



cosifer
Release 0.0.1

PhosphorylatedRabbits team

Jan 17, 2020

CONTENTS:

1	cosifer package	1
1.1	Subpackages	1
1.2	Module contents	24
2	cosifer	25
3	Indices and tables	27
	Python Module Index	29
	Index	31

COSIFER PACKAGE

1.1 Subpackages

1.1.1 cosifer.collections package

Submodules

cosifer.collections.graph module

Graph class.

class `cosifer.collections.graph.Graph` (*adjacency, labels=None, undirected=True*)

Bases: `object`

Lightweight class for handling cosifer graph objects. If the graph is undirected the adjacency matrix is stored as a sparse lower triangular matrix.

n

number of nodes.

Type `int`

labels_to_indices

label to index mapping.

Type `pd.Series`

indices_to_labels

index to label mapping.

Type `pd.Series`

adjacency

sparse adjacency.

Type `spicy.sparse.csr_matrix`

undirected

flag indicating whether edges are directed.

Type `bool`

get_scaled_adjacency ()

Get a min-max scaled version of the adjacency.

Returns the min-max scaled adjacency.

Return type `spicy.sparse.csr_matrix`

```
indices_to_labels = None
labels_to_indices = None
n = 0

set_adjacency_from_numpy(adjacency)
    Set the adjacency from a numpy ndarray.

    Parameters adjacency (np.ndarray) – adjacency matrix.

    Raises RuntimeError – in case of inconsistencies in the sizes.

set_adjacency_from_pandas(adjacency)
    Set the adjacency from a pandas dataframe.

    Parameters adjacency (pd.DataFrame) – adjacency matrix.

    Raises RuntimeError – in case of inconsistencies in the sizes.

set_adjacency_from_sparse(adjacency)
    Set the adjacency from a sparse matrix.

    Parameters adjacency (spicy.sparse.csr_matrix) – adjacency matrix.

    Raises RuntimeError – in case of inconsistencies in the sizes.

set_labels(labels)
    Set the node labels.

    Parameters labels (iterable) – labels to set.

    Raises RuntimeError – in case of inconsistencies between the labels and the graph nodes.

to_interaction_table(scaled=True, interaction_symbol='<->')
    Convert the graph to an interaction table.

    Parameters

    • scaled (bool, optional) – flag to activate min-max scaling of the edges. Defaults
      to True.

    • interaction_symbol (str, optional) – symbol to depict interactions between
      labels. Defaults to '<->'.

    Returns

    the table containing the interactions reported in the graph.

    Return type InteractionTable
```

cosifer.collections.interaction_table module

Interaction table class.

```
class cosifer.collections.interaction_table.InteractionTable(df=Empty
    DataFrame
    Columns: [e1, in-
    tensity, e2] Index:
    [], labels=None,
    interac-
    tion_symbol=None,
    force_undirected=False)

Bases: object
```

Interaction table class representation of an edge list.

df

underlying dataframe containing the edge list.

Type `pd.DataFrame`

labels

node labels.

Type `iterable`

apply_filter (*labels=None, prune_labels=True, indices=None, threshold=0.0, top_n=None*)

Apply a filter to get an InteractionTable.

Parameters

- **labels** (*iterable, optional*) – node labels to select. Defaults to None.
- **prune_labels** (*bool, optional*) – prune labels not present in the final table. Defaults to True.
- **indices** (*iterable, optional*) – indices to filter rows. Defaults to None.
- **threshold** (*float, optional*) – threshold for the edge weights. Defaults to 0.0.
- **top_n** (*int, optional*) – number of top interactions to keep. Defaults to None.

Returns a filtered interaction table.

Return type `InteractionTable`

get_df_dict (*threshold=None*)

Get a dictionary to represent the edge list.

Parameters **threshold** (*float, optional*) – threshold to filter edge by intensity. Defaults to None.

Returns a dictionary representing then edge list dataframe.

Return type `dict`

to_edge_list (*interaction_symbol='<->', weights=True*)

Returns an edge list containing the interactions.

Parameters

- **interaction_symbol** (*str, optional*) – Symbol separating the labels in the index of the edge list dataframe. Defaults to '<->'.
- **weights** (*bool, optional*) – Flag indicating whether weights are returned. Defaults to True.

Returns a list of edges represented by tuples.

Return type `list`

to_graph (*undirected=True, imposed_labels=None*)

Get a graph from the stored edges.

Parameters

- **undirected** (*bool, optional*) – flag to indicate whether the interactions are undirected. Defaults to True.
- **imposed_labels** (*iterable, optional*) – node label to consider. Defaults to None.

Returns a graph.

Return type *Graph*

`cosifer.collections.interaction_table.directed_to_undirected_interactions(directed_interactions)`

Processing of a directed table, discarding directions and keeping only the maximum edge value. It will drop and reindex with integers the table. :param: directed_interactions: directed interactions table. :return: an undirected version of the given table.

Parameters `directed_interactions` (*pd.DataFrame*) – directed interactions dataframe.

Returns undirected interactions dataframe

Return type *pd.DataFrame*

`cosifer.collections.interaction_table.interaction_table_from_dict(interaction_dictionary)`

Construct an InteractionTable from a dictionary.

Parameters `interaction_dictionary` (*dict*) – an interaction dictionary. The dictionary should contain two keys: 'labels', containing the node labels, and 'interactions', an object that can be used to construct a dataframe representing the edge list.

Returns the interaction table.

Return type *InteractionTable*

`cosifer.collections.interaction_table.interaction_table_from_edge_list(interaction_list)`

Construct an InteractionTable from a list.

Parameters `interaction_list` (*list*) – an edge list containing tuples.

Returns the interaction table for the provided edge list.

Return type *InteractionTable*

`cosifer.collections.interaction_table.interaction_table_from_gzip(filepath)`

Construct an InteractionTable from a gzipped file containing an edge list.

Parameters `filepath` (*str*) – path to the gipped file.

Returns the interaction table from the gzipped edge list.

Return type *InteractionTable*

`cosifer.collections.interaction_table.interaction_table_to_edge_list(interaction_table,
interaction_symbol='<->',
weights=True)`

Convert an InteractionTable to an edge list.

Parameters

- **interaction_table** (*InteractionTable*) – an interaction table.
- **interaction_symbol** (*str, optional*) – Symbol separating the labels in the index of the edge list dataframe. Defaults to '<->'.
- **weights** (*bool, optional*) – Flag indicating whether weights are returned. Defaults to True.

Returns a list of edges represented by tuples.

Return type *list*

`cosifer.collections.interaction_table.process_group(row)`

Process a grouped edge list dataframe

Parameters `row` (*pd.Series*) – a dataframe row.

Returns a list with entity labels and weight.

Return type list

`cosifer.collections.interaction_table.process_index(index, intensity, interaction_symbol)`

Process index to get edge tuple. Disregard edge weight.

Parameters

- **index** (*str*) – index of the edge list dataframe.
- **intensity** (*float*) – intensity of the edge.
- **interaction_symbol** (*str*) – symbol used to separate node labels.

Returns a tuple containing edge labels.

Return type tuple

`cosifer.collections.interaction_table.process_index_with_weights(index, intensity, interaction_symbol)`

Process index to get edge tuple including edge weight.

Parameters

- **index** (*str*) – index of the edge list dataframe.
- **intensity** (*float*) – intensity of the edge.
- **interaction_symbol** (*str*) – symbol used to separate node labels.

Returns a tuple containing edge labels and weight.

Return type tuple

Module contents

1.1.2 cosifer.combiners package

Submodules

cosifer.combiners.cit module

Combiner for interaction tables.

class `cosifer.combiners.cit.CombineInteractionTables` (*combine_tables=<function mean_table>*, *interaction_symbol='<->'*, *name='cit'*, ***kwargs*)

Bases: `cosifer.combiners.network_combiner.NetworkCombiner`

Combine interaction tables representing networks using a function.

name

name of the combiner.

Type str

combine_tables

function to combine interaction tables.

Type function

interaction_symbol

symbol used to indicate interactions in the index of the dataframe.

Type str

graph

combined graph.

Type cosifer.collections.Graph

```
cosifer.combiners.cit.combine_tables(table_list, interaction_symbol='<->', processing_concatenated_intensities_fn=<function  
<lambda>>, reduce_fn=<function <lambda>>,  
**kwargs)
```

Compute the combined intensities from a list of interaction tables.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str, optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.
- **processing_concatenated_intensities_fn** (*function, optional*) – function to apply on the concatenated intensities. Defaults to identity.
- **reduce_fn** (*function, optional*) – function to reduce intensities over dataframe rows. Defaults to mean.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.get_scaled_ranks(dataframe)
```

Scaled ranks from an intensity dataframe.

Parameters **dataframe** (*pd.DataFrame*) – intensity dataframe.

Returns scaled ranks dataframe

Return type *pd.DataFrame*

```
cosifer.combiners.cit.hard_mean_scaled_ranks_table(table_list,  
                                                    interaction_symbol='<->',  
                                                    **kwargs)
```

Compute the mean on the scaled ranks without considering interaction existence.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str, optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.hard_mean_table(table_list, interaction_symbol='<->', **kwargs)
```

Compute the mean on the intensities without considering interaction existence.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.max_scaled_ranks_table(table_list, interaction_symbol='<->',
                                             **kwargs)
```

Compute the maximum on the scaled ranks.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.max_table(table_list, interaction_symbol='<->', **kwargs)
```

Compute the maximum on the intensities.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.mean_scaled_ranks_table(table_list, interaction_symbol='<->',
                                             **kwargs)
```

Compute the mean on the scaled ranks.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.mean_table(table_list, interaction_symbol='<->', **kwargs)
```

Compute the mean on the intensities.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.median_scaled_ranks_table(table_list, interaction_symbol='<->',  
                                                **kwargs)
```

Compute the median on the scaled ranks.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.median_table(table_list, interaction_symbol='<->', **kwargs)
```

Compute the median on the intensities.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.min_scaled_ranks_table(table_list, interaction_symbol='<->',  
                                              **kwargs)
```

Compute the minimum on the scaled ranks.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

```
cosifer.combiners.cit.min_table(table_list, interaction_symbol='<->', **kwargs)
```

Compute the minimum on the intensities.

Parameters

- **table_list** (*list*) – a list of interaction tables.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns the combined interaction table.

Return type *InteractionTable*

cosifer.combiners.core module

Core combiner utilities.

```
cosifer.combiners.core.combined_df_to_interaction_table(combined, table_list,  
                                                         interaction_symbol='<->')
```

Transform combined intensities dataframe into an InteractionTable.

Parameters

- **combined** (*pd.DataFrame*) – combined intensities dataframe.
- **table_list** (*list*) – a list of *InteractionTable* objects.
- **interaction_symbol** (*str, optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns

an *InteractionTable* representing the combined intensities.

Return type *InteractionTable*

`cosifer.combiners.core.concatenate_intensities(table_list, threshold_rate=None)`
 Concatenate intensities from a list of *InteractionTable* objects.

Parameters

- **table_list** (*list*) – a list of *InteractionTable* objects.
- **threshold_rate** (*float, optional*) – threshold rate for the NAs. Defaults to None, a.k.a no threshold applied.

Returns

an *InteractionTable* representing the combined intensities.

Return type *InteractionTable*

cosifer.combiners.network_combiner module

Interface for a network combiner.

class `cosifer.combiners.network_combiner.NetworkCombiner` (*filepath=None, **kwargs*)

Bases: `cosifer.handlers.network_handler.NetworkHandler`

Abstract interface for a network combiner.

trained

flag to indicate whether the combiner is already trained.

Type bool

combine (*results_list*)

Apply the combination method. Checking whehter the combiner has been already trained.

Parameters **results_list** (*list*) – a list of *InteractionTable* objects.

cosifer.combiners.snf module

SNF combiner.

class `cosifer.combiners.snf.SNF` (*interaction_symbol='<->', name='snf', **kwargs*)

Bases: `cosifer.combiners.network_combiner.NetworkCombiner`

Combine interaction tables representing networks using SNF.

interaction_symbol

symbol used to indicate interactions in the index of the dataframe.

Type str

name

name of the combiner.

Type str

```
cosifer.combiners.snf.compute_snf(results_list, labels, K=20, T=10,
                                   snf=ipy2.robjecs.packages.Package as a <module
                                   'SNFtool'>)
```

Compute combination via SNF.

Parameters

- **results_list** (*list*) – a list of InteractionTable objects.
- **labels** (*list*) – labels to consider.
- **K** (*int, optional*) – number of nearest neighbors. Defaults to 20.
- **T** (*int, optional*) – number of steps in the diffusion process. Defaults to 10.
- **snf** (*object, optional*) – SNF rpy2 object. Defaults to importr('SNFtool').

Returns the combined graph.**Return type** *Graph***cosifer.combiners.summa module**

Combiner using SUMMA.

```
class cosifer.combiners.summa.Summa(interaction_symbol='<->', name='summa', **kwargs)
```

Bases: *cosifer.combiners.network_combiner.NetworkCombiner*

SUMMA algorithm for combining results.

interaction_symbol

symbol used to indicate interactions in the index of the dataframe.

Type str**name**

name of the combiner.

Type str**summa_object**

SUMMA object.

Type pySUMMA.Summa**summa_object = None**

```
cosifer.combiners.summa.fill_na_ranks(a_series)
```

Fill NA ranks in a pd.Series.

Parameters **a_series** (*pd.Series*) – series to fill.**Returns** the filled series.**Return type** pd.Series

```
cosifer.combiners.summa.summa_scores_table(table_list, interaction_symbol='<->',
                                             **kwargs)
```

Apply SUMMA as a method on list of interaction tables.

Parameters

- **table_list** (*list*) – a list of InteractionTable objects.
- **interaction_symbol** (*str*, *optional*) – symbol used to indicate interactions in the index of the dataframe. Defaults to '<->'.

Returns interaction table with SUMMA scores.

Return type *InteractionTable*

Module contents

Combiner module.

1.1.3 cosifer.handlers package

Submodules

cosifer.handlers.file_handler module

FileHandler interface.

class cosifer.handlers.file_handler.**FileHandler** (**kwargs)
Bases: object

FileHandler interface class.

dump (*buffer*)
Dump object to the file.

Parameters **buffer** (*object*) – a buffer-like object.

exist ()
Check whether the file exists.

load ()
Load object from the file.

cosifer.handlers.fs_handler module

FileSystemHandler inferencer.

class cosifer.handlers.fs_handler.**FileSystemHandler** (*filepath*, **kwargs)
Bases: *cosifer.handlers.file_handler.FileHandler*

FileSystemHandler inferencer.

filepath
path to the file.

Type str

dump (*buffer*, *file_type*='w')
Dump object to the file.

Parameters

- **buffer** (*object*) – a buffer-like object.
- **file_type** (*str*, *optional*) – type of the file. Defaults to 'w'.

exist()
Check whether the file exists.
Returns true if the file exists, false otherwise.
Return type bool

filepath = None

load(file_type='r')
Load object from the file.
Argd: file_type (str, optional): type of the file. Defaults to 'r'.
Returns the loaded file.
Return type IOBase

cosifer.handlers.network_handler module

NetworkHandler interface.

class cosifer.handlers.network_handler.**NetworkHandler** (filepath=None, **kwargs)
Bases: *cosifer.handlers.fs_handler.FileSystemHandler*
NetworkHandler interface.

graph
graph representing the network.
Type cosifer.collections.graph,*Graph*

trained
flag to indicate whether the inference has been performed.
Type bool

filepath
path to the file where the graph is stored.
Type str

parameters
parameters for the inferencer.
Type dict

dump (scaled=True, interaction_symbol='<->', threshold=None, compression='gzip')
Dump graph to the file.
Parameters

- **scaled** (*bool*, *optional*) – flag to activate min-max scaling of the edges. Defaults to True.
- **interaction_symbol** (*str*, *optional*) – symbol to depict interactions between labels. Defaults to '<->'.
- **threshold** (*float*, *optional*) – threshold to apply to the intensity. Defaults to None.
- **compression** (*str*, *optional*) – compression type. Defaults to 'gzip'.

filepath = None


```

graph = None
load (compression='gzip')
    Load graph from the file.

    Argd: compression (str, optional): compression type. Defaults to 'gzip'.

trained = False

```

Module contents

1.1.4 cosifer.inferencers package

Submodules

cosifer.inferencers.aracne module

Aracne inferencer.

```

class cosifer.inferencers.aracne.Aracne (estimator, disc='none', method='ARACNE',
                                         **kwargs)

```

Bases: *cosifer.inferencers.network_inferencer.NetworkInferencer*

Aracne inferencer implementation.

estimator

estimator type.

Type str

disc

discretization type.

Type str

method

name of the method.

Type str

cosifer.inferencers.clr module

CLR inferencer.

```

class cosifer.inferencers.clr.CLR (estimator, disc='none', method='CLR', **kwargs)

```

Bases: *cosifer.inferencers.network_inferencer.NetworkInferencer*

CLR inferencer implementation.

estimator

estimator type.

Type str

disc

discretization type.

Type str

method

name of the method.

Type str

cosifer.inferencers.correlation module

Correlation inferencer.

```
class cosifer.inferencers.correlation.Correlation (method=None, correction=None,  
                                                confidence_threshold=0.05,  
                                                **kwargs)
```

Bases: *cosifer.inferencers.network_inferencer.NetworkInferencer*

Correlation inferencer.

method

correlation method.

Type str

correction

correction method.

Type str

confidence_threshold

confidence threshold.

Type float

method = None

cosifer.inferencers.funchisq module

FunChisq inferencer.

```
class cosifer.inferencers.funchisq.FunChisq (k_min=3, k_max=7, k_step=1,  
                                              method='FunChisq', correction=None, con-  
                                              fidence_threshold=0.05, undirected=True,  
                                              **kwargs)
```

Bases: *cosifer.inferencers.network_inferencer.NetworkInferencer*

FunChisq inferencer.

k_min

minimum number of quantization bins.

Type int

k_max

maximum number of quantization bins.

Type int

k_step

number of steps for bins search.

Type int

method

name of the method.

Type str

correction

correction method.

Type str**confidence_threshold**

confidence threshold.

Type float**undirected**

flag to indicate an undirected network.

Type bool`cosifer.inferencers.funchisq.sort_interaction_entities(row)`

Sort the entities over a row in lexicographic order.

Parameters `row` (*pd.Series*) – row containing the entities to be sorted.**Returns**

a list containing the sorted entities and the rest of the elements of the row unchanged.

Return type list**cosifer.inferencers.genie3 module**

GENIE3 inferencer.

```
class cosifer.inferencers.genie3.GENIE3(tree_method='RF', k='sqrt', n_trees=1000,
                                         regulators=rpy2.rinterface.NULL, tar-
                                         gets=rpy2.rinterface.NULL, n_cores=4, ver-
                                         bose=False, method='GENIE3', **kwargs)
```

Bases: *cosifer.inferencers.network_inferencer.NetworkInferencer*

GENIE3 inferencer.

tree_method

tree method.

Type str**k**

k criterion.

Type str**n_trees**

number of trees.

Type int**regulators**

known regulators.

Type object**targets**

known targets.

Type object**n_cores**

number of cores.

Type int
verbose
toggle verbosity.
Type bool
method
name of the method.
Type str

cosifer.inferencers.glasso module

Glasso inferencer.

```
class cosifer.inferencers.glasso.Glasso (correction=None, method='gLasso', **kwargs)  
    Bases: cosifer.inferencers.network_inferencer.NetworkInferencer  
    Glasso inferencer.  
    method  
        name of the method.  
    Type str
```

cosifer.inferencers.jrf module

JRF inferencer.

```
class cosifer.inferencers.jrf.JointRandomForest (ntree=500, mtry=None,  
                                                merger=<function JointRandom-  
                                                Forest.<lambda>>, correction=None,  
                                                method='JRF', **kwargs)  
    Bases: cosifer.inferencers.network_inferencer.NetworkInferencer  
    JRF inferencer.  
    ntree  
        number of trees.  
    Type int  
    mtry  
        number of variables for splitting.  
    Type int  
    merger  
        a merger function.  
    Type function  
    method  
        name of the method.  
    Type str
```

cosifer.inferencers.mrnet module

MRNET inferencer.

```
class cosifer.inferencers.mrnet.MRNET (estimator, disc='none', method='MRNET',
                                     **kwargs)
    Bases: cosifer.inferencers.network_inferencer.NetworkInferencer
    MRNET inferencer.

    estimator
        estimator type.
        Type str

    disc
        discretization type.
        Type str

    method
        name of the method.
        Type str
```

cosifer.inferencers.network_inferencer module

NetworkInferencer abstract interface.

```
class cosifer.inferencers.network_inferencer.NetworkInferencer (**kwargs)
    Bases: cosifer.handlers.network_handler.NetworkHandler
    Network inferencer interface.

    graph
        graph representing the network.
        Type cosifer.collections.graph,Graph

    trained
        flag to indicate whether the inference has been performed.
        Type bool

    graph = None

    infer_network (data)
        Infer the network.

        Parameters data (pd.DataFrame) – data to be used for the inference.

    trained = False
```

cosifer.inferencers.tigress module

TIGRESS inferencer.

```
class cosifer.inferencers.tigress.TIGRESS (tf_list=ipy2.rinterface.NULL, k=-1, alpha=0.2,
                                           n_steps_lars=5, n_bootstrap=1000, scoring='area',
                                           verbose=False, use_parallel=True,
                                           n_cores=4, method='TIGRESS', **kwargs)
    Bases: cosifer.inferencers.network_inferencer.NetworkInferencer
```

TIGRESS inferencer.

tf_list

list of transcription factor.

Type object

k

number of edges to return.

Type int

alpha

alpha parameter.

Type float

n_step_lars

number of LARS steps.

Type int

n_bootstrap

bootstrap number.

Type int

scoring

scoring criterion.

Type str

verbose

toggle verbosity.

Type bool

use_parallel

enable parallelism.

Type bool

n_cores

number of cores.

Type int

method

name of the method.

Type str

Module contents

Inferencer module.

1.1.5 cosifer.pipelines package

Submodules

cosifer.pipelines.pipeline_cli module

COSIFER client pipeline.

`cosifer.pipelines.pipeline_cli.get_interaction_tables(output_directory)`

Transform graphs from the output directory into interaction tables.

Parameters `output_directory` (*str*) – path to the output directory

Returns interaction tables from each method in a dictionary.

Return type dict

`cosifer.pipelines.pipeline_cli.method_selection(methods=None)`

Select inference methods.

Parameters `methods` (*list*) – list of inferencers to run. Defaults to None, a.k.a., use defaults.

Returns a dictionary keyed by method name and inferencers as values.

Return type dict

`cosifer.pipelines.pipeline_cli.run(filepath, output_directory, standardize=True, samples_on_rows=True, sep='\t', fillna=0.0, header=0, index_col=0, methods=None, combiner=None, gmt_filepath=None, **kwargs)`

Run COSIFER client pipeline.

Parameters

- **filepath** (*str*) – path to the file.
- **output_directory** (*str*) – path where to store the results.
- **standardize** (*bool, optional*) – toggle data standardization. Defaults to True.
- **samples_on_rows** (*bool, optional*) – flag to indicate whether data are following the format where each row represents a sample. Defaults to True.
- **sep** (*str, optional*) – field separator. Defaults to ‘\t’.
- **fillna** (*float, optional*) – value used to fill NAs. Defaults to 0.
- **header** (*int, optional*) – line for the header in the input file. Defaults to 0.
- **index_col** (*int, optional*) – column index for the input index. Defaults to 0.
- **methods** (*list, optional*) – inference methods. Defaults to None, a.k.a., only recommended methods.
- **combiner** (*str, optional*) – combiner type. Defaults to None, a.k.a., no combination.
- **gmt_filepath** (*str, optional*) – GMT file containing feature sets. Defaults to None, a.k.a., no GMT file provided.

`cosifer.pipelines.pipeline_cli.run_combiner(combiner_name, interaction_tables_dict, output_directory)`

Combine interaction tables received from every methods and save the interaction table in the output directory.

Parameters

- **combiner_name** (*str*) – combiner type.
- **interaction_tables_dict** (*dict*) – dictionary containing interaction tables from each method.

- **output_directory** (*str*) – path to the output directory.

`cosifer.pipelines.pipeline_cli.run_inference(data, selected_methods, output_directory)`

Perform network inference of the data given a set of methods and save the predicted graphs in an output directory.

Parameters

- **data** (*pd.DataFrame*) – input dataframe.
- **selected_methods** (*dict*) – selected inference methods.
- **output_directory** (*str*) – output directory.

cosifer.pipelines.pipeline_gui module

COSIFER GUI pipeline.

`cosifer.pipelines.pipeline_gui.method_selection(methods=None)`

Select inference methods.

Parameters **methods** (*list*) – list of inferencers to run. Defaults to None, a.k.a., use defaults.

Returns a dictionary keyed by method name and inferencers as values.

Return type *dict*

`cosifer.pipelines.pipeline_gui.run(data, results_filepath, methods=None, combiner='summa')`

Run COSIFER GUI pipeline.

Parameters

- **data** (*pd.DataFrame*) – data used for inference.
- **results_filepath** (*str*) – path where to store the results.
- **methods** (*list, optional*) – inference methods. Defaults to None, a.k.a., only recommended methods.
- **combiner** (*str, optional*) – combiner type. Defaults to *summa*.

`cosifer.pipelines.pipeline_gui.run_combiner(combiner_name, interaction_tables_dict, results_filepath)`

Combine interaction tables received from every methods and save the interaction table to a output file.

Parameters

- **combiner_name** (*str*) – combiner type.
- **interaction_tables_dict** (*dict*) – dictionary containing interaction tables from each method.
- **results_filepath** (*str*) – path to the results.

`cosifer.pipelines.pipeline_gui.run_inference(data, selected_methods)`

Perform network inference of the data given a set of methods and save the predicted graphs in an output directory.

Parameters

- **data** (*pd.DataFrame*) – input dataframe.
- **selected_methods** (*dict*) – selected inference methods.

Returns interaction tables inferred with the selected methods.

Return type *dict*

Module contents

1.1.6 cosifer.utils package

Submodules

cosifer.utils.data module

Data utilities.

`cosifer.utils.data.get_synthetic_data(n_samples, n_features, precision_matrix=None, alpha=0.98, seed=1)`

Generate synthetic data using a covariance matrix obtained by inverting a randomly generated precision matrix.

Parameters

- **n_samples** (*[type]*) – [description]
- **n_features** (*[type]*) – [description]
- **precision_matrix** (*[type]*, *optional*) – [description]. Defaults to None.
- **alpha** (*float*, *optional*) – [description]. Defaults to 0.98.
- **seed** (*int*, *optional*) – [description]. Defaults to 1.

Returns

a tuple with two elements. The first is a **pd.DataFrame** representing the data. The second is the precision matrix used to generate the data.

Return type

`cosifer.utils.data.read_data(filepath, standardize=True, samples_on_rows=True, sep='\t', fillna=0.0, **kwargs)`

Read data from file.

Parameters

- **filepath** (*str*) – path to the file.
- **standardize** (*bool*, *optional*) – toggle data standardization. Defaults to True.
- **samples_on_rows** (*bool*, *optional*) – flag to indicate whether data are following the format where each row represents a sample. Defaults to True.
- **sep** (*str*, *optional*) – field separator. Defaults to ‘\t’.
- **fillna** (*float*, *optional*) – value used to fill NAs. Defaults to 0.

Returns a dataframe parsed from the provided filepath.

Return type

`cosifer.utils.data.read_gmt(filepath)`

Read a GMT file.

Parameters **filepath** (*str*) – path to a GMT file.

Returns a dictionary containing sets of features.

Return type

`cosifer.utils.data.scale_graph(graph, threshold=0.0)`

Min-max scale a matrix representing a graph assuming positive values.

Parameters

- **graph** (*pd.DataFrame*) – a dataframe representing a graph.
- **threshold** (*float, optional*) – threshold to impose on the edge weights. Defaults to .0.

Returns a dataframe representing the scaled graph.

Return type *pd.DataFrame*

cosifer.utils.stats module

Statistics utils.

`cosifer.utils.stats.benjamini_hochberg_correction(p_values, q_star)`

Return indices of pValues that make reject null hypothesis at given significance level with a Benjamini-Hochberg correction. Used implementation robust to nan values through statsmodels.

Parameters

- **p_values** (*iterable*) – p-values to be used for correction.
- **q_star** (*float*) – false discovery rate.

Returns indices of significant p-values.

Return type *list*

`cosifer.utils.stats.benjamini_yekutieli_correction(p_values, q_star)`

Return indices of pValues that make reject null hypothesis at given significance level with a Benjamini-Yekutieli correction. Used implementation robust to nan values through statsmodels.

Parameters

- **p_values** (*iterable*) – p-values to be used for correction.
- **q_star** (*float*) – false discovery rate.

Returns indices of significant p-values.

Return type *list*

`cosifer.utils.stats.bonferroni_correction(p_values, q_star)`

Return indices of pValues that make reject null hypothesis at given significance level with a Bonferroni correction. Used implementation robust to nan values through statsmodels.

Parameters

- **p_values** (*iterable*) – p-values to be used for correction.
- **q_star** (*float*) – false discovery rate.

Returns indices of significant p-values.

Return type *list*

`cosifer.utils.stats.from_precision_matrix_partial_correlations(precision, scaled=False)`

Compute partial correlations from the precision matrix.

Parameters

- **precision** (*np.ndarray*) – a precision matrix.
- **scaled** (*bool, optional*) – flag to min-max scale the correlations. Defaults to False.

Returns the partial correlation matrix.

Return type np.ndarray

cosifer.utils.vector_quantization module

Vector quantization utilities.

`cosifer.utils.vector_quantization.k_means_bic(X, clusters_centers, clusters_labels, sigma_eps=1.0)`

Compute BIC for K-means clustering.

Parameters

- **X** (*np.ndarray*) – clustered data.
- **clusters_centers** (*np.ndarray*) – cluster centers.
- **clusters_labels** (*np.ndarray*) – cluster labels.
- **sigma_eps** (*float, optional*) – standard deviation. Defaults to 1..

Returns BIC score.

Return type float

`cosifer.utils.vector_quantization.k_means_optimized_with_bic(X, k_min=3, k_max=9, k_step=1, sigma_eps=1.0, n_init=100, **kwargs)`

Find an optimal K-mean model minizing the BIC score.

Parameters

- **X** (*np.ndarray*) – data to cluster.
- **k_min** (*int, optional*) – minimum number of clusters. Defaults to 3.
- **k_max** (*int, optional*) – maximum number of clusters. Defaults to 9.
- **k_step** (*int, optional*) – number of cluster steps. Defaults to 1.
- **sigma_eps** (*float, optional*) – standard deviation. Defaults to 1..
- **n_init** (*int, optional*) – number of K-means initializations. Defaults to 100.

Returns

a tuple containing two elements: the first is the optimal model, the second one is a dictionary mapping the number of clusters to the BIC score.

Return type tuple

`cosifer.utils.vector_quantization.k_means_vector_quantization(x, k_min=3, k_max=9, k_step=1, sigma_eps=1.0, n_init=100, **kwargs)`

Quantize a vector using K-means optimized via BIC score.

Parameters

- **x** (*np.ndarray*) – array to quantize.
- **k_min** (*int, optional*) – minimum number of clusters. Defaults to 3.
- **k_max** (*int, optional*) – maximum number of clusters. Defaults to 9.
- **k_step** (*int, optional*) – number of cluster steps. Defaults to 1.
- **sigma_eps** (*float, optional*) – standard deviation. Defaults to 1..
- **n_init** (*int, optional*) – number of K-means initializations. Defaults to 100.

Returns the quantized vector.

Return type np.ndarray

Module contents

1.2 Module contents

COSIFER, a module for consensus network inference.

CHAPTER

TWO

COSIFER

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

C

- `cosifer`, [24](#)
- `cosifer.collections`, [5](#)
- `cosifer.collections.graph`, [1](#)
- `cosifer.collections.interaction_table`,
[2](#)
- `cosifer.combiners`, [11](#)
- `cosifer.combiners.cit`, [5](#)
- `cosifer.combiners.core`, [8](#)
- `cosifer.combiners.network_combiner`, [9](#)
- `cosifer.combiners.snf`, [9](#)
- `cosifer.combiners.summa`, [10](#)
- `cosifer.handlers`, [13](#)
- `cosifer.handlers.file_handler`, [11](#)
- `cosifer.handlers.fs_handler`, [11](#)
- `cosifer.handlers.network_handler`, [12](#)
- `cosifer.inferencers`, [18](#)
- `cosifer.inferencers.aracne`, [13](#)
- `cosifer.inferencers.clr`, [13](#)
- `cosifer.inferencers.correlation`, [14](#)
- `cosifer.inferencers.funchisq`, [14](#)
- `cosifer.inferencers.genie3`, [15](#)
- `cosifer.inferencers.glasso`, [16](#)
- `cosifer.inferencers.jrf`, [16](#)
- `cosifer.inferencers.mrnet`, [17](#)
- `cosifer.inferencers.network_inferencer`,
[17](#)
- `cosifer.inferencers.tigress`, [17](#)
- `cosifer.pipelines`, [21](#)
- `cosifer.pipelines.pipeline_cli`, [19](#)
- `cosifer.pipelines.pipeline_gui`, [20](#)
- `cosifer.utils`, [24](#)
- `cosifer.utils.data`, [21](#)
- `cosifer.utils.stats`, [22](#)
- `cosifer.utils.vector_quantization`, [23](#)

A

adjacency (*cosifer.collections.graph.Graph* attribute), 1
 alpha (*cosifer.inferencers.tigress.TIGRESS* attribute), 18
 apply_filter() (*cosifer.collections.interaction_table.InteractionTable* method), 3
 Aracne (*class in cosifer.inferencers.aracne*), 13

B

benjamini_hochberg_correction() (*in module cosifer.utils.stats*), 22
 benjamini_yekutieli_correction() (*in module cosifer.utils.stats*), 22
 bonferroni_correction() (*in module cosifer.utils.stats*), 22

C

CLR (*class in cosifer.inferencers.clr*), 13
 combine() (*cosifer.combiners.network_combiner.NetworkCombiner* method), 9
 combine_tables (*cosifer.combiners.cit.CombineInteractionTables* attribute), 5
 combine_tables() (*in module cosifer.combiners.cit*), 6
 combined_df_to_interaction_table() (*in module cosifer.combiners.core*), 8
 CombineInteractionTables (*class in cosifer.combiners.cit*), 5
 compute_snf() (*in module cosifer.combiners.snf*), 10
 concatenate_intensities() (*in module cosifer.combiners.core*), 9
 confidence_threshold (*cosifer.inferencers.correlation.Correlation* attribute), 14
 confidence_threshold (*cosifer.inferencers.funchisq.FunChisq* attribute), 15
 correction (*cosifer.inferencers.correlation.Correlation* attribute), 14
 correction (*cosifer.inferencers.funchisq.FunChisq* attribute), 14

Correlation (*class in cosifer.inferencers.correlation*), 14
 cosifer (*module*), 24
 cosifer.collections (*module*), 5
 cosifer.collections.graph (*module*), 1
 cosifer.collections.interaction_table (*module*), 2
 cosifer.combiners (*module*), 11
 cosifer.combiners.cit (*module*), 5
 cosifer.combiners.core (*module*), 8
 cosifer.combiners.network_combiner (*module*), 9
 cosifer.combiners.snf (*module*), 9
 cosifer.combiners.summa (*module*), 10
 cosifer.handlers (*module*), 13
 cosifer.handlers.file_handler (*module*), 11
 cosifer.handlers.fs_handler (*module*), 11
 cosifer.handlers.network_handler (*module*), 12
 cosifer.inferencers (*module*), 18
 cosifer.inferencers.aracne (*module*), 13
 cosifer.inferencers.clr (*module*), 13
 cosifer.inferencers.correlation (*module*), 14
 cosifer.inferencers.funchisq (*module*), 14
 cosifer.inferencers.genie3 (*module*), 15
 cosifer.inferencers.glasso (*module*), 16
 cosifer.inferencers.jrf (*module*), 16
 cosifer.inferencers.mrnet (*module*), 17
 cosifer.inferencers.network_inferencer (*module*), 17
 cosifer.inferencers.tigress (*module*), 17
 cosifer.pipelines (*module*), 21
 cosifer.pipelines.pipeline_cli (*module*), 19
 cosifer.pipelines.pipeline_gui (*module*), 20
 cosifer.utils (*module*), 24
 cosifer.utils.data (*module*), 21
 cosifer.utils.stats (*module*), 22
 cosifer.utils.vector_quantization (*module*), 23

D

`df` (*cosifer.collections.interaction_table.InteractionTable* attribute), 3

`directed_to_undirected_interactions()` (in module *cosifer.collections.interaction_table*), 4

`disc` (*cosifer.inferencers.aracne.Aracne* attribute), 13

`disc` (*cosifer.inferencers.clr.CLR* attribute), 13

`disc` (*cosifer.inferencers.mrnet.MRNET* attribute), 17

`dump()` (*cosifer.handlers.file_handler.FileHandler* method), 11

`dump()` (*cosifer.handlers.fs_handler.FileSystemHandler* method), 11

`dump()` (*cosifer.handlers.network_handler.NetworkHandler* method), 12

E

`estimator` (*cosifer.inferencers.aracne.Aracne* attribute), 13

`estimator` (*cosifer.inferencers.clr.CLR* attribute), 13

`estimator` (*cosifer.inferencers.mrnet.MRNET* attribute), 17

`exist()` (*cosifer.handlers.file_handler.FileHandler* method), 11

`exist()` (*cosifer.handlers.fs_handler.FileSystemHandler* method), 11

F

`FileHandler` (class in *cosifer.handlers.file_handler*), 11

`filepath` (*cosifer.handlers.fs_handler.FileSystemHandler* attribute), 11, 12

`filepath` (*cosifer.handlers.network_handler.NetworkHandler* attribute), 12

`FileSystemHandler` (class in *cosifer.handlers.fs_handler*), 11

`fill_na_ranks()` (in module *cosifer.combiners.summa*), 10

`from_precision_matrix_partial_correlations()` (in module *cosifer.utils.stats*), 22

`FunChisq` (class in *cosifer.inferencers.funchisq*), 14

G

`GENIE3` (class in *cosifer.inferencers.genie3*), 15

`get_df_dict()` (*cosifer.collections.interaction_table.InteractionTable* method), 3

`get_interaction_tables()` (in module *cosifer.pipelines.pipeline_cli*), 19

`get_scaled_adjacency()` (*cosifer.collections.graph.Graph* method), 1

`get_scaled_ranks()` (in module *cosifer.combiners.cit*), 6

`get_synthetic_data()` (in module *cosifer.utils.data*), 21

`Glasso` (class in *cosifer.inferencers.glasso*), 16

`Graph` (class in *cosifer.collections.graph*), 1

`graph` (*cosifer.combiners.cit.CombineInteractionTables* attribute), 6

`graph` (*cosifer.handlers.network_handler.NetworkHandler* attribute), 12

`graph` (*cosifer.inferencers.network_inferencer.NetworkInferencer* attribute), 17

H

`hard_mean_scaled_ranks_table()` (in module *cosifer.combiners.cit*), 6

`hard_mean_table()` (in module *cosifer.combiners.cit*), 6

I

`indices_to_labels` (*cosifer.collections.graph.Graph* attribute), 1

`infer_network()` (*cosifer.inferencers.network_inferencer.NetworkInferencer* method), 17

`interaction_symbol` (*cosifer.combiners.cit.CombineInteractionTables* attribute), 6

`interaction_symbol` (*cosifer.combiners.snf.SNF* attribute), 9

`interaction_symbol` (*cosifer.combiners.summa.Summa* attribute), 10

`interaction_table_from_dict()` (in module *cosifer.collections.interaction_table*), 4

`interaction_table_from_edge_list()` (in module *cosifer.collections.interaction_table*), 4

`interaction_table_from_gzip()` (in module *cosifer.collections.interaction_table*), 4

`interaction_table_to_edge_list()` (in module *cosifer.collections.interaction_table*), 4

`InteractionTable` (class in *cosifer.collections.interaction_table*), 2

J

`JointRandomForest` (class in *cosifer.inferencers.jrf*), 16

K

`k` (*cosifer.inferencers.genie3.GENIE3* attribute), 15

`k` (*cosifer.inferencers.tigress.TIGRESS* attribute), 18

`k_max` (*cosifer.inferencers.funchisq.FunChisq* attribute), 14

`k_means_bic()` (in module *cosifer.utils.vector_quantization*), 23

`k_means_optimized_with_bic()` (in module `cosifer.utils.vector_quantization`), 23
`k_means_vector_quantization()` (in module `cosifer.utils.vector_quantization`), 23
`k_min` (`cosifer.inferencers.funchisq.FunChisq` attribute), 14
`k_step` (`cosifer.inferencers.funchisq.FunChisq` attribute), 14

L

`labels` (`cosifer.collections.interaction_table.InteractionTable` attribute), 3
`labels_to_indices` (`cosifer.collections.graph.Graph` attribute), 1, 2
`load()` (`cosifer.handlers.file_handler.FileHandler` method), 11
`load()` (`cosifer.handlers.fs_handler.FileSystemHandler` method), 12
`load()` (`cosifer.handlers.network_handler.NetworkHandler` method), 13

M

`max_scaled_ranks_table()` (in module `cosifer.combiners.cit`), 7
`max_table()` (in module `cosifer.combiners.cit`), 7
`mean_scaled_ranks_table()` (in module `cosifer.combiners.cit`), 7
`mean_table()` (in module `cosifer.combiners.cit`), 7
`median_scaled_ranks_table()` (in module `cosifer.combiners.cit`), 7
`median_table()` (in module `cosifer.combiners.cit`), 8
`merger` (`cosifer.inferencers.jrf.JointRandomForest` attribute), 16
`method` (`cosifer.inferencers.aracne.Aracne` attribute), 13
`method` (`cosifer.inferencers.clr.CLR` attribute), 13
`method` (`cosifer.inferencers.correlation.Correlation` attribute), 14
`method` (`cosifer.inferencers.funchisq.FunChisq` attribute), 14
`method` (`cosifer.inferencers.genie3.GENIE3` attribute), 16
`method` (`cosifer.inferencers.glasso.Glasso` attribute), 16
`method` (`cosifer.inferencers.jrf.JointRandomForest` attribute), 16
`method` (`cosifer.inferencers.mrnet.MRNET` attribute), 17
`method` (`cosifer.inferencers.tigress.TIGRESS` attribute), 18
`method_selection()` (in module `cosifer.pipelines.pipeline_cli`), 19
`method_selection()` (in module `cosifer.pipelines.pipeline_gui`), 20
`min_scaled_ranks_table()` (in module `cosifer.combiners.cit`), 8
`min_table()` (in module `cosifer.combiners.cit`), 8
`MRNET` (class in `cosifer.inferencers.mrnet`), 17
`mtry` (`cosifer.inferencers.jrf.JointRandomForest` attribute), 16

N

`n` (`cosifer.collections.graph.Graph` attribute), 1, 2
`n_bootstrap` (`cosifer.inferencers.tigress.TIGRESS` attribute), 18
`n_cores` (`cosifer.inferencers.genie3.GENIE3` attribute), 15
`n_cores` (`cosifer.inferencers.tigress.TIGRESS` attribute), 18
`n_step_lars` (`cosifer.inferencers.tigress.TIGRESS` attribute), 18
`n_trees` (`cosifer.inferencers.genie3.GENIE3` attribute), 15
`name` (`cosifer.combiners.cit.CombineInteractionTables` attribute), 5
`name` (`cosifer.combiners.snf.SNF` attribute), 9
`name` (`cosifer.combiners.summa.Summa` attribute), 10
`NetworkCombiner` (class in `cosifer.combiners.network_combiner`), 9
`NetworkHandler` (class in `cosifer.handlers.network_handler`), 12
`NetworkInferencer` (class in `cosifer.inferencers.network_inferencer`), 17
`ntree` (`cosifer.inferencers.jrf.JointRandomForest` attribute), 16

P

`parameters` (`cosifer.handlers.network_handler.NetworkHandler` attribute), 12
`process_group()` (in module `cosifer.collections.interaction_table`), 4
`process_index()` (in module `cosifer.collections.interaction_table`), 5
`process_index_with_weights()` (in module `cosifer.collections.interaction_table`), 5

R

`read_data()` (in module `cosifer.utils.data`), 21
`read_gmt()` (in module `cosifer.utils.data`), 21
`regulators` (`cosifer.inferencers.genie3.GENIE3` attribute), 15
`run()` (in module `cosifer.pipelines.pipeline_cli`), 19
`run()` (in module `cosifer.pipelines.pipeline_gui`), 20
`run_combiner()` (in module `cosifer.pipelines.pipeline_cli`), 19
`run_combiner()` (in module `cosifer.pipelines.pipeline_gui`), 20
`run_inference()` (in module `cosifer.pipelines.pipeline_cli`), 20

`run_inference()` (in module `use_parallel` (`cosifer.inferencers.tigress.TIGRESS` attribute), 18
`cosifer.pipelines.pipeline_gui`), 20

S

`scale_graph()` (in module `cosifer.utils.data`), 21
`scoring` (`cosifer.inferencers.tigress.TIGRESS` attribute), 18
`set_adjacency_from_numpy()`
(`cosifer.collections.graph.Graph` method), 2
`set_adjacency_from_pandas()`
(`cosifer.collections.graph.Graph` method), 2
`set_adjacency_from_sparse()`
(`cosifer.collections.graph.Graph` method), 2
`set_labels()` (`cosifer.collections.graph.Graph` method), 2
`SNF` (class in `cosifer.combiners.snf`), 9
`sort_interaction_entities()` (in module `cosifer.inferencers.funchisq`), 15
`Summa` (class in `cosifer.combiners.summa`), 10
`summa_object` (`cosifer.combiners.summa.Summa` attribute), 10
`summa_scores_table()` (in module `cosifer.combiners.summa`), 10

T

`targets` (`cosifer.inferencers.genie3.GENIE3` attribute), 15
`tf_list` (`cosifer.inferencers.tigress.TIGRESS` attribute), 18
`TIGRESS` (class in `cosifer.inferencers.tigress`), 17
`to_edge_list()` (`cosifer.collections.interaction_table.InteractionTable` method), 3
`to_graph()` (`cosifer.collections.interaction_table.InteractionTable` method), 3
`to_interaction_table()`
(`cosifer.collections.graph.Graph` method), 2
`trained` (`cosifer.combiners.network_combiner.NetworkCombiner` attribute), 9
`trained` (`cosifer.handlers.network_handler.NetworkHandler` attribute), 12, 13
`trained` (`cosifer.inferencers.network_inferencer.NetworkInferencer` attribute), 17
`tree_method` (`cosifer.inferencers.genie3.GENIE3` attribute), 15

U

`undirected` (`cosifer.collections.graph.Graph` attribute), 1
`undirected` (`cosifer.inferencers.funchisq.FunChisq` attribute), 15

V

`verbose` (`cosifer.inferencers.genie3.GENIE3` attribute), 16
`verbose` (`cosifer.inferencers.tigress.TIGRESS` attribute), 18