AbacusSummit

Nina Maksimova, Lehman Garrison, Daniel Eisenstein, Boryana H

Sep 17, 2021

OVERVIEW

AbacusSummit is a suite of large, high-accuracy cosmological N-body simulations. These simulations were designed to meet (and exceed!) the Cosmological Simulation Requirements of the Dark Energy Spectroscopic Instrument (DESI) survey. AbacusSummit was run on the Summit supercomputer at the Oak Ridge Leadership Computing Facility under a time allocation from the DOE's ALCC program.

AbacusSummit was run using the Abacus N-body code. For more information about the code, see *Abacus*.

The specifications of the ~150 simulations that comprise AbacusSumit are given on the *Simulations* page.

The cosmologies used by these simulations are specified on the *Cosmologies* page.

The available data products (halo catalogs, lightcones, etc) are given on the *Data Products* page.

The abacusutils package can be used to work with AbacusSummit data: https://abacusutils.readthedocs.io. Full examples coming soon!

This documentation and code is hosted in the abacusorg/AbacusSummit GitHub repository. Please file an issue there for questions about AbacusSummit!

CHAPTER

ONE

ABACUS

1.1 What is Abacus?

Abacus is a high-accuracy, high-performance cosmological N-body simulation code. It is optimized for GPU architectures and for large volume, moderately clustered simulations. It is extremely fast: we clock over 30 million particle updates per second on commodity dual-Xeon, dual-GPU computers and nearly 70 million particle updates per second on each node of the Summit supercomputer. But it is also extremely accurate: typical force accuracy is below 10^{-5} and we are using global timesteps, so the leapfrog timesteps away from the cluster cores are much smaller than the dynamical time.

Abacus has been described in several publications. See *Papers & Citation* for a list of these papers.

CompaSO, the Abacus on-the-fly halo finder, is described on the *CompaSO Halo Finder* page.

CHAPTER

TWO

ABACUSSUMMIT

2.1 What is AbacusSummit?

AbacusSummit is a suite of large, high-accuracy cosmological N-body simulations. These simulations were designed to meet (and exceed!) the Cosmological Simulation Requirements of the Dark Energy Spectroscopic Instrument (DESI) survey. AbacusSummit was run on the Summit supercomputer at the Oak Ridge Leadership Computing Facility under a time allocation from the Department of Energy's ALCC program.

Most of the simulations in AbacusSummit are 6912^3 = 330 billion particles in 2 Gpc/*h* volume, yielding a particle mass of about $2 \times 10^9\ \mathrm{M}_\odot/h$.

AbacusSummit consists of over 140 of these simulations, plus other larger and smaller simulations, totaling about 60 trillion particles. Detailed specifications of the *Simulations* and *Cosmologies* are available on other pages.

Key portions of the suite are:

- A primary Planck2018 LCDM cosmology with 25 base simulations (330 billion particles in 2 Gpc/*h*).
- Four secondary cosmologies with 6 base simulations, phase matched to the first 6 of the primary boxes.
- A grid of 79 other cosmologies, each with 1 phase-matched base simulation, to support interpolation in an 8-dimensional parameter space, including w_0, w_a, N_{eff}, and running of the spectral index.
- A suite of 1800 small boxes at the base mass resolution to support covariance estimation
- Other base simulations to match the cosmology of external flagship simulations and to explore the effects of our neutrino approximation.
- A 6x higher mass resolution simulation of the primary cosmology to allow study of group finding, and a large-volume 27x lower mass resolution simulation of the primary cosmology to provide full-sky light cone to z>2.
- Specialty simulations including those with fixed-amplitude white noise and scale-free simulations.
- Extensive data products including particle subsamples, halo catalogs, merger trees, kernel density estimates, and light cones.

CHAPTER

THREE

PAPERS & CITATION

3.1 Citation

Use of AbacusSummit should cite Maksimova et al. (in prep) for the simulation suite and Garrison et al. (2019) and Garrison et al. (2018) for the Abacus code, and Metchnik (2009) for the initial method. Applications using the CompaSO halos should cite Hadzhiyska et al. (in prep) for that method. Other citations may be requested as we publish more of the numerical methods.

Note: We provide a BibTeX file with these references here.

3.2 Papers

Abacus is described in Garrison et al. (2019, MNRAS, 485, 3370), where we detail its performance on the Schneider et al. (2016) code comparison simulation, and in Garrison et al. (2018, ApJS, 236, 43), which released an early suite of 125 simulations from 40 cosmologies (https://lgarrison.github.io/AbacusCosmos/).

Garrison et al. (2016) describes our initial condition methods. Hadzhiyska et al. (in prep) will describe the CompaSO group finding method. Pinto et al. (in prep) will describe the Abacus far-field method. Using scale-free simulations, Joyce et al. (2021) describes accuracy tests, and Garrison et al. (2021) validates the force softening scheme.

CHAPTER

FOUR

SIMULATIONS

4.1 Specifications

This page contains the specification of the simulations in AbacusSummit. We tabulate the simulations below.

Simulation specifications are given a descriptive label, which is included in the simulation name:

- **Base**: this is our standard size, 6912^3 particles in 2 Gpc/h.
- **High**: A box with 6x better mass resolution, 6300^3 in 1 Gpc/h.
- **Highbase**: A 1 Gpc/h box with the base mass resolution.
- **Huge**: these are larger boxes run with 27x worse mass resolution.
- **Hugebase**: Re-runs of some 2 Gpc/h boxes with the same 27x worse mass resolution.
- **Fixedbase**: Simulations with the base mass resolution but fixed-amplitude initial conditions, 4096^3 in 1.18 Gpc/h.
- **Small**: Simulations with base mass resolution but 1728^3 particles in 0.5 Gpc/h.

4.2 Run-Time Products

Only a few of our simulations include the full timeslice output; we typically output only subsamples. The "Full Outputs" column in the simulations table below specifies the redshifts (if any) for which a simulation has full timeslices. This column refers to sets of redshifts using the following abbreviations:

- **Full**: z = 3.0, 2.5, 2.0, 1.7, 1.4, 1.1, 0.8, 0.5, 0.4, 0.3, 0.2, 0.1
- **Partial**: z = 2.5, 1.4, 0.8, 0.2
- **Partial+HiZ**: Adds z = 3.0, 2.0 to the Partial list

Subsamples of particles, with positions, velocities, ID numbers, and kernel density estimates, are typically provided at the same 12 redshifts as the Full list (see *Data Products*). CompaSO group finding is run at these redshifts as well as 21 others (see *CompaSO Halo Finder*).

Note: The 21 "secondary" redshifts are only approximate and may not match from simulation to simulation. We do not shorten the Abacus timestep to land exactly on these secondary redshifts as we do for the primary redshifts. Always use the redshift in the header, not the directory name.

The huge and hugebase sims have fewer group finding and subsample epochs.

Base sims and Huge sims have light-cone outputs; others do not.

A base simulation typically produces about 10 TB of subsampled output, and each output slice is another 4 TB above that.

4.3 Covariance (Small) Boxes

We ran 2000 small simulations in 500 Mpc/h at the base mass resolution, intended for studies of covariance matrices in periodic boundary conditions. These have particle subsample outputs at z = 1.4, 1.1, 0.8, 0.5, and 0.2, as well as halo finding at all redshifts >0.2. However, about 15% of these simulations crashed due to some unresolved issue, almost certainly uncorrelated with any property of the large-scale structure in the simulation. Some of the crashed ones still produced usable outputs at higher redshifts. We have chosen to present the 1883 that yielded outputs at z = 1.1; 1643 of these reached z = 0.2. The numbering between ph3000 and ph4999 will be irregular.

4.4 Simulations Table

Download the simulations table here.

The cosmologies in the "Cosm" column are tabulated in *Cosmologies*.

The "PPD" column is the number of particles-per-dimension.

Note: The following table is wide, you may have to scroll to the right to see all the columns.

SimName	Cosm	Phase	PPD	Box (Mpc/h)	z_Final	Full Outputs
AbacusSummit_base_c000_ph000	000	000	6912	2000	0.1	Full
AbacusSummit_base_c000_ph{001-005}	000	001-005	6912	2000	0.1	Partial+HiZ
AbacusSummit_base_c000_ph{006-024}	000	006-024	6912	2000	0.1	none
AbacusSummit_high_c000_ph100	000	100	6300	1000	0.1	Full
AbacusSummit_highbase_c000_ph100	000	100	3456	1000	0.1	Full
AbacusSummit_huge_c000_ph201	000	201	8640	7500	0.1	1.4, 1.1, 0.8, 0.5, 0.2
AbacusSummit_huge_c000_ph202	000	202	8640	7500	0.1	1.4, 1.1, 0.8, 0.5, 0.2
AbacusSummit_hugebase_c000_ph{000-024}	000	000-024	2304	2000	0.1	1.4, 1.1, 0.8, 0.5, 0.2
AbacusSummit_fixedbase_c000_ph099	000	099	4096	1185	0.1	Full
AbacusSummit_fixedbase_c000_ph098	000	098	4096	1185	0.1	Full
AbacusSummit_small_c000_ph{3000-4999}	000	3000-4999	1728	500	0.2	none
AbacusSummit_base_c001_ph000	001	000	6912	2000	0.1	Partial+HiZ
AbacusSummit_base_c001_ph{001-005}	001	001-005	6912	2000	0.1	Partial
AbacusSummit_base_c002_ph000	002	000	6912	2000	0.1	Partial+HiZ
AbacusSummit_base_c002_ph{001-005}	002	001-005	6912	2000	0.1	Partial
AbacusSummit_base_c003_ph000	003	000	6912	2000	0.1	Partial+HiZ
AbacusSummit_base_c003_ph{001-005}	003	001-005	6912	2000	0.1	Partial
AbacusSummit_base_c004_ph000	004	000	6912	2000	0.1	Partial+HiZ
AbacusSummit_base_c004_ph{001-005}	004	001-005	6912	2000	0.1	Partial
AbacusSummit_fixedbase_c{001-004}_ph099	001-005	099	4096	1185	0.1	Partial
AbacusSummit_base_c009_ph000	009	000	6912	2000	0.1	Partial
AbacusSummit_base_c010_ph000	010	000	6912	2000	0.1	none
AbacusSummit_base_c012_ph000	012	000	6912	2000	0.1	none
AbacusSummit_base_c013_ph000	013	000	6912	2000	0.1	none
AbacusSummit_base_c014_ph000	014	000	6912	2000	0.1	none

Table 1 – continued from previous page

SimName	Cosm	Phase	PPD	Box (Mpc/h)	z_Final	Full Outputs
AbacusSummit_base_c015_ph000	015	000	6912	2000	0.1	none
AbacusSummit_base_c016_ph000	016	000	6912	2000	0.1	none
AbacusSummit_base_c017_ph000	017	000	6912	2000	0.1	none
AbacusSummit_base_c018_ph000	018	000	6912	2000	0.1	none
AbacusSummit_base_c019_ph000	019	000	6912	2000	0.1	none
AbacusSummit_base_c020_ph000	020	000	6912	2000	0.1	none
AbacusSummit_highbase_c021_ph000	021	000	3456	1000	0.1	Partial
AbacusSummit_highbase_c022_ph000	022	000	3456	1000	0.1	Partial
AbacusSummit_base_c{100-115}_ph000	100-115	000	6912	2000	0.1	none
AbacusSummit_base_c116_ph000	116	000	6912	2000	0.1	none
AbacusSummit_base_c{117-126}_ph000	117-126	000	6912	2000	0.1	none
AbacusSummit_base_c{130-181}_ph000	130-181	000	6912	2000	0.1	none

4.5 Missing Data

In the course of running AbacusSummit, a few rare instances of irreversible data loss occurred before the data could be archived. The resulting inhomogeneities in the data products presented by AbacusSummit are recorded here.

- AbacusSummit_base_c103_ph000, for redshifts 0.8 and above, is missing all field particles and all halo RVs (but not halo PIDs). Consequently, the halo cleaning is not present at redshift 0.8 and above.
- AbacusSummit_base_c004_ph003 is missing full (pack9) time slice outputs for redshifts 0.8 and 1.4. And, as a result, no time slice power spectra exist for those redshifts (AB power spectra exist as normal).

CHAPTER

FIVE

COSMOLOGIES

5.1 Cosmology Specifications

This page describes the specification of the Cosmologies and the CLASS parameters that they define. The CLASS parameter files and resulting power spectra and transfer functions are available in the AbacusSummit/Cosmologies directory.

We have given the cosmologies numbers, so that the simulations will refer to c123.

There are various sets of cosmologies in this list:

- The primary cosmology (c000) is a Planck2018 LCDM version, specifically the mean of base_plikHM_TTTEEE_lowl_lowE_lensing. This cosmology has 25 base boxes, plus other mass resolution options. Planck, and AbacusSummit, include a massive neutrino in this cosmology.
- c009 is the same cosmology, but with massless neutrinos, at fixed omega_cb, H0, and sigma_cb. c019 and c020 have two and zero 60 meV neutrino species, at fixed omega_cb, theta_CMB, and sigma_cb.
- There are 4 secondary cosmologies (c001-4), each with 6 base boxes. There is a low-omega_c choice based on WMAP7, a wCDM choice, a high-Neff choice, and a low-sigma8 choice.
- There are ten reference cosmologies (c010, c012-c018, c021-022) that match the choices employed in flagship runs from other groups, so as to ease comparisons between code bases. Note that most of these have massless neutrinos. c011 was not run.
- There is a linear derivative grid (c100-c116) that provides 8 matched pairs, with small positive and negative steps in an 8-dimensional parameter space, plus one additional simulation that is the high-sigma8 partner to the low-sigma8 secondary cosmology. Another set of ten simulations (c117-c126) are pairs with smaller spacing.
- There is a larger unstructured emulator grid (c130-c181) that provides a wider coverage of this 8-dimensional space.

Further details are below the table.

All cosmologies use tau=0.0544. Most use 60 meV neutrinos (also the choice of the Planck 2018 baseline cosmology), omega_nu = 0.00064420, scaling from $z = 1$. We use HyRec, rather than RecFast.

CLASS is run with the pk_ref.pre precision choices, unless the name ends with _fast, in which case we use the defaults. There was one case where CLASS underflowed an integration tolerance with the pk_ref precisions; we reverted to pk_permille.pre for this.

Remember that Omega_m = (omega_b+omega_cdm+omega_ncdm)/h^2.

We output five redshifts from CLASS, $z = 0.0$, 1.0, 3.0, 7.0, and 49, which are called z1,z2,z3,z4,z5.

We use the CDM+Baryon power spectrum at $z = 1$ (z2_pk_cb) and scale back by the ratio of growth factors $D(z_{\rm init})/D(1)$ to define our matter-dominated CDM-only simulation IC. The growth function includes the neutrinos as a smooth component.

5.2 Cosmologies Table

Download the cosmologies table here. However, in analysis applications, users are encouraged to use the cosmological parameters stored as in the `header` field of the ASDF data product files (which is loaded into the `meta` field of Astropy tables, or the `header` field of `CompaSOHaloCatalog` objects) rather than referencing the cosmologies table.

Note: The following table is wide, you may have to scroll to the right to see all the columns.

root	notes	omega_b	omega_cdm
abacus_cosm000	Baseline LCDM, Planck 2018 base_plikHM_TTTEEE_lowl_lowE_lensing mean	0.02237	0.1200
abacus_cosm001	WMAP9+ACT+SPT LCDM, Calabrese++ 2017	0.02242	0.1134
abacus_cosm002	wCDM with thawing model w0 = -0.7, wa = -0.5	0.02237	0.1200
abacus_cosm003	Neff=3.70, from base_nnu_plikHM_TT_lowl_lowE_Riess18_post_BAO	0.02260	0.1291
abacus_cosm004	Low sigma8_matter = 0.75, otherwise Baseline LCDM	0.02237	0.1200
abacus_cosm009	Baseline LCDM with massless neutrinos matching omega_cb & sigma8_cb	0.02237	0.1200
abacus_cosm010	AbacusCosmos Planck LCDM cosmology	0.02222	0.1199
abacus_cosm011	AbacusCosmos Planck LCDM cosmology +10% in sigma8	0.02222	0.1199
abacus_cosm012	Euclid Flagship1 LCDM, sigma8_m=0.8279	0.02200	0.1212
abacus_cosm013	Euclid Flagship2 LCDM, sigma8_m=0.813715, sigma8_cb=0.817135	0.02200	0.1206
abacus_cosm014	OuterRim LCDM, sigma8=0.80	0.02258	0.1109
abacus_cosm015	DarkSky LCDM, sigma8=0.835	0.02215	0.1175
abacus_cosm016	Horizon Run 4 LCDM, sigma8=0.7937	0.02281	0.1120
abacus_cosm017	IllustrisTNG LCDM, sigma8=0.8159	0.02230	0.1194
abacus_cosm018	MultiDark Planck LCDM, sigma8=0.8228	0.02214	0.1189
abacus_cosm019	Baseline LCDM with two 0.06 eV neutrinos	0.02237	0.1200
abacus_cosm020	Baseline LCDM w/ massless neutrinos matching theta_star & sigma8_cb	0.02237	0.1200
abacus_cosm021	MassiveNUs massless base	0.02254	0.12446
abacus_cosm022	MassiveNUs with one 0.1 eV massive neutrino species	0.02254	0.12339
abacus_cosm100	Baseline +2% ln(omega_b)	0.02282	0.1200
abacus_cosm101	Baseline -2% ln(omega_b)	0.02193	0.1200
abacus_cosm102	Baseline +3.3% ln(omega_c)	0.02237	0.1240
abacus_cosm103	Baseline -3.3% ln(omega_c)	0.02237	0.1161
abacus_cosm104	Baseline +0.01 n_s	0.02237	0.1200
abacus_cosm105	Baseline -0.01 n_s	0.02237	0.1200
abacus_cosm106	Baseline +0.02 nrun	0.02237	0.1200
abacus_cosm107	Baseline -0.02 nrun	0.02237	0.1200
abacus_cosm108	Baseline +0.1 w0	0.02237	0.1200
abacus_cosm109	Baseline -0.1 w0	0.02237	0.1200
abacus_cosm110	Baseline +0.4 wa, -0.1 w0	0.02237	0.1200
abacus_cosm111	Baseline -0.4 wa, +0.1 w0	0.02237	0.1200
abacus_cosm112	Baseline +2% sigma8	0.02237	0.1200
abacus_cosm113	Baseline -2% sigma8	0.02237	0.1200
abacus_cosm114	Baseline +0.3 nnu, +3.3% ln(omega_c), +0.01 n_s	0.02237	0.1240
abacus_cosm115	Baseline -0.3 nnu, -3.3% ln(omega_c), -0.01 n_s	0.02237	0.1161

Table 1 – conti

root	notes	omega_b	omega_cdm
abacus_cosm116	Emulator grid around baseline cosmology	0.02237	0.1200
abacus_cosm117	Baseline +0.83% ln(omega_c)	0.02237	0.1210
abacus_cosm118	Baseline -0.83% ln(omega_c)	0.02237	0.1190
abacus_cosm119	Baseline +0.003 n_s	0.02237	0.1200
abacus_cosm120	Baseline -0.003 n_s	0.02237	0.1200
abacus_cosm121	Baseline +0.025 w0	0.02237	0.1200
abacus_cosm122	Baseline -0.025 w0	0.02237	0.1200
abacus_cosm123	Baseline +0.1 wa, -0.025 w0	0.02237	0.1200
abacus_cosm124	Baseline -0.1 wa, +0.025 w0	0.02237	0.1200
abacus_cosm125	Baseline +0.5% sigma8	0.02237	0.1200
abacus_cosm126	Baseline -0.5% sigma8	0.02237	0.1200
abacus_cosm130	Emulator grid around baseline cosmology	0.02237	0.1200
abacus_cosm131	Emulator grid around baseline cosmology	0.02237	0.1086
abacus_cosm132	Emulator grid around baseline cosmology	0.02237	0.1200
abacus_cosm133	Emulator grid around baseline cosmology	0.02237	0.1200
abacus_cosm134	Emulator grid around baseline cosmology	0.02237	0.1326
abacus_cosm135	Emulator grid around baseline cosmology	0.02237	0.1200
abacus_cosm136	Emulator grid around baseline cosmology	0.02073	0.1192
abacus_cosm137	Emulator grid around baseline cosmology	0.02212	0.1271
abacus_cosm138	Emulator grid around baseline cosmology	0.02108	0.1138
abacus_cosm139	Emulator grid around baseline cosmology	0.02416	0.1128
abacus_cosm140	Emulator grid around baseline cosmology	0.02096	0.1221
abacus_cosm141	Emulator grid around baseline cosmology	0.02381	0.1272
abacus_cosm142	Emulator grid around baseline cosmology	0.02287	0.1130
abacus_cosm143	Emulator grid around baseline cosmology	0.02206	0.1278
abacus_cosm144	Emulator grid around baseline cosmology	0.02210	0.1130
abacus_cosm145	Emulator grid around baseline cosmology	0.02428	0.1186
abacus_cosm146	Emulator grid around baseline cosmology	0.02097	0.1180
abacus_cosm147	Emulator grid around baseline cosmology	0.02113	0.1215
abacus_cosm148	Emulator grid around baseline cosmology	0.02289	0.1201
abacus_cosm149	Emulator grid around baseline cosmology	0.02188	0.1200
abacus_cosm150	Emulator grid around baseline cosmology	0.02248	0.1216
abacus_cosm151	Emulator grid around baseline cosmology	0.02315	0.1254
abacus_cosm152	Emulator grid around baseline cosmology	0.02165	0.1148
abacus_cosm153	Emulator grid around baseline cosmology	0.02192	0.1199
abacus_cosm154	Emulator grid around baseline cosmology	0.02158	0.1148
abacus_cosm155	Emulator grid around baseline cosmology	0.02369	0.1184
abacus_cosm156	Emulator grid around baseline cosmology	0.02202	0.1261
abacus_cosm157	Emulator grid around baseline cosmology	0.02247	0.1214
abacus_cosm158	Emulator grid around baseline cosmology	0.02201	0.1262
abacus_cosm159	Emulator grid around baseline cosmology	0.02206	0.1261
abacus_cosm160	Emulator grid around baseline cosmology	0.02223	0.1185
abacus_cosm161	Emulator grid around baseline cosmology	0.02282	0.1274
abacus_cosm162	Emulator grid around baseline cosmology	0.02136	0.1048
abacus_cosm163	Emulator grid around baseline cosmology	0.02135	0.1290
abacus_cosm164	Emulator grid around baseline cosmology	0.02173	0.1091
abacus_cosm165	Emulator grid around baseline cosmology	0.02225	0.1278
abacus_cosm166	Emulator grid around baseline cosmology	0.02306	0.1151
abacus_cosm167	Emulator grid around baseline cosmology	0.02248	0.1400

Table 1 – conti

root	notes	omega_b	omega_cdm
abacus_cosm168	Emulator grid around baseline cosmology	0.02157	0.1084
abacus_cosm169	Emulator grid around baseline cosmology	0.02189	0.1222
abacus_cosm170	Emulator grid around baseline cosmology	0.02319	0.1076
abacus_cosm171	Emulator grid around baseline cosmology	0.02348	0.1272
abacus_cosm172	Emulator grid around baseline cosmology	0.02282	0.1032
abacus_cosm173	Emulator grid around baseline cosmology	0.02322	0.1161
abacus_cosm174	Emulator grid around baseline cosmology	0.02302	0.1064
abacus_cosm175	Emulator grid around baseline cosmology	0.02173	0.1336
abacus_cosm176	Emulator grid around baseline cosmology	0.02238	0.1199
abacus_cosm177	Emulator grid around baseline cosmology	0.02239	0.1344
abacus_cosm178	Emulator grid around baseline cosmology	0.02234	0.1204
abacus_cosm179	Emulator grid around baseline cosmology	0.02240	0.1067
abacus_cosm180	Emulator grid around baseline cosmology	0.02258	0.1209
abacus_cosm181	Emulator grid around baseline cosmology	0.02169	0.1112

5.3 Running CLASS

We used this fork of CLASS with N-body gauge support: https://github.com/wullm/class_public. Specifically, we used git hash 6cf8e3.

The CLASS outputs stored in AbacusSummit/Cosmologies can be reproduced by running this code passing both the `CLASS.ini` file for a given cosmology and the `abacus_base.pre` or `abacus_base_fast.pre` file. For example:

```
$ cd AbacusSummit/Cosmologies/abacus_cosm000
$ <path to class executable> CLASS.ini ../abacus_base.pre
```

5.4 Additional Details

Beyond the Planck2018 LCDM primary cosmology, we chose 4 other secondary cosmologies. One was WMAP7, to have a large change in omega_m, H0, and sigma8. Others were one wCDM, one high Neff, and one low S8.

wCDM: Chose w0=-0.7, wa=-0.5 to be an extreme thawing model.

Neff=3.70 cosmology: Took the chains from base_nnu_plikHM_TT_lowl_lowE_Riess18_post_BAO and averaged those in 3.595 < nnu < 3.90, chosen so that the weighted mean was 3.70. Also standardized As to tau=0.0544.

Low sigma8: Opted to drop the amplitude by about 7.7%, to make sigma8(matter)=0.75. This is a sizeable shift, but there's lots of ways to damp power.

Then we are doing a large grid of cosmologies to provide control of first and second derivatives around the primary LCDM model.

For the grid of positive/negative excursions for linear derivatives around the baseline LCDM, we opted for the simplicity of rectalinear derivatives in ln(omega_b), ln(omega_c), ns, nrun, sigma8_m, w0. Note that we treat sigma8_m, not As, as the independent variable, in the expectation that this will keep large-scale structure closer to constant. For wa, we opt to hold w(z=0.333)=w0+0.25*wa fixed, close to the mirage model. For Neff, the Planck chains suggested substantial degeneracies with omegac and ns, so we opt to move these two along with Neff to stay close to the CMB degeneracy direction.

We added one extra simulation to be the paired opposite to the low-sigma8 secondary cosmology.

For the broader emulator set, we construct the unstructured grid as follows: We place points on the surface of an 8-dimensional unit sphere, denoting these v0..v7, then map them into the 8-dimensional parameter space by:

- sigma8cb = 0.811355 (1 + 0.12 v0 - 0.125 v4 + 0.06 u0), where u0 is another random number, uniform in [-1,1].
- omega_c = 0.1200 exp(0.100 v1 + 0.165 v6)
- ns = 0.9649 + 0.06 v2 + 0.05 v6
- omegab = 0.02237 exp(0.10 v3)
- w0 = -1.0 + 0.3 v4 -0.2 v5
- wa = 0.8 v5
- Nur = 2.0328 + 1.2 v6
- alpha_s = 0.05 v7

These parameter ranges were chosen to be relatively large (5-8 sigma) beyond today's CMB+LSS constraints, but it is important to note that most of an 8-d sphere is not close to the extreme in any one parameter, but rather 1/sqrt(8) of that extreme.

We have continued to have omega_c and ns vary with Nur, and w0 to vary with wa (so that variations in wa hold w(z=0.333) constant). In addition, we opted to have sigma8 vary with w(0.333), not as much as a pure wCDM fit to the CMB would imply, but to partially track that behavior.

Finally, we add extra +-6% scatter to sigma8. Note that if we were holding the amplitude of the CMB anisotropies fixed (and fixed tau), then our parameter variations would vary sigma8 quite a lot. But we have not varied tau or neutrino mass, so we want to allow some scatter in sigma8.

Next, we have to specify the distribution of points on the 8-d unit sphere. We want to keep the points well separated, but also impose some constraints. We seek to have some of the points sit in subspaces, so that one doesn't have to be using the entire 8-dimensional space in all fits. And we want to avoid most antipodal points, as these provide only redundant information about second derivatives (since we already have the linear derivative set). We also want to mildly exceed the number of simulations needed to constrain the second derivatives, so that there is some ability to drop simulations for cross-validation.

We use 49 antipodal pairs of points on the sphere. These are subject to the constraints below, but otherwise are evolved from their random start to an electrostatic glass, resulting in a well dispersed set of points. The constraints:

a) The first 3 pairs are forced to be at the unit vectors in the v0, v1, and v2 directions, which will map to individual extremal excursions in sigma8cb, omega_c, and ns. We retain both points of each pair in the grid, as these are particularly important directions. In all cases below, we keep only the first point of each pair.

b) The next 11 pairs sample only the v0..v3 directions and are constrained to have v4..v7 = 0, so that they will only sample sigma8cb, omega_c, ns, and omega_b. We note that the 4-dimensional space has 10 second derivatives, for which we're 17 simulations (and 14 non-antipodal).

c) The next 14 pairs sample the v0..v5 directions, holding v6..v7 = 0. These will add w0, wa to the space. This introduces 11 new second derivatives.

d) The next 14 pairs sample the v0..v3 + v6..7 directions, holding v4..v5 = 0. These will add Nur and alpha_s to the LCDM space. Again, this introduces 11 new second derivatives.

e) The last 7 pairs sample the full space, and hence have excursions in w0, wa, Nur, and alpha_s. This last subspace has 4 new second derivatives to measure.

The randomness of the starting point was subjected to some patterns on the sign of certain coordinates in order to encourage a glass with better balance in 2-d projections. This was judged simply by eye.

CHAPTER

SIX

N-BODY DETAILS

Here we describe technical choices with respect to the N-body method that affect the accuracy of the outputs (e.g. softening, time stepping, ICs).

All of the simulations start at z = 99 utilizing second-order Lagrangian Perturbation Theory initial conditions following corrections of first-order particle linear theory; these are described in Garrison et al. (2016, see *Papers*) and have a target correction redshift of 12. The particles are displaced from a cubic grid.

The simulations use spline force softening, described in Garrison et al. (2018). Force softening for the standard boxes is 7.2 kpc/h (Plummer equivalent), or 1/40th of the initial particle grid spacing. This softening is fixed in proper (not comoving) distance and capped at 0.3 of the interparticle spacing at early times.

We use global time steps that begin capped at $\Delta(\ln a) = 0.03$ but quickly drop, tied to a criteria on the ratio of the Mpc-scale velocity dispersion to the Mpc-scale maximum acceleration, with the simulation obeying the most stringent case. This is scaled by a parameter eta, which is 0.25 in these simulations. Simulations require about 1100 time steps to reach z = 0.1.

Users of the outputs probably don't need to know much of the numerical details of the code, but there is one numerical concept that enter into the data products. Abacus uses a cubic grid of size CPD^3, chosen to tune code speed. For AbacusSummit, CPD is ususally 1701. Processing proceeds in y-z slabs of cells, and particle outputs are ordered into these cells and slabs. Before being presented to users, these outputs are aggregated into "superslabs" of 51 or 52 slabs.

CHAPTER

SEVEN

DATA PRODUCTS

7.1 Overview

We have designed a set of data products aimed at supporting mock catalogs to be constructed using halo occupation distributions, as well as efficient access to measurements of the density fields. Minimization of data volume has been a high priority. Indeed, we have stored full particle snapshots only for a few simulations. Even with this economy, we are producing 2 PB of data products.

The key data products are:

1. **Halo catalogs** with various statistics computed on-the-fly from the full particle set.
2. **10% subsamples of particles**, consistently selected across redshift, output with position, velocity, kernel density, and particle ID, organized so as to preserve halo membership.
3. A **light cone** stretching from the corner of the box and including a single second periodic copy of the box. This provides an octant of sky to z=0.8 and about 800 sq deg to z=2.4. The outputs are the subsample of particles, as well as the Nside=16384 healpix pixel number for all particles.
4. **Full particle catalogs** for a few timeslices of a few boxes.

We perform group finding at 12 primary redshifts and 21 secondary redshifts. Most users should focus on the primary redshifts.

- **Primary redshifts**: z=0.1, 0.2, 0.3, 0.4, 0.5, 0.8, 1.1, 1.4, 1.7, 2.0, 2.5, 3.0
- **Primary + Secondary redshifts**: z=0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.575, 0.65, 0.725, 0.8, 0.875, 0.95, 1.025, 1.1, 1.175, 1.25, 1.325, 1.4, 1.475, 1.55, 1.625, 1.7, 1.85, 2.0, 2.25, 2.5, 2.75, 3.0, 5.0, 8.0

Note: The 21 "secondary" redshifts are only approximate and may not match from simulation to simulation. We do not shorten the Abacus timestep to land exactly on these secondary redshifts as we do for the primary redshifts. Always use the redshift in the header, not the directory name.

At each primary redshift, we output the properties computed for the L1 halos in `halo_info` files (see *CompaSO Halo Finder*). The list of properties is below.

We also output a subsample of the particle, split into 3% and 7% sets (so 10% total), called "A" and "B", so that users can minimize their data access depending on application. These subsamples are consistent across redshift and are selected based on a hash of the particle ID number, so effectively randomly.

The particles are further split into files based on whether the particle is in a L0 halo or not. The particles belonging to a given L1 halo are contiguous in each of the A and B files, and the start and number of the two sets are in in the `halo_info` files. This allows a user to easily fetch a random subsample of particles from the L1 halo; we use these for satellite galaxy positions.

Particle positions and velocities are output in one file. A separate file contains the unique particle ID number, which is easily parsed as the initial grid position, as well as the kernel density estimate and a sticky bit that is set if the particle has ever been in the most massive L2 halo of a L1 halo with more than 35 particles.

Hence, each primary redshift yields 9 set of files: `halo_info` and then {A,B} x {field,halo} x {RV,PID}.

Each set of files is split into 34 files, containing 50 slabs each (51 for the last), to permit users to work on portions of the simulation.

A potential confusing aspect is that field particles are assigned to a file based on the slab number of their cell, but halo particles are assigned to a file based on the slab number of their L0 halo. L0 halos can span up to 20 slabs (though most are much smaller, as a slab is just over 1 Mpc wide) and can involve hundreds of cells. A L0 halo is assigned to a slab based on the lowest x-value of its cells. In other words, if a halo has particles from slabs C..D, it will be in the file for slab C. So one cannot simply take the halo and field files of matching slab range and get the union of all particles.

Note: Most applications will need to load at least one padding file on either side of the file under consideration in order to ensure all halos and particles within a compact range of X coordinates are present in memory.

For the 21 secondary redshifts, we output the halo catalogs and the halo subsample particle IDs (w/densities and sticky L2 tag) only, so not the positions/velocities nor the field particles.

The purpose of the secondary redshifts is to support generation of merger trees, as the PIDs can be used to associate halos between snapshots. It may also be possible to use the finer redshift sampling to associate halo catalogs to the light cones, as the subsample of particles are the same. For example, one could use the halo catalog to generate a HOD at a redshift, assign a given PID to be a satellite galaxy, then go find that PID in the light cone files.

7.2 Data Model

All files are in ASDF format. Most of the files have only one binary block. The only exception are the `halo_info` files, where every column of the table is a separate block. This allows the user to load only the columns needed for an analysis.

Note: Loading a narrow subset of halo catalog columns can save substantial time and memory. Within abacusutils, use the `fields` argument to the CompaSOHaloCatalog constructor to achieve this.

Within ASDF, we apply compression to each binary block. We do this via the Blosc package, using bit/byte transposition followed by zstd compression. We have found that transposition gives substantially better compression ratios (we chose bit vs byte empirically for each file type) and that `zstd` provides fast decompression, fast enough that one CPU core can keep up with most network disk array read speeds. In https://github.com/lgarrison/asdf, we have provided a fork of the ASDF library that includes `blosc` as a compression option, so that decompression should be invisible to the user. One needs to use this fork of ASDF with `abacusutils`.

The ASDF header is human-readable, meaning one can use a Linux command line tool like `less` to examine the simulation metadata stored in every ASDF file. This was one motivation for choosing ASDF over HDF5. We include various descriptive and quantitative aspects of the simulation in this header.

The halo statistics are in `halo_info` files. These columns of these outputs are described below. Most are substantially compressed, including using ratios (e.g., of radii) to scale variables. As such, the binary format of the columns will differ from that revealed by the Python package.

CRC32 checksums are provided for all files in the `checksums.crc32` file that resides in each directory. These should match the GNU `cksum` utility, pre-installed in most Linux environments. We also offer a fast implementation of `cksum` with about 10x better performance here: https://github.com/abacusorg/fast-cksum.

7.3 Halo Statistics

Here is the list of statistics computed on each CompaSO halo. In most cases, these quantities are condensed to reduce the bit precision and thereby save space; this is in addition to the transposition/compression performed in the ASDF file storage. Sometimes the condensing is simple: e.g., when we have the chance to store a quantity (often a ratio) in the range [0,1], we multiply by 32000 and store as an int16. Others are more complicated, e.g., the Euler angles of the eigenvectors are stored to about 4 degree precision and all packed into an uint16.

We provide a Python package to undo this condensation and expose Astropy tables (and therefore NumPy arrays) to the user. See https://abacusutils.readthedocs.io for details and installation instructions.

The listing below gives the data format in the binary files, but also gives the format that is revealed to the user by the Python when that differs.

Keep in mind that the halo catalog consists of purely L1 halos (see *CompaSO Halo Finder*), and that the spherical overdensity definition is a function of epoch. The value is stored in the `SODensityL1` header field.

- `uint64_t id`: A unique halo number.
- `uint64_t npstartA`: Where to start counting in the particle output for subsample A
- `uint64_t npstartB`: Where to start counting in the particle output for subsample B
- `uint32_t npoutA`: Number of taggable particles pos/vel/aux written out in subsample A
- `uint32_t npoutB`: Number of taggable particles pos/vel/aux written out in subsample B
- `uint32_t ntaggedA`: Number of tagged particle PIDs written out in subsample A. A particle is tagged if it is taggable and is in the largest L2 halo for a given L1 halo.
- `uint32_t ntaggedB`: likewise for subsample B;
- `uint32_t N`: The number of particles in this halo. This is the primary halo mass field.
- `uint32_t L2_N[N_LARGEST_SUBHALOS]`: The number of particles in the largest L2 subhalos
- `uint32_t L0_N`: The number of particles in the L0 parent group
- `float SO_central_particle[3]`: Coordinates of the SO central particle
- `float SO_central_density`: Density of the SO central particle.
- `float SO_radius`: Radius of SO halo (distance to particle furthest from central particle, or a constant if the SO crossing is not reached)
- `float SO_L2max_central_particle[3]`: Coordinates of the SO central particle for the largest L2 subhalo.
- `float SO_L2max_central_density`: Density of the SO central particle of the largest L2 subhalo.
- `float SO_L2max_radius`: Radius of SO halo (distance to particle furthest from central particle) for the largest L2 subhalo

The following quantities are computed using a center defined by the center of mass position and velocity of the largest L2 subhalo. In addition, the same quantities with `_com` use a center defined by the center of mass position and velocity of the full L1 halo.

All second moments and mean speeds are computed only using particles in the inner 90% of the mass relative to this center.

- `float x_L2com[3]`: Center of mass pos of the largest L2 subhalo.
- `float v_L2com[3]`: Center of mass vel of the largest L2 subhalo.
- `float sigmav3d_L2com`: The 3-d velocity dispersion, i.e., the square root of the sum of eigenvalues of the second moment tensor of the velocities relative to the center of mass.

- `float meanSpeed_L2com`: Mean speed of particles, relative to the center of mass.
- `float sigmav3d_r50_L2com`: Velocity dispersion (3-d) of the inner 50% of particles.
- `float meanSpeed_r50_L2com`: Mean speed of the inner 50% of particles.
- `float r100_L2com`: Radius of 100% of mass, relative to L2 center.
- `float vcirc_max_L2com`: Max circular velocity, relative to the center of mass position and velocity, based on the particles in this L1 halo .
- `int16_t sigmavMin_to_sigmav3d_L2com`: Min(sigmav_eigenvalue) / sigmav3d, condensed to [0,30000].
- `int16_t sigmavMax_to_sigmav3d_L2com`: Max(sigmav_eigenvalue) / sigmav3d, condensed to [0,30000].
- `uint16_t sigmav_eigenvecs_L2com`: Eigenvectors of the velocity dispersion tensor, condensed into 16 bits.
- `int16_t sigmavrad_to_sigmav3d_L2com`: sigmav_rad / sigmav3d, condensed to [0,30000].
- `int16_t sigmavtan_to_sigmav3d_L2com`: sigmav_tan / sigmav3d, cndensed to [0,30000].
- `int16_t r10_L2com`, `r25_L2com`, `r33_L2com`, `r50_L2com`, `r67_L2com`, `r75_L2com`, `r90_L2com`, `r95_L2com`, `r98_L2com`: Radii of this percentage of mass, relative to L2 center. Expressed as ratios of r100 and condensed to [0,30000].
- `int16_t sigmar_L2com[3]`: The square root of eigenvalues of the moment of inertia tensor, as ratios to r100, condensed to [0,30000].
- `int16_t sigman_L2com[3]`: The square root of eigenvalues of the weighted moment of inertia tensor, in which we have computed the mean square of the normal vector between the COM and each particle, condensed to [0,30000].
- `uint16_t sigmar_eigenvecs_L2com`: The eigenvectors of the inertia tensor, condensed into 16 bits
- `uint16_t sigman_eigenvecs_L2com`: The eigenvectors of the weighted inertia tensor, condensed into 16 bits
- `int16_t rvcirc_max_L2com`: radius of max circular velocity, relative to the L2 center, stored as the ratio to r100 condensed to [0,30000].

7.3.1 Units

The units of positions/radii and velocities, as unpacked by `abacusutils` in Python, are comoving Mpc/h and proper km/s.

In the raw `halo_info` files on disk, positions and radii (where not normalized in a ratio) are in units of the unit box, while velocities are in km/s. Densities are in units of the cosmic mean (so the mean density is 1).

The Abacus convention is to store positions in the range [-BoxSize/2, BoxSize/2), so if your code expects [0, BoxSize) positions, you may need to apply periodic wrap. A wrap is recommended instead of a shift of +BoxSize/2 because the former preserves the origin of the box, which is sometimes useful when comparing with other data products or other N-body codes that have run the same simulation.

The primary halo mass field is `N`, the number of particles in the halo. This can be converted to M/h units with the `ParticleMassHMsun` header field.

The conversion of proper km/s to comoving redshift-space displacement may be achieved by multiplying by `BoxSize/VelZSpace_to_kms`. The second factor gets to unit-box comoving RSD, and the first brings it to BoxSize-box.

7.3.2 Known Bugs

None at present.

7.3.3 Frequently Asked Questions

SO_radius special value

Some fraction of low-mass halos have `SO_radius` all equal to the same value, which is approximately 1.36 Mpc/*h* in the "base" sims. This occurs when the halo still does not drop below the SO threshold even at its most distant particle. The enclosed density is guaranteed to be below the SO threshold at this special value, but if the exact value is desired, one can trivially compute this using the halo mass and the mean background density. (Previously, this special value had been listed on this website as a bug, but this has now been clarified as intended behavior of CompaSO).

The exact value of the special number is a computational detail but may be computed as `X*BoxSize/CPD*sqrt(3)/3`, where `X` is equal to 2 in the overwhelming majority of cases. It is allowed to be equal to 3 or greater integers, but in AbacusSummit base simulations, this is about 7 orders of magnitude rarer.

None of the other radial fields (e.g. the radial percentiles, `r10`, `r25`, etc) should exhibit a special value in this way.

7.4 Particle data

The particle positions and velocities from subsamples are stored in "RV" files. The positions and velocities have been condensed into three 32-bit integers, for x, y, and z. The positions map [-0.5,0.5] to +-500,000 and are stored in the upper 20 bits. The velocites are mapped from [-6000,6000) km/s to [0,4096) and stored in the lower 12 bits. The resulting Nx3 array of int32 is then compressed within ASDF.

The particle positions and velocities from full timeslices are stored in `pack9` files. These provide mildly higher bit precision, albeit with some complexity. Particles are stored in cells (a cubic grid internal to Abacus). Each cell has a 9-byte header, containing the cell 3-d index and a velocity scaling, and then each particle is stored as 9 bytes, with 12 bits for each position and velocity component. As the base simulations have 1701 cells per dimension, this is about 23 bits of positional precision.

The particle ID numbers and kernel densities are stored in `PID` files packed into a 64-bit integer. The ID numbers are simply the `(i,j,k)` index from the initial grid, and these 3 numbers are placed as the lower three 16-bit integers. The kernel density is stored as the square root of the density in cosmic density units in bits 1..12 of the upper 16-bit integer. Bit 0 is used to mark whether the particle has ever been inside the largest L2 halo of a L1 halo with more than 35 particles; this is available to aid in merger tree construction.

When using the `npstartA'' and ``npoutA` fields to index the halo particle subsamples, one might noticed that `sum(halos['npoutA'])` is less than `len(halo_subsamples)`. In other words, there are unindexed halo particles. This is because the subsamples are taken from the L0 particles, but only the L1 particles are indexed by halos (see *CompaSO Halo Finder* for the distinction).

7.5 Light Cones

For the base boxes, the light cone is structured as three periodic copies of the box, centered at (0,0,0), (0,0,2000), and (0,2000,0) in Mpc/h units. This is observed from the location (-950, -950, -950), i.e., 50 Mpc/h inside a corner. This provides an octant to a distance of 1950 Mpc/h (z=0.8), shrinking to two patches each about 800 square degrees at a distance of 3950 Mpc/h (z=2.4).

The three boxes are output separately and the positions are referred to the center of each periodic copy, so the particles from the higher redshift box need to have 2000 Mpc/h added to their z coordinate.

Particles are output from every time step (recall that these simulations use global time steps for each particle). In each step, we linearly interpolate to find the time when the light cone intersects this each particle, and then linearly update the position and velocity to this time.

Each time step generates a separate file, which includes the entire box, for each periodic copy.

We store only a subsample of particles, the union of the A and B subsets (so 10%). Positions are in the "RV" format; ID numbers and kernel density estimates are in the "PID" format.

The HealPix pixels are computed using $+z$ as the North Pole, i.e., the usual (x,y,z) coordinate system. We choose Nside=16384 and store the resulting pixel numbers as int32. We output HealPix from all particles. Particle pixel numbers from each slab in the box are sorted prior to output; this permits better compression (down to 1/3 byte per particle!).

For the huge boxes, the light cone is simply one copy of the box, centered at (0,0,0). This provides a full-sky light cone to the the half-distance of the box (about 4 Gpc/h), and further toward the eight corners.

CHAPTER

EIGHT

COMPASO HALO FINDER

All group finding in AbacusSummit is done on the fly. We are using a hybrid FoF-SO algorithm, dubbed CompaSO, summarized as follows.

First, we compute a kernel density estimate around all particles. This uses a weighting $(1 - r^2/b^2)$, where b is 0.4 of the interparticle spacing. We note that the effective volume of this kernel is equivalent to a top-hat of $0.737b$, so 85 kpc/*h* comoving, and that the mean weighted counts at an overdensity δ is about $\delta/10$ with a variance of 4/7 of the mean.

Second, we segment the particle set into what we call L0 halos. This is done with the FOF algorithm with linking length 0.25 of the interparticle spacing, but only for particles with Delta>60. We note that b_FOF=0.25 normally would percolate at a noticeably lower density, Delta~41 (More et al. 2011, ApJS, 195, 4). The intention is that the bounds of the L0 halo set be set by the kernel density estimate, which has lower variance than the nearest neighbor method of FOF and imposes a physical smoothing scale.

Note: In Abacus, L0 groups are large, "fluffy" sets of particles that typically encompass several L1 groups. L1 groups correspond to classical "halos". L2 groups correspond to "halo cores" or perhaps "subhalos". Only L1 halos are in the catalog.

We stress that all L1/L2 finding and all halo statistics are based solely on the particles in the L0 halo.

Third, within each L0 halo, we construct L1 halos by a competitive spherical overdensity algorithm. We begin by selecting the particle with the highest kernel density; this is a nucleus. We then search outward to find the innermost radius in which the enclosed density (of L0 particles only!) dips below 200. Particles interior to that radius are tentatively assigned to that group. Particles interior to 80% of R200 are marked as ineligible to be a future nucleus. We then search the remaining eligible particles to find the one with the highest remaining kernel density that also meets the criteria of being a density maximum. This latter condition is that a particle must be denser than all other particles (eligible or not) within 0.4 of the interparticle spacing. Once located, if this particle has a high enough density (a condition set by estimating what density would be generated by a singular isothermal sphere of M200=35 particles), we start another nucleus.

With each successive nucleus, we again search for the SO(200) radius, using all L0 particles. Now a particle is assigned to the new group if is previously unassigned *or* if it is estimated to have an enclosed density with respect to the new group that is twice that of the enclosed density with respect to its assigned group. In detail, these enclosed densities are not computed exactly, but rather scaled from the SO radius assuming a inverse square density profile.

The idea here is to allow nearby nuclei to divide particles by a principle similar to that of tidal radii. However, we stress that it is not our goal to find subhalos, and so we do not allow new nuclei to form inside 80% of the SO(200) radius of other L1 halos. That said, large substructure just inside this radius may yield a nucleus away from the center of the secondary object, and the subsequent enclosed density competition can result in some deblending of the two objects.

We do not perform any unbinding of particles, such as is sometimes done with estimates of the gravitational potential and resulting particle energy.

Fourth, within each L1 halo (and limited to only those particles), we repeat the SO algorithm to find L2 halos with an enclosed radius of 800. We store the masses of the 5 largest such subhalos, which may help to mark cases of over-merged L1 halos. But the main purpose is that we use the center of mass of the largest L2 subhalo to define a center for the output of the L1 statistics. We do not otherwise store information about the L2 subhalos.

All density thresholds are scaled upward as the cosmology departs from Einstein-deSitter, in keeping with spherical collapse estimates for low density universes. The FOF linking length is scaled as the inverse cube root of that change. The kernel density scale is not changed.

Note: Use the `SODensityL1` field in the header to get the exact density threshold at any epoch.

We output properties for all L1 halos with more than 35 particles.

CHAPTER

NINE

DATA ACCESS

9.1 DESI Members & Other NERSC Users

NERSC users, including DESI collaboration members, can access AbacusSummit data products at the following public path on NERSC:

`/global/project/projectdirs/desi/public/cosmosim/AbacusSummit`

For DESI members: more advice on getting set up to use AbacusSummit in a DESI NERSC software environment is given here: https://desi.lbl.gov/trac/wiki/CosmoSimsWG/Abacus#AbacusSummit

9.2 Public

We are pleased to be able to offer two online portals to access AbacusSummit data:

- the full 2 PB via OLCF's Constellation, tape-backed;
- a 750 TB subset via NERSC, disk-backed.

Constellation, as a tape-backed portal, is appropriate for bulk transfers between supercomputer centers. NERSC, as a disk-backed portal, and is appropriate for fetching narrow subsets of the data. The Constellation portal also contains the HighZ and ScaleFree simulations.

9.2.1 OLCF Constellation: Full Data on Tape

Oak Ridge Leadership Computing Facility's Constellation portal hosts the full 2 PB AbacusSummit data set on the magnetic-tape-backed High Performance Storage System (HPSS). HPSS offers high throughput, but high access latency. To amortize the latency, we aggregate the simulation files with coarse granularity, such that in most cases one must download many TB of data. For example, the halo catalogs for each simulation are in a single tarball (per simulation), which is 6.6 TB for a `base` simulation.

The primary DOI of AbacusSummit is `10.13139/OLCF/1811689`. This is a persistent identifer to the access information at the following URL, from where the AbacusSummit data may be browsed and downloaded via Globus: https://doi.ccs.ornl.gov/ui/doi/355

Note: Use the "Download" button at the top-right of https://doi.ccs.ornl.gov/ui/doi/355 to access the data

Note that it can take many hours before a transfer from Constellation begins if the tape drive is busy. Once it starts, though, the typical bandwidth is several GB/s.

The availability of Constellation depends on the status of HPSS, which undergoes regular downtime for maintenance. If the data is inaccessible, please check the status of HPSS on the following page: https://www.olcf.ornl.gov/for-users/center-status/

9.2.2 NERSC: Subset of Data on Disk

NERSC's Community File System (CFS) hosts a 750 TB subset of the most important AbacusSummit data products (includes most products except for the 7% "B" particle subsample and the 100% time slice outputs). The portal to this data is here: https://abacussummit-portal.nersc.gov/

Using that portal, you can select the desired subset of simulations, data products, and redshifts, and initiate the transfer via Globus. See Using Globus.

Some data products (initial conditions, merger trees) are not yet exposed via the web interface of this portal, but they can still be manually accessed by browsing the directory tree via Globus.

The availability of the NERSC portal depends on the availability of CFS and the DTNs (data transfer nodes). If the data is inaccessible, please check the CFS and DTN status on the following page: https://www.nersc.gov/live-status/motd/

9.2.3 Using Globus

Both the disk-backed and tape-backed portals use the Globus interface. See here for instructions on using Globus: https://docs.globus.org/how-to/get-started/

Note that most university and large computing centers have Globus endpoints already configured. But for transfers to other sites without pre-configured endpoints, such as a personal computer, one can use Globus Connect Personal.

9.3 What data are available?

The *Data Products* page documents the data products. All products are available at the Constellation portal (including ScaleFree and HighZ), and most products except for the 7% "B" particle subsample and the 100% time slice outputs are available at the NERSC portal.

Some data products (initial conditions, merger trees) are not yet exposed via the web interface of the NERSC portal, but they can still be manually accessed by browsing the directory tree via Globus.

Note that you will almost certainly need to use the utilities at https://abacusutils.readthedocs.io/ to unpack the outputs.

9.4 Acknowledgements

At OLCF, we are grateful to Ross Miller and the Constellation team for providing the opportunity to host this data and for their expert assistance during the creation of the DOI.

The NERSC hosting was made possible with the spport of Stephen Bailey, Benjamin Weaver, Eli Dart, Debbie Bard, and Lisa Gerhardt, who we thank warmly.

For additional acknowledgements related to the creation of the suite proper, please see authors:acknowledgements-thanks.

CHAPTER

TEN

DISK SPACE

For users interested in downloading AbacusSummit data, this breakdown of the file sizes of the various data products may be helpful.

The following is an annotated summary of `du -BMB` (so units of 1,000,000 bytes, as disk drives use). This is for AbacusSummit_base_c000_ph008, which is typical, supplemented by AbacusSummit_base_c000_ph001 for some information about the full time slices.

```
# The total outputs are 7.85 TB per simulation, if there are no full time slices.
7851475 AbacusSummit_base_c000_ph008

# When there are full time slices, they are about 3.5 TB *each*,
# with a mild trend in redshift, due to slight changes in compression efficiency.
# Sims have between 0 and 12 full time-slice epochs.
3578826   slices/z0.200
3534315   slices/z0.800
3498498   slices/z1.400
3472193   slices/z2.000
3451782   slices/z2.500
3431805   slices/z3.000

# Here's the breakdown; most of the data is in the halos.
# Most users won't need to look at the logs.
1       [top-level directory]
1      info
5649    log
1274121 lightcones
6571688 halos

# The lightcones contain 10% of the particles in rv+pid, and 100% in heal.
# rv: particle pos/vel
# pid: can be used to connect particles in the lightcone to halos in the timeslices;
#      also contains the kernel density estimate from the fully sampled density field.
# heal: The Nside=16384, ring=True healpix projection
# These are broken into files for each time step, typically d(ln(1+z)) ~ 0.002.
703830  lightcones/rv
329456  lightcones/heal
240836  lightcones/pid

# The halos come in two flavors: 12 primary time slices with more outputs,
# and then 21 secondary time slices with only limited outputs.
# Most users will only want the primary slices; the secondary ones
```

(continues on next page)

(continued from previous page)

```
# are intended for merger trees and for matching halos onto the light cones.
#
# The primary slices are bigger, typically 400 GB each:
412661  halos/z0.100
412863  halos/z0.200
412748  halos/z0.300
412340  halos/z0.400
411654  halos/z0.500
408158  halos/z0.800
402861  halos/z1.100
396109  halos/z1.400
388176  halos/z1.700
379538  halos/z2.000
364212  halos/z2.500
348863  halos/z3.000
# Total of primary slices: 4750 GB

# The secondary slices are smaller, typically 100 GB and decreasing at high z:
111493  halos/z0.150
110879  halos/z0.250
110041  halos/z0.350
109006  halos/z0.450
107437  halos/z0.575
106324  halos/z0.650
105147  halos/z0.725
102498  halos/z0.875
101105  halos/z0.950
99575   halos/z1.025
96330   halos/z1.175
94619   halos/z1.250
92831   halos/z1.325
89070   halos/z1.475
87182   halos/z1.550
85311   halos/z1.625
79098   halos/z1.850
67995   halos/z2.250
53962   halos/z2.750
11267   halos/z5.000
351    halos/z8.000
# Total of secondary slices: 1822 GB

# But one might opt only to keep certain types of files on disk, so here is the summary
# of the types.

# For example, a minimal installation might be only the halo_info and halo_rv_A files,
# which are 1.1 TB, and perhaps only for some of the primary epochs.  E.g., five
# epochs might knock the storage down to 0.5 TB per simulation.

# Another example would be to install only the primary epochs, without the
# field B samples.  This saves 1.83 TB per sim, so one is at 6 TB/sim.

# The halo info files contain the stats about all halos, typically 70-75 GB/epoch.
```

(continues on next page)

(continued from previous page)

```
# Note that the file format supports reading only subsets of columns; many users
# will need to load only a small fraction of these.
73188   halos/z0.100 halo_info
74301   halos/z0.200 halo_info
75190   halos/z0.300 halo_info
75864   halos/z0.400 halo_info
76323   halos/z0.500 halo_info
76416   halos/z0.800 halo_info
74722   halos/z1.100 halo_info
71488   halos/z1.400 halo_info
67014   halos/z1.700 halo_info
61615   halos/z2.000 halo_info
51419   halos/z2.500 halo_info
40797   halos/z3.000 halo_info
# Primary halo_info: 818 GB

73764   halos/z0.150 halo_info
74763   halos/z0.250 halo_info
75550   halos/z0.350 halo_info
76107   halos/z0.450 halo_info
76508   halos/z0.575 halo_info
76591   halos/z0.650 halo_info
76548   halos/z0.725 halo_info
76120   halos/z0.875 halo_info
75755   halos/z0.950 halo_info
75270   halos/z1.025 halo_info
74008   halos/z1.175 halo_info
73240   halos/z1.250 halo_info
72376   halos/z1.325 halo_info
70387   halos/z1.475 halo_info
69312   halos/z1.550 halo_info
68204   halos/z1.625 halo_info
64279   halos/z1.850 halo_info
56553   halos/z2.250 halo_info
45891   halos/z2.750 halo_info
10066   halos/z5.000 halo_info
303    halos/z8.000 halo_info
# Secondary halo_info: 1362 GB

# The particles associated to these halos, with 3% consistent subsample
# in A and 7% in B, all indexed out of the halo_info files.
# First, we have the positions and velocities.
# Users painting HOD satellite galaxies into the halos probably could just use the A set.
31415   halos/z0.100 halo_rv_A
30859   halos/z0.200 halo_rv_A
30213   halos/z0.300 halo_rv_A
29498   halos/z0.400 halo_rv_A
28733   halos/z0.500 halo_rv_A
26264   halos/z0.800 halo_rv_A
23699   halos/z1.100 halo_rv_A
21216   halos/z1.400 halo_rv_A
18767   halos/z1.700 halo_rv_A
```

(continues on next page)

(continued from previous page)

```
16457   halos/z2.000 halo_rv_A
12965   halos/z2.500 halo_rv_A
9965    halos/z3.000 halo_rv_A
# Primary 3% halo rv: 280 GB

70448   halos/z0.100 halo_rv_B
69199   halos/z0.200 halo_rv_B
67735   halos/z0.300 halo_rv_B
66123   halos/z0.400 halo_rv_B
64399   halos/z0.500 halo_rv_B
58825   halos/z0.800 halo_rv_B
53036   halos/z1.100 halo_rv_B
47321   halos/z1.400 halo_rv_B
41824   halos/z1.700 halo_rv_B
36626   halos/z2.000 halo_rv_B
28821   halos/z2.500 halo_rv_B
22132   halos/z3.000 halo_rv_B
# Primary 7% halo rv: 626 GB

# And then we have the PIDs, with kernel densities embedded.
# These are used build merger trees, but could also be used
# to track particular particles as part of galaxy assignment,
# e.g., to find the densest particle in a progenitor halo and
# use its late-time position.
13680   halos/z0.100 halo_pid_A
13325   halos/z0.200 halo_pid_A
12947   halos/z0.300 halo_pid_A
12548   halos/z0.400 halo_pid_A
12140   halos/z0.500 halo_pid_A
10870   halos/z0.800 halo_pid_A
9652    halos/z1.100 halo_pid_A
8493    halos/z1.400 halo_pid_A
7421    halos/z1.700 halo_pid_A
6442    halos/z2.000 halo_pid_A
4989    halos/z2.500 halo_pid_A
3777    halos/z3.000 halo_pid_A
# Primary 3% halo pid: 116 GB

30217   halos/z0.100 halo_pid_B
29390   halos/z0.200 halo_pid_B
28501   halos/z0.300 halo_pid_B
27584   halos/z0.400 halo_pid_B
26637   halos/z0.500 halo_pid_B
23780   halos/z0.800 halo_pid_B
21020   halos/z1.100 halo_pid_B
18439   halos/z1.400 halo_pid_B
16061   halos/z1.700 halo_pid_B
13899   halos/z2.000 halo_pid_B
10731   halos/z2.500 halo_pid_B
8097    halos/z3.000 halo_pid_B
# Primary 7% halo pid: 254 GB
```

(continues on next page)

(continued from previous page)

```
# For the secondary epochs, we provide only the PID+density file.
# These are slightly smaller because they include only L1 particles, not L0 particles.
11726   halos/z0.150 halo_pid_A
11238   halos/z0.250 halo_pid_A
10743   halos/z0.350 halo_pid_A
10256   halos/z0.450 halo_pid_A
9655    halos/z0.575 halo_pid_A
9292    halos/z0.650 halo_pid_A
8940    halos/z0.725 halo_pid_A
8258    halos/z0.875 halo_pid_A
7941    halos/z0.950 halo_pid_A
7620    halos/z1.025 halo_pid_A
7007    halos/z1.175 halo_pid_A
6716    halos/z1.250 halo_pid_A
6429    halos/z1.325 halo_pid_A
5883    halos/z1.475 halo_pid_A
5628    halos/z1.550 halo_pid_A
5389    halos/z1.625 halo_pid_A
4680    halos/z1.850 halo_pid_A
3621    halos/z2.250 halo_pid_A
2563    halos/z2.750 halo_pid_A
395    halos/z5.000 halo_pid_A
16     halos/z8.000 halo_pid_A
# Secondary 3% halo pid: 144 GB

26003   halos/z0.150 halo_pid_B
24879   halos/z0.250 halo_pid_B
23749   halos/z0.350 halo_pid_B
22643   halos/z0.450 halo_pid_B
21274   halos/z0.575 halo_pid_B
20442   halos/z0.650 halo_pid_B
19659   halos/z0.725 halo_pid_B
18120   halos/z0.875 halo_pid_B
17410   halos/z0.950 halo_pid_B
16685   halos/z1.025 halo_pid_B
15316   halos/z1.175 halo_pid_B
14664   halos/z1.250 halo_pid_B
14026   halos/z1.325 halo_pid_B
12802   halos/z1.475 halo_pid_B
12243   halos/z1.550 halo_pid_B
11719   halos/z1.625 halo_pid_B
10139   halos/z1.850 halo_pid_B
7822    halos/z2.250 halo_pid_B
5509    halos/z2.750 halo_pid_B
806    halos/z5.000 halo_pid_B
34     halos/z8.000 halo_pid_B
# Secondary 7% halo pid: 316 GB

# We provide the rest of the density field, i.e., the complement of the halo set,
# in the subsamples.  These would be used in matter-field statistics or if one wanted
# to associate particles in the periphery of halos.  Or if one wanted to run a different
# group finder (admittedly on only 10% of the dynamical particles).
```

(continues on next page)

(continued from previous page)

```
43432   halos/z0.100 field_rv_A
44044   halos/z0.200 field_rv_A
44724   halos/z0.300 field_rv_A
45445   halos/z0.400 field_rv_A
46195   halos/z0.500 field_rv_A
48542   halos/z0.800 field_rv_A
50898   halos/z1.100 field_rv_A
53164   halos/z1.400 field_rv_A
55295   halos/z1.700 field_rv_A
57277   halos/z2.000 field_rv_A
60189   halos/z2.500 field_rv_A
62610   halos/z3.000 field_rv_A
# Primary 3% field rv: 612 GB

97497   halos/z0.100 field_rv_B
98866   halos/z0.200 field_rv_B
100381  halos/z0.300 field_rv_B
101991  halos/z0.400 field_rv_B
103666  halos/z0.500 field_rv_B
108916  halos/z0.800 field_rv_B
114209  halos/z1.100 field_rv_B
119330  halos/z1.400 field_rv_B
124160  halos/z1.700 field_rv_B
128634  halos/z2.000 field_rv_B
135233  halos/z2.500 field_rv_B
140699  halos/z3.000 field_rv_B
# Primary 7% field rv: 1374 GB

# Field PIDs are probably not used much, but these do relate particles across epochs
# and the PID encodes the initial grid location for Lagrangian displacements.
16508   halos/z0.100 field_pid_A
16556   halos/z0.200 field_pid_A
16627   halos/z0.300 field_pid_A
16709   halos/z0.400 field_pid_A
16800   halos/z0.500 field_pid_A
17124   halos/z0.800 field_pid_A
17468   halos/z1.100 field_pid_A
17809   halos/z1.400 field_pid_A
18128   halos/z1.700 field_pid_A
18443   halos/z2.000 field_pid_A
18833   halos/z2.500 field_pid_A
19093   halos/z3.000 field_pid_A
# Primary 3% field pid: 210 GB

36280   halos/z0.100 field_pid_B
36327   halos/z0.200 field_pid_B
36434   halos/z0.300 field_pid_B
36582   halos/z0.400 field_pid_B
36765   halos/z0.500 field_pid_B
37426   halos/z0.800 field_pid_B
38160   halos/z1.100 field_pid_B
38855   halos/z1.400 field_pid_B
```

(continues on next page)

(continued from previous page)

```
39510   halos/z1.700 field_pid_B
40150   halos/z2.000 field_pid_B
41037   halos/z2.500 field_pid_B
41698   halos/z3.000 field_pid_B
# Primary 7% field pid: 459 GB

# For the full time slices, they are split into L0 and field (non-L0) sets.
# However, this was just due to convenience in the code; the L0 particles
# are not indexed in halo_info.  Only concatenations will be useful.
# The fractional split of L0 to field increases to low redshift.
#
# The position+velocity data is in the pack9 format, which gives
# somewhat higher precision than RVint.  These files average about
# 2.8 TB per epoch, which is 8.5 bytes per particle.
1682068   slices/z0.200/field_pack9
1083440   slices/z0.200/L0_pack9

1864771   slices/z0.800/field_pack9
925909    slices/z0.800/L0_pack9

2061023   slices/z1.400/field_pack9
748046    slices/z1.400/L0_pack9

2242314   slices/z2.000/field_pack9
581440    slices/z2.000/L0_pack9

2371831   slices/z2.500/field_pack9
458272    slices/z2.500/L0_pack9

2480799   slices/z3.000/field_pack9
352132    slices/z3.000/L0_pack9

# The PID and kernel density estimate are in the pid files.
# These average about 0.7 TB per epoch, increasing toward low redshift.
510862    slices/z3.000/field_pack9_pid
88014     slices/z3.000/L0_pack9_pid

501868    slices/z2.500/field_pack9_pid
119812    slices/z2.500/L0_pack9_pid

489859    slices/z2.000/field_pack9_pid
158581    slices/z2.000/L0_pack9_pid

472529    slices/z1.400/field_pack9_pid
216902    slices/z1.400/L0_pack9_pid

454954    slices/z0.800/field_pack9_pid
288682    slices/z0.800/L0_pack9_pid

444629    slices/z0.200/field_pack9_pid
368691    slices/z0.200/L0_pack9_pid
```

CHAPTER

ELEVEN

VISUALIZATIONS

The following is a collection of visualizations produced from AbacusSummit data. These images may be used under the CC-BY-4.0 license, with attribution to "The AbacusSummit Team".

11.1 Summary Figures

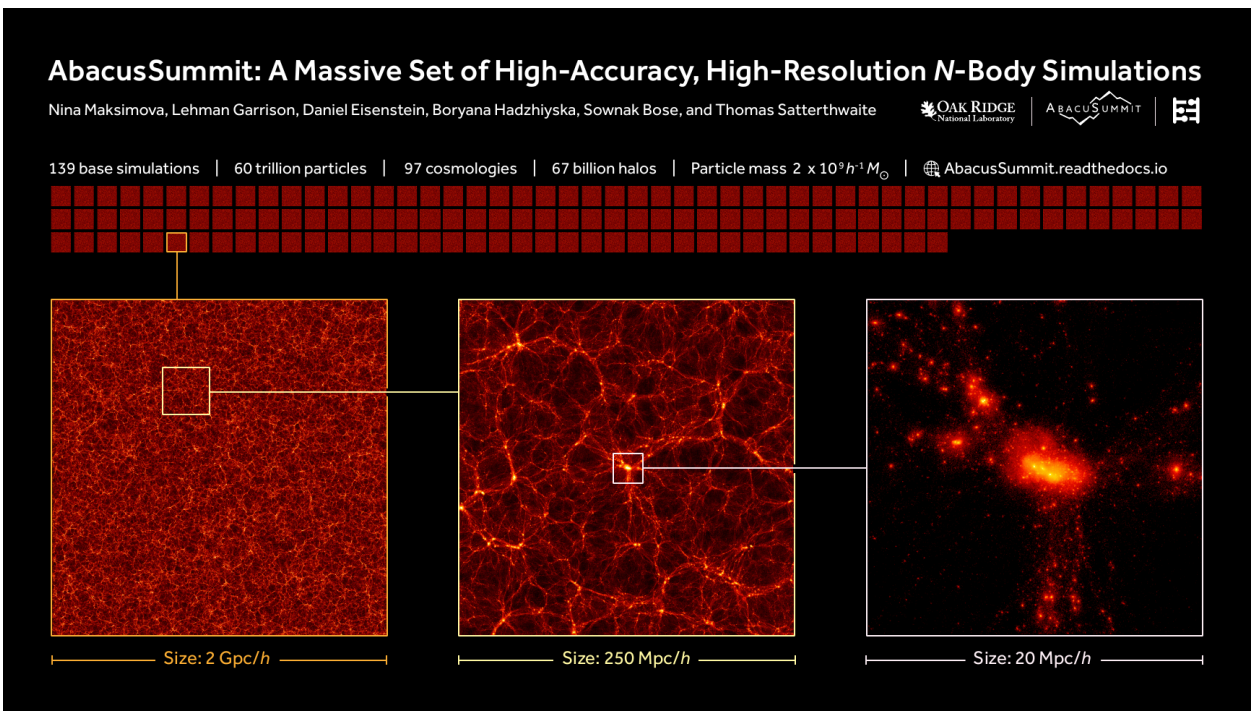


Fig. 1: A summary slide/poster. Download here: `72 dpi PNG (1.2 MB)`. A high-res PDF version of this image is available here: `PDF (7.7 MB)`. Layout and design credit: Lucy Reading-Ikkanda.

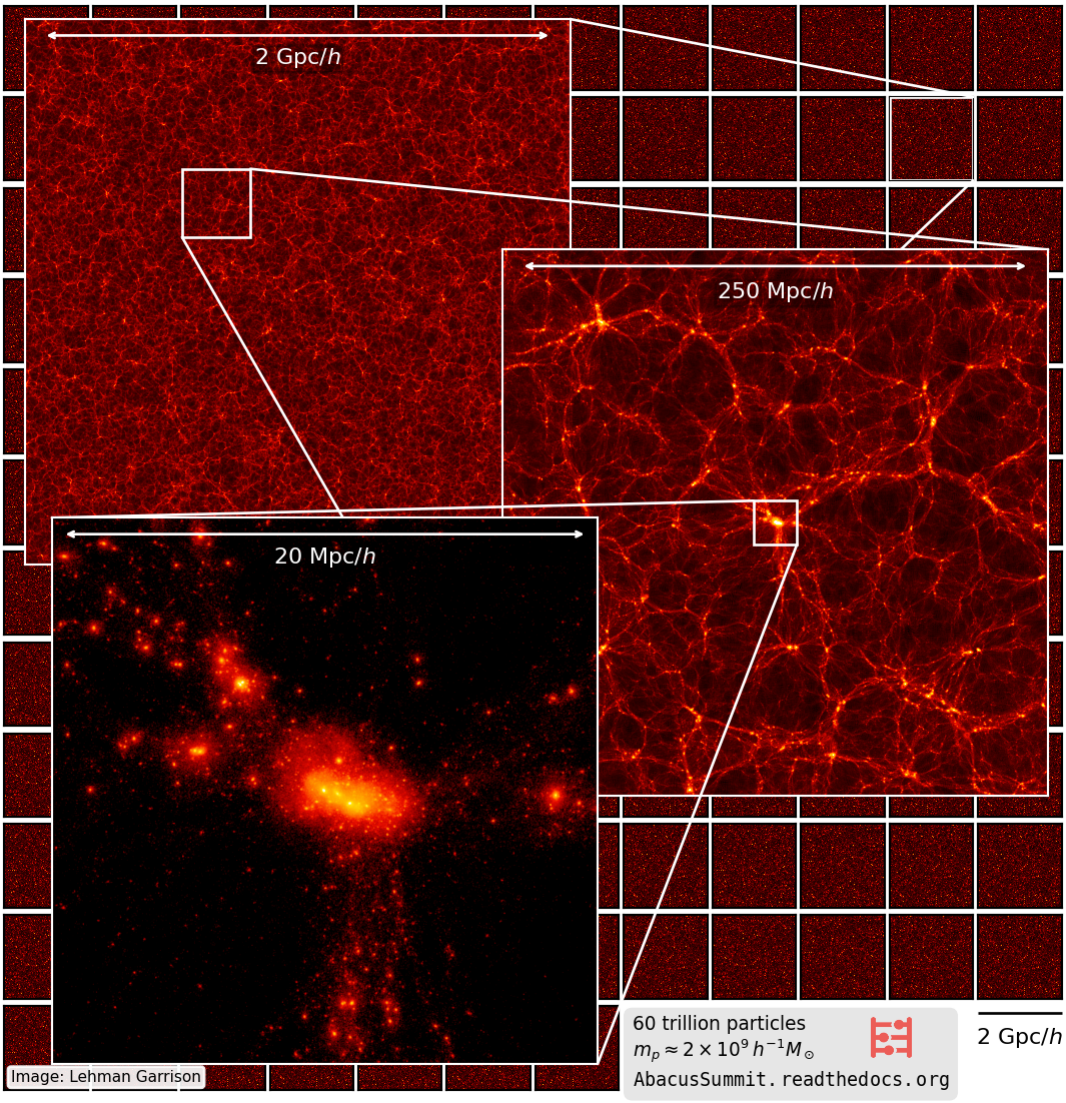


Fig. 2: A vertical summary slide/poster. Download here: `192 dpi PNG (1.4 MB)`. An older version with a different color scheme is available here: `192 dpi PNG (2 MB)`

11.2 Phase-sheet Evolution

11.3 Phase-matching test

11.4 Same halo, two cosmologies

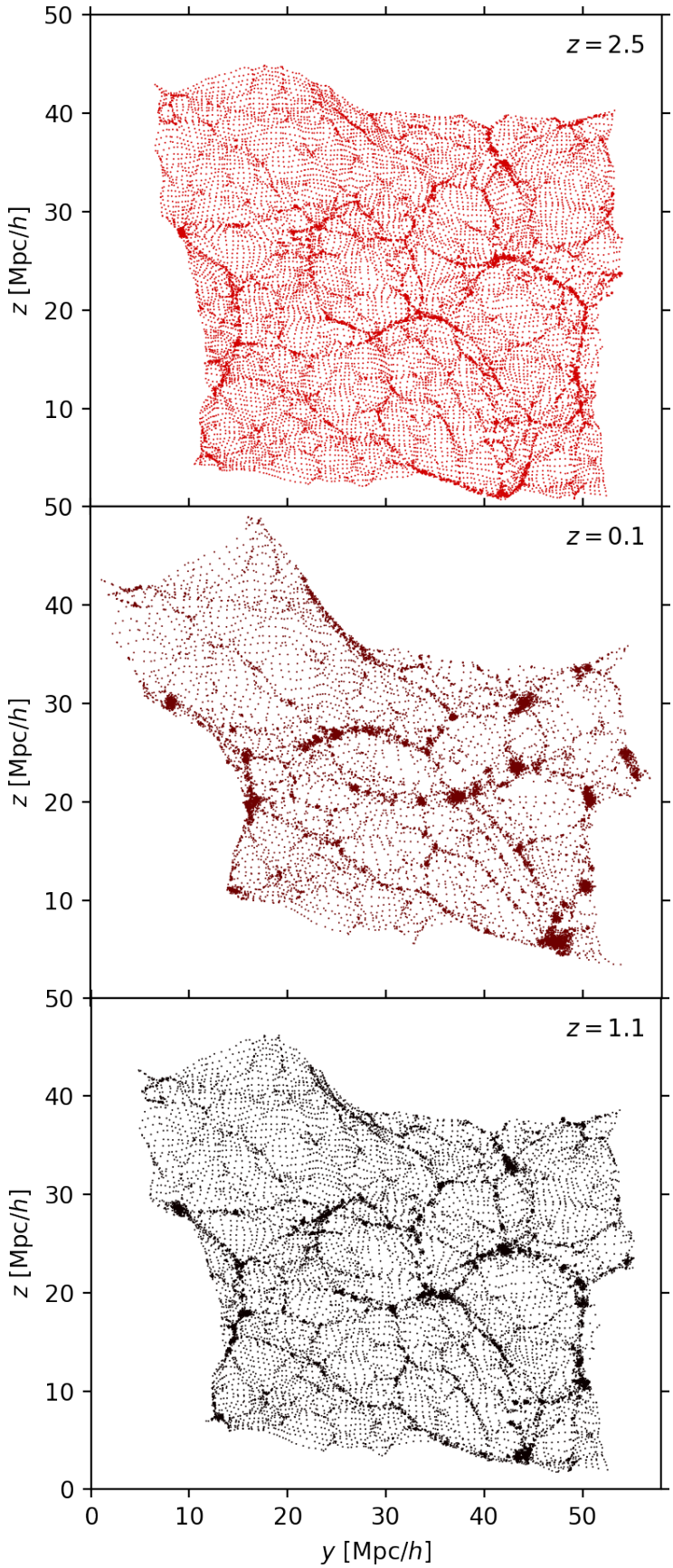
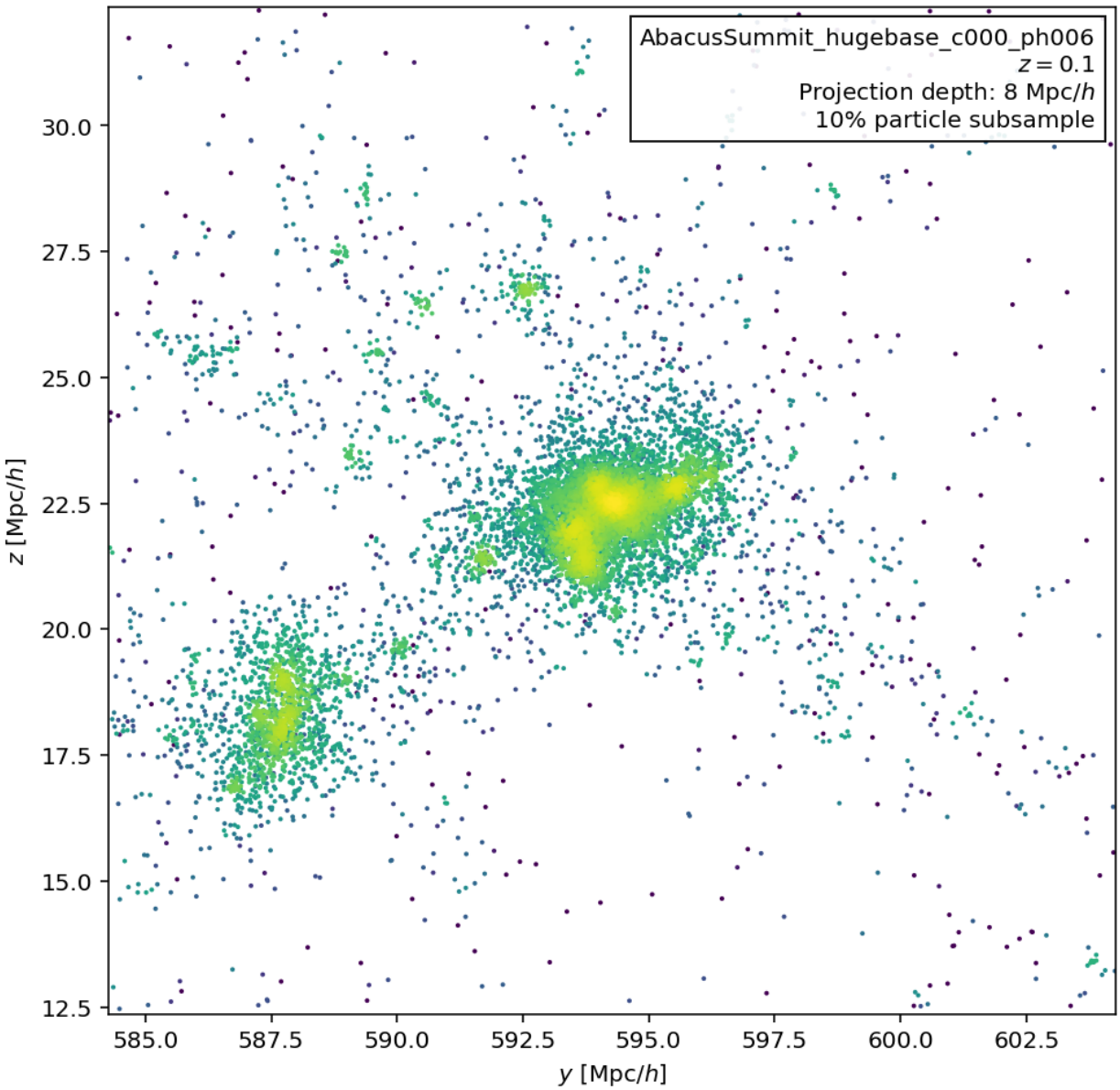


Fig. 3: A single cutout of a Lagrangian plane—a square from a particle plane selected in the initial conditions—at redshifts $z = 2.5$, $z = 1.1$, $z = 0.1$ from the highbase simulation. The "memory" of the initial lattice configuration persists to low redshift in low-density regions. Download here: `PNG (725 KB)`.


AbacusSummit_hugebase_c000_ph006
$z = 0.1$
Projection depth: 8 Mpc/h
10% particle subsample
30.0
27.5
25.0
22.5
20.0
17.5
15.0
12.5
z [Mpc/h]
585.0
587.5
590.0
592.5
595.0
597.5
600.0
602.5
y [Mpc/h]

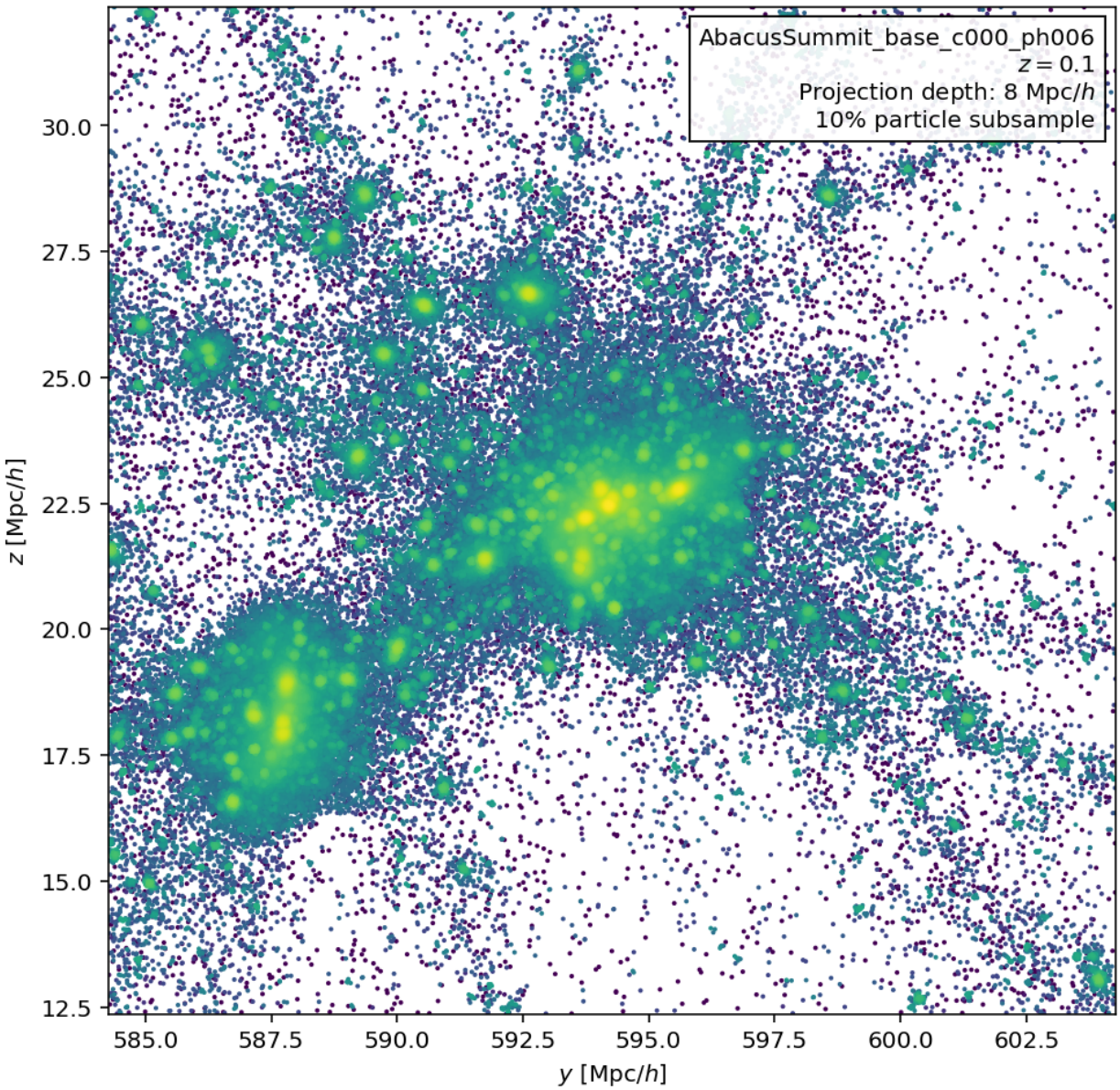
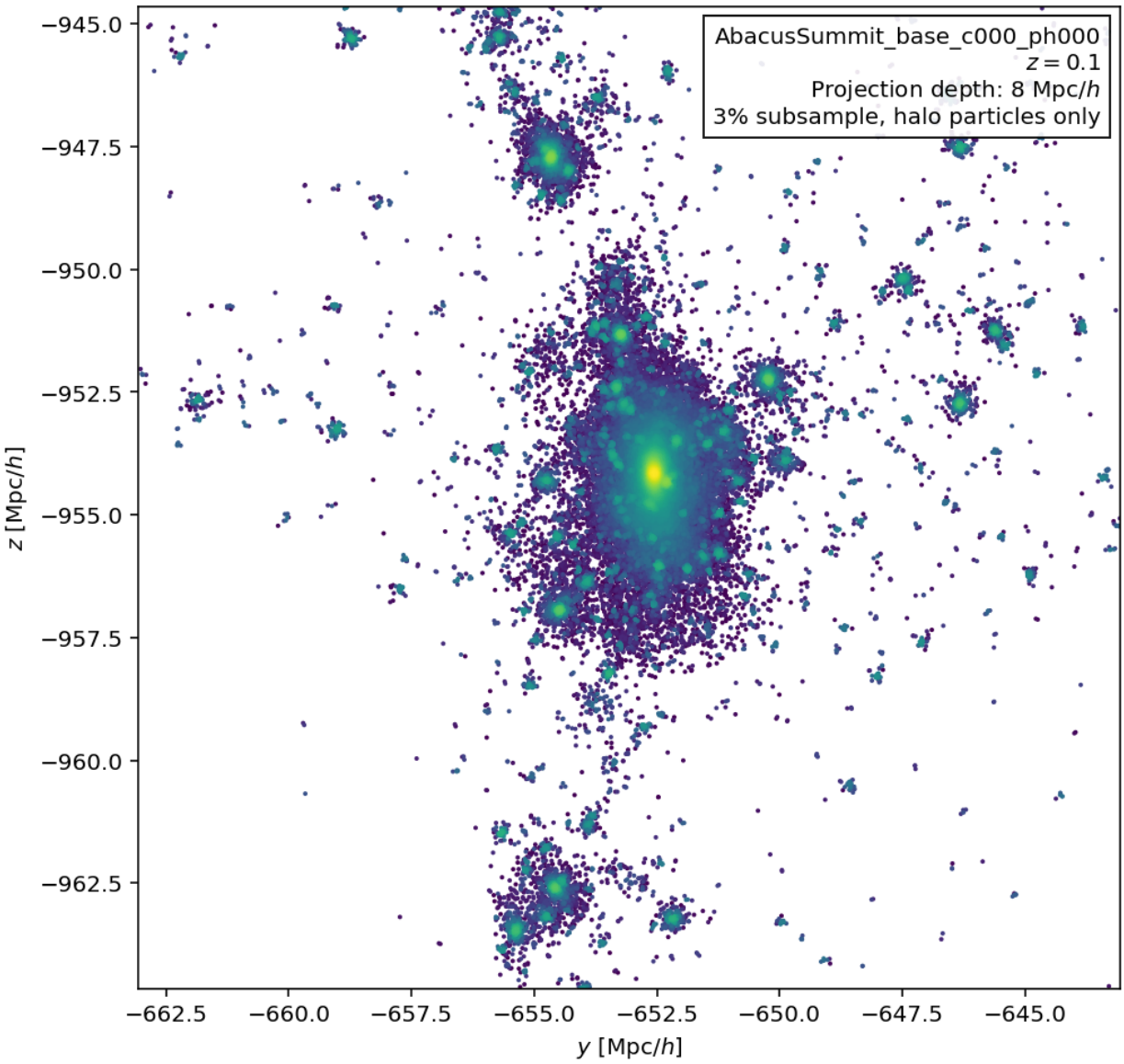


Fig. 4: A visualization of the same halo at two different mass resolutions. Try opening both of these images and blinking back and forth between them!


AbacusSummit_base_c000_ph000
$z = 0.1$
Projection depth: 8 Mpc/h
3% subsample, halo particles only
−945.0
−947.5
−950.0
−952.5
−955.0
−957.5
−960.0
−962.5
z [Mpc/h]
−662.5
−660.0
−657.5
−655.0
−652.5
−650.0
−647.5
−645.0
y [Mpc/h]

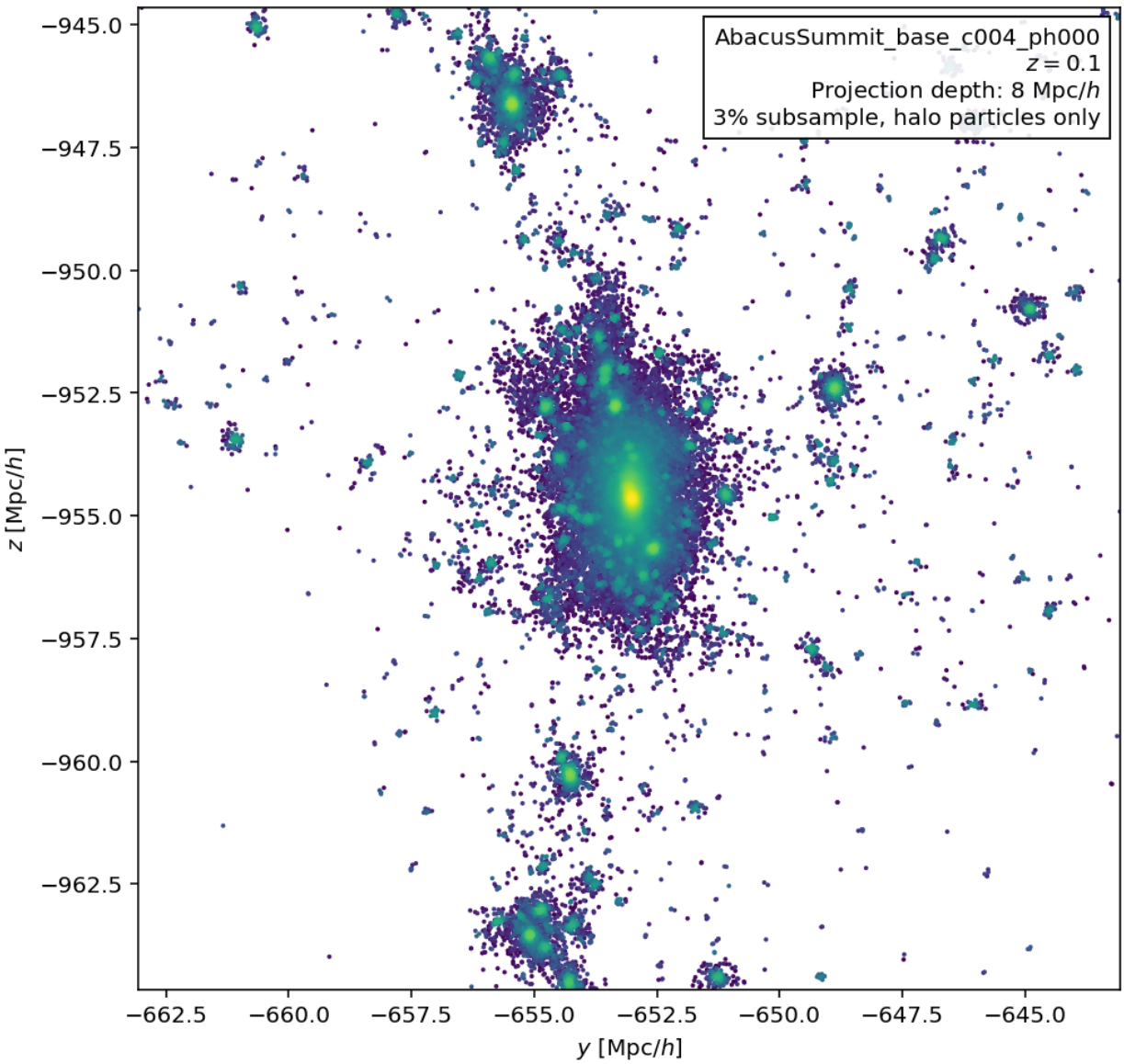


Fig. 5: A visualization of the same halo in c000 (the base cosmology) and c004 (low sigma8 cosmology). Try opening both of these images and blinking back and forth between them!

CHAPTER

TWELVE

AUTHORS & ACKNOWLEDGEMENTS

12.1 Authors

Abacus is actively developed by Lehman Garrison, Nina Maksimova, and Daniel Eisenstein, with contributions from Boryana Hadzhiyska and Sownak Bose and consulting from Philip Pinto. Abacus was initially developed by Marc Metchnik and Philip Pinto, with contributions from Daniel Eisenstein and later development led by Douglas Ferrer.

12.2 Contact

If you would like to get in touch via email, please email Lehman, Nina, and Daniel at lgarrison@flatironinstitute.org, nina.maksimova@cfa.harvard.edu, and deisenstein@cfa.harvard.edu. The GitHub Issues page is also a good way to get in touch.

12.3 Acknowledgements & Thanks

Abacus development has been supported by NSF AST-1313285 and more recently by DOE-SC0013718, as well as by Simons Foundation funds and Harvard University startup funds. NM was supported as a NSF Graduate Research Fellow. The AbacusCosmos simulations were run on the El Gato supercomputer at the University of Arizona, supported by grant 1228509 from the NSF; the AbacusSummit simulations have been supported by OLCF projects AST135 and AST145, the latter through the Department of Energy ALCC program.

This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725, and resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231.

We would like to thank the OLCF and NERSC support teams for their expert assistance throughout this project.

We would also like to thank Stephen Bailey, Chia-Hsun Chuang, Shaun Cole, Pablo Fosalba, Salman Habib, Katrin Heitmann, Core Francisco Park, Joachim Stadel, Risa Wechsler, and Sihan Yuan for useful conversations about the AbacusSummit program.

12.4 Citation

Please see the *Papers & Citation* page for information on how to cite AbacusSummit.

CHAPTER

THIRTEEN

INDICES AND TABLES

- genindex
- modindex
- search