graphical$_{m}odel_{l}earning Documentation$

Chandler Squires

Jun 14, 2022

CONTENTS

CHAPTER

ONE

LEARNING FROM OBSERVATIONAL DATA

gsp(nodes, ci_tester[, depth, nruns, ...])	Estimate the Markov equivalence class of a DAG using the Greedy Sparsest Permutations (GSP) algorithm.
pcalg(nodes[, ci_tester, skel, sepset, ...])	Use the PC (Peters-Clark) algorithm to estimate the Markov equivalence class of the data-generating DAG.

1.1 graphical_model_learning.gsp

graphical_model_learning.**gsp**(*nodes: set, ci_tester: ~conditional_independence.ci_tests.ci_tester.CI_Tester, depth: ~typing.Optional[int] = 4, nruns: int = 5, verbose: bool = False, initial_undirected: ~typing.Optional[~typing.Union[str, ~graphical_models.classes.undirected.undirected_graph.UndirectedGraph]] = 'threshold', initial_permutations: ~typing.Optional[~typing.List] = None, fixed_orders={}, fixed_adjacencies={}, fixed_gaps={}, use_lowest=True, max_iters=inf, factor=2, progress_bar=False, summarize=False) -> (<class 'graphical_models.classes.dags.dag.DAG'>, typing.List[typing.List[typing.Dict]]*)

Estimate the Markov equivalence class of a DAG using the Greedy Sparsest Permutations (GSP) algorithm.

Parameters

- **nodes** – Labels of nodes in the graph.
- **ci_tester** – A conditional independence tester, which has a method is_ci taking two sets A and B, and a conditioning set C, and returns True/False.
- **depth** – Maximum depth in depth-first search. Use None for infinite search depth.
- **nruns** – Number of runs of the algorithm. Each run starts at a random permutation and the sparsest DAG from all runs is returned.
- **verbose** – TODO
- **initial_undirected** – Option to find the starting permutation by using the minimum degree algorithm on an undirected graph that is Markov to the data. You can provide the undirected graph yourself, use the default 'threshold' to do simple thresholding on the partial correlation matrix, or select 'None' to start at a random permutation.
- **initial_permutations** – A list of initial permutations with which to start the algorithm. This option is helpful when there is background knowledge on orders. This option is mutually exclusive with initial_undirected.
- **fixed_orders** – Tuples (i, j) where i is known to come before j.

- **fixed_adjacencies** – Tuples (i, j) where i and j are known to be adjacent.
- **fixed_gaps** – Tuples (i, j) where i and j are known to be non-adjacent.

See also:

pcalg, *igsp*, *unknown_target_igsp*

Return type
(est_dag, summaries)

1.2 graphical_model_learning.pcalg

graphical_model_learning.**pcalg**(*nodes*, *ci_tester: Optional[CI_Tester] = None*, *skel=None*, *sepset=None*, *solve_conflict: bool = False*, *max_cond_set: Optional[int] = None*, *verbose: bool = False*) → PDAG

Use the PC (Peters-Clark) algorithm to estimate the Markov equivalence class of the data-generating DAG.

Parameters

- **nodes** – Labels of nodes in the graph.
- **ci_tester** – A conditional independence tester, which has a method is_ci taking two sets A and B, and a conditioning set C, and returns True/False.
- **skel** – An estimated skeleton. If not provided, uses the *skeleton* method to estimate.
- **sepset** – The separating sets for non-adjacent nodes in the estimated skeleton.
- **solve_conflict** – If False, any disagreements on v-structures are simply overwritten. If True, allow both orientations (represented by a bidirected edge).
- **verbose** – If True, print decisions made by the algorithm.

See also:

gsp

Return type
est_dag

CHAPTER

TWO

LEARNING FROM INTERVENTIONAL DATA

igsp(setting_list, nodes, ci_tester, ...[, ...])	TODO
unknown_target_igsp(setting_list, nodes, ...)	Use the Unknown Target Interventional Greedy Sparsest Permutation algorithm to estimate a DAG in the I-MEC of the data-generating DAG.

2.1 graphical_model_learning.igsp

graphical_model_learning.**igsp**(*setting_list: List[Dict]*, *nodes: set*, *ci_tester: CI_Tester*, *invariance_tester: InvarianceTester*, *depth: Optional[int] = 4*, *nruns: int = 5*, *initial_undirected: Optional[Union[str, UndirectedGraph]] = 'threshold'*, *initial_permutations: Optional[List] = None*, *verbose: bool = False*)

TODO

Parameters
TODO –

Examples

TODO

2.2 graphical_model_learning.unknown_target_igsp

graphical_model_learning.**unknown_target_igsp**(*setting_list: ~typing.List[~typing.Dict]*, *nodes: set*, *ci_tester: ~conditional_independence.ci_tests.ci_tester.CI_Tester*, *invariance_tester: ~conditional_independence.invariance_tests.invariance_tester.InvarianceTester*, *depth: ~typing.Optional[int] = 4*, *nruns: int = 5*, *initial_undirected: ~typing.Optional[~typing.Union[str, ~graphical_models.classes.undirected.undirected_graph.UndirectedGraph]] = 'threshold'*, *initial_permutations: ~typing.Optional[~typing.List] = None*, *verbose: bool = False*, *use_lowest=True*, *tup_score=True*, *no_targets=False*) -> (<class 'graphical_models.classes.dags.dag.DAG'>, *typing.List[typing.Set[int]]*)

Use the Unknown Target Interventional Greedy Sparsest Permutation algorithm to estimate a DAG in the I-MEC of the data-generating DAG.

Parameters

- **setting_list** – A list of dictionaries that provide meta-information about each non-observational setting.
- **nodes** – Nodes in the graph.
- **ci_tester** – A conditional independence tester object, which has a method is_ci taking two sets A and B, and a conditioning set C, and returns True/False.
- **invariance_tester** – An invariance tester object, which has a method is_invariant taking a node, two settings, and a conditioning set C, and returns True/False.
- **depth** – Maximum depth in depth-first search. Use None for infinite search depth.
- **nruns** – Number of runs of the algorithm. Each run starts at a random permutation and the sparsest DAG from all runs is returned.
- **initial_undirected** – Option to find the starting permutation by using the minimum degree algorithm on an undirected graph that is Markov to the data. You can provide the undirected graph yourself, use the default 'threshold' to do simple thresholding on the partial correlation matrix, or select 'None' to start at a random permutation.
- **initial_permutations** – A list of initial permutations with which to start the algorithm. This option is helpful when there is background knowledge on orders. This option is mutually exclusive with initial_undirected.
- **no_targets** – if True, leave out information on known intervention targets.

CHAPTER

THREE

BAYESIAN METHODS

CHAPTER

FOUR

ACTIVE LEARNING

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

INDEX