Package ‘causaloptim’

March 25, 2020

Encoding UTF-8

Type Package

Title An Interface to Specify Causal Graphs and Compute Bounds on Causal Effects

Version 0.6.5

Date 2020-03-24

Maintainer Michael C Sachs <sachsmc@gmail.com>

Description When causal quantities are not identifiable from the observed data, it still may be possible to bound these quantities using the observed data. We outline a class of problems for which the derivation of tight bounds is always a linear programming problem and can therefore, at least theoretically, be solved using a symbolic linear optimizer. We extend and generalize the approach of Balke and Pearl (1994) <doi:10.1016/B978-1-55860-332-5.50011-0> and we provide a user friendly graphical interface for setting up such problems via directed acyclic graphs (DAG), which only allow for problems within this class to be depicted. The user can then define linear constraints to further refine their assumptions to meet their specific problem, and then specify a causal query using a text interface. The program converts this user defined DAG, query, and constraints, and returns tight bounds. The bounds can be converted to R functions to evaluate them for specific datasets, and to latex code for publication. The methods and proofs of tightness and validity of the bounds are described in a preprint by Sachs, Gabriel, and Sjölander (2020) <https://sachsmc.github.io/causaloptim/articles/CausalBoundsMethods.pdf>.

License MIT + file LICENSE

Imports methods, Rcpp (>= 1.0.1), shiny

Depends igraph

LinkingTo Rcpp

RoxygenNote 7.0.2

Suggests knitr, rmarkdown

VignetteBuilder knitr

URL https://github.com/sachsmc/causaloptim

BugReports https://github.com/sachsmc/causaloptim/issues
**R topics documented:**

causaloptim-package ......................................................... 2
analyze_graph ................................................................. 3
const.to.sets ................................................................. 4
expand_cond ................................................................. 4
find_cycles ................................................................. 5
interpret_bounds ............................................................ 5
latex_bounds ................................................................. 6
optimize_effect ............................................................ 7
parse_constraints ............................................................. 7
parse_effect ................................................................. 8
pastestar ................................................................. 8
plot_graphres ................................................................. 9
reduce.sets ................................................................. 9
shortentxt ................................................................. 10
simulate_bounds ............................................................. 10
specify_graph ............................................................... 11
symb.subtract ............................................................... 11

Index 12

---

**causaloptim-package**

*An Interface to Specify Causal Graphs and Compute Bounds on Causal Effects*

**Description**

Specify causal graphs using a visual interactive interface and then analyze them and compute symbolic bounds for the causal effects in terms of the observable parameters.

**Details**

Run the shiny app by results <- specify_graph(). See detailed instructions in the vignette browseVignettes("causaloptim").
**analyze_graph**

**Author(s)**

Michael C Sachs, Arvid Sjölander, Alexander Balke, Colorado Reed, and Erin Gabriel

Maintainer:

Michael C Sachs <sachsmc at gmail.com>

**References**


**See Also**

browseVignettes('causaloim')

---

**analyze_graph**

*Analyze the causal graph to determine constraints and objective*

**Description**

The graph must contain edge attributes named "leftside" and "lrconnect" that take values 0 and 1. Only one edge may have a value 1 for lrconnect. The shiny app returns a graph in this format.

**Usage**

analyze_graph(graph, constraints, effectt)

**Arguments**

- **graph**: An *aaaigraph-package* object that represents a directed acyclic graph
- **constraints**: A vector of character strings that represent the constraints
- **effectt**: A character string that represents the causal effect of interest

**Value**

A list with the following components. This list can be passed to optimize_effect which interfaces with Balke's code:

- **variables**: Character vector of variable names of potential outcomes, these start with 'q' to match Balke's notation
- **parameters**: Character vector of parameter names of observed probabilities, these start with 'p' to match Balke's notation
- **constraints**: Character vector of parsed constraints
- **objective**: Character string defining the objective to be optimized in terms of the variables
\textbf{p.vals} Matrix of all possible values of the observed data vector, corresponding to the list of parameters.

\textbf{q.vals} Matrix of all possible values of the response function form of the potential outcomes, corresponding to the list of variables.

**Examples**

```r
### confounded exposure and outcome
b <- igraph::graph_from_literal(X <- Y, Ur <- X, Ur <- Y)
V(b)\$leftside <- c(0,0,0)
V(b)\$latent <- c(0,0,1)
E(b)$rlconnect <- E(b)$edge.monotone <- c(0,0,0)
analyze_graph(b, constraints = NULL, effectt = "p(Y(X = 1) = 1) - p(Y(X = 0) = 1)")
```

---

**const.to.sets**

*Translate lists of constraints to lists of vectors*

**Description**

Translate lists of constraints to lists of vectors

**Usage**

```r
const.to.sets(constr, objterms)
```

**Arguments**

- \textbf{constr} List of constraint terms as character strings
- \textbf{objterms} Vector of terms in the objective function

---

**expand_cond**

*Expand potential outcome conditions*

**Description**

Expand potential outcome conditions

**Usage**

```r
expand_cond(cond, obsnames)
```

**Arguments**

- \textbf{cond} Text string of the condition
- \textbf{obsnames} Vector of names of observed variables
find_cycles

**Description**
Find cycles in a graph

**Usage**
find_cycles(g)

**Arguments**
g an igraph object

**Value**
A list of vectors of integers, indicating the vertex sequences for the cycles found in the graph

interpret_bounds

**Description**
Convert bounds string to a function

**Usage**
interpret_bounds(bounds, parameters)

**Arguments**
bounds The bounds element as returned by optimize_effect
parameters Character vector defining parameters, as returned by analyze_graph

**Value**
A function that takes arguments for the parameters, i.e., the observed probabilities and returns a vector of length 2: the lower bound and the upper bound.
Examples

b <- graph_from_literal(X -+ Y, Ur -+ X, Ur -+ Y)
V(b)$leftside <- c(0,0,0)
V(b)$latent <- c(0,0,1)
E(b)$rlconnect <- E(b)$edge.monotone <- c(0, 0, 0)
obj <- analyze_graph(b, constraints = NULL, effectt = "p(Y(X = 1) = 1) - p(Y(X = 0) = 1)")
bounds <- optimize_effect(obj)
bounds_func <- interpret_bounds(bounds$bounds, obj$parameters)
bounds_func(.1, .1, .4, .3)
# vectorized
do.call(bounds_func, lapply(1:4, function(i) runif(5)))

latex_bounds

\textit{latex bounds} \hspace{1cm} \textit{Latex bounds equations}

Description

Latex bounds equations

Usage

latex_bounds(bounds, parameters, prob.sym = "P")

Arguments

bounds Vector of bounds as returned by \texttt{optimize_effect}
parameters The parameters object as returned by \texttt{analyze_graph}
prob.sym Symbol to use for probability statements in latex, usually "P" or "pr"

Value

A character string with latex code for the bounds

Examples

b <- graph_from_literal(X -+ Y, Ur -+ X, Ur -+ Y)
V(b)$leftside <- c(0,0,0)
V(b)$latent <- c(0,0,1)
E(b)$rlconnect <- E(b)$edge.monotone <- c(0, 0, 0)
obj <- analyze_graph(b, constraints = NULL, effectt = "p(Y(X = 1) = 1) - p(Y(X = 0) = 1)")
bounds <- optimize_effect(obj)
l latex_bounds(bounds$bounds, obj$parameters)
l latex_bounds(bounds$bounds, obj$parameters, "Pr")
**optimize_effect**

_Run the Balke optimizer_

**Description**

Given a object with the linear programming problem set up, compute the bounds using the c++ code developed by Alex Balke. Bounds are returned as text but can be converted to R functions using `interpret_bounds`, or latex code using `latex_bounds`.

**Usage**

```r
optimize_effect(obj)
```

**Arguments**

- **obj**
  
  Object as returned by `analyze_graph`

**Value**

An object of class "balkebound" that contains the bounds and logs as character strings

**Examples**

```r
b <- graph_from_literal(X -+ Y, Ur -+ X, Ur -+ Y)
V(b)$leftside <- c(0,0,0)
V(b)$latent <- c(0,0,1)
E(b)$rlconnect <- E(b)$edge.monotone <- c(0, 0, 0)
obj <- analyze_graph(b, constraints = NULL, effectt = "p(Y(X = 1) = 1) - p(Y(X = 0) = 1)")
optimize_effect(obj)
```

---

**parse_constraints**

_Parse text that defines a the constraints_

**Description**

Parse text that defines a the constraints

**Usage**

```r
parse_constraints(constraints, obsnames)
```

**Arguments**

- **constraints**
  
  A list of character strings

- **obsnames**
  
  Vector of names of the observed variables in the graph
Value
A data frame with columns indicating the variables being constrained, what the values of their parents are for the constraints, and the operator defining the constraint (equality or inequalities).

parse_effect
Parse text that defines a causal effect

Description
Parse text that defines a causal effect

Usage
parse_effect(text)

Arguments
text Character string

Value
A nested list that contains the following components:

vars For each element of the causal query, this indicates potential outcomes as names of the list elements, the variables that they depend on, and the values that any variables are being fixed to.

oper The vector of operators (addition or subtraction) that combine the terms of the causal query.

values The values that the potential outcomes are set to in the query (0 or 1).

pcheck List of logicals for each element of the query that are TRUE if the element is a potential outcome and FALSE if it is an observational quantity.

pastestar
Paste with asterisk sep

Description
Paste with asterisk sep

Usage
pastestar(...)
**plot_graphres**

Plot the analyzed graph object

**Description**

Special plotting method for igraphs of this type

**Usage**

plot_graphres(graphres)

**Arguments**

- graphres: an igraph object

**Value**

None

---

**reduce.sets**

Algebraically reduce sets

**Description**

Identifies and reduces redundant variables

**Usage**

reduce.sets(sets)

**Arguments**

- sets: List of constraints as sets of variables
shortentxt

Description
Shorten strings to 80 characters wide

Usage
shortentxt(x)

Arguments

x String

simulate_bounds

Description
Run a simple simulation based on the bounds. For each simulation, sample the set of counterfactual probabilities from a uniform distribution, translate into a multinomial distribution, and then compute the objective and the bounds in terms of the observable variables.

Usage
simulate_bounds(obj, bounds, nsim = 1000)

Arguments

obj Object as returned by analyze_graph
bounds Object as returned by optimize_effect
nsim Number of simulation replicates

Value
A data frame with columns: objective, bound.lower, bound.upper

Examples

b <- graph_from_literal(X -+ Y, Ur -+ X, Ur -+ Y)
V(b)$leftside <- c(0,0,0)
V(b)$latent <- c(0,0,1)
E(b)$rlconnect <- E(b)$edge.monotone <- c(0, 0, 0)
obj <- analyze_graph(b, constraints = NULL, effectt = "p(Y(X = 1) = 1) - p(Y(X = 0) = 1)"
bounds <- optimize_effect(obj)
simulate_bounds(obj, bounds, nsim = 5)
**specify_graph**

*Shiny interface to specify network structure and compute bounds*

**Description**

This launches the Shiny interface in the system’s default web browser. The results of the computation will be displayed in the browser, but they can also be returned to the R session by assigning the result of the function call to an object. See below for information on what is returned.

**Usage**

```r
specify_graph()
```

**Value**

If the button "Exit and return graph object" is clicked, then only the graph is returned as an `aaaigraph-package` object.

If the bounds are computed and the button "Exit and return objects to R" is clicked, then a list is returned with the following elements:

- `graphres` The graph as drawn and interpreted, an `aaaigraph-package` object.
- `obj` The objective and all necessary supporting information. This object is documented in `analyze_graph`. This can be passed directly to `optimize_effect`.
- `bounds.obs` Object of class 'balkebound' as returned by `optimize_effect`.
- `constraints` Character vector of the specified constraints. NULL if no constraints.
- `effect` Text describing the causal effect of interest.
- `boundsFunction` Function that takes parameters (observed probabilities) as arguments, and returns a vector of length 2 for the lower and upper bounds.

**symb.subtract**

*Symbolic subtraction*

**Description**

Like `setdiff` but doesn’t remove duplicates `x1 - x2`

**Usage**

```r
symb.subtract(x1, x2)
```

**Arguments**

- `x1` First term (subtract from)
- `x2` Second term (subtract)
Index

aaa-igraph-package, 3, 11
analyze_graph, 3, 5–7, 10, 11
causaloptim (causaloptim-package), 2
causaloptim-package, 2
const.to.sets, 4
expand_cond, 4
find_cycles, 5
interpret_bounds, 5, 7
latex_bounds, 6, 7
optimize_effect, 3, 5, 6, 7, 10, 11
parse_constraints, 7
parse_effect, 8
pastestar, 8
plot_graphres, 9
reduce.sets, 9
shortentxt, 10
simulate_bounds, 10
specify_graph, 11
symb.subtract, 11