bioRxiv preprint doi: https://doi.org/10.1101/2020.11.16.385377; this version posted November 17, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC 4.0 International license.

Tysserand - Fast reconstruction of spatial networks from bioimages

Alexis Coullomb*,[1] and Vera Pancaldi[1]

[1]Centre de Recherches en Cancérologie de Toulouse, Toulouse, France

16/11/2020

Abstract

Motivation: Networks provide a powerful framework to analyze spatial omics experiments. However, we lack tools that integrate several methods to easily reconstruct networks for further analyses with dedicated libraries. In addition, choosing the appropriate method and parameters can be challenging.
Summary: We propose *tysserand*, a Python library to reconstruct spatial networks from spatially resolved omics experiments. It is intended as a common tool where the bioinformatics community can add new methods to reconstruct networks, choose appropriate parameters, clean resulting networks and pipe data to other libraries.
Availability: tysserand software and tutorials with a Jupyter notebook to reproduce the results are available at https://github.com/VeraPancaldiLab/tysserand
Contact: vera.pancaldi@inserm.fr
Supplementary information: Supplementary data are available at *Bioarxiv* online.

1 Introduction

Recent technologies have made it possible to produce phenotypic data at the resolution of single cells (or higher) in intact sample slices, both at the levels of proteins [1, 2, 3, 4] or mRNA [5, 6, 7, 8]. Taking advantage of spatial information is determinant for revealing the biology of healthy organs and dissecting the complex processes involved in cancer, such a tumor progression and response to treatments[9, 10].

Existing spatial omics analysis libraries such as trendsceek[11], SpatialDE[12] and PySpacell[13] use marked point processes theory. Another fruitful approach is to represent tissues as networks, where nodes are cells and edges are interactions between cells which are established through physical contact. Network theory is already used for spatial analysis in the Python Spatial Analysis Library (PySAL)[14] for geospatial data science, and PySpacell, based on PySAL, provides 3 methods to reconstruct networks: k-nearest neighbors, radial distance neighbors and cell contact neighbors. However, due to its dependence on PySAL, it is not ideally suited to test other network reconstruction methods and PySAL methods do not scale well with big datasets, such as the ones typically produced after nuclei segmentation in Whole Slide Images investigated by anatomopathologists in a medical setting. Moreover, the choice of a reconstruction method and parameters can be hard, and the potential to use the reconstructed network with other dedicated network analysis libraries remains a priority. Here, following the Unix philosophy[15] according to which *programs do one thing and do it well* and *programs work together*, we present tysserand, a Python library to reconstruct spatial networks starting from object positions (cells, nuclei, ...) or image segmentation results. We aim at encouraging the centralization of efforts in network construction from the bioinformatics community in one place, to promote integration of new reconstruction methods, providing algorithms for parameters selection, network processing to remove reconstruction artifacts, computational performance improvement and the addition of interfaces with external network analysis libraries such as NetworkX[16], iGraph[17] or Scanpy[18] for single-cell data analysis.

2 Materials and methods

Tysserand can consider two types of alternative inputs (Figure 1). First, an M×2 array of the M cells's x/y coordinates or, second, a "segmentation image" with integers ranging from 0 to K representing K segmented areas in a microscopy image, with 0 values indicating the background. For the coordinates array input, 3 methods are already available to reconstruct networks, based on the Scipy[19] library implementation: k-nearest neighbors (knn), radial distance neighbors (rdn) and Delaunay triangulation. We think Delaunay triangulation is best suited to represent tissues and interactions between contacting cells, whereas the rdn method is more appropriate to model interactions by diffusing chemicals,

*Corresponding author: alexis.coullomb@inserm.fr

bioRxiv preprint doi: https://doi.org/10.1101/2020.11.16.385377; this version posted November 17, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC 4.0 International license.

as already noticed by PySpacell authors[13]. For inputs consisting in cell segmentation images, we implemented the area contact neighbors method. It leverages the scikit-image[20] library to detect for each area which neighbors are in direct contact or closer than a given distance. The tysserand library provides simple visualization utilities to choose appropriate parameters for each network construction method and tools to clean the resulting networks from typical artifacts, such as very long edges between nodes on the border of samples after Delaunay triangulation (Figure S2).

Internally, tysserand adopts simple and efficient representations of networks to allow rapid prototyping of new network construction or cleaning methods. A network is represented by 2 arrays: a first M×2 array for nodes coordinates (that are the center of segmented objects if a segmentation image is provided as input), and an L×2 array to represent the L edges between nodes indicated by their index in the first coordinates array. Finally, tysserand can convert networks into formats used by specialized libraries such as NetworkX, iGraph and Scanpy.

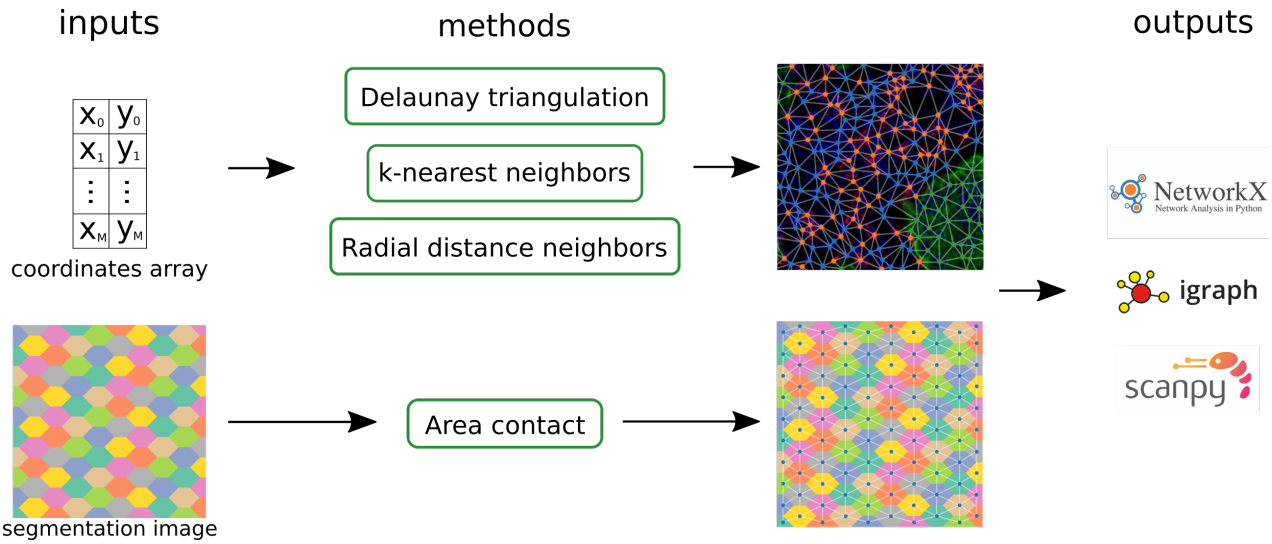

Figure 1: tysserand can take as inputs an array of nodes positions or an image resulting from segmentation processing. 4 methods are implemented for now to reconstruct spatial networks. The resulting networks can then be exported to formats compatible with libraries dedicated to network analysis or using networks in their downstream analyses.

3 Results

To qualitatively assess the differences between the Delaunay triangulation, knn and rdn, we used a set of nodes consisting of manually marked nuclei in a single tile from a multiplex immuno-fluorescence (mIF) Whole Slide Image of a lung-cancer biopsy sample (data available on the GitHub repository, description in supplementary materials, Figure S1). We applied each of these methods and upon visual inspection of the resulting networks we observed the following (Figure S2): 1) The Delaunay triangulation with edge trimming produces a network that looks similar to what we can expect from a tissue, i.e., most edges link contacting cells, as visible on the mIF image (Figure S3 b); 2) rdn produces excessively connected areas where the density of nodes is high; 3) knn produces a network with missing edges where we could expect them based on cell contacts, as well as edges passing through neighboring cells, which is not suitable to model interactions dependent on direct physical contact. We thus conclude that Delaunay triangulation is the most suitable method for these biological tissue images.

Finally, we compared the performance of the tysserand Delaunay triangulation and the mathematically equivalent Voronoi tessellation implemented in PySAL on randomly generated sets of node positions. The PySAL implementation, being based on shape objects, does not produce long edges artifacts. Since the tysserand Delaunay method does produce this type of artifacts, we also benchmarked tysserand's method including automated edge trimming, which can slow down the performance but is necessary to obtain results that are comparable to PySAL (description in supplementary material). Across all sizes of node sets, the tysserand implementation of the Delaunay triangulation, including the automated network artifact removal option, is always at least 42 times faster than PySAL's (Table 1, Figure S7). It is likely that the lower performance of the PySAL library is due to the use of 'shape' objects, which are common and important in geographical sciences but compromise the algorithm scalability. Since biological image applications often do not require the concept of 'shape', considerable improvements in scalability can be obtained using tysserand. The speed up provided by tysserand is valuable in the range of several tens of thousands of nodes, which is often the size of reconstructed networks from tissue sample Whole Slide Images.

bioRxiv preprint doi: https://doi.org/10.1101/2020.11.16.385377; this version posted November 17, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC 4.0 International license.

size	tysserand	tysserand cleaned	PySAL	speed up	speed up cleaned
100	0.000768	0.000966	0.114	148	118
300	0.0022	0.00224	0.485	221	217
1000	0.00642	0.00648	0.689	107	106
3000	0.0224	0.0211	2.17	96.8	103
10000	0.248	0.0879	7.57	30.5	86.1
30000	0.478	0.48	22.9	48	47.7
100000	1.33	1.85	79	59.3	42.7

Table 1: Mean execution time (s) of tysserand and PySAL for Delaunay/Voronoi network reconstruction methods

4 Conclusion

tysserand can reconstruct spatial networks from different inputs, such as sets of node positions or segmented areas, and the resulting networks can be further processed with dedicated network analysis libraries. It already implements 4 common network reconstruction methods as well as tools to facilitate the choice of parameters for network construction and artifact removal. tysserand Delaunay triangulation is faster than the PySAL equivalent method and scales better with larger datasets, which is important for the analysis of multiple tissue samples. We hope the bioinformatic community will be willing to participate in the implementation of new methods for more accurate and domain specific spatial network reconstruction to advance the field of bioimage processing.

Acknowledgements

We thank Professor Pierre Brousset and the Imag'IN facility at IUCT Oncopole, Toulouse for providing the multiplex immuno-fluorescence images.

Funding

This work was funded by INSERM; Fondation Toulouse Cancer Santé and Pierre Fabre Research Institute as part of the Chair of Bioinformatics in Oncology of the CRCT.

References

[1] Charlotte Giesen, Hao AO Wang, Denis Schapiro, Nevena Zivanovic, Andrea Jacobs, Bodo Hattendorf, Peter J Schüffler, Daniel Grolimund, Joachim M Buhmann, Simone Brandt, et al. Highly multiplexed imaging of tumor tissues with subcellular resolution by mass cytometry. *Nature methods*, 11(4):417–422, 2014.

[2] Michael Angelo, Sean C Bendall, Rachel Finck, Matthew B Hale, Chuck Hitzman, Alexander D Borowsky, Richard M Levenson, John B Lowe, Scot D Liu, Shuchun Zhao, et al. Multiplexed ion beam imaging of human breast tumors. *Nature medicine*, 20(4):436–442, 2014.

[3] Jia-Ren Lin, Mohammad Fallahi-Sichani, Jia-Yun Chen, and Peter K Sorger. Cyclic immunofluorescence (cycif), a highly multiplexed method for single-cell imaging. *Current protocols in chemical biology*, 8(4):251–264, 2016.

[4] Jennifer Eng, Guillaume Thibault, Shiuh-Wen Luoh, Joe W Gray, Young Hwan Chang, and Koei Chin. Cyclic multiplexed-immunofluorescence (cmif), a highly multiplexed method for single-cell analysis. In *Biomarkers for Immunotherapy of Cancer*, pages 521–562. Springer, 2020.

[5] Simone Codeluppi, Lars E Borm, Amit Zeisel, Gioele La Manno, Josina A van Lunteren, Camilla I Svensson, and Sten Linnarsson. Spatial organization of the somatosensory cortex revealed by osmfish. *Nature methods*, 15(11):932–935, 2018.

[6] Kok Hao Chen, Alistair N Boettiger, Jeffrey R Moffitt, Siyuan Wang, and Xiaowei Zhuang. Spatially resolved, highly multiplexed rna profiling in single cells. *Science*, 348(6233), 2015.

[7] Chee-Huat Linus Eng, Michael Lawson, Qian Zhu, Ruben Dries, Noushin Koulena, Yodai Takei, Jina Yun, Christopher Cronin, Christoph Karp, Guo-Cheng Yuan, et al. Transcriptome-scale super-resolved imaging in tissues by rna seqfish+. *Nature*, 568(7751):235–239, 2019.

[8] Sanja Vickovic, Gökcen Eraslan, Fredrik Salmén, Johanna Klughammer, Linnea Stenbeck, Denis Schapiro, Tarmo Äijö, Richard Bonneau, Ludvig Bergenstråhle, José Fernandéz Navarro, et al. High-definition spatial transcriptomics for in situ tissue profiling. *Nature methods*, 16(10):987–990, 2019.

bioRxiv preprint doi: https://doi.org/10.1101/2020.11.16.385377; this version posted November 17, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC 4.0 International license.

[9] Lili Cao, Xiaofang Che, Xueshan Qiu, Zhi Li, Bowen Yang, Shuo Wang, Kezuo Hou, Yibo Fan, Xiujuan Qu, and Yunpeng Liu. M2 macrophage infiltration into tumor islets leads to poor prognosis in non-small-cell lung cancer. *Cancer management and research*, 11:6125, 2019.

[10] Sebastian Lundgren, Jacob Elebro, Margareta Heby, Björn Nodin, Karin Leandersson, Patrick Micke, Karin Jirström, and Artur Mezheyeuski. Quantitative, qualitative and spatial analysis of lymphocyte infiltration in periampullary and pancreatic adenocarcinoma. *International Journal of Cancer*, 146(12):3461–3473, 2020.

[11] Daniel Edsgärd, Per Johnsson, and Rickard Sandberg. Identification of spatial expression trends in single-cell gene expression data. *Nature methods*, 15(5):339–342, 2018.

[12] Valentine Svensson, Sarah A Teichmann, and Oliver Stegle. Spatialde: identification of spatially variable genes. *Nature methods*, 15(5):343–346, 2018.

[13] France Rose, Luca Rappez, Sergio H Triana, Theodore Alexandrov, and Auguste Genovesio. Pyspacell: A python package for spatial analysis of cell images. *Cytometry Part A*, 97(3):288–295, 2020.

[14] Sergio J Rey and Luc Anselin. Pysal: A python library of spatial analytical methods. In *Handbook of applied spatial analysis*, pages 175–193. Springer, 2010.

[15] Eric S Raymond. Basics of the unix philosophy. *Als Online-Dokument: http://www. faqs. org/docs/artu/ch01s06. html# id2877537*, 2003.

[16] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[17] Gabor Csardi, Tamas Nepusz, et al. The igraph software package for complex network research. *InterJournal, complex systems*, 1695(5):1–9, 2006.

[18] F Alexander Wolf, Philipp Angerer, and Fabian J Theis. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology*, 19(1):15, 2018.

[19] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

[20] Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.