GADAG : un paquet R dédié à l'inférence de Graphes Acycliques Dirigés par maximum de vraisemblance pénalisé

Magali Champion, Victor Picheny et Matthieu Vignes

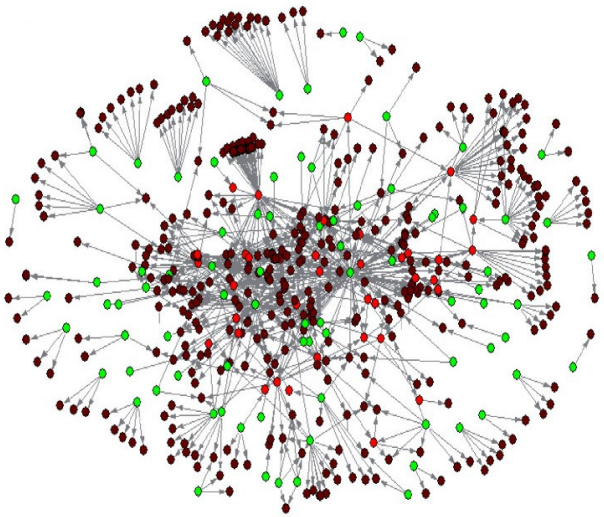
Rencontres R - Anglet

30 Juin 2017

Context and motivations

Main goal : graph inference to understand how a set of variables interact each other.

$\rightarrow$ *Example : gene regulatory network inference*



Highlights

- High dimension :
  number of nodes $p >> n$ sample size
- Sparsity : control the number of edges s
- DAG constraint.

Mathematical modelling : Variables $X^1, \ldots, X^p$, associated to nodes $1, \ldots, p$ of the DAG $\mathcal{G}_0$ are **linearly** linked

$$\forall j \in [\![1, p]\!], \quad X^j = \sum_{i=1, i \neq j}^{p} (G_0)_i^j X^i + \varepsilon^j.$$

The penalized maximum likelihood estimator

Network inference through the **penalized** maximum likelihood estimator :

$$\hat{G} = \underset{G \in \mathcal{G}_{DAG}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{p} \sum_{k=1}^{n} \left((X - XG)_k^j \right)^2 + \boldsymbol{\lambda} \sum_{\boldsymbol{i,j=1}}^{\boldsymbol{p}} \left| \boldsymbol{G_i^j} \right| \right\},$$

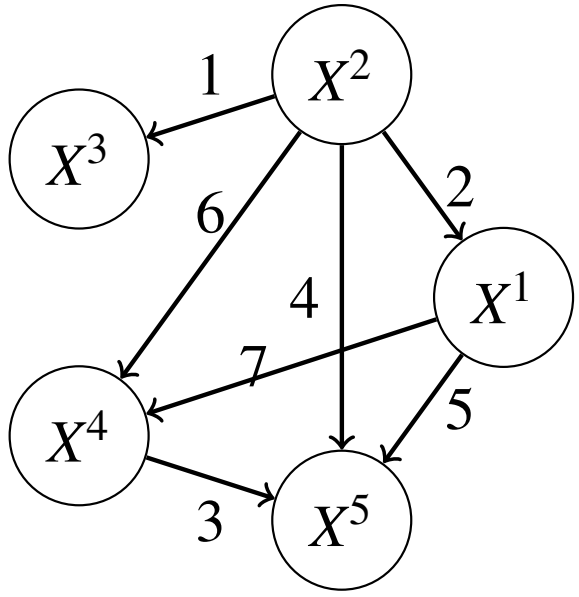
where $\mathcal{G}_{DAG}$ is the set of matrices compatible with a DAG.

DAG decomposition

A matrix G is compatible with a DAG if and only if :

$$G = PTP^T,$$

where P and T are permutation and lower triangular matrices.



The penalized maximum likelihood estimator

Network inference through the **penalized** maximum likelihood estimator :

$$\hat{G} = \underset{G \in \mathcal{G}_{DAG}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{p} \sum_{k=1}^{n} \left((X - XG)_k^j \right)^2 + \boldsymbol{\lambda} \sum_{\boldsymbol{i,j=1}}^{\boldsymbol{p}} \left| \boldsymbol{G_i^j} \right| \right\},$$

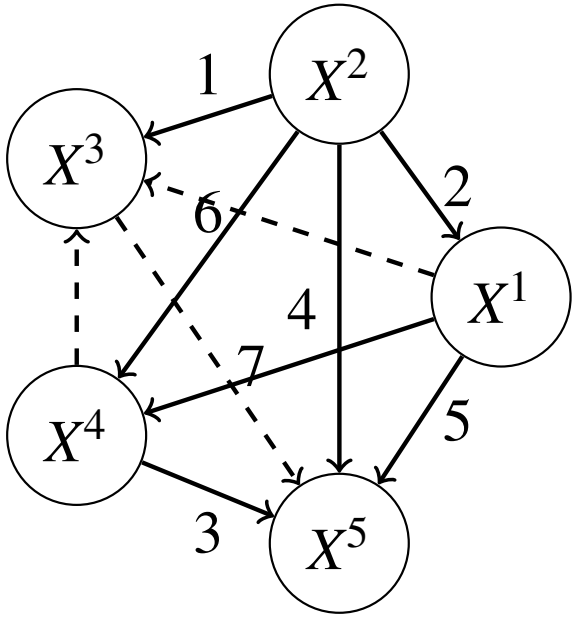
where $\mathcal{G}_{DAG}$ is the set of matrices compatible with a DAG.

DAG decomposition

A matrix G is compatible with a DAG if and only if :

$$G = PTP^T,$$

where P and T are permutation and lower triangular matrices.



The penalized maximum likelihood estimator

Network inference through the **penalized** maximum likelihood estimator :

$$\hat{G} = \underset{G \in \mathcal{G}_{DAG}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{p} \sum_{k=1}^{n} \left((X - XG)_k^j \right)^2 + \boldsymbol{\lambda} \sum_{\boldsymbol{i,j=1}}^{\boldsymbol{p}} \left| \boldsymbol{G_i^j} \right| \right\},$$

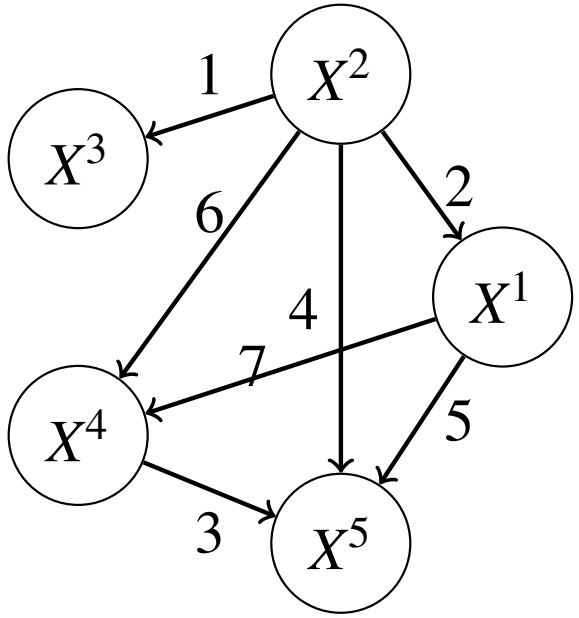
where $\mathcal{G}_{DAG}$ is the set of matrices compatible with a DAG.

DAG decomposition

A matrix G is compatible with a DAG if and only if :

$$G = PTP^T,$$

where P and T are permutation and lower triangular matrices.



The penalized maximum likelihood estimator

Network inference through the **penalized** maximum likelihood estimator :

$$(\hat{P}, \hat{T}) = \underset{P \in \mathbb{P}_p, T \in \mathbb{T}_p}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{p} \sum_{k=1}^{n} \left((X - XPTP^T)_k^j \right)^2 + \boldsymbol{\lambda} \sum_{\boldsymbol{i,j=1}}^{\boldsymbol{p}} \left| \boldsymbol{T_i^j} \right| \right\},$$

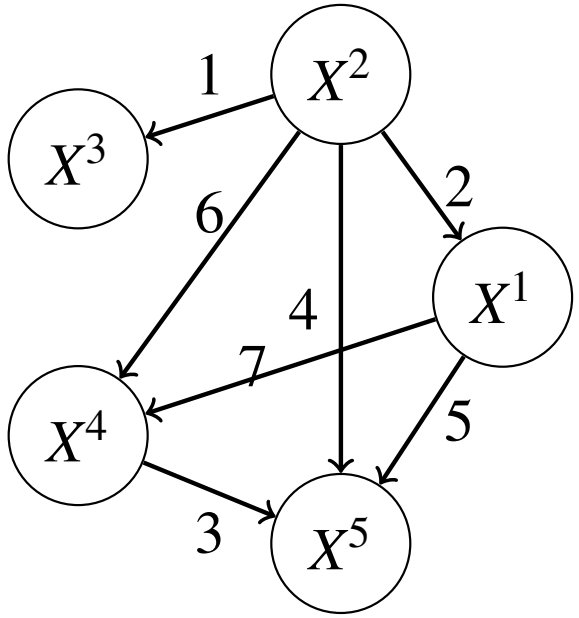
where $\mathbb{P}_p$ and $\mathbb{T}_p$ are the set of permutation and triangular matrices.

DAG decomposition

A matrix G is compatible with a DAG if and only if :

$$G = PTP^T,$$

where P and T are permutation and lower triangular matrices.



GADAG : a hybrid genetic algorithm for learning DAGs

$$(\hat{P}, \hat{T}) = \underset{P \in \mathbb{P}_p, T \in \mathbb{T}_p}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{j=1}^{p} \sum_{k=1}^{n} \left((X - XPTP^T)_k^j \right)^2 + \lambda \sum_{i,j=1}^{p} \left| T_i^j \right| \right\} \quad (E)$$

1. $\hat{P} \in \mathbb{P}_p(\mathbb{R})$ being fixed, the optimization problem is convex and we can find $\hat{T} \in \mathbb{T}_p(\mathbb{R})$ solution of (E).

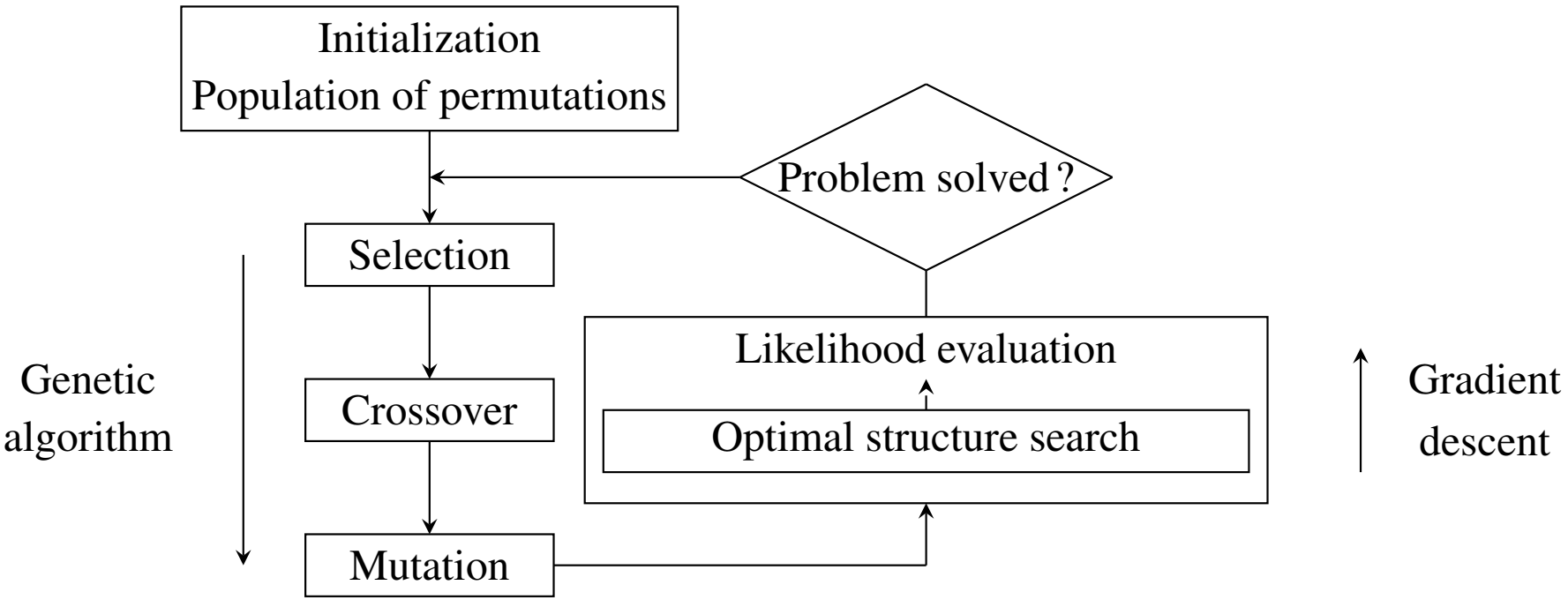
   $\rightarrow$ *ad-hoc gradient descent algorithm to solve it*

2. Exploring the space of permutations is difficult due to its dimension and structure (discrete and non-convex).

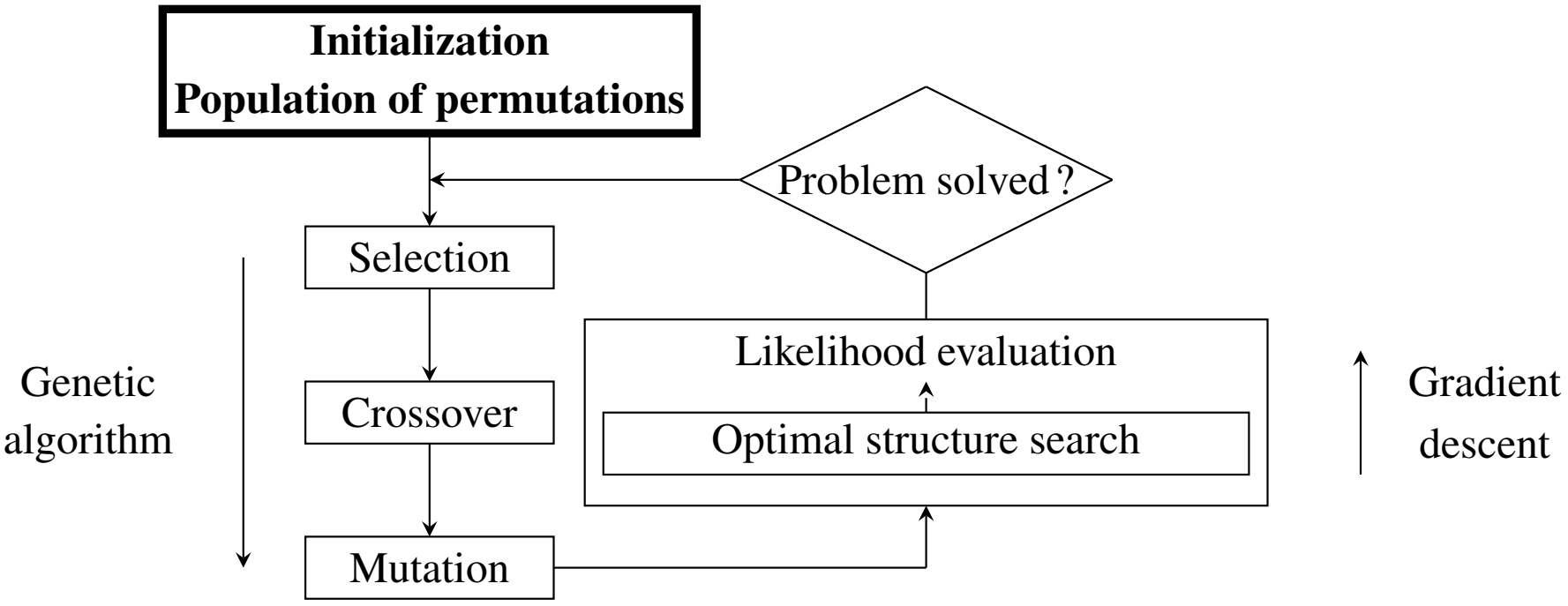
   $\rightarrow$ *genetic algorithm to explore the set of permutations*

GADAG : a hybrid genetic algorithm for learning DAGs

$$(\hat{P},\hat{T}) = \underset{P\in\mathbb{P}_p, T\in\mathbb{T}_p}{\operatorname{argmin}} \left\{ \frac{1}{n}\sum_{j=1}^{p}\sum_{k=1}^{n}\left((X - XPTP^T)_k^j\right)^2 + \lambda \sum_{i,j=1}^{p}\left|T_i^j\right| \right\} \quad (E)$$

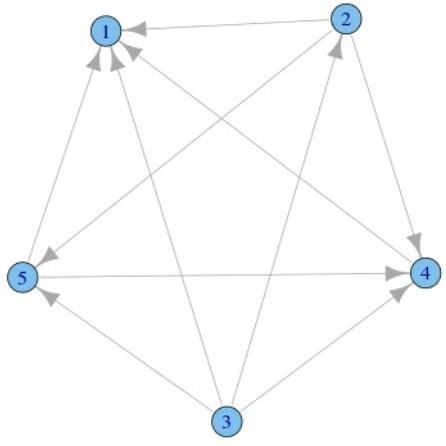


GADAG : a hybrid genetic algorithm for learning DAGs
- Initialization step -



GADAG : a hybrid genetic algorithm for learning DAGs - Initialization step -

- Population of N "individuals" $(x_1, \ldots, x_N)$
- Each individual x_1 is a "chromosomose" representation of a permutation P

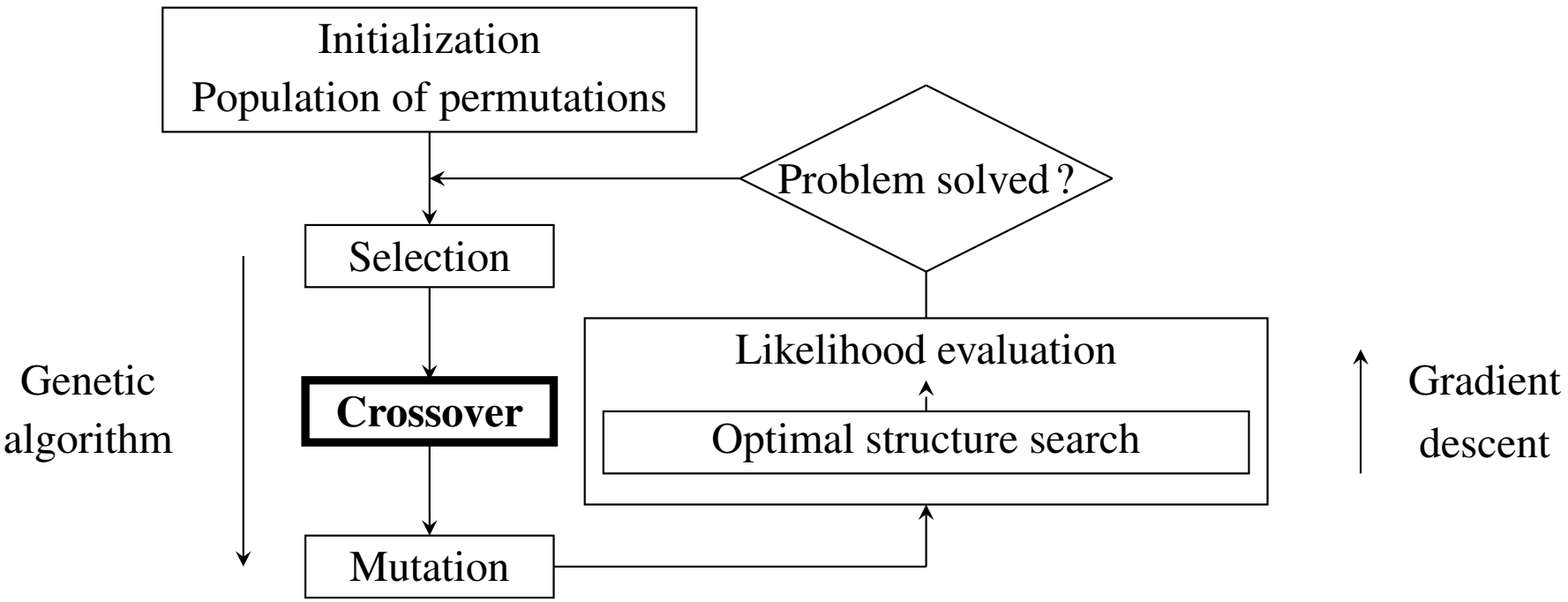


the node with the most children

1 4 5 2 3

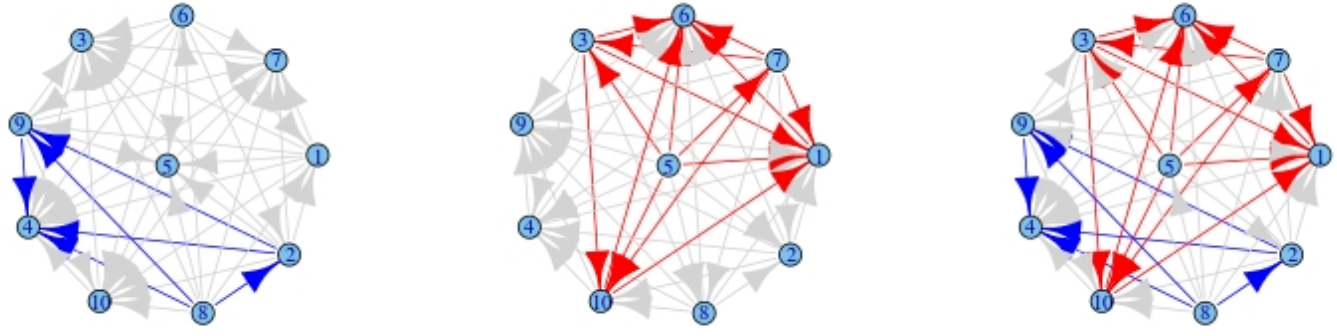
```
create.population(p, pop.size=5*p)
```

GADAG : a hybrid genetic algorithm for learning DAGs
- Crossover -



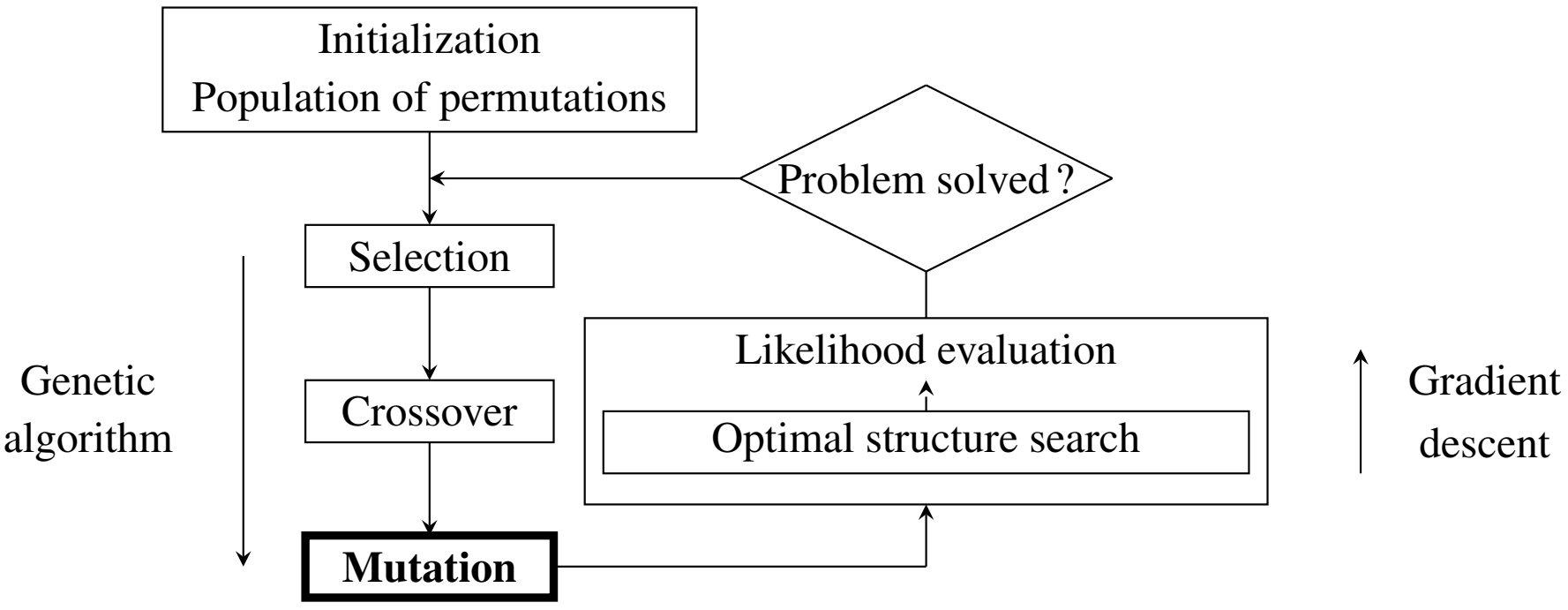
GADAG : a hybrid genetic algorithm for learning DAGs
- Crossover -

	4	3	10	7	5	9	1	2	6	8
+	6	1	~~9~~	~~4~~	10	~~2~~	~~8~~	3	7	5
=	4	*	*	*	*	9	*	2	*	8



```
crossover(Pop, p.xo=0.25)
```

GADAG : a hybrid genetic algorithm for learning DAGs
- Mutation -

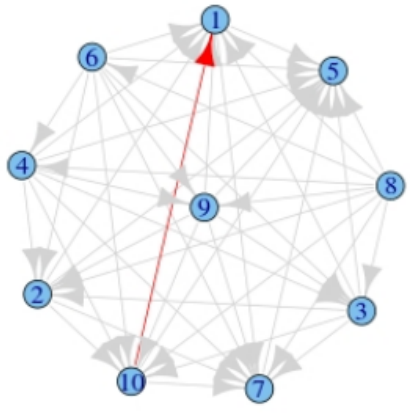


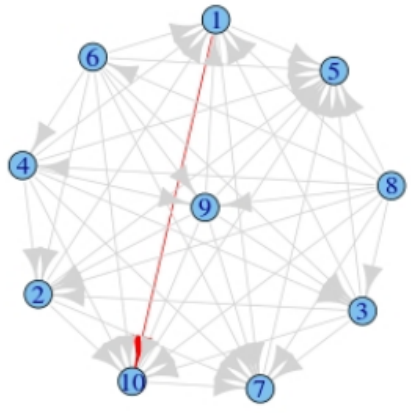
GADAG : a hybrid genetic algorithm for learning DAGs
- Mutation -

5 7 1 10 2 3 9 4 6 8

↓

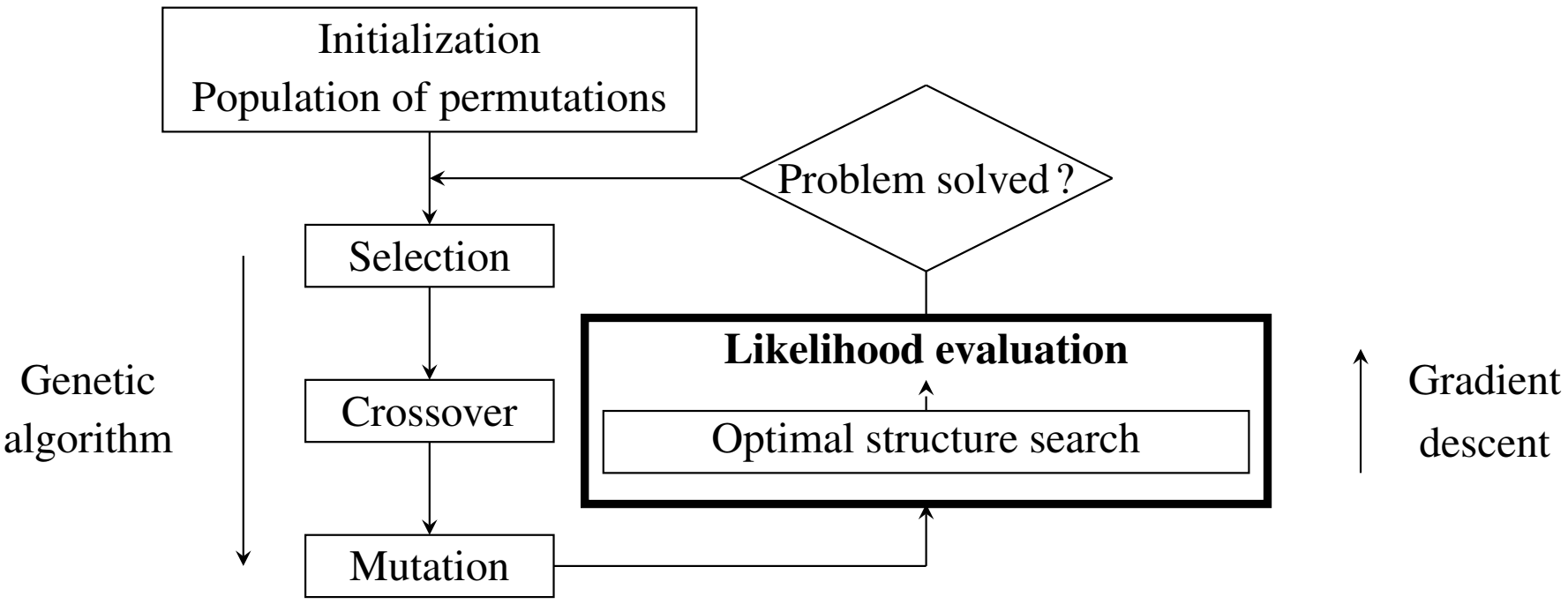
5 7 10 1 2 3 9 4 6 8






```
mutation(Pop, p.mut=0.05)
```

GADAG : a hybrid genetic algorithm for learning DAGs
- Likelihood evaluation -



GADAG : a hybrid genetic algorithm for learning DAGs - Likelihood evaluation -

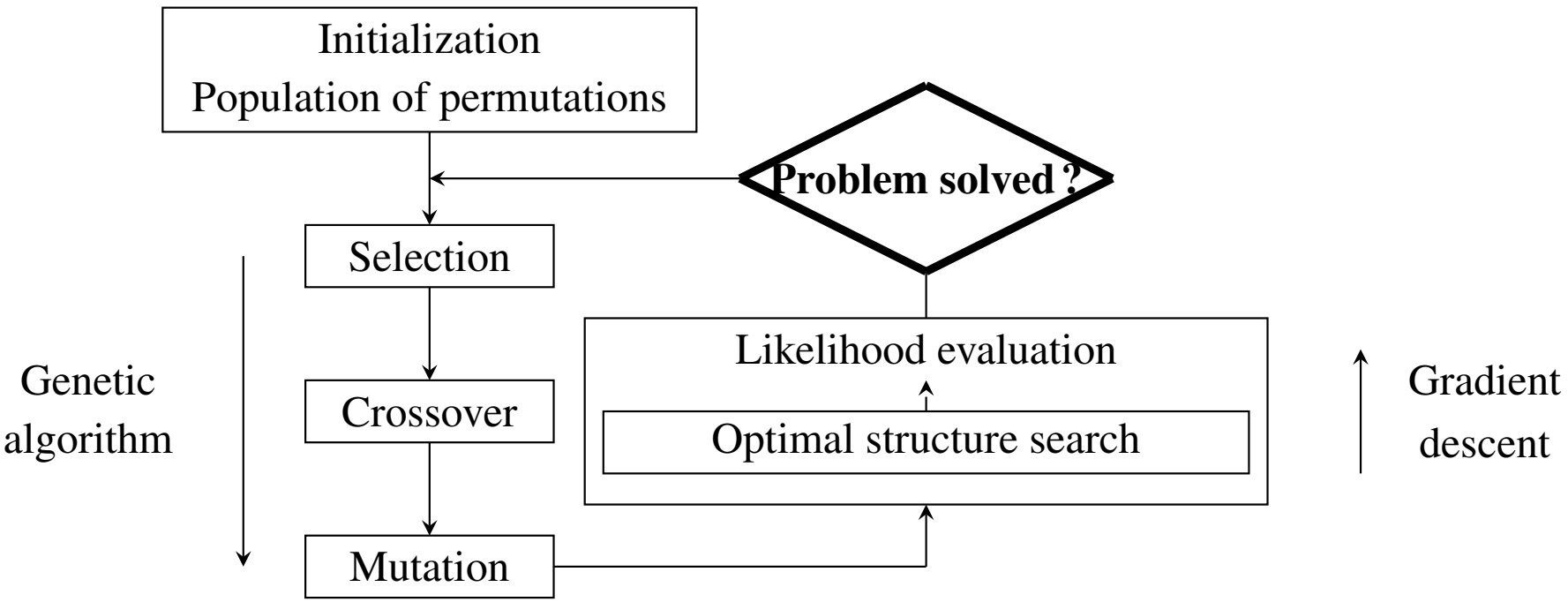
- Optimal structure search (best T with P fixed) using the evaluation function :

```
evaluation(Pop, X, lambda, grad.control =
list(tol.obj=1e-6, max.ite=50), ncores=1)
```

- Likelihood evaluation :

```
fitness(P, X, T, lambda)
```

GADAG : a hybrid genetic algorithm for learning DAGs
- End of the algorithm -



GADAG : a hybrid genetic algorithm for learning DAGs - End of the algorithm -

The algorithm ends if one of the three condition holds :

- the mean of the objective function no longer changes,
- we reach the maximal number of population generations (`n.gen`) or the maximal number of calls of the evaluation function (`max.eval`),
- the entropy of Shannon of the population, measuring the heterogeneity of the population, converges toward 0 (`tol.Shannon`)

```
GADAG_Run(X, lambda, GADAG.control =
list(n.gen = 100, tol.Shannon = 1e-6, max.eval
= 1e4, pop.size = 10, p.xo = 0.25, p.mut =
0.05))
```

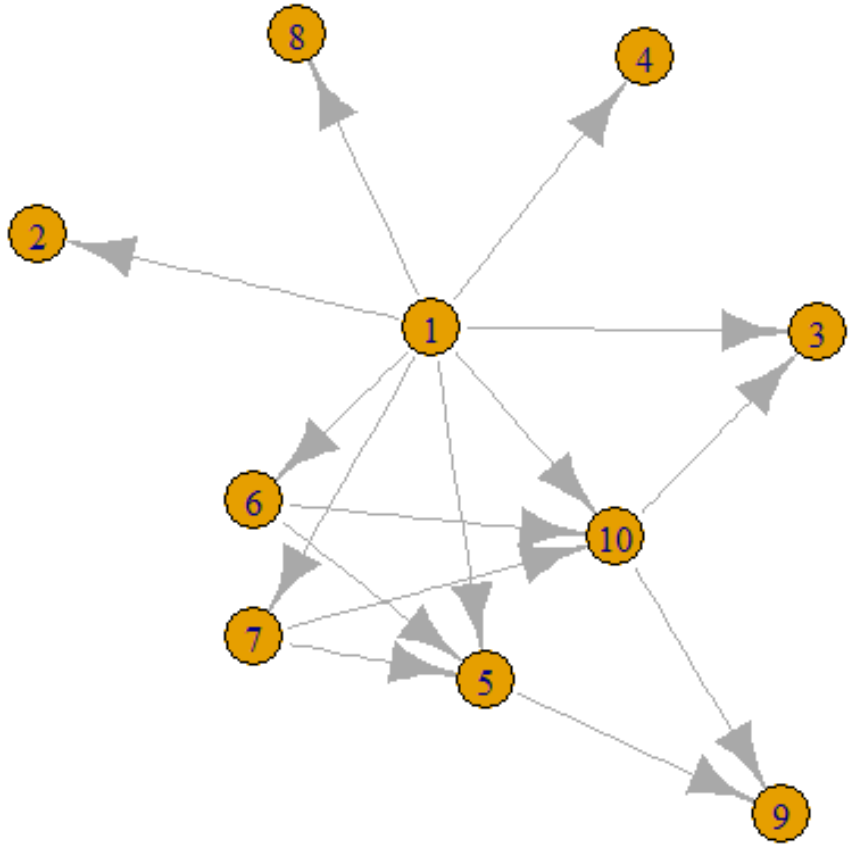
Numerical results

```
GADAG_Analyze(GADAG_results, G, X)
```

Numerical results

`GADAG_Analyze(GADAG_results, G, X)`

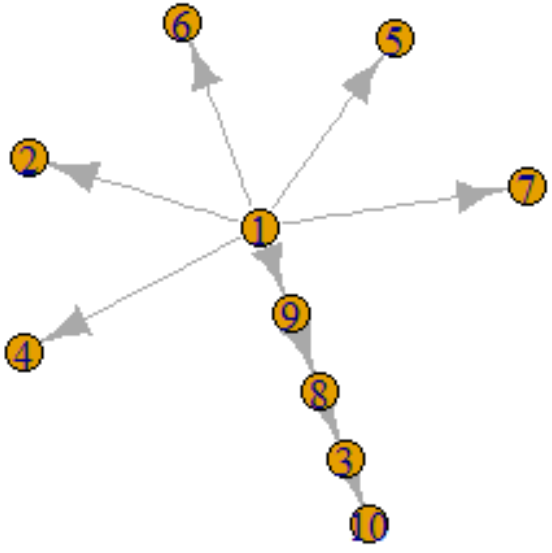
1. You know **nothing** !

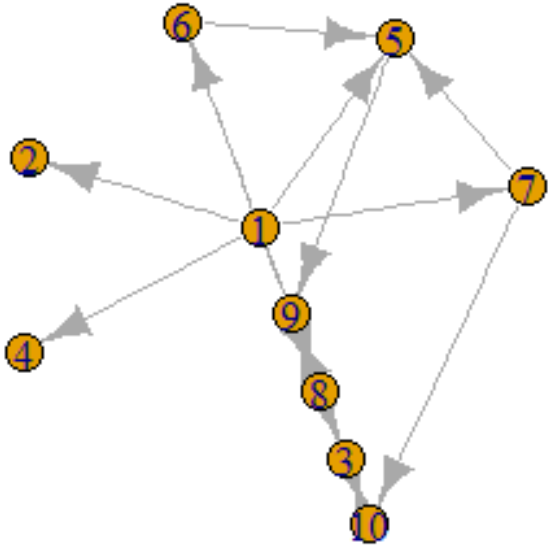


Numerical results

`GADAG_Analyze(GADAG_results, G, X)`

2. You know the truth !



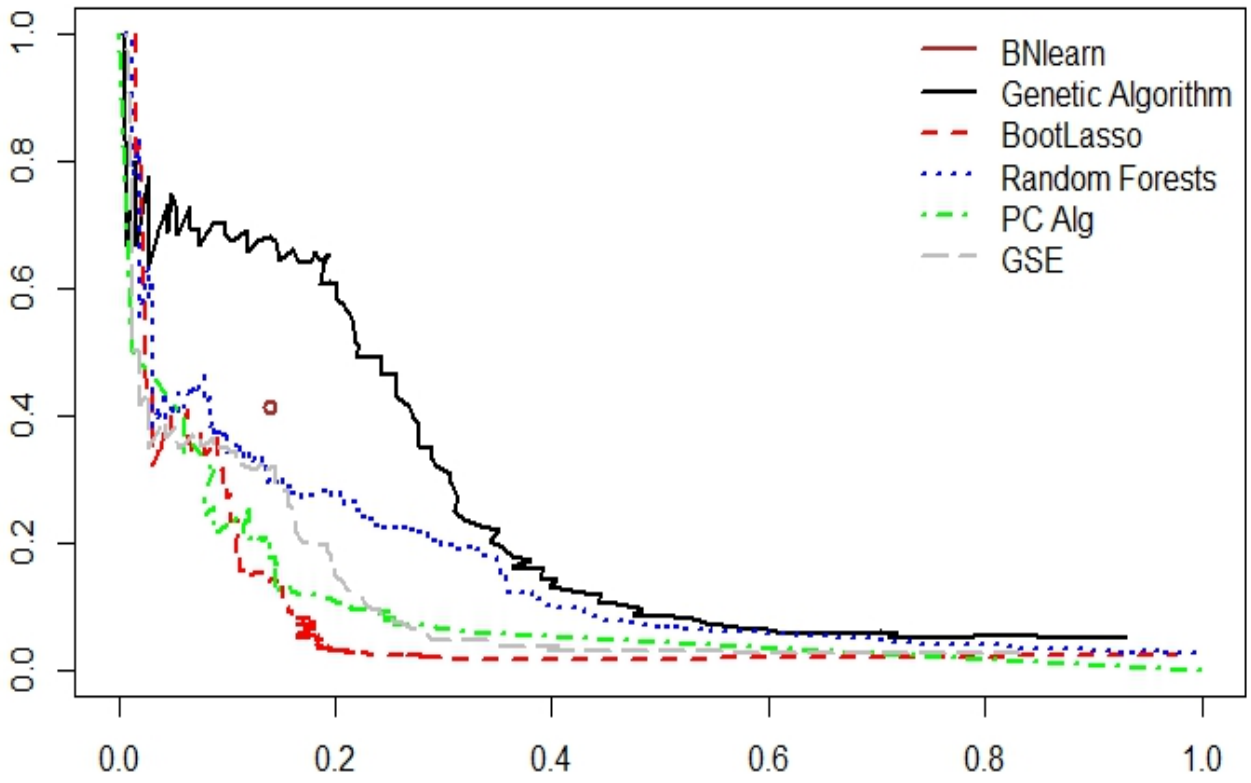


FP	FN	TP	TN	precision	recall
7	1	8	74	0.5333	0.8889

Numerical results

`GADAG_Analyze(GADAG_results, G, X)`

3. You want more !



Conclusion

Inferring causal relationships in GRN is a very challenging problem, mainly due to :

- identifiability (strong assumption on the noise variances),
- exploration of the set of DAGs (NP-problem).

The hybrid genetic algorithm we propose provides powerful results in the context of GRN inference but still needs improvements :

- proof of convergence,
- application to very-high dimensional DAGs (to be done),
- extension to the non-identifiable case (use of intervention data).

Thank you for your attention !

- M. Champion, V. Picheny, M. Vignes. Inferring large graphs using ℓ_1-penalized likelihood. Submitted.
- B. Liu, A. de la Fuente, I. Hoeschele. Gene network inference via structural equation modeling in genetical genomics experiments. Genetics, 178 : 1763–1776, 2008.
- J. Peters, P. Bühlmann. Identifiability of Gaussian structural equation models with equal error variances. Biometrika, 101 :219-228, 2014.
- R. Tibshirani. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society, 58(1) : 267-288, 1996.
- S. van de Geer, P. Bühlmann. ℓ_0-penalized maximum likelihood for sparse directed acyclic graphs. The Annals of Statistics,41(2) : 536-567.