Postprocessing in Paraview

Paraview is a powerful postprocessing tool. We shall explore how to handle the VTK files generated in EX 4.

Installation

- Go to the following link: https://www.paraview.org/download/
- Download the `ParaView-5.8.0-Windows-Python3.7-msvc2015-64bit.exe` file
- Launch the executable and perform the standard installation (No need to uncheck anything)
- Paraview takes approximately 1.1GB.

Fixing the VTK files

The VTK files that are generated by `FiPy` have a bug and need to be modified which can be done quickly with a short python script or a one-line command in linux. The python script `fix_vtk.py` is in the EX4 folder. Dont worry about the details of the script if you arent interested. Just run the following command at the end of the simulation

```
python fix_vtk.py
```

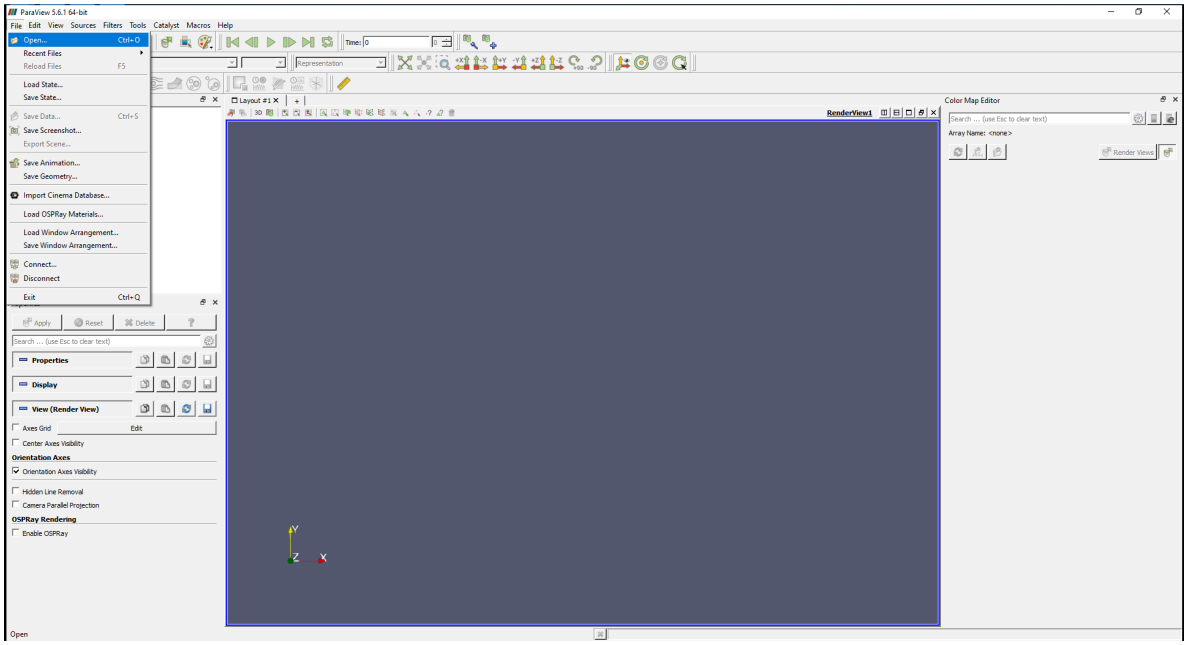
If you are using Linux / WSL, you can use the following line of code which is arguably more elegant:

```
sed -i "s/^41$/9/" *.vtk
```

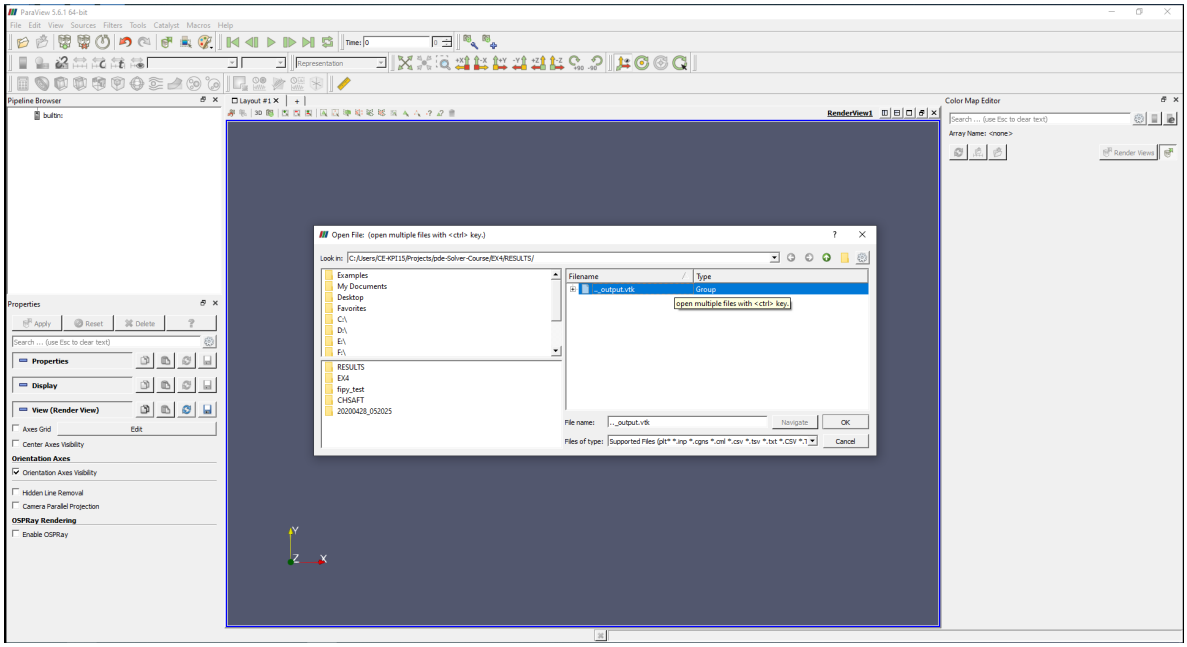
Opening the VTK Files and doing basic visualization

Paraview is quite smart and does not require the user to manually open every single VTK file. It can recognize patterns in the file name and generate a series.

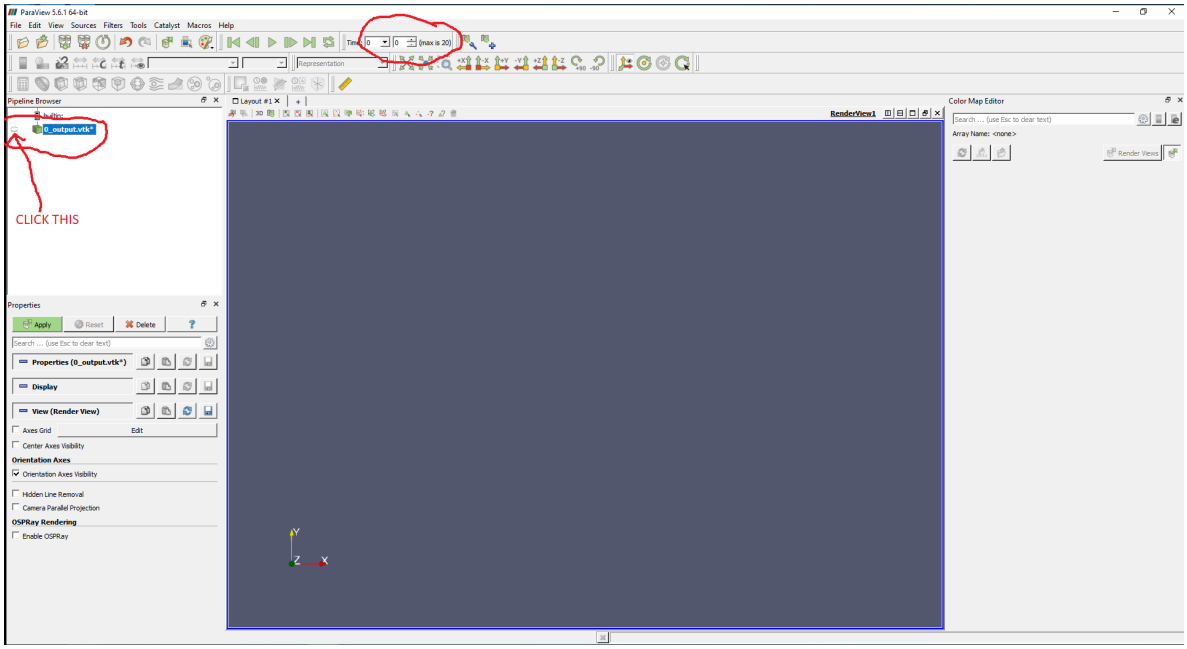
Click `File` and then `Open`:



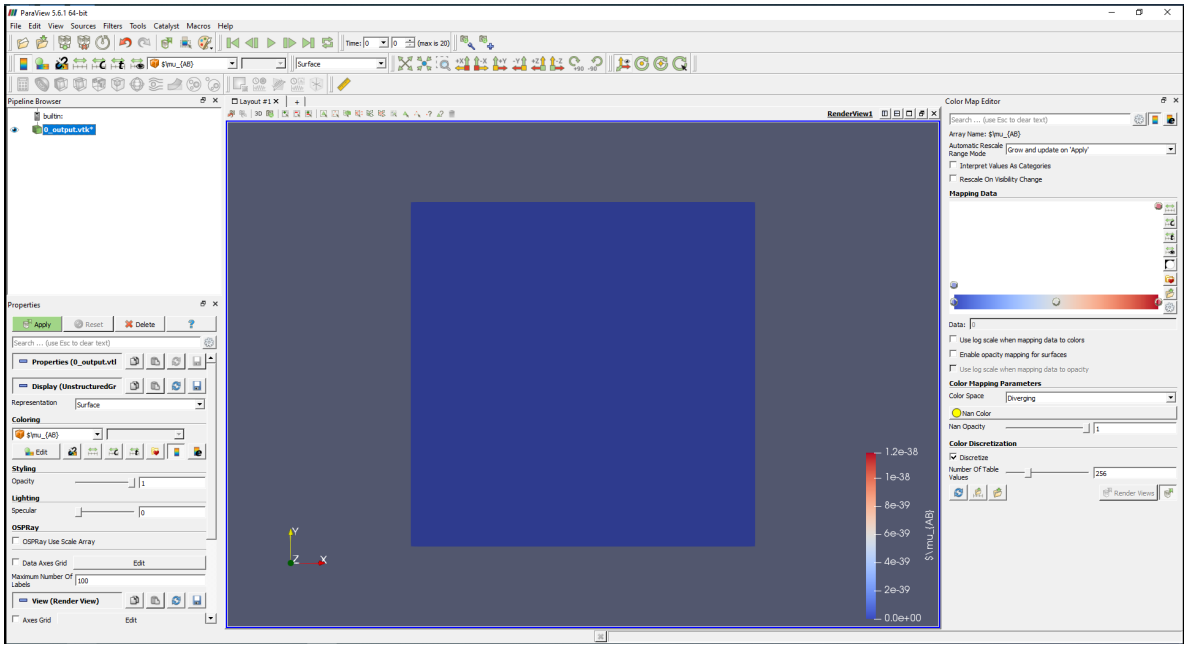
You are able to observe that in the folder, Paraview has lumped all the VTK files into one group. You just need to load that single file into the software.



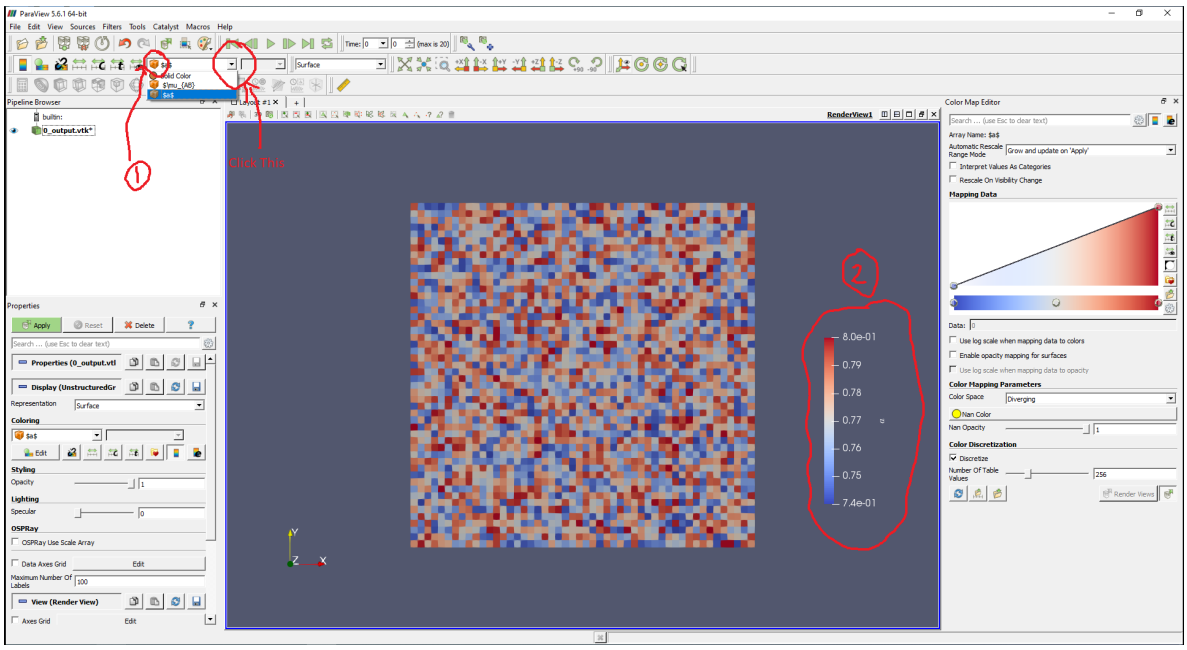
If you observe the parts of the user interface circled in red, you will see that the file has been successfully loaded.



Click the eye icon to see the data:

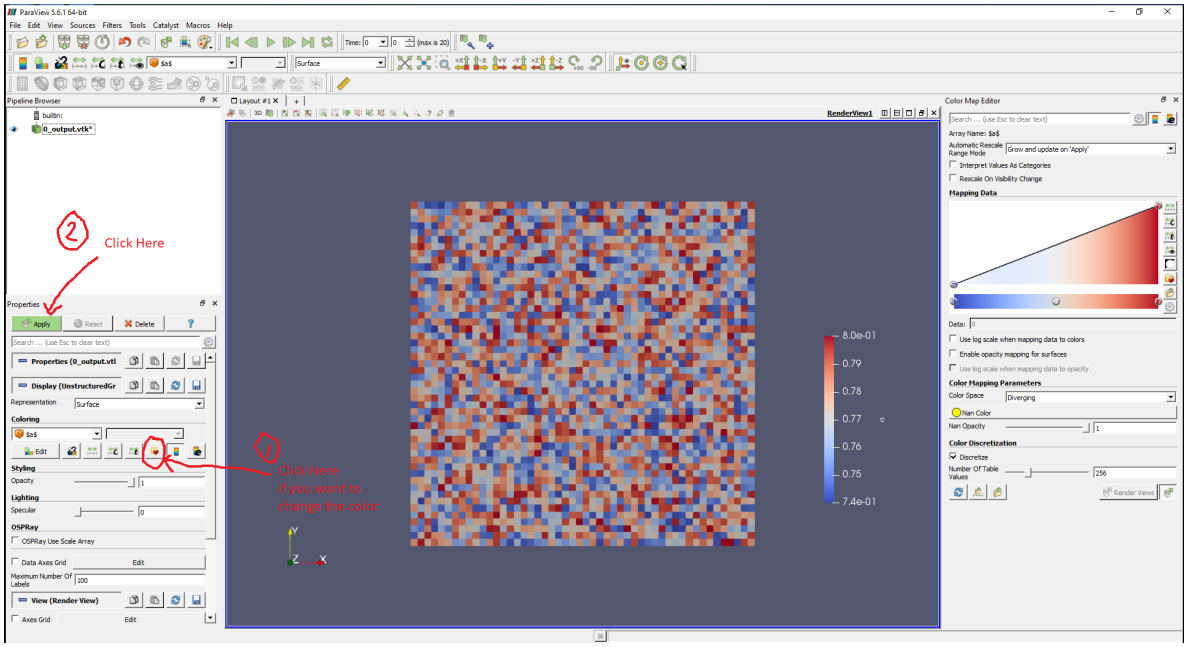


The data we see is the chemical potential μ_{AB} at $\tilde{t} = 0$. To change the view to the composition field, click on the button listed on the next screenshot and select the a field:

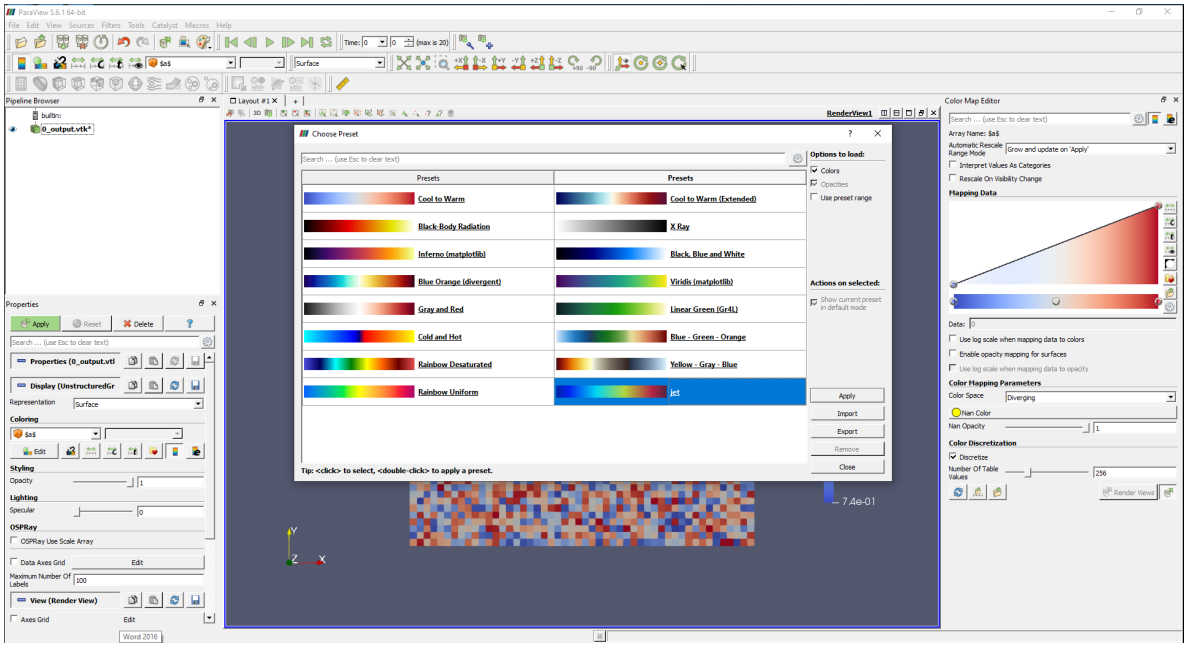


If you observe point 1, you can see that there is a box icon next to `$a$` sign on the list of fields. This indicates that the data is the cell data. This explains why the the image looks so pixelated. The next point 2 shows the range of the colour bar. $a(t=0)$ has an average of 0.77 and a range of ± 0.03 which is as indicated in the simulation step-up.

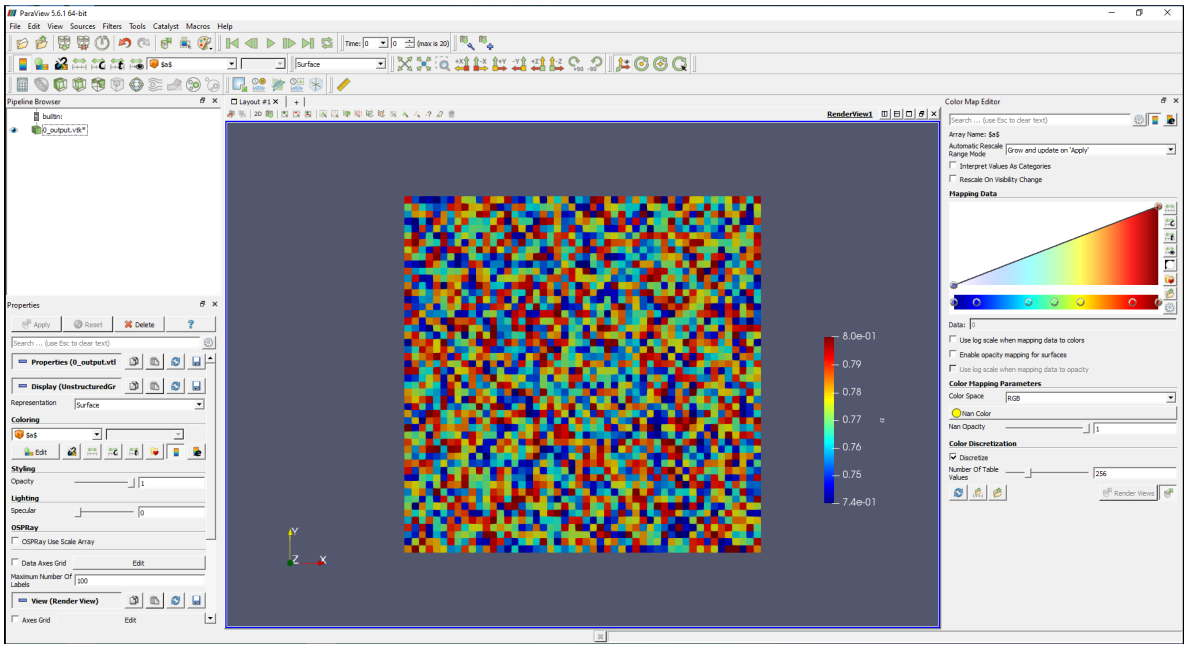
Next we need to smooth out how the data looks. This requires applying a `Cell Data to Point Data` filter. To do this, we first need to click the apply button on the properties panel. We could also apply a different colour scheme which requires us to click the button in point 1:



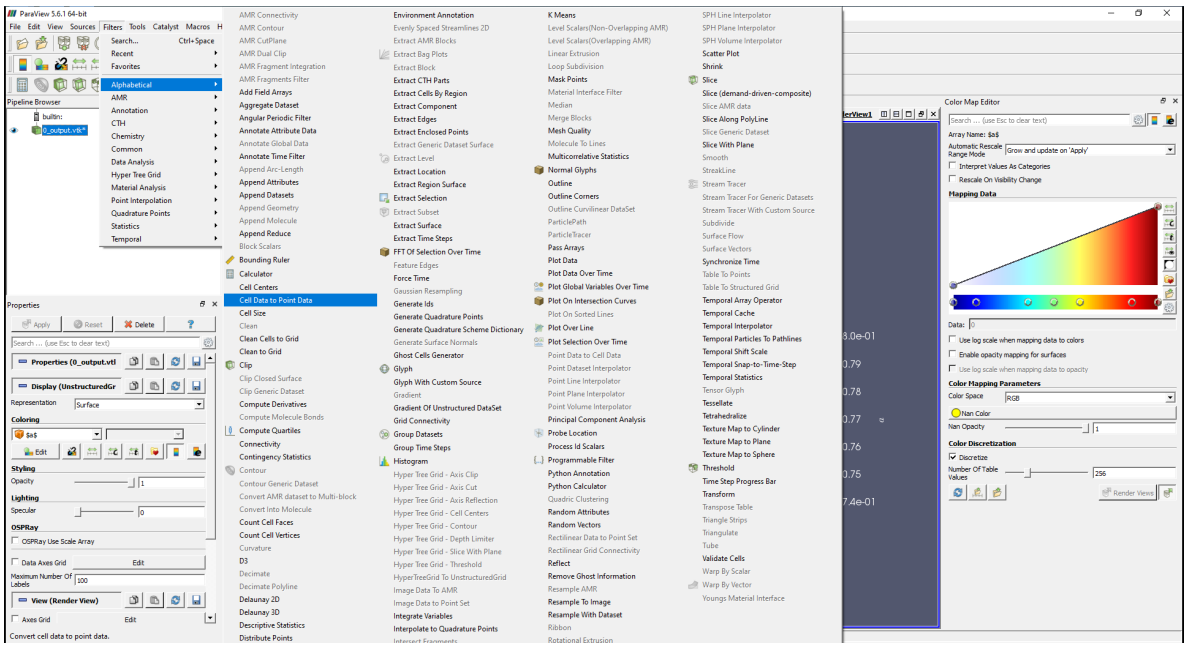
The list of various presets available can be found here. We shall select the `Jet` preset and apply it. There are more available:



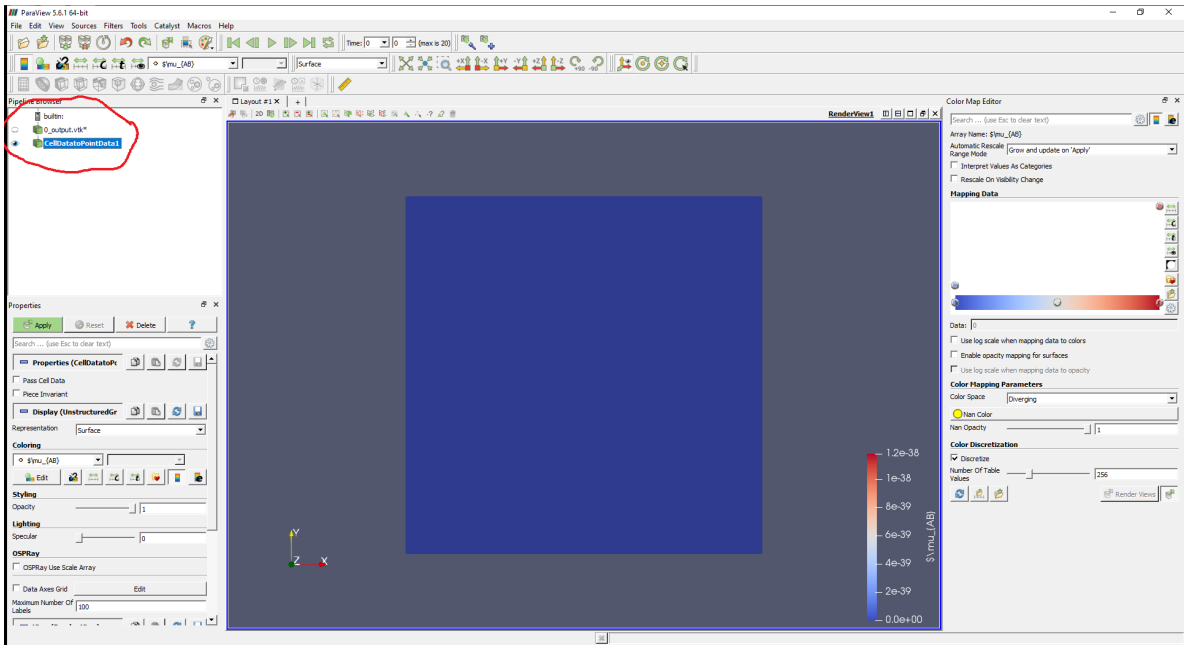
We can see the impact of selecting the new colour scheme and clicking `Apply` in the previous image :



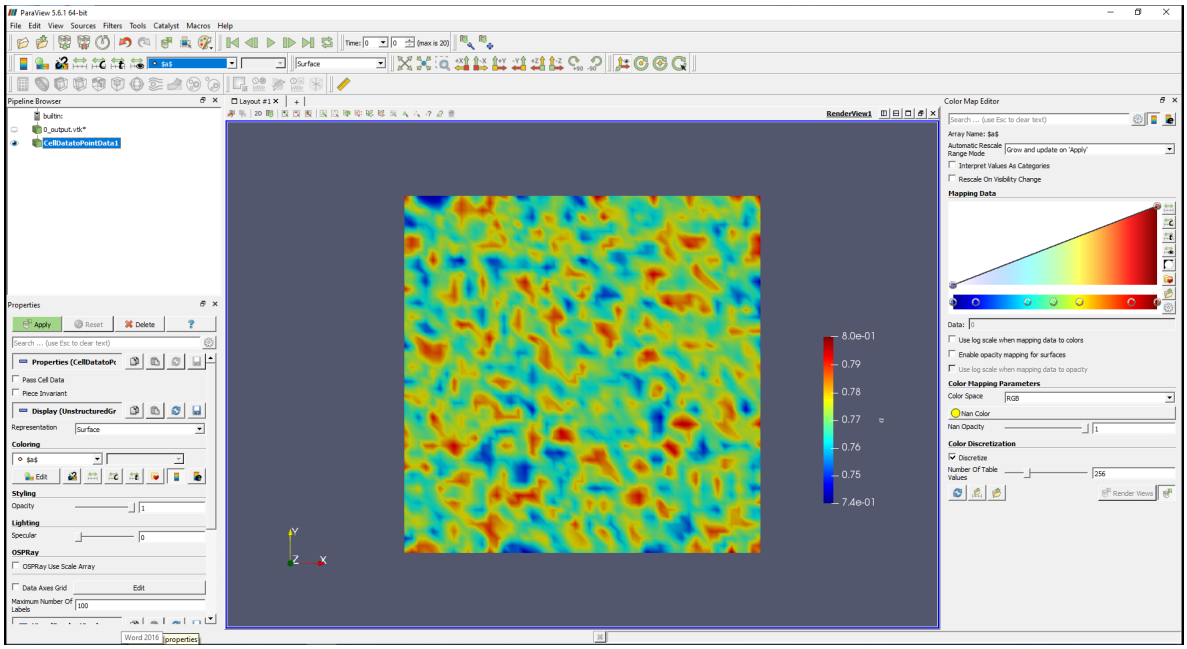
We can now apply the `Cell Data to Point Data` Filter. First highlight the `0_output.vtk*` in the `Pipeline Browser` by clicking it once, then click `Filters` -> `Alphabetical` -> `Cell Data to Point Data`:



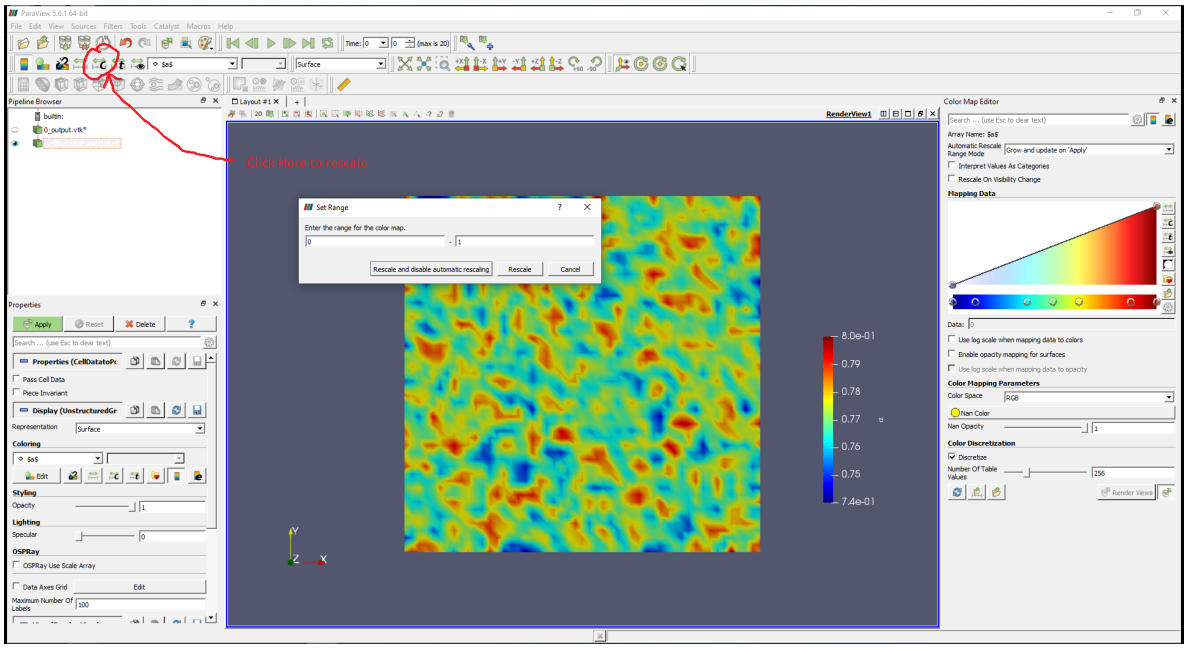
We can see a new item in the pipeline browser which shows us the previously applied filter:



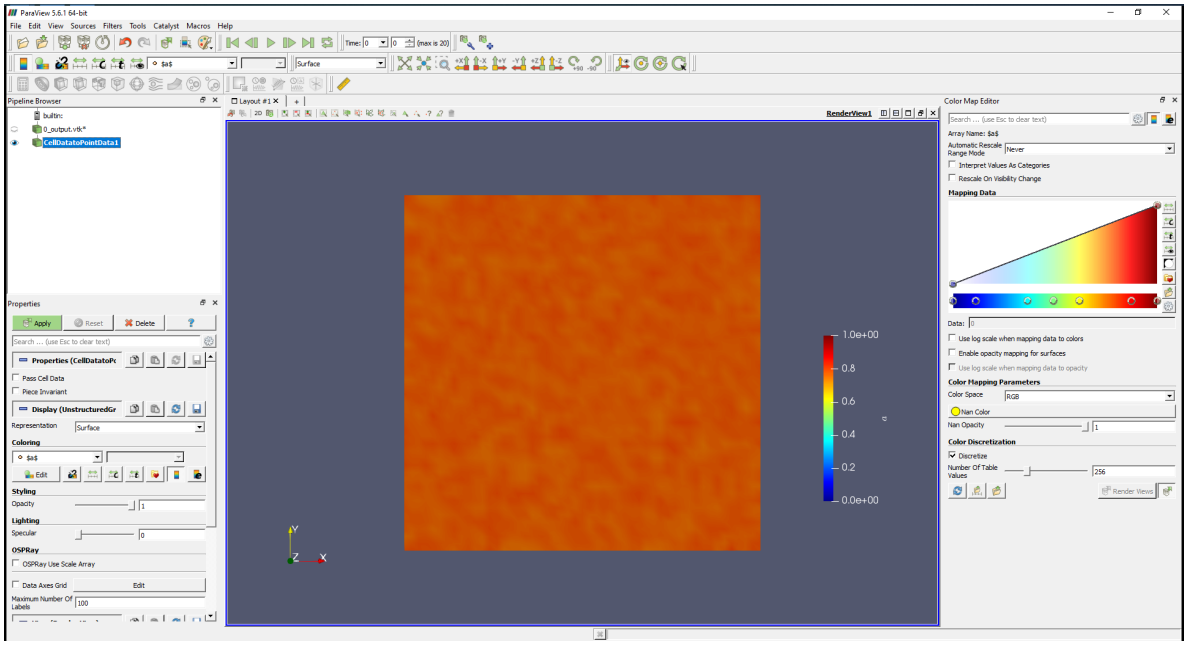
We then re-select the a field and see how the smoothened field looks like:



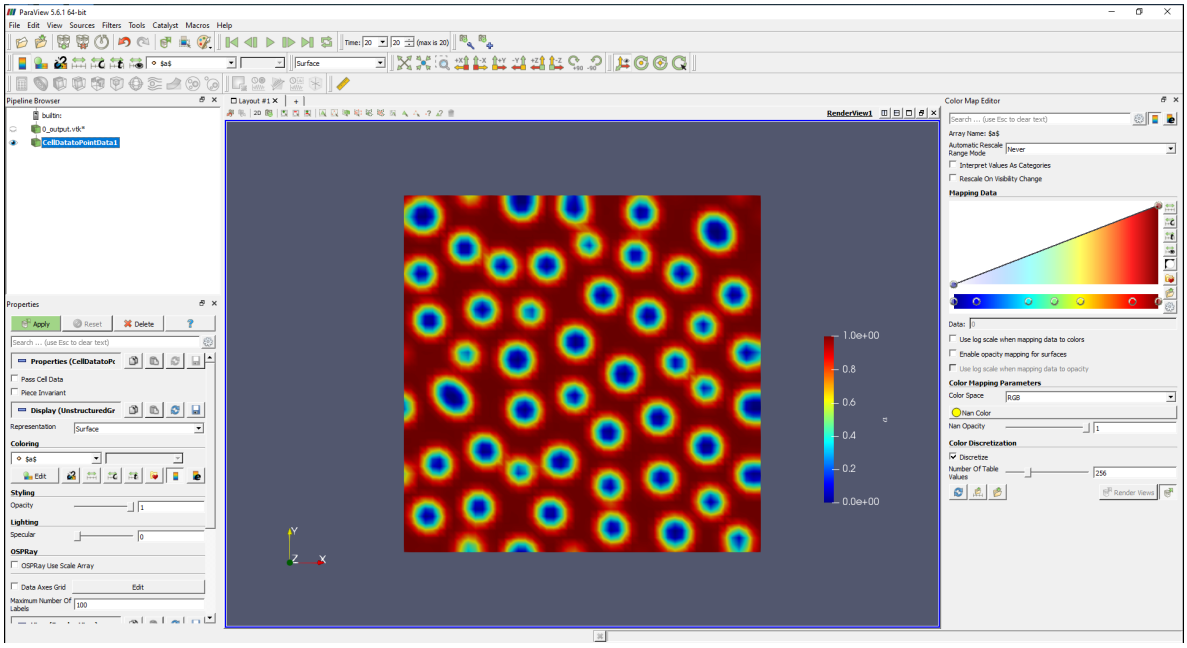
We can now rescale the data so that the color bar reflects the full range of the mole fraction i.e. $a \in [0, 1]$:



Click `Rescale and disable automatic rescaling` and we can see how the data looks when its rescaled. Note how the color bar reflects the updated scaling:



We can now view the how the field evolves. All we need to do is to click the play button. Let us see the results at the final timestep.

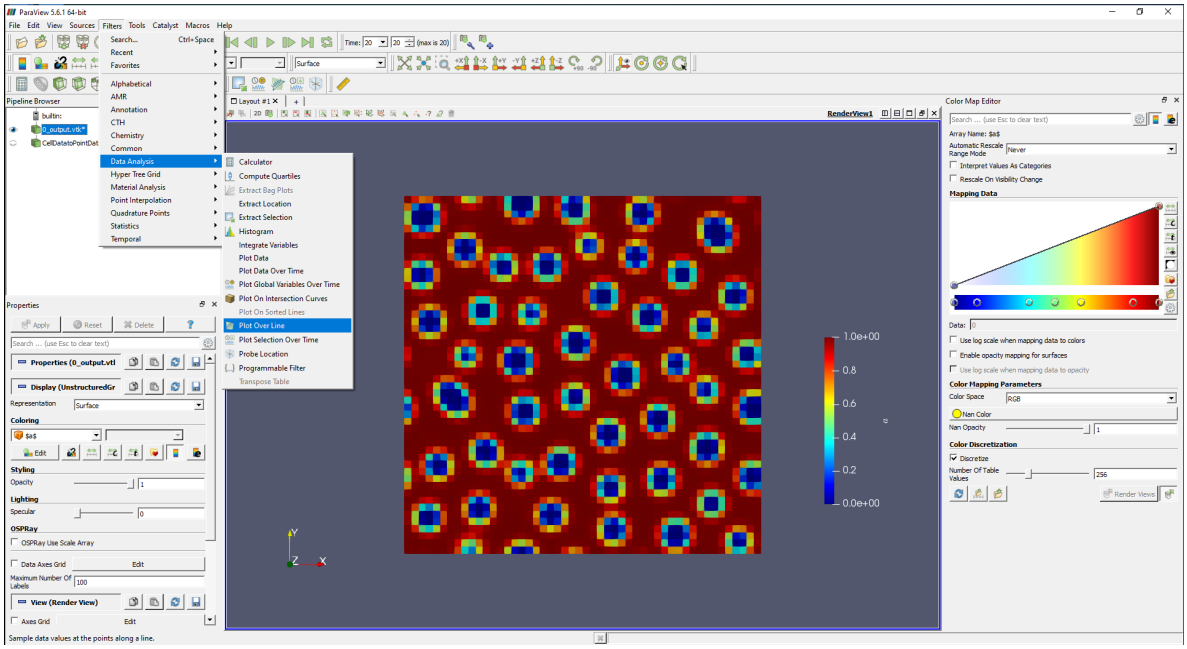


We can see how the the demixing occurred and small spherical particles appeared.

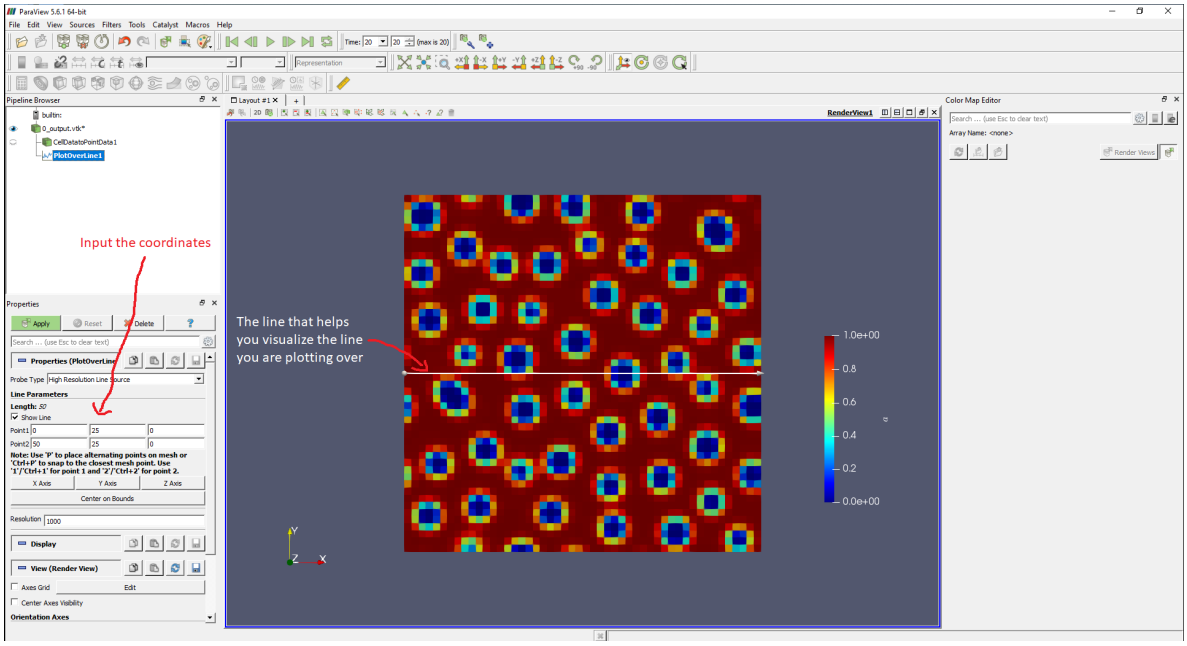
Creating a 1D plot

Suppose we want to extract a 1D plot of the a across a line. That can be done in Paraview quite easily.

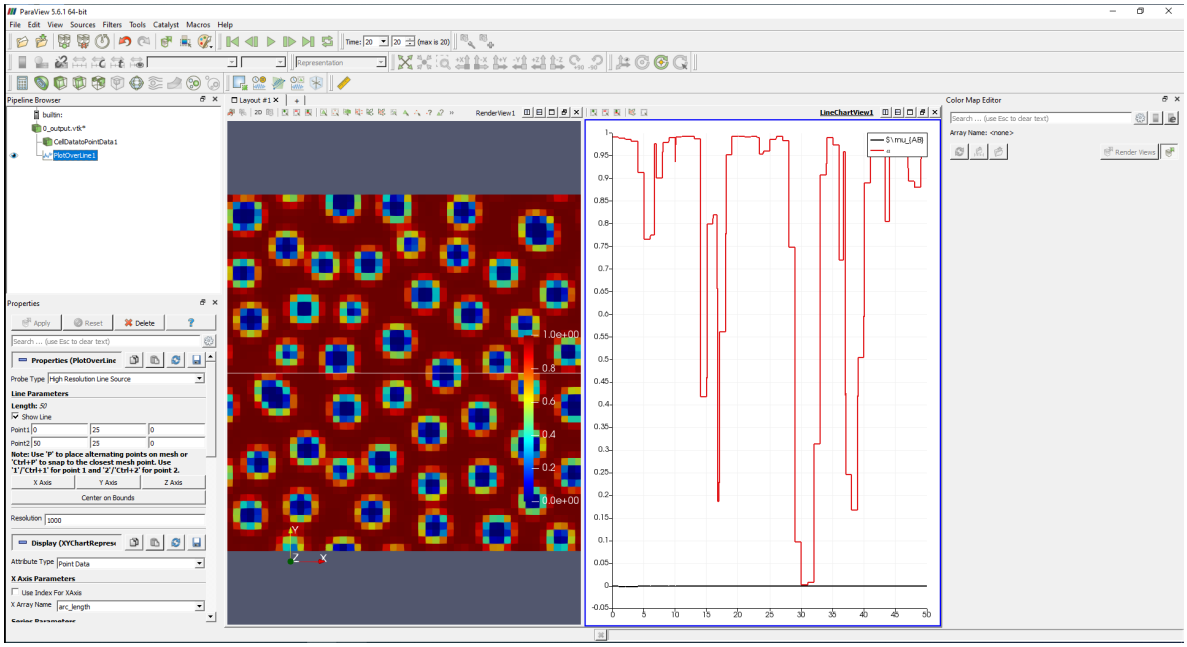
We first highlight the data in the pipeline browser which in this `0_putput.vtk*` and apply the `Plot Over Line` filter.



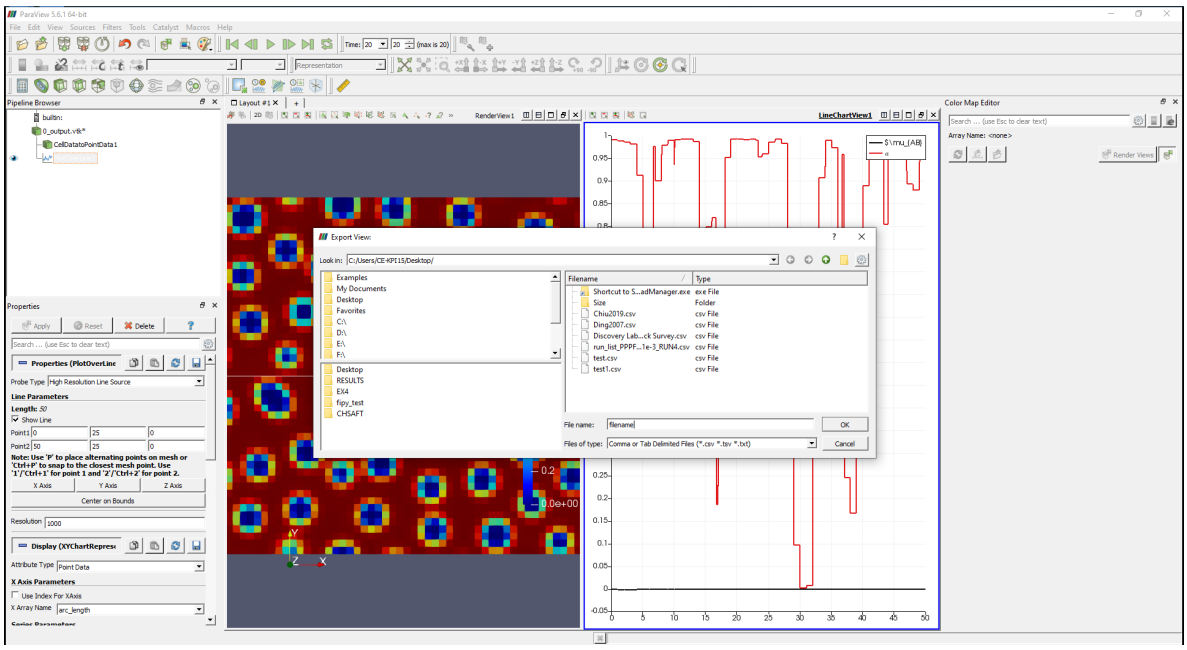
On the `Properties` box on the left, we can see the settings for the plot. There is a line that appears that helps to visualise the line that will be used to plot. We need to define `Point 1` and `Point 2` in the `Properties` box which defines the start and end of the line.



Once we picked a line that we desire. In this case, I want to plot across the center of the 2D domain as shown above, we just click `Apply` in the Properties box. This creates a new window where the plot is displayed. Both variables a and μ_{AB} are present. This is essentially the same as doing a 1D simulation.



Suppose we want to export the plotted data into an excel (.csv) file so that we can do our own plotting, that is readily achievable. In the pipeline browser, highlight the `PlotOverLine1` filter and click `File` -> `Export Scene`. We see that we can nicely export the data as a `.csv` file.



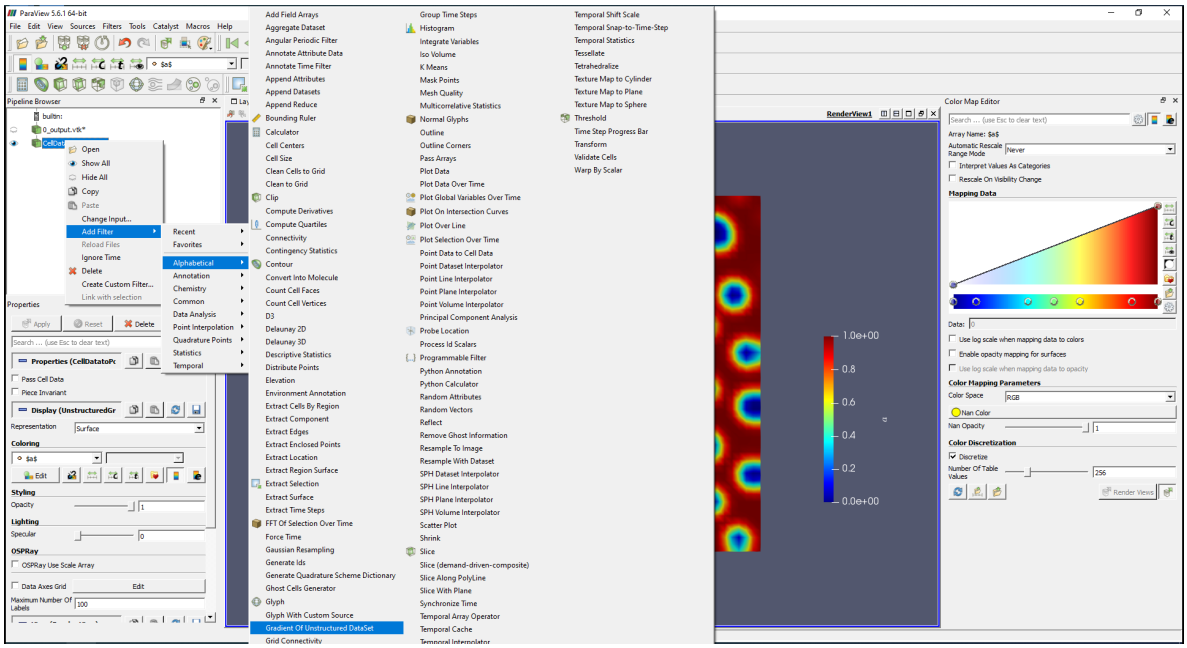
Evaluating the total Gibbs energy over time.

If we go through the notes for exercise 4, we can find the following integral which if evaluated lets us see how the total energy of the system evolves:

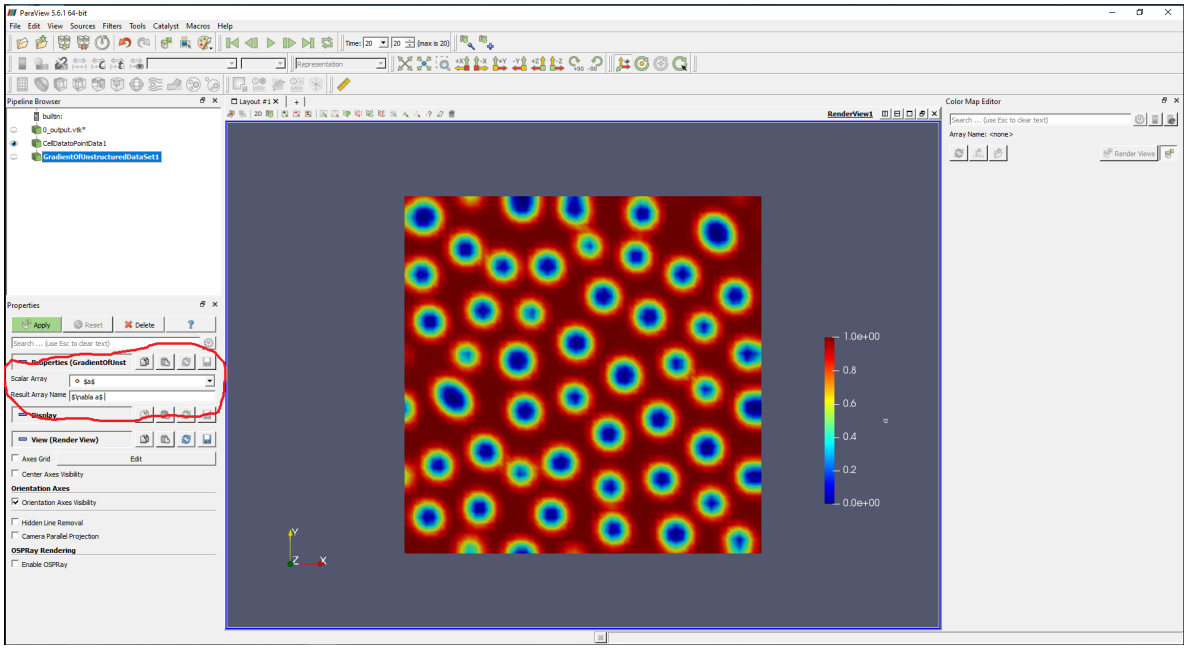
$$G_{system} = \int_V g(x_1) + \frac{\kappa}{2}(\nabla x_1)^2 dV$$

We can use Paraview to evaluate this integral. However, the workflow is not so trivial as we can trivially evaluate the above integral using one step in Paraview. We have to break it down one step at a time.

We first need to create a separate variable to capture ∇x_1. Select `CellDatatoPointData1` in the pipeline browser and apply the `Gradient of Unstructured Dataset` filter:



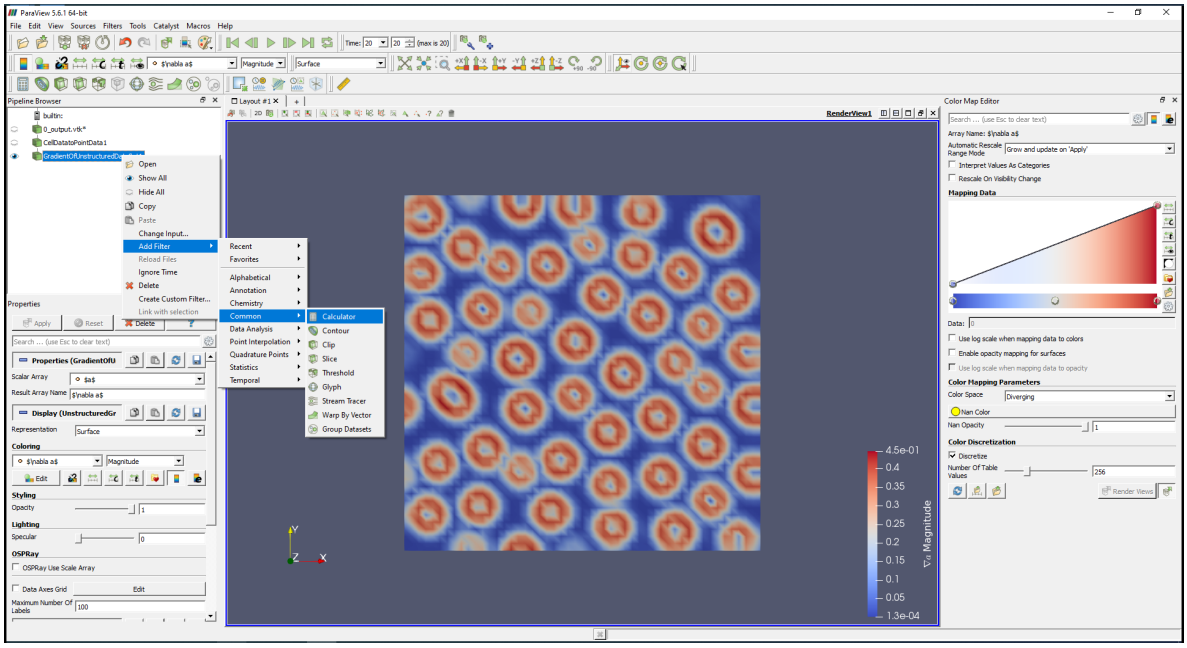
Select the desired variable you want to evaluate the gradient of which in this case is listed as `$a$` and give the new variable a name (I chose `$\nabla a$`). Click Apply.



Now that we have access to the ∇x_1, we can use the `Calculator` filter to calculate the following equation at each point:

$$G = g(x_1) + \frac{\kappa}{2}(\nabla x_1)^2$$

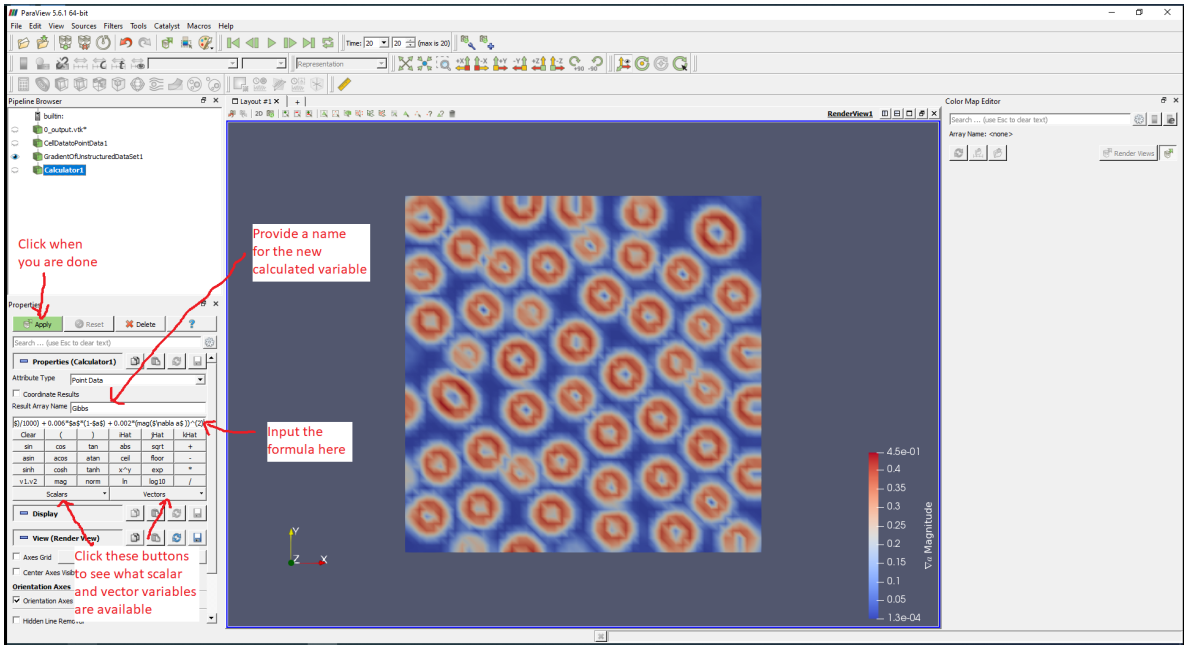
To apply the calculator filter, select the `GradientofUnstructuredDataset1` in the pipeline browser and select the `Calculator` filter as follows:



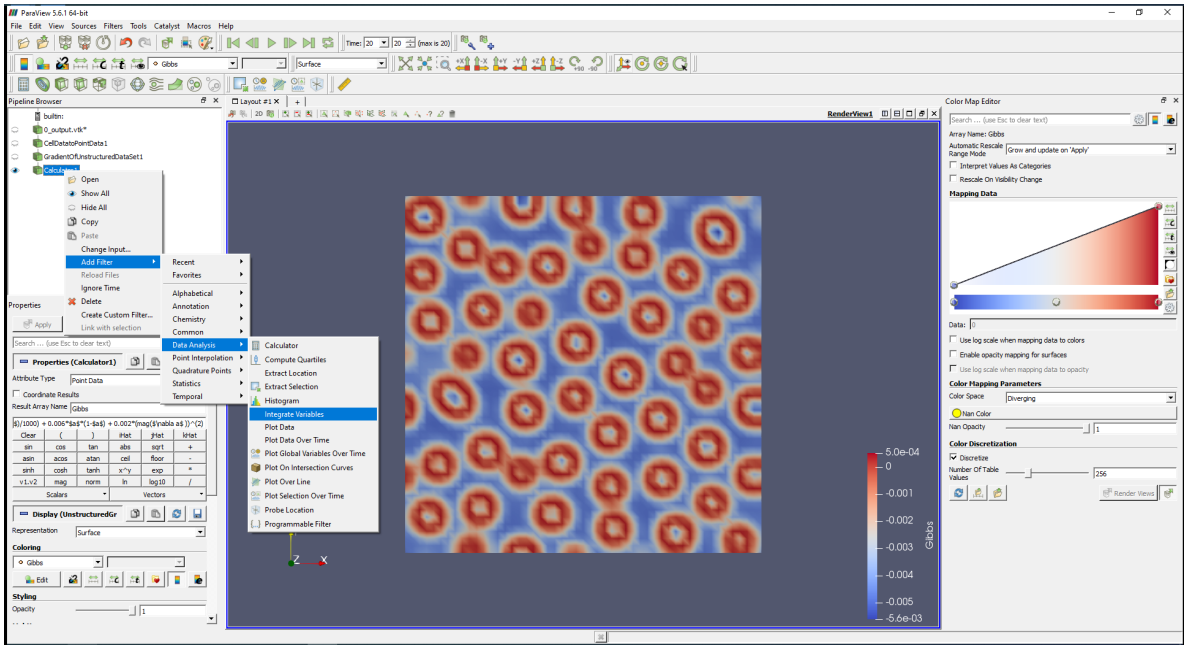
We can then input the equation we need to calculate. You should note that ∇x_1 is a vector and you should input its magnitude into the calculation. The variable name can be specified by changing the input in the `Result Array Name` field.

The formula is inputted as follows:

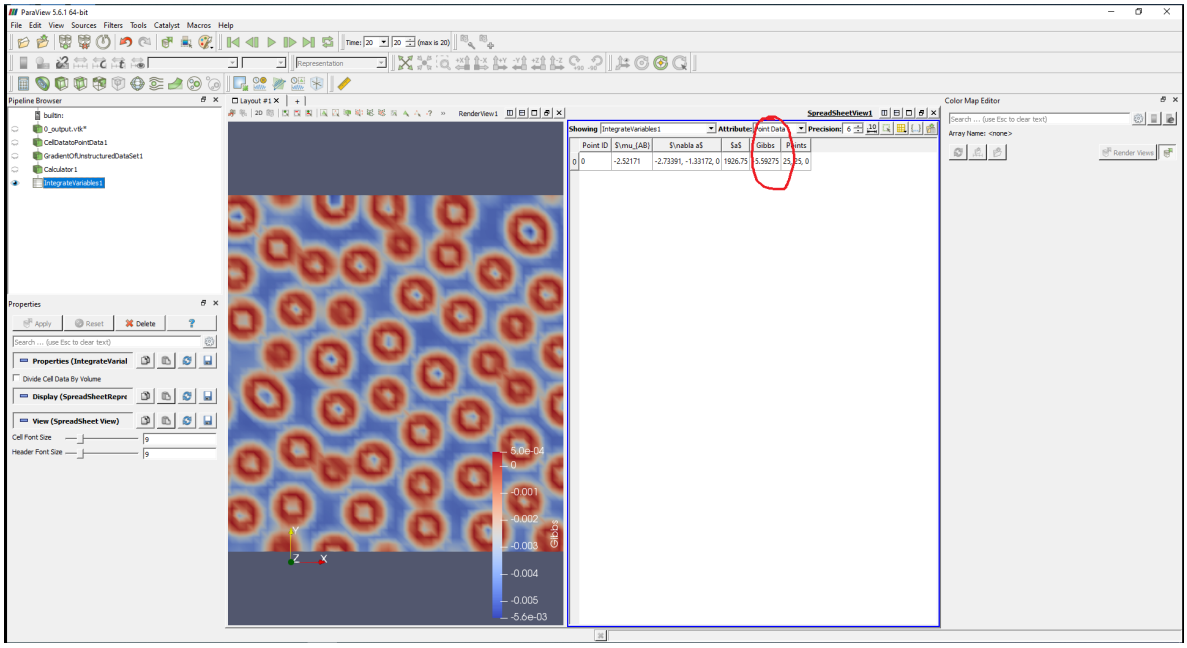
```
(ln($a$)/1000) + (ln(1-$a$)/1000) + 0.006*$a$*(1-$a$) + 0.002*(mag($\nabla a$
))^(2)
```



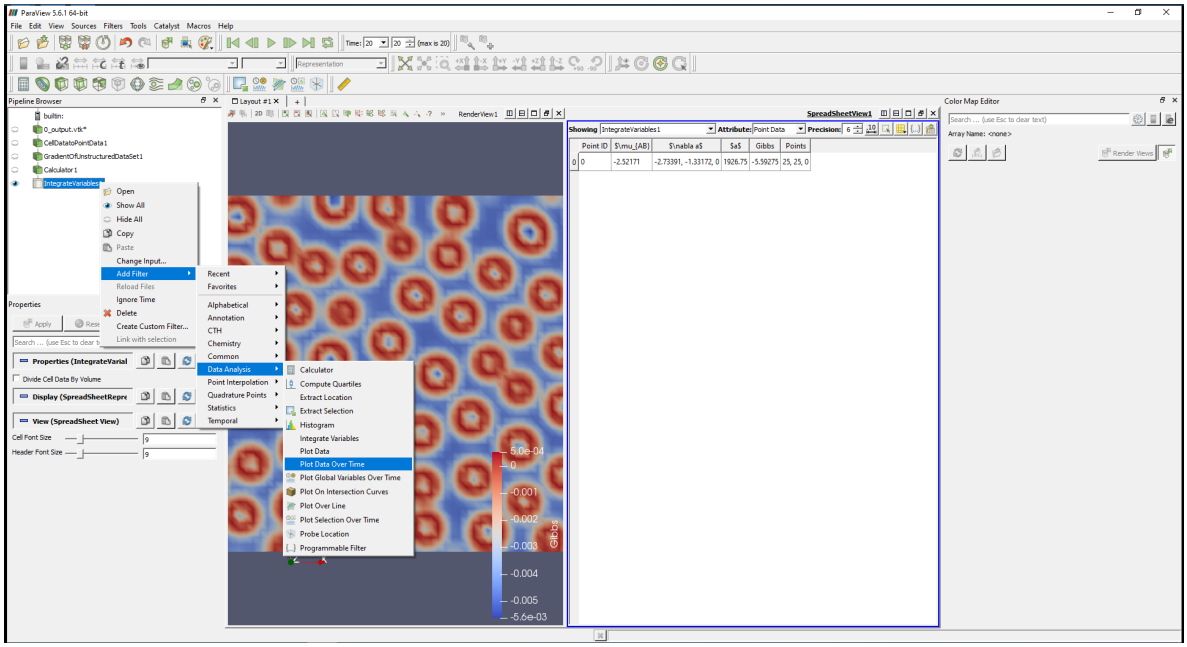
We now have generated a Gibbs variable. We can proceed to integrate this variable. We can apply an `Integrate Variables` filter. Selected the `Calculator1` in the pipeline browser and apply the `Integrate Variables` filter as follows:



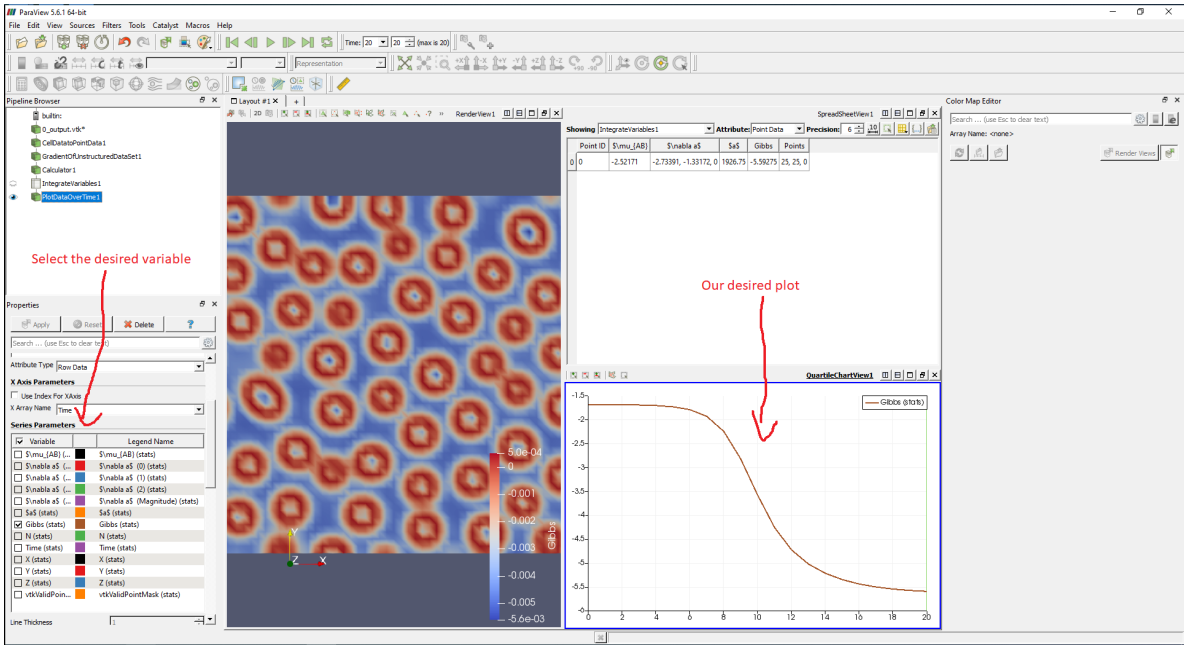
When you click `Apply`, you should see a window pop up that gives a spreadsheet view and you can see our variable of interest `Gibbs`!!!:



Now is the last step where we obtain the time series. We need to apply the `Plot Data Over Time` filter. Selecte the `IntegrateVariables1` filter in the pipeline browser and apply the `Plot Data Over Time` filter:



We are able to select multiple variable to plot over time. But we are only interest in Gibbs. Unselect the rest and we should be able to see how G_{system} evolves with time:



You can similarly export the data and plot it yourself as we did in the 1D plot example.