[Harshit Ghosh](mailto:hg2604@columbia.edu), Marek Kampa

**Code Repository:**

<https://github.com/Karmpje/COMS4761_Spring2023_Group18>

**Introduction**

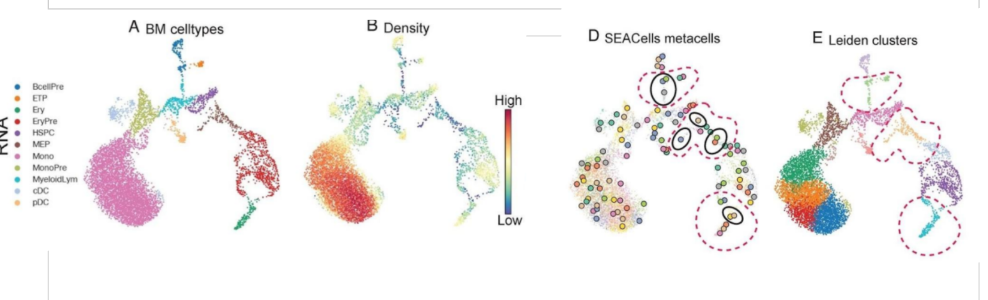
Single-cell genomics gives us the possibility to discern biological phenomena at the resolution of a single cell (as the name suggests). It allows us to identify heterogeneous cell states within tissue samples, hitherto not possible, as bulk sequencing aggregates everything and gives us an average dominated by the most abundant cell states.

All that being said, single-cell data comes with its own host of problems. First, only a small fraction (~10%)[citation] of the target population (transcripts(scRNAseq) or Nucleic Acid (Assay Based Seq, ex:scATACseq) is captured in any single cell assay. As a result, single-cell data is very noisy. Second, because of this low capture rate, lowly expressed reads have the possibility of not being captured at all, hence we end up with a very sparse raw count matrix. Making things even more complicated, single-cell measurements also depend on the experimental protocol and the instrument used, referred to as the technical batch effect. There is no way to easily differentiate between technical noise and biological variation making further downstream analysis difficult.

**Traditional Clustering & Metacell**

Traditionally, to make single-cell data amenable for analysis, the first step has always been the clustering of the data. However, clustering algorithms have been shown to group different cell types together, i.e., there is non-homogeneity within clusters. Also, these algorithms fail to recognize rare cellular states, which play a vital role in the pathogenesis of most diseases.

To overcome these problems, recently the concept of a Metacell was introduced. Metacells are defined as groups of cells that represent distinct cell states whereby within metacell variation is due to technical rather than biological sources. They are constructed in a manner that optimises for homogeneity within cell groups. The measurements within a Metacell can be thought of as coming from the same distribution.

****

**Fig:** SEACell vs Leiden Clustering. Leiden Clustering fails to identify rare cell states (1A vs 1E). SEACells has no trouble.(1A vs 1D)

**How are Metacells Constructed?**

**The SEACells algorithm:**

**Assumptions:**

Biological Systems consist of well-defined and finite sets of cell-states.

**Inputs:**

Raw count Matrices, low-dimensional representation of data, # of Metacells

**Steps**:

1. kNN graph using Euclidean distances between cells in the low-dimensional space.
2. Affinity matrix (kernel matrix) of cell-to-cell similarities, constructed by transforming distances in the nearest neighbour graph to similarity scores. This is done using an adaptive gaussian kernel, which can account for the change in density as we move along the phenotypic space.
3. Kernel matrix bridging to archetypal analysis. (See below)
4. Identification of groupings results in the SEACells metacells.
5. Normalization of metacell count matrices opens avenues for further analyses (gene-regulatory analyses, visualization, data integration, trajectory inference, etc.)

Metacells are identified by performing Archetypal analysis on the Kernal Matrix. Archetypal analysis finds archetypes in a given dataset. Each observation can be thought of as a linear combination of the archetypal observations. At the same time archetypes themselves are linear combinations of the observations.

The archetypal analysis algorithm finds archetypes as points that sit on the convex hull of the data set. They can be thought of as the vertices of a convex polytope in the multi-dimensional phenotypic space.

**Aim of The Project:**

At the outset, our goal was to interpret the different facets of the d-dimensional polytope. In an attempt to understand what defines (in terms of biological significance) an edge connecting two vertices or a plane defined by multiple vertices of the polytope.

**Motivation:**

Currently the number of metacells is a parameter to the SEACells algorithm. We proposed that a better understanding of the architecture of the convex hull would help us better optimize the number of metacells.

Dataset Used: CD34+ Bone Marrow Multiome Data (6881 cells)

**Projecting cells to the facets of the Convex Hull**

Our metacells (Archetypes) lie in the kernal space which has a dimensionality of 6881. For the initial analysis we decided to run the SEAcells algorithm (<https://github.com/dpeerlab/SEACells/blob/main/notebooks/SEACell_computation.ipynb>) for 90 metacells. This gave us a convex hull with 90 vertices in 6881 dimensions. The resulting polytope was an 89-dimensional simplex in 6881-D.

We projected our cells that lie in this kernal space onto the 1-d (90 choose 2) and 2-d facets (90 choose 3) of this polytope. We did not go beyond 2-dimensional facets as the number of possible facets increases very fast (90 choose d) and it becomes computationally expensive. Also, most of these facets have no cell projected onto them.

The scripts for the projection and construction of the convex hull are in “Group\_18\_Notebook\_1.ipynb”

This notebook does the following:

This notebook, borrowing from <https://github.com/dpeerlab/SEACells/blob/main/notebooks/SEACell_computation.ipynb> implements code that

* Builds the convex hull of SEACells metacells
* Projects single cells to the edges of this convex hull
* Projects single cells to the planes of this convex hull
* Creates the hash tables with intermediary results
* Creates the Pickle objects (downloadable) for further analysis offline.

Everything has been built and tested in Google Collaboratory.

In particular, there are the following sections of interest:

**Pickle Code**

Creates Pickle objects, which can be downloaded/uploaded for further analysis in other systems or offline.

**Edge Projections**

Implements:

* MB\_graph(M matrix @ B matrix) -> complete graph representing the convex hull
* project\_cells\_MB(single cell coordinates, MB graph) -> project cells to the closest edge. Returns: the closest edge assignment, minimum distance to the closest edge, relative position on the closest edge (expressed in range(0,1)), list of distances to all edges of the MB graph.
* metacells\_distance(MB graph) -> computes the distance between each two vertices of the convex hull (between each two metacells)

**Dictionaries**

The following section builds dictionaries that are subsequently downloaded as Pickle objects for further, faster and more optimized, processing.

* metacell\_dictionary(MB matrix) -> generates the metacells hash table with SEACell assignment as key and location as the value.
* edge\_dictionary(graph of MB) -> generates the edge hash table with edge number as the key, tuple(length and 2 Metacells) as the value.
* cell\_dictionary(hard SEACells Assignment , A matrix, M matrix, graph\_MB, dictionary range start value, dictionary range end value) -> generates the hash table with cell number as the key, tuple(hard SEACells Assignment, closest edge, distance to all edges, relative position on the edge, weight) as the value.

**3D Projections and Examples**

* create\_3combinations\_matrix(MB matrix) -> simplifies the procedure of computing N choose 3 by producing a matrix whose 3 consecutive rows represent the output of N choose 3 function. Returns this matrix as well as a list which keeps track of original indices of each of the matrix rows.
* plane\_graph(M matrix, original indices) -> builds a graph in which each vertex represents a plane of the N choose 3 SEACells metacells. Edges in this graph would represent the intersection between two given planes [NOT IMPLEMENTED].
* project\_cell\_to\_graph(single cell coordinates, graph of planes) -> project the single cell to planes in the graph of planes. Returns the closest plane assignment, distances to all the planes in the graph, as well as the position on the closest plane.

**Analysis of Projections**

The scripts for the analysis of projected cells can be found in “Group\_18\_Notebook\_2.ipynb”

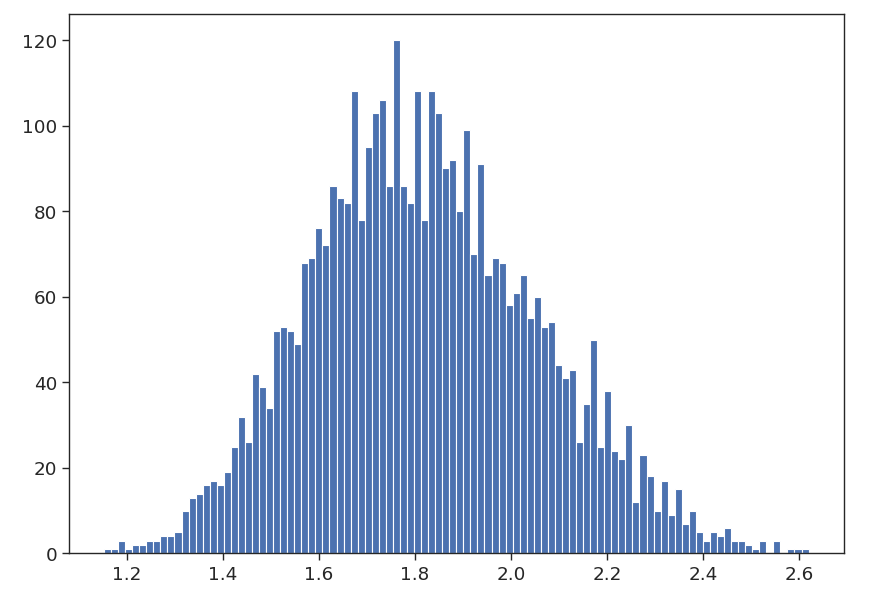


Fig: Distribution of the edge lengths of the polytope. The distances are in the kernel space.

The edge lengths of the polytope are normally distributed with a mean of 1.8 and a standard deviation of 0.2.

**Projection of Cells onto Edges:**

****

**Fig:** Cells are projected onto an edge that contains its metacell assignment 90% of the time. This is expected as the cell is expected to be nearest to its metacell. Surprisingly though, when this is not the case, the certainty in the metacell assignment is higher (0.6 compared to 0.42).

Cells are assigned to an edge that is closest to it. Next, we plotted the distances of a few cells to all the edges of the polytope.

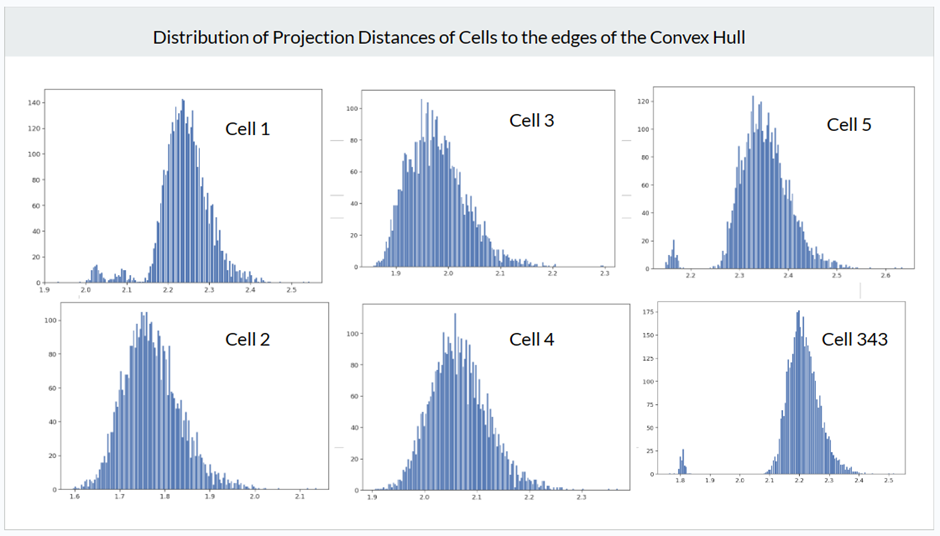
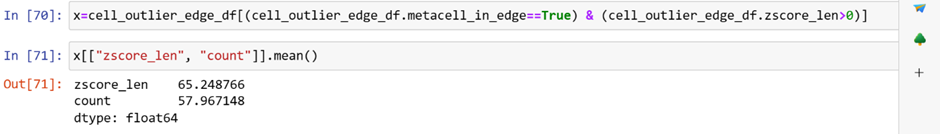


Fig2: Distance to edges for Cells 1-5 and 343.

As can be seen in the figure above, the distance to the edges is normally distributed, but for certain cells, it seems to be a multi-modal distribution.

To understand this further, for each cell, we identified the set of edges that were 3 standard deviations below the mean, henceforth referred to as outlier edges. We asked whether these outlier edges have any metacell in common.



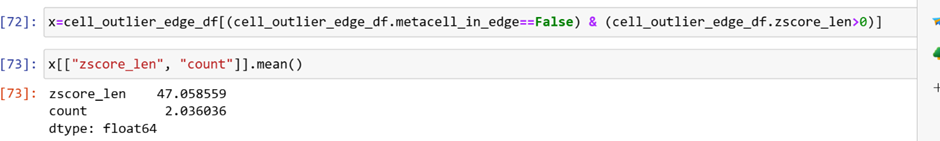
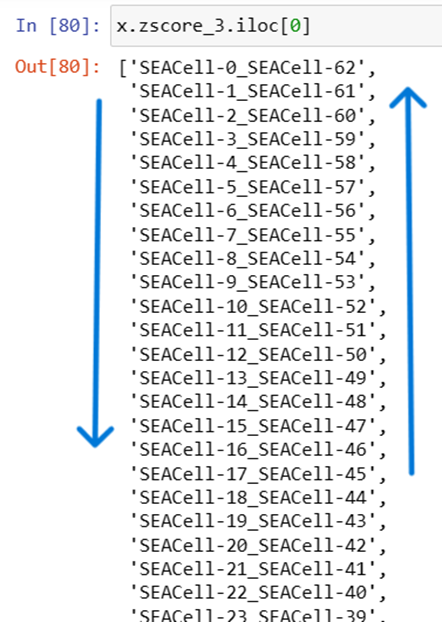


Fig:

In the above figure, zcore\_len represents the number of outlier edges, and count represents the number of those edges that contains the cell's metacell assignment. In cases when the edge closest to a cell contains its associated metacell, most of the outlier edges also contain the assigned metacell (58/65~89%; Top Table).

In cases where the closest edge does not contain the associated metacell (Bottom Table). The outlier edges seem to be random(2/47~4%). With no Metacell being common to the edges. Although, for a few cells I did see a pattern as indicated by the blue arrows, attributing this to chance feels eerie. Maybe, it has something to do with how the convex hull is being constructed.(See below figure)



**Statement about Angles**

In our final presentation, we also presented angles between these outlier edges. The result we had got said that all the edges were perpendicular to each other, this turned out to be incorrect. Our script was calculating the angles between Metacell-origin-Metacell instead of Metacell1-Metacell2-Metacell3.

**Regulatory And Structural Correlations**

The scripts for the analysis of projected cells can be found in “Group\_18\_Notebook\_3.ipynb”

Next, we wanted to see whether the structure of the convex hull could tell us something about the overall regulatory architecture of the metacells, where each cellular state should ideally be representing a single cell state.

To compare gene regulation between two Metacells, we decided to compare which genes are differentially regulated among a given pair. We used the metric “Gene Score” to define the SEACells paper for this purpose. The “Gene Score” was computed using both the RNAseq and ATACseq modalities and represents the extent to which a given gene is being regulated. We utilized the public notebook: [SEACells/SEACell\_ATAC\_analysis.ipynb at main · dpeerlab/SEACells (github.com)](https://github.com/dpeerlab/SEACells/blob/main/notebooks/SEACell_ATAC_analysis.ipynb)

We wanted to know whether there was a correlation between the edge length of the polytope and differential regulatory architecture.

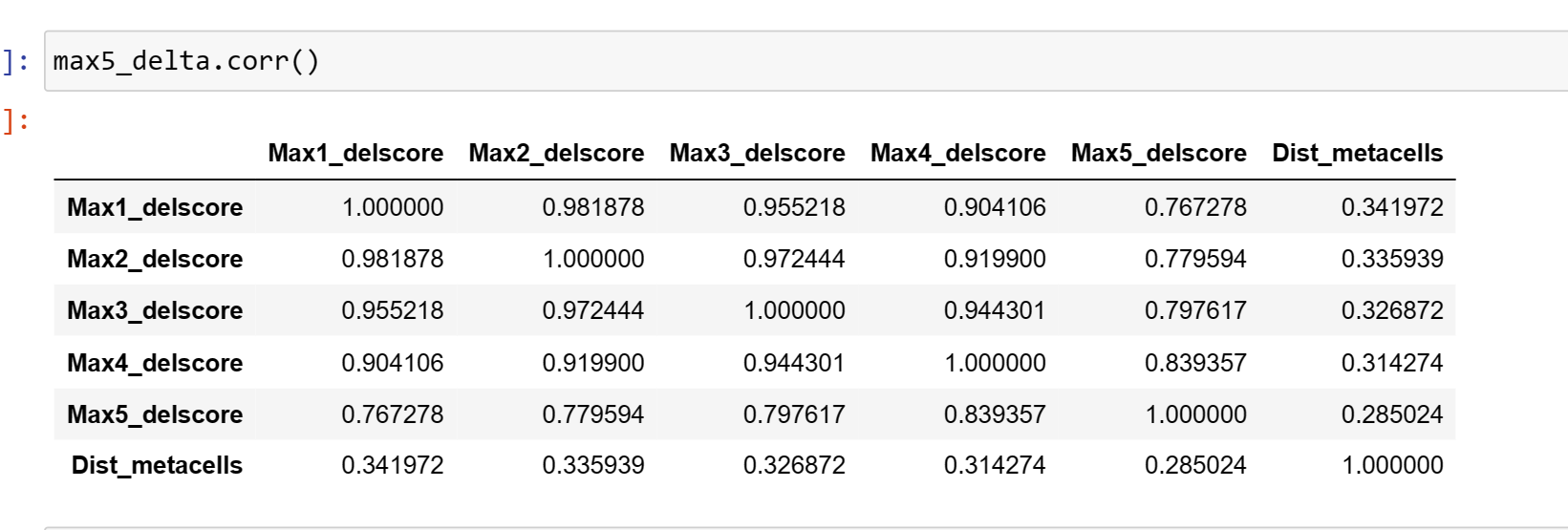
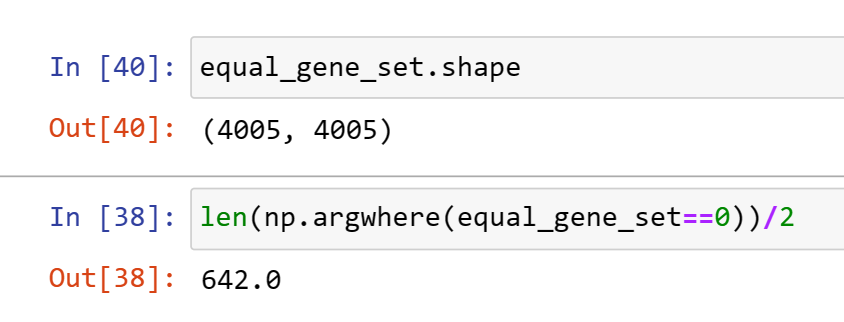


Fig: Correlation between Edge length (‘Dist\_metacells’) and the top 5 most differentially regulated genes among the metacells that constitute the edge.

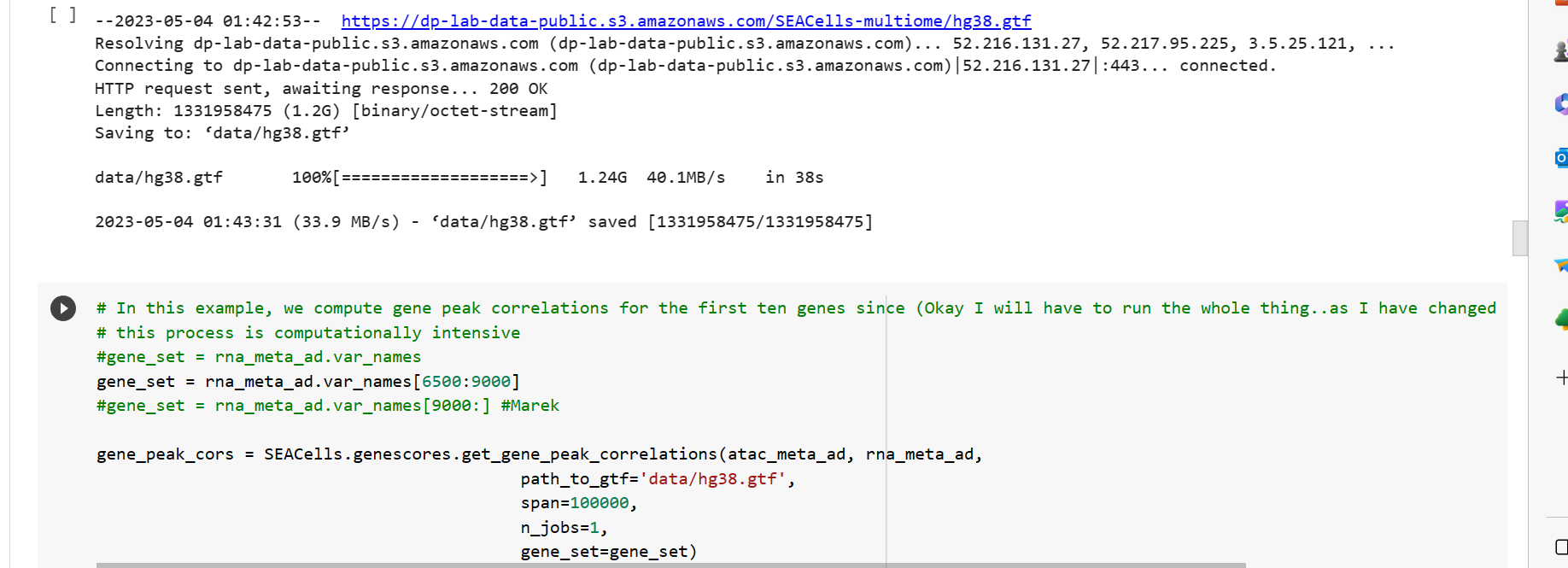
From the above table (last column) we see a mild correlation, indicating that metacells further apart are in “more different” cell states!

To dig further into the specifics, of the 12500 genes, we selected the 1500 most variable genes. We let each edge in the polytope represent the difference in gene\_scores (for the 1500 genes) between the metacells that the edge connects. This is tantamount to subtracting two 1500-dimensional vectors. For each edge, we set a cut-off (1.2) to identify genes that have significant regulatory differences between the metacells. After performing these steps we now had a set of genes tied to each edge. If every possible cell state was represented by a single metacell then the gene set tied to each edge should be unique, indicative of the regulatory differences between the cell states.



Our polytope has a total of 4005 edges, which means comparing every edge to every other edge results in 4 million comparisons. Of these 4 million comparisons 642 were identical.

The next step would have been to reduce the number of metacells in the SEAcell algorithm to 81, or some other number less than 90, and see how this changes. Are all edges, unique? Does the number 642 go down? Unfortunately, we were not able to ascertain this, as we were not able to compute the gene scores for the case of 81 metacell.



For some reason, the script “[SEACell\_ATAC\_analysis.ipynb](https://github.com/dpeerlab/SEACells/blob/main/notebooks/SEACell_ATAC_analysis.ipynb)” kept crashing at this step, when run for 81 metacells.