

# 10th International Conference on Computational Social Science

## Network Diffusion — Framework to Simulate Spreading Processes in Complex Networks

**Michał Czuba**<sup>1</sup>, Mateusz Nurek<sup>1</sup>, Damian Serwata<sup>1</sup>, Yu-Xuan Qi<sup>2</sup>,  
Mingshan Jia<sup>2</sup>, Katarzyna Musiał<sup>2</sup>, Radosław Michalski<sup>1</sup>, Piotr  
Bródka<sup>1</sup>

<sup>1</sup>Wrocław University of Science and Technology

<sup>2</sup>University of Technology Sydney

20.07.2024



# Agenda

- 1 Motivation
- 2 Key Features of `network-diffusion`
- 3 Example - Custom Model
- 4 Resources and References
- 5 Limitations of `network-diffusion`

# Motivation

Spreading phenomena are one of the problems considered by a network science. They can be observed in various areas like: dynamics of political opinions, marketing campaigns, spread of epidemics, computer viruses, etc.



Figure: Artistic representation of a social network.<sup>1</sup>

<sup>1</sup>Source: [www.uniroma3.it/articoli/seminario-biased-opinion-dynamics-when-the-devil-is-in-the-details-138122](http://www.uniroma3.it/articoli/seminario-biased-opinion-dynamics-when-the-devil-is-in-the-details-138122)

Analytical approaches to such phenomena are often insufficient for large graphs, prompting researchers to use computational methods, i.e. simulations.

Analytical approaches to such phenomena are often insufficient for large graphs, prompting researchers to use computational methods, i.e. simulations.

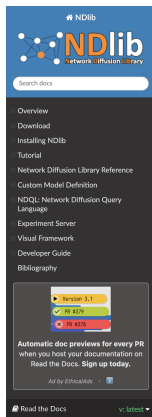


Thus, many tools were developed to address that issue, allow researchers to get rid of the need to start their experiments from scratch, and enhance the reproducibility of results.

# Motivation

Here is a bunch of tools that help in simulating diffusion processes in networks:

- **NDlib**[5],
- **GLEaMviz**[2],
- **SimInf**[6],
- **STEM**[3],
- **EpiModel**[4],
- **Sisspread**[1],
- ...



Docs > NDlib - Network Diffusion Library

[Edit on GitHub](#)

## NDlib - Network Diffusion Library

**NDlib** is a Python software package that allows to describe, simulate, and study diffusion processes on complex networks.

Date	Python Versions	Main Author	GitHub	pypl
2023-09-08	>=3.6	<a href="#">Giulio Rossetti</a>	<a href="#">Source</a>	<a href="#">Distribution</a>

If you use **NDlib** as support to your research consider citing:

G. Rossetti, L. Milli, S. Rinzivillo, A. Sirbu, D. Pedreschi, F. Giannotti. "NDlib: a Python Library to Model and Analyze Diffusion Processes Over Complex Networks" Journal of Data Science and Analytics. 2017. DOI:10.1007/s41060-017-0086-6 (pre-print available on [arXiv](#))

G. Rossetti, L. Milli, S. Rinzivillo, A. Sirbu, D. Pedreschi, F. Giannotti. "NDlib: Studying Network Diffusion Dynamics" IEEE International Conference on Data Science and Advanced Analytics, DSAA, 2017.

## NDlib Dev Team

Name	Contribution
<a href="#">Giulio Rossetti</a>	Library Design/Documentation
<a href="#">Lertiza Milli</a>	Epidemic Models
<a href="#">Alina Sirbu</a>	Opinion Dynamics Model
<a href="#">Salvatore Rinzivillo</a>	Visual Platform
<a href="#">Mathijs Majer</a>	Continuous Model

Figure: NDlib's website.

However, if we consider more complex...

...network models (e.g. multilayer graphs)...

...diffusion models (e.g. spreading of coexisting processes)...

**...a gap among the available toolkits emerges.**

# Motivation

My research group works on problems related to multilayer/temporal networks, spreading phenomena, influence maximisation, data streams, etc.



# Motivation

My research group works on problems related to multilayer/temporal networks, spreading phenomena, influence maximisation, data streams, etc.



During the research, we produced a code that can be reused as a boilerplate for further experiments, with interfaces allowing anyone to quickly implement a custom spreading model.

# Motivation

My research group works on problems related to multilayer/temporal networks, spreading phenomena, influence maximisation, data streams, etc.



During the research, we produced a code that can be reused as a boilerplate for further experiments, with interfaces allowing anyone to quickly implement a custom spreading model.



With regard to the observed gap, we decided to merge and wrap the code into a library to share it with the community as a small contribution to make experiments easier.

# Key Features of `network-diffusion`

Functionalities of the library:

- end-to-end workflow for simulations of spreading phenomena,

# Key Features of `network-diffusion`

Functionalities of the library:

- end-to-end workflow for simulations of spreading phenomena,
- predefined (basic) spreading models,

# Key Features of `network-diffusion`

Functionalities of the library:

- end-to-end workflow for simulations of spreading phenomena,
- predefined (basic) spreading models,
- an interface for implementing custom discrete spreading models,

# Key Features of `network-diffusion`

Functionalities of the library:

- end-to-end workflow for simulations of spreading phenomena,
- predefined (basic) spreading models,
- an interface for implementing custom discrete spreading models,
- support for temporal networks (CogSNet and snapshot-based),

Functionalities of the library:

- end-to-end workflow for simulations of spreading phenomena,
- predefined (basic) spreading models,
- an interface for implementing custom discrete spreading models,
- support for temporal networks (CogSNet and snapshot-based),
- support for the multilayer networks,

Functionalities of the library:

- end-to-end workflow for simulations of spreading phenomena,
- predefined (basic) spreading models,
- an interface for implementing custom discrete spreading models,
- support for temporal networks (CogSNet and snapshot-based),
- support for the multilayer networks,
- most of centrality metrics extended to multilayer networks.



# Key Features of `network-diffusion`

Environmental requirements/features:

- released under MIT licence,
- support for Linux, macOS, and Windows<sup>2</sup>,
- Python  $\geq 3.10$  compatibility,
- C snippets in the CogSNet module to speed-up computations,
- NetworkX compatibility.

---

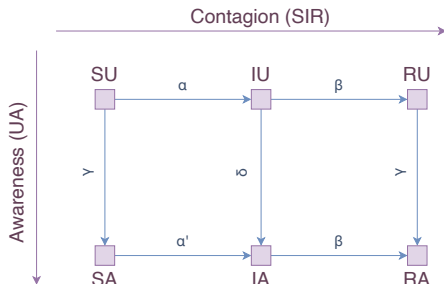
<sup>2</sup>Although we develop and use it mostly on Unix OSs

# Example - Custom Model

In this example we will consider a joint disease-awareness model (SIR-UA) that can be used e.g. to assess the effectiveness of various countermeasures against the spread of COVID-19:

**Table:** Transition weights with explanation.

Symbol	Description
$\alpha$	probability of infection for unaware agents
$\alpha'$	probability of infection for aware agents
$\beta$	probability of recovery
$\gamma$	probability of awareness for uninfected agents
$\delta$	probability of awareness for infected agents



**Figure:** State and transition graph for SIR-UA.

# Example - Custom Model

To create an own spreading model, we have to extend the abstract base class (`nd.models.BaseModel`) by implementing:

- a field `_compartmental_graph`,
- a field `_seed_selector`,
- a method `determine_initial_states`,
- a method `agent_evaluation_step`,
- a method `network_evaluation_step`,
- methods `__str__` and `get_allowed_states`.

The following entities (with implemented method `update_network`) are utilised in the class `nd.Simulator` which orchestrates the experiment.

# Example - Custom Model

Let's model this problem with `network-diffusion`!

The library can be installed with:

```
pip install network-diffusion
```

We have also published other useful resources:

- PyPI website: [pypi.org/project/network-diffusion](https://pypi.org/project/network-diffusion),
- GitHub page: [github.com/anty-filidor/network\\_diffusion](https://github.com/anty-filidor/network_diffusion),
- Reference guide: [network-diffusion.readthedocs.io](https://network-diffusion.readthedocs.io),
- Paper with more use-cases presented: [arxiv.org/abs/2405.18085](https://arxiv.org/abs/2405.18085).

There is still some work to do...

- computational performance can be better,
- we are limited to discrete spreading models,
- we cannot model agent's complex states (e.g. internal/external),
- much less predefined spreading models than in NDlib,
- library maintenance guaranteed up to my graduation 😊.

- [1] Fabian Alvarez et al. “Sispread: A Software to Simulate Infectious Diseases Spreading on Contact Networks”. In: Methods of information in medicine 46 (2007), p. 19. DOI: [10.1055/s-0038-1627827](https://doi.org/10.1055/s-0038-1627827). URL: <https://doi.org/10.1055/s-0038-1627827>.
- [2] Wouter Van den Broeck et al. “The GLEaMviz Computational Tool, a Publicly Available Software to Explore Realistic Epidemic Spreading Scenarios at the Global Scale”. In: BMC Infectious Diseases 11.1 (Feb. 2011), p. 37. ISSN: 1471-2334. DOI: [10.1186/1471-2334-11-37](https://doi.org/10.1186/1471-2334-11-37). URL: <https://doi.org/10.1186/1471-2334-11-37>.
- [3] Judith Douglas et al. “STEM: An Open Source Tool for Disease Modeling”. In: Health security (Aug. 2019). DOI: [10.1089/hs.2019.0018](https://doi.org/10.1089/hs.2019.0018). URL: <https://doi.org/10.1089/hs.2019.0018>.

# References II

- [4] Samuel M Jenness, Steven M Goodreau, and Martina Morris.  
“EpiModel: an R package for mathematical modeling of infectious disease over networks”. In: Journal of statistical software 84 (2018). DOI: 10.18637/jss.v084.i08. URL: <https://doi.org/10.18637/jss.v084.i08>.
- [5] Giulio Rossetti et al. “NDlib: A Python Library to Model and Analyze Diffusion Processes Over Complex Networks”. In: Int. J. Data Sci. Anal. 5.1 (Feb. 2018), pp. 61–79. ISSN: 2364-4168. DOI: 10.1007/s41060-017-0086-6. URL: <https://doi.org/10.1007/s41060-017-0086-6>.
- [6] Stefan Widgren et al. “SimInf: An R Package for Data-Driven Stochastic Disease Spread Simulations”. In: Journal of Statistical Software 91.12 (2019), pp. 1–42. DOI: 10.18637/jss.v091.i12. URL: <https://doi.org/10.18637/jss.v091.i12>.



# Thank you for your attention!



Scan this QR code to reach the main page of the project 😊