M3C: A Monte Carlo reference-based consensus clustering algorithm

Christopher R. John[1*], David Watson[2], Myles Lewis[1], Dominic Russ[1], Katriona Goldmann[1], Michael Ehrenstein[3], Costantino Pitzalis[1], Michael Barnes[1]

[1]William Harvey Research Institute, Bart's and The London School of Medicine and Dentistry, Queen Mary University of London, Charterhouse Square, London, EC1M 6BQ, United Kingdom

[2]Oxford Internet Institute, University of Oxford, 1 St. Giles, Oxford, OX1 3JS

[3]Rayne Institute, University College London, 5 University Street, London, WC1E 6JF, United Kingdom

*To whom correspondence should be addressed

Abstract

Genome-wide data is used to stratify patients into classes for precision medicine using class discovery algorithms. A widely applied method is consensus clustering; however, the approach is prone to overfitting and identification of false positives. These problems arise from not considering null reference distributions while selecting the number of classes (K). As a solution, we developed a reference-based consensus clustering algorithm called Monte Carlo consensus clustering (M3C). M3C uses a Monte Carlo simulation to generate null distributions along the range of K, which are used to decide its value and reject the null hypothesis. The M3C method clearly removes the limitations of consensus clustering as demonstrated in both simulations and investigation of The Cancer Genome Atlas expression data. M3C can quantify structural relationships between clusters and uses self-tuning spectral clustering to analyse complex structures. In parallel, we developed clusterlab, a flexible Gaussian cluster simulator to test class discovery tools. Clusterlab can simulate high dimensional Gaussian clusters with precise control over spacing, variance, and size. This computational framework should prove useful in the development of precision medicine.

Contact: christopher.john@qmul.ac.uk or m.r.barnes@qmul.ac.uk

Introduction

Stratified medicine is the concept that patients may be divided into groups related to their prognosis and it hinges on the discovery of clinically relevant patient classes. Increasingly, patient genome-wide data is being used to perform class discovery[1-6] and this is powered by class discovery algorithms. Class discovery analysis of genome-wide data (transcriptomics, epigenomics, proteomics, DNA copy number) has been a successful approach in the field of cancer biology for finding subtypes with distinct clinical outcomes[1-6], and there are signs of this starting to occur for other diseases[7-9]. Therefore, there exists a demand for algorithms that deliver robust results. Example class discovery methods applied in the field include the original consensus clustering[10], NMF consensus clustering[11], the GAP-statistic[12], CLEST[13], progeny clustering[14], and hierarchical clustering combined with dendrogram cutting[15]. Typical problems with class discovery tools include; inability to reject the null hypothesis K=1, overfitting due to a lack of the appropriate reference, and inadequate demonstration of newly developed methods using both real and simulated data.

Şenbabaoğlu, et al. (2014) made an important contribution as they demonstrated that false positive structures could be found in cluster less data using consensus clustering[16]. However, hypothesis testing was not performed in this study. The authors suggested to generate null datasets with the same correlation structure as the real for a formal test of cluster strength. Although, they gave no method for conducting a formal test. They described a method called the proportion of ambiguous clustering (PAC) score as an alterative and better way for chosing K than the original delta K metric[10] after consensus clustering. However, the PAC score does not take into account null reference

distributions and so is subject to overfitting and declaration of false positives. Lastly, although the authors did conduct simulations to demonstrate their methods performance they did not test its performance on real data.

The original consensus clustering[10] and non-negative matrix factorisation (NMF) consensus clustering[11] algorithms are among the most widely used automated class discovery tools by The Cancer Genome Atlas (TCGA) and other influential consortia[1-9,17-20]. Therefore, for our initial investigations of potential bias, we chose consensus clustering with the PAC metric and NMF consensus clustering with its cophenetic co efficient metric as representative of state-of-the-art methods. By demonstrating an overfitting effect, a lack of objective assessment of confidence, and a tendency to declare false positive structures in both methods, we show the logic for applying our solution called Monte Carlo consensus clustering (M3C). M3C uses a multi-core enabled Monte Carlo simulation to generate null datasets with the same correlation structure as the real. These null datasets are passed into its consensus clustering algorithm and used to calculate reference distributions of PAC scores for each value of K. The reference distributions are used to compare the real PAC scores with the expected reference under the null. This change results in a more statistically principled approach to consensus clustering. M3C is also capable of dealing with complex structures using self-tuning spectral clustering[21], and can quantify structural relationships between consensus clusters using hierarchical clustering and SigClust[22].

During the development of M3C, we found that current Gaussian cluster simulation methods were inadequate for systematic testing of class discovery tools. MixSim[23], generates Gaussian clusters, however it is not possible to precisely control their positioning. The Python scikit-learn machine learning module[24] contains a Gaussian cluster simulator, but it generates them randomly and controlled positioning is not possible. Another method allows controlled spacing[16], but does not

generate Gaussian clusters, instead the clusters resemble triangular slices and the variance and size cannot be set. Therefore, in this study, we also developed Clusterlab (https://cran.r-project.org/web/packages/clusterlab/index.html) a flexible Gaussian cluster simulator that solves the above problems. Clusterlab uses a novel method which involves generating Gaussian clusters on the perimeter of a circle. By rotating and multiplying sample coordinates it can generate complex arrangements in NXN dimensional space.

The development of M3C presented us with an opportunity to test two hypotheses on real data. First, that pre-existing high-profile publications contain results that declare evidence of structure when in fact there is none. Second, that not considering reference distributions along the range of K when discovering classes leads to systematic bias. The results in this manuscript imply a more rigorous approach is required. Our solution to this is available from the Bioconductor (https://www.bioconductor.org/packages/3.7/bioc/html/M3C.html).

Results

Systematic bias detected in widely applied consensus clustering algorithms

Using clusterlab, we first generated a null dataset where no genuine clusters are found (Fig. 1a). Running consensus clustering with the PAC score yielded steadily decreasing PAC scores, falsely indicating increased stability according to its decision rule, giving an optimal K of 8 (Fig. 1b). A similar, but reversed effect, was observed in the cophenetic metric of NMF where it decreased steadily indicating decreased stability, giving an optimal K of 2 (Fig. 1c). This is an overfitting effect. Both methods also declare evidence of structure when it does not exist. These findings point to the requirement for a reference-based consensus clustering tool that can test the null hypothesis K=1.

To further demonstrate the functionality of clusterlab, we generated a ring of four Gaussian clusters, four clusters with varying variance, and a more complex multi-ringed structure consisting of 25 Gaussian clusters (Supplementary Fig. 2).

M3C provides a one-step solution to class discovery hypothesis testing

The method we developed to solve the problems with current consensus clustering algorithms is overviewed in a schematic in Figure 2a. For our initial investigations of performance, we tested M3C on a negative control, a simulated dataset where no real structure is found (Fig. 2b). The Relative Cluster Stability Index (RCSI) could not distinguish real from false structure. In contrast, the calculation of Monte Carlo p values by M3C correctly inferred that there is no genuine structure in this negative control dataset (alpha = 0.05), and no overfitting effect favouring lower or higher values of K was observed. To reinforce this result, 1000 random normally distributed datasets without structure were generated then tested for significance using M3C. Across these simulations M3C correctly called K=1, 91% of the time. Next, M3C was tested on a positive control simulated dataset where 4 clusters exist (Fig. 2c). The PAC score and the RCSI correctly identified 4 as the optimal value of K. A very low Monte Carlo p value was found by M3C for K=4 ($p = 9.95x10^{-21}$), correctly implying this is the optimal value of K and adding a degree of objective confidence to the decision.

Next, we reanalysed a range of high profile stratified medicine datasets where structure had been declared to test for false positive structures (Table 1 & Supplementary Table 1). Because of the ease of data availability, these were predominately, but not exclusively, from TCGA. Table I demonstrates the pervasive use of consensus clustering and NMF consensus clustering for class discovery in the field. Using M3C, we identified two of these datasets where no significant evidence of structure

could be identified. Firstly, a systemic lupus erythematosus (SLE) microarray dataset was analysed where 7 major subtypes were reported using hierarchical clustering and dendrogram cutting. However, none of the p values along the range of K calculated by M3C reached statistical significance (the lowest was for K = 3, p = 0.15) (Fig. 2d). Secondly, a breast cancer miRNA-seq dataset was identified with no significant evidence of structure (the lowest was for K = 4, p = 0.27), whereas 7 subtypes were originally reported using NMF (Fig. 2e). These findings imply that false positive structures exist in the literature through not conducting hypothesis testing.

Table I: Datasets selected for assessment using M3C. More detailed information on the data used is included in Supplementary Table I.

Publication	Year	Data type	Original algorithm	M3C optimal K
Glioblastoma[3]	2008	Microarray	Consensus clustering	4
Ovarian carcinoma[4]	2011	Microarray	NMF	5
Lung cancer[5]	2012	RNA-seq	NMF	2
Breast cancer[18]	2012	miRNA-seq	NMF	1
Diffuse glioma[1]	2016	RNA-seq	Consensus clustering	8
Lupus[9]	2016	Microarray	Hierarchical clustering	1
Pheochromocytoma[2]	2017	RNA-seq	Consensus clustering	6

Further evidence for systematic bias in class discovery

Of those datasets that exhibited significant evidence of structure using M3C, we used this as an opportunity to contrast the clarity of the M3C results with those from consensus clustering with the PAC score[16], the NMF cophenetic co efficient[11], and the GAP-statistic[12]. Our intention in these analyses was not to dispute the original optimal K, but we hypothesised that methods that do not consider reference distributions along the range of K would lead to visible biases. In these analyses, it was demonstrated that the GAP-statistic continuously increased implying improving stability regardless of the structure (Supplementary Fig. 1). These findings imply the GAP-statistic is not well suited to the high dimensional setting. Across these datasets, we also demonstrate why M3C fits a beta distribution to the data to estimate extreme tail values, as for K=2, the beta distribution fits the reference slightly better than a normal (Supplementary Fig. 2 and 3). This step is important as it

removes the limitations on p value derivation imposed by a finite number of simulations (Supplementary Fig. 4).

The PAC score displayed the same overfitting effect as observed earlier on simulated null datasets, decreasing steadily regardless of the structure, implying increased stability (Figure 3a-e). This effect is more of a problem in datasets where the clustering is not very clear. For the GBM dataset[3], while a PAC elbow can be seen at K=4, the global optimal value is K=10 (Fig. 3a). The problem with the PAC score resembles the problem encountered by Tibshirani, et al. (2001), when the authors developed the GAP-statistic to overcome the subjective decision regarding the elbows location. For the GBM case, the Monte Carlo p values and the RCSI demonstrated a clear optimal point at K=4 ($p = 0.00059$), with additional significant evidence for structure at K=5 ($p = 0.0071$).

For the ovarian dataset[4], a global optimal PAC value is seen for K=2, which is supported by the RCSI (Fig. 3b). However, when the Monte Carlo p values are calculated, it is in fact K=5 which is the optimal K ($p = 0.0078$). This happens because on some datasets a skew exists in the consensus clustering reference distribution for K=2 that is towards lower PAC scores (Supplementary Fig. 2b), so they are inherently favoured by the algorithm, neither the PAC score or the RCSI correct for this problem. Only by calculating p values for each value of K can these types of systematic biases be dealt with.

In cases where the clustering is very clear, the PAC score does perform well. In the lung cancer dataset[5], a global PAC optimal K can be seen at K=2, which is supported by both the RCSI and the Monte Carlo p value ($p = 0.0018$) (Fig. 3c). Although this conflicts with the original decision of K=4, the M3C p value for K=4 was also significant ($p = 0.0032$), implying this would be another reasonable

choice. However, the overfitting problem of consensus clustering can be observed again on the diffuse glioma dataset[1] (Fig. 3d). Here the PAC score continuously decreases until it reaches a global optimum at K=10. However, considering the reference distributions, M3C informs us that K=8 is the most significant option ($p = 3.5x10^{-9}$), which is also supported by the RCSI score. For the paraganglioma dataset[2], the RCSI calls K=6 the optimal value and the Monte Carlo p value supports this ($p = 1.6x10^{-6}$), while the PAC score continually decreases, not giving clarity on the appropriate choice of K (Fig. 3e). This is another example of why the reference distribution matters, as the RCSI method shows a local maximum for K=2, while the Monte Carlo p value does not support this. This is due to the uneven shape of the PG reference distribution for K=2 which has kurtosis towards a higher PAC score (Supplementary Fig. 3b). These findings imply results relying just on relative scores or mean comparisons with the reference can be potentially misleading.

In agreement with our findings on null simulated data, it was observed, the NMF cophenetic co efficient has a tendency towards calling K=2 on real data (Fig. 3a-e). Only in the diffuse glioma dataset[1] did the maximum cophenetic co efficient change from K=2 to another value of K. Although there are numerous variant decision rules for NMF in use[4,5,18], none consider the reference distribution shape. Instead of taking the most stable consensus matrix (highest cophenetic co efficient) as the optimal K, local maxima are often selected[4,5]. Notably, for the ovarian dataset[4] a local maximum in the NMF cophenetic co efficient was observed at K=5, which was supported by the M3C decision, in this instance. Additional support was observed for the lung cancer (2012) optimal K, as a NMF global maximum cophenetic co efficient was detected for K=2, and the M3C p value also detected this K as optimal (p=0.0018). However, since a tendency in NMF towards K=2 on null datasets has been observed in this study, how confident we may be in this decision is unclear. Overall, these findings highlight the value of M3C's class discovery method.

M3C demonstrates good performance in class discovery simulations

Next, we sought to evaluate the performance of M3C on simulated data with a range of clusters from 2 to 6 and compare its performance to existing algorithms. In these tests, the clusterlab alpha parameter was varied which controls the distance between the clusters and we sought algorithms which we able to detect the true K from further apart cluster conditions (alpha=2) to closer ones (alpha=1) (Fig. 4a,b). Typically, in genome wide analyses many clusters will be overlapping and hard to distinguish from one another, therefore sensitivity under these conditions is very valuable. This analysis found that M3C using the RCSI score performed better than consensus clustering with the PAC score, M3C using the p value score, the GAP-statistic, CLEST, the original consensus clustering with the delta K score, NMF, and progeny clustering. Notably, while M3C with the RCSI score was approximately 10% higher in accuracy than M3C with p values, the GAP-statistic, and consensus clustering with PAC, these three methods performed similarly, within 4% of one another. Overall, these simulations reinforce our previous findings on real data that M3C performs better than the current state-of-the-art consensus clustering algorithms.

M3C can deal with complex structures using spectral clustering

The performance of M3C is dependent on the inner algorithm of consensus clustering. Although k-means and PAM perform well on the types of data generally encountered in genome-wide studies, they assume the clusters are approximately spherical and equal in variance, this may not be true. Spectral clustering is a widely applied technique due to its ability to cope with a broad range of structures[20]. Therefore, to increase its capabilities, M3C was coded to include self-tuning spectral clustering[22]. We tested spectral clustering as M3C's inner algorithm versus PAM and k-means on two synthetic datasets, one where the clusters were anisotropic (Fig. 5a), and a second where one cluster had a far smaller variance than its neighbouring cluster (Fig. 5b). Under these conditions, it was observed that M3C using PAM and k-means both had problems assigning the true K and

classifying the members of each cluster correctly. On the other hand, M3C using spectral clustering did not suffer these drawbacks. Using spectral clustering, M3C is also capable of recognising more complex non-Gaussian shapes, such as half-moons and concentric circles (Supplementary Fig. 5). The addition of spectral clustering to M3C allows greater flexibility in the range of structures that may be examined.

M3C can quantify structural relationships between consensus clusters

An important question when the optimal K has been decided is, how do the discovered clusters relate to each other? Inherently, consensus clustering does not distinguish between flat versus hierarchical structure. To solve this, M3C takes the medoids of each consensus cluster and performs hierarchical clustering on those. To make the analysis statistically principled, M3C uses the SigClust method[22] on each pair of consensus clusters iteratively, then display the pairwise p values for each split of the dendrogram. Testing M3C on the PG dataset revealed a hierarchical relationship between the 6 clusters (Fig. 6a), with, for example, consensus clusters 1 and 2 grouping together ($p = 1.2x10^{-80}$). In contrast, testing M3C on a null dataset without clusters demonstrated insignificant SigClust p values (alpha = 0.05) and a flat dendrogram (Fig. 6b). The addition of a hierarchical clustering stage, post choosing the optimal K, should prove helpful in quantifying structural relationships and adds another layer of confidence assessment.

M3C can perform quickly in a multi-core setting

As a final step, we decided to evaluate M3C's internal parameters using the PAM algorithm. A sensitivity analysis of the number of inner replications and outer simulations found M3C generally yielded stable results across 6 TCGA datasets (with inner replications = 100, outer simulations = 100) (Supplementary Fig. 6-7). We also examined the speed of M3C across the 5 datasets (using 8 cores)

to find that the runtime was between 2-25 minutes, depending on dimensionality (Fig. 7). We consider this acceptable considering the unique results M3C generates.

Discussion

We report on the development of a reference-based consensus clustering algorithm with a Monte Carlo simulation driven hypothesis framework, named M3C. Our investigations into state-of-the-art methods imply there are systematic biases existing in current widely used class discovery algorithms. These occur due to not considering the reference distribution along the range of K when deciding on its value. Although considering these distributions is a relatively straightforward procedure, it has major implications. To date, testing of the null hypothesis by TCGA, if performed, has been conducted in a post-hoc manner by SigClust[22] after deciding on the value of K using the standard methods[2,6,18]. The advantage of the M3C approach is that it is a one-step procedure that combines optimal K selection and null hypothesis testing. This way bias from not considering the reference distributions when deciding on K does not enter the analysis.

Our reanalysis of high profile stratified medicine studies, predominantly from TCGA[1-5,9,18], questions the value of applying conventional algorithms without considering the appropriate reference distributions. Both consensus clustering and NMF displayed an inherent tendency towards high or low values of K. This overfitting problem coupled with subjective decision making as to what constitutes the optimal K, similar to the original elbow problem solved by the GAP-statistic[12], may lead to misleading results. Demonstrated in this study, was the identification of two cases in the literature where structure had been declared despite M3C indicating no significant structure. In the case of the SLE study, seven subtypes were originally declared in a major transcriptomic analysis[9].

Within the context of these new findings, it is perhaps better to describe these subtypes as existing within a noisy spectrum of non- distinct states. This finding could have implications for the development of personalised therapies.

It is necessary to remark on the limitations of our approach. Although from a statistical standpoint it is possible to more objectively decide on the number of classes using M3C, other values of K should be considered rather than just those that are most significant. If choices are non-significant or less significant, but the relationship between the clinical variables and the clusters is significant and can be validated in an external dataset, this is of course, an indicator of a compelling finding. The main caveat of M3C is that it is computationally expensive. With the limits on computational power, the implementation of extreme tail estimation and multi-core ability for the M3C algorithm were key steps to establishing a working hypothesis testing framework.

The M3C methodological developments are built primarily on the work of Monti, et al. (2003), Şenbabaoğlu, et al. (2014), and Tibshirani, et al. (2001). M3C uses the original Monti, et al. (2003) consensus clustering algorithm with the Şenbabaoğlu, et al. (2014) PAC score to measure stability, but now it is reference-based with an outer Monte Carlo simulation, distribution fitting, and new scoring metrics. We also extended the framework with the ability to investigate more complex structures using self-tuning spectral clustering, and to quantify structural relationships and their significance post-hoc using hierarchical clustering and SigClust[22]. M3C contains convienient functions that interface with its main function for performing PCA and tSNE to assess structure. Overall, M3C should prove a straightforward and powerful tool to drop into existing pipelines.

Finally, it is important to mention the methodological contributions of clusterlab. Clusterlab generates data similar in nature to real cancer gene expression datasets, which are typically high dimensional and Gaussian[22]. We consider it more flexible, realistic, and precise than current alternatives[16,23,24]. This should prove useful in testing class discovery tools in the field. Its option to simulate more complex multi-ringed structures arranged in a customisable manner may appeal to those testing specialised single cell RNA-seq clustering algorithms, where there are thousands of samples, and also increases its appeal to researchers in other fields.

Conclusions

We report on the development of a reference-based consensus clustering algorithm for class discovery, and its implementation, in a R software tool called M3C. By generating reference distributions and comparing real stability scores against them, M3C provides a more statistically principled way to decide on the number of classes. In comparison, this study demonstrated systematic biases exist in current consensus clustering algorithms due to not considering reference distributions for each value of K. The framework also includes methods for investigating complex structures and quantifying structural relationships. Additionally, clusterlab, the sister method of M3C, is a flexible Gaussian cluster simulator that should be of broad interest. We anticipate M3C and clusterlab will contribute towards the development of precision medicine.

Endnotes

Competing interests: The authors declare that they have no competing interests.

Funding: No funding source to declare.

Acknowledgements: No specific acknowledgements to declare.

Authors' contributions C.R.J conceived and designed the approach. C.R.J and D.W wrote the manuscript. C.R.J and D.W wrote the code. C.R.J, D.W, and D.R performed data analyses. All authors reviewed and edited the manuscript. M.B, C.P, M.E, and M.L supervised the project.

Methods

M3C. *Simulation of the reference dataset.* There are a range of options for the generation of reference datasets in M3C's Monte Carlo simulation, however, M3C retains the correlation structure of the features of the input matrix, an approach first taken by Tibshirani et al. (2001). With an input matrix, $T \in \mathbb{R}^{S*F}$, using principal component analysis (PCA) we can extract the input data's eigenvector matrix, $A \in \mathbb{R}^{F*S}$ and its principal component score matrix, $Y \in \mathbb{R}^{S*S}$, where, F, is the number of features in the provided matrix, and, S, the number of samples. Let n be the nth Monte Carlo simulation for $n = 1 \dots N$. The steps taken to generate random data are then as follows:

1. Conduct PCA to obtain the orthogonal matrix of eigenvectors, A of the input data T:

$$Y_{S*S} = T_{S*F} * A_{F*S} \quad (1)$$

2. Next, a random PC score matrix is generated for the nth simulation, $Y^n \in \mathbb{R}^{S*S}$, where the ith column is filled with random values from a normal distribution with mean zero and standard deviation equal to the ith column in Y. Let, D_i be the standard deviation of Y_{*i} and for $i = 1 \dots S$:

$$Y_{*i}^{n} \sim N(0, D_i) \quad (2)$$

3. Multiplying Y^n with the transpose of A yields $Q^n \in \mathbb{R}^{S*F}$, a single simulated null dataset with the same feature correlation structure as T, but without clusters.

$$Q^n_{S*F} = Y^n_{S*S} * A^n_{S*F} \quad (3)$$

Steps 1-3 are repeated by M3C for each Monte Carlo simulation until N is reached, and for the nth simulation one random dataset, Q^n is passed into the consensus clustering algorithm (described below) to calculate null reference stability scores for $K = 1 \ldots maxK$. After N simulations, the consensus clustering algorithm is run just once on the input data for comparison using procedures we will go on to detail. M3C is set to use $N = 100$ and this was the parameter setting used for the simulations in this study.

Consensus clustering. The Monti et al. (2003) consensus clustering algorithm resamples with replacement the input data sample-wise, H times, and with each resampling iteration, clusters each perturbed dataset using a user defined inner clustering algorithm (e.g. PAM) for each value of K. It then measures the stability of the sample cluster assignments over all resampling iterations to decide K. M3C includes, PAM, k-means, and spectral clustering as options, with PAM set by default due to its superior speed. Let, $D^{(1)}, D^{(2)}, \ldots, D^{(H)}$ be the list of H perturbed datasets, and let $M^{(h)} \in \{0,1\}^{N*N}$ be the connectivity matrix resulting from clustering dataset $D^{(h)}$, the entries of $M^{(h)}$ are then defined as:

$$M^{(h)}(i,j) = \begin{cases} 1 & \text{if samples i and j are in the same cluster} \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

To keep count of the number of times samples i and j are resampled together in the perturbed dataset $D^{(h)}$ an indicator matrix $I^{(h)} \in \{0,1\}^{N*N}$ is defined:

$$I^{(h)}(i,j) = \begin{cases} 1 & \text{if samples i and j are in dataset } D^{(h)} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

The consensus matrix, $M \in [0,1]^{N*N}$, is defined as the normalised sum of all the connectivity matrices of all H perturbed datasets:

$$M(i,j) = \frac{\sum_{h=1}^{H} M^{(h)}(i,j)}{\sum_{h=1}^{H} I^{(h)}(i,j)} \quad (6)$$

The entry (i,j), or consensus index, is the number of times that two samples cluster together divided by the total number of times they were sampled together across all the perturbed datasets. A value of 1 would correspond to a perfect score as the two samples are always found in the same cluster across all resampling runs, while a value of 0 would correspond to the worst score as the two samples never are found in the same cluster. A consensus matrix is generated for every value of K and then the stability of each matrix quantified using an empirical cumulative distribution (CDF) plot. For any given consensus matrix M, the CDF is calculated and is defined over the range $[0,1]$ as follows:

$$CDF(c) = \frac{\sum_{i<j} 1\{M(i,j) \leq c\}}{N(N-1)/2} \quad (7)$$

Where $1\{\ldots\}$ denotes the indicator function, $M(i,j)$, denotes entry (i,j) of the consensus matrix M, N is the number of rows (and columns) of M, and c is the consensus index value.

Calculation of the PAC score. The CDF plot has consensus index values on the x axis and CDF values on the y axis. A perfectly stable cluster solution will have a flat CDF plot representing a matrix purely of 0s and 1s, therefore the degree of CDF flatness for each K is a measure of the stability of K. To quantify this, M3C uses the PAC score, a metric shown to perform well in simulations[16]. PAC is defined as the fraction of sample pairs with consensus index values falling in the intermediate sub-interval $(x_1, x_2) \in [0,1]$. For a given value of K, $CDF(c)$ corresponds to the fraction of sample pairs with consensus index values less than or equal to c and PAC is defined as:

$$PAC_K(x_1, x_2) = CDF_K(x_2) - CDF_K(x_1) \quad (8)$$

M3C calculates the PAC score with $x_1 = 0.1$ and $x_2 = 0.9$. Although the PAC window is a user defined parameter, we have found these settings to perform well in our experience.

Calculation of the RCSI. To account for the reference PAC scores from 1 to N, where N is the total number of Monte Carlo simulations, M3C uses the RCSI. Let, $Pref_{Kn}$ be the reference PAC score from the nth Monte Carlo simulation for a given K, and, $Preal_K$ the real PAC score for that K, then the $RCSI_K$ is defined as:

$$RCSI_K = \log_{10}\left(\frac{1}{N}\sum_{n=1}^{N} Pref_{Kn}\right) - \log_{10}(Preal_K) \quad (9)$$

Calculation of the Monte Carlo p value. To improve the selection of the optimal K, M3C derives Monte Carlo p values by testing the real PAC score for each K against the null PAC distribution, generated using simulated structureless data. Let o_K be the number of observed PAC scores in the reference less than or equal to the real PAC score, let N be the total number of Monte Carlo simulations, and the p value for that value of K, P_K is then defined as:

$$P_K = \frac{o_K+1}{N+1} \quad (10)$$

Where 1 is added the numerator and denominator to avoid p values of zero[25].

Calculation of the beta distribution p value. To estimate p values beyond the range of the Monte Carlo simulation, M3C fits a beta distribution. This is because it can deal with skew and kurtosis observed for the PAC reference scores for K=2 better than a normal distribution, and it falls within the interval $[0,1]$, like the PAC score. The α and β shape parameters required for the beta distribution are derived using the mean, μ, and variance, σ^2, of the reference PAC scores for any given K:

$$\mu = \frac{1}{N}\sum_{n=1}^{N} Pref_{Kn} \quad (11)$$

$$\sigma^2 = \frac{1}{N}\left(\sum_{n=1}^{N} Pref_{Kn} - \mu\right)^2 \quad (12)$$

$$\alpha = \left(\frac{1-\mu}{\sigma^2} - \frac{1}{\mu}\right)\mu^2 \quad (13)$$

$$\beta = \alpha\left(\frac{1}{\mu} - 1\right) \quad (14)$$

These α and β shape parameters are then used by M3C to generate the reference distribution for K. The real PAC score is used as a test statistic for derivation of the estimated p value. The beta distribution is defined on the interval $[0,1]$, let x denote the reference PAC score, then the beta probability density function (PDF) is defined as:

$$PDF(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\beta(\alpha,\beta)} \quad (15)$$

Self-tuning spectral clustering. Spectral clustering is a family of algorithms that cluster data using eigenvectors derived from the input data's graph Laplacian. M3C implements the Ng et al. (2002) spectral clustering algorithm, except a locally scaled kernel is used[26]. The kernel calculation depends on the value of σ, which is often a global parameter, and the clustering depends greatly on its value. However, local scaling uses a different σ for each sample, which is equal to the distance between the sample and its kth nearest neighbour. This allows the adjacency matrix to tune to both the data's scale and the sample's local statistics. Let, N be the number of samples, let the set of sample points to cluster be, $S = \{s_1, \dots, s_N\}$, and let $d(s_i, s_j)$ be the Euclidean distance between the ith and jth sample. The steps taken are then as follows:

1. Calculate the Euclidean distance between the ith sample, for $i = 1 \dots N$, and its kth nearest neighbour (M3C uses $k = 3$ which was an empirical decision based on our experience). This value corresponds to the sigma for the ith sample:

$$\sigma_i = d(s_i, s_k) \quad (16)$$

2. Generate the affinity matrix, $A \in \mathbb{R}^{N*N}$, using a radial basis function kernel, where $i \neq j$ and $A_{ii} = 0$. This matrix is resampled by M3C to determine the optimal K:

$$A_{ij} = \exp\left(\frac{-d^2(s_i,s_j)}{\sigma_i\sigma_j}\right) \quad (18)$$

3. Using D, the diagonal matrix whose (i, i) element is the sum of A's ith row, construct the matrix, L, the normalised graph Laplacian:

$$L = D^{-1/2} A D^{-1/2} \quad (19)$$

4. Find the $x_1, x_2, \dots\ x_K$, K largest eigenvectors of L, then form the matrix, $X = [x_1, x_2, \dots\ x_K] \in \mathbb{R}^{N*K}$ by stacking the eigenvectors in columns. K will be the Kth cluster in M3C's iteration over $1 \dots K$.

5. Form the matrix Y from X by renormalizing each of X's rows to have unit length:

$$Y_{ij} = \frac{X_{ij}}{(\sum_j X_{ij}^2)^{1/2}} \quad (20)$$

6. Each row of Y is treated as a sample, s_i, then all samples are clustered into K clusters using k-means.

Simulating NXN dimensional Gaussian clusters in a precise manner. *Generating evenly spaced points on the perimeter of a circle.* To control the spacing, size, and variance of synthetic clusters, clusterlab works within a 2D Cartesian coordinate system with an origin at $(0,0)$. First, the algorithm generates a set $S = \{w_i \in \mathbb{R}^2, i = 1, \dots X\}$ of X evenly spaced pairs of coordinates, where $w_i = (x_i, y_i)$, on the perimeter of a circle. Each of these coordinates later will be the centre of a Gaussian cluster, therefore, X is also the number of clusters to be generated. Let, r be the radius of the circle, then, for the ith cluster centre from $i = 1 \dots X$ we need to set $i = 0$ for the first cluster centre, so for $i = 0 \dots X - 1$, the coordinate pairs are calculated as follows:

$$x_i = \cos\frac{2\pi}{X \cdot i} r \quad (21)$$

$$y_i = \sin\frac{2\pi}{X \cdot i} r \quad (22)$$

This naturally leaves the r parameter as a means of controlling the spacing of the cluster centres. However, at this point, we also introduce an additional parameter for moving the ith cluster centre, α_i. α_i is a scalar that can be used to push each coordinate pair (or vector) away from its starting

point, yielding the transformed coordinates (x_i', y_i'). In the case of a cluster being left stationary, $\alpha_i = 1$. More specifically, for all pairs in set S, from $i = 1 \dots X$:

$$(x_i', y_i') = \alpha_i(x_i, y_i) \quad (23)$$

We also leave the option to add a final coordinate to S at (0,0), to allow a central cluster within the middle of the ring to be generated later.

Generation of more complex multi-ringed structures. As an optional next step to extend the single ring system, clusterlab can create multiple rings or concentric circles of 2D coordinates. After simulating the qth ring, as described above, from $q = 1 \dots Q$, the qth rings 2D coordinates are pushed away from the origin using vector multiplication with a scalar, let this scalar be β_q, let the newly transformed coordinates be (x_i'', y_i''), and so for $i = 1 \dots X$:

$$(x_i'', y_i'') = \beta_q(x_i', y_i') \quad (24)$$

Our new total number of samples, T, will be, $T = X * Q$. With each iteration from $q = 1 \dots Q$, the ith transformed coordinates, $d_i = (x_i'', y_i'')$, are added to a new set, $R = \{d_i \in \mathbb{R}^2, i = 1, \dots T\}$. Optionally, another layer of complexity may be added by using vector rotations of the qth rings coordinate pairs from $i = 1 \dots X$, by setting $\theta_q \neq 0$ in the following equation. To calculate each of the rings new coordinates (x_i''', y_i''') from $i = 1 \dots X$, the following calculation is performed for every pair:

$$x_i''' = x_i'' \cos(\theta_q) - y_i'' \sin(\theta_q) \quad (25)$$

$$y_i''' = x_i'' \sin(\theta_q) + y_i'' \cos(\theta_q) \quad (26)$$

Generation of Gaussian clusters. At this point we will assume that multiple rings have not been generated and we are working with, S, a set of (x_i', y_i') coordinates described by equation 23. However, the method that generates the Gaussian cluster multi-ringed system is identical to the single ringed system described below, except we start with the multiplied (x_i'', y_i'') or multiplied and rotated set of (x_i''', y_i''') points from the multi ring 2D coordinate set, R.

To form X Gaussian clusters of size M_i per cluster, we add Gaussian noise from a normal distribution, $N(0, D_i)$, to the ith pair of cluster centre 2D coordinates, $k_i = (x_i', y_i')$, to create the new coordinates, $t_i = (x_j, y_j)$. Performing this M_i times for each cluster centre, giving a total of $Z = \sum_{i=1}^{Z} M_i$ coordinate pairs, yields the final set, $J = \{t_i \in \mathbb{R}^2, i = 1, \dots Z\}$. The number of samples in each cluster may be set by varying M_i, and the clusters variance, by setting D_i. The new coordinate pairs, (x_j, y_j), to be added to, J, for all samples are calculated as follows:

$$(x_j, y_j) = (x_i' + N(0, D_i), y_i' + N(0, D_i)) \quad (27)$$

Projection of the final 2D coordinates into N dimensions. We transform the cluster sample coordinates into N dimensions with a previously explained method which uses a reverse PCA[16]. First, two random vectors are generated of length V, where V will equal the number of features in the final matrix, from a normal distribution $N(0,0.1)$, let these be v_1 and v_2. The SD of 0.1 was chosen empirically after examination of the scale of the simulated PC plots compared to those from real expression datasets. The v_1 and v_2 vectors are treated as fixed eigenvectors in this method, and each of our previously simulated coordinate pairs are treated as 2D PC scores. The final matrix, $F \in \mathbb{R}^{Z*V}$, comprised of Z rows (samples) and V columns (features), is formed by linear combinations of the fixed eigenvectors with the pairs of PC scores. Let, x_i and y_i be the PC scores of the ith sample, from $i = 1 \dots Z$ from set J, then the ith row of the output matrix F is given by:

$$F_{i*} = x_i * v_1 + y_i * v_2 \quad (28)$$

Non-Gaussian structures. For generating structures used in the spectral clustering analysis, the CRAN clusterSim package version 0.47 was used[27]. For the anisotropic and unequal variance clusters, 90 samples were simulated with two dimensions with the cluster.Gen function using the default settings. For the half-moon clusters, the shapes.two.moon function was used with 90 samples, and for the concentric circles the shapes.two.circles function was used with 180 samples, both using default settings. The sample number was increased in the latter to prevent gaps forming in the concentric circles.

Real test datasets. All test datasets, apart from the SLE dataset, were already normalised and downloaded directly through TCGA publication page (https://tcga-data.nci.nih.gov/docs/publications/) during the period of April to June 2017, further details are provided in Supplementary Table 1. The diffuse glioma (DG) dataset is a RNA-seq matrix consisting of 2266 features and 667 samples[1] (https://tcga-data.nci.nih.gov/docs/publications/lgggbm_2015/LGG-GBM.gene_expression.normalized.txt). The GBM dataset, is a microarray matrix consisting of 1740 features and 206 samples[3] (https://tcga-data.nci.nih.gov/docs/publications/gbm_exp/unifiedScaledFiltered.txt), the feature list used was taken from a later publication on the same dataset[6]. The lung cancer (LC) dataset[5] used was a RNA-seq matrix consisting of 178 samples and 2257 features (https://tcga-data.nci.nih.gov/docs/publications/lusc_2012/gaf.gene.rpkm.20111213.csv.zip), the feature list used to filter this dataset was from an earlier publication where four subtypes had been identified[17] (http://cancer.unc.edu/nhayes/publications/scc/wilkerson.scc.tgz). The paraganglioma (PG) dataset downloaded was a RNA-seq matrix consisting of 173 samples and 3000 features (https://tcga-data.nci.nih.gov/docs/publications/pcpg_2017/PCPG_mRNA_expression_naRM.log2.csv.zip), the gene wise filtering scheme used was the same as described as in the corresponding publication[2]. The ovarian cancer (OV) dataset[4] was a RNA-seq matrix of 489 samples and 800 features (https://tcga-data.nci.nih.gov/docs/publications/ov_2011/TCGA_489_UE.zip), and the gene list used for subsequent filtering was obtained from an earlier publication that detected four subtypes[19]. The SLE dataset[9] used was a microarray matrix of 82 samples and 48 features, the data was obtained from GEO (GSE65391), normalised, and filtered in the manner described in the associated publication.

References

1 Ceccarelli, M. *et al.* Molecular profiling reveals biologically discrete subsets and pathways of progression in diffuse glioma. *Cell* **164**, 550-563 (2016).

2 Fishbein, L. *et al.* Comprehensive molecular characterization of pheochromocytoma and paraganglioma. *Cancer cell* **31**, 181-193 (2017).

3 Network, C. G. A. R. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature* **455**, 1061 (2008).

4 Network, C. G. A. R. Integrated genomic analyses of ovarian carcinoma. *Nature* **474**, 609 (2011).

5 Network, C. G. A. R. Comprehensive genomic characterization of squamous cell lung cancers. *Nature* **489**, 519 (2012).

6 Verhaak, R. G. *et al.* Integrated genomic analysis identifies clinically relevant subtypes of glioblastoma characterized by abnormalities in PDGFRA, IDH1, EGFR, and NF1. *Cancer cell* **17**, 98-110 (2010).

7 Lefaudeux, D. *et al.* U-BIOPRED clinical adult asthma clusters linked to a subset of sputum omics. *Journal of Allergy and Clinical Immunology* **139**, 1797-1807 (2017).

8 Ottoboni, L. *et al.* An RNA profile identifies two subsets of multiple sclerosis patients differing in disease activity. *Science translational medicine* **4**, 153ra131-153ra131 (2012).

9 Banchereau, R. *et al.* Personalized immunomonitoring uncovers molecular networks that stratify lupus patients. *Cell* **165**, 551-565 (2016).

10 Monti, S., Tamayo, P., Mesirov, J. & Golub, T. Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine learning* **52**, 91-118 (2003).

11 Gaujoux, R. & Seoighe, C. A flexible R package for nonnegative matrix factorization. *BMC bioinformatics* **11**, 367 (2010).

12 Tibshirani, R., Walther, G. & Hastie, T. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **63**, 411-423 (2001).

13 Giancarlo, R., Scaturro, D. & Utro, F. Computational cluster validation for microarray data analysis: experimental assessment of Clest, Consensus Clustering, Figure of Merit, Gap Statistics and Model Explorer. *BMC bioinformatics* **9**, 462 (2008).

14 Hu, C. W., Kornblau, S. M., Slater, J. H. & Qutub, A. A. Progeny clustering: a method to identify biological phenotypes. *Scientific reports* **5** (2015).

15 Johnson, S. C. Hierarchical clustering schemes. *Psychometrika* **32**, 241-254 (1967).

16 Şenbabaoğlu, Y., Michailidis, G. & Li, J. Z. Critical limitations of consensus clustering in class discovery. *Scientific reports* **4** (2014).

17 Wilkerson, M. D. *et al.* Lung squamous cell carcinoma mRNA expression subtypes are reproducible, clinically-important and correspond to different normal cell types. *Clinical cancer research*, clincanres. 0199.2010 (2010).

18 Network, C. G. A. Comprehensive molecular portraits of human breast tumours. *Nature* **490**, 61 (2012).

19 Verhaak, R. G. *et al.* Prognostically relevant gene signatures of high-grade serous ovarian carcinoma. *The Journal of clinical investigation* **123** (2012).

20 Network, C. G. A. R. Integrated genomic characterization of oesophageal carcinoma. *Nature* **541**, 169-175 (2017).

21 Ng, A. Y., Jordan, M. I. & Weiss, Y. in *Advances in neural information processing systems.* 849-856.

22 Liu, Y., Hayes, D. N., Nobel, A. & Marron, J. Statistical significance of clustering for high-dimension, low–sample size data. *Journal of the American Statistical Association* **103**, 1281-1293 (2008).

23 Melnykov, V., Chen, W.-C. & Maitra, R. MixSim: An R package for simulating data to study performance of clustering algorithms. *Journal of Statistical Software* **51**, 1 (2012).

24 Pedregosa, F. *et al.* Scikit-learn: Machine learning in Python. *Journal of machine learning research* **12**, 2825-2830 (2011).

25 Phipson, B. & Smyth, G. K. Permutation P-values should never be zero: calculating exact P-values when permutations are randomly drawn. *Statistical applications in genetics and molecular biology* **9** (2010).

26 Zelnik-Manor, L. & Perona, P. in *Advances in neural information processing systems.* 1601-1608.

27 Walesiak, M., Dudek, A. & Dudek, M. clusterSim: Searching for optimal clustering procedure for a data set. *R package version 0.36-1* (2008).

Figure legends

Figure 1. Initial investigations into bias in consensus clustering algorithms. (A) A principal component analysis plot of a simulated null dataset where no real clusters should be declared. (B) Result from consensus clustering using the PAC score to measure stability of the solution. (C) Result from NMF consensus clustering to determine the class number of the same data. Either approach declares evidence of structure when there is none.

Figure 2. Overview of the M3C algorithm and initial results. (A) A schematic of the M3C software framework. After exploratory PCA to investigate structure, the M3C function may be run which includes two functions; M3C-ref and M3C-real. The M3C-ref function runs consensus clustering with simulated random data sets that maintain the same gene-gene correlation structure of the input

data. While, the M3C-real function runs the same algorithm for the input data. Afterwards, the relative cluster stability index (RCSI), Monte Carlo p values, and beta p values are calculated. Structural relationships are then analysed using hierarchical clustering of the consensus cluster medoids with SigClust[22] to calculate significance of the dendrogram branch points. (B) Results from running M3C on a simulated null dataset, it can be clearly seen that the p values do not reach significance along the range of K, therefore the correct result is inferred, K=1. (C) Results from running M3C on a simulated dataset where 4 clusters are found, the correct decision is made by M3C. (D) Using M3C, a systemic lupus erythematosus dataset was detected with no significant evidence of structure. (E) Similarly, a breast cancer dataset was identified with no significant evidence of structure.

Figure 3. Further evidence of systematic bias existing in widely applied consensus clustering algorithms. (A) Results from running M3C on a glioblastoma dataset[3] found the optimal K was 4. Consensus clustering using the PAC-score shows an optimal K of 10, and NMF of 2. (B) Results from running M3C on an ovarian cancer dataset[4] found the optimal K was 5. Consensus clustering using the PAC-score shows an optimal K of 2, and NMF also of 2. (C) Results from running M3C on a lung cancer dataset[17] found the optimal K was 2. Consensus clustering using the PAC-score shows an optimal K of 2, and NMF also of 2. (D) Results from running M3C on a diffuse glioma dataset[1] found the optimal K was 8. Consensus clustering using the PAC-score shows an optimal K of 10, and NMF of 4. (E) Results from running M3C on a paraganglioma dataset[2] found the optimal K was 6. Consensus clustering using the PAC-score shows an optimal K of 10, and NMF of 2. It can be observed, consensus clustering using the PAC-score and NMF both tend towards K=10 or K=2, respectively, on real data.

Figure 4. M3C demonstrates good performance in class discovery on simulated data. (A) A sensitivity analysis was conducted for every algorithm for K=2 to K=6 while varying the alpha parameter of clusterlab (degree of Gaussian cluster separation). Accuracy was calculated as the fraction of correct optimal K decisions, and for each alpha, with 25 iterations performed at each step. CC(original) refers to the Monti et al. (2003) consensus clustering method, GAP-STAT refer to the GAP-statistic, CC(PAC) refers to consensus clustering with the PAC-score. (B) Performance was calculated across the range of K tested for each algorithm as the mean accuracy.

Figure 5. M3C uses spectral clustering to deal with complex structures. (A) Results from running M3C using either spectral, PAM, or k-means clustering on anisotropic structures. The results for K=2 for each inner algorithm are shown in all cases, in the corner of the plots are the optimal K decisions using the M3C RCSI (B) Similarly, results from testing different internal algorithms on structures of unequal variance.

Figure 6. M3C can investigate structural relationships between consensus clusters. M3C calculates the medoids of each consensus cluster, then hierarchical clustering is performed on these, SigClust[22] is run to detect the significance of each branch point. (A) Results from M3C structural analysis of the 6 clusters obtained from the paraganglioma dataset analysis[2], all p values were strongly significant, supporting the M3C decision of the declaration of structure. (B) Results from the same analysis run on a simulated null dataset of the same dimensions, no p values were significant.

Figure 7. M3C performs quickly in a multi-core setting. M3C runtimes (in minutes) for 5 datasets used in the analysis. Performance was measured on an Intel Core i7-5960X CPU running at 3.00GHz

using 8 cores. M3C was set in its default mode, this uses 100 Monte Carlo simulations and 100 inner iterations of consensus clustering, using the PAM algorithm.

Supplementary figure 1. Clusterlab demonstration of the simulation of three synthetic datasets. For explanation of parameters see methods. (A) Simulation of 4 Gaussian clusters with $D = 1$, $M = 50$, $r = 8$. (B) Simulation of 4 Gaussian clusters with $D = (0.5,1,1.5,3)$, $M = 50$, $r = 8$. (C) Simulation of 5 rings of 5 Gaussian clusters with $D = 6$, $M = 100$, $r = 7$, $\beta = (2,4,6,8,10)$, $\theta = (30,90,180,0,0)$.

Supplementary figure 2. Results from running the GAP-statistic across 5 datasets using the PAM algorithm. A potentially misleading trend towards increased stability at higher values of K can clearly be observed.

Supplementary figure 3. Reference distributions generated using M3C of PAC scores across the range of K from 2-5. 100,000 simulations were performed to generate the distributions and a normal (yellow) and beta (blue) fit were compared. Using the Akaike information criterion (AIC) it can be observed that for K=2, a beta distribution fits slightly better than the normal (indicated by a more negative AIC). This is due to the skew and kurtosis that the reference distributions often have at K=2.

Supplementary figure 4. See supplementary figure 3 legend for details, the same analysis was performed, but with different datasets.

Supplementary figure 5. Comparing M3C Monte Carlo p values with p values estimated from a beta distribution. The beta distribution is generated using parameters from the initial N Monte Carlo

simulations M3C performs. Tail estimation allows the limits of a finite number of simulations to be overcome.

Supplementary figure 6. Comparing the M3C results using spectral clustering, PAM, or K-means on concentric circles or half-moon shapes.

Supplementary figure 7. A sensitivity analysis of M3C parameters across 5 real datasets. (A) Results from the RCSI are shown. The inner resampling replications of M3C were varied, while the outer Monte Carlo simulations were left to default (value of 100). Stability is measured on the y axis as the percentage of times the same result was obtained across 25 iterations. (B) Results from the beta distribution after conducting the same analysis described in (A).

Supplementary figure 8. Same analysis described in supplementary figure 7, except the outer Monte Carlo simulations of M3C were varied, the inner resampling replications were left to default (value of 100).

Figure 1

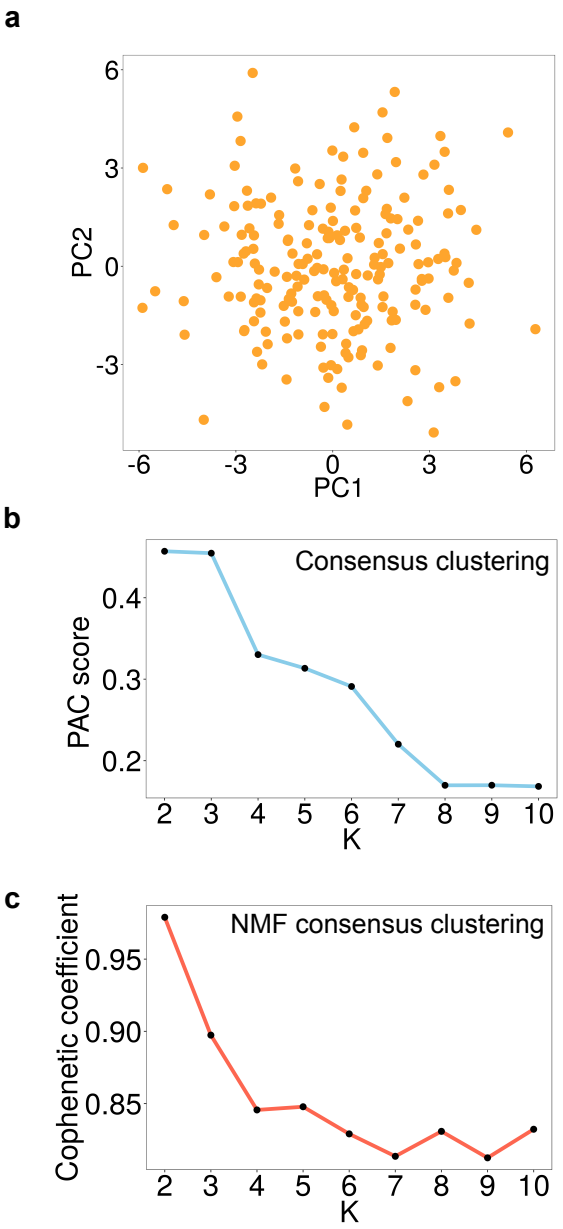

a
6
3
0
-3
PC2
-6
-3
0
3
6
PC1
b
Consensus clustering
0.4
0.3
0.2
PAC score
2
3
4
5
6
7
8
9
10
K
c
NMF consensus clustering
0.95
0.90
0.85
Cophenetic coefficient
2
3
4
5
6
7
8
9
10
K

Figure 2

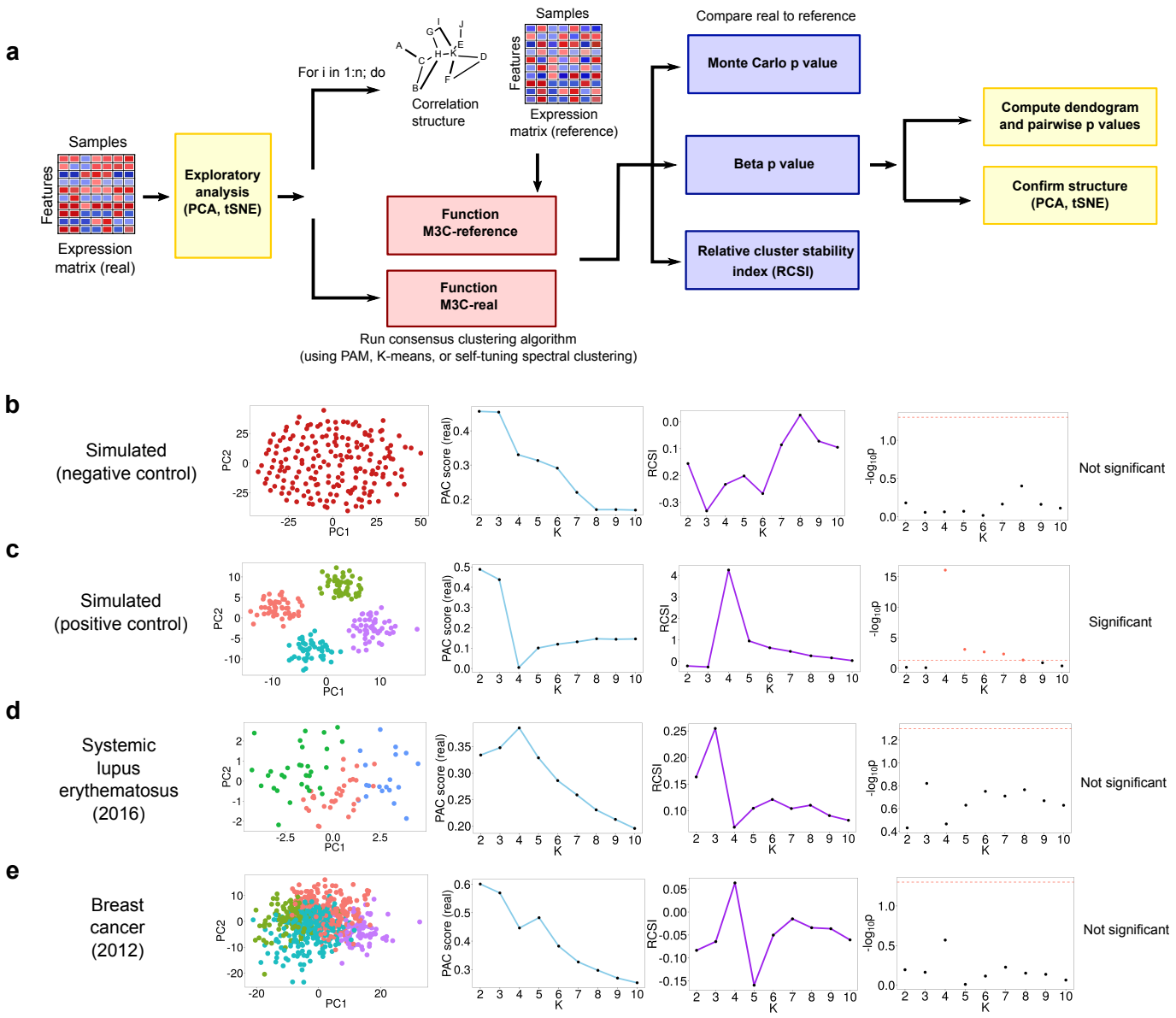

a
Samples
Features
Expression matrix (real)
Exploratory analysis (PCA, tSNE)
For i in 1:n; do
Correlation structure
Samples
Features
Expression matrix (reference)
Function M3C-reference
Function M3C-real
Run consensus clustering algorithm (using PAM, K-means, or self-tuning spectral clustering)
Compare real to reference
Monte Carlo p value
Beta p value
Relative cluster stability index (RCSI)
Compute dendogram and pairwise p values
Confirm structure (PCA, tSNE)
b
Simulated (negative control)
Not significant
c
Simulated (positive control)
Significant
d
Systemic lupus erythematosus (2016)
Not significant
e
Breast cancer (2012)
Not significant
PC1
PC2
PAC score (real)
RCSI
K

Figure 3

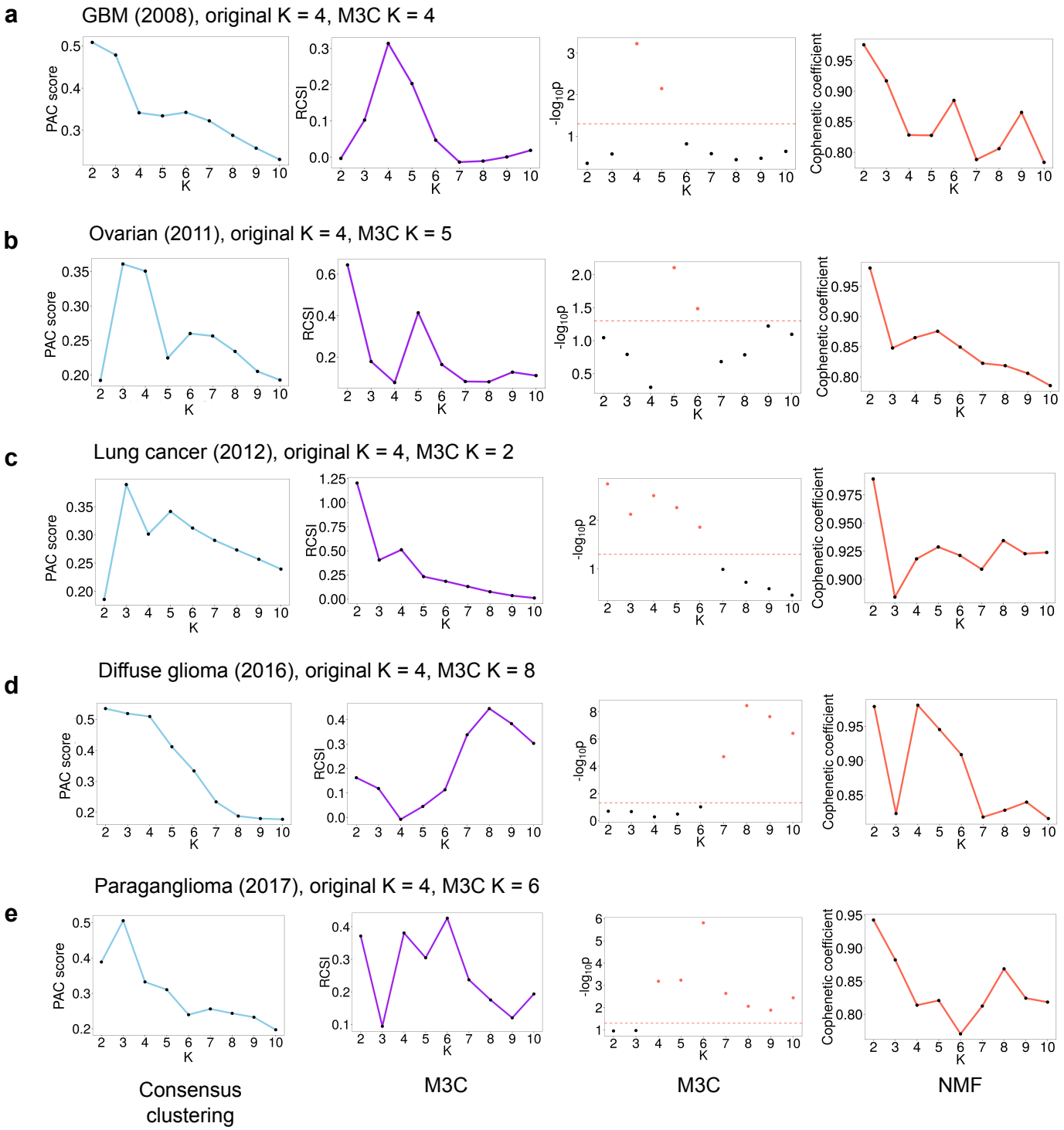
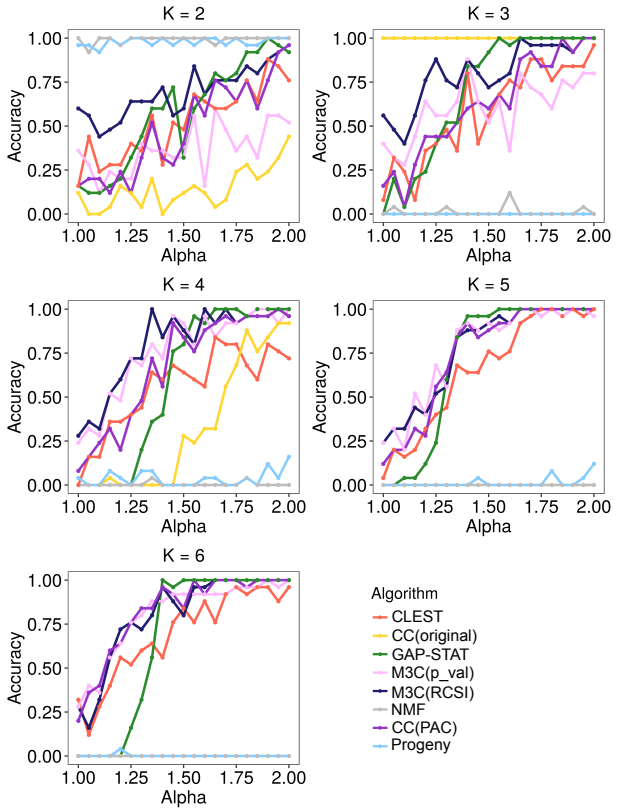

a
GBM (2008), original K = 4, M3C K = 4
b
Ovarian (2011), original K = 4, M3C K = 5
c
Lung cancer (2012), original K = 4, M3C K = 2
d
Diffuse glioma (2016), original K = 4, M3C K = 8
e
Paraganglioma (2017), original K = 4, M3C K = 6
PAC score
RCSI
-log10p
Cophenetic coefficient
K
Consensus clustering
M3C
M3C
NMF

Figure 4

a


K = 2
K = 3
K = 4
K = 5
K = 6
Accuracy
Alpha
1.00
0.75
0.50
0.25
0.00
1.00 1.25 1.50 1.75 2.00
Algorithm
CLEST
CC(original)
GAP-STAT
M3C(p_val)
M3C(RCSI)
NMF
CC(PAC)
Progeny

b

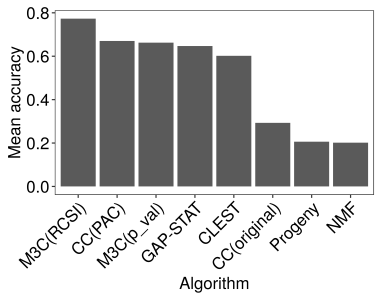

Mean accuracy
0.8
0.6
0.4
0.2
0.0
M3C(RCSI)
CC(PAC)
M3C(p_val)
GAP-STAT
CLEST
CC(original)
Progeny
NMF
Algorithm

Figure 5

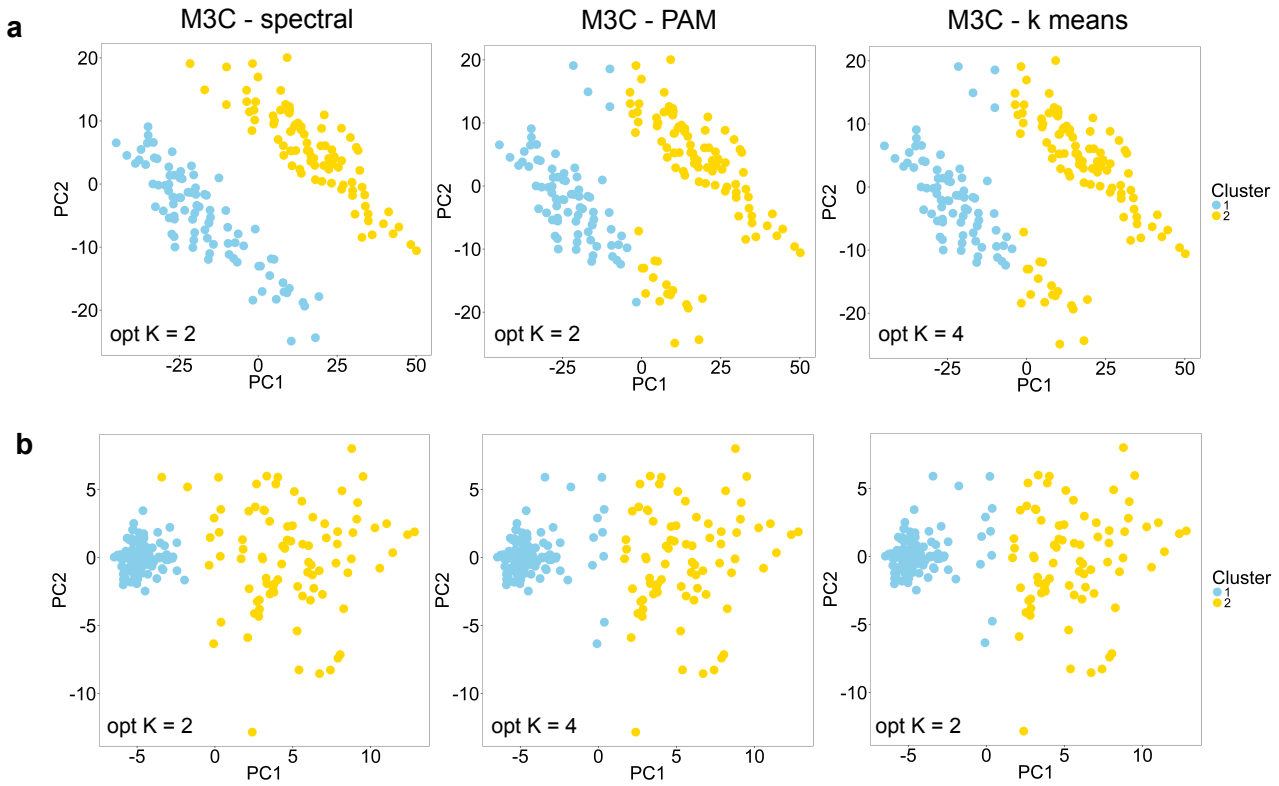

a
M3C - spectral
M3C - PAM
M3C - k means
PC2
PC1
opt K = 2
opt K = 2
opt K = 4
Cluster
1
2
b
opt K = 2
opt K = 4
opt K = 2

Figure 6

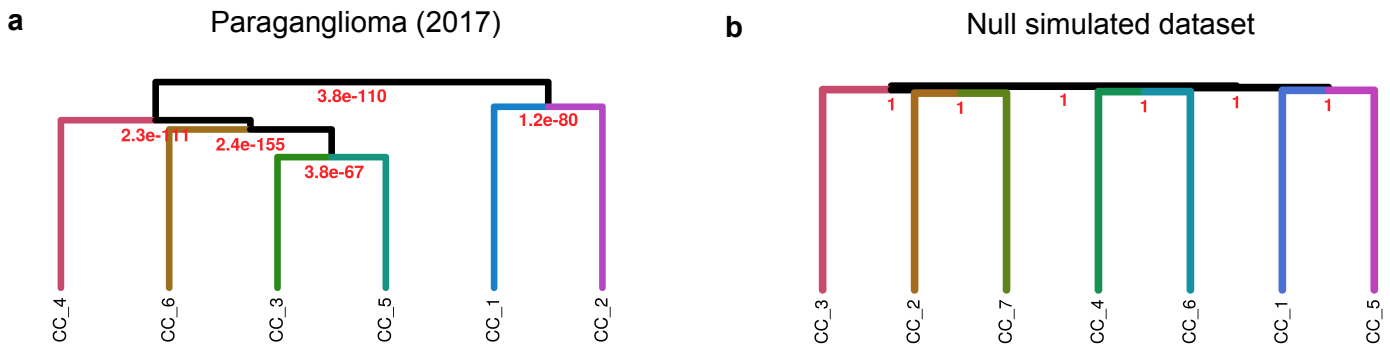
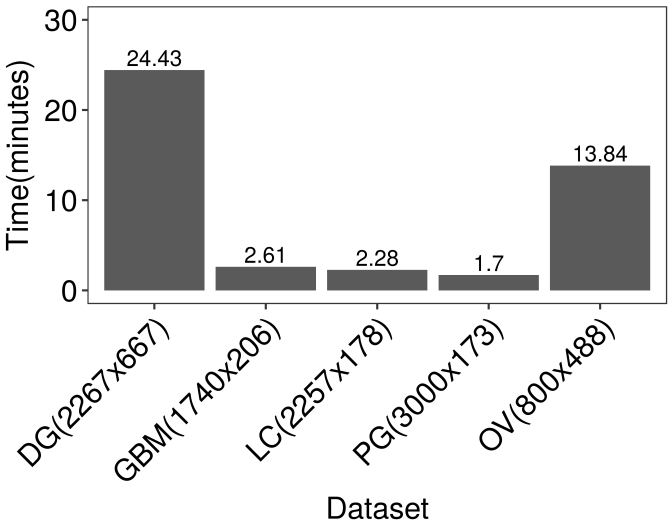

a
Paraganglioma (2017)
3.8e-110
2.3e-111
2.4e-155
3.8e-67
1.2e-80
CC_4
CC_6
CC_3
CC_5
CC_1
CC_2
b
Null simulated dataset
1
1
1
1
1
1
CC_3
CC_2
CC_7
CC_4
CC_6
CC_1
CC_5

Figure 7


30
20
10
0
Time(minutes)
24.43
2.61
2.28
1.7
13.84
DG(2267x667)
GBM(1740x206)
LC(2257x178)
PG(3000x173)
OV(800x488)
Dataset