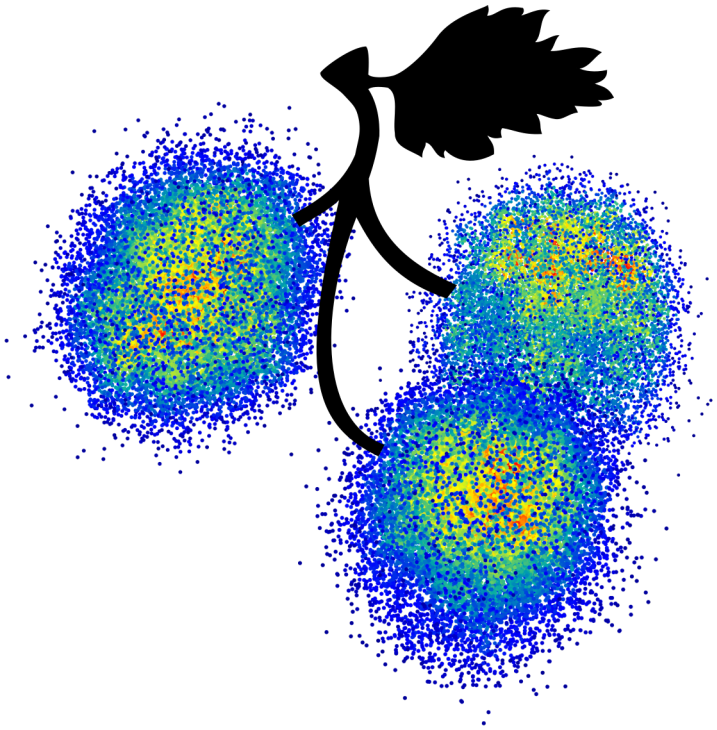
μBerry - Manual



Contents

1 Before starting ... 1

2 Program 1

2.1 General information 1
2.2 General tab 2
2.3 Slices tab 2
2.4 Videos tab 3
2.5 Color function tab 4
2.6 Colorbar tab 5
2.7 Additional features tab 6
2.8 Plotting parameters tab 8

1 Before starting ...

Before starting the program, please check the following points:

1. Verify that Mathematica is installed on the computer you are using and that you have access to the Mathematica notebook μBerry.nb.
2. An image called Logo.png should be located in the same directory as the notebook.
3. If you want to make videos using μBerry, you need the program ffmpeg.exe. It should come with the notebook, but is otherwise downloadable for free from ffmpeg.com.
4. Your input data files should be saved in a directory called Input located in your main directory.

2 Program

2.1 General information

If you have checked all points, you can now use the program. Open the notebook in Mathematica and start it by pressing Shift + Enter. Make sure that you have actually currently selected the notebook (by simply left-clicking somewhere in the code); otherwise, the code won't run. Alternatively, you could start the program by selecting Evaluation from the top menu and then selecting Evaluate cells.

Once started, the program will produce a user interface (for more information about the individual options of the UI, see below). Fill in all the data that is necessary for your desired output (everything else which is not needed should be grayed out and inactive) and press the OK button. The program will now run and create your output.

The speed of the program is (naturally) dependent on the number and size of your input files and the desired resolution of the output files. When first starting the program, you should consider using low resolution for the output files and deselecting the option to include slices and videos in your output to increase speed. This will also give you the opportunity to check whether your input and the plotting parameters are correct and the output is to your liking. If so, increase the resolution and choose all the output options you want.

As long as the Mathematica notebook isn't closed (or forcefully shut down), your input in the UI is saved. Thus, if you want to change some parameters after a run, it saves you a lot of time. If you want to shut down the program at any point, select Evaluation in the top menu and then choose Quit kernel (last option), followed by local.

Please note that μBerry will automatically create directories for your output in your main directory. If these directories already exist, their contents will be deleted before each run! Especially for the videos, this feature is mandatory - so if you have created data that you like in the directory, save it to a secure location!

2.2 General tab

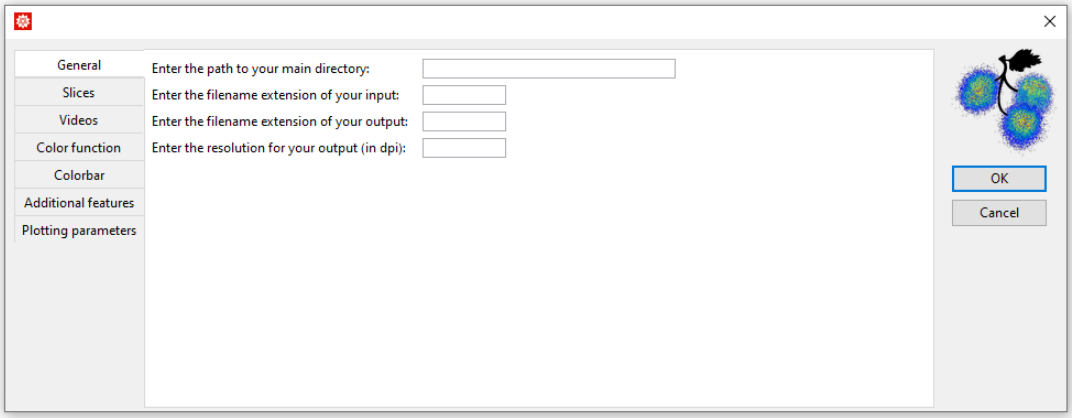


Figure 1: General tab of the user interface

The General tab of the UI is probably the most straightforward of all the tabs. The path of your main directory can be simply copied from Windows Explorer; otherwise, you can of course enter it yourself. Make sure to use \ in the path! The filename extensions should contain the dot followed by the extension. For input files, .3d and .csv work well. For output files, you can only choose the filename extension of images (.png, .jpeg, etc.). All videos made with µBerry will come as .mp4 - if you want a different format, you have to convert it yourself.

2.3 Slices tab

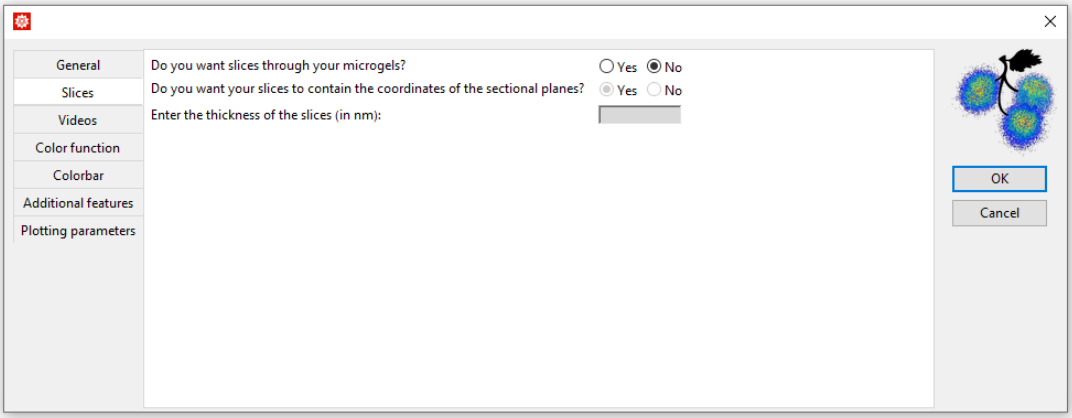


Figure 2: Slices tab of the user interface

If you want your output to contain slices through your microgels, choose yes. By default, the program starts with no selected. If you chose yes, you have to enter the thickness of your slices (in nm). Additionally, you can choose an option to add the coordinates of the current slice to the image.

2.4 Videos tab

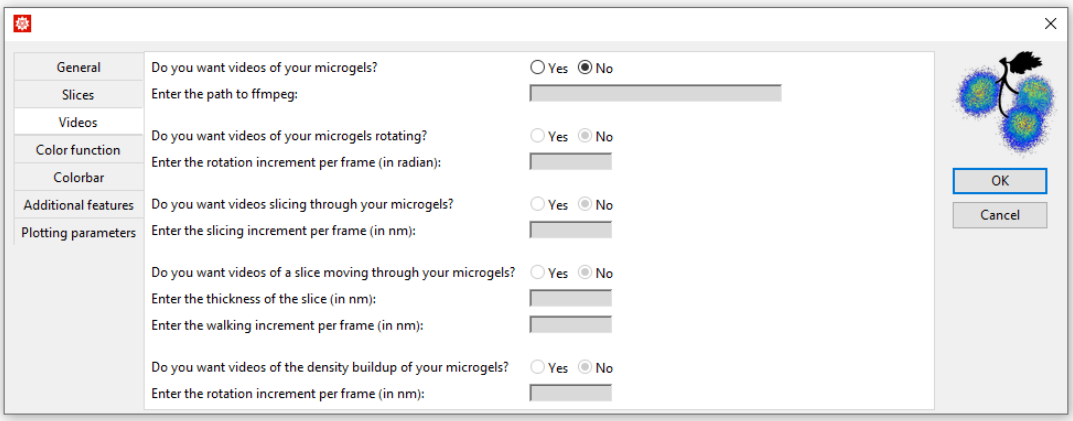


Figure 3: Videos tab of the user interface

If you want your output to contain videos of your microgels, choose yes. Since videos are very time consuming (in comparison to simple images or slices), you can choose which types of videos you want in your output.

If you want videos of your microgels rotating around the x, y, and z axes, choose yes and enter the rotation increment in radian per frame. The microgels will always make one full rotation (360° or 2π). The example videos which come with the manual were made with an increment of 0.005 rad per frame.

If you want your output to contain videos slicing through your microgels along the x, y, and z axes, choose yes and enter the the slice increment in nm per frame. The video will always slice through the entire microgel and then return the microgel on the way back. The example videos which come with the manual were made with an increment of 2 nm per frame.

If you want your output to contain videos of a single slice walking through your microgels along the x, y, and z axes, choose yes and enter the thickness of the slice in nm and the walking increment in nm per frame. The slice will walk through your microgels twice. The example videos which come with the manual were made with a slice thickness of 100 nm and a walking increment of 2 nm per frame.

If you want your output to contain a video of your microgel being build up from the densest points, choose yes and enter the rotation increment in radian per frame. The microgel will always make one full rotation around the z axis and complete the buildup with the last frame. The example video which comes with the manual was made with a rotation increment of 0.005 rad per frame.

Since Mathematica itself produces crappy videos, the program ffmpeg is used instead. This means that Mathematica will create the individual frames of the video and ffmpeg will stitch them together to make the video. This process is by far the most time consuming part of

µBerry! Usually, depending on the input parameters, the videos need roughly 5 min each to be created; so choose carefully, if you want videos / how many you want to make.

2.5 Color function tab

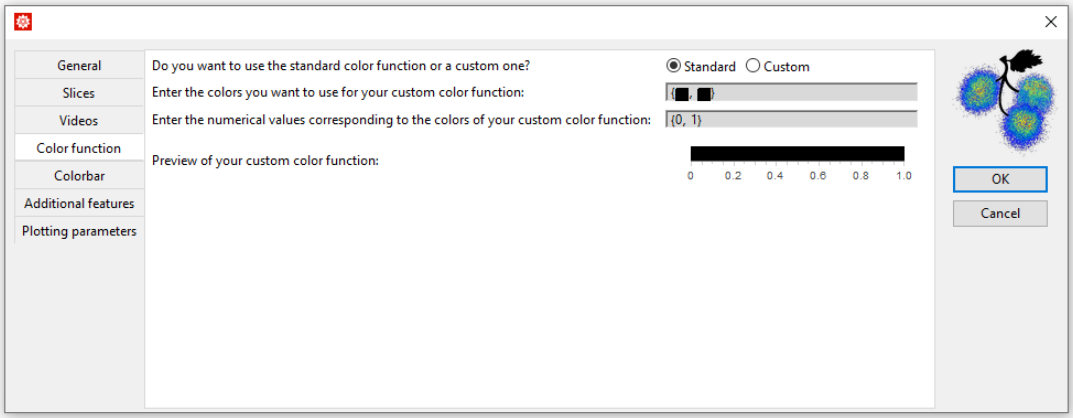


Figure 4: Color function tab of the user interface

By default, µBerry operates in density mode and uses a jet like color function for the graphics. If you do not color your data according to density and want to use a different color function, you have the option to do so by choosing custom.

To make your own color function, you have to enter two lists: a list of colors to be used in the color function and a list of numbers corresponding to the colors. Lists in Mathematica have to be written in { } and individual elements of the lists are separated by commas. As an example, a completely black color function is loaded by default.

There are multiple ways in which you can specify colors in Mathematica:

1. The easiest way is to simple use the name of common colors like Blue, Red, Yellow, Green, etc. Please note that these colors always have to start with a capital letter!

2. You can also use RGB codes as colors. Write RGBColor[red, green, blue] to add the color. Please note that Mathematica doesn't operate with the usual range of 0-255, but with the range of 0-1. Instead of adding the RGB codes of your color, you can also add the hexadecimal code of your number (also using RGBColors[ ]). Please note that the code has to be in "".

3. Mathematica also has a vast number of named colors and color functions which you can use. Choose Palettes from the top menu, then Color schemes to access a list of colors directly in Mathematica. You should use an empty notebook to do this because Mathematica lets you import the color from the menu directly into the code. This way, you don't accidentally add code where it doesn't belong. To use color functions as the basis of your color, write ColorData[Color Scheme in "", number] to get a specific color

(e.g. ColorData["Rainbow", 1] will produce the color at the numerical value 1 of the color scheme Rainbow).

The UI contains a preview of the color function you enter. This means you can check and alter the color function without having to make the output. Please note that the preview is updated only every time you switch objects like text boxes or radio buttons. Additionally, once you have entered the color specifications and change objects, Mathematica should transform your code into little squares in the color you have chosen. In case you have made a wrong input for the color function, the preview will turn dark red / brown color. This usually means that the number of colors and numbers doesn't match.

2.6 Colorbar tab

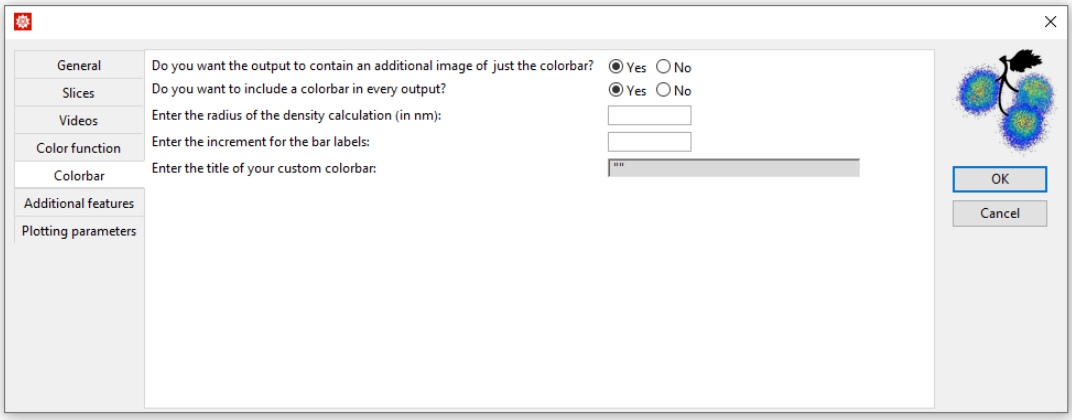


Figure 5: Colorbar tab of the user interface

If you want your output to contain an additional image of just the colorbar, choose yes. It will be saved in the directory Images. By default, this option is selected.

If you want every single image / video to contain the colorbar, choose yes. If you choose no, only the grids will have the colorbar, as they use a slightly different colorbar (it's longer).

If you have chosen the standard color function, you have to enter the radius of your density calculation in nm. This information is needed for the correct title of the colorbar. Additionally, this value will be taken for the calculation of the density (if needed, see below).

No matter which kind of color function you have chosen, you have to enter the increment for the bar labels. The colorbar will always have a label at its minimum and maximum value and additional labels depending on the increment you have chosen. In the case of a density color function, you most likely have to guess a first value (as you most likely don't know the maximum density) and then correct the increment in a second run.

If you have chosen a custom color function, you have to enter the title of the colorbar:

1. If your title is a simple word / phrase, just enter it as a string (in "").
2. If bold or italic elements, use Style[string, Bold / Italic].
3. If you want your title to contain superscript or subscript elements, use ToString[Superscript[normal string, superscript string], StandardForm] (or Subscript...)
4. Strings can be connected to each other by using <>.

2.7 Additional features tab

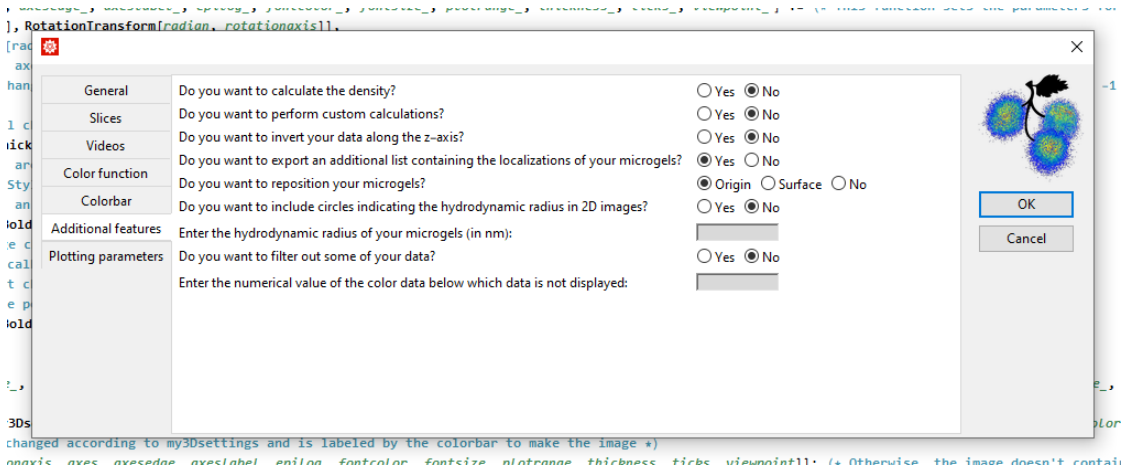


Figure 6: Additional features tab of the user interface

In case you haven't already calculated your data's densities, you can do so with this program. The density calculation takes a little time (a data set containing roughly 15,000 data points takes roughly 2-3 minutes). If you choose this option, the program will automatically export .csv files containing your input data and the calculated densities to a directory called Density in your main directory. This way, you won't have to do the calculation again in the future.

In case you want to perform your own calculations, you can choose yes. In this case, you have to manually add code to the program to make your calculations. In the standard code, there is an empty if condition which you can fill with your code (press CTRL + f to search for (* Custom calculations *) to find it.

1. At this point of the program, the microgel data is stored in a list called data. To access individual microgels in the list, use data[[# of microgel]]. Please note that the first element of a Mathematica list is indexed as 1 and not as 0, as is common in other languages.
2. If you want to access parts of a microgel's data, use data[[# of microgel]][[# of row, # of column]]. Instead of entering a number, you can also use All to select all rows / columns. For example, [[All, 3]] will select the entire third column (meaning all z values). If you want to access multiple adjacent elements, use number ;; number, e.g. [[1, 1;;3]] will select the first three entries (x, y, and z coordinates) of the first point.

3. Most likely, you want to perform the calculations in a loop for all your microgels and a for loop would be the standard choice for this in many languages. Do NOT do this in Mathematica as for loops are very slow and prone to bugs! Instead, use a do loop (which is basically the same as a for loop but actually works in Mathematica). Do loops are entered as Do[commands, {iteration variable, min value, max value}]. If you want to use more than one command in the loop, seperate them by using semicolons.

4. After each line of a code, you should always include a semicolon. Otherwise, Mathematica will print everything you do in the notebook which dramatically decreases speed.

5. Simple calculations like adding, subtracting, multiplying, and dividing are performed using +, -, *, and /. You can also combine them with = to automatically apply the operation to the variable, e.g. variable *= 2 is the same as variable = variable * 2.

6. If you want to use if conditions, write If[condition, true, false, neither]. The options false and neither are optional; only true has to be added (although it can be left empty).

7. If you want to select only certain parts of your data, you can use Select[list, condition]. The condition in this case has to be followed by an additional &! If you want to access certain numbers of your list, you can use #[[number]]. For example, the condition #[[3]] => 0 & will select all entries of the list whose z coordinate is larger or equal to 0. If multiple conditions have to be true, combine them using &&. If one true condition out of a group of conditions is sufficient, you can use ||.

8. If you want to sort your list, you can use SortBy[list, condition]. In this case, the condition should be one entered as #[[# of column]] &. Depending on how you want your list to be ordered (ascending or descending), you may have to add a - before the condition (as a simple workaround).

9. If you need additional commands for your custom calculation, you can use the online resources of Mathematica. Mathematica has explanations (with examples) for virtually any command and there are many great tips in online forums etc. Alternatively, you could try going through the actual code of µBerry. There are many operations which cannot be discussed here in detail and which may be useful for you. In general, the code has comments to explain how certain commands work and why they are used.

If you want your data to be inverted along the z axis, choose yes. Due to the new program for 3D analysis, this option will most likely become obsolete for future measurements, but old data could still need to be inverted.

If you want a list of the number of localizations of all your microgels, choose yes. A txt file containing the localizations is exported to the Images folder.

In general, the program allows you to position the microgel in 3 different ways. If you choose origin, the microgel is shifted to the origin of the coordinate system. If you choose surface, the microgel is shifted to the origin of the coordinate system and then it is shifted along the z axis to only positive z values. If you choose no, the microgels aren't moved at all. Depending on which choice you make here, you have to enter different plotting parameters (see below).

In case you have chosen to reposition your microgels to the origin, you have the option to include circles in your 2D images (xy face, xz face, and yz face). These circles indicate the hydrodynamic radius which you have to enter (in nm).

µBerry also comes with the option to filter out data according to the forth column (and the forth column only!). If you choose yes, you can enter the minimum numerical value which data points need to have to be displayed. This option might be useful to remove stray data points with e.g. low density to make better images. Obviously, this option is turned off by default and should be used with caution as simply deleting data is bad scientific practice...

2.8 Plotting parameters tab

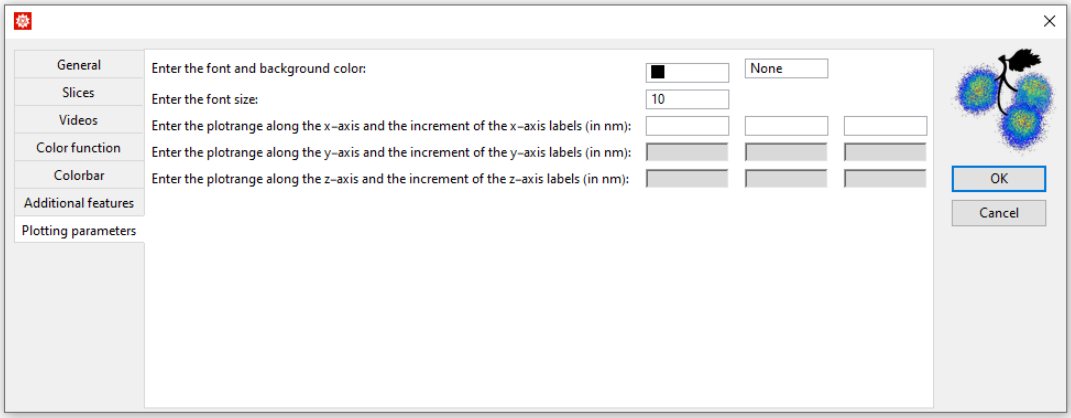


Figure 7: Plotting parameters tab of the user interface

The last tab contains information needed to plot your data. You can enter different font and background colors for your images (see above for information about colors in Mathematica). By default, the background is set to transparent (None).

You can change the font size of the image by entering the value. By default, the font size is 10. If you choose a much larger font size, the labels will unfortunately start overlapping the axes and each other in the 3D images.

Depending on which positioning method you have chosen previously, you have to enter the following data:

1. If you have chosen origin, the program will create a cube around the microgel. You have to enter the minimum and maximum coordinate along the x axis which should still be displayed in the images (effectively the size of the cube). Additionally, you have to enter the increment for the axis labels. 0 will always be labelled! Since a cube has identical sides, you only have to enter the values for one axis.

2. If you have chosen surface, the program will create a square prism around the microgel. For this, you have to enter the minimum and maximum coordinate along x (which will

also be used for the y axis) and the minimum and maximum coordinate along z. You can give each axes different increments.

3. If you have chosen no, the program will create a cuboid around the microgel and you have to enter the minimum and maximum coordinates as well as the labelling increments for each axis individually.