Methods for scalable, performant analysis

Davide Risso and Aedín Culhane

Scalable method to cluster millions of single cells

Fast: we want to be able to quickly cluster (multiple times) thousands to millions of cells in PCA space (data may fit in memory)

On-disk: we may need to quickly cluster full data matrices (millions of cells by thousands of genes) which do not fit in memory.

In some cases (e.g., normalization) speed is more important than accuracy

How much of a problem is it?

RNA SEQUENCING

2.5 Millions cells

Slide-seq: A scalable technology for measuring genome-wide expression at high spatial resolution

Samuel G. Rodriques[1,2,3]*, Robert R. Stickels[3,4,5]*, Aleksandrina Goeva[3], Carly A. Martin[3], Evan Murray[3], Charles R. Vanderburg[3], Joshua Welch[3], Linlin M. Chen[3], Fei Chen[3]†‡, Evan Z. Macosko[3,6]†‡

Article | Published: 20 February 2019

2 Millions cells

The single-cell transcriptional landscape of mammalian organogenesis

Junyue Cao, Malte Spielmann, Xiaojie Qiu, Xingfan Huang, Daniel M. Ibrahim, Andrew J. Hill, Fan Zhang, Stefan Mundlos, Lena Christiansen, Frank J. Steemers, Cole Trapnell & Jay Shendure

2.2 Millions cells

High-definition spatial transcriptomics for in situ tissue profiling

Sanja Vickovic[1,2*], Gökcen Eraslan[1,12], Fredrik Salmén[2,12], Johanna Klughammer[1,12], Linnea Stenbeck[2,12], Denis Schapiro[1,3], Tarmo Äijö[4], Richard Bonneau[5,6], Ludvig Bergenstråhle[2], José Fernandéz Navarro[2], Joshua Gould[1], Gabriel K. Griffin[1,6], Åke Borg[7], Mostafa Ronaghi[8], Jonas Frisén[9], Joakim Lundeberg[2,10*], Aviv Regev[1,11] and Patrik L. Ståhl[2]

NEUROGENOMICS

1 Million cells

Molecular, spatial, and functional single-cell profiling of the hypothalamic preoptic region

Jeffrey R. Moffitt*, Dhananjay Bambah-Mukku*, Stephen W. Eichhorn†, Eric Vaughn†, Karthik Shekhar, Julio D. Perez, Nimrod D. Rubinstein, Junjie Hao, Aviv Regev, Catherine Dulac‡§, Xiaowei Zhuang‡§

Source: www.nxn.se/single-cell-studies/gui

k-means clustering

Given a set of *n* data points (**x**) and a number *k*, k-means partitions the data in *k* clusters.

More formally, k-means clustering aims at minimizing the within-cluster sum of squares:

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \ :$$

In practice, we use an iterative algorithm based on two steps:

1. **Assignment**: given a set of centroids, assign each observation to the closest centroid.
2. **Update**: compute new centroids for each cluster.

Mini-batch k-means clustering (Sculley, 2010)

At each iteration, use small random subsets of the data ("mini-batches")

- No need to store the whole dataset in memory.
- At each iteration, only the distances between a mini-batch and the *k* centroids need to be computed.
- At each iteration, one only needs to have a subset of the data (mini-batch) and the k centroids in memory.
- This makes it a natural candidate for clustering on-disk data.

https://www.eecs.tufts.edu/~dsculley/papers/fastkmeans.pdf

Our implementation: the ***mbkmeans*** package

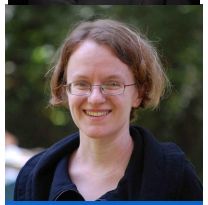
New Results

Comment on this paper

mbkmeans: fast clustering for single cell data using mini-batch *k*-means

Stephanie C. Hicks, Ruoxi Liu, Yuwei Ni, Elizabeth Purdom, Davide Risso

doi: https://doi.org/10.1101/2020.05.27.119438



mbkmeans

platforms all | rank 961 / 1905 | posts 0 | in Bioc 1 year

build ok | updated before release | dependencies 102

DOI: 10.18129/B9.bioc.mbkmeans

Mini-batch K-means Clustering for Single-Cell RNA-seq

Why (mini-batch) k-means?

- One of the most popular clustering methods.
- Building block of two Bioconductor packages for the clustering of single-cell RNA-seq, *clusterExperiment* and *SC3.*
- We envision scalable versions of these packages that leverage our implementation.

What is HDF5?

HDF5 is a unique technology suite that makes possible the management of extremely large and complex data collections.

- A versatile data model
- A completely portable file format
- A software library: high-level APIs with interfaces in C, C++, python, R, …

http://portal.hdfgroup.org/display/support

The HDF Group

Why HDF5?

***De facto* standard for single-cell RNA-seq data**

- 10X Genomics *Cell Ranger* software stores pre-processed data as a HDF5 file
- Scanpy's data format, **anndata**, is based on HDF5
- The **loompy** data format is based on HDF5

What is a DelayedArray?

- A convenient way to deal with HDF5 files in R/Bioconductor is via the **DelayedArray** framework.
- Data are stored on-disk in a HDF5 file.
- Operations are **delayed** and only performed on the subset of the data for which it is needed.

```r
library(TENxBrainData)
tenx <- TENxBrainData()
tenx
```

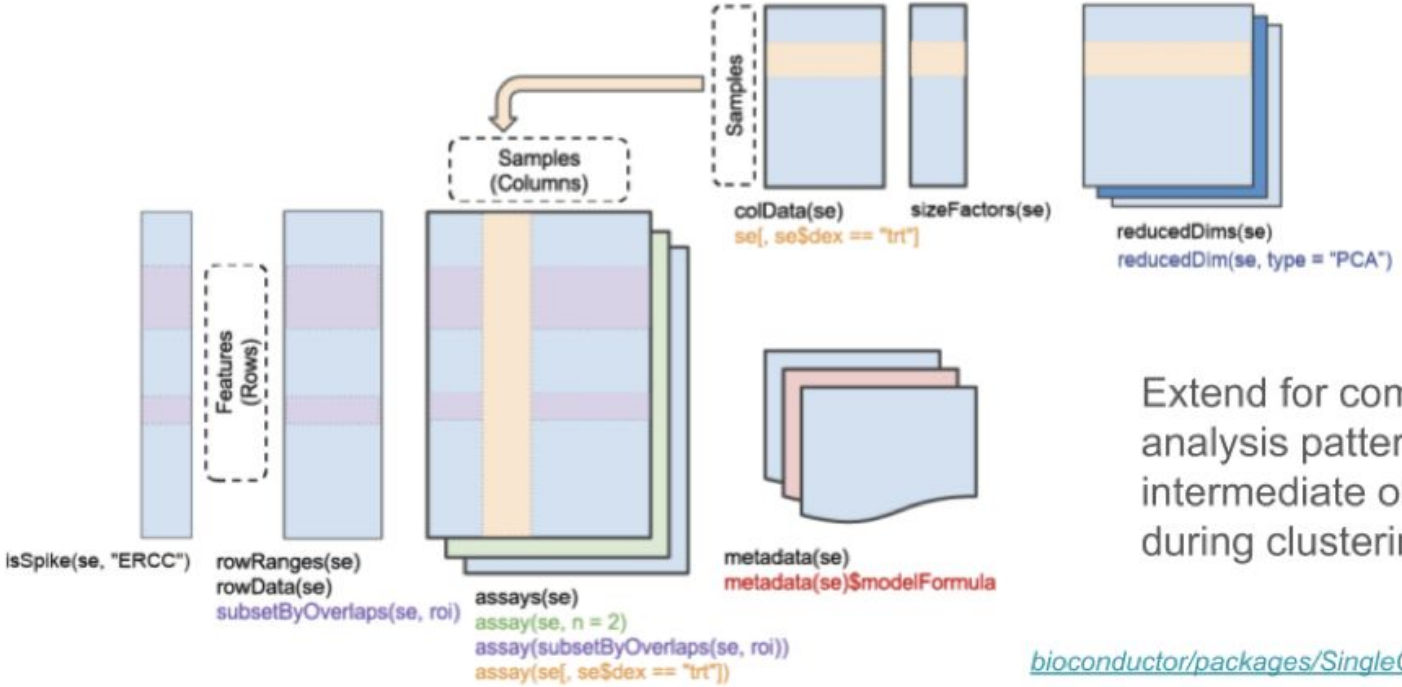
```
## class: SingleCellExperiment
## dim: 27998 1306127
## metadata(0):
## assays(1): counts
## rownames: NULL
## rowData names(2): Ensembl Symbol
## colnames(1306127): AAACCTGAGATAGGAG-1 AAACCTGAGCGGCTTC-1 ...
##   TTTGTCAGTTAAAGTG-133 TTTGTCATCTGAAAGA-133
## colData names(4): Barcode Sequence Library Mouse
## reducedDimNames(0):
## spikeNames(0):
## altExpNames(0):
```

```r
object_size(tenx)
```

```
## 200 MB
```

How do I use them in practice?

Common data structures for single-cell data



Extend for common analysis patterns, e.g., intermediate objects during clustering.

bioconductor/packages/SingleCellExperiment

Subsampling analysis

- 1.3M Brain cells from 10X Genomics
- 5,000 most variable genes
- Subsampled to: 75k, 150k, 300k, 500k, 750k, 1M cells
- iMac with 64GB RAM

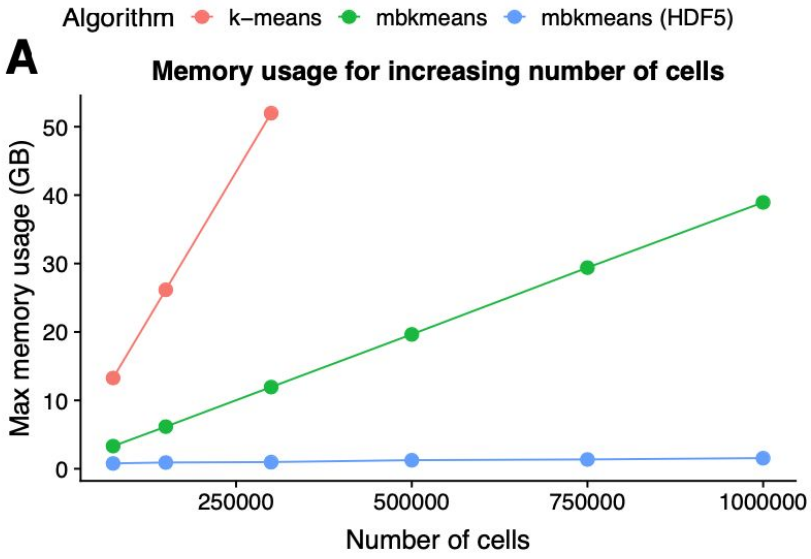
```r
library(TENxBrainData)
tenx <- TENxBrainData()
tenx
```

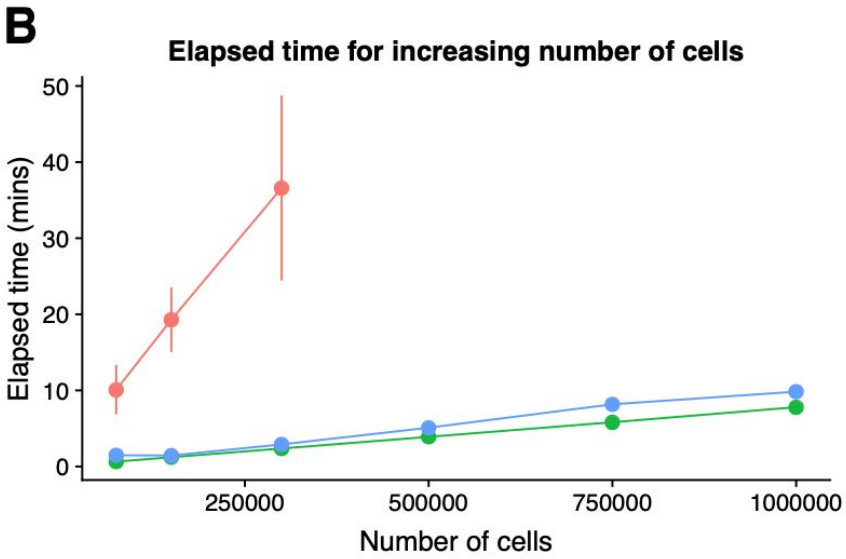
```
## class: SingleCellExperiment
## dim: 27998 1306127
## metadata(0):
## assays(1): counts
## rownames: NULL
## rowData names(2): Ensembl Symbol
## colnames(1306127): AAACCTGAGATAGGAG-1 AAACCTGAGCGGCTTC-1 ...
##   TTTGTCAGTTAAAGTG-133 TTTGTCATCTGAAAGA-133
## colData names(4): Barcode Sequence Library Mouse
## reducedDimNames(0):
## spikeNames(0):
## altExpNames(0):
```

```r
object_size(tenx)
```

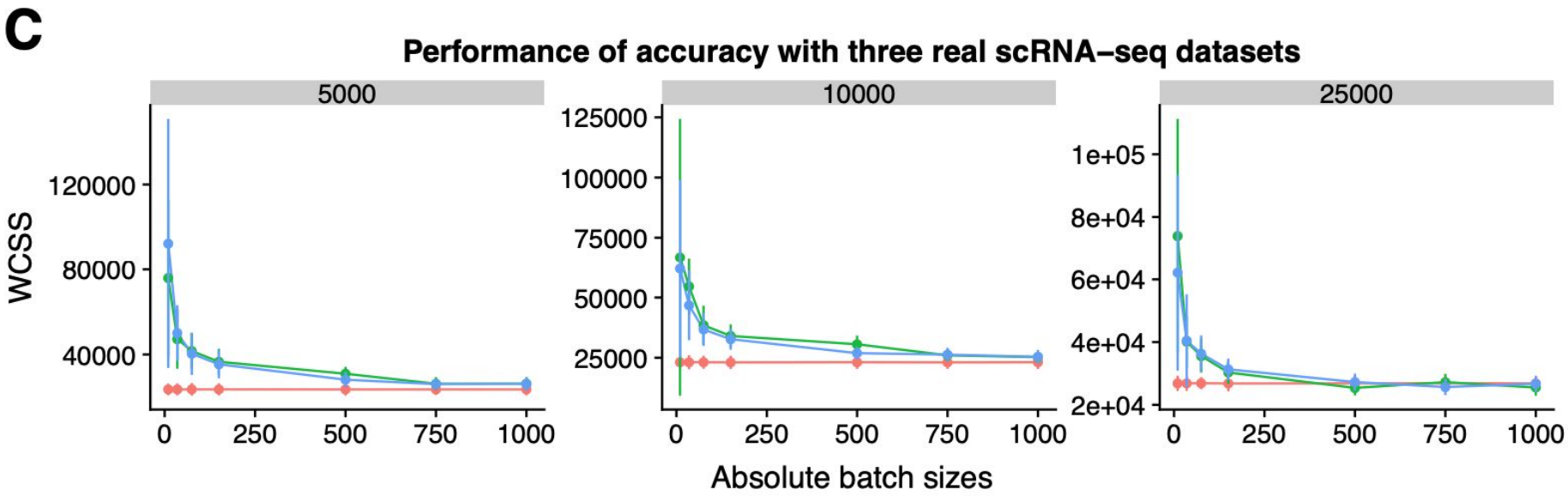
```
## 200 MB
```

Scalability





Accuracy


c
Performance of accuracy with three real scRNA–seq datasets
5000
10000
25000
WCSS
120000
80000
40000
125000
100000
75000
50000
25000
1e+05
8e+04
6e+04
4e+04
2e+04
0
250
500
750
1000
Absolute batch sizes

A complete analysis of 1.3M Cells

- Remove low-quality cells with ***scater***
- Keep only genes with at least one UMI in 1% of the cells
- ***scran*** normalization*
- First 50 Principal Components (using 1,000 most variable genes) using ***BiocSingular****'s irlba* PCA
- Mini-batch k-means clustering with ***mbkmeans*** (batch size of 500, k = 15).

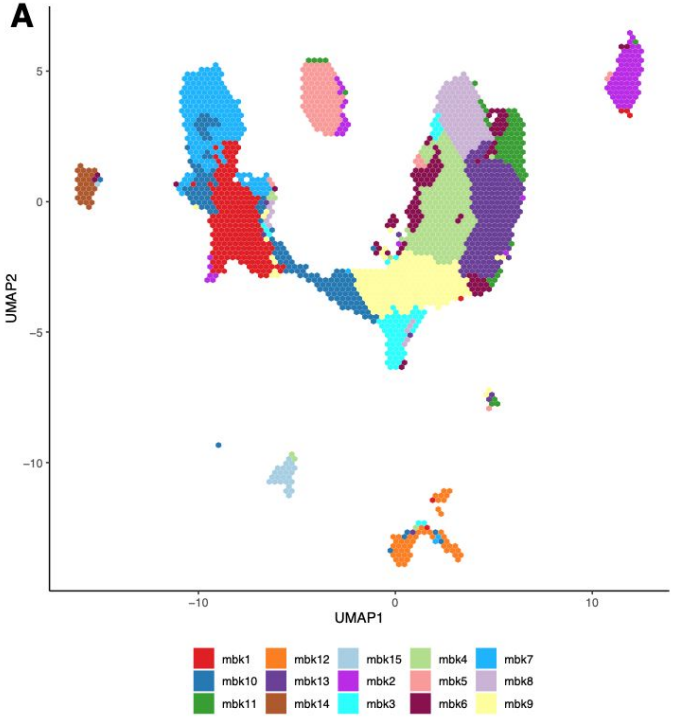
**mbkmeans* was used as a preliminary step for *scran* normalization

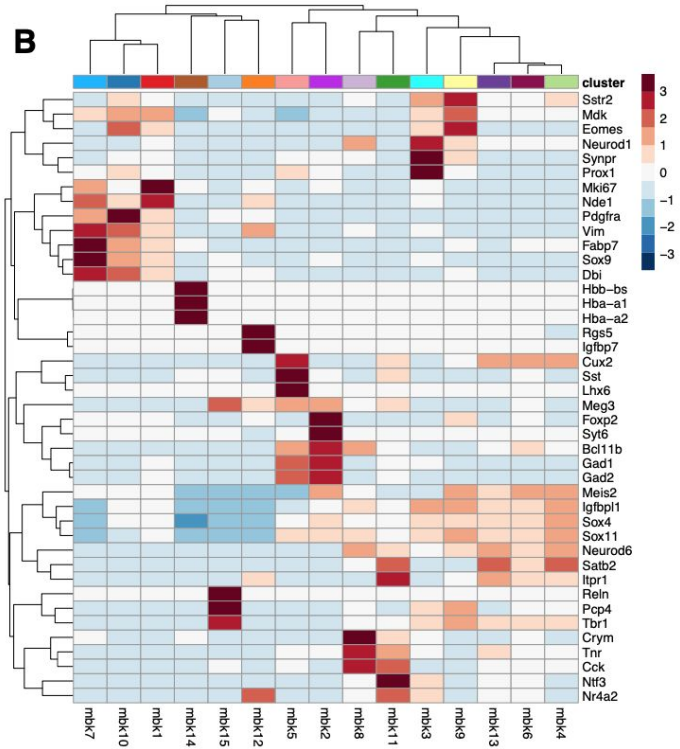
Compute time

- **104 hours** for complete workflow (starting from UMI counts).
- **8.5 mins for the clustering of the full 1,232,055 x 11,720 matrix** (*HDF5*; useful for normalization).
- 5 hours for normalization (parallel; 6 cores).
- 96 hours for irlba PCA (parallel; 6 cores).
- **3 mins for the clustering of the 1,232,055 x 50** matrix of the top 50 Principal Components (in memory).
- 2.5 hours for visualization (t-sne) or 20 mins (umap).

(Note that some of these steps may be further optimized)

A complete analysis of 1.3M Cells



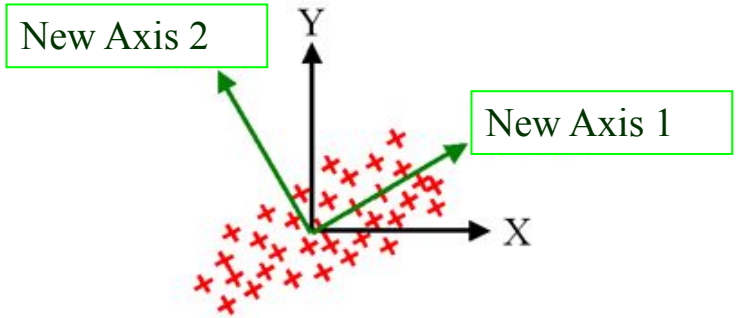


Dimensionality reduction is a key step

- **96 hours for irlba PCA (parallel; 6 cores)**
  - Note that random PCA may be faster
  - Note that a different HDF5 geometry may be much faster

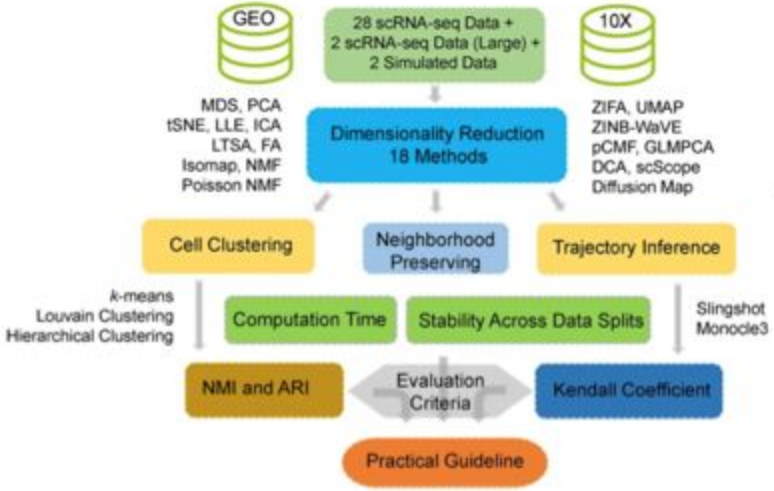
- **PCA may not be the best choice, methods designed for count data are preferable.**

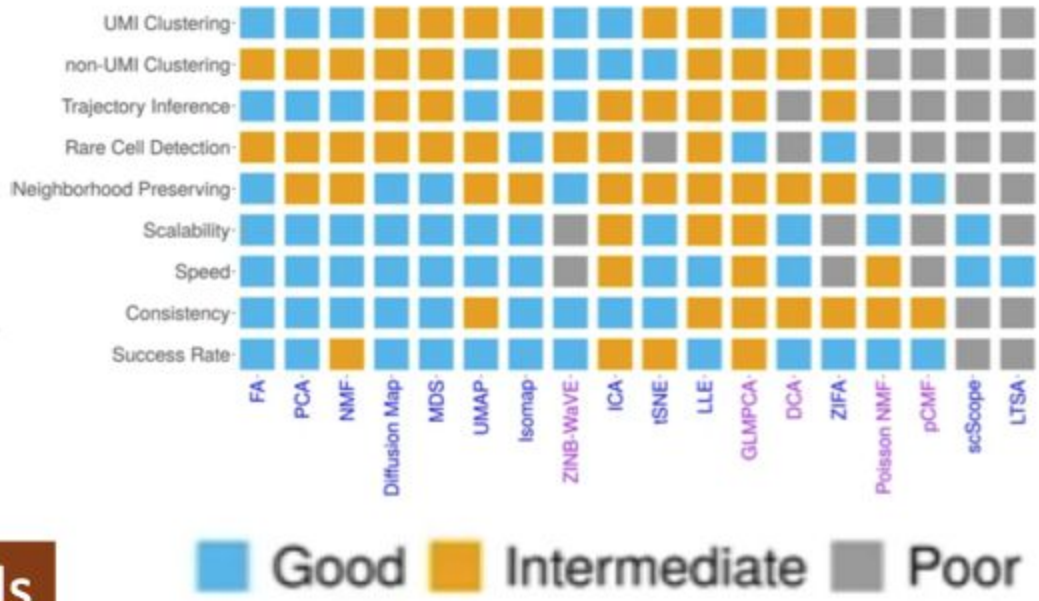
Matrix Factorization or dimension reduction methods, including PCA are typical first step in almost all single cell ‘omics analysis



Reduce a data matrix to a small number of linear vectors that explain most of the variance in the data

Assessment of the Accuracy, robustness and scalability of dimensionality reduction methods for single-cell RNA-seq analysis



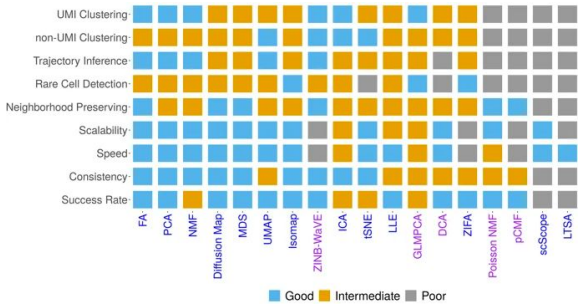


Comparison of 18 Methods

Sun, S., Zhu, J., Ma, Y. *et al.* . *Genome Biol* **20,** 269 (2019)

PCA

Sun et al., 2019 applied PCA of covariance matrix



Good Intermediate Poor

```
tryCatch({
        ct <- system.time({
        res_pca <- prcomp(norm_counts, center = TRUE, scale. = FALSE)
        res_pca <- res_pca$rotation[, seq_len(num_pc), drop = FALSE]
        })
        # count time
        ct <- c(user.self = ct[["user.self"]], sys.self = ct[["sys.self"]],
    user.child = ct[["user.chi
    elapsed = ct[["elapsed"]])
        list(res = res_pca, ct
},
```

prcomp

```
function (x, retx = TRUE, center = TRUE, scale. = FALSE, tol = NULL,
    rank. = NULL, ...)
{
    chkDots(...)
    x <- as.matrix(x)
    x <- scale(x, center = center, scale = scale.)
    cen <- attr(x, "scaled:center")
    sc <- attr(x, "scaled:scale")
    if (any(sc == 0))
        stop("cannot rescale a constant/zero column to unit variance")
    n <- nrow(x)
    p <- ncol(x)
    k <- if (!is.null(rank.)) {
        stopifnot(length(rank.) == 1, is.finite(rank.), as.integer(rank.) >
            0)
        min(as.integer(rank.), n, p)
    }
    else min(n, p)
    s <- svd(x, nu = 0, nv = k)
    j <- seq_len(k)
    s$d <- s$d/sqrt(max(1, n - 1))
```

https://github.com/xzhoulab/DRComparison/blob/master/algorithms/call_PCA.R

2 forms of PCA Covariance, Correlation

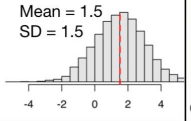
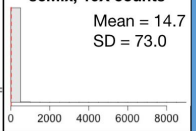
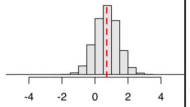
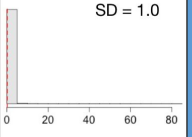
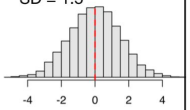
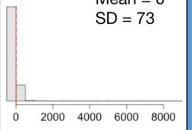
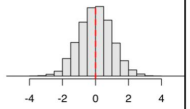
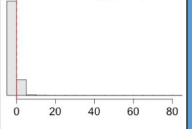
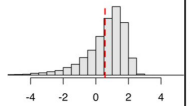
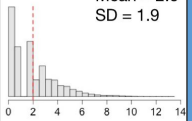
Goal :: Transform data so that variance will be informative when pulled apart with SVD.

Covariance-based PCA

Correlation-based PCA

Need to address

- sparsity
- heteroscedasticity

Example raw datasets		Graphical Examples: Toy data	Graphical Examples: scMix, 10X counts
		Mean = 1.5 SD = 1.5 	Mean = 14.7 SD = 73.0 
Method	**Formula**		
Scale	$x_{i,j}^{*} = \frac{x_{i,j}}{scaling\ factor}$ where the *scaling factor* can be a size or data dispersion measure	Mean = 0.7 SD = 0.7 	Mean = 0.2 SD = 1.0 
Center	$x_{i,j}^{*} = x_{i,j} - column\ mean$	Mean = 0 SD = 1.5 	Mean = 0 SD = 73 
Standardize	$x_{i,j}^{*} = \frac{x_{i,j} - column\ mean}{scaling\ factor}$ where the *scaling factor* can be a size or data dispersion measure. For example z-score subtracts means, divides by standard deviation	Mean = 0 SD = 1 	Mean = 0 SD = 1 
Transform	$x_{i,j}^{*} = f(x_{i,j})$ where *f(x)* is the transformation function, for example logarithms are commonly used	Mean = 0.5 SD = 1.4  *Log_2 transformation*	Mean = 2.0 SD = 1.9  *Log_2 transformation, pseudocount of 1*

Research Topic

Multi-omic Data Integration in Oncology

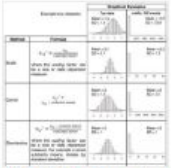
Submission closed.

frontiers in Oncology | Cancer Genetics

Overview | 13 Articles | 121 Authors

Articles

By Views | By Type | By Date



Impact of Data Preprocessing on Integrative Matrix Factorization of Single Cell Data

Lauren L. Hsu and Aedin C. Culhane

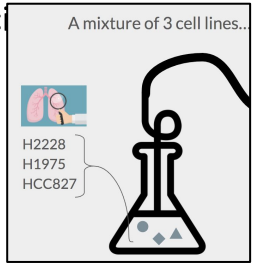
Mini Review Integrative, single-cell analyses may provide unprecedented insights into cellular and spatial diversity of the tumor microenviro sparsity, noise, and high dimensionality of these data present unique challenges. Whilst approaches for ...

Published on 23 June 2020
Front. Oncol. doi: 10.3389/fonc.2020.00973

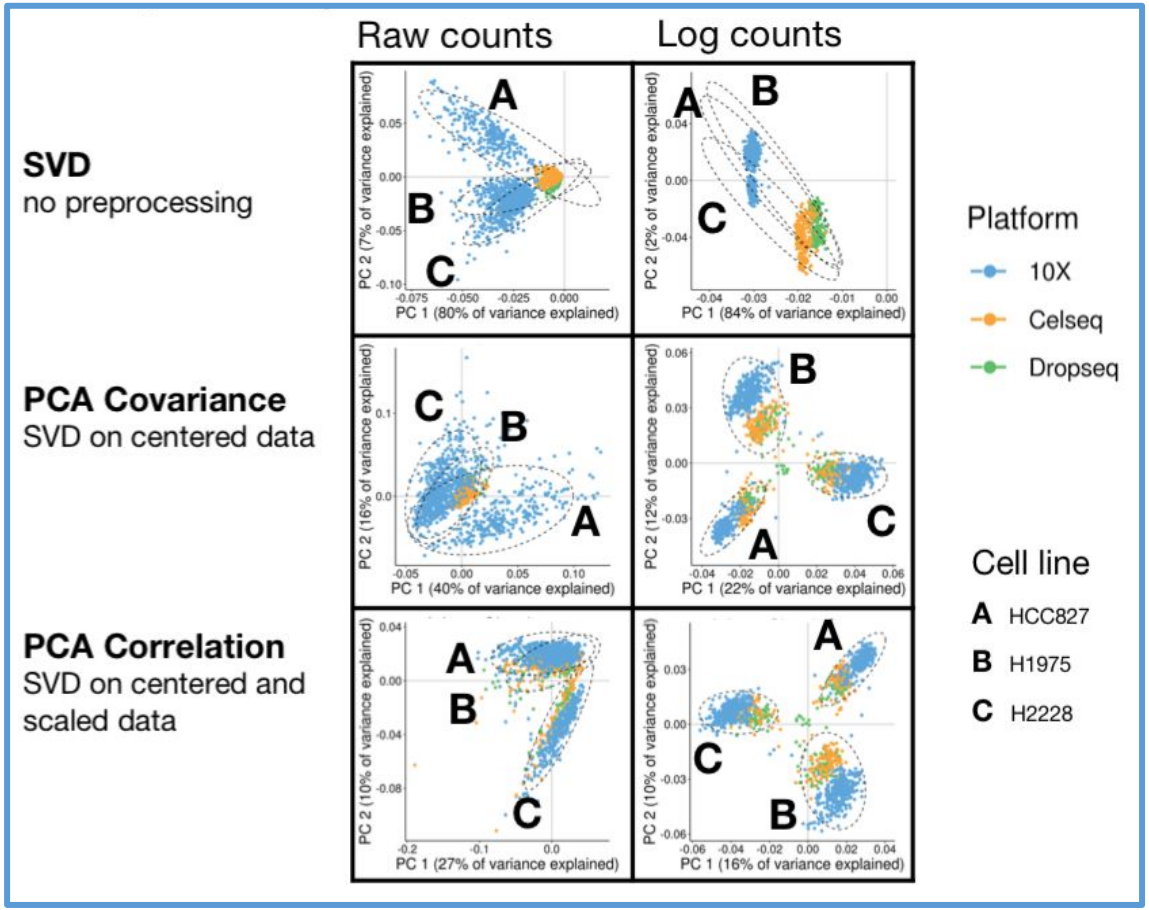
3,467 total views Altmetric 102

Processing steps impact results

human lung adenocarci
cell lines
HCC827
H1975
H2228

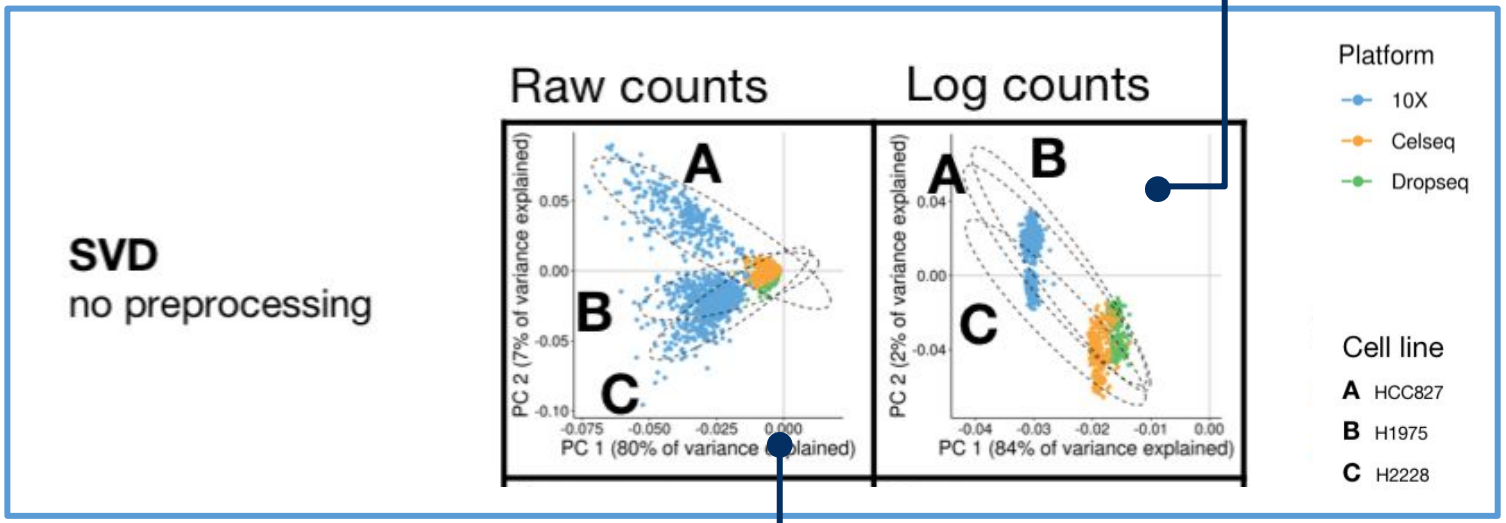


Hsu & Culhane,



Preprocessing Impacts on PC1

Centering is important: orthogonal vectors are uncorrelated only when at least one of them has mean 0.



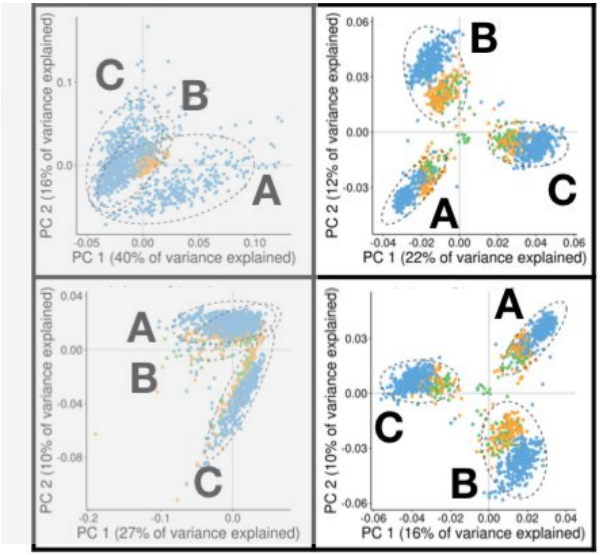
Arch effect: points on PC1 lie on 1 side of the origin

Even when cell lines cluster & distinct, a platform effect remains

Raw Counts Log Counts

PCA Covariance
SVD on centered data

PCA Correlation
SVD on centered and scaled data

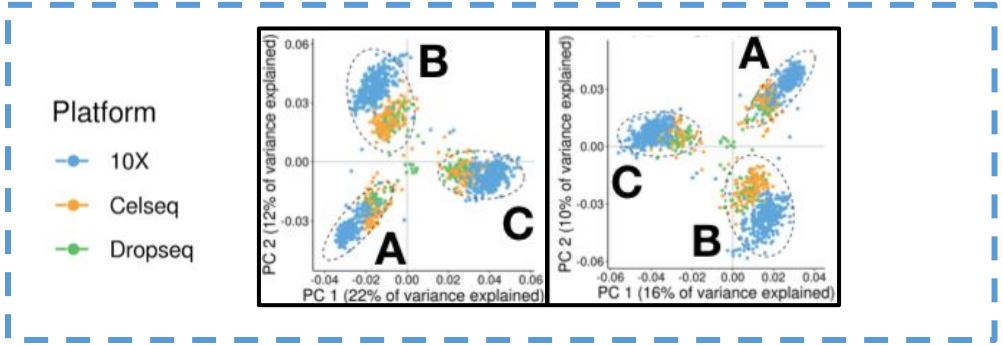


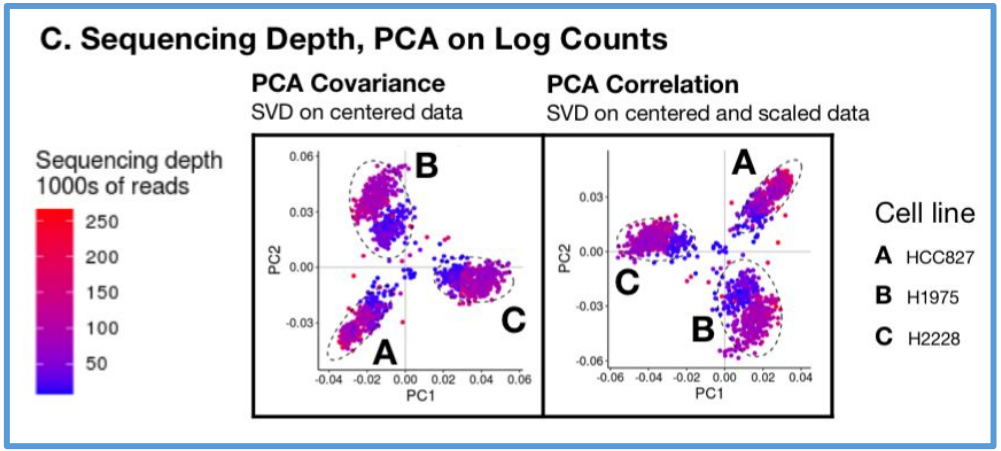
Platform
- 10X
- Celseq
- Dropseq

Cell line
- **A** HCC827
- **B** H1975
- **C** H2228

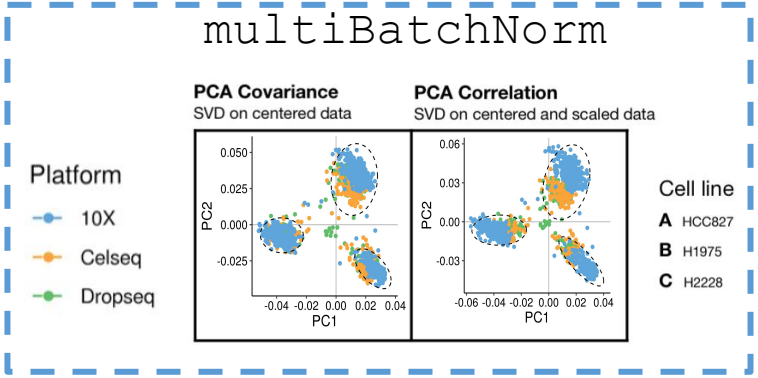
Hsu & Culhane, 2020

Sequencing depth + PCA





Correction with `multiBatchNorm`



The separation of the 10X batch from the others is driven by systematic differences in sequencing depth between the platforms. Not correct by Z-score

Beyond PCA- Other matrix factorization methods

Table 2. Dimension reduction methods for one data set

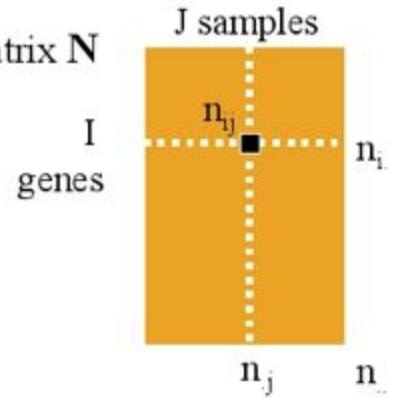
Method	Description	Name of R function {R package}
PCA	Principal component analysis	prcomp{stats}, princomp{stats}, dudi.pca{ade4}, pca{vegan}, PCA{FactoMineR}, principal{psych}
CA, COA	Correspondence analysis	ca{ca}, CA{FactoMineR}, dudi.coa{ade4}
NSC	Nonsymmetric correspondence analysis	dudi.nsc{ade4}
PCoA, MDS	Principal co-ordinate analysis/multiple dimensional scaling	cmdscale{stats} dudi.pco{ade4} pcoa{ape}
NMF	Nonnegative matrix factorization	nmf{nmf}
nmMDS	Nonmetric multidimensional scaling	metaMDS{vegan}
sPCA, nsPCA, pPCA	Sparse PCA, nonnegative sparse PCA, penalized PCA. (PCA with feature selection)	SPC{PMA}, spca{mixOmics}, nsprcomp{nsprcomp}, PMD{PMA}
NIPALS PCA	Nonlinear iterative partial least squares analysis (PCA on data with missing values)	nipals{ade4} pca{pcaMethods}[a] nipals{mixOmics}
pPCA, bPCA	Probabilistic PCA, Bayesian PCA	pca{pcaMethods}[a]
MCA	Multiple correspondence analysis	dudi.acm{ade4}, mca{MASS}
ICA	Independent component analysis	fastICA{FastICA}
sIPCA	Sparse independent PCA (combines sPCA and ICA)	sipca{mixOmics} ipca{mixOmics}
plots	Graphical resources	R packages including scatterplot3d, ggord[b], ggbiplot[c], plotly[d], explor

[a]Available in Bioconductor.
[b]On github: devtools::install_github ('fawda123/ggord').
[c]On github: devtools::install_github ('ggbiplot', 'vqv').
[d]On github: devtools::install_github ('ropensci/plotly').

From our Briefings in Bioinformatics review. Meng et al., 2016
https://academic.oup.com/bib/article/17/4/628/2240645

COA: Initial Transformation

Matrix **N**

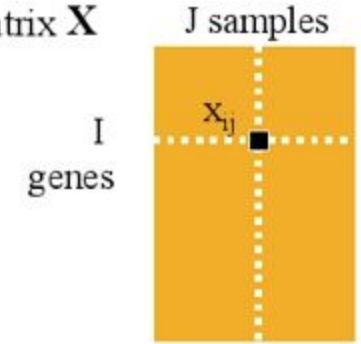


$$c_j = n_{.j}/n_{..}$$

$$r_i = n_{i.}/n_{..}$$

$$p_{ij} = n_{ij}/n_{..}$$

Matrix **X**



$$x_{ij} = (p_{ij} - r_i c_j) / \sqrt{r_i c_j}$$

Pearson chi-square statistic $O_{ij}-E_{ij} / \sqrt{E_{ij}}$

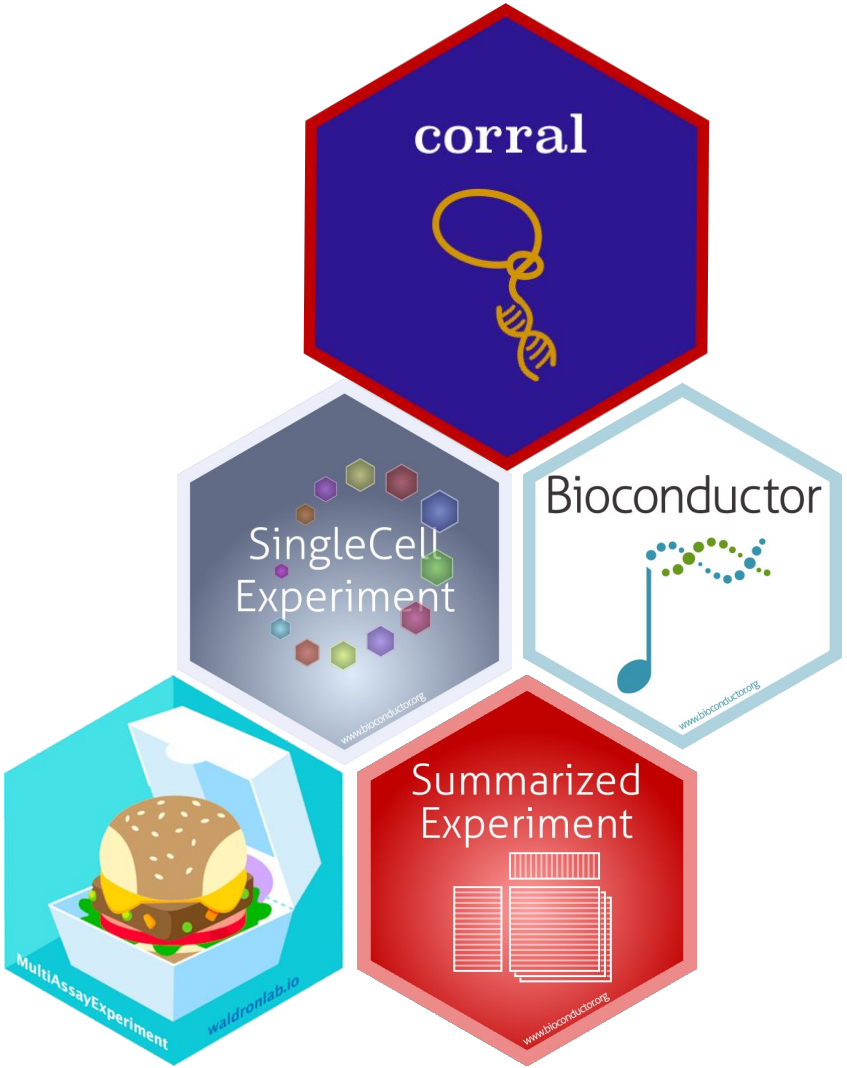
`corral` package

Optimized for single cell data

- Uses sparse matrices (`Matrix`)
- Applies fast SVD approximation (`irlba`)
- Modular: can easily apply random or other SVD
- Interacts directly with Bioconductor objects

Chan Zuckerberg Initiative

HUMAN CELL ATLAS



Corral v PCA of scMix Data

Cell line

A HCC827

B H1975

C H2228

Platform

10X

Celseq

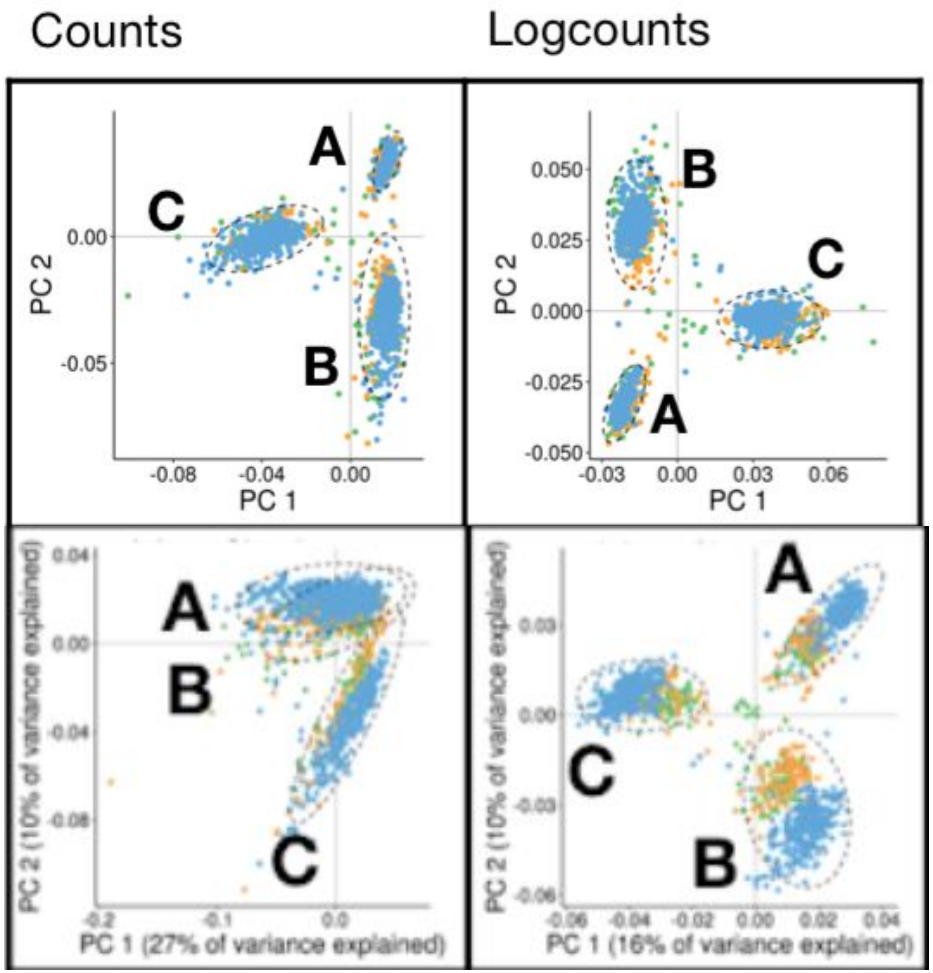
Dropseq

corral

corralm

PCA

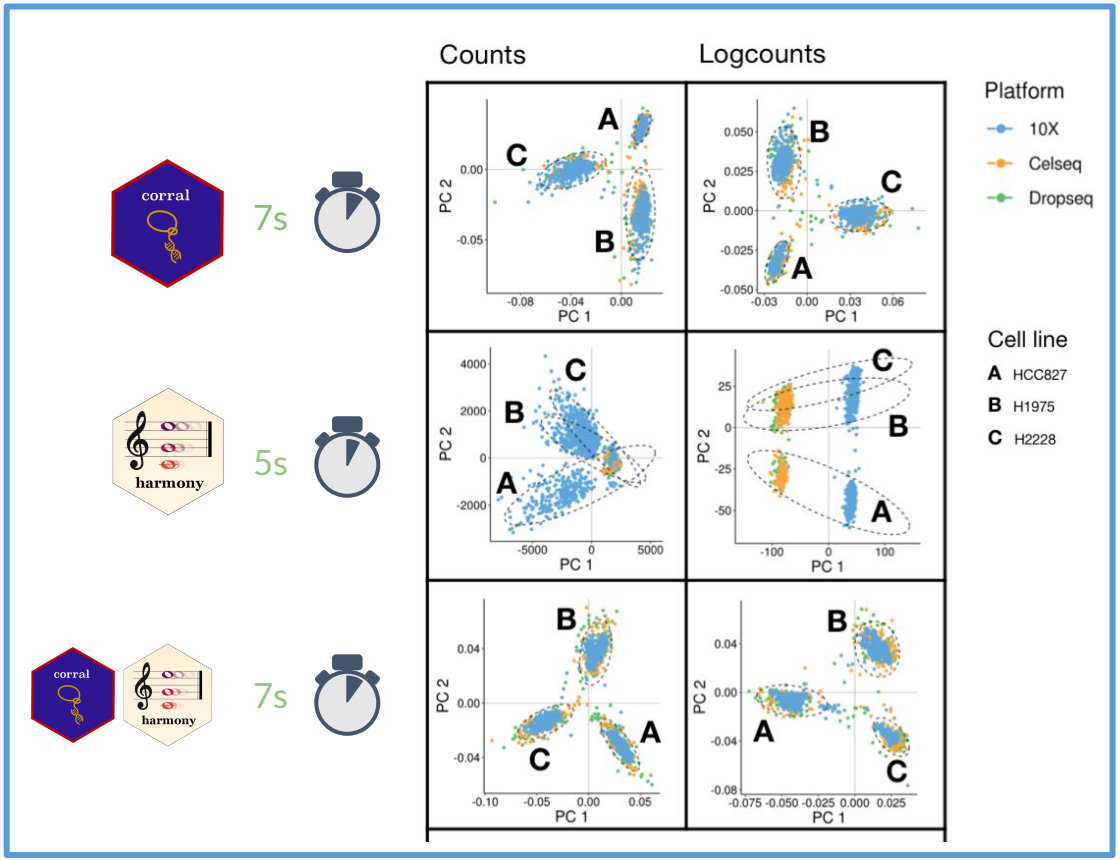
Correlation
Matrix
Scale+Center



`corral` is fast and can replace PCA to improve scRNAseq workflows;

- Data integration
- Clustering

Performs PCA, then iterative soft clustering
(Korsunsky, et al. 2019)

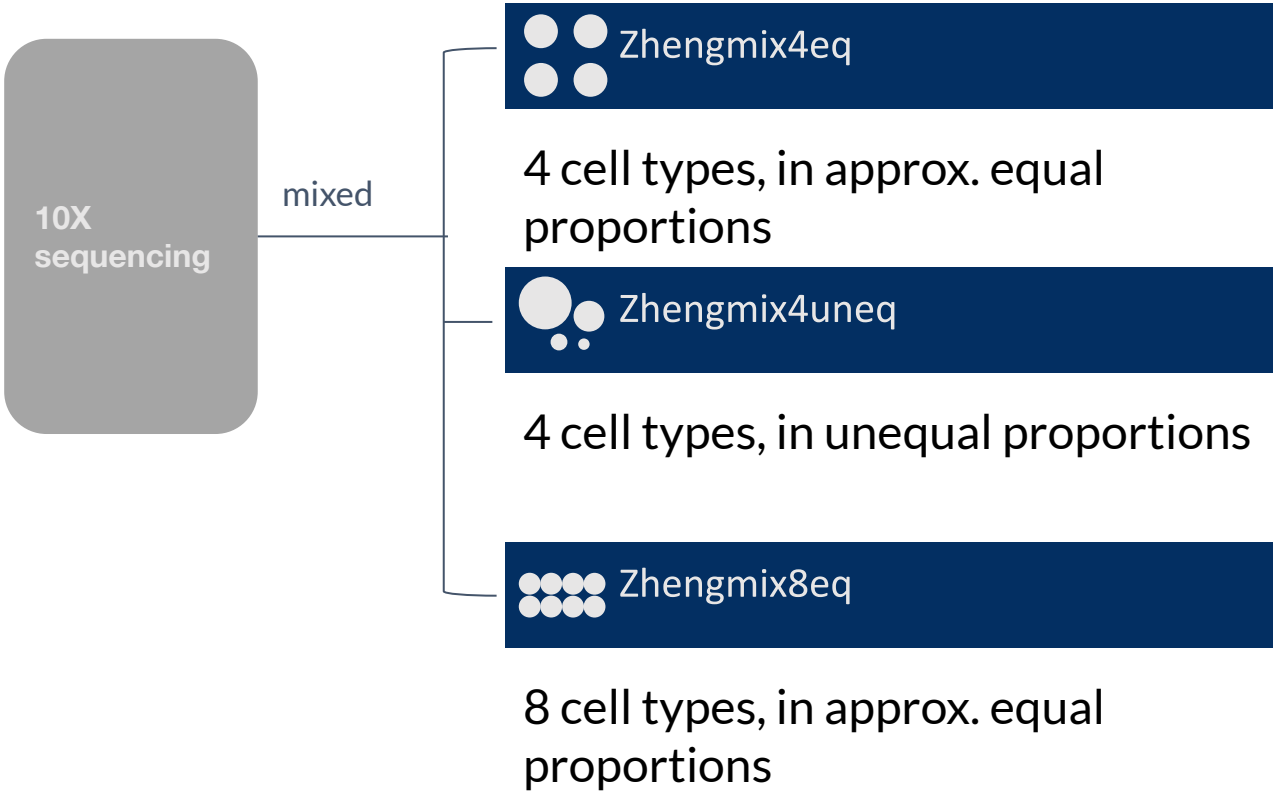


Preliminary data, Unpublished

Benchmarking Clustering: Zhengmix (`DuoClustering2018`)

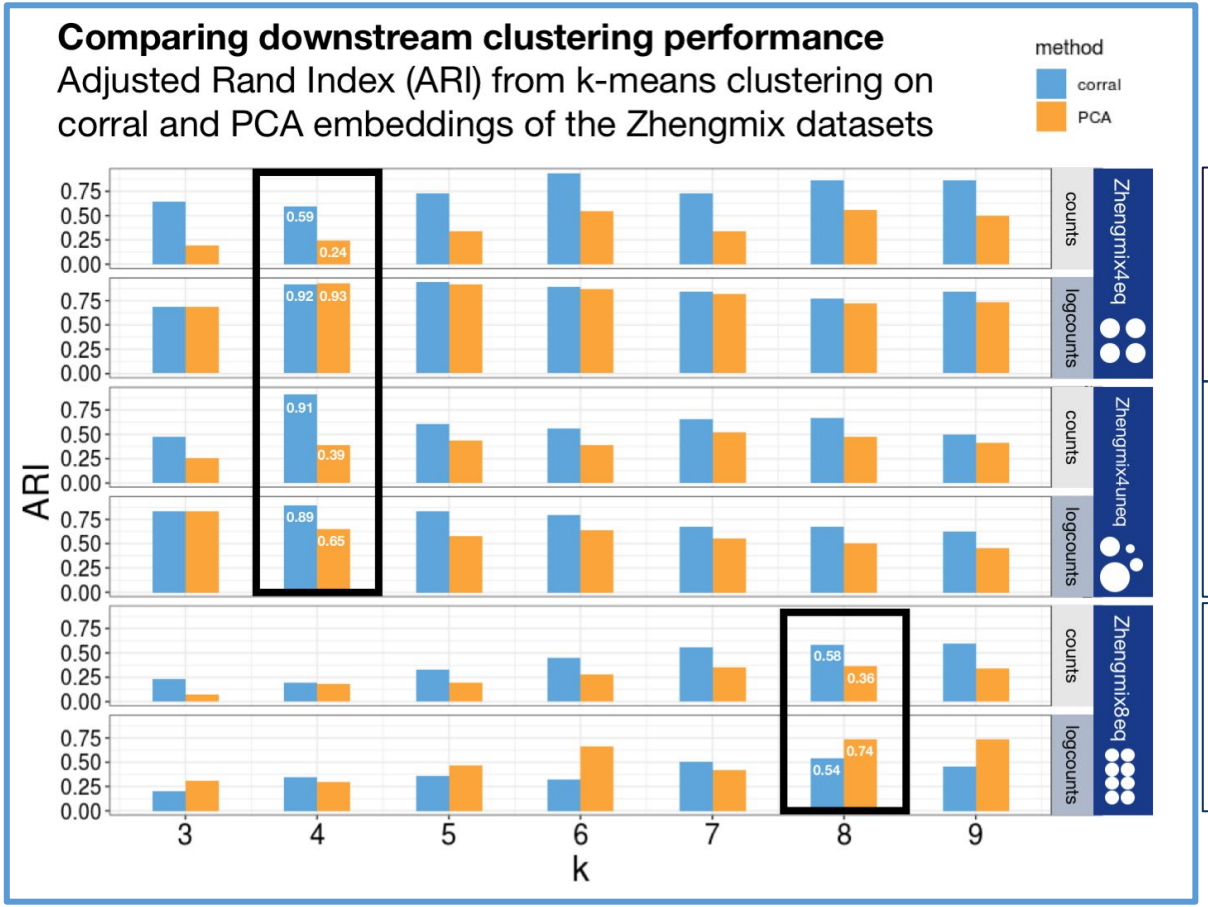
Pre-sorted cells, including:

1. B-cells
2. CD14 monocytes
3. CD4 T-helper cells
4. CD56 NK cells
5. memory T-cells
6. naive cytotoxic T-cells
7. naive T- cells
8. regulatory T-cells



Clustering Performance: Corral v PCA before K-means clustering

Corral > PCA in all but 1 comparison



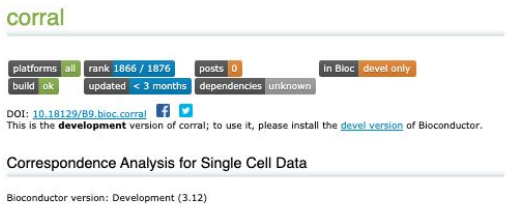
4 Clusters Equal proportions

4 Clusters Unequal proportions

8 Clusters Equal proportions

Preliminary data, Unpublished

`corral` package
Bioconductor devel



Improve performance

-> Speed (IRLBA)

-> Performance/HDF5

-> Multi-dataset

Talk to be presented at BioC2020 conference