Single Cell RNAseq
Data Analysis with Scanpy

Oxford Biomedical Data Science Training Programme

Overview

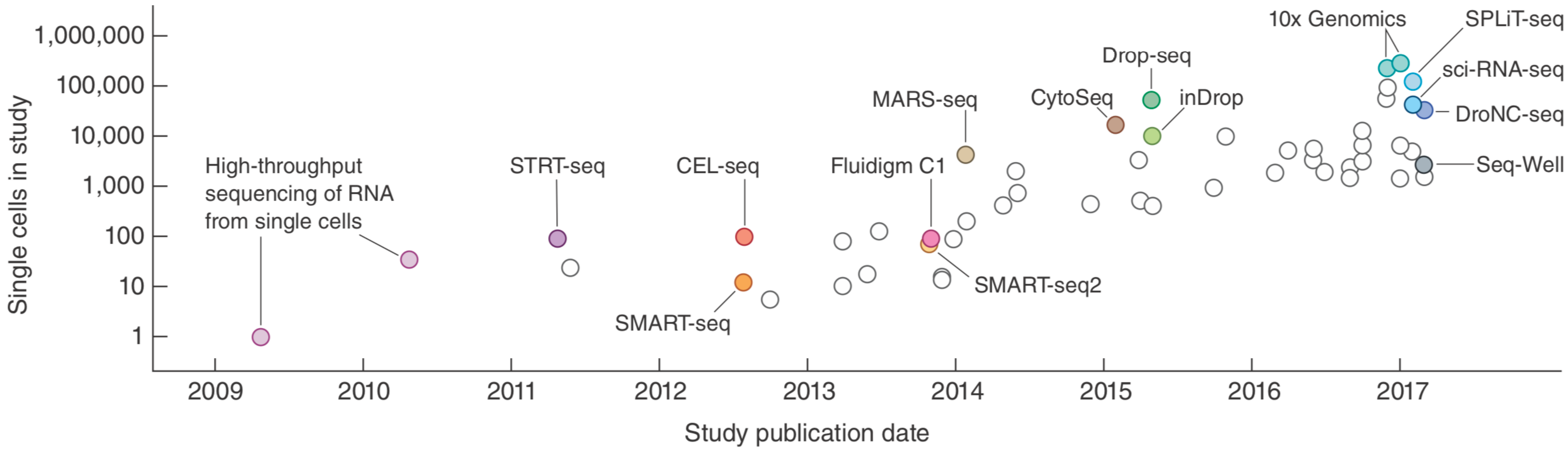
Brief intro to:

- What is single cell?, What are the main techniques?
- How does 10X assay work?
- What happens before you get to scanpy?

Scanpy:

- What and why?
- Anndata object and how to navigate.
- Standard analysis steps and why

What is single cell?

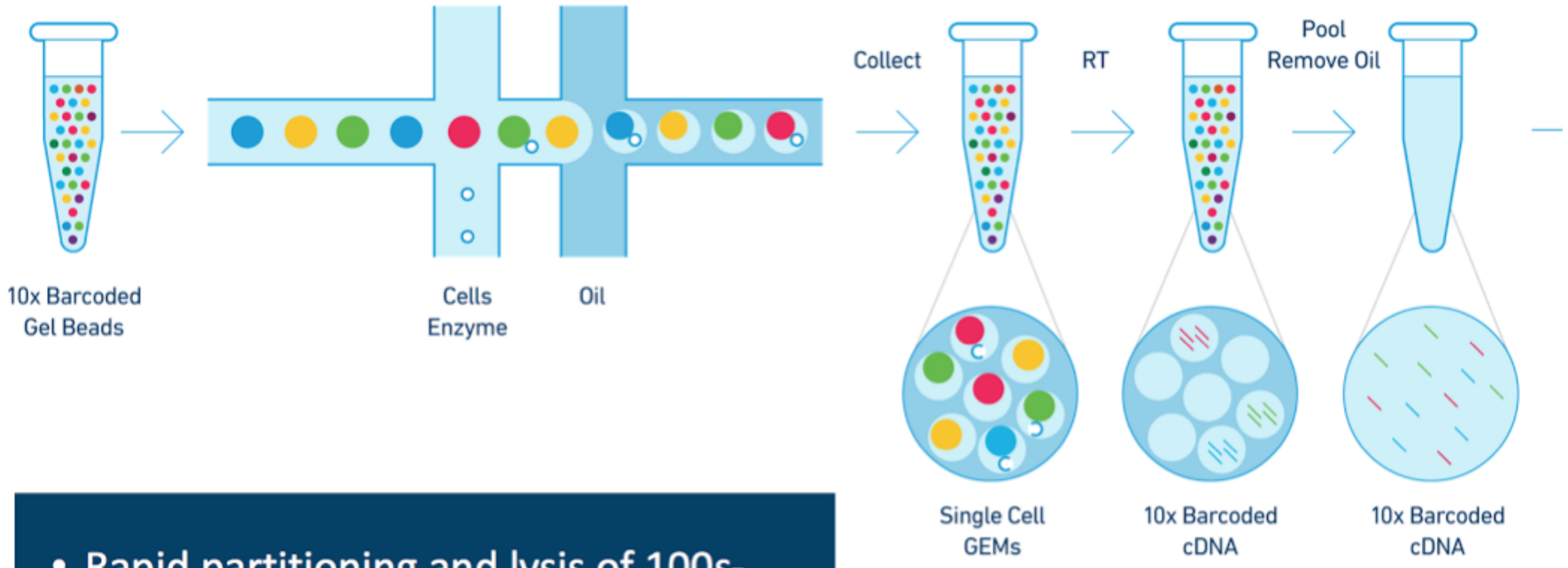


Smart-seq2 vs 10X

- Read depth or lots of cells??
- More cells → more resolution?

- 10X has shiny new assays

How does 10X work?



- Rapid partitioning and lysis of 100s-10,000s of cells in < 7 minutes
- Low cell loss
- No lower limit on cell size

Initial steps (not covered today)

- Cellranger or custom pipeline.
  - Illumina basecalls -- > fastq files (e.g. cellranger mkfastq of bcl2fastq)
  - Alignment,
  - filtering
  - barcode counting
  - UMI counting

	Cell 1	Cell 2	…	Cell N
Gene 1	0	0	..	0
Gene 2	0	10	…	3
…	…	…	…	…
Gene N	0	0	…	0

Analysis of single cell data

Regardless of software the same workflow still applies:

1. Preprocessing: Filtering out dead cells and junk, batch correction if necessary.
2. Normalisation
3. Identification of highly variable genes (HVGs)
4. Dimension reduction
5. Clustering
6. Finding marker genes for clusters

This is frequently an iterative process.

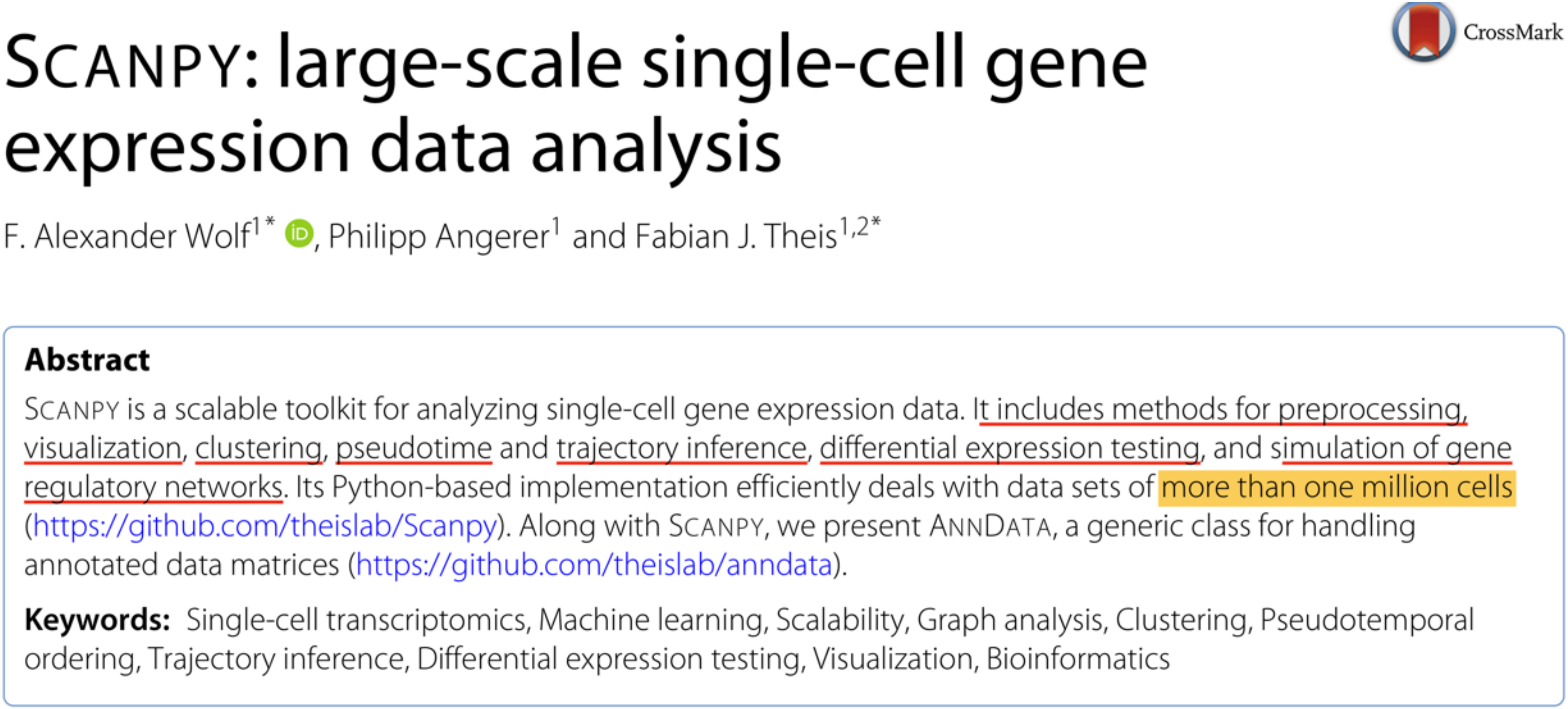
Three main frameworks for single cell analysis

R:

- Seurat
- SingleCellExperiment/scater

Python:

- **Scanpy**



Scanpy: large-scale single-cell gene expression data analysis

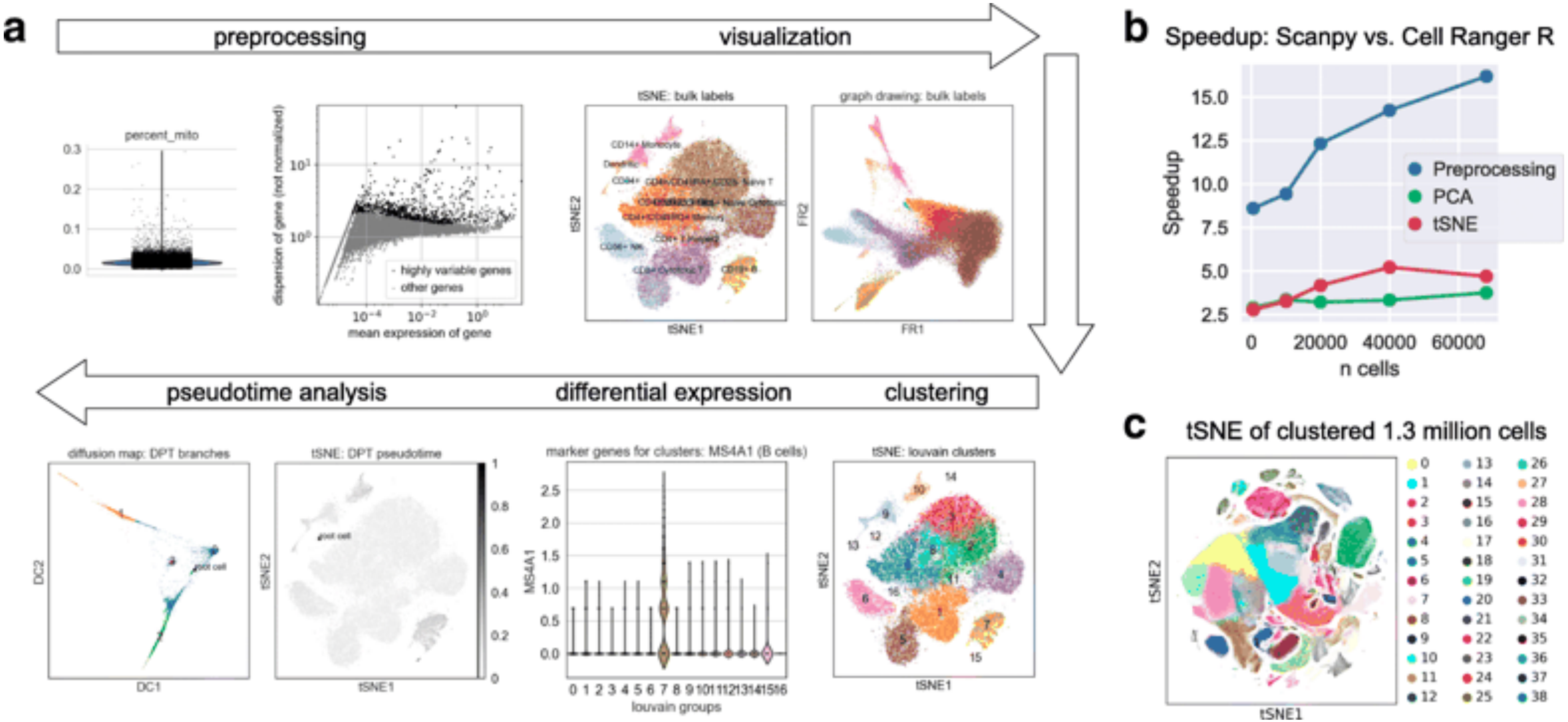
F. Alexander Wolf[1*], Philipp Angerer[1] and Fabian J. Theis[1,2*]

Abstract

Scanpy is a scalable toolkit for analyzing single-cell gene expression data. It includes methods for preprocessing, visualization, clustering, pseudotime and trajectory inference, differential expression testing, and simulation of gene regulatory networks. Its Python-based implementation efficiently deals with data sets of more than one million cells (https://github.com/theislab/Scanpy). Along with Scanpy, we present AnnData, a generic class for handling annotated data matrices (https://github.com/theislab/anndata).

Keywords: Single-cell transcriptomics, Machine learning, Scalability, Graph analysis, Clustering, Pseudotemporal ordering, Trajectory inference, Differential expression testing, Visualization, Bioinformatics

Scanpy integrates canonical analysis methods in a scalable way



Scanpy: Compares well with existing tools

- Was compared existing tools using a number of published datasets:
  - 5-90 times faster compared to Seurat
  - Illustrated using a dataset with 1.3 million cells

- It makes it easy to incorporate advanced machine learning techniques from either scikit-learn or Tensorflow

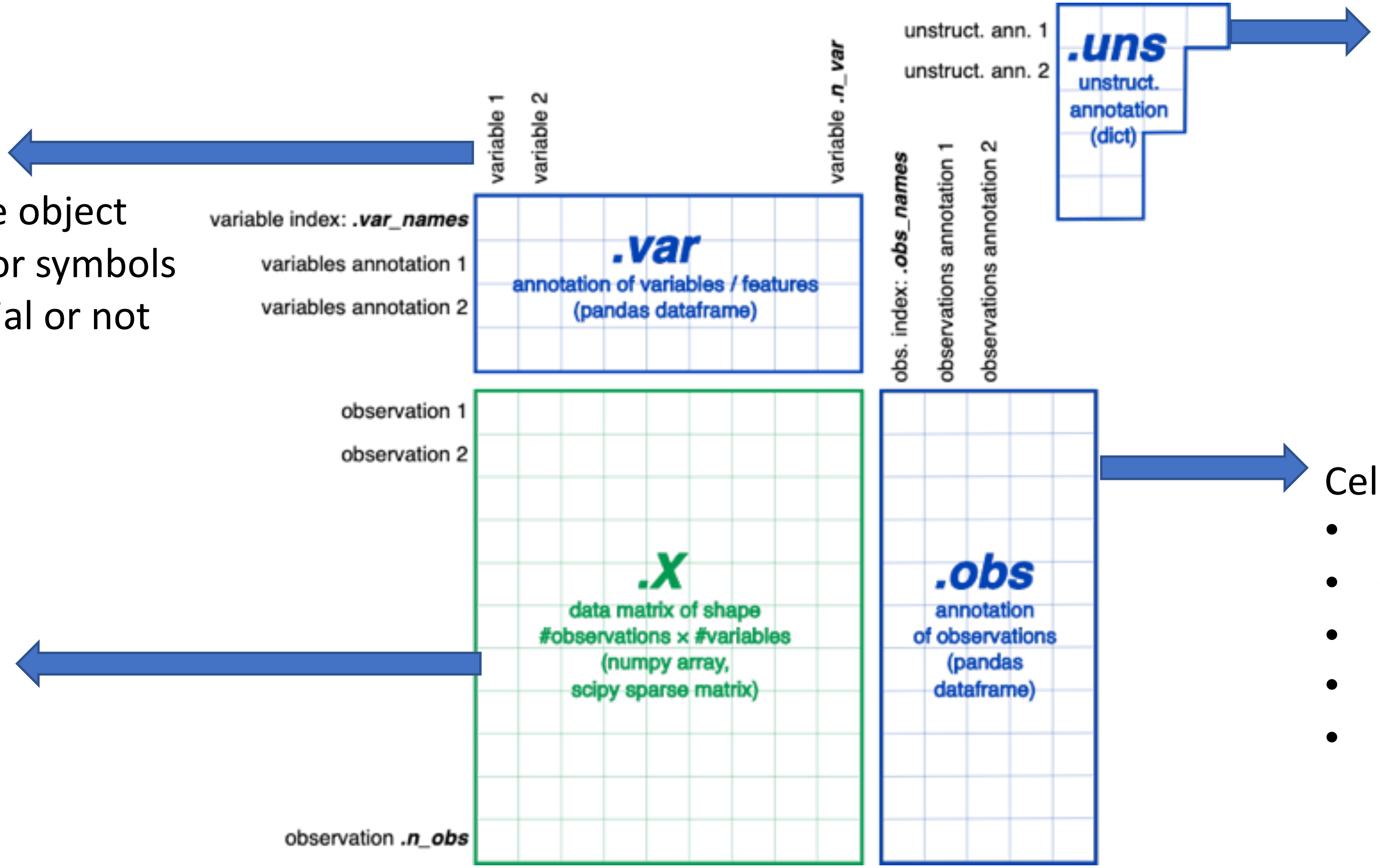
AnnData class

- Scanpy introduces efficient modular implementation choices
- It is built on ANNDATA class:
  - Stores a data matrix X
  - Cells metadata information as ‘Observed’
  - Gene metadata information as ‘Variable’

AnnData class

Gene metadata:

- Dataframe-like object
- Gene’s name or symbols
- Is mitochondrial or not


unstruct. ann. 1
unstruct. ann. 2
.uns
unstruct. annotation (dict)
variable 1
variable 2
variable .n_var
variable index: .var_names
variables annotation 1
variables annotation 2
.var
annotation of variables / features (pandas dataframe)
obs. index: .obs_names
observations annotation 1
observations annotation 2
observation 1
observation 2
observation .n_obs
.X
data matrix of shape #observations × #variables (numpy array, scipy sparse matrix)
.obs
annotation of observations (pandas dataframe)

Unstructured information:

- Dimension reductions

Stored as dictionary pairs of keys and values.

```
adata.uns.keys()
adata.uns.values()
```

Cell metadata:

- Dataframe-like object
- Cell Ids
- Batch information
- Cell cycle information
- Patient/sample information

Data matrix:
counts or UMIs

Scanpy

- Pretty much everything you need to know about scanpy is available at https://scanpy.readthedocs.io/en/stable/
- This includes documentation on how to install and how to use

Analysis of single cell data with scanpy

(Almost) every command is called on the adata object.

API has three main groups of functions:

- `sc.pp` for preprocessing
- `sc.tl` for tools (normalisation and dimension reduction etc)
- `sc.pl` for plotting

Scanpy tools operate *inplace* on adata. E.g.

```
        sc.tl.umap(adata)
```

Will store the computed umap data in the adata object automatically, you don’t have to assign it to anything.

Scanpy tutorial

- 3k PBMCs from a Health Donor dataset from 10x Genomics.

- As we go through the tutorial keep an eye on the various structures in the adata object and how they change to get an idea of how everything fits together.

- https://scanpy-tutorials.readthedocs.io/en/latest/pbmc3k.html