# COMPUTER ANALYSIS OF EQUATIONS USING MATHEMATICA

by

**Vikash R Jugoo**

Submitted in partial fulfilment of the
requirements for the degree of
Master of Science
in the
School of Mathematical and Statistical Sciences
University of Natal

**Durban**
**2001**

# Abstract

In this thesis we analyse particular differential equations that arise in physical situations. This is achieved with the aid of the computer software package called Mathematica. We first describe the basic features of Mathematica highlighting its capabilities in performing calculations in mathematics. Then we consider a first order Newtonian equation representing the trajectory of a particle around a spherical object. Mathematica is used to solve the Newtonian equation both analytically and numerically. Graphical plots of the trajectories of the planetary bodies Mercury, Earth and Jupiter are presented. We attempt a similar analysis for the corresponding relativistic equation governing the orbits of gravitational objects. Only numerical results are possible in this case. We also perform a perturbative analysis of the relativistic equation and determine the amount of perihelion shift. The second equation considered is the Emden-Fowler equation of order two which arises in many physical problems, including certain inhomogeneous cosmological applications. The analytical features of this equation are investigated using Mathematica and the Lie analysis of differential equations. Different cases of the related autonomous form of the Emden-Fowler equation are investigated and graphically represented. Thereafter, we generate a number of profiles of the energy density and the pressure for a particular solution which demonstrates that a numerical approach for studying inhomogeneity, in cosmological models in general relativity, is feasible.

*To Srila Prabhupada*
*who taught me to understand a better perspective of life*

# Declaration

The study described in this thesis was carried out in the School of Mathematical and Statistical Sciences, University of Natal, Durban, during the period August 1999 to January 2001. This thesis was completed under the supervision of Dr K S Govinder and Professor S D Maharaj.

The research contained in this study represents original work by the author. It has not been submitted in any form to another University nor has it been published previously. Where use was made of the work of others it has been duly acknowledged in the text.

V R Jugoo

January 2001

# Acknowledgements

I wish to offer my sincere gratitude to the following people for their contributions and sacrifices that made this thesis possible:

- My wife Jeshika for her patience, encouragement and proofreading of this thesis. Furthermore, thank you for being a wonderful person.
- My parents for their ongoing encouragement and support to develop and improve myself. Thank you for the years of financial assistance in the past.
- My sister Nandarani for her proofreading of this thesis and her spiritual guidance.
- My supervisor, Dr Kesh Govinder, for his willingness, enthusiasm and valuable knowledge in numerical mathematics. Thank you for making financial assistance available.
- My co-supervisor Professor Sunil Maharaj, for making this masters possible. Thank you for your words of encouragement and the ability to always calm me down in moments of pressure.
- To both Dr Kesh Govinder and Professor Sunil Maharaj for their dedication, tolerance, kindness and most of all patience in helping me produce this thesis.
- The National Research Foundation and the University of Natal, Durban for financial support during my course of study.
- Nikasha Sankar for borrowing library books at short notice and for paying my fines.
- Professor J Banasiak for interesting discussions on Mathematica.
- My Head of Department at Mangosuthu Technikon Mrs K Reddy for providing me with a laptop which proved to be a life saver.

# Contents

# Chapter 1

# Introduction

A large proportion of physical, chemical and industrial processes are governed by differential equations. If these equations accurately represent natural phenomena, they are invariably highly nonlinear and hence become difficult to solve using conventional analytic techniques. In order to accurately represent these solutions we need to utilise numerical methods to augment existing conventional methods. Due to time constraints, we look for quick yet accurate solutions. Fortunately, with technology advancing at such a rapid rate, we have electronic devices to help us achieve this task. With the introduction of the computer many tasks that were manually performed are now being carried out faster and more accurately. The great speed of computers these days coupled with the development of advanced software packages allows one to perform basic word processing, advanced computer animation or even solve complicated equations from mathematics. Nowadays it has become popular to use mathematical packages in performing complex calculations since answers can be obtained within seconds. One such program that helps in this regard is the software package called Mathematica. The following quote by Blachman (1992) summarises the thrust of this program:

"As computer power and memory have increased, so have the capacity of software developers to write programs to assist people with time

consuming tasks. Mathematica is such a program. It works problems that are impractical to do by hand, freeing your time for your own work. Mathematica is a useful tool for those who do quantitative analysis, symbolic calculations and manipulations, as well as for those who want to visualize functions or data. With it you can calculate, model, prototype, and analyze results".

Mathematica is an interpreted language. In other words, it reads an expression, evaluates the result and then displays the output. Being interactive makes it easier to use than programming languages, such as Pascal, C or Fortran. However, Mathematica is also flexible in that it is programmable. Any function that is not predefined may be programmed by the user. Mathematica offers many primitives and constructs found in programming languages such as Pascal, C, etc. In addition to procedural programming, Mathematica supports rule based programming. This software package works on many different operating systems and can be installed on a personal computer or a network.

In chapter 2 of this thesis, we present a basic review of the software package Mathematica focusing on its structure and capabilities. In §2.1 we provide a brief background on Steven Wolfram who was instrumental in the creation of this software package. In §2.2 we provide a broad outline emphasising the requirements of this program for various platforms together with the underlying components that form the driving force of this software package. The graphical user interface is explored in §2.3 with some of the computational features explained in §2.4. Thereafter the resources are highlighted in §2.5 and in §2.6 we point out the need for additional packages. A detailed description of Mathematica is given in §2.7 and we highlight some of those functions which are used often in practical calculations in the form of an interactive tour. Commands that are frequently used in this thesis are discussed in §2.8.

In chapter 3, we make use of the commands that were discussed in Chapter 2 to analytically solve the Newtonian equation governing the orbits of planetary bodies.

This solution is utilised to plot the planetary orbits of Mercury, Earth and Jupiter using the `PolarPlot` command in §3.2. The Newtonian equation is then adapted to the corresponding relativistic equation in §3.3. Again we attempt to obtain analytical solutions and graphs for the planetary orbits of Mercury, Earth and Jupiter. In §3.4 numerical solutions are found which are compared with the analytical results. A perturbative analysis is performed on the relativistic equation and the amount of perihelion shift is obtained in §3.5.

In chapter 4 of this thesis, we analyse the second order Emden-Fowler equation which arises in certain cosmological applications. In §4.2 analytic features of this equation are reviewed using the Lie analysis of differential equations. The solution of the Emden-Fowler equation is reduced to a simpler second order autonomous equation. The integrability of this reduced equation is comprehensively investigated by considering plots of all the sub cases that arise. In §4.3 we generate plots of the energy density and pressure for a range of values corresponding to particular radii and time values for different cases that arise in our treatment.

In Chapter 5, we briefly summarise the results of this thesis.

# Chapter 2

# Mathematica

## 2.1 The Developer

Mathematica is a computer algebra system for mathematical problems. It is currently used by thousands of researchers, engineers and analysts, as well as students from high school to graduate school. Mathematica was released on June 23, 1988 by Steven Wolfram. Copyright of this product is the property of Wolfram Research Inc. which is a privately held company based in Champaign, Illinois. Wolfram initially developed software for scientific computing in the late 1970s with development of Mathematica starting only in 1986. The developer, a well known scientist in the academic community, achieved the status of full Professor in the fields of physics, mathematics and computer science at the University of Illinois. He has received wide acclaim for his research contributions in cellular automata and complexity.

Whilst the original concept and the kernel were created by Steven Wolfram, there is a host of other individuals who were responsible for the development of the Mathematica software package. For example, the front end concept was created by Theodore W. Gray. For more information with regard to the design and development of the program, the reader is referred to the `About Mathematica...` option which can be

found on the Help menu of the program and the comprehensive Mathematica manual (Wolfram 1991).

## 2.2 Overview of Mathematica

Mathematica first became popular in the fields of physical sciences, engineering and mathematics. Now it is also used in financial analysis, medical research, computer science and education (Crandall 1991). Whilst Mathematica is predominantly used for mathematics, a small percentage of users utilise it for creating graphics, prototyping computer systems and generating interactive documents. Mathematica belongs to a suite of symbolic manipulation packages which include Maple, Mathcad and Matlab. However, Mathematica has become the most commonly used software because of its many attractive features. The environment caters for numerical, symbolic and graphical exploration allowing arithmetic and algebraic calculations.

The software is available on one CD and the full installation takes up just over 150 megabytes of hard disk space. Of the 150 megabytes, 100 megabytes are optional and includes the online version of the Mathematica book, the standard add-on packages and the MathLink developer's kit. Mathematica can work on a 80386 processor or later. It requires a minimum of 16 megabytes of RAM, although 24 megabytes is recommended for operation in Microsoft Windows 95/98 (32 megabytes for Windows NT).

Wolfram Inc. has a sophisticated password protection system. It works on the following principle: every computer has a unique `MathID` number, which is automatically generated during the installation process. This `MathID` number together with the licence number must then be e-mailed to Wolfram Inc., and a corresponding password is returned to the user in order to complete the installation process. The response time to obtain a password is reported to be approximately 48 hours; however our interactions have generated a much faster response. Alternatively, the `MathID` number

can be entered on the web and the password generated immediately.

The program is available for most popular operating systems including Sun, Apple Macintosh, Windows 95/98 and Windows NT 4.0 or later. It can work on a standalone machine or a network. The front end, which provides the user interface, is a separate software component from the kernel which does the computations. As a result, the two components may be installed on different computers across a network.

There have been many different versions of Mathematica dating back to DOS (Disk Operating System). Now a completely graphical user interface is available which is more user friendly and much easier to use. The latest version released is Mathematica version 4.1 (December 2000). However, in this thesis comments will be made concerning the features of Mathematica version 4.0.

Mathematica is divided into two parts: the kernel and the front end. The kernel is the computational engine that calculates and computes the results. The front end, also known as the notebook interface, is the graphical user interface to the Mathematica kernel and is separate from the kernel. The notebook handles formatting of text and sends a calculation to the kernel only when instructed. This is achieved when the user enters the keystrokes `<shift> + <enter>`. The kernel is where the actual computations take place. It may run on the same machine as the front end, or on a different machine. This is possible because Mathematica has its own communications standard called MathLink. MathLink also allows Mathematica to interface with other external programs, however, this standard is not foolproof.

This software package contains many built-in functions. These functions are based on a number of algorithms from computer science and mathematics. Although Mathematica contains hundreds of functions, the developers realised that they could not anticipate the needs of all users. Therefore, it supports its own high level programming language to produce user defined algorithms. Mathematica allows users to carry out projects that would be extremely laborious in traditional programming environments by integrating the built-in functions with the programming language.

The application program of this software package is developed from a combination of the Mathematica language itself and the programming language C.

Once in Mathematica, help can be obtained on built-in functions, add-ons, getting started/demos and other information. Help can also be obtained in the form of a master index. Furthermore the Mathematica book is contained on-line for explanations, detailed examples and syntax on built-in functions.

In order to continue working on a previously saved notebook, it must be retrieved and evaluated (i.e. each line of the input processed by the kernel) in order to use the previous variables or values in the new Mathematica session. At the end of any Mathematica session, the notebook can be saved for future use and the program may be terminated using the `Exit` option on the `File` menu.

## 2.3 User Interface

When Mathematica starts up, it usually presents a blank notebook that consists of a sequence of input and output cells that are produced in pairs. (In what follows we maintain the input in the Mathematica format while the output is typeset to make the results easily read.) A cell is where a Mathematica command is entered or where output is received from the program. Within a particular cell, we can use the standard positioning and editing capabilities of the graphical user interface. Each time a calculation is sent to the kernel for processing, Mathematica assigns a number $n$ in sequence for each input and output. For example, the first input in a Mathematica session is labelled `In[1]:=` and the corresponding output is labelled `Out[1]=`. Note that the `In[n]:=` and `Out[n]=` only appear after the `<shift>` + `<enter>` combination has been pressed.

Mathematica has standard editing features to prepare input which may cover several lines. The program will automatically continue reading successive lines until it has received a complete expression. Thus, if the user types an opening parenthesis on

one line, Mathematica will continue reading successive lines of input until it finds the corresponding closing parenthesis.

The basic program file, or the notebook, is in ASCII text. This allows us to copy and paste into other applications such as Microsoft Word. Furthermore, notebooks containing text and graphics can be sent via electronic mail without corruption.

Mathematica contains palettes which are like an extension of the keyboard. There are many different palettes that have buttons that can be selected to produce the syntax of built-in functions and special symbols. One such palette allows for the use of Greek letters. Input can also be captured using special keys on the keyboard. Pressing one of these keys does not lead to an ordinary character being entered, but instead causes some action to occur or some structure to be created. For example, typing the text `\[Alpha]` produces the Greek letter $\alpha$. Alternatively, the keystrokes `Esc a Esc` also produces an $\alpha$ and the keystrokes `Esc != Esc` produces the $\neq$ symbol. Both input and output can produce Greek letters and other special symbols.

The first calculation that is processed in Mathematica takes a long time to be evaluated even though it may be a simple calculation. This occurs because the software loads into memory in two stages. When the Mathematica icon is clicked, the user interface portion of the program is loaded. The kernel is only loaded once `<shift> + <enter>` is pressed to process the first calculation. However, one can set up the program to load the kernel as soon as the package is initialised.

## 2.4 Computational Features

There are three classes of Mathematica's computations: numerical, symbolic and graphical. Once Mathematica is loaded, it can be used to perform simple arithmetic like a calculator. However, Mathematica can evaluate expressions with a much higher degree of precision than traditional calculators. It also has the capability of performing operations on functions, manipulating algebraic formulas and evaluating expres-

sions in calculus. Both two and three dimensional graphs can also be produced. We discuss some of the features below.

Numerical computation:

Mathematica can perform all the functions such as addition, multiplication, trigonometric functions etc. which one would find on a standard scientific calculator. These calculations can be performed to any desired number of decimal places. It also has a much larger library of built-in functions available which may be used in the numerical evaluation of definite integrals. `NDSolve` is a numerical function that is frequently used in this thesis.

Symbolic computation:

Mathematica has a rich set of symbolic manipulation routines. Most of the arithmetic operators and standard mathematical functions can work with symbolic expressions as well as with numeric expressions. These symbolic routines include algebraic manipulation and simplification, factorisation and expansion, as well as differentiation and integration. In this thesis `DSolve` is a symbolic function that is frequently used.

Graphics:

The graphical capabilities of Mathematica is one of the qualities that have contributed greatly to its success. It is very easy to create two-dimensional, three-dimensional, contour and density plots and draw arbitrary figures and objects. There are also powerful features to control the appearance of the graphical results. Some high-level routines that are available for generating graphical representations are `Plot`, `ListPlot` and `PolarPlot`. These routines are frequently used in this thesis. Furthermore, Mathematica provides a useful routine to view many graphs on one set of axis via the `Show` command enabling an easy comparison of results. Another interesting graphical feature of this software is animation. For more information on graphics the reader is referred to Gaylord and Wellin (1995), Gray and Glynn (1991) and Smith and Blachman (1995).

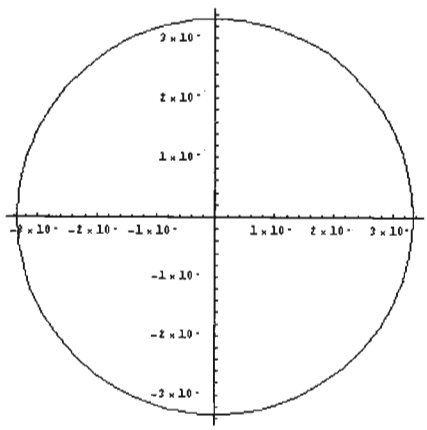
Mathematica also allows graphics to be exported to other software packages. This is achieved using the `Display` command which allows for the export of graphics in a specified format. The syntax is

```
Display[path, graphics, "format-type"]
```

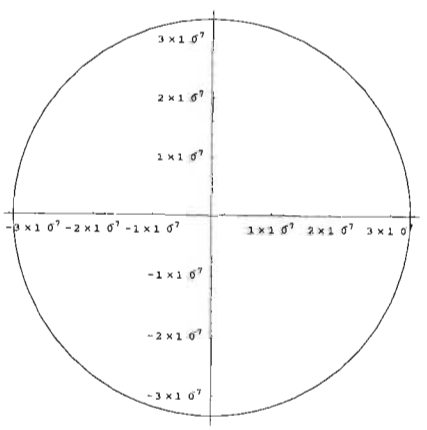
where `path` is the location where the graphic is to be stored, `graphics` refers to the object we wish to export and `format-type` is the format of the output that is required. The two formats that were considered for the exporting of graphs in this thesis were of type bitmap and metafile. Below is a graph using both the formats for the expression

$$\frac{1}{3 \times 10^6 + \sin(1 - x)}$$

over the range $(-\pi, \pi)$. The bitmap format produces



and the metafile format results in



Comparing the graphs above we find that the metafile format produces a smoother curve than the bitmap format. Furthermore, the axis labelling of the indices for the bitmap format is not clear and as the result all graphs in subsequent sections were exported using the metafile format.

Programming:

Mathematica contains all the necessary features such as variables, loops and conditional branches enabling it to be used as a high level programming language. It also includes a range of programming paradigms such as procedural, list-based, functional, rule-based, string-based and object-oriented programming.

Typesetting:

Mathematica is not only a mathematics package, it also has all the features of a text editor combined with calculations. It can be used to create professional documents as well as interactive presentations.

Help:

Typing `?command` at the Mathematica prompt will display help on the topic `command`. Typing `??command` will display a more explicit explanation of `command`. For example `?Solve` at the prompt will display

```
Solve[eqns, vars] attempts to solve an equation or set of equations
for the variables vars. Solve[eqns, vars, elims] attempts to solve
the equations for vars, eliminating the variables elims.
```

and `??Solve` will display

```
Solve[eqns, vars] attempts to solve an equation or set of equations
for the variables vars. Solve[eqns, vars, elims] attempts to solve the
equations for vars, eliminating the variables elims.
```

```
Attributes[Solve] = {Protected}

Options[Solve] = {InverseFunctions -> Automatic, MakeRules -> False,
Method -> 3, Mode -> Generic, Sort -> True, VerifySolutions ->
Automatic, WorkingPrecision -> ∞}
```

Furthermore using the command `?L*` displays information on all Mathematica objects with names beginning with `L`. When there is more than one object, Mathematica will only list the object names. Another way to obtain information on Mathematica functions is by using the `Options` command. `Options[object]` returns a list of the available options associated with the object.

Complete selections:

The notebook interface provides a feature to reduce the amount of typing involved in entering Mathematica input. This is called the `Complete Selection` command. If part of a name typed is known to Mathematica, then using the key strokes `<Ctrl> + <k>` will either complete the name if it is unique or provide a menu of possible completions.

Errors:

Mathematica usually goes about its work silently, giving output only when it is finished performing the calculations requested. The input entered must follow a definite syntax. For example, input like `4 +/ 5` does not follow the syntax and cannot be processed by Mathematica. In this instance Mathematica will reject the input. Typically, the computer will bleep and display the error message

```
Syntax::sntxf: "4 +" cannot be followed by "/5".
```

Similarly, processing the command `Sqrt[4,5]` will display

```
Sqrt::argx: Sqrt called with 2 arguments; 1 argument is expected.
```

Interrupt evaluation:

It is possible to stop Mathematica in the middle of a calculation if it is taking too long to produce a result, using the `Interrupt Evaluation...` option which can be found on the `Kernel` menu of the program. Alternatively, the following shortcut keystroke can be used to achieve the same result `<Alt> + <,>`

Bracket matching:

Mathematica has a colour based assistant which ensures that left and right brackets are correctly matched. This feature is especially useful in complicated expressions and commands. For example, by typing `Plot[(x-2)^2,{x,0,1}]`, each left bracket is echoed to the screen in mauve, until it turns to black when the matching right bracket is typed. It is therefore easy to identify brackets that are not matched. However, the assistant does not prevent attempting something mathematically acceptable, but syntactically incorrect. For example, `sin(x)` will be accepted as valid in spite of the correct syntax being `Sin[x]`.

Lists:

Mathematica frequently makes use of Lists. A list is a collection of objects separated by commas and enclosed in braces. In performing calculations, it is often convenient to collect together several objects, and treat them as a single entity. Arithmetic can then be performed on the whole list at once, or by assigning the whole list to be the value of a variable. To gain a better understanding of lists the reader is referred to Abel and Braselton (1994).

Using previous results:

Very frequently we need to use the results that were obtained from the previous calculation. Mathematica allows us to use the result obtained from the previous output using the % symbol, which stands for the last result. For example

```
In[1]:= 15^4 + 150
Out[1]= 50775

In[2]:= % * 2
Out[2]= 101550
```

Using %% refers to the next-to-last result. Similarly, the statement

```
In[3]:= %1 + %2
Out[3]= 152325
```

will add the output generated in the lines 1 and 2.

Replacement rule:

When Mathematica simplifies an expression such as $x + x$ into $2x$, it is treating the variable $x$ in a purely symbolic manner. In such cases, $x$ is a symbol which can stand for any expression. However, there are some instances when we need to replace a symbol like $x$ with a definite value which may be a number or another expression. The two symbols that are used to perform this task is /. which is the replacement operator and -> which is the assignment operator. For example consider the expression $1 + 2x$. In Mathematica this would be

```
In[4]:= 1 + 2x
Out[4]= 1 + 2x
```

Now to replace the symbol $x$ in the previous expression with the value 3, we perform

```
In[5]:= % /. x -> 3
Out[5]= 7
```

## 2.5 Mathematica Resources

Apart from the online help, there are many other sources that provide supplementary material on Mathematica. These are in the form of books, application packages and the world wide web. Wolfram Inc. provides a well established web site (*www.wolfram.com*) where one can obtain information on their products, services, solutions, resource library, news, on-line store, the company and other relevant information. Any errors that are found within the application program can be e-mailed to the technical support services of Wolfram Inc. One can expect a twenty four hour response time.

Included on their web site is a MathSource hyperlink which is a vast electronic library of Mathematica materials including Mathematica programs, documents and examples. This facility allows the user to either browse the archive or search by author, title, keyword or item number. All items are in the public domain.

A Mathematica Service subscription grants one access to technical support services and discounts on various Mathematica products and services. Furthermore, there are over 200 books and journals in 14 languages that cover topics from the basics to specific mathematical applications in engineering, physics, education and more.

## 2.6 Add-on Packages

In order to perform most calculations in Mathematica the standard built-in functions are sufficient. However, there are more specialised functions that are not built-in. These functions are contained in Mathematica packages, which are files written in the Mathematica language and contain a collection of algorithms.

There are many different types of packages that contain a large number of additional commands that are shipped with each version of Mathematica. New and updated packages can be obtained from MathSource and user groups via the world wide web,

whilst experienced users can program their own packages. In order to view more information on Mathematica packages, the reader is referred to Maeder (1990) and Wolfram Research (1993).

In order to load a package, the name as well as the path must be specified using the following syntax

```
<<directory`packagename`
```

where `directory` is the location of the package and `packagename` refers to its name. For example

```
In[1]:= <<DiscreteMath`CombinatorialFunctions`
```

reads in the package that contains various combinatorial functions. One of the functions within the `DiscreteMath` package is called `Subfactorial` and can be used in the following way after loading the package

```
In[2]:= Subfactorial[10]
Out[2]= 1334961
```

## 2.7 A Tour of Mathematica

Before we commence our tour of Mathematica, we comment on its use of brackets. Each kind of bracketing has a very different meaning. For example

| | |
|---|---|
| $()$ | parentheses for grouping of terms |
| $f[x]$ | square brackets delimit the arguments of functions |
| $\{a, b, c\}$ | curly braces for lists |
| $v[[i]]$ | double brackets for indexing of lists |

This is important in clarifying the syntax of what follows.

Mathematica can be used as a simple calculator

```
In[1]:= 2+3
Out[1]= 5
```

Unlike a calculator, Mathematica can produce results to a high degree of precision

```
In[2]:= 3^100
Out[2]= 515377520732011331036461129765621272702107522001
```

Mathematica returns approximate numerical results just as a simple calculator would by making use of the `//N` operator,

```
In[3]:= 3^100 //N
```
Out[3]= $5.15378 \times 10^{47}$

We can find solutions to a polynomial by using the `Solve` command

```
In[4]:= Solve[x^3 - 3 x^2 - 17 x + 51 == 0, x]
```
Out[4]= $x \to 3,\ x \to -\sqrt{17},\ x \to \sqrt{17}$

Products and positive integer powers can also be expanded using `Expand`

```
In[5]:= Expand[(1+x)^2]
```
Out[5]= $1 + 2x + x^2$

A variation of `Expand` is `PowerExpand` which evaluates all powers of products and powers unevaluated by `Expand`

```
In[6]:= Expand[Sqrt[(x y)^2]]
```
Out[6]= $\sqrt{x^2 y^2}$

Without more knowledge of $x$ and $y$, Mathematica cannot proceed any further. However, using

```
In[7]:= PowerExpand[Sqrt[(x y)^2]]
```
Out[7]= $xy$

produces an output one frequently requires. One must exercise extreme caution in the use of this command as $\sqrt{(1-x)^2}$ and $\sqrt{(x-1)^2}$ will always give different results.

Simplifying polynomials can be performed using the `Simplify` command

```
In[8]:= Simplify[x^2 + 2x + 1]
```
$$\text{Out[8]} = (1+x)^2$$

`Simplify` is designed to attempt various standard algebraic transformations on the given expression. However, it can take more sophisticated transformations to make progress in finding the simplest form of an expression. `FullSimplify` tries a much wider range of transformations, involving not only algebraic functions, but many other kinds of functions as well. For fairly small expressions, `FullSimplify` will often succeed in making some remarkable simplifications. However, for larger expressions it often becomes unmanageably slow. Consider

```
In[9]:= Simplify[Sqrt[22619537 + 15994428 Sqrt[2]] (1 - sin[x]^2)]
```
$$\text{Out[9]} = \sqrt{22619537 + 15994428\sqrt{2}}\ \cos^2(x)$$

The `Simplify` command is not able to reduce this expression to its simplest form. However, using the `FullSimplify` command

```
In[10]:= FullSimplify[Sqrt[22619537 + 15994428 Sqrt[2]] (1 - sin[x]^2)]
```
$$\text{In[10]} := (3363 + 2378\sqrt{2})\cos^2(x)$$

produces a more concise result.

Mathematica can perform symbolic integration and differentiation

```
In[11]:= Integrate[x sin[x]^2, x]
```
$$\text{Out[11]} = \frac{x^2}{4} - \frac{\cos(2x)}{8} - \frac{x\sin(2x)}{4}$$

```
In[12]:= D[%, x]
```
$$\text{Out[12]} = \frac{x}{2} - \frac{x\cos(2x)}{2}$$

Note that the result in `Out[12]` is not in its simplest form. Mathematica does not automatically simplify an algebraic expression like this. The `Simplify` command is required thereafter

```
In[13]:= Simplify[%]
```
$$\text{Out[13]} = x\sin^2(x)$$

To perform long calculations, it is often convenient to specify names to intermediate results, as is done in standard mathematics or other computer languages

```
In[14]:= x=5
Out[14]= 5

In[15]:= x^2
Out[15]= 25
```

The contents of a variable can be freed by using the `Clear` command

```
In[16]:= Clear[x]
```

In order to store numbers in a list we use

```
In[17]:= mylist = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100}
Out[17]= {10, 20, 30, 40, 50, 60, 70, 80, 90, 100}
```

The above list contains ten objects. To access the fifth element of the list, we use double square brackets to enclose the index

```
In[18]:=  mylist[[5]]
Out[18]= 50
```

Arithmetic can be performed on the entire list in the following way

```
In[19]:= mylist + 1
```
$$\text{Out[19]} = \{11, 21, 31, 41, 51, 61, 71, 81, 91, 101\}$$

Mathematica has a built-in function called `Table` that can be used to generate lists

```
In[20]:=  Table[n!, {n,1,8}]
```

Out[20]= $\{1, 2, 6, 24, 120, 720, 5040, 40320\}$

The above statement creates a list containing the factorial of the numbers within the range one to eight. Note that the second parameter of `Table` is itself a list. It is a special kind of list called an iterator.

A matrix can be generated using the command

```
In[21]:=  twod = {{a, b}, {c, d}}
```

Out[21]= $\{\{a, b\}, \{c, d\}\}$

Note that a matrix (and hence an array) is a nested list. In order to obtain the first row we use

```
In[22]:= twod[[1]]
```

Out[22]= $\{a, b\}$

Now, to obtain the second element in the first row we type

```
In[23]:= twod[[1,2]]
```

Out[23]= $b$

A command related to lists is the `Flatten` command. This command flattens out a list thereby deleting the inner braces.

```
In[24]:= Flatten[%21]
```

Out[24]= $\{a, b, c, d\}$

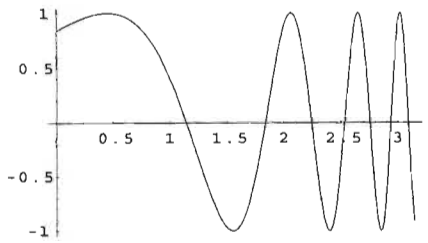
Complex numbers can also be used by including the symbol I, which represents $\sqrt{-1}$

```
In[25]:= Sqrt[-4]
```

Out[25]= 2I

It is easy to represent graphs. The function $\sin(e^x)$ is illustrated via the command

```
In[26]:= Plot[sin[Exp[x]], {x, 0, Pi}]
```



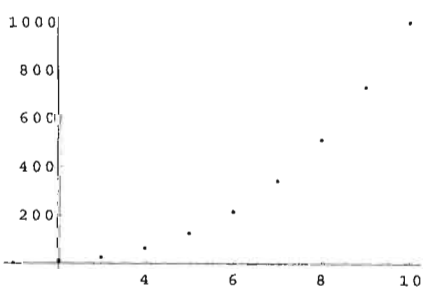

```
Out[26]= -Graphics-
```

Another type of graph that can be used to represent a list of numbers is `ListPlot`. It takes a list of $y$ values or a pair of $(x, y)$ values to produce a graph. This is particularly useful in the graphical representation of numerical results. Consider the sequence of numbers generated using the `Table` command

```
In[27]:= Table[i^3, {i, 10}]
Out[27]= {1, 8, 27, 64, 125, 216, 343, 512, 729, 1000}
```

The previous set of values can be graphically represented using the command

```
In[28]:= ListPlot[%]
```

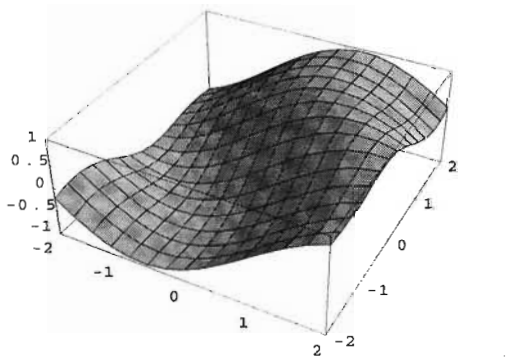



```
Out[28]= -Graphics-
```

These points can be joined using the `PlotJoined->True` option in `ListPlot`.

Three dimensional graphs can also be obtained. For example plotting a three dimensional graph for the function $\sin(x + \sin(y))$ is obtained via

```
In[29]:= Plot3D[sin[x + sin[y]], {x, -2, 2}, {y, -2, 2}]
```




```
Out[29]= -SurfaceGraphics-
```

Mathematica literally has thousands more commands to perform various calculations. This tutorial simply provides a little insight on the capabilities of Mathematica. For an introduction, the reader is referred to Blachman (1992), Gray (1994), Gray and Glynn (1992) and Wagon (1991).

## 2.8 Commands Frequently Used in this Thesis

`DSolve`:

This is a symbolic function that is used to solve differential equations and contains variable parameters. The syntax is

```
DSolve[eqn, y, x] solves a differential equation for the function y,
with independent variable x.
DSolve[{eqn1, eqn2, ... }, {y1, y2, ... }, x] solves a list of
```

```
differential equations.
DSolve[eqn, y, {x1, x2, ... }] solves a partial differential equation.
```

An example using `DSolve`

```
In[1]:= DSolve[y'[x]==ay[x],y[x],x]
```

Out[1]= $y(x)$ -> $e^{ax}C_1$

where $C_1$ is a constant of integration. Note that the solution that is returned from `DSolve` is given as a rule i.e. Mathematica makes use of the symbol `->`.

`NDSolve`:

We can find numerical solutions to differential equations using the function `NDSolve`. This function can handle both single differential equations and systems of differential equations. When using `NDSolve`, initial conditions must be specified. These initial conditions must be sufficient to determine the solution. The results of `NDSolve` are given in terms of interpolating functions.

The syntax is

```
NDSolve[eqns, y, {x, xmin, xmax}] finds a numerical solution to the
ordinary differential equations eqns for the function $y$ with the
independent variable $x$ in the range xmin to xmax.
```

The initial conditions are included in `eqns`.

```
NDSolve[eqns, y, {x, xmin, xmax}, {t, tmin, tmax}] finds a numerical
solution to the partial differential equations eqns.
NDSolve[eqns, {y1, y2, ... }, {x, xmin, xmax}] finds numerical solutions
for the functions yi.
```

There are a number of parameters that are associated with `NDSolve`. These are listed below

| Option | Default value | Explanation |
|---|---|---|
| AccuracyGoal | Automatic | digits of absolute accuracy sought |
| Compiled | True | whether to compile the original equations |
| InterpolationPrecision | Automatic | the precision of the interpolation data returned |
| MaxSteps | Automatic | maximum number of steps to take |
| MaxStepSize | Infinity | maximum size of each step |
| PrecisionGoal | Automatic | digits of precision sought |
| StartingStepSize | Automatic | initial step size used |
| WorkingPrecision | $Machine Precision | the number of digits used in internal computations |

NDSolve follows the general procedure of reducing the step size until it tracks solutions accurately. There is a problem when the true solution has a singularity. In this case NDSolve may continue reducing the step size indefinitely. To avoid this problem the default setting for MaxSteps is 1000 for ordinary differential equations and 200 for partial differential equations. NDSolve stops when either the AccuracyGoal or the PrecisionGoal specified is met. The default setting of Automatic for AccuracyGoal and PrecisionGoal yields goals equal to the setting for WorkingPrecision minus ten digits. AccuracyGoal effectively specifies the absolute error allowed in solutions, while PrecisionGoal specifies the relative error. If solutions must be followed accurately when their values are close to zero, AccuracyGoal should be set larger, or to infinity. The setting for InterpolationPrecision specifies the number of digits of precision to use inside the InterpolatingFunction object generated by NDSolve. The default setting of Automatic for InterpolationPrecision uses the current setting for WorkingPrecision. When NDSolve evaluates a particular set of differential equations, it always tries to choose a step size appropriate for those equations. In some cases, the very first step size that NDSolve takes may be too

large and it may miss an important feature in the solution. To avoid this problem, the `StartingStepSize` option can be set explicitly to specify the size to choose for the first step.

Mathematica represents numerical approximations to functions as `InterpolatingFunction` objects. The `InterpolatingFunction` stores a table of values for $y(x_i)$. It then interpolates this table to find an approximation of $y(x)$ at the particular value of $x$. Suppose that we wish to approximate the solution to the system

$$x' = -y - x^2 \tag{2.1}$$

$$y' = 2x - y \tag{2.2}$$

subject to the conditions $x(0) = y(0) = 1$ and $0 \leq t \leq 10$. The Mathematica command is

```
sol = NDSolve[{x'[t] == -y[t] - x[t]^2, y'[t] == 2 x[t] - y[t],
      x[0] == y[0] == 1}, {x[t], y[t]}, {t, 0, 10}]
```

which returns the following solutions that are stored in the variable `sol`

$$x(t) = \text{InterpolatingFunction}(0, 10) \tag{2.3}$$

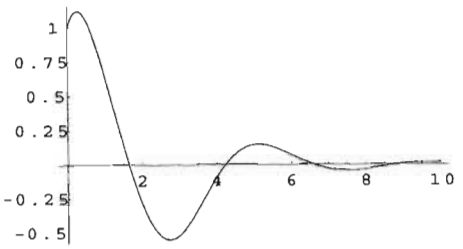
$$y(t) = \text{InterpolatingFunction}(0, 10) \tag{2.4}$$

`Plot`:

Many graphical representations are illustrated in this thesis. The commands that are frequently used are `Plot` and `PolarPlot`. A graph can be obtained for (2.4) using the command

```
Plot[Evaluate[y[t] /. sol], {t, 0, 10}]
```

where `sol` represents the solution in (2.4). This command produces the following graph in Mathematica




The syntax for `Plot` is

```
Plot[f, {x, xmin, xmax}] generates a plot of f as a function of x
from xmin to xmax. Plot[{f1, f2, ... }, {x, xmin, xmax}] plots several
functions fi.
```

`PolarPlot`:

This command is used when plotting the radius as a function of the angle. It is not a built-in function and is contained in a package called `Graphics`. It is loaded using the command

```
<<Graphics`Graphics`
```

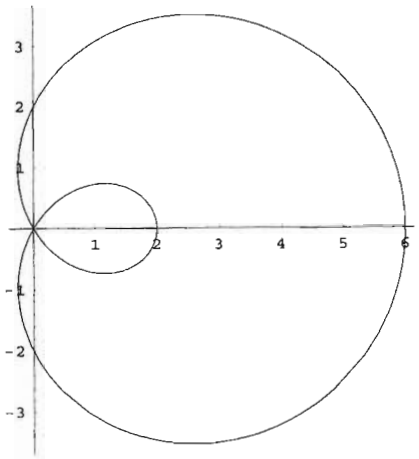
The syntax for `PolarPlot` is

```
PolarPlot[r,{t,min,tmax}] generates a polar plot of the radius r as a
function of the angle t.
```

An example using `PolarPlot`

```
In[2]:= <<Graphics`Graphics`
```

```
In[3]:= PolarPlot[4 cos[t] - 2, {t, 0, 2 Pi}]
```




```
Out[3]= -Graphics-
```

RKSolve:

The RKSolve function is an implementation of the Runge-Kutta method for finding numerical solutions to ordinary differential equations. This function was used to verify all the results obtained in Chapter 3 and some of the results in Chapter 4 via NDSolve. This function is not a built-in function and is contained in the package RungeKutte.m. The syntax for RKSolve is as follows

```
RKSolve[{e1,e2,..}, {y1,y2,..}, {a1,a2,..}, {t1, dt}] numerically
integrates the ei as functions of the yi with initial values ai. The
integration proceeds in steps of dt from 0 to t1. RKSolve[{e1,e2,..},
{y1,y2,..}, {a1,a2,..}, {t, t0, t1, dt}] integrates a time-dependent
system from t0 to t1.
```

The algorithmic code for the RKSolve routine is listed below

```
BeginPackage["ProgrammingInMathematica`RungeKutta`"]

RKSolve::usage =
"RKSolve[{e1,e2,..}, {y1,y2,..}, {a1,a2,..}, {t1, dt}]
```

```
numerically integrates the ei as functions of the yi with initial values ai.
The integration proceeds in steps of dt from 0 to t1.
RKSolve[{e1,e2,..}, {y1,y2,..}, {a1,a2,..}, {t, t0, t1, dt}] integrates
a time-dependent system from t0 to t1."
Begin["`Private`"]

RKStep[f_, y_, y0_, dt_] :=
Module[{ k1, k2, k3, k4 },
k1 = dt N[ f /. Thread[y -> y0] ];
k2 = dt N[ f /. Thread[y -> y0 + k1/2] ];
k3 = dt N[ f /. Thread[y -> y0 + k2/2] ];
k4 = dt N[ f /. Thread[y -> y0 + k3] ];
y0 + (k1 + 2 k2 + 2 k3 + k4)/6
]
RKSolve[f_List, y_List, y0_List, {t1_, dt_}] :=
NestList[ RKStep[f, y, #, N[dt]]&, N[y0], Round[N[t1/dt]] ] /;
Length[f] == Length[y] == Length[y0]
RKSolve[f_List, y_List, y0_List, {t_, t0_, t1_, dt_}] :=
Module[{res},
res = RKSolve[ Append[f, 1], Append[y, t], Append[y0, t0], {t1 - t0, dt} ];
Drop[#, -1]& /@ res
] /; Length[f] == Length[y] == Length[y0]
End[]
Protect[ RKSolve ]
EndPackage[]
```

An example using `RKSolve` is given below:

Consider the following equation

$$y'' = 0 \tag{2.5}$$

with initial conditions $y'(0) = 1$ and $y(0) = 0$. We make the following substitution

$$v = y' \tag{2.6}$$

thereby making

$$v' = y'' \tag{2.7}$$

Substituting (2.7) into (2.5) we have

$$v' = 0 \tag{2.8}$$

We first load the package in order to use the `RKSolve` command

```
In[4]:= << RungeKutta.m
```

Using the syntax already given for `RKSolve`, the Mathematica command for (2.5) with the given initial conditions, translates to

```
In[5]:= RKSolve[{v, 0}, {y, v}, {0, 1}, {x, 0, 10, 1}]
Out[5]= {{0., 1.}, {1., 1.}, {2., 1.}, {3., 1.}, {4., 1.}, {5., 1.},
          {6., 1.}, {7., 1.}, {8., 1.}, {9., 1.}, {10., 1.}}
```

Plotting the above data directly results in a graph in phase space. However, we are interested in the behaviour of $y$ as a function of $x$. As a result we proceed by flattening out the nested lists above with the following command

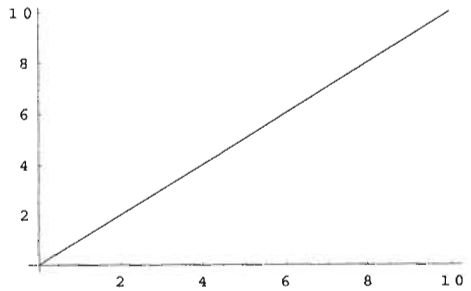
```
In[6]:= Flatten[%]
Out[6]= {0., 1., 1., 1., 2., 1., 3., 1., 4., 1., 5., 1., 6., 1., 7.,
          1., 8., 1., 9., 1., 10., 1.}
```

A table of values is then generated in order to obtain a graphical representation via the command

```
In[7]:= Table[{i, %[[2i + 1]]}, {i, 0, 10}]
Out[7]= {{0, 0.}, {1, 1.}, {2, 2.}, {3, 3.}, {4, 4.}, {5, 5.}, {6, 6.},
          {7, 7.}, {8, 8.}, {9, 9.}, {10, 10.}}
```

where the `2i + 1` term is necessary to pick out only the $y(x)$ results. This is then illustrated using the command

```
In[8]:= ListPlot[%, PlotJoined->True]
```




```
Out[8]= -Graphics-
```

This algorithm insists on a starting point of zero. For general analyses this would be quite restrictive. However, as our independent variables are time and radius, we will only be concerned with positive values.

Having introduced the package and explained the commands used to obtain the results in this thesis, we are now in a position to present those results. In doing so, we emphasise the power and shortcomings of this software package.

# Chapter 3

# Orbit Equations

## 3.1 Introduction

There are different applications that arise in many practical situations involving differential equations. In this chapter we investigate the utility and scope of Mathematica in solving and illustrating the solution of a special class of differential equations using the functions: `DSolve`, `NDSolve` and `PolarPlot`. The syntax and use of these functions has already been discussed in Chapter 2 (§2.8 in particular). Greater details are provided in the Mathematica book (Wolfram 1996). To obtain more information on solving differential equations using Mathematica the reader is referred to Abel and Braselton (1997), Ganzha and Vorozhtsov (1996), Kythe *et al* (1997) and Shaw and Tigg (1994). The special class of differential equations that we investigate arise in Newtonian theory and in general relativity. These differential equations describe orbits of planets around the sun. Our objective is to solve these orbit equations analytically and thereafter to graphically represent the solutions for the planetary bodies Mercury, Earth and Jupiter. We attempt to plot the relevant orbits in both the Newtonian and relativistic cases and then perform a numerical approximation to find the solutions. Finally, we undertake a perturbative analysis of the equations.

## 3.2 Newtonian Equation

Consider a particle moving in the equatorial plane in the gravitational field of a spherical object of mass $M$ situated at the origin. In this section we assume that Newtonian equations are applicable and neglect relativistic effects. Then the conservation of angular momentum and conservation of energy generate the equation

$$(y')^2 + y^2 = E + \frac{2GM}{h^2}y \tag{3.1}$$

where $y = 1/r$, $E$ is the energy of the orbit, $h$ is the angular momentum per unit mass, $G$ is the universal gravitational constant and the prime denotes differentiation with respect to $x$ - the angular displacement. The angular momentum is represented by

$$h = \dot{x}R^2 = \frac{2\pi}{T}R^2 \tag{3.2}$$

where $R$ is the semi major axis length and $T$ is the period. The energy $E$ is given by

$$E = \frac{(e^2-1)G^2M^2}{h^4} \tag{3.3}$$

where $e$ is the eccentricity of the orbit. In the above we have used the notation and conventions of Foster and Nightingale (1998).

Equation (3.1) is highly nonlinear; however we are in a position to solve it with the help of Mathematica. We first attempt to solve equation (3.1) analytically. The `DSolve` command is used and the syntax is

```
DSolve[y'[x]^2 + y[x]^2 == a + b y[x], y[x], x]
```

where we have set

$$a = E \tag{3.4}$$

$$b = \frac{2GM}{h^2} \tag{3.5}$$

Mathematica returns the following solutions

$$y(x) = \frac{b + b\tan(x - C_1)^2 - \sqrt{4\,a + b^2}\,\tan(x - C_1)\sqrt{1 + \tan(x - C_1)^2}}{2\left(1 + \tan(x - C_1)^2\right)} \tag{3.6}$$

$$y(x) = \frac{b + b\tan(x - C_1)^2 + \sqrt{4\,a + b^2}\,\tan(x - C_1)\sqrt{1 + \tan(x - C_1)^2}}{2\left(1 + \tan(x - C_1)^2\right)} \tag{3.7}$$

where $C_1$ represents a constant of integration. It is important to observe that the two solutions (3.6) and (3.7) arise from the first order equation which is of degree two. In spite of the nonlinearity of equation (3.1), Mathematica is still able to provide closed form solutions. We now try to simplify equations (3.6) and (3.7) using the `PowerExpand` and `Simplify` commands in Mathematica. Equations (3.6) and (3.7) are then reduced to

$$y(x) = \frac{b - \sqrt{4\,a + b^2}\,\sin(x - C_1)}{2} \tag{3.8}$$

$$y(x) = \frac{b + \sqrt{4\,a + b^2}\,\sin(x - C_1)}{2} \tag{3.9}$$

This example illustrates the utility and power of Mathematica; the complicated expressions (3.6) and (3.7) have been reduced to the considerably simpler expressions (3.8) and (3.9).

We observe that differentiating (3.1) yields

$$y'' + y = \frac{GM}{h^2} \tag{3.10}$$

which is now a linear equation in contrast to the nonlinear equation (3.1). Using the command

```
DSolve[y''[x] + y[x] == (G M) / h^2 , y[x], x]
```

we obtain the general solution

$$y(x) = \frac{GM}{h^2} + C_1 \cos(x) - C_2 \sin(x) \tag{3.11}$$

In order to obtain physically meaningful results, we have to solve (3.10) subject to the initial conditions

$$\begin{aligned} y'(0) &= 1 \\ y'(0)^2 - by(0) - a &= 0 \end{aligned}$$

where we have to solve the final expression above to obtain $y(0)$. Unfortunately we cannot proceed with the equation and initial conditions in this form due to the presence of relatively large numbers. As the result we rescale $y$ as follows: Let

$$\bar{y} = \frac{1}{\bar{r}} = \frac{R_e}{r} = R_e y \tag{3.12}$$

where $R_e$ is the radius of the earth. Now (3.10) becomes

$$\bar{y}'' + \bar{y} = \frac{bR_e}{2} \tag{3.13}$$

and the initial conditions are

$$\begin{aligned} \bar{y}'(0) &= 0 \\ \bar{y}'(0)^2 - bR_e\bar{y}(0) - aR_e^2 &= 0 \end{aligned}$$

where we have to solve the final expression above to obtain $\bar{y}(0)$. We will always choose the lower value of $\bar{y}(0)$ to conform with our choice of the origin. We solve the above equation using the command

```
DSolve[{yb''[x] + yb[x] == (G m re)/h2, yb[0] == 1/2(b re -
        Sqrt[4a + b^2] re), yb'[0] == 0}, yb[x], x]
```

where `yb` will always represent $\bar{y}$, `h2` is the angular momentum $h^2$, `m` denotes the mass of the sun $M$ and `re` refers to the radius of the earth $R_e$. The solution obtained is

$$\bar{y}(x) = 1/2(bR_e - \sqrt{4a + b^2}R_e \cos(x)) \tag{3.14}$$

We now analyse the motions of specific planetary bodies. In particular, we consider the values for the planets Mercury, Earth and Jupiter. In order to obtain specific plots for these planets the following table of values were utilised

| | $e$ | $T(years)$ | $R(au)$ |
|---|---|---|---|
| Mercury | 0.206 | 0.2402 | 0.387 |
| Earth | 0.0167 | 1 | 1 |
| Jupiter | 0.0489 | 11.865 | 5.202 |

TABLE 3.1: Physical parameters for some planetary bodies

where $1au = 1.4953 \times 10^{11}m$, $1year = 365.2564 \times 86400s$ (Moulton 1970). Using the above values, we have

$$\bar{y}(x) = 2.58903 - 0.533336 \cos(x) \tag{3.15}$$

for Mercury,

$$\bar{y}(x) = 1.00655 - 0.0168094 \cos(x) \tag{3.16}$$

for Earth, and

$$\bar{y}(x) = 0.193503 - 0.00946228 \cos(x) \tag{3.17}$$

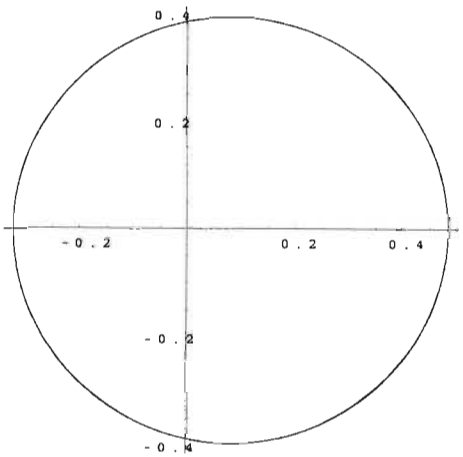
for Jupiter.

To obtain a more meaningful, visual interpretation of the motions of these planetary bodies, we generate graphical representations of (3.15), (3.16) and (3.17). Recall

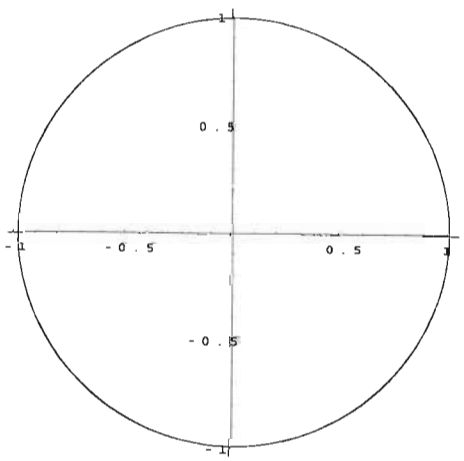
that the variables $y$ and $x$ in (3.1) relate to the radius and angular displacement respectively, for planetary bodies. As a result, we utilise `PolarPlot` to obtain an accurate reflection of the motion of these objects. In the case of Mercury we use the command

```
PolarPlot[1/yb[x], {x, -5Pi, 5Pi}]
```

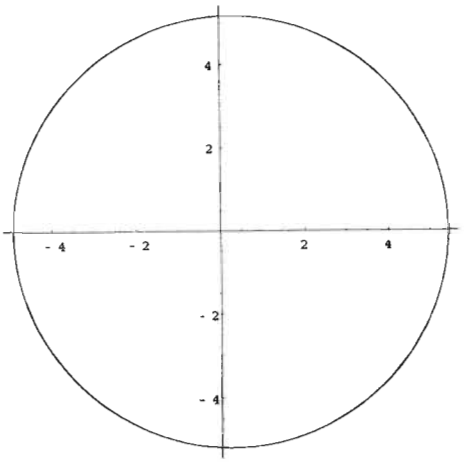
which produces the output



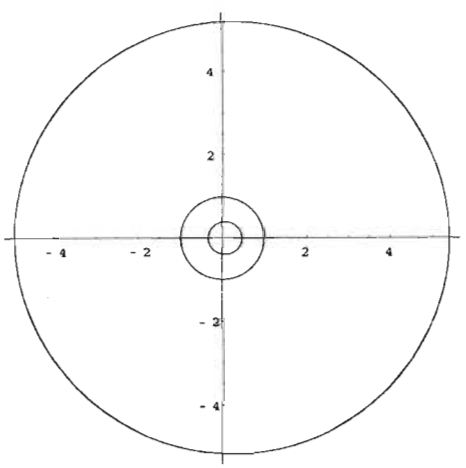

In a similar manner the following graph was obtained for the planet Earth



and the graph for Jupiter is given by




In order to compare the three graphs we need to plot them on the same system of axes via the `Show` command. The following representation is generated




where the graphs from inner to outer represent Mercury, Earth and Jupiter respectively. From the above illustration it would appear that the planetary orbits are circular, however, this is not the case because the eccentricity $e$ for the planets considered (see Table 3.1) is not zero which is required for a circle.

## 3.3 Relativistic Equation

We once again consider a particle moving in the equatorial plane under the gravitational attraction of a spherical object of mass $M$ as in §3.2. However, in this case we incorporate relativistic effects. The Einstein field equations, which relate the matter content to the spacetime curvature in a nonlinear manner, then hold. For detailed treatments of general relativity, the Einstein field equations and background differential geometry, the reader is referred to the classical treatments of de Felice and Clark (1990), Hawking and Ellis (1973), Misner *et al* (1973) and Schutz (1985), amongst others.

The relativistic orbit equation corresponding to (3.1) is given by

$$(y')^2 + y^2 = E + \frac{2GM}{h^2}y + \frac{2GM}{c^2}y^3 \tag{3.18}$$

where $c$ refers to the speed of light. Equation (3.18) is more complex than equation (3.1) which was considered in the previous section. The difficulty in solving (3.18) is related to the addition of the relativistic term

$$\frac{2GM}{c^2}y^3 \tag{3.19}$$

which is highly nonlinear. The solution of (3.18) cannot be expressed solely in terms of elementary functions; elliptic integrals are necessary in general (Foster and Nightingale 1998). For experimental results justifying the inclusion of the relativistic term (3.19) the reader should consult Dicke and Goldstein (1967), Duncombe (1956), Nordtvedt and Will (1972), Shapiro (1972) and Shapiro *et al* (1973). These treatments provide detailed arguments of the physical importance of the relativistic equation (3.18) for planetary orbits.

Differentiating (3.18) we obtain

$$y'' + y = \frac{GM}{h^2} + \frac{3GM}{c^2}y^2 \tag{3.20}$$

We apply the transformation (3.12) to equation (3.20) to produce

$$\bar{y}'' + \bar{y} = \frac{bR_e}{2} + \frac{3GM}{c^2 R_e}\bar{y}^2 \tag{3.21}$$

Attempting to solve (3.21) using the `DSolve` command proves to be too complicated for Mathematica. The following error message is obtained

`Solve::dinv : The expression EllipticF[ArcSin[ `$\sqrt{\frac{\text{<<1>>}}{\text{<<1>>}}}$`], `$\frac{\text{<<1>>}}{\text{<<1>>}}$` ] involves unknowns in more than one argument, so inverse functions cannot be used.`

This is not surprising as no solution in terms of elementary functions exist.

## 3.4 Numerical Approach

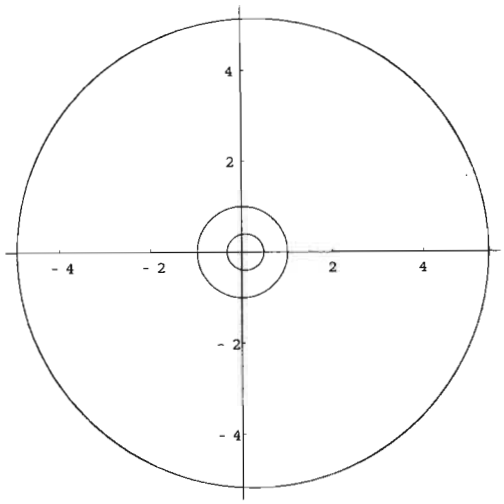
In this section we will attempt to solve our equation numerically using `NDSolve` and compare our results with the previous section. We start with (3.13) because of its simplicity and the presence of an analytic solution for verification of the numerical results. The appropriate Mathematica command to solve (3.13) numerically is

```
NDSolve[{yb''[x] + yb[x] == (b re)/2, yb'[0] == 0, yb[0] == 2.05569},
        yb[x], {x, -5 Pi, 5 Pi}]
```

where we have used the values of $b$ corresponding to Mercury and `re` refers to the radius of the earth. This produced the following solution

$$\bar{y}(x) = \text{InterpolatingFunction}(-15.708, 15.708)$$

In a similar way solutions can be obtained for Earth and Jupiter. Plotting all the solutions on the same system of axis produces the following output




These results are consistent with the analytical approach.

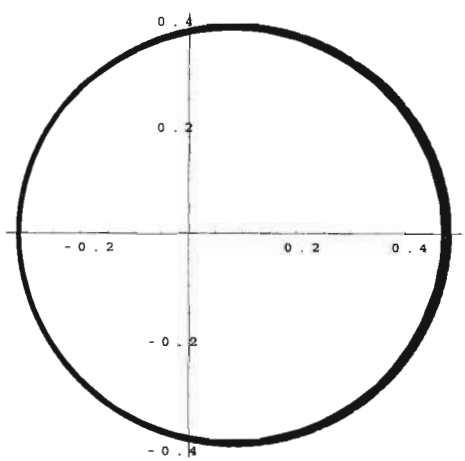
We now attempt to solve (3.21) numerically. The command used in Mathematica for the planet Mercury is

```
NDSolve[{yb''[x] + yb[x] == ((G m re)/h^2) + (3 G m /(c^2 re)) yb[x]^2,
        yb'[0] == 0, yb[0] == 2.05569} , yb[x], {x, -1000, 1000}]
```

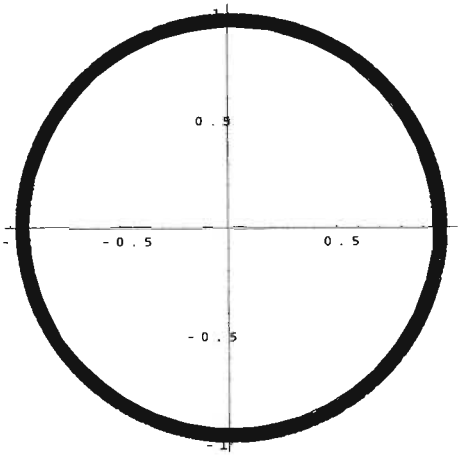
to produce the following solution

$$\bar{y}(x) = \text{InterpolatingFunction}(-182.377, 182.377) \tag{3.22}$$

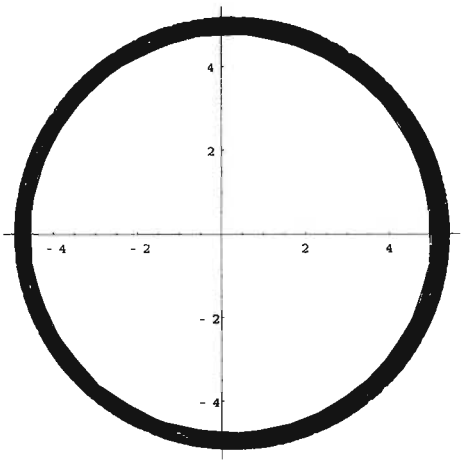
Graphically representing the above solution produces the following output



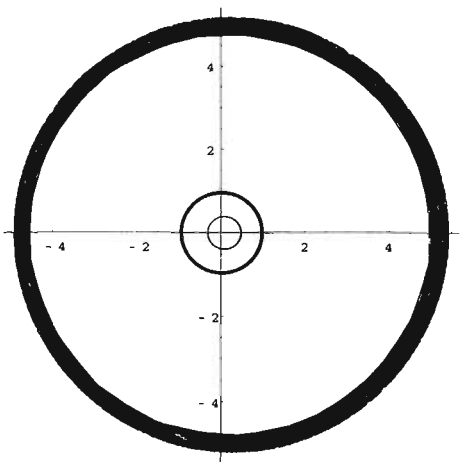
Similarly, Earth has the graph



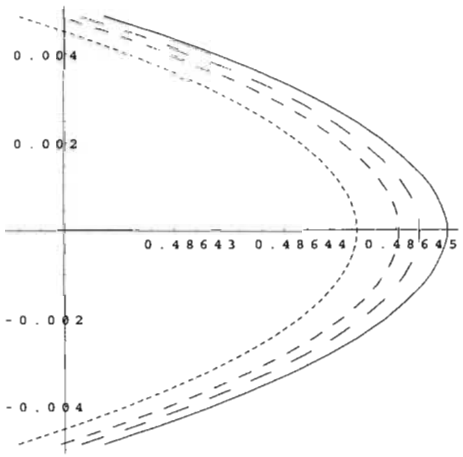

and Jupiter



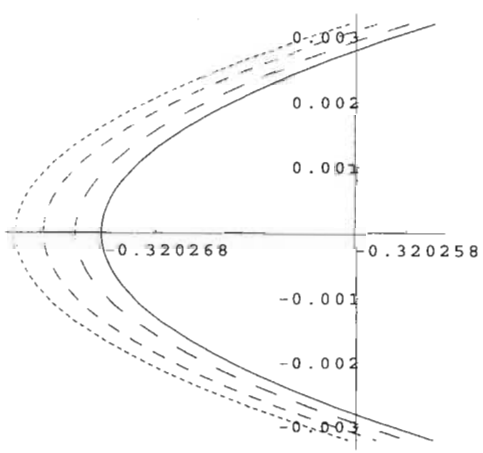

In order to compare all graphs on the same system of axis we have



We observe from the above graphs a distinct thickening in the plotting of the orbit. To determine the behaviour of the graph we now investigate the orbit of Mercury by looking at a small portion of the graph. The intervals chosen for each graph were $[-0.01, 0.01]$, $[-0.01+4\pi, 0.01+4\pi]$, $[-0.01+8\pi, 0.01+8\pi]$ and $[-0.01+12\pi, 0.01+12\pi]$ to produce the following




To determine the behaviour of the graph on the opposite end of the axis, we use the following intervals: $[-0.01, 0.01]$, $[-0.01+3\pi, 0.01+3\pi]$, $[-0.01+7\pi, 0.01+7\pi]$ and $[-0.01+11\pi, 0.01+11\pi]$ to obtain




It is evident that whilst the graphs for the relativistic case appears to be a solid line, the thickening is due to a precession of the planetary orbits. The small values obtained on the $x$ axis above explain why the phenomenon was not noticed on the previous plots.

## 3.5 Perturbative Approach

We have observed that `DSolve` could not handle the relativistic equation. We have obtained some results using `NDSolve` but wish to verify them using an alternative technique. The relativistic equation has the form

$$y'' + y = \frac{GM}{h^2} + \frac{3GM}{c^2}y^2 \tag{3.23}$$

We will use a perturbative approach in this section and will represent the complete Mathematica session displaying both the input and the output statements. Again due to the presence of relatively large numbers we rescale (3.23) using the transformation given in (3.12) to produce

$$\bar{y}'' + \bar{y} = \frac{GMR_e}{h^2} + \frac{3GM}{c^2R_e}\bar{y}^2 \tag{3.24}$$

We start the Mathematica session with (3.24) as follows

```
In[1]:= eq = yb''[x] + yb[x] - (G M re/h^2) - (3 G M/(c^2 re)) yb[x]^2
```

$$\texttt{Out[1]=} \; -\frac{GMre}{h^2} + yb[x] - \frac{3GMyb[x]^2}{c^2re} + yb''[x]$$

where `eq` is set to zero and `yb` refers to $\bar{y}$ in (3.24). Defining

$$A = \frac{GM}{h^2} \tag{3.25}$$

$$\epsilon = \frac{3GMA}{c^2} = \frac{3G^2M^2}{c^2h^2} \tag{3.26}$$

requires the following Mathematica statements

```
In[2]:= eq = eq /. (G M/h^2) -> A
```

$$\texttt{Out[2]=} \; -Are + yb[x] - \frac{3GMyb[x]^2}{c^2re} + yb''[x]$$

```
In[3]:= eq = eq /. (-3 G M)/c^2 -> -ep /A
```

$$\texttt{Out[3]=} \; -Are + yb[x] - \frac{epyb[x]^2}{Are} + yb''[x]$$

where `ep` refers to $\epsilon$. Having made the substitutions above, (3.24) now becomes

$$\bar{y}'' + \bar{y} = AR_e + \frac{\epsilon}{AR_e}\bar{y}^2 \tag{3.27}$$

We note that $\epsilon$ is a small dimensionless quantity. We assume a solution of the form

$$\bar{y}(x) = \bar{y}_0 + \epsilon w(x) + O(\epsilon^2) \tag{3.28}$$

In Mathematica this solution is written as

```
In[4]:= yb[x] = y0[x] + ep w[x] + O[ep^2]
```

$$\text{Out[4]= } (epw[x] + y0[x]) + O[ep^2]^1$$

where y0 represents $\bar{y}_0$ in (3.28). We now substitute (3.28) into (3.27) in the following way

```
In[5]:= eq = eq  /. yb''[x] ->  D[yb[x], {x, 2}]
```

$$\text{Out[5]= } \left(\left(-Are + epw[x] + y0[x] - \frac{ep(epw[x]+y0[x])^2}{Are} + epw''[x] + y0''[x]\right) + O[ep^2]^1\right) + O[ep^2]^1$$

The coefficients of $\epsilon$ from the above output are obtained using the command

```
In[6]:= CoefficientList[eq, ep]
```

$$\text{Out[6]= } \left\{-Are + y0[x] + y0''[x], w[x] - \frac{y0[x]^2}{Are} + w''[x], -\frac{2w[x]y0[x]}{Are}, -\frac{w[x]^2}{Are}\right\}$$

We obtain the zeroth order terms in $\epsilon$ as follows

```
In[7]:= %6[[1]]
```

$$\text{Out[7]= } -Are + y0[x] + y0''[x]$$

The equation for the zeroth order terms in $\epsilon$ translate to

$$\bar{y}_0'' + \bar{y}_0 = AR_e \tag{3.29}$$

We note that (3.29) is equivalent to (3.10) with solution

$$\bar{y}_0(x) = AR_e + C_1\cos(x) - C_2\sin(x) \tag{3.30}$$

We rewrite this solution as

$$\bar{y}_0(x) = AR_e + B\cos(x) \tag{3.31}$$

where we have reoriented the axis to make $C_2$ zero and $B$ refers to the constant $C_1$. We now obtain the first order term in $\epsilon$ using the command

```
In[8]:= %6[[2]]
```
Out[8]= $w[x] - \frac{y0[x]^2}{Are} + w''[x]$

The equation for the first order terms in $\epsilon$ is given as

$$w'' + w = \frac{\bar{y}_0^2}{AR_e} \tag{3.32}$$

We now replace $\bar{y}_0$ in (3.32) with (3.31) and expand the expression as follows

```
In[9]:= -%[[2]] /. y0[x] -> (A re) + B Cos[x] // Expand
```
Out[9]= $Are + 2BCos[x] + \frac{B^2Cos[x]^2}{Are}$

Rewriting and expanding the above expression in terms of $\cos(2x)$ requires the following substitution

```
In[10]:= % /. Cos[x]^2 -> (1/2 + 1/2 Cos[2x]) // Expand
```
Out[10]= $\frac{B^2}{2Are} + Are + 2BCos[x] + \frac{B^2Cos[2x]}{2Are}$

Equation (3.32) now becomes

$$w'' + w = \frac{B^2}{2Are} + Are + 2B\cos[x] + \frac{B^2\cos[2x]}{2Are} \tag{3.33}$$

Since (3.33) is a linear equation in $w''$ we can write

$$w = w_a + w_b + w_c \tag{3.34}$$

where $w_a$, $w_b$ and $w_c$ are solutions of the equations

$$w_a'' + w_a = \frac{B^2}{2Are} + Are \tag{3.35}$$

$$w_b'' + w_b = 2B\cos[x] \tag{3.36}$$

$$w_c'' + w_c = \frac{B^2\cos[2x]}{2Are} \tag{3.37}$$

and we used the principle of superposition of solutions of linear equations. The

inhomogeneous solutions to the system are now calculated. We solve and simplify (3.35) in Mathematica as follows

```
In[11]:= Simplify[DSolve[w''[x] + w[x] == %[[1]] + %[[2]], w[x], x]]
```

$$\text{Out[11]= } \left\{\left\{w[x] \to \frac{B^2}{2Are} + Are + C[2]Cos[x] - C[1]Sin[x]\right\}\right\}$$

We are only interested in the inhomogeneous solutions and therefore set `C[1]` and `C[2]` to zero in the solutions $w_a$, $w_b$ and $w_c$. The solution for $w_a$ is calculated in Mathematica as follows

```
In[12]:= wa = (First[w[x] /. wa]) /. {C[1] -> 0, C[2] -> 0}
```

$$\text{Out[12]= } \frac{B^2}{2Are} + Are$$

From the above output we now have

$$w_a = \frac{B^2}{2AR_e} + AR_e \tag{3.38}$$

Equation (3.36) is solved in a similar way using the commands

```
In[13]:= Simplify[DSolve[w''[x] + w[x] == %10[[3]], w[x], x]]
```

$$\text{Out[13]= } \{\{w[x] \to (B + C[2])Cos[x] + (Bx - C[1])Sin[x]\}\}$$

```
In[14]:= wb = (First[w[x] /. wb]) /. {C[1] -> 0, C[2] -> -B}
```

$$\text{Out[14]= } BxSin[x]$$

which is given by

$$w_b = Bx\sin(x) \tag{3.39}$$

Finally solving (3.37) we have

```
In[15]:= Simplify[DSolve[w''[x] + w[x] == %10[[4]], w[x], x]]
```

$$\text{Out[15]= } \left\{\left\{w[x] \to C[2]Cos[x] - \frac{B^2Cos[2x]}{6Are} - C[1]Sin[x]\right\}\right\}$$

```
In[16]:= wc = (First[w[x] /. wc]) /. {C[1] -> 0, C[2] -> 0}
```

$$\text{Out[16]= } -\frac{B^2Cos[2x]}{6Are}$$

to obtain the solution

$$w_c = -\frac{B^2 \cos(2x)}{6AR_e} \tag{3.40}$$

To obtain and simplify $w$ in Mathematica we perform the following

```
In[17]:= w = Simplify[wa + wb + wc]
Out[17]= (3B^2+6A^2re^2-B^2Cos[2x]+6ABrexSin[x])/(6Are)
```

Our final solution for $w$ now becomes

$$w = \frac{3B^2 + 6A^2R_e^2 - B^2\cos(2x) + 6ABR_e x\sin(x)}{6AR_e} \tag{3.41}$$

Assigning values to the constants for the planet Mercury requires the following statements

```
In[18]:= B = 0.533336
Out[18]= 0.533336

In[19]:= R = 0.387*1.4953*10^11
Out[19]= 5.78681 × 10^10

In[20]:= re = 1 * 1.4953*10^11
Out[20]= 1.4953 × 10^11

In[21]:= T = 0.2402*365.2564*86400
Out[21]= 7.58027 × 10^6

In[22]:= G = 6.67*10^-11
Out[22]= 6.67 × 10^-11

In[23]:= M = 2*10^30
Out[23]= 2000000000000000000000000000000

In[24]:= h = ((2*Pi) / T) * R^2
Out[24]= 2.77571 × 10^15
```

```
In[25]:= A = (G M) / h^2
```

$$\text{Out[25]= } 1.73144 \times 10^{-11}$$

From (3.28) we replace $\bar{y}_0$ with (3.31) using the command

```
In[26]:= yb = (A re) + B Cos[x] + (ep w)
```

$$\text{Out[26]= } 2.58903 + 0.533336Cos[x] + 0.0643743ep(41.0717 - 0.284447Cos[2x] + 8.28492xSin[x])$$

We now set $\epsilon = 0.02$ and simplify the above expression in the following manner

```
In[27]:= yb = Simplify[yb /. ep -> 0.02]
```

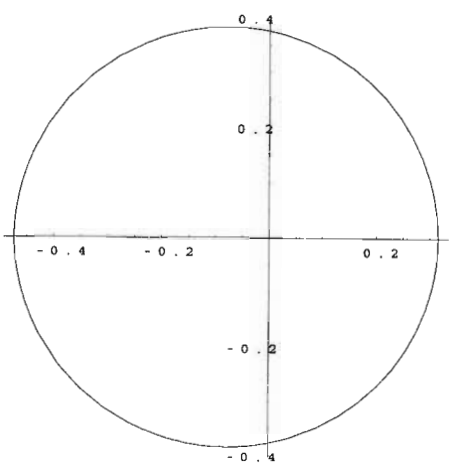
$$\text{Out[27]= } 2.64191 + 0.533336Cos[x] - 0.000366222Cos[2x] + 0.0106667xSin[x]$$

Thus we now have

$$\bar{y} = 2.64191 + 0.533336\cos(x) - 0.000366222\cos(2x) + 0.0106667x\sin(x) \quad (3.42)$$

We plot the graph of $1/\bar{y}$ over the range $[-\pi, \pi]$ to obtain a graphical representation using the commands

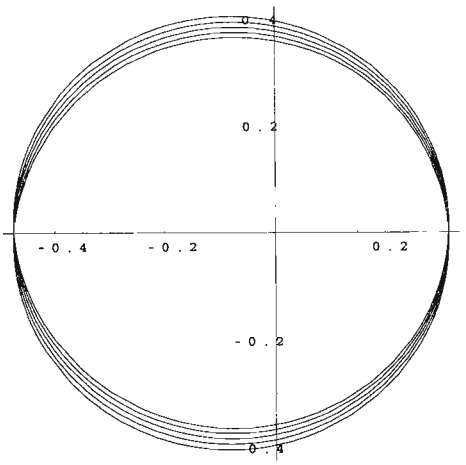
```
In[28]:= << Graphics`Graphics`
In[29]:= PolarPlot[1/yb, {x, -Pi, Pi}, AspectRatio -> 1]
```




```
Out[29]= -Graphics-
```

Furthermore, plotting the graph of $1/\bar{y}$ over the range $[-5\pi, 5\pi]$

```
In[30]:= PolarPlot[1/yb, {x, -5Pi, 5Pi}, AspectRatio -> 1]
```




```
Out[30]= -Graphics-
```

Using the perturbative approach we notice the distinct precession if $\epsilon = 0.02$ in the graph over the range $[-5\pi, 5\pi]$. We were not able to notice this effect in the previous section for the same range. This is unsurprisingly as $\epsilon \ll 1$ once the physical values of the various constants are taken into account.

The perihelion of a planet occurs when $r$ is a minimum ($y$ is a maximum). If we write (3.42) as

$$\bar{y}(x) = A + B\cos(x - \epsilon x) + O(\epsilon) \tag{3.43}$$

We see that $\bar{y}$ in (3.42) is a maximum when

$$x(1 - \epsilon) = 2\pi n \tag{3.44}$$

or approximately

$$x = 2\pi n(1 + \epsilon) \tag{3.45}$$

Therefore successive perihelia will occur at intervals of

$$\Delta x = 2\pi(1 + \epsilon) \tag{3.46}$$

instead of $2\pi$ as in periodic motion. Thus the perihelion shift per revolution is given

by

$$\Delta x = 2\pi\epsilon = 2\pi\frac{3G^2M^2}{c^2h^2} \tag{3.47}$$

In the case of Mercury, (3.47) gives a total shift of $4.83752 \times 10^{-7}$ and a shift of $1.8807 \times 10^{-7}$ for Earth whilst Jupiter has a shift of $3.61555 \times 10^{-8}$.

# Chapter 4

# Emden-Fowler Equation

## 4.1 Introduction

There are many special techniques and approaches that can be used to solve differential equations. One such procedure is to utilise the symmetries associated with the equation that is being considered. This procedure was studied and perfected by Lie (1891) and is based on the invariance of a differential equation under a point transformation. Many differential equations that arise in fields such as classical mechanics, hydrodynamics and relativity can be investigated using this powerful technique of solving differential equations. In particular, the Emden-Fowler equation is of fundamental importance when analysing the gravitational behaviour in many spherically symmetric cosmological models in general relativity. In this chapter we analyse this second order equation which arises in certain inhomogeneous cosmological applications. The Emden-Fowler equation is first written in autonomous form using the Lie analysis. We consider different cases of this autonomous equation and plot graphs for each case using Mathematica. For a particular model chosen we generate profiles of the pressure gradient and density; graphical representations of the density and the pressure are produced. This demonstrates that a numerical analysis of the behaviour of the pressure and density is possible and complements previous analytical studies.

## 4.2 Lie Analysis

Spherically symmetric gravitational fields are important in cosmology and these fields have been studied by Stephani (1968), Sussman (1987, 1988a, 1988b, 1989a, 1989b), Szafron (1977) and Szekeres (1966), amongst others. We investigate the field equations of a symmetric shear-free fluid that arise in inhomogeneous cosmological models. For this situation we can introduce a comoving and isotropic coordinate system $x^i = (t, r, \theta, \phi)$ such that the metric can be written as

$$ds^2 = -e^{2\nu(t,r)}dt^2 + e^{2\lambda(t,r)}[dr^2 + r^2(d\theta^2 + \sin^2\theta d\phi^2)] \tag{4.1}$$

where the functions $\nu(t,r)$ and $\lambda(t,r)$ represent the gravitational potentials. The gravitational potentials are restricted by the Einstein field equations. For the metric (4.1), the Einstein field equations reduce to

$$e^{\nu} = \lambda_t e^{-f(t)} \tag{4.2}$$

$$e^{\lambda}(\lambda_{rr} - \lambda_r^2 - \lambda_r/r) = -\tilde{F}(r) \tag{4.3}$$

where $f(t)$ and $\tilde{F}(r)$ are arbitrary functions of integration. The energy density $\mu$ and pressure $p$ then assume the forms

$$\mu = 3e^{2f} - e^{-2\lambda}(2\lambda_{rr} + \lambda_r^2 + 4\lambda/r) \tag{4.4}$$

$$p = \lambda_t^{-1}e^{-3\lambda}\frac{\partial}{\partial t}\left[e^{\lambda}(\lambda_r^2 + 2\lambda_r/r) - e^{3\lambda+2f}\right] \tag{4.5}$$

To find an exact solution of the field equations (4.2) and (4.3) the functions $f(t)$ and $\tilde{F}(r)$ need to be specified and equation (4.3) has to solved for $\lambda$. The quantity $\nu$ is given by (4.2). Thereafter the dynamical quantities $\mu$ and $p$ can be computed from (4.4) and (4.5). Thus the equation (4.3) is the master equation: a solution of this equation generates a solution to the system (4.2)-(4.5). The transformation

$$x = r^2 \tag{4.6}$$

$$L(t,x) = e^{-\lambda} \tag{4.7}$$

$$\tilde{F}(x) = F/4r^2 \tag{4.8}$$

reduces equation (4.3) to

$$L_{xx} = F(x)L^2 \tag{4.9}$$

which is the fundamental equation governing the behaviour of the model. This equation is known as the Emden-Fowler equation of index two. This equation has been studied extensively by researchers in cosmology. Recent investigations include the treatments of Hallburd (1999), Herlt (1996), Maharaj *et al* (1996), Srivastava (1987, 1992) and Stephani and Wolf (1996).

Now we consider a widely applied technique to reduce equation (4.9) to a first order differential equation. A systematic method to determine whether a second order differential equation can be solved by quadratures is that of Lie. For more information on the Lie analysis of differential equations and applications to mathematical physics the reader is referred to Bluman and Kumei (1989), Govender (1997), Leach (1981), Mahomed and Leach (1990, 1991), Moodley (1998). On performing the Lie analysis we find that (4.9) is invariant under the action of symmetry

$$G = A(x)\frac{\partial}{\partial x} + [B(x)L + C(x)]\frac{\partial}{\partial L} \tag{4.10}$$

provided that

$$F(x) = MA^{-5/2}\exp\left(\frac{\alpha}{2}\int\frac{dx}{A}\right) \tag{4.11}$$

$$A_{xxx} = 2MCA^{-5/2}\exp\left(\frac{\alpha}{2}\int\frac{dx}{A}\right) \tag{4.12}$$

$$B(x) = (A_x - \alpha)/2 \tag{4.13}$$

$$C(x) = C_0 + C_1 x \tag{4.14}$$

where $\alpha$, $M$, $C_0$ and $C_1$ are constants.

A solution of equation (4.12) gives $F(x)$ by equation (4.11). Then equation (4.9) can be reduced to a first order differential equation. Note that the existence of the symmetry (4.10) permits this reduction to first order. If two symmetries are known, the second order equation (4.9) may be reduced to quadratures. The Lie analysis was first used by Kustaanheimo and Qvist (1948) in this context for the analysis of expanding inhomogeneous models; they considered the case $\alpha = 0$ and $C = 0$. The case that we are interested in corresponds to $\alpha \neq 0$ and $C = 0$. This appears to give a generalisation of Kustaanheimo and Qvist since equations (4.11) and (4.12) imply

$$A = ax^2 + bx + c \tag{4.15}$$

$$F = MA^{-5/2} \exp\left(\int \frac{\alpha}{2A} dx\right) \tag{4.16}$$

We now let

$$\mathcal{Y} = LA^{-1/2} \exp\left(\int \frac{\alpha}{2A} dx\right) \tag{4.17}$$

$$\mathcal{X} = \int \frac{dx}{A} \tag{4.18}$$

Then equation (4.9) can be written in the autonomous form

$$\mathcal{Y}'' - \alpha \mathcal{Y}' + \left(\frac{\alpha^2}{4} + ac - \frac{1}{4}b^2\right)\mathcal{Y} + M\mathcal{Y}^2 = 0 \tag{4.19}$$

where $\mathcal{Y}$ is a function of $\mathcal{X}$. In equation (4.19), we set $M = 1$ and $\Delta$ represents $b^2 - 4ac$ giving

$$\mathcal{Y}'' - \alpha\mathcal{Y}' + \left(\frac{\alpha^2}{4} - \frac{1}{4}\Delta\right)\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.20}$$

Maharaj *et al* (1996) have shown that (4.20) can be reduced to quadratures for special values of $\alpha$ and $\Delta$. They also commented on the general integrability of this equation for other values of $\alpha$ and $\Delta$. Given the importance of $\alpha$ and $\Delta$ in the behaviour of the solution of (4.20), we consider numerical solutions to (4.20) for all possible classes of values of $\alpha$ and $\Delta$. As a consequence, the following cases are considered

$$\begin{aligned}
\alpha = 0 \quad &\text{and} \quad \Delta = 0 \\
\alpha = 0 \quad &\text{and} \quad \Delta > 0 \\
\alpha = 0 \quad &\text{and} \quad \Delta < 0 \\
\alpha > 0 \quad &\text{and} \quad \Delta = 0 \\
\alpha > 0 (\alpha <> 5\sqrt{\Delta}) \quad &\text{and} \quad \Delta > 0 \\
\alpha > 0 \quad &\text{and} \quad \Delta < 0 \\
\alpha = 5\sqrt{\Delta} \quad &\text{and} \quad \Delta > 0 \\
\alpha = -5\sqrt{\Delta} \quad &\text{and} \quad \Delta > 0 \\
\alpha < 0 \quad &\text{and} \quad \Delta = 0 \\
\alpha < 0 (\alpha <> -5\sqrt{\Delta}) \quad &\text{and} \quad \Delta > 0 \\
\alpha < 0 \quad &\text{and} \quad \Delta < 0
\end{aligned}$$

For each of the above cases, Mathematica was used to numerically solve the respective equation using the `NDSolve` function. Thereafter, to interpret the results obtained, a graphical representation is illustrated using the `Plot` function. All the solutions obtained via the `NDSolve` function were verified using the `RKSolve` function which was discussed in Chapter 2.

### 4.2.1 $\alpha = 0$ and $\Delta = 0$

Under these conditions equation (4.20) reduces to

$$\mathcal{Y}'' + \mathcal{Y}^2 = 0 \tag{4.21}$$

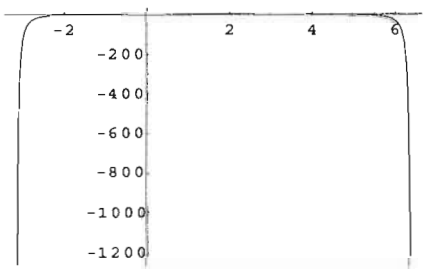
where $a = 1$, $b = 2$ and $c = 1$, thereby making $\Delta = 0$. The Mathematica command

```
NDSolve[{y''[x] + y[x]^2 == 0, y'[0] == 0, y[0] == 1}, y[x], {x, -10, 10}]
```

yields

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-3.21017, 6.42038) \tag{4.22}$$

as the solution. The graphical representation of this solution is given by




Note that the initial conditions $\mathcal{Y}'(0) = 0$ and $\mathcal{Y}(0) = 1$ will be used for all the succeeding cases.

### 4.2.2 $\alpha = 0$ and $\Delta > 0$

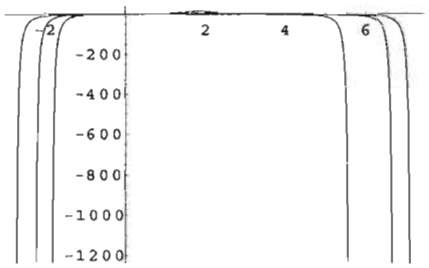
Using the above conditions equation (4.20) takes the form

$$\mathcal{Y}'' - \frac{17}{4}\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.23}$$

where $a = 1$, $b = 3$ and $c = 1$, hence making $\Delta = 5$. The numerical solution returned by Mathematica is

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-2.28918, 6.72194) \tag{4.24}$$

We also consider the case when $a = 2$, $b = 5$ and $c = 1$ making $\Delta = 17$ and the case $a = 5$, $b = 10$ and $c = 3$ making $\Delta = 40$ to determine the behaviour of the solution for other positive values of $\Delta$. All three results are graphically represented below.




### 4.2.3 $\alpha = 0$ and $\Delta < 0$

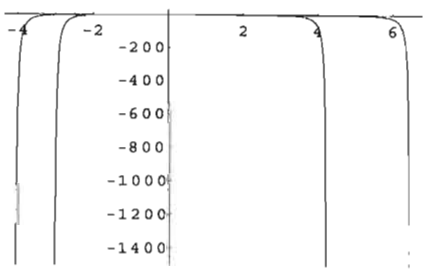
Equation (4.20) now becomes

$$\mathcal{Y}'' + \frac{3}{4}\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.25}$$

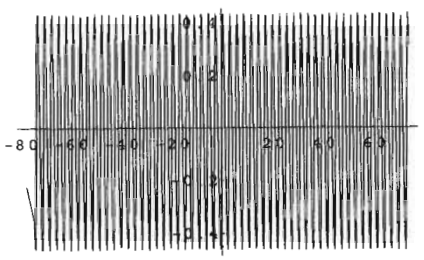
where $a = 1$, $b = 1$ and $c = 1$, thereby making $\Delta = -3$. The solution is

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-3.75471, 6.33594) \tag{4.26}$$

If $a = 2$, $b = 2$ and $c = 1$, then $\Delta = -4$. Illustrating the two graphs produce



Furthermore, using the values $a = 3$, $b = 2$ and $c = 2$, makes $\Delta = -20$. This produces the graph



Interestingly, this oscillation is observed for $\Delta \leq -6$.

### 4.2.4 $\alpha > 0$ and $\Delta = 0$

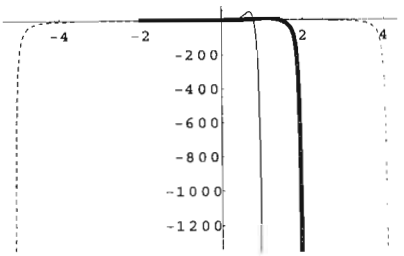
For this case equation (4.20) takes the form

$$\mathcal{Y}'' - \mathcal{Y}' - \frac{1}{4}\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.27}$$

where $\alpha = 1$, $a = 1$, $b = 2$ and $c = 1$, thereby making $\Delta = 0$. The solution returned is

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-5.16155, 4.12919) \tag{4.28}$$

To determine the behaviour of the solution for other positive values of $\alpha$ we consider the case when $\alpha = 5$ and $\alpha = 15$. All three results are graphically represented below.




The dotted line represents the case when $\alpha = 1$ and the thick line is given by $\alpha = 5$.

### 4.2.5 $\alpha > 0$ $(\alpha <> 5\sqrt{\Delta})$ and $\Delta > 0$

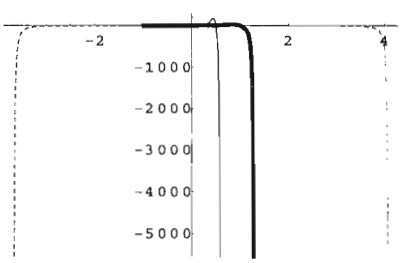
Using the above conditions equation (4.20) takes the form

$$\mathcal{Y}'' - \mathcal{Y}' - \mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.29}$$

where $\alpha = 1$, $a = 1$, $b = 3$ and $c = 1$, making $\Delta = 5$. The solution becomes

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-3.70133, 4.10063) \tag{4.30}$$

If we consider the case $\alpha = 10$, $a = 3$, $b = 7$ and $c = 4$ then $\Delta = 1$, and the case $\alpha = 30$, $a = 7$, $b = 10$ and $c = 3$ then $\Delta = 16$. This produces the graph



The dotted line represents the case when $\alpha = 1$ and the thick line is given by $\alpha = 10$.

### 4.2.6 $\alpha > 0$ and $\Delta < 0$

Equation (4.20) becomes

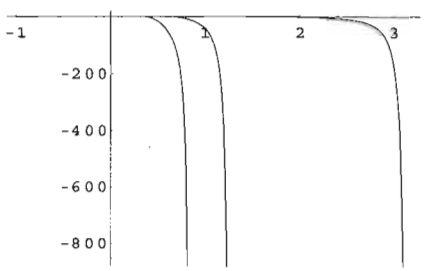
$$\mathcal{Y}'' - 2\mathcal{Y}' + \frac{7}{4}\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.31}$$

where $\alpha = 2$, $a = 1$, $b = 1$ and $c = 1$, hence making $\Delta = -3$. This produces the solution

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-10, 3.1692) \tag{4.32}$$

We also consider the case when $\alpha = 10$, $a = 6$, $b = 3$ and $c = 4$ making $\Delta = -87$ and the case $\alpha = 20$, $a = 9$, $b = 10$ and $c = 11$ making $\Delta = -296$.

The graphs are illustrated below.




### 4.2.7 $\alpha = 5\sqrt{\Delta}$ and $\Delta > 0$

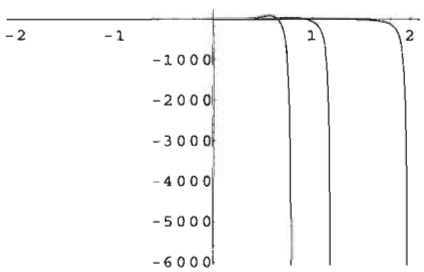
Under these conditions equation (4.20) reduces to

$$\mathcal{Y}'' - 5\mathcal{Y}' + 6\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.33}$$

where $a = 3$, $b = 5$ and $c = 2$, making $\Delta = 1$. This yields the solution

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-10, 2.00423) \tag{4.34}$$

We also consider the case when $a = 1$, $b = 3$ and $c = 1$ hence making $\Delta = 5$ and the case $a = 7$, $b = 10$ and $c = 3$ making $\Delta = 16$. A graphical illustration is given below to determine the behaviour of the above cases.




### 4.2.8 $\alpha = -5\sqrt{\Delta}$ and $\Delta > 0$

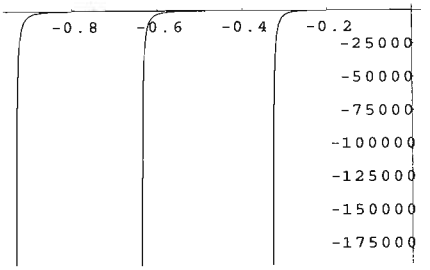
Under these conditions equation (4.20) becomes

$$\mathcal{Y}'' + 5\sqrt{5}\mathcal{Y}' + 30\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.35}$$

where $a = 1$, $b = 3$ and $c = 1$, thereby making $\Delta = 5$. This produces the solution

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-0.939805, 10) \tag{4.36}$$

We also consider the case when $a = 2$, $b = 5$ and $c = 1$ making $\Delta = 17$ and the case $a = 7$, $b = 15$ and $c = 4$ making $\Delta = 113$. To determine the behaviour of the solution for the above cases we present a graphical representation below.




### 4.2.9 $\alpha < 0$ and $\Delta = 0$

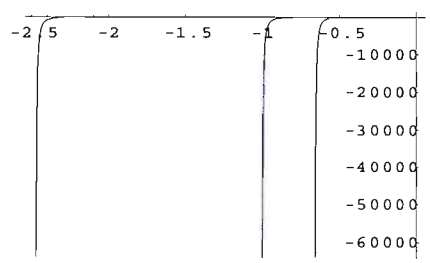
For this case (4.20) takes the form

$$\mathcal{Y}'' + \mathcal{Y}' + \frac{1}{4}\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.37}$$

where $\alpha = -1$, $a = 1$, $b = 2$ and $c = 1$. This yields the solution

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-2.48181, 10) \tag{4.38}$$

We also consider the case when $\alpha = -10$, $a = 2$, $b = 4$ and $c = 2$ and the case $\alpha = -20$, $a = 2$, $b = 4$ and $c = 2$. The graphical representation is depicted below.



### 4.2.10 $\alpha < 0$ $(\alpha <> -5\sqrt{\Delta})$ and $\Delta > 0$

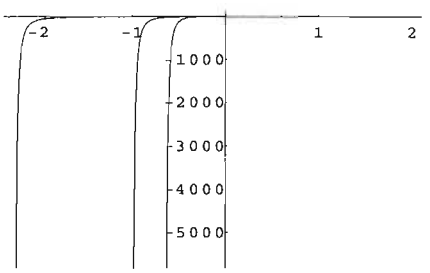
For this case (4.20) takes the form

$$\mathcal{Y}'' + \mathcal{Y}' - \mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.39}$$

where $\alpha = -1$, $a = 1$, $b = 3$ and $c = 1$, thereby making $\Delta = 5$. The solution becomes

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-2.2698, 10) \tag{4.40}$$

To determine the behaviour of the solution for other negative values of $\alpha$ we consider the case when $\alpha = -10$, $a = 3$, $b = 5$ and $c = 2$ making $\Delta = 1$ and $\alpha = -20$, $a = 7$, $b = 10$ and $c = 3$ making $\Delta = 16$. All three graphs are graphically represented below.




### 4.2.11 $\alpha < 0$ and $\Delta < 0$

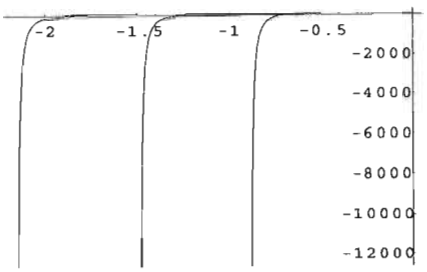
Under these conditions equation (4.20) becomes

$$\mathcal{Y}'' + 2\mathcal{Y}' + \frac{7}{4}\mathcal{Y} + \mathcal{Y}^2 = 0 \tag{4.41}$$

where $\alpha = -2$, $a = 1$, $b = 1$ and $c = 1$, thereby making $\Delta = -3$. The solution is

$$\mathcal{Y}(\mathcal{X}) = \text{InterpolatingFunction}(-2.16977, 10) \tag{4.42}$$

For the case $\alpha = -10$, $a = 6$, $b = 3$ and $c = 4$, thereby making $\Delta = -87$, and the case $\alpha = -20$, $a = 9$, $b = 10$ and $c = 11$ thereby making $\Delta = -296$. A graphical representation of the three graphs is




### 4.2.12 Discussion

It is evident from the above numerical results that the behaviour of solutions of equation (4.20) falls into three distinct classes: $\alpha = 0$, $\alpha < 0$ and $\alpha > 0$. In fact one can argue that the latter two are also related. The importance of $\alpha$ in (4.20) is then clear. As the coefficient of $\mathcal{Y}'(\mathcal{X})$ it plays a pivotal role in the behaviour of the solution of (4.20). It is interesting to note that the special cases $\alpha = \pm 5\sqrt{\Delta}$, $\Delta > 0$ do not stand out in any way in spite of the analytical results presented in Maharaj *et al* (1996). What does stand out is the dramatic change in behaviour for $\Delta \leq -6$ in §4.2.3. A dynamical systems analysis of this case bears further investigation; we intend to pursue this research in future work. Analyses utilising the dynamical system approach in cosmology are generating useful results as shown in the works of Billyard *et al* (1999a, 1999b, 2000). Also, see the comprehensive review of Wainright and Ellis (1997).

In the preceding sections, we were only interested in the behaviour of (4.20). How-

ever, $\mathcal{Y}(\mathcal{X})$ has no obvious physical meaning. The physical variables are $\lambda$, $\nu$ and $r$. As a result, we reverse the transformations (4.6)-(4.8) in the case of $\alpha = 0$, $\Delta = 0$ to gain some feel for the behaviour of the metric functions $\lambda$ and $\nu$.

From (4.7) we have

$$\lambda = -\log[L] \tag{4.43}$$

If $\alpha = 0$, then (4.17) becomes

$$\mathcal{Y} = LA^{-\frac{1}{2}} \tag{4.44}$$

Solving for $L$ in (4.44) produces

$$L = \mathcal{Y}A^{\frac{1}{2}} \tag{4.45}$$

Now, using (4.45) in (4.43), $\lambda$ takes the form

$$\lambda = -\log[\mathcal{Y}A^{\frac{1}{2}}] \tag{4.46}$$

and solving (4.18) for $x$ returns

$$x = \frac{-1-\mathcal{X}}{\mathcal{X}} \tag{4.47}$$

Using (4.47) and (4.15), we find that $A$ simplifies to

$$A = \frac{1}{\mathcal{X}^2} \tag{4.48}$$

From (4.46) we have

$$\lambda = -\log\left[\frac{\mathcal{Y}}{\mathcal{X}}\right] \tag{4.49}$$

The solution for $\mathcal{Y}$ is given via

```
NDSolve[y''[x] + y[x]^2 == 0, y'[0]==1, y[0]==0}, y[x], {x, -10, 10}]
```

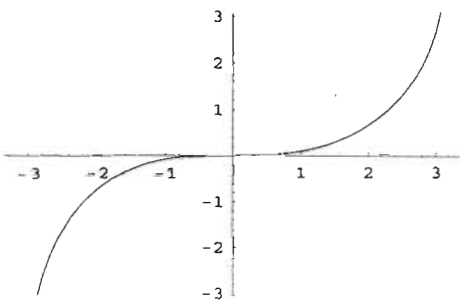
Equation (4.49) now takes the form

$$\lambda = -\log\left[\frac{1}{\mathcal{X}}\ \text{InterpolatingFunction}(-3.21017, 6.42038)\right] \tag{4.50}$$

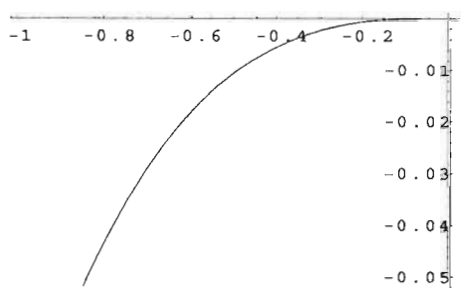
where the range for $\mathcal{X}$ is $(-3.21017, 6.42038)$. Finally we illustrate a graph for $\lambda$ via the `Plot` command

```
Plot[lambda  , {cx, -3.210170579073206, 6.420384693295558}]
```

where `lambda` $= \lambda$ and `cx` $= \mathcal{X}$ to produce the graph




Note that the range of $\mathcal{X}$ is calculated via (4.47) and noting that $x = r^2$ where $r$ is the radius. We now plot the graph for $\mathcal{X}$ over the range $(-1, 0)$




Due to the numerical solution of (4.20), the results only hold for a static spacetime.

This means that $e^{\nu}$ in (4.2) must be zero. This makes the above results unphysical. However, they were presented purely for illustrative purposes only.

## 4.3 A Particular Example

The behaviour of the energy density $\mu$ and the pressure $p$ in inhomogeneous spherically symmetric models is not well understood as pointed out by Krasinski (1997). This is related to the complex nature of the gravitational potentials which makes an analytical treatment difficult. Even in simple cases studied by Bonnor (1956) little progress was possible. We pursue the study of the profiles of $\mu$ and $p$ using a numerical approach in this section in contrast to the conventional analytic treatment. This approach proves to generate useful results. In the previous section we obtained a form for $F$ in (4.11), and thereafter solved for $L$ using the Lie analysis of differential equations. However, the process was rather convoluted. A number of transformations were made which obscured the physical significance of the solutions obtained. Here, we show that (4.9) can be directly solved for a particular $F$.

Following Srivastava (1987) we chose

$$F(x) = (x+1)^{-5} \tag{4.51}$$

Then (4.9) has solution

$$L = \frac{6(1+x)^3}{((1+x)c(t)-1)^2} \tag{4.52}$$

where $c(t)$ is an arbitrary function of integration.

We can now evaluate the expressions for the energy density $\mu$ and the pressure $p$ as given in equations (4.4) and (4.5) respectively. Both the energy density and the pressure are dependant on $\lambda$. From equation (4.7) we find that

$$\lambda = -\log[L] \tag{4.53}$$

Taking (4.6) and (4.52) into account we have

$$\lambda = -\log\left[\frac{6(1+r^2)^3}{((1+r^2)c(t)-1)^2}\right] \tag{4.54}$$

We can now evaluate the energy density, $\mu$, in equation (4.4) (where $f = 1$) using the $\lambda$ obtained in equation (4.54). Mathematica then simplifies (4.4) to

$$\begin{aligned}
\mu \;=\; & \left[\frac{1}{r(-1+(1+r^2)c(t))^5}\right]\Bigg(3\Bigg(-r(e^2-144(1+r^2)^4(-1+4r^2)) \\
& -10e^2r(1+r^2)^2c(t)^2+10e^2r(1+r^2)^3c(t)^3-5e^2r(1+r^2)^4c(t)^4 \\
& +e^2r(1+r^2)^5c(t)^5-48(1+r^2)^6\log\left[\frac{6(1+r^2)^3}{(-1+(1+r^2)c(t))^2}\right] \\
& +(1+r^2)c(t)\Bigg(r(5e^2-48(1+r^2)^4(-1+2r^2))+48(1+r^2)^6 \\
& \times\;\log\left[\frac{6(1+r^2)^3}{(-1+(1+r^2)c(t))^2}\right]\Bigg)\Bigg)\Bigg)
\end{aligned} \tag{4.55}$$

In a similar manner the expression for the pressure in (4.5) is then obtained as

$$p = -3e^2 - \frac{144(1+r^2)^4(-2+r^2+(1+r^2)c(t))}{(-1+(1+r^2)c(t))^5} \tag{4.56}$$

Thus far the energy density $\mu$ and the pressure $p$ were obtained in general with $c(t)$ being an arbitrary function of integration. However, to obtain meaningful density and pressure profiles we must fix $c(t)$. The different cases considered are

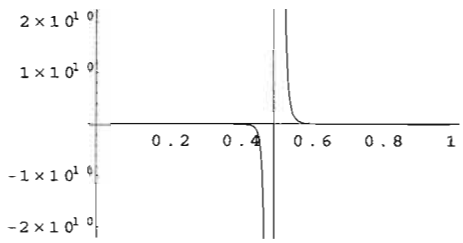
$$\begin{aligned}
c(t) &= t \\
c(t) &= t^2 \\
c(t) &= \exp t \\
c(t) &= \sin(t)
\end{aligned}$$

### 4.3.1 $c(t) = t$

We are interested in the profiles at specified radii and particular times. For this particular case, at $r = 1$, (4.55) becomes

$$\mu = \frac{3\left(768(9-2t) + e^2(-1+2t)^5 + 3072(-1+2t)\log\left[\frac{48}{(1-2t)^2}\right]\right)}{(-1+2t)^5} \tag{4.57}$$

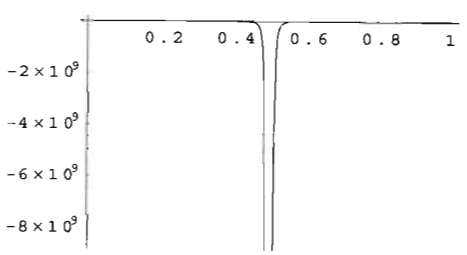
The graphical representation of $\mu$ is given by




The pressure $p$ at $r = 1$ in (4.56) is

$$p = -\frac{3(768 + e^2(1-2t)^4)}{(1-2t)^4} \tag{4.58}$$

The graph for this expression takes the form

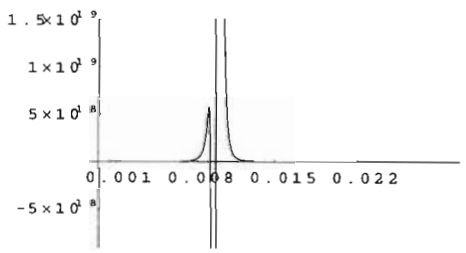


Note the presence of singularities in both graphs at $t = \frac{1}{2}$.

We now consider another expression for the energy density $\mu$, this time at a radius $r = 10$. Equation (4.55) becomes

$$\mu = 3\left(5(e^2(-1+101t)^5 - 4994899248(-1197+20099t)) + 25476483614424 \right.$$
$$\left. \times (-1+101t)\log\left[\frac{6181806}{(1-101t)^2}\right]\right) \div (5(-1+101t)^5) \tag{4.59}$$

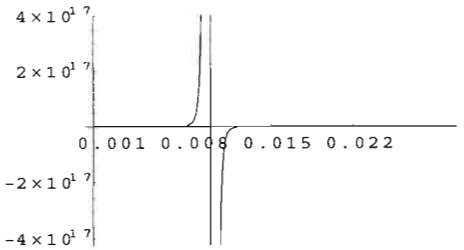
which is illustrated as




Equation (4.56) (with $r = 10$) is reduced to

$$p = -\frac{3(e^2(-1+101t)^5 + 4994899248(98+101t))}{(-1+101t)^5} \tag{4.60}$$

and the corresponding graph is

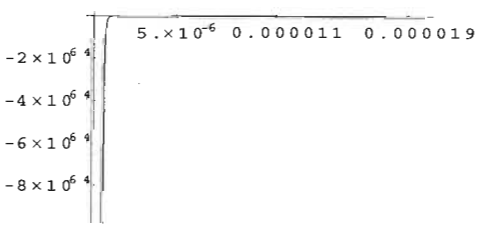



In these graphs singularities arise at $t = \frac{1}{1+10^2}$.

We now present profiles at specific times while varying the radii. The first case considered is $t = 1$. Equation (4.55) reduces to

$$\mu = 3\left(r(e^2r^{10} - 48(1+r^2)^4(2-11r^2+2r^4)) + 48r^2(1+r^2)^6 \right.$$
$$\left. \times \log\left[\frac{6(1+r^2)^3}{r^4}\right]\right) \div r^{11} \tag{4.61}$$

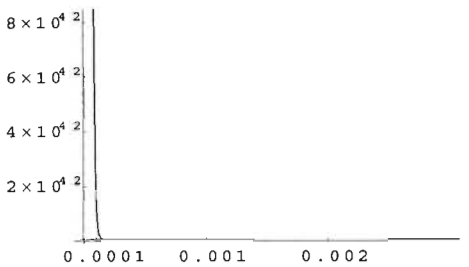
with graphical representation




Equation (4.56) takes the form

$$p = -\frac{3\,(e^2 r^{10} + 48(1+r^2)^4(-1+2r^2))}{r^{10}} \tag{4.62}$$

for $t = 1$ which is illustrated as



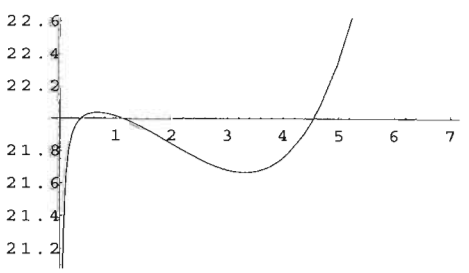

As is clear from (4.55) and (4.56), the graphs have singularities at $r = 0$.

In the case of $t = 10$, equation (4.55) for the energy density becomes

$$\mu = 3\Big(r(e^2(9+10r^2)^5 - 48(1+r^2)^4(-7-2r^2+20r^4)) + 48(1+r^2)^6$$

$$\times\,(9+10r^2)\log\left[\frac{6(1+r^2)^3}{(9+10r^2)^2}\right]\Big) \div (r(9+10r^2)^5) \tag{4.63}$$

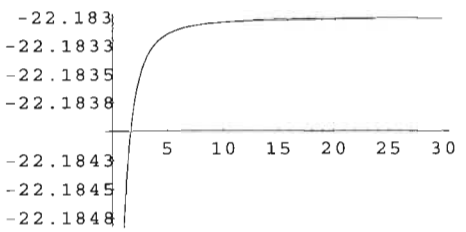
with representation



Equation (4.56) becomes

$$p = -\frac{3(e^2(9+10r^2)^5 + 48(1+r^2)^4(8+11r^2))}{(9+10r^2)^5} \tag{4.64}$$

when $t = 10$. The corresponding graph is

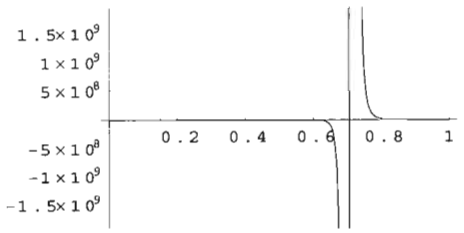



At $t = 10$, we have moved away from the singularities experienced before.

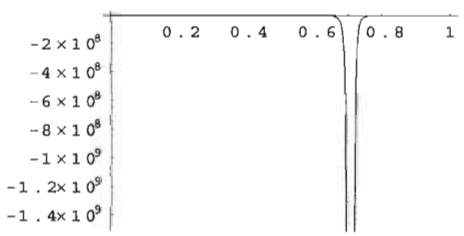
It is clear from the above eight graphs that both the energy density $\mu$ and pressure $p$ are singular at some $t$ and at some $r$. In an attempt to verify whether these singularities can be avoided for other functions $c(t)$, we plot similar graphs of $\mu$ and $p$ in the next section.

### 4.3.2 $c(t) = t^2$

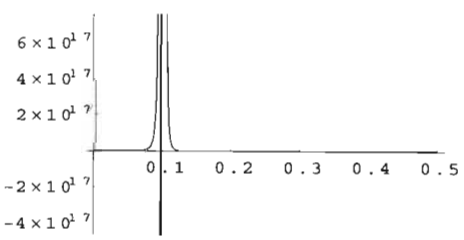
We now consider the case $c(t) = t^2$. Taking $r = 1$ the energy density $\mu$ is illustrated below.



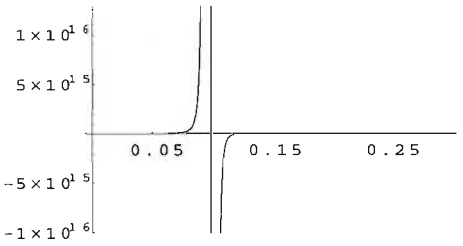

and the pressure $p$ is graphically represented as



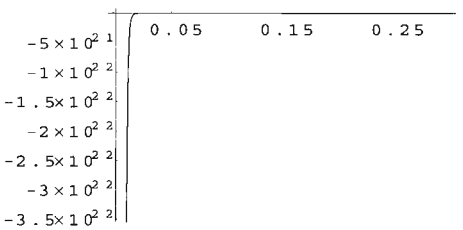

If $r = 10$, then the graph for the energy density is given by



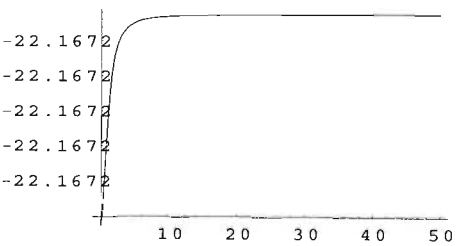
with pressure being



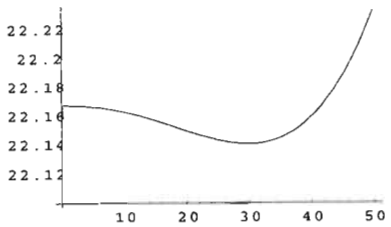

Fixing the time for the energy density at $t = 1$ we have



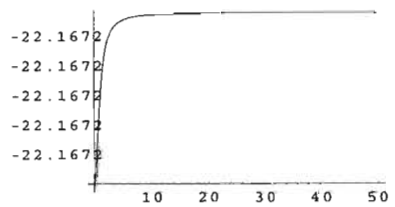

with the pressure being



At time $t = 10$ the energy density is

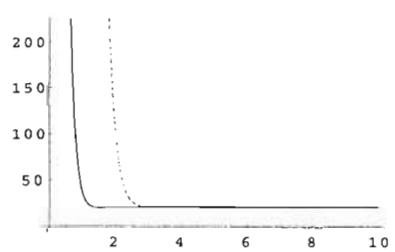



and the pressure is illustrated as

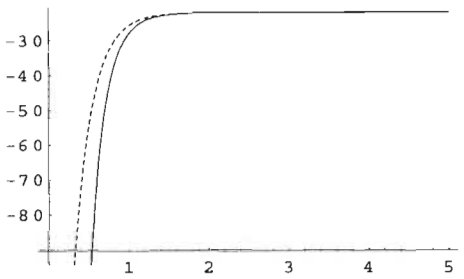



### 4.3.3 $c(t) = \exp t$

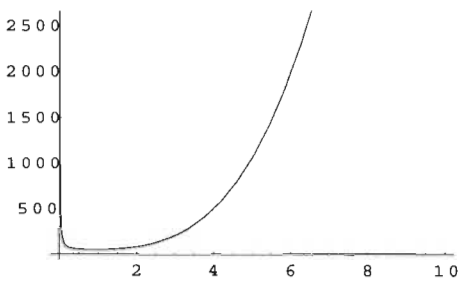
The next case that we evaluate is $c(t) = \exp t$. For this situation the graphical representation for $r = 1$ and $r = 10$ (dotted line) for the energy density is



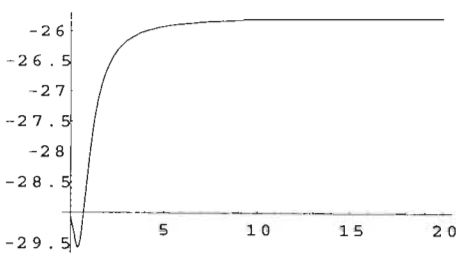
The pressure is given by



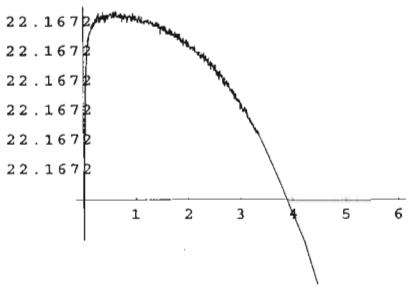

Similarly the energy density for $t = 1$ is represented as



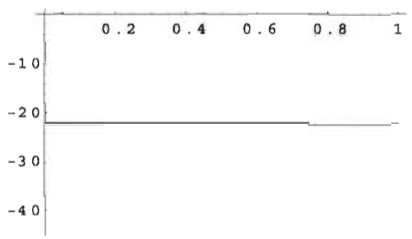

and the pressure is



We now consider the energy density at $t = 10$

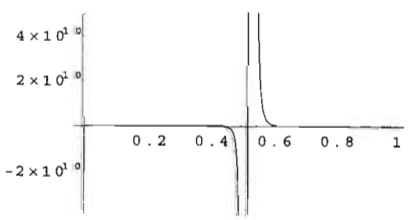



and the graph for the pressure is

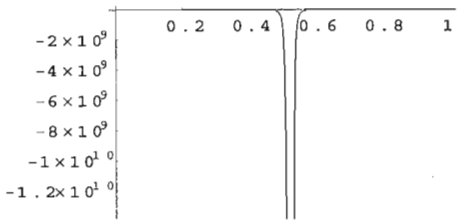



### 4.3.4 $c(t) = \sin t$

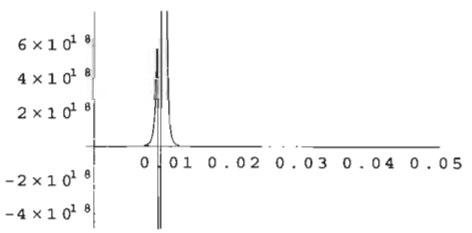
Finally we have the case $c(t) = \sin(t)$. The energy density for radius $r = 1$ is



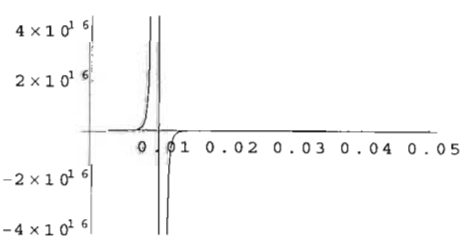
with the pressure being



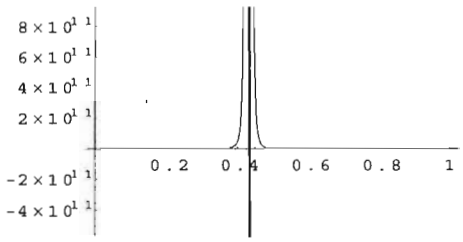

We now consider the energy density for the case $r = 10$ and obtain



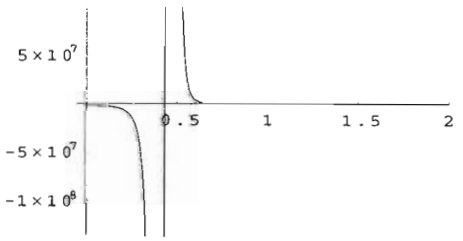

The pressure is given by



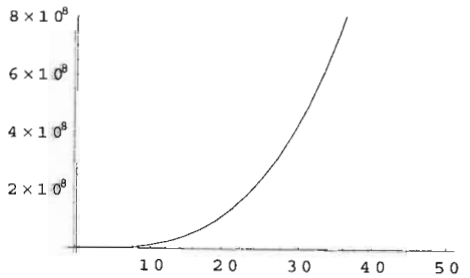
We now view the energy density at $t = 1$



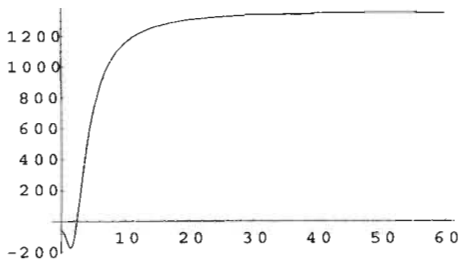

and the illustration for the pressure is




At $t = 10$ the energy density becomes



with pressure




It is evident from the graphs for the energy density and pressure that the various choices for $c(t)$ yield singularities in most of the plots. As a result, our solutions can only be used to describe cosmological situations for restricted regions of spacetime; the astrophysical applications are quite limited. However, the function $c(t)$ is arbitrary. It would be of considerable interest to obtain an appropriate form of $c(t)$ that leads to nonsingular results. This work is ongoing.

# Chapter 5

# Conclusion

The research conducted in this thesis investigated the depth, scope and power of Mathematica in solving differential equations that arise in physical situations. In most cases graphical representations were presented to visually interpret the behaviour of the solutions. With the aid of this software package, we analysed and investigated equations governing planetary orbits which arose in Newtonian theory and relativistic theory. Analytic solutions were first obtained and a numerical approach was used to verify the numerical algorithms within Mathematica. A perturbative approach was then used to solve the relativistic equation and hence confirm the numerical results. As a second application we considered a nonlinear Emden-Fowler equation which governs the behaviour of inhomogeneous gravitational fields. A detailed analysis of the related autonomous equation, arising in the Lie analysis, was performed. Thereafter a number of plots describing the profiles of the energy density and the pressure were generated. We now highlight the main points and conclusions arrived at in this thesis.

The principle tool used in this entire thesis to analyse and investigate particular differential equations was the software package called Mathematica. It is obvious that a program of this nature helps in performing calculations quickly and accurately, in most cases. With the use of this technology, time may be freed to analyse the solution

rather than on solving the problem. However, this program does have some flaws that were discovered during the course of this thesis with some of the algorithms.

In particular, the `NDSolve` algorithms were reported to have failed to achieve desired results. In the case of the first order Newtonian and relativistic equations in Chapter 3, nonsensical results (not reported here) were obtained. Once we considered the linear second order versions of the equation we experienced no problems at all. Despite these shortcomings we have found Mathematica to be extremely useful and contains features far outweighing the shortcomings. However, mathematics is based on accuracy and the user is cautioned to verify results.

Having explained the capabilities of Mathematica we then used it to investigate the differential equations for planetary orbits. In particular we analysed the orbits of Mercury, Earth and Jupiter. The analytic algorithms in the function `DSolve` provided simple solutions for a complex equation in Newtonian theory; however, it could not provide any meaningful results for the relativistic equation. The `NDSolve` command was used to obtain numerical solutions which were then compared to the analytical results. The numerical results were consistent with the analytical solutions. In plotting the graph of the relativistic equation, we saw a distinct thickening of the orbits for the planets Mercury, Earth and Jupiter. On closer analysis we discovered a distinct precession of the orbits. A perturbative approach was also used to analyse the relativistic equation for the planet Mercury which verified the numerical results. Using this technique we noticed a distinct precession in the graph over the range $[-5\pi, 5\pi]$. We were not able to notice this effect using the numerical approach for the same range. The numerical amount of perihelion shift was also obtained for the planets Mercury, Earth and Jupiter.

Mathematica was again used to investigate the nonlinear Emden-Fowler equation of order two. Having performed the Lie analysis we obtained a simple second order autonomous equation and found that the behaviour of the solutions fell into three distinct classes $\alpha = 0$, $\alpha < 0$ and $\alpha > 0$. One can argue that the cases $\alpha < 0$ and $\alpha > 0$ are related. We also found that the special case $\alpha = \pm\sqrt{\Delta}$ for $\Delta > 0$ does

not stand out in any way in spite of the analytical results presented by Maharaj *et al* (1996). What was evident was the dramatic change in behaviour for $\Delta \leq -6$ for the case $\alpha = 0$ and $\Delta < 0$. Thereafter we generated plots of the energy density and pressure for a range of values corresponding to radii and time values for different cases that arose. This is a preliminary treatment and simple forms were chosen that may not correspond to realistic situations. However these treatments demonstrate that it is possible to investigate these nonlinear gravitational fields, in principle, even though the original equation is nonlinear. It is the nonlinearity that prevented comprehensive analyses in the past. Only special cases were considered in previous investigations. Our treatment demonstrates that a comprehensive analysis is possible for realistic matter distributions.

# Bibliography


[1] Abel M L and Braselton J P 1994 *Mathematica by example* (California: Academic Press)

[2] Abel M L and Braselton J P 1997 *Differential Equations with Mathematica* (California: Academic Press)

[3] Billyard A P, Coley A A and Lidsey J E 1999a *Phys. Rev. D* **59** 123505

[4] Billyard A P, Coley A A and Lidsey J E 1999b *J. Math. Phys.* **40** 5092

[5] Billyard A P, Coley A A and Lidsey J E 2000 *J. Math. Phys.* **41** 6277

[6] Blachman N 1992 *Mathematica: A Practical Approach* (New Jersey: Prentice-Hall)

[7] Bluman G W and Kumei S 1989 *Symmetries and Differential Equations* (New York: Springer-Verlag)

[8] Bonnor W B 1956 *Z. Astrophysik* **39** 143

[9] Crandall R E 1991 *Mathematica for the Sciences* (California: Addison-Wesley)

[10] de Felice F and Clark C J S 1990 *Relativity on Manifolds* (Cambridge: Cambridge University Press)

[11] Dicke R H and Goldstein H M 1967 *Phys. Rev. Lett.* **18** 313

[12] Duncombe R L 1956 *Astronom. J.* **61** 174

[13] Foster J and Nightingale J D 1998 *A Short Course in General Relativity* (New York: Springer)

[14] Ganzha V G and Vorozhtsov E V 1996 *Numerical Solutions for Partial Differential Equations - Problem Solving Using Mathematica* (New York: CRC Press)

[15] Gaylord R J and Wellin P R 1995 *Computer Simulations with Mathematica* (New York: Telos)

[16] Govender J 1997 *Spherically Symmetric Cosmological Solutions* (Ph.D. dissertation: University of Natal)

[17] Gray J W 1994 *Mastering Mathematica* (California: Academic Press)

[18] Gray T W and Glynn J 1991 *Exploring Mathematics with Mathematica* (California: Addison-Wesley)

[19] Gray T W and Glynn J 1992 *The Beginner's Guide to Mathematica 2* (California: Addison-Wesley)

[20] Hallburd 1999 *Nonlinearity* **12** 931

[21] Hawking S W and Ellis G F R 1973 *The Large Scale Structure of Spacetime* (Cambridge: Cambridge University Press)

[22] Herlt E 1996 *Gen. Rel. Grav.* **28** 919

[23] Krasinski A 1997 *Inhomogeneous Cosmological Models* (Cambridge: Cambridge University Press)

[24] Kustaanheimo P and Qvist B 1948 *Soc. Sci. Fennica, Commentationes Physico-Mathematicae* **XIII** 16

[25] Kythe P K, Puri P and Schaferkotter M R 1997 *Partial Differential Equations and Mathematica* (New York: CRC Press)

[26] Leach P G L 1981 *J. Math Phys.* **22** 465

[27] Lie S 1912 *Vorlesungen uber Differentialgleichungen* (Leipzig/Belin: Teubner)

[28] Maeder R 1990 *Programming in Mathematica* (California: Addison-Wesley)

[29] Maharaj S D, Leach P G L and Maartens R 1996 *Gen. Rel. Grav.* **28** 35

[30] Mahomed F and Leach P G L 1990 *J. Math Anal. Appl.* **151** 80

[31] Mahomed F and Leach P G L 1991 *J. Math Phys.* **32** 2051

[32] Misner C W, Thorne K S and Wheeler J A 1973 *Gravitation* (San Francisco: Freeman)

[33] Moodley K 1998 *Aspects of Spherically Symmetric Cosmological Models* (M.Sc. dissertation: University of Natal)

[34] Moulton F R 1970 *An Introduction to Celestial Mechanics* (New York: Dover)

[35] Nordtveldt K and Will C M 1972 *Astrophys. J.* **177** 775

[36] Schutz B F 1985 *A First Course in General Relativity* (Cambridge: Cambridge University Press)

[37] Shapiro I I 1972 *Gen. Rel. Grav.* **3** 135

[38] Shapiro I I, Pettengill G H, Ash M E, Ingalls R P, Campbell D B and Ryce R B 1973 *Phys. Rev. Lett.* **28** 1594

[39] Shaw W T and Tigg J 1994 *Applied Mathematica* (California: Addison-Wesley)

[40] Smith C and Blachman N 1995 *The Mathematica Graphics Guidebook* (California: Addison-Wesley)

[41] Srivastava D C 1987 *Class. Quantum Grav.* **4** 1093

[42] Srivastava D C 1992 *Fortschr. Physik* **40** 31

[43] Stephani H 1968 *Commun. Math. Phys.* **9** 53

[44] Stephani H and Wolf T 1996 *Class. Quantum Grav.* **13** 1261

[45] Sussman R 1987 *J. Math. Phys.* **28** 1118

[46] Sussman R 1988a *J. Math. Phys.* **29** 945

[47] Sussman R 1988b *J. Math. Phys.* **29** 1177

[48] Sussman R 1989a *Gen. Rel. Grav.* **21** 1281

[49] Sussman R 1989b *Phys. Rev. D* **40** 1364

[50] Szafron D A 1977 *J. Math. Phys.* **18** 1673

[51] Szekeres P 1966 1989b *J. Math. Phys.* **7** 751

[52] Wagon S 1991 *Mathematica in Action* (San Francisco: Freeman)

[53] Wainwright J and Ellis G F R 1997 *Dynamical Systems in Cosmology* (Cambridge: Cambridge University Press)

[54] Wolfram S 1991 *Mathematica: A System for Doing Mathematics by Computer* (California: Addison-Wesley)

[55] Wolfram S 1996 *The Mathematica Book* (Cambridge: Cambridge University Press)

[56] Wolfram Research 1993 *Guide to Standard Mathematica Packages: Technical Report* (Illinois: Wolfram Research Inc)