

# Introduction to Mathematica

Seth F. Oppenheimer

The purpose of this handout is to familiarize you with Mathematica. The Mathematics and Statistics Department computer lab is on the fourth floor of Allen Hall and is open most afternoons and evenings. All of your social security numbers have been put into the machines there so that you can log on.

What you will now see is a printout of an actual Mathematica session. Mathematica can act as a scientific calculator as in the examples below:

**2+3.6**

In order to evaluate this, you must hold the shift key and while holding the shift key, press enter or return.

5.6

**Sin[Pi/4]**

$$\frac{1}{\text{Sqrt}[2]}$$

Now that is an exact answer! We can use the notation %25 to indicate output 25. The command N[ ] yields a numerical approximation for what is in the square brackets. N[ , 100] would give that approximation to 100 places.

**%25**

$$\frac{1}{\text{Sqrt}[2]}$$

```
N[%25]
```

```
0.707107
```

```
N[%25,100]
```

```
0.707106781186547524400844362104849039284835937688474\  
0365883398689953662392310535194251937671638207864
```

```
N[Pi]
```

```
3.14159
```

```
N[Pi,30]
```

```
3.14159265358979323846264338328
```

Cool, huh? Case matters in Mathematica. All built in functions such as Sin, Cos, Exp, Sqrt, and so on are started with upper case letters. When you define your own functions, make sure you spell them with lower case letters. Let us define a function `f` that takes a number squares it and adds three to that square.

```
f[x_]:= x^2 + 3
```

We may now evaluate `f` at a variety of numbers:

```
f[2]
```

```
7
```

```
f[47]
```

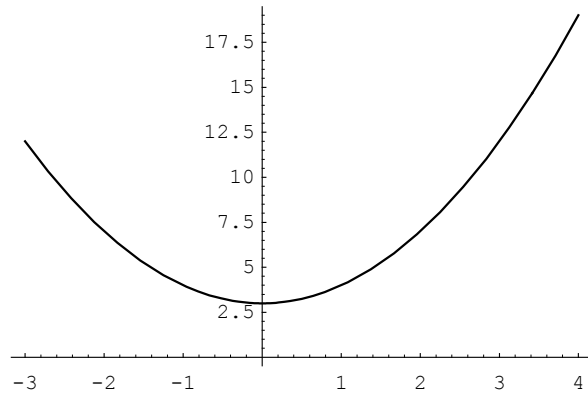
```
2212
```

```
f[-32]
```

```
1027
```

We can now plot `f`, differentiate it with respect to `x`, find the indefinite integral of `f` with respect to `x`.

```
Plot[f[x], {x, -3, 4}]
```



```
-Graphics-
```

```
D[f[x], x]
```

```
2 x
```

```
Integrate[f[x], x]
```

$$3x + \frac{x^3}{3}$$

Notice that you have to provide the arbitrary constant. Now we can take definite integrals as well, say from 1 to 3:

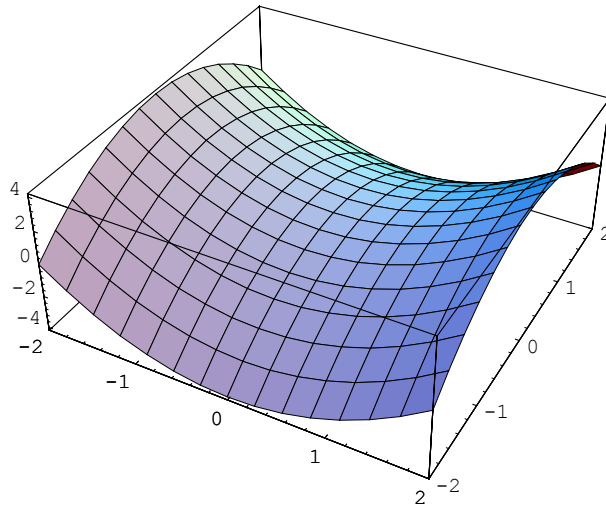
```
Integrate[f[x], {x, 1, 3}]
```

$$\frac{44}{3}$$

We can even do three dimensional graphs and parametric plots in one and two dimensions:

```
g[x_, y_] := x^2 - y^2
```

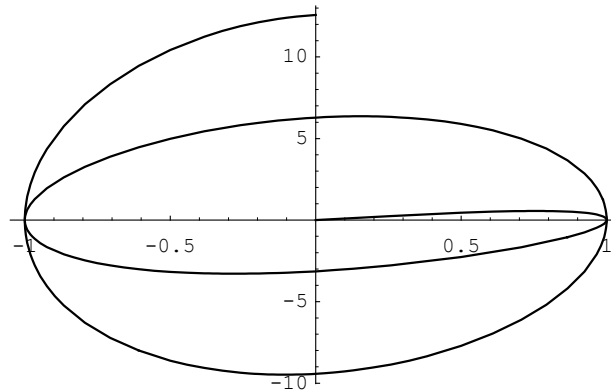
```
Plot3D[g[x,y],{x,-2,2},{y,-2,2}]
```



```
-SurfaceGraphics-
```

Now let us plot the parametric curve of  $(\sin[t], t \cos[t])$  as  $t$  goes from 0 to  $4\pi$ .

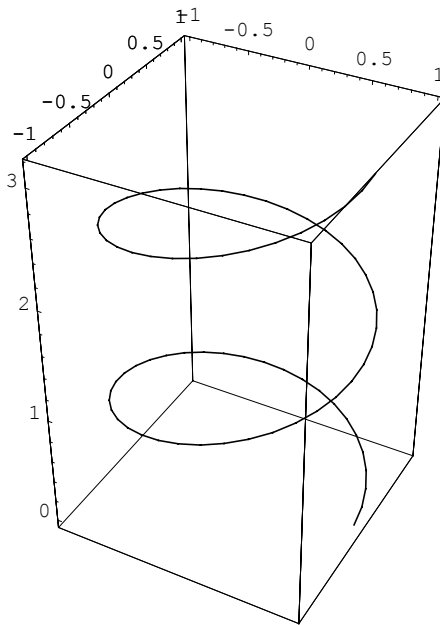
```
ParametricPlot[{Sin[t], t Cos[t]},{t,0,4 Pi}]
```



```
-Graphics-
```

Or how about the helix in three dimensions given by  $(\cos[4t], \sin[4t], t)$  for  $t$  from 0 to  $\pi$ ?

```
ParametricPlot3D[{Cos[4 t], Sin[4 t], t}, {t, 0, Pi}]
```



-Graphics3D-

It should be observed that there are all sort of settings that can be used for three dimensional graphing, from adjusting the scale to choosing the viewpoint. Look in the help index of the program for further information.

## ■ Differential Equations

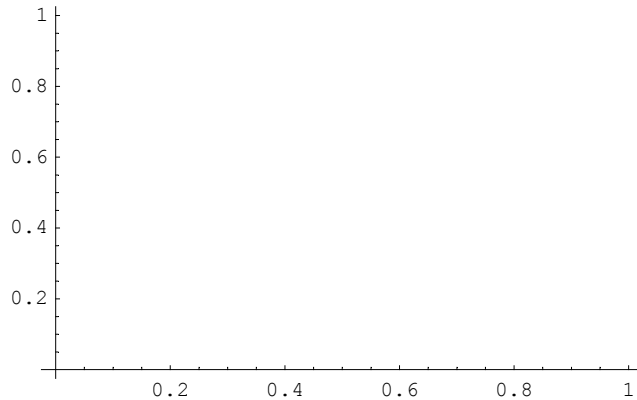
We can use *Mathematica* for differential equations. We will now learn the commands that will allow us to solve a single differential or a system of differential equations exactly (sometimes!) or numerically if necessary. The command to solve a differential equation or a system of differential equations is `DSolve`.

```
rule1=DSolve[{x'[t]==x[t], x[0]==1}, x[t], t]
{{x[t] -> e^t}}
```

Now what I have done is taken the ODE  $x'=x$ ,  $x(0)=1$  and solved for  $x(t)=\exp(t)$ . I have generated a rule, `rule1`, which will tell me how to evaluate the solution  $x$ . If I just ask for a graph of  $x[t]$ , the computer won't know what to do.

```
Plot[x[t], {t, 0, 2}]
```

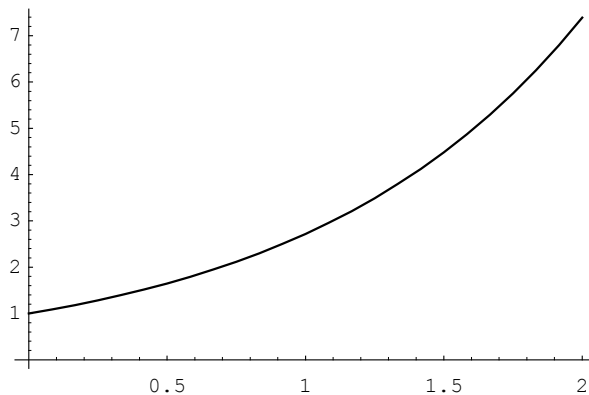
```
Plot::plnr : x[t] is not a machine-size real number at t = 8.333333333333333`*^-8.
Plot::plnr : x[t] is not a machine-size real number at t = 0.08113398314583158`.
Plot::plnr : x[t] is not a machine-size real number at t = 0.16961759971874735`.
General::stop : Further output of Plot::plnr will be suppressed during this calculation.
```



- Graphics -

I must tell the computer what rule to use to find  $x[t]$ .

```
Plot[Evaluate[x[t]/.rule1], {t, 0, 2}]
```

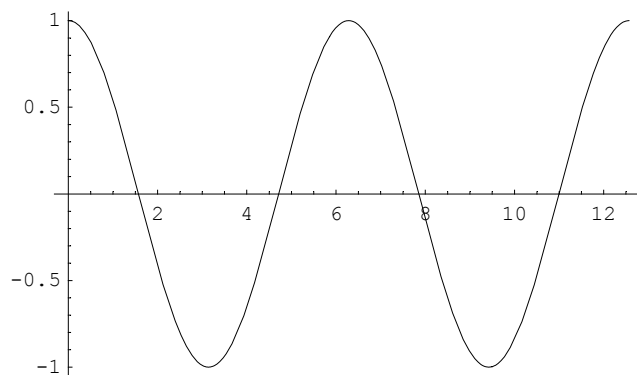


-Graphics-

We can do this with systems and higher order equations as well. First we will solve  $x''+x=0$  subject to  $x(0)=1$  and  $x'(0)=0$ :

```
rule2=DSolve[{x''[t]+x[t]==0,x[0]==1,x'[0]==0},x[t],t]
{{x[t] -> Cos[t]}}
```

```
Plot[Evaluate[x[t]/.rule2],{t,0,4 Pi}]
```



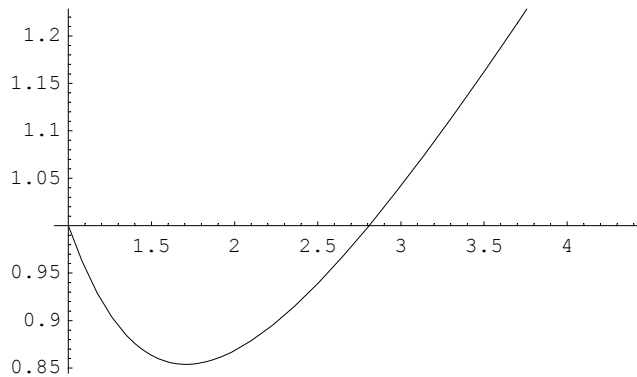
- Graphics -

Now we will do a system. Say  $x'=x+y$ ,  $y'=x-2y$ , subject to  $x(0)=1$  and  $y(0)=1$ .

```
rule3=DSolve[{x'[t]== x[t] + y[t], y'[t]== x[t] - (2 y[t]),
x[0]==1, y[0]==1},{x[t],y[t]},t]
```

$$\left\{ \left\{ \begin{aligned} x[t] &\rightarrow \frac{1}{26} \left( 13 e^{\frac{1}{2}(-1-\sqrt{13})t} - 5\sqrt{13} e^{\frac{1}{2}(-1-\sqrt{13})t} + 13 e^{\frac{1}{2}(-1+\sqrt{13})t} + 5\sqrt{13} e^{\frac{1}{2}(-1+\sqrt{13})t} \right), \\ y[t] &\rightarrow \frac{1}{26} \left( 13 e^{\frac{1}{2}(-1-\sqrt{13})t} + \sqrt{13} e^{\frac{1}{2}(-1-\sqrt{13})t} + 13 e^{\frac{1}{2}(-1+\sqrt{13})t} - \sqrt{13} e^{\frac{1}{2}(-1+\sqrt{13})t} \right) \end{aligned} \right\} \right\}$$

```
ParametricPlot[Evaluate[{x[t],y[t]}/.rule3],{t,0,1}]
```



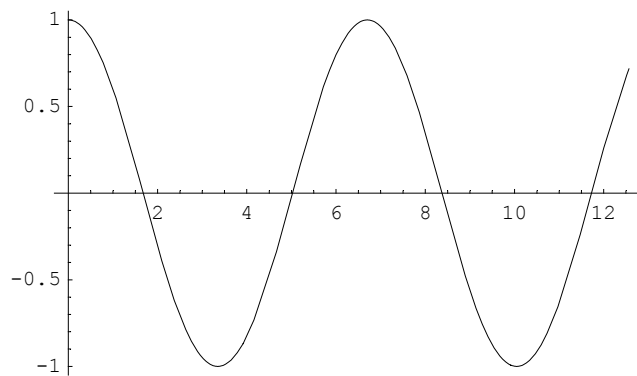
- Graphics -

Sometimes we can not find solutions to a differential equation or system of differential equations in terms of elementary functions. When this occurs, we use the numerical solution command, NDSolve. The only difference in terms of typing this command in and the DSolve command is that one must specify a range of the dependent variable over which to find an approximate solution:

```
rule4=NDSolve[{x'[t] + Sin[x[t]]==0,x[0]==1,x'[0]==0},
x[t],{t,0,4 Pi}]
```

```
{{x[t] → InterpolatingFunction[{{0., 12.5664}}, <>][t]}}
```

```
Plot[Evaluate[x[t]/.rule4],{t,0,4 Pi}]
```

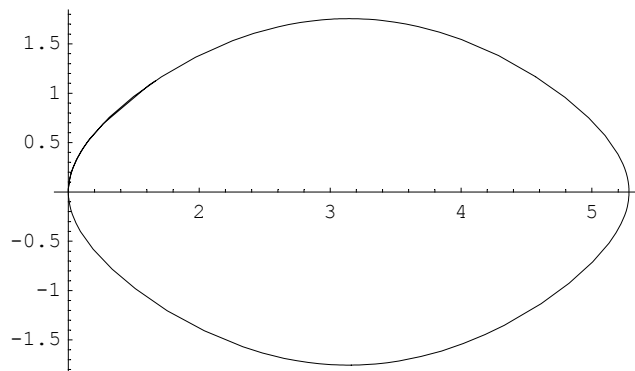


- Graphics -

```
rule5=NDSolve[{x'[t]==y[t],y'[t]==Sin[x[t]],x[0]==1,
y[0]==0},{x[t],y[t]},{t,0,10}]
```

```
{{x[t] → InterpolatingFunction[{{0., 10.}}, <>][t],
y[t] → InterpolatingFunction[{{0., 10.}}, <>][t]}}
```

```
ParametricPlot[Evaluate[{x[t], y[t]}/.rule5], {t, 0, 10}]
```



- Graphics -

## ■ Linear Algebra

We can use *Mathematica* to do matrix computations. We type in a matrix as a list of lists. The first list being the first row, the second list the second row, and so on. We will type in a matrix whose first row is a b and whose second row is c d. We will call the matrix amat.

```
amat = {{a, b}, {c, d}}
```

```
{{a, b}, {c, d}}
```

We can see what this looks like in standard form by using the command MatrixForm.

```
MatrixForm[amat]
```

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

We can find the transpose and inverse as well.

```
Transpose[amat]
```

```
{{a, c}, {b, d}}
```

```
MatrixForm[%]
```

$$\begin{pmatrix} a & c \\ b & d \end{pmatrix}$$

**Inverse[amat]**

$$\left\{ \left\{ \frac{d}{-b c + a d}, -\frac{b}{-b c + a d} \right\}, \left\{ -\frac{c}{-b c + a d}, \frac{a}{-b c + a d} \right\} \right\}$$

**MatrixForm[%]**

$$\begin{pmatrix} \frac{d}{-b c + a d} & -\frac{b}{-b c + a d} \\ -\frac{c}{-b c + a d} & \frac{a}{-b c + a d} \end{pmatrix}$$

We can do this with a numerical example as well.

**bmat = {{1, 2, 3}, {2, 3, 4}, {1, 2, 1}}**

{{1, 2, 3}, {2, 3, 4}, {1, 2, 1}}

**MatrixForm[bmat]**

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 1 & 2 & 1 \end{pmatrix}$$

**Transpose[bmat]**

{{1, 2, 1}, {2, 3, 2}, {3, 4, 1}}

**MatrixForm[%]**

$$\begin{pmatrix} 1 & 2 & 1 \\ 2 & 3 & 2 \\ 3 & 4 & 1 \end{pmatrix}$$

**Inverse[bmat]**

{{ $-\frac{5}{2}$ , 2,  $-\frac{1}{2}$ }, {1, -1, 1}, { $\frac{1}{2}$ , 0,  $-\frac{1}{2}$ }}

**MatrixForm[%]**

$$\begin{pmatrix} -\frac{5}{2} & 2 & -\frac{1}{2} \\ 1 & -1 & 1 \\ \frac{1}{2} & 0 & -\frac{1}{2} \end{pmatrix}$$

Determinants, eigenvalues, and eigenvectors can also be computed.

**Det[bmat]**

2

```
Eigenvalues[bmat]
```

```
{-1, 3 -  $\sqrt{11}$ , 3 +  $\sqrt{11}$ }
```

```
Eigenvectors[bmat]
```

```
{{-2, -1, 2}, {-2 -  $\sqrt{11}$ , 2, 1}, {-2 +  $\sqrt{11}$ , 2, 1}}
```

How do we handle matrix multiplication? We simply interpose a period between matrices.

```
cmat = {{1, 2, 1}, {2, 2, 5}, {4, 3, 2}}
```

```
{{1, 2, 1}, {2, 2, 5}, {4, 3, 2}}
```

```
MatrixForm[%]
```

```

$$\begin{pmatrix} 1 & 2 & 1 \\ 2 & 2 & 5 \\ 4 & 3 & 2 \end{pmatrix}$$

```

```
bmat . cmat
```

```
{{17, 15, 17}, {24, 22, 25}, {9, 9, 13}}
```

```
MatrixForm[%]
```

```

$$\begin{pmatrix} 17 & 15 & 17 \\ 24 & 22 & 25 \\ 9 & 9 & 13 \end{pmatrix}$$

```

We can also act on column vectors.

```
xmat = {{x}, {y}, {z}}
```

```
{{x}, {y}, {z}}
```

```
MatrixForm[xmat]
```

```

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

```

```
bmat . xmat
```

```
{ {x + 2 y + 3 z}, {2 x + 3 y + 4 z}, {x + 2 y + z} }
```

```
MatrixForm[%]
```

$$\begin{pmatrix} x + 2 y + 3 z \\ 2 x + 3 y + 4 z \\ x + 2 y + z \end{pmatrix}$$

We can solve a linear system without use inverses using the command `LinearSolve`. We will solve the system  $\mathbf{bmat} \cdot \mathbf{xmat} = \mathbf{bvec}$ , where  $\mathbf{bvec}$  is the column vector (1,2,3) transpose.

```
bvec = {{1}, {2}, {3}}
```

```
{{1}, {2}, {3}}
```

```
xmat = LinearSolve[bmat, bvec]
```

```
{{0}, {2}, {-1}}
```

```
MatrixForm[%]
```

$$\begin{pmatrix} 0 \\ 2 \\ -1 \end{pmatrix}$$

Other linear algebra commands can be found using the help index.

You may also find the tutorial by Dr. T. L. Miller helpful. It may be found on Quartz\labs\math\data\miller.

## ■ Difference Equations

We will consider difference equations in this section. We will assume that we can solve our  $n$ th order difference equation for the highest order term. For instance

$y[k+n]=f[k,y[k],y[k+1],\dots,y[k+n-1]]$ . Say we wish to program the fibonacci equation  $y[k+2] = y[k] + y[k+1]$ . Here  $n = 2$  and

$f[x,y,z] = y+z$ . Here is how we define this in Mathematica. To make Mathematica happy, we will set the highest order term as  $k$  and use subtraction as in  $y[k] = y[k-2] + y[k-1]$ . We need to give the initial values at  $k=0$  and  $k=1$ .

```
y[k_] := y[k] = y[k - 2] + y[k - 1]
y[1] = 1
y[2] = 1

1

1
```

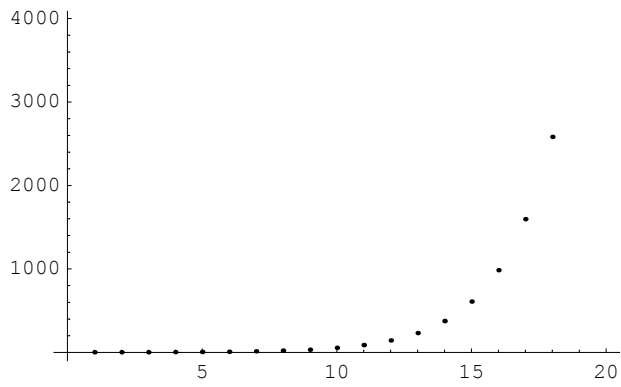
We now generate the first 20 terms using the Table command and put the list of values into the list variable list1.

```
list1 = Table[y[k], {k, 1, 20}]

{1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765}
```

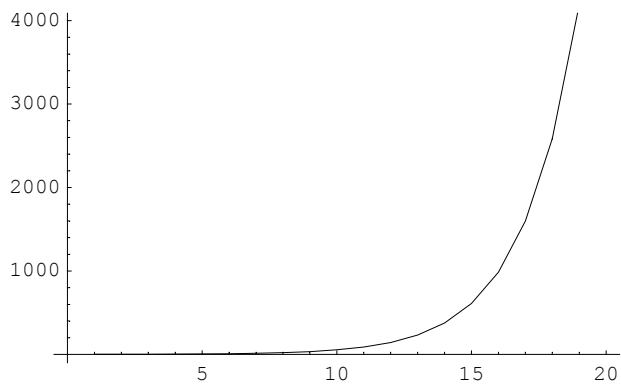
We now plot this with the ListPlot command. First we do the plot as discrete points and then we joint the points together.

```
ListPlot[list1]
```



- Graphics -

```
ListPlot[list1, PlotJoined -> True]
```



- Graphics -