# *Mathematica* Project Help

## Introduction

Throughout the semester, we will learn to use *Mathematica* to solve and visualize various problems related to MULTIVARIATE CALCULUS. First, we will mention some basic features of *Mathematica,* and then do some exercises with objects in the 3D coordinate system and vectors.

Let's consider the function f(x) = $x^3 + 2x^2 - 3$. We can enter functions into *Mathematica* with the syntax below.

```
f[x_] := x^3 + 2 x^2 − 3
```

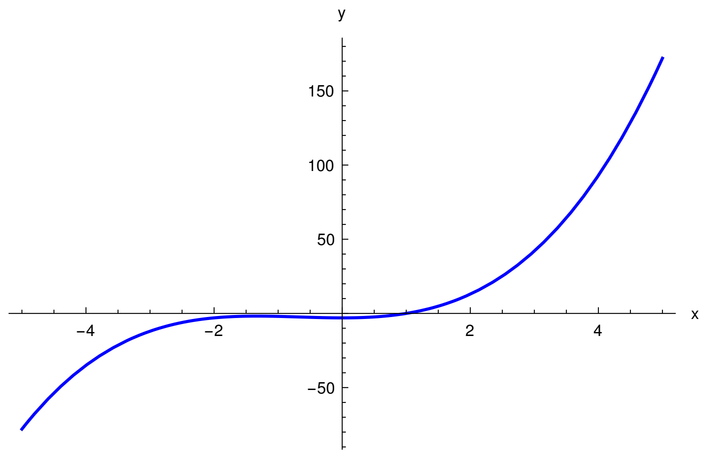
Notice that function arguments are ALWAYS placed inside square brackets. When we first define a function, each function argument is followed by an underscore. After we declare the function name and arguments, we follow the up with a semi-colon and an equal sign. This tells *Mathematica* that we are defining a new function. After this, we simply tell *Mathematica* what to do with the function arguments! We will learn some of the basic functions built into Mathematica as we go. Also, you have to press SHIFT + ENTER simultaneously to enter a command in *Mathematica*. If you only press ENTER, you will get a new input line, but *Mathematica* won't evaluate anything until SHIFT + ENTER are pressed together.

The first thing to note about *Mathematica* is that EVERY BUILT-IN FUNCTION STARTS WITH A CAPITAL LETTER. As mentioned above, the arguments for this function are placed inside square brackets (for instance, Sin[x], Cos[x], Exp[x], etc).

Notice the use of the equal sign. In *Mathematica*, = (a single equals) is used to assign values to variables. We have already seen the use of := to define functions. The double equals, ==, tells *Mathematica* to check whether one expression equals another. One of the most common mistakes in *Mathematica* programming is using a single = when you should really use ==. If you make this mistake, often the only thing to do is to correct it, save your file, close the program completely, and then reload your file.

We will now produce a (2D) plot of our function f. Conveniently, the command to produce a plot is Plot.

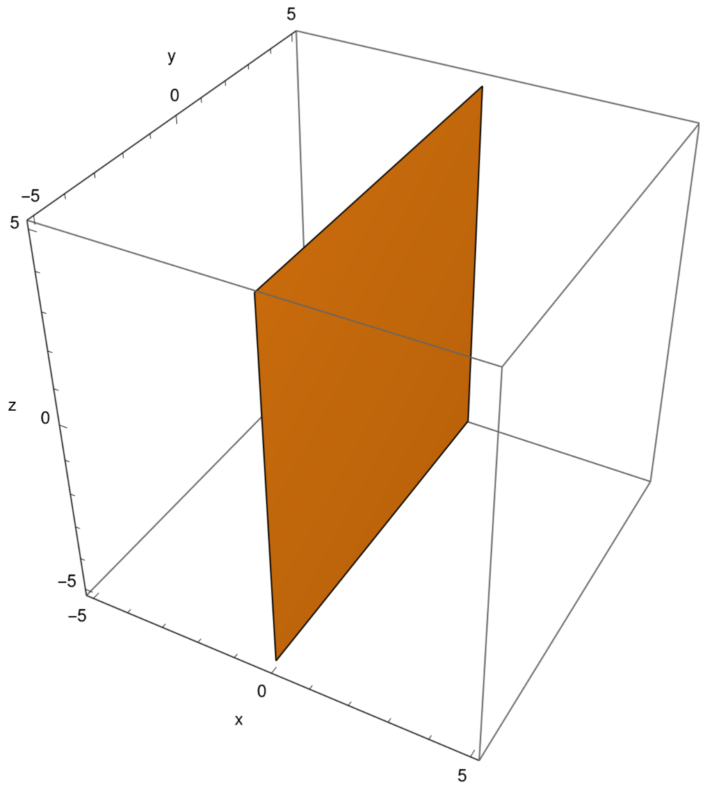
```
(* this is how you write a comment *)
Plot[f[x], {x, −5, 5}, PlotStyle → Blue, AxesLabel → {"x", "y"}]
```




Once again, EVERY BUILT-IN FUNCTION IN *Mathematica* STARTS WITH A CAPITAL LETTER. The first argument of Plot is the function (or functions) you want to plot. The second argument is a list containing the function variable followed by the a specification of the range you want plotted. Officially, these are the only arguments you need to give Plot to produce a graph. The arguments after the first two are optional. The first specifies the color of the plot while the second gives labels for the axes. Notice that these options are specified by replacement rules. The way to get the arrow symbol is to type a dash, - , directly followed by greater than, >. Mathematica will then convert -> to the arrow symbol. By the way, copying and pasting code into Mathematica often results in errors because the arrow (and other special symbols) may not paste correctly!

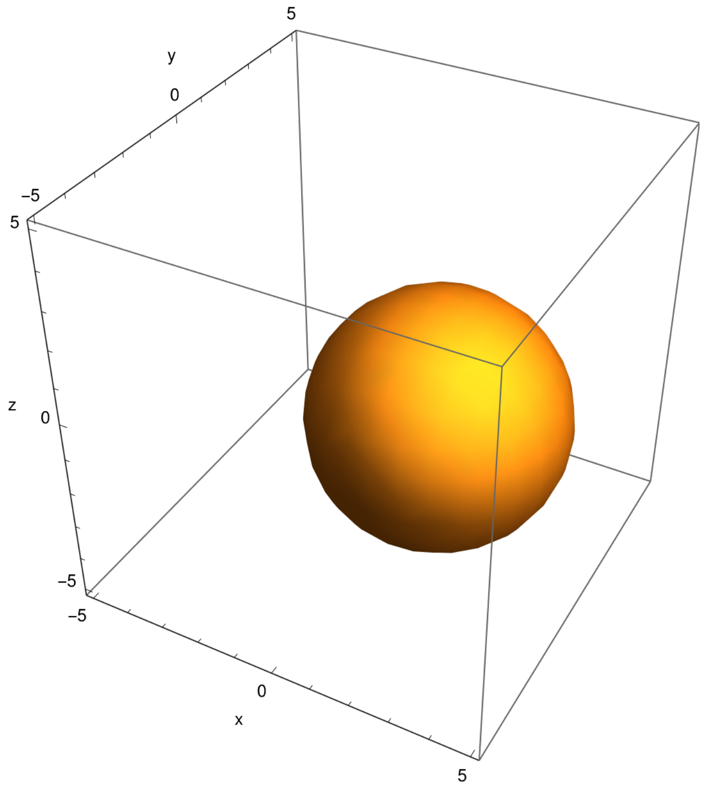
In our course, we will often use ContourPlot3D to generate 3D plot of functions of x, y, and z, or plot contour surfaces (covered later). For example, the following command produces the yz-plane. Note that the arguments after the equation “x==0” (notice the use of ==) and intervals for x, y, and z are optional:

```
ContourPlot3D[x == 0, {x, −5, 5}, {y, −5, 5}, {z, −5, 5}, Mesh → None, AxesLabel → {"x", "y", "z"}]
```



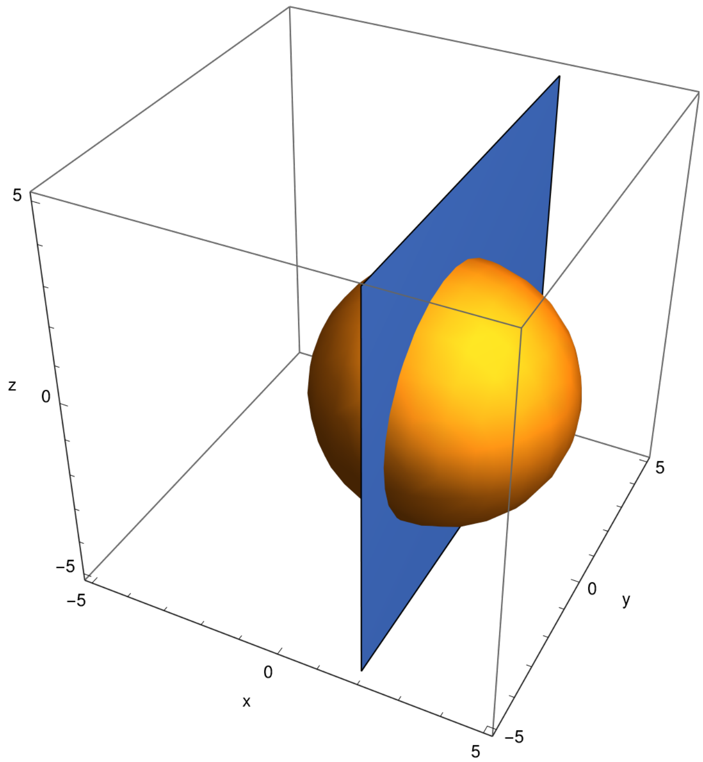

Next we recall the standard equation for a sphere: $(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = a^2$, and produce the plot of the sphere of radius 3 centered at (2,-1,0) in *Mathematica*:

```
ContourPlot3D[(x − 2)^2 + (y + 1)^2 + z^2 == 9, {x, −5, 5},
  {y, −5, 5}, {z, −5, 5}, Mesh → None, AxesLabel → {"x", "y", "z"}]
```




To plot more than one surface in the same figure, do the following (below you see the same sphere and the plane x=2 plotted together):

```
ContourPlot3D[{(x - 2)^2 + (y + 1)^2 + z^2 == 9, x == 2},
  {x, -5, 5}, {y, -5, 5}, {z, -5, 5}, Mesh → None, AxesLabel → {"x", "y", "z"}]
```




To compute the distance *d* between two points in space, for instance, P(3,2,1) and Q(2,-1,4), we can use the function EuclideanDistance (or one can type in the actual formula):

```
d = EuclideanDistance[{3, 2, 1}, {2, -1, 4}]
```

$\sqrt{19}$

To define a vector in *Mathematica* we use the format {x,y,z}. For example, vector u = <-4,5,6> is produced by

```
u = {-4, 5, 6}
```

{-4, 5, 6}

The length of u is found using the formula:

```
lu = Sqrt[(-4)^2 + 5^2 + 6^2]
```

$\sqrt{77}$

Let us find 3u, u+v, u-5v for vectors u=<5,4,6> and v = <4,3,-2>:

```
u = {5, 4, 6}
```

{5, 4, 6}

```
v = {4, 3, -2}
```

```
{4, 3, -2}
```

```
3 u
```

```
{15, 12, 18}
```

```
u + v
```

```
{9, 7, 4}
```

```
u - 5 v
```

```
{-15, -11, 16}
```

Now we will find the dot and cross products of u and v using the following commands in *Mathematica:*

```
u.v
```

```
20
```

```
Cross[u, v]
```

```
{-26, 34, -1}
```

Next let us find the angle $\theta$ between u and v:

theta = ArcCos$\left[\frac{u.v}{\sqrt{u.u}\ \sqrt{v.v}}\right]$

ArcCos$\left[\frac{20}{\sqrt{2233}}\right]$

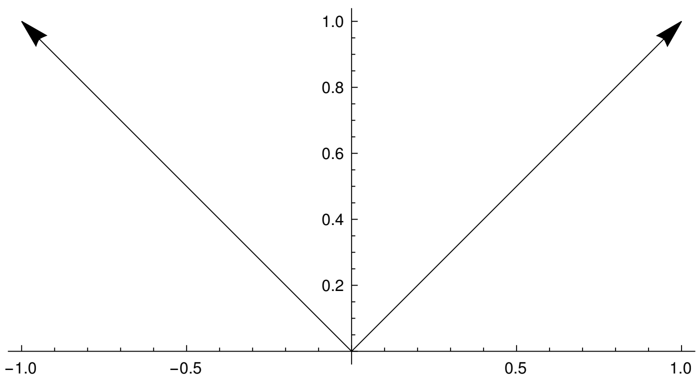
(Notice that to find the lengths of u and v for the formula, we used the fact that $u \cdot u = |u|^2$.)
We can also find the numerical value of the angle by using command N:

theta = N$\left[\text{ArcCos}\left[\frac{u.v}{\sqrt{u.u}\ \sqrt{v.v}}\right]\right]$

```
1.13378
```

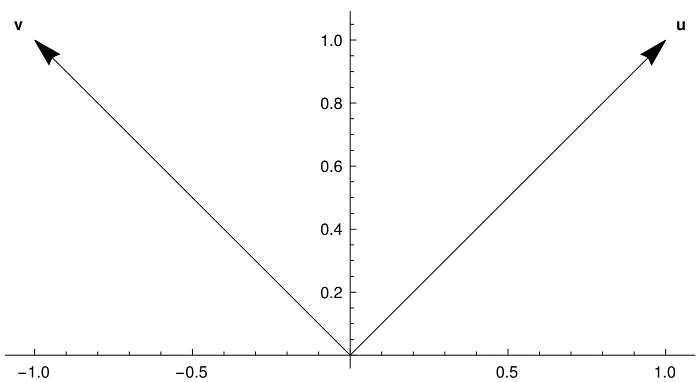
We will do some vector plotting now, using commands Graphics for 2D and Graphics3D in space combined with command Arrow:
First consider vectors u=<1,1> and v=<-1,1> on the plane. Recall that for vectors in component form, the initial point is the origin:

```
(* in 2D *)
Graphics[{Arrow[{{0, 0}, {1, 1}}], Arrow[{{0, 0}, {-1, 1}}]}, Axes → True]
```



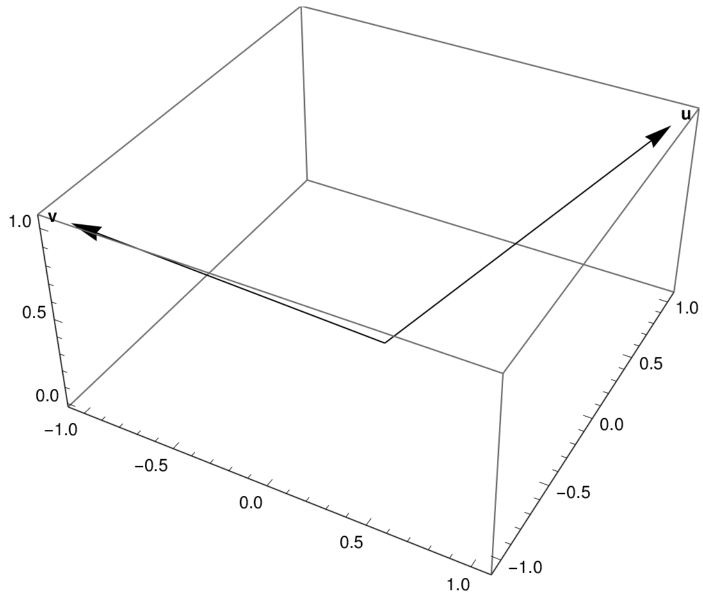

We can produce the same vectors adding the text:

```
Graphics[{Arrow[{{0, 0}, {1, 1}}], Text[Style["u", Bold], {1.05, 1.05}],
    Arrow[{{0, 0}, {-1, 1}}], Text[Style["v", Bold], {-1.05, 1.05}]}, Axes → True]
```




Now consider vectors u = <1,1,1> and v = <-1-1,1> in space:

```
(* in 3D *)
Graphics3D[{Arrow[{{0, 0, 0}, {1, 1, 1}}], Text[Style["u", Bold], {1.05, 1.05, 1.05}],
  Arrow[{{0, 0, 0}, {-1, -1, 1}}], Text[Style["v", Bold], {-1.05, -1.05, 1.05}]},
 Axes → True]
```

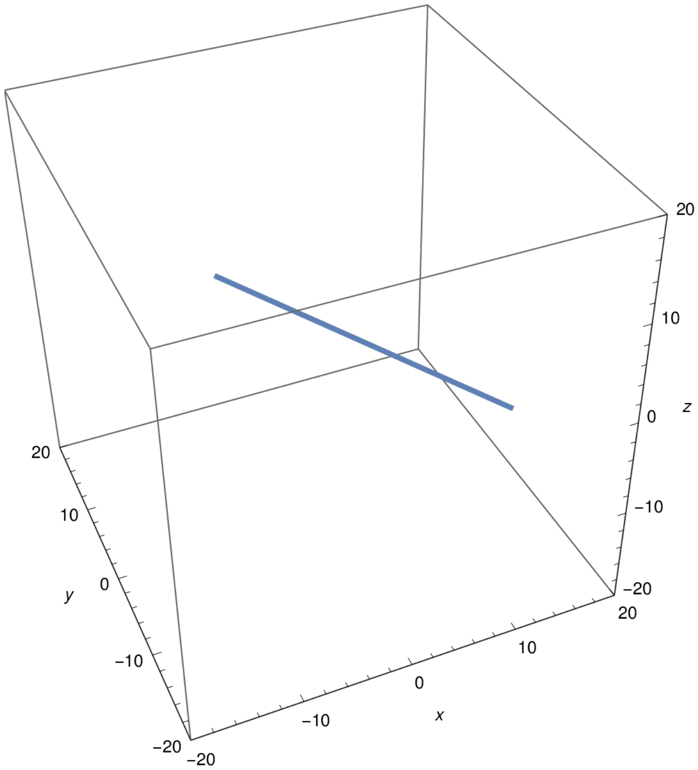



Make sure to explore the discussed functions/commands and their options.

## Lines and Planes

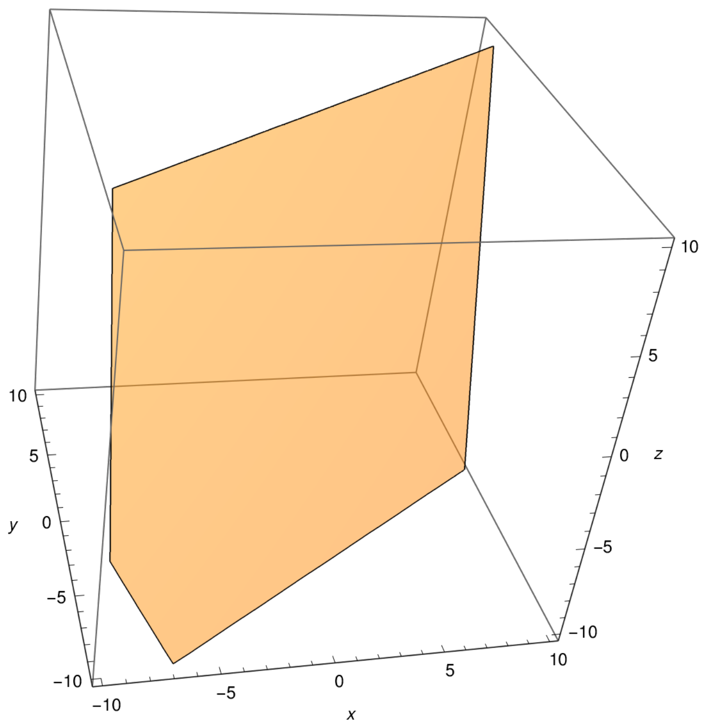
In *Mathematica*, one can plot a parametrically defined line in space using the function *Parametric-Plot3D:*

```
ParametricPlot3D [{2 + t, 3 - 4 t, t}, {t, -10, 10}, PlotRange → 20, AxesLabel → {x, y, z}]
```



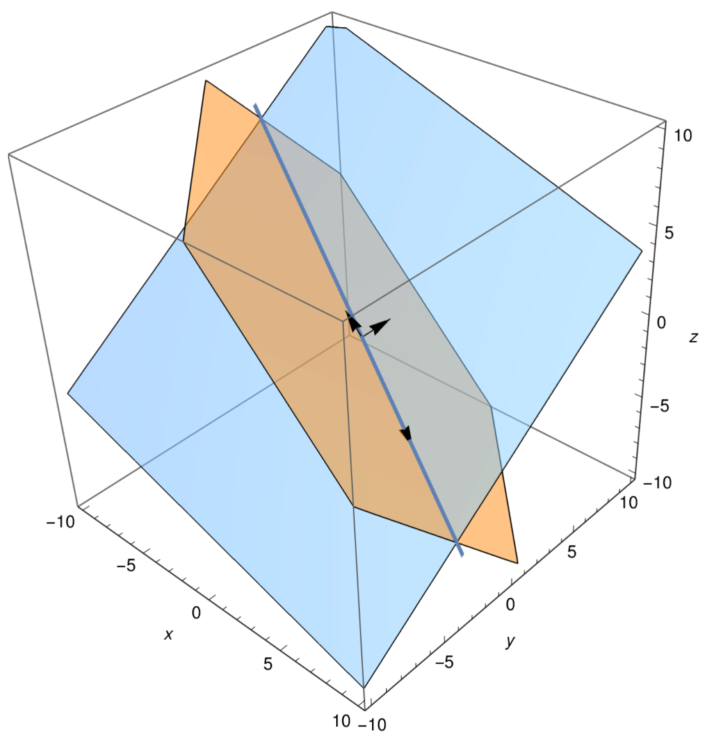

A plane in space can be given by a component equation A(x- $x_0$)+B(y-$y_0$)+C(z-$z_0$) = 0 or its simplified version Ax+By+ Cz=D using *ContourPlot3D*:

```
ContourPlot3D[2 x - 3 y + z == 6, {x, -10, 10}, {y, -10, 10}, {z, -10, 10},
 Mesh → None, ContourStyle → Opacity[0.5], AxesLabel → {x, y, z}]
```




In the next example, we have two intersecting planes and the line of intersection plotted. Command *Show* is used to combine the graphs in one figure.
Additionally, the normal vectors to the planes and a vector the line is parallel to are shown:

```
Show[
 ContourPlot3D[{x + y + z == 1, x - 2 y + 3 z == 1},
  {x, -10, 10}, {y, -10, 10}, {z, -10, 10}, AxesLabel → {x, y, z},
  Mesh → None, ContourStyle → Opacity[.5]],
 ParametricPlot3D[{1 + 5 t, -2 t, -3 t},
  {t, -10, 10}, PlotRange → 20, AxesLabel → {x, y, z}],
 Graphics3D[{Arrowheads[Small], Arrow[{{1, 0, 0}, {2, 1, 1}}],
   Arrow[{{1, 0, 0}, {2, -2, 3}}], Arrow[{{1, 0, 0}, {6, -2, -3}}]}]
]
```



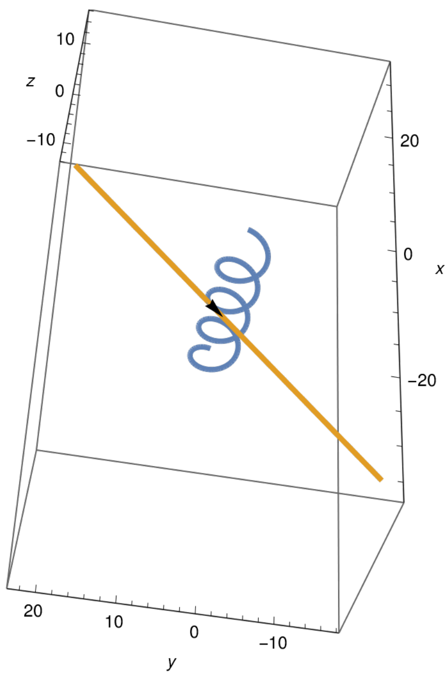

## Space Curves

Space curve parametrizations in Mathematica are defined using vector functions. The command below shows the syntax for defining the space curve
$\boldsymbol{r}_1$ = <2 cos t, 3 sin t, t>:

```
r1[t_] := {3 Cos[t], 3 Sin[t], t}
```

The tangent vector $\mathbf{dr_1/dt}$ (or the derivative of $\boldsymbol{r}_1$) is defined by $\boldsymbol{r}_1$'.
A parametrization of the tangent line to the curve $\boldsymbol{r}_1$ when t = $2\pi/3$ is determined by the point (2 cos $2\pi/3$, 3 sin $2\pi/3$, $2\pi/3$) and tangent vector
Below is the graph of the curve together with the tangent line and tangent vector at t = $2\pi/3$.

```
TangentLine[t_] := r1[2 Pi / 3] + t * r1'[2 Pi / 3] (*vector equation of the tangent line*)
Show[
 (*tangent line*)
 ParametricPlot3D[{r1[t], TangentLine[t]}, {t, -4 Pi, 4 Pi}, AxesLabel → {x, y, z}],
 (*tangent vector*)
 Graphics3D[{Arrowheads[.03], Arrow[{r1[2 Pi / 3], r1[2 Pi / 3] + r1'[2 Pi / 3]}]}]
]
```




Next, let's compute the arc length of the curve $\boldsymbol{r}_1$ as t increases from 0 to $2\pi/3$ using command *Integrate:*

```
Integrate[Sqrt[r1'[t].r1'[t]], {t, 0, 2 Pi / 3}]
```

$$\frac{2\sqrt{10}\,\pi}{3}$$

To find the numerical value of the integral, use command *NIntegrate:*

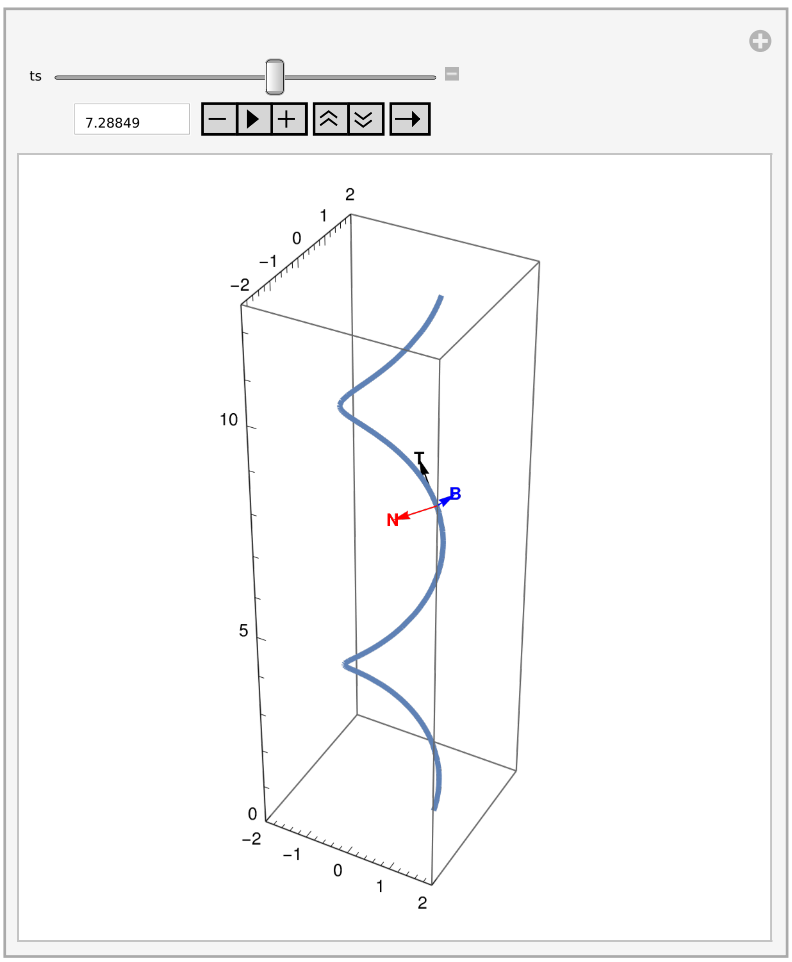
```
NIntegrate[Sqrt[r1'[t].r1'[t]], {t, 0, 2 Pi / 3}]
```

6.62306

## The TNB-frame for a Space Curve

Consider the space curve $\boldsymbol{r}_2$ = <cos t, sin t, t>. Let us plot the curve and the TNB-frame in dynamics using *Manipulate:*

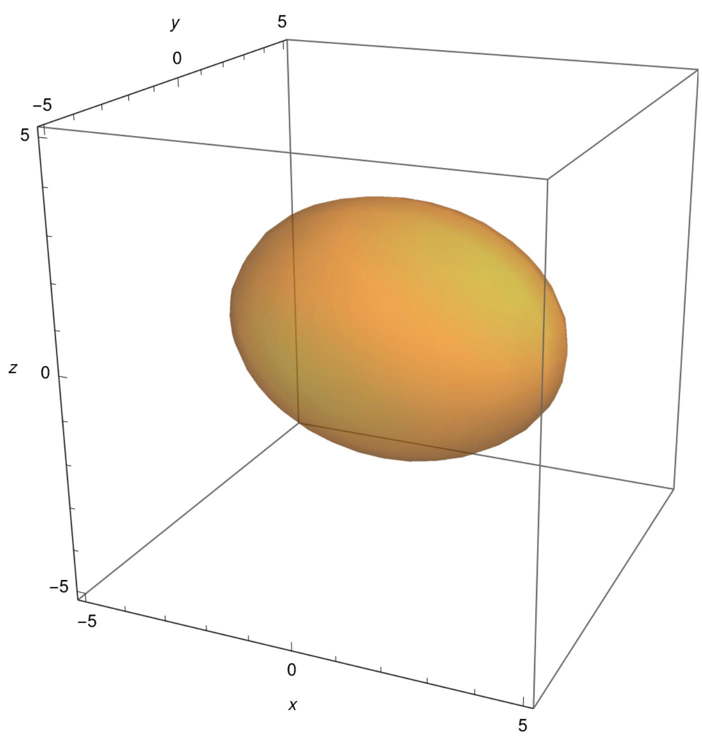
```
r2[t_] := {Cos[t], Sin[t], t}(* our curve is a helix *)
rT[t_] := r2'[t]/Sqrt[r2'[t].r2'[t]] (* T, the unit tangent to the curve *)
rN[t_] := rT'[t]/Sqrt[rT'[t].rT'[t]](* N, the principal unit normal vector *)
rB[t_] := Cross[rT[t], rN[t]] (* B, the binormal unit vector *)

Manipulate[
 Show[
  ParametricPlot3D [r2[t], {t, 0, 4 Pi}, PlotRange → {{-2, 2}, {-2, 2}, {0, 4 Pi}}],
  Graphics3D [{Arrowheads [.025],
            Arrow[{r2[ts], r2[ts] + rT[ts]}], Text[Style["T", Bold], {r2[ts] + rT[ts]}],
            Red, Arrow[{r2[ts], r2[ts] + rN[ts]}], Text[Style["N", Bold], {r2[ts] + rN[ts]}],
            Blue, Arrow[{r2[ts], r2[ts] + rB[ts]}], Text[Style["B", Bold], {r2[ts] + rB[ts]}]
   }]
 ],
 {ts, 0, 4 Pi}
]
```



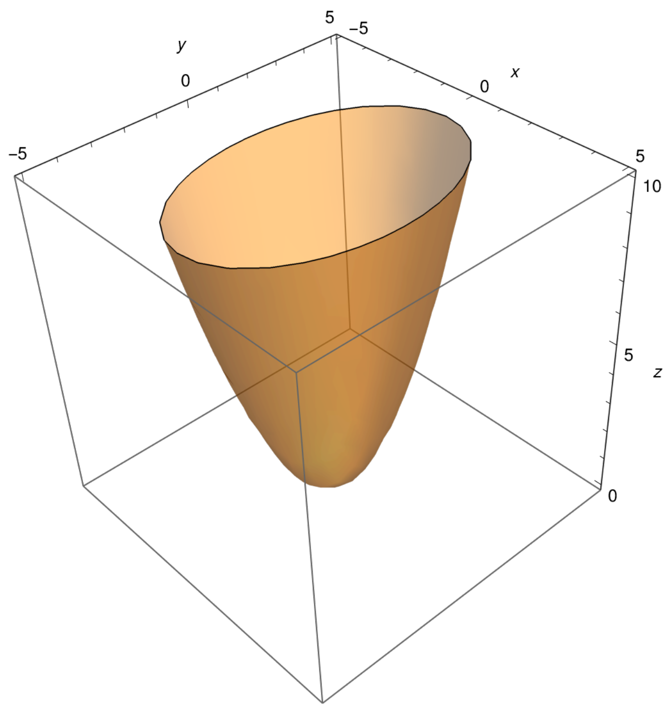
## Graphing Quadrics

Below are several examples of quadrics plotted using *ContourPlot3D.* (Find more examples in the demo notebook on quadrics posted on our web page.)

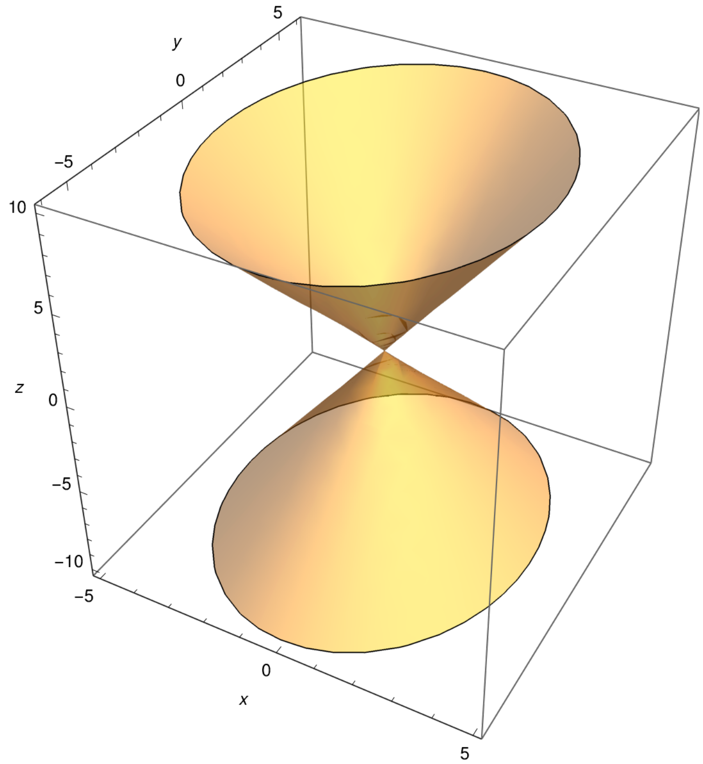
```
(* ellipsoid *)
ContourPlot3D [(x / 4) ^ 2 + (y / 2) ^ 2 + (z / 3) ^ 2 == 1,
 {x, -5, 5}, {y, -5, 5}, {z, -5, 5}, AxesLabel → {x, y, z},
 Mesh → None, ContourStyle → Opacity[.5]]
```



```
(* elliptical paraboloid *)
ContourPlot3D[(x / 2)^2 + (y / 3)^2 == z / 5,
 {x, -5, 5}, {y, -5, 5}, {z, 0, 10}, AxesLabel → {x, y, z},
 Mesh → None, ContourStyle → Opacity[.5]]
```



```
(* elliptical cone *)
ContourPlot3D[(x / 2)^2 + (y / 3)^2 == (z / 5)^2,
 {x, -5, 5}, {y, -6, 6}, {z, -10, 10}, AxesLabel → {x, y, z},
 Mesh → None, ContourStyle → Opacity[.5]]
```




Note:

*Examples above as well as demo notebooks posted on our course page will help with your first Mathematica project.*

*Good luck!*