Exercise 3: Galactic outflows

Oscar Agertz, Jon Ramsey

In this exercise you will be using the `RAMSES` code to model galactic outflows.

Prerequisites:

You will need the following tools:

- `git` for downloading of material and code
- `gfortran` or another Fortran compiler for compiling RAMSES
- The `RAMSES` code for running your simulations
- `python` and `YT` for data analysis

`git`, `gfortran`, `python and YT` should all be installed as part of your system installation your laptop. Follow the instructions on the homepage:

`https://indico.nbi.ku.dk/internalPage.py?pageId=9&confId=933`

if you didn't do it already. Obtain the exercise material:

- Download into PhD-School-2017-exercise-3 the exercise files, either the .zip or the .tar file, from the following dropbox link:

`https://www.dropbox.com/sh/jstfht3mj88ymct/AACDJOjMO4Q4MMd5zH8rkEvYa?dl=0`

Unpack them by typing either

```
> tar -xf exercise3.tar
```

or

```
> unzip  exercise3.zip
```

You now have a directory called `exercise3` with all relevant files in it.

Obtain the RAMSES code:

- It is publicly available and can be downloaded using git from bitbucket:
  `git clone https://bitbucket.org/rteyssie/ramses.git`

Obtain the YT plotting tools

- It is publicly available and can be downloaded here:
  `http://yt-project.org`

Ramses

By now you are all familiar with the RAMSES code. Once you have the exercise repository, `YT`, and `RAMSES` available locally, go to the folder where you have stored exercise 3. You should see the following subfolders:

```
exercise3  ramses
```

Inside the exercise folder, you have the files needed to carry out the exercise:

```
> ls exercise3
> Makefile blowout boom.nml  flux.py images.py
```

First, you have to compile `RAMSES` for hydrodynamical experiments in three dimensions. To accomplish this, copy the patch `'blowout'` to the patch directory, i.e. type

```
> cp -r blowout ramses/trunk/ramses/patch/
```

Then change to the `bin` subfolder of the `RAMSES` installation:

```
> cd ramses/trunk/ramses/bin
```

Then copy the `Makefile` from the exercise folder to the `bin` subfolder:

```
> cp ../../../../exercise3/Makefile .
```

If you use the `gfortran` compiler you are now ready to compile the code. However, if you use another compiler you will have to open the `Makefile` in an editor, comment out the lines setting the compiler, and comment in the other compiler (normally the Intel `ifort` compiler). Now you can compile the code:

```
> make
```

This will provide a lot of output about the compilation. If everything is successful it should end with a (long) linking line similar to:

```
gfortran -O3 -frecord-marker=4 -fbacktrace -ffree-line-length-none -
g amr_parameters.o amr_commons.o random.o pm_parameters.o
pm_commons.o poisson_parameters.o  poisson_commons.o
hydro_parameters.o hydro_commons.o cooling_module.o bisection.o
sparse_mat.o clfind_commons.o gadgetreadfile.o write_makefile.o
write_patch.o write_gitinfo.o read_params.o init_amr.o init_time.o
init_refine.o adaptive_loop.o amr_step.o update_time.o output_amr.o
flag_utils.o physical_boundaries.o virtual_boundaries.o
refine_utils.o nbors_utils.o hilbert.o load_balance.o title.o sort.o
cooling_fine.o units.o light_cone.o movie.o init_hydro.o
init_flow_fine.o write_screen.o output_hydro.o courant_fine.o
godunov_fine.o uplmde.o umuscl.o interpol_hydro.o godunov_utils.o
condinit.o hydro_flag.o hydro_boundary.o boundana.o
read_hydro_params.o synchro_hydro_fine.o init_part.o output_part.o
rho_fine.o synchro_fine.o move_fine.o newdt_fine.o particle_tree.o
add_list.o remove_list.o star_formation.o sink_particle.o feedback.o
clump_finder.o clump_merger.o flag_formation_sites.o init_sink.o
output_sink.o init_poisson.o phi_fine_cg.o interpol_phi.o
force_fine.o multigrid_coarse.o multigrid_fine_commons.o
multigrid_fine_fine.o multigrid_fine_coarse.o gravana.o
boundary_potential.o rho_ana.o output_poisson.o  ramses.o -o
ramses3d
rm write_makefile.f90
rm write_patch.f90
```

The RAMSES code executable is called "ramses3d". You can now copy this executable to the exercise folder and go there:

```
> cp ramses3d ../../../../exercise3/
> cd ../../../../exercise3/
```

This step by step procedure:

- go to the bin sub-folder of RAMSES;
- change or copy the Makefile;
- make a new executable with the command make;
- copy the executable to the folder where you will run RAMSES,

has to be followed every time you need to produce **a new executable.** Note that before compiling a new executable, it is good practice to first have to clean up from the last time by invoking "make clean" in the bin subfolder. In this exercise we will resume the same executable, so recompiling ought not to be necessary; you can reuse your executable for all substeps.

To run the code you need an input text file. An input file contains small blocks of code, called namelists, that specify parameters for the code.

For this exercise, we have prepared a ready made namelist called boom.nml. In subsequent sections we will go through the parameters that you will want to change for this exercise.

You are now ready to execute the code. To execute the code, type:

```
./ramses3d boom.nml
```

It should take roughly 15-25 minutes to run the code on a reasonably modern laptop. You will see *a lot* of output scrolling across the screen while the code runs. It basically gives the status of how many cells are in use at each AMR refinement level at a given timestep, what is the current time, how much memory is in use, what is the size of the timestep etc. Like in the previous exercises, in the simulation directory you will find an increasing number of directories starting with "output":

```
output_00001  output_00002  output_00003  output_00004  output_00005
 output_00006 .........
```

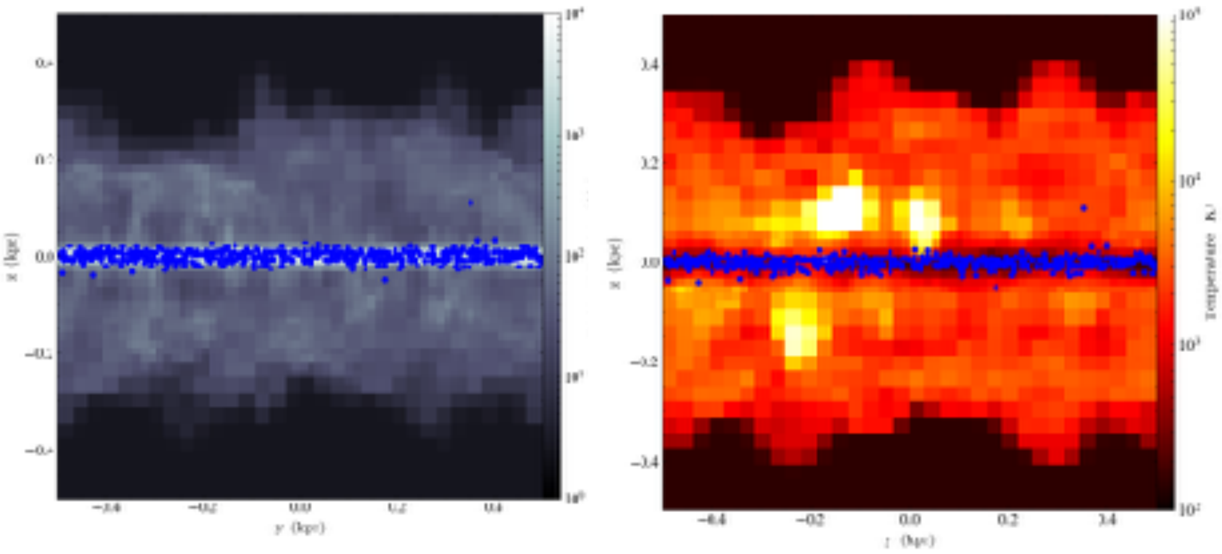

Fig 1. Gas density and temperature of the simulation in a 10^11 Msun halo. The points are star particles formed out of the gas.

Galactic outflows

As you have learned in the lectures on stellar feedback, the resulting outflow of gas from galaxies is of fundamental importance to galaxy formation. Numerical simulations without some form of stellar feedback fail to reproduce observations, as galaxies become too massive and compact. Outflows controls the baryon cycle of galaxies, and lead to high angular momentum systems.
A large number of sources are likely responsible for this behavior, e.g. supernovae, stellar winds and ionizing radiation. From an energetics point of view, the most likely dominant candidate for galactic outflows is supernovae. Individual explosions in homogenous media are well describe by the Sedov-Taylor solution you studied in Exercise 1. But what happens when multiple supernovae explode in a disc like geometry? This is what you will explore in this lab.

Using RAMSES, you will gain insight into the following questions:

1) When/how do supernovae drive outflows?
2) Do outflow properties depend on the host halo mass?
3) Do outflow properties depend on star formation parameters?
4) Is star formation regulated by feedback?

Note that most of these questions are unsolved problems, or at least debated vigorously in the astrophysics community.

The setup

The setup is very simple. You are carrying out a 3D simulation in a 1 kpc^3 box of a gas slab in an NFW dark matter halo potential. The halo's force acts only in the z-direction. The gas slab is representative of a patch of a disc galaxy, and stars form at high gas densities and inject energy from SNe, leading to galactic fountains and/or winds. Energy is injected *immediately* when stars form. In reality, a single stellar population will feature type II Sen for almost 40 Myr, the lifetime of 8 Msun stars. Because we directly model star formation, we can also study if feedback regulates this process.
You will run the simulation for at most ~100 Myr, which is sufficient to get an understanding of whether winds develop. Figure 1 shows the gas density and temperature in a test run, where the points symbolizes the star particles that have formed. The python scripts allows you to produce similar images.

The numerical resolution in the fiducial setup is quite low in order to be able to run several simulations. Note however that the spatial resolution reaches ~15 pc resolution or so which is higher than what all state-of-the-art galaxy formation models get (and on a laptop!). If you want to increase the simulation resolution (nlevelmax), feel free to do so in part 4 of this exercise, but be aware that it may severely extend the simulation run time.

You will quantify the mass outflow rate, the outflow velocity, the wind mass loading factor etc. This is all done with prepared scripts, meaning you should more care about varying the physics of the setup (halo mass, gas density, etc). Some simple additions to the setup is encouraged if you finish early.

In the RAMSES parameter file (`boom.nml`) you can specify the setup using the `gravity_params` array. The parameters refer to:

```
gravity_params=Sigma1,T1,Sigma2,T2,z_disc,M200
```

where

- `Sigma1`= initial surface density of the slab [Msun/pc^2]
- `T1`= initial gas temperature of the slab [Kelvin]
- `Sigma2`= initial surface density of the surrounding medium [Msun/pc^2]
- `T2`= initial gas temperature of the surrounding medium [Kelvin]
- `z_disc`= slab thickness [kpc]
- `M200`= dark matter halo virial mass [Msun]

By default it is set to:

```
gravity_params=1.e2,1e4,0.5,2e6,0.2,1.0d10
```

Note that the precise density of the coronal gas may matter in the end, but we will study systematic effects here, and will keep these fixed to limit parameter space. Brave student who finish early may experiment with changes to those variables.

In addition to these parameters, you may want to modify the maximum numerical resolution (**nlevelmax**) and the efficiency of star formation per free fall time (**eps_star**). We will discuss these further in the lecture.

Q1. Outflows properties vs halo mass

Assuming the same slab of gas exists in all halos. Change the halo mass (`M200`) to study how star formation and outflows proceed. We suggest that you try three different halo masses; M200=10^10, 10^11 and 10^12 Msun. The former is typical of dwarf galaxies, whereas the latter is roughly the halo massif the Milky Way

Create three directories:

```
      mkdir 1e10
      mkdir 1e11
      mkdir 1e12
```

Place a copy of `ramses3d` and `boom.nml` (use 'cp') in each directory, as well as the python scripts `flux.py` and `images.py`. Adjust the array `gravity_params` in each directory to the relevant halo mass, i.e :

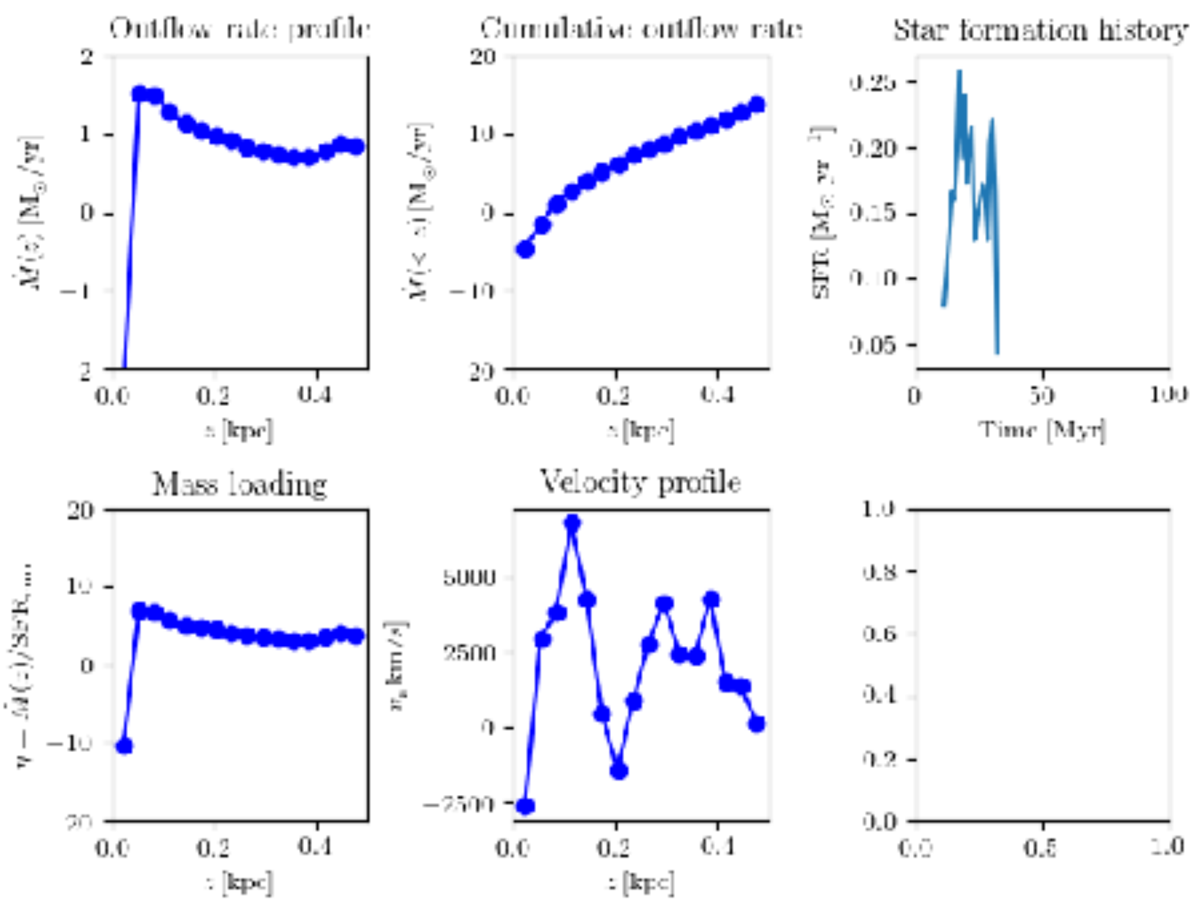

Fig 2. Gas density and temperature of the simulation in a 10^11 Msun halo. The points are star particles formed out of the gas.

1e10 Msun: `gravity_params=1.e2,1e4,0.5,2e6,0.2,1.0d10`
1e11 Msun: `gravity_params=1.e2,1e4,0.5,2e6,0.2,1.0d11`
1e12 Msun: `gravity_params=1.e2,1e4,0.5,2e6,0.2,1.0d12`

Run each model for the pre-set ~70 Myr. This is done with the command

```
     ./ramses3d boom.nml
```

If you are working in groups (highly encouraged), feel free to run one simulations per group member to speed up the run time! The screen output will look like this:

```
 _/_/_/       _/_/     _/    _/    _/_/_/   _/_/_/_/    _/_/_/
 _/    _/    _/  _/    _/_/_/_/   _/    _/  _/         _/    _/
 _/    _/   _/    _/   _/ _/ _/   _/        _/         _/
 _/_/_/     _/_/_/_/   _/    _/     _/_/    _/_/_/       _/_/
 _/    _/   _/    _/   _/    _/         _/  _/               _/
 _/    _/   _/    _/   _/    _/   _/    _/  _/         _/    _/
 _/    _/   _/    _/   _/    _/    _/_/_/   _/_/_/_/    _/_/_/
                        Version 3.0
        written by Romain Teyssier (University of Zurich)
                (c) CEA 1999-2007, UZH 2008-2014

 Working with nproc =    2 for ndim = 3
 Using solver = hydro with nvar =  5

 compile date = 08/29/17-20:41:23
 patch dir    = ../patch/blowout
 remote repo  = https://bitbucket.org/rteyssie/ramses
 local branch = master
 last commit  = 25a23a98dbeecc4abf7e7bcb63438fca2ac3896a

 Computing cooling model
 Building initial AMR grid
 Load balancing AMR grid...
 Load balancing AMR grid...
 Load balancing AMR grid...
 Load balancing AMR grid...
 Time elapsed since startup:   4.6200129985809326
 Initial mesh structure
 Level  1 has          1 grids (       0,       1,       0,)
 Level  2 has          8 grids (       4,       4,       4,)
 Level  3 has         64 grids (      32,      32,      32,)
 Level  4 has        512 grids (     256,     256,     256,)
 Level  5 has       4096 grids (    2048,    2048,    2048,)
 Level  6 has       8192 grids (    4096,    4096,    4096,)
 Starting time integration
 Load balancing AMR grid...
 Fine step=      0 t= 0.00000E+00 dt= 2.502E-05 a= 1.000E+00 mem=
0.8%  0.0%
 Fine step=      1 t= 2.50179E-05 dt= 2.502E-05 a= 1.000E+00 mem=
0.8%  0.0%
 Time elapsed since last coarse step:    0.83 s       11.22 mus/pt
11.22 mus/pt (av)
 Total running time:   5.44999981     s
```

You will quickly see output files being generated in the directory. These can readily be analyzed using the python scripts, but remember that you **must** have `YT` installed beforehand.

In each script you can set the range of outputs to be analysed with `ini` (first output) and `fin` (last output). The default range is `ini=1` to `fin=70`. Feel free to start analyzing the outputs immediately as they are being generated by `RAMSES`. Run the scripts by typing

```
     python < flux.py
```
and
```
     python < images.py
```

The `images.py` scripts will generate density (`rho.X.png`) and temperature (`T.X.png`) images for each snapshot, like the ones shown in figure 1. `flux.py` will generate generate a series of figures called “`results`”, as shown in figure 2, with four panels showing

(top-left): mass outflow rate vs height above the disc
(top-middle): Cumulative mass outflow rate (integral of mass outflow rate)
(top-right): Star formation history (so far)
(bottom-left): mass loading factor profile
(bottom-middle): average z-velocity vs height over disc plane.

The mass loading factor is of key importance for understanding the efficiency of wind driving. It is defined as the ratio between the mass flow rate and the rate of star formation, i.e. for every solar mass of stars being formed, how may solar masses are ejected. Outflows have `eta>0`, inflows `eta<0`. The star formation rate (SFR) is taken to be that of the previous “bin” in the SFR history, whereas the mass flow rate is the vertical flow profile. This gives us the mass loading vs vertical height above the disc, which makes it easy to discern whether winds are driven far from the disc plane.

Using your set of three simulations, and the provided scripts, discuss the following topics:

1) What do the outflows visually look like in the different halos? Scan through the images, or make a movie from the .png files, and qualitatively describe the evolution.
2) How far do winds reach? Do you simply get fountains in some cases?
3) What mass loading factors do you get? Do they change significantly overtime? Why or why not?
4) Are the star formation histories different in the different haloes? Why or why not?

Q2. Regulation of star formation

How important is supernova feedback for regulating the galactic star formation rate? We will investigate this by running the M200=10^10 Msun halo (the dwarf) again but without feedback. Create the following directory:

```
    mkdir 1e10noSF
```

Copy `ramses3d`, `boom.nml` and the python scripts into this directory. In `&PHYSICS_PARAMS` in the parameter file, change the fraction of stellar mass per star particle to explode in SNe to zero:

```
    eta_sn=0
```

As before, set `gravity_params` to:

```
gravity_params=1.e2,1e4,0.5,2e6,0.2,1.0d10
```

and run the simulation.

1) Does the SFR differ from before?
2) Discuss what physically may change the star formation rate when energy is injected immediately after a star particle is formed. Can feedback both increase and decrease the SFR?
3) Which purely numerical issues can you imagine when modeling how star formation is regulated by stellar feedback?

Q3. Energy or momentum driven winds?

In the exercise intro we learned that one can quantify galactic winds as energy or momentum driven. How do the winds from supernovae behave? To understand this you need to compute an average mass loading factor and compare to the circular velocity of the halo.

Run the script `flux.py` for a range of outputs, say `ini=30` to `fin=60`.

```
    python < flux.py
```

Don't carry out the analysis for early snapshots, as no wind (if any) has yet developed. At the end of the analysis you will get the mean mass loading factor and the standard deviation. It will look something like this:

```
Mean(mass loading) for snapshots 30 - 61 = 6.23754568647
Sigma(mass loading) for snapshots 30 - 61 = 2.13435832369
```

Having obtained the mean eta for each halo, convert your halo virial mass into a virial circular velocity. The halo virial velocity is roughly

```
v200= = 1.63e-2 * M200^(1/3) [km/s]
```

Remember that for:

Energy driven winds: eta ~ v200^(-2)
Momentum driven winds: eta ~ v200^(-1)

Do your results confirm any of these models? Why or why not?

Q4. (Time permitting) Sensitivity to numerical resolution

The spatial resolution is likely an important factor for resolving the full effect of stellar feedback. Increase nlevelmax in the parameter file and rerun one of the halos. You have full freedom to choose nlevelmax here, but note that your simulation quickly gets very expensive to run.

1) Do properties of the winds change visually?
2) How sensitive are your wind and star formation results? Is this good or bad news for the galaxy formation community that are struggling to even reach the resolution in your default settings?

Q5. (Time permitting) Sensitivity to star formation parameters

We will here study wether local star formation physics play a role. If some of your halos did not managed to drive winds, could the star formation efficiency per free-fall time or the threshold for star formation change this, i.e. is small scale physics to blame?

Pick one of the halos and try, for example, the following combinations:

```
     eps_star=1.0    !Vigorous local efficiency
     n_star=50       !Default
```

and

```
     eps_star=0.01   !Default low efficiency (1%)
     n_star=1            ! Stars form in diffuse gas -> feedback in
diffuse gas
```

1) What happens visually in these two cases?
2) Have the wind properties changed?
3) Can you physically motivate these two scenarios?