

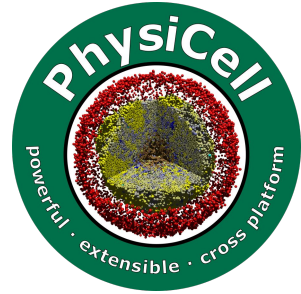
# PhysiCell data analysis with physicell data loader

Elmar Bucher

PhD Student

2023-09-29

## Fertig Lab meeting



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

PhysiCell Project

**PhysiCell.org**

 **@PhysiCell**

# PhysiCell

classical mechanics based cell and tissue simulator

Agent based modeling:

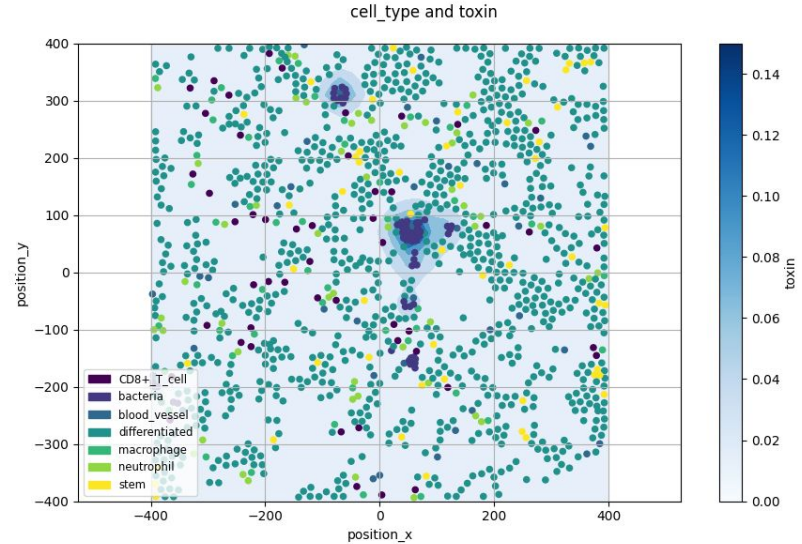
- each cell is an agent.
- agents have states (parameters and features).
- agents run functions.

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

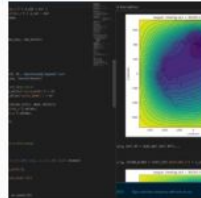
MULTICELLS, BIOFVM, PHYSICELL, MULTICELLULAR SYSTEMS BIOLOGY, AND CANCER

[News Home](#)[BioFVM Tutorials](#)[PhysiCell Tutorials](#)[In the News](#)[Contact](#)[About Me](#)[Return to MathCancer](#)

### 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>



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

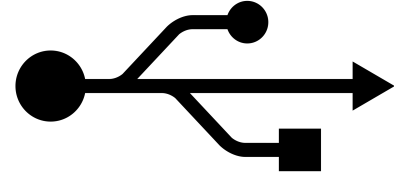
**PhysiCell Project**

**PhysiCell.org**

 **@PhysiCell**

# 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.**



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

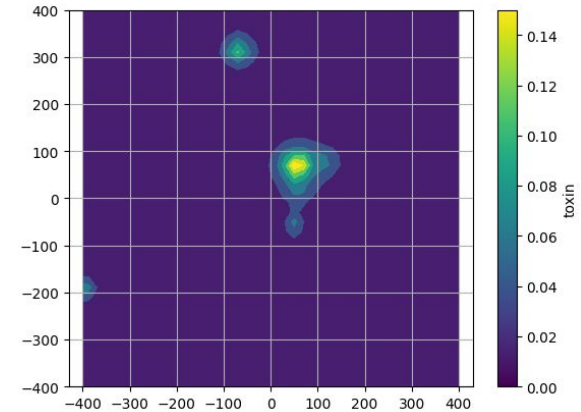
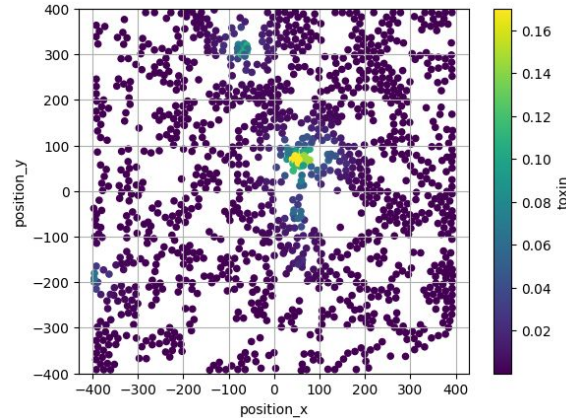
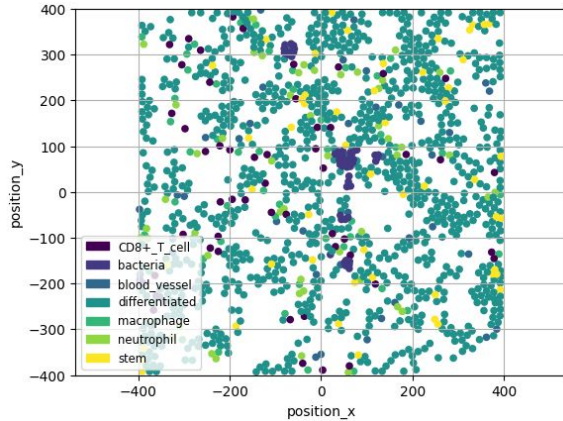
PhysiCell Project

**PhysiCell.org**

 **@PhysiCell**

# pcdl plots (battery included)

- `plot_scatter()`, `plot_contour()`, `plot_timeseries()`
- [matplotlib](#) plots!



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

# pandas

R library for python



- R Vector  $\equiv$  pandas Series
- R DataFrame  $\equiv$  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

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

filter dataframes

- `mcdsts.get_cell_df_states()` # agent
- `mcdsts.get_conc_df_states()` # substrat

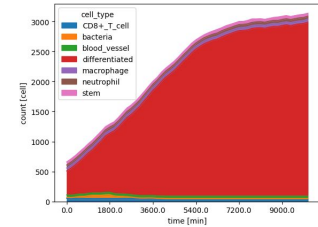
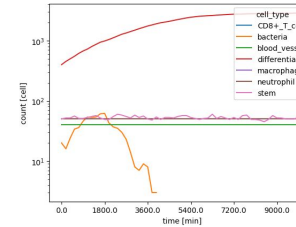
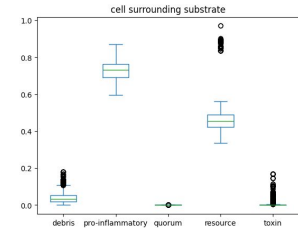
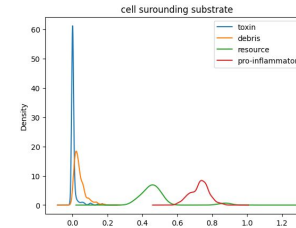
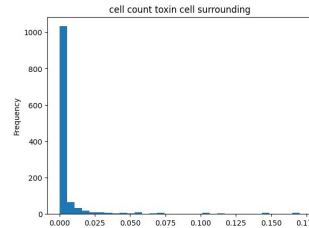
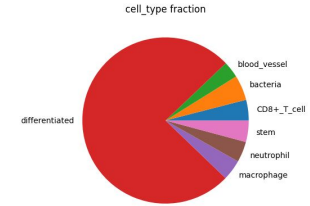
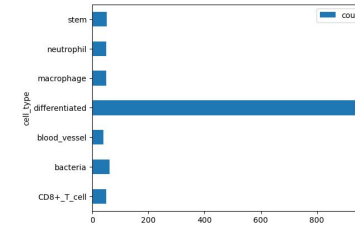
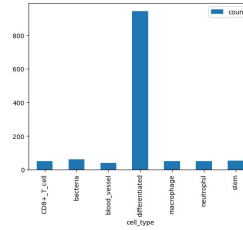


# pandas plots (battery included)

one line of code:

- `df.plot(kind='bar')`
- `df.plot(kind='barh')`
- `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

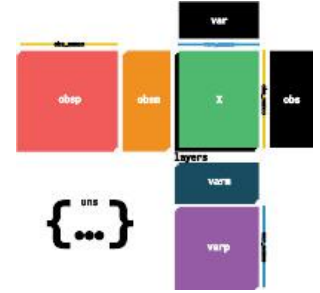
homepage: <https://scverse.org/>

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**: single 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>



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

PhysiCell Project

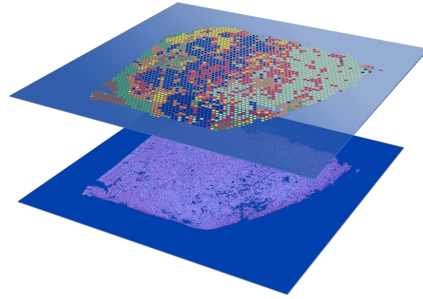
[PhysiCell.org](https://PhysiCell.org)

[@PhysiCell](https://twitter.com/PhysiCell)

# scverse & visum x10 data

10x  
GENOMICS®

+



+



squidpy:

- [https://squidpy.readthedocs.io/en/stable/notebooks/tutorials/tutorial\\_visium\\_hne.html](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_visium_fluo.html)
- [https://squidpy.readthedocs.io/en/stable/notebooks/tutorials/tutorial\\_tangram.html](https://squidpy.readthedocs.io/en/stable/notebooks/tutorials/tutorial_tangram.html)



**LUDDY**

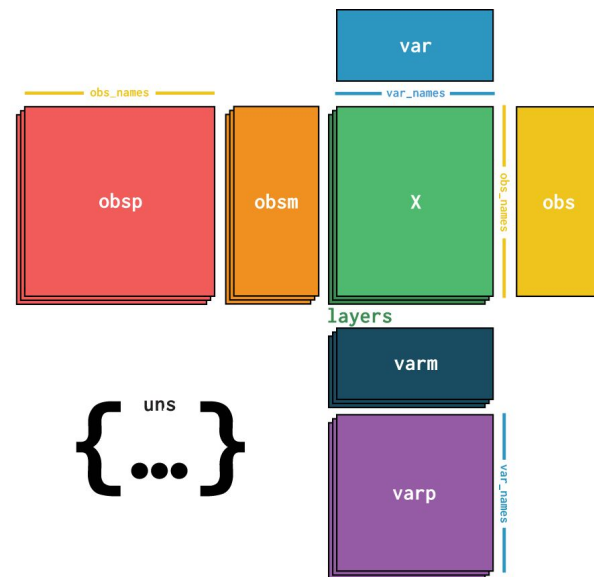
SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

PhysiCell Project

PhysiCell.org

@PhysiCell

# pcdl & scverse



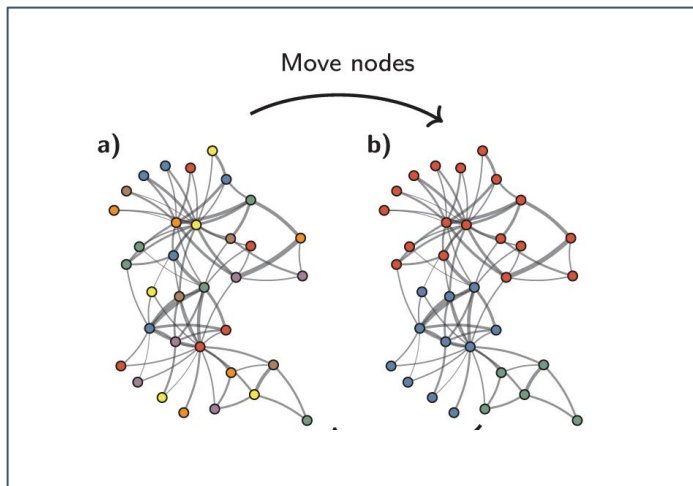
fetch anndata objects

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

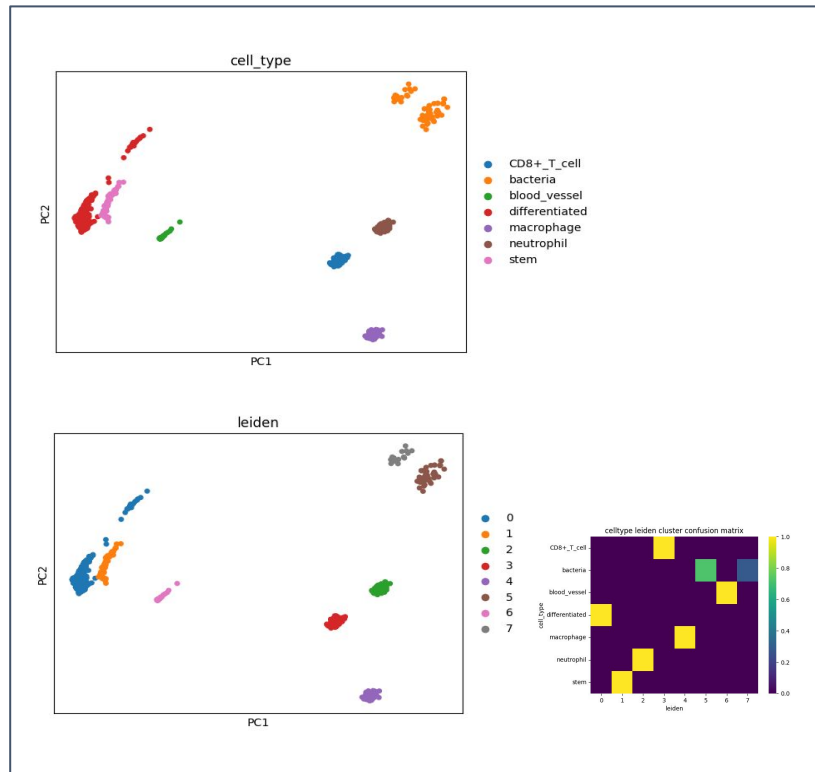
# scanpy & neighborhood graph clustering

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>



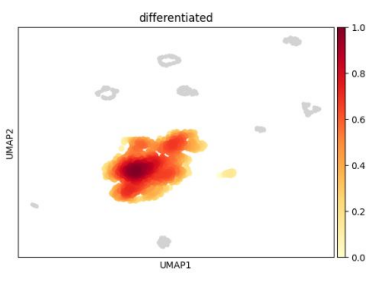
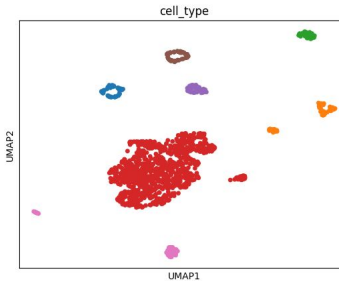
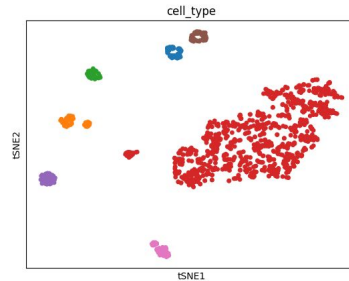
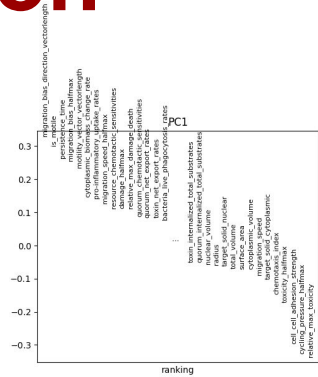
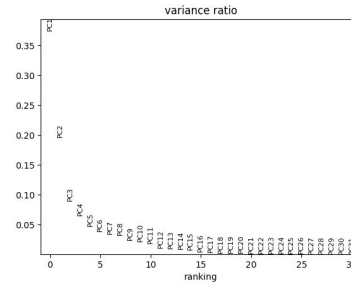
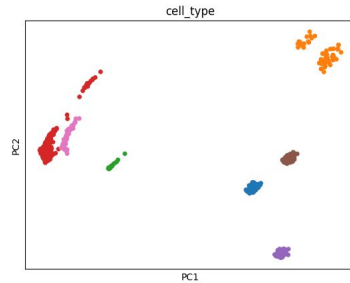
# scanpy & dimensional reduction

two lines of code:

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

methods:

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



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

[https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl\\_repl\\_programming.ipynb](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

**PhysiCell Project**

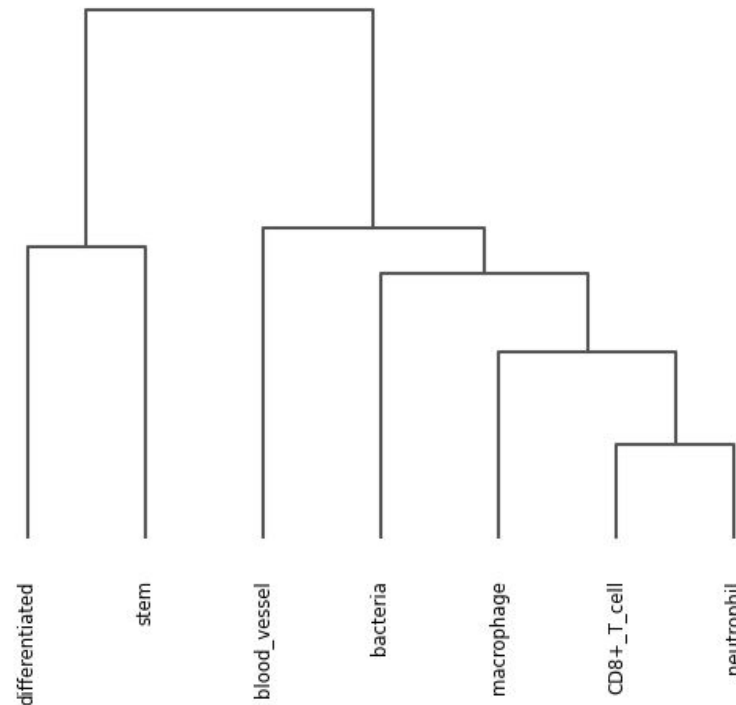
**PhysiCell.org**

**@PhysiCell**

# scanpy & dendrogram

two lines of code:

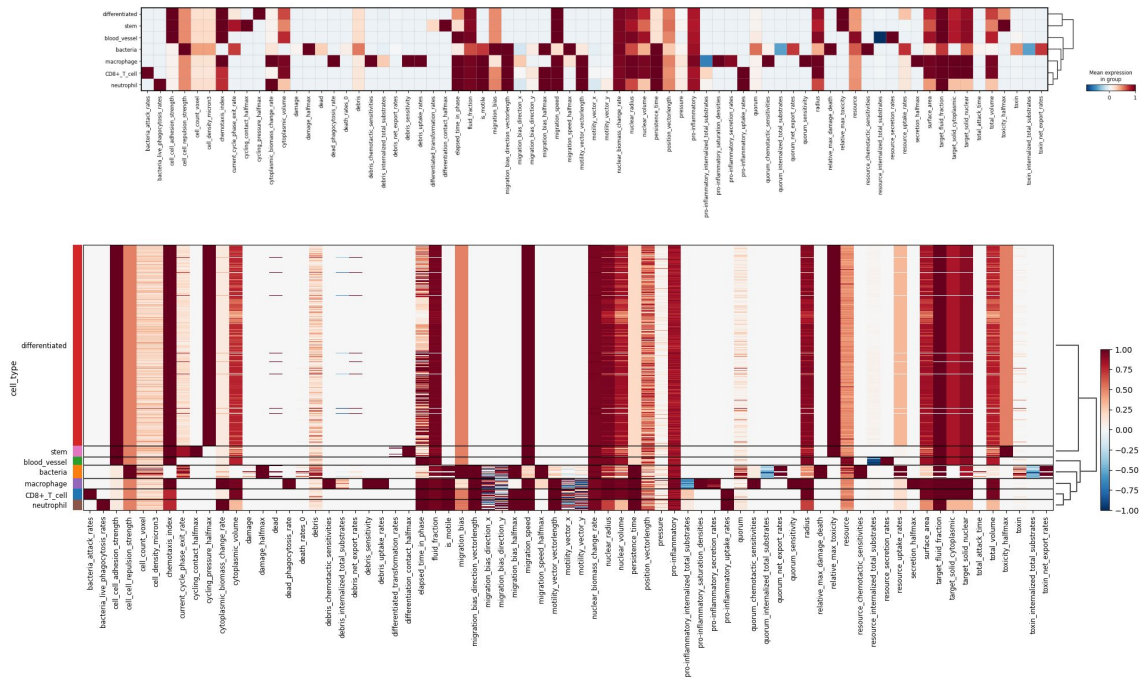
- `sc.tl.dendrogram(ann, groupby='cell_type')`
- `sc.pl.dendrogram(ann, groupby='cell_type')`



# scanpy & heatmaps

one lines of code:

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



# LUDDY

## SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

[https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl\\_repl\\_programming.ipynb](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

# PhysiCell Project

PhysiCell.org

 @PhysiCell

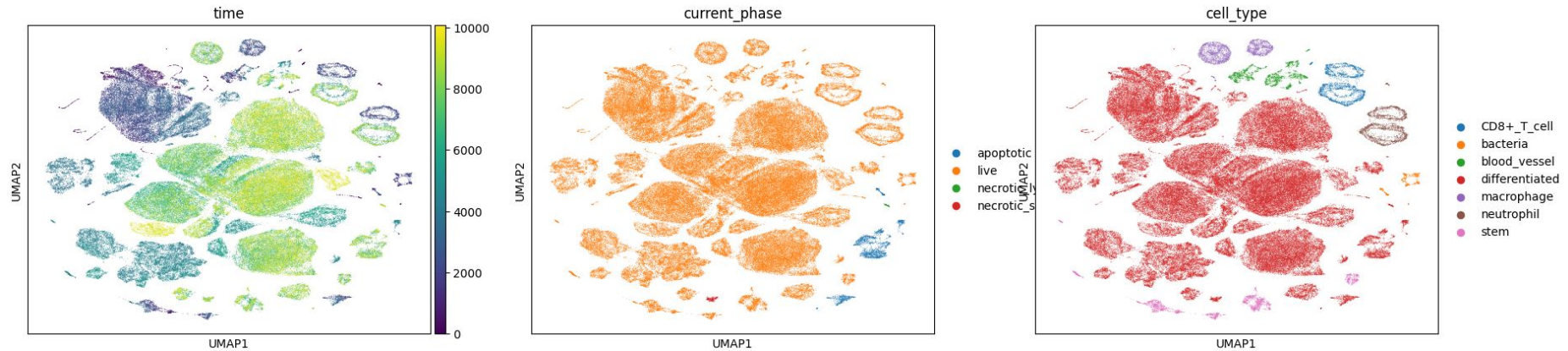


# scanpy & timeseries

two lines of code:

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

this is physicell output!



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

[https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl\\_repl\\_programming.ipynb](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

**PhysiCell Project**

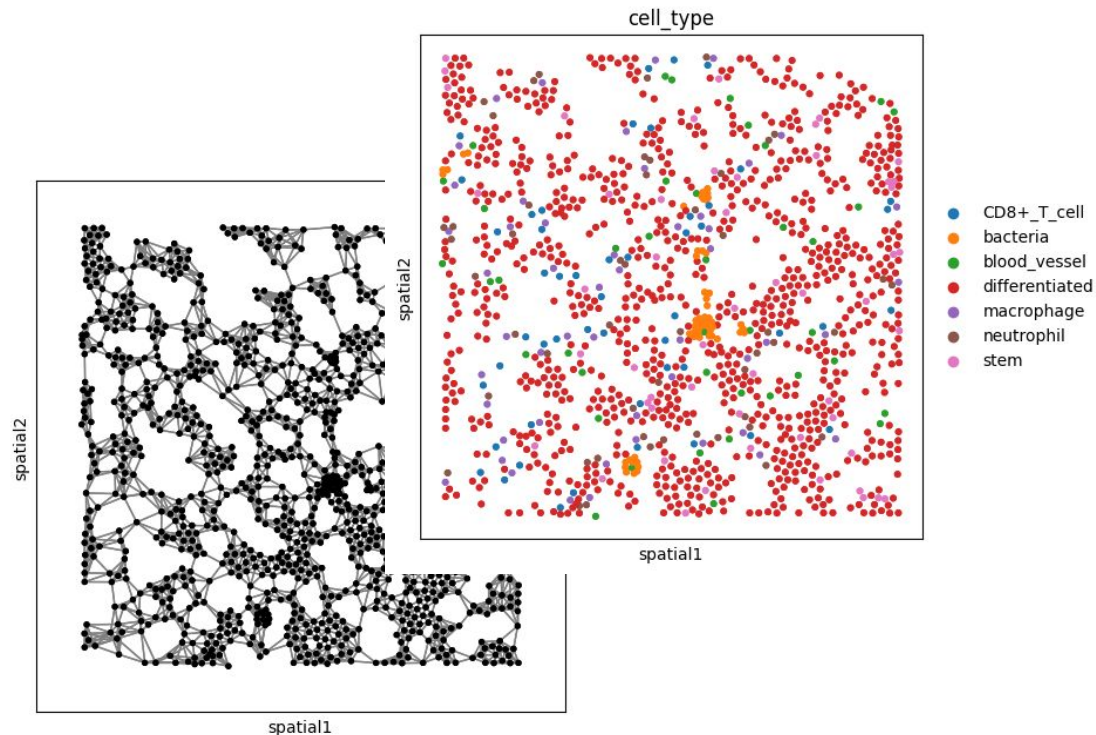
**PhysiCell.org**

 **@PhysiCell**

# squidpy & spatial

one line of code:

- `sq.pl.spatial_scatter(ann, ...)`
- `sq.pl.spatial_neighbors(ann, ...)`



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

[https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl\\_repl\\_programming.ipynb](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

**PhysiCell Project**

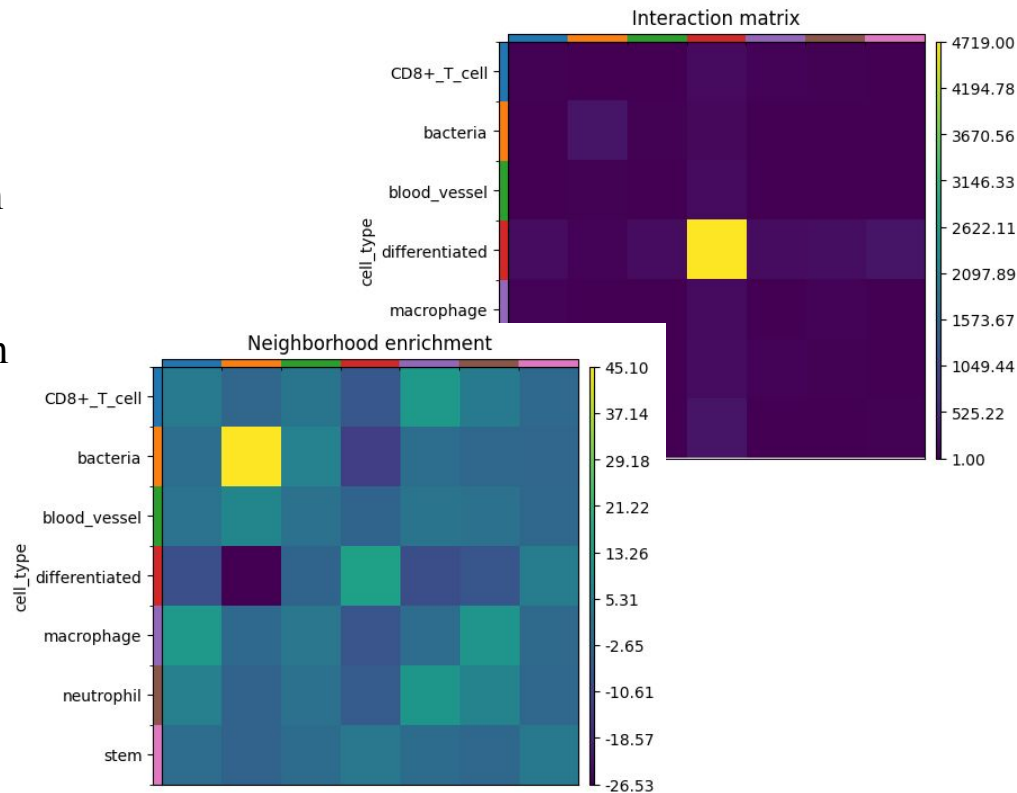
**PhysiCell.org**

**@PhysiCell**

# squidpy & neighbourhood

two lines of code:

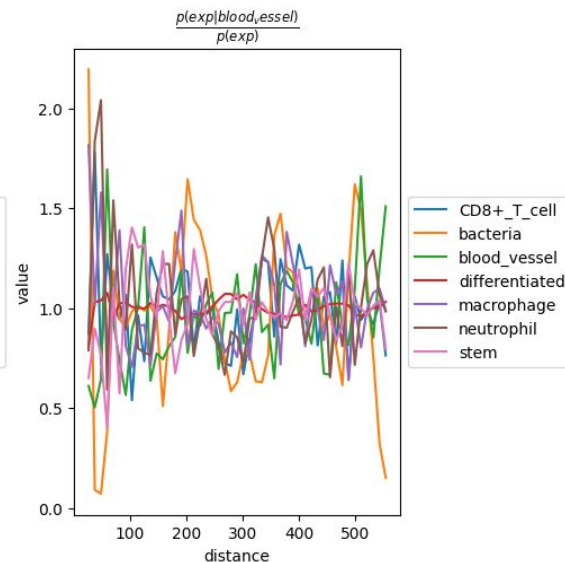
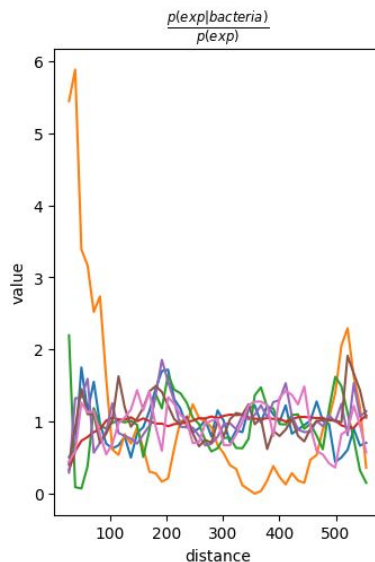
- `sq.gr.interaction_matrix(ann, ...) # graph`
- `sq.pl.interaction_matrix(ann, ...) # plot`
- `sq.gr.nhood_enrichment(ann, ...) # graph`
- `sq.gr.nhood_enrichment(ann, ...) # plot`



# squidpy & co\_occurrence

two lines of code:

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



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

[https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl\\_repl\\_programming.ipynb](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

**PhysiCell Project**

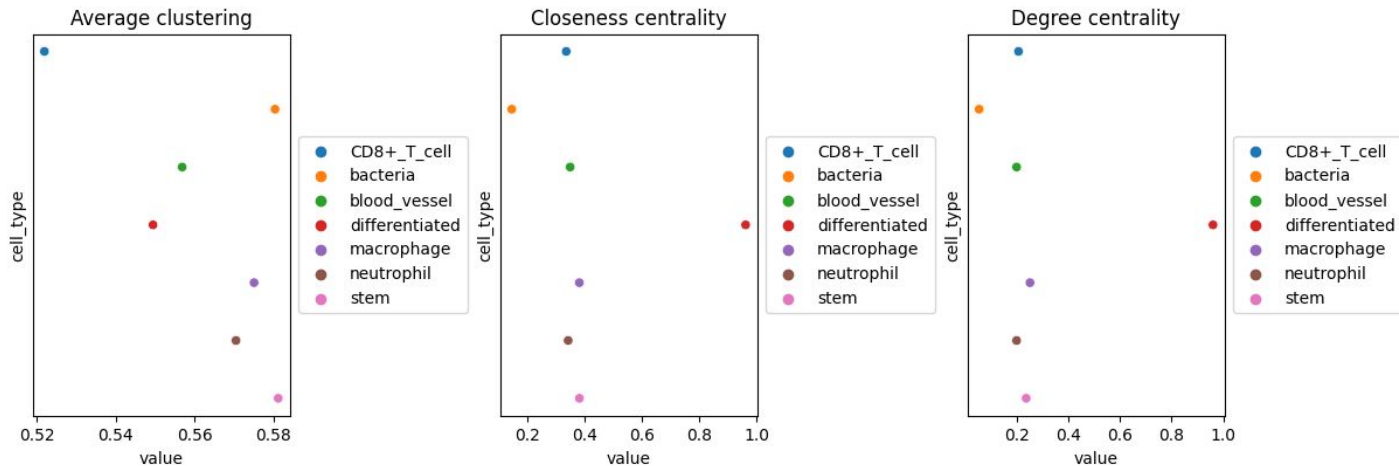
**PhysiCell.org**

 **@PhysiCell**

# squidpy & centrality\_scores

two lines of code:

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



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

[https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl\\_repl\\_programming.ipynb](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

**PhysiCell Project**

**PhysiCell.org**

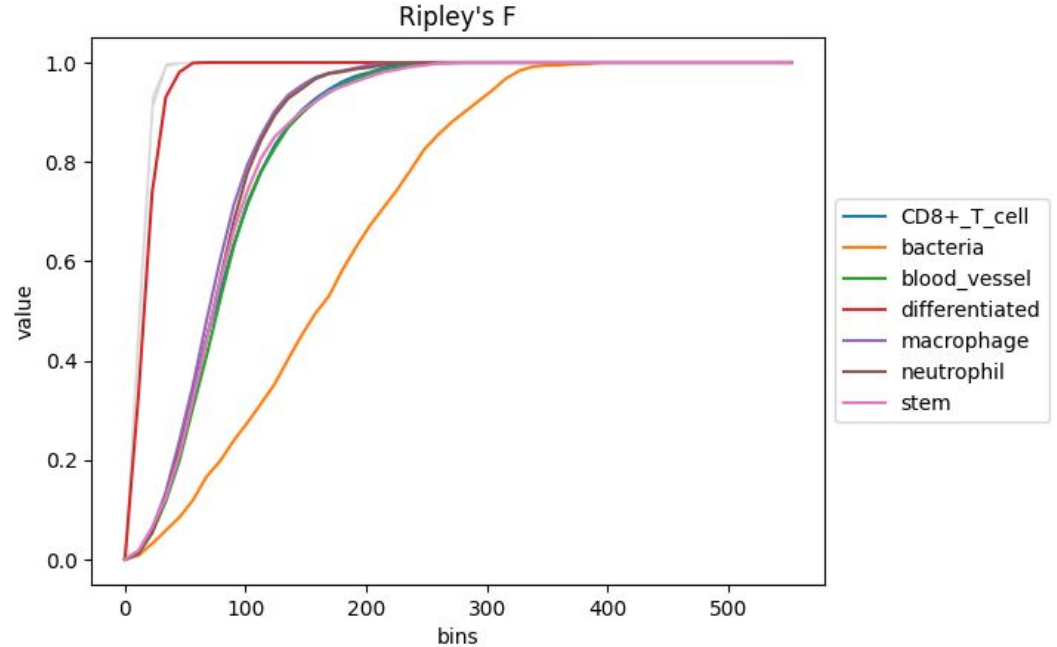
**@PhysiCell**

# squidpy & Ripley's statistics

for point processes

two lines of code:

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



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

[https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl\\_repl\\_programming.ipynb](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

**PhysiCell Project**

**PhysiCell.org**

**@PhysiCell**

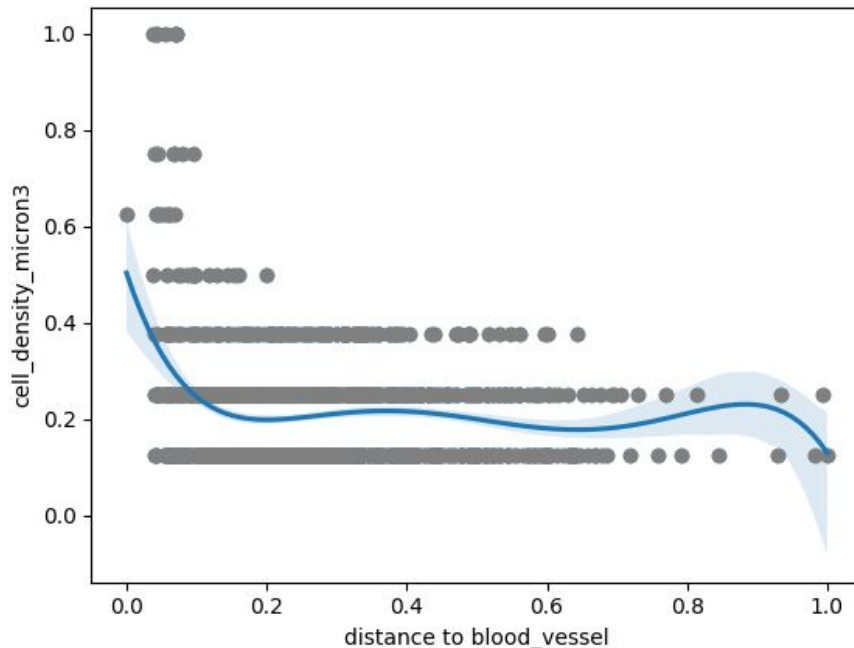


# squidpy & var\_by\_distance

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

	I	pval_norm	var_norm	pval_norm_fdr_bh		C	pval_norm	var_norm	pval_norm_fdr_bh
position_vectorlength	0.992993	0.0	0.000273	0.0	position_vectorlength	0.009540	0.0	0.000273	0.0
quorum	0.934202	0.0	0.000273		quorum	0.030820	0.0	0.000273	0.0
debris	0.927604	0.0	0.000273		debris	0.062550	0.0	0.000273	0.0
pro-inflammatory	0.870561	0.0	0.000273		toxin	0.080504	0.0	0.000273	0.0
toxin	0.853246	0.0	0.000273		pro-inflammatory	0.128708	0.0	0.000273	0.0
damage_halfmax	0.603295	0.0	0.000273		cell_cell_repulsion_strength	0.340886	0.0	0.000273	0.0
cell_cell_repulsion_strength	0.603295	0.0	0.000273		relative_max_damage_death	0.340886	0.0	0.000273	0.0
relative_max_damage_death	0.603295	0.0	0.000273		resource_chemotactic_sensitivities	0.340886	0.0	0.000273	0.0
resource_chemotactic_sensitivities	0.603295	0.0	0.000273		damage_halfmax	0.340886	0.0	0.000273	0.0
migration_speed	0.595514	0.0	0.000273		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](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

**PhysiCell Project**

**PhysiCell.org**

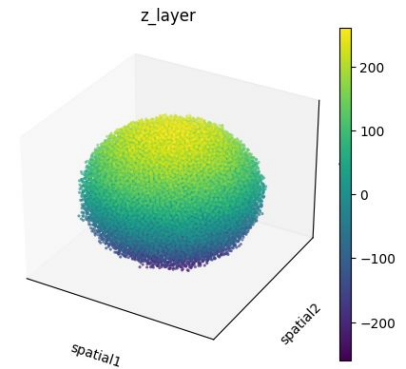
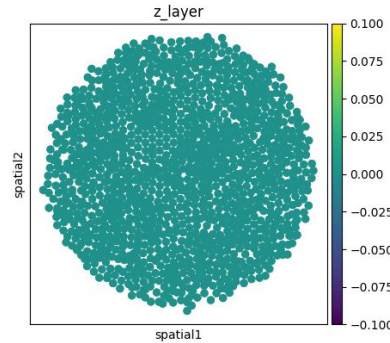
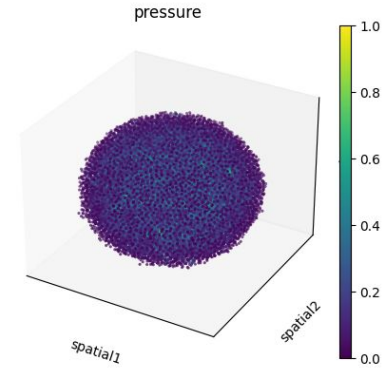
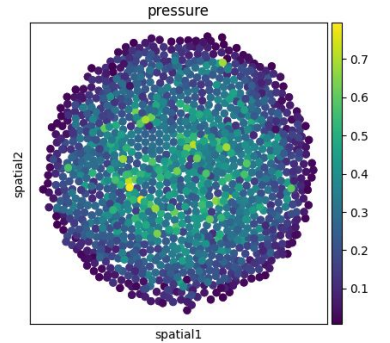
**@PhysiCell**

# squidpy & z-stacks

one line of code:

- `sq.pl.spatial_scatter(ann3d ...)`
- `sc.pl.embedding(ann3d ...)`

this is physicell output!



**LUDDY**

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

[https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl\\_repl\\_programming.ipynb](https://github.com/elmbeech/physicelldataloader/blob/master/man/jupyter/pcdl_repl_programming.ipynb)

**PhysiCell Project**

**PhysiCell.org**

**@PhysiCell**

# 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
- ...

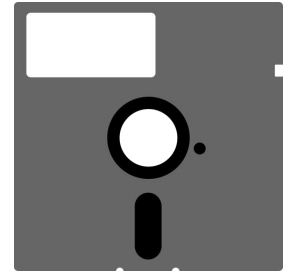
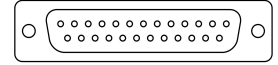
# pcdl homepage

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

The screenshot shows the GitHub repository page for `elmbeech/physicelldataloader`. The repository is a fork of `PhysiCell-Tools/python-loader`. It has 6 branches, 26 tags, and 297 commits. The file list includes `.github/workflows`, `man`, `pcdl`, `test`, `.gitignore`, `LICENSE`, `README.md`, `data_timeseries_2d.targz`, `data_timeseries_3d.targz`, and `pyproject.toml`. The `README.md` file is selected, showing the `physicell data loader` logo and an abstract. The abstract states: "physicelldataloader (pcdl) provides a platform independent, python3 based, pip installable interface to load output, generated with the PhysiCell agent based modeling framework, into python3." The right sidebar shows the repository is a python3 physicell data output loader, has a BSD-3-Clause license, 6 stars, 12 forks, and 1 release dated 2023-07-23. The languages section shows Python at 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 field  
**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.

