CONTENTS

Introduction 5

Chapter 1  An Introduction to MATLAB 7

Numerical Calculations 7
Writing Scripts (m-files) 10
Defining Functions 12
Graphics 13
Symbolic Calculations 21
Differentiation and Integration 24
Solving Equations 26

Chapter 2  Kinematics of Particles 37

2.1 Sample Problem 2/4 (Rectilinear Motion) 38
2.2 Problem 2/87 (Rectangular Coordinates) 41
2.3 Problem 2/126 (n-t Coordinates) 46
2.4 Sample Problem 2/9 (Polar Coordinates) 48
2.5 Sample Problem 2/10 (Polar Coordinates) 52
2.6 Problem 2/183 (Space Curvilinear Motion) 55
2.7 Sample Problem 2/16 (Constrained Motion of Connected Particles) 57

Chapter 3  Kinetics of Particles 61

3.1 Sample Problem 3/3 (Rectilinear Motion) 62
3.2 Problem 3/98 (Curvilinear Motion) 65
3.3 Sample Problem 3/17 (Potential Energy) 67
3.4 Problem 3/218 (Linear Impulse/Momentum) 70
3.5 Problem 3/250 (Angular Impulse/Momentum) 72
3.6 Problem 3/365 (Curvilinear Motion) 73
INTRODUCTION

Computers and software have had a tremendous impact upon engineering education over the past several years and most engineering schools now incorporate computational software such as MATLAB in their curriculum. Since you have this supplement the chances are pretty good that you are already aware of this and will have to learn to use MATLAB as part of a Dynamics course. The purpose of this supplement is to help you do just that.

There seems to be some disagreement among engineering educators regarding how computers should be used in an engineering course such as Dynamics. I will use this as an opportunity to give my own philosophy along with a little advice. In trying to master the fundamentals of Dynamics there is no substitute for hard work. The old fashioned taking of pencil to paper, drawing free body and mass acceleration diagrams, struggling with equations of motion and kinematic relations, etc. is still essential to grasping the fundamentals of Dynamics. A sophisticated computational program is not going to help you to understand the fundamentals. For this reason, my advice is to use the computer only when required to do so. Most of your homework can and should be done without a computer.

The problems in this booklet are based upon problems taken from your text. The problems are slightly modified since most of the problems in your book do not require a computer for the reasons discussed in the last paragraph. One of the most important uses of the computer in studying Mechanics is the convenience and relative simplicity of conducting parametric studies. A parametric study seeks to understand the effect of one or more variables (parameters) upon a general solution. This is in contrast to a typical homework problem where you generally want to find one solution to a problem under some specified conditions. For example, in a typical homework problem you might be asked something about the trajectory of a particle launched at an angle of 30 degrees from the horizontal with an initial speed of 30 ft/sec. In a parametric study of the same problem you might typically find the trajectory as a function of two parameters, the launch angle $\theta$ and initial speed $v$. You might then be asked to plot the trajectory for different launch angles and speeds. A plot of this type is very beneficial in visualizing the general solution to a problem over a broad range of variables as opposed to a single case.
As you will see, it is not uncommon to find Mechanics problems that yield equations that cannot be solved exactly. These problems require a numerical approach that is greatly simplified by computational software such as MATLAB. Although numerical solutions are extremely easy to obtain in MATLAB this is still the method of last resort. Chapter 1 will illustrate several methods for obtaining symbolic (exact) solutions to problems. These methods should always be tried first. Only when these fail should you generate a numerical approximation.

Many students encounter some difficulties the first time they try to use a computer as an aid to solving a problem. In many cases they are expecting that they have to do something fundamentally different. It is very important to understand that there is no fundamental difference in the way that you would formulate computer problems as opposed to a regular homework problem. Each problem in this booklet has a problem formulation section prior to the solution. As you work through the problems be sure to note that there is nothing peculiar about the way the problems are formulated. You will see free-body and mass acceleration diagrams, kinematic equations etc. just like you would normally write. The main difference is that most of the problems will be parametric studies as discussed above. In a parametric study you will have at least one and possibly more parameters or variables that are left undefined during the formulation. For example, you might have a general angle $\theta$ as opposed to a specific angle of 20°. If it helps, you can “pretend” that the variable is some specific number while you are formulating a problem.

This supplement has eight chapters. The first chapter contains a brief introduction to MATLAB. If you already have some familiarity with MATLAB you can skip this chapter. Although the first chapter is relatively brief it does introduce all the methods that will be used later in the book and assumes no prior knowledge of MATLAB. Chapters 2 through 8 contain computer problems taken from chapters 2 through 8 of your textbook. Thus, if you would like to see some computer problems involving the kinetics of particles you can look at the problems in chapter 3 of this supplement. Each chapter will have a short introduction that summarizes the types of problems and computational methods used. This would be the ideal place to look if you are interested in finding examples of how to use specific functions, operations etc.

This supplement uses the student edition of MATLAB version 7.1. MATLAB is a registered trademark of The Mathworks, Inc., 24 Prime Park Way, Natick, Massachusetts, 01760.
AN INTRODUCTION TO MATLAB

This chapter provides an introduction to the MATLAB programming language. Although the chapter is introductory in nature it will cover everything needed to solve the computer problems in this booklet.

1.1 Numerical Calculations

When you open MATLAB you should see a prompt something like the following.

EDU>

To the right of this prompt you will write some sort of command. The following example assigns a value of 200 to the variable a. Simply type “a = 200” at the prompt and then press enter.

EDU> a=200
a =
200

There are many situations where you may not want MATLAB to print the result of a command. To suppress the output simply type a semi-colon “;” after the command. For example, to suppress the output in the above case type “a = 200;” and the press enter. The basic mathematical operations of addition, subtraction, multiplication, division and raising to a power are accomplished exactly as you would expect if you are familiar with other programming languages, i.e. with the keys +, -, *, \, and ^. Here are a couple of examples.

EDU> c=2*5-120/4
  c =
-20
MATLAB has many built in functions that you can use in calculations. If you already know the name of the function you can simply type it in. If not, you can find a list of functions in the “Help Desk (HTML)” file available under Help in the main menu bar. Here are a few examples.

EDU» e=sin(.5)+tan(.2)
e =
0.6821

As with most mathematical software packages, the default unit for angles will be radians.

EDU» f=sqrt(16)
f =
4

EDU» g=log(5)
g =
1.6094

EDU» h=log10(5)
h =
0.6990

In the last two examples be careful to note that log is the natural logarithm. For a base 10 logarithm (commonly used in engineering) you need to use log10.

**Range Variables**

In the above examples we have seen many cases where a variable (or name) has been assigned a numerical value. There are many instances where we would like a single variable to take on a range of values rather than just a single value. Variables of this type are often referred to as range variables. For example, to have the variable x take on values between 0 and 3 with an increment of 0.25 we would type “x=0:0.25:3”.

EDU» x=0:0.25:3
x =
Columns 1 through 7
 0 0.2500 0.5000 0.7500 1.0000 1.2500 1.5000
Columns 8 through 13
Range variables are very convenient in that they allow us to evaluate an expression over a range with a single command. This is essentially equivalent to performing a loop without actually having to set up a loop structure. Here's an example.

```
EDU» x=0:0.25:3;
EDU» f=3*sin(x)-2*cos(x)
f =
Columns 1 through 7
    -2.0000  -1.1956  -0.3169    0.5815    1.4438    2.2163    2.8510
Columns 8 through 13
    3.3084    3.5602    3.5906    3.3977    2.9936    2.4033
```

Although range variables are very convenient and almost indispensable when making plots, they can also lead to considerable confusion for those new to MATLAB. To illustrate, suppose we wanted to compute a value \( y = 2x - 3x^2 \) over a range of values of \( x \).

```
EDU» x=0:0.5:3;
EDU» y = 2*x-3*x^2
??? Error using ==> mpower
Matrix must be square.
```

What went wrong? We have referred to \( x \) as a range variable because this term is most descriptive of how we will actually use variables of this type in this manual. As the name implies, MATLAB is a program for doing matrix calculations. Thus, internally, \( x \) is really a row matrix and operations such as multiplication or division are taken, by default, to be matrix operations. Thus, MATLAB takes the operation \( x^2 \) to be matrix multiplication \( x \times x \). Since matrix multiplication of two row matrices doesn’t make sense, we get an error message. In a certain sense, getting this error message is very fortunate since we are not actually interested in a matrix calculation here. What we want is to calculate a certain value \( y \) for each \( x \) in the range specified. This type of operation is referred to as term-by-term. For term-by-term operations we need to place a period “.” in front of the operator. Thus, for term by term multiplication, division, and raising to a power we would write “.*”, “./” and “.^” respectively. Since matrix and term-by-term addition or subtraction are equivalent, you do not need a period before + or -. The reason that our earlier calculations did not produce an error is that they all involved scalar quantities. To correct the error above we write.

```
EDU» x=0:0.5:3;
EDU» y = 2*x-3*x.^2
y =
```
Note that we wrote “2*x” instead of “2.*x” since 2 is a scalar. If in doubt, it
never hurts to include the period.

1.2 Writing Scripts (m-files)

If you have been working through the examples above you have probably noticed
by now that you cannot modify a line once it has been entered into the worksheet.
This makes it rather difficult to debug a program or run it again with a different
set of parameters. For this reason, most of your work in MATLAB will involve
writing short scripts or m-files. The term m-file derives from the fact that the
programs are saved with a “.m” extension. An m-file is essentially just a list of
commands that you would normally enter at a prompt in the worksheet. After
saving the file you just enter the name of the file at the prompt after which
MATLAB will execute all the lines in the file as if they had been entered. The
chief advantage is that you can alter the script and run it as many times as you
like.

To get started, select “File...New...M-File” from the main menu. A screen like
that below will then pop up.
Now enter the program line by line into the editor. Basically, you just enter the commands you would normally enter at a prompt (do not, however, enter >> at the beginning of a line). If you want to enter comments, begin the line with “%”. These lines will not be executed. Once you have finished entering the script into the editor you can save the file (be sure to remember where you saved it). The file name can contain numbers but cannot begin with a number. It also cannot contain any mathematical operators such as +, -, *, / etc. Also, the name of the file should not be the same as a variable it computes.

To run the script all you have to do is type the name of the program at a prompt (>>), but without the “.m” extension. A common error at this point is that MATLAB will give some response indicating that it hasn’t a clue what you are doing. The reason this happens is that the file you saved is not in the current path. You can check (or change) the path by selecting File…Set Path… from the main menu.

Following is a simple example of a script file that sets up a range variable x and then calculates two functions over the range.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script calculates two
% functions over a specified range
x=0:0.2:1;
f=sin(x)./(1+x)
g=sin(x).*exp(x)
%%%%%%%%%%%%%%%% end of script %%%%%%%%%%%%%%%%%

The program was saved as example1, so here is the output you should see when the program is entered at a prompt.

EDU» example1
f =
   0  0.1656  0.2782  0.3529  0.3985  0.4207
g =
   0  0.2427  0.5809  1.0288  1.5965  2.2874

To save space we will, in the future, show the output of a script immediately after the script is given without actually showing the file name being entered at a prompt.
1.3 Defining Functions

Although MATLAB has many built in functions (see the “Help Desk (HTML)” file available under Help in the main menu bar), it is sometimes advantageous to define your own functions. While user defined functions are actually special types of m-files (they will be saved with the .m extension) their use is really quite different from the m-files discussed above. For example, suppose you were to create a function that you call f1. You would then use this function as you would any built in function like sin, cos, log etc.

Since functions are really special types of m-files you start by opening an editor window with “File…New…M-File”. The structure of the file is, however, quite different from a script file. Following is an m-file for a function called load1. Note that you should save the file under the same name as the function. The file would thus show up as load1.m in whatever directory you save it. As with script files, you need to be sure the current path contains the directory in which the file was stored.

```matlab
%%%%%% a function m-file %%%%%%%%%%%%%%%%
function y = load1(x)
% this file defines the function load1(x)
y = 2*(x-1)+3*(x-2).^2;
%%%%%% end of function m-file %%%%%%%%%%%
```

It is important to understand the structure of a function definition before you try to create some on your own. The name of this function is “load1” and will be used as you would any other function. y and x are local variables which MATLAB uses to calculate the function. There is nothing special about y and x, other variable names will work as well. Now, suppose you were to type load1(3) at a prompt or within a script file. MATLAB will calculate the expression for y substituting x = 3. This value of y will then be returned as the value for load1(3).

Here are a few examples using our function load1.

EDU» load1(3)
ans =
   7

EDU» f=load1(5)-load1(3)
f =
   28
EDU» load1(load1(3))
ans =
     87

EDU» x=1:0.2:2;
EDU» g=load1(x)
g =
     3.0000   2.3200   1.8800   1.6800   1.7200   2.0000

1.4 Graphics

One of the most useful things about a computational software package such as MATLAB is the ability to easily create graphs of functions. As we will see, these graphs allow one to gain a lot of insight into a problem by observing how a solution changes as some parameter (the magnitude of a load, an angle, a dimension etc.) is varied. This is so important that practically every problem in this supplement will contain at least one plot. By the time you have finished reading this supplement you should be very proficient at plotting in MATLAB. This section will introduce you to the basics of plotting in MATLAB.

MATLAB has the capability of creating a number of different types of graphs. Here we will consider only the X-Y plot. The most common and easiest way to generate a plot of a function is to use range variables. The following example will guide you through the basic procedure.

EDU» x=-3:0.1:3;
EDU» f=x.*exp(-x.^2);
EDU» plot(x,f)

The procedure is very simple. First define a range variable covering the range of the plot, then define the function to be plotted and issue the plot command. After typing in the above you should see a graph window pop up which looks something like the following figure.
Things like titles and axis labels can be added in one of two ways. The first is by line commands such as those shown below.

EDU» xlabel('x')
EDU» ylabel('x*exp(-x^2)')
EDU» title('A Simple Plot')

Take a look at the graph window after you type each of the lines above. At this point, the graph window should look something like that shown below.
There are many other things that you can add or change on the plot. The easiest way to do this is with the second of the two approaches mentioned above. In the graph menu select “Tools...Edit Plot”. Now you can edit the plot in a number of different ways. To see some of the options try either double clicking or right clicking on the graph. You should spend some time experimenting with a graph window to see the range of possibilities. In particular, be sure to try clicking the show plot tools button.

Once you have things like you want them you can save the graph to a file. You can also export the graph to some format suitable for inserting in, say, a word processor. Select “File...Export Setup” in the graph window to see the range of possible formats.
**Parametric Studies**

One of the most important uses of the computer in studying Mechanics is the convenience and relative simplicity of conducting parametric studies (not to be confused with parametric plotting discussed below). A parametric study seeks to understand the effect of one or more variables (parameters) upon a general solution. This is in contrast to a typical homework problem where you generally want to find one solution to a problem under some specified conditions. For example, in a typical homework problem you might be asked to find the reactions at the supports of a structure with a concentrated force of magnitude 200 lb that is oriented at an angle of 30 degrees from the horizontal. In a parametric study of the same problem you might typically find the reactions as a function of two parameters, the magnitude of the force and its orientation. You might then be asked to plot the reactions as a function of the magnitude of the force for several different orientations. A plot of this type is very beneficial in visualizing the general solution to a problem over a broad range of variables as opposed to a single case.

Parametric studies generally require making multiple plots of the same function with different values of a particular parameter in the function. As a simple example, consider the following function.

\[ f = 5 + x - 5x^2 + ax^3 \]

What we would like to do is gain some understanding of how \( f \) varies with both \( x \) and \( a \). It might be tempting to make a three dimensional plot in a case like this. Such a plot can, in some cases, be very useful. Usually, however, it is too difficult to interpret. This is illustrated by the following three dimensional plot of \( f \) versus \( x \) and \( a \).

```matlab
% This script produces a 3-d plot of f versus x and a.
x=linspace(-10,10,30);
a=linspace(-3,3,30);
[X,A]=meshgrid(x,a);
F=5+X-5*X.^2+A.*X.^3;
B=0.*A+2;
mesh(X,A,F)
colormap gray
%%%%%% end of script %%%%%%%%%%%%%%%%%%%
```
The plot above certainly is interesting but, as mentioned above, not very easy to interpret. In most cases it is much better to plot the function several times (with different values of the parameter of interest) on a single two-dimensional graph. We will illustrate this by plotting $f$ as a function of $x$ for $a = -1$, 0, and 1. This is accomplished in the following script.

```matlab
% script for plotting $f$ versus $x$ for several
% values of $a$.
x=-5:0.1:5;
g=5+x-5*x.^2;
f1=g-x.^3;  % $a = -1$
f2=g;       % $a = 0$
f3=g+x.^3;  % $a = 1$
plot(x,f1,x,f2,x,f3)
xlabel('x')
ylabel('f')
%%%% end of script %%%%%%%%%%%%%%%%%%%%%%
```
Parametric Plots

It often happens that one needs to plot some function $y$ versus $x$ but $y$ is not known explicitly as a function of $x$. For example, suppose you know the $x$ and $y$ coordinates of a particle as a function of time but want to plot the trajectory of the particle, i.e., you want to plot the $y$ coordinate of the particle versus the $x$ coordinate. A plot of this type is generally called a parametric plot. Parametric plots are easy to obtain in MATLAB. You start by defining the two functions in terms of the common parameter and then define the common parameter as a range variable. Then issue a plot command using the two functions as arguments. The following script illustrates this procedure.
% This script file illustrates parametric plotting.
% A function f is plotted versus another function g.
% The two functions are related by the
% common parameter a.
a=-1:0.05:3.5;
f=10*a.*(2-a);
g=sin(3*a);
plot(g,f)
xlabel('g')
ylabel('f')

% end of script

The range selected for the parameter can have a big, and sometimes surprising
effect on the resulting graph. To illustrate, try increasing the upper limit on the
range on a a few times and see how the graph changes.

You can, of course, also plot g as a function of f.
% Another example of parametric plotting
a=-1:0.05:6;
f=10*a.*(2-a);
g=sin(3*a);
plot(f,g)
xlabel('f')
ylabel('g')
%%% end of script ........................
1.5 Symbolic Calculations

One of the most useful features of many modern mathematical software packages is the capability of doing symbolic math. By symbolic math we mean mathematical operations on expressions containing undefined variables rather than numbers. Maple is, perhaps, the best known computer algebra program. Many other programs (including MATLAB) have a Maple engine that is called whenever symbolic operations are performed. The Maple engine is part of the Symbolic Math Toolbox so, if you do not have this toolbox, you will not be able to do symbolic math. If you have a student edition of MATLAB you should have the Symbolic Math Toolbox. If you do not have the Symbolic Math Toolbox, you will need to perform whatever symbolic calculations are done in this manual by hand.

How does MATLAB decide whether to call Maple and attempt a symbolic calculation? Perhaps the best way to learn this is by studying the examples below and by practicing. Maple will always be used to manipulate any expression represented as a character string (i.e. an expression enclosed in ‘quotation’ marks). In the remainder of this manual we will refer to character strings of this type as Maple expressions. Here are a few examples of Maple expressions.

```
EDU» 'x+a*x^2-b*x^3'
ans =
x+a*x^2-b*x^3
EDU» f='A*sin(x)*exp(-x^2)'
f =
A*sin(x)*exp(-x^2)
```

Instead of placing the entire expression in quotations, you can also first declare certain variables to be symbols and then write the expressions as usual. This is more convenient in some situations but can get confusing. Here are some examples of this type.

```
EDU» syms x a b A % Sets x, a, b and A as symbols
EDU» g=x+a*x^2-b*x^3
g =
x+a*x^2-b*x^3
EDU» f=A*sin(x)*exp(-x^2)
f =
A*sin(x)*exp(-x^2)
```

The above are just examples of Maple expressions and do not involve any symbolic math. Two of the most important applications of symbolic math will be discussed in the next two sections, namely symbolic calculus (integration and...
differentiation) and symbolic solution of one or more equations. The purpose of
the present section is to introduce you to the basic procedures of symbolic math
as well as to give a few other useful applications. One particularly useful
symbolic operation is substitution and is illustrated by the following examples.

EDU» g='x+a*x^2-b*x^3'

   g =
   x+a*x^2-b*x^3

EDU» subs(g,'a',3)   % substitutes 3 for a in g. Note the quotation marks about a.

   ans =
   x+3*x^2-b*x^3

EDU» g1=subs(g,'x','(y-c)')

   g1 =
   ((y-c))+a*((y-c))^2-b*((y-c))^3

EDU» syms x y A B

EDU» f=A*sin(x)*exp(-B*x^2)

   f =
   A*sin(x)*exp(-B*x^2)

EDU» subs(f,B,4)   % Note that B doesn't have to be in quotes since it
                    was declared to be a symbol.

   ans =
   A*sin(x)*exp(-4*x^2)

Another very convenient feature available in the Symbolic Toolbox provides the
capability of defining “inline” functions thus avoiding the necessity for creating
special m-files as described above. Here is an example.

EDU» f=inline('x^2*exp(-x)')

   f =
        Inline function: 
        f(x) = x^2*exp(-x)

EDU» f(3)

   ans =
        0.4481

In the previous section we plotted the function \( f = 5 + x - 5x^2 + ax^3 \) for several
values of a. It turns out to be somewhat more convenient to do this with an inline
function. Let’s see how it works.
EDU» f=inline('5+x-5*x^2+a*x^3')
f =
    Inline function:
    f(a,x) = 5+x-5*x^2+a*x^3

At this point we might anticipate a minor problem. When we plot this as a function of x we will have a range variable as one of the arguments for f. This creates a problem as MATLAB will be performing the numerical calculations and it will take the operations in f as matrix operations rather than term by term. Fortunately we can use the “vectorize” operator to insert periods in the appropriate places.

EDU» f=vectorize(f)
f =
    Inline function:
    f(a,x) = 5+x-5.*x.^2+a.*x.^3

EDU» x=-5:0.1:5;
EDU» plot(x,f(-1,x),x,f(0,x),x,f(1,x))
1.6 Differentiation and Integration

Here we will consider only symbolic differentiation and integration since these are most useful in the mechanics applications in this manual. For differentiation we will use the Maple \texttt{diff} command. Care should be exercised here since \texttt{diff} is also a MATLAB command. To guarantee that you are using the Maple command, be sure that you use \texttt{diff} only on Maple expressions.

\begin{verbatim}
EDU» f='a*sec(b*t)';
EDU» diff(f, 't')        % Note that t must be placed in quotation marks since
                          % it is a symbol.
ans =
a*sec(b*t)*tan(b*t)*b
\end{verbatim}

We can avoid using the quotation marks by making a, b and t symbols.

\begin{verbatim}
EDU» syms a b t
EDU» f=a*sec(b*t);
EDU» diff(f, t)
ans =
a*sec(b*t)*tan(b*t)*b
\end{verbatim}

To find higher order derivatives use the command \texttt{diff(f, x, n)} where \texttt{n} is the order of the derivative. For example, to find the third derivative of \(a \log(b+x)\) we would write

\begin{verbatim}
EDU» diff('a*log(b+x)', 'x', 3)
ans =
2*a/(b+x)^3
\end{verbatim}

Symbolic integration will be performed with the \texttt{int} command. The general format of this command is \texttt{int(f, x, a, b)} where \texttt{f} is the integrand (a Maple expression), \texttt{x} is the integration variable and \texttt{a} and \texttt{b} are the limits of integration. If \texttt{x}, \texttt{a} and \texttt{b} have not been declared symbols with the \texttt{syms} command, they must be placed in quotation marks. If the integration limits are omitted, the indefinite integral will be evaluated. Here are several examples of definite and indefinite integrals.

\begin{verbatim}
EDU» int('sin(b*x)','x')
ans =
-1/b*cos(b*x)
\end{verbatim}
EDU» \text{int('sin(b*x)','x','c','d')} \\
\text{ans} = \\
-(\cos(b*d)-\cos(b*c))/b

EDU» \text{syms } x \ b \ c \ d \\
EDU» \text{f=b*log(x);} \\
EDU» \text{int(f,x)} \\
\text{ans} = \\
b*x*log(x)-b*x

EDU» \text{int(f,x,c,d)} \\
\text{ans} = \\
b*d*log(d)-b*d-b*c*log(c)+b*c

If a definite integral contains no unknown parameters either in the integrand or the integration limits, the \text{int} command will provide numerical answers. Here are a few examples.

EDU» \text{int('x+3*x^3','x',0,3)} \\
\text{ans} = \\
261/4

EDU» \text{int('log(x)','x',2,5)} \% \text{Don’t forget that log is the natural logarithm.} \\
\text{ans} = \\
5*log(5)-3-2*log(2)

Note that Maple will always try to return an exact answer. This usually results in answers containing fractions or functions as in the above examples. This is very useful in some situations; however, one often wants to know the numerical answer without having to evaluate a result such as the above with a calculator. To obtain numerical answers use Maple’s \text{eval} function as in the following examples.

EDU» \text{eval(int('x+3*x^3','x',0,3))} \\
\text{ans} = \\
65.2500

EDU» \text{eval(int('log(x)','x',2,5))} \\
\text{ans} = \\
3.6609
1.7 Solving Equations

Our preferred approach to solving equations is to use Maple’s solve command available in the Symbolic Toolbox. We will also discuss briefly MATLAB’s built in function fzero which can be used to solve single equations numerically.

Solving single equations

Here are a couple of examples illustrating how to use Maple’s solve command to solve a single equation symbolically. The first example is the (hopefully) familiar quadratic equation.

EDU» f='a*x^2+b*x+c=0';
EDU» solve(f, 'x')
ans =
   1/2/a*(-b+(b^2-4*a*c)^(1/2))
   1/2/a*(-b-(b^2-4*a*c)^(1/2))

EDU» g='a*sin(x)-cos(x)=1';
EDU» solve(g, 'x')
ans =
   pi
   atan2(2*a/(1+a^2),(a^2-1)/(1+a^2))

Note that both examples produced two solutions with each solution being placed on separate lines. One of the solutions to the second example is the number \(\pi\). The second solution illustrates something you should always watch out for. In many cases you may get a solution in terms of a function that you are not familiar with, i.e. atan2 that appears in the second solution to the second example above. Whenever this happens you should type “help atan2” at a prompt in order to learn more about the function.

If the variable being solved for is the only unknown in the equation, solve will return a number as the result. Here are a couple of examples.

EDU» f='2*x^2+4*x-12';
EDU» solve(f, 'x')
ans =
   -1+7^(1/2)
   -1-7^(1/2)

EDU» g='5*sin(x)-cos(x)=1';
EDU» solve(g, 'x')
ans =


If you want the answer as a floating point number instead of an exact result, use the Maple command \textit{eval}. For example, with \( g \) defined as above we would write

\[
\text{EDU» eval(solve(g, 'x'))}
\]

\[
\text{ans =}
\begin{align*}
3.1416 \\
0.3948
\end{align*}
\]

Although Maple will always attempt to return an exact (symbolic) result, you may occasionally have a case where a floating point number is returned without using \textit{eval}. Here is an example.

\[
\text{EDU» f='10*sin(z)=2*z+1'};
\]

\[
\text{EDU» solve(f, 'z')}
\]

\[
\text{ans =}
\begin{align*}
.12541060097666150276238831440340
\end{align*}
\]

Whenever this occurs, Maple was unable to solve the equation exactly and switched automatically to a numerical solution. In a certain sense this is very convenient as the same command is being used for both symbolic and numerical calculations. The problem is that nonlinear equations quite often have more than one solution, as illustrated by the examples above. If you have the full version of Maple then you will have various options available for finding multiple numerical solutions. But since we are dealing now with numerical rather than symbolic solutions it makes more sense just to use MATLAB.

The appropriate MATLAB command for the numerical solution of an equation is \texttt{fzero}. This command numerically determines the point at which some defined function has a value of zero. This means that we need to rewrite our equation in terms of an expression whose zeros (roots) are solutions to the original expression. This is easily accomplished by rearranging the original equation into the form \textit{expression} = 0. We will call this expression \( g \) in this example. Rewriting the above equation \( f \) we have \( g = 2z + 1 - 10\sin(z) \). The general format for \texttt{fzero} is \texttt{fzero(‘function_name’, \( x_0 \))} where ‘function_name’ is the name of the m-file function that you have created and \( x_0 \) is an initial guess at the root (zero) of the function. The initial guess is very important since, if there is more than one solution, MATLAB will find the solution closest to the initial guess. Well, you may be wondering how we can determine an approximate solution if we do not yet know the solution. Actually, this is very easy to do. First we define a function \( g(z) \) whose roots will be the solution to the equation of interest. Next, we plot this function in order to estimate the location of points where \( g(z) = 0 \). Here’s how it works for the present example.
%% function m-file 
function y = gz(z)
    y = 2*z + 1 - 10*sin(z);
%% end of m-file 

Above is our user defined function (m-file) which we have named gz. Now we can plot this function over some range as illustrated below.

EDU» z=-1:0.01:3;
EDU» plot(z,gz(z),z,0) % we also plot a line at g=0 to help find the roots.
EDU» xlabel('z')
EDU» ylabel('g(z)')

The plot shows that the function is zero at about z = 0 and 2. The first root is clearly that found by Maple’s `solve` command above. Now we can easily find both roots with the `fzero` command.

EDU» fzero('gz', 0)
ans =
    0.1254

EDU» fzero('gz', 2)
ans =
    2.4985
Thus, the two solutions are $z = 0.1254$ and $2.4985$.

**Finding Maxima and Minima of Functions**

Here we will illustrate two methods for finding maxima and minima of a function.

*Method 1. Using the symbolic `diff` and `solve` functions.*

The usual method for finding maxima or minima of a function $f(x)$ is to first determine the location(s) $x$ at which maxima or minima occur by solving the equation $\frac{df}{dx} = 0$ for $x$. One then substitutes the value(s) of $x$ thus determined into $f(x)$ to find the maximum or minimum. Consider finding the maximum and minimum values of the following function:

$$f = 1 + 2x - x^3$$

Before proceeding, it is a good idea to make a plot of the function $f$. This will give us a rough idea where the minima and maxima are as a check on the results we obtain below. The following script will plot $f$ as a function of $x$.

```matlab
% This script plots f as a function of x
x = -2:0.05:2;
f = 1 + 2*x - x.^3;
plot(x, f)
xlabel('x')
ylabel('f')

%%%%%%%%%%%%%%%%% end of script %%%%%%%%%%%%%%%%%
```
This plot shows we have a minimum of about 0 at x of about −0.9 and a maximum of about 2 at x of about 0.9. Now we can proceed to find the results more precisely.

EDU» f = '1+2*x-x^3'

f = 
1+2*x-x^3

EDU» dfdx = diff(f,'x')
dfdx = 
2-3*x^2

EDU» solve(dfdx,'x')
ans = 
1/3*6^(1/2)
-1/3*6^(1/2)

If an equals sign is omitted (as above) solve will find the root of the supplied Maple expression. In other words, it will solve the equation dfdx = 0. It is more convenient in the present case to have the solutions as floating point numbers. Thus, we should have used eval as in the following.
EDU» eval(solve(dfdx,'x'))
ans =
 0.8165
-0.8165

The above results are the locations (x) where the minima and maxima occur. To find the maximum and minimum values of the function f we use subs to substitute these results back into f.

EDU» subs(f,'x',0.8165)
ans =
 2.0887

EDU» subs(f,'x',-0.8165)
ans =
-0.0887

Thus, \( f_{\text{max}} = 2.0887 \) at \( x = 0.8165 \) and \( f_{\text{min}} = -0.0887 \) at \( x = -0.8165 \).

**Method 2. Using MATLAB’s min and max functions.**

EDU» x = -2:0.05:2;
EDU» f = 1 + 2*x - x.^3;

These two lines are identical to those in the script used to plot f above. If you have just run that script you do not need to execute these lines. Now we can find the minimum and maximum values of f by typing \text{min}(f)\ and \text{max}(f).

EDU» min(f)
ans = -3

EDU» max(f)
ans = 5

These are clearly not the answers we obtained above. What went wrong? It is important to understand that \text{min} and \text{max} do not find true minima and maxima in the mathematical sense, i.e. they do not find locations where df/dx = 0. Instead, they find the minimum or maximum value of all those calculated in the vector f. To see this, go back and look at the plot of f above and you will see that the minimum and maximum values of f are indeed –3 and 5. This is another good reason to plot the function first. To avoid the above difficulty we can change the range of x to insure we pick up the true minima and maxima.

EDU» x = -1:0.05:1;
In most cases, we want not only a minimum or maximum but also the location where they occur. This being the case we should use the following command.

EDU» [f_min, i] = min(f)

f_min = -0.0880
i = 5

The command [f_min, i] = min(f) finds the minimum value of f and assigns it to the variable f_min. The index of this value is assigned to the variable i. Now we can find the location of the minimum by printing the 5th value of the vector x.

EDU» x_min = x(i)

x_min = -0.8000

Now we can repeat the above for the maximum.

EDU» [f_max, i] = max(f)

f_max = 2.0880
i = 37

EDU» x_max = x(i)

x_max = 0.8000

You may have noticed at this point that the answers are somewhat different from those found by the first method above. The reason is, once again, that min and max do not find true minima and maxima. To obtain more accurate results you can use a closer spacing for the range variable (x). To see this, repeat the above for a spacing of 0.01 instead of 0.05.
Solving several equations simultaneously

EDU» eqn1='x^2+y^2=12'

eqn1 =
x^2+y^2=12

EDU» eqn2='x*y=4'

eqn2 =
x*y=4

In the above we have defined two equations which we will now solve for the two unknowns, x and y.

EDU» [x,y] = solve(eqn1,eqn2)

x =
 5^(1/2)-1
-1-5^(1/2)
 5^(1/2)+1
 1-5^(1/2)

y =
 5^(1/2)+1
 1-5^(1/2)
 5^(1/2)-1
-1-5^(1/2)

Note that four solutions have been found, the first being $x = \sqrt{5} - 1$, $y = \sqrt{5} + 1$. The following example illustrates a case where there are more unknowns than there are equations. In this case, the specified variables will be solved for in terms of the others.

EDU» f='x^2 + a*y^2 = 0'

f =
x^2 + a*y^2 = 0

EDU» g='x-y = b'

g =
x-y = b
In this case, there are two solutions. Now let’s consider an example with three equations.

EDU» eqn1='x^2+y^2=12'

```
eqn1 =
x^2+y^2=12
```

EDU» eqn2='x*y=4'

```
eqn2 =
x*y=4
```

EDU» eqn3='x-y=z'

```
eqn3 =
x-y=z
```

EDU» [x,y,z] = solve(eqn1,eqn2,eqn3)

```
x =
5^(1/2)+1
1-5^(1/2)
5^(1/2)-1
-1-5^(1/2)
```
y =
    5^(1/2)-1
    -1-5^(1/2)
    5^(1/2)+1
    1-5^(1/2)

z =
    2
    2
    -2
    -2
Kinematics involves the study of the motion of bodies irrespective of the forces that may produce that motion. MATLAB can be very useful in solving particle kinematics problems. Problem 2.1 is a rectilinear motion problem illustrating integration with the \texttt{int} command. The formulation of this problem results in an equation that cannot be solved exactly except with some rather sophisticated mathematics. When this occurs it is generally easiest to obtain either a graphical or numerical solution. This problem illustrates both approaches. Problem 2.2 is a rectangular coordinates problem that illustrates \texttt{diff} and \texttt{solve}. Problem 2.3 is a relatively straightforward problem where MATLAB is used to generate a plot that might be useful in a parametric study. The $r$-$\theta$ components of velocity and acceleration are plotted in problem 2.4 as well as the trajectory of a particle. In problem 2.5, the $r$-$\theta$ components of the velocity are determined using symbolic differentiation (\texttt{diff}). The problem also illustrates how computer algebra can simplify what might normally be a rather tedious algebra problem. The \texttt{diff} command is further illustrated in problems 2.6 and 2.7. Problem 2.7 is particularly interesting in that it requires differentiation with respect to time of a function whose explicit time dependence is unknown. This happens rather frequently in Dynamics so it is useful to know how to accomplish this with MATLAB.
2.1 Sample Problem 2/4 (Rectilinear Motion)

A freighter is moving at a speed of 8 knots when its engines are suddenly stopped. From this time forward, the deceleration of the ship is proportional to the square of its speed, so that \( a = -kv^2 \). The sample problem in your text shows that it is rather easy to determine the constant \( k \) by measuring the speed of the boat at some specified time. Show how \( k \) could be found by (a) measuring the speed after some specified distance and (b) measuring the time required to travel some specified distance. In both cases let the initial speed be \( v_0 \).

**Problem Formulation**

(a) Since time is not involved, the easiest approach is to integrate the equation \( v\,dv = a\,ds \).

\[
vdv = ads = -kv^2\,ds
\]

\[
\int_{v_0}^{v} \frac{dv}{v} = -k\int_{0}^{s} ds
\]

\[ks = \ln(v_0/v)\]

With this result it is easy to find \( k \) given \( v \) at some specified \( s \). To illustrate, assume that \( v_0 = 8 \) knots and that the speed of the boat is determined to be 3.9 knots after it has traveled one nautical mile.

\[k(1) = \ln\left(\frac{8}{3.9}\right) \quad k = 0.718 \text{ mi}^{-1}\]

(b) Here we follow the general approach in the sample problem. Integrating \( a = \frac{dv}{dt} \) yields

\[
\int_{v_0}^{v} \frac{dv}{v^2} = -k\int_{0}^{t} dt
\]

\[-kt = \frac{v - v_0}{vv_0} \quad v = \frac{v_0}{1 + ktv_0}\]

To obtain the distance \( s \) as a function of time we integrate \( v = ds/dt \)

\[
\int_{0}^{s} ds = \int_{0}^{t} v\,dt = \int_{0}^{t} \frac{v_0}{1 + ktv_0} dt
\]

\[s = \frac{1}{k} \ln(1 + ktv_0)\]

This equation turns out to be very difficult to solve for $k$. A good mathematician or someone familiar with symbolic algebra software might be able to find the general solution for $k$ in terms of the so-called LambertW function ($\text{LambertW}(x)$ is the solution of the equation $ye^y = x$). Even if this solution were found it would be of little use in most practical situations. For example, you would have to spend some time familiarizing yourself with the function. Once this is done you would still have to use a program like Maple or a mathematical handbook to evaluate the function.

For these reasons it is probably easiest to find $k$ either graphically or numerically. Obtaining a numerical solution with MATLAB is so easy that there is little reason not to use this approach. It is generally advisable though to use a graphical approach even when a numerical solution is being obtained. This is the best way to identify whether there are multiple solutions to the problem and also serves as a useful check on the numerical results. Thus, both approaches are illustrated below.

The usual way to generate a graphical solution is to rearrange the equation so as to give a function that is zero at points that are solutions to the original equation. Rearranging the equation above in this manner yields,

$$f = ks - \ln(1 + ktv_0) = 0$$

Given values of $s$, $t$, and $v_0$, $f$ can be plotted versus $k$. The value of $k$ at which $f = 0$ provides the solution to the original equation.

**MATLAB Worksheet and Scripts**

Although the integrations are simple in this problem, we'll go ahead and evaluate them symbolically for purposes of illustration.

EDU >> s_a = '-1/k'*int('1/x','x','v0','v')  % part (a)

s_a = -1/k*(log(v)-log(v0))

EDU >> s_b = int('v0/(1+k*x*v0)','x',0,'t')  % part (b)

s_b = log(1+t*k*v0)/k

The following script implements a graphical solution for part (b). To illustrate, we take $v_0 = 8$ knots and assume that the boat is found to move 1.1 nautical miles after 10 minutes.
The above graph shows that \( k \) is about 0.34 \( \text{mi}^{-1} \). Now let's try to find a solution with the symbolic \textit{solve}. First we write a Maple expression for \( f \) given our assumed values for \( v_0 \), \( s \), and \( t \).

\[
\text{EDU} \gg f = '1.1*k - \log(1+k*8*10/60) = 0'
\]

\[
f = 1.1*k - \log(1+k*8*10/60)=0
\]

\[
\text{EDU} \gg \text{solve}(f, 'k')
\]

\[
\text{ans} =
\]

\[0.
\]

\[.33923053342470867736031513009887
\]

Thus, \( k = 0.3392 \text{ mi}^{-1} \).
2.2 Problem 2/87 (Rectangular Coordinates)

A long-range rifle is fired at \( A \) with the projectile hitting the mountain at \( B \). (a) If the muzzle velocity is \( u \) \( \nabla \) 400 m/s, determine the two angles of elevation \( \theta \) which will permit the projectile to hit the mountain target \( B \) and plot the two trajectories. (b) Determine the smallest muzzle velocity that will allow the projectile to strike at \( B \) and the angle at which it must be fired. Repeat the plot for part (a) and include the trajectory of the projectile for this minimum initial velocity.

\[ \begin{align*}
( v_x )_0 &= u \cos \theta \\
( v_y )_0 &= u \sin \theta 
\end{align*} \]

The acceleration is constant with components \( a_x = 0 \) and \( a_y = -g \). Integrating these two accelerations twice and applying initial conditions yields (see page 44 of your text if you need additional details),

\[ \begin{align*}
x &= u \cos \theta t \\
y &= u \sin \theta t - \frac{1}{2} gt^2
\end{align*} \]

Plotting \( y \) in terms of \( x \) for different times \( t \) will yield the trajectory of the projectile. This type of plot is called a parametric plot since the items plotted \((x\ and\ y)\) are each known in terms of another parameter \((t)\).

Anytime you have a projectile motion problem and you know the coordinates of a point on the trajectory (our point \( B \)) you should solve for \( x \) and \( y \) (as we have done above) and then obtain two equations by substituting the coordinates of the points. These two equations can then be solved for two unknowns. Note that in most cases one of the two unknowns will be the time of flight.

**Part (a)** Substituting \( x = 5,000 \) m, \( y = 1,500 \) m and \( u = 400 \) m/s gives

\[ \begin{align*}
5000 &= 400 \cos \theta t \\
1500 &= 400 \sin \theta t - \frac{1}{2} (9.81) t^2
\end{align*} \]
We will let MATLAB solve these two equations simultaneously. MATLAB actually finds four solutions, however two of the four can be discarded since they involve negative times. The other two solutions correspond to the two solutions shown in the illustration accompanying the problem statement. The results are,

$$\theta_1 = 26.6^\circ$$ and $$\theta_2 = 80.6^\circ$$

**Part (b)** It should be intuitively obvious why there must be a minimum initial velocity below which the projectile cannot reach B. How do we go about finding it? We still have the two equations for the coordinates of point B,

$$5000 = u\cos \theta t \quad 1500 = u\sin \theta t - \frac{1}{2}(9.81)t^2$$

however there are now three unknowns ($u$, $\theta$, $t$). Suppose for the moment that the launch angle $\theta$ were given and we were asked to calculate the required initial speed $u$ so that the projectile strikes B. In this case we would have two equations and two unknowns. From this observation we see that $u$ is a function of $\theta$ from which we get our general solution strategy:

(a) Eliminate $t$ from the above two equations and solve for $u$ as a function of $\theta$.

(b) Differentiate this function with respect to $\theta$ to find the location of the minimum.

Solving the first equation for $u$ gives

$$u = \frac{5000}{t \cos \theta}$$

Substituting into the second yields

$$1500 = 5000 \tan \theta - \frac{1}{2} gt^2$$

This equation is now solved for $t = \sqrt{\frac{2(5000 \tan \theta - 1500)}{g}}$. This can be substituted back into $u$ to give

$$u = \frac{5000}{\cos \theta \sqrt{2(5000 \tan \theta - 1500)/g}}$$

We will let MATLAB differentiate this equation and solve for the minimum speed and the associated launch angle. The result is

$$u_{\text{min}} = 256.8 \text{ m/s at } \theta = 53.3^\circ$$
MATLAB Worksheet and Scripts

Part (a)

EDU>> eqn1='5000 = 400*cos(x)*y'
eqn1 =
5000 = 400*cos(x)*y

EDU>> eqn2='1500 = 400*sin(x)*y-1/2*9.81*y^2'
eqn2 =
1500 = 400*sin(x)*y-1/2*9.81*y^2

EDU>> [x,y] = solve(eqn1,eqn2)

x =
-1.7350349681568043127091034228443
-2.6858972177500184529892570306420
.4556954358397747854338635263748
1.4065576854329889257535399604352

y =
-76.452007728583426926316546391465
-13.920516408172829238156414017816
13.920516408172829238156414017816
76.452007728583426926316546391465

MATLAB finds four solutions but two can be excluded because they have time (y) being negative. The two positive times and corresponding angles are

\[ t_{B1} = 13.9205 \text{ secs} \quad \theta_1 = 0.4557 \text{ rads (26.6°)} \]

and \[ t_{B2} = 76.4520 \text{ secs} \quad \theta_2 = 1.4066 \text{ rads (80.6°)} \]

%========== script #1 ========================
% Note that we need to set up two time scales in order to
% insure that both trajectories end at point B.
% t1 = 0:0.01:13.92;
t2 = 0:0.01:76.45;
x1 = 400*cos(.455695)*t1;
y1 = 400*sin(.455695)*t1-1/2*9.81*t1.^2;
x2 = 400*cos(1.4066)*t2;
y2 = 400*sin(1.4066)*t2-1/2*9.81*t2.^2;
plot(x1/1000,y1/1000,x2/1000,y2/1000)
xlabel('x (km)')
ylabel('y (km)')
title('plot for part (a)')
%======== end script #1 =======================
Part (b)

EDU>> u = '5000/cos(x)/sqrt(2*(5000*tan(x)-1500)/9.81)'

EDU>> dudx = diff(u,'x');
EDU>> solve(dudx,'x')

ans =

.93112656063638185561346315653632
-0.63966976615851476361785853510343

Only one of the two solutions is between 0 and 90°.

EDU>> .9311*180/pi
ans =

53.3481
EDU>> subs(u,.9311)
ans =
256.7581

Thus, $u_{\text{min}} = 256.8$ m/s at $\theta = 53.3^\circ$

Before plotting we need to find the time to reach point $B$. This can be done from the equation $x = 5000 = u \cos \theta$.

EDU>> 5000/256.7581/cos(.9311)
ans =
32.6217

%============== script #2 ========================
% this script should be run after script 1
% t3 = 0:0.01:32.62;
x3 = 256.8*cos(.9311)*t3;
y3 = 256.8*sin(.9311)*t3-1/2*9.81*t3.^2;
plot(x1/1000,y1/1000,x2/1000,y2/1000,x3/1000,y3/1000)
title('plot for part (b)')
%======== end script #2 =======================
2.3 Problem 2/126 (n-t Coordinates)

A baseball player releases a ball with initial conditions shown in the figure. Plot the radius of curvature of the path just after release and at the apex as a function of the release angle $\theta$. Explain the trends in both results as $\theta$ approaches 90°.

**Problem Formulation**

*Just after release*

\[
a_n = g \cos \theta = \frac{v_0^2}{\rho}, \quad \rho = \frac{v_0^2}{g \cos \theta}
\]

*At the apex*

At the apex, $v_y = 0$ and $v = v_x = v_0 \cos \theta$. Since $v$ is horizontal, the normal direction is vertically downward so that $a_n = g$.

\[
a_n = g = \frac{(v_0 \cos \theta)^2}{\rho}, \quad \rho = \frac{(v_0 \cos \theta)^2}{g}
\]

**MATLAB Script**

```matlab
v0 = 100; g = 32.2; theta = 0:0.01:pi/2; rho_i = v0^2/g./cos(theta); rho_a = (v0*cos(theta)).^2/g; plot(theta*180/pi,rho_i, theta*180/pi,rho_a) xlabel('theta (deg)') title('radius of curvature (ft)') axis([0 90 0 800]) % note that we need to limit the vertical axis since the % initial rho approaches infinity as theta approaches % 90 degrees.
```

********** end of script **********
Note that as $\theta$ approaches 90°, the initial $\rho$ goes to infinity while $\rho$ at the apex approaches zero. When $\theta = 90^\circ$, the ball travels along a straight (vertical) path. As you recall, straight paths have a radius of curvature of infinity. At the apex, the velocity will be zero giving a radius of curvature of zero.
2.4 Sample Problem 2/9 (Polar Coordinates)

Rotation of the radially slotted arm is governed by \( \theta = 0.2t + 0.02t^3 \), where \( \theta \) is in radians and \( t \) is in seconds. Simultaneously, the power screw in the arm engages the slider \( B \) and controls its distance from \( O \) according to \( r = 0.2 + 0.04t^2 \), where \( r \) is in meters and \( t \) is in seconds. Calculate the magnitudes of the velocity and acceleration of the slider as a function of time \( t \).

(a) Plot \( v_r, v_\theta \) and \( v_\theta \) for \( t \) between 0 and 5 sec. (b) Plot \( a_r, a_\theta \) and \( a_\theta \) for \( t \) between 0 and 5 sec. (c) Plot the path of the slider \( B \) and compare with the result in your book.

Problem Formulation

The first part of this problem solution will be identical to that in the Sample Problem in your text except that everything will be left in terms of \( t \). To summarize,

\[
\begin{align*}
  r &= 0.2 + 0.04t^2 & \dot{r} &= 0.08t & \ddot{r} &= 0.08 \\
  \theta &= 0.2t + 0.02t^3 & \dot{\theta} &= 0.2 + 0.06t^2 & \ddot{\theta} &= 0.12t 
\end{align*}
\]

Now all we have to do is substitute these expressions into the definitions for the velocity and acceleration. As usual, there is no need to make an explicit substitution when using the computer.

\[
\begin{align*}
  v_r &= \dot{r} = 0.08t & v_\theta &= \dot{\theta} \\
  v &= \sqrt{v_r^2 + v_\theta^2} \\
  a_r &= \ddot{r} - r\dot{\theta}^2 & a_\theta &= \dot{r}\ddot{\theta} + 2\dot{r}\dot{\theta} \\
  a &= \sqrt{a_r^2 + a_\theta^2}
\end{align*}
\]

The plot for part (c) can be obtained by using the suggestion in your book. First write

\[
\begin{align*}
  x &= r \cos \theta & y &= r \sin \theta
\end{align*}
\]

and then plot \( y \) versus \( x \). This type of plot is called a parametric plot since \( y \) is not known explicitly as a function of \( x \). Instead, both \( x \) and \( y \) are known in terms of another parameter, namely \( t \).
MATLAB Scripts

%====== script #1 ================
% this script plots the velocities for part (a)
t = 0:0.01:5;
r = 0.2+0.04*t.^2;
rd = 0.08*t;
rd = 0.08;
theta = 0.2*t+0.02*t.^3;
theta = 0.2+0.06*t.^2;
theta = 0.12*t;
vr = rd;
vtheta = r.*theta;
v = sqrt(vr.^2+vtheta.^2);
plot(t,v,t,vr,t,vtheta)
xlabel('time (seconds)')
ylabel('velocity (m/s)')
title('part (a) velocity')
%====== end of script#1============
%====== script #2 ============================
% this script plots the accelerations for part (b)
% run script # 1 first
ar = rdd - r.*thetad.^2;
atheta = r.*thetadd+2*rd.*thetad;
a=sqrt(ar.^2+atheta.^2);
plot(t,a,t,ar,t,atheta)
xlabel('time (seconds)')
ylabel('acceleration (m/s^2)')
title('part (b) acceleration')
%=======end of script 2======================

%====== script #3 ====================
% This script plots the path of the
% slider for part (b).
% Run script # 1 first.
x = r.*cos(theta);
y = r.*sin(theta);
plot(x, y)
xlabel('x (m)')
ylabel('y (m)')
title('part (c) Path of the Slider')
%======end of script 3===============
part (c) Path of the Slider

$t = 0$

t = 5s
2.5 Sample Problem 2/10 (Polar Coordinates)

A tracking radar lies in the vertical plane of the path of a rocket which is coasting in unpowered flight above the atmosphere. For the instant when \( \theta = 30^\circ \), the tracking data give \( r = 25(10^4) \) feet, \( \dot{r} = 4000 \) ft/s, and \( \dot{\theta} = 0.8 \) deg/s. Let this instant define the initial conditions at time \( t = 0 \) and plot \( v_r \) and \( v_\theta \) as a function of time for the next 150 seconds. You may assume that \( g \) remains constant at 31.4 ft/s\(^2\) during this time interval.

**Problem Formulation**

Place a Cartesian coordinate system at the radar with \( x \) positive to the right and \( y \) positive up. Since the rocket is coasting in unpowered flight we can use the equations for projectile motion.

\[
\begin{align*}
x &= x_0 + v_0 \cos(\beta) t \\
y &= y_0 + v_0 \sin(\beta) - \frac{1}{2} gt^2
\end{align*}
\]

Where \( x_0 = 25(10^4) \sin(30) \) ft, \( y_0 = 25(10^4) \cos(30) \) ft, \( v_0 \) is the initial speed (5310 ft/sec, see the sample problem) and \( \beta \) is the angle that \( v_0 \) makes with the horizontal. From the figure shown to the right we can find the angle between \( v_0 \) and the \( r \) axis as \( \phi = \tan^{-1}(3490/4000) = 41.11^\circ \). Since the \( r \) axis is 60\(^\circ\) from the horizontal, \( \beta = 60 - 41.11 = 18.89^\circ \).

With \( r \) and \( \theta \) defined as in the sample problem we have, at any time \( t \)

\[
r = \sqrt{x^2 + y^2} \quad \theta = \tan^{-1}(x/y)
\]

Now we find \( v_r \) and \( v_\theta \) from their definitions.

\[
\begin{align*}
v_r &= \dot{r} \\
v_\theta &= r \dot{\theta}
\end{align*}
\]

Substitution of \( x \) and \( y \) into the above equations and carrying out the derivatives with respect to time gives \( v_r \) and \( v_\theta \) as functions of time. The results are very
messy and will not be given here. Remember, though, that substitutions such as this can be made automatically when using computer software such as MATLAB.

**MATLAB Scripts**

```matlab
% This script obtains symbolic results for the r and theta components of the velocity
syms x y t x0 y0 v0 beta g
x = x0+v0*cos(beta)*t
y = y0+v0*sin(beta)*t-1/2*g*t^2
r = sqrt(x^2+y^2)
theta = atan(x/y)
vr = diff(r,t);
vtheta = r*diff(theta,t);
% The results from these operations will be copied and pasted into script #2 for plotting. Before they can be used they have to be "vectorized", which places periods in front of the operators to insure term by term rather than matrix operations. We'll go ahead and do that here and then copy and paste the vectorized results.
vr = vectorize(vr)
vtheta = vectorize(vtheta)
```

**Output from Script #1**

\[
x = x0+v0*cos(beta)*t \\
y = y0+v0*sin(beta)*t-1/2*g*t^2 \\
r = ((x0+v0*cos(beta)*t)^2+(y0+v0*sin(beta)*t-1/2*g*t^2)^2)^(1/2) \\
theta = atan((x0+v0*cos(beta)*t)/(y0+v0*sin(beta)*t-1/2*g*t^2)) \\
vr = \\
1./2./((x0+v0.*cos(beta).*t).^2+(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).^2).^(1./2).*(2.*(x0+v0.*cos(beta).*t).*v0.*cos(beta)+2.*(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).*(v0.*sin(beta)-g.*t)) \\
vtheta = \\
((x0+v0.*cos(beta).*t).^2+(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).^2).^(1./2).*(v0.*cos(beta)./(y0+v0.*sin(beta).*t-1./2.*g.*t.^2)-v0.*sin(beta)-g.*t))./(1+(x0+v0.*cos(beta).*t).^2./(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).^2)
\]
This script plots the r and theta components of the velocity as functions of time.
x0 = 25*10^4*sin(30*pi/180);
y0 = 25*10^4*cos(30*pi/180);
v0 = 5310;
beta = 18.89*pi/180;
g = 31.4;
t = 0:0.1:150;
% The following expressions are copied and pasted from the results obtained with script #1.
vr = 1./2./((x0+v0.*cos(beta).*t).^2+(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).^2).^(1./2).*(2.*(x0+v0.*cos(beta).*t).*v0.*cos(beta)+2.*(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).*v0.*sin(beta)-g.*t))/((x0+v0.*cos(beta).*t).^2+(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).^2).

vtheta = ((x0+v0.*cos(beta).*t).^2+(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).^2).^(1./2).*(v0.*cos(beta)/(y0+v0.*sin(beta).*t-1./2.*g.*t.^2)-(x0+v0.*cos(beta).*t)/(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).^2).*v0.*sin(beta)-g.*t)/(1+(x0+v0.*cos(beta).*t).^2./(y0+v0.*sin(beta).*t-1./2.*g.*t.^2).^2);

plot(t, vr, t, vtheta)
xlabel('t (sec)')
ylabel('velocity (ft/sec)')
2.6 Problem 2/183 (Space Curvilinear Motion)

The base structure of the firetruck ladder rotates about a vertical axis through $O$ with a constant angular velocity $\dot{\theta} = \Omega$. At the same time, the ladder unit $OB$ elevates at a constant rate $\phi = \Psi$, and section $AB$ of the ladder extends from within section $OA$ at the constant rate $\dot{R} = \Lambda$. Find general expressions for the components of acceleration of point $B$ in spherical coordinates if, at time $t = 0$, $\theta = 0$, $\phi = 0$, and $AB = 0$. Express your answers in terms of $\Omega$, $\Psi$, $\Lambda$, $R_0$ and $t$, where $R_0 = OA$ and is constant. Plot the components of acceleration of $B$ as a function of time for the case $\Omega = 10$ deg/s, $\Psi = 7$ deg/s, $\Lambda = 0.5$ m/s, and $R_0 = 9$ m. Let $t$ vary between 0 and the time at which $\phi = 90^\circ$.

**Problem Formulation**

The components of acceleration in spherical coordinates are,

\[
a_r = \ddot{R} - R \dot{\phi}^2 - R \dot{\theta}^2 \cos^2 \phi
\]

\[
a_\theta = \frac{\cos \phi}{R} \frac{d}{dt} \left(R^2 \dot{\theta}\right) - 2R \dot{\theta} \dot{\phi} \sin \phi
\]

\[
a_\phi = \frac{1}{R} \frac{d}{dt} \left(R^2 \dot{\phi}\right) + R \dot{\theta}^2 \sin \phi \cos \phi
\]

The components may be obtained as functions of time by substituting,

\[R = R_0 + \Lambda t, \quad \theta = \Omega t \quad \text{and} \quad \phi = \Psi t\]

Differentiation and substitution will be performed in MATLAB. The results are,

\[a_r = (R_0 + \Lambda t) \left(\Psi^2 - \Omega^2 \cos^2 (\Psi t)\right)\]

\[a_\theta = 2\Omega \Lambda \cos(\Psi t) - 2\Omega \Psi (R_0 + \Lambda t) \sin(\Psi t)\]
\[ a_\rho = 2 \Psi \Lambda + (R_0 + \Lambda t) \Omega^2 \sin(\Psi t) \cos(\Psi t) \]

**MATLAB Scripts**

```
% This script calculates the components of the acceleration symbolically
% O = Omega; P = Phi; L = Lambda
syms O P L t R0
R = R0+L*t;
theta = O*t;
phi = P*t;
% Even though there are some obvious simplifications
% in this case, we still write the most general
% expressions for the spherical components of the
% acceleration. In this way we can consider other types
% of time dependence without modifying the script.
a_R = diff(R,t,2)-R*diff(phi,t)^2-R*diff(theta,t)^2*cos(phi)^2
a_theta = cos(phi)/R*diff(R^2*diff(theta,t),t)-2*R*diff(theta,t)*diff(phi,t)*sin(phi)
a_phi = 1/R*diff(R^2*diff(phi,t),t)+R*diff(theta,t)^2*sin(phi)*cos(phi)
```

Output of script #1

\[ a_R = -(R0+L*t)*P^2-(R0+L*t)*O^2*cos(P*t)^2 \]
\[ a_{\theta} = 2*cos(P*t)*O*L-(2*R0+2*L*t)*O*P*sin(P*t) \]
\[ a_{\phi} = 2*P*L+(R0+L*t)*O^2*sin(P*t)*cos(P*t) \]

```
% This script plots the components of the acceleration as functions of time
% O = Omega; P = Phi; L = Lambda
L = 0.5; O = 10*pi/180;
P = 7*pi/180; R0 = 9;
tf = pi/2/P; % time at which phi=pi/2
% Set the parameters for plotting
t=0:0.05:tf;
a_R = -(R0+L*t)*P^2-(R0+L*t).*O^2.*cos(P*t).*2;
a_theta = 2*cos(P*t).*O*L-(2*R0+2*L*t).*O*P.*sin(P*t);
a_phi = 2*P*L+(R0+L*t).*O^2.*sin(P*t).*cos(P*t);
plot(t,a_R,t,a_theta,t,a_phi)
```
2.7 Sample Problem 2/16 (Constrained Motion of Connected Particles)

The tractor $A$ is used to hoist the bale $B$ with the pulley arrangement shown. If $A$ has a forward velocity $v_A$, determine an expression for the upward velocity $v_B$ of the bale in terms of $x$. Put the result in nondimensional form by introducing the velocity ratio $\eta = \frac{v_B}{v_A}$ and nondimensional position $\chi = \frac{x}{h}$. Plot $\eta$ versus $\chi$ for $0 \leq \chi \leq 2$.

**Problem Formulation**

The length $L$ of the cable can be written

$$L = 2(h - y) + l + \text{cons} \tan ts = 2(h - y) + \sqrt{h^2 + x^2} + \text{cons} \tan ts$$

Now, $\dot{L} = 0$ will be used to obtain a relation between $v_A (= \dot{x})$ and $v_B (= \dot{y})$.

$$\dot{L} = 0 = -2\dot{y} + \frac{x\ddot{x}}{\sqrt{h^2 + x^2}} \quad v_B = \frac{1}{2} \frac{xv_A}{\sqrt{h^2 + x^2}}$$
The nondimensional result is now obtained by substituting \( v_B = \eta v_A \) and \( x = \chi h \).

\[
\eta = \frac{1}{2} \frac{\chi}{\sqrt{1 + \chi^2}}
\]

Even though these operations are rather easily performed by hand, it is instructive to have MATLAB do them. In particular, it will be instructive to see how to evaluate \( \dot{L} \) even though \( x \) and \( y \) are not known explicitly as functions of time.

**MATLAB Worksheet and Script**

EDU\» syms t h vA vB chi eta
EDU\» x = sym('x(t)'); y = sym('y(t)');
EDU\» L = 2*(h-y)+sqrt(h^2+x^2)
L =
2*h-2*y(t)+(h^2+x(t)^2)^(1/2)

Note that we need to differentiate \( L \) with respect to time. Both \( x \) and \( y \) depend on time, however, exactly how they depend on time is not known. It turns out that this is not a problem. All we need to do is let MATLAB know \( x \) and \( y \) depend on time by writing \( x = \text{sym('x(t)')} \) and \( y = \text{sym('y(t)')} \). To illustrate, consider the following.

EDU\» diff(x, t)
an =
diff(x(t),t)

This derivative would have been evaluated as zero had we not declared \( x \) to be a function of \( t \). Now let’s proceed by differentiating \( L \) with respect to \( t \). We give the result a name (\( \text{eqn} \)) to facilitate later substitutions.

EDU\» eqn = diff(L,t)
eqn =
-2*diff(y(t),t)+1/(h^2+x(t)^2)^(1/2)*x(t)*diff(x(t),t)

EDU\» eqn = subs(eqn,diff(y,t),eta*vA) % substitutes \( v_B = \eta v_A \) for \( \dot{y} \) (diff(y,t))
eqn =
-2*eta*vA+1/(h^2+x(t)^2)^(1/2)*x(t)*diff(x(t),t)

EDU\» eqn = subs(eqn,diff(x,t),vA)
eqn =
\[-2\eta v_A + \frac{1}{(h^2 + x(t)^2)^{1/2}}x(t)v_A\]

EDU» eqn = subs(eqn,x,chi*h)
\[\text{eqn} = -2\eta v_A + \frac{1}{(h^2 + \chi^2 h^2)^{1/2}}\chi h v_A\]

Now let’s remember that \(eqn\) is just a name for \(dL/dt\) which is zero. We now solve this equation for \(\eta(\eta)\). Also recall that \text{solve}(eqn, \eta)\) actually solves the equation \(eqn = 0\) for \(\eta\).

EDU» eta = solve(eqn, eta)
\[\eta = \frac{1}{2} \chi \frac{h}{(h^2 + \chi^2 h^2)^{1/2}}\]

We note finally that the \(h\) cancels in the above expression yielding the result given in the problem formulation section above. Now we can produce the required plot.

%%%%%%% Script for plotting \(\chi\) versus \(\eta\) %%%%%%
chi = 0:0.01:2;
eta = 1/2*chi./sqrt(1+chi.^2);
plot(chi,eta)
xlabel('x/h')
title('v_B/v_A')

end of script %%%%%%%%%%%%%%%%%

![Graph showing \(v_B/v_A\) vs. \(x/h\)]
The kinetics of particles is concerned with the motion produced by unbalanced forces acting on a particle. This chapter considers three approaches to the solution of particle kinetics problems: (1) direct application of Newton’s second law, (2) work and energy, and (3) impulse and momentum. Problem 3.1 is a rectilinear motion problem where `solve` is used to solve three equations symbolically for three unknowns. In problem 3.2, `dsolve` is used to solve a second order differential equation with initial conditions. The absolute path of a particle is then plotted using `polar`. Problem 3.3 uses MATLAB to study the effect of initial spring stretch upon the velocity of a slider. A physical interpretation of the results is also required. Problem 3.4 is a typical ballistic pendulum problem requiring both work/energy and conservation of momentum to relate the velocity of a projectile to the maximum swing angle of a pendulum. Problem 3.5 is a relatively straightforward conservation of angular momentum problem where MATLAB is used to generate a plot that might be useful in a parametric study. In problem 3.6, two equations are solved symbolically for two unknowns using `solve`. The maximum value of a function is then determined using `diff` and `solve`. 
3.1 Sample Problem 3/3 (Rectilinear Motion)

The 250-lb concrete block \( A \) is released from rest in the position shown and pulls the 400-lb log up the 30° ramp. Plot the velocity of the block as it hits the ground at \( B \) as a function of the coefficient of kinetic friction \( \mu_k \) between the log and the ramp. Let \( \mu_k \) vary between 0 and 1. Why does the computer not plot results for the entire range specified?

**Problem Formulation**

The constant length of the cable is \( L = 2s_C + s_A \) (see figure). Differentiating this expression twice yields a relation between the acceleration of \( A \) and \( C \) (note that \( a_C = a_{LOG} \)).

\[
0 = 2a_C + a_A \quad (1)
\]

From the free-body diagram for the log

\[
\begin{align*}
\Sigma F_y &= ma_y = 0 \\
N - 400\cos(30) &= 0
\end{align*}
\]

\[
\begin{align*}
\Sigma F_x &= ma_x \\
\mu_k N - 2T + 400\sin(30) &= \frac{400}{32.2} a_C
\end{align*}
\]

Substituting \( N \) yields,

\[
400\mu_k \cos(30) - 2T + 400\sin(30) = \frac{400}{32.2} a_C \quad (2)
\]

From the free-body diagram for block \( A \)

\[
\begin{align*}
\downarrow \Sigma F &= ma \\
250 - T &= \frac{250}{32.2} a_A
\end{align*}
\]

MATLAB will be used to solve the three equations above for \( a_A, a_C \) and \( T \) in terms of \( \mu_k \). Since the accelerations are constant, \( v_A^2 = 2a_A d \) where \( d \) is the vertical distance through which block \( A \) has fallen. Thus, the velocity of \( A \) when it strikes the ground \((d = 20 \text{ ft})\) is
\[ v_{Af} = \sqrt{40a_A} = v_B \]

**MATLAB Scripts**

\\begin{verbatim}
%%%%%%%%%%%%%%%%% Script #1 %%%%%%%%%%%%%%%%%%%%%%%%%
% This script solves the three equations developed
% in the problem formulation section for the two
% accelerations (aC and aA) and the tension T.
% When solving multiple equations, it is a good idea
% to let the variables solved for be single characters.
% Thus, we set x = aC, y = aA, z = T.
eqn1 = '2*x+y = 0';
eqn2 = '400*muk*cos(theta)-2*z+400*sin(theta)=400/32.2*x';
eqn3 = '250-z = 250/32.2*y';
[x,y,z]=solve(eqn1,eqn2,eqn3)
%%%%%%%%%%%%%%%%%%% end of script %%%%%%%%%%%%%%%%%%%%
\\end{verbatim}

Output from Script #1

\[ x = 7.9674*muk - 6.9 \]
\[ y = -15.9349*muk + 13.8 \]
\[ z = 142.857+123.7179*muk \]

The only thing we are interested in here is \( a_A \) (\( y \) in our solution above). Thus,

\[ a_A = 13.8 - 15.9349\mu_k \]

Note that the accelerations may be either positive or negative depending on the value of \( \mu_k \). The largest value of \( \mu_k \) for which the block will move up can thus be found by solving the equation \( a_A = 0 \) for \( \mu_k \). This yields \( \mu_k = 13.8/15.935 = 0.866 \).

\\begin{verbatim}
%%%%%%%%%%%%%%%%% Script #2 %%%%%%%%%%%%%%%%%%%%%%%%%
% This script plots the vB (the velocity of the
% block as it hits the ground) versus the friction
% coefficient mu_k
muk = 0:0.01:1;
aA=-15.9349*muk+13.8;
vB = sqrt(40*aA);
plot(muk,vB)
xlabel('mu_k')
ylabel('v_B (ft/sec)')
%%%%%%%%%%%%%%%%%%% end of script %%%%%%%%%%%%%%%%%%%%
\\end{verbatim}
At first sight, it seems that no results are plotted beyond the limiting value for $\mu_k$ (0.866) that was determined above. If the plot were in color, you would see that MATLAB has actually plotted zeros beyond this point. From a numerical point of view this occurs because MATLAB will plot only the real part of complex numbers. If you have MATLAB print the values of $v_B$ you will find that the real parts of all the complex numbers generated are zero. Whenever imaginary or complex values result there is usually some physical explanation. In this problem, the physical explanation is that the log will not slide up the incline if the coefficient of friction is too large.
3.2 Problem 3/98 (Curvilinear Motion)

The particle $P$ is released at time $t = 0$ from the position $r = r_0$ inside the smooth tube with no velocity relative to the tube, which is driven at the constant angular velocity $\omega_0$ about the vertical axis. Determine the radial velocity $v_r$, the radial position $r$, and the transverse velocity $v_\theta$ as functions of time $t$. Plot the absolute path of the particle during the time that it is inside the tube for $r_0 = 0.1$ m, $l = 1$ m, and $\omega_0 = 1$ rad/s.

**Problem Formulation**

From the free-body diagram to the right,

\[ \sum F_r = 0 = m a_r = m \left( \ddot{r} - r \dot{\theta}^2 \right) \]

\[ \ddot{r} = r \dot{\theta}^2 = r \omega_0^2 \]

Any book on differential equations will have the solution to this equation in terms of the hyperbolic sine and cosine,

\[ r = A \sinh(\omega_0 t) + B \cosh(\omega_0 t) \]

The constants $A$ and $B$ are found from the initial conditions. These conditions are that $r = r_0$ and $\dot{r} = 0$ at $t = 0$. The second condition comes from the fact that the particle has no velocity (initially) relative to the tube. Before evaluating this condition we must first differentiate $r$ with respect to time.

\[ \dot{r} = A \omega_0 \cosh(\omega_0 t) + B \omega_0 \sinh(\omega_0 t) \]

\[ r(t = 0) = r_0 = A \sinh(0) + B \cosh(0) = B \]

\[ \dot{r}(t = 0) = 0 = A \omega_0 \cosh(0) + B \omega_0 \sinh(0) = A \omega_0 \]

From the above we have $B = r_0$ and $A = 0$. Thus,

\[ r = r_0 \cosh(\omega_0 t) \]

From this we can obtain the radial and transverse velocities,
The absolute path of the particle will be graphed using polar plotting. For this we need \( r \) as a function of \( \theta \). Since \( \theta = \omega_0 t \) we have,

\[
    r = r_0 \cosh(\theta)
\]

We want to plot this function only up to the point where the particle leaves the tube. Substituting \( r = 1 \) we have \( 1 = 0.1 \cosh(\theta) \), or \( \theta = \cosh^{-1}(10) = 2.993 \) rads. Thus, the particle leaves the tube when \( \theta = 2.993 \) rads (171.5\(^\circ\)).

**MATLAB Worksheet**

EDU> theta = 0:0.01:2.993;
EDU> r = 0.1*cosh(theta);
EDU> polar(theta,r)
3.3 Sample Problem 3/17 (Potential Energy)

The 10-kg slider $A$ moves with negligible friction up the inclined guide. The attached spring has a stiffness of 60 N/m and is stretched $\delta$ m at position $A$, where the slider is released from rest. The 250-N force is constant and the pulley offers negligible resistance to the motion of the cord. Plot the velocity of the slider as it passes $C$ as a function of the initial spring stretch $\delta$. Let $\delta$ vary between –0.4 and 0.8 m and explain the results when $\delta$ exceeds a value of about 0.65 m.

**Problem Formulation**

The change in the elastic potential energy is

$$\Delta V_e = \frac{1}{2} k (x_f^2 - x_i^2) = \frac{1}{2} k ((1.2 + \delta)^2 - \delta^2)$$

The other results in the sample problem are unchanged,

$$U'_{1-2} = 250(0.6) = 150 \text{ J} \quad \Delta T = \frac{1}{2} m (v^2 - v_0^2) = \frac{1}{2} (10)v^2$$

$$\Delta V_g = mg\Delta h = 10(9.81)(1.2 \sin 30) = 58.9 \text{ J}$$

$$U'_{1-2} = 150 = \frac{1}{2} (10)v^2 + 58.9 + \frac{1}{2} (60)((1.2 + \delta)^2 - \delta^2))$$

This equation can be solved for $v$ either by hand or by using MATLAB. The result is

$$v = \frac{1}{10} \sqrt{958 - 1440\delta}$$

**MATLAB Worksheet and Script**

First we will solve the work/energy equation symbolically for $v$.

EDU» eqn='150=5*v^2+58.9+30*((1.2+del)^2-del^2)'
eqn = 150 = 5*v^2 + 58.9 + 30*((1.2 + del)^2 - del^2)

EDU» solve(eqn,'v')
ans =
   0.1*(958 - 1440*del)^(1/2)
   -0.1*(958 - 1440*del)^(1/2)

% This script plots the velocity of the slider at C versus the initial spring stretch del
% del = -0.4:0.01:0.8;
v = 1/10*sqrt(958-1440*del);
plot(del,v)
axis([-0.4 0.8 0 4])
xlabel('initial spring stretch (m)')
ylabel('velocity (m/s)')

% end of script
At first sight, it seems that no results are plotted beyond $\delta \equiv 0.65$ m. If the plot were in color, you would see that MATLAB has actually plotted zeros beyond this point. Why? One reason is to observe from the above equation that $v$ becomes imaginary when $\delta > 958/1440 = 0.665$ m. MATLAB interprets imaginary numbers as complex numbers with a real part equal to zero. When asked to plot a complex number it will plot only the real part. Thus, we have zeros plotted after $\delta = 0.665$ m. But this is a numerical reason instead of a physical explanation. Usually, imaginary answers signify a situation that is physically impossible for some reason. One way of understanding this is as follows. If the spring is initially compressed it will, at least for some part of the motion, be pushing up and thus be aiding the 250 N force in overcoming the weight of the slider. If the spring is initially stretched, it will always be pulling back on the slider. Thus the 250 N force will have to overcome not only the weight but also the spring force. It stands to reason then that there will be some value for the initial spring stretch beyond which the 250 N force will not be able to pull the slider all the way to $C$. This value is found from the limiting case where $v = 0$. Thus, the block never reaches $C$ if $\delta > 0.665$ m.
3.4 Problem 3/218 (Linear Impulse/Momentum)

The ballistic pendulum is a simple device to measure the projectile velocity \( v \) by observing the maximum angle \( \theta \) to which the box of sand with embedded particle swings. As an aid for a laboratory technician, make a plot of the velocity \( v \) in terms of the maximum angle \( \theta \). Assume that the weight of the box is 50-lb while the weight of the projectile is 2-oz.

**Problem Formulation**

(1) **Impulse/Momentum**

During impact, \( \Delta G = 0 \) and \( G_1 = G_2 \)

\[
\left(\frac{2/16}{32.2}\right)v + \left(\frac{50}{32.2}\right)(0) = \left(\frac{2/16 + 50}{32.2}\right)v_b
\]

\[v = 401v_b\]

where \( v \) is the velocity of the projectile while \( v_b \) is the velocity of the box of sand immediately after impact.

(2) **Work/Energy**

Now we use the work/energy equation with our initial position being the position where the pendulum is still vertical (\( \theta = 0 \)) and the final position is that where the pendulum has rotated through the maximum angle \( \theta \).

\[U_{1-2} = 0 = \Delta T + \Delta V_g = \frac{1}{2}m(0^2 - v_b^2) + mg\Delta h\]

where \( m \) is the combined mass of the box and the projectile.

\[v_b = \sqrt{2g\Delta h} = \sqrt{2(32.2)(6)(1 - \cos \theta)}\]

\[v = 401v_b = 7882\sqrt{1 - \cos \theta}\]
MATLAB Worksheet

EDU» theta = 0:0.01:pi/2;
EDU» v = 7882*sqrt(1-cos(theta));
EDU» plot(theta*180/pi,v)
EDU» xlabel('theta (degrees)')
EDU» title('velocity of projectile (ft/s)')
3.5 Problem 3/250 (Angular Impulse/Momentum)

The assembly of two 5-kg spheres is rotating freely about the vertical axis at 40 rev/min with \( \theta = 90^\circ \). The force \( F \) that maintains the given position is increased to raise the base collar and reduce the angle from 90\(^\circ\) to an arbitrary angle \( \theta \). Determine the new angular velocity \( \omega \) and plot \( \omega \) as a function of \( \theta \) for \( 0 \leq \theta \leq 90^\circ \). Assume that the mass of the arms and collars is negligible.

**Problem Formulation**

Since the summation of moments about the vertical axis is zero we have conservation of angular momentum about that axis. The spheres are rotating in a circular path about the vertical axis. The angular momentum of a particle moving in a circular path of radius \( r \) with angular velocity \( \omega \) is \( H = mr^2 \omega \). Thus, from the conservation of angular momentum we have,

\[
2mr^2_0 \omega_0 = 2mr^2 \omega \quad \Rightarrow \quad \omega = \frac{r_0^2}{r^2} \omega_0
\]

where \( r_0 = 0.1 + 0.6 \cos(45^0) \) and \( r = 0.1 + 0.6 \cos(\theta / 2) \)

**MATLAB Script**

```matlab
%%%%%%%%%%% script %%%%%%%%%%%%%%%
theta = 0:0.05:pi/2;
r0 = 0.1+0.6*cos(pi/4);
r = 0.1+0.6*cos(theta/2);
w0 = 40*2*pi/60;
w = r0^2./r.^2*w0;
plot(theta*180/pi, w)
xlabel('theta (degrees)')
ylabel('omega (rad/s)')
%%%%%%%% end of script %%%%%%%%%%%
```
This diagram “begins” at $\theta = 90^\circ$ where $\omega = \omega_i = 40(2\pi)/60 = 4.19 \text{ rad/s}$.

### 3.6 Problem 3/365 (Curvilinear Motion)

The 26-in. drum rotates about a horizontal axis with a constant angular velocity $\Omega = 7.5 \text{ rad/sec}$. The small block $A$ has no motion relative to the drum surface as it passes the bottom position $\theta = 0$. Determine the coefficient of static friction $\mu_s$ that would result in block slippage at an angular position $\theta$, plot your expression for $0 \leq \theta \leq 180^\circ$. Determine the minimum required coefficient value $\mu_{\text{min}}$ that would allow the block to remain fixed relative to the drum throughout a full revolution. For a friction coefficient slightly less than $\mu_{\text{min}}$, at what angular position $\theta$ would slippage occur?

#### Problem Formulation

From the free body and mass acceleration diagrams,

\[
\begin{align*}
\Sigma F_n &= ma_n \quad N - mg \cos \theta = mr\Omega^2 \\
\Sigma F_t &= ma_t \quad F - mg \sin \theta = 0
\end{align*}
\]
For impending slip we have $F = \mu_s N$. Substituting $F$ into the above and solving gives,

$$\mu_s = \frac{g \sin \theta}{g \cos \theta + r\Omega^2} = \frac{\sin \theta}{1.8925 + \cos \theta}$$

The last two questions can be answered only after plotting $\mu_s$ as a function of $\theta$.

**MATLAB Worksheet and Scripts**

```
% This script solves our two equations symbolically for $\mu_s$ and $N$
% O = Omega
% x = mu_s
% y = N
syms theta O g m r x y
eqn1 = y-m*g*cos(theta)-m*r*O^2;
eqn2 = x*y-m*g*sin(theta);
[x,y] = solve(eqn1,eqn2)
```

```
% This script plots $\mu_s$ as a function of $\theta$
theta = 0:0.02:pi;
mu_s = sin(theta)./(1.8925+cos(theta));
plot(theta*180/pi, mu_s)
xlabel('theta (degrees)')
title('coefficient of static friction')
```
If the block is not to slip at any angle $\theta$, the coefficient of friction must be greater than or equal to any value shown on the plot above. Thus, the minimum required coefficient value $\mu_{\text{min}}$ that would allow the block to remain fixed relative to the drum throughout a full revolution is equal to the maximum value in the plot above. The location where this maximum occurs can be found by solving the equation $d\mu_s / d\theta = 0$ for $\theta$. This $\theta$ can then be substituted into $\mu_s$ to yield the required value for $\mu_{\text{min}}$. Here's how we do this with MATLAB.

```
EDU» syms theta
EDU» mu_s = sin(theta)/(1.8925+cos(theta));
EDU» dmu = diff(mu_s,theta)
    dmu = cos(theta)/(757/400+cos(theta))+sin(theta)^2/(757/400+cos(theta))^2

EDU» theta_m = solve(dmu,theta)
    theta_m =
          -atan(1/302800*236697316401^(1/2))+pi
              atan(1/302800*236697316401^(1/2))-pi

EDU» eval(theta_m)
    ans =
          2.1275
             -2.1275

EDU» mu_min = subs(mu_s,theta,2.1275)
    mu_min = 0.6224
```

From the above we see that $\mu_{\text{min}} = 0.622$. If $\mu_s$ is slightly less than this value, the block will slip when $\theta = 2.128$ rads (121.9°).
This chapter concerns the extension of principles covered in chapters two and three to the study of the motion of general systems of particles. The chapter first considers the three approaches introduced in chapter 3 (direct application of Newton’s second law, work/energy, and impulse/momentum) and then moves to other applications such as steady mass flow and variable mass. Problem 4.1 considers an application of the conservation of momentum to a system comprised of a small car and an attached rotating sphere. MATLAB is used to plot the velocity of the car as a function of the angular position of the sphere. The absolute position of the sphere is also plotted. Problem 4.2 uses the concept of steady mass flow to study the effects of geometry upon the design of a sprinkler system. One of the main purposes of this problem is to illustrate how a problem can be greatly simplified using non-dimensional analysis. In particular, an equation containing seven parameters is reduced to a non-dimensional equation with only three parameters. Problem 4.3 is a variable mass problem in which MATLAB is used to integrate the kinematic equation $v dv = adx$. 
4.1 Problem 4/26 (Conservation of Momentum)

The small car, which has a mass of 20 kg, rolls freely on the horizontal track and carries the 5-kg sphere mounted on the light rotating rod with \( r = 0.4 \) m. A geared motor drive maintains a constant angular speed \( \dot{\theta} = 4 \) rad/s of the rod. If the car has a velocity \( v = 0.6 \) m/s when \( \theta = 0 \), plot \( v \) as a function of \( \theta \) for one revolution of the rod. Also plot the absolute position of the sphere for two revolutions of the rod. Neglect the mass of the wheels and any friction.

**Problem Formulation**

Since \( \Sigma F_x = 0 \) we have conservation of momentum in the \( x \) direction. The diagram to the right shows the system at \( \theta = 0 \) and at an arbitrary angle \( \theta \). From the relative velocity equation, the velocity of the sphere is the vector sum of the velocity of the car (\( v \)) and the velocity of the sphere relative to the car (\( r\dot{\theta} \)).

\[
\begin{align*}
(G_x)_{\theta=0} &= 20(0.6) + 5(0.6) = 15 \text{ Ns} \\
(G_x)_{\theta} &= 20v + 5(v - r\dot{\theta}\sin\theta) = 25v - 8\sin\theta
\end{align*}
\]

Setting \( (G_x)_{\theta=0} = (G_x)_{\theta} \) and solving yields,

\[ v = 0.6 + 0.32\sin\theta \]

Now let time \( t = 0 \) be the time when \( \theta = 0 \) and place an \( x-y \) coordinate system at the center of the car as shown in the diagram so that \( x(t) \) is the position of the center of the car. Since \( v = \frac{dx}{dt} \) and \( \theta = 4t \) we have,

\[
x = \int_0^t vdt = \int_0^t (0.6 + 0.32\sin(4t))dt = 0.6t + 0.08(1 - \cos(4t))
\]

The \( x \) and \( y \) components of the sphere can now be determined as,

\[
x_s = x + r\cos\theta = 0.08 + 0.6t + 0.32\cos(4t)
\]
\[ y_s = r \sin \theta = 0.4 \sin(4t) \]

The absolute position of the sphere can be obtained by plotting \( y_s \) versus \( x_s \). The time required for two revolutions of the arm is \( 4\pi/4 = \pi \) seconds.

**MATLAB Scripts**

\[ \text{Script #1} \]
\[
\theta = 0:0.02:2*\pi; \\
v = 0.6 + 0.32 \sin(\theta); \\
plot(\theta*180/\pi, v) \\
xlabel('\theta (degrees)') \\
ylabel('v (m/s)') \\
\]

\[ \text{Script #2} \]
\[
t = 0:0.01:\pi; \\
x_s = 0.08 + 0.6* t + 0.32 \cos(4* t); \\
y_s = 0.4 \sin(4* t); \\
plot(x_s, y_s) \\
xlabel('x (m)') \\
ylabel('y (m)') \\
title('position of the sphere') \\
\]

![Graph of \( v \) vs \( \theta \)]

![Graph of position of the sphere]
4.2 Problem 4/62 (Steady Mass Flow)

The sprinkler is made to rotate at the constant angular velocity $\omega$ and distributes water at the volume rate $Q$. Each of the four nozzles has an exit area $A$. Write an expression for the torque $M$ on the shaft of the sprinkler necessary to maintain the given motion. Here we would like to study the effects of the geometry of the sprinkler upon this torque. To this end, it is helpful to introduce the non-dimensional parameters $M' = M / 4\rho Ar^2$, $\Omega = \omega r / u$, and $\beta = b / r$ where $u = Q / 4A$ is the velocity of the water relative to the nozzle and $\rho$ is the density of the water. Plot the non-dimensional torque $M'$ versus $\beta$ for $\Omega = 0.5, 1, 2$. Let $\Omega_0$ be the non-dimensional velocity $\Omega$ at which the sprinkler will operate with no applied torque. Plot $\Omega_0$ versus $\beta$. For both plots let $\beta$ range between 0 and 1.

**Problem Formulation**

The figure to the right shows the three components of the absolute velocity of the water at the exit. $u (= Q / 4A)$ is the velocity of the water relative to the nozzle. The mass flow rate $m' = \rho Q$. Taking clockwise as positive, the application of equation 4/19 of your text yields,

$$\sum M_o = -M = \rho Q\left(\omega r^2 + \omega b^2 - ur - 0\right)$$
Now we want to introduce the non-dimensional parameters defined in the problem statement. For many undergraduate students, non-dimensional analysis is a very confusing topic. It is important to realize that the difficulty is really that of determining which non-dimensional parameters are appropriate for a particular problem. If these parameters have already been defined, as in this problem, all you have to do is substitute. In this case we merely substitute \( M = 4\rho Ar^2 M', \ \omega = \Omega u/r, \) and \( b = r\beta \) into the equation above. When this is done many terms will cancel yielding,

\[
M' = 1 - \Omega \left( 1 + \beta^2 \right)
\]

Setting \( M' = 0 \) we can solve for \( \Omega_0 \),

\[
\Omega_0 = \frac{1}{1 + \beta^2}
\]

**MATLAB Scripts**

```
%% Script #1  
% This script plots the non-dimensional torque 
% M' as a function of beta = b/r for Omega = 
% 0.5, 1, and 2
beta = 0:0.01:1;
Mp = inline('1-Omega*(1+beta.^2)')
plot(beta, Mp(0.5,beta),beta, Mp(1,beta),beta, Mp(2,beta))
xlabel('beta = b/r')
title('non-dimensional torque')
```

%%% end of script %%%%%%%
This script plots the non-dimensional angular velocity $\Omega_0$ as a function of $\beta = b/r$

```matlab
beta = 0:0.01:1;
Omega0 = 1./(1+beta.^2);
plot(beta, Omega0)
xlabel('beta = b/r')
```

% This script plots the non-dimensional angular velocity $\Omega_0$ as a function of $\beta = b/r$
beta = 0:0.01:1;
Omega0 = 1./(1+beta.^2);
plot(beta, Omega0)
xlabel('beta = b/r')

end of script
4.3 Problem 4/86 (Variable Mass)

The open-link chain of length \( L \) and mass \( \rho \) per unit length is released from rest in the position shown, where the bottom link is almost touching the platform and the horizontal section is supported on a smooth surface. Friction at the corner guide is negligible. Determine (a) the velocity \( v_1 \) of end \( A \) as it reaches the corner and (b) its velocity \( v_2 \) as it strikes the platform. Plot \( v_1 \) and \( v_2 \) as functions of \( h \) for \( L = 5 \) m.

**Problem Formulation**

Let \( x \) be the displacement of the chain and \( T \) be the tension in the chain at the corner as shown in the diagram to the right. Note that the acceleration of the horizontal and vertical sections are both equal to \( \ddot{x} \).

For the horizontal section,

\[
\left[ \Sigma F_x = ma_x \right] \quad T = \rho(L - h - x)\ddot{x}
\]

For the vertical section,

\[
\left[ \Sigma F_y = ma_y \right] \quad \rho gh - T = \rho \ddot{x}
\]

Substituting \( T \) from the first equation into the second and simplifying gives,

\[
\ddot{x} = \frac{gh}{L - x}
\]

Now we use the relation \( vdv = adx \) to write,

\[
\int_{0}^{v_1} vdv = \int_{0}^{L-h} \frac{gh}{L - x}dx \quad \Rightarrow \quad v_1^2 = 2 \int_{0}^{L-h} \frac{gh}{L - x}dx = 2gh \ln\left(\frac{L}{h}\right)
\]

\[
v_1 = \sqrt{2gh \ln(L/h)}
\]
After end $A$ has passed the corner it will be in free-fall until it hits the platform. With $y$ positive down we have $v dv = g dy$ yielding,

$$\frac{1}{2} \left( v_2^2 - v_1^2 \right) = gh$$

Substituting for $v_1$ and solving,

$$v_2 = \sqrt{2gh \left( 1 + \ln(L/h) \right)}$$

**MATLAB Worksheet and Script**

First we solve the equation

$$\int_0^{v_1} v dv - \int_0^{L-h} gh / (L-x) dx = 0$$

for $v_1$, omitting (as usual) the “$=0$”.

```
EDU» syms v v1 v2 h g x L

First we solve the equation

```
EDU» eqn1 = int(v,v,0,v1)-int(g*h/(L-x),x,0,L-h)

```
eqn1 =

```
1/2*v1^2+log(h)*g*h-log(L)*g*h
```

```
EDU» solve(eqn1,v1)

```
an =

```
(2*log(h)*g*h+2*log(L)*g*h)^(1/2)
```

MATLAB has found two solutions. The first is the one we want since it is positive. This solution can easily be simplified to the result given above in the problem formulation section. Once $v_1$ is known it is rather easy to find $v_2$. We’ll do it symbolically here for purposes of illustration.

First we copy and paste the first solution above to define $v_1$. Then we solve the equation

$$\int_{v_1}^{v_2} v dv - \int_0^h g dy = 0$$

for $v_2$. Note how the result for $v_1$ is automatically substituted.

```
EDU» v1 = (2*log(h)*g*h+2*log(L)*g*h)^(1/2);
EDU» eqn2 = int(v,v,v1,v2)-int(g,x,0,h)

```
eqn2 =

```
1/2*v2^2+log(h)*g*h-log(L)*g*h-g*h
```

```
EDU» solve(eqn2,v2)

```
an =

```
(-2*log(h)*g*h+2*log(L)*g*h)^(1/2)
```

```
(-2*log(h)*g*h+2*log(L)*g*h)^(1/2)
```

After end $A$ has passed the corner it will be in free-fall until it hits the platform. With $y$ positive down we have $v dv = g dy$ yielding,

$$\frac{1}{2} \left( v_2^2 - v_1^2 \right) = gh$$

Substituting for $v_1$ and solving,

$$v_2 = \sqrt{2gh \left( 1 + \ln(L/h) \right)}$$

**MATLAB Worksheet and Script**

EDU» syms v v1 v2 h g x L

First we solve the equation

$$\int_0^{v_1} v dv - \int_0^{L-h} gh / (L-x) dx = 0$$

for $v_1$, omitting (as usual) the “$=0$”.

```
EDU» eqn1 = int(v,v,0,v1)-int(g*h/(L-x),x,0,L-h)

```
eqn1 =

```
1/2*v1^2+log(h)*g*h-log(L)*g*h
```

```
EDU» solve(eqn1,v1)

```
an =

```
(2*log(h)*g*h+2*log(L)*g*h)^(1/2)
```

MATLAB has found two solutions. The first is the one we want since it is positive. This solution can easily be simplified to the result given above in the problem formulation section. Once $v_1$ is known it is rather easy to find $v_2$. We’ll do it symbolically here for purposes of illustration.

First we copy and paste the first solution above to define $v_1$. Then we solve the equation

$$\int_{v_1}^{v_2} v dv - \int_0^h g dy = 0$$

for $v_2$. Note how the result for $v_1$ is automatically substituted.

```
EDU» v1 = (2*log(h)*g*h+2*log(L)*g*h)^(1/2);
EDU» eqn2 = int(v,v,v1,v2)-int(g,x,0,h)

```
eqn2 =

```
1/2*v2^2+log(h)*g*h-log(L)*g*h-g*h
```

EDU» solve(eqn2,v2)

```
an =

```
(-2*log(h)*g*h+2*log(L)*g*h)^(1/2)
```

```
(-2*log(h)*g*h+2*log(L)*g*h)^(1/2)
```

---
EDU» solve(eqn2,v2)
ans =
(-2*log(h)*g*h+2*log(L)*g*h+2*g*h)^(1/2)
-(-2*log(h)*g*h+2*log(L)*g*h+2*g*h)^(1/2)

Once again, the first solution will simplify to that given in the problem formulation section above.
This chapter extends the kinematic analysis of particles covered in chapter 2 to rigid bodies by taking into account the rotational motion of the body. Thus, the motion of rigid bodies involves both translation and rotation. Problem 5.1 is a straightforward problem that compares the angular position of a rotating disk to the total number of revolutions. Problem 5.2 is an interesting application of absolute motion analysis. The problem illustrates the usefulness of non-dimensional analysis in conducting a parametric study of the effects of geometry upon the performance of a hydraulic lift. In problem 5.3 the velocity of the piston in a reciprocating engine is plotted versus the angular orientation of the crank. The maximum velocity and the corresponding orientation (of the crank) are obtained using \textit{max} after first trying \textit{diff} and \textit{solve}. The instantaneous center of zero velocity is used in problem 5.4 to relate the velocity of a vertical control rod to the angular velocity of a rotating bar in a switching device. Problem 5.4 is a fairly straightforward relative acceleration problem. Problem 5.5 considers the reciprocating engine of problem 5.3 but uses absolute rather than relative motion analysis. The problem illustrates the use of computer algebra for carrying out some tedious calculations including differentiation and substitution.
5.1 Problem 5/3 (Rotation)

The angular velocity of a gear is controlled according \( \omega = 12 - 3t^2 \) where \( \omega \), in radians per second, is positive in the clockwise sense and where \( t \) is the time in seconds. Find the net angular displacement \( \Delta \theta \) and the total number of revolutions \( N \) through which the gear turns in terms of the time \( t \). Plot \( \Delta \theta \) and \( N \) for time \( t \) up to 4 seconds.

**Problem Formulation**

You may want to have a look at sample problem 5/1 in your text before continuing with this problem. In particular it is important to note the difference between the angular displacement \( \Delta \theta \) and total number of revolutions \( N \). The angular displacement is the integral of \( \omega \) over time and can be positive or negative depending on whether the rotation is clockwise or counterclockwise. Referring to the graph of \( \omega \) to the right, \( \Delta \theta \) will be the total area under the curve up to a particular time. The area is negative when the curve dips below the time axis. As the name implies, the total number of revolutions simply counts the number of times the disk rotates and is not concerned with whether the rotation is clockwise or counterclockwise. Thus, \( N \) will be proportional to the magnitude of the area under the angular velocity curve. For example, suppose the gear rotates two revolutions clockwise followed by two revolutions counterclockwise. In this case \( \Delta \theta = 0 \) but \( N = 4 \).

Based upon the above it is essential to know beforehand the time when \( \omega = 0 \) (i.e. when the gear changes direction) in order to calculate \( N \). Setting \( 12 - 3t^2 = 0 \) gives \( t = 2 \) seconds. For time greater than 2 sec. we need to break the integral up into two parts when calculating \( N \). For \( \Delta \theta \) we do not need to keep track of when the gear changes directions. Thus,

\[
\left[ \omega = \frac{d\theta}{dt} \right] \quad \Delta \theta = \int_0^t \omega \, dt = \int_0^t \left( 12 - 3u^2 \right) du = 12t - t^3
\]
for \( t < 2 \) sec
\[
N = \frac{1}{2\pi} \int_0^t \omega \, dt = \frac{1}{2\pi} \left[ (12 - 3t^2) \right]_0^t = \frac{1}{2\pi} \left( 12t - t^3 \right)
\]

for \( t > 2 \) sec
\[
N = \frac{1}{2\pi} \int_0^2 \omega \, dt - \frac{1}{2\pi} \int_t^2 \omega \, dt = \frac{1}{2\pi} \left( \int_0^2 (12 - 3t^2) \, dt - \int_t^2 (12 - 3t^2) \, dt \right)
\]
\[
N = \frac{1}{2\pi} \left( 32 - 12t + t^3 \right)
\]

In summary,
\[
\Delta \theta = 12t - t^3 \quad \text{all } t
\]
\[
N = \begin{cases} 
\frac{1}{2\pi} \left( 12t - t^3 \right) & t < 2 \text{ sec} \\
\frac{1}{2\pi} \left( 32 - 12t + t^3 \right) & t > 2 \text{ sec}
\end{cases}
\]

Although the expression for \( N \) is rather simple, plotting the result can be a little tricky due to the change in the expression at time \( t = 2 \) sec. How do we tell MATLAB to stop plotting one expression and start plotting the other?

Actually, this is considerably easier to do in MATLAB than in some other applications (e.g. Maple) due to the fact that you are not limited to a single range variable for the ordinate. Thus we can easily set up two time variables \( t_1 \) and \( t_2 \) and then write the two expressions in terms of those two variables.

**MATLAB Scripts**

```matlab
%==== script #1 =========================
% This script plots the angular velocity as
% a function of time. Note the way a horizontal
% line at zero is plotted. This is useful for
% visualizing where omega becomes negative.
t = 0:0.05:4;
omega = 12 - 3*t.^2;
zero = 0*t;
plot(t,omega,t,zero)
xlabel('time (sec)')
title('angular velocity (rad/s)')
%====== end of script ======================
```
%==== script #2 ========================== 
% This script plots the angular position 
% (in degrees) as a function of time.

```
t = 0:0.05:4;
theta = (12*t - t.^3)*180/pi;
zero = 0*t;
plot(t,theta,t,zero)
xlabel('time (sec)')
title('angular position (degrees)')
```
%======= end of script ===================
% This script plots the number of revolutions
% as a function of time. Note that different
% time variables are used for the two different
% expressions for N

t1 = 0:0.05:2;
N1 = (12*t1 - t1.^3)/2/pi;
t2 = 2:0.05:4;
N2 = (32 - 12*t2 + t2.^3)/2/pi;
plot(t1,N1,t2,N2)
xlabel('time (sec)')
title('total number of revolutions')

% end of script
null
5.2 Problem 5/44 (Absolute Motion)

Derive an expression for the upward velocity \( v \) of the car hoist system in terms of \( \theta \). The piston rod of the hydraulic cylinder is extending at the rate \( \dot{s} \). Plot the non-dimensional velocity \( v / \dot{s} \) as a function of \( \theta \) for \( b/L = 0.1, 0.5, 1, \) and 2.

**Problem Formulation**

From the diagram to the right,

\[
y = 2b \sin \theta
\]

\[
\dot{y} = 2b \dot{\theta} \cos \theta
\]

If the angular velocity \( \dot{\theta} \) were known as a function of \( \theta \) we would be finished. The motion of the car hoist system is controlled by the extension rate \( \dot{s} \) of the hydraulic cylinder rather than the angular velocity. Thus, to complete the problem we need to relate \( \dot{\theta} \) and \( \dot{s} \).

\[
s^2 = L^2 + b^2 - 2Lb \cos \theta
\]

\[
2s \dot{s} = 0 + 0 + 2Lb \dot{\theta} \sin \theta
\]

\[
\dot{\theta} = \frac{s \dot{s}}{Lb \sin \theta}
\]

Substituting,

\[
v = \frac{2b s \dot{s}}{Lb \sin \theta} \cos \theta = \frac{2 \dot{s} \sqrt{L^2 + b^2 - 2Lb \cos \theta}}{L \tan \theta}
\]

\[
v / \dot{s} = \frac{2 \sqrt{1 + (b / L)^2 - 2(b / L) \cos \theta}}{\tan \theta} \cos \frac{\beta^2 - 2 \beta \cos \theta}{\tan \theta}
\]

where \( \beta = b/L \).
MATLAB Script

```matlab
v = inline('2/tan(theta)*sqrt(1^2+beta^2-2*beta*cos(theta))');
v = vectorize(v);

th = 10*pi/180:0.01:pi/2;
deg = th*180/pi;
plot(deg,v(.1,th),deg,v(.5,th),deg,v(1,th),deg,v(2,th))
axis([10 90 0 5])
xlabel('theta (degrees)')
title('non-dimensional velocity')
```

![Graph showing non-dimensional velocity as a function of theta for different values of beta]
5.3 Sample Problem 5/9 (Relative Velocity)

The common configuration of a reciprocating engine is that of the slider crank mechanism shown. If crank \( OB \) has a clockwise rotational speed of 1500 rev/min; (a) Plot \( v_A \) versus \( \theta \) for one revolution of the crank. (b) Find the maximum speed of the piston \( A \) and the corresponding value of \( \theta \).

Problem Formulation

Let \( l \) be the length of connecting rod \( AB \). Start with the relative velocity equation

\[ v_B = v_A + v_{B/A} \]

The crank pin velocity is \( v_B = r\omega \) and is normal to \( OB \). The velocity of \( A \) is horizontal while the velocity of \( B/A \) has magnitude \( l\omega_{AB} \) and is directed perpendicular to \( AB \). The angle \( \beta \) can be found in terms of \( \theta \) by using the law of sines,

\[ \sin \beta = \frac{r}{l} \sin \theta \]

Also, \( \cos \beta = \sqrt{1 - \sin^2 \beta} = \sqrt{1 - \left(\frac{r}{l}\right)^2 \sin^2 \theta} \)

From the vector diagram to the right,

\[ v_B \cos \theta = v_{B/A} \cos \beta \]

Thus, \( v_{B/A} = \frac{\cos \theta}{\cos \beta} \cdot r\omega \)

Also from the diagram, \( v_A = v_B \sin \theta + v_{B/A} \sin \beta \).

Substitution yields

\[ v_A = r\omega (\sin \theta + \cos \theta \tan \beta) = r\omega \sin \theta \left(1 + \frac{r \cos \theta}{l \sqrt{1 - \left(\frac{r}{l}\right)^2 \sin^2 \theta}}\right) \]

Note that \( v_A \) has been expressed explicitly in terms of \( \theta \) by substituting for \( \cos \beta \) and \( \tan \beta \). This has been done only for sake of clarity. When working with a
computer such substitutions will be automatic once $\beta$ has been defined in terms of $\theta$ ($\beta = \sin^{-1}(r \sin \theta / l)$).

In part (b) we first try to find the angle $\theta$ at which the maximum value of $v_A$ occurs by solving the equation $d v_A / d \theta = 0$ for $\theta$. Although a solution is obtained it is extremely messy and occupies many screen widths. For this reason we switch to an approximate approach which utilizes Matlab's `max` function. This method gives the maximum speed of the piston as $v_A = 69.6$ ft/sec at $\theta = 72.2^\circ$.

**MATLAB Worksheet and Scripts**

```
%%%%%%%%%%%%%%% Script #1 %%%%%%%%%%%%
% This script solves for $v_A$ using symbolic algebra
syms r L theta omega
vB = r*omega;
beta = asin(r/L*sin(theta));
vBA = cos(theta)/cos(beta)*r*omega;
vA = vB*sin(theta)+vBA*sin(beta)

%%%%%%%%%%%%% end of script %%%%%%%%%%%%

Output of script #1

$v_A =
\frac{r \omega \sin(\theta) + \cos(\theta) \left(1 - \frac{r^2}{L^2} \sin(\theta)^2\right)^{\frac{1}{2}} \frac{r^2 \omega}{L} \sin(\theta)}{1 - \frac{r^2}{L^2} \sin(\theta)^2}

%%%%%%%%%%%%% Script #2 %%%%%%%%%%%%
% This script plots the velocity of piston A
% as a function of theta
r = 5/12;
L = 14/12;
o = 1500*2*pi/60;  % o=omega
x = 0:0.01:2*pi;   % x=theta
vA = r*o*sin(x)+cos(x)./(1-r^2/L^2*sin(x).^2).^(1/2)*r^2*o/L.*sin(x);
plot(x*180/pi,vA)
xlabel('theta (degrees)')
title('velocity of piston A (ft/sec)')

%%%%%%%%%%%%% end of script %%%%%%%%%%%%
```
Part (b)

% This script solves for the values of theta which make \( v_A \) a maximum
r = 5/12;
L = 14/12;
omega = 1500*2*pi/60;
syms theta
% The following is copied and pasted from the output of % script #1
vA = r*omega*sin(theta)+cos(theta)/(1-r^2/L^2*sin(theta)^2)^(1/2)*r^2*omega/L*sin(theta)
dvAdtheta = diff(vA,theta)
solve(dvAdtheta,theta)

Output of script #3

\[
\begin{align*}
vA &= 125/6*\pi*sin(\theta)+625/84*cos(\theta)/(1-25/196*sin(\theta)^2)^(1/2)*\pi*sin(\theta) \\
dvAdtheta &= 125/6*\pi*cos(\theta)-625/84*sin(\theta)^2/(1-25/196*sin(\theta)^2)^(1/2)*\pi+15625/16464*cos(\theta)^2/(1-25/196*sin(\theta)^2)^(3/2)*\pi*sin(\theta)^2+625/84*cos(\theta)^2/(1-25/196*sin(\theta)^2)^(1/2)*\pi
\end{align*}
\]
ans = [output omitted]

The output was omitted because it was very messy and extended over several screen widths making it practically unusable. For some reason, these problems do not occur in Matlab 5. Due to these difficulties we try a different approach, using Matlab’s max function.

After running script #1 type the following into the command window,

EDU>> [vA_max, i] = max(vA)

vA_max =  
69.5514

i =  
127

EDU>> x(i)

ans =  
1.2600

EDU>> ans*180/pi

ans =  
72.1927

Thus, the maximum speed of the piston is $v_A = 69.6$ ft/sec at $\theta = 72.2^\circ$. 
5.4 Problem 5/123 (Relative Acceleration)

The two rotor blades of radius \( r = 800\text{-mm} \) rotate counterclockwise with a constant angular velocity \( \omega \) about the shaft at \( O \) mounted in the sliding block. The acceleration of the block is \( a_o \). Determine the magnitude of the acceleration of the tip \( A \) of the blade in terms of \( r, a_o, \omega, \) and \( \theta \). Plot the acceleration of \( A \) as a function of \( \theta \) for one revolution if \( a_o = 3 \text{ m/s} \). Consider three cases: \( \omega = 2, 4, \) and \( 6 \text{ rad/s} \).

Problem Formulation

The acceleration of \( A \) relative to \( O \) is

\[
\ddot{a}_A = \ddot{a}_O + (\ddot{a}_{A/O})_n + (\ddot{a}_{A/O})_t
\]

The acceleration of \( O \) is to the right while the normal relative acceleration must point from \( A \) towards \( O \). Since \( \omega \) is constant, the tangential relative acceleration will be zero. These considerations lead to the vector diagram shown to the right. Using the law of cosines,

\[
a_A = \sqrt{a_O^2 + (a_{A/O})_n^2 - 2a_O(a_{A/O})_n \cos \theta}
\]

\[
a_A = \sqrt{a_O^2 + (r\omega^2)^2 - 2a_O(r\omega^2)\cos \theta}
\]

MATLAB Script

```matlab
%======== script ==============================
aA = inline('sqrt(3^2+0.8^2*omega^4-2*3*0.8*omega^2*cos(theta))')
th = 0:.05:2*pi;
p.plot(th,aA(2,th),th,aA(4,th),th,aA(6,th))
xlabel('theta (radians)')
ylabel('acceleration (m/s^2)')
%======== end of script =======================
```
5.5 Sample Problem 5/15 (Absolute Motion)

The common configuration of a reciprocating engine is that of the slider crank mechanism shown. If crank $OB$ has a clockwise rotational speed of 1500 rev/min; (a) Plot $v_A$ and $v_G$ versus time for two revolutions of the crank. (b) Plot $a_A$ and $a_G$ versus time for two revolutions of the crank.

Problem Formulation

This problem appears in sample problems 5/9 and 5/15 in your text. Sample problem 5/9 considers a relative velocity analysis while sample problem 5/15 uses a relative acceleration analysis. Generally speaking, the easiest approach to use with a computer is an absolute motion analysis, provided you have software capable of doing symbolic algebra and calculus such as MATLAB. We will use the present problem to illustrate this approach.
We start by using the law of sines \((\sin(\beta)/r = \sin(\theta)/l)\) to express \(\beta\) as a function of \(\theta\)

\[
\beta = \sin^{-1}\left(\frac{r}{l} \sin \theta\right)
\]

where \(l\) is the length of connecting rod \(AB\) and \(\beta\) is the angle between \(AB\) and the horizontal. Now place an \(x\)-\(y\) coordinate system at \(O\) with \(x\) positive to the right and \(y\) positive up and write expressions for the coordinates of \(A\) and \(G\) in terms of \(\theta\) and \(\beta\)

\[
x_A = -r \cos \theta - l \cos \beta \\
x_G = -r \cos \theta - \bar{r} \cos \beta \\
y_G = (l - \bar{r}) \sin \beta
\]

where \(\bar{r}\) is the distance from \(B\) to \(G\) (4 in. in the figure). All that is needed to find the velocities \(v_A\), \(v_{Gx}\), and \(v_{Gy}\) is to differentiate these expressions with respect to time. The magnitude of the velocity of \(G\) is then found from

\[
v_G = \sqrt{v_{Gx}^2 + v_{Gy}^2}
\]

The accelerations \(a_A\), \(a_{Gx}\), and \(a_{Gy}\) are then found by differentiating \(v_A\), \(v_{Gx}\), and \(v_{Gy}\) with the magnitude of the acceleration of \(G\) being obtained from,

\[
a_G = \sqrt{a_{Gx}^2 + a_{Gy}^2}
\]

Since we will be differentiating with respect to time, the first thing we will do in the computer program is to define \(\theta\) as a function of time. Then, when we write the above expressions for \(\beta\), \(x_A\), \(x_G\), and \(y_G\), the computer will automatically substitute for \(\theta\) rendering each of these as functions of time. Assuming that \(\theta\) is initially zero,

\[
\theta(t) = \omega t = \frac{1500(2\pi)}{60} t = 157.1t
\]

The problem statement asks us to plot versus time for two revolutions (\(\theta = 4\pi\) radians) of the crank. The time required for two revolutions is \(4\pi/157.1 = 0.08\) sec.
MATLAB Worksheet and Scripts

%%%%%% Script #1 %%%%%%%
% This script uses symbolic algebra to find
% the velocity and acceleration of A and G
% w = constant angular velocity
% rb = r_bar (4 in. in the figure)
syms r rb L w t
theta = w*t;
beta = asin(r/L*sin(theta));
xA = -r*cos(theta)-L*cos(beta)
xG = -r*cos(theta)-rb*cos(beta)
yG = (L-rb)*sin(beta)
vA = diff(xA,t)
vGx = diff(xG,t)
vGy = diff(yG,t)
vG = sqrt(vGx^2+vGy^2)
aA = diff(vA,t)
aGx = diff(vGx,t)
aGy = diff(vGy,t)
aG = sqrt(aGx^2+aGy^2)

%%%%%% end of script %%%%%%%

Output of Script #1

xA = -r*cos(w*t)-L*(1-r^2/L^2*sin(w*t)^2)^(1/2)
xG = -r*cos(w*t)-rb*(1-r^2/L^2*sin(w*t)^2)^(1/2)
yG = (L-rb)*r/L*sin(w*t)
vA = r*sin(w*t)*w+1/L/(1-r^2/L^2*sin(w*t)^2)^(1/2)*r^2*sin(w*t)*cos(w*t)*w
vG = ((r*sin(w*t)*w+rb/(1-r^2/L^2*sin(w*t)^2)^(1/2)*r^2/L^2*cos(w*t)^2*w^2)^(1/2)+(L-rb)^2*r^2/L^2*cos(w*t)^2*w^2)^(1/2)
aA = r*cos(w*t)*w^2+1/L^3/(1-r^2/L^2*sin(w*t)^2)^(3/2)*r^4*sin(w*t)^2*cos(w*t)^2*w^2+1/L/(1-r^2/L^2*sin(w*t)^2)^(1/2)*r^2*sin(w*t)^2*w^2
aG = ((r*cos(w*t)*w^2+rb/(1-r^2/L^2*sin(w*t)^2)^(3/2)*r^4/L^4*sin(w*t)^2*cos(w*t)^2*w^2+rb/(1-r^2/L^2*sin(w*t)^2)^(3/2)*r^4/L^4*sin(w*t)^2*cos(w*t)^2*w^2)^(1/2)
\[
\begin{align*}
rb \left(1 - r^2/L^2 \sin(w^2t)^2\right)^{1/2} & \cdot r^2/L^2 \sin(w^2t)^2 \cdot w^2 \cdot \left\{ (L - rb)^2 \cdot r^2/L^2 \sin(w^2t)^2 \cdot w^4 \right\}^{1/2} \\
\end{align*}
\]

The results for the velocities and accelerations will be used to produce the required plots in the following two scripts. The easiest thing to do is to copy the results from the worksheet into the scripts. Before doing this you should use the `vectorize` command to place periods at appropriate places in order to ensure term by term rather than matrix operations. After running the script type “vectorize(vA)” in the worksheet and then copy and paste the result into the script. Repeat this process for vG, aA, and aG.

\texttt{%%%%%%%%%%%%%%%%%% Script #2 %%%%%%%%%%%%%%%%%%%%}
\texttt{" This script plots vA and vG versus time\}
\texttt{r = 5/12; \}
\texttt{rb = 4/12; \% r_bar\}
\texttt{L = 14/12; \}
\texttt{w = 1500*2*pi/60; \% w=omega\}
\texttt{t = 0:0.001:0.08; \}
\texttt{vA = r.*sin(w.*t).*w+1./L./(1-r.^2./L.^2.*sin(w.*t).^2).^\}
\texttt{(1./2).*r.^2.*sin(w.*t).*cos(w.*t).*w; \}
\texttt{vG = ((r.*sin(w.*t).*w+rb./(1-r.^2./L.^2.*sin(w.*t).^2).^\}
\texttt{(1./2).*r.^2./L.^2.*sin(w.*t).*cos(w.*t).*w).^2+(L-rb).^2.*r.^2./L.^2.*cos(w.*t).^2.*w.^2).^\}
\texttt{(1./2); \}
\texttt{plot(t, vA, t, vG \}
\texttt{" % end of script %%%%%%%%%%%%%%%%%%%%%%%%%~~~~~~~

\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{speed.plot.png}
\caption{Graph showing vA and vG versus time.}
\end{figure}
Script #3

% This script plots $a_A$ and $a_G$ versus time
r = 5/12;
rb = 4/12;
L = 14/12;
w = 1500*2*pi/60; % w=omega
t = 0:0.001:0.08;

$a_A = r.*\cos(w.*t).*w.^2+1./L.^3./(1-r.^2./L.^2.*
\sin(w.*t).^2).^(3./2).*r.^4.*\sin(w.*t).*^2.*\cos(w.*t).*^2.*w.
^2+1./L./(1-r.^2./L.^2.*\sin(w.*t).^2).^(1./2).*r.^2.*
\cos(w.*t).^2-1./L./(1-r.^2./L.^2.*\sin(w.*t).^2).^2.*w.
^2+1./L./(1-r.^2./L.^2.*\sin(w.*t).^2).^(1./2).*r.^2.*
\sin(w.*t).^2.*w.^2-1./L./(1-r.^2./L.^2.*\sin(w.*t).^2).^2.*w.
^2+1./L./(1-r.^2./L.^2.*\sin(w.*t).^2).^(1./2).*r.^2.*
\sin(w.*t).^2.*w.^2;

$a_G = ((r.*\cos(w.*t).*w.^2+rb./(1-r.^2./L.^2.*
\sin(w.*t).^2).^(3./2).*r.^4./L.^4.*\sin(w.*t).*^2.*\cos(w.*t).
^2.*w.^2+rb./(1-r.^2./L.^2.*\sin(w.*t).^2).^(1./2).*r.^2.*
\cos(w.*t).^2-1.*r.^2./L.^2.*\sin(w.*t).*^2.*w.^4).^(1./2);

plot(t, aA, t, aG)

end of script

/*!
For further study

Suppose you wanted to solve this problem for the case where crank $OB$ has a constant angular acceleration $\alpha = 60 \text{ rad/s}^2$. It turns out that you can solve this problem using exactly the same approach as above, changing only one line defining the dependence of $\theta$ upon time. Assuming that the system starts from rest at $\theta = 0$, the appropriate expression for $\theta$ is $\theta(t) = \frac{1}{2} \alpha t^2 = 30t^2$. The accelerations for this case are shown below in case you want to give it a try.
This chapter concerns the motion (translation and rotation) of rigid bodies that results from the action of unbalanced external forces and moments. In problem 6.1, five equations are solved for five unknowns. The problem illustrates an alternative to the blunt (but straightforward) simultaneous solution of multiple equations. Instead, the equations are solved in such a way that there is never more than one unknown. In this way, the results are immediately obtained via automatic substitution, thus avoiding some tedious algebra. The force at the hinge of a pendulum is plotted versus the angular position of the pendulum in problem 6.2. The algebra is rather simple in this case and MATLAB is used primarily for purposes of plotting. Problems 6.3 and 6.4 consider rigid bodies in general plane motion. `solve` is used in problem 6.3 to solve two equations simultaneously for two unknowns. The maximum acceleration of a point on the rigid body is then obtained by using `diff` and `solve`. In this problem, `solve` finds five solutions to the equation and it is necessary to determine which of these is physically correct. This problem is also interesting since a very natural “guess” of the value for the maximum acceleration turns out to be incorrect. Problem 6.4 is an example of a kinetics problem that also requires some kinematics. The angular acceleration of a bar is determined by summing moments. The kinematic equation $\ddot{\theta} = \alpha \dot{\theta}$ is then integrated to obtain the angular velocity. Problem 6.5 is an interesting work and energy problem that is complicated considerably by the fact that a spring is engaged for only part of the motion of a rotating bar. Symbolic algebra simplifies this problem considerably, though it is still rather tedious. It is common in Dynamics to find problems that require a combination of methods for their solution. Problem 6.6 is a good example involving both conservation of momentum and work/energy.
6.1 Sample Problem 6/2 (Translation)

The vertical bar $AB$ has a mass of $m = 150$ kg with center of mass $G$ midway between the ends. The bar is elevated from rest at $\theta = 0$ by means of the parallel links of negligible mass, with a couple $M$ applied to the lower link at $C$. Plot the force at $A$ and at $B$ as functions of $\theta$ (between 0 and $60^\circ$) for two cases, (a) a constant couple $M = 5$ kN-m, and (b) a constant angular acceleration $\alpha = 5$ rad/sec$^2$.

Problem Formulation

The free-body diagram (FBD) and mass acceleration diagram (MAD) are shown to the right. Since the vertical bar undergoes curvilinear translation, the acceleration of all points on the bar will be identical. Thus, we can obtain the acceleration of $G$ immediately from that of point $A$ which moves in a circular path about $C$. Also note that BD is a two force member since the mass is negligible.

Here we take a somewhat different approach than that in your textbook. The main reason for this is that the case of constant moment and constant angular acceleration require a slightly different approach. The difference between the two approaches is easiest seen by writing all the equations first.

From the free-body diagram of the connecting link $AC$,

$$[\Sigma M_C = 0] \quad M - \vec{r}A_i = 0 \quad (1)$$

As pointed out in the sample problem, the force and moment equations are identical to the equilibrium equations whenever the mass is negligible.

From the free-body diagram of the vertical bar,
\[
\Sigma M_A = m\ddot{a}d \quad \quad r_{AB} B \cos \theta = m\ddot{r} \omega^2 \cos \theta \quad r_{AG} + m\ddot{r} \alpha \sin \theta \quad r_{AG}
\]

(2)

\[
\Sigma F_i = m\ddot{a} \quad A_i - mg \cos \theta = m\ddot{r} \alpha
\]

(3)

\[
\Sigma F_n = m\ddot{a}_n \quad B - A_n + mg \sin \theta = m\ddot{r} \omega^2
\]

(4)

And, finally, the kinematics equation

\[
\int_0^\omega \alpha d\omega = \int_0^\theta \alpha d\theta
\]

(5)

At this point we have a total of 5 equations and 6 unknowns \((M, \alpha, \omega, A_t, A_n, \text{ and } B)\). In part (a), \(M\) is specified while in part (b) \(\alpha\) is given. In both cases we will have 5 equations and 5 unknowns; however, it will be necessary to solve no more than one equation at a time provided they are done in the right order. This is what yields a different procedure for parts (a) and (b).

**Part (a)**

With \(M\) known we can find \(A_t\) from Equation (1) and then substitute the result into Equation (3) to get \(\alpha\) as a function of \(\theta\).

\[
A_t = \frac{M}{r}
\]

\[
\alpha = \frac{A_t}{m\ddot{r}} - \frac{g}{r} \cos \theta
\]

As usual, notice that there is no need to make an explicit substitution of \(A_t\) into \(\alpha\). The computer will make such substitutions automatically. Now we substitute \(\alpha\) into Equation (5) and integrate,

\[
\frac{1}{2} \omega^2 = \int_0^\theta \left( \frac{A_t}{m\ddot{r}} - \frac{g}{r} \cos \theta \right) d\theta
\]

\[
\omega^2 = 2 \left( \frac{A_t}{m\ddot{r}} \theta - \frac{g}{r} \sin \theta \right)
\]

Finally, we can substitute into Equations (2) and (4) to find \(B\) and then \(A_n\).
\[ B = \frac{m \bar{r}_{AG}}{r_{AB} \cos \theta} \left( \omega^2 \cos \theta + \alpha \sin \theta \right) \]

\[ A_n = B + mg \sin \theta - m\bar{r} \omega^2 \]

\[ A = \sqrt{A_n^2 + A_i^2} \]

**Part (b)**

With \( \alpha \) known instead of \( M \) we have to take a different approach, starting with Equations (5) and (3),

\[ \omega^2 = 2 \int_0^\theta \alpha d\theta = 2 \alpha \left[ \theta \right]_0^\theta = 2\alpha \theta \]

\[ A_i = mg \cos \theta + m\bar{r} \alpha \]

Now we can substitute into Equations (1), (2) and (4) to find \( M, B \) and then \( A_n \).

\[ M = \bar{r} A_i \]

\[ B = \frac{m \bar{r}_{AG}}{r_{AB} \cos \theta} \left( \omega^2 \cos \theta + \alpha \sin \theta \right) \]

\[ A_n = B + mg \sin \theta - m\bar{r} \omega^2 \]

\[ A = \sqrt{A_n^2 + A_i^2} \]
MATLAB Scripts

%========== script for part a ==========

m = 0.15; g = 9.81; r = 1.5;
rAB = 1.8; rAG = 1.2; rAC = r;
M = 5;
theta = 0:0.1:60;
thr = theta*pi/180;
At = M/r;
alpha = At/m/r - g/r*cos(thr);
omsq = 2*(At*thr/m/r - g/r*sin(thr));
B = m*r*rAG*(omsq*cos(thr)+alpha*sin(thr))/rAB./cos(thr);
An = B + m*g*sin(thr)-m*r*omsq;
A = sqrt(At.^2 + An.^2);
plot(theta,B,theta,A)
xlabel('theta (degrees)')
ylabel('force (kN)')
title('part (a): constant M')
======= end of script=====================

--- end of script ---
%======== script for part b ==========
m = 0.15; g = 9.81; r = 1.5;
rAB = 1.8; rAG = 1.2; rAC = r;
alpha = 5;
theta = 0:0.1:60;
thr = theta*pi/180;
omsq = 2*alpha*thr;
At = m*g*cos(thr) + m*r*alpha;
M = r*At;
B = m*r*rAG*(omsq.*cos(thr)+alpha.*sin(thr))/rAB./cos(thr);
An = B + m*g*sin(thr)-m*r*omsq;
A = sqrt(At.^2 + An.^2);
plot(theta,B,theta,A)
xlabel('theta (degrees)')
ylabel('force (kN)')
title('part (b): constant alpha')
====== end of script================= 
6.2 Sample Problem 6/4 (Fixed-Axis Rotation)

The pendulum has a mass of 7.5 kg with a mass center at \( G \) and a radius of gyration about the pivot \( O \) of 295 mm. If the pendulum is released from rest when \( \theta = 0 \), plot the total force supported by the bearing at \( O \) along with its normal and tangential components as a function of \( \theta \). Let \( \theta \) range between 0 and 180°.

**Problem Formulation**

The free body and mass acceleration diagrams are identical to those in the sample problem. The main difference in approach is that we will obtain results at an arbitrary angle \( \theta \) rather than at 60°.

\[
\sum M_O = I_\theta \alpha = mg \bar{r} \cos \theta = mk_0^2 \alpha, \quad \alpha = \frac{g\bar{r}}{k_0} \cos \theta
\]

\[
[\alpha \omega = \alpha d\theta] \quad \int_0^\theta \alpha \omega = \frac{g\bar{r}}{k_0} \int_0^\theta \cos \alpha d\theta = \frac{g\bar{r}}{k_0^2} \sin \theta
\]

\[
\omega^2 = \frac{2g\bar{r}}{k_0^2} \sin \theta
\]

\[
[\sum F_n = m\bar{r} \omega^2] \quad O_n - mg \sin \theta = m\bar{r} \omega^2
\]

\[
[\sum F_t = m\bar{r} \alpha] \quad -O_t + mg \cos \theta = m\bar{r} \alpha
\]

After substituting for \( \alpha \) and \( \omega \) we have,

\[
O_n = mg \left(1 + \frac{\bar{r}^2}{k_0^2}\right) \sin \theta \quad \quad O_t = mg \left(1 - \frac{\bar{r}^2}{k_0^2}\right) \cos \theta
\]

The magnitude of the force at \( O \) is,

\[
O = \sqrt{(O_n)^2 + (O_t)^2}
\]
After substituting $r = 0.25$ m, $k_0 = 0.295$ m, $m = 7.5$ kg, and $g = 9.81$ m/s$^2$ all forces will be functions of $\theta$ only.

**MATLAB Script**

```matlab
%% Script
% This script plots the force at O and its components as functions of theta
rb = 0.25;
k0 = 0.295;
mg = 7.5*9.81;
theta = 0:0.01:pi;
On = mg*(1+2*(rb/k0)^2)*sin(theta);
Ot = mg*(1-(rb/k0)^2)*cos(theta);
O=sqrt(On.^2+Ot.^2);
% converts to degrees
td=180*theta/pi;
plot(td,O,td,On,td,Ot)
xlabel('theta (degs)')
title('Force at O (Newtons)')
```
6.3 Problem 6/98 (General Plane Motion)

The slender rod of mass \(m\) and length \(l\) is released from rest in the vertical position with the small roller at end \(A\) resting on the incline. (a) Determine the initial acceleration of \(A\) \((a_A)\) and plot \(a_A\) versus \(\theta\) for \(0 \leq \theta \leq 90^\circ\). (b) Determine the maximum value of \(a_A\) over this range and the angle \(\theta\) at which it occurs.

**Problem Formulation**

The free-body and mass acceleration diagrams are shown to the right. Shown on the mass acceleration diagram are the two components of the acceleration of the center of mass \(G\) obtained from the following kinematic relation,

\[
a_G = a_A + (a_{G/A})_n + (a_{G/A})_t
\]

where \((a_{G/A})_n = \frac{1}{2} \omega^2 = 0\), \((a_{G/A})_t = \frac{l}{2} \alpha\).

\[
\left[ \Sigma M_A = l \alpha + m \ddot{a}_n \right] \quad 0 = \frac{1}{12} ml^2 \alpha + m \frac{l}{2} \alpha \frac{l}{2} - ma_A \frac{l}{2} \cos \theta
\]

\[
\left[ \Sigma F_x = ma_n \right] \quad mg \sin \theta = m \left( a_A - \frac{l}{2} \alpha \cos \theta \right)
\]

These two equations can be solved simultaneously to give,

\[
\alpha = \frac{6(g/l) \sin \theta \cos \theta}{4 - 3 \cos^2 \theta} \quad a_A = \frac{4g \sin \theta}{4 - 3 \cos^2 \theta}
\]

At first, the answer to part (b) seems obvious. Intuitively, we would like to say that the maximum acceleration is \(a_A = g\) and occurs at \(\theta = 90^\circ\). But this intuition neglects the effects of the bar’s rotation upon the acceleration. As we will see below, the maximum acceleration is somewhat larger than \(g\).

The maximum acceleration is obtained in the usual manner. The orientation where the maximum occurs is first found by solving the equation \(da_A / d\theta = 0\). 

---

---

---
for $\theta$. This angle is then substituted back into the expression for $a_A$ to yield the maximum acceleration.

**MATLAB Worksheet and Script**

When using the *solve* command to solve simultaneous equations, be sure that the variables solved for are single characters. Thus, in the following we let $x = a_A$ and $y = \alpha$. Also remember that the equations solved must be in the form where the right hand side is zero. The equals sign is omitted.

```matlab
EDU» syms x y L theta g m
EDU» eqn1 = 1/12*m*L^2*y+m*L/2*y*L/2-m*x*L/2*cos(theta)
eqn1 = 1/3*m*L^2*y-1/2*m*x*L*cos(theta)
EDU» eqn2 = m*g*sin(theta)-m*(x-L/2*y*cos(theta))
eqn2 = m*g*sin(theta)-m*(x-1/2*L*y*cos(theta))
EDU» [x,y]=solve(eqn1,eqn2)
x = -4*g*sin(theta)/(-4+3*cos(theta)^2)
y = -6*cos(theta)*g*sin(theta)/L/(-4+3*cos(theta)^2)
```

The result for $x$ will be used in the script below to plot $a_A$ versus $\theta$. It is convenient to go ahead and do part (b) first as we have everything already set up.

```matlab
EDU» aA = subs(x,g,9.81)
aA = -981/25*sin(theta)/(-4+3*cos(theta)^2)
EDU» daA = diff(aA,theta)
daA =
-981/25*cos(theta)/(-4+3*cos(theta)^2)^2-5886/25*sin(theta)^2/
(-4+3*cos(theta)^2)^2*cos(theta)
EDU» solve(daA,theta)
ans =
 1/2*pi
  atan(1/6*3^(1/2)*6^(1/2))
 -atan(1/6*3^(1/2)*6^(1/2))
 -atan(1/6*3^(1/2)*6^(1/2)) + pi
 atan(1/6*3^(1/2)*6^(1/2)) - pi
```

MATLAB has found five solutions. Now we use the *eval* command to put these in a form easier to understand.
Only two of the five solutions are in the range from 0 to 90°. These two solutions are \( \frac{\pi}{2} \) (90°) and 0.6155 rad (35.3°). Substitution will reveal which is the maximum. Of course, we could also look at the plot below to see that the second solution corresponds to a maximum.

EDU» subs(aA,theta,.6155)
ans = 11.3276

EDU» subs(aA,theta,pi/2)
ans = 9.8100 % this result shouldn’t be surprising

Thus, \((a_A)_{\text{max}} = 11.33 \text{ m/s}^2\) when \(\theta = 35.3^\circ\).

%%%%%%%%%%%%%%%%%% Script %%%%%%%%%%%%%%%%%
% This script plots aA versus theta
% g = 9.81;
theta = 0:0.01:pi/2;
aA = -4*9.81*sin(theta)./(-4+3*cos(theta).^2);
plot(theta*180/pi, aA)
xlabel('theta (deg)')
ylabel('a_A (m/s^2)')
%%%%%%%%%%%%%%%% end of script %%%%%%%%%%%%
6.4 Problem 6/104 (General Plane Motion)

The uniform 12-ft pole is hinged to the truck bed and released from the vertical position as the truck starts from rest with an acceleration $a$. If the acceleration remains constant during the motion of the pole, derive an expression for the angular velocity $\omega$ in terms of $a$, $g$, and $L$ where $L$ is the length of the pole. Plot $\omega$ versus $\theta$ for $a = 3$ ft/s$^2$.

Problem Formulation

The free-body and mass acceleration diagrams are shown to the right. Shown on the mass acceleration diagram are the three components of the acceleration of the center of mass $G$ obtained from the following kinematic relation,

$$\mathbf{a}_G = \mathbf{a}_O + (\mathbf{a}_{G/O})_n + (\mathbf{a}_{G/O})_t$$

where $a_O = a$, $(\mathbf{a}_{G/O})_n = \bar{r} \omega^2$, $(\mathbf{a}_{G/O})_t = \bar{r} \alpha$, and $\bar{r} = L/2 = 6$ feet.

$$\sum M_O = \bar{r} \alpha + m \ddot{r}$$

Now we integrate the relation $\omega d\omega = d\theta$ to obtain,

$$\frac{1}{2} \omega^2 = \frac{3}{2L} \int_0^\theta (g \sin \theta + a \cos \theta) d\theta = \frac{3}{2L} (g(1 - \cos \theta) + a \sin \theta)$$

$$\omega = \sqrt{\frac{3}{L} (g(1 - \cos \theta) + a \sin \theta)}$$
MATLAB Script

L = 12; g = 32.2; a = 3;
theta = 0:0.01:pi/2;
omega = sqrt(3/L*(g*(1-cos(theta))+a*sin(theta)));
plot(theta*180/pi,omega)
xlabel('theta (deg)')
ylabel('omega (rad/s)')

end of script
6.5 Sample Problem 6/10 (Work and Energy)

The 4-ft slender bar weighs 40 lb with a mass center at B and is released from rest in the position for which \( \theta \) is essentially zero. Point B is confined to move in the smooth vertical guide, while end A moves in the smooth horizontal guide and compresses the spring as the bar falls. Plot the angular velocity of the bar and the velocities of A and B as a function of \( \theta \) from 0 to 90°. The stiffness of the spring is 30 lb/in.

**Problem Formulation**

From the figure to the right, the lengths \( CB \) and \( CA \) are \( 2\sin\theta \) and \( 2\cos\theta \) respectively. Using the instantaneous center C we can write the two velocities in terms of the angular velocity \( \omega \)

\[
v_A = CA \omega = 2\omega \cos\theta \quad v_B = CB \omega = 2\omega \sin\theta
\]

Now we need to divide the range for \( \theta \) into two distinct intervals depending upon whether or not the spring has been engaged. Since the velocities for A and B are known in terms of \( \omega \) and \( \theta \), we need to find only the angular velocity in these two intervals. From the diagram we see that A will first contact the spring at an angle \( \theta = \sin^{-1}(18/24) = 0.8481 \) rads (48.6°).

(a) Before the spring is engaged (\( \theta \leq 48.6^\circ \)).

\[
[T = \frac{1}{2} m\dot{\theta}^2 + \frac{1}{2} I\omega^2] \quad \Delta T = \frac{1}{2} \frac{40}{32.2} (2\omega \cos\theta)^2 + \frac{1}{2} \left( \frac{1}{12} \frac{40}{32.2} \right) \omega^2
\]

\[
\Delta T = 0.8282(4 - 3\cos^2\theta)\omega^2
\]

\[
[\Delta V_g = W\Delta h] \quad \Delta V_g = 40(2\cos\theta - 2) = 80(\cos\theta - 1)
\]

We now substitute into the energy equation \( U'_{1-2} = \Delta T + \Delta V_g = 0 \),
\[
0 = 0.8282(4 - 3\cos^2\theta)\omega^2 + 80(\cos\theta - 1), \text{ from which we find}
\]
\[ \omega = 9.829 \sqrt{\frac{1 - \cos \theta}{4 - 3\cos^2 \theta}} \]

(b) After the spring is engaged \((48.6 \leq \theta \leq 90^\circ)\). The kinetic and potential energies are the same as in part (a). At any angle \(\theta\), point \(A\) has moved \(2\sin \theta\) feet to the left. Thus, the spring is compressed by \(2\sin \theta - 18/12\) feet.

\[
[V_e = \frac{1}{2} kx^2] \quad \Delta V_e = \frac{1}{2} \left(30 \text{ lb/in}\right) \left(12 \text{ in/ft}\right) \left(2\sin \theta - \frac{18}{12}\right)^2 - 0
\]

\[
\Delta V_e = 180 \left(2\sin \theta - \frac{3}{2}\right)^2
\]

Again, we substitute into the energy equation \(U_{1-2}' = \Delta T + \Delta V_g + \Delta V_e = 0\),

\[
0 = 0.8282(4 - 3\cos^2 \theta)\omega^2 + 80(\cos \theta - 1) + 180 \left(2\sin \theta - \frac{3}{2}\right)^2
\]

\[\omega = 2.457 \sqrt{\frac{216\sin \theta - 16\cos \theta - 144\sin^2 \theta - 65}{4 - 3\cos^2 \theta}}\]

**MATLAB Scripts**

In the following, terms ending with "_a" are for part (a) where \(\theta\) is between 0 and 48.6\(^\circ\) while terms ending with "_b" are for part (b) where \(\theta\) is greater than 48.6\(^\circ\).

```
% This script solves the energy equation for omega
% for cases a and b

DT = 0.8282*(4-3*cos(theta)^2)*omega^2;
DVg = 80*(cos(theta)-1);
DVe = 180*(2*sin(theta)-3/2)^2;
U12_a = DT + DVg
U12_b = DT + DVg + DVe
omega_a = solve(U12_a,omega)
omega_b = solve(U12_b,omega)
```

% end of script
Output of script #1

\[
U_{12\_a} = (\frac{4141}{1250} - \frac{12423}{5000} \cos(\theta)^2)\omega^2 + 80\cos(\theta) - 80
\]

\[
U_{12\_b} = (\frac{4141}{1250} - \frac{12423}{5000} \cos(\theta)^2)\omega^2 + 80\cos(\theta) - 80 + 180(2\sin(\theta) - \frac{3}{2})^2
\]

\[
\omega_a = -\frac{200}{(-16564 + 12423 \cos(\theta)^2)\omega^2 \cdot 41410^{(1/2)} \cdot ((-4 + 3\cos(\theta)^2) \cdot (\cos(\theta)-1))^{(1/2)}}
\]

\[
\omega_b = 50\left(\frac{4141 + 12423 \sin(\theta)^2}{(41410 + 124230 \sin(\theta)^2) \cdot (16\cos(\theta) + 65 + 144\sin(\theta)^2 - 216\sin(\theta))}\right)^{(1/2)}
\]

It shouldn’t be surprising that MATLAB has found two solutions to the equation. As you can see, they have the same magnitude with one being positive and the other negative. The solution of interest is the positive one which turns out to be the second solution. This solution is copied and pasted into script #2.

\[
\omega_b = -\frac{50}{(4141 + 12423 \sin(\theta)^2)\omega^2 \cdot (-41410 + 124230 \sin(\theta)^2) \cdot (16\cos(\theta) + 65 + 144\sin(\theta)^2 - 216\sin(\theta))}\right)^{(1/2)}
\]

In this case, the first solution is positive.

*************** Script #2 ***************
% This script plots omega versus theta. Note that two range variables are used to divide the range for theta into the two regions.
% a = theta for theta < 48.6 deg. (.8481 rads)
% b = theta for theta > 48.6 deg.
% The results for omega are copied and pasted from the output of script #1
a = 0:0.001:0.8481;
b = 0.8481:0.001:pi/2;
omega_a = -200\left((-16564 + 12423 \cos(\theta)^2)\omega^2 \cdot 41410^{(1/2)} \cdot ((-4 + 3\cos(\theta)^2) \cdot (\cos(\theta)-1))^{(1/2)}}\right);
omega_b = 50\left((4141 + 12423 \sin(b)^2)\omega^2 \cdot (-41410 + 124230 \sin(b)^2) \cdot (16\cos(b) + 65 + 144\sin(b)^2 - 216\sin(b))}\right)^{(1/2)};
plot(a*180/pi, omega_a, b*180/pi, omega_b)
xlabel('theta (degrees)')
ylabel('omega (rad/sec)')
%%%%%%%%%%%%%%%% end of script %%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%% Script #3 %%%%%%%%%%%%%%%%%
% This script plots vA and vB versus theta.

a = 0:0.001:0.8481;
b = 0.8481:0.001:pi/2;
omega_a = -200./(-16564+12423*cos(a).^2)*41410^(1/2).*((-4+3*cos(a).^2).*(cos(a)-1)).^(1/2);
omega_b = 50./(4141+12423*sin(b).^2).*(-(41410+124230*sin(b).^2).*(16*cos(b)+65+144*sin(b).^2).*216*sin(b))).^(1/2);
vA_a = 2*omega_a.*cos(a);
vA_b = 2*omega_b.*cos(b);
vB_a = 2*omega_a.*sin(a);
vB_b = 2*omega_b.*sin(b);
plot(a*180/pi, vA_a, b*180/pi, vA_b, a*180/pi, vB_a, b*180/pi, vB_b)
xlabel('theta (degrees)')
ylabel('velocity (ft/sec)')
%%%%%%%%%%%%%%%% end of script %%%%%%%%%%%%%%%%%
For Further Study

A striking feature of the velocity curves above is the sudden change in shape when the spring is engaged at about $\theta = 49^\circ$. The details depend very much upon the relative magnitudes of the weight and spring constant. To illustrate, the figure below shows the angular velocity for several different values of the spring constant $k$.

Note that for stiff springs, the angular velocity goes to zero before reaching $\theta = 90^\circ$. The physical explanation for this is that, for a stiff spring, the bar will rebound before it reaches the horizontal position.
6.6 Problem 6/206 (Impulse/Momentum)

Determine the minimum velocity $v$ that the wheel may have and just roll over the obstruction. The centroidal radius of gyration of the wheel is $k$, and it is assumed that the wheel does not slip. Plot $v$ versus $h$ for three cases: $k = \frac{1}{2}, \frac{3}{4},$ and $1$ m. For each case take $r = 1$ m.

Problem Formulation

During Impact: Conservation of Angular Momentum

As usual, we neglect the angular impulse of the weight during the short interval of impact. With this assumption we have conservation of angular momentum about point $A$. Immediately before impact, the center of the wheel is not moving in a circular path about $A$ and we need to use the formula for general plane motion. Note that $I = mk^2$.

\[ H_A = \bar{I} \dot{\omega} + m \ddot{v}d = mk^2 \omega + mv(r - h) = mk^2 \frac{v}{r} + mv(r - h) \]

We will use primes to denote the state immediately after impact. Since the wheel now rotates about $A$ we can use the simpler formula $H_A = I_A \omega$. Note that, by the parallel axis theorem, $I_A = \bar{I} + mr^2 = m(k^2 + r^2)$.

\[ H_A' = I_A \omega' = \left(k^2 + r^2\right) \frac{v'}{r} \]

Setting $H_A = H_A'$ and solving yields,

\[ v' = v \left(1 - \frac{rh}{k^2 + r^2}\right) \]

After Impact: Work-Energy

\[ \Delta T + \Delta V_g = 0 = \frac{1}{2} I_A \left(\dot{\omega}'^2 - \omega^2\right) + mgh \]
\[
\frac{1}{2} m \left( k^2 + r^2 \right) \left( \frac{v'}{r} \right)^2 = mgh
\]

Substituting the result for \( v' \) into the above equation followed by simplification yields,

\[
v = \frac{r \sqrt{2gh(k^2 + r^2)}}{k^2 + r^2 - rh}
\]

**MATLAB Scripts**

%%% Script #1 %%%%%%%

% symbolic solution for \( v \)

```matlab
syms m g k r h v vp
Ib = m*k^2; IA = m*(k^2+r^2);
HA = Ib*v/r+m*v*(r-h);
HAp = IA*vp/r;
vp = solve(HA-HAp,vp) % conservation of angular momentum
eqn = 1/2*IA*vp^2/r^2-m*g*h; % work/energy
solve(eqn,v)
```

%%% end of script %%%%%%%

Output of script #1

\[
vp = v*(k^2+r^2-r*h)/(k^2+r^2)
\]

\[
ans =
(2*g*h*r^2+2*g*h*k^2)*(1/2)*r/(k^2+r^2-r*h)
-2*g*h*r^2+2*g*h*k^2)*(1/2)*r/(k^2+r^2-r*h)
\]

The first solution is the one we want.

%%% Script #2 %%%%%%%

% This script plots \( v \) versus \( h \) for the specified values of \( k \).
% The result for \( v \) below was obtained by substituting \( g = 9.81 \) and \( r = 1 \) in to the symbolic result found in script #1

\[
v = \text{inline}('4.429*sqrt(h+h*k^2)/(1+k^2-h)');
v = \text{vectorize}(v); \text{ puts "."'s in expression}

h = 0:0.01:1;
plot(h,v(h,1/2),h,v(h,3/4),h,v(h,1))
xlabel('h (m)')
ylabel('v (m/s) ')
```

%%% end of script %%%%%%%
This chapter presents a brief introduction to rigid body dynamics in three dimensions. In problem 7.1, the general 3-D motion of three connected bars is investigated. In particular, the angular velocities of two of the bars are plotted versus the length of the third bar. `solve` is used to solve four equations symbolically for four unknowns. In problem 7.2 we consider a bent plate rotating about a fixed axis. The problem illustrates a simplified version of what engineers might do in a real design situation. Two dimensions of the bent plate are left as variables and the objective of the problem is to find all suitable values of those dimensions which satisfy several constraints simultaneously. Here we illustrate a graphical approach to this type of design problem.
7.1 Sample Problem 7/3 (General Motion)

Crank \( CB \) rotates about the horizontal axis with an angular velocity \( \omega_1 = 6 \text{ rad/s} \), which is constant for a short interval of motion that includes the position shown. Link \( AB \) has a ball-and-socket fitting on each end and connects crank \( DA \) with \( CB \). Let the length of crank \( CB \) be \( d \) mm (instead of 100 mm as in the sample problem in your text) and plot \( \omega_2 \) and \( \omega_n \) as a function of \( d \) for \( 0 \leq d \leq 200 \text{ mm} \). \( \omega_2 \) is the angular velocity of crank \( DA \) while \( \omega_n \) is the angular velocity of link \( AB \).

**Problem Formulation**

Our analysis will follow closely that in the sample problem in your text.

\[
v_A = v_B + \omega_n \times r_{A/B}
\]

where

\[
v_A = 50 \omega_2 \hat{j} \quad v_B = 6d \hat{i} \quad r_{A/B} = 50 \hat{i} + 100 \hat{j} + d \hat{k}
\]

Substitution into the velocity equation gives

\[
50\omega_2 \hat{j} = 6d \hat{i} + \begin{vmatrix}
\hat{i} & \hat{j} & \hat{k} \\
\omega_{nx} & \omega_{ny} & \omega_{nz} \\
50 & 100 & d
\end{vmatrix}
\]

Expanding the determinant and equating the \( i \), \( j \), and \( k \) components yields the following three equations

\[
d(6 + \omega_{ny}) - 100\omega_{nz} = 0 \quad 50(\omega_2 - \omega_{nz}) + d\omega_{nx} = 0 \quad 2\omega_{nx} - \omega_{ny} = 0
\]

At this point we have three equations with four unknowns. As explained in the sample problem in your text, the fourth equation comes by requiring \( \omega_n \) to be normal to \( v_{A/B} \)

\[
\omega_n \cdot r_{A/B} = 50 \omega_{nx} + 100 \omega_{ny} + d\omega_{nz} = 0
\]

These four equations will be solved simultaneously for \( \omega_2 \), \( \omega_{nx} \), \( \omega_{ny} \), and \( \omega_{nz} \). Once this is done,
\[
\omega_n = \sqrt{\omega_{nx}^2 + \omega_{ny}^2 + \omega_{nz}^2}
\]

**MATLAB Scripts**

%%%%%%%%%% Script #1 %%%%%%%%%%%
% This script solves four equations for
% omega_2 (w) and the three components of
% omega_n (x, y, and z). The magnitude of
% omega_n is then found from its components.
syms w x y z d
eqn1 = d*(6+y)-100*z;
eqn2 = 50*(w-z)+d*x;
eqn3 = 2*x-y;
eqn4 = 50*x+100*y+d*z;
[w,x,y,z]=solve(eqn1,eqn2,eqn3,eqn4)
omega_n = sqrt(x^2+y^2+z^2);
omega_n = simplify(omega_n)
%%%%%%%%%% end of script %%%%%%%%%%%%

Output of script #1

\[
\begin{align*}
  w &= \frac{3}{50}d \\
  x &= -\frac{3d^2}{d^2+12500} \\
  y &= -\frac{6d^2}{d^2+12500} \\
  z &= \frac{750d}{d^2+12500} \\
  \omega_n &= 3\sqrt{2} \left( \frac{d^2}{d^2+12500} \right)^{1/2}
\end{align*}
\]

%%%%%%%%%% Script #2 %%%%%%%%%%%
% This script plots omega_2 and the magnitude of
% omega_n as functions of d
d = 0:0.5:200;
omega_2 = 3/50*d;
omega_n = 3*5^(1/2)*(d.^2/(d.^2+12500)).^(1/2);
plot(d, omega_2, d, omega_n)
xlabel('d (mm)')
title('angular velocity (rad/s)')
%%%%%%%%%% end of script %%%%%%%%%%%
7.2 Sample Problem 7/6 (Kinetic Energy)

The bent plate has a mass of 70 kg per square meter of surface area and revolves around the $z$-axis at the rate $\omega = 30$ rad/s. Let the dimensions of part $B$ be $a$ and $b$ where $a$ is the dimension parallel to the $x$-axis and $b$ is the dimension parallel to the $z$-axis. Part $A$ remains unchanged.

(a) Find all suitable values for $a$ and $b$ which satisfy the following conditions: $a \leq 0.2$ m, $b \leq 0.6$ m, and $15 \leq T \leq 30$ J where $T$ is the kinetic energy of the plate. (b) Find $a$ and $b$ for the case where $T = 40$ J and $H_0 = 5$ Nm·s where $H_0$ is the magnitude of the angular momentum about $O$.

**Problem Formulation**

Substitution of $\omega_x = 0$, $\omega_y = 0$, and $\omega_z = \omega$ into equations 7/11 and 7/18 of your text yields:

$$H_0 = \alpha(-I_{xz} \mathbf{i} - I_{yz} \mathbf{j} + I_{zz} \mathbf{k})$$

$$H_0 = \sqrt{H_{\omega x}^2 + H_{\omega y}^2 + H_{\omega z}^2} = \omega \sqrt{I_{xz}^2 + I_{yz}^2 + I_{zz}^2}$$
The moments and products of inertia for part \( A \) remain unchanged. For part \( B \), \( m_B = 70ab \) where \( a \) and \( b \) are in meters. The moments and products of inertia for part \( B \) are

\[
I_{zz} = \bar{I}_{zz} + md^2 = \frac{m_B}{12}a^2 + m_B \left( \frac{125}{2} \right)^2 + \left( \frac{a}{2} \right)^2 \\
I_{xx} = \bar{I}_{xx} + md_x d_z = 0 + m_B \left( \frac{b}{2} \right)^2 \\
I_{yy} = \bar{I}_{yy} + md_y d_z = 0 + m_B \left( \frac{b}{2} \right)^2
\]

The total moment and products of inertia are found by adding the above to those found for part \( A \) (see the sample problem in your text). After substituting for \( m_B \) and simplifying we have,

\[
I_{zz} = 0.00456 + 1.094ab + 23.33a^3b \\
I_{xx} = 17.5a^2b^2 \\
I_{yy} = 0.00273 + 4.375ab^2
\]

Substitution of these results will give \( H_0 \) and \( T \) as functions of \( a \) and \( b \).

**Part (a)**

The most efficient way to show the acceptable ranges for \( a \) and \( b \) is to find the required relationship between these two dimensions in order to satisfy the upper and lower bounds on \( T \). This is accomplished by substituting these bounds for \( T \) in the equation above and then solving that equation for \( b \) as a function of \( a \). To illustrate, consider the lower limit on \( T \) (15 J). Substituting \( T = 15 \) into the equation above gives

\[
T = 15 = \frac{1}{2} I_{zz} (30)^2 = 2.052 + 492.2ab + 10,500a^3b
\]

Solving for \( b \),

\[
b = \frac{0.078921}{a(3 + 64a^2)} \quad \text{(for } T = 15 \text{ J)}
\]

A similar result can be obtained for the upper limit,


\[ b = \frac{0.17035}{a\left(3 + 64a^2\right)} \]  
(for \( T = 30 \) J)

Plotting these two functions defines the acceptable regions for \( a \) and \( b \).

Part (b)

This is similar to (a) except that we solve two equations \((T = 40 \text{ and } H_0 = 5)\) simultaneously for two unknowns, \( a \) and \( b \). The result is \( a = 0.1211 \text{ m} \) and \( b = 0.4852 \text{ m} \).

**MATLAB Scripts**

```matlab
%%%%%%%%%%% Script #1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The first part of this script solves the equations  
% T = 15 and T = 30 symbolically for b in terms of a.  
% These two results are named b15 and b30 respectively.  
% When using the symbolic solve it is best to rewrite  
% the equations in the form expression = 0. The "=0"  
% is automatically understood by MATLAB and doesn't  
% have to be typed in. In the following, our two  
% expressions are named eqn1 and eqn2.

syms a b
mB = 70*a*b;
omega = 30;
Izz = mB/12*a^2+mB*(0.125^2+(a/2)^2)+0.00456;
Ixz = mB*a/2*b/2;
Iyz = mB*0.125*b/2+0.00273;
H0 = omega*sqrt(Ixz^2+Iyz^2+Izz^2);
T = 1/2*Izz*omega^2

eqn1 = T - 15;
eqn2 = T - 30;
b15 = solve(eqn1,b)
b30 = solve(eqn2,b)

% This part of the script solves the two equations  
% for part b.

eqn3 = T - 40;
eqn4 = H0 - 5;
[a,b]=solve(eqn3,eqn4)

%%%%%%%%%%% end of script %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

**Output of script #1**

\[ H_0 = 1/10000*(27562500000000*a^4*b^4+1722656250000*a^2*b^4+2149875000*a*b^2+25 \\
421854900000000000*a^6*b^2+4593750000000*a^4*b^2+19152000000*a^3*b+107 \\
666015625*a^2*b^2+897750000*a*b)^{(1/2)} \]
We see from the above that MATLAB has found ten solutions to our equations. Of these only the first and last are real. The first solution has negative values for $a$ and $b$ and thus can be excluded. This leaves only the last solution,

$$a = 0.1211 \text{ m; } b = 0.4852 \text{ m}$$

```matlab
%% Script #2 % This script plots the two expressions for b % obtained with script #1
a = 0.01:0.001:0.2;
b15 = 8632/109375./a./(64*a.^2+3);
b30 = 18632/109375./a./(64*a.^2+3);
plot(a, b15, a, b30)
xlabel('a (m)')
```
The two curves above represent the values of $a$ and $b$ for which $T$ is exactly 15 or 30 J. Thus, the acceptable values of $a$ and $b$ satisfying the condition $15 \leq T \leq 30$ J are all those combinations lying on or between the two curves.
This chapter considers an important class of Dynamics problems that involve linear or angular oscillations of a body or structure about some equilibrium position or configuration. Very few of the homework problems in your text require you to actually plot the oscillations of a body versus time. For this reason, all of the problems in this chapter will involve such plots. This is very useful in visualizing the time response of a vibrating system, especially for the case of damped or forced vibrations. Problem 8.1 looks at the effects of damping coefficient upon time response while Problem 8.3 considers the effects of initial conditions.
8.1 Sample Problem 8/2 (Free Vibration of Particles)

The 8-kg body is moved 0.2 m to the right of the equilibrium position and released from rest at time \( t = 0 \). Plot the displacement as a function of time for three cases, \( c = 8, 32, \) and \( 56 \) N\( \cdot \)s/m. The spring stiffness \( k \) is 32 N/m.

**Problem Formulation**

As in the sample problem, the natural circular frequency is \( \omega_n = \sqrt{k/m} = \sqrt{32/8} = 2 \) rad/sec. Now we find the damping ratio for each case (from \( \zeta = c/(2m\omega_n) \)) with the result \( \zeta = 0.25, 1, \) and 1.75 for \( c = 8, 32, \) and \( 56 \) N\( \cdot \)s/m respectively.

(a) \( \zeta = 0.25 \). Since \( \zeta < 1 \), the system is underdamped. The damped natural frequency is \( \omega_d = \omega_n\sqrt{1-\zeta^2} = 1.937 \) rad/sec. The displacement and velocity are

\[
\begin{align*}
x &= Ce^{-\zeta\omega_d t} \sin(\omega_d t + \psi) = Ce^{-t/2} \sin(1.937t + \psi) \\
\dot{x} &= -0.5Ce^{-t/2} \sin(1.937t + \psi) + 1.937Ce^{-t/2} \cos(1.937t + \psi)
\end{align*}
\]

From the initial conditions \( x_0 = 0.2 \) and \( \dot{x}_0 = 0 \) we find \( C = 0.207 \) m and \( \psi = 1.318 \) rad.

(b) \( \zeta = 1 \). For \( \zeta = 1 \), the system is critically damped. The displacement and velocity are

\[
\begin{align*}
x &= (A_1 + A_2 t)e^{-\omega_d t} = (A_1 + A_2 t)e^{-2t} \\
\dot{x} &= A_2e^{-2t} - 2(A_1 + A_2 t)e^{-2t}
\end{align*}
\]

From the initial conditions \( x_0 = 0.2 \) and \( \dot{x}_0 = 0 \) we find \( A_1 = 0.2 \) m and \( A_2 = 0.4 \) m/s.

(c) \( \zeta = 1.75 \). Since \( \zeta > 1 \), the system is overdamped. The displacement and velocity are

\[
\begin{align*}
x &= B_1e^{\left(-\zeta - \sqrt{\zeta^2 - 1}\right)\omega_d t} + B_2e^{\left(-\zeta + \sqrt{\zeta^2 - 1}\right)\omega_d t} = B_1e^{-0.628t} + B_2e^{-6.372t}
\end{align*}
\]
\[ \dot{x} = -0.628B_1 e^{-0.628t} - 6.372B_2 e^{-6.372t} \]

From the initial conditions \( x_0 = 0.2 \) and \( \dot{x}_0 = 0 \) we find \( B_1 = 0.222 \) m and \( B_2 = -0.0219 \) m/s.

**MATLAB Script**

```matlab
%% Script

t=0:0.01:5;
C = 0.207; psi = 1.318;
xa = C*sin(1.937*t+psi).*exp(-t/2);
A1 = 0.2; A2 = 0.4;
xb = (A1+A2*t).*exp(-2*t);
B1 = 0.222; B2 = -0.0219;
xc = B1*exp(-0.628*t)+B2*exp(-6.372*t);
x0 = 0.*t;
plot(t,xa,t,xb,t,xc,t,x0)
xlabel('time (sec)')
ylabel('x (m)')
```

%%% end of script

![Graph showing the time response of a vibration system with different damping coefficients.](image)
8.2 Problem 8/139 (Damped Free Vibration)

The mass of a critically damped system having a natural frequency $\omega_n$ is released from rest at an initial displacement $x_0$. (a) Determine the time $t$ required for the mass to reach the position $x = 0.1x_0$ if $\omega_n = 4$ rad/s. (b) Plot the non-dimensional displacement $x/x_0$ for $\omega_n = 2$, 4, and 8 rad/s.

Problem Formulation

Start with the equation for the critically damped case on page 606 of your text.

$$x = (A_1 + A_2 t)e^{-\omega_n t}$$

First we have to evaluate the constants in terms of $x_0$ using the initial conditions. This will require the derivative of $x$.

$$\dot{x} = A_2 e^{-\omega_n t} - (A_1 + A_2 t)\omega_n e^{-\omega_n t}$$

Now, substituting the initial conditions,

$$x(t = 0) = x_0 = A_1$$

$$\dot{x}(t = 0) = 0 = A_2 - A_1 \omega_n$$

These two equations give

$$A_1 = x_0 \quad \text{and} \quad A_2 = x_0 \omega_n$$

$$x = (x_0 + x_0 \omega_n t)e^{-\omega_n t}$$

or

$$\eta = (1 + \omega_n t)e^{-\omega_n t}$$

where $\eta$ is the non-dimensional displacement $x/x_0$.

Part (a) will require the solution to the equation
\[ 0.1 = (1 + 4t)e^{-4t} \]

For part (b) we will simply plot \( \eta \) for three different natural frequencies \( \omega_n \).

**MATLAB Worksheet and Script**

**Part (a)**

EDU>> \( f = '(1 + 4*t)*exp(-4*t) = 0.1' \)

\( f = (1 + 4*t)*exp(-4*t) = 0.1 \)

EDU>> solve(f,'t')

\( \text{ans} = \)

\[-2.4044468956330014201486061769676 \]
\[ .97243004246685726447599505623170 \]

Thus, the time \( t \) required for the mass to reach the position \( x = 0.1x_0 \) when \( \omega_n = 4 \) rad/s is \( t = 0.972 \) sec.

**Part (b)**

\%========= script for part b ========
eta = inline('\( (1 + \omega_n*t)\cdot\exp(-\omega_n*t) \)')
t = 0:0.05:3;
plot(t,eta(t,2),t,eta(t,4),t,eta(t,8))
xlabel('time (sec)')
title('non-dimensional displacement')
%======== end of script =========

\( \text{eta} = \)

\( \text{Inline function:} \)
\( \text{eta}(t,\omega_n) = (1 + \omega_n*t)\cdot\exp(-\omega_n*t) \)
8.3 Sample Problem 8/6 (Forced Vibration of Particles)

The 100-lb piston is supported by a spring of modulus \( k = 200 \text{ lb/in} \). A dashpot of damping coefficient \( c = 85 \text{ lb-sec/ft} \) acts in parallel with the spring. A fluctuating pressure \( p = 0.625 \sin(30t) \) (psi) acts on the piston, whose top surface area is 80 in\(^2\). Plot the response of the system for initial conditions \( x_0 = 0.05 \text{ ft} \) and \( \dot{x}_0 = 5, 0, \text{ and } -5 \text{ ft/sec} \).

**Problem Formulation**

The particular (steady state) solution was found in the sample problem in your text,

\[
x_p = X \sin(\omega t - \phi)
\]

where \( X = 0.01938 \text{ m}, \ \phi = 1.724 \text{ rad} \) and \( \omega = 30 \text{ rad/sec} \). Also from the sample problem, \( \omega_n = \sqrt{k / m} = 27.8 \text{ rad/sec} \) and \( \zeta = c / 2m\omega_n = 0.492 \).

The complete solution is found by adding the complementary (transient) and particular solutions. Since the system is underdamped (\( \zeta < 1 \)), the complementary solution is,

\[
x_c = Ce^{-\zeta\omega_d t} \sin(\omega_d t + \psi)
\]

where \( \omega_d = \omega_n \sqrt{1 - \zeta^2} = 24.2 \text{ rad/sec} \). The displacement of the system is

\[
x = x_c + x_p = Ce^{-\zeta\omega_d t} \sin(\omega_d t + \psi) + X \sin(\omega t - \phi)
\]

\[
x = Ce^{-13.68t} \sin(24.2t + \psi) + 0.01938 \sin(30t - 1.724)
\]

The velocity is found by differentiating \( x \),

\[
\dot{x} = C\omega_d e^{-\zeta\omega_d t} \cos(\omega_d t + \psi) - C\zeta\omega_n e^{-\zeta\omega_d t} \sin(\omega_d t + \psi) + X\omega \cos(\omega t - \phi)
\]

\[
\dot{x} = Ce^{-13.68t}(24.2\cos(24.2t + \psi) - 13.68\sin(24.2t + \psi)) + 0.5814 \cos(30t - 1.724)
\]

The constants \( C \) and \( \psi \) are found from the initial conditions,
$x_0 = 0.05 = C \sin \psi + X \sin \phi = C \sin \psi - 0.0192$

$\dot{x}_0 = C \omega_n \cos \psi - C \zeta \omega_n \sin \psi + X \omega \cos \phi = 24.2C \cos \psi - 13.7C \sin \psi - 0.0887$

The first equation can be solved for $C = 0.0692 / \sin \psi$. Substitution into the second equation gives $\psi$.

$$\psi = \tan^{-1} \left( \frac{1.675}{x_0 + 1.035} \right)$$

$$C = \frac{0.0692}{\sin \psi}$$

This yields the following values for $C$ and $\psi$.

$x_0 = 0.05 \text{ ft, } \dot{x}_0 = 5 \text{ ft/s, } C = 0.259 \text{ ft, } \psi = 0.271 \text{ rad}$

$x_0 = 0.05 \text{ ft, } \dot{x}_0 = 0 \text{ ft/s, } C = 0.081 \text{ ft, } \psi = 1.017 \text{ rad}$

$x_0 = 0.05 \text{ ft, } \dot{x}_0 = -5 \text{ ft/s, } C = -0.178 \text{ ft, } \psi = -0.400 \text{ rad}$

**MATLAB Script**

```matlab
% This script plots x versus t for the three sets of initial conditions
% x = inline('C*sin(24.2*t+psi)*exp(-13.68*t)+.01938*sin(30*t-1.724)');
x = vectorize(x);
t = 0:0.001:0.5;
plot(t,x(.259,.271,t),t,x(.081,1.017,t),t,x(-.178,-.4,t))
xlabel('t (sec)')
ylabel('x (ft)')
```

-------------------- end of script ------------------------
Notice how quickly the three cases converge to the steady state solution.