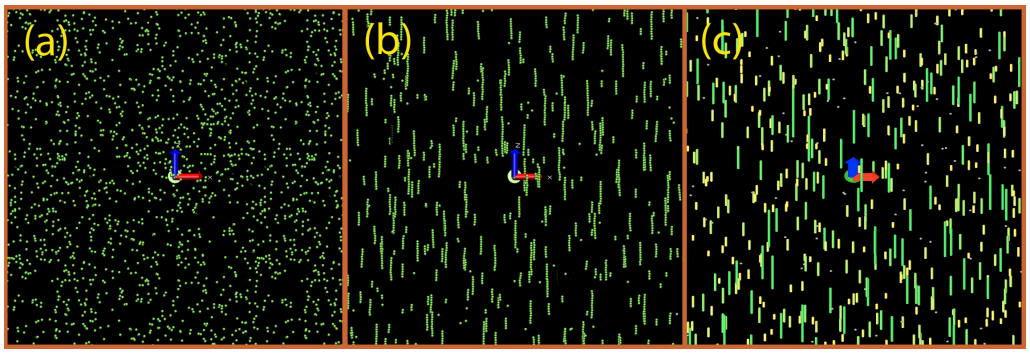
Institut de Ciència de Materials de Barcelona (ICMAB-CSIC)
Universitat Autònoma de Barcelona
Soft Matter Theory Group
Statistical Physics Group

MagChain tutorial



MagChain Developers: J.S. Andreu, C. Calero, A. Giménez-Romero, J. Camacho, J. Faraudo
Tutorial Contributors: A. Giménez-Romero

June 25, 2019

Contents

Symbols

- Γ: Magnetic coupling parameter.
- Φ_0: Volume fraction of the suspension occupied by the colloids.
- ρ_p: Colloid density.
- d: Colloid diameter.
- T: Temperature.
- η: Viscosity.
- D_1: Colloid diffusion coefficient.
- N_p: Number of colloids.
- L_x: Lenght of the simulation box in the x direction.
- L_y: Lenght of the simulation box in the y direction.
- L_z: Lenght of the simulation box in the z direction.
- Δt: Time step.
- t_f: Total time / final time.

1 Download

The MagChain program is written in C++ and it runs in any single-processor machine. The code has been successfully compiled in different platforms using the standard gnu C++ compiler. No additional libraries are required for its compilation:

- Mac OSX Leopard 10.5.8 (32bit) with g++ 4.0.1.
- Mac OSX Snow Leopard 10.6.8 (64bit) with g++ 4.2.1.
- LINUX openSUSE 11.3 2.6.34.8-0.2-desktop x86 64 with g++ 4.5.0.

In order to download the whole MagChain package you can use both of the below links:

- *http://www.icmab.es/softmattertheory/ .*
- *https://github.com/magchainsimulationpackage*

Once the zip file is download, extract the files. A folder will be created containing two sub-folders:

- **MagChain Simulation Software:** Contains the following folders
  - **Docs:** A short documentation.
  - **Examples:** Some simple examples for the different simulating options.
  - **Source:** The source code of MagChain Simulation Software
  - **Tools:** Complementary software to use with MagChain.
- **MagChain Visualisation Software:** Contains the following folders
  - **MagChain File Converter:** Converts the output non-standard xyz trajectory output of MagChain into a standard xyz.
  - **MagChain GL Viewer:** Trajectory visualiser in C++.
  - **MagChain Py Viewer:** Trajectory visualiser in VPython.

2 Introduction

This tutorial is a basic example of how to use MagChain software package. It contains 4 sections: Generating input files, Running the simulation, Visualising the simulation and Analysing the simulation results. The system that will be simulated in the example is the same that was reported as "Case 1" in [And+12].

The system properties and the simulation parameters are resumed in the following tables

Table 1: Characteristics of the colloidal dispersion of superparamagnetic particles simulated with CG

Γ	ϕ_0	ρ_p [g/cm^3]	d [nm]	T [K]	η $[Pa \cdot s]$	D_1 [m^2/s]
40	5.23×10^{-4}	1.0	100	300	1.0×10^{-3}	4.39×10^{-12}

Table 2: Required parameters used in the example simulation

N_p	L_x	L_y	L_z	Δt (s)	t_f (s)
8000	128	128	512	2.28×10^{-4}	5

In order to simulate this system some changes need to be made into the time variables, as MagChain simulation software uses adimensional units rather than real units. As reported in the software documentation, the change between real time and simulation time units is given by the following expression

$$t_{real} = t_{sim} \cdot \tau \tag{1}$$

where τ is the characteristic time scale of the system given by

$$\tau = \frac{d^2}{D_1} \qquad \text{with} \qquad D_1 = \frac{k_B T}{3\pi\eta d} \tag{2}$$

where D_1 is the diffusion coefficient for a single particle.

Thus, for our system, the characteristic time scale can be found by substituting the values in Section 2 in Eq. (2). Then we just have to divide the real time with the time scale to get the simulation time.

$$\tau = 0.002278 \qquad t_{sim} = 2195 \qquad \Delta t_{sim} = 0.1$$

As we will see below, MagChain takes as input the number of steps desired to simulate, so our input will be given by

$$N_{steps} = t_{sim}/\Delta tsim = 21950$$

3 Generating input files

The input files necessary to run MagChain simulations are simply plain text files which contain, at least, the required parameters to be set by the user. This input files can be written by hand by the user, using .dat extension, or using the MagChain Editor. Moreover, in this input files another input file is required, which gives the radius of the attraction zone for aggregates of different sizes. To generate this file a C++ tool has been implemented.

3.1 Attraction radius

First of all, we have to generate a file which contains the attraction radius [1] for different aggregate sizes. To do it, follow the following steps

1. Go to the *attraction_radius_sat* folder inside the the Tools folder.
2. Open the *attradius.cpp* with a text editor
3. Change the *maxsize* (maximum aggregate size) and g (Γ) values to 200 and 40 for our example
4. Open the terminal in the current directory and type make. Then press intro.
5. Now a C++ executable should appear in the folder. If you run it by double clicking the parameters used will be the ones setted in step 3. However, if you execute the binary from the terminal this parameters are asked as inputs.
6. In a few seconds some files should appear: *attradius.dat* is one of the input files required by MagChain simulation software and *log.txt* resumes the parameters used to generate the attradius file.

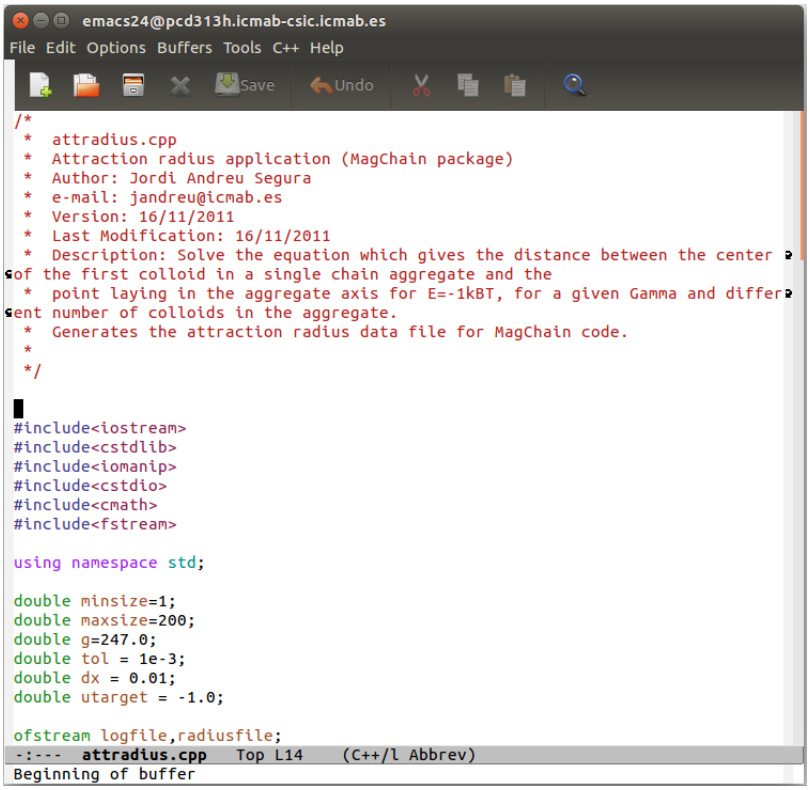


Figure 1: Snapshot of the attradius.cpp file

[1] Take a look on the physical model of the simulation method written in the documentation to understand what is the attraction radius

3.2 MagChain Editor

In order to automate the input file writing process a simple Qt graphical interface has been implemented. Required parameters are indicated with an asterisk sign (*) while optional parameters are simply left in blank. A part from the required parameters the attradius file must be loaded. The following snapshot shows the parameters given to MagChain Editor to run the example simulation, which are resumed in Table 2

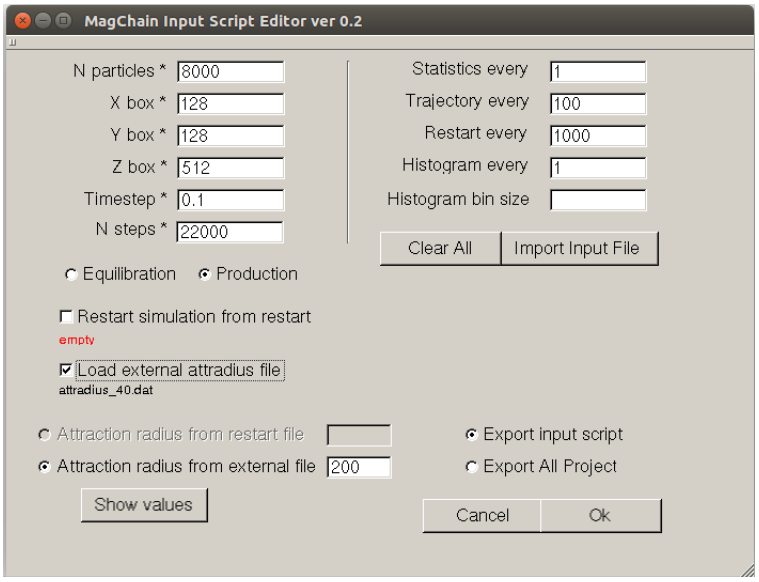


Figure 2: MagChain Editor snapshot

To use the editor follow this steps:

1. Go to the *MagChainEditor* folder inside the *Tools* folder. Then open the *Editor* folder.
2. To compile the program open a console and type the following commands

```
> qmake CONFIG+=debug CONFIG+=declarative_debug
-o Makefile ../src/MagChainEditor.pro

> make
```

3. Then just run the executable by double clicking or from the console.

Once the parameters are filled the only thing left to do is to generate the input script by pushing the ok button. If “export All project” is selected instead of “Export input script” the attraction radius file will also be exported. As can be seen in the snapshot, one can also choose between an equilibration phase or a production one, which would correspond to the simulation itself. An option to restart the simulation from a restart file is also available.

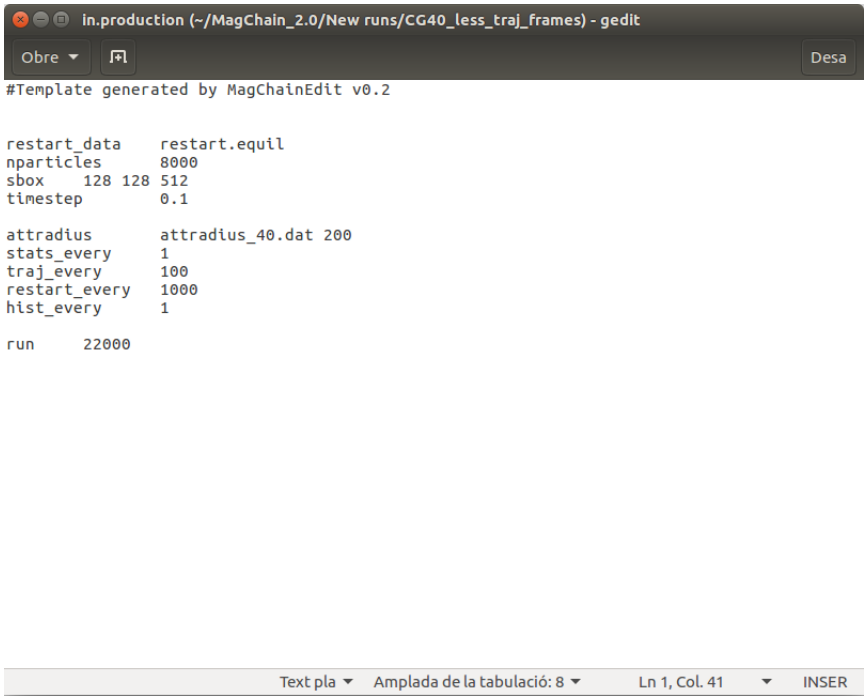


Figure 3: Input file generated by MagChain Editor

4 Running the simulation

Once the input files are done, it's time to run the simulation. First of all the MagChain executable need to be created. To do so follow the next steps:

1. Go to the source folder and open a console.
2. Type *make* and the MagChain executable will be created.

Then follow this instructions to run the simulation

1. Copy the attraction radius and the input file and the MagChain executable in the same folder.
2. Open a console in the current directory and run the MagChain executable specifying the input filename. As it is in the same folder just type *./magchain inputfilename*
3. Then wait until the simulation finish. For this example it will take about half an hour. As the simulations are usually long, you can run them in the background by typing *nohup ./magchain inputfilename &*

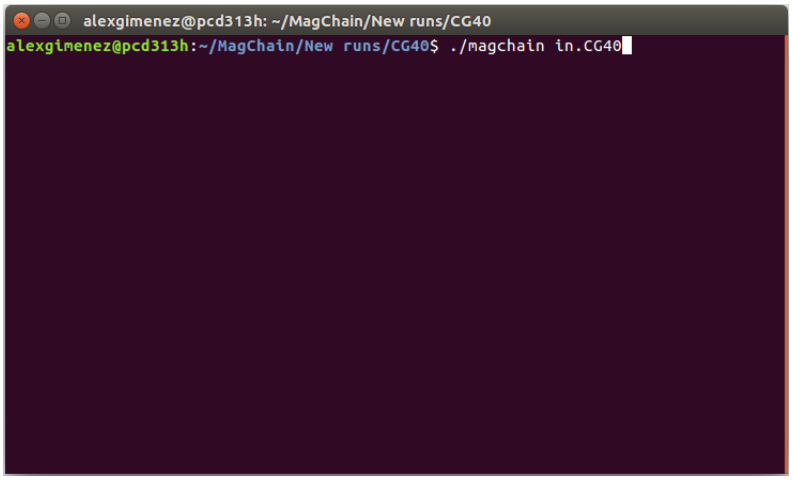


Figure 4: Running the simulation from the terminal

Moreover, a graphical interface in Python 3 has been implemented to run the simulations in background. To use it, follow the next steps.

Requirements

- Python 3 installed
- PyQt5 library: pip3 install PyQt5 or python3 -m pip install PyQt5

Steps

1. Go to the *MagChainSimulator* folder inside the *Tools* folder.
2. Load the required input files and the MagChain executable.
3. Click the run button.
4. Close the graphical interface, as the simulation will be running in the background.
5. To check it, open a console and type *top* and a magchain process should be shown.

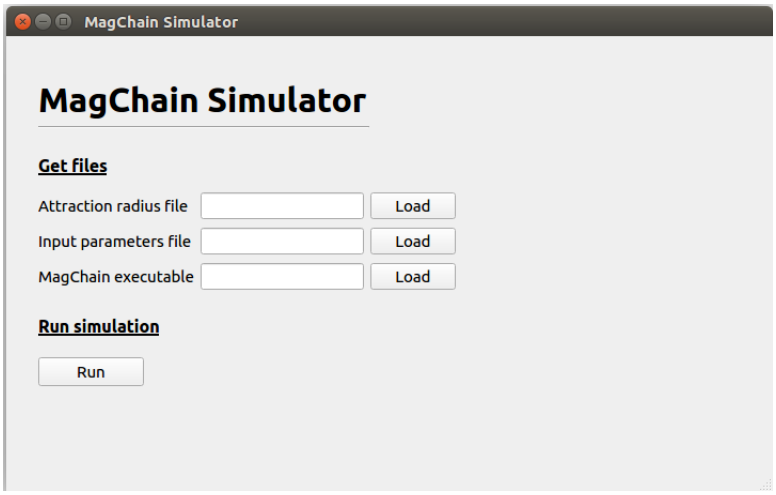


Figure 5: MagChainSimulator snapshot

5 Visualising the simulation

To visualise the simulation two tools are available: a GL Viewer, built in C++ and a VPy Viewer, built in VPython.

5.1 MagChain GL Viewer

To visualise the simulation in the GL Viewer follow the next steps:

1. Go to the MagChainGLViewer folder in the tools folder
2. Open a console in the current directory and type *make linux* or *make mac* depending on your operative system.[1]
3. Once the order is executed an executable should be created
4. Now in the command line type *./executablename trajectoryfilename*
5. For simplicity you can copy the trajectory file to visualise and the GLViewer executable in the same folder. If your trajectory filename is *trajectoryGC.xyz* and your executable name *mcviewer-linux* you just have to type *./mcviewer-linux trajectoryGC.xyz*

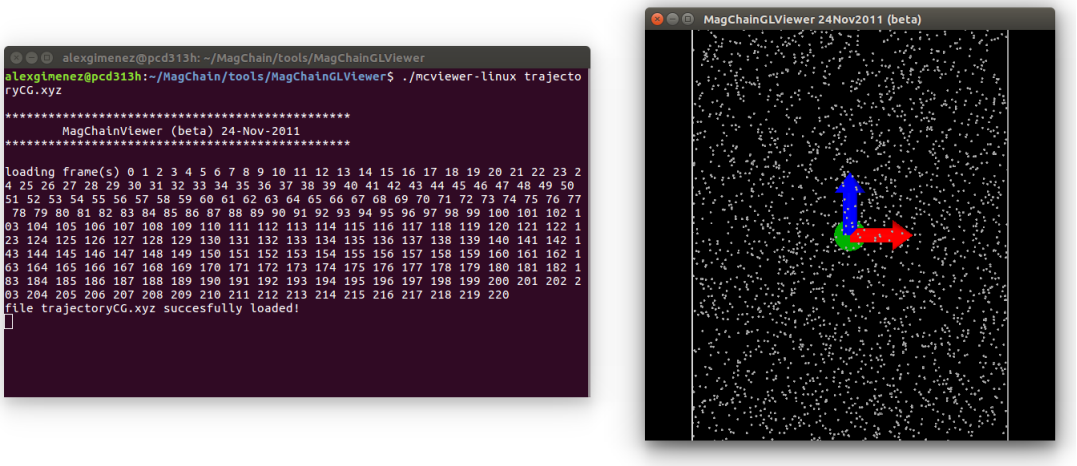


Figure 6: Opening MagChain GLViewer

By clicking the application window with the right-mouse button we get a list of all the available commands, which can be also found in the README.tools.txt in the tools folder. This commands are the following

- 'P': Set perspective.
- 'O': Set orthographic.
- 'T': Activate Translations.
- 'R': Activate Rotations.
- 'a': Place axis reference at the center of the box.
- 'A': Place axis reference at origin.
- 'x': Rotate/Translate clockwise respect to the axis.
- 'X': Rotate/Translate anti-clockwise respect to the axis.
- 'y': Rotate/Translate clockwise respect to the axis.
- 'Y': Rotate/Translate anti-clockwise respect to the axis.

[1] If you just type *make* it will raise an error. Then you will need yo execute *make clean* before compiling again with the right option.

- 'z': Rotate/Translate clockwise respect to the axis.
- 'Z': Rotate/Translate anti-clockwise respect to the axis.
- '+': Zoom IN.
- '-': Zoom OUT.
- 'p': Toggle Play/Pause.
- 'r': Resart movie.
- 'q': Restart view.
- 'Q': Quit the application.

5.2 MagChain VPy Viewer

To use MagChain VPy Viewer just copy the trajectory file in the same folder where is the VPyViewer.py executable. Then just open and execute the executable with VIDLE, which you must have installed.

5.3 MagChain File Converter

With this simple graphical interface built in Python3 you can convert the non-standard *trajectoryCG.xyz* provided by MagChain into a standard .xyz file format trajectory. This way, you can visualise the new standard xyz trajectory with standard visualisation softwares as VMD or Pymol. A .pdb file can be also generated after the xyz by selecting the *also pdb* option.

Requirements

- Python 3 installed. Need to be executed with Python3
- PyQt5 library: pip3 install PyQt5 or python3 -m pip install PyQt5
- MDAnalysis library: python3 -m pip install MDAnalysis

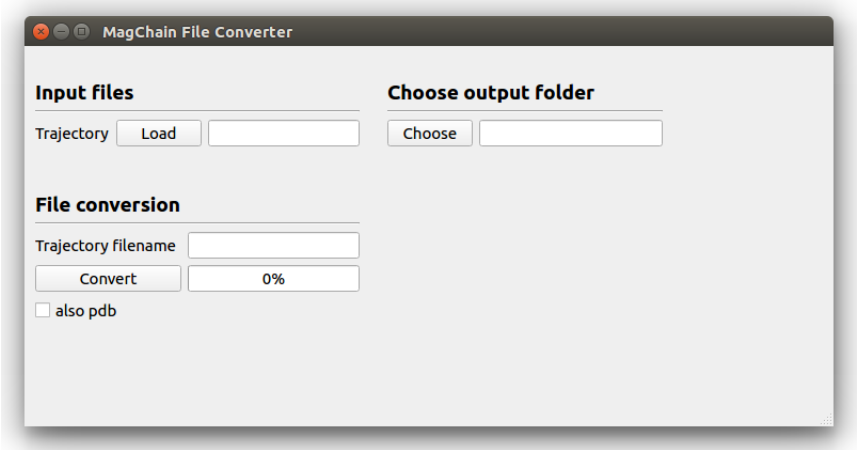


Figure 7: MagChain File Converter interface

6 Analysing the simulation results

To make an analysis of MagChain outputs a graphical interface has been developed, which can be found in the tools folder. See its documentation for installation prerequisites, installation guide and further information.

Requirements

- Python 3 installed. Need to be executed with Python3
- PyQt5 library: pip3 install PyQt5 or python3 -m pip install PyQt5
- NumPy and Matplotlib libraries, also installable with pip.

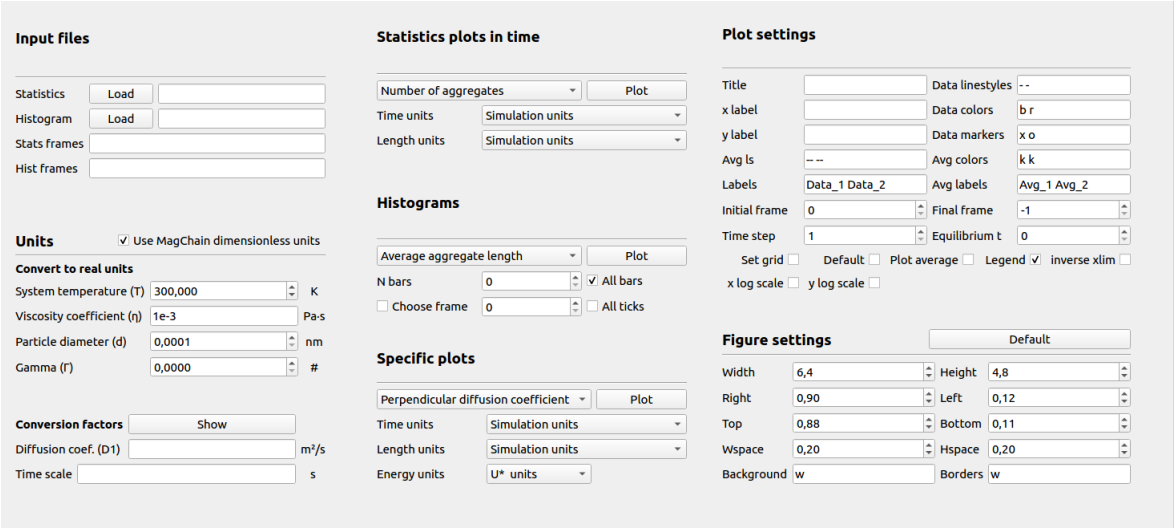


Figure 8: MagChain Analysis tool snapshot

6.1 Input data files

The data files analysed by MagChain Analysis are the outputs files provided by MagChain simulation software. So, in order to analyse this data we will have to load *stats.txt* and *histogram.txt* files. Once the files are loaded the number of frames for each data file will appear in the text lines below.

6.2 Unit conversion

We can choose whether we want to display the analysed results in dimensionless or in real units. If the box *Use MagChain dimensionless units* is checked all the results will be displayed in dimensionless units independently on the options selected in the specific plot sections. If we want to use real units, we have to fill the fields in *Convert to real units* section. Moreover, if we want to see the diffusion coefficient and time scale for the parameters used we can click the *Show* button.

6.3 Analysis and plot types

Several plots can be done from the *stats.txt* and *histogram.txt* output files provided by MagChain. The different options can be selected by clicking the dropdown menue in each section. To display the plot simply click the *Plot* button of the current section.

COMMENT: In the histogram section some display options have been implemented. Here yo can choose whether to plot all the bars provided by the *histogram.txt* file or select the numbers of bars to plot. Moreover, a frame selector is available (by default the last frame is displayed) and a

checkbox to plot all x-ticks of the histogram.

Now we are ready to plot some data, follow the next steps to generate some interesting plots

- First steps
  - Load the *stats.txt* and *histogram.txt* files
  - In order to display the plot in real units, fill the corresponding fields with the parameters in **??** and check that you get the same diffusion coefficient. Then deselect the *Use MagChain dimensionless units*
  - In the plot settings section change the first default marker "x" for "s", which stands for a squared marker. Then select a frame step of 1000 and select the *Set grid* box.
- Aggregates plot
  - Select "Number of aggregates" in the dropwdown menu, change time units to seconds and click the plot button. The following plot should be generated

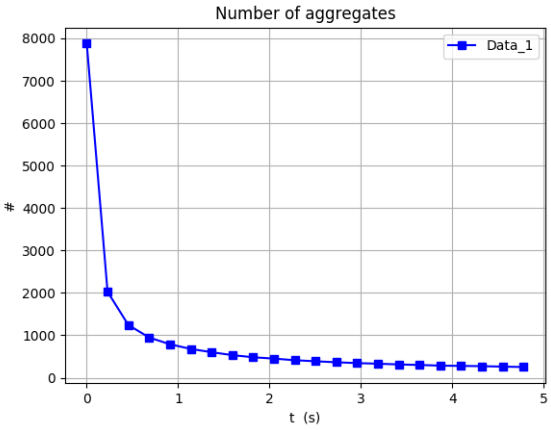


Figure 9: Number of aggregates in time

- Average length plot
  - Change the color from "b" to "r"
  - Select "Average length" in the dropwdown menu and click the plot button. The following plot should be generated

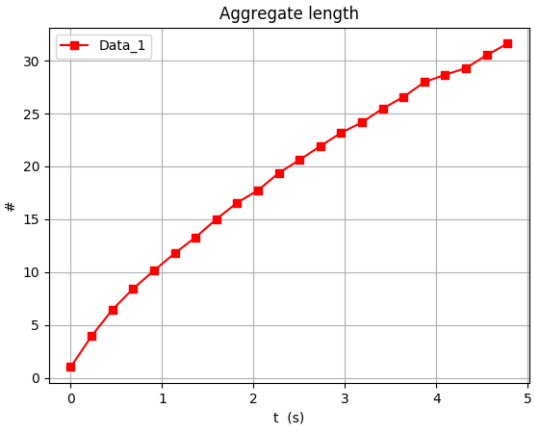


Figure 10: Aggregate average lenght in time

- In order to display a logaritmic scale select the *x log scale* and *y log scale* boxes. Then set the first marker to "None".
- Click the plot button, the following plot should be generated.

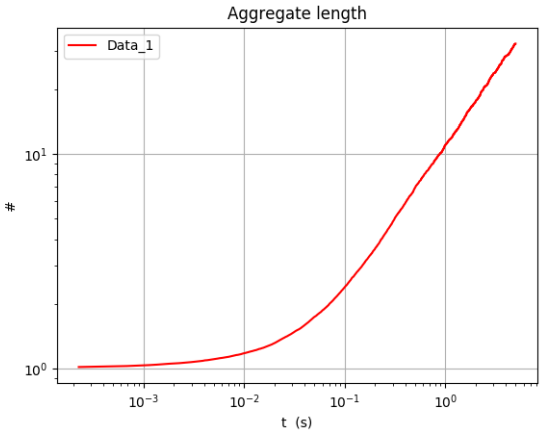


Figure 11: Aggregate average length logaritmic scale

- Histogram
  - Change the first color from "r" to "g"
  - Click the plot button in the histrogram section. A figure similar to the shown below should appear

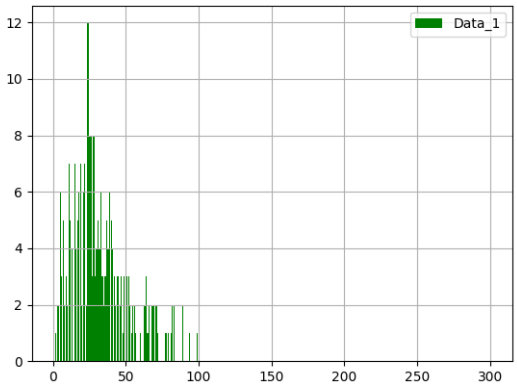


Figure 12: Histogram with all possible aggregate lengths

  - In order to display a more representative histogram set the *N bars* value to 100 and deselect the *All bars* checkbox.
  - Click the plot button again and compare ypur result with the following figure

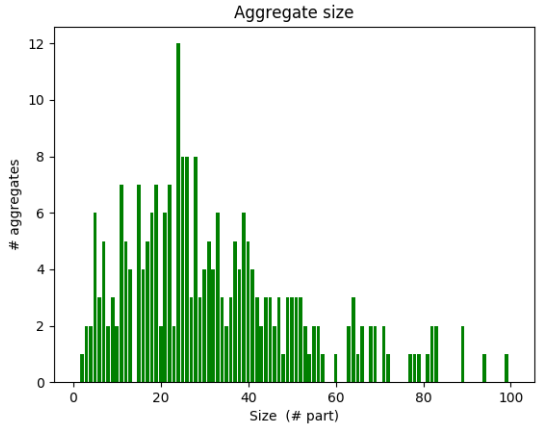


Figure 13: Histogram up to a length of 100 particles for aggregate

– Finally, select the *Choose frame* checkbox and plot the 5000$^{\text{th}}$ frame

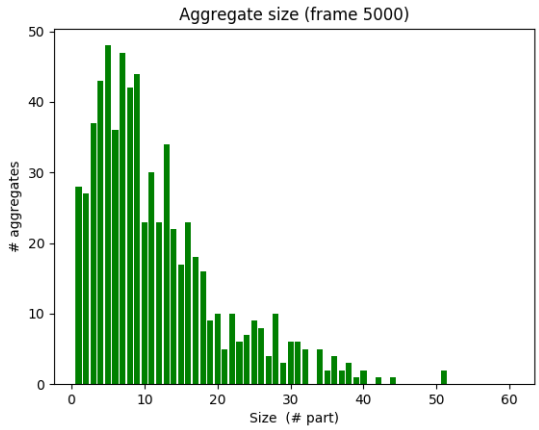


Figure 14: Histogram for the 5000$^{\text{th}}$ frame

- Energy per colloid
  - Change the first colour to "b" again and set the first marker to "x".
  - Select a frame step of 1000 and deselect the *Set grid* box.
  - Click the plot button and after some seconds of calculation the plot will be displayed. Compare it with the figure below

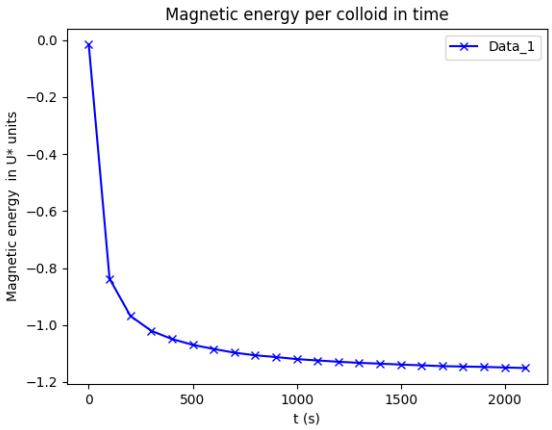


Figure 15: Energy per colloid

6.4 Customising plots

Almost all matplotlib.pyplot parameters can be changed in an easy way by the user. The available ones are the following:

IMPORTANT: In order to customise several lines, separate the parameters by a space.

- **Title:** Write here the plot title.
- **xlabel:** Write here the label for the x axis.
- **ylabel:** Write here the label for the y axis.
- **Data labels:** Write here the labels for the different lines that will be set in the legend.
- **Data linestyles:** Write here the linestyles for the different lines ploted following the matplotlib.pyplot syntax (*https://matplotlib.org/gallery/lines_bars_and_markers/line_styles_reference.html*).
- **Data colors:** Write here the colors for the different lines ploted following the matplotlib.pyplot syntax (*https://matplotlib.org/2.0.2/examples/color/named_colors.html*).
- **Data markers:** Write here the markers for the different lines ploted following the matplotlib.pyplot syntax (*https://matplotlib.org/api/markers_api.html*).

The same is applied to the average lines.

In the line below some checkbox are available:

- **Default:** Set default matplotlib.pyplots settings.
- **Grid:** Plot a background grid.
- **Plot average:** Plot an average computed from the equilibrium value.
- **Legend:** Show the written labels in a legend.
- **Inverse xlim:** Inverse the x axis limits (Use it to plot 'add dipole progression' graph)

Finally, you can also customise the ploted figure by changing it's shape, background colour or other parameters.

6.5 Output data files

The specific plots section makes a further non-trivial analysis into the *stats.txt* or *histogram.txt* files in order to calculate concrete magnitudes as the diffusion coefficient along the parallel or perpendicular axis or the energy per colloid of the system. To plot this new data, maybe the user is not satisfied with the options available in the graphical interface and wants to generate a more specific plot type. For this reason, the data obtained with the analysis mentioned is stored in some output files and saved into the *Outfiles* folder generated by the program.

References

[And+12] Jordi S. Andreu et al. "On-the-fly coarse-graining methodology for the simulation of chain formation of superparamagnetic colloids in strong magnetic fields". In: *Phys. Rev. E* 85 (3 Mar. 2012), p. 036709. DOI: *10.1103/PhysRevE.85.036709*. URL: *https://link.aps.org/doi/10.1103/PhysRevE.85.036709*.

[FAC13] Jordi Faraudo, Jordi S. Andreu, and Juan Camacho. "Understanding diluted dispersions of superparamagnetic particles under strong magnetic fields: a review of concepts, theory and simulations". In: *Soft Matter* 9 (29 2013), pp. 6654–6664. DOI: *10.1039/C3SM00132F*. URL: *http://dx.doi.org/10.1039/C3SM00132F*.

[Seg13] Jordi Andreu Segura. "Statistical Mechanics of Superparamagnetic Colloidal Dispersions Under Magnetic Fields". PhD thesis. Universitat Autònoma de Barcelona, 2013.

[Far+16] Jordi Faraudo et al. "Predicting the Self-Assembly of Superparamagnetic Colloids under Magnetic Fields". In: *Advanced Functional Materials* 26.22 (2016), pp. 3837–3858. DOI: *10.1002/adfm.201504839*. eprint: *https://onlinelibrary.wiley.com/doi/pdf/10.1002/adfm.201504839*. URL: *https://onlinelibrary.wiley.com/doi/abs/10.1002/adfm.201504839*.