

## RESEARCH ARTICLE

# Determination of essential phenotypic elements of clusters in high-dimensional entities—DEPECHE

Axel Theorell<sup>1</sup>, Yenan Troi Bryceson<sup>2,3</sup>, Jakob Theorell<sup>2,4\*</sup>

**1** IBG-1: Biotechnology, Institute of Bio- and Geosciences, Forschungszentrum Jülich GmbH, Jülich, North Rhine-Westphalia, Germany, **2** Center for Hematology and Regenerative Medicine, Department of Medicine Huddinge, Karolinska Institutet, Stockholm, Sweden, **3** Broegelmann Research Laboratory, Department of Clinical Medicine, University of Bergen, Bergen, Norway, **4** Nuffield Department of Clinical Neurosciences, University of Oxford, Oxford, United Kingdom

\* [Jakob.Theorell@ki.se](mailto:Jakob.Theorell@ki.se)**OPEN ACCESS**

**Citation:** Theorell A, Bryceson YT, Theorell J (2019) Determination of essential phenotypic elements of clusters in high-dimensional entities—DEPECHE. PLoS ONE 14(3): e0203247. <https://doi.org/10.1371/journal.pone.0203247>

**Editor:** Zhi Wei, New Jersey Institute of Technology, UNITED STATES

**Received:** August 12, 2018

**Accepted:** January 11, 2019

**Published:** March 7, 2019

**Copyright:** © 2019 Theorell et al. This is an open access article distributed under the terms of the [Creative Commons Attribution License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

**Data Availability Statement:** All the data presented in the paper is freely available. All data sources are described in the [S2 File](#).

**Funding:** The authors received no specific funding for this work.

**Competing interests:** The authors have declared that no competing interests exist.

## Abstract

Technological advances have facilitated an exponential increase in the amount of information that can be derived from single cells, necessitating new computational tools that can make such highly complex data interpretable. Here, we introduce DEPECHE, a rapid, parameter free, sparse k-means-based algorithm for clustering of multi- and megavariable single-cell data. In a number of computational benchmarks aimed at evaluating the capacity to form biologically relevant clusters, including flow/mass-cytometry and single cell RNA sequencing data sets with manually curated gold standard solutions, DEPECHE clusters as well or better than the currently available best performing clustering algorithms. However, the main advantage of DEPECHE, compared to the state-of-the-art, is its unique ability to enhance interpretability of the formed clusters, in that it only retains variables relevant for cluster separation, thereby facilitating computational efficient analyses as well as understanding of complex datasets. DEPECHE is implemented in the open source R package DepecheR currently available at [github.com/Theorell/DepecheR](https://github.com/Theorell/DepecheR).

## Introduction

Since the introduction of the first single colour flow cytometers in the 1960s, there has been a remarkable increase in the complexity of data that can be generated with single-cell resolution. Currently, flow and mass cytometers able to simultaneously assess up to 40 cellular traits are becoming widely available [1]. In parallel, the development of high-throughput sequencing technology has facilitated single-cell transcriptomic analyses gauging expression of thousands of distinct transcripts [2]. Furthermore, development of high-resolution single-cell proteomic analyses are underway, similarly providing highly complex datasets [3].

These technological advances necessitate new computational approaches to analyses of multi- and megavariable single cell data [4–7]. Previous algorithms have contributed to automating analyses, thereby enhancing reproducibility and avoiding a need for *a priori* biological

knowledge. Thus, these algorithms have major advantages over the classical analysis strategies of cytometry data that are based on manually defined uni- or bivariate filters, commonly referred to as gates [8]. Automated analysis algorithms, not restricted to uni- or bivariate displays of the data, have also made it possible to extract much more of the information embedded in multivariate data. Furthermore, as they generally scale well, automated analysis algorithms can also be applied to datasets where manual gating is impossible, such as single-cell transcriptomic data with thousands of transcripts assessed for each cell. To date, however, analysis strategies based on manual gating still dominate where they can be used and are still considered as the gold standard for cytometry data analysis. One likely reason for this is that cell types defined by manual gates are easy to comprehend, as they are defined by few markers. In an attempt to combine the objectiveness, reproducibility and scalability of automated analysis pipelines with the high interpretability of manual gating strategies, we have developed an algorithm termed Determination of Essential Phenotypic Elements of Clusters in High-dimensional Entities (DEPECHE). DEPECHE simultaneously clusters and simplifies the data by identifying the variables that contribute to separate individual clusters from the rest of the data. We have implemented DEPECHE in R [9] (in the open source package *DepecheR*), providing a complete software suite for statistical analysis and visualization of single cell omics data.

## Results and discussion

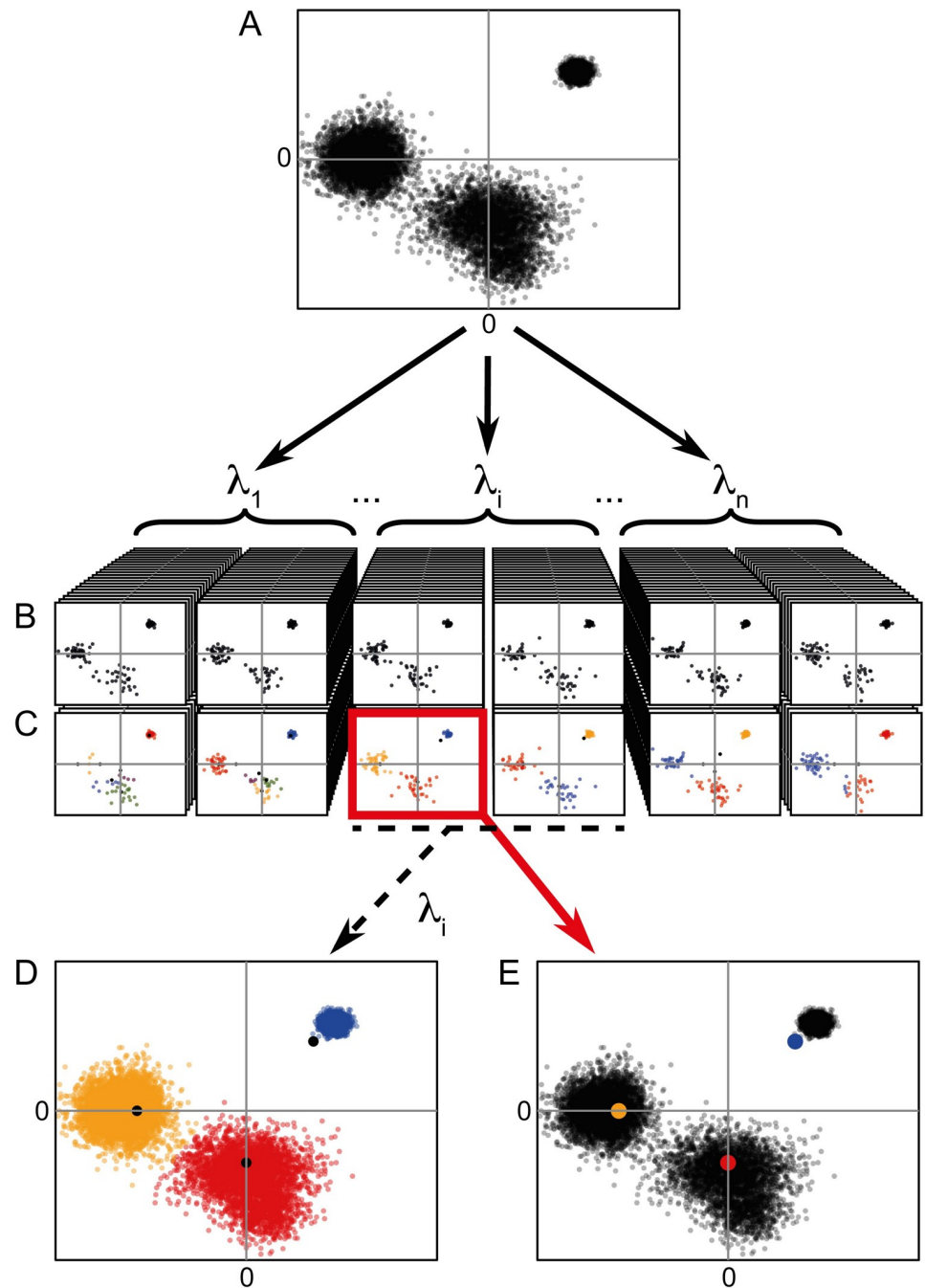
At its core, DEPECHE uses a penalized  $k$ -means clustering algorithm, related to the standard  $k$ -means algorithm [10]. Both  $k$ -means and penalized  $k$ -means clusters data by fitting  $k$  cluster centers to the data. In  $k$ -means, the fitting objective is to minimize the sum of squared distances from each data point to its closest cluster center. In penalized  $k$ -means, the  $k$ -means objective has been complemented by a penalization objective which minimizes the sum of the L1-norms of the cluster centers, thereby moving them towards the origin.

Generally, when measuring a large number of biological variables, observations tend to agglomerate around certain positions in some variables, whereas in other variables, the observations are spread more or less evenly. Due to their even spread, the latter variables can be considered uninformative. In this situation, the dual clustering objective of penalized  $k$ -means will fit the cluster centers to the observation agglomerates in the informative dimensions, whereas the penalty will draw the cluster centers to the origin in the uninformative dimensions and thus discard them. The resulting cluster definitions are referred to as sparse, as the clusters are only defined in the informative dimensions. The relative importance of the  $k$ -means objective and the penalization objectives in the penalized  $k$ -means algorithm is controlled by the penalty parameter  $\lambda$ . A high penalty  $\lambda$  implies that many variables will be considered uninformative, implying that many variables will be discarded. When all variables of a cluster are discarded, the cluster is discarded too. Thus, a high penalty  $\lambda$  results in high sparsity and a low cluster resolution, and vice versa [11] (see [Methods](#), “Clustering with DEPECHE”). A key feature of penalized  $k$ -means is that the penalty addresses each variable in each cluster center independently, implying that different sparsity patterns will emerge from distinct clusters. This feature differentiates penalized  $k$ -means from sparse  $k$ -means [12] and regularized  $k$ -means [13], which only identify variables that are uninformative for all clusters. To illustrate this, a 10,000 point, 11-variate dataset with 10 normally distributed clusters was constructed. In this dataset, 10 of the variables contributed to the separation of two clusters ([S1 Fig](#)). In this example, sparse  $k$ -means could only exclude the single variable, which did not define any clusters, whereas DEPECHE correctly excluded all nine dispensable variables from each cluster individually. Note that if  $k$  is chosen so large that at least one cluster is discarded, the resolution of the

emerging clusters depends entirely on the magnitude of the penalty  $\lambda$  and not on  $k$ , since DEPECHE redistributes observations belonging to clusters that are pulled to the origin to the remaining clusters. In DEPECHE, the penalty  $\lambda$  is tuned to identify the most *reproducible* clustering resolution, here termed the “optimal resolution” [13]. To illustrate what we mean by *reproducible*, we constructed an example, featuring a bi-variate dataset  $D$  (Fig 1A). Visually, dataset  $D$  contains three clusters, where the centers of the two larger clusters are located close to either axis. For these two clusters, one variable is sufficient to define their position. Now, if multiple datasets were generated from the same data source as  $D$ , for example by repeated experiments, we assume that they would contain the same clusters. Hence, the optimal penalty  $\lambda_i$  (that corresponds to the optimal resolution of  $D$ ) is defined as the penalty for which the resulting clusters have the highest similarity, where similarity is measured in the Adjusted Rand Index (ARI) [14] (see Methods, “Tuning the penalty”), when clustering the generated datasets independently. In contrast, when clustering the same datasets with a penalty  $\lambda$  that differs significantly from the optimal penalty, the stochastic differences between the datasets are likely to induce solutions that deviate in cluster number, number of defining variables, and cluster center positions. In practice, DEPECHE tests a range of penalty values,  $(\lambda_1 < \dots < \lambda_{N\lambda})$ , each on a collection of dataset pairs which are generated by sampling  $N_r$  data points from  $D$  (Fig 1B) with resampling. The optimal resolution is defined as the penalty  $\lambda_i$ , which yields the lowest average variability within each dataset pair, as measured by the ARI. In our example, this corresponds to the penalty  $\lambda_i$  that yields 3 clusters, since 3 similar clusters are identified in each resampled dataset of  $D$  (Fig 1C). The penalties  $\lambda_1$  and  $\lambda_{N\lambda}$  are considered suboptimal, since with these penalties, the stochastic differences in the resampled datasets lead to less coherent clustering results compared to results obtained with the optimal penalty  $\lambda_i$ . From here (Fig 1C) DEPECHE uses two alternative routes. If the dataset  $D$  has few data points, the full dataset  $D$  is clustered using the optimal penalty  $\lambda_i$  (Fig 1D). If instead the number of data points in the dataset  $D$  is high (default in DepecheR  $> 10^4$ ), the most generalizable cluster centers that were produced using the optimal penalty are chosen (see Methods, “Simultaneous Clustering and Parameter Tuning”) and the data points of  $D$  are allocated directly to their closest cluster center (Fig 1E). This default setting enables rapid iterated clustering of large datasets possible (see calculation times in S2 Fig).

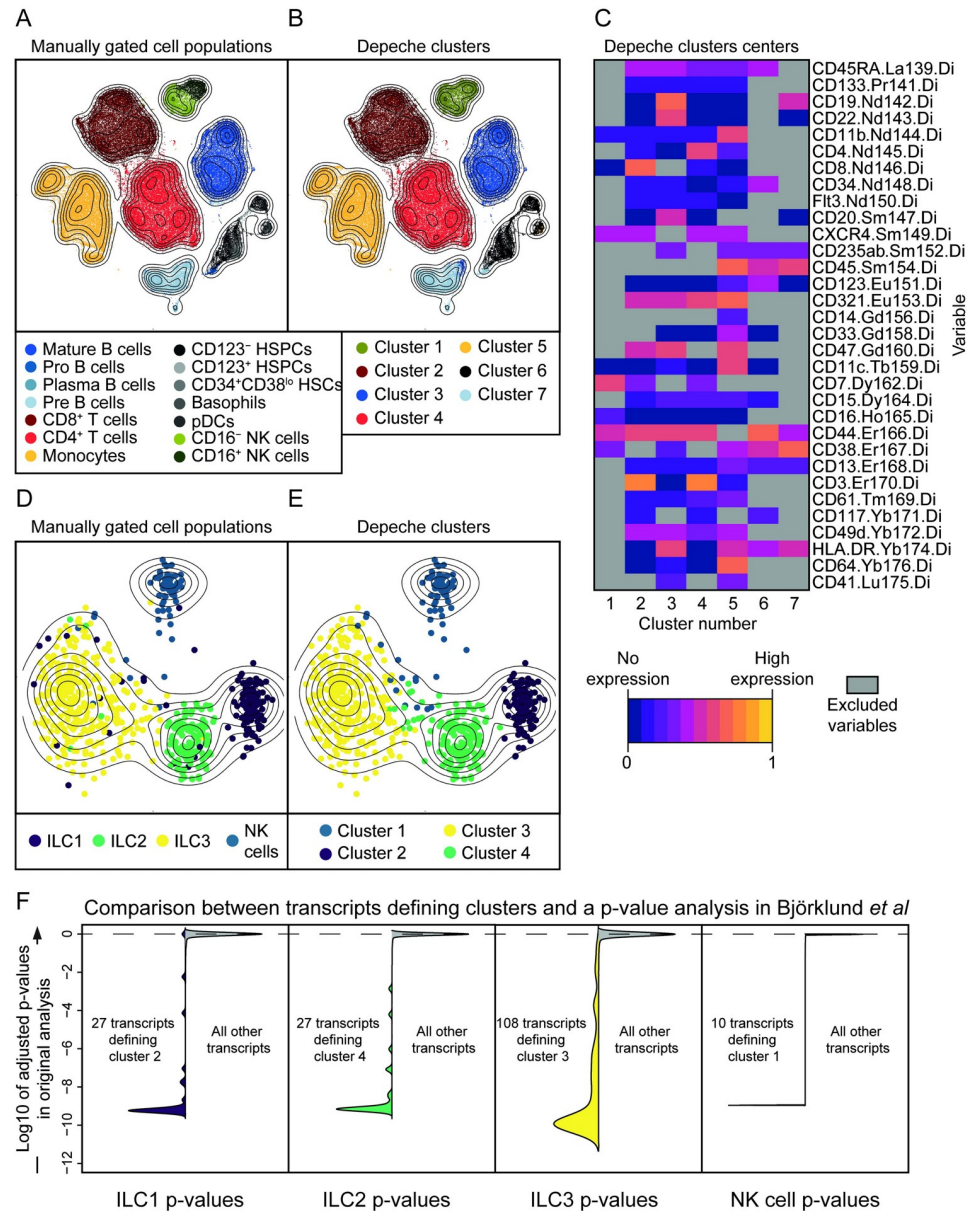
To evaluate how biologically accurate DEPECHE clustering is on mass cytometry data, a 32-variate mass cytometry bone marrow dataset [14] was clustered, and the overlap to 14 visually pre-defined cell populations was quantified using the ARI. With this dataset, DEPECHE identified 7 clusters at the optimal resolution, corresponding to all large pre-defined cell populations and to agglomerates of smaller cell populations, rendering an average ARI of 0.96, where an ARI of 1 corresponds to exact reproduction and an ARI of 0 means that the produced clusters are no more accurate than random allocation (Fig 2A and 2B, S2A Fig). Furthermore, using the DEPECHE algorithm, the number of variables defining each cluster was reduced from 32 to a range from 8 to 28, thereby enhancing interpretability (Fig 2C). When comparing to other state-of-the-art clustering algorithms [14–19], DEPECHE obtained similar ARI as the best algorithms for both the 32-variate dataset and another 14-variate, 24 population, mass cytometry dataset [20] (S3A Fig, Table 1).

Single-cell transcriptomic datasets feature tens of thousands of variables. Thus, compared to cytometry datasets, the need to exclude variables of low interest is even more pressing. We therefore evaluated DEPECHE’s ability to cluster and extract the key transcripts defining clusters of a previously published single-cell transcriptomic dataset (Fig 2D–2F) [21]. In this dataset, a total of 648 ILC1, ILC2, ILC3 and NK cells from three donors’ tonsils were index-sorted prior to RNA sequencing. Hence, these cell types, defined by expression of specific sets of markers using manual gating, can be compared to clusters unbiasedly determined by RNA



**Fig 1. Illustration of the DEPECHE workflow.** A) The original dataset  $D$ . B) Resampled datasets with  $N_r$  data points per dataset, are generated by sampling data points from  $D$  with resampling. Each resampled dataset has a corresponding penalty  $\lambda_i$  ( $i = 1 \dots N_\lambda$ ). C) Each dataset in b is clustered with sparse k-means, using its corresponding penalty  $\lambda_i$ . The red frame highlights the clustering with the strongest attractors, *i.e.* the most generalizable solution (see [Methods](#), “Simultaneous Clustering and Parameter Tuning”). D, E) Finally, the full data set is clustered. In cases where it is computationally feasible, the full data set is re-clustered with the optimal penalty (D). Otherwise, the final clustering is produced by allocating each data point to its closest cluster center, using the most generalizable cluster center solution produced with the optimal penalty in B (E).

<https://doi.org/10.1371/journal.pone.0203247.g001>



**Fig 2. DEPECHE performance with real datasets with 32 or 35177 variables.** A-B) bi-variate t-distributed stochastic neighbor embedding (tSNE) representation of the 32-variate mass cytometry data. A) distribution of manually defined cell populations over the tSNE field. B) distribution of DEPECHE clusters over the tSNE field. C) Heatmap of the expression pattern at each cluster center. Light colors show that the marker is highly expressed at the cluster center, whereas dark colors shows the opposite. Grey color indicates that the variable in question does not contribute to defining the cluster. For Fig a-c all, 104184 cells have been clustered. D-E) tSNE representation of the 137-variate data subset that could efficiently distinguish the clusters in the 35177-variate single-cell transcriptome dataset. D) distribution of the cell types defined by index-sorting and manual gating on protein expression profiles shown over the tSNE field. E) distribution of DEPECHE clusters over the tSNE field. F) Violin plots illustrating the overlap between the original analysis by Björklund *et al* and the DEPECHE analysis. For each subplot, the left and right side illustrate the distribution of the transcripts defining the clusters, and all other transcripts, respectively. The y-axis shows the log10 of the p-values in the original analysis adjusted for multiple comparisons. For Fig D-F, all 648 cells have been clustered.

<https://doi.org/10.1371/journal.pone.0203247.g002>

Table 1. Background information on all datasets.

Dataset	Data origin	n cells	n variables in analysis	n clusters in original
Levine[14]	Mass cytometry	104184	32	14
Bendall[20]	Mass cytometry	81747	14	24
Björklund[21]	scRNAseq	648	35177	4
Biase[5]	scRNAseq	56	19571	3
Deng[5]	scRNAseq	268	13867	10
Goolam[5]	scRNAseq	124	15487	5
Kolodziejczyk[5]	scRNAseq	704	15117	3
Pollen[5]	scRNAseq	301	13860	11
Yan[5]	scRNAseq	90	13608	7

<https://doi.org/10.1371/journal.pone.0203247.t001>

expression profiles [21]. In the DEPECHE analysis, no pre-selection of transcripts was performed, and hence, 35177 unique transcripts were included for each of the 648 cells. With the optimal penalty  $\lambda$ , four clusters were identified (Fig 2E). These corresponded well to the cell types as defined by protein expression; 84, 97, 91 and 97 percent of ILC1, ILC2, ILC3 and NK cells sorted into separate clusters, respectively (Fig 2D and 2E, S2B Fig), leading to an average ARI of 0.78, which was comparable to clusters obtained in the original publication [21]. Notably, cluster 1–4, corresponding to NK cells, ILC1, ILC3 and ILC2, were defined by 10, 27, 108 and 27 transcripts, respectively (Figs 2F and 3), leading to a 99.9% average decrease in the number of variables. The transcripts identified to define the clusters in our analysis corresponded to those identified as most differentially expressed in the original study [21] (Fig 2F). Thus, by identifying a finite number of variables, DEPECHE analysis can increase interpretability and aide down-stream analyses. When DEPECHE clustering was compared to that of state-of-the-art algorithms [5–7] on the aforementioned dataset as well as six other datasets (see Table 1), it consistently performed well as indicated by ARI (S3B Fig). Thus, when applied to megavariable single-cell transcriptomic data, DEPECHE identifies clusters corresponding to known cell types and reduces the complexity of the results thousand-fold by finding a subset of markers that define each cluster in a way that compares well to previous knowledge. Notably, as DEPECHE selects the most robust identifiers of each cluster, some potentially interesting transcripts might be excluded by DEPECHE clustering, *e.g.* transcripts that are highly expressed in a minority, but lacking from a majority of cells in a cluster.

In conclusion, DEPECHE turns the penalized k-means methodology into a parameter free analysis technique guided by efficient calculation of the optimal clustering resolution. By doing so, it can simultaneously address the problems of finding biologically relevant clusters and identifying specific variables that define these clusters. This is crucial in order to comprehend the noisy and often over-complicated data generated with current single cell technologies.

## Methods

### Clustering with DEPECHE

Clustering in DEPECHE is performed using a penalized version of the k-means algorithm, which is related to the k-means algorithm [10]. In this section, the k-means algorithm is outlined, followed by an explanation of how it is extended to penalized k-means.

The k-means algorithm clusters data by fitting a mixture of normal distributions to the data with  $k$  equal mixture components and unit variance. Formally,  $k$   $d$ -dimensional cluster centers, denoted  $\mu_{m,j}$  where  $m = 1 \dots k$  and  $j = 1 \dots d$ , are fitted to the  $n$   $d$ -dimensional datapoints  $x_{i,j}$ ,

Cluster 1/NK cells			Cluster 4 (ILC2)			Cluster 3/ILC3			Cluster 3/ILC3		
Transcript	Log10 of adjusted original p-value	Cluster center	Transcript	Log10 of adjusted original p-value	Cluster center	Transcript	Log10 of adjusted original p-value	Cluster center	Transcript	Log10 of adjusted original p-value	Cluster center
CMC1	-8.95	0.23	A2M	-7.08	0.23	CD3G	-9.66	-0.06	PCDH9	-9.66	1.25
CST7	-8.95	0.17	AC092580.4	-9.17	-0.89	CD63	-6.03	0.21	PDCD4	-3.37	-0.02
GNLY	-8.95	1.82	CD2	-9.17	-2.69	CNN2	-9.66	-0.61	PECAM1	-9.66	0.06
GZMA	-8.95	0.23	CD300LF	-9.17	-0.06	COTL1	-8.04	-0.25	PRR5	-9.66	0.38
GZMK	-8.95	0.09	CD3E	-4.21	-0.40	CPNE7	-6.13	0.25	PTPN22	-4.79	0.05
KLRC1	-8.95	0.57	EMP3	-2.86	0.31	CTSA	-5.65	0.25	RBPJ	-5.14	0.07
KLRD1	-8.95	1.89	FCER1G	-7.08	-0.59	DCAF11	-6.48	0.04	RHOC	-9.66	0.76
KLRF1	-8.95	1.29	GATA3	-8.33	0.34	DHR33	-3.72	0.02	RP11-264B17.3..1	-1.41	-0.18
NKG7	-8.95	1.93	GSN	-9.17	-0.40	DOCK5	-9.66	0.30	RP11-330A16.1	-9.47	0.01
PRF1	-8.95	0.14	HPGDS	-9.17	0.37	ELOVL6	-9.66	0.31	RP11-466H18.1	-0.35	-0.02
Cluster 2/ILC1			IL10RA	-8.84	0.03	EMP3	-7.26	-0.94	RP11-845M18.6	-9.66	0.44
Transcript			IL17RB	-9.17	0.14	ENPP1	-9.66	0.17	RPS8	-1.54	-0.06
Log10 of adjusted original p-value			IL23R	-9.17	-0.47	FAIM3	-3.79	-0.23	S100A6	-6.94	-0.16
Cluster center			IL2RB	-9.17	-0.64	FAM65B	-3.38	-0.47	S1PR1	-9.66	-0.08
AE000661.37			IL32	-9.10	1.04	FCER1G	-9.66	1.02	SELL	-8.49	-0.97
CCR7			KLRC1	-9.17	-0.51	FES	-9.66	0.11	SELPGL	-7.57	-0.37
CD27			KLRG1	-9.17	0.50	GIMAP4	-7.49	-0.38	SERINC5	-9.66	-0.13
CD3D			KRT1	-9.17	0.21	GIMAP7	-5.46	-0.39	SH2D1B	-9.66	0.90
CD3E			SH2D1B	-9.17	-0.64	GSN	-9.66	1.24	SLA	-5.90	0.16
CD3G			TESPA1	-6.02	0.01	HCST	-0.34	-0.11	SLC38A1	-1.48	-0.06
CD4			TMIGD2	-9.03	-0.11	HDAC9	-9.66	0.30	SLC4A10	-9.66	0.32
CD6			TRAC	-9.17	-0.54	HLA-DRB1	-3.19	0.34	SORL1	-8.48	-0.23
CNN2			TXK	-8.53	-1.21	IL10RA	-9.66	0.97	SPINK2	-9.66	0.67
COTL1			TYROBP	-9.17	-1.40	IL1R1	-9.66	0.42	SPRY1	-8.67	0.11
CTSW			VWA5A	-9.17	-0.32	IL23R	-9.66	1.67	STARD3NL	-9.66	0.23
FCER1G			XCL1	-9.17	-0.62	IL32	-9.66	-0.85	TC2N	-6.72	-0.66
KLRB1			XCL2	-9.17	-0.12	IL4I1	-9.66	1.50	TCIRG1	-7.11	0.10
LITAF			Cluster 3/ILC3			ISG20	-2.10	-0.01	TLE3	-9.66	0.06
LST1			Transcript			ITGB2	-6.95	-0.76	TMIGD2	-9.66	1.09
RNU2-6P			Log10 of adjusted original p-value			ITM2C	-9.66	0.41	TNFRSF18	-9.66	0.79
RP11-466H18.1			Cluster center			KIAA1324	-9.66	1.07	TNFRSF25	-3.37	0.30
SH2D1B			A2M			KIT	-9.66	1.30	TNFRSF4	-7.13	0.04
SIT1			AC092580.4			KLRG1	-9.66	-0.26	TNFSF11	-9.66	0.31
TC2N			ADAM10			KRT81	-9.66	1.00	TNFSF13B	-9.66	1.02
TNFRSF18			ADAM28			LAT2	-6.18	0.16	TOX	-9.66	0.35
TOB1			AFF3			LDHB	-9.35	-0.18	TOX2	-9.66	0.69
TRDC			AHR			LINC00299	-9.66	1.43	TRAC	-0.80	0.46
TRDJ2			AMICA1			LITAF	-9.66	-0.20	TRAJ45	-8.56	0.70
TYROBP			ARL4C			LST1	-9.66	1.68	TRAT1	-7.81	-0.12
U2..37			ATP8B4			LTA4H	-9.66	0.89	TRDJ2	-6.30	0.07
U2..55			BST2			LY6E	-3.30	-0.27	TRGJP1	-7.31	0.11
			C1orf162			MPG	-8.65	0.49	TXK	-5.59	0.14
			CAT			NCR2	-9.66	1.00	TYROBP	-9.66	1.15
			CD2			NKG7	-1.82	-0.04	VWA5A	-9.66	1.61
			CD300LF			NRP1	-9.66	0.35	XCL1	-9.66	0.80
			CD3D			NSMCE1	-9.66	0.64	XCL2	-9.66	0.03
						OTUD5	-9.66	0.73			

**Fig 3. Transcripts defining clusters in Björklund *et al* dataset.** Blue background color indicates 10 transcripts associated with cluster 1/NK cells. Violet background color indicates 27 transcripts associated with cluster 2/ILC1. Green background color indicates 27 transcripts associated with cluster 4/ILC2. Yellow color indicates 108 transcripts associated with cluster 3/ILC3.

<https://doi.org/10.1371/journal.pone.0203247.g003>

where  $l = 1 \dots n$ , by maximizing the score function

$$Q = \sum_{m=1}^k \sum_{l=1}^n z_{l,m} \sum_{j=1}^d (x_{l,j} - \mu_{m,j})^2, \tag{1}$$

where  $z_{m,l}$  is 1 if the  $l^{\text{th}}$  data point belongs to the  $m^{\text{th}}$  cluster and zero otherwise. The score  $Q$  is optimized using an Expectation Maximization (EM) algorithm [22], *i.e.* so called E- and M-steps are iterated alternately until the score  $Q$  stops improving. In the E-step, the allocation variables  $z_{m,l}$  are updated so that each data point is allocated to its closest cluster according to the Euclidean norm. That means that  $z_{m,l} = 1$  if the  $m^{\text{th}}$  cluster center is the cluster center closest to the  $l^{\text{th}}$  data point and  $z_{m,l} = 0$  otherwise [23]. In the M-step, each cluster center  $\mu_{m,j}$  is

moved to the center of the data points allocated to it. When no more reallocation occurs in the E-step, the algorithm has converged.

In order to reduce the influence of uninformative dimensions that only contribute with noise, penalized k-means introduces an L1-penalty for each element of each cluster center  $\mu_{m,j}$  to the optimization objective. Formally, the score function  $Q$  in Eq (1), is updated:

$$Q = \sum_{m=1}^k \sum_{l=1}^n z_{m,l} \sum_{j=1}^d (x_{l,j} - \mu_{m,j})^2 - \lambda \sum_{m=1}^k \sum_{j=1}^d |\mu_{m,j}|, \tag{2}$$

where  $\lambda$  is a positive penalty term. The additional term in the score function, introduced in Eq (2) results in a change in the M-step of the original EM-algorithm of the k-means algorithm. Keeping  $z_{m,l}$  for all  $l$  fixed and optimizing  $Q$  with respect to  $\mu_{m,j}$ , the M-step is:

$$\mu_{m,j} = \text{sign} \left( \frac{\sum_{l=1}^n z_{m,l} x_{l,j}}{\sum_{l=1}^n z_{m,l}} \right) \cdot \max \left( \left| \frac{\sum_{l=1}^n z_{m,l} x_{l,j}}{\sum_{l=1}^n z_{m,l}} \right| - \frac{\lambda}{2 \sum_{l=1}^n z_{m,l}}, 0 \right) \tag{3}$$

Depending on the choice of the penalty parameter  $\lambda$ , some components of some clusters centers will be set to 0 in the M-step. Note that penalized k-means with penalty  $\lambda = 0$  reduces to the original k-means algorithm.

In DEPECHE, cluster centers that are moved to the origin in the M-step are eliminated and not assigned any data points in the E-step. Due to the elimination of clusters, the number of produced clusters is independent of  $k$  and dependent on the penalty  $\lambda$  as long as at least one cluster is eliminated. In DEPECHE,  $k$  is always chosen to be so large that at least one cluster is eliminated. Eq (2) is a special case of the penalized model based clustering algorithm by Pan and Shen with unit variance and equal mixture components [11]. By imposing the penalty for each dimension and each cluster, penalized k-means identifies the dimensions that do not distinguish a particular cluster from the rest of the data, thus leaving these dimensions out of the definition of that cluster.

Penalized k-means, as well as k-means, relies on a procedure for initializing the positions of the cluster centers. Therefore, in DepecheR, the initial cluster positions are generated using the well performing seed generation algorithm of k-means++ by Arthur and Vassilvitskii [24]. The early iterations of the EM-algorithm are particularly delicate in DEPECHE, due to the elimination of clusters at the origin in the E-step. Poor initialization of the clusters, in combination with a high penalty  $\lambda_i$ , might lead to elimination of too many clusters in the early E-steps, yielding fewer clusters in the end result than necessary to optimize  $Q$ . To remedy this, DEPECHE starts each run of penalized k-means, regardless of the chosen  $\lambda_i$ , with penalty  $\lambda = 0$ . The penalty is then increased linearly over a number of E-steps until it reaches the predetermined value  $\lambda_i$ .

The EM-algorithm guarantees convergence to an optimum of the score  $Q$ , but not necessarily to the global optimum. In order to diminish the influence of the starting state, the EM-algorithm is run several times with random initialization, and the solution with optimal score  $Q$  is chosen. For penalty optimization, the number of runs is determined according to a set of rules that are outlined in the section “Tuning the penalty”. For producing the final results (Fig 1E), a fixed number of runs is performed (21 (3 x (8-1)) processor cores) runs is the default in DepecheR). To further decrease stochastic variability in the computational results,  $k$  is set considerably higher than the expected number of final clusters, which also diminishes the sensitivity to the starting state. In the extreme case where  $k$  is set equal to the number of data points  $n$ , the outcome of penalized k-means is deterministic.



## Tuning the penalty

In this section, we describe the optimization scheme which is used for tuning the linear penalty  $\lambda$ . The outline of the algorithm:

1. Choose a wide and exponentially distributed range of penalty terms  $\lambda_i, i = 1..N_\lambda$ , that are considered for clustering the dataset  $D$ .
2. Create 2 datasets per penalty term  $\lambda_i$ , called  $D_{1,i}$  and  $D_{2,i}$ , by sampling  $N_r$  data points from  $D$  with replacement.
3. Run the penalized k-means algorithm on the datasets  $D_{1,i}$  and  $D_{2,i}$ , yielding sets of cluster centers, denoted  $M_{1,i}$  and  $M_{2,i}$ .
4. Create the partitions  $P_{1,i}$  and  $P_{2,i}$  by allocating all data points of the dataset  $D$  to their nearest cluster center of the sets  $M_{1,i}$  and  $M_{2,i}$ . The allocation is equivalent to one E-step of the k-means algorithm.
5. Determine the Adjusted Rand Index (ARI), denoted  $r(\lambda_i)$  from  $P_{1,i}$  to  $P_{2,i}$  [25].
6. Repeat step 2–5 and average the obtained ARIs  $r(\lambda_i)$  penalty wise until a stopping criteria regarding the statistical certainty of the obtained ARIs  $r(\lambda_i)$  is met (remark 2, section “Tuning the penalty”).
7. Choose the optimal penalty  $\lambda_i$ , which is the penalty with the largest ARI  $r(\lambda_i)$ .

Some remarks:

1. To increase the likelihood of finding the optimal penalty value in the pre-defined range, datasets with very high kurtosis are log2-transformed and all datasets are divided by their total standard deviation prior to sampling. With this setup, all tested datasets have their optimal penalty in the default range. However, new datasets might have other requirements for penalties. In cases where the most extreme low or high penalty value is selected by the optimization procedure, DepecheR will warn the user, and suggest another range of penalties, or a larger sample size. The default penalty range in DepecheR is  $2^0, 2^{0.5} \dots 2^5$ .
2. The repetition of Step 6 is necessary, since the obtained ARI  $r(\lambda_i)$  is a random variable, due to the random procedure for creating the datasets  $D_{1,i}$  and  $D_{2,i}$  and the random procedure for initializing the penalized k-means algorithm. To determine the necessary number of runs, after an initial fixed number of runs (default 20 in DepecheR), DEPECHE uses three stopping criteria: The first criterion creates an interval of width 2 standard errors around the obtained mean of  $r(\lambda_i)$  and checks if the interval around the optimal ARI  $r(\lambda_i)$  has a zero overlap with the other intervals. The second criterion checks whether the standard error of the mean of  $r(\lambda_i)$  for the optimal penalty  $\lambda_i$  is below a threshold. The third criterion terminates the process after a maximal number of runs (default 100 in DepecheR).
3. Step 2 requires a samples size  $N_r$ . A natural choice is to set  $N_r$  equal to the number of data points,  $n$ . However, in cases where  $n$  is very large, so that the computational load of the optimization scheme becomes limiting, it is preferable to choose a smaller  $N_r$ . In DepecheR,  $N_r = 10^4$  by default, in case  $n \geq 10^4$ . Notice that when an optimal penalty  $\lambda_i$  is discovered using sample size  $N_r \neq n$ , the corresponding optimal penalty when sampling the full dataset  $D$  with magnitude  $n$  is (approximately)  $\lambda_i \cdot n/N_r$ , since the attraction force of a cluster is proportional to the number of data points in it.
4. In step 5, the ARI is chosen as similarity measure between the partitions, since it corrects for chance. This means that it gives a zero similarity for trivial partitions, such as having all

data points in a single cluster. Exact calculation of the ARI is computationally intractable for large datasets. Therefore, DEPECHE relies on an approximate ARI computation, based on  $10^4$  random pairs of data points.

### Simultaneous clustering and parameter tuning

For very large datasets ( $n > 10^8$ ), not only the penalty optimization, but also the final clustering once the optimal penalty has been found, may be computationally intractable. However, increasing the size of the dataset does not necessarily lead to an increase in number of clusters at the optimal resolution. In this case, it is feasible to cluster a subset of the full dataset  $D$  to obtain cluster centers  $M$  and then allocate the remaining data points of  $D$  to their closest clusters in  $M$ . This boosts computational efficiency since allocation imposes a much smaller computational load than clustering. Since several subsets of  $D$  are produced and clustered during the tuning of the penalty parameter  $\lambda$ , it is computationally favorable to retrieve cluster centers  $M$  that were produced during the parameter tuning and use them to cluster  $D$ .

When picking a set of cluster centers  $M$  from the penalty tuning, the question arises which set of centers  $M$  to take, since several sets of centers, denoted  $M_{i,p}$  ( $p = 1 \dots N_p$ ), are produced for the optimal penalty  $\lambda_i$ . In DEPECHE, the centers  $M_{i,p}$  that have the strongest similarity (on average) to the remaining  $N_p - 1$  centers is chosen and is referred to as the most generalizable cluster set. The level of similarity between the centers  $M_{i,p}$  and  $M_{i,p'}$  is quantified using the ARI between the partitions  $P_{i,p}$  and  $P_{i,p'}$ , induced by allocating each data point of  $D$  to its closest cluster center in  $M_{i,p}$  and  $M_{i,p'}$  respectively.

**Empiric performance of the penalty tuning scheme.** Roughly speaking, DEPECHE combines a flavored penalized k-means algorithm with a parameter tuning scheme, which identifies an optimal resolution. A naturally arising question is then whether the parameter tuning scheme is able to determine a biologically relevant resolution or if other penalized k-means clustering resolutions outperform the resolution chosen by DEPECHE. Using a range of datasets (Table 2), the biological relevance (measured in ARI to the manually curated solution) of the optimized DEPECHE partitions were compared to the biologically optimal partition among all partitions generated with 20 repetitions on each of a range of 11 penalties per dataset. Overall, the DEPECHE resolution-selection showed close to optimal performance, as the selected solutions only had a median of 0.02 lower ARI to the gold standard (range 0–0.065) than the best possible solution with all penalties (Table 2).

Table 2. ARI between DEPECHE partitions and golden standard partitions.

Dataset	Median ARI in S2 Fig	Maximal ARI with any penalty	Difference
Levine[14]	0.961	0.975	0.015
Bendall[20]	0.841	0.873	0.032
Björklund[21]	1	1	0
Biase[5]	0.782	0.842	0.06
Deng[5]	0.827	0.848	0.021
Goolam[5]	0.629	0.639	0.009
Kolodziejczyk[5]	0.992	1	0.008
Pollen[5]	0.863	0.928	0.065
Yan[5]	0.626	0.691	0.064

<https://doi.org/10.1371/journal.pone.0203247.t002>

## Transforming and centering the data

The clusters produced by DEPECHE, as well as their interpretation, depends on the transformation and centering of the data. The transformation determines the relative importance of the measured variables, where variables with a larger spread have stronger influence on the clustering. The centering defines where zero occurs in each variable, thereby influencing the clustering results due to the linear penalty.

DEPECHE is applicable to a large range of datasets where the numbers of dimensions,  $d$ , and the number of data points,  $n$ , can vary with many orders of magnitude. The differing characteristics of these datasets require different treatments with respect to transformation and centering.

**Transformation.** Empirically, a majority of single-cell transcriptome datasets tend to have a few variables where the variance is many orders of magnitude greater than in the other variables. In this case, the high-variance variables will *de-facto* determine the clustering, implying that the clustering will fail to take the majority of the measured information into account. To even out the influence of these high-variance variables on the clustering outcome, the data is log transformed when such variables are present. In DepecheR, this data behavior is detected automatically by concatenating all variables into a one-dimensional vector, for which the kurtosis is calculated. A high kurtosis indicates that the variables differ greatly in their variance. For datasets with low kurtosis, refraining from the log transform is preferable to avoid unnecessary data distortion.

**Centering.** Centering the origin to be close to the bulk of the data is preferable, in order to have all biological clusters at approximately the same distance from the origin. Having some biological clusters close to the origin and some far off is often unwanted, since the linear penalty then imposes a preference for creating clusters close to the origin. Apart from influencing the clustering, the centering also determines the interpretation of the obtained sparsity. Just as for scaling, which centering scheme to apply depends on the dataset.

For low dimensional datasets ( $d > 100$ ), DEPECHE applies maximal density centering, which sets the zero in each dimension to coincide with the highest data density. The density is computed by collecting the data in equally spaced bins (default number of bins in DepecheR is the number of data points  $n$  divided by 50), where the bin with the highest number of data points has the highest density. Using this scheme, sparsity (*i.e.* that a variable is non-contributing to the definition of a cluster) is interpreted as non-deviance from the most common outcome. It also ensures that the origin is relatively close to the bulk of the data, since it is located at the most common outcome for each variable respectively. The benefit of this scheme is that it boosts sparsity, by declaring the most common outcome non-defining. However, for high dimensional datasets ( $d \geq 100$ ), maximal density centering can push the origin so far away from the center of mass of the dataset, that the penalty starts to impose an unwanted, artificial influence on the clustering, hampering the biological relevance of the clusters. To avoid this, DEPECHE imposes a mean centering scheme for such datasets, which locates the origin at the center of mass of the dataset.

A potential complication, related to centering, occurs when a biologically relevant cluster is located very close to the origin, since DEPECHE creates no clusters in the origin and will then force the cluster to merge with other clusters. However, this scenario was never detected in real data.

## Experimental procedures

### Generation and analysis of synthetic example for k-means algorithm comparisons.

The synthetic dataset was generated using base package functions. The standard deviation was

identical for all clusters and dimensions. For classical k-means analysis, the k-means function in the stats package was used. For sparse k-means, the sparcl package was used [26].

**Preprocessing of mass cytometry data.** The benchmark datasets from Levine *et al* [14] and Bendall *et al* [20] were transformed using the flowTrans package [27] before used in any clustering algorithm.

**Preprocessing of single-cell transcriptomic data.** The dataset from Björklund *et al* [21] was normalized using the sva package [28] as in the original manuscript. For this dataset, non-expressed transcripts were removed, lowering the number of variables from 64443 to 35177.

The gold-standard datasets used for benchmarking in the publication by Kiselev *et al* [5] were obtained in a pre-processed state. Before clustering with any algorithm, the gene filter function used in the sc3 package was used [5], with settings removing the genes that were expressed in more than 90% of the cells. This resulted in the number of transcripts presented in Table 1 (range 13608–19571 transcripts).

## Code availability

All code necessary to generate the figures and tables in the manuscript are included in S1 File. The software package DepecheR is available for download at (<https://github.com/Theorell/DepecheR>).

## Supporting information

**S1 Fig. A comparison of classical K-means, sparse K-means and DEPECHE.** A: Overlap between true clusters and clusters generated with classical k-means, sparse K-means, or DEPECHE. Numbers in heatmaps denote percent of the true cluster present in the generated cluster in question. Red color indicates high overlap, blue color low overlap. B: Total loadings for all input vectors with sparse k-means. Light color indicates a strong contribution to separation of the clusters and vice versa. Grey color indicates that the variable has been excluded. C: Cluster center matrix for DEPECHE analysis. Rows indicate the DEPECHE clusters, columns indicate the variables that contribute to separating at least one cluster. A light color indicates that the cluster center is located in the upper part of the distribution of values in the vector in question, and vice versa. Grey color indicates that the variable has been excluded. For DEPECHE, only 10 dimensions are shown, as the 11<sup>th</sup> did not contribute to separating any cluster.

(TIF)

**S2 Fig. Heatmaps comparing the golden standard partitions to the DEPECHE partitions for A) the 32-variate Levine dataset and B) the 35166-variate Björklund dataset.** Red color indicates large overlap, blue color indicates low overlap between a gold standard-vs-DEPECHE cluster pair. Numbers in heatmaps denote percent of the golden standard cluster present in the DEPECHE cluster in question.

(TIF)

**S3 Fig. Algorithm comparisons.** For all graphs, the x-axis shows the algorithms and the y-axis shows the Adjusted Rand Index comparing the clustering result with the golden standard clustering. Below each graph is the average computational time in seconds for the benchmarking performed on a laptop computer with 4 2.8 GHz Intel Core i7 processors. A) Subsamples with 20000 unique cells from two mass cytometry datasets published by Levine *et al* and Bendall *et al* were clustered with DEPECHE and six previously published algorithms. For each dataset and algorithm, clustering was performed on 20 unique subsamples. For flowClust, flowPeaks and SamSPECTRAL, that do not perform internal parameter tuning, a range of parameter

values were evaluated and the parameter value sets generating the highest ARI values were selected for display. B) The full Björklund dataset, as well as six other datasets previously used for benchmarking by Kiselev *et al* were clustered 20 times with DEPECHE and three other algorithms. The Björklund dataset was normalized to reduce batch effects, with the procedure described in the original publication. These six datasets were also automatically log<sub>2</sub>-transformed within DEPECHE, and thus, log<sub>2</sub>-transformation was applied also for Sincera and pcaReduce, whereas sc3 was fed both log<sub>2</sub>- and untransformed data. The lower and upper hinges of all boxplots extend to the 25:th and 75:th percentile, whereas the line in the middle describes the median. The whiskers extend to the lowest and highest value no further than 1.5 times the distance between the 25:th and 75:th percentile. Outside of this range, the observations are considered outliers and are shown as dots.

(TIF)

**S1 File. The code used generate all figures.**

(ZIP)

**S2 File. Information on how to retrieve the data used for this study.**

(PDF)

## Acknowledgments

The authors are grateful for all important input that has come from the initial users of the DepecheR software, especially Dr Niklas Björkström, Dr Jakob Michaëlsson and Sigrun Stultz. Other colleagues that have contributed seminally to the framework of ideas that have led to the creation of DEPECHE are Dr Bruce Bagwell, Dr Ryan Ramanujam and Dr Geoffrey Hart.

## Author Contributions

**Conceptualization:** Axel Theorell, Jakob Theorell.

**Data curation:** Axel Theorell, Jakob Theorell.

**Formal analysis:** Axel Theorell, Jakob Theorell.

**Investigation:** Axel Theorell.

**Methodology:** Axel Theorell, Jakob Theorell.

**Software:** Axel Theorell, Jakob Theorell.

**Supervision:** Yenan Troi Bryceson, Jakob Theorell.

**Validation:** Axel Theorell, Jakob Theorell.

**Visualization:** Jakob Theorell.

**Writing – original draft:** Jakob Theorell.

**Writing – review & editing:** Axel Theorell, Yenan Troi Bryceson, Jakob Theorell.

## References

1. Spitzer MH, Nolan GP. Mass Cytometry: Single Cells, Many Features. *Cell*. 2016; 165: 780–791. <https://doi.org/10.1016/j.cell.2016.04.019> PMID: 27153492
2. Tanay A, Regev A. Scaling single-cell genomics from phenomenology to mechanism. *Nature*. 2017; 541: 331–338. <https://doi.org/10.1038/nature21350> PMID: 28102262
3. Budnik B, Levy E, Slavov N. Mass-spectrometry of single mammalian cells quantifies proteome heterogeneity during cell differentiation. *bioRxiv*. 2017; 102681. <https://doi.org/10.1101/102681>

4. Saeys Y, Gassen SV, Lambrecht BN. Computational flow cytometry: helping to make sense of high-dimensional immunology data. *Nat Rev Immunol*. 2016; 16: 449–462. <https://doi.org/10.1038/nri.2016.56> PMID: 27320317
5. Kiselev VY, Kirschner K, Schaub MT, Andrews T, Yiu A, Chandra T, et al. SC3: consensus clustering of single-cell RNA-seq data. *Nat Methods*. 2017; 14: 483–486. <https://doi.org/10.1038/nmeth.4236> PMID: 28346451
6. Guo M, Wang H, Potter SS, Whitsett JA, Xu Y. SINCERA: A Pipeline for Single-Cell RNA-Seq Profiling Analysis. *PLoS Comput Biol*. 2015; 11: e1004575. <https://doi.org/10.1371/journal.pcbi.1004575> PMID: 26600239
7. Žurauskienė J, Yau C. pcaReduce: hierarchical clustering of single cell transcriptional profiles. *BMC Bioinformatics*. 2016; 17: 140. <https://doi.org/10.1186/s12859-016-0984-y> PMID: 27005807
8. Bashashati A, Brinkman RR. A Survey of Flow Cytometry Data Analysis Methods. *Adv Bioinforma*. 2009; 2009. <https://doi.org/10.1155/2009/584603> PMID: 20049163
9. R Core Team. R: A Language and Environment for Statistical Computing [Internet]. Vienna, Austria: R Foundation for Statistical Computing; 2017. <https://www.R-project.org>
10. MacQueen J. Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*. Berkeley, California: University of California Press; 1967. pp. 281–297. <https://projecteuclid.org/euclid.bsm/1200512992>
11. Pan W. Penalized model-based clustering with application to variable selection. *J Mach Learn Res*. 2007; 8: 1145–1164.
12. Witten DM, Tibshirani R. A framework for feature selection in clustering. *J Am Stat Assoc*. 2010; 105: 713–726. <https://doi.org/10.1198/jasa.2010.tm09415> PMID: 20811510
13. Sun W, Wang J, Fang Y. Regularized k-means clustering of high-dimensional data and its asymptotic consistency. *Electron J Stat*. 2012; 6: 148–167. <https://doi.org/10.1214/12-EJS668>
14. Levine JH, Simonds EF, Bendall SC, Davis KL, Amir ED, Tadmor MD, et al. Data-Driven Phenotypic Dissection of AML Reveals Progenitor-like Cells that Correlate with Prognosis. *Cell*. 2015; 162: 184–197. <https://doi.org/10.1016/j.cell.2015.05.047> PMID: 26095251
15. Lo K, Hahne F, Brinkman RR, Gottardo R. flowClust: a Bioconductor package for automated gating of flow cytometry data. *BMC Bioinformatics*. 2009; 10: 145. <https://doi.org/10.1186/1471-2105-10-145> PMID: 19442304
16. Aghaeepour N. flowMeans: Non-parametric Flow Cytometry Data Gating. 2010.
17. Sørensen T, Baumgart S, Durek P, Grützkau A, Häupl T. immunoClust—An automated analysis pipeline for the identification of immunophenotypic signatures in high-dimensional cytometric datasets. *Cytom Part J Int Soc Anal Cytol*. 2015; 87: 603–615. <https://doi.org/10.1002/cyto.a.22626> PMID: 25850678
18. Ge Y, Sealfon SC. flowPeaks: a fast unsupervised clustering for flow cytometry data via K-means and density peak finding. *Bioinforma Oxf Engl*. 2012; 28: 2052–2058. <https://doi.org/10.1093/bioinformatics/bts300> PMID: 22595209
19. Zare H, Shooshtari P, Gupta A, Brinkman RR. Data reduction for spectral clustering to analyze high throughput flow cytometry data. *BMC Bioinformatics*. 2010; 11: 403. <https://doi.org/10.1186/1471-2105-11-403> PMID: 20667133
20. Bendall SC, Simonds EF, Qiu P, Amir ED, Krutzik PO, Finck R, et al. Single-Cell Mass Cytometry of Differential Immune and Drug Responses Across a Human Hematopoietic Continuum. *Science*. 2011; 332: 687–696. <https://doi.org/10.1126/science.1198704> PMID: 21551058
21. Björklund ÅK, Forkel M, Picelli S, Konya V, Theorell J, Friberg D, et al. The heterogeneity of human CD127(+) innate lymphoid cells revealed by single-cell RNA sequencing. *Nat Immunol*. 2016; 17: 451–460. <https://doi.org/10.1038/ni.3368> PMID: 26878113
22. Dempster AP, Laird NM, Rubin DB. Maximum Likelihood from Incomplete Data via the EM Algorithm. *J R Stat Soc Ser B Methodol*. 1977; 39: 1–38.
23. Bishop C. *Pattern Recognition and Machine Learning* [Internet]. New York: Springer-Verlag; 2006. [www.springer.com/us/book/9780387310732](http://www.springer.com/us/book/9780387310732)
24. Arthur D, Vassilvitskii S. K-means++: The Advantages of Careful Seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics; 2007. pp. 1027–1035. <http://dl.acm.org/citation.cfm?id=1283383.1283494>
25. Hubert L, Arabie P. Comparing partitions. *J Classif*. 1985; 2: 193–218. <https://doi.org/10.1007/BF01908075>

26. Witten DM, Tibshirani R. sparcl: Perform Sparse Hierarchical Clustering and Sparse K-Means Clustering [Internet]. 2018. <https://CRAN.R-project.org/package=sparcl>
27. Finak G, Manuel-Perez J, Gottardo R. flowTrans: Parameter Optimization for Flow Cytometry Data Transformation. 2010.
28. Leek JT, Johnson WE, Parker HS, Jaffe AE, Storey JD. The sva package for removing batch effects and other unwanted variation in high-throughput experiments. *Bioinforma Oxf Engl*. 2012; 28: 882–883. <https://doi.org/10.1093/bioinformatics/bts034> PMID: 22257669