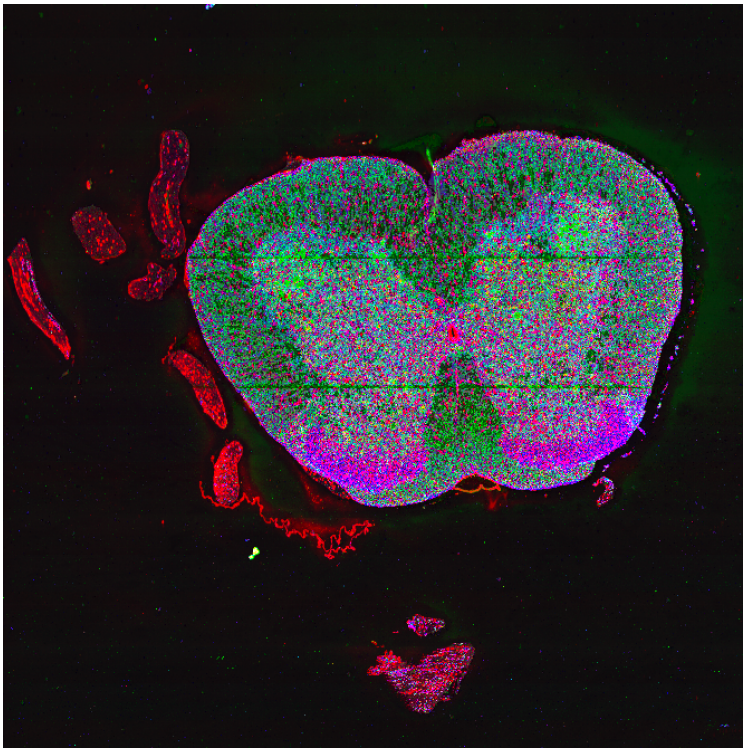
# *HUDSON*

A computational pipeline for spatial analysis of multiplexed images.



***Co-First Authors:*** *Jagjit Singh, Kunal Pandit*
***Supervisor****: Sanja Vickovic*
***Collaborators:*** *Rui Fu*

**INDEX**

## Overview:

Multiplexed imaging technologies are amongst the fastest-growing areas of multi-omics. These technologies capture many parameters of single cells while preserving their spatial location. As such, there is a need for a flexible, robust, and easily deployed end-to-end computational pipeline for processing and analysis of the data such technologies produce. The goal of such a pipeline is to capture cell-type compositions in a localized shape-preserving manner. To help researchers achieve this goal, we have developed Hudson. It is a fully automated computational pipeline for spatial analysis of multiplexed images obtained primarily via PySeq2500, which is an open-source toolkit for repurposing the HiSeq2500 sequencing system as a versatile fluidics and imaging platform. Hudson preprocesses and segments the multiplexed image to identify micro-environments within the tissue section and performs spatial analysis upon the computed micro-environments. Hudson uses the Snakemake workflow management system and can be scaled up from a personal computer to a research computing cluster for heavy workloads. The pipeline begins with the essential pre-processing of the image. This includes background correction, registering channels, stitching the image tiles together, and removing overlapping regions. As part of preprocessing, Hudson deploys the PICASOnn algorithm to perform unmixing of signals from dyes with overlapping emission spectra. Once pre-processing is complete, the spatial analysis phase begins with instance segmentation to detect image labels and computation of mean intensity per label. The mean intensities for all available markers in each label are then used for cell type identification via clustering and comparison with a reference single-cell sequencing dataset. The cell type identifiers and other image properties including the labels and label centroids are output as an Anndata object. Finally, Hudson extracts data from the Anndata object to build local spatial environments, compute relevant spatial statistics, and build a cell connectivity graph for researchers to use in their own downstream analysis. Importantly, the end result is consistent with the input data regardless of the underlying system architecture on which Hudson is deployed. Given the computationally intense and programmatically advanced nature of analyzing highly multiplexed images, Hudson saves significant time for researchers wanting to incorporate spatial information and serves the overarching goal of making spatial analysis more accessible.

## Jagjit's Role as Co-First Author :

My role was to develop programs and algorithms to help achieve a major percentage of the goal mentioned in the overview. To be specific, it required me to write multiple programs which collectively implement multiplexed proteomic spatial and single-cell analyses of whole mouse and human tissue sections from imaging datasets. These programs include anatomical annotation of sections, identifying cell types, computing spatial-temporal patterns, and finally storing the computed patterns in compressed format. I also had to use advanced statistical methodology to compute spatial autocorrelation of specific protein markers. My work resulted in the development of spatial neighborhoods/microenvironments over the tissue section being studied.

I also had to find better ways of annotating and compressing the imaging data. This is especially important as it makes collaboration much easier and reduces the cost of storing summary statistics from bio-medical analysis which can cumulatively become very costly for research labs. Since the images average around tens of millions of pixels, the task required the code to be run parallelly over multiple machines in a research computing cluster. Furthermore, I also added functionality that facilitates the use of graphical processing units for accelerating the computation.

## Background

New multiplexed imaging technologies are helping us capture many parameters of single cells while preserving their spatial location. As stated previously, the goal is to capture cell-type compositions in a localized shape-preserving manner. This is important for the development of treatments for diseases such as Colorectal Cancer (CRC) as distinct, histologically observed forms of its spatial architecture are linked to prognosis. Moreover, spatial analysis of tissue sections can be used as a classification tool between mutant and non-mutant tissue sections. Spatial analysis is essentially studying the spatial distribution of the cell types over the tissue region. These spatial distributions are used for detecting spatial patterns of cancer cell communication to check if a therapy is working or not.

# Methods

The main objective of the pipeline is to develop spatial neighborhoods over the observed tissue section, which is achieved through multiple individual steps, the steps being:

## 1. Save Raw

Data preprocessing begins with converting the unprocessed images to a Zarr store. Zarr is a format for the storage of chunked, compressed, n-dimensional arrays. Thus, it is very suitable for storing multiplexed tissue images as they are of the same form. Thus, imaging from the section is turned into a multidimensional Zarr data array containing all channels and objective steps for that section. The Zarr array is then output with a YAML file of the section summary. Additionally, performance log files are generated for debugging purposes.

## 2. Fix Lighting

This rule performs image pre-processing via a number of different tasks. The tasks are background correction, registering channels, stitching the image tiles together, and removing overlapping regions. In the current use case of an Illumina HiSeq 2500 sequencing system, the camera it employs is the Hamamatsu line scanning CCD Timing Delaying Integration (TDI). TDI enables capturing an image of a moving object while transferring integrated signal charges synchronously as the object moves to achieve high sensitivity, which is particularly suitable for low-light-level scanning applications. Starting with background correction the pixels in these cameras are arranged in a line in groups of 256 or 512 pixels (depending on the model). Each group has a different dark pixel value which is corrected to create a flat background. That is followed by channel registration. The context here is that the HiSeq system captures 4 emission channels simultaneously, however, the images from these channels are not perfectly aligned and differ from machine to machine. Thus, machine-specific sub-pixel channel shifts are pre-computed during machine commissioning. This is done via phase cross-correlation. Channel shifts are applied using a rigid affine transformation.

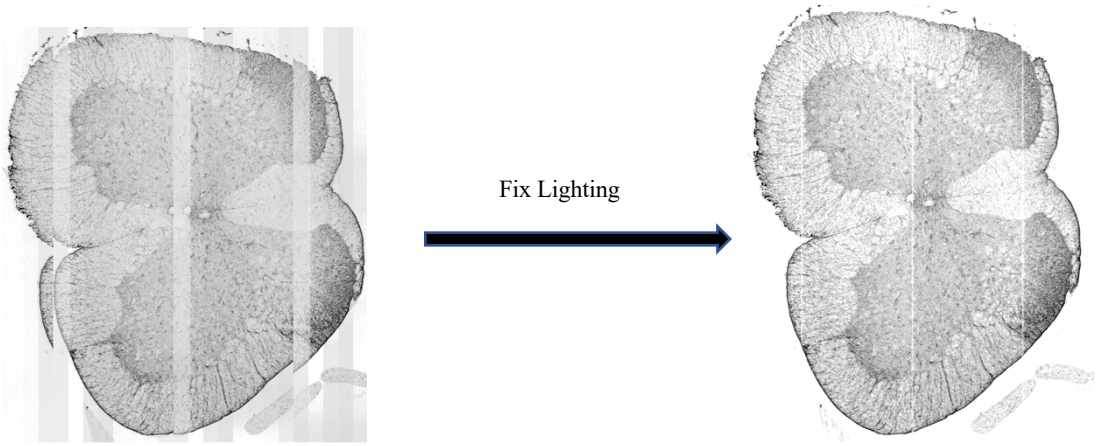



*Figure 1: Fix Lighting step starts with the raw Zarr image (Left) and produces the processed image (Right)*

Figure 1 is an example of the input and output from this step. We can see that the output is much more clear and sharper, and most importantly the vertical bars have been minimized.

## 3. PICASSOnn

PICASSOnn is an algorithm to remove spillover fluorescence by minimizing the mutual information (MI) between sink and source images. MI between pairs of sink and source images is estimated and minimized with a neural network (MINE) using stochastic gradient descent and GPU acceleration. Following is a figure of the input and output from the PICASSOnn step

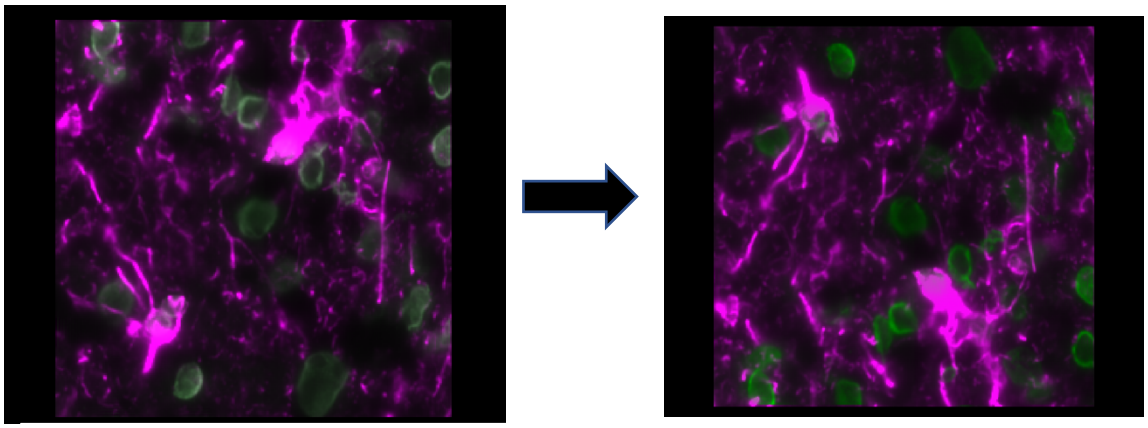


*Figure 2: PICASSOnn step applies the PICASSOnn algorithm to the image for spectral unmixing*

From figure 2 we can see that the green objects are much more enhanced. Thus, it really helps in better cell segmentation.

## 4. Segmentation

In the segmentation step, the pipeline performs instance segmentation on the image to generate a masked array of the same dimension as the input. The masked array is essentially the assignment of each pixel to a certain cell in the image. Hudson performs segmentation based on both cell and nucleic markers. Each cell is indexed as a label starting from 1$^{st}$ to the n$^{th}$ cell that is segmented. Below are images of sample segmentation by this step with the original image (left) and masks outlining the different labels/cells in the image (right).

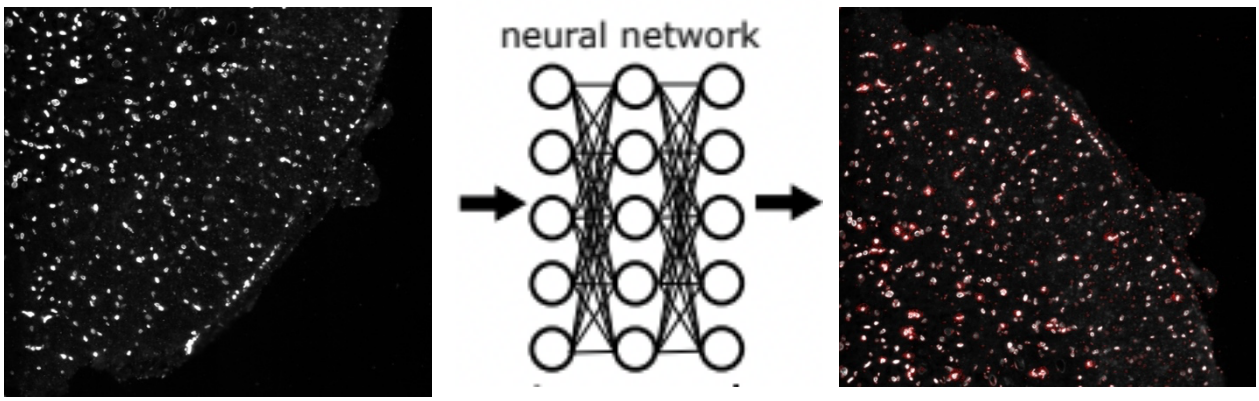



Figure 3: Unsegmented Image (Left) versus Segmented Image (Right)

## 5. Intensity Computation

This step in the pipeline computes mean intensities for all available markers in each label. This is one of the most computationally intensive steps and is crucial for detecting cell type composition. Following is an example of the mean pixel intensity per label in the case of Lamin-B1 and MBP nucleic marker.

{'LMN1b':[374.4911504424779,564.6321839080459,...],'MBP':[134.27433628318585,181.333
33333333334,..]}

*Figure 4: Mean pixel intensity per label for LMN1b and MBP markers*

As we can see above the pixel intensity per marker is saved as a 1-dimensional array for all available markers in the image. The length of the array is equal to the number of labels/cells found for that marker. All the intensity information is saved as a dictionary with the markers being keys and the values being the corresponding arrays containing the mean intensities. The pipeline will automatically scale the code over the compute cluster. The computation is further accelerated if a graphical processing unit is present in the system or cluster.

## 6. Data Object

Highly multiplexed images and can have a lot of characteristic data. These are essentially the regional properties of the image which characterize it. More importantly, the number of labels.

In this step, the image labels along with their cell type composition and location are condensed and stored in an Anndata object. Anndata is a Python package for handling annotated data matrices in memory and on disk, positioned between Pandas and Xarray. The region properties are then stacked in a 2-dimensional matrix, along with the mean intensities per label for each of the protein markers, which we covered in the mean intensity section. There is additional metadata and variable information added to the object. Following that, the data object is condensed using Gzip into an H5ad object. H5ad is AnnData's native storage format which is based on HDF.

## 7. Cell Type Identification

Processing and clustering: We treat protein intensity measurements and scalar spatial parameters for each segmented cell as separate modes of data to be separately arcsine transformed, scaled, and batch-corrected (if needed). Then, clustering is achieved through Phenograph. Alternatively, Seurat Weighed Nearest Neighbor Analysis (WNN) can be used to cluster cells and project into

two-dimensional space using both modalities, which also allows for flexibility in integrating multiple datasets.

*Cell type classification from reference*: Automated cell type classification is achieved by comparing all protein intensities with a reference dataset. This can be previously labeled scRNA-seq or PySeq2500 4i data, stored as a Seurat object in RDS format, as specified in the experiment `config/config.yaml` file (and metadata column name that contains cell type information in the reference). The underlying algorithm used here, Seurat Canonical Correlation Analysis (CCA), has been shown to be performant in cross-modal reference mapping. Cell type results are saved as CSV files in the output `tables` directory, to be imported into AnnData for downstream steps. If no appropriate reference is used, clustering results from above are saved instead.

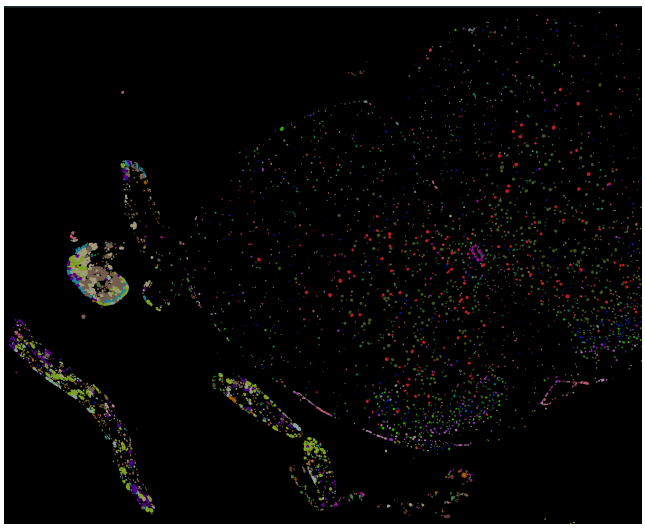


*Figure 5: Cell type clustering visualized on the tissue section.*

## 8. Network Graph

Hudson builds an undirected graph of cell connectivity based on a Delaunay triangulation. A Delaunay triangulation of a vertex set is a triangulation of the vertex set with the property that no vertex in the vertex set falls in the interior of the circumcircle (circle that passes through all three

vertices) of any triangle in the triangulation. Each node of the graph is a cell. Figure 6 is a plot of approximately a hundred nodes/cells sampled from a bigger triangulation of cells of a mouse spinal cord section.

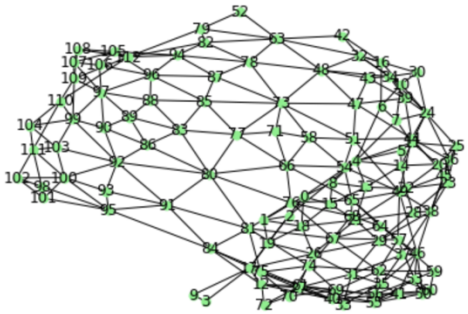


*Figure 6: Sample from cell network developed from Delaunay triangulation*

## RESULTS

Once we have the cell network and each cell's spatial coordinates, we sample cells/nodes from each network and then use the subset to generate Voronoi Regions. A Voronoi diagram of a vertex set is a subdivision of the plane into polygonal regions (some of which may be infinite), where each region is the set of points in the plane that are closer to some input vertex than to any other input vertex. (The Voronoi diagram is the geometric dual of the Delaunay triangulation.) Below is a sample image of a Voronoi tessellation looks:

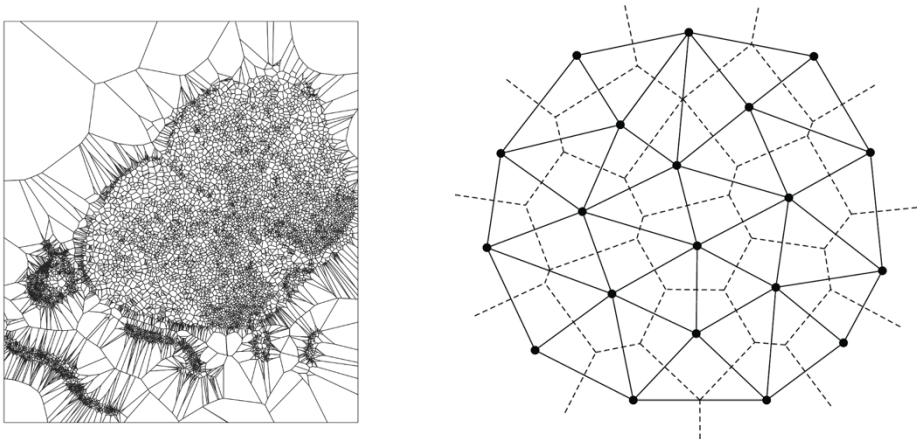


*Figure 7: Voronoi tessellation of the cells of the tissue section (Left) and visual representation of a Delaunay triangulation overlayed on a Voronoi tessellation (Right).*

And finally, we can create spatial neighborhoods using the Voronoi regions. The geometry column denotes the geometry of each spatial neighborhood, and we can see the count occurrence of the cell types within that neighborhood. Hudson also generates an alphanumeric index for each neighborhood which we can see on the leftmost side of the data frame.

| | Astrocytes | Cholinergic neurons | Endothelial cells | Excitatory neurons | Inhibitory neurons | Microglia/Macrophages | Oligodendrocytes | geometry |
|---|---|---|---|---|---|---|---|---|
| XUNCMB | 2 | 0 | 0 | 0 | 0 | 0 | 2 | POLYGON ((1399.000 1402.024, 861.333 1405.620,... |
| QXLBYZ | 2 | 0 | 1 | 0 | 1 | 0 | 0 | POLYGON ((7292.880 -4604.804, 1364.155 1353.80... |
| 2HIG1N | 3 | 0 | 2 | 0 | 2 | 0 | 1 | POLYGON ((861.333 1405.620, 1399.000 1402.024,... |

*Figure 8: First few rows of the spatial neighborhood/microenvironment data frame*

## Component Analysis:

As the pipeline had various components/rules, it required a collection of scientific computing libraries and frameworks. Below is a visual representation of the external libraries used:


*Figure 9: A visual representation of the component wise workflow*

## Conclusion

Therefore, we can see that Hudson is well able to segment the image, identify the different cell types in the image and assign each cell detected to a certain cell type. It then condenses all the detected information into a compressed, shareable data structure. Following the compression, it creates a cell network and samples from the network to build several spatial neighborhoods/microenvironments over the tissue section being studied and records the different cell type compositions of each spatial neighborhood. This information can then be used in downstream analysis for example, building a detection tool for cancerous vs non-cancerous tissue sections as the cell type compositions should differ.

## References


- Seo, J. et al. PICASSO allows ultra-multiplexed fluorescence imaging of spatially overlapping proteins without reference spectra measurements. Nat Commun 13, 2475 (2022).
- Belghazi, M. I. et al. MINE: Mutual Information Neural Estimation. arXiv:1801.04062 [cs, stat] (2018).
- Pandit K. & Petrescu, J. et al DOI: 10.1038/s41598-022-08740-w (2022)
- Stringer, C. & Pachitariu, M DOI: 2022.04.01.486764 (2022)
- Hao Y & Hao S, et al DOI: 10.1016/j.cell.2021.04.048
- Uwitonze A, et al DOI: 10.1371/journal.pone.0193350