APPENDIX A

A fast manual to use the fargo_tools package

This is a brief manual to introduce FARGO3D code users who want to learn the use of fargo_tools. This package manual includes how to get the data from the hydrodynamical simulations and to visualize 3D data with the Paraview software. It is assumed you use some Linux distribution and you are familiarized with the terminal environment.

The text in verbatim after the $ symbol means the commands that you need enter in the terminal. If you copy these command lines one by one you could follow this tutorial very easy.

A.1 Creating some FARGO3D outputs

Let's start in our home directory, (in this tutorial is assumed that you have the fargo_tools directory in your home and also that you will download fargo3d there in)

```
$ cd
$ pwd
  /home/oscar
```

At first, you need to download the FARGO3D code

```
$ wget http://fargo.in2p3.fr/downloads/fargo3d-1.0.tar.gz
```

the next step is to decompress it (the name can change depending on the FARGO3D version)

```
$ tar -xvf fargo3d-1.0.tar.gz
```

This will create a directory called fargo3d-1.0 (The name can vary depending on the version). Change to the fargo3d directory

```
$ cd fargo3d-1.0
```

Inside this directory you can compile the fargo3d code if you have the correct compilers (see FARGO3D's manual, http://fargo.in2p3.fr/manuals/html/index.html). Let's compile the default p3disof setup, which is a full 3D isothermal circumstellar disk

```
$ make SETUP=p3disof para
                .
                .
                .
All objects are OK. Linking stage

          FARGO3D SUMMARY:
          ===============

This built is PARALLEL (MPI). Use "make seq" to change that


SETUP:      'p3disof'
(Use "make SETUP=[valid_setup_string]" to change set up)
(Use "make list" to see the list of setups implemented)
(Use "make info" to see the current sticky build options)

rm zmax_bound.c ymax_bound.c ymin_bound.c zmin_bound.c
make[1]: Leaving directory '/home/oscar/fargo3d-1.0/bin'
```

I decided to compile it to run in parallel with CPUs, but you can compile it with GPU or in the sequential slow way (again see FARGO3D manual). If the compilation does not mark an error you are ready to generate your data, if it does, you need to check your compilers.

It is time to run the simulation

```
$ mpirun -np 8 ./fargo3d -m setups/p3disof/p3disof.par
                .
                .
                .
Doesn't feel the disk potential
Doesn't feel the other planets potential

Found 32 communicators
Found 32 communicators
Found 32 communicators
Found 32 communicators
Found 32 communicators
Found 32 communicators
Found 32 communicators
Found 32 communicators
OUTPUTS 0 at date t = 0.000000 OK
........TRYING TO OPEN ./outputs/p3disof//FG000000/
TRYING TO OPEN ./outputs/p3disof//FG000000/
TRYING TO OPEN ./outputs/p3disof//FG000000/
TRYING TO OPEN ./outputs/p3disof//FG000000/

TRYING TO OPEN ./outputs/p3disof//FG000000/
TRYING TO OPEN ./outputs/p3disof//FG000000/
Process 0 creates the directory ./outputs/p3disof//FG000000/
TRYING TO OPEN ./outputs/p3disof//FG000000/
TRYING TO OPEN ./outputs/p3disof//FG000000/
.......
.......
.......
```

I ran the simulation with eight processors, this can vary depending on your computer or if you are using a sequential or GPU run. The `-m` flag is quite important to use, since this generate the data in the way that `fargo_tools` needs. Wait some time until the system evolves, let's stop the simulation after 10 outputs, you can do this typing `Ctrl + C`.

```
........
........
OUTPUTS 10 at date t = 31.415927 OK
......
Ctrl + C
```

A.2 Using t2csv

The `fargo_tools` directory contains a directory called `t2csv`, lets move there in

```
$ cd ~/fargo_tools/t2csv
```

Typing the `ls` command, you will see a `makefile` file, and if `OPENMP` is correctly installed, it should compile, then an executable file called `t2csv` and some `.o` files are created.

```
$ pwd
  /home/oscar/fargo_tools/t2csv
$ ls
  makefile
$ make
  mpif90 -cpp -finit-local-zero -fno-automatic -ffloat-store -c ../sources_f90/t2csv.f90
  mpif90 -cpp -finit-local-zero -fno-automatic -ffloat-store -c ../sources_f90/trans_coords.f90
  mpif90 -cpp -finit-local-zero -fno-automatic -ffloat-store -c ../sources_f90/create_names.f90
  mpif90 -cpp -finit-local-zero -fno-automatic -ffloat-store -c ../sources_f90/input_file.f90
  mpif90 -cpp -finit-local-zero -fno-automatic -ffloat-store t2csv.o trans_coords.o
  create_names.o input_file.o -o t2csv
$ ls
  create_names.o  input_file.o  makefile  t2csv  t2csv.o  trans_coords.o
```

If you obtain some like that in your computer, you are ready to use `t2csv`. The next step is to copy the `t2csv` file to your output file directory, this directory must be inside `fargo3d-1.0/outputs/p3disof/`, after copy it, lets move to this directory.

```
$ cp t2csv ~/fargo3d-1.0/outputs/p3disof/
$ cd ~/fargo3d-1.0/outputs/p3disof/
$ ls
  bigplanet0.dat   FG000008      gasenergy0.dat   gasvx2.dat   gasvy5.dat   gasvz8.dat
  density0_2d.dat  FG000009      gasenergy10.dat  gasvx3.dat   gasvy6.dat   gasvz9.dat
  dimensions.dat   FG000010      gasenergy1.dat   gasvx4.dat   gasvy7.dat   grid000.inf
  domain_x.dat     gasdens0.dat  gasenergy2.dat   gasvx5.dat   gasvy8.dat   grid001.inf
  domain_y.dat     gasdens10.dat gasenergy3.dat   gasvx6.dat   gasvy9.dat   IDL.var
  domain_z.dat     gasdens1.dat  gasenergy4.dat   gasvx7.dat   gasvz0.dat   orbit0.dat
  FG000000         gasdens2.dat  gasenergy5.dat   gasvx8.dat   gasvz10.dat  planet0.dat
  FG000001         gasdens3.dat  gasenergy6.dat   gasvx9.dat   gasvz1.dat   t2csv
  FG000002         gasdens4.dat  gasenergy7.dat   gasvy0.dat   gasvz2.dat   torq_planet_0.dat
  FG000003         gasdens5.dat  gasenergy8.dat   gasvy10.dat  gasvz3.dat   tqwk0.dat
  FG000004         gasdens6.dat  gasenergy9.dat   gasvy1.dat   gasvz4.dat   variables.par
  FG000005         gasdens7.dat  gasvx0.dat       gasvy2.dat   gasvz5.dat   vx0_2d.dat
  FG000006         gasdens8.dat  gasvx10.dat      gasvy3.dat   gasvz6.dat   vy0_2d.dat
  FG000007         gasdens9.dat  gasvx1.dat       gasvy4.dat   gasvz7.dat   vz0_2d.dat
```

Before running `t2csv`, an input file must be created, such a file must be named `input.dat`, and it has to have the next data in the following order

1. Number of cells in the X direction
2. Number of cells in the Y direction
3. Number of cells in the Z direction
4. Minimum output file to be transformed
5. Maximum output file to be transformed
6. Size of the jump between output files to be transformed
7. Min value in X direction
8. Max value in X direction
9. Min value in Y direction
10. Max value in Y direction
11. Min value in Z direction
12. Max value in Z direction
13. Grid Geometry. s for spherical, c for cylindrical, r for cartesian

If you are familiarized with the `FARGO3D` code, the values for 1-3, 7-12 can be obtained from the `.par` file, the value for 13 is obtained from the `.opt` file and the values for 4-6 are defined by the user. You can fill the `input.dat` file according to your simulation. For the `p3disof` setup, open a file called `input.dat` with your favorite text editor and fill it with the next lines

```
100      #Number of cells in the X direction
80       #Number of cells in the Y direction
40       #Number of cells in the Z direction
0        #minimum output file to be transformed
10       #maximum output file to be transformed
1        #size of the jump between output files to be transformed
-3.14159 #Min value in X direction
3.14159  #Max value in X direction
0.6      #Min value in Y direction
1.5      #Max value in Y direction
1.42     #Min value in Z direction
1.72     #Max value in Z direction
s        #s for spherical, c for cylindrical, r for cartesian
```

That means that our grid is 100, 80 and 40 cells in the X, Y, Z coordinate, respectively. We want to transform the output files from the 0 to the 10 output, one by one (it means that the output files from 0,1,2,3,4,5,6,7,8,9,10 will be used, if we would want 0,2,4,6,8,10 we had specified 2 in the size of the jump, and so on). The domain of our grid goes from $[-\pi : \pi]$ in azimuth, $[0.6 : 1.5]$ in radius and $[1.42 : 1.72]$ in colatitude. The grid geometry is spherical so the `s` character is used. Now the `input.dat` file must be in the same directory as `t2csv`

```
$ ls
  bigplanet0.dat   FG000009        gasenergy1.dat  gasvx5.dat   gasvy9.dat   input.dat
  density0_2d.dat  FG000010        gasenergy2.dat  gasvx6.dat   gasvz0.dat   orbit0.dat
  dimensions.dat   gasdens0.dat    gasenergy3.dat  gasvx7.dat   gasvz10.dat  planet0.dat
  domain_x.dat     gasdens10.dat   gasenergy4.dat  gasvx8.dat   gasvz1.dat   t2csv
  domain_y.dat     gasdens1.dat     gasenergy5.dat  gasvx9.dat   gasvz2.dat   torq_planet_0.dat
  domain_z.dat     gasdens2.dat     gasenergy6.dat  gasvy0.dat   gasvz3.dat   tqwk0.dat
  FG000000         gasdens3.dat     gasenergy7.dat  gasvy10.dat  gasvz4.dat   variables.par
  FG000001         gasdens4.dat     gasenergy8.dat  gasvy1.dat   gasvz5.dat   vx0_2d.dat
  FG000002         gasdens5.dat     gasenergy9.dat  gasvy2.dat   gasvz6.dat   vy0_2d.dat
  FG000003         gasdens6.dat     gasvx0.dat      gasvy3.dat   gasvz7.dat   vz0_2d.dat
  FG000004         gasdens7.dat     gasvx10.dat     gasvy4.dat   gasvz8.dat
  FG000005         gasdens8.dat     gasvx1.dat      gasvy5.dat   gasvz9.dat
  FG000006         gasdens9.dat     gasvx2.dat      gasvy6.dat   grid000.inf
  FG000007         gasenergy0.dat   gasvx3.dat      gasvy7.dat   grid001.inf
  FG000008         gasenergy10.dat  gasvx4.dat      gasvy8.dat   IDL.var
```

Now `t2csv` can be executed

```
$ mpirun -np 4 ./t2csv
  Creating disk          0 .csv file by processor           1
  Creating disk          1 .csv file by processor           2
  Creating disk          3 .csv file by processor           4
  Creating disk          2 .csv file by processor           3
  Creating disk          6 .csv file by processor           3
  Creating disk          5 .csv file by processor           2
  Creating disk          4 .csv file by processor           1
  Creating disk          7 .csv file by processor           4
  Creating disk         10 .csv file by processor           3
  Creating disk          9 .csv file by processor           2
  Creating disk          8 .csv file by processor           1
  The csv files have been created inside csv_files directory
```

If you obtain a similar output, it means a new directory called `csv_files` has been created. This new directory contains the disk$m$.csv files. You should obtain some like this

```
$ cd csv_files
$ ls
  disk0.csv   disk1.csv  disk3.csv  disk5.csv  disk7.csv  disk9.csv
  disk10.csv  disk2.csv  disk4.csv  disk6.csv  disk8.csv
$ head disk10.csv
  X, Y, Z, lden, vx, vy, vz, lenergy
  -0.59319105770894343      ,  -1.5740857362501043E-006 ,   9.0135281947011495E-002 ,
  -4.8342743671298392       ,   1.0866022838070270E-006 ,  -0.40948389480738567     ,
  0.0000000000000000        ,  -1.3010299956639813
  -0.59199667437804870      ,  -3.7624041166033385E-002 ,   9.0135281947011495E-002 ,
  -4.8342743696691226       ,   2.5972136210949341E-002 , -0.40865940464833644      ,
   0.0000000000000000       ,  -1.3010299956639813
                                                   .
                                                   .
                                                   .
```

The `X, Y, Z` columns correspond to the rectangular coordinates, `lden` and `lenergy` to the base 10 logarithm of density and energy, respectively and `vx, vy, vz` to the velocity components in rectangular coordinates. These files can be used to visualize the data.

A.3 Visualizing 3D data with Paraview

In general, csv files can be opened with several programs with different proposes, in this work it is shown how to visualize 3D-data with Paraview (http://www.paraview.org/). Once you have installed it, you can open it from the terminal (you must be in your `csv_files` directory)

```
$ pwd
  /home/oscar/fargo3d-1.0/outputs/p3disof/csv_files
$ paraview &
```

The visual version of Paraview will start. To understand better how to visualize the data, follow the next steps following figures A.1, A.2, A.3, A.4, A.5, A.6 and the numbers labeled there in.

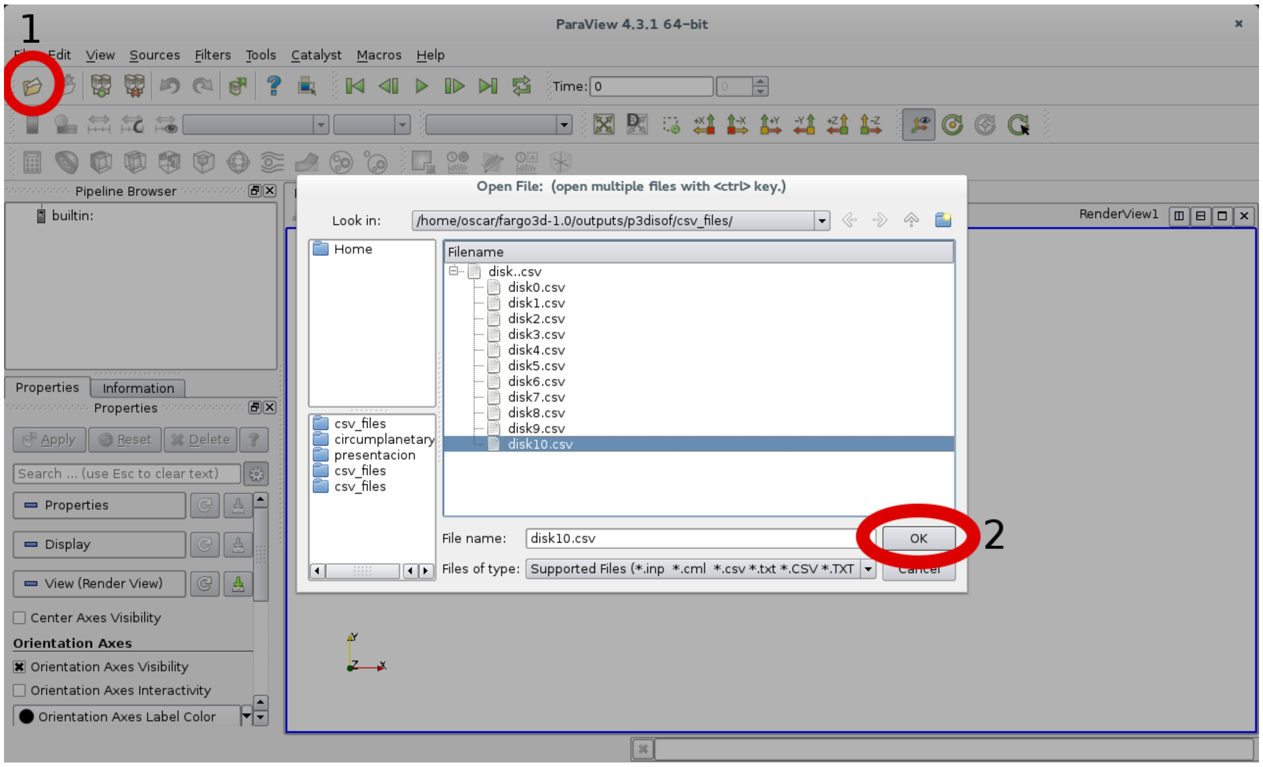
1. Open the file manager clicking on the button marked in figure A.1.
2. Select the file disk10.csv and click on the "ok" button.
3. When the file is opened, it appears in the pipeline browser, select it as is shown in figure A.2.
4. Click on the apply button, and the csv file will be displayed as a table (see figure A.2).
5. The next step converts the data into a grid form. Click on the Filters button and search for Table to Structured Grid.
6. Now it appears a new attribute in the pipeline browser called TableToStructuredGrid1, select it (see Fig. A.3).
7. You need to fill correctly the Whole Extent section and the X, Y, Z columns. The Whole extent must coincide with the original grid dimension. Remember the simulation is 100, 80, 40 cells in X, Y, Z. Then we fill the Whole Extent section with these values (see figure A.3). The X, Y, Z columns must coincide with the csv columns labeled as X, Y, Z (see figure A.3). When this is done, click on the apply button, now the data is displayed as seen in figure A.3.
8. Click on the + button to create another layout to visualize the data.
9. Click on the eye shape symbol (see figure A.4) and you will see the outline grid visualization as the one showed in figure A.4.
10. To visualize a field data, change outline to surface view (see A.5).
11. Select lden to visualize the density logarithm data, you should obtain something like figure A.5.
12. To finish, let's make a cut on the grid to visualize the density profile. Select the cut icon (see Fig. A.6).
13. Select a cut normal to Y.
14. Press apply, you should obtain something like in the figure A.6. This is the typical 3D-density profile of an isothermal circumstellar disk.

Now you can manipulate a circumstellar disk with your own hands! These are the basics for Paraview visualization. A lot of amazing things you can do using the csv files and this software. To learn more about Paraview capabilities, read the online documentation in `http://www.paraview.org/Wiki/The_ParaView_Tutorial`.

Another powerful characteristic of Paraview, is that allows to create python scripts to analyze multiple files. This is very useful in analyzing `FARGO3D` data. To learn more about to this check `http://www.paraview.org/Wiki/ParaView/Python_Scripting`.

A.4 Using `extracter`

....



Open File: (open multiple files with <ctrl> key.)

Look in: /home/oscar/fargo3d-1.0/outputs/p3disof/csv_files/

Home

csv_files
circumplanetary
presentacion
csv_files
csv_files

Filename
disk..csv
disk0.csv
disk1.csv
disk2.csv
disk3.csv
disk4.csv
disk5.csv
disk6.csv
disk7.csv
disk8.csv
disk9.csv
disk10.csv

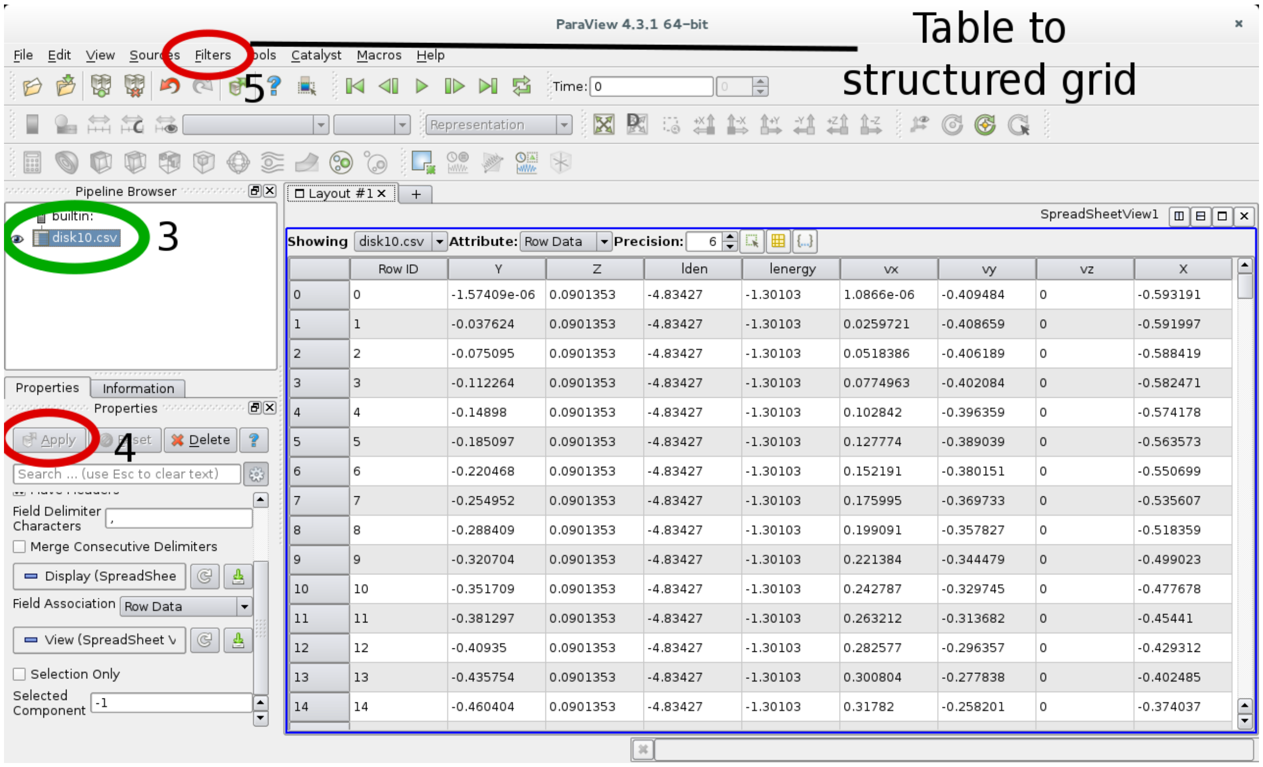
File name: disk10.csv

Files of type: Supported Files (*.inp *.cml *.csv *.txt *.CSV *.TXT

OK
Cancel

Center Axes Visibility
Orientation Axes Visibility [✓]
Orientation Axes Interactivity
Orientation Axes Label Color

Figure A.1: 3D visualization with Paraview. Steps 1-2.

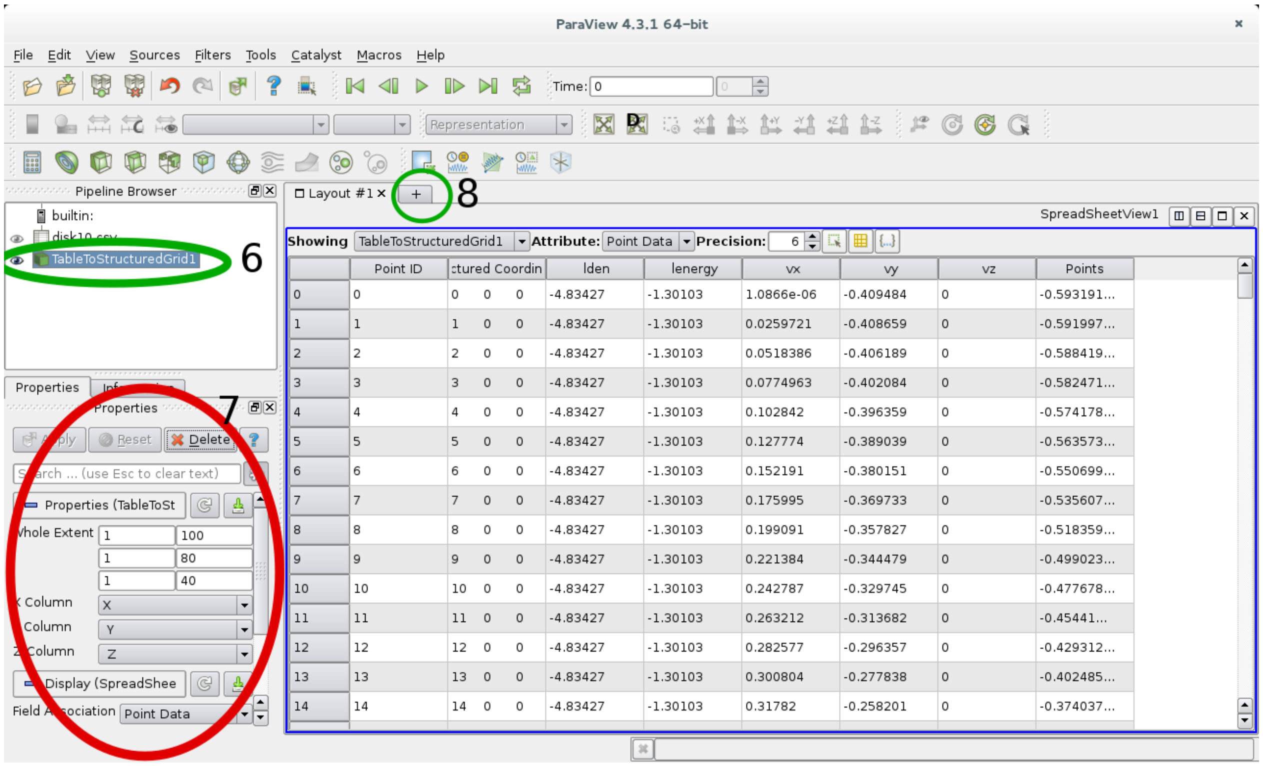


Field Delimiter Characters ,
Merge Consecutive Delimiters
Display (SpreadSheet)
Field Association Row Data
View (SpreadSheet V
Selection Only
Selected Component -1

Showing disk10.csv Attribute: Row Data Precision: 6

	Row ID	Y	Z	Iden	Ienergy	vx	vy	vz	X
0	0	-1.57409e-06	0.0901353	-4.83427	-1.30103	1.0866e-06	-0.409484	0	-0.593191
1	1	-0.037624	0.0901353	-4.83427	-1.30103	0.0259721	-0.408659	0	-0.591997
2	2	-0.075095	0.0901353	-4.83427	-1.30103	0.0518386	-0.406189	0	-0.588419
3	3	-0.112264	0.0901353	-4.83427	-1.30103	0.0774963	-0.402084	0	-0.582471
4	4	-0.14898	0.0901353	-4.83427	-1.30103	0.102842	-0.396359	0	-0.574178
5	5	-0.185097	0.0901353	-4.83427	-1.30103	0.127774	-0.389039	0	-0.563573
6	6	-0.220468	0.0901353	-4.83427	-1.30103	0.152191	-0.380151	0	-0.550699
7	7	-0.254952	0.0901353	-4.83427	-1.30103	0.175995	-0.369733	0	-0.535607
8	8	-0.288409	0.0901353	-4.83427	-1.30103	0.199091	-0.357827	0	-0.518359
9	9	-0.320704	0.0901353	-4.83427	-1.30103	0.221384	-0.344479	0	-0.499023
10	10	-0.351709	0.0901353	-4.83427	-1.30103	0.242787	-0.329745	0	-0.477678
11	11	-0.381297	0.0901353	-4.83427	-1.30103	0.263212	-0.313682	0	-0.45441
12	12	-0.40935	0.0901353	-4.83427	-1.30103	0.282577	-0.296357	0	-0.429312
13	13	-0.435754	0.0901353	-4.83427	-1.30103	0.300804	-0.277838	0	-0.402485
14	14	-0.460404	0.0901353	-4.83427	-1.30103	0.31782	-0.258201	0	-0.374037

Figure A.2: 3D visualization with Paraview. Steps 3-5.

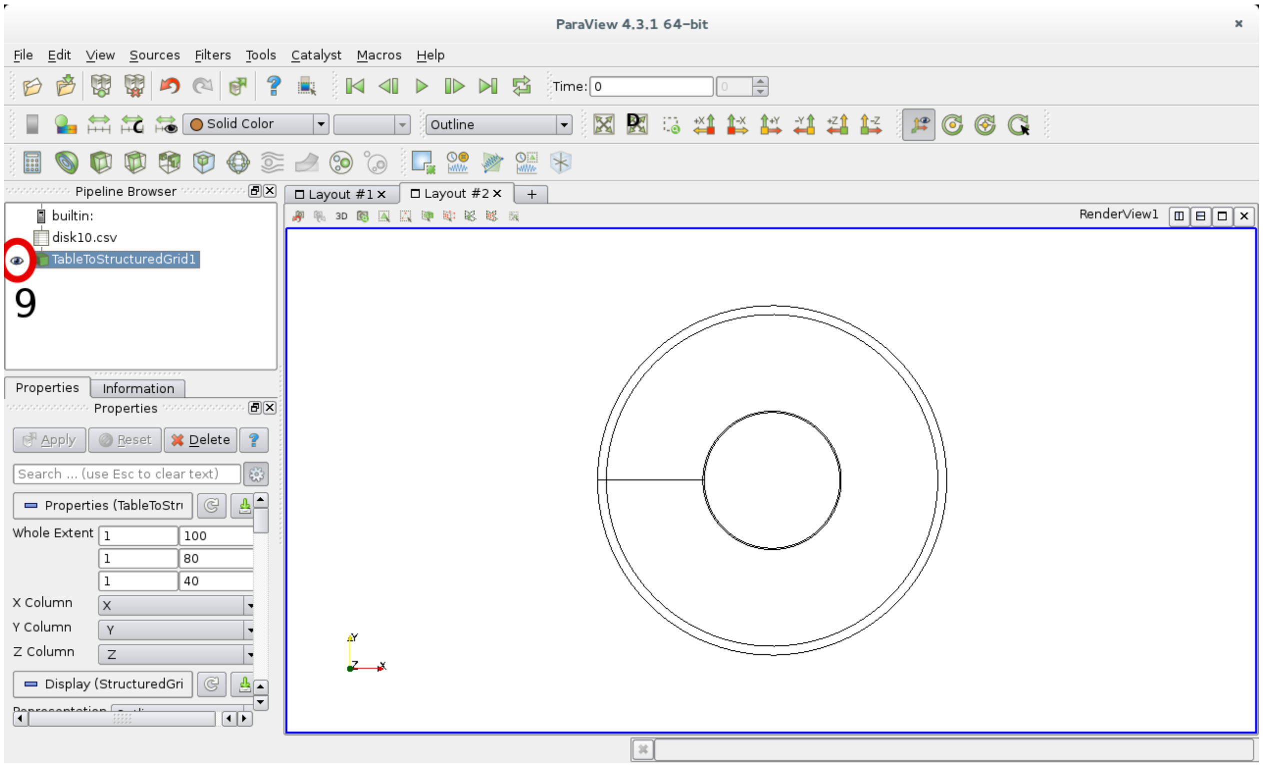


Whole Extent 1 100
1 80
1 40
X Column X
Y Column Y
Z Column Z
Field Association Point Data

Showing TableToStructuredGrid1 **Attribute:** Point Data **Precision:** 6

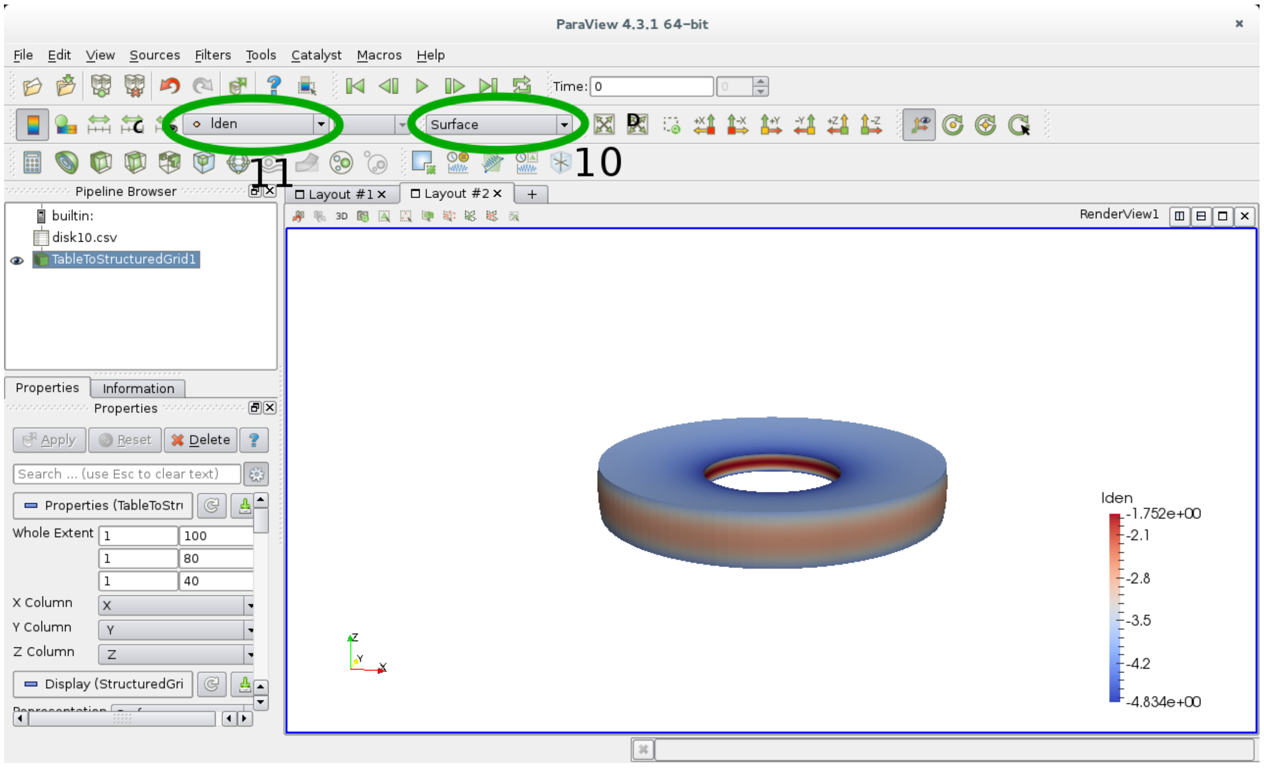
	Point ID	ctured Coordin	Iden	Ienergy	vx	vy	vz	Points
0	0	0 0 0	-4.83427	-1.30103	1.0866e-06	-0.409484	0	-0.593191...
1	1	1 0 0	-4.83427	-1.30103	0.0259721	-0.408659	0	-0.591997...
2	2	2 0 0	-4.83427	-1.30103	0.0518386	-0.406189	0	-0.588419...
3	3	3 0 0	-4.83427	-1.30103	0.0774963	-0.402084	0	-0.582471...
4	4	4 0 0	-4.83427	-1.30103	0.102842	-0.396359	0	-0.574178...
5	5	5 0 0	-4.83427	-1.30103	0.127774	-0.389039	0	-0.563573...
6	6	6 0 0	-4.83427	-1.30103	0.152191	-0.380151	0	-0.550699...
7	7	7 0 0	-4.83427	-1.30103	0.175995	-0.369733	0	-0.535607...
8	8	8 0 0	-4.83427	-1.30103	0.199091	-0.357827	0	-0.518359...
9	9	9 0 0	-4.83427	-1.30103	0.221384	-0.344479	0	-0.499023...
10	10	10 0 0	-4.83427	-1.30103	0.242787	-0.329745	0	-0.477678...
11	11	11 0 0	-4.83427	-1.30103	0.263212	-0.313682	0	-0.45441...
12	12	12 0 0	-4.83427	-1.30103	0.282577	-0.296357	0	-0.429312...
13	13	13 0 0	-4.83427	-1.30103	0.300804	-0.277838	0	-0.402485...
14	14	14 0 0	-4.83427	-1.30103	0.31782	-0.258201	0	-0.374037...

Figure A.3: 3D visualization with Paraview. Steps 6-8.



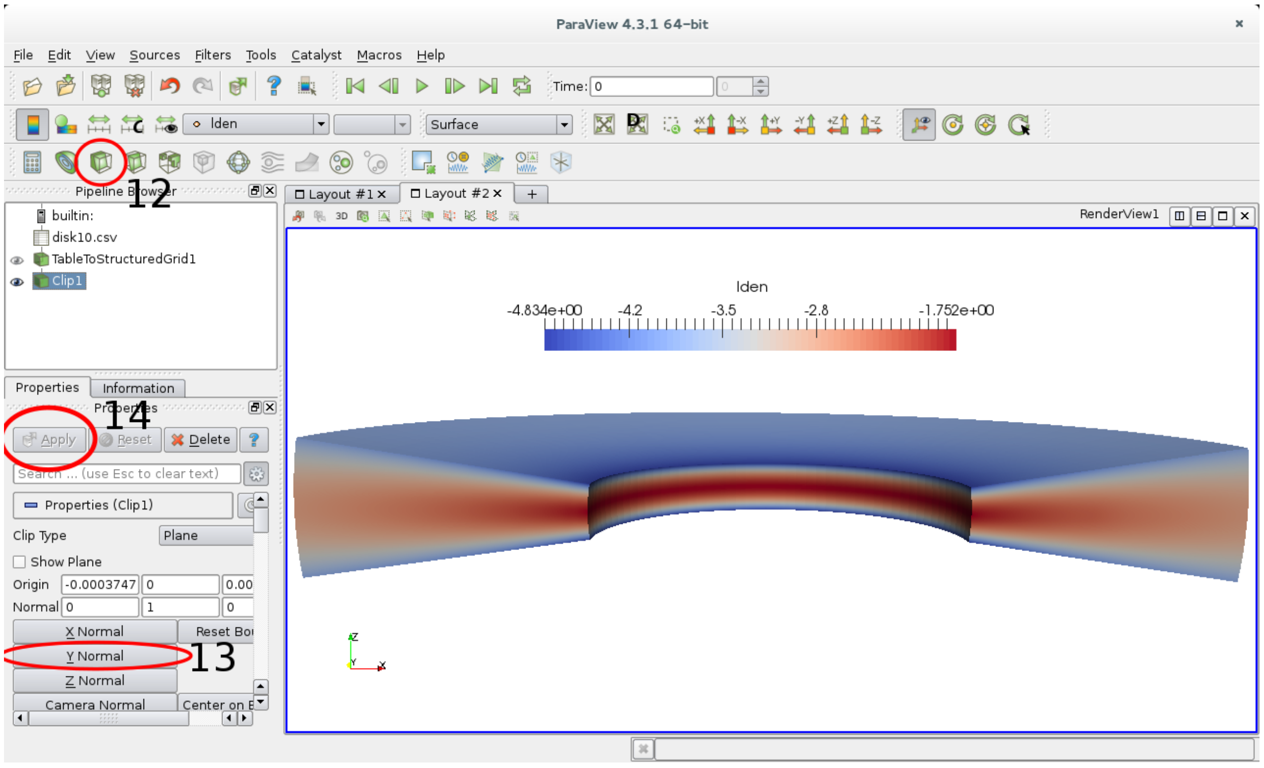
Whole Extent 1 100
1 80
1 40
X Column X
Y Column Y
Z Column Z

Figure A.4: 3D visualization with Paraview. Step 9.



Whole Extent 1 100
1 80
1 40
X Column X
Y Column Y
Z Column Z

Figure A.5: 3D visualization with Paraview. Steps 10-11.



Clip Type Plane
Show Plane
Origin -0.0003747 0 0.00
Normal 0 1 0

Figure A.6: 3D visualization with Paraview. Steps 12-14.