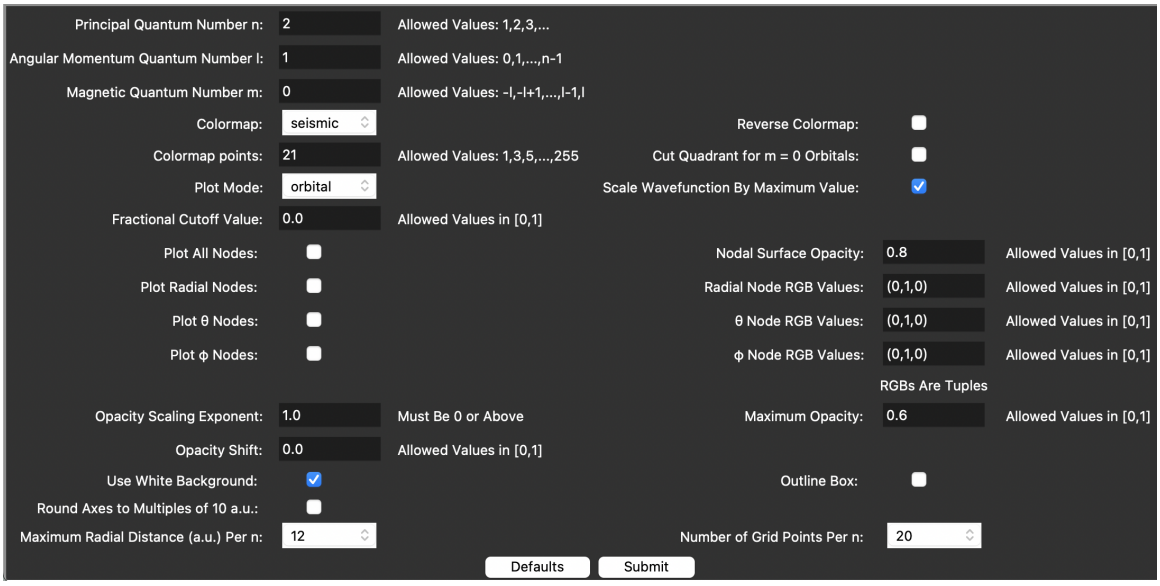
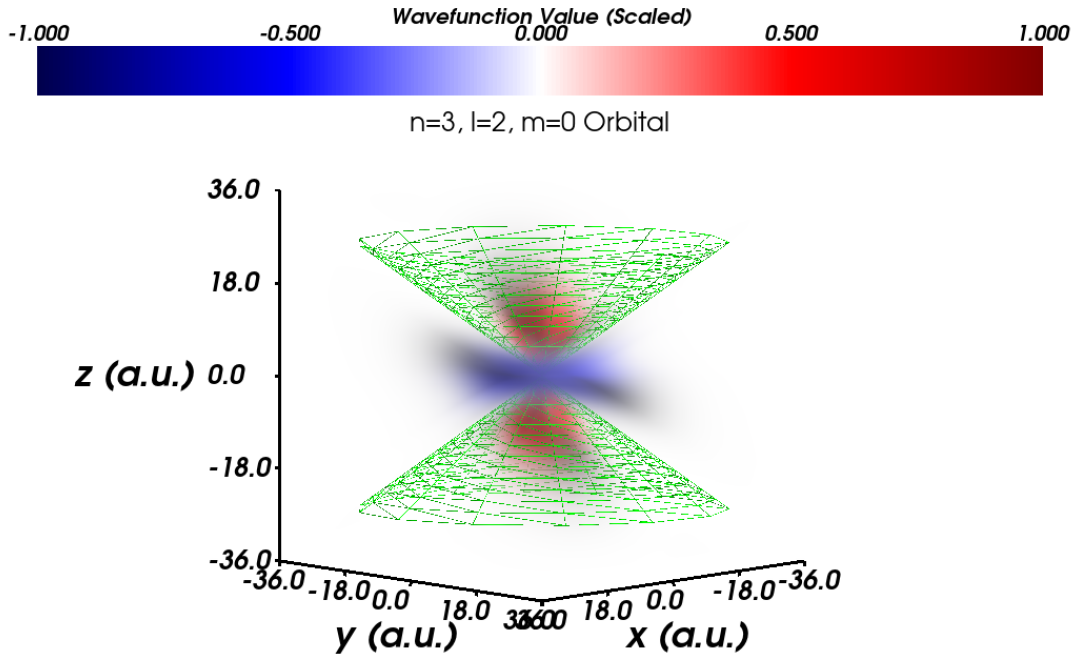
Contents

1 Preface

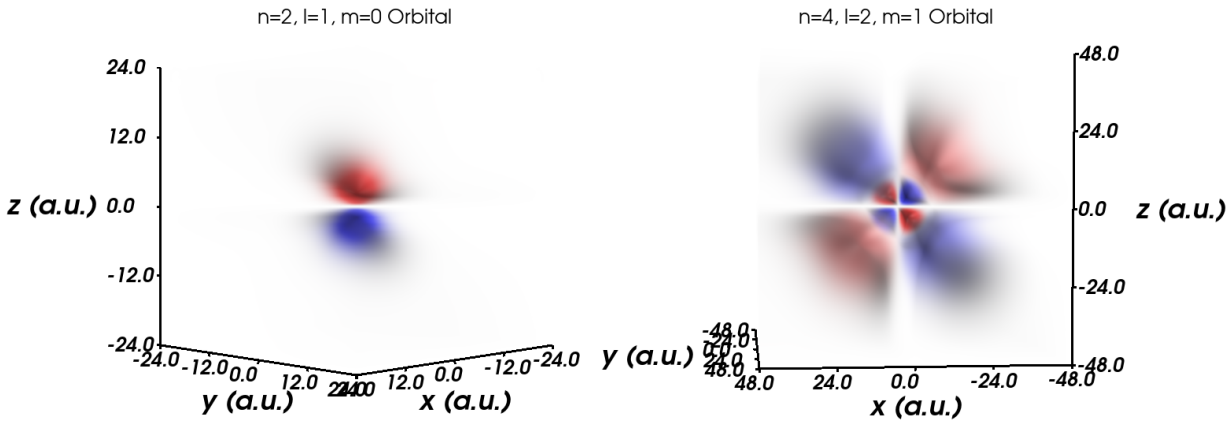
This file contains a breakdown of various features present in the Atomic_Orbitals_Visualizer programs. These settings are modified through the use of a GUI built in Python's tkinter library, which is shown below. An example output from the program is shown as well. The various sections of this document are broken down in order of the options present in the GUI. The relevant portion of the GUI will be highlighted at the start of each section.



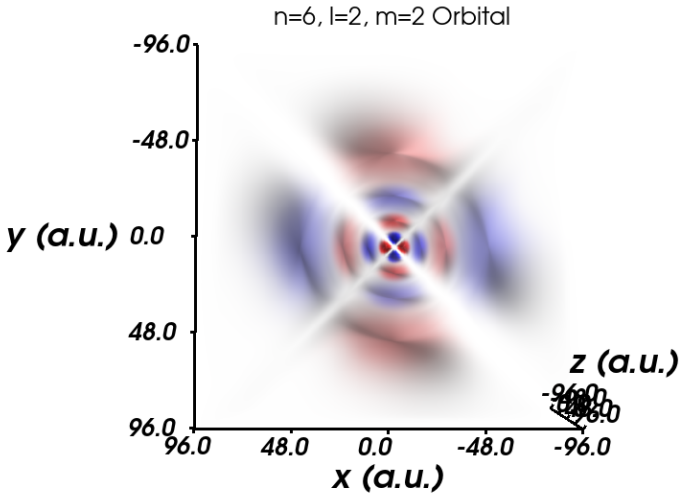


2 Quantum Numbers

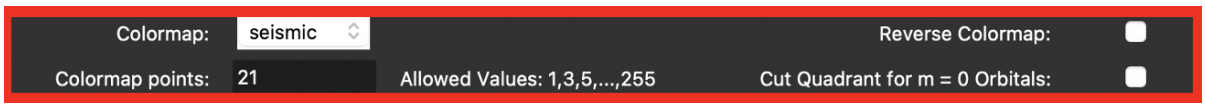
The most obvious utility available in this visualizer program is the ability to change the quantum numbers of the orbitals of interest. This allows the user to visualize any type of hydrogen orbital, even those that extend beyond the occupied orbitals on the periodic table. For example, the $n = 2$, $l = 1$, $m = 0$ and $n = 4$, $l = 2$, $m = 1$ orbitals are rather different.



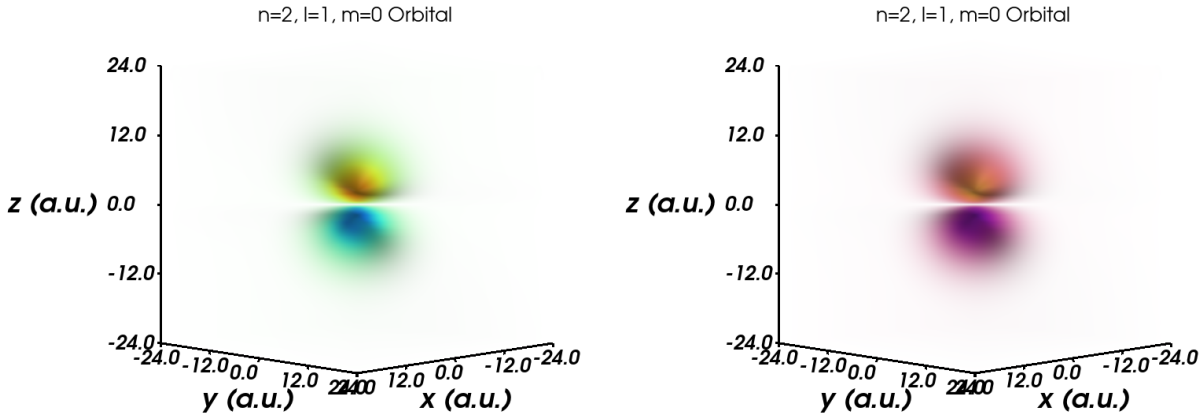
Note that in this and most examples throughout this guide the colorbar at the top or the orbital visualizer window will be omitted to eliminate visual clutter. The user is not limited to "typical" orbitals, however, and more exotic examples can be made easily, as shown below.



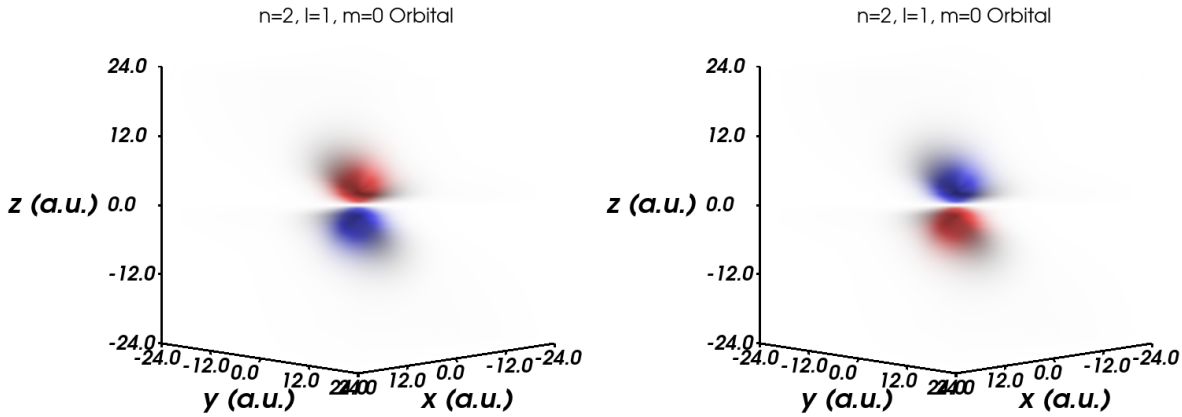
3 Colormaps and Quadrant Cuts



The user can choose to vary the colormap used to represent the amplitude of the orbital wavefunction, though the default or similar colormaps like "coolwarm" and "bwr" are among the best options. Two examples of orbitals with alternate dramatic colormaps called "jet" and "plasma" are shown below.

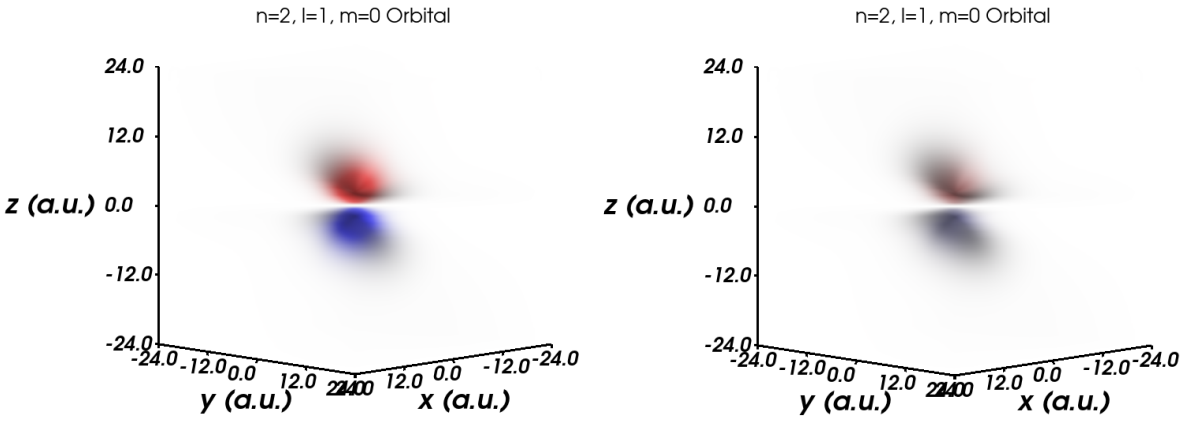


If desired, the reverse colormap checkbox will simply flip the order of the colors in the colormap, as shown below. This feature is purely cosmetic and will alter nothing about the values of the orbital wavefunction or the range of the colormap.

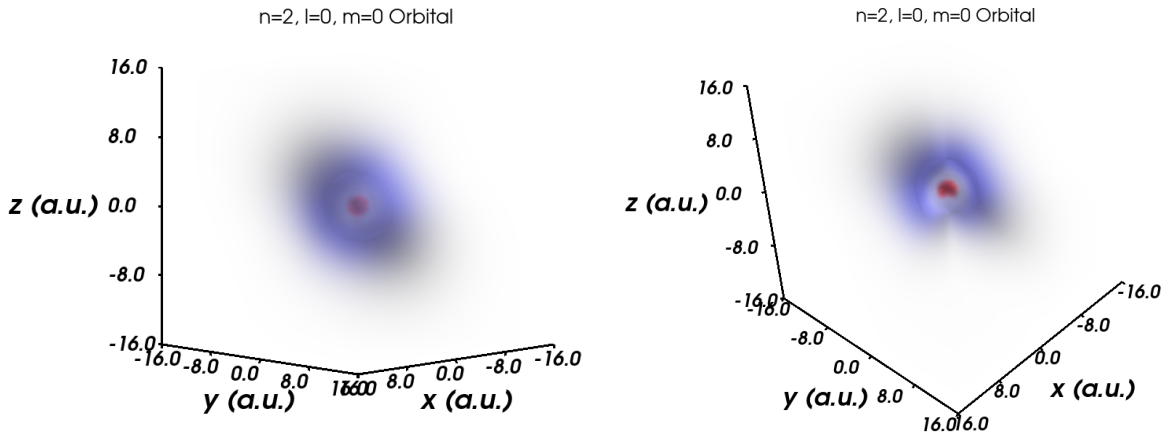


In addition, varying the number of points used for the colormap can alter its precision. This does not need to be changed, as the default value is quite sensitive. A particularly silly example with only 3 values along the colormap is shown as a contrast to the default of 21

values.

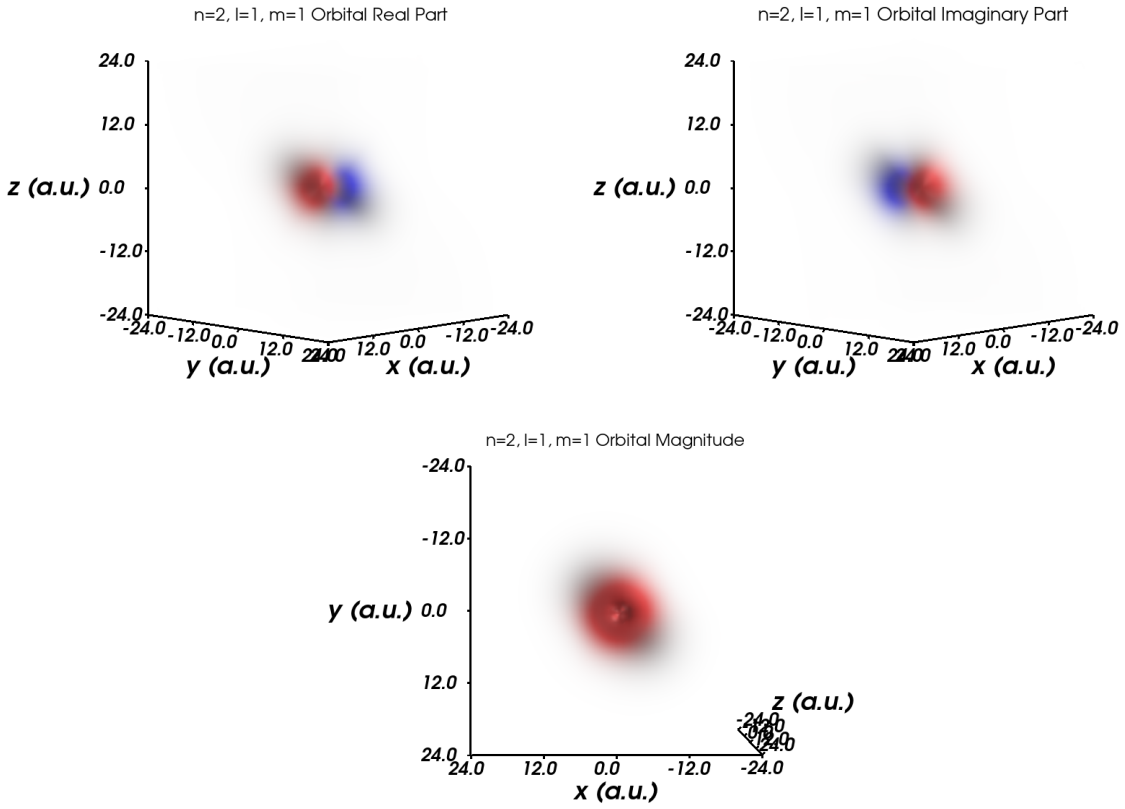


The option to cut a quadrant out for $m = 0$ orbitals makes visualizing the internal structure of the s orbitals dramatically easier. This is especially helpful when the value of n gets larger than 3. This feature can also help parse the structure of orbitals with larger values of l, especially when radial nodes are present.



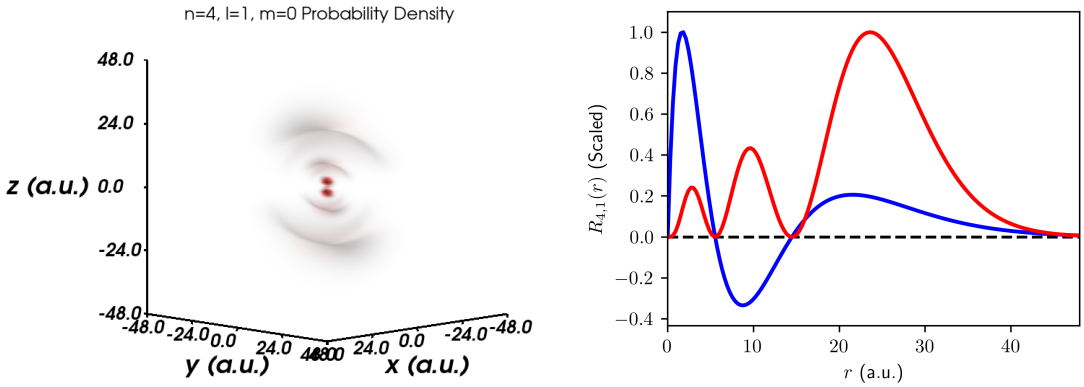
4 Plot Mode and Wavefunction Scaling and Cutoff

Changing the plot mode allows the user to see the real and imaginary parts of the complex wavefunctions $\psi_{n,l,m}$ along with the magnitude, as shown below for the $n = 2$, $l = 1$, $m = 1$ orbital.

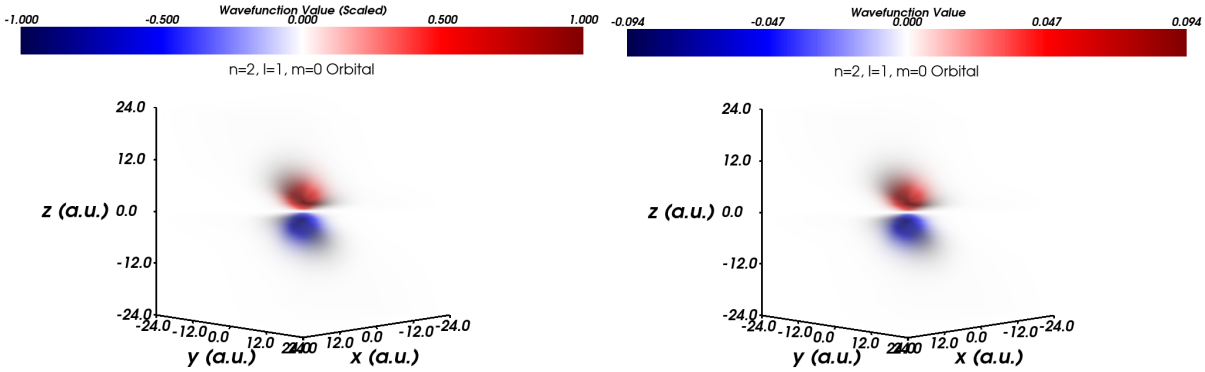


The default is to show the purely real representation of the orbitals $\phi_{n,l,m}$, which is what all of the plots in prior sections have shown. Finally, the probability density $\phi^2_{n,l,m}$ can be shown. Note that the probability density does not include the factor of r^2 necessary to integrate the radial probability in spherical coordinates! This is for consistency, since there is no factor of $\sin(\theta)$ included in the angular wavefunction in this mode either. It should be noted that when the nodes are plotted or the probability density is requested, the radial probability function $r^2 R_{n,l}(r)^2$ is plotted along with the radial wavefunction. An example of

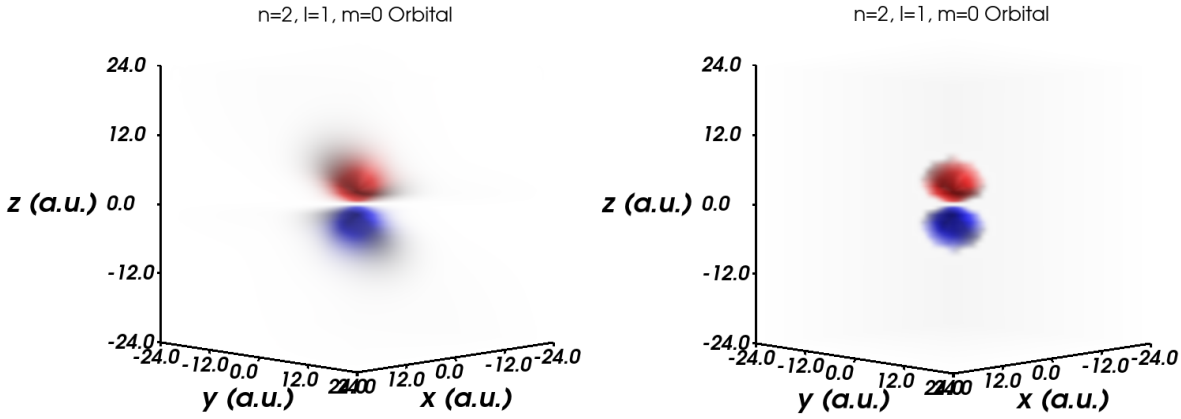
the probability density and the radial probability plot (red curve) is given below.



The code also allows the user to specify whether the wavefunction is reported with actual numbers or with scaled relative values. This only affects the value range on the colorbar as shown below and is entirely optional.

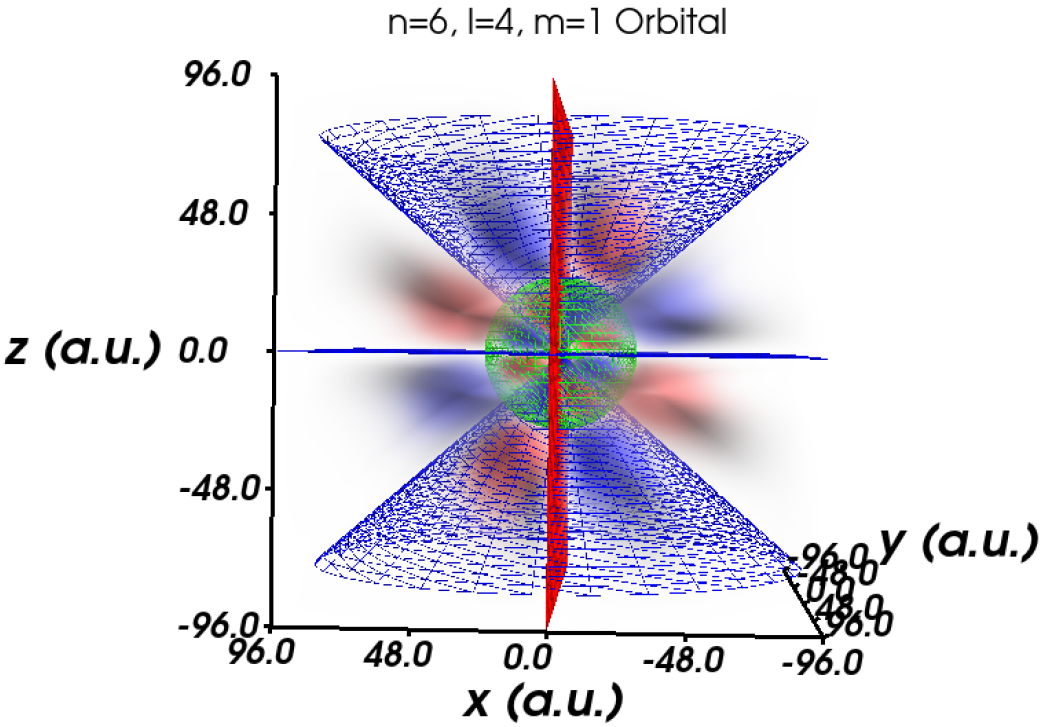


Finally, the user can also choose to set a minimal fractional value (compared to the maximum amplitude) that the wavefunction must have in order to be plotted. This can be used if excessive low-amplitude points are cluttering the view. An example of a cutoff of 0.2 is shown below.



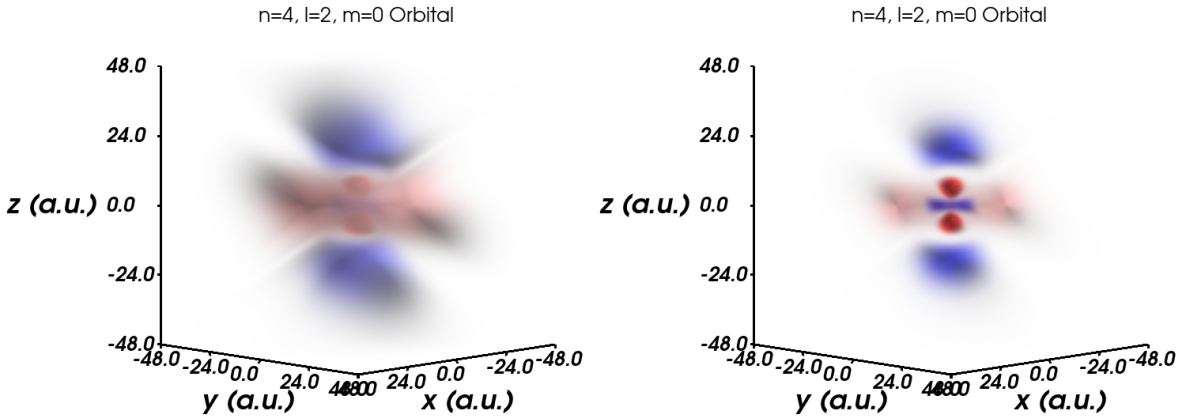
5 Nodal Surfaces

This section of the GUI allows the user to specify whether or not to plot nodal surfaces along with the wavefunction. The surfaces will always be represented by mesh surfaces whose mesh density and width follow a relatively robust scaling formula within the code. All nodes or some subset thereof can be chosen using the check boxes. By default, all nodal surfaces are green and fully opaque, as this contrasts nicely with the default colormap. An example illustrating every type of node is given below. The radial node (sphere) is colored green with the RGB tuple (0,1,0), which is the default. The θ nodes (plane and cones) that cut through the z axis are colored blue with an RGB tuple of (0,0,1). The ϕ node (plane) that cuts perpendicularly through the $x-y$ plane is colored red with the RGB tuple (1,0,0).

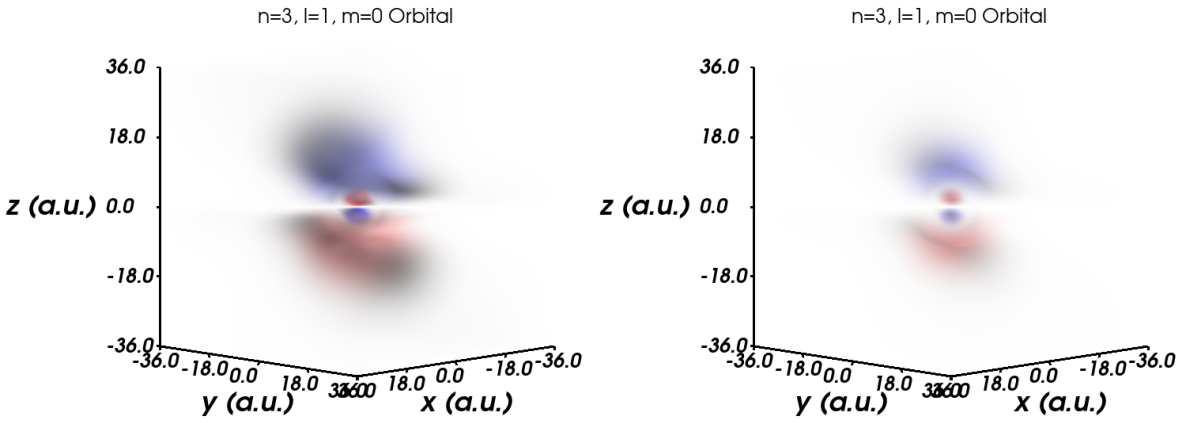


6 Wavefunction Opacity Scaling

These settings allow the user to control how opaque or "cloudy" the orbitals will look when plotted. The opacity scaling exponent allows the user to control how uniformly the opacity scales to the data. The default has the opacity tied exactly to the scaled value of the wavefunction amplitude. Increasing the value will cause low-amplitude values to appear to vanish, where decreasing the scaling exponent will cause the data to appear more uniform in opacity. This is a good way to improve the visual clarity of an orbital visualization without deleting data with a wavefunction cutoff. A good value is 1.5 when trying to reduce the low amplitude data's visibility. Comparison between the default value of 1.0 and a larger scaling exponent of 2.0 are shown below.



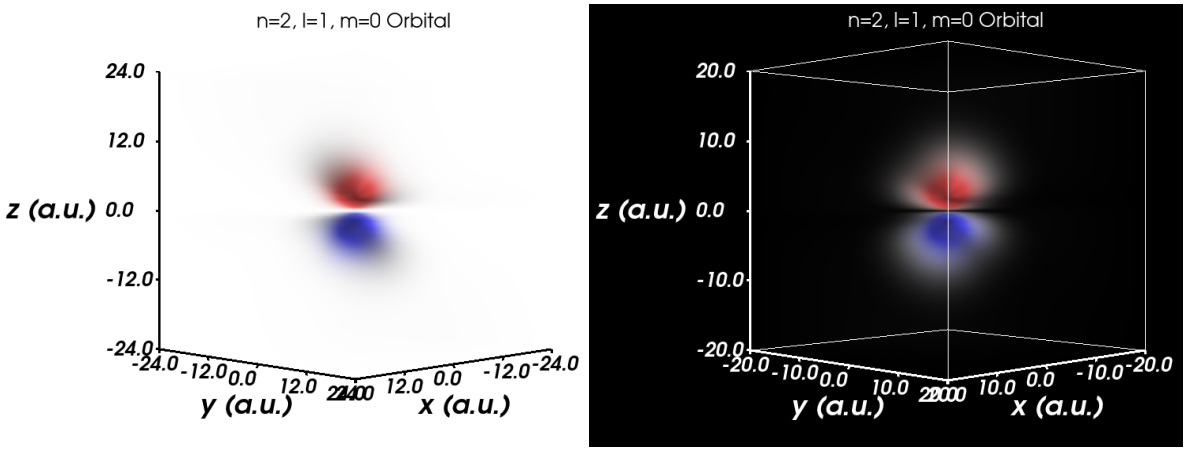
The maximum opacity is reached at the largest amplitude point(s) of the wavefunction. Setting this to 1 makes the largest amplitude points impassable visually, where small values like 0.1 make the orbital look extremely diffuse, as shown below.



The opacity shift sets a minimum opacity of all points for the wavefunction. This will typically result in more opaque figures if it is not paired with a wavefunction cutoff. This feature is not often useful, but can be used if low-amplitude data needs to be emphasized.

7 General Plotting Settings

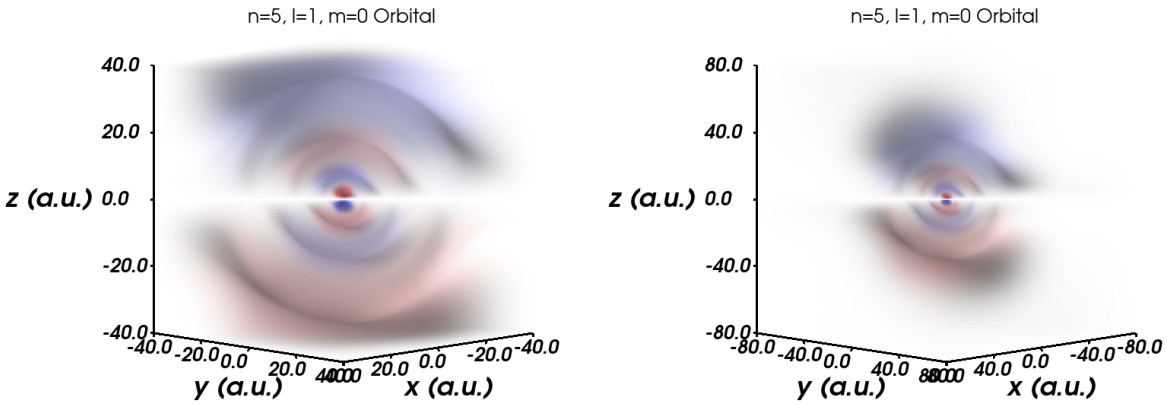
This portion of the GUI gives some options that modestly affect the overall output aesthetics. First, the option to use a white background is on by default, but turning this feature off results in a black background. The checkbox for the outline box allows the box to fully frame the orbital illustration, if desired. The color of the outline will shift if the background color is changed. Finally, the user can choose to round the length of the axes to the nearest 10 a.u., if desired. An example showing these features compared to the default is shown below.



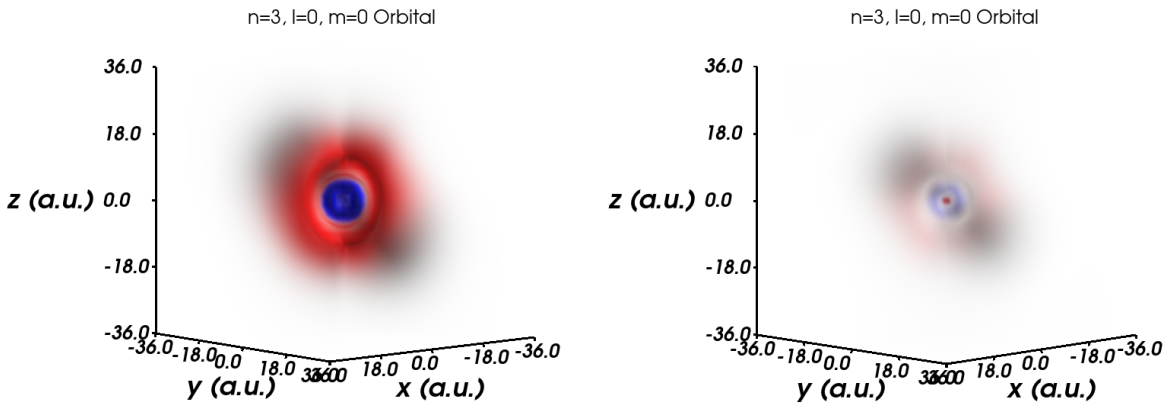
In the pure Python version of the code there is an additional checkbox that determines whether the plots of the r, θ, and ϕ components of the wavefunction will be shown. This is because these will open in separate windows that are more cumbersome than the Jupyter version of the code to manage. Users should check this box if they wish to see these plots.

8 Axis Length and Grid Point Scaling

The user can also specify how far the x, y, and z axes should go. The value scales with the value of n because increasing n increases the radial extent of the orbital. For small values of n and for s orbitals, it is useful to shrink the default value to around 8 a.u. per n. For n values larger than 5 it can be useful to increase the axis size to prevent artifically cutting off low-amplitude parts of the wavefunction. Examples of lower and higher values for the same orbital are shown below.



Finally, the user can directly change the number of grid points used to generate the wavefunction. The number in the box is multiplied by the value of n chosen and is used along the x, y, and z axes. This means that the number of grid points scales as the cube of the number chosen, so caution should be used when using a large number of grid points with a larger value of n, as this can cause massively computer slowdown and even crash the program on low-end computers. This feature is the most direct way to improve the visual quality of an orbital illustration. A visual comparison between an orbital with low (10 per n) and high (50 per n) grid points is shown below.



Note that this example also highlights that an inappropriately small grid can cause artifacts like overly dense opacities and missing features of the wavefunction.