Clustering and variable selection evaluation of 13 unsupervised methods for multi-omics data integration

Morgane Pierre-Jean,   Jean-François Deleuze,   Edith Le Floch[†] and Florence Mauger[†]

Corresponding author: Morgane Pierre-Jean, Centre National de Recherche en Génomique Humaine (CNRGH), Institut de Biologie François Jacob, CEA, Université Paris-Saclay, F-91057, Evry, France. Tel.: +33(0)160878392; Fax: +33(0)160878485; E-mail: mpierre@cng.fr
[†]The authors contributed equally to this work.

Abstract

Recent advances in NGS sequencing, microarrays and mass spectrometry for omics data production have enabled the generation and collection of different modalities of high-dimensional molecular data. The integration of multiple omics datasets is a statistical challenge, due to the limited number of individuals, the high number of variables and the heterogeneity of the datasets to integrate. Recently, a lot of tools have been developed to solve the problem of integrating omics data including canonical correlation analysis, matrix factorization and SM. These commonly used techniques aim to analyze simultaneously two or more types of omics. In this article, we compare a panel of 13 unsupervised methods based on these different approaches to integrate various types of multi-omics datasets: iClusterPlus, regularized generalized canonical correlation analysis, sparse generalized canonical correlation analysis, multiple co-inertia analysis (MCIA), integrative-NMF (intNMF), SNF, MoCluster, mixKernel, CIMLR, LRAcluster, ConsensusClustering, PINSPlus and multi-omics factor analysis (MOFA). We evaluate the ability of the methods to recover the subgroups and the variables that drive the clustering on eight benchmarks of simulation. MOFA does not provide any results on these benchmarks. For clustering, SNF, MoCluster, CIMLR, LRAcluster, ConsensusClustering and intNMF provide the best results. For variable selection, MoCluster outperforms the others. However, the performance of the methods seems to depend on the heterogeneity of the datasets (especially for MCIA, intNMF and iClusterPlus). Finally, we apply the methods on three real studies with heterogeneous data and various phenotypes. We conclude that MoCluster is the best method to analyze these omics data. **Availability:** An R package named `CrIMMix` is available on GitHub at https://github.com/CNRGH/crimmix to reproduce all the results of this article.

Key words: multi-omics; unsupervised integrative methods; benchmarks; real data; performance evaluation

Introduction

Recent technological improvements for omics data production including NGS sequencing, microarray and mass spectrometry have enabled the production and collection of different modalities of high-dimensional molecular data such as genomic, epigenomic, transcriptomic, proteomic, lipidomic and metabolomic [1, 2]. Combining these various types of multi-omics datasets and clinical data could have the potential to enhance a comprehensive view of the mechanisms of disease or biological process [3–5]. Understanding molecular behaviors, pathway interactions and relationships between and within the different types of data could improve the diagnosis, prognosis and monitoring therapy treatment of cancer [6]. For instance, The Cancer Genome Atlas (TCGA) [7] and the International Cancer Genome Consortium [8] produced and collected thousands of tumor samples at different molecular levels including DNA (somatic mutation, copy number variation), DNA methylation, RNA (or microRNA) and also proteins for the same patients [9]. Although cancer is the main application that uses multi-omic integration, the analysis of complex diseases or single-cell is now emerging [6, 10–14]. Metabolomic and lipidomic were also combined with genomic for a better knowledge of phenotypes [13, 15–17]. The analysis of multi-omics datasets could be performed using specific statistical unsupervised integrative methods, which allow the combination of multiple blocks. These methods could provide new subgroups of patients of a specific disease [18, 19] and identify candidate biomarkers.

However, integrating multi-omics datasets involves several challenges including the high dimension of datasets (for instance, whole genome sequencing analyzes the 3 billions of base pairs of the human genome), the limited number of patients, the heterogeneity of datasets and the modeling of interactions between the different types of omics data [20, 21]. Indeed, multi-omics data are obtained using several technologies that generate different types of datasets with various dimensions. Due to the high dimensionality of the produced data, a lot of methods are based on dimension reduction techniques [22] or SM for clustering [23]. Recently, several unsupervised and supervised integrative methods for multi-omics have been compared [20, 22–26].

In this article, we evaluated 13 unsupervised methods that are mainly based on three types of statistical approaches: matrix factorization (MF), canonical correlation and co-inertia analysis (CCA and CIA) and similarity matrices (SM). MF and CCA are based on the same principles, namely dimension reduction and latent variable models [22].

The selected unsupervised methods include regularized and sparse generalized canonical correlation analysis [27, 28], integrative non-negative matrix factorization [29], multiple co-inertia analysis [30], similarity network fusion [18], two multiple kernel learning methods [31, 32], multi-omics factor analysis (MOFA) [33], low-rank approximation method [34], consensus clustering [35, 36], perturbation clustering [37] and two integrative clustering methods (iClusterPlus [38] and MoCluster [39]).

Methods

One of the most important criteria to select the methods is the availability of the code. Thus, we assess 13 methods that are freely available as R packages, which ensures a good documentation. These methods are listed in Table 1 and described below.

Table 1. Description of selected methods for multi-omics data integration: name, type of method (CCA, MF, SM), brief description, capacity to perform variable selection, availability of a clustering output, name of R package and the platform

Method	Type	Name	Variable selection	Clustering output	Package R	Platform
RGCCA [27]	CCA	Regularized generalized canonical correlation analysis	No	No	RGCCA	CRAN
intNMF [29]	MF	Integrative non-negative matrix factorization		Yes	intNMF	CRAN
SNF [18]	SM	Similarity network fusion		Yes	SNFtool	CRAN
LRACluster [34]	SM	Low-rank clustering		Yes	LRACluster	Personal website
PINSPlus [40]	SM	Perturbation clustering for data INtegration and disease Subtyping		Yes	PINSPlus	CRAN
ConsensusClustering [35, 36]	SM	Consensus clustering		Yes	Consensus ClusteringPlus	Bioconductor
MCIA [30]	CCA-like	Multiple-co-inertia analysis		No	Omicade4	Bioconductor
mixKernel [31]	SM	Multiple kernel learning		No	mixKernel	CRAN
SGCCA [28]	CCA	Sparse generalized canonical correlation analysis	Yes	No	RGCCA	CRAN
iClusterPlus [38]	MF	Integrative clustering and modeling of datasets extension		Yes	iClusterPlus	Bioconductor
MoCluster [39]	MF	Clustering and identifying joint patterns across multiple-omic datasets		Yes	mosga	Bioconductor
CIMLR [32]	SM	Cancer integration via multikernel learning		Yes	CIMLR	Github
MOFA [33]	Bayesian MF	MOFA		No	MOFAtools	Github

Notations

In the following, $\mathbf{A}$ denotes a matrix, $\mathbf{a}$ a vector and a a scalar. For each method, we consider K matrices $\mathbf{X}_1, \ldots \mathbf{X}_K$ as input of each method. Each matrix $\mathbf{X}_k$ is of size $n \times J_k$ (n is the number of samples and J_k the number of variables for the block k).

Regularized generalized canonical correlation analysis and sparse generalized canonical correlation analysis

Regularized generalized canonical correlation analysis (RGCCA) has been introduced by Tenenhaus *et al.* in [27]. This method considers a connection network between the various blocks of data by defining a design matrix $\mathbf{C} = c_{jk} : c_{jk} = 1$ if blocks j and k are connected and 0 otherwise. The RGCCA is defined as the following problem for one latent variable by block:

$$\max_{\mathbf{a}_1 \ldots, \mathbf{a}_K} \sum_{j,k=1, j \neq k}^{K} c_{jk} g(Cov(\mathbf{X}_j \mathbf{a}_j, \mathbf{X}_k \mathbf{a}_k)) \tag{1}$$

subject to the following constraints $\tau_k \|\mathbf{a}_k\|^2 + (1 - \tau_k) Var(\mathbf{X}_k \mathbf{a}_k) = 1$, $k = 1, \ldots K$. The parameter τ is defined by the user and can be seen as a shrinkage parameter. τ is in fact a parameter that controls the compromise between covariance and correlation. If $\tau_k = 1$, then the users give priority to the covariance between the latent variables $\mathbf{X}_k \mathbf{a}_k$ compared to the correlation. Otherwise, if $\tau_k = 0$, then the users give priority to the correlation between latent variables. The function g can be either the identity, the absolute value or the square function. For more details on this parameter see [27].

The authors of [27] proposed an algorithm to solve the problem defined by 1 and also an R package called RGCCA.

The authors of [28] expand RGCCA to address the variable selection.

The sparse generalized canonical correlation analysis (SGCCA) is defined as the following problem:

$$\max_{\mathbf{a}_1 \ldots, \mathbf{a}_K} \sum_{j,k=1, j \neq k}^{K} c_{jk} g(Cov(\mathbf{X}_j \mathbf{a}_j, \mathbf{X}_k \mathbf{a}_k)) \tag{2}$$

subject to the following constraints $\|\mathbf{a}_k\|_2 = 1$ and $\|\mathbf{a}_k\|_1 \leq s_k$, $k = 1, \ldots, K$, where s_j is a positive constant that determines the amount of sparsity $\mathbf{a}_j$, $j = 1, \ldots, J$. The smaller s_j, the larger the degree of sparsity for $\mathbf{a}_j$.

In both RGCCA and SGCCA, users need to tune multiple parameters: for RGCCA: τ and the number of latent variables for each block and for SGCCA s_j and the number of latent variables for each block.

Multiple co-inertia analysis

Multiple co-inertia analysis (MCIA) [30] is an exploratory data analysis method that identifies relationships between multiple high dimensional datasets. The first step is to apply on each dataset separately a table ordination method, such as principal component analysis (PCA). This step allows to transform data into comparable lower dimensional spaces. After this step, we obtain K transformed matrices $\tilde{\mathbf{X}}_k$ for each considered dataset, and $\tilde{\mathbf{X}}_k$ is centered.

MCIA can be written as a particular case of RGCCA. Following equation 1, $g = x^2$ and $c_{jk} = 1$, MCIA optimize the equation:

$$\max f(\mathbf{a}_1, \ldots, \mathbf{a}_K) = \max \sum_{k=1}^{K} cov^2 \left(\tilde{\mathbf{X}}_k \mathbf{a}_k, \tilde{\mathbf{X}} \mathbf{a} \right) \tag{3}$$

where $\tilde{\mathbf{X}} = (\tilde{\mathbf{X}}_1 | \ldots | \tilde{\mathbf{X}}_K)$ is a superblock with the constraint $\|\mathbf{a}_k\|^2 = Var(\mathbf{X}\mathbf{a}) = 1$ and $\mathbf{a} = (\mathbf{a}_1 | \ldots | \mathbf{a}_K)$. $\mathbf{a}$ represents the global loading vector where the individuals will be represented. MCIA allows only one common latent variable space with $\mathbf{a}$. MCIA is available in R package omicade4.

Integrative-NMF

The main goal of the MF technique is to decompose a matrix $\mathbf{X}_k$ into two matrices: a common basis matrix denoted $\mathbf{W}$ and a specific coefficient matrix denoted $\mathbf{H}_k$ by assuming that

$$\mathbf{X}_k \approx \mathbf{W}\mathbf{H}_k \tag{4}$$

where $\mathbf{X}_k$ is a matrix of size $n \times J_k$, $\mathbf{W}$ is a matrix of size $n \times d$ and $\mathbf{H}_k$ is a matrix of size $d \times J_k$. The latent variables for this model are $\mathbf{H}_k$ and $\mathbf{W}$.

The non-negative matrix factorization (NMF) is an approach that has already been used for disease subtype classification. For instance, it has been used in cancer subtype discovery from gene expression data [19, 41]. In [29] an extension is proposed to include simultaneously a single analysis multiple omic datasets and named it integrative-NMF (intNMF). intNMF uses the model define by equation 4 where all coefficients of $\mathbf{W}$ and $\mathbf{H}_k$ are non-negative. The classical objective function in this kind of problem is defined as the Frobenius-norm. intNMF solves the following problem:

$$\min_{\mathbf{W}, \mathbf{H}_1, \ldots \mathbf{H}_k} \sum_{k=1}^{K} \|\mathbf{X}_k - \mathbf{W}\mathbf{H}_k\|_F \tag{5}$$

subject to $\mathbf{W} \geq 0$ and $\mathbf{H}_k \geq 0$. intNMF added a parameter θ_k for each dataset, specified by users, which aims to control the weight of each dataset in the analysis. Then the objective function becomes

$$\min_{\mathbf{W}, \mathbf{H}_1, \ldots \mathbf{H}_k} \sum_{k=1}^{K} \theta_k \|\mathbf{X}_k - \mathbf{W}\mathbf{H}_k\|_F \quad \text{s.t} \quad \mathbf{W} \geq 0 \text{ and} \mathbf{H}_k \geq 0. \tag{6}$$

By default, θ_k is the maximum of the mean sums of squares on the matrix among all data divided by the mean sums of squares of each data (see equation 7). This implies that the dataset with the largest values has a weight of 1, and the dataset with the smallest values has a weight larger than 1.

$$\theta_k = \frac{\max_{k=1, \ldots, K} mean \|\mathbf{X}_k\|_F}{mean \|\mathbf{X}_k\|_F}. \tag{7}$$

Several algorithms are well-known to solve this kind of problem [42]. intNMF does not assume any distributional form for the data $\mathbf{X}_k$ and the non-negative constraint could induce a certain level of sparsity in both $\mathbf{W}$ and $\mathbf{H}_k$. An R package named intNMF is available and is easy to use.

iCluster and iClusterPlus

iCluster and iClusterPlus [38, 42] also aim to recover $\mathbf{W}$ and $\mathbf{H}_k$ the latent variables in equation 4. The updated version, `iClusterPlus`, allows modeling Poisson, binomial, Gaussian and multinomial distributions. Here, we present only modelization for continuous and binary data, the two types of datasets that have been used in this article. As previously, $\mathbf{X}_k$ is a matrix of size $n \times J_k$.

The continuous data model can be written as follows:

$$x_{ijk} = \alpha_{jk} + \mathbf{h}_{jk}\mathbf{w}_i + \epsilon_{ijk} \tag{8}$$

where $\epsilon_{ijk} \sim \mathcal{N}(0, \sigma_{jk}^2)$.

The Binary model can be written as follows:

$$\log \frac{P(x_{ijk} = 1|\mathbf{w}_i)}{1 - P(x_{ijk} = 1|\mathbf{w}_i)} = \alpha_{jk} + \mathbf{h}_{jk}\mathbf{w}_i. \tag{9}$$

In equation 9, $P(x_{ijk}|\mathbf{w}_i)$ is the probability of gene j mutated in sample i given the latent factor $\mathbf{w}_i$. α_{jk} is an intercept term, and $\mathbf{h}_{jk}$ is a row vector of size p that determine the weight of the genomic feature j.

To obtain a sparse model, the likelihood is solved with a lasso penalty (L1-norm) on $\mathbf{H}_k$ (equation 10).

$$\max_{\alpha_{jk}, \mathbf{h}_{jk}} \ell(x_{ijk}, \mathbf{w}_i; \alpha_{jk}, \mathbf{h}_{jk}) - \sum_{k=1}^{K} \sum_{j=1}^{J^k} \lambda_k \|\mathbf{h}_{jk}\|_1 \tag{10}$$

As intNMF, the latent variables of iClusterPlus are the matrices $\mathbf{W}$ and $\mathbf{H}_k$. This method is implemented in the package named `iClusterPlus`, which is well documented and relatively easy to use.

MoCluster approach

Another MF method that we assess is `MoCluster` [39]. The model can be expressed as those described by equation 4.

The first step of MoCluster uses the consensus PCA (CPCA) algorithm to estimate the block latent variables for each block denoted $\mathbf{H}_k, k = 1, \dots K$. Then, to define a common space of variable the algorithm maximizes the correlation between $\mathbf{H}_k$ and $\mathbf{H}$ (the joint latent variables). The MoCluster algorithm introduces sparsity in feature coefficient vectors ($\mathbf{H}$) to improve the biological interpretation of clustering results. The sparsity is included by the way of a Soft-thresholding operator that controls the number of nonzero coefficient. The main advantage of the CPCA algorithm, in contrast to iCluster/iClusterPlus that uses an EM-algorithm, is the time-saving. Then, $\mathbf{W}$ matrix used to perform clustering of samples is defined by the projection of the samples on the new subspace defined by $\mathbf{H}$.

CPCA is closely related to RGCCA and MCIA, but has had less exposure to the omics data community. CPCA optimizes the same criterion as RGCCA and MCIA and is subject to the same constraints of normality as MCIA [22, 43], i.e. $\|\mathbf{H}\| = 1$. The algorithm used in [39] guarantees the orthogonality of the global scores $\mathbf{H}$ and tends to find common patterns in the datasets.

`MoCluster` is implemented in the R package named `mogsa` and is well documented.

MOFA

More recently, the model MOFA has been introduced in [33]. It can be written as that of intNMF:

$$\mathbf{X}_k = \mathbf{W}\mathbf{H}^k + \epsilon^k, k = 1, \dots K \tag{11}$$

$\mathbf{W} \in \mathbb{R}^{n \times d}$ is the weight matrix for the individuals, $\mathbf{H}^k \in \mathbb{R}^{d \times J_k}$ is the loading matrix for variables in the low dimensional space and $\epsilon^k \in \mathbb{R}^{n \times J_k}$ denotes the residual noise.

For the Gaussian model $\epsilon_{ij}^k \sim \mathcal{N}(0, 1/\tau_j^k)$, with Gamma prior on the precision parameters $1/\tau_j^k$. $1/\tau_j^k \sim \mathcal{G}(a_0, b_0)$, with $a_0 = b_0 = 1e - 14$ to obtain informative priors.

Then, in the Gaussian case, $\mathbf{X}_{ij}^k \sim \mathcal{N}(\mathbf{W}_{i\bullet}\mathbf{H}_{\bullet j}^k, 1/\tau_j^k)$. Authors added sparsity constraints on both $\mathbf{W}$ and $\mathbf{H}^k$ matrices by using an automatic relevance determination prior: a re-parametrisation of spike-and-slab priors.

MOFA models other kinds of noise such as binary and count data. MOFA uses respectively Bernoulli and Poisson distributions for these two kinds of data. The model likelihood is given by $\mathbf{X}_{ij}^k \sim \mathcal{B}(f(\mathbf{W}_{i\bullet}\mathbf{H}_{\bullet j}^k))$ for the Binary data where f denotes the logistic inverse function $f(x) = (1 + e^{-x})^{-1}$.

MOFA is implemented in R as a well-documented open-source software and comes with tutorials and example workflows.

SNF

To finish, we explored five methods based on SM. The first one is the similarity network fusion [18] that aims to cluster the patients into several groups in order to discover new subtypes of a given disease. A patient similarity network for a dataset k is represented by a graph $G_k = (V, E_k)$ where V corresponds to the patients and the edges E_k are weighted by how similar the patients are. $\mathbf{X}_k$ is still the datasets k of size $n \times J_k$. The first step of this method consists in computing an SM $\mathbf{W}_k$ for each type of dataset $k = 1, \dots K$.

$$\mathbf{W}_k(i, i') = \exp\left(-\frac{\rho^2(\mathbf{X}_k(i, \bullet)\mathbf{X}_k(i', \bullet))}{\mu \epsilon_{i,i'}}\right) \tag{12}$$

The parameter μ can be empirically set. Authors advise to set this parameter in the range of $[0.3, 0.8]$ and is 0.5 by default.

Let us denote by N_i the set of $\mathbf{X}_k(i, \bullet)$'s neighbors including itself in a graph G_k for the dataset k. The size of N_i can be defined by the user and is equal to 10 by default. Then, $\rho_{ki} = \rho(\mathbf{X}_k(i, \bullet), N_i)$ is the average value of the distance between $\mathbf{X}_k(i, \bullet)$ and each neighbors in the N_i.

$\epsilon_{i,i'}$ is used to fix the scaling problem and is defined by the following formula:

$$\epsilon_{i,i'} = \frac{mean(\rho_{ki}) + mean(\rho_{ki'}) + \rho(\mathbf{X}_k(i, \bullet), \mathbf{X}_k(i', \bullet))}{3}.$$

The next step is to combine the different SM and to perform clustering on this output. Two matrices $\mathbf{P}$ and $\mathbf{S}$ are defined. The first one $\mathbf{P}$ carries the full information about the similarity to all

others and the second one **S** only represents the similarity to the *J* most similar patients for each patient.

$$
\mathbf{P}_k(i, i') = \begin{cases} \frac{\mathbf{W}_k(i,i')}{2\sum_{i \neq i'} \mathbf{W}_k(i,i')}, i \neq i' \\ 1/2, i = i' \end{cases}
$$

$$
\mathbf{S}_k(i, i') = \begin{cases} \frac{\mathbf{W}_k(i,i')}{\sum_{i' \in N_i} \mathbf{W}_k(i,i')}, i' \in N_i \\ 0, otherwise \end{cases}
$$

By using a new matrix $\tilde{\mathbf{P}}_k$ defined by

$$
\tilde{\mathbf{P}}_k = S_k \times \frac{\sum_{k' \neq k} \mathbf{P}_{k'}}{K-1} \times S_k^{\mathrm{T}}, k = 1, \dots, K. \qquad (13)
$$

Then, the SM used to perform the clustering is the normalized matrix sum defined by the previous equation 13.

An R package named `SNFtool` that implemented this method is available.

MixKernel

Recently, Kernel learning methods emerged in multi-omics analysis [31, 44]. Kernel learning methods compute SM of datasets from kernels. Consequently, it is easy to combine the kernel of each dataset to obtain a global SM.

For a given set of observations $(x_i)_{i=1,\dots,N}$, taking values in an arbitrary space $\mathcal{X}$, we call kernel a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that provides pairwise similarities between the observations: $K_{ij} := K(x_i, x_j)$. Moreover, this function is assumed to be symmetric $K_{ij} = K_{ji}$ and positive. According to [45], this ensures that K is the dot product in a uniquely defined Hilbert space of the images.

This kernel can thus be used in subsequent analyses (support vector machine, kernel PCA (KPCA), kernel self-organizing map) as it is supposed to provide an integrated summary of the samples. We used here the KPCA method on our simulations. The R package `MixKernel` enables to run the expanded KPCA.

CIMLR

Recently, CIMLR [32], a multikernel learning method, has been developed to classify cancers. First, the method uses the several Gaussian kernel defined by

$$
\mathcal{K}(\mathbf{X}_k(i, \bullet), \mathbf{X}_k(j, \bullet)) = \frac{1}{\epsilon_{ij}\sqrt{2\pi}} \exp\left(-\frac{\|\mathbf{X}_k(i, \bullet) - \mathbf{X}_k(j, \bullet)\|_2^2}{2\epsilon_{ij}^2}\right) \qquad (14)
$$

where $\mathbf{X}_k(i, \bullet)$ and $\mathbf{X}_k(j, \bullet)$ are the i-th and the j-th rows of the matrix $\mathbf{X}_k$. ϵ_{ij}^2 is a variance term that can be calculated with different ways [46].

CIMLR method performs the above computation for each block independently, to obtain a set of P Gaussian kernels with different variance per block. Then, it optimizes the following equation defined in [46]:

$$
\min_{\mathbf{S}, \mathbf{L}, w} - \sum_{i,j} \left\{ \sum_{l=1}^{P} \sum_{k=1}^{K} w_{lk} \mathcal{K}_l(\mathbf{X}_k(i, \bullet), \mathbf{X}_k(j, \bullet)) \right\} \mathbf{S}_{ij} +
$$

$$
\beta \|\mathbf{S}\|_F^2 + \gamma tr(\mathbf{L}^{\mathrm{T}}(\mathbf{I}_n - \mathbf{S})\mathbf{L}) + \rho \sum_l \sum_k w_{lk} \log(w_{lk})
$$

$$
\text{subject to} \mathbf{L}^{\mathrm{T}}\mathbf{L} = \mathbf{I}_C, \sum_l \sum_k w_{lk} = 1, w_{lk} \geq 0,
$$

$$
\sum_j \mathbf{S}_{ij} = 1, \text{ and} \mathbf{S}_{ij} \geq 0 \qquad (15)
$$

where, n is the number of individuals, C the number of clusters, i and j are two individuals and l the Kernel index from 1 to P. The equation is solved for **S** the $n \times n$ SM, w_{lk} the weights of each kernel in each block of data and **L** an auxiliary low-dimensional matrix of size $n \times C$ enforcing the low rank constraint on **S**. $\mathbf{I}_n$ and $\mathbf{I}_C$ are $n \times n$ and $C \times C$ identity matrices, β, γ and ρ are non-negative tuning parameters and $\|\mathbf{S}\|_F^2$ is the Frobenius norm of **S**. The clustering is deduced from **S** by the k-means method.

CIMLR includes a second step to define which are the relevant features. For each subgroup, a hypergeometric test is used on variables to discriminate those that drive the clustering. From this test, a P-value is then attributed to each variable.

The package `CIMLR` is available.

PINS and PINSPLUS

PINS (Perturbation clustering for data INtegration and disease Subtyping) is a method based on SM. For the steps detailed below, let us first assume that there is one block of data. The first step of PINS [37] consists in computing a SM and then performing a hierarchical clustering of the patients that is cut for each possible number of clusters from $c = 2$ to $c = C$, where the partitioning in c clusters is denoted by $\mathcal{C}_c$. After this step, we have $(C-1)$ partitions. Then, PINS uses a pair-wise connectivity matrix for each partitioning. Two individuals are connected if they are clustered together. The connectivity matrix $\mathbf{M}_c$ from the partitioning $\mathcal{C}_c$ is defined by

$$
\mathbf{M}_c = \begin{cases} 1 & \text{if individuals} \quad i, j \quad \text{belong to the same group} \\ 0 & \text{otherwise} \end{cases}
$$

$$(16)$$

$C-1$ connectivity matrices are computed (for each partition $\mathcal{C}_c$); therefore, $C-1$ connectivity matrices $\mathbf{M}_c$ have been obtained.

PINS generates perturbed matrices to evaluate the stability of clustering. The method consists in perturbing the dataset by adding Gaussian noise to the original data. From these perturbed datasets, they compute connectivity matrices and then an average of these connectivity matrices.

If the original connectivity matrix $\mathbf{M}_c$ for a given c is identical to the average of perturbed connectivity matrices this means that perturbations do not affect the clustering results. The best number of subgroups is the one that minimizes the difference between the original (without the perturbation) connectivity matrix and the perturbed connectivity matrix.

For multi-omics data integration, they compute one connectivity matrix for each dataset and then they perform an average of these matrices. The choice of the best number of subgroups is made using the perturbation procedure described above on such averaged (over blocks) connectivity matrices. Finally for the

best number of subgroups, they apply a hierarchical clustering on the average connectivity matrix. The package `PINSPlus` that implements PINS methods faster than the original package PINS [40] has been used.

Consensus clustering

The Consensus clustering method described in [35] combines several omics datasets in order to cluster patients. The first step is the concatenation of the K matrices $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_K$, denoted by the super block $\mathbf{X} = (\mathbf{X}_1 | \ldots | \mathbf{X}_K)$. After computing a SM on $\mathbf{X}$, a first clustering is carried out for each number of clusters from $c = 1$ to $c = C$.

Then, a connectivity matrix is defined by the same equation as in PINSPlus (equation 16) for each possible value of c. ConsensusClustering perturbs the original superblock $\mathbf{X}$ by sub-sampling individuals. The P perturbed datasets are denoted by $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(P)}$. Then, $\mathbf{I}^{(p)}$ is a $n \times n$ matrix that indicates if individuals i and j are both present in the same dataset $\mathbf{X}^{(p)}$, and the p-th connectivity matrix associated to $\mathbf{X}^{(p)}$ is denoted by $\mathbf{M}_c^{(p)}$.

A consensus matrix ($\mathbf{C}_c$) for c clusters is an $n \times n$ matrix that stores, for each pair of individuals, the proportion of clustering runs in which two individuals are clustered together. $\mathbf{C}_c$ is obtained by taking the average over the connectivity matrices of every perturbed dataset.

$$\mathbf{C}_c(i, j) = \frac{\sum_p \mathbf{M}_c^{(p)}(i, j)}{\sum_p \mathbf{I}^{(p)}(i, j)} \tag{17}$$

By definition, $\mathbf{C}_c$ is symmetric and provides a similarity measure that can be used as an input of a hierarchical clustering method. The package R `ConsensusClusteringPlus` presented in [36] has been used.

LRAcluster

LRAcluster [34] is based on the low-rank approximation method. For LRAcluster, individuals are in columns and features are in rows. Then, here $\mathbf{X}_k(j, i)$, is the value of the feature j and the individual i for the block k. The user can specify the distribution $Pr(\mathbf{X}_k(j, i)|\theta_{ji}^k)$ of each block among Gaussian, Binary and Poisson distributions where θ_{ji}^k is the parameter for each distribution. In this article, we used the Gaussian and the Binary distributions. Then, the likelihood function is defined by

$$L(\theta^k; \mathbf{X}_k) = -\sum_{ji} ln(Pr(\mathbf{X}_k(j, i)|\theta_{ji}^k)). \tag{18}$$

The overall likelihood function is the sum of the likelihood functions of the different blocks (equation 19).

$$L(\theta) = \sum_k L(\theta^k; \mathbf{X}_k) \tag{19}$$

The assumption of the model is that θ has a low-rank structure. This assumption is used to penalize the degree of freedom of the model and leads to the following optimization problem:

$$\arg\min_\theta L(\theta) + \mu|\theta|^* \tag{20}$$

where $\theta = (\theta_1, \ldots, \theta_K)$, μ is a tuning parameter and $|\theta|^*$ is the nuclear norm of the matrix θ [47]. The authors of LRAcluster implement a fast low-rank approximation described in [34]. Then, they use a singular vector decomposition (SVD) of θ, and then, to identify clusters, k-means or hierarchical clustering can be used on the reduced low-dimensional space given by the SVD. An R package is available at http://bioinfo.au.tsinghua.edu.cn/member/jgu/lracluster/.

Results

In this section, we compared the 13 unsupervised methods described in the previous section. First, we simulated heterogeneous datasets to evaluate the classification into subgroups and we also evaluated the performance in terms of detection of the variables that can drive subgroups. However, simulations cannot illustrate perfectly the biological mechanisms with true interactions. For this reason, we also evaluate the performance of methods for three real studies [11, 48, 49]. These three studies contain multiple types of omics datasets, including proteomics, metabolomics, transcriptomics, epigenomics and genomics (somatic mutations, and copy number variations). Different phenotypical outcomes have been studied including the type of physical exercise, the diet and the subtype of cancer. The subgroups that compose these three studies are also different. For instance, two studies contain two balanced subgroups with a limited number of samples, whereas the third study is composed of four subgroups with a larger number of samples (see Application to three real studies section).

We evaluate the performance of all the methods at multiple levels: the computing time in Computing time section, the ability of each method to classify the individuals into the true subgroups (Clustering evaluation section for simulated data and Clustering evaluation section for real datasets) and the ability to find the relevant variables in each dataset that drive the multiple subgroups of patients (Variable importance evaluation section for simulated data and Analysis of variable selection section for real data).

Note that MOFA has difficulties to converge on all simulation benchmarks and it does not find the simulated structures even on easy benchmarks. For this reason, the results do not appear in the Results section. Furthermore, we found that the sparsity parameters are not easy to tune.

Simulations

We simulated datasets from three kinds of distributions (Gaussian, Beta-like and Binary). These distributions simulate the heterogeneity of the various multi-omics datasets. For instance, the Gaussian distribution could represent abundance quantification such as metabolomic, proteomic and transcriptomic data. Then, the Beta-like distribution could simulate the percentage of DNA methylation. Finally, the Binary distribution could represent the presence/absence of a mutation in a gene.

We present below the procedure to generate simulated datasets.

i. Define the total number of individuals (n)
ii. Define the group of each individual (with n_l the number of individuals in subgroup l)
iii. Define the number of datasets and their size (K and J_k)
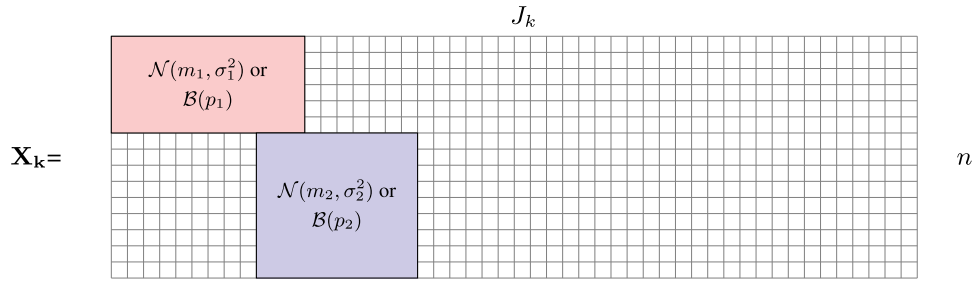iv. Choose the type of omics of each dataset (Gaussian, Beta-like, Binary)

Figure 1. Illustration of the simulated matrix generated at step 1. The red and blue boxes follow either Gaussian distribution or a Binary distribution, the white squares represent zero values.

   v. Tune parameters for each type of dataset (mean and standard deviation for Gaussian and Beta-like, proportions for Binary)
   vi. Define the percentage of relevant variables that drive subgroups
  vii. Set background noise
 viii. Finally, we obtain K matrices $\mathbf{X}_1, \ldots,$ and $\mathbf{X}_K$

To simulate matrices we perform the following steps.

Step 1: We consider two groups of size C_1 and C_2 such as $C_1 + C_2 = n$. Then, we simulate the red and blue sub-matrices of sizes $C_1 \times r_1$ and $C_2 \times r_2$, where r_1 and r_2 are the numbers of relevant variables (Figure 1). The intersection between the two groups of relevant variables can be non-empty.

- Gaussian: the two sub-matrices follow $\mathcal{N}(m_1, \sigma_1^2)$ and $\mathcal{N}(m_2, \sigma_2^2)$.
- Binary: the two sub-matrices follow $\mathcal{B}(p_1)$ and $\mathcal{B}(p_2)$.
- Beta-like: the two sub-matrices follow $\mathcal{N}(m_1, \sigma_1^2)$ and $\mathcal{N}(m_2, \sigma_2^2)$ with m_1 negative and m_2 positive.

The non-relevant variables for each group are set to zero.

Step 2: Noise is simulated by different ways according to the type of data that the user wants to simulate.

- Gaussian: a matrix of size $n \times J_k$ following $\mathcal{N}(0, \sigma^2)$ is added to $\mathbf{X}_k$.
- Binary: a matrix of size $n \times J_k$ following $\mathcal{B}(p)$.
- Beta-like: a matrix of size $n \times J_k$ following $\mathcal{N}(0, \sigma^2)$ is added to $\mathbf{X}_k$.

Step 3: For Beta-like the transformation $f(x) = 1/(1 + exp(-x))$ is used to obtain a distribution between 0 and 1.

For more details on the simulations, see Section A1.1 in the appendix.

Computing time

The first question that we address is the computing time of each method to analyze datasets with a large number of individuals. For example, TCGA datasets often contain more than 300 individuals per cancer type. Additionally, as we related in the introduction (Introduction section), the dimension of omics datasets could be very high. Thus, we decided to compare the computing time of the methods when the number of individuals increases.

Let us denote by $\mathbf{X}_k$ (of size $n \times p_k$) the dataset simulated with the procedure described above. We compare the different methods at the computing time level using three heterogeneous datasets ($\mathbf{X}_1$, $\mathbf{X}_2$, and $\mathbf{X}_3$) where $J_1 = 1000$, $J_2 = 500$ and $J_3 = 5000$. The total number of individuals varies in a fixed grid ($n = 60$, 120, 180 and 240). The number of individuals could be considered as small, but the collection of multiple omics data for the same patient is expensive and limited by the quantity of biological materials.

Figure 2 shows that all methods have computing time between few seconds to few minutes per run. The fastest method is SNF (less than one second for all values of *n*) due to the computation of only one SM (Methods section). The three slowest methods are iClusterPlus, CIMLR and intNMF (Figure 2). For iClusterPlus, this is due to the high complexity of the Monte Carlo Newton–Raphson Algorithm for maximizing a penalized log-likelihood (equation 10). For intNMF, the authors use a derived non-negativity constrained alternating least squares algorithm that remains slow. CIMLR is also slow probably due to the large number of kernels that are computed an by the optimization of the equation 15 (CIMLR section). Nevertheless, there is no parameter to optimize for this method unlike intNMF and iClusterPlus. Besides, for SGCCA, PINSPlus and ConsensusClustering methods, the computing times slightly increase when the number of individuals raises. Therefore, analyses of a large number of individuals could be performed with these methods.

Influence of parameter tuning

All methods have parameters that are necessary to tune (Methods section). Indeed, MoCluster, SGCCA, RGCCA, intNMF and MCIA require to tune the number of latent profiles. Then, SGCCA, MoCluster and iClusterPlus need to tune the sparsity parameters. Finally, RGCCA involves to tune the shrinkage parameter τ (Methods section). The results of this part are in the supplementary materials S1. The influence of the number of latent variables d, the shrinkage parameter τ and the sparsity parameters have been studied through the Adjusted Rand Index (ARI) and the receiver operating characteristic (ROC) curves.

Tuning parameters of methods could be time consuming. For this reason, we show the influence for sparsity parameter for MoCluster and SGCCA only. Indeed, Computing time section shows that iClusterPlus is one of the slowest methods and it has a lot of parameters to tune (Methods section). Consequently, the investigation of the influence of its parameters was not performed.

A too small or too high value of d decreases the ability to recover the relevant variables and the true clustering. For a too high sparsity, a large number of relevant variables is missed, and for a too low sparsity the false positive rate (FPR) increases. Furthermore, ARI is not always optimal when the sparsity is low.

Then, in the vignette, we evaluate the performance of RGCCA with various values of τ. We demonstrate that on the simulated data τ does not influence the results.

In the following, for all the methods based on latent variables, we fix this number to the true number of simulated clusters. For sparse methods, we show the results with the sparsity parameters that give the best results for the ARIs and we fixed τ to 1 (the default parameter).
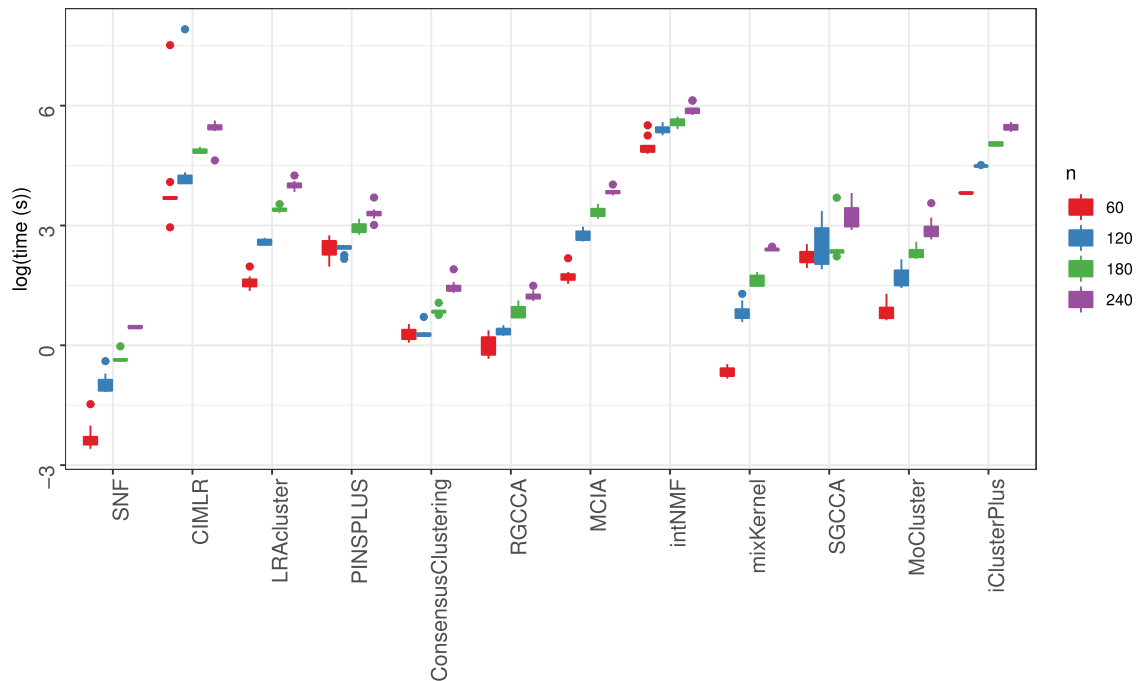
Figure 2. Computing time comparison. Each method is run 10 times for 60, 120, 180 and 240 samples in the datasets. Methods are in x-axis and time in logarithm of (seconds) is in y-axis.

Simulation scenarios

We simulate eight benchmarks with multiple parameters. The eight benchmarks have features described in Table 2. For benchmarks 1 to 4, only the signal to noise ratio changes from one to another. Then, only the proportion of relevant variables has changed between benchmarks 5 and 1. Finally, the number of subgroups changes from one to another from benchmarks 6 to 8. All benchmarks have been simulated 50 times. These benchmarks will enable us to highlight the cases where some methods fail.

Choice of the number of clusters

On benchmark 1, we evaluate the ability of methods to recover the true number of clusters (Section A1.2 in the appendix). In this benchmark, we simulate 4 unbalanced groups (see Simulation scenarios section in Table 2). Computing time section shows that iClusterPlus and intNMF are the two slowest methods. Moreover, these methods require to be run once for each C (the number of clusters). For this reason, we limited the performance evaluation of the choice of C to a subset of simulated datasets.

In Figure 3, LRAcluster, MoCluster and SGCCA are the best methods to select the true C with a median equal to 4. PINSPLUS recover between 2 and 4 clusters, while RGCCA select between 3 and 5 clusters, and iClusterPlus between 2 and 5 clusters. Then, SNF, RGCCA, intNMF and mixKernel recover only 2 on 3 clusters. Finally, the three worst methods to recover C are ConsensusClustering, CIMLR and MCIA (indeed C is overestimated for the first two methods, while it is really underestimated for the last one).

In the following, we used the simulated number of clusters to compare the classifications given by the methods.

Clustering evaluation

In this section, the ability of all the methods to classify the samples in the correct subgroups was compared (Figure 4).

To evaluate the performance of each method to classify the samples into subgroups, the ARI is computed [50]. The ARI is a score that measures the similarity between two clusterings. Here, the ARI is computed between the simulated clustering and the clustering given by the methods. Seven methods (SNF, intNMF, MoCluster, ConsensusClustering, CIMLR, PINSPLUS and iClusterPlus) provide their own classification.

For the other methods (LRAcluster, mixKernel, SGCCA, RGCCA and MCIA), we performed a usual hierarchical clustering using the Euclidean distance associated with Ward's method [51] on the individual reduced space or the SM. As the number of subgroups of simulated framework is known, therefore we cut the tree from the hierarchical clustering at the true number of subgroups. For SNF, intNMF, MoCluster, ConsensusClustering, PINSPLUS, CIMLR and iClusterPlus, we give the true number of subgroups as well.

Influence of the level of noise (Figure 4)

B1, B2, B3 and B4 benchmarks were used to evaluate the influence of the level of noise of datasets. The most consistent methods are SNF, MoCluster, intNMF PINSPLUS, ConsensusClustering, LRAcluster, RGCCA and mixKernel with an average ARI larger than 0.80 for the four benchmarks. For these benchmarks, CIMLR and iClusterPlus have difficulties reaching the maximum value of ARI. MCIA and SGCCA are the methods with the greater variability.

Influence of the number of subgroups (Figure 4)

To evaluate the influence of the number of subgroups in the whole dataset, B6, B7 and B8 benchmarks were compared. These benchmarks contain respectively 2, 3 and 4 subgroups of 25 individuals.

The methods that are not sensitive to the variation of the number of subgroups are SNF, RGCCA, PINSPLUS, Consensus-Clustering, LRAcluster, MoCluster, intNMF and SGCCA (the ARIs do not change a lot). Then, when the dataset is composed of only two subgroups, mixKernel, iClusterPlus and MCIA have stability issues (huge variability of ARI).
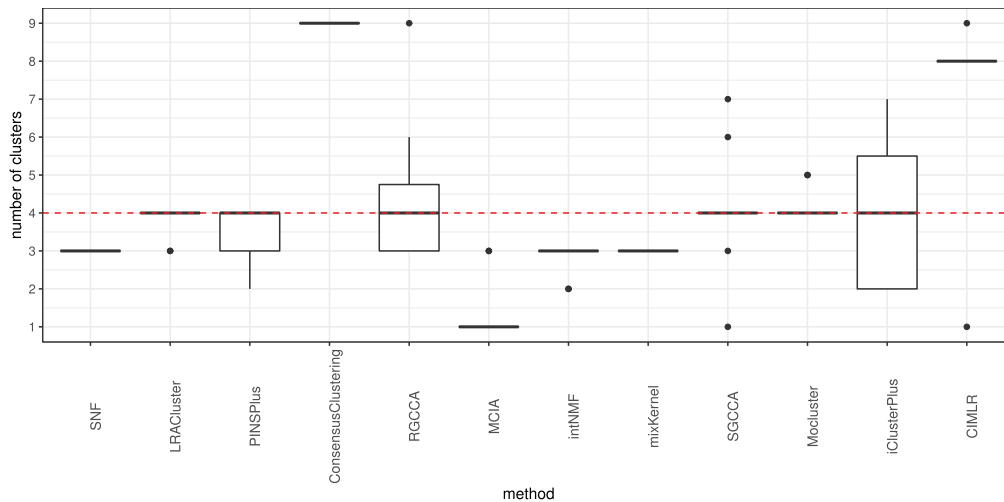
Figure 3. Estimation of the C_{best} by the integrative methods over 10 simulations. The red dashed line is the true C in the simulated datasets from benchmark 1.

Table 2. Characteristics of simulated benchmarks: type of dataset, noise (standard deviation of Gaussian and Beta-like distributions and proportion for binary distribution) and percentage of relevant variables for each dataset and number of samples per subgroup. All Gaussian datasets contain 1000 variables with mean = 2 and standard deviation = 1 for features that discriminate the subgroups. All binary datasets contain 500 variables and the parameter p to discriminate subgroups is equal to 0.6. All Beta-like datasets contain 5000 variables and the parameters to discriminate subgroups are respectively equal to (-2 and 2) for means and 0.5 for the standard deviation

Benchmark	Type Data Set	Noise	Relevant variables	Number per subgroup
B1	Gaussian	0.2	0.5%	10, 20, 5, 25
	Binary	0.01	1%	
	Beta-like	0.3	2%	
B2	Gaussian	**0.5**	0.5%	10, 20, 5, 25
	Binary	0.01	1%	
	Beta-like	0.3	2%	
B3	Gaussian	**0.1**	0.5%	10, 20, 5, 25
	Binary	**0.05**	1%	
	Beta-like	**0.3**	2%	
B4	Gaussian	0.2	0.5%	10, 20, 5, 25
	Binary	**0.02**	1%	
	Beta-like	**1.5**	2%	
B5	Gaussian	0.2	**1%**	10, 20, 5, 25
	Binary	0.01	**2%**	
	Beta-like	0.3	**4%**	
B6	Gaussian	0.2	0.5%	**25, 25**
	Binary	0.01	1%	
	Beta-like	0.3	2%	
B7	Gaussian	0.2	0.5%	**25, 25, 25**
	Binary	0.01	1%	
	Beta-like	0.3	2%	
B8	Gaussian	0.2	0.5%	**25, 25, 25, 25**
	Binary	0.01	1%	
	Beta-like	0.3	2%	

Influence of the composition of subgroups (Figure 4)

The influence of the relative number of individuals per subgroup is evaluated using benchmarks B1 (10, 20, 5 and 25 per subgroup) and B8 (25 per subgroup).

SNF, ConsensusClustering, LRAcluster and MoCluster have high ARIs in both cases. The performance of MCIA seems to be not degraded between the two cases. Then, iClusterPlus, RGCCA, CIMLR, PINSPLUS and SGCCA perform slightly worse for unbalanced groups. The smallest subgroup could be not well recovered by these four methods.

Influence of the proportion of relevant variables (Figure 4)

Benchmarks B1 and B5 evaluated the influence of the proportion of discriminant variables.

For all the methods, except MCIA and CIMLR, a higher proportion of discriminant variables in the datasets increases the performance to recover the subgroups. Indeed, MCIA and CIMLR do not recover the true subgroups because the ARIs do not reach 1.

Influence of the combination of blocks

We evaluate the influence of the various possible combinations of blocks using 10 simulated datasets from benchmark 1. The performance is evaluated using the ARI.

Figure 5 shows that, for most methods, the combination that provides the best ARI is the one with the three blocks. SGCCA has greater variability with the three blocks, and the best combination is when the block 2 (binary distribution) is removed. Furthermore, the combination with block 1 and 2 fails to find the true classification. MCIA, the ARI without the Gaussian block in the analysis, is also really low (close to 0).

To summarize, MoCluster, LRAcluster, ConsensusClustering, SNF and intNMF seem to be the best methods for clustering with the highest ARIs (close to 1). CIMLR and MCIA are the two worse methods on these simulations (ARI around 0.75). mixKernel performs well when the number of subgroups is greater than 2. SGCCA and RGCCA succeed well when the subgroups of individuals are balanced.

Variable importance evaluation

ROC curves have been used to evaluate the performance in terms of variable importance estimation. The results are summarized by computing the area under the curve (AUC) for each dataset of each benchmark (Table 3). AUC have been computed for the four methods that perform variable selection (SGCCA, CIMLR, MoCluster and iCluster) and for three others that do not perform variable selection (RGCCA, MCIA and intNMF) (Table 1). See Section A1.3 in the appendix to have a complete description of
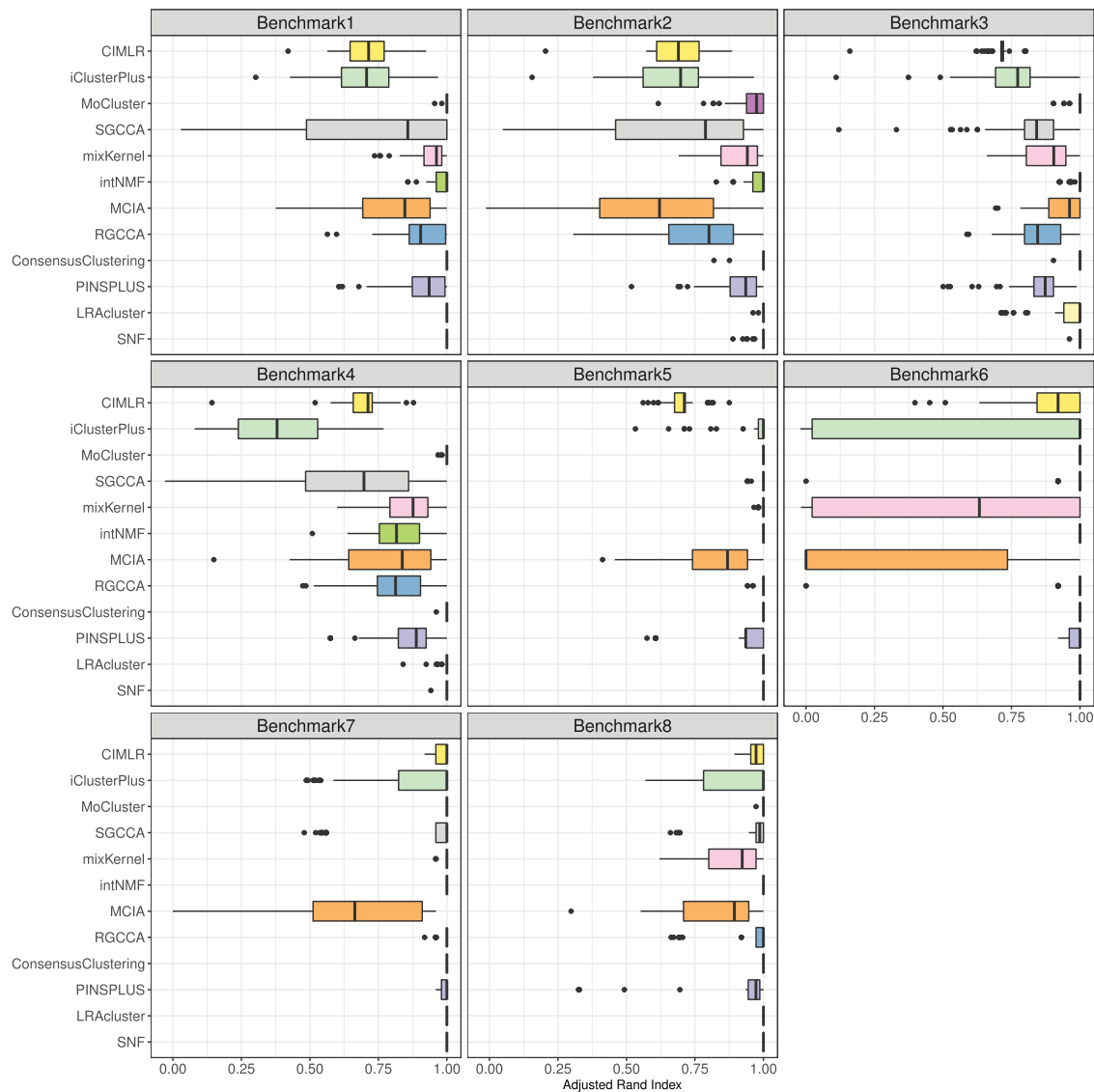
Figure 4. Performance evaluation of clustering. Boxplot of ARI results for each method and all benchmarks. Each boxplot contains 50 ARIs (number of datasets simulated under the parameters of benchmarks 1 to 8). The ARI between two classifications is equal to 1 if the groups are the same, and can be negative if the groups are completely incoherent. B1-B2-B3-B4: Influence of the level of noise. B6-B7-B8: Influence of the number of subgroups. B1-B8: Influence of the composition of subgroups. B1-B5: Influence of the proportion of relevant variables.

this step. SNF, LRAcluster, PINSPlus, ConsensusClustering and mixKernel only compute SM and the information about variable importance is not available.

For the dataset 1 (Gaussian distribution), the seven methods performed well (AUC≥ 0.98 for all benchmarks). MCIA does not perform well with binary data (AUC = 0.28). Even if iClusterPlus deals with binary data with a specific model, it seems that it has difficulties to recover the signal in this dataset (AUC = 0.48). intNMF has difficulties with the data under a Beta-like distribution (AUC = 0.28) but performs very well on the other types of data with AUC ≥ 0.99 (probably because the simulation of this dataset cannot be written as an MF problem see Methods section).

To conclude this section on variable importance evaluation, MoCluster outperforms the other with an AUC ≥ 0.99 in average. Then RGCCA, CIMLR and SGCCA perform well (AUC ≥0.95 on average on the three types of datasets).

Stability of variable selection

Only four methods perform a variable selection: MoCluster, SGCCA, CIMLR and iClusterPlus (Methods section). For these methods, we also investigated the stability of the variable selection using a jackknife-like procedure (Section A1.4 in appendix).

As the computation of the jackknife-like procedure is time consuming, we only show the results for the benchmarks 1 to 4 and for only one simulation in Figure 6 using violin plots. The initial number of selected variables is shown in Table 4.

Across the four benchmarks, the distributions of the selection frequency have the same form across Gaussian and Beta-like datasets for SGCCA, CIMLR and MoCluster (Figure 6). A non-negligible subset (around 50%) of variables is selected with a frequency higher than 80%. MoCluster has a better
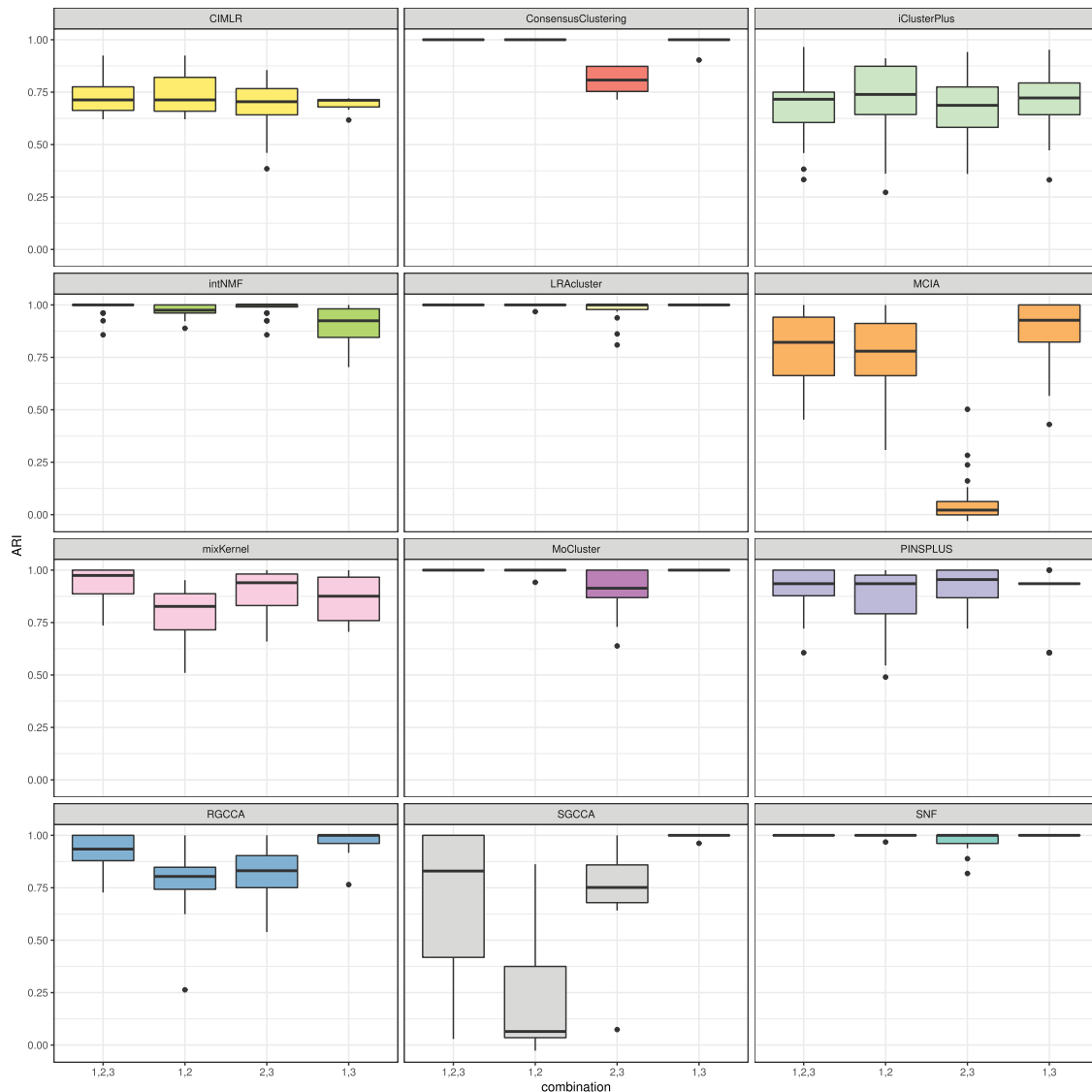
Figure 5. Influence on the clustering of the combination of the datasets on benchmark 1. The performance is measured with the ARI for CIMLR, ConsensusCLustering, iCLusterPlus, intNMF, LRACluster, MICA, mixKernel, MoCLsuter, PINSPLUS, RGCCA, SGCCA and SNF.

ability to deal with binary distribution with a majority (more than 80%) of variables selected with a frequency higher than 90%.

Finally, we assessed if the jackknife-like procedure allows decreasing the FPR for the same level of true positive rate (TPR). The original set of selected variables is pruned by keeping all the variables that have been selected in 80% of the runs in the jackknife-like procedure (results are shown in Figure 7). The pruning enables decreasing the FPR for all methods. Nevertheless the TPR of SGCCA falls to 0.60. For MoCluster, the TPR does not change (around 0.90). Therefore, the detection of variables is thus improved with the jackknife-like procedure. MoCluster globally outperforms the other methods after pruning (with a TPR ≥ 0.90 and an FPR ≤ 0.25).

The pruning does not globally improve the results of iClusterPlus. Indeed, this method provides a relatively low FPR (≤ 0.20) before the jackknife-like procedure with a relatively high TPR (around 0.70). Moreover, the TPR decreases a lot after the jackknife-like procedure. Therefore, this procedure may not be adapted for iClusterPlus.

CIMLR has already a low FPR ¡0.05 with a TPR greater than 0.5 in most cases before pruning. CIMLR is more conservative than the other ones. Therefore, pruning by the jackknife procedure does not improve the results of this method.

Application to three real studies

Even if simulations can help to understand the weaknesses of the methods, it is difficult to simulate the genomic mechanisms. For this reason, the methods were also compared using three real studies.

Three public studies with various features (types of omics data, types and number of samples, type of subgroups), murine liver (BXD) [48], Liver Cancer (Liver) [49] and obesity datasets [11], were used. Characteristics of studies are shown in Table 5.

Table 3. Means of AUCs computed for iClusterPlus, intNMF, MCIA, MoCluster, RGCCA, SGCCA and CIMLR (the seven methods with possible variable importance computations) by datasets and by benchmarks

Dataset	Benchmark	iClusterPlus	intNMF	MCIA	MoCluster	RGCCA	SGCCA	CIMLR
Dataset 1	Benchmark 1	1.00	1.00	1.00	1.00	0.98	0.99	0.99
(Gaussian)	Benchmark 2	0.96	0.99	1.00	1.00	0.96	0.97	0.97
	Benchmark 3	1.00	1.00	1.00	1.00	0.98	0.98	1.00
	Benchmark 4	0.99	1.00	1.00	1.00	0.97	0.96	0.99
	Benchmark 5	0.99	1.00	1.00	1.00	1.00	1.00	1.00
	Benchmark 6	0.97	1.00	1.00	1.00	0.99	1.00	1.00
	Benchmark 7	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Benchmark 8	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Average	**0.99**	**1.00**	**1.00**	**1.00**	0.98	**0.99**	**0.99**
Dataset 2	Benchmark 1	0.45	0.99	0.35	1.00	0.91	0.95	0.93
(Binary)	Benchmark 2	0.46	0.99	0.36	0.99	0.88	0.93	0.92
	Benchmark 3	0.52	0.98	0.16	0.99	0.88	0.91	0.91
	Benchmark 4	0.48	0.98	0.28	0.99	0.89	0.91	0.93
	Benchmark 5	0.45	0.99	0.31	1.00	0.96	0.96	0.95
	Benchmark 6	0.49	1.00	0.22	1.00	0.94	0.98	1.00
	Benchmark 7	0.49	1.00	0.29	1.00	0.97	0.98	1.00
	Benchmark 8	0.47	1.00	0.31	1.00	0.98	0.99	1.00
	Average	0.48	**0.99**	0.28	**1.00**	0.93	0.95	0.95
Dataset 3	Benchmark 1	0.84	0.29	0.85	0.97	0.91	0.94	0.93
(Beta-like)	Benchmark 2	0.83	0.29	0.86	0.96	0.90	0.94	0.92
	Benchmark 3	0.85	0.29	0.85	0.98	0.89	0.93	0.94
	Benchmark 4	0.69	0.32	0.76	0.87	0.79	0.82	0.87
	Benchmark 5	0.87	0.29	0.86	0.98	0.95	0.97	0.94
	Benchmark 6	0.85	0.25	0.81	0.98	0.99	0.98	1.00
	Benchmark 7	0.95	0.25	0.92	1.00	1.00	0.99	1.00
	Benchmark 8	0.98	0.26	0.96	1.00	1.00	0.99	1.00
	Average	0.86	0.28	0.86	**0.97**	0.93	0.95	0.95

Table 4. Initial number of selected variables for methods SGCCA, iClusterPlus, CIMLR and MoCluster, which perform variable selection on benchmarks 1 to 4 and for the three types of simulated datasets

Method	Dataset	Benchmark 1	Benchmark 2	Benchmark 3	Benchmark 4
MoCluster	Gaussian	150	163	146	154
	Binary	262	246	405	346
	Beta-Like	1664	1655	1639	1705
SGCCA	Gaussian	456	507	509	506
	Binary	128	142	239	197
	Beta-Like	3098	3355	3337	3211
iClusterPlus	Gaussian	250	250	71	250
	Binary	66	60	121	89
	Beta-Like	1250	1250	1250	1250
CIMLR	Gaussian	28	40	28	30
	Binary	15	17	13	18
	Beta-Like	295	403	304	240

Real datasets descriptions

BXD [48]. We analyzed a BXD cohort (composed of 64 samples) across a battery of metabolic tests such as *ad libitum* running-wheel access, maximal exercise capacity and glucose tolerance. The mice were tested over a 29-week program where they were exposed to different environmental conditions of diet: chow diet (CD) (6% kcal of fat) or high-fat diet (HFD) (60% kcal of fat). Measurements have been made in the livers of the entire population at the transcriptome, the proteome and the metabolome levels.

Obesity [11]. The second dataset concerns middle-aged Polynesian men and women with morbid obesity (44 kg/m2 ±10) suffering from type 2 diabetes. Measurements from biopsies of the vastus lateralis have been obtained via microarrays for RNA, DNA methylation and miRNA. Individuals are studied before and after 16 weeks of resistance (n = 7) or aerobic training (n = 6). Here, we first computed the match difference between the data after and before the training. Then, we compared genomic differences between aerobic (AER) and resistance (PRT) training.
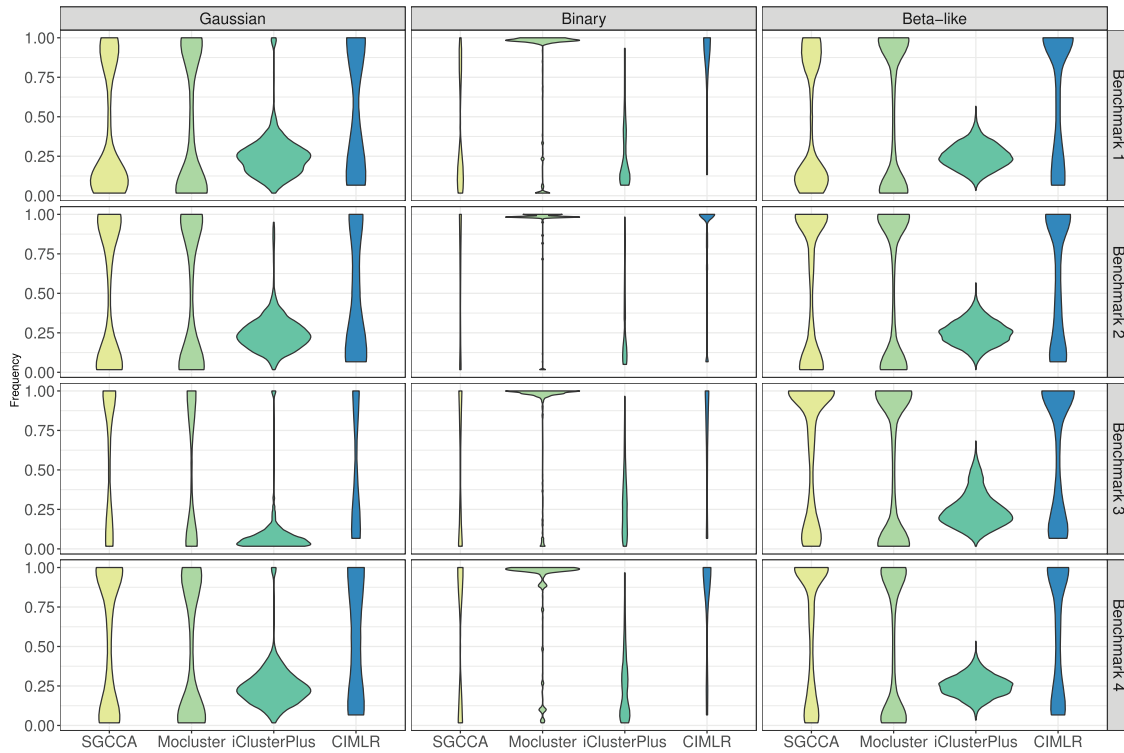
Figure 6. Jackknife results for SGCCA, MoCluster, iClusterPlus and CIMLR. Violin plots representing the distribution of the frequency of selection for each variable. The three types of datasets are shown: Gaussian, Binary and Beta-like. The frequency of selection for each variable on the *n* models was calculated. Each violin-plot shows the density of the frequency of all the variables caught at least once for each method and each dataset. A variable caught by the *n* models produces a frequency equal to one.

Table 5. Overview of three real publicly available multi-Omics datasets. Name, phenotype, number of samples, sub-types of phenotype, OMICS datasets, size of dataset, tissue and reference.

Dataset	Phenotype	Number of samples	Subgroups	Omics	Number of variables	Type of tissue
BXD[48]	Mitochondrial metabolism	64	High-fat and Chow diet	Transcriptomic	35 554	Liver
				Proteomic	21 547	
				Metabolomic	956	
Obesity[11]	Obesity and Type 2 diabete	13	Endurance	Epigenomic	468 397	Skeletal muscle
			Resistance	Transcriptomic	43 961	
				miRNA	20 248	
Liver[49]	Liver Cancer	360	Various clinical variables (type,grade, survival)	Epigenomic	18 163	Tumoral
				Transcriptomic	19 922	
				CNV	24 776	
				Mutation	5530	

Liver Cancer [*49*]. In addition, we selected a dataset from TCGA with CNV, mutation, DNA methylation, RNA datasets from liver tumors, and we tried to discriminate the stages (I to IV) of disease.

Clustering evaluation

For all real studies, the first objective is to classify the samples into subgroups by phenotypes (Table 6). We compute the F-measure in addition to ARI to evaluate the clustering obtained

by the methods. Only the results for optimized parameters are shown. The optimal number of latent variables, the sparsity parameters and the number of datasets have been tuned manually to reach the optimal ARI. For example, in BXD study, for intNMF and mixKernel, the performance is optimal when only two types of datasets are used (here the RNAseq dataset and the metabolite dataset).

BXD. For the BXD study, there are two groups: the CD and the HFD. F-scores vary between 0.57 (for MCIA) and 1 (for MoClus-
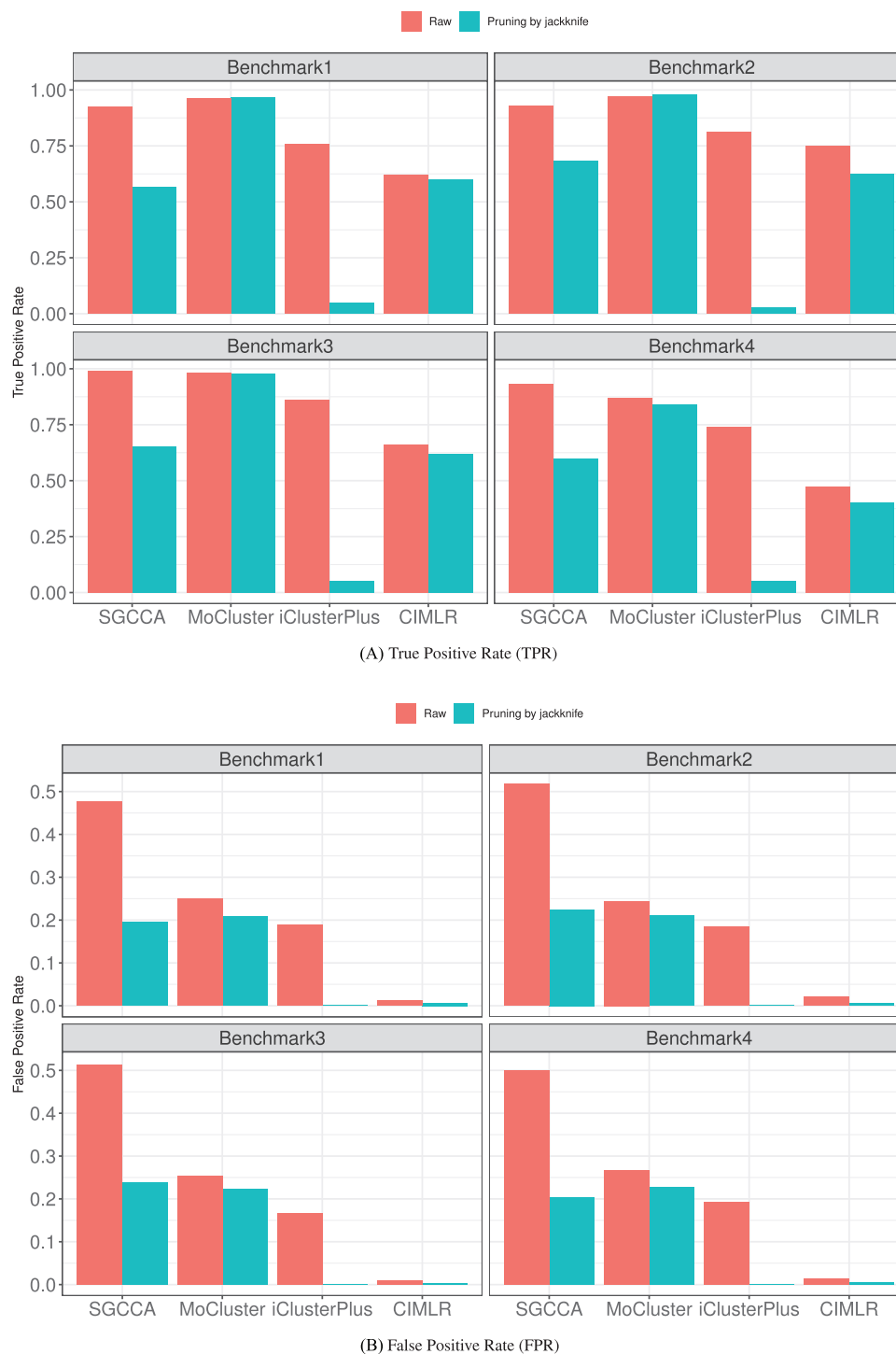
FIGURE 7. Results of pruning of the variable selection set using jackknife-like procedure. (A) TPR and (B) FPR are computed without jackknife (red) and then after pruning using the jackknife procedure (blue) for benchmarks 1 to 4 on the following methods: SGCCA, MoCluster and iClusterPlus. benchmarks 1 to 4 are different in terms of noise.

ter, RGCCA, CIMLR and SNF). ARIs range from 0 (for MCIA and mixKernel) up to 1 (for MoCluster, RGCCA, CIMLR and SNF).

Obesity. For the obesity study, individuals can be divided into two subgroups: the AER and the PRT. The AER and the PRT groups contain six and seven individuals, respectively. ARIs are not very high and mainly negative (Table 6). F-score is between 0.63 and

0.77. The three best methods are MoCluster, LRAcluster and SNF with F-score ≥ 0.70 and non-negative ARIs.

Liver cancer. The clustering obtained by the methods and the pathological stage of liver cancer has been compared. Each stage contains 133, 69, 55 and 9 patients, respectively. The clustering obtained by the integration of all the datasets does not reflect

Table 6. Performance evaluation of clustering using ARI and F-measure on real data.

Datasets	Methods	F-score	ARI	Number of datasets used
BXD	iClusterPlus	0.97	0.88	3/3
	mixKernel	0.59	0.00	2/3
	MCIA	0.59	0.00	3/3
	MoCluster	1.00	1.00	3/3
	intNMF	0.98	0.94	2/3
	RGCCA	1.00	1.00	3/3
	SGCCA	0.95	0.82	3/3
	PINSPLUS	0.79	0.63	3/3
	CIMLR	1.00	1.00	3/3
	ConsensusClustering	0.92	0.71	2/3
	LRAcluster	0.72	0.18	3/3
	SNF	1.00	1.00	3/3
Obesity	iClusterPlus	0.65	-0.01	3/3
	mixKernel	0.63	-0.02	3/3
	MCIA	0.63	-0.02	3/3
	MoCluster	0.70	0.08	3/3
	intNMF	0.63	-0.02	3/3
	RGCCA	0.63	-0.02	3/3
	SGCCA	0.63	-0.02	3/3
	PINSPLUS	0.54	-0.08	3/3
	CIMLR	0.54	-0.08	3/3
	ConsensusClustering	0.63	-0.02	3/3
	LRAcluster	0.70	0.03	3/3
	SNF	0.77	0.23	3/3
Liver cancer	iClusterPlus	0.35	0.01	4/4
	mixKernel	0.37	-0.01	3/4
	MCIA	0.35	0.00	4/4
	MoCluster	0.66	0.01	4/4
	intNMF	0.40	-0.01	2/4
	RGCCA	0.42	0.03	4/4
	SGCCA	0.65	-0.00	4/4
	PINSPLUS	0.35	0.01	4/4
	CIMLR	0.35	0.00	4/4
	ConsensusClustering	0.38	-0.00	4/4
	LRAcluster	0.38	0.02	4/4
	SNF	0.50	0.01	4/4

Table 7. P-values for survival analysis and composition of subgroups for each method on liver cancer study

Methods	P-Value	Number by Cluster
SGCCA	0.59	239,24,1,1,1
RGCCA	0.07	35,29,57,64,23,16,19,10,12,1
MCIA	**0.01**	157,109
SNF	0.57	127,139
mixKernel	0.64	105,72,89
MoCluster	**0.02**	223,38,1,1,1,1,1
iClusterPlus	0.32	58,64,69,75
intNMF	0.75	102,90,74
CIMLR	**0.02**	36,82,84,64
LRAcluster	0.19	160,106
PINSPLUS	0.34	55,59,57,95
ConsensusClustering	0.38	121,30,75,40

The diet of the mice influences the metabolome and the methods reveal abundance differences in metabolites linked to high-fat diet $C_{29}H_{50}O_2$ (vitamin E), $C_{36}H_62O_5$ (cholesteryl) and $C_{35}H_{66}O_5$ (lipid), C_4H_5NS (mustard oil). For example, cholesteryl ester is a dietary lipid [52], and vitamin E is a group of eight fat soluble compounds [53]. However, the top 10 of relevant variables are not the same for the four methods. For both metabolomic and proteomic datasets only three and two variables from the top 10 variables are common between the four methods. For the transcriptomic data, MoCluster and iClusterPlus have nine common variables. SGCCA and CIMLR have six common variables.

The proteomic dataset does not reveal differences between chow and high-fat diets. Only the protein 9130231C15Rik; AI266984; Ces6 seems having differential abundance between the two groups. This protein is involved in the lipid catabolic process.

Finally, the analysis of the transcriptome reveals some patterns in terms of differential expression linked to the two groups of mice (Figure F10 in Appendix Section A1.6). We explore the functional annotation of the selected genes with genecards.com [54]. For examples, the Saa2 gene codes for a protein that is involved in the HDL complex. Cyp2b9 oxidizes steroids, fatty acids and xenobiotics. Finally, the Cidea gene is involved in metabolism of lipids and lipoproteins; the lipid digestion, mobilization and transport; and the differentiation of white and brown adipocyte.

To conclude this part, the analyses with the four methods that perform variable selection highlight various mechanisms linked to the diet of the mice.

For the obesity and liver cancer studies, the integrative analysis does not reveal any variable associated with phenotypes (type of physical exercise and stage of liver cancer). Results are not shown in this article but can be generated with the package.

the classification of the clinical data. Indeed, F-scores range from 0.35 to 0.66, and ARIs range from -0.01 to 0.03.

In summary, there is no correlation between available clinical variables and the subgroups discovered by the various methods. Therefore, we compare the survival curves between groups. The *P*-values have been computed for each method after removing clusters composed of a single sample. Associations between subgroups and outcome were calculated by Kaplan–Meier analysis using a log-rank test. Only three methods (MCIA, MoCluster and CIMLR) provide slightly significantly different survival curves between discovered groups (*P*-values lower than 5e-2) (the results are in Table 7).

Summary of performance

The performances of each method are summarized in Table 8. The methods are not user-friendly to compare; therefore, we propose an R package named `CrIMMix` on GitHub available at https://github.com/CNRGH/crimmix, which allows running all methods by a simple command line and with a uniform output. We give an example of how to use this package in supplementary materials S2 7. The package `CrIMMix` also includes the scripts to perform the simulated data shown in this article as well as the three real studies.

Analysis of variable selection

For each of the four methods that propose variable selection (MoCluster, CIMLR, SGCCA and iClusterPlus) and for each dataset of the BXD study, we visualized the abundance of the top 10 variables (with the larger weights) for each mouse (Figures F8, F9 and F10). Two databases are used: https://www.genecards.org/ and https://pubchem.ncbi.nlm.nih.gov. Several variables have clearly differential abundance (Figure F8).

Table 8. Summary of performance. Good performances are denoted by +++ and bad performances by -. NA means not available

Name	Type	Clustering	Variable Detection (simulations)			Time	User-friendly	Number of clusters
			Gaussian	Binary	Beta-like			
RGCCA	(CCA)	+	+	+	+	++	+	+
SGCCA	(CCA)	++	+++	++	++	++	+	+++
MCIA	(CCA-like)	+	+++	-	++	++	+++	-
intNMF	(MF)	++	+++	+++	-	-	+++	++
iClusterPlus	(MF)	++	+++	-	++	-	-	+++
MoCluster	(MF)	+++	+++	+++	+++	++	++	+++
mixKernel	(SM)	++	NA	NA	NA	++	+++	++
SNF	(SM)	+++	NA	NA	NA	+++	+++	++
CIMLR	(SM)	++	+++	++	++	++	+++	-
LRACluster	(SM)	+++	NA	NA	NA	++	+++	+++
ConsensusClustering	(SM)	+++	NA	NA	NA	++	+++	-
PINSPlus	(SM)	+++	NA	NA	NA	++	+++	+++

Discussion

This article is focused on the comparison of 13 unsupervised integrative methods. Indeed, supervised methods only enable the identification of candidate biomarkers [21] for a known phenotype, whereas unsupervised methods aim to both discover new subgroups and potentially to detect relevant variables. Recently, two reviews on the clustering performance evaluation of unsupervised integrative methods were published [23, 26]. The first article [23] compared five methods including MCIA [30], SNF [18], multiple factor analysis (MFA) [55], joint and individual variation explained (JIVE) [56] and multiple canonical correlation analysis [57]. In the second article [26], the authors compared iCluster [38], MoCluster [39], JIVE [56], multiple dataset integration [58], integrative NMF from [59] and Bayesian consensus clustering [60].

In our article, 13 methods including SNF, RGCCA, MCIA, intNMF, mixKernel, SGCCA, MoCluster, iClusterPlus, CIMLR, LRAcluster, PINSPLUS, ConsensusClustering and MOFA were compared on their ability to perform clustering and recover relevant variables.

MOFA does not give any results on our simulation frameworks. The model does not converge and tuning the parameters is complicated. Therefore, the performance has not been shown.

Furthermore, in our article, the simulation framework is extended by simulating heterogeneous distributions of data (Gaussian, Binary and Beta-like). Indeed, both reviews used Gaussian frameworks. Authors of [23] simulate data using Gaussian distribution based on means and standard deviations from real gene expression of breast cancer data. Authors of [26] proposed two models to simulate multi-omics data, and the two models are under Gaussian distributions. The first model is based on NMF i.e. simulated matrices are non-negative and can be written as an MF problem. This simulated framework gives the advantage to the methods that use this type of model (iClusterPlus, MoCluster and intNMF). The second model is also based on the NMF model but the authors use multi-dimensional Gaussian distributions.

In our simulation framework, we did not simulate the data under a particular model, in this way no method is advantaged. In the simulation benchmarks, the heterogeneity of the datasets enables us to conclude that methods have not the same performance depending on the type of data. We notice that iClusterPlus and MCIA fail to recover the relevant variables on binary data (Table 3). intNMF fails on the Beta-like distribution, probably because the data cannot be expressed as an MF problem. Then, RGCCA, SGCCA, CIMLR and MoCluster methods are consistent across the three types of simulated data. MoCluster outperforms the other methods in terms of clustering across the simulations in our article and in [26]. In this work, the results for SNF and MCIA are also coherent with the previous review [23] in which SNF is the best method.

These two articles apply the different methods on real studies. Tiny *et al.* used the data from three studies: the BXD study [48], a platelet reactivity study [44] and a breast cancer study [61]. Chauvel *et al.* and Tiny *et al.* compared methods on the same breast cancer study [61] and they found similar results. Here, we applied all the methods on two original studies (obesity and liver cancer) and, as in [23], on the BXD study. The first conclusion was that when the datasets contained enough signal, many methods performed well (Table 6) and could identify the true clustering and relevant variables (BXD study).

However, the application to real studies also highlights the difficulties to integrate multi-omics. In the case of the obesity study, it seems that the high number of variables of each dataset and the small number of individuals (6 and 7) could be the reason why all the methods fail to identify the clusters. Furthermore, the types of omics data (epigenomic and transcriptomic data) are probably not sufficient to reveal the effect of the kind of physical exercise. Maybe lipidomic or metabolomic data and a larger number of patients could be more relevant.

Although there are around 300 patients in the liver cancer study, the absence of relevant results is probably due to the type of genomic data that do not reflect the stage of the tumor in this context. The results of clustering on liver cancer do not reach high ARIs ($\leq$0.70). Survival analysis has been performed on liver cancer data and only the clusterings obtained by three methods (MoCluster, MCIA and CIMLR) give slightly significantly distinct survival curves between subgroups (Table 7). The integrative methods for this study provide different results. As the methods are different, they probably catch various kinds of signal from the data. The divergence of the clusterings could be explained by the heterogeneity of tumoral tissues (clonality or presence of normal cells).

The classification of real data thus depends on the number of patients, the type of samples (type of tissues or sorted cells), the sample preparation, the type of omics data and the biological question. Moreover, environmental parameters including environment, health and lifestyle could also influence the results of

multi-omics studies. To address a biological question, one needs to carefully determine these parameters quoted above. From real studies, we also show that using several datasets often increases the clustering performance. However, it is not always the best option when there is no signal or few individuals in the datasets.

A limitation of the simulated benchmarks that could also lead to different performances compared to real data is the independence of the simulated variables within and between the blocks. The independence could influence the performance of the methods that model the correlation or covariance between the blocks (RGCCA, SGCCA, MCIA and MoCluster). However, we observed that the relative performance of the methods is maintained between simulated and real data. The independence of variables in the simulated benchmarks does not seem to decrease the performance of the methods. In particular, with two balanced groups and similar sample sizes, for both the simulated benchmark 6 (heterogeneous data) and the BXD dataset (Gaussian distributed data), the ARI is high for MoCluster, RGCCA and SGCCA.

All the methods have several parameters to tune. The number of subgroups, the number of latent variables and for some methods the sparsity parameters have to be fixed by the user and their optimization is sometimes difficult. In particular, we encountered difficulties to tune the parameters of iClusterPlus. It seems that the sparsity parameters are not independent across the datasets. Indeed, modifying the value of sparsity parameter for one dataset also influences the number of selected variables of the other datasets. Therefore, it was a challenge to tune the parameters of iClusterPlus to get a similar number of selected variables compared to other methods. The authors of iClusterPlus advise performing cross-validation to select both the number of subgroups K and penalties but iClusterPlus is already one of the slowest methods and cross-validation would thus be very time-consuming (Figure 2).

The ability of the methods to choose the true number of subgroups had already been studied [26]. Here, we conclude that the best methods to detect the true number of clusters are LRAcluster, SGCCA, MoCluster and PINSPLUS (Figure 3). Chauvel *et al*. also concluded that MoCluster find properly the number of clusters, and that intNMF suffers from under-estimation as in this article. We fixed the number of latent variables at the true number of subgroups. Indeed, one latent variable may match one subgroup. In the case of an exploratory study, these two parameters could be optimized by a cross-validation procedure. For all the methods, clustering was performed and evaluated given the true number of subgroups.

We noticed that for most methods the clustering performance is higher when all omics blocks are used in the analysis, even though their performance is not strongly influenced by the combination of blocks when there is signal in all the blocks. However, MCIA fails when the Gaussian block is missing from the analysis of the simulated data (Figure 5), and SGCCA fails when the Beta-like block is missing from the analysis and is better when the binary block is not present.

In this article, we demonstrated that sparse methods could be the best approach for clustering and detection of relevant variables. For instance, the values of ARIs for SGCCA, which is the sparse version of RGCCA, are close to the ones obtained for RGCCA but SGCCA has the advantage to select relevant variables. MoCluster, which is a sparse method, outperforms the other methods in the eight simulated benchmarks and on the real studies. However, iClusterPlus seems to be too conservative and needs to have a greater proportion of relevant variables into the data to reach the level of other methods (Figure 4). Besides,

CIMLR, SGCCA and MoCluster provide the best values of AUCs on the simulations.

However, even if the TPR are high ($\geq$0.9) for SGCCA and MoCluster, the FPR are also relatively high ($\geq$0.2). Thus, we recommend a jackknife-like procedure to decrease the FPR. iClusterPlus and CIMLR are enough stringent with a relatively good balance between TPR and FPR (Figure 7).

In this work, we focused on the clustering performance and the quality of variable selection but further analyses could also be conducted. For instance on real datasets, performing pathway analysis after the feature selection is also advised. Indeed, combining various kinds of omics in a single integrative analysis could reveal interesting enriched pathways.

Conclusion

To summarize, our study aims to present and evaluate thirteen unsupervised integrative methods based on dimension reduction to integrate multi-omics datasets. There are classified in three different types of statistical methods: MF, SM and canonical correlation analysis.

In the second part, we compare the performance of the integrative methods at many levels. The first criterion is the computing time, which is a non-negligible feature as the size of datasets is growing up. Then, we compare the methods on multiple fair simulation benchmarks to evaluate clustering and variable detection performance. Finally, we apply them on real data.

This study concludes that SNF is the best method to perform classification of individuals. MoCluster is one of the best methods to perform classification as in [26] and can also reveal candidate biomarkers associated with the subgroups.

We conclude that the results of multi-omics studies depend on sample type, environmental parameters, multi-omics data and integrative methods.

In perspective, future unsupervised integrative methods need to be relatively fast and could be based on sparse latent variable models to perform variable selection to discover candidate biomarkers.

Furthermore, the tuning of parameters must to be more automated. In the context of precision medicine, the integration of multi-omics data should be performed using a complete workflow to ensure optimal data analysis.

Key Points

- Some methods have difficulties with the heterogeneity of datasets.
- To perform clustering, SNF outperforms the others.
- MoCluster well identifies subgroups and relevant variables.
- Methods with variable selection steps could reveal interesting features.

References

1. Ritchie MD, Holzinger ER, Li R, et al. Methods of integrating data to uncover genotype–phenotype interactions. *Nat Rev Genet* 2015;**16**(2):85.

2. Yugi K, Kubota H, Hatano A, et al. Trans-omics: how to reconstruct biochemical networks across multiple omic layers. *Trends Biotechnol* 2016;**34**(4):276–90.

3. Bock C, Farlik M, Sheffield NC. Multi-omics of single cells: strategies and applications. *Trends Biotechnol* 2016;**34**(8):605–8.

4. Chakraborty S, Hosen M, Ahmed M, et al. Onco-multi-omics approach: a new frontier in cancer research. *Biomed Res Int* 2018.

5. Hu Y, An Q, Sheu K, et al. Single cell multi-omics technology: methodology and application. *Front Cell Dev Biol* 2018;6.

6. Harber KJ, Verberk SG, Van den Bossche J. Going-omics to identify novel therapeutic targets for cardiovascular disease. *EBioMedicine* 2019;**41**:7–8.

7. Weinstein JN, Collisson EA, Mills GB, et al. The cancer genome atlas pan-cancer analysis project. *Nat Genet* 2013;**45**(10):1113–20.

8. Zhang J, Baran J, Cros A, et al. International cancer genome consortium data portal-a one-stop shop for cancer genomics data. *Database* 2011;**2011**.

9. Whiteaker JR, Halusa GN, Hoofnagle AN, et al. Cptac assay portal: a repository of targeted proteomic assays. *Nat Methods* 2014;**11**(7):703.

10. Hasin Y, Seldin M, Lusis A. Multi-omics approaches to disease. *Genome Biol* 2017. ISSN 1474-760X; **18**(1):83. doi: 10.1186/s13059-017-1215-1.URL. https://doi.org/10.1186/s13059-017-1215-1.

11. Rowlands DS, Page RA, Sukala WR, et al. Multi-omic integrated networks connect DNA methylation and miRNA with skeletal muscle plasticity to chronic exercise in type 2 diabetic obesity. *Physiol Genomics* 2014;**46**(20): 747–65.

12. Sun, Y. V. and Hu, Y.-J. Integrative analysis of multi-omics data for discovery and functional studies of complex human diseases. *Adv Genet*, 93, 147–90. Elsevier, 2016.

13. Töröcsik D, Weise C, Gericke J, et al. Transcriptomic and lipidomic profiling of eicosanoid/docosanoid signalling in affected and non-affected skin of human atopic dermatitis patients. *Exp Dermatol* 2019;**28**(2):177–89.

14. Zierer J, Menni C, Kastenmuller G, et al. Integration of 'omics' data in aging research: from biomarkers to systems biology. *Aging Cell* 2015;**14**(6):933–44.

15. Cavill R, Jennen D, Kleinjans J, et al. Transcriptomic and metabolomic data integration. *Brief Bioinform* 2015;**17**(5):891–901.

16. Cavill R, Sidhu JK, Kilarski W, et al. A combined metabonomic and transcriptomic approach to investigate metabolism during development in the chick chorioallantoic membrane. *J Proteome Res* 2010;**9**(6):3126–34.

17. Liu Y, Chen Y, Momin A, et al. Elevation of sulfatides in ovarian cancer: an integrated transcriptomic and lipidomic analysis including tissue-imaging mass spectrometry. *Mol Cancer* 2010;**9**(1):186.

18. Wang B, Mezlini AM, Demir F, et al. Similarity network fusion for aggregating data types on a genomic scale. *Nat Methods* 2014;**11**(3):333.

19. Burstein MD, Tsimelzon A, Poage GM, et al. Comprehensive genomic analysis identifies novel subtypes and targets of triple-negative breast cancer. *Clin Cancer Res* 2015;**21**(7):1688–98.

20. Palsson B, Zengler K. The challenges of integrating multi-omic data sets. *Nat Chem Biol* 2010;**6**(11):787–9.

21. Wu C, Zhou F, Ren J, et al. A selective review of multi-level omics data integration using variable selection. *High-throughput* 2019;**8**(1):4.

22. Meng C, Zeleznik OA, Thallinger GG, et al. Dimension reduction techniques for the integrative analysis of multi-omics data. *Brief Bioinform* 2016;**17**(4):628–41.

23. Tini G, Marchetti L, Priami C, et al. Multi-omics integration—a comparison of unsupervised clustering methodologies. *Brief Bioinform* 2017;**20**(4):1269–1279.

24. Bersanelli M, Mosca E, Remondini D, et al. Methods for the integration of multi-omics data: mathematical aspects. *BMC Bioinform* 2016;**17**(Suppl 2):15.

25. Huang S, Chaudhary K, Garmire LX. More is better: recent progress in multi-omics data integration methods. *Front Genet* 2017;**8**:84. doi: 10.3389/fgene.2017.00084.URL. https://www.frontiersin.org/article/10.3389/fgene.2017.00084.

26. Chauvel C, Novoloaca A, Veyre P, et al. Evaluation of integrative clustering methods for the analysis of multi-omics data. *Brief Bioinform* 2019. doi: 10.1093/bib/bbz015.

27. Tenenhaus A, Tenenhaus M. Regularized generalized canonical correlation analysis. *Psychometrika* 2011;**76**(2):257–84.

28. Tenenhaus A, Philippe C, Guillemot V, et al. Variable selection for generalized canonical correlation analysis. *Biostatistics* 2014;**15**(3):569–83.

29. Chalise P, Fridley BL. Integrative clustering of multi-level omic data based on non-negative matrix factorization algorithm. *PLoS One* 2017;**12**(5): e0176278.

30. Meng C, Kuster B, Culhane AC, et al. A multivariate approach to the integration of multi-omics datasets. *BMC Bioinform* 2014;**15**(1):162. doi: 10.1186/1471-2105-15-162.URL. https://doi.org/10.1186/1471-2105-15-162.

31. Mariette J, Villa-Vialaneix N. Unsupervised multiple kernel learning for heterogeneous data integration. *Bioinformatics* 2017;**34**(6):1009–15.

32. Ramazzotti D, Lal A, Wang B, et al. Multi-omic tumor data reveal diversity of molecular mechanisms that correlate with survival. *Nat Commun* 2018;**9**(1):4453.

33. Argelaguet R, Velten B, Arnol D, et al. Multi-omics factor analysis—a framework for unsupervised integration of multi-omics data sets. *Mol Syst Biol* 2018;**14**(6): e8124.

34. Wu D, Wang D, Zhang MQ, et al. Fast dimension reduction and integrative clustering of multi-omics data using low-rank approximation: application to cancer molecular classification. *BMC Genomics* 2015;**16**(1):1022.

35. Monti S, Tamayo P, Mesirov J, et al. Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Mach Learn* 2003;**52**(1–2):91–118.

36. Wilkerson MD, Hayes DN. Consensusclusterplus: a class discovery tool with confidence assessments and item tracking. *Bioinformatics* 2010;**26**(12):1572–3.

37. Nguyen T, Tagett R, Diaz D, et al. A novel approach for data integration and disease subtyping. *Genome Res* 2017;**27**(12):2025–39.

38. Mo Q, Wang S, Seshan VE, et al. Pattern discovery and cancer gene identification in integrated cancer genomic data. *Proc Natl Acad Sci U S A* 2013;**110**(11): 4245–50.

39. Meng C, Helm D, Frejno M, et al. Mocluster: identifying joint patterns across multiple omics data sets. *J Proteome Res* 2015;**15**(3):755–65.

40. Nguyen H, Shrestha S, Draghici S, et al. Pinsplus: a tool for tumor subtype discovery in integrated genomic data. *Bioinformatics* 2018;**35**(16):2843–2846.

41. Bailey P, Chang DK, Nones K, et al. Genomic analyses identify molecular subtypes of pancreatic cancer. *Nature* 2016;**531**(7592):47.

42. Shen R, Mo Q, Schultz N, et al. Integrative subtype discovery in glioblastoma using icluster. *PLoS One* 2012;**7**(4):e35236.

43. Hanafi M, Kohler A, Qannari E-M. Connections between multiple co-inertia analysis and consensus principal component analysis. *Chemom Intel Lab Syst* 2011;**106**(1):37–40.

44. Zhu B, Song N, Shen R, et al. Integrating clinical and multiple omics data for prognostic assessment across human cancers. *Sci Rep* 2017;**7**(1):16954.

45. Aronszajn N. Theory of reproducing kernels. *Trans Am Math Soc* 1950;**68**(3):337–404.

46. Wang B, Zhu J, Pierson E, et al. Visualization and analysis of single-cell rna-seq data by kernel-based similarity learning. *Nat Methods* 2017;**14**(4):414.

47. Candès EJ, Recht B. Exact matrix completion via convex optimization. *Found Comput Math* 2009;**9**(6):717.

48. Williams EG, Wu Y, Jha P, et al. Systems proteomics of liver mitochondria function. *Science* 2016;**352**(6291):aad0189.

49. Vasaikar SV, Straub P, Wang J, et al. Linkedomics: analyzing multi-omics data within and across 32 cancer types. *Nucleic Acids Res* 2017;**46**(D1):D956–63.

50. Rand WM. Objective criteria for the evaluation of clustering methods. *J Am Stat Assoc* 1971;**66**(336):846–50.

51. Ward JH, Jr. Hierarchical grouping to optimize an objective function. *J Am Stat Assoc* 1963;**58**(301):236–44.

52. Harvey RA, Ferrier DR, et al. *Lippincott's Illustrated Reviews: Biochemistry*, 2011. LWW; Sixth, North American edition (June 1, 2013).

53. Monsen ER. Dietary reference intakes for the antioxidant nutrients: vitamin C, vitamin E, selenium, and carotenoids. *J Acad Nutr Diet* 2000;**100**(6):637.

54. Stelzer G, Rosen N, Plaschkes I, et al. The genecards suite: from gene data mining to disease genome sequence analyses. *Curr Protoc Bioinformatics* 2016;**54**(1):1–30.

55. De Tayrac M, Lê S, Aubry M, et al. Simultaneous analysis of distinct omics data sets with integration of biological knowledge: multiple factor analysis approach. *BMC Genomics* 2009;**10**(1):32.

56. Lock EF, Hoadley KA, Marron JS, et al. Joint and individual variation explained (JIVE) for integrated analysis of multiple data types. *Ann Appl Stat* 2013;**7**(1):523.

57. Witten DM, Tibshirani RJ. Extensions of sparse canonical correlation analysis with applications to genomic data. *Stat Appl Genet Mol Biol* 2009;**8**(1):1–27.

58. Kirk P, Griffin JE, Savage RS, et al. Bayesian correlated clustering to integrate multiple datasets. *Bioinformatics* 2012;**28**(24):3290–7.

59. Yang Z, Michailidis G. A non-negative matrix factorization method for detecting modules in heterogeneous omics multi-modal data. *Bioinformatics* 2015;**32**(1):1–8.

60. Lock EF, Dunson DB. Bayesian consensus clustering. *Bioinformatics* 2013;**29**(20):2610–6.

61. Cancer Genome Atlas Network. Comprehensive molecular portraits of human breast tumours. *Nature* 2012;**490**(7418):61–70. doi: 10.1038/nature11412.

62. Charrad M, Ghazzali N, Boiteau V, et al. NbClust: an R package for determining the relevant number of clusters in a data set. *J Stat Softw* 2014;**61**(6):1–36. http://www.jstatsoft.org/v61/i06/.

63. Fawcett T. An introduction to ROC analysis. *Pattern Recognit Lett* 2006;**27**(8):861–74.

64. Bissell A. The jacknife. *J Appl Stat* 1977;**4**(1):55–64.

Appendix 1

A1.1 Simulated data

In this article, to simulate heterogeneous datasets we focus on three types of distributions.

To simulate the data, we first fix the number of subgroups (nclust), their sizes (n_byClust) and the number of variables J. Then, we fix the proportion of relevant variables that drive subgroups.

```
nclust <- 2
n_byClust <- c(10,10)
J <- 1000
prop <- 0.005
```

For the Gaussian distribution, we need to fix the means (m) and the standard deviations (sd) for the relevant variables. We draw randomly five different variables for each subgroup 1 and 2. The first five variables follow a $\mathcal{N}(m_1, sd_1^2)$ in the first subgroup, while the five last variables follow a $\mathcal{N}(m_2, sd_2^2)$ in the second subgroup.

```
means <- rep(2, nclust)
sds <- rep(1, nclust)
params_norms <- mapply(function (m, sd) {
    c(mean=m, sd=sd)}
    ,means, sds, SIMPLIFY=FALSE)
```

We build a matrix with these 10 relevant variables and full of zeros elsewhere. The final step is to set the background noise σ. For the Gaussian distribution we add a matrix centered and normally distributed $\mathcal{N}(0, \sigma^2)$ to the previous matrix.

```
sigma <- 0.2

dat1 <- simulateY(nclust=nclust,
            n_byClust=n_byClust, J=J,
            flavor="normal",
            prop=0.005,
            params=params_norms,
            noise=sigma)
```

For the binary distribution, we only define the proportion p of '1' values for the relevant variables.

```
params_bin <- list(c(p=0.6))
```

To finish the noise is defined as the proportion of random '1' values across the full matrix.

```
p_noise <- 0.02

dat2 <- simulateY(nclust=nclust,
            n_byClust=n_byClust, J=J,
            flavor="binary",
            params=params_bin,
            prop=prop, noise=p_noise)
```

The last type of dataset is a beta-like distribution. We begin by simulating a normal mixture distribution with two modes on the 10 relevant variables. Other variables are uniformly distributed. We add to this matrix a Gaussian noise and we transform the data with the following formula: $f(x) = \frac{1}{exp(-x)}$

```
params_beta <- list(c(mean1=-2,
mean2=2, sd1=0.5, sd2=0.5))
sd <- 0.02

dat3 <- simulateY(nclust=nclust,
        n_byClust=n_byClust, J=J,
        flavor="beta", params=params_beta,
        prop=prop, noise=sd)
```

A1.2 *Evaluation of the number of clusters*

SNF, ConsensusClustering, iClusterPlus, intNMF, CIMLR and PINSPlus have their own functions to evaluate the number of clusters.

For the other, we use the matrix that represents the individuals on the low dimensional space and we apply a hierarchical clustering using euclidean distance and Ward method. Then, we apply the function `NbClust` from the NbClust package that compute 26 criteria to evaluate the number of clusters [62]. If all the criteria do not lead to the same number of clusters we keep the value selected by the majority of criteria.

A1.3 *Variable importance evaluation*

To evaluate the performance of each method to find the correct variables in each single dataset that split the patients into subgroups, ROC curves [63] were used to illustrate the TPR or sensitivity as a function of the FPR. The TPR is defined by $TPR = \frac{TP}{TP+FN}$ where TP is the number of true positives, and FN is the number of false negatives. The FPR is defined by $FPR = \frac{FP}{FP+TN}$, where FP is the number of false positives, and TN is the number of true negatives.

If the TPR is equal to 1, this means that all the correct variables are detected. Simultaneously, if the FPR is equal to 0, this means that there is no error in the detection.

Usually, a ROC curve is a plot with a discrimination threshold between that varies. For each variable, a global coefficient is computed by sum the absolute value on each latent variable. Here, the variables are sorted by order of importance (from the largest global coefficient to the smallest from the matrices **H** or **a**) for each method. We then make the threshold of the ROC curve

vary by adding one variable at a time into the set of detected variables.

Then, ROC curves from TPR and FPR were summarized with the AUC. A good performance corresponds to an AUC equal to 1 and a bad performance corresponds to an AUC equal to 0. AUC measures allow evaluating the capacity of all the methods to detect the correct variables linked to the subgroups.

A1.4 *Evaluation of the stability of variable selection*

To explore the stability of variable selection, we proceed by bootstrapping and more precisely by the jackknife-like method [64]. It is a standard procedure that consists in running a method multiple times on the same dataset by removing one observation at each time.

At the end of the jackknife procedure, for one method, n models have been run. Then, we can compare which variables have been selected across these n models.

A1.5 *Preprocessing of the data*

For simulated data and mutation data in liver cancer, a column full of zeros in the binary block was filtered out from the analysis. According to the recommendations of the authors of the methods, we applied adapted various preprocessing. For MoCluster, data were centered, SGCCA data were scaled. For intNMF, data were transformed to be positive. ConsensusClustering centered the data with the median. The other methods do not perform any preprocessing of the data.

A1.6 *Heatmap for BXD dataset*

A1.7 *Session Information*

```
R version 3.5.1 (2018-07-02)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS Sierra 10.12.6
attached base packages:
[1] parallel  stats  graphics grDevices utils
datasets  methods    base

other attached packages:
 [1]   ggplot2_3.1.0  dplyr_0.8.0.1
 CrIMMixdata_0.1.0   CrIMMix_0.1.9-9000
```