2009

Employing The Spectral Collocation Method in the Modeling of Laminar Tube Flow Dynamics

Corey Michael Thibeault
Minnesota State University - Mankato

Employing The Spectral Collocation Method in the Modeling of Laminar Tube Flow Dynamics

By

Corey M. Thibeault

A Thesis submitted

In partial fulfillment

Of the requirements for the degree

Master of Science

In

Engineering

Department of Mechanical Engineering

Minnesota State University, Mankato

Mankato, Minnesota

April 2009

4/1/09

This thesis has been examined and approved.

Examining Committee:

Patrick Tebbe

Dr. Patrick Tebbe, P.E. – Committee Chairperson

Saeed Moaveni

Dr. Saeed Moaveni, P.E. – Examining Committee Member

Dr. Namyoung Lee – Examining Committee Member

Patrick Tebbe

Dr. Patrick Tebbe, P.E. – Graduate Coordinator

Dr. Charles W. Johnson, P.E. – Chair Department of Mechanical and Civil Engineering

I. Acknowledgments

I want to begin by thanking the Mechanical Engineering Department for providing me with the perfect environment for continuing my education. The size and atmosphere were exactly what I needed to learn and grow as both an engineer and a person. I am especially grateful to my graduate committee, Dr. Tebbe, Dr. Moaveni and Dr. Lee, for their comments and suggestions in reviewing this manuscript.

I would also like to thank my family for their unwavering support throughout my seemingly endless academic career. Additionally, a special acknowledgement is in order for my grandfather, Stan Smith Jr., whose guidance and advice has influenced my life more than anyone.

Finally, I am forever indebted to my thesis advisor, Dr. Tebbe, who is directly responsible for both the subject of this thesis and my attendance of the graduate program. It was his seemingly innocuous question during my senior year at MSUM, asking my intentions after graduation that started me along my current path.

To all of you, thank you.

II. Abstract

The spectral collocation method is a numerical approximation technique that seeks the solution of a differential equation using a finite series of infinitely differentiable basis functions. This inherently global technique enjoys an exponential rate of convergence and has proven to be extremely effective in computational fluid dynamics. This paper presents a basic review of the spectral collocation method. The derivation is driven with an example of the approximation to the solution of a 1D Helmholtz equation. A Matlab code modeling two fluid dynamics problems is then given. First, the classic two-dimensional Graetz problem is simulated and compared to an analytical solution, a finite difference formulation and a published series solution. An implementation that includes the effects of axial conduction is then compared to the classic series solution for low Peclet number flow, as well as several published results. Finally, two-dimensional laminar diffusion in a tube is modeled and compared to a published analytical solution. The application of spectral collocation to these problems is unique to this study and the results suggest that significant speedup can be achieved in other areas. In addition, the results are in excellent agreement with published data and the Matlab code provides an example of a simple yet effective pseudospectral method implementation.

III. Table of Contents

IV. Table of Figures

V. Table of Listings

VI. Table of Tables

VII. Nomenclature

c_p specific heat of the fluid at constant pressure, $\frac{kJ}{kgK}$

D particle diffusion coefficient, $\frac{m^2}{s}$

$\tilde{D}^1$ first order Chebyshev differentiation matrix

$\tilde{D}^2$ second order Chebyshev differentiation matrix

k thermal conductivity, $\frac{kg}{m^3}$

L tube length, m

n particle concentration

R tube radius, m

r^* dimensionless radius

$\overline{T}$ weighted average temperature

u^* dimensionless velocity

$V_{\max}$ maximum axial velocity, $\frac{m}{s}$

V_{Tz} thermophoretic velocity in z, $\frac{m}{s}$

V_{Tr} thermophoretic velocity in r, $\frac{m}{s}$

z^* dimensionless axial distance

Greek symbols

α thermal diffusivity, $\frac{k}{\rho\ c_p}$

μ dimensionless diffusion

ρ fluid density, $\frac{kg}{m^3}$

Θ dimensionless temperature

θ_m dimensionless bulk fluid temperature

ξ dimensionless axial distance $\frac{z}{L}$

Named Dimensionless Values

$\Pr$ Prandtl Number, $\frac{c_P\mu}{k}$

Re Reynolds Number, $\frac{VL}{\nu}$

Pe Peclet Number, $\mathrm{Re}\Pr$

Nu Nusselt Number, $\frac{hL}{k}$

1. Introduction

This manuscript describes several complex areas of engineering and applied mathematics. The primary focus is on computational fluid dynamics and the spectral collocation method. Spectral collocation is a numerical approximation technique that is especially appealing in fluid modeling but one that can be difficult to develop and apply to generic problems. Presented here is its derivation, as well as illustrations of its strengths and weaknesses, when applied to two specific engineering problems.

The two problems offered are the extended Graetz problem and simple particle diffusion in a tube. Although both of these have enjoyed extensive research, there is no precedent in the literature of their approximation using spectral collocation. A simple and efficient solution to both of these problems can be useful in several different areas of manufacturing and engineering; particularly aerosol research.

In manufacturing, aerosol processes are used to produce optical fibers, ceramics and certain electronics [1]. Conversely, particle deposition is undesirable when it can cause corrosion or fouling of pipelines, heat exchangers, and power generation equipment [2,3]. These problems have also been applied to the estimation of inhaled particles in human airways [4]. An understanding of the temperature distribution and deposition rates in these fields can help predict equipment failures and maintenance schedules, as well as production yields.

The layout of this paper begins with a background and review fluid dynamics as well as the two benchmark problems modeled, the Graetz problem and the simple diffusion model. Chapter 3 then presents a brief introduction to the Spectral Collocation Method along with an example of its application. The numerical discretization for the benchmark problems is then presented in Chapter 4 with the results and discussion of the simulations included in Chapter 5. Finally, Chapter 6 offers some conclusions and future applications of the spectral collocation code developed here.

2. Background

2.1 Computational Fluid Dynamics

Computational Fluid Dynamics (CFD) is the area of mechanics concerned with the numerical modeling of fluid motion. Described by the Navier-Stokes (NS) equations, fluid mechanics is one of the most difficult physical phenomena to model. Most realistic fluids are characterized by the non-linear and coupled forms of the Navier-Stokes equations. Even with sophisticated solution algorithms, high-performance computers, and simplified forms of the NS equations, it can be difficult to achieve accurate approximations [5,6].

2.1.1 Navier-Stokes Equations

Unlike the general equations of motion that relate one or more parameters to each other, such as mass and acceleration to force, the NS equations relate gradients to one another [7]. They can be derived by basic conservation equations, as illustrated below. For most problems there is no analytical solution and numerical methods are required.

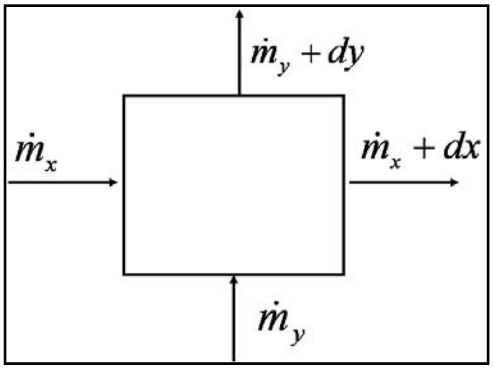

Figure 2.1 Differential Volume for conservation of Mass.

Figure 2.1 shows a differential volume for a fluid particle in two dimensions. The equation can be verbalized as

$$\left\{\begin{matrix}\text{rate of}\\ \text{increase in}\\ \text{Mass}\end{matrix}\right\} = \left\{\begin{matrix}\text{rate of}\\ \text{mass}\\ \text{in}\end{matrix}\right\} - \left\{\begin{matrix}\text{rate of}\\ \text{mass}\\ \text{out}\end{matrix}\right\}$$

In differential vector notation this reduces to Equation (2.1) [7].

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \bar{u} = 0 \tag{2.1}$$

Figure 2.2 shows how momentum is conserved in one dimension using a differential volume element.

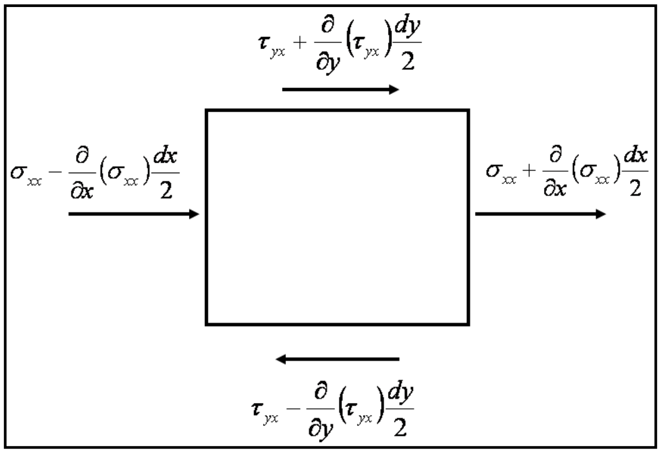
$\tau_{yx}+\frac{\partial}{\partial y}(\tau_{yx})\frac{dy}{2}$
$\sigma_{xx}-\frac{\partial}{\partial x}(\sigma_{xx})\frac{dx}{2}$
$\sigma_{xx}+\frac{\partial}{\partial x}(\sigma_{xx})\frac{dx}{2}$
$\tau_{yx}-\frac{\partial}{\partial y}(\tau_{yx})\frac{dy}{2}$

Figure 2.2 Differential Volume for Conservation of Momentum.

This can verbalized as

$$\left\{\begin{array}{c}\text{rate of}\\ \text{increase}\\ \text{of momentum}\end{array}\right\} = \left\{\begin{array}{c}\text{rate of}\\ \text{momentum}\\ \text{in}\end{array}\right\} - \left\{\begin{array}{c}\text{rate of}\\ \text{momentum}\\ \text{out}\end{array}\right\} + \left\{\begin{array}{c}\text{external}\\ \text{force on}\\ \text{fluid}\end{array}\right\}$$

The resulting differential equation is shown in Equation (2.2) [7]. A detailed description of the derivation of Equation (2.2) from the differential element is left to the reader to explore further.

$$\rho\left(\frac{\partial\overline{u}}{\partial t}+\overline{u}\cdot\nabla\overline{u}\right)=-\nabla p+\mu\nabla^{2}\overline{u}+f \tag{2.2}$$

The Conservation of Energy and Species are also included in many descriptions of the NS equations. Abandoning the differential volume description, the Conservation of Energy can be described as

$$\left\{\begin{array}{c}\text{Rate of}\\ \text{increase of}\\ \text{kinetic and}\\ \text{internal energy}\end{array}\right\} = \left\{\begin{array}{c}\text{net rate of kinetic}\\ \text{and internal}\\ \text{energy addition}\\ \text{by convective}\\ \text{transport}\end{array}\right\} + \left\{\begin{array}{c}\text{net rate of heat}\\ \text{addition by}\\ \text{molecular}\\ \text{transport}\\ \text{(conduction)}\end{array}\right\} +$$

$$\left\{\begin{array}{c}\text{rate of work}\\ \text{done on system}\\ \text{by molecular}\\ \text{mechanisms}\\ \text{(i.e., by stresses)}\end{array}\right\} + \left\{\begin{array}{c}\text{rate of work}\\ \text{done on system}\\ \text{by external}\\ \text{forces}\\ \text{(e.g., by gravity)}\end{array}\right\} .$$

Equation (2.3) expresses this in differential form [7].

$$\begin{aligned}&\frac{\partial}{\partial t}\left(\tfrac{1}{2}\rho v^2 + \rho\hat{U}\right) = -\left(\nabla\cdot\left(\tfrac{1}{2}\rho v^2 + \rho\hat{U}\right)\bar{v}\right) - \left(\nabla\cdot\bar{q}\right) - \\ &\left(\nabla\cdot p\bar{v}\right) - \left(\nabla\cdot\left[\bar{\tau}\cdot\bar{v}\right]\right) + \rho\left(\bar{v}\cdot\bar{g}\right)\end{aligned} \qquad \textbf{(2.3)}$$

The conservation of mass species for component A of a mixture can be presented as:

$$\left\{\begin{array}{c}\text{rate of}\\ \text{mass of}\\ \text{A in}\end{array}\right\} - \left\{\begin{array}{c}\text{rate of}\\ \text{mass of}\\ \text{A out}\end{array}\right\} + \left\{\begin{array}{c}\text{rate of production of}\\ \text{mass of A by}\\ \text{Homogeneous reaction}\end{array}\right\} = 0$$

This is described by Equation (2.4) [7].

$$\frac{\partial\rho_\alpha}{\partial t} = -\left(\nabla\cdot\rho_\alpha\bar{v}\right) - \left(\nabla\cdot\bar{j}_\alpha\right) + r \qquad \textbf{(2.4)}$$

2.1.2 Spectral Method

The spectral method is a numerical modeling technique for approximating the solution of partial and ordinary differential equations. Related to the method of weighted residuals, spectral methods employ infinitely differentiable functions as trial functions. The result is a global method with an exponential rate of convergence for problems with smooth solutions. Since the discovery of fast Fourier transforms the use and practicality of spectral methods has steadily increased. This is especially true in fluid mechanics research.

2.1.3 Spectral Method Literature Review

The idea of using spectral representations as numerical solutions to ordinary differential equations is often accredited to the work of Lanczos in 1938 [8,9]. Although many different variations of the method of weighted residuals and orthogonal collocation methods were developed, it was the works of Kreiss and Oliger as well as a separate work by Orszag in 1972 that presented the earliest applications of the spectral collocation method. At the time it was referred to as a Fourier method and pseudospectral method respectively [9]. The advancements of the method were made possible at the time by the newly discovered Fast-Fourier Transform algorithm.

The first published treatment of spectral methods was completed by Gottlieb and Orszag in 1977 [10]. This monograph paved the way for the more advanced extensions of the method for a larger variety of problems; including the applications to fluid dynamics by Canuto et al. [9], [11], as well as the generic treatments of Boyd [12], Guo [13], Fornberg [8] and Trefethen [14].

Although the spectral method has been applied to many different CFD problems, there are still many models that have been investigated with the finite difference and finite element methods that could be reopened to the spectral method.

2.1.4 Finite Difference Method

The finite difference method is one of the most popular numerical modeling techniques. This popularity is due to its ease of implementation along with its ability to model complex geometries [15]. Numerical differentiation requires an estimate to the derivative or slope of a function by using the values at a set of discrete points along the function. For the implementation here, a central difference scheme is offered. This is essentially a linear interpolation about the point of interest. Equations (2.5) through (2.7) present examples of numerical approximations for derivative terms found in the non-dimensionalized conservation of energy equation [5-6,15-16]. These can be proven by a Taylor series expansion; which is left to the reader to explore further.

$$\frac{d\Theta}{dz^*}=\frac{\Theta(i,j+1)-\Theta(i,j-1)}{2\Delta z^*} \tag{2.5}$$

$$\frac{d\Theta}{dr^*}=\frac{\Theta(i+1,j)-\Theta(i-1,j)}{2\Delta r^*} \tag{2.6}$$

$$\frac{d^2\Theta}{dr^{*2}}=\frac{\Theta(i+1,j)-2\Theta(i,j)+\Theta(i-1,j)}{\left(\Delta r^*\right)^2} \tag{2.7}$$

2.2 The Graetz Problem

The Graetz problem is concerned with the heat transfer between a flowing fluid and a circular tube. Essentially a fluid at constant temperature is forced along the length of a pipe until the flow becomes laminar. At that point the fluid velocity occurs parallel to the length of the pipe and there is no momentum in the radial or circumferential directions. At a point, $z=0$, the wall temperature changes. This difference in temperature causes a net heat-flux between the fluid and the wall. The distribution in fluid temperature along the length of the pipe is what is considered by the Graetz problem.

For laminar tube flow the velocity profile is parabolic; described by Equation (2.8) [7,18].

$$v(r) = V_{\max}\left(1 - \left(\frac{r}{R}\right)^2\right) \tag{2.8}$$

Where:

$V_{\max}$ is the velocity along the centerline of the tube.

R is the tube radius.

Figure 2.3 is an illustration of laminar tube flow.

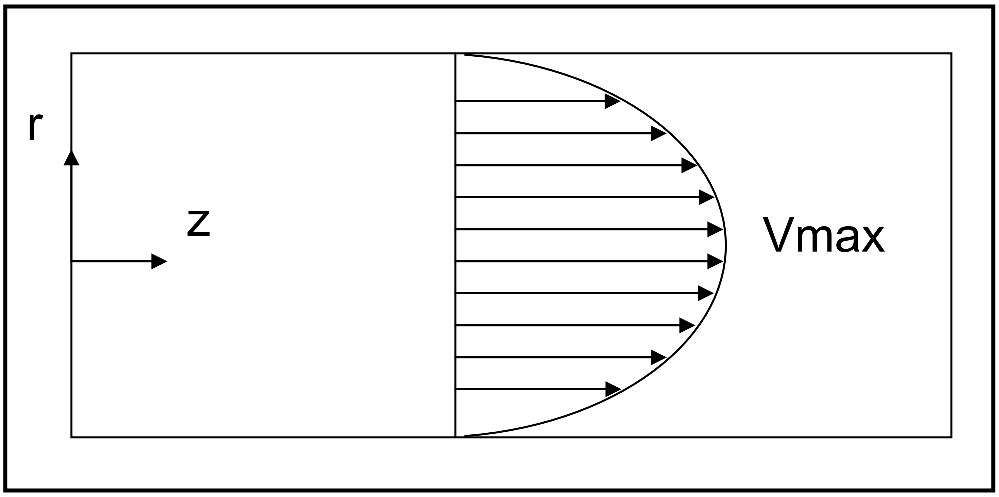

Figure 2.3 Laminar Tube Flow.

The laminar assumption allows Equation (2.8) to be substituted in the Energy equation directly. This is developed further below.

2.2.1 Literature Review

The heat transfer of a fluid flowing in a pipe is a classic problem. With over 120 years of investigation there are numerous analytical and numerical solutions of the problem. The original analytical solutions proposed by Graetz and Nusselt ignored the effects of axial conduction on the temperature distribution [17]. With these solutions the first few Eigen values of the asymptotic expansion were found along with the corresponding coefficients. This problem and method of solution is generally referred to as the Graetz problem and Graetz method respectively, due to the seminal work published by German physicist Leo Graetz in 1883 [17,19]. Subsequent studies focused on expanding the accuracy and number of eigenvalues for the expansion, solving the problem for other geometries or further defining the entrance region.

The simplified solution has been shown to be appropriate for fluids with high Peclet numbers. The breakdown of the solution occurs as the flow slows down and two things occur. The first is the presence of axial conduction. The fluid down-stream will affect the up-stream fluid's temperature distribution. The second is the preheating effects of the incoming fluid. The majority of analytical solutions for low Peclet flow, most often referred to as the Extended Graetz problem, will break the physical domain in two. The boundary conditions are enforced at the origin and two different expansions are generally used to solve the overall problem.

Graetz was able to calculate the first two eigenfunctions of the series solution to the problem. This was later extended to three by Nusselt in 1910 [20]. In 1955 Sellars et al. [21] was able to extend that number of eigenvalues from 3 to 10 for the classic Graetz problem. However, it was shown that accurate solutions could be achieved within the first 4 eigenvalues. In addition, Sellars explored the inverse problem of finding the temperature at constant heat flux. Similarly this case was explored by Bird et al. [7]; where an asymptotic solution was reached for long tubes. Recently Housaidas et al. [22] built on the work of Sellars [21] by determining mathematically where the general asymptotic solution breaks down and the Leveque solution becomes more appropriate.

There have been many other analytic and numerical methods applied to the Graetz problem. Jones et al. [18] considered an infinitely long tube with a circular cross-section. One semi-infinite half is at taken at one temperature T_0 and the other half is at Temperature T_1. It was assumed that the physical properties were independent of the temperature. The solution was arrived at by first employing a two-sided Laplace transform on the energy equation (developed below). An inverse Laplace was then applied to obtain the general form of the solution. A series solution, or more formally a Frobenius method, was then used to find the eigenfunctions and subsequent eigenvalues of the general expression.

Michelsen et al. [23] used a combination orthogonal collocation and matrix diagonalization. In addition, an asymptotic solution was developed based on perturbation of the eigenfunctions for the simplified model. It was initially questioned whether or not the classical Fourier series methods provided the most practical numerical representation of the solution. The conclusion was that the overall solution could be reached with much higher accuracy with fewer nodes than the asymptotic solution. However, both methods struggle with convergence in the entrance region.

Ou et al. [24] presented a method that solved the energy equations using a DuFort-Frankel explicit method. This is a finite difference technique that can be more stable than traditional differencing schemes. It does however require two known conditions at the boundaries. This solution offered comparable accuracy to the more difficult series solutions.

Bayazitoglu et al. [25] presented a finite integral transform technique. The resulting method was applicable to both laminar and turbulent flow inside conduits. Convective boundary conditions, axial conduction and internal energy sources were also included. The method presented was a combination of a numerical scheme and an asymptotic expansion. The overall domain was divided in half and a two-sided Laplace transform technique was used. The series solution was then achieved by expanding the nonorthogonal eigienfunctions in terms of an auxiliary system. In this case those were Gramm-Schmidt orthogonal functions. Laohakul et al. [26] extended this work to better solve low Peclet number flows. The finite integral technique was extended to include the analysis of multiple heating-cooling sections along the tube using Greens function.

Ku et al. [27] developed two computational methods. The first a Chebyshev-finite difference method and the second a Chebyshev-finite element method. The so called Chebyshev-finite difference method was constructed using a Chebyshev-collocation scheme in the radial direction with a second order finite difference approximation on equally spaced points in the radial direction. The Chebyshev-finite element method used a global element approach with the axial domain divided into a small number of elements in an attempt to overcome Gibbs oscillations. Essentially Chebyshev expansions are developed for individual sub-regions along the axial domain.

Poirier et al. [28] looked exclusively at the entrance region using a finite volume technique. The resulting analysis revealed that the solution was extremely sensitive to the number and configuration of node points. The conclusion was that any numerical analysis of the entrance region should be conducted by increasing the number of node points until convergence is reached. At which point a coordinate transform can be used to reduce the number of grid points.

Ebdian et al. [29] considered the extended Graetz problem using an analytical solution and Fourier integral transformation. Similarly Telles et al. [30] used the orthogonal Gram-Charlier basis functions to form an infinite expansion to the solution. The solution was valid over the entire domain of $(-\infty,\infty)$ and does not require the usual domain separation techniques.

Mercado et al. [31] looked at the Graetz problem under turbulent flow. A combination numerical model was developed. In the radial direction a fourth order finite difference technique was used. The resulting set of equations was then integrated in the axial direction using a fourth order Runge-Kutta method. In the laminar case this method proved to be non-iterative and the authors concluded that this made it more desirable and faster than conventional computational fluid dynamics packages.

2.2.2 The Simple Graetz Problem

In order to simplify the tube flow problem presented above, some assumptions need to be made. First, the flow is irrotational, meaning the molecules that comprise the fluid have no rotational velocity. The bulk fluid can still have non-linear motion, but there is no spin to the individual elements [7]. Second, the viscosity of the fluid will be neglected, meaning that there is no viscous dissipation. Third, there is no particle slip at the boundary. Fourth, all other forms of heat generation are negligible. Fifth, there is no temperature dependence on the fluid properties. Finally, for the classic case considered here, conduction in the axial direction is negligible.

Additionally, it is assumed that the flow is axisymmetric (mirrored about the tube centerline). As stated before this means the velocity only changes with respect to the distance radially from the center.

Equations (2.1) and (2.2) will automatically be satisfied by the given velocity profile, which can then be substituted directly into Equation (2.3). The resulting energy conservation equation is

$$\rho\, c_p V_{\max}\left(1-\left(\frac{r}{R}\right)^2\right)\frac{\partial T}{\partial z}-k\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right)\right]=0 \tag{2.9}$$

Equation (2.9) can be further simplified by introducing non-dimensional variables. Non-dimensionalization provides a way to compare systems of different length scales.

Introducing the following non-dimensionalized variables [7].

$$r^*=\frac{r}{R},\qquad z^*=\frac{z}{R},\qquad u^*=\frac{u}{u_{\max}},\qquad \mathrm{Re}=\frac{u_{\max}\cdot R}{\upsilon},\qquad \Pr=\frac{\upsilon}{\alpha}$$

$$\Theta=\frac{T-T_{rw}}{T_{re}-T_{rw}}$$

The Reynolds number (Re) relates the inertial forces of the fluid with the viscous forces [32], and the Prandtl number (Pr) relates the viscous diffusivity of the fluid with the thermal diffusivity [7].

Substituting these into Equation (2.9),

$$\mathrm{Re}\cdot\Pr\cdot\left(1-\left(r^*\right)^2\right)\frac{\partial\Theta}{\partial z^*}-\frac{1}{r^*}\frac{\partial}{\partial r^*}\left(r^*\frac{\partial\Theta}{\partial r^*}\right)=0 \tag{2.10}$$

Expanding the derivatives in Equation (2.10),

$$\mathrm{Re}\cdot \mathrm{Pr}\cdot \left(1-\left(r^*\right)^2\right)\frac{\partial \Theta}{\partial z^*}-\frac{1}{r^*}\frac{\partial \Theta}{\partial r^*}-\frac{\partial^2 \Theta}{\partial {r^*}^2}=0 \qquad \textbf{(2.11)}$$

2.2.3 The Extended Graetz Problem

When the Peclet number is sufficiently low, $pe < 10$, axially conduction will have a significant effect on the heat transfer of the fluid. This is referred to as the extended Graetz problem. The addition of axial conduction to Equation (2.11) will result in

$$\mathrm{Re}\cdot \mathrm{Pr}\cdot \left(1-\left(r^*\right)^2\right)\frac{\partial \Theta}{\partial z^*}-\frac{\partial^2 \Theta}{\partial {z^*}^2}-\frac{1}{r^*}\frac{\partial \Theta}{\partial r^*}-\frac{\partial^2 \Theta}{\partial {r^*}^2}=0 \qquad \textbf{(2.12)}$$

2.2.4 Classic Graetz Series Solution

The Graetz series solution is formed using a recurrence relation for the series approximation of the energy equation after a separation of variables. This method has been shown to be very accurate for all regions of the domain with the exception of the entrance region. At the entrance the convergence of the series is very slow. In fact there are some cases where convergence is not reached with upwards of 150 terms. Leveque solved the problems the series solution experiences in the entrance region by using a method of similarity transform [6]. This alternative method is only appropriate for the thermal entrance region and simplifications are made based on that. This is not explored here but can be found in several references [6,20].

Originally Graetz treated the problem as a solid rod moving through a pipe; known as slug flow. Later it was treated as Laminar or Poiseuille flow. The solution, which takes considerably more numerical computation, is presented for reference here.

2.2.5 Series Solution Development

Beginning with Equation (2.9) and the following dimensionless variables,

$$\eta = \frac{r}{R}, \qquad \xi = \frac{z/R}{\mathrm{Re}\,\mathrm{Pr}}, \qquad \Theta = \frac{T - T_w}{T_e - T_w}$$

Notice that these are slightly different than the values presented above. This dimensionalization makes a separation of variables easier. This Results in

$$\left(1 - (\eta)^2\right)\frac{\partial \Theta}{\partial \xi} - \frac{1}{\eta}\frac{\partial \Theta}{\partial \eta} - \frac{\partial^2 \Theta}{\partial \eta^2} = 0 \tag{2.13}$$

The solution is assumed to be of the form

$$\Theta = X(\xi)R(\eta) \tag{2.14}$$

When substituted into Equation (2.13) the result is

$$\frac{1}{X}\frac{\partial \Theta}{\partial \xi} = \frac{1}{R\left(1-(\eta)^2\right)}\left[\frac{1}{\eta}\frac{\partial R}{\partial \eta} + \frac{\partial^2 R}{\partial \eta^2}\right] = -\lambda^2 \tag{2.15}$$

From this

$$\frac{\partial \Theta}{\partial \xi} + \lambda^2 X = 0 \tag{2.16}$$

is found and has the known solution [20],

$$X(\xi) = Ae^{-\lambda^2 \xi} \tag{2.17}$$

A is just an arbitrary constant. The other side of the separation results in [20],

$$\frac{\partial^2 R}{\partial \eta^2} + \frac{1}{\eta}\frac{\partial R}{\partial \eta} + \lambda^2\left(1-(\eta)^2\right)R = 0 \tag{2.18}$$

With the prescribed boundary conditions this will form a Sturm-Louville system; where $R_n(\eta)$ are the eigenfunctions and λ_n are the eigenvalues [20]. An explanation of this can be found in [33]. Once the values for $R_n(\eta)$ are known the general solution is Equation (2.19). Graetz found the eigenfunctions using a series expansion of Equation (2.18). Eventually numerical integration allowed more eigenfunctions to be found.

$$\Theta = \sum_{n=0}^{\infty} C_n R_n(\eta) e^{-\lambda_n^2 \xi} \tag{2.19}$$

2.2.6 Graetz Series Solution Code

The Graetz series code was originally written by Dr. Patrick Tebbe and is included here with permission. The solution is based on the work of [22] where Equation (2.19) is expressed as the asymptotic formulas of λ_n, R_n and Λ_n. It should be noted that Λ_n results from the solution of the expansion coefficient C_n and is equal to $\left(\frac{\partial R_n}{\partial \lambda_n}\right)_{\eta=1}$; this is explained in [20-21]. Equation (2.19) is further rearranged to solve for the bulk fluid:

$$\Theta_M(\xi) = 8\sum_{n=0}^{\infty} \frac{R_n(\eta)e^{-\lambda_n^2\xi}}{\lambda_n^3\Lambda_n} \tag{2.20}$$

The main computation loop is included in Listing 2.1 below. The full code listing is included in Appendices 7.1 and 7.2.

```
    for loop=1:N+1
        count=count+1;
        theta=0;
        n=0;
        % the error value must be initialized to a large value
        error=1000;
        % the non-dimensional axial location is calculated
        phi=z(loop)/(Rad*Pe);
        % a loop is started for the terms in the series solution,
        % it keeps going until error is below the limit or 10 terms
        % have been added
        while (error>emax)&&(n<=10)
            n=n+1;
            % the graetz function is used to calculated the
            % eigenvalues, etc.
           [lambda,R,LAMBDA] = graetz(n,S,P,Q);
            % the term is added to the summation and the error
            % is calculated
           old=theta;
           %  [Housiadas eq. 2]
           theta = theta + 8*R* …
                        exp(-1*lambda^2*phi)/(lambda^3*LAMBDA);
           error=(theta-old)/theta*100;
        end;
        % final values are output to the arrays for later plotting
        % the terms array keeps track of how many terms are used at
        % each axial location
        z(count)=loop;
        relerror(count)=error;
        terms(count)=n;
        T(count)=theta;%
    end;
```

Listing 2.1 Computation loop for Graetz series solution

2.3 Simple Diffusion Model

The second problem considered is the deposition of particles dispersed in a fluid under laminar flow conditions in a cylindrical tube. When particles are exposed to a concentration gradient those particles will diffuse by Brownian motion from high concentrations to low concentrations. When the particles are dispersed in a fluid that is traveling down a tube or between parallel plates, the walls of the vessel will act like a sink to those particles. For most models the concentration at the wall is taken to be zero and the particles will deposit there.

2.3.1 Literature Review

The modeling of aerosol deposition in a tube is a well characterized problem. There have been several analytical solutions for the deposition efficiency. The first was developed by Gormley and Kennedy in 1949 [34-35]. This is the benchmark solution referred to by most of the subsequent studies. C. N. Davies presented an analytical solution that differed from the Gormley results [36]. However, Ingham [36] later showed that there was a mistake in that evaluation. In addition, a solution for small values of the diffusion coefficient was developed.

2.3.2 Analytical Solution

The analytical solution first developed by Gormley and Kennedy and slightly reformed by Hinds [34], is

$$\begin{gathered} \mu < 0.009 \\ P = 1 - 5.5 \cdot \mu^{\left(\frac{2}{3}\right)} + 3.77 \cdot \mu \\ \mu \geq 0.009 \\ P = 1 - 0.819 \cdot e^{(-11.5\mu)} + 0.0975 \cdot e^{(-70.1\mu)} \end{gathered} \tag{2.21}$$

Where μ is defined as

$$\mu = \frac{D \cdot L}{Q} \tag{2.22}$$

2.3.3 Mathematical Formulation

For the simple problem it is assumed that there is no formulation of aerosols and the mass transfer along the direction of flow due to diffusion can be neglected. These are both valid as long as the Peclet number is significantly greater than 1. As before the flow is considered laminar. Beginning with the two-dimensional species equation, Equation (2.4) ,

$$\frac{1}{r}\frac{\partial}{\partial r}\left[r\rho\left(u+V_{Tr}\right)n\right]+\frac{\partial}{\partial z}\left[\rho\left(V+V_{Tz}\right)n\right]=\frac{1}{r}\frac{\partial}{\partial r}\left(r\rho D\frac{\partial n}{\partial r}\right)+\frac{\partial}{\partial z}\left(\rho D\frac{\partial n}{\partial z}\right) \tag{2.23}$$

Neglecting:

Axial Diffusion: $\frac{\partial}{\partial z}\left(\rho D\frac{\partial n}{\partial z}\right)$

Thermophoresis in z and r: V_{Tz} & V_{Tr}

Velocity in r: u

Will result in

$$\frac{\partial n}{\partial z}\left[V\right]=\frac{1}{r}\frac{\partial}{\partial r}\left(rD\frac{\partial n}{\partial r}\right) \tag{2.24}$$

Using Diffusivity that is not dependent on temperature and plugging in parabolic velocity, results in the governing equation:

$$v_{z,\max}\left[1-\left(\frac{r}{R}\right)^2\right]\frac{\partial n}{\partial z}=D\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial n}{\partial r}\right) \tag{2.25}$$

3. Spectral Methods

The spectral method can be described as an approximation to the solution of a differential equation using a finite series of smooth basis functions. Derived from the Method of Weighted Residuals, spectral methods enjoy a so called exponential rate of convergence (compared to the polynomial convergence rate of most other numerical methods). They are best suited for simple domains (i.e. squares or circles) and have had broad success in several areas including Weather, Turbulence, Seismic and Quantum Modeling [8-9,11-13].

There are several features that set spectral methods apart from other numerical solutions to partial differential equations (PDE). The first is the computational domain. In Finite Element (FE) modeling, for example, the overall physical domain is broken up into a number of subdomains (elements) and a local basis function is chosen to be non-zero over a couple of sub-intervals within that domain. Conversely, spectral methods chose a basis function that is global to the entire computational domain and is non-zero except at isolated points. It is this reason that spectral methods are often referred to as a global numerical method.

The second distinguishing feature of spectral methods is the choice of basis functions. Spectral methods select basis functions that are high degree polynomials or trigonometric polynomials that are infinitely differentiable. The basis functions for FE methods are generally low-order polynomials and as stated before, local in nature.

Because of this, FE methods are better suited for complex geometries but suffer from lower accuracy as compared to spectral methods.

3.1 Derivation

The derivation of the spectral method can be motivated by the Method of Weighted Residuals (MWR). There are two main components of the MWR, those are the trial or basis functions and the weight or test functions. Suppose that there is a linear operator L acting on a function $u(x)$ and the result is a function $f(x)$,

$$Lu(x) = f(x) \tag{3.1}$$

It is assumed that the unknown $u(x)$ can be approximated by a linear combination of $N+1$ basis functions, $\phi_n(x)$,

$$u(x) \approx u_N(x) = \sum_{n=0}^{N} a_n \phi_n(x) \tag{3.2}$$

When the approximation $u_N(x)$ is substituted into Equation (3.1) the result is not exactly the desired function $f(x)$. Instead the result is the residual equation,

$$R(x,;a_0, a_1, ..., a_N) = Lu_N(x) - f(x) \tag{3.3}$$

For the exact solution the residual R will be zero. The general idea is to find the series coefficients that bring the residual to a minimum.

The residual is forced to zero in some average sense over the computational domain. That is, where the number of weight functions, W_i , is exactly equal to the number of unknown constants,

$$\int_X R(x)W_i dx = 0 \tag{3.4}$$

The result is set of algebraic equations for the unknown constants a_i .

As stated above, it is the choice of basis functions, $\phi_n(x)$, that separates spectral methods from finite difference and finite element methods. Similarly, it is the choice of weight functions W_i that separate the different types of spectral methods.

3.1.1 Spectral/Galerkin

The first method is referred as the spectral or Galerkin spectral method. Here, the weight functions are the same as the basis functions. They are infinitely smooth functions which individually satisfy the boundary conditions. The differential equation is then enforced by requiring that the integral of the residual times each test function be zero; as explained above [12].

3.1.2 Tau

In the Tau method the boundary conditions are not directly satisfied by the global expansion functions. Some equations are required to ensure those boundaries are satisfied. Now Equation (3.2) becomes Equation (3.5), [13].

$$u_N(x) = u_B(x) + \sum_{n=0}^{N} a_n \phi_n(x) \tag{3.5}$$

3.1.3 Collocation/Pseudospectral

In the collocation method the weight functions, W_i , are selected from the group of Dirac delta functions. These functions have the properties expressed in Equation (3.6).

$$\delta(x - x_i) = \begin{cases} 1 & x = x_i \\ 0 & x \neq x_i \end{cases} \tag{3.6}$$

The integration of (3.4) now results in:

$$R(x_i) = 0 \tag{3.7}$$

The spectral collocation method employs Gauss-type quadrature formulas where the collocation points that are chosen are roots of the basis functions, $\phi_{n+1}(x)$. Therefore, Equation (3.2) is satisfied exactly at the selected points. Rather than solving for the expansion coefficients (a_n) as in the Galerkin Spectral Method, the collocation method uses them as an intermediate step, specifically, in the analytic differentiation [9].

3.2 Chebyshev Polynomials

Chebyshev polynomials are developed as special cases of ultraspherical polynomials [33]. As far as spectral methods are concerned, the most noteworthy aspect of Chebyshev polynomials is that their expansion is nothing more than Fourier cosine series in disguise [12]. This is such an important consequence because it allows the application of Spectral Methods to non-periodic domains using the same theorems and identities as Fourier methods. The domain of convergence for the expansion however now changes from $[0,\pi]$ to $[-1,1]$. For the problems presented in the paper Chebyshev expansions and collocation points are used exclusively.

The Chebyshev expansion of a function $u \in L_w^2(-1,1)$ is [11]:

$$u(x)=\sum_{k=0}^{\infty}\hat{u}_k T_k(x), \quad \hat{u}_k = \frac{2}{\pi c_k}\int_{-1}^{1} u(x)T_K(x)w(x)dx \tag{3.8}$$

Where $T_k(x)$ is the Chebyshev polynomial defined as

$$T_k(x)=\cos k\theta, \quad \theta = \arccos x \tag{3.9}$$

As stated before, Chebyshev polynomials are just cosine series after a change of independent variable. The polynomials can now be expanded in power series to form

$$T_k(x) = \frac{k}{2}\sum_{l=0}^{\left[K/2\right]}(-1)^k \frac{(k-l-1)!}{l!(k-2l)!}(2x)^{k-2l} \tag{3.10}$$

This can then be described by the recursive equation

$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x) \tag{3.11}$$

The restriction that Chebyshev polynomials place on the domain can be explained by Darboux's principle. Stated plainly, the convergence of a spectral series of $u(x)$ is controlled by the singularities of $u(x)$. Singularities are any points in the complex plain where $u(x)$ ceases to be analytic. In reality this can be a number of influences, including square roots, logarithms, simple poles, step function discontinuities and infinities or abrupt discontinuities in any of the derivatives of $u(x)$ [12].

3.3 Chebyshev Derivative Matrix

Differentiation using spectral methods can be a confusing concept. The first thing to take note of is the change of variables that occurs when moving from Chebyshev polynomials to Fourier series. With that in mind, the unique aspect of Chebyshev polynomials is that the basis functions and their derivatives can be constructed using trigonometric functions. Also of note, differentiation as defined here, can be achieved in either transform space or in physical space. For this paper all differentiation is done in physical space.

It is difficult to completely grasp the pseudospectral method without some understanding of polynomial interpolation. Without diverging from the topics of this paper too much, a brief background is presented. In general an unknown function $f(x)$ can be approximated by a polynomial of Nth degree. In general N+1 points can be fit by a polynomial using Equation (3.12) [12].

$$I_N(x) = \sum_{i=0}^{N} u(x_i)\psi_i(x) \tag{3.12}$$

This is the Lagrange Interpolation Formula, where $\psi_i(x)$ represents the cardinal functions and are polynomials of degree N that satisfy Equation (3.13) [12].

$$\psi_i(x) = \delta_{ij} \tag{3.13}$$

These are defined by

$$\psi_i(x) = \prod_{j=0,\, j\neq i}^{N} \frac{x - x_j}{x_i - x_j} \tag{3.14}$$

The use of this concept should become clear below.

The most important tool in this paper is the Chebyshev derivative matrix. To help explain how this is developed consider the derivative of a function u expanded in Chebyshev polynomials, based on Equation (3.8) above.

$$u' = \sum_{k=0}^{\infty} \tilde{u}_k^{(1)} T_k \qquad \textbf{(3.15)}$$

Using the identity $2\sin\theta\cos k\theta = \sin(k+1)\theta - \sin(k-1)\theta$ and Equation (3.11) will give Equation (3.16) [11].

$$c_k \tilde{u}_k^{(1)} = \tilde{u}_{k+2}^{(1)} + 2(k+1)\tilde{u}_{k+1} \qquad \textbf{(3.16)}$$

These yield,

$$\hat{u}_k^{(1)} = \frac{2}{c_k} \sum_{\substack{p=k+2 \\ p+k \ odd}}^{\infty} p\hat{u}_k, \qquad k \geq 0 \qquad \textbf{(3.17)}$$

The second derivative is

$$u'' = \sum \hat{u}_k^{(2)} T_k \qquad \textbf{(3.18)}$$

The resulting coefficients can be calculated using Equation (3.19) [11].

$$\hat{u}_k^{(2)} = \frac{1}{c_k} \sum_{\substack{p=k+2 \\ p+k \ odd}}^{\infty} p\left(p^2 - k^2\right) \hat{u}_k^{(2)} \qquad k \geq 0 \qquad \textbf{(3.19)}$$

The Gauss-Lobatto points defined by Equation (3.20) below are points along the domain $[-1,1]$ that are roots of the polynomial $\left(1 - x^2\right)\phi_N^{'}$.

$$x_j = \cos\frac{\pi j}{N}, \qquad w_j = \begin{cases} \dfrac{\pi}{2N}, & j = 0, N \\ \dfrac{\pi}{N}, & j = 1,\ldots, N-1 \end{cases} \tag{3.20}$$

Using these, the characteristic Lagrange polynomials defined in Equation (3.14) can be expressed as:

$$\psi_i(x) = \frac{(-1)^{i+1}\left(1-x^2\right)T_N^{'}(x)}{\bar{c}_i N^2 \left(x-x_i\right)} \tag{3.21}$$

From this the Chebyshev first derivative interpolation matrix is defined by Equation (3.22) [11-12,14].

$$\left(D_N\right)_{ji} = \begin{cases} \dfrac{\bar{c}_j}{\bar{c}_j}\dfrac{(-1)^{j+i}}{x_j - x_i}, & j \neq i \\ -\dfrac{x_i}{2\left(1-x_i^2\right)}, & 1 \leq j = i \leq N-1 \\ \dfrac{2N^2+1}{6}, & j = i = 0 \\ -\dfrac{2N^2+1}{6}, & j = i = N \end{cases} \tag{3.22}$$

This is expressed best visually and is shown below in Figure 3.1. This figure is based on [14] pp. 53.

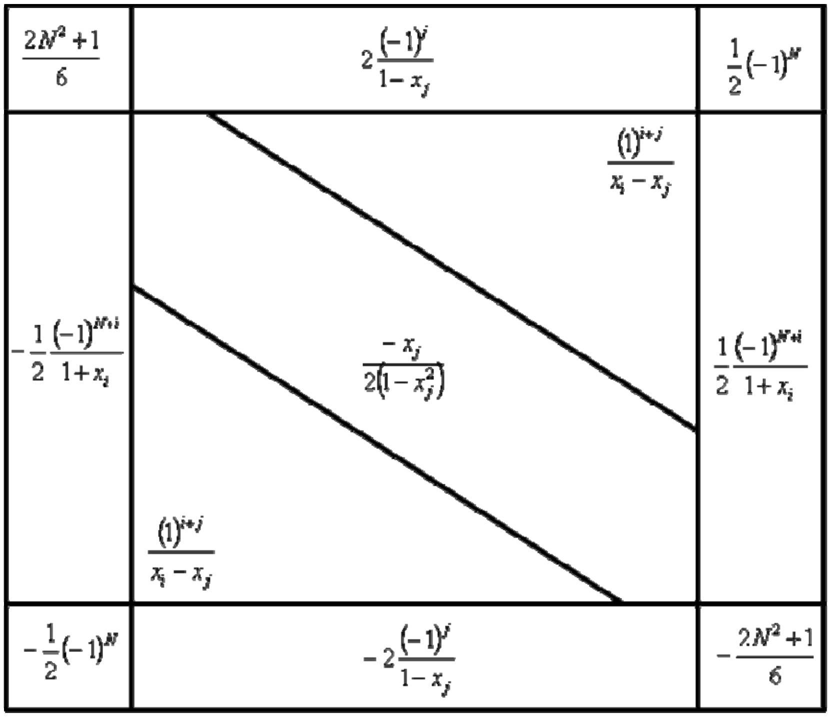
$\frac{2N^2+1}{6}$
$2\frac{(-1)^j}{1-x_j}$
$\frac{1}{2}(-1)^N$
$\frac{(1)^{i+j}}{x_i - x_j}$
$-\frac{1}{2}\frac{(-1)^{N+i}}{1+x_i}$
$\frac{-x_j}{2(1-x_j^2)}$
$\frac{1}{2}\frac{(-1)^{N+i}}{1+x_i}$
$\frac{(1)^{i+j}}{x_i - x_j}$
$-\frac{1}{2}(-1)^N$
$-2\frac{(-1)^j}{1-x_j}$
$-\frac{2N^2+1}{6}$

Figure 3.1 Chebyshev Differentiation Matrix Schematic.

Listing 3.1 is the Matlab/Octave code for constructing the Chebyshev derivative matrix and the collocation points described by Equations (3.22) and (3.20) respectively. This code is based on Cheb.m from [14].

```
function [D,x] = cheb_derivative(N)
  if N==0, D=0;
      x=1;
  return, end
  % Collocation Points
  x = cos(pi*(0:N)/N)';
  X = repmat(x,1,N+1);
  dX = X-X';
  % Derivative Matrix
  D=zeros(N+1,N+1);
  % Top and Bottom Rows
  top = (2*(-1).^(1:N-1))./(1-reshape(x(2:N),1,N-1));
  bottom=-top(N-1:-1:1);
  % Left and Right
  right = -reshape(bottom./4,(N-1),1);
  left= -right(N-1:-1:1);
  %top
  D(1,2:N) = top;
  D(N+1,2:N) = bottom;
  D(2:N,1)=left;
  D(2:N,N+1) =right;
  % Middle
  ij = repmat([0:N],N+1,1)+repmat(reshape([0:N],N+1,1),1,N+1);
  mid = ((-1).^ij)./dX;
  D(2:N,2:N) = mid(2:N,2:N);
  % Diagonal
  dia=(1:N+1:(N+1)*(N+1)) + (0:N);
  dia = dia(2:N);
  D(dia) = -(x(2:N)./(2*(1-x(2:N).^2)));
  % Corners
  D(1) = (2*N^2+1)/6;
  D((N+1)*(N+1))=-D(1);
  D(N+1) = -0.5*(-1)^N;
  D((N+1)*(N+1)-(N)) = -D(N+1);
```

Listing 3.1 Matlab/Octave code for producing Chebyshev collocation points and derivative matrix

3.4 Example

An example will help illustrate the theoretical concepts presented above. A simple one-dimensional Helmholtz problem is described as

$$\frac{d^2y}{dx^2} + k^2y = 0 \tag{3.23}$$

For reference, the solution to this differential equation is

$$y = C_1\cos(kx) + C_2\sin(kx) \tag{3.24}$$

Where C_1 and C_2 are constants based on the given boundary conditions. If $k = 2$ and the boundary conditions are chosen as

$$y(1) = y(-1) = 1 \tag{3.25}$$

The resulting constants will be $C_1 = -2.403$ and $C_2 = 0$. The exact solution for the given conditions then becomes:

$$y = -2.403\cos(2x) \tag{3.26}$$

To begin, a polynomial approximation using the collocation method is presented. The open source mathematical package Sage was used and the worksheet listing is included in Appendix B. For most problems a polynomial approximation is ideal [12]. Equation (3.27) is selected for this problem.

$$y_2 = 1 + \left(1 - x^2\right)\left(a_0 + a_1 x + a_2 x^2\right) \quad \textbf{(3.27)}$$

The residual for this approximation will then be:

$$R\left(x; a_0, a_1, a_2\right) = y_{2,xx} + 4 y_2 \quad \textbf{(3.28)}$$

Which will result in

$$R = 4\left(\left(1 - x^2\right)\left(a_2 x^2 + a_1 x + a_0\right) + 1\right) - 2\left(a_2 x^2 + a_1 x + a_0\right) + 2a_2\left(1 - x^2\right) - 4x\left(2a_2 x + a_1\right) \quad \textbf{(3.29)}$$

Once again the goal of the collocation method is to reduce the residual to zero over a number of points that is equal to the number of unknown coefficients. Arbitrarily the points are chosen to be $x_i = \left(-\frac{1}{2}, 0, \frac{1}{2}\right)$. Substituting x_i into Equation (3.29) results in:

$$eq1 = \tfrac{3}{2} a_2 - 2\left(\tfrac{1}{4} a_2 - \tfrac{1}{2} + a_0\right) + 2\left(a_1 - a_2\right) + 3\left(\tfrac{1}{4} a_2 - \tfrac{1}{2} a_1 + a_0\right) + 4 \quad \textbf{(3.30)}$$

$$eq2 = 2a_2 + 4\left(a_0 + 1\right) - 2a_0 \quad \textbf{(3.31)}$$

and

$$eq3 = -2\left(a_2 + a_1\right) - \tfrac{3}{2} a_2\left(\tfrac{1}{4} a_2 - \tfrac{1}{2} a_1 + a_0\right) + 3\left(\tfrac{1}{4} a_2 - \tfrac{1}{2} a_1 + a_0\right) + 4 \quad \textbf{(3.32)}$$

Setting $eq1 = eq2 = eq3 = 0$ and solving the set of simultaneous equations results in:

$$\begin{aligned} a_0 &= -\frac{18}{5} \\ a_1 &= 0 \\ a_2 &= \frac{8}{5} \end{aligned} \qquad \textbf{(3.33)}$$

These coefficients can then be substituted back into Equation (3.27). Figure (3.3) below, illustrates that this simple approximation is surprisingly accurate. Figure (3.4) is the error (exact-approximation) plotted over the domain.

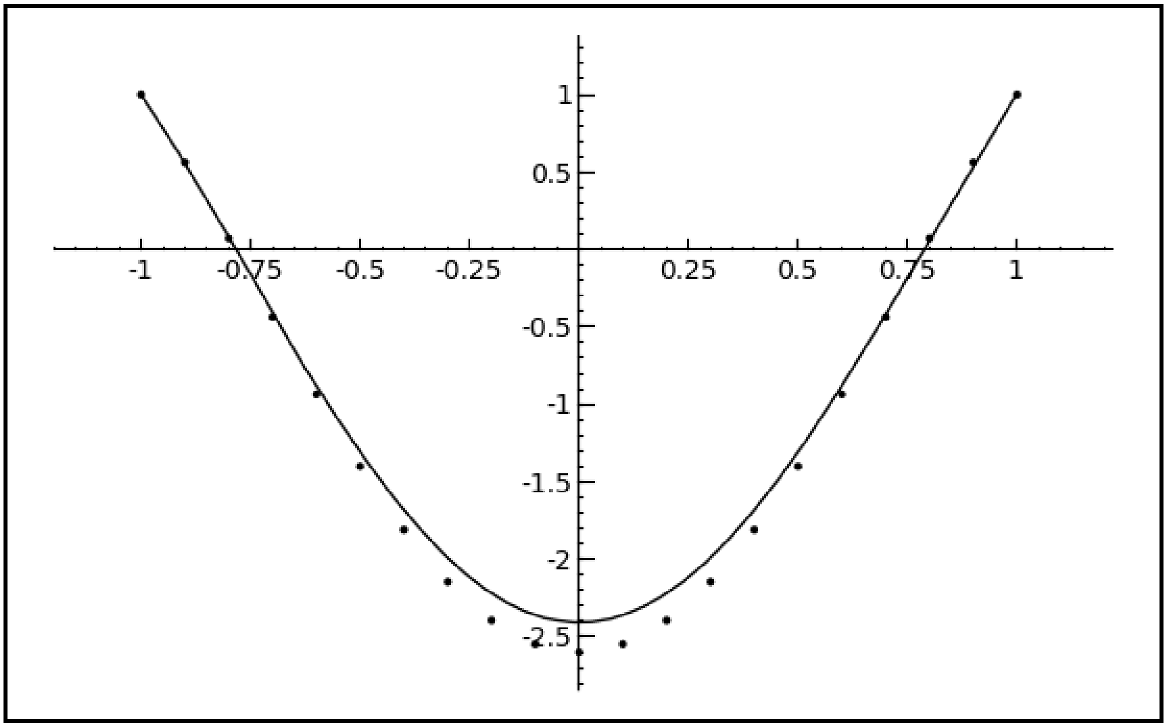

Figure 3.2 Exact Solution (Solid) Compared to the Approximation (Dotted).

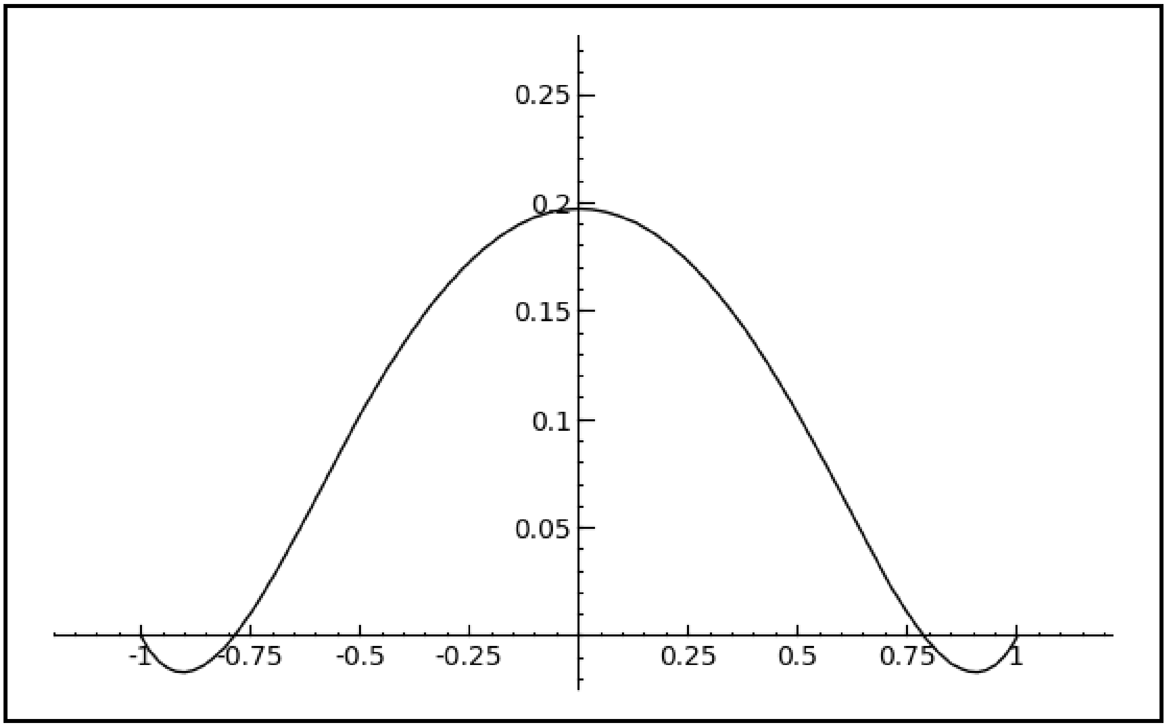

Figure 3.3 Error, Exact-Approximation

Under the same guise as the simple polynomial approximation, the next example employs Spectral Collocation and the Chebyshev derivative matrix. Now instead of solving for the coefficients of the approximation equation the value of the solution at each collocation point is found. The Matlab code implementing this is shown in Listing 3.2.

```
%------------------------------------------------------------
    N = 11;                     % Collocation points
    k = 2;
%------------------------------------------------------------
%   Create the Chebyshev differentiation Matrix
    [D,x] = cheb_derivative(N);
    D2 = D^2;
%------------------------------------------------------------
%   Compute the tensor product spectral grid.
%------------------------------------------------------------
    L = D2 + k^2*eye(N+1);
%------------------------------------------------------------
%   Create a column matrix to hold the RHS of the energy
%   equation
%------------------------------------------------------------
    size = (N+1);
    rhs = zeros(size,1);
%------------------------------------------------------------
%   Apply the Boundary Conditions
%------------------------------------------------------------
%   Find the bounday points of the coordinate vectors:
    b = find(abs(x)==1);
%   Impose the boundary conditions on the right side vector.
    rhs(b) = (x(b)==1).*1 + (x(b)==-1).*1;
%   Apply the boundary conditions to the laplacian
    % Zero the rows out
    L(b,:) = zeros(length(b),length(rhs));
    % put in 1 for known points
    L(b,b) = eye(length(b));
    reshape(rhs,(N+1),1);
%------------------------------------------------------------
%   Solve the Matrix equation
%------------------------------------------------------------
    u = L\rhs;
%------------------------------------------------------------
```

Listing 3.2 Spectral Collocation code for Helmholtz example.

In Figure 3.4 it can be seen that the collocation method is a poor approximation when only 3 points are used. However, with 11 points (Figure 3.6) the method provides an extremely accurate approximation. In fact the accuracy does not really change when the number of points is increased to 41 (Figure 3.8). This is an excellent example of how accurate the collocation method can be with a relatively small number of points.

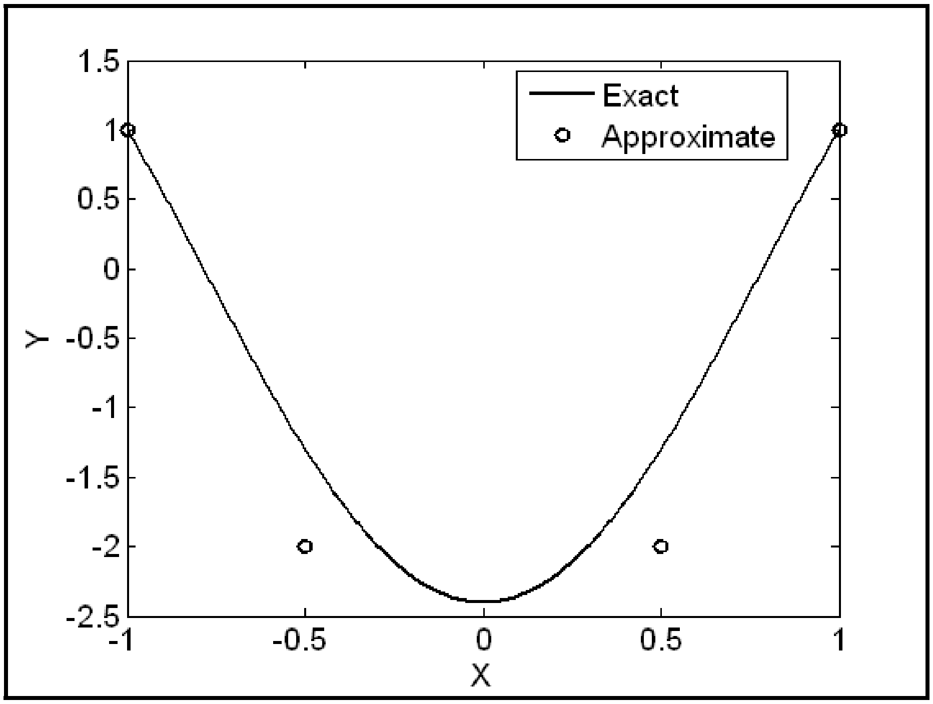

Figure 3.4 Helmholtz approximation (N=3)

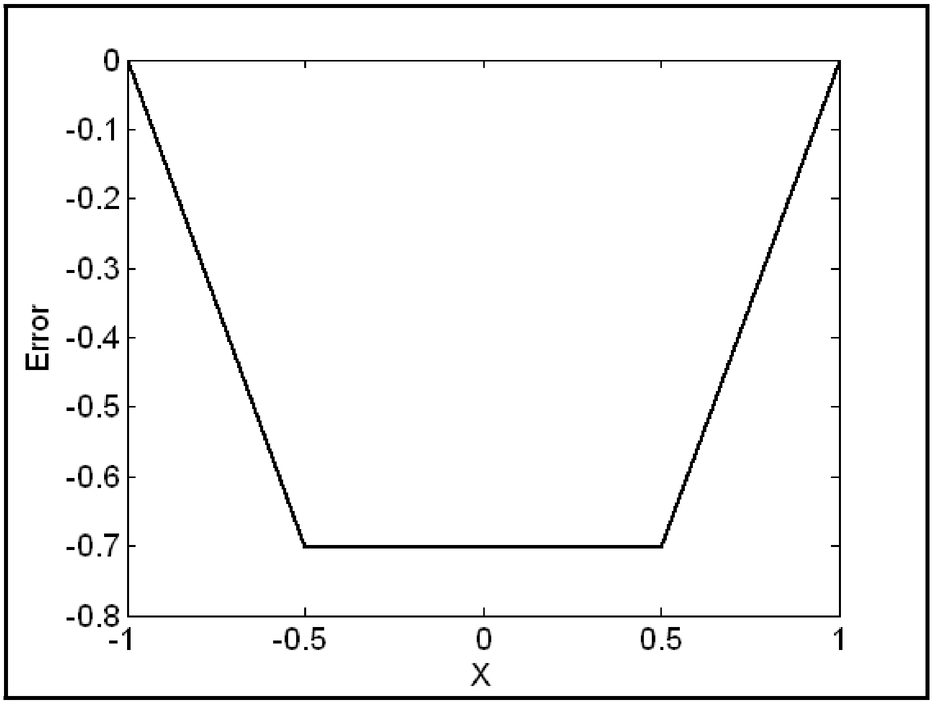

Figure 3.5 Helmholtz approximation Error, Actual-Approximation (N=3)

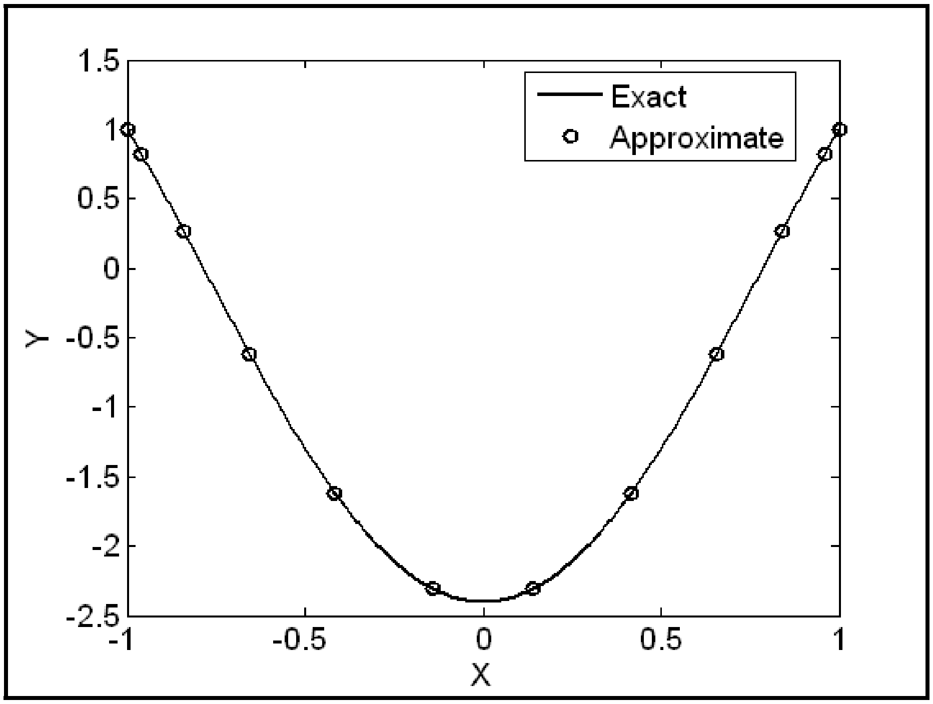

Figure 3.6 Helmholtz approximation (N=11)

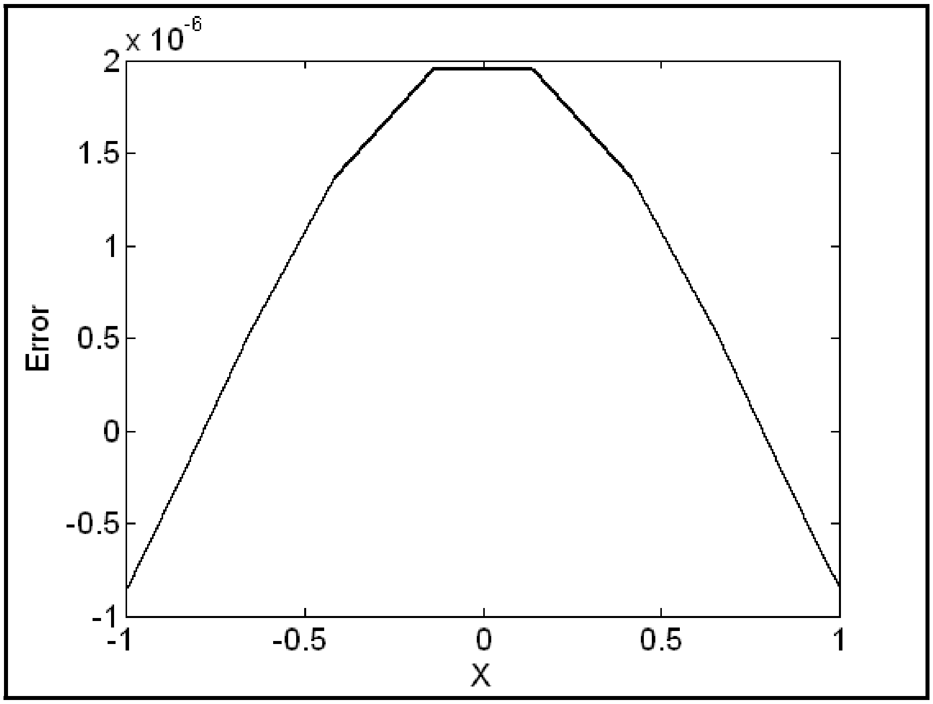

Figure 3.7 Helmholtz approximation Error, Actual-Approximation (N=11)

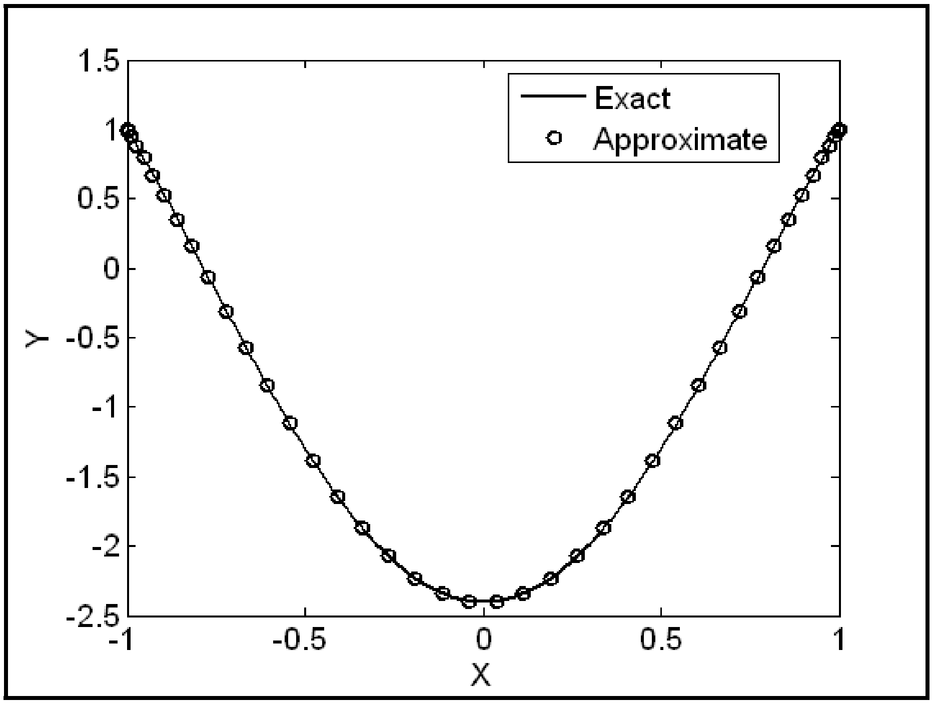

Figure 3.8 Helmholtz approximation (N=41)

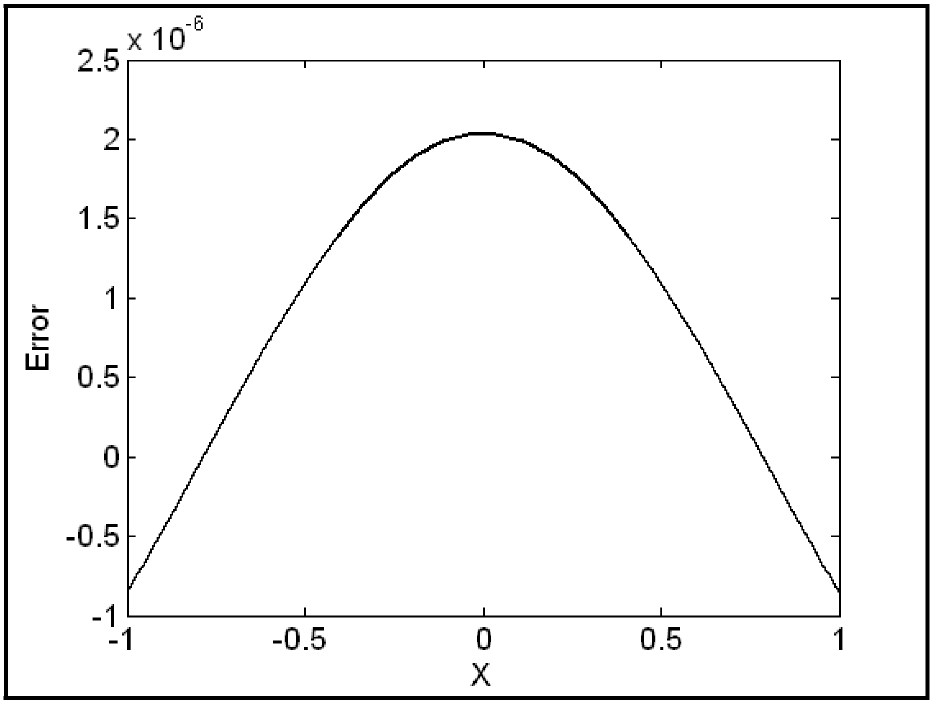

Figure 3.9 Helmholtz approximation Error, Actual-Approximation (N=41)

4. Numerical Methods

4.1 Spectral Discretization: The Simple Graetz Problem

Before discretizing the governing differential equations, the coordinates of the solution domain must be projected onto the computational domain. Since r is already mapped over the domain [-1,1], due to the dimensionless form, z is the only value that needs to be remapped. Beginning with the axial coordinates, z can be mapped using Equation (4.1).

$$z = \eta \frac{L}{2} + \frac{L}{2} \tag{4.1}$$

The dimensionaless coordinate can then be defined as:

$$z^* = \frac{z}{R} = \eta \frac{L}{2R} + \frac{L}{2R} \tag{4.2}$$

Solving for η:

$$\eta = \frac{2R}{L} z^* - 1 \tag{4.3}$$

Differentiating Equation (4.3):

$$\frac{\partial \eta}{\partial z^*} = \frac{2R}{L} \tag{4.4}$$

This can be used to define the derivative $\partial/_{\partial z^*}$:

$$\frac{\partial}{\partial z^*}=\frac{\partial}{\partial \eta}\frac{\partial \eta}{\partial z^*}=\frac{2R}{L}\frac{\partial}{\partial \eta} \tag{4.5}$$

Substituting into Equation (2.11),

$$\frac{2\cdot R\cdot \mathrm{Re}\cdot \mathrm{Pr}}{L}\cdot\left(1-\left(r^*\right)^2\right)\frac{\partial \Theta}{\partial \eta}-\frac{1}{r^*}\frac{\partial \Theta}{\partial r^*}-\frac{\partial^2 \Theta}{\partial {r^*}^2}=0 \tag{4.6}$$

Equation (4.6) can be discretized using Chebyshev differentiation matrices which will result in:

$$\left[\left(A\otimes \tilde{D}^1_{z^*}\right)-\left(\tilde{D}^1_{r^*}.*\Omega\otimes I\right)-\left(\tilde{D}^2_{r^*}\otimes I\right)\right]\{\Theta\}=\{f\} \tag{4.7}$$

Where:

$$\mathrm{A}=diag\left(\frac{2\cdot R\cdot \mathrm{Re}\cdot \mathrm{Pr}}{L}\cdot\left(1-{r^*}^2\right)\right) \tag{4.8}$$

$$\Omega=diag\left(\frac{1}{r^*}\right) \tag{4.9}$$

$$I=eye\left(M_{z^*}\right) \tag{4.10}$$

4.1.1 Spectral Discretization: The Extended Graetz Problem

When the Peclet number is low, axial conduction can no longer be ignored. With the addition of axial conduction Equation (2.11) will become

$$\mathrm{Re}\cdot \mathrm{Pr}\cdot \left(1-\left(r^{*}\right)^{2}\right)\frac{\partial \Theta}{\partial z^{*}}-\frac{\partial^{2}\Theta}{\partial {z^{*}}^{2}}-\frac{1}{r^{*}}\frac{\partial \Theta}{\partial r^{*}}-\frac{\partial^{2}\Theta}{\partial {r^{*}}^{2}}=0 \qquad \textbf{(4.11)}$$

From Equation (4.5) the second derivative can be determined:

$$\frac{\partial^{2}}{\partial {z^{*}}^{2}}=\frac{\partial^{2}}{\partial \eta^{2}}\frac{\partial \eta^{2}}{\partial {z^{*}}^{2}}=\frac{4R^{2}}{L^{2}}\frac{\partial^{2}}{\partial \eta^{2}} \qquad \textbf{(4.12)}$$

Substituting in:

$$\frac{2\cdot R\cdot \mathrm{Re}\cdot \mathrm{Pr}}{L}\cdot\left(1-\left(r^{*}\right)^{2}\right)\frac{\partial \Theta}{\partial \eta}-\left(\frac{2\cdot R}{L}\right)^{2}\frac{\partial^{2}\Theta}{\partial {\eta^{*}}^{2}}-\frac{1}{r^{*}}\frac{\partial \Theta}{\partial r^{*}}-\frac{\partial^{2}\Theta}{\partial {r^{*}}^{2}}=0 \qquad \textbf{(4.13)}$$

As before, Equation (4.13) can be discretized using Chebyshev differentiation matrices resulting in:

$$\left[\left(A\otimes \tilde{D}^{1}_{z^{*}}\right)-\left(B\otimes \tilde{D}^{2}_{z^{*}}\right)-\left(\tilde{D}^{1}_{r^{*}}.*\Omega\otimes I\right)-\left(\tilde{D}^{2}_{r^{*}}\otimes I\right)\right]\{\Theta\}=\{f\} \qquad \textbf{(4.14)}$$

Where:

$$\mathrm{A}=diag\left(\frac{2\cdot R\cdot \mathrm{Re}\cdot \mathrm{Pr}}{L}\cdot\left(1-{r^{*}}^{2}\right)\right) \qquad \textbf{(4.15)}$$

$$\mathrm{B} = \left(\frac{2 \cdot R}{L}\right)^2 \cdot eye\left(N_{r^*}\right) \tag{4.16}$$

$$\Omega = diag\left(\frac{1}{r^*}\right) \tag{4.17}$$

$$I = eye\left(M_{z^*}\right) \tag{4.18}$$

4.1.2 Simple Diffusion

For the simple diffusion model the coordinate transformations must be done in both directions. First, the values for the axial projection will be:

$$z = \frac{L}{2}\eta + \frac{L}{2} \tag{4.19}$$

$$\eta = \frac{2}{L}z - 1 \tag{4.20}$$

$$\frac{\partial \eta}{\partial z} = \frac{2}{L} \tag{4.21}$$

From Equation (4.21) the first partial derivative with respect to z can be found:

$$\frac{\partial}{\partial z} = \frac{\partial}{\partial \eta}\frac{\partial \eta}{\partial z} = \frac{\partial}{\partial \eta}\frac{2}{L} \tag{4.22}$$

Similarly, the second derivative will be:

$$\frac{\partial^2}{\partial z^2} = \frac{\partial^2}{\partial \eta^2}\frac{\partial \eta^2}{\partial z^2} = \frac{\partial^2}{\partial \eta^2}\frac{4}{L^2} \tag{4.23}$$

Likewise, in the radial direction,

$$r = R\xi \tag{4.24}$$

$$\xi = \frac{r}{R} \tag{4.25}$$

$$\frac{\partial \xi}{\partial r} = \frac{1}{R} \tag{4.26}$$

$$\frac{\partial}{\partial r} = \frac{\partial}{\partial \xi}\frac{\partial r}{\partial \xi} = \frac{\partial}{\partial \xi}\frac{1}{R} \tag{4.27}$$

$$\frac{\partial^2}{\partial r^2} = \frac{\partial^2}{\partial \xi^2}\frac{\partial \xi^2}{\partial r^2} = \frac{\partial^2}{\partial \xi^2}\frac{1}{R^2} \tag{4.28}$$

Substituting in the coordinate transforms into Equation (2.25),

$$\frac{2}{L}\frac{\partial n}{\partial \eta}\left[V_{\max}\left(1-\xi^2\right)\right] = D\left(\frac{1}{R^2\xi}\frac{\partial n}{\partial \xi} + \frac{1}{R^2}\frac{\partial^2 n}{\partial \xi^2}\right) \tag{4.29}$$

and rearranging results in

$$\frac{2}{L}\frac{\partial n}{\partial \eta}\left[V_{\max}\left(1-\xi^2\right)\right] - D\left(\frac{1}{R^2\xi}\frac{\partial n}{\partial \xi} + \frac{1}{R^2}\frac{\partial^2 n}{\partial \xi^2}\right) = 0 \tag{4.30}$$

Finally, discretizing

$$\left[\left[A\otimes\tilde{D}_{\eta}^{1}\right]-\left[\left(\Omega\cdot\frac{1}{R^{2}}\cdot\tilde{D}_{\xi}^{(1)}\otimes\tilde{I}\right)D+\left(\frac{1}{R^{2}}\cdot\tilde{D}_{\xi}^{(2)}\otimes\tilde{I}\right)D\right]\right]\{\Theta\}=\{f\} \tag{4.31}$$

Where:

$$\mathrm{A}=diag\left(\frac{2}{L}V_{\max}\left(1-\xi^{2}\right)\right) \tag{4.32}$$

$$\Omega=diag\left(\frac{1}{\xi}\right) \tag{4.33}$$

$$I=eye\left(M_{\eta}\right) \tag{4.34}$$

4.2 Finite Difference Method

4.2.1 The Graetz Problem: Non-dimensional

The problem can be discretized by substituting the approximation Equations (2.14)-(2.16) into Equation (2.11):

$$\begin{aligned}&\mathrm{Re}\cdot\mathrm{Pr}\cdot\left(1-\left(r^{*}\right)^{2}\right)\frac{\Theta(i,j+1)-\Theta(i,j-1)}{2\Delta z^{*}}-\frac{1}{r^{*}}\frac{\Theta(i+1,j)-\Theta(i-1,j)}{2\Delta\,\mathrm{r}^{*}}\\&-\frac{\Theta(i+1,j)-2\Theta(i,j)+\Theta(i-1,j)}{\left(\Delta\,\mathrm{r}^{*}\right)^{2}}=0\end{aligned} \tag{4.35}$$

In this numerical scheme r^{*} can be expressed as $i\cdot\Delta r^{*}-1$. Substituting into Equation (4.35) results in

$$\begin{aligned}&\mathrm{Re}\cdot \mathrm{Pr}\cdot \left(1-\left(i\cdot \Delta r^{*}-1\right)^{2}\right)\frac{\Theta(i,j+1)-\Theta(i,j-1)}{2\Delta z^{*}}-\\&\frac{1}{\left(i\cdot \Delta r^{*}-1\right)}\frac{\Theta(i+1,j)-\Theta(i-1,j)}{2\Delta \mathrm{r}^{*}}-\frac{\Theta(i+1,j)-2\Theta(i,j)+\Theta(i-1,j)}{\left(\Delta \mathrm{r}^{*}\right)^{2}}=0\end{aligned} \tag{4.36}$$

rearranging,

$$\begin{aligned}&\left(\Delta \mathrm{r}^{*}\right)^{2}\mathrm{Re}\cdot \mathrm{Pr}\cdot \left(1-\left(i\cdot \Delta r^{*}-1\right)^{2}\right)\frac{\Theta(i,j+1)-\Theta(i,j-1)}{2\Delta z^{*}}\\&-\frac{\left(\Delta \mathrm{r}^{*}\right)}{2\cdot \left(i\cdot \Delta r^{*}-1\right)}\left(\Theta(i+1,j)-\Theta(i-1,j)\right)-\left(\Theta(i+1,j)-2\Theta(i,j)+\Theta(i-1,j)\right)=0\end{aligned} \tag{4.37}$$

The final result is shown in Equation (4.38). It is this difference equation that the software will satisfy at each node point.

$$\Theta(i,j)=\frac{1}{2}\left[\begin{aligned}&-\frac{\left(\Delta \mathrm{r}^{*}\right)^{2}}{\Delta z^{*}}\frac{\mathrm{Re}\cdot \mathrm{Pr}}{2}\cdot \left(1-\left(i\cdot \Delta r^{*}-1\right)^{2}\right)\cdot \left(\Theta(i,j+1)-\Theta(i,j-1)\right)\\&+\frac{\left(\Delta \mathrm{r}^{*}\right)}{2\left(i\cdot \Delta r^{*}-1\right)}\left(\Theta(i+1,j)-\Theta(i-1,j)\right)+\Theta(i+1,j)+\Theta(i-1,j)\end{aligned}\right] \tag{4.38}$$

For the problem modeled here a simple Jacobi iteration method is used. This can be thought of as a brute force point-by-point solution to the finite-difference problem. The discretized physical domain results in a two-dimensional array of nodal points. The differential equation is satisfied at each node by considering the effect of the four neighboring nodes. The method solves the difference equation at each iteration until the system converges (i.e. a specified tolerance between a node point about successive iterations is satisfied). The code implementing this solution can be found in Listing 4.1.

```
function Simulate_Tube()
    % Input Variables
    Re = 1; Pr = 1; R = 1; L = 0.5;
    % Grid Setup
    N = 21; M = 21;
    Grid = zeros(M+1,N+1);
    %  Apply Boundary Conditions
    Grid(1,1:N) = 0; Grid(M,1:N) = 0;
    Grid(2:M,1) = 1; Grid(1:M,N) = 0;
    Grid_Old = Grid;
    iterations = 0;
    flag = 0; ESPI = 1e-9;
    del_r = 2/M; del_z = (L/R)/N;
    while flag == 0
        flag = 1;
        % Calculate the enry Region.
        for i=2:M
            A = -((del_r^2)/del_z)* ...
                  ((Re*Pr)/2)* ...
                  (1-((i-1)*del_r-1)^2);
            B = del_r/(2*((i-1)*del_r-1));
            for j = 2:N
                Grid(i,j) = 0.5*(A*(Grid_Old(i,j+1)- ...
                                    Grid_Old(i,j-1))+ ...
                                 B*(Grid_Old(i+1,j)- ...
                                    Grid_Old(i-1,j))+ ...
                                    Grid_Old(i-1,j)+ ...
                                    Grid_Old(i+1,j));
                if(((Grid(i,j)-Grid_Old(i,j))-ESPI) > 0)
                    flag = 0;
                end;
            end;
        end;
        Grid_Old = Grid;
        iterations = iterations + 1;
        if iterations > 1000
            flag = 1;
            'Error: increments > 1000'
        end;
    end;
```

Listing 4.1 Finite Difference Code for Step change in tube flow.

4.3 Weighted Average Temperature

In order to make a comparison with published results and other numerical methods, some standard calculations are required.

The weighted average temperature can be calculated using Equation (4.39).

$$\overline{T} = \frac{\int_0^{2\pi}\int_0^{R} T(r,z) r dr d\theta}{\int_0^{2\pi}\int_0^{R} r dr d\theta} \tag{4.39}$$

$$\overline{T} = \frac{2\int_0^{R} T(r,z)\cdot r\cdot dr}{R^2} \tag{4.40}$$

This is the average temperature at an axial location along the length of the pipe that will weight the influence each point has on the average based on the spatial distribution of the points.

4.4 Bulk Fluid Temperature

The bulk fluid temperature is often referred as the cup-mixing temperature. It is the temperature that would be obtain if the fluid passing a cross-section was collected in a cup and mixed together. This is defined in terms of the thermal energy transport, which can be reached by integrating internal energy multiplied by the mass flux over the cross sectional area. This is defined below in Equation (4.41).

$$\dot{E}_T = \int_{A_C} \rho\, u c_v T dA_C \tag{4.41}$$

The mean temperature is then defined by Equation (4.42).

$$\dot{E}_T = \dot{m} c_v T_m \tag{4.42}$$

Then Equation (4.46) will result.

$$T_m = \frac{\int_{A_C} \rho\, u c_v T dA_C}{\dot{m} c_v} \tag{4.43}$$

Using the axisymmetric conditions and the non-dimensional values, Equation (4.44) is obtained.

$$\Theta_m = \frac{\int_0^1 T\left(r^*, Z\right) v_z\left(r^*\right) r^* dr^*}{\int_0^1 T\left(r^*, 0\right) v_z\left(r^*\right) r^* dr^*} \tag{4.44}$$

Where:

$$v_z\left(r^*\right) = V_{\max}\left(1 - {r^*}^2\right)$$

$$n(r^*, 0) = 1$$

Substituting and rearranging results in:

$$\Theta_m = \frac{V_{\max}\int_0^1 T\left(r^*, Z\right)\left(r^* - r^{*3}\right) dr^*}{V_{\max}\big/4} \tag{4.45}$$

$$\Theta_m = 4\int_0^1 T\left(r^*, Z\right)\left(r^* - r^{*3}\right) dr^* \tag{4.46}$$

4.5 Particle Penetration Primitive Variables

The particle concentration at a given cross-section along the axis of the tube can be found using Equation (4.47).

$$P = \frac{\int_0^R n(r,Z)\rho(r,Z)v(r,Z)2\pi r dr}{\int_0^R n(r,0)\rho(r,0)v(r,0)2\pi r dr} \tag{4.47}$$

This can be reduced to:

$$P = \frac{2\pi V_{\max}\rho\int_0^R n(r,Z)\left(1-\left(\frac{r}{R}\right)^2\right) r dr}{2\pi V_{\max} n_{in}\rho\int_0^R \left(1-\left(\frac{r}{R}\right)^2\right) r dr} \tag{4.48}$$

$$P = \frac{\int_0^R n(r,Z)\left(r - \frac{r^3}{R^2}\right)dr}{n_{in}\int_0^R \left(r - \frac{r^3}{R^2}\right)dr} \qquad \textbf{(4.49)}$$

$$P = \frac{\int_0^R n(r,Z)\left(r - \frac{r^3}{R^2}\right)dr}{n_{in}\left(\frac{R^2}{2} - \frac{R^4}{4R^2}\right)} \qquad \textbf{(4.50)}$$

$$P = \frac{\int_0^R n(r,Z)\left(r - \frac{r^3}{R^2}\right)dr}{\frac{n_{in}R^2}{4}} \qquad \textbf{(4.51)}$$

5. Results and Discussion

5.1 The Graetz Problem

The results of the classic Graetz problem are compared below with three different solutions, the aforementioned series and finite difference solutions, as well as a set of published values for the benchmark solution. Figures 5.1-5.3 are example plots of the spectral approximation to the Graetz problem. Figures 5.4 and 5.5 are the bulk fluid temperature along the length of the tube over different Peclet numbers.

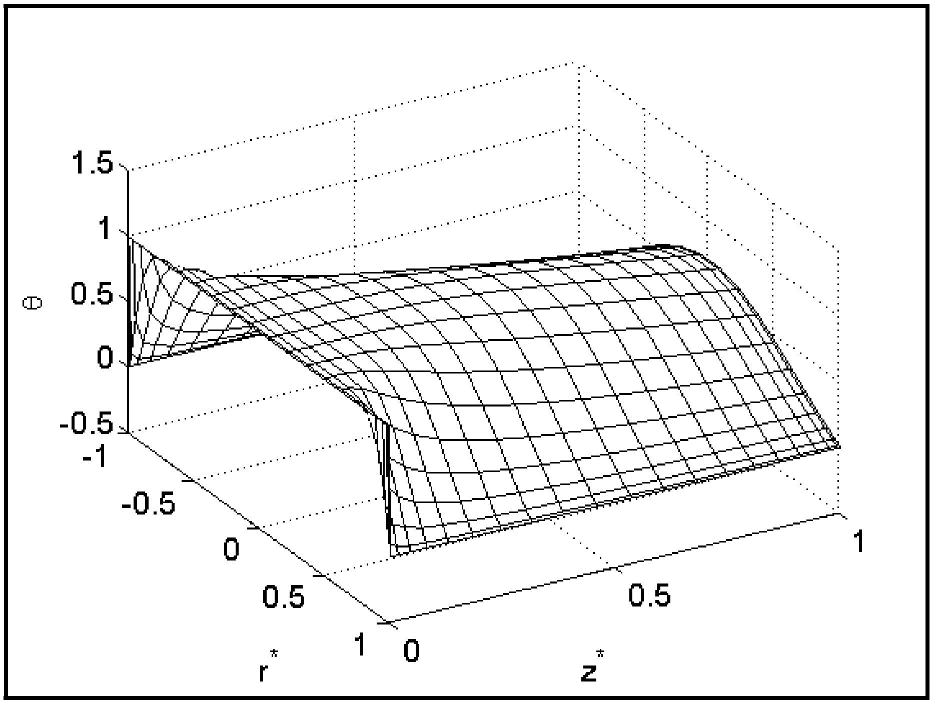

Figure 5.1 Temperature Distribution; Re = 10; Pr = 1; R = 1 m; Length = 1 m 21x21 grid.

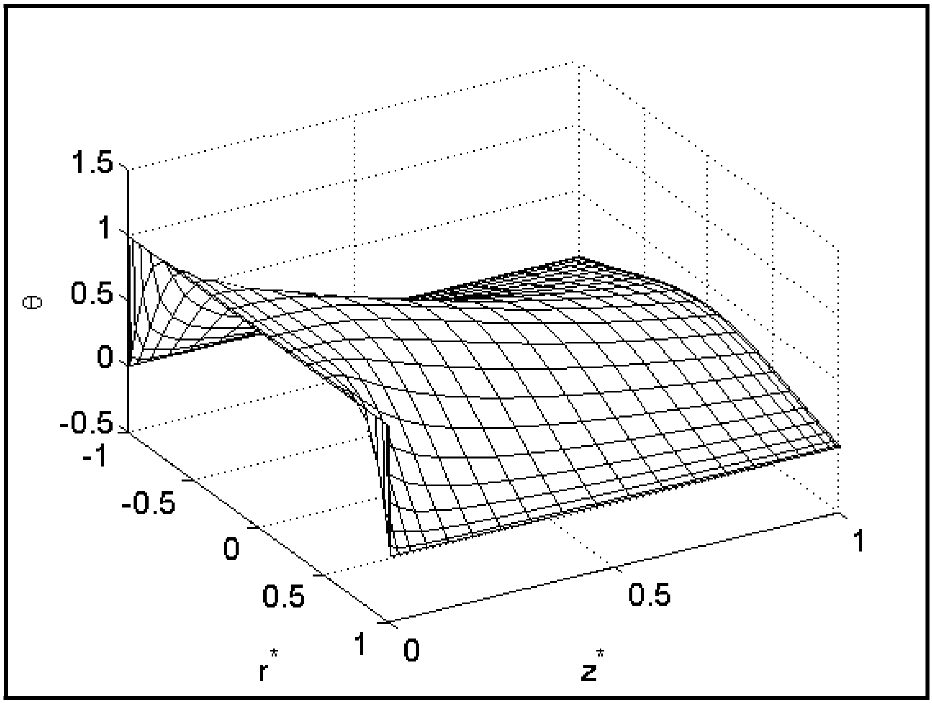

Figure 5.2 Temperature Distribution; Re = 5; Pr = 1; R = 1 m; Length = 1 m 21x21 grid.

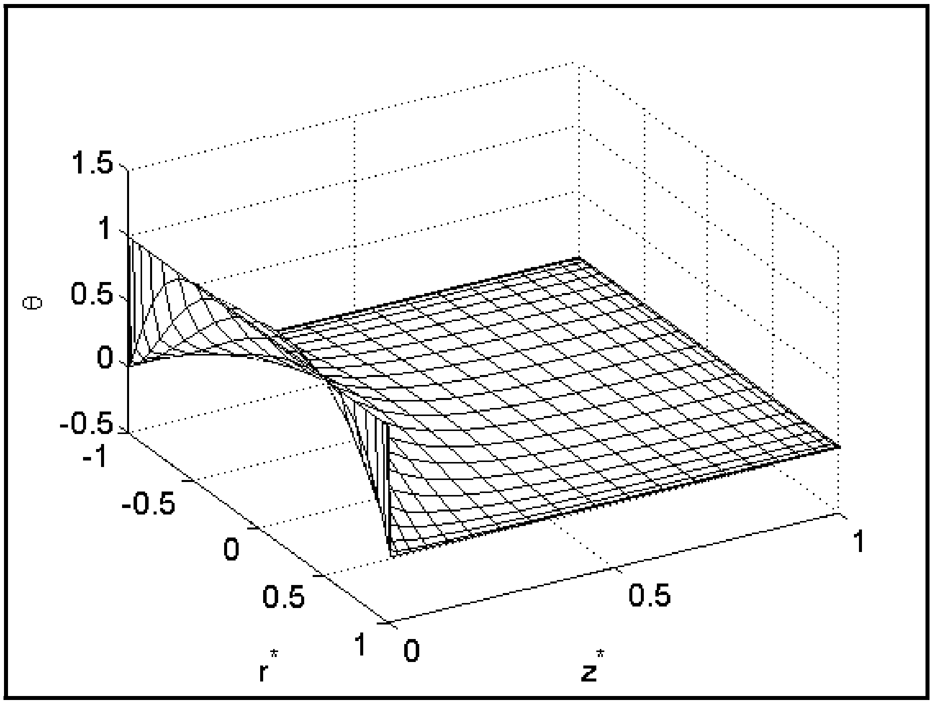

Figure 5.3 Temperature Distribution; Re = 1; Pr = 1; R = 1 m; Length = 1 m 21x21 grid.

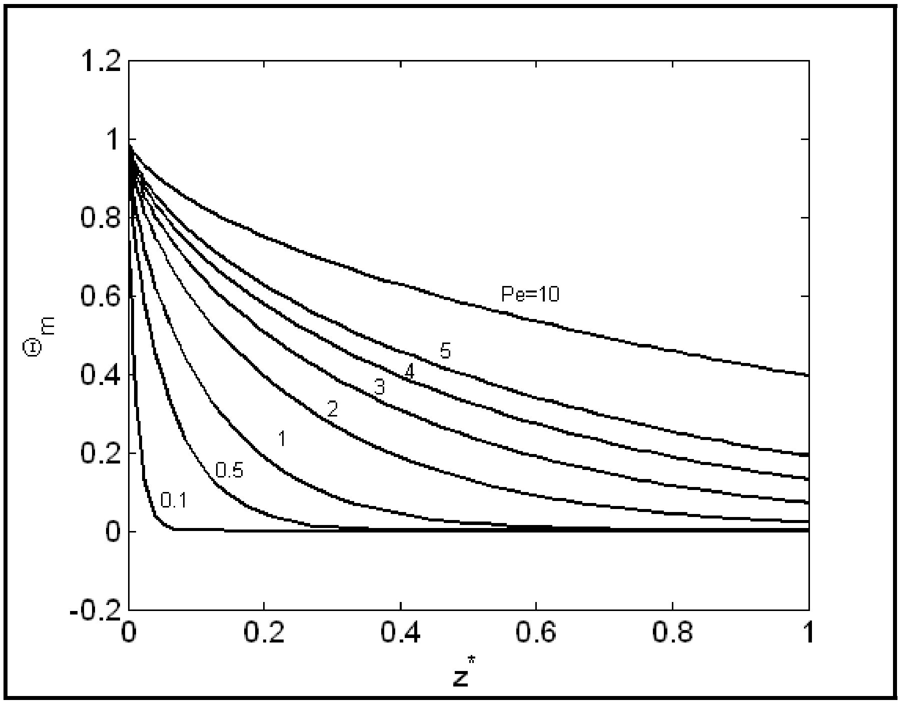

Figure 5.4 Graetz Solution for different values of Pe, labeled to the right of the respective line. Length = 1m; Radius = 1m, Pr = 1; 69x69 point grid.

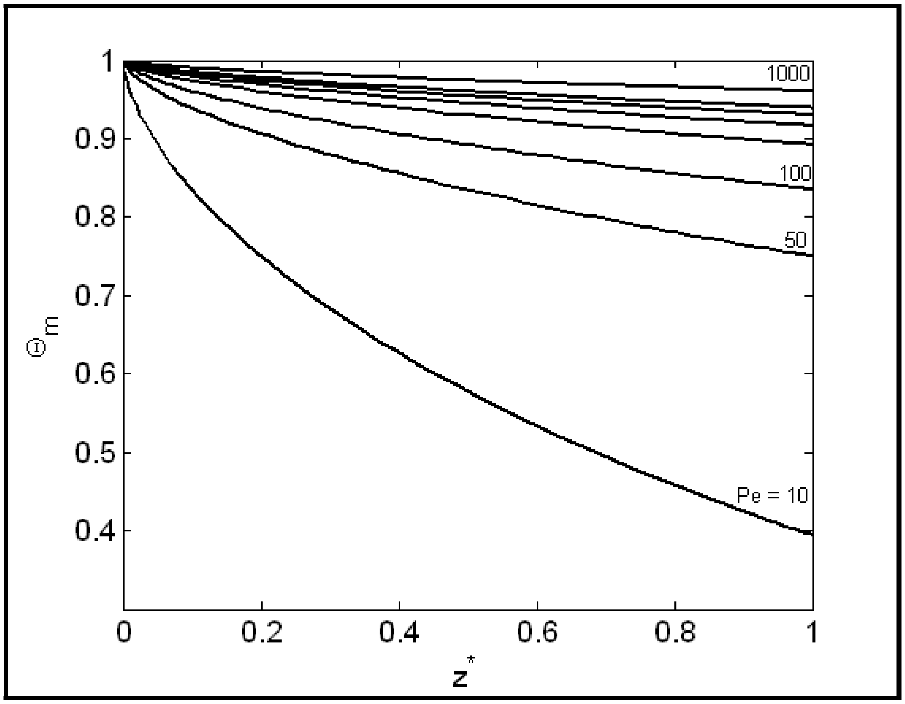

Figure 5.5 Graetz Solution for different values of Pe, labeled above the respective line. The missing values are Pe = 200,300,400 and 500 respectively. Length = 1m; Radius = 1m, Pr = 1; 69x69 point grid.

5.2 Comparison with Graetz Series Solution

The Graetz problem code was first validated using the Graetz series solution from Section 2.2.3. The results are shown below for various values of Re and Pe along a 69x69 point grid. The figures are accompanied by a plot of the difference between the two methods. The inlay of the plot is the difference in the entrance region. This is summarized in Table 1 Below. Notice that the maximum difference between the two methods is constant throughout the parameter space examined here. This is due to the inaccuracy of the Graetz series solution within the entrance region as explained above.

Table 1: Summary of results of comparison to graetz series solution.

Tube Length (m)	Tube Radius (m)	Pr	Re	Pe	θ_m Max Diff.	θ_m Ave Diff.
1	1	1	0.1	0.1	0.011	0.0002
			0.5	0.5	0.011	0.0002
			1	1	0.011	0.00021
			2	2	0.011	0.00021
			3	3	0.011	0.00025
			4	4	0.011	0.00032
			5	5	0.011	0.00041

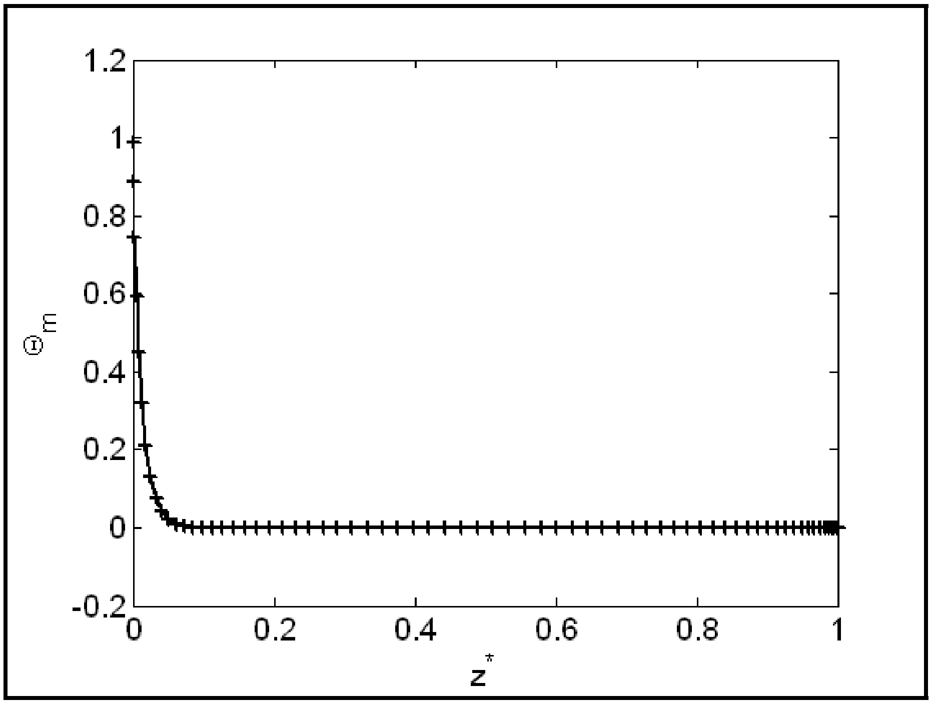

Figure 5.6 Spectral (--) compared to Graetz (+) Re:0.1; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 0.1;

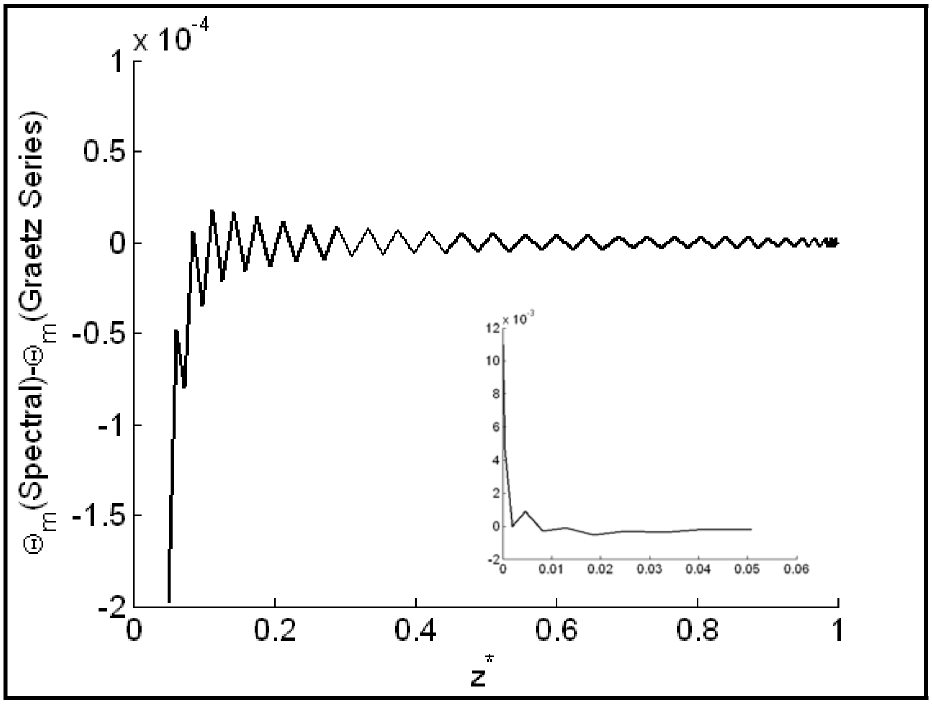

Figure 5.7 Re:0.1; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 0.1;

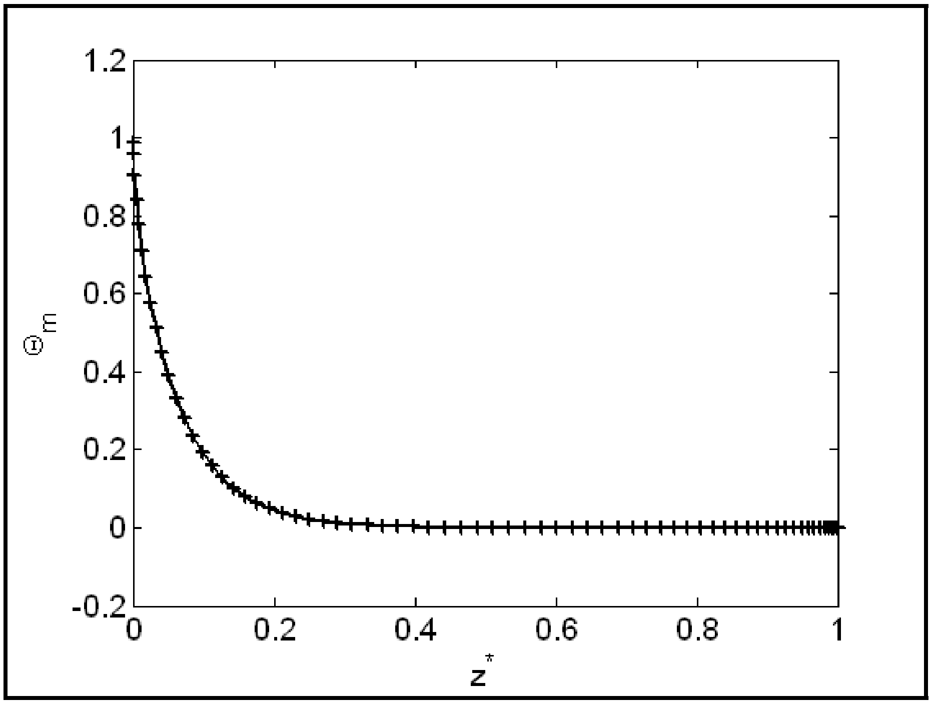

Figure 5.8 Spectral (--) compared to Graetz (+) Re:0.5; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 0.5;

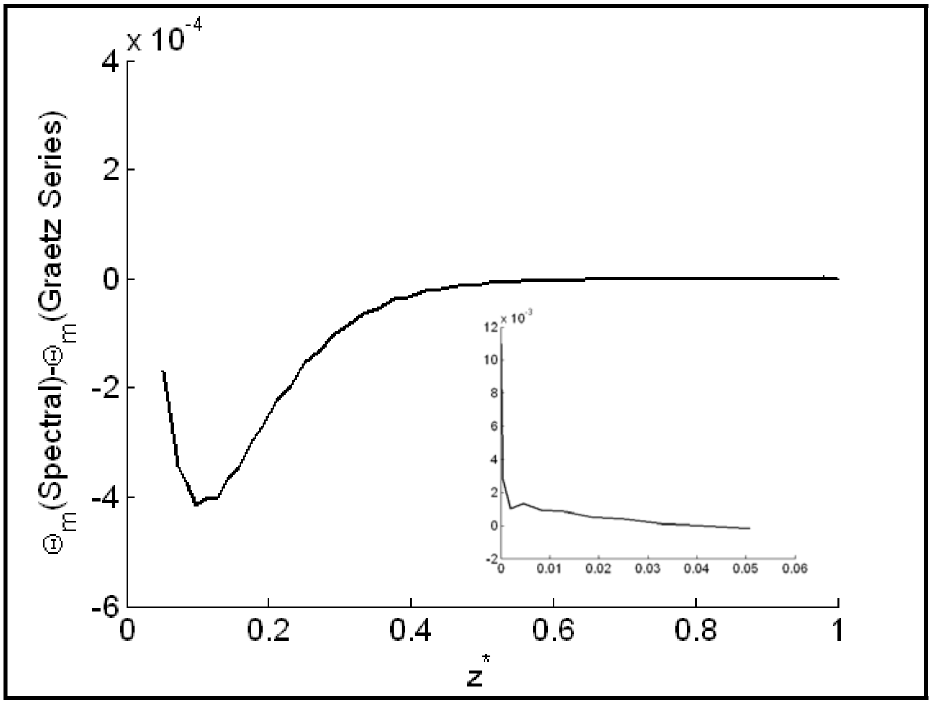

Figure 5.9 Re:0.5; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 0.5;

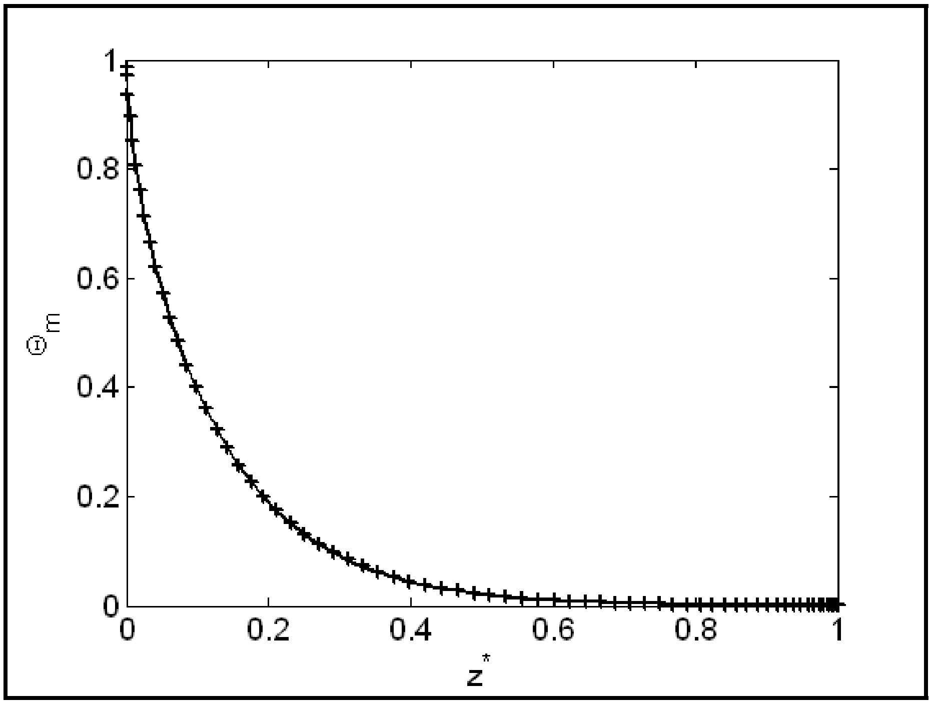

Figure 5.10 Spectral (--) compared to Graetz (+) Re:1.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 1.0;

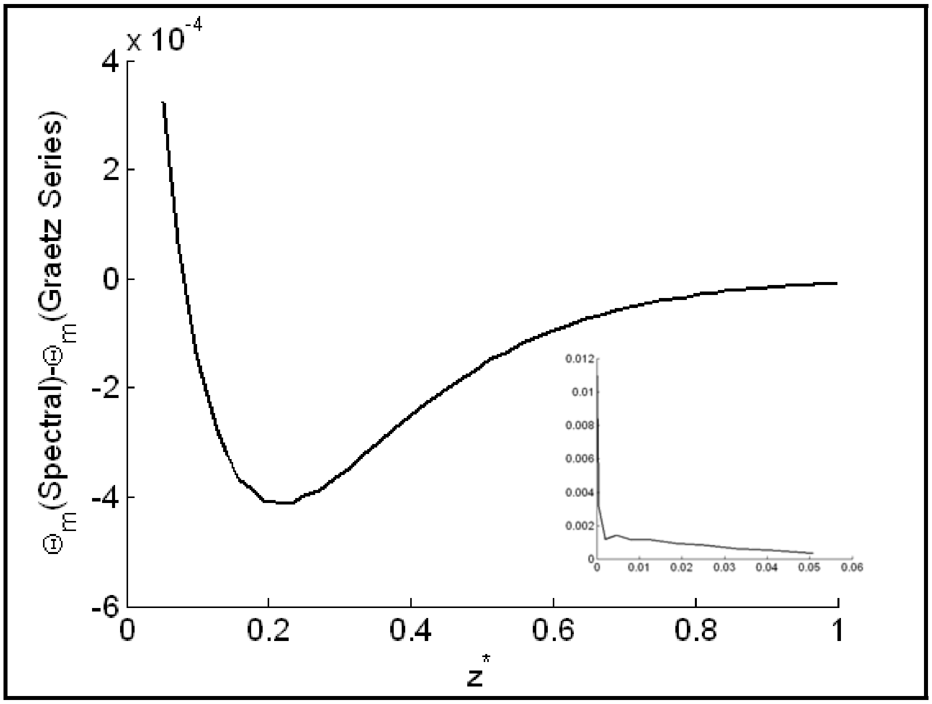

Figure 5.11 Re:1.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 1.0;

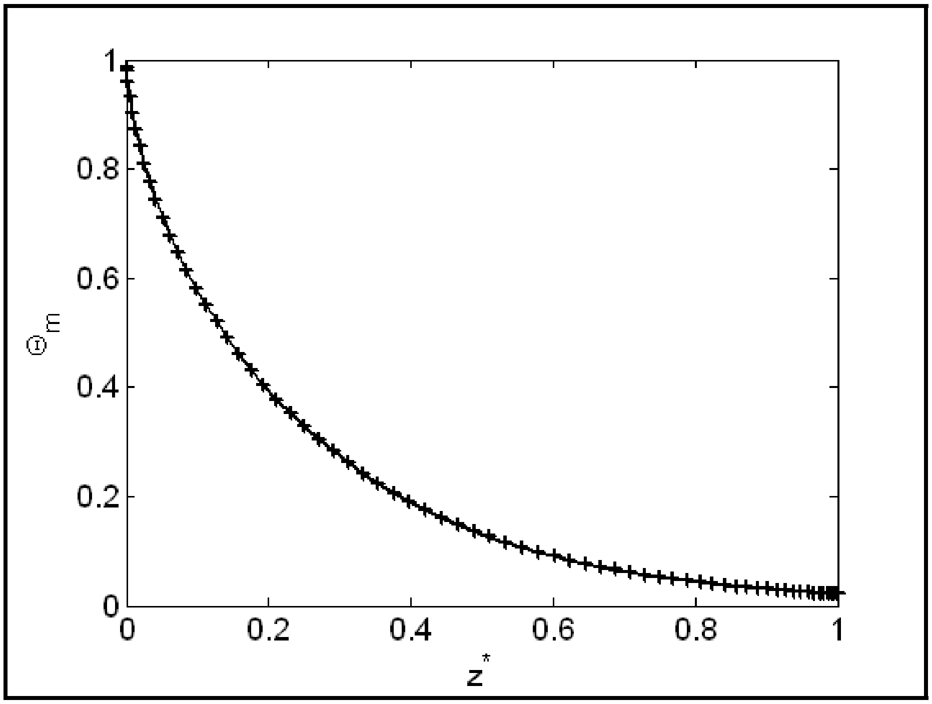

Figure 5.12 Spectral (--) compared to Graetz (+) Re:2.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 2.0;

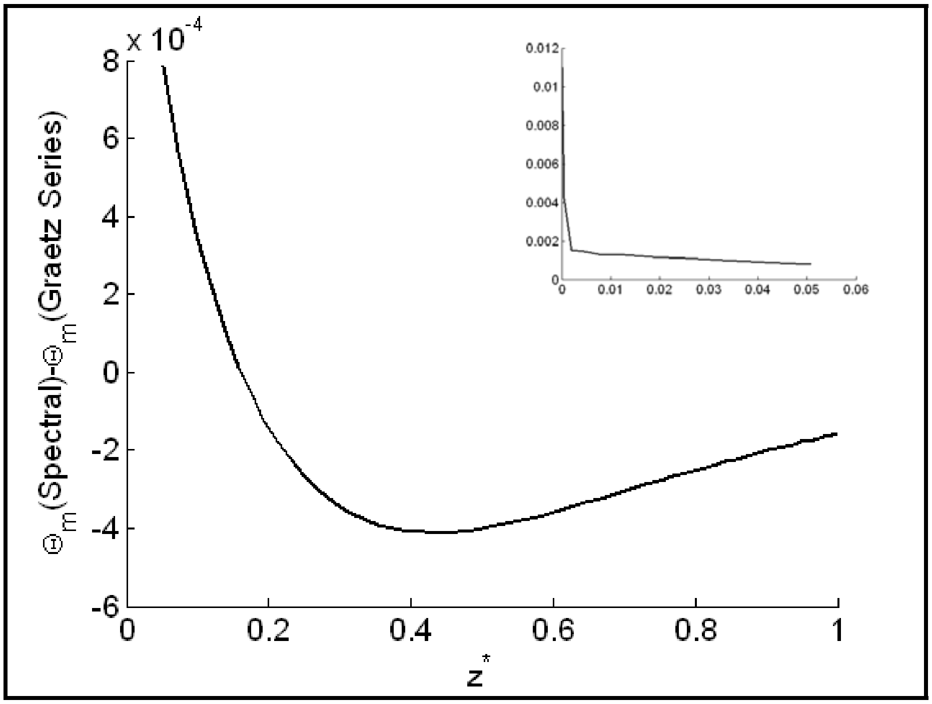

Figure 5.13 Re:2.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 2.0;

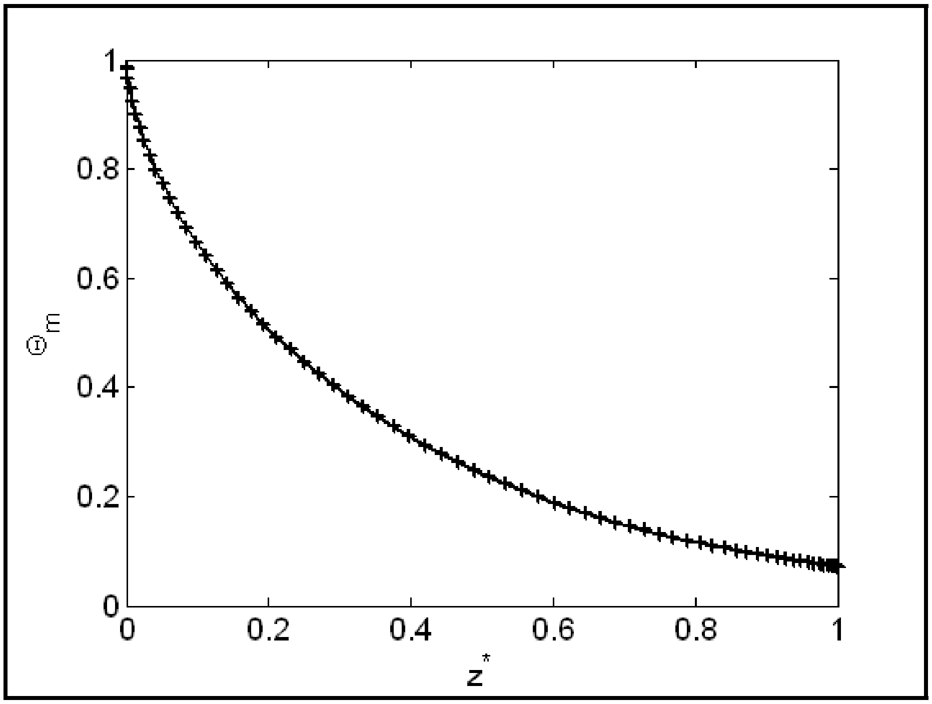

Figure 5.14 Spectral (--) compared to Graetz (+) Re:3.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 3.0;

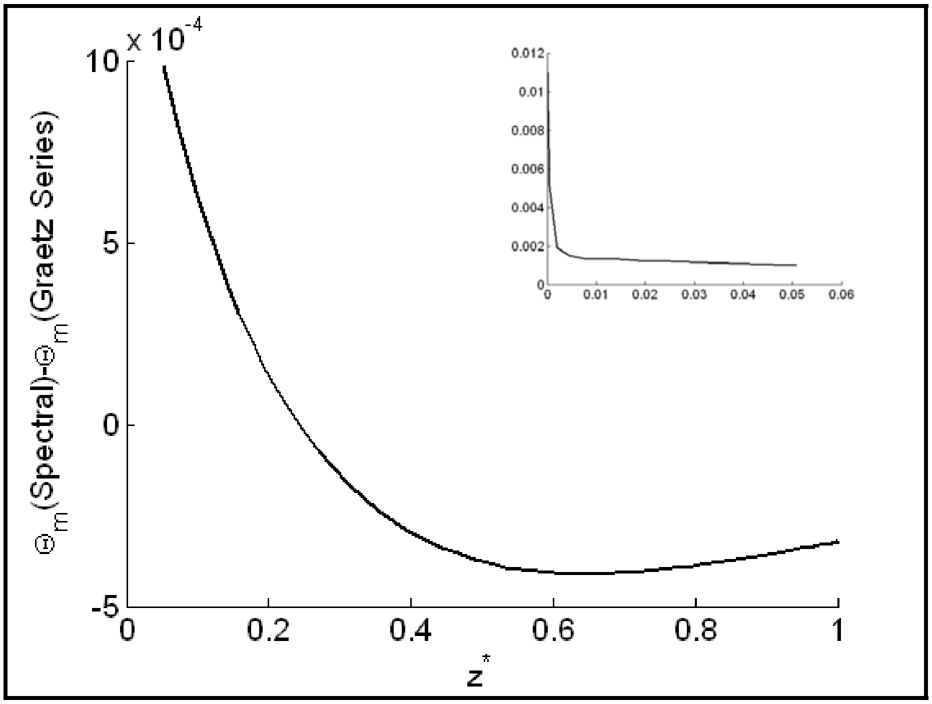

Figure 5.15 3.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 3.0;

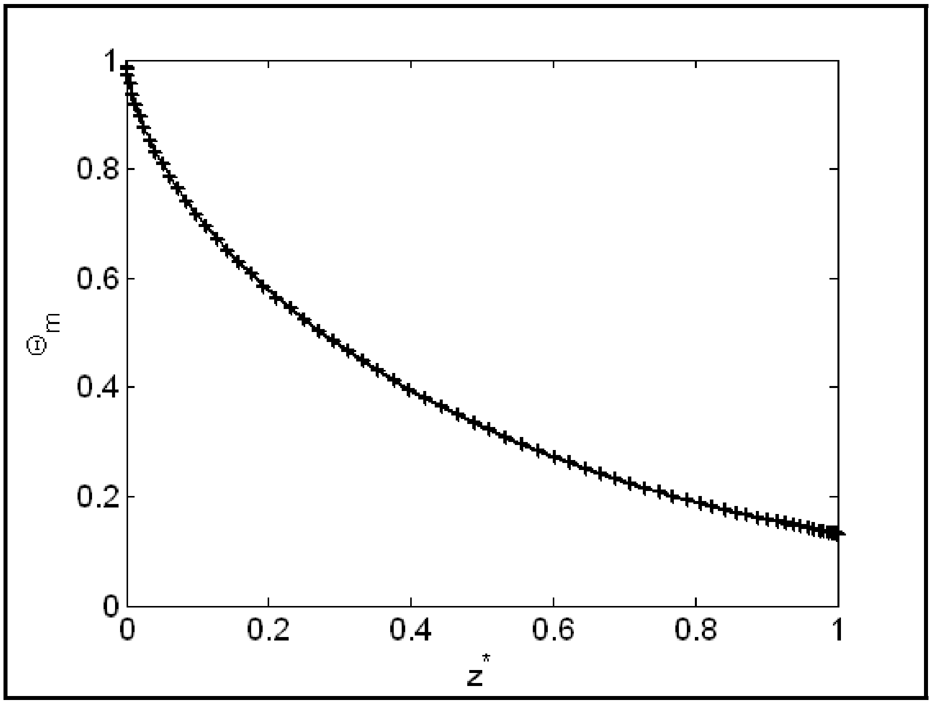

Figure 5.16 Spectral (--) compared to Graetz (+) Re:4.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 4.0;

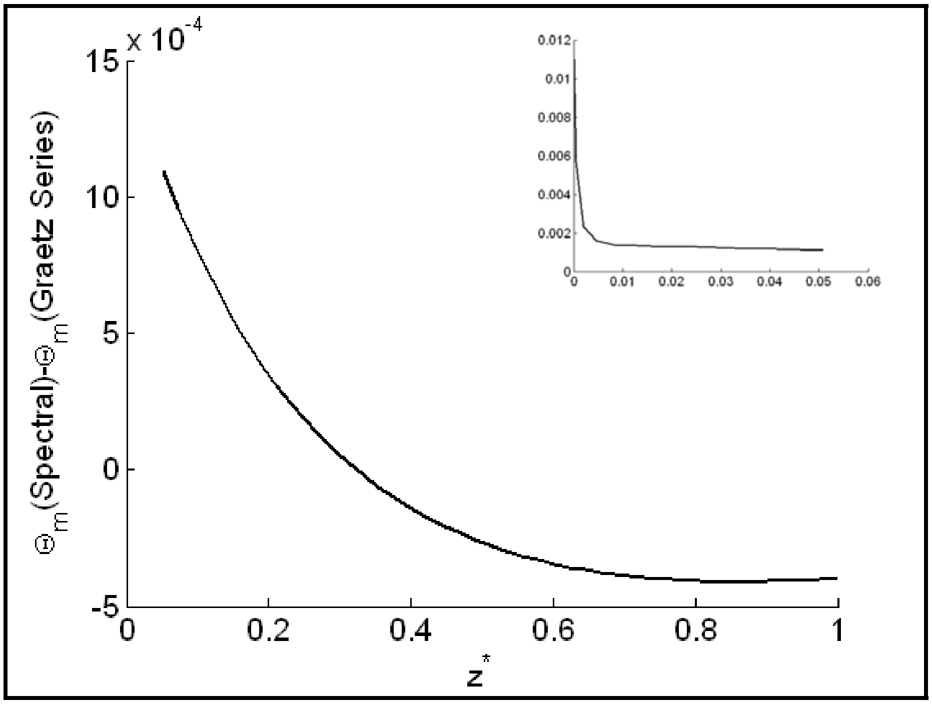

Figure 5.17 Re:4.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 4.0;

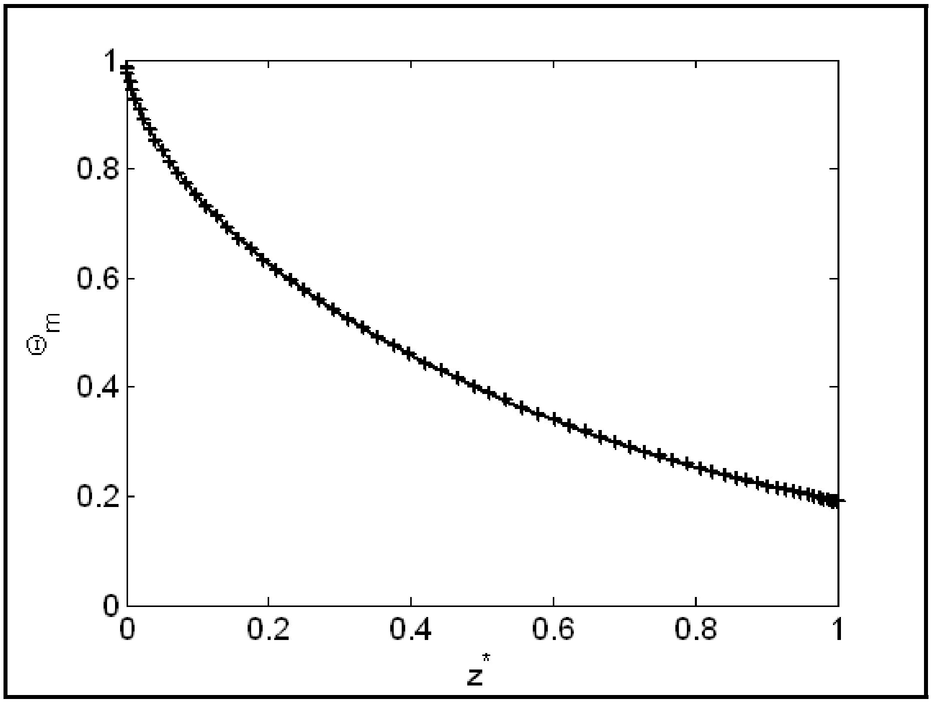

Figure 5.18 Spectral (--) compared to Graetz (+) Re:5.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 5.0;

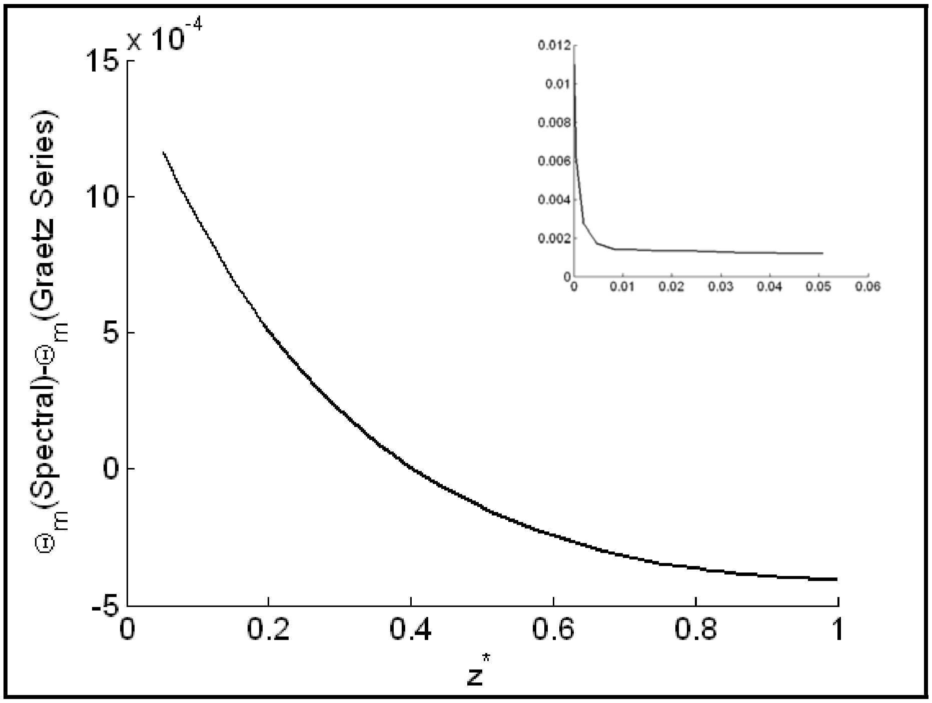

Figure 5.19 Re:5.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 5.0;

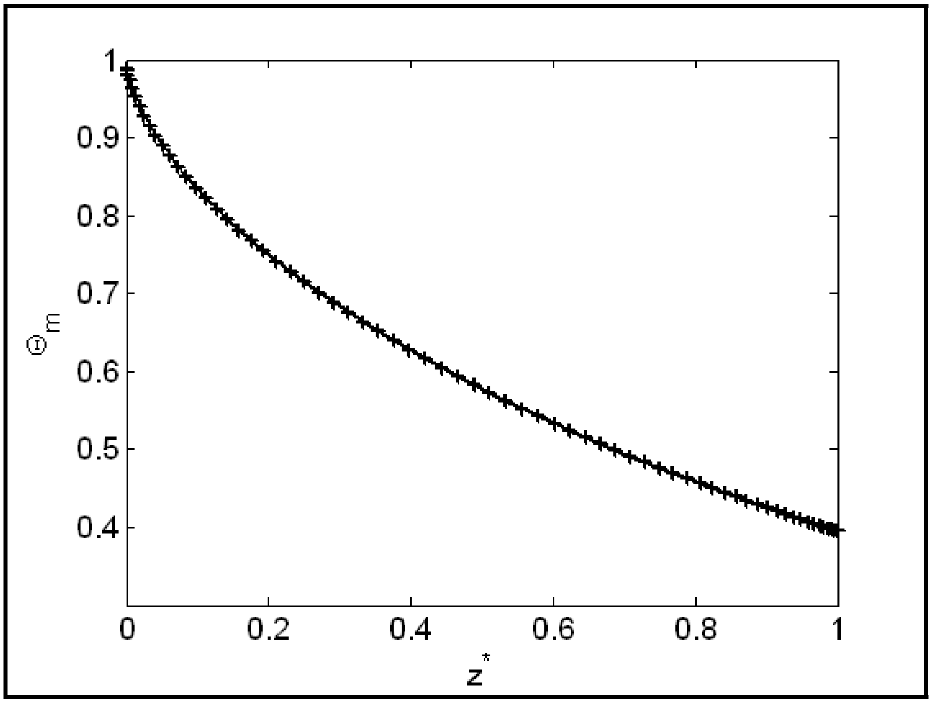

Figure 5.20 Spectral (--) compared to Graetz (+) Re:10.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 10.0;

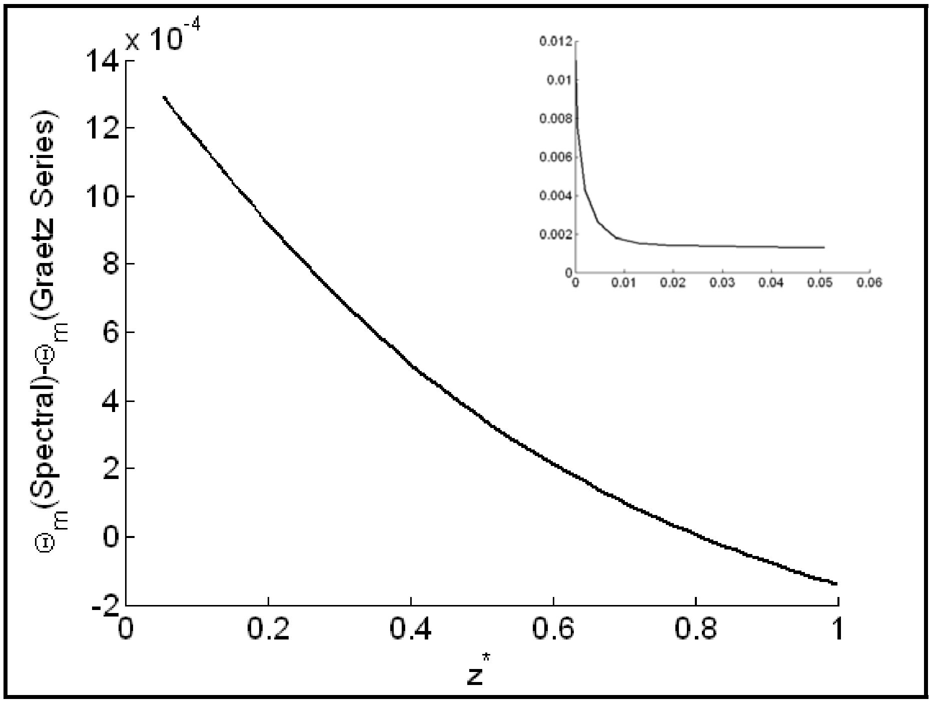

Figure 5.21 Re:10.0; Pr:1.0; R:1.0 m; Length: 1.0 m; Pe: 10.0;

5.3 Comparison with Gutti [38]

Figure 5.22 is a comparison with Figure 1 of [38]. This is a contrast of the series solution with the spectral method over several dimensionless axial points (listed above the lines). Note that a cubic spline interpolation was used on the spectral solution in order to calculate the values at the given dimensionless axial points.

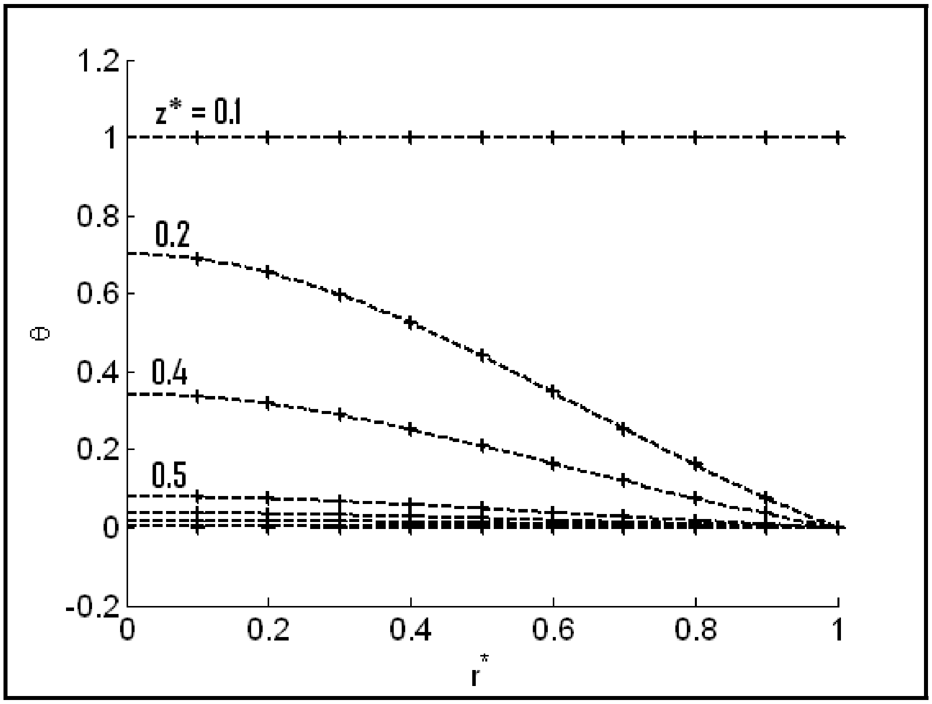

Figure 5.22 Temperature Profile at different values of z^*, Spectral solution(-), Series Solution (.)

5.4 Comparison to Finite Difference

As an example of the efficiency of the Spectral Method, a comparison with the finite difference method was completed. As mentioned before, a Jacobi solution method was used with the finite difference discretization. This is certainly not the most efficient method but it gives a good indication of the relative performance of the Spectral Method, even with a small number of nodes. Figure 5.23 plots the results of a 31x31 spectral grid and a 51x51 finite difference grid. The Spectral Method took 0.65 seconds. Whereas the finite difference method took 1.31 seconds. The results match reasonably well with the maximum average temperature difference of 0.05. Note that here the dimensionless weighted average temperature is computed, not the bulk temperature as before. The results of the comparison are presented in Table 2.

Table 2: Finite Difference Comparison Summary

Spectral		**Finite Difference**		
Grid Size	**Compute Time (s)**	**Grid Size**	**Compute Time (s)**	**Maximum Difference**
31x31	0.65	51x51	1.31	0.05
31x31	0.65	61x61	2.62	0.05
31x31	0.65	71x71	4.63	0.02

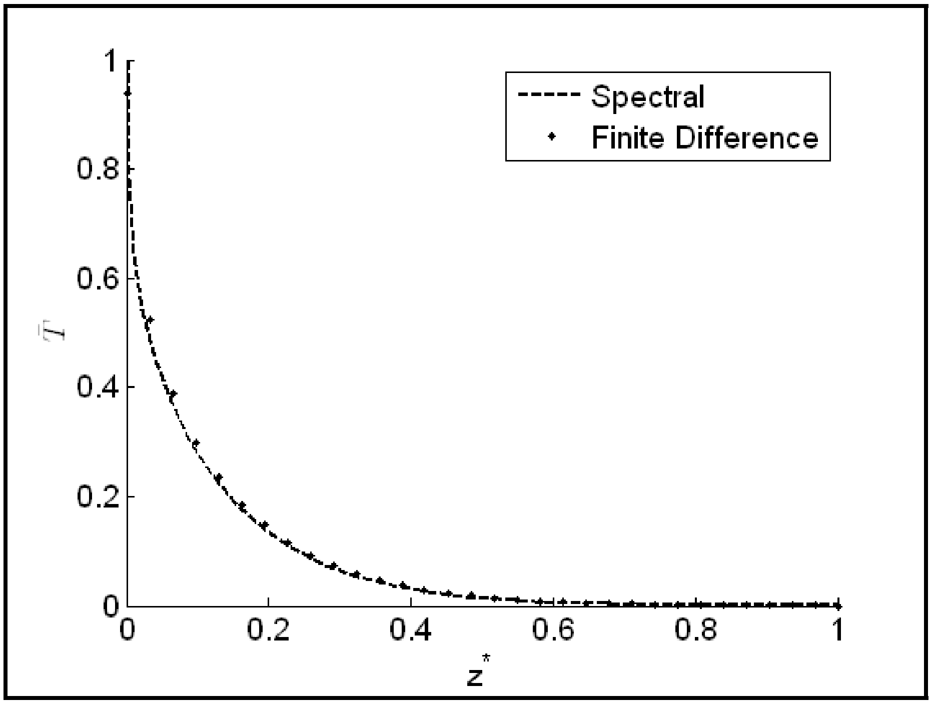

Figure 5.23 Weighted Average Temperature Re:1.0; Pr:1.0; R:1.0 m; Tube Length: 1.0 m;

Figure 5.24 is a similar comparison for a 31x31 spectral grid compared to a 61x61 finite difference grid. The spectral simulation took 0.65 seconds and the finite difference method took 2.62 seconds. With the increased grid size the maximum difference was still 0.05.

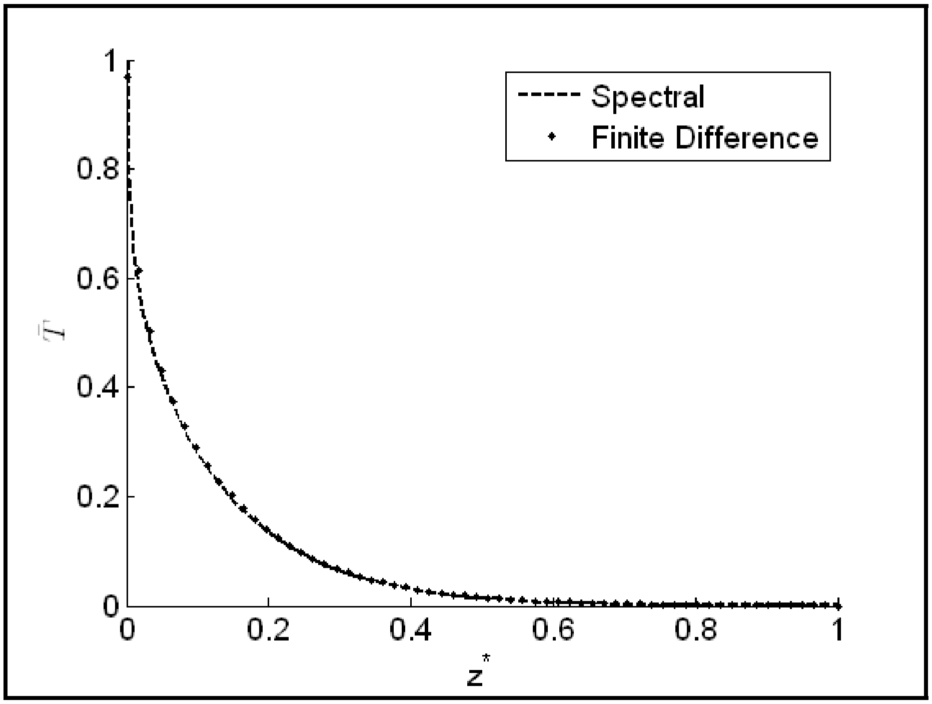

Figure 5.24 Weighted Average Temperature Re:1.0; Pr:1.0; R:1.0 m; Tube Length: 1.0 m;

After increasing the Finite Difference grid to 71x71 nodes the maximum difference is reduced to 0.02. However, the simulation time took 4.63 seconds compared to the 0.65 seconds for the spectral method. This is plotted below in Figure 5.25.

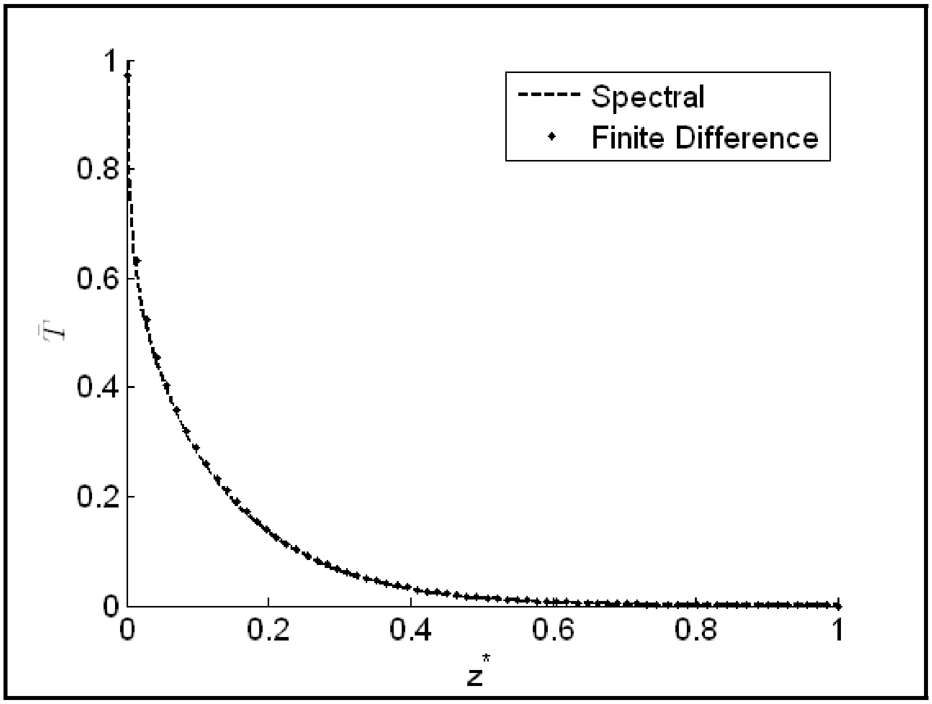

Figure 5.25 Weighted Average Temperature Re:1.0; Pr:1.0; R:1.0 m; Tube Length: 1.0 m.

5.5 Extended Graetz Problem

When the Peclet number is low $(Pe < 10)$ axial conduction becomes a significant heat transport mechanism. To illustrate that, a comparison between the Spectral Method with axial conduction and the Graetz series solution is presented for low Peclet values. Notice as the Peclet number increases and convection dominates, the two methods begin to agree.

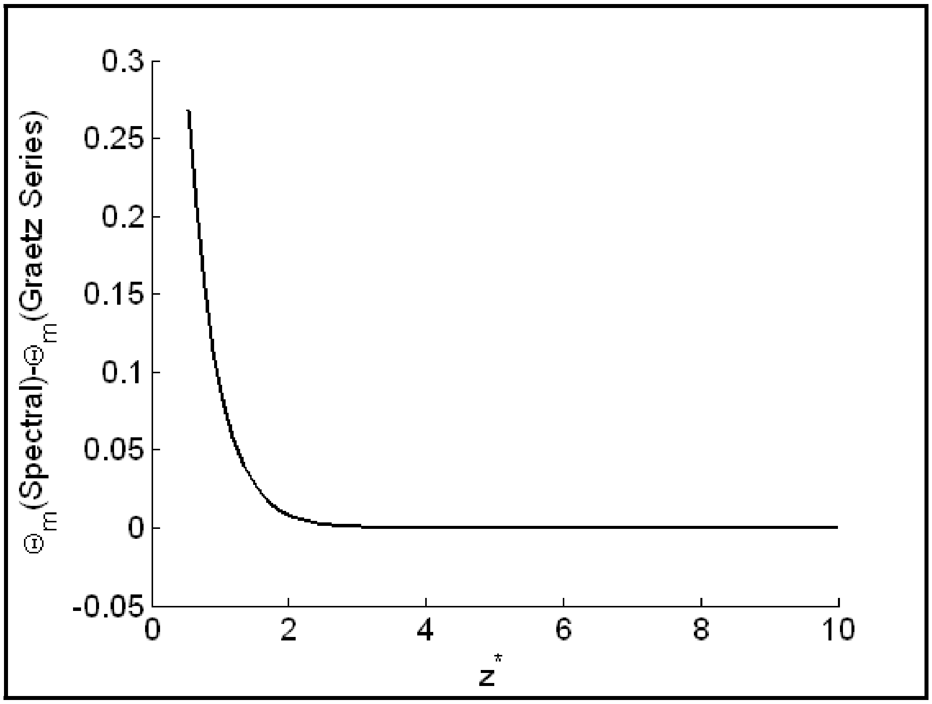

Figure 5.26 Spectral (-) compared to Graetz (+)Re:0.1; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 0.1; 67x67 spectral grid.

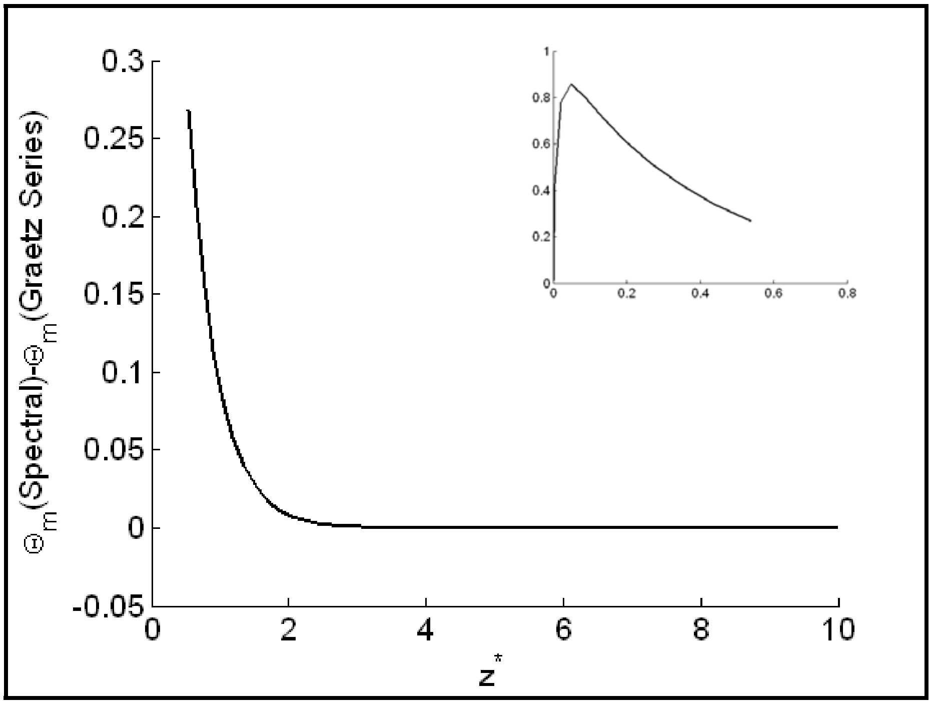

Figure 5.27 Difference between Spectral and Graetz methods, note that the entrance region is in the inlay. Re:0.1; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 0.1;

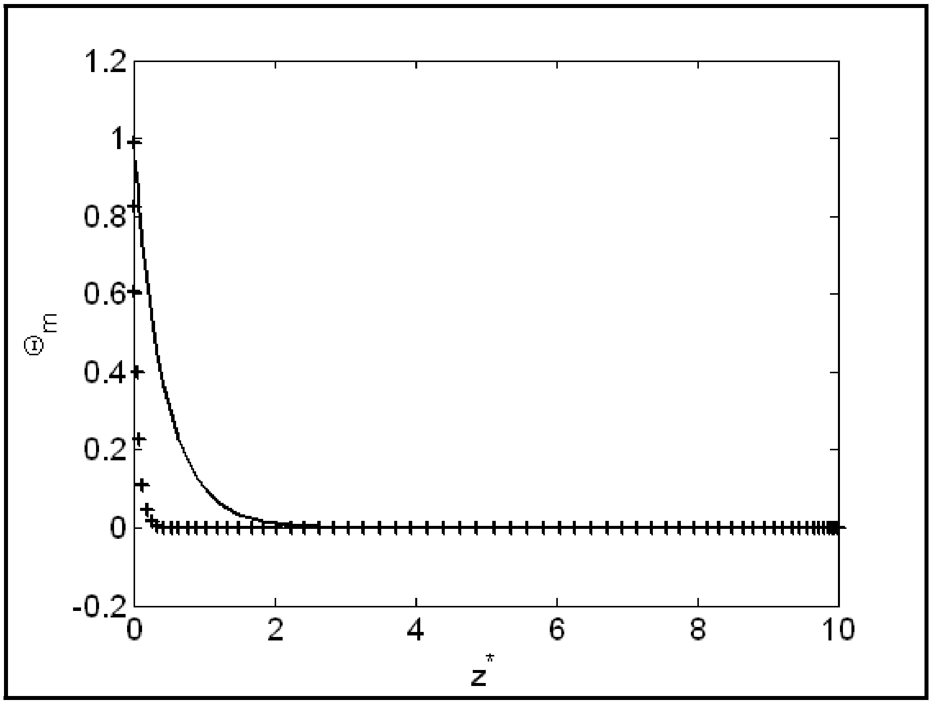

Figure 5.28 Spectral (-) compared to Graetz (+)Re:0.5; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 0.5; 67x67 spectral grid.

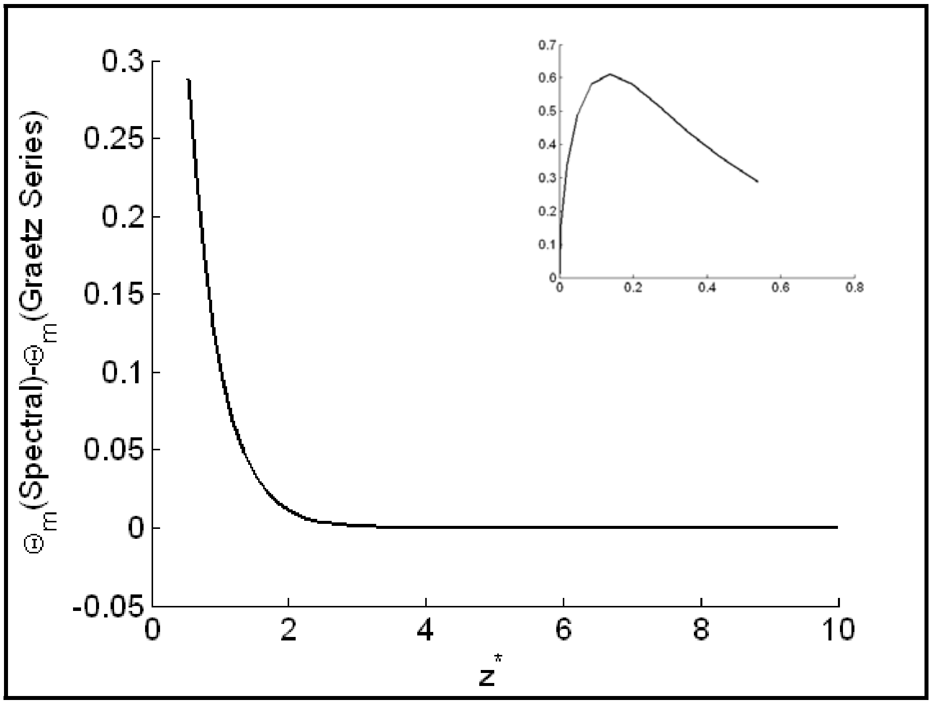

Figure 5.29 Difference between Spectral and Graetz methods, note that the entrance region is in the inlay. Re:0.5; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 0.5;

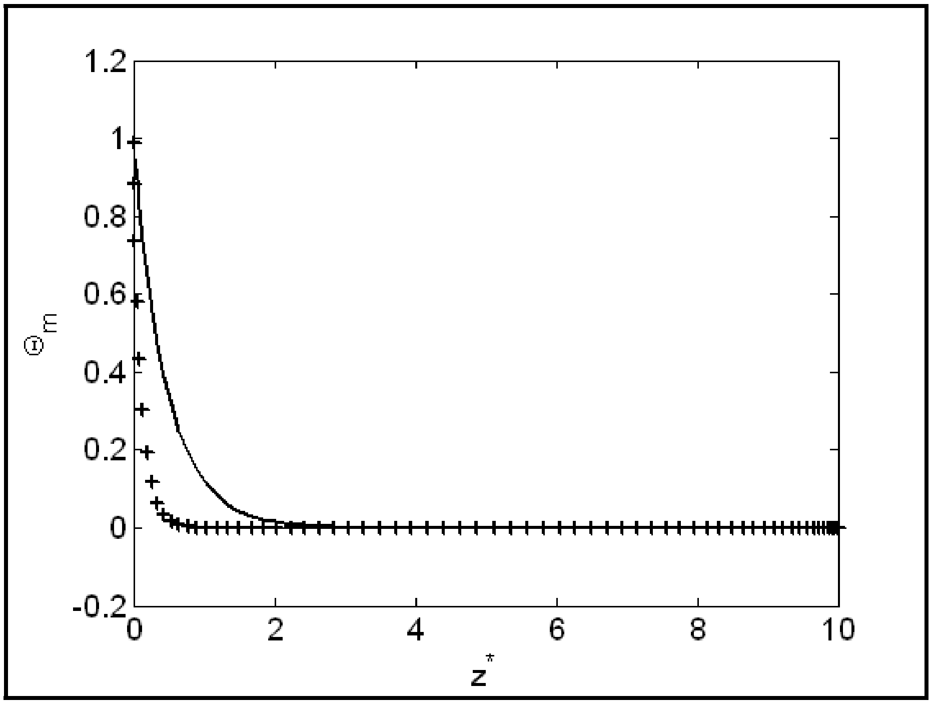

Figure 5.30 Spectral (-) compared to Graetz (+)Re:1.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 1.0; 67x67 spectral grid.

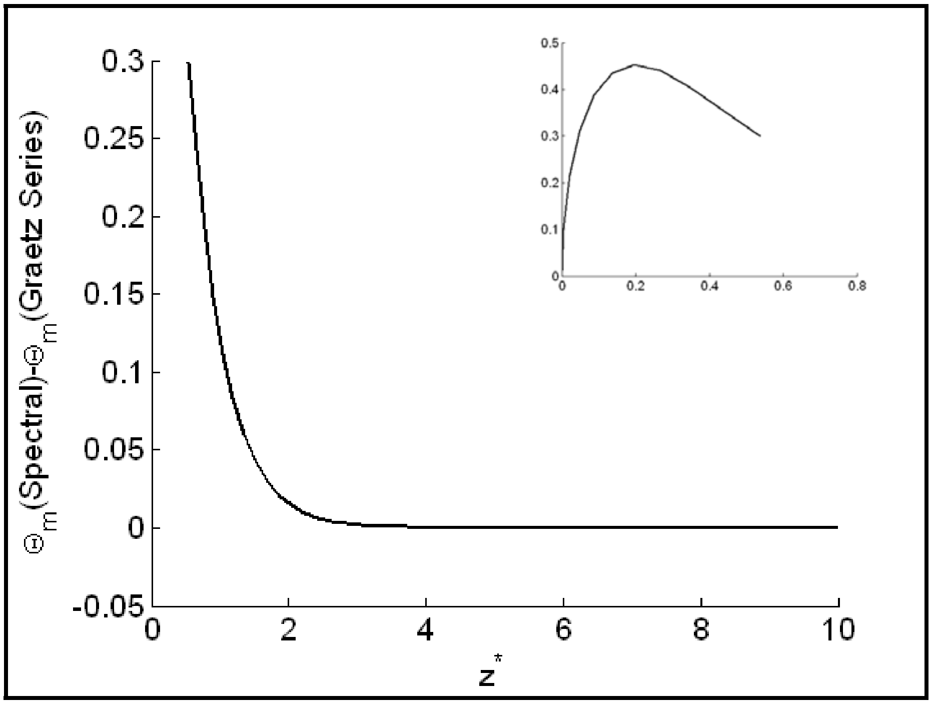

Figure 5.31 Difference between Spectral and Graetz methods, note that the entrance region is in the inlay. Re:1.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 1.0;

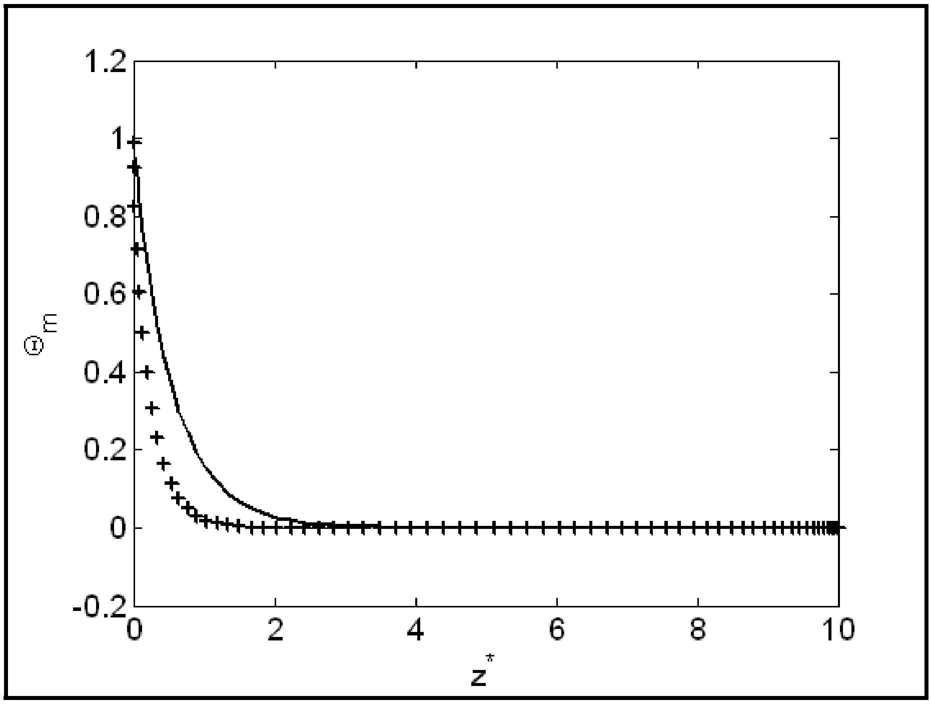

Figure 5.32 Spectral (-) compared to Graetz (+)Re:2.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 2.0; 67x67 spectral grid.

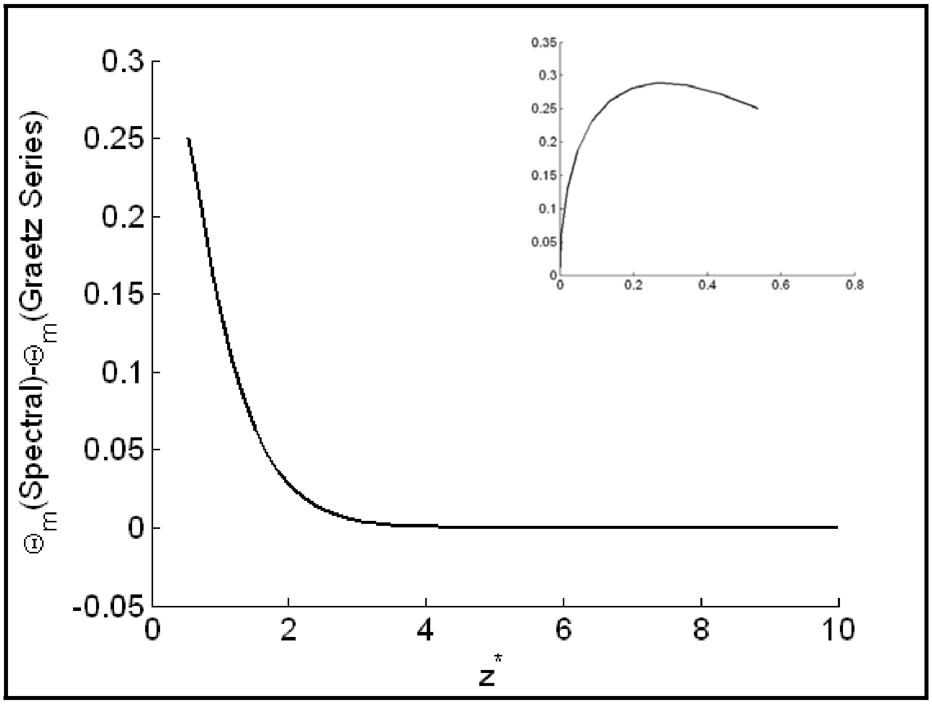

Figure 5.33 Difference between Spectral and Graetz methods, note that the entrance region is in the inlay. Re:2.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 2.0;

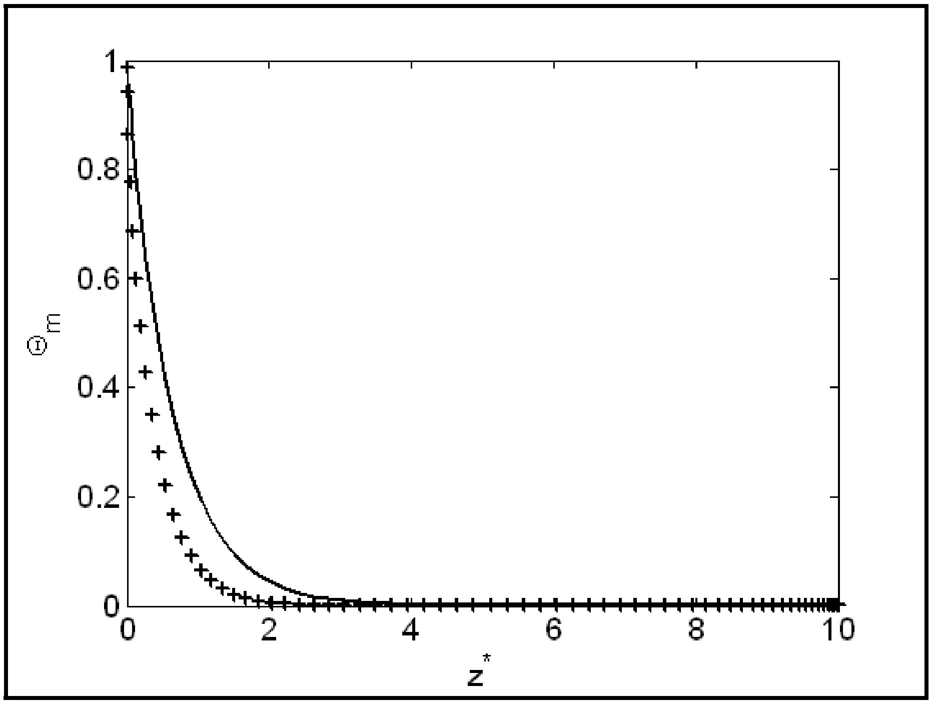

Figure 5.34 Spectral (-) compared to Graetz (+)Re:3.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 3.0; 67x67 spectral grid.

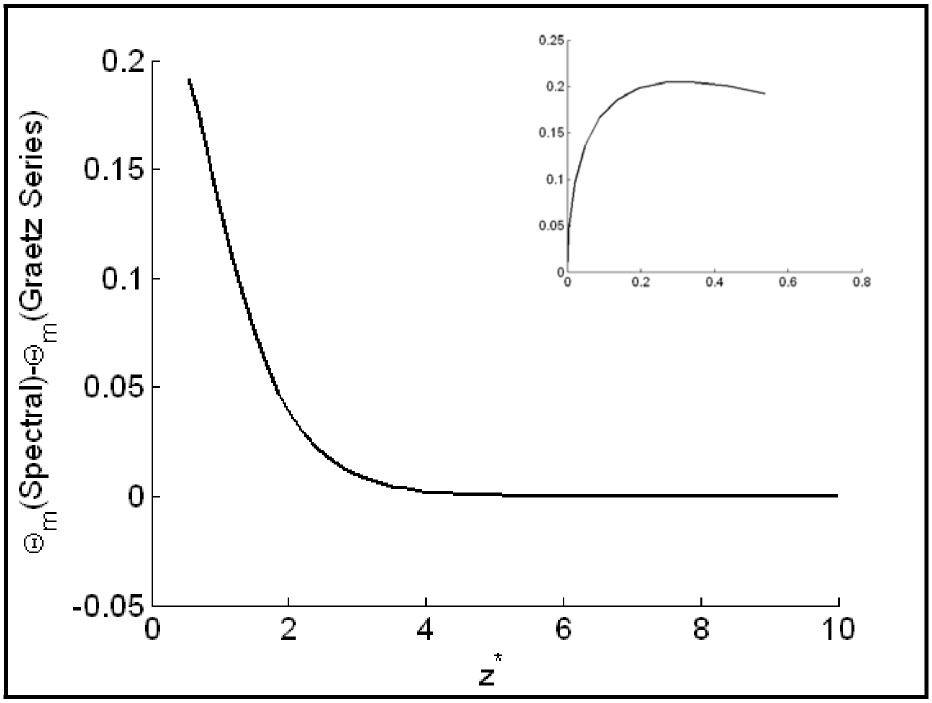

Figure 5.35 Difference between Spectral and Graetz methods, note that the entrance region is in the inlay. Re:3.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 3.0;

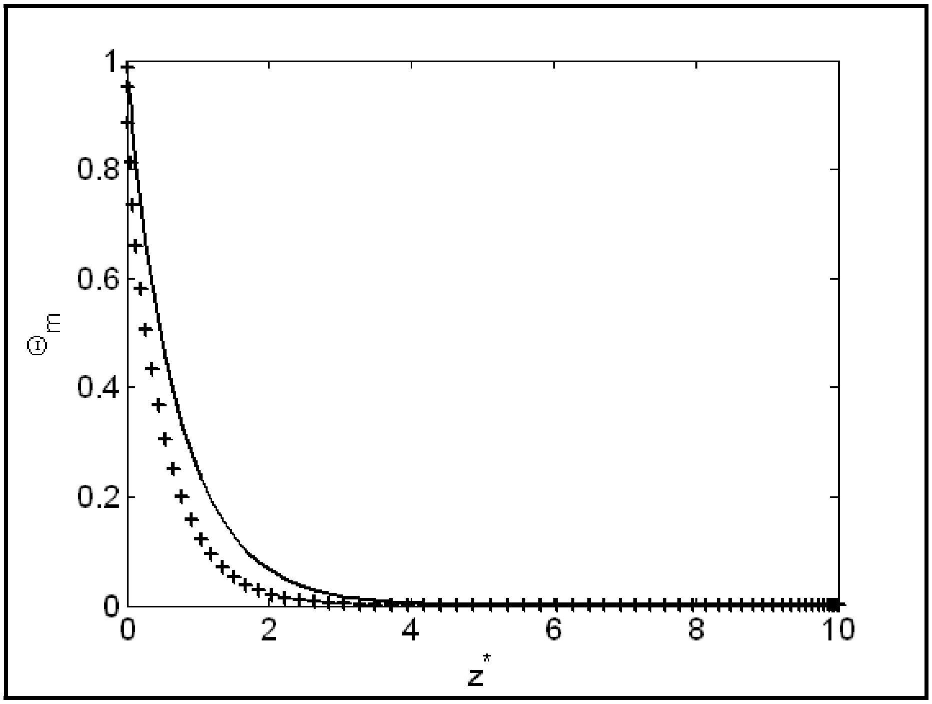

Figure 5.36 Spectral (-) compared to Graetz (+)Re:4.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 4.0; 67x67 spectral grid.

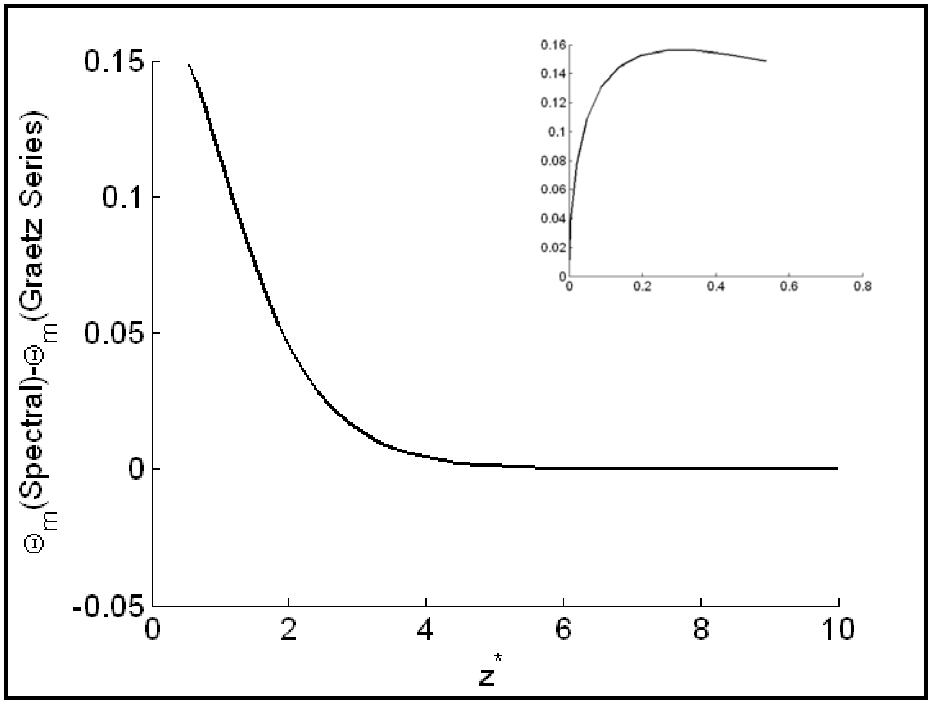

Figure 5.37 Difference between Spectral and Graetz methods, note that the entrance region is in the inlay. Re:4.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 4.0;

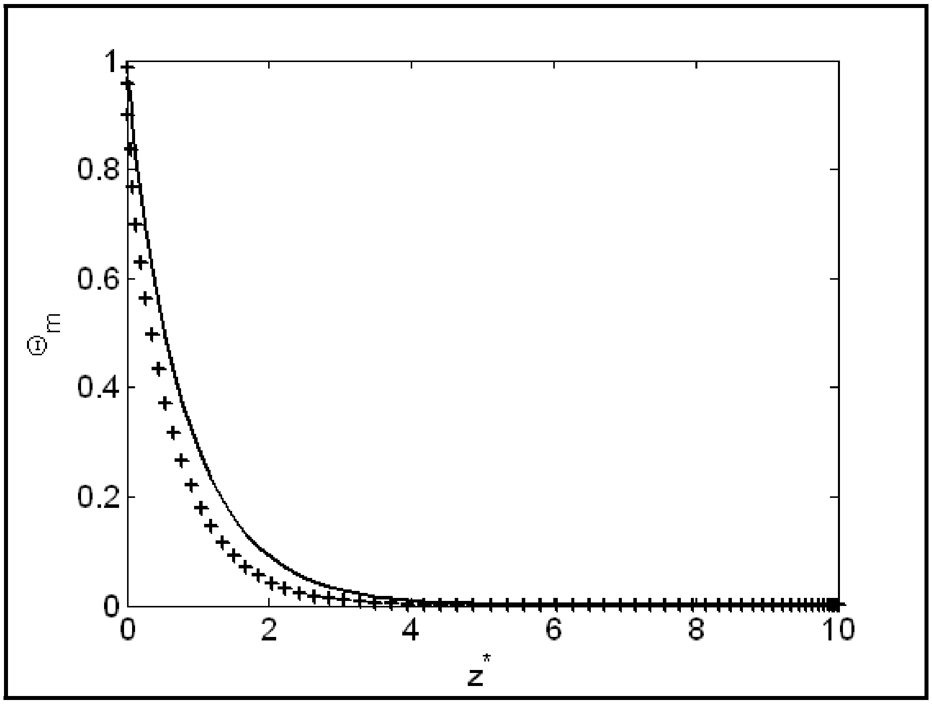

Figure 5.38 Spectral (-) compared to Graetz (+)Re:5.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 5.0; 67x67 spectral grid.

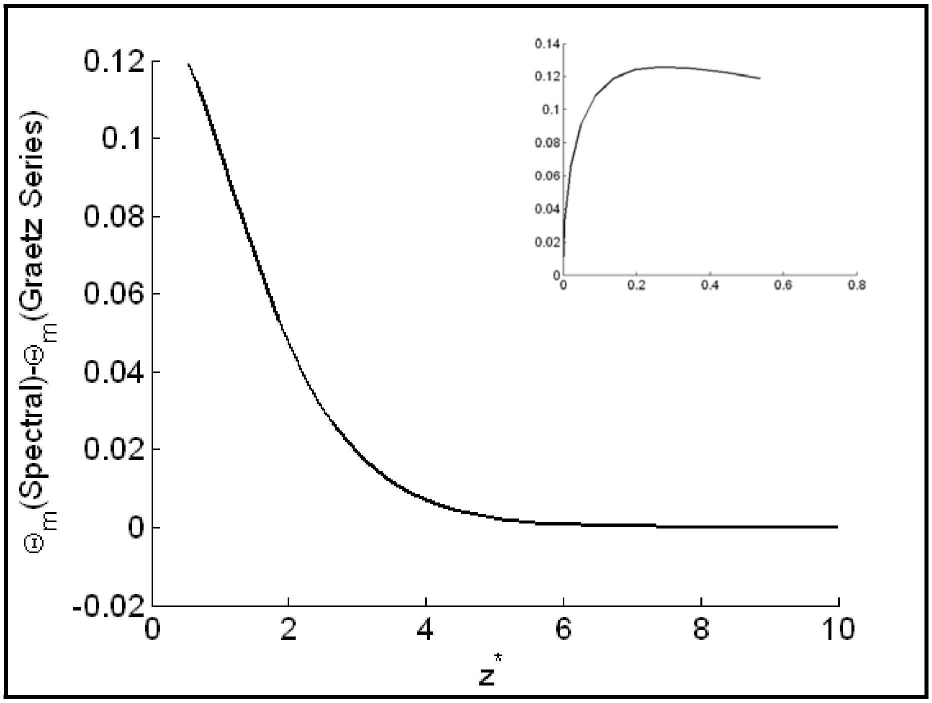

Figure 5.39 Difference between Spectral and Graetz methods, note that the entrance region is in the inlay. Re:5.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 5.0;

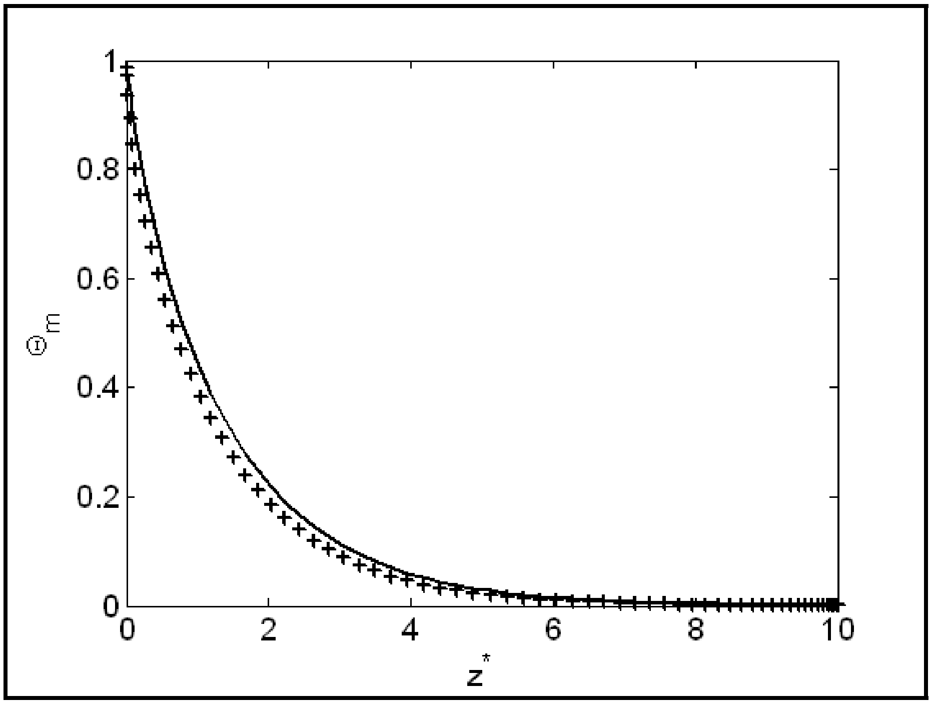

Figure 5.40 Spectral (-) compared to Graetz (+) Re:10.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 10.0; 67x67 spectral grid.

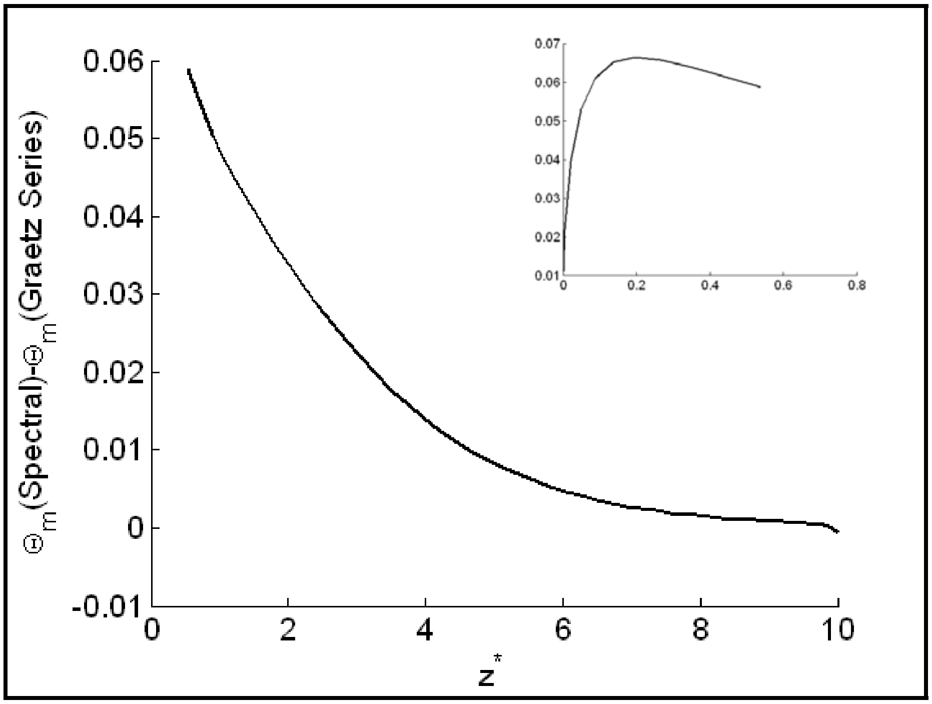

Figure 5.41 Difference between Spectral and Graetz methods, note that the entrance region is in the inlay. Re:10.0; Pr:1.0; R:1.0 m; Tube Length: 10.0 m; Pe: 10.0;

5.5.1 Comparison with Bayazitoglu et. al. [25]

Bayazitoglu et. al. [25] presented a simple finite integral transform technique for the solution of the extended Graetz problem over the dimensionless axial position ξ. A comparison with Table 1 of that paper is given here in Figure 5.42 and Table 3. Notice that the Peclet number chosen in [25] is significantly greater than 10. The solution using spectral collocation without axial conduction is included to illustrate how close the values are to each other. The Matlab code listing for this can be found in Appendix A. Section: 7.8.

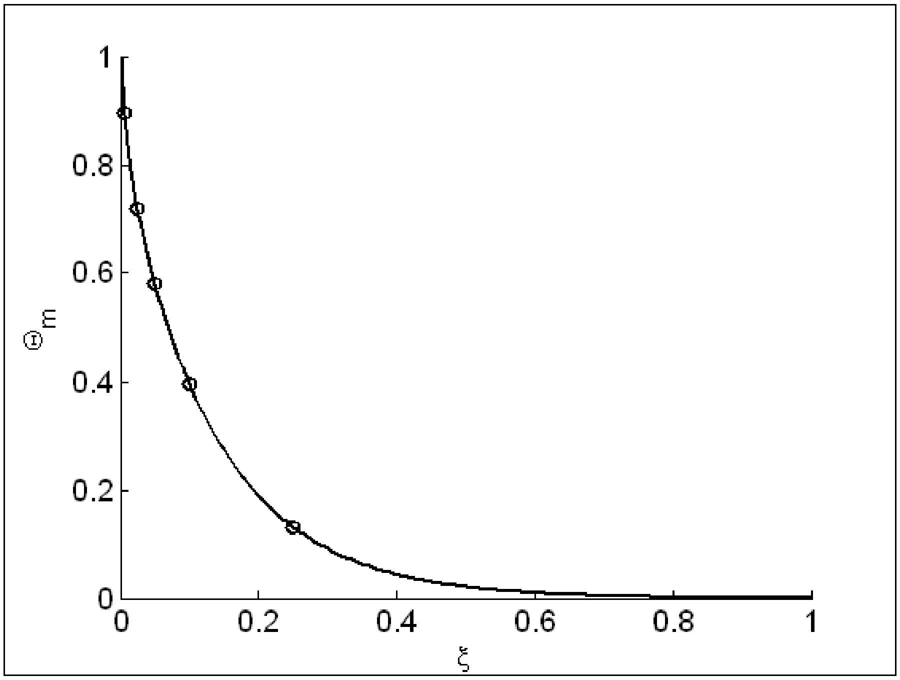

Figure 5.42: Extended Graetz Solution (-) compared to [25] (O), R = 1m, L = 50m, Re = 25, Pr = 2, Pe = 50, 41x41 spectral grid interpolated onto a 201x201 grid.

Table 3: Comparison with [25] for Pe = 50.

	Spectral Collocation with Axial Conduction	Spectral Collocation without Axial Conduction	[25] Lowest Order Solution	Singh's [39]
ξ	**θm**	θm	θm	θm
0.005	0.907	0.8914	0.896	0.910
0.025	0.725	0.7161	0.718	0.730
0.050	0.585	0.5787	0.580	0.590
0.100	0.400	0.3953	0.396	0.400
0.250	0.134	0.1316	0.132	0.134

5.5.2 Simulations with temperature boundary at the Centerline

To illustrate how axial conduction will affect the temperature distribution upstream of the step change, simulations were completed so the step change in temperature occurred at the axial center. The Matlab code can be found in Appendix A: Section 7.9.

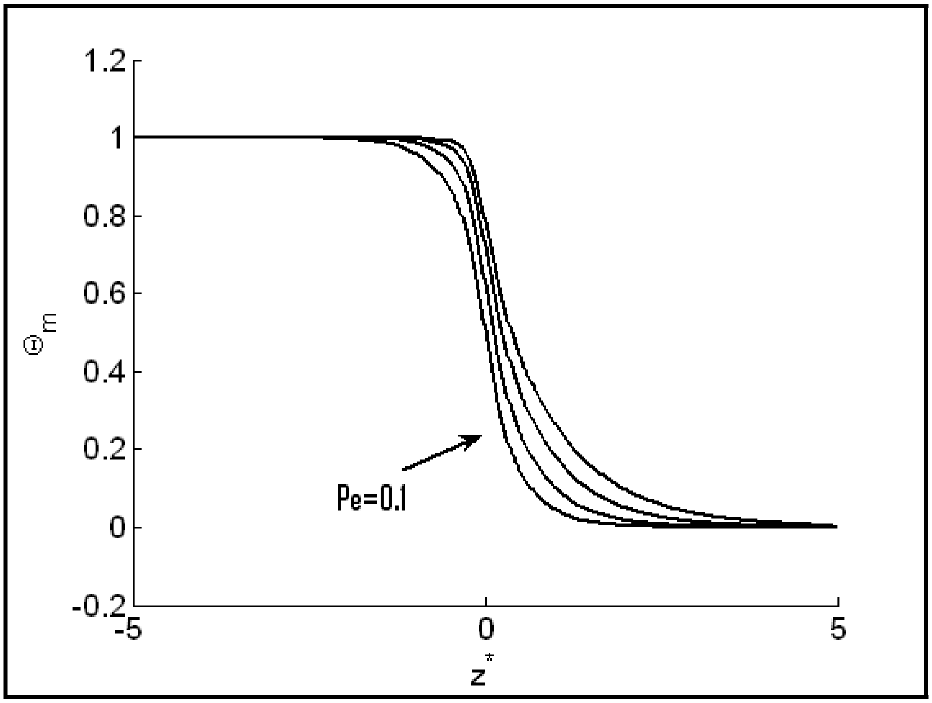

Figure 5.43: Spectral Collocation solution for Graetz Problem with Axial conduction over different Peclet numbers (Pe = 0.1, 2, 4, 6 in the direction of the arrow respectively). Note that the step change in temperature occurs at the centerline. Re = 1; Pr = 1; R = 1m; L = 10m; Computed on a 41x41 spectral grid and interpolated onto a 201x201 grid.

To demonstrate the accuracy for this particular set of boundary conditions a comparison with Table I of [27] was completed; this is presented in Table 4 below. The method used by Ku et. al. [27] was described as a Chebyshev-Finite-element method. Note that these are bulk fluid temperature values along the centerline of the tube where the step change in temperature occurs.

Table 4: Comparison with [27]

Pe	**5**		**10**	
r^*	**CFE** [27]	**Spectral Collocation**	**CFE** [27]	**Spectral Collocation**
0.975	0.5197	0.5183	0.5330	0.5292
0.904	0.5740	0.5698	0.6189	0.6106
0.794	0.6493	0.6459	0.7319	0.7250
0.654	0.7269	0.7267	0.8351	0.8339
0.500	0.7913	0.7932	0.9071	0.9090
0.345	0.8349	0.8377	0.9469	0.9497
0.206	0.8591	0.8621	0.9654	0.9680
0.000	0.8645	0.8748	0.9668	0.9763

5.6 Simple Diffusion

The spectral solution for the simple diffusion case was validated against the analytical solution presented in Section 2.3.2. Figure 5.44 is a plot of the penetration along the dimensionless value μ. Figure 5.45 is the difference between the two methods. As can be seen the Spectral Method is in excellent agreement with the analytical solution. The Penetration was calculated using Equation (4.51).

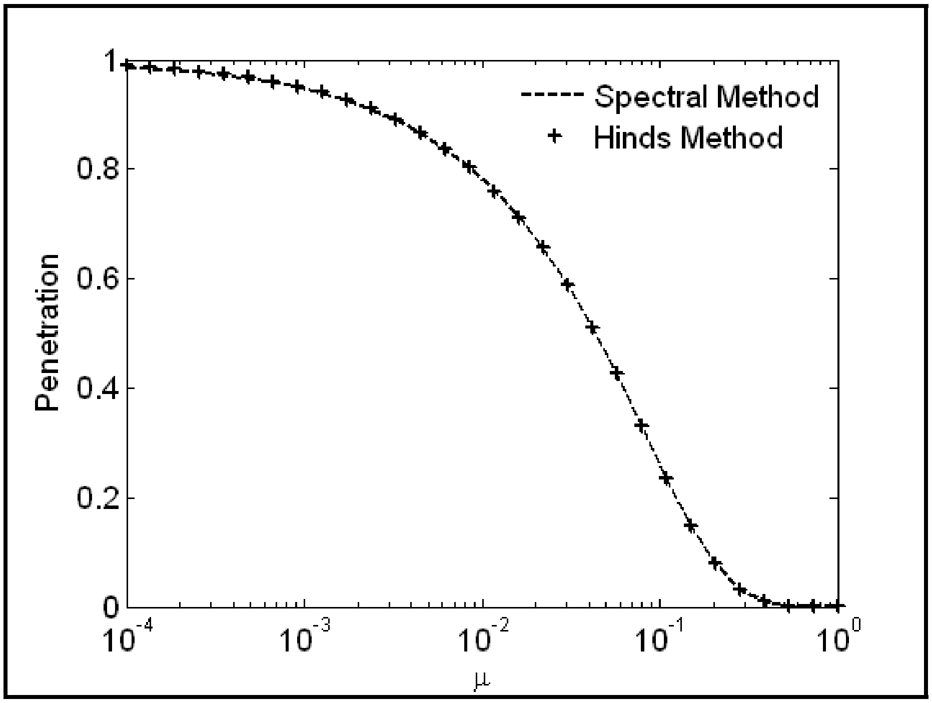

Figure 5.44 Spectral Method compared to Hinds.

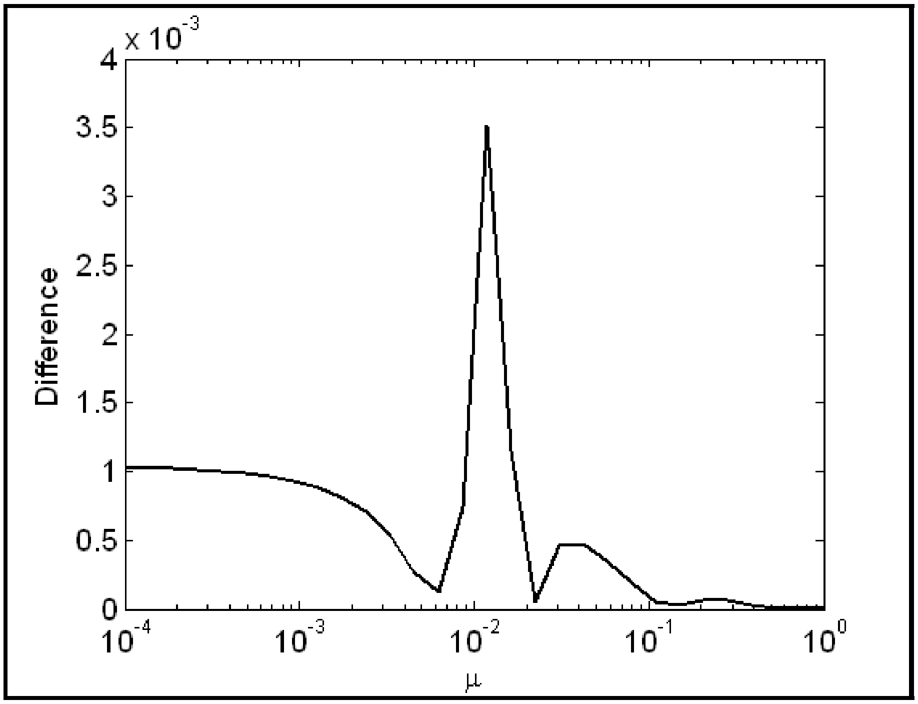

Figure 5.45 Difference between Spectral Method and Hinds solution.

6. Conclusions

When applied to two classic fluid dynamics problems the spectral collocation method proved to be in excellent agreement with previously published results. The constraints that spectral collocation puts on the computational domain were not an issue here, but its use in more complex geometries would be a limiting factor as a general computational fluid dynamics tool.

There are several aspects of the problems modeled here that could be extended. The first would be to model the Graetz problem for both the constant heat flux case and non-linear boundary conditions. The second is in the evaluation of the microtube Graetz problem. Finally, deposition by thermophoresis is an ideal candidate for modeling with the spectral collocation method.

There are also several areas of the spectral method and its application that could be extended. For instance, it was shown here that when coupled with cubic-spline interpolation the spectral collocation method could be extremely efficient in terms of execution time and memory usage. An investigation into the accuracy of such combinations of methods would be useful in elucidating a simple and generic spectral collocation solver. An investigation into the affects of both the grid size and the types of collocation points when applied to CFD problems would also be useful.

Another logical extension of this is the combination of spectral methods and finite-element methods. The topic was avoided in this paper but the so called spectral-element methods are incredibly powerful. They can achieve high accuracy in complex geometrical domains by combining the exponential rate of convergence of the spectral method with flexibility of the finite-element method. A superficial review of this method suggests that it could be useful in developing three-dimensional simulations of thermophoresis and could lead to three-dimensional models of physical vapor transport.

This paper presented an introduction to the spectral collocation method and yet still only managed to break the surface of this complicated and fascinating topic. Spectral collocation and spectral methods as a whole continue to grow and expand in both applied mathematics and engineering. The application and use of these methods as generalized tools is also growing. As seen here, spectral methods are useful and perhaps even necessary tools in computational fluid dynamics research.

References

[1] Stratmann, F., Fissan, H. (1991) Nondimensional Investigation of Submicron Particle Transport on Cooled Laminar Tube Flow. J. Aerosol Sci., Vol. 22, Suppl. 1, pp. S211-S214.

[2] Munoz-Bueno, R., Hontanon, E., Rucandio, M. I. (2004) Deposition of fine aerosols laminar tube flow at high temperature with large gas-to-wall temperature gradients. Aerosol Science 36, pp. 495-520.

[3] Tandon, P., Adewumi, M. A. (1997) Particle deposition from turbulent flow in a pipe. J. Aerosol Sci. Vol 29, No. 1/2, pp. 141-156.

[4] Chen, D. R., Pui, D. Y. H. (1995) Numerical and Experimental Studies of Particle Deposition in a Tube with A Conical Contraction-Laminar Flow Regime. J. Aerosol Sci., Vol 26, No. 4, pp. 563-574.

[5] Ferziger, J. H., Peric, M., (2002). *Computational Methods for Fluid Dynamics.* Springer, New York.

[6] Majumdar, P. (2005). *Computational Methods for Heat and Mass Transfer.* Taylor & Francis, New York.

[7] Bird, R.B., Stewart, W.E., Lightfoot, E.N., (2007). *Transport Phenomena.* Wiley, New York.

[8] Fornberg, B., (1996), A Practical Guide to Pseudospectral Methods. Cambridge University Press, New York, NY, USA.

[9] Canuto, C., Hussaini, M. Y., Quarteroni, A., Zang, T. A., (1987), Spectral Methods in Fluid Dynamics, Springer-Verlag, New York, NY, USA.

[10] Gottleib, D., Orszag, S. A., (1977), Numerical Analysis of Spectral Methods: Theory and Applications, Regional Conference Series in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

[11] Canuto, C., Hussaini, M.Y., Quarteroni, A., Zang, T.A., (2007). Spectral Methods Evolution to Complex Geometries and Applications to fluid Dynamics. Springer, New York.

[12] Boyd, J. P., (2001), Chebyshev and Fourier Spectral Methods Second Edition, Dover Publications, Mineola, NY.

[13] Ben-Yu, G., 1998, Spectral Methods and Their Applications, World Scientific, New Jersey, USA.

[14] Trefethen, L. N., 2000, Spectral Methods in Matlab, SIAM, Philadelphia, PA.

[15] James, M.L., Smith, G.M., Wolford, J.C. (1993). *Applied Numerical Methods for Digital Computation, Fourth Edition.* Harper Collins, New York.

[16] Fausett, L. V., (2008). *Applied Numerical Analysis Using Matlab.* Pearson, New Jersey.

[17] Reddy, J.N., (2008). *An introduction to Continuum Mechanics.* Cambridge, New York.

[18] Jones, A. S., (1971), Extensions to the Solution of the Graetz Problem, Int. J. Heat Mass Transfer, Vol. 14, pp. 619-623.

[19] Sandner, H., (1982), Leo Graetz, Pioneers in Heat Transfer, Seventh International Heat Transfer Conference, Munich, 1982, pp. 407.

[20] Kakac, S., Yener, Y., (1994), Convective Heat Transfer Second Edition, CRC Press, Ann Arbor.

[21] Sellers, J. R., Tribus, M., Klein, J. S., (1955) Heat Transfer to Laminar Flow in a Round Tube or Flat Conduit-The Graetz Problem Extended, Diamond Jubilee Semi-Annual Meeting, Boston, MA. June 19-23, 1955, ASME. Paper No. 55-SA-66, pp. 441-448.

[22] Housiadas, C., Larrode, F. E., Drossinos, Y., (1999), Technical Note Numerical Evaluation of the Graetz Series, Int. J. Heat Mass Transfer, Vol. 42, pp. 3013-3017.

[23] Michelsen, M. L., Villadsen, J., (1974), The Graetz Problem with Axial Heat Conduction, Int. J. Heat Mass Transfer, Vol. 17, pp. 1391-1402.

[24] Ou, J. W., Cheng, K. C., (1977), Natural Convection Effects on Greatz Problem in Horizontal Tubes, Int. J. Heat Mass Transfer, Vol. 20, pp. 953-960.

[25] Bayazitoglu, Y., Ozisik, M. N., (1980), Shorter Communication on the Solution of Graetz Type Problems with Axial Conduction, Int. J. Heat Mass Transfer, Vol. 23, pp. 1399-1402.

[26] Laohakul, C., Chan, C. Y., Look, K. Y., Tan, C. W., (1985) On Approximate Solutions of The Graetz Problem With Axial Conduction, Int. J. Heat Mass Transfer. Vol. 28, No. 3, pp. 541-545.

[27] Ku, H. C., Hatziavramidis, D., (1984), Chebyshev Expansion Methods for the Solution of the Extended Graetz Problem, Journal of Computational Physics, Vol. 56, pp. 495-512.

[28] Poirier, N. A., Mujumdar, A. S., (1989), Numerical Solution to The Graetz Problem: Achieving Accuracy in The Entrance Region, Int. Comm. Heat Mass Transfer, Vol. 16, pp. 205-214.

[29] Ebadian, M. A., Zhang, H. Y., (1989), An Exact Solution of Extended Graetz Problem with Axial Heat Conduction, Int. J. Heat Mass Transfer, Vol. 32, No. 9, pp. 1709-1717.

[30] Telles, A. S., Queiroz, E. M., Elmor Filho, G., (2000), Solutions of The Extended Graetz Problem, Int. J. Heat Mass Transfer, Vol. 44, pp. 471-483.

[31] Mercado, E. R. L., Souza, V. C., Guirardello, R., Nunhez, J. R., (2001), Modeling Flow and Heat Transfer in Tubes Using a Fast CFD Formulation, Computers and Chemical Engineering, Vol. 25, pp. 713-722.

[32] Crowe, C.T., Elger, D. F., Roberson, J. A., (2001). *Engineering Fluid Mechanics.* Wiley, New York.

[33] Arfken, G.B., Weber, H.J, (2005). Mathematical Methods for Physicists, Sixth Edition. Elsevier, New York.

[34] Hinds, W.C., (1999). Aerosol Technology, 2nd Ed., John Wiley & Sons, New York.

[35] Baron, P.A., Willeke, K. (2001), Aerosol Measurement Principles, Techniques, and Applications, John Wiley & Sons, New York.

[36] Ingham, D.B., (1975). Diffusion of Aerosols from a Stream Flowing Through a Cylindrical Tube. Aerosol Science, Vol 6. pp. 125-132.

[37] Conley, N., Lawal, A., Majumdar, A.S., (1985). An Assessment of the Accuracy of Numerical Solutions to The Graetz Problem, Int. Comm. Heat Mass Transfer. Vol. 12, pp. 209-218.

[38] Gutti, V. R., Loyalka, S. K. (2008). Thermophoretic Deposition in a Cylindrical Tube: Computations and Comparison with Experiments

[39] Singh, S. N., (1958). Heat transfer by laminar flow in a cylindrical tube. Applied Science Research, A7, pp. 325-340.

7. Appendix A: Matlab Code Listings

7.1 Graetz Problem: Non-Dimensional Spectral Solution

```
%
%   Section: 2D Non-Dimensionalized Temperature Profile using
%            Spectral Collocation.
%
%   This computes the temperature distribution profile of fluid flow in
%   a circular pipe.  This uses the spectral collocation or
%   psuedospectral method to approximate the temperature of steady
%   state flow based on the Navier Stokes conservation of energy
%   equation.  The solution is based on coordinate transformations to
%   the domain [1,-1].
%
%   function [u,Dr,Dr2,r,Dz,Dz2,z] = spec_tube_temp_nonD(Re,Pr,R, ...
%                                                Len,N,M,ramp, ...
%                                                debug,graph)
%
%   Re:     Reynolds Number
%   Pr:     Prandtl Number
%   R:      Radius of the Tube (m)
%   Len:    Length of the Tube (m)
%   N:      Number of Points in the axial direction (Must be odd)
%   M:      Number of Points in the radial direction (Must be odd)
%   debug:  if set to 1 debug information is printed.
%   graph:  if set to 1 the solution will be plotted out.
%
%   function [u,uu,Dr,Dr2,r,Dz,z,zz_r,L] = spec_tube_temp_nonD(
%                                                       Re,Pr,R,...
%                                                       Len,N,M,...
%                                                       graph)
%
function [u,Dr,Dr2,r,Dz,z,L] = spec_tube_temp_nonD( Re,Pr,R, ...
                                                    Len,N,M, ...
                                                    graph)
% ---------------------------------------------------------------------
%   If the user has not input parameters use defaults.
    if nargin < 7
        Re = 1;
        Pr = 1;
        R = 1;
        Len = 1;
        N = 21;                     % Radial Direction
        M = 21;                     % Axial Direction
        graph = 1;
    end;
% ---------------------------------------------------------------------
%   Create the Chebyshev differentiation Matrices
%       * Note that in the accompanying literature r and z are
%         Xi and Eta respectively.
    [Dr,r] = cheb_derivative(N);               % Radial Direction
    [Dz,z] = cheb_derivative(M);               % Axial Direction
```

```
%   Compute the second order Chebyshev differentiation matrix directly
    Dr2 = Dr^2;
    Dz2 = Dz^2;
% -------------------------------------------------------------------
%   Compute Energy Equation coefficients and resulting diagonal
matrices
% -------------------------------------------------------------------
%   Calculate the first coefficient of the energy equation as a column
%   matrix.
    a = ((2*R*Re*Pr)/Len).*(1 - (r.^2));
    A = diag(a);
%    B = eye(N+1);
%   Create identity matrix.
    I = eye(M+1);
%   Compute Omega.
    O = diag(1./r);
% -------------------------------------------------------------------
%   Compute the tensor product spectral grid.
% -------------------------------------------------------------------
    L = kron2(A,Dz) - kron2(O*Dr,I) - kron2(Dr2,I);
% -------------------------------------------------------------------
%   Create a column matrix to hold the RHS of the energy equation
% -------------------------------------------------------------------
    size = ((N+1)*(M+1));
    rhs = zeros(size,1);
%   Create two matrices containing the coordinate points inside the
%   solution plane.
    [rr,zz] = meshgrid(r,z);
%   Create one column vector that consists of the two coordinate
vectors
%   stacked on top of each other.
    rr = rr(:);
    zz = zz(:);
% -------------------------------------------------------------------
%   Apply the Boundary Conditions
% -------------------------------------------------------------------
%   Find the bounday points of the coordinate vectors:
    b = find(zz==-1 | abs(rr)==1);
%   Impose the boundary conditions on the right side vector.
    rhs(b) = (zz(b)==-1).*1 + (rr(b)==1).*0 + (rr(b)==-1).*0;
%   Correct the entry boundary conditions at the top and bottom
%   corners.
    rhs(1) = 0;
    rhs(size - M) = 0;
%   Apply the boundary conditions to the laplacian
    L(b,:) = zeros(length(b),length(rhs));   % Zero the rows out
    L(b,b) = eye(length(b));                 % put in 1 for known
points
% -------------------------------------------------------------------
%   Solve the Matrix equation
% -------------------------------------------------------------------
    reshape(rhs,(M+1),(N+1));
    u = L\rhs;
    reshape(u,(M+1),(N+1));
% -------------------------------------------------------------------
%   Plot the results.
% -------------------------------------------------------------------
    if(graph == 1)
        uu = reshape(u,(M+1),(N+1));
        [rr,zz] = meshgrid(r,z);
        rr=R*rr;                    % Transform Xi back to [R,-R]
        zz=(Len/(2*R))*(1+zz);      % Transform Nu back to [z*,0]
        zz=R*zz;
        zz_r=(Len/(2*R))*(1+z);      % Transform Nu back to [z*,0]

        figure( ...
            'Name','Tube Flow', ...
            'NumberTitle','off', ...
            'BackingStore','off');

        subplot(2,2,1), surfc(rr,zz,uu),
            xlabel('Radial Distance (m)'),
            ylabel('Axial Distance (m)'), zlabel('Temp'),
            title('Temperature Distribution');

        subplot(2,2,2), contourf(rr,zz,uu),
            xlabel('Radial Distance (m)'),
            ylabel('Axial Distance (m)'),
            title('Temperature Distribution'), colorbar;

%       Plot the Temperature profile at various flow lengths to compare
%       with Gutti & Loyalka. X*=0,0.1,0.2,0.3,0.4,0.5
        [diff,x1] = min(abs(zz_r - 0.0));
        [diff,x2] = min(abs(zz_r - 0.1));
        [diff,x3] = min(abs(zz_r - 0.2));
        [diff,x4] = min(abs(zz_r - 0.3));
        [diff,x5] = min(abs(zz_r - 0.4));
        [diff,x6] = min(abs(zz_r - 0.5));

        N2 = (N+1)/2;
        r_temp = r(N2:N+1);
        r_temp(1) = 0;
        r_temp = r_temp*-1;

        subplot(2,2,3),plot(r_temp,uu(x1,N2:N+1),'-k');
        hold on;
        subplot(2,2,3),plot(r_temp,uu(x2,N2:N+1),'--b');
        subplot(2,2,3),plot(r_temp,uu(x3,N2:N+1),'-.r');
        subplot(2,2,3),plot(r_temp,uu(x4,N2:N+1),'-+c');
        subplot(2,2,3),plot(r_temp,uu(x5,N2:N+1),'--*m');
        subplot(2,2,3),plot(r_temp,uu(x6,N2:N+1),'-.og'),
                xlabel('Dimensionalized Radias'),
                ylabel('Dimensionalized Temp.'),
                title('Temperature Profile at Various Flow Lengths');
            h = legend(sprintf('%.2f',zz_r(x1)), ...
                       sprintf('%.2f',zz_r(x2)), ...
                       sprintf('%.2f',zz_r(x3)), ...
                       sprintf('%.2f',zz_r(x4)), ...
                       sprintf('%.2f',zz_r(x5)), ...
                       sprintf('%.2f',zz_r(x6)));
            legend(h, 'boxoff');
        hold off;

        str = sprintf('Reynolds: %.1f',Re);
        str = strvcat(str, sprintf('Prandtl %.1f',Pr));
        str = strvcat(str, sprintf('Tube Radius: %.3f m',R));
        str = strvcat(str, sprintf('Tube Length: %.1f m',Len));
        str = strvcat(str, sprintf('Collocation Points (Radial): %d',
                                    N));
        str = strvcat(str, sprintf('Collocation Points (Axial): %d',
                                    M));

        uicontrol('style','text',...
            'units','normalized','position',[0.55 0.05, 0.4, 0.4], ....
            'HorizontalAlignment',['left'],'string',str);
    end;
% -------------------------------------------------------------------
```

7.2 Graetz Problem: Non-Dimensional Spectral Solution

```
%
%   Section:    2D Non-Dimensionalized Temperature Profile using
%               Spectral Collocation.
%
%   This computes the temperature distribution profile of fluid flow in
%   a circular pipe.  This uses the spectral collocation or
%   psuedospectral method to approximate the temperature of steady
%   state flow based on the Navier Stokes conservation of energy
%   equation.  The solution is based on coordinate transformations to
%   the domain [1,-1].
%
%   function [u,Dr,Dr2,r,Dz,Dz2,z] = spec_tube_temp_nonD_wAC(Re,Pr,R,
%                                                     Len,N,M,ramp,
%                                                     debug,graph)
%
%   Re:     Reynolds Number
%   Pr:     Prandtl Number
%   R:      Radius of the Tube (m)
%   Len:    Length of the Tube (m)
%   N:      Number of Points in the axial direction (Must be odd)
%   M:      Number of Points in the radial direction (Must be odd)
%   debug:  if set to 1 debug information is printed.
%   graph:  if set to 1 the solution will be plotted out.
%
%   function [u,uu,Dr,Dr2,r,Dz,z,zz_r,L] = spec_tube_temp_nonD(
%                                                         Re,Pr,R,...
%                                                         Len,N,M,...
%                                                         graph)
%
function [u,Dr,Dr2,r,Dz,z,L] = spec_tube_temp_nonD_wAC( Re,Pr,R, ...
                                                     Len,N,M, ...
                                                     graph)
%-------------------------------------------------------------------
%   If the user has not input parameters use defaults.
    if nargin < 7
        Re = 1;
        Pr = 1;
        R = 1;
        Len = 1;
        N = 21;                     % Radial Direction
        M = 21;                     % Axial Direction
        graph = 1;
    end;
%-------------------------------------------------------------------
%   Create the Chebyshev differentiation Matrices
%       * Note that in the accompanying literature r and z are
%         Xi and Eta respectively.
    [Dr,r] = cheb_derivative(N);              % Radial Direction
    [Dz,z] = cheb_derivative(M);              % Axial Direction
%   Compute the second order Chebyshev differentiation matrix directly
    Dr2 = Dr^2;
    Dz2 = Dz^2;
%---------------------------------------------------------------------
%   Compute Energy Equation coefficients and resulting diagonal
matrices
%---------------------------------------------------------------------
%   Calculate the first coefficient of the energy equation as a column
%   matrix.
    a = ((2*R*Re*Pr)/Len).*(1 - (r.^2));
    A = diag(a);
%    B = eye(N+1);
%   Create identity matrix.
    I = eye(M+1);
%   Compute Omega.
    O = diag(1./r);
%---------------------------------------------------------------------
%   Compute the tensor product spectral grid.
%---------------------------------------------------------------------
    L = kron2(A,Dz) - kron2(eye(N+1),((4*R^2)/Len^2).*Dz2) - ...
        kron2(O*Dr,I) - kron2(Dr2,I);
%---------------------------------------------------------------------
%   Create a column matrix to hold the RHS of the energy equation
%---------------------------------------------------------------------
    size = ((N+1)*(M+1));
    rhs = zeros(size,1);
%   Create two matrices containing the coordinate points inside the
%   solution plane.
    [rr,zz] = meshgrid(r,z);
%   Create one column vector that consists of the two coordinate
vectors
%   stacked on top of each other.
    rr = rr(:);
    zz = zz(:);
%---------------------------------------------------------------------
%   Apply the Boundary Conditions
%---------------------------------------------------------------------
%   Find the bounday points of the coordinate vectors:
    b = find(abs(zz)==1 | abs(rr)==1);
%   Impose the boundary conditions on the right side vector.
    rhs(b) = (zz(b)==-1).*1 + (rr(b)==1).*0 + ...
             (rr(b)==-1).*0 + (zz(b)==1).*0;
%   Correct the entry boundary conditions at the top and bottom
%   corners.
    rhs(1) = 0;
    rhs(size - M) = 0;
%   Apply the boundary conditions to the laplacian
    L(b,:) = zeros(length(b),length(rhs));  % Zero the rows out
    L(b,b) = eye(length(b));                % put in 1 for known points
%---------------------------------------------------------------------
%   Solve the Matrix equation
%---------------------------------------------------------------------
    reshape(rhs,(M+1),(N+1));
    u = L\rhs;
    reshape(u,(M+1),(N+1));
%---------------------------------------------------------------------
%   Plot the results.
```

```
%---------------------------------------------------------------------
    if(graph == 1)
        uu = reshape(u,(M+1),(N+1))
        [rr,zz] = meshgrid(r,z);
        rr=R*rr;                    % Transform Xi back to [R,-R]
        zz=(Len/(2*R))*(1+zz);      % Transform Nu back to [z*,0]
        zz=R*zz;
        zz_r=(Len/(2*R))*(1+z);      % Transform Nu back to [z*,0]

        figure( ...
            'Name','Tube Flow', ...
            'NumberTitle','off', ...
            'BackingStore','off');

        mesh(rr,zz,uu),
            xlabel('Radial Distance (m)'),
            ylabel('Axial Distance (m)'), zlabel('Temp'),
            view(60,40), colormap(1e-2*[1 1 1]);

% --------------------------------------------------------------------
```

7.3 Graetz Problem: Housaidas Series Solution

Adapted from code provided by Dr. Patrick Tebbe (Included with permission).

```
%--------------------------------------------------------------------
%
% this program solves the Graetz problem, it outputs the average
% fluid temperature calculated axially down a tube
%
% this version uses a set error limit and determines the necessary
% number of terms to use in the series, for each axial location
%
%    by Patrick A. Tebbe
%
%--------------------------------------------------------------------
function [T] = graetz_tube_nonD(Rad, Len, z, Pe, emax, N, graph)

    % If the user hasn't input any variables
    if nargin < 6
        Rad = 0.1;
        Len = 0.0001;
        Pe = 10;
        emax = 0.001;
        N = 100;
        z = [0:Len/N:Len];
        graph = 1;
    end;

    % the series coefficients [Housiadas 10]
    S=[0.159152288 0.0114856354 -0.224731440 -0.033772601];
    P=[0.7116134100  0.0721675797 0.0577777781 -0.0026040798 ...
      -0.0977347533 -0.0153141806];
    Q=[0.7026286927 -0.0721675799 -0.0577777778 0.2122030514 ...
       0.0078122394  0.1060741183  0.0306283612];

    % the count variable indexes the arrays.
    count = 0;
    for loop=1:N+1
        count=count+1;
        theta=0;
        n=0;
        % the error value must be initialized to a large value
        error=1000;
        % the non-dimensional axial location is calculated
        phi=z(loop)/(Rad*Pe);
        % a loop is started for the terms in the series solution,
        % it keeps going until error is below the limit or 10 terms
        % have been added
        while (error>emax)&&(n<=10)
            n=n+1;
            % the graetz function is used to calculated the
            % eigenvalues, etc.
           [lambda,R,LAMBDA] = graetz(n,S,P,Q);
            % the term is added to the summation and the error
            % is calculated
           old=theta;
           %  [Housiadas eq. 2]
           theta = theta + 8*R*exp(-1*lambda^2*phi)/(lambda^3*LAMBDA);
           error=(theta-old)/theta*100;
        end;
        % final values are output to the arrays for later plotting
        % the terms array keeps track of how many terms are used at
        % each axial location
        z(count)=loop;
        relerror(count)=error;
        terms(count)=n;
        T(count)=theta;%
    end;

    if graph == 1
        plot(z,T);
        % the figure command opens a second window
        figure;
        plot(z,terms);
    end;

%
%   function to determine the eigenvalues and associated
%   constants for the Graetz problem series solution
%   by Patrick A. Tebbe
%
%
% the term number and general constants are input to the function
%
function [lambda,R,LAMBDA] = graetz (n,S,P,Q)
    % [Housiadas eqs. 6 & 7]
    omega=4*n-(4/3);
    lambda=omega+S(1)*omega^(-4/3)+S(2)* ...
                      omega^(-8/3)+S(3)* ...
                      omega^(-10/3)+S(4)* ...
                      omega^(-11/3);
    % [Housaidas eq. 8]
    R=(-1)^n*P(1)*lambda^(1/3)*(1+P(2)*omega^(-4/3)+ ...
                                  P(3)*omega^(-2)+ ...
                                  P(4)*omega^(-8/3)+ ...
                                  P(5)*omega^(-10/3)+ ...
                                  P(6)*omega^(-11/3));
    % [Housaidas eq. 9]
    LAMBDA=(-1)^n*Q(1)*lambda^(-1/3)*(1+ ...
                                      Q(2)*omega^(-4/3)+ ...
                                      Q(3)*omega^(-2)+ ...
                                      Q(4)*omega^(-7/3)+ ...
                                      Q(5)*omega^(-8/3)+ ...
                                      Q(6)*omega^(-10/3)+ ...
                                      Q(7)*lambda^(-11/3));
end
```

7.4 Spectral Method compared to Graetz Series

```
function Spec_Graetz_Compare_nonD(Re,Pr,R, ...
                                  Len,N,M, ...
                                  X,Y,limit, ...
                                  emax,num_pts,graph)
%----------------------------------------------------------------------
%   If the user has not input parameters use defaults.
    if nargin < 9
        Re = 1;
        Pr = 1;
        R = 1;
        Len = 1;
        N = 69;                     % Radial Direction
        M = 69;                     % Axial Direction
        emax=0.0001;
        num_pts=69;
        graph = 0;
    end;
    plot_difference = 1;
    plot_single_compare = 1;
    combine_plots = 1;          % This will negate the first two.
    plot_flag = 0;
%----------------------------------------------------------------------
    Pe = [.1,.5,1,2,3,4,5,10,10,50,100,200,300,400,500,1000];
    count = 8;
    Re=Pe./Pr;

    for i = 1:count
        T_avg = zeros(M+1,1);
        %--------------------------------------------------------------
        %   Psuedospectral Method
        %--------------------------------------------------------------
       [T_spec,Dr,Dr2,rad,Dz,z,L] = spec_tube_temp_nonD(Re(i),Pr,R, ...
                                                        Len,N,M, ...
                                                        graph);
        T_spec = reshape(T_spec,(M+1),(N+1));
        for j = 1:M+1
            T_avg(j) = bulk_fluid_temp_michelsen(T_spec(j,:),rad,N); %
        end;
        [rr,zz] = meshgrid(rad,z);
        z=(Len/(2*R))*(1+z);      % Transform Nu back to [z*,0]
        z=R*z;
        %--------------------------------------------------------------
        %   Graetz Method
        %--------------------------------------------------------------
        [T_graetz] = graetz_tube_nonD(R, Len, z, Pe(i), ...
                                      emax, num_pts, graph);
        T_graetz = reshape(T_graetz,num_pts+1,1);
        %--------------------------------------------------------------
        %   Plot the results.
        %--------------------------------------------------------------
```

```
if(combine_plots == 1)

    sprintf('Comparison Re:%.1f; Pr:%.1f; R:%.1f m; ...
            Tube Length: %.1f m; Pe: %.1f;', ...
            Re(i),Pr,R,Len,Pe(i));

    if(plot_flag == 0 )
        figure('Name', 'Combined Results',
               'NumberTitle','off', ...
               'BackingStore','off');
        plot_flag = 1;
    end;

    H1 = plot(z,T_avg);
         xlabel('Axial Position'),
         ylabel('\Theta_m');
    hold on;
    set(H1,'LineStyle','-');
    set(H1,'LineWidth',2);
    set(H1,'color','k');
else
    if(plot_single_compare == 1)
        str = sprintf('Comparison Re:%.1f; Pr:%.1f;
                      R:%.1f m; Tube Length: %.1f m;
                      Pe: %.1f;',Re(i),Pr,R,Len,Pe(i));

        figure('Name',str, ...
               'NumberTitle','off', ...
               'BackingStore','off');
        H1 = plot(z,T_avg);
        hold on;
        H2 = plot(z,T_graetz);
             xlabel('Axial Position'),
             ylabel('\Theta_m');
        set(H1,'LineStyle','-');
        set(H1,'LineWidth',2);
        set(H1,'color','k');
        set(H2,'LineStyle','+');
        set(H2,'LineWidth',2);
        set(H2,'color','k');
        hold off;
    end

    if(plot_difference == 1)

        figure( ...
             'Name',sprintf('Diff ent.: Re:%.1f; Pr:%.1f;
                            R:%.1f m; Tube Length: %.1f m;
                            Pe: %.1f;',Re(i),
                            Pr,R,Len,Pe(i)), ...
             'NumberTitle','off', ...
             'BackingStore','off');
            hold on;
            plot(z(num_pts+1:-1:num_pts-9),
                    (T_avg(num_pts+1:-1:num_pts-9)-
                    T_graetz(num_pts+1:-1:num_pts-9)),
                    'k');
            hold off;

          figure( ...
             'Name',sprintf('Difference.: Re:%.1f; Pr:%.1f; R:%.1f
                                m; Tube Length: %.1f m;
                                Pe: %.1f;',Re(i),
                                Pr,R,Len,Pe(i)), ...
             'NumberTitle','off', ...
             'BackingStore','off');

            hold on;
            plot(z(num_pts-9:-1:1),
                (T_avg(num_pts-9:-1:1)-T_graetz(num_pts-9:-1:1)),
                 'k');
                xlabel('Axial Position'),
                ylabel('\Theta_m(Spectral)-\Theta_m(Graetz
                                                        Series)');
            hold off;

            max_diff = max(abs(T_avg-T_graetz))
            mean_diff = mean(T_avg-T_graetz)

            sprintf('Re:%.1f; Pr:%.1f; R:%.1f m;
                    Tube Length: %.1f m; Pe: %.1f; max= %.5f,
                    mean= %.5f',Re(i),Pr,R,Len,Pe(i),
                    max_diff,mean_diff)
        end; % if( plot_difference...

    end; % if(Combine_Plots...

    T_avg = 0;
    T_graetz = 0;
end;

hold off;
```

7.5 Graetz Problem: Non-Dimensional Finite Difference

```
function Simulate_Tube(path)
    % Input Variables
    Re = 1; Pr = 1; R = 1; L = 0.5;
    % Grid Setup
    N = 21; M = 21;
    Grid = zeros(M+1,N+1);
    %  Apply Boundary Conditions
    Grid(1,1:N) = 0; Grid(M,1:N) = 0;
    Grid(2:M,1) = 1; Grid(1:M,N) = 0;
    Grid_Old = Grid;
    iterations = 0;
    flag = 0; ESPI = 1e-9;
    del_r = 2/M; del_z = (L/R)/N;
    while flag == 0
        flag = 1;
        % Calculate the entry Region.
        for i=2:M
            A = -((del_r^2)/del_z)* ...
                  ((Re*Pr)/2)* ...
                  (1-((i-1)*del_r-1)^2);
            B = del_r/(2*((i-1)*del_r-1));
            for j = 2:N
                Grid(i,j) = 0.5*(A*(Grid_Old(i,j+1)- ...
                                    Grid_Old(i,j-1))+ ...
                                 B*(Grid_Old(i+1,j)- ...
                                    Grid_Old(i-1,j))+ ...
                                    Grid_Old(i-1,j)+ ...
                                    Grid_Old(i+1,j));
                if(((Grid(i,j)-Grid_Old(i,j))-ESPI) > 0)
                    flag = 0;
                end;
            end;
        end;
        Grid_Old = Grid;
        iterations = iterations + 1;
        if iterations > 1000
            flag = 1;
            'Error: increments > 1000'
        end;
    end;
```

7.6 Weighted Temperature Calculation

```
%
%   This computes the weighted average of the radial
%   temperature at an axial location along a tube.
%
%   function [avg] = Weighted_Average(T,R)
%
%   T:      Temperature column matrix
%   r_p:    Radial points
%   N:      Number of radial points
%   R:      Radius
%
function [avg] = Weighted_Average(T,r_p,N,R)
    N2 = (N+1)/2;
    % reshape the temperature array
    T = T(N2+1:-1:1);
    % reshape the radial coordinates to go from 0 to R
    r_p=R.*r_p;
    r_p = r_p(N2+1:-1:1);
    r_p(1) = 0.0;
    r_p = reshape(r_p,1,N2+1);
    % Calculate the integral on the numerator
    num = trapz(r_p,T.*r_p);
    avg = (2*num)/R^2;
```

7.7 Bulk Fluid Temperature Calculation

```
%
%   This computes the bulk fluid temperature at an axial
%   location along a tube.  The problem is assumed to be
%   non-dimensional
%
%   Follows Michelsen (1974)
%
%   function [avg] = bulk_fluid_temp(T,r_p,N)
%
%   T:      Temperature column matrix
%   r_p:    Radial points
%   N:      Number of radial points
%
function [avg] = bulk_fluid_temp_michelsen(T,r_p,N)
    N2 = (N+1)/2;
    % reshape the temperature array
    T = T(N2+1:-1:1);
    % reshape the radial coordinates to go from 0 to 1
    r_p = r_p(N2+1:-1:1);
    r_p(1) = 0.0;
    r_p = reshape(r_p,1,N2+1);
    % Calculate the integral on the numerator
    num = trapz(r_p,T.*r_p.*(1-r_p.^2));
    avg = 4*(num);
```

7.8 Spectral Method: Comparison with [25]

```
%
%   This it to compare to:
%   SHORTER COMMUNICATION:
%       ON THE SOLUTION OF GRAETZ TYPE PROBLEMS
%       WITH AXIAL CONDUCTION
%
%       Y. Bayazitoglu and M. N. Ozisik 1984.
%
function Bayazitoglu_Compare()

if nargin < 9
        Re = 25;
        Pr = 2;
        Rad = 1;
        Len = 50;
        N = 41;                     % Radial Direction
        M = 41;                     % Axial Direction
        graph = 0;
    end;

    PLOT_TEMP = 1;
    PLOT_NUSSELT = 0;
    %-----------------------------------------------------------------
    %
    %-----------------------------------------------------------------
    [T_spec,Dr,Dr2,rad,Dz,z,L] = spec_tube_temp_nonD(Re,Pr,Rad, ...
                                                 Len,N,M, ...
                                                 graph);

% Uncomment to compare with the Extended Graetz Solution
%   [T_spec,Dr,Dr2,rad,Dz,z,L] = spec_tube_temp_nonD_wAC(Re,Pr,Rad, ...
%                                                  Len,N,M, ...
%                                                  graph);

    T_spec = reshape(T_spec,(M+1),(N+1));

    [xx,yy] = meshgrid(z,rad);
    [xxx,yyy] = meshgrid(-1:.01:1,-1:.01:1);
    uuu = interp2(xx,yy,T_spec,xxx,yyy,'cubic');

    if(PLOT_TEMP == 1)
        % Reshape so we can calculate the builk fluid temp.
        xxx = reshape(xxx(1,:),size(xxx,1),1);

        for j = 1:size(uuu,1)
            T_avg(j) =
                  bulk_fluid_temp_michelsen(uuu(j,:),xxx,size(uuu,1));
        end;
    end;

    if(PLOT_NUSSELT == 1)
        for j = 1:size(uuu,1)
            Nu(j) = spectral_local_nusselt(uuu(j,:),xxx,size(uuu,1)); %
        end;
    end;

    z=(Len/(2*Rad))*(1+yyy);      % Transform Nu back to [z*,0]
    z = z./(Rad*Re*Pr);

    %contour(uuu);

    %size(uuu);
    %-------------------------------------------------------------------
    %       Setup Bayazitoglu Solution for PE = 50.
    %-------------------------------------------------------------------
    r = [0.005,0.025,0.05,0.1,0.25];
    graetz_temp = [0.896,0.718,0.580,0.396,0.132];
    %-------------------------------------------------------------------
    %       Setup Spectral
    %-------------------------------------------------------------------
    z(:,1)
    z = reshape(z(:,1),size(z,1),1)
    [diff,x(1)] = min(abs(z - 0.005));
    [diff,x(2)] = min(abs(z - 0.025));
    [diff,x(3)] = min(abs(z - 0.05));
    [diff,x(4)] = min(abs(z - 0.1));
    [diff,x(5)] = min(abs(z - 0.25));

    if(PLOT_TEMP == 1)
        T_avg(x)
        figure( ...
                'Name','Graetz Problem', ...
                'NumberTitle','off', ...
                'BackingStore','off');
        hold on
        plot(r,graetz_temp, 'ok');
        plot(z,T_avg,'-k'),
            xlabel('\xi'),
            ylabel('\Theta_m'),

        hold off;
    end;

    if(PLOT_NUSSELT == 1)
        Nu(x)
        figure( ...
                'Name','Nusselt', ...
                'NumberTitle','off', ...
                'BackingStore','off');
        hold on

        plot(z(1:100),Nu(1:100),'--k'),
            xlabel('\xi'),
            ylabel('Nu'),
        hold off;
```

```
end;
```

7.9 Spectral Method: Extended Graetz Solution with step change at the centerline

```
%
%   Section:2D Non-Dimensionalized Temperature Profile using Spectral
%              Collocation.
%
%   This computes the temperature distribution profile of fluid flow in
%   a circular pipe subjected to a step change in temperature right
%   after the axial centerline.  This uses the spectral collocation or
%   psuedospectral method to approximate the temperature of steady
%   state flow based on the Navier Stokes conservation of energy
%   equation.  The solution is based on coordinate transformations to
%   the domain [1,-1].
%
% function [u,Dr,Dr2,r,Dz,Dz2,z] = spec_tube_temp_nonD_wAC(Re,Pr,R, ...
%                                                    Len,N,M,ramp, ...
%                                                    debug,graph)
%
%   Re:     Reynolds Number
%   Pr:     Prandtl Number
%   R:      Radius of the Tube (m)
%   Len:    Length of the Tube (m)
%   N:      Number of Points in the axial direction (Must be odd)
%   M:      Number of Points in the radial direction (Must be odd)
%   debug:  if set to 1 debug information is printed.
%   graph:  if set to 1 the solution will be plotted out.
%
%
function [u,Dr,Dr2,r,Dz,z,L] = spec_tube_temp_nonD_wAC_centerline(
                                                     Re,Pr,R, ...
                                                     Len,N,M, ...
                                                     graph)
%-----------------------------------------------------------------------
%   If the user has not input parameters use defaults.
    if nargin < 7
        Re = 1;
        Pr = 1;
        R = 1;
        Len = 10;
        N = 41;                     % Radial Direction
        M = 41;                     % Axial Direction
        graph = 1;
    end;
%-----------------------------------------------------------------------
%   Create the Chebyshev differentiation Matrices
%       * Note that in the accompanying literature r and z are
%         Xi and Eta respectively.
    [Dr,r] = cheb_derivative(N);               % Radial Direction
    [Dz,z] = cheb_derivative(M);               % Axial Direction
%   Compute the second order Chebyshev differentiation matrix directly
    Dr2 = Dr^2;
    Dz2 = Dz^2;
%----------------------------------------------------------------------
%   Compute Energy Equation coefficients and resulting diagonal
matrices
%----------------------------------------------------------------------
%   Calculate the first coefficient of the energy equation as a column
%   matrix.
    a = ((2*R*Re*Pr)/Len).*(1 - (r.^2));
    A = diag(a);
%    B = eye(N+1);
%   Create identity matrix.
    I = eye(M+1);
%   Compute Omega.
    O = diag(1./r);
%----------------------------------------------------------------------
%   Compute the tensor product spectral grid.
%----------------------------------------------------------------------
    L = kron2(A,Dz) - kron2(O*Dr,I) - kron2(Dr2,I);
%----------------------------------------------------------------------
%   Create a column matrix to hold the RHS of the energy equation
%----------------------------------------------------------------------
    size = ((N+1)*(M+1));
    rhs = zeros(size,1);
%   Create two matrices containing the coordinate points inside the
%   solution plane.
    [rr,zz] = meshgrid(r,z);
%   Create one column vector that consists of the two coordinate
vectors
%   stacked on top of each other.
    rr = rr(:);
    zz = zz(:);
%----------------------------------------------------------------------
%   Apply the Boundary Conditions
%----------------------------------------------------------------------
%   Find the bounday points of the coordinate vectors:
    b = find(abs(zz)==1 | abs(rr)==1);
%   Impose the boundary conditions on the right side vector.
    rhs(b) =    (zz(b)==-1).*1 + ...
                (rr(b)==1 & zz(b) > 0).*0 + (rr(b)==1 & zz(b) < 0).*1 +
...
                (rr(b)==-1 & zz(b) > 0).*0 + (rr(b)==-1 & zz(b) <
0).*1;
%   Correct the entry boundary conditions at the top and bottom
%   corners.
    rhs(1) = 0;
    rhs(size - M) = 0;
    rhs(size) = 1;
    rhs(M+1) = 1;
%   Apply the boundary conditions to the laplacian
    L(b,:) = zeros(length(b),length(rhs));   % Zero the rows out
    L(b,b) = eye(length(b));                 % put in 1 for known
points
%----------------------------------------------------------------------
%   Solve the Matrix equation
%----------------------------------------------------------------------
```

```
    reshape(rhs,(M+1),(N+1));
    u = L\rhs;
%----------------------------------------------------------------------
%   Plot the results.
%----------------------------------------------------------------------
    if(graph == 1)
        uu = reshape(u,(M+1),(N+1));
        [rr,zz] = meshgrid(r,z);
        rr=R*rr;                    % Transform Xi back to [R,-R]
        %zz=(1/(2*R))*(1+zz);      % Transform Nu to [z*,0]
        zz = zz*(Len/2);

        zz_r=(Len/(2*R))*(1+z);      % Transform Nu back to [z,0]

        figure( ...
            'Name','Tube Flow', ...
            'NumberTitle','off', ...
            'BackingStore','off');

        mesh(rr,zz,uu),
            xlabel('Radial Position (m)'),
            ylabel('Axial Position (m)'), zlabel('\Theta_m'),
            view(60,40), colormap(1e-2*[1 1 1]);
    end;
% ---------------------------------------------------------------------
```

7.10 Simple Non-Dimensional Diffusion Spectral Method

```
%
%   This computes the particle number concentration of fluid flow in
%   a circular pipe.  This assumes a constant particle diffusion
%   coefficient D and does not account for thermophoresis.  The
%   convection-diffusion equation is solved using the spectral
%   collocation or psuedospectral method.
%
%   function spec_tube_diff_no_Vt_const_D(D,Vmax,R,Len,N,M,debug,graph)
%
%   D:      Particle Diffusion Coefficient (m^2/s)
%   Vmax:   Maximum Velocity along the tube axis (m/s)
%   R:      Radius of the Tube (m)
%   Len:    Length of the Tube (m)
%   N:      Number of Points in the radial direction (Must be odd)
%   M:      Number of Points in the axial direction (Must be odd)
%   debug:  if set to 1 debug information is printed.
%   graph:  if set to 1 the solution will be plotted out.
%
function [u,uu,r,z] = spec_tube_diff_no_Vt_const_D(D,Vmax,R, ...
                                                   Len,N,M, ...
                                                   debug,graph)
% ------------------------------------------------------------------

%   If the user has not input parameters use defaults.
    if nargin < 8
        D = 6.23e-9;
        Vmax = 1;
        R = .001;
        Len = 10;
        N = 41;                     % Radial Direction
        M = 21;                     % Axial Direction
        debug = 1;
        graph = 1;
    end;

    Boundary_in = 1;
    Boundary_Side = 0;

% ------------------------------------------------------------------
%   Create the Chebyshev differentiation Matrices
%       * Note that in the accompanying literature r and z are
%         Xi and Eta respectively.
    [Dr,r] = cheb_derivative(N);              % Radial Direction
    [Dz,z] = cheb_derivative(M);              % Axial Direction

%   Compute the second order Chebyshev differentiation matrix directly
    Dr2 = Dr^2;
% ------------------------------------------------------------------
%   Compute the coefficients and resulting diagonal matrices
% ------------------------------------------------------------------
%   Calculate the first coefficient of the energy equation as a column
%   matrix.
    %a = ((2/Len)*Vmax)*(1 - (R^2)*(r.^2));
    a = ((2/Len)*Vmax)*(1 - (r.^2));
    A = diag(a(1:N+1));

%   Create identity matrix.
    I = eye(M+1);

%   Compute Omega.
    O = (1/(R^2)).*diag(1./r);
% -------------------------------------------------------------------
%   Compute the tensor product spectral grid.
% -------------------------------------------------------------------
    L = kron2(A,Dz) - kron2((1/(R^2)).*Dr2,I).*D - kron2(O*Dr,I).*D;
% -------------------------------------------------------------------
%   Create a column matrix to hold the RHS of the energy equation
% -------------------------------------------------------------------
    size = ((N+1)*(M+1));
    rhs = zeros(size,1);
%   Create two matrices containing the coordinate points inside the
%   solution plane.
    [rr,zz] = meshgrid(r,z);
%   Create one column vector that consists of the two coordinate
%   vectors stacked on top of each other.
    rr = rr(:);
    zz = zz(:);
% -------------------------------------------------------------------
%   Apply the Boundary Conditions
% -------------------------------------------------------------------

%   Find the bounday points of the coordinate vectors:
    b = find(zz==-1 | abs(rr)==1);
%   Impose the boundary conditions on the right side vector.
    rhs(b) = (zz(b)==-1).*Boundary_in + ...
             (rr(b)==1).*Boundary_Side + ...
             (rr(b)==-1).*Boundary_Side;
%   Correct the entry boundary conditions at the top and bottom
%   corners.
    rhs(1) = Boundary_Side;
    rhs(size - M) = Boundary_Side;
%   Apply the boundary conditions to the laplacian
    L(b,:) = zeros(length(b),length(rhs));   % Zero the rows out
    L(b,b) = eye(length(b));                 % put in 1 for known
points
% -------------------------------------------------------------------
%   Solve the Matrix equation
% -------------------------------------------------------------------
    u = L\rhs;
% -------------------------------------------------------------------
%   Plot the results.
% -------------------------------------------------------------------
    uu = reshape(u,(M+1),(N+1));
    if(graph == 1)
        [rr,zz] = meshgrid(r,z);
```

```
%       Transform Xi back to [R,-R] domain
        rr=R*rr;
% -------------------------------------------------------------------
%       Transform the Axial Direction coordinates from (Eta) [1,-1] to
%       z [0,length]
% -------------------------------------------------------------------
%       transpose into column matrix
        z = z';
%       create a matrix of the same size
        a = ones(1, M+1);
        for j = 2: M+1
            a(1,j) = (z(1,1) + z(1,j)) / 2;
        end;
        a(1,1) = 0;
        a = Len*a;
        zz = ones(M+1, N+1);
        for i = 1: N+1
            for j = 1: M+1
                zz(j,i) = a(1,j);
            end;
        end;
% -------------------------------------------------------------------
        figure( ...
            'Name','Tube Flow', ...
            'NumberTitle','off', ...
            'BackingStore','off');

        subplot(2,2,1), surfc(rr,zz,uu),
            xlabel('Radial Distance (m)'),
            ylabel('Axial Distance (m)'),
            zlabel('Particle Concentration'),
            title('Particle Concentration');

        subplot(2,2,2), contourf(rr,zz,uu),
            xlabel('Radial Distance (m)'),
            ylabel('Axial Distance (m)'),
            title('Particle Concentration'),
            colorbar;

        str = sprintf('Particle Diffusion Coefficient: %.9f',D);
        str = strvcat(str, sprintf('Maximum Velocity: %.1f m/s',Vmax));
        str = strvcat(str, sprintf('Tube Radius: %.3f m',R));
        str = strvcat(str, sprintf('Tube Length: %.1f m',Len));
        str = strvcat(str, sprintf('Collocation Points (Radial): %d',…
                                    N));
        str = strvcat(str, sprintf('Collocation Points (Axial): %d', …
                                    M));

        uicontrol('style','text',...
            'units','normalized','position',[0.55 0.05, 0.4, 0.4], ....
            'HorizontalAlignment',['left'],'string',str);

    end;
% -------------------------------------------------------------------
```

7.11 Hinds

```
%
%   This computes the deposition of particles in a cylindrical tube
%   for laminar flow.  A penetration value is returned representing
%   n_out/n_in
%
%   function [p] = hinds_Penetration(mu)
%
function [P] = Hinds_Penetration_mu(mu)

    if mu < 0.009
        P = 1 - 5.50 * mu^(2/3) + 3.77*mu;
    else
        P = 0.819*exp(-11.5*mu) + 0.0975*exp(-70.1*mu);
    end
```

7.12 Spectral Method and Hinds Comparison Code:

```
function Spec_Diff_Test(D,R,Len,num_pts,M,N,graph)

%-----------------------------------------------------------------
    if nargin < 7
        D = 6.23e-9;
        R = 5;
        Len = 10;
        num_pts = 30;
        N = 41;                    % Radial Direction
        M = 41;                    % Axial Direction
        graph = 0;
    end;
    mu = logspace(-4,0,num_pts);
    Q = (D*Len)./mu;
    Vmax = (Q.*2)./(pi*R^2);
    mu = 0;
    debug = 1;
    method = 2;
%-----------------------------------------------------------------
    for i=1:num_pts
        [u,uu,r,z] =
                  spec_tube_diff_no_Vt_const_D(D,Vmax(i),R,Len, …
                                               N,M,debug,graph);
        [P1(i)] = spec_tube_penetration(uu,r,N,M,Vmax(i),R,1,method);
    end;
%-----------------------------------------------------------------
    for i=1:num_pts
        [P2(i), mu(i)] = Hinds_Penetration(D,Vmax(i),R,Len,debug);
    end;
%-----------------------------------------------------------------
    figure( ...
         'Name','Diffusion', ...
         'NumberTitle','off', ...
         'BackingStore','off');
    H1 = semilogx(mu,P1);
    hold on
    H2 = semilogx(mu,P2);
         xlabel('mu'),
         ylabel('Penetration');
    set(H1,'LineStyle','--');
    set(H1,'LineWidth',2);
    set(H1,'color','k');
    set(H2,'LineStyle','+');
    set(H2,'LineWidth',2);
    set(H2,'color','k');
    h = legend('Spectral Method', 'Hinds Method');
    legend(h, 'boxoff');
    hold off

    figure( ...
             'Name','Difference', ...
             'NumberTitle','off', ...
             'BackingStore','off');

    semilogx(mu, abs(P1-P2), 'k'), xlabel('mu'),
                 ylabel('Difference');
```

7.13 Penetration Calculation

```
%
%   This takes in the particle distribution for flow in a tube and the
%   interpolation points in the radial direction.  It then calculates
%   the penetration P (n_out/n_in) of the particle distribution.
%
%   function P = spec_tube_penetration(u,r,N,M)
%
%   n:          Particle Distribution (2D solution)
%   r:          Radial interpolation points
%   N:          Number of Points in the radial direction (Must be odd)
%   M:          Number of Points in the axial direction (Must be odd)
%   Vmax:       Maximum Velocity along the tube axis (m/s)
%   R:          Radius of the Tube (m)
%   rho:        Density Distribution (2D solution)
%
function P = spec_tube_penetration(n,r,N,M,Vmax,R,rho,method)

    num = 0;        % numerator
    N2 = (N+1)/2;
    % Get the denominator
    den = R^2/4;
    % reshape the radial coordinates to go from 0 to R
    r=R.*r;
    r = r(N2+1:-1:1);
    r(1) = 0.0;
    r = reshape(r,1,N2+1);
    % calculate the points along the function to be integrated.
    for i = 1:N2+1
        func(i) = n(1,i+(N2-1)) * (r(i) - r(i)^3/(R^2));
    end; %for i
    num = trapz(r,func);
    P = num/den;
```

7.14 Kronecker Product

```
%   [Z] = kron2(X,Y)
%
%   KRON2 - Kronecker Tensor Product
%
%   KRON2(X,Y) takes the kronecker tensor product of X and Y.  For
%   dense matrices over ~30 elements this has much better performace
%   over MATLAB's KRON.
%
%   5/07
%   -- C. Thibeault, Minnesota State University, Mankato.
%
function [Z] = kron2(X,Y)

%   Calculate the Dimensions
    [X_row, X_col] = size(X);
    [Y_row, Y_col] = size(Y);

%   preallocate the solution Matrix.
    Z = zeros(X_row * Y_row, X_col * Y_col);

    for i = 1:X_row
        for j = 1:X_col
            row_start = (i*Y_row)-(Y_row -1);
            row_end = i*Y_row;
            col_start =j*Y_col-(Y_col-1);
            col_end = j*Y_col;
            Z(row_start:row_end, col_start:col_end) = X(i,j).*Y;
        end % j
    end % i
```

7.15 Chebyshev Derivative Matrix and Collocation Points

```
function [D,x] = cheb_derivative(N)
  if N==0, D=0;
      x=1;
  return, end
  % Collocation Points
  x = cos(pi*(0:N)/N)';
  X = repmat(x,1,N+1);
  dX = X-X';
  % Derivative Matrix
  D=zeros(N+1,N+1);
  % Top and Bottom Rows
  top = (2*(-1).^(1:N-1))./(1-reshape(x(2:N),1,N-1));
  bottom=-top(N-1:-1:1);
  % Left and Right
  right = -reshape(bottom./4,(N-1),1);
  left= -right(N-1:-1:1);
  %top
  D(1,2:N) = top;
  D(N+1,2:N) = bottom;
  D(2:N,1)=left;
  D(2:N,N+1) =right;
  % Middle
  ij = repmat([0:N],N+1,1)+repmat(reshape([0:N],N+1,1),1,N+1);
  mid = ((-1).^ij)./dX;
  D(2:N,2:N) = mid(2:N,2:N);
  % Diagonal
  dia=(1:N+1:(N+1)*(N+1)) + (0:N);
  dia = dia(2:N);
  D(dia) = -(x(2:N)./(2*(1-x(2:N).^2)));
  % Corners
  D(1) = (2*N^2+1)/6;
  D((N+1)*(N+1))=-D(1);
  D(N+1) = -0.5*(-1)^N;
  D((N+1)*(N+1)-(N)) = -D(N+1);
```

8. Appendix B: Polynomial expansion of Helmholtz Equation (Sage Worksheet)

```
var('x a0 a1 a2 u')
```

```
          (x, a0, a1, a2, u)
```

```
u2=1+(1-x^2)*(a0+a1*x+a2*x^2)
```

```
Resid=diff(u2,x,2)+(4)*u2
```

```
eq1=Resid(x=-0.5)
```

```
eq2=Resid(x=0)
```

```
eq3=Resid(x=0.5)
```

```
solve([eq1,eq2,eq3],a0,a1,a2)
```

```
          [[a0 == -18/5, a1 == 0, a2 == 8/5]]
```

```
a_0=-18/5
```

```
a_1=0
```

```
a_2=8/5
```

```
from numpy import arange
```

```
E1=[1+(1-x^2)*(a_0+a_1*x+a_2*x^2) for x in arange(-1,1.1,0.1)]
```

```
p1=list_plot(zip(arange(-1,1.1,0.1),[1+(1-x^2)*(a_0+a_1*x+a_2*x^2) for
x in arange(-1,1.1,0.1)]), rgbcolor=(0,0,0))
```

```
p2=plot(-2.4030*cos(2*z),-1,1,rgbcolor='black')
```

```
(p1+p2)
```

```
E=[(-2.4030*cos(2*x))-(1+(1-x^2)*(a_0+a_1*x+a_2*x^2)) for x in
arange(-1,1.01,0.01)]
```

```
P3=list_plot(zip(arange(-1,1.01,0.01),E), plotjoined=True, rgbcolor=
(0,0,0))
```