Chapter 1

An Introduction to *Mathematica*

COPYRIGHTED MATERIAL

1.1 The Very Basics

Mathematica is an extremely powerful mathematical software package (or computer algebra system) that incorporates text editing, mathematical computation, and programming as well as 2D and 3D graphics capabilities. You can literally write a complete mathematics textbook using only *Mathematica* where your book includes all of the text and graphics in one smoothly flowing document. If you have never or only slightly used *Mathematica* before, then it will take some effort to learn how it works—believe me that it is well worth the time expended for the ability to do mathematically almost anything you can dream of that a computer might be able to do for you. In this introduction to *Mathematica*, you will see only a fraction of its capabilities, but hopefully enough to get you well on your way in doing 2D and 3D graphics, solving of equations, defining and using functions, lists and matrices, along with some basic mathematical programming.

This chapter discusses the fundamentals of using *Mathematica* for the novice user. If you are already familiar with *Mathematica*, you may wish to skip this chapter, although we warn you that to do so would be at your own risk. The new user of *Mathematica* will find it quite difficult in the beginning, but with practice and patience, you will master all of the basics and in time come to enjoy using *Mathematica*.

Mathematica files are called *notebooks*, and in a notebook you can place text along with input commands and their associated outputs which can be literally anything such as graphics, tables or lists, and functions. You can group the material in a notebook into different types of cells that are indicated on the right side of the notebook by brackets. At the top of the notebook you will see

Principles of Linear Algebra with *Mathematica*®, First Edition.
Kenneth Shiskowski and Karl Frinkle.
© John Wiley & Sons, Inc. Published 2011 by John Wiley & Sons, Inc.

the tab *Palettes* and under it is the *Writing Assistant*, which will allow you to create new cells and/or modify cells. You can use *Writing Assistant* to change the font, color, and size of the text in your cells and you can also do this using the *Format* tab at the top of the notebook. The word processing capabilities of *Mathematica* are very similar to those of *Microsoft Word* with $\boxed{\text{Ctrl}}+\boxed{\text{C}}$ as copy and $\boxed{\text{Ctrl}}+\boxed{\text{V}}$ as paste, $\boxed{\text{Ctrl}}+\boxed{\text{X}}$ as delete and copy, which can be pasted elsewhere.

The commands $\boxed{\text{Alt}}+\boxed{9}$ and $\boxed{\text{Alt}}+\boxed{7}$ will create *Input* and *Text* cells, respectively, after a horizontal line break between cells. Almost all *Mathematica* cells are *Input* or *Text* cells, or *Output* cells that are created when you activate an *Input* cell. *Input* and *Output* cells are normally in pairs with *Output* second directly following its *Input*. You can also create a new cell after a line break by typing in some text where you can control the type of cell you are writing in by using the menu which is open at the upper left of the screen next to the *Save* (or disk) icon.

Each section or chapter of a notebook file in *Mathematica* should be created as a section where the first cell of the section is a *Subtitle* cell that can be created by placing a horizontal bar between or just after a cell and then choosing *Subtitle* from the pulldown menu at the very top left of the lower ruler at the top of your screen. In order to create a subsection of this section (or chapter), do the same as just described but choose the *Subsection* from this menu. If you have not already used the *Window* tab to insert the *Toolbar* in your notebook, then please do so now. With the *Toolbar* in place, you can now change the type of cell you are in by using the pulldown menu at its far left. The *Ruler* can also be inserted into your notebook if you want it from the *Window* tab. Note that for those of us who like our text in a larger style, *Window* also has a *Magnification* feature that is quite handy.

If you wish to delete a cell (use $\boxed{\text{Ctrl}}+\boxed{\text{X}}$) or modify its entire contents in some way, then click on the cell tag or bracket on the right and then carry out the desired operation using *Writing Assistant* or the tabs at the top of the screen or simply $\boxed{\text{Ctrl}}+\boxed{\text{X}}$ for a complete deletion. In text, in order to create a new paragraph in a cell, use $\boxed{\text{Enter}}$. To do the same in an *Input* cell where commands are placed, use $\boxed{\text{Enter}}$ as well. If you wish to split a cell, then use $\boxed{\text{Shift}}+\boxed{\text{Enter}}$ with the cursor at the location of the split. In using *Writing Assistant* or any pulldown tab, if you click on the triangles on the left you will open or close one of the sections inside this tab. Note that text paragraphs are not necessarily indented automatically, so you must indent them yourself manually if you want this to happen.

If you wish to close a group of cells and see only the first cell of the group (which should be the title cell of the group), then double-click on the far-right bracket for the group. You will then see a cell bracket with an arrow to the right of the cell bracket of the title or first cell of the group. If you double-click on this arrow, then you can open all the cells of this group. The copy $\boxed{\text{Ctrl}}+\boxed{\text{C}}$ and paste $\boxed{\text{Ctrl}}+\boxed{\text{V}}$ features of *Mathematica* are the same as those of *Microsoft*

Word and other software. If you wish to change the size, font, or other feature of a collection of cells, select one of them by clicking on its bracket and then hold the $\boxed{\text{Ctrl}}$ key down while you select the rest of the cells—now go to the *Format* tab or other location and carry out your change.

It is strongly recommended that you save (use $\boxed{\text{Ctrl}}+\boxed{\text{S}}$) your work constantly since, like all software, *Mathematica* can glitch, which could cause you to loose some or all of your material. You should have backup copies of all of your work on a separate computer or flash drive since from our own personal experience, we know that unfortunate problems can occur.

If you are using *Mathematica* to do homework problems, it is strongly suggested that you place each problem in a single group of cells with the first cell as the title of the problem. This will make it much easier to organize your work both for yourself and the instructor who may read your material. After you have finished working in a particular *Mathematica* notebook, it is also recommended that you delete all of your output from the file unless it will take too long to recompute it. Most, if not all, of the size of a *Mathematica* file will be due to graphics, especially 3D graphics, and such files can become very large and consequently take *Mathematica* quite a while to open or save, and at such times an error can occur. Under the tab *Cell*, you have the command *Delete All Output*, which removes all output from the entire file—you might use this periodically while working in a notebook in order to shorten the file. When you reopen a notebook where all output has been deleted, you can reconstruct it all by going to the tab *Evaluation* and using the command *Evaluate Cells*; the *Input* cells will then be evaluated from the first one of the notebook to the last one.

If a *Mathematica* calculation is taking too long and/or you notice that there is an error in the input, then, in order to terminate the calculation, you should go to the *Evaluation* tab at the top of the screen and select *Abort Evaluation*. This should immediately halt the calculation in its tracks unless *Mathematica* is stuck in some enormous loop and cannot find its way out—then your only alternative might be to use $\boxed{\text{Ctrl}}+\boxed{\text{Alt}}+\boxed{\text{Delete}}$ and/or turn your computer off, that is, gently pull the plug on the machine, while apologizing to it.

Beware of using capital letters to define a quantity in *Mathematica* as it might already be a built-in command name that you cannot override with something else. You should also avoid using the capital letters C, D and N for any kind of variable or name in *Mathematica* as they are also command names. The commands **Clear** and **ClearAll** will undefine a quantity that you have named. If you use the command **Exit**[], it should clear everything from memory that you have defined and *Mathematica* has produced as output by quitting the *Mathematica* kernel, which is the core of *Mathematica*.

In order to define or name a quantity in *Mathematica*, you must first decide on an appropriate name that cannot be a previously used name or *Mathematica* command name, nor should it be a common variable name like x, y, and z, which you might use in equations or functions/expressions as a variable symbol.

You can never use the same symbol or name in *Mathematica* for more than one thing. Once you decide on a name such as *TrialName*, then in an *Input* cell say **TrialName** = (or :=) where, after the equal sign, you must give the expression that is the definition of *TrialName*. In *Mathematica*, an equal sign = is used for definitions, while a double equal sign == is used in equations. You can use != for not equals. The := is often used for defining functions since it suppresses output and evaluation of the named quantity.

If you are using a *Mathematica* command, but have forgotten how to use it, then place the cursor in the middle of the command name and hit the $\boxed{\text{F1}}$ key to have *Mathematica* bring up the *Help* file for this command name. You can also go directly to *Help* and type in the command name yourself, especially if you have forgotten its correct spelling. Don't forget that every command name in *Mathematica* has its first letter capitalized.

If you place a semicolon (;) at the end of a named input, then *Mathematica* will not give any associated output even though it internally carried out your command and stored it to the name given it. This feature can be useful when the output would be very long and you do not need to see it all displayed, only have it computed and/or stored.

Mathematica can use standard mathematical notation for powers k^2 and fractions $\frac{a}{b}$ where $\boxed{\text{Ctrl}}+\boxed{6}$ after the base k is typed will give a power location and $\boxed{\text{Ctrl}}+\boxed{/}$ after a numerator is typed will create a fraction and placement for the denominator—both keys must be used simultaneously. If you use these keys after a space, then *Mathematica* creates blank shells $\square^{\square}$ and $\frac{\square}{\square}$ for the appropriate quantities to be inserted.

Finally, if you are a novice or beginner at using *Mathematica*, then besides this introductory material there are many videos on *YouTube* that explain most of the basic features of *Mathematica*. It is strongly suggested that you seek these out and hopefully will find a few useful ones for doing what you are interested in. *Mathematica* itself has tutorials that you should consider using if you find them useful.

1.2 Basic Arithmetic

In this section, we will start to use *Mathematica* to do some basic arithmetic and algebra computations. In the arithmetic, which is done first, we will add and multiply, factor positive integers into products of powers of primes, find the greatest common divisor (GCD) and the least common multiple (LCM) of two positive integers, and more. In the algebra, we will factor polynomials, divide one polynomial into another to get their quotient and remainder, solve for the roots of a polynomial and also solve equations for their unknowns, and perform other algebraic operations.

In order to create an *Input* cell where you can do your calculations, go to the tab *Palettes* and bring up *Classroom Assistant*. Now click on the tab *Create*

Input Cell with the cursor at the end of our prior work. Now you will have a new *Input* cell as part of your current group of cells. Both palettes, *Classroom Assistant* and *Basic Math Assistant*, have symbols such as π in them as well as the natural number e.

In the first input cell below, you will find the command **1 + 1**. If you hit $\boxed{\text{Shift}}$+$\boxed{\text{Enter}}$ with the cursor on this line, then *Mathematica* will carry out your command and produce 2 as an output. After you get the output, *Mathematica* automatically assumes that you want another *Input* cell, and so typing right after an output will be in a new *Input* cell. You can also go back and insert a new *Input* cell by creating a horizontal bar between two *Input* cells by clicking on the region between the two cells, and then using $\boxed{\text{Alt}}$+$\boxed{9}$ while $\boxed{\text{Alt}}$+$\boxed{7}$ creates a text cell. You can insert a *Text* cell between *Input* cells by creating the horizontal line divider between cells by clicking on the space between the cells and then using *Text Cell* out of the tab *Text Cells* in *Writing Assistant*. In addition, if you type in a *Mathematica* command name such as **FactorInteger**, but now you have forgotten precisely how it works and need its help file, then put the cursor in the command name and hit $\boxed{\text{F1}}$.

Note that *Mathematica* will also recognize a space in a product as multiplication, although for safety sake you might want to put in all of your multiplications directly. *Mathematica* gives exact answers in a calculation if the inputs are all also exact values, but any value with a decimal point in it is treated by *Mathematica* as an approximation and it gives an approximate answer back. The command **N[V, m]** will give the approximate value of the quantity V to m digits of accuracy. *Mathematica* can answer most computational questions to an arbitrary number of digits of precision—look up the command **WorkingPrecision** to see how it can be done as an alternative to the use of the command **N**. In the last example of this arithmetic section we also multiply three complex numbers using **Product**—a complex number in *Mathematica* is expressed as $a + bi$ for a and b real numbers.

In order to get a power of something in *Mathematica* that is placing a superscript, use $\boxed{\text{Ctrl}}$+$\boxed{6}$ together to get an exponent location after you have already typed in the base. If you want a fraction in the same standard way, then type the numerator followed by $\boxed{\text{Ctrl}}$+$\boxed{/}$ and then the denominator in the location created:

$\mathbf{1+1}$

2

$\mathbf{Sum[k^2, \{k, 1, 10\}]}$

385

$\mathbf{Sum[\pi^k, \{k, 1, 3\}]}$

$\pi+\pi^2+\pi^3$

N[Sum[π^k, {k, 1, 3}], 10]

44.01747373

Sum[3.14159265^k, {k, 1, 3}]

44.0175

Product[$\frac{1}{k}$, {k, 1, 5}]

$\frac{1}{120}$

FactorInteger[90]

{{2, 1}, {3, 2}, {5, 1}}

$\mathbf{2^1 3^2 5^1}$

90

GCD[210, 90]

30

LCM[210, 90]

630

QuotientRemainder[83, 5]

{16, 3}

$\mathbf{5 \times 16 + 3}$

83

Now we switch from arithmetic to algebra. Our algebra will be mainly polynomial and similar to what we did in the arithmetic part above, although we will find the roots of polynomials as well. After the first two computations of multiplying out two polynomials and then dividing the one of larger degree by the smaller-degree one, we will name or define the two polynomials as *Poly1* and *Poly2*, and then repeat the process to see that *Mathematica* understands what we want. Also, we define the list called *QR* below which is the quotient first and remainder second in our division of *Poly2* by *Poly1*—a list is an ordered collection of objects that *Mathematica* places { } around its elements, with commas between the elements. Then *QR*[[1]] is the quotient and *QR*[[2]]

is the remainder in the division. We will discuss lists and matrices in the next chapter.

Mathematica expresses its equations with a double equal sign == while it uses a single equal sign to make a definition or assignment of a quantity to a name such as in the use of *Poly1* and *Poly2* below. Hence, **5 x + 3 y == 9** is an equation in *Mathematica*, while **Eqn1 = 5 x + 3 y == 9** assigns the name *Eqn1* to this equation for the time you are using this notebook unless you decide to change it. In *Mathematica*, you do not need to insert a multiplication sign, as *Mathematica* usually places a space between objects, which indicates multiplication.

One last bit of useful information is that *Mathematica* uses the percent sign **%** to refer to the last computed output and **%%** to refer to the next to last computed output. This can be helpful, as we will see below. We will use the **%** below when we want to change the roots of *Poly1* to a set of rules that can then be substituted back into *Poly1* to see that we get 0 (or very close to 0) back. As well, it is probably better to assign a name to your quantities in order to be better able to use them later and know what you are specifically talking about, and we do this for the roots of *Poly2*:

Expand$[(7x^3+5x^2-9x+1)(-4x^6-x^5+3x^4-2x^3+x^2-8x+5)]$

$5-53x+98x^2-16x^3-30x^4-31x^5+6x^6+52x^7-27x^8-28x^9$

PolynomialQuotientRemainder$[-4x^6-x^5+3x^4-2x^3+x^2-8x+5,$
$7x^3+5x^2-9x+1, x]$

$$\left\{\frac{1179}{2401}-\frac{170x}{343}+\frac{13x^2}{49}-\frac{4x^3}{7}, \frac{10826}{2401}-\frac{7407x}{2401}-\frac{14841x^2}{2401}\right\}$$

Poly1 $= 7x^3+5x^2-9x+1$

$1-9x+5x^2+7x^3$

Poly2 $= -4x^6-x^5+3x^4-2x^3+x^2-8x+5$

$5-8x+x^2-2x^3+3x^4-x^5-4x^6$

Expand[Poly1 Poly2]

$5-53x+98x^2-16x^3-30x^4-31x^5+6x^6+52x^7-27x^8-28x^9$

QR = PolynomialQuotientRemainder[Poly2, Poly1, x]

$$\left\{\frac{1179}{2401}-\frac{170x}{343}+\frac{13x^2}{49}-\frac{4x^3}{7}, \frac{10826}{2401}-\frac{7407x}{2401}-\frac{14841x^2}{2401}\right\}$$

Expand[QR[[1]] Poly1 + QR[[2]]]

$5-8x+x^2-2x^3+3x^4-x^5-4x^6$

NRoots[Poly1 == 0, x]

$x == -1.58331 \,\|\, x == 0.120547 \,\|\, x == 0.74848$

{ToRules[%]}

$\{\{x \to -1.58331\}, \{x \to 0.120547\}, \{x \to 0.74848\}\}$

Poly1 /. %

$\{0., 2.08167\times10^{-16}, 0.\}$

Poly2Roots = N[Roots[Poly2 == 0, x], 10]

$x == -1.532415683 \,\|\, x == 0.6278496205 \,\|$
$x == -0.4656906912 - 0.9837390675\,i \,\|$
$x == -0.4656906912 + 0.9837390675\,i \,\|$
$x == 0.7929737223 - 0.6840534163\,i \,\|$
$x == 0.7929737223 + 0.6840534163\,i$

Rules = {ToRules[Poly2Roots]}

$\{\{x \to -1.532415683\}, \{x \to 0.6278496205\},$
$\{x \to -0.4656906912 - 0.9837390675\,i\},$
$\{x \to -0.4656906912 + 0.9837390675\,i\},$
$\{x \to 0.7929737223 - 0.6840534163\,i\},$
$\{x \to 0.7929737223 + 0.6840534163\,i\}\}$

Poly2 /. Rules

$\{0.\times10^{-8}, 0.\times10^{-9}, 0.\times10^{-8} + 0.\times10^{-8}\,i,$
$0.\times10^{-8} + 0.\times10^{-8}\,i, 0.\times10^{-8} + 0.\times10^{-8}\,i, 0.\times10^{-8} + 0.\times10^{-8}\,i\}$

Rules[[2]][[1]][[2]]

0.6278496205

Poly2Roots[[2]][[2]]

0.6278496205

Poly = Expand[−4Product[x−Poly2Roots[[k]][[2]], {k, 1, 6}]]

$(5.00000000 + 0. \times 10^{-9}\,i) - (8.0000000 + 0. \times 10^{-8}\,i)\,x +$
$(1.0000000 + 0. \times 10^{-8}\,i)\,x^2 - (2.0000000 + 0. \times 10^{-8}\,i)\,x^3 +$
$(3.00000000 + 0. \times 10^{-9}\,i)\,x^4 - (1.00000000 + 0. \times 10^{-9}\,i)\,x^5 - 4x^6$

Sum[Re[Coefficient[Poly, x, k]] x$^{\mathbf{k}}$, {k, 0, 6}]

$5.00000000 - 8.0000000\,x + 1.0000000\,x^2 -$
$2.0000000\,x^3 + 3.00000000\,x^4 - 1.00000000\,x^5 - 4\,x^6$

Chop[Poly, 10^{-7}]

$5.00000000 - 8.0000000\,x + 1.0000000\,x^2 -$
$2.0000000\,x^3 + 3.00000000\,x^4 - 1.00000000\,x^5 - 4\,x^6$

Product[k − (k + 2) I, {k, 0, 3}]

40+160 i

1.3 Lists and Matrices

Now we look more closely at the different ways in which we can organize information into lists and matrices (a list of lists), and how these different structures work and can be manipulated. In *Mathematica*, there are no sets, since *Mathematica* requires that an ordering be placed on its data, and so it deals with lists that can be treated as sets if you ignore the order of the elements in the list. The list L of the elements a, b, and c in this order is given as **L = {a, b, c}** in *Mathematica*. Shortly, we will look at taking the union, intersection, complement, and concatenation (or joining) of lists as well as taking out parts of a list. The empty list is **{ }**, and **L[[k]]** is the kth element of list L.

A string S is a grouping or ordered collection of characters or symbols with quotes around them such as **S = "Mary had a little lamb"**. The string assigned to the name S is the sentence between the two double quotes. Strings often show up as the title to *Mathematica* plots and similar structures.

A sequence or table in *Mathematica* is a list of objects created from a formula such as $\{1, 4, 9\}$ can be created as the output from the input **Table[k$^{\mathbf{2}}$, {k,1,3}]**.

L1 = {a, b, 1, 3, c, 5}

{a, b, 1, 3, c, 5}

```
L2 = {7, 2, 5, 2, 5, 1, c, b}

{7, 2, 5, 2, 5, 1, c, b}

Union[L1, L2]

{1, 2, 3, 5, 7, a, b, c}

Intersection[L1, L2]

{1, 5, b, c}

Complement[L1, L2]

{3, a}

Join[L1, L2]

{a, b, 1, 3, c, 5, 7, 2, 5, 2, 5, 1, c, b}

Length[L1]

6

L1[[2]]

b

Squares = Table[k^2, {k, 1, 10}]

{1, 4, 9, 16, 25, 36, 49, 64, 81, 100}

Sum[Squares[[k]], {k, 1, 10}]

385
```

Now we switch to matrices or two dimensional arrays of rows and columns of entries. *Mathematica* considers a matrix M to be a list of lists where each element of M is one of the rows of the matrix M, and the rows must all have the same length. As such, **M = {{1,2,3}, {4,5,6}, {7,8,9}}** has $\{1, 2, 3\}$ as its first row, $\{4, 5, 6\}$ as its second row, and finally $\{7, 8, 9\}$ as its third row. We will look at examples of adding, multiplying, inverting, transposing, and finding determinants of matrices where multiplication of matrices is indicated by a period or dot (**.**) between their names as given in the fifth line of input below.

In order to define a matrix M and have it displayed in the proper matrix format as rows and columns, use round parentheses around your definition of M followed by **// MatrixForm**. Then if you wish to manipulate M with other matrices so defined, there will be no problems, but remember to always use // **MatrixForm** after your computations or definitions to get the proper matrix format:

M = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}

{{1,2,3}, {4,5,6}, {7,8,9}}

MatrixForm[M]

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

K = {{−5, 7, 1}, {0, −4, 6}, {2, −1, 9}}

{{−5,7,1}, {0,−4,6}, {2,−1,9}}

MatrixForm[K]

$$\begin{pmatrix} -5 & 7 & 1 \\ 0 & -4 & 6 \\ 2 & -1 & 9 \end{pmatrix}$$

(L = M + K) // MatrixForm

$$\begin{pmatrix} -4 & 9 & 4 \\ 4 & 1 & 12 \\ 9 & 7 & 18 \end{pmatrix}$$

MatrixForm[M.K]

$$\begin{pmatrix} 1 & -4 & 40 \\ -8 & 2 & 88 \\ -17 & 8 & 136 \end{pmatrix}$$

Det[M]

0

Det[K]

242

Det[M.K]

0

Inverse[M]

Inverse::sing : Matrix {{1,2,3}, {4,5,6}, {7,8,9}} is singular. ≫
Inverse[{{1, 2, 3},{4, 5, 6}, {7, 8, 9}}]

Inverse[K] // MatrixForm

$$\begin{pmatrix} -\frac{15}{121} & -\frac{32}{121} & \frac{23}{121} \\ \frac{6}{121} & -\frac{47}{242} & \frac{15}{121} \\ \frac{4}{121} & \frac{9}{242} & \frac{10}{121} \end{pmatrix}$$

Transpose[M] // MatrixForm

$$\begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

1.4 Expressions versus Functions

This section will look at the differences in *Mathematica* between expressions and functions and how to manipulate and evaluate both. You should think of an expression as the rule for some function f; that is, if $f(x) = 5x + 9$, then $5x + 9$ is an expression which is the rule of the function f. *Mathematica* treats a function very differently than it treats an expression—you should think of an expression as a string of symbols while a function is a string of symbols (its rule or expression) with a method of evaluating its expression at different values of the variable(s) in the expression.

Let's now look at the difference in *Mathematica* between the expression $g = 5x + 9$ and the function f with rule $f(x) = 5x + 9$, and how each of them can be evaluated at $x = 1$. Note that normal function evaluation can be done so that f evaluated at $x = 1$ is **f[1]** while the expression g can be evaluated at $x = 1$ by the substitution command **g /. x→1**. If you want to compose the function f with the built-in function **Sin[x]**, then **f[Sin[x]]** will do it—this composition can also be done using **f@Sin[x]** or **Nest[f, Sin[x], 1]**, and its result is an expression, not a function. If you want f to be composed with itself k times, then use **Nest[f, f[x], k]**.

```
g = 5 x + 9
```

9+5 x

```
g /. x→1
```

14

```
f[x_] = 5 x + 9
```

9+5 x

```
f[1]
14
g /. x→Sin[x]
9+5 sin[x]
f[Sin[x]]
9+5 sin[x]
h = f@f[x]
9+5 (9+5 x)
Simplify[h]
54+25 x
Nest[f, f[x], 1]
9+5 (9+5 x)
Simplify[%]
54+25 x
Nest[f, f[x], 1] /. x→2
104
Nest[f, f[x], 5]
9+5 (9+5 (9+5 (9+5 (9+5 (9+5 x)))))
Simplify[%]
35154+15625 x
```

As the last topic of this section, let's do an example of a piecewise function and its graph (see Fig. 1.1). A piecewise function is one whose rule is given in parts or pieces where each part is used only when certain conditions are satisfied. Happily, *Mathematica* has a **Piecewise** command that we can utilize. Note that in defining the function $f(x)$ below that **&&** is the *Mathematica* notation for the logical AND in joining two statements:

f[*x*_] = Piecewise[{{Sin[x], x ≤ −2 }, {Cos[x], x > −2 && x ≤ 3}, {−x + 5, x > 3}}]

$$\begin{cases} \text{Sin[x]} & x \le -2 \\ \text{Cos[x]} & x > -2 \,\&\&\, x \le 3 \\ 5 - x & x > 3 \\ 0 & \text{True} \end{cases}$$

Plot[f[x], {x, −π, 2π}, PlotStyle→{Red, Thick}]

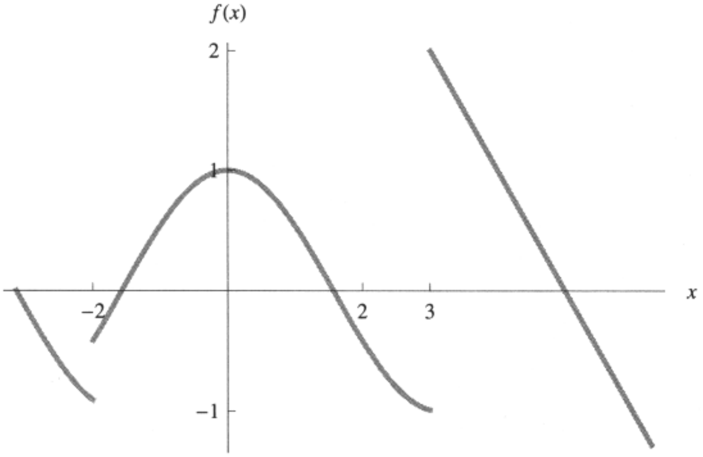

Figure 1.1: Plot of the piecewise function f.

1.5 Plotting and Animations

We begin our investigation into plotting with the basic 2D plotting of the graphs of functions $y = f(x)$ and expressions as well as plotting parametric curves $x = f(t)$, $y = g(t)$. We will plot single functions and parametric curves as well as several together and in combination in different colors. Besides these two types of curves, we will implicitly plot equations in the two variables x and y such as the unit circle with equation $x^2 + y^2 = 1$ or something more complicated. The implicit plotting of equations can be done using the **ContourPlot** command and **ContourStyle** option to control color, thickness, etc. of the plot.

Let's begin with simple function or expression plotting, and then move on to parametric curve plotting. In the first example, remember to get base e for the exponential function from the palette *Basic Math Assistant*. Also, in order to get a superscript or exponent, use $\boxed{\text{Ctrl}}$+$\boxed{6}$ together after the base is already in place. In order to create a fraction, first type the numerator and then hit $\boxed{\text{Ctrl}}$+$\boxed{/}$ to be able to place the denominator.

The plot option of **PlotStyle** can control color and thickness for graphics and control whether the graph lines are solid or dashed except when doing implicit plotting of equations using **ContourPlot** when **PlotStyle** switches to **ContourStyle** (see Figs. 1.2–1.7).

f[*x*_] = x Sin[x² ⅇˣ]

$x \operatorname{Sin}\left[e^{x} x^{2}\right]$

Plot[f[x], {x, −1, 2}, PlotStyle→Red, PlotRange→{−2, 2}]

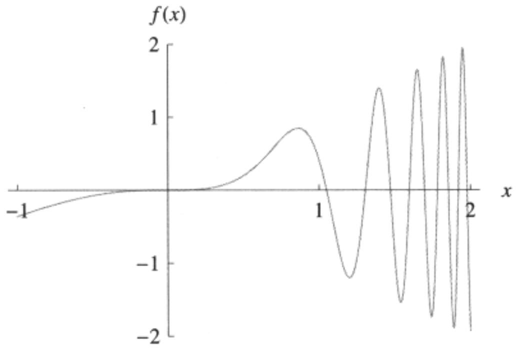

Figure 1.2: Output of the **Plot** command with various options.

g = $\mathbb{e}^{-\frac{1}{2}\,\mathrm{Sin}[3x]}$

$\mathbb{e}^{-\frac{1}{2}\,\mathrm{Sin}[3x]}$

Plot[{f[x],g}, {x,−1,2}, PlotStyle→{Directive[Red,Thickness[.005]], Directive[Blue,Thickness[.01]]}]

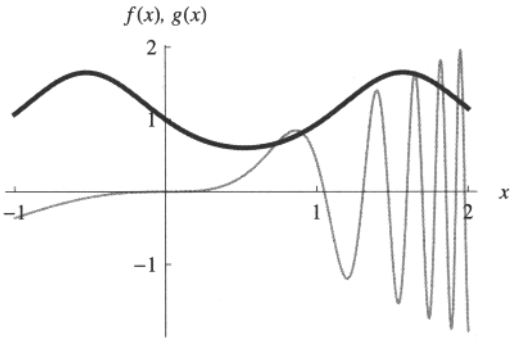

Figure 1.3: Plot of f and g together with separate options for each curve.

ParametricPlot[{Sin[2 t], Sin[3 t]}, {t, 0, 2π}, PlotStyle→Directive [Blue, Thick]]

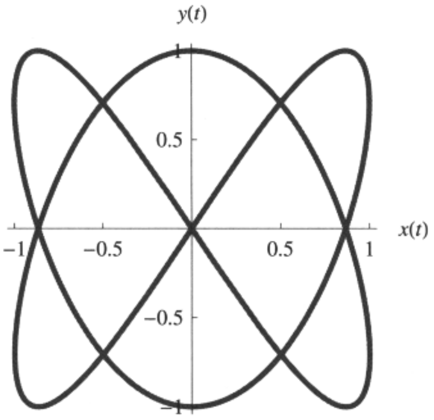

Figure 1.4: Plot of the parametric equation $(\sin(2t), \sin(3t))$.

ParametricPlot[{{Sin[2 t], Sin[3 t]}, {Sin[t], Cos[t]}},{t, 0, 2π}, PlotStyle→{Directive[Blue, Thick], Directive[Red, Thick]}]

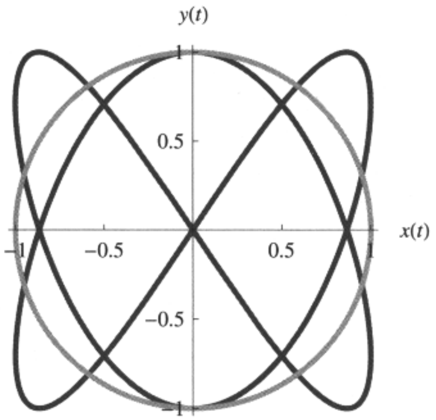

Figure 1.5: Two parametric curves plotted together.

ContourPlot$\left[\left\{\mathbf{x^2+y^2 == 1,\ x^4+y^4 == 1}\right\}\right.$**, {x, −1, 1}, {y, −1, 1}, ContourStyle→{Directive[Red, Thick], Directive[Blue, Thick]}]**

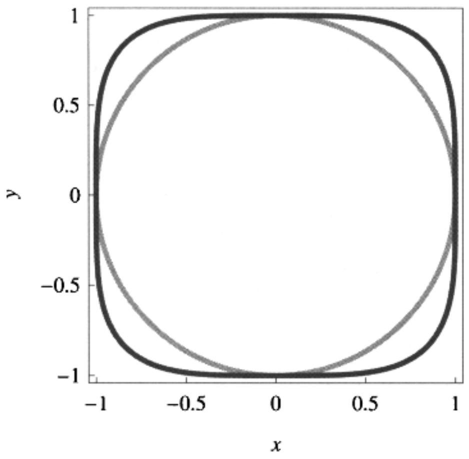

Figure 1.6: Example of the **ContourPlot** command with two implicitly defined relations.

ContourPlot$\left[\left\{\mathbf{\frac{(x-5)^2}{49}+\frac{(y-10)^2}{144} == 1,\ \frac{(x-5)^2}{49}-\frac{(y-10)^2}{144} == 1}\right\}\right.$**, {x,−10,25}, {y,−10,25}, ContourStyle→{Directive[Red,Thick], Directive[Blue, Thick]}]**

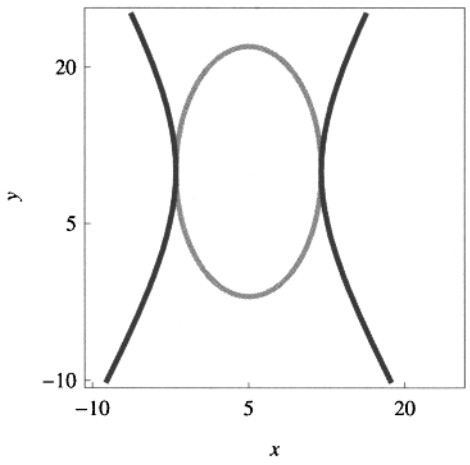

Figure 1.7: Second example of the **ContourPlot** command with two implicitly defined relations.

Now we turn our attention to creating a movie or animation in the xy-plane whose frames consist of plots of the three different types. Let's begin by plotting the function $y = \sin(x)$ from $x = 0$ to $x = A$ where the animation parameter A goes from 0 to 4π (see Figs. 1.8 and 1.9). It is followed by running two $y = \sin(x)$ animations at once based on the same animation parameter A:

Animate[Plot[Sin[x], {x, 0, A}, PlotRange→{{0,4π}, {−1.01,1.01}}, PlotStyle →Directive[Blue, Thick]], {A, 0.01, 4π}, AnimationRunning→False]

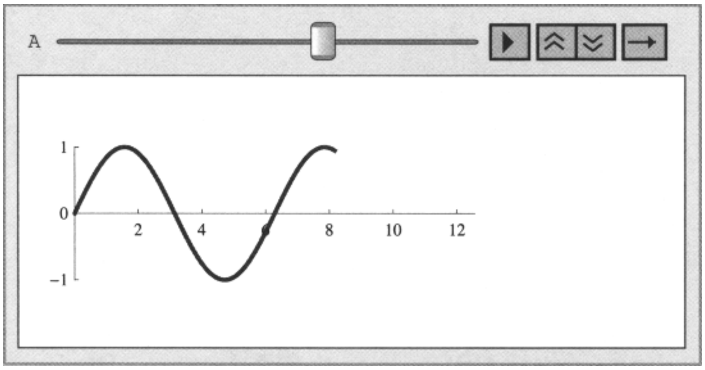

Figure 1.8: Animation of the plot of $\sin(x)$ for $x \in [0, A]$, here $A = 2.6\pi$.

Animate[{Plot[Sin[x], {x, 0, A}, PlotRange →{{0, 4π}, {−1, 1}}, PlotStyle→Directive[Blue, Thick]], Plot[Sin[x + A], {x, 0, 4π}, PlotStyle→Directive[Red, Thick]]}, {A, 0.01, 4π}, AnimationRunning→ False]

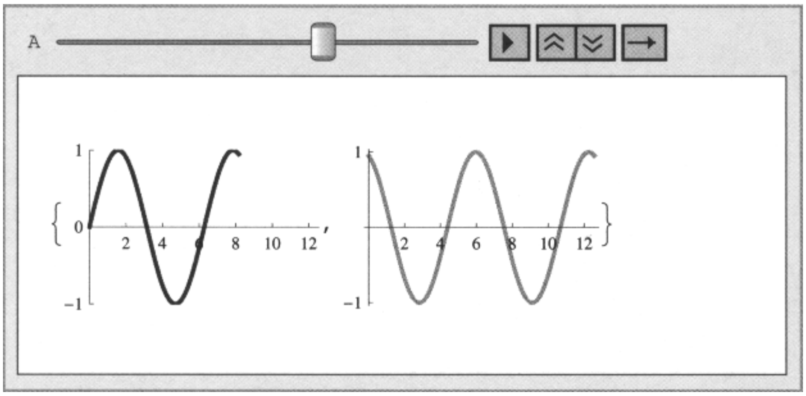

Figure 1.9: Animation of $\sin(x)$ and $\sin(x + A)$, $x \in [0, A]$. Here $A = 2.6\pi$.

We next do an animation (see Fig. 1.10) involving implicit plotting of an ellipse where the center is moving along the circle with center at the origin and radius 10. Here we make use of the **Epilog** option to put into each frame of our movie the circle of the ellipses' centers:

$$\mathbf{ellipses} = \frac{(\mathbf{x} - \mathbf{10\,Sin[A]})^2}{\mathbf{4.}} + \frac{(\mathbf{y} - \mathbf{10\,Cos[A]})^2}{\mathbf{25.}} == \mathbf{1;}$$

Animate[ContourPlot[Evaluate[ellipses /. A→B], {x, −15, 15}, {y, −15, 15}, ContourStyle→Directive[Blue, Thick], PlotPoints→100, Epilog→{Red, Thick, Circle[{0, 0}, 10]}], {B, 0, 2π}, AnimationRate →.15, AnimationRunning→False]

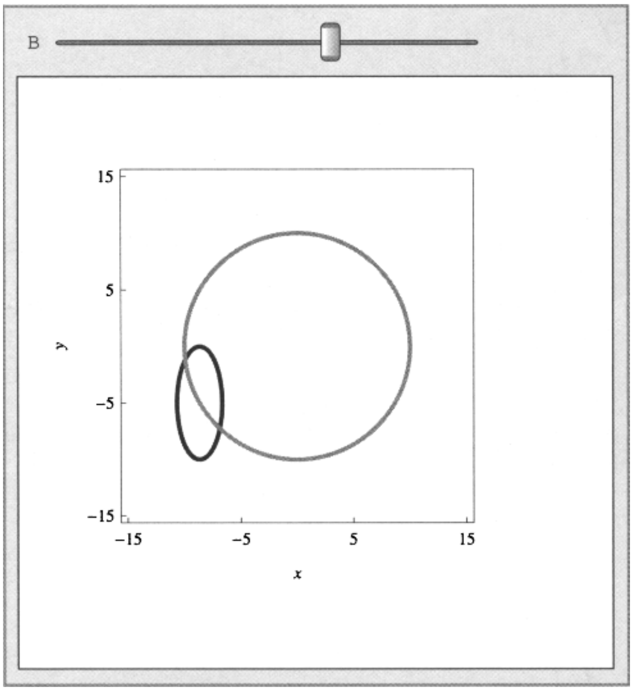

Figure 1.10: Animation of the rotation of the ellipse, here $B = \frac{4}{3}\pi$.

It is time to move function, parametric, and implicit plotting from the xy-plane to xyz-space. We begin by plotting functions $z = f(x, y)$, which give surfaces in xyz-space (see Fig. 1.11). This is followed by parametric curve and surface plotting:

$\mathbf{f[}x\_\mathbf{,}\ y\_\mathbf{]} = \mathbf{Sin[x + y]\ Cos[x - y] + 3;}$

$\mathbf{g} = \mathbf{Sin[x\ y]} + e^{-(x^2+y^2)}\mathbf{;}$

Plot3D[{f[x, y], g}, {x, −3, 3}, {y, −2, 2}, PlotStyle→{Red, Blue}, AxesLabel→Automatic]

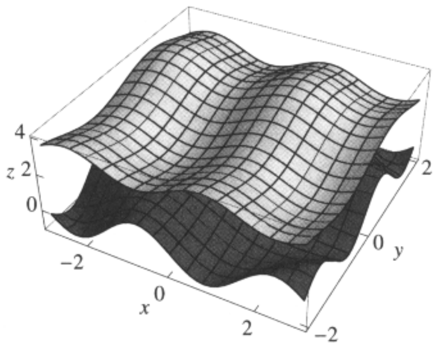

Figure 1.11: Plot of two surfaces.

Now we examine *parametric curve* plotting in space. A parametric curve, or *spacecurve* (see Figs. 1.12 and 1.13), is of the form $x = f(t)$, $y = g(t)$, $z = h(t)$ with one independent variable t. For spacecurves, we use the **ParametricPlot3D** command which is nearly identical to the **ParamatricPlot** command that was previously introduced.

ParametricPlot3D[{Cos[2 t], Sin[4 t], Cos[6 t]}, {t, 0, 2π}, PlotStyle →Directive[Blue,Thick], PlotPoints→250, AxesLabel→Automatic]

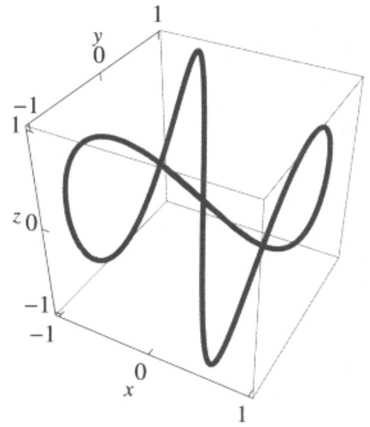

Figure 1.12: Plot of a spacecurve.

ParametricPlot3D$\left[\left\{e^{\frac{t}{100}}\text{ Cos[t]}, e^{\frac{t}{100}}\text{ Sin[t]}, \frac{t}{10}\right\}\right.$, {t, 0, 48π}, PlotStyle→Directive[Blue, Thick], PlotPoints→250]

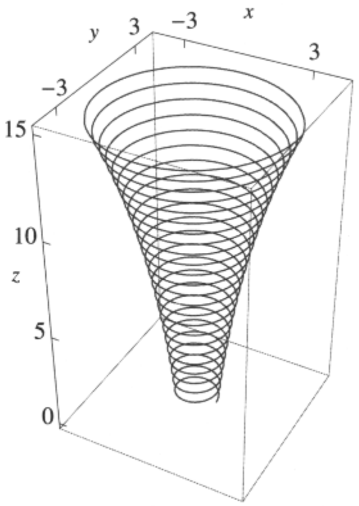

Figure 1.13: Plot of a helical spacecurve.

A *parametric surface* is given by $x = f(u,v)$, $y = g(u,v)$, $z = h(u,v)$ with two independent variables u and v. We first plot a torus (Fig. 1.14) followed by three interlocking mutually perpendicular tori (Fig. 1.15).

ParametricPlot3D[{(7 + 3 Sin[u]) Sin[v], (7 + 3 Sin[u]) Cos[v], 2 Cos[u]}, {u, 0, 2π}, {v, 0, 2π}, PlotStyle→Blue]

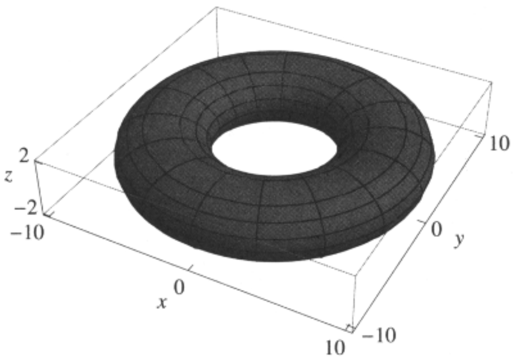

Figure 1.14: Plot of the torus, a parametric surface.

ParametricPlot3D[{{(7 + 3 Sin[u]) Sin[v], (7 + 3 Sin[u]) Cos[v], 2 Cos[u]}, {2 Cos[u], (7 + 3 Sin[u]) Sin[v], (7 + 3 Sin[u]) Cos[v]}, {(7 + 3 Sin[u]) Sin[v], 2 Cos[u], (7 + 3 Sin[u]) Cos[v]}}, {u, 0, 2π}, {v, 0, 2π}, PlotStyle→{Blue, Red, Green}]

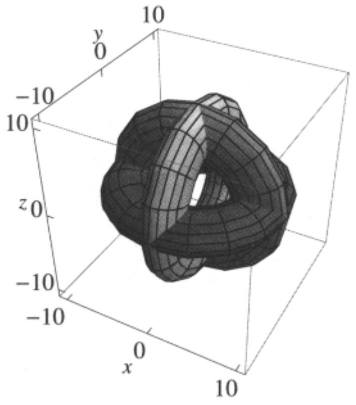

Figure 1.15: Three intersecting tori.

Next, we do implicit plotting of equations in the three variables x, y, and z, whose resulting plot gives a surface. We will start with a cylindrical surface, which is a surface whose equation has only two of the three space variables in it. This equation is really a curve in the plane, with certain radially symmetric properties, of its two variables, while adding the third variable (direction) results in a surface (see Figs. 1.16–1.18).

ContourPlot3D[$x^2 + z^2$ == 9, {x,−3,3}, {y,−3,3}, {z,−5,5}, ContourStyle→Blue, BoxRatios→Automatic, AxesLabel→Automatic]

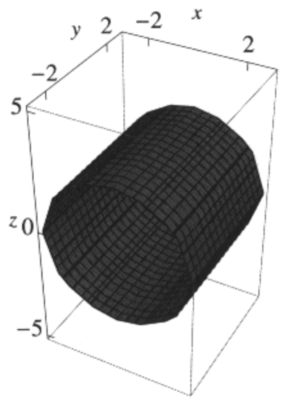

Figure 1.16: A cylinder defined implicitly.

ContourPlot3D[{x^2 + y^2 + z^2 == 16, x^2 + y^2 == 9}, {x, −5, 5}, {y, −5, 5}, {z, −7, 7}, ContourStyle→{Blue, Red}, BoxRatios→Automatic, AxesLabel→Automatic, Mesh→None]

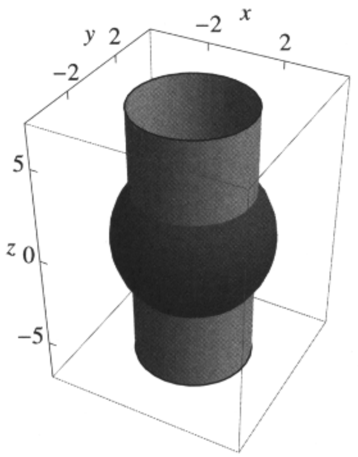

Figure 1.17: A cylinder intersecting a sphere.

Animate[ParametricPlot3D[{(C + A Sin[u]) Sin[v], (C + A Sin[u]) Cos[v], B Cos[u]}, {u, 0, 2π}, {v, 0, 2π}, PlotStyle→Blue, AxesLabel →{x, y, z}, PlotRange→{{−11, 11}, {−11, 11}, {−5, 5}}, PlotPoints→100, Mesh→None], {A, 1, 3}, {B, 1, 4}, {C, 5, 8}, AnimationRunning→False, ControlType→Slider, ControlPlacement →Right]

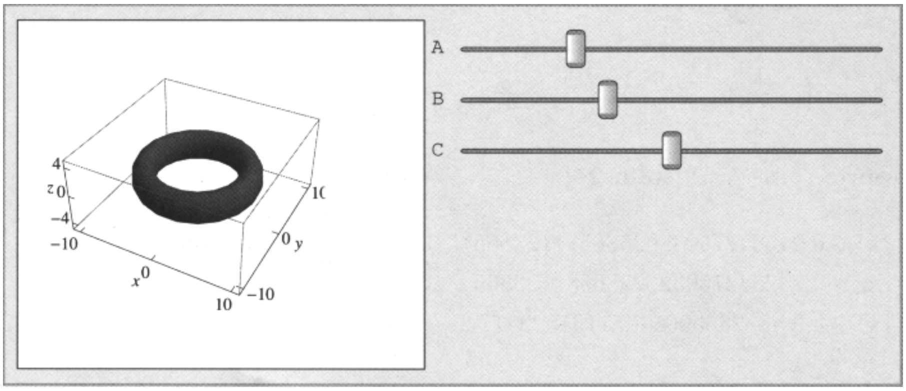

Figure 1.18: Animation of parameters in the graph of a torus. The parameter values for this frame are $A = 1.5$, $B = 2$, and $C = 6.5$.

A word of warning in doing 3D animations. This type of animation can use a great deal of memory, so we highly recommend that you delete the output from memory before closing the file unless it took a great deal of time to produce the animation and you do not want to have to do it again. We animated the torus with three animation parameters. Play around with each slider, corresponding to the three parameters, to see how they affect the shape of the torus.

1.6 Solving Systems of Equations

Mathematica can solve single equations as well as simultaneous systems of equations, both linear and nonlinear. It can also find approximate or exact solutions, although for exact solutions the equation or system must be capable of being solved for exact solutions by some known method. Also, *Mathematica* can find both real and complex solutions. The solution given by the command **Solve** will give a list of replacements for the variables solved for in your solution. In order to get the actual values of the solutions as a list and only for a specific variable solved for like z, you must use the input **z /. soln**, where *soln* is the result of **Solve** and z is one of the variables solved for in **Solve**. The command **Solve** seeks exact solutions while **NSolve** always gives approximate solutions.

We begin by solving some single polynomial equations for real and complex solutions, both approximate and exact. **Solve** will give all real and complex solutions to a single polynomial equation, that is, it will find all of the real and complex roots of any complex coefficient polynomial:

PolyN = 35 x^3 + 4;

soln = Solve[PolyN == 0, x]

$$\left\{\left\{x \to \left(-\frac{1}{35}\right)^{1/3} 2^{2/3}\right\}, \left\{x \to -\frac{(-2)^{2/3}}{35^{1/3}}\right\}, \left\{x \to -\frac{2^{2/3}}{35^{1/3}}\right\}\right\}$$

approxsolns = N[soln, 25]

```
{{x → 0.2426427503202586581123865 + 0.4202695716429374683285407 ⅈ},
 {x → 0.2426427503202586581123865 − 0.4202695716429374683285407 ⅈ},
 {x → −0.4852855006405173162247729} }
```

PolyN /. %

$$\left\{0.\times10^{-24}+0.\times10^{-24}\,i,\ 0.\times10^{-24}+0.\times10^{-24}\,i,\ 0.\times10^{-24}\right\}$$

```
xsolns = x /. approxsolns
```

```
{0.24264275032025865811238 65 + 0.42026957164293746832854 07 i,
 0.24264275032025865811238 65 - 0.42026957164293746832854 07 i,
 - 0.48528550064051731622477 29}
```

```
Q = 5 x^3 - 7 x^2 + I x - 4 I
```

$$-4\,i+i\,x-7\,x^2+5\,x^3$$

```
soln = Solve[Q == 0, x]
```

$$\Bigg\{\Bigg\{x\to\frac{7}{15}+\frac{\frac{49}{15}-i}{\left(\frac{1}{2}\left((686+2385i)+15\sqrt{-24693+16404i}\right)\right)^{1/3}}+\frac{1}{15}\left(\frac{1}{2}\left((686+2385i)+15\sqrt{-24693+16404i}\right)\right)^{1/3}\Bigg\},$$

$$\Bigg\{x\to\frac{7}{15}-\frac{\left(\frac{49}{15}-i\right)\left(1-i\sqrt{3}\right)}{2^{2/3}\left((686+2385i)+15\sqrt{-24693+16404i}\right)^{1/3}}-\frac{1}{30}\left(\frac{1}{2}\left((686+2385i)+15\sqrt{-24693+16404i}\right)\right)^{1/3}\left(1+i\sqrt{3}\right)\Bigg\},$$

$$\Bigg\{x\to\frac{7}{15}-\frac{1}{30}\left(\frac{1}{2}\left((686+2385i)+15\sqrt{-24693+16404i}\right)\right)^{1/3}\left(1-i\sqrt{3}\right)-\frac{\left(\frac{49}{15}-i\right)\left(1+i\sqrt{3}\right)}{2^{2/3}\left((686+2385i)+15\sqrt{-24693+16404i}\right)^{1/3}}\Bigg\}\Bigg\}$$

```
approxsolns = N[soln, 15]
```

```
{{x → 1.48095933119864 + 0.21092675658740 i},
 {x → 0.430101246109932 - 0.658941196856415 i},
 {x → -0.511060577308570 + 0.448014440269016 i} }
```

```
Q /. approxsolns
```

$$\left\{0.\times10^{-13}+0.\times10^{-14}\,i,\ 0.\times10^{-14}+0.\times10^{-14}\,i,\ 0.\times10^{-14}+0.\times10^{-14}\,i\right\}$$

```
xsolns = x /. approxsolns
```

```
{1.48095933119864 + 0.21092675658740 i,
 0.430101246109932 - 0.658941196856415 i,
 - 0.511060577308570 + 0.448014440269016 i}
```

Chop[Expand[5 (x − xsolns[[1]]) (x − xsolns[[2]]) (x − xsolns[[3]])]]

$-4.0000000000000\,i + 1.0000000000000\,i\,x - 7.0000000000000\,x^2 + 5\,x^3$

Now we look at some examples of solving systems of equations that are both linear and nonlinear. We begin by finding the intersection point of two lines in the xy-plane (Fig. 1.19) followed by finding the intersection points of two circles (Fig. 1.20):

Eqn1 = 5 x + 3 y == −4; Eqn2 = −7 x + 2 y == 6;
Soln = Solve[{Eqn1, Eqn2}, {x, y}]

$\left\{\left\{x \to -\frac{26}{31}, y \to \frac{2}{31}\right\}\right\}$

{Eqn1, Eqn2} /. Soln

{{True, True}}

approxsolns = N[Soln, 10]

$\left\{\left\{x \to -0.8387096774, y \to 0.06451612903\right\}\right\}$

{xsolns, ysolns} = {x, y} /. Flatten[approxsolns]

{−0.8387096774, 0.06451612903}

ContourPlot$\left[\left\{5\,x + 3\,y == -4,\ -7\,x + 2\,y == 6,\ (x - \text{xsolns})^2 + (y - \text{ysolns})^2 == \frac{1}{16}\right\}\right.$, {x, −5, 5}, {y, −5, 5}, ContourStyle→{Directive[Red, Thick], Directive[Blue, Thick], Directive[Black, Thick]}$\Big]$

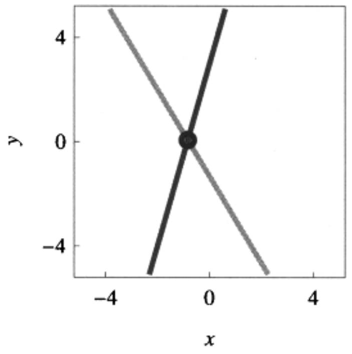

Figure 1.19: Solution to the system of two lines is a point.

{Eqn1, Eqn2} = {(x−7)^2+(y−2)^2 == 25, (x−1)^2+(y−5)^2 == 16};
Soln = Solve[{Eqn1, Eqn2}, {x, y}]

$$\left\{\left\{x \to \frac{1}{5}\left(17-2\sqrt{11}\right), y \to \frac{1}{5}\left(19-4\sqrt{11}\right)\right\}, \left\{x \to \frac{1}{5}\left(17+2\sqrt{11}\right), y \to \frac{1}{5}\left(19+4\sqrt{11}\right)\right\}\right\}$$

approxsolns = N[Soln]

```
{{x → 2.07335, y → 1.1467}, {x → 4.72665, y → 6.4533}}
```

solns = {x, y} /. approxsolns

```
{{2.07335, 1.1467}, {4.72665, 6.4533}}
```

{IntersPt1, IntersPt2} = {solns[[1]], solns[[2]]};
ContourPlot[{(x−7)^2+(y−2)^2 == 25, (x−1)^2+(y−5)^2 == 16, (x− IntersPt1[[1]])^2 + (y − IntersPt1[[2]])^2 == 1/8, (x − IntersPt2[[1]])^2 + (y − IntersPt2[[2]])^2 == 1/8}, {x,−5,15}, {y,−5,15}, ContourStyle→{ Directive[Red,Thick], Directive[Blue,Thick], Directive[Black,Thick], Directive[Black, Thick]}]

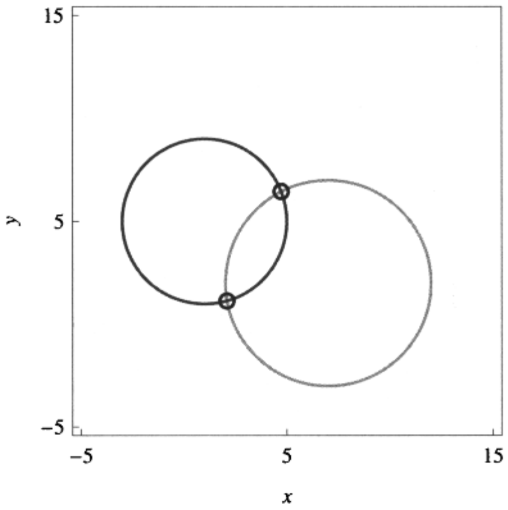

Figure 1.20: Intersection of two circles is a pair of points.

Eqns = $\{(\mathbf{x}-\mathbf{7})^2+(\mathbf{y}-\mathbf{2})^2==\mathbf{25},\ (\mathbf{x}+\mathbf{2})^2+(\mathbf{y}-\mathbf{5})^2==\mathbf{16}\}$;
Soln = Solve[Eqns, {x, y}]

$$\left\{\left\{x\to\frac{1}{20}\left(41-i\sqrt{89}\right),\ y\to\frac{1}{20}\left(73-3\,i\sqrt{89}\right)\right\},\ \left\{x\to\frac{1}{20}\left(41+i\sqrt{89}\right),\ y\to\frac{1}{20}\left(73+3\,i\sqrt{89}\right)\right\}\right\}$$

N[Soln]

$\{\{x\to 2.05-0.471699\,i,\ y\to 3.65-1.4151\,i\},$
$\{x\to 2.05+0.471699\,i,\ y\to 3.65+1.4151\,i\}\}$

1.7 Basic Programming

In this final section of Chapter 1, we will discuss some basic mathematical programming concepts such as the **For** loop command, the use of commands such as **While** and **Do**, and the general concept of a procedure in *Mathematica* that consists a series of commands with semicolons following each step of the procedure. A procedure's steps will be executed in order from left to right if they appear all on the same line or from first to last if they are on successive lines in the same cell. In general, by design, the output of a procedure is the result of its final step.

Let's begin by doing a few examples involving **For** loops such as printing values, doing sums and products, and using the commands **Do** and **If**. The first **For** loop will print z^k+z starting with $z=x^2$ as k goes from **1** to **4**, where at each step it takes the previous value of z and plugs it into z^k+z for the next value of k. The semicolons in the **For** loop separate the main parts of the procedure inside the **For** command, which are first the initialization of the loop variable k, the instructions to be carried out successively as k increases in implementing the loop, and finally the output that you desire at each step (value of k) of the loop.

For$\left[\mathbf{k}=\mathbf{1};\ \mathbf{z}=\mathbf{x}^2,\ \mathbf{k}\le\mathbf{4},\ \mathbf{k}++,\ \mathbf{z}=\mathbf{z}^{\mathbf{k}}+\mathbf{z};\ \mathbf{Print[z]}\right]$

$2x^2$

$2x^2+4x^4$

$2x^2+4x^4+\left(2x^2+4x^4\right)^3$

$2x^2+4x^4+\left(2x^2+4x^4\right)^3+\left(2x^2+4x^4+\left(2x^2+4x^4\right)^3\right)^4$

The second **For** loop will compute and display the sums of the reciprocals of the factorial function. The results get closer and closer to the number e which is approximately 2.71828.

For$\left[\mathbf{k = 1,\ k \leq 10,\ k{+}{+},}\right.$
  Print$\left[\mathbf{NSum}\left[\frac{\mathbf{1}}{\mathbf{n!}},\ \mathbf{\{n,\ 0,\ k\},\ WorkingPrecision \to 10}\right]\right]\Big]$

```
2.000000000
2.500000000
2.666666667
2.708333333
2.716666667
2.718055556
2.718253968
2.718278770
2.718281526
2.718281801
```

The next two procedures will do an addition of the first 10 positive integers and then a multiplication of the same integers using the **Do** loop command:

```
sum = 0;
Do[sum = sum + k, {k, 1, 10}]; sum
```

```
55
```

```
Sum[k, {k, 1, 10}]
```

```
55
```

```
prod = 1;
Do[prod = k prod, {k, 1, 10}]; prod
```

```
3 628 800
```

```
10!
```

```
3 628 800
```

```
prod = 1;
Do[prod = prod k, {k, 1, 10}]; prod
```

```
3 628 800
```

Sometimes the most convenient way to create a somewhat complicated function is to use a procedure for its rule. We will first look at two simple examples that create the factorial function and the add function using the procedures of the previous section to do it. Remember to place parentheses around all of the steps in your procedure. The output of the function will be the result of the last step of your procedure, which is the function's rule. The variable n in both the factorial and add functions below takes on only a positive integer value:

```
factorial[n_] := (prod = 1; Do[prod = k prod, {k, 1, n}]; prod)
factorial[10]
```

```
3628800
```

```
add[n_] := (sum = 0; Do[sum = sum + k, {k, 1, n}]; sum)
add[10]
```

```
55
```

Now we turn to a more sophisticated example involving piecewise functions and the use of the nested **If** command. *Mathematica* has a **Piecewise** command to build piecewise functions, but it is not always sufficient or convenient for all purposes. Let's create a function called *quadrant* that takes a point in the plane and tells us which quadrant it is in or if it is on an axis and thus is not in any quadrant. Note that **&&** is the *Mathematica* logical AND for joining two conditions so that both must be true for them together to be true. The logical OR is || with no space between the two vertical bars. The **Return** command used below will return an output as well as terminate the **If**, **While**, **For** or **Do** loop that you might be in. There is also a command **Break** which terminates loops.

Be very careful in *Mathematica* to avoid using the letters C, D, and N as variables in your functions or anywhere else as they are *Mathematica* command names. It is better to stick to using lowercase letters as your variable names and output for any function or procedure:

```
quadrant[pt_] := If[pt[[1]] > 0 && pt[[2]] > 0, Return[quadrant1],
  If[pt[[1]] < 0 && pt[[2]] > 0, Return[quadrant2],
    If[pt[[1]] < 0 && pt[[2]] < 0, Return[quadrant3],
       If[pt[[1]] > 0 && pt[[2]] < 0, Return[quadrant4], axis]]]]
```

```
quadrant[{-3, 9}]
```

```
quadrant2
```

```
quadrant[{0, 9}]
```

```
axis
```