SoaPy

m-a-huber

Mar 13, 2022

CONTENTS:

`class soapy.SFS(*params)`

Bases: `object`

This is a class for working with orientable Seifert fibered spaces (SFS) whose base orbifold is the 2-sphere.

`params`

List of integer coefficients representing the SFS specified. These do not necessarily coincide with the input parameters, but rather are normalized in such a way that the corresponding integer surgery diagram is definite.

Type list[int]

`central_weight`

The weight of the central vertex of the normalized surgery description of the SFS specified.

Type int

`branch_weights`

Tuple containing the rational surgery coefficients of the exceptional fibers of the SFS specified.

Type tuple(sym.Rational)

`fractional_branch_weights`

Tuple containing the fractional parts of the branch weights.

Type tuple(sym.Rational)

`euler_number`

The orbifold Euler number of the normalized surgery description of the SFS specified.

Type sym.Rational

`exceptional_fibers`

The number of exceptional fibers of the normalized surgery description of the SFS specified.

Type int

`classmethod from_plumbing(central_weight, *lists_of_coeffs)`

Allows one to construct a soapy.SFS object from an integer plumbing description, all of whose weights are non-zero.

Parameters

- `central_weight` (*int*) – The weight of the central vertex of the plumbing tree.
- `*lists_of_coeffs` (*list[int]*) – A variable number of lists of weights of the branches, each read starting from the central vertex

Raises `Exception` – If any of the weights specified is zero.

Returns The SFS corresponding to the integer plumbing description specified.

Return type *soapy.SFS*

`to_plumbing()`

Returns the definite plumbing (equivalently: an integral surgery description) corresponding to the SFS specified.

Returns A tuple whose first elements is the central weight, followed by the lists of integer weights on the branches (read starting from the central vertex).

Return type tuple(int, lists[int])

`seifert_invariants()`

Returns the Seifert invariants of the SFS specified.

Returns A tuple of the format (Euler number, (tuple of fractional branch weights)).

Return type tuple(sym.Rational, tuple(sy.Rational))

`linking_matrix()`

Returns the linking matrix of the SFS specified.

Returns A SymPy-matrix with SymPy-integers as entries, representing the linking matrix of the integer plumbing corresponding to the SFS specified.

Return type sym.Matrix

`first_homology()`

Returns the first homology of the SFS specified.

Returns The orders of the non-trivial cyclic summands of the first homology of the SFS specified.

Return type tuple(int)

`order_of_first_homology()`

Returns the order of the first homology of the SFS specified.

Returns The order of the first homology of the SFS specified.

Return type int

`spinc_to_HF()`

Computes HF^+ in each spin^c-structure of the SFS specified. The Z[U]-module-structure of HF^+ is encoded as a dictionary of the format {'order of Z[U]-module-summand' : 'list of bottommost gradings of all Z[U]-module-summands of that order'}.

Returns A dictionary of the format {'spin^c-structure' : 'Z[U]-module-structure of HF^+'}.

Return type dict

`print_HF()`

Prints HF^+ of the SFS specified by a definite plumbing in a more legible manner.

Returns Just prints HF^+ of the SFS specified.

Return type None

`correction_terms()`

Returns a list of the corrections terms of the SFS specified.

Returns List of all correction terms of the SFS specified.

Return type list[sym.Rational]

`is_lspace()`

Checks whether or not the SFS specified is a Heegaard Floer L-space.

Returns Whether or not the the SFS specified is an L-space.

Return type bool

`casson_walker()`

Computes the Casson-Walker invariant of the SFS specified.

Returns The Casson-Walker invariant of the SFS specified.

Return type sym.Rational

`is_lens_space()`

Checks whether or not the SFS specified is homeomorphic to a lens space.

Returns Whether or not the the SFS specified is homeomorphic to a lens space.

Return type bool

`to_lens_space()`

Transforms the SFS specified into the corresponding lens space.

Raises `Exception` – If the SFS specified is not homeomorphic to any lens space.

Returns Lens space homeomorphic to the SFS specified.

Return type *soapy.Lens*

`is_prism_mfld()`

Checks whether or not the SFS specified is homeomorphic to a prism manifold.

Returns Whether or not the the SFS specified is homeomorphic to a prism manifold.

Return type bool

`to_prism_mfld()`

Transforms the SFS specified into the corresponding prism manifold.

Raises `Exception` – If the SFS specified is not homeomorphic to any prism manifold.

Returns Prism manifold homeomorphic to the SFS specified.

Return type *soapy.Prism*

`class soapy.Lens`(*p*, *q*)

Bases: `soapy.SFS`

This is a subclass of SFS representing lens spaces.

`p`

The first parameter of the lens space specified, normalized to be greater than zero.

Type int

`q`

The second parameter of the lens space specified, normalized so that p > q > 0.

Type int

`classmethod from_linear_lattice`(**params*)

Allows one to construct a soapy.Lens object from a linear lattice specifying a lens space, all of whose weights are non-zero.

Parameters `*params` (*int*) – A variable number of integer weights of the linear lattice, read starting from either end.

Raises `Exception` – If any of the weights specified is zero.

Returns The lens space corresponding to the linear lattice specified.

Return type *soapy.Lens*

`to_SFS()`

Transforms the lens space specified into a SFS.

Returns The SFS homeomorphic to the lens space specified.

Return type *soapy.SFS*

`to_linear_lattice`(*epsilon=- 1*)

Returns the weights of the linear lattice bounded by the lens space specified. By default, the negative definite linear lattice is returned, unless epsilon is set to 1.

Parameters `epsilon` (*int, optional*) – The sign of definiteness of the linear lattice to be returned. Defaults to -1.

Raises `Exception` – If epsilon is not 1 in absolute value.

Returns A tuple containing the weights of the linear lattice bounded by the lens space specified.

Return type tuple(int)

`class soapy.`**`Prism`**`(`*p*, *q*`)`

Bases: *soapy.SFS*

This is a subclass of SFS representing prism manifolds.

`p`

The first parameter of the prism manifold specified, normalized to be greater than 1.

Type int

`q`

The second parameter of the prism manifold specified; can be any non-zero integer.

Type int

`to_SFS``()`

Transforms the prism manifold specified into a SFS.

Returns The SFS homeomorphic to the prism manifold specified.

Return type *soapy.SFS*

`class soapy.`**`Brieskorn`**`(`**params*`)`

Bases: *soapy.SFS*

This is a subclass of SFS representing Brieskorn homology spheres.

`params`

The parameters of the Brieskorn homology sphere specified.

Type list[int]

`to_SFS``()`

Transforms the Brieskorn homology sphere specified into a SFS.

Returns The SFS homeomorphic to the Brieskorn homology sphere specified.

Return type *soapy.SFS*

PYTHON MODULE INDEX