

Plotting Three-Dimensional Surfaces Using *Mathematica*

In this exercise, we explore some of *Mathematica*'s capabilities for plotting multivariable functions.

Mathematica Primer:

Let's get started. Launch *Mathematica* on your computer. You should see a blank window called "Untitled". This is your *Mathematica* notebook. A *Mathematica* notebook is an interactive environment. You type in a command (or sequence of commands) and instruct *Mathematica* to execute it (or them).

Saving your work early and often. Go to the File pulldown menu and select "Save As..." Navigate to the folder where you want to save your file and type in a file name. *Mathematica* notebook files are denoted by the extension ".nb". As you continue working, save your file often by selecting "Save" under the File pulldown menu.

Type in $1 + 1$ followed by Shift-Enter to execute the command on Windows (on a Mac, you execute a command by pressing Enter or Shift-Return). You will see the following:

```
In[1]:= 1 + 1
```

```
Out[1]= 2
```

Looking closely at what happened, each command you type is echoed on the screen in boldface Courier font, preceded by `In[n]:=`, where n is the number of the command in the notebook. *Mathematica*'s response appears in plain Courier font, preceded by `Out[n]:=`. Note the brackets on the right side of the window. The leftmost set denotes input and output "cells," and the rightmost bracket is a "grouping bracket."

To edit the contents of an input cell and regenerate the output, place the cursor in the cell, edit accordingly, and enter the command. To delete a cell altogether, click on the appropriate bracket and hit the delete key.

The basic arithmetic operations of addition, subtraction, multiplication and division are indicated by +, -, *, and /, respectively. Multiplication can also be done by omitting the * and leaving a space between two numbers, in which case *Mathematica* will add a `x` to indicate that multiplication was performed. Exponentiation is indicated by ^.

If you are performing more than one arithmetic operation per command, you should be aware that there is a precedent in the order of operations: left to right, exponentiation before multiplication and division, before addition and subtraction. If you want to be sure that the operations are carried out in a definite order, you should use parentheses to group terms.

Mathematica has thousands of built-in functions. The name of the function is case-sensitive and the argument of the function is enclosed in square brackets. A very useful function is the numerical approxi-

mation function, $N[x,d]$, which gives a decimal approximation of the argument x to d significant digits. If you don't specify d , the default will be six significant digits. You can also insert comments into input cells between the delimiters (* and *):

A few examples are shown below:

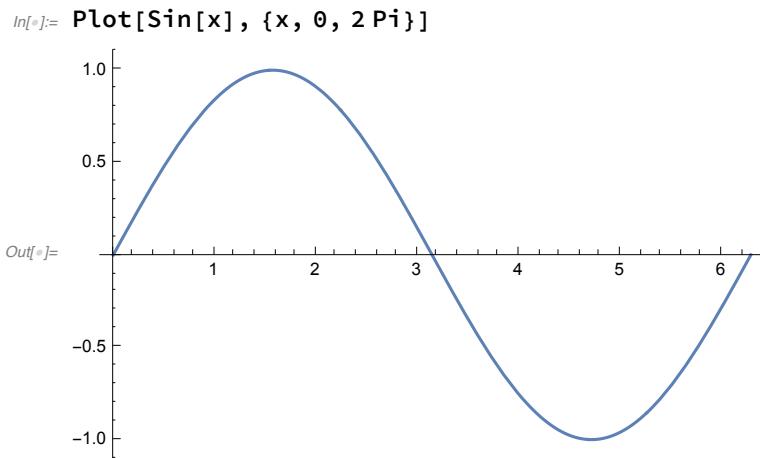
```
In[]:= Sin[Pi]
Out[]= 0

In[]:= Sqrt[3]
Out[]= √3

In[]:= N[Sqrt[3], 4] (* output 4 digits*)
Out[]= 1.732
```

Many more are evident in the Basic Math Assistant palette, and even more can be found in the Documentation.

Mathematica is very useful for making plots. To make a plot of functions of one variable using the **Plot** command, specify the function to be plotted and define its argument, and the range of the argument over which we want the function to be plotted. The syntax is illustrated by the following example, in which we plot the sine of x from $x = 0$ to $x = 2\pi$:



Now on to plotting 3D surfaces with Mathematica:

Consider a quantum mechanical problem of a particle in a two-dimensional box, i.e. quantum particle confined in two dimension between “walls” that correspond to regions of infinite potential energy, with wavefunction

$$\psi_{n_1,n_2}(x,y) = \left(\frac{2}{L}\right) \sin\left(\frac{n_1\pi x}{L}\right) \sin\left(\frac{n_2\pi y}{L}\right)$$

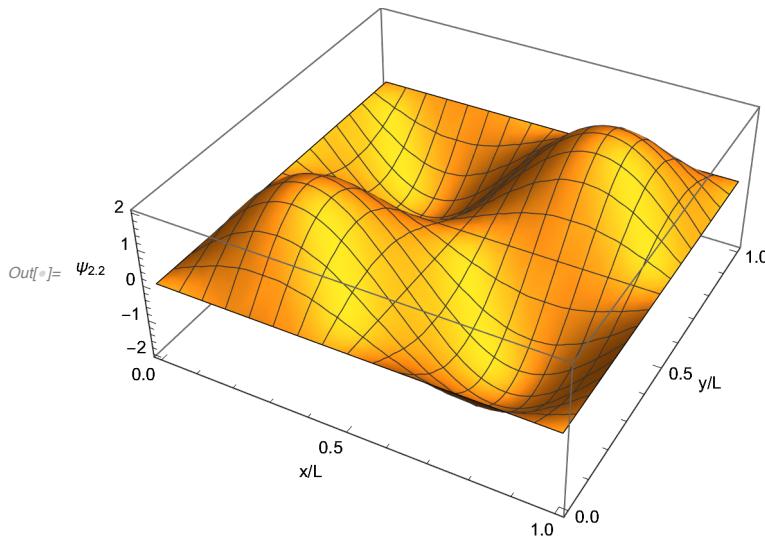
where L is the length of the box (assumed to be the same in the x and y directions, and n_1 and n_2 are quantum numbers that are restricted to the positive integers (1, 2, 3, ...). The solution of the Schrödinger equation for this system gives the quantized energies:

$$E_{n_1, n_2} = \left(n_1^2 + n_2^2 \right) \left(\frac{\hbar^2}{8 m L^2} \right)$$

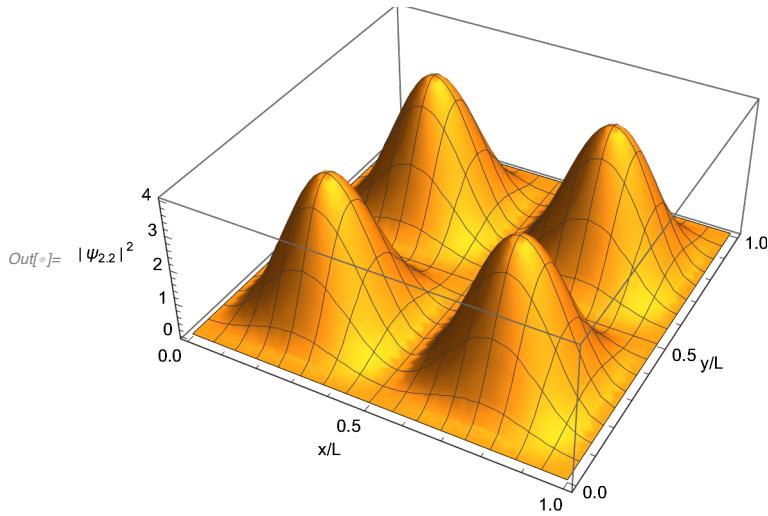
where m is the mass of the particle and \hbar is Plank's constant. According to the Born interpretation of the wavefunction, $|\psi|^2$ is the probability density of the quantum particle, i.e., $|\psi|^2 dx dy$ is the probability of finding the particle within $dx dy$ of the point (x, y) .

In the following, we will plot several wavefunctions and the corresponding probability densities with x and y in units of L (i.e., we set $L = 1$ in the above equation) using the **Plot3D** command, which is analogous to the Plot command for functions of one variable:

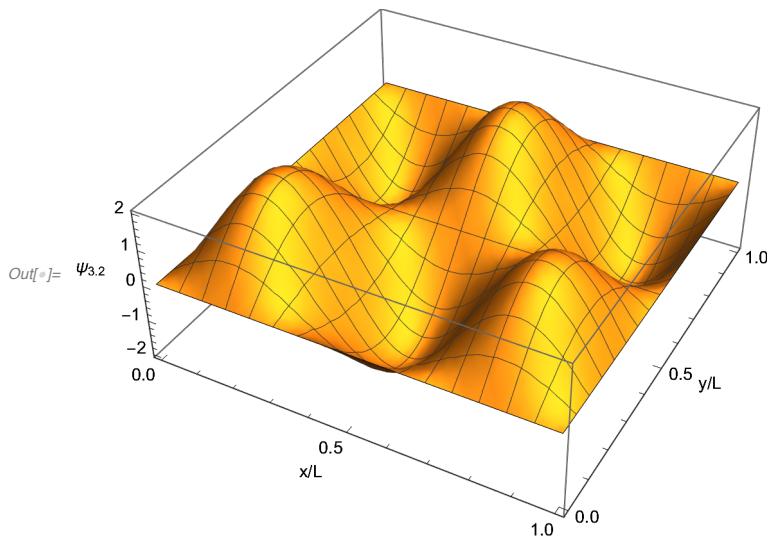
```
In[1]:= psi[x_, y_] := 2 Sin[n1 Pi x] Sin[n2 Pi y]; (* L= *)  
Plot3D[psi[x, y] /. {n1 -> 2, n2 -> 2}, {x, 0, 1}, {y, 0, 1},  
AxesLabel -> {"x/L", "y/L", "\u03c8\u2082.\u2082"}] (* setting n1=2 and setting n2=2 *)
```



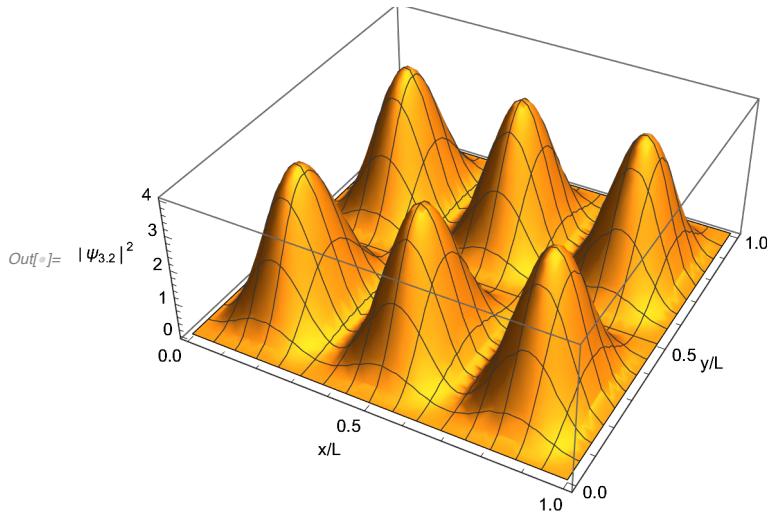
```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 2, n2 → 2},
{x, 0, 1}, {y, 0, 1}, AxesLabel → {"x/L", "y/L", "|ψ₂₂|²"}]
```



```
In[]:= Plot3D[psi[x, y] /. {n1 → 3, n2 → 2},
{x, 0, 1}, {y, 0, 1}, AxesLabel → {"x/L", "y/L", "ψ₃₂"}]
(* changing n1 and n2 values to 3 and 2, respectively *)
```

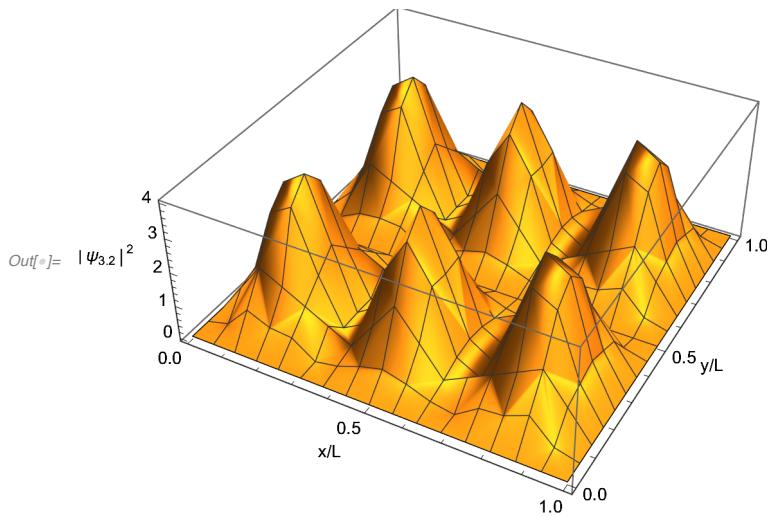


```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 3, n2 → 2}, {x, 0, 1}, {y, 0, 1}, AxesLabel → {"x/L", "y/L", "|ψ<sub>3.2</sub>|²"}]
```

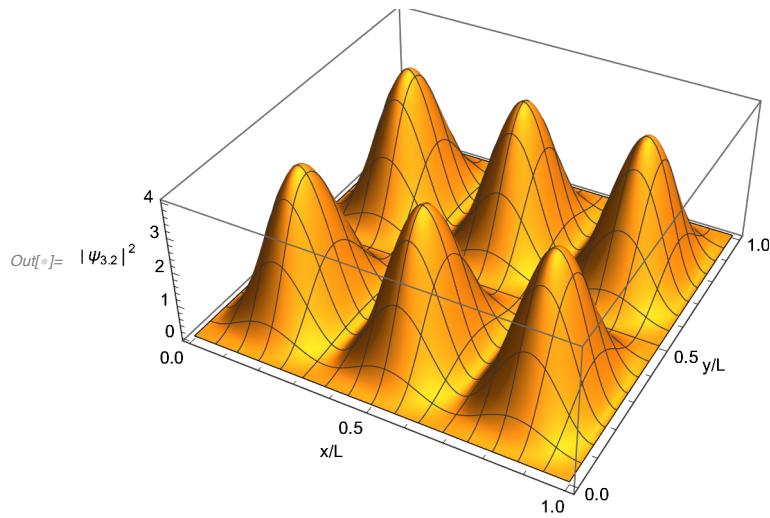


Plot3D has many options that enable you to customize your plot.

```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 3, n2 → 2}, {x, 0, 1}, {y, 0, 1}, AxesLabel → {"x/L", "y/L", "|ψ<sub>3.2</sub>|²"}, PlotPoints → 5]
```

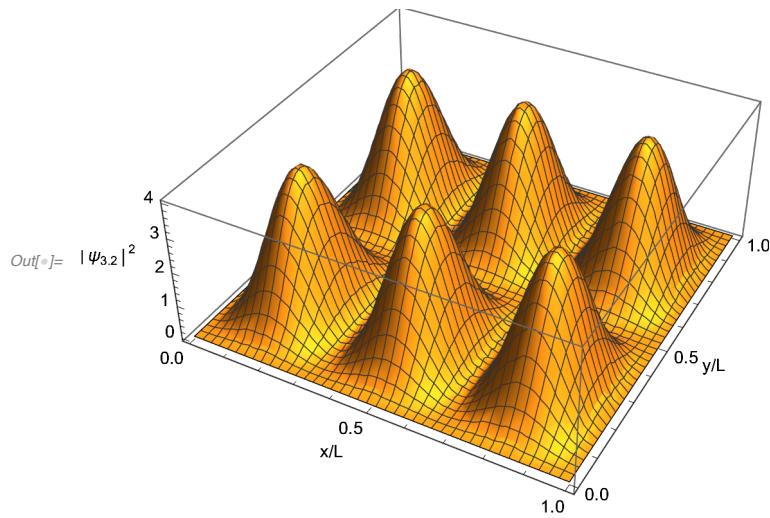


```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 3, n2 → 2}, {x, 0, 1},
{y, 0, 1}, AxesLabel → {"x/L", "y/L", "|ψ3.2|^2"}, PlotPoints → 100]
```

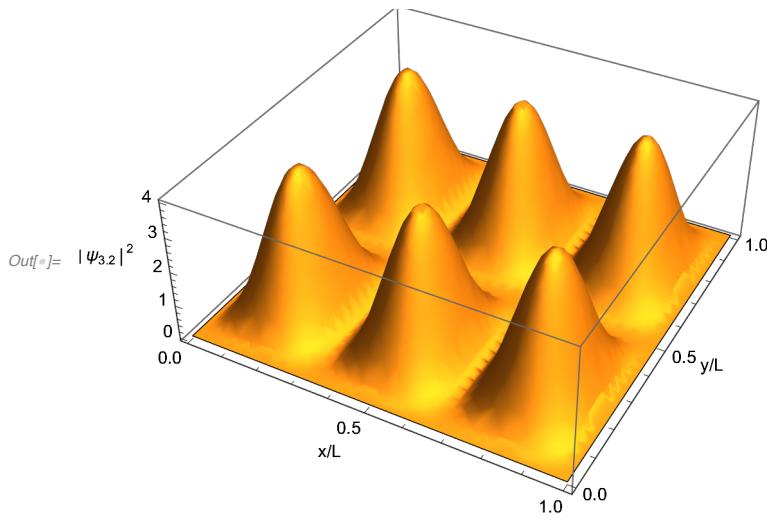


You can also change the mesh, or remove it altogether:

```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 3, n2 → 2}, {x, 0, 1},
{y, 0, 1}, AxesLabel → {"x/L", "y/L", "|ψ3.2|^2"}, Mesh → 40]
```

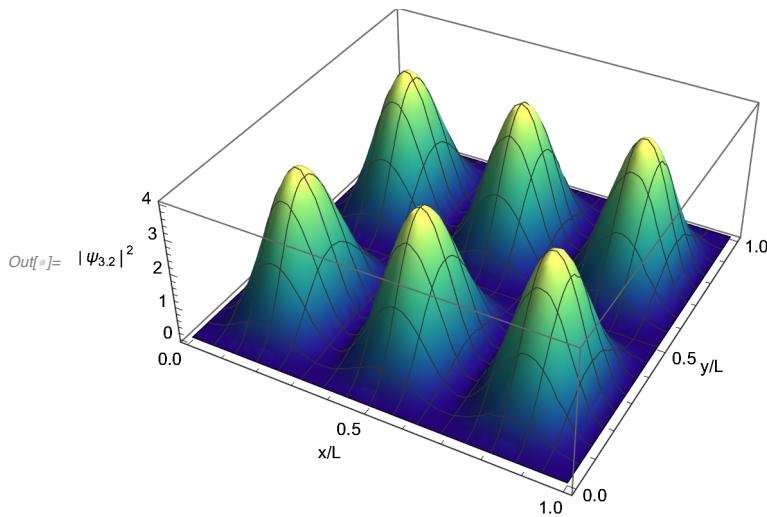


```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 3, n2 → 2}, {x, 0, 1}, {y, 0, 1}, AxesLabel → {"x/L", "y/L", "|ψ<sub>3.2</sub>|²"}, Mesh → None]
```

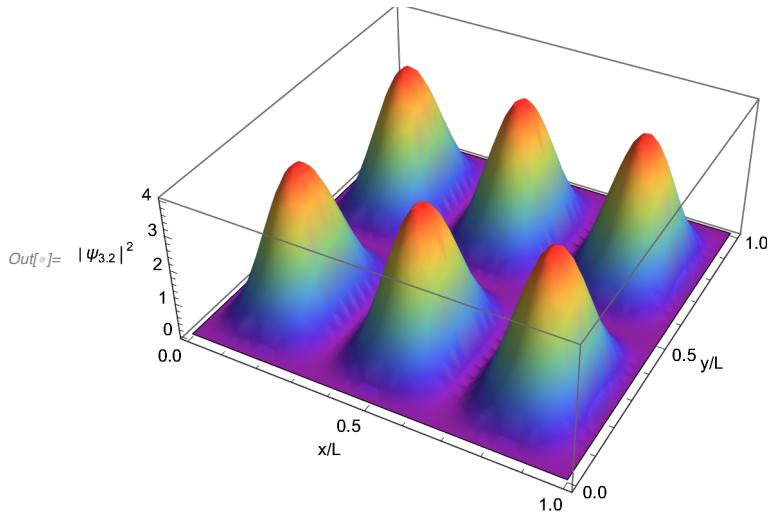


There are all sort of ways to change the coloring scheme. Here are a few examples using the **ColorFunction** option (read ColorSchemes in the Documentation Center for more):

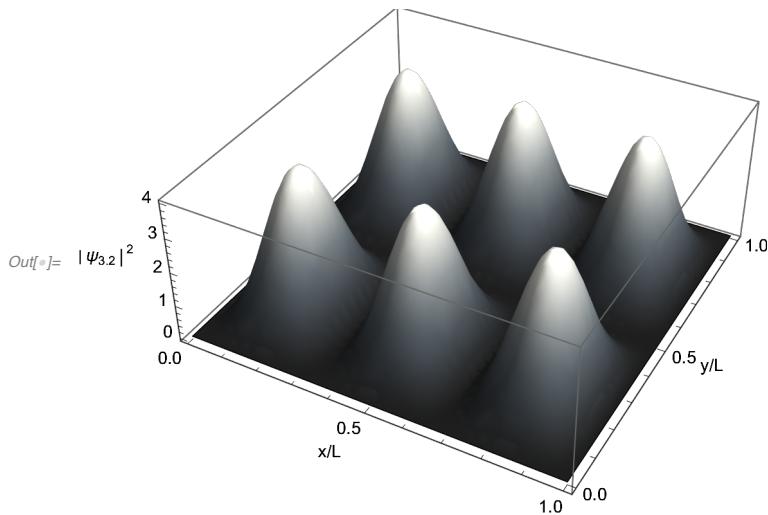
```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 3, n2 → 2}, {x, 0, 1}, {y, 0, 1}, AxesLabel → {"x/L", "y/L", "|ψ<sub>3.2</sub>|²"}, ColorFunction → "BlueGreenYellow"]
```



```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 3, n2 → 2}, {x, 0, 1}, {y, 0, 1},
AxesLabel → {"x/L", "y/L", "|ψ3,2|2"}, ColorFunction → "Rainbow", Mesh → None]
```



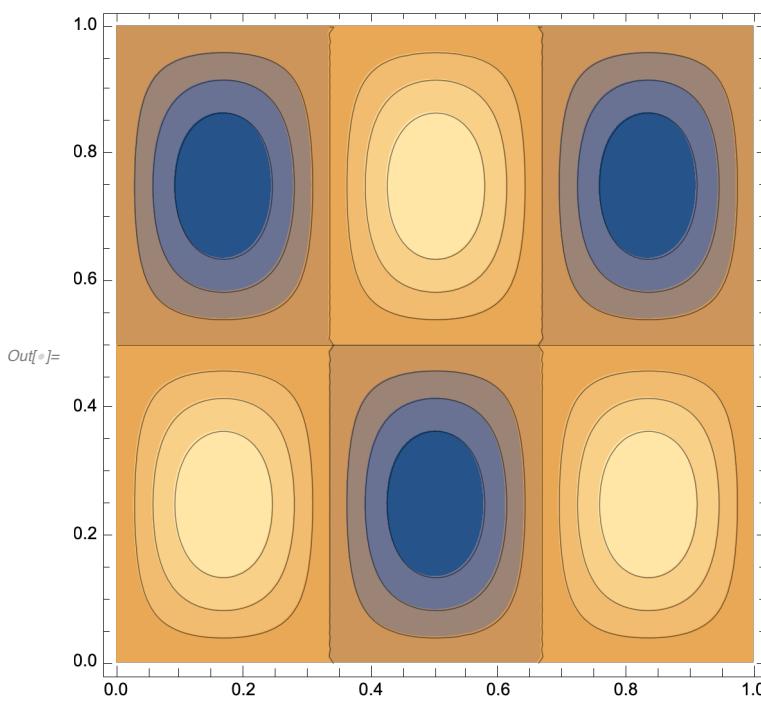
```
In[]:= Plot3D[(psi[x, y])^2 /. {n1 → 3, n2 → 2}, {x, 0, 1}, {y, 0, 1},
AxesLabel → {"x/L", "y/L", "|ψ3,2|2"}, ColorFunction → "GrayTones", Mesh → None]
```



Contour Plots of functions of Two Variables

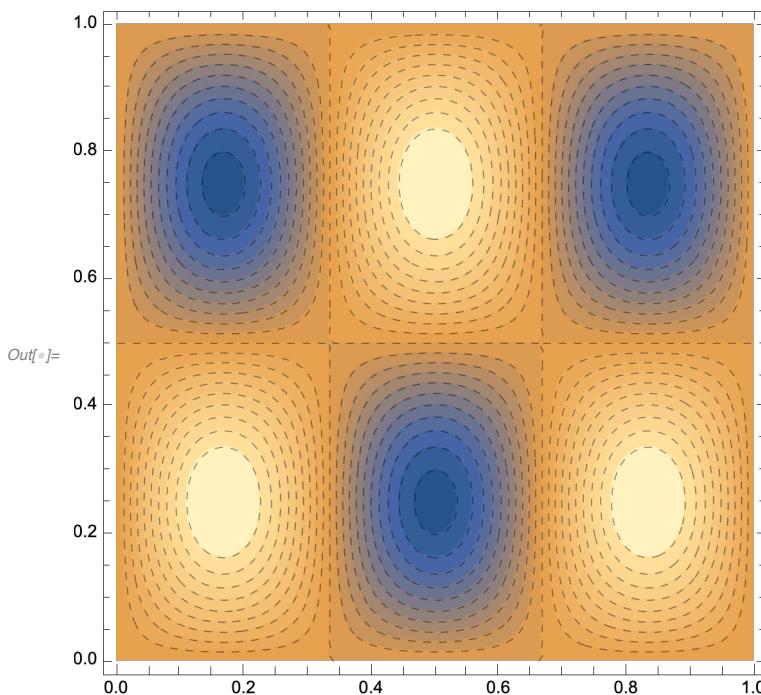
A contour plot is a two-dimensional representation of a three-dimensional surface. A topographical map is an example of a contour plot. The contour lines represent “level sets”, sets of points that have the same value of z (elevation in the case of a topo map). A contour plot is generated in *Mathematica* using the **ContourPlot** command:

```
In[]:= ContourPlot[psi[x, y] /. {n1 → 3, n2 → 2}, {x, 0, 1}, {y, 0, 1}]
```



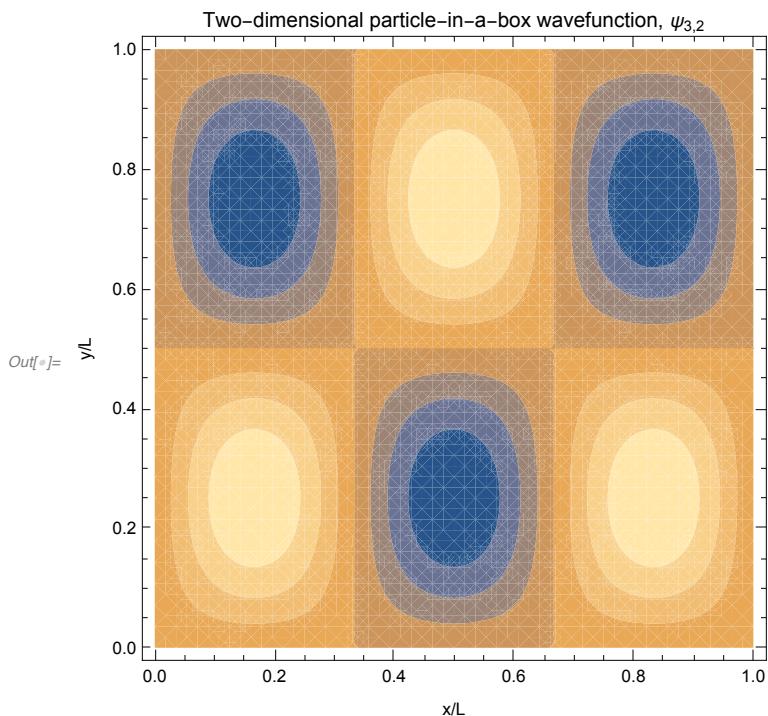
You can change the number of contours and the style with which the contours are drawn:

```
In[]:= ContourPlot[psi[x, y] /. {n1 → 3, n2 → 2},  
{x, 0, 1}, {y, 0, 1}, Contours → 20, ContourStyle → Dashed]
```



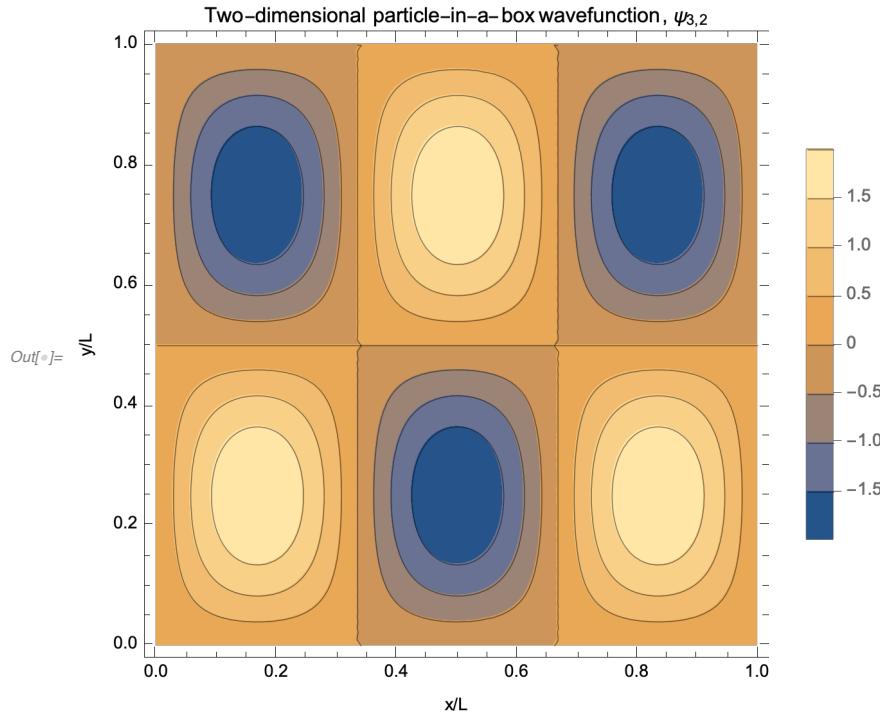
Below shows some of the options for contour plots:

```
In[]:= ContourPlot[psi[x, y] /. {n1 → 3, n2 → 2}, {x, 0, 1},  
{y, 0, 1}, FrameLabel → {"x/L", "y/L"}, PlotLabel →  
"Two-dimensional particle-in-a-box wavefunction, ψ3,2", ContourStyle → None]
```



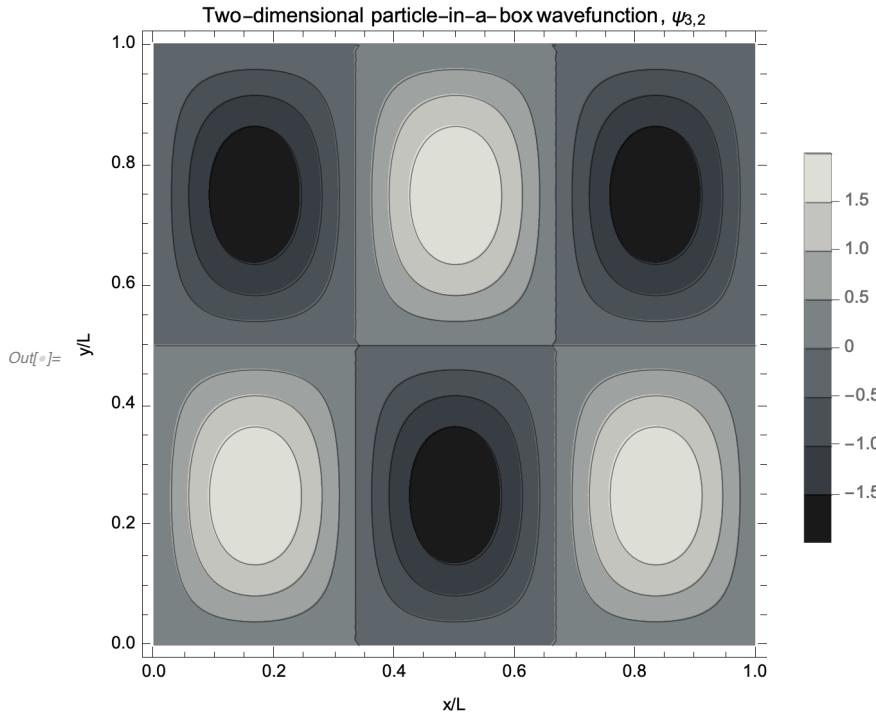
You can show the legends:

```
In[5]:= ContourPlot[psi[x, y] /. {n1 → 3, n2 → 2}, {x, 0, 1},  
{y, 0, 1}, FrameLabel → {"x/L", "y/L"}, PlotLegends → Automatic,  
PlotLabel → "Two-dimensional particle-in-a-box wavefunction, ψ3,2"]
```



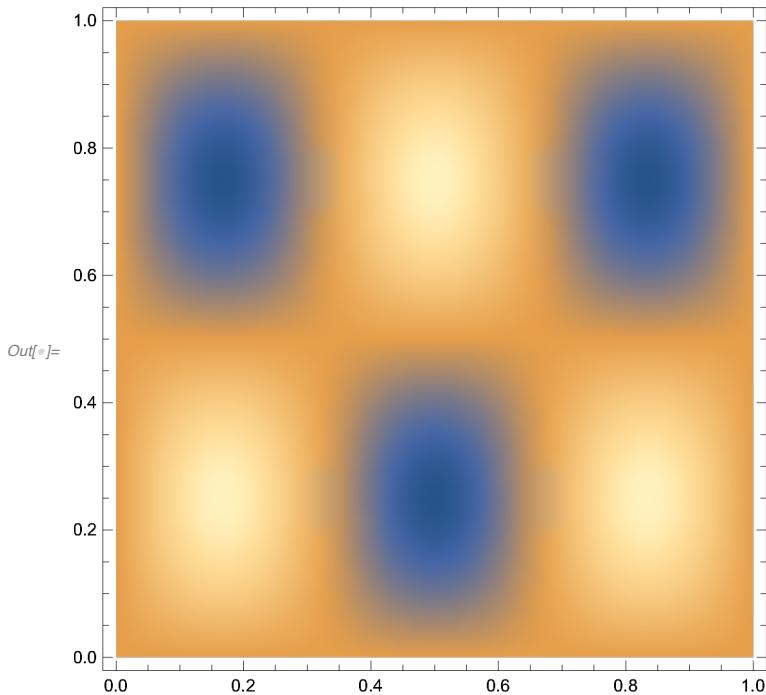
We can also change the color scheme of the plot and legend to grayscale.

```
In[5]:= ContourPlot[psi[x, y] /. {n1 → 3, n2 → 2}, {x, 0, 1},  
{y, 0, 1}, FrameLabel → {"x/L", "y/L"}, PlotLegends → Automatic,  
PlotLabel → "Two-dimensional particle-in-a-box wavefunction, ψ3,2",  
ColorFunction → "GrayTones"]
```



Density Plots

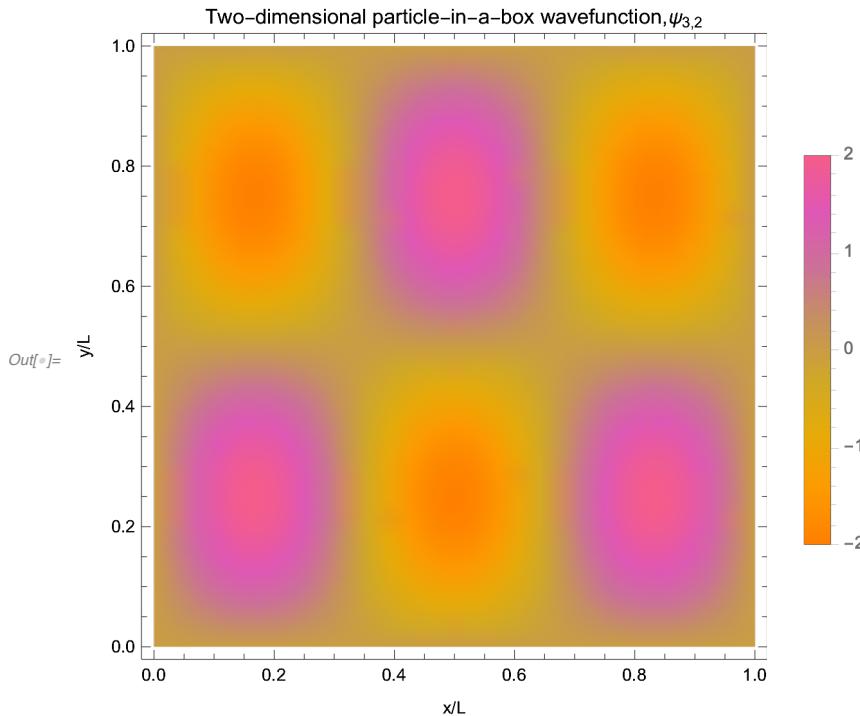
```
In[6]:= DensityPlot[psi[x, y] /. {n1 → 3, n2 → 2}, {x, 0, 1}, {y, 0, 1}]
```



The usual myriad of options for the plots are available, e.g.,

```
In[5]:= DensityPlot[psi[x, y] /. {n1 → 3, n2 → 2}, {x, 0, 1},
{y, 0, 1}, FrameLabel → {"x/L", "y/L"}, PlotLegends → Automatic,
PlotLabel → "Two-dimensional particle-in-a-box wavefunction, ψ3,2",

ColorFunction → "FruitPunchColors"]
```



Contour Plots of Functions of Three Variables

Many functions that are of interest in chemistry are functions of three variables, and hence, are four-dimensional. Important examples include atomic and molecular wavefunctions (orbitals). We can visualize three-dimensional contour plots of four-dimensional functions of three variables. The plots of atomic orbitals (e.g., $1s$, $2p_z$, $3d_{z^2}$, etc.) are “boundary surfaces”, where the boundary is defined by choosing a particular value (or contour), c , that defines the contour of the four-dimensional surface to be displayed, i.e., $f(x,y,z) = c$ defines a three-dimensional contour of the four-dimensional surface described by $f(x,y,z)$.

Mathematica has a function, `ContourPlot3D`, that generates a three-dimensional contour plot of a four-dimensional function. To illustrate this, we will plot the boundary surfaces of H atom atomic orbitals and their corresponding probability densities below. The H atom wavefunctions (orbitals) are obtained by solving the Schrödinger equation for a negatively charged electron interacting with a positively charged nucleus. The standard approach to the problem is to use spherical-polar coordinates, r, θ, ϕ , where r is the distance of the electron from the nucleus and θ and ϕ are angles with respect to the Cartesian axes (x , y , and z). The general form of the solution is:

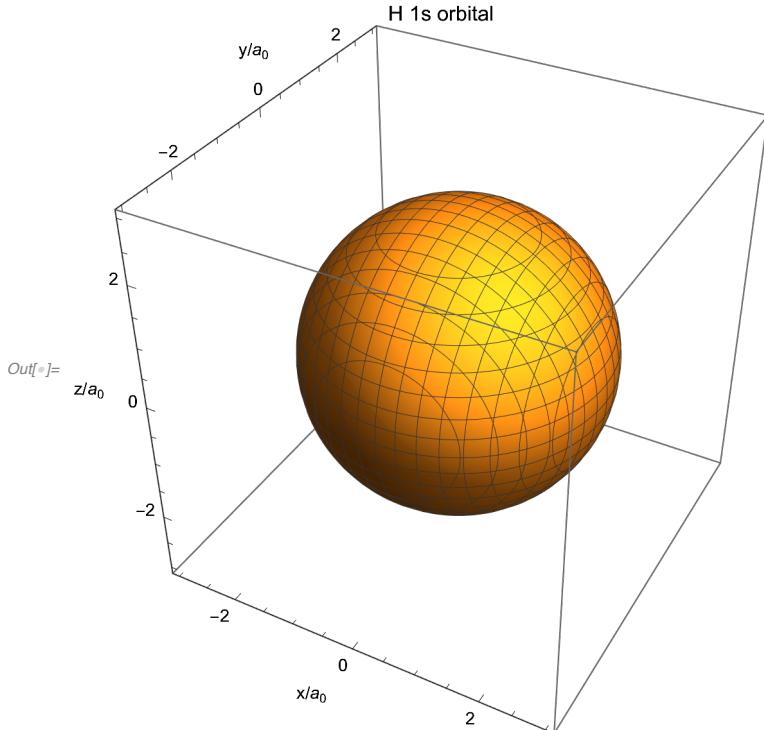
$$\psi_{n,l,m_l} = R_{n,l}(r) Y_{l,m_l}(\theta, \phi)$$

The indices n , l , and m_l are the familiar principal, orbital angular momentum, and magnetic quantum numbers, respectively. The functions R describe the radial (distance) dependence, and the functions Y (the so-called spherical harmonics) describe the angular dependence (shape) of the wavefunction. R and Y can be expressed in terms of polynomial “special functions” that are defined in *Mathematica*. The functions Y are complex, and hence are not easily displayed. The orbitals that are displayed in textbooks are actually linear combinations of the complex functions that are real functions in Cartesian coordinates. In Cartesian coordinates, the orbitals have the form:

$$f(x,y,z) = A g(x,y,z) h(r)$$

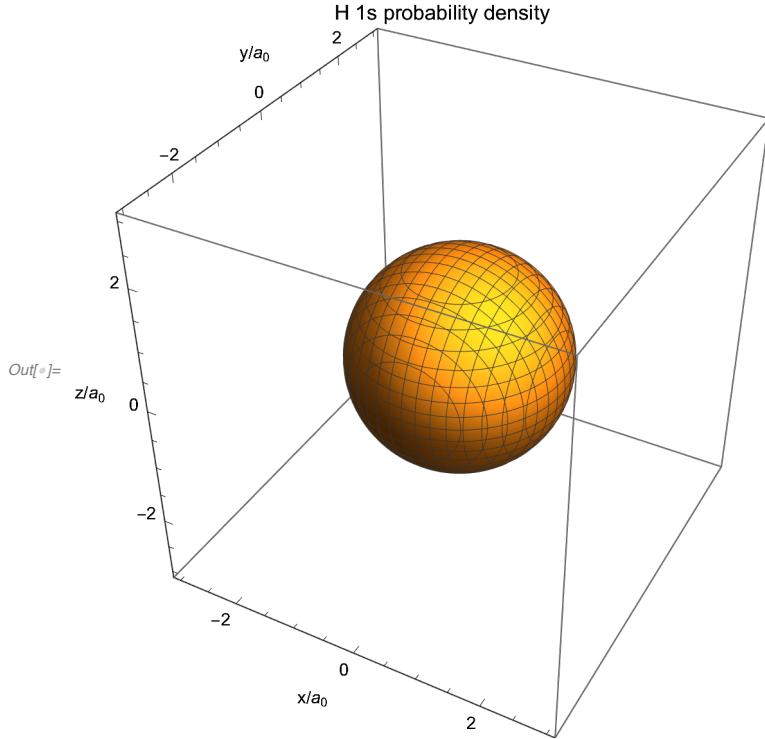
where A is a constant and $r = r(x,y,z) = (x^2 + y^2 + z^2)$. For example, if we take r to be in units of a_0 (the Bohr radius), for the 1s orbital, $g(x,y,z) = 1$ and $h(r) = e^{-r/2}$; and for the $3d_z$ orbital, $g(x,y,z) = 3z^2 - r^2$ and $h(r) = e^{-r/3}$. It is straightforward to display boundary surfaces of the orbitals (and their corresponding probability densities) in Cartesian coordinates.

```
In[]:= r[x_, y_, z_] := Sqrt[x^2 + y^2 + z^2];
f[x_, y_, z_] := E^(-r[x, y, z]);
ContourPlot3D[f[x, y, z] == 0.1, {x, -3, 3}, {y, -3, 3}, {z, -3, 3},
AxesLabel -> {"x/a₀", "y/a₀", "z/a₀"}, PlotLabel -> "H 1s orbital"]
```



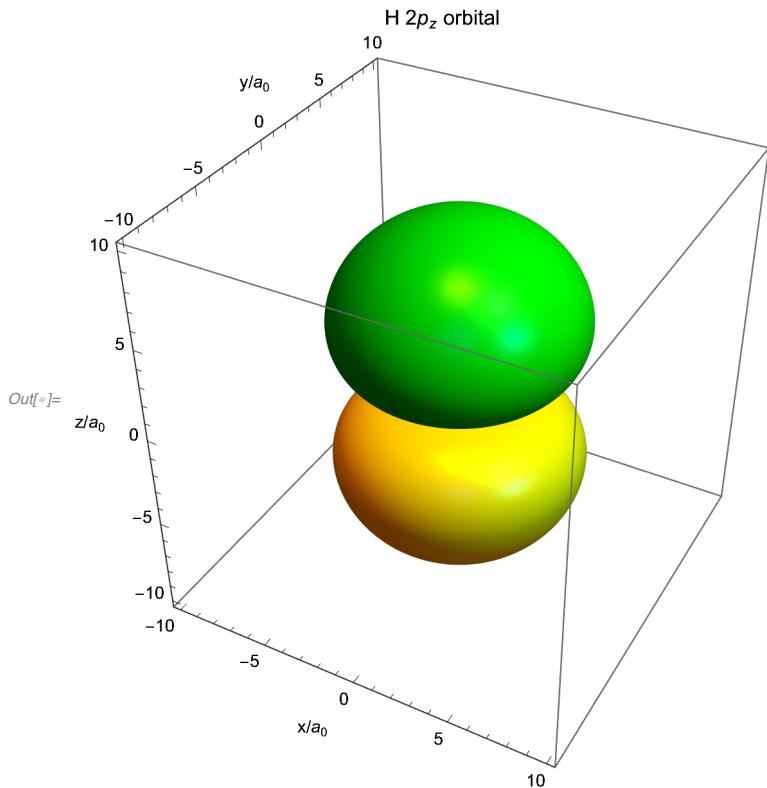
Here is $r^2 f^2$, which is proportional to the electron's probability density:

```
In[5]:= probdens[x_, y_, z_] := (r[x, y, z])^2 * (f[x, y, z])^2;
ContourPlot3D[probdens[x, y, z] == 0.1, {x, -3, 3}, {y, -3, 3}, {z, -3, 3},
AxesLabel → {"x/a₀", "y/a₀", "z/a₀"}, PlotLabel → "H 1s probability density"]
```



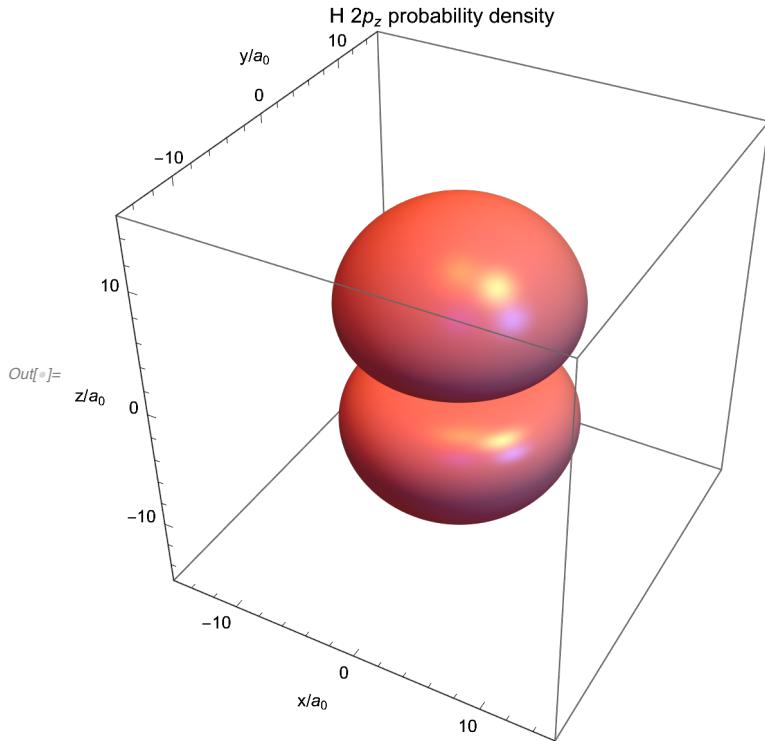
Here is the $2p_z$ orbital. Note that one lobe has a positive amplitude, and the other has a negative amplitude, so we plot two contours, $f(x,y,z) = 0.1$ as a green surface and $f(x,y,z) = -0.1$ as a yellow surface.

```
In[ $\circ$ ]:= Clear[f, probdens];
f[x_, y_, z_] := z E^(-r[x, y, z] / 2);
ContourPlot3D[{f[x, y, z] == 0.1, f[x, y, z] == -0.1}, {x, -10, 10}, {y, -10, 10},
{z, -10, 10}, AxesLabel -> {"x/a0", "y/a0", "z/a0"}, PlotLabel -> "H 2pz orbital",
Mesh -> None, ContourStyle -> {Directive[Green, Specularity[White, 100]],
Directive[Yellow, Specularity[White, 100]]}]
```



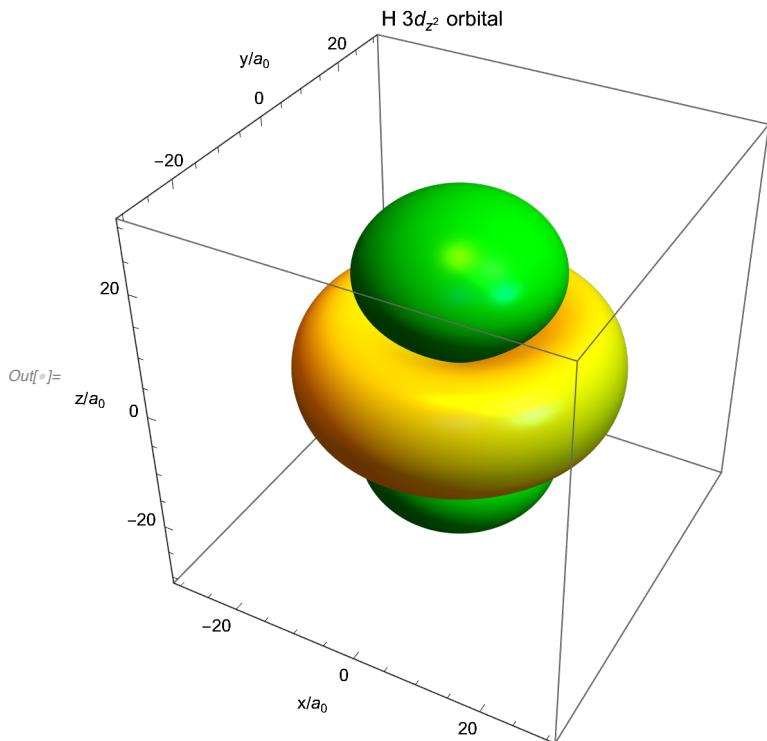
Here is the $2p_z$ probability density. Note that the probability density is everywhere positive, so we only need one surface, which is plotted here in pink:

```
In[6]:= probdens[x_, y_, z_] := (r[x, y, z])^2 * (f[x, y, z])^2;
ContourPlot3D[probdens[x, y, z] == 0.1, {x, -15, 15}, {y, -15, 15}, {z, -15, 15},
AxesLabel -> {"x/a0", "y/a0", "z/a0"}, PlotLabel -> "H 2pz probability density",
Mesh -> None, ContourStyle -> Directive[Pink, Specularity[White, 100]]]
```



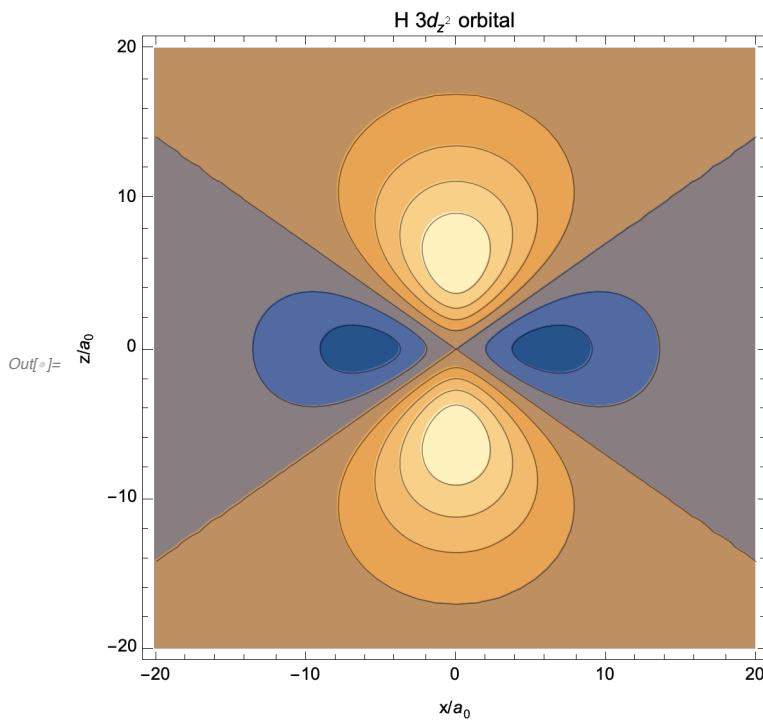
Here is the $3d_{z^2}$ orbital.

```
In[]:= Clear[f];
f[x_, y_, z_] := (3 z^2 - (r[x, y, z])^2) E^(-r[x, y, z] / 3);
ContourPlot3D[{f[x, y, z] == 0.2, f[x, y, z] == -0.2}, {x, -30, 30}, {y, -30, 30},
{z, -30, 30}, AxesLabel -> {"x/a₀", "y/a₀", "z/a₀"}, PlotLabel -> "H 3dz² orbital",
Mesh -> None, ContourStyle -> {Directive[Green, Specularity[White, 100]],
Directive[Yellow, Specularity[White, 100]]}]
```

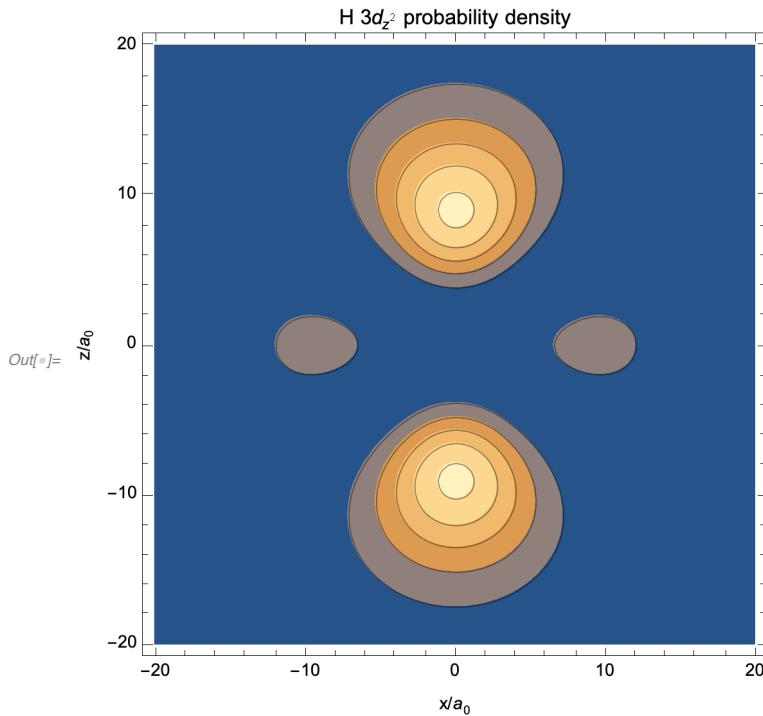


Contour and density plots are also useful for visualizing projections of orbitals. For example, we can slice through the orbital in the xz plane by setting $y = 0$, i.e., removing y from our function definitions. Below we display the $3d_{z^2}$ orbital and corresponding probability density in the xz plane using **ContourPlot** and **DensityPlot**. The latter is particularly nice for depicting the “fuzziness” of orbitals and probability densities. Note that the option **PlotRange → All** is used in the contour plots to keep the function from being “cut off”.

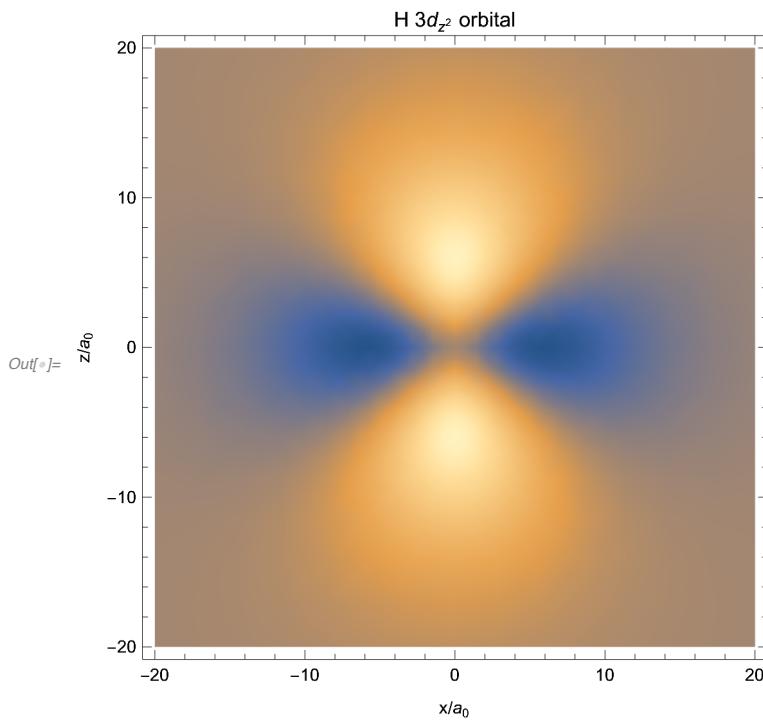
```
In[]:= Clear[r, f, probdens];
r[x_, z_] := Sqrt[x^2 + z^2];
f[x_, z_] := (3 z^2 - (r[x, z])^2) E^(-r[x, z] / 3);
probdens[x_, z_] := (r[x, z])^2 * (f[x, z])^2;
ContourPlot[f[x, z], {x, -20, 20}, {z, -20, 20}, PlotRange -> All,
FrameLabel -> {"x/a₀", "z/a₀"}, PlotLabel -> "H 3dz² orbital", PlotRange -> All]
```



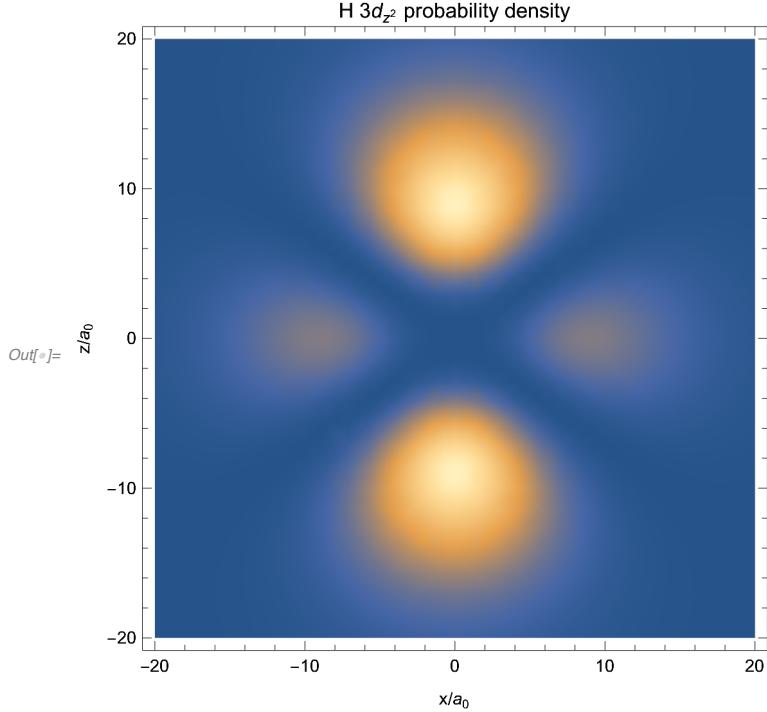
```
In[6]:= ContourPlot[probdens[x, z], {x, -20, 20},
{z, -20, 20}, PlotRange → All, FrameLabel → {"x/a₀", "z/a₀"}, PlotLabel → "H 3dz² probability density", PlotRange → All]
```



```
In[7]:= DensityPlot[f[x, z], {x, -20, 20}, {z, -20, 20}, PlotRange → All,
FrameLabel → {"x/a₀", "z/a₀"}, PlotLabel → "H 3dz² orbital"]
```



```
In[5]:= DensityPlot[probdens[x, z], {x, -20, 20}, {z, -20, 20}, PlotRange -> All,
FrameLabel -> {"x/a0", "z/a0"}, PlotLabel -> "H 3dz2 probability density"]
```



Reference:

Tobias, Douglas Ph.D. Chemistry 5 (UCI OpenCourseWare: University of California, Irvine), License: Creative Commons Attribution-ShareAlike 3.0 United States License