Computational Statistics - Assignment 1

Cay Rahn, David Pomerenke, Alexander Prochnow

I. Introduction

In various disciplines, Genetic Algorithms are applied to problems where the search space is too large for exact search algorithms to find optimal solutions in a reasonable time. In Causal Discovery, we have such a problem, since we are searching for the most plausible causal graph in a set of all possible graphs for a given dataset.

Therefore, for Causal Discovery on the dataset `a1_data`, we chose to develop a Genetic Algorithm (GA), where each individual represents a possible Causal Graph. Over time, a population of these individuals will converge towards graphs that describe the causal structures within the data better when a fitness function is chosen to facilitate this trend.

In this report, we will first state the assumptions made by our GA. Next, we will describe the procedure as well as the design choices made during its development, followed by details on the implementation. Finally, we will present our results as well as limitations of our procedure.

II. Assumptions

- Causal graph in the shape of a **directed acyclic graph** (DAG). This is a standard assumption in causal inference.
- **Linear additive models**: Every variable has values $Y = \beta \cdot \text{parents}(Y) + \epsilon$. This allows us to use linear regression.
- The data is **not standardized**, so *sortnregress* can be applied as a baseline.
- Criteria for a good model:
  - **Small residuals** ϵ are good, because they show that information can be inferred from other variables.
  - **Small variances** of the residuals ϵ are good, because they show that – beyond an intercept – information can be inferred from other variables.
  - **Sparse graphs** are good, because they are easily interpretable.
  - All variables should be **weighted equally** for evaluation. They are of a **similar scale**, so we can compare values and variances of the variables and of resulting residuals.
- The data may contain **hidden sub-groups** to which varying causal mechanisms apply to some extent. Nonetheless, we assume that the data can be modeled by a single DAG.

III. Procedure

A genetic algorithm such as the implemented one is a procedure working similarly to biological evolution. A start population is created, altered and improved through several generations. Mutations and crossovers lead to new solutions based on the previous generation of solutions and eventually the procedure converges to the fittest, meaning the best solution.

The *sortnregress* procedure proposed in the paper Reisach, Seiler, and Weichwald [1] showed to be a simple baseline method to discover a DAG of a dataset. In our implemented GA, a similar procedure is used to create the starting population. Randomness and therefore variation between the individuals is achieved using a Lasso-Regression with a randomized regularization parameter.

The starting population is further diversified by adding DAGs that have been fitted on single clusters of the data points. In section III-A we identify 13 clusters, and for each cluster we discover a DAG using *sortnregress*, which we then add to the population.

The original sortnregress algorithm uses a Linear Regression followed by a model selection with the use of Lasso-Lars with Bayesian information criterion. Because this does not allow for variation that can be randomized, the randomized procedure was used. Both procedures are comparable in their base ideas but the randomized procedure does not guaranty to find the best possible DAG but to provide different alternatives.

A combination of two individuals is achieved by including all of the edges in the resulting child that are present in both parent individuals. If an edge is only present in one of the parents, it is decided randomly whether the edge should be included in the child. To avoid cyclic graphs, an edge is only included if it does not create a cycle in the graph[2]. This procedure ensures that connections found by multiple individuals are included in the next generation as well. Eventually this will result in an individual that contains all edges that are very likely.

To improve the procedure and avoid local optima mutations are used to introduce randomness in the process. The most simple option is to remove a randomly chosen edge from the DAG. This always keeps the graph acyclic. More complex mutations that may improve the procedure are to allow mutations to remove a random number of edges or to add edges randomly. The latter is promising to work against the trend of removing more and more edges due to mutations. Due to the lack of time it was not implemented in the scope of this assignment. It would also create more variation in a population to overcome local minima.

The fitness of the individuals is evaluated using a the MSE per random variable when it is predicted according to the DAG using the variables that are found as causes. Since there are

multiple nodes for which the MSE is calculated, a summarized value is obtained using the mean of the MSE for all variables.

As a second fitness value the total number of edges in the DAG is considered. This second fitness values allows to include a sparsity constraint. While sparsity is not a necessarily needed property of the true underlying DAG, the evaluation based on MSE tends to favor more highly connected DAGs. Therefore the sparsity constraint is included in the final fitness evaluation by weighting and adding it to the MSE.

An alternative approach is to fit the weight of the DAG to the data as before, but then to calculate the variance of the residuals, i.e. prediction errors.

A. *Clustering*

Inspection of the scatterplots reveals that the dataset contains distinct clusters. We run two clustering algorithms – the very common k-means algorithm, and spectral clustering, which is suitable for discrete connected shapes, as they can be visually detected in the scatterplots – and let them partition the data into $k = 2...30$ clusters. We evaluate the results based on their silhouette scores, which measure inter- and intra-cluster distances and can serve as an unsupervised measure for the fit of a set of clusters. The highest silhouette score (0.2875) is achieved for $k = 13$ clusters with spectral clustering. These clusters are also visually salient, see for example the histogram and scatterplot between variables I and J:

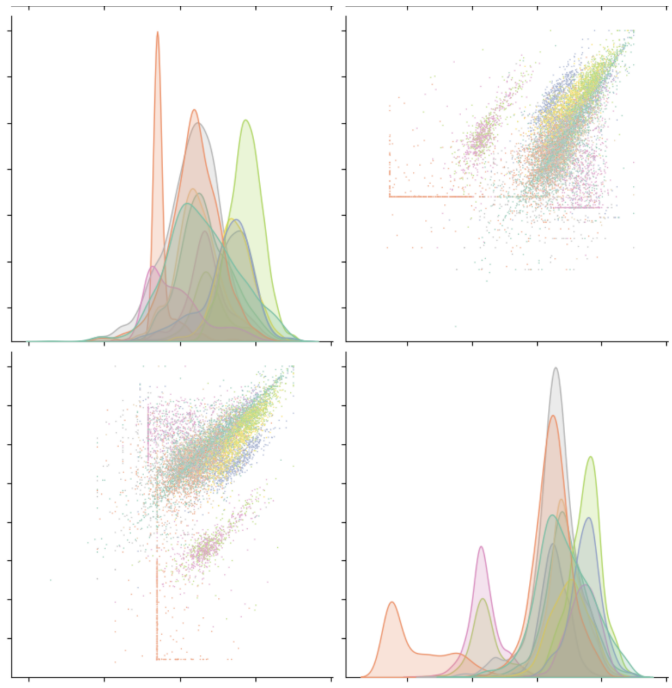

Fig. 1. 13 clusters, shown here for variables I and J and the scatterplot between them.

Since in some scatterplots certain clusters show a higher correlation than others, we hypothesize that the clusters have an effect on the causal mechanism between the variables. We find it hard to model such an effect. As a simple test for the relevance of the clusters for the causal effects, we add a dummy variable for each cluster (that is, we add 13 columns to the dataset where the value is 1 iff. the data point is contained in the respective cluster and 0 otherwise). We observe that a) Lasso regression makes use of some of these dummy variables, and b) when discovering a DAG with *sortnregress* on only the data points of a single cluster, the resulting DAGs differ slightly. We do not see a feasible way to model a "multidimensional DAG", but at least we can use these discovered DAGs as additional plausible starting points for our evolutionary algorithm (see section III).

algorithm	α	MSE(ϵ)	Var(ϵ)	n_edges
sortnregress	–	1.27	0.208	36
sortnregress-lasso	0.035	5.29	0.231	7
sortnregress-lasso	0.07	8.17	0.289	3
evolutionary algorithm	–	1.115	0.201	30

TABLE I
RESULTS.

We give 3 evaluation measures, the mean of the mean squared errors, the variance of the residual at each node, and the number of edges in the graph. Lower values are better in all cases.

IV. IMPLEMENTATION

For implementation, the Python package `deap`[3] is used. It provides a toolbox for Genetic Algorithms to customize mating, mutation and the evaluation of individuals. The code written in the scope of this assignement can be found at https://github.com/Zianor/computational-statistics-1.

The foundation for a hyperparameter search using the Python framework `optuna` [4] is given but not used yet due to the lack of time.

V. RESULTS

In general, the implemented algorithm converged very quickly (in 8 generations). This indicates that the implemented mutation, the removal of a random edge, is not enough to create variation in the population. Another reason might be a non-diverse starting population. A variation of the weighting of both fitness values might also improve the results.

In the results table we can see that our adapted *sortnregress* algorithm using Lasso regression outperforms the original with respect to the number of edges and the MSE, but only comes close with respect to the variance of the residuals. A reduction in edges from 40 to 7 goes along with an increase in variance of only 30%. However, the mean variance of the original variables is only 0.2245, so one can question whether variance is the right objective after all.

The evolutionary algorithm converges towards a high number of edges, typically between 20 and 40. Occasionally it decreases the number of edges, but we could not tune it to do so reliably. The reason for this is most likely the rather simple mutations and a lack of hyperparameter tuning due to time constraints. This can be improved in further work.

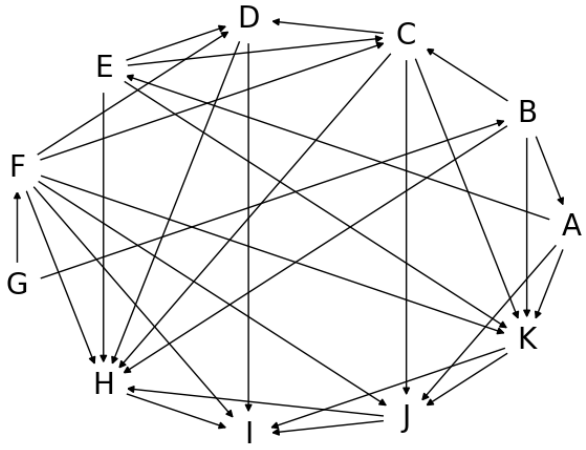

Fig. 2. Causal graph returned from our GA as best candidate. This is the graph mentioned in table I with 30 edges.

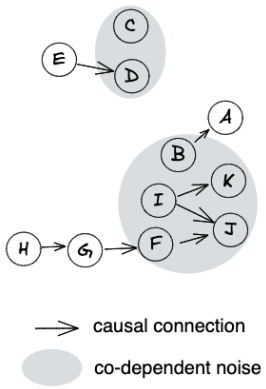

Fig. 3. A sparse graph obtained via sortnregress-lasso for $\alpha = 0.035$. Arrows symbolize causal connections, a grey background indicates connected patterns in the noise, possibly hinting at a common confounder (cf. the end of section VI.).

VI. Limits of our procedure

The main limitation of our algorithm is the dependence on the *sortnregress* algorithm, which comes with several limitations itself. This limitation can be removed by the use of several different algorithms for causal discovery for population initialization. The implementation of this was not possible in the scope of the assignment.

A comparison based on the MSE per node in the resulting DAGs makes the procedure prone to misleading results by different data scales. This is also a limitation given by the use of the sortnregress algorithm[5] for population initialization.

Because sortnregress is used in our procedure, there is also the chance that standardized data performs worse though the genetic algorithm developed works against this weakness of sortnegress.

If the made assumptions about the linear additive model does not hold, the algorithm is not able to produce meaningful results. This may be the case for real-life applications.

Since the quality of the results that the GA produces heavily depend on settings of hyperparameters, e.g. mutation rates, and randomness, e.g. through the initial population, it is not directly applicable to other problems. Instead, for each problem, new hyperparameters must be found in order to achieve good results.

One advantage of the developed approach is that the resulting DAG is not necessarily fully connected.

The data contains a special kind of patterns that we can identify, but cannot fully explain. An example can be seen in Figure 1. In both variables I and J (as well as some others), there is an accumulation of duplicate values. These values, naturally, are more dispersed in the respectively other variable, presumably due to randomly distributed data points that have been added to it. Curiously, the lines touch each other and form a 2-dimensional rectangle. A process for the creation of such a pattern could be that a 2-dimensional threshold has been applied to a 2-dimensional random distribution. This suggests that the variables participating in the pattern are neither independent, nor a cause-effect pair, but have been caused (in a sense beyond the additive linear model) by a common confounder; or that it really is a single multidimensional variable rather than two separate variables. However, these speculations only apply to special clusters of the data, and the largest part of the data suggests that there is indeed a simple correlation between the variables. So we can only conclude that the data has been created by multiple very different processes.

VII. Conclusion

References

[1] Alexander Reisach, Christof Seiler, and Sebastian Weichwald. "Beware of the simulated dag! causal discovery benchmarks may be easy to game". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 27772–27784.

[2] Shuyu Dong and Michèle Sebag. *From graphs to DAGs: a low-complexity model and a scalable algorithm*. 2022. DOI: 10.48550/ARXIV.2204.04644. URL: https://arxiv.org/abs/2204.04644.

[3] Félix-Antoine Fortin et al. "DEAP: Evolutionary Algorithms Made Easy". In: *Journal of Machine Learning Research* 13 (July 2012), pp. 2171–2175.

[4] Takuya Akiba et al. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019.

[5] Zhengyin Chen, Kun Liu, and Wenpin Jiao. "A Genetic Algorithm for Causal Discovery Based on Structural Causal Model". In: *Artificial Intelligence*. Ed. by Lu Fang et al. Cham: Springer Nature Switzerland, 2022, pp. 39–54. ISBN: 978-3-031-20503-3.

APPENDIX

The 13 clusters identified via spectral clustering (we just think the colours are beautiful):

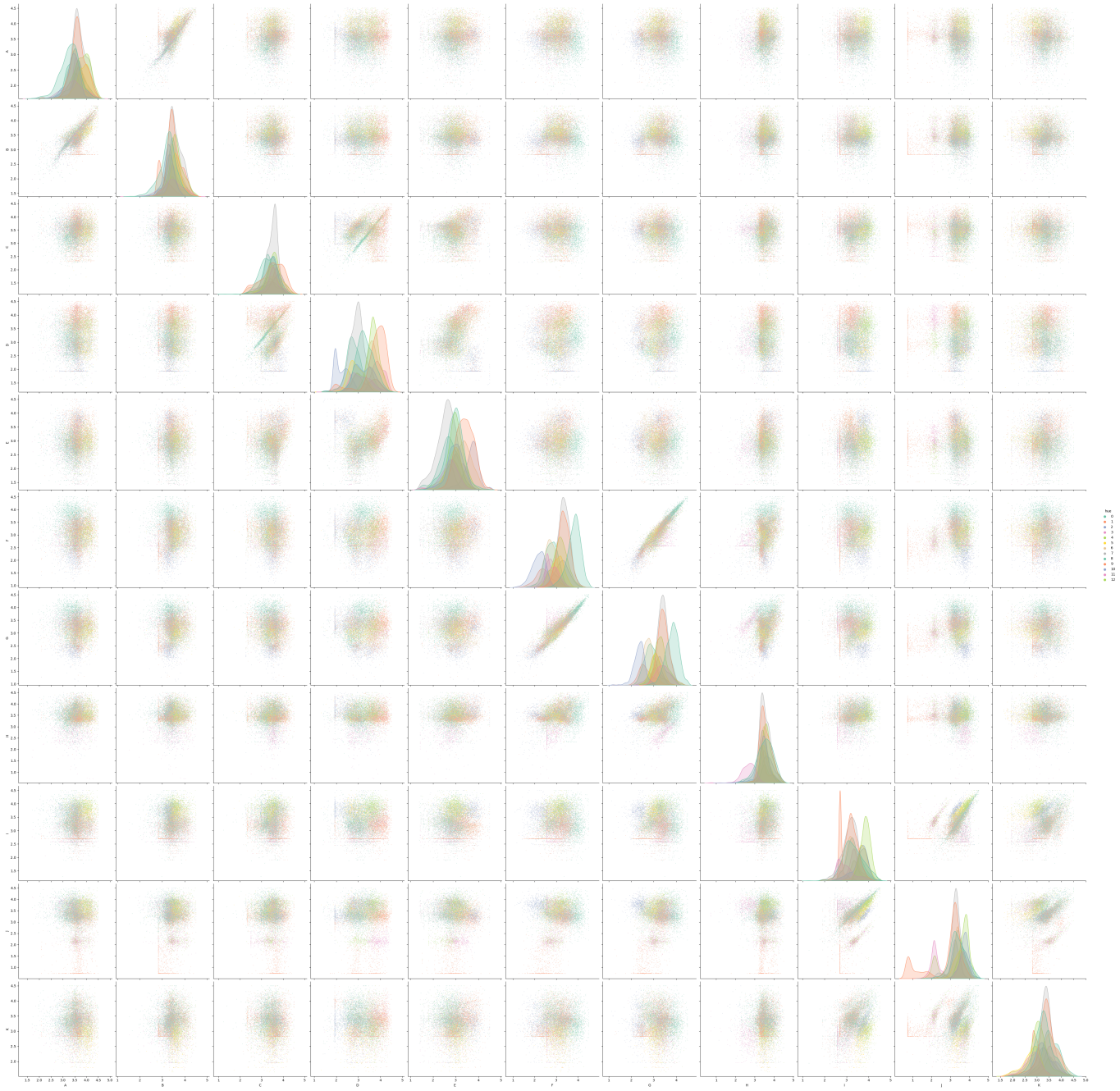

Fig. 4. Spectral custering result