Castanho *et al. BMC Bioinformatics (2023) 24:457*
https://doi.org/10.1186/s12859-023-05587-4

SOFTWARE **Open Access**

G-bic: generating synthetic benchmarks for biclustering

Eduardo N. Castanho[1*†], João P. Lobo[1†], Rui Henriques[2] and Sara C. Madeira[1]

†Eduardo N. Castanho and João P. Lobo have contributed equally to this work.

*Correspondence: ejcastanho@ciencias.ulisboa.pt

[1] LASIGE, Faculdade de Ciências, Universidade de Lisboa, Campo Grande 016, 1749-016 Lisbon, Portugal
[2] INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Av. Rovisco Pais 1, 1900-001 Lisbon, Portugal

Abstract

Background: Biclustering is increasingly used in biomedical data analysis, recommendation tasks, and text mining domains, with hundreds of biclustering algorithms proposed. When assessing the performance of these algorithms, more than real datasets are required as they do not offer a solid ground truth. Synthetic data surpass this limitation by producing reference solutions to be compared with the found patterns. However, generating synthetic datasets is challenging since the generated data must ensure reproducibility, pattern representativity, and real data resemblance.

Results: We propose G-Bic, a dataset generator conceived to produce synthetic benchmarks for the normative assessment of biclustering algorithms. Beyond expanding on aspects of pattern coherence, data quality, and positioning properties, it further handles specificities related to mixed-type datasets and time-series data.G-Bic has the flexibility to replicate real data regularities from diverse domains. We provide the default configurations to generate reproducible benchmarks to evaluate and compare diverse aspects of biclustering algorithms. Additionally, we discuss empirical strategies to simulate the properties of real data.

Conclusion: G-Bic is a parametrizable generator for biclustering analysis, offering a solid means to assess biclustering solutions according to internal and external metrics robustly.

Keywords: Biclustering, Synthetic data generator, Time series analysis, Unsupervised learning

Background

Unsupervised learning techniques such as clustering can find global patterns from multivariate observations based on the similarities found along all variables. However, biomedical patterns (including regulatory modules from multivariate omic data) are local, only correlated on a subset of attributes [1]. In this context, biclustering algorithms yield unique advantages. First, they can unravel local patterns, i.e., group observations meaningfully correlated on a subset of all attributes. Second, they place greater flexibility in the definition of different types of patterns. Third, biclusters can overlap, meaning that a subject, sample, or biological entity can belong to more than one bicluster [2, 3]. Table 1 lists prominent areas for biclustering applications.

Biclustering has been pervasively applied to diverse domains, such as clinical and biological [4–6], gene expression data [7, 8], recommendation tasks [9, 10], text mining [11], and time series analysis [12, 13].

While there are studies that use real data to compare the capacities of biclustering algorithms, this type of data has significant challenges due to the non-existence of a solid ground truth (a set of known and well-characterized internal data structure, regularities, or patterns) that would enable assessing the accuracy, sensitivity, and precision of the existing methods. Therefore, real data analysis only evaluates the internal coherency of the retrieved bicluster, disregarding an evaluation of false discovery rates [12, 26].

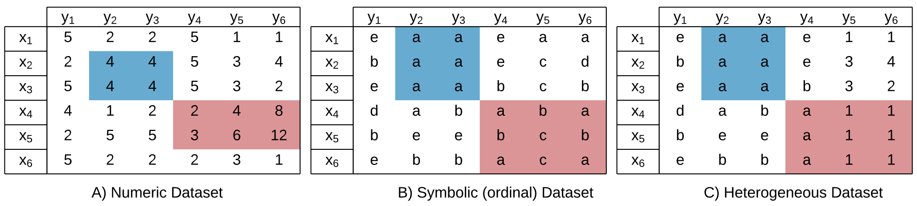
Considering the limitations of real data, biclustering studies have relied on synthetic data to test and improve their algorithms, where ground truth is available, and used external metrics to compare the capacity of biclustering algorithms to retrieve planted biclusters [1, 27–30]. Currently, the approach followed by previous studies suffers from three issues. *First*, each study generates its synthetic datasets, which is not only a time-consuming task but also is generally biased towards the approach under evaluation [31]. *Second*, most comparison studies do not make the produced data available; synthetic datasets are still scarce and poorly characterized, making replicating the data with the same characteristics unfeasible. *Third*, available synthetic data lack expressivity concerning temporal dynamics, flexible structures, and real data resemblance.

We propose *G-Bic*, a synthetic data generator conceived to produce heterogeneous datasets for biclustering. Several options for pattern coherence, biclustering structure, and overlapping are available. In particular, G-Bic is the first contribution to generating multivariate data with numeric, symbolic, and time-series data; therefore, it conforms to diverse application domains. G-Bic is implemented as a Java GUI for generating datasets with usability guarantees and a programmatic API. In addition to the software description of the tool, a set of empirical principles are given to properties for dataset generation in several domains.

The article is organized as follows: this remaining section defines the biclustering task, the evaluation strategies for biclustering algorithms, and the related work regarding simulating synthetic data. Section Implementation discusses the features of the generator. Section Results presents baseline datasets for algorithm assessment and discusses empirical methodologies to generate synthetic similar to real data, illustrating

Table 1 Application scenarios for biclustering

Domain	Data	Illustrative biclusters
Biomedical	Clinical [14, 15]	Groups of patients whose health records are correlated
	Gene expression [5, 16]	Groups of genes involved in functional processes and pathways
	Biological networks [17, 18]	Modules of genes and proteins with local interaction
	Physiological [4, 12]	Modules of sliding partitions of the signal across a subset of stimuli responses
Other	Recommendation systems [9, 19]	Groups of users who share the same rating patterns and behavioral patterns
	Text mining [11, 20]	Groups of documents with similar sets of words or topics
	Climate data [21, 22]	Regions with similar meteorological patterns
	Social sciences [23, 24]	Regions with similar social vulnerabilities
	Resources utilization [13, 25]	Regions with similar water or energy consumption patterns



	y_1	y_2	y_3	y_4	y_5	y_6
x_1	5	2	2	5	1	1
x_2	2	4	4	5	3	4
x_3	5	4	4	5	3	2
x_4	4	1	2	2	4	8
x_5	2	5	5	3	6	12
x_6	5	2	2	2	3	1

A) Numeric Dataset

	y_1	y_2	y_3	y_4	y_5	y_6
x_1	e	a	a	e	a	a
x_2	b	a	a	e	c	d
x_3	e	a	a	b	c	b
x_4	d	a	b	a	b	a
x_5	b	e	e	b	c	b
x_6	e	b	b	a	c	a

B) Symbolic (ordinal) Dataset

	y_1	y_2	y_3	y_4	y_5	y_6
x_1	e	a	a	e	1	1
x_2	b	a	a	e	3	4
x_3	e	a	a	b	3	2
x_4	d	a	b	a	1	1
x_5	b	e	e	a	1	1
x_6	e	b	b	a	1	1

C) Heterogeneous Dataset

Fig. 1 Example of a biclustering solution (made of two biclusters: blue and red) on three datasets: **A** numerical, **B** symbolic, and **C** heterogeneous

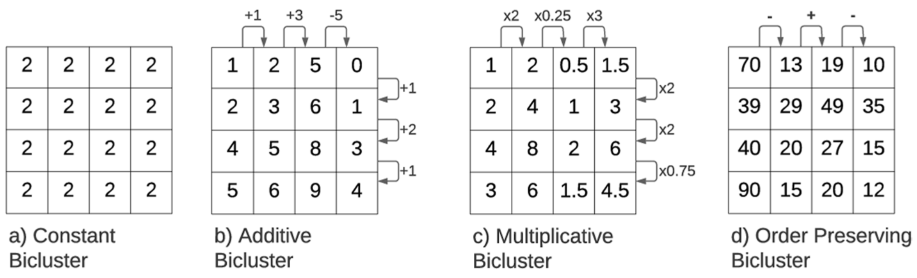


Fig. 2 The coherency of a bicluster shows how the values inside the bicluster correlate with each other

the capacities of G-Bic to generate synthetic datasets and emulate real datasets. Finally, Conclusion draws concluding remarks and implications from this work.

Biclustering

Given a two-dimensional dataset A, defined by n observations (rows) $X = \{x_1, ..., x_n\}$ and m attributes (columns) $Y = \{y_1, ..., y_m\}$, a **Bicluster** B corresponds to a subset of rows $I \subseteq X$ and columns $J \subseteq Y$ of the original matrix. We denote by b_{ij} the value corresponding to row i and column j in the bicluster B [2]. A **Biclustering Solution** is a set of q biclusters discovered by an algorithm, $\{B^1, B^2, ..., B^q\}$, that satisfy specific homogeneity and statistical significance criteria [2, 32]. Figure 1 shows biclustering solutions on numerical, symbolic, and heterogeneous datasets.

The **coherence** identifies the correlations inside a bicluster [1]. We highlight four significant coherency types, illustrated in Fig. 2: *Constant*, assuming that the bicluster has constant values, *Additive*, assuming that the values inside the bicluster are explained by a sum of factors, *Multiplicative*, assumes that the values are explained by a product of factors, and *Order Preserving* if there is a permutation of the attributes such that the values show a linear ordering across the dimension.

Usually, a biclustering algorithm does not consider the relative position between rows or columns. In time series data, columns correspond to time points, so the relative positioning between columns is relevant. Biclustering algorithms are classified as having temporal contiguity if they discover bicluster with contiguous columns, and the biclusters discovered by those algorithms are known as **C-Biclusters** [12, 13, 33, 34]. Figure 3 visually depicts the difference between a traditional bicluster and a C-Bicluster.

The elements of a data matrix can be part of more than one bicluster. The contributions per biclusters to the data matrix are explained by plaid models [35]. Lazzeroni

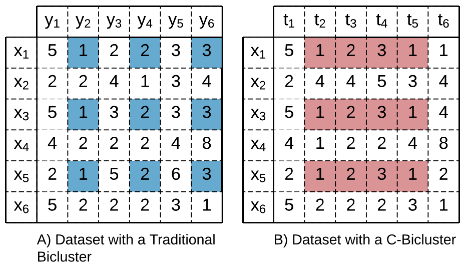

A) Dataset with a Traditional Bicluster
B) Dataset with a C-Bicluster

Fig. 3 Diference between a **A** Traditional Bicluster and a **B** C-Bicluster. A C-Bicluster represents temporal patterns. Hence, placing column contiguity can be desirable

and Owen [36] proposed an *additive model* (Fig. 4) where the elements a_{ij} of the data matrix are viewed as a sum of terms,

$$a_{ij} = \sum_{t=0}^{q} \theta_{ijt}\rho_{it}\kappa_{jt}, \tag{1}$$

where θ_{ijt} defines a contribution for each bicluster t, and ρ_{it} and κ_{jt} are boolean values that state if the observation i and attribute j is present on the bicluster.

Madeira and Oliveira [2] proposed the *multiplicative model* (Fig. 5) that assumes that a product of bicluster layers generates the matrix values,

$$a_{ij} = \prod_{t=0}^{q} \theta_{ijt}\rho_{it}\kappa_{jt}. \tag{2}$$

Henriques and Madeira [37] introduced the *interpoled model* (Fig. 6) to better capture behaviors in alternative domains. In this model, each value is obtained by averaging the bicluster values.

$$a_{ij} = \frac{\sum_{t=0}^{q} \theta_{ijt}\rho_{it}\kappa_{jt}}{q}. \tag{3}$$

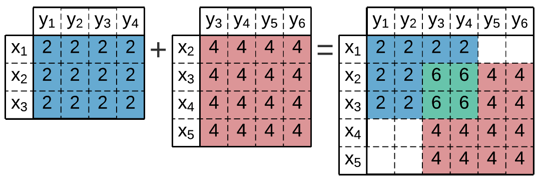


Fig. 4 Plaid model with an **additive** cumulative function between two constant biclusters

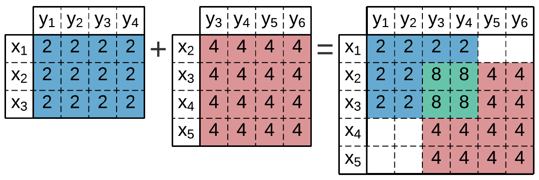


Fig. 5 Plaid model with an **multiplicative** cumulative function between two constant biclusters

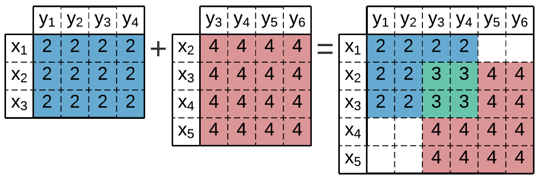


Fig. 6 Plaid model with an **interpoled** cumulative function between two constant biclusters

Evaluating biclustering algorithms

Evaluating biclustering correctness, homogeneity and statistical significance is challenging, as highlighted by a proliferation of different metrics [26, 38].

External metrics, also known as extrinsic or similarity evaluation metrics, are used when ground truth is known and quantify how well the biclustering algorithm recovers a planted/hidden biclustering solution [38]. The solutions obtained by biclustering algorithms are compared to the reference using a similarity measure based on variations of the F-Measure [39] or Jaccard scores [40]. These metrics are an essential tool to compare the correctness of the algorithms retrieving a biclustering and are used in studies developing new algorithms [5, 40] and in comparative studies [1, 27–29]. For a survey on external measures in biclustering, we refer to Horta and Campello [38].

Internal metrics, also known as quality metrics or coherence metrics, are used to evaluate the internal homogeneity of the bicluster [26]. Since biclustering has several types of patterns, defining homogeneity will depend on the assumed bicluster pattern. Different internal metrics are suited for different internal patterns. Examples are the *Variance*, formulated to detect constant biclusters, or the *Mean Squared Residue* focused on additive patterns [41, 42]. A few studies have recently used standardization techniques in the bicluster to develop flexible internal metrics to detect diverse patterns [26]. As indicated by Divina et al. [43], an advantage of this approach is to characterize their tendency. One way of doing it is to standardize data by row,

$$\hat{b}_{ij} = \frac{b_{ij} - \mu_i}{\sigma_i}, 1 \leq i \leq |I|, 1 \leq j \leq |J|, \tag{4}$$

where μ_i is the mean value of row x_i in the bicluster and σ_i is the standard deviation of all the column values of row x_i. Let a virtual pattern be defined as:

$$\hat{\rho}_j = \frac{1}{\mid I \mid} \sum_{i=1}^{|I|} \hat{b}_{ij}. \tag{5}$$

This pattern represents an average behavior of all observations in each attribute, and the virtual error (VE) [26, 43, 44] is thus defined as a difference between the row values and the virtual row,

$$VE(B) = \frac{1}{\mid I \mid \times \mid J \mid} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} \mid \hat{b}_{ij} - \hat{\rho}_j \mid, \tag{6}$$

where b_{ij} refers to the element (i, j) of the data subspace B, and $|I|$, $|J|$ represent the total number of rows and columns. This metric is sensitive to constant, additive, and multiplicative patterns. Although order preserving patterns may fail to show a zero virtual error, significantly lower virtual errors for these subspaces are expected compared to non-homogeneous subspaces.

A more reduced set of intrinsic metrics has been proposed to evaluate the homogeneity of symbolic biclustering solutions. The symbolic biclusters are generally evaluated by assuming a symbolic pattern and measuring the noise as the percentage of elements in the bicluster that do not respect the pattern. Other strategies convert symbolic onto numeric patterns (using a traditional metric for numeric data) or consider metrics developed for clustering [45–47].

Statistical significance metrics evaluate how relevant the bicluster is in the data matrix compared to randomized data or null assumptions. These metrics are relevant since highly homogeneous biclusters can appear by chance, and these metrics reduce the occurrence of false positives. These metrics are used in previous application studies, often combined with internal metrics, to assess biclustering solutions [13].We refer to Henriques and Madeira [35] for a survey on statistical significance metrics.

Related work

In contrast to real data (where no ground truth is given), synthetic datasets offer the advantage of evaluating biclustering algorithms based on the correctness of the retrieved biclustering solutions [38, 48]. Synthetic datasets are widely used in the biclustering literature to support algorithmic development [16, 40, 49, 50], the design of merit functions and evaluation metrics [26, 38], and algorithm comparison [27, 29].

Hochreiter et al. [40] compared their proposed algorithm in datasets generated by previous authors [16, 27] and custom-generated datasets. A biclustering generator was made available in the "fabia" R package; however, it uses the same additive generation model that the FABIA algorithm assumes to be true. The "isa2" R package also makes a synthetic data generator available, but it lacks diverse coherency options and only generates overlapping biclusters of equal size. Horta and Campello [38] generated seven synthetic data collections to evaluate external measures. These datasets were made with the intent of reproducing previous studies [2, 27, 29, 40]. Pontes et al. [26] evaluated experimentally internal metrics by generating synthetic datasets with different coherences and noise values.

Synthetic datasets are also used in most comparative studies, i.e., studies developed to systematically compare the performance of biclustering algorithms. Prelić et al. [27] generated biclustering datasets with constant and additive coherencies. Their objective was to compare biclustering algorithms given varying degrees of noise and overlap. This approach was later followed by Bozdağ et al. [28], and Padilha and Campello [30] by expanding the study variables, including the data size, the number, and shape of biclusters, and their possible coherencies.

Generating synthetic data is a time-consuming task due to the development and running of the generating scripts. Assumptions about the patterns in the data must be made, risking a bias toward the algorithm under evaluation. The generated data (and accompanying scripts) are often unavailable, raising reproducibility issues. Challenges with generating synthetic data lead to the development of independent data generators. Data generators developed in areas such as pattern mining [51] and clustering [52] are not helpful because they cannot simulate the pattern diversity in biclustering and the overlapping between biclusters. Therefore, there is a need for data generators specific to biclustering. It is also worth noticing that synthetic data for biclustering demands well-defined ground truth and parametrizable patterns. Although the use of real data simulators [53–55], or generative AI approaches [56, 57] can provide unique opportunities for simulating the properties of real data, they lack the flexibility to be applied in several domains, create a well-defined ground truth, and provide parameterization facilities within usable interfaces.

BiBench is a contribution capable of generating synthetic datasets [29]. This generator creates datasets with coherency patterns following definitions by Aguilar-Ruiz [44] and overlapping between biclusters given by the additive plaid model [36]. However, BiBench was limited in bicluster patterns: the generated datasets were numeric, all biclusters in the dataset followed the same type of pattern, and the patterns showed constant values across columns. This generator also lacked overlapping and noise options. Therefore, BiBench cannot generate data where the observations show different correlations between the attributes.

BiGen, later proposed by Henriques [58], fixed these limitations and broadened the set of features by allowing the generation of biclusters of both symbolic and numeric natures, with dynamic structures, considering all patterns, and with parameterizable overlapping and quality options. However, this data generator lacks background definitions, the generation of heterogeneous data, and temporal biclusters.

Developed for triclustering (an unsupervised task with similar data challenges as biclustering), *G-Tric* introduced new concepts such as generating (possibly distinct) patterns over more than one dimension, admitting that the background and the bicluster could follow different noise structures, and contiguity options that are useful to simulate time series data. Since G-Tric was developed for triclustering, the generated datasets are not directly applicable to biclustering. Additionally, compared to G-Bic, G-Tric lacks options to generate heterogeneous data.

Implementation

We propose *G-Bic*, a fully parametrized generator of heterogeneous and temporal data focused on the necessities of biclustering. G-Bic is the first generator with options for (1) generating heterogeneous data (i.e., mixed variable types) with planted patterns; (2) handling temporal data, including contiguity assumptions on the time dimension; and (3) higher degrees of flexibility for parameterizing the coherence (type and strength), structure (number and size), and quality of the biclusters, surpassing the limitations of earlier generators. In particular, G-Bic:

1. Allows to define the pattern in both rows and columns;
2. Uses personalized quality options for the dataset background and the planted biclusters;
3. Extends the overlapping parametrization to allow defining the fraction of patterns that can overlap and their overlapping extent;
4. Generates either homogeneous or heterogeneous datasets, with non-identically distributed variables, combining symbolic and numeric variables;

5. Generates temporally contiguous biclusters.

Furthermore, *G-Bic* is implemented as a GUI (Graphical User Interface), developed using JavaFX for easy usability and a programmatic API for additional control over the data simulation facilities. Figure 7 shows the GUI implementation. As supplementary material, a tutorial is made available for the guided use of G-Bic. In the remainder of this section, we describe the proposed generator. Each subsection describes a tab of the G-Bic GUI.

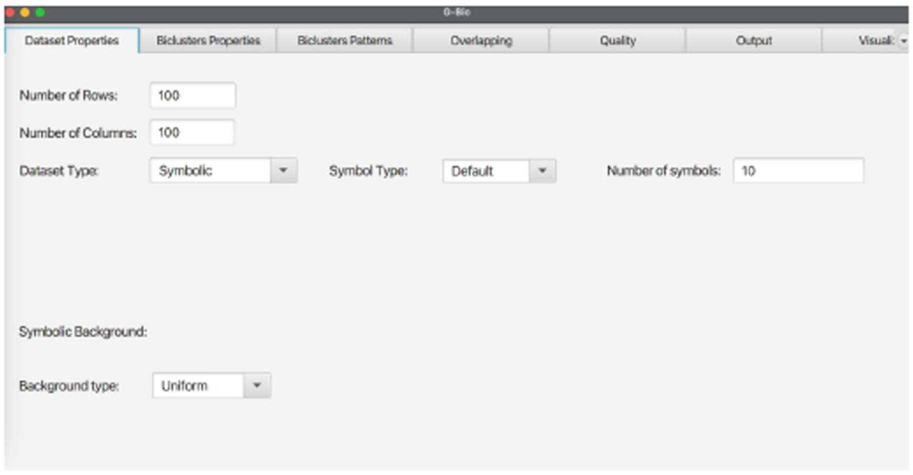
Dataset properties

The initial step in dataset generation entails delineating its structure without a planted bicluster, an option available within the *Dataset Properties* tab. The structure is determined by specifying the dataset's dimensions in terms of rows and columns, elucidating the attribute types present in the dataset (numeric, symbolic, or heterogeneous), and elucidating the statistical underpinnings of the dataset.

When considering heterogeneous data, the user controls the percentage of numeric attributes compared to symbolic ones. If the user selects numeric features, he controls the attributes, integers, or real-valued, and the maximum and minimum values. Symbolic attributes are best used when the features are represented by symbols (such as characters) or a general trend. The user can then select the number of symbols by defining a custom list of symbols or a default alphabet.

The background controls the possible distributions for the values in the dataset. G-Bic makes available four types of backgrounds:

- *Uniform*, assuming that each alphabet element has equal probability to occur;
- *Normal*, considering a gaussian distribution to generate the dataset;
- *Discrete*, which gives the control of the probabilities of each symbol to the user;
- *Missing*, that assumes a null background.



Number of Rows: 100
Number of Columns: 100
Dataset Type: Symbolic
Symbol Type: Default
Number of symbols: 10
Symbolic Background:
Background type: Uniform

Fig. 7 Overview of the GUI implementation, guaranteeing usability for data simulation. Accompanying the executable, the source code and a tutorial are available at https://github.com/jplobo1313/G-Bic

Bicluster properties

The *Bicluster Properties* controls the general properties of the generated biclusters, essentially, the number of planted biclusters and their size. The number of rows and columns of the biclusters are variable and are defined by either a *Gaussian* or *Uniform* distribution. If the objective of G-Bic is to simulate biclusters for time series analysis, the *Contiguity* option can be controlled to generate C-Biclusters.

Bicluster patterns

The third stage is the definition of patterns likely to be observed in terms of type and shape that will be planted on the dataset.

There are four types of coherency patterns for biclustering: *Constant*, *Additive*, *Multiplicative*, *Order Preserving*. In addition, G-Bic considers a fifth type, *None*, that represents the non-existence of coherency across one dimension. This last type is proper when the desired biclusters are only coherent across one dimension, i.e., the rows or the columns.

Compared to previous biclustering generators BiBench and BiGen, G-Bic allows defining the patterns in both dimensions, meaning that a bicluster can have a constant coherency over columns and additive over rows. The bicluster element b_{ij} is a mathematical relationship between a seed value μ, a row factor α_i, and a column factor β_j (except for the order preserving).

Due to incompatibilities related to the mathematical modeling of the patterns, not all patterns are compatible. Table 2 shows each coherency's possible pattern configurations and mathematical expressions.

In the context of time series analysis and the generation of C-biclusters, if the coherency is chosen as order preserving, G-Bic allows the generation of temporal profiles that are either: *Monotonically Increasing*, *Monotically Decreasing*, or *Random*. These subcoherencies, illustrated in Fig. 8, are exclusive for C-Biclusters, since the relative positioning of the columns is relevant.

Overlapping

After generating the isolated biclusters, the next step is to define how they overlap.

Table 2 G-Bic allows users to select different patterns over rows and columns

Coherency	Row pattern	Column pattern	Mathematical formula	Dataset type
Order preserving	Order Preserving	None	n/a	Numeric and symbolic
	None	Order preserving		
Constant	Constant	Constant	$b_{ij} = \mu$	Numeric and symbolic
	None	Constant	$b_{ij} = \alpha_i$	
	Constant	None	$b_{ij} = \beta_j$	
Additive	Additive	Additive	$b_{ij} = \mu + \alpha_i + \beta_j$	Numeric
	Constant	Additive	$b_{ij} = \mu + \beta_j$	
	Additive	Constant	$b_{ij} = \mu + \alpha_i$	
Multiplicative	Multiplicative	Multiplicative	$b_{ij} = \mu \times \alpha_i \times \beta_j$	Numeric
	Constant	Multiplicative	$b_{ij} = \mu \times \beta_j$	
	Multiplicative	Constant	$b_{ij} = \mu \times \alpha_i$	

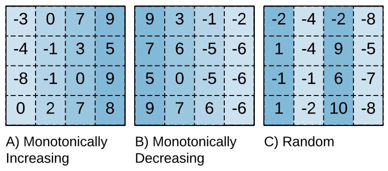


Fig. 8 Considering the specific context of biclustering for time series and the Order Preserving coherency, G-Bic has three time profiles: **A** Monotonically Increasing, **B** Monotonically Decreasing or **C** Random

G-Bic admits four possible *plaid coherencies*, i.e., how the values of overlapping biclusters add to the final value in the matrix: *additive* as defined in Eq. 1, *Multiplicative* as defined in Eq. 2, *Interpoled* as defined in Eq. 3 and, *None*, where the value in the datasets corresponds to the last generated bicluster.

The percentage of overlapping biclusters and the maximum number of interactions can be controlled. If the percentage is defined as 40% and the maximum number of interactions is defined as 3, then 60% of the biclusters will not overlap, and the ones that do overlap no more than 3 biclusters will overlap in a single matrix element. It is also possible to define the percentage of overlapping rows, columns, and elements between biclusters.

Quality

The quality of the data and generated subspaces is parameterizable by assigning *missing*, *noise*, or *error* elements. The distinction between *noise* and *error* is done by a personalized *noise threshold* parameter, defining the tolerated deviation for the numerical values in the dataset and the biclusters. In G-Bic, noise is defined by fluctuations within the threshold, and errors are fluctuations beyond that.

Output and visualization

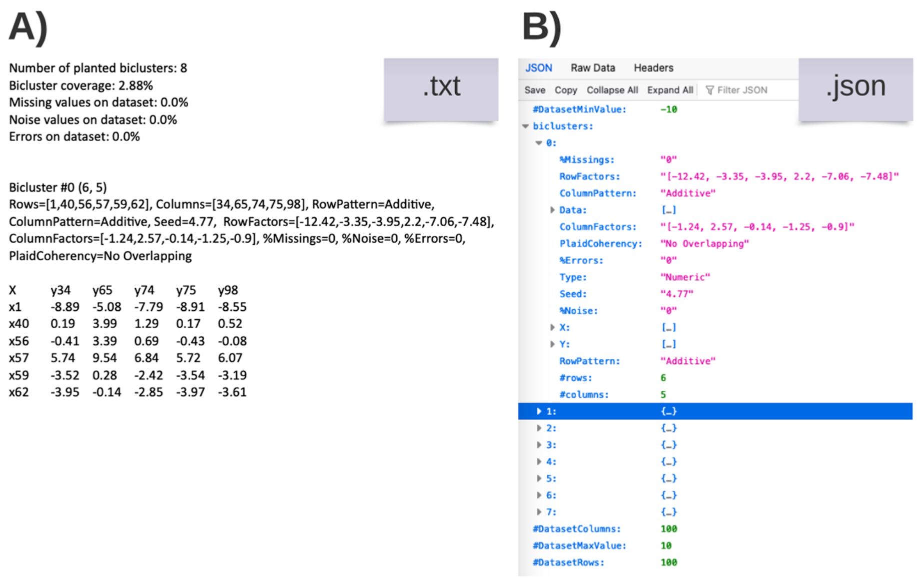
After generating the biclustering solutions, G-Bic produces three files:

- **.json file**: Containing the settings that generated the dataset;
- **.txt file**: Listing the generated biclusters;
- **.tsv file**: Containing the dataset.

The structure of the *.json* and *.txt* files are shown in Fig. 9. The three files make the results from G-Bic easily readable either by humans or automatically by any programming language with a *json* and *tsv* reader library. In addition to the files, G-Bic makes available a graphical tool to visualize the produced subspaces using heatmaps.

Results

In this section, we provide default configurations to generate reproducible benchmarks to evaluate and compare the properties of biclustering algorithms. **First**, the simulation of simple synthetic datasets to evaluate the performance of biclustering algorithms by controlling essential parameters of G-Bic. We begin by producing three illustrative synthetic datasets. Then, we discuss baseline datasets and provide a guiding criterion for using G-Bic to establish experimental comparisons between biclustering algorithms. The



```
Number of planted biclusters: 8
Bicluster coverage: 2.88%
Missing values on dataset: 0.0%
Noise values on dataset: 0.0%
Errors on dataset: 0.0%


Bicluster #0 (6, 5)
Rows=[1,40,56,57,59,62], Columns=[34,65,74,75,98], RowPattern=Additive,
ColumnPattern=Additive, Seed=4.77,  RowFactors=[-12.42,-3.35,-3.95,2.2,-7.06,-7.48],
ColumnFactors=[-1.24,2.57,-0.14,-1.25,-0.9], %Missings=0, %Noise=0, %Errors=0,
PlaidCoherency=No Overlapping

X      y34    y65    y74    y75    y98
x1     -8.89  -5.08  -7.79  -8.91  -8.55
x40    0.19   3.99   1.29   0.17   0.52
x56    -0.41  3.39   0.69   -0.43  -0.08
x57    5.74   9.54   6.84   5.72   6.07
x59    -3.52  0.28   -2.42  -3.54  -3.19
x62    -3.95  -0.14  -2.85  -3.97  -3.61
```

```
#DatasetMinValue:    -10
▾ biclusters:
  ▾ 0:
      %Missings:       "0"
      RowFactors:      "[-12.42, -3.35, -3.95, 2.2, -7.06, -7.48]"
      ColumnPattern:   "Additive"
    ▸ Data:            […]
      ColumnFactors:   "[-1.24, 2.57, -0.14, -1.25, -0.9]"
      PlaidCoherency:  "No Overlapping"
      %Errors:         "0"
      Type:            "Numeric"
      Seed:            "4.77"
      %Noise:          "0"
    ▸ X:               […]
    ▸ Y:               […]
      RowPattern:      "Additive"
      #rows:           6
      #columns:        5
  ▸ 1:                 {…}
  ▸ 2:                 {…}
  ▸ 3:                 {…}
  ▸ 4:                 {…}
  ▸ 5:                 {…}
  ▸ 6:                 {…}
  ▸ 7:                 {…}
#DatasetColumns:     100
#DatasetMaxValue:    10
#DatasetRows:        100
```

Fig. 9 G-Bic generates files with a focus on accessibility. In **A**, the *.txt* generated by G-Bic has the information for each bicluster in a human-friendly view. In **B**, *.json* file has the information associated with the dataset and the biclusters structured to import the data into a programming script easily

second part of this section concerns the simulations of data resembling real data. We begin by discussing conclusions by authors in multiple biclustering domains and provide the settings to generate reference datasets for each application domain. Finally, we show an empirical approach in a scenario where no studies are available and how to use biclustering algorithms to infer properties for synthetic dataset generation. This empirical approach is illustrated with a gene expression microarray dataset.

Illustrative datasets

In this section, we illustrate the capacities of G-Bic to simulate synthetic datasets for comparative studies. Assessing biclustering algorithms in simple solutions is an essential step to characterize the behavior of the algorithms and identify their limitations [1, 27–30]. The settings for these datasets are in Table 3. We consider three scenarios: in the *first* scenario, a synthetic dataset with a single bicluster, illustrated in Fig. 10. *Second*, we illustrate the plaid models by illustrating a dataset with an overlapping between two biclusters, illustrated in Fig. 11. *Finally*, the last illustrative dataset shows the temporal patterns created by G-Bic, illustrated in Fig. 12.

Baseline datasets

Six data domains were considered, each varying with regards to variable types (**R**-Real Valued, **S**-Symbolic, **B**-Binary, **I**-Integer, **C**-contiguous, and **H**-Heterogeneous), represented in Table 4. While these datasets do not help compare biclustering algorithms, they serve as a reference baseline to generate comparable, reproducible dataset benchmarks.

The most studied parameter in biclustering algorithms is their robustness to the presence of noise in numeric datasets [1, 27, 29, 30]. G-Bic has three different parameters to control the quality of the datasets: the number of missing values, the noise (small

Table 3 Setting for the illustrative datasets

Properties		Illustrative 1	Illustrative 2	Illustrative 3
Dataset	Data type	Real-valued		
	Size	100 × 100		
	Alphabet	[-100,100]		
	Background	N(0,30)		
Biclusters	Number	1	2	10
	Number of rows	50	50	U(3,12)
	Number of cols	50	50	U(10,20)
	Row Pattern	Constant	Constant	None
	Col Pattern	Constant	Constant	Order Preserving
	Contiguity	No	No	Yes
	Time Profile	None	None	Monotonically Decreasing
Overlapping	Plaid Coherency	No Overlapping	Additive	No Overlapping
	% of Overlapping biclusters	N.A	100 %	N.A
	Maximum number of bicluster interations	N.A	2	N.A
	% of overlapping elements	N.A	50 %	N.A
	% of overlapping rows	N.A	50 %	N.A
	% of overlapping columns	N.A	50 %	N.A
	Quality	No noise, errors or missing values		

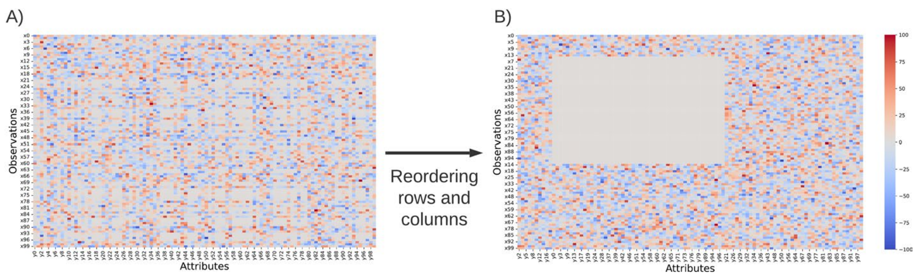


Fig. 10 Synthetic dataset with a single constant bicluster. The produced dataset has the appearance of (**A**). For visualizing the biclustering solution, a reordering of rows and columns is needed to visualize the bicluster in (**B**)

fluctuations of the dataset) and the number of errors in the dataset (quantity of significant fluctuations of the dataset).

Another dataset characteristic often studied is the overlap between bicluster solutions. This is challenging due to the inherent restrictions associated with some biclustering searches and the complexity of plaid models [59]. Previous studies compare the performance of biclustering algorithms in overlapping biclusters by varying the row and column overlap degree [27, 29, 30]. G-Bic allows not only to personalize the percentage of overlapping elements over rows and columns but also to define how many biclusters will overlap and the type of plaid coherency (additive, multiplicative, and interpoled).

Simulating datasets based on a literature review

In this section, we begin discussing the simulation of data similar to real data. To ensure the coverage of different data domains, we focus on five popular areas of application

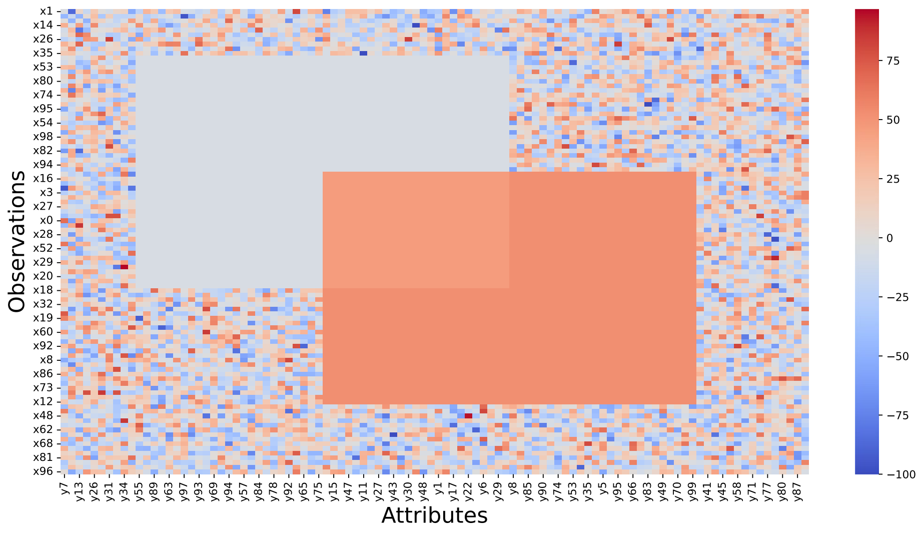


Fig. 11 G-Bic generates biclusters with plaid assumptions, representing biological scenarios where genes participate in several biological processes

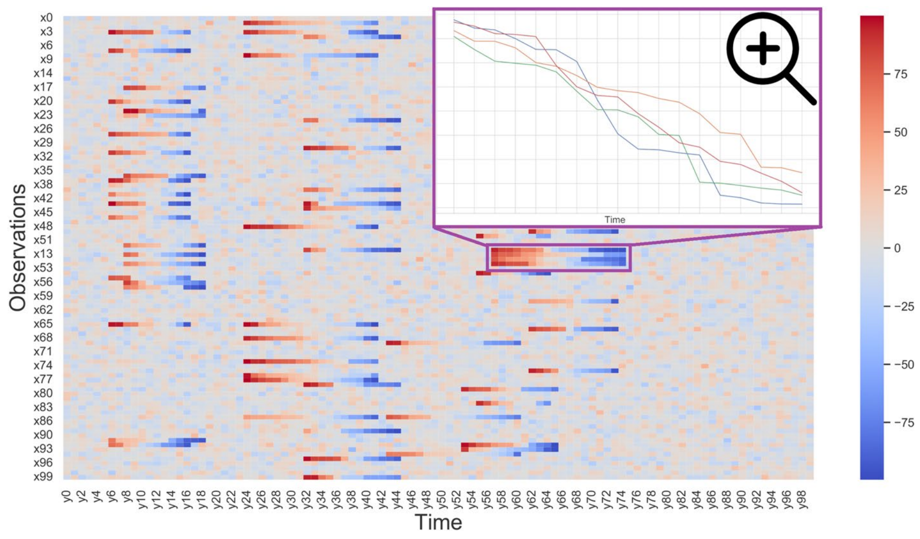


Fig. 12 G-Bic generates biclusters representing temporal contiguity patterns. Parameters were selected to simplify the visualization. A close-up of one of the order-preserving biclusters shows four rows with a similar pattern over a few contiguous time points

of biclustering: gene expression data, text mining, recommendation systems, biomedical, and Spatio-temporal data. We choose one reference dataset for each applicational domain based on a previous biclustering study, summarized in Table 5 , and explained below.

In the context of **gene expression** data, it is worth noticing that transcriptomic data is a prevalent application domain for biclustering, so studies often consider RNAseq or microarray data when comparing the performance of their algorithms by using both internal and biological relevance metrics [38]. Comparative studies [1, 27–29] and applications reviews [7, 32] highlight this observation. From these

Table 4 Setting for baseline datasets

	Properties	Dataset R	Dataset S	DatasetB	Dataset I	Dataset C	Dataset H
dataset	Data type	Real-valued	Symbolic	Integer	Integer	Real-valued	Heterogeneous (50% real, 50% symbolic)
	Size	1000 × 100	1000 × 100	1000 × 100	1000 × 100	1000 × 100	1000 × 100
	Alphabet	[-100,100]	{a,b,c,d,e}	{1,0}	[0,100]	[-100,100]	[-100,100] and {a,b,c,d,e}
	Background	N(0,30)	Missing	Missing	Uniform	Uniform	Missing
bics	Number	1	1	1	1	1	1
	Dimensions	50 × 50	50 × 50	50 × 50	50 × 50	50 × 50	50 × 50
	Contiguity	No	No	No	No	Yes	No

Table 5 Example datasets considered for each simulation scenario

	Dataset Context	Description	Dimensions	Size
1	Gene Expression	Arabidopsis [60]	Genes × Conditions	21031 × 351
2	Recommendation Systems	MovieLens-20M [19, 61]	Users × Movies	138000 × 27000
3	Text Mining	Reuters-21578 [62]	Terms × Documents	29930 × 21578
4	Clinical Data	PMSI2013 [6]	Patients × Clinical Data	49231 × 7941
5	Spatio-Temporal data	fMRI time series [12]	Brain Regions × Time	30 × 150

studies, we choose the arabidosis microarray data that captures the expression levels of a few thousand genes on a few hundred experimental conditions [60] , hence representative of the dimensionality and data size of transcriptomic data.

For **recommendation systems**, we considered the MovieLens-20 M dataset, a popular benchmark dataset in several biclustering studies. Recommendations system datasets use integers to represent the preferences of thousands of users over thousands of products, frequently associated with a high sparsity [19, 61, 63–65].

In **text mining**, Reuters-21578 is used in most biclustering studies and consists of thousands of terms over thousands of documents, with a high sparsity [62, 66].

The fourth dataset considers **Clinical data**. While clinical data is often used in biclustering studies, their general characteristics vary from application to application. Our primary reference is the study by Vandromme et al. [6]. In this context, the reference clinical dataset has a heterogeneous nature, high sparsity, and hundreds of observations and attributes.

The final simulated dataset comes from **Spatio-temporal data**. Spatio-temporal data has two challenges: First, since the application domains, so do the data characteristics. Second, most studies do not share the data or its specific characteristics. We considered the work by Castanho et al. [12], who evaluated the capacities of biclustering on fMRI data.

Considering the settings based on the definitions from Madeira and Oliveira [2] and the conclusions by studies that apply biclustering algorithms in each domain, the parameters for the simulated datasets are present in Table 6.

Table 6 Settings to simulate real datasets

Parameters		**Gene expression**	**Recommendation systems**	**Text mining**	**Clinical data**	**Spatio-temporal data**
Dataset properties	Number of rows	10000	200000	30000	50000	30
	Number of columns	100	30000	20000	8000	150
	Heterogeneous?	No	No	No	Yes	No
	Properties	Background with 10% noise	Background with 95,5% Missing values	Background with 99,8% missing values	Background with 99,8% missing values	Background with 50% noise and 20% errors
Bicluster properties	Number of biclusters	500	3000	70	30	20
	Rows structure	U(80,400)	U(30,70)	U(1000,10000)	U(20,100)	U(2,4)
	Columns structure	U(20,40)	U(3,7)	U(600,6000)	U(5,15)	U(7,10)
	Contiguity	No	No	No	No	Yes
Biclustering patterns		Additive and Order Preserving	Order Preserving	Constant and Order Preserving	Order Preserving	Additive and Multiplicative
Overlapping		10% bics with additive overlap	None	None	None	None

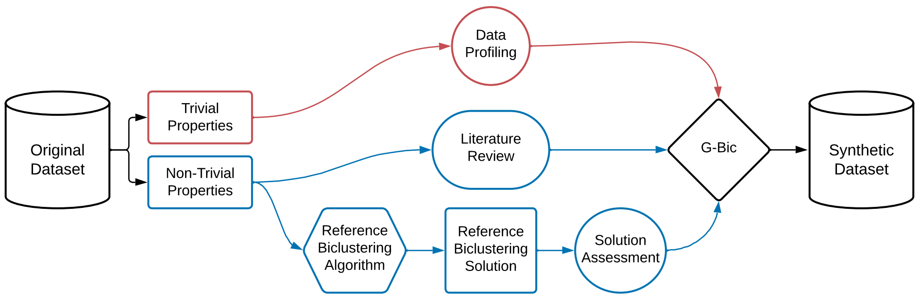


Fig. 13 Methodology to simulate synthetic data resembling real data. A reference biclustering algorithm produces results under different homogeneity criteria, from which synthetic datasets are produced

A methodology to simulate real data

In the previous section, we discussed the parameterizations to generate datasets based on previous research. This section proposed an empirical methodology to decide on the G-Bic parameters for dataset simulation. **First**, we explain the general methodology, which is illustrated in Fig. 13. **Second**, we illustrate this methodology by applying it to a gene expression dataset.

In this methodology, we consider a reference dataset. Any dataset has two types of properties, illustrated in Fig. 14: Trivial and Non-trivial properties. The trivial properties can be directly observed in the dataset. Examples of trivial properties are the dataset size, the background distribution, or the percentage of missing values. However, there are a few non-trivial properties, such as the coherence assumption in the

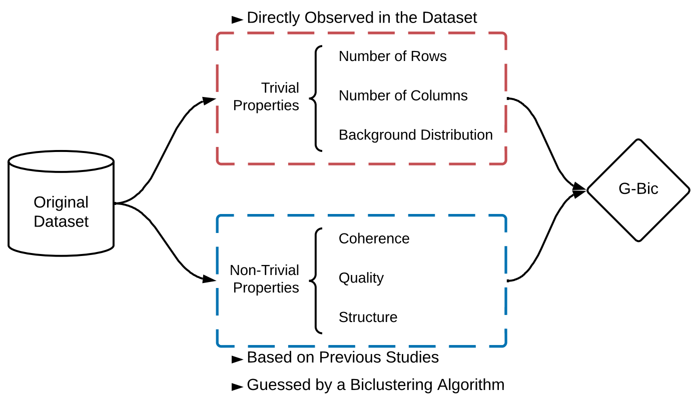


Fig. 14 The datasets have trivial properties that can be directly observed in the dataset and non-trivial properties. To determine these non-trivial properties, we suggest an extrapolation based on previous studies on the application domains or using a biclustering algorithm as a reference

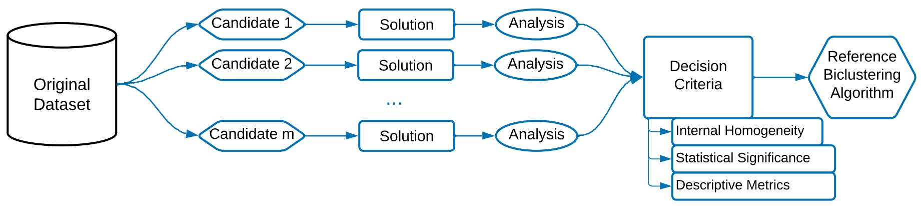


Fig. 15 To guarantee that the biclusters represent patterns in the original data, several candidates could be used with criteria for decision-making

data and the number and structure of the biclusters. For these, we propose that the parameters should be either decided from: (**A**) the properties of the biclustering solutions on previous studies on the application domain or (**B**) a reference biclustering algorithm as a reference to identify the properties of the patterns in the original data.

This methodology has the challenge of being dependent on the choice and parameterization of a reference biclustering algorithm. While several biclustering algorithms could be used to infer the dataset's properties, we suggest the use of BicPAMS since it is a flexible, highly parametrizable algorithm with well-defined solutions in terms of coherency patterns, integrating state-of-the-art contributions on pattern-based biclustering [5, 37].

A potential issue with using a biclustering algorithm as a reference to extract relevant features for the synthetic data generation is the risk of generating biased results towards the specific algorithm. To guarantee that the reference biclustering solution is representative of the patterns in the original dataset, we propose that several reference biclustering candidates should be used, either various biclustering algorithms or different parametrizations of the same algorithm, as illustrated in Fig. 15. These candidates should be tested and evaluated with decision criteria, which could consist of either (**A**) internal metrics, selecting the candidate that produced the most homogeneous solutions given an adequate metric, (**B**) statistical significance metrics, selecting the candidate that produced the most statistically relevant biclustering solution or (**C**) descriptive metrics such as the number of biclusters or their size as decisive criteria.

Illustrative results

To illustrate the approach, we considered the simulation of synthetic data resembling the *dlblc dataset*, which can be accessed in https://web.ist.utl.pt/rmch/bicpams/. The dlblc dataset measures the expression of 660 genes across 180 experimental conditions [37, 67]. To decide the coherence of the biclustering solution, we considered four iterations of BicPAMS, considering four coherencies: Constant, Additive, Multiplicative, and Order Preserving. Table 7 describes the relevant parameters used in BicPAMS. We considered internal homogeneity a decision criterion evaluated by the virtual error metric. Figure 16 shows the virtual error for each configuration, and Table 8 shows the size of each biclustering algorithm. Based on the homogeneity decision, we generate data with constant biclusters. We considered results from

Table 7 Parametrization of BicPAMS

Parameters	Values
Coherency strength	6
Merging overlap	80%
Pattern orientation	Rows
Minimum columns	4

We applied BicPAMS to the reference dataset *dlblc* based on the previously discussed methodology. The table only has the relevant parameters; the remaining were set as default

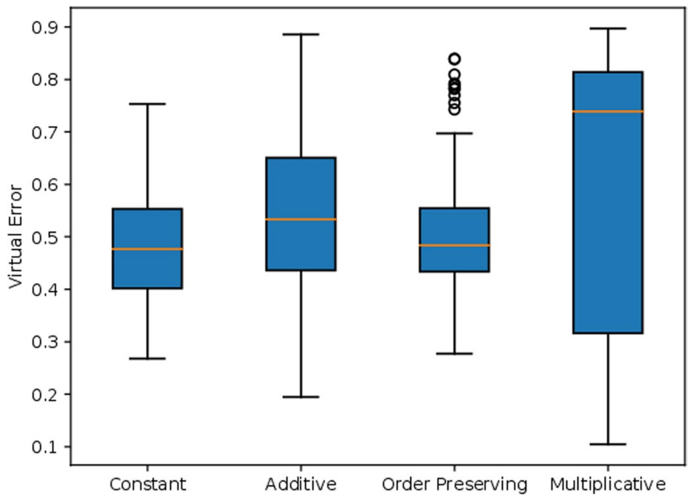


Fig. 16 Virtual Error of each reference candidate algorithm. Each candidate generates a biclustering solution with a different homogeneity. The "constant" candidate creates solutions with lower virtual error

Table 8 The size of each BicPAMS candidate solution

Configuration (coherency)	Number of biclusters	Number of rows (average)	Number of columns (average)
Constant	52	73	4
Additive	7702	51	4
Order preserving	129	88	4
Multiplicative	1134	30	4

the previous section (overlapping and quality) for the additional parameters. Table 9 shows the G-Bic configurations that generate the synthetic dataset.

Figure 17 compares the homogeneity distributions of the reference biclustering (which we assume to be representative of the original data) and the simulated biclustering (produced by G-Bic). We can observe that the distributions are similar.

Conclusion

Biclustering algorithms, initially developed for gene expression data analysis, are gaining increased importance in several domains, which raises challenges regarding intrinsic data diversity. Since every developed algorithm tries to improve the state-of-the-art, reproducible synthetic benchmarks for a fair algorithmic comparison are increasingly important.

The main contribution of this study is the proposal of G-Bic, a synthetic data generator for biclustering. G-Bic provides a programmatic and graphical facility for the functional and standardized assessment of existing and upcoming unsupervised learning contributions in the field. Biclustering algorithms expand upon traditional clustering due to their flexibility in detecting patterns. G-Bic allows the generation of datasets with diverse characteristics, namely varying coherency and overlapping options. It is also the

Table 9 Parametrization of G-Bic

Parameters		Values
Dataset properties	Number of rows	660
	Number of columns	180
	Dataset type	Numeric
	Data type of rows	Real valued
	Min value	−100
	Max value	100
	Background type	Normal (0,45)
Bicluster properties	Number of biclusters	52
	Row distribution	Normal (73,3)
	Number of columns	4
	Contiguity	None
Bicluster patterns	Row Pattern	Constant
	Column pattern	None
Overlapping settings	Plaid Coherency	Additive
	% of overlapping biclusters	10
	Maximum number of bicluster interactions	5
	% of overlapping rows	10
	% of overlapping rows	10
Quality	% Missing values on background	0
	% Missing values on biclusters	0
	% Noise values on background	0
	% Noise values on background	40
	Noise deviation	90
	% Errors values on background	0
	% Errors values on background	10

We used G-Bic to generate a synthetic dataset with similar properties based on the reference biclustering

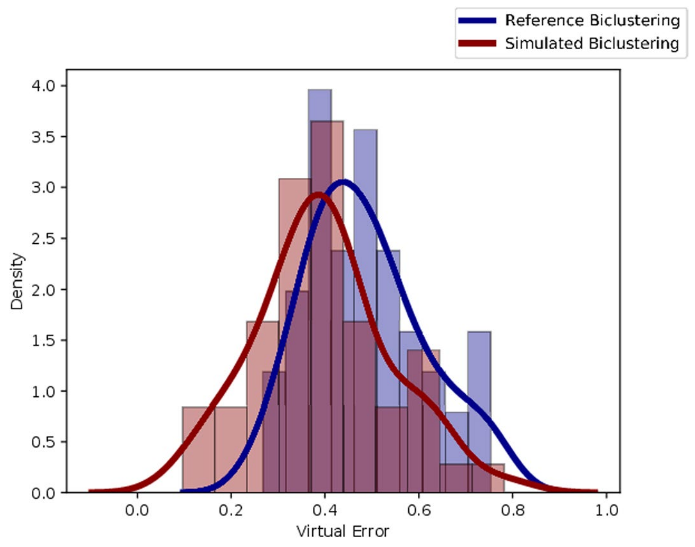


Fig. 17 Distribution of the virtual error. The two biclustering solutions have a similar distribution in terms of internal quality, an indicator that the dataset produced by G-Bic potentially resembles the original data

first contribution providing the ability to generate heterogeneous and time series data with planted pattern solutions, accompanying the expanding application scenarios for biclustering.

Complementarily, we provide benchmark datasets for a normative assessment of the characteristics of biclustering algorithms. Parameters to simulate real datasets are provided based on previous literature, and an empirical methodology to generate datasets similar to real data is given. These indications are expected to improve the current status on the evaluation of pattern mining and subspace clustering algorithms.

Acknowledgements
Not applicable

Author contributions
EC, SCM, and RH designed the algorithms for data generation, the methodology to generate datasets used in the paper, and the software. JL implemented the data generation algorithms, the software interface, and the code generating the datasets used in the paper. EC further drafted the manuscript. All authors revised and approved the final manuscript.

Funding
Funding was provided by Fundação para a Ciência e a Tecnologia (Grant Nos. 2021.07810.BD, DSAIPA/AI/0033/2019 and PTDC/CCI-CIF/4613/2020), H2020-RIA (Grant No. 818290), INESC-ID Research Unit (Grant No. UIDB/50021/2020), LASIGE Research Unit (Grant No. UIDB/00408/2020 and UIDP/00408/2020).

Availability of data and materials
The datasets generated during and/or analyzed during the current study are available in the G-Bic repository, https://github.com/jplobo1313/G-Bic. Availability and requirements: Project name: G-Bic. Project home page: https://github.com/jplobo1313/G-Bic. Operating system(s): Platform Independent. Programming language: Java. Other requirements: Java 16 or above. License: GNU GPLv3. ny restrictions to use by non-academics: No.

Declarations

Ethics approval and consent to participate
Not applicable

Consent for publication
Not applicable

Competing interests
Not applicable

Received: 29 May 2023 Accepted: 28 November 2023
Published: 6 December 2023

References

1. Henriques R, Antunes C, Madeira SC. A structured view on pattern mining-based biclustering. Pattern Recogn. 2015;48:3941–58. https://doi.org/10.1016/j.patcog.2015.06.018.
2. Madeira SC, Oliveira AL. Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Trans Comput Biol Bioinform. 2004;1:24–45. https://doi.org/10.1109/TCBB.2004.2.
3. Santamaría R, Therón R, Quintales L. A visual analytics approach for understanding biclustering results from microarray data. BMC Bioinform. 2008;9:247. https://doi.org/10.1186/1471-2105-9-247.
4. Busygin S, Boyko N, Pardalos PM, Bewernitz M, Ghacibeh G. Biclustering EEG data from epileptic patients treated with vagus nerve stimulation. AIP Conf Proc. 2007;953:220–31. https://doi.org/10.1063/1.2817345.
5. Henriques R, Madeira SC. Bicpam: pattern-based biclustering for biomedical data analysis. Algorithms Mol Biol. 2014;9:27. https://doi.org/10.1186/s13015-014-0027-z.
6. Vandromme M, Jacques J, Taillard J, Jourdan L, Dhaenens C. A biclustering method for heterogeneous and temporal medical data. IEEE Trans Knowl Data Eng. 2020. https://doi.org/10.1109/TKDE.2020.2983692.
7. Xie J, Ma A, Fennell A, Ma Q, Zhao J. It is time to apply biclustering: a comprehensive review of biclustering applications in biological and biomedical data. Brief Bioinform. 2019;20:1450–65. https://doi.org/10.1093/bib/bby014.
8. Xie J, Ma A, Zhang Y, Liu B, Cao S, Wang C, Xu J, Zhang C, Ma Q. Qubic2: a novel and robust biclustering algorithm for analyses and interpretation of large-scale RNA-seq data. Bioinformatics. 2020;36:1143–9. https://doi.org/10.1093/bioinformatics/btz692.
9. Singh M, Mehrotra M. Impact of biclustering on the performance of biclustering based collaborative filtering. Expert Syst Appl. 2018;113:443–56. https://doi.org/10.1016/j.eswa.2018.06.001.
10. Singh M. Scalability and sparsity issues in recommender datasets: a survey. Knowl Inf Syst. 2020;62:1–43. https://doi.org/10.1007/s10115-018-1254-2.
11. Busygin S, Prokopyev O, Pardalos PM. Biclustering in data mining. Comput Oper Res. 2008;35:2964–87. https://doi.org/10.1016/j.cor.2007.01.005.
12. Castanho EN, Aidos H, Madeira SC. Biclustering fMRI time series: a comparative study. BMC Bioinform. 2022;23:192. https://doi.org/10.1186/s12859-022-04733-8.
13. Silva MG, Madeira SC, Henriques R. Water consumption pattern analysis using biclustering: when, why and how. Water. 2022;14:1954. https://doi.org/10.3390/w14121954.
14. Liu Y, Gu Q, Hou JP, Han J, Ma J. A network-assisted co-clustering algorithm to discover cancer subtypes based on gene expression. BMC Bioinform. 2014;15:37. https://doi.org/10.1186/1471-2105-15-37.
15. Carreiro AV, Ferreira AJ, Figueiredo MAT, Madeira SC. Towards a classification approach using meta-biclustering: impact of discretization in the analysis of expression time series. J Integr Bioinform. 2012;9:105–20. https://doi.org/10.1515/jib-2012-207.
16. Li G, Ma Q, Tang H, Paterson AH, Xu Y. Qubic: a qualitative biclustering algorithm for analyses of gene expression data. Nucleic Acids Res. 2009;37:101. https://doi.org/10.1093/nar/gkp491.
17. Henriques R, Madeira SC. Bicnet: flexible module discovery in large-scale biological networks using biclustering. Algorithms Mol Biol. 2016;11:14. https://doi.org/10.1186/s13015-016-0074-8.
18. Tanay A, Sharan R, Kupiec M, Shamir R. Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genomewide data. Proc Natl Acad Sci. 2004;101:2981–6. https://doi.org/10.1073/pnas.0308661100.
19. Castro PAD, Franca FO, Ferreira HM, Zuben FJV. Applying biclustering to perform collaborative filtering. In: 2007 7th international conference on intelligent systems design and applications. IEEE; 2007. p. 421–6. https://doi.org/10.1109/ISDA.2007.91. https://ieeexplore.ieee.org/document/4389645/.
20. Diaz AKR, Peres SM. Biclustering and coclustering: concepts, algorithms and viability for text mining. Rev Inform Teór Apl. 2019;26:81–117. https://doi.org/10.22456/2175-2745.89063.
21. Shaharudin SM, Ismail S, Nor SMCM, Ahmad N. An efficient method to improve the clustering performance using hybrid robust principal component analysis-spectral biclustering in rainfall patterns identification. IAES Int J Artif Intell. 2019;8:237. https://doi.org/10.11591/ijai.v8.i3.pp237-243.
22. Wu X, Cheng C, Qiao C, Song C. Spatio-temporal differentiation of spring phenology in China driven by temperatures and photoperiod from 1979 to 2018. Sci China Earth Sci. 2020;63(10):1485–98. https://doi.org/10.1007/s11430-019-9577-5.
23. Galvani M, Torti A, Menafoglio A, Vantini S. A novel spatio-temporal clustering technique to study the bike sharing system in lyon. In: EDBT/ICDT workshops; 2020.
24. Kaban PA, Kurniawan R, Caraka RE, Pardamean B, Yuniarto B. Sukim: biclustering method to capture the spatial pattern and to identify the causes of social vulnerability in indonesia: a new recommendation for disaster mitigation policy. Procedia Comput Sci. 2019;157:31–7. https://doi.org/10.1016/j.procs.2019.08.138.
25. Pinto-Roa P, D., Medina, H., Román F, García-Torres M, Divina F, Gómez-Vela F, Morales F, Veláizquez G, Daumas F, L. VázquezNoguera J, Sauer Ayala C, Gardel-Sotomayor PE. Parallel evolutionary biclustering of short-term electric energy consumption. In: Computer science & information technology (CS & IT). AIRCC Publishing Corporation; 2021. p. 123–37. https://doi.org/10.5121/csit.2021.111110. https://aircconline.com/csit/papers/vol11/csit111110.pdf. Accessed 26 Sept 2022
26. Pontes B, Girldez R, Aguilar-Ruiz JS. Quality measures for gene expression biclusters. PLoS ONE. 2015;10:0115497. https://doi.org/10.1371/journal.pone.0115497.

27. Prelić A, Bleuler S, Zimmermann P, Wille A, Bühlmann P, Gruissem W, Hennig L, Thiele L, Zitzler E. A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics. 2006;22:1122–9. https://doi.org/10.1093/bioinformatics/btl060.
28. Bozdağ D, Kumar AS, Catalyurek UV. Comparative analysis of biclustering algorithms. In: Proceedings of the first ACM international conference on bioinformatics and computational biology. BCB '10. Association for Computing Machinery, New York, NY, USA; 2010. p. 265–274. https://doi.org/10.1145/1854776.1854814.
29. Eren K, Deveci M, Kucuktunc O, Catalyurek UV. A comparative analysis of biclustering algorithms for gene expression data. Brief Bioinform. 2013;14:279–92. https://doi.org/10.1093/bib/bbs032.
30. Padilha VA, Campello RJGB. A systematic comparative evaluation of biclustering techniques. BMC Bioinform. 2017;18:55. https://doi.org/10.1186/s12859-017-1487-1.
31. Lobo J, Henriques R, Madeira SC. G-tric: generating three-way synthetic datasets with triclustering solutions. BMC Bioinform. 2021;22:16. https://doi.org/10.1186/s12859-020-03925-4.
32. Henriques R, Madeira SC. Triclustering algorithms for three-dimensional data analysis. ACM Comput Surv. 2019;51:1–43. https://doi.org/10.1145/3195833.
33. Madeira SC, Oliveira AL. A linear time biclustering algorithm for time series gene expression data. https://doi.org/10.1007/11557067_4.
34. Gonçalves JP, Madeira SC, Oliveira AL. Biggests: integrated environment for biclustering analysis of time series gene expression data. BMC Res Notes. 2009;2:124. https://doi.org/10.1186/1756-0500-2-124.
35. Henriques R, Madeira SC. Bsig: evaluating the statistical significance of biclustering solutions. Data Min Knowl Discov. 2018;32:124–61. https://doi.org/10.1007/s10618-017-0521-2.
36. Lazzeroni L, Owen A. Plaid models for gene expression data. Stat Sin. 2002;12:61–86.
37. Henriques R, Ferreira FL, Madeira SC. Bicpams: software for biological data analysis with pattern-based biclustering. BMC Bioinform. 2017;18:82. https://doi.org/10.1186/s12859-017-1493-3.
38. Horta D, Campello RJGB. Similarity measures for comparing biclusterings. IEEE/ACM Trans Comput Biol Bioinform. 2014;11:942–54. https://doi.org/10.1109/TCBB.2014.2325016.
39. Goutte C, Gaussier E. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: Losada DE, Fernández-Luna JM, editors. Advances in information retrieval. Berlin: Springer; 2005. p. 345–59.
40. Hochreiter S, Bodenhofer U, Heusel M, Mayr A, Mitterecker A, Kasim A, Khamiakova T, Sanden SV, Lin D, Talloen W, Bijnens L, Göhlmann HWH, Shkedy Z, Clevert D-A. Fabia: factor analysis for bicluster acquisition. Bioinformatics. 2010;26:1520–7. https://doi.org/10.1093/bioinformatics/btq227.
41. Hartigan JA. Direct clustering of a data matrix. J Am Stat Assoc. 1972;67:123–9. https://doi.org/10.1080/01621459.1972.10481214.
42. Cheng Y, Church GM. Biclustering of expression data. In: Proceedings of the international conference on intelligent systems for molecular biology, vol 8; 2000. p. 93–103.
43. Divina F, Pontes B, Giráildez R, Aguilar-Ruiz JS. An effective measure for assessing the quality of biclusters. Comput Biol Med. 2012;42:245–56. https://doi.org/10.1016/j.compbiomed.2011.11.015.
44. Aguilar-Ruiz JS. Shifting and scaling patterns from gene expression data. Bioinformatics. 2005;21:3840–5. https://doi.org/10.1093/bioinformatics/bti641.
45. Foss AH, Markatou M, Ray B. Distance metrics and clustering methods for mixed-type data. Int Stat Rev. 2019;87(1):80–109. https://doi.org/10.1111/insr.12274.
46. Boriah S, Chandola V, Kumar V. Similarity Measures for categorical data: a comparative evaluation. In: Proceedings of the 2008 SIAM international conference on data mining. Society for Industrial and Applied Mathematics; 2008. p. 243–254. https://doi.org/10.1137/1.9781611972788.22 . https://epubs.siam.org/doi/10.1137/1.9781611972788.22. Accessed 02 Oct 2021.
47. Preud'homme G, Duarte K, Dalleau K, Lacomblez C, Bresso E, Smaïl-Tabbone M, Couceiro M, Devignes M-D, Kobayashi M, Huttin O, Ferreira JP, Zannad F, Rossignol P, Girerd N. Head-to-head comparison of clustering methods for heterogeneous data: a simulation-driven benchmark. Sci Rep. 2021;11:4202. https://doi.org/10.1038/s41598-021-83340-8.
48. Patrikainen A, Meila M. Comparing subspace clusterings. IEEE Trans Knowl Data Eng. 2006;18:902–16. https://doi.org/10.1109/TKDE.2006.106.
49. Rodriguez-Baena DS, Perez-Pulido AJ, Aguilar-Ruiz JS. A biclustering algorithm for extracting bit-patterns from binary datasets. Bioinformatics. 2011;27:2738–45. https://doi.org/10.1093/bioinformatics/btr464.
50. Tanay A, Sharan R, Shamir R. Discovering statistically significant biclusters in gene expression data. Bioinformatics. 2002;18:136–44. https://doi.org/10.1093/bioinformatics/18.suppl_1.S136.
51. Omari A, Langer R, Conrad S. Tartool: A temporal dataset generator for market basket analysis. Springe; 2008. p. 400–410. https://doi.org/10.1007/978-3-540-88192-6_37.
52. Pei Y, Zaiane OR. A synthetic data generator for clustering and outlier analysis. 2006. https://doi.org/10.7939/R3B23S.
53. de Melo CM, Torralba A, Guibas L, DiCarlo J, Chellappa R, Hodgins J. Next-generation deep learning based on simulators and synthetic data. Trends Cogn Sci. 2022;26(2):174–87. https://doi.org/10.1016/j.tics.2021.11.008.
54. Yeomans J, Thwaites S, Robertson WSP, Booth D, Ng B, Thewlis D. Simulating time-series data for improved deep neural network performance. IEEE Access. 2019;7:131248–55. https://doi.org/10.1109/ACCESS.2019.2940701.
55. Erhardt EB, Allen EA, Wei Y, Eichele T, Calhoun VD. SimTB, a simulation toolbox for fMRI data under a model of spatiotemporal separability. Neuroimage. 2012;59(4):4160–7. https://doi.org/10.1016/j.neuroimage.2011.11.088.
56. Coletta A, Prata M, Conti M, Mercanti E, Bartolini N, Moulin A, Vyetrenko S, Balch T. Towards realistic market simulations: a generative adversarial networks approach. In: Proceedings of the second ACM international conference on AI in finance. ICAIF '21. Association for Computing Machinery, New York, NY, USA; 2022. https://doi.org/10.1145/3490354.3494411.
57. Xu M, Niyato D, Chen J, Zhang H, Kang J, Xiong Z, Mao S, Han Z. Generative AI-empowered Simulation for autonomous driving in vehicular mixed reality metaverses; 2023.
58. Henriques R. Learning from high-dimensional data using local descriptive models.

59. Henriques R, Madeira SC. Biclustering with flexible plaid models to unravel interactions between biological processes. IEEE/ACM Trans Comput Biol Bioinf. 2015;12:738–52. https://doi.org/10.1109/TCBB.2014.2388206.
60. Wang S, Yin Y, Ma Q, Tang X, Hao D, Xu Y. Genome-scale identification of cell-wall related genes in Arabidopsis based on co-expression network analysis. BMC Plant Biol. 2012;12(1):138. https://doi.org/10.1186/1471-2229-12-138.
61. Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM international conference on information and knowledge management. ACM, Beijing China; 2019. p. 1441–1450. https://doi.org/10.1145/3357384.3357895. https://dl.acm.org/doi/10.1145/3357384.3357895. Accessed 14 Jan 2022.
62. Mimaroglu S, Uehara K. Bit sequences and biclustering of text documents. In: Seventh IEEE international conference on data mining workshops (ICDMW 2007). IEEE, Omaha, NE, USA; 2007. p. 51–56. https://doi.org/10.1109/ICDMW.2007.38. http://ieeexplore.ieee.org/document/4476646/. Accessed 14 Jan 2022.
63. Sun J, Zhang Y. Recommendation system with biclustering. Big Data Min Anal. 2022;5:282–93. https://doi.org/10.26599/BDMA.2022.9020012.
64. Silva MG, Henriques R, Madeira SC. User-specific bicluster-based collaborative filtering: handling preference locality, sparsity and subjectivity; 2022.
65. Choi S, Ha H, Hwang U, Kim C, Ha J-W, Yoon S. Reinforcement learning based recommender system using biclustering technique. 2018. arXiv preprint arXiv:1801.05532.
66. Gallo CA, Maguitman AG, Carballido JA, Ponzoni I. Biclustering in data mining using a memetic multi-objective evolutionary algorithm. In: XIV Congreso Argentino de Ciencias de la Computación; 2008.
67. Rosenwald A, Wright G, Chan WC, Connors JM, Campo E, Fisher RI, Gascoyne RD, Muller-Hermelink HK, Smeland EB, Giltnane JM, Hurt EM, Zhao H, Averett L, Yang L, Wilson WH, Jaffe ES, Simon R, Klausner RD, Powell J, Duffey PL, Longo DL, Greiner TC, Weisenburger DD, Sanger WG, Dave BJ, Lynch JC, Vose J, Armitage JO, Montserrat E, López-Guillermo A, Grogan TM, Miller TP, LeBlanc M, Ott G, Kvaloy S, Delabie J, Holte H, Krajci P, Stokke T, Staudt LM. The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma. N Engl J Med. 2002;346(25):1937–47. https://doi.org/10.1056/NEJMoa012914.