Ye and Ho *BMC Systems Biology* 2019, **13**(Suppl 2):35
https://doi.org/10.1186/s12918-019-0690-2

BMC Systems Biology

METHODS Open Access

Ultrafast clustering of single-cell flow cytometry data using FlowGrid

Xiaoxin Ye[1,2] and Joshua W. K. Ho[1,2,3*]

From The 17th Asia Pacific Bioinformatics Conference (APBC 2019)
Wuhan, China. 14–16 January 2019

Abstract

Background: Flow cytometry is a popular technology for quantitative single-cell profiling of cell surface markers. It enables expression measurement of tens of cell surface protein markers in millions of single cells. It is a powerful tool for discovering cell sub-populations and quantifying cell population heterogeneity. Traditionally, scientists use manual gating to identify cell types, but the process is subjective and is not effective for large multidimensional data. Many clustering algorithms have been developed to analyse these data but most of them are not scalable to very large data sets with more than ten million cells.

Results: Here, we present a new clustering algorithm that combines the advantages of density-based clustering algorithm DBSCAN with the scalability of grid-based clustering. This new clustering algorithm is implemented in python as an open source package, FlowGrid. FlowGrid is memory efficient and scales linearly with respect to the number of cells. We have evaluated the performance of FlowGrid against other state-of-the-art clustering programs and found that FlowGrid produces similar clustering results but with substantially less time. For example, FlowGrid is able to complete a clustering task on a data set of 23.6 million cells in less than 12 seconds, while other algorithms take more than 500 seconds or get into error.

Conclusions: FlowGrid is an ultrafast clustering algorithm for large single-cell flow cytometry data. The source code is available at https://github.com/VCCRI/FlowGrid.

Keywords: Clustering, Flow cytometry, Single cell, DBSCAN

Background

Recent technological advancement has made it possible to quantitatively measure the expression of a handful of protein markers in millions of cells in a flow cytometry experiment [1]. The ability to profile such a large number of cells allows us to gain insights into cellular heterogeneity at an unprecedented resolution. Traditionally, cell types are identified based on manual gating of several markers in flow cytometry data. Manual gating relies on visual inspection of a series of two dimensional scatter plots, which makes it difficult to discover structure in high dimensions. It also suffers subjectivity, in terms of the order in which pairs of protein markers are explored, and the inherent uncertainty of manually drawing the cluster boundaries [2]. An emerging solution is to use unsupervised clustering algorithms to automatically identify clusters in potentially multidimensional flow cytometry data.

The Flow Cytometry Critical Assessment of Population Identification Methods (Flow-CAP) challenge has compared the performance of many flow cytometry clustering algorithms [3]. In the challenge, ADIcyt has the highest accuracy but has a long runtime, which makes it impractical for routine usage. Flock [4] maintains a high accuracy and reasonable runtime. After the challenge, several algorithms have been built for flow cytometry data analysis such as FlowPeaks [5], FlowSOM [6] and BayesFlow [7].

*Correspondence: jwkho@hku.hk
[1]Victor Chang Cardiac Research Institute, Sydney, Australia
[2]University of New South Wales, Sydney, Australia
Full list of author information is available at the end of the article

FlowPeaks and Flock are largely based on k-means clustering. k-means clustering requires the number of clusters (k) to be defined prior to the analysis. It is hard to determine a suitable k in practice. FlowPeaks performs k-means clustering with a large initial k, and iteratively merges nearby clusters that are not separated by low density regions into one cluster. Flock utilises grids to identify high density regions, which the algorithm then uses to identify initial cluster centres for k-means clustering. This grid-based method of identifying high density region allows k-means clustering to converge much quicker compared to using random initialisation of cluster centres, and also directly identifies a suitable value for k. FlowSOM starts with training Self-Organising Map (SOM), followed by consensus hierarchical clustering of the cells for meta-clustering. In the algorithm, the number of clusters (k) is required for meta-clustering.

BayesFlow uses a Bayesian hierarchical model to identify different cell populations in one or many samples. The key benefit of this method is its ability to incorporate prior knowledge, and captures the variability in shapes and locations of populations between the samples [7]. However, BayesFlow tends to be computational expensive as Markov Chain Monte Carlo sampling requires a large number of iterations. Therefore, BayesFlow is often impractical for flow cytometry data sets of realistic size.

These algorithms perform well on the Flow-CAP data sets, but they may not be scalable to larger data sets that we are dealing with nowadays – those with tens of millions of cells. Aiming to quantify cell population heterogeneity in huge data sets, we have to develop an ultrafast and scalable clustering algorithm.

In this paper, we present a new clustering algorithm that combines the benefit of DBSCAN [8] (a widely-based density-based clustering algorithm) and a grid-based approach to achieve scalability. DBSCAN is fast and can detect clusters with complex shapes in the presence of outliers [8]. DBSCAN starts with identifying core points that have a large number of neighbours within a user-defined region. Once the core points are found, nearby core points and closely located non-core points are grouped together to form clusters. This algorithm will identify clusters that are defined as high-density regions that are separated by the low-density regions. However, DBSCAN is memory inefficient if the data set is very large, or has large highly connected components.

To reduce the computational search space and memory requirement, our algorithm extends the idea of DBSCAN by using equal-spaced grids like Flock. We implemented our algorithm in an open source python package called FlowGrid. Using a range of real data sets, we demonstrate that FlowGrid is much faster than other state-of-the-art flow cytometry clustering algorithms, and produce similar clustering results. The detail of the algorithm is presented in the Methods section.

Methods

The key idea of our algorithm is to replace the calculation of density from individual points to discrete bins as defined by a uniform grid. This way, the clustering step of the algorithm will scale with the number of non-empty bins, which is significantly smaller than the number of points in lower dimensional data sets. Therefore the overall time complexity of our algorithm is dominated by the binning step, which is in the order of $O(N)$. This is significantly better than the time complexity of DBSCAN, which is in the order of $O(NlogN)$. The definition and algorithm are presented in the following subsections.

Definition

The key terms involved in the algorithm are defined in this subsection. A graphical example can be found in Fig. 1.

- N_{bin} is the number of equally sized bins in each dimension. In theory, there are $(N_{bin})^d$ bins in the data space, where d is the number of dimensions. However, in practice, we only consider the non-empty bins. The number of non-empty bins (N) is less than $(N_{bin})^d$, especially for high dimensional data. Each non-empty bin is assigned an integer index $i = 1 \ldots N$.
- Bin_i is labelled by a tuple with d positive integers $C_i = (C_{i1}, C_{i2}, C_{i3}, \ldots, C_{id})$ where C_{i1} is the coordinate (the bin index) at dimension 1. For

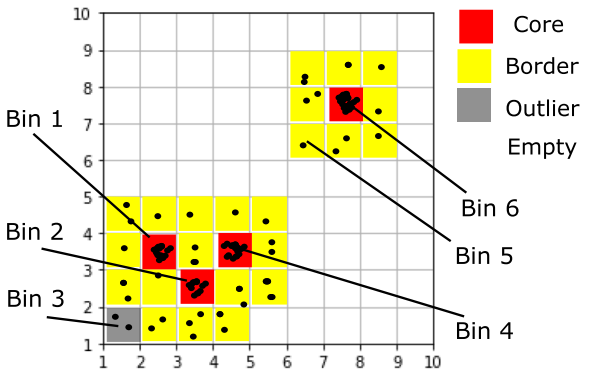

Fig. 1 An illustrative example of the FlowGrid clustering algorithm. In this example, Bin 1, Bin 2, Bin 3 and Bin 6 are core bins as their Den_b are larger than $MinDen_b$ (5 in this example), their Den_c are larger than $MinDen_c$ (20 in thi s example), and their Den_b are larger than ρ% (75% in this example) of its directly connected bins. $Dist(C_1, C_2) = \sqrt{1^2 + 1^2} = \sqrt{2} \leq \sqrt{\epsilon}$ ($\epsilon = 2$ in this example), so Bin 1 and Bin 2 are directly connected. $Dist(C_2, C_4) = \sqrt{1^2 + 1^2} = \sqrt{2} \leq \sqrt{\epsilon}$, so Bin 2 and Bin 4 are directly connected. Therefore, Bin 1, Bin 2 and Bin 4 are mutually connected, and they are assigned into the same cluster. Bin 5 is not a core bin but is a border bin, as it is directly connected to Bin 6, which is a core bin. Bin 3 is a outlier bin, as it is not a core bin nor a border bin. In practice, $MinDen_b$ is set to be 3, $MinDen_c$ is set to 40 and ρ is set to be 85

example, if Bin_i has coordinate $C_i = (2, 3, 5)$, this bin is located in second bin in dimension 1, third bin in dimension 2 and the fifth bin in dimension 3.

- The distance between Bin_i and Bin_j is defined as

$$Dist(C_i, C_j) = \sqrt{\sum_{k=1}^{d} (C_{ik} - C_{jk})^2} \quad (1)$$

- Bin_i and Bin_j are defined to be directly connected if $Dist(C_i, C_j) \leqslant \sqrt{\epsilon}$, where ϵ is a user-specified parameter.
- $Den_b(C_i)$ is the density of Bin_i, which is defined as the number of points in Bin_i.
- $Den_c(C_i)$ is the collective density of Bin_i, calculated by

$$Den_c(C_i) = \sum_{\{j | Bin_j \text{ and} Bin_i \text{ are directly connected}\}} Den_b(C_j) \quad (2)$$

- Bin_i is a core bin if

  1 $Den_b(C_i)$ is larger than $MinDen_b$, a user-specified parameter.
  2 $Den_b(C_i)$ is larger than ρ% of its directly connected bins, where ρ is a user-specified parameter.
  3 $Den_c(C_i)$ is larger than $MinDen_c$, a user-specified parameter.

- Bin_i is a border bin if it is not a core bin but it is directly connected to a core bin.
- Bin_i is an outlier bin, if it is not a core bin nor a border bin.
- Bin_a and Bin_b are in the same cluster, if they satisfy one of the following conditions:

  1 they are directly connected and at least one of them is core bin;
  2 they are not directly connected but are connected by a sequence of directly connected core bins.

- Two points are in the same cluster, if they belong to the same bin or their corresponding bins belong to the same cluster.

Algorithm 1: FlowGrid

input : $X, N_{bin}, \epsilon, \rho, MinDen_b, MinDen_c$
output: DataLabel
1 Normalise the data X ranging from 1 to $(N_{bin} + 1)$
2 Assign data into corresponding bins based on the integer of normalised value
3 Identify S_{bin} as the set of non-empty bins
4 Search the core bins and their directly connected bins by SearchCore
5 Group connected bins into a cluster by Breadth First Search(BFS)
6 Label cells by the label of their corresponding bins

Algorithm 2: SearchCore

input : $S_{bin}, \epsilon, \rho, MinDen_b, MinDen_c$
output: S_{core}, L
Initial an empty adjacency list L.
$S_{core} = \{\}$
forall the Bin_i *in* S_{bin} **do**
  if $Den_b(i) > MinDen_b$ **then**
    nnBin=radiusNeighbors(S_{bin}, Bin_i, ϵ)
    nnCount counting the number of points for each bin in nnBin
    if $Den_b(i)$ *is greater than* ρ% *of nnCount* **then**
      $Den_c(i)$= the sum of nnCount
      if $Den_c(i) > MinDen_c$ **then**
        $S_{core} = S_{core} \cup \{i\}$
        mapping bin_i with nnBin in L
      end
    end
  end
end
The input of radiusNeighbors is all non-empty bins, the query bin and the maximum query distance $\sqrt{\epsilon}$. The output is the bins whose distance with the query bin are less than $\sqrt{\epsilon}$ (including the query bin).

Algorithm

Algorithm 1 describes the key steps of FlowGrid, starting with normalising the values in each dimension to range between 1 and $(N_{bin} + 1)$. Then, we use the integer part of the normalised value as the coordinate of its corresponding bin. Then, the SearchCore algorithm is applied to discover the core bins and their directly connected bins. Once the core bins and connections are found, Breadth First Search(BFS) is used to group the connected bins into a cluster. The cells are labelled by the label of their corresponding bins.

Evaluation

Procedure

FlowGrid aims to be an ultrafast and accurate clustering algorithm for very large flow cytometry data. Therefore, both the accuracy and scalability performance need to be evaluated. The benchmark data sets from Flow-CAP [3], the multi-centre CyTOF data from Li et al. [9] and the SeaFlow project [10] are selected to compare the performance of FlowGrid against other state-of-the-art algorithms, FlowSOM, FlowPeaks, and FLOCK. These three algorithms are chosen because they are widely used,

Algorithm 3: Breadth First Search(BFS)

```
input  : S_core, S_bin, adjacency list L
output: Bin Label
Label every bin as -1
Index=1
for Bin_i in S_core do
    if the laebl of Bin_i is -1 then
        Queue={}
        Label Bin_i as Index
        Queue.push(Bin_i)
        while Queue is not empty do
            Bin_1= Queue.pop()
            forall the directed connected Bin_2 of Bin_1
            do
                if the laebl of Bin_2 is -1 then
                    Label Bin_2 as Index
                    if Bin_2 is core bin then
                        Queue.push(Bin_2)
                    end
                end
            end
        end
        index=index +1
    end
end
```

are generally considered to be quite fast, and have good accuracy.

Three benchmark data sets from Flow-CAP [3] are selected for evaluation, including the Diffuse Large B-cell Lymphoma (DLBL), Hematopoietic Stem Cell Transplant (HSCT), and Graft versus Host Disease(GvHD) data set. Each data set contains 10-30 samples with 3-4 markers, and each sample includes 2,000-35,000 cells.

The multi-centre CyTOF data set from Li et al. [9] provides a labelled data set with 16 samples. Each samples contains 40,000-70,000 cells and 26 markers. Since only 8 out fo 26 markers are determined to be relevant markers in the original paper [9], only these 8 markers are used for clustering.

We also use three data sets from the SeaFlow project [10] and they contain many samples. Instead of analysing the independent samples, we analyse the concatenated data sets as the original paper [10] and these concatenated data sets contain 12.7, 22.7 and 23.6 millions of cells respectively. Each data sets include 15 features but the original study only uses four features for clustering analysis. The four features are forward scatter (small and perpendicular), phycoerythrin, and chlorophyll (small) [10].

In the evaluation, we treat the manual gating label as the gold standard for measuring the quality of clustering. In the pre-precessing step, we apply the inverse hyperbolic function with the factor of 5 to transform the multi-centre data and the SeaFlow data. As the Flow-CAP and multi-centre CyTOF data contain many samples and we treat each sample as a data set, we run all algorithms on each sample. The performances are measured by the ARI and runtime, which are reported by the arithmetic means ($\bar{x}$) and standard deviation (sd). For the Seaflow data sets, we treat each concatenated data set as a data set. In the evaluation, all algorithms are applied on these concatenated data sets.

To evaluate the scalability of each algorithm, we down-sample the largest concatenated data set from the SeaFlow project, generating 10 sub-sampled data sets in which the numbers of cells range from 20 thousand to 20 million.

Performance measure

The efficiency performance is measured by the runtime while the clustering performance is measured by Adjusted Rand Index (ARI). ARI is used to measure the clustering performance. ARI is the corrected-for-chance version of the Rand index [11]. Although it may result in negative values if the index is less than expected, it tends to be more robust than many other measures like F-measure and Rand index.

ARI is calculated as follow. Given a set S of n elements, and two groups of cluster labels (one group of ground truth label and one group of predicted labels) of these elements, namely $X = \{X_1, X_2, \ldots, X_r\}$ and $Y = \{Y_1, Y_2, \ldots, Y_s\}$, the overlap between X and Y can be summarized by n_{ij} where n_{ij} denotes the number of objects in common between X_i and Y_j: $n_{ij} = |X_i \cap Y_j|$.

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}$$

where $a_i = \sum_j n_{ij}$ and $b_j = \sum_i n_{ij}$

Experimentation

FlowGrid is publicly available as an open source program on GitHub. FlowSOM and FlowPeaks are available as R packages from Bioconductor. The source code of Flock is downloaded from its Sourceforge repository. To reproduce all the comparisons presented in this paper, the source code and data can be downloaded from the GitHub repository *FlowGrid_compare*. We run all the experiments on six 2.60 GHz cores CPU with 32 G RAM.

FlowPeaks and Flock provide automated version without any user-input parameter. FlowSOM requires one user-supplied parameter (k, the number of clusters in meta-clustering step). FlowGrid requires two user-supplied parameters (bin_n and ϵ). To optimise the result,

Table 1 Comparison of runtime (in seconds) of FlowGrid against other clustering algorithms

Data set	Samples	Markers	Cells	Time in second ($\bar{x} \pm sd$)			
				FlowGrid	FlowSOM	FlowPeaks	Flock
Multi-center	16	8	29-77 $\times 10^3$	0.23± 0.09	4.01± 1.08	2.27± 0.61	10.3± 3.45
Flow-CAP-GvHD	12	4	12-33$\times 10^3$	0.07± 0.04	2.16± 0.54	0.28± 0.16	0.58± 0.28
Flow-CAP-DLBL	30	3	2-25$\times 10^3$	0.04± 0.01	1.25± 0.32	0.10± 0.09	0.22± 0.16
Flow-CAP-HSCT	30	4	6-9$\times 10^3$	0.04± 0.02	1.35± 0.28	0.11± 0.02	0.28± 0.06
Seaflow0	-	4	23.6$\times 10^6$	11.51	572.65	NA	6628.30
Seaflow1	-	4	12.7$\times 10^6$	3.09	312.95	258.13	NA
Seaflow11	-	4	22.7$\times 10^6$	6.37	544.79	NA	NA

NA represents that the algorithm got error in the data set

we try many k for FlowSOM and many combinations of bin_n and ϵ for our algorithm.

Results

Performance comparison

Table 1 summaries the performance of our algorithm and three other algorithms – FlowSOM, FlowPeaks, and Flock in terms of runtime. Our algorithm is substantially faster than other clustering algorithms in all the data sets. This improvement of runtime is especially substantial in the Seaflow data sets. FLOCK and FlowPeaks sometimes fail to complete in some of the data sets. In a data set of 23.6 million cells, FlowSOM completes the execution in 572 s, whereas FlowGrid completes the execution in only 12 s. This is a 50× speed up. Table 2 summaries the clustering accuracy performance. In Flow-CAP and the multi-centre data sets, FlowGrid shares the similar clustering accuracy (in terms of ARI) with other clustering algorithms but in Seaflow data sets, FlowGrid gives higher accuracy than other clustering algorithms.

Figure 2 shows that the clustering results of our algorithm and three other algorithms in a HSCT sample. FlowGrid, FlowSOM and FlowPeaks recover similar number of clusters, and the clustering results are largely similar.

Table 2 Comparison of accuracy (in ARI) of FlowGrid against other clustering algorithms

Data set	ARI ($\bar{x} \pm sd$)			
	FlowGrid	FlowSOM	FlowPeaks	Flock
Multi-center	0.66± 0.20	0.75± 0.17	0.68± 0.20	0.66± 0.16
Flow-CAP-GvHD	0.79± 0.15	0.85± 0.11	0.72± 0.16	0.47± 0.20
Flow-CAP-DLBL	0.85± 0.10	0.84± 0.10	0.82± 0.15	0.84± 0.09
Flow-CAP-HSCT	0.90± 0.08	0.87± 0.14	0.83± 0.24	0.57± 0.27
Seaflow0	0.94	0.81	NA	0.27
Seaflow1	0.59	0.54	0.34	NA
Seaflow11	0.77	0.33	NA	NA

NA represents that the algorithm got error in the data set

Flock generates too many clusters in this case. It is important to note that FlowGrid also identifies cells that do not belong to a main cluster (i.e., a high density region). These cells can be viewed as 'outliers', and are labelled as '-1' in Fig. 2. This is a feature that is not present in other clustering algorithms.

Scalability analysis

To further evaluate the scalability of the algorithms, we sub-sample one Seaflow data set and the sampled data sets range from 20 thousand to 20 million cells. Figure 3 shows the scalability of our algorithm and three other algorithms. Flock has a low runtime when processing a small data set, but its runtime dramatically increases to 6640 s for a 20 million-cell data set. FlowPeaks and FlowSOM share similar scalability but FlowPeaks is not able to execute 20 million data set. Our algorithm have the best performance in the evaluation as FlowGrid is faster than other algorithm in all the sampled data by an order of magnitude.

Parameter robustness analysis

Like other density-based clustering algorithm, parameter setting is important. In our experience, Bin_n and ϵ are data-set-dependent. We recommend trying out different combinations of Bin_n between 4 and 15, and ϵ between 1 and 5. To pick the best parameter combinations, some prior knowledge is helpful such as the expected number of clusters and the proportion of outliers which should be less than 10% in our experience.

We found that other parameters, namely $MinDen_b$, $MinDen_c$ and ρ are mostly robust across a wide range of values.

To demonstrate this robustness, we used the benchmark data sets from Flow-CAP for a parameter sensitivity analysis. For these experiments, we first set 3, 40, 85, 4 and 1 as the default value for $MinDen_b$, $MinDen_c$, ρ, Bin_n and ϵ, respectively. In each experiment, we only change

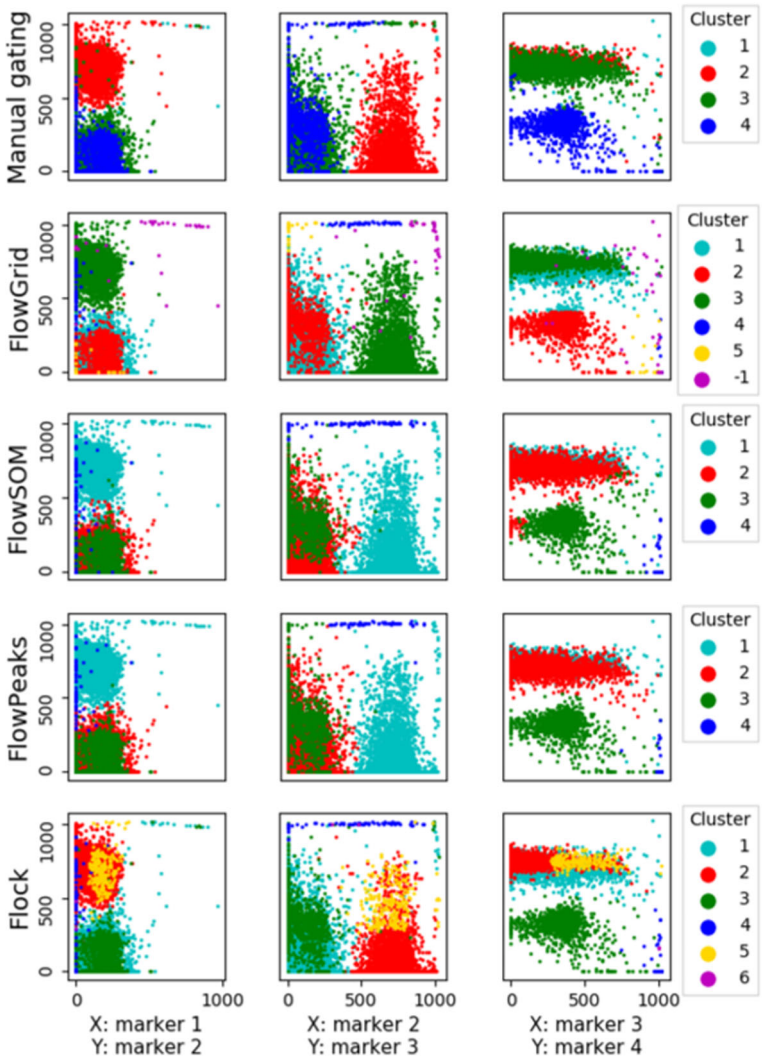

Fig. 2 Visual comparison of the clustering performance of FlowGrid, FlowPeaks, FlowSOM, and Flock using manual gating (top row) as the gold standard

one parameter to test its sensitivity to the overall classification result. The performance is measured by ARI and runtime. In the first experiment, we varied $MinDen_b$ from 1 to 50 while fixing other parameters. In the second experiment, we varied $MinDen_c$ ranging from 10 to 300 while fixing other parameters. In the third experiment, we varied ρ ranging from 70 to 95 while fixing other parameters.

Figure 4 demonstrates that the clustering accuracy and runtime are largely insensitive to $MinDen_b$, $MinDen_c$ and ρ across a large range of parameter values. The experiments are applied to all the benchmark data sets from Flow-CAP and similar results are observed in all the benchmark data sets. In our experiments, when $MinDen_b$, $MinDen_c$ and ρ are set to be 3, 40 and 85 respectively, FlowGrid maintains good clustering performance and excellent runtime. They are therefore set as the default parameters for FlowGrid.

Discussion

In this paper, we have developed an ultrafast clustering algorithm, FlowGrid, for single-cell flow cytometry analysis, and compared it against other state-of-the-art algorithms such as Flock, FlowSOM and FlowPeaks. FlowGrid borrows ideas from DBSCAN for detection of high density regions and outliers. It does not only perform well in the presence of outliers, but also have great scalability without getting into memory issues. It is both time efficient and memory efficient. FlowGrid

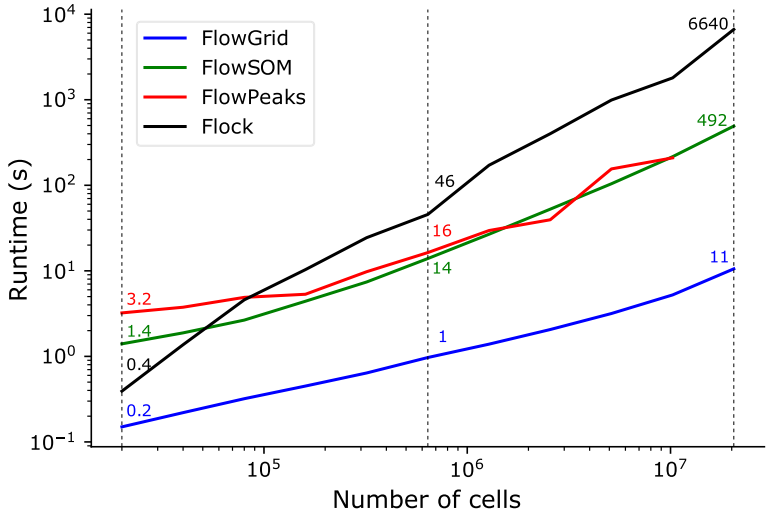

Fig. 3 Comparison of the runtime of FlowGrid, FlowPeaks, FlowSOM, and Flock using data sets with different number of cells

shares similar clustering accuracy with state-of-the-art flow cytometry clustering algorithms, but it is substantially faster than them. With any given number of markers, the runtime of FlowGrid scales linearly with the number of cells, which is a useful property for extremely large data sets.

$MinDen_b$ and $MinDen_c$ are density threshold parameters to reduce the search space of high density bins. If the parameters are set very low, the runtime may fractionally increase but the accuracy is not likely to be affected. However, if the parameters are set very high, the runtime will also fractionally decreases but it may lead to separation of real clusters and create spurious outliers. In any case, we showed that the performance of FlowGrid is generally robust against changes in $MinDen_b$, $MinDen_c$ and ρ.

The current implementation of FlowGrid is already very fast for most practical purposes. In the future, if the data size grows even larger, it is possible to further speed up FlowGrid by parallelising the binning step of the algorithm, which is currently the most computationally intensive step of the algorithm.

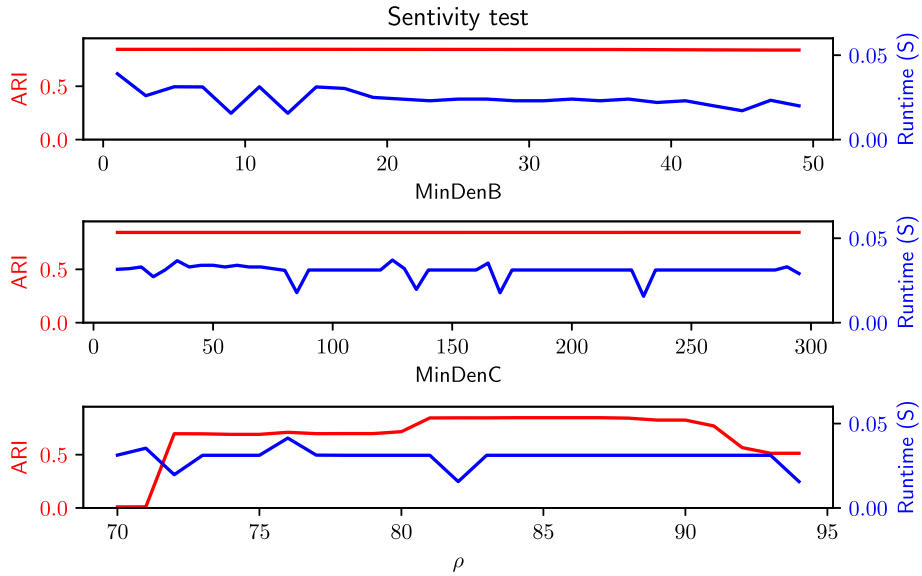

Fig. 4 Sensitivity analysis of three different parameters on clustering accuracy (as measured by adjusted rand index; ARI) and runtime (seconds)

Abbreviations
ARI: Adjusted rand index; BFS: Breadth first search; CyTOF: Mass cytometry; DBSCAN: Density-based spatial clustering of applications with noise; Flow-CAP: Flow cytometry critical assessment of population identification methods; SOM: Self-organising map

Acknowledgments
We thank members of the Ho Laboratory for their valuable comments.

Funding
This work was supported in part by funds from the New South Wales Ministry of Health, a National Health and Medical Research Council Career Development Fellowship (1105271 to JWKH), and a National Heart Foundation Future Leader Fellowship (100848 to JWKH). Publication charge is supported by the Victor Chang Cardiac Research Institute.

Availability of data and materials
- Project Name: FlowGrid
- Project Home Page: https://github.com/VCCRI/FlowGrid
- Operating Systems: Unix, Mac, Windows
- Programming Languages: Python
- Other Requirements: sklearn, numpy
- License: MIT Public License
- Any Restrictions to Use By Non-Academics: None

About this supplement
This article has been published as part of *BMC Systems Biology Volume 13 Supplement 2, 2019: Selected articles from the 17th Asia Pacific Bioinformatics Conference (APBC 2019): systems biology*. The full contents of the supplement are available online at https://bmcsystbiol.biomedcentral.com/articles/supplements/volume-13-supplement-2.

Authors' contributions
XY and JWKH initiated and designed the project. XY implemented the algorithm, carried out all the experiments, and wrote the paper. JWKH revised the paper. Both authors approved the final version of the paper.

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

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

Publisher's Note

Author details
[1]Victor Chang Cardiac Research Institute, Sydney, Australia. [2]University of New South Wales, Sydney, Australia. [3]School of Biomedical Sciences, Li Ka Shing Faculty of Medicine, The University of Hong Kong, Pokfulam, Hong Kong.

Published: 5 April 2019

References
1. Weber LM, Robinson MD. Comparison of clustering methods for high-dimensional single-cell flow and mass cytometry data. Cytom Part A. 2016;89(12):1084–96.
2. Saeys Y, Van Gassen S, Lambrecht BN. Computational flow cytometry: helping to make sense of high-dimensional immunology data. Nat Rev Immunol. 2016;16(7):449.
3. Aghaeepour N, Finak G, Hoos H, Mosmann TR, Brinkman R, Gottardo R, Scheuermann RH, Consortium F, Consortium D, et al. Critical assessment of automated flow cytometry data analysis techniques. Nat Methods. 2013;10(3):228.
4. Qian Y, Wei C, Eun-Hyung Lee F, Campbell J, Halliley J, Lee JA, Cai J, Kong YM, Sadat E, Thomson E, et al. Elucidation of seventeen human peripheral blood B-cell subsets and quantification of the tetanus response using a density-based method for the automated identification of cell populations in multidimensional flow cytometry data. Cytom Part B Clin Cytom. 2010;78(S1):69–82.
5. Ge Y, Sealfon SC. flowPeaks: a fast unsupervised clustering for flow cytometry data via k-means and density peak finding. Bioinformatics. 2012;28(15):2052–8.
6. Van Gassen S, Callebaut B, Van Helden MJ, Lambrecht BN, Demeester P, Dhaene T, Saeys Y. FlowSOM: Using self-organizing maps for visualization and interpretation of cytometry data. Cytom Part A. 2015;87(7):636–45.
7. Johnsson K, Wallin J, Fontes M. BayesFlow: latent modeling of flow cytometry cell populations. BMC Bioinformatics. 2016;17(1):25.
8. Ester M, Kriegel H-P, Sander J, Xu X, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96). Portland: AAAI Press; 1996. p. 226–231.
9. Li H, Shaham U, Stanton KP, Yao Y, Montgomery RR, Kluger Y. Gating mass cytometry data by deep learning. Bioinformatics. 2017;33(21):3423–30.
10. Hyrkas J, Clayton S, Ribalet F, Halperin D, Virginia Armbrust E, Howe B. Scalable clustering algorithms for continuous environmental flow cytometry. Bioinformatics. 2015;32(3):417–23.
11. Hubert L, Arabie P. Comparing partitions. J Classif. 1985;2(1):193–218.