# Voiager

***Release 1.0.0***

**Nico Hamaus**

**Feb 26, 2024**

# CONTENTS

**Voi**d dyn**a**mics and **g**eometry **e**xplore**r** (*Voiager*, or short Vger) provides a framework to perform cosmological analyses using voids identified in large-scale structure survey data. The code measures dynamic and geometric shape distortions in void stacks and propagates them to constraints on cosmological parameters using Bayesian inference. More detailed information on this method can be found under *Publications*.

---

**Note:** *Voiager* is an homage to NASA's Voyager program. The twin Voyager 1 and 2 spacecrafts are exploring where nothing from Earth has flown before. Continuing on their nearly 50-year journey since their 1977 launches, they each are much farther away from Earth and the sun than Pluto.

*"Voyager did things no one predicted, found scenes no one expected, and promises to outlive its inventors. Like a great painting or an abiding institution, it has acquired an existence of its own, a destiny beyond the grasp of its handlers."* — Stephen J. Pyne

---

**Note:** *Vger*, the abbreviation for *Voiager*, is an homage to Star Trek: the center of the enormous vessel contained the oldest part of V'ger – Voyager 6, an unmanned deep space probe launched by NASA in the late 20th century. The entire vessel surrounding the Voyager probe had been built by an unknown race of machine entities (possibly the Borg) in order to help it complete what the latter interpreted to be its primary programming: "learn all that is learnable", and return that knowledge to its creator. During its journey, the probe had come to think of itself as V'ger after the only remaining legible letters from its original name (the "O", "Y", "A", and "6" on the nameplate having been obscured from encounters with previous spatial hazards), and amassed knowledge to such a degree as to become self-aware.

*"On its journey back, it amassed so much knowledge, it achieved consciousness itself. It became a living thing."* — James T. Kirk, 2270s

---

CHAPTER

ONE

# FEATURES

The main functionality of *Voiager* includes the following steps:

- Reading of tracer and void catalogs (based on VIDE)
- Generation of random catalogs (based on healpy)
- Estimation of void-galaxy cross-correlation functions (based on scipy k-d trees)
- Projection and deprojection along the line of sight (based on PyAbel)
- Estimation of covariances (based on jackknife resampling)
- Models and likelihood functions (based on linear theory and Gaussian statistics)
- Maximization of likelihoods (based on scipy.optimize)
- Sampling of likelihoods (based on emcee)
- Plotting of summary statistics and results (based on matplotlib and GetDist)

CHAPTER

TWO

# INSTALLATION

The latest stable version on https://pypi.org is obtained via:

```
pip install Voiager
```

You may want to append the `--user` flag, or use a virtual environment, in case you are working on a machine without admin rights. The latest development version on github is obtained via:

```
git clone https://github.com/nhamaus/Voiager.git
cd Voiager
pip install .
```

**Note:** External package requirements are provided in the file requirements.txt. Follow the instructions provided on the VIDE wiki to install VIDE. For correctly rendering the plots you additionally need LaTeX. To install the necessary LaTeX packages on a linux machine, you can do

```
sudo apt-get install texlive texlive-latex-extra cm-super dvipng
```

CHAPTER

# THREE

# USAGE

The simplest way to get started is to run the executable `voiager` from a terminal within the Voiager/ package directory. This launches *Voiager* following launch.py with python (version 3). For further help on its usage, type:

```
voiager --help
```

*Voiager* is set up to perform an example run from the Beyond-2pt blind data challenge, which consists of a simulated mock-galaxy lightcone in a *w*CDM cosmology (C_mock_lightcone.h5). It is based on a public halo catalog within the AbacusSummit set of simulations and a halo occupation distribution (HOD) formalism (credits: Andres Salcedo, Yosuke Kobayashi, and Elisabeth Krause). The input data is retrieved from the catalogs/ folder including void catalogs produced with VIDE, while the output of the code is stored in the results/ directory.

**Note:** The example data in catalogs/ and the example results in results/ are not automatically downloaded upon installation, but they can be accessed via the repository. Once the data is publicly released, the catalogs/ folder can be downloaded as a compressed archive catalogs.tar.gz and unpacked within the main code directory via

```
tar -xvzf catalogs.tar.gz
```

The executable `voiager` will produce the following output:

```
Launching Voiager...
Loading data (angles, redshifts, catalog sizes, void properties)...
Loading info...
Loading voids...
Read 16897 voids
Loading macrocenters...
Loading derived void information...
Generating void randoms...
Transforming coordinates...
Building stacks...
=> Loading previous stack
Finding best fit...
=> Parameters:  ['f/b', 'qper', 'qpar', 'M', 'Q']
 [[0.45943433 1.01968878 1.02069274 1.17909194 0.91292996]
 [0.44439309 1.00533279 1.00509703 1.21756796 1.196998  ]]
 qper/qpar:
 [0.99901639 1.00023456]
 Reduced chi-square:
 [1.46139827 1.44428081]
MCMC sampling...
```

(continues on next page)

(continued from previous page)

```
=> Deleting previous chain
Constraining cosmology...
=> Deleting previous chain
Plotting...
Done.
```

**Note:** All the parameters of *Voiager* are configured in the file params.yaml, which is used as default configuration. You can also point to your customized parameter file in yaml format like this:

```
voiager 'path/to/parameter_file.yaml'
```

Previously calculated data vectors are saved in the file stacks.dat, this step in the pipeline will be skipped in case the file already exists (i.e., it needs to be removed to start a new calculation, but note that it requires about 32GB of available memory). Similarly, the files named chains.dat contain previously run MCMCs, which the code will continue sampling when found. Human readable ASCII versions of the chains are also produced.

The file params.yaml contains the main adjustable parameters of the code, each of which appears with a brief comment about its meaning:

```
Info: # Survey information
  survey:     'Beyond2pt' # Name of survey
  sample:     'C_mock_lightcone' # Name of tracer sample
  random:     'C_mock_lightcone_R10' # Name of random sample
  version:    '_0300' # Version (suffix) of void catalog
  redshift:   [0.8,1.3] # Redshift range
  sky:        3759.6159 # Sky area in square degrees (full sky ~ 41253)
  cosmology: "wCDM" # Cosmological model to constrain (current options: "LCDM", "wCDM",
↪"w0waCDM")


IO: # Input / output
  runExec:         True # If True, run executable when called
  basePath:        '' # Location of the top level code directory relative to current␣
↪working directory
  tracerPath:      'catalogs/tracers/' # Location of tracer catalogs relative to basePath
  voidPath:        'catalogs/voids/' # Location of void catalogs relative to basePath
  outPath:         'results/' # Location to store output files relative to basePath
  plotPath:        'plots/' # Location to store plots relative to outPath
  inputFormat:     'hdf5' # Filetype for input tracer and random catalogs (supported␣
↪types: https://docs.astropy.org/en/stable/io/unified.html)
  inputExtension: 'h5' # Filename extension for input tracer and random catalogs
  figFormat:       'pdf' # Format to save figures (e.g., pdf, png, jpg)
  columnNames:     ['RA','DEC','Z_red'] # Tracer and random catalog column headers for␣
↪right ascension, declination, redshift (angles in degrees)
  stackFile:       'stacks' # Filename for data of stacks
  chainFile:       'chains' # Filename for data of chains
  continueStack:   True # If True, continue using previous stacks. If False, delete old␣
↪stacks.
  continueChain:   False # If True, continue sampling of previous chains. If False,␣
↪delete old chains.
```

(continues on next page)

(continued from previous page)

```
Selection: # Void selection
  zv:  [0.8,1.3]   # Void redshift range
  rv:  [35.,1.e+9] # Void radius range
  mv:  [0.0,1.e+9] # Void mass range (number of tracers per void)
  dv:  [0.0,1.e+9] # Void core (minimum) density range
  Cv:  [0.0,1.e+9] # Void compensation range
  ev:  [0.0,1.e+9] # Void ellipticity range
  mgs: [2.0,1.e+9] # Void radius range in units of mean galaxy separation (mgs)


Bins: # Binning parameters
  vbin:      'zv' # 'zv': void-redshift bins, 'rv': void-radius bins
  binning:   'lin' # 'eqn': equal number of voids, 'lin': linearly spaced, 'log':␣
↪logarithmicly spaced. Alternatively, provide a list for custom bin edges
  Nvbin:     2 # Number of void bins
  Nrbin:     20 # Number of radial bins in correlation function
  Nrskip:    1 # Number of radial bins to skip in fit (starting from the first bin)
  rmax:      3. # Maximum radial distance in units of void radius
  ell:       [0,2,4] # Multipole orders to consider
  Nside:     128 # Mask resolution for generating random voids
  symLOS:    True # If True, assume void-centric symmetry along LOS (no odd multipoles).
  project2d: True # If True, the projected correlation function is calculated from the␣
↪POS vs. LOS 2d correlation function
  rescov:    False # If True, calculate covariance matrix for residuals between data and␣
↪model (experimental!)
  datavec:   '2d' # Define data vector, '1d': multipoles, '2d': POS vs. LOS 2d correlation␣
↪function


Computing: # Computing parameters
  Ncpu:    16 # Number of CPUs to use
  Nmock:   1 # Number of mock realizations (for observation = 1)
  Nbin_nz: 20 # Number of bins for redshift distributions
  Nbin_nv: 8 # Number of bins for void abundance function
  Nspline: 200 # Number of nodes for splines (only for visualization in plots, does not␣
↪affect fit)
  Nsmooth: 0.5 # Smoothing factor for splines (for no smoothing = 0)
  Nwalk:   16 # Number of MCMC walkers (ideally equal to Ncpu)
  Nchain:  1000 # Length of each MCMC chain
  Nburn:   3.0 # Initial burn-in steps of chain to discard, in units of auto-correlation␣
↪time
  Nthin:   0.5 # Thinning factor of chain, in units of auto-correlation time
  Nmarg:   4.0 # Margin size for parameter limits in plots, in units of standard␣
↪deviation


Model: # Model parameters with fiducial values and priors
  par:
    f/b:  0.5 # Redshift-space distortion parameter
    qper: 1.0 # Perpendicular Alcock-Paczynski parameter
    qpar: 1.0 # Parallel Alcock-Paczynski parameter
```

(continues on next page)

(continued from previous page)

```
    M:    1.0 # Monopole nuisance parameter
    Q:    1.0 # Quadrupole nuisance parameter

  prior:
    f/b:  [-10.,10.]
    qper: [-10.,10.]
    qpar: [-10.,10.]
    M:    [-10.,10.]
    Q:    [-10.,10.]


Cosmo: # Cosmological parameters with fiducial values and priors
  par_cosmo:
    Om: 0.30 # Omega_matter
    Ok: 0.00 # Curvature
    w0: -1.0 # DE eq. of state (constant)
    wa: 0.00 # DE eq. of state (slope)
    s8: 0.80 # RMS of density fluctuations of scale 8 Mpc/h
    h:  0.70 # Reduced Hubble constant
    b0: 1.20 # Linear tracer bias at z=0

  prior_cosmo: # Only provide for sampled parameters
    Om: [0.0,1.0]
    w0: [-2,-0.333]
    #wa: [-10,10]

  blind: True # If True, subtract mean of cosmology posterior
```

CHAPTER

# FOUR

# PLOTS

*Voiager* produces several figures to visualize the input data and the inferred results, these are all stored in the results/ folder.

## 4.1 Catalog properties

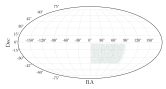

Fig. 1: The file named `void_sky` contains a Mollweide projection of the angular distribution of void centers on the sky, color-coded by redshift.

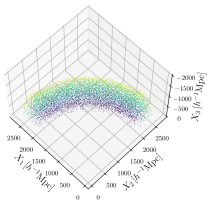

Fig. 2: In the file `void_box` the same void centers are shown in a Cartesian (comoving) coordinate system.

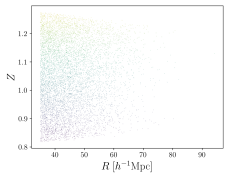


Fig. 3: `void_redshift` is a scatter plot of the effective radius vs. the redshift of each void in the catalog.

## 4.2 Void abundance

The most basic void features can be summarized via distributions. These are typically expressed as number densities per logarithmic interval of the void property. The file names start with `n_`, followed by one of the following properties: `compensation`, `core-density`, `density-contrast`, `effective-radius`, `ellipticity`, `redshift`, `richness`. Some examples are shown below:

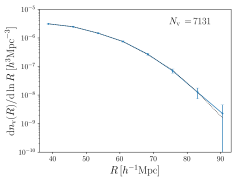


Fig. 4: Distribution of void `effective-radius` (void size function) and corresponding randoms (dotted).

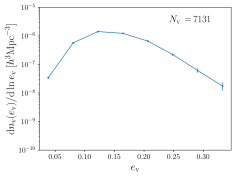


Fig. 5: Distribution of void `ellipticity` (void shape function).

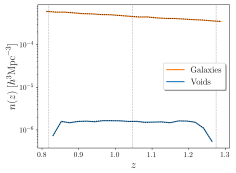


Fig. 6: Distribution of void (and galaxy) `redshift` (selection function) and corresponding randoms (dotted).

## 4.3 Correlation functions

Spatial correlations between void centers and tracers (galaxies) are provided in three variants: projected correlations on the sky named `xi_p`, multipoles of the 3D correlation function named `xi_ell`, and plane-of-sky vs. line-of-sight 2D correlation functions named `xi_2d`. In addition, the file `xi_ell=` provides only one specified multipole order, appended as an integer in the name. Finally, the file `xi_p_test` features a test for the projected and deprojected correlation function from a best-fit HSW profile as a template for the real data. All of these file names are appended by an additional integer that specifies the particular bin of the data (e.g., redshift or effective radius).

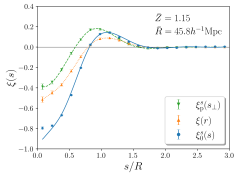


Fig. 7: `xi_p`: projected void-galaxy correlation function (downward triangles with model as dashed line), its deprojection (upward triangles with model as dotted line), and monopole in redshift space (circles with model as solid line).

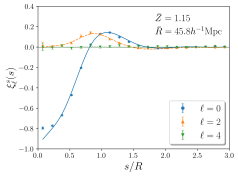


Fig. 8: `xi_ell`: monopole (circles with model as solid line), quadrupole (upward triangles with model as dashed line), and hexadecapole (downward triangles with model as dotted line) of void-galaxy correlation function.

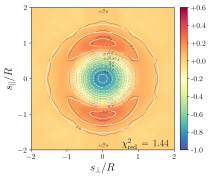


Fig. 9: `xi_2d`: 2D void-galaxy cross-correlation function along and perpendicular to the line of sight (black contours with color scale) with best-fit model (white contour lines).

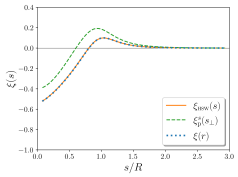


Fig. 10: `xi_p_test`: best-fit HSW profile to deprojected void-galaxy correlation function (solid line), its projection on the sky (dashed line), and subsequent deprojection based on the inverse Abel transform (dotted line).

## 4.4 Covariance

Covariance matrices for the void-galaxy correlation function are available, both for its multipoles, as well as for its 2D version with directions along and perpendicular to the line of sight.

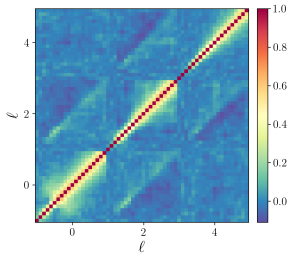


Fig. 11: `cov_ell`: covariance matrix for multipoles of the void-galaxy correlation function, normalized by its diagonal.

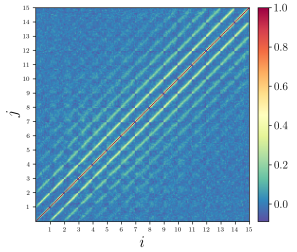


Fig. 12: `cov_2d`: covariance matrix for the 2D void-galaxy correlation function, normalized by its diagonal.

## 4.5 Parameter inference

The likelihood analysis of the void-galaxy correlation function provides constraints on dynamic and geometric distortions, which can be paramerized via the ratio of growth rate and bias $f/b$, and the *Alcock-Paczynski* parameter ratio $q_\perp/q_\parallel$, respectively. The files named `triangle` contain a corner plot of the posterior probability distribution (including nuisance parameters $\mathcal{M}$ and $\mathcal{Q}$) for each bin of the data vector. These parameters are proportional to the product of growth rate and rms fluctuation amplitude $f\sigma_8$, as well as the product of comoving angular diameter distance and Hubble rate $D_\mathrm{A}H$, which are shown in the file `fs8_DAH`. In turn, measurements of $D_\mathrm{A}H$ probe the expansion history of the Universe and can be used to infer some fundamental cosmological parameters. The files named `triangle`, appended by the particular assumed model (e.g., `LCDM`, `wCDM`, or `w0waCDM`) contain the posterior distribution of the constrained cosmological parameters of the specified model.

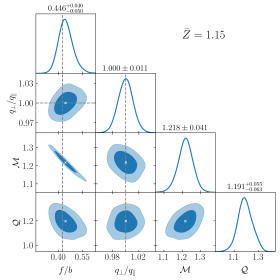


Fig. 13: `triangle`: corner plot of posterior distribution for model and nuisance parameters with its maximum (white cross). Shown are 68% and 95% confidence regions and the fiducial model as dashed lines.

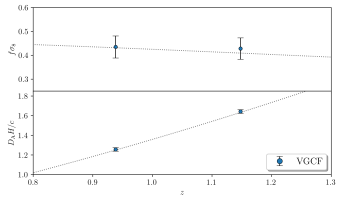


Fig. 14: `fs8_DAH`: measurements of $f\sigma_8$ and $D_\mathrm{A}H$ as a function of redshift (fiducial model as dotted line).

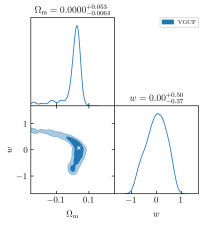


Fig. 15: `triangle_wCDM`: posterior of model parameters in a $w$CDM cosmology (blinded by mean) with its maximum (white cross). An extended banana-shaped degeneracy emerges, which leads to projection effects in the marginalized posteriors.

CHAPTER

FIVE

# CHECKS

*Voiager* should not be used as a black box! Consistency checks are essential in order to gain confidence in the stability of your results. Below is a non-exhaustive list of possible tests to identify systematic effects. The variables in params.yaml can be used to explore those.

- Variation of void selection cuts, especially minimum size, maximum ellipticity and core density
- Variation of binning schemes in redshift or scale (number of bins, min and max range, etc.)
- Consistency between different data vectors (multipoles vs. 2D correlation via variable `datavec`)
- Variation of prior ranges for model parameters
- Variation of fiducial parameter values

CHAPTER

SIX

# MODULES

VOId dynAmics and Geometry ExploreR provides a framework to perform cosmological analyses using voids identified in large-scale structure survey data. The code measures dynamic and geometric shape distortions in void stacks and propagates those to constraints on cosmological parameters using Bayesian inference.

**class** voiager.**Voiager**(*params*)

Bases: `object`

VOId dynAmics and Geometry ExploreR main class.

**Parameters**
**params** (*dict*) – dictionary of parameters from input file

**static parseParamsFile**()

Find and read parameter file from default or specified location

**Returns**
*parser (object)* – instance of ArgumentParser from argparse

voiager.**launch**(*vger*)

Launcher of Voiager.

**Parameters**
**vger** (*object*) – instance of Voiager class

## 6.1 datalib

voiager.datalib.**Chi2**(*par*, *prior*, *xit*, *xi*, *xiC*, *xiCI*, *ell=[0]*, *Nrskip=1*, *symLOS=True*, *Ndof=1*, *Nmock=1*)

Reduced chi-square.

**Parameters**

- **par** (*ndarray,Npar*) – fiducial model parameter values used as initial guess
- **prior** (*dict*) – boundary values for uniform parameter priors
- **xit** (*ndarray,**) – theory model for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xi** (*ndarray,**) – data for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xiC** (*ndarray,**) – covariance of void-tracer correlation function and its inverse
- **xiCI** (*ndarray,**) – covariance of void-tracer correlation function and its inverse
- **ell** (*int list*) – multipole orders to calculate (default = [0,])

- **Nrskip** (*int*) – Number of radial bins to skip in fit (starting from the first bin, default = 1)
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **Ndof** (*int*) – number of degrees of freedom (data points - free parameters)
- **Nmock** (*int*) – number of mock realizations if Nmock > 1 (default = 1)

**Returns**
*chi2 (float)* – reduced chi-squared value

voiager.datalib.**DA0**(*z*, *par_cosmo*)

Comoving angular diameter distance D_A(z) in units of Mpc/h in flat LCDM (fast from astropy).

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts
- **par_cosmo** (*dict*) – cosmological parameter values

**Returns**
*DA0 (ndarray,len(z))* – Comoving angular diameter distance D_A(z)

voiager.datalib.**HSW**(*r*, *r_s=1.0*, *d_c=-0.8*, *a=2.0*, *b=8.0*)

HSW void density profile (arXiv:1403.5499).

**Parameters**

- **r** (*ndarray,len(r)*) – radial distances from void center in units of void radius
- **r_s** (*float*) – scale radius in units of void radius (default = 1)
- **d_c** (*float*) – central underdensity (default = -0.8)
- **a** (*float*) – power-law index alpha (default = 2)
- **b** (*float*) – power-law index beta (default = 8)

**Returns**
*delta_HSW (ndarray,len(r))* – HSW void density profile

voiager.datalib.**Hz**(*z*, *par_cosmo*)

Hubble rate H(z) in units of km/s/Mpc.

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts
- **par_cosmo** (*dict*) – cosmological parameter values

**Returns**
*Hz (ndarray,len(z))* – Hubble rate H(z)

voiager.datalib.**Omz**(*z*, *par_cosmo*)

Omega_m(z) in flat LCDM.

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts
- **par_cosmo** (*dict*) – cosmological parameter values

**Returns**
*Omz (ndarray,len(z))* – Omega_m(z)

voiager.datalib.**bestFit**(*par*, *prior*, *par_cosmo*, *zvi*, *xit*, *xi*, *xiC*, *xiCI*, *ell=[0]*, *datavec='2d'*, *Nrskip=1*, *symLOS=True*, *Nmock=1*)

Find best fit of model to data.

**Parameters**

- **par** (*dict*) – model parameter values
- **prior** (*dict*) – boundary values for uniform parameter priors
- **par_cosmo** (*dict*) – cosmological parameter values
- **zvi** (*ndarray,Nvbin*) – average void redshift per bin
- **xit** (*ndarray,[Nvbin,*]*) – theory model for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xi** (*ndarray,[Nvbin,*]*) – data for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xiC** (*ndarray,[Nvbin,*]*) – covariance of void-tracer correlation function and its inverse
- **xiCI** (*ndarray,[Nvbin,*]*) – covariance of void-tracer correlation function and its inverse
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **datavec** (*str*) – Define data vector, '1d': multipoles, '2d': POS vs. LOS 2d correlation function (default)
- **Nrskip** (*int*) – Number of radial bins to skip in fit (starting from the first bin, default = 1)
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **Nmock** (*int*) – number of mock realizations if Nmock > 1 (default = 1)

**Returns**

*p0 (ndarray,[Nvbin,Npar])* – fiducial model parameter values used as initial guess

p1 (ndarray,[Nvbin,Npar]): best-fit model parameter values

chi2 (ndarray,Nvbin): reduced chi-squared values of best fit

voiager.datalib.**bz**(*z*, *par_cosmo*)

Linear bias b(z) of tracers, assuming simple inverse growth factor scaling.

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts
- **par_cosmo** (*dict*) – cosmological parameter values

**Returns**

*bz (ndarray,len(z))* – linear bias b(z)

voiager.datalib.**coordTrans**(*X*, *par_cosmo*, *Ncpu=1*)

Transformation from angles and redshifts to comoving coordinates.

**Parameters**

- **X** (*ndarray,[len(X),3]*) – RA, Dec, redshift
- **par_cosmo** (*dict*) – cosmological parameter values
- **Ncpu** (*int*) – number of CPUs for parallel calculation (default = 1 for serial)

**Returns**

*x (ndarray,[len(X),3])* – x1, x2, x3

voiager.datalib.**estimator**(*DDm*, *DRm*, *RDm*, *RRm*, *dim=1*, *rmax=3*)

Landy-Szalay estimator.

**Parameters**

- **DDm** (*ndarray,**) – stacked void-tracer correlations between data and randoms
- **DRm** (*ndarray,**) – stacked void-tracer correlations between data and randoms
- **RDm** (*ndarray,**) – stacked void-tracer correlations between data and randoms
- **RRm** (*ndarray,**) – stacked void-tracer correlations between data and randoms
- **dim** (*int*) – dimension of data vector [0: projected, 1: multipoles (default), 2: POS vs. LOS]
- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)

**Returns**

*xi (ndarray,)** – void-tracer correlation function [0: projected, 1: multipoles (default), 2: POS vs. LOS]

voiager.datalib.**f_b_z**(*z*, *par_cosmo*)

RSD parameter f(z)/b(z) of tracers.

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts
- **par_cosmo** (*dict*) – cosmological parameter values

**Returns**

*fz(z)/bz(z) (ndarray,len(z))* – RSD parameter f(z)/b(z)

voiager.datalib.**fz**(*z*, *par_cosmo*)

Linear growth rate f(z) in flat LCDM.

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts
- **par_cosmo** (*dict*) – cosmological parameter values

**Returns**

*fz (ndarray,len(z))* – Linear growth rate f(z)

voiager.datalib.**getBins**(*yv*, *binning='eqn'*, *Nbin=2*)

Define a binning scheme.

**Parameters**

- **yv** (*ndarray,len(yv)*) – void property to use for binning
- **binning** (*str / list*) – 'eqn' for equal number of voids (default), 'lin' for linear, 'log' for logarithmic. Alternatively, provide a list for custom bin edges.
- **Nbin** (*int*) – number of bins (default = 2)

**Returns**

*bins (ndarray,Nbin+1)* – bin edges

voiager.datalib.**getData**(*DDp*, *DRp*, *RDp*, *RRp*, *DD*, *DR*, *RD*, *RR*, *DD2d*, *DR2d*, *RD2d*, *RR2d*, *rv*, *rvr*, *zv*, *zvr*, *par*, *par_cosmo*, *vbin='zv'*, *binning='eqn'*, *Nvbin=2*, *Nrbin=20*, *rmax=3*, *ell=[0]*, *symLOS=True*, *project2d=True*, *rescov=False*, *Ncpu=1*)

Retrieve data vectors for two-point correlations and covariances.

**Parameters**

- **DDp** (*ndarray,[len(rv),Nrbin]*) – projected void density profiles between data and randoms
- **DRp** (*ndarray,[len(rv),Nrbin]*) – projected void density profiles between data and randoms
- **RDp** (*ndarray,[len(rv),Nrbin]*) – projected void density profiles between data and randoms
- **RRp** (*ndarray,[len(rv),Nrbin]*) – projected void density profiles between data and randoms
- **DD** (*ndarray,[len(rv),len(ell),Nrbin]*) – multipoles of void density profiles between data and randoms
- **DR** (*ndarray,[len(rv),len(ell),Nrbin]*) – multipoles of void density profiles between data and randoms
- **RD** (*ndarray,[len(rv),len(ell),Nrbin]*) – multipoles of void density profiles between data and randoms
- **RR** (*ndarray,[len(rv),len(ell),Nrbin]*) – multipoles of void density profiles between data and randoms
- **DD2d** (*ndarray,[len(rv),Nrbin,Nrbin]*) – POS vs. LOS void density profiles between data and randoms
- **DR2d** (*ndarray,[len(rv),Nrbin,Nrbin]*) – POS vs. LOS void density profiles between data and randoms
- **RD2d** (*ndarray,[len(rv),Nrbin,Nrbin]*) – POS vs. LOS void density profiles between data and randoms
- **RR2d** (*ndarray,[len(rv),Nrbin,Nrbin]*) – POS vs. LOS void density profiles between data and randoms
- **rv** (*ndarray,len(rv)*) – effective void radii
- **rvr** (*ndarray,len(rvr)*) – effective void radii randoms
- **zv** (*ndarray,len(zv)*) – void redshifts
- **zvr** (*ndarray,len(zv)*) – void redshifts randoms
- **par** (*dict*) – model parameter values
- **par_cosmo** (*dict*) – cosmological parameter values
- **vbin** (*str*) – binning strategy, 'zv': void-redshift bins (default), 'rv': void-radius bins
- **binning** (*str / list*) – 'eqn' for equal number of voids (default), 'lin' for linear, 'log' for logarithmic. Alternatively, provide a list for custom bin edges.
- **Nvbin** (*int*) – number of void bins (default = 2)
- **Nrbin** (*int*) – number of distance bins per dimension (default = 20)
- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)

- **project2d** (*bool*) – if True, use POS vs. LOS void density profiles to calculate projected void density profiles (default = True)
- **rescov** (*bool*) – if True, calculate covariance matrix for residuals between data and model (default = False, experimental!)
- **Ncpu** (*int*) – number of CPUs for parallel calculation (default = 1 for serial)

**Returns**

*Nvi (ndarray,Nvbin)* – number of voids per bin

rvi (ndarray,Nvbin): average effective void radius per bin

zvi (ndarray,Nvbin): average void redshift per bin

rmi (ndarray,[Nvbin,Nrbin]): radial distances from void center for each bin

rmi2d (ndarray,[Nvbin,(2*)Nrbin2d]): POS and LOS distances from void center for each bin

xip, xipE (ndarray,[Nvbin,Nrbin]): LOS projected void-tracer correlation function and its error

xi, xiE (ndarray,[Nvbin,len(ell),Nrbin]): multipoles of void-tracer correlation function and its error

xiC, xiCI (ndarray,[Nvbin,len(ell)*Nrbin,len(ell)*Nrbin]): covariance of multipoles and its inverse

xi2d (ndarray,[Nvbin,Nrbin2d,(2*)Nrbin2d]): POS vs. LOS 2d void-tracer correlation function

xi2dC, xi2dCI (ndarray,[Nvbin,(2*)Nrbin2d**2,(2*)Nrbin2d**2]): covariance of POS vs. LOS 2d void-tracer correlation function and its inverse

voiager.datalib.**getModel**(*rmi*, *rmi2d*, *rvi*, *xip*, *xipE*, *rmax=3*, *ell=[0]*, *Nsmooth=0.0*, *Nspline=200*, *weight=None*)

Retrieve theory model for two-point correlations.

**Parameters**

- **rmi** (*ndarray,[Nvbin,Nrbin]*) – radial distances from void center for each bin
- **rmi2d** (*ndarray,[Nvbin,(2*)Nrbin2d]*) – POS and LOS distances from void center for each bin
- **rvi** (*ndarray,Nvbin*) – average effective void radius per bin
- **xip** (*ndarray,[Nvbin,Nrbin]*) – LOS projected void-tracer correlation function and its error
- **xipE** (*ndarray,[Nvbin,Nrbin]*) – LOS projected void-tracer correlation function and its error
- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **Nsmooth** (*float*) – smoothing factor for spline of xip and xid in units of average variance, increase for more smoothing (default = 0 for no spline)
- **Nspline** (*int*) – number of nodes for spline if Nsmooth > 0 (default = 200)
- **weight** (*ndarray,[Nvbin,Nrbin]*) – weights for spline (default = None)

**Returns**

*rs (ndarray,Nspline)* – splined radial distances from void center in units of void effective radius (if Nsmooth > 0)

xips (ndarray,[Nvbin,Nspline]): spline of LOS projected void-tracer correlation function (if Nsmooth > 0)

xid (ndarray,[Nvbin,Nrbin]): deprojected void-tracer correlation function

xids (ndarray,[Nvbin,Nspline]): spline of deprojected void-tracer correlation function (if Nsmooth > 0)

Xid (ndarray,[Nvbin,Nrbin]): radially averaged deprojected void-tracer correlation function

Xids (ndarray,[Nvbin,Nspline]): spline of radially averaged deprojected void-tracer correlation function (if Nsmooth > 0)

xit (ndarray,[Nvbin,len(ell),Nrbin]): theory model for multipoles of void-tracer correlation function

xits (ndarray,[Nvbin,len(ell),Nspline]): spline of theory model for multipoles of void-tracer correlation function (if Nsmooth > 0)

xi2dt (ndarray,[Nvbin,Nrbin2d,(2*)Nrbin2d]): theory model for POS vs. LOS 2d void-tracer correlation function

xi2dts (ndarray,[Nvbin,10*Nspline,20*Nspline]): spline of theory model for POS vs. LOS 2d void-tracer correlation function (if Nsmooth > 0)

voiager.datalib.**getStack**(*xv*, *xg*, *rv*, *zv*, *Nv*, *Ng*, *ngz*, *zgm*, *wg=None*, *rmax=3*, *Nbin=20*, *ell=[0]*, *symLOS=True*, *dim=1*, *Nmock=1*, *Ncpu=1*)

Stacked void density profiles from tracer (galaxy) distribution, as a function of void-centric distance in units of effective void radius.

**Parameters**

- **xv** (*ndarray,[len(xv),3]*) – comoving coordinates of void centers
- **xg** (*ndarray,[len(xg),3]*) – comoving coordinates of tracers (galaxies)
- **rv** (*ndarray,len(rv)*) – effective void radii
- **zv** (*ndarray,len(zv)*) – void redshifts
- **Nv** (*int list,len(Nv)*) – number of voids (in each mock catalog)
- **Ng** (*int list,len(Ng)*) – number of tracers (in each mock catalog)
- **ngz** (*ndarray,len(ngz)*) – number density of tracers as function of redshift
- **zgm** (*ndarray,len(zgm)*) – binned tracer redshifts for ngz
- **wg** (*ndarray,len(wg)*) – weights for tracers (default = None)
- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **Nbin** (*int*) – number of distance bins per dimension (default = 20)
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **dim** (*int*) – dimension of data vector [0: projected, 1: multipoles (default), 2: POS vs. LOS]
- **Nmock** (*int*) – number of mock catalogs (default = 1)

- **Ncpu** (*int*) – number of CPUs for parallel calculation (default = 1 for serial)

**Returns**

Void density profile, normalized by mean tracer density

n (ndarray,[len(xv),Nrbin]) if dim=0, projected along line-of-sight

n (ndarray,[len(xv),len(ell),Nrbin]) if dim=1, multipoles

n (ndarray,[len(xv),Nrbin,Nrbin]) if dim=2, POS vs. LOS

voiager.datalib.**jackknife**(*DD*, *DDm*, *rv*, *V*, *dim=1*, *Ncpu=1*)

Generate jackknife samples from all void density profiles.

**Parameters**

- **DD** (*ndarray,[len(rv),*]*) – void density profiles
- **DDm** (*ndarray,**) – stacked void density profile
- **rv** (*ndarray,len(rv)*) – effective void radii
- **V** (*ndarray,**) – shell volumes
- **dim** (*int*) – dimension of data vector [0: projected, 1: multipoles (default), 2: POS vs. LOS]
- **Ncpu** (*int*) – number of CPUs for parallel calculation (default = 1 for serial)

**Returns**

*DDj (ndarray,[len(rv),])** – jackknife samples of stacked void density profile

voiager.datalib.**jackknife1**(*DD*, *DDm*, *rv*, *V*, *dim=1*, *idv=0*)

Generate a single delete-one jackknife sample from all void density profiles.

**Parameters**

- **DD** (*ndarray,[len(rv),*]*) – void density profiles
- **DDm** (*ndarray,**) – stacked void density profile
- **rv** (*ndarray,len(rv)*) – effective void radii
- **V** (*ndarray,**) – shell volumes
- **dim** (*int*) – dimension of data vector [0: projected, 1: multipoles (default), 2: POS vs. LOS]
- **idv** (*int*) – void id (default = 0)

**Returns**

*DDj (ndarray,[len(rv),])** – jackknife sample of void density profile

voiager.datalib.**lnL**(*par*, *prior*, *xit*, *xi*, *xiC*, *xiCI*, *ell=[0]*, *Nrskip=1*, *symLOS=True*, *Nmock=1*)

Log likelihood.

**Parameters**

- **par** (*ndarray,Npar*) – model parameter values
- **prior** (*dict*) – boundary values for uniform parameter priors
- **xit** (*ndarray,**) – theory model for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xi** (*ndarray,**) – data for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xiC** (*ndarray,**) – covariance of void-tracer correlation function and its inverse

- **xiCI** (*ndarray,**) – covariance of void-tracer correlation function and its inverse
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **Nrskip** (*int*) – Number of radial bins to skip in fit (starting from the first bin, default = 1)
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **Nmock** (*int*) – number of mock realizations if Nmock > 1 (default = 1)

**Returns**
*lnL (float)* – Logarithm of the likelihood multiplied by the prior

voiager.datalib.**lnL_DAH**(*par_cosmo*, *prior_cosmo*, *z*, *DAH_fit*, *DAH_err*)

Log likelihood for D_A(z)*H(z)/c.

**Parameters**

- **par_cosmo** (*dict*) – cosmological parameter values
- **prior_cosmo** (*dict*) – boundary values for uniform cosmological parameter priors
- **z** (*ndarray,len(z)*) – redshifts
- **DAH_fit** (*ndarray,len(z)*) – measured values of D_A(z)*H(z)/c
- **DAH_err** (*ndarray,len(z)*) – measured errors of D_A(z)*H(z)/c

**Returns**
*lnL_DAH (float)* – Logarithm of the likelihood for D_A(z)*H(z)/c multiplied by the prior

voiager.datalib.**lnP**(*par*, *prior*)

Logarithm of uniform prior.

**Parameters**

- **par** (*dict*) – model parameter values
- **prior** (*dict*) – boundary values for uniform parameter priors

**Returns**
*lnP (float)* – Logarithm of the uniform prior

voiager.datalib.**loadData**(*vger*, *tracerPath*, *voidPath*, *survey*, *sample*, *random*, *inputFormat*, *inputExtension*, *version*, *columnNames*, *Nmock=1*, *mockid='_{0:04d}'*)

Load tracer and void catalogs (from observation or mocks).

**Parameters**

- **vger** (*object*) – instance of Voiager class
- **tracerPath** (*path*) – name of input folder for tracer file(s)
- **voidPath** (*path*) – name of input folder fot void file(s)
- **survey** (*str*) – name of survey
- **sample** (*str*) – name of tracer sample
- **random** (*str*) – name of random sample
- **inputFormat** (*str*) – file type for input tracer and random catalogs
- **inputExtension** (*str*) – filename extension for input tracer and random catalogs
- **version** (*str*) – name of void sample

- **columnNames** (*str list*) – names of column headers for [RA, DEC, Z] in tracer and random catalog (angles in degrees)
- **Nmock** (*int*) – number of mock realizations if Nmock > 1 (default = 1)
- **mockid** (*str*) – format string for mock id in file names (e.g. '_1234' as default)

**Returns**

*Ngc, Nvc (int list)* – number of objects in each tracer and void catalog

Xg, Xr, Xv (ndarray,[len(X),3]): RA, Dec, redshift of tracers, randoms, voids

rv (ndarray,sum(Nvc)): effective void radii

tv (ndarray,sum(Nvc)): tree levels in void hierarchy

dv (ndarray,sum(Nvc)): void core (minimum) densities in units of mean

cv (ndarray,sum(Nvc)): void density contrasts

mv (ndarray,sum(Nvc)): void richness (number of tracers)

vv (ndarray,sum(Nvc)): void volumes

Cv (ndarray,sum(Nvc)): void compensations (average void densities in units of mean)

ev (ndarray,sum(Nvc)): void ellipticities

eval (ndarray,sum(Nvc)): eigenvalues of void inertia tensor

evec (ndarray,sum(Nvc)): eigenvectors of void inertia tensor

mgs (ndarray,sum(Nvc)): mean tracer (galaxy) separation at void redshift in units of effective void radius

voiager.datalib.**loadMCMC**(*filename*, *Nburn*, *Nthin*, *Nmarg=4.0*, *Nvbin=2*, *vbin='zv'*, *outPath='results/'*)

Load previous MCMC run from file, remove burn-in and apply thinning.

**Parameters**

- **filename** (*str*) – name of input file for emcee chains
- **Nburn** (*float*) – initial burn-in steps of chain to discard, in units of auto-correlation time
- **Nthin** (*float*) – thinning factor of chain, in units of auto-correlation time
- **Nmarg** (*float*) – Margin size for parameter limits in plots, in units of standard deviation (default = 4)
- **Nvbin** (*int*) – number of void bins (default = 2)
- **vbin** (*str*) – binning strategy, 'zv': void-redshift bins (default), 'rv': void-radius bins
- **outPath** (*path*) – name of output path for chains (default = 'results/')

**Returns**

*samples (ndarray list,[Nvbin,Nchain,Npar])* – MCMC samples after thinning and burn-in removal

logP (ndarray list,[Nvbin,Nchain]): log likelihood values of chains

pMean (ndarray list,[Nvbin,Npar]): mean parameter values

pStd (ndarray list,[Nvbin,Npar]): standard deviation of parameters

pErr (ndarray list,[Nvbin,Npar]): relative errors of parameters (pStd/pMean)

pLim (ndarray list,[Nvbin,Npar,2]): limits for parameter margins around their mean value in plots (Nmarg*pStd on each side)

voiager.datalib.**loadMCMC_cosmo**(*filename*, *cosmology*, *Nburn*, *Nthin*, *Nmarg=4.0*, *blind=True*, *outPath='results/'*)

Load previous MCMC run for D_A(z)*H(z)/c from file, remove burn-in and apply thinning.

**Parameters**

- **filename** (*str*) – name of input file for emcee chains
- **cosmology** (*str*) – cosmological model to consider [either 'LCDM' (default), 'wCDM', or 'w0waCDM']
- **Nburn** (*float*) – initial burn-in steps of chain to discard, in units of auto-correlation time
- **Nthin** (*float*) – thinning factor of chain, in units of auto-correlation time
- **Nmarg** (*float*) – Margin size for parameter limits in plots, in units of standard deviation (default = 4)
- **blind** (*bool*) – If true, subtract mean from chains (default = True)
- **outPath** (*path*) – name of output path for chains (default = 'results/')

**Returns**

*samples (ndarray,[Nchain,Npar])* – MCMC chain after thinning and burn-in removal

logP (ndarray,Nchain): log likelihood values of chain

pMean (ndarray,Npar): mean parameter values

pStd (ndarray,Npar): standard deviation of parameters

pErr (ndarray,Npar): relative errors of parameters (pStd/pMean)

pLim (ndarray,[Npar,2]): limits for parameter margins around their mean value in plots (Nmarg*pStd on each side)

voiager.datalib.**makeRandom**(*X*, *N=10.0*, *Nside=128*, *Nbin_nz=20*, *rv=None*, *seed=1*)

Produce unclustered randoms from a spatial distribution of objects (tracers or voids).

**Parameters**

- **X** (*ndarray,[len(X),3]*) – RA, Dec, redshift of input catalog
- **N** (*float*) – number of desired randoms per number of input objects (default = 10)
- **Nside** (*int*) – healpix resolution (default = 128)
- **Nbin_nz** (*int*) – number of bins for redshift distribution
- **rv** (*ndarray,len(rv)*) – if given, generates random void radii from an input void radius distribution (default = None)
- **seed** (*int*) – seed for generation of randoms (default = 1)

**Returns**

*Xr (ndarray,[N*len(X),3])* – RA, Dec, redshift of random catalog

rvr (ndarray,N*len(rv)): if rv is given, random void radii

voiager.datalib.**minChi2**(*par*, *prior*, *xit*, *xi*, *xiC*, *xiCI*, *ell=[0]*, *Nrskip=1*, *symLOS=True*, *Ndof=1*, *Nmock=1*)

Minimize the reduced chi-square.

**Parameters**

- **par** (*ndarray,Npar*) – fiducial model parameter values used as initial guess
- **prior** (*dict*) – boundary values for uniform parameter priors
- **xit** (*ndarray,[Nvbin,*]*) – theory model for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xi** (*ndarray,[Nvbin,*]*) – data for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xiC** (*ndarray,[Nvbin,*]*) – covariance of void-tracer correlation function and its inverse
- **xiCI** (*ndarray,[Nvbin,*]*) – covariance of void-tracer correlation function and its inverse
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **Nrskip** (*int*) – Number of radial bins to skip in fit (starting from the first bin, default = 1)
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **Ndof** (*int*) – number of degrees of freedom (data points - free parameters)
- **Nmock** (*int*) – number of mock realizations if Nmock > 1 (default = 1)

**Returns**

*fit.x (ndarray,Npar)* – best-fit model parameter values

fit.fun (float): reduced chi-squared value of best fit

voiager.datalib.**mockShift**(*x*, *N*, *Nmock=1*, *Lmock=100000.0*)

Coordinate shift of mock catalog. Needed to associate tracers with voids from the same mock.

**Parameters**

- **x** (*ndarray,[len(x),3]*) – comoving coordinates
- **N** (*int list,len(N)*) – number of objects in each mock catalog
- **Nmock** (*int*) – number of mock catalogs (default = 1)
- **Lmock** (*float*) – comoving distance between mock catalogs (along x1 axis)

**Returns**

*xs (ndarray,[len(x),3])* – shifted comoving coordinates

voiager.datalib.**numberDensity**(*z*, *Nbin*, *sky*, *par_cosmo*, *Nmock=1*)

Number density as function of redshift.

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts of all objects
- **Nbin** (*int*) – number of redshift bins
- **sky** (*float*) – sky area in square degrees
- **par_cosmo** (*dict*) – cosmological parameter values
- **Nmock** (*int*) – number of mock catalogs (default = 1)

**Returns**

*zm (ndarray,Nbin)* – mean redshift per bin (arithmetic mean of bin edges)

nm (ndarray,Nbin): mean number density

voiager.datalib.**profile**(*xv*, *xg*, *xvs*, *xgs*, *rv*, *ngz*, *wg*, *rmax*, *Nbin*, *ell*, *symLOS*, *dim*, *idv*)

Wrapper of profile1() with KDTree construction. Needed for parallel execution in getStack(), cannot be pickled inside function.

voiager.datalib.**profile1**(*xv*, *xg*, *xvs*, *xgs*, *xgTree*, *rv*, *ngz*, *wg=None*, *rmax=3*, *Nbin=20*, *ell=[0]*, *symLOS=True*, *dim=1*, *idv=0*)

Individual void density profile from tracer (galaxy) distribution, as a function of void-centric distance in units of effective void radius.

**Parameters**

- **xv** (*ndarray,[len(xv),3]*) – comoving coordinates of void centers
- **xg** (*ndarray,[len(xg),3]*) – comoving coordinates of tracers (galaxies)
- **xvs** (*ndarray,[len(xv),3]*) – shifted comoving coordinates of void centers if Nmock > 1
- **xgs** (*ndarray,[len(xg),3]*) – shifted comoving coordinates of tracers (galaxies) if Nmock > 1
- **xgTree** (*object*) – instance of KDTree class of tracer (galaxy) distribution
- **rv** (*ndarray,len(rv)*) – effective void radii
- **ngz** (*ndarray,len(ngz)*) – mean number density of tracers at redshift of void center
- **wg** (*ndarray,len(wg)*) – weights for tracers (default = None)
- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **Nbin** (*int*) – number of distance bins per dimension (default = 20)
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **dim** (*int*) – dimension of data vector [0: projected, 1: multipoles (default), 2: POS vs. LOS]
- **idv** (*int*) – void id (default = 0)

**Returns**

Void density profile, normalized by mean tracer density

n/ngz (ndarray,Nrbin) if dim==0, projected along line-of-sight

n/ngz (ndarray,[len(ell),Nrbin]) if dim==1, multipoles

n/ngz (ndarray,[Nrbin,Nrbin]) if dim==2, POS vs. LOS

voiager.datalib.**rho_c**(*z*, *par_cosmo*)

Critical density rho_c(z) in units of (Msol/h)/(Mpc/h)^3.

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts
- **par_cosmo** (*dict*) – cosmological parameter values

**Returns**

*rho_c (ndarray,len(z))* – critical density rho_c(z)

voiager.datalib.**runMCMC**(*p1*, *par*, *prior*, *xit*, *xi*, *xiC*, *xiCI*, *vbin='zv'*, *ell=[0]*, *datavec='2d'*, *Nrskip=1*, *symLOS=True*, *Nmock=1*, *Nwalk=1*, *Nchain=100*, *filename='chains.dat'*, *outPath='results/'*)

Monte Carlo Markov Chain sampler.

**Parameters**

- **p1** (*ndarray,[Nvbin,Npar]*) – best-fit model parameter values
- **par** (*dict*) – model parameter values
- **prior** (*dict*) – boundary values for uniform parameter priors
- **xit** (*ndarray,**) – theory model for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xi** (*ndarray,**) – data for void-tracer correlation function (either multipoles or POS vs. LOS)
- **xiC** (*ndarray,**) – covariance of void-tracer correlation function and its inverse
- **xiCI** (*ndarray,**) – covariance of void-tracer correlation function and its inverse
- **vbin** (*str*) – binning strategy, 'zv': void-redshift bins (default), 'rv': void-radius bins
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **datavec** (*str*) – Define data vector, '1d': multipoles, '2d': POS vs. LOS 2d correlation function (default)
- **Nrskip** (*int*) – Number of radial bins to skip in fit (starting from the first bin, default = 1)
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **Nmock** (*int*) – number of mock realizations if Nmock > 1 (default = 1)
- **Nwalk** (*int*) – number of MCMC walkers (default = 1)
- **Nchain** (*int*) – length of each MCMC chain (default = 100)
- **filename** (*str*) – name of output file for chains (default = 'chain.dat')
- **outPath** (*path*) – name of output path for chains (default = 'results/')

**Returns**

*sampler (object list,Nvbin)* – instance of EnsembleSampler class containing the chains for each void bin

voiager.datalib.**runMCMC_cosmo**(*z*, *par_cosmo*, *prior_cosmo*, *DAH_fit*, *DAH_err*, *Nwalk*, *Nchain*, *filename*, *cosmology='LCDM'*, *outPath='results/'*)

Monte Carlo Markov Chain sampler for D_A(z)*H(z)/c.

**Parameters**

- **z** (*ndarray,len(z)*) – redshifts
- **par_cosmo** (*dict*) – cosmological parameter values
- **prior_cosmo** (*dict*) – boundary values for uniform cosmological parameter priors
- **DAH_fit** (*ndarray,len(z)*) – measured values of D_A(z)*H(z)/c
- **DAH_err** (*ndarray,len(z)*) – measured errors of D_A(z)*H(z)/c

- **Nwalk** (*int*) – number of MCMC walkers
- **Nchain** (*int*) – length of each MCMC chain
- **filename** (*str*) – name of output file for chains
- **cosmology** (*str*) – cosmological model to consider [either 'LCDM' (default), 'wCDM', or 'w0waCDM']
- **outPath** (*path*) – name of output path for chains (default = 'results/')

**Returns**

*sampler (object)* – instance of EnsembleSampler class containing the chains for cosmological parameters

voiager.datalib.**voidAbundance**(*yv*, *Nbin*, *zmin*, *zmax*, *sky*, *par_cosmo*, *Nmock=1*)

Void abundance function.

**Parameters**

- **yv** (*ndarray,len(yv)*) – arbitrary void property (e.g., effective radius, redshift, core density, ellipticity)
- **Nbin** (*int*) – number of bins
- **zmin** (*float*) – minimum redshift
- **zmax** (*float*) – maximum redshift
- **sky** (*float*) – sky area in square degrees
- **par_cosmo** (*dict*) – cosmological parameter values
- **Nmock** (*int*) – number of mock catalogs (default = 1)

**Returns**

*ym (ndarray,Nbin)* – mean void property per bin (arithmetic mean of bin edges)

nm (ndarray,Nbin): mean number density of voids per logarithmic bin (dn/dlnx)

nE (ndarray,Nbin): error on mean space density of voids, assuming Poisson statistics

voiager.datalib.**xi_model**(*ell*, *rm*, *rd*, *d*, *D*, *f=0.5*, *qper=1.0*, *qpar=1.0*, *M=1.0*, *Q=1.0*)

Model for void-tracer correlation function (either POS vs. LOS or multipoles).

**Parameters**

- **ell** (*int list*) – multipole orders to calculate, POS vs. LOS 2d correlation if ell = None
- **rm** (*ndarray,len(rm)*) – radial distances from void center to calculate
- **rd** (*ndarray,len(rd)*) – radial distances from void center for model void density profile
- **d** (*ndarray,len(rd)*) – model void density profile
- **D** (*ndarray,len(rd)*) – radially averaged model void density profile
- **f** (*float*) – linear growth rate (default = 0.5)
- **qper** (*float*) – AP distortion perpendicular to the LOS (default = 1)
- **qpar** (*float*) – AP distortion parallel to the LOS (default = 1)
- **M** (*float*) – monopole amplitude parameter (default = 1)
- **Q** (*float*) – quadrupole amplitude parameter (default = 1)

**Returns**

*xi2d (ndarray,[len(rm),len(rm)])* – model for POS vs. LOS 2d void-tracer correlation function if ell is None

xi (ndarray,[len(ell),len(rm)]): model for multipoles of void-tracer correlation function if ell is not None

## 6.2 plotlib

voiager.plotlib.**fs8_DAH**(*zvi*, *zmin*, *zmax*, *fs8*, *fs8e*, *DAH*, *DAHe*, *legend*, *par_cosmo*, *Nspline=200*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot measurements of f*sigma_8 and D_A*H/c against redshift.

**Parameters**

- **zvi** (*ndarray,Nvbin*) – average void redshift per bin
- **zmin** (*float*) – minimum redshift
- **zmax** (*float*) – maximum redshift
- **fs8** (*ndarray list,[len(fs8),Nvbin]*) – measured values and uncertainties of f*sigma_8
- **fs8e** (*ndarray list,[len(fs8),Nvbin]*) – measured values and uncertainties of f*sigma_8
- **DAH** (*ndarray list,[len(DAH),Nvbin]*) – measured values and uncertainties of D_A*H/c
- **DAHe** (*ndarray list,[len(DAH),Nvbin]*) – measured values and uncertainties of D_A*H/c
- **legend** (*str list*) – legend labels for different measurements
- **par_cosmo** (*dict*) – cosmological parameter values
- **Nspline** (*int*) – number of nodes for spline if Nsmooth > 0 (default = 200)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*fs8_DAH.pdf (pdf file)* – measurements of f*sigma_8 and D_A*H/c and fiducial cosmological model prediction

voiager.plotlib.**logo**(*xi2dts*, *p1*, *Nvbin=2*, *Nspline=200*, *rmax=3.0*, *vmin=-0.8*, *vmax=0.4*, *Nlev=10*, *cmap='Spectral_r'*, *plotPath='plots/'*)

Plot Voiager logo background.

**Parameters**

- **xi2dts** (*ndarray,[Nvbin,10*Nspline,20*Nspline]*) – spline of theory model for POS vs. LOS 2d void-tracer correlation function
- **p1** (*ndarray,[Nvbin,Npar]*) – best-fit model parameter values
- **Nvbin** (*int*) – number of void bins (default = 2)
- **Nspline** (*int*) – number of nodes for spline if Nsmooth > 0 (default = 200)

- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **vmin** (*float*) – minimum and maximum for contour map (default = -0.8, 0.4)
- **vmax** (*float*) – minimum and maximum for contour map (default = -0.8, 0.4)
- **Nlev** (*int*) – number of contour lines (default = 10)
- **cmap** (*str*) – colormap from matplotlib (default = 'Spectral_r'), see https://matplotlib.org/stable/tutorials/colors/colormaps.html
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*logo.png (image file)* – Voiager logo background

voiager.plotlib.**redshiftDistribution**(*zgm*, *zvm*, *ngm*, *nvm*, *zv=None*, *zgrm=None*, *zvrm=None*, *ngrm=None*, *nvrm=None*, *vbin='zv'*, *binning='eqn'*, *Nvbin=2*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot redshift distribution of tracers (galaxies) and voids.

**Parameters**

- **zgm** (*ndarray,Nbin_nz*) – mean redshift of tracers, voids per bin (arithmetic mean of bin edges)
- **zvm** (*ndarray,Nbin_nz*) – mean redshift of tracers, voids per bin (arithmetic mean of bin edges)
- **ngm** (*ndarray,Nbin_nz*) – mean number density of tracers, voids per bin
- **nvm** (*ndarray,Nbin_nz*) – mean number density of tracers, voids per bin
- **zv** (*ndarray,len(zv)*) – void redshifts (default = None)
- **zgrm** (*ndarray,Nbin_nz*) – mean redshift of randoms per bin (default = None)
- **zvrm** (*ndarray,Nbin_nz*) – mean redshift of randoms per bin (default = None)
- **ngrm** (*ndarray,Nbin_nz*) – mean number density of randoms per bin (default = None)
- **nvrm** (*ndarray,Nbin_nz*) – mean number density of randoms per bin (default = None)
- **vbin** (*str*) – binning strategy, 'zv': void-redshift bins (default), 'rv': void-radius bins
- **binning** (*str / list*) – 'eqn' for equal number of voids (default), 'lin' for linear, 'log' for logarithmic. Alternatively, provide a list for custom bin edges.
- **Nvbin** (*int*) – number of void bins (default = 2)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*n_zv.pdf (pdf file)* – redshift distribution of tracers and voids (redshift bins indicated if used)

voiager.plotlib.**tracerBias**(*zg*, *bg*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot tracer (galaxy) bias as function of redshift.

**Parameters**

- **zg** (*ndarray,len(zg)*) – tracer redshifts
- **bg** (*ndarray,len(zg)*) – tracer bias

- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*bias.pdf (pdf file)* – tracer bias as function of redshift

voiager.plotlib.**triangle**(*samples*, *p0*, *p1*, *rvi*, *zvi*, *pLim*, *pop*, *par*, *Nvbin=2*, *vbin='zv'*, *legend=None*, *title=None*, *figFormat='pdf'*, *plotPath='plots/'*)

Make triangle plot of MCMC posterior.

**Parameters**

- **samples** (*ndarray list,[len(samples),Nvbin,Nchain,Npar]*) – list of MCMC samples after thinning and burn-in removal
- **p0** (*ndarray list,[len(samples),Nvbin,Npar]*) – fiducial model parameter values
- **p1** (*ndarray list,[len(samples),Nvbin,Npar]*) – best-fit model parameter values
- **rvi** (*ndarray,Nvbin*) – average effective void radius per bin
- **zvi** (*ndarray,Nvbin*) – average void redshift per bin
- **pLim** (*ndarray list,[len(samples),Nvbin,Npar,2]*) – limits for parameter margins around their mean value
- **pop** (*str list,[len(samples),len(pop)]*) – parameters to exclude from plot, for no exclusion use None
- **par** (*dict*) – model parameter values
- **Nvbin** (*int*) – number of void bins (default = 2)
- **vbin** (*str*) – binning strategy, 'zv': void-redshift bins (default), 'rv': void-radius bins
- **legend** (*str list,len(samples)*) – legend labels for different samples (default = None)
- **title** (*str*) – plot title (default = None)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*triangle.pdf (pdf file)* – triangle plot of posterior parameter distribution

voiager.plotlib.**triangle_cosmo**(*samples*, *logP*, *pLim*, *cosmology*, *par_cosmo*, *blind=True*, *legend=None*, *figFormat='pdf'*, *plotPath='plots/'*)

Make triangle plot of MCMC posterior for cosmological parameters.

**Parameters**

- **samples** (*ndarray list,[len(samples),Nchain,Npar]*) – list of MCMC samples after thinning and burn-in removal
- **logP** (*ndarray,Nchain*) – log likelihood values of first sample
- **pLim** (*ndarray list,[len(samples),Npar,2]*) – limits for parameter margins around their mean value
- **cosmology** (*str*) – cosmological model to consider [either 'LCDM', 'wCDM', or 'w0waCDM']
- **par_cosmo** (*dict*) – cosmological parameter values
- **blind** (*bool*) – If true, subtract mean from chains (default = True)

- **legend** (*str list,len(samples)*) – legend labels for different samples (default = None)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**
*triangle_cosmology.pdf (pdf file)* – triangle plot of posterior cosmological parameter distribution

voiager.plotlib.**voidAbundance**(*yv*, *Nbin*, *zmin*, *zmax*, *sky*, *par_cosmo*, *ysymb*, *yunit*, *ystring*, *ylim=[1e-10, 1e-05]*, *yvr=None*, *Nmock=1*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot void abundance as a function of void properties.

**Parameters**

- **yv** (*ndarray,len(yv)*) – arbitrary void property (e.g., effective radius, redshift, core density, ellipticity)
- **Nbin** (*int*) – number of bins
- **zmin** (*float*) – minimum redshift
- **zmax** (*float*) – maximum redshift
- **sky** (*float*) – sky area in square degrees
- **par_cosmo** (*dict*) – cosmological parameter values
- **ysymb** (*str*) – mathematical symbol of void property
- **yunit** (*str*) – unit of void property
- **ystring** (*str*) – name of void property (abbreviation)
- **ylim** (*tuple,2*) – lower and upper y-axis limit (default = 1e-10,1e-5)
- **yvr** (*ndarray,len(yv)*) – void property random (default = None)
- **Nmock** (*int*) – number of mock catalogs (default = 1)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**
*n_ystring.pdf (pdf file)* – void abundance distribution

voiager.plotlib.**voidBox**(*xv*, *zv*, *azim=45.0*, *elev=-120.0*, *plotPath='plots/'*)

Plot 3d distribution of void centers in a comoving box.

**Parameters**

- **xv** (*ndarray,[len(xv),3]*) – comoving coordinates of void centers
- **zv** (*ndarray,len(zv)*) – void redshifts
- **azim** (*float*) – azimuthal viewing angle in degrees (default = 45.)
- **elev** (*float*) – elevation viewing angle in degrees (default = -120.)
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**
*void_box.png (image file)* – 3d view of void centers in a box, color-coded by redshift

voiager.plotlib.**voidRedshift**(*rv*, *zv*, *rvr=None*, *zvr=None*, *plotPath='plots/'*)

Plot redshift distribution for voids of different effective radius.

**Parameters**

- **rv** (*ndarray,len(rv)*) – effective void radii
- **zv** (*ndarray,len(zv)*) – void redshifts
- **rvr** (*ndarray,len(rvr)*) – effective void radii randoms (default = None)
- **zvr** (*ndarray,len(zvr)*) – void redshift randoms (default = None)
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*void_redshift.png (image file)* – void distribution across effective radius and redshift (color-coded)

voiager.plotlib.**voidSky**(*Xv*, *Xvr=None*, *plotPath='plots/'*)

Plot angular distribution of void centers on the sky.

**Parameters**

- **Xv** (*ndarray,[len(Xv),3]*) – RA, Dec, redshift of void centers
- **Xvr** (*ndarray,[len(Xvr),3]*) – RA, Dec, redshift of void center randoms (default = None)
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*void_sky.png (image file)* – Mollweide projection of void distribution on the sky, color-coded by redshift

voiager.plotlib.**xi**(*xi*, *xiE*, *xits*, *rmi*, *rs*, *rvi*, *zvi*, *p1*, *chi2*, *Nvbin=2*, *rmax=3*, *ell=[0]*, *datavec='2d'*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot multipoles of correlation function with the best-fit model.

**Parameters**

- **xi** (*ndarray,[Nvbin,len(ell),Nrbin]*) – multipoles of void-tracer correlation function and their error
- **xiE** (*ndarray,[Nvbin,len(ell),Nrbin]*) – multipoles of void-tracer correlation function and their error
- **xits** (*ndarray,[Nvbin,len(ell),Nspline]*) – spline of theory model for multipoles of void-tracer correlation function
- **rmi** (*ndarray,[Nvbin,Nrbin]*) – radial distances from void center for each bin
- **rs** (*ndarray,Nspline*) – splined radial distances from void center in units of void effective radius
- **rvi** (*ndarray,Nvbin*) – average effective void radius per bin
- **zvi** (*ndarray,Nvbin*) – average void redshift per bin
- **p1** (*ndarray,[Nvbin,Npar]*) – best-fit model parameter values
- **chi2** (*ndarray,Nvbin*) – reduced chi-squared values of best fit
- **Nvbin** (*int*) – number of void bins (default = 2)

- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **datavec** (*str*) – Define data vector, '1d': multipoles, '2d': POS vs. LOS 2d correlation function (default)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*xi_ell.pdf (pdf file)* – multipoles of correlation function with the best-fit model

voiager.plotlib.**xi_2d**(*xi2d*, *xi2dts*, *rmi2d*, *rvi*, *zvi*, *p1*, *chi2*, *Nvbin=2*, *Nspline=200*, *rmax=3*, *datavec='2d'*, *symLOS=True*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot POS vs. LOS 2d correlation function with the best-fit model.

**Parameters**

- **xi2d** (*ndarray,[Nvbin,Nrbin2d,(2*)Nrbin2d]*) – POS vs. LOS 2d void-tracer correlation function
- **xi2dts** (*ndarray,[Nvbin,10*Nspline,20*Nspline]*) – spline of theory model for POS vs. LOS 2d void-tracer correlation function
- **rmi2d** (*ndarray,[Nvbin,(2*)Nrbin2d]*) – POS and LOS distances from void center for each bin
- **rvi** (*ndarray,Nvbin*) – average effective void radius per bin
- **zvi** (*ndarray,Nvbin*) – average void redshift per bin
- **p1** (*ndarray,[Nvbin,Npar]*) – best-fit model parameter values
- **chi2** (*ndarray,Nvbin*) – reduced chi-squared values of best fit
- **Nvbin** (*int*) – number of void bins (default = 2)
- **Nspline** (*int*) – number of nodes for spline if Nsmooth > 0 (default = 200)
- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **datavec** (*str*) – Define data vector, '1d': multipoles, '2d': POS vs. LOS 2d correlation function (default)
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*xi_2d.pdf (pdf file)* – POS vs. LOS 2d correlation function with the best-fit model

voiager.plotlib.**xi_cov**(*xiC*, *dim=1*, *Nvbin=2*, *ell=[0]*, *symLOS=True*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot normalized covariance matrix of correlation function.

**Parameters**

- **xiC** (*ndarray,[Nvbin,*,*]*) – covariance of (either 1d or 2d) correlation function
- **dim** (*int*) – dimension of data vector [1: multipoles (default), 2: POS vs. LOS]
- **Nvbin** (*int*) – number of void bins (default = 2)

- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **symLOS** (*bool*) – if True, assume symmetry along LOS (default)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*cov_ell.pdf (pdf file)* – covariance of multipoles (if dim=1)

cov_2d.pdf (pdf file): covariance of POS vs. LOS 2d correlation function (if dim=2)

voiager.plotlib.**xi_ell**(*xi*, *xiE*, *xits*, *rmi*, *rs*, *rvi*, *zvi*, *p1*, *Nvbin=2*, *rmax=3*, *ell=[0]*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot multipoles of the same order for all void bins with the best-fit models.

**Parameters**

- **xi** (*ndarray,[Nvbin,len(ell),Nrbin]*) – multipoles of void-tracer correlation function and their error
- **xiE** (*ndarray,[Nvbin,len(ell),Nrbin]*) – multipoles of void-tracer correlation function and their error
- **xits** (*ndarray,[Nvbin,len(ell),Nspline]*) – spline of theory model for multipoles of void-tracer correlation function
- **rmi** (*ndarray,[Nvbin,Nrbin]*) – radial distances from void center for each bin
- **rs** (*ndarray,Nspline*) – splined radial distances from void center in units of void effective radius
- **rvi** (*ndarray,Nvbin*) – average effective void radius per bin
- **zvi** (*ndarray,Nvbin*) – average void redshift per bin
- **p1** (*ndarray,[Nvbin,Npar]*) – best-fit model parameter values
- **Nvbin** (*int*) – number of void bins (default = 2)
- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **ell** (*int list*) – multipole orders to calculate (default = [0,])
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*xi_ell=.pdf (pdf file)* – multipoles of the same order with the best-fit models

voiager.plotlib.**xi_p**(*xip*, *xipE*, *xips*, *xid*, *xidE*, *xids*, *xi*, *xiE*, *xits*, *rmi*, *rs*, *rvi*, *zvi*, *p1*, *Nvbin=2*, *rmax=3*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot projected and deprojected correlation function with its redshift-space monopole.

**Parameters**

- **xip** (*ndarray,[Nvbin,Nrbin]*) – LOS projected void-tracer correlation function and its error
- **xipE** (*ndarray,[Nvbin,Nrbin]*) – LOS projected void-tracer correlation function and its error

- **xips** (*ndarray,[Nvbin,Nspline]*) – spline of LOS projected void-tracer correlation function
- **xid** (*ndarray,[Nvbin,Nrbin]*) – deprojected void-tracer correlation function and its error
- **xidE** (*ndarray,[Nvbin,Nrbin]*) – deprojected void-tracer correlation function and its error
- **xids** (*ndarray,[Nvbin,Nspline]*) – spline of deprojected void-tracer correlation function
- **xi** (*ndarray,[Nvbin,len(ell),Nrbin]*) – multipoles of void-tracer correlation function and its error
- **xiE** (*ndarray,[Nvbin,len(ell),Nrbin]*) – multipoles of void-tracer correlation function and its error
- **xits** (*ndarray,[Nvbin,len(ell),Nspline]*) – spline of theory model for multipoles of void-tracer correlation function
- **rmi** (*ndarray,[Nvbin,Nrbin]*) – radial distances from void center for each bin
- **rs** (*ndarray,Nspline*) – splined radial distances from void center in units of void effective radius
- **rvi** (*ndarray,Nvbin*) – average effective void radius per bin
- **zvi** (*ndarray,Nvbin*) – average void redshift per bin
- **p1** (*ndarray,[Nvbin,Npar]*) – best-fit model parameter values
- **Nvbin** (*int*) – number of void bins (default = 2)
- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = '/plots/')

**Returns**

*xi_p.pdf (pdf file)* – projected and deprojected correlation function with its redshift-space monopole (and best fit)

voiager.plotlib.**xi_p_test**(*rs*, *rmi*, *rvi*, *xid*, *p0=[1, -0.8, 2.0, 8.0]*, *Nvbin=2*, *rmax=3*, *figFormat='pdf'*, *plotPath='plots/'*)

Plot projected and deprojected correlation function for best-fit HSW profile as a test template.

**Parameters**

- **rs** (*ndarray,len(rs)*) – radial distances from void center in units of void effective radius for template
- **rmi** (*ndarray,[Nvbin,Nrbin]*) – radial distances from void center for each bin
- **rvi** (*ndarray,Nvbin*) – average effective void radius per bin
- **xid** (*ndarray,[Nvbin,Nrbin]*) – deprojected void-tracer correlation function
- **xidE** (*ndarray,[Nvbin,Nrbin]*) – deprojected void-tracer correlation function
- **p0** (*ndarray,Npar*) – initial parameter values for HSW profile (default r_s=1, d_c=-0.8, a=2, b=8)
- **Nvbin** (*int*) – number of void bins (default = 2)

- **rmax** (*float*) – maximum distance from void center in units of effective void radius (default = 3)
- **figFormat** (*str*) – format to save figure (default 'pdf')
- **plotPath** (*path*) – name of output path for plot (default = 'plots/')

**Returns**

*xi_p_test.pdf (pdf file)* – projected and deprojected correlation function for best-fit HSW profile

CHAPTER

SEVEN

# PUBLICATIONS

The development of *Voiager* is based on a consecutive series of papers: Hamaus et al. 2015, 2016, 2017, 2020, 2022. Please consider citing those that are relevant for your particular application of *Voiager*. Citation records can be retrieved via the NASA ADS server:

- *Probing cosmology and gravity with redshift-space distortions around voids*
  (Hamaus, Sutter, Lavaux, Wandelt, JCAP 2015)
- *Constraints on Cosmology and Gravity from the Dynamics of Voids*
  (Hamaus, Pisani, Sutter, Lavaux, Escoffier, Wandelt, Weller, PRL 2016)
- *Multipole analysis of redshift-space distortions around cosmic voids*
  (Hamaus, Cousinou, Pisani, Aubert, Escoffier, Weller, JCAP 2017)
- *Precision cosmology with voids in the final BOSS data*
  (Hamaus, Pisani, Choi, Lavaux, Wandelt, Weller, JCAP 2020)
- *Euclid: Forecasts from redshift-space distortions and the Alcock-Paczynski test with cosmic voids*
  (Hamaus, Aubert, Pisani, Contarini, Verza, et al., A&A 2022)

CHAPTER

EIGHT

# LICENSE

CHAPTER

NINE

# CONTACT

You can reach me via the information provided on my personal homepage.

CHAPTER

TEN

# ACKNOWLEDGMENTS

I am grateful for invaluable scientific interactions with the following people who, among others, shaped the design of *Voiager*:

- Alice Pisani
- Ben Wandelt
- Carlos Correa
- Giorgia Pollina
- Giovanni Verza
- Giulia Degni
- Guilhem Lavaux
- Jin-Ah Choi
- Jochen Weller
- Marie Aubert
- Marie-Claude Cousinou
- Nico Schuster
- Paul Sutter
- Sebastian Stammler
- Sofia Contarini
- Stéphanie Escoffier
- genindex
- modindex
- search

# PYTHON MODULE INDEX

## V

# INDEX

## X