CAMELS-SAM

Release 0.1

Perez, L.A.

Sep 14, 2023

CONTENTS

CAMELS-SAM is the newest and third 'hump' of the CAMELS project: https://camels.readthedocs.io/en/latest/index.html.

Like CAMELS, CAMELS-SAM is made up of more than 1,000 simulation products and galaxy catalogs for machine learning, covering a broad parameter space of cosmology and astrophysics. Unlike CAMELS, CAMELS-SAM span a periodic volume of $(100\ h^{-1}\mathrm{Mpc})^3$ over 100 N-body only snaphots, and uses the Santa Cruz semianalytic model (SC-SAM) of galaxy formation and evolution.

See the *Description of Simulations* section for complete details about the suite, including the process for *Data Access*. There are many various simulation *Data Products* available, as well as a guide for *Using the SC-SAM catalogs*.

CHAPTER

ONE

CONTENTS

1.1 Description of Simulations

1.1.1 Overview of CAMELS-SAM

A complete description of the suite, its creation, and application can be found in Perez, Genel, et al. (2023, in press as of July 2023): https://arxiv.org/abs/2204.02408 .

The backbone of CAMELS-SAM are the 1,000+ N-body only `AREPO` simulations we generated across 100 snapshots between $z = 127$ to $z = 0$ across the broad cosmological parameter space of CAMELS. Each simulation was run through `ROCKSTAR` to generate halo catalogs at each snapshot, and then `ConsistentTrees` to generate merger trees. We feed this finely sampled merger tree history into the `Santa Cruz semi-analytic model` (SC-SAM, hence CAMELS-SAM) for galaxy formation. The parameters we vary across the suite are $\Omega_{\rm M}$, σ_8, $A_{\rm SN1}$, $A_{\rm SN2}$, and $A_{\rm AGN}$. The later 3 are pre-factors to three SC-SAM parameters controlling, respectively, the amplitude and rate of mass outflow from massive stars out of a galaxy, and the strength of the radio jet mode of AGN.

In this documentation, **simulation** indicates that there exists an N-body only simulation (i.e. snapshots) that generated the product; **simulation products** refers to the `ROCKSTAR` and `ConsistentTrees` products generated from the simulations; and **galaxy catalog** refers to the output halo and subhalo/galaxy catalogs that the Santa Cruz SAM generated. The raw simulations are enormously large (>450TB after compression), and are therefore not able to be publically released as the CAMELS snapshots are. Please contact the CAMELS team to discuss access to the CAMELS-SAM snapshots, and otherwise refer to the CAMELS data acess instructions for how to access everything else: https://camels.readthedocs.io/en/latest/data_access.html.

1.1.2 Simulation sets within the suite

Within CAMELS-SAM, these are these three sets of products publically released:

- **LH**. This set contains 1,000 simulation products and galaxy catalogs. Each simulation has a different value of the cosmological and astrophysical parameters, are arranged in a latin-hypercube. Each simulation has a different value of the initial random seed. This set represents the main CAMELS-SAM dataset. LH stands for Latin-Hypercube. These can be found within `camels/PUBLIC_RELEASE/SCSAM/LH_?` .
- **CV**. This set contains 5 simulation products and galaxy catalogs. All the simulations share the value of the cosmological and astrophysical parameters, and they only differ in the value of their initial random seed. This set is typically used to study the effect of cosmic variance due to the simulated volume size. CV stands for Cosmic Variance. These can be found within `camels/PUBLIC_RELEASE/SCSAM/CV_?` .
- **1P**. This set contains 12 galaxy catalogs. In this set, the SC-SAM was run at the smallest and largest value of each SAM parameter for two of the CV simulations $(0, 1)$. The value of the random seed is the same in all galaxy catalogs generated atop the same CV simulation. This set can be used to study the change induced by the astrophysical SC-SAM parameters in a given quantity. 1P stands for 1-parameter at a time. (We emphasize

this differs from the CAMELS 1P set, which more finely samples the parameter space and also focus on the cosmological parameters.) These can be found within These can be found within `camels/PUBLIC_RELEASE/SCSAM/CV_0` and `CV_1`, in folders named e.g. `Asn1x0p25-sc-sam` .

1.1.3 Characteristics

The CAMELS-SAM simulations follow the evolution of 640^3 dark matter particles within a periodic comoving volume of $(100\ h^{-1}\mathrm{cMpc})^3$ from $z = 127$ down to $z = 0$. All simulations share the value of these cosmological parameters:

Ω_b	h	n_s	w	M_ν	Ω_k
0.049	0.6711	0.9624	-1	0.0 eV	0.0

CAMELS-SAM has been designed to sample the parameter space of cosmology (by varying Ω_M and σ_8) and of the feedback astrophysics in the SC-SAM model (by varying A_SN1, A_SN2, and A_AGN). The physical meaning of these parameters is such:

Parameter	Meaning	Parameter coverage
Ω_m	Fraction of energy density in matter (dark matter + baryons)	[0.1,0.5] linearly, fid.=0.3
σ_8	Variance of the linear field on $8\ h^{-1}\mathrm{Mpc}$ at $z = 0$	[0.6,1.0] linearly, fid.=0.8
A_SN1	Multiplicative factor for the normalization of mass outflow rate of cold gas due to SN winds & SN feedback efficiency (Somerville et al. 2015, Eq. 2)	[0.25,4.0] logarithmically, fid.=1.0
A_SN2	Additive factor to the power-law slope of mass outflow rate; SN feedback slope (Somerville et al. 2015, Eq. 2)	[-2.0,2.0] linearly, fid.=1.0
A_AGN	Multiplicative factor for the n Normalization of 'radio mode' AGN feedback (Somerville et al. 2008, Eq. 20)	[0.25,4.0] logarithmically, fid.=1.0

We note that each simulation's file describing their parameter, `CosmoAstro_params.txt`, lists the for the astrophysical parameters the convolution with the base fiducial SC-SAM value. Each file will list: Ω_m , σ_8 , $A_\mathrm{SN1} \times 1.7$, $A_\mathrm{SN2} + 3$, $A_\mathrm{AGN} \times 0.002$, and a vestigial parameter set to 0.5. These are the parameters that went directly into creating each CAMELS-SAM N-body simulation and SC-SAM catalog. For those who are curious, this comes from how the latin hypercube was created: the minimum/lower and maximum/upper bounds are, respectively: $\Omega_\mathrm{m} = [0.1, 0.6]$, $\sigma_8 = [0.6, 1.0]$, $A_\mathrm{SN1} = [0.25, 4.0] \times 1.7$, $A_\mathrm{SN2} = [-2, 2] + 3$, $A_\mathrm{AGN} = [0.25, 4.0] \times 0.002$

1.1.4 Redshifts

The CAMELS-SAM simulations were generated over 100 snapshots between z=20 and z=0, following the same steps as `IllustrisTNG`:

Snapshot[#]	Redshift	Age [Gyr]	Lookback time [Gyr]
0	20.05	0.179	13.624
1	14.99	0.271	13.532
2	11.98	0.370	13.433
3	10.98	0.418	13.385

continues on next page

Table 1 – continued from previous page

Snapshot[#]	Redshift	Age [Gyr]	Lookback time [Gyr]
4	10.00	0.475	13.328
5	9.39	0.517	13.286
6	9.00	0.547	13.256
7	8.45	0.596	13.207
8	8.01	0.640	13.163
9	7.60	0.687	13.116
10	7.24	0.732	13.071
11	7.01	0.764	13.039
12	6.49	0.844	12.959
13	6.01	0.932	12.871
14	5.85	0.965	12.838
15	5.53	1.036	12.767
16	5.23	1.112	12.691
17	5.00	1.177	12.626
18	4.66	1.282	12.521
19	4.43	1.366	12.437
20	4.18	1.466	12.337
21	4.01	1.540	12.263
22	3.71	1.689	12.115
23	3.49	1.812	11.991
24	3.28	1.944	11.859
25	3.01	2.145	11.658
26	2.90	2.238	11.565
27	2.73	2.384	11.419
28	2.58	2.539	11.264
29	2.44	2.685	11.118
30	2.32	2.839	10.964
31	2.21	2.981	10.823
32	2.10	3.129	10.674
33	2.00	3.285	10.519
34	1.90	3.447	10.356
35	1.82	3.593	10.210
36	1.74	3.744	10.059
37	1.67	3.902	9.901
38	1.60	4.038	9.766
39	1.53	4.206	9.597
40	1.50	4.293	9.510
41	1.41	4.502	9.301
42	1.36	4.657	9.147
43	1.30	4.816	8.987
44	1.25	4.980	8.823
45	1.21	5.115	8.688
46	1.15	5.289	8.514
47	1.11	5.431	8.372
48	1.07	5.577	8.226
49	1.04	5.726	8.077
50	1.00	5.878	7.925
51	0.95	6.073	7.730
52	0.92	6.193	7.610
53	0.89	6.356	7.447

continues on next page

Table 1 – continued from previous page

Snapshot[#]	Redshift	Age [Gyr]	Lookback time [Gyr]
54	0.85	6.522	7.281
55	0.82	6.692	7.111
56	0.79	6.822	6.981
57	0.76	6.998	6.805
58	0.73	7.132	6.671
59	0.70	7.314	6.489
60	0.68	7.453	6.350
61	0.64	7.642	6.161
62	0.62	7.786	6.017
63	0.60	7.932	5.872
64	0.58	8.079	5.724
65	0.55	8.280	5.523
66	0.52	8.432	5.371
67	0.50	8.587	5.216
68	0.48	8.743	5.060
69	0.46	8.902	4.901
70	0.44	9.062	4.741
71	0.42	9.225	4.578
72	0.40	9.389	4.414
73	0.38	9.556	4.247
74	0.36	9.724	4.079
75	0.35	9.837	3.966
76	0.33	10.009	3.794
77	0.31	10.182	3.621
78	0.30	10.299	3.504
79	0.27	10.535	3.269
80	0.26	10.654	3.149
81	0.24	10.834	2.969
82	0.23	11.016	2.787
83	0.21	11.138	2.665
84	0.20	11.323	2.480
85	0.18	11.509	2.294
86	0.17	11.635	2.169
87	0.15	11.824	1.979
88	0.14	11.951	1.852
89	0.13	12.143	1.660
90	0.11	12.337	1.466
91	0.10	12.467	1.336
92	0.08	12.663	1.140
93	0.07	12.795	1.008
94	0.06	12.993	0.810
95	0.05	13.127	0.676
96	0.03	13.328	0.475
97	0.02	13.463	0.340
98	0.01	13.667	0.136
99	0.00	13.803	0.000

*ages and lookback time from IllustrisTNG300 documentation, and therefore their assumed cosmology .. raw:: html

</details>

1.2 Data Products

1.2.1 Halo catalogs and Merger Trees

The folder `Rockstar/CAMELS-SAM/LH_???/Rockstar` contains the CAMELS-SAM Rockstar halo catalogs for each simulation. The format of the catalogs is the default of the code, and we refer the user to the Rockstar documentation for further details. Due to storage limitations, we have kept just the *out_??.list* files and all `ROCKSTAR` configuration files that were used to run it. Additional files are stored on long-term magnetic tape; please contact the CAMELS team to discuss accessing them.

We also release the merger trees constructed through `ConsistentTrees`, *tree_?_?_?.dat' for each subvolume created in the simulation within ``Rockstar/CAMELS-SAM/LH_???/ConsistentTrees`*. We refer the reader to the consistent trees documentation for details on the format of the files. The SC-SAM can be rerun with new parameters upon these trees.

1.2.2 The Santa Cruz SAM for galaxy formation

We use the Santa Cruz semi-analytic model (SAM) for galaxy formation (Somerville et al. 2008, 2015, 2021) to robustly populate our large N-body simulations with galaxies. SAMs can be thought of an alternative to full hydrodynamic simulations, which apply simplified empirical recipes for physical processes of galaxy formation and evolution within dark matter halo 'merger trees' (e.g. from `ConsistentTrees` our of `ROCKSTAR`). Like numerical simulations, SAMs include free parameters that are calibrated to galaxy observations (e.g. Yung et al. 2019 for the local universe). The SC-SAM pipeline used in this work is nearly identical to that applied to IllustrisTNG300 in Gabrielpillai et al. (2021); see their work for complete justification and verification for the fiducial set up.

A note: the SC-SAM that was run for CAMELS-SAM did not store the averaged SFR properties for the galaxies, due to our particular needs and priorities, so all those will be equal to zero.

Field	Shape	Units	Description
GalpropBirthHaloID	N	–	ID of host halo from Haloprop
GalpropHaloIndex	N	–	Index of host halo in Haloprop (recommend usage: when loading whole subvolu
GalpropHaloIndex_Snapshot	N	–	Index of host halo in Haloprop (recommended usage: when loading snapshots)
GalpropMBH	N	10^9 M	Black hole mass
GalpropMH2	N	10^9 M	Molecular hydrogen cold gas mass
GalpropMHI	N	10^9 M	Neutral hydrogen cold gas mass
GalpropMHII	N	10^9 M	Singly ionized hydrogen cold gas mass
GalpropMaccdot	N	M / yr	accretion rate onto black hole
GalpropMaccdot_radio	N	M / yr	accretion rate onto black hole in radio mode
GalpropMbulge	N	10^9 M	Stellar mass of the bulge
GalpropMcold	N	10^9 M	Total cold gas mass
GalpropMvir	N	10^9 M	Total dark matter halo mass
GalpropMstar	N	10^9 M	Total stellar mass
GalpropMstar_merge	N	10^9 M	Stellar mass entering via mergers
GalpropMstrip	N	10^9 M	Stripped mass of DM sub-halo |
GalpropMu_merger	N	–	mass ratio of last merger (see S08)
GalpropOutflowRate_Mass	N	M / yr	rate of outflowing gas from ISM
GalpropOutflowRate_Metal	N	M / yr	rate of outflowing metals from ISM
GalpropPos	N, 3	cMpc	x, y, and z coordinates
GalpropRbulge	N	kpc	3D half-mass Radius of bulge
GalpropRdisk	N	kpc	3D half-mass Radius of disk
GalpropRedshift	N	–	redshift

continues on next p

Table 2 – continued from previous page

Field	Shape	Units	Description
GalpropRfric	N	Mpc	distance from halo center
GalpropRhalo	N	Mpc	halo virial radius
GalpropRootHaloID	N	–	ID of root halo at z = 0 from Haloprop
GalpropSatType	N	–	0 = central
GalpropSfr	N	M / yr	instantaneous SFR
GalpropSfrave100myr	N	M / yr	SFR averaged over 100 Myr
GalpropSfrave1gyr	N	M / yr	SFR averaged over 1 Gyr
GalpropSfrave20myr	N	M / yr	SFR averaged over 20 Myr
GalpropSigmaBulge	N	km / s	velocity dispersion of bulge
GalpropTmerger	N	Gyr	time since last merger
GalpropTmerger_major	N	Gyr	time since last major merger
GalpropTsat	N	Gyr	time since galaxy became a satellite in this halo
GalpropVdisk	N	km / s	rotation velocity of disk
GalpropVel	N, 3	km / s	x, y, and z components of velocity
GalpropZcold	N	Z * M	metal mass in cold gas
GalpropZstar	N	Z * M	metal mass in stars

Field	Shape	Units	Description
HalopropC_nfw	N	–	NFW concentration parameter for DM halo
HalopropHaloID	N	–	Halo ID given by Consistent-Trees
HalopropIndex	N	–	Index of halo in the file (recommended usage: when loading whole subvolumes)
HalopropIndex_Snapshot	N	–	Index of halo in file (recommended usage: when loading snapshots)
HalopropMaccdot_metal	N	M / Z / yr	accretion rate of metals into the halo
HalopropMaccdot_pristine	N	M / yr	accretion rate of pristine gas into the halo
HalopropMaccdot_radio	N	M / yr	accretion rate onto the BH in radio mode
HalopropMaccdot_reaccreate	N	M / yr	accretion rate of "recycled" gas
HalopropMaccdot_reaccreate_metal	N	M / Z / yr	accretion rate of "recycled" metals
HalopropMass_ejected	N	10^9 M	total gas mass in "ejected" reservoir
HalopropMcooldot	N	10^9 M / yr	rate of gas cooling/accretion from halo into ISM
HalopropMdot_eject	N	M / yr	rate of ejection of gas from halo
HalopropMdot_eject_metal	N	M / Z / yr	rate of ejection of metals from halo
HalopropMetal_eject	N	Z / yr	total mass of metals in "ejected" reservoir
HalopropMhot	N	10^9 M	mass of hot (CGM) gas in halo
HalopropMstar_diffuse	N	10^9 M	mass of stars in a diffuse stellar halo (from disrupted satellites)
HalopropMvir	N	10^9 M	halo virial mass
HalopropRedshift	N	–	redshift
HalopropRockstarHaloID	N	–	Halo ID from the Rockstar run
HalopropRootHaloID	N	–	Halo ID of the root halo for this merger tree
HalopropSnapNum	N	–	snapshot file number
HalopropSpin	N	N	spin of DM halo
HalopropSubfindID_DMO	N	–	Subfind index in TNG DMO simulation of bijective match
HalopropSubfindID_FP	N	–	Subfind index in TNG FP simulation of bijective match
HalopropZhot	N	10^9 M / Z	metal mass in hot halo (CGM)

1.2.3 Identifying each simulation's parameters

Within each of these data product directories and within each simulation's folder exists the file `CosmoAstro_params.txt`. This file includes the exact values of the cosmological and astrophysical parameters that created each CAMELS-SAM N-body simulation and halo/galaxy catalogs. There are six numbers listed, which are:

1. Ω_M
   The true value of Ω_M of the given simulation.
2. σ_8
   The true value of σ_8 of the given simulation.
3. $A_{\rm SN1} \times 1.7$ for SC-SAM
   The value given to the SC-SAM for the $\epsilon_{\rm SN0}$ free parameter, which is equal to the prefactor $A_{\rm SN1}$ *times* 1.7 (the best-fit fiducial value for this SC-SAM free amplitude parameter for local observations.) $A_{\rm SN1}$ was generated log-uniformly between 0.25 and 4.0.
4. $A_{\rm SN2} + 3.0$ for SC-SAM
   The value given to the SC-SAM for the $\alpha_{\rm rh}$ free parameter, which is equal to the prefactor $A_{\rm SN2}$ *plus* 3.0 (the best-fit fiducial value for this SC-SAM free power law slope for local observations.) $A_{\rm SN2}$ was generated uniformly between -2.0 and 2.0.
5. $A_{\rm AGN} \times 0.002$ for SC-SAM
   The value given to the SC-SAM for the $\kappa_{\rm radio}$ free parameter, which is equal to the prefactor $A_{\rm AGN}$ *times* 0.002 (the best-fit fiducial value for this SC-SAM free amplitude parameter for local observations.) $A_{\rm AGN}$ was generated log-uniformly between 0.25 and 4.0.
6. 0.005
   This is an unused parameter in CAMELS-SAM for $\epsilon_{\rm wind,QSO}$ from the SC-SAM. This was originally meant to be the analogous to $A_{\rm AGN2}$ in CAMELS, but the iteration of the SC-SAM used in this work showed little to no response to varying this parameter beyond its fiducial value.

1.3 Data Access

The available CAMELS-SAM data is stored in the Rusty cluster of the Flatiron Institute in New York City, and its data can be accessed through four different methods:

Note: We gently remind users that CAMELS-SAM is large, even as we have limited ourselves to the most broadly useful data products. The *smallest* data products in CAMELS-SAM are the SC-SAM galaxy catalogs, which are on average 5GB per simulation, totaling 7TB across the suite. All available CAMELS-SAM data products total just under 40TB.

1.3.1 Rusty

Users with an account on the Flatiron Institute Rusty cluster, can find all CAMELS-SAM data in `/mnt/ceph/users/camels/PUBLIC_RELEASE`, within the relevant folders as 'SCSAM' or 'CAMELS-SAM'. For example, the `ConsistentTrees` merger tree files can be found in `/mnt/ceph/users/camels/PUBLIC_RELEASE/Rockstar/CAMELS-SAM/LH_??/ConsistentTrees/tree_?_?_?.dat`.

1.3.2 Binder

Binder is a system that allows users to read and manipulate data that is hosted at the Flatiron Institute through either a Jupyter notebook or a unix shell. The user can find some basic documentation here. All CAMELS-SAM data can be accessed, read, and manipulated through Binder.

Warning: Two important things need to be taken into account when using Binder. First, the Binder environment is ephemeral - after a few days of inactivity its contents are deleted, so one has to be vigilant about downloading any analysis results in time. Second, Binder is not designed to carry out long and heavy calculations. In this case we recommend the user to download the data and work with it locally.

Link to Binder

1.3.3 Globus

The CAMELS-SAM data can be downloaded via globus, an online system designed to efficiently transfer large amounts of data. This is the method we recommend to transfer the data.

Globus link

1.3.4 url

We provide access to the CAMELS-SAM via a simple uniform resource locator (url). We do not recommend downloading large amounts of data through this system, as can be slow and unstable. However, for small or individual files it may be convenient.

URL link

Note: CAMELS-SAM is not currently configured for access and exploration through Flathub. When is it, we will update this page (expected March 2022). This page was adapted from the CAMELS documentation: https://camels.readthedocs.io/en/latest/data_access.html.

1.4 Using the SC-SAM catalogs

The Santa Cruz SAM galaxy and halo catalogs are stored as text files with comma-separated values, as *galprop_0-99.dat* and *haloprop_0-99.dat* within all the SC-SAM-generated subvolumes. These files contain information about the halo and galaxies from all snapshots of a given simulation.

We give here an example for how to open these files and pull out galaxy or halo data fulfilling certain conditions. This runs in `python3`, and most crucially requires `pandas`, `subprocess`, and `numpy`. As the most relevant example for the original creators, this is how L.A. Perez pulled out information at a single redshift:

```
def ProcessSAMdat_single_redshift(path_to_SAM, Nsubvols, sought_z, fieldswanted, gal_or_
↪halo):
    #NOTE: these ?_colnames are the columns, in order, of the data in the original .dat␣
↪files. You can give fieldswanted in nearly any order, but the order that the columns␣
↪will take in the final array will depend on the fields' order in these lists. NOTE: if␣
↪you care about halo_index, birthhaloID, or roothaloID, redshift will NOT be in the 0th␣
↪index; update the line below that assumes that!
```

(continues on next page)

(continued from previous page)

```
    g_colnames = ['halo_index', 'birthhaloid', 'roothaloid', 'redshift', 'sat_type',
                  'mhalo', 'm_strip', 'rhalo', 'mstar', 'mbulge', 'mstar_merge', 'v_disk
↪',
                  'sigma_bulge', 'r_disk', 'r_bulge', 'mcold', 'mHI', 'mH2', 'mHII',
↪'Metal_star',
                  "Metal_cold", 'sfr', 'sfrave20myr', 'sfrave100myr', 'sfrave1gyr',
                  'mass_outflow_rate', 'metal_outflow_rate', 'mBH', 'maccdot', 'maccdot_
↪radio',
                  'tmerge', 'tmajmerge', 'mu_merge', 't_sat', 'r_fric', 'x_position',
                  'y_position', 'z_position', 'vx', 'vy', 'vz']
    h_colnames = ['halo_index', 'halo_id', 'roothaloid', 'orig_halo_ID', 'redshift', 'm_
↪vir', 'c_nfw',
                  'spin', 'm_hot', 'mstar_diffuse', 'mass_ejected', 'mcooldot',
                  'maccdot_pristine', 'maccdot_reaccrete', 'maccdot_metal_reaccrete',
                  'maccdot_metal', 'mdot_eject', 'mdot_metal_eject', 'maccdot_radio',
                  'Metal_hot', 'Metal_ejected', 'snap_num']
    g_header_rows = []
    for i in range(0, len(g_colnames)):
        g_header_rows.append(i)
    h_header_rows = []
    for i in range(0, len(h_colnames)):
        h_header_rows.append(i)

    input_path=path_to_SAM

    if type(fieldswanted) == list:
        print(type(fieldswanted))
    else:
        return "Fieldswanted should be a list with the fields you want as strings!"

    All_halos=np.zeros((1,len(fieldswanted)))
    if gal_or_halo=="gal":
        checknums=0
        for x_i in np.arange(0,Nsubvols,1):
            for x_j in np.arange(0,Nsubvols,1):
                for x_k in np.arange(0,Nsubvols,1):
                    galprop = pd.read_csv('{}/{}_{}_{}/galprop_0-99.dat'.format(input_
↪path, x_i, x_j, x_k),
                                          delimiter=' ', skiprows=g_header_rows, names=g_
↪colnames)
                    # print('galprop read for ',x_i, x_j, x_k,' shape:', galprop.shape)
                    current_galprops=galprop[fieldswanted[:]].to_numpy()
                    print('For subvolume ',x_i,x_j,x_k, current_galprops.shape)
                    unique_redshifts=set(current_galprops[:,0]) #update this if redshift␣
↪will NOT be in the 0th column; see note above
                    unique_redshifts = np.array(sorted(unique_redshifts))
                    # print(unique_redshifts)
                    idx = (np.abs(unique_redshifts - sought_z)).argmin()
                    current_galprops_z=current_galprops[np.where(current_galprops[:,
↪0][:]==unique_redshifts[idx])[0],:]
                    # print(current_galprops_z.shape)
                    checknums=checknums+len(current_galprops_z)
```

(continues on next page)

(continued from previous page)

```
                    All_halos=np.concatenate((All_halos,current_galprops_z))
        print(All_halos.shape, checknums)
        return All_halos
    elif gal_or_halo=="halo":
        checknums2=0
        for x_i in np.arange(0,Nsubvols,1):
            for x_j in np.arange(0,Nsubvols,1):
                for x_k in np.arange(0,Nsubvols,1):
                    haloprop = pd.read_csv('{}/{}_{}_{}/haloprop_0-99.dat'.format(input_
↪path, x_i, x_j, x_k),
                                           delimiter=' ', skiprows=h_header_rows,␣
↪names=h_colnames)
                    current_haloprops=haloprop[fieldswanted[:]].to_numpy()
                    print('For subvolume ',x_i,x_j,x_k, current_haloprops.shape)
                    unique_redshifts=set(current_haloprops[:,0])
                    unique_redshifts = np.array(sorted(unique_redshifts))
                    idx = (np.abs(unique_redshifts - sought_z)).argmin()
                    current_haloprops_z=current_haloprops[np.where(current_haloprops[:,
↪0][:]==unique_redshifts[idx])[0],:]
                    # print(current_haloprops_z.shape)
                    checknums2=checknums2+len(current_haloprops_z)
                    All_halos=np.concatenate((All_halos,current_haloprops_z))
        print(All_halos.shape, checknums2)
        return All_halos[1:,:]
    else:
        print("gal_or_halo need to be a string, either 'gal' or 'halo', to get galprop␣
↪or haloprop respectively. Make sure the fields you want are actually reflected!")
        print("Column names of galprop file: ", g_colnames)
        print("Column names of haloprop file: ", h_colnames)
        return All_halos
```

Lines commented out are to confirm the files are being read correctly; *checknums* and *All_halos.shape* should be the same length, if all (sub)halos were correctly accessed at each redshift. If your fieldswanted includes 'halo_index', 'birthhaloid', or 'roothaloid', update "unique_redshifts=set(current_galprops[:,0])" to reflect that redshfit won't be the first column.

Due to know the `.dat` format is organized, one must specify exactly which properties should be collected as the `fieldswanted` string. See *Data Products* for the complete list of available properties for galaxies (*galprop*) and halos (*haloprop*). Additionally, the number of subvolumes is important, and corresponds to how the SC-SAM automatically splits up large volumes for processing (either 1 or 8 for CAMELS-SAM). For example, to access galaxy data at *z=0, 0.1, 0.5, 1.0* for CAMELS-SAM simulations LH0 through LH5:

```
import numpy as np
import pandas as pd
import os
import subprocess
import math

galprop_fields = ['redshift', 'sfr', 'mstar', 'mhalo', 'Metal_star', 'sat_type', 'x_
↪position', 'y_position', 'z_position']
haloprop_fields = ['redshift','snap_num','spin','m_vir']
redshifts=[0.0, 0.1, 0.5, 1.0]
```

(continues on next page)

(continued from previous page)

```
StartSim=0
EndSim=5

for i in np.arange(StartSim,EndSim,1):

    os.chdir('/mnt/ceph/users/camels/Sims/SCSAM/cLH'+str(i)) #!!!!!UPDATE LOCATIONS!!!!!!
↪!
    os.system('ls')
    totaldirs=(subprocess.check_output('''ls -l sc-sam/ | grep -c ^d''', shell=True,
↪text=True))
    totaldirs=np.float64(totaldirs)
    Nsubvol=np.int64(totaldirs**(1./3.))

    for ZS in np.arange(0,len(redshifts),1):
        Z=redshifts[ZS]
        CurrentSAMdat= ProcessSAMdat_single_redshift('/mnt/ceph/users/camels/Sims/SCSAM/
↪cLH'+str(i)+'/sc-sam', Nsubvol, Z, galprop_fields, 'gal')
        print('CAMELS-SAM simulation LH',i,' at redshift ', Z, ' has this many galaxies:
↪', CurrentSAMdat.shape)
        '''This output is a numpy array that can be manipulated in whatever way you like!␣
↪Columns will be galprop_fields as listed above, each row is a SAM galaxy at redshift Z␣
↪(or more exactly, whatever Nbody simulation redshift is closest to what you've␣
↪requested).'''
```

Here is a more generalized way to open the SAM files for a given set of fields, and not only at a single redshift. Many thanks to Phil Bull for writing it up and allowing me to share it with other users!

```
import glob
root_dir_PB=str('/mnt/ceph/users/camels/PUBLIC_RELEASE/SCSAM/LH_'+str(i)+'/sc-sam')
#For the LH_ suite, i here will go from 0 to 999; CV_0 to CV_4 also available. For the␣
↪1P set, use CV_0 or CV_1, and note the name of the sc-sam folder.
galprop_fields = ['redshift', 'sfr', 'mstar', 'mhalo', 'Metal_star', 'sat_type', 'x_
↪position', 'y_position', 'z_position']
haloprop_fields = ['redshift','snap_num','spin','m_vir']

def load_catalog(fields, root_dir, halos=False, max_rows_per_subvol=None, verbose=True):
  """
  Load a catalog of galaxies or halos from the data files.
  """
  # Galaxy/halo catalog column names
  g_cols = ['halo_index', 'birthhaloid', 'roothaloid', 'redshift', 'sat_type',
                'mhalo', 'm_strip', 'rhalo', 'mstar', 'mbulge', 'mstar_merge', 'v_disk',
                'sigma_bulge', 'r_disk', 'r_bulge', 'mcold', 'mHI', 'mH2', 'mHII',
↪'Metal_star',
                'Metal_cold', 'sfr', 'sfrave20myr', 'sfrave100myr', 'sfrave1gyr',
                'mass_outflow_rate', 'metal_outflow_rate', 'mBH', 'maccdot', 'maccdot_
↪radio',
                'tmerge', 'tmajmerge', 'mu_merge', 't_sat', 'r_fric', 'x_position',
                'y_position', 'z_position', 'vx', 'vy', 'vz']
  h_cols = ['halo_index', 'halo_id', 'roothaloid', 'orig_halo_ID', 'redshift', 'm_vir',
↪'c_nfw',
                'spin', 'm_hot', 'mstar_diffuse', 'mass_ejected', 'mcooldot',
```

(continues on next page)

(continued from previous page)

```
                'maccdot_pristine', 'maccdot_reaccrete', 'maccdot_metal_reaccrete',
                'maccdot_metal', 'mdot_eject', 'mdot_metal_eject', 'maccdot_radio',
                'Metal_hot', 'Metal_ejected', 'snap_num']

    # Select set of columns
    cols = h_cols if halos else g_cols
    data_file = "haloprop_0-99.dat" if halos else "galprop_0-99.dat"

    # Check that requested fields are valid
    bad_fields = []
    for f in fields:
        if f not in cols:
            bad_fields.append(f)
    if len(bad_fields) > 0:
        raise KeyError("Fields %s not found. Available fields: %s" % (bad_fields, cols))

    # Determine which column indices to keep
    use_cols = [cols.index(col) for col in cols if col in fields]

    # Count subvolumes
    subvol_dirs = glob.glob("%s/*_*_*" % root_dir)

    # Loop over sub-volumes and load data
    d = []
    for i, subvol_dir in enumerate(subvol_dirs):
        if verbose:
            print("Loading subvolume %d / %d" % (i+1, len(subvol_dirs)))
        _d = np.genfromtxt(os.path.join(subvol_dir, data_file),
                           comments='#',
                           usecols=use_cols,
                           max_rows=max_rows_per_subvol)
        d.append(_d)
        if verbose:
            print("    Loaded %d rows" % _d.shape[0])
    d = np.concatenate(d)
    return d
```

The original pipeline created by Gabrielpillai et al. (2021) included a final step to organize these data into smaller `.hdf5` files, but once all the modifications to the pipeline were made for CAMELS-SAM, this step was no longer feasible. Therefore, we share the raw `.dat` data format to guarantee all properties are accessible, even if at the cost of larger files that take longer to process.'

1.5 Contact

For questions about CAMELS-SAM, please reach out to original creator L.A. Perez (lucia.perez.phd 5 *at* gmail.com). For questions about accessing the data, also contact camel.simulations *at* gmail.com.

We also hope you will check out the paper introducing the CAMELS-SAM suite, Perez, Genel et al. (2022): https://arxiv.org/abs/2204.02408 For more details about the CAMELS project as a whole, please see: https://camels.readthedocs.io/ and Villaescusa-Navarro, Angles-Alcazar, Genel, et al. (2020), and the many publications the team has put out in https://camels.readthedocs.io/en/latest/publications.html.