# SIB days tutorial

# Analysis of spatial transcriptomics data part 2

**Heidi Tschanz-Lischer, Staff Scientist**

# Overview

1. Integration
2. Identifying clusters
3. Identify marker genes for each cluster
4. Cell type identification
5. Identify spatial variable features
6. Cell type deconvolution
7. Neighbourhood analysis
8. HD Visium 10x Genomics

Covered in exercises

# Integration

- Central callenge in most scRNA-seq and spatial transcriptomcs: batch effects
- Technical variability
  - Differences in sample handling or quality
  - Sequencing
- Biological variability
  - Donor variation
  - Sampling location

- Removal of batch effects is essential for joint analysis

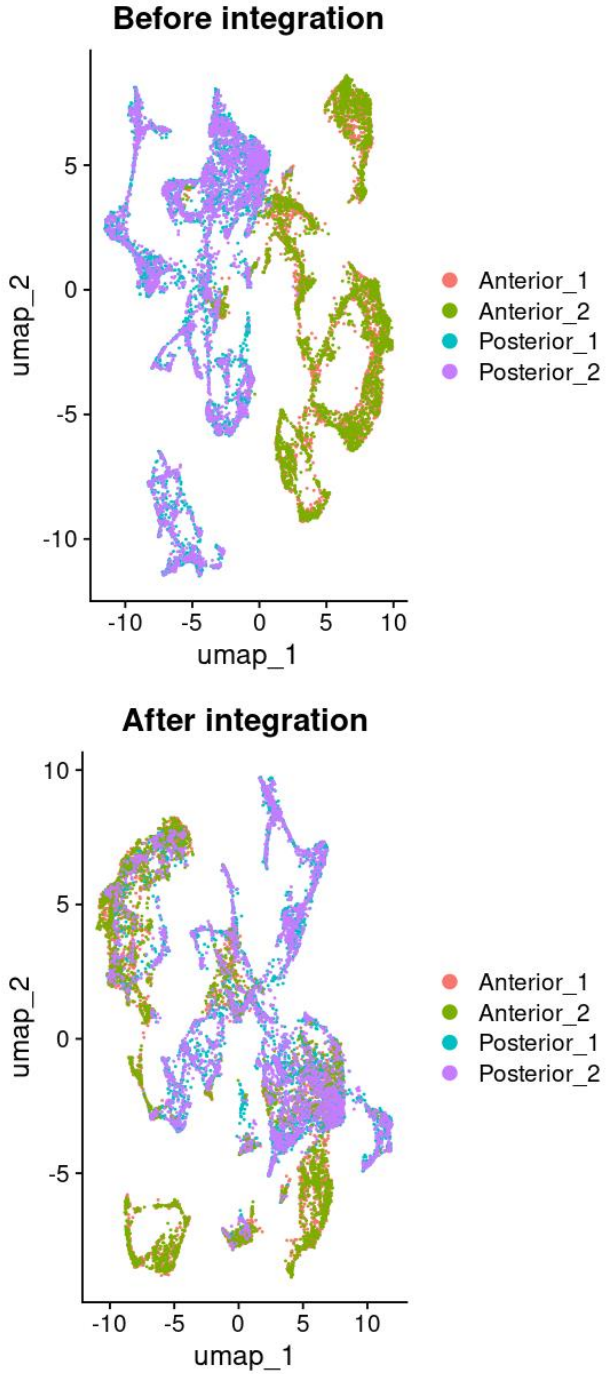
  → we want to identify cell types which are present in all samples

# Integration

Main principle of data integration:

- Find corresponding cells across datasets
- Compute a data adjustment based on correspondances between cells
- Apply adjustment

→ Not always needed!



# Integration

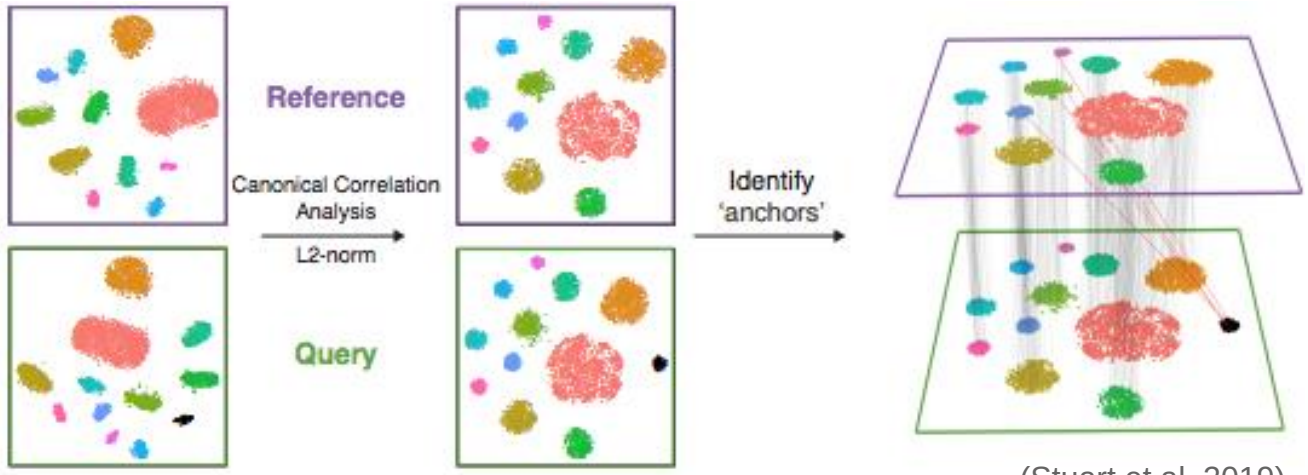
Methods can be divided in 4 categories:

- Global models:
  - Originated from bulk
  - Model the batch effect as an consistent effect accross all cells
  - Not optimal for scRNA-seq or spatial transcriptomic data

  - E.g. ComBat (Johnson et al. 2007)

# Integration

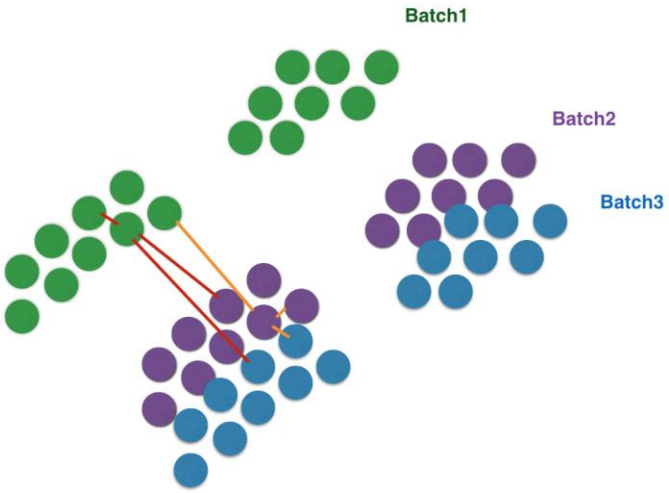
- Linear embedding models:
  - First single-cell-specific batch removal
  - Use variant singular value decomposition to embed the data
  - Then look for local neighborhoods of similar cells across batches («anchors»)
  - Correct batch effect in a locally adaptive manner




(Stuart et al. 2019)

- E.g. mutual nearest neighbors (MNN) method (Haghverdi et al. 2018), Seurat (CCA/RPCA + anchors) (Butler et al. 2018, Stuart et al. 2019), Scanorama (Hie et al. 2019), FastMNN (Haghverdi et al. 2018), Harmony (Korsunsky et al. 2019)

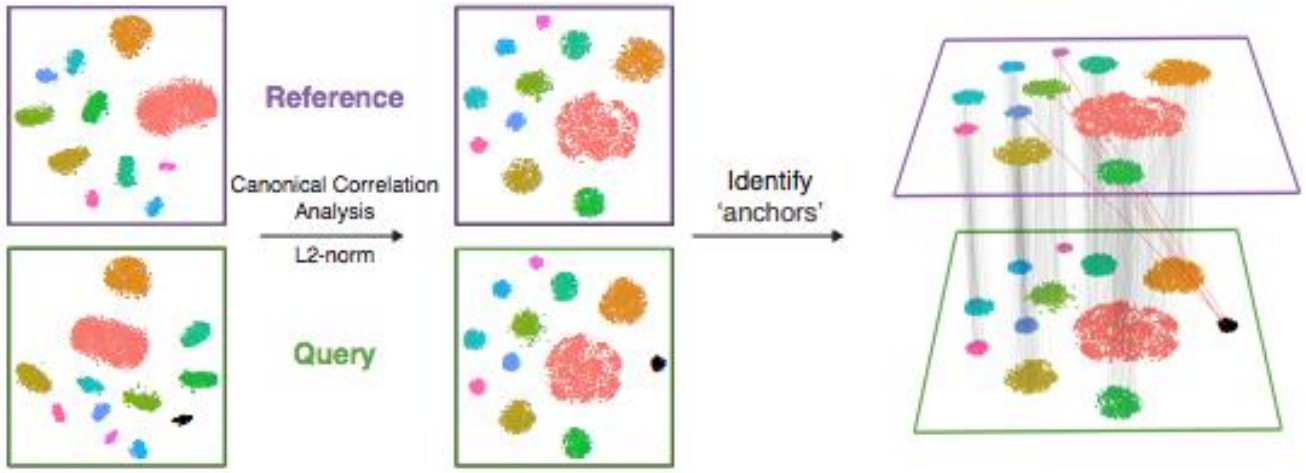
# Integration




- Graph-based methods:
  - fastest methods
  - Use nearest-neighbour graph to represent data from each batch
  - Then enforce graph connections between different batches
  - E.g. Batch-Balanced k-Nearest Neighbour (BBKNN) (Polanski et al. 2019)

- Deep learning approaches:
  - Require most data for good performance
  - Mostly based on autodecoder networks
  - E.g. scVI (Lopez et al. 2018)
    scANVI (Xu et al. 2021)
    scGen (Lotfollahi et al. 2019)

# Integration
## Seurat

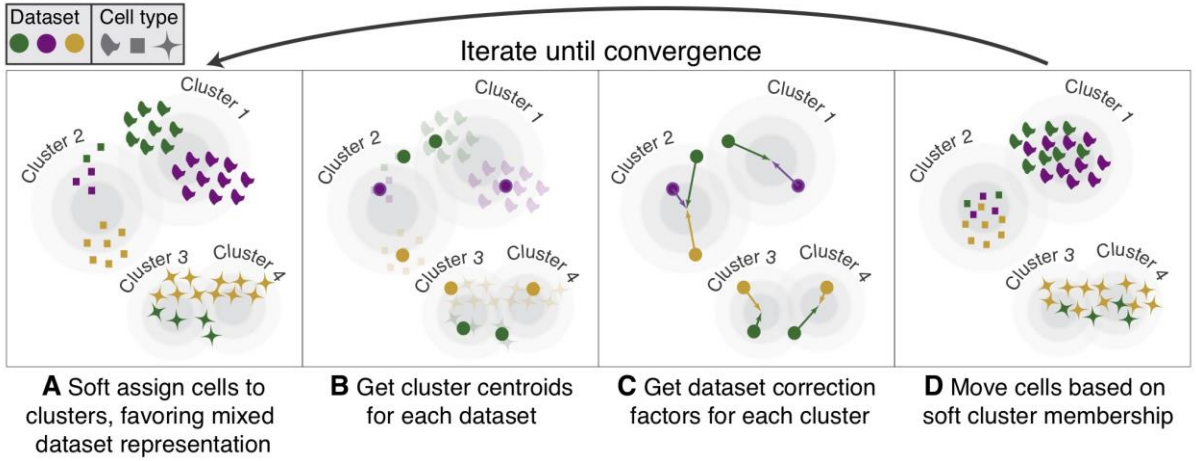



Available integration methods:

- CCA + anchors:
  - Cell types are conserved, but different in gene expression
  - If experimental conditions introduce very strong expression shifts
  - Analysis across species
  - May lead to overcorrection if large proportion of cells are non-overlapping

- RPCA + anchors:
  - Faster
  - More conservative
  - If substantial fraction of cells in one dataset have no matching type in the other

# Integration
## Seurat




- Harmony:
  - Applies a transformation to the principal component (PCs) values
  - Fast and lower memory requirements

- FastMNN:
  - Fast version of the mutual nearest neighbours (MNN) method
  - Applies PCA to reduce dimensionaliy

- scVI:
  - Deep learning approach
  - Scalable to very large datasets (>1 million cells)

# Integration
## Seurat

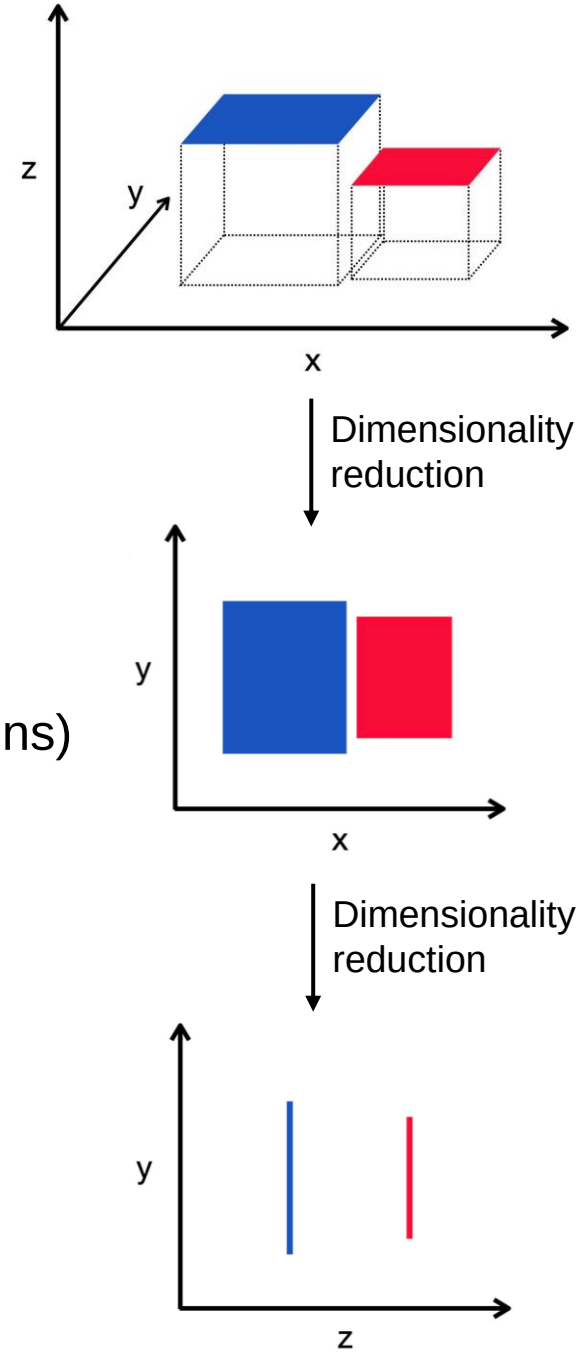
```
IntegrateLayers(object = data,
                method = RPCAIntegration,
                normalization.method = "SCT",
                new.reduction = "integrated.rpca")
```

→ Check if integration is needed

→ If yes, run several methods and compare

# Dimensionality reduction

- “Remove” redundancies in the data
- Identify the most relevant information (find and filter noise)
- Simplify complexity
  - Easier to work with
  - Reduce computational time for downstream procedures
  - Facilitate clustering (some algorithms struggle with too many dimensions)
- Data visualization
- Most common used:
  - PCA: Principal Component Analysis
  - TSNE: T-distributed stochastic neighbourhood embedding
  - UMAP: Uniform manifold approach and projection



# Dimensionality reduction

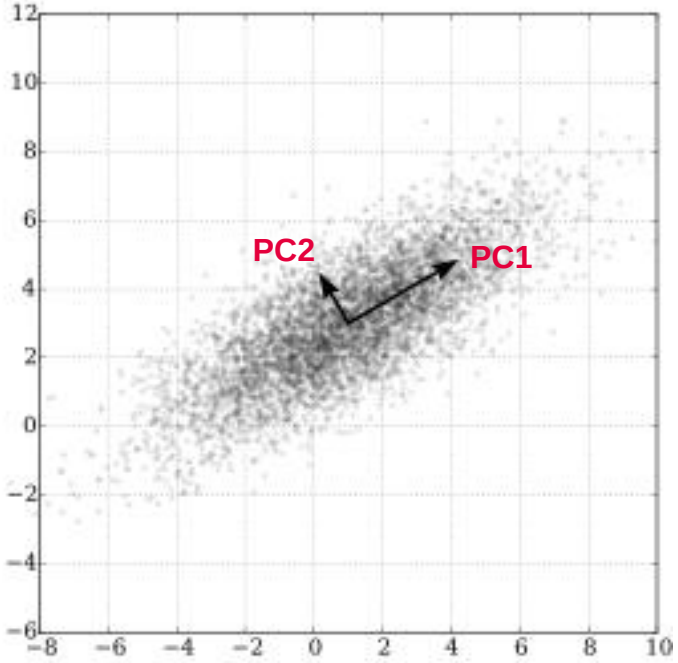
Don’ts:

- They are not perfect representation of the high dimension
- One does loose information
- What is close in the projection might actually be far
- What is far might actually be close
- Conclusions (specially biologically relevant conclusions) should NOT be based on the dimensionality reduction

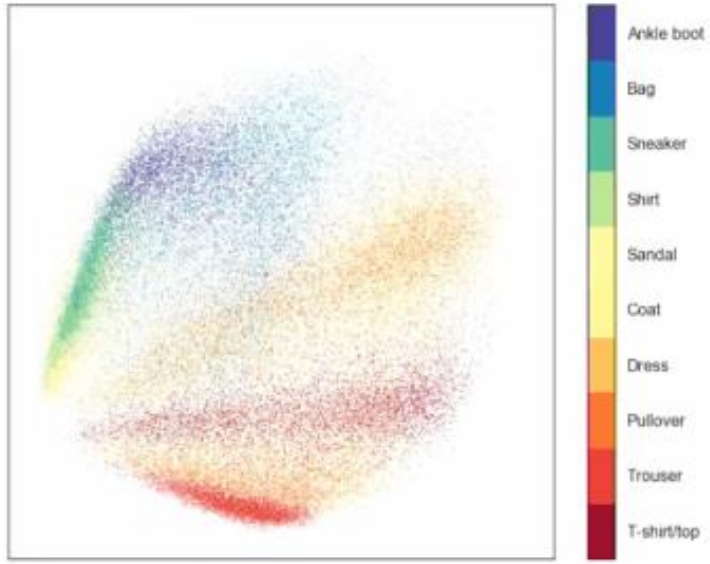
# Dimensionality reduction

## PCA

- Based on variance
- Data is linearly transformed onto a new coordinate system such that the principal components capture the variations
- Largest variance first
  - The top principal components contain higher variance from the data
  - Can be used as filtering by selecting only the top significant PCs
- Data is usually scaled prior to PCA
  → if one variable is on a different scale, it will dominate the PCA procedure
- Problems:
  - First two PC often account only for a few percent of the total variance
  - It performs poorly to separate cells in 0 inflated data types (non-linear)
- Seurat: **`RunPCA(data, npcs = 50)`**

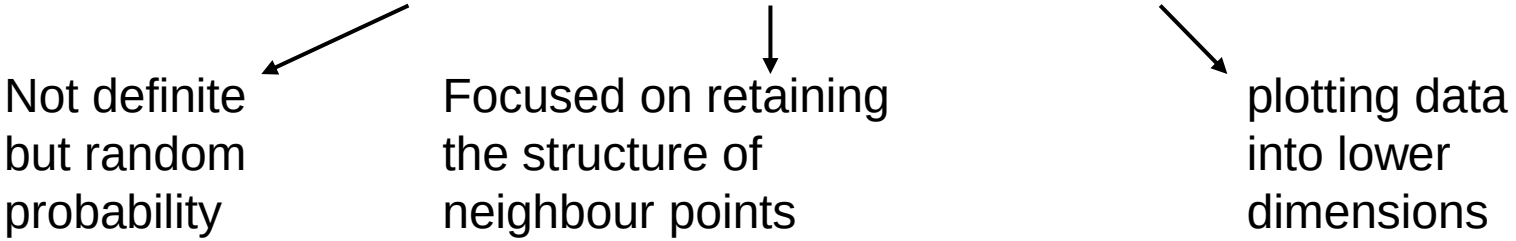



PCA – Fashion MNIST



# Dimensionality reduction
## T-SNE

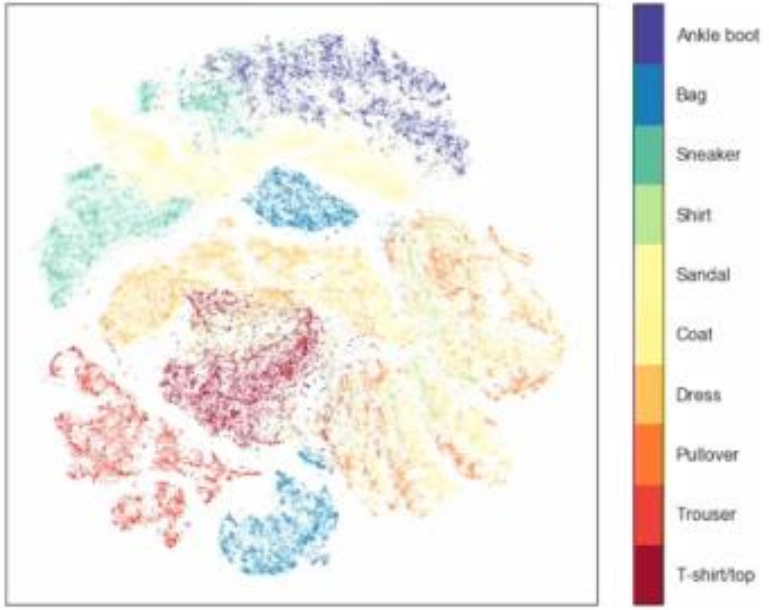
T-SNE = t-distributed stochastic neighbourhood embedding




- Developed by Laurens van der Maaten and Geoffrey Hinton in 2008
- Machine learning algorithm
- Nonlinear dimensionality reduction
- Statistical method for visualizing high-dimensional data by giving each datapoint a location in a two or three-dimensional map
- Seurat: **`RunTSNE(data, reduction = "pca")`**

# Dimensionality reduction
## t-SNE

T-SNE – Fashion MNIST




Problems:

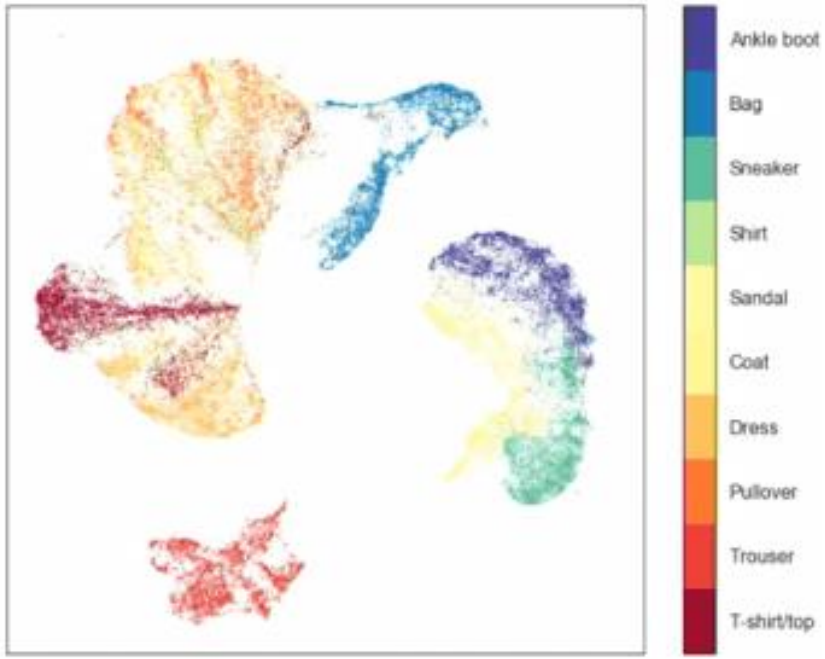
- not a very quantitative method:
  - t-SNE axes don’t really mean anything
  - Axes are just the distribution along which t-SNE has clustered your data so that they are separated → just for visualization
- Doesn’t preserve the global structure of the data
  - Two data points being in one cluster tells us something about their high-dimensional similarity
  - But distance between two clusters doesn't really tell us anything about the inter-cluster similarity
- Stochastic algorithm → generates slightly different results each time
- Non-parametric → cannot add samples to a preexisting t-SNE → need to rerun
- Computationally very intensive algorithm

# Dimensionality reduction
## UMAP

UMAP: Uniform Manifold Approximation and Projection

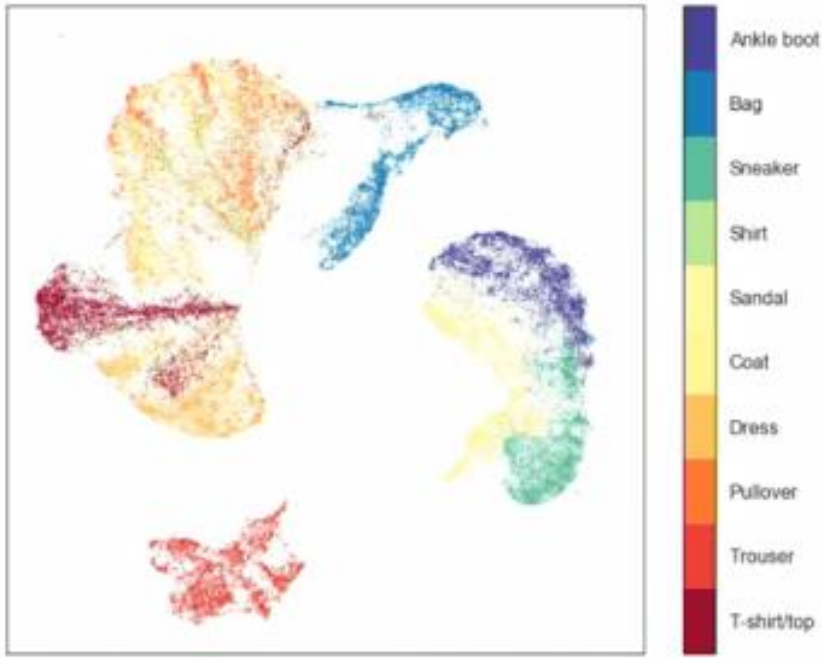
- Leland McInnes, John Healy and James Melville in 2018
- Non-linear graph-based method of dimensionality reduction
- Very similarly to t-SNE
- Defines both local and global distances
- Tends to better preserve the global structure of the data
- No stochastic part → same result every time
- Can be applied to new data points
- Seurat: **`RunUMAP(data, reduction = "pca")`**

UMAP – Fashion MNIST



# Dimensionality reduction
## UMAP

UMAP – Fashion MNIST

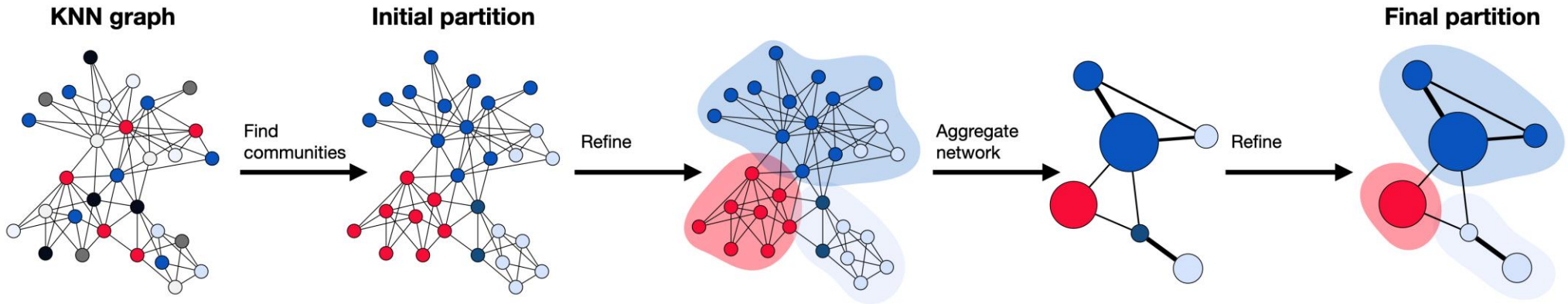



Problems:

- Interpretability of axis is lacking
- Size of clusters relative to each other is essentially meaningless
- Distances between clusters is likely to be meaningless
  - Global positions of clusters are better preserved in UMAP
  - But the distances between them are not meaningful
- UMAP tends to find manifold structure within data noise
  - The larger the dataset, the less noise
  - UMAP is recommended for big datasets but not small once

# Identifying clusters

- Next natural step is is the identification of cellular structure in the dataset
- Group cells with similar properties
- Identify different cell states / cell types
- Graph based methods:
  - First calculate a Euclidean distance matrix on the PC-reduced expression space for all cells
  - Connect each cell to its K most similar cells (KNN-graph, Wolf et al. 2019)
  - Dense regions are detected by methods like Leiden and Louvain (Blondel et al. 2008)
  - Iterative process (moves single nodes from one community to another to find a partition)



# Identifying clusters

Seurat:

- Graph-based clustering
- Clustering is based on PCA based on variable genes (could also be set to spatial variable)
- «resolution»: Granularity of clustering
  - important argument → need to be optimized for every experiment (often between 0.2 – 1.8)
  - Higher resolution → more clusters
- ```
  integrated <- FindNeighbors(integrated, reduction="pca", dims=1:50)
  integrated <- FindClusters(object=integrated, resolution=seq(0.2,1.8,0.2)
  ```

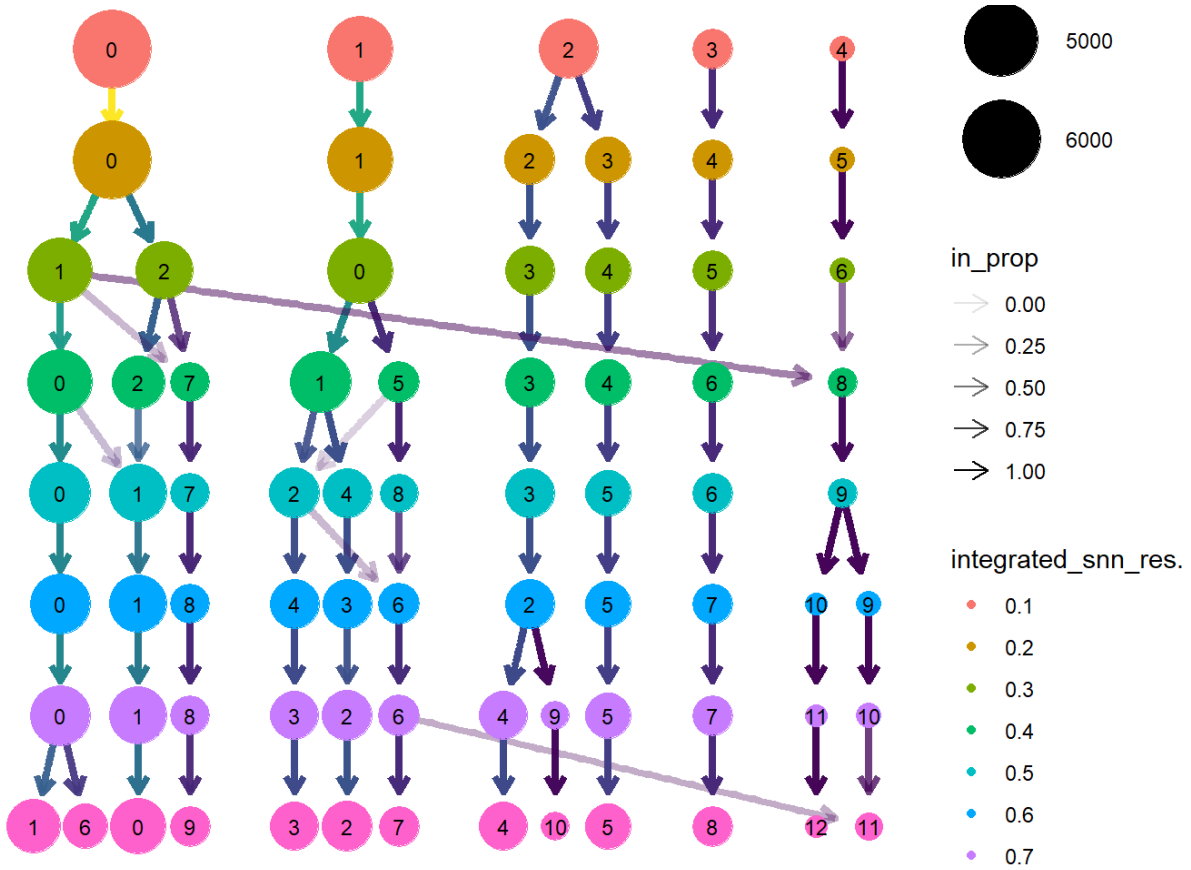
Challenges:

- Number of clusters
- What is a cell type
- Clustering is subjective → not ground truth
- Stability of clusters
  ```

# Identifying clusters

clustree (Zappia et al. 2018):

- Deciding what resolution to use can be a difficult question
- One way to approach this problem is to look at how cells move as the resolution increases
- Nodes: clusters
- Edges:
  cells in a cluster at a lower resolution that end up in a cluster at the next higher resolution
- Nodes with multiple incoming edges
  → good indication that data is over-clustered



# Identify marker genes for each cluster

Two types of gene expression analysis:

- Marker gene identification:
  - Genes overexpressed by clusters
  - Identify representative marker genes for each cluster
  - Can help in cell type annotation
  - Surat: **`FindAllMarkers(integrated, only.pos = TRUE)`**
    - Default: Wilcoxon test
    - Finds genes that are different between one cluster and all other cells

# Identify marker genes for each cluster

- Differential gene expression analysis:
  - Genes impacted by experimental conditions within a cluster
  - Seurat:
    **`FindMarkers(data, ident.1 = cluster1, ident.2 = cluster2)`**
    - Default: Wilcoxon test
    - Pairwise comparison between cluster 1 and cluster 2
  - More complex designs (factorial design,..) use limma or edgeR

# Cell type identification

## What is a cell type?

- Fundamental unit of life
- Originally defined in terms of function, location tissue type, cell morphology
- Later extended to
  - presence/absence of cell surface markers
  - gene expression (molecular profile)
- Currently very much less fixed
  - cell cycle phase
  - migration state
  - differentiation: cell state

# Cell type identification

Why should we identify cell types?

- To determine which cell types might communicate with each other
- To compare the abundance of cell types in different conditions
- Find new cell types which have been missed by using “standard” surface markers

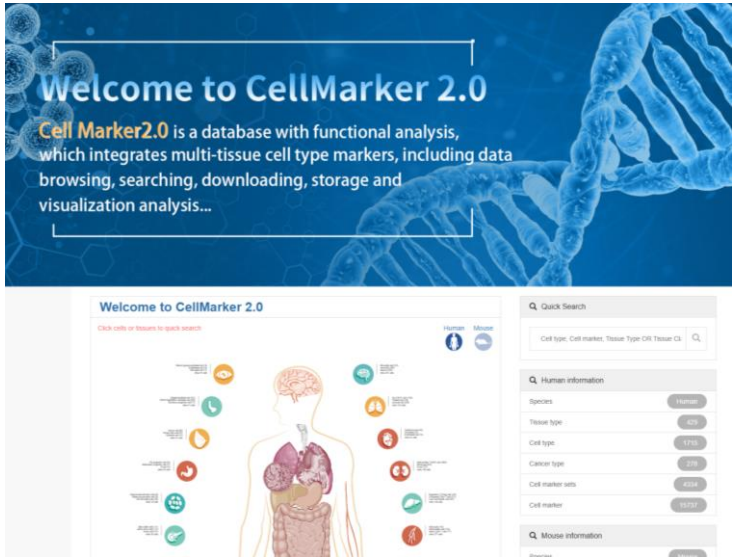
# Cell type identification
## Manual annotation

- Data often clustered before annotation
  - annotate group of cells
  - more robust to noise: cell might not have a count for marker gene (only sequence a small subset of the total amount of RNA)
- Based on known marker genes
  - Identify marker genes per cluster
  - link those to known biology (cell types/states)
- Time consuming
- Requires expert knowledge
- Sometimes subjective and inaccurate

# Cell type identification
## Automatic annotation

- Problem: requires a reference for given species / tissue
  - pre-defined sets of markers
  - pre-existing full scRNA-seq datasets
  - Can miss cell types if they are not included in the reference
- Not dependent on a partitioning of the data into clusters
- Uncertainty measures improve quality and usability of method
  - Uncertain annotation → can highlight unseen cell types or states
- Resulting annotation can be of varying quality
  - it's a start-point rather than an end-point of annotation process
- Methods:
  - Assign a cell type per individual cell or per cluster of cells (better per cell)
  - Assignment of cell type via correlation of each cell/cluster to the "reference"

# Cell type identification
## Reference databases




- PanglaoDB (https://panglaodb.se; Franzen et al. 2019)
  - mouse and human
  - https://cran.r project.org/web/packages/rPanglaoDB/index.html
- CellMarker 2.0 (http://bio-bigdata.hrbmu.edu.cn/CellMarker/) (Hu et al. 2022)
  - mouse and human
- SingleR (https://bioconductor.org/packages/release/bioc/html/SingleR.html) (Aran et al. 2019)
  - access via celldex package
- Human Cell Atlas (https://www.humancellatlas.org) (Regev et al.)
  - scRNA-seq atlas
  - also some mouse data
- Single cell portal: (https://singlecell.broadinstitute.org/single_cell)

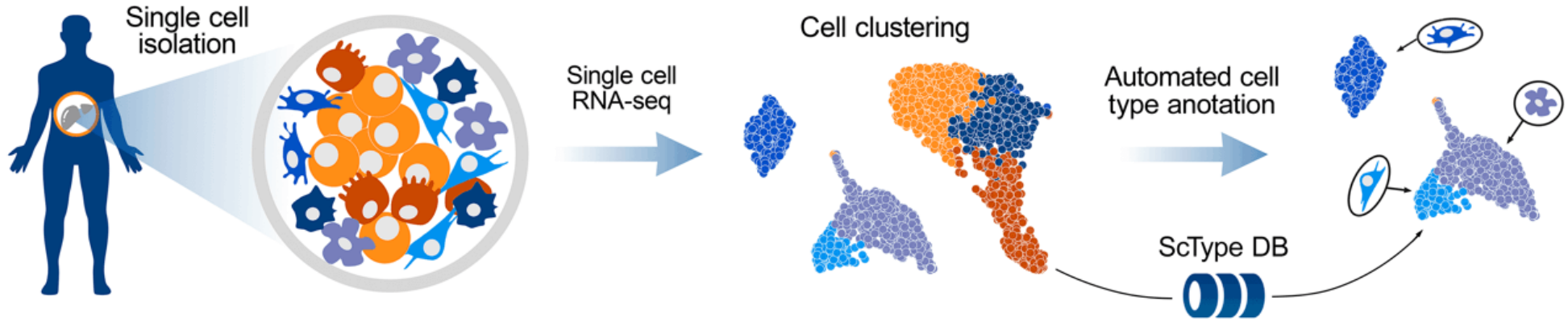
HUMAN CELL ATLAS

# Cell type identification
## Methods

- CellTypist (https://www.celltypist.org) (Conde et al. 2022):
  - Upload portal
  - Python script
- Ucell (https://github.com/carmonalab/UCell) (Andreatta and Carmona 2021):
  - R package for scoring gene signatures based on Mann-Whitney U statistic
- SingleR (https://www.bioconductor.org/packages/release/bioc/html/SingleR.html) (Aran et al. 2019)
  - R package for automatic annotation based on reference dataset
- Symphony (https://github.com/immunogenomics/symphony) (Kang et al. 2021)
  - R package for efficient and precise single-cell reference atlas mapping

# Cell type identification
## Methods

- Seurat:
  - probabilistic transfer of annotations from a reference scRNA-seq to a query set
  - get prediction scores for each spot for each class
  - **`FindTransferAnchors()`** and **`TransferData()`**
- ScType (https://github.com/IanevskiAleksandr/sc-type) (Ianevski et al. 2022)
  - Fully-automated and ultra-fast cell-type identification using specific marker database



# Identify spatial variable features

- One main analysis step for single-cell data is to identify highly-variable genes
  - neglect the spatial context of cells
  - A gene could be highly variable, but not show distinct spatial pattern
- Identify spatial variable genes
  - identify molecular features that correlate with spatial location in the absence of pre-annotation
- Spatial variation could be caused by
  - Cell-type composition
  - Overall functional dependencies
  - Cell-cell communication events

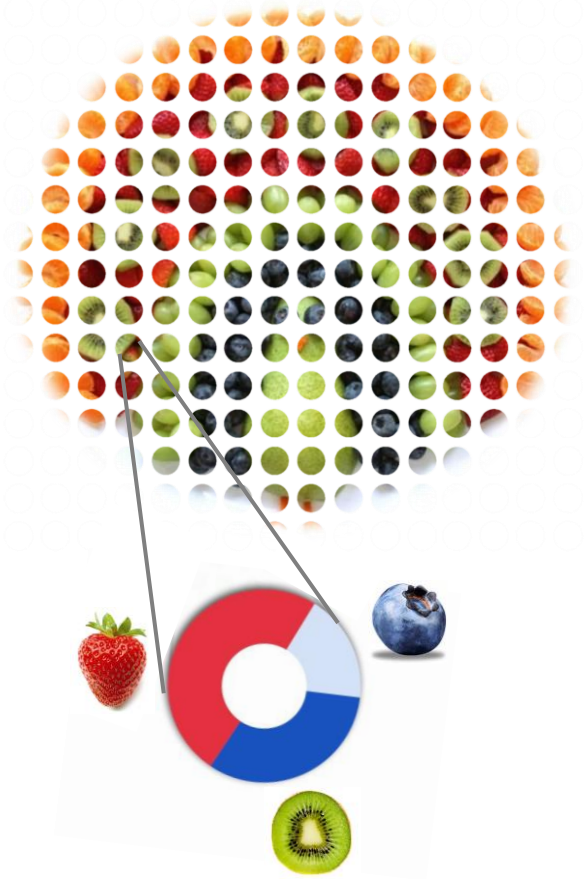
→ Helps to understand underlying tissue biology

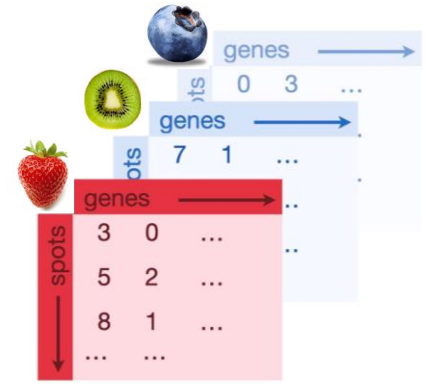
# Identify spatial variable features

- Methods:
  - SpatailDE (Svensson et al. 2018): identify genes which significantly depend on spatial coordinates in non-linear and non-parametric ways
  - SPARK (Sun et al. 2020): based on generalized spatial linear models
  - Trendsceek (Edsgärd et al. 2018): identify genes with spatial expression trends
  - HMRF (Zhu et al. 2018): hidden-Markov random field approach
  - Splotch (Stahl et al. 2016): hierarchical generative probabilistic model
  - Semla (Larsson and Franzen 2023): compute spatial autocorrelation scores, fast
  - Seurat: `FindSpatiallyVariableFeatures(data, assay = "SCT", selection.method = "markvariogram")`
    - Inspired by Trendsceek
    - Calculates gamma(r) values, measuring dependence between two spots a certain “r” distance apart
    - We could also base the clustering on spatial variable genes

# Cell-type deconvolution

- Problem: e.g. Visium 10x Genomics average resolution
  1 to10 cells per spot
  - Like to demix expression profiles back to individual cells
  - Referred to as deconvolution
- Different flavours:
  - Obtain proportions of different cells or cell types per spot
  - Obtain expression profiles of each cell type per spot
    → more complex
- Often depend on scRNA-seq references

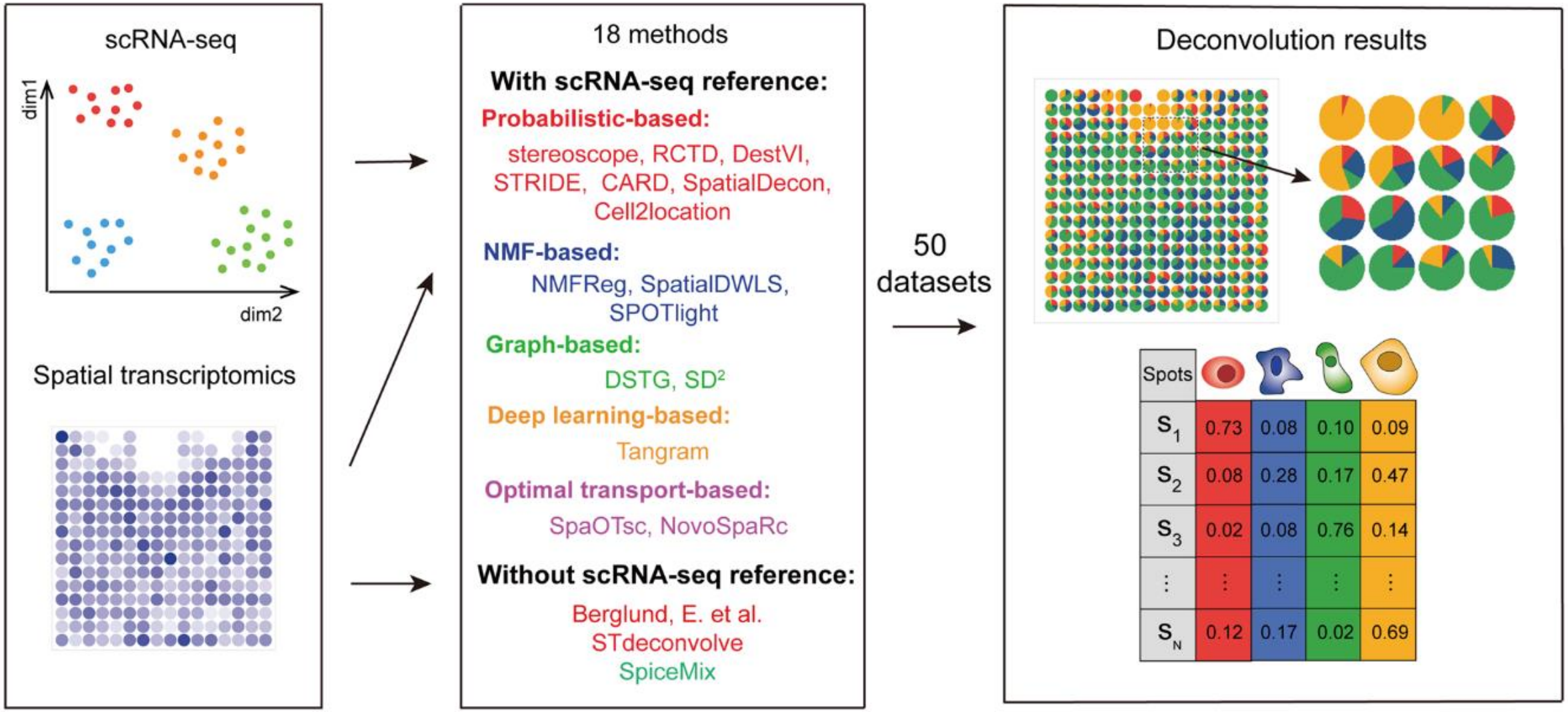




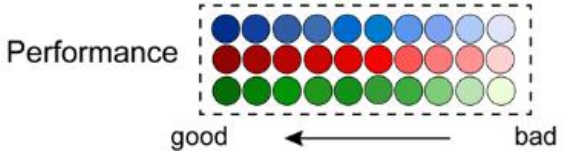

Cell type specific expression

# Cell-type deconvolution
## Methods

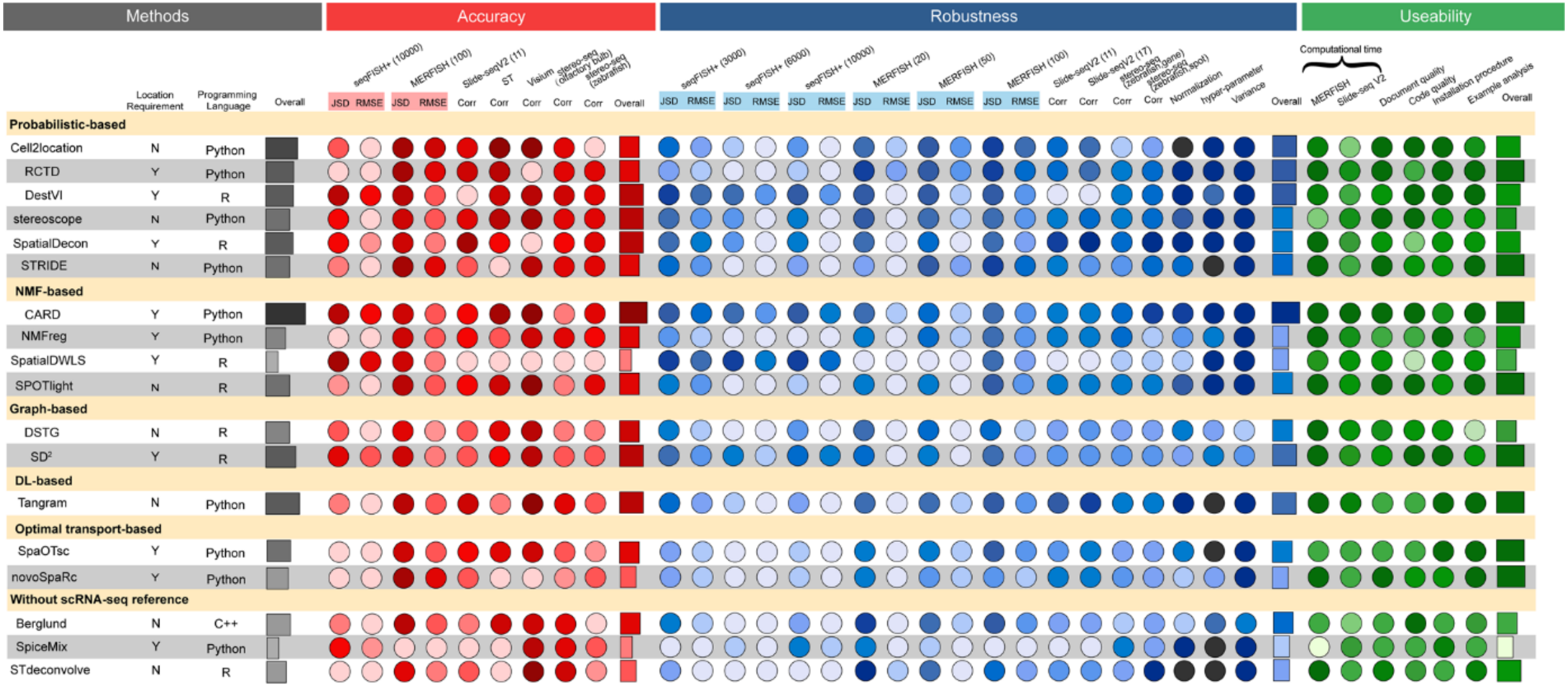
(Li et al. 2023)



# Cell-type deconvolution
## Methods




(Li et al. 2023)



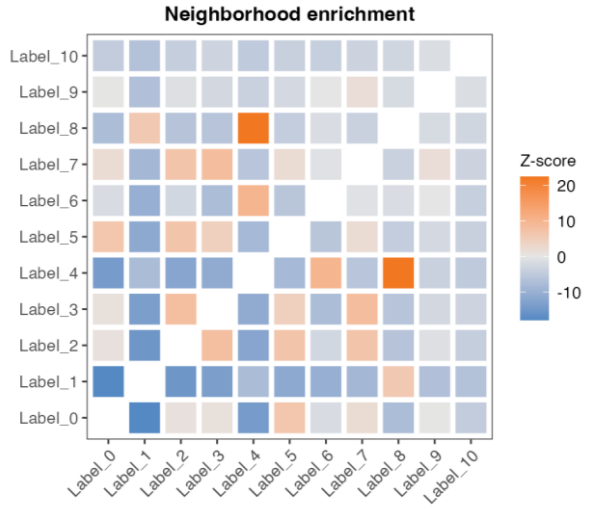
# Neighbourhood analysis

- After annotation we can analyze cellular neighbourhoods across the tissue
- Helps to understand the cellular composition of the tissue
- Identify candidates for more in-depth analysis:
  - Candidates for cell-cell communication
  - Interactions between spatial communities

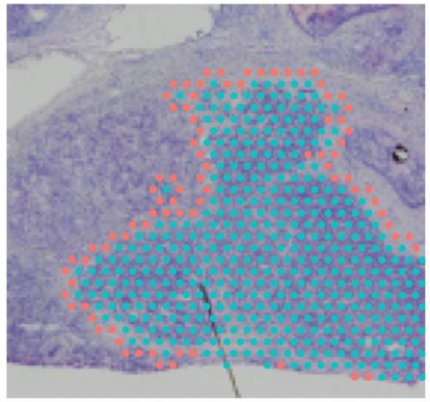
Methods:

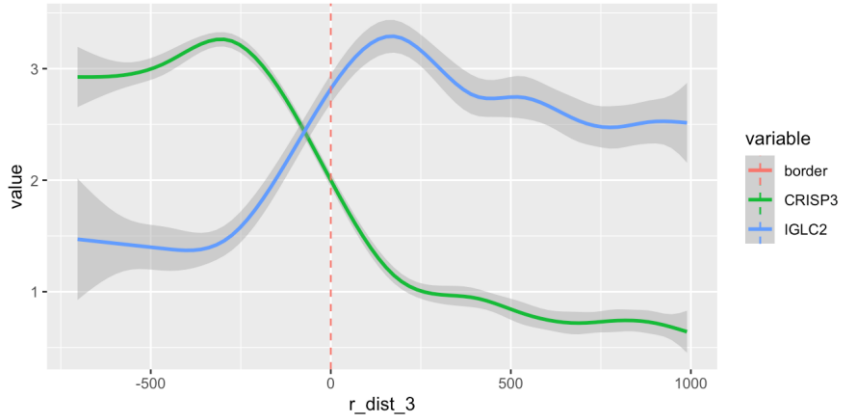
- Squidpy (https://squidpy.readthedocs.io/en/stable/) (Palla et al. 2022)
- Semla (https://ludvigla.github.io/semla/) (Larsson and Franzen 2023)

# Neighbourhood analysis
## Semla



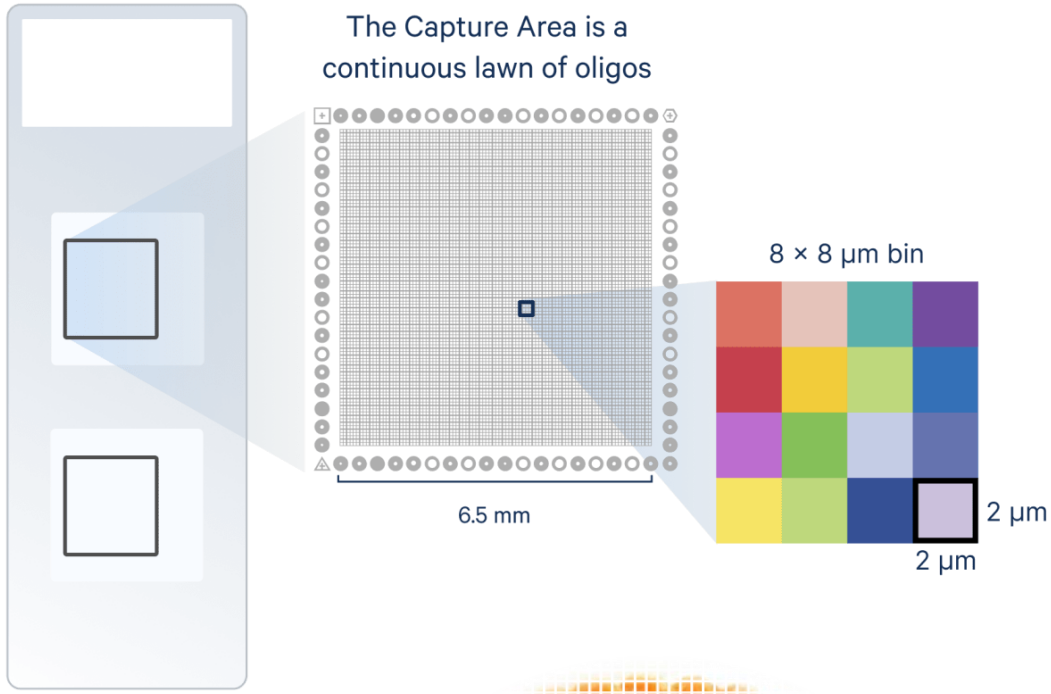

- Estimate a neighbourhood enrichment score:
  - determines if cells belonging to two different clusters are close to each other more often than expected
- Identify cells at borders of annotations:
  - Differential expression test between the outer and inner borders (e.g. a tumor)
  - Allows to characterize microenvironment around region of interest
- Explore expression of genes as a function of distance from a region of interest

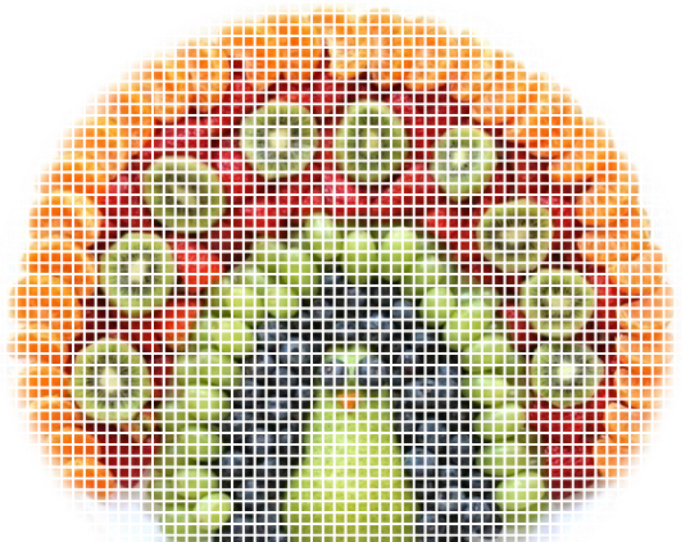




# HD Visium 10x Genomics

- FFPE samples
- two 6.5 x 6.5 mm Capture Areas
- Continuous lawn of oligonucleotides:
  - arrayed in millions of 2 x 2 µm barcoded squares
  - without gaps
    → achieving single cell–scale resolution
  - data output in multiple bin sizes
    → 8 x 8 µm bin is recommended starting point



# HD Visium 10x Genomics

Seurat support both Visium and Visium HD data

- `Load10X_Spatial(data.dir = localdir, bin.size = c(8, 16))`
- `DefaultAssay(object) <- "Spatial.008um"`
- Seurat v5 sketch clustering workflow recommended for Visium HD
  - aim to ‘subsample’ large datasets in a way that preserves rare populations
  - then project the cluster labels back to the full dataset
  - exhibits improved performance, especially for identifying rare and spatially restricted groups
- Identifying spatially-defined tissue domains:
  - incorporating neighbourhood information for clustering
  - based on BANKSY (Singhal et al. 2024)

# Contact

Dr.

**Heidi Tschanz-Lischer**

Staff Scientist

heidi.tschanz-lischer@unibe.ch

+41 31 684 46 40