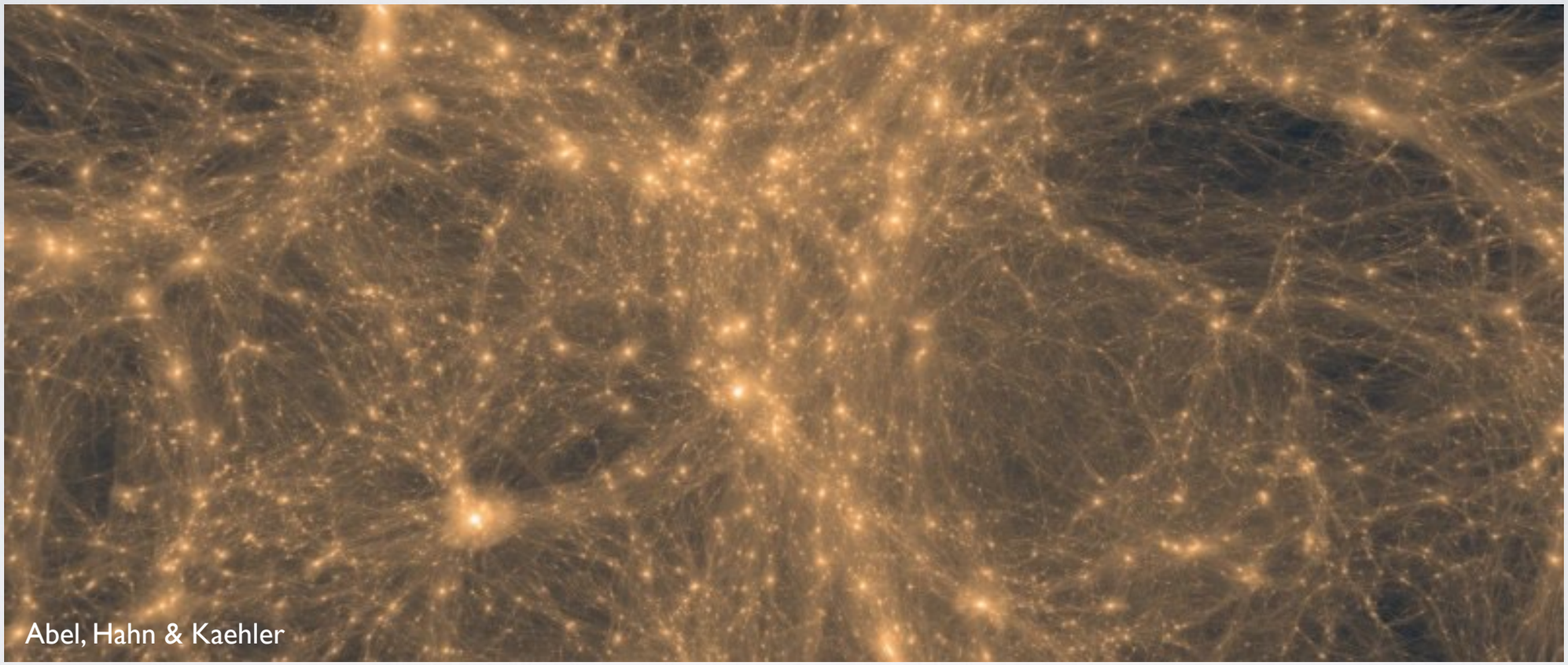
KIAS Astrophysics Summer School Lecture #2

Astrophysics Simulations: Practical Examples



Ji-hoon Kim (Seoul National University)

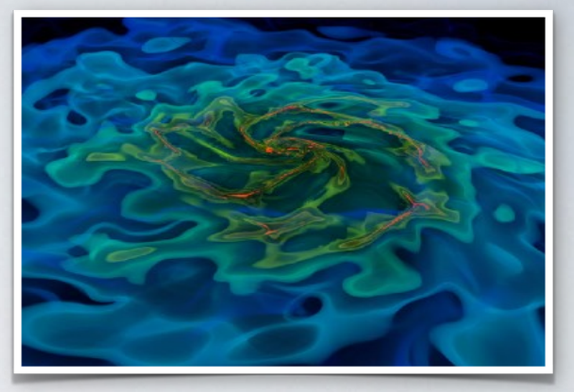
This material is for the educational purpose only, and is not for public consumption or mass distribution without the author's explicit consent.
Despite his best efforts, the author apologizes for any inaccurate or outdated information and references.

Today's Lecture: Outline

- Practical Examples of Computational Astrophysics

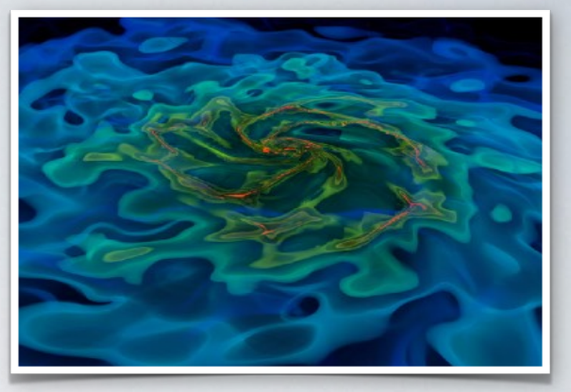
- How to Run Cosmological Simulations
  (+ How to Formulate Their Initial Conditions)

- How to Run Idealized, Isolated Simulations



Step-by-Step Instruction Slides:

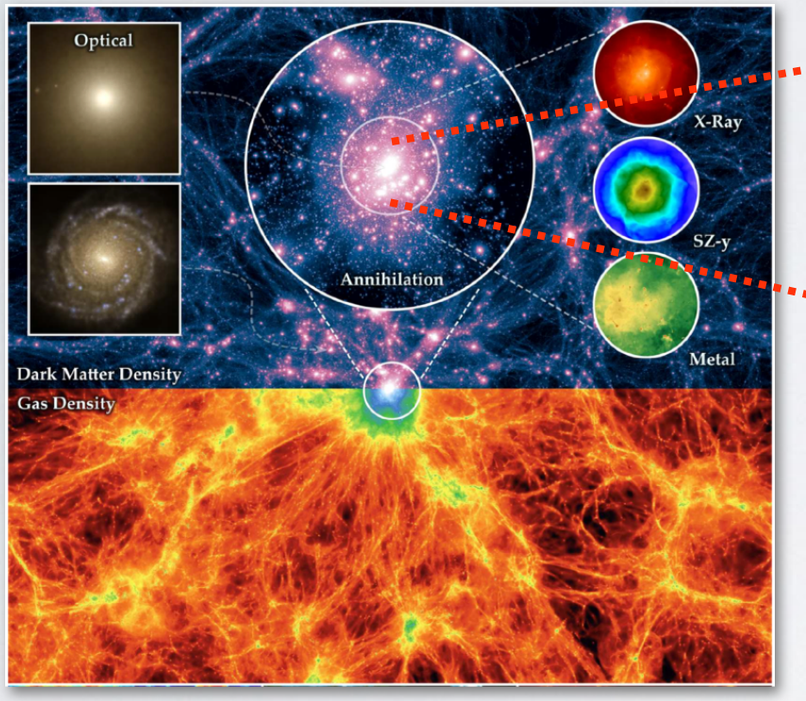
tinyurl.com/46xwk5n6



Cosmological Simulations and Numerical Resolution

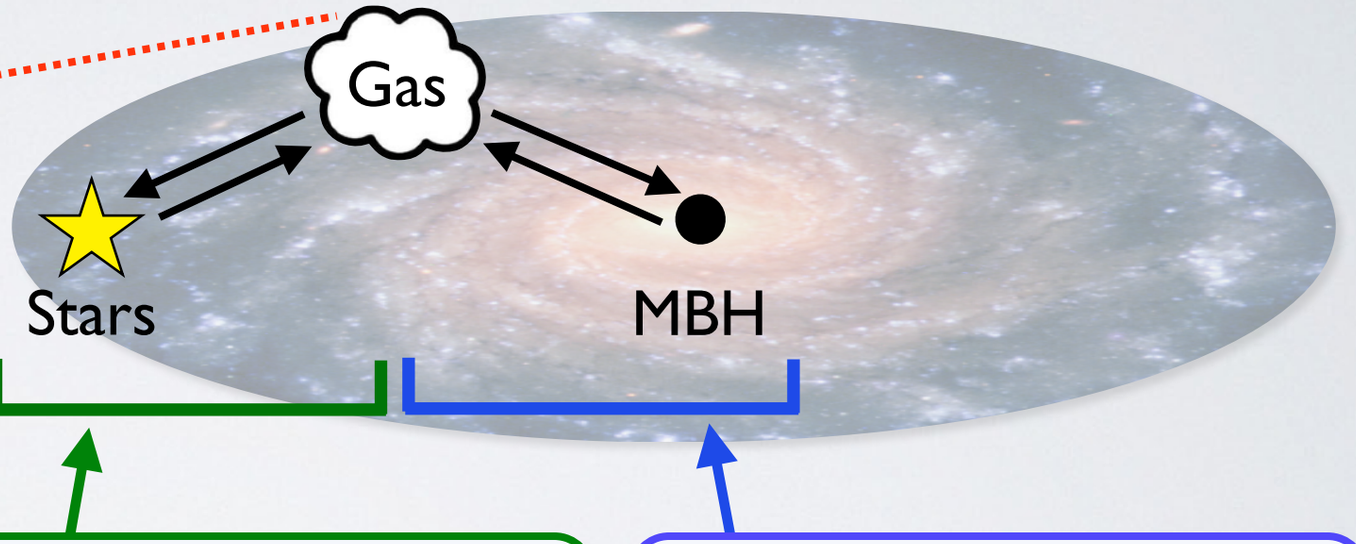
Will It Ever Be Enough?

- Computational tools are getting better; resolution is improving.
- However, they will never be enough for ambitious simulators.



Vogelsberger et al. 2014

2 x 1820^3 particles/cells in $(75\ h^{-1}\ \mathrm{Mpc})^3$



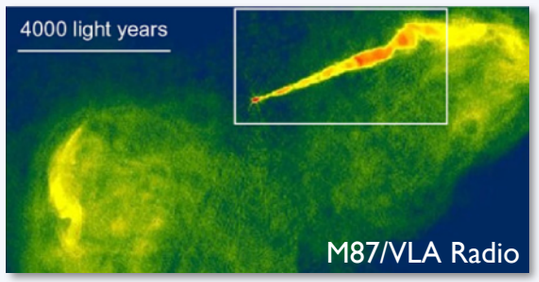
Star Cluster or GMC Formation & Feedback

$R_{GMC} \sim 10$ pc

Massive Black Hole (MBH) Accretion & Feedback

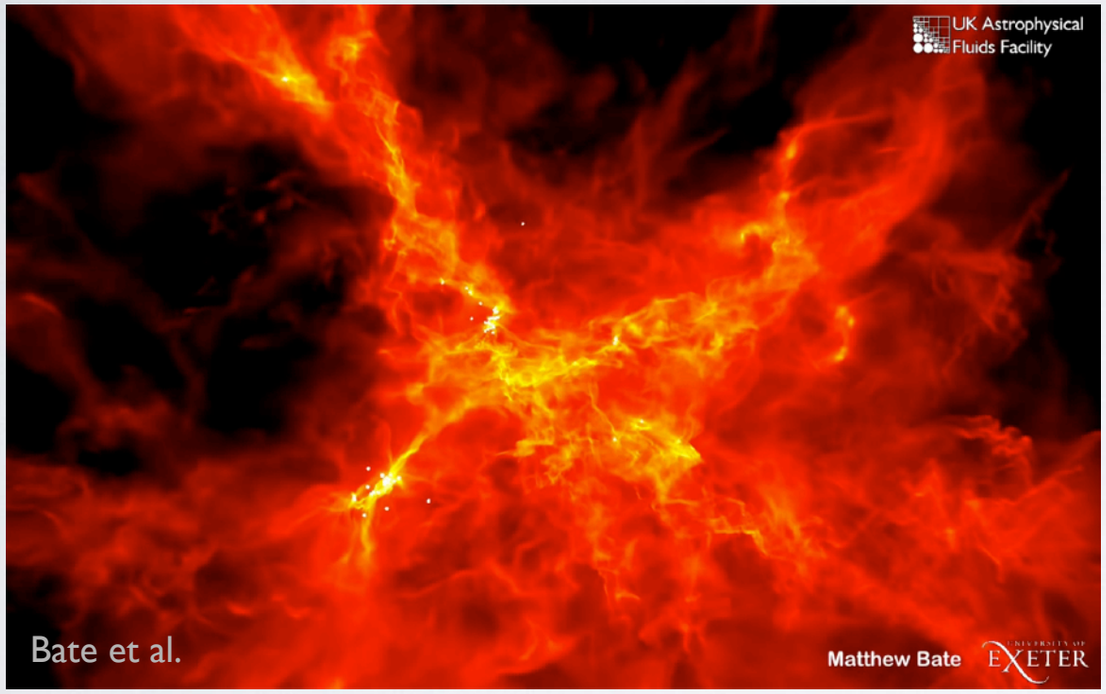
$R_{Bondi} \sim$ pc

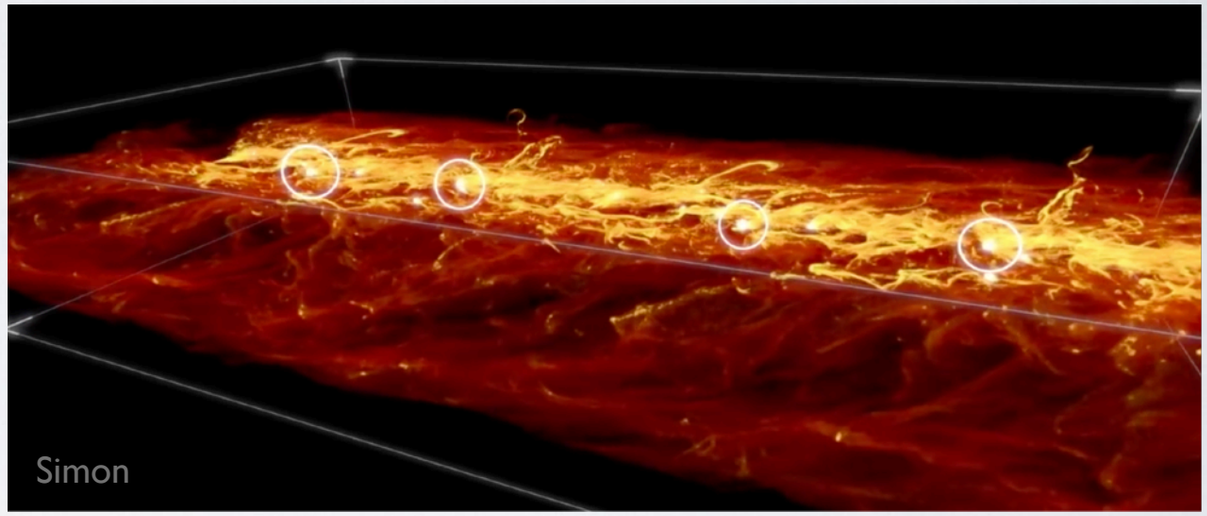
$R_{Schw.} \sim$ AU

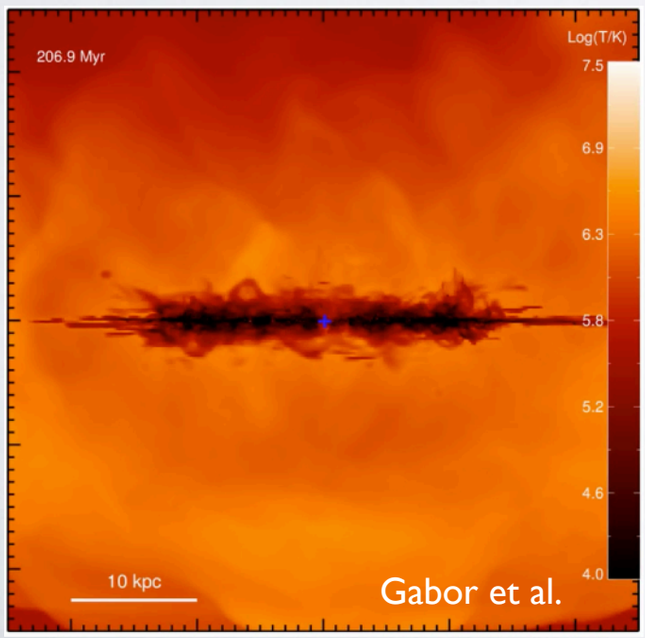


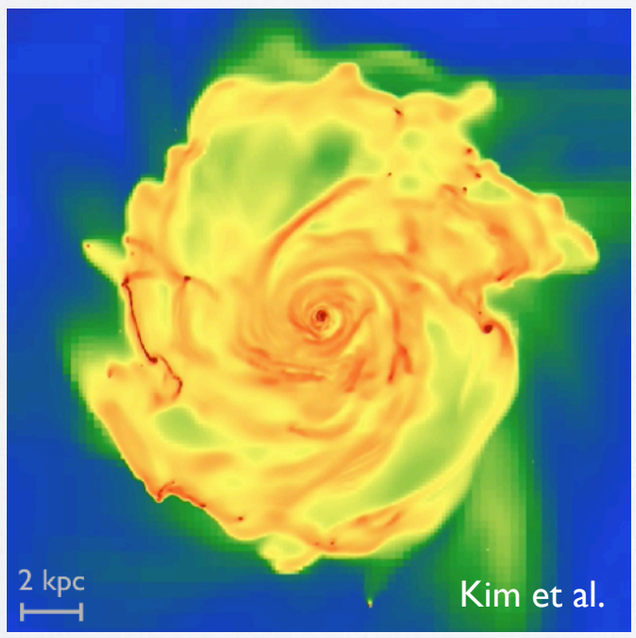
“Idealized” Initial Conditions

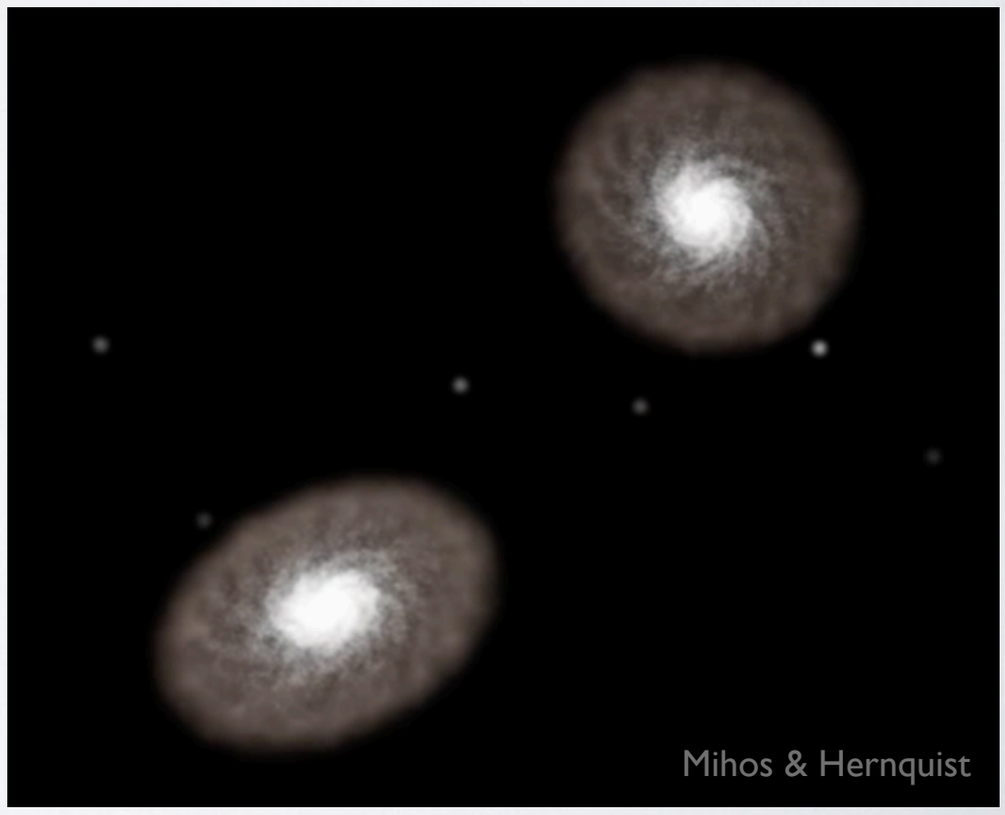
- Start from physically-motivated, yet idealized, often isolated ICs.





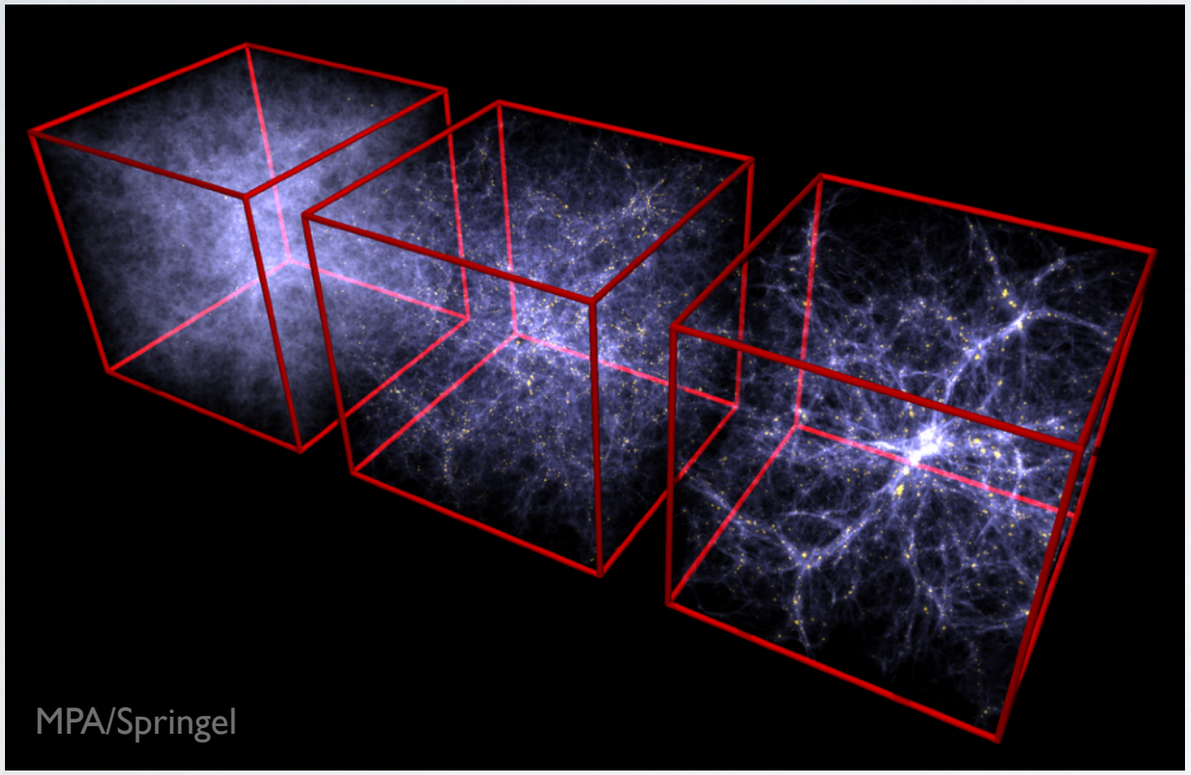




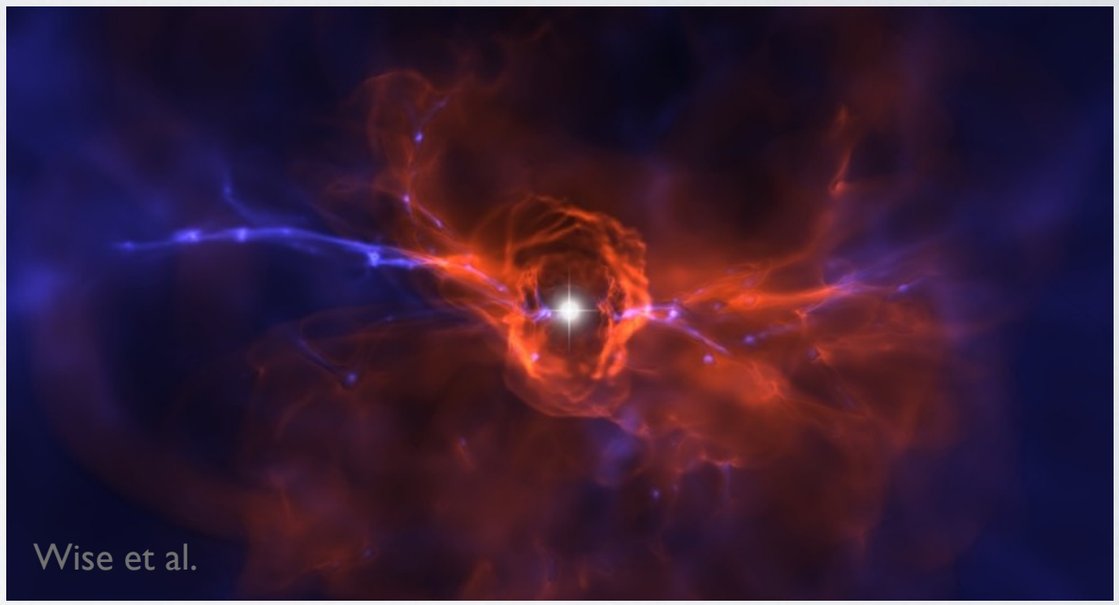


“Cosmological” Initial Conditions

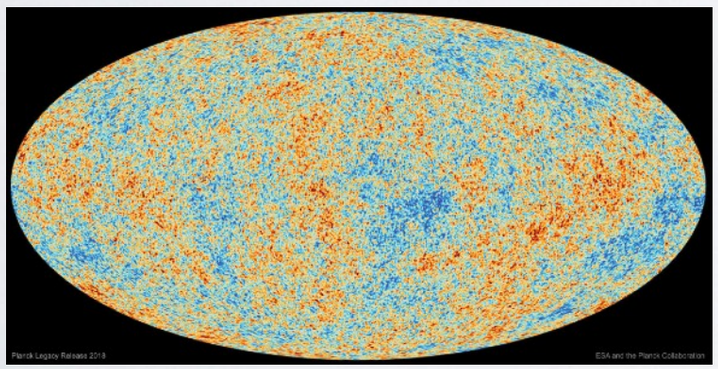
- Start from perturbed density distribution motivated by CMB.



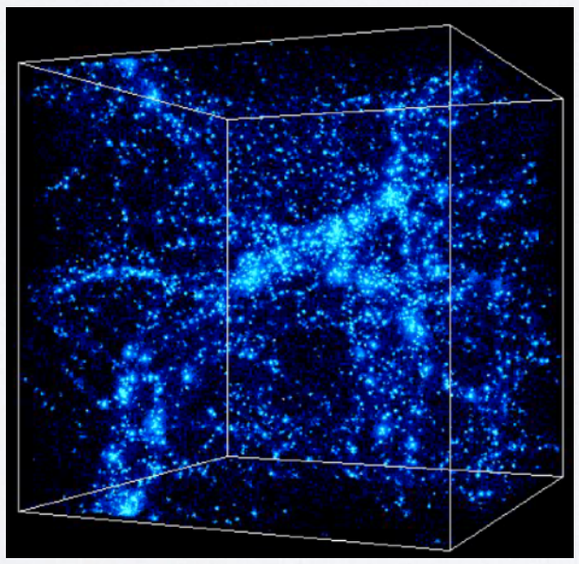
MPA/Springel



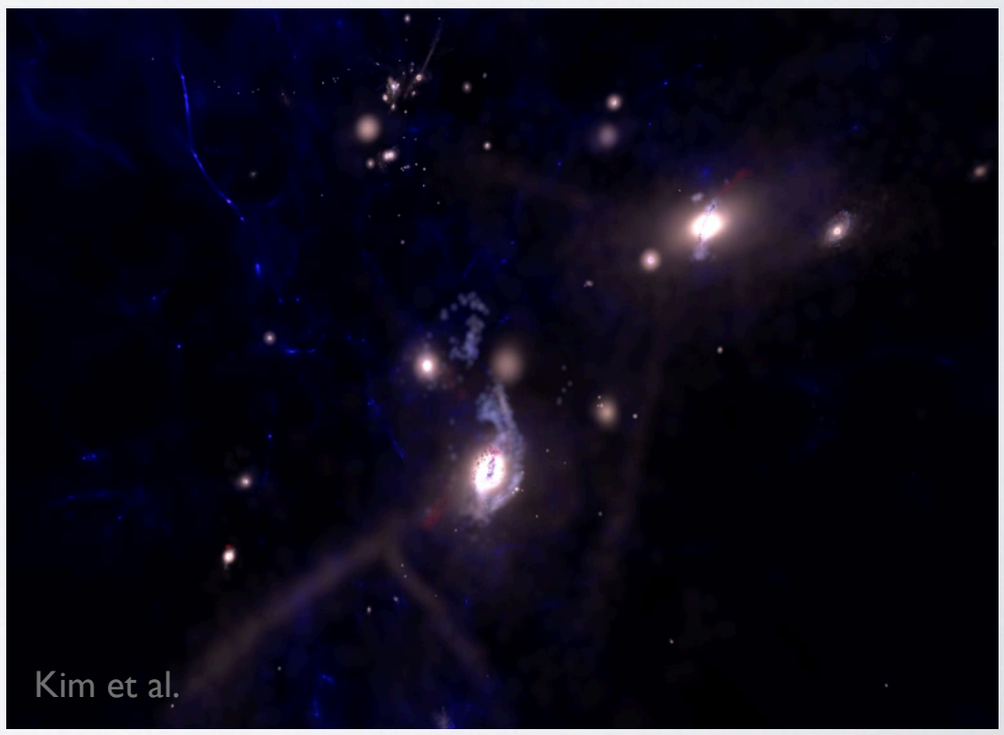
Wise et al.



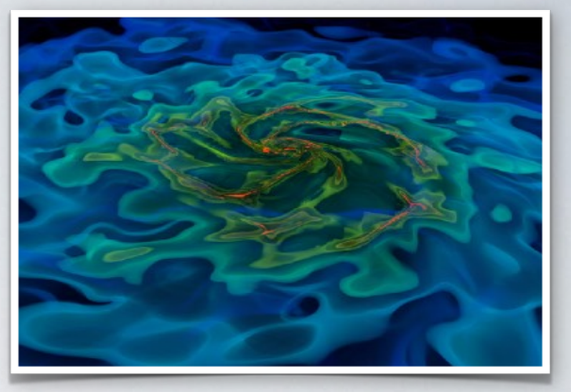
ESA/Planck



Kravtsov et al.

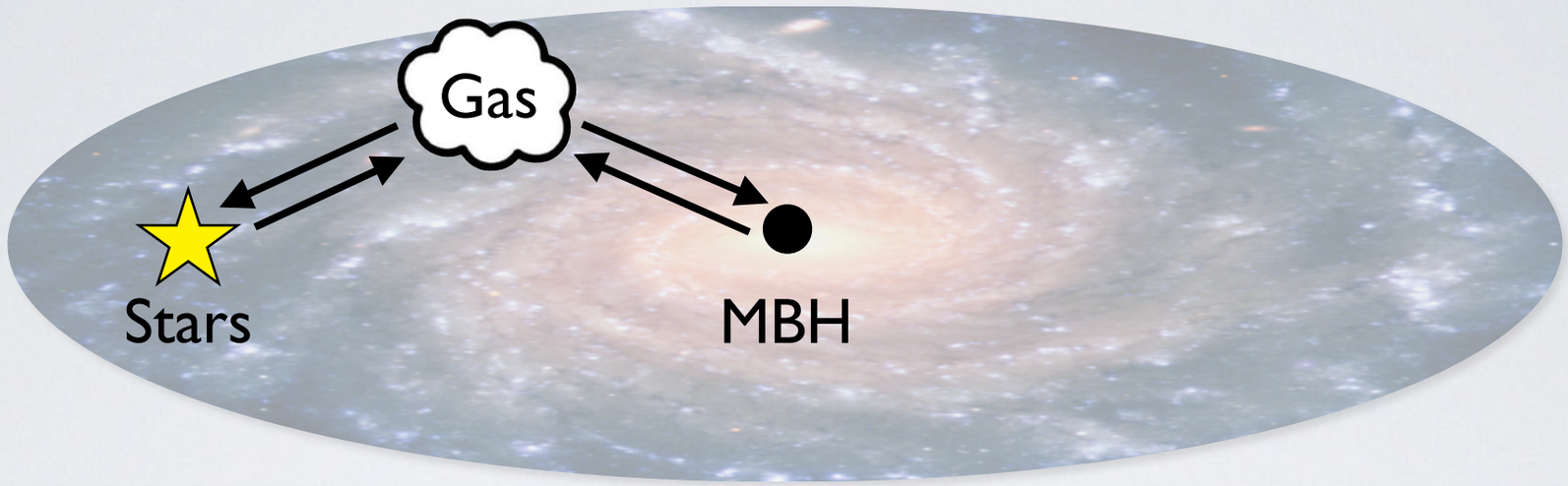


Kim et al.

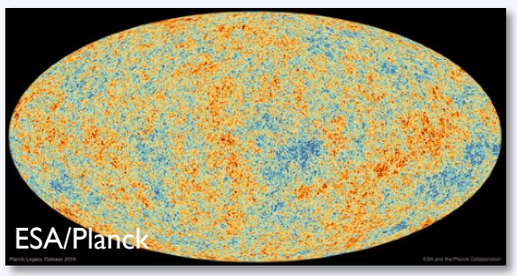


Computational Astrophysics: Practical Examples

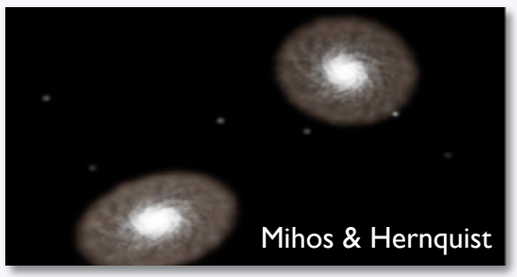
Running An Astrophysical Simulation



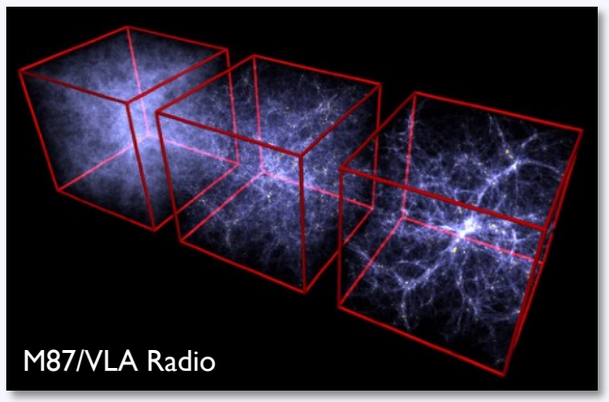
Generate Cosmological IC



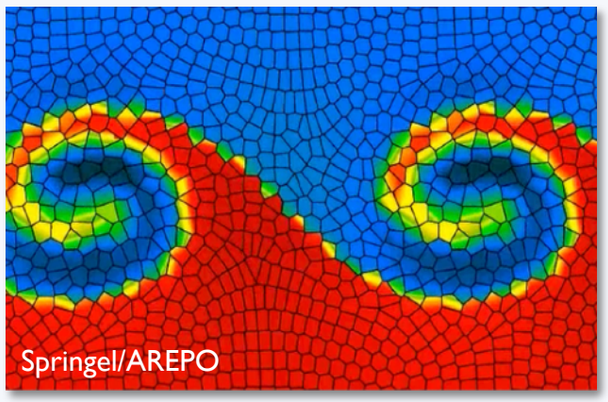
Generate Isolated IC



Run A Simulation

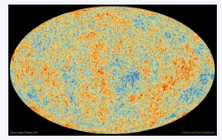


Analyze/Visualize The Simulation



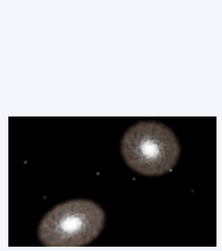
Generate
Cosmological IC

MUSIC
MULTI
SCALE
INITIAL
CONDITIONS



Generate
Isolated IC

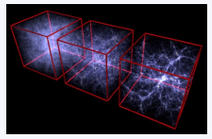
DICE



Run A Simulation

GIZMO

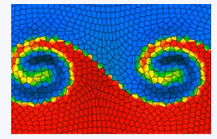
Enzo



Analyze/Visualize
The Simulation

matplotlib

yt



Install
Initialization Software
Along With
Required Libraries

GNU Scientific Library

HDF5

FFTW

Install
Simulation Software
Along With Required
Libraries / Physics Packages

Grackle

Intel® MPI Library

Intel® oneAPI DPC++/C++ Compiler

Intel® oneAPI Threading Building Blocks

Install
Analysis Software
Along With
Required Libraries

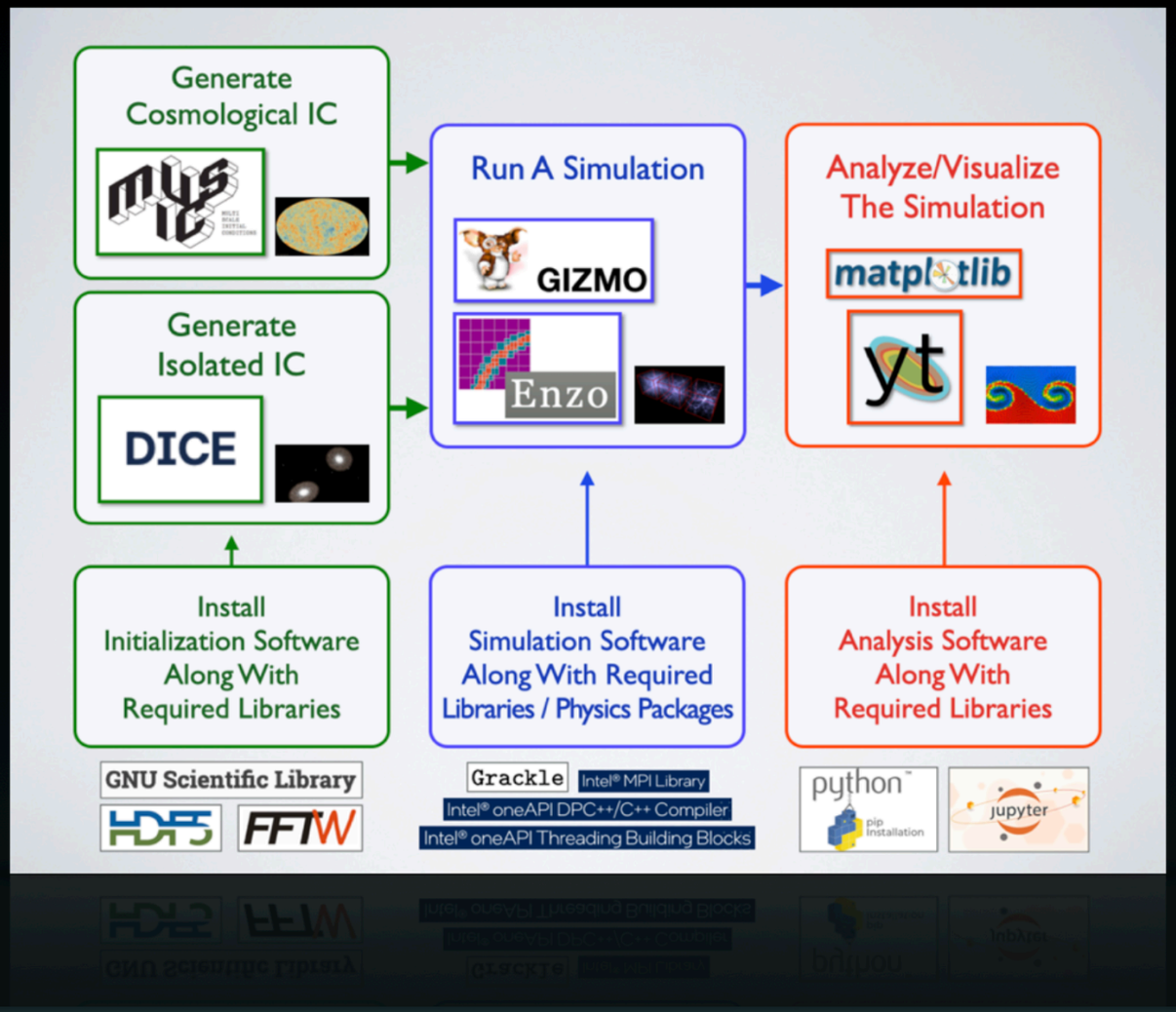
python™

pip
Installation

jupyter

WSL on Windows, Terminal on MacOS, ssh to “Grammar” cluster, VPN client (for off-site use), etc.

linux OS commands, environment setup (.bashrc, module load), library paths, git (version control), slurm manager, shell scripts, etc.



Prerequisites

- Terminal that can access to the KIAS server
  - e.g. WSL2 for window, default terminal in Mac
- Basic knowledge about Linux command (e.g. cd, ls, etc…)
  - If not, refer this page (https://www.hanbit.co.kr/channel/view.html?cmscode=CMS6390061632)
- In this guide, we are going to use following modules (module load xxx)

intel/tbb/2021.3.0	intel/compiler/latest	intel/compiler-rt/2021.3.0
intel/mpi/latest	hdf5_intel19	fftw

- To install something in Linux environment, you must specify correct compiler, path, directory in Makefile. Please keep in mind this

Prerequisites

- ~/.bashrc file
  - this is a configuration file that runs every time when a new interactive shell starts.
  - If you define something here, you don't need to declare it every time manually! (e.g. "export LD_LIBRARY_PATH=…", "module load xxx")
- It is typically used to
  - Set up environment variables (like PATH)
  - Define aliases (shortcut key) and custom functions

TA's .bashrc

```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi


# User specific environment
if ! [[ "$PATH" =~ "$HOME/.local/bin:$HOME/bin:" ]]
then
    PATH="$HOME/.local/bin:$HOME/bin:$PATH"
fi
export PATH
# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/astro2025/install/grackle/grackle-2.
1/lib:/home/astro2025/install/grackle/grackle-3.3/lib:/home/astro2025/install/gsl-
2.8/lib:/opt/ohpc/pub/libs/fftw/lib

# User specific aliases and functions

module load intel/compiler/latest intel/mpi/latest hdf5_intel19 fftw python/3.11.2
alias sc='vi ~/.bashrc'
alias apply='source ~/.bashrc'
alias qs='squeue -u astro2025'
alias err='cat stderr'
alias out='cat stdout'
alias wt='watch -n 1 squeue -u astro2025'
alias qu='squeue'
```

Prerequisites

- ~/.bashrc file
  - this is a configu
    time when a ne
  - If you define so
    need to declare
    (e.g. “export LD
    “module load xx
- It is typically use
  - Set up environm
  - Define aliases (
    functions

Hyeonyong’s slide (page 3)

Softwares - CAC Documents

cac.kias.re.kr/docs/Basic/Softwares/

Finish update

CAC Documents

Search

CAC Documents

User Guide

Basic Usage

Slurm

Softwares

SCHOOL

Softwares

Available softwares

Basic softwares

Program	Version	Module command	Installed Clusters
GCC	12.2.0	module load gnu12/12.2.0	All
GCC	9.4.0	module load gnu9/9.4.0	All
Intel Compiler(PSXE)	19.1.3.304	module load intel/19.1.3.304	All
Intel Compiler(OneAPI)	2021.3.0	module load intel/compiler/2021.3.0	All
Intel MPI(OneAPI)	2021.3.0	module load intel/mpi/2021.3.0	All
Intel MKL(OneAPI)	2021.3.0	module load intel/mkl/2021.3.0	All
Intel Compiler(OneAPI)	2024.2.1	module load intel2024/compiler	Grammar
Intel MPI(OneAPI)	2024.2.1	module load intel2024/mpi	Grammar
Intel MKL(OneAPI)	2024.2.1	module load intel2024/mkl	Grammar
Python	3.11.2	module load python/3.11.2	All

Table of contents

Available softwares

Basic softwares

Licensed softwares

CUDA softwares

OpenSource softwares

Prerequisites

- ~/.ba
  - th
    tir
  - If
    ne
    (e
    "m
- It is t
  - Se
  - De
    fu

index — mornkr@grammar:~ — ssh mornkr@grammar.kias.re.kr — bash

```
[mornkr@grammar ~]$ module avail

------------------------------------------------------------------------------ /opt/ohpc/pub/modulefiles ------------------------------------------------------------------------------
   EasyBuild/4.6.2                    intel/dal/latest                          intel/tbb/latest                          intel2024/ifort32/latest
   IDL/8.8.1                          intel/dal/2021.3.0              (D)       intel/tbb/2021.3.0                (L,D)   intel2024/ifort32/2024.2.1          (D)
   IDL/8.9.0                   (D)    intel/debugger/latest                     intel/tbb32/latest                        intel2024/intel_ipp_ia32/latest
   Mathematica/13.1                   intel/debugger/10.1.2           (D)       intel/tbb32/2021.3.0              (D)     intel2024/intel_ipp_ia32/2021.12    (D)
   Matlab/R2023b_Parallel             intel/dev-utilities/latest                intel/vpl/latest                          intel2024/intel_ipp_intel64/latest
   Matlab/R2023b               (D)    intel/dev-utilities/2021.3.0    (D)       intel/vpl/2021.4.0                (D)     intel2024/intel_ipp_intel64/2021.12 (D)
   ROOT                               intel/dnnl-cpu-gomp/latest                intel/vtune/latest                        intel2024/intel_ippcp_ia32/latest
   ROOT_1                             intel/dnnl-cpu-gomp/2021.3.0    (D)       intel/vtune/2021.5.0              (D)     intel2024/intel_ippcp_ia32/2021.12  (D)
   amd/aocc/4.1.0                     intel/dnnl-cpu-iomp/latest                intel2024/advisor/latest                  intel2024/intel_ippcp_intel64/latest
   amd/aocl/4.1.0                     intel/dnnl-cpu-iomp/2021.3.0    (D)       intel2024/advisor/2024.2          (D)     intel2024/intel_ippcp_intel64/2021.12 (D)
   amd/aocl_gnu/4.0                   intel/dnnl-cpu-tbb/latest                 intel2024/ccl/latest                      intel2024/mkl/latest
   autotools                          intel/dnnl-cpu-tbb/2021.3.0     (D)       intel2024/ccl/2021.13.1           (D)     intel2024/mkl/2024.2                (D)
   cmake/3.24.2                (L)    intel/dnnl/latest                         intel2024/compiler-intel-llvm/latest      intel2024/mkl32/latest
   emacs                              intel/dnnl/2021.3.0             (D)       intel2024/compiler-intel-llvm/2024.2.1 (D)  intel2024/mkl32/2024.2              (D)
   fftw                        (L)    intel/dpct/latest                         intel2024/compiler-intel-llvm32/latest    intel2024/mpi/latest
   gnu12/12.2.0                       intel/dpct/2021.3.0             (D)       intel2024/compiler-intel-llvm32/2024.2.1 (D) intel2024/mpi/2021.13               (D)
   gnu9/9.4.0                         intel/dpl/latest                          intel2024/compiler-rt/latest              intel2024/tbb/latest
   hdf5_GNU                           intel/dpl/2021.4.0              (D)       intel2024/compiler-rt/2024.2.1    (D)     intel2024/tbb/2021.13               (D)
   hdf5_intel                         intel/init_opencl/latest                  intel2024/compiler-rt32/latest            intel2024/tbb32/latest
   hdf5_intel19                (L)    intel/init_opencl/2021.3.0      (D)       intel2024/compiler-rt32/2024.2.1  (D)     intel2024/tbb32/2021.13             (D)
   hwloc/2.7.0                        intel/inspector/latest                    intel2024/compiler/latest                 intel2024/vtune/latest
   intel/19.1.3.304                   intel/inspector/2021.3.0        (D)       intel2024/compiler/2024.2.1       (D)     intel2024/vtune/2024.2              (D)
   intel/advisor/latest               intel/intel_ipp_intel64/latest            intel2024/compiler32/latest               julia/1.10.4
   intel/advisor/2021.3.0      (D)    intel/intel_ipp_intel64/2021.3.0 (D)      intel2024/compiler32/2024.2.1     (D)     llvm/19.1.0
   intel/ccl/latest                   intel/intel_ippcp_intel64/latest          intel2024/debugger/latest                 nodejs
   intel/ccl/2021.3.0          (D)    intel/intel_ippcp_intel64/2021.3.0 (D)    intel2024/debugger/2024.2.1       (D)     openmpi/4.1.5_GNU
   intel/clck/latest                  intel/itac/latest                         intel2024/dev-utilities/latest            openmpi/4.1.5
   intel/clck/2021.3.0         (D)    intel/itac/2021.3.0             (D)       intel2024/dev-utilities/2024.2.0  (D)     openmpi/4.1.6_GNU
   intel/compiler-rt/latest           intel/mkl/latest                          intel2024/dnnl/latest                     openmpi/4.1.6_INTEL                 (D)
   intel/compiler-rt/2021.3.0  (L,D)  intel/mkl/2021.3.0               (D)       intel2024/dnnl/3.5.0              (D)     os
   intel/compiler-rt32/latest         intel/mkl32/latest                        intel2024/dpct/latest                     pmix/4.2.1
   intel/compiler-rt32/2021.3.0 (D)   intel/mkl32/2021.3.0            (D)       intel2024/dpct/2024.2.0           (D)     prun/2.2
   intel/compiler/latest       (L)    intel/mpi/latest                (L)       intel2024/dpl/latest                      python/3.8.8
   intel/compiler/2021.3.0     (D)    intel/mpi/2021.3.0              (D)       intel2024/dpl/2022.6              (D)     python/3.11.2                       (L,D)
   intel/compiler32/latest            intel/oclfpga/latest                      intel2024/ifort/latest                    ucx/1.11.2
   intel/compiler32/2021.3.0   (D)    intel/oclfpga/2021.3.0          (D)       intel2024/ifort/2024.2.1          (D)     valgrind/3.19.0

  Where:
   D:  Default Module
   L:  Module is loaded

If the avail list is too long consider trying:

"module --default avail" or "ml -d av" to just list the default modules.
"module overview" or "ml ov" to display the number of modules for each name.

Use "module spider" to find all possible modules and extensions.
Use "module keyword key1 key2 ..." to search for all possible modules matching any of the "keys".


[mornkr@grammar ~]$
```

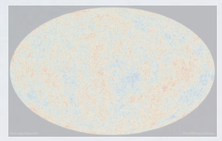
Installation (GSL)

- Required library to compile a simulation code
- Installation step

1. Find the latest gsl file on the internet (Latest version is GSL-2.8)
2. Download the file by using the command `wget` and unzip the file (`tar -xvf`)
3. Go to the GSL directory and start installation by typing following commands
   - ./configure --prefix={your_directory} (this is important!)
   - make
   - make install
   - export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/your_directory/lib

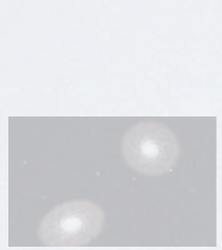
Generate
Cosmological IC

MUSIC
MULTI
SCALE
INITIAL
CONDITIONS



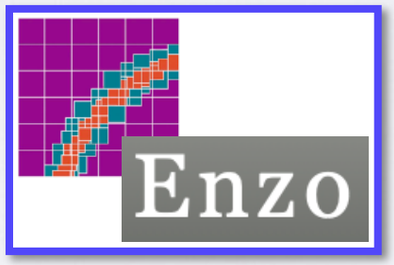
Generate
Isolated IC

DICE

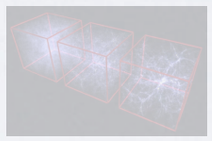


Run A Simulation

GIZMO



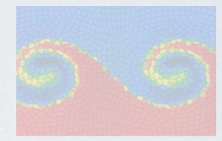
Enzo



Analyze/Visualize
The Simulation

matplotlib

yt



Install
Initialization Software
Along With
Required Libraries

GNU Scientific Library

HDF5

FFTW

Install
Simulation Software
Along With Required
Libraries / Physics Packages

Grackle

Intel® MPI Library

Intel® oneAPI DPC++/C++ Compiler

Intel® oneAPI Threading Building Blocks

Install
Analysis Software
Along With
Required Libraries

python

pip
installation

jupyter

Installation (GRACKLE for GIZMO)

- Chemistry library for the simulation
- Installation step

1. git clone --branch grackle-2.1 --depth 1 https://github.com/grackle-project/grackle.git
   (git clone is the way how we download files in github. You must download old version)
2. Go to this website (https://grackle.readthedocs.io/en/latest/Installation.html) and follow the instruction titled **"Building with Classic Build-System"**

Modify `Make.mach.linux-gnu` file based on the image in the next slides
You have to modify these compile files according to your environment

Installation (GRACKLE for GIZMO)

```
#-----------------------------------------------------------------------
# Install paths (local variables)
#-----------------------------------------------------------------------

LOCAL_HDF5_INSTALL = /opt/ohpc/pub/libs/hdf5_intel19

#-----------------------------------------------------------------------
# Compiler settings
#-----------------------------------------------------------------------
MACH_CC_NOMPI  = icc -std=c11 # C compiler
MACH_CXX_NOMPI = icpc # C++ compiler
MACH_FC_NOMPI  = ifort # Fortran 77
MACH_F90_NOMPI = ifort # Fortran 90
MACH_LD_NOMPI  = icc # Linker
MACH_LIBTOOL   = libtool


#-----------------------------------------------------------------------
# Machine-dependent defines
#-----------------------------------------------------------------------

MACH_DEFINES   = -DLINUX -DH5_USE_16_API -fPIC

#-----------------------------------------------------------------------
# Compiler flag settings
#-----------------------------------------------------------------------

MACH_CPPFLAGS = -P -traditional
MACH_CFLAGS   =
MACH_CXXFLAGS =
MACH_FFLAGS   = -assume no2underscore -extend-source 132
MACH_F90FLAGS = -assume no2underscore -extend-source 132
MACH_LDFLAGS  =
```

Hyeonyong's slide (page 13)

Installation (GRACKLE for GIZMO)

```
#-----------------------------------------------------------------------
# Libraries
#-----------------------------------------------------------------------

LOCAL_LIBS_HDF5   = -L$(LOCAL_HDF5_INSTALL)/lib -lhdf5 # HDF5 libraries
LOCAL_LIBS_MACH   = # -lgfortran # Machine-dependent libraries

MACH_LIBS         = $(LOCAL_LIBS_HDF5) $(LOCAL_LIBS_MACH) -L/opt/ohpc/pub/intel/oneapi/compiler/2021.3.0/linux/compi
ler/lib/intel64_lin -lifcoremt -lifport -limf -lsvml -liomp5 -ldl -lm

#-----------------------------------------------------------------------
# Installation
#-----------------------------------------------------------------------

MACH_INSTALL_PREFIX = /home/astro2025/install/grackle/grackle-2.1
MACH_INSTALL_LIB_DIR = /home/astro2025/install/grackle/grackle-2.1/lib
MACH_INSTALL_INCLUDE_DIR = /home/astro2025/install/grackle/grackle-2.1/include
```

Please note that `MACH_INSTALL_XXX` is the directories where you installed grackle files.
`/home/astro2025/...` should be changed to your own specific paths

Installation (GIZMO)

- If you successfully install GRACKLE, the next step is to install GIZMO
- Installation step
  - Git clone https://github.com/pfhopkins/gizmo-public.git
  - Go to this website and follow installation guide (http://www.tapir.caltech.edu/~phopkins/Site/GIZMO_files/gizmo_documentation.html#tutorial)
  - Change name of `Template_Config.sh` to `Config.sh` and specify `SYSTYPE="Grammar"` in `Makefile.systype`, Add additional lines in `Makefile` with the image in the next slides
  - In `Config.sh`, please uncomment(remove #) following options
    - HYDRO_MESHLESS_FINITE_MASS, PMGRID=512, GALSF, GALSF_FB_MECHANICAL, COOLING, METALS, COOL_GRACKLE, MULTIPLEDOMAINS=16, USE_MPI_IN_PLACE, DOUBLEPRECISION_FFTW
    - BOX_PERIODIC for cosmological simulation only. Please don't use this option for isolated simulation

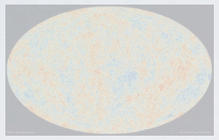
Installation (GIZMO)

```
#-------------------------------------------------------------------
SYSTYPE="Grammar"
ifeq ($(SYSTYPE),"Grammar")
CC       =  mpiicc   # sets the C-compiler
CXX      =  mpiicpc
#OPTIMIZE =  -g -Wall #-fopenmp
GSL_INCL =  -I/home/astro2025/install/gsl-2.8/include
GSL_LIBS =  -L/home/astro2025/install/gsl-2.8/lib -lgsl -lgslcblas -lm
FFTW_INCL=  -I/opt/ohpc/pub/libs/fftw/include
FFTW_LIBS=  -L/opt/ohpc/pub/libs/fftw/lib -lfftw3 -lfftw3_mpi
MPICHLIB =  -L/opt/ohpc/pub/intel/oneapi/mpi/2021.3.0/lib
HDF5INCL = -I/opt/ohpc/pub/libs/hdf5_intel19/include -DH5_USE_16_API
HDF5LIB = -L/opt/ohpc/pub/libs/hdf5_intel19/lib -lhdf5 -lz
OPT     +=  -DOLD_HDF5
endif
#-------------------------------------------------------------------
#-------------------------------------------------------------------
```

- Please note that `/home/astro2025/…` should be changed to your own specific paths
- Don’t forget to specify grackle library path in `Makefile`

Generate
Cosmological IC

MUSIC
MULTI
SCALE
INITIAL
CONDITIONS



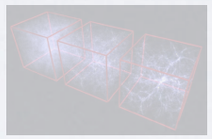
Generate
Isolated IC

DICE

Run A Simulation

GIZMO

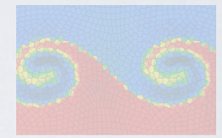
Enzo



Analyze/Visualize
The Simulation

matplotlib

yt



Install
Initialization Software
Along With
Required Libraries

GNU Scientific Library

HDF5

FFTW

Install
Simulation Software
Along With Required
Libraries / Physics Packages

Grackle

Intel® MPI Library

Intel® oneAPI DPC++/C++ Compiler

Intel® oneAPI Threading Building Blocks

Install
Analysis Software
Along With
Required Libraries

python™

pip
installation

jupyter

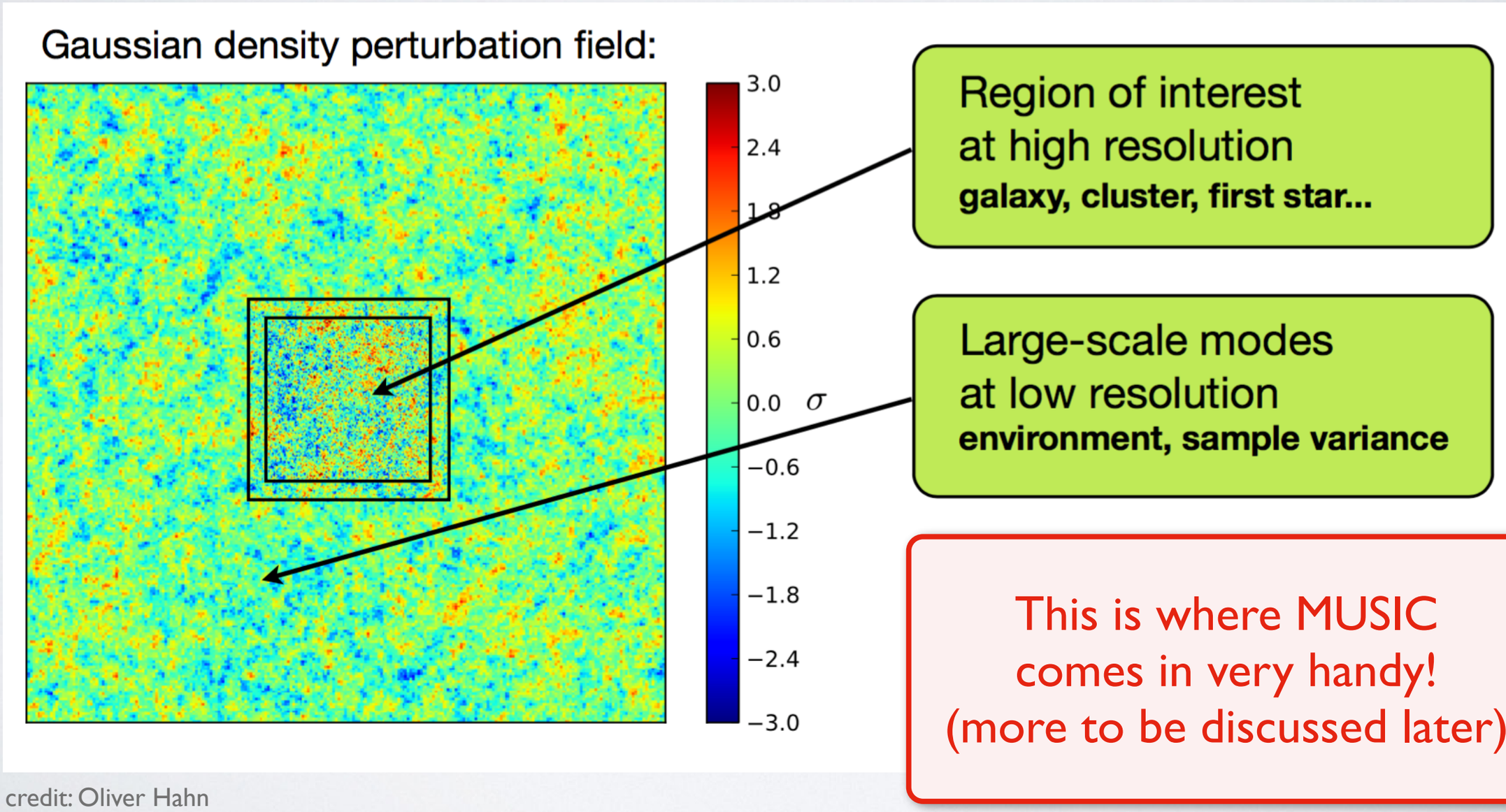
Installation (MUSIC and DICE – IC generator)

```
###############################################
### compiler and path settings
CC       = # Compiler
OPT      = # Optimization
CFLAGS   = # Flags for include
LFLAGS   = # Flags for lib
CPATHS   = # Paths for include
LPATHS   = # Paths for lib

###############################################
```

- You need hdf5, GSL, FFTW for installation
- Installation step (MUSIC)
  - Go to this website and follow their installation guide (https://www-n.oca.eu/ohahn/MUSIC/)
  - Modify `Makefile` file according to your environment (compiler, include and lib path).
    Now you should be able to compile the MUSIC by yourself.
- Installation step (DICE)
  - Go to this website and follow their installation guide (https://bitbucket.org/vperret/dice/src/master/)
  - For "`cmake`", please load the module `cmake/3.24.2`
  - Don't forget to specify the right path of libraries when compiling
    - `cmake .. -DCMAKE_INSTALL_PREFIX={where you want to install DICE} -DGSL_PATH={location of GSL} -DFFTW3_PATH={location of FFTW3}`

Initial Condition: Clever Strategy Needed

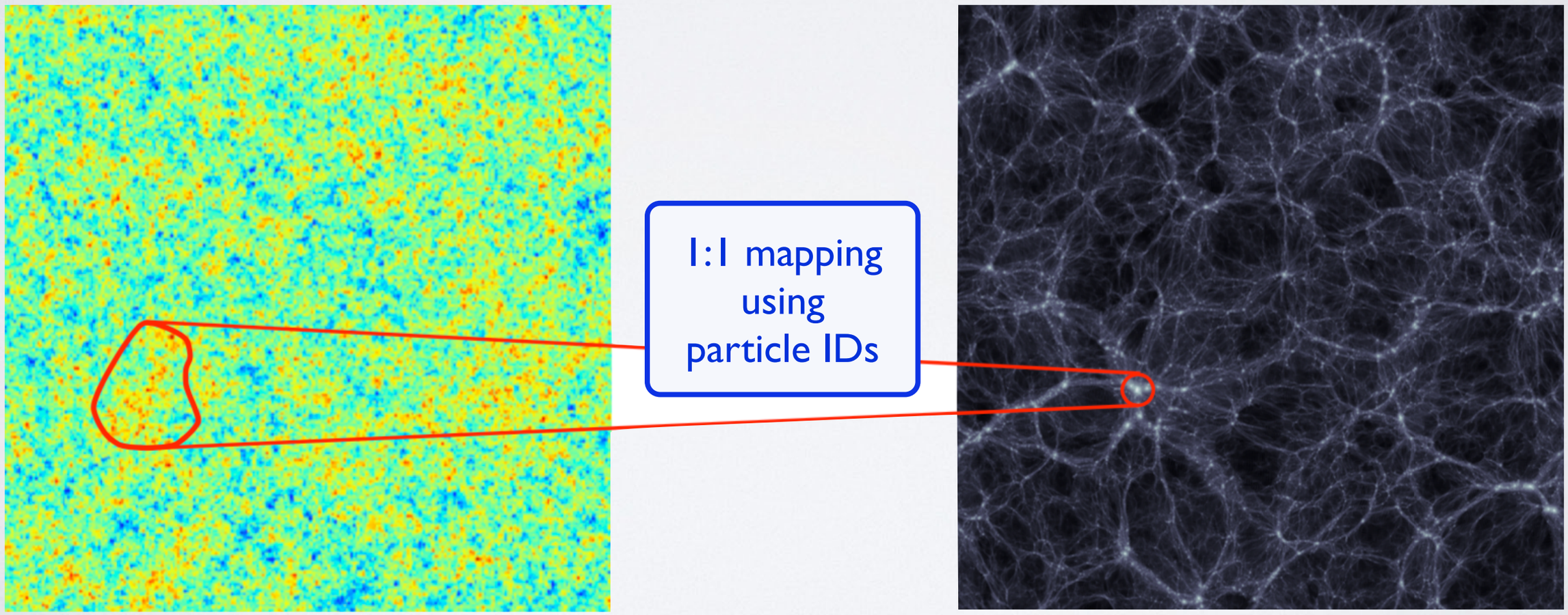
- Clever strategy to best utilize your resources on regions of interest.
- Nested zoom-in IC to capture both large- and small-scale modes.



credit: Oliver Hahn

How To Create "Zoom-in" ICs

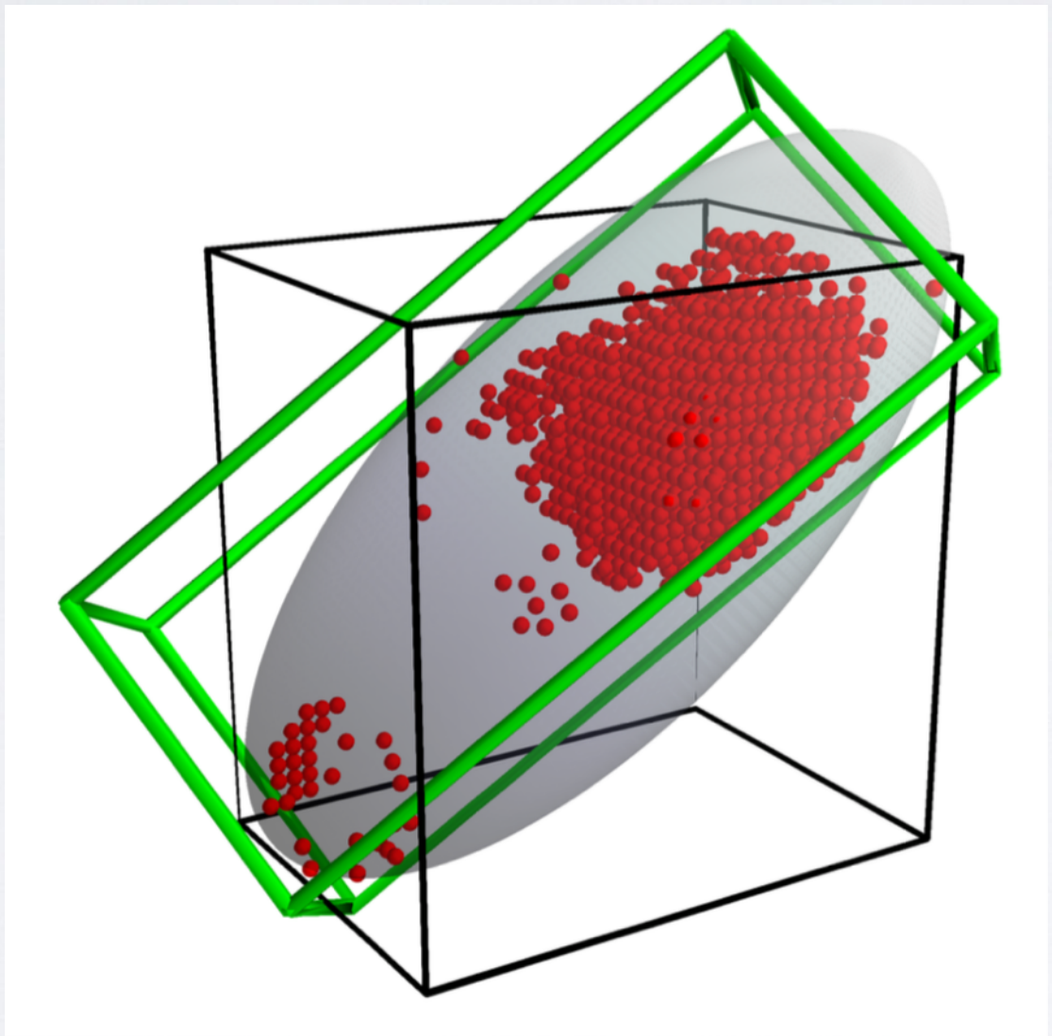
- First, run a low-resolution pathfinder sim to identify target objects.
- Then, find a Lagrangian volume at $z_{initial}$ that encloses all member particles of your target object at z_{final}.



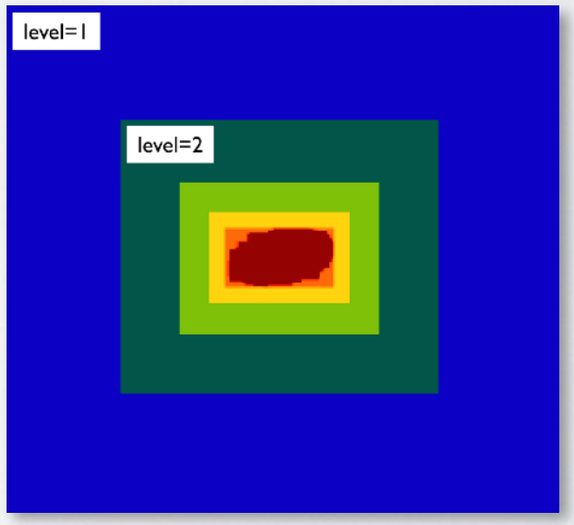
credit: Oliver Hahn

Selecting Lagrangian Volume

- E.g. a Lagrangian volume that encloses all member particles within a viral radius (R_{vir}) of the target halo. (See Onorbe+ 2014)
- It critically depends on the problem you want to investigate.



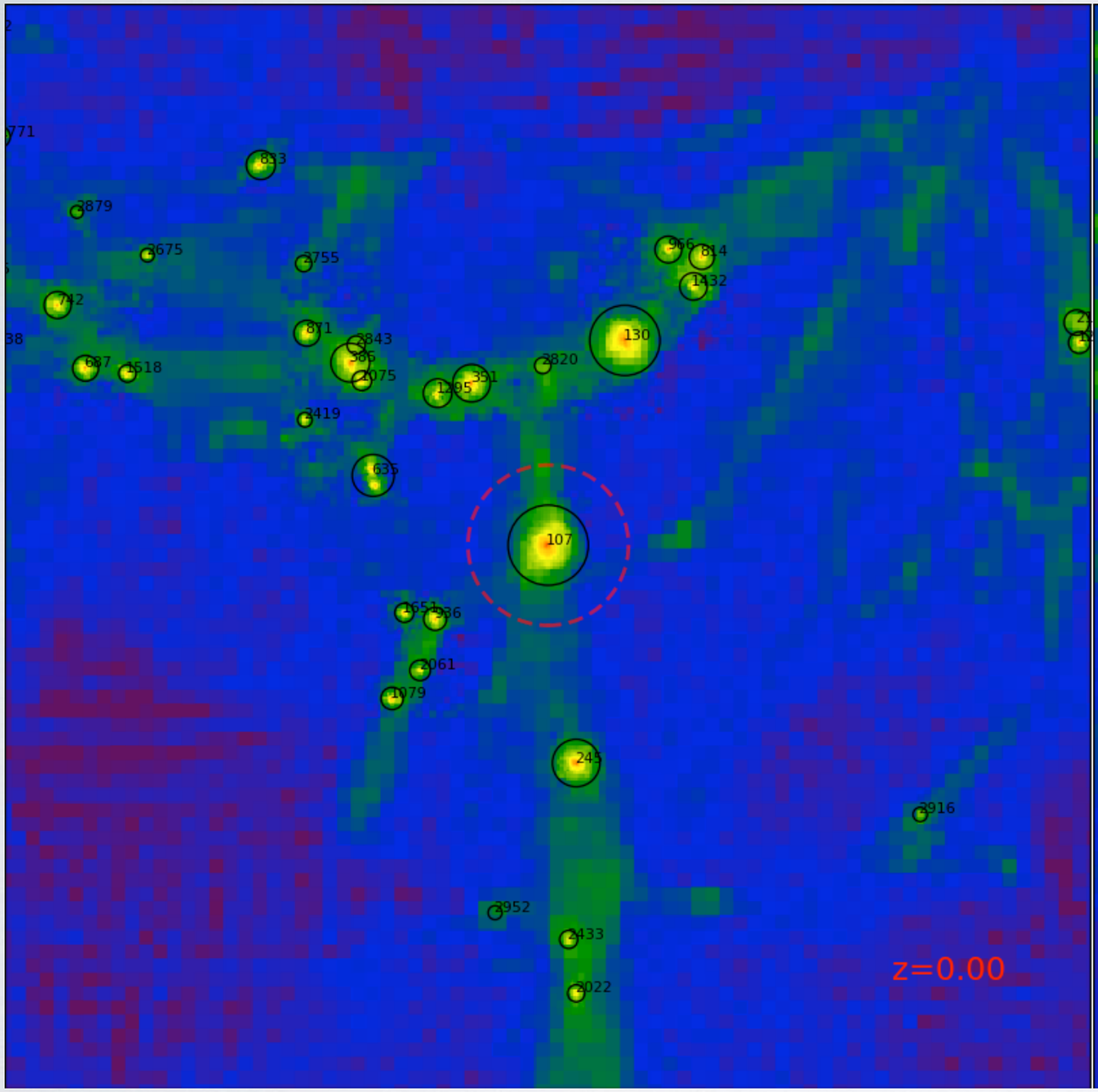
Onorbe et al. (2014)

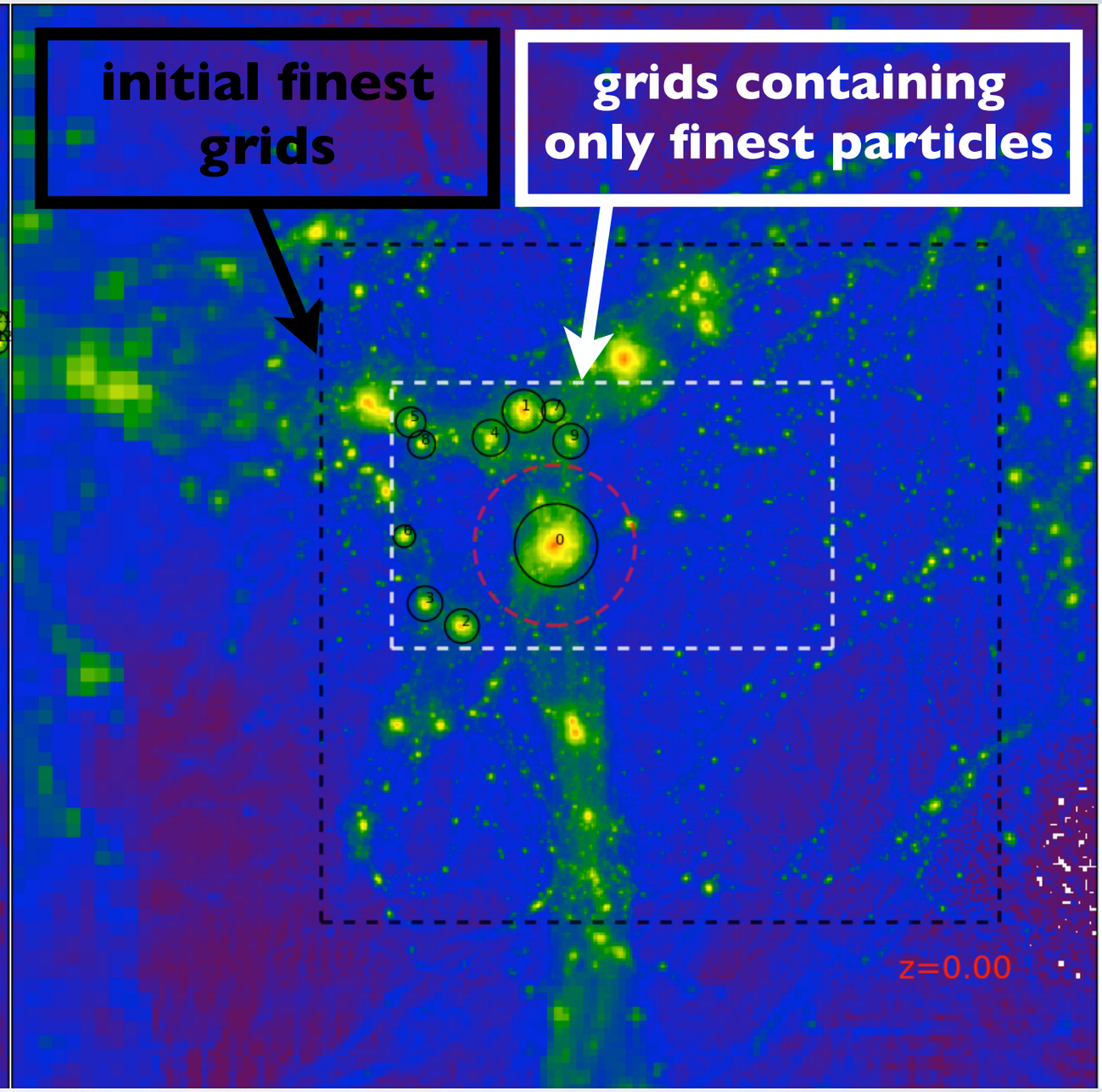


MUSIC IC with finest resolution volume in an ellipsoidal shape

~$10^{12}M_{\odot}$ Halo: Evolution

1e12v

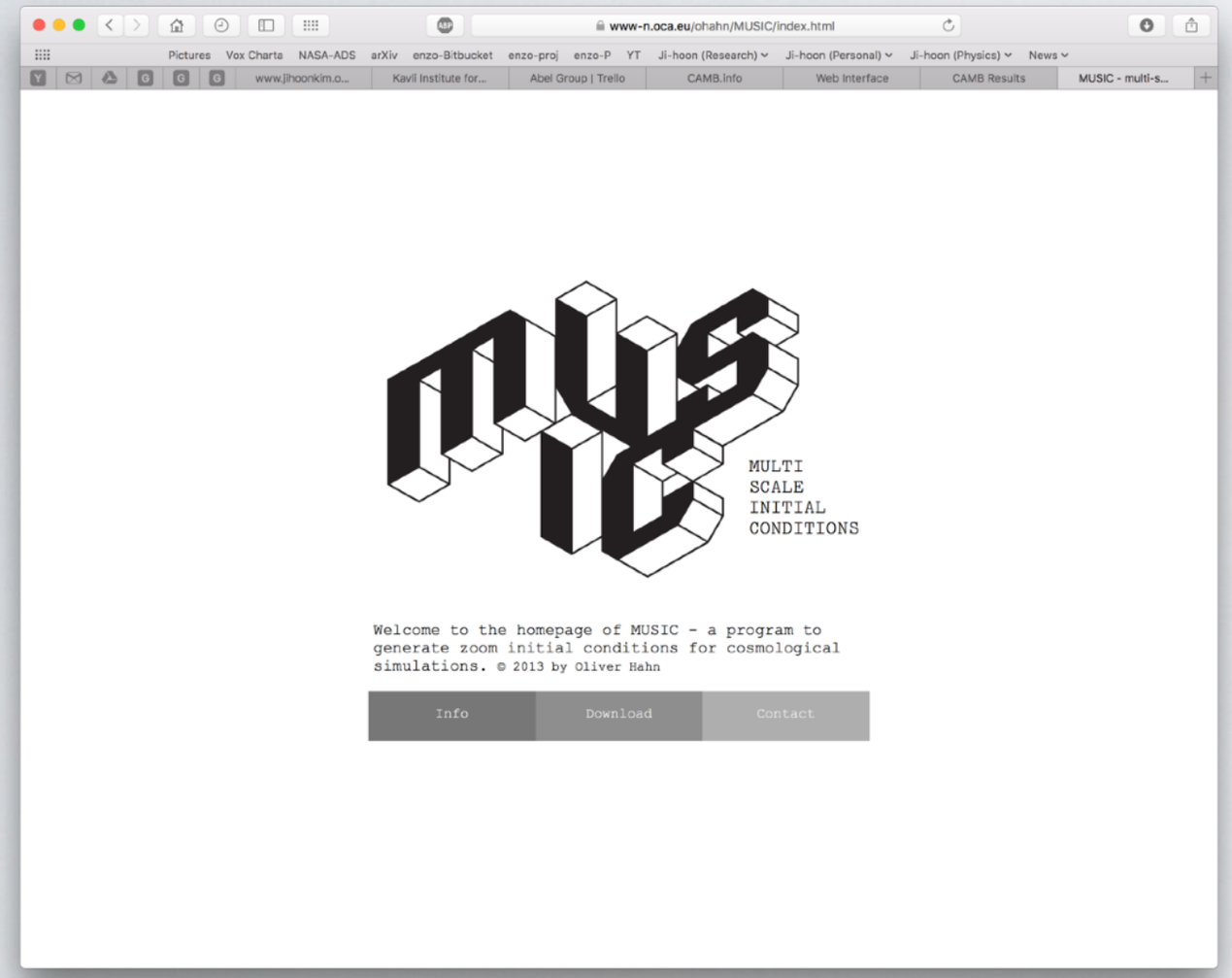




- 1.20×10^{12} $M_{\odot}$ @z=0, 9.15×10^{11} $M_{\odot}$ @z=1
- Lagrangian region of a sphere of radius 946 kpc @z=0 (384×384×512)

MUSIC (2011 - present)

- Webpage: http://www-n.oca.eu/ohahn/MUSIC
- Method paper: Hahn & Abel 2011 (astro-ph/1103.6031)



MUSIC - (MUlti-Scale-Initial-Conditions)

is a computer program to generate nested grid initial conditions for high-resolution "zoom" cosmological simulations. MUSIC currently has the following features:

- Supports output for many cosmological simulation codes via plugins: RAMSES, ENZO, Gadget-2/3, Arepo, ART, Pkdgrav/Gasoline and NyX are currently supported. New codes can be added easily.

- Support for first (1LPT) and second order (2LPT) Lagrangian perturbation theory

- Pluggable transfer functions, currently CAMB, Eisenstein&Hu, BBKS, Warm Dark Matter variants. Distinct baryon+CDM fields are possible.

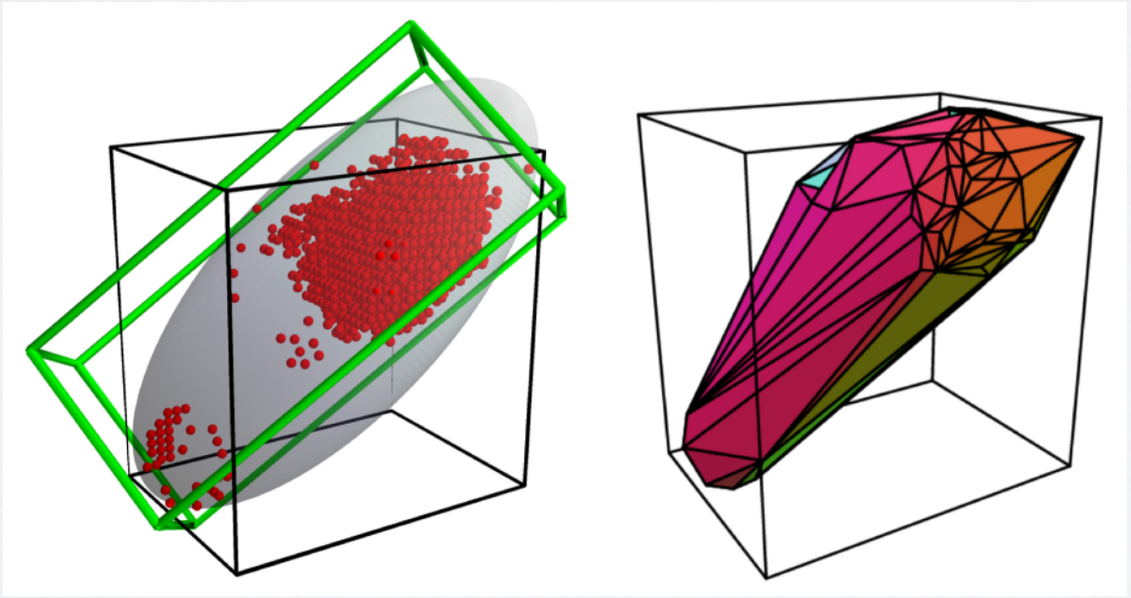
- Minimum bounding ellipsoid and convex hull shaped high-res regions supported with most codes optimizing the high-resolution volume. supports refinement mask generation for RAMSES.

- Parallelized with OpenMP

Requires FFTW (v2 or v3), GSL (and HDF5 for output for some codes)

MUSIC (2011 - present)

- LPT of 2LPT for DM, LLA for baryons in grid codes
- Highly portable ICs: A single parameter is all you need.
- Various output formats: AMR, SPH, AREPO
- Various TF inputs: fitting formula, CAMB
- Various shapes for Lagrangian volume



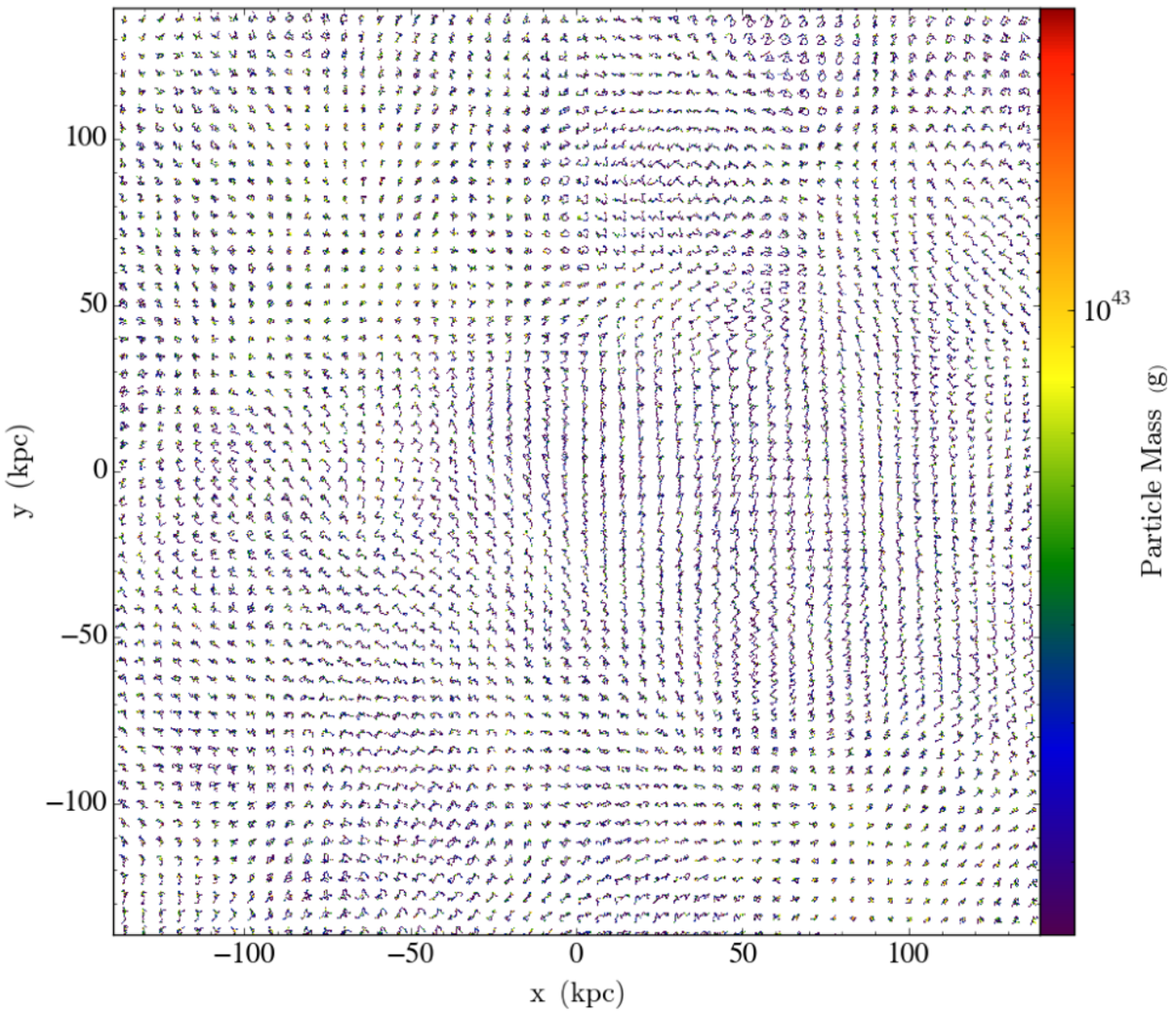
Onorbe et al. (2014)

```
[setup]
boxlength                = 60
zstart                   = 100
levelmin                 = 7
levelmin_TF              = 9
levelmax                 = 12
padding                  = 16
overlap                  = 4
align_top                = no
baryons                  = no
use_2LPT                 = no
use_LLA                  = no
region                   = ellipsoid
region_ellipsoid_matrix[0]   = 2710.833984, -498.042755, -260.366791
region_ellipsoid_matrix[1]   = -498.042755, 1496.330933, 864.111267
region_ellipsoid_matrix[2]   = -260.366791, 864.111267, 5030.364746
region_ellipsoid_center      = 0.638273, 0.576312, 0.447929

[cosmology]
Omega_m                  = 0.272
Omega_L                  = 0.728
Omega_b                  = 0.0455
H0                       = 70.2
sigma_8                  = 0.807
nspec                    = 0.961
transfer                 = eisenstein

[random]
cubesize                 = 256
seed[8]                  = 95064
seed[9]                  = 31415
seed[10]                 = 27183

[output]
format          = enzo
filename        = ic.enzo
##enzo_refine_region_fraction = 0.8

[poisson]
fft_fine                 = yes
accuracy                 = 1e-6
grad_order               = 6
laplace_order            = 6
```

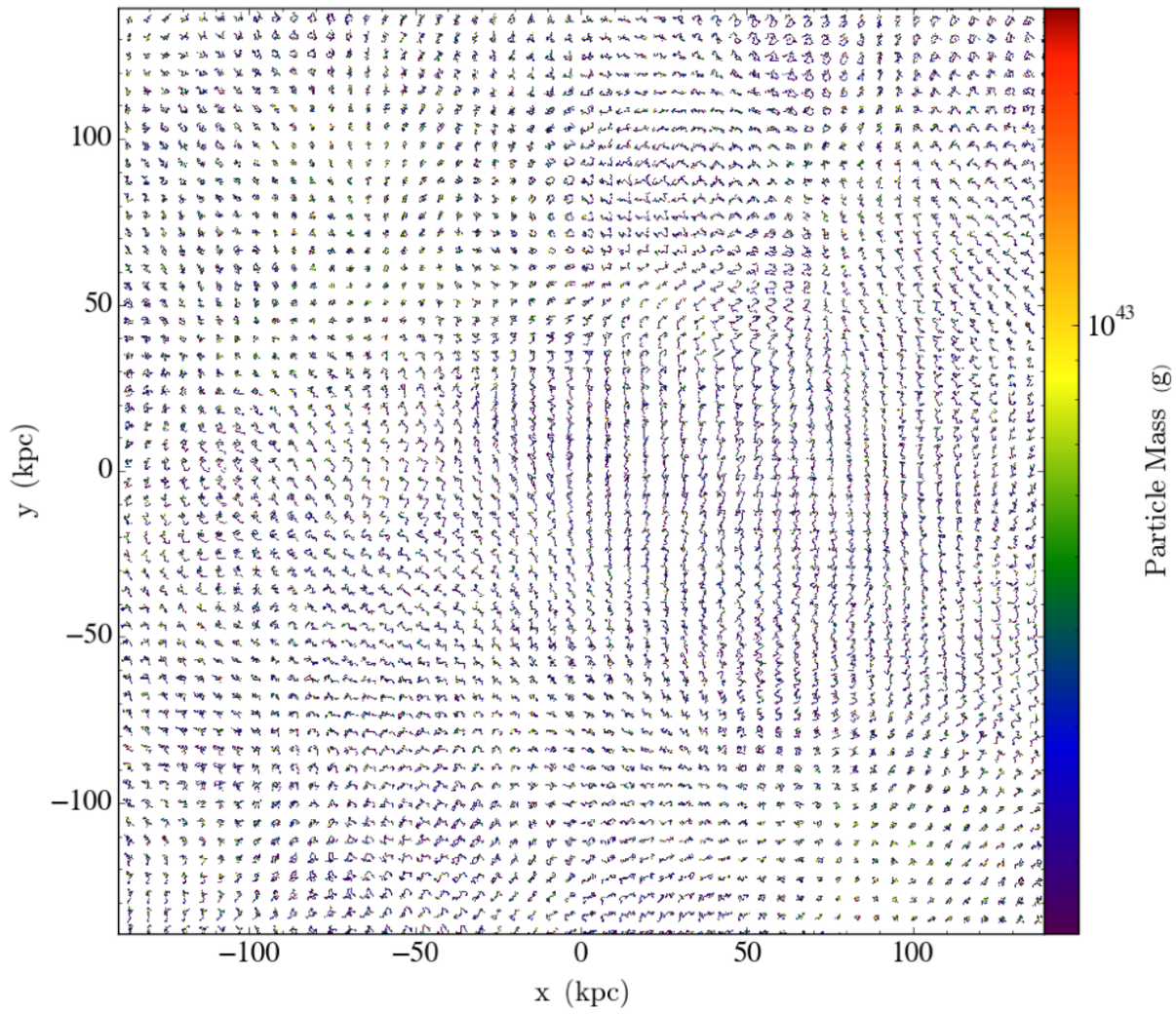
MUSIC parameter file

MUSIC: Position Displacements

- yt's ParticleProjectionPlot() on ENZO and GADGET IC's in the central 10% of the box, $(280\ \mathrm{kpc})^3$.



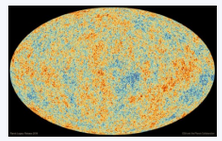
MUSIC IC for ENZO



MUSIC IC for GADGET

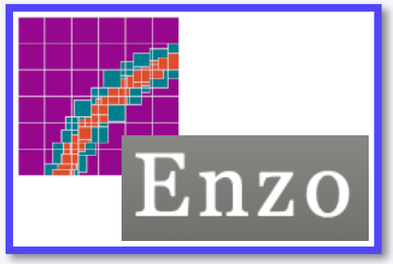
Generate
Cosmological IC

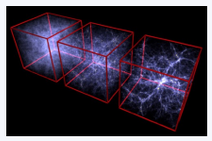
MUSIC
MULTI
SCALE
INITIAL
CONDITIONS



Run A Simulation

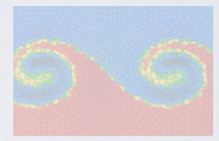
GIZMO





Analyze/Visualize
The Simulation

matplotlib



Generate
Isolated IC

DICE

Install
Initialization Software
Along With
Required Libraries

GNU Scientific Library

HDF5

FFTW

Install
Simulation Software
Along With Required
Libraries / Physics Packages

Grackle

Intel® MPI Library

Intel® oneAPI DPC++/C++ Compiler

Intel® oneAPI Threading Building Blocks

Install
Analysis Software
Along With
Required Libraries

python™

pip
Installation

jupyter

Simulation - Cosmological

- Please visit `/home/astro2025/files` directory
- Copy `lv7_DMO_gizmo.conf, gizmo_cosmo_analysis.ipynb, cosmo_gizmo.tar` to your directory
- Initial condition
  - With your MUSIC execution file, do `./MUSIC lv7_DMO_gizmo.conf` command
  - Then you will get `lv7_gizmo.dat`, initial condition for the cosmological simulation
- Parameter file
  - Unzip `cosmo_gizmo.tar` and put your GIZMO execution file to `cosmo_gizmo`
  - `cosmo_gizmo.tar` includes essential files for running simulations.
- Running a simulation
  - Submit your job by doing `sbatch run.sh` command
  - You can check your status by using `squeue -u {your_ID}`

Simulation - Cosmological

`run.sh` – job submission file. Please do not modify this unless you have a problem

```
#!/bin/bash
#
#SBATCH --job-name=GIZMO_sim            #Job name
#SBATCH --partition=astro               #Partition name
#SBATCH --ntasks=24                     #Number of cores
#SBATCH --cpus-per-task=1               #CPU per task (multi-thread option)
#SBATCH --time=20:00:00                 #Maximum time limiation
#SBATCH --output=slurm-%j.out           # output_log (%j: Job ID)
#SBATCH --error=slurm-%j.err            # err_log

#modules required by this job
module purge
module load intel/compiler/latest intel/mpi/latest hdf5_intel19 fftw

# environmental variables
export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK
export I_MPI_PMI_LIBRARY=/usr/lib64/libpmi2.so

# Main command
srun --mpi=pmi2 ./GIZMO parameter_file.txt 1>stdout 2>stderr

# output log will be stored in stdout, err log will be stored in stderr
```

Hyeonyong's slide (page 19)

Simulation - Cosmological

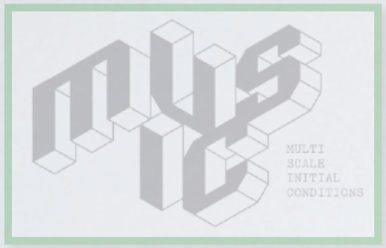
Job status. If it properly runs, R will be shown in ST column

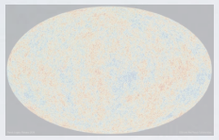
```
[astro2025@grammar gizmo]$ squeue -u astro2025
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
            302230    normal GIZMO_si astro202  R       0:24      6 grammar[021-024,107-108]
```

Output log. You can check your simulation status by "`cat stdout`"

```
Initializing cooling ...
Grackle Initialized
 ..read ionization table [TREECOOL] with 214 non-zero UVB entries in file `TREECOOL'. Make sure to cite the authors
from which the UV background was compiled! (See user guide for the correct references).
Initializing Ewald correction...
 ..initialization of periodic boundaries finished.
Allocated 16.0625 MByte for rhogrid.
Allocated 72.0312 MByte for particle data storage.
Allocated 52.5 MByte for storage of hydro data.

Reading file `lv7.dat' on task=0 (contains 2097152 particles.)
 ..distributing this file to tasks 0-63
Type 0 (gas):          0  (tot=     0000000000) masstab=0
Type 1 (halo):   2097152  (tot=     0002097152) masstab=7.54854
Type 2 (alt):          0  (tot=     0000000000) masstab=0
Type 3 (pic):          0  (tot=     0000000000) masstab=0
Type 4 (stars):        0  (tot=     0000000000) masstab=0
Type 5 (sink):         0  (tot=     0000000000) masstab=0

reading block 0 (Coordinates)...
reading block 1 (Velocities)...
reading block 2 (ParticleIDs)...
reading block 5 (Masses)...
reading block 6 (InternalEnergy)...
Reading done. Total number of particles :  0002097152
```

Generate
Cosmological IC





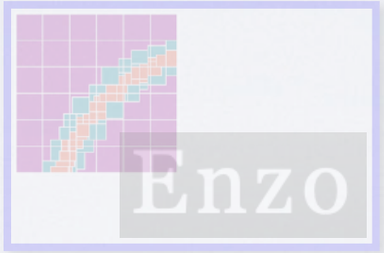
Generate
Isolated IC

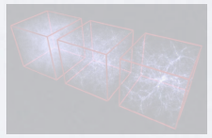
Install
Initialization Software
Along With
Required Libraries

GNU Scientific Library

Run A Simulation

GIZMO



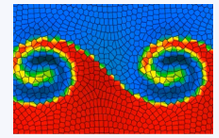


Install
Simulation Software
Along With Required
Libraries / Physics Packages

Grackle
Intel® MPI Library
Intel® oneAPI DPC++/C++ Compiler
Intel® oneAPI Threading Building Blocks

Analyze/Visualize
The Simulation

matplotlib



Install
Analysis Software
Along With
Required Libraries

python™
pip
Installation

jupyter

Simulation - Cosmological

- If everything works well, you will get simulation results in the “output” directory
- With this output files, we are going to generate halo mass function
- Analysis tool installation
  - module load python/3.11.2
  - pip install yt
  - pip install yt-astro-analysis
  - For detailed information, refer this page (https://yt-project.org/doc/installing.html#installing-yt)
  - pip install colossus

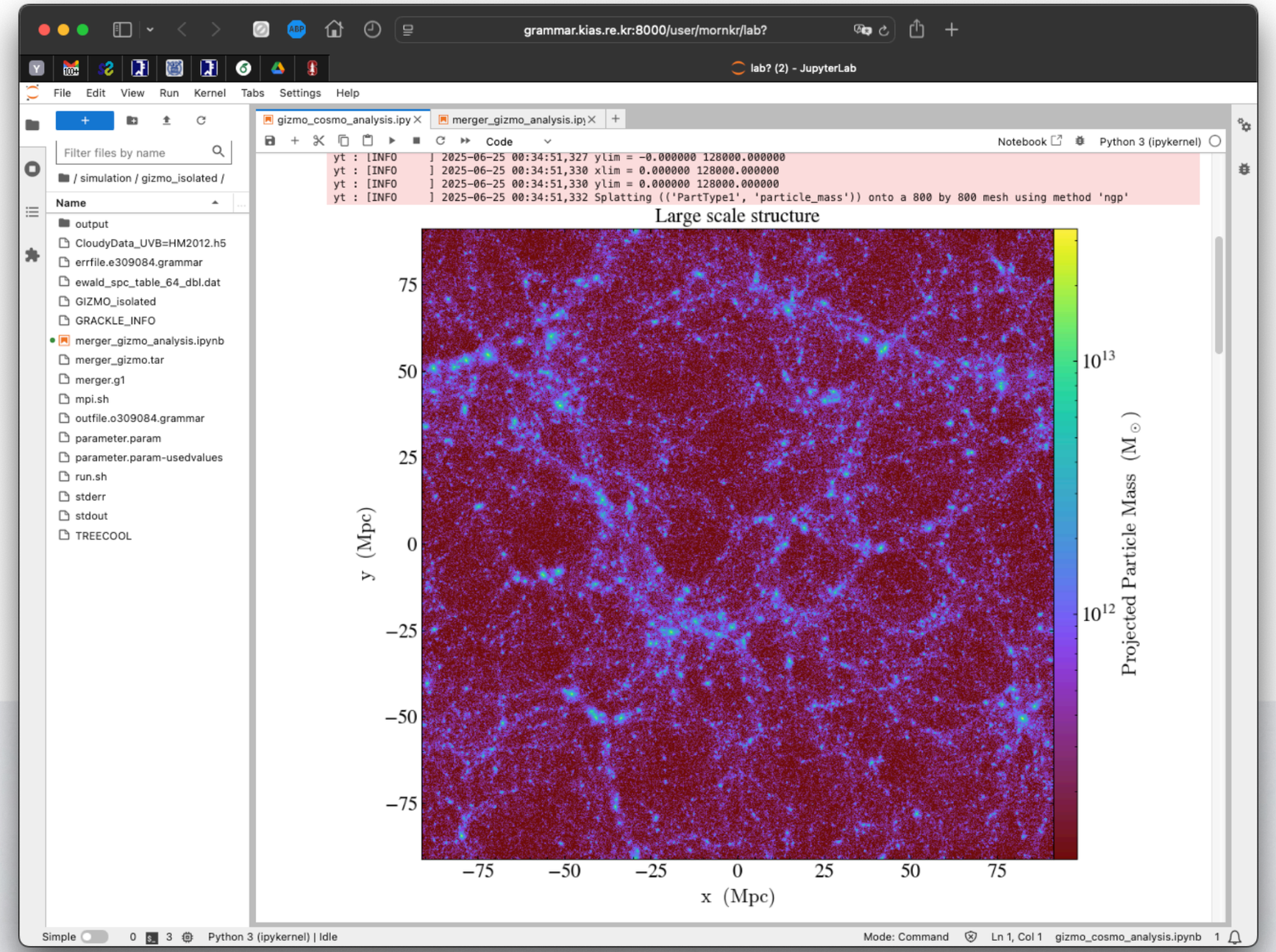
```
yt                          4.4.0
yt-astro-analysis           1.1.3
```

Check by using pip list. Make sure to use the same version here

- Copy `cosmo_gizmo_analysis.ipynb` in your `cosmo_gizmo` directory
- Follow the instruction on the python skeleton file

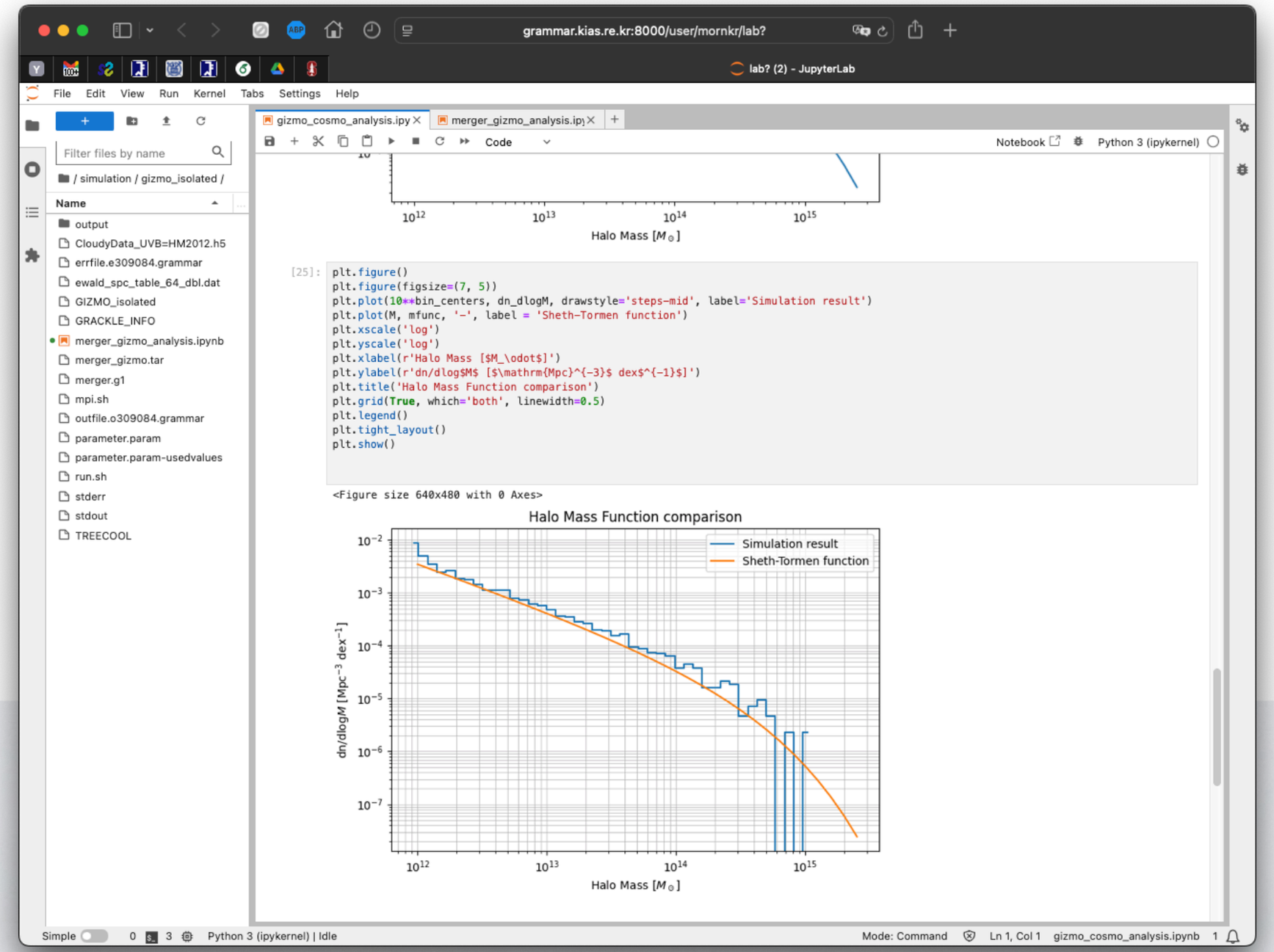
Simulation - Jupyter notebook

- Please visit this website (https://grammar.kias.re.kr:8000/). Here we can use jupyter python notebook



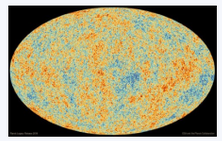
Simulation - Jupyter notebook

- Please visit this website (https://grammar.kias.re.kr:8000/). Here we can use jupyter python notebook



Generate
Cosmological IC

MUSIC
MULTI
SCALE
INITIAL
CONDITIONS

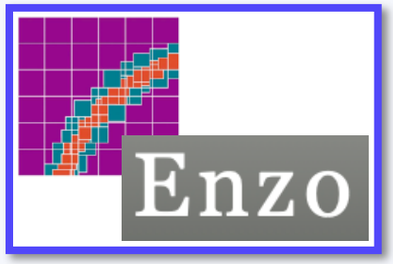


Generate
Isolated IC

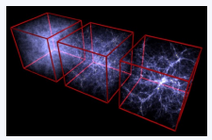
DICE

Run A Simulation

GIZMO



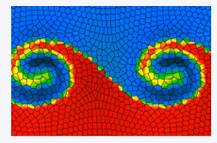
Enzo



Analyze/Visualize
The Simulation

matplotlib

yt



Install
Initialization Software
Along With
Required Libraries

GNU Scientific Library

HDF5

FFTW

Install
Simulation Software
Along With Required
Libraries / Physics Packages

Grackle

Intel® MPI Library

Intel® oneAPI DPC++/C++ Compiler

Intel® oneAPI Threading Building Blocks

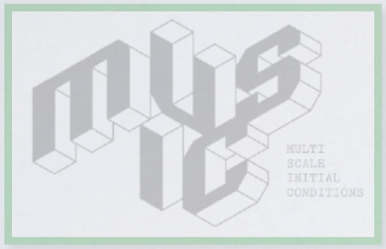
Install
Analysis Software
Along With
Required Libraries

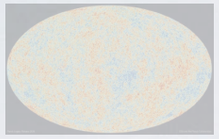
python™

pip
Installation

jupyter

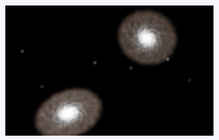
Generate
Cosmological IC





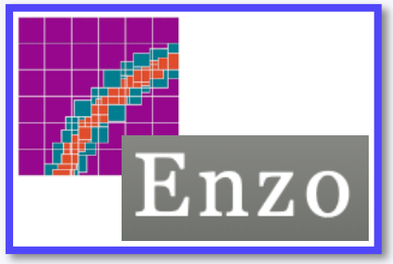
Generate
Isolated IC

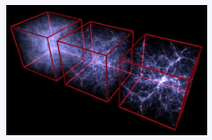
DICE



Run A Simulation

GIZMO

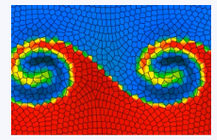




Analyze/Visualize
The Simulation

matplotlib

yt



Install
Initialization Software
Along With
Required Libraries

GNU Scientific Library

HDF5

FFTW

Install
Simulation Software
Along With Required
Libraries / Physics Packages

Grackle

Intel® MPI Library

Intel® oneAPI DPC++/C++ Compiler

Intel® oneAPI Threading Building Blocks

Install
Analysis Software
Along With
Required Libraries

python

pip
installation

jupyter

Simulation - Isolated

- Please visit `/home/astro2025/files` directory
- Copy `merger.config, 1.4e10Msf3C5.params, merger_gizmo.tar, merger_gizmo_analysis.ipynb` to your directory
- Initial condition
  - With your DICE execution file, do `./dice merger.config` command
  - Then you will get `merger.g1`, initial condition for the isolated simulation
- Parameter file
  - Unzip `merger_gizmo.tar` and put your GIZMO execution file to `merger_gizmo`
  - `merger_gizmo.tar` includes essential files for running simulations.
- Running a simulation
  - Submit your job by doing `sbatch run.sh` command
  - You can check your status by using `squeue -u {your_ID}`
- Copy `merger_gizmo_analysis.ipynb` in your `merger_gizmo` directory and follow the instruction on the python skeleton file

Simulation - Jupyter notebook

- Please visit this website (https://grammar.kias.re.kr:8000/). Here we can use jupyter python notebook