PhysiCell data analysis with physicell data loader

Elmar Bucher
PhD Student

2023-09-29

Fertig Lab meeting

PhysiCell

classical mechanics based cell and tissue simulator

Agent based modeling:

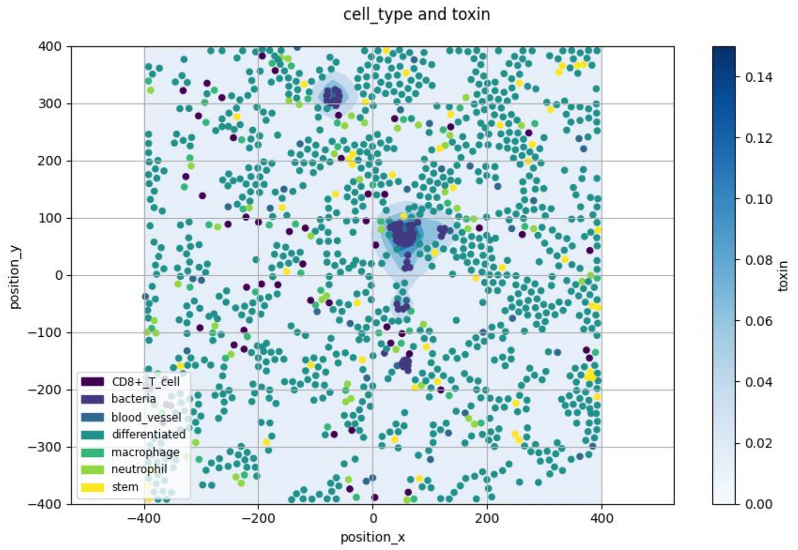
- each cell is an agent.
- agents have states (parameters and features).
- agents run functions.

Diffusion reaction solver:

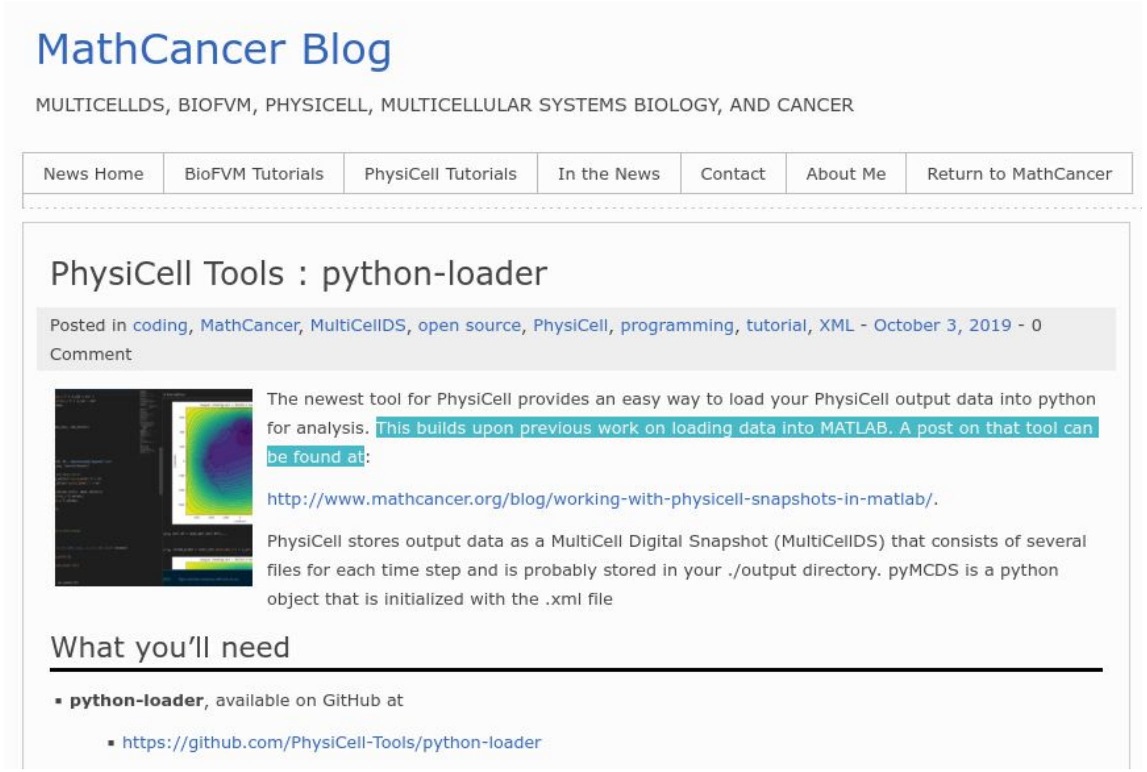
- substrate source, sink, and decay.
- finite volume method (BioFVM).

Multiscale modeling

- space: intracellular < cell < domain (tissue)
- time: diffusion 0.01[min] < mechanics 0.1[min] < phenotype 6[min]



python data loader (2019-9-28)



MathCancer Blog

MULTICELLDS, BIOFVM, PHYSICELL, MULTICELLULAR SYSTEMS BIOLOGY, AND CANCER

PhysiCell Tools : python-loader

Posted in coding, MathCancer, MultiCellDS, open source, PhysiCell, programming, tutorial, XML - October 3, 2019 - 0 Comment

The newest tool for PhysiCell provides an easy way to load your PhysiCell output data into python for analysis. This builds upon previous work on loading data into MATLAB. A post on that tool can be found at:

http://www.mathcancer.org/blog/working-with-physicell-snapshots-in-matlab/.

PhysiCell stores output data as a MultiCell Digital Snapshot (MultiCellDS) that consists of several files for each time step and is probably stored in your ./output directory. pyMCDS is a python object that is initialized with the .xml file

What you'll need

- **python-loader**, available on GitHub at
  - https://github.com/PhysiCell-Tools/python-loader

pcdl - physicell data loader

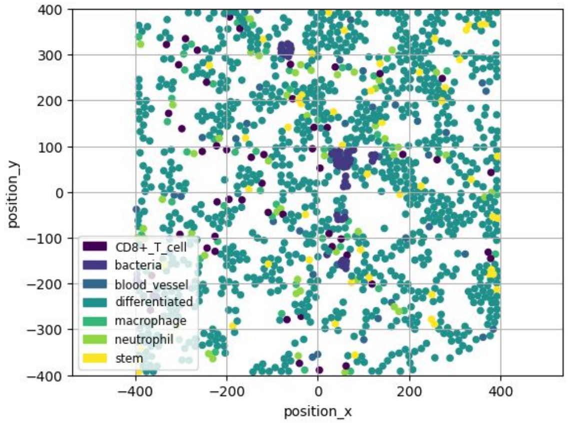
load PhysiCell output into python3

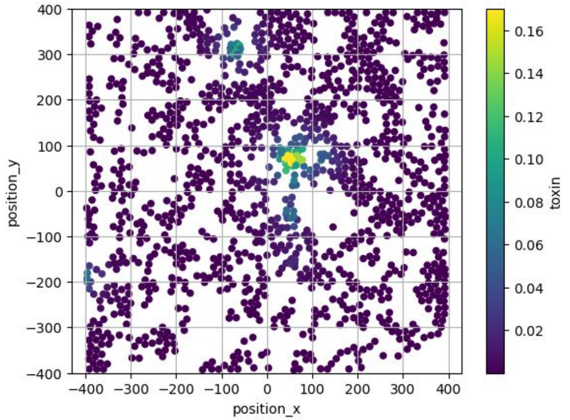
Aim:

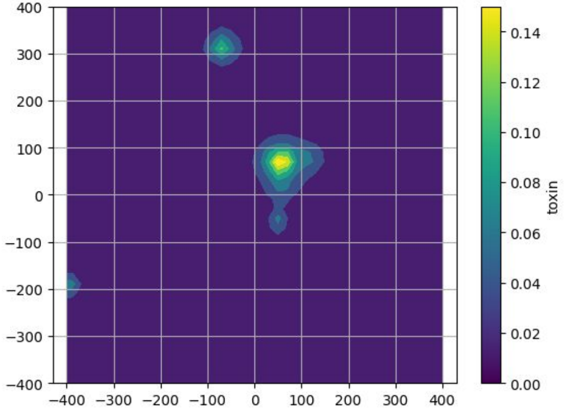
- platform independent (**Windows**, **MacOS**, **Linux**).
- **backwards compatible** (`pyMCDS.py`).
- **pip** installable.
- pcdl **is not analysis software**!
- pcdl is your **default connector between PhysiCell output and analysis software**.

pcdl plots (battery included)

- `mcds.plot_scatter()`, `mcds.plot_contour()`, `mcds.plot_timeseries()`
- matplotlib plots!







- `mcdsts.make_gif()`, and `mcdsts.make_movies()`

pandas

R library for python

R + python powered ≡ pandas

- R Vector ≡ pandas Series
- R DataFrame ≡ pandas DataFrame
  - `df.loc[row, column]`
  - `df.iloc[row_number, column_number]`

- if you miss ggplot - plotnine might serve you:
  - https://plotnine.readthedocs.io/en/stable/index.html#

pcdl and pandas

fetch dataframes

- `df = mcds.get_cell_df()` # agent
- `df = mcds.get_conc_df()` # substrat

filter dataframes

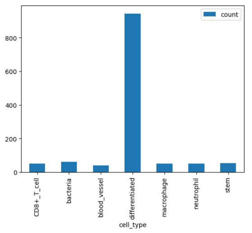
- `df = mcdsts.get_cell_df_features(feature_values=2)` # agent
- `df = mcdsts.get_conc_df_features(feature_values=2)` # substrat

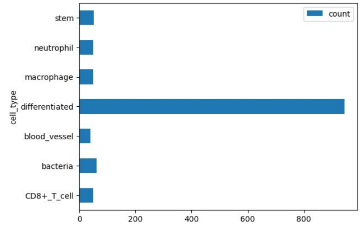
pandas plots (battery included)

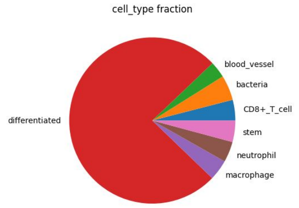
one line of code:

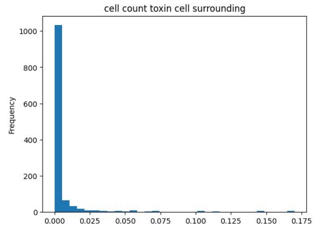
- df.plot(kind='bar')
- df.plot(kind='bah')
- df.plot(kind='pie', …)
- df.plot(kind='hist')
- df.plot(kind='kde')
- df.plot(kind='box')
- df.plot(kind='line')
- df.plot(kind='area')

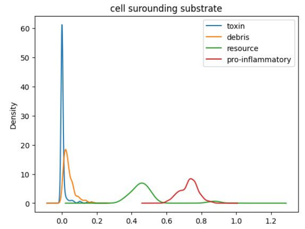
matplotlib plots!

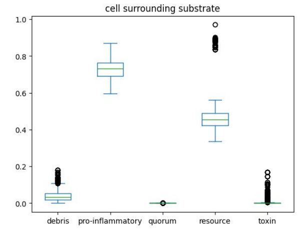


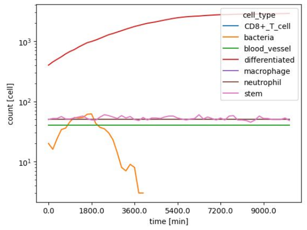


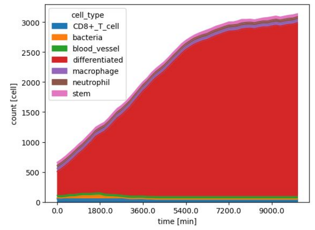












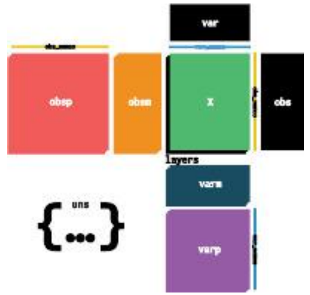
scverse - the single cell universe

Fabian Theis Lab

homepage: https://scverse.org/

data formats:

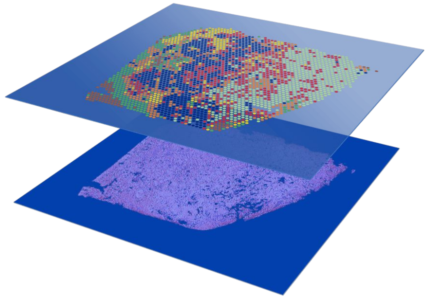
- **anndata**: https://anndata.readthedocs.io/en/latest/
- spatialdata: https://spatialdata.scverse.org/en/latest/
- mudata: https://mudata.readthedocs.io/en/latest/



data analysis:

- **scanpy**: sincle cell analysis: https://scanpy.readthedocs.io/en/latest/
- **squidpy**: spatial single cell analysis: https://squidpy.readthedocs.io/en/stable/
- **squidpy embeded napari**: https://napari.org/stable/
- **scvi-tools**: single-cell deep learning: https://scvi-tools.org/
- muon: multimodal omics analysis: https://muon.scverse.org/
- scirpy: single cell immune sequencing analysis: https://scirpy.scverse.org/en/latest/
- **scverse ecosystem**: https://scverse.org/packages/#ecosystem

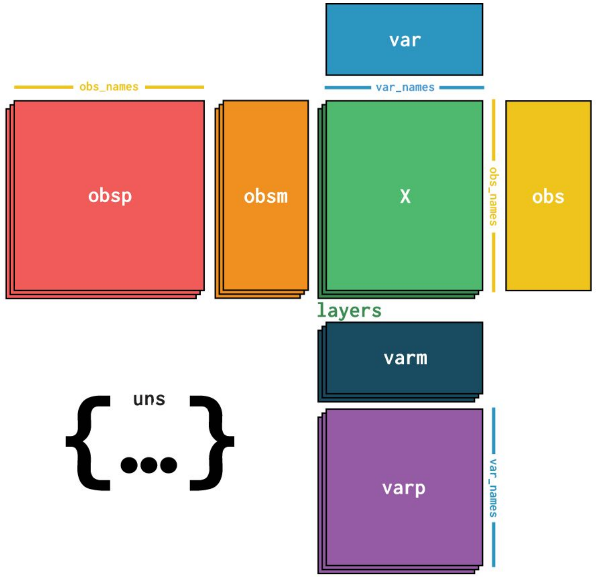
scverse & visum x10 data

10x GENOMICS® +  + squidpy

squidpy:

- https://squidpy.readthedocs.io/en/stable/notebooks/tutorials/tutorial_visium_hne.html
- https://squidpy.readthedocs.io/en/stable/notebooks/tutorials/tutorial_visium_fluo.html
- https://squidpy.readthedocs.io/en/stable/notebooks/tutorials/tutorial_tangram.html

pcdl & scverse



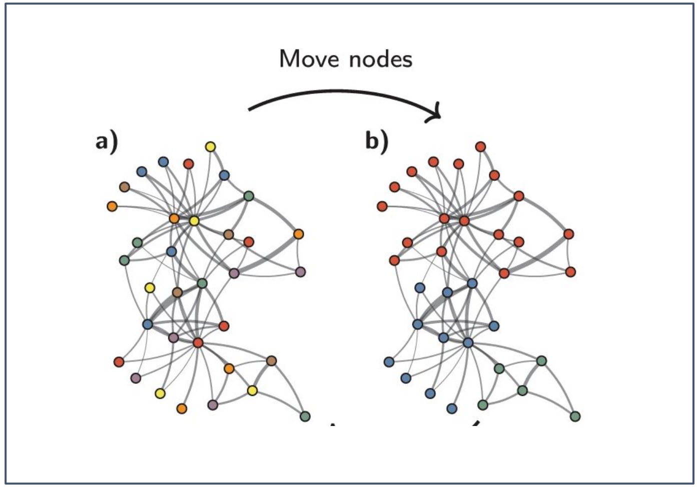
fetch anndata objects

- `ann = mcds.get_anndata()`
- `ann = mcdsts.get_anndata()`
- `ann = mcdsts.get_anndata(collapsed=False)`

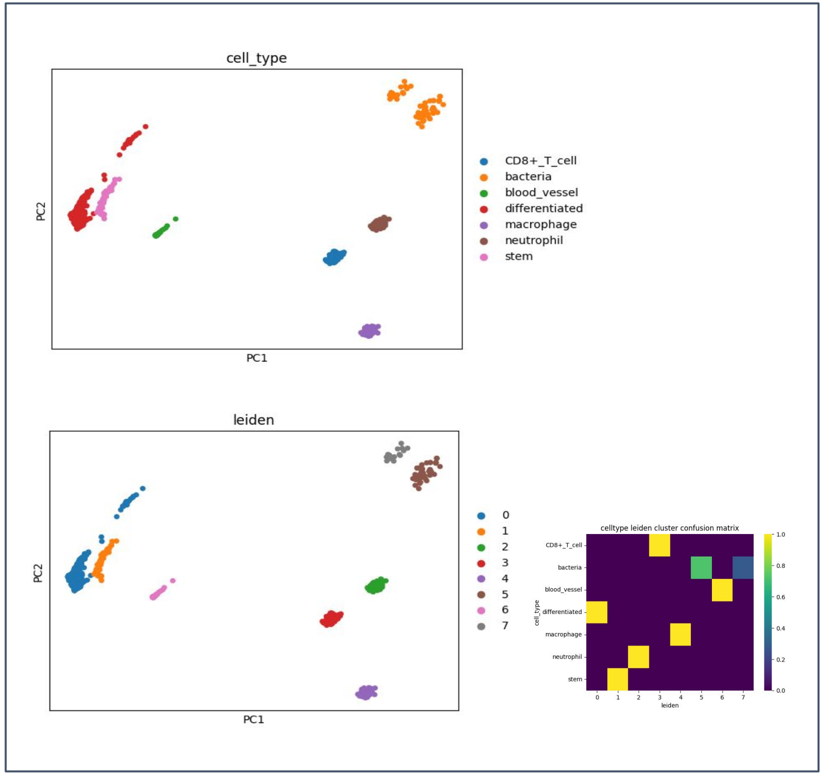
scanpy & neigberhood graph clustering

two lines of code:

- `sc.pp.neighbors(ann, n_neighbors=15)` # preprocess
- `sc.tl.leiden(ann, resolution=0.01)` # tool



Traag et al. From Louvain to Leiden: guaranteeing well-connected communities. Sci Rep 9, 5233 (2019). https://doi.org/10.1038/s41598-019-41695-z



https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb

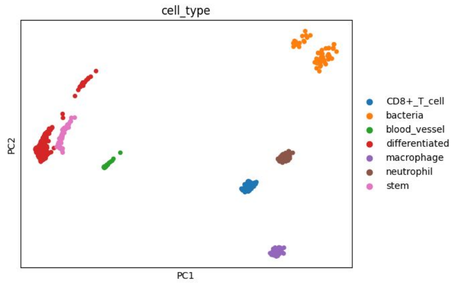
scanpy & dimensional reduction

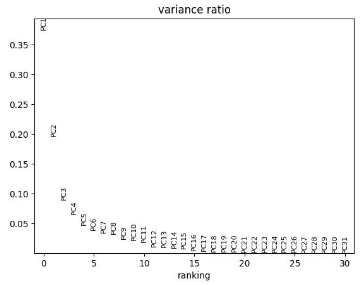
two lines of code:

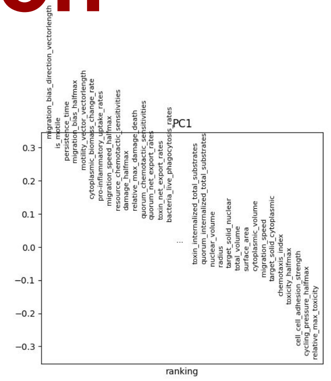
- `sc.tl.tsne(ann)` # tool
- `sc.tl.tsne(ann)` # plot

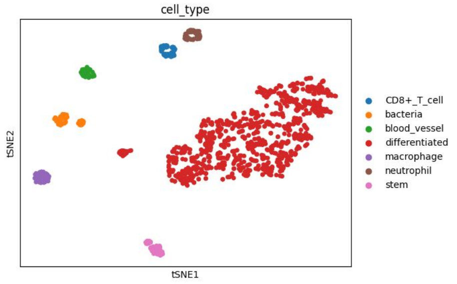
methods:

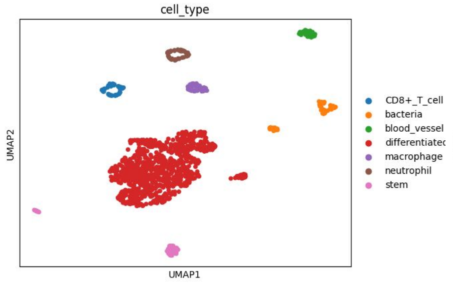
- Distributed Recursive
- ForceAtlas2
- Fruchterman Reingold
- Grid Fruchterman Reingold
- Kamadi Kawai
- Large Graph
- **PCA**
- Reingold Tilford Tree
- Reingold Tilford Circular
- **tSNE**
- **UMAP**

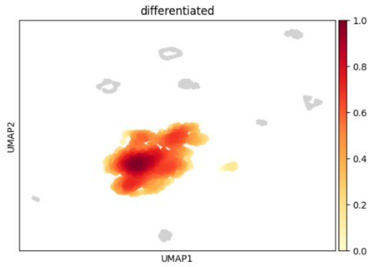








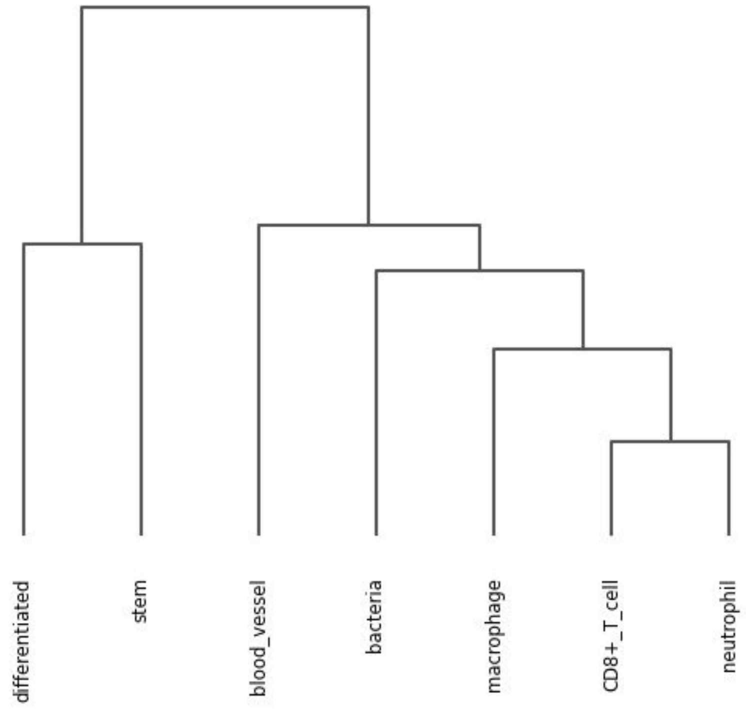




scanpy & dendrogram

two lines of code:

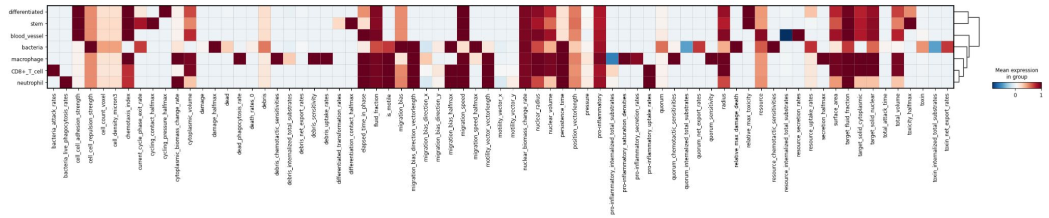
- `sc.tl.dendrogram(ann, groupby='cell_type')` # tool
- `sc.pl.dendrogram(ann, groupby='cell_type')` # plot

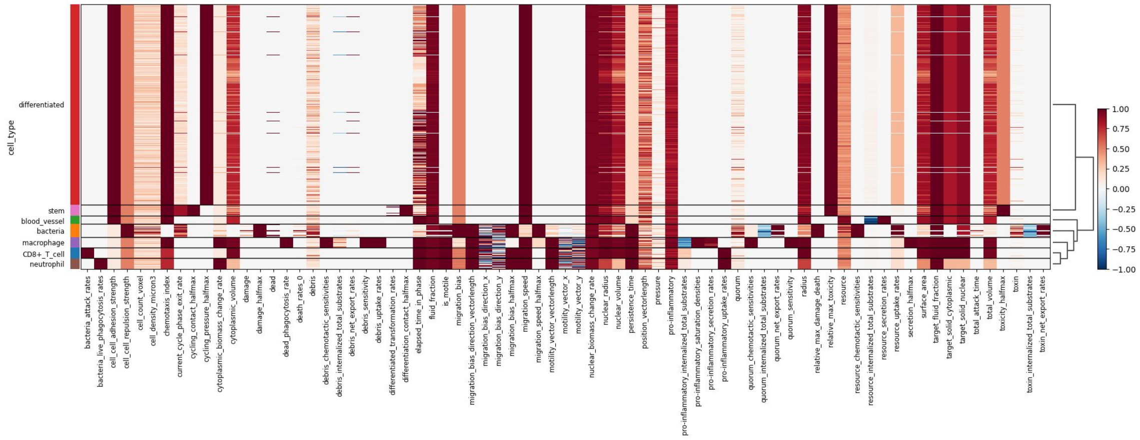


scanpy & heatmaps

one lines of code:

- `sc.pl.matrixplot(ann, …)` # plot
- `sc.pl.dotplot(ann, …)` # plot
- `sc.pl.stacked_violin(ann, …)` # plot
- `sc.pl.tracksplot(ann, …)` # plot
- `sc.pl.heatmap(ann, …)` # plot
- `sc.pl.clustermap(ann, …)` # plot



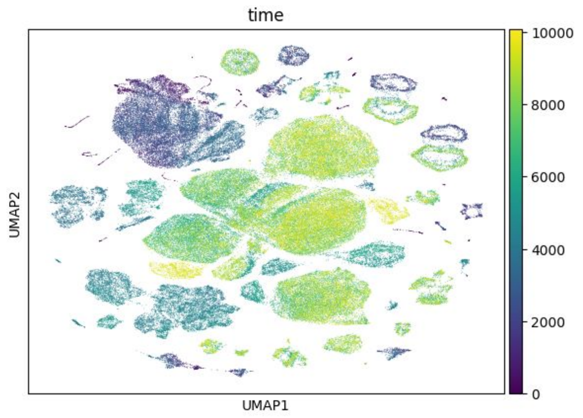


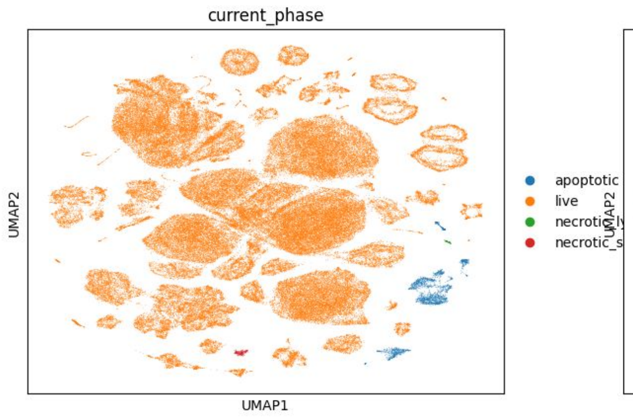
scanpy & timeseries

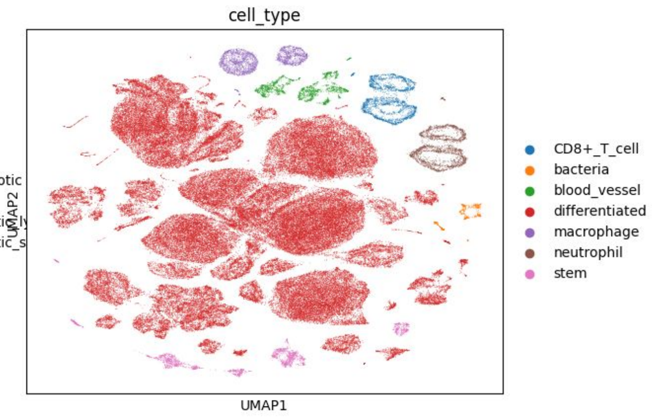
two lines of code:

- `sc.tl.umap(ann, …)` # tool
- `sc.tl.umap(ann, …)` # plot

this is physicell output!



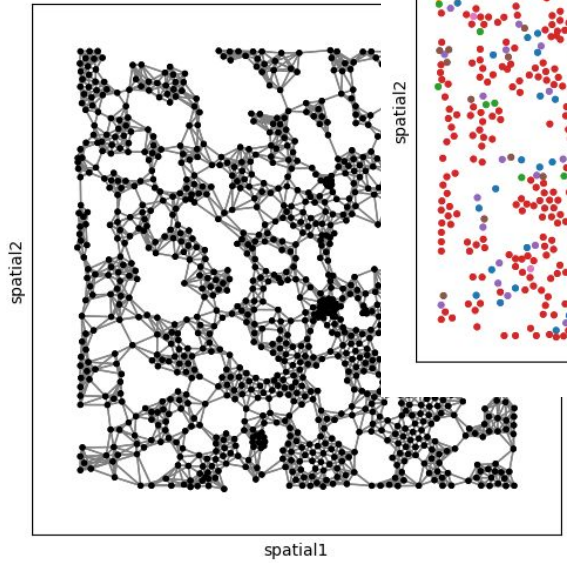


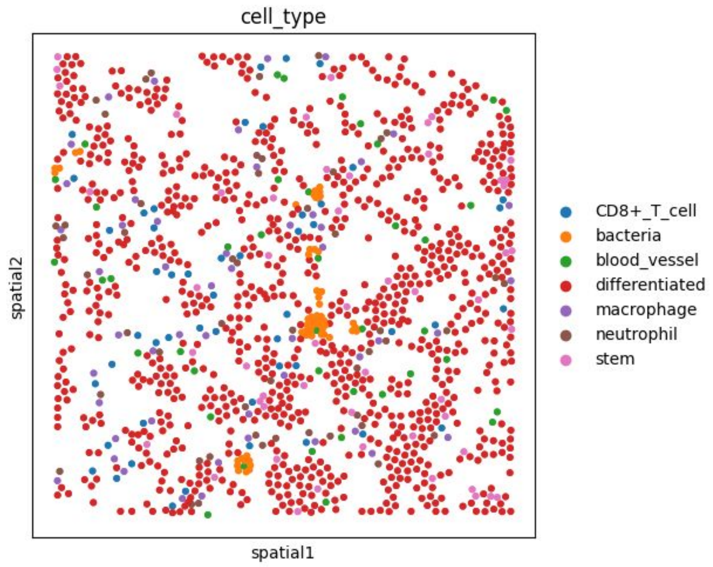


squidpy & spatial

one line of code:

- `sq.pl.spatial_scatter(ann, …)` # plot
- `sq.pl.spatial_neighbors(ann, …)` # plot



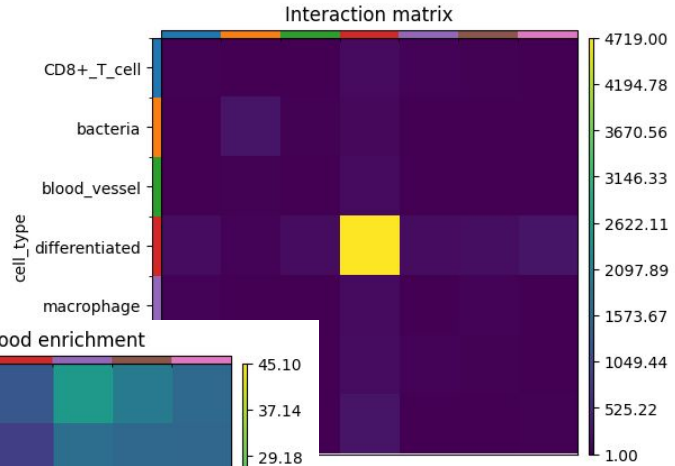


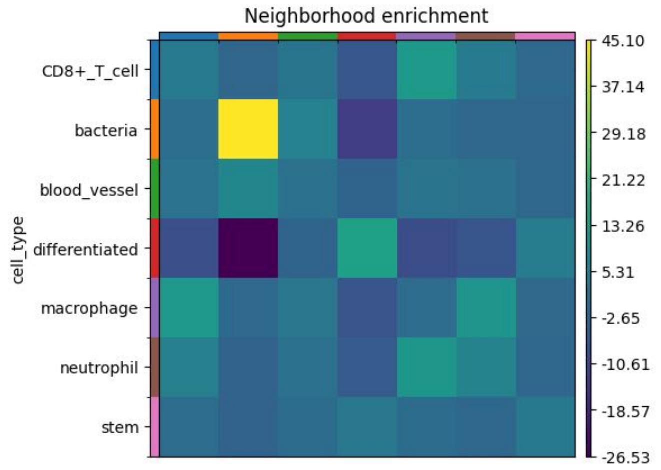
squidpy & neigbourhood

two lines of code:

- `sq.gr.interaction_matrix(ann, …)` # graph
- `sq.pl.interaction_matrix(ann, …)` # plot

- `sq.gr.nhood_enrichment(ann, …)` # graph
- `sq.gr.nhood_enrichment(ann, …)` # plot

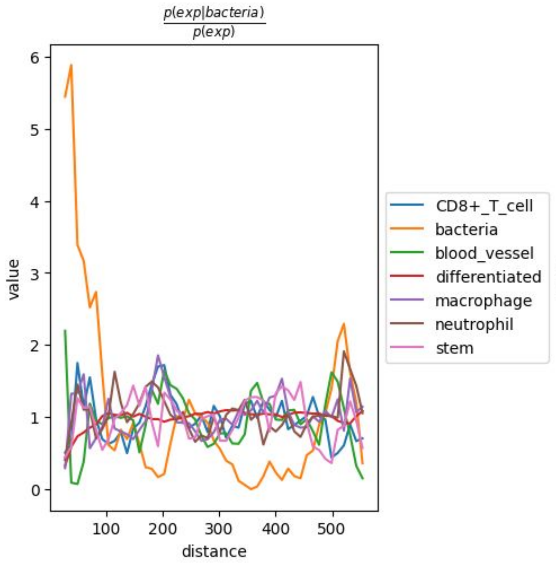


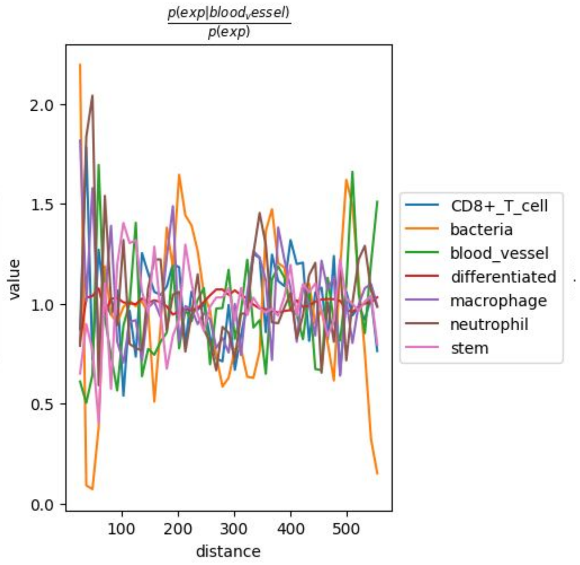


squidpy & co_occurrence

two lines of code:

- `sq.gr.co_occurrence(ann, …)` # graph
- `sq.pl.co_occurrence(ann, …)` # plot

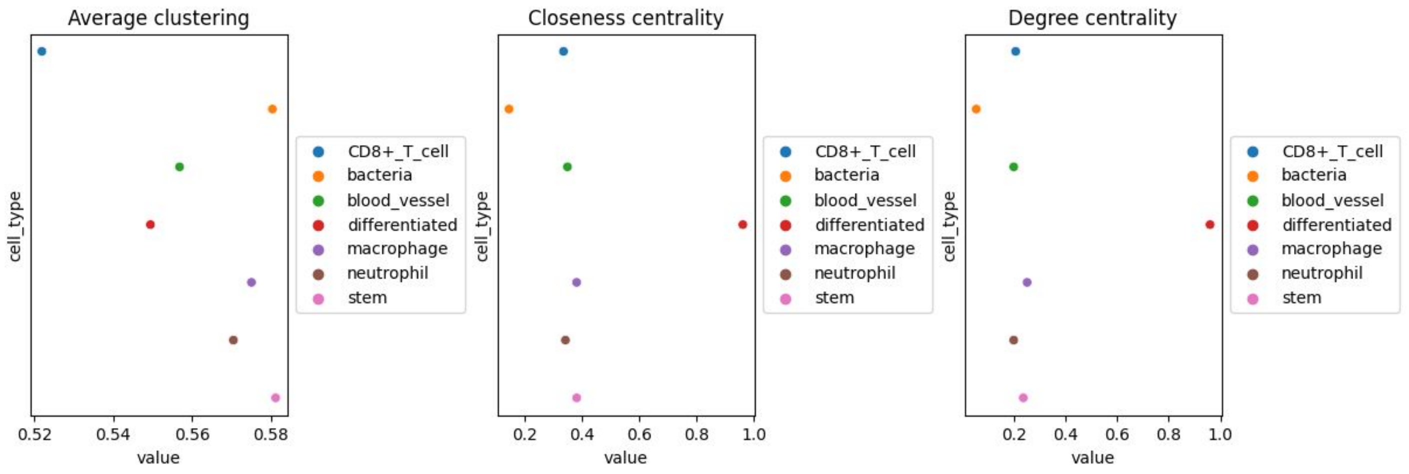




squidpy & centrality_scores

two lines of code:

- `sq.gr.centrality_scores(ann, …)` # graph
- `sq.pl.centrality_scores(ann, …)` # plot

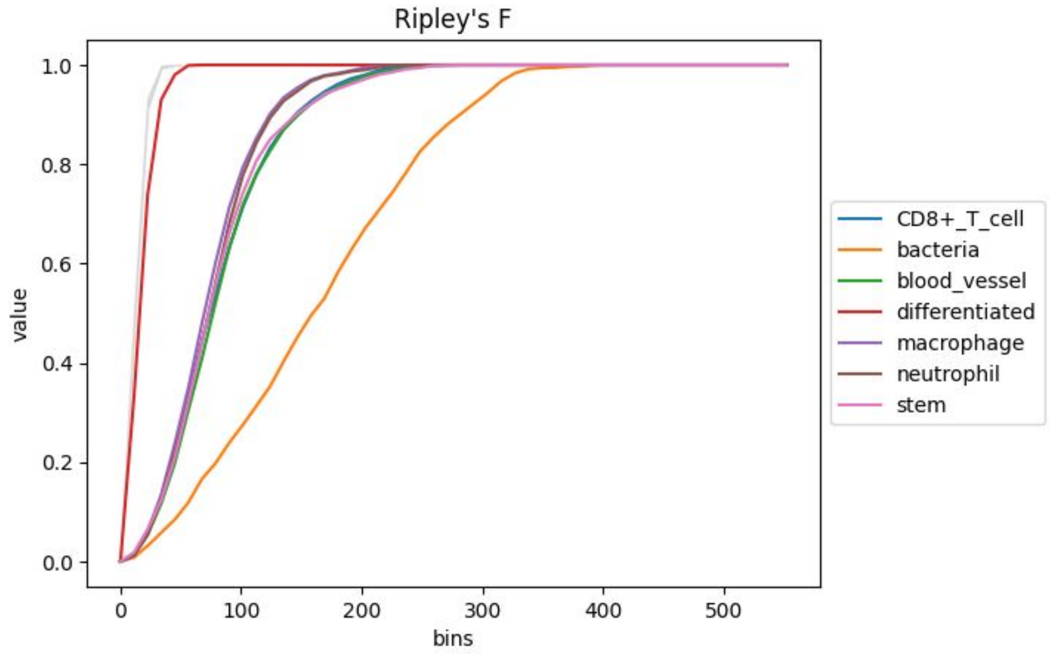


squidpy & Ripley's statistics

for point processes

two lines of code:

- `sq.gr.ripley(ann, …)` # graph
- `sq.gr.ripley(ann, …)` # plot

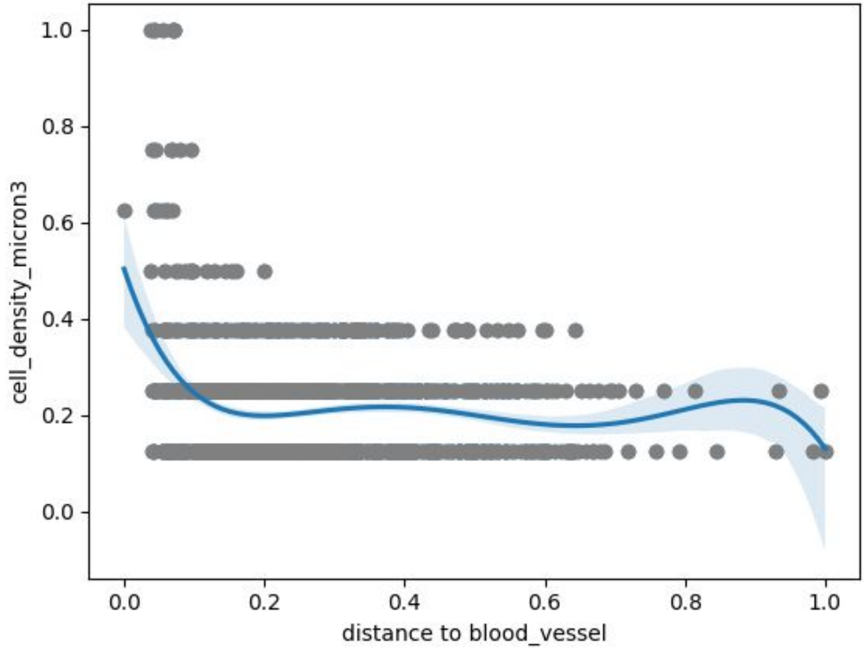


squidpy & var_by_distance

plot one **numerical feature (var; y-axis)** against a **categorical parameter (obs; x-axis)**.
e.g. voxel cell density against distance to blood vessel.

two lines of code:

- `sq.tl.var_by_distance(ann, …)` # tool
- `sq.pl.var_by_distance(ann, …)` # plot

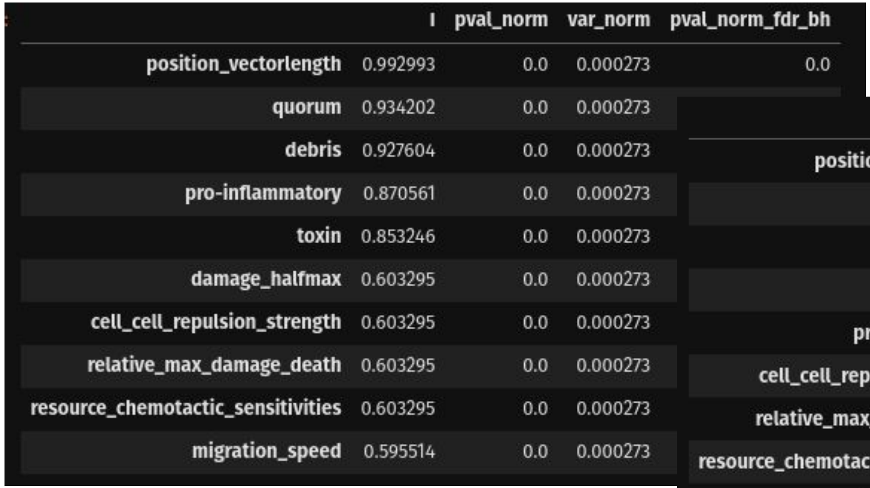


squidpy & global autocorrelation statistic

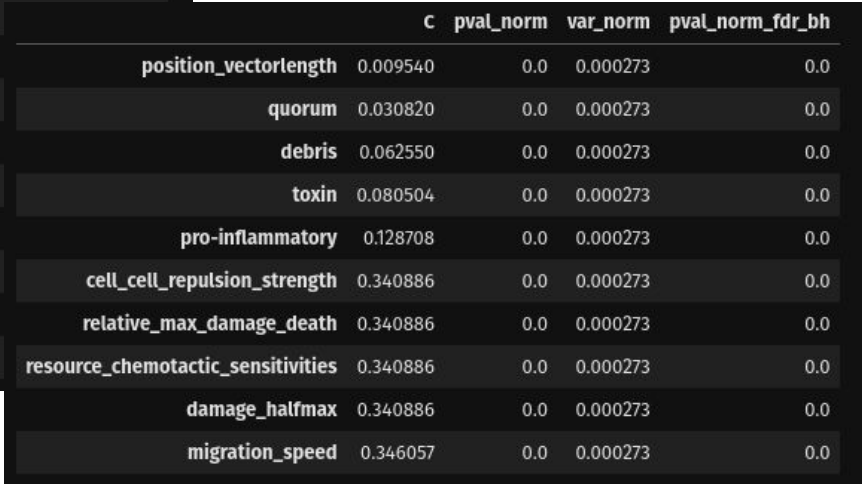
Moran's I and Geary's C

one line of code:

- `sq.gr.spatial_autocorr(ann, mode='moran')` # graph
- `ann.uns["moranI"].head(10)` # datafame



	I	pval_norm	var_norm	pval_norm_fdr_bh
position_vectorlength	0.992993	0.0	0.000273	0.0
quorum	0.934202	0.0	0.000273	
debris	0.927604	0.0	0.000273	
pro-inflammatory	0.870561	0.0	0.000273	
toxin	0.853246	0.0	0.000273	
damage_halfmax	0.603295	0.0	0.000273	
cell_cell_repulsion_strength	0.603295	0.0	0.000273	
relative_max_damage_death	0.603295	0.0	0.000273	
resource_chemotactic_sensitivities	0.603295	0.0	0.000273	
migration_speed	0.595514	0.0	0.000273	



	C	pval_norm	var_norm	pval_norm_fdr_bh
position_vectorlength	0.009540	0.0	0.000273	0.0
quorum	0.030820	0.0	0.000273	0.0
debris	0.062550	0.0	0.000273	0.0
toxin	0.080504	0.0	0.000273	0.0
pro-inflammatory	0.128708	0.0	0.000273	0.0
cell_cell_repulsion_strength	0.340886	0.0	0.000273	0.0
relative_max_damage_death	0.340886	0.0	0.000273	0.0
resource_chemotactic_sensitivities	0.340886	0.0	0.000273	0.0
damage_halfmax	0.340886	0.0	0.000273	0.0
migration_speed	0.346057	0.0	0.000273	0.0

LUDDY
SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb

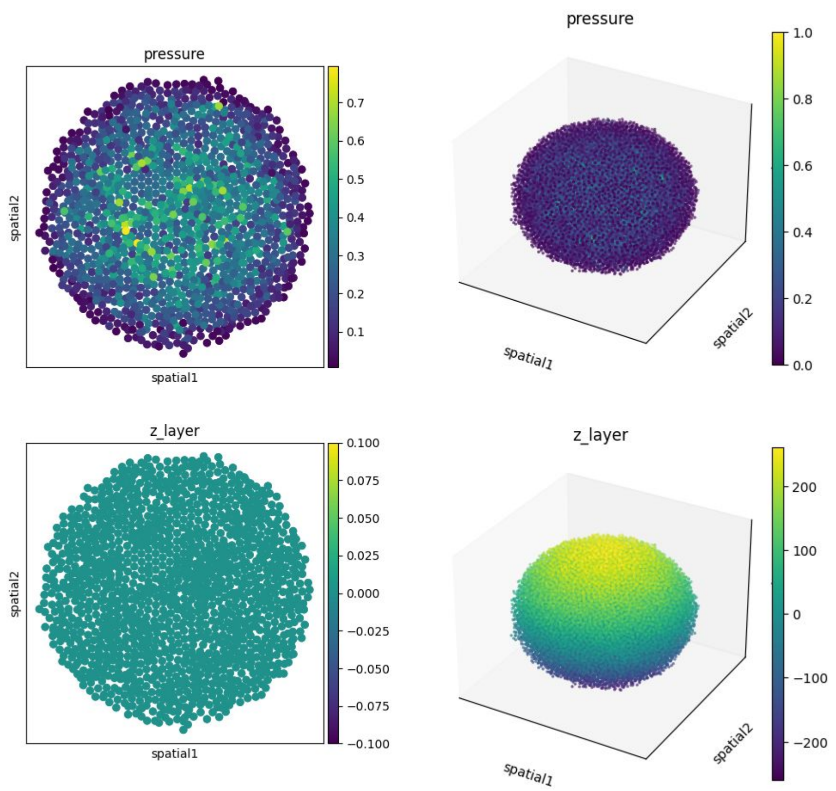
PhysiCell Project
PhysiCell.org
@PhysiCell

squidpy & z-stacks

one line of code:

- `sq.pl.spatial_scatter(ann3d … )` # plot
- `sc.pl.embedding(ann3d …)` # plot

this is physicell output!



Where to go from here?

the basics:

- learn core python3: https://www.python.org/
  [resources: https://nostarch.com/python-kids-2nd-edition, https://realpython.com/]
- learn numpy: https://numpy.org/
- learn scipy: https://scipy.org/
- learn pandas: https://pandas.pydata.org
- learn matplotlib: https://matplotlib.org/

depending based on your needs, learn:

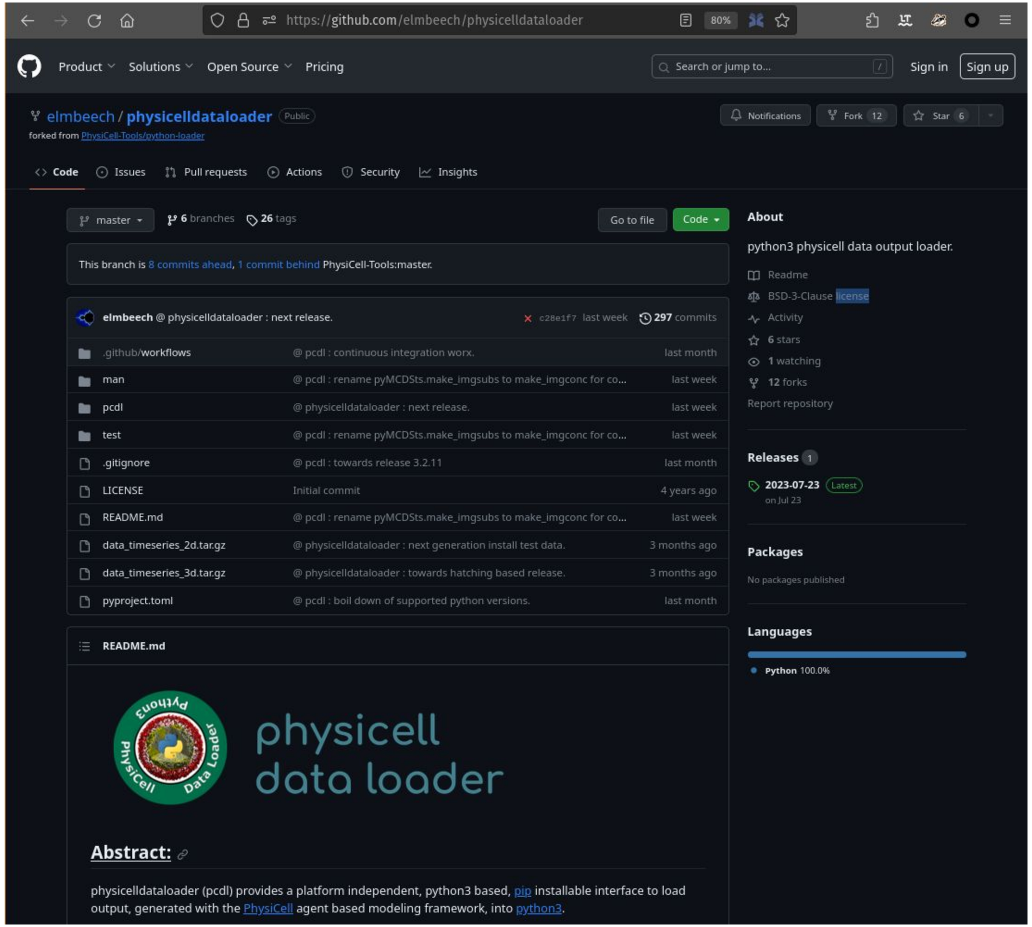
- statsmodels: https://www.statsmodels.org/stable/index.html
- sklearn: https://scikit-learn.org/stable/
- pytorch: https://pytorch.org/
- skimage: https://scikit-image.org/
- …

at https://scverse.org :

- read the anndata docs - the basic data object
- read the scanpy docs - single cell analysis
- read the squidpy docs - spatial single cell analysis
- read the scvi-tools docs - single cell machine learning
- …

pcdl homepage

- work through the **TUTORIAL.md**.
- work through the **Jupyter notebook.**
- read the **docstrings** from the "**workhorse functions**" mentioned in the **REFERENCE.md**



elmbeech / physicelldataloader Public

forked from PhysiCell-Tools/python-loader

6 branches 26 tags

This branch is 8 commits ahead, 1 commit behind PhysiCell-Tools:master.

elmbeech @ physicelldataloader : next release. c28e1f7 last week 297 commits

.github/workflows	@ pcdl : continuous integration worx.	last month
man	@ pcdl : rename pyMCDSts.make_imgsubs to make_imgconc for co...	last week
pcdl	@ physicelldataloader : next release.	last week
test	@ pcdl : rename pyMCDSts.make_imgsubs to make_imgconc for co...	last week
.gitignore	@ pcdl : towards release 3.2.11	last month
LICENSE	Initial commit	4 years ago
README.md	@ pcdl : rename pyMCDSts.make_imgsubs to make_imgconc for co...	last week
data_timeseries_2d.tar.gz	@ physicelldataloader : next generation install test data.	3 months ago
data_timeseries_3d.tar.gz	@ physicelldataloader : towards hatching based release.	3 months ago
pyproject.toml	@ pcdl : boil down of supported python versions.	last month

README.md

physicell data loader

Abstract:

physicelldataloader (pcdl) provides a platform independent, python3 based, pip installable interface to load output, generated with the PhysiCell agent based modeling framework, into python3.

About

python3 physicell data output loader.

Readme
BSD-3-Clause license
Activity
6 stars
1 watching
12 forks
Report repository

Releases 1

2023-07-23 Latest
on Jul 23

Packages

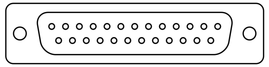
No packages published

Languages

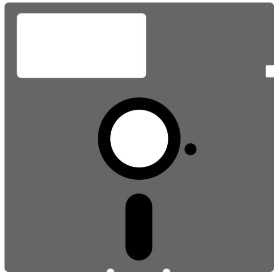
Python 100.0%

Conclusion

- pcdl is simply a **connector**, an **interface**.



- for software development: don’t re-invent the wheel! learn, make use of, and contribute to in the filed **well established high-level libraries**.



Acknowledgement

The whole **MathCancer** lab!

python-loader original implementation:

- Pat Wall
- Randy Heiland
- Paul Macklin

pcdl 2022 / 2023 evolution:

- Ben Jacobs (get_graph)
- Furkan Kurtoglu (get_vtk)
- Heber Rocha (testing)
- Marshal Gress (plot_scatter)
- Thierry-Pascal Fleurant (plot_timeseries)

OHSU:

- Jenny Eng (scanpy)
- Tina Ghodsi Asnaashari (abm)

Software - because pcdl is standing on the shoulder of giants:

- PhysiCell
- PhysiCell Studio (pyMCDS.py)
- Python3 core library
- numpy, scipy, pandas, matplotlib
- anndata
- vtk
- http: requests

Funding:

- 2023-2024 STEM Châteaubriand Fellowship - Office for Science & Technology of the Embassy of France in the United States.

🐧 🐧 🐧