BAYESIAN NETWORK QUANTIZATION METHOD AND STRUCTURAL LEARNING

Rafael Rodrigues Mendes Ribeiro

UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

Rafael Rodrigues Mendes Ribeiro

Bayesian Network Quantization Method and Structural Learning

São Carlos

2023

Rafael Rodrigues Mendes Ribeiro

Bayesian Network Quantization Method and Structural Learning

Tese apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo, para obtenção do título de Doutor em Ciências, Programa de Pós-Graduação em Ciências da Engenharia Eletrica.

Área de concentração: Sistemas Dinâmicos

Advisor: Prof. Dr. Carlos Dias Maciel

São Carlos

2023

Catalog card prepared by Patron Service at "Prof. Dr. Sergio Rodrigues Fontes" Library at EESC/USP

```
        Ribeiro, Rafael Rodrigues Mendes
R484b      Bayesian network quantization method and structural
        learning / Rafael Rodrigues Mendes Ribeiro; Dissertation
        directed by Carlos Dias Maciel. -- São Carlos, 2023.

            Doctoral (Dissertation) - Graduate Program in
        Electrical Engineering and Research Area in Dynamic
        Systems -- São Carlos School of Engineering of the
        University of São Paulo, 2023.

            1. Bayesian network. 2. Structural learning.
        3. Quantization. 4. Evolutionary algorithm. I. Title.
```

Elena Luzia Palloni Gonçalves – CRB 8/4464

FOLHA DE JULGAMENTO

Candidato: Engenheiro **RAFAEL RODRIGUES MENDES RIBEIRO**.

Título da tese: "Método de Quantização para Redes Bayesianas e Aprendizagem Estrutural de Redes Bayesianas".

Data da defesa: 05/02/2024.

Comissão Julgadora	**Resultado**
Prof. Associado Carlos Dias Maciel **(Orientador)** (Escola de Engenharia de São Carlos/EESC-USP)	APROVADO
Prof. Titular André Carlos Ponce de Leon Ferreira de Carvalho (Instituto de Ciências Matemática e de Computação/ICMC-USP)	APROVADO
Prof. Dr. Cassio Polpo de Campos (Eindhoven University of Technology)	APROVADO
Prof. Dr. Eduardo Antônio Barros da Silva (Universidade Federal do Rio de Janeiro/UFRJ)	APROVADO
Prof. Associado Ricardo Zorzetto Nicoliello Vencio (Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto/FFCLRP-USP)	APROVADO

Coordenador do Programa de Pós-Graduação em Engenharia Elétrica:
Prof Associado **Marcelo Andrade da Costa Vieira**

Presidente da Comissão de Pós-Graduação:
Prof. Titular **Carlos De Marqui Junior**

ACKNOWLEDGEMENTS

I would like to thank my family for all the support during all these years. My mother by always wanting to read my papers, even though she didn't understand them, and always saying that she thought that the figures were pretty. My sisters Leticia and Marina by helping me deal with the stress of doing research and the publishing procedures. My father by always having a series or film to recommend for me to relax.

I would like to thank my friends for always keeping me company, no matter how far apart we were, and for all the games we played together.

To my colleagues at the Signal Processing Laboratory for all the help and companionship. My thesis would not be the same without your help and suggestions during the whole process. A special thanks to Henrique and Matheus who were always at the lab and brought a lot of joy. Finally, I would like to thank Professor Maciel for helping me start my research path, from when I was an undergraduate student until now. Your help was essential for me to reach this point.

FUNDING

This thesis received funding from São Paulo Research Foundation (FAPESP), grant #2018/23139-8. The opinions, hypotheses and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of FAPESP

"If you find a path with no obstacles,
it probably doesn't lead anywhere.
Frank A. Clark

RESUMO

RIBEIRO, R. R. M. **Método de Quantização para Redes Bayesianas e Aprendizagem Estrutural de Redes Bayesianas**. 2023. 161p. Tese (Doutorado) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2023.

Redes Bayesianas (BNs) são modelos versáteis para capturar relações complexas e são amplamente aplicados em diversos campos. Este estudo concentra-se em BNs com variáveis discretas. A qualidade do modelamento depende do volume adequado de dados, especialmente para construir tabelas de probabilidade condicional (CPTs). A quantidade de dados necessários varia com o Grafo Direcionado Acíclico (DAG) escolhido para a BN.

A aprendizagem estrutural da BN envolve um problema NP-difícil com um espaço de busca DAG superexponencial. Esta tese propõe investigar a otimização multiobjetivo na aprendizagem estrutural de BN (BNSL) para equilibrar critérios conflitantes. A abordagem utiliza conjuntos de Pareto e Algoritmos Genéticos (GAs) multiobjetivo.

Para realizar a BNSL, desenvolveu-se um GA multiobjetivo adaptativo paralelo com ajuste automático de parâmetros, denominado Algoritmo Genético Adaptativo com Tamanho de População Variável (AGAVaPS). Esse algoritmo proposto é extensivamente testado em diversas aplicações e em BNSL, mostrando-se superior a HillClimbing e Tabu Search em algumas métricas utilizadas.

O estudo também explora o impacto da quantização de dados no espaço de busca de BNSL. Introduz ainda um método de quantização chamado Quantização Baseada em Limite de CPT (CLBQ) que equilibra qualidade do modelo, fidelidade aos dados e pontuação da estrutura. A eficácia desse método é testada, demonstrando sua capacidade de ser usado na BNSL baseada em busca e pontuação. CLBQ obtém bons resultados, escolhendo quantizações com um bom erro médio quadrático e modelando bem as distribuições das variáveis. Além disso, CLBQ é adequado para uso em BNSL.

Palavras-chave: Redes Bayesianas. Aprendizagem estrutural. Quantização. Algoritmos evolutivos.

ABSTRACT

RIBEIRO, R. R. M. **Bayesian Network Quantization Method and Structural Learning**. 2023. 161p. Thesis (Doctor) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2023.

Bayesian Networks (BNs) are versatile models for capturing complex relationships, widely applied in diverse fields. This study focuses on discrete variable BNs. Modeling quality depends on adequate data volume, especially for constructing conditional probability tables (CPTs). The quantity of required data varies with the chosen BN Directed Acyclic Graph (DAG).

Structural learning of the BN involves an NP-hard problem with a super-exponential DAG search space. This thesis proposes investigating multi-objective optimization in BN structural learning (BNSL) to balance conflicting criteria. The approach utilizes Pareto sets and multi-objective Genetic Algorithms (GAs).

To perform BNSL, a parallel GA with automatic parameter adjustment is developed, called Adaptive Genetic Algorithm with Varying Population Size (AGAVaPS). This proposed algorithm is thoroughly tested on different applications and BNSL. AGAVaPS is found to be a good algorithm to be used in BNSL, performing better than HillClimbing and Tabu Search for some of the metrics measured.

The study also explores the impact of data quantization on the BNSL search space. It also introduces a quantization method called CPT Limit-Based Quantization (CLBQ) that balances model quality, data fidelity, and structure score. The effectiveness of this method is tested and its capability of being used in search and score BNSL is investigated. CLBQ is found to be a good quantization algorithm, choosing quantization that has a good mean squared error and modeling well the variables' distributions. Also, CLBQ is suitable to be used on BNSL.

Keywords: Bayesian network. Structural learning. Quantization. Evolutionary algorithm.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ABC	Artificial Bee Colony
AGAVaPS	Adaptive Genetic Algorithm with Varying Population Size
AIC	Akaike Information Criterion
aSI	Averaged Silhouette Index
BD	Bayesian Dirichlet
BDeu	Bayesian Dirichlet Equivalent Uniform
BIC	Bayesian Information Criterion
blip	Bayesian network Learning Improved Project
BN	Bayesian Network
BNSL	Bayesian Network Structural Learning
BSF	Balanced Scoring Function
CB	Constraint-based Methods
CI	Conditional Independence
CLBQ	CPT Limit-Based Quantization
CPD	Conditional Probabilities Distribution
CPT	Conditional Probabilities Table
DD	Dynamic Discretization
EA	Evolutionary Algorithm
ECD	Equal Cases Discretization
EM	Expectation Maximization
GAVaPS	Genetic Algorithm with Varying Population Size
GA	Genetic Algorithm
HC	Hill Climbing
MDL	Minimum Description Length

MDS	Multi-Dimensional Scaling
MLE	Maximum Likelihood Estimation
MOEA	Multi-objective Evolutionary Algorithm
MSE	Mean Squared Error
nAMGAVaPS	Negative Assortative Mating Genetic Algorithm with Varying Population Size
niGAVaPS	Non-incest Genetic Algorithm with Varying Population Size
PP	Prediction Power
PSO	Particle Swarm Optimization
SAD	Structure Aware Discretization
SB	Score-based Methods
SHD	Structural Hamming Distance
STD	Standard Deviation
SUD	Structure-unaware Stage Discretization
TABU	Tabu Search

CONTENTS

1 INTRODUCTION

Bayesian Network (BN) is a probabilistic graphical model capable of modeling complex and nonlinear relationships (FANG *et al.*, 2023; JACKSON-BLAKE *et al.*, 2022). BNs are used for modeling and reasoning and have been applied in various fields, such as assisted reproductive technology (TIAN *et al.*, 2023), network traffic prediction (SHIOMOTO; OTOSHI; MURATA, 2023), risk assessment (XU *et al.*, 2023) and water quality (BERTONE; ROUSSO; KUFEJI, 2023). The BN model is composed of two parts, a directed acyclic graph (DAG) which indicates probabilistic dependencies between variables represented by nodes, and the associated parameters that capture these probabilistic dependencies (HAO *et al.*, 2021; LUO; ZHAO; DU, 2019; KOLLER; FRIEDMAN, 2009). When working with continuous variables the associated parameters are conditional probability distributions (CPDs) and when working with discrete variables the associated parameters are conditional probability tables (CPTs) (KOLLER; FRIEDMAN, 2009).

There are several challenges on the application of BNs that can impact their effectiveness. These challenges can be broadly categorized into issues related to model development, data requirements, and computational demands.

- Model Development Challenges
  1. Model Structure: Defining the structure of a BN, including its nodes and edges, requires a deep understanding of the domain and the relationships between variables. The BN structural learning (BNSL), meaning learning the edges of a BN given a dataset, is very difficult (NP-hard problem) (VASIMUDDIN; CHOCKALINGAM; ALURU, 2018) because the DAGs search space increases super-exponentially as the number of variables increases (GROSS *et al.*, 2019; ROBINSON, 1977).
  2. Discretization of Continuous Variables: BNs often require continuous variables to be discretized into a discrete set of states, which can lead to a loss of information and potentially affect the computational efficiency, accuracy, and interpretability of the BN model (CHEN; WHEELER; KOCHENDERFER, 2015).
- Data-Related Challenges
  1. Data Requirements: Accurate modeling with BNs requires large amounts of data to learn the conditional probability distributions of the network (WILSON *et al.*, 2022; JACKSON-BLAKE *et al.*, 2022). The lack of data can lead to missing or ill-informed probabilities on the CPTs. This deteriorates the model quality

and performance (RU *et al.*, 2023; MAYFIELD *et al.*, 2017). The quantity of data needed to model well a system is dependent on the BN DAG used because the CPT dimensions are tied to the quantization of the variable and its parents (ROHMER, 2020).

- Computational Challenges

    1. Computational Demands: The computational effort associated to BNs grows exponentially with the number of states of its variables (ZWIRGLMAIER; STRAUB, 2016). This can make the use of BNs impractical for very large or complex networks.

In summary, while BNs offer a powerful framework for probabilistic modeling and decision-making under uncertainty, addressing these challenges requires advancements in methodologies, better data management strategies, and efforts to simplify the use and interpretation of BNs for end-users. This thesis will approach the problem of the BNSL, including the discretization of variables and data requirements that affect it.

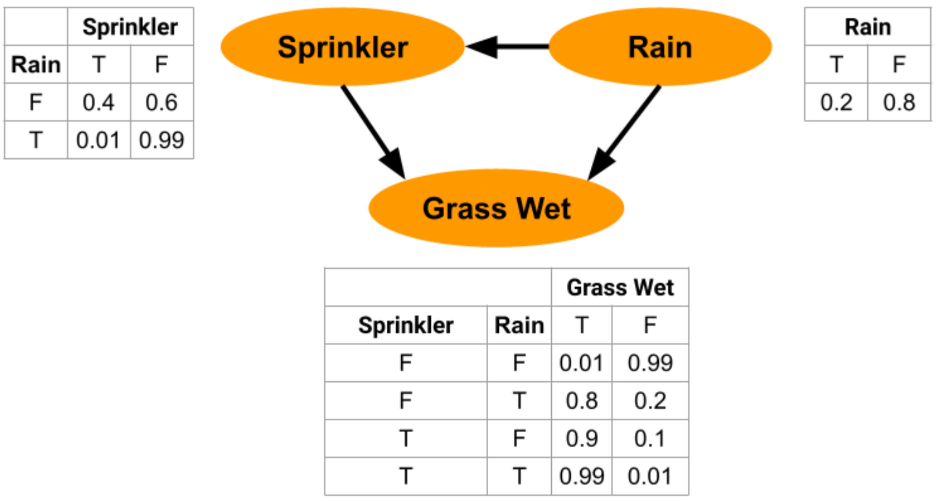
1.1 General Concepts and Initial Definitions

Bayesian Networks (BNs) are powerful tools for modeling uncertainty and making predictions across various fields, including reproductive technology (TIAN *et al.*, 2023), network traffic prediction (SHIOMOTO; OTOSHI; MURATA, 2023), risk assessment (XU *et al.*, 2023) and water quality (BERTONE; ROUSSO; KUFEJI, 2023).

The BN model is composed of two parts: a directed acyclic graph (DAG), which indicates probabilistic dependencies between variables represented by nodes, and the associated parameters that capture these probabilistic dependencies (HAO *et al.*, 2021; LUO; ZHAO; DU, 2019; KOLLER; FRIEDMAN, 2009). When working with continuous variables, the associated parameters are conditional probability distributions (CPDs), and when working with discrete variables, the associated parameters are conditional probability tables (CPTs) (KOLLER; FRIEDMAN, 2009).

Examples of BNs with discrete variables can be seen in Figure 1. On it, the variables are indicated by the nodes with their names, the edges indicate the dependencies between the variables, and the tables show the CPTs. Variables without parents have a CPT with only one row stating their possible states and probabilities. Nodes with parents have CPTs with one row for each parent's state combination. So in Figure 1a, "Sprinkler" has a CPT with two rows, as its parent "Rain" only has two possible combinations. Meanwhile, "Grass Wet" has four rows since there are four possible combinations of its parent states. The number of states of the variable gives the number of columns.

Figure 1 – Examples of BNs with discrete variables, showing the DAG (structure) and the CPTs. Variables are indicated by the nodes with their names, the dependencies between the variables are indicated by the edges and the CPTs are shown by the tables.

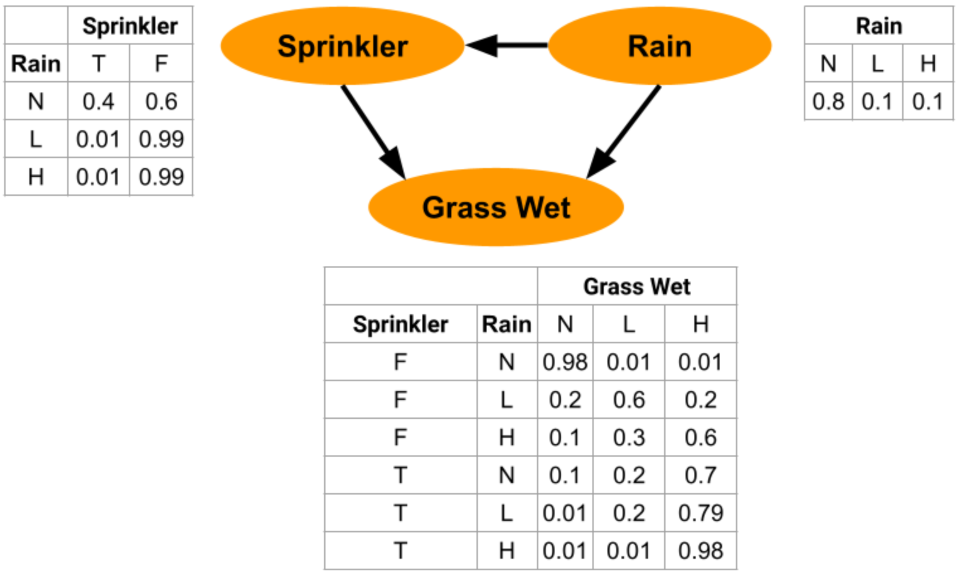


	Sprinkler	
Rain	T	F
F	0.4	0.6
T	0.01	0.99

Rain	
T	F
0.2	0.8

		Grass Wet	
Sprinkler	Rain	T	F
F	F	0.01	0.99
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

(a) With 3 variables with 2 states each. T stands for true and F for false.



	Sprinkler	
Rain	T	F
N	0.4	0.6
L	0.01	0.99
H	0.01	0.99

Rain		
N	L	H
0.8	0.1	0.1

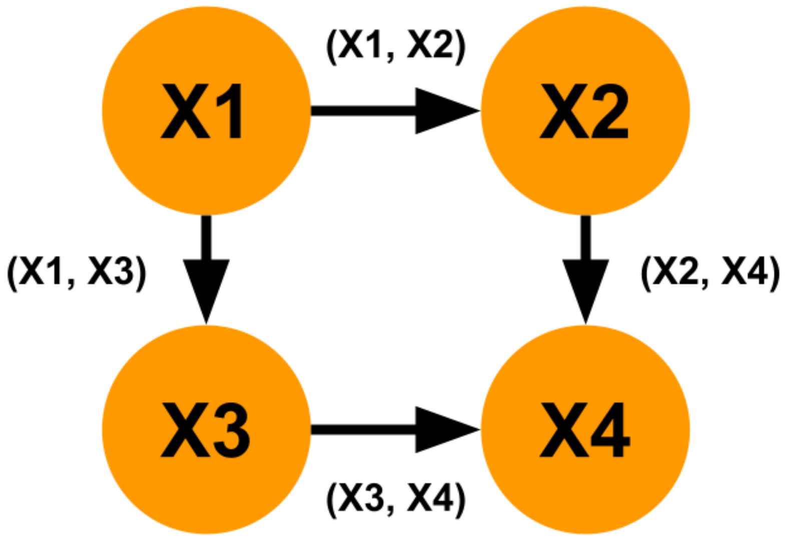
		Grass Wet		
Sprinkler	Rain	N	L	H
F	N	0.98	0.01	0.01
F	L	0.2	0.6	0.2
F	H	0.1	0.3	0.6
T	N	0.1	0.2	0.7
T	L	0.01	0.2	0.79
T	H	0.01	0.01	0.98

(b) With 2 variables with 3 states and 1 variable with 2 states. N stands for no, L for low, H for high, T for true and F for false.

Source: Adapted from Lokrantz, Gustavsson and Jirstrand (2018)

To understand how BNs work, it's important to review some graph theory. A directed graph is represented by a pair (V, E), where V is a finite set of elements called nodes and E is a set of edges (ordered pairs of different elements of V), such that if $(X, Y) \in E$, it is said that there is an edge from X to Y and that they are incident to the edge. Nodes that are connected by an edge are adjacent to each other. Considering a set of nodes $[X_1, X_2, \ldots, X_k]$, with $k \geq 2$, and $(X_{i-1}, X_i) \in E$ for $2 \leq i \leq k$, then the set of edges connecting the k nodes is called a path from X_1 to X_k. The interior nodes of this path are, $X_2, \ldots, X_{k-1}$ and a subpath from X_i to X_j is the path $[X_i, X_{i+1}, \ldots, X_j]$ where $1 \leq i < j \leq k$. A directed acyclic graph (DAG) is a directed graph $\mathbb{G}$, which does not have a path from a node to itself (directed cycle). Given a DAG $\mathbb{G} = (V, E)$ and nodes X and Y in V, Y is a parent of X if $(Y, X) \in E$ and X is a decedent of Y. An example of this DAG notation can be seen in Figure 2, on it the variables and edges are indicated and the notation used is exemplified. In this example DAG, X_2 and X_3 are parents of X_4, and X_3 is a decent of X_1.

Figure 2 – Example of a directed acyclic graph and its mathematical notation. The DAG $\mathbb{G} = (V, E)$ is composed of a set of nodes (variables) $V = [X_1, X_2, X_3, X_4]$ and a set of edges $E = [(X_1, X_2), (X_1, X_3), (X_2, X_4), (X_3, X_4)]$. The nodes and the edges are shown and identified in the figure. An example of a path from X_1 to X_4 is $[X_1, X_2, X_4]$.



Source: Author

Being P the joint probability distribution of the random variables in a set V and a DAG $\mathbb{G} = (V, E)$, then P equals the product of its conditional probability distributions of all nodes given values of their parents in $\mathbb{G}$ (eq. 1.1), when these conditional distributions exist and $(\mathbb{G}, P)$ satisfies the Markov condition. The Markov condition states that every node in a BN is conditionally independent of its non-descendants, given its parents. A BN is called a $(\mathbb{G}, P)$ that satisfies the Markov condition. When using discrete variables, the Markov condition is always satisfied.

When $(\mathbb{G}, P)$ satisfies the Markov condition, P is equal to the product of its conditional distributions of all nodes given values of their parents when these conditional distributions exist. This factorization allows its joint probability density function to be written as a product of the individual density functions, conditional on their parent variables (KOLLER; FRIEDMAN, 2009):

$$P(x) = \prod_{X_i \in V} P\left(x_{X_i} \,\middle|\, x_{\mathrm{Pa}_{\mathrm{G}}(X_i)}\right) \tag{1.1}$$

where $Pa_G(X_i)$ is the set of parents of X_i. For the case of Figure 1a, an example of calculation of the joint probability distribution would be

$$P(GrassWet, Sprinkler, Rain) = P(GrassWet|Sprinkler, Rain)P(Sprinkler|Rain)P(Rain) \tag{1.2}$$

that means that the probability of the case where $GrassWet = T$, $Sprinkler = F$ and $Rain = T$ is given by

$$\begin{aligned} P(GrassWet = T, Sprinkler = F, Rain = T) =& P(GrassWet = T|Sprinkler = F, Rain = T) \\ & P(Sprinkler = F|Rain = T)P(Rain = T) \\ =& (0.8)(0.99)(0.2) = 0.1584 \end{aligned} \tag{1.3}$$

Additionally, it is possible to check the probability of having a specific value in only some of the variables, such as

$$\begin{aligned} P(Sprinkler = T) =& P(Sprinkler = T|Rain = T)P(Rain = T) \\ & + P(Sprinkler = T|Rain = F)P(Rain = F) \\ =& (0.01)(0.2) + (0.4)(0.8) = 0.002 + 0.32 = 0.322 \end{aligned} \tag{1.4}$$

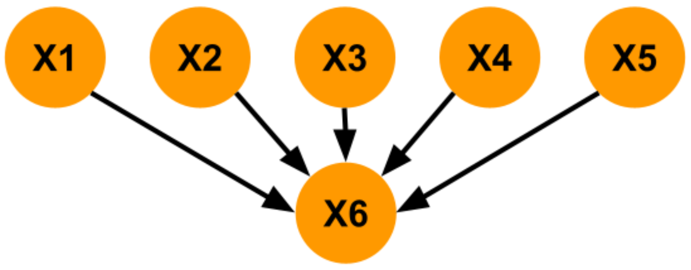
An important thing to guarantee quality of prediction is having good statistic estimates on the CPTs, that truly model the variables' distributions. A good amount of data is needed to achieve this, as the most used estimator used to construct the CPTs from data is a frequentist approach (WILSON *et al.*, 2022; JACKSON-BLAKE *et al.*, 2022). The lack of data can lead to missing or ill-informed probabilities on the CPTs.

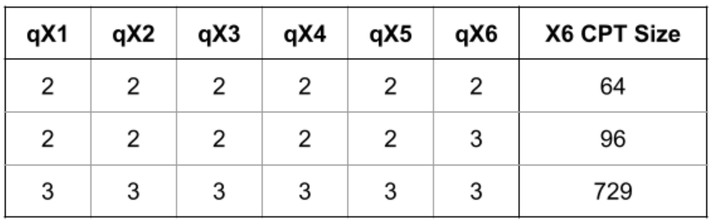
This deteriorates the model quality and performance (RU *et al.*, 2023; MAYFIELD *et al.*, 2017). As it can be seen in Figure 1, the quantity of data needed to model a system well is dependent on the BN DAG used because the CPT dimensions are tied to the quantization of the variable and its parents (ROHMER, 2020). The size of a variables' CPT ($\mathcal{L}$) given a DAG $\mathbb{G} = (V, E)$, a variable $X_i \in V$ and a set of k parents of this variable $Pa(X_i) = [P_1, \ldots, P_k]$ can be described by the following equation

$$\mathcal{L} = q_{X_i} \prod_{i=1}^{k} q_{P_i} \tag{1.5}$$

where q_X represents the quantization of the variables X with $X \in V$ (ROHMER, 2020). In this case, the quantization refers to the number of states a variable can assume. In the examples of Figure 1, it can be seen that the CPT size for “Grass Wet” changes from $\mathcal{L} = 2 \cdot 2 \cdot 2 = 8$ to $\mathcal{L} = 3 \cdot 2 \cdot 3 = 18$ based on the increase of quantization of variables “Grass Wet” and “Rain” (meaning the increase of the number of the states that these variables can assume). Thus, it can be seen that the data volume needed, the quantization used for each variable and the DAG are intertwined. An additional example of this can be seen in Figure 3, where a BN structure is shown and the CPT size for the variable X_6 is calculated for different quantization values of each variable. For this type of structure, where one variable has many parents, a small increase in one variable's quantization can have a big impact on the CPT size. Additionally, a small increase in the quantization of all variables greatly increases the CPT size.

Figure 3 – Example of BN with the CPT size for the variable X_6 considering different quantization of the variables (number of states each variable can assume). The CPT was calculated using Equation 1.5.





qX1	qX2	qX3	qX4	qX5	qX6	X6 CPT Size
2	2	2	2	2	2	64
2	2	2	2	2	3	96
3	3	3	3	3	3	729

Source: Author

In addition to the issue of modelling quality and CPT size, the choice of quantization of the variables affects the computational efficiency, accuracy, and interpretability of the resulting BN model (CHEN; WHEELER; KOCHENDERFER, 2015). The problem of data quantization causes this.

When discretizing data, the first step is to sample the continuous signal ($x_c(t)$). This can be mathematically represented by

$$x_s(t) = \sum_{n=-\infty}^{\infty} x_c(nT)\delta(t - nT) \tag{1.6}$$

where $\delta(t)$ is the unit impulse, T is the sample period and n is the sample index. The sampled data can also be represented by $x[n]$. After the sampling, the samples are quantized, which means they are rounded to the nearest quantisation value. This process is represented by

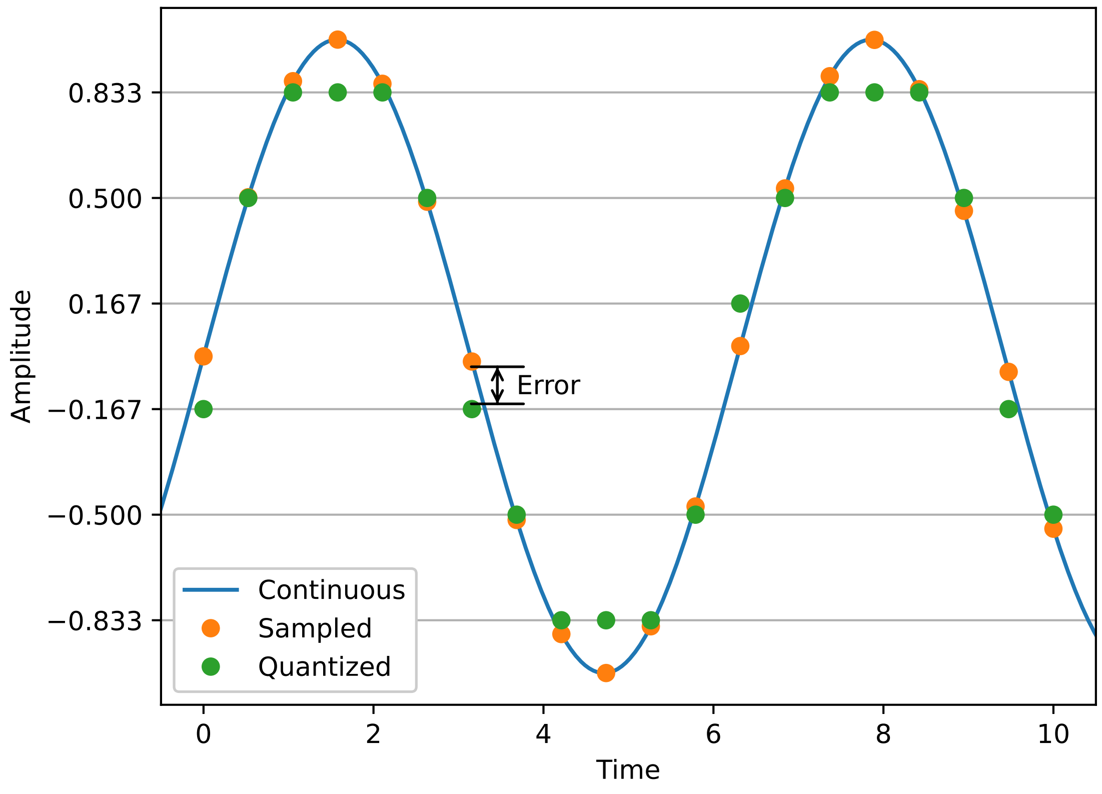
$$\hat{x}[n] = Q(x[n]) \tag{1.7}$$

where Q is the quantization function that reduces the precision of the samples to a fixed number of states. The difference between the real sample $x[n]$ and the quantized sample $\hat{x}[n]$ is called the quantization error, and is defined as

$$e[n] = \hat{x}[n] - x[n] \tag{1.8}$$

An example of the continuous, sampled and quantized signals can be seen in Figure 4. On it, a continuous signal, a sampled signal and a quantized signal are shown. Also, the graphical meaning of the quantization error is indicated for one of the samples.

Figure 4 – Example of a continuous signal, a sampled signal from it and a quantized signal. The error value is also indicated for one of the sample points.



Source: Author

An issue with the sampling process is that the sampling frequency must obey the Nyquist condition not to incur a misrepresentation of the signal and aliasing. The Nyquist condition is

$$\Omega_S > \Omega_N \tag{1.9}$$

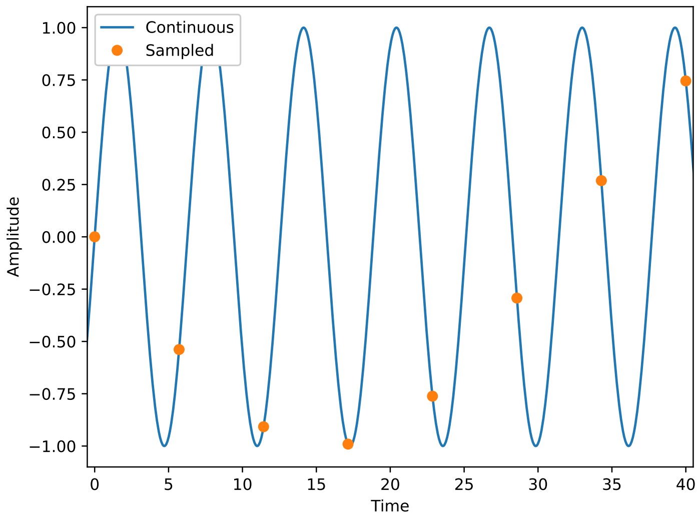
where Ω_S is the sampling frequency, Ω_N is the maximum frequency of a signal $x_c(t)$ with limited bandwidth, such that

$$X_c(j\Omega) = 0 \text{ for } |\Omega| > \Omega_N \tag{1.10}$$

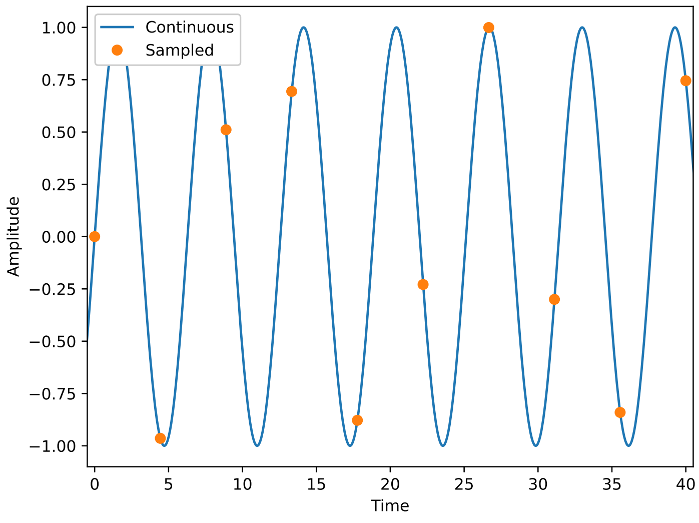
where $X_c(j\Omega)$ is the Fourier transformation of $x_c(t)$, thus being a frequency analysis of $x_c(t)$.

The problem of not respecting the Nyquist condition for sampling is exemplified in Figure 5. On it two different sample frequencies that do not respect the Nyquist condition are used to sample a sine function. In 5a the resulting sampled signal is a sine function of a different frequency. Meanwhile, in 5b the resulting sampled signal does not reflect the original signal at all.

Figure 5 – Example of the problem of not following the Nyquist condition for sampling. In both plots, the sampling frequency is under the value defined by the Nyquist condition. In (a) this results in a sine signal of different frequency, meanwhile in (b) the original signal is not recognizable considering its sample.



(a)



(b)

Source: Author

Alternatively, the quantization process can be seen as an addition of noise when the equation above is manipulated

$$\hat{x}[n] = x[n] + e[n] \tag{1.11}$$

this shows that when the variable is quantized in a few states (small quantization value), it would be equivalent to adding a high noise value to your original data. This, in return, can result in poor data modeling and the original data not being recognizable (OPPENHEIM; SCHAFER; BUCK, 1999) and presenting different dependencies than originally. An example of this alteration of dependencies resulting from quantization error is shown after the BIC score is presented further down in the text.

With the quantization influence, the issue of learning a BN from data can be further approached. Learning BN from data can be divided into two steps: learning the structure (DAG) and learning the parameters (CPTs). The BN structural learning (BNSL) is a very challenging problem (NP-hard) (VASIMUDDIN; CHOCKALINGAM; ALURU, 2018) because the structure's search space increases super-exponentially as the number of variables increases (GROSS *et al.*, 2019; ROBINSON, 1977). An example of this super-exponential increase of the search space as the number of variables increases can be seen in Table 1; these numbers were calculated using the following recurrence

Table 1 – Number of DAGs (structures) for a given number of variables.

Number of variables	**Number of DAGs**
3	2,50e+**01**
4	5,43e+**02**
5	2,93e+**04**
6	3,78e+**06**
7	1,14e+**09**
8	7,84e+**11**
9	1,21e+**15**
10	4,18e+**18**
11	3,16e+**22**
12	5,22e+**26**
13	1,87e+**31**
14	1,44e+**36**
15	2,38e+**41**
16	8,38e+**46**
17	6,27e+**52**
18	9,94e+**58**
19	3,33e+**65**
20	2,34e+**72**
21	3,47e+**79**
22	1,08e+**87**
23	6,97e+**94**
24	9,44e+**102**
25	2,66e+**111**

Source: Author

$$H(n) = \sum_{k=1}^{n} (-1)^{k+1} \binom{n}{k} 2^{k(n-k)} H(n-k) \tag{1.12}$$

where $H(n)$ is the number of DAGs for a n number of variables and $\binom{n}{k}$ is the binomial coefficient defined as

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \tag{1.13}$$

As can be seen, the search space can get so huge that even searching one million DAGs would not be a significant search.

There are three main approaches to performing BNSL: constraint-based methods (CB), score-based methods (SB), and hybrid methods that combine the two previous approaches (CONSTANTINOU *et al.*, 2021; CONTALDI; VAFAEE; NELSON, 2019; NEAPOLITAN, 2003). CB methods apply conditional independence (CI) tests to the data to determine the BN structure. Meanwhile, SB methods, also called search and score methods, use structure scores and search algorithms to search through the DAGs search space while evaluating the fitness of the DAGs to the data (CONTALDI; VAFAEE; NELSON, 2019; NEAPOLITAN, 2003).

For SB methods that are scoring functions based on different principles, such as entropy and information, minimum description length, or Bayesian approaches. The scoring problem can be expressed by

$$G^* = \arg\max_{G \in \mathcal{G}_n} g(G : D) \tag{1.14}$$

where $D = [u^1, \ldots, u^N]$ is a complete training data set of instances of U_n, G^* is the best DAG considering the scoring function $g(G : D)$, $g(G : D)$ measures the degree of fitness of a DAG G to the dataset D and $\mathcal{G}_n$ is the set of all DAGs defined on U_n.

Some scores are decomposable, which means their value for a structure can be expressed as a sum of values from each node and its parents in the logarithmic space:

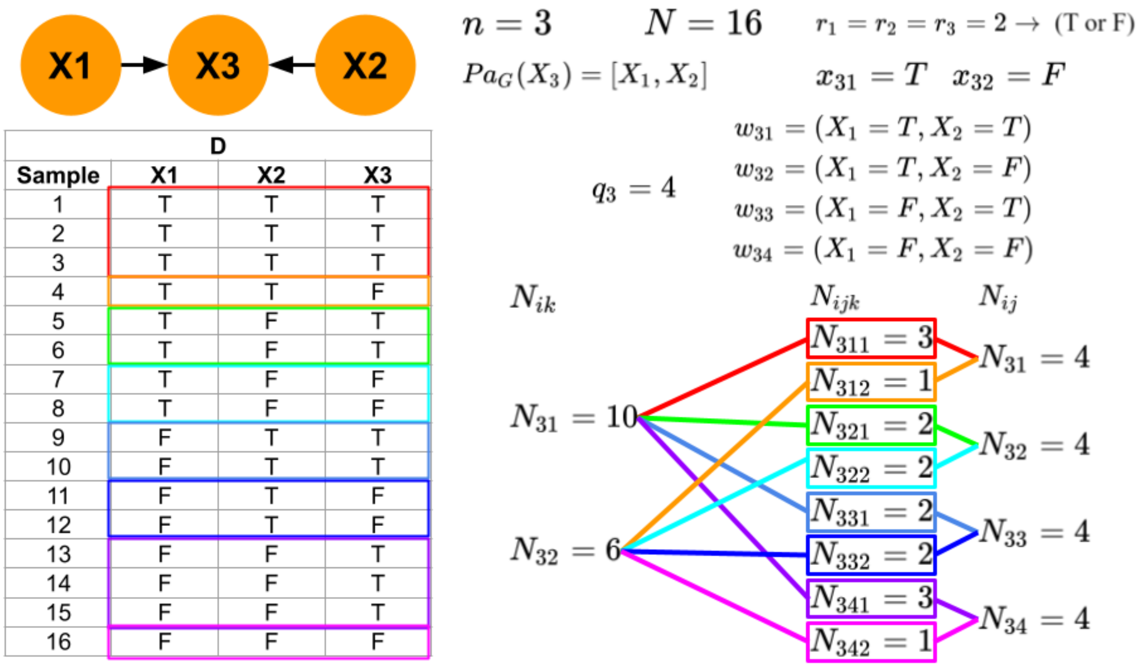
$$\begin{aligned} g(G : D) &= \sum_{X_i \in U_n} g(X_i, Pa_G(X_i) : D) \\ g(X_i, Pa_G(X_i) : D) &= g(X_i, Pa_G(X_i) : N^D_{X_i, Pa_G(X_i)}) \end{aligned} \tag{1.15}$$

where $N^D_{X_i, Pa_G(X_i)}$ are the sufficient statistic[1] of the set of variables $[X_i] \cup Pa_G(X_i)$ in D, meaning the number of instances in D corresponding to each possible configuration of $[X_i] \cup Pa_G(X_i)$. A scoring function can be score-equivalent by assigning the same value to all DAGs representing the same essential graph.

The notation used for the following discussion and formulations on scoring functions is as follows: n is the number of variables; r_i is the number of states of the variable X_i; q_i is the number of possible configurations of the parent set $Pa_G(X_i)$ of X_i, where $q_i = \prod_{X_j \in Pa_G(X_i)} r_j$; w_{ij}, $j = 1, \ldots, q_i$ represents a configuration of $Pa_G(X_i)$; N_{ijk} is the number of instances in the data set D where the variables X_i take the value x_{ik} and the set of variables $Pa_G(X_i)$ take the value w_{ij}; N_{ij} is the number of instances in the data set where the variables in $Pa_G(X_i)$ take their configuration w_{ij}, where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$; N_{ik} is the number of instances in the data set where the variable takes its value x_{ik}, where $N_{ik} = \sum_{j=1}^{q_i} N_{ijk}$; N is the total number of instances in D. An example of all these values can be seen in Figure 6.

[1] In statistics, a statistic is a function of a set of data. A sufficient statistic contains all the information about the statistical model that the original data does. This means that adding additional information will not improve the model.

Figure 6 – Example of the BN notation used on the BN structural scores formulation. The variable used for the example was X_3. n is the number of variables; r_i is the number of states of the variable X_i; q_i is the number of possible configurations of the parent set $Pa_G(X_i)$ of X_i, where $q_i = \prod_{X_j \in Pa_G(X_i)} r_j$; w_{ij}, $j = 1, \ldots, q_i$ represents a configuration of $Pa_G(X_i)$; N_{ijk} is the number of instances in the data set D where the variables X_i take the value x_{ik} and the set of variables $Pa_G(X_i)$ take the value w_{ij}; N_{ij} is the number of instances in the data set where the variables in $Pa_G(X_i)$ take their configuration w_{ij}, where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$; N_{ik} is the number of instances in the data set where the variable takes its value x_{ik}, where $N_{ik} = \sum_{j=1}^{q_i} N_{ijk}$; N is the total number of instances in D.



D

Sample	X1	X2	X3
1	T	T	T
2	T	T	T
3	T	T	T
4	T	T	F
5	T	F	T
6	T	F	T
7	T	F	F
8	T	F	F
9	F	T	T
10	F	T	T
11	F	T	F
12	F	T	F
13	F	F	T
14	F	F	T
15	F	F	T
16	F	F	F

Source: Author

For Bayesian scoring functions the best network is the one that maximizes the posterior probability $p(G|D)$, since $p(D)$ is the same for all structures it is only needed to compute $p(G, D)$. One such score is the score K2 (COOPER; HERSKOVITS, 1992) which uses the assumptions of lack of missing values, multinomiality, parameter modularity, parameter independence and uniformity of the prior distribution of the parameters given the DAG. K2 is expressed by:

$$g_{K2}(G : D) = \log(p(G)) + \sum_{i=1}^{n} \left[\sum_{j=1}^{q_i} \left[\log\left(\frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \right) + \sum_{k=1}^{r_i} \log(N_{ijk}!) \right] \right] \tag{1.16}$$

where $p(G)$ is the prior probability of the DAG G; n is the number of variables; r_i is the number of states of the variable X_i; q_i is the number of possible configurations of the parent set $Pa_G(X_i)$ of X_i, where $q_i = \prod_{X_j \in Pa_G(X_i)} r_j$; w_{ij}, $j = 1, \ldots, q_i$ represents a configuration of $Pa_G(X_i)$; N_{ijk} is the number of instances in the data set D where the variables X_i

takes the value x_{ik} and the set of variables $Pa_G(X_i)$ take the value w_{ij}; N_{ij} is the number of instances in the data set where the variables in $Pa_G(X_i)$ take their configuration w_{ij}, where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$; N_{ik} is the number of instances in the data set where the variable takes its value x_{ik}, where $N_{ik} = \sum_{j=1}^{q_i} N_{ijk}$; N is the total number of instances in D. An example of these values can be seen in Figure 6.

The Bayesian Dirichlet (BD) score (HECKERMAN; GEIGER; CHICKERING, 1995) was proposed as a generalization of K2 defined by the equation

$$g_{BD}(G : D) = \log(p(G)) + \sum_{i=1}^{n}\left[\sum_{j=1}^{q_i}\left[\log\left(\frac{\Gamma(\eta_{ij})}{\Gamma(N_{ij}+\eta_{ij})}\right) + \sum_{k=1}^{r_i}\log\left(\frac{\Gamma(N_{ijk}+\eta_{ijk})}{\Gamma(\eta_{ijk})}\right)\right]\right] \tag{1.17}$$

where $\Gamma(\cdot)$ is the function Gamma

$$\Gamma(c) = \int_{0}^{\infty} e^{-u}u^{c-1}du \tag{1.18}$$

and η_{ijk} are the hyper-parameters for the Dirichlet prior distributions of the parameters given a network structure. It must be noted that if all hyper-parameters are one ($\eta_{ijk} = 1$) the score becomes the score K2. Also, n is the number of variables; r_i is the number of states of the variable X_i; q_i is the number of possible configurations of the parent set $Pa_G(X_i)$ of X_i, where $q_i = \prod_{X_j \in Pa_G(X_i)} r_j$; w_{ij}, $j = 1, \ldots, q_i$ represents a configuration of $Pa_G(X_i)$; N_{ijk} is the number of instances in the data set D where the variables X_i takes the value x_{ik} and the set of variables $Pa_G(X_i)$ take the value w_{ij}; N_{ij} is the number of instances in the data set where the variables in $Pa_G(X_i)$ take their configuration w_{ij}, where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$; N_{ik} is the number of instances in the data set where the variable takes its value x_{ik}, where $N_{ik} = \sum_{j=1}^{q_i} N_{ijk}$; N is the total number of instances in D.

Determining the hyper-parameters η_{ijk} is a difficult task, however if the assumption of likelihood equivalence is used the hyper-parameters can be determined by

$$\eta_{ijk} = \eta \times p(x_{ik}, w_{ij}|G_0) \tag{1.19}$$

where $p(\cdot|G_0)$ is the probability distribution associated with a prior BN G_0 and η is a parameter symbolizing the equivalent sample size. When the prior network assigns a uniform probability to each configuration of $[X_i] \cup Pa_G(X_i)$, $p(x_{ik}, w_{ij}|G_0) = \frac{1}{r_i q_i}$, the BDeu score is defined, expressed by the equation

$$g_{BDeu}(G : D) = \log(p(G)) + \sum_{i=1}^{n}\left[\sum_{j=1}^{q_i}\left[\log\left(\frac{\Gamma(\frac{\eta}{q_i})}{\Gamma(N_{ij}+\frac{\eta}{q_i})}\right) + \sum_{k=1}^{r_i}\log\left(\frac{\Gamma(N_{ijk}+\frac{\eta}{r_i q_i})}{\Gamma(\frac{\eta}{r_i q_i})}\right)\right]\right] \tag{1.20}$$

where $\Gamma(\cdot)$ is the function Gamma (Equation 1.18); η is the equivalent sample size and a parameter of the BDeu score; n is the number of variables; r_i is the number of states of the variable X_i; q_i is the number of possible configurations of the parent set $Pa_G(X_i)$ of X_i, where $q_i = \prod_{X_j \in Pa_G(X_i)} r_j$; w_{ij}, $j = 1, \ldots, q_i$ represents a configuration of $Pa_G(X_i)$; N_{ijk} is the number of instances in the data set D where the variables X_i takes the value x_{ik} and the set of variables $Pa_G(X_i)$ take the value w_{ij}; N_{ij} is the number of instances in the data set where the variables in $Pa_G(X_i)$ take their configuration w_{ij}, where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$; N_{ik} is the number of instances in the data set where the variable takes its value x_{ik}, where $N_{ik} = \sum_{j=1}^{q_i} N_{ijk}$; N is the total number of instances in D. For both scores, $p(G)$ is normally assumed to be a uniform distribution, thus it becomes a constant and can be removed.

On the scores based on information theory, there are scores based on the minimum description length (MDL) principle and Schwarz information criterion (SCHWARZ, 1978). These scores try to minimize the sum of the description length of the model and the description length of the data given the model. The description length of a BN, also called network complexity, is given by

$$C(G) = \sum_{i=1}^{n} (r_i - 1) q_i \tag{1.21}$$

and the description of the data given the model is calculated using the negative of the log-likelihood, calculated as follows

$$LL_D = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \left(\frac{N_{ijk}}{N_{ij}} \right) \tag{1.22}$$

where n is the number of variables; r_i is the number of states of the variable X_i; q_i is the number of possible configurations of the parent set $Pa_G(X_i)$ of X_i, where $q_i = \prod_{X_j \in Pa_G(X_i)} r_j$; w_{ij}, $j = 1, \ldots, q_i$ represents a configuration of $Pa_G(X_i)$; N_{ijk} is the number of instances in the data set D where the variables X_i takes the value x_{ik} and the set of variables $Pa_G(X_i)$ take the value w_{ij}; N_{ij} is the number of instances in the data set where the variables in $Pa_G(X_i)$ take their configuration w_{ij}, where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$; N_{ik} is the number of instances in the data set where the variable takes its value x_{ik}, where $N_{ik} = \sum_{j=1}^{q_i} N_{ijk}$; N is the total number of instances in D.

The scores based on MDL are then defined as $g(G : D) = LL_D - C(G) f(N)$, where $f(N)$ is a non-negative penalization function. Two main scores use this format: the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). AIC uses $f(N) = 1$ and BIC uses $f(N) = \frac{1}{2} \log(N)$, their complete formulation, are shown bellow (CAMPOS, 2006). It is important to note that the BIC score is based on the Schwarz information criterion, which coincides with the MDL score. Another interesting thing to note is that for the likelihood LL_D the best structure is always a complete graph including all possible arcs, that is why the penalization function is very important for these scores.

$$g_{AIC}(G:D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\log\left(\frac{N_{ijk}}{N_{ij}}\right) - \sum_{i=1}^{n}(r_i-1)q_i \tag{1.23}$$

$$g_{BIC}(G:D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\log\left(\frac{N_{ijk}}{N_{ij}}\right) - \frac{1}{2}\log(N)\sum_{i=1}^{n}(r_i-1)q_i \tag{1.24}$$

On Equations 1.23 and 1.24, n is the number of variables; r_i is the number of states of the variable X_i; q_i is the number of possible configurations of the parent set $Pa_G(X_i)$ of X_i, where $q_i = \prod_{X_j \in Pa_G(X_i)} r_j$; w_{ij}, $j = 1,\ldots,q_i$ represents a configuration of $Pa_G(X_i)$; N_{ijk} is the number of instances in the data set D where the variables X_i takes the value x_{ik} and the set of variables $Pa_G(X_i)$ take the value w_{ij}; N_{ij} is the number of instances in the data set where the variables in $Pa_G(X_i)$ take their configuration w_{ij}, where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$; N_{ik} is the number of instances in the data set where the variable takes its value x_{ik}, where $N_{ik} = \sum_{j=1}^{q_i} N_{ijk}$; N is the total number of instances in D.

Now that the BIC score has been presented and explained, the miss identification of a dependency resulting from the quantization error can be exemplified. Given a dataset D from two samples of normal distributions called ‘A’ and ‘B’. ‘A’ is a sample of a normal distribution with $\mu = 1$, $\sigma = 0.1$ and 1000 samples. ‘B’ is a sample of a normal distribution with $\mu = 2$, $\sigma = 0.2$ and 1000 samples. Two different values of quantization were used to calculate the BIC score of a BN with no edges (BN Empty) and a BN with an edge between ‘A’ and ‘B’ (BN edge). The quantization values and the score values can be seen in Table 2. From it, it can be seen that for a smaller quantization (smaller number of states that each variable can assume) the dependency between ‘A’ and ‘B’ is found, while for a higher quantization, the two variables are recognized as being independent.

Table 2 – Quantization (number of states that each variable can assume) and BIC score values for the miss identification of a dependency resulting from the quantization error test.

Quantization	**BIC Score BN Empty**	**BIC Score BN Edge**
‘A’: 100 and ‘B’: 100	**-8893.07**	-33134.35
‘A’: 2 and ‘B’: 2	-1380.23	**-1379.72**

Source: Author

Now approaching the process of learning CPT from data, consider that $\theta_{ijk} = P(X_i = k|Pa(X_i) = j)$ is the k-th probability value of the CPT, where $i = 1,\ldots,n$; $j = 1,\ldots,q_i$ and $k = 1,\ldots,r_i$. These probabilities can be evaluated from the frequencies of the data when there is enough data. One such method is the maximum likelihood estimation method (MLE).

On the MLE, given a dataset D with N_{ij} samples where $Pa(X_i)$ is in the state j and N_{ijk} sampled for X_i in the state k with $Pa(X_i)$ in j state. MLE tries to maximize the log-likelihood function $l(\cdot)$ of θ given a dataset D, which can be expressed as

$$l(\theta|D) = log(P(D|\theta)) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\log(\theta_{ijk}) \tag{1.25}$$

The estimator obtained this way is

$$\hat{\theta}_{ijk} = \frac{N_{ijk}}{N_{ij}} \tag{1.26}$$

This method fails to correctly estimate the probabilities when there is not enough data for some variables' state configurations ($N_{ij} \approx 0$). This data scarcity problem is worsened when the number of nodes increases, since as it was already seen, the number of conditional probabilities increases by multiples of the number of states of the parents of a variable (Equation 1.5).

The parameter learning process is complicated when there are missing values. Popular algorithms of parameter learning with missing values are Expectation-Maximization and Gibbs sampling, however, both depend on the assumption that the values are missing at random, which may not always be true in practice.

1.2 Proposal

Considering the challenge of BNSL and the academic interest in BNSL, this thesis proposes an investigation of the use of multi-objective optimization in BNSL. The use of multi-objective optimization is interesting because it enables the balance between criteria represented by scores that are possibly conflicting (LV *et al.*, 2019). When solving multi-objective problems, we seek solutions that possess a compromise between different criteria (FENG; YANG; HUANG, 2017), since it is hard or impossible to find a solution that satisfies all the objectives at the same time (CUI *et al.*, 2017).

The multi-objective optimization is done using the concept of a Pareto set, which is a set that contains all the solutions that are not dominated between themselves considering all the solutions found on the search (LI *et al.*, 2019). Given two solutions $X1$ and $X2$, $X1$ is said to dominate $X2$ when $X1$ has a performance at least equal to $X2$ for all the objectives, and $X1$ has a better performance than $X2$ for at least one of the objectives (LI *et al.*, 2019).

There are many ways of dealing with a multi-objective problem. One method used to solve a multi-objective problem is metaheuristic algorithms, more specifically multi-objective metaheuristic algorithms, because of their population-based nature metaheuristic algorithms are capable of creating a Pareto set that approximates the Pareto front (ZHOU *et al.*, 2011). Moreover, metaheuristic algorithms are also very used because of their capability of finding solutions in large and complex search spaces (DUDEK; JANIGA; WOJNAROWSKI, 2021). There are many bio-inspired metaheuristic algorithms, such as Artificial Bee Colony (ABC) (ZHOU *et al.*, 2021), Particle Swarm Optimization (PSO) (XIA *et al.*, 2020; KENNEDY; EBERHART, 1995), and Genetic Algorithm (GA) (HANH *et al.*, 2019). Despite having a wide variety of algorithms, GA is the most used to solve optimization problems occupying 56% of applications on papers between 1988 and 2017 (DUDEK; JANIGA; WOJNAROWSKI, 2021).

In problems that have huge search spaces, such as BNSL, metaheuristic algorithms can have a problem of performance deteriorating as the dimension of the search space increases (UMBARKAR; JOSHI; HONG, 2014). To deal with high complexity problems, metaheuristic algorithms must have robust mechanisms for population diversity preservation (WANG; SOBEY, 2020). This search space size can also result in a higher execution time. To optimize the execution time, parallel metaheuristic algorithms can be used (TSOULOS; TZALLAS; TSALIKAKIS, 2016). Naturally, parallelization generates a communication cost between the parallel processes, however, there are strategies that mitigate this cost (LUQUE; ALBA, 2011).

When using metaheuristic algorithms, another challenge is to choose appropriate parameters for the optimization, considering that the parameters' values directly influence the performance of the algorithms (MOBIN *et al.*, 2018). Despite that, many times the parameters are defined based on conventions, based on the literature, and limited experimental comparisons (SIPPER *et al.*, 2018).

1.3 Objetive

To address the challenge of BNSL, a parallel GA algorithm with automatic parameter adjustment was developed. In addition to that, the impact of data quantization on the BNSL search space was studied, and a quantization method that considers the BN structure was proposed. The usage of this quantization method in BNSL was evaluated.

1.4 Agenda

The remainder of this thesis is structured as follows. Chapter 2 describes related works and gives an overview of what has been done on the topics of BNSL, the concept of life in Genetic Algorithms, and quantization for BNs. Chapter 3 describes the methods and algorithms developed and their respective tests. Chapter 4 presents the results obtained from the tests made and their discussion. Chapter 5 has the conclusion made from the results obtained and also discusses possible future works. Appendix A shows the publications that directly resulted from this doctoral research and other publications that were the result of collaborations.

2 RELATED WORKS

In this chapter related works are described and an overview of what has been done on the topics is shown. This chapter is split into three sections each covering a topic important for this thesis. The three sections are Bayesian Networks Structural Learning (BNSL), the concept of Life in Genetic Algorithm, and quantization for BN.

2.1 Bayesian Networks Structural Learning (BNSL)

From 2016 to 2020, several works involving BNSL were published. Some of these discuss applications of structural learning of BN. Sevinc, Kucuk and Goltas (2020) use BNs to analyze the causes of forest fires using data from southwestern Turkey and conclude that the factors that most affect the occurrence of a fire are the month and temperature. Lytvynenko *et al.* (2019c) learn a BN to observe the dependencies of a country's Gross Domestic Product and provide investment strategies to increase it. Lytvynenko *et al.* (2019a) develop a BN model to be used in modeling drug distribution networks. Chen *et al.* (2019) learned a BN to provide early warning of accidents on Chinese roads. Lytvynenko *et al.* (2019b) use a BN to analyze the relationships of variables that affect public satisfaction with security agencies. Zhang *et al.* (2019) perform simulated transport tests considering different social factors and use a BN to simulate the synthetic population. Ruiz-Ruano and Puga (2019) learn a BN to analyze the variables that affect entrepreneurial intention and discover that the most important ones are self-efficacy, convenience, attitude, and social norm. Alfonso, Manjarrés and Pickin (2019) propose a semi-automatic method of educational competency maps from a repository of multiple-choice questionnaire answers. In Li and Liu (2018), a Dynamic Bayesian Network is used in conjunction with Wavelet decomposition to predict the path of a storm. Finally, in Ong, Yau and Looi (2016), BN is used to explore the relationships between the factors that affect the brand awareness of Malaysian university students and discover that the most significant factors are the state of origin and the year of study.

Others deal with the learning of modified BN models. In the work of Ramazzotti *et al.* (2019), a comparison is made between several structural learning methodologies for Suppes-Bayes Causal Networks, which is a subclass of BN. In the work of Villa and Stella (2018), non-stationary continuous-time Bayesian Networks are introduced, and their learning from real data is tested. Finally, in the work of Ding and Zhuang (2018), the distributed learning algorithm of BN is proposed for a cloud computing context.

Finally, some authors specifically deal with structural learning methods. In Correia, Cussens and Campos (2020), a structural learning method is proposed using pruning of the parent set considering the BDeu score. In Cambasi *et al.* (2019), a comparison of learning tools for Dynamic Bayesian Networks is made, using BayesiaLab, BayesFusion, Bayes Server, Netica, Hugin Lite, BNT, Banjo, Amidst, BNLearn, Infer.NET, Mocapy++, Libpgm, and Pgmpy. In Gross *et al.* (2019), a learning method based on the persistence of edge occurrence despite perturbation is proposed, considering an analytical threshold. The proposed threshold is derived from the random walk problem, and its value is calculated by the formula $f_{th}^{+} = (1/3) + \sqrt{2/K}$, where K is the number of learned structures. In Correia, Campos and Gaag (2019), an analysis of the effect of the Equivalent Sample Size of the BDeu score on the structural learning of BNs is made.

In Li *et al.* (2018), a BN learning method is proposed using multi-population bacterial foraging optimization. In Li and Guo (2018), a hybrid method of structural learning is proposed by combining search and score-based learning with expert knowledge. In this method, the explicit and vague knowledge of experts is considered in the scoring function. This scoring function is given by the equation

$$\begin{aligned} Score_{EVBIC} = BIC(D) + \sum_{j=1}^{R} \sum_{i=1}^{N_{explicit}} \log P({V_{explicit}}_i^j | e_i, \gamma^j) \\ + k \sum_{j=1}^{R} \sum_{i=1}^{N_{vague}} \log P({V_{vague}}_i^j | e_i, \beta^j) \end{aligned} \tag{2.1}$$

where $N_{explicit}$ is the number of pairs of nodes with explicit knowledge, N_{vague} is the number of pairs of nodes with vague knowledge, ${V_{explicit}}_i^j$ is the three types of explicit knowledge, ${V_{vague}}_i^j$ is the three types of vague knowledge, and $P({V_{explicit}}i^j | e_i, \gamma^j)$ and $P({V_{vague}}_i^j | e_i, \beta^j)$ are calculated by decision trees available in the article. In Scanagatta *et al.* (2018a), an approximate structural learning method for large BNs is proposed. In Scanagatta *et al.* (2018b), a learning method for BNs with a treewidth limit is presented. In Beretta *et al.* (2018), a comparison of structural learning methods for BNs is made. In Campos *et al.* (2018), a structural learning method is proposed using pruning of the parent set considering the BIC score.

In Amirkhani *et al.* (2017), the structural learning of BN using a score that incorporates expert opinions is proposed. The score is calculated by

$$Score_{explicit}(G; D, O, \gamma) = \log P(G) + \log P(D|G) + \log P(O|G, \gamma) \tag{2.2}$$

where G is the structure of the BN, D is the data, O is the expert opinions, and γ is the estimate of expert accuracy. In Liu *et al.* (2017), a hybrid method of structural learning for BNs is proposed, where a constraint-based method is used to divide the network into smaller pieces, then a search and score-based method is used to learn the BNs of these pieces, and finally, the pieces are joined together. In Yang *et al.* (2016), the bacterial foraging optimization algorithm is used to perform search and score-based structural learning of BNs. Finally, in Zhu, Liu and Jiang (2016), the artificial bee colony algorithm is used to perform search and score-based structural learning of BNs.

2.2 Concept of Life in Genetic Algorithm

The concept of life on GA was first introduced in Arabas, Michalewicz and Mulawka (1994). The paper proposes a Genetic Algorithm with Varying Population Size (GAVaPS), wherein each member of the population is endowed with a life value that dictates the number of generations it will remain a part of the population. Three distinct methods for determining this life parameter were proposed: linear, proportional, and bi-linear. All three methods calculate the life value based on various factors such as an individual's score, the mean score of the population, the highest and lowest scores within the population, as well as the highest and lowest scores discovered thus far. Despite being an old algorithm, it has been used in the works of Heng-zhou and Tao-shen (2016) and Abdelaziz (2017).

In 2000, Fernandes, Tavares and Rosa (2000) introduced an enhanced iteration of GAVaPS known as the non-incest Genetic Algorithm with Varying Population Size (niGAVaPS). In this version, the algorithm prohibits the reproduction of closely related individuals, aiming to preserve population diversity and prevent premature convergence. A year later, Fernandes and Rosa (2001) presented another upgraded variant of GAVaPS, termed the negative Assortative Mating Genetic Algorithm with Varying Population Size (nAMGAVaPS). The nAMGAVaPS employs non-random mating, wherein parents are chosen based on their phenotypic similarity. One parent is randomly selected, while the second parent is the one with the maximum Hamming distance from the first parent.

In contrast to earlier studies, Bäck, Eiben and Vaart (2000) introduces a GA that incorporates a life parameter similar to GAVaPS. However, this GA distinguishes itself by employing a self-adaptive mutation rate and crossover rate. Concerning the life parameter assignment method, the bi-linear approach is employed. It must be noted that the top-performing individual does not experience a reduction in its remaining life. Additionally, in contrast to GAVaPS, this GA generates and adds only two individuals during each iteration.

In a more contemporary context, Varnamkhasti and Hassan (2012) introduced a GA that incorporates the concept of life. This GA, known as the neurofuzzy inference systems genetic algorithm, also employs the bi-linear calculation method to determine life values. Furthermore, this algorithm incorporates gender and fuzzy logic in its approach. Through the utilization of fuzzy logic, it assigns linguistic labels such as “infant”, “adult” or “old” to the “age” of each individual.

Jianhua, yuanxiang and Lingling (2014) suggests incorporating a degradation mechanism into evolutionary algorithms. This mechanism categorizes individuals into three distinct age groups: childhood, prime, and old age. The assigned age group, coupled with other parameters like food consumption, significantly influences the individual's various processes, including reproduction. Unlike GAVaPS, in this proposed approach, there isn't a fixed age at which individuals are removed from the population. Instead, removal is determined by a death rate that escalates as an individual ages.

2.3 Quantization for Bayesian Networks

We conducted a literature review on the subjects of "Quantization" or "Discretization" and "Bayesian Network" resulting in the discovery of sixteen relevant papers. Several of these papers incorporate information theory into their methodologies, as exemplified by Friedman and Goldszmidt (1996) in which a method was introduced to adjust variable quantization during structural learning. This technique employed a metric based on the Minimum Description Length principle[1] (FRIEDMAN; GOLDSZMIDT, 1996) to determine the optimal quantization threshold. The performance of networks generated by various methods was evaluated using prediction accuracy (whether the predicted values match the actual values). In another study, Kozlov and Koller (1997) proposed a quantization approach that aimed to minimize information loss (maximizing the relative entropy between the quantized values and the actual values) caused by quantization through the use of non-uniform partitions.

Ciunkiewicz *et al.* (2022) introduced a dynamic quantization algorithm designed to pinpoint intervals that inadequately capture the underlying distribution, utilizing relative entropy error based on Kullback-Leibler divergence as a key metric. This approach involves a continuous process of merging and splitting intervals guided by this measurement, ultimately leading to intervals that faithfully represent the underlying distribution. To assess its efficacy, the proposed method was pitted against static quantization using a dataset pertaining to toxicity risk in breast radiotherapy. Despite the improved fidelity of quantization achieved by the proposed method, the predictive performance of the BN did not exhibit consistent enhancement.

In other studies, the techniques of expectation-maximization and likelihood are employed. For instance, in the work by Monti and Cooper (1998), a multi-variable quantization approach is introduced, wherein each continuous variable undergoes quantization based on its interactions with other variables. To achieve this, a scoring method is employed to assess the quality of the quantization, taking into account both the BN structure and the data.

Mabrouk *et al.* (2015) introduced a quantization technique that factors in the acquired conditional dependencies. This quantization process involves clustering and

[1] Model selection principle that dictates that the shortest description of the data is the best model.

leverages expectation-maximization based on the Gaussian mixture model. To assess its effectiveness, various quantization methods were subjected to comparison using the tabu search algorithm for structural learning. The evaluation of the learned BNs was based on several criteria, including the structural quality (similarity to the expected BN), computational time, and the parameter quality of the learned CPTs, gauged by their predictive accuracy.

Chen, Wheeler and Kochenderfer (2015) put forth a quantization approach designed for integration with the K2 algorithm during the structural learning of BNs. This method optimizes quantization by taking into account the network structure, which includes the variable's parents, children, and spouses. The evaluation of this quantization method involved a comparison with the Minimum Description Length (MDL) approach, considering the mean cross-validated log-likelihood of the data in relation to both the BN and the quantization.

Furthermore, there exist miscellaneous studies that tackle the quantization problem with distinct approaches. For instance, Song *et al.* (2011) presents a multi-variate quantization method that capitalizes on the strong correlations within high-dimensional data. This technique combines a non-parametric dimensionality reduction approach with a Gaussian mixture model. Initially, dimensionality reduction is applied, followed by quantization through the utilization of the mixture model. Notably, this method is specifically tailored for high-dimensional variables and underwent testing in the context of robot grasping.

In their work, Lima *et al.* (2014) introduces a quantization technique that operates by quantizing data based on peak and valley values. This approach quantizes all variables simultaneously using a GA, with the goal of minimizing the Normalized Root Mean Square Error for a selected output variable. The performance of this quantization method is assessed against quartiles-based quantization using a dataset related to the Bit's Rate of Penetration in the Brazilian pre-salt layer. The results reveal that the proposed method consistently outperformed the quartiles-based approach in terms of accuracy.

In their study, Fang *et al.* (2017) employs matrix decomposition as a means of conducting quantization. They introduce a quantization approach rooted in matrix decomposition, which enables the quantization of continuous variables into multiple states with varying probabilities. To evaluate this method, a two-node BN was employed, and the analysis was conducted with a focus on BN inference.

Talvitie, Eggeling and Koivisto (2019) introduced an algorithm for structural learning in BNs that incorporates an adaptive quantization technique for handling continuous variables. This method employs quantile quantization, with the number of divisions ranging from 2 to 7, depending on the model's free structural parameters. The algorithm's performance was assessed through a comparative analysis with other structural learning algorithms, considering the accuracy of the discovered structure and its predictive performance. Notably, the proposed procedure consistently outperformed the examined algorithms.

Mayfield *et al.* (2017) introduced the Structure Aware Discretization (SAD) algorithm, which is a structure-aware quantization method designed to optimize bin ranges and the number of bins in a way that strikes a balance between effectively filling CPTs and maintaining adequate resolution. SAD achieves this by reducing the number of bins to ensure that each bin contains a predefined minimum number of occurrences and to minimize the number of CPT combinations with insufficient cases. The study compared SAD with Equal Cases Discretization (ECD) and a method without structural awareness adjustment, called Structure-unaware Stage Discretization (SUD). These methods were evaluated using a water odor and taste dataset, with the assessment based on metrics such as the area under the receiver operating curve and the true skill statistic. Both SAD and SUD demonstrated similar performance, and both outperformed ECD.

Moreover, certain studies involve comparisons among various methods. For instance, in the work by Asghari, Qian and Stow (2017), a comparative analysis of quantization methods in BNs was conducted. This analysis examined the influence of quantization methods and the number of intervals on the resulting BNs. Notably, the BN structure was pre-established, and the bnlearn library was utilized to learn the CPTs based on the quantized data. The comparison encompassed CPTs, predictions, and recommendations. The findings indicated that these aspects varied across models, and there was no universally superior method across all metrics.

In Beuzen, Marshall and Splinter (2018), a comparison was conducted among quantization methods categorized into manual, supervised, and unsupervised classes. These methods were evaluated using a 4-node BN with a predefined structure, focusing solely on the learning of CPTs using quantized data. The comparison was based on two key metrics: model accuracy and the F-score. The findings revealed distinct strengths for each category of quantization method. Manual quantization methods yielded BNs with enhanced interpretability, supervised methods excelled in predictive capacity, and unsupervised methods demonstrated computational simplicity and versatility as their primary advantages.

In Ropero, Renooij and Gaag (2018), four distinct quantization methods were employed to discretize environmental data: Equal Frequency, Equal Width, Chi-Merge, and the Minimum Description Length principle. Of these methods, Chi-Merge stood out as a supervised approach that initially starts with intervals consisting of a single data point and employs Chi-Squared statistics to determine which intervals should be merged. Remarkably, Chi-Merge emerged as the top-performing method, demonstrating the best average performance in the conducted tests for the classification problem.

In Sari *et al.* (2021), a comparison was made among three quantization methods:

equal-width, equal-frequency, and K-means, using earthquake damage data. Notably, K-means outperformed the other methods, achieving the highest level of accuracy in the conducted test.

In Toropova and Tulupyeva (2022), an examination of the impact of various quantization methods on the performance of BNs was carried out to estimate behavioral rates. The methods under scrutiny included equal width, equal frequency, EF_Unique, and expert quantization. These methods were applied and evaluated using a dataset related to behavioral rates. The findings revealed that equal width quantization consistently delivered the highest and average levels of precision among the tested quantization techniques.

3 MATERIAL AND METHODS

To facilitate the development of the work, it was divided into four parts. The first part focused on creating or utilizing a parameter-tuning method for an evolutionary algorithm, ensuring that the search process effectively explored the search space and yielded good solutions for problems with varying characteristics. The second part involved the development of a BNSL algorithm that aimed to achieve better coverage of the search space. The third part studied the effects of quantization on the BNSL search space and the development of methods to help in quantizing data for BNs. Finally, the fourth part entailed the development of a scoring function to be used as one of the objectives within the evolutionary algorithm. Also, a section was made just describing all the datasets used. Below, you will find descriptions of how each of these aspects of the work was developed.

3.1 Datasets

In this section, the datasets used throughout this thesis are presented.

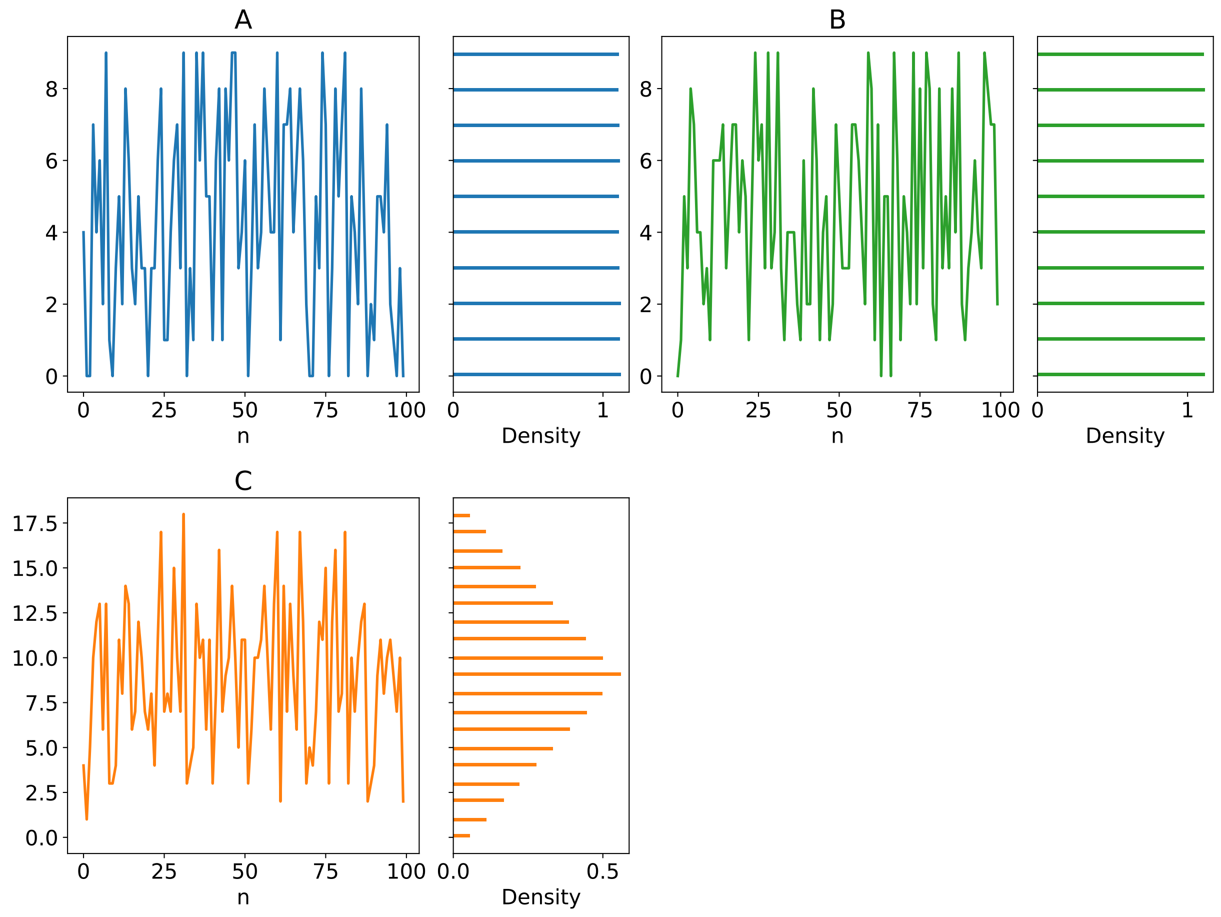
3.1.1 D3

D3 is a simulated discrete dataset. It was generated by the equation

$$\begin{cases} A = \text{rand\_int}(0, 9) \\ B = \text{rand\_int}(0, 9) \\ C = A + B \end{cases} \tag{3.1}$$

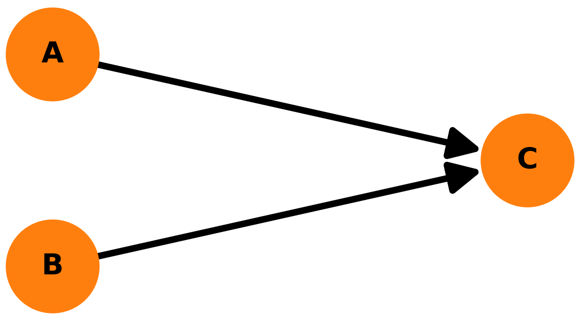
where $\text{rand\_int}(0, 9)$ indicates a sampling of a uniform distribution considering only the integer numbers from 0 to 9. A section of each variable's signal and a histogram of the variables can be seen in Figure 7. The expected BN structure for this dataset can be seen in Figure 8.

Figure 7 – A section of the signal of each variable and the variable's distribution for the D3 dataset. n is the number of the sample, density is the density of the histogram and the y-axis value is the amplitude.



Source: Author

Figure 8 – Expected BN structure for the D3 dataset.



Source: Author

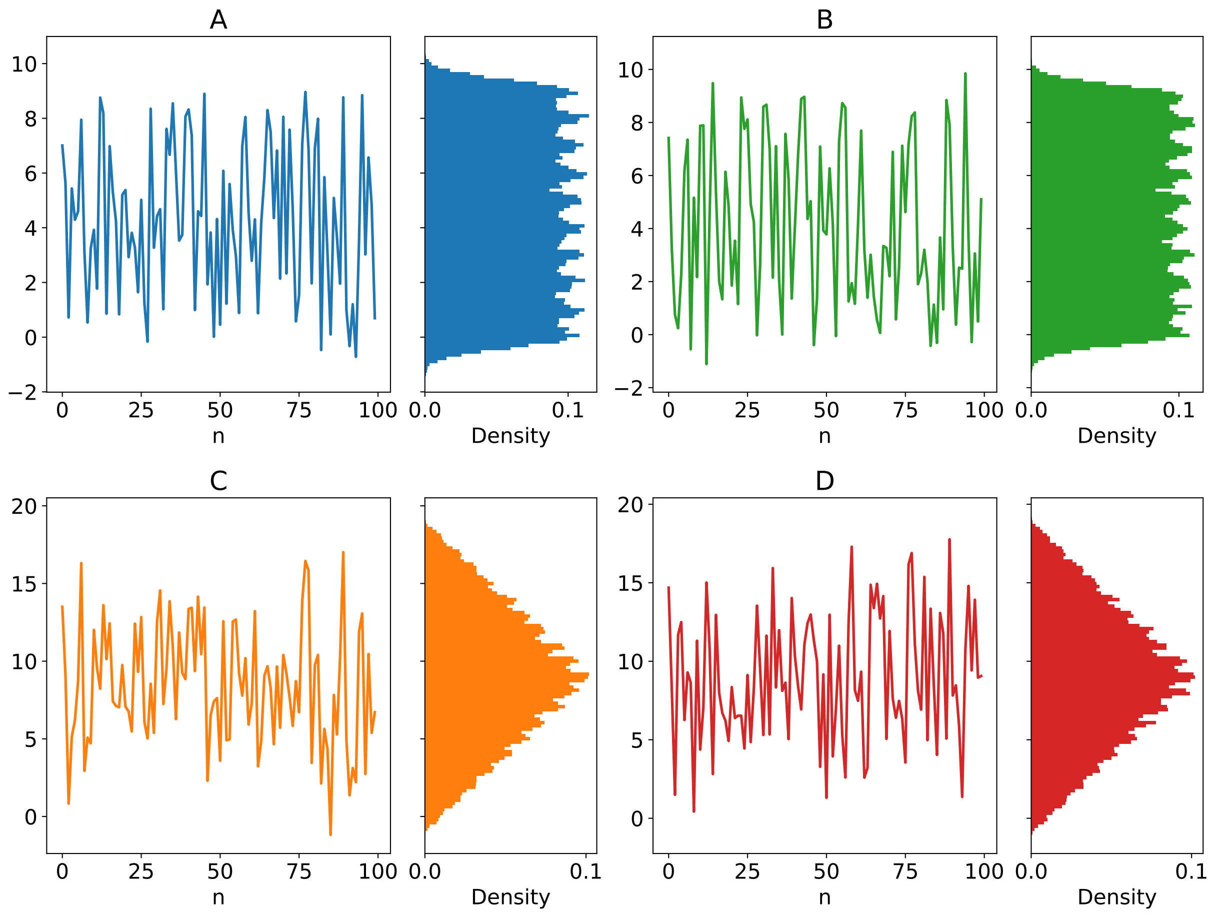
3.1.2 D4

D4 is a simulated discrete dataset with added noise to be turned into a simulated continuous dataset. It was generated by the equation

$$\begin{cases} A' = \text{rand\_int}(0,9) \\ B' = \text{rand\_int}(0,9) \\ C' = A' + B' \\ D' = A' + \text{rand\_int}(0,9) \\ A = A' + \mathcal{N}(\mu = 0, \sigma = 0.4) \\ B = B' + \mathcal{N}(\mu = 0, \sigma = 0.4) \\ C = C' + \mathcal{N}(\mu = 0, \sigma = 0.4) \\ D = D' + \mathcal{N}(\mu = 0, \sigma = 0.4) \end{cases} \tag{3.2}$$

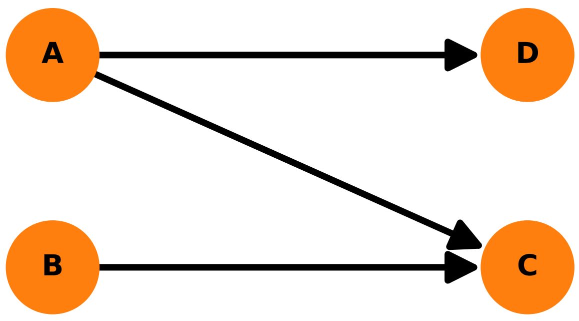
where $\text{rand\_int}(0,9)$ indicates a sampling of a uniform distribution considering only the integer numbers from 0 to 9 and $\mathcal{N}(\mu = 0, \sigma = 0.4)$ indicates a sampling of a normal distribution with $\mu = 0$ and $\sigma = 0.4$. A section of each variable's signal and a histogram of the variables can be seen in Figure 9. The expected BN structure for this dataset can be seen in Figure 10.

Figure 9 – A section of the signal of each variable and the variable's distribution for the D4 dataset. n is the number of the sample, density is the density of the histogram and the y-axis value is the amplitude.



Source: Ribeiro *et al.* (in press)

Figure 10 – Expected BN structure for the D4 dataset.



Source: Author

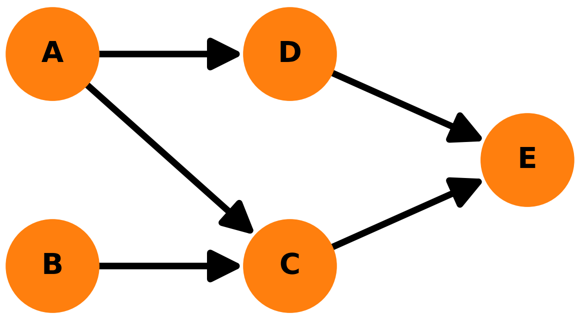
3.1.3 D5

D5 is a simulated discrete dataset. It was generated by the equation

$$\begin{cases} A = \text{rand\_int}(0,9) \\ B = \text{rand\_int}(0,9) \\ C = A + B \\ D = A + \text{rand\_int}(0,9) \\ E = C + D \end{cases} \tag{3.3}$$

where $\text{rand\_int}(0,9)$ indicates a sampling of a uniform distribution considering only the integer numbers from 0 to 9. The expected BN structure for this dataset can be seen in Figure 11.

Figure 11 – Expected BN structure for the D5 dataset.



Source: Author

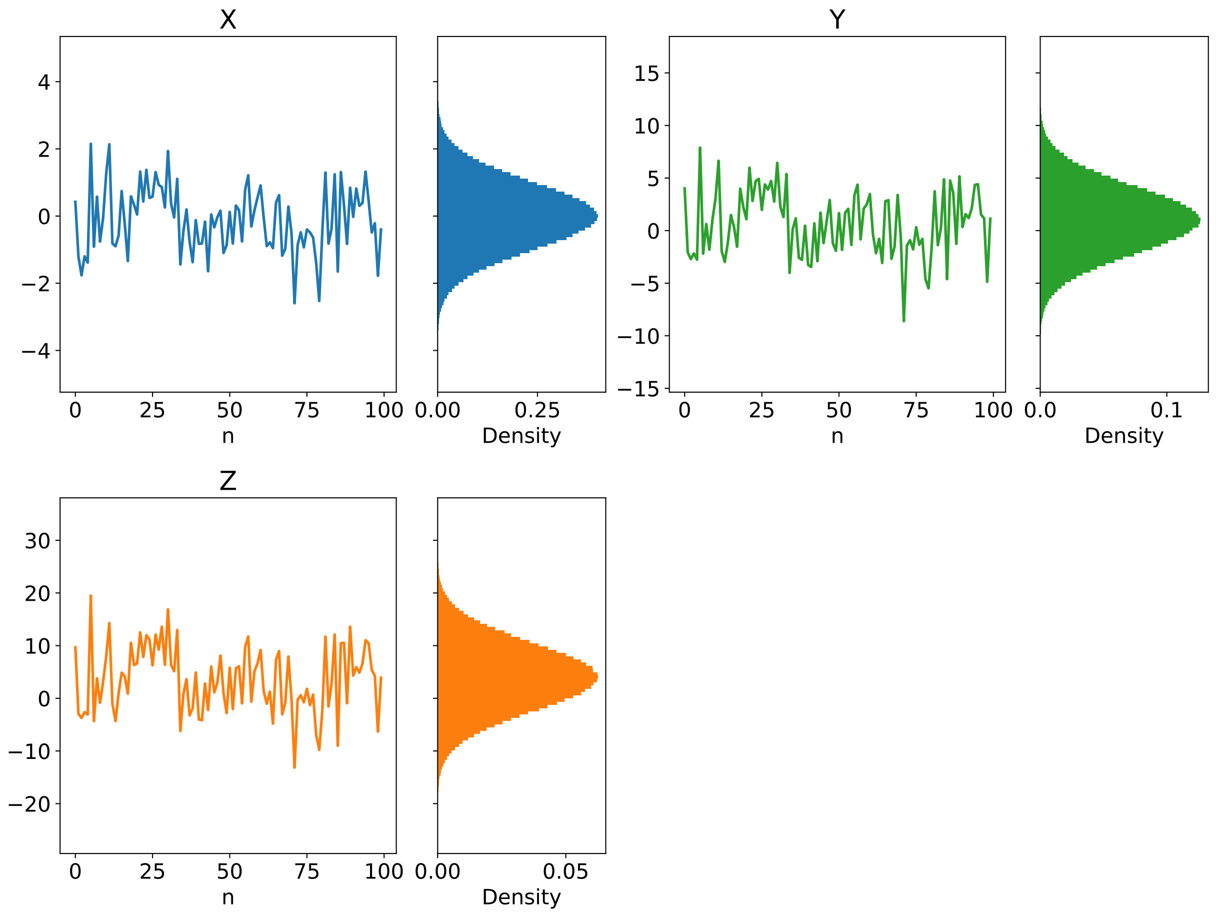
3.1.4 XYZ

XYZ is a simulated continuous dataset. It was generated by the equation

$$\begin{cases} X = \mathcal{N}(\mu = 0, \sigma = 1) \\ Y = 3 \cdot X + 1 + \mathcal{N}(\mu = 0, \sigma = 1) \\ Z = 2 \cdot X + 2 + \mathcal{N}(\mu = 0, \sigma = 1) \end{cases} \tag{3.4}$$

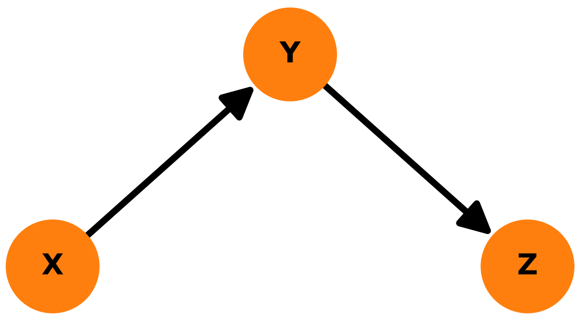
where $\mathcal{N}(\mu = 0, \sigma = 1)$ indicates a sampling of a normal distribution with $\mu = 0$ and $\sigma = 1$. A section of each variable's signal and a histogram of the variables can be seen in Figure 12. The expected BN structure for this dataset can be seen in Figure 13.

Figure 12 – A section of the signal of each variable and the variable's distribution for the XYZ dataset. n is the number of the sample, density is the density of the histogram and the y-axis value is the amplitude.



Source: Ribeiro *et al.* (in press)

Figure 13 – Expected BN structure for the XYZ dataset.



Source: Author

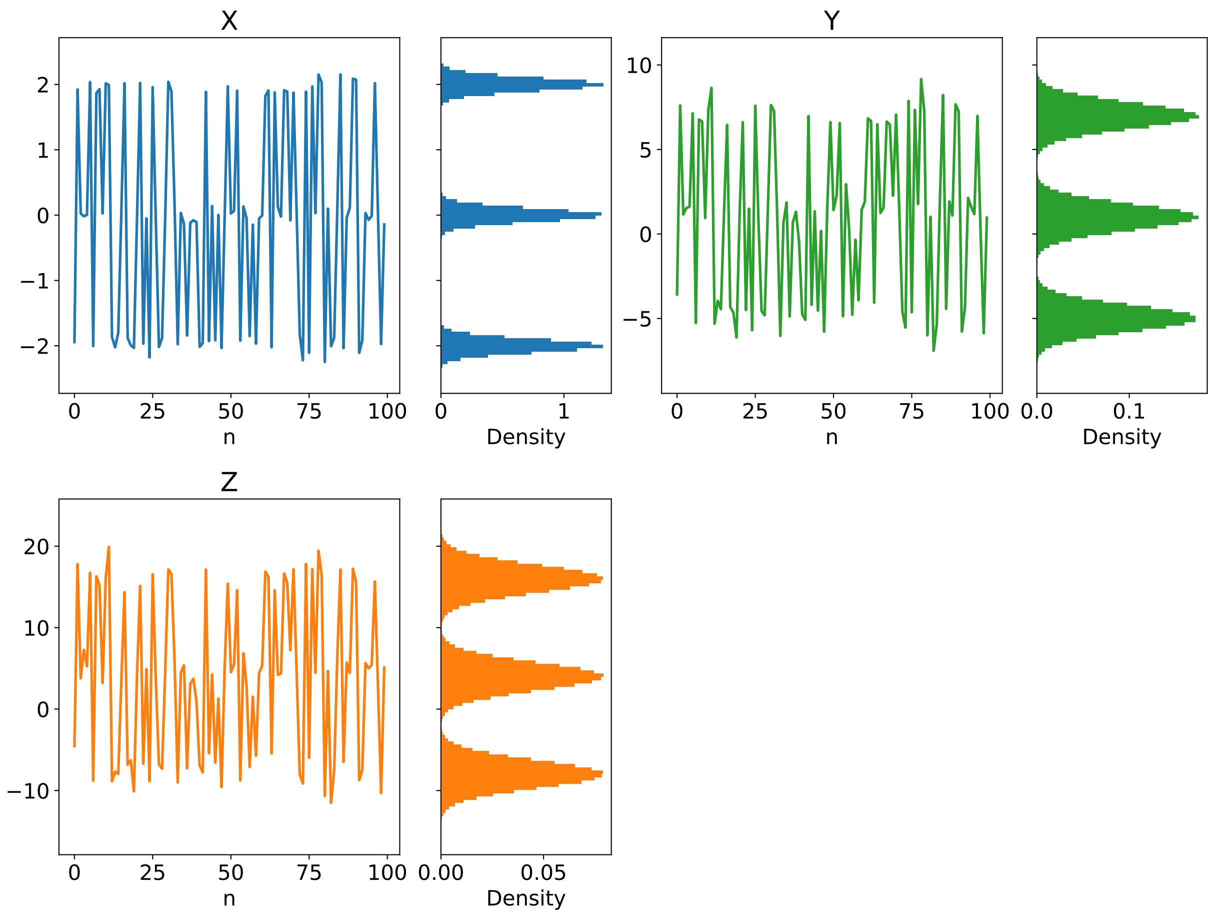
3.1.5 XYZ3

XYZ3 is a simulated continuous dataset. It was generated by the equation

$$\begin{cases} X = & \mathcal{N}(\mu = -2, \sigma = 0.1) \cup \mathcal{N}(\mu = 0, \sigma = 0.1) \\ & \cup \mathcal{N}(\mu = 2, \sigma = 0.1) \\ Y = & 3 \cdot X + 1 + 0.7 * \mathcal{N}(\mu = 0, \sigma = 1) \\ Z = & 2 \cdot X + 2 + 0.7 * \mathcal{N}(\mu = 0, \sigma = 1) \end{cases} \tag{3.5}$$

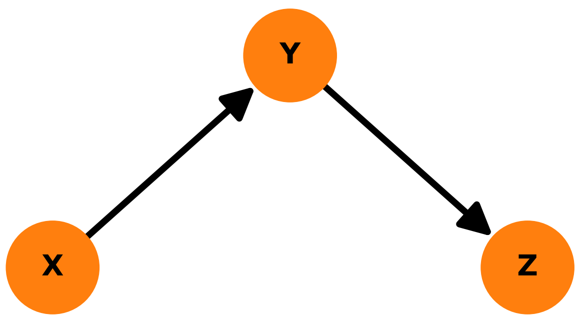
where $\mathcal{N}(\mu, \sigma)$ indicates a sampling of a normal distribution with μ and σ. A section of each variable's signal and a histogram of the variables can be seen in Figure 14. The expected BN structure for this dataset can be seen in Figure 15.

Figure 14 – A section of the signal of each variable and the variable's distribution for the XYZ3 dataset. n is the number of the sample, density is the density of the histogram and the y-axis value is the amplitude.



Source: Ribeiro *et al.* (in press)

Figure 15 – Expected BN structure for the XYZ3 dataset.



Source: Author

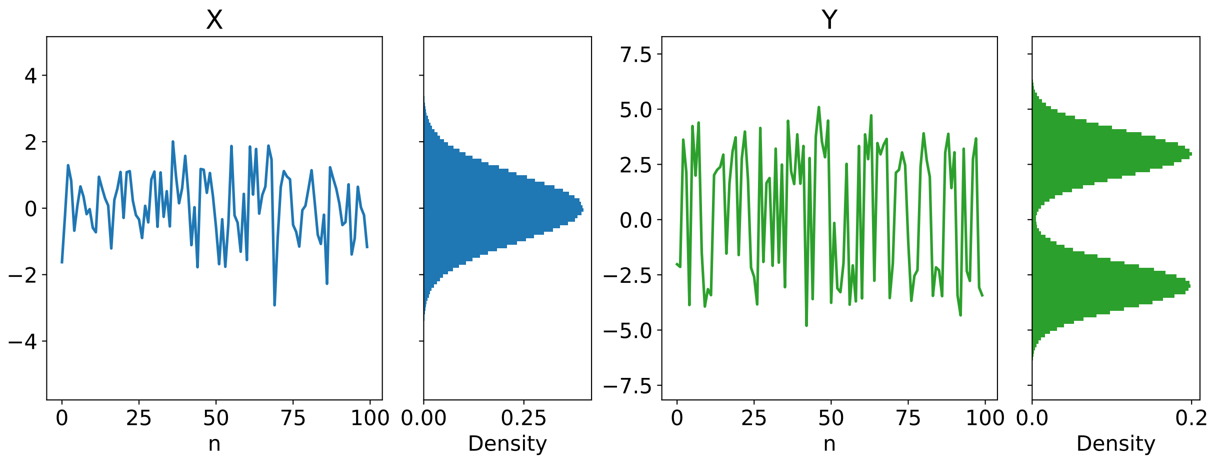
3.1.6 2T

2T is a simulated continuous dataset. It was generated by the equation

$$\begin{cases} X = \mathcal{N}(\mu = 0, \sigma = 1) \\ Y = \begin{cases} \mathcal{N}(\mu = 0, \sigma = 1) - 3 \text{ when } X \leq 0 \\ \mathcal{N}(\mu = 0, \sigma = 1) + 3 \text{ when } X > 0 \end{cases} \end{cases} \tag{3.6}$$

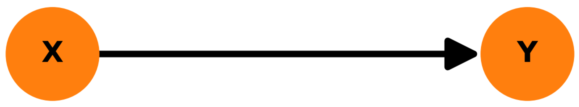
where $\mathcal{N}(\mu = 0, \sigma = 1)$ indicates a sampling of a normal distribution with $\mu = 0$ and $\sigma = 1$. A section of each variable's signal and a histogram of the variables can be seen in Figure 16. The expected BN structure for this dataset can be seen in Figure 17.

Figure 16 – A section of the signal of each variable and the variable's distribution for the 2T dataset. n is the number of the sample, density is the density of the histogram and the y-axis value is the amplitude.



Source: Author

Figure 17 – Expected BN structure for the 2T dataset.

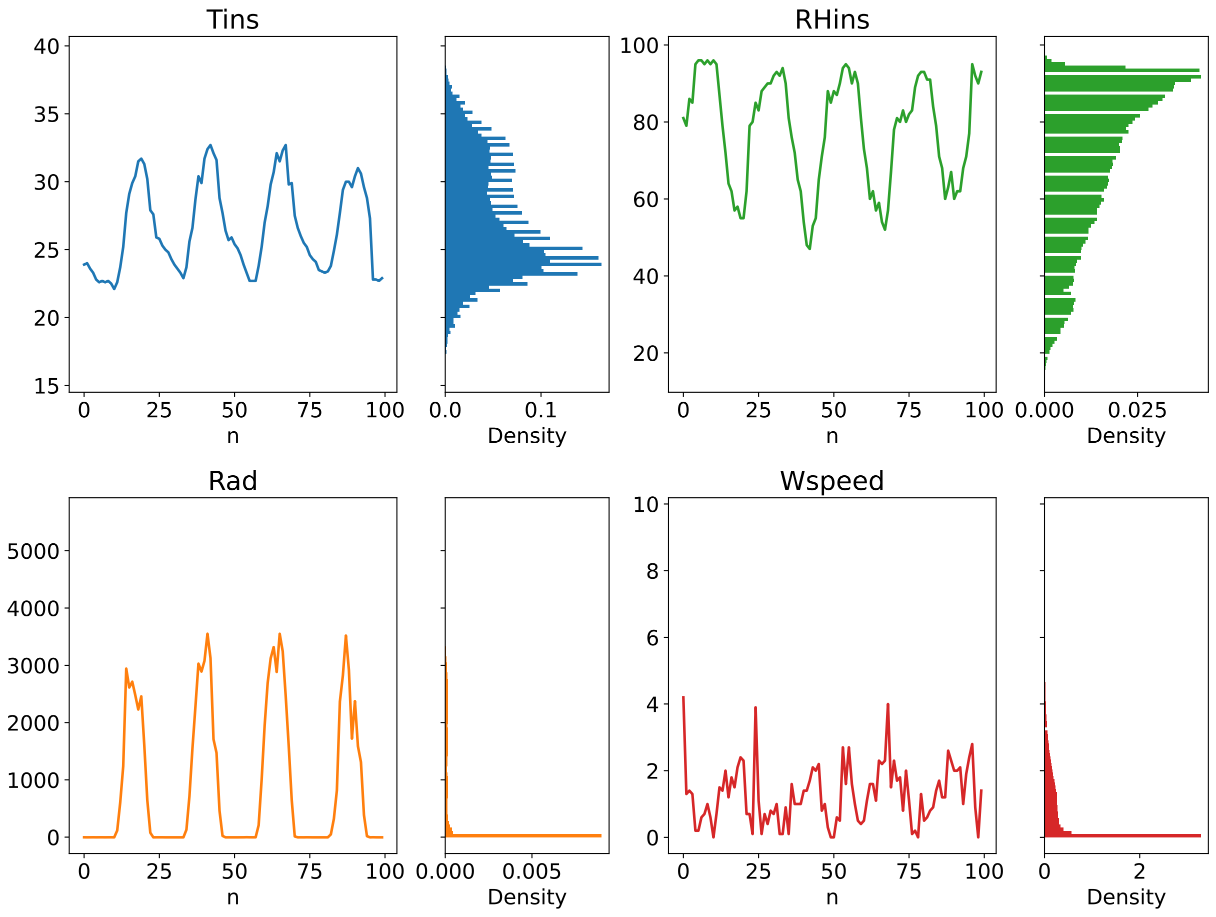


Source: Author

3.1.7 Weather

Weather is a real data dataset. It was acquired from Imperatriz (5° 31′ 33″ S, 47° 28′ 33″ W), a Brazilian city in the state of Maranhão. It contains hourly measurements of temperature, humidity, radiation and wind speed. Measurements started at 23:00 on 02/03/2008 and ended at 14:00 on 21/01/2022. The data was obtained from the National Institute of Meteorology (INMET), accessed at https://tempo.inmet.gov.br/TabelaEstacoes/A225. A section of each variable's signal and a histogram of the variables can be seen in Figure 18.

Figure 18 – A section of the signal of each variable and the variable's distribution for the Weather dataset. n is the number of the sample, density is the density of the histogram and the y-axis value is the amplitude.



Source: Ribeiro *et al.* (in press)

3.1.8 Skin

Skin is a real data dataset from the literature called Skin Segmentation that can be found at the UCI Machine Learning Repository (BHATT; DHALL, 2012). The skin dataset is collected by randomly sampling R, G, and B values from face images of various age groups, race groups, and genders. The dataset has 4 variables, the RGB values and the classification (skin or non-skin).

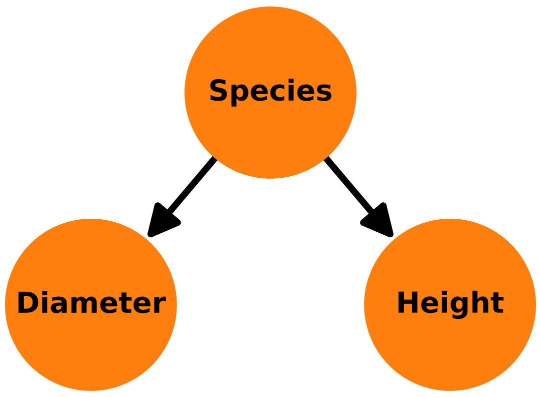
3.1.9 Haberman

Haberman is a real data dataset from the literature called Haberman's Survival that can be found at the UCI Machine Learning Repository (HABERMAN, 1999). The dataset contains cases from a study that was conducted between 1958 and 1970 at the University of Chicago's Billings Hospital on the survival of patients who had undergone surgery for breast cancer. It has 4 variables: age, operation year, positive auxillary nodes, and survival status.

3.1.10 Lizards

Lizards is a literature dataset for BNSL about the perching behavior of two species of lizards in South Bimini island. More details about it can be seen at https://www.bnlearn.com/documentation/man/lizards.html. It has 3 variables and its expected structure can be seen in Figure 19.

Figure 19 – Expected BN structure for the Lizards dataset. Species refers to the species of the lizard, Height refers to the perch height and Diameter refers to the perch diameter.

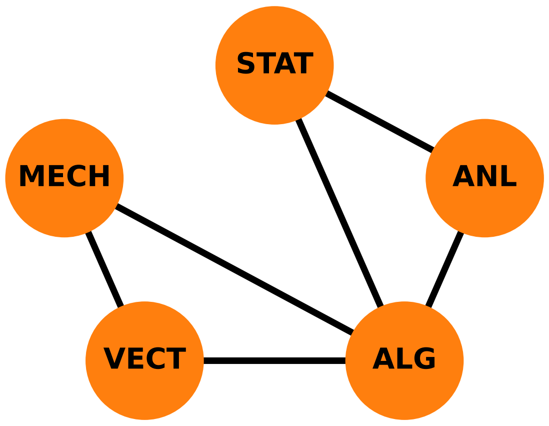


Source: Author

3.1.11 Marks

Marks is a literature dataset for BNSL containing examination marks of 88 students on five different topics. More details about it can be seen at https://www.bnlearn.com/documentation/man/marks.html. It has 5 variables and its expected structure can be seen in Figure 20.

Figure 20 – Expected BN structure for the Marks dataset. It must be noted that the expected structure for this dataset is a super-structure, meaning that it only has edges with no direction.

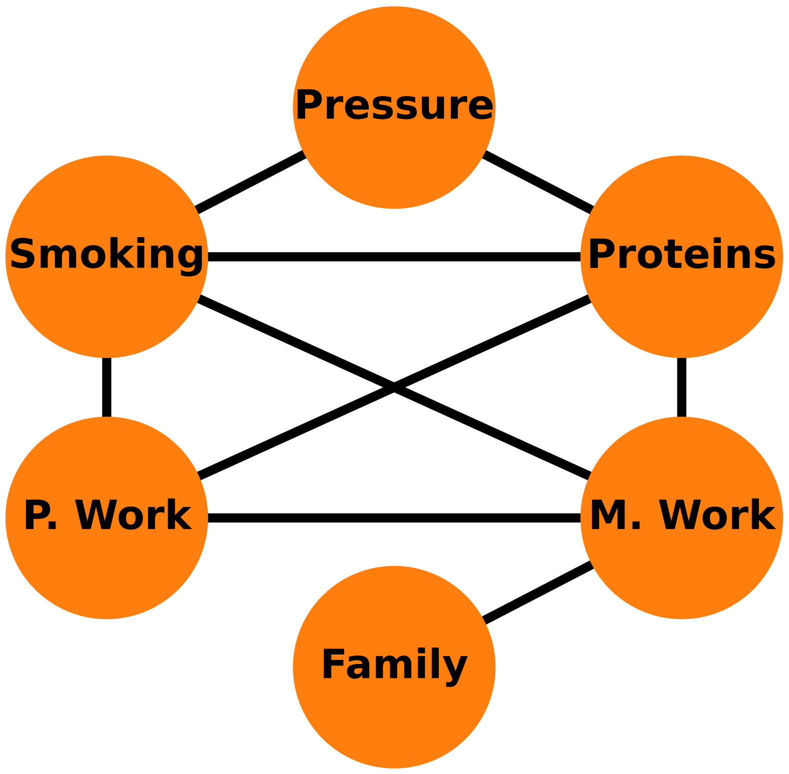


Source: Author

3.1.12 Coronary

Coronary is a literature dataset for BNSL containing the probable risk factors for coronary thrombosis from 1841 men. More details about it can be seen at https://www.bnlearn.com/documentation/man/coronary.html. It has 6 variables and its expected structure can be seen in Figure 21.

Figure 21 – Expected BN structure for the Coronary dataset. It must be noted that the expected structure for this dataset is a super-structure, meaning that it only has edges with no direction.

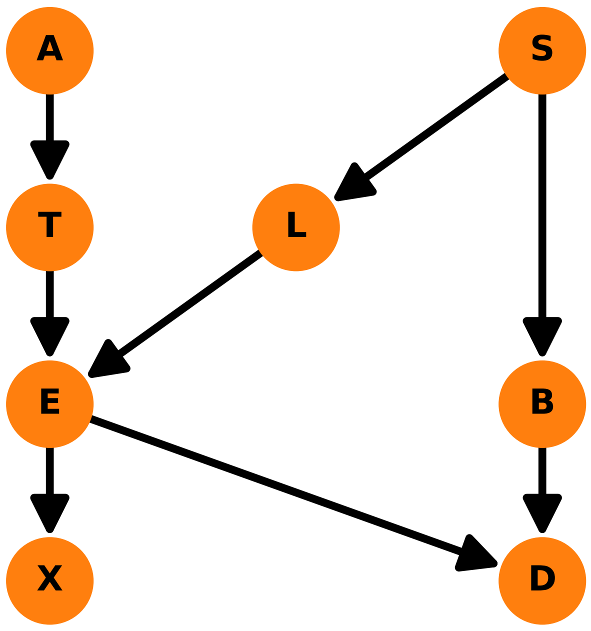


Source: Author

3.1.13 Asia

Asia is a literature dataset for BNSL about lung diseases (tuberculosis, lung cancer, or bronchitis) and visits to Asia. More details about it can be seen at https://www.bnlearn.com/documentation/man/asia.html. It has 8 variables and its expected structure can be seen in Figure 22.

Figure 22 – Expected BN structure for the Asia dataset. D refers to dyspnoea, T refers to tuberculosis, L refers to lung cancer, B refers to bronchitis, A refers to visit to Asia, S refers to smoking, X refers to chest X-ray and E refers to tuberculosis versus lung cancer/bronchitis.

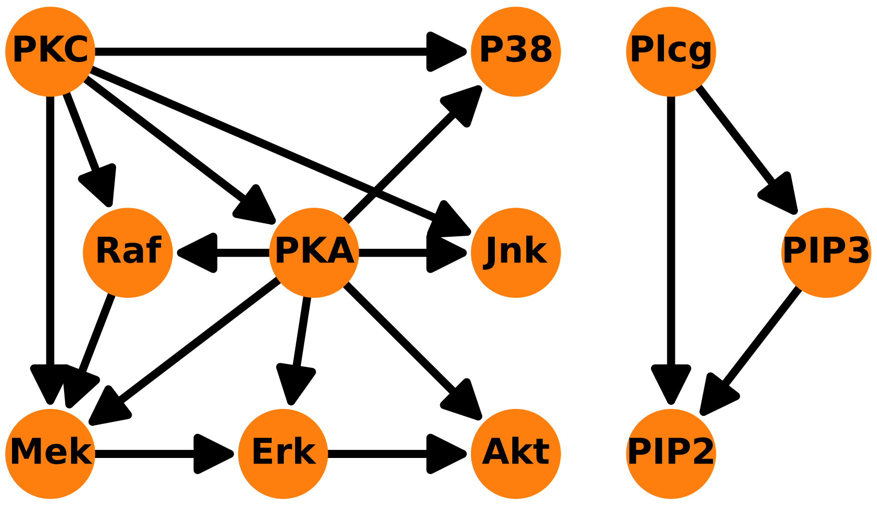


Source: Author

3.1.14 Sachs

Sachs is a literature dataset for BNSL with Protein-Signaling data. It has 11 variables and its expected structure can be seen in Figure 23.

Figure 23 – Expected BN structure for the Sachs dataset. All variables are target molecules measured in the original study by Sachs *et al.* (2005). Raf refers to phosphorylation at S259, Erk1 and Erk2 refer to phosphorylation at T202 and Y204, P38 refers to phosphorylation at T180 and Y182, Jnk refers to phosphorylation at T183 and Y185, AKT refers to phosphorylation at S473, Mek1 and Mek2 refers to phosphorylation at S217 and S221, PKA detects proteins and peptides containing a phospho-Ser/Thr residue with arginine at the -3 position, PKC detects phosphorylated PKC -α, -βI, -βII, -δ, -ϵ, -η, and -θ isoforms only at C-terminal residue homologous to S660 of PKC-βII, PLCg refers to phosphorylation at Y783, PIP2 detects PIP2 and PIP3 detects PIP3.

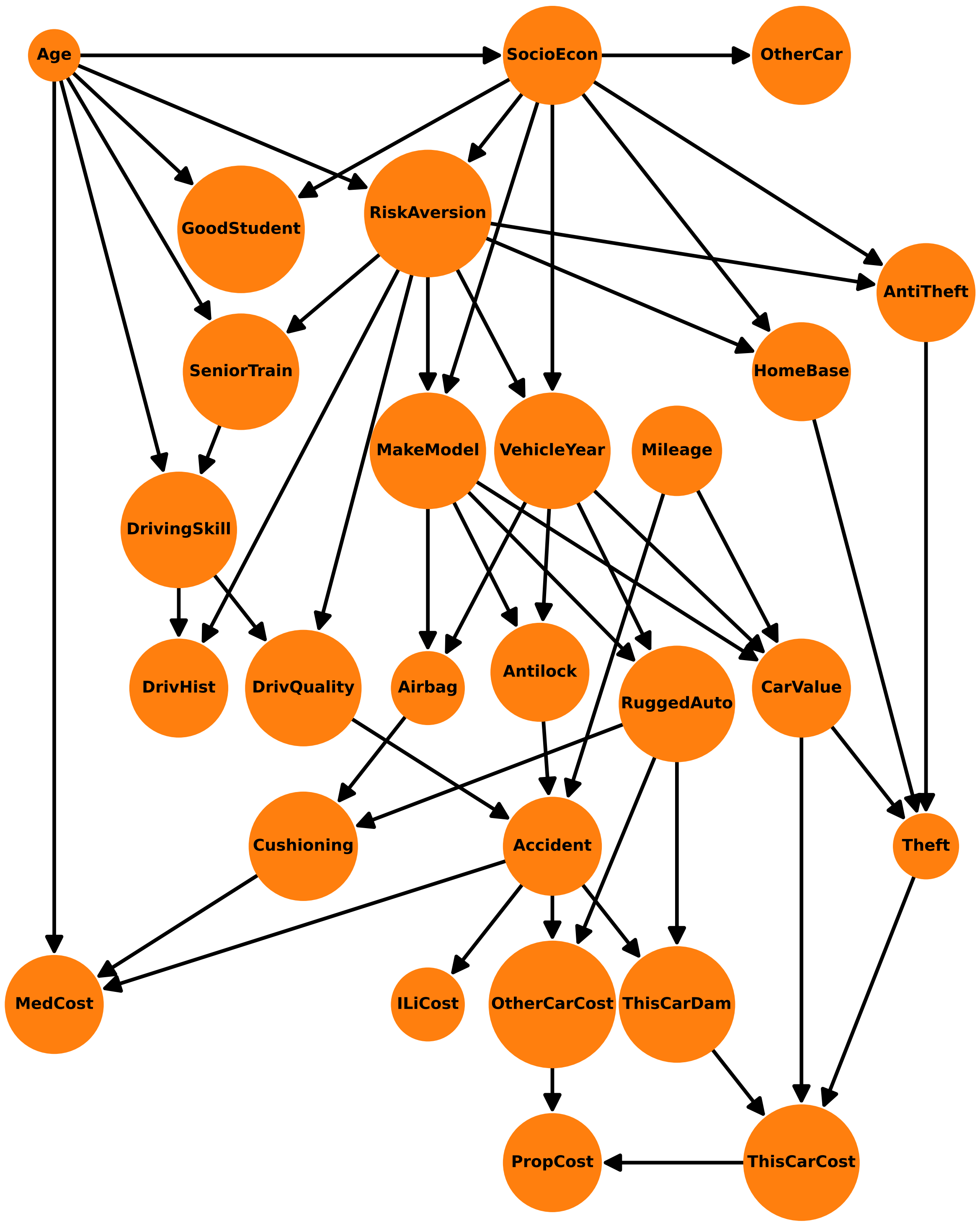


Source: Author

3.1.15 Insurance

Insurance is a literature dataset for BNSL which was created to estimate the claim cost for a car insurance policyholder. It has 27 variables and its expected structure can be seen in Figure 24.

Figure 24 – Expected BN structure for the Insurance dataset.

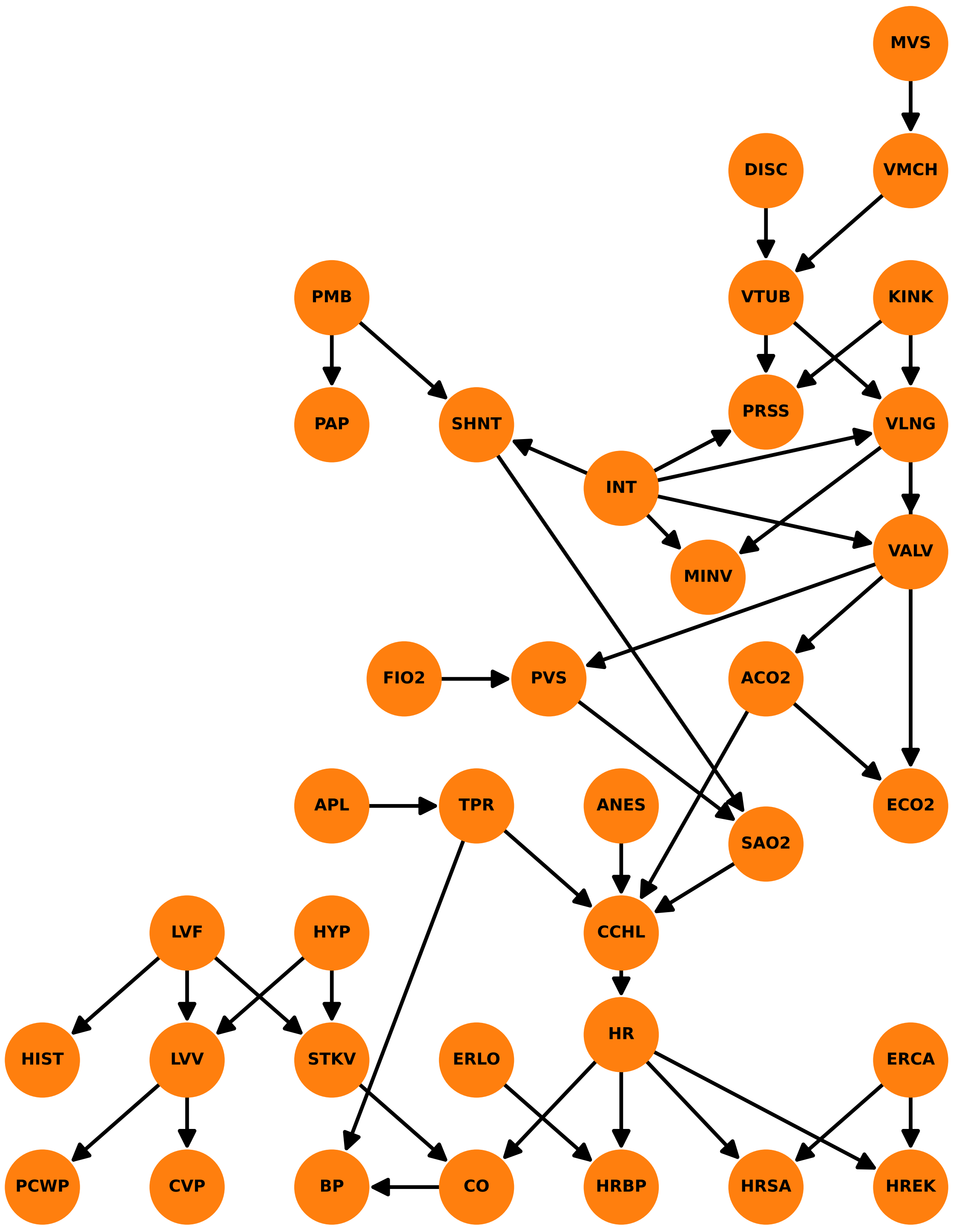


Source: Author

3.1.16 Alarm

Alarm is a literature dataset for BNSL which was created to implement an alarm message system for patient monitoring. It has 37 variables and its expected structure can be seen in Figure 25.

Figure 25 – Expected BN structure for the Alarm dataset.

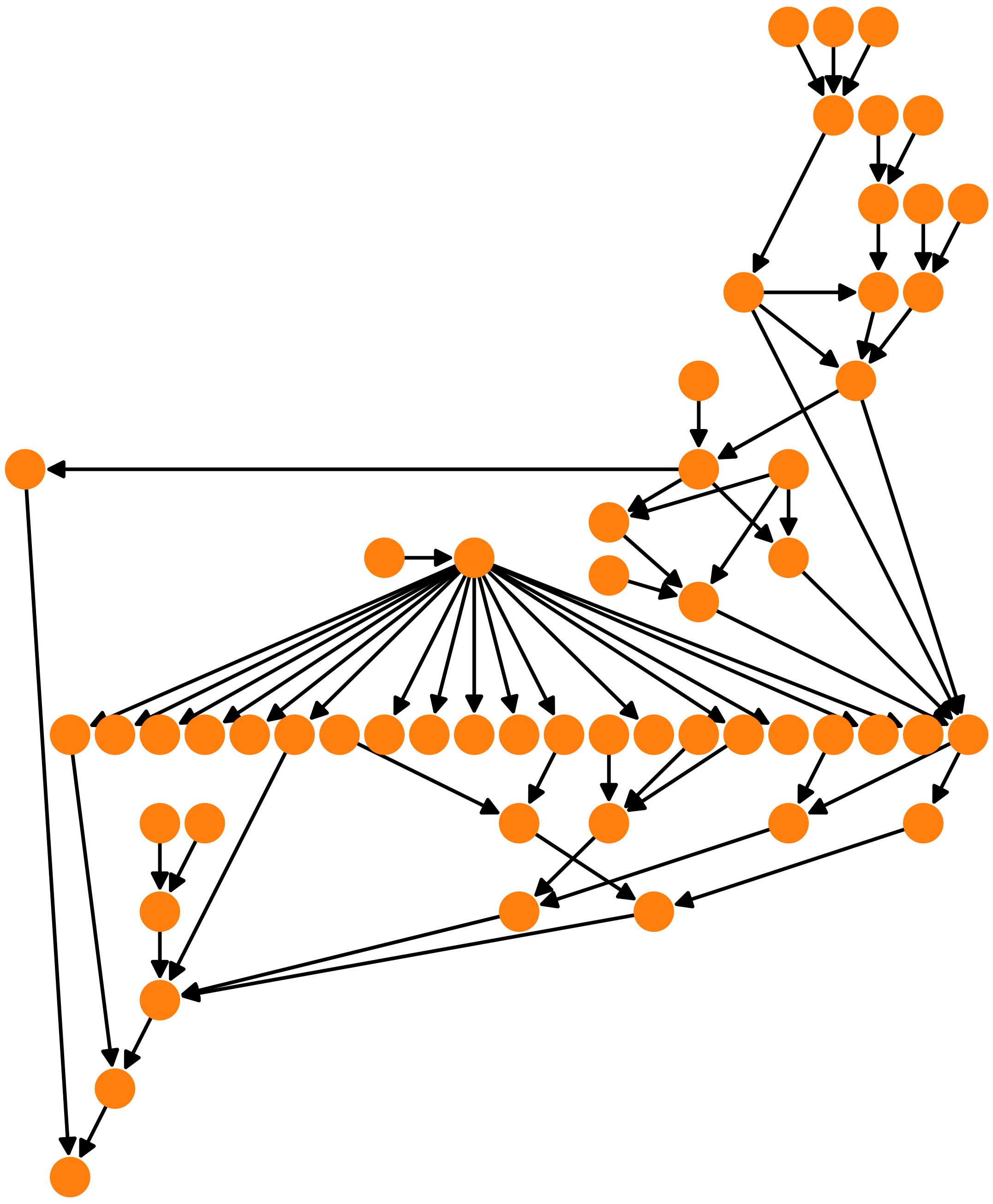


Source: Author

3.1.17 Hailfinder

Hailfinder is a literature dataset for BNSL which was created for forecasting severe weather. It has 56 variables and its expected structure can be seen in Figure 26. More information can be found in https://www.bnlearn.com/documentation/man/hailfinder.html and on its original paper by Abramson *et al.* (1996).

Figure 26 – Expected BN structure for the Hailfinder dataset. All nodes are weather measurements and weather forecast parameters. As the structure is really big and the names of the variables would not be readable, they were not included.



Source: Author

3.2 Parameter Tuning

The work on parameter tuning began with a literature review on the subject, where several relevant studies were identified. In the study by Branke and Elomari (2011), an evolutionary algorithm is used to tune another evolutionary algorithm. In Trindade and Campelo (2019), sequential optimization of regression models is employed to fine-tune metaheuristics. Tatsis and Parsopoulos (2019) utilizes approximate gradient search and line search in the parameter domain for tuning. López-Ibáñez *et al.* (2016) introduces the irace package, which uses iterated racing procedures for tuning. Montero, Riff and Rojas-Morales (2018) reviews tuning methods and analyzes the impact of parameter settings on the final results. Skakov and Malysh (2018) employs an evolutionary algorithm for tuning. Vogel and Wilke (2018) proposed a tuning method for problems with restricted budgets, where each parameter vector is evaluated only once. Finally, Joshi and Bansal (2020) adopts an approach based on the relationship between algorithm performance and functional landscape.

Considering the articles found, it was decided to test an implementation of a GA for offline parameterization (parameterization that is not performed during the execution of the algorithm (TATSIS; PARSOPOULOS, 2019; LÓPEZ-IBÁÑEZ *et al.*, 2016)). An attempt was also made to reproduce the method proposed in Vogel and Wilke (2018) because it is an interesting approach for structural learning algorithms that have a time-consuming execution when dealing with a BN with a higher number of variables.

The implementation tests of the GA for tuning did not yield promising results in initial tests, and reproducing the method proposed in Vogel and Wilke (2018) was not possible. Given this, it was decided to explore online parameterization (parameter control), where the search algorithm's parameters are adjusted during its execution (TATSIS; PARSOPOULOS, 2019; LÓPEZ-IBÁÑEZ *et al.*, 2016). To do this, a new evolutionary algorithm was implemented to adapt its parameters during execution to improve the search space coverage. This method will be described later on.

3.3 BNSL Algorithm

Due to the widespread use of the genetic algorithm (DUDEK; JANIGA; WOJNAROWSKI, 2021), it was decided to create an initial implementation of a multi-population genetic algorithm with population mixing. In this algorithm, mutation is performed by randomly selecting between changing the direction of an random edge, removing an random edge, or adding an random edge in a way that the Bayesian network does not form any cycles. Generating new members of the population involves copying one of the parents and adding the edges that are different between the two parents until the resulting Bayesian network has no cycles. Selection of the population for the next generation is done by preserving the best individuals in the population and randomly removing members from the rest of the population until the desired number is reached.

This method was developed, but initial tests did not show good performance in terms of parameter tuning, as mentioned earlier. After deciding to explore online parameterization, the algorithm was modified so that its parameters would be adjusted during execution. To achieve better coverage of the search space, it was decided to use a variable-sized population based on the concept of lifetime, and the mutation rate would be adjusted according to the current population. An in-depth discussion about the concept of life in GAs can be seen in Section 2.2.

The proposed method was named Adaptive Genetic Algorithm with Varying Population Size (AGAVaPS). It updates its parameters during execution and is focused on better covering the search space. In AGAVaPS, each individual has its own mutation rate and a parameter called "life" that determines how many iterations that individual will be part of the population. The mutation rate of each individual is sampled from a normal distribution $\mathcal{N}(\mu_{mut}, 0.1)$, where μ_{mut} is initialized to 0.5, and its value is updated at each iteration based on the average mutation rates of the current population. This update follows the formula:

$$\begin{aligned} \mu_{mut}(n)' &= \mu_{pop}(n-1) - (\mu_{mut}(n-1) - \mu_{pop}(n-1)) \\ \mu_{mut}(n) &= \begin{cases} \mu_{mut}(n)', & \text{if } 0.3 < \mu_{mut}(n)' < 0.7 \\ \mu_{pop}(n-1) - \mathcal{U}(0.1, 0.3), & \text{if } 0.7 \leq \mu_{mut}(n)' \\ \mu_{pop}(n-1) + \mathcal{U}(0.1, 0.3), & \text{if } \mu_{mut}(n)' \leq 0.3 \end{cases} \end{aligned} \tag{3.7}$$

in this way, the search conducted by the algorithm adapts according to the population. Therefore, if the population has an average mutation rate higher than the mean used in the sampling, it indicates that exploration is successful and will be reinforced by increasing the mean for the next sampling. In the opposite situation, if the population's average is lower than the sampling mean, it suggests that the population is achieving greater success in local search, and it will be reinforced by decreasing the sampling mean. Finally, a limitation was imposed on extreme values in the sampling mean to prevent the search from saturating at one of the extremes. To achieve this, if the updated mean reaches one of these limits, the population mean is perturbed in the opposite direction using a sample from a uniform distribution.

The life parameter is generated by the equation:

$$life = 10 \cdot \mathcal{N}(0.5, 0.15) \tag{3.8}$$

where *life* $\in \mathbb{N}^*$ and $\mathcal{N}(0.5, 0.15)$ indicates a sample of a normal distribution with $\mu = 0.5$ and $\sigma = 0.15$. This way, each individual has a number of iterations in which it will be part of the population. In each iteration, the lifetimes of individuals are decremented by 1, except for the top 10% best individuals that are preserved. When the life value reaches 0, the individual is removed from the population.

Additionally, the possibility for an individual to undergo the mutation process twice was added to increase the coverage of the search space. This is done as follows:

$$\begin{cases} \text{double mutation,} & \text{if } r < 0.15 \cdot \textit{mutation rate} \\ \text{single mutation,} & \text{if } 0.15 \cdot \textit{mutation rate} \leq r < \textit{mutation rate} \\ \text{no mutation,} & \text{otherwise} \end{cases} \tag{3.9}$$

where r is a randomly sampled number from a uniform distribution between 0 and 1.

The life parameter causes the population to have a variable size. The population has an initial size and a maximum size. This maximum size is set to limit the population's size due to computational constraints when necessary. If the population size exceeds the maximum value, individuals with the lowest lifetime values are removed from the population until the size matches the limit. During this removal process, the top 10% best individuals in the population are protected from removal.

The reproduction (crossover) is performed using two individuals (parents) to generate one individual (child). Parents are selected through a tournament of size "tournament size", with the default value being 3. Selection by tournament functions by randomly selecting a number of solutions equal to the tournament size and selecting the best solution (according to score/objective function value) between them to be used for the reproduction. The reproduction rate (γ) is fixed at 0.5. This way, it is expected that in the initial iterations, the population will grow in size until the number of new individuals and the number of individuals removed become approximately equal. When this happens, a balance is achieved.

This optimizer was then tested in three test cases to analyze its performance and behavior. The first test was a behavior test of the search process. The second was an optimization test using benchmark functions and comparing the performance with other algorithms. The third was an application test for complex problems, specifically in feature selection and BNSL.

For the first and second tests, the benchmark used was the single-objective optimization benchmark from CEC2017[1]. It consists of 30 functions, all minimization problems that have been shifted, rotated, and rescaled to the $[-100, 100]^D$ range. The minimum

[1] https://github.com/tilleyd/cec2017-py(AWAD *et al.*, 2016; TILLEY, 2020)

value of each function is $100 * i$ where i is the index of the function (AWAD *et al.*, 2016).

The thirty functions are divided into four types. $f1$ to $f3$ are unimodal functions. $f4$ to $f10$ are simple multimodal functions. $f11$ to $f20$ are hybrid functions. And $f21$ to $f30$ are composite functions (AWAD *et al.*, 2016). More information about the benchmark can be found in its definition document (AWAD *et al.*, 2016) and in the Python implementation used (TILLEY, 2020).

For the tests, all functions were used in their 10-dimensional (10D) forms. They were considered with both free initial space ($[-100, 100]^D$) and limited initial space ($[80, 100]^D$), where $D = 10$. The tests with limited initial space evaluate the optimizer's ability to explore the search space even in more challenging situations.

In the search behavior test, AGAVaPS was executed for function $f1$ with limited initial space and 20,000 evaluations of the objective function. At each iteration, the following values were recorded: score, population size, the number of individuals added to the population (children), the number of duplicate children removed, and the number of individuals removed from the population. These data were then plotted and analyzed.

For the comparison test, AGAVaPS was compared with five other optimizers. These five were chosen because they are commonly used and have their implementations provided by the EvoloPy library[2] (FARIS *et al.*, 2016a). The five optimizers are Genetic Algorithm (GA) (HANH *et al.*, 2019), Particle Swarm Optimization (PSO) (XIA *et al.*, 2020; KENNEDY; EBERHART, 1995), Firefly Algorithm (FFA) (YANG, 2010a), Cockoo Search Algorithm (CS) (YANG; DEB, 2009) and Bat Algorithm (BAT) (YANG, 2010b).

For this test, AGAVaPS had its reproduction and mutation methods identical to the GA implementation from EvoloPy. Reproduction involves blending the coordinates of the two parents at a randomly chosen point. Mutation is performed by resampling one of the coordinates. Additionally, AGAVaPS was tested in two different configurations, one with *tournament size* $= 3$ (referred to as AGAVaPS) and another more compatible with the low elitism of the GA with *tournament size* $= 1$ (referred to as AGAVaPS*). The population size used for both AGAVaPS configurations and the GA was 50.

The parameters used for the other algorithms were recommended by Kazikova, Pluhacek and Senkerik (2020), who conducted an analysis of the same implementation for this same benchmark. The parameter values are shown in Table 3.

[2] https://github.com/7ossam81/EvoloPy (FARIS *et al.*, 2016b)

Table 3 – Parameters used in the preliminary comparison test of AGAVaPS. The choice of the parameter was made considering the work Kazikova, Pluhacek and Senkerik (2020) and the parameters and their meaning are explained on the proposal articles PSO (XIA *et al.*, 2020; KENNEDY; EBERHART, 1995), FFA (YANG, 2010a), CS (YANG; DEB, 2009) and BAT (YANG, 2010b).

Optimizer	**Parameter**
PSO	Acceleration coefficients $c_1 = c_2 = 1.9$ Maximum speed $V_{MAX} = 6$ Maximum inertia $w_{MAX} = 0.5$ Minimum inertia $w_{MIN} = 0.5$ Population size of 30
FFA	Randomness $\alpha = 0.25$ Absorption coefficient $\gamma = 1$ Minimum beta $\beta_{MIN} = 0.2$ $\beta_0 = 1.0$ Population size of 20
CS	Discovery rate $p_a = 0.25$ $\alpha = 0.01$ Population size of 20
BAT	Loudness $A = 1.5$ Pulse rate $r = 0.5$ Minimum frequency $Q_{MIN} = 0$ Maximum frequency $Q_{MAX} = 2$ Population size of 50

Source: Author.

Each of the optimizers was run 100 times for each of the 30 functions, saving three performance measures for each run: grid coverage, best solution found, and execution time. This was done for both the case with free initial space and limited initial space. The number of objective function evaluations was limited to 100,000 evaluations ($10 \cdot 10,000$), as suggested in Awad *et al.* (2016) and also used in Kazikova, Pluhacek and Senkerik (2020).

The grid coverage measure aims to quantify how well the optimizer explored the search space. It is calculated by determining the percentage of quadrants in a grid with 100 bins that were explored by the algorithm. A quadrant is considered explored if at least one point within it was evaluated by the optimizer. For all measures, the mean and standard deviation of the 100 runs were calculated.

To compare the performance of the algorithms, an analysis similar to that conducted in Salgotra *et al.* (2021) was performed. To check for differences in performance between the algorithms, a Wilcoxon test with a significance level of 0.05 was conducted. The algorithms were ranked (f-rank) based on their performance considering the results of the

Wilcoxon tests (DERRAC *et al.*, 2011). It should be noted that for cases where algorithm A had statistically equivalent performance to algorithms B and C, but algorithms B and C did not exhibit this equivalence according to the Wilcoxon test, algorithms A, B, and C were not grouped in the f-rank.

Additionally, the performance of each algorithm was compared to the performance of AGAVaPS (rank) and AGAVaPS* (rank*) in terms of win/tie/loss. Win (+) indicates when the algorithm in question performed better than the proposed algorithm. Tie (=) occurs when the algorithm in question performed statistically equivalently to the proposed algorithm. Loss (-) indicates when the algorithm in question performed worse than the proposed algorithm. These comparisons were compiled into tables showing the average f-rank, final f-rank, and the overall win/tie/loss score for the test cases.

The feature selection and BNSL test were conducted with AGAVaPS and GA only, as the other methods are more suitable for numerical problems. The test used for the feature selection used the binary low-dimension (Low) and binary mid-dimension (Mid) datasets and followed the methodology used in Chiesa *et al.* (2020). The Low dataset is obtained by filtering and normalizing RNA-Seq data, containing 58 samples and 169 features. The Mid dataset is a characterization dataset of NMR spectrometry of metabolic profiles from urine, containing 106 samples and 701 features.

In this test, GA and AGAVaPS were run with a population size of 100 and a limit of 100 generations for the Low dataset and 150 generations for the Mid dataset. For both datasets, an individual represents a set of selected features, containing between 5 and 20 features. Mutation involved swapping one feature in the set for one that was not in the set. Reproduction was performed by combining the parent sets. For AGAVaPS, which generates only one offspring, it was generated by taking the intersection of the parent sets and randomly adding features from the parent sets. For GA, which generates two offspring, the algorithm selected an index for set separation and combined the opposite parts.

Both algorithms were run 100 times for each of the datasets. The mean, standard deviation, minimum, and maximum values were recorded for the number of selected features and the score. The score used was the same as in Chiesa *et al.* (2020). It employs Multi-Dimensional Scaling (MDS) considering the selected features and calculates the averaged Silhouette Index (aSI) for the coordinates obtained from MDS. The resulting score is aSI if $aSI > 0$ and 0 otherwise. Therefore, feature selection is a maximization problem where the best score is 1, and the worst is 0. Additionally, execution time was also recorded for comparison using mean and standard deviation.

For the BNSL test, AGAVaPS was compared to Hill Climbing (HC) and GA. HC is an optimization algorithm widely employed for BNSL. The HC implementation used is the one from pgmpy (ANKAN; PANDA, 2015). Pgmpy is a Python library designed for handling Probabilistic Graphical Models, including BNs (ANKAN; PANDA, 2015).

In this comparative analysis, we utilized three datasets sourced from the literature: Asia (LAURITZEN; SPIEGELHALTER, 1988), Coronary (REINIS *et al.*, 1981), and Sachs (SACHS *et al.*, 2005). These datasets are all accessible via the bnlearn package and are commonly employed in BNSL assessments (SCUTARI, 2010). Specifically, the Asia dataset comprises 8 variables, yielding a search space encompassing a staggering 7.84e+11 possible BN structures. In contrast, the Coronary dataset comprises 6 variables, resulting in a relatively more manageable search space of 3.78e+06 potential BN configurations. Lastly, the Sachs dataset involves 11 variables, leading to an extensive search space of approximately 3.16e+22 possible BNs. Across all datasets, both the AGAVaPS and GA algorithms were executed with a predefined limit of only 10,000 evaluations. In contrast, the HC method was executed as implemented in pgmpy, commencing from a random BN starting point.

Each of the algorithms was executed 100 times for each dataset, and the best BN discovered, along with its execution time, was recorded. These collected values were subsequently subjected to a comparative analysis, which included calculating the mean and standard deviation. Additionally, a Wilcoxon signed-rank test with a significance level of 0.05 was employed to assess the significance of the performance difference between the proposed algorithm and the comparison algorithms. To evaluate the degree of correspondence between the BNs and the datasets, the Bayesian Information Criterion (BIC) was employed as the scoring metric. The BIC is derived from the Schwarz Information Criterion and aims to minimize the conditional entropy of the variables while considering their parent nodes (CAMPOS, 2006). It's important to note that the BIC score is a maximization score that yields negative values.

After this, the multi-objective version of AGAVaPS was developed and further tested in BNSL. AGAVaPS underwent some changes to be able to deal with multi-objective optimization. The method to select the Pareto set was added. Also, the selection of the 10% best individuals was adapted to select all members of the Pareto set, and if there is still space left to select more individuals, the number of individuals is divided between the objectives, and that number of best individuals for that score are selected.

Then, the algorithm was further tested in BNSL. The first test made was a small-scale parameter analysis for AGAVaPS on BNSL was carried out. For that μ_{mut} and γ were varied and tested for different BNSL datasets. The number of evaluations used was 10,000 and the values evaluated for the parameters are shown in Table 4. AGAVaPS was run ten times for each combination of parameters and the mean BIC score of these ten runs was used to rank the parameters combinations and the combination with the best mean rank was the one used for the comparison test with other algorithms.

Table 4 – Parameter values used on the AGAVaPS parameter analysis for BNSL including the initial mutation mean (μ_{mut}) and reproduction rate (γ).

Parameter	**Values**
μ_{mut}	0.1, 0.3 and 0.5
γ	0.3 and 0.5

Source: Author.

Afterward, AGAVaPS was compared to HC and tabu search (TABU) on BNSL considering different datasets from the literature available at bnlearn (SCUTARI, 2010). The datasets and their characteristics can be seen in Table 5. HC and TABU were chosen because in Constantinou *et al.* (2021) 15 BNSL methods were compared and HC and TABU presented the best overall performance for the tests made. The algorithms were compared considering the F1 score, Structural Hamming Distance (SHD), Balanced Scoring Function (BSF), BIC score and execution time. The equations that define F1, SHD and BSF can be seen in Equations 3.10, 3.11, 3.12 and 3.13. These three metrics measure how close the expected structure and the structure found are. The best value for F1 is 1, for SHD is 0 and for BSF is 1. An explanation of what true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) mean for BNSL can be seen in Figure 27.

Table 5 – Datasets parameters including number of nodes (number of variables), number of edges, which is the number of edges of the expected structure, number of samples of the dataset, and the number of possible DAGs that the BN structure search space for the dataset has.

Dataset	**Number of nodes**	**Number of edges**	**Number of samples**	**Number of DAGs**
Alarm	37	46	20,000	3.01e+237
Asia	8	8	5,000	7.84e+11
Coronary	6	9	1,841	3.78e+06
Hailfinder	56	66	20,000	$\gg$2.11e+303
Insurance	27	52	20,000	1.90e+129
Sachs	11	17	100,000	3.16e+22
Sachs 2e4	11	17	20,000	3.16e+22

Source: Ribeiro and Maciel (2023)

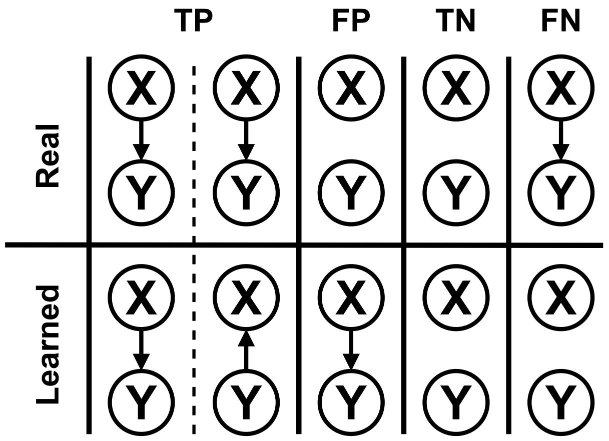
$$\text{F1 score} = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FP} + \text{FN}} \tag{3.10}$$

$$\text{SHD} = \text{FN} + \text{FP} \tag{3.11}$$

$$\text{BSF} = 0.5 \cdot \left(\frac{\text{TP}}{a} + \frac{\text{TN}}{i} - \frac{\text{FP}}{i} - \frac{\text{FN}}{a} \right) \tag{3.12}$$

$$i = \frac{|N|(|N|-1)}{2} - a \tag{3.13}$$

Figure 27 – Example of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) for BNSL used for calculating F1 score (Equation 3.10), SHD (Equation 3.11) and BSF (Equation 3.12) for the comparison of the performance of the search algorithms.



Source: Ribeiro and Maciel (2023)

Each algorithm was run thirty times for each dataset. The number of evaluations used for AGAVaPS and the max in-degree used by all algorithms can be seen in Table 6. Max in-degree is the limit of number of parents for a variable. The mean and standard variation were calculated. The algorithms were also ranked and the Wilcoxon signed-rank test with a significance level of 0.05 was used to determine which algorithms had the same performance, and thus had the same rank. The mean rank in all test cases was calculated and used for comparison.

Table 6 – Number of evaluations for the AGAVaPS for each dataset for the comparison with the other search algorithms (number of evaluations). The maximum in-degree used for each dataset for all algorithms (max in-degree) is also shown. Maximum in-degree is the limit of number of parents for a variable.

Dataset	**Number of evaluations**	**Max in-degree**
Alarm	1,000,000	4
Asia	10,000	4
Coronary	10,000	5
Hailfinder	1,000,000	4
Insurance	100,000	3
Sachs	10,000	3
Sachs 2e4	10,000	3

Source: Ribeiro and Maciel (2023)

3.4 Quantization and Bayesian Network

To better understand the BNSL search space and how the quantization of the variables affects this search space, a landscape analysis was carried out to better understand the characteristics of the BNSL search space. To achieve this, analyses were conducted on the scores of all BNs for problems with 3 to 6 variables, considering that for these numbers of variables, it was possible to evaluate all the search space. The number of DAGs for different numbers of variables can be seen in Table 7. The scores used for this analysis included K2, BDeu, and BIC, which were already implemented in the pgmpy library. The datasets used for analysis consisted of three simulated datasets, two datasets from the literature, and two real datasets from the UCI Machine Learning Repository[3]. The details about each dataset used can be seen in Table 8.

Table 7 – Number of DAGs for a given number of variables.

Number of variables	**Number of DAGs**
3	2,50e+**01**
4	5,43e+**02**
5	2,93e+**04**
6	3,78e+**06**
7	1,14e+**09**
8	7,84e+**11**
9	1,21e+**15**
10	4,18e+**18**
11	3,16e+**22**
12	5,22e+**26**
13	1,87e+**31**
14	1,44e+**36**
15	2,38e+**41**
16	8,38e+**46**
17	6,27e+**52**
18	9,94e+**58**
19	3,33e+**65**
20	2,34e+**72**
21	3,47e+**79**
22	1,08e+**87**
23	6,97e+**94**
24	9,44e+**102**
25	2,66e+**111**

Source: Author

[3] https://archive.ics.uci.edu/ml/index.php

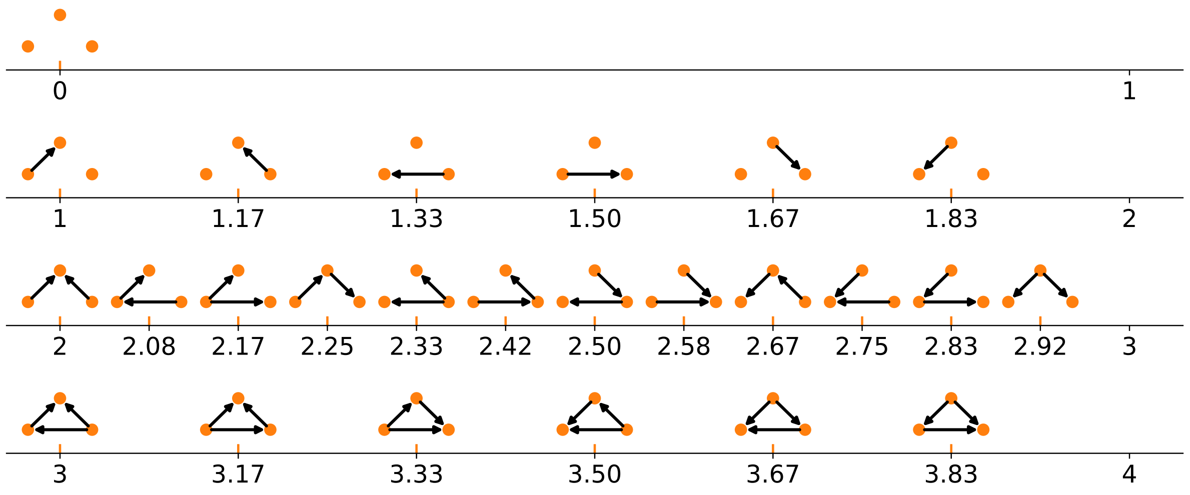
Table 8 – Datasets used for the landscape analysis.

Dataset	**Type**	**Number of variables**	**Expected Structure**
D3	Simulated	3	Figure 8
D4	Simulated	4	Figure 10
D5	Simulated	5	Figure 11
Lizards	Literature	3	Figure 19
Marks	Literature	6	Figure 20
Skin Segmentation (Skin)	Real	4	-
Haberman's Survival (Haberman)	Real	4	-

Source: Author

To visualize the search space, we implemented a technique to map BNs into numerical representations. To achieve this, all possible BNs were sorted by the number of edges, and numbers were assigned to each BN in an equidistant manner within the range $[number\ of\ edges, number\ of\ edges + 1)$. An example of this can be seen in Figure 28.

Figure 28 – Example of how the DAGs were mapped into numbers to perform the landscape analysis. This example considers a three-variable system to showcase the process, as it only has 15 possible DAGs and its visualization is easier. Each BN structure is mapped to a number considering its number of edges. The BNs were sorted by the number of edges, and numbers were assigned to each BN in an equidistant manner within the range $[number\ of\ edges, number\ of\ edges{+}1)$. The x-axis represents the numerical value attributed to a structure. Meaning that each BN structure is mapped to a numeric value on the x-axis. The structure is shown above its corresponding numeric value to exemplify this procedure.



Source: Ribeiro *et al.* (in press)

With this approach, an initial analysis of the search space was conducted for each score across all datasets. Initially, a common x-axis was used to assess how the same BNs were evaluated by different scores. Subsequently, the x-axis of each plot was arranged according to the respective score to observe the behavior of each score.

The next analysis aimed to determine whether the scores, despite their differing behaviors, agreed on which networks were the best. To achieve this, for each number of edges, it was determined how many scores identified the same structure as the best within that range.

Subsequently, the variation in the search space was evaluated following the variation in the data included in the dataset. To accomplish this, only the three simulated datasets were used. One hundred different batches of the datasets were generated, and the mean and standard deviation of the assessed scores for each BN were analyzed. Additionally, the changes in scores were examined concerning the volume of data in the dataset. To do this, the three simulated datasets were examined with sample sizes of 100, $1,000$, $10,000$, $100,000$, and $1,000,000$.

After these analyses, we started analyzing the effect of quantization on BNSL search space. The first analysis done was changing the quantization of variable A for the dataset D4, considering 2, 4, 6, 8, 10, 12, 14, 16, and 18 quantization levels for it (meaning that variable A could assume this number of states). Variable A has the ideal quantization of 10, because dataset D4 is a discrete dataset with added noise, thus having an ideal value of quantization (a specific number of states that the variable originally assumed). The search spaces for each of these cases were plotted together for analysis. The quantization was done using bins of equal width.

To further analyze how the variation of quantization affects the search space, the score of the expected structure was plotted for different values of quantization of its variables. Three types of variations were tested: varying one variable at a time, varying two variables together, and varying all three variables together. The variables that were not varied were fixed on the ideal quantization for the dataset D3 and on 200 for the dataset XYZ. The datasets used for this and the following analysis are described in Table 9.

Table 9 – Datasets used for the landscape analysis.

Dataset	**Type**	**Number of variables**	**Expected Structure**	**Variables' Distribution**
D3	Simulated discrete	3	Figure 8	Figure 7
D4	Simulated discrete	4	Figure 10	Figure 9
XYZ	Simulated continuous	3	Figure 13	Figure 12
XYZ3	Simulated continuous	3	Figure 13	Figure 14
2T	Simulated continuous	2	Figure 17	Figure 16

Source: Author

The results were interesting and to better understand them, the variation of Y for XYZ was repeated plotting each component of the score to understand how each component affected the final score value and its behavior with the varying quantization. Considering the results a proposal of method of choosing a quantization of variables was made. The proposal was to choose the quantization on the peak of the curve as the quantization of the variable and repeat the same for the other variables, varying the structure. An example of this method and its problems are shown in the Results and Discussion (Chapter 4).

After that, different proposals using BNSL to choose a quantization for its variables were tested. The base idea was to have different quantizations and bins or cut-point data inserted as variables and use BNSL. The resulting structure would give the quantization or cut-points to be used. For the BNSL in this test, the exhaustive search and blip (Bayesian network Learning Improved Project)[4] (SCANAGATTA *et al.*, 2018a) were used. Exhaustive search could only be used for a small number of variables, while blip is a BNSL algorithm developed to deal with a high number of variables (SCANAGATTA *et al.*, 2018a). These proposals were tested for XYZ.

Another approach tried out was using Expectation Maximization (EM) coupled with a structural score that penalized the zero count on the CPT of variables. Thus giving a better score when the CPT was better filled. With that in mind, a variation of the BIC score was made, where the penalization was done considering the number of empty spaces on the CPT. The score should also enable the estimator to be changed so that the EM can check if removing some points from the data could improve the score. The formulation is

$$MDLNonZero(G:D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\max_{\theta_{ij}}\left[\sum_{k=1}^{r_i} N_{ijk}\log(\hat{\theta}_{ijk}(G)) - c\cdot n\_non\_zeros(ij,\theta_{ij})\right] \tag{3.14}$$

where n is the number of variables; r_i is the number of states of the variable X_i; q_i is the number of possible configurations of the parent set $Pa_G(X_i)$ of X_i, where $q_i = \prod_{X_j\in Pa_G(X_i)} r_j$; w_{ij}, $j = 1,\ldots,q_i$ represents a configuration of $Pa_G(X_i)$; and where

$$\hat{\theta}_{ijk}(G) = \frac{\hat{N}_{ijk}+\varepsilon}{\hat{N}_{ij}+\varepsilon r_i} \tag{3.15}$$

where, $\hat{N}_{ijk}$ is the estimator of N_{ijk} which the number of instances in the data set D where the variables X_i take the value x_{ik} and the set of variables $Pa_G(X_i)$ take the value w_{ij} and $\hat{N}_{ij}$ is the estimator of N_{ij} which is the number of instances in the data set where the variables in $Pa_G(X_i)$ take their configuration w_{ij}, where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. For both estimators, they can either assume the true value of N_{ijk} and N_{ij} or zero. This way, outliers values that make the score worse, as their contribution to the score is smaller than the penalty they bring, can be disregarded by the score. ε is a very small value used to avoid having $\hat{\theta}_{ijk}(G) = 0$ which would make the log value be undefined. This score was tested and a sensitivity test to the parameters was done using the datasets XYZ and XYZ3.

[4] https://github.com/mauro-idsia/blip

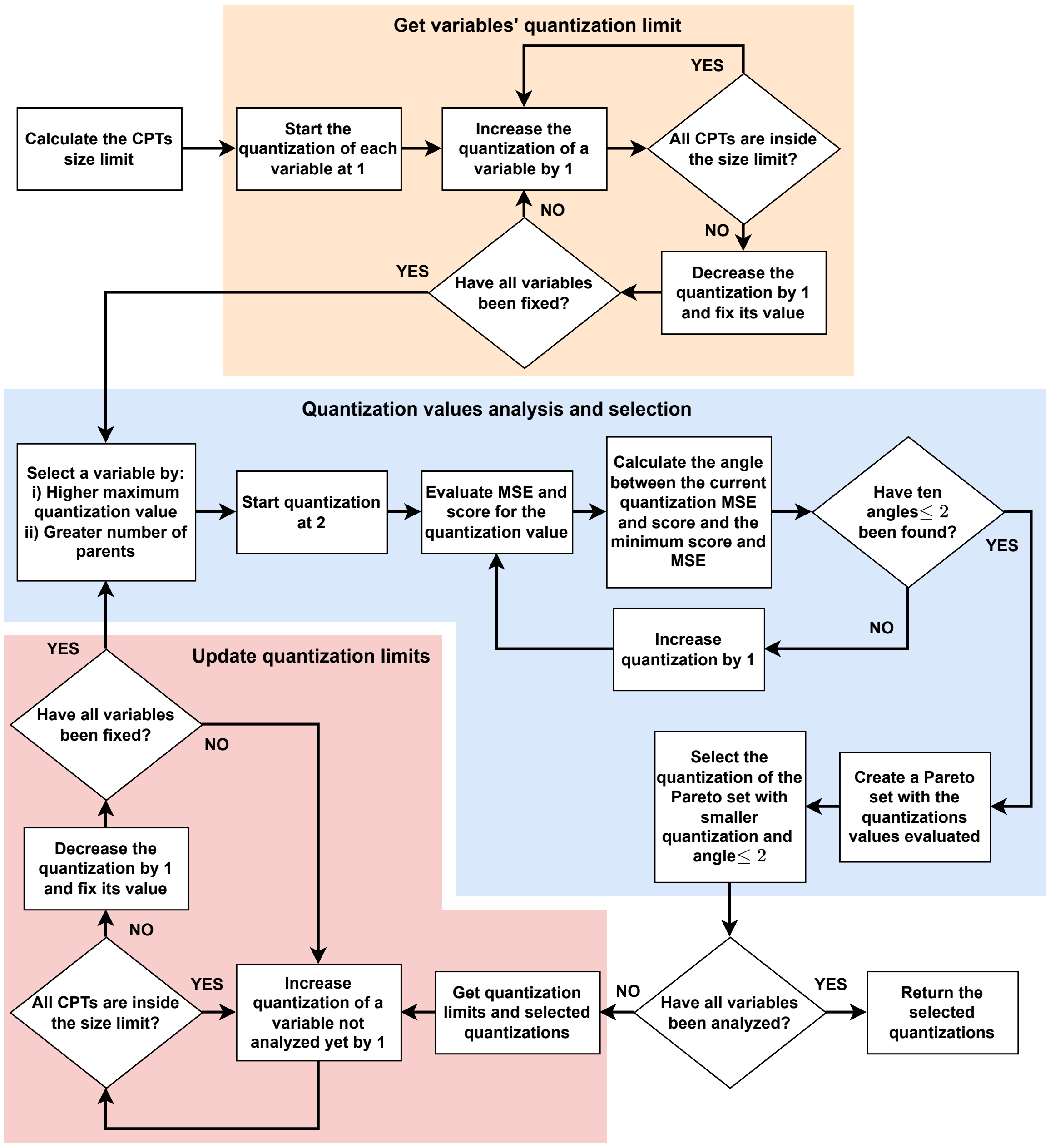
Considering the EM, after some concept work and first tests the idea that seemed most promising was doing the EM using BN learned from the discretized data, then adding connections from each variable to its raw data variable. Then the discretized data is erased for prediction. For the prediction, for each point, new CPTs are created for the raw data variables by getting the density of a normal distribution modeled on the current points included in each bin. This means, that it gets a probability of the point being predicted to belong to each bin and that information is used to build the CPT of the raw variables and do the prediction of the new discretized data. This method was tested for the XYZ and 2T datasets, plotting initial and final modeled normal distributions for each bin.

While testing and further developing this method, we had an idea for another approach. This new approach was named CPT Limit-Based Quantization (CLBQ). A flowchart detailing the method and giving an overview of the method can be seen in Figure 29. On CLBQ, a maximum size for the CPTs is determined considering the dataset size. This is done by the following equation

$$\mathcal{M} = \frac{\text{Number of samples}}{\eta} \tag{3.16}$$

where η is the desired number of samples for each element of the CPT and $\mathcal{M}$ is the maximum CPT size. This limit is put in place to guarantee model quality considering the data volume you have at your disposal, however, most of the time the quantizations chosen by CLBQ allow for a higher model quality than the one instantiated by this limit. Considering a uniform distribution, which would be the worst-case scenario, the value of $\eta = 3$ was chosen to be used on the method.

Figure 29 – Flowchart detailing CLBQ and giving an overview of the step it takes to quantize data given a BN structure. All quantization values refer to the number of states a variable can assume.


Get variables' quantization limit
Calculate the CPTs size limit
Start the quantization of each variable at 1
Increase the quantization of a variable by 1
All CPTs are inside the size limit?
YES
NO
Decrease the quantization by 1 and fix its value
Have all variables been fixed?
YES
NO
Quantization values analysis and selection
Select a variable by: i) Higher maximum quantization value ii) Greater number of parents
Start quantization at 2
Evaluate MSE and score for the quantization value
Calculate the angle between the current quantization MSE and score and the minimum score and MSE
Have ten angles≤ 2 been found?
YES
NO
Increase quantization by 1
Create a Pareto set with the quantizations values evaluated
Select the quantization of the Pareto set with smaller quantization and angle≤ 2
Have all variables been analyzed?
YES
NO
Return the selected quantizations
Get quantization limits and selected quantizations
Update quantization limits
Increase quantization of a variable not analyzed yet by 1
All CPTs are inside the size limit?
YES
NO
Decrease the quantization by 1 and fix its value
Have all variables been fixed?
YES
NO

Source: Ribeiro *et al.* (in press)

Then, with the CPT limit defined and given a BN structure, the maximum quantization of each variable is calculated using a heuristic method. In this method, each variable quantization is initialized in 1, and one by one they are increased by 1. After each increase, all the CPT sizes are calculated and checked if their size is still smaller than the size limit. If a CPT surpassed the size limit, the variable that was increased is has their quantization value fixed on the value prior to the increase. The CPT size is calculated by multiplying the quantization of the variable and its parents' quantizations (ROHMER, 2020). This process is stopped when all variables have their quantization value fixed on their limit value.

The next step of the the process is to analyze the quantization values between 2 and the quantization limit and select a quantization that presents a trade-off between the mean squared error (MSE) between the original values and the quantized values, and the BN structure score. For this analysis, one variable is analyzed at each time. The variables are selected according to the following priority list: i) higher maximum quantization value; ii) greater number of parents. The selected variables the quantization values are used to evaluate the BN score and the MSE beginning from $q_v = 2$. For each quantization value an angle θ_q is also calculated as follows.

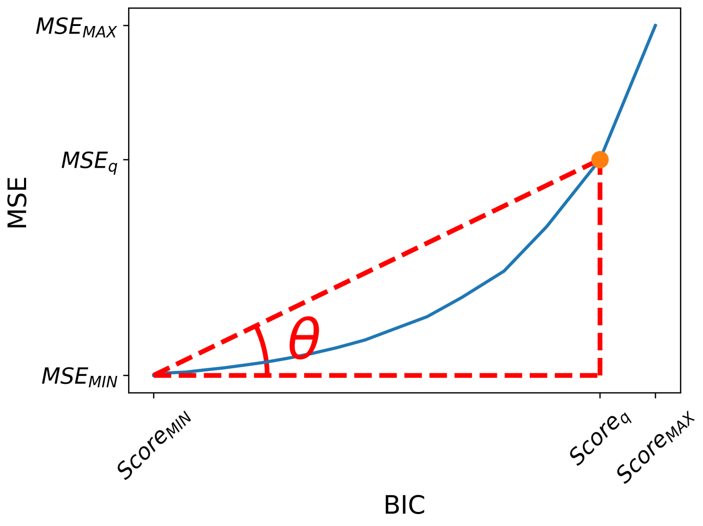
$$\theta_q = \arctan\left(\frac{\Delta\text{MSE}_q}{\Delta\text{Score}_q}\right) \tag{3.17}$$

$$\Delta\text{MSE}_q = \frac{|\text{MSE}_{MIN} - \text{MSE}_q|}{|\text{MSE}_{MIN} - \text{MSE}_{MAX}|} \tag{3.18}$$

$$\Delta\text{Score}_q = \frac{|\text{Score}_{MIN} - \text{Score}_q|}{|\text{Score}_{MIN} - \text{Score}_{MAX}|} \tag{3.19}$$

For the previous equations the values of MSE_{MIN} and Score_{MIN} are obtained by using the quantization limit value of the variable. MSE_{MAX} and Score_{MAX} use $q_v = 2$. The interpretation of the angle θ_q can be seen in Figure 30.

Figure 30 – Example of what θ_q represents. BIC is the BN structure score used. The blue curve is the values of MSE and BIC score for all the quantization values of the variable under analysis. The quantization value under analysis is represented in orange.



Source: Ribeiro *et al.* (in press)

This analysis of increasing values of q_v is stopped after ten q_v values had $\theta \leq 2$. A Pareto set is created from the evaluated quantization values and the smallest quantization from the Pareto set with $\theta \leq 2$ is selected to be used as the variable's quantization value.

When a quantization value is selected, the quantization value of that variable is fixed on it and the maximum quantization value of the variables that have not been analyzed yet is updated considering the now fixed value. For the score and MSE evaluation, the quantization of the variables that are not under analysis is either on their fixed value already selected or on the maximum quantization value found.

CLBQ was tested using four different datasets. The four datasets are described in Table 10. The first test made was a functionality test where a step-by-step analysis of CLBQ was done for it quantizing the variables of the D4 dataset considering its expected structure.

Table 10 – Datasets used for the CLBQ test and analysis.

Dataset	**Type**	**Number of variables**	**Expected Structure**	**Variables' Distribution**
D4	Simulated discrete	4	Figure 10	Figure 9
XYZ	Simulated continuous	3	Figure 13	Figure 12
XYZ3	Simulated continuous	3	Figure 13	Figure 14
Weather	Real	4	-	Figure 18

Source: Author

The second test done was a comparison between CLBQ and Dynamic Discretization (DD) algorithm proposed in Ciunkiewicz *et al.* (2022) for the D4, XYZ, XYZ3, and Weather datasets. To compare the quantization chosen by CLBQ and DD, the quantization histogram was plotted against the distribution of the variables, and the MSE of the quantizations was compared.

In addition to that, a landscape analysis for BNSL using CLBQ was also done, to check if the use of CLBQ in BNSL would keep the expected structure as one of the best structures, such that it could be found by a search algorithm performing the BNSL while using CLBQ to quantize the variables. An analysis of the structures with equal or higher score than the score of the expected structure was also performed. In this analysis, the structure, the quantization chosen by CLBQ, the score considering this chosen quantization, and the score quantization considering the quantization selected for the expected structure were analyzed.

In addition to that, an initial execution time analysis of CLBQ was done measuring the mean and standard deviation (STD) of the execution time of CLBQ for each structure when making the landscape analysis.

3.5 Score Function

The initial idea for the development of this score was to bring the predictability quality of BN to contrast with a classic score that analyzes the congruence between structure and data. With this in mind, three initial scoring proposals called P1, P2, and P3 were made. P1 is based on the prediction power (PP) measure which is calculated by the equation:

$$PP = 1 - e^{-D} \tag{3.20}$$

where D is the relative entropy between the real values of the system and the prediction error of the forecast made by the model (in our case, the model is the BN under test) (SCALASSARA; MACIEL; PEREIRA, 2009). The relative entropy is also called Kullback–Leibler divergence and can be calculated by

$$D = \sum_{x \in X} P(x) \log \left(\frac{P(x)}{Q(x)} \right) \tag{3.21}$$

where P and Q are discrete probability distributions defined on the same sample space X. The calculation is done by averaging the PP for several tests of the same model.

The P2 score is based on the difference between the distributions of the prediction and the actual values. To do this, it calculates the average of the absolute differences between the histograms of the prediction and the actual values. Both histograms use the same bins.

Finally, the P3 score is based on the normalized mean error and is calculated using the equation:

$$P3 = \frac{\sum\limits_{i=1}^{N} \frac{|value_i - forecast_i|}{max(value_i) - min(value_i)}}{N \cdot size(value_i)} \tag{3.22}$$

where N is the number of outputs the system has, $value_i$ is the actual value for output i, and $forecast_i$ is the predicted value for output i.

A preliminary test was conducted using two synthetic systems described by the summation of functions and employing the Hill Climbing learning method. In this test, the three proposed scores were compared with each other and with the K2 and BDeu scores, considering the average number of edges and the prediction accuracy rate of the outputs.

The two considered systems are composed of three output signals and one input, which represents time. In the first system, the sum of a sine wave (signal A) and a square wave (signal B) was used to generate signal C. The equations for these signals are as follows:

$$A = 2 \cdot \sin\left(\frac{6 \cdot \pi}{50} \cdot t\right) \tag{3.23}$$

$$B = \begin{cases} 0, & \text{if } 100k \leq t < 50(2k+1) \text{ for } k \in \mathbb{N}_0 \\ 50, & \text{otherwise} \end{cases} \tag{3.24}$$

$$C = A + B + U(0, 2) \tag{3.25}$$

where $U(0, 2)$ represents uniform noise ranging from 0 to 2.

The second system consists of two sine waves (signal A and B) and their sum (signal C). Here are the equations:

$$A = \sin\left(\frac{3 \cdot \pi}{100} \cdot t\right) \tag{3.26}$$

$$B = \sin\left(\frac{\pi^2}{100} \cdot t\right) \tag{3.27}$$

$$C = A + B + U(0, 0.2) \tag{3.28}$$

where $U(0, 0.2)$ represents uniform noise ranging from 0 to 0.2.

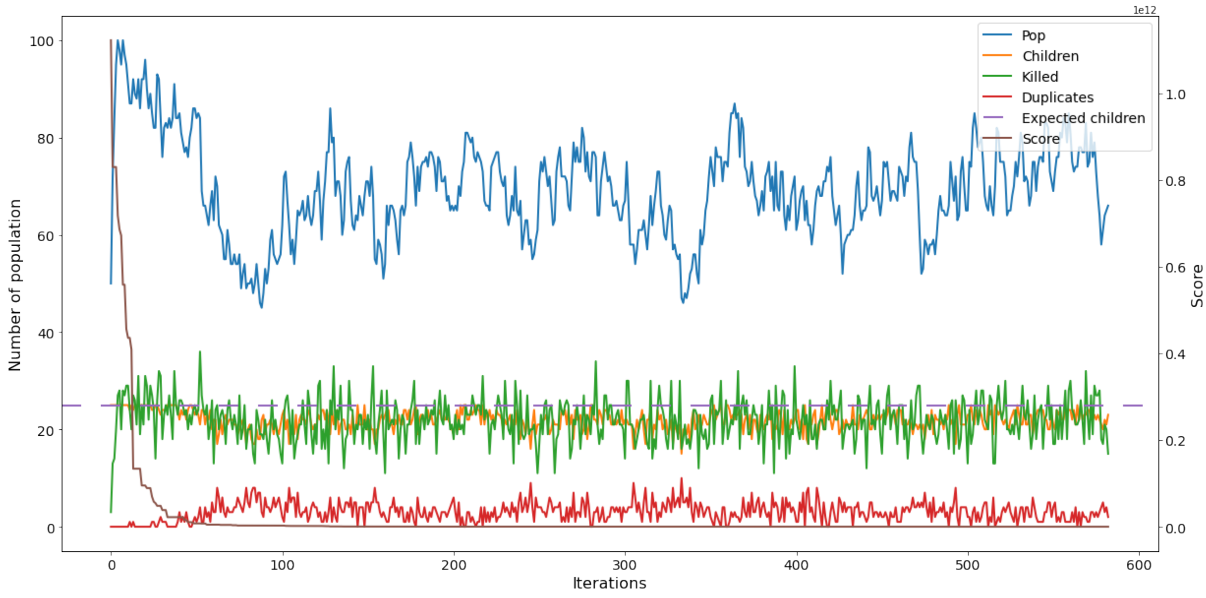
4 RESULTS AND DISCUSSION

In this chapter, the results of the tests made are shown following the order on which they were presented in the previous chapter.

4.1 AGAVaPS Behavior Test

The result of the AGAVaPS behavior test can be seen in Figure 31. As was expected the population size increases greatly for the first iterations, until the number of removed individuals reaches the number of individuals added to the population. This is a result of the sampling of the life parameter happening with a mean value of five. After this point, the population size reaches a kind of stability while AGAVaPS explores the search space and has not stagnated the search in some regions.

Figure 31 – Evolution of different values during execution of AGAVaPS for the $f1$ test function from the CEC2017 benchmark. The values are the population size (Pop), number of new individuals added to the population (Children), number of new individuals that were not added to the population because they were duplicates of solutions already in the population (Duplicates), number of individuals removed from the population (Killed), the expected number of new individuals to be added to the population (Expected children) and the best score (Score).



Source: Ribeiro and Maciel (2022)

After the stagnation happens, the population size starts to oscillate as a result of the μ_{mut} update strategy that changes its value to the opposite size when an extreme value is reached. This results in AGAVaPS oscillating between exploitation and exploration at this point of the search. When in the exploration phase, the population size increases, as the exploration of new regions increases the diversity of the population and the generation of duplicates is decreased. Meanwhile, in the exploitation phase, the population size is decreased by the increase in the duplicate generation that results from exploring the same regions and having a population that is composed of very similar solutions.

4.2 AGAVaPS CEC2017 Benchmark Test

The results of the comparison of AGAVaPS with other algorithms for the CEC2017 benchmark can be seen in Tables 11 and 12. In Table 11 the win/tie/loss results were aggregated. In Table 12 the average and overall ranks are shown. Win is when an algorithm performs better than the proposed algorithm. Tie, is when their performance was equivalent and loss is when the algorithm had a worse performance than the proposed algorithm. Free refers to the test cases with a free initial population, limited refers to the test where the initial population was limited to a small part of the search space and both is the sum of the free and limited cases.

Table 11 – Aggregated win/tie/loss for the comparison of AGAVaPS with other algorithms for the CEC2017 benchmark. “rank” is the comparison with AGAVaPS with *tournament size* = 3 and “rank*” is the comparison with AGAVaPS with *tournament size* = 1. Win is when an algorithm performs better than the proposed algorithm. Tie, is when their performance was equivalent and loss is when the algorithm had a worse performance than the proposed algorithm. Free refers to the test cases with a free initial population, limited refers to the test where the initial population was limited to a small part of the search space and both is the sum of the free and limited cases.

			GAVaPS	**GA**	**PSO**	**FFA**	**BAT**	**CS**
Grid	Free	rank	30/00/00	30/00/00	02/02/26	00/05/25	00/00/30	00/00/30
		rank*	30/00/00	30/00/00	00/01/29	00/00/30	00/00/30	00/00/30
	Limited	rank	30/00/00	30/00/00	00/00/30	00/05/25	00/00/30	00/00/30
		rank*	30/00/00	30/00/00	00/00/30	00/00/30	00/00/30	00/00/30
	Both	rank	60/00/00	60/00/00	02/02/56	00/10/50	00/00/60	00/00/60
		rank*	60/00/00	60/00/00	00/01/59	00/00/60	00/00/60	00/00/60
Best	Free	rank	03/00/27	07/02/21	12/00/18	15/02/13	01/00/29	21/01/08
		rank*	01/00/29	03/10/17	13/00/17	17/03/10	01/00/29	23/01/06
	Limited	rank	00/00/30	14/01/15	03/00/27	16/03/11	00/01/29	16/01/13
		rank*	00/00/30	09/08/13	03/00/27	16/01/13	00/01/29	16/00/14
	Both	rank	03/00/57	21/03/36	15/00/45	31/05/24	01/01/58	37/02/21
		rank*	01/00/59	12/18/30	16/00/44	33/04/23	01/01/58	39/01/20
Time	Free	rank	00/00/30	00/00/30	00/00/30	00/00/30	00/00/30	00/01/29
		rank*	00/00/30	28/00/02	01/00/29	00/00/30	18/02/10	10/01/19
	Limited	rank	00/00/30	00/00/30	00/00/30	00/00/30	00/00/30	00/01/29
		rank*	00/00/30	26/01/03	01/00/29	00/00/30	09/04/17	07/00/23
	Both	rank	00/00/60	00/00/60	00/00/60	00/00/60	00/00/60	00/02/58
		rank*	00/00/60	54/01/05	02/00/58	00/00/60	27/06/27	17/01/42

Source: Ribeiro and Maciel (2022)

Table 12 – Mean and overall rank analysis for the comparison of AGAVaPS with other algorithms for the CEC2017 benchmark. "AGAVaPS" refers to AGAVaPS with *tournament size* = 3 and "AGAVaPS*" refers to with AGAVaPS with *tournament size* = 1. Free refers to the test cases with a free initial population, limited refers to the test where the initial population was limited to a small part of the search space and both is the sum of the free and limited cases.

			AGAVaPS	**AGAVaPS***	**GAVaPS**	**GA**	**PSO**	**FFA**	**BAT**	**CS**
Grid	Free	Average	4.03	3.00	1.00	2.00	5.73	4.87	7.70	6.57
		Overall	4	3	1	2	6	5	8	7
	Limited	Average	4.00	3.00	1.00	2.00	6.37	4.83	7.70	6.17
		Overall	4	3	1	2	6	5	7	6
	Both	Average	4.02	3.00	1.00	2.00	6.05	4.85	7.70	6.37
		Overall	4	3	1	2	6	5	8	7
Best	Free	Average	2.93	3.37	5.77	3.87	4.03	2.80	6.53	1.67
		Overall	2	3	6	4	5	2	7	1
	Limited	Average	2.87	2.83	6.67	3.00	5.77	2.67	5.80	2.53
		Overall	1	1	3	1	2	1	2	1
	Both	Average	2.90	3.10	6.22	3.43	4.90	2.73	6.17	2.10
		Overall	2	2	5	3	4	2	5	1
Time	Free	Average	1.00	3.83	7.77	2.17	6.03	6.40	3.40	4.33
		Overall	1	4	7	2	6	6	3	5
	Limited	Average	1.00	3.37	7.70	2.17	5.97	6.33	3.53	4.37
		Overall	1	3	6	2	5	5	3	4
	Both	Average	1.00	3.60	7.73	2.17	6.00	6.37	3.47	4.35
		Overall	1	3	7	2	5	6	3	4

Source: Ribeiro and Maciel (2022)

These results show that AGAVaPS performed better than four algorithms out of six when considering grid coverage, only losing to GAVaPS and GA. AGAVaPS, GAVaPS, and GA had a better performance on the search space coverage than other algorithms, when considering limited initial space, showing that these algorithms are capable of overcoming poor starting conditions.

AGAVaPS also had a good performance when considering the best solution found, having tied on the best performance for the test cases with limited initial space. Moreover, when considering execution time, AGAVaPS was the fastest algorithm for all the test cases.

4.3 AGAVaPS Feature Selection Test

The measurements for the score values and execution time can be seen in Table 13. AGAVaPS has a higher mean, maximum, and minimum score than the GA. Thus, showing that AGAVaPS had a better performance on both test cases in feature selection.

Table 13 – Results of the feature selection test. “Low” refers to the binary low-dimension dataset and “Mid” refers to the binary mid-dimension dataset.

		Low		Mid	
		AGAVaPS	GA	AGAVaPS	GA
Score	mean	**0.2174**	0.1471	**0.3755**	0.2326
	std	0.0076	0.0080	0.0442	0.0364
	min	0.2004	0.1287	0.2947	0.1702
	max	0.2312	0.1739	0.4381	0.3773
	p-value	3.90e-18		3.90e-18	
Time	mean	**39.70**	68.07	**76.85**	158.16
	std	12.82	19.03	8.48	21.09
	p-value	9.77e-14		3.90e-18	

Source: Ribeiro and Maciel (2022)

When considering the execution time, AGAVaPS was faster than GA for all the test cases. The p-value of the Wilcoxon rank-sum test shows that the performance of both algorithms are different.

4.4 AGAVaPS BNSL Test

The results of the BNSL test can be seen in Table 14. AGAVaPS performed better than HC and GA when considering the score, having a higher mean score and a very small variance. When considering the execution time, HC was the best-performing algorithm, followed by GA and AGAVaPS. The performance and consistency of the AGAVaPS were remarkable and suggested that the AGAVaPS could be suitable to be used for BNSL.

Table 14 – Results of the BNSL initial test using AGAVaPS, GA and HC.

Asia		AGAVaPS	GA	HC
Score	mean	**-11,107.29**	-11,126.50	-11,130.00
	std	**3.64e-12**	12.06	17.59
	p-value		9.61e-18	3.83e-18
Time	mean	78.40	18.73	0.49
	std	4.91	0.23	0.09
	p-value		3.90e-18	3.90e-18
Coronary		AGAVaPS	GA	HC
Score	mean	**-6,717.30**	-6,718.24	-6,720.31
	std	**0.18**	0.54	2.72
	p-value		2.10e-17	5.34e-17
Time	mean	78.26	8.59	0.21
	std	4.25	0.11	0.03
	p-value		3.90e-18	3.90e-18
Sachs		AGAVaPS	GA	HC
Score	mean	**-671,029.96**	-672,021.98	-672,101.88
	std	**98.93**	318.82	1,545.56
	p-value		3.89e-18	5.91e-15
Time	mean	156.47	871.73	22.50
	std	4.32	16.76	2.39
	p-value		3.90e-18	3.90e-18

Source: Ribeiro and Maciel (2022)

4.5 AGAVaPS Parameter Analysis for BNSL

The results of the AGAVaPS parameter analysis can be seen in Tables 15 and 16. Table 15 shows the mean values of the BIC score for each parameter combination tested for each dataset. This mean score is obtained considering the highest score found for each run. Table 16 shows the ranks of the parameter combinations for the results of the previous table. From these tables, it can be seen that for many cases the differences between combinations were minimal. The parameters with the best performance were $\gamma = 0.5$ and $\mu_{mut} = 0.5$. This parameter combination was used for the comparison test.

Table 15 – Results of the parameter analysis made for AGAVaPS considering BNSL. The mean value of the BIC score is shown for each combination of parameters. γ is the reproduction rate and μ_{mut} is the initial mutation mean. The best values for each dataset are marked in bold.

Dataset	$\gamma = 0.3$ $\mu_{mut} = 0.1$	$\gamma = 0.3$ $\mu_{mut} = 0.3$	$\gamma = 0.3$ $\mu_{mut} = 0.5$	$\gamma = 0.5$ $\mu_{mut} = 0.1$	$\gamma = 0.5$ $\mu_{mut} = 0.3$	$\gamma = 0.5$ $\mu_{mut} = 0.5$
Alarm	-237,238.73	-236,307.90	-236,341.33	-236,970.01	-237,319.48	**-236,152.09**
Asia	-11,113.07	-11,113.09	-11,112.20	-11,113.08	-11,112.26	**-11,111.43**
Coronary	-6,717.78	-6,718.03	-6,718.03	-6,717.71	-6,718.35	**-6,717.65**
Hailfinder	-1,058,908.27	-1,056,481.45	-1,056,888.71	-1,055,479.40	-1,055,095.71	**-1,053,574.88**
Insurance	-279,586.47	-279,470.58	**-278,965.12**	-279,035.66	-279,547.27	-279,198.98
Sachs	**-719,733.24**	-720,224.68	-720,133.82	-720,506.56	-720,161.80	-720,588.25

Source: Ribeiro and Maciel (2023)

Table 16 – Results of the parameter analysis made for AGAVaPS considering BNSL. The rank of each parameter combination is shown for each test case. The mean rank is also shown on the line "Mean".

Dataset	$\gamma = 0.3$ $\mu_{mut} = 0.1$	$\gamma = 0.3$ $\mu_{mut} = 0.3$	$\gamma = 0.3$ $\mu_{mut} = 0.5$	$\gamma = 0.5$ $\mu_{mut} = 0.1$	$\gamma = 0.5$ $\mu_{mut} = 0.3$	$\gamma = 0.5$ $\mu_{mut} = 0.5$
Alarm	5	2	3	4	6	1
Asia	4	6	2	5	3	1
Coronary	3	5	4	2	6	1
Hailfinder	6	4	5	3	2	1
Insurance	6	4	1	2	5	3
Sachs	1	4	2	5	3	6
Mean	4.17	4.17	2.83	3.50	4.17	**2.17**

Source: Ribeiro and Maciel (2023)

4.6 Comparison of AGAVaPS, TABU and HC on BNSL

The results of the comparison of AGAVaPS, TABU and HC on BNSL can be seen in Table 17 to 21. Tables 17, 18 and 19 show the mean, standard deviation and rank of the F1, SHD and BSF metrics. Table 20 shows the same values for the BIC score and Table 21 shows the same for the execution time.

Table 17 – Results of the comparison of AGAVaPS, TABU and HC on BNSL. The values of F1 (mean (std)) (Equantion 3.10) are shown for each dataset and algorithm in the upper part of the table. The bottom part of the table shows the ranks of each algorithm for each test case.

Datasets	**HC F1**	**TABU F1**	**AGAVaPS F1**
Alarm	0.744 (0.049)	0.732 (0.054)	0.776 (0.026)
Asia	0.655 (0.135)	0.732 (0.118)	0.899 (0.032)
Coronary	0.892 (0.060)	0.879 (0.060)	0.859 (0.051)
Hailfinder	0.554 (0.053)	0.560 (0.043)	0.554 (0.042)
Insurance	0.683 (0.064)	0.661 (0.072)	0.659 (0.023)
Sachs	0.823 (0.099)	0.836 (0.120)	0.926 (0.021)
Sachs 2e4	0.843 (0.112)	0.850 (0.103)	0.932 (0.022)
Alarm rank	2	2	1
Asia rank	2	2	1
Coronary rank	1	1	1
Hailfinder rank	1	1	1
Insurance rank	1	1	1
Sachs rank	2	2	1
Sachs 2e4 rank	2	2	1
Mean Rank	1.57	1.57	**1.00**

Source: Ribeiro and Maciel (2023)

From the F1 score results it can be seen that the algorithms were tied for half of the test cases. In addition to that, AGAVaPS had a better performance than HC and TABU for all the test cases that were not a full tie. Furthermore, HC and TABU had the same performance for all the test cases, thus the improvement from using a tabu list was not observed.

Table 18 – Results of the comparison of AGAVaPS, TABU and HC on BNSL. The values of SHD (mean (std)) (Equation 3.11) are shown for each dataset and algorithm in the upper part of the table. The bottom part of the table shows the ranks of each algorithm for each test case.

Datasets	**HC SHD**	**TABU SHD**	**AGAVaPS SHD**
Alarm	28.367 (6.253)	30.033 (7.418)	18.467 (2.045)
Asia	6.500 (2.872)	4.933 (2.421)	1.533 (0.499)
Coronary	1.867 (0.991)	2.100 (0.978)	2.400 (0.841)
Hailfinder	68.900 (10.543)	67.867 (8.671)	45.700 (2.923)
Insurance	34.167 (7.572)	36.967 (8.503)	30.533 (1.688)
Sachs	6.500 (3.704)	6.067 (4.546)	2.533 (0.763)
Sachs 2e4	5.700 (4.157)	5.467 (3.845)	2.367 (0.752)
Alarm rank	2	2	1
Asia rank	2	2	1
Coronary rank	1	1	1
Hailfinder rank	2	2	1
Insurance rank	2	2	1
Sachs rank	2	2	1
Sachs 2e4 rank	2	2	1
Mean Rank	1.86	1.86	**1.00**

Source: Ribeiro and Maciel (2023)

From the SHD metric results, it can be seen that AGAVaPS was the best-performing algorithm, being the first-ranked algorithm for all test cases. Moreover, once more HC and TABU were tied for all test cases.

Table 19 – Results of the comparison of AGAVaPS, TABU and HC on BNSL. The values of BSF (mean (std)) (Equation 3.12) are shown for each dataset and algorithm in the upper part of the table. The bottom part of the table shows the ranks of each algorithm for each test case.

Datasets	**HC BSF**	**TABU BSF**	**AGAVaPS BSF**
Alarm	0.852 (0.045)	0.839 (0.040)	0.728 (0.018)
Asia	0.520 (0.203)	0.636 (0.162)	0.836 (0.045)
Coronary	0.748 (0.121)	0.719 (0.118)	0.728 (0.088)
Hailfinder	0.613 (0.053)	0.622 (0.040)	0.445 (0.055)
Insurance	0.642 (0.072)	0.621 (0.081)	0.563 (0.024)
Sachs	0.760 (0.144)	0.776 (0.175)	0.899 (0.030)
Sachs 2e4	0.785 (0.162)	0.797 (0.149)	0.911 (0.038)
Alarm rank	1	1	2
Asia rank	2	2	1
Coronary rank	1	1	1
Hailfinder rank	1	1	2
Insurance rank	1	1	2
Sachs rank	2	2	1
Sachs 2e4 rank	2	2	1
Mean Rank	**1.43**	**1.43**	**1.43**

Source: Ribeiro and Maciel (2023)

From the BSF metric results, it can be seen that all algorithms were tied when considering the overall mean rank. Once more, HC and TABU were tied for all the test cases. AGAVaPS was the best-performing algorithm for three test cases (Asia, Sachs and Sachs 2e4) and tied with HC and TABU for Coronary. These results indicate that AGAVaPS performed better on datasets with fewer nodes when considering BSF.

Table 20 – Results of the comparison of AGAVaPS, TABU and HC on BNSL. The values of the BIC score (mean (std)) are shown for each dataset and algorithm in the upper part of the table. The bottom part of the table shows the ranks of each algorithm for each test case.

Datasets	**HC Score**	**TABU Score**	**AGAVaPS Score**
Alarm	-2.232759e+05 (1.8e+03)	-2.230440e+05 (1.7e+03)	-2.275161e+05 (1.1e+03)
Asia	-1.116352e+04 (1.4e+02)	-1.113056e+04 (1.7e+01)	-1.111352e+04 (4.7e+00)
Coronary	-6.719031e+03 (1.4e+00)	-6.719848e+03 (2.3e+00)	-6.717676e+03 (6.9e-01)
Hailfinder	-1.002213e+06 (6.5e+03)	-1.001578e+06 (4.9e+03)	-1.043391e+06 (6.7e+03)
Insurance	-2.735643e+05 (4.4e+03)	-2.749877e+05 (5.9e+03)	-2.744720e+05 (1.1e+03)
Sachs	-7.299457e+05 (9.5e+03)	-7.302275e+05 (1.1e+04)	-7.201837e+05 (1.4e+03)
Sachs 2e4	-1.466913e+05 (2.2e+03)	-1.465159e+05 (1.9e+03)	-1.447779e+05 (2.8e+02)
Alarm rank	1	1	2
Asia rank	2	2	1
Coronary rank	2	2	1
Hailfinder rank	1	1	2
Insurance rank	1	1	1
Sachs rank	2	2	1
Sachs 2e4 rank	2	2	1
Mean Rank	1.57	1.57	**1.28**

Source: Ribeiro and Maciel (2023)

From the BIC scores results, it can be seen that AGAVaPS was the best-performing algorithm for four test cases (Asia, Coronary, Sachs and Sachs 2e4) and tied with HC and TABU for Insurance. AGAVaPS performed better for datasets with 27 variables or less when considering the BIC score. This may be a result of AGAVaPS not evaluating enough structures. Again, HC and TABU had the same performance for all test cases.

Table 21 – Results of the comparison of AGAVaPS, TABU and HC on BNSL. The values of the execution time (mean (std)) are shown for each dataset and algorithm in the upper part of the table. The bottom part of the table shows the ranks of each algorithm for each test case.

Datasets	**HC Time**	**TABU Time**	**AGAVaPS Time**
Alarm	88.35 (68.35)	77.44 (38.65)	18,319.84 (3,005.39)
Asia	0.55 (0.09)	0.53 (0.06)	357.56 (29.58)
Coronary	0.26 (0.04)	0.26 (0.02)	500.70 (30.09)
Hailfinder	1,128.75 (1,245.82)	1,233.54 (1,455.88)	15,699.09 (946.69)
Insurance	13.12 (1.22)	11.44 (0.94)	784.67 (34.78)
Sachs	71.80 (9.80)	66.62 (13.84)	418.22 (6.50)
Sachs 2e4	3.87 (0.61)	3.66 (0.60)	70.04 (3.81)
Alarm rank	1	1	2
Asia rank	1	1	2
Coronary rank	1	1	2
Hailfinder rank	1	1	2
Insurance rank	2	1	3
Sachs rank	1	1	2
Sachs 2e4 rank	1	1	2
Mean Rank	1.14	**1.00**	2.14

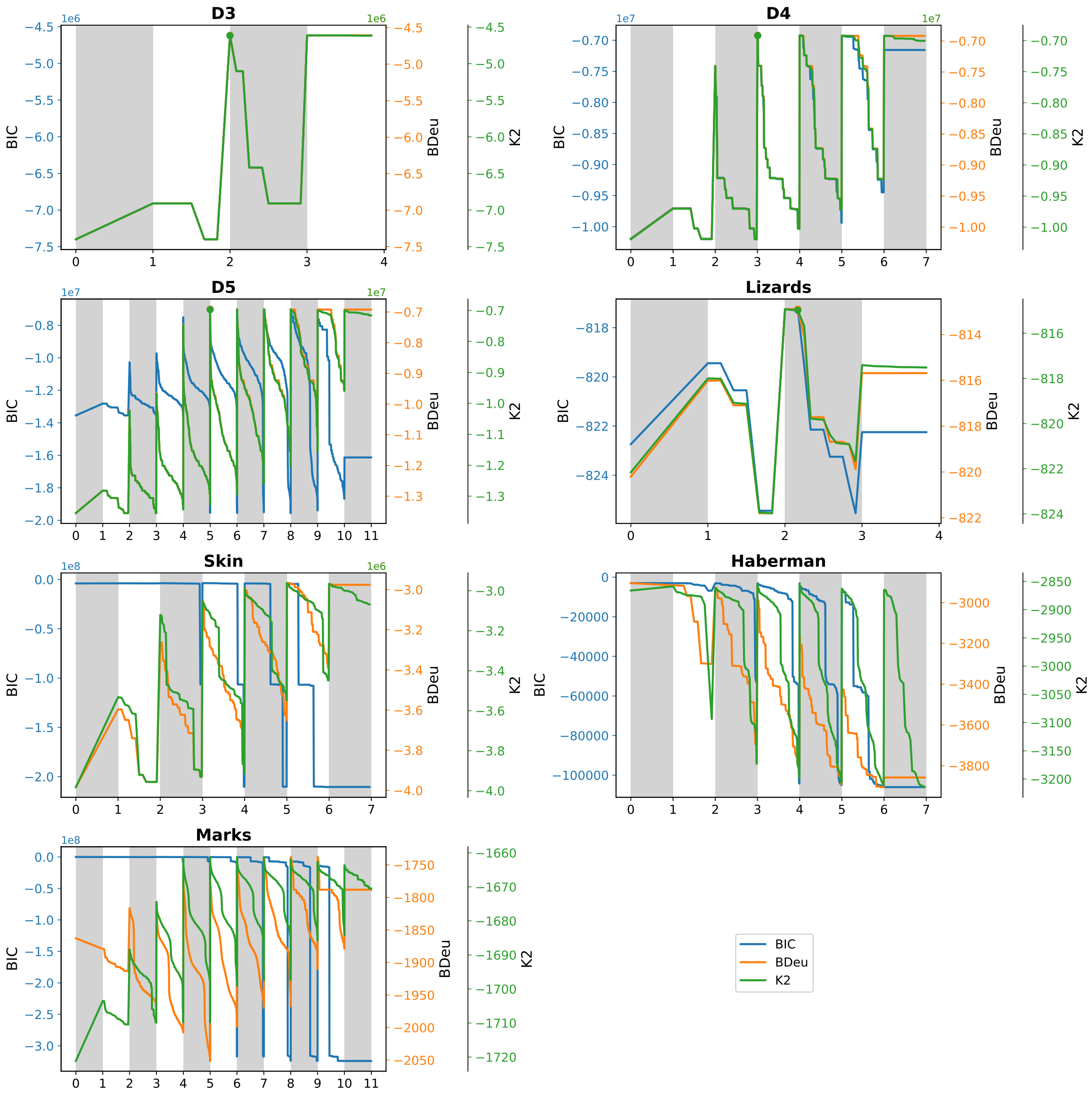
Source: Ribeiro and Maciel (2023)

From the execution time results, it can be seen that TABU was the fastest algorithm. TABU was tied with HC for five out of seven test cases. HC and TABU being faster than AGAVaPS was expected since both are simpler search algorithms.

4.7 BNSL Landscape Analysis

In Figure 32 the landscape analysis for the datasets D3, D4, D5, Lizards, Skin, Haberman and Marks can be seen. In it, the BIC, BDeu and K2 scores are considered, each one with a different color. The x-axis for each landscape plot is mapped to different structures so that each score plot would be sorted by the score for each number of edges.

Figure 32 – Landscape of the scores BIC, BDeu and K2 for the datasets D3, D4, D5, Lizards, Skin, Haberman and Marks. The expected structure is marked by a circle. For this plot, the x-axis is mapped to different structures for each score, having the structure sorted by each score. Skin, Haberman and Marks are real data datasets that don't have an expected structure, thus the location of the expected structure was not marked.

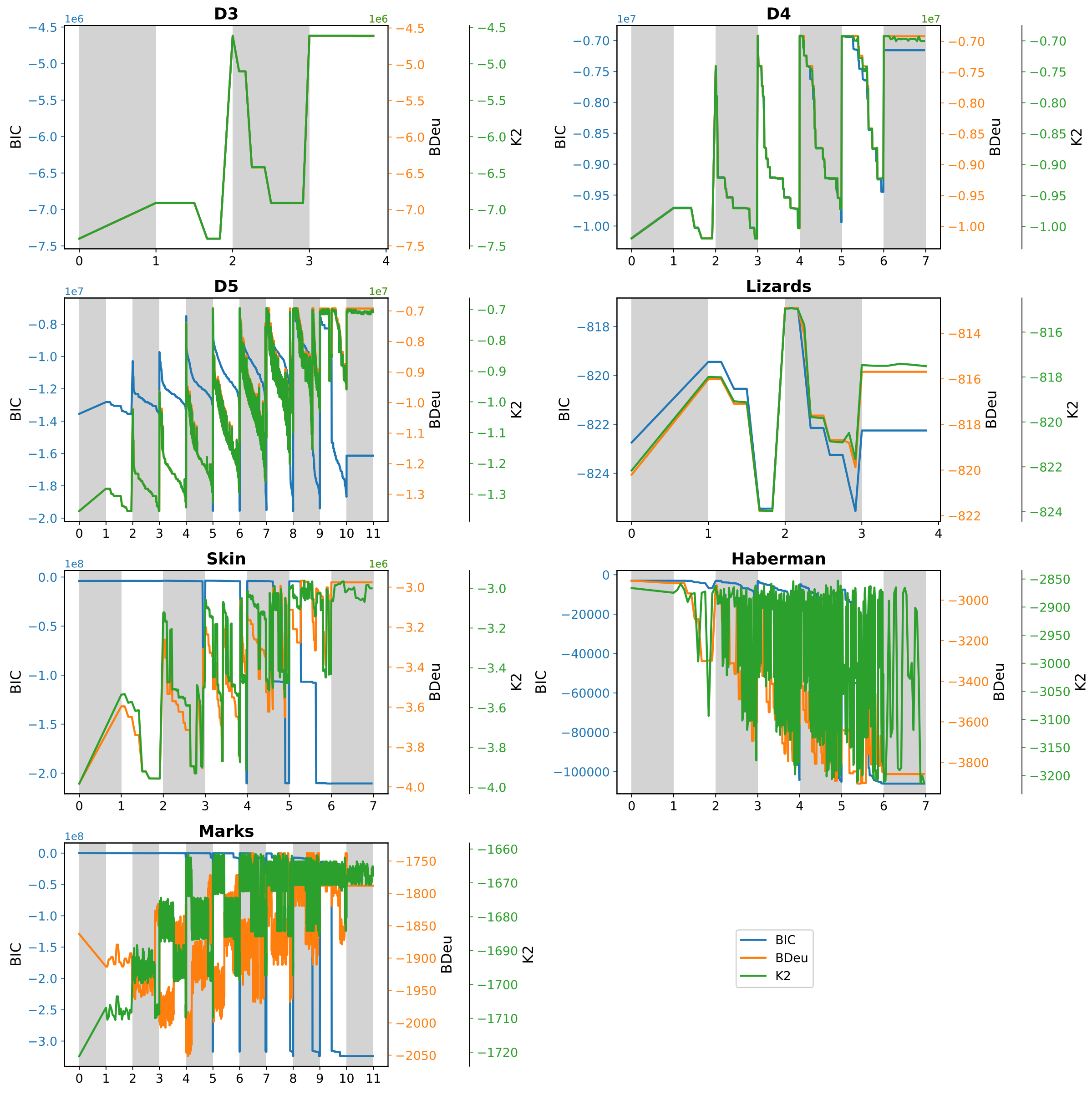


Source: Author

In Figure 32 it can be seen that the expected structures were between the best structures. In addition to that, it could be seen that BDeu and K2 had similar behavior. This similarity was expected since K2 is a specific case of the BDeu score. In addition to that, for the datasets with a smaller number of variables it could be seen that when there is enough data, all scores behave considerably similarly, which was also expected.

In Figure 33 the landscape analysis for the datasets D3, D4, D5, Lizards, Skin, Haberman and Marks can be seen. In it, the BIC, BDeu and K2 scores are considered, each one with a different color. The x-axis for each landscape plot is mapped to the same structures for all scores. The structures were sorted considering the BIC score. In it, it could be seen that despite having similar behaviors, BDeu and K2 present different evaluations from structures and have their differences. Also, it could be seen that many times the scores did not match which structure was the best.

Figure 33 – Landscape of the scores BIC, BDeu and K2 for the datasets D3, D4, D5, Lizards, Skin, Haberman and Marks. For this plot, the x-axis is mapped to the same structures for all scores. The structures were sorted considering the BIC score.



Source: Author

In Table 22 the match of best structures (number of scores that had the same best structure) for each number of edges can be seen. For some datasets, the scores agree a lot and for others, they don't. All the times when only 2 scores agreed, the match was between K2 and BDeu.

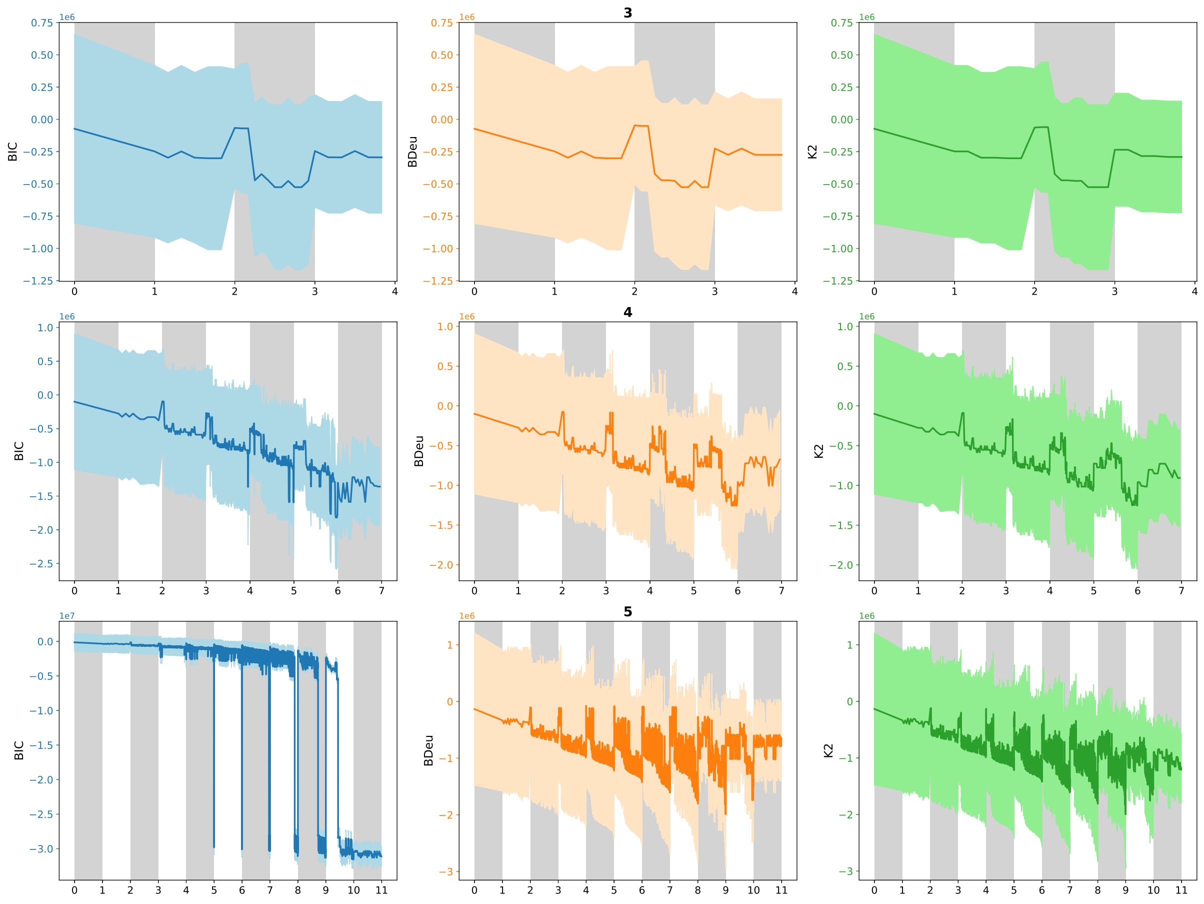
Table 22 – Match of best structures between BIC, BDeu and K2 scores for each number of edges.

Dataset	**Number of edges**									
	1	2	3	4	5	6	7	8	9	10
D3	3	3	0	-	-	-	-	-	-	-
D4	2	3	2	0	0	0	-	-	-	-
D5	2	3	3	3	2	0	0	0	0	0
Lizards	2	2	2	-	-	-	-	-	-	-
Skin	2	2	0	0	0	0	-	-	-	-
Haberman	3	0	0	0	0	0	-	-	-	-
Marks	2	0	0	0	0	0	0	0	0	0

Source: Author

After this first analysis, the variation of this landscape considering different dataset sizes and different samples was explored. In Figure 34 the mean and standard deviation of the scores when varying the samples of a dataset are shown. From it, it can be seen that the samples included in a dataset greatly influence the resulting landscape and score. This indicates that having a large dataset is important, to decrease this effect.

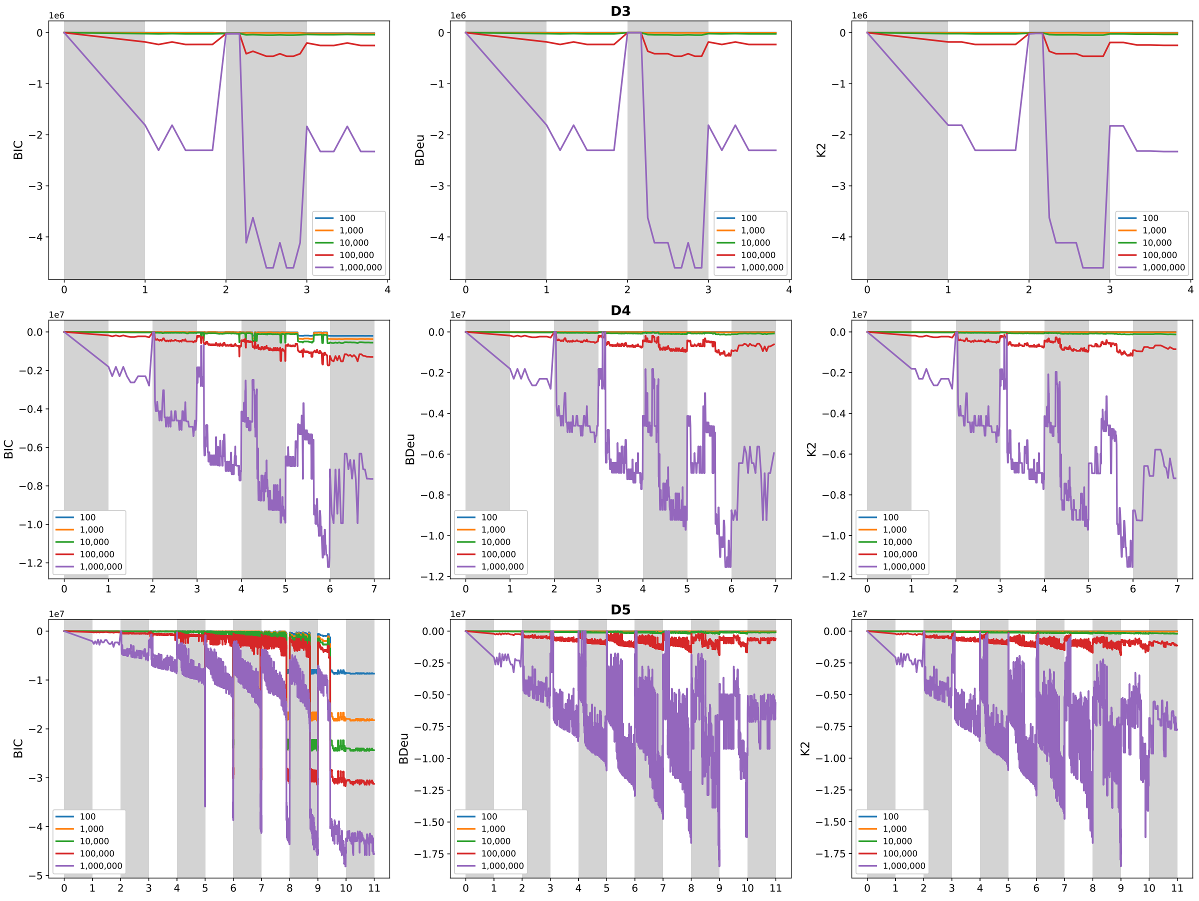
Figure 34 – Landscape analysis with the variation of the scores resulting from having different samples of the same system. The lighter-colored fill indicates the standard deviation of the score and the dark-colored line indicates the mean value of the score.



Source: Author

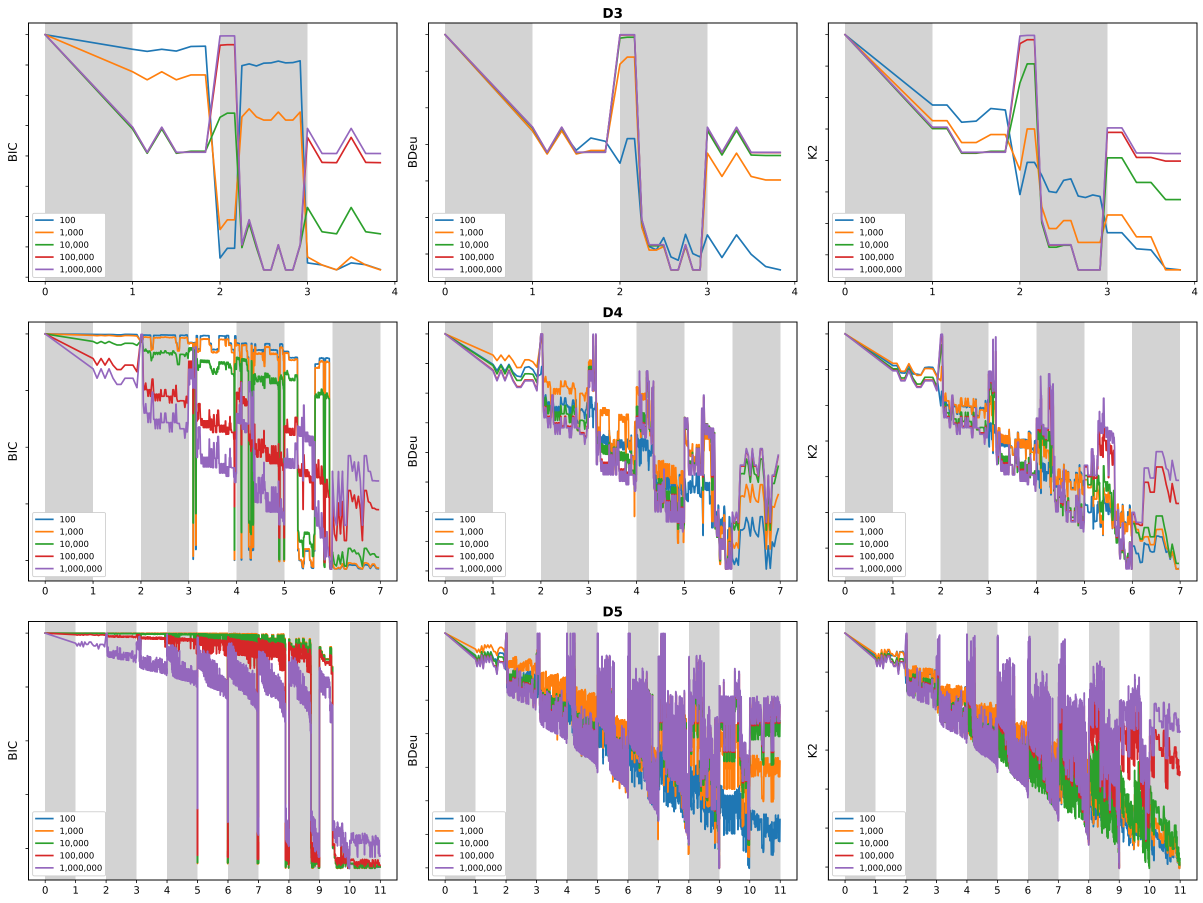
In Figures 35 and 36 the effect of dataset size on the landscape can be seen. In Figure 35 the landscape plots are done on the same y-axis, while in Figure 36 they are plotted using different y-axis so that the individual behaviors can be seen. From the plots, it can be seen that all datasets presented a change in the landscape with the size change. The most obvious change was the range of the values of the score. There were also changes in which structures were the best. In addition to that, BIC was the score that presented less variation in behavior when the dataset size was changed.

Figure 35 – Landscape analysis considering different dataset sizes and using the same x-axis and y-axis.



Source: Author

Figure 36 – Landscape analysis considering different dataset sizes and using the same x-axis and different y-axis.

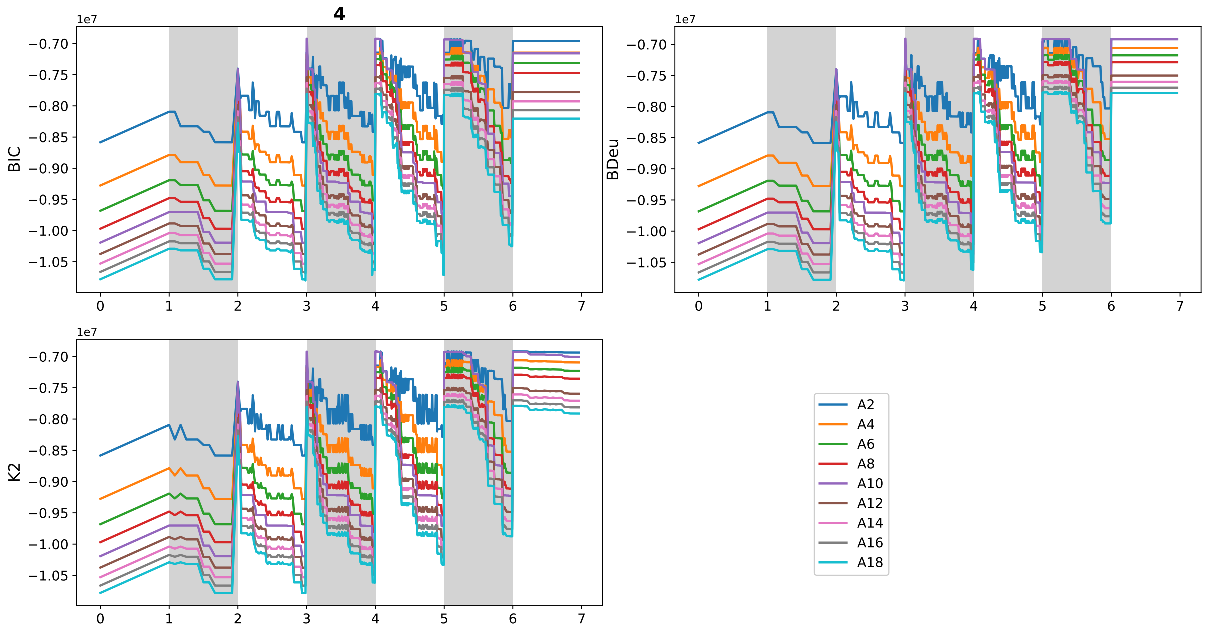


Source: Author

4.8 BNSL Landscape Variation Considering Quantization

The result for the variation of the quantization of one variable for the D4 dataset can be seen in Figure 37. From it, it can be seen that the variation of a single variable quantization is capable of changing the overall range of values of the score and can make small changes to the landscape, including changing the best structure by a slight score value.

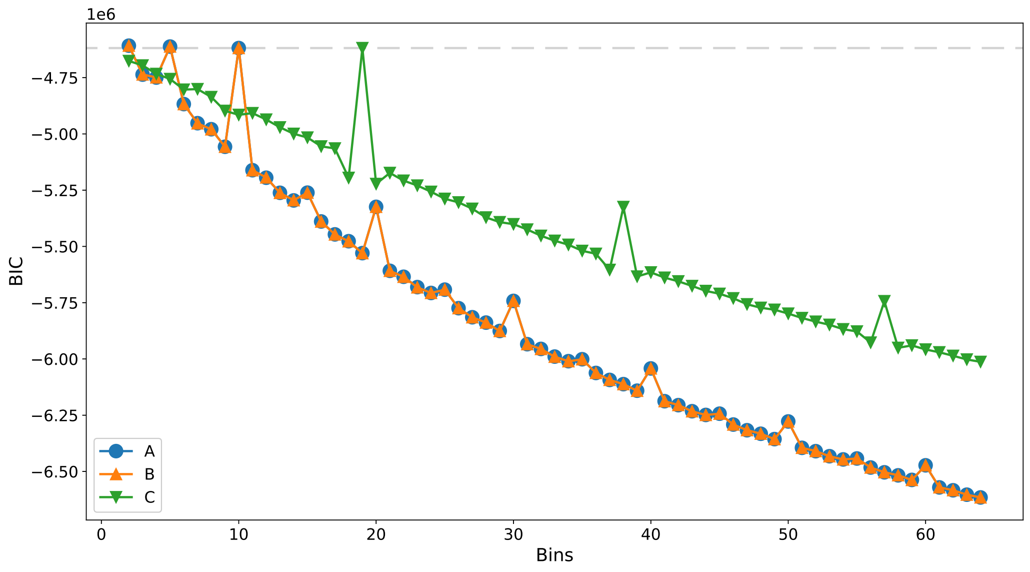
Figure 37 – Landscape variation when varying the quantization of variable A for the D4 dataset. The ideal quantization of A is 10.



Source: Author

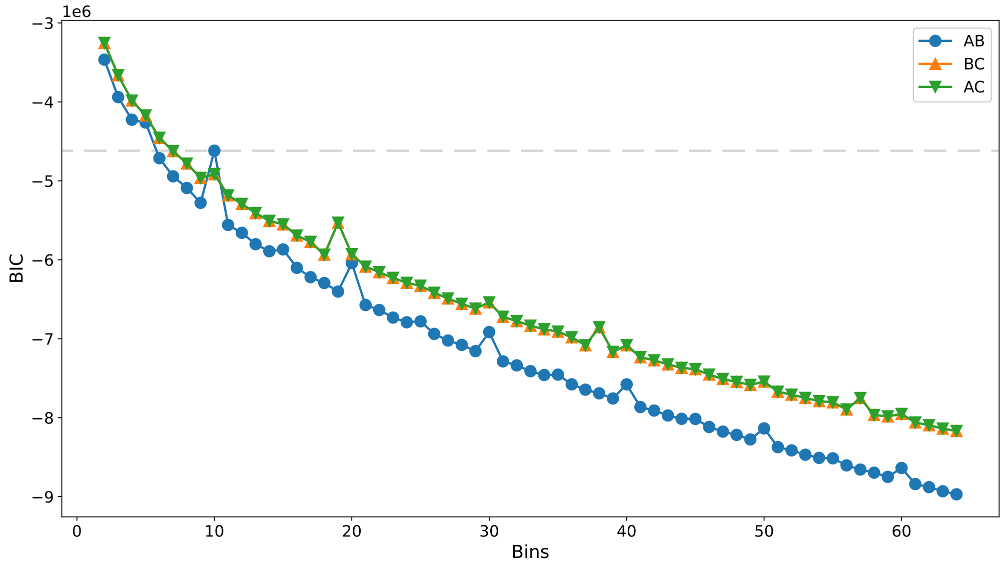
In Figures 38, 39 and 40 the variation of the landscape when varying one, two, and three variables at the same time for the D3 dataset can be seen. For all of them the quantization refers to the number of states a variable can assume. This dataset has a determined ideal quantization of ten bins for A and B and nineteen for C. It is interesting to observe that in all three figures for the ideal quantization values and its multiples an improvement of the score occurs. It is also interesting to observe that for all of them, the score for low quantization values tends to be higher.

Figure 38 – Landscape variation when varying the quantization of one variable at a time for the D3 dataset. The dashed line indicates the score when considering the ideal quantization values.



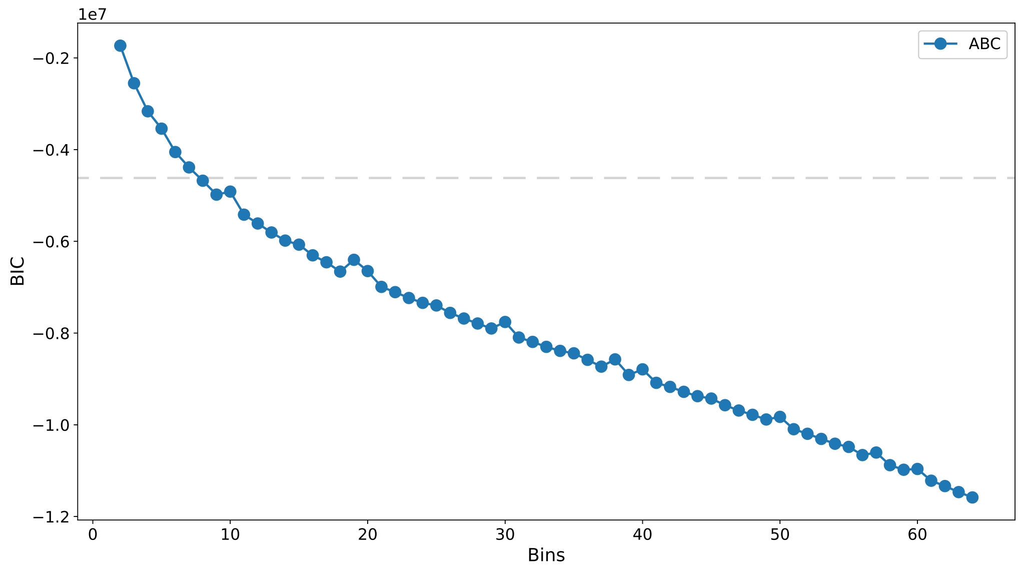
Source: Author

Figure 39 – Landscape variation when varying the quantization of two variables at a time for the D3 dataset. The dashed line indicates the score when considering the ideal quantization values.



Source: Author

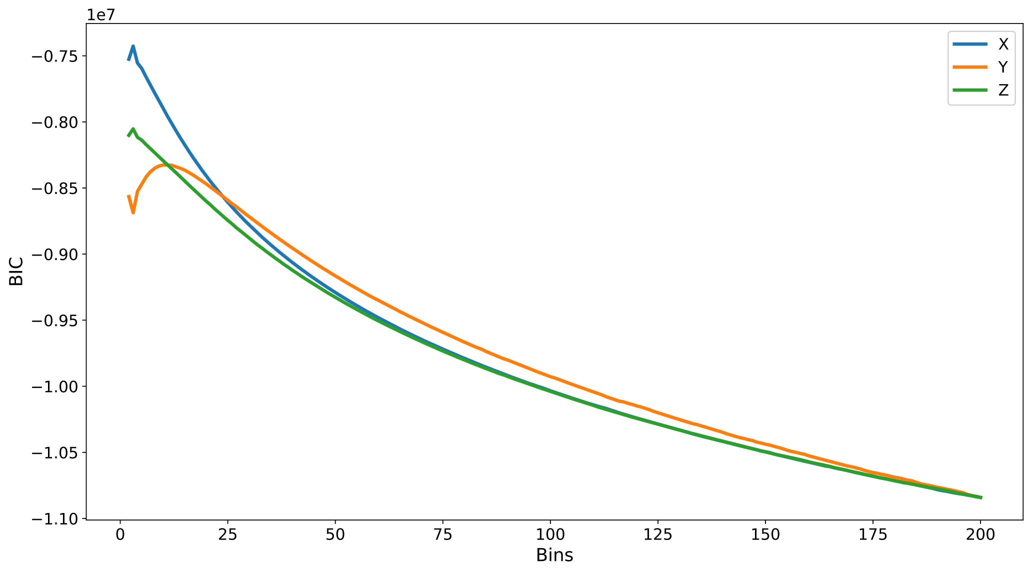
Figure 40 – Landscape variation when varying the quantization of all variables at the same time for the D3 dataset. The dashed line indicates the score when considering the ideal quantization values.



Source: Author

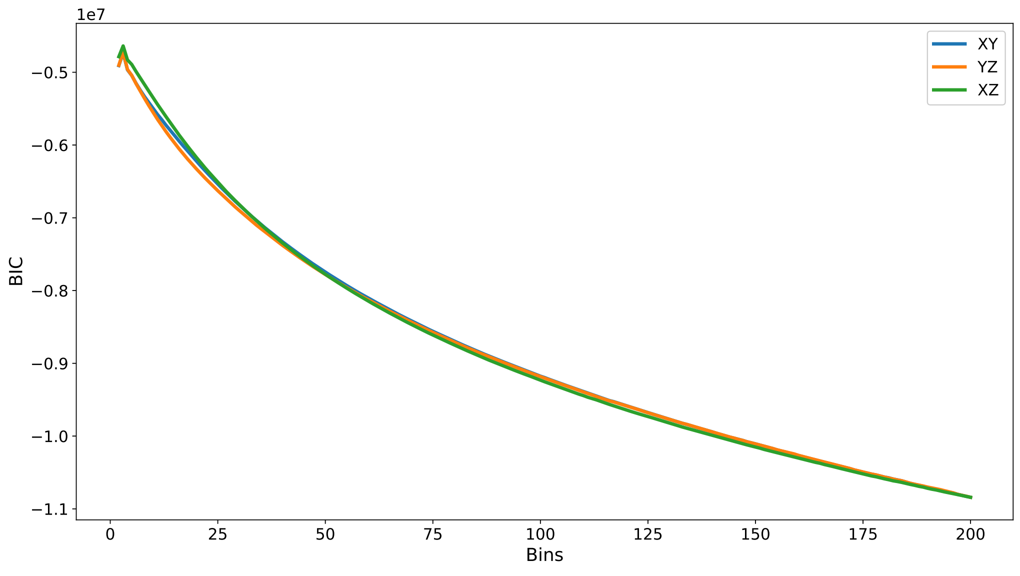
In Figures 41, 42 and 43 the variation of the landscape when varying one, two, and three variables at the same time for the XYZ dataset can be seen. From it, it can be seen that differently from D3 which had an ideal quantization, on these landscapes no specific quantization resulted in a sudden increase in the score. In addition to that, the general behavior of the score being higher for lower quantization values remained.

Figure 41 – Landscape variation when varying the quantization of one variable at a time for the XYZ dataset.



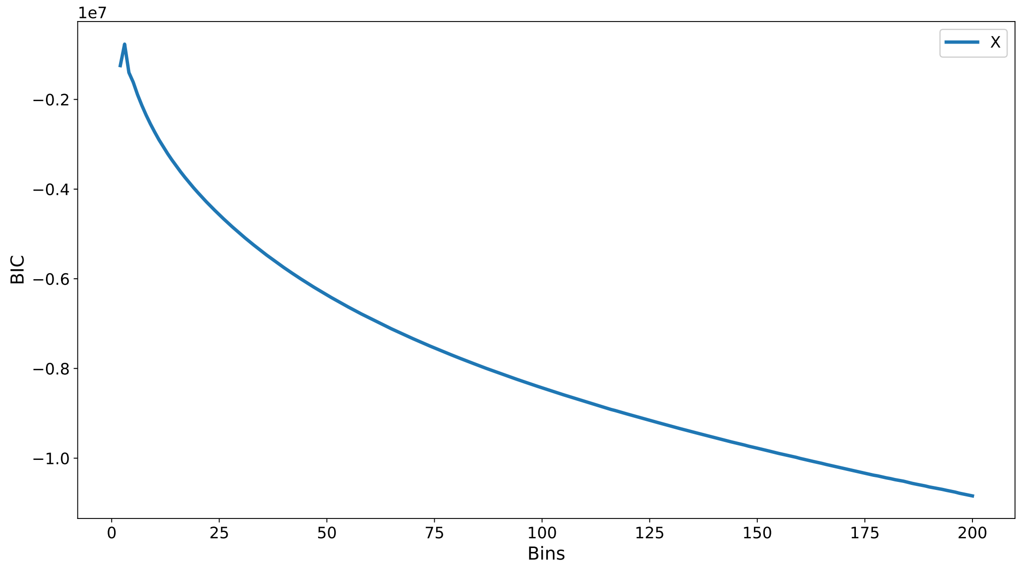
Source: Author

Figure 42 – Landscape variation when varying the quantization of two variables at a time for the XYZ dataset.



Source: Author

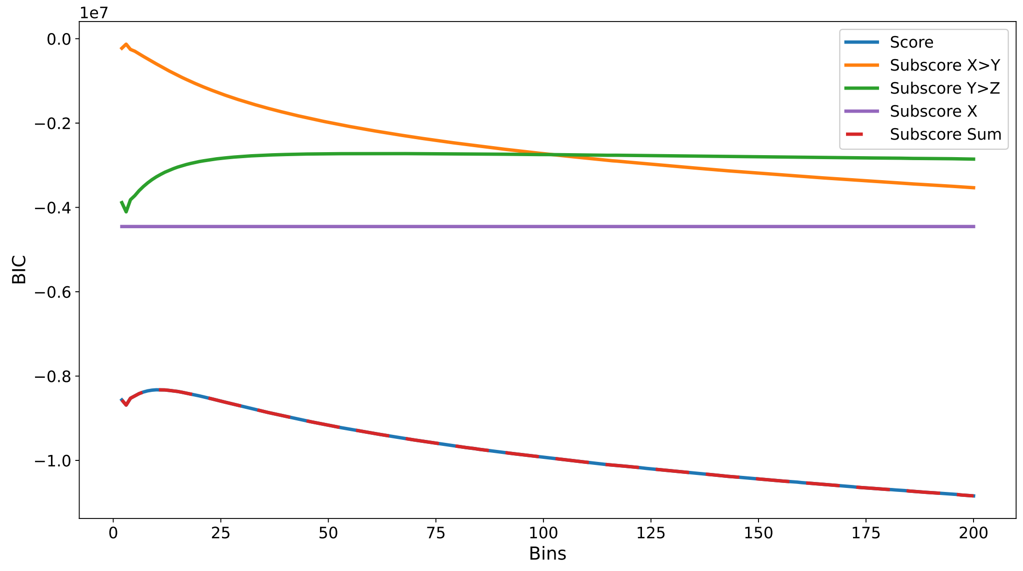
Figure 43 – Landscape variation when varying the quantization of all variables at the same time for the XYZ dataset.



Source: Author

When the Y variable, which is the middle variable on the expected structure, was the only variable an upward variation was observed before the declining behavior. To investigate what caused this behavior the components of the score were analyzed separately while the quantization of Y was varied. This analysis can be seen in Figure 44. From it, it can be observed that the contribution of the edge $X \to Y$ presents the general behavior of descent as the quantization increases. Meanwhile, the contribution of the edge $Y \to Z$ presents an increasing behavior that seems to converge to a value.

Figure 44 – BIC score components when varying the quantization of one variable at a time for the XYZ dataset.

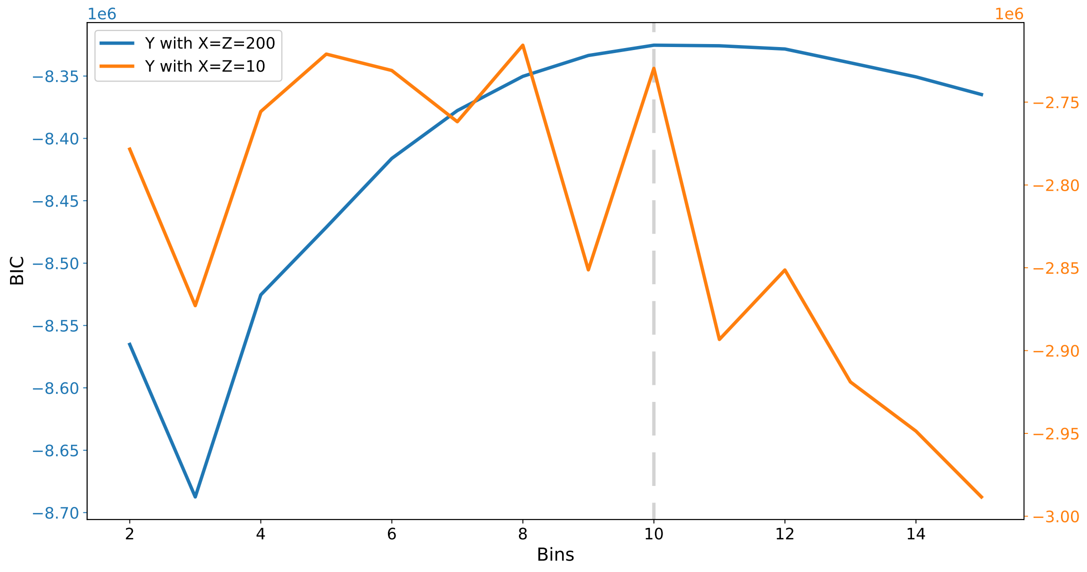


Source: Author

Thus, it can be seen that the decrease in the quantization of a successor is beneficial for the score, while the increase in the quantization of a predecessor is beneficial to the score, but only until a certain point. With this in mind, choosing the peak point of the curve as the quantization of Y seems like a good quantization strategy.

In Figure 45 a representation of this method and its limitations is shown. The blue curve was made with X and Z quantizations fixed on 200, for it the best quantization for Y would be 10. However, if X and Z quantizations are fixed in 10, the curve becomes very noisy and the best quantization for Y would be 8. However, since the curve becomes so noisy, the method does not seem as trustworthy as it seemed at first glance.

Figure 45 – Landscape for the XYZ dataset varying variable Y quantization with two different fixed values for variables X and Z quantizations. The plots were made with different y-axis, each axis is indicated by different colors, matched to the color of the curve plotted on it.

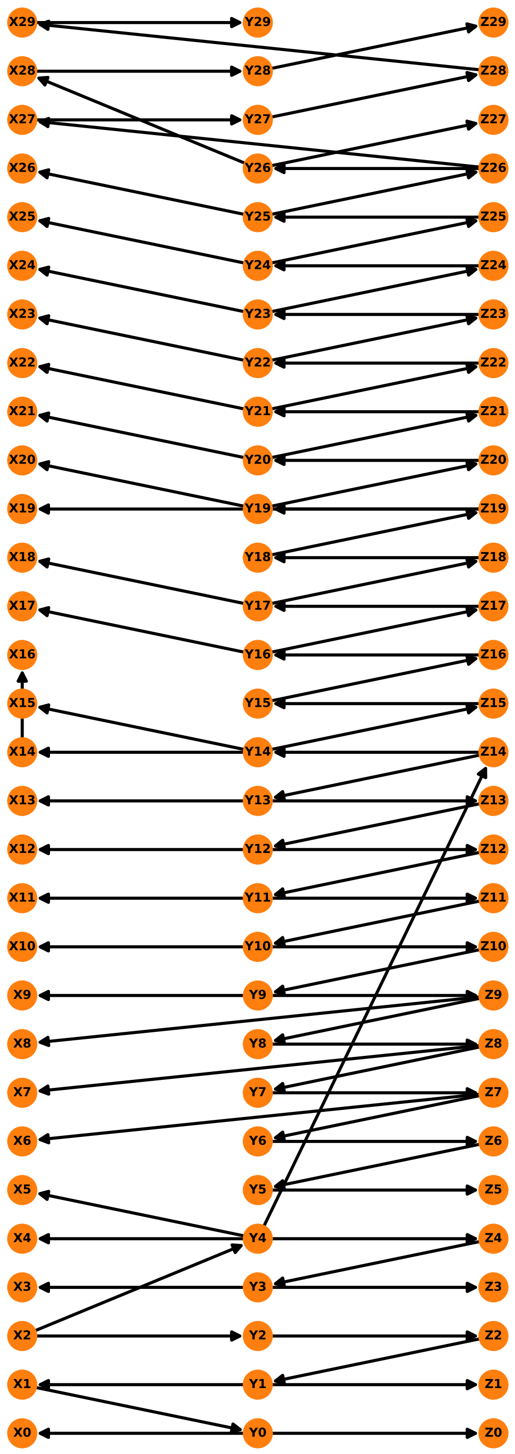


Source: Author

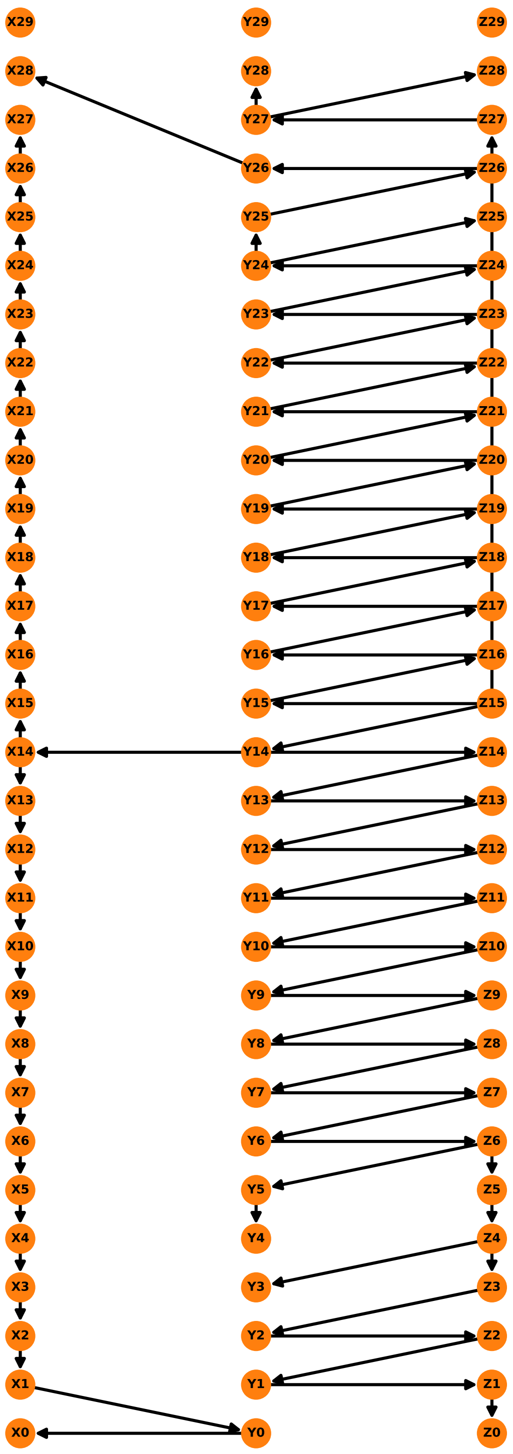
4.9 BNSL Quantization by Variable Transformation

The results of the tests from variable transformation to bins variables and cut-points variables can be seen in Figure 46. The expectation for this method to work to reduce the quantization would be to have many bins that relate to the same bin in other variables for the bins variables case and to have cut points that did not connect to the other variables for the cut-points variables. However, as it can be seen the results obtained did not match that.

Figure 46 – BN structure learned by Blip for the transformation of the variables to bins and cut points for the XYZ dataset.



(a) Bins Variables

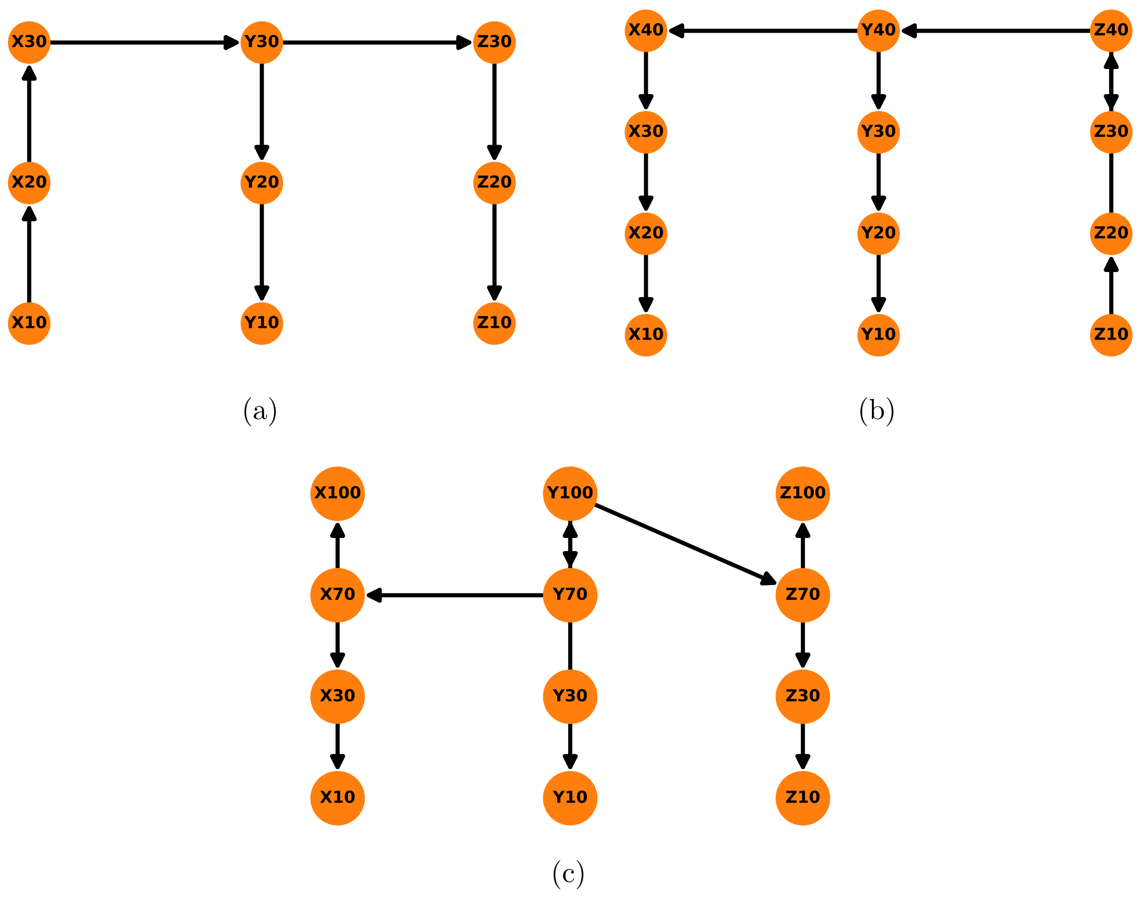


(b) Cut-points Variables

Source: Author

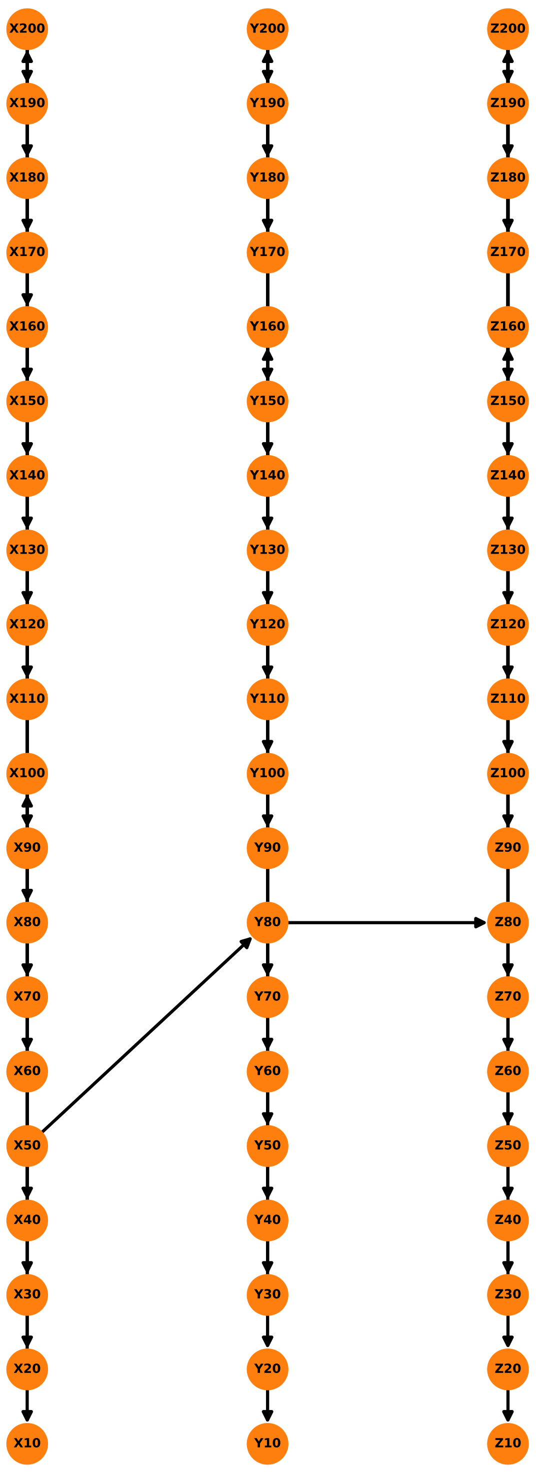
The results of the transformation in variables of multiple quantizations for each variable can be seen in Figures 47 and 48. From them, it can be seen that the result was exactly as expected, in which there is only one edge connecting one variable to the other. However, it still lacks consistency of the result depending on the quantizations used.

Figure 47 – BN structure learned by Blip for the transformation to variables multiple quantizations for the XYZ dataset.



Source: Author

Figure 48 – BN structure learned by Blip for the transformation to variables multiple quantizations for the XYZ dataset.

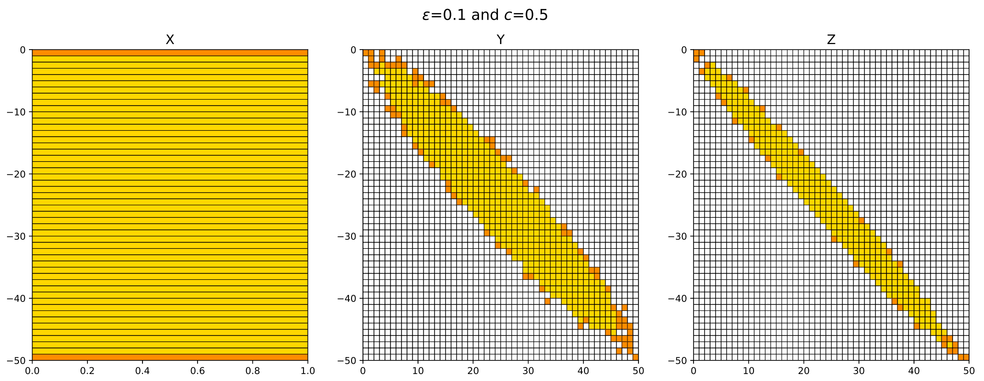


Source: Author

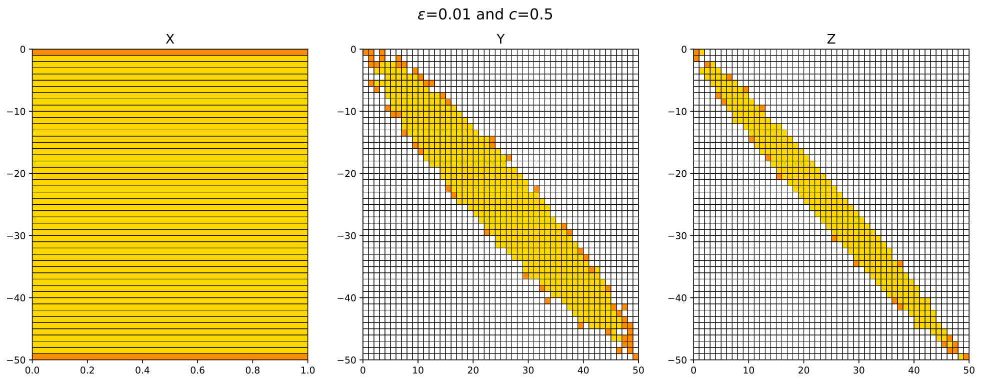
4.10 MDL Non-Zero Score Test

The results of the application of the MDL Non-Zero score to the datasets XYZ and XYZ3 can be seen in Figures 49 and 50. Figure 49 includes different values of the parameters ε and c to give a better understanding of their effect. From the results, it's clear that the MDL Non-Zero score would rather that outliers values did not occur to improve the score. This means that these values had their estimator $\hat{N}_{ijk}$ set to zero and that improved the score.

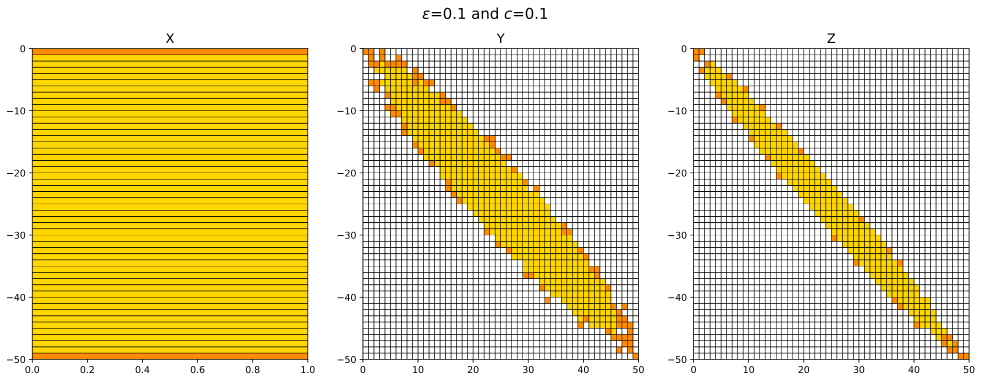
Figure 49 – Results of the application of the MDL Non-Zero score to the XYZ dataset considering different values of ε and c. The values of the CPT that were considered as relevant by MDL Non-Zero are marked in yellow and the values that the MDL Non-Zero score would rather have not existed to improve the score value are marked in orange.



(a) $\varepsilon = 0.1$ and $c = 0.5$



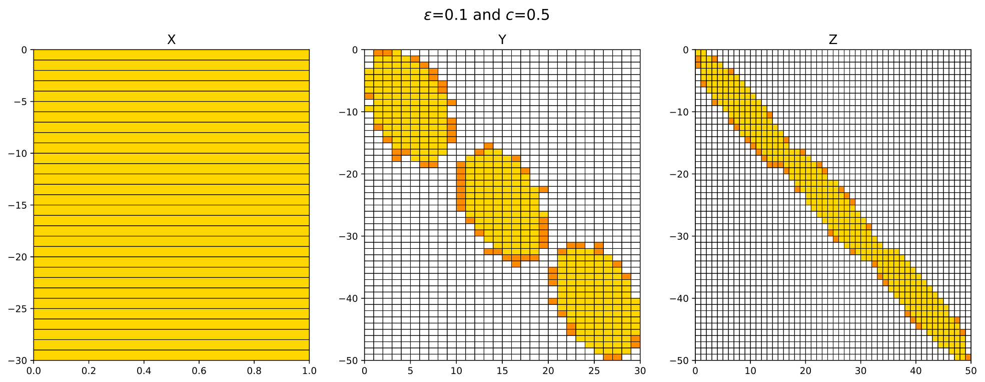
(b) $\varepsilon = 0.01$ and $c = 0.5$



(c) $\varepsilon = 0.1$ and $c = 0.1$

Source: Author

Figure 50 – Result of the application of the MDL Non-Zero score to the XYZ3 dataset considering $\varepsilon = 0.1$ and $c = 0.5$. The values of the CPT that were considered as relevant by MDL Non-Zero are marked in yellow and the values that the MDL Non-Zero score would rather have not existed to improve the score value are marked in orange.

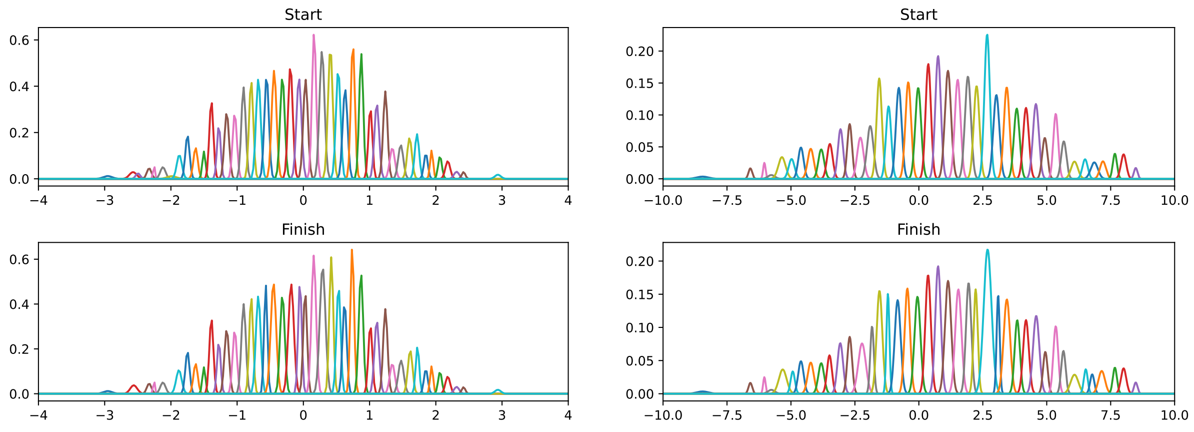


Source: Author

4.11 EM BN Test

The results of the EM BN tests for the XYZ dataset only using the X and Y variables and for the dataset 2T can be seen in Figures 51 and 52. In both of them, it can be seen that the distributions used at the start and end of the execution have differences, indicating the changing of points from one bin to another. However, the changes a very slight and do not significantly change the number of bins used.

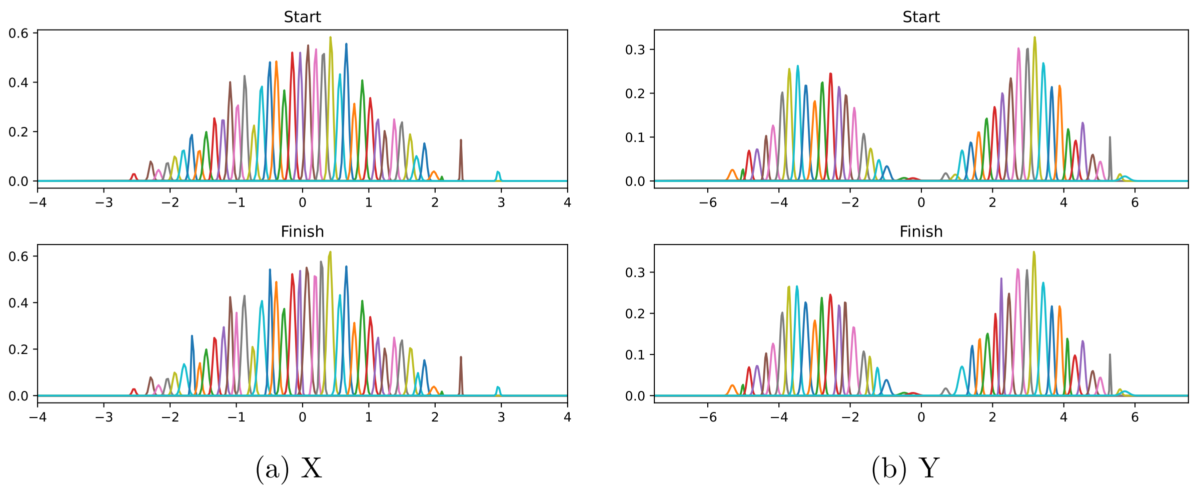
Figure 51 – Normal distributions used to model each bin at the start and finish on the EM BN test for the XYZ dataset considering only the X and Y variables.



(a) X (b) Y

Source: Author

Figure 52 – Normal distributions used to model each bin at the start and finish on the EM BN test for the 2T dataset.

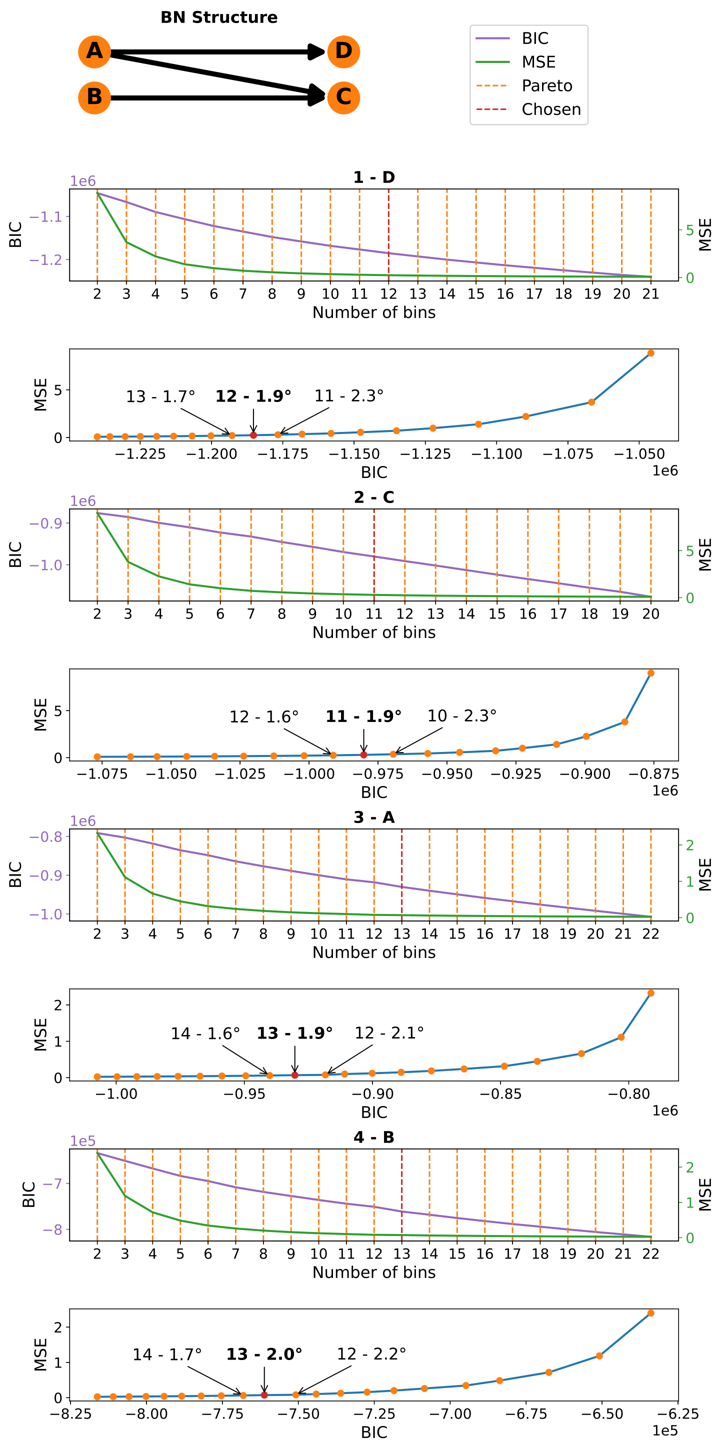


Source: Author

4.12 CLBQ Step-by-step Test

The result of the step-by-step test of CLBQ for the D4 dataset can be seen in Figure 53. On it, the process of CLBQ can be better seen and understood. For each step, the variable selected and its MSE, BIC score, and Pareto set plot are shown. The quantization value and the angle θ for the selected quantization and its neighbors are also shown on the Pareto plot.

Figure 53 – Step-by-step test of CLBQ for the D4 dataset and the structure used on it. On the BICxMSE plot, the Pareto front found by CLBQ is plotted, and the number of bins and angle θ are presented for the selected quantization (highlighted in bold letters) and its first neighbors. At the top of each chart, the number of each step is shown in bold letters and which variable is being analyzed on it. BIC is the structure score, and MSE is the mean squared error between quantized data for a given number of bins and the original data. Number of bins refers to the amount used to quantize the variable. Pareto refers to the members of the Pareto set found. Chosen refers to the selected quantization value from the Pareto set.



Source: Ribeiro *et al.* (in press)

4.13 CLBQ Performance Test

The results of the quantization chosen by CLBQ and DD and their MSE for all datasets can be seen in Tables 23, 24, 25 and 26. For the Weather dataset three structures were considered generating different quantizations. The first structure considered was the structure learned using the PC algorithm that can be seen in Figure 56. The other two structures are the best scoring structures for the Weather dataset with 3 and 4 edges, which can be seen in Figure 61b. The comparison of the histogram of the quantizations and the variables' actual distribution can be seen in Figures 54 and 57 for CLBQ and in Figures 55 and 58 for DD. From these results, it can be seen that CLBQ presented better modeling of the variables when considering MSE and also had more interesting histograms. The results in Figure 54c are especially curious since CLBQ chooses a quantization that models each peak of the distribution with one bin.

Table 23 – Results comparing the quantization values and MSE of the ideal quantization, the quantization obtained by CLBQ, and the quantization obtained by DD for the D4 dataset.

	q_A	q_B	q_C	q_D	**MSE**$_A$	**MSE**$_B$	**MSE**$_C$	**MSE**$_D$
Ideal	10	10	19	19	0.120	0.126	**0.100**	**0.097**
CLBQ	13	13	11	12	**0.069**	**0.074**	0.298	0.245
DD	9	13	11	4	5.763	0.988	1.334	2.208

Source: Ribeiro *et al.* (in press)

Table 24 – Results comparing the quantization values and MSE of the quantization obtained by CLBQ and the quantization obtained by DD for the XYZ dataset.

	q_X	q_Y	q_Z	**MSE**$_X$	**MSE**$_Y$	**MSE**$_Z$
CLBQ	9	10	10	**0.095**	**0.788**	**3.140**
DD	4	4	4	0.502	5.118	20.403

Source: Ribeiro *et al.* (in press)

Table 25 – Results comparing the quantization values and MSE of the quantization obtained by CLBQ and the quantization obtained by DD for the XYZ3 dataset.

	q_X	q_Y	q_Z	**MSE**$_X$	**MSE**$_Y$	**MSE**$_Z$
CLBQ	5	11	12	**0.010**	**0.254**	**0.912**
DD	4	4	4	0.470	2.323	10.997

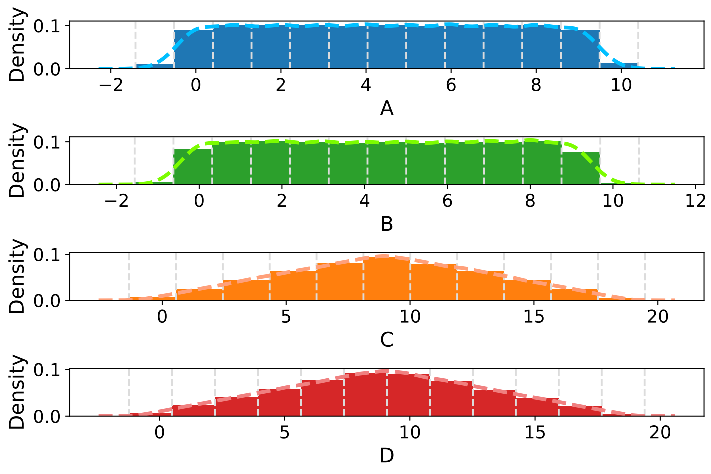
Source: Ribeiro *et al.* (in press)

Table 26 – Results comparing the quantization values and MSE of the quantization obtained by CLBQ for three different structures and the quantization obtained by DD for the XYZ dataset.

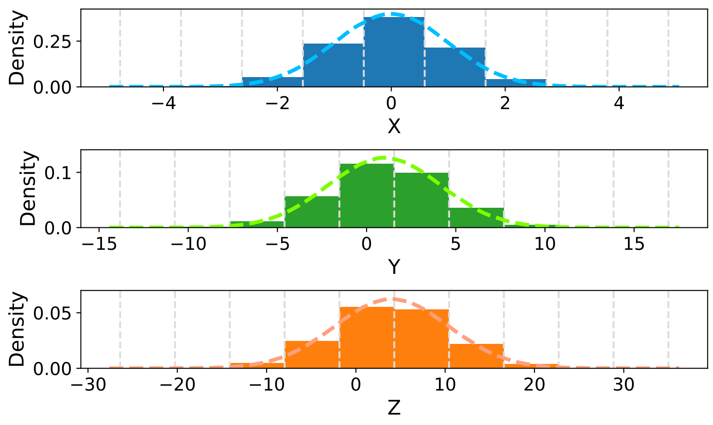
	$q_{\textbf{Tins}}$	$q_{\textbf{RHins}}$	$q_{\textbf{Rad}}$	$q_{\textbf{Wspeed}}$	$\textbf{MSE}_{\textbf{Tins}}$	$\textbf{MSE}_{\textbf{RHins}}$	$\textbf{MSE}_{\textbf{Rad}}$	$\textbf{MSE}_{\textbf{Wspeed}}$
Q_{PC}	14	14	14	13	**0.241**	**3.261**	**26,498.219**	**0.075**
Q_3	13	13	11	11	0.281	3.557	43,433.104	0.105
Q_4	12	13	11	11	0.327	3.557	43,433.104	0.105
DD	4	6	11	8	4.759	32.588	73,400.493	0.118

Source: Ribeiro *et al.* (in press)

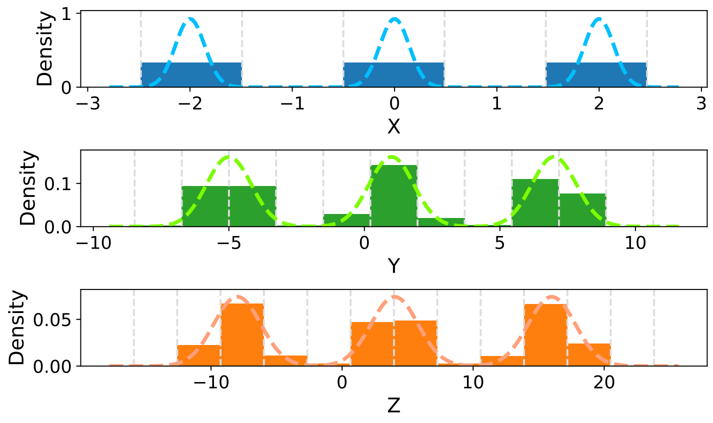
Figure 54 – Plot of the histogram of the quantization found by CLBQ for the expected structure and the actual variables' distribution for the D4, XYZ, XYZ3 datasets. The variables' distribution is shown in lighter-colored dashed lines.



(a) D4



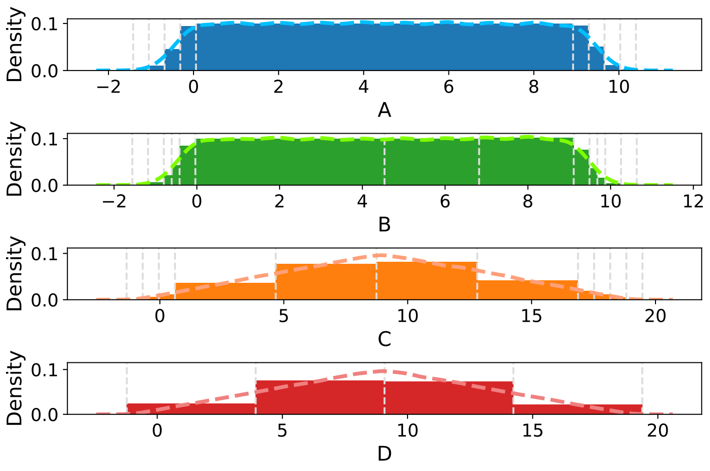
(b) XYZ



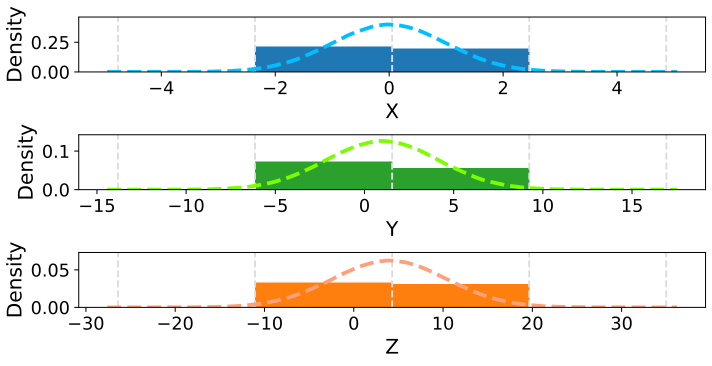
(c) XYZ3

Source: Ribeiro *et al.* (in press)

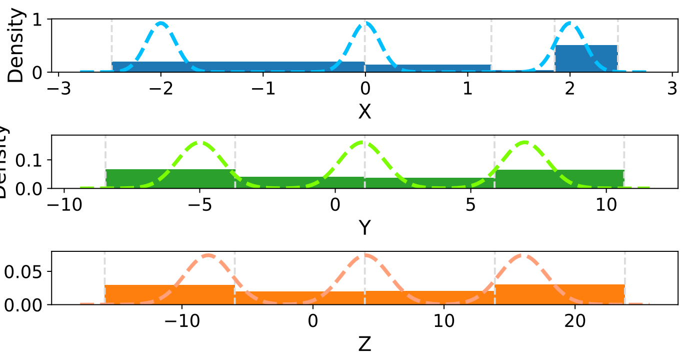
Figure 55 – Plot of the histogram of the quantization found by DD and the actual variables' distribution for the D4, XYZ, XYZ3 datasets. The variables' distribution is shown in lighter-colored dashed lines.



(a) D4



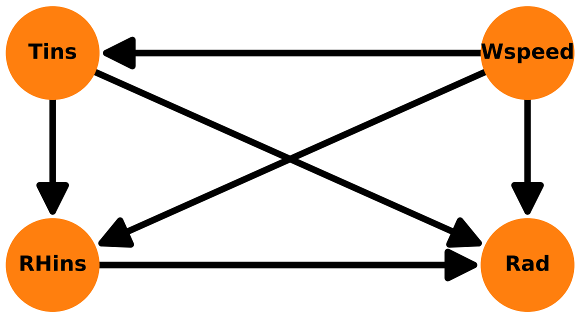
(b) XYZ



(c) XYZ3

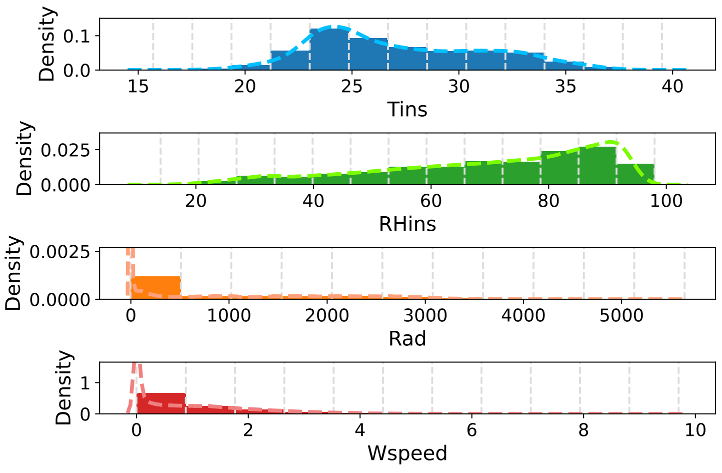
Source: Ribeiro *et al.* (in press)

Figure 56 – BN structure found using the PC algorithm for the Weather dataset. Tins refers to the current temperature, Wspeed refers to the wind speed, RHins refers to the current relative humidity and Rad refers to the current radiation.

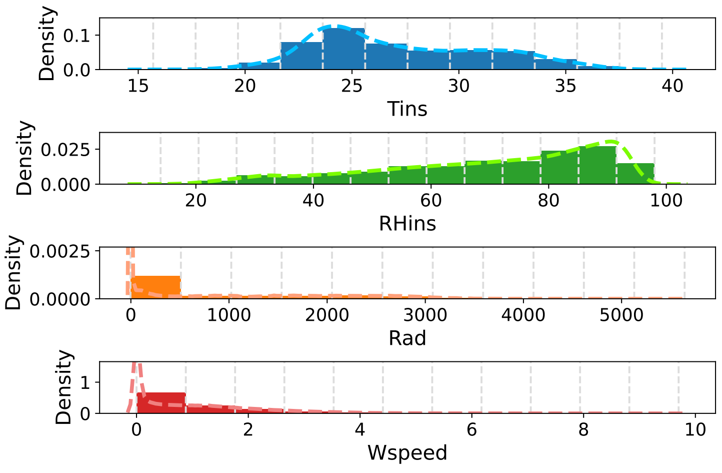


Source: Ribeiro *et al.* (in press)

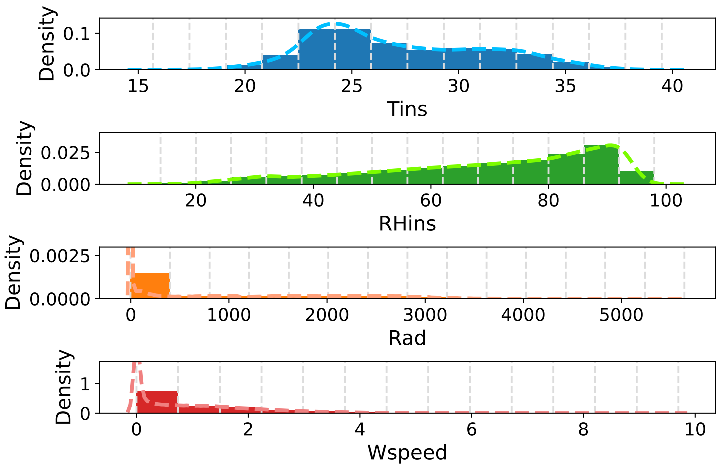
Figure 57 – Plot of the histogram of the quantization found by CLBQ for the best structure found with 3 and 4 edges and the structure found using PC, and the actual variables' distribution for the Weather dataset. The variables' distribution is shown in lighter-colored dashed lines.



(a) Q_3



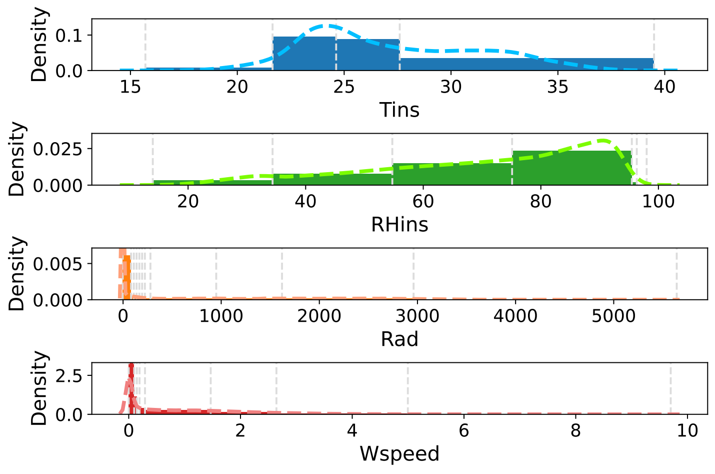
(b) Q_4



(c) PC

Source: Ribeiro *et al.* (in press)

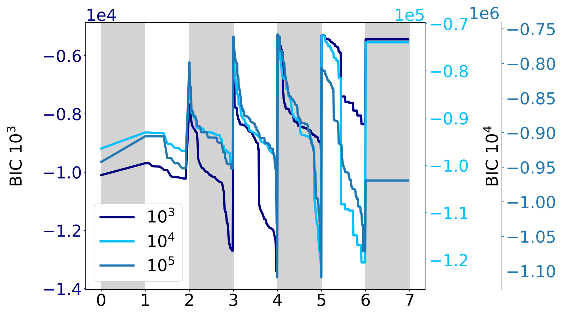
Figure 58 – Plot of the histogram of the quantization found by DD and the actual variables' distribution for the Weather dataset. The variables' distribution is shown in lighter-colored dashed lines.



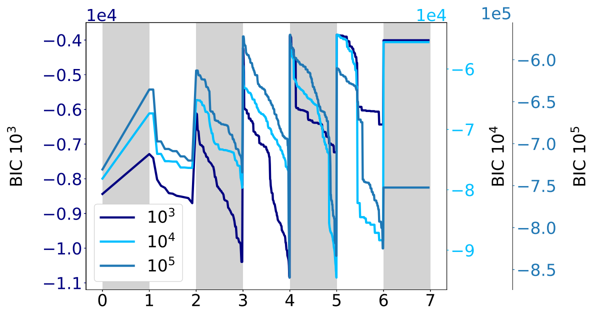
Source: Ribeiro *et al.* (in press)

The landscape analysis with different sample sizes can be seen in Figure 59 and the mean and STD execution time of CLBQ from making these landscapes can be seen in Table 27. From the landscape it can be seen that there was a consistency in selecting the same structures as the best ones despite the change in dataset size and the variance of the landscape affects structures with worse scores. This consistency of the best suggests that the use of CLBQ on BNSL is possible.

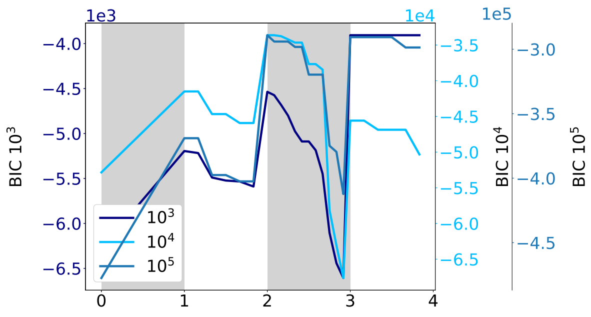
Figure 59 – Landscape plots made for the BIC score while using CLBQ considering different dataset sizes. The dataset sizes used were 10^3, 10^4, and 10^5 samples.



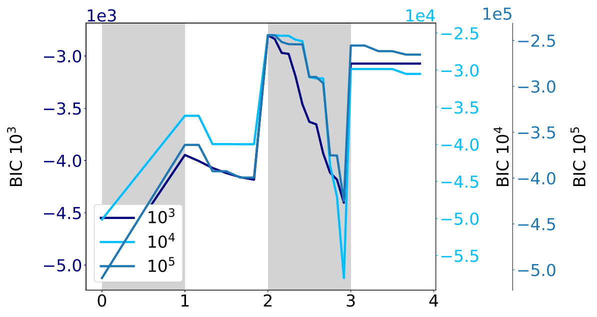
(a) D4



(b) Weather



(c) XYZ



(d) XYZ3

Source: Ribeiro *et al.* (in press)

Table 27 – Mean and STD execution time of CLBQ when building the landscape plots with different dataset sizes. All values are in seconds and the results are shown in the format mean (std).

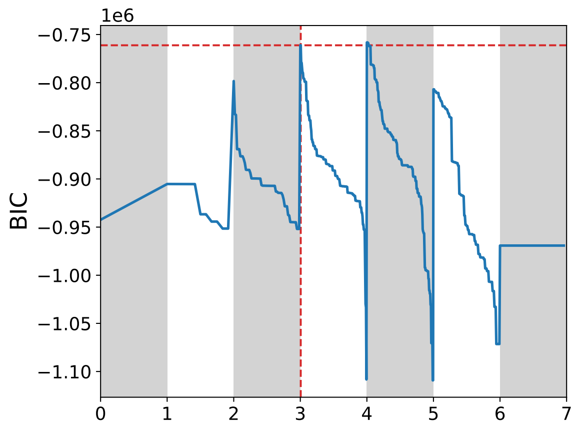
Dataset	**Size** $= 10^3$	**Size** $= 10^4$	**Size** $= 10^5$
D4	1.55 (0.66)	6.69 (1.87)	55.88 (9.98)
Weather	1.32 (0.66)	2.93 (0.86)	10.56 (1.92)
XYZ	1.29 (0.53)	4.02 (0.45)	29.73 (2.38)
XYZ3	1.17 (0.36)	3.70 (0.35)	28.48 (2.30)

Source: Ribeiro *et al.* (in press)

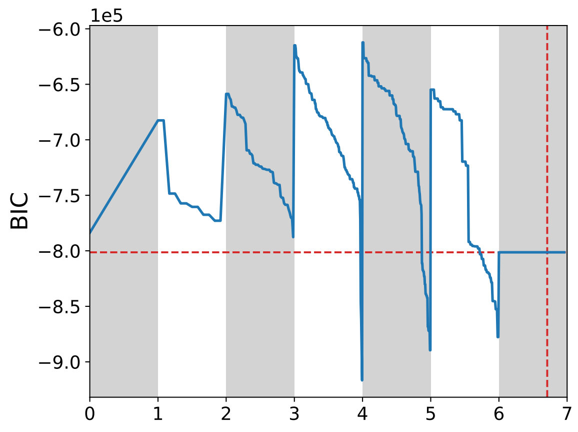
Considering the execution time, the execution time of CLBQ varies a lot between dataset sizes and between datasets. This discrepancy is likely attributed to the evaluation of the score and the distributions of variables. The BIC score utilized considers the number of data points for each element in every CPT, leading to an evaluation time dependent on the dataset size. Additionally, variations in the distributions of variables could account for differences between datasets, as the number of quantization values assessed until a trade-off is reached varies based on the variables' distribution, influencing both the structural score and the MSE values employed in CLBQ. While the time required for CLBQ to select quantizations may not be as minimal as desired, it proves to be quite efficient for a single-structure evaluation. Despite potentially contributing to an increased running time in the context of search and score BNSL, this additional time could be considered valuable in exchange for having the quantization tailored for each structure, especially given the inherently time-intensive nature of the BNSL process.

The analysis of the location of the expected structure on the CLBQ landscape can be seen in Figure 60. It is observed that for dataset D4, the expected BN proved to be the optimal structure with three edges. However, for datasets XYZ and XYZ3, the expected BN, while not the most favorable, still exhibited a commendable score. In the case of the Weather dataset, the expected BN derived from the PC algorithm did not align with the best structure identified in the landscape analysis. This discrepancy can be elucidated by the BIC score's penalizing mechanism, which discourages the addition of edges unless it significantly enhances data description (CAMPOS, 2006). Consequently, for the specific dataset in question, a model complexity balanced with data description was achieved with just 3 or 4 edges. BNs with 5 or 6 edges incurred higher penalization due to increased complexity, resulting in a lower score compared to the optimal BNs with 3 or 4 edges. An examination of the superior structures was undertaken to discern which structures outperformed the expected ones.

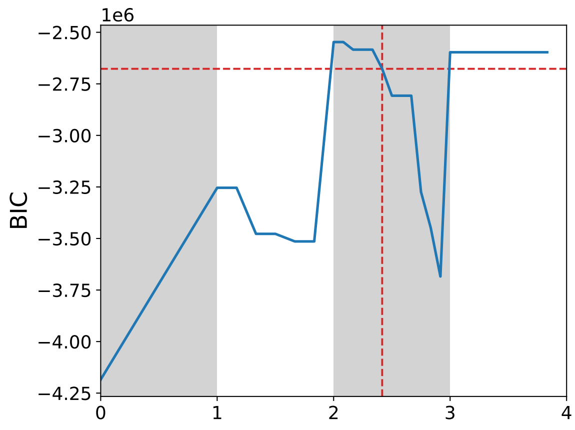
Figure 60 – Location of the expected structure or the structure learned using PC for all datasets. The location is marked by red dashed lines. This landscape was built using the whole dataset and CLBQ was used to quantize the data for each structure.



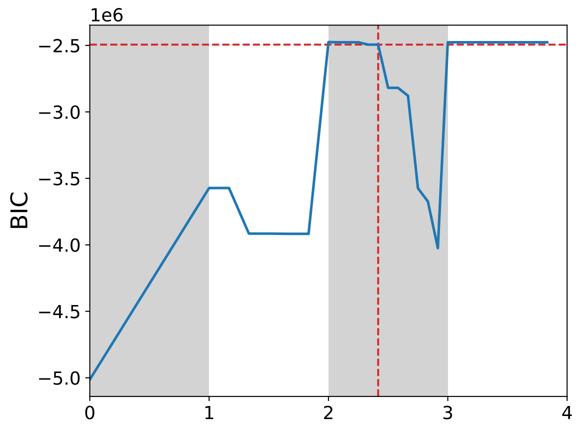
(a) D4



(b) Weather



(c) XYZ



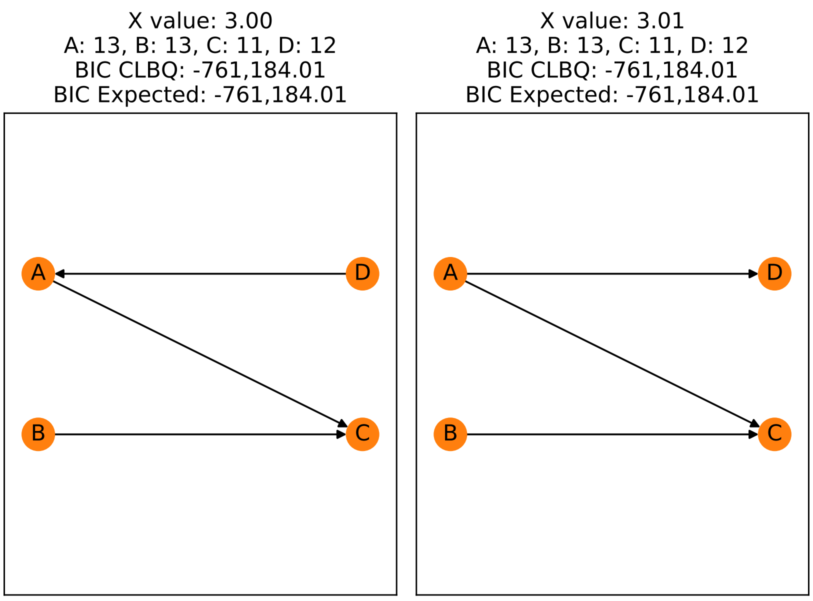
(d) XYZ3

Source: Ribeiro *et al.* (in press)

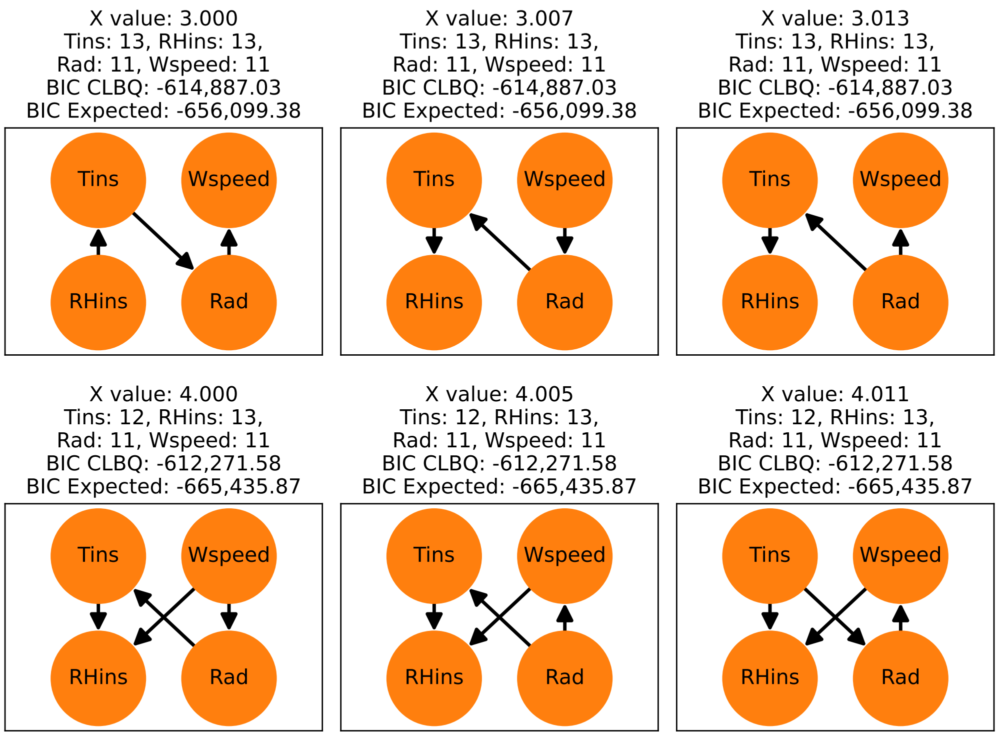
The results of this analysis of the best structures can be seen in Figures 61 and 62. From it, it is evident that for dataset D4, the only BN with a score matching that of the expected structure is the one with a permutation of one of the edges from the expected structure. This underscores the suitability of using CLBQ in the search and score process of BNSL, as the expected structure could be selected given a robust search algorithm. Regarding XYZ and XYZ3, it is observed that, under the same quantization, these structures exhibit either identical or very close scores to those of the expected structure. Notably, the quantization values are remarkably similar, aligning with expectations given the minor limitations on structure and the substantial dataset size. This suggests that subtle variations in variable quantization impact the BIC score, leading to a mix of the best structures. Furthermore, many structures with equal or higher scores are permutations of edges from the expected structure. This is promising, as the direction of edges in a BN does not necessarily follow causal relationships (LUO; ZHAO; DU, 2019). Thus, while it may not achieve the same level as observed for dataset D4, it still indicates the appropriateness of employing CLBQ for BNSL.

Lastly, for the Weather dataset, all the best structures feature edges indicating dependencies present in the PC structure, although with varying orientations. This implies that these connections represent crucial dependencies for data description based on the BIC score. Considering the characteristics of the BIC score and the identified optimal BNs, these results are promising and suggest a favorable potential for using CLBQ in BN structure learning.

Figure 61 – Analysis of BNs with scores higher or equal to the score of the expected BN. X is the numeric value assigned to the structure on the x-axis for the landscape plot. The quantization selected is shown in front of the variables' names. BIC CLBQ is the score found using CLBQ quantization for that structure and BIC Expected is the score found by using the quantization found for the expected structure.


X value: 3.00
A: 13, B: 13, C: 11, D: 12
BIC CLBQ: -761,184.01
BIC Expected: -761,184.01
A
D
B
C
X value: 3.01
A: 13, B: 13, C: 11, D: 12
BIC CLBQ: -761,184.01
BIC Expected: -761,184.01
A
D
B
C

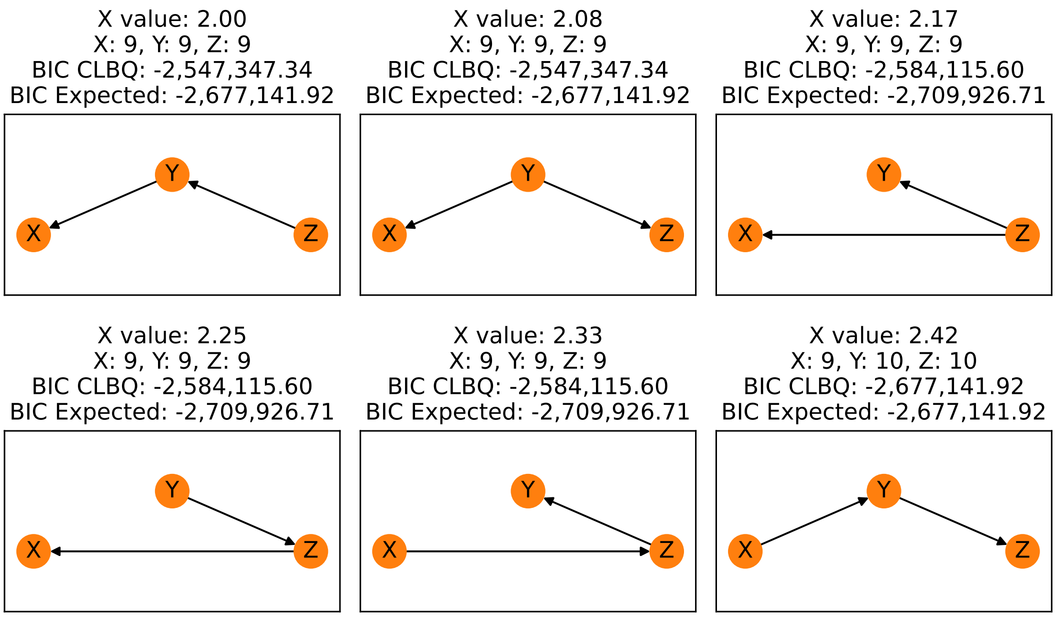
(a) D4



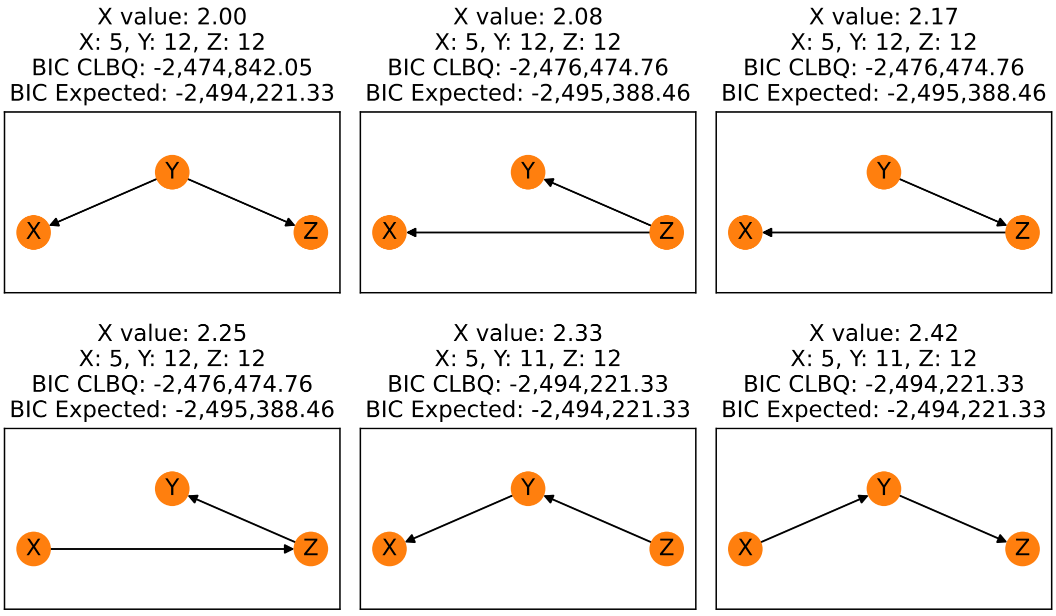
(b) Weather

Source: Ribeiro *et al.* (in press)

Figure 62 – Analysis of BNs with scores higher or equal to the score of the expected BN. X is the numeric value assigned to the structure on the x-axis for the landscape plot. The quantization selected is shown in front of the variables' names. BIC CLBQ is the score found using CLBQ quantization for that structure and BIC Expected is the score found by using the quantization found for the expected structure.



(a) XYZ



(b) XYZ3

Source: Ribeiro *et al.* (in press)

4.14 BNSL Scores Test

The results of the preliminary test conducted with the predictability scores P1, P2, and P3 can be seen in Table 28. From them, we can observe that for both systems, the proposed scores did not show good performance. However, this can be explained by the fact that many times the selected RBs had either no edges or only one edge. This indicates that one possibility is that the Hill Climbing method may not be well-suited for the proposed scores, as it adds one edge at a time. A method that tests already structured RBs may perhaps achieve better results.

Table 28 – Results of the predictability scores test.

System	**Score**	**Edges mean**	**Hit rate** *A*	**Hit rate** *B*	**Hit rate** *C*
1	**K2**	4	1.00	1.00	0.53
	BDeu	3	1.00	1.00	0.53
	P1	0	0.08	0.50	0.14
	P2	0	0.08	0.50	0.14
	P3	2	1.00	0.50	0.52
2	**K2**	3	0.57	0.91	0.57
	BDeu	1	1.00	1.00	0.57
	P1	0	1.00	1.00	0.00
	P2	1	1.00	1.00	0.00
	P3	2	0.57	0.86	0.00

Source: Author

5 CONCLUSIONS

Many aspects and methods for BNSL were covered in this thesis. Considering the proposed algorithm to perform the search and score BNSL AGAVaPS, from its behavior test it was seen how the algorithm performed its search varied from exploration to exploitation. The test with the CEC2017 Benchmark showed that AGAVaPS had a competitive performance against other well-established search algorithms, having a good coverage of the search space without losing the quality of the solution, which could make it a good choice for problems with huge search spaces or where the search space coverage is of interest.

The feature selection test corroborates the suitability of the use of AGAVaPS for problems with huge search spaces, with AGAVaPS being the best-performing algorithm on it. Moreover, the preliminary test of AGAVaPS on BNSL also had AGAVaPS as the best-performing algorithm on it. In addition to that, AGAVaPS presented a very small STD showing that AGAVaPS was capable of finding the best solutions even with different starting conditions.

The parameter analysis of AGAVaPS for BNSL showed that $\gamma = 0.5$ and $\mu_{mut} = 0.5$ were the best parameters to be used for BNSL considering the datasets used, although the difference in performance between the parameters combination was small. In the comparison test of AGAVaPS, TABU, and HC on BNSL it was seen that HC and TABU performed the same for almost all metrics, only having a small difference in execution time. Meaning that no relevant improvement of performance resulted from the use of a tabu list was seen on the tests performed.

Moreover, AGAVaPS performed better for the F1 score, SHD, and BIC score. For the F1 score and SHD, AGAVaPS was the best-ranked algorithm for all tests. For the BIC score, AGAVaPS was ranked first for the test cases with 27 nodes or less. This may be a result of a lack of enough evaluations for AGAVaPS to properly cover the search space. For BSF, all algorithms tied in the mean rank with AGAVaPS performing better for the test cases with a smaller number of nodes.

When considering the execution time, the result was that HC and TABU were faster than AGAVaPS. This result was expected, as TABU and HC are simpler algorithms than AGAVaPS and were projected to be fast. Even though AGAVaPS takes more time to perform the search it is still an interesting algorithm for BNSL because the BNSL is usually only performed once and taking more time to learn a better structure is a trade-off that can be of interest.

On this comparison test, AGAVaPS had a poorer performance than the other algorithms for test cases with a higher number of nodes for two metrics. Thus, future work on AGAVaPS would be to study ways to improve its performance in datasets with a higher number of nodes, considering making changes to parameter values, problem definition, and mutation and reproduction processes. Furthermore, the use of parallelism and code optimization also need to be explored to improve the execution time.

For the BNSL landscape analysis, the differences and convergence of behavior for the scores were observed. The convergence of behavior happens when there is enough data for that, so for the datasets with 3 edges, the scores had similar landscapes, while for the datasets with more variables presented different evaluations of structures between the scores.

When looking at the variation of the landscape due to samples of the dataset and dataset size, it was seen that the samples of a dataset can greatly influence the score landscape, thus having a big dataset is important. The dataset size affects the range of the values of the scores and also introduces differences in the score landscape, which is most likely related to having more different samples in the dataset, and as seen this affects the landscape.

The test varying the quantization of one variable showed that the quantization of one variable can already change the score range and also add small changes to the landscape. When considering a fixed structure and analyzing how the varying quantization of the variables affected the score of the structure, it was seen that in cases where an ideal quantization exists, peaks of the score will occur for that ideal quantization and its multiples. However, when there is not a specific ideal quantization this behavior was not observed.

For both cases, it was observed a tendency to decrease the score resulted from increasing the quantization value. Moreover, for the continuous variables test it was observed that the BIC score component of a parent to a descendant when varying the quantization of the parent improved as the quantization increased until reaching a limit. The quantization method based on this characteristic presented inconsistency based on the quantization value of other variables.

The approach of quantization choice by BNSL using variables of bins, cut-points, and variables quantization also had problems with inconsistency. Specifically, the approach using bins and cut-points did not have the expected result to decide a better quantization, and the one using variables quantization had interesting results, however was not very consistent.

When looking at the MDL Non-Zero score, the score produced the results expected and was capable of identifying points that the existence worsened the score. This capacity to identify outliers is very interesting. The EM BN quantization method presented the interesting capacity of changing points of bins, however, these changes were not big enough to significantly alter the quantization and reduce the number of bins.

Considering the CLBQ tests, for the comparison of quantization using MSE and histogram, CLBQ did not find the ideal quantization for the D4 dataset test case, however, for all cases it found quantizations that had good MSE and modeled well the variables' distribution. CLBQ performed better than DD for all the test cases and showed its capacity to balance data fidelity, model quality, and structure score.

For the landscapes with different dataset sizes, no major variances were observed on the landscapes, showing that the use of CLBQ was resilient to dataset size, maintaining the best networks as the best. This indicated that the use of CLBQ on BNSL is possible.

Regarding the execution time, the CLBQ execution time varies with the dataset size and between datasets. The execution time for one structure is relatively small, however, for its application of BNSL, the trade-off between the added time and the benefit of having the quantization done for each structure needs to be considered.

When considering the location of the expected structure on the landscape, not always the expected structure was the best scoring structure, however, the best structure had the same dependencies in different directions. Since the direction of the dependencies is not an issue for the BN, this indicated that the application of CLBQ on BNSL is possible and would return a good BN.

The limitations of CLBQ detected by the tests are that it does not guarantee to return the ideal quantization and that CLBQ is very dependent on data volume, with a small data volume CLBQ will sacrifice data fidelity in exchange for model quality.

Addressing the predictability scores, the test showed very poor performance of all scores, especially for P1 and P2. In addition to that, the method of evaluation of these scores is quite time-consuming. Thus, there is a need to study other possibilities and formulations for a score based on predictability.

The next steps are to improve the AGAVaPS coding, to improve its execution time. Also, to explore other problems' definitions for the BNSL problem to improve AGAVaPS performance for larger problems with a higher number of variables. Finally, run more extensive tests with the proposed algorithm (AGAVaPS and CLBQ) to further confirm the results obtained and to further analyze their performance.

REFERENCES

ABDELAZIZ, M. Distribution network reconfiguration using a genetic algorithm with varying population size. **Electric Power Systems Research**, v. 142, p. 9–11, 2017. ISSN 0378-7796. Available at: https://www.sciencedirect.com/science/article/pii/S0378779616303273.

ABRAMSON, B. *et al.* Hailfinder: A bayesian system for forecasting severe weather. **International Journal of Forecasting**, v. 12, n. 1, p. 57–71, 1996. ISSN 0169-2070. Probability Judgmental Forecasting. Available at: https://www.sciencedirect.com/science/article/pii/0169207095006648.

ALFONSO, D.; MANJARRÉS, A.; PICKIN, S. Semi-automatic generation of competency maps based on educational data mining. **International Journal of Computational Intelligence Systems**, v. 12, p. 744–760, 2019. ISSN 1875-6883. Available at: https://doi.org/10.2991/ijcis.d.190627.001.

AMIRKHANI, H. *et al.* Exploiting experts' knowledge for structure learning of bayesian networks. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 39, n. 11, p. 2154–2170, Nov 2017. ISSN 0162-8828.

ANKAN, A.; PANDA, A. pgmpy: Probabilistic graphical models using python. *In*: CITESEER. **Proceedings of the 14th Python in Science Conference (SCIPY 2015)**. [*S.l.: s.n.*], 2015.

ARABAS, J.; MICHALEWICZ, Z.; MULAWKA, J. GAVaPS-a genetic algorithm with varying population size. *In*: **Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence**. [*S.l.: s.n.*], 1994. p. 73–78 vol.1.

ASGHARI, F. N.; QIAN, S. S.; STOW, C. A. Comparative analysis of discretization methods in bayesian networks. **Environmental Modelling & Software**, v. 87, p. 64–71, 2017. ISSN 1364-8152.

AWAD, N. H. *et al.* **Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Bound Constrained Real-Parameter Numerical Optimization**. [*S.l.*], 2016.

BÄCK, T.; EIBEN, A. E.; VAART, N. A. L. v. d. An empirical study on GAs "without parameters". *In*: **Proceedings of the 6th International Conference on Parallel Problem Solving from Nature**. Berlin, Heidelberg: Springer-Verlag, 2000. (PPSN VI), p. 315–324. ISBN 3540410562.

BERETTA, S. *et al.* Learning the structure of bayesian networks: A quantitative assessment of the effect of different algorithmic schemes. **Complexity**, 2018. Available at: https://doi.org/10.1155/2018/1591878.

BERTONE, E.; ROUSSO, B. Z.; KUFEJI, D. A probabilistic decision support tool for prediction and management of rainfall-related poor water quality events for a drinking water treatment plant. **Journal of Environmental Management**, v. 332, p. 117209, 2023. ISSN 0301-4797.

BEUZEN, T.; MARSHALL, L.; SPLINTER, K. D. A comparison of methods for discretizing continuous variables in bayesian networks. **Environmental Modelling & Software**, v. 108, p. 61–66, 2018. ISSN 1364-8152.

BHATT, R.; DHALL, A. **Skin Segmentation**. 2012. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5T30C.

BRANKE, J.; ELOMARI, J. Simultaneous tuning of metaheuristic parameters for various computing budgets. *In*: **Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation**. New York, NY, USA: Association for Computing Machinery, 2011. (GECCO '11), p. 263–264. ISBN 9781450306904. Available at: https://doi.org/10.1145/2001858.2002006.

CAMBASI, H. *et al.* Comparison of dynamic bayesian network tools. *In*: **2019 Innovations in Intelligent Systems and Applications Conference (ASYU)**. [*S.l.: s.n.*], 2019. p. 1–6. ISSN null. Available at: https://ieeexplore.ieee.org/document/8946390.

CAMPOS, C. P. de *et al.* Entropy-based pruning for learning bayesian networks using BIC. **Artificial Intelligence**, v. 260, p. 42–50, 2018. ISSN 0004-3702.

CAMPOS, L. M. de. A scoring function for learning bayesian networks based on mutual information and conditional independence tests. **The Journal of Machine Learning Research**, p. 2149–2187, 2006.

CHEN, K. *et al.* An early warning method for highway traffic accidents based on bayesian networks. *In*: **Proceedings of the 2019 International Conference on Artificial Intelligence and Computer Science**. New York, NY, USA: Association for Computing Machinery, 2019. (AICS 2019), p. 111–116. ISBN 9781450371506. Available at: https://doi.org/10.1145/3349341.3349386.

CHEN, Y.-C.; WHEELER, T.; KOCHENDERFER, M. Learning discrete bayesian networks from continuous data. **Journal of Artificial Intelligence Research**, v. 59, 12 2015.

CHIESA, M. *et al.* GARS: Genetic algorithm for the identification of a robust subset of features in high-dimensional datasets. **BMC Bioinformatics**, v. 21, 2020.

CIUNKIEWICZ, P. *et al.* Improved design of bayesian networks for modelling toxicity risk in breast radiotherapy using dynamic discretization. *In*: **2022 International Joint Conference on Neural Networks (IJCNN)**. [*S.l.: s.n.*], 2022. p. 01–08.

CONSTANTINOU, A. C. *et al.* Large-scale empirical validation of bayesian network structure learning algorithms with noisy data. **International Journal of Approximate Reasoning**, v. 131, p. 151–188, 2021. ISSN 0888-613X. Available at: https://www.sciencedirect.com/science/article/pii/S0888613X21000025.

CONTALDI, C.; VAFAEE, F.; NELSON, P. C. Bayesian network hybrid learning using an elite-guided genetic algorithm. **Artificial Intelligence Review**, v. 52, n. 1, p. 245–272, Jun 2019. ISSN 1573-7462.

COOPER, G. F.; HERSKOVITS, E. A bayesian method for the induction of probabilistic networks from data. **Machine Learning**, v. 9, p. 309–347, 1992.

CORREIA, A. H. C.; CAMPOS, C. P. de; GAAG, L. C. van der. An experimental study of prior dependence in bayesian network structure learning. *In*: BOCK, J. D. *et al.* (ed.). **Proceedings of the Eleventh International Symposium on Imprecise Probabilities: Theories and Applications**. Thagaste, Ghent, Belgium: PMLR, 2019. (Proceedings of Machine Learning Research, v. 103), p. 78–81. Available at: http://proceedings.mlr.press/v103/correia19a.html.

CORREIA, A. H. C.; CUSSENS, J.; CAMPOS, C. de. **On Pruning for Score-Based Bayesian Network Structure Learning**. 2020.

CUI, Q. *et al.* Globally-optimal prediction-based adaptive mutation particle swarm optimization. **Information Sciences**, v. 418-419, p. 186 – 217, 2017. ISSN 0020-0255. Available at: http://www.sciencedirect.com/science/article/pii/S0020025517308563.

DERRAC, J. *et al.* A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. **Swarm and Evolutionary Computation**, v. 1, n. 1, p. 3–18, 2011. ISSN 2210-6502. Available at: https://www.sciencedirect.com/science/article/pii/S2210650211000034.

DING, F.; ZHUANG, Y. Distributed bayesian network learning algorithm using storm topology. **International Journal of Grid and Distributed Computing**, v. 11, n. 4, p. 113–126, 2018. Available at: https://www.doi.org/10.14257/ijgdc.2018.11.4.10.

DUDEK, J.; JANIGA, D.; WOJNAROWSKI, P. Optimization of CO2-EOR process management in polish mature reservoirs using smart well technology. **Journal of Petroleum Science and Engineering**, v. 197, p. 108060, 2021. ISSN 0920-4105. Available at: http://www.sciencedirect.com/science/article/pii/S0920410520311153.

FANG, H. *et al.* Discretization of continuous variables in bayesian networks based on matrix decomposition. *In*: **2017 International Conference on Computing Intelligence and Information System (CIIS)**. [*S.l.: s.n.*], 2017. p. 184–187.

FANG, W. *et al.* An efficient bayesian network structure learning algorithm based on structural information. **Swarm and Evolutionary Computation**, v. 76, p. 101224, 2023. ISSN 2210-6502.

FARIS, H. *et al.* EvoloPy: An open-source nature-inspired optimization framework in Python. *In*: . [*S.l.: s.n.*], 2016.

FARIS, H. *et al.* **EvoloPy: An open source nature-inspired optimization toolbox for global optimization in Python**. 2016. https://github.com/7ossam81/EvoloPy. Commit: 5f12bf5b5f54332c91b72069ec5dfc06a192c0c8.

FENG, S.; YANG, Z.; HUANG, M. Hybridizing adaptive biogeography-based optimization with differential evolution for multi-objective optimization problems. **Information**, v. 8, n. 3, 2017. ISSN 2078-2489. Available at: https://www.mdpi.com/2078-2489/8/3/83.

FERNANDES, C.; ROSA, A. A study on non-random mating and varying population size in genetic algorithms using a royal road function. *In*: **Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)**. [*S.l.: s.n.*], 2001. v. 1, p. 60–66 vol. 1.

FERNANDES, C.; TAVARES, R.; ROSA, A. C. NiGAVaPS — outbreeding in genetic algorithms. *In*: **Proceedings of the 2000 ACM Symposium on Applied Computing - Volume 1**. New York, NY, USA: Association for Computing Machinery, 2000. (SAC '00), p. 477–482. ISBN 1581132409. Available at: https://doi.org/10.1145/335603.335917.

FRIEDMAN, N.; GOLDSZMIDT, M. Discretizing continuous attributes while learning bayesian networks. *In*: **Proceedings of the Thirteenth International Conference on International Conference on Machine Learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1996. (ICML'96), p. 157–165. ISBN 1558604197.

GROSS, T. J. *et al.* An analytical threshold for combining bayesian networks. **Knowledge-Based Systems**, v. 175, p. 36 – 49, 2019. ISSN 0950-7051. Available at: http://www.sciencedirect.com/science/article/pii/S0950705119301352.

HABERMAN, S. **Haberman's Survival**. 1999. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5XK51.

HANH, N. T. *et al.* An efficient genetic algorithm for maximizing area coverage in wireless sensor networks. **Information Sciences**, v. 488, p. 58 – 75, 2019. ISSN 0020-0255. Available at: http://www.sciencedirect.com/science/article/pii/S0020025519301823.

HAO, J. *et al.* Transfer learning of bayesian network for measuring qos of virtual machines. **Applied Intelligence**, v. 51, p. 8641–8660, 2021.

HECKERMAN, D.; GEIGER, D.; CHICKERING, D. M. Learning bayesian networks: The combination of knowledge and statistical data. **Machine Learning**, v. 20, p. 197–243, 1995.

HENG-ZHOU, Y.; TAO-SHEN, L. Probability based virtual machines placement for green data center. **International Journal of Database Theory and Application**, v. 9, n. 12, p. 131–140, 2016. ISSN 2005-4270.

JACKSON-BLAKE, L. A. *et al.* Seasonal forecasting of lake water quality and algal bloom risk using a continuous gaussian bayesian network. **Hydrology and Earth System Sciences**, v. 26, n. 12, p. 3103–3124, 2022.

JIANHUA, G.; YUANXIANG, L.; LINGLING, W. Evolutionary algorithm based on degradation mechanism. *In*: **Proceeding of the 11th World Congress on Intelligent Control and Automation**. [*S.l.: s.n.*], 2014. p. 4951–4954.

JOSHI, S. K.; BANSAL, J. C. Parameter tuning for meta-heuristics. **Knowledge-Based Systems**, v. 189, p. 105094, 2020. ISSN 0950-7051. Available at: http://www.sciencedirect.com/science/article/pii/S0950705119304708.

KAZIKOVA, A.; PLUHACEK, M.; SENKERIK, R. Why tuning the control parameters of metaheuristic algorithms is so important for fair comparison? **MENDEL**, v. 26, n. 2, p. 9–16, Dec. 2020. Available at: https://mendel-journal.org/index.php/mendel/article/view/120.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. *In*: **Proceedings of ICNN'95 - International Conference on Neural Networks**. [*S.l.: s.n.*], 1995. v. 4, p. 1942–1948 vol.4.

KOLLER, D.; FRIEDMAN, N. **Probabilistic graphical models: principles and techniques**. [*S.l.: s.n.*]: MIT press, 2009.

KOZLOV, A. V.; KOLLER, D. Nonuniform dynamic discretization in hybrid networks. *In*: **Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. (UAI'97), p. 314–325. ISBN 1558604855.

LAURITZEN, S. L.; SPIEGELHALTER, D. J. Local computations with probabilities on graphical structures and their application to expert systems. **Journal of the Royal Statistical Society. Series B (Methodological)**, [Royal Statistical Society, Wiley], v. 50, n. 2, p. 157–224, 1988. ISSN 00359246. Available at: http://www.jstor.org/stable/2345762.

LI, H.; GUO, H. A hybrid structure learning algorithm for bayesian network using experts' knowledge. **Entropy**, v. 20, n. 8, 2018. Available at: https://www.doi.org/10.3390/e20080620.

LI, M.; LIU, K. Application of intelligent dynamic bayesian network with wavelet analysis for probabilistic prediction of storm track intensity index. **Atmosphere**, v. 9, n. 6, 2018. ISSN 2073-4433. Available at: https://doi.org/10.3390/atmos9060224.

LI, Z. *et al.* DCDG-EA: Dynamic convergence–diversity guided evolutionary algorithm for many-objective optimization. **Expert Systems with Applications**, v. 118, p. 35 – 51, 2019. ISSN 0957-4174. Available at: http://www.sciencedirect.com/science/article/pii/S095741741830602X.

LI, Z. *et al.* Structural learning of bayesian network by multi-population bacterial foraging optimization. *In*: **2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)**. [*S.l.: s.n.*], 2018. p. 441–445. ISSN null. Available at: https://doi.org/10.1109/ITOEC.2018.8740547.

LIMA, M. D. *et al.* Heuristic discretization method for bayesian networks. **Journal of Computer Science**, Science Publications, n. 5, p. 869–878, Jan. 2014.

LIU, H. *et al.* A new hybrid method for learning bayesian networks: Separation and reunion. **Knowledge-Based Systems**, v. 121, p. 185 – 197, 2017. ISSN 0950-7051. Available at: http://www.sciencedirect.com/science/article/pii/S0950705117300412.

LOKRANTZ, A.; GUSTAVSSON, E.; JIRSTRAND, M. Root cause analysis of failures and quality deviations in manufacturing using machine learning. **Procedia CIRP**, v. 72, p. 1057–1062, 2018. ISSN 2212-8271. 51st CIRP Conference on Manufacturing Systems. Available at: https://www.sciencedirect.com/science/article/pii/S2212827118303895.

LÓPEZ-IBÁÑEZ, M. *et al.* The irace package: Iterated racing for automatic algorithm configuration. **Operations Research Perspectives**, v. 3, p. 43 – 58, 2016. ISSN 2214-7160. Available at: http://www.sciencedirect.com/science/article/pii/S2214716015300270.

LUO, G.; ZHAO, B.; DU, S. Causal inference and bayesian network structure learning from nominal data. **Applied Intelligence**, v. 49, p. 253–264, 2019.

LUQUE, G.; ALBA, E. **Parallel genetic algorithms: Theory and real world applications**. [*S.l.: s.n.*]: Springer, 2011. v. 367.

LV, L. *et al.* Multi-objective firefly algorithm based on compensation factor and elite learning. **Future Generation Computer Systems**, v. 91, p. 37 – 47, 2019. ISSN 0167-739X. Available at: http://www.sciencedirect.com/science/article/pii/S0167739X18313955.

LYTVYNENKO, V. *et al.* The use of bayesian methods in the task of localizing the narcotic substances distribution. *In*: **2019 IEEE 14th International Conference on Computer Sciences and Information Technologies (CSIT)**. [*S.l.: s.n.*], 2019. v. 2, p. 60–63. ISSN null. Available at: https://ieeexplore.ieee.org/document/8929835.

LYTVYNENKO, V. *et al.* Development, validation and testing of the bayesian network to evaluate the national law enforcement agencies' work. *In*: **2019 9th International Conference on Advanced Computer Information Technologies (ACIT)**. [*S.l.: s.n.*], 2019. p. 252–256. ISSN null. Available at: https://ieeexplore.ieee.org/abstract/document/8780079.

LYTVYNENKO, V. *et al.* Development, validation and testing of the bayesian network of educational institutions financing. *In*: **2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)**. [*S.l.: s.n.*], 2019. v. 1, p. 412–417. ISSN null. Available at: https://ieeexplore.ieee.org/document/8924307.

MABROUK, A. *et al.* Multivariate cluster-based discretization for bayesian network structure learning. *In*: BEIERLE, C.; DEKHTYAR, A. (ed.). **Scalable Uncertainty Management**. Cham: Springer International Publishing, 2015. p. 155–169. ISBN 978-3-319-23540-0.

MAYFIELD, H. *et al.* Structurally aware discretisation for bayesian networks. *In*: . [*S.l.: s.n.*], 2017.

MOBIN, M. *et al.* A hybrid desirability function approach for tuning parameters in evolutionary optimization algorithms. **Measurement**, v. 114, p. 417 – 427, 2018. ISSN 0263-2241. Available at: http://www.sciencedirect.com/science/article/pii/S0263224117306310.

MONTERO, E.; RIFF, M.-C.; ROJAS-MORALES, N. Tuners review: How crucial are set-up values to find effective parameter values? **Engineering Applications of Artificial Intelligence**, v. 76, p. 108 – 118, 2018. ISSN 0952-1976. Available at: http://www.sciencedirect.com/science/article/pii/S0952197618301878.

MONTI, S.; COOPER, G. F. A multivariate discretization method for learning bayesian networks from mixed data. *In*: **Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998. (UAI'98), p. 404–413. ISBN 155860555X.

NEAPOLITAN, R. **Learning Bayesian Networks**. [*S.l.: s.n.*], 2003. ISBN 9780123704771.

ONG, H. C.; YAU, S. E.; LOOI, J. Y. Y. A bayesian network approach to study undergraduates' brand consciousness. *In*: **2016 International Conference on Multimedia Systems and Signal Processing (ICMSSP)**. [*S.l.: s.n.*], 2016. p. 51–55. ISSN null. Available at: https://www.doi.org/10.1109/ICMSSP.2016.020.

OPPENHEIM, A. V.; SCHAFER, R. W.; BUCK, J. R. **Discrete-Time Signal Processing (2nd Ed.)**. USA: Prentice-Hall, Inc., 1999. ISBN 0137549202.

RAMAZZOTTI, D. *et al.* Efficient computational strategies to learn the structure of probabilistic graphical models of cumulative phenomena. **Journal of Computational Science**, v. 30, p. 1 – 10, 2019. ISSN 1877-7503. Available at: http://www.sciencedirect.com/science/article/pii/S1877750318309967.

REINIS, Z. *et al.* Prognostic significance of the risk profile in the prevention of coronary heart disease. **Bratisl Lek Listy**, v. 76, p. 137–150, 1981.

RIBEIRO, R. R. M. *et al.* Conditional probability table limit based quantization for bayesian networks: model quality, data fidelity and structure score. **Applied Intelligence**, in press.

RIBEIRO, R. R. M.; MACIEL, C. D. AGAVaPS - adaptive genetic algorithm with varying population size. *In*: **2022 IEEE Congress on Evolutionary Computation (CEC)**. [*S.l.: s.n.*], 2022. p. 1–8.

RIBEIRO, R. R. M.; MACIEL, C. D. Bayesian network structural learning using adaptive genetic algorithm with varying population size. **Machine Learning and Knowledge Extraction**, v. 5, n. 4, p. 1877–1887, 2023. ISSN 2504-4990. Available at: https://www.mdpi.com/2504-4990/5/4/90.

ROBINSON, R. W. Counting unlabeled acyclic digraphs. *In*: LITTLE, C. H. C. (ed.). **Combinatorial Mathematics V**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1977. p. 28–43. ISBN 978-3-540-37020-8.

ROHMER, J. Uncertainties in conditional probability tables of discrete bayesian belief networks: A comprehensive review. **Engineering Applications of Artificial Intelligence**, v. 88, p. 103384, 2020. ISSN 0952-1976.

ROPERO, R.; RENOOIJ, S.; GAAG, L. van der. Discretizing environmental data for learning bayesian-network classifiers. **Ecological Modelling**, v. 368, p. 391–403, 2018. ISSN 0304-3800. Available at: https://www.sciencedirect.com/science/article/pii/S0304380016308377.

RU, X. *et al.* Bayesian network parameter learning using constraint-based data extension method. **Applied Intelligence**, v. 53, p. 9958–9977, 2023.

RUIZ-RUANO, A.-M.; PUGA, J. L. Modelling academic entrepreneurial intention with bayesian networks / modelado de la intención emprendedora académica con redes bayesianas. **International Journal of Social Psychology**, Routledge, v. 34, n. 2, p. 383–411, 2019. Available at: https://doi.org/10.1080/02134748.2019.1589783.

SACHS, K. *et al.* Causal protein-signaling networks derived from multiparameter single-cell data. **Science**, v. 308, n. 5721, p. 523–529, 2005. Available at: https://www.science.org/doi/abs/10.1126/science.1105809.

SALGOTRA, R. *et al.* Self adaptive cuckoo search: Analysis and experimentation. **Swarm and Evolutionary Computation**, v. 60, p. 100751, 2021. ISSN 2210-6502. Available at: https://www.sciencedirect.com/science/article/pii/S2210650220304041.

SARI, D. *et al.* Discretization methods for bayesian networks in the case of the earthquake. **Bulletin of Electrical Engineering and Informatics**, v. 10, n. 1, p. 299–307, 2021. ISSN 2302-9285. Available at: https://beei.org/index.php/EEI/article/view/2007.

SCALASSARA, P. R.; MACIEL, C. D.; PEREIRA, J. C. Predictability analysis of voice signals. **IEEE Engineering in Medicine and Biology Magazine**, v. 28, n. 5, p. 30–34, Sep. 2009. ISSN 1937-4186.

SCANAGATTA, M. *et al.* Approximate structure learning for large bayesian networks. **Machine Learning**, v. 107, 09 2018.

SCANAGATTA, M. *et al.* Efficient learning of bounded-treewidth bayesian networks from complete and incomplete data sets. **International Journal of Approximate Reasoning**, v. 95, p. 152 – 166, 2018. ISSN 0888-613X. Available at: http://www.sciencedirect.com/science/article/pii/S0888613X17307272.

SCHWARZ, G. Estimating the Dimension of a Model. **The Annals of Statistics**, Institute of Mathematical Statistics, v. 6, n. 2, p. 461 – 464, 1978. Available at: https://doi.org/10.1214/aos/1176344136.

SCUTARI, M. Learning bayesian networks with the bnlearn R package. **Journal of Statistical Software**, v. 35, n. 3, p. 1–22, 2010.

SEVINC, V.; KUCUK, O.; GOLTAS, M. A bayesian network model for prediction and analysis of possible forest fire causes. **Forest Ecology and Management**, v. 457, p. 117723, 2020. ISSN 0378-1127. Available at: http://www.sciencedirect.com/science/article/pii/S0378112719311776.

SHIOMOTO, K.; OTOSHI, T.; MURATA, M. A novel network traffic prediction method based on a bayesian network model for establishing the relationship between traffic and population. **Annals of Telecommunications**, v. 78, p. 53–70, 2023.

SIPPER, M. *et al.* Investigating the parameter space of evolutionary algorithms. **BioData Mining**, v. 11, p. 2, 2018.

SKAKOV, E. S.; MALYSH, V. N. Parameter meta-optimization of metaheuristics of solving specific NP-hard facility location problem. **Journal of Physics: Conference Series**, IOP Publishing, v. 973, p. 012063, mar 2018. Available at: https://doi.org/10.1088\%2F1742-6596\%2F973\%2F1\%2F012063.

SONG, D. *et al.* Multivariate discretization for bayesian network structure learning in robot grasping. *In*: **2011 IEEE International Conference on Robotics and Automation**. [*S.l.: s.n.*], 2011. p. 1944–1950.

TALVITIE, T.; EGGELING, R.; KOIVISTO, M. Learning bayesian networks with local structure, mixed variables, and exact algorithms. **International Journal of Approximate Reasoning**, v. 115, p. 69–95, 2019. ISSN 0888-613X.

TATSIS, V. A.; PARSOPOULOS, K. E. Dynamic parameter adaptation in metaheuristics using gradient approximation and line search. **Applied Soft Computing**, v. 74, p. 368 – 384, 2019. ISSN 1568-4946. Available at: http://www.sciencedirect.com/science/article/pii/S1568494618305519.

TIAN, T. *et al.* A bayesian network model for prediction of low or failed fertilization in assisted reproductive technology based on a large clinical real-world data. **Reproductive Biology and Endocrinology**, v. 21, p. 8, 2023.

TILLEY, D. **CEC 2017 Python**. 2020. https://github.com/tilleyd/cec2017-py. Commit: a2cf0de6144881dbcfead844278578ccfb32941f.

TOROPOVA, A. V.; TULUPYEVA, T. V. Discretization of a continuous frequency value in a model of socially significant behavior. *In*: **2022 XXV International Conference on Soft Computing and Measurements (SCM)**. [*S.l.: s.n.*], 2022. p. 28–30.

TRINDADE, Á. R.; CAMPELO, F. Tuning metaheuristics by sequential optimisation of regression models. **Applied Soft Computing**, v. 85, p. 105829, 2019. ISSN 1568-4946. Available at: http://www.sciencedirect.com/science/article/pii/S1568494619306106.

TSOULOS, I. G.; TZALLAS, A.; TSALIKAKIS, D. PDoublePop: An implementation of parallel genetic algorithm for function optimization. **Computer Physics Communications**, v. 209, p. 183 – 189, 2016. ISSN 0010-4655. Available at: http://www.sciencedirect.com/science/article/pii/S0010465516302776.

UMBARKAR, A.; JOSHI, M.; HONG, W.-C. Multithreaded parallel dual population genetic algorithm (MPDPGA) for unconstrained function optimizations on multi-core system. **Applied Mathematics and Computation**, v. 243, p. 936 – 949, 2014. ISSN 0096-3003. Available at: http://www.sciencedirect.com/science/article/pii/S0096300314008698.

VARNAMKHASTI, M. J.; HASSAN, N. Neurogenetic algorithm for solving combinatorial engineering problems. **Journal of Applied Mathematics**, v. 2012, 2012.

VASIMUDDIN, M.; CHOCKALINGAM, S. P.; ALURU, S. A parallel algorithm for bayesian network inference using arithmetic circuits. *In*: **Proceedings - 2018 IEEE 32nd International Parallel and Distributed Processing Symposium, IPDPS 2018**. [*S.l.: s.n.*], 2018. p. 34–43.

VILLA, S.; STELLA, F. Learning continuous time bayesian networks in non-stationary domains. *In*: **Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18**. International Joint Conferences on Artificial Intelligence Organization, 2018. p. 5656–5660. Available at: https://doi.org/10.24963/ijcai.2018/804.

VOGEL, A. J.; WILKE, D. N. Spatially distributed statistical significance approach for real parameter tuning with restricted budgets. **Applied Soft Computing**, v. 70, p. 648–664, 2018. ISSN 1568-4946. Available at: https://www.sciencedirect.com/science/article/pii/S1568494618303235.

WANG, Z.; SOBEY, A. A comparative review between genetic algorithm use in composite optimisation and the state-of-the-art in evolutionary computation. **Composite Structures**, v. 233, p. 111739, 2020. ISSN 0263-8223. Available at: http://www.sciencedirect.com/science/article/pii/S0263822319328636.

WILSON, S. F. *et al.* Using bayesian networks to map winter habitat for mountain goats in coastal british columbia, canada. **Frontiers in Environmental Science**, v. 10, 2022. ISSN 2296-665X.

XIA, X. *et al.* An expanded particle swarm optimization based on multi-exemplar and forgetting ability. **Information Sciences**, v. 508, p. 105 – 120, 2020. ISSN 0020-0255. Available at: http://www.sciencedirect.com/science/article/pii/S002002551930814X.

XU, Q. *et al.* Dynamic risk assessment for underground gas storage facilities based on bayesian network. **Journal of Loss Prevention in the Process Industries**, v. 82, p. 104961, 2023. ISSN 0950-4230.

YANG, C. *et al.* Structural learning of bayesian networks by bacterial foraging optimization. **International Journal of Approximate Reasoning**, v. 69, p. 147 – 167, 2016. ISSN 0888-613X. Available at: http://www.sciencedirect.com/science/article/pii/S0888613X15001620.

YANG, X.; DEB, S. Cuckoo search via lévy flights. *In*: **2009 World Congress on Nature Biologically Inspired Computing (NaBIC)**. [*S.l.: s.n.*], 2009. p. 210–214.

YANG, X.-S. Firefly algorithm, stochastic test functions and design optimisation. **International Journal of Bio-inspired Computation**, v. 2, 03 2010.

YANG, X.-S. A new metaheuristic bat-inspired algorithm. *In*: _____. **Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 65–74. ISBN 978-3-642-12538-6. Available at: https://doi.org/10.1007/978-3-642-12538-6_6.

ZHANG, D. *et al.* Connected population synthesis for transportation simulation. **Transportation Research Part C: Emerging Technologies**, v. 103, p. 1 – 16, 2019. ISSN 0968-090X. Available at: http://www.sciencedirect.com/science/article/pii/S0968090X18318515.

ZHOU, A. *et al.* Multiobjective evolutionary algorithms: A survey of the state of the art. **Swarm and Evolutionary Computation**, v. 1, n. 1, p. 32–49, 2011. ISSN 2210-6502. Available at: https://www.sciencedirect.com/science/article/pii/S2210650211000058.

ZHOU, X. *et al.* Enhancing artificial bee colony algorithm with multi-elite guidance. **Information Sciences**, v. 543, p. 242 – 258, 2021. ISSN 0020-0255. Available at: http://www.sciencedirect.com/science/article/pii/S0020025520307003.

ZHU, M.; LIU, S.; JIANG, J. Learning bayesian networks in the space of structures by a hybrid optimization algorithm. **International Journal of Computers Communications & Control**, v. 11, n. 6, p. 889–901, 2016. ISSN 1841-9844. Available at: http://univagora.ro/jour/index.php/ijccc/article/view/2502.

ZWIRGLMAIER, K.; STRAUB, D. A discretization procedure for rare events in bayesian networks. **Reliability Engineering System Safety**, v. 153, p. 96–109, 2016. ISSN 0951-8320. Available at: https://www.sciencedirect.com/science/article/pii/S0951832016300229.

APPENDIX

APPENDIX A – PUBLICATIONS

The following papers resulted from this doctoral research:

- **Journals:**
  - **Accepted:** RIBEIRO, R. R. M.; OLIVEIRA JR., J. N.; CAMPOS, C. P.; MACIEL, C. D. Conditional Probability Table Limit Based Quantization for Bayesian Networks: model quality, data fidelity and structure score. **Applied Intelligence**, Springer, 2023.
  - **Published:** RIBEIRO, R. R. M.; MACIEL, C. D. Bayesian Network Structural Learning Using Adaptive Genetic Algorithm with Varying Population Size. **Machine Learning and Knowledge Extraction**, v. 5, p. 1877-1887, 2023. Available at: https://doi.org/10.3390/make5040090.
- **Conference:**
  - **Published:** RIBEIRO, R. R. M.; MACIEL, C. D. AGAVaPS - Adaptive Genetic Algorithm with Varying Population Size. **2022 IEEE Congress on Evolutionary Computation (CEC)**, 2022. Available at: http://www.doi.org/10.1109/CEC55065.2022.9870394.

The following papers were the results of collaborations and are not directly related to this thesis.

- **Conference:**
  - **Published:** FOGLIATTO, M. S. S.; RIBEIRO, R. R. M.; CAETANO, H. O.; DESUÓ N., L.; MACIEL, C. D. Reducing greenhouse gases emissions aiming a carbon-neutral grid. **Simpósio Brasileiro de Sistemas Elétricos 2023**, 2023.
  - **Published:** FOGLIATTO, M. S. S.; DESUÓ N., L.; RIBEIRO, R. R. M.; SANTOS, T. M. O.; LONDON JR., J. B. A.; BESSANI, M.; MACIEL, C. D. Time to Event Analysis for Failure Causes in Electrical Power Distribution Systems. **Congresso Brasileiro de Automática 2020**, 2020.
  - **Published:** RIBEIRO, R. R. M.; FOGLIATTO, M. S. S.; CAETANO, H. O.; PEREIRA JR.; B. R.; MACIEL, C. D. Metaheuristic Search for Optimum Cost-Benefit Resilience Level by Redundancy Adding. **Congresso Brasileiro de Automática 2020**, 2020.

- **Published:** RIBEIRO, R. R. M.; SANTOS, T. M. O.; GROSS, T.; OLIVEIRA JR., J. N.; TSUKAHARA; V. H. B.; MACIEL, C. D. APPLYING PDC FOR THE RECOGNITION OF FIREARM'S CALIBRE. **Anais do 14º Simpósio Brasileiro de Automação Inteligente**, 2019.
- **Published:** OLIVEIRA JR., J. N.; SANTOS, T. M. O.; RIBEIRO, R. R. M.; ÁVILA, I.; MACIEL, C. D. Entropy: from thermodynamics to signal processing. **Anais do 14º Simpósio Brasileiro de Automação Inteligente**, 2019.

EESC • USP