General Purpose SAT-Solvers for Causal Discovery

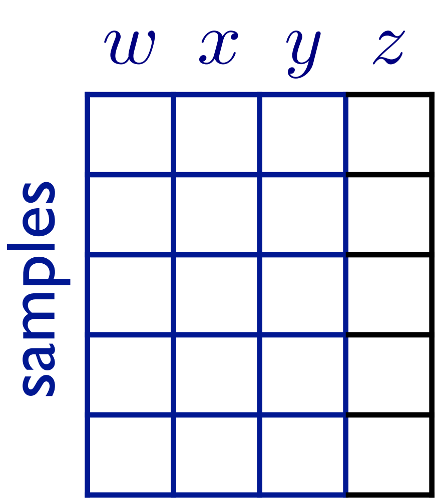
Frederick Eberhardt
Caltech

[joint work with Antti Hyttinen and Matti Jarvisalo]

CCD 1/21/2016

Causal Discovery

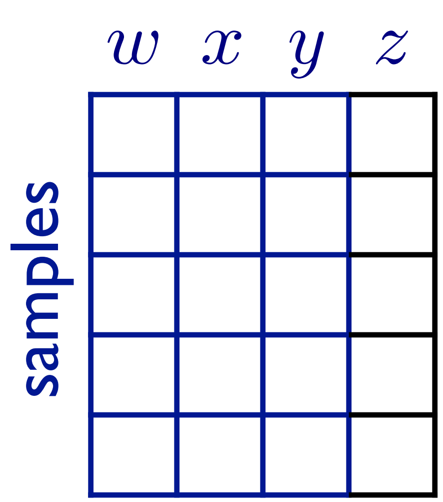
data
sample



Causal Discovery

assumptions, e.g.
- causal Markov
- causal faithfulness
- functional form
- etc.

data sample



inference algorithm

Causal Discovery

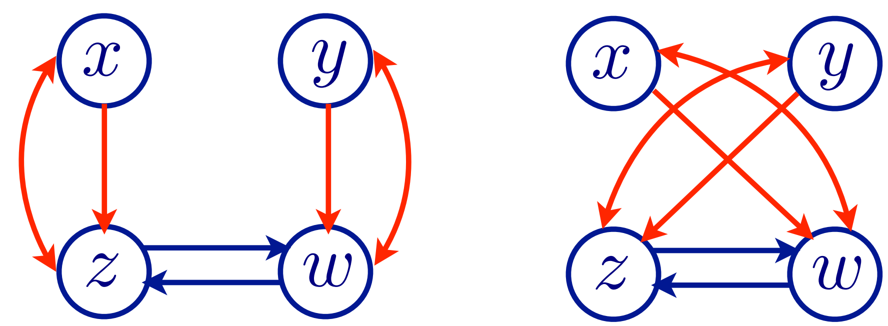
data sample

	w	x	y	z
samples				

assumptions, e.g.
- causal Markov
- causal faithfulness
- functional form
- etc.

inference algorithm

equivalence classes



model specifications

	w	x	y	z
w	0	0	?	a
x	0	0	0	0
y	0	0	0	0
z	b	?	?	0

direct edges

	w	x	y	z
w	0	0	?	?
x	0	0	0	?
y	?	0	0	0
z	?	?	0	0

confounders

General Model Space

assumption / algorithm
Markov
faithfulness
causal sufficiency
acyclicity
parametric assumption

General Model Space

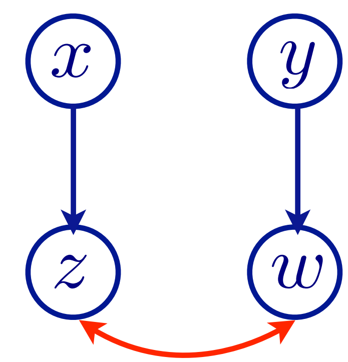
assumption / algorithm	PC / GES	FCI	CCD	LiNGaM	non-linear additive noise
Markov	✓	✓	✓	✓	✓
faithfulness	✓	✓	✓	✗	minimality
causal sufficiency	✓	✗	✓	✓	✓
acyclicity	✓	✓	✗	✓	✓
parametric assumption	✗	✗	✗	linear non-Gaussian	non-linear additive noise

General Model Space

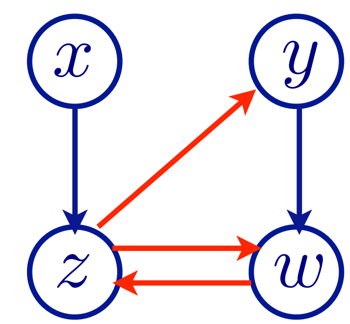
assumption / algorithm	PC / GES	FCI	CCD	LiNGaM	non-linear additive noise
Markov	✓	✓	✓	✓	✓
faithfulness	✓	✓	✓	✗	minimality
causal sufficiency	✓	✗	✓	✓	✓
acyclicity	✓	✓	✗	✓	✓
parametric assumption	✗	✗	✗	linear non-Gaussian	non-linear additive noise

General Model Space

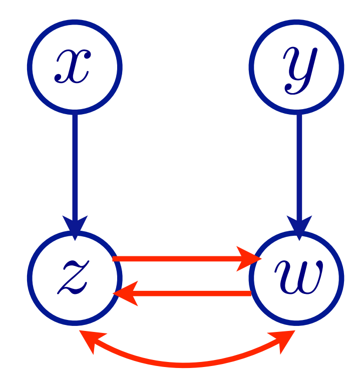
assumption / algorithm	PC / GES	FCI	CCD	LiNGaM	non-linear additive noise
Markov	✓	✓	✓	✓	✓
faithfulness	✓	✓	✓	✗	minimality
causal sufficiency	✓	✗	✓	✓	✓
acyclicity	✓	✓	✗	✓	✓
parametric assumption	✗	✗	✗	linear non-Gaussian	non-linear additive noise



w/ latents



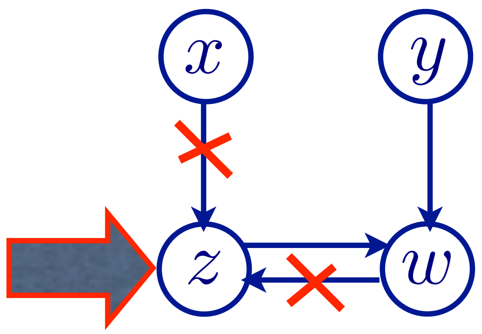
w/ cycles



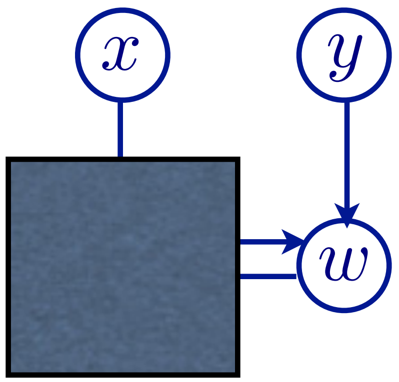
w/ both

Combining Experiment and Observation

experiment

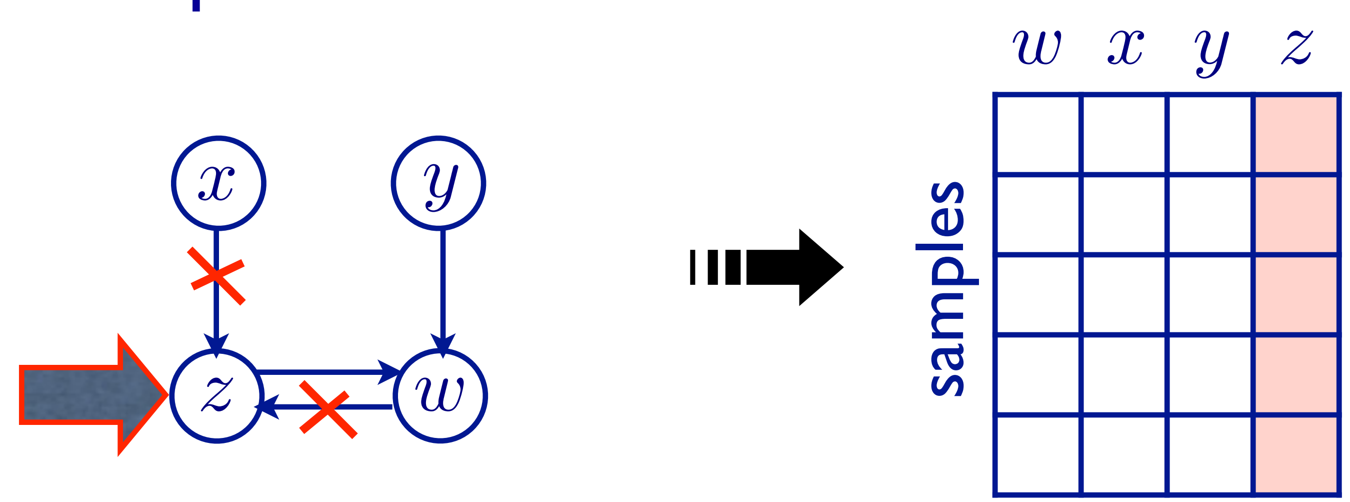


observational study

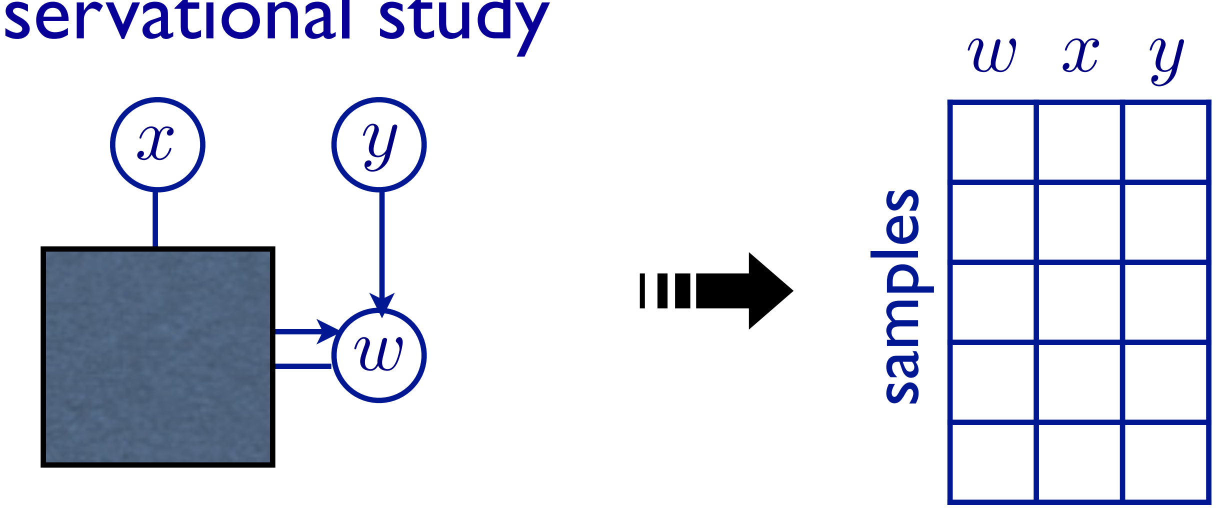


Combining Experiment and Observation

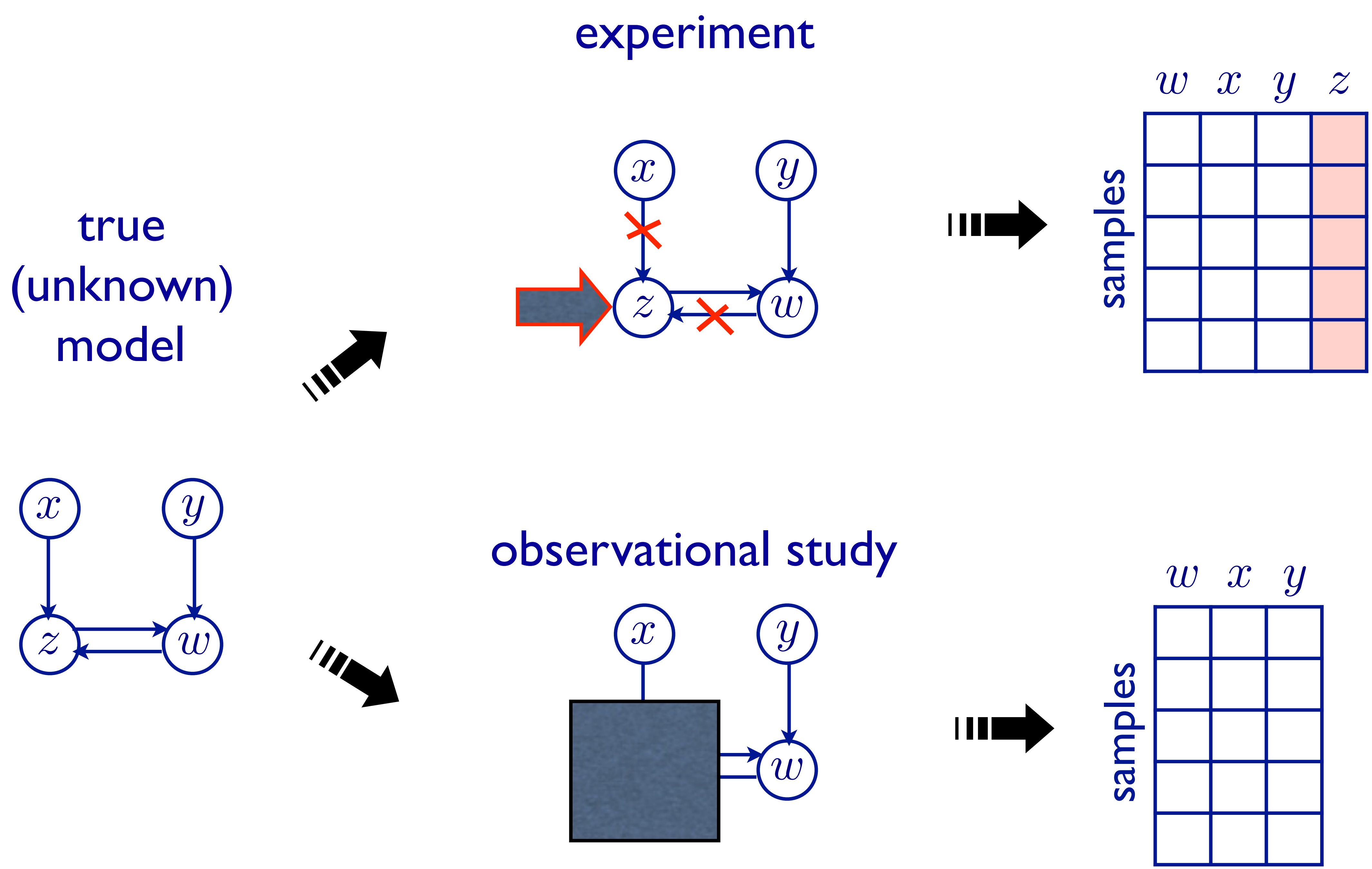
experiment



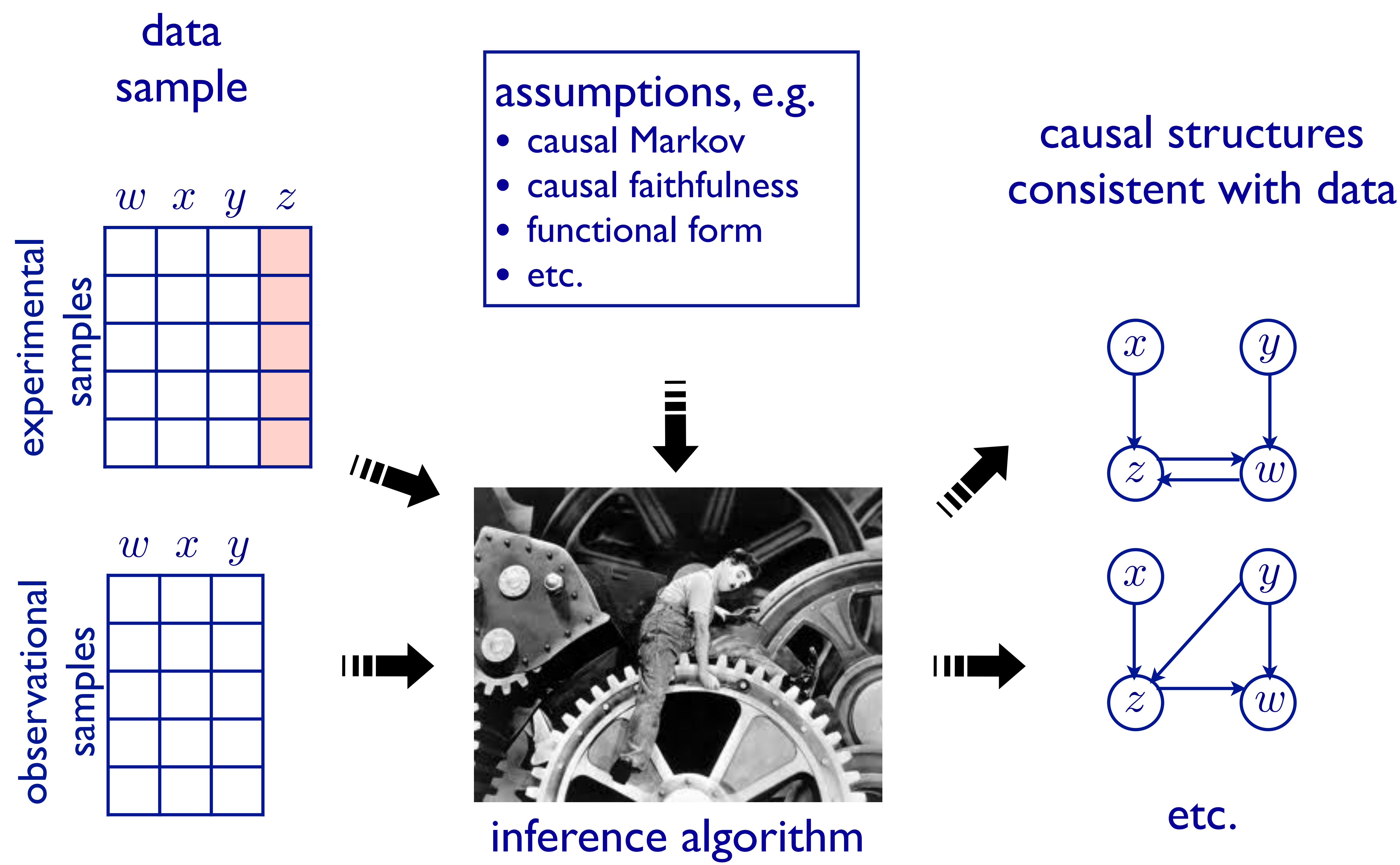
observational study



Combining Experiment and Observation



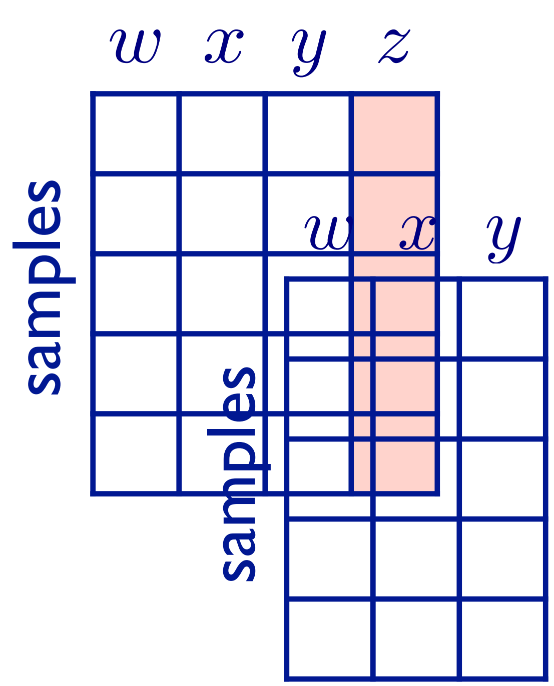
Causal Discovery



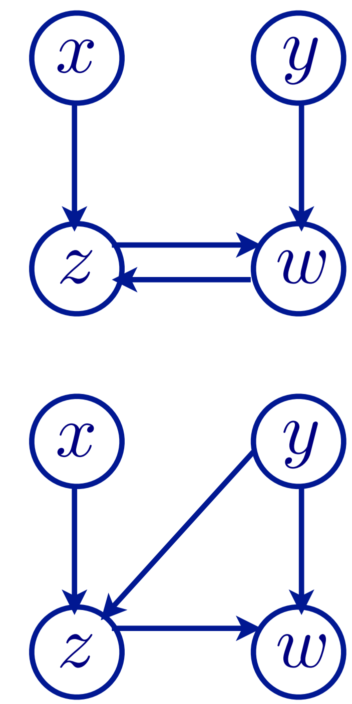
Causal Discovery

assumptions, e.g.
- causal Markov
- causal faithfulness
- functional form
- etc.

causal structures consistent with data



inference algorithm



etc.

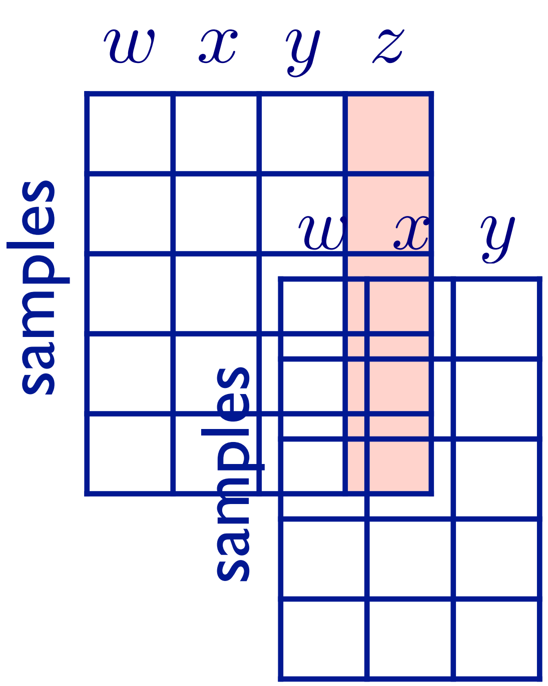
Causal Discovery

background knowledge

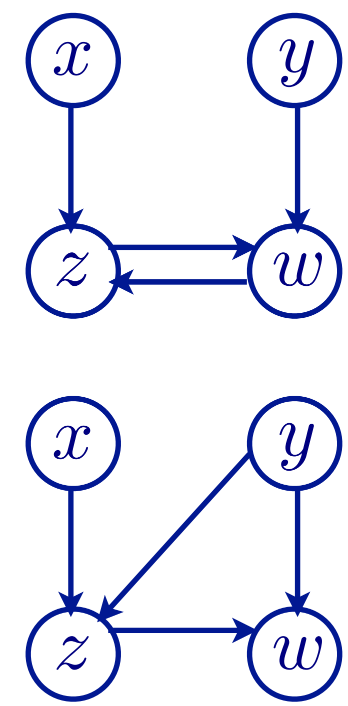
assumptions, e.g.

- causal Markov
- causal faithfulness
- functional form
- etc.

causal structures consistent with data



inference algorithm



etc.

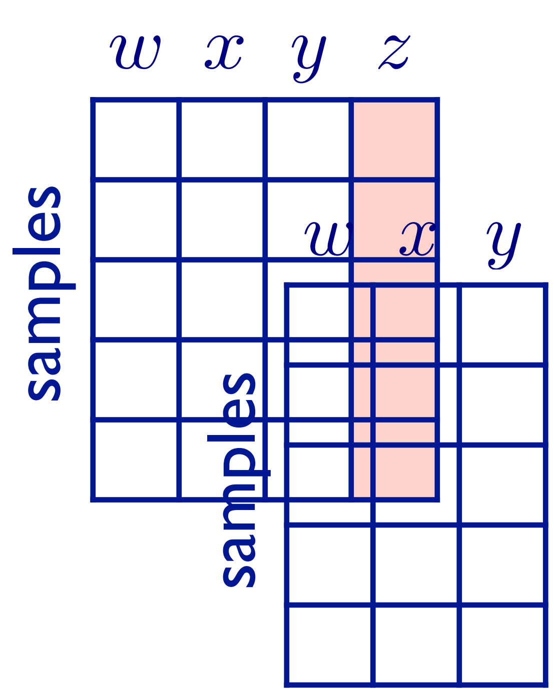
Causal Discovery

background knowledge

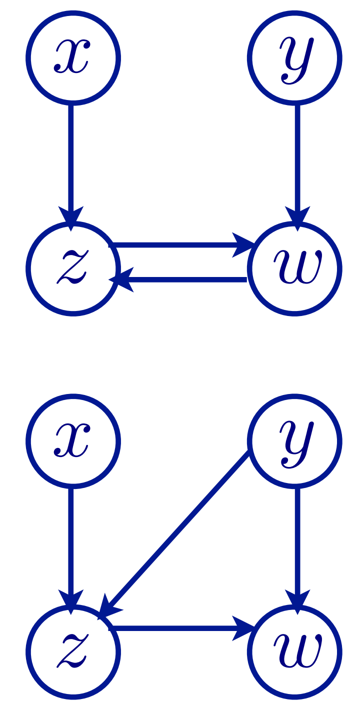
- edge presences/ absences

assumptions, e.g.
- causal Markov
- causal faithfulness
- functional form
- etc.

causal structures consistent with data



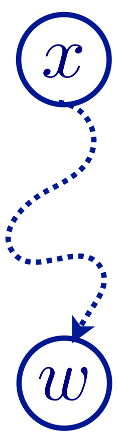
inference algorithm



etc.

Causal Discovery

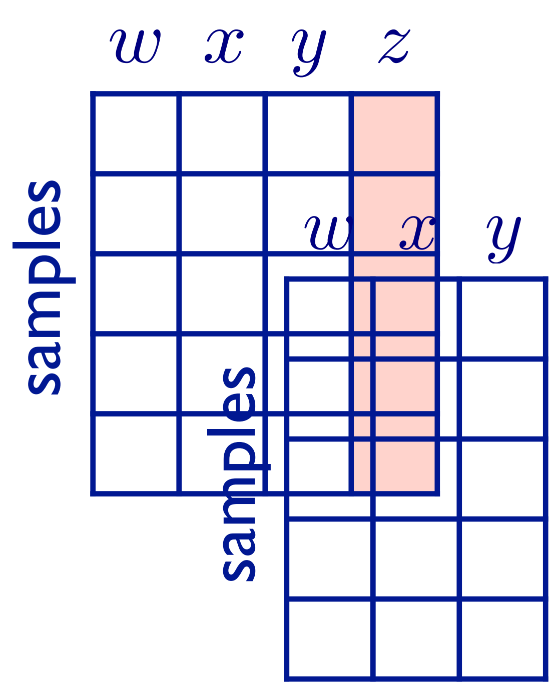
background knowledge



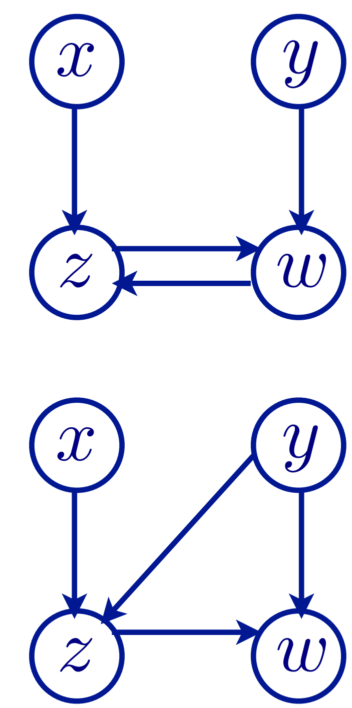
- edge presences/ absences
- pathways

assumptions, e.g.
- causal Markov
- causal faithfulness
- functional form
- etc.

causal structures consistent with data



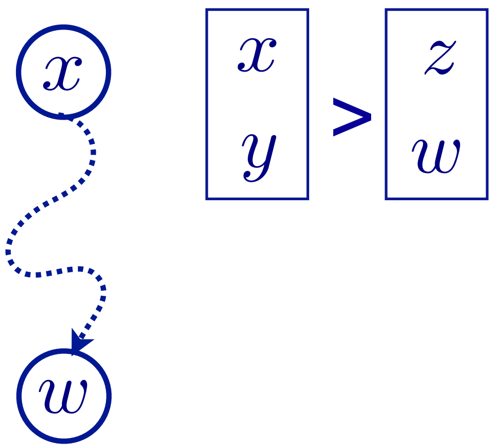
inference algorithm



etc.

Causal Discovery

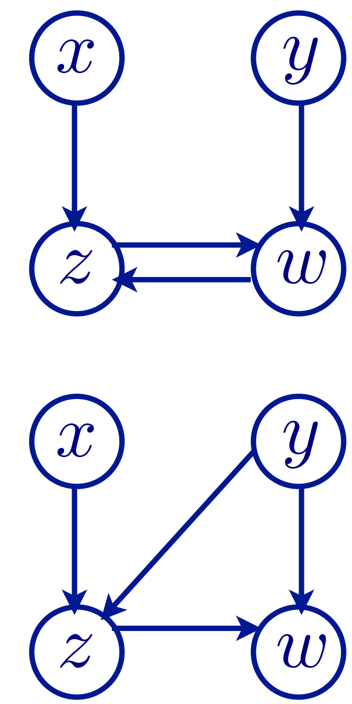
background knowledge



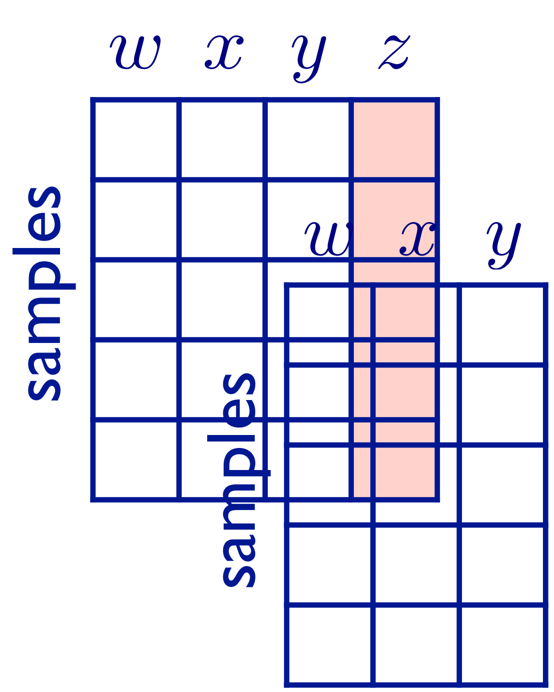
- edge presences/ absences
- pathways
- tier orderings

assumptions, e.g.
- causal Markov
- causal faithfulness
- functional form
- etc.

causal structures consistent with data



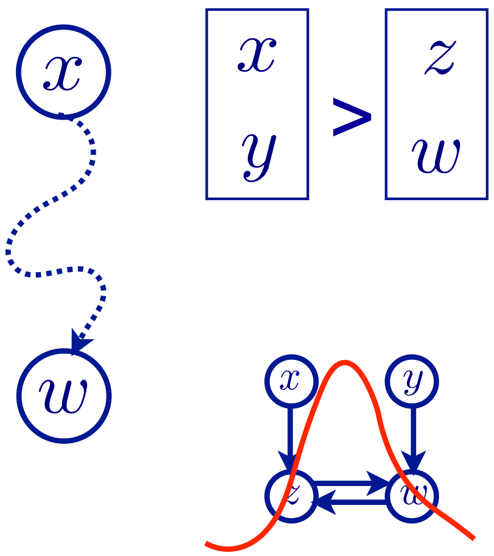
etc.



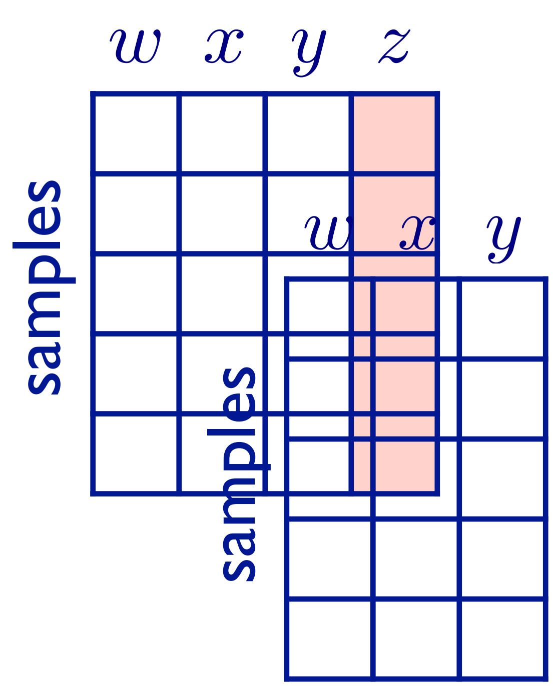
inference algorithm

Causal Discovery

background knowledge



- edge presences/ absences
- pathways
- tier orderings
- "priors"

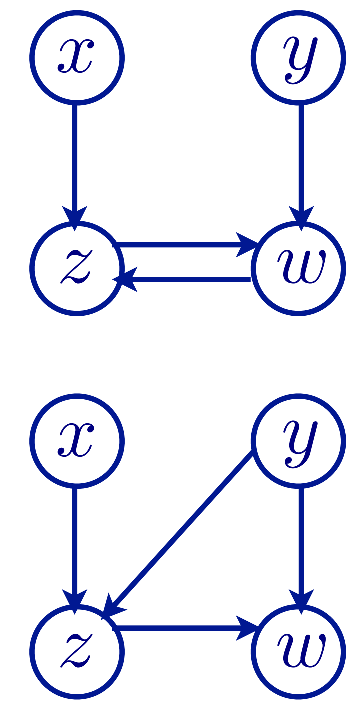


inference algorithm

assumptions, e.g.

- causal Markov
- causal faithfulness
- functional form
- etc.

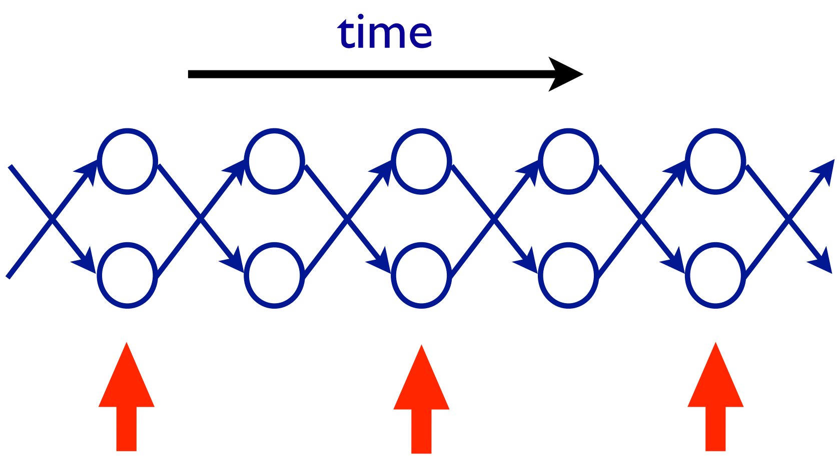
causal structures consistent with data



etc.

Settings: examples

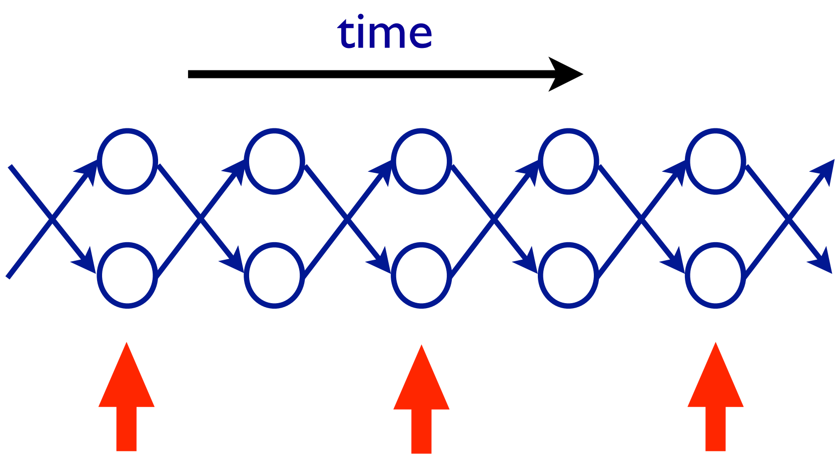
subsampled time series

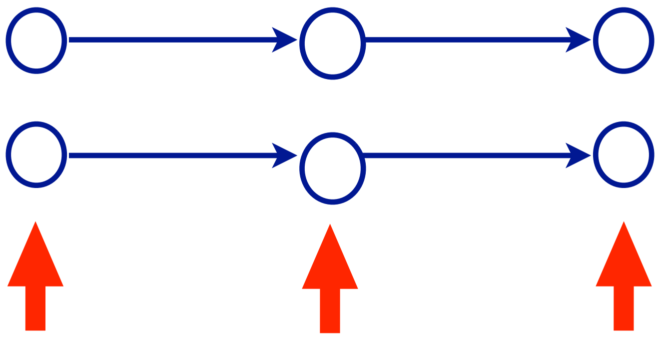


(cf. work by Plis & Danks)

Settings: examples

subsampled time series

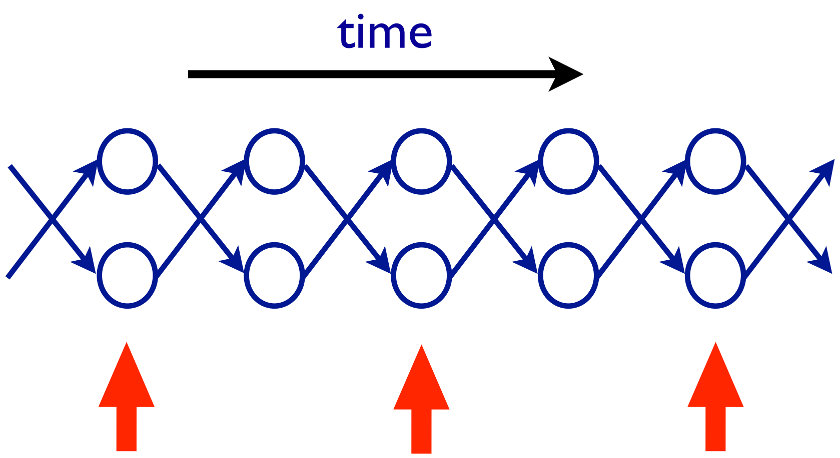


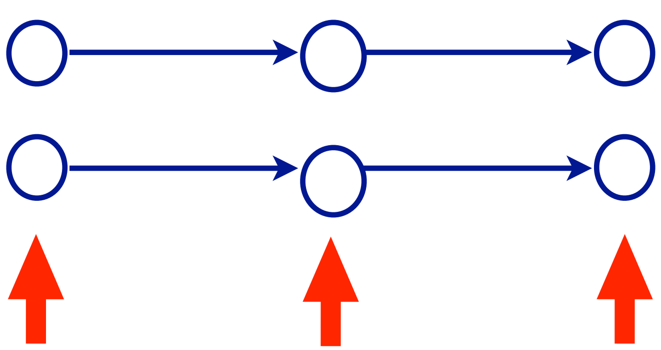


(cf. work by Plis & Danks)

Settings: examples

subsampled time series



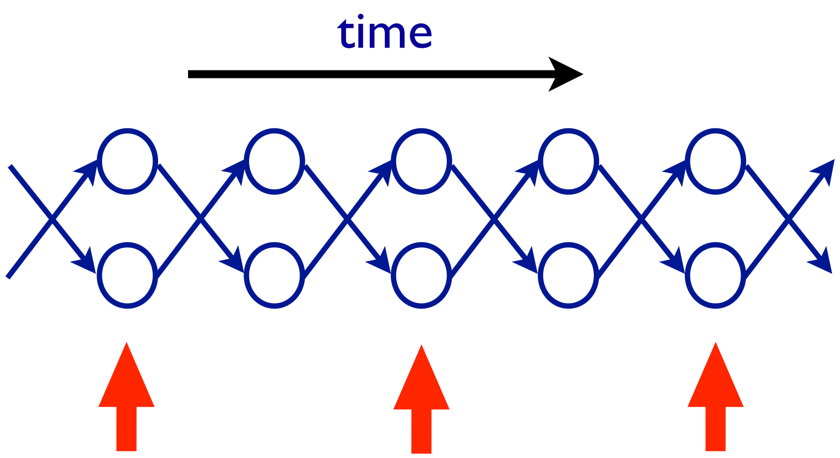


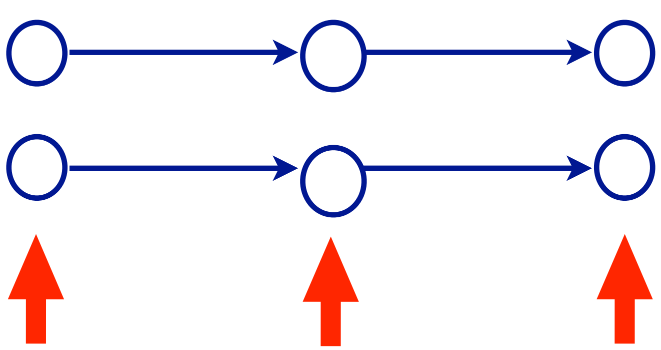
(cf. work by Plis & Danks)

inference algorithm

Settings: examples

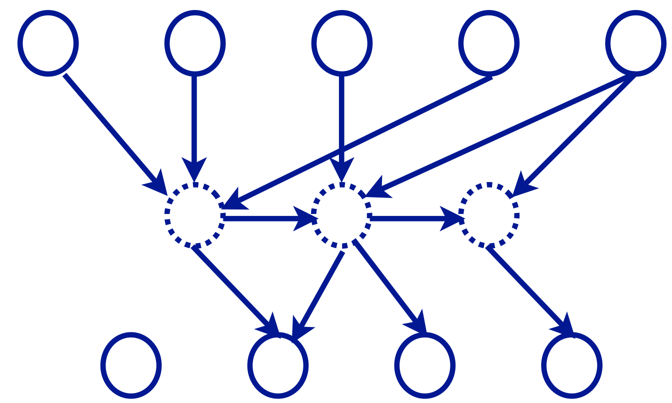
subsampled time series





(cf. work by Plis & Danks)

biological settings



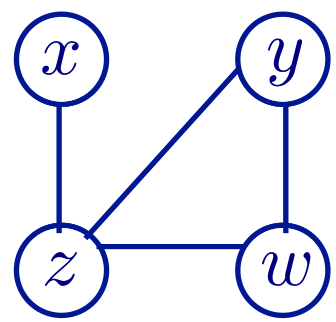
(cf. work by Murray-Watters & Glymour)

inference algorithm

Causal Effects

inference algorithm

Causal Effects

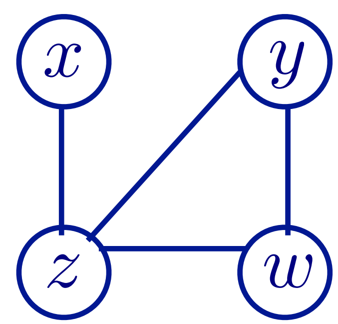


inference algorithm

equivalence class

Causal Effects

inference algorithm



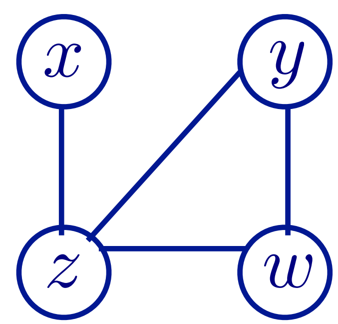
equivalence class

$$P(y|do(w))$$

causal effect

Causal Effects

inference algorithm



equivalence class

$$P(y|do(w))$$

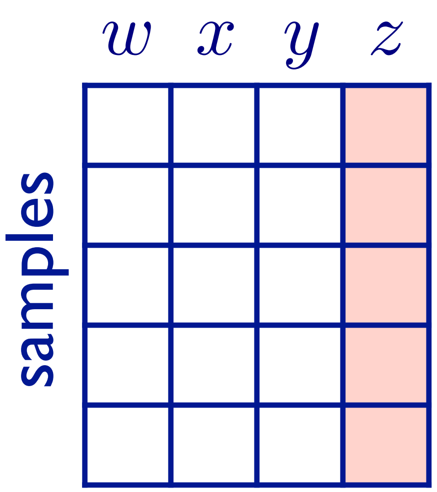
causal effect

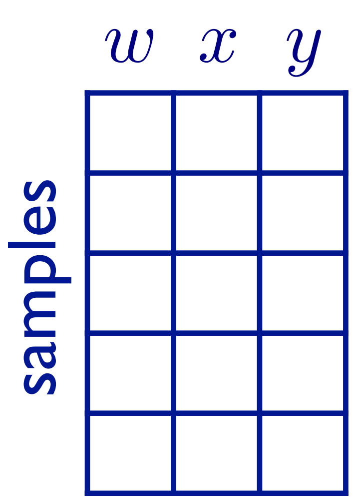
How to apply the do-calculus in settings when the causal structure is underdetermined?

High-Level

High-Level

data
sample

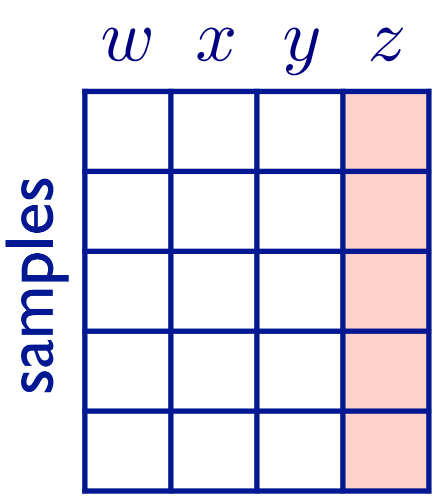


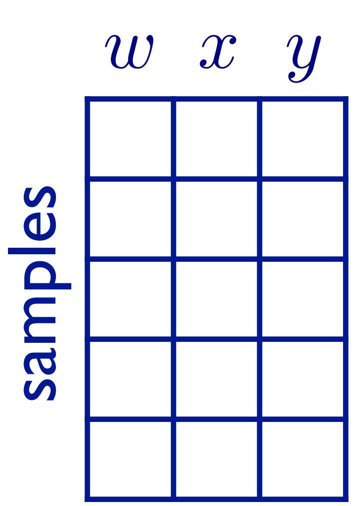


High-Level

data
sample

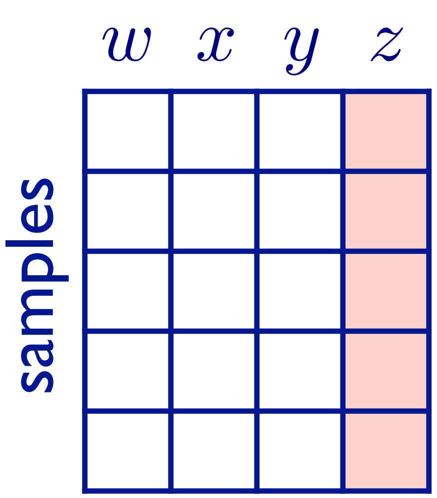
assumptions, e.g.
- causal Markov
- causal faithfulness
- etc.

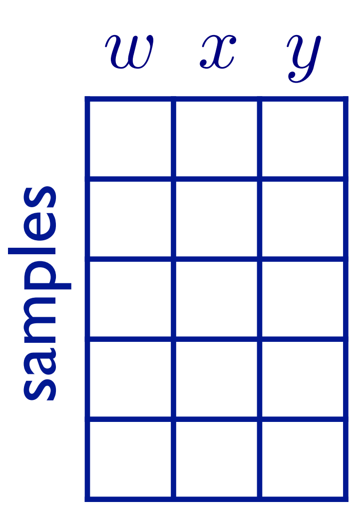




High-Level

data
sample





assumptions, e.g.
- causal Markov
- causal faithfulness
- etc.

background knowledge, e.g.
- pathways
- tier ordering
- “priors”
- etc.

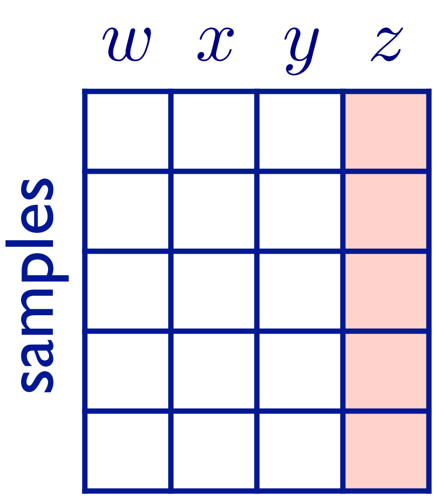
High-Level

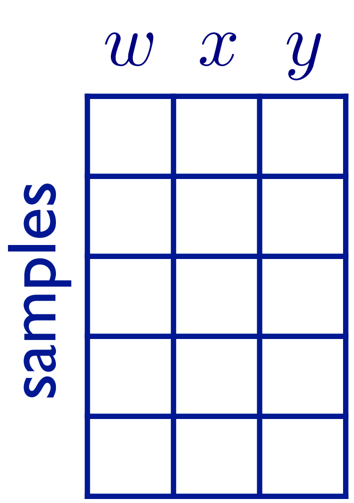
data sample

assumptions, e.g.
- causal Markov
- causal faithfulness
- etc.

background knowledge, e.g.
- pathways
- tier ordering
- “priors”
- etc.

setting
- subsampled time series
- tier structure





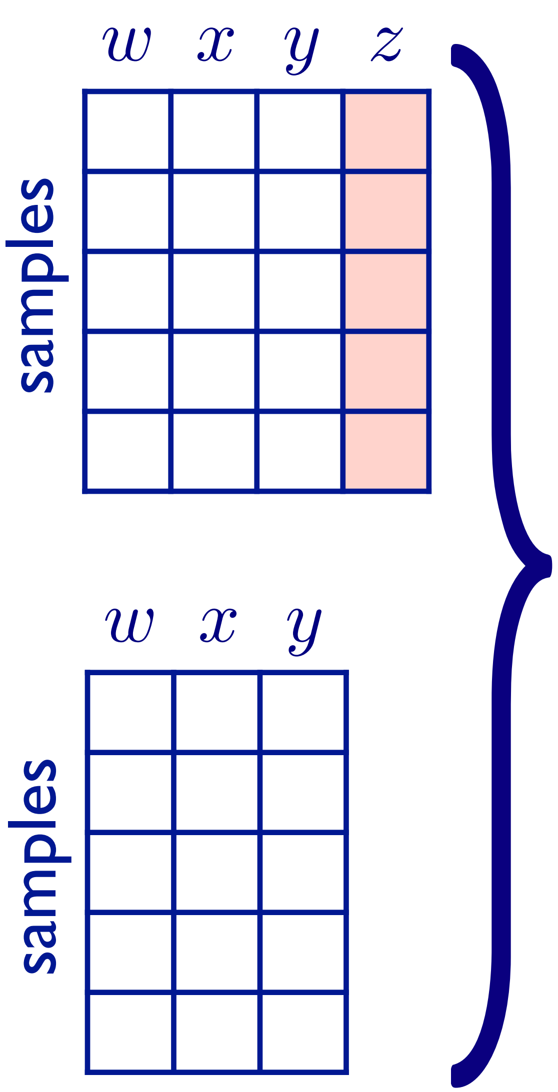
High-Level

data sample

assumptions, e.g.
- causal Markov
- causal faithfulness
- etc.

background knowledge, e.g.
- pathways
- tier ordering
- “priors”
- etc.

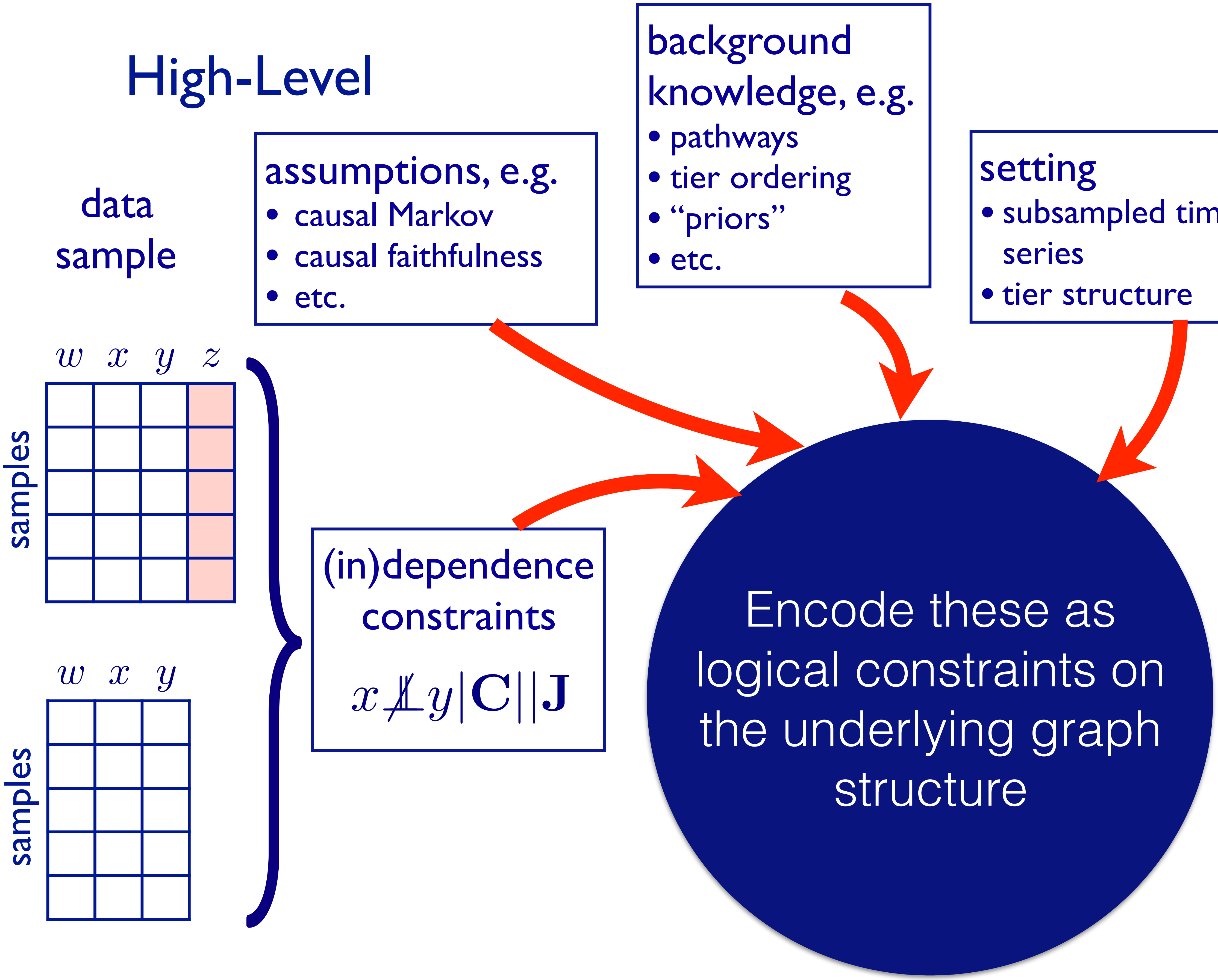
setting
- subsampled time series
- tier structure

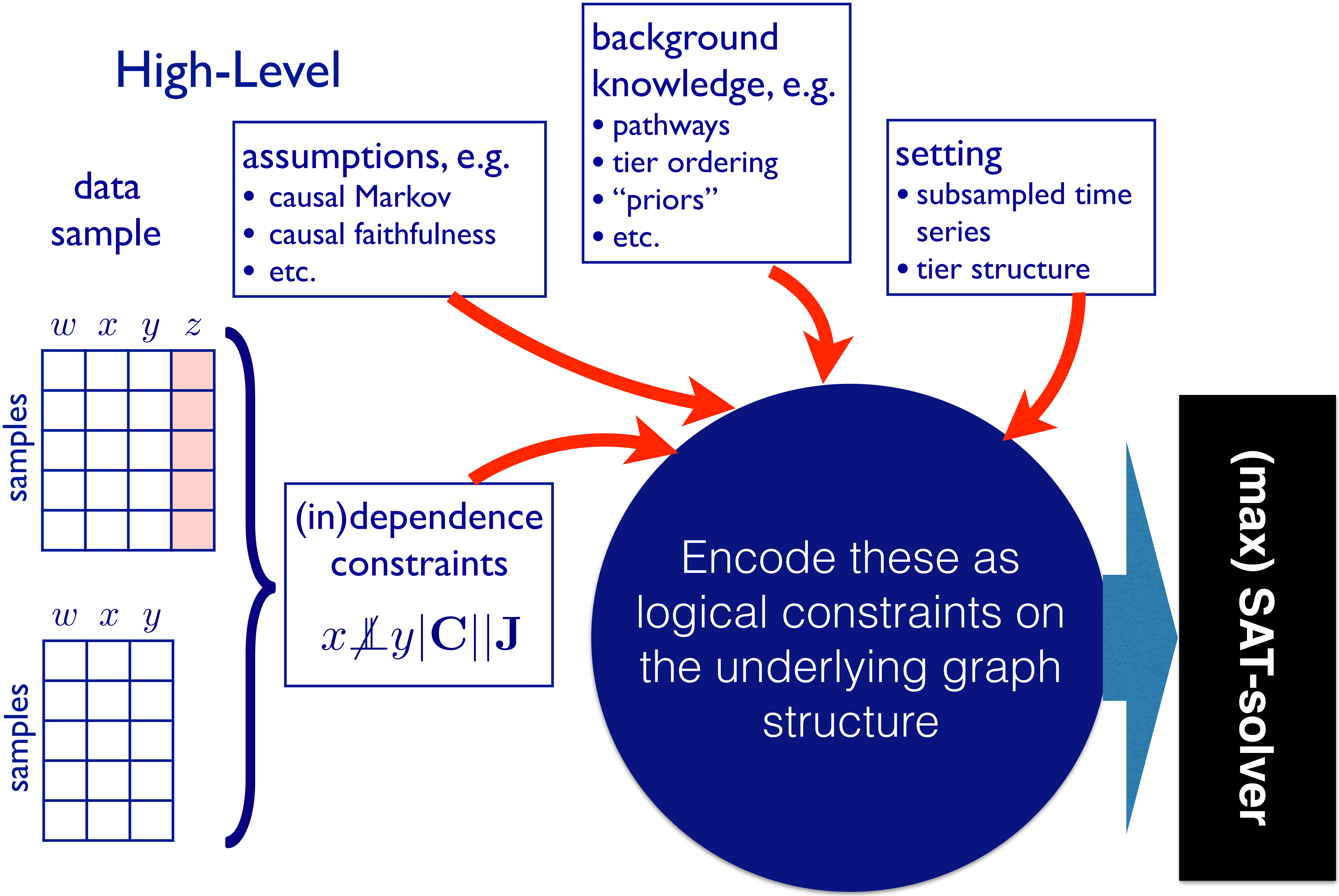


(in)dependence constraints

$$x \not\perp\!\!\!\perp y \mid \mathbf{C} \,||\, \mathbf{J}$$

High-Level


data sample
w x y z
samples
w x y
samples
assumptions, e.g.
• causal Markov
• causal faithfulness
• etc.
background knowledge, e.g.
• pathways
• tier ordering
• "priors"
• etc.
setting
• subsampled time series
• tier structure
(in)dependence constraints
$x \not\perp\!\!\!\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$
Encode these as logical constraints on the underlying graph structure


High-Level
data sample
w x y z
samples
w x y
samples
assumptions, e.g.
• causal Markov
• causal faithfulness
• etc.
background knowledge, e.g.
• pathways
• tier ordering
• "priors"
• etc.
setting
• subsampled time series
• tier structure
(in)dependence constraints
$x \not\perp\!\!\!\perp y \,|\, \mathbf{C} \,||\, \mathbf{J}$
Encode these as logical constraints on the underlying graph structure
(max) SAT-solver

d-separation and independence

- Under the assumption of causal Markov and causal Faithfulness:

$$x \not\perp y \mid \mathbf{C} \;||\; \mathbf{J} \iff x \not\perp\!\!\!\perp y \mid \mathbf{C} \;||\; \mathbf{J}$$

d-separation and independence

- Under the assumption of causal Markov and causal Faithfulness:

$$x \not\perp y \mid \mathbf{C} \,||\, \mathbf{J} \iff x \not\perp\!\!\!\perp y \mid \mathbf{C} \,||\, \mathbf{J}$$

x and y are d-connected given **C** when variables in **J** are subject to intervention $\iff$ x and y are dependent given **C** when variables in **J** are subject to intervention

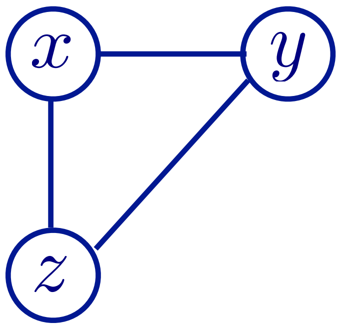
Example with 3 variables: x, y, z

$$x \perp\!\!\!\perp y$$

Example with 3 variables: x, y, z

$$x \perp\!\!\!\perp y$$

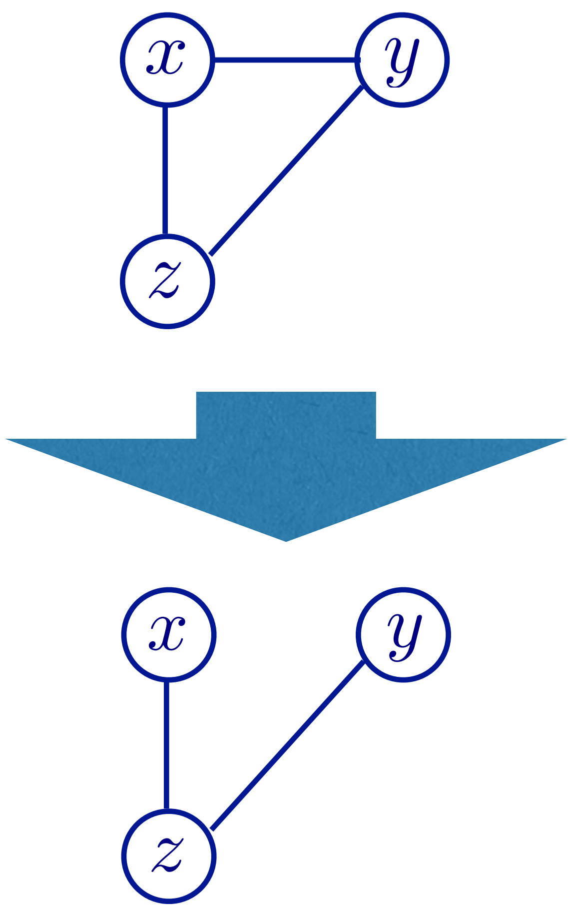
PC-algorithm



Example with 3 variables: x, y, z

$$x \perp\!\!\!\perp y$$

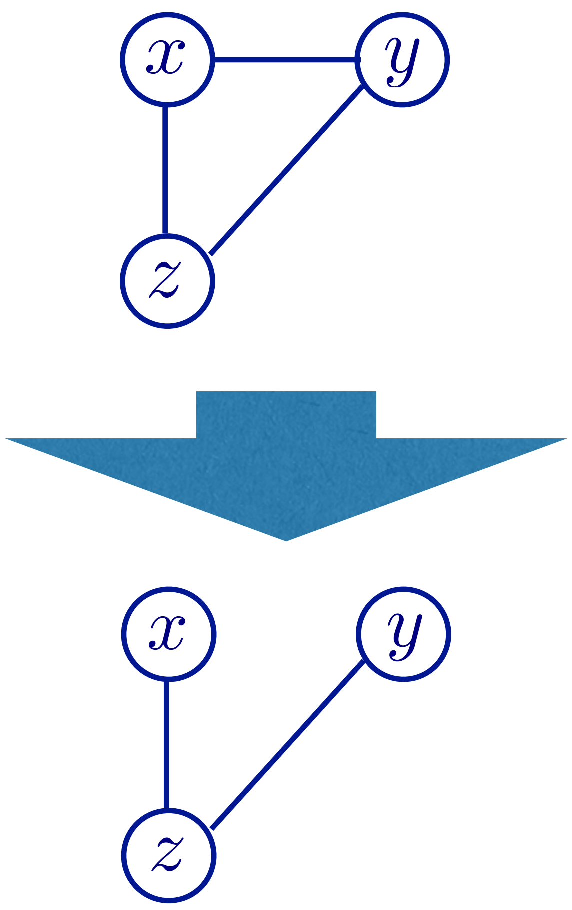
PC-algorithm



Example with 3 variables: x, y, z

$$x \perp\!\!\!\perp y$$

PC-algorithm



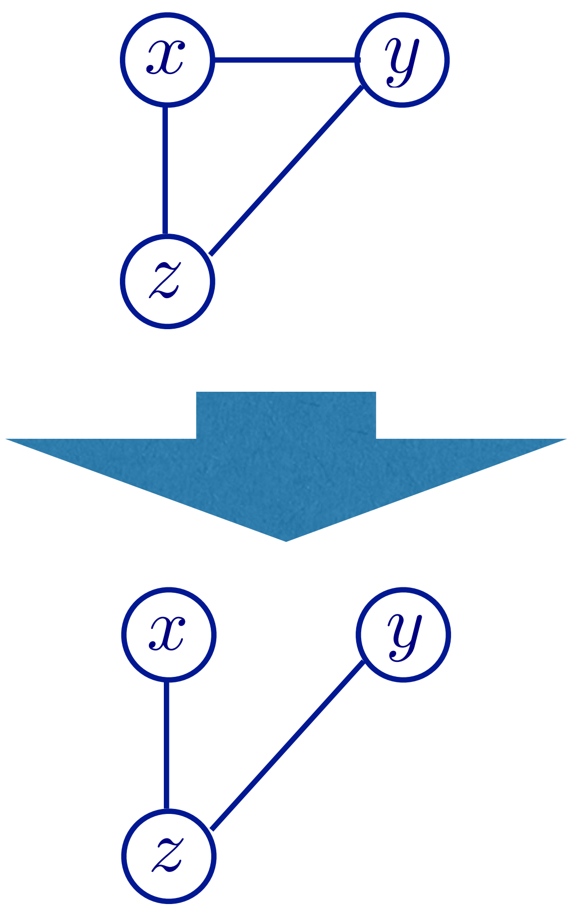
SAT-algorithm

define atoms
$$\begin{cases} A := \text{``}x \to y \in G\text{''} \\ B := \text{``}y \to x \in G\text{''} \\ C := \text{``}z \to x \in G\text{''} \\ D := \text{``}z \to y \in G\text{''} \\ \quad \ldots \end{cases}$$

Example with 3 variables: x, y, z

$$x \perp\!\!\!\perp y$$

PC-algorithm



SAT-algorithm

define atoms

$$\begin{cases} A := \text{“}x \to y \in G\text{”} \\ B := \text{“}y \to x \in G\text{”} \\ C := \text{“}z \to x \in G\text{”} \\ D := \text{“}z \to y \in G\text{”} \\ \quad \ldots \end{cases}$$

encode constraint

$$\begin{cases} \neg A \wedge \neg B & \text{// direct edges} \\ \wedge \neg (C \wedge D) & \text{// common causes} \\ \wedge \neg \ldots & \text{// indirect paths} \end{cases}$$

SAT-based Causal Discovery

- **Formulate the independence constraints in propositional logic**

$$x \perp\!\!\!\perp y \iff \neg A \wedge \neg B \ldots$$

$$A = \text{`}x \rightarrow y \text{ is present'}$$

SAT-based Causal Discovery

- **Formulate the independence constraints in propositional logic**

$$x \perp\!\!\!\perp y \iff \neg A \wedge \neg B \ldots$$

$$A = \text{`}x \rightarrow y \text{ is present'}$$

- **Encode the constraints into one formula.**

$$\neg A \wedge \neg B \wedge \neg(C \wedge D) \wedge \neg \ldots$$

SAT-based Causal Discovery

- **Formulate the independence constraints in propositional logic**

$$x \perp\!\!\!\perp y \iff \neg A \wedge \neg B \ldots$$

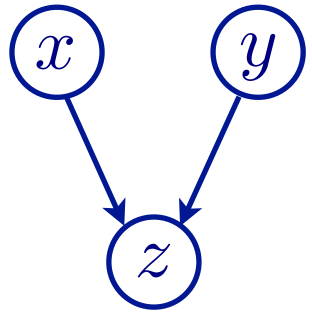
$$A = \text{`}x \to y \text{ is present'}$$

- **Encode the constraints into one formula.**

$$\neg A \wedge \neg B \wedge \neg (C \wedge D) \wedge \neg \ldots$$

- **Find satisfying assignments using a SAT-solver**

$$A = false$$
$$B = false \iff$$
$$\ldots$$



SAT-based Causal Discovery

- **Formulate the independence constraints in propositional logic**

$$x \perp\!\!\!\perp y \iff \neg A \wedge \neg B \ldots$$

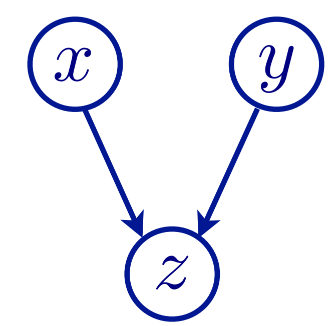
$$A = \text{`}x \rightarrow y \text{ is present'}$$

- **Encode the constraints into one formula.**

$$\neg A \wedge \neg B \wedge \neg (C \wedge D) \wedge \neg \ldots$$

- **Find satisfying assignments using a SAT-solver**

$$\begin{aligned} A &= false \\ B &= false \\ &\ldots \end{aligned} \iff$$



➡ very general setting (allows for cycles and latents) and trivially complete

SAT-based Causal Discovery

- **Formulate the independence constraints in propositional logic**

$$x \perp\!\!\!\perp y \iff \neg A \wedge \neg B \ldots$$

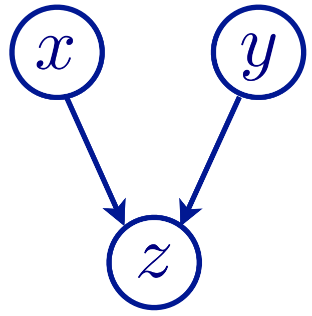
$$A = \text{`}x \rightarrow y \text{ is present'}$$

- **Encode the constraints into one formula.**

$$\neg A \wedge \neg B \wedge \neg (C \wedge D) \wedge \neg \ldots$$

- **Find satisfying assignments using a SAT-solver**

$$A = false$$
$$B = false \iff$$
$$\ldots$$



➡ very general setting (allows for cycles and latents) and trivially complete

➡ **BUT**: erroneous test results induce conflicting constraints: UNsatisfiable

Conflicts and Errors

- **Statistical independence tests produce errors**

constraint

$$x \not\perp\!\!\!\perp z$$

$$y \not\perp\!\!\!\perp z$$

$$x \perp\!\!\!\perp y$$

$$x \perp\!\!\!\perp y | z$$

Conflicts and Errors

- **Statistical independence tests produce errors**

➡ **Conflict: no graph can produce the set of constraints**

constraint

$$x \not\perp\!\!\!\perp z$$

$$y \not\perp\!\!\!\perp z$$

$$x \perp\!\!\!\perp y$$

$$x \perp\!\!\!\perp y \mid z$$

Conflicts and Errors

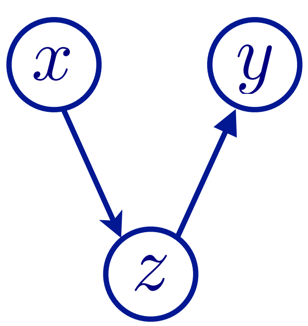
- **Statistical independence tests produce errors**

  ➡ **Conflict: no graph can produce the set of constraints**

constraint

$$x \not\perp\!\!\!\perp z$$

$$y \not\perp\!\!\!\perp z$$



$x \not\perp\!\!\!\perp y$

$$x \perp\!\!\!\perp y$$

$$x \perp\!\!\!\perp y \,|\, z$$

Conflicts and Errors

- **Statistical independence tests produce errors**

  ➡ **Conflict: no graph can produce the set of constraints**

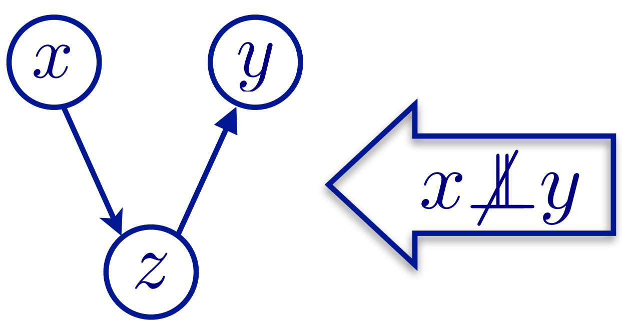
constraint

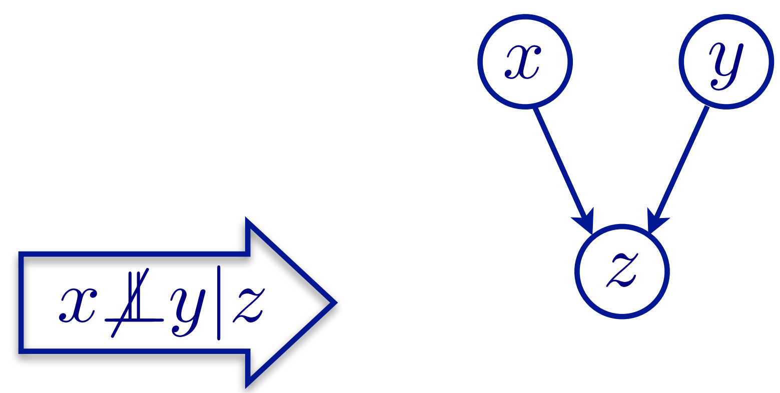
$$x \not\perp\!\!\!\perp z$$

$$y \not\perp\!\!\!\perp z$$

$$x \perp\!\!\!\perp y$$

$$x \perp\!\!\!\perp y \mid z$$

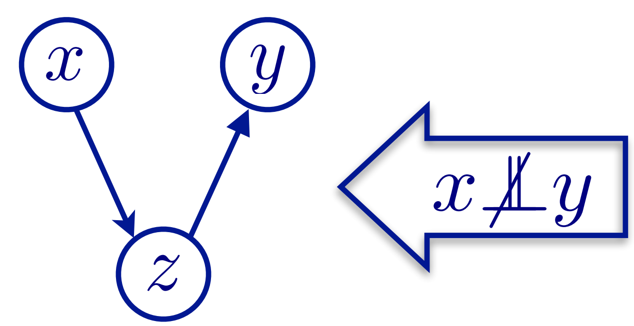




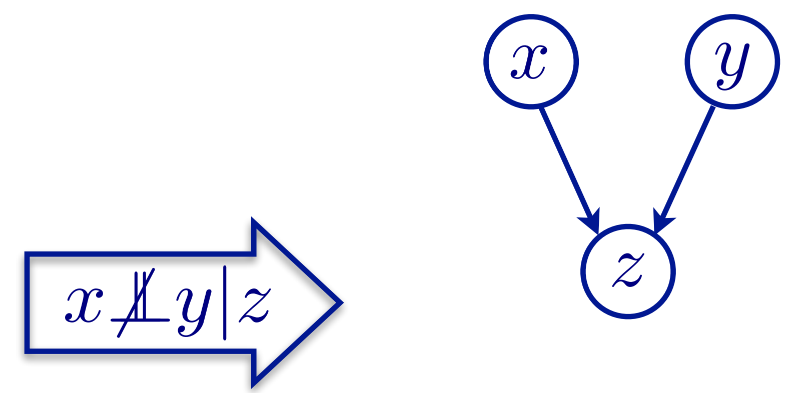
Conflicts and Errors

- **Statistical independence tests produce errors**

  ➡ **Conflict: no graph can produce the set of constraints**



constraint	weight
$x \not\perp\!\!\!\perp z$	3000
$y \not\perp\!\!\!\perp z$	2500
$x \perp\!\!\!\perp y$	500
$x \perp\!\!\!\perp y \mid z$	250



Constraint Satisfaction Approach

- **INPUT: (in)dependence constraints weighted according to reliability**

$$\min_{G} \sum_{k\,:\,\text{constraint } k \text{ is } \textbf{not} \text{ satisfied by } G} w(k)$$

- **OUTPUT: a graph G that minimizes the cost**

Constraint Satisfaction Approach

- **INPUT: (in)dependence constraints weighted according to reliability**

$$\min_G \sum_{k\,:\,\text{constraint } k \text{ is } \textbf{not} \text{ satisfied by } G} w(k)$$

- **OUTPUT: a graph G that minimizes the cost**

What are suitable weights?

Weighting Schemes

- Constant weights
  - unit weights for all constraint

Weighting Schemes

- Constant weights
  - unit weights for all constraint
- Hard dependencies
  - only treat rejections of the null-hypothesis as hard constraints, in line with classical statistics
  - give dependences infinite weight, maximize the independences (unit weight) in light of these dependences

Weighting Schemes

- Constant weights
  - unit weights for all constraint
- Hard dependencies
  - only treat rejections of the null-hypothesis as hard constraints, in line with classical statistics
  - give dependences infinite weight, maximize the independences (unit weight) in light of these dependences
- Log weights
  - obtain the probability of an (in)dependence and weigh it according to the log of the probability
  - Model selection with Bayes rule:

$$\begin{array}{ccc} x \not\perp\!\!\!\perp y \mid C & & x \perp\!\!\!\perp y \mid C \\ P(x|C)P(y|x,C) & \text{vs.} & P(x|C)P(y|C) \end{array}$$

Weighting Schemes

- Constant weights
  - unit weights for all constraint
- Hard dependencies
  - only treat rejections of the null-hypothesis as hard constraints, in line with classical statistics
  - give dependences infinite weight, maximize the independences (unit weight) in light of these dependences
- Log weights
  - obtain the probability of an (in)dependence and weigh it according to the log of the probability
  - Model selection with Bayes rule:

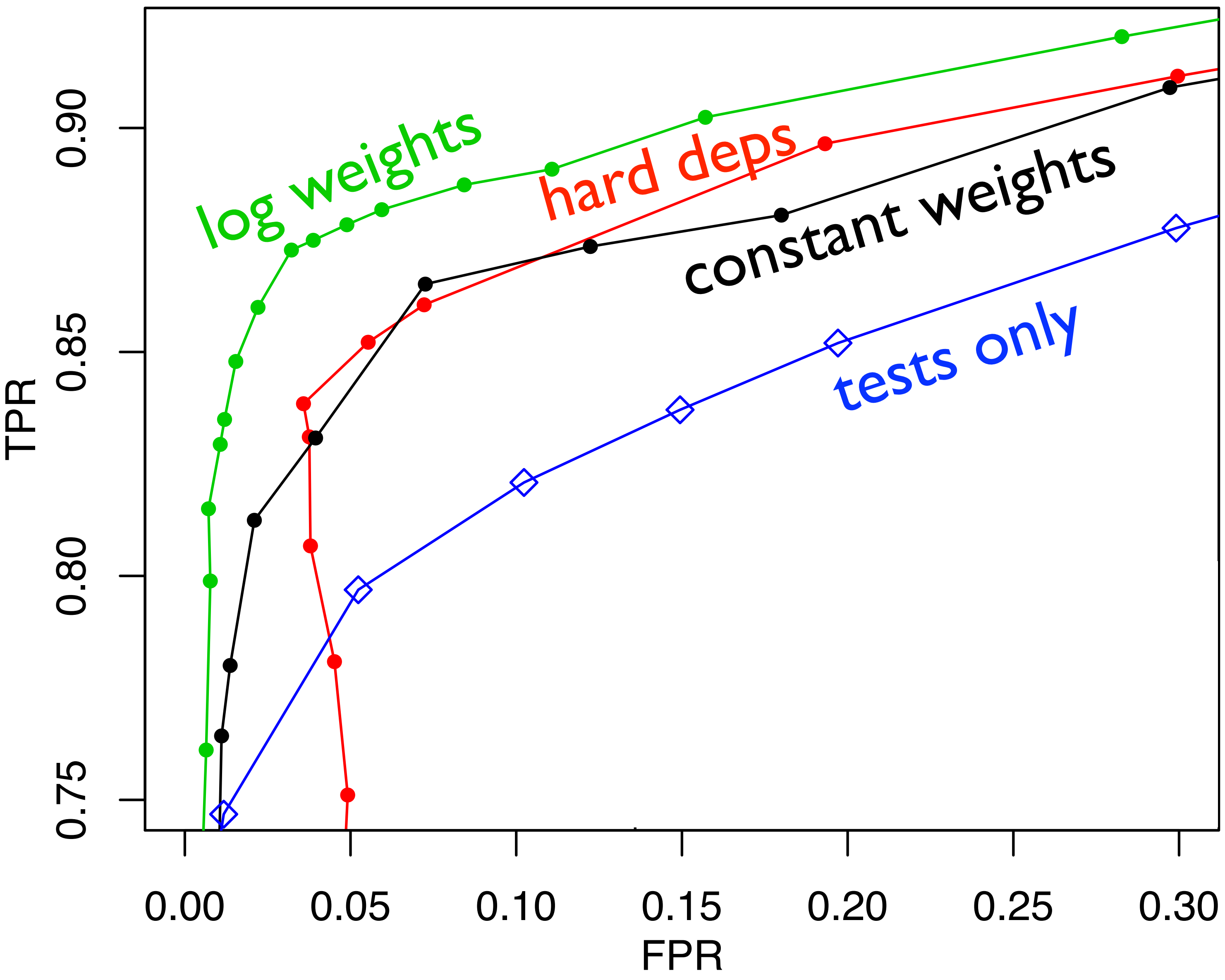
$$\begin{array}{ccc} x \not\perp\!\!\!\perp y | C & & x \perp\!\!\!\perp y | C \\ P(x|C)P(y|x, C) & \text{vs.} & P(x|C)P(y|C) \end{array}$$

  - probabilistic classifier: find G such that if it were true, test results would be optimal in the sense of a proper score

Optimization

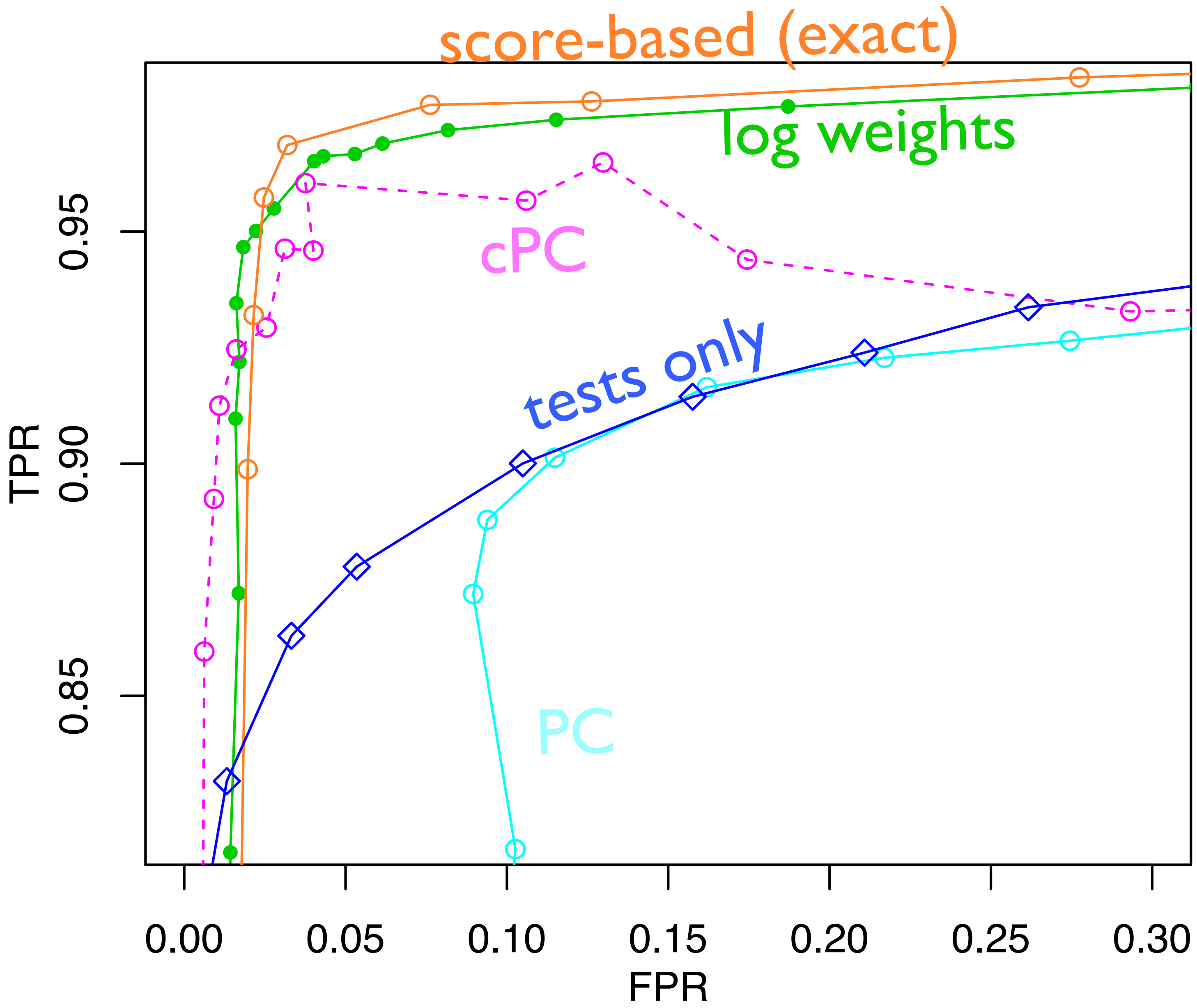
- Answer Set Programming (ASP) is a modern declarative programming paradigm
  - solver used: Clingo
  - SAT-solver and branch and bound algorithm
  - finds globally optimal weighted maxSAT solution

Simulation 1: cycles and latents



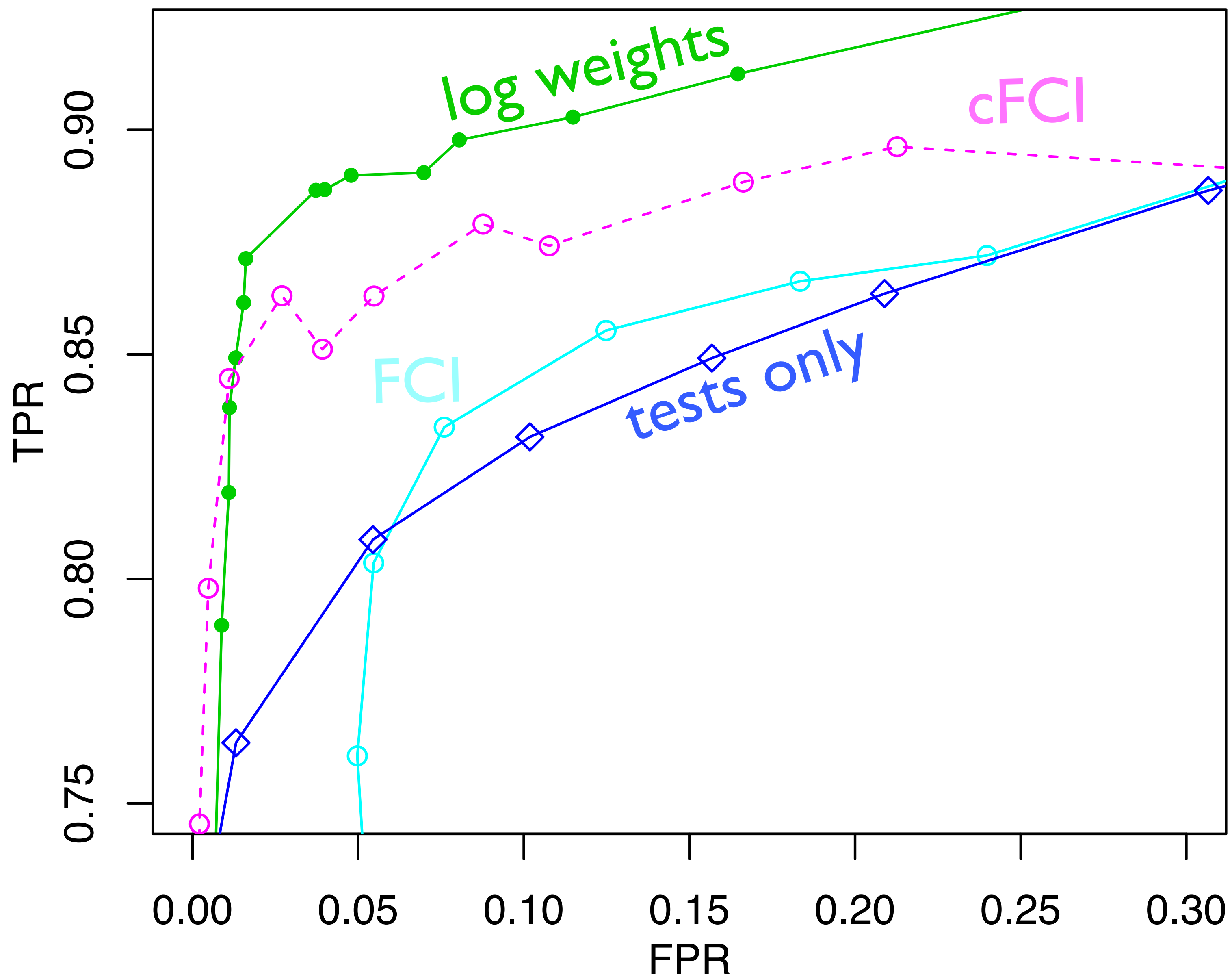
- ROC of dependences, passive observational data set, 6 observed variables, average degree 2; 500 samples, 200 models, linear Gaussian parameterization

Simulation 2: no cycles, no latents



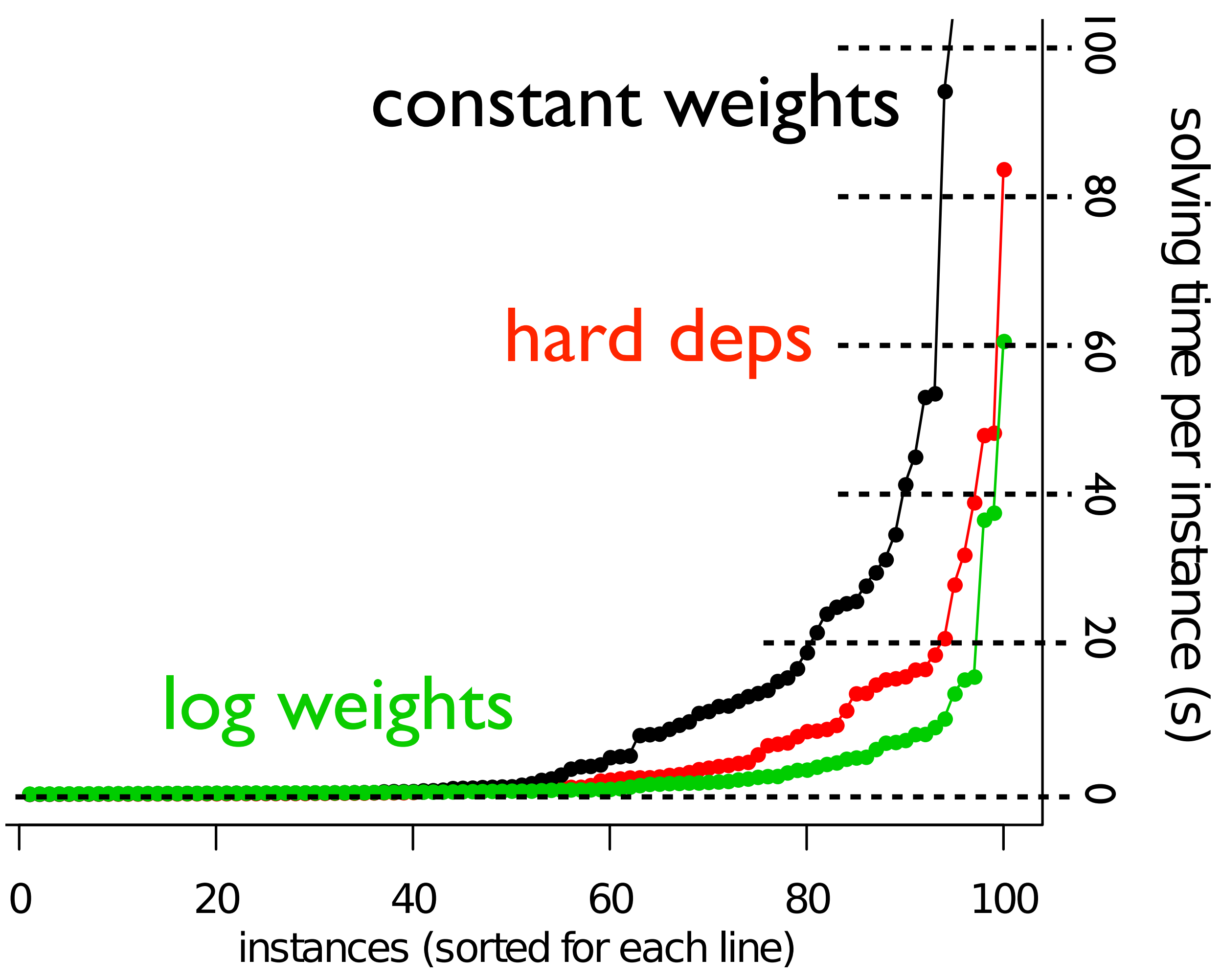
- cPC returns a fully determined output only 58/200 times at its optimum

Simulation 3: no cycles, but latents



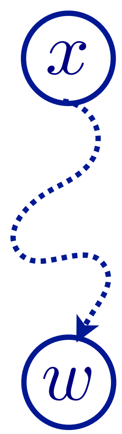
- cFCI only returns unambiguous results 61/200 times at its optimum

Simulation 4: Scalability



- up to 7 variables and only a few data sets for now (9x10^18 graphs)

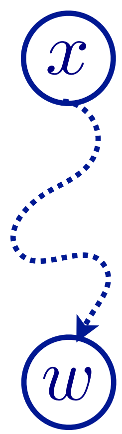
Background Knowledge



$$\boxed{\begin{matrix} x \\ y \end{matrix}} > \boxed{\begin{matrix} z \\ w \end{matrix}}$$

“priors”

Background Knowledge

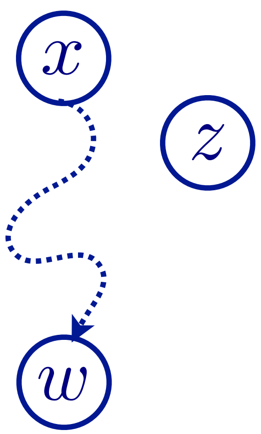


$$\Longleftrightarrow \qquad x \not\perp\!\!\!\perp w \,||\, x$$

$$\boxed{\begin{matrix} x \\ y \end{matrix}} > \boxed{\begin{matrix} z \\ w \end{matrix}}$$

“priors”

Background Knowledge

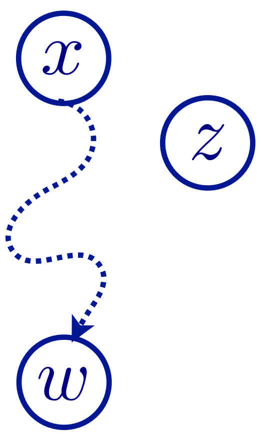


$$\Longleftrightarrow \qquad x \not\perp\!\!\!\perp w \,||\, x \ z$$

$$\boxed{\begin{array}{c} x \\ y \end{array}} > \boxed{\begin{array}{c} z \\ w \end{array}}$$

"priors"

Background Knowledge

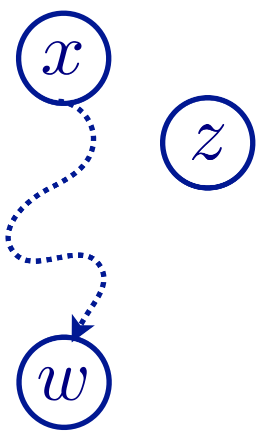


$$\Longleftrightarrow \qquad x \not\perp\!\!\!\perp w \,||\, x \; z \qquad weight = 0.8$$

$$\boxed{\begin{matrix} x \\ y \end{matrix}} > \boxed{\begin{matrix} z \\ w \end{matrix}}$$

“priors”

Background Knowledge

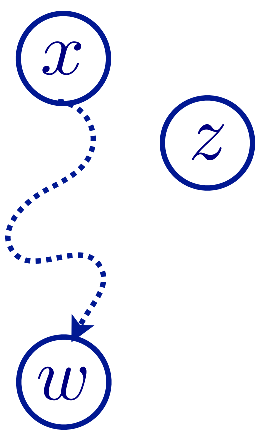


$$\Longleftrightarrow \qquad x \not\perp\!\!\!\perp w \,||\, x \; z \qquad weight = 0.8$$

$$\boxed{\begin{array}{c} x \\ y \end{array}} > \boxed{\begin{array}{c} z \\ w \end{array}} \qquad \Longleftrightarrow \qquad \begin{array}{l} (x > z) \wedge (x > w) \\ \wedge\, (y > z \wedge (y > w) \end{array}$$

"priors"

Background Knowledge

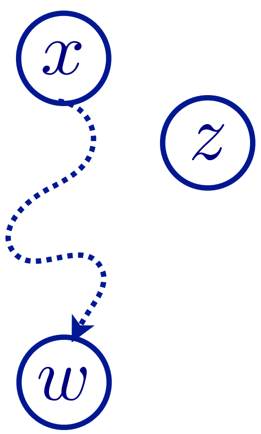


$$\Longleftrightarrow \quad x \not\perp\!\!\!\perp w \,||\, x\ z \qquad weight = 0.8$$

$$\boxed{\begin{matrix} x \\ y \end{matrix}} > \boxed{\begin{matrix} z \\ w \end{matrix}} \quad \Longleftrightarrow \quad \begin{matrix} (x > z) \wedge (x > w) \\ \wedge\ (y > z \wedge (y > w) \end{matrix}$$

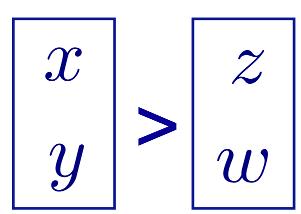
"priors" $\Longleftrightarrow$

Background Knowledge



$\Longleftrightarrow$

$$x \not\!\perp\!\!\!\perp w \,||\, x \; z \qquad weight = 0.8$$



$\Longleftrightarrow$

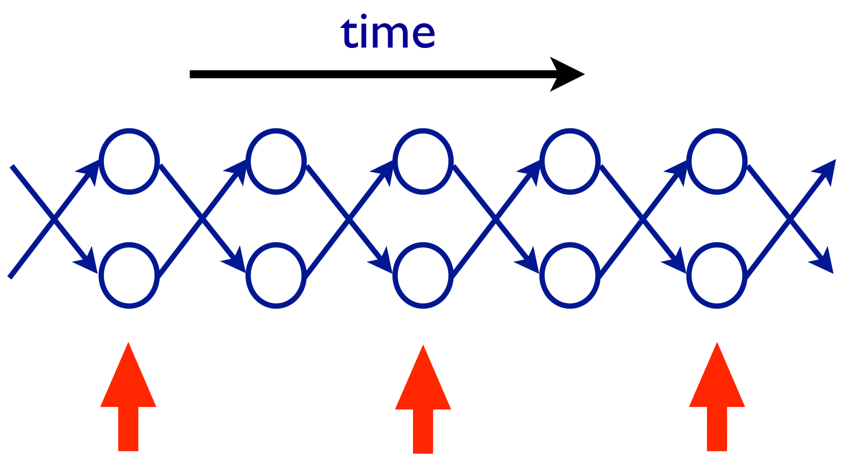
$$(x > z) \wedge (x > w)$$
$$\wedge \; (y > z \wedge (y > w)$$

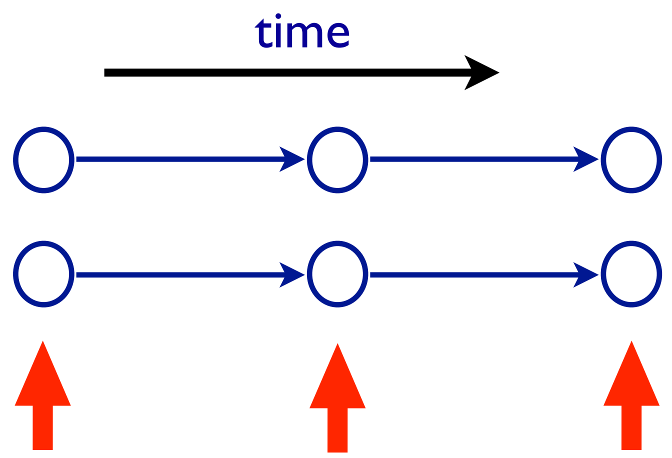
“priors”

$\Longleftrightarrow$

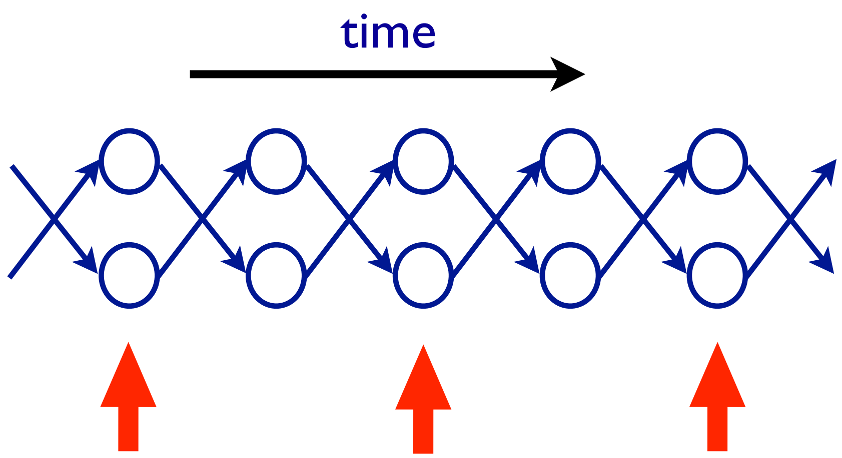
- specific probabilities for each graph
- soft sparsity constraint
- …

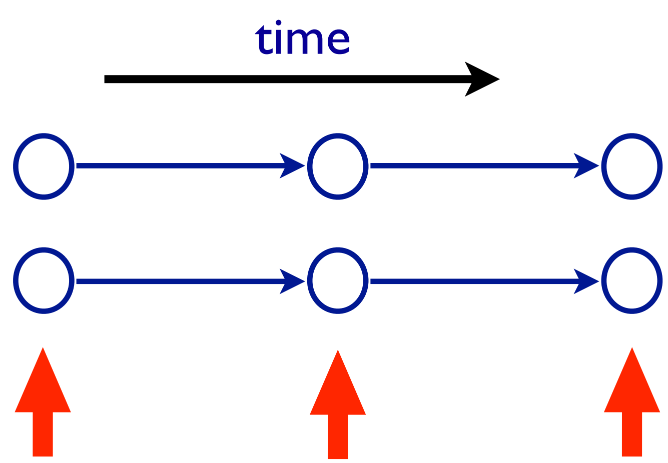
Settings





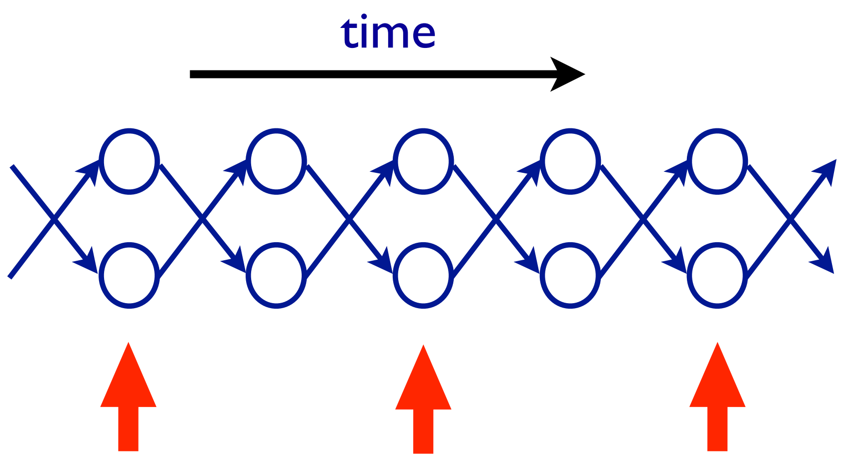
Settings

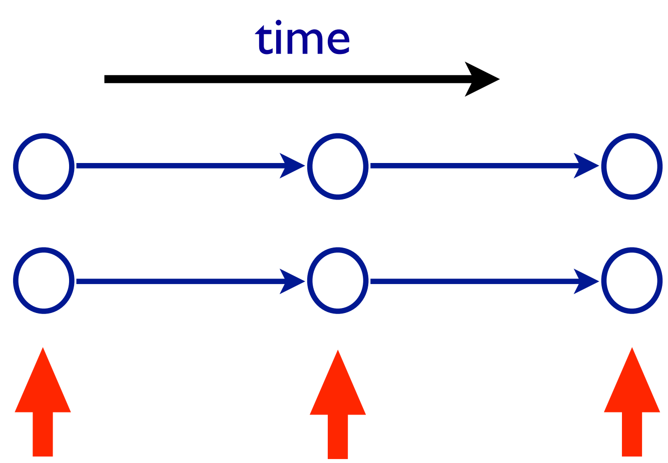





```
urange(1..5).

1 { u(U): urange(U) } 1.

{ edge1(X,Y) } :- node(X), node(Y).

path(X,Y,1) :- edge1(X,Y).
path(X,Y,L) :- path(X,Z,L-1), edge1(Z,Y),
L <= U, u(U).

edge1u(X,Y) :- path(X,Y,L), u(L).


conf1u(X,Y) :- path(Z,X,L), path(Z,Y,L),
node(X),node(Y), node(Z),
X < Y, L < U, u(U).

:- edgeu(X,Y), not edge1u(X,Y).
:- no_edgeu(X,Y), edge1u(X,Y).
:- confu(X,Y), not conf1u(X,Y).
:- no_confu(X,Y), conf1u(X,Y).
```

Settings





range for rate of subsampling

```
urange(1..5).
```

subsampling rate is unique

```
1 { u(U): urange(U) } 1.
```

def. of edge in graph

```
{ edge1(X,Y) } :- node(X), node(Y).

path(X,Y,1) :- edge1(X,Y).
path(X,Y,L) :- path(X,Z,L-1), edge1(Z,Y),
L <= U, u(U).
```

recursive def. of path

```
edge1u(X,Y) :- path(X,Y,L), u(L).
```

def. of edge in subsampled graph

```
conf1u(X,Y) :- path(Z,X,L), path(Z,Y,L),
node(X),node(Y), node(Z),
X < Y, L < U, u(U).
```

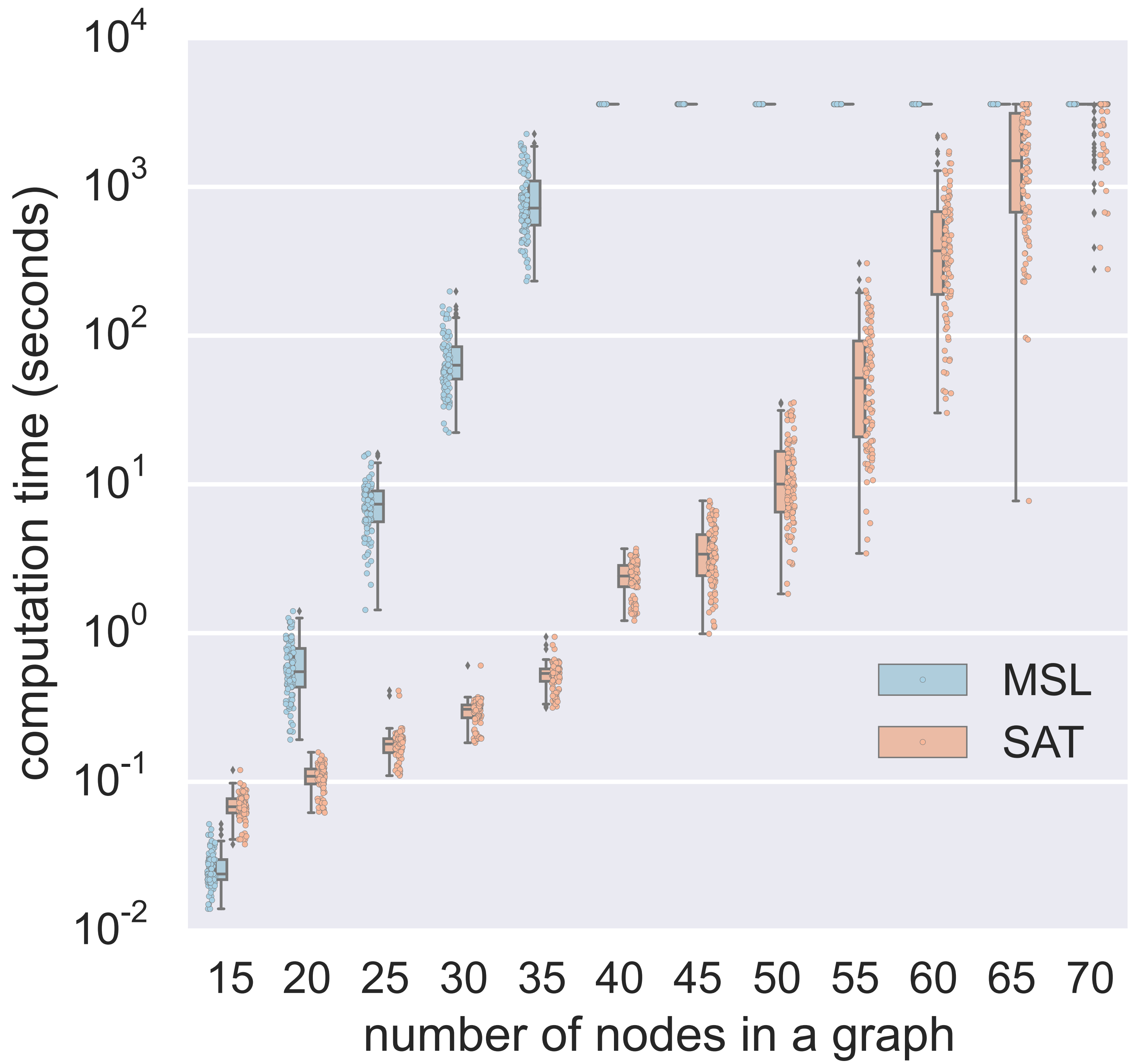
def. of how confounders arise due to subsampling

```
:- edgeu(X,Y), not edge1u(X,Y).
:- no_edgeu(X,Y), edge1u(X,Y).
:- confu(X,Y), not conf1u(X,Y).
:- no_confu(X,Y), conf1u(X,Y).
```

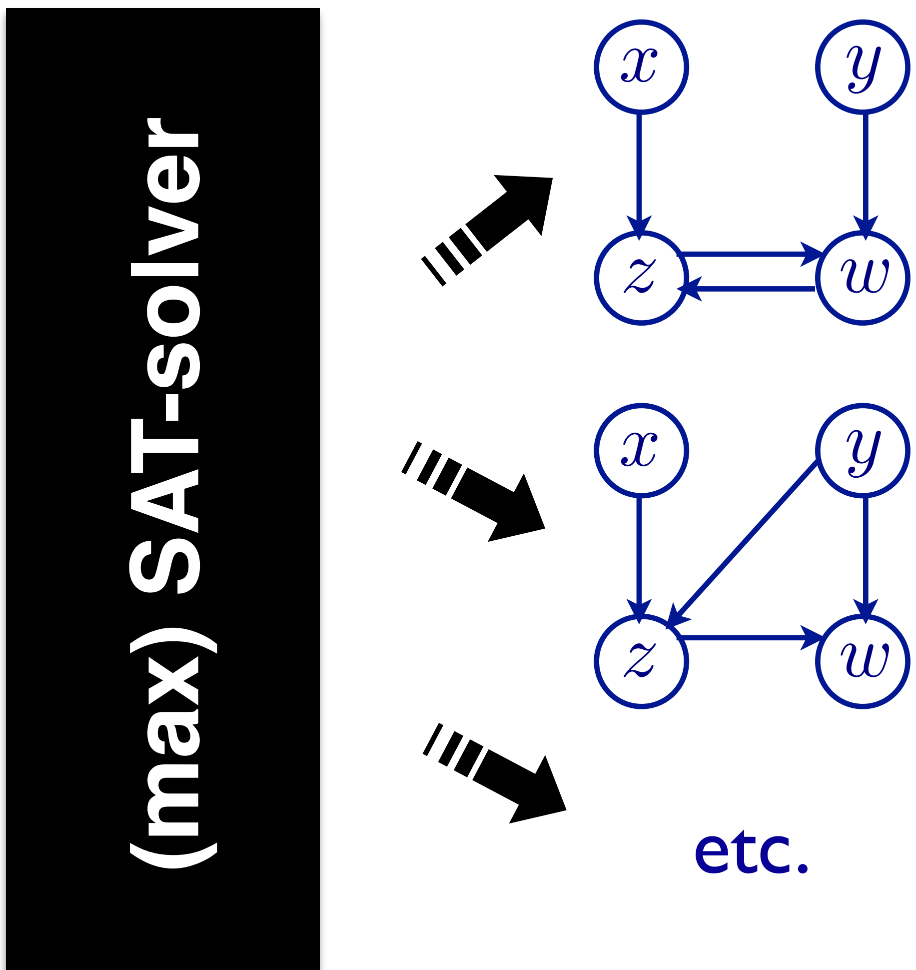
constraints on how edges in subsampled graph relate to edges in true graph

Runtime comparison

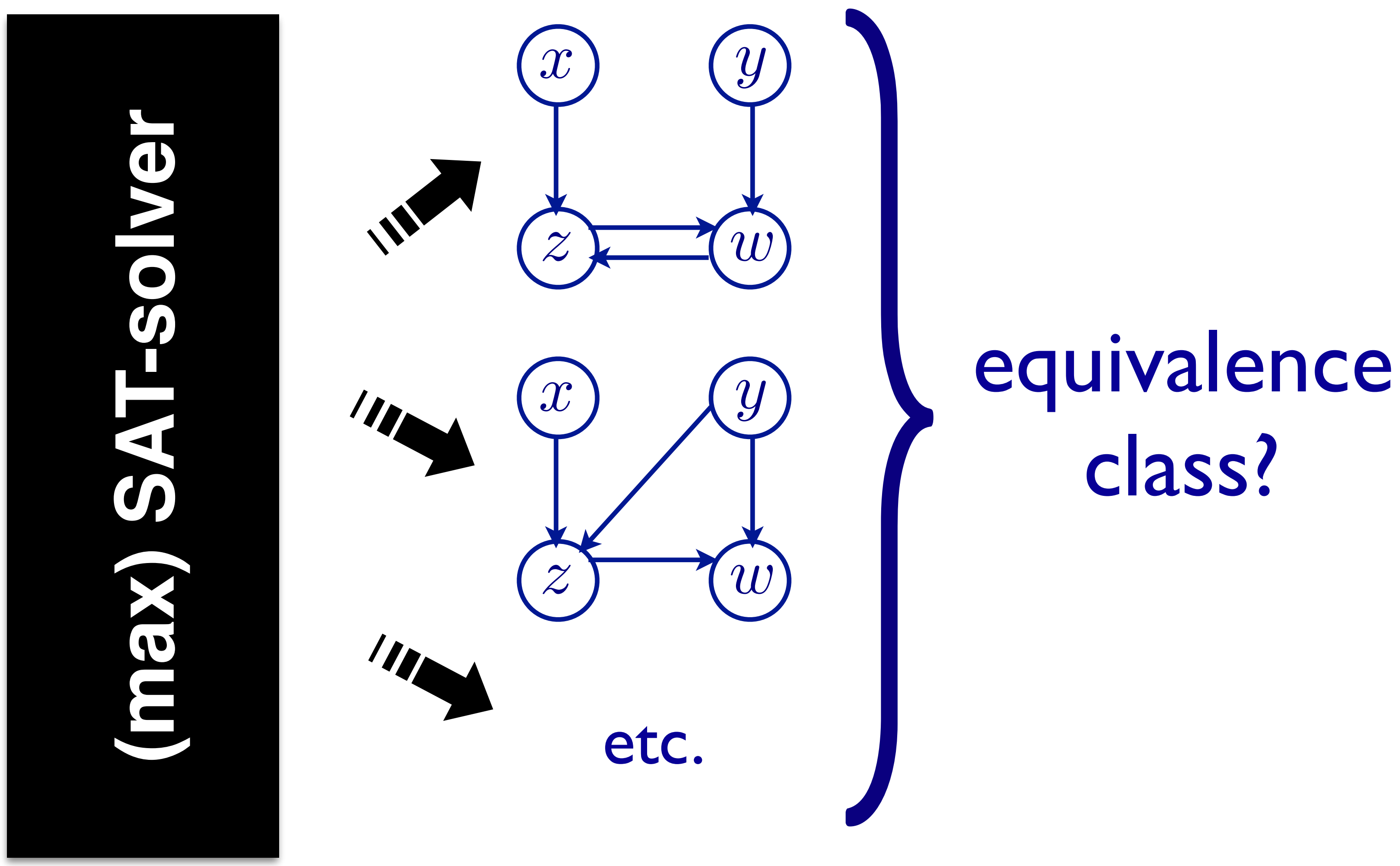
For a graph determined at subsampling rate 2, infer the equivalence class of graphs at the system time scale (1-step)



Output of Causal Search Algorithms



Output of Causal Search Algorithms

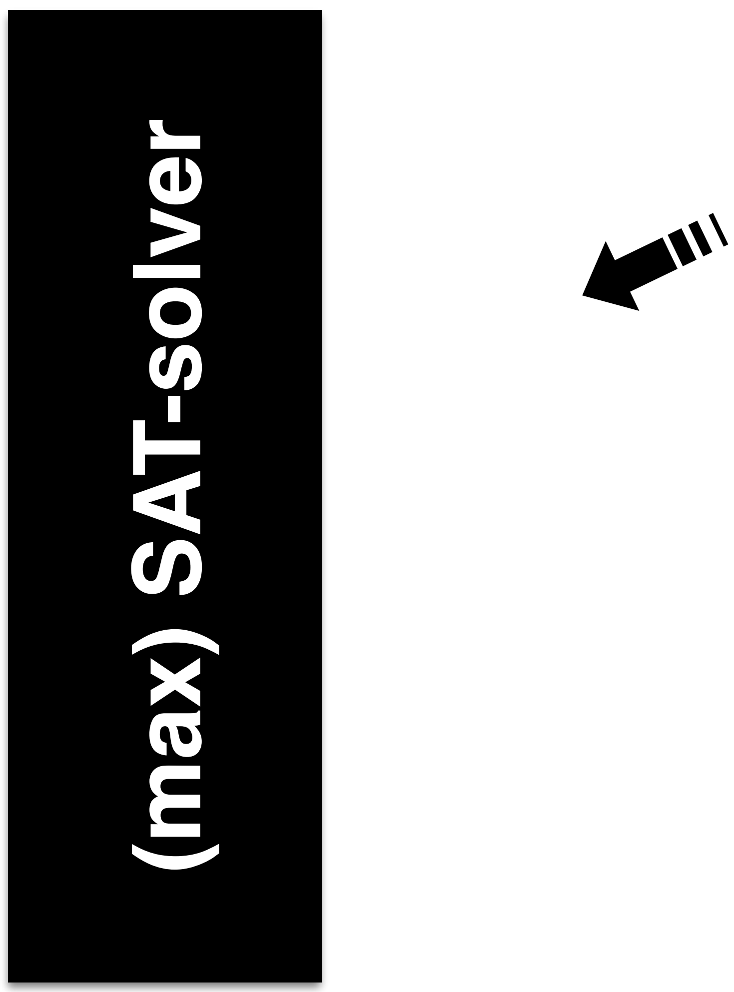


Output of Causal Search Algorithms

Query:

(max) SAT-solver

Output of Causal Search Algorithms



Query:

- list the structures in the equivalence class

Output of Causal Search Algorithms

(max) SAT-solver

Query:

- list the structures in the equivalence class
- what structural features are determined?
  - edges, confounders
  - ancestral relations
  - pathways

Output of Causal Search Algorithms

(max) SAT-solver

Query:

- list the structures in the equivalence class
- what structural features are determined?
  - edges, confounders
  - ancestral relations
  - pathways
- what are the highest scoring equivalence classes?

Output of Causal Search Algorithms

(max) SAT-solver

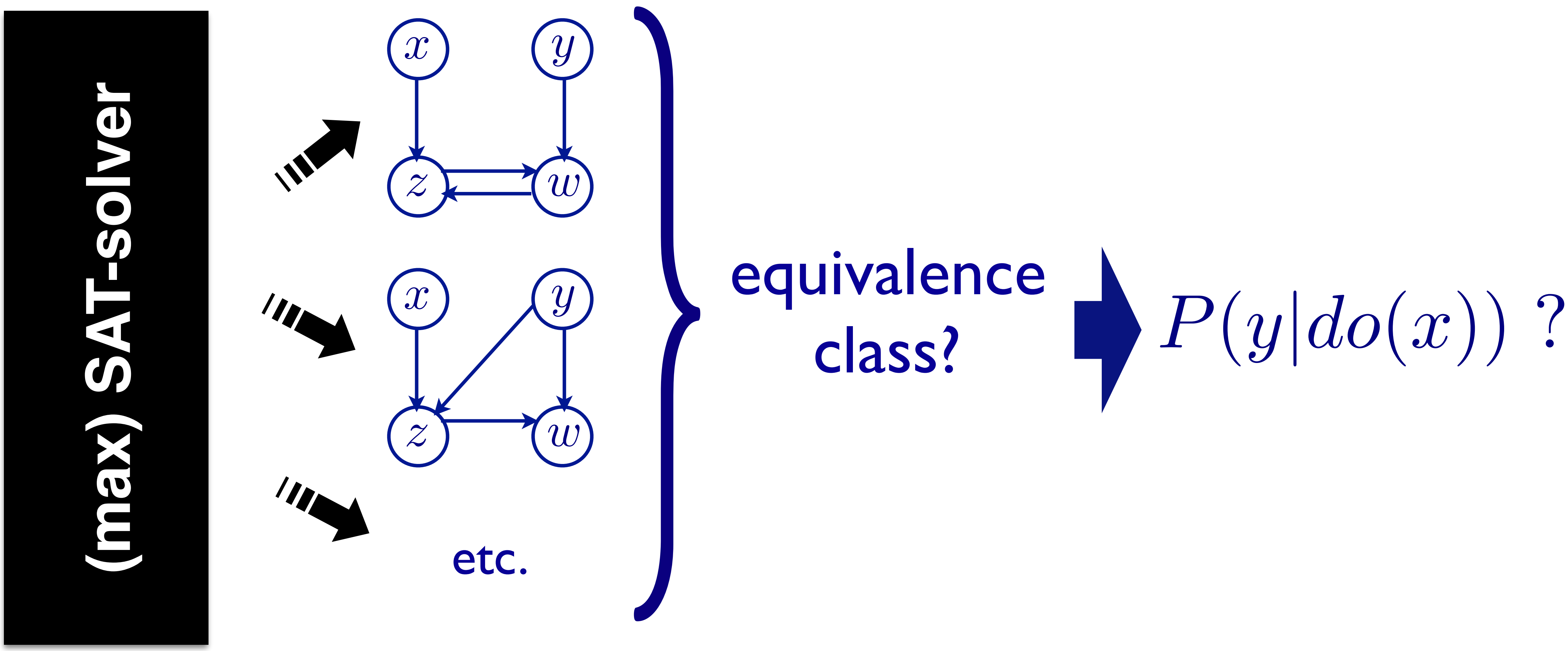
Query:

- list the structures in the equivalence class
- what structural features are determined?
  - edges, confounders
  - ancestral relations
  - pathways
- what are the highest scoring equivalence classes?

Response:

- enumeration of solutions
- “backbone” of the SAT-instance
- …

Computing Causal Effects



equivalence class? ➡ $P(y|do(x))$?

equivalence class? ➡ $P(y|do(x))$?

- enumerate each graph in the equivalence class and run the Tian-Shpitser algorithm to determine the causal effect?

equivalence class? $\Rightarrow$ $P(y|do(x))$?

- enumerate each graph in the equivalence class and run the Tian-Shpitser algorithm to determine the causal effect?
- Alternative:

equivalence class? ➡ $P(y|do(x))$?

- enumerate each graph in the equivalence class and run the Tian-Shpitser algorithm to determine the causal effect?
- Alternative:

do-calculus

Rule 1 (insertion/deletion of observations)

$$P(y|do(x), z, w) = P(y|do(x), w) \text{ if } Y \perp\!\!\!\perp Z|X, W||X$$

Rule 2 (action/observation exchange)

$$P(y|do(x), do(z), w) = P(y|do(x), z, w) \text{ if } Y \perp\!\!\!\perp I_Z|X, Z, W||X$$

Rule 3 (insertion/deletion of actions)

$$P(y|do(x), do(z), w) = P(y|do(x), w) \text{ if } Y \perp\!\!\!\perp I_Z|X, W||X$$

equivalence class? ➡ $P(y|do(x))$?

- enumerate each graph in the equivalence class and run the Tian-Shpitser algorithm to determine the causal effect?
- Alternative:

do-calculus

Rule 1 (insertion/deletion of observations)

$$P(y|do(x), z, w) = P(y|do(x), w) \text{ if } Y \perp\!\!\!\perp Z|X, W||X$$

Rule 2 (action/observation exchange)

$$P(y|do(x), do(z), w) = P(y|do(x), z, w) \text{ if } Y \perp\!\!\!\perp I_Z|X, Z, W||X$$

Rule 3 (insertion/deletion of actions)

$$P(y|do(x), do(z), w) = P(y|do(x), w) \text{ if } Y \perp\!\!\!\perp I_Z|X, W||X$$

equivalence class? ➡ $P(y|do(x))$?

- enumerate each graph in the equivalence class and run the Tian-Shpitser algorithm to determine the causal effect?
- Alternative:

do-calculus

Rule 1 (insertion/deletion of observations)

$$P(y|do(x), z, w) = P(y|do(x), w) \text{ if } Y \perp\!\!\!\perp Z|X, W||X$$

Rule 2 (action/observation exchange)

$$P(y|do(x), do(z), w) = P(y|do(x), z, w) \text{ if } Y \perp\!\!\!\perp I_Z|X, Z, W||X$$

Rule 3 (insertion/deletion of actions)

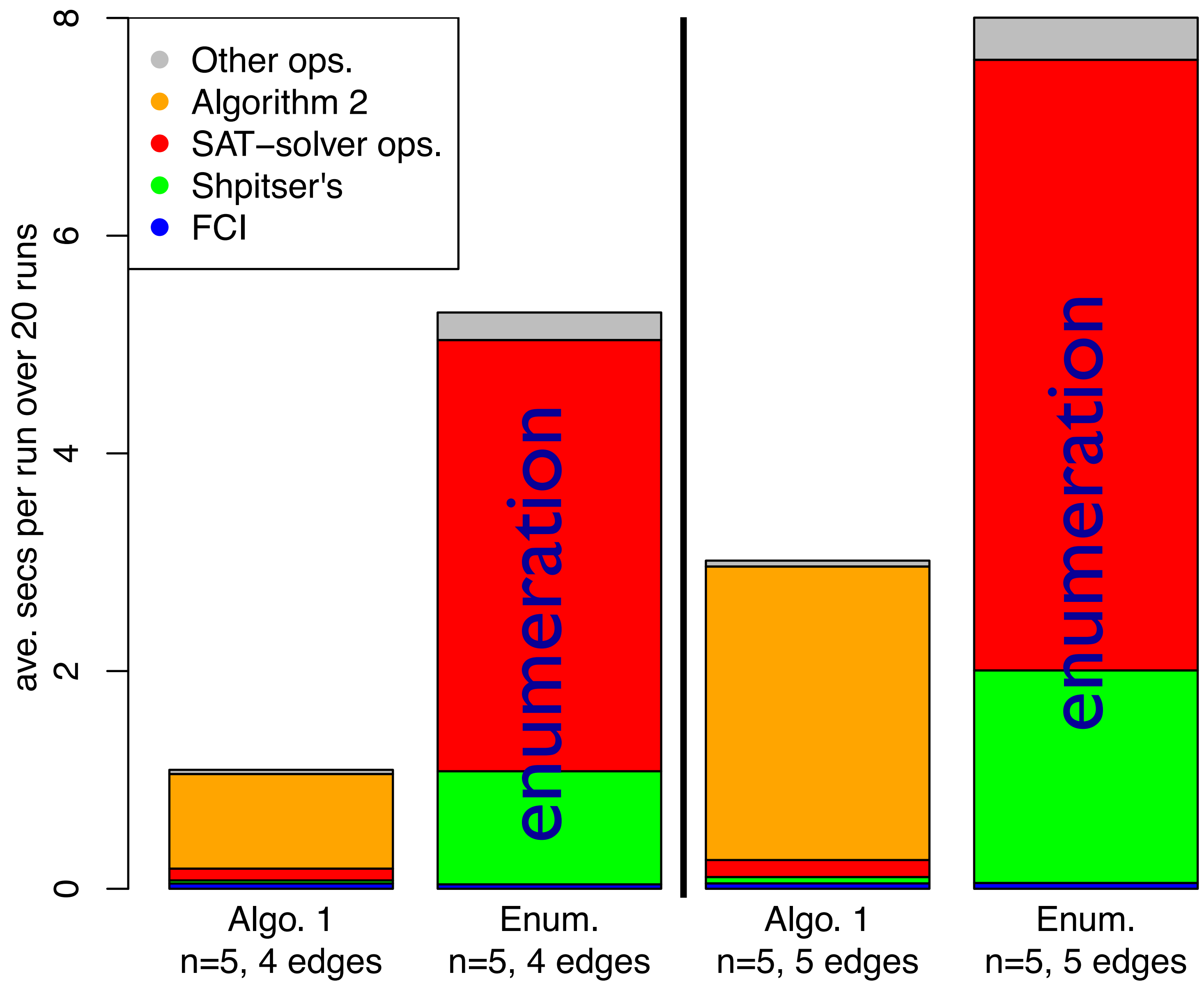
$$P(y|do(x), do(z), w) = P(y|do(x), w) \text{ if } Y \perp\!\!\!\perp I_Z|X, W||X$$

➡ search in the equivalence class over the possible applications of the *do*-calculus rules by *querying* the satisfaction of their conditions

Algorithm for the *do*-calculus when the graph is unknown

- determine the equivalence class implicitly using a SAT-solver
- query one solution graph G
- run the Tian-Shpitser-algorithm on G to determine whether the causal effect *P(y|do(w))* is determined for G
- if it is, determine which *do*-calculus rules were applied and record the constraints $C_1, \ldots, C_n$ that were used
  - add $\neg C_1 \vee \ldots \vee \neg C_n$ as a constraint to refine the current equivalence class
- if not, determine the "hedge" H and add $\neg H$ to refine the current equivalence class
- repeat until the equivalence is exhausted
- return the set of estimates of the causal effect and NA if it cannot be determined in one member of the equivalence class

Comparison of our approach to enumeration



In sum: *do*-calculus using a SAT-solver

- enables computation of the causal effect when the graph structure is underdetermined

In sum: *do*-calculus using a SAT-solver

- enables computation of the causal effect when the graph structure is underdetermined

➡ how should one estimate a causal effect when the equivalence class of causal structures was determined on the basis of a set of conflicted constraints?

In sum: *do*-calculus using a SAT-solver

- enables computation of the causal effect when the graph structure is underdetermined

➡ how should one estimate a causal effect when the equivalence class of causal structures was determined on the basis of a set of conflicted constraints?

- some avenues one can explore with the query-based approach:
  - explore more closely the conditions involved in determining the causal effect
  - find multiple different estimators
  - even though the overall graph structure may not be determinable without resolving conflicts, some causal effects may be

Conclusion

- the use of general purpose SAT-solvers provides an extraordinarily versatile tool for causal discovery
- it opens new avenues for handling background knowledge and the computation of causal effects when the causal structure is underdetermined
- it provides a query based approach in contrast to a representation of an equivalence class of causal structures
- it suggests that current general purpose constraint solvers outperform domain specific approaches

References

- Hyttinen, Eberhardt & Järvisalo (2015). *Do-calculus when the true graph is unknown.* UAI 2015.
- Hyttinen, Eberhardt & Järvisalo (2014). *Constraint-based Causal Discovery: Conflict Resolution with Answer Set Programming*. UAI 2014.
- Hyttinen, Hoyer, Eberhardt & Järvisalo (2013). *Discovering Cyclic Causal Models with Latent Variables: A General SAT-Based Procedure*. UAI 2013.
- {Hyttinen, Plis, Danks, Eberhardt & Järvisalo} (work in progress). *Causal Discovery from Subsampled Time Series Data by Constraint Optimization.*

Other relevant work that is closely related:

- Triantafillou & Tsamardinos (2015). *Constraint-based Causal Discovery from Multiple Interventions Over Overlapping Variable Sets*. JMLR 16(Nov):2147−2205.
- Claassen & Heskes (2011). *A logical characterization of constraint-based causal discovery*. UAI 2011.
- Triantafillou, Tsamardinos & Tollis (2010). *Learning Causal Structure from Overlapping Variable Sets*. AISTATS 2010.

Thank you!