Computational and Structural Biotechnology Journal 20 (2022) 6473–6489

COMPUTATIONAL ANDSTRUCTURAL BIOTECHNOLOGY JOURNAL

journal homepage: www.elsevier.com/locate/csbj

Automatic, fast, hierarchical, and non-overlapping gating of flow cytometric data with `flowEMMi v2`

Carmen Bruckmann [a,c], Susann Müller [a], Christian Höner zu Siederdissen [b,c,*]

[a] Helmholtz-Centre for Environmental Research, Department of Environmental Microbiology, Permoserstraße 15, D-04318 Leipzig, Germany
[b] Bioinformatics/High-Throughput Analysis, Faculty of Mathematics and Computer Science, Friedrich Schiller University Jena, Leutragraben 1, D-07743 Jena, Germany
[c] Bioinformatics Group, Department of Computer Science, and Interdisciplinary Center for Bioinformatics, Leipzig University, Härtelstraße 16-18, D-04107 Leipzig, Germany

ARTICLE INFO

Article history:
Received 18 July 2022
Received in revised form 14 November 2022
Accepted 14 November 2022
Available online 17 November 2022

Keyword:
microbial community flow cytometry
automated gating
cytometric data evaluation
automated cluster analysis
constrained Expectation–Maximization

ABSTRACT

Flow cytometry has become a powerful technology for studying microbial community dynamics and ecology. These dynamics are tracked over long periods of time based on two-parameter community fingerprints consisting of subsets of cell distributions with similar cell properties. These subsets are highlighted by cytometric gates which are assembled into a gate template. Gate templates then are used to compare samples over time or between sites. The template is usually created manually by the operator which is time consuming, prone to human error and dependent on human expertise. Manual gating thus lacks reproducibility, which in turn might impact ecological downstream analyses such as various diversity parameters, turnover and nestedness or stability measures. We present a new version of our `flowEMMi` algorithm – originally designed for an automated construction of a gate template, which now (i) generates non-overlapping elliptical gates within a few minutes. Gate templates (ii) can be created for both single measurements and time-series measurements, allowing immediate downstream data analyses and on-line evaluation. Furthermore, it is possible to (iii) adjust gate sizes to Gaussian distribution confidence levels. This automatic approach (iv) makes the gate template creation objective and reproducible. Moreover, it can (v) generate hierarchies of gates. `flowEMMi v2` is essential not only for exploratory studies, but also for routine monitoring and control of biotechnological processes. Therefore, `flowEMMi v2` bridges a crucial bottleneck between automated cell sample collection and processing, and automated flow cytometric measurement on the one hand as well as automated downstream statistical analysis on the other hand.

1. Introduction

In the past 10 years microbial communities have become a focus in medical and biotechnological research. They are known to have a major impact on human health [1], but they are also an indispensable building block in the circular economy, especially for the biotechnological production of basic and valuable platform chemicals [2,3], energy production [4,5] and the recovery of non-renewable resources such as phosphorus [6]. To study microbial communities in their composition and function a number of technologies are available. Most commonly, sequencing technologies are used to decipher either community composition based on the 16S rRNA gene or function through metagenomics to gain insight into the presence of functional genes. In addition, functions can be tracked using RNA-seq to study changes in gene expression over time in microbial communities or through proteomic approaches. All these technologies are based on relative data and, in general, are applied to whole microbial communities. Flow cytometry can provide not only structural and functional information about microbial communities on the single cell level, but also quantitative information by determining the number of cells per volume. In addition, high-throughput flow cytometry provides data in very short time intervals and is therefore perfect for tracking substructural changes in microbial community dynamics [7,8]. Based on light scattering and fluorescent labeling of cellular components of each cell, phenotypic cell states can be described. Light scattering, preferably forward scattering, correlates to cell size. Preferably all-cell DNA staining is used at a resolution such that the chromosome

* Corresponding author at: Bioinformatics/High-Throughput Analysis, Faculty of Mathematics and Computer Science, Friedrich Schiller University Jena, Leutragraben 1, D-07743 Jena, Germany

E-mail addresses: carmen.bruckmann@ufz.de (C. Bruckmann), susann.mueller@ufz.de (S. Müller), christian.hoener.zu.siederdissen@uni-jena.de (C. Höner zu Siederdissen).

https://doi.org/10.1016/j.csbj.2022.11.033

number per cell can be determined. These two parameters, together with the changes in community composition and function over time, allow us to understand and sometimes even control community behavior. Additional fluorescent markers can always be added, but because most members of natural communities are unknown and cannot be cultured, these markers often lack cell type specificity and are therefore not used for fingerprinting. Measuring only two cell features from 200,000 cells per sample is already sufficient to create a fingerprint that reflects the specific characteristics of a community at the time of sampling. Fingerprinting has proven reliable in many applications; it is cheap to obtain, reliable to use, easy to measure, and quick to evaluate. Many bioinformatics tools are available to provide information on community evolution, ecology, and function based on fingerprinting.

In order to evaluate dynamics in time-series of community fingerprints, cells of each sample need to be captured by a defined and universal cell gate comprising all cells and excluding instrumental noise, calibration beads and cell debris. Within the cell gate a gate template needs to be created which highlights subcommunities according to their appearance and their relative cell abundances. Commercially available software can be used such as FlowJo (www.flowjo.com, Ashland, Oregon), FCS Express (www.denovosoftware.com, Pasadena, California) and Summit (Dako Colorado Inc. Summit, Fort Collins, Colorado). A cell gate and a gate template must be defined for a whole time-series of an experiment. A gate template is typically created by hand, marks up to 100 subcommunities by non-overlapping elliptical gates and ideally includes any subcommunity that arises throughout the experiment. The manual creation of a meaningful gate template is time-consuming, error-prone, and dependent on the expert who creates the template. In the course of an experiment subcommunities may disappear and new ones may emerge. Therefore, not all gates are always occupied at every time-point and it is permissible for some gates to be empty if the corresponding subcommunity is not present in the particular sample.

The resulting cell abundances per time-point and gate provide the basic data for calculating the ecological behavior of the community, making the gate template a critical factor in deciphering both community structure and function in spatiotemporal experiments. The quality of the gate template influences all further downstream analyses. First, cell abundances per gate are collected over time and translated into structural community dynamics using the R-based tool flowCyBar [9]. The same gate-template-based data generation subsequently yields diversity measures for microbial communities such as α- and γ-diversity values as well as intra- and inter-community β-diversity values, also using R tools [10,11]. In addition, ecological measures of community variation such as turnover and nestedness are based on the correct setting of a gate template [12]. Because microbial communities can exhibit high stochasticity in different environments, the calculation of stability measures can become important in several applications [13,8]. Stability can be described by several properties, such as resistance and resilience [14,13] which may alter community structure and function after a disturbance. In the case of a wastewater community, this could diminish its ability to remove carbon, nitrogen and phosphorus, and in the case of a biotechnological production process, an unstable community will eventually reduce the yield of the desired product [15]. Stability measures using flow cytometric data also depend on precise gate templates. In addition, to reveal interactions indicative of specific cell functions, abiotic parameters such as changes in pH, temperature, ammonium or carbon content can be correlated with gate template-based cell abundances by flowCyBar [9]. Gates containing cells with functions of interest can be sorted by flow cytometry imposing an electrical charge on each cell-containing droplet, allowing sorting into different tubes. These cells can be further analyzed e.g., using omics technologies. The evaluation of spatiotemporal community behavior based on gate templates is routinely in use and contributes to deep understanding of ecology [11]. The outcome of gate template based evaluations (without cell sorting) can be provided within 2 h to 3 h after gate template setting. Therefore, its automated setting will bridge a critical bottleneck between automated sample collection and processing, flow cytometry, and automated downstream statistical analysis. As a result, the ecology of microbial community behavior can be studied and understood easily and quickly, opening a window to near-online controlled or manipulated microbial community handling in biotechnology or medicine.

In this study, we present a significantly expanded version of our algorithm for automatic gating of flow cytometric communities together with an updated implementation under the name `flowEMMi v2`. This version addresses a number of shortcomings in the original design and implementation. Our aim with this new algorithm is to guarantee non-overlapping elliptical gate templates. Furthermore, the old version of our algorithm was only able to handle single data sets. Now gate templates for complete time-series are needed, which are required to capture the dynamic behavior of communities over long time periods. As in the previous version, the new tool is based on the EM algorithm [16]. Several other gating approaches have been developed before, all of them are available as R packages. FlowFP divides the data into bins and rectangular regions by density [17] that however are biologically not meaningful, and will result in an unnecessary large amount of gates. Other automatic gating tools are also available, which include non-parametric and parametric methods. flowMeans [18] and SamSPECTRAL [19] are non-parametric methods that assign a classification-label to each cell. However, the actual gates in the form of a geometric object are not provided in these tools. Parametric methods like flowMerge [20], flowClust [21] and PhenoGMM [22] use mixture models of distributions for the underlying data. The probability distributions could in principle be used to define irregular non-overlapping polygons as gates. However, (similar to flowMeans and SamSPECTRAL) these polygons are not included in the output of flowMerge, flowClust and PhenoGMM. Therefore, cell-sorting could not be performed directly. If multiple samples are given, then PhenoGMM will combine them into one data set (with optional down-sampling), and a gate template will be created for this combined set. Apart from these applications of unsupervised machine learning there is also supervised machine learning, which requires a set of already gated samples as input. We need to point out here that this method is not applicable to our kind of data since every cell type, or in our case subcommunity, needs to be known to provide a set of training data in advance. Since cells in microbial communities are mostly not culturable or change their physiology during growth in communities, training sets are not available for most cell types. For human cells, however, such tools are available. The package optimalFlow [23] performs supervised machine learning, and is able to distinguish between human cell-types. Reiter et al. [24] also present a theoretical approach, where Gaussian Mixture Models are used. PhenoGMM comes most close to the capabilities of `flowEMMi v2`. For this reason we will discuss the advantages and disadvantages in more detail later. Furthermore, to our knowledge `flowEMMi v2` is the only tool that is able to create non-overlapping elliptical gates on microbial time-series data.

To summarize, in this paper, we formalize the goals of gating in flow cytometry and expand the `flowEMMi` [25] algorithm with the following features:

(i) Gates are always non-overlapping. This enables us to sort the cells later if needed, and near on-line data evaluation for the description of ecological community properties such

as diversity metrics, stability measures or trend analyses such as turnover and nestedness can be performed without bias.
(ii) The tool is not only able to gate single samples but it can also handle time-series of flow cytometry data via the creation of a gate template.
(iii) Each gate encloses a region of high cell count. The tightness of the enclosure can optionally be set by the user.
(iv) The gate template creation is objective and reproducible. This is crucial since the standard approach of manual gating is dependent on the user.
(v) Hierarchical gating should be possible to increase the resolution of cell subsets within a gate.
(vi) The performance of the algorithm is retained with respect to running time below typical bacterial generation times.

This paper is structured in the following way: Section 2 will give a brief insight into flow cytometry and the two cytometric data sets used for testing the tool. Furthermore, we will describe our new algorithm `flowEMMi v2` in simple words. Next, in Section 3 the mathematical details of our new algorithm will be explained step-wisely. In Section 4 we will show the outcome of `flowEMMi v2` on two original data sets. We will then discuss the results and give a comparison with other automated gating procedures in Section 5. In addition, we discuss possible extensions of `flowEMMi v2` to higher dimensions in this section. We conclude our work in Section 6.

2. Material and Methods

2.1. Flow cytometry

The flow cytometric test data used in this paper were obtained from two earlier studies [26,8]. In both cases a BD Influx v7 Cell Sorter (Becton, Dickinson and Company, Franklin Lakes, NJ, USA) was used. The instrument was equipped with a 488 nm Sapphire OPS laser (400 mW) and a 355 nm Genesis CX laser (100 mW, both Coherent, Santa Clara, CA, USA). The 488 nm laser light was used for the detection of the forward scatter (FSC, 488/10) and the side scatter (SSC, 488/10, trigger signal). While the forward scatter is related to the size of a cell, the side scatter measures cell granularity. DAPI (4',6-Diamidino-2-phenylindol) indicates chromosome numbers per cell, by binding to AT rich regions of the DNA and was measured at PMT9 (460/50) after excitation with 355 nm laser light. Per sample, 200,000 cells were characterized based on these parameters, resulting in typical fingerprints, called dot plot, for each sample.

In both studies the fluidic system was run at 33 psi using a 70 µm nozzle. The sheath fluid consisted of 0.5 x FACSFlow buffer (BD). In both studies, for the daily optical calibration of the cytometer in the linear range, 1 µm blue fluorescent FluoSpheres (Molecular Probes, F-8815, Eugene, OR, USA) and 2 µm yellow-green fluorescent FluoSpheres (ThermoFisher Scientific, F8827, Waltham, MA, USA) were used. For calibration in the logarithmic range, 0.5 µm UV Fluoresbrite Microspheres (Polysciences, 18339, Warrington, PA, USA) were used. Prior to measurement, DAPI stained cells were spiked with 0.5 and 1 µm UV Fluoresbrite Microspheres (both Polysciences, 18339 and 17458, Warrington, PA, USA). The microspheres served as internal standards to monitor instrument stability and to allow correct comparison of samples.

As beads were used for the calibration of the cytometer, they were excluded from any further analysis by a cell gate together with the instrumental noise and cell debris before a manual gate template was created using the software FlowJo [27] www.flowjo.com, Oregon, USA. In the dot plots cell-densities are highlighted by colors: Regions with a high virtual cell density are depicted red in this paper. The color scheme then follows orange, yellow, green and blue as the density decreases. The raw data for data set 1 (the cytometric mock communities) are available at FlowRepository [28] https://flowrepository.org/:, accession number FR-FCM-Z2CJ. The raw data for data set 2 (time-series data of various microbial communities) can also be accessed at FlowRepository by the accession number FR-FCM-ZYWX.

2.2. The workflow of `flowEMMi`

We now give a quick and simplified outline of the algorithm in `flowEMMi` [25]. Full mathematical details can be found in Section 3.1.

The tool `flowEMMi` operates on a set of data points obtained by the measurement of two physiological properties of a single cell. `flowEMMi` finds a set of overlapping ellipses (mathematically represented by a Gaussian mixture model) that fits the data points. This was implemented via the EM-algorithm [16]: First, the centers of the ellipses, their shape and their weight are drawn from a random distribution. The weight of an ellipse indicates its importance in the model. These weights can be thought of as the number of virtual cells lying inside the respective ellipse. Using the normal distribution underlying each ellipse we can calculate the probability of observing each data point for each ellipse. This results in a so-called membership weight matrix computed in the Expectation step (E-step) of the EM-algorithm. Next, in the maximization step (M-step) the weights, centers and shapes of the ellipses are updated based on the values of the E-step. For simplicity this can be imagined as the points pulling on the ellipses. Points that are close to an ellipse pull on it more strongly than points that are far away. The E-step and the M-step are performed alternately. This way the ellipses are shifted towards regions in the plot that have a high density of points. After every iteration the sizes of the ellipses are checked since very small ellipses might cause numerical issues. This can for example be the case if the technical noise and other noise, e.g. cell debris, was not cut out properly from the data set by the cell gate. The EM-algorithm stops when the likelihood of the model does not improve significantly anymore. For reasons of running time efficiency `flowEMMi` starts the EM-algorithm on a subset of the data-points. Since the number of ellipses needed for the data set is not known beforehand, `flowEMMi` executes the EM-algorithm several times on this subset for a range of numbers of ellipses. Next, the best model on the subset of data is chosen according to the Bayesian information criterion (BIC), which in turn takes into account the likelihood of the model and the number of ellipses. This is useful to prevent over-fitting. Now, K is defined as the number of ellipses that were used in the best model. The models with a similar number of ellipses, e.g. between $K-2$ and $K+2$, are also of interest for modeling the whole data set. The EM-algorithm is then again performed on the whole data set, this time starting with the models that were evaluated on the subset of data instead of a random distribution. In the end, the best model on the whole data set is again chosen according to the BIC.

2.3. Operations on ellipses in `flowEMMi v2`

In `flowEMMi v2` there are three main operations which are designed to solve the problem of overlapping ellipses; namely deletion, merging and shrinking.

Deletion: Large ellipses that (partly) contain smaller ellipses are deleted. They mostly do not provide additional information and also contain cells which should lie in the background. Those large ellipses are not a good gate for a gate template according to goals (i) "gates should not overlap" and (iii) "each gate encloses a region

of high cell count". Therefore, the ellipses which overlap with at least two more ellipses by more than a given threshold will be deleted. An illustration of this scenario is given in Fig. 2A. For more mathematical details see Section 3.5.

Merging: After checking for ellipses that need to be deleted in `flowEMMi v2` the set of ellipses can still overlap and thus contradict goal (i) "gates should not overlap". If the overlapping area relative to the size of the two overlapping ellipses is large, then they should be merged, that is, they are combined into one new ellipse that covers the data points which lie in the overlap area. Note that ellipses (geometric) can be transformed to bi-variate Gaussian distributions and vice versa. Since the new ellipse should account for the virtual cells in both ellipses, the Gaussian distribution of this new ellipse needs to reflect the probability of a data point to belong to one ellipse and also to the other one. Therefore, the two original Gaussian distributions are multiplied and scaled by the weights of their ellipses. The new ellipse will be more similar to the original ellipse with the higher weight. In the end of a merging step the two original ellipses are replaced by the new one in the mixture model. An illustration of merging is given at the end of Section 3.7. Additionally, an example is given in the supplementary files, see supplemental Fig. S1. For more mathematical details see Section 3.6.

Shrinking: Another important process used in `flowEMMi v2` is shrinking, which is used when the overlap of two ellipses is too small for merging. In this case both ellipses are shrunk, i.e. made smaller. The centers of both ellipses remain at their positions while the axes of both ellipses are reduced. In order not to reduce them too much the shrinking depends on the size and position of the overlap area. That way goal (iii) "each gate encloses a region of high cell count", is pursued. Moreover, the more important ellipse, i.e. the one with the highest weight will be reduced less. For mathematical details see Section 3.7. After one step of shrinking the two ellipses might still overlap. If they are still big enough, they will be shrunk further. Otherwise, the two original ellipses will be merged instead. An illustration of shrinking is given at the end of Section 3.7. Additionally, an example is given in the supplementary files, see supplemental Fig. S2.

2.4. The workflow of *flowEMMi v2*

The backbone of `flowEMMi v2` is a heavily modified version of the `flowEMMi` algorithm, and thus the EM-algorithm. After a certain number of EM-steps, e.g. in every 20th step, the EM-algorithm is paused. At this point the model consists of overlapping ellipses. These overlaps need to be removed. We point out that all operations modify the model without requiring the underlying set of data points, which in practice means that the additional running time costs due to our modified algorithm are negligible which thus contributes to goal (vi) "running time below typical bacterial generation times".

A visualization of the overlap removal implemented in `flowEMMi v2` can be found in Fig. 1. First, it is checked whether there is one ellipse that overlaps with at least two or more ellipses by at least a given threshold (called t_1 in Section 3.8). These large ellipses are deleted. Next, pairs of ellipses for which both centers lie inside the intersection area will be merged. Then, the algorithm proceeds with the pair of ellipses that overlap the most. If the relative overlap meets the threshold for merging (called t_2 in Section 3.8), they will be merged, and the algorithm proceeds with the next pair of ellipses that overlaps the most. If the two original ellipses do have an overlap, which however is not large enough for merging, they are treated by the shrinking procedure. After every step of shrinking it is checked whether the two ellipses are still large enough, that is, if both their minor axes have at least a certain size to prevent degenerate ellipses. When violated, the original pair of overlapping ellipses is merged instead. Otherwise, and if the two shrunk ellipses still overlap, they are shrunk again. If they do not overlap anymore, the algorithm continues with the next pair of ellipses that overlap the most, until there are no overlaps anymore. These steps are repeated again and again until the final gates are ready to evaluate the data. Two examples of just the deletion, merging and shrinking steps are presented step by step in Fig. S3 and Fig. S4 for two different samples coming from a reactor experiment [8], which we will be using also in the following as a time-series analysis example.

2.5. Putting it all together: gate templates

The methods above describe how `flowEMMi v2` creates gates in a single sample. For time-series of flow cytometry data a gate template is needed which should approximately fit every sample and capture every upcoming subcommunity as formulated in goal (ii). A gate template can be created by first running `flowEMMi v2` on each single sample. The function for creating gate templates in `flowEMMi v2` takes the sets of ellipses per sample as input. When they are combined into one set we again have overlaps. Since the overall number of ellipses in a time-series can be very large and the calculation of overlap areas is done for every pair of ellipses, we cannot simply run the overlap removal for all the ellipses at once. Therefore, the set of all ellipses is partitioned and the overlaps are removed in every partition. Then, we build pairs of these partitions, that is, the now non-overlapping ellipses of one partition are combined with the non-overlapping ellipses of another partition. However, this combination of ellipses can be overlapping again. The algorithm continues to remove the overlaps in the pairs of partitions and combines the results into new pairs again until all the original partitions are combined into one non-overlapping set of ellipses. This is implemented in a recursive procedure.

2.6. Tools

Automated gating with `flowEMMi v2` can be applied to flow cytometry data measured on any device. The raw data should be read into R by the function read.FCS of the package flowCore [29] without any transformation. Via this method the user makes sure to input the data in a way that their scale reflects the subcommunities in their typical elliptical form. After reading in the data, the resulting object in R contains metadata (like the day of measurement, the device, etc.) as well as the actual optical measurements. Therefore, for reduction of information, the two parameters of interest need to be extracted before the automated gating is performed.`flowEMMi v2` builds upon a number of R-packages [30]. Rcpp [31–33] and RcppEigen [34] were used for foreign function calls to the numerical core library of `flowEMMi` written in C++. mvtnorm [35,36] provided a function to calculate the likelihood of a point under a multivariate Gaussian distribution. gtools [37] was used to draw values from a Dirichlet distribution in the initialization step. It was also used to randomize the order of the ellipses that provided the input for the gate template. For plotting the gated results the R-packages colortools [38], mixtools [39], gplots [40], ggplot2 [41] and KernSmooth [42] were required. For conventional plots (like e.g., in FlowJo or Summit) the result can be given on a logarithmic scale. For evaluating the running time tictoc [43] was used. For the NMDS-plots additionally vegan [44] was used to calculate the position of the samples within the plot. The manual gating procedure was done with the help of FlowJO [27]. Finally, flowWorkspace [45] was used to import the manual gating workspace into R in order to plot it the same way as the results of `flowEMMi v2`.

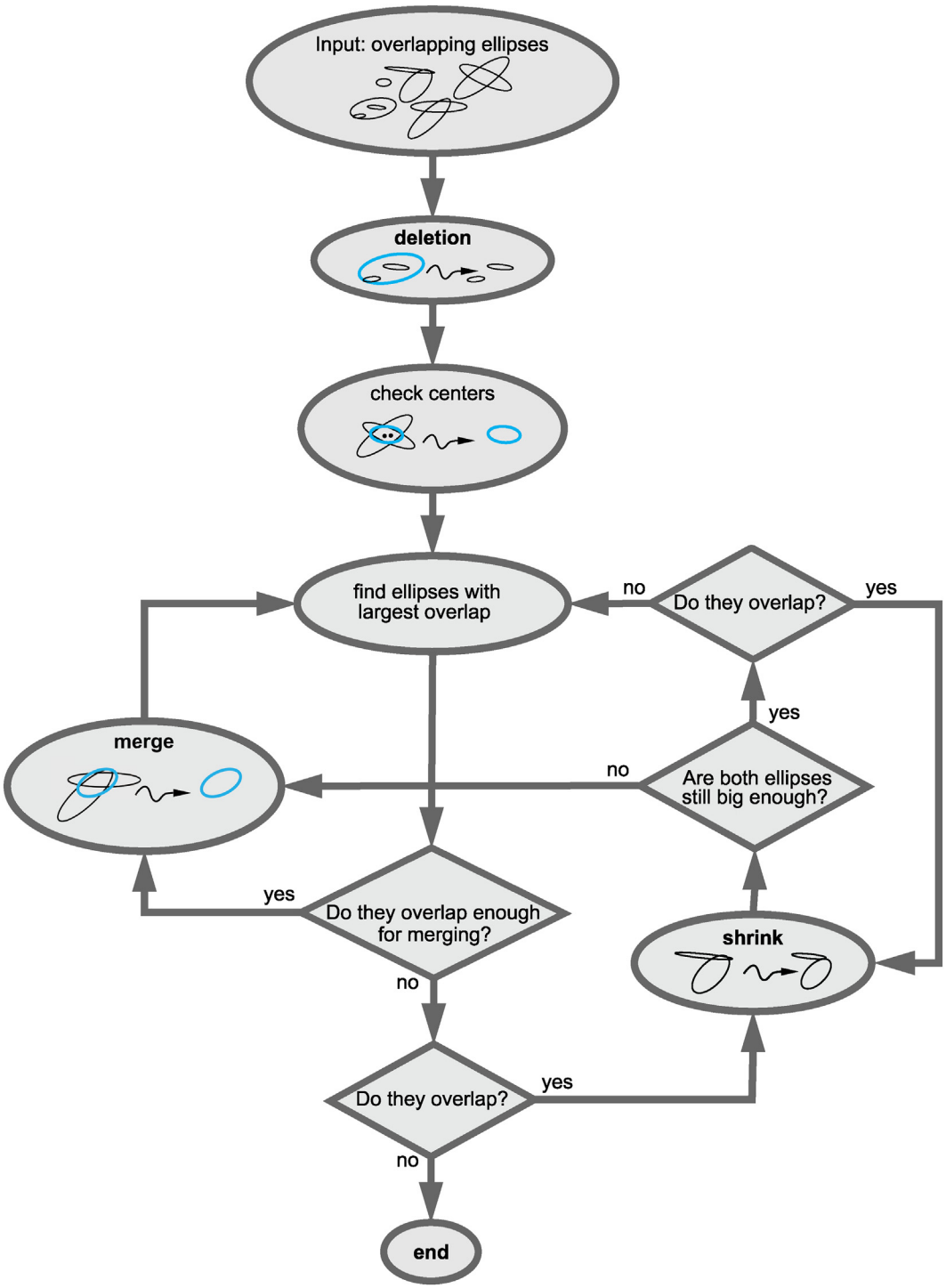

Fig. 1. Visualized workflow of the overlap removal in `flowEMMi v2`.

3. Theory

3.1. The EM-algorithm

As input we are given a set of N points $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$. Each x_i is a single measurement of two physiological properties of a cell. A gate template is represented as a mixture of 2-dimensional Gaussians. A finite mixture model $\theta = \{\theta_1(\pi_1, \mu_1, \Sigma_1), \ldots, \theta_K(\pi_K, \mu_K, \Sigma_K)\} \in \Theta$ consists of normal distributions $\mathcal{N}(\mu_k, \Sigma_k)$, and corresponding mixture weights $\pi_k \in \mathbb{R}^+$ with $k \in \{1, \ldots, K\}$ and $\sum_{k=1}^{K} \pi_k = 1$. Given $\mathcal{X}$ one wants to infer θ. A locally optimal θ can be found using the Expectation–Maximization (EM) algorithm [16], which iteratively improves on θ until further steps yield improvements that are negligible. The posterior for a single $x \in \mathcal{X}$ is given by

$$P(\theta|x) \propto P(\theta) \cdot \mathcal{L}(x|\theta) = P(\theta) \cdot \sum_{k=1}^{k=K} \pi_k \cdot P(x|\theta_k)$$
$$= P(\theta) \cdot \sum_{k=1}^{k=K} \pi_k \cdot \mathcal{N}(x|\mu_k, \Sigma_k). \quad (1)$$

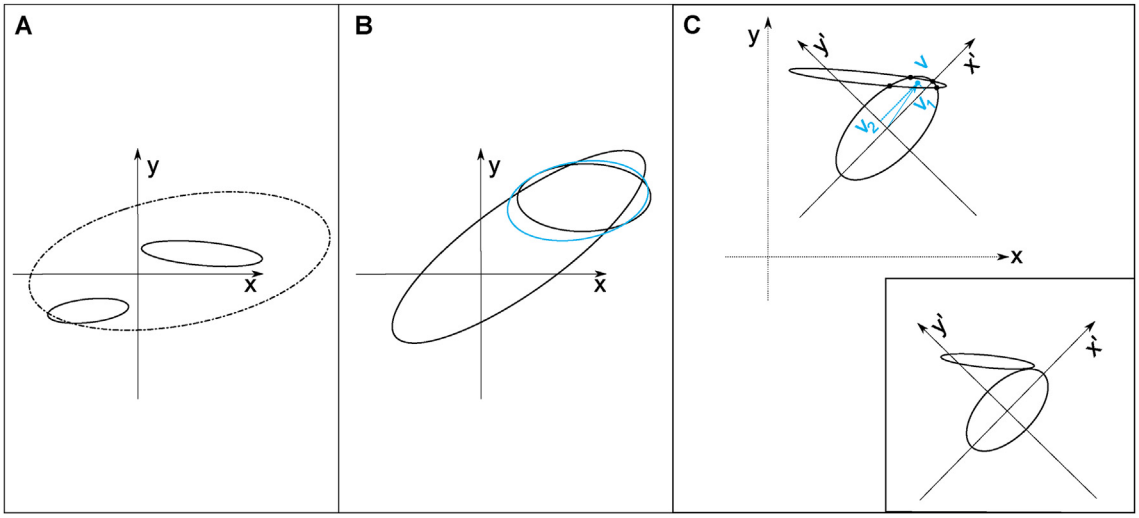

Fig. 2. Illustrations for the three operations during overlap removal. **A:** Deletion. In case an ellipse (dashed-dotted line) overlaps with at least two more ellipses by more than a given threshold, then it is deleted from the mixture model. **B:** Merging. If two ellipses (in this case the black ones) are selected by the algorithm to be merged, then they will be replaced by an ellipse (blue) that covers the overlap area and thus represents the product of the two underlying density distributions scaled by their weight. **C:** Shrinking. In the process of shrinking both overlapping ellipses get smaller. This reduction of the ellipses depends on their weight in the mixture model and on the position of the intersection points. For more details see Sections 3.5, 3.6 and 3.7, respectively.

The prior $P(\theta)$ captures the penalty for a given number of distributions $K = |\theta|$. The full posterior for the complete set $\mathscr{X}$ is

$$P(\theta|\mathscr{X}) \propto P(K = |\theta|) \cdot \mathscr{L}(\mathscr{X}|\theta) = P(K = |\theta|) \cdot \prod_{x \in \mathscr{X}} \mathscr{L}(x|\theta). \tag{2}$$

In general finding a global optimum for Eq. 2 is costly as it requires finding both, the number K of components and the optimal shape, position, and weight of the mixture components [16] for a non-convex function. Finding local optima for Eq. 2 is done via the Expectation–Maximization (EM) [16] algorithm. First, the prior on the number of mixture components is replaced by an exhaustive search over a small set of candidate values. The penalty for a given K is given by the Bayesian information criterion (BIC) [46], which favors smaller K unless offset by a correspondingly larger gain in log-likelihood. The penalty is based on the number of free parameters per mixture component. For the weight we have 1, for the mean 2, and for the covariance matrix 3 free parameters, and thus there is a total of $c = 6$ free parameters per mixture component. The BIC penalty then is $-0.5 \cdot c \cdot K \cdot \log|\mathscr{X}| = -3 \cdot K \cdot \log|\mathscr{X}|$. Note that this is rewritten compared to Eq. 4 in [25] to allow for the maximization of $\log P(\theta, \mathscr{X})$ under this penalty. With the prior on the number of components K taken care of, we now give a quick overview of the EM steps, assuming a fixed K.

The EM algorithm alternates between computing a weight matrix of assumed responsibilities $w \in \mathbb{R}^{|\mathscr{X}| \times K}$ and the Gaussian mixture θ.

Instead of maximizing the likelihood in Eq. 2 directly we can also maximize the log-likelihood $\log(\mathscr{L}(\mathscr{X}|\theta))$ since the log-function is strictly monotonously increasing and the resulting function is numerically more stable.

Therefore, we have

$$\begin{aligned}\log(\mathscr{L}(\mathscr{X}|\theta)) &= \log(\prod_{i=1}^{N}\sum_{k=1}^{K}\pi_k \cdot \mathscr{N}(x_i|\mu_k, \Sigma_k)) \\ &= \sum_{i=1}^{N}\log(\sum_{k=1}^{K}\pi_k \cdot \mathscr{N}(x_i|\mu_k, \Sigma_k)).\end{aligned} \tag{3}$$

In the initial step each $\theta_k = \mathscr{N}(\mu_k, \Sigma_k)$ is drawn randomly. First, for a given $x_i \in \mathscr{X}$ we calculate the membership weight w_{ik} for each normal distribution θ_k. The membership weights reflect our uncertainty, given x_i and θ about which of the K normal distributions generated vector x_i, without any "mixing" in the generative process. These weights are computed in the E-step of the algorithm.

$$w_{ik} = \frac{\pi_k \cdot P(x_i|\theta_k)}{\sum_{l=1}^{K}\pi_l \cdot P(x_i|\theta_l)} = \frac{\pi_k \cdot \mathscr{N}(x_i|\mu_k, \Sigma_k)}{\sum_{l=1}^{K}\pi_l \cdot \mathscr{N}(x_i|\mu_l, \Sigma_l)} \tag{4}$$

In the M-step of the algorithm the mixture weights π_k and the normal distributions $\mathscr{N}(\mu_k, \Sigma_k)$ are updated: The new μ_k are set to the weighted average of the x_i. Recall that variance is defined as the (weighted) average of the squared distance from each point of a sample to the mean of the sample. Therefore, the new weights, means, and covariance matrices can be calculated as follows:

$$\pi_k^{\text{new}} = \frac{\sum_{i=1}^{N} w_{ik}}{N}, 1 \leqslant k \leqslant K \tag{5}$$

$$\mu_k^{\text{new}} = \frac{1}{\sum_{i=1}^{N} w_{ik}} \cdot \sum_{i=1}^{N} w_{ik} \cdot x_i, 1 \leqslant k \leqslant K \tag{6}$$

$$\Sigma_k^{\text{new}} = \sum_{i=1}^{N} \frac{w_{ik}}{\sum_{j=1}^{N} w_{jk}} \cdot (x_i - \mu_k^{\text{new}}) \cdot (x_i - \mu_k^{\text{new}})^T, 1 \leqslant k \leqslant K \tag{7}$$

Now, the E-step followed by the M-step will be executed iteratively until the log-likelihood does not improve significantly anymore, that is until $\log(P(\mathscr{X}|\theta^s)) - \log(P(\mathscr{X}|\theta^{s+1})) \leqslant \epsilon$ for some step s and some pre-defined ϵ. The likelihood increases in each iteration, but the EM-algorithm can get stuck in local optima. In `flowEMMi` the EM-algorithm is performed for a given range of K and for different initial random start distributions. In the end, the model θ with the best BIC is chosen.

In every step of the EM-algorithm it is checked whether each covariance matrix Σ satisfies the minimum size constraints of all semi-minor axes given by the user. If an ellipse is too small, then

the respective semi-minor axis is set to the minimum size and the covariance matrix Σ is adapted accordingly.

3.2. Multivariate Gaussian distributions

A 2-dimensional real random Variable V is said to be normally distributed with 2-dimensional mean vector μ and 2×2- covariance matrix Σ if it has a density

$$f_V(v) = \frac{1}{\sqrt{(2\pi)^2 \det(\Sigma)}} \cdot \exp\left(-\frac{1}{2}(v-\mu)^T \Sigma^{-1}(v-\mu)\right) \tag{8}$$

This will be denoted by $V \sim \mathcal{N}(\mu, \Sigma)$. Alternatively, V can be defined as multivariate normally distributed if every (univariate) linear combination of its components is normally distributed. Note that, therefore, each of the univariate components of V have to be normally distributed. We make use of Gaussians – and hence ellipses – since they are a common model for cell gates and are computationally convenient as well. Not every cell type has the same size or DNA-content (due to the cell-cycle), but cell types can be well-described by normal distributions. In this work we always have two dimensions; i.e. $V = (X, Y), \mu = (\mu_x, \mu_y)$ and $\Sigma = \begin{pmatrix} \mathrm{Var}(X) & \rho \cdot \sqrt{\mathrm{Var}(X)} \cdot \sqrt{\mathrm{Var}(Y)} \\ \rho \cdot \sqrt{\mathrm{Var}(X)} \cdot \sqrt{\mathrm{Var}(Y)} & \mathrm{Var}(Y) \end{pmatrix}$, with $\mathrm{Var}(X)$ and $\mathrm{Var}(Y)$ being the variances of the random variables X and Y, respectively and $\rho \in [-1, 1]$ being their correlation coefficient. The axes of the corresponding ellipse are located in the direction of the eigenvectors of Σ.

3.3. Ellipses

Any 2-dimensional Gaussian distribution can be visualized by an ellipse. Let us consider an Euclidian space with dimensions x and y. An ellipse ℓ can be defined by four properties: It has a center $\mu = (\mu_x, \mu_y)$, a semi-major axis of length a, a semi-minor axis of length b and an angle γ to the x-axis. The area of ℓ, denoted by $|\ell|$ equals $a \cdot b \cdot \pi$. In order to transform a Gaussian distribution $D = (\mu, \Sigma)$ into an ellipse, we need a confidence level $\alpha \in (0, 1)$ giving the proportion of density that will be covered by the ellipse. Let λ_1 and λ_2 be the eigenvalues of Σ with λ_1 being the larger one, and let $e_1 = (e_{1,x}, e_{1,y})$ be the eigenvector that belongs to λ_1. Then [47],

$$\begin{aligned} s &= -2 \cdot \log_e(1-\alpha) \\ a &= \sqrt{s \cdot \lambda_1}, b = \sqrt{s \cdot \lambda_2} \\ \gamma &= \tan^{-1}\left(\tfrac{e_{1,y}}{e_{1,x}}\right) \end{aligned} \tag{9}$$

Conversely, an ellipse can be transformed back into the Gaussian distribution $D = (\mu, \Sigma)$ with $\Sigma = \begin{pmatrix} \mathrm{Var}(X) & \mathrm{Cov}(X,Y) \\ \mathrm{Cov}(X,Y) & \mathrm{Var}(Y) \end{pmatrix}$, with X and Y being the random variables used in D and confidence level α:

$$\begin{aligned} s &= -2 \cdot \log_e(1-\alpha) \\ \mathrm{Var}(X) &= \left(a^2 \cdot \cos(\gamma)^2 + b^2 \cdot \sin(\gamma)^2\right)/s \\ \mathrm{Var}(Y) &= \left(a^2 \cdot \sin(\gamma)^2 + b^2 \cdot \cos(\gamma)^2\right)/s \\ \mathrm{Cov}(X,Y) &= \left((a^2 - b^2)\sin(\gamma) \cdot \cos(\gamma)\right)/s \end{aligned} \tag{10}$$

Moreover, every ellipse can be written as an equation as follows [48]:

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0, \quad \text{with} \quad B^2 - 4AC < 0 \tag{11}$$

$$\begin{aligned} A &= a^2 \cdot \sin(\gamma)^2 + b^2 \cdot \cos(\gamma)^2 & D &= -2A\mu_x - B\mu_y \\ B &= 2(b^2 - a^2) \cdot \sin(\gamma) \cdot \cos(\gamma) & E &= -B\mu_x - 2C\mu_y \\ C &= a^2 \cdot \cos(\gamma)^2 + b^2 \cdot \sin(\gamma)^2 & F &= A\mu_x^2 + B\mu_x\mu_y + C\mu_x^2 - a^2b^2 \end{aligned} \tag{12}$$

Every point $p = (x, y)$ that fulfills Eq. 11 lies on the respective ellipse ℓ. If p lies inside ℓ, then this term will be < 0, and if p lies outside ℓ, then this term will be > 0. Therefore, Eq. 11 discriminates between points that lie inside, on and outside the respective ellipse. This information is not only crucial in the computation of the overlap area of two given ellipses but also when the cell numbers per gate are calculated in order to reveal the community dynamics.

3.4. Ellipse intersection points

A pair of ellipses can have 1, 2, 3 or 4 intersection points. Given two ellipses ℓ_1 and ℓ_2 with centers μ_1 and μ_2, we arbitrarily translate, then rotate the first ellipse such as to have $\mu_1 = (0, 0)$ and $\gamma_1 = 0$. Now we can define all points on the border of the ellipse with the rational representation (where $-\infty < u < \infty$) [48]:

$$\begin{aligned} x(u) &= a\tfrac{1-u^2}{1+u^2} \\ y(u) &= \tfrac{2bu}{1+u^2} \end{aligned} \tag{13}$$

We can now substitute x, y in Eq. 11 with those from Eq. 13 which gives an equation of fourth order:

$$\begin{aligned} &A\left(a\frac{1-u^2}{1+u^2}\right)^2 + Ba\frac{1-u^2}{1+u^2} \cdot \frac{2bu}{1+u^2} + C\left(\frac{2bu}{1+u^2}\right)^2 + Da\frac{1-u^2}{1+u^2} \\ &\quad + E\frac{2bu}{1+u^2} + F = 0 \end{aligned} \tag{14}$$

Now we multiply the whole equation by $(1+u^2)^2$:

$$\begin{aligned} &A(a(1-u^2))^2 + Ba(1-u^2)2bu + C(2bu)^2 + Da(1-u^2)(1+u^2) \\ &\quad + E2bu(1+u^2) + F(1+u^2)^2 = 0 \\ &\Rightarrow a^2A(1 - 2u^2 + u^4) + aB(2bu - 2bu^3) + 4b^2Cu^2 + aD(1-u^4) \\ &\quad + E(2bu + 2bu^3) + F(1 + 2u^2 + u^4) = 0 \\ &\Rightarrow a^2A - 2a^2Au^2 + a^2Au^4 + 2abBu - 2abBu^3 + 4b^2Cu^2 \\ &\quad + aD - aDu^4 + 2bEu + 2bEu^3 + F + 2Fu^2 + Fu^4 = 0 \\ &\Rightarrow u^0(a^2A + aD + F) + u^1(2abB + 2bE) + u^2(-2a^2A + 4b^2C + 2F) \\ &+ u^3(2bE - 2abB) + u^4(a^2A - aD + F) = 0 \end{aligned} \tag{15}$$

Using polyroot of the base package in R [30] we can now get the (four) roots of this polynomial. We only take the (nearly) real roots and substitute those back into Eq. 13 to get the intersection points in the space where the first ellipse is centered at $(0, 0)$ and aligned with the coordinate axes. The intersection points can easily be transformed back to the original space. These intersection points are used in the process of shrinking, see Section 3.7 and also in the calculation of the overlap area $|\ell_1 \cap \ell_2|$ of the two given ellipses, see section S4.1, which was done as described in [49]. In practice we are not only interested in the absolute overlap area $|\ell_1 \cap \ell_2|$, but in the relative area $\frac{|\ell_1 \cap \ell_2|}{\min(|\ell_1|, |\ell_2|)}$ which is meant by overlap in the following.

3.5. Deletion of ellipses in `flowEMMi v2`

If there is an ellipse ℓ_1 that overlaps with at least two more ellipses ℓ_2 and ℓ_3 by more than a given threshold t_1; i.e.

$$\frac{|\ell_1 \cap \ell_2|}{\min(|\ell_1|, |\ell_2|)} \geqslant t_1 \text{and} \frac{|\ell_1 \cap \ell_3|}{\min(|\ell_1|, |\ell_3|)} \geqslant t_1, \tag{16}$$

then delete ℓ_1. An illustration of this scenario is given in Fig. 2A.

3.6. Merging in `flowEMMi v2`

In case two ellipses ℓ_1 and ℓ_2 (with centers μ_1 and μ_2, covariance matrices Σ_1 and Σ_2 and weights π_1 and π_2) overlap more than a given threshold or if they would get too small when the shrinking-step (Section 3.7) is applied, they need to be merged into a new ellipse ℓ_c. In this case, we assume that ℓ_1 and ℓ_2 cover the same subcommunity of the underlying sample. The combining ellipse ℓ_c should capture the overlapping area of ℓ_1 and ℓ_2. Additionally, ℓ_c depends on the weights π_1 and π_2 of ℓ_1 and ℓ_2, respectively. That is, ℓ_c will be more similar to the ellipse with the highest weight. This merging is the product of two Gaussian densities, since we want the probability distribution of points lying in *both* ellipses. Since the covariance matrices Σ_1 and Σ_2 of ℓ_1 and ℓ_2 are both symmetric and positive definite, we can use the Cholesky decomposition and re-scale them by dividing the diagonal elements as follows:

$$\Sigma_i = L_i \cdot D_i \cdot L_i^T \text{ for } i = 1,2 \tag{17}$$

$$\Sigma_i\prime = L_i \cdot \frac{1}{w_i} \cdot D_i \cdot L_i^T \text{ for } i = 1,2 \tag{18}$$

Then, the new μ_c and $\Sigma_c\prime$ of $\ell_c\prime$, which will be re-scaled to ℓ_c, can be derived as the product of two Gaussians [50] p. 42:

$$\mu_c = \left(\Sigma_1'^{-1} + \Sigma_2'^{-1}\right) \cdot \left(\Sigma_1'^{-1} \cdot \mu_1 + \Sigma_2'^{-1} \cdot \mu_2\right) \tag{19}$$

$$\Sigma_c\prime = \left(\Sigma_1'^{-1} + \Sigma_2'^{-1}\right)^{-1} \tag{20}$$

We use $\Sigma_c\prime = \left(\Sigma_1'^{-1} + \Sigma_2'^{-1}\right)^{-1} = \Sigma_1\prime \cdot (\Sigma_1\prime + \Sigma_2\prime)^{-1} \cdot \Sigma_2\prime$ for numerical stability [51] p. 16. For the re-adjustment of the weights, we again need to decompose $\Sigma_c\prime$ and multiply the diagonal elements by the sum of the weights.

$$\Sigma_c\prime = L_c\prime \cdot D_c\prime \cdot L_c'^T \tag{21}$$

$$\Sigma_c = L_c\prime \cdot (\pi_1 + \pi_2) \cdot D_c\prime \cdot L_c'^T \tag{22}$$

In the end, the original ellipses ℓ_1 and ℓ_2 will be deleted and replaced by ℓ_c, i.e. by μ_c and Σ_c. An illustration of merging is given in Fig. 2B. Additionally, an example is given in the supplementary files, see supplemental Fig. S1.

3.7. Shrinking in `flowEMMi v2`

If a pair of ellipses ℓ_1 and ℓ_2 does not overlap enough to be merged, then they will be shrunk. In this case ℓ_1 and ℓ_2 are likely to cover two different sub-populations of the underlying sample. Note that the following procedure does not change the centers of ℓ_1 and ℓ_2. This is sensible since the centers will already be situated at the areas of highest density of the input data. Let ℓ_i be represented by the length of the semi-major axis a_i, length of semi-minor axis b_i, center μ_i and angle γ_i. Let $|\ell_1| = a_1 \cdot b_1 \cdot \pi$ and $|\ell_2| = a_2 \cdot b_2 \cdot \pi$ denote[1] the sizes of E_1 and E_2, respectively and let o be the size of the overlap area of ℓ_1 and ℓ_2. Optionally, the first step is to reduce the angle between the given ellipses by a certain amount. This is often done in manual gating of time-series data because it naturally reduces the overlap area. Now, each ellipse is shrunk individually. To this end ℓ_1 is transformed such that its center now lies at $(0,0)$ and its major axis is aligned with the x-axis. Consequently its minor axis is aligned with the y-axis. The same transformation is applied to ellipse ℓ_2 such that their relative position and thereby their overlap area stays the same. Next, the intersection points of the transformed ellipses are determined as described in Section 3.4. Let the vector $v = (v_1, v_2)$ be the average of all intersection points. The vector v can be calculated by the individual average of the intersection points in each dimension. Let $f_1 = \frac{\pi_2}{\pi_1+\pi_2}$ and $f_2 = \frac{\pi_1}{\pi_1+\pi_2}$ be the ratios of the weights given by the mixture model. Then, the reduction factors for ℓ_1 are calculated as follows:

$$\begin{aligned} r_{\text{major}} &= \sqrt{\frac{o}{|\ell_1|} \cdot \frac{|v_1|}{|v_1|+|v_2|} \cdot f_1} \\ r_{\text{minor}} &= \sqrt{\frac{o}{|\ell_1|} \cdot \frac{|v_2|}{|v_1|+|v_2|} \cdot f_1} \end{aligned} \tag{23}$$

Since $|\ell_1| = a_1 \cdot b_1 \cdot \pi \leqslant a_1^2 \cdot \pi$ the size of an ellipse can be seen as a quadratic function. To compensate for that the square-root is taken in Eq. 23. In case the size of the overlap area o is big, then also the reduction of the axis of ℓ_1 will be big. If the average of the intersection points, observed in the transformed space, is more in the direction of the major axis, then the major axis will be reduced more. Otherwise, the minor axis will be reduced more. By this application of weights for the reduction of the ellipse axis we try to minimize the loss of area covered by ℓ_1. Furthermore, if the second ellipse ℓ_2 has a high weight, then f_1 will be high as well, and thus ℓ_1 will be shrunk more. The new length of the major semi-axis $a_1\prime = a_1 \cdot (1 - r_{\text{major}})$ and the minor semi-axis $b_1\prime = b_1 \cdot (1 - r_{\text{minor}})$ of the new ellipse $\ell_1\prime$ will be calculated next. The same process is then repeated for ℓ_2. That is, both ellipses are transformed such that the center of ℓ_2 now lies at $(0,0)$ and its major axis is aligned with the x-axis. Then, also ℓ_2 will be shrunk by the respective factors as in Eq. 23, and the overlap of $\ell_1\prime$ and $\ell_2\prime$ is calculated. Note that the centers stay the same, ensuring that they still cover the areas of high density. Fig. 2C illustrates the shrinking process. Additionally, an example is given in the supplementary files, see supplemental Fig. S2. To prevent endless recursion, shrinking is stopped when the ellipses only overlap by a marginal amount; i.e. the fraction of the overlap area divided by the area of the smaller ellipse is less or equal to 10^{-4}. In the process of shrinking it is always ensured that both the minor axes $b_1\prime$ and $b_2\prime$ stay larger than a given threshold. If this is not fulfilled at some point of the shrinking procedure, then the original ℓ_1 and ℓ_2 will be merged to prevent the ellipses from covering too few cells. If none of the two termination criterions are fulfilled yet, $\ell_1\prime$ and $\ell_2\prime$ will be shrunk further recursively.

3.8. Workflow of `flowEMMi v2`

An overview of the workflow of `flowEMMi v2` is provided in Section 2.4. Pseudocode for the full algorithm is given in Alg. 1, with lines 1–7, 24 corresponding to `flowEMMi` [25]. `flowEMMi v2` is a heavily modified and expanded version of `flowEMMi`. In particular, intermediate EM steps are interleaved with with constraint-resolving steps. This means that while the algorithm is iteratively closing in on a local optimum it will adapt to non-overlap conditions by transforming, merging, and deleting ellipses as necessary.

This corresponds to lines 8–23 in Alg. 1. We point out that all operations modify the model θ without requiring the data set $\mathcal{X}$, which in practice means that the additional running time costs due to our modified algorithm are negligible which thus contributes to goal (vi) "running time below typical bacterial generation times".

[1] $\pi \approx 3.14$, and *not* a mixture component weight here.

Algorithm 1: The `flowEMMi v2` algorithm, adapted to compute non-overlapping elliptical gates. Parameters of the algorithm are the number K of initial gates, the step size m between merging operations, the confidence level α (which determines the gate sizes), the minimum size of the minor axes $b^{\min}$, and the ratios t_1, t_2 at which partially overlapping gates are shrunk (t_1) or merged (t_2). Finally the algorithm requires the input data $\mathcal{X}$. Lines 8–23 show the modifications necessary to allow for shrinking and merging operations compared to the original algorithm.

1: $\theta = \{\theta_1 = (\pi_1\mu_1, \Sigma_1), \ldots, \theta_K = (\pi_K, \mu_K, \Sigma_K)\} \leftarrow K$ uniformly sampled starting points
2: $c \leftarrow 0$
3: **repeat**
4: $\quad c \leftarrow c + 1$ ▷ Step counter
5: $\quad L^{\text{old}} \leftarrow \text{logLikelihood}\,(\theta, \mathcal{X})$
6: $\quad \theta \leftarrow \text{EM}(\theta, \mathcal{X})$ ▷ Perform a single EM-step
7: $\quad L^{\text{new}} \leftarrow \text{logLikelihood}\,(\theta, \mathcal{X})$
8: $\quad$ **if** $c \bmod m = 0$ **or** $L^{\text{new}} - L^{\text{old}} < \varepsilon$ **then** ▷ every m'th step and after final iteration
9: $\quad\quad$ **for** $\theta_i \in \theta$ **do**
10: $\quad\quad\quad$ **if** $\text{overlap}(\alpha, \theta_i, \theta_j) > t_1$ **and** $\text{overlap}(\alpha, \theta_i, \theta_k) > t_1$ with $\theta_j, \theta_k \in \theta \setminus \{\theta_i\}$ **then**
11: $\quad\quad\quad\quad \theta \leftarrow \theta \setminus \{\theta_i\}$
12: $\quad\quad$ **while** $\exists \theta_i, \theta_{j\neq i} \in \theta$ with $\text{overlap}(\alpha, \theta_i, \theta_j) > 0$, sorted by descending relative area of overlap **do**
13: ▷ check all ordered pairs
14: $\quad\quad\quad$ **if** $\text{overlap}(\alpha, \theta_i, \theta_j) \geq t_2$ **or** $\mu_i, \mu_j \in \theta_i \cap_\alpha \theta_j$ **then** ▷ merge gates with substantial overlap
15: $\quad\quad\quad\quad \theta_m \leftarrow \text{merge}(\theta_i, \theta_j)$ ▷ θ_m denotes the single, merged gate
16: $\quad\quad\quad\quad \theta \leftarrow \theta \setminus \{\theta_i, \theta_j\} \cup \{\theta_m\}$ ▷ remove merged gates, and add new gate
17: $\quad\quad\quad$ **if** $0 < \text{overlap}(\alpha, \theta_i, \theta_j) < t_2$ **then** ▷ shrink gates with small overlap
18: $\quad\quad\quad\quad \theta_i^{\text{new}}, \theta_j^{\text{new}} \leftarrow \text{shrink}(\theta_i, \theta_j)$ ▷ shrink θ_i, θ_j until they do not overlap
19: $\quad\quad\quad\quad$ **if** $\min(\text{minorAxis}(\theta_i), \text{minorAxis}(\theta_j)) < b^{\min}$ **then** ▷ merge if at least one axis is too small
20: $\quad\quad\quad\quad\quad \theta_m \leftarrow \text{merge}(\theta_i, \theta_j)$
21: $\quad\quad\quad\quad\quad \theta \leftarrow \theta \setminus \{\theta_i, \theta_j\} \cup \{\theta_m\}$
22: $\quad\quad\quad\quad$ **else**
23: $\quad\quad\quad\quad\quad \theta_i \leftarrow \theta_i^{\text{new}}; \theta_j \leftarrow \theta_j^{\text{new}}$ ▷ Update θ with the shrunken $\theta_i^{\text{new}}, \theta_j^{\text{new}}$
24: **until** $L_{\text{new}} - L_{\text{old}} < \varepsilon$

Recap that by overlap of two ellipses we mean the overlap area divided by the size of the smaller ellipse. Also keep in mind that the confidence level α determines the general size of an ellipse when being inferred from a Gaussian distribution. Thus, the "overlap of two Gaussian distributions" calculated via $\text{overlap}(\alpha, \theta_i, \theta_j)$ also depends on α.

Delete large "background" ellipses (Alg. 1, lines 9–11): We first consider each Gaussian $\theta_i \in \theta$ and test for all pairs $\theta_j, \theta_{k\neq j} \in \theta \setminus \{\theta_i\}$ whether θ_i is overlapping both θ_j and θ_k by a fraction larger than a threshold t_1 given by the user. If this is the case θ_i is removed from θ.

Pairs of ellipses to merge (Alg. 1, lines 13–14): We repeatedly select the pair of ellipses with highest overlap and merge if this overlap exceeds threshold t_2. In addition, we merge if the overlap area contains both center points μ_i and μ_j. The merged gate θ_m replaces the two gates θ_i and θ_j. The merge function, described in detail in Section 3.6, treats this as a Bayesian update of two Gaussians, giving the most likely posterior density, taking into account the mixture weights π_i and π_j.

Pairs of ellipses to shrink (Alg. 1, lines 17–23): Overlapping gates with an overlap less than threshold t_2 must be shrunk instead of merged. The shrinking procedure shrink (Alg. 1, line 18, details in Section 3.7) performs a π_i, π_j-weighted rescaling of the ellipses θ_i, θ_j and returns $\theta_i^{\text{new}}, \theta_j^{\text{new}}$. According to goal (ii), gates should not be shrunk to sizes less than some threshold that needs to be set once per experiment. This requires us to check the minor axes b_i, b_j against the $b_{\min}$ threshold constraining the smaller axis and $\theta_i^{\text{new}}, \theta_j^{\text{new}}$ if either is too small. In that case, instead of shrinking we merge these two ellipses using their original values θ_i, θ_j.

Still, at most $O(\binom{K}{2})$ selection steps are being performed in line 13. In addition, we have $K^2 \ll N$, the number of gates is much smaller than the number of data points. Hence these additional steps take negligible time compared to $\text{EM}(\theta, \mathcal{X})$, and thus pursue goal (vi) "running time below typical bacterial generation times".

4. Results

4.1. Automated gating of an artificial mock community

Artificial microbial mock communities are ideal as a benchmark for testing tools designed for gating of cytometrically measured cell samples, since their composition, and thus the position and cell-numbers of the used strains in the plot are well-defined. As it was already done for `flowEMMi` [25], `flowEMMi v2` was tested on (the same) two artificial microbial cytometric mock communi-

ties consisting of either four or three different bacterial species [26]. As described in [26] the strains (*Stenotrophomonas rhizophila* DSM 14405, *Escherichia coli* DSM 4230, *Kocuria rhizophila* DSM 348, and *Paenibacillus polymyxa* DSM 36) of the liquid mock community were cultivated and harvested from liquid culture while the strains of the plate mock community (liquid mock community without *Escherichia coli*) were cultivated and harvested from cultivation plates. The cells were handled, fixated and DNA stained as described in [26]. The used proportions were determined by optical density ($d_{\lambda\ 700nm}$ = 0.5 cm). The composition of the liquid mock community was *S. rhizophila*: 2.5%; *K. rhizophila*: 20%, *P. polymyxa*: 70%, and *E. coli*: 7.5%, while the plate mock community consisted of *S. rhizophila*: 1%, *K. rhizophila*: 19%, and *P. polymyxa*: 80%. After the two mock communities were measured the resulting flow cytometric patterns underwent the automatic gating procedure of `flowEMMi v2`. The data shown in Fig. 3 clearly highlight the powerful performance of `flowEMMi v2` which can separate the four, respective three strains of the two mock communities and their subpopulations. The ellipses are non-overlapping and tightly enclosing the regions of high cell density.

4.2. Automated gating of a microbial cytometric time-series

`flowEMMi v2` was also tested on time-series data of a natural microbial community [8]. The study comprised five insular identically operated bioreactors which were inoculated with the same microbial community that was taken from a wastewater treatment plant in Eilenburg, Germany. Two of the reactors (C1 and C2) served as a control and were run at a constant temperature. In the other three bioreactors (D1, D2 and D3) the temperature varied between 30°C and 40°C. The five bioreactors were operated for 91 days and samples were taken every 2 to 4 days. This resulted in 65 samples per reactor. One additional sample was taken at the inoculation. Therefore, 326 samples were available in total. The bioreactors had a final working volumse of 800 mL medium and the environmental conditions were set to 350 rpm and to an aeration rate of 150 rpm compressed sterile filtered ambient air. The reactors had a dilution rate of 0.72 per day, and thus a working volume exchange of $(1/0.72) \cdot 24 \approx 33.3$ h. Microorganisms with a generation time longer than 23.1 h were washed out of the system. The whole procedure is described in [8]. In the study a gate template was set manually (see Fig. 4C), which was used to determine the number of cells per gate and per sample. The cell numbers per gate enabled comparing the composition of the microbial communities over time and between the different reactors in order to find out whether community variability can be controlled by implementing soft temperature stressors as potential synchronizers [8].

Fig. 4A shows the automated gate setting of sample 52 from reactor C1 as an example. The same workflow was done for all 326 samples using an ellipse-size of $\alpha = 0.5$. Next, all 326 sets of ellipses were combined into one set, computing the gate template. The result is shown in Fig. 4B. In Fig. 4C the gate template is shown which was set by hand in [8] based on the same sample set. The same comparison was made for three more samples in Fig. S5. Fur-

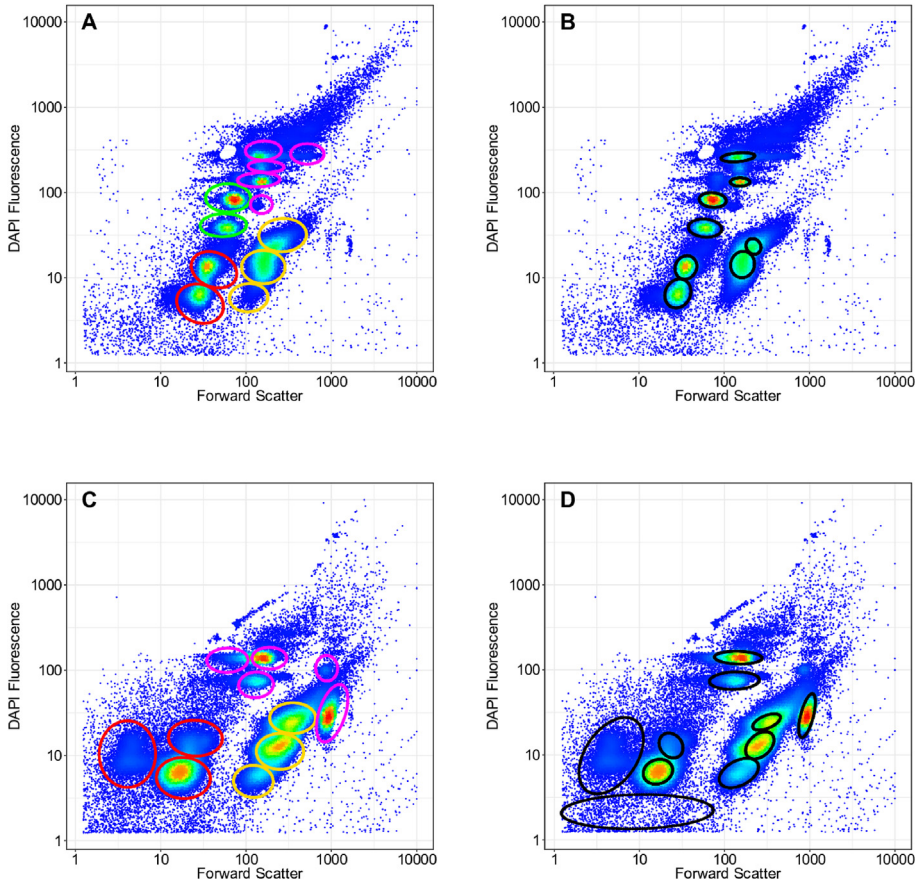

Fig. 3. Comparison of **A**, **C**: manual gating with **B**, **D**: automated gating of the two microbial cytometric mock communities using `flowEMMi v2`. **A**, **B**: For the liquid mock community the strains were mixed at proportions *S. rhizophila*: 2.5% (red); *K. rhizophila*: 20% (orange), *P. polymyxa*: 70% (pink) and *E. coli*: 7.5% (green). **C**, **D**: For the plate mock community the strains were mixed at proportions *S. rhizophila*: 1% (red), *K. rhizophila*: 19% (orange), and P. polymyxa: 80% (pink). The gate template was created inside of a cell gate where instrumental noise was excluded. The beads were removed from the data set beforehand.

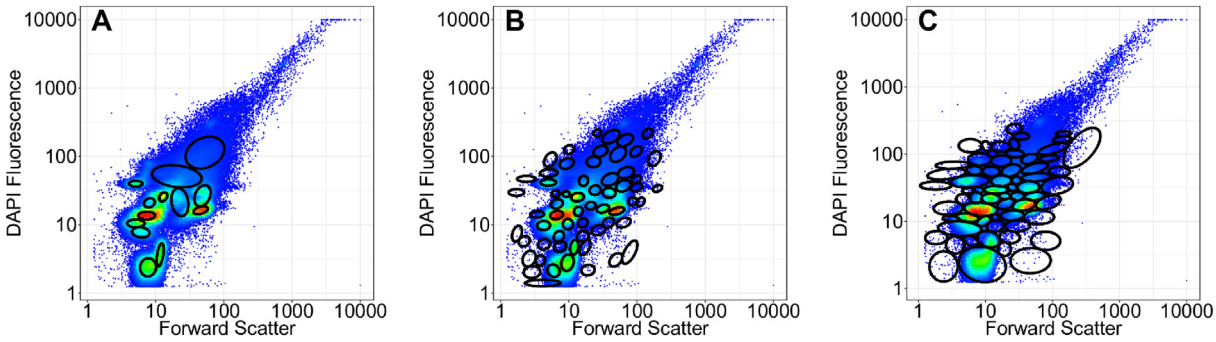

Fig. 4. Comparison of automated gating using `flowEMMi v2` with manual gating of samples taken from [8]. The cytometric dot plot shows the cell distributions of sample 52 (day 72) from reactor C1 [8]. **A:** The outcome of the automated gating is shown for sample 52 using `flowEMMi v2`. All cell clusters at higher cell abundances were captured with an ellipse. **B:**The automated gate template is shown for all 326 samples using `flowEMMi v2`. **C:**The gate template which was set by hand in [8].

thermore, the movies in the supplementary material, C1_singleTimePoints.mov and gateTemplateC1.mov show all samples of C1 gated individually and inside the gate template, respectively.

In order to compare the gate template that was generated by `flowEMMi v2` to the one that was set by hand, we need to check whether they reveal the same community dynamics. Therefore, the cell numbers per ellipse and sample were calculated for the automatically created gate template (using `flowEMMi v2`) and the one that was set by hand (using FlowJo [27]). Both approaches deliver cell numbers per gate per sample over time. The resulting data were evaluated in a multi-dimensional space according to the number of gates that were set. Via non-metric multidimensional scaling (NMDS) the distance of cell numbers per gate between all samples was computed which arranged samples according to similarity. For the creation of the NMDS plots we used the metaMDS function of the vegan package [44] in R. Fig. 5 shows NMDS plots of the time-series data by using the gate template created by hand (A,D) and by `flowEMMi v2` (B,E) - on the basis of 326 samples, analyzed by forward scatter and DAPI fluorescence. Each circle belongs to one sample and the size of each circle corresponds to the time of sampling. Consecutive samples are indicated by a line that connects them. Fig. 5C show the evolution of microbial communities in the undisturbed control reactors, while Fig. 5F show the evolution of communities in the temperature disturbed reactors. The outcome clearly demonstrates that `flowEMMi v2` is able to capture the same community dynamics as the manually set gate template. This indicates that `flowEMMi v2` sets gates in a meaningful way.

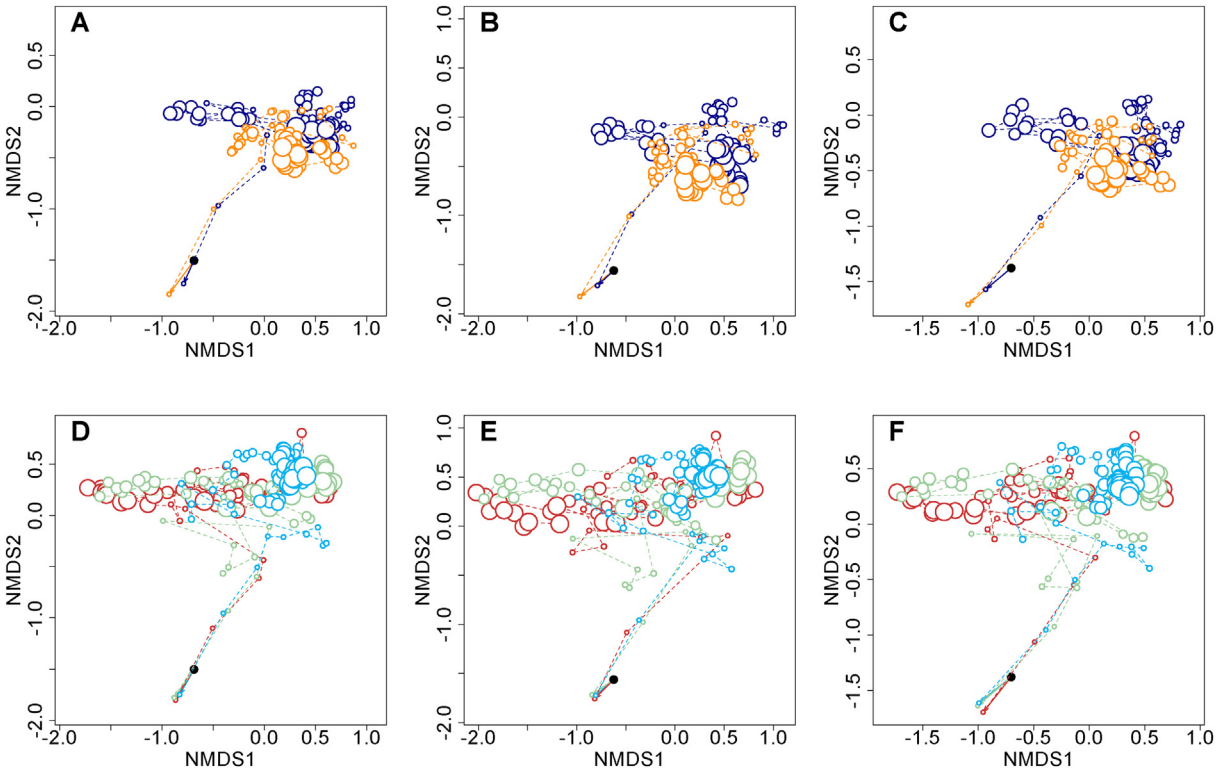

Fig. 5. Non-metric multidimensional scaling (NMDS) plots of the time-series data from [8]. Cells were analyzed according to their forward scatter and their DAPI fluorescence. **A**, **B** and **C** show the data of the undisturbed control reactors C1 and C2, and **D**, **E**, and **F** show the data of the temperature disturbed reactors D1, D2 and D3. A gate template was created by hand in [8] (**A**, **D**), or automatically using `flowEMMi v2` (**B**, **E**) or PhenoGMM (**C**, **F**). Each circle corresponds to a particular sample. The samples of C1 and C2 are shown in orange and dark blue, while the samples of D1, D2 and D3 are depicted in light blue, red and green, respectively. The size of each circle corresponds to the time of sampling. Consecutive samples are indicated by a line that connects them.

To compare `flowEMMi v2` with the recent tool PhenoGMM [22], which also operates on Gaussian mixture models, we tested the tool on the same time-series. PhenoGMM provides the option to gate multiple samples at once by combining all the data of the different samples into one data set. The package also provides a sub-sampling option. Since the experiment comprised 326 samples with 200,000 cells each, PhenoGMM crashed when it was applied to the whole time-series, and we thus applied the sub-sampling. For each of the 326 samples 2,000 cells were taken resulting in a total of 652,000 cells which were gated at once. In contrast to PhenoGMM, gating was performed with all cells when using gating by hand and `flowEMMi v2` resulting in gate templates containing 65 and 61 gates, respectively. Therefore, the maximum number of gates to be tested by PhenoGMM was set to 80 but only 48 gates were found. Thus, the gate setting by hand and by `flowEMMi v2` provided higher resolutions. Unlike PhenoGMM, where all cells in a sample were assigned to a gate by the underlying Gaussian mixture model, the gate setting by hand and `flowEMMi v2` excluded background cells (i.e., off-gate cells), increasing the exclusivity of cell types per gate. This is an important feature in the event that a gate is sorted for downstream OMICs analysis, which requires pure cell subpopulations. The gate templates of `flowEMMi v2` and PhenoGMM are compared in Fig. S6. PhenoGMM creates overlapping ellipses and assigns cells from the vicinity into the respective gates and colors gate and assigned cells with a representative color. A successful color representation of the 48 gates found by PhenoGMM was realized by the additional use of the R-package RColorBrewer [52]. Nevertheless, both automatic gating tools `flowEMMi v2` and PhenoGMM were able to clearly define gates for the whole data set of 326 samples.

The results of the gate template created by PhenoGMM are shown in Fig. 5C and F. The data are visually comparable to the outcomes of `flowEMMi v2` and the hand-set gate template, thus, also PhenoGMM was able to reveal the community dynamics.

To further test the comparability of the three approaches, the distances of the samples were calculated. The distances in the NMDS plot are not scaled metrically, which means that the similarity of the plots cannot be measured by the coordinates of the samples in the NMDS plots. Instead, the similarity of the outcome of the different methods needs to be measured directly by the cell numbers. Thus, for each method a 326×326 distance matrix was computed and normalized on the relative cell numbers giving the pairwise distances of the 326 samples. Now, the two matrices of `flowEMMi v2` and PhenoGMM were compared with the distance matrix of the manual gate template. This was done by calculating the absolute differences between the distance matrices. Since these absolute differences are not normally distributed, a Wilcoxon Rank Sum test [30] was applied showing that `flowEMMi v2` yields cell numbers which are significantly more close to the cell numbers generated by the manual template, with a p-value below $2.2 \cdot 10^{-16}$. Therefore, `flowEMMi v2` performs better on this time-series than PhenoGMM.

4.3. Ellipse sizes

The user of `flowEMMi v2` can choose the general size of the ellipses in the model via a parameter $\alpha \in (0, 1)$. With increasing α the ellipses get larger, see Fig. 6. We recommend using a small α when evaluating a time-series of flow cytometry data, as it was done in the experiment of Fig.4B, where α was 0.5. Choosing a low α results in small ellipses that cover the cells in the center of a subcommunity. From experience we advise to use an α of at least 0.5. Otherwise the ellipses get unsuitably small, see e.g. the purple

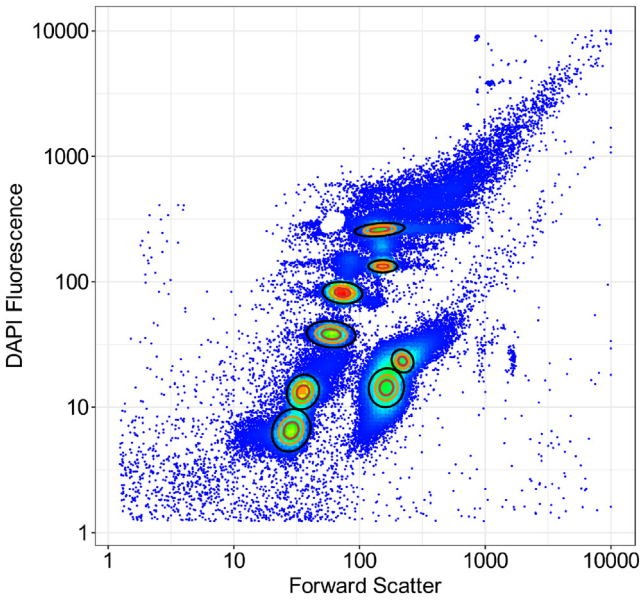

Fig. 6. Starting with the Gaussian mixture model of Fig. 3B, where the microbial mock community [26] grown in liquid medium was gated with `flowEMMi v2` at $\alpha = 0.99$ (black ellipses), α was decreased to 0.85 (orange ellipses) and 0.5 (red ellipses).

ellipses in supplemental Fig. S7. However, for specific super-clean sorting applications, such as single cell sorting for further downstream omics treatments, smaller α values could also be beneficial. We recommend using a large α when running `flowEMMi v2` on single samples like it was done in Fig. 3B and Fig. 3D with $\alpha = 0.99$. The described parameter selection pursues goal (iii) "each gate encloses a region of high cell count; the tightness of the enclosure can be chosen by the user". Mathematical details on how α influences the size of an ellipse are provided in Section 3.3.

4.4. Hierarchical gating

On closer inspection some automatically generated ellipses may cover more than one subcommunity. The colors of the density plots of individual ellipses can be re-scaled as shown in Fig. 7. This re-scaling is performed individually for each ellipse based on the respective subsets of virtual cells by using the R-package KernSmooth [42].

Afterwards the ellipses were compiled into a new plot (Fig. 7, right) depleted of off-gate events. The zoomed image of each single ellipse enables the visualization of the substructures per gate, especially if the gate comprised only low cell numbers (i.e. within the blue region) such as the ellipse at the upper right corner of Fig. 7. The ellipses may contain one to several maxima, the latter being found in the highlighted ellipse in the center of the graph, which includes two maxima. Thus, the distribution of virtual cells inside the highlighted ellipse was further resolved using `flowEMMi v2`, leading to a hierarchy of gates. The decision of which gates to resolve further is not included in the workflow of `flowEMMi v2`, and thus has to be decided by the user. If desired, the respective ellipse could be deleted from the corresponding mixture model and be replaced by the two smaller ellipses for further analysis. Because this approach allows more intricate gate settings, it can be useful for cell sorting, especially when narrowly distributed subcommunities or subpopulations are of interest. In

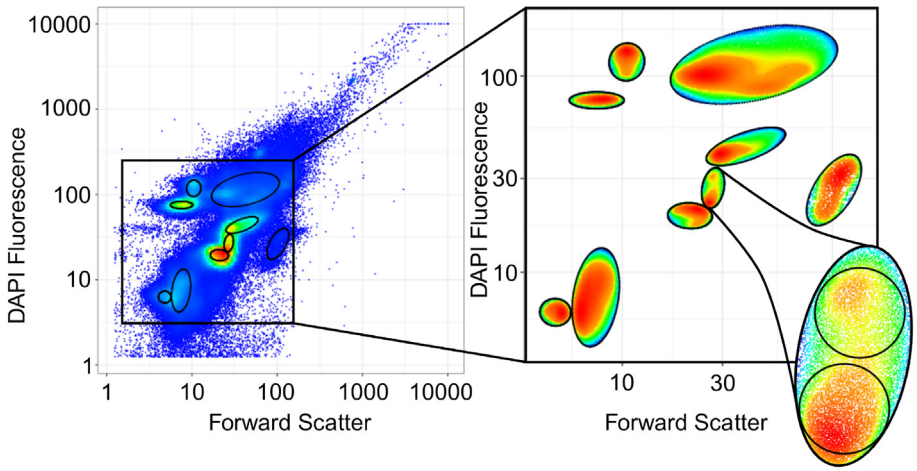

Fig. 7. Hierarchical gating. An automated gating using `flowEMMi v2` of sample 5 (day 3) from reactor C1 is shown on the left. Then, all the data-points captured by some ellipses were again plotted on the right. The colors were re-computed for each ellipse. The highlighted ellipse covers two maxima. The virtual cells within the highlighted ellipse were further resolved automatically by `flowEMMi v2`.

summary, goal (v) "hierarchical gating should be possible to increase the resolution of cell subsets within a gate", is fulfilled.

5. Discussion

When comparing gating algorithms two main characteristics are in focus: the quality of the gating process itself and the running time performance to ensure on-line or at-line applicability. `flowEMMi v2` performs well compared to other recent methods such as SamSPECTRAL [19], flowMeans [18], flowMerge [20], and PhenoGMM [22]. A more detailed discussion of the predecessor of flowEMMi v2 and the three algorithms mentioned above (with the exception of PhenoGMM, which was developed later) can be found in [25]. All tools under evaluation are available as R packages with their own plot functions. The graphical results of the gating process of `flowEMMi v2` and the four other methods are shown in Fig. S8. The tools flowMeans and SamSPECTRAL are non-parametric and feature low running times. flowMeans is based on the k-means algorithm while SamSPECTRAL uses spectral clustering, and thus is an application of graph theory. In contrast to that, both flowMerge and PhenoGMM create a model of the underlying data, using elliptical distributions, see Fig. S8 E and F. The program flowMerge is an enhancement of flowClust [21] and uses a mixture of Student's *t*-distributions with box-cox transformed data. The recently developed tool PhenoGMM [22] uses Gaussian mixture models. This tool gave the best outcomes from the four tested ones in comparison to flowEMMi v2. The number of gates can be chosen directly by the user but also an automated selection of best numbers of gates is possible. PhenoGMM is the only tool besides `flowEMMi v2` that is able to create a gate template for multiple samples. Therefore, we especially tested the outcomes of `flowEMMi v2` and PhenoGMM with hand-set gate templates.

In `flowEMMi v2` the ellipses are non-overlapping and the cell numbers per ellipse (=gate) are counted as the number of cells within each ellipse. In PhenoGMM the ellipses are overlapping (see Fig. S8 F), and the cells numbers were counted both from the ellipse and the cells in the vicinity, together called gate. In PhenoGMM, the gates are not the ellipses of the underlying model, they are constructed based on a probability function, which assigns a cell to the gate with the highest likelihood. That is, each cell is assigned to exactly one ellipse. The irregular polygons that form resulting gates are visualized by individually assigned colors, so that graphically each color represents a gate. We define gates that were constructed this way as 'color-defined' gates in the following. In flowMerge the gates are also color-defined (see Fig. S8 E). Although the geometric objects of the underlying model overlap in PhenoGMM and flowMerge, the color-defined gates are not overlapping. For the application of cell-sorting the polygons of the color-defined gates could be used. Graphical marking of these polygons, e.g. for cell sorting, would be an additional step to be done manually, as this is not offered by PhenoGMM and flowMerge. This would be an issue in an automated process.

In `flowEMMi v2` we introduce a parameter α that controls how tight gate sizes are chosen (see Section 4.3): a value between 0.5 and 0.7 should be chosen to only gate the core of a subcommunity while high values of α, e.g. 0.99, result in gates which are larger and thus more similar to the manual gating for single samples. In Fig. S9 the gating of the liquid mock community by `flowEMMi v2` is shown for different values of α. The principle of color-defined gates can, if desired by the user, be applied in `flowEMMi v2` by gating with tight gates using a low α, and then expanding the gates post hoc by setting a higher α and assigning cells to gates by likelihood. For example, we took the non-overlapping elliptical gating of the liquid mock community at $\alpha = 0.7$ shown in Fig. S9 A, but then raised α to 0.99 in the plot of Fig. S10 A. These overlapping ellipses were then transformed into color-defined gates, see Fig. S10 B. However, note that the partitioning of cells by the color-defined gates sometimes is biologically not sensible. For example, in Fig. S10 B there are cells in the yellow gate that should belong to the light blue gate from a biological point of view. Therefore, this approach results in gates which not always group cells with similar phenotypic properties together (see e.g., also Fig. S8 F, where the green gate of PhenoGMM covers a major part of the plot). Furthermore, we highly recommend elliptical gates since they match the natural shape of subcommunities. For these reasons we focused on non-overlapping elliptical gates in this work.

Since `flowEMMi v2`, PhenoGMM and flowMerge (in contrast to SamSPECTRAL and flowMeans) generate parametric models on the data, we compared them with each other. The tools have a random initialization step which might lead to different outputs for the same test sample. Therefore, the random number generator (RNG) state within R was set to a fixed value via the function set. seed. To evaluate the running time performance, we used the microbial cytometric time-series data [8], which was already investigated in Section 4.2. All samples of reactor C1 (66 samples) served as test data for `flowEMMi v2` and PhenoGMM and only the

first 20 samples were used for flowMerge. Note that the samples were gated one by one (resulting in a different set of gates per sample) since flowMerge does not provide an option for gate templates. All tools were tested for a number of gates between 1 and 15. The results are shown in Fig. S11. We found both `flowEMMi v2` and PhenoGMM approximately 50× faster than flowMerge, while in a direct comparison PhenoGMM was approximately 2× faster than `flowEMMi v2`. `flowEMMi v2` and PhenoGMM consistently achieved running times of less than 10 min making them suitable for at-line gating of active time-series. The median running time of `flowEMMi v2` was about twice the median running time of `flowEMMi`. Thus, the increase of running time due to the removal of overlaps is acceptable. The median running time of `flowEMMi v2` for the given data sets with approximately 200,000 cells is ≈ 6 minutes, and thus fulfills goal (vi) "the performance of the algorithm is retained with respect to running time below typical bacterial generation times".

Next, as a quality check, we tested the outcomes of `flowEMMi v2`, PhenoGMM and flowMerge. One major problem in the evaluation of automated gating tools is that an actual gold-standard for microbial communities is missing. In lieu of such a gold standard we use a carefully curated mock community [26] as a benchmark. This mock community was already used in Section 4.1. This artificial microbial community with known physiological characteristics of the strains was developed as a standard procedure to calibrate the instruments and experimental setup [26]. We therefore used expert gating as reasonable control for the automated gating tools. Each tool was run on the same subset of data points excluding technical noise and beads. The gating of the mock community grown in liquid medium is given in Fig. S8.

The F1-score was applied to measure the similarity between the outcome of an automated gating tool and a manually curated gating. For this comparison, we first need to define which gate in the automated gating corresponds to which manually curated gate. For example, in Fig. S8 the two red gates in the manually curated gating (A) were assigned to the two ellipses on the bottom left in the automated gating of `flowEMMi v2` (B). Mathematically this was done by a greedy approach (by iterative pairing of the most frequent matching assignment between curated gates and automatically constructed gates). The calculation of the F1-score requires the number of true positives (TP), the number of false positives (FP) and the number of false negatives (FN) for each gate. We define as a true positive event a cell that has been assigned to equivalent gates both in the manually curated gating and by the automated gating tool. Accordingly, a false negative event is a cell in the manually curated gating that does not lie in the equivalent automated gate. This happens if the automated gate is too small. Finally, a false positive event is a cell in the automated gating that is not inside the equivalent manually curated gate. This happens if the automated gate is too large.

Furthermore, for each manually curated gate the positives (P) is the number of cells inside that gate. Accordingly, the negatives (N) of a manually curated gate is the number of cells which are outside of that gate. Now, for each gate the true positive rate (TPR = TP/P) is the number of true positives divided by the positives. The false negative rate (FNR = FN/P) is the number of false negatives divided by the positives. The false positive rate (FPR = FP/N) is the number of false positives divided by the negatives. Therefore, the TPR is desired to be high (close to 1), while the FNR and FPR are desired to be low (close to 0).

The overall TPR, overall FNR and overall FPR of the model were calculated according to the relative proportions of the cells in the manually curated gates. That is, manually curated gates with a high number of cells influence the overall TPR, FNR and FPR more than gates with low cell numbers. Finally, the overall F1-score was calculated: $(2 \cdot \text{overallTP})/(2 \cdot \text{overallTP} + \text{overallFN} + \text{overallFP})$. The F1-score ranges from 0 to 1 with 1 meaning that the automated and manual gating were identical.

Supplementary Table S1 shows the individual overall F1-scores, TPR, FPR and FNR for the different tools on the carefully curated liquid and plate mock communities. Clearly, if we use the color-defined gates in `flowEMMi v2` at an $\alpha = 0.99$, then `flowEMMi v2` outperforms all other tools (Table S1). flowMerge is outperformed by all other tools. Table 1 shows the values of `flowEMMi v2` with the use of elliptical gates. PhenoGMM, flowMerge and `flowEMMi` are counting cells in polygonal gates that are color-defined. As expected, when lowering α step-wisely from 0.99 to 0.7, the F1-score of `flowEMMi v2` decreases (Table 1). The TP, FN, and FP values of `flowEMMi v2` at $\alpha = 0.99$ obtained for each elliptical gate (Table S2 and Table S3 for the liquid and plate mock communities, respectively) are mostly lower compared to those of PhenoGMM. Gate 1 (gray) comprised the cells that lie in the background and highlights the fact that `flowEMMi v2` excludes background cells better. Some of the other gates could not be associated with the manual gates of the mock communities by both PhenoGMM and `flowEMMi v2` and were given a value of 0. On the other hand, the model also found gates that were not set in the manual gating. The unassigned gates indicated over-fitting and were found by PhenoGMM (2 color-defined gates) and `flowEMMi v2` (1 elliptical gate). However, for the reasons stated above, we focus primarily on elliptical (which exclude background events) and non-overlapping gates which results from the workflow (Fig. 1). Due to the more restricted gating these gates are expected to be smaller and thus more different from the manual gating, as shown for the different values of α in Fig. S9 and Table 1. With these smaller ellipses `flowEMMi v2` has slightly lower F1-scores compared to

Table 1
Test of PhenoGMM, flowMerge, `flowEMMi` and `flowEMMi v2` on the data of the artificial microbial communities. `flowEMMi v2` was run with non-overlapping elliptical gate setting.

	liquid				plate			
	F1-score	TPR	FNR	FPR	F1-score	TPR	FNR	FPR
PhenoGMM	0.849	0.845	0.155	0.013	0.807	0.831	0.169	0.018
flowMerge	0.552	0.609	0.391	0.063	0.622	0.680	0.320	0.061
`flowEMMi`	0.838	0.825	0.175	0.015	0.823	0.807	0.193	0.010
`flowEMMi v2`, $\alpha = 0.7$	0.709	0.665	0.335	0.044	0.616	0.595	0.405	0.076
`flowEMMi v2`, $\alpha = 0.8$	0.738	0.703	0.297	0.039	0.755	0.737	0.263	0.039
`flowEMMi v2`, $\alpha = 0.9$	0.783	0.764	0.236	0.031	0.782	0.762	0.238	0.031
`flowEMMi v2`, $\alpha = 0.95$	0.805	0.792	0.208	0.027	0.780	0.766	0.234	0.030
`flowEMMi v2`, $\alpha = 0.99$	0.822	0.816	0.184	0.024	0.767	0.749	0.251	0.035

For quality control PhenoGMM, flowMerge, `flowEMMi v2` and `flowEMMi` (`flowEMMi v2` without the removal of overlaps) were run on the microbial mock communities of [26], which were already investigated in Section 4.1. The non-overlapping elliptical gates of `flowEMMi v2` were created at different values of α. Each model was compared to the manual gating. The individual corrsponding F1-score, true positive rate (TPR), false negative rate (FNR) and false positive rate (FPR) are shown.

PhenoGMM and the older version of `flowEMMi`, where no restrictions were set with regard to non-overlapping and elliptical gate positions and shapes (Table 1). Nevertheless, the F1-score of `flowEMMi v2` remained high. The TPR values were lowest for flowMerge and highest for the gating tools which used overlapping gates and added background to the gates. TPR values decreased with α value in `flowEMMi v2`. The FPR was low for all approaches while the FNR was highest for flowMerge and `flowEMMi v2` at the value of $\alpha = 0.7$. While flowMerge does not set classical elliptical gates the high value for `flowEMMi v2` was the result of the much more distinct gating around the highest cell density of Gaussian cell distributions. These results suggest a trade-off between accurate gate definition and loss of cells for downstream biostatistical analysis or cell sorting for downstream OMICS analysis using `flowEMMi v2` and lower α values compared with inaccurate and overlapping gates that accommodate neighboring and background cells that do not belong to the target subpopulation or subcommunity but increase cell number for downstream biostatistical and OMICs applications.

The analysis of time-series flow cytometry data is of paramount importance in microbial community ecology. `flowEMMi v2` and PhenoGMM are the only tools known to date that can create gate templates necessary for statistical downstream analyses. Using a microbial community set of 326 samples, `flowEMMi v2` ($\alpha = 0.5$) set 61 gates for the gate template while PhenoGMM only set 48 gates. This points to a higher resolution of a microbial community by `flowEMMi v2`, which possibly could even be increased more by hierarchical gating. To test the reliability of the automatically set gate template by `flowEMMi v2` or PhenoGMM ecological properties of five microbial communities were tested for the 326 samples [8]. Using each method (manual gating, `flowEMMi v2` and PhenoGMM) the number of dominant gates (explanation in [8]) per sample was calculated (Fig. S12 A, B and Fig. S13 A, B, C) to determine alpha-diversity, which is showing changes in the composition of the communities. The values of the manual approach are depicted blue, while the numbers of `flowEMMi v2` and PhenoGMM are given in red and orange, respectively. In addition, the intra-community beta-diversity was calculated for each reactor. That is, the number of subcommunities, which change their status from being dominant in one sample to non-dominant in the successive sample and the other way round, is counted. High intra-community beta-diversity values indicate rapid, even stochastic changes in the community composition (Fig. S12 C, D and Fig. S13 D, E, F). The values of the manual approach are again depicted blue, while the numbers of `flowEMMi v2` and PhenoGMM are given in red and orange, respectively. At first glance, the values and trends of the two properties, alpha-diversity and intra-community beta-diversity, were similar across all three approaches. To find differences in the diversity values of `flowEMMi v2` and PhenoGMM compared to those obtained from the manual gate template, Spearman's correlation and Kendall's correlation were used. Both correlation methods revealed significant correlations (p-value $< 2.2 \cdot 10^{-16}$) of the alpha-diversity values obtained from both `flowEMMi v2` and PhenoGMM with those obtained from the manual gate template. The same was observed for the intra-community beta-diversities. This shows that both `flowEMMi v2` and PhenoGMM are suitable for evaluating flow cytometric time-series data. In every case the correlation coefficient of `flowEMMi v2` was higher than the one of PhenoGMM indicating a better performance of `flowEMMi v2`, (Table S4).

Additionally, the subsistence of subcommunities was measured by the nestedness. In contrast to that, turnover measures the replacement of dominant subcommunities. For each gating method turnover and nestedness were calculated per reactor using the Sørensen dissimilarity index of the R-package betapart [10]. The results are given in Fig. S14. A binomial test showed that the turnover values obtained by `flowEMMi v2` were significantly closer to the values obtained from the manual gate template (p-value $\approx$ 3%). For the nestedness values there was no significant difference.

PhenoGMM and `flowEMMi v2` both have special features. One advantage of PhenoGMM is its ability to evaluate samples multi-dimensionally, which means that more parameters of a cell can be used for each cell characterization and to create a gate template. This feature is not yet available for `flowEMMi v2`. Both tools are able to handle time-series of flow cytometric data. However, PhenoGMM cannot incorporate all cells into the calculation of a gate template if the time-series is long. In this case PhenoGMM needs to draw a sub-sample of cells from each fcs-file. This might be the reason why `flowEMMi v2` performed better on the time-series data above. Moreover, PhenoGMM does not apply hierarchical gating, and the sizes of the gates cannot be adjusted. Manual gating procedures and also `flowEMMi v2` identify cells in the background. There is no such identification in PhenoGMM (and also not in flowMerge and SamSPECTRAL). Therefore, the gates are not well-separated and the cells between the gates run the risk of being assigned to the wrong gates. Furthermore, to our knowledge `flowEMMi v2` is the only tool that is able to create non-overlapping elliptical gates on microbial time-series data.

5.1. Extensions to higher dimensions

`flowEMMi v2` is designed as an algorithm to handle flow cytometric data of microbial communities. In this context fingerprinting on two cell properties has proven reliable. Preferably forward scatter, which correlates to cell size, and all-cell DNA staining, which reflects the DNA content of the cells, are used. However, it can be useful to measure additional cell properties, e.g. side scatter or further fluorescence parameters.

This naturally raises the question of how `flowEMMi v2` can be extended. The ellipse equations in Section 3.3 are unwieldy and care has to be taken in their implementation and the corresponding merging and shrinking operations already for two dimensions.

To solve the problem in higher dimensions, at least three algorithmic approaches are possible. These consist of using the explicit equations of ellipsoids, finding the solution numerically, and using two-dimensional heuristics.

The first suggested algorithm follows the basic idea laid out in this paper. The ellipsoids are transformed in such a manner that one of the ellipsoids is converted into the unit sphere. Formulas for area and volume of hyperspherical caps have been published [53]. An adaptation to the problem of finding the hyper-ellipse that forms the boundary of the intersection of the two ellipsoids will require further research.

Instead of aiming for an analytic solution, it is also possible to solve the problem numerically for all dimensions. In the case of, say, overlapping ellipsoids that are to be shrunk until the volume of intersection is zero, one can perform what amounts to a binary search within the shrinking procedure. The scale of both ellipsoids is adapted until the largest scaling factor (< 1.0) has been found with no intersection between the ellipsoids. Using a binary search algorithm allows finding the solution in (at most) linear time.

A third solution presents itself in the form of a heuristic, utilizing the algorithm developed in this paper. Instead of trying to solve the higher-dimensional ellipsoid equation in k dimensions, we instead iteratively perform the following operations. First, we run the EM steps as currently implemented. Then instead of trying to solve the problem of finding non-overlapping ellipsoids in higher dimensions, we solve the non-overlapping constraints for all $\binom{k}{2}$ combinations of two dimensions. Given that k is assumed to still

be low-dimensional (i.e. maybe as low as $k = 3$) this procedure will be computationally efficient as solving the overlap constraints is fast compared to running the EM steps. The number of calculations for the overlap constraints is bounded by $O(k^2)$.

The resulting ellipsoids will be intersection-free in all 2-dimensional subspaces, and also in the original k-dimensional space. Which of these solutions is to be preferred is a topic of further research.

6. Conclusion

There is a big demand for automated clustering procedures for microbial communities, not only in exploratory studies but also in monitoring and control of biotechnological processes for the evaluation of microbial cytometric samples derived from biotechnology, natural environment as well as agricultural and human health disciplines.

This demand has led to a number of publications in the past few years with recent advances in both quality of gating procedures and the running times of the algorithms. Our previous work focused on establishing an algorithm whose capabilities allow for at-line gating. However, our previous work yielded color-defined gates, for which the underlying ellipses overlap. As pointed out in the discussion the partitioning of cells by the color-defined gates sometimes is biologically not sensible because they not always group cells with similar phenotypic properties together. Moreover, elliptical gates correspond to the typical Gaussian distribution of population and subcommunity characteristics. For these reasons we focused on non-overlapping elliptical gates in this work.

For single samples, the main improvement is the creation of non-overlapping elliptical gates fulfilling goal (i) "gates are always non-overlapping". Non-overlapping gate templates immediately enable flow cytometric cell sorting of interesting cell types within microbial communities, which in turn allows for plenty of downstream analyses to be performed. Furthermore, time-series analysis is simplified, since `flowEMMi v2` is able to create gate templates which are used to capture the changing nature of the biological processes which can be observed by the changing number of cells in individual gate templates. Therefore, goal (ii) "the tool is not only able to gate single samples but it can also handle time-series of flow cytometry data", is achieved. It was shown that the sizes of the gates created by `flowEMMi v2` can be influenced by the user and that the gates in general enclose a region of high cell count. Thus, goal (iii) is reached. Moreover, `flowEMMi v2` is an automatic procedure where the randomness involved can be avoided by controlling the random number generator (RNG) in R (see function set.seed). This is crucial since the standard approach of manual gating is dependent on the user. Hence, goal (iv) "the gate template is objective and reproducible" is fulfilled.

In addition, we discussed a visual appearance of even higher resolutions of subcommunities in Section 4.4 and Fig. 7. Our solution establishes a hierarchy of gates, where larger elliptical gates can be further subdivided without breaking non-overlap constraints (except of course for the parent and child gates in the hierarchy). Thus, goal (v) "hierarchical gating should be possible to increase the resolution of cell subsets within a gate" is achieved. Taken together, this work provides a major improvement since we can now significantly increase the quality of the gating procedure, without sacrificing running time performance, and thus fulfills goal (vi) "the performance of the algorithm is retained with respect to running time below typical bacterial generation times".

Therefore, we highly recommend the use of `flowEMMi v2` as a valuable and reliable tool for automated gating of spatiotemporal series of cytometrically measured microbial communities, providing reliable data for ecological interpretation of community behavior and dynamics. The tool bridges the crucial bottleneck between automated sampling, sample-processing, flow cytometry and automated downstream analysis.

Funding

This work was funded by the Federal Ministry for Economic Affairs and Energy (16KN074623, INET-BitCa), by the German DFG Collaborative Research Centre AquaDiva (CRC 1076 AquaDiva), by the German state of Thuringia via the Thüringer Aufbaubank (2021 FGI 0009), and by the Open Access Publishing Fund of Leipzig University supported by the German Research Foundation within the program Open Access Publication Funding.

CRediT authorship contribution statement

Carmen Bruckmann: Formal analysis, Investigation, Software, Visualization. **Susann Müller:** Supervision, Writing - review & editing. **Christian Höner zu Siederdissen:** Investigation, Supervision, Writing - review & editing.

Declaration of Competing Interest

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

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.csbj.2022.11.033.

References

[1] Ley R, Turnbaugh P, Klein S, Gordon J. Microbial ecology: human gut microbes associated with obesity. Nature 2007;444:1022–3. https://doi.org/10.1038/4441022a.
[2] Lee JW, Kim HU, Choi S, Yi J, Lee SY. Microbial production of building block chemicals and polymers. Curr Opin Biotechnol 2011;22(6):758–67. https://doi.org/10.1016/j.copbio.2011.02.011.
[3] Chubukov V, Mukhopadhyay A, Petzold C, Keasling J, García Martín H. Synthetic and systems biology for microbial production of commodity chemicals. NPJ Syst Biol Appl 2016;2(1):16009. https://doi.org/10.1038/npjsba.2016.9.
[4] Koch C, Müller S, Harms H, Harnisch F. Microbiomes in bioenergy production: from analysis to management. Curr Opin Biotechnol 2014;27:65–72. https://doi.org/10.1016/j.copbio.2013.11.006.
[5] Koch C, Kuchenbuch A, Kretzschmar J, Wedwitschka H, Liebetrau J, Müller S, Harnisch F. Coupling electric energy and biogas production in anaerobic digesters – impacts on the microbiome. RSC Adv 2015;5:31329–40. https://doi.org/10.1039/C5RA03496F.
[6] Vučić V, Süring C, Harms H, Müller S, Günther S. A framework for P-cycle assessment in wastewater treatment plants. Sci Total Environ 2021;760:. https://doi.org/10.1016/j.scitotenv.2020.143392143392.
[7] Edwards BS, Oprea T, Prossnitz ER, Sklar LA. Flow cytometry for high-throughput, high-content screening. Curr Opin Chem Biol 2004;8(4):392–8. https://doi.org/10.1016/j.cbpa.2004.06.007.
[8] Liu Z, Cichocki N, Hübschmann T, Süring C, Ofiţeru I, Sloan WT, et al. Neutral mechanisms and niche differentiation in steady-state insular microbial communities revealed by single cell analysis. Environ Microbiol 2019;21(1):164–81. https://doi.org/10.1111/1462-2920.14437.
[9] Koch C., Schumann J., Günther S., Fetzer I., Müller S., flowCyBar: analyze flow cytometric data using gate information, R package version 1.18.1 (2019).
[10] Baselga A, Orme C, David L. betapart: an R package for the study of beta diversity. Methods Ecol Evol 2012;3(5):808–12. https://doi.org/10.1111/j.2041-210X.2012.00224.x.
[11] Li S, Abdulkadir N, Schattenberg F, da Rocha UN, Grimm V, Müller S, et al. Stabilizing microbial communities by looped mass transfer. Proceedings of the National Academy of Sciences 2022;119(17):. https://doi.org/10.1073/pnas.2117814119e2117814119.
[12] Liu Z, Müller S. Bacterial community diversity dynamics highlight degrees of nestedness and turnover patterns. Cytometry Part A 2020;97(7):742–8. https://doi.org/10.1002/cyto.a.23965.

[13] Liu Z, Cichocki N, Bonk F, Günther S, Schattenberg F, Harms H, Centler F, Müller S. Ecological stability properties of microbial communities assessed by flow cytometry. mSphere 2018;3(1). https://doi.org/10.1128/mSphere.00564-17. e00564–17.
[14] Grimm V, Wissel C. Babel, or the ecological stability discussions: an inventory and analysis of terminology and a guide for avoiding confusion. Oecologia 1996;109(3):323–34. https://doi.org/10.1007/s004420050090.
[15] Jia X, Wang Y, Ren L, Li M, Tang R, Jiang Y, Hou J. Early warning indicators and microbial community dynamics during unstable stages of continuous hydrogen production from food wastes by thermophilic dark fermentation. Int J Hydrogen Energy 2019;44(57):30000–13. https://doi.org/10.1016/j.ijhydene.2019.08.082.
[16] Dempster AP, Laird NM, Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Series B (Methodological) 1977;39(1):1–38.
[17] Rogers W, Holyst H. FlowFP: a bioconductor package for fingerprinting flow cytometric data. Advances in Bioinformatics 2009:193947. https://doi.org/10.1155/2009/193947.
[18] Aghaeepour N, Nikolic R, Brinkman RR, Hoos H. Rapid cell population identification in flow cytometry data. Cytometry Part A 2011;79(1):6–13. https://doi.org/10.1002/cyto.a.21007.
[19] Zare H, Shooshtari P, Gupta A, Brinkman RR. Data reduction for spectral clustering to analyse high throughput flow cytometry data. BMC Bioinformatics 2010;11(403). https://doi.org/10.1186/1471-2105-11-403.
[20] Finak G, Bashashati A, Brinkman RR, Gottardo R. Merging mixture components for cell population identification in flow cytometry. Adv Bioinform 2009:. https://doi.org/10.1155/2009/247646247646.
[21] Lo K, Hahne F, Brinkman RR, Gottardo R. FlowClust: a bioconductor package for automated gating of flow cytometry data. BMC Bioinform 2009;10(145). https://doi.org/10.1186/1471-2105-10-145.
[22] Rubbens P, Props R, Kerckhof F-M, Boon N, Waegeman W. PhenoGMM: gaussian mixture modeling of cytometry data quantifies changes in microbial community structure. mSphere 2021;6(1):e00530–20. https://doi.org/10.1128/mSphere.00530-20.
[23] Del Barrio E, Inouzhe H, Loubes J-M, Matrán C, Mayo A. optimalFlow: optimal transport approach to flow cytometry gating and population matching. BMC Bioinform 2020;21(479). https://doi.org/10.1186/s12859-020-03795-w.
[24] Reiter M, Rota P, Kleber F, Diem M, Groeneveld-Krentz S, Dworzak M. Clustering of cell populations in flow cytometry data using a combination of gaussian mixtures. Pattern Recogn 2016;60:1029–40. https://doi.org/10.1016/j.patcog.2016.04.004.
[25] Ludwig J, Höner zu Siederdissen C, Liu Z, Stadler PF, Müller S. flowEMMi: an automated model-based clustering tool for microbial cytometric data. BMC Bioinform 2019;20(643). https://doi.org/10.1186/s12859-019-3152-3.
[26] Cichocki N, Hübschmann T, Schattenberg F, Kerckhof F-M, Overmann J, Müller S. Bacterial mock communities as standards for reproducible cytometric microbiome analysis. Nat Protoc 2020;15(9):2788–812. https://doi.org/10.1038/s41596-020-0362-0.
[27] Becton, Dickinson and Company, FlowJO software, for Windows, version 10.8 (2021).
[28] Spidlen J, Breuer K, Rosenberg C, Kotecha N, Brinkman RR. Flowrepository: a resource of annotated flow cytometry datasets associated with peer-reviewed publications. Cytometry Part A 2012;81(9):727–31. https://doi.org/10.1002/cyto.a.22106.
[29] Ellis B., Haaland P., Hahne F., Le Meur N., Gopalakrishnan N., Spidlen J., Jiang M., Finak G., flowCore: basic structures for flow cytometry data, R package version 1.50.0 (2019).
[30] R Core Team, R: a language and environment for statistical computing, R Foundation for Statistical Computing, Vienna, Austria (2020).
[31] Eddelbuettel D, François R. Rcpp: seamless R and C++ integration. J Stat Softw 2011;40(8):1–18. https://doi.org/10.18637/jss.v040.i08.
[32] Eddelbuettel D. Seamless R and C++ integration with Rcpp. New York: Springer; 2013, ISBN 978-1-4614-6867-7.
[33] Eddelbuettel D, Balamuta JJ. Extending R with C++: a brief introduction to Rcpp. Am Stat 2017;72(1):28–36. https://doi.org/10.1080/00031305.2017.1375990.
[34] Bates D, Eddelbuettel D. Fast and elegant numerical linear algebra using the RcppEigen package. J Stat Softw 2013;52(5):1–24. https://doi.org/10.18637/jss.v052.i05.
[35] Genz A., Bretz F., Miwa T., Mi X., Leisch F., Scheipl F., Hothorn T. mvtnorm: multivariate normal and t distributions, R package version 1.1-0 (2020).
[36] Genz A, Bretz F. Computation of multivariate normal and t probabilities. Heidelberg: Springer-Verlag; 2009. ISBN 978-3-642-01688-2.
[37] Warnes GR, Bolker B, Lumley T, gtools: various R programming tools, R package version 3.8.2 (2020).
[38] Sanchez G. colortools: tools for colors in a hue-saturation-value (HSV) color model, R package version 0.1.5 (2013).
[39] Benaglia T, Chauveau D, Hunter DR, Young D. mixtools: an R package for analyzing finite mixture models. J Stat Softw 2009;32(6):1–29. https://doi.org/10.18637/jss.v032.i06.
[40] Warnes G.R., Bolker B., Bonebakker L., Gentleman R., Huber W., Liaw A. et al., gplots: various R programming tools for plotting data, R package version 3.0.3 (2020).
[41] Wickham H. ggplot2: elegant graphics for data analysis. New York: Springer-Verlag; 2016, ISBN 978-0-387-98141-3.
[42] Wand M., Moler C., Ripley B. KernSmooth: functions for kernel smoothing supporting Wand & Jones (1995), R package version 2.23-16 (2019).
[43] Izrailev S. tictoc: functions for timing R scripts, as well as implementations of stack and list structures., R package version 1.0 (2014).
[44] Oksanen J, Blanchet FG, Friendly M, Kindt R, Legendre P, McGlinn D, et al., Vegan: community ecology package, R package version 2.5-6 (2019).
[45] Finak G., Jiang M. flowWorkspace: infrastructure for representing and interacting with gated and ungated cytometry data sets., R package version 3.30.2 (2019).
[46] Schwarz G. Estimating the dimension of a model. Ann Stat 1978;6(2):461–4. https://doi.org/10.1214/aos/1176344136.
[47] Spruyt V. How to draw a covariance error ellipse?, retrieved in Dec. 2021 (2014). URL: https://www.visiondummy.com/2014/04/draw-error-ellipse-representing-covariance-matrix/#Arbitrary_confidence_ellipses.
[48] Wikipedia contributors, Ellipse, retrieved in Dec. 2021, permalink: https://en.wikipedia.org/w/index.php?title=Ellipse&oldid=1061346573 (Dec 2021). URL: https://en.wikipedia.org/wiki/Ellipse#General_ellipse.
[49] Hughes G, Chraibi M. Calculating ellipse overlap areas. Comput Vis Sci 2011;15(5):291–301. https://doi.org/10.1007/s00791-013-0214-3.
[50] Petersen K.B., Pedersen M.S., The matrix cookbook, retrieved in Aug. 2021 (2012). URL: https://www.math.uwaterloo.ca/ hwolkowi/matrixcookbook.pdf.
[51] Freier N., Kalman filter: multiplying normal distributions, retrieved in Dec. 2021 (Oct 2013). URL: https://www.norbert-freier.de/dateien/kalman_filter_multiplying_normal_distributions_norbert_freier_2013.pdf.
[52] Neuwirth E. RColorBrewer: ColorBrewer palettes, R package version 1.1-2 (2014).
[53] Li S. Concise formulas for the area and volume of a hyperspherical cap. Asian J Math Stat 2011;4(1):66–70.