A COMPARATIVE STUDY OF COLLOCATION METHODS FOR THE NUMERICAL SOLUTION OF DIFFERENTIAL EQUATIONS

By

MARGARET MODUPE KAJOTONI

Submitted in fulfillment of the

academic requirements for the degree of

Master of Science in the

School of Mathematical Sciences

University of KwaZulu-Natal

Durban

October 2008

Abstract

The collocation method for solving ordinary differential equations is examined. A detailed comparison with other weighted residual methods is made. The orthogonal collocation method is compared to the collocation method and the advantage of the former is illustrated. The sensitivity of the orthogonal collocation method to different parameters is studied. Orthogonal collocation on finite elements is used to solve an ordinary differential equation and its superiority over the orthogonal collocation method is shown. The orthogonal collocation on finite elements is also used to solve a partial differential equation from chemical kinetics. The results agree remarkably with those from the literature.

Dedicated to my parents

Mr and Mrs R.O. Kajotoni

Acknowledgments

I will remain grateful to the founder of AIMS, professor Neil Turok and its director, professor Fritz Hahne for giving me such an exceptional opportunity to realize my dreams. My gratitude also goes to the entire AIMS family. I acknowledge the University of KwaZulu-Natal research office and professor Jacek Banasiak for also providing funding for this work.

I feel indebted to my supervisor, Dr P. Singh for all his reliable guidance and commitments throughout the course of my study. His approach to problem solving will definitely help me in future. I also appreciate the enormous contribution of my co-supervisor, Dr N. Parumasur.

Many thanks to my parents, Mr and Mrs R.O. Kajotoni for their expression of love. I also acknowledge the contributions of my brothers Kayode and Tunde and my sister Mrs Ojerinde and her family. To the Eniolorunda's and Kajotoni's home and abroad, I say thank you all.

My irrevocable gratitude goes to my beloved fiance Raphael Olushola Folorunsho for his love, patience and tactfulness throughout my stay in South Africa. Many thanks to the Folorunsho's for all their love and concern.

My appreciation also goes to Faye, Salvie, Mrs Petro, Mrs Henning, Dr P. Pillay, Dr Moopanar, prof Baboolal Dharmanand, Dr Kutu and family, Annemarie, Soren, Simon, Lucie, Mary, Emmanuel, Alfred, Elmah, Rudo, Ivy, Iyabo, Lola and all my colleagues in the School of Mathematical Sciences for all their love and hospitality.

More so, I will not forget all my lovely friends both home and abroad, His People Christian Church Durban and The Redeemed Christian Church of God(RCCG) Durban for being part of my life and for all their kind gestures which has led to the successful completion of my study.

My utmost gratitude goes to my Lord and my personal saviour, Jesus Christ for enabling me accomplish this dissertation.

Preface

I Margaret Modupe Kajotoni affirm that the work in this thesis was carried out in the School of Mathematical Sciences, University of KwaZulu-Natal, Durban, during the period of August 2007 to October 2008. It was completed by the author under the supervision of Dr P. Singh and co-supervised by Dr N. Parumasur.

The research contained in this thesis has not been submitted to any University nor has it been published previously. Where use was made of the work of others it has been duly acknowledged in the text.

MM Kajotoni
October 2008

Declaration-Plagiarism

I Margaret Modupe Kajotoni declare that

1. The research reported in this thesis, except where otherwise indicated, is my original research.

2. This thesis has not been submitted for any degree or examination at any other university.

3. This thesis does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.

4. This thesis does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
   a. Their words have been re-written but the general information attributed to them has been referenced.
   b. Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.

5. This thesis does not contain text, graphs or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the Reference sections.

Signed: Date:

Notation

L	Linear differential operator
R	Residual
ϕ_i	Basis function
$w(x)$	Weight function
K	Condition number
X	Domain of interest
$C(X)$	Continuous function over the domain X
y	Exact solution to differential equations
y_a	Approximate solution to weighted residual methods
e_a	Error from the approximate solution y_a
y_c	Approximate solution by the collocation method
y_s	Approximate solution by the sub-domain method
y_l	Approximate solution by the least square method
y_m	Approximate solution by the moment method
y_g	Approximate solution by the Galerkin method
$\|\cdot\|_2$	Euclidean norm in $\mathbb{R}^n$
$\langle\ ,\ \rangle$	Inner product on $C(X)$
N	Order of polynomial
T_N	Chebyshev Polynomial of the first kind of order N
U_N	Chebyshev Polynomial of the second kind of order N
P_N	Legendre Polynomial of order N

x_k	Interpolation nodes
l_k	Lagrange polynomial function
$\psi(x)$	Nodal polynomials
N_e	Number of finite elements
h_i	Length of the i_{th} element
y_i	Approximate solutions using equally spaced finite elements
e_i	Error from the approximate solution y_i
$\bar{y}_i$	Approximate solutions using unequally spaced finite elements
$\bar{e}_i$	Error from the approximate solution $\bar{y}_i$
y_e	Exit solution
P	Peclet number

List of Tables

List of Figures

Contents

Introduction

Differential equations are used for mathematical modeling in science and engineering. However, it is apparent that there may be no known analytic solution to such differential equations and even when it does exist it might be very difficult to obtain. In such cases, the numerical approximations are used. Of special interest are numerical techniques that reduce excessive demand for computational time and at the same time giving a close approximation to the exact solution.

In this thesis, we present a simple but effective numerical technique called the orthogonal collocation method for solving differential equations.

Chapter 1 gives an overview of the five different methods of weighted residuals. We introduce the orthogonal collocation method in chapter 2. Chapter 3 explains the features of the orthogonal collocation method on finite elements and it is then used to solve an ODE. Chapter 4 gives a practical example of the orthogonal collocation on finite elements applied to a PDE problem in chemical engineering. Each chapter gives the details of the numerical procedures used.

Chapter 1

Method of Weighted Residuals

1.1 Introduction

The method of weighted residuals was used to find approximate solutions to differential equations before the finite element method came into existence [21]. The method was introduced by Crandall [16] in 1956 . Other scientists like Finlayson and Scriven (1966) [23], Vichnevetsky (1969) [71] and Finlayson (1972) [22] have also used this method. The method is used to solve boundary value problems arising from fluid flow, structural mechanics and heat transfer. It is well known because of the interactive nature of the first step. The user provides an initial guess for the solution which is then forced to satisfy the governing differential equations and its boundary conditions. The guessed solution is chosen to satisfy the boundary conditions, however it does not necessarily satisfy the differential equations [28, 59].

The method of weighted residuals requires two types of functions namely the trial functions and the weight functions. The former is used to construct the trial solution while the latter is used as a criterion to minimize the residual [21].

Consider a linear differential equation

$$L\ y(x) = f(x), \tag{1.1}$$

together with k boundary conditions. Here L is a bounded linear differential operator and $f(x) \in C(X)$, where X denotes the domain. The unknown solution $y(x)$ is approximated by a function $y_a(x)$ and we write

$$y(x) \approx y_a(x) = \sum_{i=1}^{N} c_i \phi_i(x), \tag{1.2}$$

where the set $\{\phi_i(x) | i = 1, 2, ..., N\}$ constitutes a basis for a finite dimensional subspace of $C(X)$ and the c_i's are constant coefficients. When substituted into the differential equation (1.1) we obtain

$$L\ y_a(x) = f_a(x), \tag{1.3}$$

where in general

$$f_a(x) \neq f(x). \tag{1.4}$$

Subtracting equation (1.1) from equation (1.3) yields the residual equation

$$R(x) = L\,(y_a(x) - y(x)) = L\ y_a(x) - L\ y(x) = L\ y_a(x) - f(x) \neq 0. \tag{1.5}$$

From equation (1.5), it is easy to show that

$$\|y_a(x) - y(x)\| = \|L^{-1}R(x)\| \leq \frac{\|L\|\|L^{-1}\|}{\|L\|}\|R(x)\| = \frac{K}{\|L\|}\|R(x)\|, \tag{1.6}$$

where K is the condition number of L and $\|\cdot\|$ denotes compatible operator and function norms on $C(X)$. Hence for a well conditioned problem, the closer the residual is to zero, the better is the approximate solution. The unknown coefficients c_i's are chosen to force the residual to zero in some average sense over the domain X by requiring that

$$\int_X R(x) w_i(x)\ dx = 0, \quad i = 1, 2, ..., N - k, \tag{1.7}$$

where the weight functions $w_i(x)$ are specified and we require $N - k$ of them to yield $N - k$ equations which together with k boundary conditions are sufficient to determine the unknowns c_i.

1.2 Variation on Theme of Weighted Residuals

There are five widely used variations of the method of weighted residuals for engineering and science applications, namely

1. The collocation method,
2. The sub-domain method,
3. The least squares method,
4. The moment method,
5. The Galerkin method.

The distinguishing factor between these methods is the choice of the weight functions used in minimizing the residual. We shall briefly discuss the different features of each method.

1.2.1 Collocation Method

In this method, the weight functions are taken from the family of the Dirac delta (δ) functions in the domain X, that is $w_i(x) = \delta(x - x_i)$. The Dirac delta function is defined by [47]

$$\delta(x - x_i) = \begin{cases} 1 & x = x_i, \\ 0 & \text{otherwise} \end{cases} \tag{1.8}$$

and has the property that

$$\int_X f(x)\delta(x - x_i)dx = f(x_i). \tag{1.9}$$

Hence the integration of the weighted residual (1.7) results in forcing the residual to zero at specific points in the domain. That is, with this choice of the weight function, equation (1.7) reduces to

$$\int_X R(x)\delta(x - x_i)dx = R(x_i) = 0, \quad i = 1, 2, ..., N - k. \tag{1.10}$$

As the number of collocation points x_i increases, we satisfy the differential equation at more points and hence force the approximate solution to approach the exact solution. The ease of performing integrals with the Dirac delta function is an added advantage of the collocation method.

The collocation method was first used by Frazer et al [24] in 1937 . Thereafter Bickley [7] used it in 1941 along with the least squares method and the Galerkin method to solve unsteady heat condition problems. In 1962, Jain introduced an extremal-point collocation by sampling the residual at the zeros x_i of the Chebyshev polynomials and requiring that

$$R(x_{i+1}) - (-1)^i R(x_1) = 0, \quad i = 1, 2, ..., N - k. \tag{1.11}$$

He chose the zeros of the Chebyshev polynomials because they are known to minimize the maximum error [30]. Jain combined this method with the Newton's method to solve a viscous fluid problem.

In 1963, Schetz applied the low-order collocation method to a number of boundary-layer problems [63]. In 1975, Panton and Sallee applied a collocation method to problems of unsteady heat conduction and boundary layer flows [52]. They used B-splines as trial functions and compared their results with results obtained from the finite-difference method. The results were more accurate than those obtained from the finite-difference method. Viviand and Ghazzi in 1974 applied a collocation method to the problem of computing the viscid flow about an inclined three-dimensional wing [74].

The collocation method has been used over a wide range of problems in recent times [15, 32, 68, 70].

1.2.2 Sub-domain Method

Here the weight function is always $w(x) \equiv 1$. The domain is split into $N - k$ subsections X_i called sub-domains sufficient to evaluate the unknown parameters

c_i, hence

$$\int_X R(x)w_i(x)dx = \sum_{i=1}^{N-k}\left(\int_{X_i} R(x)dx\right). \tag{1.12}$$

This method forces the residual to be zero over the various sub-domains, that is

$$\int_{X_i} R(x)dx = 0. \tag{1.13}$$

This method originated from the work of Biezeno and Koch in 1923 [8]. Pallone [51] and Bartlett [5] used it to solve Laminar boundary layers and Murphy has used it to solve incompressible Navier-Stokes equations [46] .

1.2.3 Least Squares Method

In this method, we minimize the Euclidean norm of the residual, namely

$$S = \|R(x)\|_2^2 = \int_X R^2(x)dx. \tag{1.14}$$

To obtain the minimum of this scalar function, the derivatives of S with respect to the unknown parameters c_i must be zero. This yields,

$$\int_X R(x)\frac{\partial R(x)}{\partial c_i}dx = 0. \tag{1.15}$$

Comparing equation (1.7) and equation (1.15), the weight functions for the least squares method are identified as the derivatives of the residual with respect to the unknown constants:

$$w_i(x) = \frac{\partial R(x)}{\partial c_i}. \tag{1.16}$$

The least squares method is the oldest of all the methods of weighted residuals [25]. Crandall stated in his work in 1956 that "*the least square method was discovered by Gauss in* 1775" [16]. In 1937, Frazer et al [24] also used this method.

1.2.4 Moment Method

In this method, the weight functions are chosen from the family of polynomials. That is

$$w_i(x) = x^{i-1},\ i = 1, 2, ..., N-k. \tag{1.17}$$

In 1948, Yamada applied this method to a nonlinear diffusion problem [77]. His formulation is described in Ames [2].

1.2.5 Galerkin Method

This method is a modification of the least squares method. The weight functions are chosen as the derivative of the approximating function with respect to c_i, that is

$$w_i(x) = \frac{\partial y_a(x)}{\partial c_i}. \tag{1.18}$$

From equation (1.2), it is easily seen that $w_i(x)$ is identical to the basis function $\phi_i(x)$. Hence the unknowns c_i are determined so that the residual $R(x)$ is perpendicular to the subspace spanned by $\{\phi_{i_p}(x)|p = 1, 2, ..., N-k\}$ where $\{i_p|p = 1, 2, ..., N-k\} \subseteq \{1, 2, ..., N\}$.

This method was first introduced by Galerkin in 1915 [26]. Duncan used it to study the dynamics of aeronautical structures [19, 18]. It was later used by Bickley to solve unsteady heat conduction problems [7].

1.3 Numerical Examples

We illustrate the five variations of the method of weighted residuals with some examples.

Example 1.3.1 *Consider the differential equation defined on* $[0, 1]$ *with its boundary conditions which represents a diffusion-reaction equation in chemistry given by*

$$\begin{aligned} \frac{d^2y}{dx^2} + \frac{dy}{dx} - \alpha^2 y &= 0, \\ y'(0) &= 0, \\ y(1) &= 1, \end{aligned} \tag{1.19}$$

where α is a constant parameter that denotes the reaction rate which is termed slow for small values of α and fast for larger α's. The solution $y(x)$ describes the concentration of a chemical substance at distance x.

The exact solution to this equation is given by

$$y(x) = \frac{e^{\left(\frac{1}{2}-\frac{x}{2}\right)}\left(2\beta\cosh(\beta x) + \sinh(\beta x)\right)}{2\beta\cosh(\beta) + \sinh(\beta)}, \tag{1.20}$$

where $\beta = \frac{\sqrt{4\alpha^2+1}}{2}$. We find an approximate solution to equation (1.19), with $\alpha = 1$ using the five different methods of weighted residuals.

1.3.1 Two Terms Trial Solution for a Slow Reaction Rate

Let us pick the trial solution from a subspace of dimension two of $C[0,1]$ as

$$y_a(x) = c_1 + c_2(1-x^2). \tag{1.21}$$

Here $\phi_1(x) = 1$ and $\phi_2(x) = (1-x^2)$ are the basis functions. The requirement that $y_a(x)$ satisfies the boundary condition $y(1) = 1$ implies that $c_1 = 1$. It is noted that the boundary condition $y'(0) = 0$ is automatically satisfied by $y_a(x)$ because of the clever choice of the basis function $\phi_2(x)$. We therefore need one more condition to evaluate c_2.

Substituting the trial solution in equation (1.21) into the differential equation (1.19) gives the residual (since $f(x) = 0$):

$$R(x) = -2c_2 - 2c_2x - (1 + c_2(1-x^2)). \tag{1.22}$$

Substituting the residual in equation (1.22) into equation (1.7) gives

$$\langle R(x), w_1(x)\rangle = \int_0^1 \left[-2c_2 - 2c_2x - (1 + c_2(1-x^2))\right] w_1(x)dx = 0, \tag{1.23}$$

where $\langle\ ,\ \rangle$ denotes the inner product on $C[0,1]$. We evaluate c_2 by choosing different weight functions which depends on the method of weighted residuals used.

1. **The Collocation Method**

   The test function is $w_1(x) = \delta(x - x_1)$, where x_1 is the unknown collocation point which must be chosen from the domain $[0, 1]$. So equation (1.23) becomes

   $$\langle R(x), w_1(x)\rangle = \int_0^1 \left[-2c_2 - 2c_2x - (1 + c_2(1 - x^2))\delta(x - x_1)\right] dx. \quad (1.24)$$

   Applying the property of the Dirac delta function (see equation (1.10)) gives

   $$-2c_2 - 2c_2x_1 - (1 + c_2(1 - x_1^2)) = 0, \quad (1.25)$$

   and simplifying yields $c_2 = \frac{-1}{\left(2+2x_1+(1-x_1^2)\right)}$. Choosing $x_1 = 0.5$, we obtain $c_2 = -4/5$ and the trial solution by the collocation method is,

   $$y_c(x) = 1 - 4(1 - x^2)/5. \quad (1.26)$$

2. **Sub-domain Method**

   Since we have only one unknown, we only need one sub-domain which is the whole domain $[0, 1]$. Hence the weight function is $w_1(x) = 1$, for $0 \leq x \leq 1$ and equation (1.23) becomes

   $$\langle R(x), w_1(x)\rangle = \int_0^1 \left[-2c_2 - 2c_2x - (1 + c_2(1 - x^2))\right] dx = 0. \quad (1.27)$$

   Integrating with respect to x and evaluating gives $c_2 = -3/11$.

   Hence the trial solution by the sub-domain method is

   $$y_s(x) = 1 - 3(1 - x^2)/11. \quad (1.28)$$

3. **The Least Squares Method**

   The weight function is $w_1 = \frac{\partial R}{\partial c_2}$ and equation (1.23) becomes

   $$\int_0^1 R\frac{\partial R}{\partial c_2}dx = \frac{1}{2}\frac{\partial}{\partial c_2}\int_0^1 \left[-2c_2 - 2c_2x - (1 + c_2(1 - x^2))\right]^2 dx = 0. \quad (1.29)$$

   Integrating with respect to x and then differentiating with respect to c_2 we obtain $c_2 = -55/203$. Hence the trial solution by the least squares method is

   $$y_l(x) = 1 - 55(1 - x^2)/203. \quad (1.30)$$

4. **The Moment Method**

   The weight function is $w_1 = x^0 = 1$ which is similar to that of the sub-domain method. Since only two terms are retained with one unknown, the solution will be the same as that of the sub-domain method, hence

$$y_m(x) = 1 - 3(1 - x^2)/11. \tag{1.31}$$

5. **Galerkin Method**

   The weight function is the same as that of the trial function, so we choose $w_1(x) = (1 - x^2)$. Thus equation (1.23) becomes

$$\langle R(x), w_1(x)\rangle = \int_0^1 \left[-2c_2 - 2c_2x - (1 + c_2(1 - x^2))\right](1 - x^2)dx. \tag{1.32}$$

   Integrating and evaluating gives $c_2 = -20/71$. Hence the trial solution by the Galerkin method is

$$y_g(x) = 1 - 20(1 - x^2)/71. \tag{1.33}$$

The plots of the solutions and the errors for the above five methods are given in Figures 1.1 and 1.2 respectively.

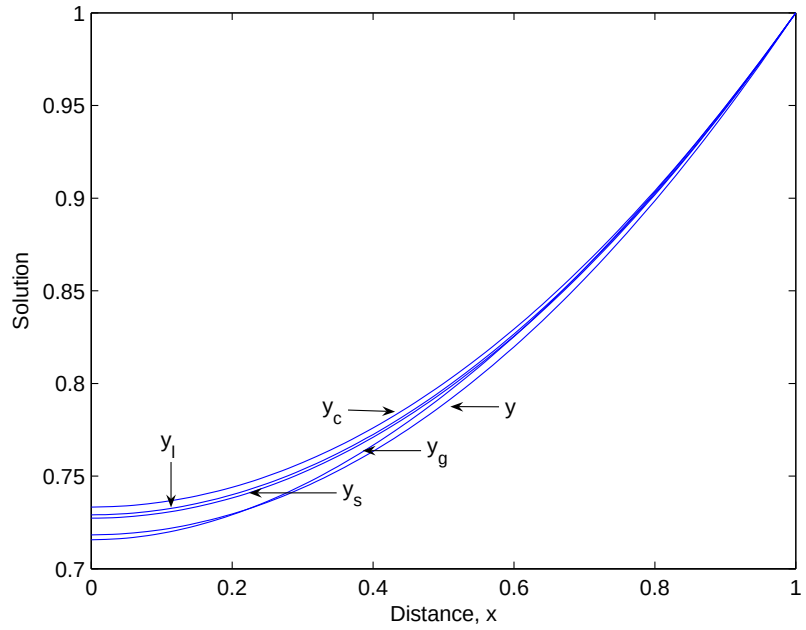

Figure 1.1: Comparison of y and y_a for Example 1.3.1.

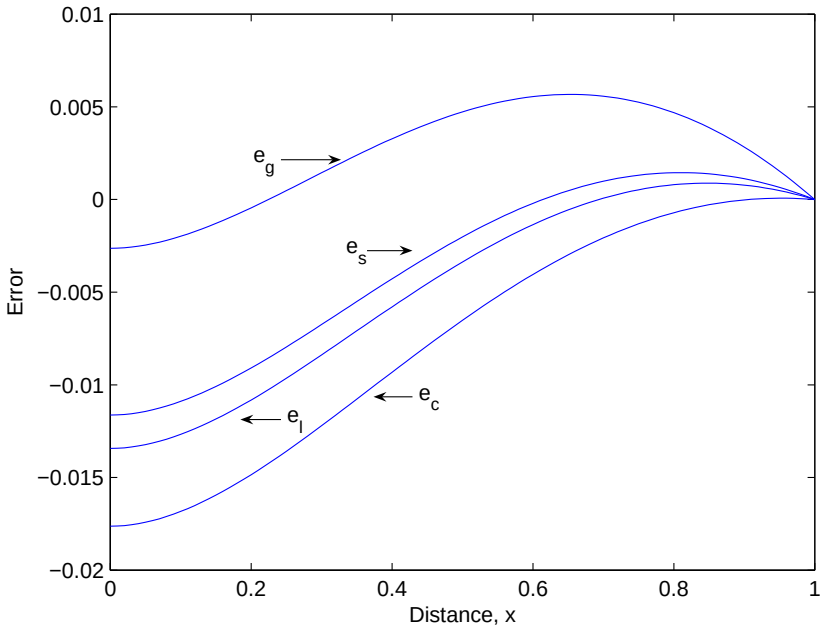

Figure 1.2: Errors $e_a = y - y_a$ for Example 1.3.1.

We will sample the exact solution $y(x)$ and the approximate solutions $y_a(x)$ at a discrete set of twenty one equally spaced points $x_i,\ i = 1, 2, ..., 21$ and denote these vectors by $\bar{y}(x)$ and $\bar{y_a}(x)$. We define the total error by

$$\text{total error } = \|\bar{y}(x) - \bar{y_a}(x)\|_2, \tag{1.34}$$

where $\|\cdot\|_2$ denotes the Euclidean norm in $\mathbb{R}^{21}$.

Table 1.1 gives the approximations and the total error for Example 1.3.1 for different values of x.

The results of the numerical solutions for different values of x agree closely with the analytic solutions (see Figure 1.1). Moreover, all the methods show small errors (see Figure 1.2) and the solution values agree closely with the exact solution (see Table 1.1) even though only two terms were retained in the trial solution. This is because of the slow reaction rate $\alpha = 1$. Hence the concentration profile of the chemical substance is shallow and can easily be described by only two terms. From Table 1.1, we see that the Galerkin method has a slightly lower total error and hence gives the best approximation to this particular problem. If we allow a higher reaction rate, the concentration profile becomes steep and the trial solution with two terms is unable to track the solution. Let us illustrate this by picking $\alpha = 10$ in equation (1.19).

x	y	y_c	y_s	y_l	y_g
0	0.7157	0.7333	0.7273	0.7291	0.7183
0.0500	0.7165	0.7340	0.7280	0.7297	0.7190
0.1000	0.7191	0.7360	0.7300	0.7318	0.7211
0.1500	0.7233	0.7393	0.7334	0.7352	0.7246
0.2000	0.7291	0.7440	0.7382	0.7399	0.7296
0.2500	0.7364	0.7500	0.7443	0.7460	0.7359
0.3000	0.7451	0.7573	0.7518	0.7534	0.7437
0.3500	0.7552	0.7660	0.7607	0.7623	0.7528
0.4000	0.7666	0.7760	0.7709	0.7724	0.7634
0.4500	0.7794	0.7873	0.7825	0.7839	0.7753
0.5000	0.7935	0.8000	0.7955	0.7968	0.7887
0.5500	0.8087	0.8140	0.8098	0.8110	0.8035
0.6000	0.8253	0.8293	0.8255	0.8266	0.8197
0.6500	0.8430	0.8460	0.8425	0.8435	0.8373
0.7000	0.8619	0.8640	0.8609	0.8618	0.8563
0.7500	0.8820	0.8833	0.8807	0.8815	0.8768
0.8000	0.9033	0.9040	0.9018	0.9025	0.8986
0.8500	0.9257	0.9260	0.9243	0.9248	0.9218
0.9000	0.9493	0.9493	0.9482	0.9485	0.9465
0.9500	0.9741	0.9740	0.9734	0.9736	0.9725
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	Total error $\Rightarrow$	**0.0455**	**0.0274**	**0.0325**	**0.0165**

Table 1.1: Numerical comparison of the different approximations in Example 1.3.1.

1.3.2 Two Terms Trial Solution for a Fast Reaction Rate

We now solve equation (1.19) with a fast reaction rate relative to diffusion.

Example 1.3.2 *Using only two terms as in Example* 1.3.1 *and picking* $\alpha = 10$ *in Equation (*1.19*), our residual becomes*

$$R(x) = -2c_2 - 2c_2x - 100\left(1 + c_2(1 - x^2)\right). \tag{1.35}$$

We will again apply the five different weighted residuals methods as in Example 1.3.1. The summary of the approximate solutions using the five different methods is given in Table 1.2. Hence forth we will approximate the coefficients to four decimal places.

Method	**Approximate Solution**
Collocation	$y_c(x) = 1 - 1.2821(1 - x^2)$
Sub-domain	$y_s(x) = 1 - 1.4354(1 - x^2)$
Least squares	$y_l(x) = 1 - 1.2202(1 - x^2)$
Moment	$y_m(x) = 1 - 1.4354(1 - x^2)$
Galerkin	$y_g(x) = 1 - 1.2085(1 - x^2)$

Table 1.2: Approximate solutions for Example 1.3.2.

The plots of the solutions and their errors are shown in Figures 1.3 and 1.4 respectively. A table of values resulting from the different approximations is shown in Table 1.3.

We notice from Figure 1.3 that the approximate solutions deviate sharply from the exact solution hence the error between the approximate and the exact solution is large (See Figure 1.4). This is due to the sharp concentration profile. We notice from Table 1.3 that even the Galerkin method which on the average is the best approximation is still not a very good approximation to the exact solution. We

can reduce the error by increasing the dimension of the approximation space. In general, the more terms used in the trial solution, the better the accuracy but the computation becomes expensive.

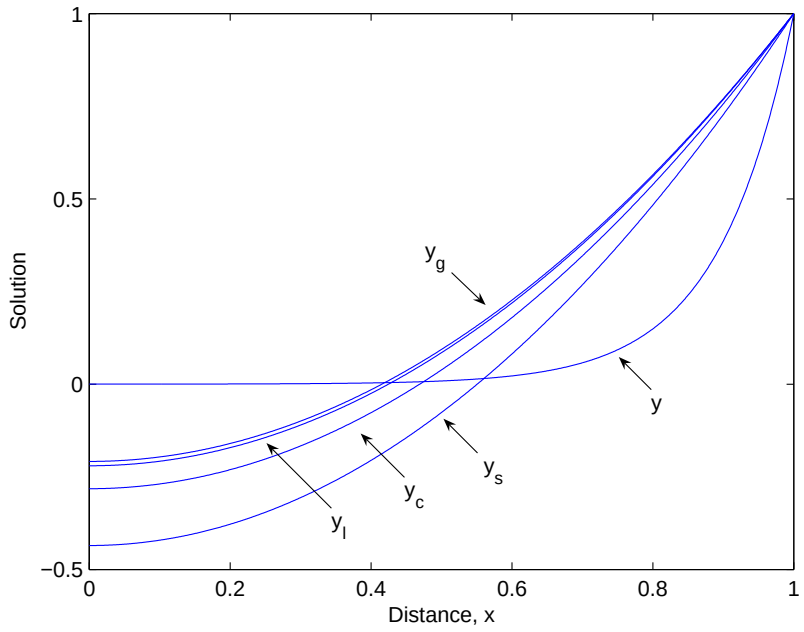

Figure 1.3: Comparison of y and y_a for Example 1.3.2.

x	y	y_c	y_s	y_l	y_g
0	0.0001	−0.2821	−0.4354	−0.2202	−0.2085
0.0500	0.0002	−0.2789	−0.4318	−0.2171	−0.2055
0.1000	0.0002	−0.2693	−0.4210	−0.2080	−0.1964
0.1500	0.0003	−0.2533	−0.4031	−0.1927	−0.1813
0.2000	0.0005	−0.2308	−0.3780	−0.1714	−0.1602
0.2500	0.0008	−0.2020	−0.3457	−0.1439	−0.1330
0.3000	0.0013	−0.1667	−0.3062	−0.1104	−0.0997
0.3500	0.0021	−0.1250	−0.2596	−0.0707	−0.0605
0.4000	0.0033	−0.0770	−0.2057	−0.0250	−0.0151
0.4500	0.0053	−0.0225	−0.1447	0.0269	0.0362
0.5000	0.0086	0.0384	−0.0766	0.0849	0.0936
0.5500	0.0138	0.1057	−0.0012	0.1489	0.1571
0.6000	0.0223	0.1795	0.0813	0.2191	0.2266
0.6500	0.0358	0.2596	0.1711	0.2953	0.3021
0.7000	0.0576	0.3461	0.2679	0.3777	0.3837
0.7500	0.0927	0.4391	0.3720	0.4662	0.4713
0.8000	0.1492	0.5384	0.4833	0.5607	0.5649
0.8500	0.2401	0.6442	0.6017	0.6614	0.6646
0.9000	0.3863	0.7564	0.7273	0.7682	0.7704
0.9500	0.6215	0.8750	0.8600	0.8810	0.8822
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	Total error ⇒	**1.1154**	**1.3327**	**1.0854**	**1.0841**

Table 1.3: Numerical comparison of the different approximations in Example 1.3.2.

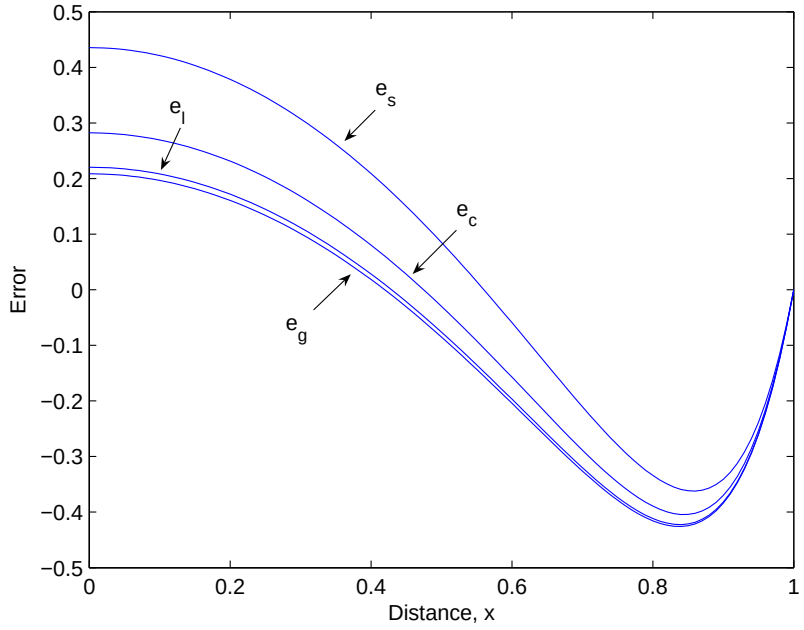

Figure 1.4: Errors $e_a = y - y_a$ for Example 1.3.2.

1.3.3 Three Terms Trial Solution for a Fast Reaction Rate

We seek to improve the accuracy of the results obtained in Example 1.3.2 by using a subspace of dimension three.

Example 1.3.3 *In order to improve the accuracy of the solution for a fast reaction rate* $\alpha = 10$*, we retain three terms in the trial solution. Hence the new trial solution is*

$$y_a = c_1 + c_2(1 - x^2) + c_3 x^2(1 - x^2), \tag{1.36}$$

where the basis functions are $\{1, 1 - x^2, x^2(1 - x^2)\}$ and $c_1 = 1$ because of the boundary condition $y(1) = 1$ in equation (1.19). The boundary condition $y'(0) = 0$ is automatically satisfied by $y_a(x)$.

The residual becomes

$$\begin{aligned} R(x) = &-2c_2 + 2c_3 - 12c_3x^2 - 2c_2x + 2c_3x - 4c_3x^3 \\ &- 100(1 + c_2(1 - x^2) + c_3x^2(1 - x^2)). \end{aligned} \tag{1.37}$$

1. **The Collocation Method**

   Since we have two unknowns, we will need two weight functions given by $w_1(x) = \delta(x - x_1)$ and $w_2(x) = \delta(x - x_2)$, where x_1 and x_2 are the unknown collocation points which we choose as $x_1 = 1/3$ and $x_2 = 2/3$. Next we evaluate the following two inner products

   $$\langle R(x), w_1(x)\rangle = \int_0^1 [R(x)\delta(x - x_1)]\, dx = 0 \tag{1.38}$$

   and

   $$\langle R(x), w_2(x)\rangle = \int_0^1 [R(x)\delta(x - x_2)]\, dx = 0, \tag{1.39}$$

   by applying the property of the Dirac delta function (equation (1.10)), and evaluating at the collocation points $x_1 = 1/3$ and $x_2 = 2/3$ to obtain

   $$R(x_1) = 0 \tag{1.40}$$

   and

   $$R(x_2) = 0. \tag{1.41}$$

   Simplifying equations (1.40) and (1.41) yields the linear system

   $$\begin{aligned} -91.5577c_2 - 8.6894c_3 &= 100 \\ -58.8845c_2 - 27.8778c_3 &= 100, \end{aligned} \tag{1.42}$$

   which we solve simultaneously to yield $c_2 = -0.9403$ and $c_3 = -1.6010$. So the trial solution by the collocation method is

   $$y_c(x) = 1 - 0.9403(1 - x^2) - 1.6010x^2(1 - x^2). \tag{1.43}$$

2. **Sub-domain Method**

   Since we have two unknowns, we will split the interval $[0, 1]$ into two sub-domains: $[0, 0.5]$ and $[0.5, 1]$. The weight functions are $w_1(x) = w_2(x) = 1$. So we will need to evaluate the following two inner products

   $$\langle R(x), w_1(x)\rangle = \int_0^{0.5} R(x)dx = 0 \tag{1.44}$$

and

$$\langle R(x), w_2(x)\rangle = \int_{0.5}^{1} R(x)dx = 0. \tag{1.45}$$

These give the linear system

$$\begin{aligned} -47.0833c_2 - 2.8542c_3 &= 50 \\ -22.5833c_2 - 12.4792c_3 &= 50, \end{aligned} \tag{1.46}$$

which we solve simultaneously to yield $c_2 = -0.9200$ and $c_3 = -2.3418$. Hence the trial solution by the sub-domain method is

$$y_s(x) = 1 - 0.9200(1 - x^2) - 2.3418x^2(1 - x^2). \tag{1.47}$$

3. **The Least Squares Method**

The weight functions are $w_1 = \frac{\partial R}{\partial c_2}$ and $w_2 = \frac{\partial R}{\partial c_3}$. So we will need to evaluate the following inner products

$$\langle R(x), w_1(x)\rangle = \int_0^1 R\frac{\partial R}{\partial c_2}dx = \frac{1}{2}\frac{\partial}{\partial c_2}\int_0^1 R^2(x)dx = 0 \tag{1.48}$$

and

$$\langle R(x), w_2(x)\rangle = \int_0^1 R\frac{\partial R}{\partial c_3}dx = \frac{1}{2}\frac{\partial}{\partial c_3}\int_0^1 R^2(x)dx = 0. \tag{1.49}$$

Integrating equations (1.48) and (1.49) with respect to x and then differentiating with respect to c_2 and c_3 respectively gives the linear system

$$\begin{aligned} 11418.70c_2 + 1647.01c_3 &= 13933.30 \\ 1647.01c_2 + 717.10c_3 &= 3066.67, \end{aligned} \tag{1.50}$$

which we solve simultaneously to obtain $c_2 = -0.9029$ and $c_3 = -2.2000$. Hence the trial solution by the least squares method is

$$y_l(x) = 1 - 0.9029(1 - x^2) - 2.2000x^2(1 - x^2). \tag{1.51}$$

4. **The Moment Method**

Here the weight functions are $w_1 = 1$ and $w_2(x) = x$. So we will need to evaluate the integrals

$$\langle R(x), w_1(x)\rangle = \int_0^1 R(x)dx = 0 \tag{1.52}$$

and

$$\langle R(x), w_2(x)\rangle = \int_0^1 R(x)x dx = 0. \tag{1.53}$$

Evaluating the integrals in equations (1.52) and (1.53) gives the linear system

$$\begin{aligned} -69.6667c_2 - 15.3333c_3 &= 100 \\ -26.6667c_2 - 10.4667c_3 &= 50, \end{aligned} \tag{1.54}$$

and solving simultaneously yields $c_2 = -0.8742$ and $c_3 = -2.5498$. Hence the trial solution by the moment method is

$$y_m(x) = 1 - 0.8742(1 - x^2) - 2.5498x^2(1 - x^2). \tag{1.55}$$

5. **The Galerkin Method**

The weight functions are chosen as $w_1(x) = (1 - x^2)$ and $w_2(x) = x^2(1 - x^2)$. Evaluating the integrals

$$\langle R(x), w_2(x)\rangle = \int_0^1 R(x)(1 - x^2)dx = 0 \tag{1.56}$$

and

$$\langle R(x), w_2(x)\rangle = \int_0^1 R(x)[x^2(1 - x^2)]dx = 0, \tag{1.57}$$

yields the linear system

$$\begin{aligned} -56.0000c_2 - 7.71905c_3 &= 66.6667 \\ -8.15238c_2 - 2.95873c_3 &= 13.3333. \end{aligned} \tag{1.58}$$

Solving simultaneously yields $c_2 = -0.9179$ and $c_3 = -1.9772$. The trial solution by the Galerkin method is

$$y_g(x) = 1 - 0.9179(1 - x^2) - 1.9772x^2(1 - x^2). \tag{1.59}$$

The plots of the solutions and the errors for the above five methods are given in Figures 1.5 and 1.6 respectively. A table of values resulting from different approximations is shown in Table 1.4.

From Figure 1.6 and Table 1.4, we notice that the error is now reduced but is still large.

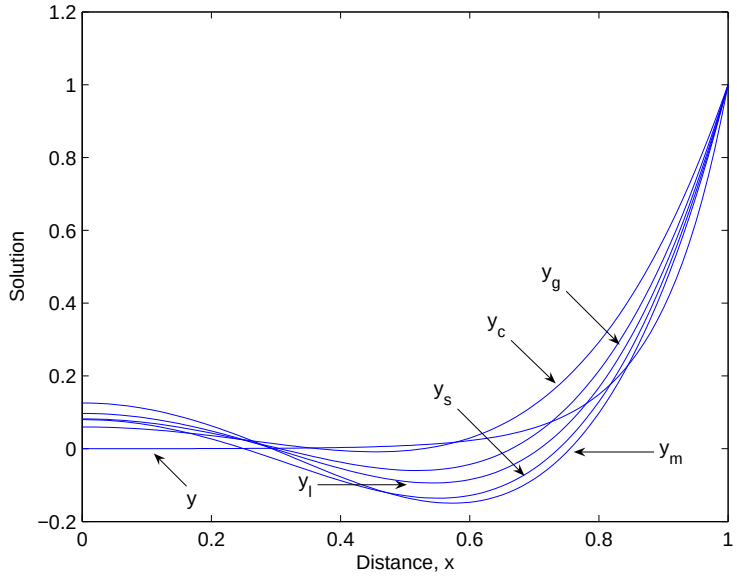

Figure 1.5: Comparison of y and y_a for Example 1.3.3.

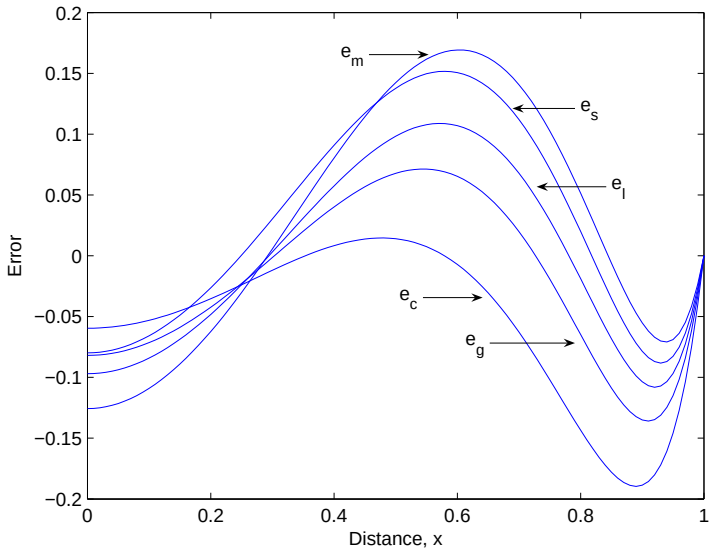

Figure 1.6: Errors $e_a = y - y_a$ for Example 1.3.3.

x	y	y_c	y_s	y_l	y_m	y_g
0	0.0001	0.0597	0.0800	0.0971	0.1258	0.0821
0.0500	0.0002	0.0581	0.0765	0.0939	0.1216	0.0795
0.1000	0.0002	0.0533	0.0660	0.0843	0.1093	0.0717
0.1500	0.0003	0.0456	0.0492	0.0690	0.0894	0.0593
0.2000	0.0005	0.0358	0.0269	0.0487	0.0629	0.0429
0.2500	0.0008	0.0247	0.0003	0.0246	0.0310	0.0236
0.3000	0.0013	0.0132	−0.0290	−0.0018	−0.0044	0.0028
0.3500	0.0021	0.0028	−0.0590	−0.0288	−0.0412	−0.0180
0.4000	0.0033	−0.0050	−0.0875	−0.0541	−0.0770	−0.0368
0.4500	0.0053	−0.0084	−0.1119	−0.0753	−0.1090	−0.0513
0.5000	0.0086	−0.0054	−0.1291	−0.0897	−0.1337	−0.0592
0.5500	0.0138	0.0063	−0.1358	−0.0940	−0.1477	−0.0574
0.6000	0.0223	0.0293	−0.1284	−0.0847	−0.1470	−0.0430
0.6500	0.0358	0.0663	−0.1027	−0.0582	−0.1270	−0.0125
0.7000	0.0576	0.1204	−0.0544	−0.0103	−0.0830	0.0378
0.7500	0.0927	0.1946	0.0212	0.0636	−0.0100	0.1118
0.8000	0.1492	0.2926	0.1292	0.1681	0.0978	0.2140
0.8500	0.2401	0.4181	0.2752	0.3084	0.2462	0.3489
0.9000	0.3863	0.5749	0.4648	0.4899	0.4415	0.5213
0.9500	0.6215	0.7674	0.7042	0.7184	0.6904	0.7365
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
	Total error ⇒	**0.3725**	**0.4022**	**0.3403**	**0.4679**	**0.3060**

Table 1.4: Numerical comparison of the different approximations in Example 1.3.3.

1.3.4 Different Choice of Collocation Points

For the collocation method in Example 1.3.3, we chose equally spaced collocation points on the domain $[0, 1]$ as $x_1 = 1/3$ and $x_2 = 2/3$. If instead we choose unequally spaced collocation points as $x_1 = 1/4 = 0.25$ and $x_2 = 3/4 = 0.75$, then we will obtain $c_2 = -0.9587$ and $c_3 = -1.8513$ and the trial solution becomes

$$y_{\bar{c}}(x) = 1 - 0.9587(1 - x^2) - 1.8513x^2(1 - x^2). \tag{1.60}$$

From Figures 1.7 and 1.8, we notice that with this new choice of collocation points, the approximate solution $y_{\bar{c}}$ by the collocation method gives a better approximation as compared to y_c. Moreover the total error 0.3012 obtained from the former is better than the value 0.3725 obtained from the latter (see Table 1.5). With this choice of collocation points, the collocation method gives the best approximation to this particular problem. This means that the choice of the collocation points is very critical to getting good results for the collocation method [22, 72]. We will expatiate on this in chapter 2. However, note that it does not necessarily mean that unequally spaced collocation points are the best.

Remark 1.3.4 *We have seen that of all the five variations of the method of weighted residuals, the collocation method is the easiest to work with. This is because of the property of the Dirac delta function which makes computation easier. The sub-domain and least squares method are tedious to use. The Galerkin method gives good accuracy to most problems but the collocation method is preferred for most practical problems because of the ease of computation [21, 59].*

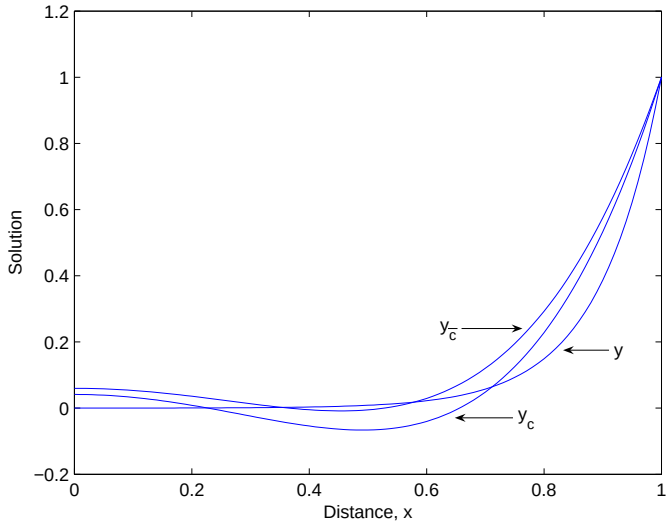

Figure 1.7: Comparison of y, y_c and $y_{\bar{c}}$ for Example 1.3.3.

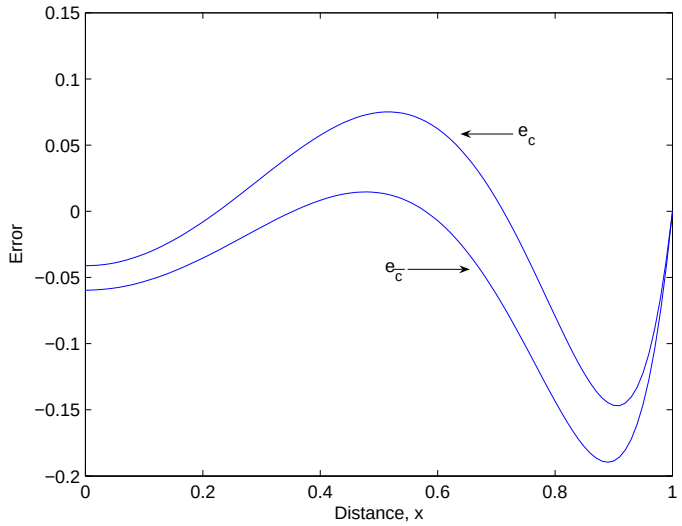

Figure 1.8: Errors $e_c = y - y_c$ and $e_{\bar{c}} = y - y_{\bar{c}}$ for Example 1.3.3.

x	y	y_c	$y_{\bar{c}}$
0	0.0001	0.0597	0.0413
0.0500	0.0002	0.0581	0.0391
0.1000	0.0002	0.0533	0.0326
0.1500	0.0003	0.0456	0.0222
0.2000	0.0005	0.0358	0.0086
0.2500	0.0008	0.0247	−0.0073
0.3000	0.0013	0.0132	−0.0240
0.3500	0.0021	0.0028	−0.0403
0.4000	0.0033	−0.0050	−0.0541
0.4500	0.0053	−0.0084	−0.0635
0.5000	0.0086	−0.0054	−0.0661
0.5500	0.0138	0.0063	−0.0593
0.6000	0.0223	0.0293	−0.0401
0.6500	0.0358	0.0663	−0.0054
0.7000	0.0576	0.1204	0.0484
0.7500	0.0927	0.1946	0.1250
0.8000	0.1492	0.2926	0.2283
0.8500	0.2401	0.4181	0.3628
0.9000	0.3863	0.5749	0.5329
0.9500	0.6215	0.7674	0.7436
1.0000	1.0000	1.0000	1.0000
	Total error ⇒	**0.3725**	**0.3012**

Table 1.5: Numerical comparison of y, y_c and $y_{\bar{c}}$ for Example 1.3.3.

Chapter 2

Orthogonal Collocation Method

2.1 Introduction

We have seen from Example 1.3.2 and 1.3.3 in chapter 1 that the choice of the collocation points is critical and has to be judiciously selected. This clever selection of the collocation points marks the difference between the collocation method and the orthogonal collocation method.

The orthogonal collocation method was first introduced by Villadsen and Stewart in 1967. They discovered that collocation points chosen as the roots of orthogonal polynomials gave good results due to some attractive features of these polynomials [72]. They chose the trial functions as the Jacobi polynomials and picked the collocation points as the corresponding zeros of these polynomials [72]. Thereafter in 1972, Finlayson used it to solve many problems in chemical engineering [22]. Fan, Chen and Erickson used it to solve equations arising from chemical reactors [20]. Finlayson applied it to nonlinear problems in 1980 [21]. In recent times, so many investigators [1, 34, 36, 61] have applied this method to solve a variety of chemical engineering problems.

We now lay some basic foundations.

2.2 Orthogonal Polynomials

Let $[a, b]$ be an interval in the real line $\mathbb{R}$. Let $w(x) : [a, b] \to \mathbb{R}$ be a function such that $w(x) > 0$ on (a, b). Here $w(x)$ is the weight function and it must satisfy the finite integral $\int_a^b \phi_N(x)w(x)dx < \infty$ for any polynomial $\phi_N(x)$ of degree N [36]. Two polynomials $\phi_N(x)$ and $\phi_M(x)$, where $N \neq M$, satisfying the above condition are said to be orthogonal to each other with respect to $w(x)$ on the interval $[a, b]$ if their inner product is zero, that is

$$\langle \phi_N(x), \phi_M(x) \rangle = \int_a^b \phi_N(x)\phi_M(x)w(x)dx = 0. \tag{2.1}$$

Orthogonal polynomials have useful properties that are exploited in the solution of mathematical and physical problems. These features provide a natural way to solve, expand and interpret many important differential equations.

2.2.1 Useful Properties of Orthogonal Polynomials

1. **Recurrence Relation**

   A set $\{\phi_k | k = 1, 2, ..., N\}$ of orthogonal polynomials satisfies the three point recurrence relation

   $$\phi_{k+1}(x) = (a_k x + b_k)\phi_k(x) + c_k\phi_{k-1}(x), \tag{2.2}$$

   where a_k, b_k , c_k are constant coefficients.

2. **Existence of Real Roots**

   Each polynomial in an orthogonal sequence has all of its roots real, distinct, and strictly inside the interval of orthogonality. This property is not common with other polynomials.

3. **Interlacing of Roots**

   The roots of $\phi_k(x)$ strictly separate the roots of $\phi_{k+1}(x)$.

Remark 2.2.1 *Villadsen and Stewart [72], Carey and Finlayson [11], Villadsen and Sorensen [72] and Finlayson [22], have proposed choosing the collocation points as the zeros of Jacobi polynomials. However in this present study, we shall use the zeros of the Chebyshev polynomials of the first and second kind and the Legendre polynomials. We now summarize the properties of these orthogonal polynomials.*

2.2.2 Chebyshev Polynomials of the First Kind $\mathbf{T_N(x)}$

The Chebyshev polynomials of the first kind [60] arise as the solution to the Chebyshev differential equation

$$(1-x^2)\,y''(x) - x\,y'(x) + N^2\,y(x) = 0. \tag{2.3}$$

They are defined by the recurrence relation

$$T_0(x) = 1, \tag{2.4}$$

$$T_1(x) = x, \tag{2.5}$$

$$T_{N+1}(x) = 2xT_N(x) - T_{N-1}(x). \tag{2.6}$$

They can alternatively be defined by the trigonometric identity:

$$T_N(x) = \cos(N\cos^{-1}x), \tag{2.7}$$

where $T_N(\cos(\theta)) = \cos(N\theta)$, for $N = 0, 1, 2, 3, ...$

The Chebyshev polynomials of the first kind are orthogonal with respect to the weight function $w(x) = 1/\sqrt{1-x^2}$ on the interval $[-1, 1]$, that is

$$\int_{-1}^{1} \frac{T_N(x)\,T_M(x)}{\sqrt{1-x^2}}\,dx = \begin{cases} 0 & N \neq M, \\ \pi & N = M = 0, \\ \frac{\pi}{2} & N = M \neq 0. \end{cases} \tag{2.8}$$

One can easily show that the roots of $T_N(x)$ are

$$x_k = \cos\left(\frac{(2k-1)\pi}{2N}\right), \quad k = 1, 2, ..., N. \tag{2.9}$$

2.2.3 Chebyshev Polynomials of the Second Kind $\mathbf{U_N(x)}$

The Chebyshev polynomials of the second kind [39] arise as the solution to the Chebyshev differential equation

$$(1-x^2)\,y''(x) - 3x\,y'(x) + N(N+2)\,y(x) = 0. \tag{2.10}$$

They are defined by the recurrence relation

$$U_0(x) = 1, \tag{2.11}$$

$$U_1(x) = 2x, \tag{2.12}$$

$$U_{N+1}(x) = 2xU_N(x) - U_{N-1}(x). \tag{2.13}$$

They can alternatively be defined by the trigonometric identity:

$$U_N(x) = \frac{\sin((N+1)\theta)}{\sin\theta}, \tag{2.14}$$

with $x = cos\theta$. They are orthogonal with respect to the weight function $w(x) = \sqrt{1-x^2}$ on the interval $[-1, 1]$, that is

$$\int_{-1}^{1} U_N(x)U_M(x)\sqrt{1-x^2}\,dx = \begin{cases} 0 & N \neq M, \\ \frac{\pi}{2} & N = M. \end{cases} \tag{2.15}$$

The roots of $U_N(x)$ are given by

$$x_k = \cos\left(\frac{k\ \pi}{N+1}\right), \quad k = 1, 2, ..., N. \tag{2.16}$$

2.2.4 Legendre Polynomials $\mathbf{P_N(x)}$

The Legendre polynomials [47, 59] are solutions to the Legendre's differential equation:

$$(1-x^2)\ y''(x) - 2x\ y'(x) + N(N+1)\ y(x) = 0. \tag{2.17}$$

They may be determined using Rodrigues formula, that is

$$P_N(x) = \frac{1}{2^N N!}\frac{d^N}{dx^N}\left[(x^2-1)^N\right]. \tag{2.18}$$

They are orthogonal with respect to the weight function $w(x) = 1$ on the interval $[-1, 1]$, that is

$$\int_{-1}^{1} P_N(x)P_M(x)\,dx = \frac{2}{2N+1}\delta_{NM}, \tag{2.19}$$

where δ_{NM} denotes the Kronecker delta defined by

$$\delta_{NM} = \begin{cases} 1 & \text{if } N = M, \\ 0 & \text{otherwise.} \end{cases} \tag{2.20}$$

Unfortunately, there is no formula for the roots of of the Legendre polynomials hence they are usually determined numerically. By differentiating $(x^2-1)^N$, N times with respect to x and then substituting into the Rodrigues formula equation (2.18) we obtain the expression

$$P_N(x) = \frac{1}{2^N}\sum_{k=0}^{\frac{N}{2}} \frac{(-1)^{\frac{N}{2}-k}(N+2k)!x^{2k}}{(\frac{N}{2}+k)!(\frac{N}{2}-k)!(2k)!}\ , \quad \text{for } n \text{ even} \tag{2.21}$$

and

$$P_N(x) = \frac{1}{2^N}\sum_{k=0}^{\frac{N-1}{2}} \frac{(-1)^{\frac{N-1}{2}-k}(N+2k+1)!x^{2k+1}}{(\frac{N+1}{2}+k)!(\frac{N-1}{2}-k)!(2k+1)!}\ , \quad \text{for } n \text{ odd}, \tag{2.22}$$

we can then use the MATLAB built in function **polyroot** to evaluate the roots since the coefficients are known.

Remark 2.2.2 *Using the linear transformation $L : x \to (b-a)/2x + (a+b)/2$, orthogonal polynomials on $[-1, 1]$ can be shifted into any interval $[a, b]$. The roots in $[a, b]$ can then be determined. This is equivalent to shifting the roots in $[-1, 1]$ to $[a, b]$ using the same linear transformation.*

2.3 Lagrange Interpolation

The Lagrange polynomial was first published by Waring in 1779, then rediscovered by Euler in 1783 and published by Lagrange in 1795 [60, 47]. They are useful in

interpolation theory. Polynomials in general are preferred for interpolation because they have derivatives and integrals which are themselves polynomials hence making them easy to work with.

As discussed by Villadsen and Stewart [72], we use the Lagrange interpolation polynomial to effect an approximate solution, that is

$$y_a(x) = \sum_{k=1}^{N+1} c_k l_k(x), \tag{2.23}$$

where N is the order of the polynomial and $l_k(x)$ is the Lagrange polynomial function defined by

$$l_k(x) = \prod_{\substack{j=1 \\ j \neq k}}^{N+1} \frac{x - x_j}{x_k - x_j}. \tag{2.24}$$

where $\{x_j | \, j = 1, 2, ..., N{+}1\}$ are the interpolation nodes. The Lagrange polynomial function has the property that

$$l_k(x_j) = \begin{cases} 1, & j = k, \\ 0, & j \neq k, \end{cases} \tag{2.25}$$

moreover the set $\{l_k(x) | k = 1, 2, ..., N + 1\}$ is linearly independent and forms a basis for the space of the polynomials of degree less than or equal to N on the interval $[a, b]$.

For second order differential equations, we require only the first two derivatives of the Lagrange polynomial which are given in equations (2.26) and (2.27) respectively,

$$\frac{dy_a(x)}{dx} = \sum_{k=1}^{N+1} c_k l_k'(x), \tag{2.26}$$

$$\frac{d^2 y_a(x)}{dx^2} = \sum_{k=1}^{N+1} c_k l_k''(x). \tag{2.27}$$

Using property (2.25), it is clear that $c_k = y_a(x_k)$ and hence the approximate solution at the nodes x_k are just the coefficients in equation (2.23).

The Lagrange function in equation (2.24) can easily be rewritten as

$$l_k(x) = \frac{\psi(x)}{(x - x_k)\psi'(x_k)}, \tag{2.28}$$

where $\psi(x)$ is a $N+1$ degree polynomial, called the nodal polynomial and it is defined by

$$\psi(x) = \prod_{j=1}^{N+1}(x - x_j), \tag{2.29}$$

where $\{x_i |\ j = 1, 2, ..., N+1\}$ are the interpolation points. Equation (2.28) can be rearranged as

$$l_k(x)(x - x_k)\psi'(x_k) = \psi(x). \tag{2.30}$$

We obtain the expression for the first derivative of the Lagrange function evaluated at x_j by differentiating equation (2.30) with respect to x, that is

$$l_k'(x_j) = \frac{\psi'(x_j)}{\psi'(x_k)(x_j - x_k)}, \text{ for } j \neq k. \tag{2.31}$$

Similarly, we obtain the second derivative evaluated at x_j by differentiating equation (2.30) twice with respect to x to yield

$$l_k''(x_j) = \frac{\psi''(x_j)}{(x_j - x_k)\psi'(x_k)} - 2l_k'(x_j). \tag{2.32}$$

We can also obtain an expression for the first derivative of the Lagrange function evaluated at x_k by differentiating the Lagrange function in equation (2.24) with respect to x, that is

$$l_k'(x_k) = \sum_{\substack{j=1 \\ j\neq k}}^{N+1} \frac{1}{(x_k - x_j)}. \tag{2.33}$$

Differentiating equation (2.24) twice with respect to x and evaluating at x_k yields

$$l_k''(x_k) = \sum_{\substack{i,j=1 \\ i,j\neq k \\ i<j}}^{N+1} \frac{1}{(x_k - x_i)(x_k - x_j)}. \tag{2.34}$$

2.3.1 Chebyshev Interpolation Nodes

When $f(t) \in C[-1,1]$ is approximated by the Lagrange interpolation polynomial $P_N(t)$ of degree at most N, the error at the point $x \in [-1,1]$ is given by

$$E_N(x) = f(x) - P_N(x) = \frac{f^{N+1}(\xi(x))}{(N+1)!}\psi(x), \tag{2.35}$$

where

$$\psi(t) = (t-x_1)(t-x_2)\cdots(t-x_{N+1}), \tag{2.36}$$

is a monic polynomial and $\xi(x) \in (-1,1)$ is usually unknown.

From equation (2.35),

$$|E_N(x)| = \frac{|f^{N+1}(\xi(x))|}{(N+1)!}|\psi(x)| \leq \frac{\|f^{N+1}(t)\|_\infty}{(N+1)!}\|\psi(t)\|_\infty = M\|\psi(t)\|_\infty, \tag{2.37}$$

where $\|\psi(t)\|_\infty = \underset{t\in[-1,1]}{Max}|\psi(t)|$ and $M = \frac{\|f^{N+1}(t)\|_\infty}{(N+1)!} = \underset{t\in[-1,1]}{Max}\frac{|f^{N+1}(t)|}{(N+1)!}$.

Let $q(t) \in C[-1,1]$ be a monic polynomial of degree $N+1$ such that $\|q(t)\|_\infty \leq \|\psi(t)\|_\infty$, then it is shown in [60] that $q(t) = \frac{T_{N+1}(t)}{2^N}$ and $\|q(t)\|_\infty = \frac{1}{2^N}$. Hence in order to minimize the interpolation error we require that

$$\frac{T_{N+1}(t)}{2^N} = (t-x_1)(t-x_2)\cdots(t-x_{N+1}). \tag{2.38}$$

From this we clearly see that the optimal choice of the interpolation nodes $\{x_k | k = 1,2,...,N+1\}$ are the roots of the Chebyshev polynomial $T_{N+1}(t)$ of degree $N+1$ which are easily obtained from equation (2.9), [40].

2.4 Relationship Between Galerkin and Collocation Method

From Examples 1.3.1 and 1.3.2 in chapter 1, we have seen that the Galerkin method is the best of all the five weighted residual methods to provide a good accuracy which is in agreement with the work done by Finlayson [22]. However, we noticed

from chapter 1 that the Galerkin method can be very expensive especially when we retain more terms in the trial solution. For the Galerkin method we require that $\langle R(x), \phi_j(x)\rangle = 0$, for $j = 1, 2, ..., N$, where $R(x) = R(x, a_1, a_2, ..., a_N)$ is the residual and $\{\phi_j(x)|\ j = 1, 2, ..., N\}$ are the basis functions which in this discussion will be assumed to also be an orthogonal set that automatically satisfies the boundary conditions. We approximate $\langle R(x), \phi_j(x)\rangle$ numerically by an N point quadrature formula

$$\langle R(x), \phi_j(x)\rangle = \int_X R(x)\phi_j(x)dx \approx \sum_{k=1}^{N} w_k R(x_k)\phi_j(x_k), \tag{2.39}$$

where w_k are the quadrature weights and x_k are the quadrature points in X. If $\{x_k|k = 1, 2, ..., N\}$ are the zeros of $\phi_N(x)$ then $\langle R(x), \phi_N(x)\rangle \approx 0$. Since $\phi_j(x_k) \neq 0$ for $j = 1, 2, ..., N-1$ and $k = 1, 2, ..., N$ (recall property 3, section 2.2.1), we force $\langle R(x), \phi_j(x)\rangle \approx 0$ for $j = 1, 2, ..., N-1$ by requiring that $R(x_k) = 0$, for $k = 1, 2, ..., N$ which is simply the collocation method.

Thus if the collocation method is used with the collocation points chosen as roots of the orthogonal polynomial $\phi_N(x)$, then the collocation method will closely approximate the Galerkin method. If $\phi_N(x)$ were not an orthogonal polynomial then its roots could be complex and also may lie outside the domain X. The collocation method is easy to apply and to program and its accuracy can be comparable to the Galerkin method if the collocation points are judiciously chosen [21, 22].

Remark 2.4.1 *We are now ready to illustrate the orthogonal collocation method with an example. Note that our choice of collocation points* $x_1 = 0.25$ *and* $x_2 = 0.75$ *in chapter* 1 *(See section* 1.3.4*) gave good results because we used the shifted roots of the Chebyshev polynomial of the second kind which falls under the family of orthogonal polynomials.*

2.5 Numerical Examples

Example 2.5.1 *Consider the differential equation (1.19) which we solved in chapter 1 using the five different methods of weighted residuals. Let us solve the same equation using the orthogonal collocation method.*

2.5.1 One Point Collocation for Example 2.5.1

The one point collocation method is normally used to quickly investigate the behavior of the solution to any particular differential equation as a function of the parameters [68]. In this particular example, we can use it to investigate the behavior of the solution as we increase the reaction rate α.

Here we assume a quadratic approximate solution:

$$y_a(x) = \sum_{k=1}^{3} c_k l_k(x). \tag{2.40}$$

We pick the first and last interpolation points to coincide with the left and right boundary points respectively, that is $x_1 = 0$ and $x_3 = 1$ and pick the third interpolation point as the shifted zero of the Chebyshev polynomial T_1, that is $x_2 = 0.5$. The collocation point x_2^c is chosen as the shifted root of T_1 and hence coincides with the internal interpolation point x_2. Hence we have three second order Lagrange polynomials given by

$$l_1(x) = 2x^2 - 3x + 1, \tag{2.41}$$

$$l_2(x) = 4x - 4x^2, \tag{2.42}$$

$$l_3(x) = 2x^2 - x. \tag{2.43}$$

Substituting the approximate solution (2.40) into the differential equation (1.19) gives the residual equation

$$\begin{aligned} R(x) = (-2\alpha^2x^2 + 3\alpha^2x + 4x - \alpha^2 + 1)c_1 + (4\alpha^2x^2 - 4\alpha^2x - 8x - 4)c_2 \\ + (-2\alpha^2x^2 + \alpha^2x + 4x + 3)c_3. \end{aligned} \tag{2.44}$$

Substituting the collocation point $x_2^c = 0.5$ into the residual equation (2.44) gives

$$R(x_2^c) = 3c_1 - (8 + \alpha^2)c_2 + 5c_3 = 0. \tag{2.45}$$

Satisfying the boundary conditions yield

$$-3c_1 + 4c_2 - c_3 = 0 \tag{2.46}$$

and

$$c_3 = 1, \tag{2.47}$$

where we have employed property (2.25) of the Lagrange polynomial in evaluating c_3. Hence the approximate solution for the one point collocation method is given by

$$y_a(x) = c_1 l_1(x) + c_2 l_2(x) + l_3(x), \tag{2.48}$$

where $c_1 = \frac{12-\alpha^2}{3(4+\alpha^2)}$ and $c_2 = \frac{4}{4+\alpha^2}$. The plots of the solutions and the error are shown in Figures 2.1 and 2.2 for $\alpha = 1$ and Figures 2.3 and 2.4 for $\alpha = 10$ respectively.

Hence forth, we will sample the total error with 21 mesh points. Tables 2.1 and 2.2 give the numerical comparison for different values of x.

From Tables 2.1 and 2.2, we observe that the total error is high especially for $\alpha = 10$. From Figure 2.4 we notice that the error is unacceptably high. The solution which represents the concentration of the chemical becomes negative for $\alpha = 10$ and since this cannot happen in reality, we infer that more collocation points are needed to produce a higher degree approximation.

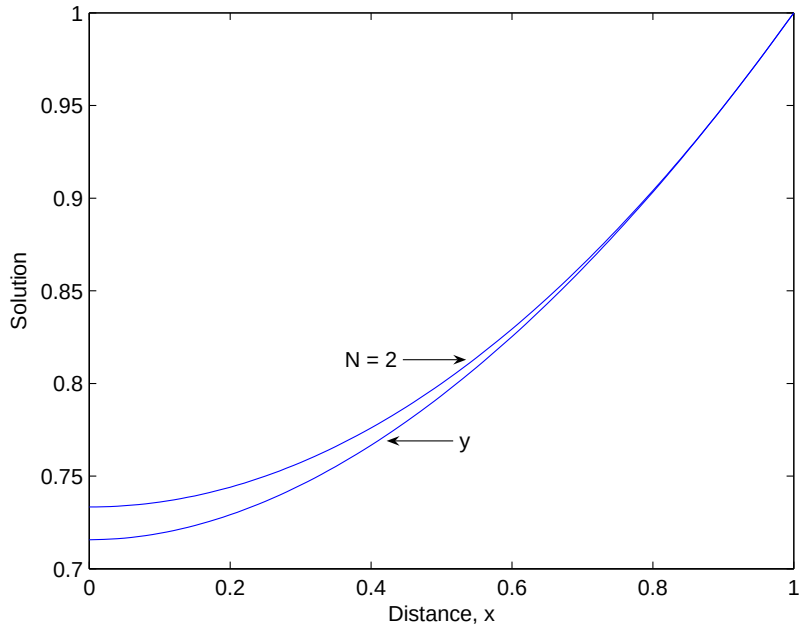

Figure 2.1: Comparison of y and y_a for Example 2.5.1 with $\alpha = 1$ and order $N = 2$.

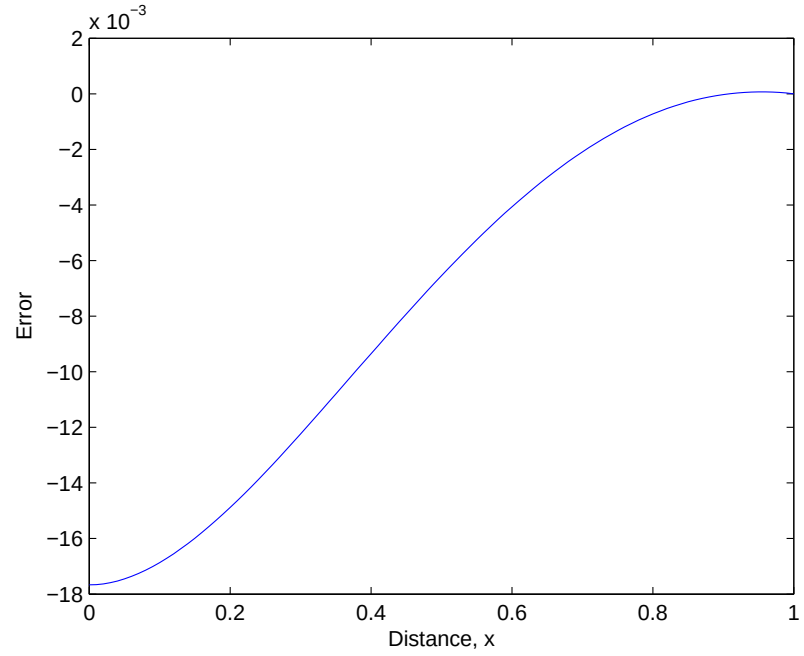

Figure 2.2: Error between y and y_a for Example 2.5.1 with $\alpha = 1$ and order $N = 2$.

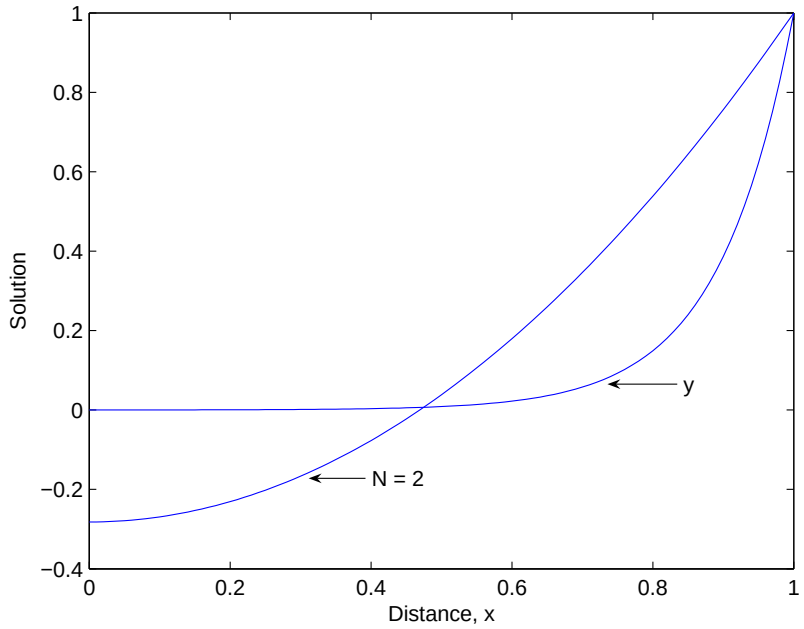

Figure 2.3: Comparison of y and y_a for Example 2.5.1 with $\alpha = 10$ and order $N = 2$.

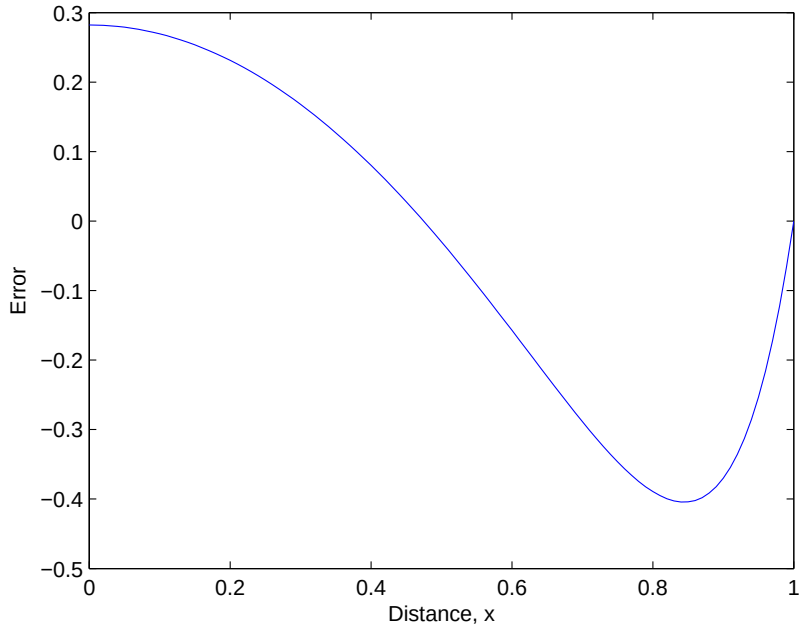

Figure 2.4: Error between y and y_a for Example 2.5.1 with $\alpha = 10$ and order $N = 2$.

x	y	y_a
0	0.7157	0.7333
0.0500	0.7165	0.7340
0.1000	0.7191	0.7360
0.1500	0.7233	0.7393
0.2000	0.7291	0.7440
0.2500	0.7364	0.7500
0.3000	0.7451	0.7573
0.3500	0.7552	0.7660
0.4000	0.7666	0.7760
0.4500	0.7794	0.7873
0.5000	0.7935	0.8000
0.5500	0.8087	0.8140
0.6000	0.8253	0.8293
0.6500	0.8430	0.8460
0.7000	0.8619	0.8640
0.7500	0.8820	0.8833
0.8000	0.9033	0.9040
0.8500	0.9257	0.9260
0.9000	0.9493	0.9493
0.9500	0.9741	0.9740
1.0000	1.0000	1.0000
	Total Error $\Rightarrow$	**0.0456**

Table 2.1: Numerical comparison for one point collocation for Example 2.5.1 with $\alpha = 1$ and order $N = 2$.

x	y	y_a
0	0.0001	−0.2821
0.0500	0.0002	−0.2788
0.1000	0.0002	−0.2692
0.1500	0.0003	−0.2532
0.2000	0.0005	−0.2308
0.2500	0.0008	−0.2019
0.3000	0.0013	−0.1667
0.3500	0.0021	−0.1250
0.4000	0.0033	−0.0769
0.4500	0.0053	−0.0224
0.5000	0.0086	0.0385
0.5500	0.0138	0.1058
0.6000	0.0223	0.1795
0.6500	0.0358	0.2596
0.7000	0.0576	0.3462
0.7500	0.0927	0.4391
0.8000	0.1492	0.5385
0.8500	0.2401	0.6442
0.9000	0.3863	0.7564
0.9500	0.6215	0.8750
1.0000	1.0000	1.0000
	Total Error ⇒	**1.1154**

Table 2.2: Numerical comparison for one point collocation for Example 2.5.1 with $\alpha = 10$ and order $N = 2$.

2.5.2 Generalization of Example 2.5.1

We can improve the accuracy of our approximation by increasing the number of collocation points but as the number of collocation points increases, the number of equations and unknowns also increases, thus making computation by hand very tedious. Thus we will first generalize the problem and reduce it to a linear system. We can then use any programming language with a high speed computer to solve for the unknowns.

In general, we substitute the approximate solution of the form (2.40) into the differential equation (1.19) to obtain

$$\sum_{k=1}^{N+1} \left[c_k l_k''(x) + c_k l_k'(x) - \alpha^2 c_k l_k(x) \right] = 0. \tag{2.49}$$

We first satisfy the left hand boundary condition at x_1, then evaluate the residual (2.49) at the collocation points x_j^c, $j = 2, 3, ..., N$, and finally satisfy the right hand boundary condition at x_{N+1}. The resulting linear system of equations can be cast in the matrix vector form $\mathbf{Ac} = \mathbf{b}$, where $\mathbf{A} = a_{ij}$ is a $(N+1) \times (N+1)$ matrix with

$$a_{1j} = l_j'(x_1), \tag{2.50}$$

$$a_{i,j} = l_j''(x_i^c) + l_j'(x_i^c) - \alpha^2 l_j(x_i^c), \tag{2.51}$$

$$a_{N+1,j} = l_j(x_{N+1}), \tag{2.52}$$

for $j = 1, 2, ..., N+1$, $i = 2, 3, ..., N$, $\mathbf{b}$ is a column vector of dimension $N+1$ with

$$b_j = 0; \quad j = 1, 2, ..., N, \tag{2.53}$$

$$b_{N+1} = 1 \tag{2.54}$$

and

$$\mathbf{c} = c_j; \quad j = 1, 2, ..., N+1, \tag{2.55}$$

is the vector of the unknowns. Hence the solution vector is $\mathbf{c} = \mathbf{A}^{-1}\mathbf{b}$. This system has been coded in MATLAB and the code is given in Appendix A.

We first solve the problem by choosing the collocation points x_j^c, $j = 2, 3, ..., N$ as the shifted roots of the Chebyshev polynomial of the first kind, T_{N-1}. In this instance we emphasize that the collocation points agree with the internal interpolation points, that is $x_j^c = x_j$ for $j = 2, 3, ..., N$. The plots of the solutions and the errors for orders $N = 3, 4$ and 5 are given in Figures 2.5 and 2.6 respectively. The case $N = 16$ is displayed in Figure 2.7.

In Table 2.3 we summarize the total error for different orders N. We notice the gradual decrease in the total error as we increase the order of the polynomial. The total error for orders $N \geq 9$ are small. If we use polynomials of orders $N = 16$ and above, the error levels off at $3.1380e - 006$.

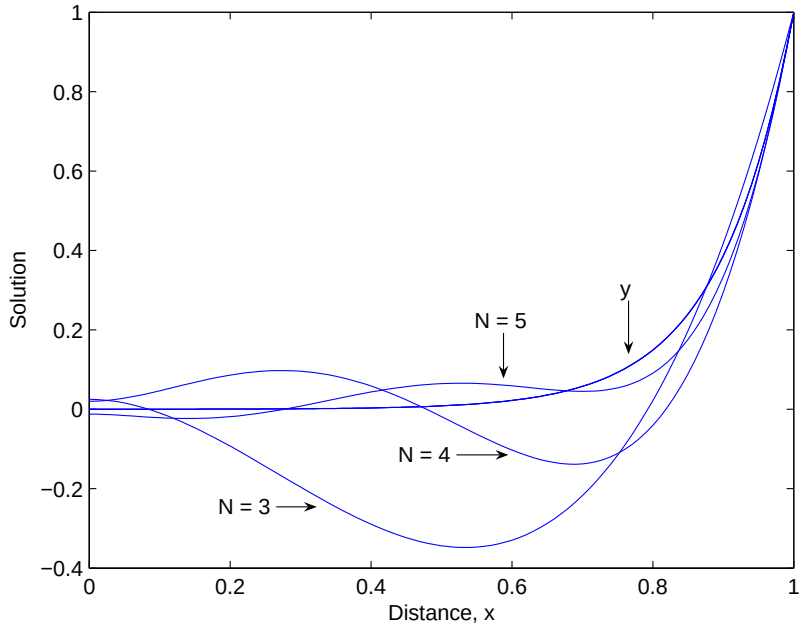

Figure 2.5: Comparison of y and y_a for Example 2.5.1 for orders $N = 3, 4, 5$, with collocation points chosen as the shifted roots of T_{N-1}.

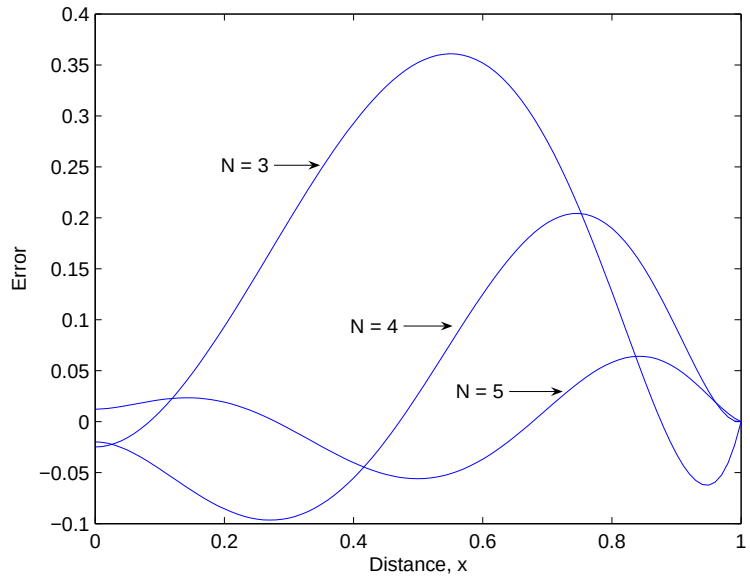

Figure 2.6: Errors for Example 2.5.1 for orders $N = 3, 4, 5$, with collocation points chosen as the shifted roots of T_{N-1}.

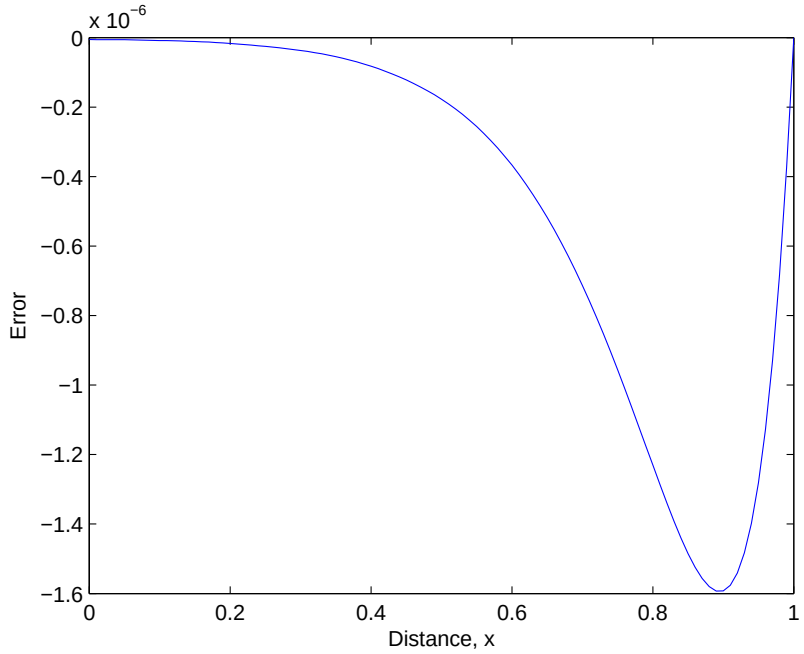

Figure 2.7: Error for order $N = 16$ for Example 2.5.1, with collocation points chosen as the shifted roots of T_{15}.

Secondly we choose the collocation points as the shifted roots of the Chebyshev polynomial of the second kind, U_{N-1}. The plots of the solutions and the errors for orders $N = 3, 4$ and 5 are given in Figures 2.8 and 2.9 respectively.

Thirdly we choose the collocation points as the shifted roots of the Legendre polynomial, P_{N-1}. The plots of the solutions and the errors for orders $N = 3, 4$ and 5 are given in Figures 2.10 and 2.11 respectively.

Comparing the total error from Table 2.3, we observe that the total error is least for U_{N-1}. However, in all cases due to truncation error, the total error levels off at order $N = 16$ with value $3.1380e - 006$.

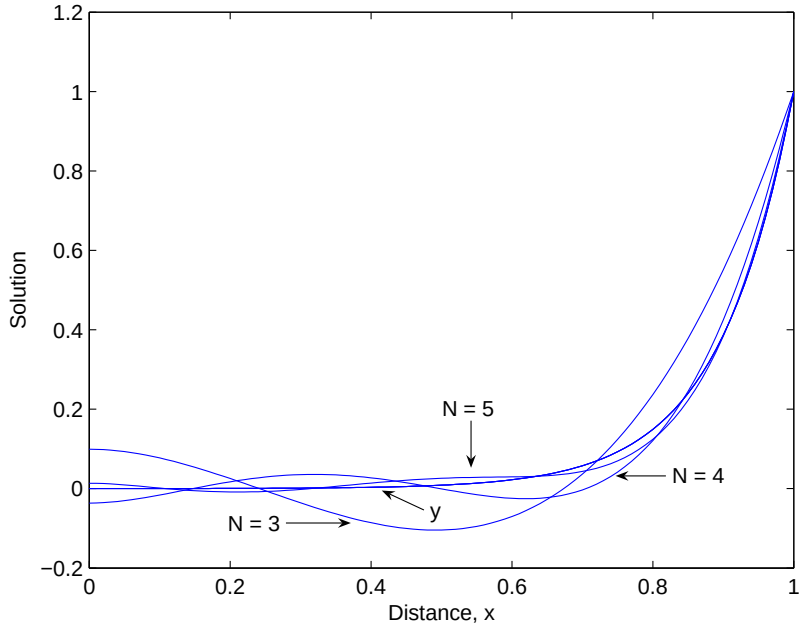

Figure 2.8: Comparison of y and y_a for Example 2.5.1 for orders $N = 3, 4, 5$, with collocation points chosen as the shifted roots of U_{N-1}.

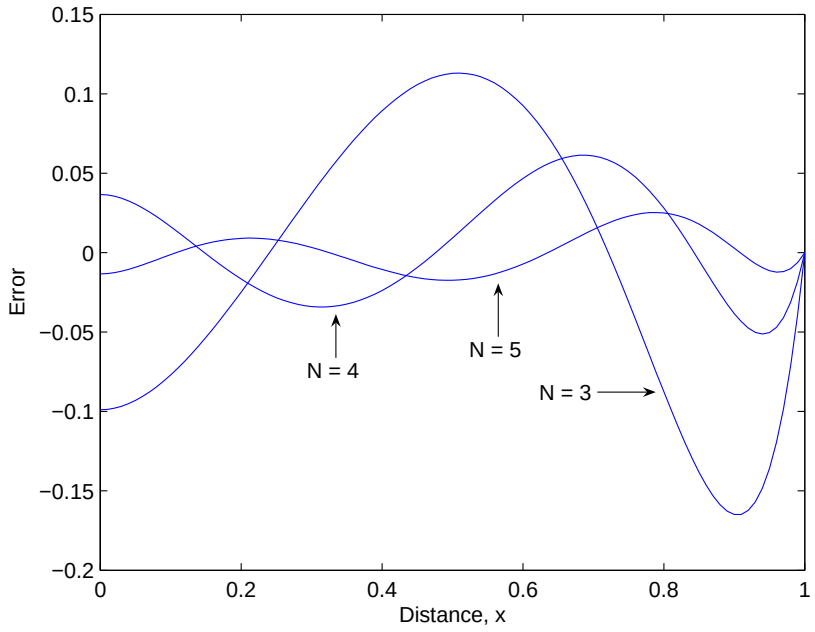

Figure 2.9: Errors for Example 2.5.1 for orders $N = 3, 4, 5$, with collocation points chosen as the shifted roots of U_{N-1}.

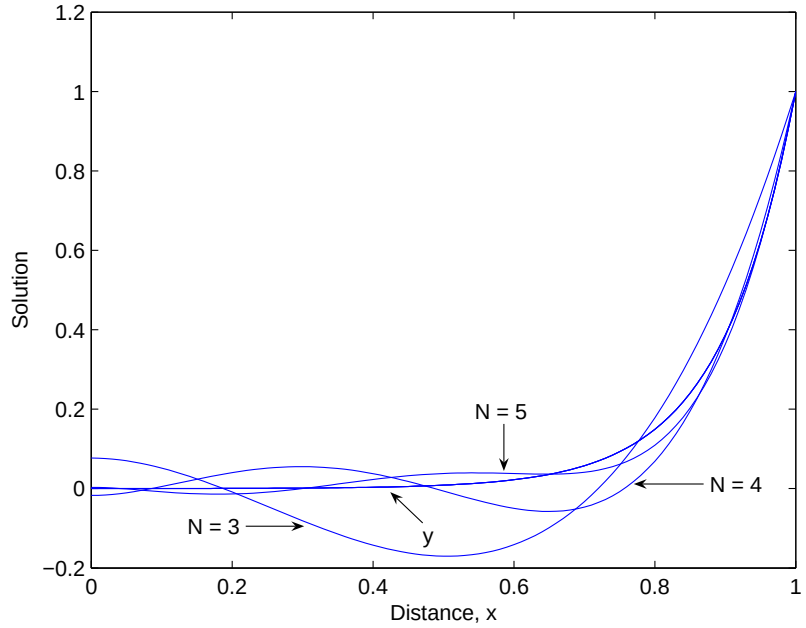

Figure 2.10: Comparison of y and y_a for Example 2.5.1 for orders $N = 3, 4, 5$, with collocation points chosen as the shifted roots of P_{N-1}.

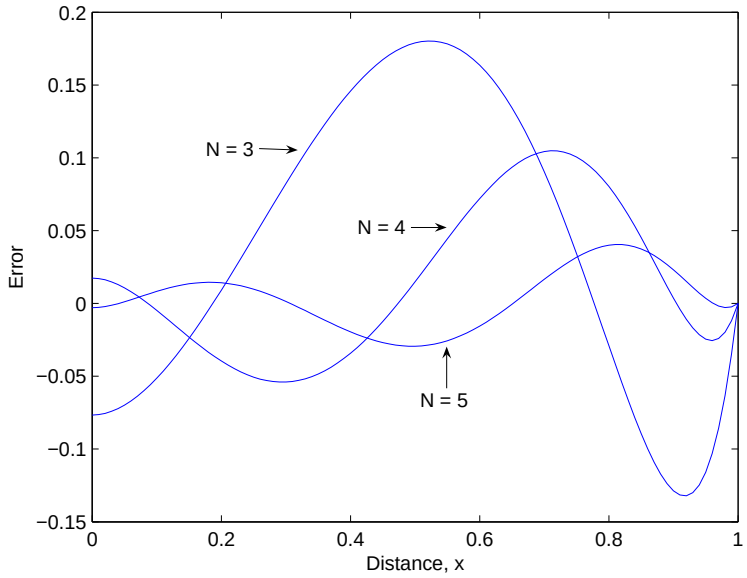

Figure 2.11: Errors for Example 2.5.1 for orders $N = 3, 4, 5$, with collocation points chosen as the shifted roots of P_{N-1}.

Order N	$\mathbf{T_{N-1}}$	$\mathbf{U_{N-1}}$	$\mathbf{P_{N-1}}$
2	1.1154	1.1154	1.1154
3	0.9755	0.4047	0.4925
4	0.4908	0.1554	0.2406
5	0.1631	0.0568	0.0913
6	0.0460	0.0181	0.0277
7	0.0120	0.0051	0.0073
9	$6.1817e-004$	$2.9281e-004$	$3.9050e-004$
11	$2.3159e-005$	$1.2620e-005$	$1.5287e-005$
13	$3.2087e-006$	$3.1525e-006$	$3.1684e-006$
15	$3.1382e-006$	$3.1379e-006$	$3.1381e-006$
16	$3.1380e-006$	$3.1380e-006$	$3.1380e-006$

Table 2.3: Numerical comparison of the errors for Example 2.5.1 for different orders N, with collocation points chosen as the shifted roots of T_{N-1}, U_{N-1} and P_{N-1} respectively.

Example 2.5.2 *In order to investigate how well the collocation method can track a hump, we solve the following non homogeneous boundary value problem on the interval* $[0, 3.5]$*:*

$$\begin{aligned} \frac{d^2y}{dx} + 6\frac{dy}{dx} + 9y &= e^{-3x}, \\ y(0) &= 0, \\ y(3.5) &= 9.625e^{-10.5}. \end{aligned} \tag{2.56}$$

Note that in the previous example, we shifted the roots of the orthogonal polynomials from $[-1, 1]$ to the domain of the problem namely $[0, 1]$. Here we choose to transform the domain $[0, 3.5]$ of the problem to the interval $[-1, 1]$ by using the linear transformation $L : x \to 2x/h - 1$, where $h = 3.5$ is the length of the domain $[0, 0.35]$. With this transformation, the differential equation (2.56) together with its boundary conditions becomes

$$\begin{aligned} \frac{4}{h^2}\frac{d^2y}{dx^2} + \frac{12}{h}\frac{dy}{dx} + 9y &= e^{-1.5h(x+1)}, \\ y(-1) &= 0, \\ y(1) &= 9.625e^{-10.5}. \end{aligned} \tag{2.57}$$

The exact solution to equation (2.56) is given by

$$y(x) = xe^{-3x} + 0.5x^2e^{-3x}. \tag{2.58}$$

The exact solution in equation (2.58) has a maximum value at $x \approx 0.3874$ and damps off quickly as illustrated in Figure 2.12.

2.5.3 Two Points Collocation for Example 2.5.2

Here we assume a cubic approximate solution:

$$\sum_{k=1}^{4} c_k l_k(x). \tag{2.59}$$

We pick the first and last interpolation points to coincide with the left and right boundary points respectively, that is $x_1 = -1$ and $x_4 = 1$ and pick the remaining interpolation points as the zeros of the Chebyshev polynomial T_2, that is $x_2 = -1/\sqrt{2}$ and $x_3 = 1/\sqrt{2}$. The two collocation points x_2^c and x_3^c are chosen as the roots of T_2 and hence coincides with the internal interpolation points x_2 and x_3 respectively. Hence we have four cubic Lagrange polynomials given by

$$\begin{aligned} l_1(x) &= -x^3 + x^2 + 0.5000x - 0.5000, && (2.60)\\ l_2(x) &= 1.4142x^3 - x^2 - 1.4142x + 1.0000, && (2.61)\\ l_3(x) &= -1.4142x^3 - x^2 + 1.4142x + 1.0000, && (2.62)\\ l_4(x) &= x^3 + x^2 - 0.5000x - 0.5000. && (2.63) \end{aligned}$$

Substituting the approximate solution (2.59) into the differential equation (2.57) and forming the residual gives

$$\begin{aligned} R(x) = &\left(-9.0000x^3 - 1.2857x^2 + 9.3977x - 2.1326\right) c_1 \\ &+ \left(12.7278x^3 + 5.5462x^2 - 16.8141x + 3.4981\right) c_2 \\ &+ \left(-12.7278x^3 - 23.5459x^2 + 3.10011x + 13.1955\right) c_3 \\ &+ \left(9.0000x^3 + 19.2853x^2 + 4.3163x - 5.5610\right) c_4 = e^{-1.5h(x+1)}. \end{aligned} \quad (2.64)$$

Upon substituting the collocation points $x_2^c = -1/\sqrt{2}$ and $x_3^c = 1/\sqrt{2}$ into the residual (2.64) and requiring that $R(x_2^c) = R(x_3^c) = 0$, we obtain the linear system

$$\begin{aligned} R(x_2^c) &= -6.2387c_1 + 13.6606c_2 + 3.7305c_3 - 2.1524c_4 = 0.2149 \\ R(x_3^c) &= 0.6879c_1 - 1.1183c_2 - 0.8850c_3 + 10.3154c_4 = 0.0001. \end{aligned} \quad (2.65)$$

We also satisfy the boundary conditions obtaining

$$y_a(-1) = c_1 = 0, \quad (2.66)$$

$$y_a(1) = c_4 = 9.625e^{-10.5}. \quad (2.67)$$

Hence $c_1 = 0$ and $c_4 \approx 0.0003$. Solving the system of equations (2.65) simultaneously yields $c_2 = 0.0228$ and $c_3 = -0.0259$. Hence the approximate solution is

$$y_a(x) = 0.0228l_2(x) - 0.0259l_3(x) + 0.0003l_4(x). \quad (2.68)$$

The plots of the solutions and the error are shown in Figures 2.12 and 2.13 respectively. Table 2.4 gives the numerical comparison for different values of x.

Notice from Table 2.4 that the numerical solution for different values of x does not closely agree with the analytic solution. Moreover the error is high (see Figure 2.13), hence we require more collocation points to obtain a better approximation.

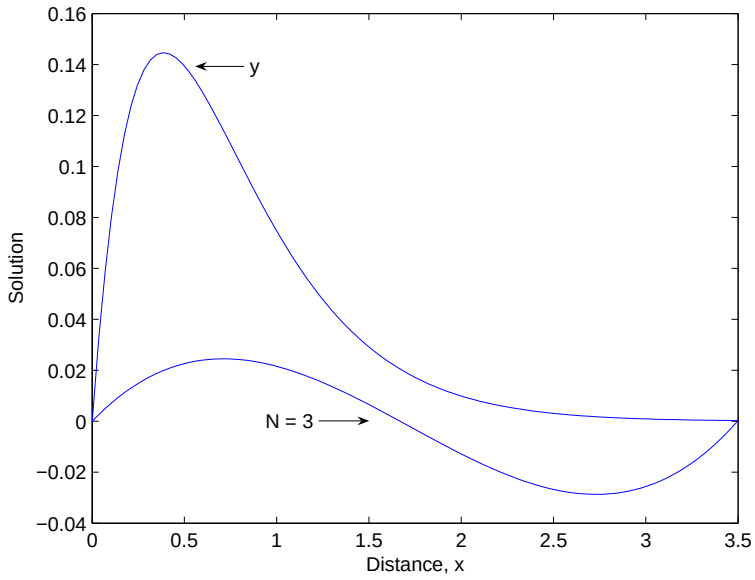

Figure 2.12: Comparison of y and y_a for Example 2.5.2 for order $N = 3$, with collocation points chosen as roots of T_2.

2.5.4 Generalization of Example 2.5.2

In order to improve the accuracy of our result, we generalize to higher orders as done in Example 2.5.1. In general, when we substitute the approximate solution of the form (2.59) into the differential equation (2.57), we obtain

$$\sum_{k=1}^{N+1}\left[\frac{4}{h^2}\ c_k l_k''(x)+\frac{12}{h}\ c_j l_k'(x)+9\ c_k l_k(x)\right]=e^{-1.5h(x+1)}. \tag{2.69}$$

x	y	y_a
0	0	0.000
0.1750	0.1126	0.0112
0.3500	0.1439	0.0188
0.5250	0.1372	0.0231
0.7000	0.1157	0.0245
0.8750	0.0911	0.0235
1.0500	0.0686	0.0205
1.2250	0.0501	0.0160
1.4000	0.0357	0.0102
1.5750	0.0250	0.0037
1.7500	0.0172	−0.0032
1.9250	0.0117	−0.0100
2.1000	0.0079	−0.0163
2.2750	0.0053	−0.0218
2.4500	0.0025	−0.0259
2.6250	0.0023	−0.0283
2.8000	0.00015	−0.0285
2.9750	0.00010	−0.0262
3.1500	0.0006	−0.0209
3.3250	0.0004	−0.0122
3.5000	0.0001	0.0003
	Total error ⇒	**0.7472**

Table 2.4: Numerical comparison for Example 2.5.2 for order $N = 3$, with collocation points chosen as roots of T_2.

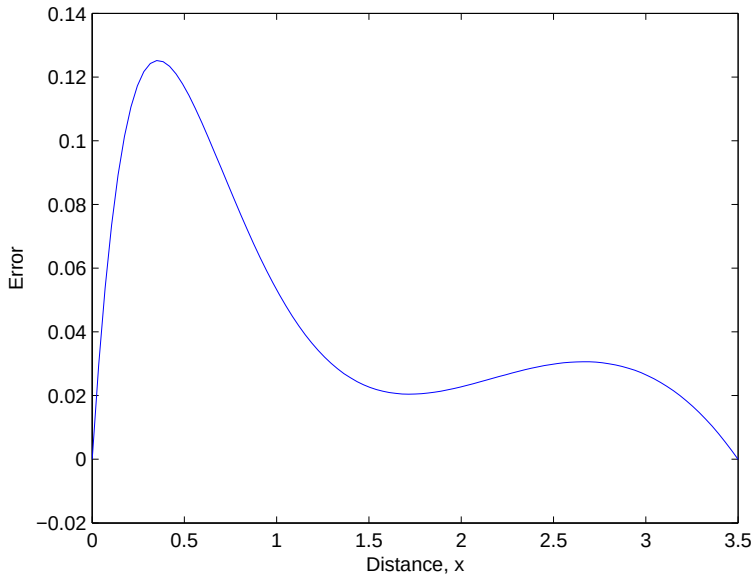

Figure 2.13: Error for Example 2.5.2 for order $N = 3$, with collocation points chosen as roots T_2.

Hence we have the matrix vector form $\mathbf{Ac} = \mathbf{b}$, where $\mathbf{A} = a_{ij}$ is a $(N+1)\times(N+1)$ matrix with

$$a_{1j} = l_j(x_1), \tag{2.70}$$

$$a_{i,j} = \frac{4}{h^2}\, l_j''(x_i^c) + \frac{12}{h}\, l_j'(x_i^c) + 9 l_j(x_i^c), \tag{2.71}$$

$$a_{N+1,j} = l_j(x_{N+1}), \tag{2.72}$$

for $j = 1, 2, ..., N+1$, $i = 2, 3, ..., N$, $\mathbf{b}$ is a column vector of dimension $N+1$ with

$$b_1 = 0, \tag{2.73}$$

$$b_j = e^{-1.5h(x_j^c+1)}; \quad j = 2, ..., N, \tag{2.74}$$

$$b_{N+1} = 9.625e^{-10.5}, \tag{2.75}$$

and

$$\mathbf{c} = c_j; \quad j = 1, 2, ..., N+1. \tag{2.76}$$

A MATLAB code to this system is given in Appendix B.

The plots of the solutions and the errors for orders $N = 4, 6, 8, 10$ are given in

Figures 2.14 and 2.15 respectively. In Figure 2.16 we present the plot for $N = 20$. Table 2.5 summarizes the total errors for different orders.

From Figure 2.14, we observe that as we increase the order of the polynomial, the approximate solution gradually converges to the exact solution and the error is simultaneously lessened as shown in Figure 2.15 and Table 2.5. However notice from Figures 2.14 and 2.15 that the solution obtained from order $N = 10$ is not a very good approximation. Due to truncation error, the total error levels off at order $N = 20$ with value $1.4216e - 005$. This is one limitation of higher order polynomials and we will have to consider other means if we want to obtain more accurate results.

Secondly if we choose the roots of the Chebyshev polynomial of the second kind, U_{N-1} as the collocation points, then we obtain the plots of the solutions and the corresponding errors for orders $N = 4, 6, 8, 10$ as shown in Figures 2.17 and 2.18 respectively. The error levels off at order $N = 21$ with value $1.4216e - 005$.

Thirdly, we choose the roots of the Legendre polynomials P_{N-1} as the collocation points. The plots of the solution and errors are presented in Figures 2.19 and 2.20 respectively.

Notice from Table 2.5 that the Legendre polynomial gives better results for higher orders (orders 10 and above). The error levels off at order $N = 17$ while the error from the Chebyshev polynomials of the first and second kind levels off at order $N = 20$ with value $1.4216e - 005$. From the error plots in Figures 2.15, 2.18 and 2.20 , we notice that the approximate solution is far better in the region $x \geq 2$ where the solution flattens as compared to the region near the hump or peak.

Remark 2.5.3 *Notice from Examples* 2.5.1 *and* 2.5.2 *that higher order polynomials are limited in their accuracy since the total error levels off for higher orders. Thus it is advisable to devise other means to obtain better results.*

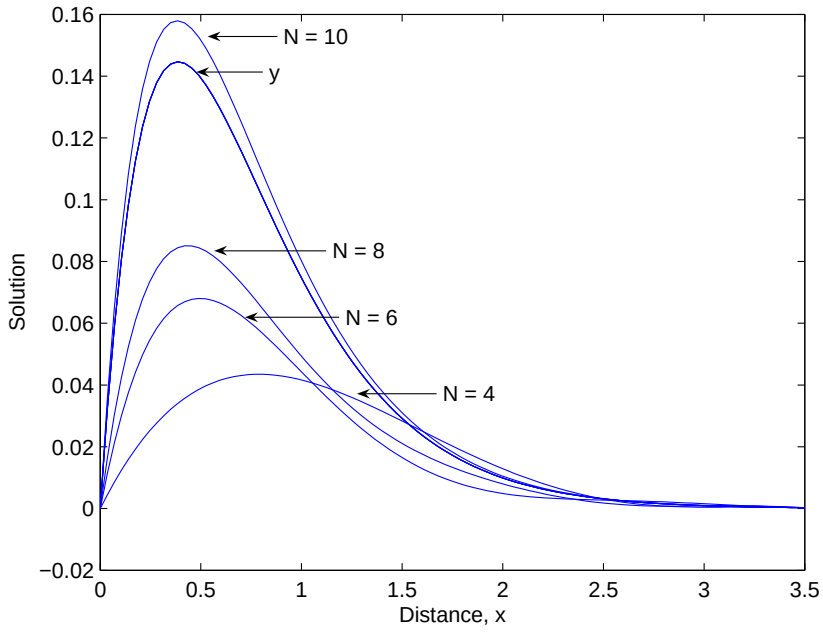

Figure 2.14: Comparison of y and y_a for Example 2.5.2 for orders $N = 4, 6, 8, 10$, with collocation points chosen as the roots of T_{N-1}.

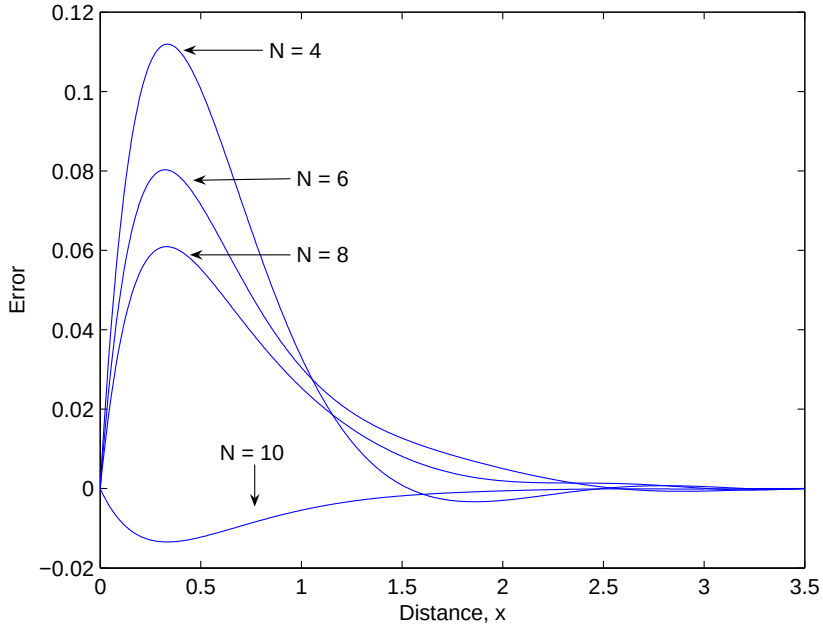

Figure 2.15: Errors for Example 2.5.2 for orders $N = 4, 6, 8, 10$, with collocation points chosen as the roots of T_{N-1}.

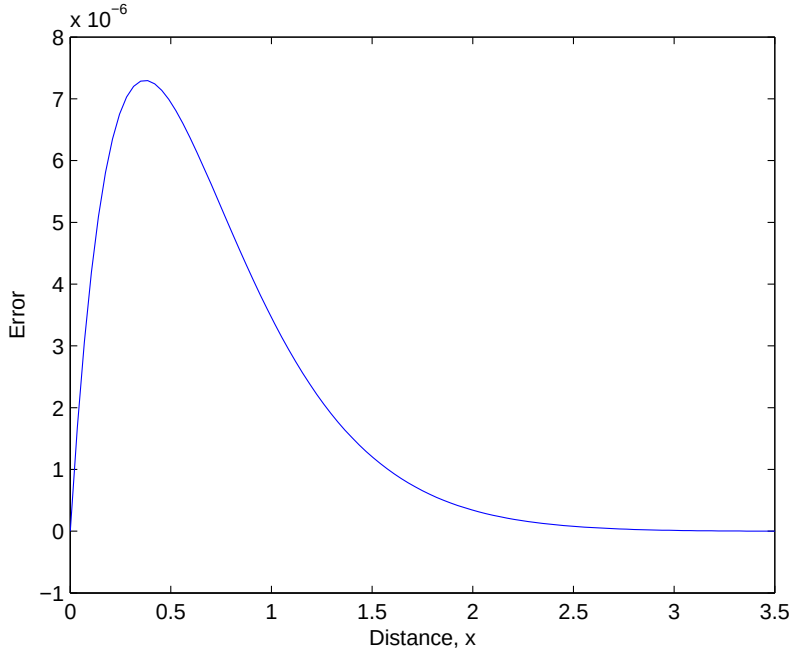

Figure 2.16: Error for order $N = 20$ for Example 2.5.2, with collocation points chosen as the roots of T_{19}.

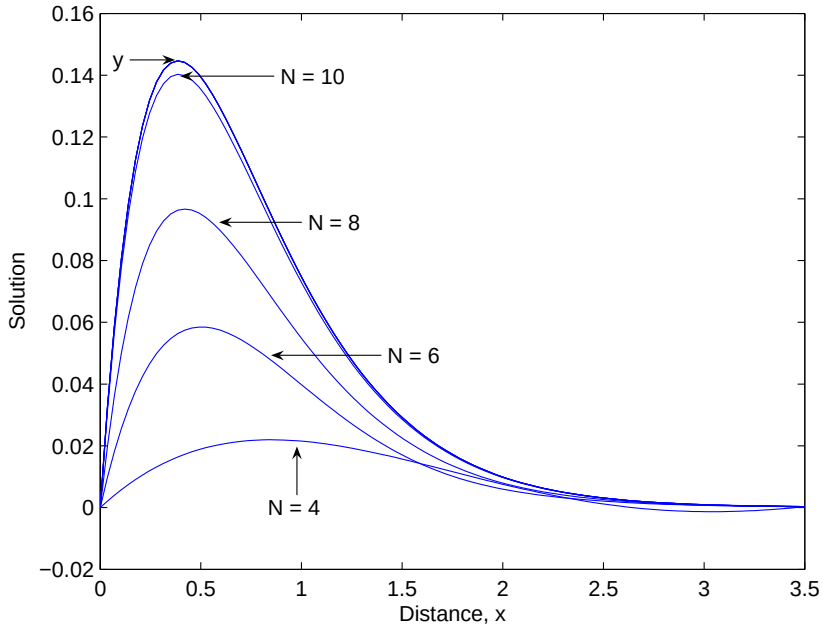

Figure 2.17: Comparison of y and y_a for Example 2.5.2 for orders $N = 4, 6, 8, 10$ with collocation points chosen as the roots of U_{N-1}.

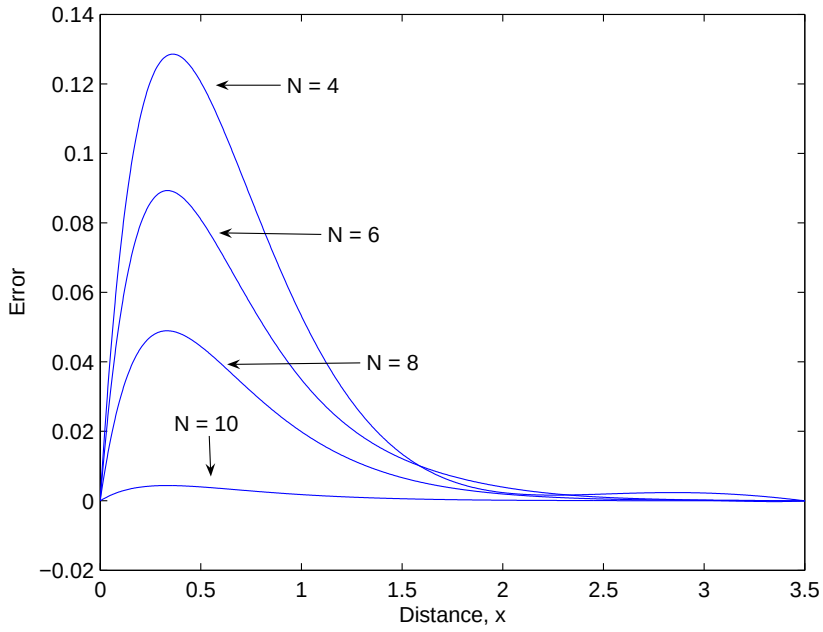

Figure 2.18: Errors for Example 2.5.2 for orders $N = 4, 6, 8, 10$, with collocation points chosen as the roots of U_{N-1}.

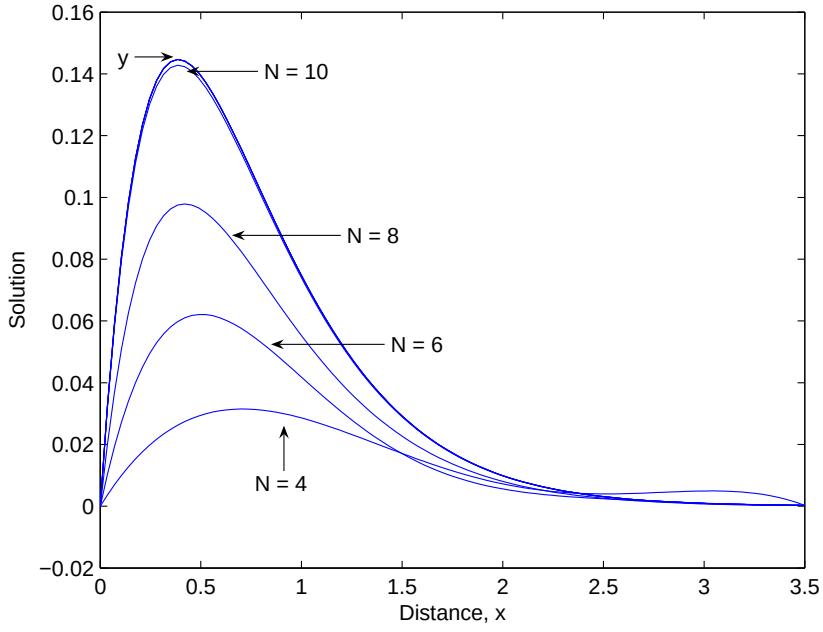

Figure 2.19: Comparison of y and y_a for Example 2.5.2 for orders $N = 4, 6, 8, 10$, with collocation points chosen as the roots of P_{N-1}.

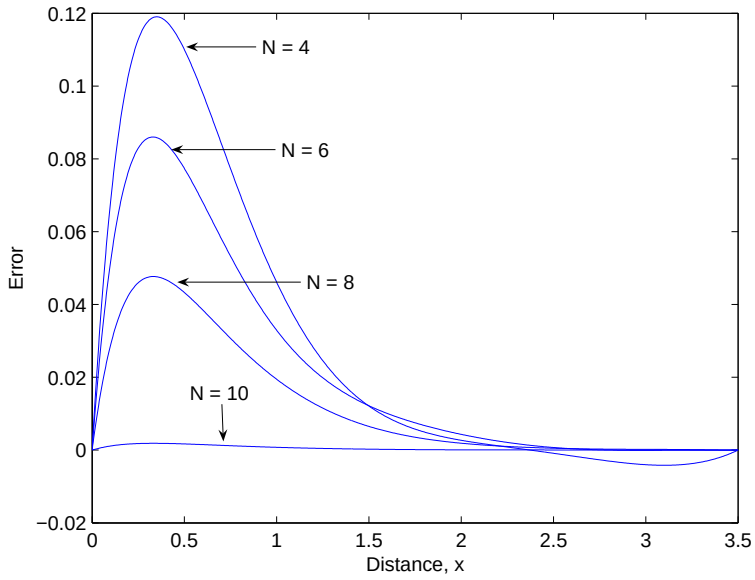

Figure 2.20: Errors for Example 2.5.2 for Orders $N = 4, 6, 8, 10$, with collocation points chosen as the roots of P_{N-1}.

Order N	$\mathbf{T_{N-1}}$	$\mathbf{U_{N-1}}$	$\mathbf{P_{N-1}}$
3	0.2504	0.2735	0.2678
4	0.1982	0.2418	0.2208
6	0.1479	0.1656	0.1587
8	0.1144	0.0915	0.0892
9	0.0252	0.0082	0.0035
10	0.0385	0.1393	0.2776
13	$1.2784e-004$	$1.5388e-004$	$1.7086e-005$
17	$1.4114e-005$	$1.4404e-005$	$1.4216e-005$
20	$1.4216e-005$	$1.4215e-005$	$1.4216e-005$
21	$1.4216e-005$	$1.4216e-005$	$1.4216e-005$

Table 2.5: Numerical comparison of the total errors for Example 2.5.2 for different orders, with collocation points chosen as the roots of T_{N-1}, U_{N-1} and P_{N-1} respectively.

Chapter 3

Application of Orthogonal Collocation on Finite Elements (OCFE) to Solving ODE's

3.1 Introduction

The orthogonal collocation method exhibits slow convergence in the case of stiff systems of boundary value problems [4, 65]. To circumvent this problem, we combine the features of the collocation method and the finite element method which gives rise to the orthogonal collocation on finite elements method, hence forth simply referred to as OCFE.

The finite element method is a widely used general purpose technique for the numerical solution of differential equations in engineering and applied mathematics. It involves a mesh discretization of a continuous domain into a set of discrete subdomains, called elements [35, 50]. The finite element method originated from the need for solving complex elasticity and structural analysis problems in civil and aeronautical engineering [6, 58, 75]. It was first proposed by Alexander Hrennikoff

(1941) and Richard Courant (1942) [14, 13]. Since then it has found its uses in fields of engineering and applied mathematics [27, 37, 56]. The accuracy, compatibility and easy adaptability of the finite element method makes it convenient to apply along with the collocation method [53, 61, 66].

OCFE was first proposed by Paterson and Cresswell [53]. Thereafter, Carey and Finlayson [11] used it to solve problems arising from chemistry. Many other investigators have used this method to solve several problems [4, 38]. The OCFE is particularly useful in the field of chemical engineering. In the case of orthogonal collocation, when the reaction rate is very high compared to the diffusion rate, the concentration profile of the chemical is very sharp, hence a high order polynomial will be required to achieve a reasonable accuracy [45, 62, 67]. This is not always advisable as discussed in chapter 2 (see Remark 2.5.3).

3.2 Methodology

Assume that a second order differential equation in the dependent variable y and independent variable x is defined on the domain $[a, b]$, with two known boundary conditions. This domain $[a, b]$ is divided into N_e smaller sub-domains of finite length called elements, then the orthogonal collocation method is applied within each element. The solution is hence a function of the number of elements N_e and will sometimes be denoted by y_{N_e} for equally spaced elements and $\bar{y}_{N_e}$ for unequally spaced elements in order to avoid confusion.

Let $x_i,\ i = 1, 2, ..., N_e + 1$ denote the coordinates of the element boundaries and $h_i = x_{i+1} - x_i$, for $i = 1, 2, ..., N_e$ denote the length of the i_{th} element. Each element $[x_i, x_{i+1}]$ is mapped to the interval $[a, b]$ by using the linear transformation

$$u^i = \frac{b-a}{h_i}(x - x_i) + a. \tag{3.1}$$

Hence forth we shall simply use the variable u to denote this transformation and bear in mind that u is a function of the element boundaries. As x varies from x_i

to x_{i+1}, u varies from a to b. This notation is illustrated in Figure 3.1.

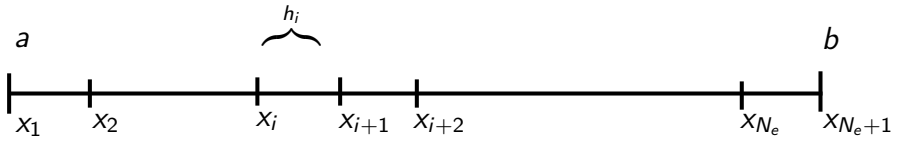

Figure 3.1: Arrangement of sub-domains in the OCFE method.

Let $y^i(u)$ denote the trial solution in the i_{th} element, then we can write

$$y^i(u) = \sum_{k=1}^{N+1} c_k^i l_k(u), \tag{3.2}$$

where N denotes the degree of the polynomial solution and

$$l_k(u) = \prod_{\substack{j=1 \\ j\neq k}}^{N+1} \frac{u - u_j}{u_k - u_j} \tag{3.3}$$

is the Lagrange polynomial of degree k in the variable u. Here u_j, $j = 1, 2, ..., N+1$ are the interpolation points with $u_1 = a$, $u_{N+1} = b$ and u_j, $j = 2, 3, ..., N$ are the roots of T_{N-1} shifted to the interval $[a, b]$. We note that the first and the last interpolation points coincide with the left and right boundaries respectively. There are therefore a total of $N_e(N+1)$ unknown coefficients c_k^i, $i = 1, 2, ..., N_e$, $k = 1, 2, ..., N+1$ to solve for.

We shall satisfy the boundary conditions and for a smooth solution require that the solution and its first derivative be continuous at the boundary of the elements x_i, $i = 2, 3, ..., N_e$. There are a total of $2(N_e - 1) + 2 = 2N_e$ conditions. We require $N_e(N+1) - 2N_e = N_e(N-1)$ additional conditions to solve the problem uniquely. This is achieved by choosing $N-1$ collocation points per element and satisfying the residual here, giving a total of $N_e(N-1)$ additional conditions. These collocation points will be denoted by u_i^c, $i = 2, 3, ..., N$. This notation is illustrated in Figure 3.2.

If we require the approximate solution of the differential equation at a point $x \in [a, b]$, we first check in which element x lies. If x belongs to the i_{th} element, that is

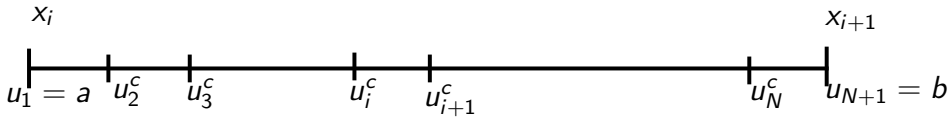

Figure 3.2: Arrangement of the collocation points in the i_{th} element $[x_i, x_{i+1}]$.

$x \in [x_i, x_{i+1}]$, then we evaluate u from equation (3.1) and recover the approximate solution from equation (3.2).

3.2.1 Distribution of Elements

Identifying how to distribute elements within any given domain $[a, b]$ is a complicated step that has been quite challenging in the method of finite elements. Investigators like Carey and Finlayson [11], Paterson and Cresswell [53], Liu and Jacobsen [37], Arora, Dhaliwal and Kukreja [3] have given several formulas for placing elements. However, we will stick to the convention given by Finlayson [22]. He proposed that better results can be obtained in the case of unequal element spacing than the equal element spacing. For the unequal element spacing, we will place smaller elements in regions where the solution is steep and larger elements where the solution varies less.

3.3 Numerical Example

Example 3.3.1 *Consider the differential equation* (2.57) *in Example* 2.5.2*, chapter* 2*. We are particularly interested in overcoming the poor approximation obtained in the region close to the hump of the exact solution (see Figure* 2.12*) so we solve this problem using the OCFE method.*

3.3.1 Two Finite Elements for Example 3.3.1

We use two elements ($N_e = 2$) with sub-domains $[x_1, x_2]$ and $[x_2, x_3]$. Each element is then mapped to the domain $[-1, 1]$ using the transformation

$$u = \frac{2}{h_i}(x - x_i) - 1, \quad i = 1, 2, \tag{3.4}$$

which is obtained from equation (3.1). Using a polynomial of order $N = 3$ for the trial solution, we write

$$y^i(u) = \sum_{k=1}^{4} c_k^i l_k(u), \quad i = 1, 2. \tag{3.5}$$

The approximate solution in equation (3.5) satisfies the differential equation

$$\frac{16}{h^2 h_i^2}\frac{d^2 y^i}{du^2} + \frac{24}{h h_i}\frac{dy^i}{du} + 9y = e^{-1.5h[h_i(u+1)/2 + x_i + 1]}, \quad i = 1, 2, \tag{3.6}$$

where $h = 3.5$ is the length of the original domain $[0, 3.5]$. Since we require two collocation points per element, they will be chosen as the roots of an orthogonal polynomial of order two, namely u_2^c and u_3^c. Together with the boundary points $u_1 = -1$ and $u_4 = 1$, we have four nodes: $u_1 = -1$, u_2^c, u_3^c and $u_4 = 1$.

Substituting the approximate solution in equation (3.5) into the differential equation (3.6) gives the residual in the i_{th} element

$$\begin{aligned} R^i(u) = \sum_{k=1}^{4} \left[\frac{16}{h^2 h_i^2} c_k^i l_k''(u) + \frac{24}{h h_i} c_k^i l_k'(u) + 9 c_k^i l_k(u) \right] \\ - e^{-1.5h[h_i(u+1)/2 + x_i + 1]}, \ i = 1, 2. \end{aligned} \tag{3.7}$$

We satisfy the residual equation for each element at the collocation points u_2^c and u_3^c to obtain

$$R^i(u_j^c) = 0, \quad i = 1, 2. \ \ j = 2, 3. \tag{3.8}$$

The left boundary condition falls in element one, hence $y^1(x_1) = y^1(u_1) = 0$. This yields

$$c_1^1 = 0. \tag{3.9}$$

Similarly the right boundary condition falls in element two, hence $y^2(x_3) = y^2(u_4) = 9.625e^{-10.5}$. This yields

$$c_4^2 = 9.625e^{-10.5}. \tag{3.10}$$

Since we have a total of eight unknowns, we need two more additional equations for a unique solution. These are obtained from the continuity conditions

$y^1(x_2) = y^2(x_2)$ and $\frac{dy^1}{dx}|_{x_2} = \frac{dy^2}{dx}|_{x_2}$ which are equivalent to $y^1(u_4) = y^2(u_1)$ and $\frac{2}{h_1}\frac{dy^1}{du}|_{u_4} = \frac{2}{h_2}\frac{dy^2}{du}|_{u_1}$ in the variable u. The continuity of the functions yields

$$\sum_{k=1}^{4} c_k^1 l_k(u_4) = \sum_{k=1}^{4} c_k^2 l_k(u_1), \tag{3.11}$$

which simplifies to

$$c_4^1 - c_1^2 = 0. \tag{3.12}$$

and the continuity of the derivatives yields

$$\sum_{k=1}^{4} \left[c_k^1 \frac{l_k'(u_4)}{h_1} - c_k^2 \frac{l_k'(u_1)}{h_2} \right] = 0. \tag{3.13}$$

Thus we have a system of equations which we choose to arrange in the order, (3.9); (3.8), for $i = 1$, $j = 1, 2$; (3.12); (3.13); (3.8), for $i = 2$, $j = 1, 2$ and (3.10). This gives a matrix vector form $\mathbf{Ac} = \mathbf{b}$ which is depicted in the system of equations (3.14).

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} & 0 & 0 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ \frac{l_1'(u_4)}{h_1} & \frac{l_2'(u_4)}{h_1} & \frac{l_3'(u_4)}{h_1} & \frac{l_4'(u_4)}{h_1} & \frac{-l_1'(u_1)}{h_2} & \frac{-l_2'(u_1)}{h_2} & \frac{-l_3'(u_1)}{h_2} & \frac{-l_4'(u_1)}{h_2} \\ 0 & 0 & 0 & 0 & a_{65} & a_{66} & a_{67} & a_{68} \\ 0 & 0 & 0 & 0 & a_{75} & a_{76} & a_{77} & a_{78} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_1^1 \\ c_2^1 \\ c_3^1 \\ c_4^1 \\ c_1^2 \\ c_2^2 \\ c_3^2 \\ c_4^2 \end{pmatrix} = \begin{pmatrix} 0 \\ b_2 \\ b_3 \\ 0 \\ 0 \\ b_6 \\ b_7 \\ 9.625e^{-10.5} \end{pmatrix} \tag{3.14}$$

where

$$a_{ij} = \frac{16}{h^2 h_1^2} l_j''(u_i^c) + \frac{24}{h h_1} l_j'(u_i^c) + 9 l_j(u_i^c), \tag{3.15}$$

$$a_{i+4\ j+4} = \frac{16}{h^2 h_2^2} l_j''(u_i^c) + \frac{24}{h h_2} l_j'(u_i^c) + 9 l_j(u_i^c), \tag{3.16}$$

and

$$b_i = e^{-1.5h[h_1(u_i^c+1)/2+x_1+1]}, \tag{3.17}$$

$$b_{i+4} = e^{-1.5h[h_2(u_i^c+1)/2+x_2+1]}, \tag{3.18}$$

for $j = 1, 2, 3, 4$ and $i = 2, 3$.

Firstly, we use two equally spaced elements with sub-domains $[-1, 0]$ and $[0, 1]$. The interpolation points are $u_1 = -1$, $u_2 = -1/\sqrt{2}$, $u_3 = 1/\sqrt{2}$ and $u_4 = 1$. Where u_2 and u_3 are chosen as the roots of T_2. The collocation points $u_2^c = -1/\sqrt{3}$ and $u_3^c = 1/\sqrt{3}$ are chosen as the roots of the Legendre polynomial, P_2 since they gave good results in chapter 2. The solution y_2 is given in Figure 3.3 and the error is shown in Figure 3.4.

We observe from Figure 3.3 that the approximate solution y_2 is poor in the interval $(0, 1)$. Hence we consider the case of two unequally spaced elements $[0, 1]$ and $[1, 3.5]$ which corresponds to $[-1, -0.4286]$ and $[-0.4286, 1]$ on the domain $[-1, 1]$ and denote the solution by $\bar{y}_2$. The solution and error are also presented in Figures 3.3 and 3.4 respectively.

A set of discrete numerical results is summarized in Table 3.1 for y_2 and $\bar{y}_2$. Comparing the the total error for y_2 and $\bar{y}_2$ from Table 3.1, we note a reduction of 17.24% for the latter case as compared to the former. Hence it is obvious from our analysis that properly chosen unequal element spacing gives superior results as proposed by Finlayson [22].

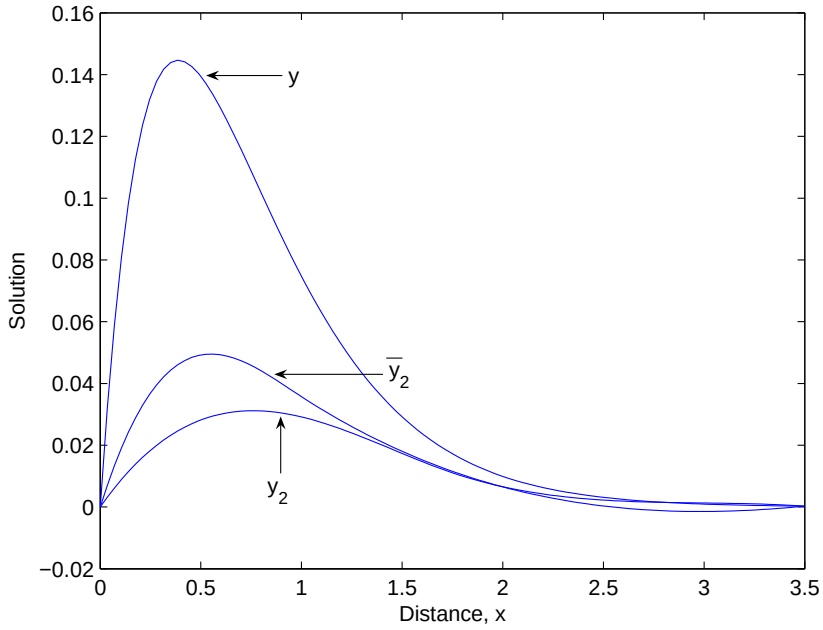

Figure 3.3: Comparison of y, y_2 and $\bar{y}_2$.

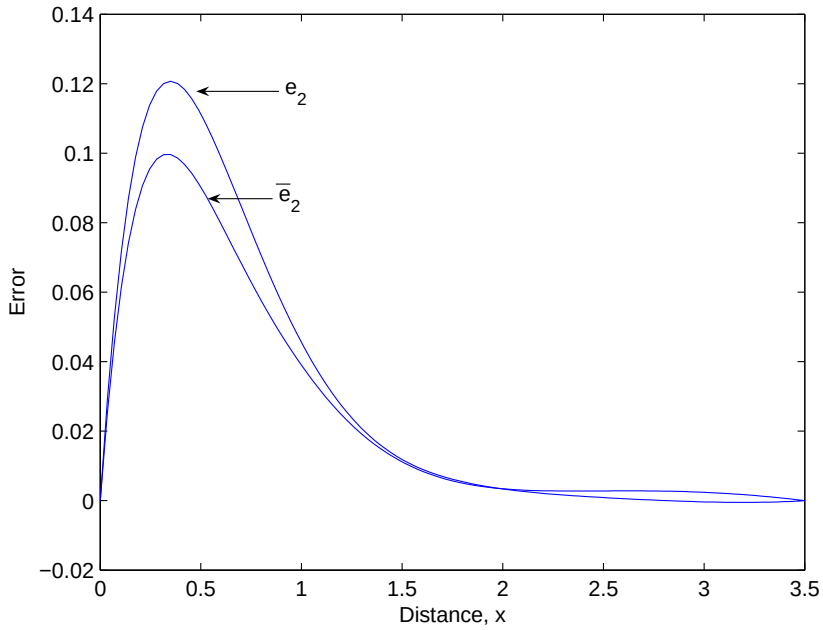

Figure 3.4: Errors $e_2 = y - y_2$ and $\bar{e}_2 = y - \bar{y}_2$.

x	y	y_2	$\bar{y}_2$
0	0	−0.0000	0
0.1750	0.1126	0.0138	0.0288
0.3500	0.1439	0.0232	0.0443
0.5250	0.1372	0.0287	0.0494
0.7000	0.1157	0.0310	0.0473
0.8750	0.0911	0.0307	0.0411
1.0500	0.0686	0.0283	0.0336
1.2250	0.0501	0.0246	0.0269
1.4000	0.0357	0.0200	0.0210
1.5750	0.0250	0.0153	0.0159
1.7500	0.0172	0.0111	0.0115
1.9250	0.0117	0.0077	0.0079
2.1000	0.0079	0.0052	0.0049
2.2750	0.0053	0.0035	0.0025
2.4500	0.0035	0.0025	0.0007
2.6250	0.0023	0.0018	−0.0005
2.8000	0.0015	0.0015	−0.0012
2.9750	0.0010	0.0013	−0.0015
3.1500	0.0006	0.0011	−0.0013
3.3250	0.0004	0.0008	−0.0007
3.5000	0.0003	0.0003	0.0003
	Total error ⇒	**0.2227**	**0.1843**

Table 3.1: Numerical comparison of y, y_2 and $\bar{y}_2$ at different values of x.

3.3.2 Generalization of Example 3.3.1

We can improve the accuracy of the result by using more elements (N_e elements). In general substituting the approximate solution of the form equation (3.5) into the differential equation (3.6) for $i = 1, 2, ..., N_e$ gives the residual in the i_{th} element, thus the residual equation in equation (3.7) holds for $i = 1, 2, ..., N_e$. We satisfy the residual equation for each element at the collocation points u_2^c and u_3^c, thus equation (3.8) holds for $i = 1, 2, ..., N_e$.

The left boundary condition will always lie in the first element hence equation (3.9) holds. The right boundary conditions falls in the last element N_e, hence $y^{N_e}(u_4) = 9.625e^{-10.5}$. This yields

$$c_4^{N_e} = 9.625e^{-10.5}. \tag{3.19}$$

The continuity conditions at x_{i+1} are $y^i(u_4) = y^{i+1}(u_1)$ and $\frac{2}{h_i}\frac{dy^i}{du}|_{u_4} = \frac{2}{h_{i+1}}\frac{dy^{i+1}}{du}|_{u_1}$ in the variable u. The continuity of the function yields

$$\sum_{k=1}^{4} c_k^i l_k(u_4) = \sum_{k=1}^{4} c_k^{i+1} l_k(u_1), \quad i = 1, 2, ..., N_e - 1 \tag{3.20}$$

which simplifies to

$$c_4^i - c_1^{i+1} = 0, \tag{3.21}$$

and the continuity of the derivative yields

$$\sum_{k=1}^{4} \left[c_k^i \frac{l_k'(u_4)}{h_i} - c_k^{i+1} \frac{l_k'(u_1)}{h_{i+1}} \right] = 0, \quad i = 1, 2, ..., N_e - 1. \tag{3.22}$$

Thus we have a system of equations which gives the matrix vector form $\mathbf{Ac} = \mathbf{b}$ having a similar form as the one given in the system of equation (3.14) where $\mathbf{A}$ is a $4N_e \times 4N_e$ matrix and $\mathbf{b}$ and $\mathbf{c}$ are $4N_e \times 1$ column vectors. A MATLAB code for this system is given in Appendix C.

Of particular interest is the solution y_3 for three equally spaced elements and $\bar{y}_3$ for the corresponding unequally spaced elements. The solution y_3 for three equally

spaced sub-domains $[-1, -1/3]$, $[-1/3, 1/3]$ and $[1/3, 1]$ is given in Figure 3.5 and the error is shown in Figure 3.6.

The negative slope near the origin for y_3 is a cause for concern (see Figure 3.5). In addition, the function is negative on $(0, 0.5)$ and concave up. Therefore we consider using three unequally spaced sub-domains, namely $[0, 0.1]$, $[0.1, 1.5]$, and $[1.5, 3.5]$ which corresponds to $[-1.0000, -0.9429]$, $[-0.9429, -0.3143]$, and $[-0.3143, 1.0000]$ on the domain $[-1, 1]$. As observed from Figure 3.5, we obtain a better approximation $\bar{y}_3$.

A set of discrete numerical results for y_3 and $\bar{y}_3$ is summarized in Table 3.2, from which we deduce that the error is reduced by 28.92% when we use unequally spaced elements. Hence it is obvious from our analysis that properly chosen unequal element spacing gives superior results as proposed by Finlayson [22]. The corresponding error plot is as shown in Figure 3.6.

As we increase the number of elements, the approximate solution tends to converge to the exact solution as shown in Figure 3.7 for y_6 and $\bar{y}_6$. The corresponding error plots are given in Figure 3.8. The error plots for y_{18}, y_{21}, y_{24} and y_{27} for more elements are given in Figure 3.9. Observe the gradual decrease in the error as we increase the number of elements. If we use $N_e = 75$ elements, the total error levels off at $2.7509e - 007$ which is better than the corresponding total error of $1.4216e - 005$ obtained in chapter 2, Example 2.5.2 (see Table 2.5).

We can optimize the OCFE method by concentrating elements in areas with steep gradients [11]. Table 3.3 gives a comparison of the total error for different number of equal and unequally spaced elements N_e.

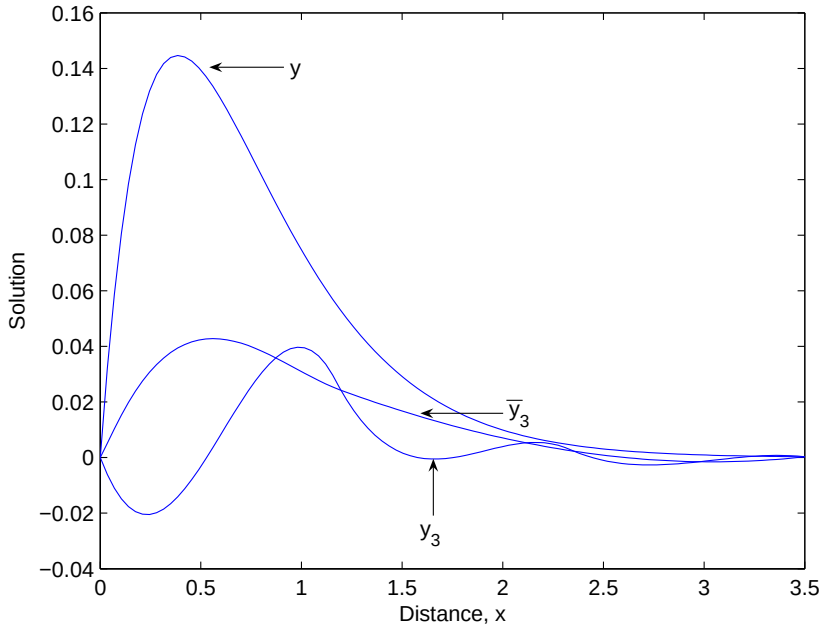

Figure 3.5: Comparison of y, y_3 and $\bar{y}_3$.

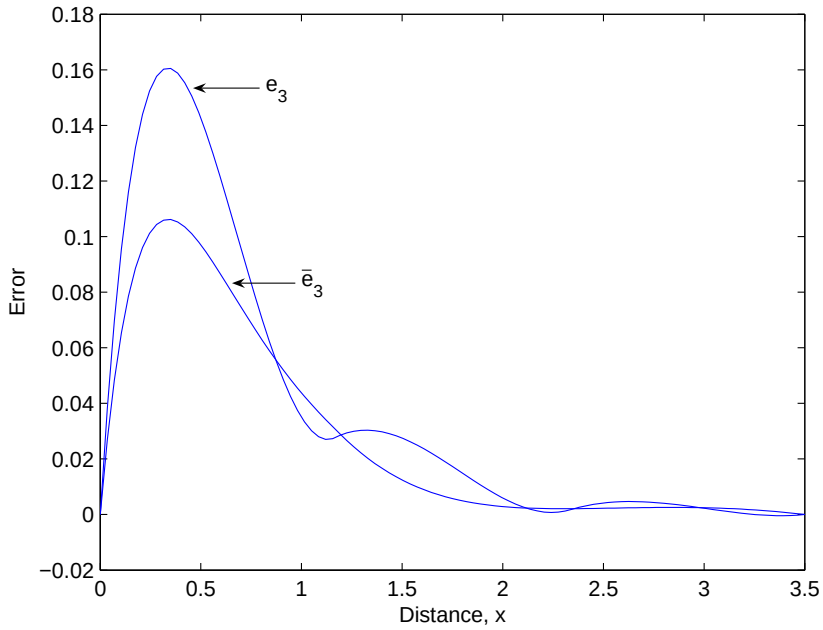

Figure 3.6: Errors $e_3 = y - y_3$ and $\bar{e}_3 = y - \bar{y}_3$.

x	y	y_3	$\bar{y}_3$
0	0	−0.0000	0
0.1750	0.1126	−0.0194	0.0239
0.3500	0.1439	−0.0166	0.0378
0.5250	0.1372	−0.0004	0.0427
0.7000	0.1157	0.0201	0.0412
0.8750	0.0911	0.0361	0.0357
1.0500	0.0686	0.0385	0.0290
1.2250	0.0501	0.0207	0.0234
1.4000	0.0357	0.0059	0.0191
1.5750	0.0250	0.0000	0.0151
1.7500	0.0172	−0.0000	0.0115
1.9250	0.0117	0.0027	0.0083
2.1000	0.0079	0.0052	0.0055
2.2750	0.0053	0.0044	0.0032
2.4500	0.0035	−0.0001	0.0013
2.6250	0.0023	−0.0024	−0.0001
2.8000	0.0015	−0.0025	−0.0010
2.9750	0.0010	−0.0015	−0.0015
3.1500	0.0006	−0.0001	−0.0014
3.3250	0.0004	0.0007	−0.0009
3.5000	0.0003	0.0003	0.0003
	Total Error ⇒	**0.2794**	**0.1986**

Table 3.2: Numerical comparison of y_3 and $\bar{y}_3$.

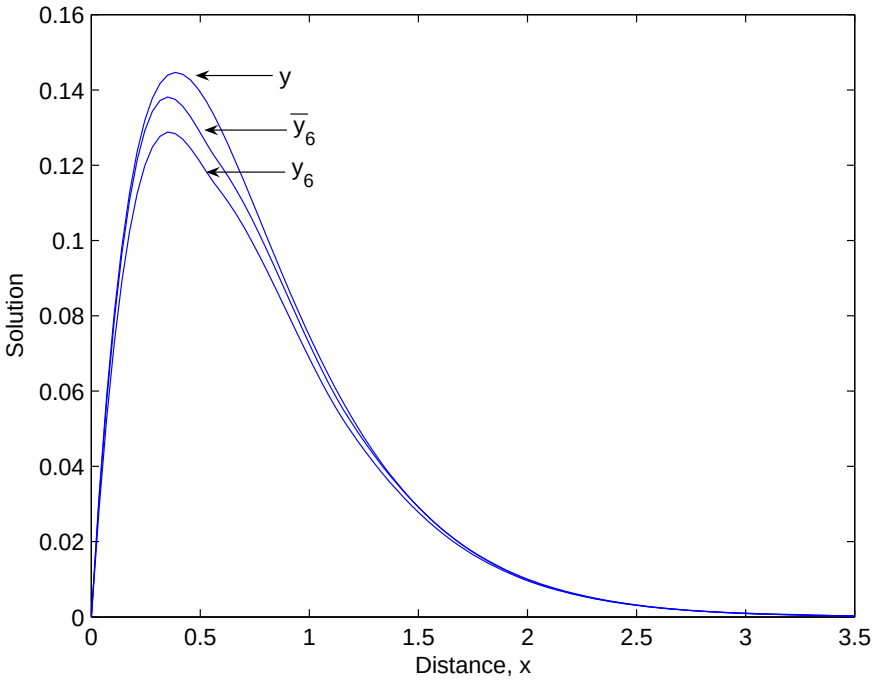

Figure 3.7: Comparison of y, y_6 and $\bar{y}_6$.

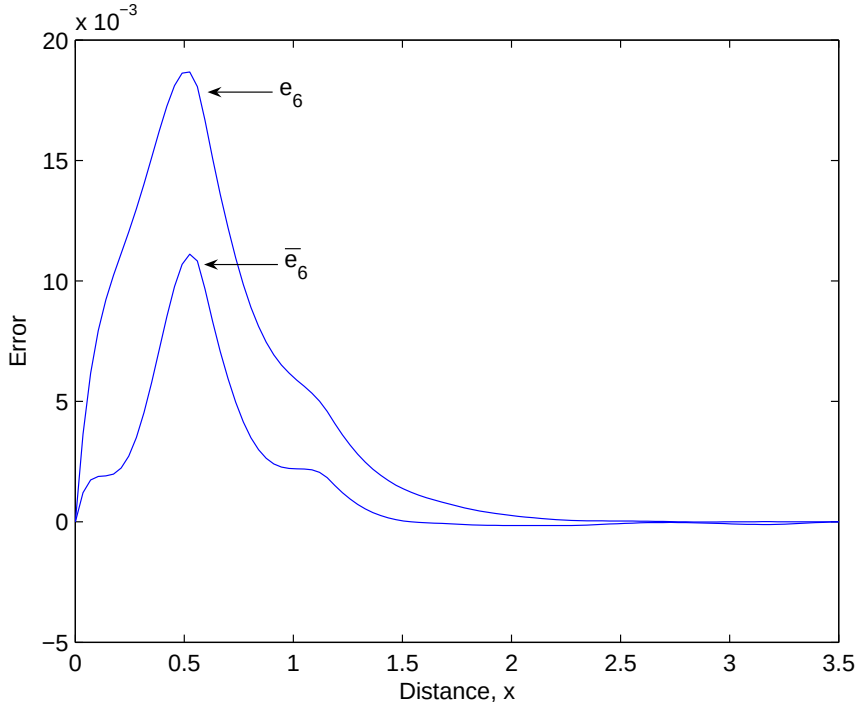

Figure 3.8: Errors $e_6 = y - y_6$ and $\bar{e}_6 = y - \bar{y}_6$.

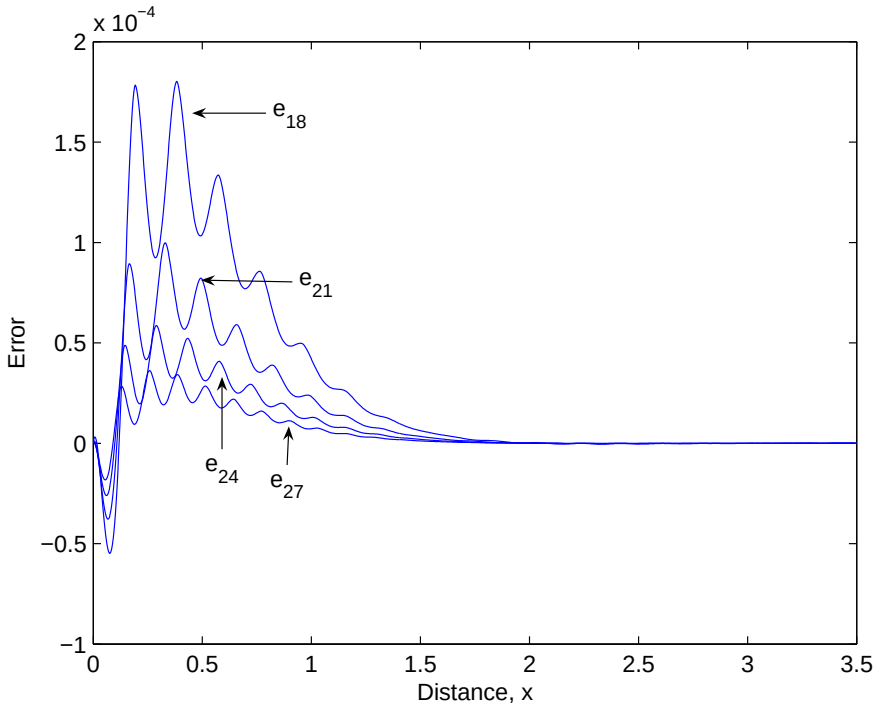

Figure 3.9: Errors $e_i = y - y_i$.

i	**Total error for $\mathbf{y_i}$**	**Total error for $\bar{\mathbf{y}}_\mathbf{i}$**
2	0.2227	0.1937
3	0.2794	0.1893
15	$4.7522e-004$	$9.4921e-005$
18	$2.6755e-004$	$6.4803e-005$
21	$1.5664e-004$	$5.6105e-005$
24	$6.8622e-005$	$9.4835e-006$
27	$4.3342e-005$	$9.4578e-006$
30	$3.0521e-005$	$9.2446e-006$
75	$2.7509e-007$	$3.2246e-008$

Table 3.3: Total error as a function of the number of elements(**i**) for equal and unequal element spacing.

Remark 3.3.2 *Notice that we have used a local numbering for the coefficients* c_k^i *in the* i_{th} *element for* $i = 1, 2, ..., N_e$, $k = 1, 2, 3, 4$, *there are therefore* $4N_e$ *unknowns. However since* $c_4^i = c_1^{i+1}$, $i = 1, 2, ..., N_e - 1$ *from equation* (3.21), *the number of unknowns can be reduced to* $3N_e + 1$ *and the solution in the* i_{th} *element written as*

$$y^i(u) = \sum_{k=1}^{4} c_{k+3(i-1)} l_k(u), \ i = 1, 2, ..., N_e. \tag{3.23}$$

This yields a global numbering as shown in Figure 3.10. *The advantage of this is that the dimension of the matrix is reduced and so is the numerical effort for the solution of the linear system.*

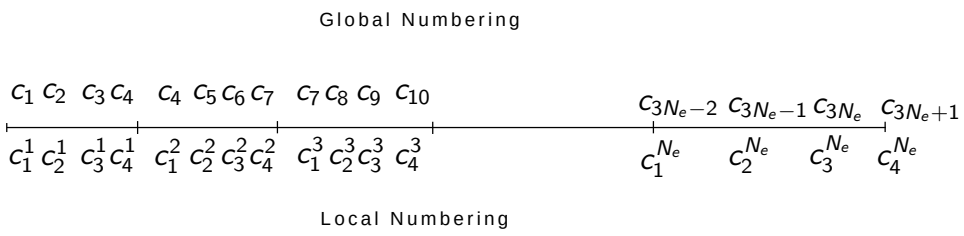

Figure 3.10: Global and local numbering of coefficients for the OCFE method.

Chapter 4

Application of Orthogonal Collocation on Finite Elements (OCFE) to Solving PDE's

4.1 Introduction

Partial differential equations in engineering can be solved analytically by using techniques like the separation of variables and the Laplace transform. However some of these techniques have limited applications. In particular the Laplace transform which is a useful tool for solving partial differential equations in chemical engineering could involve the solution of complicated and transcendental equations which are time consuming [32, 73].

The solutions to these partial differential equations can be approximated by using the method of OCFE. Here, the process is more complicated than OCFE applied to ODE's since it gives rise to a combination of ordinary differential equations and algebraic equations. Hence we obtain a set of coupled differential algebraic equations which we call DAE's for short [49, 54].

4.2 Numerical Example

Example 4.2.1 *Consider a linear diffusion convection problem with mixed boundary conditions and initial condition which is encountered in different branches of chemical engineering:*

$$\begin{aligned} \frac{\partial y(x,t)}{\partial t} &= \frac{1}{4P}\frac{\partial^2 y(x,t)}{\partial x^2} - \frac{\partial y(x,t)}{\partial x}, \\ y(x,t) - \frac{1}{4P}\frac{\partial y(x,t)}{\partial x} &= 0, \quad \text{at } x = 0, \quad \text{for} \quad t \geq 0, \\ \frac{\partial y(x,t)}{\partial x} &= 0, \quad \text{at} \quad x = 1, \quad \text{for} \quad t \geq 0, \\ y(x,0) &= 1, \quad \text{for all} \quad x. \end{aligned} \tag{4.1}$$

The problem is used to describe the displacement of an initial homogeneous solute from a medium (called bed in chemistry) of finite length x (in this case $x = 1$) by the introduction of a solvent. Here y represents the concentration profile of the solute and P is a constant parameter called the Peclet number.

Equation (4.1) has been solved analytically by Brenner [9]. The solution is given by

$$y(x,t) = 2e^{P(2x-t)} \sum_{k=1}^{\infty} \frac{P\lambda_k}{(\lambda_k^2 + P^2 + P)(\lambda_k^2 + P^2)} (\lambda_k \cos 2\lambda_k x + P \sin 2\lambda_k x) e^{-\lambda_k^2 t/P}, \tag{4.2}$$

where λ_k for $k = 1, 2, ...$ are the positive roots taken in order of increasing magnitude of the transcendental equation

$$\tan 2\lambda_k = \frac{2\lambda_k P}{\lambda_k^2 - P^2}. \tag{4.3}$$

4.2.1 Exit Solution

Chemical engineers are particularly interested in the exit solution, that is

$$ye_1(1,t) = e^{P(2-t)} \sum_{k=1}^{\infty} e^{-\lambda_k^2 t/P} \frac{\lambda_k \sin 2\lambda_k}{\lambda_k^2 + P^2 + P}, \tag{4.4}$$

of the solute leaving the bed which is obtained from equation (4.2) by setting $x = 1$. However, the exit solution given in equation (4.4) converges too slowly to be of much practical use for large P and/or small t. Hence Brenner [9] derived an asymptotic solution using Laplace transforms for large P and/or small t. This asymptotic concentration is given by

$$\begin{aligned} y(x,t) = 1 - 0.5\ \mathrm{erfc}(z_1) - (4Pt/\pi)^{0.5}\, e^{-P(x-t)^2/t} \\ + 0.5\,[1 + 4P(x+t)]\, e^{4Px}\mathrm{erfc}(z_2) \\ - 2\,(4Pt/\pi)^{0.5}\,[1 + P(2-x+t)]e^{4P - \frac{P(2-x+t)^2}{t}} \\ + 2P\left[2(2-x+t) + t + 2P(2-x+t)^2\right] e^{4P}\mathrm{erfc}(z_3), \end{aligned} \tag{4.5}$$

where

$z_1(x,t) = (P/t)^{0.5}\,(x-t)$, $z_2(x,t) = (P/t)^{0.5}\,(x+t)$, $z_3(x,t) = (P/t)^{0.5}\,(2-x+t)$ and erfc denotes the complimentary error function defined in terms of the error function erf, that is

$$\mathrm{erfc}(z) = 1 - \mathrm{erf}(z) = \frac{2}{\sqrt{\pi}} \int_z^\infty e^{-y^2} dy. \tag{4.6}$$

We obtain the required exit solution by setting $x = 1$ in equation (4.5) and simplifying, thus we have

$$\begin{aligned} ye_2(1,t) = 1 - 0.5\ \mathrm{erfc}(z_1) - (4Pt/\pi)^{0.5}\,[3 + 2P(1+t)]e^{-P(1-t)^2/t} \\ + \left(0.5 + 2P(3+4t) + 4P^2(1+t)^2\right) e^{4P}\mathrm{erfc}(z_2), \end{aligned} \tag{4.7}$$

where $z_1(1,t) = (P/t)^{0.5}\,(1-t)$ and $z_2(1,t) = z_3(1,t) = (P/t)^{0.5}\,(1+t)$.

Here we choose to work with the original domain $[0,1]$. As in the case of OCFE applied to ODEs, the domain $[0,1]$ is split into N_e elements. Each element is then mapped to the domain $[0,1]$ using the transformation $u = \frac{x - x_i}{h_i}$, where as before $h_i = x_{i+1} - x_i$ is the i_{th} element width.

We obtain the approximate solution in the i_{th} element by using a cubic Lagrange polynomial as in chapters 3, but noting that the approximate solution is a function of both u and t, we write

$$y^i(u,t) = \sum_{k=1}^{4} c_k^i(t) l_k(u). \tag{4.8}$$

The approximate solution in equation (4.8) satisfies the partial differential equation

$$\begin{aligned}
\frac{\partial y^i(u,t)}{\partial t} &= \frac{1}{4P\ h_i^2}\frac{\partial^2 y^i(u,t)}{\partial u^2} - \frac{1}{h_i}\frac{\partial y^i(u,t)}{\partial u}, \\
y^1(u,t) - \frac{1}{4P\ h_1}\frac{\partial y^1(u,t)}{\partial u} &= 0, \text{ at } u=0, \text{ for } t \geq 0, \\
\frac{1}{h_{N_e}}\frac{\partial y^{N_e}(u,t)}{\partial u} &= 0, \text{ at } u=1, \text{ for } t \geq 0, \\
y(u,0) &= 1, \text{ for all } u. \qquad (4.9)
\end{aligned}$$

The interpolation points $u_2 = \frac{1}{2}(1-\frac{1}{\sqrt{2}})$ and $u_3 = \frac{1}{2}(1+\frac{1}{\sqrt{2}})$ are chosen as the roots of the Chebyshev polynomials T_2 shifted to $[0,1]$. So together with the boundary points $u_1 = 0$ and $u_4 = 1$ we have four interpolation points. The collocation points $u_2^c = \frac{1}{2}(1-\frac{1}{\sqrt{3}})$ and $u_3^c = \frac{1}{2}(1+\frac{1}{\sqrt{3}})$ are chosen as the roots of the Legendre polynomial P_2 shifted to $[0,1]$.

Substituting the approximate solution in equation (4.8) into the partial differential equation (4.9) gives the residual in the i_{th} element, that is

$$R^i(u) = \sum_{k=1}^{4}\left[\frac{c_k^i(t)l_k''(u)}{4P\ h_i^2} - \frac{c_k^i(t)l_k'(u)}{h_i} - \frac{dc_k^i(t)}{dt}l_k(u)\right], \quad i = 1,2,...,N_e. \qquad (4.10)$$

We satisfy the residual equation at the collocation points u_j^c, for $j = 2,3$, that is

$$R^i(u_j^c) = 0, \quad i = 1,2,...,N_e,\ j = 2,3. \qquad (4.11)$$

The continuity of the function and its first derivative at x_{i+1}, $i = 1,2,...,N_e-1$ yields

$$\begin{aligned}
\sum_{k=1}^{4}\left[c_k^i(t)l_k(u_4) - c_k^{i+1}(t)l_k(u_1)\right] &= 0, \qquad (4.12) \\
\sum_{k=1}^{4}\left[c_k^i(t)\frac{l_k'(u_4)}{h_i} - c_k^{i+1}(t)\frac{l_k'(u_1)}{h_{i+1}}\right] &= 0. \qquad (4.13)
\end{aligned}$$

The left and right boundary conditions are given by

$$\begin{aligned}
\sum_{k=1}^{4}\left[c_k^1(t)l_k(u_1) - c_k^1(t)\frac{l_k'(u_1)}{4Ph_1}\right] &= 0, \qquad (4.14) \\
\sum_{k=1}^{4} c_k^{N_e}(t)l_k'(u_4) &= 0. \qquad (4.15)
\end{aligned}$$

The system can be written in the form

$$\mathbf{M}\frac{d\mathbf{c}}{dt} = \mathbf{J}\mathbf{c}, \tag{4.16}$$

where $\mathbf{M}$, which is called the mass matrix, is the coefficient matrix of $\frac{d\mathbf{c}}{dt}$. The vector $\mathbf{c}$ represents the unknown entries, that is

$$\mathbf{c}(t) = \left[c_1^1(t)\ c_2^1(t)\ c_3^1(t)\ c_4^1(t)\ c_1^2(t)\ ...\ c_4^{N_e}(t)\right]^T$$

and $\mathbf{J}$ is the coefficient matrix of the vector $\mathbf{c}$ which coincides with the Jacobian of the system given in equation (4.16). Since the mass matrix $\mathbf{M}$ is singular, we obtain a differential algebraic system. There is a total of $4N_e$ coupled DAE's which we solve using MATLAB with the ode15s subroutine [64]. The code is given in Appendix D. It should be noted that for this particular problem, we will only consider equal element spacing. For simplicity, the approximate solution from the OCFE method using N_e equally spaced elements will be denoted by y_{N_e}.

To obtain the exit solution in equation (4.4) for $P = 0.8$, we need to find the corresponding λ_k's which satisfy equation (4.3). Although the first six roots for different Peclet numbers have been given by Carslaw and Jaeger [12], we will show how more roots can be obtained. Define the function $f(\lambda)$ by

$$f(\lambda) = \tan 2\lambda - \frac{2\lambda P}{\lambda^2 - P^2}. \tag{4.17}$$

The zeros of $f(\lambda)$ are the roots of equation (4.3). From a plot of $f(\lambda)$, we ascertain approximations to the zeros and these are refined by using the built in Newton root finder **fsolve** in MATLAB. The code has been incorporated into the general code in Appendix D. The values of the first 17 roots for $P = 0.8$ are given in Appendix E. The error plot for y_7 at $x = 1$ with $P = 0.8$ is given in Figure 4.1. Notice from Figure 4.1 that the error for as low as 7 elements is small for $P = 0.8$.

Figure 4.2 gives the graphical representation of y_3 at $x = 1$ with $P = 20$ and the corresponding error plot is presented in Figure 4.3. As we increase the number of elements, the approximate exit solution y_i rapidly converges to the exit solution

ye_2 and the error is decreased. This is clearly seen from the error plots in Figure 4.4. The error levels off at y_{37} with value $1.1948e-004$. A similar trend is seen in Figure 4.5 for $P = 40$. Here the error levels off at y_{43} with value $9.8323e-005$ (see Figure 4.6).

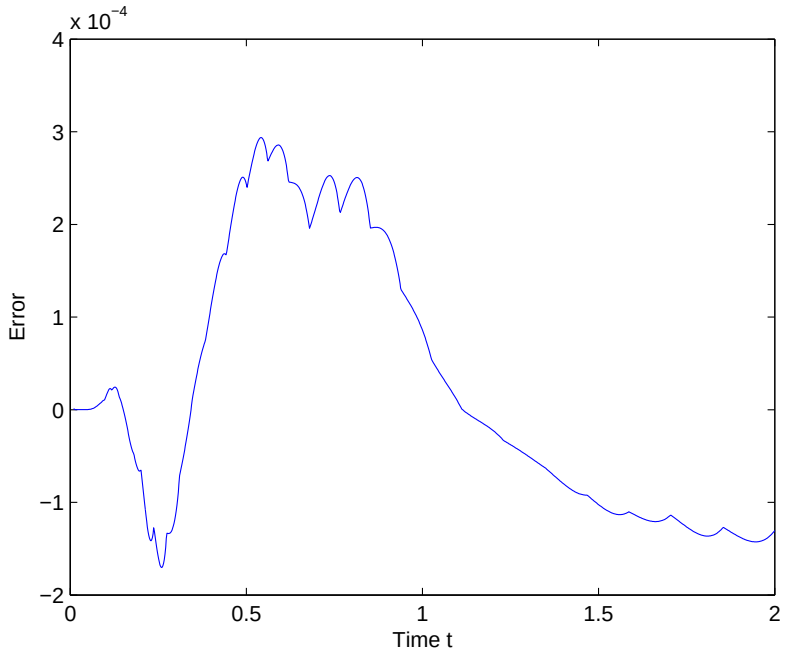

Figure 4.1: Error $ye_1 - y_7$ at $x = 1$ for $P = 0.8$.

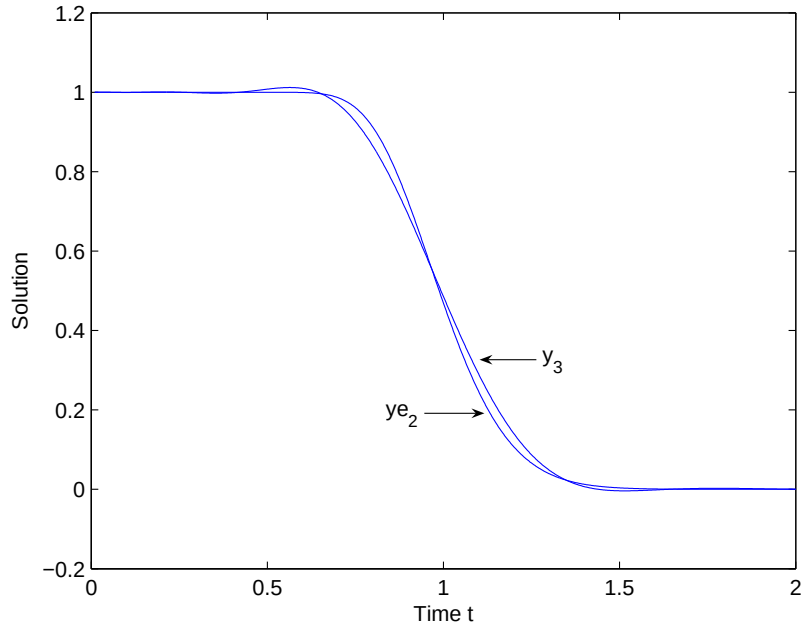

Figure 4.2: Comparison of ye_2 and y_3 at $x = 1$ for $P = 20$.

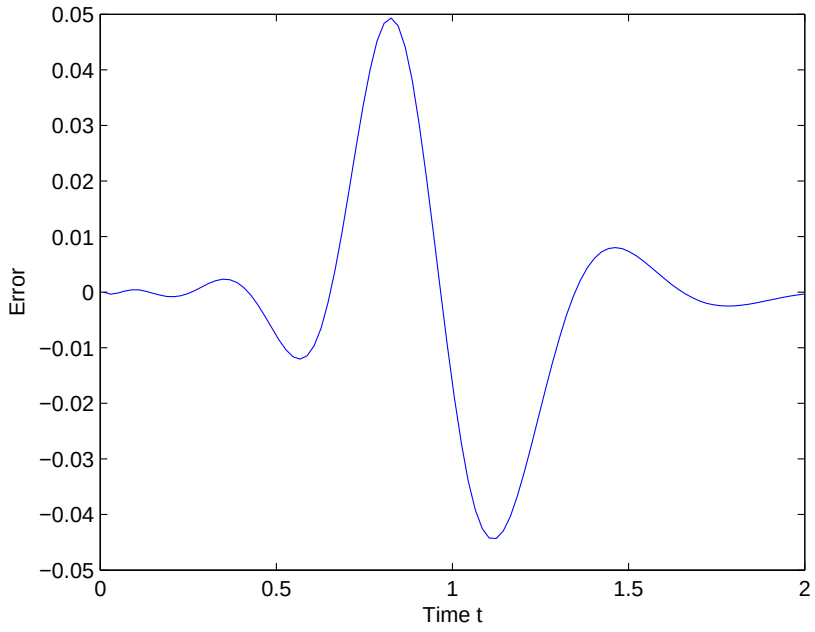

Figure 4.3: Error $ye_2 - y_3$ at $x = 1$ for $P = 20$.

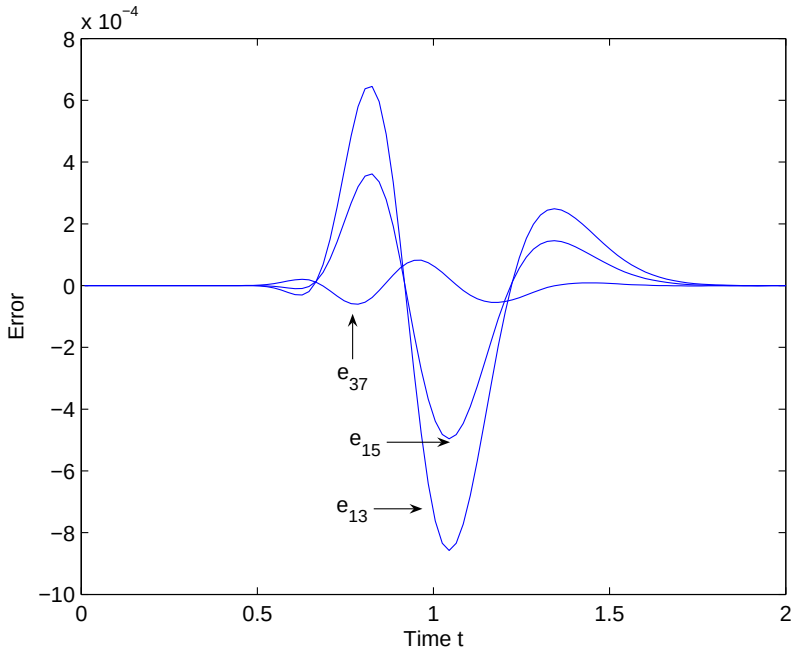

Figure 4.4: Errors $e_i = ye_2 - y_i$ at $x = 1$ for $P = 20$.

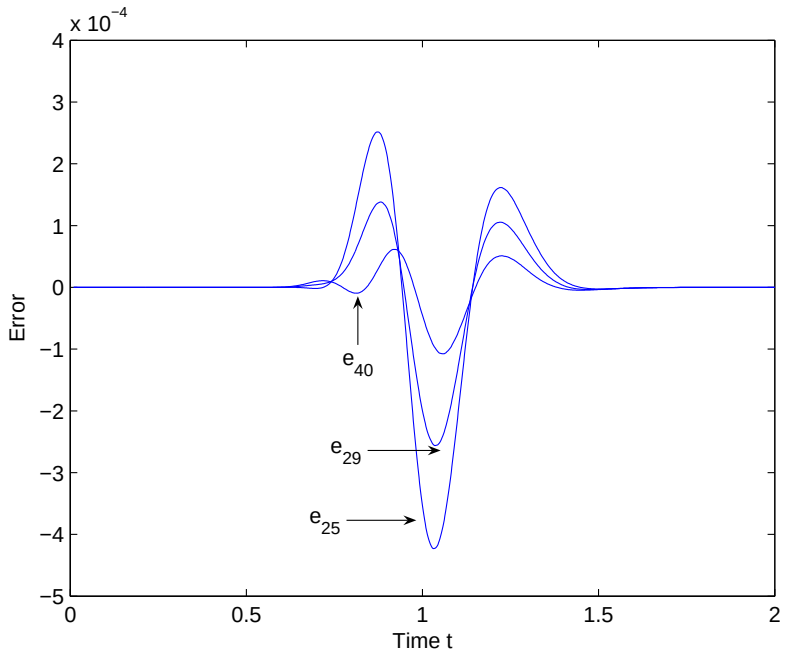

Figure 4.5: Errors $e_i = ye_2 - y_i$ at $x = 1$ for $P = 40$.

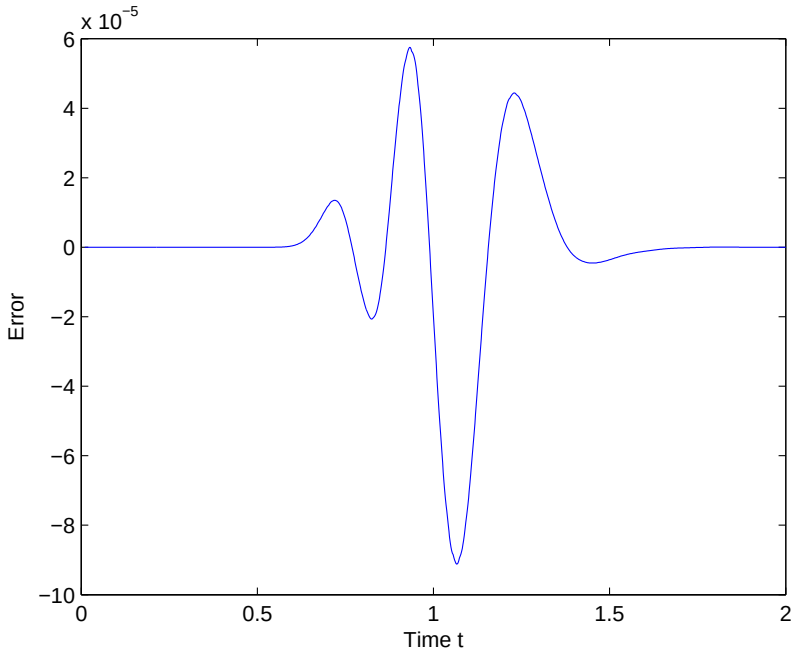

Figure 4.6: Error $e_{43} = ye_{43} - y_{43}$ at $x = 1$ for $P = 40$.

4.2.2 General Solution

For the purpose of mathematical interest, let us investigate the behavior of the general solution given in equation (4.2) at particular times in space.

The plots of the solutions and the corresponding errors for y_2, y_3 and y_4 at $t = 0.001$, $t = 1$ and $t = 2$ are presented in Figures 4.7 to 4.12. Notice from Figure 4.7 that at $t = 0.001$ which is close to $t = 0$, the exact solution deviates very slightly from $y = 1$ which is in agreement with the initial condition given in equation (4.1). As we increase the number of elements, the approximate solution y_i tends to converge to the exact solution y and the error is decreased as expected (see Figures 4.13 and 4.14 for y_{50} at $t = 0.001$). A similar trend occurs for $t = 1$ and $t = 2$ as evident from Figures 4.15 and 4.16 respectively.

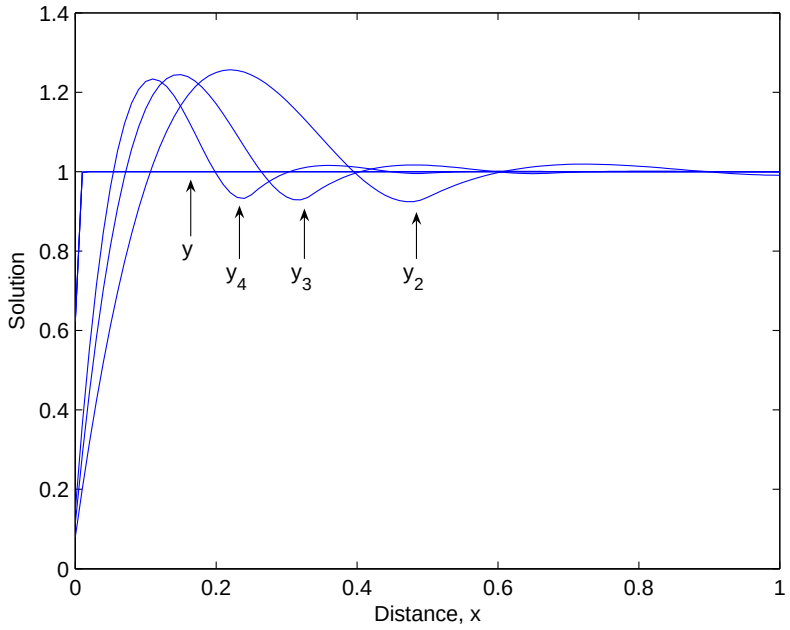

Figure 4.7: Comparison of y with y_i at $t = 0.001$ for $P = 40$.

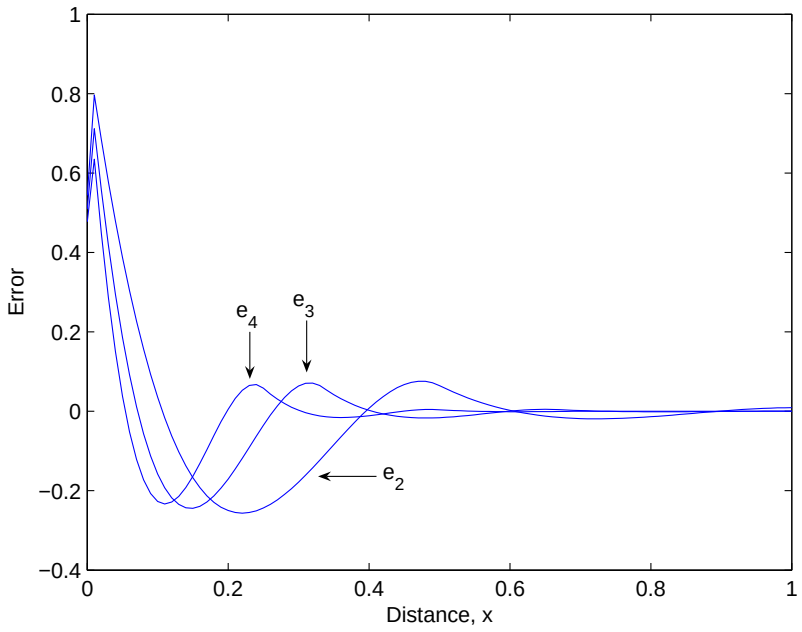

Figure 4.8: Errors $e_i = y - y_i$ at $t = 0.001$ for $P = 40$.

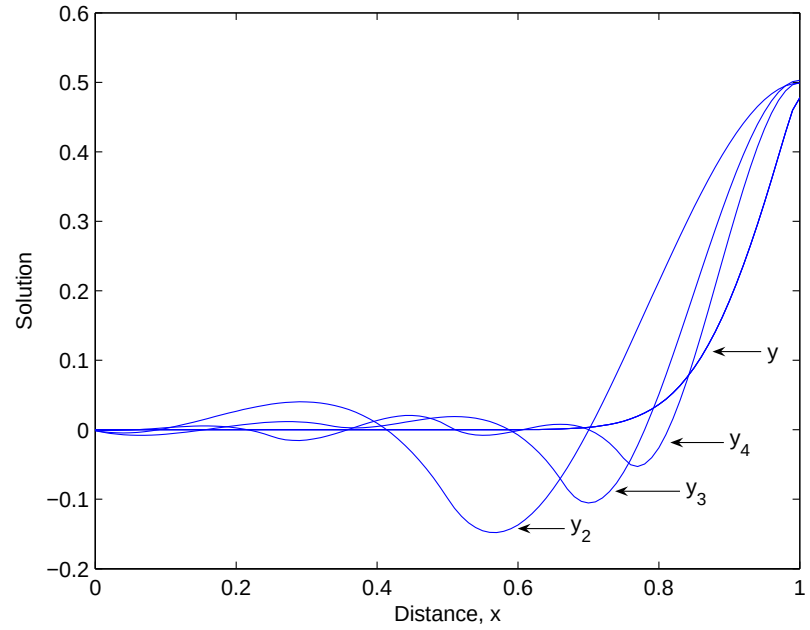

Figure 4.9: Comparison of y with y_i at $t = 1$ for $P = 40$.

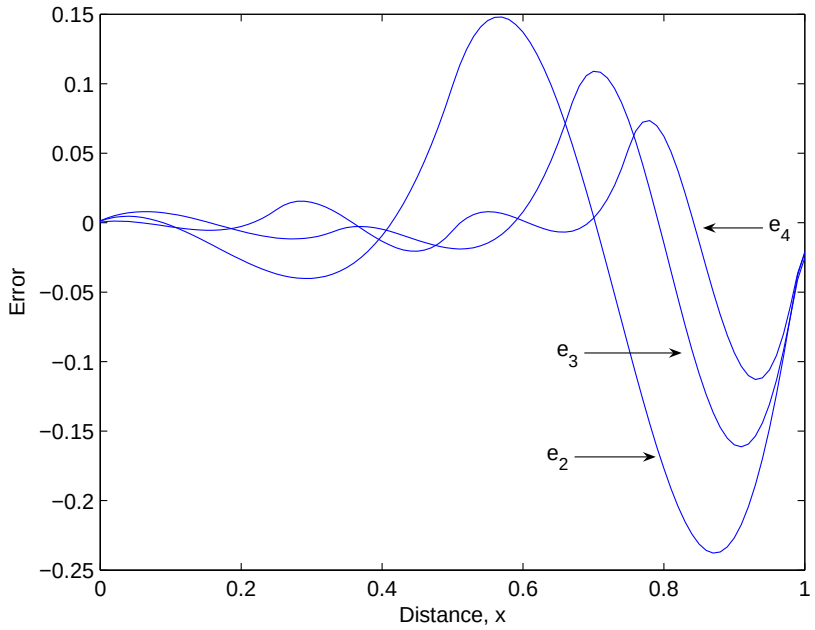

Figure 4.10: Errors $e_i = y - y_i$ at $t = 1$ for $P = 40$.

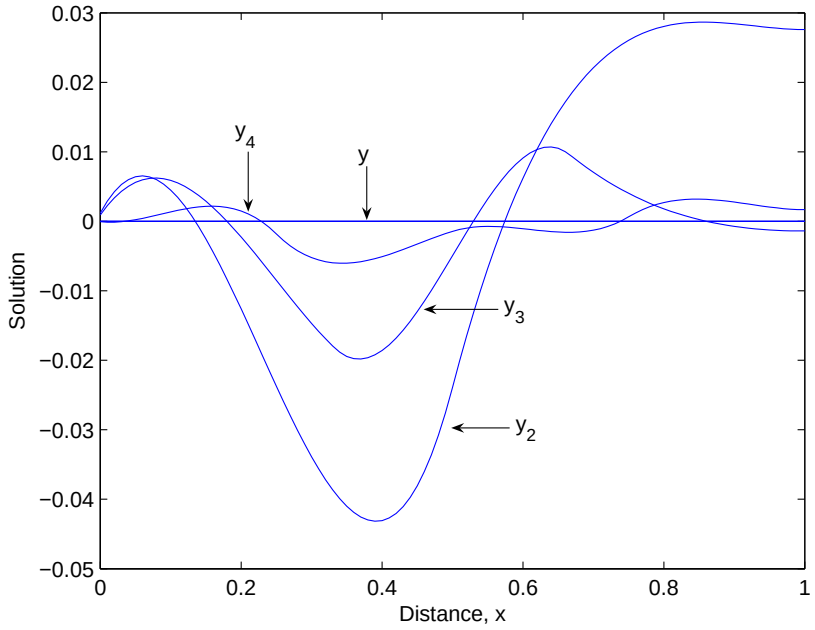

Figure 4.11: Comparison of y with y_i at $t = 2$ for $P = 40$.

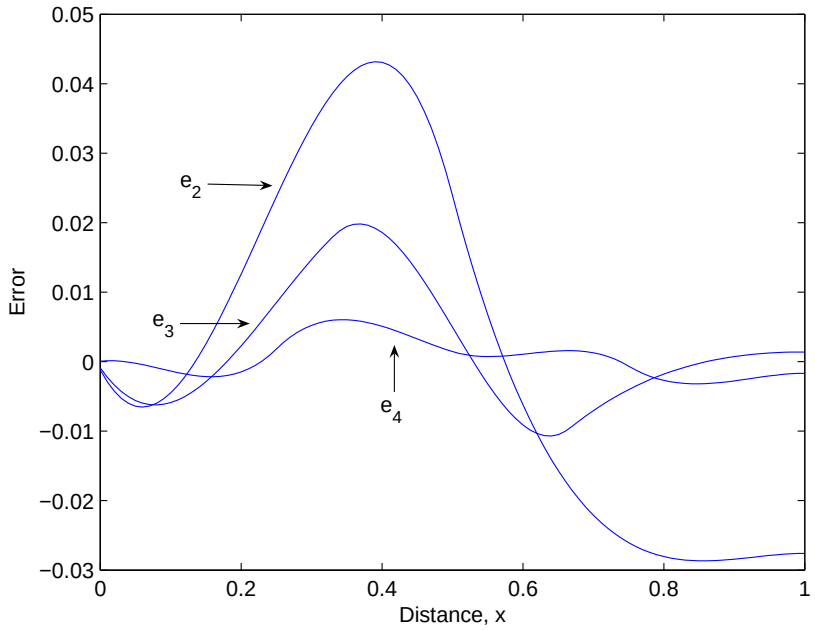

Figure 4.12: Errors $e_i = y - y_i$ at $t = 2$ for $P = 40$.

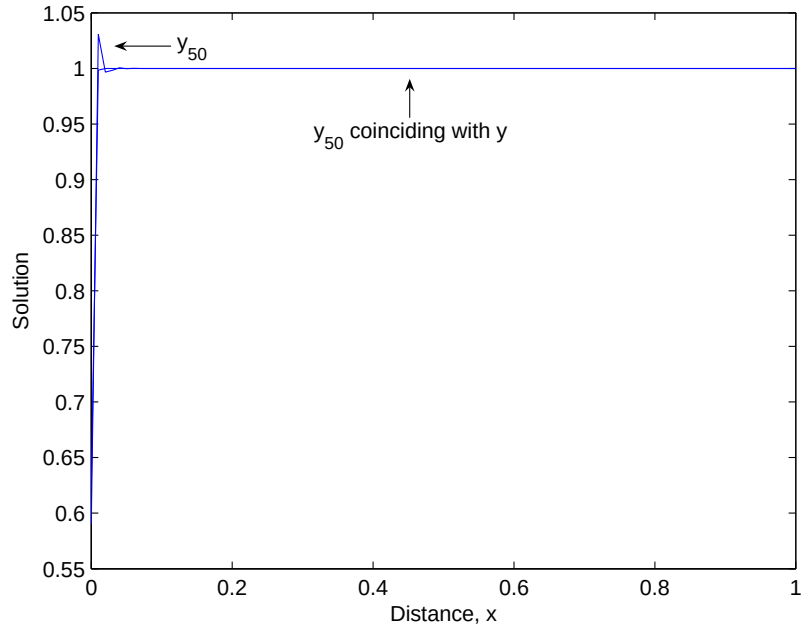

Figure 4.13: Comparison of y and y_{50} at $t = 0.001$ for $P = 40$.

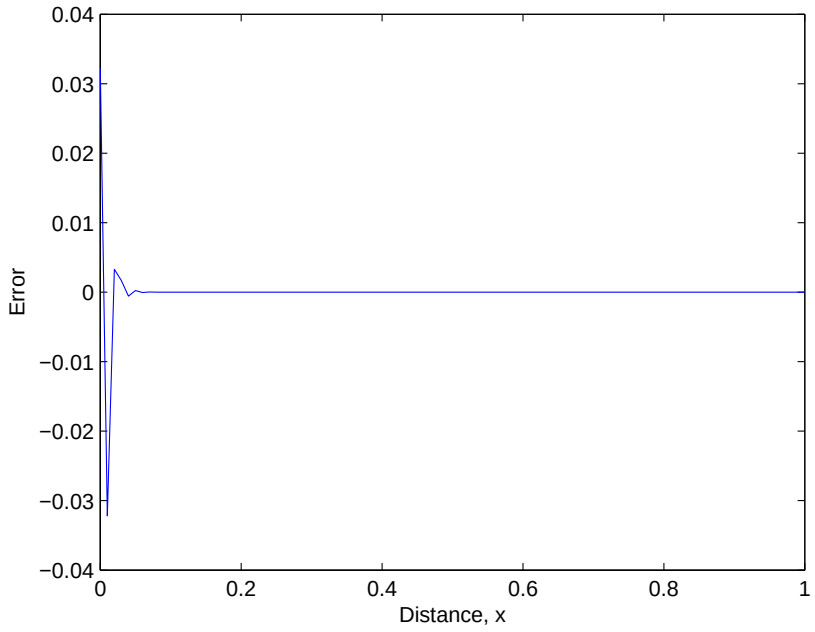

Figure 4.14: Error $y - y_{50}$ at $t = 0.001$ for $P = 40$.

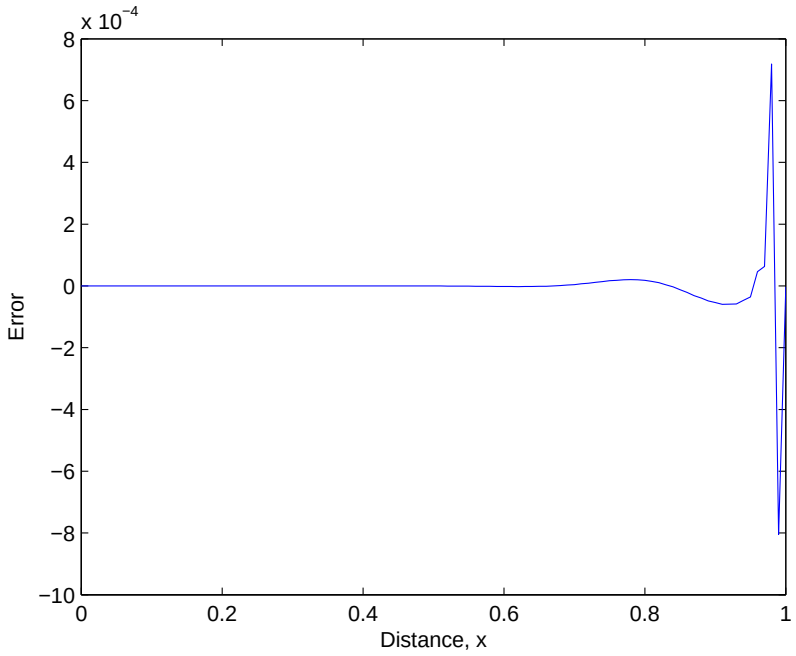

Figure 4.15: Error $y - y_{50}$ at $t = 1$ for $P = 40$.

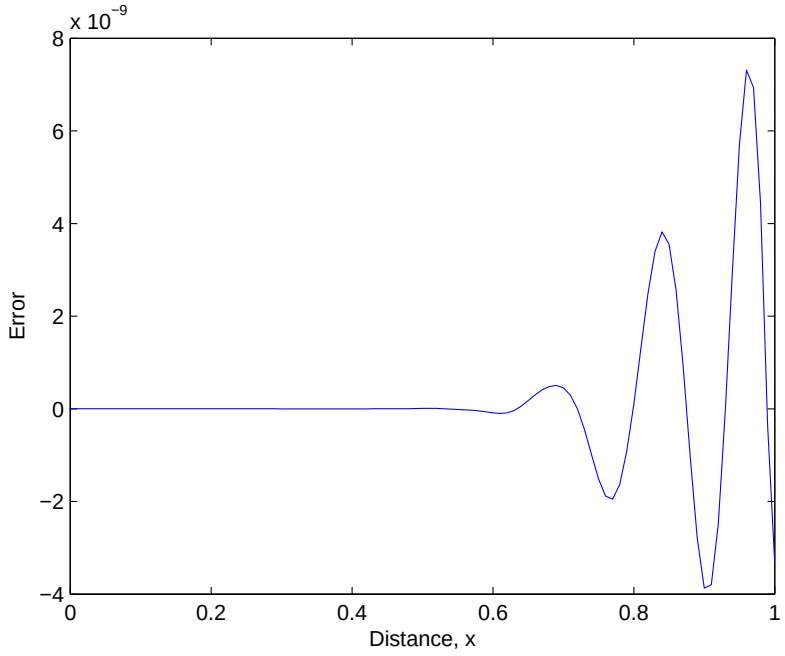

Figure 4.16: Error $y - y_{50}$ at $t = 2$ for $P = 40$.

4.2.3 Limiting Cases

It is of mathematical and chemical importance to examine the solution profile of the diffusion equation given in (4.1) for extreme values of the parameters P, x and t. We now discuss three limiting cases.

Perfect Mixing $P \to 0$

When the Peclet number approaches zero, the leading term $e^{-\lambda_1^2 t/P}$ in the infinite summation in equation (4.4) dominates the other terms because of its exponential nature. Notice from equation (4.3) that as P tends to zero ($P \to 0$), $\tan 2\lambda_1$ becomes small and using $\tan 2\lambda_1 \approx 2\lambda_1$, equation (4.3) reduces to

$$2\lambda_1 \approx \frac{2\lambda_1 P}{\lambda_1^2 - P^2}, \tag{4.18}$$

and it is obvious from equation (4.18) that as $P \to 0$, $\lambda_1 \approx \sqrt{P}$. In this limit, the exit solution given in equation (4.4) reduces to e^{-t}. Thus the solution takes the shape of an exponential graph for P small. This is depicted in Figure 4.17 for $P = 1.0 \times 10^{-5}$. This is called perfect mixing in chemistry. That is, the solvent introduced into the bed mixes with the content of the bed and an equal volume of liquid is displaced from the bed [9].

Perfect Displacement $P \to \infty$ and or $t \to 0$

Consider the asymptotic solution given in equation (4.5). As $P \to \infty$, both z_2 and $z_3 \to \infty$ and using the fact that $\operatorname{erfc}(\infty) = 0$, it is only necessary to retain the first two terms, hence

$$\lim_{P\to\infty} y(x,t) = 1 - 0.5 \lim_{P\to\infty} \operatorname{erfc}((P/t)^{0.5}(x-t)). \tag{4.19}$$

From this it is clear that the solution $y(x,t)$ depends entirely on the algebraic sign of $(x-t)$. Using the fact that $\int_{-\infty}^{\infty} e^{-y^2} dy = \sqrt{\pi}$, it is easy to show from equation

(4.6) that erfc$(-\infty) = 2$. Thus in the limit $P \to \infty$, $y(x,t)$ from equation (4.19) satisfies

$$y(x,t) = \begin{cases} 1 & \text{for } x > t, \\ 0.5 & \text{for } x = t, \\ 0 & \text{for } x < t. \end{cases} \tag{4.20}$$

In Figure 4.18 we consider $P = 1000$ (large) and fix $x = 0.4$. The solution using 100 elements y_{100} is plotted as a function of time. The behaviour predicted in equation (4.20) is obvious since for $t = x = 0.4$, the concentration is $y \approx 0.5$ and rises rapidly to 1 for $t < 0.4$ and declines sharply to 0 for $t > 0.4$.

Behavior as $t \to \infty$

As $t \to \infty$, it becomes sufficient to retain only the lead term (that is for $k = 1$) in the exit solution given in equation (4.4) since the other terms vanish rapidly owing to the exponential nature of the term $e^{-\lambda_k^2 t/P}$ which approaches zero quicker for larger λ_k's. Hence we obtain

$$ye_1(1,t) \approx e^{P(2-t)} e^{-{\lambda_1}^2 t/P} \frac{\lambda_1 \sin 2\lambda_1}{{\lambda_1}^2 + P^2 + P}, \tag{4.21}$$

Using $\sin 2\lambda_1 = \frac{2\lambda_1 P}{{\lambda_1}^2 + P^2}$, equation (4.21) can be written as

$$ye_1(1,t) \approx e^{-(P+{\lambda_1}^2/P)t} \frac{2{\lambda_1}^2 P e^{2P}}{({\lambda_1}^2 + P^2 + P)({\lambda_1}^2 + P^2)}, \tag{4.22}$$

where $\lambda_1 = 0.791034$ has been calculated (see Appendix E). The plot of the solution (4.22) and the OCFE solution for y_{100} is shown in Figure 4.19. We see a close agreement between the solutions for large t.

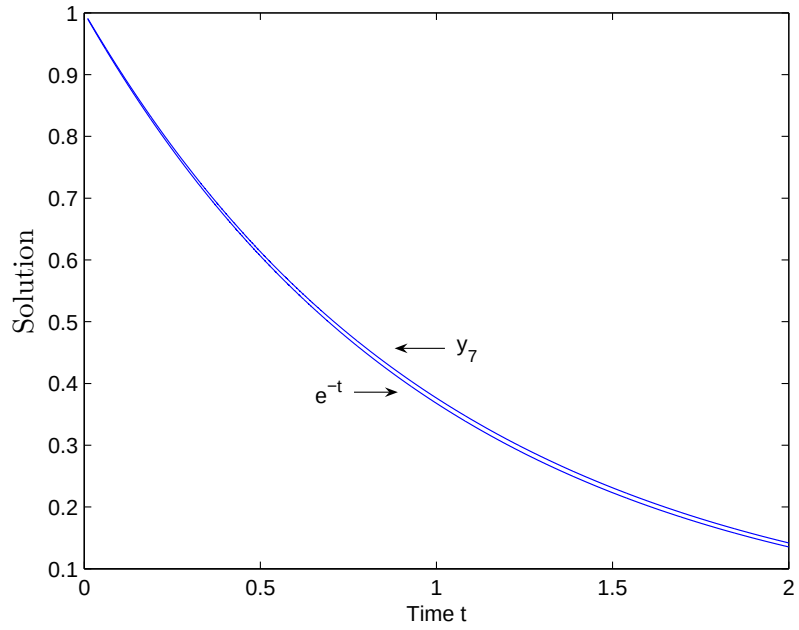

Figure 4.17: Almost perfect mixing for y_7 at $x = 1$ for $P = 1.0 \times 10^{-5}$.

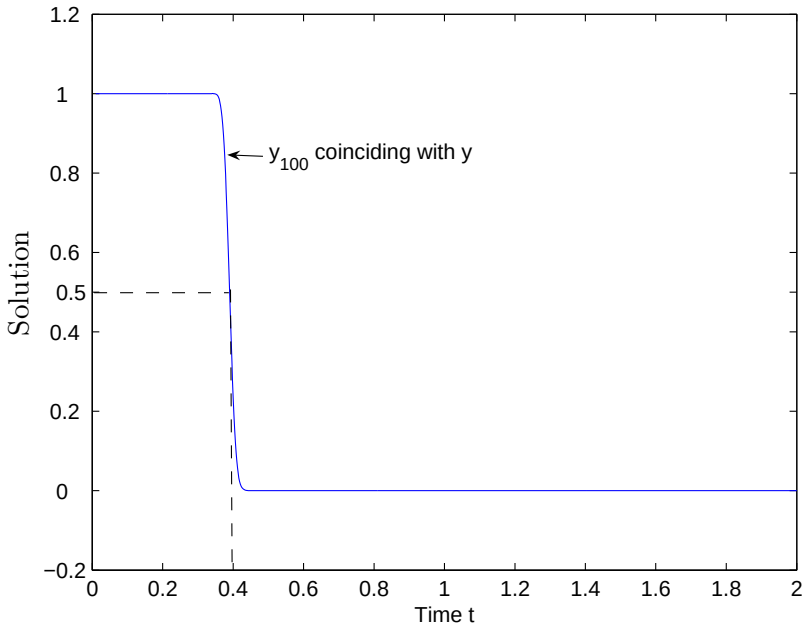

Figure 4.18: Almost perfect displacement at $x = 0.4$ for y_{100} for $P = 1000$.

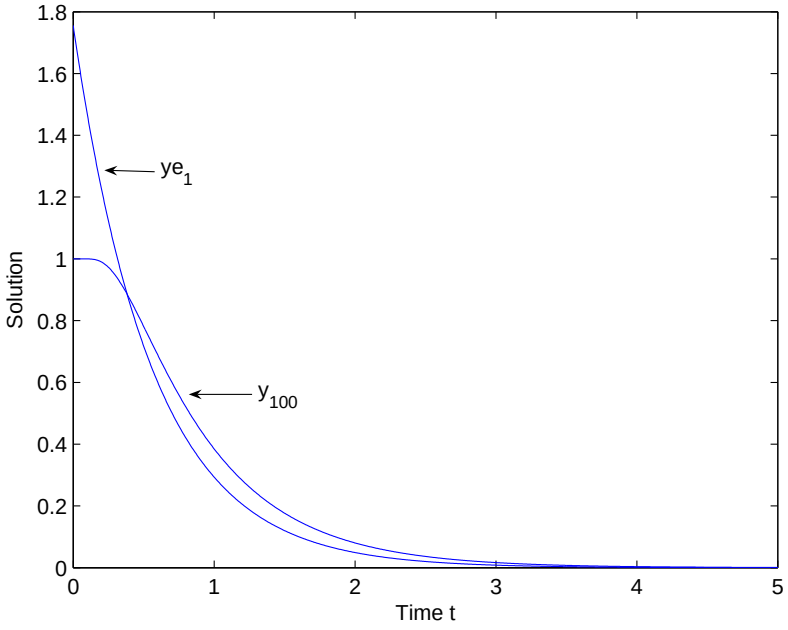

Figure 4.19: Solution for y_{100} and ye_1 from equation (4.22) for $P = 0.8$.

Chapter 5

Conclusion

5.1 Conclusion

The research done in this thesis clearly explains the orthogonal collocation method and the orthogonal collocation on finite elements method.

Prior to the work of Villadsen and Stewart [72], the choice of the collocation points was arbitrary. However, in the orthogonal collocation method, the zeros of the orthogonal polynomials are chosen to be the collocation points owing to their attractive features as discussed in section 2.2.1. Within the framework of the method of weighted residuals, the orthogonal collocation method is well known for its simplicity, since it avoids integration (see the Dirac delta property in equation (1.10)), and accuracy due to the optimal choice of the collocation points (see section 1.3.4).

In this present study, we chose a Lagrange basis which has the advantage that the approximate solution at the nodes x_k, $k = 1, 2, ..., N+1$, are just the coefficients of the basis polynomials given in (2.23). However, higher order Lagrange polynomials can be expensive to evaluate and program. One could however use an orthogonal basis which can easily be evaluated by using a three point recurrence relation of the

form given in equation (2.2). In this case one loses the advantage of the Lagrange basis.

One could also apply OCFE using a cubic Hermite polynomial basis. These basis polynomials are automatically continuous at the element boundaries and so are their first derivatives. This reduces the number of equations to be solved hence reducing the computational time.

The method of OCFE which combines the features of the orthogonal collocation method with the finite element method was found to be more numerically stable and reliable than the orthogonal collocation method especially for problems with steep gradients (see Example 3.3.1). The orthogonal collocation method provides the accuracy whereas the finite element provides the stability to the numerical results. Moreover the method of OCFE is comparable with the high-order finite difference method in terms of the CPU time and average numerical errors. The computational time is compensated for by the accuracy achieved by using more elements.

For the OCFE applied to an ODE using N_e equally spaced elements, the $4N_e \times 4N_e$ matrix $\mathbf{A}$ is sparse and only involves 16 different entries to be evaluated (see rows 2, 3 and 5 of (3.14)). The sub matrix defined by rows 2 to 5 is simply repeated, however translated to the right by an integer multiple of 4.

For the orthogonal collocation method using an order N approximation, we require to evaluate $(N-1) \times (N+1)$ entries of the sub matrix defined by rows 2 to N of $\mathbf{A}$ (see equation (2.71)). These corresponds to evaluating the residuals at $N-1$ collocation points. This is an expensive process compared to the OCFE, moreover the matrix is dense.

Most investigators have studied the exit solution (see equations (4.4) and (4.7)) to the diffusion convection equation given in (4.1). However the solution profile of the exact solution given in equation (4.2) at particular time in space has only been studied by Arora, Dhaliwal and Kukreja [4]. This thesis further expatiates on the

solution profile of equation (4.2).

Most investigators of the orthogonal collocation method [21, 36] have solved initial and boundary value problems using the zeros of the Jacobi polynomials. However, in this thesis, we have explored the orthogonal collocation method and the OCFE method with collocation points chosen as the roots of the Chebyshev polynomials of the first and second kind and the Legendre polynomials and the results are comparable with previous work [21]. In the course of this study, the Legendre polynomials were found to give good results to some problems as compared to other orthogonal polynomials used (see Examples 2.5.2). However in Example 2.5.1, the difference in the errors are negligible. Hence one could choose the collocation points as the roots of any of the orthogonal polynomials, unless in cases where investigations have shown otherwise.

In the case of the OCFE method, we restricted the trial solution to lower order polynomials (order 3 to be specific) and the results are comparable to the ones given in previous work (see [4]).

As mentioned in section 3.2.1, the placing of the elements can be a very difficult task. In this present study, we improved the solution obtained from equal element spacing by concentrating smaller unequally spaced elements in regions with steep solutions. However, one of the draw backs of this method is that one might sometimes need to think intuitively to determine the exact points to place smaller elements. Alternatively, one could concentrate elements in regions with steep gradients by noting that the essence of all weighted residual methods is to make the residual close to zero as much as possible. Hence we try to insert more elements in regions with large residuals, thus forcing the residual to zero at those points and making the solution converge faster. This implies that the residual has to be forced to zero over each element.

Carey and Finlayson used this idea and observed that the largest residual usually occurred at the end points of the elements since a continuity condition is imposed

at the nodes x_i $(i = 2, 3, ..., N_e)$, rather than setting the residual to zero. Thus they deduced that the function values at the end points of each element should be the criteria for inserting more elements rather than considering the residual on the whole domain which can be very cumbersome and expensive to implement. The function values at these end points are denoted by

$$y_{N_e}(x_i),\ \ i \in I = \{1, 2, ..., N_e + 1\}\,. \tag{5.1}$$

Let ϵ denote $1/100_{\text{th}}$ of the difference between the maximum and minimum values obtained from (5.1), that is

$$\epsilon = 0.01(\max_{i \in I}\ y_{N_e}(x_i) - \min_{i \in I}\ y_{N_e}(x_i)). \tag{5.2}$$

The absolute solution difference between consecutive end points, that is

$$|y_{N_e}(x_i) - y_{N_e}(x_{i+1})|,\ \ i = 1, 2, ..., N_e, \tag{5.3}$$

is compared with the value of ϵ obtained from equation (5.2). If

$$|y_{N_e}(x_i) - y_{N_e}(x_{i+1})| > \epsilon,\ \ i = 1, 2, ..., N_e, \tag{5.4}$$

then additional elements are inserted between x_i and x_{i+1}.

Notice that all the examples we looked at in this thesis already had known exact solutions. However in real situations, the exact solutions to differential equations might not be readily available. In such cases, we could resort to a similar approach given above for the placing of elements. We could also solve the problem by using equal element spacing h and denote the solution by y^1. We could then solve the problem using equal element spacing of $\frac{1}{2}h$ and denote the solution by y^2. We could then check if $\|y^2 - y^1\|_\infty < \epsilon$ for some predefined tolerance ϵ. If not we could continue the process by halving the element spacing until the tolerance is met. There is much investigations which has to be done in the area of placement of elements.

There are so many other interesting aspects of orthogonal collocation which we have not been able to investigate in this thesis. Research has shown that the finite

element method handles partial differential equations over complex geometries and boundaries with relative ease [17, 10, 44, 48], unlike the finite difference method which is restricted to rectangular shapes. Thus a possible area of future research is the application of the OCFE method to higher dimensional problems. However, it should be noted here that the finite difference method also gives good results to some practical problems like computational fluid dynamics. One could also investigate the application of OCFE to non-linear PDE's as done by Ma and Guiochon [38] and also the spline collocation method as done by several investigators like Juha Anttila [33], Costabel, Saranen [15] and Greenwell-Yanik and Fairweather [29]. Spectral collocation is another interesting area which can be used to solve singular PDE's (see the work done by Huang and Sun [31]). Collocation using wavelet basis is another new area of research.

References

[1] Adomaitis R.A. and Lin Y., A Technique for Accurate Collocation Residual Calculations, Chem. Eng. J. **71** (1998), 127 – 134.

[2] Ames W.F. (1965), Nonlinear Partial Differential Equation in Engineering, Academic Press New York.

[3] Arora S., Dhaliwal S.S. and Kukreja V.K. , Application of Orthogonal Collocation on Finite Elements for Solving Non-Linear Boundary Problems, Science Direct Appl. Math. Comput. **180** (2006), 516 – 523.

[4] Arora S., Dhaliwal S.S. and Kukreja V.K., Solution of Two Point Boundary Value Problems Using Orthogonal Collocation on Finite Elements, Science Direct Appl. Math. Comput. **171**(1) (2005), 358 – 370.

[5] Bartlett E.P. and Kendall R.M. (1968), R.M.NASA CR-1062.

[6] Becker E.B., Carey G.F. and Oden J.T. (1981), Finite Elements- An Introduction, Prentice-Hall Englewood Cliffs.

[7] Bickley W.G., Phil. Mag. **32**(7) (1941), 50 – 66.

[8] Biezeno C.B. and Koch J.J., Ingenieur **38** (1923), 25 – 36.

[9] Brenner H., The Diffusion Model of Longitudinal Mixing in Bed of Finite Length, Numerical Values, Science Direct, Chem. Eng. Sci. **17** (1962), 229 – 243.

[10] Brenner S. and Scott R. (1994), Variational and Computational Methods, CRC Press, New York. The Mathematical Theory of Finite Element Methods Springer-Verlag.

[11] Carey G.F. and Finlayson B.A., Orthogonal Collocation on Finite Elements, Science Direct Chem. Eng. Sci. **30** (1975), $587 - 596$.

[12] Carslaw H.S. and Jaeger J.C. (1959), Conduction of Heat in Solids 2nd ed., $114 - 120$, $491 - 492$, $494 - 496$, Oxford University Press.

[13] Clough R.W., Thoughts About the Origin of the Finite Element Method, Science Direct Finite Elem. Anal. Desig. **79** (2001), $2029 - 2030$.

[14] Clough R.W., Original Formulation of the Finite Elements, Science direct Finite Elem. Anal. Desig. **7** (1990), $89 - 101$.

[15] Costabel M. and Saranen J., Spline Collocation for Convolutional Parabolic Boundary Integral Equations, ACM Numer. Math. **84** (2000), $417 - 449$.

[16] Crandall S.H. (1956), Engineering Analysis, McGraw-Hill New York.

[17] Daryl L. Logan (2006), A First Course in the Finite Element Method, $4th$ ed. Thomson Learning UK.

[18] Duncan W.J. (1938), ARC R and M 1848.

[19] Duncan W.J. (1937), ARC R and M 1798.

[20] Fan L.T., Chen G.K.C. and Erickson L.E., Efficiency and Utility of the Collocation Methods in Solving the Performance Equations Flow Chemical Reactors with Axial Dispersion, Science Direct Chem. Eng. Sci. **26** (1971), $379 - 387$.

[21] Finlayson B.A. (1980), Nonlinear Analysis in Chemical Engineering, McGraw-Hill Inc. United States of America Chemical Engineering.

[22] Finlayson B.A. (1972), The Method of Weighted Residuals and Variational Principles, Academic Press New York.

[23] Finlayson B.A. and Scriven L.E., App. Mech. Rev. **19** (1966), $735 - 748$.

[24] Frazer R.A., Jones W.P. and Skan S.W. (1937), ARC R and M 1799, Springer-Verlag New York Inc.

[25] Fletcher C.A.J. (1984), Computational Galerkin Method, Springer-Verlag New York Inc.

[26] Galerkin B.G., Vestnik Inzhenerov, Tech. **19** (1915), $897 - 908$.

[27] Grahs L.E. (1974), Washing of Cellulose Fibres, Analysis of Displacement Washing Operation, PhD Thesis, Chalmers University of Technology, Goteborg, Sweden.

[28] Grandin H. (1991), Fundamentals of the Finite Element Method, Waveland Press, Waveland.

[29] Greenwell-Yanik C.E. and Fairweather G., Analysis of Spline Collocation Methods for Parabolic and Hyperbolic Problems in Two Space Variables, SIAM Journal on Numerical Analysis **23**(2) (1986), $282 - 296$.

[30] Hamming R.W., Numerical Methods for Scientist and Engineers. McGraw-Hill, New York 2nd ed. Dover reprint 1985; translated into Russian **41**(6) (1973), $2333 - 2349$.

[31] Huang W., Ma H. and Sun W., Convergence Analysis of Spectral Collocation Methods for a Singular Differential Equation, SIAM J. Appl. Numer. Math. **41**(6) (2003), $2333 - 2349$.

[32] Huang W.H. and Russell R.D., A Moving Collocation Method for Solving Time Dependent Partial Differential Equation, SIAM J.Appl. Numer. Math. **20** (1996), $101 - 116$.

[33] Juha Anttila, A Spline Collocation Method for Parabolic Pseudo Differential Equations, ACM Comput. and Appl. Math. **140** (2002), 41 – 61.

[34] Khalid Alhumaizi, A Moving Collocation Method for the Solution of the Transient Convection-Diffusion-Reaction Problems, Comput. and Appl. Mathematics **193** (2006), 484 – 496.

[35] Kythe P. K. K. and Wei D. (2003), An Introduction to Linear and Nonlinear Finite Element Analysis, Birkhauser Verlag Basel.

[36] Lefervre L., Dochain D., Feyo de Azevedo S. and Magnus A., Optimal Selection of Orthogonal Polynomials Applied to the Integration of Chemical Reactors, Science Direct Comput. Chem. Eng. **24** (2000), 2571 – 2588 .

[37] Liu F. and Bhatia S.K. , Computational Efficient Solution Techniques for Adsorption Problems Involving Steep Gradients Bidisperse Particles, Science Direct Comput. Chem. Eng. **23** (1999), 933 – 943.

[38] Ma Z. and Guiochon G., Application of Orthogonal Collocation on Finite Elements in the Simulation of Non-Linear Chromatography, Science Direct Comput. Chem. Eng. Sci. **15**(6) (1991), 415 – 426.

[39] Martin Avery Snyder (1966), Chebyshev Methods in Numerical Approximation, Prentice-Hall Englewood Cliffs, N.J.

[40] Mathews J.H. and Kurtis D. Fink (2004), Numerical Methods Using Matlab, Person Education, Inc, New Jersey.

[41] Michelsen M.L. and Villadsen J. (1980), Polynomial Solution of Differential Equations, Proceedings of the First Conference on Computer Aided Process Design Held at New Hampshire, July 6 – 11, 1980, Prentice Hall, Englewood Cliffs.

[42] Michelsen M.L. and Villadsen J. (1978), Solutions of Differential Equation Models by polynomial Approximation, Prentice Hall, Engelwood Cliffs, New Jersey.

[43] Michelsen M.L. and Villadsen J., A Convenient Computational Procedure for Collocation Constants, Science Direct Chem. Eng. Sci. **4** (1971), $64 - 68$.

[44] Morton K.W. (1991), Lecture Notes on Finite Element Methods, Oxford University Computing Laboratory.

[45] Moustafa Soliman A., Khalid Alhumaizi (2004), An Efficient Collocation Method for Diffusion-Convection Problems with Chemical Reactors, King Saud University Press, Saudi Arabia.

[46] Murphy J.D., AIAA J. **15** (1977), $1307 - 1314$.

[47] Murray R. Spiegel (1971), Advanced Mathematics for Engineers and Scientists, McGraw-Hill Inc, Great Britain.

[48] Norrie D. H. and De Vries G. (1978), An Introduction to Finite Element Analysis, Academic Press, New York-London.

[49] Onah S.E, Asymptotic Behavior of the Galerkin and the Finite Element Collocation Methods for a Parabolic Equation, Science Direct Appl. Math. Comput. **127** (2002), $207 - 213$.

[50] Owen D. R. J. and Hinton, E. (1980), A Simple Guide to Finite Element, Pineridge Press, Swansea.

[51] Pallone A. J., Aero. Sci. **28** (1961), $449 - 456$.

[52] Panton R. Z. and Salle H. B., Computers and Fluids **3** (1975), $257 - 269$.

[53] Paterson W.R. and Cresswell D.L., A Simple Method for the Calculation of Effectiveness Factor, Chem. Eng. Sci. **26** (1971), $605 - 616$.

[54] Prenter P.M. and R. D. Russell, Orthogonal Collocation for Elliptic Partial Differential Equations, SIAM Numerical Analysis **13**(6) (1976), $923 - 939$.

[55] Raghavan N.S. and Ruthven D.M., Numerical Simulation of the Fixed Bed Adsorption Column by the Method of Orthogonal Collocation, Science Direct **29**(6) (1983), 922 – 925 .

[56] Ramachandran P.A. and Dudukovic M.P., A Moving Finite Element Collocation Method for Transient Problems with Steep Gradients, Chem. Eng. Sci. **39**(7/8) (1984), 1321 – 1324.

[57] Rao C.R., Toutenburg H., Fieger A., Heumann C., Nittner T. and Scheid S. (1999), Linear Models: Least Squares and Alternatives, Springer Series in Statistics.

[58] Reddy, J. N. (2005), An Introduction to the Finite Element Method, McGraw-Hill Book Co. New York.

[59] Richard G. Rice and Duong D. Do (1995), Applied Mathematics and Modeling for Chemical Engineers, John Wiley and sons Inc. Canada.

[60] Richard L. Burden, J. Douglas Faires (2001), Numerical Analysis (7th ed.), Wadsworth Group. Books, Youngstown State University.

[61] Ruthven D.M. (1984), Principles of Adsorption and Adsorption Process, Wiley-Inter Science Publication, New York.

[62] Seferlis P., Hrymak A.N., Adaptive Collocation on Finite Elements Model for the Optimization of of Multistage Distillation Units, Science Direct Chem. Eng. Sci. **49**(9) (1994), 1369 – 1382.

[63] Schetz J.A., J. Appl. Mech. **30** (1963), 263 – 268.

[64] Shampil L.F., Gladwell L., Thompson S. (2003), Solving ODEs with MATLAB, Cambridge University Press.

[65] Shirashi F., Highly Accurate Solution of the Axial Dispersion Model Expressed in S-system Canonical Form by Taylor Series Method, Chem. Eng. J. **83** (2003), 175 – 183.

[66] Smith G. N. (1971), An Introduction to Matrix and Finite Element Methods in Civil Engineering, Applied Science Publ. London.

[67] Soliman M.A. (2004), The Method of Orthogonal Collocation, King Saud University Press, Saudi Arabia.

[68] Sridhar P., Implementation of the One Point Collocation Method to an Affinity Packed Bed Model, Indian Chem. Eng. Sec. **41**(1) (1999), $39 - 46$.

[69] Stewart W.E, Levien K.L. and Stewart M., Simulation of Fractionation by Orthogonal Collocation Method, Science Direct Chem. Eng. Sci. **40**(3) (1985), $409 - 421$.

[70] Thamban Naira M. and Sergei V. Pereverzevb, Regularized Collocation Method for Fredholm Integral Equations of the First Kind, Journal of Complexity **23**$(4 - 6)$ (2007), $454 - 467$.

[71] Vichnevetsky R., IEEE Trans. Comp. $\mathbf{C} - \mathbf{18}$ (1969), $499 - 512$.

[72] Villadsen J. and Stewart W.E., Solution of Boundary Value Problems by Orthogonal Collocation, Science Direct Chem. Eng. Sci. **22** (1967), $1483 - 1501$.

[73] Villadsen J. and Sorensen J.P., Solution of of Parabolic Partial Differential Equations by a Double Collocation Method, Science Direct Chem. Eng. Sci. **24** (1969), $1337 - 1349$.

[74] Viviand H. and Ghazzi W., La Recherche Aerospatiale $\mathbf{1974 - 5}$ (1974), 5, $247 - 260$.

[75] Walter Wunderlich and Walter D.Pilkey (2002), Mechanics of Structures, Variational and Computational Methods, 2nd ed.

[76] Wolberg J. (2005), Data Analysis Using the Method of Least Squares: Extracting the Most Information from Experiments, Springer.

[77] Yamada H., Rept. Res. Inst. Fluid Eng. Kyushu Univ. **4** (1948), 27 − 42.

Appendix A

Programme for Example 2.5.1

```
% Specifying the order, m of the polynomial
clear all
for m = 3:1:6
n=m-1;
nn=m+1;
h=0.001;
v=1;
v2=v*v;
p=10;
j=n:-1:1;
% Commnent the one that is not applicable
% Shifted roots of T_m-1
xin(n+1-j) = 0.5*(cos((2*j-1)*pi/(2*n))+1)
% Shifted roots of U_m-1
 col(n+1-j)= 0.5*(cos((j*pi)/(n+1))+1)
% Shifted roots of P_m-1
% Replace all fact with factorial
xi=zeros(n+1,1);
if mod(n,2)==0;
    for P=0:2:n;
```

```
        np1=(n-P)/2;
        np2=(n+P)/2;
        np3=n+P;
        xi(n+1-P)=(-1)^np1*fact(np3)/(fact(np2)*fact(np1)*fact(P));
    end
else
    for P=1:2:n;
        np1=(n-P)/2;
        np2=(n+P)/2;
        np3=n+P;
        xi(n+1-P)=(-1)^np1*fact(np3)/(fact(np2)*fact(np1)*fact(P));
   end
end
xii=xi/max(xi);
y=roots(xii);
col=sort(y)';
col=(col+1)/2;
% Collocation points
col=[0 col 1];
% Interpolation points
xin=[0 xin 1];
x= xin;
% Boundary conditions
y1=0;ynn=1;
% Divisors of the Lagrange polynomials
for j=1:nn;
    prod=1.0;
    for k=1:nn;
    if(k~=j);
        prod=prod*(xin(j)-xin(k));
    else prod=prod*1;
    end
```

```
    end
    d(j)=prod;
end
h=0.001;
% Evaluating the Lagrange polynomials at x
for j=1:nn;
    prod=1.0;
    for k=1:nn;
    if(k~=j);
        prod=prod.*(x-xin(k));
    else prod=prod.*1;
    end
    end
    l(:,j)=prod/d(j);
end
l
xh=x+h;
% Evaluating the derivatives of the Lagrange polynomials
for j=1:nn;
    prod=1.0;
    for k=1:nn;
    if(k~=j);
        prod=prod.*(xh-xin(k));
    else prod=prod.*1;
    end
    end
    lh(:,j)=prod/d(j);
end
xhh=x-h;
for j=1:nn;
    prod=1.0;
    for k=1:nn;
```

```
    if(k~=j);
        prod=prod.*(xhh-xin(k));
    else prod=prod.*1;
    end
    end
    lhh(:,j)=prod/d(j);
end

dl=(lh-lhh)/(2*h);
ddl=(lh+lhh-2*l)/(h*h);
% Assigning the matrix elements
for i=2:m;
%  for i= 1:nn;
    for j=1:nn;
        a(i,j)=ddl(i,j)+dl(i,j)-p^2*l(i,j);
    end
    b(i)=0;
end
b(1)=y1;
b(nn)=ynn;
b=b';
for j=1:nn;
    a(1,j)=dl(1,j);
    a(nn,j)=l(nn,j);
end
c=a\b;
% Evaluating the approximate solution
f=101;
t=linspace(0,v,f)';
size(t);
tp=2*t/v-1;
for j=1:nn;
```

```
    prod=1;
    for k=1:nn;
    if(k~=j);
        prod=prod.*(t-xin(k));
    else prod=prod.*1;
    end
    end
    lp(:,j)=prod/d(j);
end
yaprox=lp*c;m;
% Exact solution
r=sqrt(4.*p.^2+1)./2;
yexact = (exp(1./2-t./2).*(2.*r.*cosh(r.*t)+...
sinh(r.*t))).//((2.*r.*cosh(r))+sinh(r));
% Plotting the solutions
plot(t,yexact);
hold on
plot(t,yaprox);
xlabel('Distance, x')
ylabel('Solution')
% Plotting the errors
% plot(t,(y - yapp));
         clear all;
hold on
end
```

Appendix B

Programme for Example 2.5.2

```
% Specifying the order, m of the polynomial
 clear all
for m = 4:2:10;
n=m-1;
nn=m+1;
h=0.001;
v=3.5;
r=101;
v2=v*v;
j=n:-1:1;
% Comment the one that is not applicable
% Roots of T_m-1
xin(n+1-j)= cos((2*j-1)*pi/(2*n));
% Roots of the U_m-1
col(n+1-j)= cos((j*pi)/(n+1));
% Roots of P_m-1
% Replace all fact with factorial
xi=zeros(n+1,1);
if mod(n,2)==0;
    for p=0:2:n;
```

```
        np1=(n-p)/2;
        np2=(n+p)/2;
        np3=n+p;
        xi(n+1-p)=(-1)^np1*fact(np3)/(fact(np2)*fact(np1)*fact(p));
    end
else
    for p=1:2:n;
        np1=(n-p)/2;
        np2=(n+p)/2;
        np3=n+p;
        xi(n+1-p)=(-1)^np1*fact(np3)/(fact(np2)*fact(np1)*fact(p));
   end
end
xi=xi/max(xi);
y=roots(xi);
col=sort(y)';
col=[-1 col 1];
% Interpolation points
xin =[-1 xin 1];
% The collocation points
x = col;
% Boundary conditions
y1=0;ynn =(0.5*v2+v)*exp(-3*v);
% Defining the divisors of Lagrange polynomials
for j=1:nn;
    prod=1.0;
    for k=1:nn;
    if(k~=j);
        prod=prod*(xin(j)-xin(k));
    else prod=prod*1;
    end
    end
```

```
    d(j)=prod;
end
h=0.001;
% Evaluating the Lagrange polynomials at x
for j=1:nn;
    prod=1.0;
    for k=1:nn;
    if(k~=j);
        prod=prod.*(x-xin(k));
    else prod=prod.*1;
    end
    end
    l(:,j)=prod/d(j);
end
xh=x+h;
% Evaluating the derivatives of the Lagrange polynomials
for j=1:nn;
    prod=1.0;
    for k=1:nn;
    if(k~=j);
        prod=prod.*(xh-xin(k));
    else prod=prod.*1;
    end
    end
    lh(:,j)=prod/d(j);
end
xhh=x-h;
for j=1:nn;
    prod=1.0;
    for k=1:nn;
    if(k~=j);
        prod=prod.*(xhh-xin(k));
```

```
    else prod=prod.*1;
    end
    end
    lhh(:,j)=prod/d(j);
end
dl=(lh-lhh)/(2*h);
ddl=(lh+lhh-2*l)/(h*h);
% Assigning the matrix elements
for i=2:m;
    for j=1:nn;
        a(i,j)=4*ddl(i,j)/v2+2*6*dl(i,j)/v+9*l(i,j);
    end
    b(i)=exp(-3*0.5*v*(x(i)+1));
end
for j=1:nn;
    a(1,j)=l(1,j);
    a(nn,j)=l(nn,j);
end
b(1)= y1;
b(nn)= ynn;
b=b';
c=a\b;

% Evaluating the approximate solution
t=linspace(0,v,r)';
tp=2*t/v-1;
for j=1:nn;
    prod=1;
    for k=1:nn;
    if(k~=j);
        prod=prod.*(tp-xin(k));
    else prod=prod.*1;
```

```
    end
    end
    lp(:,j)=prod/d(j);
end
% Exact solution
yexact = t.*exp(-3.*t)+0.5.*t.^2.*exp(-3.*t);
yaprox=lp*c;m;
% Plotting the solutions
plot(t,yexact,'linewidth', 2);
hold on
plot(t,yaprox,'linewidth', 2);
xlabel('Distance, x')
ylabel('Solution')
% Ploting the error
%   plot(t,(yy-yaprx))
         clear all;
hold on
end
```

Appendix C

Programme for Example 3.3.1

```
clear all
% Specify the number of elements
for ne = 18:3:27;
% Initilizing the matrices
a = zeros(4*ne,4*ne);b = zeros(4*ne,1);
% Step size for the Lagrange function
h = 0.001; h1=2*h;h2=h^2;
r =  101;
% Original variable
t = linspace(0,3.5,r);
% Transformed variable
tt = (2*t/3.5)-1;
x = linspace(-1,1,ne+1);
% Step size of elements
for i = 1:ne;
H(i) = (x(i+1)-x(i));
end
% Constants
aa = 16/(3.5)^2; bb = 24/3.5; cc = 3.5/2;
% Interpolation points with internal points chosen as roots of T_2
```

```
u1=-1;u2=-1/sqrt(2);u3=1/sqrt(2);u4 = 1;u = [u1 u2 u3 u4];
% Collocation points with internal points chosen as roots of P_2
v1=-1;v2=-1/sqrt(3);v3=1/sqrt(3);v4=1;v=[v1 v2 v3 v4];
% Defining the divisors of the Lagrange function
for i = 1:4;
    prod = 1.0;
    for j = 1:4;
    if (i~=j)prod=prod*(u(i)-u(j));
    else prod = prod*1;
    end
    end
    d(i) = prod;
end

% The Lagrange functions
l1 = @(t)(t-u2).*(t-u3).*(t-u4)/d(1);
l2 = @(t)(t-u1).*(t-u3).*(t-u4)/d(2);
l3 = @(t)(t-u1).*(t-u2).*(t-u4)/d(3);
l4 = @(t)(t-u1).*(t-u2).*(t-u3)/d(4);

% The first derivative of the Lagrange functions
l1d = @(t)(l1(t+h)-l1(t-h))/(h1);
l2d = @(t)(l2(t+h)-l2(t-h))/(h1);
l3d = @(t)(l3(t+h)-l3(t-h))/(h1);
l4d = @(t)(l4(t+h)-l4(t-h))/(h1);

% The second derivatives of the Lagrange functions
l1dd = @(t)(l1(t+h)+l1(t-h)-2*l1(t))/(h2);
l2dd = @(t)(l2(t+h)+l2(t-h)-2*l2(t))/(h2);
l3dd = @(t)(l3(t+h)+l3(t-h)-2*l3(t))/(h2);
l4dd = @(t)(l4(t+h)+l4(t-h)-2*l4(t))/(h2);
```

```
% Cell array for the Lagrange functions and their derivatives
l={l1 l2 l3 l4};ld={l1d l2d l3d l4d};ldd={l1dd l2dd l3dd l4dd};
l=repmat(l,1,4*ne);ld=repmat(ld,1,4*ne);ldd=repmat(ldd,1,4*ne);

% Assigning matrix elements
% Residues
for s = 1:ne;
    for j = 1:4;
         a(2*s,4*s-4+j)=aa/(H(s))^2*ldd{j}(v2)+ bb/H(s)*ld{j}(v2)+9*l{j}(v2);
         a(2*s+1,4*s-4+j)=aa/(H(s))^2*ldd{j}(v3)+bb/H(s)*ld{j}(v3)+9*l{j}(v3);
    end
end
% The boundary conditions
for s = 1:ne-1;
        for j=1:4;
    a(1,j) = l{j}(u1);
    a(4*ne,4*ne-4+j) = l{j}(u4);
        end
end
% Continuity of the elements
 for s = 1:ne-1;
    for j = 1:4;
           a(2*ne+s+1,4*s-4+j) = l{j}(u4);
           a(2*ne+s+1,4*s+j)= -(l{j}(u1));
      end
 end
% Continuity of the derivatives
   for s = 1:ne-1;
      for j=1:4;
     a(2*ne+ne+s,4*s-4+j) = ld{j}(u4)/H(s);
     a(2*ne+ne+s,4*s+j) = -(ld{j}(u1))/H(s+1);
        end
```

```
   end
% Defining the b's
b(4*ne) = 9.625*exp(-10.5);
 for s = 1:ne;
b(2*s) = exp(-5.25*(H(s)/2*(v2+1)+ x(s)+1));
b(2*s+1) = exp(-5.25*(H(s)/2*(v3+1)+ x(s)+1)) ;
 end
% Determining the coefficients
c = inv(a)*b
 for s = 1:4;i = 1:ne;
        C(i,s)=c(4*(i-1)+s);
     end
% The approximate solution
     for i = 1:ne;
         for k = 1:r;
           uu =((2*(tt(k)-x(i)))/(x(i+1)-x(i)))-1;
           if(x(i)<=tt(k) & tt(k)<=x(i+1))j=i;
yaprx(k) = C(i,1)*l1(uu) + C(i,2)*l2(uu)+ C(i,3)*l3(uu)+ C(i,4)*l4(uu);

 end
         end
         end
% Exact solution
y = t.*exp(-3.*t)+0.5*t.^2.*exp(-3.*t);
% Plotting the solutions
plot(t,y);
hold on
plot(t,yaprx);
xlabel('Distance, x')
ylabel('Solutions')
% Total error
[sqrt(sum((y-yaprx).^2))]
```

```
clear all;

hold on

end
```

Appendix D

Programme for Example 4.2.1

```
clear all
% Enter the number of elements
for ne= 2:1:5;
% Initializing the matrices
M = zeros(4*ne,4*ne);
J = zeros(4*ne,4*ne)';
% Peclet number
p =  40;
t1 = 2 ;
t2 = 1;
r = 101;
% Step size for the Lagrange function
h = 0.001;
x = linspace(0,1,ne+1);
%step size of elements
H = 1/ne;
% The interpolation points chosen as the roots of T_n-1
u1 = 0; u2 = (-1/sqrt(2)+1)/2 ;u3 = (1/sqrt(2)+1)/2;u4 = 1; u = [u1 u2 u3 u4];
% The collocation points chosen as the roots of P_n-1
v1 = 0; v2 = (-1/sqrt(3)+1)/2 ;v3 = (1/sqrt(3)+1)/2;v4 = 1; v = [v1 v2 v3 v4];
```

```
% The divisors of the Lagrange function
for i = 1:4;
    prod = 1.0;
    for j = 1:4;
    if (i~=j)prod = prod*(u(i)-u(j));
    else prod=prod*1;
    end
    end
    d(i)=prod;
end

% The lagrange functions
l1 = @(t)(t-u2).*(t-u3).*(t-u4)/d(1);
l2 = @(t)(t-u1).*(t-u3).*(t-u4)/d(2);
l3 = @(t)(t-u1).*(t-u2).*(t-u4)/d(3);
l4 = @(t)(t-u1).*(t-u2).*(t-u3)/d(4);

% The first derivative of the Lagrange functions
l1d = @(t)(l1(t+h)-l1(t-h))/(2*h);
l2d = @(t)(l2(t+h)-l2(t-h))/(2*h);
l3d = @(t)(l3(t+h)-l3(t-h))/(2*h);
l4d = @(t)(l4(t+h)-l4(t-h))/(2*h);

% The second derivatives of the Lagrange functions
l1dd = @(t)(l1(t+h)+l1(t-h)-2*l1(t))/(h^2);
l2dd = @(t)(l2(t+h)+l2(t-h)-2*l2(t))/(h^2);
l3dd = @(t)(l3(t+h)+l3(t-h)-2*l3(t))/(h^2);
l4dd = @(t)(l4(t+h)+l4(t-h)-2*l4(t))/(h^2);

% Cell array for the Lagrange functions and their derivatives
l = {l1 l2 l3 l4};ld = {l1d l2d l3d l4d};ldd = {l1dd l2dd l3dd l4dd};
l = repmat(l,1,4*ne);ld = repmat(ld,1,4*ne);ldd = repmat(ldd,1,4*ne);
```

```
% Assigning matrix elements
% Defining the mass matrix for the DAE
for s = 1:ne;
    for j = (4*s)-3:4*s;
         M((2*s),j) = l{j}(v2);
         M(2*s+1,j) = l{j}(v3);
    end
end
% Defining the Jacobian matrix
 % The residual entries
for s=1:ne;
    for j=1:4;
         J(2*s,4*s-4+j) = (ldd{j}(v2)/(H^2*4*p)) -(ld{j}(v2)/H);
         J(2*s+1,4*s-4+j) =(ldd{j}(v3)/(H^2*4*p)) -(ld{j}(v3)/H);
    end
end

% Boundary conditions
% The left bc and right bc
    for s = 1:ne-1;
        for j = 1:4;
    J(1,j) = -l{j}(u1)+ (ld{j}(u1))/(4*p*H);
    J(4*ne,4*ne-4+j) = -(ld{j}(u4))/H;
    end
    end
% Continuity of the elements
for s=1:ne-1;
    for j=1:4;
   J(2*ne+s+1,4*s-4+j) = -l{j}(u4);
  J(2*ne+s+1,4*s+j) = l{j}(u1);
end
```

```
end
 % Continuity of the derivatives
 for s=1:ne-1;
      for j=1:4;
   J(2*ne+ne+s,4*s-4+j) = -ld{j}(u4)/H;
   J(2*ne+ne+s,4*s+j)= ld{j}(u1)/H;
 end
 end
% Solving the DAE
tspan =linspace(0.01,t1,r);tspan = tspan';
c0 = ones(4*ne,1);
options = odeset('Mass',M,'Jacobian',J);
f = @(t,c) J*c;
sol = ode15s(f,[0 t1],c0,options);
% Uncommnet this to obtain the exit solution at x = 1
% yaprox = deval(sol,[0, 2],4*ne)
c = deval(sol,[t2],[1:4*ne]);
for i=1:ne;
     for j = 1:4;
        C(i,j) = c(4*(i-1)+j);

     end
 end

tt = linspace(0,1,r);
     for i = 1:ne;
         for k = 1:r;
           uu =(tt(k)-x(i))/(x(i+1)-x(i));
           if(x(i)<=tt(k) & tt(k)<=x(i+1))j=i;
    yaprox(k) = C(i,1)*l1(uu)+C(i,2)*l2(uu)+C(i,3)*l3(uu)+C(i,4)*l4(uu);
           end
             end
```

```
     end
% The exact solution
  if(p>1);
% Uncomment this to get the exit solution at x = 1
% x  = 1;
% The solution at different values of t
t = t2;
x = tt;
z1=(p./t).^(0.5).*(x-t);
z2=(p./t).^(0.5).*(x+t);
z3 = (p./t).^0.5.* (2-x+t);
% The exact solution for large p's
yexit = 1-0.5.*erfc(z1)-(4*p.*t./pi).^(0.5).* exp(-p.*(x-t).^2./t)+...
(0.5.* (1 + 4.*p.*(x+t))).*exp(4.*p.*x).*erfc(z2)...
-2.*( 4.*p.*t./pi).^0.5.*(1+p.*(2-x+t)).*exp(4.*p-(p.*(2-x+t).^2)./t)+...
(2.*p.* (2.*(2-x+t) + t + 2.*p.*(2-x+t).^2).*exp(4.*p)).* erfc(z3);
  else
% Calculating the exact solution for small p's
% Finding the roots x1 in the exact solution for small p's
g = @(x1) tan(2*x1)-(2*x1*p)./(x1.^2-p^2);
% Initial guess
x1(1) = fsolve(g,(pi/4 + 0.8)/2);
% Finding the subsequent roots
for k=0:15
x1(k+2)=fsolve(g,2*(k+1)*pi/4);
end
L= [x1'];
for i = 1:length(t);
sum=0;
for k = 1:length(L);
sum = sum+L(k)*sin(2*L(k))/(L(k)^2+p^2+p)*exp(-L(k)^2*t(i)/p);
end
```

```
% Exact solution for small p's
yexit(i)= exp(p*(2-t(i)))*sum;
end
end
% if p<=1
% yexit = yexit';
% end
% Ploting the approximate solutions
% Uncomment this to plot the exit solution against time
% tt = t
% plot(x,yexit);
% hold on
% plot(x,yaprox)
% xlabel('Distance, x')
% ylabel('Solution')
%  Plotting the error
plot(tt,yexit-yaprox,'-')
xlabel('Distance, x')
ylabel('Error')
% Total error
[sqrt(sum((yexit-yaprox).^2))]
         clear all;
 hold on
end
```

Appendix E

First 17 Roots of (4.17), P = 0.8.

0.79103365241119

1.95857472200193

3.37437573129271

4.87504049558401

6.40739819058517

7.95422009780640

9.50871385303978

11.06773107291779

12.62962923880507

14.19347132241174

15.75868538705734

17.32490312439622

18.89187688116745

20.45943410516168

22.02745093960250

23.59583620712632

25.16452131511047