Computer Methods and Programs in Biomedicine 269 (2025) 108846

Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

journal homepage: https://www.sciencedirect.com/journal/computer-methods-and-programs-in-biomedicine

Exhaustive biclustering driven by self-learning evolutionary approach for biomedical data

Adrián Segura-Ortiz [a,*], Adán José-García [c,d], Laetitia Jourdan [c], José García-Nieto [a,b]

[a] Dept. de Lenguajes y Ciencias de la Computación, ITIS Software, Universidad de Málaga, Málaga, 29071, Spain
[b] Biomedical Research Institute of Málaga (IBIMA), Universidad de Málaga, Málaga, Spain
[c] Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France
[d] Univ. Lille, Inserm, CHU Lille, U1286 - INFINITE - Institute for Translational Research in Inflammation, F-59000 Lille, France

ARTICLE INFO

Keywords:
Biclustering
Evolutionary algorithm
Biomedical domain
Gene co-expression
Multi-objective
Knowledge injection
Parameter self-configuration

ABSTRACT

Background and Objective: Biclustering is a key data analysis technique that identifies submatrices with coherent patterns, widely applied in biomedical fields such as gene co-expression analysis. Despite its importance, in the context of evolutionary algorithms, traditional partial representations in biclustering algorithms face significant limitations, such as redundancy and limited adaptability to domain-specific objectives. This study aims to overcome these challenges by introducing MOEBA-BIO, a new evolutionary biclustering framework for biomedical data.

Methods: MOEBA-BIO is designed as a flexible framework based on the evolutionary metaheuristics scheme. It includes a self-configurator that dynamically adjusts the algorithm's objectives and parameters based on contextual domain knowledge. The framework employs a complete representation, enabling the integration of new domain-specific objectives and the self-determination of the number of biclusters, addressing the limitations of traditional representations. The source code is available through the following git repository: https://github.com/AdrianSeguraOrtiz/MOEBA-BIO.

Results: Experimental results demonstrate that MOEBA-BIO overcomes the limitations of classical partial representations. Furthermore, its application to simulated and real-world gene expression datasets highlights its ability to specialize in specific biological domains, improving accuracy and functional enrichment of biclusters compared to other state-of-the-art techniques.

Conclusions: MOEBA-BIO represents a significant advancement in biclustering applied to bioinformatics. Its innovative framework, combining adaptability, self-configuration, and integration of domain-specific objectives, addresses the main limitations of traditional methods and offers robust solutions for complex biomedical datasets.

1. Introduction

Biclustering is a data analysis technique that aims to identify subsets of rows and columns in a data matrix that exhibit simultaneous correlation patterns [1]. Unlike traditional clustering, biclustering allows grouping in two dimensions, a feature that has proven highly useful in biomedical data [2].

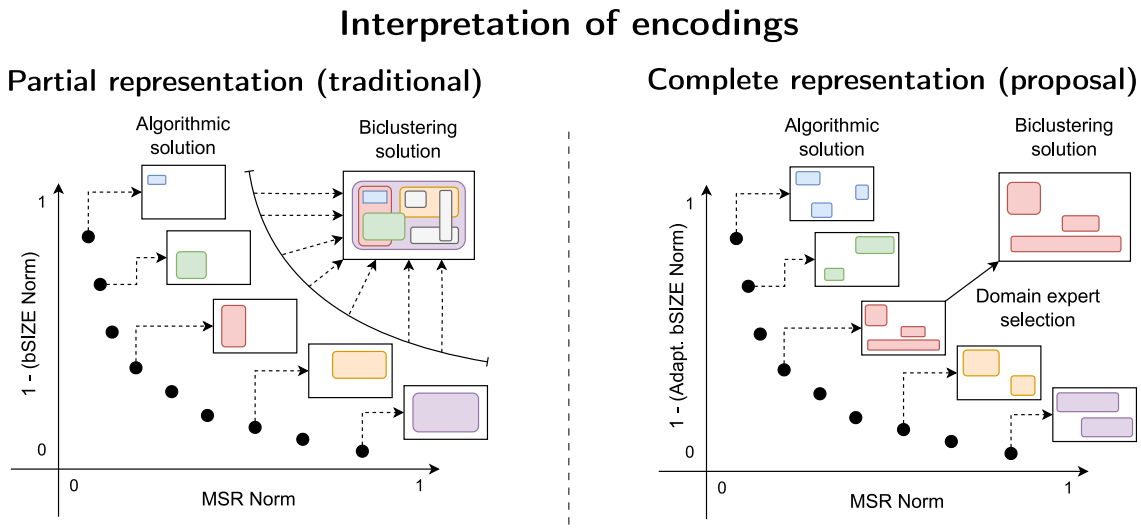
Among the several applications of biclustering in this field, its use in gene expression data stands out, as it allows the identification of gene subsets that exhibit co-expression patterns under certain conditions [3–5]. Another application of biclustering has been shown in the analysis of clinical data, where it helps to identify subgroups of patients with similar biomarker profiles [6–8], which has direct implications for personalized medicine. Finally, its use in medical imaging is also noteworthy, aiding in the detection of patterns of interest in images such as MRIs or CT scans [9,10].

Biclustering aims to group subsets of rows and columns whose cells exhibit a certain coherence in their values. This coherence is determined by the type of pattern being sought in the data (constant, additive coherence, multiplicative coherence, etc.), which depends on the data context and the intended interpretation of the results. A final solution to the biclustering problem is composed of multiple biclusters, whose rows and columns may be shared with other biclusters depending on the overlap constraints.

The biclustering problem has been addressed over the years through various techniques, known for their diversity [11]. However, due to the complexity of optimal biclustering (NP-Complete) [1] and the

* Corresponding author.
E-mail address: adrianseor.99@uma.es (A. Segura-Ortiz).

https://doi.org/10.1016/j.cmpb.2025.108846
Received 7 January 2025; Received in revised form 24 April 2025; Accepted 10 May 2025
Available online 29 May 2025



(a) For the partial representation an individual of the algorithm represents a single bicluster. This means that the solution of the real biclustering problem is obtained from the union of the algorithmic solutions of the front, leading to redundancies, quality heterogeneity, loss of overlap control and exclusion of learning of the number of biclusters.

(b) In the complete representation, each individual in the population represents a possible real solution to the biclustering problem. Therefore, in this case there is a direct equivalence between the algorithmic and the real solution, which allows a better injection of contextual knowledge and overcomes the limitations imposed by traditional encoding.

Fig. 1. Interpretation of each coding of individuals in the approximate Pareto front obtained by the algorithm.

need to satisfy multiple objectives, metaheuristics have gained popularity in solving this problem [12,13]. Specifically, they have found special applicability in the biomedical field for the analysis of gene coexpression [14,15], where the high exploration capacity of the solution space and the optimization of multiple criteria constitute key qualities of this approach, yielding positive results in this context.

Population-based metaheuristics are algorithms that follow an iterative evolutionary process inspired by biological principles [16]. In each iteration or generation, a set of candidate solutions (individuals) undergoes selection, crossover, and mutation to explore the solution space and enhance the population's overall quality. Through this evolutionary process, solutions are progressively refined towards meeting multiple objectives, aiming to balance exploration (searching new areas of the solution space) and exploitation (focusing on promising solutions). This approach allows the algorithm to produce diverse solutions that can effectively represent different trade-offs among objectives, leading to an approximate Pareto front in the final generation. In the biclustering problem, depending on the encoding, this front can have a very different interpretation.

Most of the proposals based on metaheuristics have used integer or binary encoding in their representation, where each individual represents one bicluster by specifying the rows and columns it comprises [12]. From the perspective of the biomedical domain and knowledge injection, this partial codification of individuals presents two main drawbacks:

- **Distortion of reality**: In current approaches, an individual does not represent a possible real solution to the problem, but rather a part of it. Traditionally, the solution of the problem has been taken as the approximate Pareto front obtained by the algorithm (see Fig. 1(a) for a better understanding), which brings two additional issues: subjective post-processing of the algorithm's solution is required, and within the final solution, there are biclusters specialized in certain objectives that belong to the front at the expense of not satisfying the others. This can lead to large biclusters with little cohesion between their data and vice versa [17].
- **Learning limitation**: The fitness of an individual can only be measured by the individual qualities of the bicluster it represents, without considering other fundamental aspects such as the global distribution of all possible biclusters, differentiation between them, or the overall coverage of the original data matrix. This limitation is even more significant if the problem is focused on specific domains, where this lack of perspective prevents the implementation of objectives that go beyond the general purpose and go deeper into particular aspects of the biological domain of the data.

In related fields where evolutionary algorithms are also employed to solve bioinformatics problems, recent publications have demonstrated that injecting domain-specific biomedical knowledge can significantly improve the accuracy of the results [18,19]. Specifically, in the closely related field of gene regulatory network inference, MO-GENECI [20] has proven this by designing new objectives related to the topology and regulatory patterns of such networks.

This article presents MOEBA-BIO (Multi-Objective Evolutionary Biclustering Algorithm for BIOmedical applications), a new biclustering framework designed to make better use of biological domain knowledge in order to maximize learning during the algorithm's execution and extend its capabilities to areas not addressed by other approaches, such as parameter self-configuration and the self-determination of the number of biclusters.

The designed framework proposes a new broader-perspective encoding in which each individual represents a complete set of biclusters equivalent to a final solution to the problem (see Fig. 1(b) for a better understanding). In this codification, the number of biclusters is not predefined and becomes part of the algorithm's learning process. Unlike previous approaches, this representation deletes the need for subjective post-processing or stochastic combinations of partial solutions.

From a general application perspective, the proposed representation in MOEBA-BIO opens the door to the integration of global objective functions that evaluate aspects such as the distribution and differentiation of the biclusters as a whole, rather than just individual qualities. This is particularly useful in biomedical problems, where interpreting

Algorithm 1 Self-configuring scheme of evolutionary metaheuristics offered by MOEBA-BIO.

Input Problem p, Objectives o, Max evaluations $maxEvals$
Output Pareto front approximation $front$

```
population ← generate(p)
evaluations ← 0
while evaluations < maxEvals do
    evaluated ← evaluate(population, o)
    selected ← select(evaluated)
    offspring ← crossover(selected)
    mutated ← mutate(offspring)
    population ← update(population, mutated)
    evaluations ← evaluations + |mutated|
front ← nonDominated(population)
return front
```

the results requires a more holistic data structure perspective. Moreover, when biclustering is focused on a specific problem, it allows for the inclusion of objectives that leverage this global vision of the problem and address domain-specific qualities, enabling a more in-depth analysis tailored to the nature of the data.

MOEBA-BIO implements the evolutionary metaheuristics scheme by making available a wide range of variants and configurations, allowing the context-driven self-design of complex algorithmic proposals. This scheme, compatible with all the algorithms considered and selectable within the framework, is detailed in Algorithm 1. Each phase of this scheme has multiple options in MOEBA-BIO, made selectable through a clear hierarchy of parameters and subparameters, detailed in subsequent sections.

The hypothesis of this study is that the new complete encoding, along with its integration into the designed framework, will enable an expansion of contextual learning that allows for the integration of more realistic and domain-specific objectives, self-determination of the number of biclusters, and the ability to automatically construct new algorithmic designs in biclustering specialized for the exact biological domain of the data.

The contributions to this article are:

1. **New biclustering framework for biomedical data**: MOEBA-BIO stands out in the current state of the art as the first framework for designing evolutionary biclustering algorithms specialized in the biomedical field. It incorporates up to seven well-known multi-objective meta-heuristics (including their subparameters), multiple biclustering objectives, various crossover and mutation operators, the two studied encodings (the traditional and the proposed one), and a wide range of observers that enable accurate monitoring of the evolutionary process throughout execution.
2. **New codification with a holistic perspective**: The integrated encoding within the framework provides a more realistic perspective on the problem by directly incorporating domain-specific biological knowledge.
3. **Self-learning of the number of biclusters**: Context-guided self-determination of solution size, free from post-processing and redundancies.
4. **Context-driven automatic design of algorithmic proposals**: Implementation of a sophisticated self-configurator that not only adjusts traditional parameters based on technical metrics like hypervolume [21], but also employs supervised metrics directly related to the application domain. This enables the selection of metrics tailored to the data context, ensuring that the objectives and their self-configured subparameters are correctly aligned with the particularities of the dataset. In this way, MOEBA-BIO is pre-configured using representative academic data, ensuring that the resulting configuration is adapted for execution on real-world data. Thus, self-configured parameters accurately and confidently reflect biomedical domain knowledge, providing precise and reliable tuning for real-world problems.
5. **New objectives of global perspective and general purpose**: To showcase the potential of the complete encoding even in a general context, this study proposes two new objectives, previously unattainable with traditional encoding. These objectives utilize knowledge from other biclusters within the individual to enhance the global coherence of the solution: *Adaptive bicluster size* (Adaptive bSIZE) and *Bicluster differentiation* (bDIFF).
6. **New holistic objective for gene co-expression**: The validity of this proposal is also demonstrated in the specific application domain of gene co-expression. For this purpose, a new fitness function based on the modularity of the individual concerning the gene regulatory network inferred by domain-specific tools is integrated: *Regulatory coherence*.

In addition to the previous contributions, the implementation style of this project, based on modularity and interoperability, allows for easy extension of objectives, encodings, and operators. This flexibility opens the door to specialization in other domains and data types with greater heterogeneity.

The article is organized into five main sections. Section 2 reviews the current efforts and proposals in the literature to solve the biclustering problem using multi-objective evolutionary algorithms. Section 3 introduces the algorithmic approach proposed in this study, explaining the newly designed representation, the self-configurator, and the objectives. Section 4 details the different phases of experimentation undertaken. Section 5 analyzes the experimental results, comparing the accuracy of this approach with the traditional representation and other well-known biclustering techniques. Section 6 covers the discussion of the study and highlights the implications of the above results for the current state-of-the-art. The final Section 7 concludes the study, highlighting its contributions to biomedicine and possible future work.

2. State of the art

Among the traditional approaches to address the biclustering problem, greedy algorithms stand out, where a set of well-known proposals include Cheng and Church's Algorithm (CCA) [22], Order-Preserving Submatrix (OPSM) [23], Conserved Gene Expression Motifs (xMOTIFs) [24], Iterative Signature Algorithm (ISA) [25], and Large Average Submatrices (LAS) [26]. Other traditional approaches include divide-and-conquer methods, such as the Binary Inclusion-Maximal Biclustering Algorithm (Bimax) [27], and exhaustive enumeration-based algorithms, such as the Bit-Pattern Biclustering Algorithm (BiBit) [28]. Finally, algorithms focused on identifying distribution parameters, such as Plaid [29] and Spectral [30], have also been applied to this problem, aiming to fit statistical models that explain the underlying structure of the data.

From a different viewpoint, multi-objective evolutionary algorithms have become a trend for providing a solution to this problem [14,15]. However, these proposals have a fundamental limitation related to the representation used to encode individuals. Most state-of-the-art methods employ a partial representation, where each individual corresponds to a single bicluster. This hinders a direct correspondence between the solutions obtained by the algorithm and the complete solutions of the actual problem.

To achieve this partial representation of a single bicluster, the scientific community has adopted two main approaches. On one hand, studies such as [14,31,32] have opted for a binary encoding, where the vector is divided into two segments: the first segment uses binary values to indicate which rows belong to the bicluster, while the second follows the same principle to determine the included columns. On the other hand, other works such as [15,33,34] implement integer-based encoding in their algorithms, where the vector has a variable length

Table 1
Comparison between the proposals with non-traditional encodings and the proposal of this work.

Issue	Aspect	BI-MOCK	PBD-SPEA2	BiClustSMEA	MOEBA-BIO
Encoding	Multiple biclusters	Yes	Yes	Yes	Yes
	Variable number of biclusters	Yes	No	Yes	Yes
	Self-learning of quantity	Yes	No	No	Yes
	Real equivalence	No	No	Yes	Yes
Objectives	Individual objectives	Yes	Yes	Yes	Yes
	Global objectives	No	No	No	Yes
	Domain-specific objectives	No	No	No	Yes
	Penalizing heterogeneity between biclusters	No	No	No	Yes
Applicability	Free search space design	No	No	No	Yes
	Self-configuration	No	No	No	Yes
	Open source	No	No	No	Yes

and starts by listing the indices of the bicluster's rows, followed by the indices of its columns.

The major limitations of these partial encodings have been previously discussed in the introduction of this work: distortion of reality and learning constraints. This led to the emergence of new representations that provided a more global perspective, meaning encodings where the individual represents a set of biclusters. Although these approaches aimed to overcome the aforementioned limitations, they did so only partially.

2.1. Alternative representations

In the literature, three main proposals have been identified whose encodings differ from the traditional approach and consider the inclusion of multiple biclusters within a single individual.

First, BI-MOCK [35] uses a representation where the specification of interactions between genes, given by the value of a position and the position itself, forms gene groups. This makes the number of biclusters variable and part of the algorithm's convergence behavior. However, in each individual, all biclusters share the same columns, which again eliminates the direct equivalence between the individual and the actual solution to the problem. As a result, neither global nor domain-specific functions can be included. In fact, only two traditional objectives, applicable individually to biclusters, are considered: Bicluster Size (bSIZE) and Mean Squared Residue (MSR). For each objective, the individual's score is given by the average of its biclusters, meaning there is no penalty for having biclusters with heterogeneous fitness values (poor biclusters can exist as long as there is a very good one to balance the average).

Second, PBD-SPEA2 [36] also uses a representation that allows the inclusion of multiple biclusters in a single individual. However, the number of biclusters must be specified, meaning it is not part of the learning process. Moreover, the final population is formed through a stochastic post-processing step where biclusters from different individuals are combined, so once again, the solution to the problem does not correspond to a specific individual in the population.

Lastly, BiClustSMEA [37] is a hybrid algorithm that, in addition to its evolutionary component, incorporates self-organizing maps in its implementation. Its complex representation, based on gene and condition centroids, allows an individual to represent multiple biclusters. However, although the user does not specify the number of biclusters, it is generated during execution using a random variable. The resulting number of biclusters affects the formation of the representation and never becomes part of the evolutionary algorithm's learning process. Moreover, similar to BI-MOCK, it uses individual objective functions generalized through the arithmetic mean.

Table 1 presents a summary of the mentioned characteristics for each proposal in comparison with MOEBA-BIO. As can be observed, beyond the aspects related to representation, MOEBA-BIO also introduces additional contributions.

On one hand, the objective space of MOEBA-BIO is not limited to traditional fitness functions such as bSIZE or MSR, which evaluate the quality of each bicluster independently. Instead, MOEBA-BIO extends the traditional objective space by incorporating new functions with a global perspective as well as others specialized in the biological domain of the data. Moreover, to avoid promoting diversity in the quality of the biclusters within the same individual, MOEBA-BIO introduces a penalty system, replacing the traditional arithmetic mean with the harmonic or geometric mean.

On the other hand, unlike the other proposals, MOEBA-BIO is not a closed-specific algorithm but rather a flexible open-source framework. This means that researchers have complete freedom to implement new objectives or use predefined ones to design the search space that best suits their data. Additionally, in cases of uncertainty regarding the selection of certain parameters or objectives, MOEBA-BIO includes an integrated self-configurator in its implementation.

Finally, it is worth mentioning that the lack of publicly available repositories hosting the software of these proposals has prevented an experimental comparison beyond the theoretical level.

3. Methods

Before presenting the methodological proposal, it is useful to briefly introduce some key concepts on which this work is based, especially to facilitate its understanding and support later aspects of the methodological design:

- Multi-objective Evolutionary Algorithm (MOEA): An optimization algorithm inspired by natural evolution, designed to search for solutions to complex problems where multiple criteria must be optimized simultaneously (e.g., maximizing the quality and internal coherence of biclusters).
- Pareto dominance and Pareto front: In this context, a solution A is said to *dominate* a solution B if A is equal to or better than B in all objectives and strictly better in at least one. The set of non-dominated solutions forms the so-called *Pareto front*, which represents the best possible trade-offs among the different objectives.
- Hypervolume: A metric used to evaluate the overall quality of a set of solutions. It measures the volume of the objective space covered by the solutions on the Pareto front, relative to a reference point. The larger this volume, the greater the diversity and quality of the solutions.
- Encoding or representation: The way each solution is structured within the algorithm. This work considers two strategies:
  - A traditional *partial* representation, where each individual represents a single bicluster.
  - A novel *complete* representation, where each individual encodes a full solution to the problem, composed of multiple biclusters.

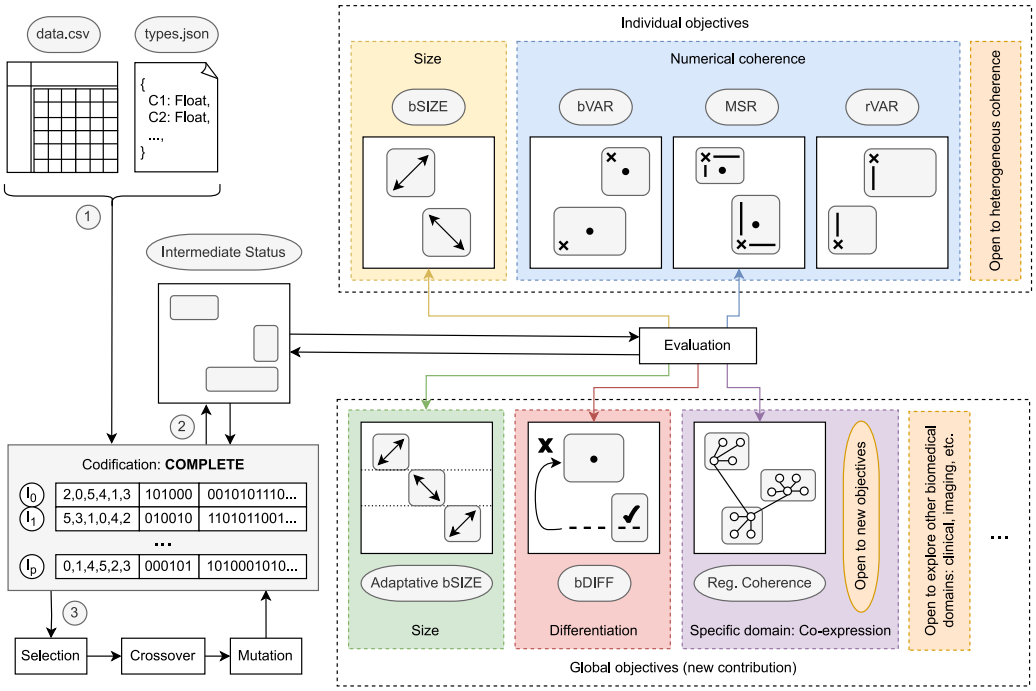


Fig. 2. The basic framework of MOEBA-BIO outlines its main phases and contributions. It starts with the input of the data matrix and the specification of the type of each column (1). Based on the selected encoding strategy (in this case, COMPLETE), MOEBA-BIO initializes the population for the evolutionary algorithm. The execution then proceeds through an iterative process until the stopping condition, defined as reaching a maximum number of evaluations, is met. Each individual is translated into an intermediate representation common to all encodings (2), enhancing interpretability and facilitating evaluation. Individuals are assessed according to the selected objective functions, with normalized fitness scores ranging from 0 (best case) to 1 (worst case). These scores are stored in the original individual, after which the standard evolutionary steps (selection, crossover, and mutation) are applied (3), depending on the chosen algorithm and the operators defined during parameter configuration.

Fig. 2 illustrates how MOEBA-BIO is a flexible multi-objective template offering a wide array of configuration options, enabling the generation of evolutionary biclustering algorithms specialized for specific biological domains. This structure supports a broad selection of meta-heuristics, objectives, and operators, all selectable through a clear hierarchy of parameters and subparameters. With its global perspective, MOEBA-BIO facilitates the creation of complex algorithmic solutions, where the injection of domain knowledge leads to complete solutions with autonomous determination of the number of biclusters.

The first version of the framework presented in this work only includes objectives for solving biclustering problems on numerical data. However, its implementation has been left open to facilitate the future incorporation of objectives associated with heterogeneous data. In fact, from the initial version, users are required to specify the data type stored in each column of the input matrix. This information is currently provided to all the objective functions implemented in this framework.[1]

Since the objective functions included in the framework are freely chosen and configured, MOEBA-BIO does not specialize in detecting any specific pattern in the data. However, in the first phase of experimentation, the goal is to detect constant biclusters using the most traditional functions to validate the new representation.

Additionally, it is worth mentioning that numerical data are pre-normalized to ensure the normalization of the objective functions, allowing the integration of any multi-objective meta-heuristics (including MOEA/D [38]). Moreover, overlapping is avoided in one of the two dimensions, which is a common constraint in biclustering proposals [6,22,27], enabling a more focused encoding design.

MOEBA-BIO is a framework built on top of jMetal [39], extended with custom parameters, encodings, operators, and objective functions specifically designed for biclustering problems in biomedical applications. Additionally, several observers have been included to facilitate the experimental tracking of important aspects of this problem, such as the number of biclusters.[2] Moreover, the framework's functionality has been extended with the implementation of a specific self-configurator for this new environment.[3]

All of these elements are selectable in the tool's configuration, including the optimization algorithm to use and traditional parameters, such as crossover probability, mutation probability, population size, etc. Given this wide range of possibilities, Table 2 lists all the top-level configurable parameters in MOEBA-BIO (without delving into subconfigurations due to space limitations). Following a general overview of these parameters, subsequent subsections focus on the most critical ones or those where this framework has significantly contributed to better highlighting its applicability to biclustering on biomedical data.

As shown in Table 2, the only two mandatory arguments are the input data matrix in CSV format and a complementary JSON file specifying the data type stored in each column (for future non-numerical implementations). The traditional representation, called "PARTIAL", was initially implemented for experimental purposes and remains available alongside the new "COMPLETE" representation introduced in this work, which is explained in the following subsections. The complete representation includes two optional parameters that define the range for the number of biclusters in the algorithm's initial population. By default, this range is set to a reasonable range between 5% and 25% of the total number of rows. While this range is not fixed during execution, as the number of biclusters in individuals can vary throughout the evolutionary process, it does serve as an initial seed for the genetic content.

[1] Java class available in: https://github.com/AdrianSeguraOrtiz/MOEBA-BIO/blob/main/src/main/java/moeba/fitnessfunction/FitnessFunction.java.

[2] Java class available in: https://github.com/AdrianSeguraOrtiz/MOEBA-BIO/blob/main/src/main/java/moeba/utils/observer/impl/BiclusterCountObserver.java.

[3] Java class available in: https://github.com/AdrianSeguraOrtiz/MOEBA-BIO/blob/main/src/main/java/moeba/parameterization/ParameterizationRunner.java.

Table 2
Configurable parameters of MOEBA-BIO framework.

Parameter	Description
`-input -dataset`	Path to the input CSV dataset for biclustering. **Default value:** N/A
`-input -column -types`	Path to the input JSON file specifying the column names and their data types. **Default value:** N/A
`-representation`	Type of representation to use. **Valid values:** COMPLETE, PARTIAL. **Default value:** COMPLETE
`-complete -initial -min -num -bics`	Initial minimum number of biclusters (only for COMPLETE representation). **Valid values:** Integer. **Default value:** 5% of the number of rows
`-complete -initial -max -num -bics`	Initial maximum number of biclusters (only for COMPLETE representation). **Valid values:** Integer. **Default value:** 25% of the number of rows
`-str -fitness -functions`	Fitness objectives to optimize, separated by semicolons. Sub-parameters can be specified in brackets after the identifier. Objectives marked with an * are only available for the COMPLETE representation. **Valid values:** BiclusterSizeNormComp, BiclusterVarianceNorm, RowVarianceNormComp, MeanSquaredResidueNorm, BiclusterSizeNumBicsNormComp*, DistanceBetweenBiclustersNormComp*, RegulatoryCoherenceNormComp*. **Default value:** BiclusterSizeNormComp; MeanSquaredResidueNorm
`-summarize -individual -objectives`	Method to summarize the overall solution quality from the individual bicluster qualities. Applicable only to COMPLETE. **Valid values:** Mean, HarmonicMean, GeometricMean. **Default value:** HarmonicMean
`-population -size`	Population size. **Valid values:** Integer. **Default value:** 500
`-max -evaluations`	Maximum number of evaluations. **Valid values:** Integer. **Default value:** 150000
`-str -algorithm`	Algorithm to use. Sub-parameters can be specified in brackets after the identifier. **Valid values (Single-objective):** GA-AsyncParallel, GA-SingleThread. **Valid values (Multi-objective):** NSGAII-AsyncParallel, NSGAII-SingleThread, MOEAD-SingleThread, MOCell-SingleThread, SPEA2-SingleThread, IBEA-SingleThread, NSGAIII-SingleThread, MOSA-SingleThread. **Default value:** NSGAII-AsyncParallel
`-crossover -probability`	Crossover probability. **Valid values:** Decimal between 0 and 1. **Default value:** 0.9
`-mutation -probability`	Mutation probability. If a progressive mutation is desired, specify a range (e.g. 0.3->0.05). **Valid values:** Decimal or range. **Default value:** 0.1
`-crossover -operator`	Crossover operator. The operator chosen depends on the representation type. Sub-parameters can be specified in brackets after identifier. **Valid interfaces combination (COMPLETE):** RowPermutationCrossover; BiclusterBinaryCrossover; CellBinaryCrossover. **Valid interfaces combination (PARTIAL):** RowColBinaryCrossover. **Default value (COMPLETE):** PartiallyMappedCrossover; BicUniformCrossover; CellUniformCrossover
`-mutation -operator`	Mutation operator. The operator chosen depends on the representation type. Sub-parameters can be specified in brackets after the identifier. **Valid interfaces combination (COMPLETE):** RowPermutationMutation; BiclusterBinaryMutation;CellBinaryMutation. **Valid interfaces combination (PARTIAL):** RowColBinaryMutation **Default value (COMPLETE):** SwapMutation; BicUniformMutation; CellUniformMutation
`-observers`	List of observers separated by semicolons. **Valid values:** BiclusterCountObserver, FitnessEvolutionMinObserver, FitnessEvolutionAvgObserver, FitnessEvolutionMaxObserver, NumEvaluationsObserver. **Default value:** BiclusterCountObserver; FitnessEvolutionMinObserver; NumEvaluationsObserver
`-num -threads`	Number of threads to use. **Valid values:** Integer. **Default value:** Number of available processors
`-output -folder`	Output folder path. **Default value:** N/A

The number and combination of objective functions are entirely flexible. On the one hand, functions specific to the complete representation, where there is a global perspective, cannot be used if the partial encoding is selected. On the other hand, and related to the next parameter in Table 2, traditional objectives aimed at evaluating a single bicluster can be used in the complete encoding, as long as a joint quality summary strategy is specified. Available strategies include the arithmetic mean, geometric mean, and harmonic mean. These strategies penalize individual quality heterogeneity to a lesser or greater degree, respectively. The aim is to prevent a solution from containing biclusters whose overall quality is good solely due to each individual's effort to excel at a specific objective (another flaw of the traditional representation).

The algorithms available in MOEBA-BIO are diverse but have been previously used for biclustering purposes [12]. In concrete, it considers: NSGAII [40], NSGAIII [41], MOEA/D [38], MOCell [42], SPEA2 [43], IBEA [44], and MOSA [45]. Other well-known algorithms, such as SMPSO [46], had to be discarded due to their incompatibility with the designed representation, as it does not follow the evolutionary scheme.

Regarding crossover and mutation probabilities, it is worth mentioning that a progressive mutation option has been implemented in addition to providing static values. This common practice favors exploration in the early stages of the evolutionary algorithm and subsequent exploitation in later stages of higher convergence.

As for the crossover and mutation operators, the decision was made to pre-establish them in the simplest way possible to avoid interfering with subsequent comparisons of encodings. To this end, the most standard operators for the nature of each part of the encodings were chosen. On the one hand, the partial representation consists of two binary encoding sections, one for rows and one for columns. Therefore, the most common approach is to use the most popular binary crossover (uniform crossover) and the most established binary mutation operator (uniform mutation). On the other hand, something similar has been implemented for the complete encoding. Although this will be explained in more detail in the following subsection, it should be mentioned that the complete encoding is divided into three parts: one permutation and two binary sections. Therefore, the same operators as before are used for the binary sections, while for the permutation, Partially Mapped Crossover and Swap mutation are employed. However, the implementation of new operators is open thanks to a sophisticated interface system, allowing the design of domain-specific operators that can also inject knowledge during the evolution of generations.

Additionally, it is worth noting that a list of observers is available for those who wish to log various aspects of the evolutionary history during execution. If an algorithm supporting multi-threaded execution is specified, limiting CPU usage during the run will also be possible.

Finally, regarding second-level parameters, it should be mentioned that, as explained in several sections of Table 2, they pertain to specific configurations of certain first-level options that can be specified directly in parentheses. The most significant cases are algorithm-specific parameters, parameters for certain objective functions, or parameters related to crossover and mutation operators. All of these will be considered during the self-configuration phase.

3.1. Representation

MOEBA-BIO introduces a new global perspective encoding, that allow to establish a direct equivalence between the algorithmic individual and a biclustering solution to the real problem. This is achieved

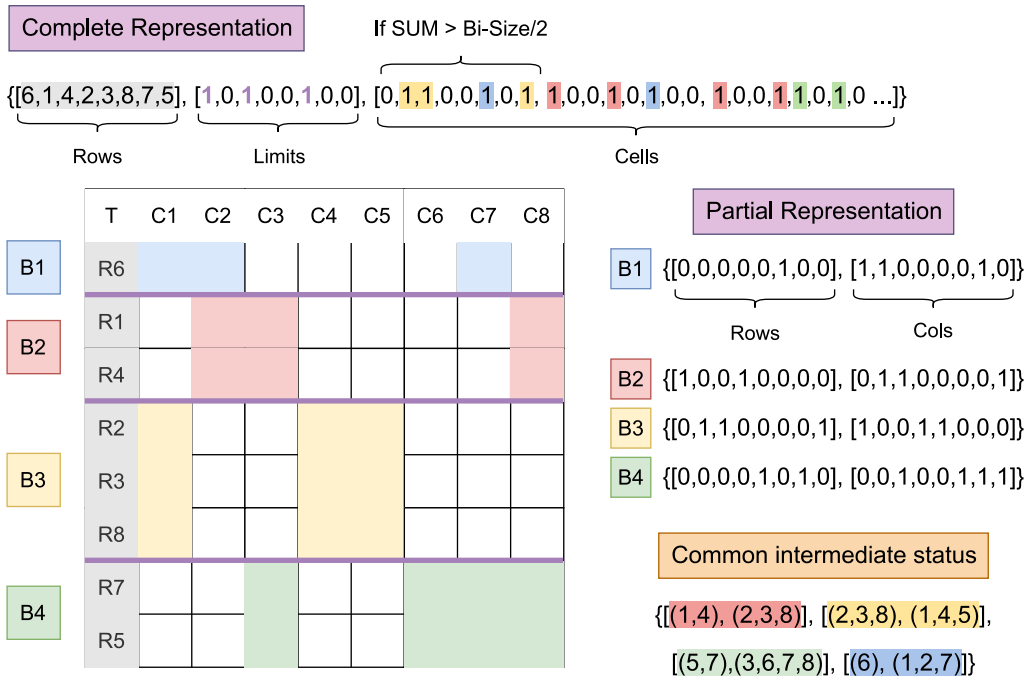


Fig. 3. An example of the COMPLETE and PARTIAL encodings, as well as their common intermediate state, on a simplified 8 × 8 matrix. A solution consisting of 4 biclusters with overlapping columns is shown.

because each individual represents an exhaustive set of biclusters, whose quantity is variable and part of the algorithm's learning process. This perspective helps in designing objective functions that evaluate new aspects of biclustering, as well as realistic functions specialized in the biological domain of the data.

This encoding, named "COMPLETE", is illustrated in Fig. 3 to facilitate understanding. As can be seen, the encoding requires three parts:

1. **Rows**: The first part of the encoding is a permutation that represents the order in which the different rows of the data matrix are arranged (Column T in Fig. 3).
2. **Limits**: It consists of a binary vector of the same length as the previous one, specifying the horizontal boundaries of the biclusters (purple lines in Fig. 3).
3. **Cells**: A binary vector with the same size as the data matrix specifying each cell's activation state. Unlike the previous vector, its interpretation does not depend on the permutation. That is, the first value refers to the activation state of the cell (R1, C1), regardless of its position according to the permutation or which bicluster it belongs to based on the boundaries. If most of the cells in the same column within a bicluster are activated, that column is activated for the bicluster in question. This approach allows for a progressive learning process of column activation and deactivation, where columns with more activated cells are more likely to remain activated. In comparison, those with fewer activated cells are more likely to be deactivated.

The example proposed in Fig. 3 is a simplified case of 4 biclusters in an 8 × 8 matrix. As can be seen, all the biclusters in the solution are represented by a single individual in the case of the complete representation, while 4 independent individuals are needed to capture all this information with the partial representation (each individual is a partial solution to the problem).

This representation is also associated with a set of features that align with observations seen in biclustering problems within the biological domain. First, this representation does not allow overlap in one of the two dimensions (if row overlap and not column overlap is desired, the input numeric matrix needs to be transposed). This constraint is also observed in other well-known algorithms, such as Cheng and Church's Algorithm (CCA) [22] or the Binary Inclusion-Maximal Biclustering Algorithm (Bimax) [27], which have been extensively applied to gene expression data.

Although it is not a restriction, this representation tends to group all elements of the non-overlapping dimension. In Fig. 3, it can be seen that all rows end up belonging to a bicluster. However, if a bicluster deactivates almost all columns or contains only one row (as in B1 in Fig. 3), the framework itself will disregard these biclusters, leaving those rows ungrouped.[4] This design decision aims to facilitate coverage of the data matrix and eliminate the risk of repetitive exploitation of the same areas. This is another limitation of the partial representation that was intended to be overcome, where the dominant quality of one bicluster over others draws all partial solutions towards it.

It is clear that handling an individual is more complex and resource-intensive in the case of the complete representation than in the partial one, a fact supported by the amount of information stored in each case. However, to minimize the memory resources required, the Java BitSet class[5] has been used to store the binary vectors. This allows significant space savings without adding computational cost when manipulating individuals.

Although the complete representation is one of the major contributions of this work, the MOEBA-BIO framework is designed for continuous extension thanks to the flexibility and modularity of its implementation. This means that integrating new encodings into MOEBA-BIO is extremely simple.[6] Thanks to the transition through a common intermediate state (also illustrated in Fig. 3), the evaluation process is completely independent of the encoding of the individuals. Therefore, the only necessary design elements (apart from the encoding itself) are

[4] Java class available in: https://github.com/AdrianSeguraOrtiz/MOEBA-BIO/blob/main/src/main/java/moeba/fitnessfunction/FitnessFunction.java.

[5] https://docs.oracle.com/javase/7/docs/api/java/util/BitSet.html.

[6] Java class available in: https://github.com/AdrianSeguraOrtiz/MOEBA-BIO/blob/main/src/main/java/moeba/representationwrapper/RepresentationWrapper.java.

the transition to this intermediate state (i.e., the interpretation of the representation) and the crossover and mutation operators, which can initially be formed by combining operators already implemented in JMetal [39].

3.2. Objectives

The fitness functions currently available in MOEBA-BIO cover traditional individual-focused objectives, global vision objectives of a generic nature for biclustering, and global perspective objectives specific to particular biomedical application domains. All of these can be reviewed in Fig. 2.

It is important to note that, to ensure compatibility between the traditional individual-centered objectives and the new framework and representation, several strategies have been implemented to summarize individual qualities. Some of these strategies include penalties that account for heterogeneity among those qualities (see Table 2).

In order to use the full range of available algorithms, all fitness functions must be normalized between 0 and 1. Therefore, traditional biclustering objectives had to be reformulated to fit this range, made possible by the prior normalization of the numerical data. Furthermore, since MOEBA-BIO is an extension of JMetal, all objectives must be oriented towards minimization. However, for individual quality summary strategies like the harmonic or geometric mean to be effective, the individual qualities of the biclusters must be oriented towards maximization. As a result, the formulas for individual biclusters are first oriented towards maximization, and after obtaining the joint quality, the complementary value of the result is calculated.

Since traditional objectives for biclustering have been extensively discussed in the literature [12], the subsequent sections focus on the implementations of the new objectives proposed here. Those with a global perspective, whether of a generic nature or associated with a specific biomedical domain, represent significant contributions to this work.

3.2.1. Adaptive bicluster size

This fitness function is designed to evaluate the size of the biclusters in relation to the total number of biclusters in a solution. It combines two main components: the normalized size of the biclusters and an adaptive penalty based on the difference between the actual size of a bicluster and a realistic maximum size, dynamically adjusted according to the number of biclusters in the solution. The balance between these two factors is controlled by a configurable coherence weight, allowing adjustment of the relative importance of size versus homogeneity within the set of biclusters. This encourages solutions that not only maximize the size of the biclusters, but also maintain structural coherence based on the total number of generated biclusters. Unlike the traditional *Bicluster Size (bSIZE)*, it has a complete problem perspective by accounting for the number of biclusters, so its convergence is not focused on obtaining a single bicluster that covers the entire data matrix.

The implementation of this fitness function is represented in the pseudocode shown in Algorithm 2. First, the maximum size a bicluster can have within the data matrix and the actual size of the bicluster under evaluation are calculated (lines 1 and 2 in Algorithm 2). Next, the recommended maximum size of a bicluster is calculated, which is obtained by dividing the maximum size of the matrix by the square of the number of biclusters (lines 3 and 4 in Algorithm 2). This allows the penalty to be dynamically adjusted according to the total number of biclusters present in the solution.

The weighted normalized size of the bicluster is obtained using the traditional bSize function, which takes into account both, the size of the bicluster and the weight assigned to the rows and consequently to the columns (line 5 in Algorithm 2).

Additionally, a penalty is calculated based on the difference between the actual size of the bicluster and the recommended maximum size (line 6 in Algorithm 2). This penalty adapts the size of the bicluster according to the total number of biclusters in the solution.

Finally, the fitness function score is calculated by combining the normalized size of the bicluster and the size penalty, weighted by the coherence parameter α (line 7 in Algorithm 2). The function returns this value as the final fitness result for the evaluated bicluster (line 9 in Algorithm 2).

Algorithm 2 Adaptive bSize fitness function.

Input Bicluster b, Set of all biclusters less b B, Data matrix D, Coherence weight α, Row weight β
Output Value of the fitness function $score$

1: $maxSize \leftarrow rows(D) \times columns(D)$
2: $bSize \leftarrow rows(b) \times columns(b)$
3: $numBiclusters \leftarrow size(B) + 1$
4: $parcelSize \leftarrow maxSize/(numBiclusters^2)$
5: $normWeightedSize \leftarrow calcNormSize(b, D, \beta)$
6: $sizePenalty \leftarrow min(1, abs(parcelSize - bSize)/parcelSize)$
7: $score \leftarrow (1 - \alpha) \times normWeightedSize + \alpha \times (1 - sizePenalty)$
8: **return** $score$

Algorithm 3 Bicluster Differentiation fitness function.

Input Bicluster b, Set of all biclusters less b B, Data matrix D
Output Value of the fitness function $score$

1: $closest \leftarrow findClosest(b, B)$
2: $meanB \leftarrow calcMean(b)$
3: $meanClosest \leftarrow calcMean(closest)$
4: $totalScore \leftarrow 0$
5: **for** each row in $closest$ **do**
6: $riClosest \leftarrow getCells(row, cols(closest))$
7: $riB \leftarrow getCells(row, cols(b))$
8: $distBMean \leftarrow calcDist(riB, meanB)$
9: $adjMeanClosest \leftarrow calcAdjMean(meanClosest, riClosest)$
10: $distAdjMean \leftarrow calcDist(riClosest, adjMeanClosest)$
11: $fitScore \leftarrow distBMean/(distBMean + distAdjMean)$
12: $totalScore \leftarrow totalScore + fitScore$
13: **return** $totalScore/size(closest)$

3.2.2. Bicluster differentiation

Leveraging the no-row-overlap constraint, this objective focuses on ensuring that each bicluster is correctly defined, without excluding rows that fit better with the averages of their columns than with others, nor including rows that are more aligned with other biclusters. The function compares the bicluster under evaluation with the nearest one, determined by the number of shared columns.

For each row of the nearest bicluster, two distances are calculated: one to the values of the evaluated bicluster and another one to the values of its own bicluster (adjusted to reflect the exclusion of the row in question). If the distance from the row to the evaluated bicluster is less than that of the adjusted averages of its own bicluster, the evaluated bicluster is penalized for excluding an aligned row.

This function rewards biclusters whose rows are better represented within them than in other biclusters, improving differentiation between biclusters and the overall coherence of the solution. Thus, the biclusters reflect more accurately the patterns of the rows they contain, contributing to greater biclustering precision without encouraging size reduction (as seen in other traditional functions).

The implementation of this fitness function is represented in the pseudocode shown in Algorithm 3. First, the bicluster closest to the one under evaluation is identified, using the number of shared columns as the criterion (line 1 in Algorithm 3).

Next, the column averages for both the evaluated bicluster and the nearest bicluster are calculated (lines 2 and 3 in Algorithm 3). These

Algorithm 4 Regulatory Coherence fitness function.

Input Set of all biclusters B, Gene regulatory network inferred by GENIE3 G
Output Value of the fitness function $score$

1: $rowBics \leftarrow assignBiclusters(B, G)$
2: $sum \leftarrow 0$
3: **for** each gi in G **do**
4: **for** each gj in G **do**
5: **if** $rowBics[gi] = rowBics[gj]$ **then**
6: $sum \leftarrow sum + G.conf(gi, gj)$
7: $sum \leftarrow sum - \frac{G.outDegree(gi) \times G.inDegree(gj)}{G.totalWeight}$
8: $modularity \leftarrow sum/G.totalWeight$
9: **return** $1 - (modularity + 1)/2$

averages are essential for comparing how well the rows of the nearest bicluster fit their own average compared to the average of the evaluated bicluster.

In the loop (lines 5–12 in Algorithm 3), the rows of the nearest bicluster are iterated over. For each row, the values of the columns in both, the nearest bicluster and the evaluated bicluster are retrieved (lines 6 and 7 in Algorithm 3). Then, the distance of these values to the averages of the evaluated bicluster is calculated (line 8 in Algorithm 3), along with the distance to the adjusted averages of the nearest bicluster, which reflects the exclusion of the given row (lines 9 and 10 in Algorithm 3).

The fitness score is calculated as the ratio of the distance from the row to the evaluated bicluster to the sum of both distances (line 11 in Algorithm 3). This metric indicates whether the row fits better in the evaluated bicluster or the nearest bicluster. If the distance to the evaluated bicluster's averages is smaller, the evaluated bicluster is penalized with a lower score, accumulating this value into the total (line 12 in Algorithm 3).

Finally, the total score for the evaluated bicluster is obtained as the average of the fit scores calculated for each row of the nearest bicluster (line 13 in Algorithm 3).

3.2.3. Co-expression: Regulatory coherence

This fitness function focuses on evaluating the regulatory coherence of biclusters within the context of gene co-expression, a specific biomedical domain. Unlike previous fitness functions that evaluate each bicluster (whether they have a global perspective of the solution or not), Regulatory Coherence assigns a global score to the complete solution without the need for summary strategies. This function aims to demonstrate MOEBA-BIO's capability to design algorithms tailored to specific biological domains, facilitating knowledge injection through the global perspective complete representation and the direct equivalence between individual and solution.

The foundation of this function is based on studies indicating that co-expressed genes tend to be regulated by the same transcription factors [47], and often, due to the typical scale-free topology of gene regulatory networks [19], they belong to the same communities. The function measures the modularity of the partition into communities that emerge from the individual's biclusters on the gene regulatory network inferred by the GENIE3 tool [48], based on the provided input data.

The final score reflects the modularity of the solution, where a value close to 0 indicates high coherence (genes well-grouped by their common regulators), and a value close to 1 indicates low coherence. This allows for evaluating how well the co-expressed genes in a bicluster are regulated by the same factors in the inferred network, providing an accurate and biologically meaningful metric to assess the biclustering solution in gene expression data.

The implementation of this fitness function is represented in the pseudocode presented in Algorithm 4. First, each gene is assigned to a bicluster, mapping the genes within the groups formed by the biclusters (line 1 in Algorithm 4).

Next, a cumulative variable called sum is initialized to store the result of the regulatory coherence calculated between the genes within each bicluster (line 2 in Algorithm 4). Then, all pairs of genes present in the regulatory network inferred by GENIE3 are iterated over (lines 3–7 in Algorithm 4).

Within this double loop, for each pair of genes g_i and g_j, it is checked whether both belong to the same bicluster (line 5 in Algorithm 4). If they do so, the cumulative sum is updated by adding the regulatory confidence value between the two genes in the network (line 6 in Algorithm 4) and subtracting an adjusted term based on the out-degrees and in-degrees of both genes, normalized by the total weight of the network (line 7 in Algorithm 4).

After processing all gene pairs, the modularity value is calculated by dividing the cumulative sum by the total weight of the regulatory network (line 8 in Algorithm 4). The final score is normalized to a range between 0 and 1 using a linear transformation (line 9 in Algorithm 4), where a value close to 0 indicates high coherence, while a value close to 1 indicates low regulatory coherence. Since this fitness function does not require a summary strategy, it is directly oriented towards minimization.

3.3. Parameter autoconfigurator

In addition to the new representation and the objectives designed thanks to its global perspective, MOEBA-BIO also facilitates the injection of domain-specific biomedical knowledge through its parameter self-configurator. Parameter self-configuration using wrapper evolutionary algorithms has already been validated in other proposals in the literature [49]. However, in this case, a self-configurator has been specifically designed for the biclustering problem when applied to particular domains.

Fig. 4 shows the diagram of the proposed self-configurator. It consists of two simple genetic algorithms instantiated with the most common parameter values, which wrap the evolutionary algorithm to be configured. It takes as input a series of data matrices associated with a specific biomedical context, the files with the reference biclusters for each matrix (gold standards), a file with the supervised parameters to be configured, and another for the unsupervised parameters, specifying the set of possible values for each one.[7] As output, it provides the best parameter configuration obtained. Additionally, tracking files are provided for the evolution of the external wrapper population and the evolution of the internal wrapper population corresponding to the external winner.

- **Supervised Phase**: This corresponds to the outer wrapper genetic algorithm (the green one in Fig. 4). In this phase, different combinations of objectives (with varying numbers of dimensions as long as $n \geq 2$) and values of their subparameters are tested to determine which configuration minimizes a validation metric related to the gold standards of the benchmark representing the biomedical domain of the data. However, since different configurations at this level imply different search spaces with varying numbers of dimensions, it would be unwise to set the same values for the other parameters in all cases. Therefore, given a configuration at this level, it is necessary to extract the best values for the remaining parameters in the next phase before proceeding with its evaluation. The evaluation is performed after obtaining the best individual from the unsupervised phase. For each data matrix, the 5 best individuals obtained on the MOEBA-BIO Pareto approximation front according to the selected supervised metric are taken, and the final score is returned as the average across all matrices.

[7] Configuration file examples in: https://github.com/AdrianSeguraOrtiz/MOEBA-BIO/tree/main/parameterization.

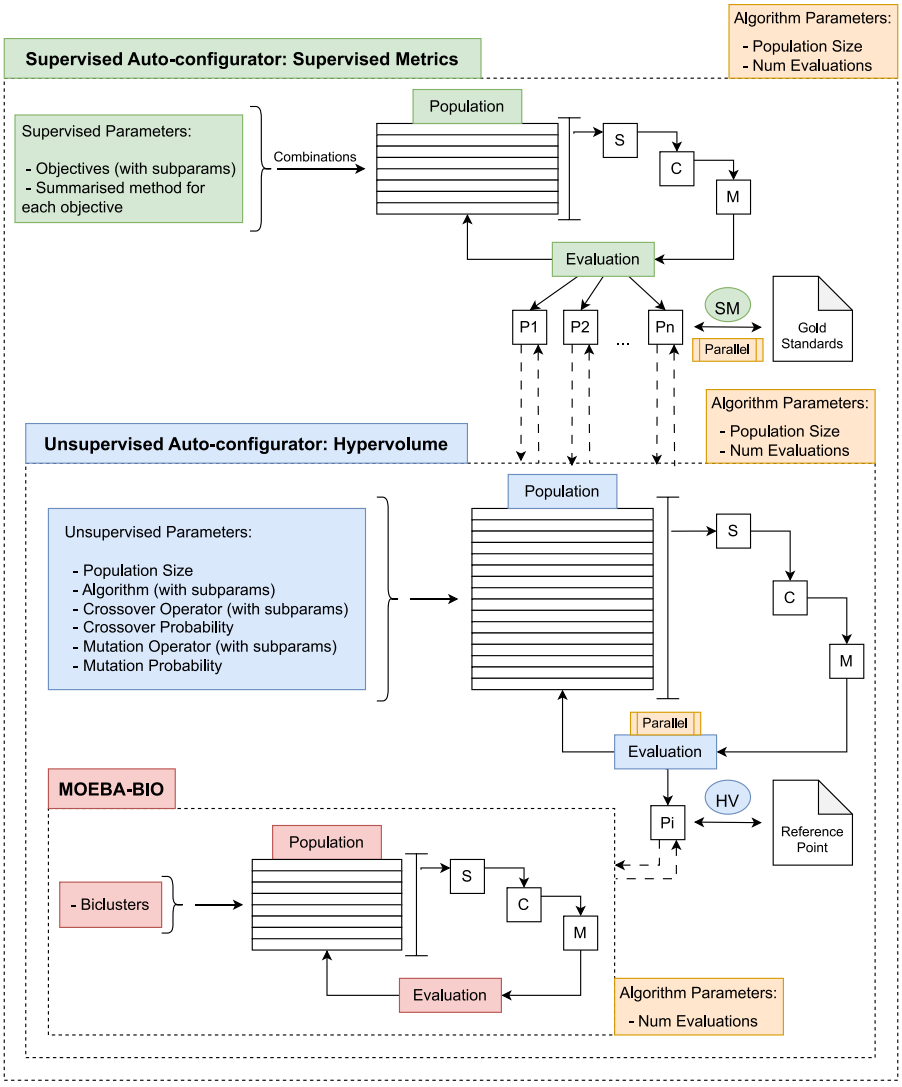


Fig. 4. Structure of the specific self-configurator in the MOEBA-BIO framework. It consists of two wrapper genetic algorithms. The outer one handles the self-configuration of objectives through a supervised evaluation that depends on the gold standards of the input data. The inner wrapper handles the self-configuration of the remaining parameters, based on unsupervised metrics such as hypervolume.

- **Unsupervised Phase**: This corresponds to the inner wrapper genetic algorithm (the blue one in Fig. 4). Once the objectives of the problem and their corresponding subparameters have been set in the external individual, this phase optimizes the values of the remaining parameters, including crucial ones, such as: the algorithm to use with its corresponding subparameters, the operators, the probabilities for each phase, the population size, and more. This optimization, now performed on comparable Pareto fronts, can be based on standard unsupervised metrics such as hypervolume, which, given that all objectives are normalized between 0 and 1, can use the value 1 in each dimension to form its reference point. Again, for each data matrix, the metric to be minimized is calculated, and the average across all matrices is returned.

It should be noted that, despite its stochastic nature, no repetitions have been implemented for the internal execution of the algorithm being configured for each parameter combination. This decision is based on the fact that, due to the redundancy of individuals in such small search spaces, it is considered that the potential variations dependent on the executions can be effectively covered by this redundancy. This measure has been adopted to ensure the computational feasibility of the self-configurator, and its validity will be demonstrated in the subsequent experimentation.

To clarify the evaluation process carried out by the self-configurator, Fig. 5 illustrates an example of how MOEBA-BIO evaluates each candidate configuration. Specifically, the figure depicts how each outer individual defines a combination of objective functions and subparameters, which is then internally assessed by an inner evolutionary layer using different algorithmic setups. The final quality of each outer individual is computed based on the performance of the inner layer across multiple datasets.

It is worth noting that both evolutionary algorithms that make up the self-configurator have been designed as single-objective optimization processes. The algorithm in the external (supervised) phase is guided exclusively by a gold standard-dependent metric (specifically, the clustering error), while the algorithm in the internal (unsupervised) phase uses only the hypervolume as its objective function. Therefore, the self-configurator does not involve any weighting or prioritization among multiple objectives, nor does it need to resolve conflicts between them. Its role is to identify the combination of objectives (and associated parameters) that yields the best results in a supervised context, optimizing the remaining parameters in an unsupervised context. This

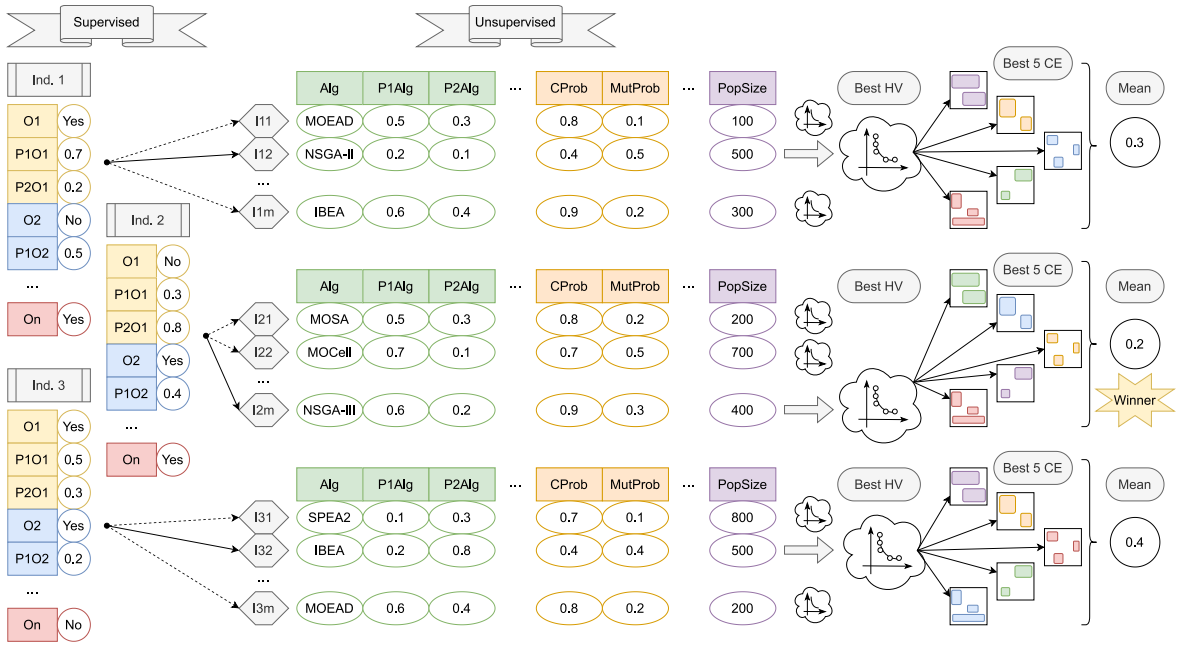


Fig. 5. Visual representation of the MOEBA-BIO self-configuration mechanism. The process is structured in two nested evolutionary phases: a supervised outer layer (left) and an unsupervised inner layer (right). Each individual in the outer population encodes a specific combination of objective functions and subparameters. For each one, the inner layer determines the best technical configuration (algorithm, operator probabilities, population size, etc.) based on hypervolume (HV). The final evaluation of each outer individual is computed by averaging the clustering error (CE) of the top 5 Pareto-optimal solutions obtained for each dataset. The best outer individual (i.e., configuration) is selected as the winner.

structure helps maintain interpretability throughout the process and facilitates its adaptation to other domains where the set of objectives or evaluation metrics may vary.

3.4. Methodological comparison

Once all the methodological components of MOEBA-BIO have been presented, it is appropriate to contrast them with the most closely related state-of-the-art approaches that also use non-traditional encodings: BI-MOCK, PBD-SPEA2, and BiClustSMEA.

Unlike these approaches, MOEBA-BIO not only modifies the representation of individuals, but also proposes a complete evolutionary framework that enables the representation of complete solutions, the definition of objectives from both a global perspective and an application-specific standpoint, and the application of structural penalization mechanisms. In addition, it integrates both supervised and unsupervised self-configuration systems, allows for decoupled objective design, and facilitates its extension to other biomedical contexts.

To highlight these differences beyond the encoding itself, Table 3 summarizes the key methodological and implementation-related aspects that distinguish MOEBA-BIO.

4. Experimentation

The experimentation in this study is divided into six main phases.

The first phase aims to validate the new complete problem encoding proposed in this work and the general context objective functions implemented thanks to it. To achieve this, conducting a fair, rigorous comparison within the most generic context is necessary.

Recent proposals in the literature that used a partial representation were analyzed to conduct this comparison. After reviewing the recent survey [12], which focuses on analyzing all biclustering algorithms with a multi-objective evolutionary approach, a list of 22 candidates was obtained: TSTP [50], BOBEA [33], MMco-Clus [51], β-SMOB [31], AMOSAB-PS [15], BP-NSGA2 [14], PBD-SPEA2 [36], AMOSAB [34], SPEA2B-δ [32], HMOBI [52], SMOB-VE [53], MOBI [54], SPEA2B [55], MOGAB [56], MOM-aiNet [57], MOACOB [58], AMOPSOB [59], CMOPSOB [60], MOPSOB [61], MOFBA [62], SMOB [63] and MOEAB [64]. However, since none of the papers on these algorithms provide public access to their software, a partial representation was integrated into MOEBA-BIO exclusively for experimental purposes. This approach enables an alternative yet fair comparison.

The first step is to set the objectives to be used during the comparison. In [12], it is shown that the most commonly used combination is Mean Squared Residue (MSR) and Bicluster Size (bSIZE). Therefore, these two functions will be established as the main objectives in this phase of experimentation. The goal is to evaluate each contribution separately, resulting in 5 configurations:

- Partial (bSIZE + MSR).
- Complete (bSIZE + MSR).
- Complete replacing bSIZE by Adaptive bSIZE with a coherence weight value of 0.25 (Adaptive bSIZE + MSR).
- Complete adding bDIFF (bSIZE + MSR + bDIFF).
- Complete representation but adding both new objective functions at the same time (Adaptive bSIZE + MSR + bDIFF).

Both complete (proposal) and partial (traditional) encoding configurations are compared within the same MOEBA-BIO framework using the standard operators for each representation. To provide a fairly diverse and flexible genetic content in this first experimental environment, the population size has been set at 500 individuals. Regarding the rest of the parameters, all have been set to their most common and reasonable values: the chosen algorithm is NSGAII [40], with a total of 100,000 evaluations, a crossover probability of 90%, and a slightly higher than normal mutation probability set at 10% due to the size of the data matrices.

The dataset used in this first phase has been artificially generated using the G-bic tool [65]. Four key variables have been defined to ensure a wide variety of data characteristics: number of biclusters, bicluster size, noise level, and degree of overlap in the columns. Each variable has been evaluated at four levels, generating three data matrices per level using different random seeds. In total, 48 numerical matrices of 1000 × 500 have been obtained, each accompanied by

Table 3
Methodological and implementation-level comparison of MOEBA-BIO with representative non-traditional encoding proposals.

Implementation aspect	BI-MOCK	PBD-SPEA2	BiClustSMEA	MOEBA-BIO (this proposal)
Control over number of biclusters	Automatically determined thanks to a newly proposed variable string length encoding scheme	User-defined, must be specified.	Generated during execution using a random variable	The number of biclusters is variable and is fully self-learned via evolutionary convergence
Aggregation of fitness across biclusters	Simple mean of the MSR and bSize over the set of biclusters encoded in the solution	Fitness is likely evaluated per bicluster	Mean across all δ-biclusters present in a chromosome for objectives like MSR, row variance and volume	Harmonic amn Geometric mean available to summarize individual bicluster qualities. Penalized heterogeneity.
Final solution selection	A post-processing step is proposed to select the set of final biclusters among Pareto front individuals.	Post-processing across individuals; biclusters from different individuals might be combined.	Sequential selection from the Pareto optimal set based on the lowest MSR value.	Each individual in the Pareto front represents a complete solution to the biclustering problem; no aggregation required.
Extensibility of objective functions	Not explicitly designed for easy extension based on the description.	Likely hard-coded within the algorithm's structure.	Requires modification to the algorithm's core to incorporate new objectives.	Plugin-based, fully decoupled objective function design; researchers have complete freedom to implement new objectives or use predefined ones.
Domain integration capacity	Only traditional objectives like bSize and MSR are mentioned.	Not considered, homogeneity and size are used as primary objectives.	Focuses on general quality metrics.	Native support for domain-specific global objectives (e.g., Regulatory Coherence for gene co-expression).
Open source/Reproducibility	No public availability.	No public availability.	No public availability.	Yes (publicly available framework): https://github.com/AdrianSeguraOrtiz/MOEBA-BIO.

Table 4
Characteristics of the data generated by G-bic for the first phase of experimentation.

Variable	Size of planted biclusters	Noise on data matrix	N° Biclusters	Overlapping in columns
NB (Number of Biclusters)	50 × 50	0.0%	{3, 5, 8, 10}	Unrestricted
SB (Size of Biclusters)	{(25,25), (50,50), (75,75), (100,100)}	0.0%	3	Unrestricted
NL (Noise Level)	50 × 50	{5, 10, 15, 20} %	10	Unrestricted
OL (Overlap Level)	50 × 50	0.0%	10	{10, 15, 20, 25} %

its corresponding reference bicluster set (gold standard). The levels of each variable are shown in Table 4. Since G-bic does not allow for the complete prohibition of row overlap during data generation, post-processing was necessary, where overlapping rows were duplicated, and the cells outside each bicluster were replaced with data from ungrouped rows, removing the latter to avoid distorting the matrix size.

When running MOEBA-BIO with each configuration, an approximate Pareto front is obtained for each input matrix. In the case of the partial configuration, the entire front is combined to form the real solution to the problem. For the complete encoding configurations, each individual in the front can be directly compared with the gold standard. Therefore, for the partial representation, the quality score is the comparison between the solution formed by the entire front and the gold standard, while for the complete configurations, the median of the quality scores from the front is calculated. This allows for the subsequent calculation of a Friedman statistical ranking complemented by Holm's non-parametric tests [66] to provide statistical rigor in the comparison of configurations.

Regarding the evaluation of the biclustering results, the clustering error [67] has been selected as the main comparison metric across biclustering algorithms. This is a robust metric that evaluates the complete biclustering solution by penalizing both the redundancy between biclusters and the difference between the number of inferred biclusters and the actual number in the gold standard. It is also a metric that has been used in many recent biclustering studies as well as in the context of evaluating evolutionary biclustering algorithms [12]. The clustering error should be minimized such that a lower value indicates a better biclustering solution. Therefore, for this phase of experimentation and the subsequent statistical significance analysis, the complementary value of the result is calculated.

This evaluation strategy ensures that the benefits obtained from the complete configuration compared to the partial one are solely due to the encoding of the individuals. Furthermore, it allows for the determination of whether each new objective individually improves upon the base use of the new encoding and whether their combined use provides greater benefits than each individual.

Additionally, notice that other biclustering metrics such as Recovery, Relevance [68] and Ayadi's score [69] are also considered in our further experiments as they are widely used in the specialized literature.

The second phase of the experimentation focuses on using the self-configurator on a dataset specific to the biomedical domain. The data comes from the simulator in the R package FABIA [70], which allows the generation of artificial gene expression data-oriented to biclustering. As in the previous experimentation, although on a smaller scale due to the domain focus, different scenarios have been considered through various configurations of the simulator. A total of 4 data matrices of different sizes have been generated, with the simulator configurations shown in Table 5.

The self-configurator has considered all possible values for both first-level and second-level parameters. For both wrapper genetic algorithms, a population size of 50 individuals has been set, with 1500

Table 5
Characteristics of the matrices generated by the FABIA simulator.

Instance	Description	Matrix size	N° Biclusters	Bicluster size	Overlap	Noise (Std. Dev.)
Inst. 1	Small Biclusters - Low Noise	200 × 100	10	20 × 10	0% × 32%	1.0
Inst. 2	Large Biclusters - Moderate Noise	500 × 200	8	65 × 25	0% × 35%	2.0
Inst. 3	Mixed Biclusters - High Noise	300 × 150	12	(20–30) × (7–12)	0% × 30%	4.0
Inst. 4	Medium Biclusters - Moderate Noise	400 × 150	10	40 × 15	0% × 37%	1.5

evaluations for the outer loop and 2000 for the inner loop, as the latter has a larger search space. These values are based on those tested in the reference meta-optimizer [49], which have also proven to be sufficient for the convergence of both algorithms. Additionally, the number of MOEBA-BIO evaluations has been reduced to 25,000, a quantity that demonstrated the ability to capture the most significant progress of the populations in the first phase of experimentation. Furthermore, clustering error [67] has been chosen as the supervised metric, and the hypervolume value, changed to a negative sign with unitary reference, has been chosen as the unsupervised metric.

The third phase tests the effectiveness of the self-configurator. To achieve this, the winning configuration from the previous phase is compared with other candidates from both the supervised and unsupervised phases. This process allows for verifying whether the clustering error metric has been minimized while simultaneously improving the hypervolume value. Each configuration is executed five times with 150,000 evaluations, allowing for a more robust comparison and justifying the decision to avoid repetition in the self-configuration process, thereby prioritizing computational feasibility without damaging the quality of the solutions.

The fourth phase, once the best configuration of MOEBA-BIO has been identified for the simulated benchmark that represents the gene co-expression context, evaluates the developed evolutionary algorithm against other proposals through a technical validation. For this comparison, widely used algorithms of various types have been selected from the literature [11]. Specifically, CCA [22], OPSM [23], xMOTIFs [24], ISA [25], LAS [26], Bimax [27], BiBit [28], Plaid [29], and Spectral [30] have been selected. For their execution, the biclustlib Python library [71] was used, providing them with the data matrices previously generated by the FABIA package [70] in R.

It is important to note that the CCA, Bimax, Plaid, xMOTIFs, and LAS algorithms require identifying the number of biclusters beforehand. This gives them a comparative advantage over MOEBA-BIO, where this information is unknown and part of the algorithm's learning process. To make the comparison fairer, the exact number was not provided; instead, an approximate value was calculated as 10% of the average between the number of rows and columns.

In the dataset generated by FABIA, this means the approximate number of biclusters is slightly higher than the actual number. This allows these algorithms the possibility of identifying all biclusters from the gold standard, while also testing their ability to avoid grouping unrelated genes in expression patterns when the researcher lacks an accurate estimate of the number of biclusters, a situation also evaluated in MOEBA-BIO and, therefore, important for validating these algorithms.

In addition to the clustering error [67], supervised individual level precision metrics such as Recovery, Relevance [68] and Ayadi's score [69] are used. These metrics do not consider redundancies and select the best-inferred bicluster for each reference bicluster in the gold standard. This approach eliminates the penalty for redundancy to demonstrate that the quality of this proposal is not due, in any case, to poor guidance from algorithms requiring knowledge of the number of biclusters beforehand. If MOEBA-BIO achieves better results in these metrics, it would highlight its ability to manage redundancy effectively when the number of biclusters is unknown. It also means that found biclusters surpass the best match achieved by the algorithms that fail to identify the correct number of biclusters.

After execution, for each state-of-the-art algorithm result, the metric value is calculated directly, while for MOEBA-BIO, the results from the previous phase are used to select the best solution and the overall median of the distribution from the 5 runs.

Additionally, to achieve a more comprehensive technical comparison, the execution times of each algorithm are measured. This aims to assess the computational performance of each proposal by analyzing both the accuracy of the obtained results and the time required to achieve them.

The fifth phase extends the validation process of this proposal to a more biological level by utilizing real-world gene expression data. Specifically, the benchmark dataset compiled in [72] is used, which gathers time-series gene expression data from various sources [73–75]. These datasets, obtained from cDNA microarray experiments in *Saccharomyces cerevisiae* (yeast), have undergone a rigorous preprocessing procedure, including the removal of genes with excessive missing values and normalization using the Multiple-Slide Normalization procedure [76]. After this processing, a total of 17 datasets are obtained, each containing approximately 1000 genes selected based on the variation in their expression over time.

Since these data are unlabeled, they do not have a gold standard for conducting a technical comparison as in the previous experimental phase. Instead, a biological validation is performed based on the functional enrichment analysis of the biclusters.

For this phase, the Python library biclustlib,[8] an extension of the homonymous project introduced in [71] (used in the previous phase), is employed. This version has been enriched with the functionalities of the well-known GOATOOLS library [77]. This software enables the integration of the same methodologies previously considered (CCA [22], OPSM [23], xMOTIFs [24], ISA [25], LAS [26], Bimax [27], BiBit [28], Plaid [29], and Spectral [30]), as well as the use of the mentioned dataset[9] to assess the biological relevance of the biclusters obtained by each approach through functional enrichment analysis with GOATOOLS.

The functional enrichment analysis is based on identifying Gene Ontology (GO) [78] terms that are significantly represented among the genes grouped within each bicluster. To achieve this, an over-representation test is applied, comparing the proportion of genes annotated with a GO term within a bicluster to the expected proportion in the reference population. Specifically, a Fisher's exact test [79] is used, with significance values corrected using the Benjamini–Hochberg procedure to control the false discovery rate (FDR) [80].

For each evaluated algorithm, the proportion of enriched biclusters is measured at different significance levels ($\alpha \in \{0.05, 0.005, 0.00001\}$). A bicluster is considered enriched if at least one of its associated GO terms has a corrected p-value below α. The higher this proportion for a given methodology, the stronger the evidence that the inferred biclusters capture functionally coherent groupings of genes, reflecting biologically relevant relationships in the gene expression data.

Once the proportions of enriched biclusters have been obtained for each methodology, dataset, and significance level, a Friedman statistical ranking is conducted along with non-parametric Holm tests [66] for each α value. This analysis allows verifying whether the biological relevance of the biclusters inferred by MOEBA-BIO is statistically superior to that of the other methodologies.

[8] Available on PyPI: https://pypi.org/project/biclustlib/.
[9] Datasets available at: https://github.com/nikitasigal/biclustlib/tree/main/src/biclustlib/benchmark/data/jaskowiak.

Table 6
Friedman mean rank with Holm's adjusted p values (0.05) for clustering error.

Clustering error		
Technique	$Friedman's\{Rank\}$	$Holm's\{Adj-p\}$
***Complete-AdapBS(0.25)+BDiff**	**1.42**	–
Complete+BDiff	2.29	6.71e−03
Complete-AdapBS(0.25)	2.85	1.69e−05
Complete	3.44	1.15e−09
Partial	5.00	4.88e−28

The sixth phase involves analyzing how the size of the input matrix affects the execution time of the algorithm constructed in the previous phases. For this purpose, multiple synthetic datasets with 100 columns were generated, progressively increasing the number of rows from 100 to 5000 in steps of 100. Since this dimension does not involve overlap and tends towards full coverage by the algorithm, it enables an accurate assessment of the scalability of the approach, allowing observation of whether the growth in execution time is linear, quadratic, or follows another pattern, and identification of potential bottlenecks resulting from increased data size.

It is important to note that the generated data are completely random, with no internal structure or presence of biclusters. As a result, the outcomes of the algorithm in this phase have no analytical or interpretative value; the purpose of the experiment is strictly limited to the analysis of computational cost. To ensure representative results, the algorithm was executed five times on each matrix, and the median execution time was subsequently calculated.

5. Results

To facilitate the understanding of this document, the results of each phase of the experimentation are presented in the same order as detailed in the previous section.

5.1. Codification and objectives

Regarding **the first phase**, Table 6 shows the results of applying a Friedman statistical ranking with Holm's non-parametric tests to the clustering error values obtained after running MOEBA-BIO on the entire benchmark of 48 matrices generated by G-Bic, using each of the configurations to be compared. As can be seen, the configuration associated with the partial encoding ranks last, resulting in a lower rank than the complete encoding when none of the new objectives is applied. In fact, aside from this distance in the overall ranking, a Wilcoxon test between these two configurations yields a p-value of 7.11e−15, which is significantly below the commonly accepted threshold of 0.05. This allows us to affirm that, under the same experimental conditions (traditional objectives, environment, data, and basic crossover and mutation operators), the use of the complete encoding offers a statistically significant improvement in result accuracy compared to the traditional encoding.

Additionally, by observing the intermediate positions in the Friedman ranking, it can be affirmed that both, the inclusion of bDIFF and the replacement of the traditional bSIZE function with Adaptive bSIZE separately outperform the basic complete configuration. In other words, both functions individually contribute to improving the accuracy of the results. Finally, the top-ranked configuration confirms that the new objectives, besides offering benefits individually, provide even better improvement when used together, and this improvement is also statistically significant compared to the other configurations in the ranking, according to Holm's non-parametric tests.

With all these observations, it can be confirmed that the complete encoding not only increases accuracy on its own, but also enables the inclusion of holistic perspective knowledge which, even in a general context, has demonstrated further improvements in the algorithm's

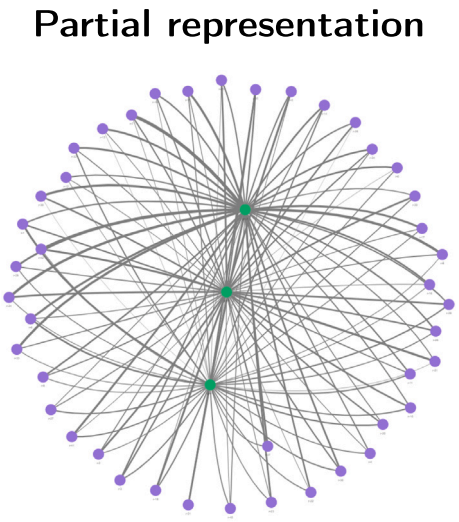


Fig. 6. Biclustering solution obtained by MOEBA-BIO using the partial representation for the simulated dataset with 3 biclusters and seed 102. The green nodes represent the biclusters from the gold standard. Each purple node in this encoding represents a partial solution from the front obtained by the algorithm since, in this case, the combination of all biclusters in the front constitutes the real biclustering solution to the problem. The graph's edges refer to the intersection between biclusters, with their thickness increasing in proportion to the number of shared rows and columns. To avoid adding noise to the graph, intersections between nodes of the same color caused by possible overlaps between biclusters have been ignored.

results. However, to better understand the reasons behind these results, a deeper analysis has been conducted to ensure that the success of this encoding is due to the premises that initially motivated its implementation.

In this regard, the first motivation behind the complete encoding was to overcome the fact that the partial representation tends to converge towards a large number of redundant biclusters, which also exhibit significant heterogeneity in their qualities. To verify this, solutions obtained using the partial encoding and winning complete configuration are shown in Figs. 6 and 7, respectively, compared to the corresponding gold standard. In concrete, Fig. 6 shows a solution from the partial encoding (obtained as the union of all individuals in the front), where the number of biclusters clearly exceeds the actual number of biclusters in the data matrix. This redundancy is not typically penalized by traditional supervised metrics [12], where each bicluster in the gold standard is only compared to the most similar bicluster in the front. In a real-world scenario, where there is no prior information about the biclusters, this excessive number of redundant clusters would add counterproductive noise.

Second, Fig. 7 presents several solutions from the complete encoding. On the left side of the figure, solutions with an intermediate position in the Pareto front are shown. This position, defined as a coherent balance between objectives, seems to correspond to solutions with a number of biclusters that are quite similar to the real number. This observation leads to two conclusions: the number of biclusters is part of the algorithm's learning process (which does not happen with the partial encoding), and the noise caused by excessive and redundant biclusters is clearly minimized.

However, it remains to demonstrate the ability of the complete encoding to ensure that the biclusters exhibit qualities consistent with the position in the front of the solution to which they belong. This demonstration is shown on the right side of Fig. 7. In Fig. 7(e), the biclusters obtained from a solution located at the extreme of the Pareto front with the best optimization of the Adaptive bSIZE objective are displayed. Since the coherence weight has been set to a value of 0.25, the normalized size of the biclusters still carries more weight. This implies that, although the convergence of this function does not directly

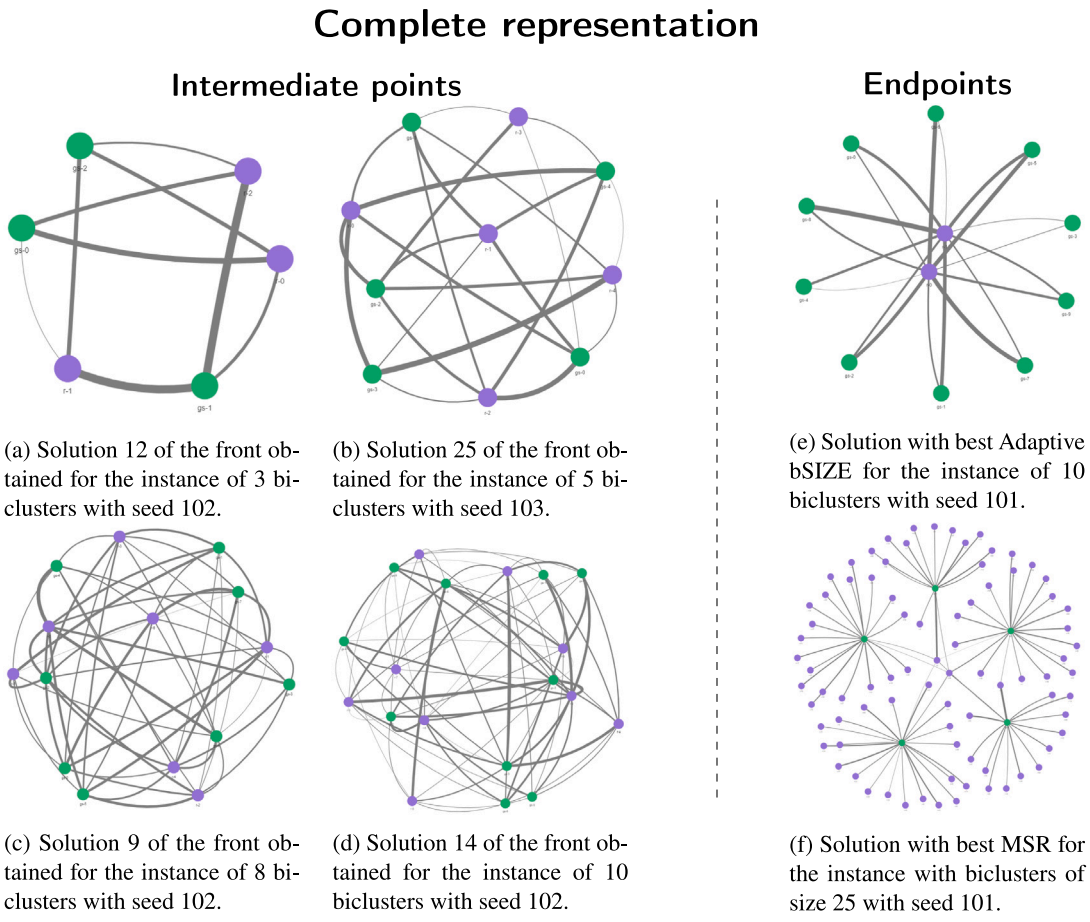


Fig. 7. Examples of solutions obtained by MOEBA-BIO during the first phase of experimentation using the complete representation and the new objective functions of adaptive bicluster size and bicluster differentiation. The interpretation of the graphs is the same as discussed in Fig. 6, except that in this case, all purple nodes belong to a single solution from the front obtained by the algorithm, as in the complete encoding, a solution from the front is equivalent to a real solution to the problem. On the left side of the figure, solutions that are balanced across the three objectives are presented. On the right side, extreme solutions from the front are shown, specifically one for the case of extreme optimization of Adaptive Bicluster Size and another for the case of extreme optimization of MSR.

lead to a single bicluster occupying the entire matrix, the optimization of this function still tends towards large biclusters with low coherence. This is what is observed in Fig. 7(e), where the solution with the best Adaptive bSIZE indeed contains few large biclusters. Meanwhile, Fig. 7(f) shows a solution with the best MSR value in its front. Similarly to what was observed with Adaptive bSIZE, the solution contains biclusters consistent with its position in the front, in this case, a large number of small biclusters with high internal coherence.

In addition to analyzing the algorithm's results, it is crucial to examine the evolution of the populations of individuals during execution. For this reason, Fig. 8 shows the evolution of the different objectives for a specific run of the winning complete configuration. Specifically, it displays the minimum value found in each generation for each fitness function. In Fig. 8(a), we can observe how the Adaptive bSIZE objective, being the simplest one and independent of the data content, exhibits a more abrupt and decisive learning curve. Meanwhile, in Fig. 8(b), although MSR converges at a point quite similar to the previous objective, its learning process seems more gradual and progressive. Finally, the complexity of the new global perspective objective, bDIFF, is shown in Fig. 8(c), where its learning process requires a larger number of generations to reach convergence. Nevertheless, it has already been demonstrated that including this objective provides significant benefits to the algorithm.

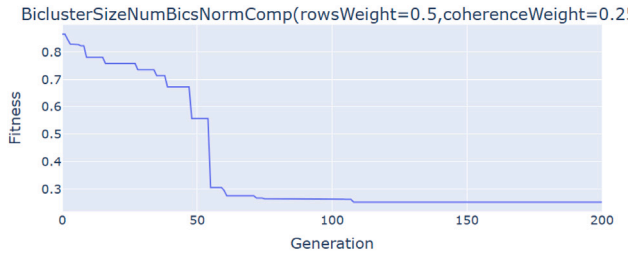
Finally, Fig. 9 represents, for the same run as in Fig. 8, the distribution of the number of biclusters contained in the individuals for each generation. This allows the definitive conclusion that the number of biclusters is part of the algorithm's learning process. Despite not being an explicit objective of the algorithm (as the correct number of biclusters is unknown), the curve presented by the distributions clearly mirrors the curves traced by the evolution of the different objectives. In other words, there is a clear direct learning process, whereby the joint optimization of objectives translates into convergence in the number of biclusters present in the individuals of the population.

For each instance, the convergence of the different objectives and the implicit learning of the number of biclusters lead to an approximate Pareto front. These fronts provide the domain expert with various complete solutions to the biclustering problem on the input dataset. Fig. 10 shows different Pareto fronts obtained by MOEBA-BIO in the first phase of experimentation with the new encoding and proposed objectives. As can be seen, the shape and distribution of solutions vary across different instances.

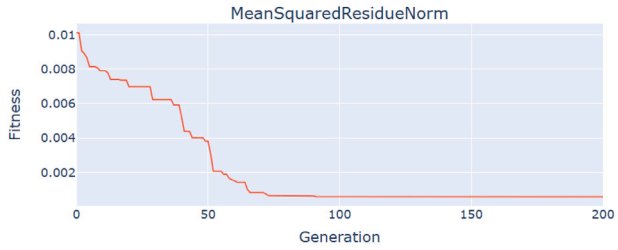
5.2. Autoconfiguration

The second phase of experimentation in this study focuses on the execution of the self-configurator on a dataset specific to the domain of gene expression. The run on a machine with 64 cores and 700 GB of RAM took approximately 6 days and 14 h. Although it takes a considerable amount of time, it is considered a valuable investment in exchange for identifying the subset of objectives and parameter values that best explain the biological domain of application.

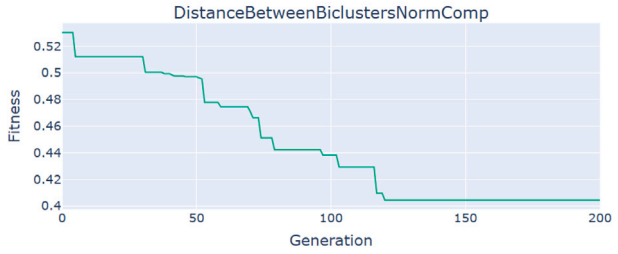
Fig. 11 provides information about the evolution of the external population of the self-configurator and the internal population of the winning supervised configuration. First, regarding the supervised



(a) Minimum values per generation for Adaptive bSIZE.



(b) Minimum values per generation for MSR.



(c) Minimum values per generation for bDIFF.

Fig. 8. Evolution of the minimum fitness values of each objective during the MOEBA-BIO run using complete representation and the new objective functions on the simulated dataset with an overlap level of 20 and seed 101.

phase, Fig. 11(a) shows the evolution of the clustering error metric as different sets of objectives and their subparameters are evaluated. The convergence of this curve indicates that the outer wrapper algorithm has reached a sufficient number of evaluations. Additionally, for this same phase, Fig. 11(b) shows the evolution of the objectives as a global count of their presence in the individuals of each generation. Two observations can be extracted from this graph: the convergence of the clustering error metric aligns perfectly with the clarification of the self-determined objectives, and the combined use of the Adaptive bSIZE, bVAR, rVAR, and Regulatory Coherence functions manage to maximize the precision of the solutions.

The appearance of the rVAR objective and the exclusion of bDIFF makes sense given the type of pattern typically observed in co-expression data. In the context of gene co-expression, it is common for co-expressed genes to maintain a constant relationship across experimental conditions, but with different levels of activation. This implies that the bicluster's rows may be at different levels, but maintain a consistent proportion across the columns of the bicluster. This characteristic makes the maximization of rVAR particularly suitable for this domain, while bDIFF, which favors rows with similar cell values, does not fit as well in this case.

Secondly, on the right side of Fig. 11, information about the unsupervised phase is shown, specifically the evolution of individuals in the internal population of the self-configurator corresponding to the winning individual from the supervised phase. In Fig. 11(c), the evolution of the signed change in the normalized hypervolume value with unitary reference is plotted, as different algorithms, parameters, and unsupervised subparameters are evaluated. Once again, the convergence of this curve ensures a sufficient number of evaluations in the internal wrapper algorithm of the self-configurator. Finally, from all the self-configured parameters, Fig. 11(d) depicts the evolution of the different candidate algorithms within the population. Of course, although two combinations may choose the same algorithm, they can have completely different subparameter values, influencing their performance. In other words, as with Fig. 11(b) for the supervised phase, Fig. 11(d) provides a simplified explanation of what occurred during the unsupervised phase. From the analysis of Fig. 11(d), it can be observed that the self-configurator converges decisively on the IBEA algorithm.

Finally, the winning configuration consists of the following fitness functions: BiclusterVarianceNorm (objectives summarized using the mean), RowVarianceNormComp (also summarized with the mean), BiclusterSizeNumBicsNormComp (coherence weight: 0.0802, objective summary: mean, row weight: 0.8746), and RegulatoryCoherenceNormComp. As for unsupervised parameters, a crossover probability of 63.79%, a mutation probability of 0.0287%, a population size of 100, and the IBEA-SingleThread algorithm (which has no subparameters) were determined. Regarding the operators, the only ones implemented so far were selected. For the crossover operator, PartiallyMappedCrossover, BicUniformCrossover, and CellUniformCrossover were used, while for the mutation operator, SwapMutation, BicUniformMutation and CellUniformMutation were selected.

In addition to the evolution of the internal and external wrapper populations, the self-configurator provides the results from the runs obtained by evaluating the winning solution vectors. That is, the execution for each instance of the winning configuration. This allows to represent in Fig. 12 the parallel coordinates of the approximate Pareto front obtained by the self-configured MOEBA-BIO in the first data matrix generated by the FABIA R package. The aim is to highlight the trade-offs between the different self-determined objectives and discard any possible redundancy during the optimization process.

5.3. Candidates comparison: autoconfigurator validation

In **the third phase** of experimentation, the effectiveness of the self-configurator was evaluated by comparing the winning configuration with other additional candidate configurations.

First, the winning configuration is compared with other candidate configurations from the supervised phase by measuring the clustering error complementary metric for each of them. These configurations exclude various key objective functions and introduce others initially discarded by the self-configurator. The results, presented in Table 7, show the median of the medians and the maximum of the maximums from each front for the complementary clustering error metric across the four instances of the gene expression dataset.

Overall, the winning configuration outperforms the candidates in most instances, achieving the best medians in instances 1 and 4, and the best maximum values in instances 1, 2, and 4. This indicates that including all self-determined objective functions (including regulatory coherence and row variance) is crucial for ensuring high performance in minimizing the complementary clustering error. Meanwhile, introducing other objectives discarded by the self-configurator also seems to reflect a decline in result accuracy.

Second, the winning configuration is compared with other candidates from the unsupervised phase. To do this, the hypervolume value is measured with respect to the reference front, constructed as the set of non-dominated solutions from all executions, which differs from the reference point used during the self-configurator.

Among the candidates, one configuration results from modifying the winning one solely by replacing the IBEA algorithm with the extended NSGA-III. Another configuration features a larger population size and higher operator probabilities using NSGA-II. Additionally, an intermediate configuration is evaluated with the SPEA2 algorithm, and a final configuration is tested with a crossover probability similar to the winning one but with a higher mutation rate using MOSA.

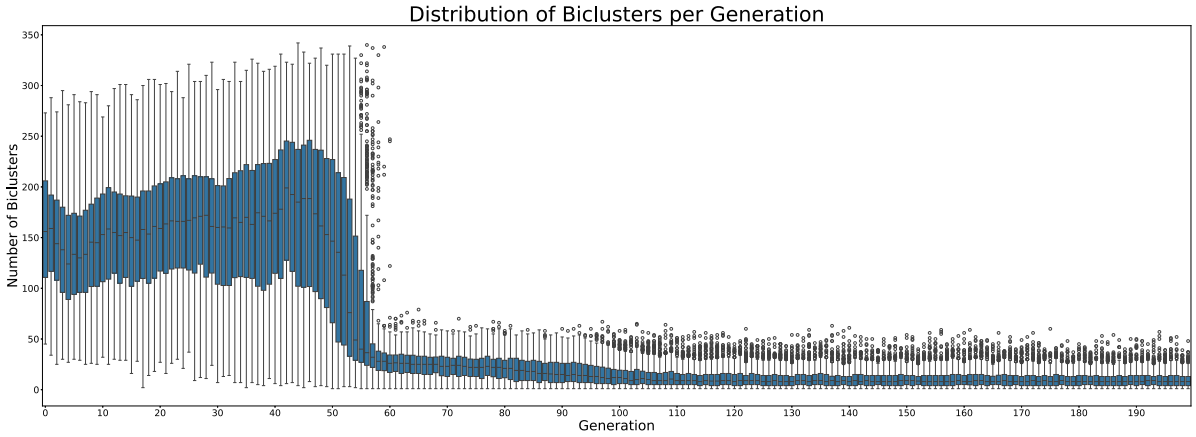
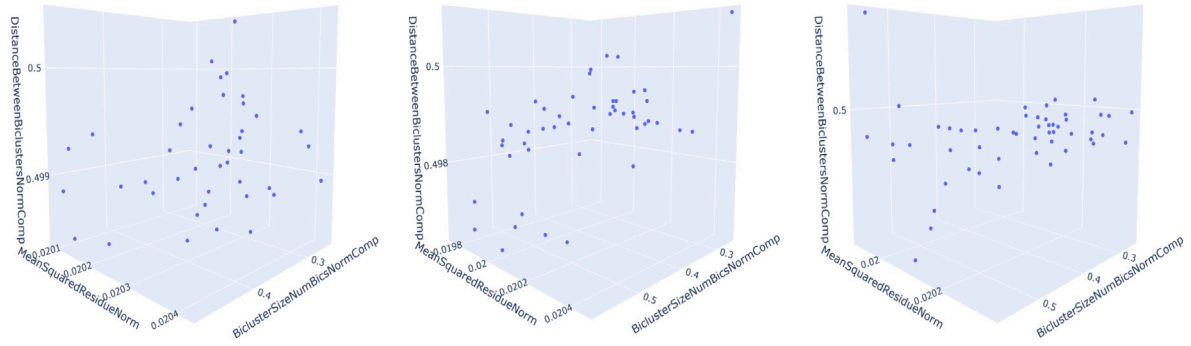


Fig. 9. Distribution of the number of biclusters contained in each individual per generation in the MOEBA-BIO run using complete representation and the new objective functions on the simulated dataset with an overlap level of 20 and seed 101.



(a) Noise level of 20 and seed 102. (b) Num of biclusters set to 3 and seed 103. (c) Overlap level of 20 and seed 102.

Fig. 10. Set of non-dominated solutions in Pareto front approximation from the final population obtained by MOEBA-BIO using complete representation and the new objective functions on the simulated dataset.

Table 7
Median of each front's medians and maximum of each front's maxima for the clustering error complementary metric after running with 5 replicates of the winning configuration and three other candidates on the gene expression dataset generated by FABIA.

Configuration	Instance 1		Instance 2		Instance 3		Instance 4	
	Median	Max	Median	Max	Median	Max	Median	Max
Winner	**0.0136**	**0.0268**	0.0164	**0.0262**	0.0094	0.0174	**0.0136**	**0.0220**
Winner - No Reg. Coherence	0.0099	0.0238	0.0146	0.0235	0.0089	**0.0200**	0.0097	0.0176
Winner - No Row Variance	0.0134	0.0237	**0.0192**	0.0257	0.0106	0.0194	0.0130	0.0204
Winner - Dist Between Bics	0.0130	0.0254	0.0182	0.0241	**0.0118**	0.0178	0.0135	0.0218

Table 8
Hypervolume values for different candidates of the unsupervised phase and the winning configuration. In particular, the median of 5 independent runs on the FABIA simulated dataset is presented.

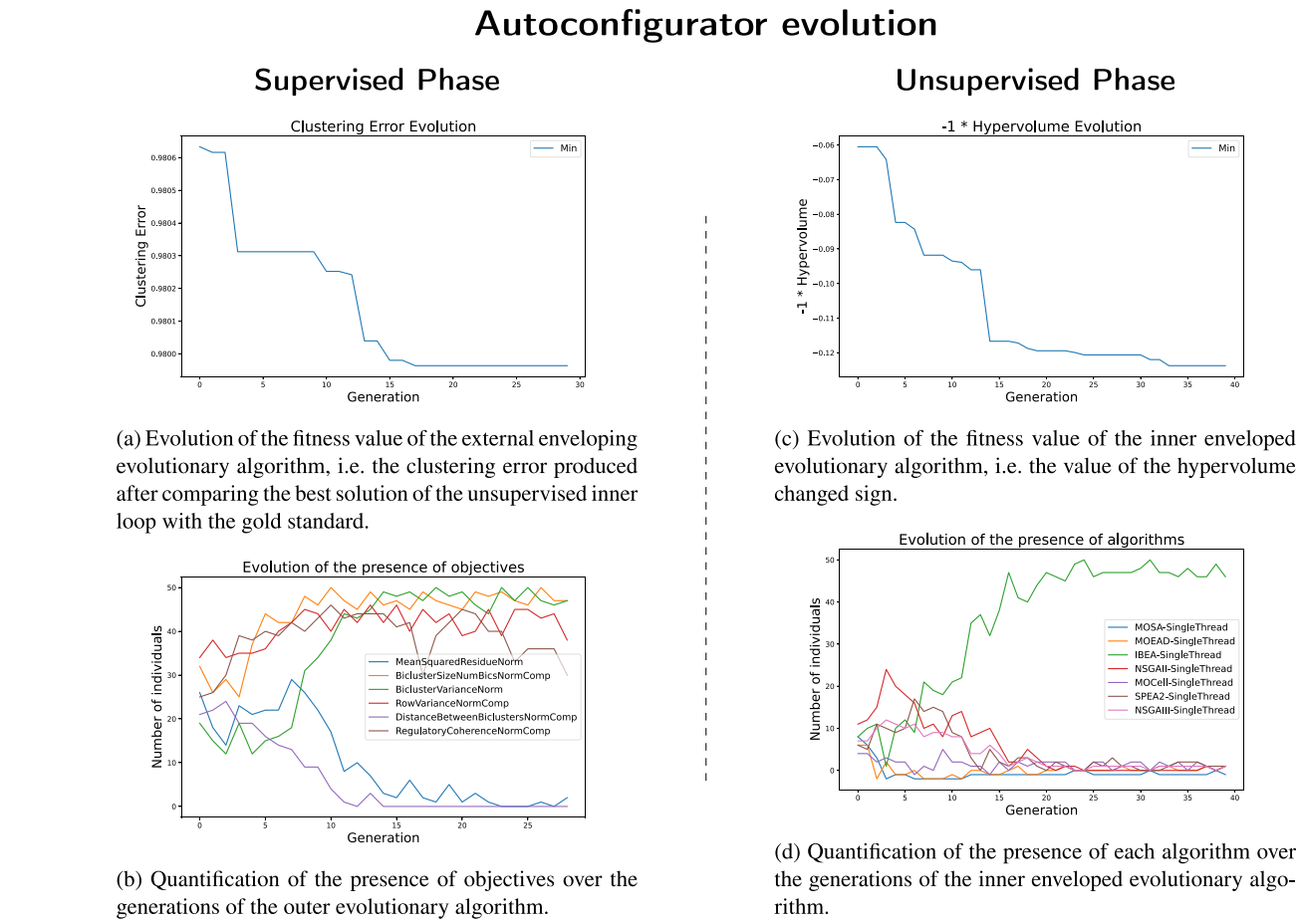
Configuration	Parameters				Hypervolume			
	Algorithm	Population size	Crossover probability	Mutation probability	Instance 1	Instance 2	Instance 3	Instance 4
Winner	IBEA	100	0.64	2.87e−04	**0.1058**	**0.0863**	**0.1378**	**0.0845**
Candidate-1	NSGAIII	100	0.64	2.87e−04	0.0924	0.0564	0.0826	0.0809
Candidate-2	NSGAII	500	0.90	0.10	0.0676	0.0447	0.0649	0.0573
Candidate-3	SPEA2	300	0.75	0.05	0.0699	0.0503	0.0705	0.0700
Candidate-4	MOSA	400	0.60	0.25	0.0483	0.0394	0.0466	0.0457

Table 8 presents the median of these values for each candidate, clearly demonstrating the dominance of the winning configuration over the others.

Finally, the analysis of these results using 5 independent runs for each instance further justifies the decision to discard such repetitions in the self-configuration process, prioritizing computational feasibility without compromising the quality of the self-configuration.

5.4. Algorithmic comparison

The fourth phase of experimentation is oriented to compare the results of the self-configured MOEBA-BIO from the previous phase with various state-of-the-art techniques. Fig. 13 shows the results obtained by the different techniques for the supervised metrics: Clustering Error [67], Recovery [68], Relevance [68], and Ayadi's Score [69].



(a) Evolution of the fitness value of the external enveloping evolutionary algorithm, i.e. the clustering error produced after comparing the best solution of the unsupervised inner loop with the gold standard.

(b) Quantification of the presence of objectives over the generations of the outer evolutionary algorithm.

(c) Evolution of the fitness value of the inner enveloped evolutionary algorithm, i.e. the value of the hypervolume changed sign.

(d) Quantification of the presence of each algorithm over the generations of the inner enveloped evolutionary algorithm.

Fig. 11. Tracking the execution of the autoconfigurator for gene co-expression data. The evolution of the outer loop (supervised phase) is shown on the left and the inner loop (unsupervised phase) on the right.

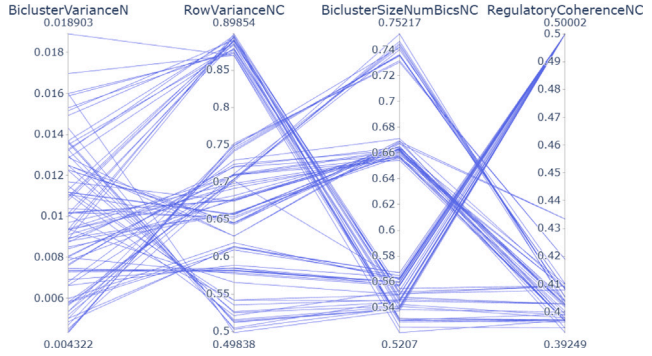


Fig. 12. Parallel coordinate plot for the front obtained by MOEBA-BIO using complete representation and the best configuration found by the autoconfigurator for the first instance of the gene expression data simulator.

As discussed in the experimentation section, clustering error is the most suitable and comprehensive metric to validate this proposal's contributions. However, two additional individual metrics have been implemented to demonstrate that the performance in terms of clustering error is not due to the specification of an inappropriate number of biclusters in specific state-of-the-art techniques that require this parameter.

As shown in Fig. 13, both the best solution of MOEBA-BIO and the median of the medians of the five fronts generated for each instance are located at the top in all metrics, particularly excelling in Clustering Error, the primary metric of this study.

It is important to note that the median of the medians, despite outperforming most state-of-the-art techniques, represents a random selection within the approximate Pareto front generated by the algorithm. This should not occur in an ideal context, as selecting the most appropriate solution within the front is a task for the domain expert, who, with a better understanding of the search space, can make more informed decisions than a random choice.

According to a Friedman statistical rank, the best solution of MOEBA-BIO leads the ranking in all metrics, followed in all cases by its median. It is worth mentioning that in the ScoreAyadi metric, the median of MOEBA-BIO shares second place with a value of 2.75, alongside the OPSM technique, which does not show outstanding performance in any of the other metrics. OPSM is well-suited to the ScoreAyadi metric, which is why, in some instances, it achieves better values than the median of MOEBA-BIO, thus affirming the validity of the No Free Lunch theorem in this context.

Finally, Fig. 14 presents the execution times required for each instance using each methodology on the same machine with eight cores and 64 GB of RAM. In the case of MOEBA-BIO, an additional, particularly costly internal step has been timed separately to assess its impact on the total execution time and provide a more detailed performance comparison of the algorithm. This step corresponds to the initial inference of the GRN required for the *regulatory coherence* objective, performed by GENIE3 [48] (gray fragment).

Although the execution times for MOEBA-BIO are higher than those of other methodologies, it is important to consider that (1) a significant portion of the execution time is due to the GRN inference using GENIE3, a component that users can replace with a lighter technique if preferred; (2) previous results demonstrate that our algorithm

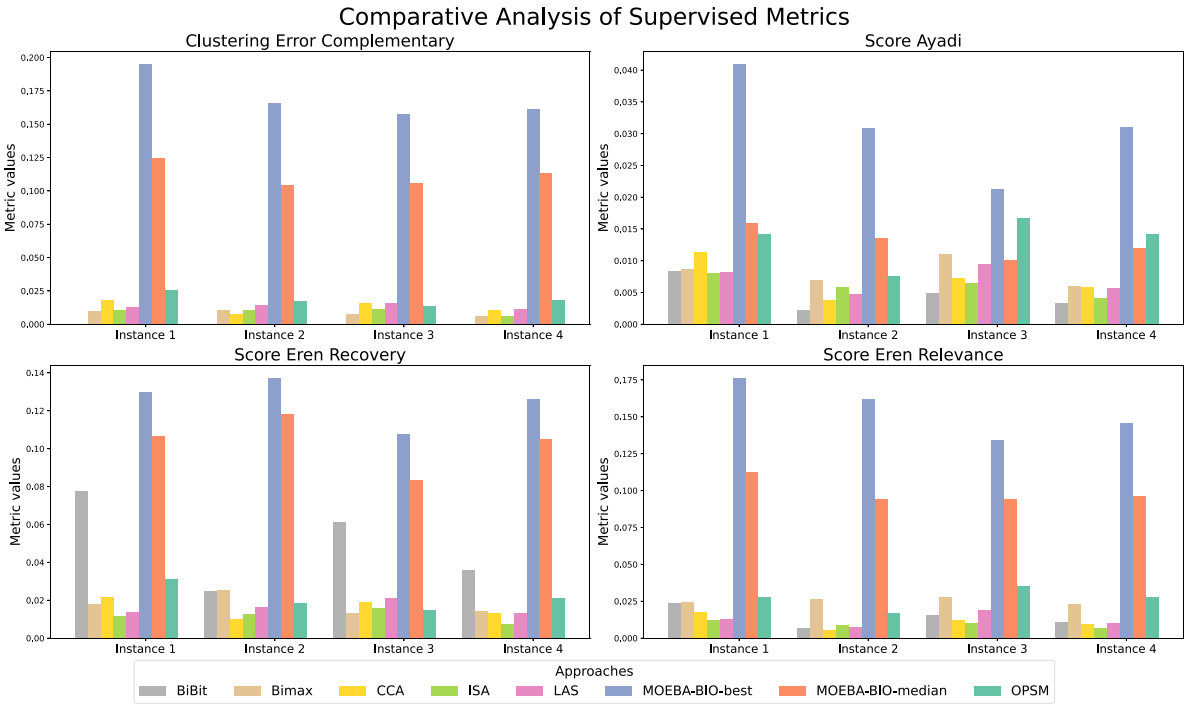


Fig. 13. Comparison of the self-generated MOEBA-BIO algorithm for the gene co-expression domain concerning different recognized state-of-the-art techniques. For each instance, the values of the supervised metrics: Clustering Error, ScoreAyadi and ScoreEren (recovery and relevance) are compared.

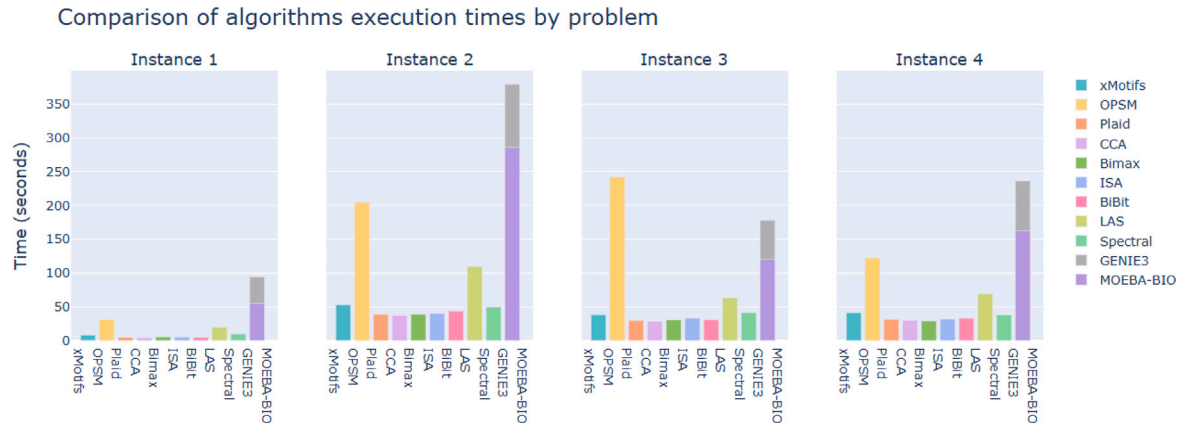


Fig. 14. Comparison of the time required by each methodology for each instance of the synthetic dataset generated by the FABIA R package. The time required by MOEBA-BIO is divided into two fragments, detailing the time required for the initial GRN inference performed by GENIE3 for the regulatory coherence objective.

achieves higher accuracy in identifying relevant biclusters; and (3) unlike the other compared methodologies, MOEBA-BIO incorporates the self-determination of the number of biclusters, which introduces additional computational complexity.

5.5. Functional enrichment comparison

The fifth phase compares the results obtained by different biclustering methodologies on real gene expression datasets by measuring the proportion of significantly enriched biclusters under various significance thresholds. This proportion ranges from 0 to 1, reaching its maximum value when all inferred biclusters exceed the significance threshold and its minimum when none do.

Fig. 15 presents these proportions for α values of 0.05, 0.005, and 0.00001. As observed, both the best solution of MOEBA-BIO and the median of the front achieve relatively high proportions, surpassing a significant number of state-of-the-art methodologies.

Table 9
Friedman mean rank with Holm's adjusted p values (0.05) for the ratio of enriched biclusters with $alpha = 0.05$.

Enriched biclusters ratio ($alpha = 0.05$)		
Technique	$Friedman's\{Rank\}$	$Holm's\{Adj - p\}$
Max MOEBA-BIO	**3.03**	–
Median MOEBA-BIO	3.29	1.17
Bimax	3.65	1.17
BiBit	4.41	0.85
Spectral	4.56	0.85
OPSM	4.59	0.85
Plaid	6.29	2.46e−02
LAS	7.56	4.79e−04
CCA	8.65	6.31e−06
ISA	9.68	4.61e−08
xMotifs	10.29	1.70e−09

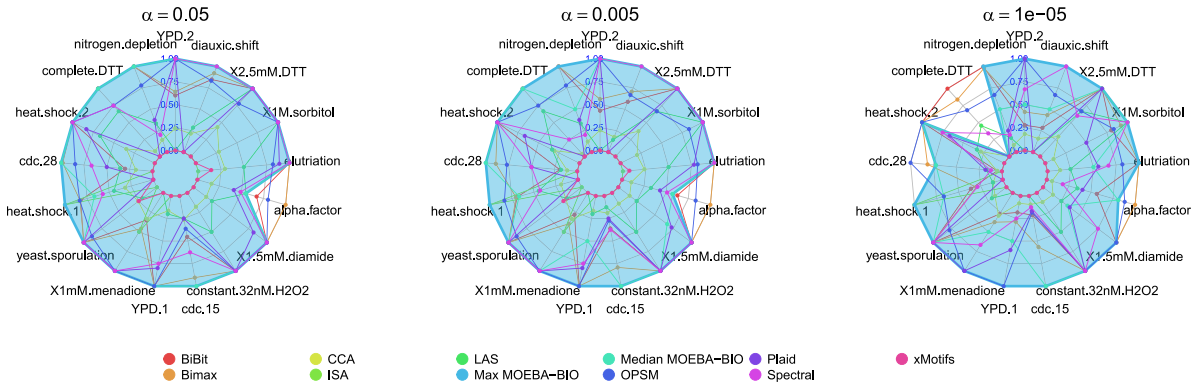


Fig. 15. Comparison of the proportion of enriched biclusters obtained by each methodology on the real-world gene expression dataset, evaluated under different significance thresholds ($\alpha \in \{0.05, 0.005, 0.00001\}$). Higher proportions indicate a stronger functional coherence among the genes within the detected biclusters. The values obtained for the best solution of the MOEBA-BIO Pareto front plot a colored area that allows visualizing the dominance of the proposal.

Table 10
Friedman mean rank with Holm's adjusted p values (0.05) for the ratio of enriched biclusters with $alpha = 0.005$.

Enriched biclusters ratio ($alpha = 0.005$)		
Technique	$Friedman's\{Rank\}$	$Holm's\{Adj - p\}$
Max MOEBA-BIO	**2.82**	–
Bimax	3.47	0.79
Median MOEBA-BIO	3.79	0.79
BiBit	4.41	0.67
OPSM	4.47	0.67
Spectral	4.53	0.67
Plaid	6.00	3.14e−02
LAS	7.94	4.79e−05
CCA	8.65	2.46e−06
ISA	9.68	1.53e−08
xMotifs	10.24	7.25e−10

Table 11
Friedman mean rank with Holm's adjusted p values (0.05) for the ratio of enriched biclusters with $alpha = 0.00001$.

Enriched biclusters ratio ($alpha = 0.00001$)		
Technique	$Friedman's\{Rank\}$	$Holm's\{Adj - p\}$
Max MOEBA-BIO	**2.76**	–
OPSM	3.53	0.88
Bimax	3.65	0.88
Median MOEBA-BIO	4.32	0.51
BiBit	4.59	0.44
Spectral	5.32	0.12
Plaid	5.65	6.77e−02
LAS	7.74	8.72e−05
CCA	8.56	2.81e−06
ISA	9.68	1.11e−08
xMotifs	10.21	6.10e−10

However, to provide greater statistical rigor to this comparison, Tables 9, 10, and 11 present the results of applying the Friedman statistical ranking with non-parametric Holm tests for α values of 0.05, 0.005, and 0.00001, respectively. These tables show that, although its dominance does not reach absolute statistical significance overall proposals, the best solution of MOEBA-BIO consistently ranks first across all α thresholds. Additionally, the median of the front achieves promising positions, frequently appearing on the ranking podium.

Finally, it is important to highlight that functional enrichment analysis, widely used to validate the quality of biclustering algorithms on unlabeled expression data, does not consider the bi-dimensionality of the results. That is, while this approach validates the coexistence of certain genes within biclusters, it does not consider the conditions under which they have been grouped. Therefore, the previous experimental phase with simulated data is essential, as it provides a more precise validation through metrics that consider both bicluster dimensions and evaluate MOEBA-BIO's ability to self-determine the number of biclusters.

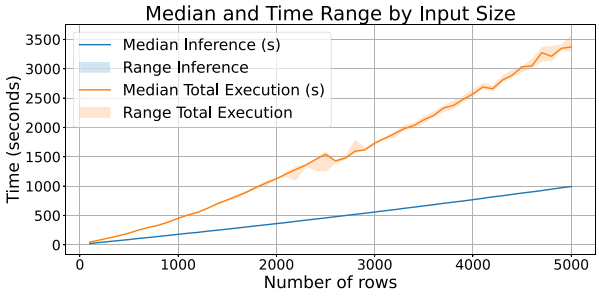


Fig. 16. Median and range of execution times (in seconds) according to input matrix size (number of rows). The plot differentiates between the total execution time of the biclustering process and the inference time required by GENIE3 to instantiate the *Regulatory coherence* objective.

5.6. Scalability study

The sixth phase of the experimentation involved a total of 250 executions of the self-constructed algorithm for the gene co-expression domain. The analysis of execution times enabled the generation of the plot shown in Fig. 16, which illustrates the evolution of total execution time and the time required by GENIE3 at the beginning of the *Regulatory Coherence* objective instantiation, as a function of the number of input rows.

The results show that both the total execution time and the time specifically required by the inference technique exhibit exponential growth as the matrix size increases. As observed, the time required by GENIE3 represents a significant portion of the total execution time. Consequently, the inference technique can be considered a relevant factor in terms of scalability.

This behavior suggests that, in scenarios where execution time is a priority, a viable alternative would be to replace the current inference technique with a more efficient one or one better suited to the scale of the available data. This is feasible thanks to the modularity of the constructed framework, which not only allows the configuration of algorithms tailored to the application domain, but also provides the flexibility to adapt the process according to computational constraints and the size of the input data.

6. Discussion

The results presented in the previous section justify the fulfillment of each of the contributions outlined in the introduction of this study. First, the new complete encoding designed in this work has demonstrated, in the results of the first phase of experimentation (Table 6, Figs. 6, and 7), its ability to overcome the limitations of partial encoding regarding accuracy, redundancy, and interpretability. This improvement surpasses many of the more traditional approaches found in the literature [12]. Additionally, this new encoding has shown its capacity to address shortcomings observed in other more recent global-perspective encodings that also include multiple biclusters in a single individual. For instance, the indirect learning of the appropriate number of biclusters (shown in Fig. 9) eliminates the need for a specific predefined value as required in [36]. Moreover, the direct equivalence between an individual and the actual solution to the problem (illustrated in Fig. 7) is an advantage compared to the encoding proposed in [35]. Finally, the set of different strategies for summarizing individual objectives allow penalizing imbalances between the biclusters within a single solution (as observed in Figs. 7(e) and 7(f)), which, for example, were not considered in [37].

Furthermore, the design of new objectives with a holistic perspective, not previously observed in the state-of-the-art, has proven to further reduce clustering error of the results, both individually and through their combined use. These objectives have contributed to improving general-type implementations (as seen in Table 6), as well as to those tailored to specific biological contexts (as seen in Table 7). Their joint integration enables the proper evolution of individuals (observed in Fig. 8) and a flexible search space adapted to the problem (illustrated in Fig. 10) with a well-balanced trade-off between objectives (depicted in Fig. 12).

The results of the second phase of experimentation, validated in the third phase, have demonstrated in Fig. 11 that the supervised outer loop of the autoconfigurator is not only novel compared to other proposals in the literature [49], but also enables the self-construction of new biclustering algorithms whose design is guided by the biological context of the problem. The academic and practical implications of this achievement are promising. It cannot only inspire new autoconfigurators in other real-world fields, such as gene regulatory network inference [20], but also it can be directly applied to design biclustering algorithms in other biological application domains, such as clinical data [8] or biomedical imaging [9].

The results from the algorithmic comparison phase demonstrate that the algorithm self-constructed by MOEBA-BIO for the specific domain of gene co-expression significantly surpasses the accuracies (clustering error, score Ayadi, and score Eren recovery and relevance) of a wide range of well-recognized state-of-the-art algorithms (as seen in Fig. 13), such as CCA [22], OPSM [23], xMOTIFs [24], ISA [25], LAS [26], Bimax [27], BiBit [28], Plaid [29], and Spectral [30]. An improvement in quality which, together with the ability to self-determine the number of biclusters, has not entailed an excessively high computational cost (as seen in Fig. 14).

In the fifth phase of experimentation, the algorithm self-constructed by MOEBA-BIO for the specific domain of gene co-expression has significantly outperformed a biological validation procedure (as seen in Fig. 15). Specifically, the proposed approach has topped all Friedman rankings, positioning itself as the methodology that obtains the highest proportion of enriched biclusters in its results (as seen in Table 9, Table 10, and Table 11).

Finally, in the last phase, the scalability study highlights that the inference step required to instantiate the *Regulatory coherence* objective using GENIE3 accounts for a substantial portion of the total execution time. Although its growth is slightly less steep than that of the overall process, it remains a relevant bottleneck when scaling to larger datasets. Nevertheless, this limitation can be effectively mitigated thanks to the modular and flexible design of the proposed framework, which allows for the substitution of the inference method with more efficient alternatives adapted to the computational constraints of each scenario.

7. Conclusions

This document has presented MOEBA-BIO, a framework for designing multi-objective evolutionary algorithms for biclustering biological data. MOEBA-BIO has rigorously demonstrated the significant benefits of its contributions and the ability of each to facilitate the integration of domain-specific biomedical knowledge: a new complete representation, indirect learning for self-determination of the number of biclusters, context-guided parameter self-configuration, the design of more realistic objective functions with a holistic perspective that capture information from the specific biomedical domain of the data, among others.

The potential of MOEBA-BIO has been demonstrated both on simulated generic data and domain-specific gene co-expression data. In general, the proposed new representation has proven to overcome limitations imposed by traditional partial encodings, such as redundancy and heterogeneity of bicluster qualities. Additionally, with the same data, the framework has demonstrated the ability to integrate unattainable global perspective objectives with traditional representations, with both individual and joint execution showing significant improvements in result accuracy. In the gene co-expression domain, the self-configurator has proven its ability to automatically design algorithms guided by the biomedical context of the data, as well as the improvements in accuracy that the injection of knowledge through the design of domain-specific objective functions can offer. Finally, the contribution of the self-generated evolutionary algorithm has been validated against a broad and representative set of state-of-the-art techniques, yielding highly competitive results that further highlight the great potential of this proposal.

Lastly, it is worth mentioning that the strategy applied in the implementation of MOEBA-BIO allows for its easy extension through the seamless incorporation of new representations, more sophisticated operators, domain-specific objectives, handling of heterogeneous data, and many other options, providing a wide range of possibilities for defining future directions for this work. Among the most direct future lines, the self-construction of specialized algorithms for new domains is contemplated, following a thorough study of each domain and the design of pertinent objective functions.

CRediT authorship contribution statement

Adrián Segura-Ortiz: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Adán José-García:** Supervision, Data curation. **Laetitia Jourdan:** Supervision, Project administration. **José García-Nieto:** Writing – review & editing, Supervision.

Funding

This work has been partially funded by grant (funded by MCIN/AEI/10.13039/501100011033/) PID2020-112540RB-C41, AETHER-UMA (A smart data holistic approach for context-aware data analytics: semantics and context exploitation) and the Junta de Andalucia, Spain, under contract QUAL21 010UMA. Funding for open access charge: Universidad de Málaga/CBUA. Adrián Segura-Ortiz is supported by Grant FPU21/03837 (Spanish Ministry of Science, Innovation and Universities).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would like to thank Professor Julie Jacques for warmly welcoming us at the CRIStAL research centre of the University of Lille, as well as for her guidance and recommendations based on her extensive experience in the field of biclustering. In addition, we thank Professor Ismael Navas Delgado for his indications and explanation of concepts related to gene co-expression.

References

[1] Eduardo N. Castanho, Helena Aidos, Sara C. Madeira, Biclustering data analysis: a comprehensive survey, Brief. Bioinform. 25 (4) (2024).
[2] Juan Xie, Anjun Ma, Anne Fennell, Qin Ma, Jing Zhao, It is time to apply biclustering: a comprehensive review of biclustering applications in biological and biomedical data, Brief. Bioinform. 20 (4) (2019) 1450–1465.
[3] Beatriz Pontes, Raúl Giráldez, Jesús S. Aguilar-Ruiz, Biclustering on expression data: A review, J. Biomed. Inform. 57 (2015) 163–180.
[4] Kath Nicholls, Chris Wallace, Comparison of sparse biclustering algorithms for gene expression datasets, Briefings Bioinform. 22 (6) (2021) bbab140.
[5] Juan Xie, Anjun Ma, Yu Zhang, Bingqiang Liu, Sha Cao, Cankun Wang, Jennifer Xu, Chi Zhang, Qin Ma, Qubic2: a novel and robust biclustering algorithm for analyses and interpretation of large-scale rna-seq data, Bioinformatics 36 (4) (2020) 1143–1149.
[6] Maxence Vandromme, Julie Jacques, Julien Taillard, Laetitia Jourdan, Clarisse Dhaenens, A biclustering method for heterogeneous and temporal medical data, IEEE Trans. Knowl. Data Eng. 34 (2) (2020) 506–518.
[7] Maxence Vandromme, Julie Jacques, Julien Taillard, Laetitia Jourdan, Clarisse Dhaenens, A scalable biclustering method for heterogeneous medical data, in: International Workshop on Machine Learning, Optimization, and Big Data, Springer, 2016.
[8] Adán José-García, Julie Jacques, Clément Chauvet, Vincent Sobanski, Clarisse Dhaenens, Hbic: A biclustering algorithm for heterogeneous datasets, 2024, arXiv preprint arXiv:2408.13217.
[9] Himanshu Mittal, Avinash Chandra Pandey, Mukesh Saraswat, Sumit Kumar, Raju Pal, Garv Modwel, A comprehensive survey of image segmentation: clustering methods, performance parameters, and benchmark datasets, in: Multimedia Tools and Applications, 2022, pp. 1–26.
[10] Md Abdur Rahaman, Jessica A. Turner, Cota Navin Gupta, Srinivas Rachakonda, Jiayu Chen, Jingyu Liu, Theo GM Van Erp, Steven Potkin, Judith Ford, Daniel Mathalon, et al., N-bic: A method for multi-component and symptom biclustering of structural mri data: Application to schizophrenia, IEEE Trans. Biomed. Eng. 67 (1) (2019) 110–121.
[11] Victor A. Padilha, Ricardo JGB. Campello, A systematic comparative evaluation of biclustering techniques, BMC Bioinformatics 18 (2017) 1–25.
[12] Adán José-García, Julie Jacques, Vincent Sobanski, Clarisse Dhaenens, Metaheuristic biclustering algorithms: From state-of-the-art to future opportunities, ACM Comput. Surv. 56 (3) (2023) 1–38.
[13] Adán José-García, Julie Jacques, Vincent Sobanski, Clarisse Dhaenens, Biclustering algorithms based on metaheuristics: a review. Metaheuristics for machine learning: new advances and tools, 2022, pp. 39–71.
[14] Zhoufan Kong, Qinghua Huang, Xuelong Li, Bi-phase evolutionary biclustering algorithm with the nsga-ii algorithm, in: 2019 IEEE 4th International Conference on Advanced Robotics and Mechatronics, ICARM, IEEE, 2019, pp. 146–149.
[15] Sudipta Acharya, Sriparna Saha, Pracheta Sahoo, Bi-clustering of microarray data using a symmetry-based multi-objective optimization framework, Soft Comput. 23 (2019) 5693–5714.
[16] E.G. Talbi, Metaheuristics: From Design To Implementation, Vol. 2, John Wiley & Sons google schola, 2009, pp. 268–308.
[17] Marta D.M. Noronha, Rui Henriques, Sara C. Madeira, Luis E. Zárate, Impact of metrics on biclustering solution and quality: A review, Pattern Recognit. 127 (2022) 108612.
[18] Adrián Segura-Ortiz, José García-Nieto, José F. Aldana-Montes, Exploiting medical-expert knowledge via a novel memetic algorithm for the inference of gene regulatory networks, in: International Conference on Computational Science, Springer, 2024, pp. 3–17.
[19] Adrián Segura-Ortiz, José García-Nieto, José F. Aldana-Montes, Ismael Navas-Delgado, Geneci: A novel evolutionary machine learning consensus-based approach for the inference of gene regulatory networks, Comput. Biol. Med. 155 (2023) 106653.
[20] Adrián Segura-Ortiz, José García-Nieto, José F. Aldana-Montes, Ismael Navas-Delgado, Multi-objective context-guided consensus of a massive array of techniques for the inference of gene regulatory networks, Comput. Biol. Med. 179 (2024) 108850.
[21] Ke Shang, Hisao Ishibuchi, Linjun He, Lie Meng Pang, A survey on the hypervolume indicator in evolutionary multiobjective optimization, IEEE Trans. Evol. Comput. 25 (1) (2020) 1–20.
[22] Yizong Cheng, George M. Church, Biclustering of expression data, in: Ismb, Vol. 8, 2000, pp. 93–103.
[23] Amir Ben-Dor, Benny Chor, Richard Karp, Zohar Yakhini, Discovering local structure in gene expression data: the order-preserving submatrix problem, in: Proceedings of the Sixth Annual International Conference on Computational Biology, 2002, pp. 49–57.
[24] T.M. Murali, Simon Kasif, Extracting conserved gene expression motifs from gene expression data, in: Biocomputing 2003, World Scientific, 2002, pp. 77–88.
[25] Sven Bergmann, Jan Ihmels, Naama Barkai, Iterative signature algorithm for the analysis of large-scale gene expression data, Phys. Rev. E 67 (3) (2003) 031902.
[26] Andrey A. Shabalin, Victor J. Weigman, Charles M. Perou, Andrew B. Nobel, Finding large average submatrices in high dimensional data, in: The Annals of Applied Statistics, 2009, pp. 985–1012.
[27] Amela Prelić, Stefan Bleuler, Philip Zimmermann, Anja Wille, Peter Bühlmann, Wilhelm Gruissem, Lars Hennig, Lothar Thiele, Eckart Zitzler, A systematic comparison and evaluation of biclustering methods for gene expression data, Bioinformatics 22 (9) (2006) 1122–1129.
[28] Domingo S. Rodriguez-Baena, Antonio J. Perez-Pulido, Jesus S. Aguilar-Ruiz, A biclustering algorithm for extracting bit-patterns from binary datasets, Bioinformatics 27 (19) (2011) 2738–2745.
[29] Heather Turner, Trevor Bailey, Wojtek Krzanowski, Improved biclustering of microarray data demonstrated through systematic performance tests, Comput. Statist. Data Anal. 48 (2) (2005) 235–254.
[30] Yuval Kluger, Ronen Basri, Joseph T. Chang, Mark Gerstein, Spectral biclustering of microarray data: coclustering genes and conditions, Genome Res. 13 (4) (2003) 703–716.
[31] Federico Divina, Francisco A. Goméz Vela, Miguel García Torres, Biclustering of smart building electric energy consumption data, Appl. Sci. 9 (2) (2019) 222.
[32] Maryam Golchin, Seyed Hashem Davarpanah, Alan Wee-Chung Liew, Biclustering analysis of gene expression data using multi-objective evolutionary algorithms, in: 2015 International Conference on Machine Learning and Cybernetics, ICMLC, Vol. 2, IEEE, 2015, pp. 505–510.
[33] Ons Maatouk, Emna Ayari, Hend Bouziri, Wassim Ayadi, Bobea: a bi-objective biclustering evolutionary algorithm for genome-wide association analysis, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, 2022, pp. 344–347.
[34] Pracheta Sahoo, Sudipta Acharya, Sriparna Saha, Automatic generation of biclusters from gene expression data using multi-objective simulated annealing approach, in: 2016 23rd International Conference on Pattern Recognition, ICPR, IEEE, 2016, pp. 2174–2179.
[35] Meriem Bousselmi, Slim Bechikh, Chih-Cheng Hung, Lamjed Ben Said, Bimock: A multi-objective evolutionary algorithm for bi-clustering with automatic determination of the number of bi-clusters, in: Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November (2017) 14–18, Proceedings, Part IV 24, Springer, 2017, pp. 366–376.
[36] Maryam Golchin, Alan Wee Chung Liew, Parallel biclustering detection using strength pareto front evolutionary algorithm, Inform. Sci. 415 (2017) 283–297.
[37] Naveen Saini, Sriparna Saha, Chirag Soni, Pushpak Bhattacharyya, Automatic evolution of bi-clusters from microarray data using self-organized multi-objective evolutionary algorithm, Appl. Intell. 50 (2020) 1027–1044.
[38] Qingfu Zhang, Hui Li, Moea/d: A multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[39] Juan J. Durillo, Antonio J. Nebro, Jmetal: A java framework for multi-objective optimization, Adv. Eng. Softw. 42 (10) (2011) 760–771.
[40] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, TAMT Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[41] Kalyanmoy Deb, Himanshu Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2013) 577–601.
[42] Antonio J. Nebro, Juan J. Durillo, Francisco Luna, Bernabé Dorronsoro, Enrique Alba, Mocell: A cellular genetic algorithm for multiobjective optimization, Int. J. Intell. Syst. 24 (7) (2009) 726–746.
[43] Marco Laumanns, Spea2: Improving the Strength Pareto Evolutionary Algorithm, Technical Report, Vol. 35, Gloriastrasse, 2001.
[44] Eckart Zitzler, Simon Künzli, Indicator-based selection in multiobjective search, in: International Conference on Parallel Problem Solving from Nature, Springer, 2004, pp. 832–842.
[45] A. Suppapitnarm, Keith A. Seffen, Geoff T. Parks, P.J. Clarkson, A simulated annealing algorithm for multiobjective optimization, Eng. Optim. 33 (1) (2000) 59–85.
[46] Antonio J. Nebro, Juan José Durillo, Jose Garcia-Nieto, CA Coello Coello, Francisco Luna, Enrique Alba, Smpso: A new pso-based metaheuristic for multi-objective optimization, in: 2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making, MCDM, IEEE, 2009, pp. 66–73.
[47] Wencheng Yin, Luis Mendoza, Jimena Monzon-Sandoval, Araxi O. Urrutia, Humberto Gutierrez, Emergence of co-expression in gene regulatory networks, PloS One 16 (4) (2021) e0247671.

[48] Vân Anh Huynh-Thu, Alexandre Irrthum, Louis Wehenkel, Pierre Geurts, Inferring regulatory networks from expression data using tree-based methods, PloS One 5 (9) (2010) e12776.
[49] José F. Aldana-Martín, Juan J. Durillo, Antonio J. Nebro, Evolver: Meta-optimizing multi-objective metaheuristics, SoftwareX 24 (2023) 101551.
[50] Jianjun Sun, Qinghua Huang, Two stages biclustering with three populations, Biomed. Signal Process. Control. 79 (2023) 104182.
[51] Laizhong Cui, Sudipta Acharya, Sumit Mishra, Yi Pan, Joshua Zhexue Huang, Mmco-clus–an evolutionary co-clustering algorithm for gene selection, IEEE Trans. Knowl. Data Eng. 34 (9) (2020) 4371–4384.
[52] Khedidja Seridi, Laetitia Jourdan, El-Ghazali Talbi, Using multiobjective optimization for biclustering microarray data, Appl. Soft Comput. 33 (2015) 239–249.
[53] Federico Divina, Beatriz Pontes, Raúl Giráldez, Jesús S. Aguilar-Ruiz, An effective measure for assessing the quality of biclusters, Comput. Biol. Med. 42 (2) (2012) 245–256.
[54] Khedidja Seridi, Laetitia Jourdan, El-Ghazali Talbi, Multi-objective evolutionary algorithm for biclustering in microarrays data, in: 2011 IEEE Congress of Evolutionary Computation, CEC, IEEE, 2011, pp. 2593–2599.
[55] Cristian Andrés Gallo, Jessica Andrea Carballido, Ignacio Ponzoni, Microarray biclustering: A novel memetic approach based on the pisa platform, in: Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics: 7th European Conference, EvoBIO 2009 TÜbingen, Germany, April (2009) 15-17 Proceedings 7, Springer, 2009, pp. 44–55.
[56] Ujjwal Maulik, Anirban Mukhopadhyay, Sanghamitra Bandyopadhyay, Finding multiple coherent biclusters in microarray data using variable string length multiobjective genetic algorithm, IEEE Trans. Inf. Technol. Biomed. 13 (6) (2009) 969–975.
[57] Guilherme Palermo Coelho, de Fabrício Olivetti França, Von Fernando J. Zuben, Multi-objective biclustering: When non-dominated solutions are not enough, in: Journal of Mathematical Modelling and Algorithms, Vol. 8, 2009, pp. 175–202.
[58] Junwan Liu, Zhoujun Li, Xiaohua Hu, Yiming Chen, Multi-objective ant colony optimization biclustering of microarray data, in: 2009 IEEE International Conference on Granular Computing, IEEE, 2009, pp. 424–429.
[59] Mohsen Lashkargir, S. Amirhassan Monadjemi, Ahmad Baraani Dastjerdi, A new biclustering method for gene expersion data based on adaptive multi objective particle swarm optimization, in: 2009 Second International Conference on Computer and Electrical Engineering, vol. 1, IEEE, 2009, pp. 559–563.
[60] Junwan Liu, Zhoujun Li, Xiaohua Hu, Yiming Chen, Biclustering of microarray data with mospo based on crowding distance, in: BMC Bioinformatics, Vol. 10, Springer, 2009, pp. 1–10.
[61] Junwan Liu, Zhoujun Li, Feifei Liu, Yiming Chen, Multi-objective particle swarm optimization biclustering of microarray data, in: 2008 IEEE International Conference on Bioinformatics and Biomedicine, IEEE, 2008, pp. 363–366.
[62] Ujjwal Maulik, Anirban Mukhopadhyay, Sanghamitra Bandyopadhyay, Michael Q. Zhang, Xuegong Zhang, Multiobjective fuzzy biclustering in microarray data: method and a new performance measure, in: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 1536–1543.
[63] Federico Divina, Jesús S. Aguilar-Ruiz, A multi-objective approach to discover biclusters in microarray data, in: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, 2007, pp. 385–392.
[64] Sushmita Mitra, Haider Banka, Multi-objective evolutionary biclustering of gene expression data, Pattern Recognit. 39 (12) (2006) 2464–2477.
[65] Eduardo N. Castanho, João P. Lobo, Rui Henriques, Sara C. Madeira, G-bic: generating synthetic benchmarks for biclustering, BMC Bioinformatics 24 (1) (2023) 457.
[66] David J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman and hall/CRC, 2003.
[67] Anne Patrikainen, Marina Meila, Comparing subspace clusterings, IEEE Trans. Knowl. Data Eng. 18 (7) (2006) 902–916.
[68] Kemal Eren, Mehmet Deveci, Onur Küçüktunç, Ümit V. Çatalyürek, A comparative analysis of biclustering algorithms for gene expression data, Brief. Bioinform. 14 (3) (2013) 279–292.
[69] Wassim Ayadi, Ons Maatouk, Hend Bouziri, Evolutionary biclustering algorithm of gene expression data, in: 2012 23rd International Workshop on Database and Expert Systems Applications, IEEE, 2012, pp. 206–210.
[70] Sepp Hochreiter, Ulrich Bodenhofer, Martin Heusel, Andreas Mayr, Andreas Mitterecker, Adetayo Kasim, Tatsiana Khamiakova, Suzy Van Sanden, Dan Lin, Willem Talloen, et al., Fabia: factor analysis for bicluster acquisition, Bioinformatics 26 (12) (2010) 1520–1527.
[71] Victor A. Padilha, Ricardo J.G.B. Campello, A systematic comparative evaluation of biclustering techniques, BMC Bioinformatics 18 (1) (2017) 55.
[72] Pablo A. Jaskowiak, Ricardo J.G.B. Campello, Ivan G. Costa, Proximity measures for clustering gene expression microarray data: a validation methodology and a comparative analysis, IEEE/ ACM Trans. Comput. Biol. Bioinform. 10 (4) (2013) 845–857.
[73] Paul T. Spellman, Gavin Sherlock, Michael Q. Zhang, Vishwanath R. Iyer, Kirk Anders, Michael B. Eisen, Patrick O. Brown, David Botstein, Bruce Futcher, Comprehensive identification of cell cycle–regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization, Mol. Biol. Cell 9 (12) (1998) 3273–3297.
[74] Audrey P. Gasch, Paul T. Spellman, Camilla M. Kao, Orna Carmel-Harel, Michael B. Eisen, Gisela Storz, David Botstein, Patrick O. Brown, Genomic expression programs in the response of yeast cells to environmental changes, Mol. Biol. Cell 11 (12) (2000) 4241–4257.
[75] Shelley Chu, Joe DeRisi, Michael Eisen, Jon Mulholland, David Botstein, Patrick O. Brown, Ira Herskowitz, The transcriptional program of sporulation in budding yeast, Science 282 (5389) (1998) 699–705.
[76] Yee Hwa Yang, Sandrine Dudoit, Percy Luu, Terence P. Speed, Ormalization for cdna microarry data, in: Microarrays: Optical Technologies and Informatics, Vol. 4266, SPIE, 2001, pp. 141–152.
[77] Drew V. Klopfenstein, Liangsheng Zhang, Brent S. Pedersen, Fidel Ramírez, Alex Warwick Vesztrocy, Aurélien Naldi, Christopher J. Mungall, Jeffrey M. Yunes, Olga Botvinnik, Mark Weigel, et al., Goatools: A python library for gene ontology analyses, Sci. Rep. 8 (1) (2018) 10872.
[78] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, et al., Gene ontology: tool for the unification of biology, Nature Genet. 25 (1) (2000) 25–29.
[79] Ronald A. Fisher, On the interpretation of χ 2 from contingency tables, and the calculation of p, J. R. Stat. Soc. 85 (1) (1922) 87–94.
[80] Yoav Benjamini, Yosef Hochberg, Controlling the false discovery rate: a practical and powerful approach to multiple testing, J. R. Stat. Soc. Ser. B Stat. Methodol. 57 (1) (1995) 289–300.