AGORA YT-Sunrise pipeline documentation

Objective and Initial Notes

The objective of the AGORA YT-Sunrise pipeline is to be able to analyse with Sunrise any galaxy simulation dataset that YT can read. With this goal in mind the pipeline has been built so that there is little or no effort to make it work with the output of any simulation code that YT supports, however, at the moment the pipeline has only been tested with ART-NMSU datasets, and the yt-3.x sunrise exporter has been written only for octree structures at the moment. Also there may be other things that will break when trying to run for other datasets that we won't know until someone tries.

Installation and Source

The pipeline is installed on the NERSC Edison system, under **/project/projectdirs/agora/.AGORA_PIPE_INSTALL**

The pipeline is composed of six executable scripts which can be found under 2 locations: **/project/projectdirs/agora/.AGORA_PIPE_INSTALL/scripts** and **/project/projectdirs/agora/scripts.** (the later being a symbolic link of the former)

The six executable scripts that compose the pipeline can also be found here https://bitbucket.org/mornkr/agora-analysis-script/src/c5b1838bf5b11365917809649dac76045f70fa6d/yt-sunrise_pipeline/?at=default, they are the runRockstar.py, findHostandMMPB.py, findGalaxyProps.py, genSunriseInput.py, setupSunriseRun.py and runSunrise.sh.

Getting some data

There are a few ways to get files from the archive at NERSC, you can read about them here **https://www.nersc.gov/users/data-and-file-systems/hpss/**

I usually do the following:

```
mkdir sim_name
cd sim_name
screen
```

hsi
cd /home/projects/mp363/ceverino/VELA_vXX/sim_name
get files_you_want (you can use wildcards)

The **get** command will put files in the directory from where you called **hsi**.

You need to get the 10MpcBox_csf512_* (hydro data), PMcrd* (particle data header) PMcrs* (particle pos and vel) and the stars_* (Stellar mass, initial mass, creation time and metallicity data) files.

Once files are copying you can do **control-a d** to detach the screen. Then you can log out and do **screen -r** to reattach the screen when you log back in.

You can also get the data of the VELA simulations on NERSC using this link http://portal.nersc.gov/archive/home/projects/mp363/www/

Setting up the Environment

Before any of the following steps you want to change your shell to bash, you can either type

$bash

everytime before using the pipeline, or even bether change your default shell to bash permanently here https://nim.nersc.gov/loginform.phtml (under Actions->Change Shell)

Then you can source the activate script that will setup your environment by typing

$source /project/projectdirs/agora/scripts/activate_yt-agora.sh

If the yt-agora environment was activated successfully you will see a **(yt-agora)** at the beginning of your command prompt.

Using the Pipeline Scripts

Once the yt-agora environment has been activated and you see **(yt-agora)** at the beginning of the command prompt you can use the scripts of the pipeline.

Every script except for runSunrise.sh (which is meant to be called within a PBS job script) can be called with the **-h** or **--help** arguments to display a help menu with how to use the script and all the possible options. For example

```
$python /project/projectdirs/agora/scripts/runRockstar.py -h
```

I also highly recommend looking at the code itself, which has been written hopefully clear enough for advanced users to understand everything the code does, and for beginner users to learn some python.

Example

Below is the PBS script that I have used to run Rockstar, find the Most Massive Progenitor branch and the galaxy properties of VELA28 across all scale factors (snapshots).

```
#PBS -S /bin/bash
#PBS -q regular
#PBS -N V28_rockstar_analysis
#PBS -l mppwidth=48
#PBS -l walltime=12:00:00
#PBS -e rockstar_analysis_V28.e
#PBS -o rockstar_analysis_V28.o

cd $PBS_O_WORKDIR
source /project/projectdirs/agora/scripts/activate_yt-agora.sh
module load mpi4py

aprun -n 24 -N 12 -S 6 python-mpi
/project/projectdirs/agora/scripts/runRockstar.py VE
LA28 -s 10MpcBox_csf512_a* --force_res=6e-5  --initial_metric_scaling 0.1
--multi_mass --particle_types specie0 specie1 specie5 > runRockstar_V28.log

python /project/projectdirs/agora/scripts/findHostandMMPB.py VELA28
 -v 1e11 > findHostandMMPB_V28.log

python /project/projectdirs/agora/scripts/findGalaxyProps.py VELA28 >
findGalaxyProps_V28.log
```

NOTE: For most of the VELA's simulations the specie5 particles are the stars, but for some, like VELA01-05, the star particles are labeled specie4. In that case you need to change the runRockstar command as follows

```
aprun -n 24 -N 12 -S 6 python-mpi
/project/projectdirs/agora/scripts/runRockstar.py VE
LA28 -s 10MpcBox_csf512_a* --force_res=6e-5  --initial_metric_scaling 0.1
--multi_mass --particle_types specie0 specie1 specie4 --recognized_star_types
[specie4] > runRockstar_V28.log
```

If the above job completes successfully the output will be written on the default output paths which in this case would be **VELA28/analysis/rockstar_output** and **VELA28/analysis/catalogs**.

The ***mmpb_props.npy** and ***galay_props.npy** under **VELA28/analysis/catalogs/** are numpy dictionaries containing the properties of the halo and the galaxy being analyzed across all the snapshots that were included in the Rockstar merger-trees. To load these dictionaries in python just do

```
module load numpy as np
catalog_dictionary = np.load('catalog_name.npy')[()]
```

Once the ***mmpb_props.npy** and ***galay_props.npy** catalogs are generated, one can proceed to generate the cameras to use in sunrise and export the data to a FITS files that sunrise can read. This is done by the genSunriseInput.py script, which also makes projection plots for some of the camera's field of views. Generating the cameras is fast, making the projection plots and exporting the data is very slow at the moment and needs to be optimized. Because exporting the data and making projection plots is so slow at the moment I recommend that you do those in separate PBS scripts. For example, to generate the cameras and just make plots (without exporting) for VELA28 I would use the following PBS script:

```
#PBS -S /bin/bash
#PBS -q regular
#PBS -N V28_genPlots
#PBS -l mppwidth=48
#PBS -l walltime=24:00:00
#PBS -e genPlots_V28.e
#PBS -o genPlots_V28.o
```

```
cd $PBS_O_WORKDIR
source /project/projectdirs/agora/scripts/activate_yt-agora.sh
module load mpi4py

aprun -n 48 python-mpi /project/projectdirs/agora/scripts/genSunriseInput.py
VELA28 --no_export > genSunriseInput_plot_only.log
```

NOTE: The code to make projection plots is written so that the halos/subhalos in the field of view are annotated with circles, however the `RockstarHaloList` module in yt is not working at the moment so for now the halos are missing in the plots.

And for just exporting I would use the following PBS script (the exporting parallelization is done by sending one snapshot to each processor, so use as many MPI tasks as snapshots you want to export)

```
#PBS -S /bin/bash
#PBS -q regular
#PBS -N V28_export
#PBS -l mppwidth=48
#PBS -l walltime=48:00:00
#PBS -e export_V28.e
#PBS -o export_V28.o

cd $PBS_O_WORKDIR
source /project/projectdirs/agora/scripts/activate_yt-agora.sh
module load mpi4py

aprun -n 48 python-mpi /project/projectdirs/agora/scripts/genSunriseInput.py
VELA28 --no_plots --max_level=10 > genSunriseInput_export_only.log
```

The genSunriseInput.py script will create the **/VELA28/analysis/sunrise_analysis/** directory with a sub-directory for each scale factor (snapshot). Plots will be placed under **VELA28/analysis/sunrise_analysis/scale_factor/yt_plots** and the camera files and the exported FITS file under **VELA28/analysis/sunrise_analysis/scale_factor**.

NOTE: At the moment the sunrise exporter in yt is very slow, it processes each snapshot in serial and it takes about 46 hrs to export one snapshot with --max_level=10. For the VELA simulations at z~1, the max refinement level is around 12, but exporting up to that level would take more than 48 hrs which is the wall time limit for jobs in Edison. Thus at the moment it is not possible to export snapshots to their max level of resolution on Edison (unless ask for a special job reservation with > 48hrs). There are ways to optimize the exporter to make it faster. A good place to start optimizing the exporter is to have yt find the children of each oct ahead of the exporting routine (maybe in parallel). As of now the children of each oct are found in serial while doing the export, slowing the exporting routine significantly.

Once you have the camera files and the exported FITS file under **sim_name/analysis/sunrise_analysis/scale_factor**. You can call the setupSunriseRun.py to setup the sunrise runs, this is a inexpensive step that can be done from the login nodes, i.e. no need to submit a job. Continuing with the example of VELA28, to setup a sunrise run tha skips the dust IR emission you would do

```
python /project/projectdirs/agora/scripts/setupSunriseRun.py VELA28
--parameter_set skipir
```

The setupSunriseRun.py script will create the **sim_name/analysis/sunrise_analysis/scale_factor/sunrise_runs/sunrise_runName** directory with all the input needed for sunrise (config files, camera files, filters, and FITS file). It will also copy the runSunrise.sh script there, and modify it accordingly to the options given to setupSunriseRun.py.

Once you have a runSunrise.sh ready to launch you can use a PBS script like the one below to run it

```
#PBS -S /bin/bash
#PBS -q regular
#PBS -N V28_runSunrise
#PBS -l mppwidth=24
#PBS -l walltime=12:00:00
#PBS -e runSunrise.e
#PBS -o runSunrise.o

cd $PBS_O_WORKDIR
source /project/projectdirs/agora/scripts/activate_yt-agora.sh

sh runSunrise.sh > runSunrise.log
```

NOTE: I was not able to compile Sunrise with MPI support on Edison this time, so you want to run the runSunrise.sh script on a single node (i.e. **#PBS mppwidth=24**). That should not be an issue since the exported FITS datasets are usually small enough to fit into the memory of a single node.