‘Flexible Bars’. In his solution, Frisch-Fay goes through a very complex process to reach his solution. The B-spline solution developed here is found to be very accurate and far simpler to solve. The geometrically non-linear beam problem was evaluated using the B-spline Collocation Method for each load case

and the RMS error was less than 0.33% for all of them. This exceeds the 0.5% benchmark established at the beginning of the research.

The background information, step-by-step procedure, worked out problems, symbolic Matlab code and extensive list of references provided in this thesis, will give a future researcher all of the necessary tools to apply the B-spline Collocation Method to solving mathematical models. The simplicity and accuracy of this method will allow a future researcher to easily apply it to many linear and nonlinear mechanics of materials type problems.

References

1. Beer, F.P., E.R. Johnston Jr. and J.T. DeWolf. *Mechanics of Materials*, 3rd Edition. New York, NY: McGraw-Hill Inc., 2002.
2. Bisshopp, K.E. and D.C. Drucker, "Large Deflection of Cantilever Beams," *Quarterly of Applied Mathematics, Vol. 3, No.3* (1945): 272-275.
3. Botella, O., "On a B-spline Collocation Method for the Solution of the Navier-Stokes Equations," *Computers & Fluids* 31 (2001): 397-420.
4. Budynas, R.G., *Advanced Strength and Applied Stress Analysis*, 2nd Edition. New York, NY: McGraw-Hill Inc., 1999.
5. Chapra, S.C. and R.P. Canale. *Numerical Methods for Engineers*, 4th Edition. New York, NY: McGraw-Hill Inc., 2002.
6. deBoor, Carl. *A Practical Guide to Splines*. New York, NY: Springer-Verlag, 1978
7. de Boor, C., and B. Swartz. "Collocation at Gaussian points." *SIAM Journal on Numerical Analysis* 10 (1973): 582-606.
8. Fairweather, G. and D. Meade. "A Survey of Spline Collocation Methods for the Numerical Solution of Differential Equations." *Mathematics for Large Scale Computing* 120 (1989): 297-341.
9. Farin, G., *Curves and Surfaces for CAGD – A Practical Guide*, 5th Edition. San Diego, CA: Academic Press, 2002.
10. Frisch-Fay, R., *Flexible Bars*. London, England: Butterworth and Co, 1962.
11. Ghoneim, H., "Investigation of the Pumping Action of a Hyperbolic Composite Shell Structure using the Spline Collocation Method." *Proceedings of the American Society of Composites, ASC 22nd Technical Conference, Seattle WA* (2007), No 028.

12. Ghoneim, H. and T. Santos. “Evaluation of the Pumping Action of a Barrel-Shaped Composite Shell Structure.” *Proceedings of IMEC2009, ASME International Mechanical Engineering Congress and Exposition, Lake Buena Vista, FL* (2009).
13. Guzman, H.F. and R.M. Morillo. “Automatic Unstructured Mesh Generation Around Two-Dimensional Domains Described by B-spline Curves.” *Proceedings of DETC’01, ASME 2001 Design Engineering Technical Conference* in, Pittsburgh, *PA* (2001).
14. Hou, Chao-Sheng, Yue Yin and Cheng-Bo Wang. “Axisymmetric Nonlinear Stability of a Shallow Conical Shell with a Spherical Cap of Arbitrary Variable Shell Thickness.” *Journal of Engineering Mechanics* (2006): 1146-1149.
15. Howell, L.L. and A. Midha. “Parametric Deflection Approximations for End-Loaded Large Delection Beams in Compliant Mechanisms,” *Journal of Mechanical Design* 117 (1995): 156-165.
16. Incropera, Frank P. and David P. DeWitt. *Fundamentals of Heat and Mass Transfer*, 5th Edition. Hoboken, New Jersey: John Wiley and Sons, 2002.
17. Jator, Samuel and Zachariah Sinkala. “A Higher Order B-spline Collocation Method for Linear Boundary Value Problems.” *Applied Mathematics and Computations* 191 (2007): 100-116.
18. Johnson, R.W., “Higher Order B-spline Collocation at the Greville Abscissa.” *Applied Numerical Mathematics* (2005): 52:63-75.
19. Johnson, R.W., “A B-spline Collocation Method for solving the Incompressible Navier-Stokes Equations Using an ad hoc Method: the Boundary Residual Method,” *Computers & Fluids* 34 (2005): 121-149.

20. Kadalbajoo, M.K. and A.S.Yadaw. “B-Spline Collocation Method for a Two-parameter Singularly Perturbed Convection-diffusion Boundary Value Problem.” *Applied Mathematics and Computations* 201 (2008): 504-513.
21. Kadalbajoo, M.K. and D. Kumar. “Fitted Mesh B-spline Collocation Method for Singularly Perturbed Differential-difference Equation with Small Delay.” *Applied Mathematics and Computations* 204 (2008): 90-98.
22. Kadalbajoo, M.K. and Vikas Gupta. “Numerical Solution of Singularly Perturbed Convection-diffusion Problem Using Parameter Uniform B-spline Collocation Method.” *Journal of Mathematical Analysis and Applications* 355 (2009): 439-452.
23. Kadalbajoo, M.K. and Puneet Arora. “B-spline Collocation Method for Singular-Perturbation Problem Using Artificial Viscosity.” *Computers and Mathematics with Applications* 57(2009): 650-663.
24. Kadalbajoo, M.K., Vikas Gupta and Ashish Awasthi. “A Uniformly Convergent B-spline Collocation Method on a Nonuniform Mesh for Singularly Perturbed One-Dimensional Time-Dependant Linear Convection-Diffusion Problem.” *Journal of Computational and Applied Mathematics* 220 (2008): 271-289.
25. Kadalbajoo, M.K. and Vivek K. Aggarwal. “Fitted Mesh B-spline Collocation Method for Solving Self-adjoint Singularly Perturbed Boundary Value Problems.” *Applied Mathematics and Computations* 161 (2008): 973-987.
26. Khattak, A.J. and Siraj-ul-Islam. “A Comparative Study of Numerical Solutions of a Class of KdV Equation.” *Applied Mathematics and Computations* 199 (2008): 425-434.
27. Kiusalaas, Jaan. *Numerical Methods in Engineering with Matlab*. New York, New York: Cambridge University Press, 2005.

28. Kreyszig, Erwin. *Advanced Engineering Mathematics*, 9th Edition. Hoboken, NJ: John Wiley and Sons, Inc, 2006.

29. Lawrence, Kent. *ANSYS Workbench Tutorial, ANSYS Release 10*. Canonsburg, PA: SDC Publications, 2006.

30. Lin, Bin, Kaitai Li and Zhengxing Cheng. " B-spline Solution of a Singularly Perturbed Boundary Value Problem Arising in Biology." *Chaos, Solitons and Fractals* 42 (2009): 2934-2948.

31. Luo, Albert C.J. "An Approximate Theory for Geometrically Nonlinear Thin Plates." *International Journal of Solids and Structures* 37 (2000): 7655-7670.

32. Mackerle, Jaroslav. "Geometric-nonlinear Analysis by Finite Element and Boundary Element Methods - A Bibliography (1997-1998)." *Finite Elements in Analysis and Design* 32 (1999): 51-62.

33. Mazzia, F., A. Sestini and D. Trigiante. "B-spline Linear Multistep Methods and their Continuous Extensions." *SIAM Journal on Numerical Analysis* 44 (2006): 1954-1973.

34. Otto, S.R. and J.P. Denier. *An Introduction to Programming and Numerical Methods in Matlab*. London, England: Springer-Verlag, 2005.

35. Prautzsch, H., W. Boehm and M. Paluszny. *Bezier and B-Spline Techniques*. Berlin, Germany: Springer-Verlag, 2002.

36. Ramadan, M.A., Talaat S. El-Danaf and Faisal E.I. Abd Alaal. "A Numerical Solution of the Burgers' Equation Using Septic B-splines." *Chaos, Solitons and Fractals* 26 (2005): 1249-1258.

37. Rao, S. C. S., and M. Kumar. "Optimal B-spline Collocation Method for Self-adjoint Singularly Perturbed Boundary Value Problems." *Applied Mathematics and Computation* 188 (2007): 749-761.

38. Rao, S.C.S and Mukesh Kumar. “Exponential B-spline Collocation Method for Self-adjoint Singularly Perturbed Boundary Value Problems.” *Applied Numerical Mathematics* 58 (2008): 1572-1581.

39. Reddy, J.N., *An Introduction to the Finite Element Method*, 3rd Edition. New York, NY: McGraw-Hill Inc., 2006.

40. Rogers, D.F. and J.A. Adams. *Mathematical Elements for Computer Graphics*. New York, NY: McGraw-Hill, 1990.

41. Saka, B. and Idris Dag. “A Numerical Study of the Burgers’ Equation.” *Journal of the Franklin Institute* 345 (2008): 328-348.

42. Sathyamoorthy, M., *Nonlinear Analysis of Structures*. New York, NY: CRC Press LLC, 1998.

43. Shariff, Karim and Robert D. Moser. “Two-Dimensional Mesh Embedding for B-spline Methods.” *Journal of Computational Physics* 145 (1998): 471-488.

44. Sun, Weiwei. “B-spline Collocation Methods for Elasticity Problems.” *Scientific Computing and Applications* by Peter Minev and Yanping Lin. Huntington, NY: Nova Science Publishers, 2001: 133-141.

45. Timoshenko, S.P. and J.N. Goodier. *Theory of Elasticity*, 3rd Edition. Singapore: McGraw-Hill Inc., 1951.

46. Von Karman, T., “The Engineer Grapples with Nonlinear Problems,” *Bulletin of American Mathematical Society* v. 46 (1940): 615-683.

47. Widjaja, Ronny, Andrew Ooi, Li Chen and Richard Manasseh. “Computational Aeroacoustics Using the B-spline Collocation Method.” *C. R. Mechanique* 333 (2005):726-731.

48. Wilson, Howard B., Louis H. Turcotte and David Halpern. *Advanced Mathematics and Mechanics Applications Using Matlab*, 3rd Edition. Boca Raton, FL: Chapman & Hall/CRC Press LLC, 2003.

49. Wu, Lai-Yun, Lap-Loi Chung and Hsu-Hui Huang. " Radial Spline Collocation Method for Static Analysis of Beams." *Applied Mathematics and Computations* 201 (2008): 184-199.

50. Yang, Won Y., Wenwu Cao, Tae-Sang Chung and John Morris. *Applied Numerical Methods Using Matlab*. Hoboken, NJ: John Wiley and Sons, Inc, 2005.

51. Young, Warren C. and Richard G. Budynas. *Roark's Formulas for Stress and Strain*, 7th Edition. New York, New York: Mc-Graw Hill, 2002.

Appendix

Matlab Code for Example 1A

```
% Jason Magoon
% June 1, 2009
% EXAMPLE PROBLEM 1a

clear all
clc
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED SYMBOLIC VARIABLES
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16
t = sym('t');
%% %%%%%%%%%%%%%%%% REQUIRED INPUTS %%%%%%%%%%%%%%%%%%%%%%%%%
K = 3;                                      % ORDER
T = [0 0 0 1 1 1];                          % KNOT VECTOR
Binit = [B1 B2 B3];                         % # of B'S = K + intvl - 1
L = 1;
        %%%%% PRE-CALCULATED BOUNDARY CONDITIONS %%%%%
%%ALSO MODIFY LINES 147 - 169%%%%%
B1 = 0;
B3 = 1;
    %%%%% DIFFERENTIAL EQUATION PARAMETERS %%%%%
% E*P''+ F*P' + G*P + H = 0
E = 1;F = 2;G = 1;H = 0;
ODE = 2;   %ORDER OF ODE
            %%%%%% CALC EXACT SOLUTION %%%%%
x = 0:0.01:L;
exact = 2.718.*x.*exp(-x);              %EXACT SOLUTION OF ODE
EXACTt = 2.718.*t.*exp(-t);             %FOR COMPARISON CALCULATIONS
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED STARTING VARIABLES
intvl = 0;
m = length(T);
n = K - 1;
p = 1; d = 1; e = K; f = 2; h = K; r = 0;
X = zeros(m-n+1,1);
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE NUMBER OF INTERVALS IN KNOT VECTOR
for i = 1 : m - 1
    if T(i) < T(i+1)
        count = 1;
    else
        count = 0;
    end
    intvl = intvl + count;
    i = i + 1;
end
intvl;
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CREATE SYMBOLIC 'N' ARRAY
for i = 1:K + intvl
    for j = 1:intvl
    N(i,j*K) = t;
    end
```

```
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE BASIS FUNCTIONS 'N'
for k = 1:intvl
    for i = 1:K + intvl
        for j = d
            if i==e
                N(i,j) = 1;
            else
                N(i,j) = 0;
            end
        end
    end
    d = d + K;
    e = e + 1;
end
for k = 1:intvl
    for j = 2:K
        for i = 1:K + intvl - 1
            lh = iszero(t - T(i),T(i+j-1) - T(i));
            rh = iszero(T(i+j) - t,T(i+j) - T(i+1));
            N(i,j+r) = ((lh) * N(i,j+r-1)) + ((rh) * N(i+1,j+r-1));
        end
    end
        r = r + K;
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ADDITIONAL MATRIX OPERATIONS
N(K+intvl,:) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
%%%%%%%%%% DELETING ALL NON'K'MULTIPLE COLUMNS OF 'N'%%%%%%%%%%
for j=1:intvl*K
        I(1,j) = j;
        j = j + 1;
end
for k = intvl:-1:1
    I(:,k*K) = [];
end
N(:,I) = [];
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% GREVILLE ABSCISSAE CALCULATION
for i = 1:m-n+1
    while i < m-n+2
        TG = sum(T(i:i+n-1));
        X(i)=(1/n)*TG;
        XCOM(i) = X(i);          %FOR COMPARISON CALCULATIONS
        i = i + 1;
    end
end
%% %%%%%%%%%% DELETE FIRST 2 AND LAST 2 VALUES OF X %%%%%%%%%%%%%%
X(m-n+1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
XCOM(m-n+1) = [];           %DELETE EXTRA ROW USED FOR CALCULATION
X(m-n) = [];                %DELETE EXTRA ROW USED FOR CALCULATION
X(2) = [];                  %DELETE EXTRA ROW USED FOR CALCULATION
X(1) = [];                  %DELETE EXTRA ROW USED FOR CALCULATION
XCOM(1) = [];               %DELETE EXTRA ROW USED FOR CALCULATION
X
%XCOM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%ADD DERIVATIVES TO 'N'
DLIM = intvl * ODE;
i = 1;
for j = intvl + 1:intvl + DLIM
        N(:,j) = diff(N(:,i));
        i = i + 1;
end
N
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE 'P'
clear t
PEQ = Binit * N;
Pinit = subs(PEQ);
P2 = transpose(Pinit);
%%%%%%%%%%%%%%%%%%CALCULATE 'B' VALUES%%%%%%%%%%%%%%%%%%%%%%%%
for i = 1:intvl
    EQ(i,1) = simple((E*P2(i+ODE*intvl))/L^2 + (F*P2(i+intvl))/L + G*P2(i));
end

VARCNT = length(X);
UNKCNT = length(Binit)-2;

i = 1;
for k = 1:VARCNT
        if X(k) <= T(K+i)
            t = X(k);
            eq(k) = subs(EQ(i,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)=G*Nf(j+1,i) + (F*Nf(j+1,i+intvl))/L +
(E*Nf(j+1,i+2*intvl))/L^2;
            end
        else
            i = i+1;
            t = X(k);
            eq(k) = subs(EQ(i,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)=G*Nf(j+1,i) + (F*Nf(j+1,i+intvl))/L +
(E*Nf(j+1,i+2*intvl))/L^2;
            end
        end
end

B1=0;B2=0;B3=0;B4=0;B5=0;B6=0;B7=0;B8=0;B9=0;B10=0;
B11=0;B12=0;B13=0;B14=0;B15=0;B16=0;

remain = transpose(-eval(eq));
B = inv(A)*remain

%%%%%%%%%%%%%%SUBSTITUTE 'B' VALUES INTO 'P'%%%%%%%%%%%%%%%%%%%
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16
%%%% ASSIGN SYMBOLIC 'B' VALUES BETWEEN BOUNDARY CONDITIONS %%%%%
B2=B(1);

clear t
for i = 1:intvl;
```

```
    ANS(i,1) = subs(P2(i));
end
ANS
%%%%%%%%%%%%%PLOT APPROXIMATE SOLUTION%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2)
v=1;
if intvl == 1
    for j = 1:K
        xfin = 0:0.01:L;
        t = xfin / L;
        FINANS = subs(ANS);
        plot(xfin,FINANS,'b');
        hold on
    end
else
    for i = 0:intvl - 1
        xfin  = T(K+i)*L:0.01:T(K+1+i)*L;
        t = xfin / L;
        i = i + 1;
        FINANS = subs(ANS(i,1));
        plot(xfin,FINANS,'b');
        v = v + 1;
        hold on
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT BASIS FUNCTIONS
figure(1)
if intvl == 1
    for j = 1:K
        t = 0:0.01:1;
        Q = subs(N(j,1));
        plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N');
        hold on
    end
else
    for i = 0:intvl - 1
        t  = T(K+i):0.01:T(K+1+i);
        i = i + 1;
        for j = 1:K+intvl-1
            Q = subs(N(j,i));
            plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N');
            hold on
        end
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT EXACT SOLUTION %%
figure(2)
plot(x,exact,'r');title('EXACT vs APPROXIMATE
SOLUTIONS');xlabel('X');ylabel('Y');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% COMPARE RESULTS %%%%%%%%%%%%%%%%%%%%%%%%%%
for i = 1:length(XCOM)
    t = XCOM(i);
    CALCANS(1,i) = subs(ANS);
    EXACTANS(i,1) = subs(EXACTt);
end
```

```
fprintf(1,'\nComputed Solutions With Error\n');
fprintf(1,'    No      x      Approx       Exact       %%RelErr\n');
fprintf(1,' ------  ------ ------------ ------------ ------------\n');
for j = 1:length(XCOM)
        err = 100*abs(1 - CALCANS(j)/EXACTANS(j));
        E(j)=err;
   fprintf(1,'%6.0f %6.2f %12.5f %12.5f
%12.4e\n',j,XCOM(j),CALCANS(j),EXACTANS(j),err);
end

NO = length(E);
E(1) = [];
ESQR = E.^2;
RMSS = sqrt(sum(ESQR)/NO);
fprintf(1,'\nComputed Root Mean Sum of the Squares Error\n');
fprintf(1,'%12.5e\n',RMSS);

function out = iszero(num,den)
if den == 0
    out = 0;
else
    out = num/den;
end
return
```

Matlab Code for Example 1B

```
% Jason Magoon
% June 1, 2009
% EXAMPLE PROBLEM 1b

clear all
clc
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED SYMBOLIC VARIABLES
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16
t = sym('t');
%% %%%%%%%%%%%%%%%% REQUIRED INPUTS %%%%%%%%%%%%%%%%%%%%%%%%%
K = 3;                                                  % ORDER
T = [0 0 0 1/3 2/3 1 1 1];               % KNOT VECTOR
Binit = [B1 B2 B3 B4 B5];                    % # of B'S = K + intvl - 1
L = 1;
        %%%%% PRE-CALCULATED BOUNDARY CONDITIONS %%%%%
%%ALSO MODIFY LINES 147 - 169%%%%%
B1 = 0;
B5 = 1;
    %%%%% DIFFERENTIAL EQUATION PARAMETERS %%%%%
% E*P''+ F*P' + G*P + H = 0
E = 1;
F = 2;
G = 1;
H = 0;
ODE = 2;   %ORDER OF ODE
            %%%%%% CALC EXACT SOLUTION %%%%%
x = 0:0.01:L;
exact = 2.718.*x.*exp(-x);              %EXACT SOLUTION OF ODE
EXACTt = 2.718.*t.*exp(-t);             %FOR COMPARISON CALCULATIONS
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED STARTING VARIABLES
intvl = 0;
m = length(T);
n = K - 1;
p = 1; d = 1; e = K; f = 2; h = K; r = 0;
X = zeros(m-n+1,1);
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE NUMBER OF INTERVALS IN KNOT VECTOR
for i = 1 : m - 1
    if T(i) < T(i+1)
        count = 1;
    else
        count = 0;
    end
    intvl = intvl + count;
    i = i + 1;
end
intvl;
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CREATE SYMBOLIC 'N' ARRAY
for i = 1:K + intvl
    for j = 1:intvl
    N(i,j*K) = t;
    end
end
```

```
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE BASIS FUNCTIONS 'N'
for k = 1:intvl
    for i = 1:K + intvl
        for j = d
            if i==e
                N(i,j) = 1;
            else
                N(i,j) = 0;
            end
        end
    end
    d = d + K;
    e = e + 1;
end
for k = 1:intvl
    for j = 2:K
        for i = 1:K + intvl - 1
            lh = iszero(t - T(i),T(i+j-1) - T(i));
            rh = iszero(T(i+j) - t,T(i+j) - T(i+1));
            N(i,j+r) = ((lh) * N(i,j+r-1)) + ((rh) * N(i+1,j+r-1));
        end
    end
        r = r + K;
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ADDITIONAL MATRIX OPERATIONS
N(K+intvl,:) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
%%%%%%%%%% DELETING ALL NON'K'MULTIPLE COLUMNS OF 'N'%%%%%%%%%%
for j=1:intvl*K
        I(1,j) = j;
        j = j + 1;
end
for k = intvl:-1:1
    I(:,k*K) = [];
end
N(:,I) = [];
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% GREVILLE ABSCISSAE CALCULATION
for i = 1:m-n+1
    while i < m-n+2
        TG = sum(T(i:i+n-1));
        X(i)=(1/n)*TG;
        XCOM(i) = X(i);          %FOR COMPARISON CALCULATIONS
        i = i + 1;
    end
end
%% %%%%%%%%%% DELETE FIRST 2 AND LAST 2 VALUES OF X %%%%%%%%%%%%%%
X(m-n+1) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
XCOM(m-n+1) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
X(m-n) = [];            %DELETE EXTRA ROW USED FOR CALCULATION
X(2) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
X(1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
XCOM(1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
X
XCOM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%ADD DERIVATIVES TO 'N'
```

```
DLIM = intvl * ODE;
i = 1;
for j = intvl + 1:intvl + DLIM
        N(:,j) = diff(N(:,i));
        i = i + 1;
end
N
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE 'P'
clear t
PEQ = Binit * N;
Pinit = subs(PEQ);
P2 = transpose(Pinit);
%%%%%%%%%%%%%%%%%%%CALCULATE 'B' VALUES%%%%%%%%%%%%%%%%%%%%%%%%%
for i = 1:intvl
    EQ(i,1) = simple((E*P2(i+ODE*intvl))/L^2 + (F*P2(i+intvl))/L + G*P2(i));
end

VARCNT = length(X);
UNKCNT = length(Binit)-2;

i = 1;
for k = 1:VARCNT
        if X(k) <= T(K+i)
            t = X(k);
            eq(k) = subs(EQ(i,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)=G*Nf(j+1,i) + (F*Nf(j+1,i+intvl))/L +
(E*Nf(j+1,i+2*intvl))/L^2;
            end
        else
            i = i+1;
            t = X(k);
            eq(k) = subs(EQ(i,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)=G*Nf(j+1,i) + (F*Nf(j+1,i+intvl))/L +
(E*Nf(j+1,i+2*intvl))/L^2;
            end
        end
end

B1=0;B2=0;B3=0;B4=0;B5=0;B6=0;B7=0;B8=0;B9=0;B10=0;
B11=0;B12=0;B13=0;B14=0;B15=0;B16=0;

remain = transpose(-eval(eq));
B = inv(A)*remain

%%%%%%%%%%%%%%SUBSTITUTE 'B' VALUES INTO 'P'%%%%%%%%%%%%%%%%%%%
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16
%%%% ASSIGN SYMBOLIC 'B' VALUES BETWEEN BOUNDARY CONDITIONS %%%%%
B2=B(1);
B3=B(2);
B4=B(3);

clear t
```

```
for i = 1:intvl;
    ANS(i,1) = subs(P2(i));
end
ANS
%%%%%%%%%%%%%PLOT APPROXIMATE SOLUTION%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2)
v=1;
if intvl == 1
    for j = 1:K
        xfin = 0:0.01:L;
        t = xfin / L;
        FINANS = subs(ANS);
        plot(xfin,FINANS,'b');
        hold on
    end
else
    for i = 0:intvl - 1
        xfin  = T(K+i)*L:0.01:T(K+1+i)*L;
        t = xfin / L;
        i = i + 1;
        FINANS = subs(ANS(i,1));
        plot(xfin,FINANS,'b');
        v = v + 1;
        hold on
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT BASIS FUNCTIONS
figure(1)
if intvl == 1
    for j = 1:K
        t = 0:0.01:1;
        Q = subs(N(j,1));
        plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N');
        hold on
    end
else
    for i = 0:intvl - 1
        t  = T(K+i):0.01:T(K+1+i);
        i = i + 1;
        for j = 1:K+intvl-1
            Q = subs(N(j,i));
            plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N');
            hold on
        end
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT EXACT SOLUTION %%
figure(2)
plot(x,exact,'r');title('EXACT vs APPROXIMATE
SOLUTIONS');xlabel('X');ylabel('Y');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% COMPARE RESULTS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
m = K+1;
if intvl == 1
    for i = 1:length(XCOM)
        t = XCOM(i);
```

```
        CALCANS(1,i) = subs(ANS);
        EXACTANS(1,i) = subs(EXACTt);
    end
else
    for i = 1:length(XCOM)
        if XCOM(i)<=T(m)
            m;
            t = XCOM(i);
            CALCANS(1,i) = subs(ANS(m-K));
            EXACTANS(1,i) = subs(EXACTt);
        else
            m = m + 1;
            t = XCOM(i);
            CALCANS(1,i) = subs(ANS(m-K));
            EXACTANS(1,i) = subs(EXACTt);
        end
    end
end

fprintf(1,'\nComputed Solutions With Error\n');
fprintf(1,'    No      x      Approx       Exact       %%RelErr\n');
fprintf(1,' ------  ------ ------------ ------------ ------------\n');
for j = 1:length(XCOM)
        err = 100*abs(1 - CALCANS(j)/EXACTANS(j));
        E(j)=err;
   fprintf(1,'%6.0f %6.2f %12.5f %12.5f
%12.4e\n',j,XCOM(j),CALCANS(j),EXACTANS(j),err);
end

NO = length(E);
E(1) = [];
ESQR = E.^2;
RMSS = sqrt(sum(ESQR)/NO);
fprintf(1,'\nComputed Root Mean Sum of the Squares Error\n');
fprintf(1,'%12.5e\n',RMSS);

function out = iszero(num,den)
if den == 0
    out = 0;
else
    out = num/den;
end
return
```

Matlab Code for Example 2

```
% Jason Magoon
% SEPTEMBER 9, 2009
% EXAMPLE PROBLEM 2

clear all
clc
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED SYMBOLIC VARIABLES
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16 B17 B18 B19 B20
t = sym('t');
%% %%%%%%%%%%%%%%%% REQUIRED INPUTS %%%%%%%%%%%%%%%%%%%%%%%%%
K = 3;                                                  % ORDER
T = [0 0 0 1 1 1];              % KNOT VECTOR
Binit = [B1 B2 B3];                     % # of B'S = K + intvl - 1
L = 1;
        %%%%% PRE-CALCULATED BOUNDARY CONDITIONS %%%%%
%%ALSO MODIFY LINES 147 - 169%%%%%
B1 = 0;
B3 = 1;
%     %%%%% DIFFERENTIAL EQUATION PARAMETERS %%%%%
% % E*P''+ F*P' + G*P + H = 0
E = 1;
F = 0;
G = 16;          %lambda^2
H = 0;
lambda = sqrt(G);
ODE = 2;   %ORDER OF ODE
            %%%%%% CALC EXACT SOLUTION %%%%%
x = 0:0.01:L;
exact = (1/sin(lambda)).*sin(lambda*x);              %EXACT SOLUTION OF ODE
EXACTt = (1/sin(lambda)).*sin(lambda*t);
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED STARTING VARIABLES
intvl = 0;
m = length(T);
n = K - 1;
p = 1; d = 1; e = K; f = 2; h = K; r = 0;
X = zeros(m-n+1,1);
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE NUMBER OF INTERVALS IN KNOT VECTOR
for i = 1 : m - 1
    if T(i) < T(i+1)
        count = 1;
    else
        count = 0;
    end
    intvl = intvl + count;
    i = i + 1;
end
intvl;
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CREATE SYMBOLIC 'N' ARRAY
for i = 1:K + intvl
    for j = 1:intvl
    N(i,j*K) = t;
    end
```

```
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE BASIS FUNCTIONS 'N'
for k = 1:intvl
    for i = 1:K + intvl
        for j = d
            if i==e
                N(i,j) = 1;
            else
                N(i,j) = 0;
            end
        end
    end
    d = d + K;
    e = e + 1;
end
for k = 1:intvl
    for j = 2:K
        for i = 1:K + intvl - 1
            lh = iszero(t - T(i),T(i+j-1) - T(i));
            rh = iszero(T(i+j) - t,T(i+j) - T(i+1));
            N(i,j+r) = ((lh) * N(i,j+r-1)) + ((rh) * N(i+1,j+r-1));
        end
    end
        r = r + K;
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ADDITIONAL MATRIX OPERATIONS
N(K+intvl,:) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
%%%%%%%%%% DELETING ALL NON'K'MULTIPLE COLUMNS OF 'N'%%%%%%%%%%
for j=1:intvl*K
        I(1,j) = j;
        j = j + 1;
end
for k = intvl:-1:1
    I(:,k*K) = [];
end
N(:,I) = [];
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% GREVILLE ABSCISSAE CALCULATION
for i = 1:m-n+1
    while i < m-n+2
        TG = sum(T(i:i+n-1));
        X(i)=(1/n)*TG;
        XCOM(i) = X(i);          %FOR COMPARISON CALCULATIONS
        i = i + 1;
    end
end
%% %%%%%%%%%% DELETE FIRST 2 AND LAST 2 VALUES OF X %%%%%%%%%%%%%%
X(m-n+1) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
XCOM(m-n+1) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
X(m-n) = [];            %DELETE EXTRA ROW USED FOR CALCULATION
X(2) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
X(1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
XCOM(1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
X
XCOM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%ADD DERIVATIVES TO 'N'
DLIM = intvl * ODE;
i = 1;
for j = intvl + 1:intvl + DLIM
        N(:,j) = diff(N(:,i));
        i = i + 1;
end
N
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE 'P'
clear t
PEQ = Binit * N;
Pinit = subs(PEQ);
P2 = transpose(Pinit);
%%%%%%%%%%%%%%%%%%CALCULATE 'B' VALUES%%%%%%%%%%%%%%%%%%%%%%%%%
for i = 1:intvl
    EQ(i,1) = simple((E*P2(i+ODE*intvl))/L^2 + G*P2(i));
end

VARCNT = length(X);
UNKCNT = length(Binit)-2;

i = 1;
for k = 1:VARCNT
        if X(k) <= T(K+i)
            t = X(k);
            eq(k) = subs(EQ(i,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)=G*Nf(j+1,i) + (E*Nf(j+1,i+2*intvl))/L^2;
            end
        else
            i = i+1;
            t = X(k);
            eq(k) = subs(EQ(i,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)=G*Nf(j+1,i) + (E*Nf(j+1,i+2*intvl))/L^2;
            end
        end
end

B1=0;B2=0;B3=0;B4=0;B5=0;B6=0;B7=0;B8=0;B9=0;B10=0;

remain = transpose(-eval(eq));
B = inv(A)*remain

%%%%%%%%%%%%%%%SUBSTITUTE 'B' VALUES INTO 'P'%%%%%%%%%%%%%%%%%%%
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16 B17 B18 B19 B20
%%%% ASSIGN SYMBOLIC 'B' VALUES BETWEEN BOUNDARY CONDITIONS %%%%%
B2=B(1);

clear t
for i = 1:intvl;
    ANS(i,1) = subs(P2(i));
end
ANS
```

```
%%%%%%%%%%%%%PLOT APPROXIMATE SOLUTION%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2)
v=1;
if intvl == 1
    for j = 1:K
        xfin = 0:0.01:L;
        t = xfin / L;
        FINANS = subs(ANS);
        COMPARE=FINANS;
        plot(xfin,FINANS,'b');
        hold on
    end
else
    for i = 0:intvl - 1
        xfin  = T(K+i)*L:0.01:T(K+1+i)*L;
        t = xfin / L;
        i = i + 1;
        FINANS = subs(ANS(i,1));
        COMPARE(i,1:length(xfin))=FINANS;
        plot(xfin,FINANS,'b');
        v = v + 1;
        hold on
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT BASIS FUNCTIONS
figure(1)
if intvl == 1
    for j = 1:K
        t = 0:0.01:1;
        Q = subs(N(j,1));
        plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N');
        hold on
    end
else
    for i = 0:intvl - 1
        t  = T(K+i):0.01:T(K+1+i);
        i = i + 1;
        for j = 1:K+intvl-1
            Q = subs(N(j,i));
            plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N');
            hold on
        end
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT EXACT SOLUTION %%
figure(2)
plot(x,exact,'r');title('EXACT vs APPROXIMATE
SOLUTIONS');xlabel('X');ylabel('Y');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% COMPARE RESULTS %%%%%%%%%%%%%%%%%%%%%%%%%%
m = K+1;
if intvl == 1
    for i = 1:length(XCOM)
        t = XCOM(i);
        CALCANS(1,i) = subs(ANS);
        EXACTANS(1,i) = subs(EXACTt);
```

```
    end
else
    for i = 1:length(XCOM)
        if XCOM(i)<=T(m)
            m;
            t = XCOM(i);
            CALCANS(1,i) = subs(ANS(m-K));
            EXACTANS(1,i) = subs(EXACTt);
        else
            m = m + 1;
            t = XCOM(i);
            CALCANS(1,i) = subs(ANS(m-K));
            EXACTANS(1,i) = subs(EXACTt);
        end
    end
end

fprintf(1,'\nComputed Solutions With Error\n');
fprintf(1,'    No      x      Approx       Exact       %%RelErr\n');
fprintf(1,' ------  ------ ------------ ------------ ------------\n');
for j = 1:length(XCOM)
        err = 100*abs(1 - CALCANS(j)/EXACTANS(j));
        E(j)=err;
        fprintf(1,'%6.0f %6.2f %12.5f %12.5f
%12.4e\n',j,XCOM(j),CALCANS(j),EXACTANS(j),err);
end
NO = length(E);
E(1) = [];
ESQR = E.^2;
RMSS = sqrt(sum(ESQR)/NO);
fprintf(1,'\nComputed Root Mean Sum of the Squares Error\n');
fprintf(1,'%12.5e\n',RMSS);

function out = iszero(num,den)
if den == 0
    out = 0;
else
    out = num/den;
end
return
```

Matlab Code for Example 3

```
% Jason Magoon
% SEPTEMBER 9, 2009
% EXAMPLE PROBLEM 3

clear all
clc
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED SYMBOLIC VARIABLES
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16
t = sym('t');
%% %%%%%%%%%%%%%%%% REQUIRED INPUTS %%%%%%%%%%%%%%%%%%%%%%%%%
K = 6;                                                  % ORDER
T = [0 0 0 0 0 0 1 1 1 1 1 1];                  % KNOT VECTOR
Binit = [B1 B2 B3 B4 B5 B6];                   % # of B'S = K + intvl - 1
L = 10;                           % BEAM LENGTH, UNITS: IN
EL = 29000000;                   % ASTM-A36, UNITS: PSI
IN = .6;                       % 2 X 2 X 1/8 WALL TUBE, UNITS: IN^4
W = 10;                           % DISTRIBUTED LOAD, UNITS: LBS/IN
%%%%% PRE-CALCULATED BOUNDARY CONDITIONS %%%%%
%%ALSO MODIFY LINES 147 - 169%%%%%
B1 = 0;          % PRECALCULATED
B2 = 0;          % PRECALCULATED
BC3 = 2;         % ORDER OF 3RD BOUNDARY CONDITION
BC4 = 3;         % ORDER OF 4TH BOUNDARY CONDITION

    %%%%% DIFFERENTIAL EQUATION PARAMETERS %%%%%
% C*P''''+ D*P'''+ E*P''+ F*P' + G*P + H = 0
C = 1;
D = 0;
E = 0;
F = 0;
G = 0;
H = W /(EL*IN);
ODE = 4;   %ORDER OF ODE
            %%%%%% CALC EXACT SOLUTION %%%%%
x = 0:0.01:L;
exact = ((W.*x.^2)/(24*EL*IN)).*(4*L.*x-x.^2-6*L^2);    %EXACT SOLUTION OF
ODE
EXACTt = ((W.*t.^2)/(24*EL*IN)).*(4*L.*t-t.^2-6*L^2);    %EXACT SOLUTION OF
ODE
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED STARTING VARIABLES
intvl = 0;
m = length(T);
n = K - 1;
p = 1; d = 1; e = K; f = 2; h = K; r = 0;
X = zeros(m-n+1,1);
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE NUMBER OF INTERVALS IN KNOT VECTOR
for i = 1 : m - 1
    if T(i) < T(i+1)
        count = 1;
    else
        count = 0;
    end
```

```
    intvl = intvl + count;
    i = i + 1;
end
intvl;
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CREATE SYMBOLIC 'N' ARRAY
for i = 1:K + intvl
    for j = 1:intvl
    N(i,j*K) = t;
    end
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE BASIS FUNCTIONS 'N'
for k = 1:intvl
    for i = 1:K + intvl
        for j = d
            if i==e
                N(i,j) = 1;
            else
                N(i,j) = 0;
            end
        end
    end
    d = d + K;
    e = e + 1;
end
for k = 1:intvl
    for j = 2:K
        for i = 1:K + intvl - 1
            lh = iszero(t - T(i),T(i+j-1) - T(i));
            rh = iszero(T(i+j) - t,T(i+j) - T(i+1));
            N(i,j+r) = ((lh) * N(i,j+r-1)) + ((rh) * N(i+1,j+r-1));
        end
    end
        r = r + K;
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ADDITIONAL MATRIX OPERATIONS
N(K+intvl,:) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
%%%%%%%%%% DELETING ALL NON'K'MULTIPLE COLUMNS OF 'N'%%%%%%%%%%
for j=1:intvl*K
        I(1,j) = j;
        j = j + 1;
end
for k = intvl:-1:1
    I(:,k*K) = [];
end
N(:,I) = [];
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% GREVILLE ABSCISSAE CALCULATION
for i = 1:m-n+1
    while i < m-n+2
        TG = sum(T(i:i+n-1));
        X(i)=(1/n)*TG;
        XCOM(i) = X(i);          %FOR COMPARISON CALCULATIONS
        i = i + 1;
    end
end
```

```
%% %%%%%%%%%% DELETE FIRST 2 AND LAST 2 VALUES OF X %%%%%%%%%%%%%%
X(m-n+1) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
XCOM(m-n+1) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
X(m-n) = [];            %DELETE EXTRA ROW USED FOR CALCULATION
X(2) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
X(1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
XCOM(1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
X
XCOM
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%ADD DERIVATIVES TO 'N'
DLIM = intvl * ODE;
i = 1;
for j = intvl + 1:intvl + DLIM
        N(:,j) = diff(N(:,i));
        i = i + 1;
end
N
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE 'P'
clear t
PEQ = Binit * N;
Pinit = subs(PEQ);
P2 = transpose(Pinit)
%%%%%%%%%%%%%%%%%%CALCULATE 'B' VALUES%%%%%%%%%%%%%%%%%%%%%%%%%%
for i = 1:intvl
    EQ(i,1) = simple(((C / L^4).*P2(i+ODE*intvl))+ H);
end
EQ
UNKCNT = length(Binit)-2;

AT1(1)= Binit * N(1:K+intvl-1,intvl+intvl*BC3);
AT1(2)= Binit * N(1:K+intvl-1,intvl+intvl*BC4);

k = 1;
t = L;
eq(k) = subs(AT1(k));
Nf = eval(N);
for j = 1:UNKCNT
    A(k,j) = Nf(j + 2,intvl+intvl*BC3);
end
k = 2;
t = L^2;
eq(k) = subs(AT1(k));
for j = 1:UNKCNT
    A(k,j) = Nf(j + 2,intvl+intvl*BC4);
end
eq;
clear t

XUSED=X;
XUSED(length(X)) = [];      %DELETE EXTRA ROW USED FOR CALCULATION
XUSED(1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
XUSED;

i = ODE*intvl+1;
a = 1;
```

```
b = 1;
c = 1;
for k = 3:UNKCNT
            if XUSED(a) <= T(K+c)
                t = XUSED(a);
                eq(k) = subs(EQ(b,1));
                Nf = eval(N);
                for j = 1:UNKCNT
                    A(k,j)=Nf(j+2,i)/L^4;
                end
                a = a + 1;
            else
            b = b + 1;
            c = c + 1;
            i = i + 1;
            t = XUSED(a);
            eq(k) = subs(EQ(b,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)=Nf(j+2,i)/L^4;
            end
            a = a + 1;
            end
end

A
eq
B1=0;B2=0;B3=0;B4=0;B5=0;B6=0;B7=0;B8=0;B9=0;B10=0;
B11=0;B12=0;B13=0;B14=0;B15=0;B16=0;

remain = transpose(-eval(eq))
B = inv(A)*remain

%%%%%%%%%%%%%%SUBSTITUTE 'B' VALUES INTO 'P'%%%%%%%%%%%%%%%%%%%%
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16
%%%% ASSIGN SYMBOLIC 'B' VALUES BETWEEN BOUNDARY CONDITIONS %%%%%
B3=B(1);
B4=B(2);
B5=B(3);
B6=B(4);

clear t
for i = 1:intvl;
    ANS(i,1) = subs(P2(i));
end
ANS
%%%%%%%%%%%%%%PLOT APPROXIMATE SOLUTION%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2)
if intvl == 1
    for j = 1:K
        xfin = 0:0.01:L;
        t = xfin;
        FINANS = subs(ANS)/L^4;
        COMPARE = FINANS;
        plot(xfin,FINANS,'b');title('APPROXIMATE
SOLUTION');xlabel('X');ylabel('Y');
        hold on
```

```
    end
else
    for i = 0:intvl - 1
        xfin  = T(K+i)*L:0.01:T(K+1+i)*L;
        t = xfin / L^4;
        i = i + 1;
        FINANS = subs(ANS(i,1));
        COMPARE(i,1:length(xfin)) = FINANS;
        plot(xfin,FINANS,'b');title('APPROXIMATE
SOLUTION');xlabel('X');ylabel('Y');
        hold on
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT BASIS FUNCTIONS
figure(1)
if intvl == 1
    for j = 1:K
        t = 0:0.01:1;
        Q = subs(N(j,1));
        plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N');
        hold on
    end
else
    for i = 0:intvl - 1
        t  = T(K+i):0.01:T(K+1+i);
        i = i + 1;
        for j = 1:K+intvl-1
            Q = subs(N(j,i));
            plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N');
            hold on
        end
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT EXACT SOLUTION %%
figure(3)
plot(x,exact,'r');title('EXACT SOLUTION');xlabel('X');ylabel('Y');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% COMPARE RESULTS %%%%%%%%%%%%%%%%%%%%%%%%%%
m = K+1;
if intvl == 1
    for i = 1:length(XCOM)
        t = XCOM(i);
        CALCANS(1,i) = subs(ANS)/L^4;
        EXACTANS(1,i) = subs(EXACTt);
    end
else
    for i = 1:length(XCOM)
        if XCOM(i)<=T(m)
            m;
            t = XCOM(i);
            CALCANS(1,i) = subs(ANS(m-K))/L^4;
            EXACTANS(1,i) = subs(EXACTt);
        else
            m = m + 1;
            t = XCOM(i);
            CALCANS(1,i) = subs(ANS(m-K))/L^4;
```

```
            EXACTANS(1,i) = subs(EXACTt);
        end
    end
end

fprintf(1,'\nComputed Solutions With Error\n');
fprintf(1,'    No      x      Approx       Exact       %%RelErr\n');
fprintf(1,' ------  ------ ------------ ------------ ------------\n');
for j = 1:length(XCOM)
        err = 100*abs(1 - CALCANS(j)/EXACTANS(j));
        E(j)=err;
        fprintf(1,'%6.0f %6.2f %12.4e %12.4e
%12.4e\n',j,XCOM(j),CALCANS(j),EXACTANS(j),err);
end
NO = length(E);
E(1) = [];
ESQR = E.^2;
RMSS = sqrt(sum(ESQR)/NO);
fprintf(1,'\nComputed Root Mean Sum of the Squares Error\n');
fprintf(1,'%12.5e\n',RMSS);

function out = iszero(num,den)
if den == 0
    out = 0;
else
    out = num/den;
end
return
```

Matlab Code for Geometrically Non-linear Beam Problem, Load Case 5

```
% Jason Magoon
% SEPTEMBER 19, 2009
% NONLINEAR CANTILEVER BEAM PROBLEM
% LOAD CASE 5

clear all
clc
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED SYMBOLIC VARIABLES
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16
t = sym('t');
s = sym('s');
%% %%%%%%%%%%%%%%%% REQUIRED INPUTS %%%%%%%%%%%%%%%%%%%%%%%%%%
K = 4;                                  % ORDER
ODE = 2;                                % DEGREE OF ODE
T = [0 0 0 0 1/2 1 1 1 1];              % KNOT VECTOR
Binit = [B1 B2 B3 B4 B5];               % # of B'S = K + intvl - 1
L = 100;                                % UNITS: IN
LOAD = 1.0;                               % UNITS: POUNDS
ELASTICITY = 29000000;                  % ASTM-A36, UNITS: PSI
INERTIA = .00003448;                     % UNITS: IN^4
        %%%%% PRE-CALCULATED BOUNDARY CONDITIONS %%%%%
%%ALSO MODIFY LINES 147 - 169%%%%%
B1 = 0;
%B5 = B4;
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REQUIRED STARTING VARIABLES
intvl = 0;
m = length(T);
n = K - 1;
p = 1; d = 1; e = K; f = 2; h = K; r = 0;
X = zeros(m-n+1,1);
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE NUMBER OF INTERVALS IN KNOT VECTOR
for i = 1 : m - 1
    if T(i) < T(i+1)
        count = 1;
    else
        count = 0;
    end
    intvl = intvl + count;
    i = i + 1;
end
intvl;
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CREATE SYMBOLIC 'N' ARRAY
for i = 1:K + intvl
    for j = 1:intvl
    N(i,j*K) = t;
    end
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE BASIS FUNCTIONS 'N'
for k = 1:intvl
    for i = 1:K + intvl
        for j = d
```

```
            if i==e
                N(i,j) = 1;
            else
                N(i,j) = 0;
            end
        end
    end
    d = d + K;
    e = e + 1;
end
for k = 1:intvl
    for j = 2:K
        for i = 1:K + intvl - 1
            lh = iszero(t - T(i),T(i+j-1) - T(i));
            rh = iszero(T(i+j) - t,T(i+j) - T(i+1));
            N(i,j+r) = ((lh) * N(i,j+r-1)) + ((rh) * N(i+1,j+r-1));
        end
    end
        r = r + K;
end
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ADDITIONAL MATRIX OPERATIONS
N(K+intvl,:) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
%%%%%%%%%% DELETING ALL NON'K'MULTIPLE COLUMNS OF 'N'%%%%%%%%%%
for j=1:intvl*K
        I(1,j) = j;
        j = j + 1;
end
for k = intvl:-1:1
    I(:,k*K) = [];
end
N(:,I) = [];
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% GREVILLE ABSCISSAE CALCULATION
for i = 1:m-n+1
    while i < m-n+2
        TG = sum(T(i:i+n-1));
        X(i)=(1/n)*TG;
        i = i + 1;
    end
end
%% %%%%%%%%%% DELETE FIRST 2 AND LAST 2 VALUES OF X %%%%%%%%%%%%%%
X(m-n+1) = [];          %DELETE EXTRA ROW USED FOR CALCULATION
X(m-n) = [];            %DELETE EXTRA ROW USED FOR CALCULATION
X(2) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
X(1) = [];              %DELETE EXTRA ROW USED FOR CALCULATION
X
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%ADD DERIVATIVES TO 'N'
DLIM = intvl * ODE;
i = 1;
for j = intvl + 1:intvl + DLIM
        N(:,j) = diff(N(:,i));
        i = i + 1;
end
N
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CALCULATE 'P'
```

```
clear t
PEQ = Binit * N;
Pinit = subs(PEQ);
P2 = transpose(Pinit);

for h = 1:intvl
    phi(h,1) = P2(h,1);
end
phi;
%%%%%%%%%%%%%%%%%%CALCULATE 'B' VALUES%%%%%%%%%%%%%%%%%%%%%%%%
         %%%%% DIFFERENTIAL EQUATION PARAMETERS %%%%%
% E*P''+ H = 0
E = 1;
for i = 1:intvl
    H(i,1) = (LOAD/(ELASTICITY*INERTIA))*cos(phi(i));
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i = 1:intvl
    EQ(i,1) = (E*P2(i+ODE*intvl))/L^2 + H(i);
end
EQ;
VARCNT = length(X);
UNKCNT = length(Binit)-2;
i = 1;
for k = 1:VARCNT
        if X(k) <= T(K+i)
            t = X(k);
            eq(k) = subs(EQ(i,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)= H(i) + (E*Nf(j+1,i+2*intvl))/L^2;
            end
        else
            i = i+1;
            t = X(k);
            eq(k) = subs(EQ(i,1));
            Nf = eval(N);
            for j = 1:UNKCNT
                A(k,j)=H(i) + (E*Nf(j+1,i+2*intvl))/L^2;
            end
        end
end
B1=0;B2=0;B3=0;B4=0;B5=0;B6=0;B7=0;B8=0;B9=0;B10=0;
remain = transpose(-eval(eq));
B = inv(A)*remain;
eq
%%%%%%%%%%%%%%SUBSTITUTE 'B' VALUES INTO 'P'%%%%%%%%%%%%%%%%%%%
syms B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15 B16
%%%% ASSIGN SYMBOLIC 'B' VALUES BETWEEN BOUNDARY CONDITIONS %%%%%
%%%%% FROM NONLINEAR SOLVER %%%%%%
B2 = .8049;
B3 = 1.4119;
B4 = 1.4418;
B5 = B4;

clear t
for i = 1:intvl;
    ANS(i,1) = subs(P2(i));
```

```
end
ANS;
%%%%%%%%%%%%%PLOT APPROXIMATE SOLUTION%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(2)
v=1;
if intvl == 1
    for j = 1:K
        xfin = 0:0.01:L;
        t = xfin / L;
        FINANS = subs(ANS);
        plot(xfin,FINANS,'k');title('LOAD CASE 5: P = 1.00 lbs');xlabel('ARC
LENGTH(in)');ylabel('SLOPE(radians)');
        hold on
    end
else
    for i = 0:intvl - 1
        xfin  = T(K+i)*L:0.01:T(K+1+i)*L;
        t = xfin / L;
        i = i + 1;
        FINANS = subs(ANS(i,1));
        plot(xfin,FINANS,'k');title('SLOPE VS. ARCLENGTH');xlabel('ARC
LENGTH(in)');ylabel('SLOPE(radians)');
        v = v + 1;
        hold on
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% PLOT BASIS FUNCTIONS
figure(1)
if intvl == 1
    for j = 1:K
        t = 0:0.01:1;
        Q = subs(N(j,1));
        plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N(t)');
        hold on
    end
else
    for i = 0:intvl - 1
        t  = T(K+i):0.01:T(K+1+i);
        i = i + 1;
        for j = 1:K+intvl-1
            Q = subs(N(j,i));
            plot(t,Q);title('BASIS FUNCTIONS');xlabel('t');ylabel('N(t)');
            hold on
        end
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% COMPARE RESULTS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
j = 1;
BS =length(FINANS);
CALCANS = FINANS(BS);
exact = 82*(pi/180);

fprintf(1,'\nComputed Solutions With Error\n');
fprintf(1,'    No     approx       exact       %%RelErr\n');
fprintf(1,' ------ ------------ ------------ ------------\n');
err = 100*abs(1 - CALCANS(j)/exact(j));
```

```
fprintf(1,'%6.0f %12.5f %12.5f %12.4e\n',j,CALCANS(j),exact(j),err);

t = s/L;
FINEQ = subs(ANS)

function out = iszero(num,den)
if den == 0
    out = 0;
else
    out = num/den;
end
return
```

Non-Linear Matlab Function for Example Problem 6

```
function F = myfun(x)
P = 1.00;
F = [((-
11/5000)*x(1)+(1/2500)*x(2)+(1/5000)*x(3))+(P/1000)*cos(0.5648*x(1)+0.1296*x
(2)+.009259*x(3));
     (3/5000)*x(1)-
(3/2500)*x(2)+(3/5000)*x(3)+(P/1000)*cos(0.25*x(1)+0.5*x(2)+.25*x(3));
     (1/5000)*x(1)+(1/2500)*x(2)-
(3/5000)*x(3)+(P/1000)*cos(.009259*x(1)+0.1296*x(2)+.8611*x(3))];
```

Command Lines Needed in Matlab

```
x0=[0;1;1.4];
options = optimset('TolFun',.00000000000000000001);
options = optimset('Display','iter');
[x,fval] = fsolve(@myfun,x0,options)
```