Exploring Network Topologies from Routing Optimization Problems

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Maria Daniela Leite de Souza, M. Sc.
aus Barbalha, Brasilien

Tübingen
2024

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation: 31.07.2024

Dekan: Prof. Dr. Thilo Stehle
1. Berichterstatterin: Dr. Caterina De Bacco
2. Berichterstatter: Prof. Dr. Georg Martius

Disclaimer

This thesis uses the *Clean Thesis* template style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Acknowledgements

While I could fill an entire thesis just to express my heartfelt gratitude to everyone who has supported me from the very beginning of my studies, I would like to particularly highlight those who guided my scientific journey throughout the past years of the PhD.

First, I am deeply grateful to my advisor, Caterina De Bacco, for her continuous support and guidance, both academically and personally. Her patience, advice and constant encouragement have truly meant a lot to me. Many thanks for the countless hours of discussions, for the dedicated assistance and support, and for helping me finding the path of my research. Her encouragement kept me going especially when I struggled to believe in myself. And thanks for building a fantastic research group in Tübingen. Being part of PIO has been an incredible opportunity!

I extend my gratitude to Georg Martius for reviewing this thesis, and to Philipp Hennig, Gabriele Schweikert, Michael Menth and Claire Vernade for kindly accepting to be in my thesis advisory committee. I am grateful again to Philipp and Georg, alongside with Anna Levina, to be part of my examination committee.

I am also grateful to all my scientific collaborators, whose insights and expertise have been invaluable for this thesis. I look forward to continuing our collaborations in the future.

I am deeply grateful to all the friends that I made in Tübingen, many of whom have become like my second family, and to all the people who have enriched my life with new learnings every day. The coffee chats, dinners, cooking experiences, table soccer/volleyball/beachvolleyball matches, trips, and the shared memories have all been invaluable, and I will always carry these good memories with me. I could never have reached this point without any of you.

Finally, I am endlessly grateful my family for their incredible support and encouragement throughout this journey. Despite all the difficulties and barriers that life gave us, they taught me to always see the beauty and bright side of life, to stay positive, to look forward and pursue my dreams. This achievement is as much theirs as it is mine.

Com todo o meu amor e gratidão, dedico a vocês, meus pais.

Abstract

While routing optimization problems are relevant in various domains, solving them computationally is often challenging. This thesis explores Optimal Transport (OT) principles to solving routing optimization problems, focusing on a biologically-inspired dynamical formulation. We first explore how to translate the solutions of a dynamical system of equations into meaningful network topologies, while preserving the important optimality properties of these solutions. Our graph extraction method provides a valuable tool to help practitioners bridging the gap between the abstract mathematical principles of OT and the more interpretable language of network theory.

In addition to addressing the graph extraction problem, we explore how to find community structures inspired by the problem of measuring the OT Wasserstein distance, combined with the notion of geometric curvature in a graph. The proposed approach allows for tuning between different transportation regimes, while controlling the information shared between nodes' neighborhoods. We evaluated our algorithm's performance with various synthetic and real networks, achieving comparable or superior results to other OT-based methods on synthetic data, while identifying communities that more accurately represent node metadata in real data.

Last, we study how our graph extraction method can be extended to designing and optimizing urban transportation networks. We use only a limited number of origins and destination points to generate network structures directly from a continuous space, representing the initial stage of development of a transportation infrastructure. By tuning one parameter, we show how our method can simulate a range of different subway, tram and train networks that can be further used to suggest possible improvements in terms of relevant transportation properties.

In summary, this thesis investigates various applications of optimal transport principles for solving routing optimization problems. We provide a tool to explore networks that are solutions from a mathematical formulation of optimal transport theory, showing how it can be explored to design more efficient urban transportation networks, and including a measure of graph similarity which is then applied to a geometric-based community detection method.

Zusammenfassung

Obwohl Routing-Optimierungsprobleme in verschiedenen Bereichen relevant sind, ist deren rechnerische Lösung oft eine Herausforderung. Diese Arbeit untersucht Prinzipien des Optimalen Transports (OT) zur Lösung von Routing-Optimierungsproblemen und konzentriert sich dabei auf eine biologisch inspirierte dynamische Formulierung. Zunächst untersuchen wir, wie die Lösungen eines dynamischen Gleichungssystems in sinnvolle Netzwerktopologien übersetzen werden können, wobei die wichtigen Optimalitätseigenschaften dieser Lösungen erhalten bleiben. Unsere Methode zur Grafextraktion bietet ein wertvolles Werkzeug, die Lücke zwischen den abstrakten mathematischen Prinzipien des OT und der besser interpretierbaren Sprache der Netzwerktheorie zu überbrücken.

Neben der Behandlung des Grafextraktionsproblems untersuchen wir, wie durch das Messen der OT-Wasserstein-Distanz inspirierte Gemeinschaftsstrukturen gefunden werden können, kombiniert mit dem Konzept der geometrischen Krümmung in einem Graphen. Der vorgeschlagene Ansatz ermöglicht das Feinabstimmen zwischen verschiedenen Transporttypen, während die geteilten Informationen zwischen den Nachbarschaften der Knoten kontrolliert werden. Wir haben die Leistung unseres Algorithmus mit verschiedenen synthetischen und realen Netzwerken bewertet und dabei vergleichbare oder überlegene Ergebnisse zu anderen OT-basierten Methoden bei synthetischen Daten erzielt, während wir Gemeinschaften identifizierten, die die Knotendaten in realen Daten genauer darstellen.

Abschließend untersuchen wir, wie unsere Methode zur Grafextraktion auf die Gestaltung und Optimierung städtischer Verkehrsnetze erweitert werden kann. Wir verwenden nur eine begrenzte Anzahl von Ausgangs- und Zielpunkten, um Netzwerkstrukturen direkt aus einem kontinuierlichen Raum zu erzeugen, was die Anfangsphase der Entwicklung einer Verkehrsinfrastruktur darstellt. Durch die Manipulationen eines Parameters zeigen wir, wie unsere Methode eine Vielzahl unterschiedlicher U-Bahn-, Straßenbahn- und Zugnetze simulieren kann, die weiter verwendet werden können, um mögliche Verbesserungen in Bezug auf relevante Verkehrseigenschaften vorzuschlagen.

Zusammenfassend untersucht diese Arbeit verschiedene Anwendungen der Prinzipien des Optimalen Transports zur Lösung von Routing-Optimierungsproblemen. Wir bieten ein Werkzeug, um Netzwerke zu erkunden, die Lösungen einer mathematischen Formulierung der Optimalen Transporttheorie sind, und zeigen, wie es zur Gestaltung effizienterer städtischer Verkehrsnetze genutzt werden kann.

Contents

1 Introduction

1.1 Motivation

Optimal Transport (OT) problems aim at answering the fundamental question of finding the optimal way to move a certain amount of resources from one point to another. The first mathematical formulation of it has its origins around the 18th century by the French mathematician Gaspard Monge. He proposed a way to calculate the most effective strategies of moving large amounts of dust from a starting to a final configuration with the least amount of effort [1]. Around 140 years later, Kantorovich proposed a relaxed version of the original Monge's problem, aiming to distribute goods from factories to warehouses and stores effectively [2]. Nowadays, these problems appear in multiple contexts and are relevant in diverse fields, from data science and machine learning [3, 4] to economics [5] and biology [6, 7].

In the context of network science, OT problems also play an important role. For instance, human-built systems like ground transportation, airport routes, communication or water supply networks are designed to minimize the costs of construction and maintenance, while ensuring the optimality of the moving resources. This is also frequently observed in natural transportation networks, such as vascular systems in plants, river or fungi networks [8, 9]. In both cases, finding a mathematical formulation and an efficient computational framework for simulating these problems is fundamental to gain a better understanding of how such diverse systems work and evolve over time.

Despite the importance and wide range of applications for these kinds of optimization problems, solving them is still very challenging computationally, in particular, due to the high dimensionality of the data or the complexity of the mathematical formulations to model them accurately. Numerous approximation techniques were proposed to overcome such complexities, like semi-discrete methods, which reduce the number of variables to be optimized by approximating continuous variables to discrete ones [10, 11], or linear programming [12, 13], which imposes heuristics traffic constraints or capacity limitations in the network.

In this thesis, we explore solving routing optimization problems with an OT-based approach, focusing on a dynamical formulation. This borrows inspiration from the dynamical networks built by the *Physarum Polycephalum* (PP), a single-cell slime mold that has the remarkable ability of building efficient tubular networks while foraging for food resources [14]. The slime has intrigued researchers with its capacity to solve complex optimization problems through its adaptive strategy of connecting food sources while reallocating nutrients in its body. The proposed models to mimic the slime behavior have been used across different domains, such as finding the shortest-path in a maze [15], nutrient balancing in distributed systems [16], or the traveling salesman problem [17]. Here, we consider as a starting point the so-called *Dynamic Monge-Kantorovich* (DMK) approach, a recent result that bridges an OT problem formulation (the known Monge-Kantorovich equations), with a generalized dynamics of the PP [18, 19]. This approach generalizes the original PP discrete dynamic proposed by Tero *et al.* [14] to a continuous domain, mapping a computationally hard optimization problem into a more accessible model via the asymptotic behavior of a system of dynamic partial differential equations.

An important aspect of the DMK model is the possibility of recovering solutions that resemble optimal network configurations for both congested and branched transportation problems, via a numerically accessible and computationally efficient framework. Starting with these solutions, we will first explore the challenging problem of translating them into meaningful network structures, while preserving the important optimality properties. We will then use this model to explore two parallel research directions: the problem of designing efficient urban transportation systems and that of finding community structures.

On one hand, as transportation systems impact us daily, understanding the underlying principles that reveal how real transportation networks (like subways, buses, trains, etc.) evolve over time is fundamental for planning and optimizing future development of such structures. Most of the available approaches in the literature consider optimizing already existing structures, not necessarily looking at the stage where there is no network at all. This is an important scenario because it involves the initial design of a transportation network, where decisions can be made to make them more efficient, resilient and sustainable over time.

On the other hand, the problem of detecting communities in a network is challenging and relevant in many real-world applications like social, biological or ecological networks [20, 21, 22]. The existing algorithms for community detection in the literature highly depend on the type of data and nature of the network, so choosing the most accurate method depends on the objective of the analysis. In our case, inspired by the problem of measuring the OT Wasserstein distance, and combining this with the notion of geometric curvatures in a graph, we propose a method that tunes between different transportation regimes to find the number of communities automatically from the data, in contrast to approaches that require this as input parameter.

In summary, this thesis explores different applications of optimal transport principles for solving routing optimization problems. It presents a new framework designed to translate a more abstract mathematical formulation of optimal transport theory to the more concrete and understandable language of network theory. By exploring how these principles can be leveraged to designing more efficient urban transportation networks, and including a measure of graph similarity which is then applied to a geometric-based community detection method, we contribute to the broader landscape of network analysis and optimization.

1.2 Outline

This thesis is divided into two main parts. The first one is dedicated to an introduction of the DMK model and related concepts, together with the relevant notations and motivation for the main applications. The second part contains the main scientific contributions and discussions of their impact, as well as future research directions. This part is based on two published works and one preprint currently under review in a peer-reviewed journal. For each chapter in this part (excluding Chapter 6) there is a complementary appendix containing additional examples and discussions of the algorithmic implementations.

1.2.1 Detailed list of contributions

The main results detailed in this thesis are based on two published works with different collaborators, and one preprint (currently under review in a peer-reviewed journal). Each of the chapters from Chapter 3 to Chapter 5 are closely based on the original works, with only minor changes made in comparison with the original texts.

Chapter 3 introduces a method for extracting optimal network topologies from routing optimization problems. This chapter is based on the peer-reviewed journal paper:

- Diego Baptista*, Daniela Leite*, Enrico Facca, Mario Putti, and Caterina De Bacco. "Network extraction by routing optimization." Scientific reports 10, no. 1 (2020): 20806. *equal contribution

I was first author, together with Diego Baptista. My contributions included proposing insights for the graph extraction problem, looking at the different representations one could make from the solutions of the continuous routing optimization problem, and how to efficiently filter possible redundancies from this representation. I also contributed to comparing the main advantages of our formulation with using techniques for graph extraction from images, which were important to improve our approach. Together with my coauthors, we proposed a rigorous evaluation metric to measure the performance of the various extracted graphs. I also contributed to the numerical implementation of our pipeline, building, for instance, a set of configurations for sources and sinks, and developing an efficient algorithm that is open-source and available online. All authors contributed with writing and analyzing the results. I also worked on extending some experiments to incorporate the feedback from reviews, which considerably improved the quality of the paper.

Chapter 4 introduces an OT method for community detection by combining the approach presented in Chapter 3 with the geometric concept of Ollivier–Ricci curvature. The novelty of it comes from enabling the possibility of tuning among different transportation regimes to better control the shared information between the nodes' neighborhoods. This chapter is based on the peer-reviewed journal paper:

- Daniela leite*, Diego Baptista*, Abdullahi A. Ibrahim, Enrico Facca, and Caterina De Bacco. "Community detection in networks by dynamical optimal transport formulation." Scientific Reports 12, no. 1 (2022): 16811.
  *equal contribution

I was first author, with Diego Baptista. In addition, I took the leadership of this project in managing the collaboration between the different authors, as each had specific tasks which required management. The motivation came from a problem I encountered while working on another project, where we needed to measure the similarity between two given graphs that visually seemed to have the same topology. While this is a very natural question when working with graphs, it is nevertheless challenging to solve it. We then worked on conceptualizing the main ideas and developing the theory. Together with the other co-authors, we then worked on not only generating different network structures for synthetic and semi-synthetic data, but also on collecting real network data from opened datasets. We performed together several experiments and implemented various other methods to compare with ours, discussing and analyzing the results. All authors contributed to the writing, editing and incorporating feedback from the reviewers.

Both papers were published at *Scientific Reports*, a peer-reviewed and open-access journal published by the Nature Portfolio since 2011. This journal publishes original research works from across all areas of the natural sciences, psychology, medicine and engineering. It is considered the 5^{th} most cited journal in the world, receiving significant attention in policy documents and in the media. They provide rigorous objective and constructive peer-review, led by the same editorial and ethical guidelines as other journals from the Nature Portfolio, ensuring that all published research maintains originality, scientific rigor and the highest quality standards. They have adopted an open-access policy, giving researchers high visibility for their work.

Chapter 5 presents an application of the network extraction model introduced and detailed in Chapter 3 to real public transportation systems. The main idea behind this paper was to use the approach introduced in Chapter 3 in real-world urban transportation networks. With only a little information in input, i.e. a small set of origin and destination points, we generate structures that reveal a surprising similarity to those of public transportation networks. We show that our method can simulate a range of different subway, tram and train networks. Our model can be further used to suggest possible improvements in terms of relevant transportation properties, ultimately contributing to the development of more sustainable and efficient public transportation networks. This chapter is based on the following peer-reviewed journal paper:

- Daniela Leite and Caterina De Bacco. "Similarity and economy of scale in urban transportation networks and optimal transport-based infrastructures". In: *Nature Communications* 15.1 (2024), p. 7981

In this work I was the first author. Together with the other author, we actively discussed the method and details of the numerical implementation. I contributed by collecting, preprocessing and analyzing the data from multiple transportation systems around the world, collecting from several data sources and providing an extensive analysis in terms of network properties. This paper significantly changed after the first round of reviews. We started looking only at small networks like the Dublin rail or the Rome subway, but then extended to bigger and more complex structures such as the New York subway. An interesting point raised by the reviewers was to include information from real population data and land usage, so we looked at the density of points of interest from some of the studied cities as a reference to build a transportation infrastructure. I worked on these additional examples that substantially improved the quality of our results. Overall, given the major changes made after the round of review, the paper is substantially improved. Another important contribution of this paper was a principled quantitative measure of similarity between two networks, which can be used beyond transportation networks. I also co-wrote this paper and collaborated to have a numerically efficient implementation of our algorithm.

The peer-reviewed journal *Nature Communications* is an open-access and multidisciplinary, dedicated to publishing high-quality research in all areas of the biological, health, physical, chemical, Earth, social, mathematical, applied, and engineering sciences. This journal aims to publish papers that represent important advances of significance to specialists within each field. It is editorially independent and makes its own decisions independently of other Nature journals, allowing authors to decide where to submit their manuscripts based on the relevance and potential impact of their work. It also operates as an open-access model, so all content is freely accessible to readers.

Finally, in Chapter 6 we present a more detailed discussion of the presented methods, and explore future research directions that can be explored from our contributions.

An additional published work during my PhD that is not included in this thesis is:

- Abdullahi Adinoyi Ibrahim, Daniela Leite, and Caterina De Bacco. "Sustainable optimal transport in multilayer networks." Physical Review E 105.6 (2022): 064302.

In this paper I was second author. Together with the other authors, we actively discussed the method, numerical implementation and results. I contributed with cleaning the code, planning the experiments and analyzing the data. I also contributed to the writing, proofreading and incorporating feedback from the multiple rounds of review. In this paper we start from a different scenario compared to the ones within the scope of this thesis: the congested multilayer network case, where multiple types of transportation are treated as a different layer in a network. We proposed an approach also based on optimal transport theory to efficiently

distribute passengers along layers that are more carbon-efficient than roads, like rails for instance. We show that our approach is more carbon efficient than shortest-path minimization. As this paper deals with a different optimization dynamics and network type, it will not be explored here.

This work was published at Physical Review E (PRE), a peer-reviewed, open-access and multidisciplinary journal established in 1993. It is part of the *Physical Review* family, recognized as a leading journal in the interconnected fields of statistical, nonlinear, biological, and soft matter physics. PRE is also considered a venue for high-quality work in traditional and emerging research areas, making it an essential resource for multiple disciplines.

2 The Dynamic Monge-Kantorovich formulation

The slime mold has fascinating capabilities of decision-making despite not having a brain. For instance, it always chooses the path that requires the least amount of energy to get a sufficient amount of food when placed in mazes. This behavior has intrigued biologists and mathematicians for many years, which then formulated the adaptation rules to describe the slime dynamics, later using such optimization abilities to simulate the design of the railways of Tokyo. The idea of applying Optimal Transport to understand how organisms like the slime reallocate nutrients within their bodies comes from the observations of these resource optimization usages to maximize survival.

In this chapter, we describe the model that is used as a basis for most of the works developed along this thesis: the Dynamic Monge-Kantorovich (DMK) formulation, both on its continuous and discrete versions. We start describing the dynamics of the Physarum Policephalum (PP) in Section 2.1 and how it connects to the continuous DMK model in Section 2.2. We then extend this problem to the discrete domain in section Section 2.3, briefly describing how we translate its solutions to the networks language in Section 2.2.2. We will explore this model from Chapters 3 to 5.

2.1 The *Physarum Polycephalum* dynamics

The *Physarum Polycephalum (PP)* is a slime mold that has the ability of finding shortest paths when placed in a maze with food sources. Its veins must be efficiently routed to balance between transportation efficiency, capacity and resilience to damage [14, 24]. The dynamics behind such behavior was first proposed by Tero et al. [14] and later exploited by Bonifaci et al. [25], that proved why the path described by the mold is, in fact, the shortest one. In this section we briefly describe this model and its relevance to our work.

In the works of Tero et.al. [14], the slime is modeled as an electrical network with time-varying resistors, where its tubular channels are represented as edges from an undirected graph, here denoted as $G = (V, E)$, and the places where the food is located as s_0 and s_1 (the source and sink, respectively). For each edge $e = (i, j)$ there is a corresponding conductivity D_{ij}(with $(i, j) \in E$), which can be described as the edge capacities to transport a flow, and for each node j there is an associated potential (or hydrostatic pressure) p_j ($j \in V$). The model assumes Poiseuille flow (i.e., $D_{ij} \propto$ radius of the pipe). The goal is to find the optimal distribution of the pair (D_{ij}, p_j) that satisfies the following conditions:

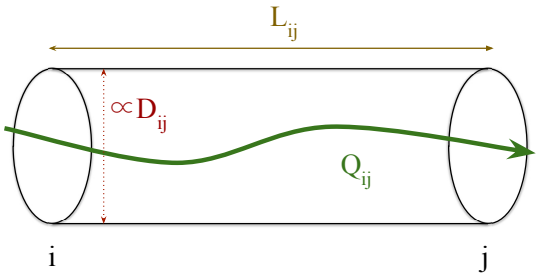


Fig. 2.1: Illustration of the tubular channels of the slime as an edge $e = (i, j)$ of a graph G.

$$\sum_i Q_{ij}(t) = f_j = \begin{cases} 1, & \text{if } j = s_0, \\ -1, & \text{if } j = s_1, \\ 0, & \text{otherwise;} \end{cases} \tag{2.1a}$$

$$Q_{ij}(t) = \frac{D_{ij}(t)}{L_{ij}}(p_i(t) - p_j(t)), \tag{2.1b}$$

$$\frac{d}{dt}D_{ij}(t) = g(|Q_{ij}|) - D_{ij}(t), \tag{2.1c}$$

$$D_{ij}(t = 0) > 0. \tag{2.1d}$$

The system can be explained as follows. The first equation is the Kirchhoff's law, stating that the sum of in and out flows is equal to zero for all nodes except s_0 and s_1 (the flow is regulated by the node function f_j). The second equation defines the flow rate Q_{ij}, where each edge flow is proportional to the discrete gradient of the pressure $(p_i - p_j)$ multiplied by conductivity coefficient D_{ij}, and inversely proportional to its length L_{ij}. The third equation governs the system dynamics, ensuring the natural optimal allocation of fluxes. As the conductivity evolves (positively or negatively) over time, the diameter of the pipes will adjust accordingly to accommodate larger fluxes. Notice that the conductivity is not only proportional to the absolute value of the flow but it is also subject to a decay, ensuring that the diameter of that edge is decreased in case the flow is low. The last equation provides the initial configuration of the system.

It was proven in [25] that when $g(x) = x$, the distribution of D_{ij} is the shortest path in asymptotic times for a general graph G. They also show that under the condition of

$$\sum_{j \in V} f_j = 0, \tag{2.2}$$

the PP dynamics is equivalent to an Optimal Transportation Problem (OTP) in G. We can simply reframe this problem as that of finding Q_{ij} such that

$$\min_{Q_{ij}} \sum_{(i,j) \in E} Q_{ij} L_{ij}, \tag{2.3a}$$

$$\sum_{(i,j) \in E} Q_{ij} = f_j, \forall j \in V. \tag{2.3b}$$

Moreover, given specific general assumptions on the graph structure, the solution of the PP dynamics in Equation (2.1) converges to a stationary solution Q^*_{ij}, that is also a solution of the OTP in G.

2.2 Dynamic Monge-Kantorovich

The model proposed by Facca et al. [18] removes the graph structure of Equation (2.1) and defines the problem in a continuous domain as an open and bounded subset Ω of $\mathbb{R}^2$. It considers the case where $g(x) = x$ and the continuous version of the forcing function in Equation (2.2), an analogous of the problem described in [25]. Their model reads as follows. Consider the system of partial differential equations:

$$-\nabla \cdot (\mu(t,x)\nabla u(t,x)) = f^+(x) - f^-(x) \tag{2.4a}$$

$$\frac{\partial_\mu(t,x)}{\partial t}\mu(t,x) = (\mu(t,x)\nabla u(t,x))^\beta - \mu(t,x) \tag{2.4b}$$

$$\mu(0,x) = \mu_0(x) > 0, \tag{2.4c}$$

complemented by zero Neumann boundary conditions [26]. Notice that $\nabla = \nabla_x$. In this new scenario, the problem consists of finding a pair of functions (μ, u) satisfying Equation (2.4), also known as the continuous Dynamic Monge-Kantorovich model (DMK). This model introduces an exponent β that allows the system to reach different configurations for the transportation problem: when $\beta < 1$ the converged solution is such that the transported mass avoids congestion; when $\beta > 1$ the solution has a setting from a branch transportation problem; $\beta = 1$ is shortest path-like, since the mass goes in a straight line from the source $f^+(x)$ to the sink $f^-(x)$ (in this case, we have the exact continuous equivalent of Equation (2.1)).

But how does the PP dynamical problem connects with DMK?
As proposed in [18], under certain conditions, the solutions for a mold in the maze are equivalent to an optimal transportation problem. The authors conjectured that for the case of $\beta = 1$ in the regime of large times ($t \to \infty$) this solution pair (μ, u) converges to the pair (μ^*, u^*), where μ^* is the transport density and u^* the Kantorovich potential, a solution pair of the Optimal Transport Problem (OTP). Having a closer look at Equation (2.1) and Equation (2.4), we notice that the first connection reads as

$$D_{ij}(t) \longrightarrow \mu(t,x). \tag{2.5}$$

This implies that from a discrete to a continuous case, the edges' diameter is described by an initial distribution μ, therefore describing how the edges are changing on time until the solution reaches convergence. As for the mass flow, the amount of mass going from node i to node j is stored in the quantity Q_{ij}. In the continuous scenario, this is given by the product of the density distribution $\mu(t,x)$ with the gradient of the potential $\nabla u(x,t)$, which leads to

$$Q_{ij}(t) \longrightarrow \mu(t,x)\nabla u(t,x). \tag{2.6}$$

The third correspondence is given by the positions of sources and sinks. The sources or sink nodes (i or j), with associated hydrostatic pressure p_i and p_j, are equivalent to the source and sink functions $f(x)$ (respectively, $f^+(x)$ and $f^-(x)$),

$$f_{ij} \longrightarrow f(x), \quad \text{with} \quad \int_\Omega f dx = 0. \tag{2.7}$$

The last correspondence refers to the boundary conditions of each dynamics. While in the graph, the injection term for initial times must be positive, in the continuous case, the amount of mass that needs to be sent from a source to a sink, must also be positive. From a more intuitive perspective, if the initial distributions are equal to zero, this means that no mass is being transported. In this case,

$$Q_{ij}(t=0) \longrightarrow \mu(0,x) = \mu_0(x) > 0. \tag{2.8}$$

Among the novelties of this model, the authors proposed in [18] an algorithm for solving this dynamics (from now labeled as the *DMK solver*) in which it was possible to emulate the results found by [14] for the discrete formulation of the PP. By tuning some parameters like the exponent β, they were able not only to model different problems belonging to the optimal transportation theory, but also to solve them using their mentioned tools. Each time that the solver is executed, we find a sequence of functions $(\mu_t, u_t)_t$, where μ_t is the transport density at time t and u_t is the potential at a given time t. If this sequence converges, its limit is denoted as (μ^*, u^*).

2.2.1 The Lyapunov cost

In [18] the authors conjectured that in the regime of asymptotic times, the solution of the system Equation (2.4) when $\beta = 1$ is the unique minimizer of a functional that takes the form of

$$\mathcal{L}(\mu, u) := \mathcal{E}(\mu, u) + \mathcal{M}(\mu), \tag{2.9}$$

where the first term is interpreted as an energy functional and the second one a mass functional. They show that the optimal transport density (μ^*) is the unique minimizer of a functional with such form, and that the minimum equals the Wasserstein distance between the source f^+ and the sink f^-, with cost equal to the Euclidean distance. Under appropriate regularity assumptions, it can be shown that the equilibrium solution is a minimizer of the following form of Equation (2.9):

$$\mathcal{L}(\mu) = \frac{1}{2}\int_\Omega \mu |\nabla u|^2 dx \; + \; \int_\Omega \left(\frac{\beta}{2-\beta}\right) \mu^{\frac{2-\beta}{\beta}} dx. \tag{2.10}$$

We empirically interpret this functional as comprising two main parts. The first one is the transport cost, assumed to be proportional to the total dissipated energy transporting the mass from f^+ to f^-. The second part is the cost of building the transport infrastructure, assumed to be a nonlinear function of the total capacity of the system. Notice that when $0 < \beta \leq 1$ the second term takes a convex form, penalizing the concentration of the transport density μ, a scenario that reflects congested transportation problems [27, 28]. Instead, when $1 < \beta < 2$ the second term becomes concave, favoring the μ to concentrate in branches. Notice that each value assumed by β represents an optimal solution for that specific configuration, so every time we change the forcing and the value of β, we obtain a new optimal solution.

2.2.2 Outputs of the DMK solver and the Graph Extraction Problem

As the DMK solutions depend on the values of β, we show in Figure 2.2 some examples for the different configurations of the transportation problem. We will show examples for different configurations later in Chapter 3.

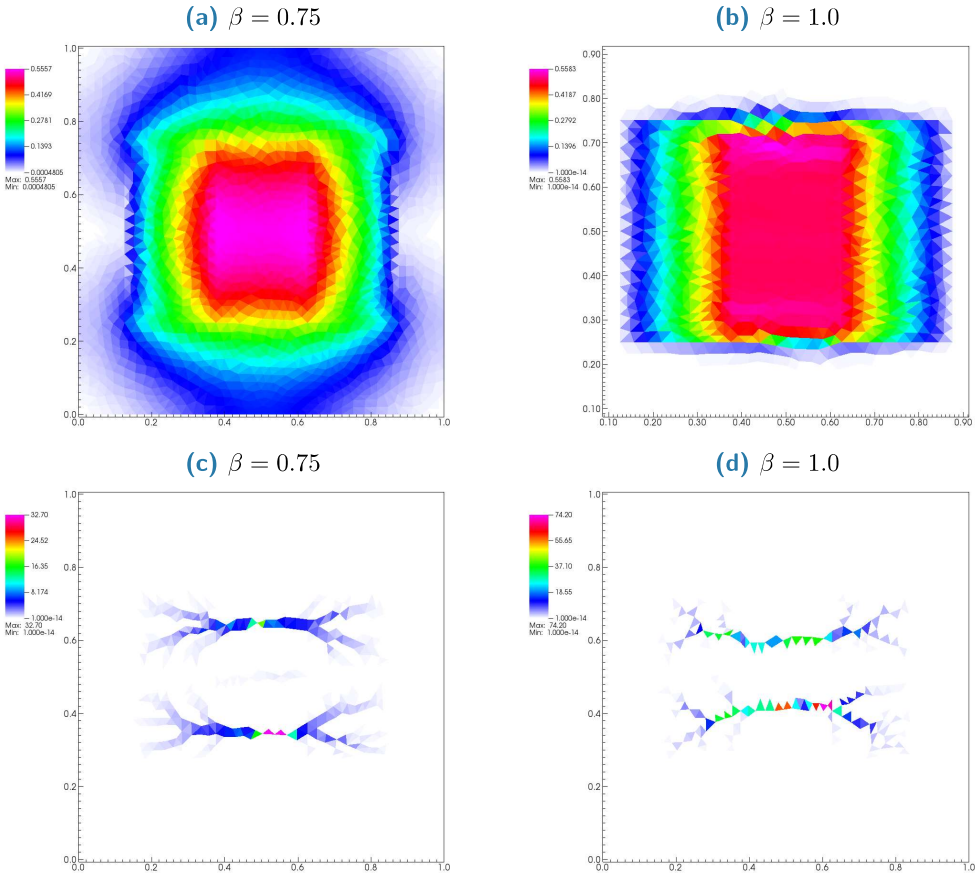


Fig. 2.2: Different DMK outputs. We consider the same source/sink pairs and change the value of β to obtain different structures. In (a) the converged solution is such that the mass avoids congestion, while in (b) it converges to shortest-path. In (c) and (d), we notice the emergence of branches. The colors indicate the distribution of μ^* in the grid.

The main idea behind using the DMK model here is to exploit the robustness of the topologies of its solutions. We start with translating the solution pair (μ^*, u^*) into a network framework, as detailed in Chapter 3, aiming on understanding more general transportation problems from the network theory perspective. We will then explain how to map as a graph the solution pair from the continuous scenario (μ^*, u^*), like the ones in Figure 2.2. We will give in Chapter 3 a theoretical description of how to extract the graphs given the solutions from the DMK solver as inputs, in order to justify our adopted graph representation, highlighting the main features of the DMK solver.

2.3 Solving OTP in graphs

In this section, we introduce a discrete version of the continuous DMK-Solver (we will refer to it as the discrete-DMK-Solver). This formulation is a natural way of looking at solving the

optimal transport problem in undirected weighted graphs. In fact, it was proven in [29] that solving the PP dynamical problem will lead to the solution of the Basis Pursuit (BP) problem (also known as *optimal transshipment problem*).

We define the signed incidence matrix $\mathbf{B} \in \mathbb{M}_{N\times M}$ of a weighted graph $G = (V, E, W)$, with entries $B_{ie} = \pm 1$ if the edge e has node i as start/end point, or 0 otherwise; $N = |V|$ the nodes and $M = |E|$ the edges; the vector of edge lengths $\ell = \{\ell_e\}_e$; and $\mathbf{f}$ a N-dimensional vector of source and sink values with entries satisfying $\sum_{i\in V} f_i = 0$. This last condition is the discrete version of the forcing term from the continuous case (i.e., $f(x)$).

We then define $\mu_e(t)$ as the edge conductivities, and $u_i(t)$ as the potential on the nodes, which have the same interpretation as in the continuous problem, but are now vectors instead of functions. We can rewrite Equation (2.4) as

$$f_i = \sum_e B_{ie} \frac{\mu_e(t)}{\ell_e} \sum_j B_{ej} u_j(t) \quad , \tag{2.11}$$

$$\mu_e'(t) = \left[\frac{\mu_e(t)}{\ell_e} |\sum_j B_{ej}\, u_j(t)| \right]^\beta - \mu_e(t) \quad , \tag{2.12}$$

$$\mu_e(0) > 0 \quad , \tag{2.13}$$

where $|\cdot|$ is the absolute value element-wise. In this new system, the first equation corresponds to Kirchoff's law, the second one is the discrete dynamics, with β_d a parameter controlling for different routing optimization mechanisms (similarly to the β in Equation (2.4)); the last equation is the initial condition.

This system has an interesting theoretical correspondence, as its equilibrium point corresponds to the minimizer of a clearly-defined Lyapunov functional interpreted as the global energy, in an analogous form of Equation (2.10), that takes the following discrete form

$$\mathcal{L}_\beta(\mu(t)) = \frac{1}{2} \sum_e \mu_e(t) \left(\frac{1}{\ell_e} \sum_j B_{ej}\, u_j(\mu(t)) \right)^2 \ell_e + \frac{1}{2} \sum_e \frac{\mu_e(t)^{P(\beta)}}{P(\beta)} \ell_e \,, \tag{2.14}$$

where $P(\beta) = 2 - \beta/\beta$ and $u(\mu(t))$ is a function implicitly defined as the solution of Equation (1.11). The first term is the energy dissipated during transport, which one can interpret as the costs of operating the network, while the second term represents the cost of building the infrastructure.

The equilibrium point of $\mu_e(t)$ is stationary, and for $\beta_d = 1$ it acts also as the global minimizer of Equation (2.14) due to its convexity. When $\beta_d > 1$ the energy is not convex, thus in general the functional will present several local minima towards which the dynamics will be attracted. The case $\beta_d < 1$ encourages trajectories to spread through the network.

Similarly to the graph pre-extraction step, the filtering is also valid for networks that are not necessarily solutions of Equation (2.4). It can be applied to more general inputs if defined on a discrete space, for instance, images. In fact, this idea was used in [30] to develop an efficient algorithm for extracting networks from images, as an alternative to more standard image processing techniques, which often require domain-specific knowledge.

The discrete-DMK-Solver will be used in Chapter 3 as a fundamental step to filter possible redundancies of the networks obtained from the graph representation of the continuous solutions (it will be later explored in Section 3.4.1).

3 Network Extraction from routing optimization

Routing optimization is a relevant problem in many contexts. Solving directly this type of optimization problem is often computationally intractable. Recent studies suggest that one can instead turn this problem into one of solving a dynamical system of equations, which can instead be solved efficiently using numerical methods. This results in enabling the acquisition of optimal network topologies from a variety of routing problems. However, the actual extraction of the solution in terms of a final network topology relies on numerical details which can prevent an accurate investigation of their topological properties. In fact, in this context, theoretical results are fully accessible only to an expert audience and ready-to-use implementations for non-experts are rarely available or insufficiently documented. In particular, in this framework, final graph acquisition is a challenging problem in-and-of-itself.

In this chapter, we introduce a method to extract network topologies from dynamical equations related to routing optimization under various parameters' settings. Our method is made of three steps: first, it extracts an optimal trajectory by solving a dynamical system in a continuous domain. Then it pre-extracts a network with a pre-extraction step. Finally, it filters out potential redundancies. Remarkably, we propose a principled model to address the filtering in the last step, and give a quantitative interpretation in terms of a transport-related cost function. This principled filtering can be applied to more general problems such as network extraction from images, thus going beyond the scenarios envisioned in the first step.

Overall, the algorithm described in details along this chapter allows practitioners to easily extract optimal network topologies by combining basic tools from numerical methods, optimization and network theory. Thus, we provide an alternative to manual graph extraction which allows a grounded extraction from a large variety of optimal topologies. The analysis of these may open up the possibility to gain new insights into the structure and function of optimal networks. We provide an open source implementation of the code at the Github repository `https://github.com/Danielaleite/NextRout`.

3.1 Introduction

Investigating optimal network topologies is a relevant problem in several contexts, with applications varying from transportation networks [31, 32, 33, 34], communication systems [35, 36, 37], biology [38, 39] and ecology [40, 41, 42]. Depending on the specified objective function and set of constraints of a routing optimization problem [43], optimal network topologies can be determined by different processes ranging from energy-minimizing tree-like structures ensuring steeper descent through a landscape as in river basins [40] to the opposite scenario of loopy structures that favor robustness to fluctuations and damage as in leaf venation [44, 42], the retina vascular system [45, 46] or noise-cancelling networks [37]. In many applications, optimal networks can arise from an underlying process defined on a continuous space rather than a discrete network as in standard combinatorial optimization routing problems [47, 48, 49, 50].

Optimal routing networks try to move resources by minimizing the transportation cost. This cost may be taken to be a function of the traveled distance, such as in Steiner trees, or proportional to the dissipated energy, such as optimal channel networks or resistance networks. The common denominator of these configurations is that they have a tree-like shape, i.e., optimal routing networks are loopless [31, 51]. Recent developments in the mathematical theory of optimal transport [41, 43] have proved that this is indeed the case and that complex fractal-like networks arise from branched optimal transport problems [52]. While the theory starts to consolidate, efficient numerical methods are still in a pre-development stage, in particular in the case of branched transport, where only a few results are present [53, 54], reflecting the obstacle that all these problems have an NP-hard genesis.

Recent promising results [18, 19] map a computationally hard optimization problem into finding the long-time behavior of a system of dynamic partial differential equations, the so-called Dynamic Monge-Kantorovich (DMK) approach, which is instead numerically accessible, computationally efficient, and leads to network shapes that resemble optimal structures [27].

Working in discretized continuous space, and in many network-based discretizations such as lattice-like networks as well, requires the use of threshold values for the identification of active network edges. This has the main consequence that there might be no obvious final resulting network, an output that would be trivial when starting from an underlying search space formed by predefined selected network structures. For example, the output of a numerically discretized (by, e.g., the Finite Element method) routing optimization problem in a 2D space is a real-valued function on a set of (x, y) points defined on a grid or triangulation, which already has a graph structure. Despite the underlying graph, this grid function contains numerous side features, such as small loops and dangling vertices, that prevent the recognition of a clear optimal network structure. Obtaining this requires a suitable identification of vertices and edges that should contain the optimal network properties embedded in the underlying continuous space. In other words, the output of a routing optimization problem in continuous space carries unstructured information about optimality that is hard to interpret in terms of network properties.

Extracting a network topology from this unstructured information would allow, on one hand, better interpretability of the solution and enable the comparison with networks resulting from discrete space. On the other hand, the use of tools from network theory to investigate the properties related to optimality, for instance, to perform clustering or classification tasks based on a set of network features. One can frame this problem as that of properly compressing the information contained in the "raw" solution of a routing problem in continuous space into an interpretable network structure while preserving the important properties connected to optimality. This is a challenging task, as compression might result in losing important information. The problem is made even more complex because one may not know in advance what are the relevant properties for the problem at hand, a knowledge that could help drive the network extraction procedure. This is the case for any real network, where the intrinsic optimality principle is elusive and can only be speculated about by observing trajectories, an approach adopted for instance when processing images in biological networks [55, 56, 57, 58].

Several works have been proposed to tackle domain-specific network extraction. These methods include using segmentation techniques on a set of image pixels to extract a skeleton [56, 55, 59] that is then converted into a network; a pipeline combining different segmentation algorithms building from OpenCV, [60] which is made available with an intuitive graphical interface [61]; graph-based techniques [62] that sample junction-points from input images; methods that use deep convolutional neural networks [63] or minimum cost path computations [64] to extract road networks from images. These are mainly using image processing

techniques as the input is an image or photograph, which might not necessarily be related to a routing optimization problem.

In this chapter, we propose a new approach for the extraction of network topologies and build a protocol to address this problem. This can take in input the numerical solution of a routing optimization problem in continuous space as described in [18, 19, 27] and then processes it to finally output the corresponding network topology in terms of a weighted adjacency matrix. However, it can also be applied to more general inputs, such as images, which may not necessarily come from the solution of an explicit routing transportation problem. Specifically, our work features a collection of numerical routines and graph algorithms designed to extract network structures that can then be properly analyzed in terms of their topological properties.

The extraction pipeline consists in a sequence of three main algorithmic steps:

(i) compute the steady state solutions of the DMK equations (dmkS);

(ii) extract the optimal network solution of the routing optimization problem (*graph pre-extraction*);

(iii) filter the network removing redundant structure (*graph filtering*).

While for this work we test and demonstrate our algorithm on routing scenarios coming from DMK, which constitute our main motivation, we remark that only the first step is specific to these, whereas the last two steps are applicable beyond these settings. The graph pre-extraction step consists of a set of rules aiming at building a network from an input that is not explicitly a topological structure made of nodes and edges. The filtering step is based on a principled mathematical model inspired by that of the first step, which leads to an efficient algorithmic implementation. Our network filter has a nice interpretation in terms of a cost function that interpolates between an operating cost and an infrastructure one, contrarily to common approaches used in image processing for filtering, which often relies on heuristics.

A successful execution will return a representation of the network in terms of an edge-weighted undirected network. The resulting weights are related to the optimal flow, solution of the routing problem. Once the network is obtained, practitioners can deploy arbitrary available network analysis software [65, 66, 67, 68] or custom-written scripts to investigate properties of the optimal topologies. While our primary goal is to provide a framework and tool to solve the research question of how to extract network topologies resulting from routing optimization problems in continuous space or any other image containing a network structure, we also aim at encouraging non-expert practitioners to automatically extract networks from such problems or from more general settings beyond that. Thus we make available an open-source algorithmic implementation at `https://github.com/Danielaleite/Nextrout`.

3.2 The routing optimization problem

In this section, we describe the main ideas and refer to the frameworks introduced in Chapter 2. We briefly introduce the dynamical system of equations corresponding to the DMK routing optimization problem as proposed by Facca et al. [18, 19, 27]. In these works, the authors first generalize the discrete dynamics of the slime mold *Physarum Polycephalum* (PP) to a continuous domain; then they conjecture that, like its discrete counterpart, its solution tends to an equilibrium point which is the solution of the Monge-Kantorovich optimal mass transport [69] as time goes to infinity.

We denote the space where the routing optimization problem is set as $\Omega \in \mathbb{R}^n$, an open bounded domain that compactly contains $f(x) = f^+(x) - f^-(x) \in \mathbb{R}$, the forcing function, describing the flow generating sources $f^+(x)$ and sinks $f^-(x)$. It is assumed that the system is isolated, i.e., no fluxes are entering or exiting the domain from the boundary. This imposes the constraint $\int_\Omega f(x)dx = 0$ to ensure mass balance. It is assumed that the flow is governed by a transient Fick-Poiseuille type flux $q = -\mu\nabla u$, where $\mu(t,x), u(t,x)$ are called *conductivity* or *transport density* and *transport potential*, respectively.

The continuous set dynamical Monge-Kantorovich (DMK) equations were introduced in Equation (2.4), with Equation (2.4a) stating the spatial balance of the Fick-Poiseuille flux and complemented by no-flow Neumann boundary conditions; Equation (2.4b) enforcing the system dynamics in analogy with the discrete PP model and Equation (2.4c) providing the initial configuration of the system. The parameter β captures different routing transportation mechanisms. A value of $\beta < 1$ enforces optimal solutions to avoid traffic congestion; $\beta = 1$ is shortest path-like; while $\beta > 1$ encourages consolidating the flow so to use a smaller amount of network-like infrastructure, and is related to branched transport [70, 41]. Within a network-like interpretation, qualitatively, $\mu(x,t)$ describes the capacity of the network edges [71]. Thus, its initial distribution $\mu_0(x)$ describes how the initial capacities are distributed.

In this work, solving the routing optimization problem consists of finding the steady state solution $(\mu^*, u^*) : \Omega \to \mathbb{R}_{\geq 0} \times \mathbb{R}$ of Equation (2.4a), i.e. $(\mu^*(x), u^*(x)) = \lim_{t\to+\infty}(\mu(t,x), u(t,x))$. Numerical solution of the above model can be obtained by means of a double discretization in time and space [18, 19, 27]. The resulting solver (from now referred as DMK-Solver) has been shown to be efficient, robust and capable of identifying the typically singular structures that arise from the original problem. In Figure 3.1, some visual examples of the numerical μ^* obtained for different values of β are shown. The same authors showed that the DMK-Solver is able to emulate the results for the discrete formulation of the PP model proposed by Tero et al. [14].

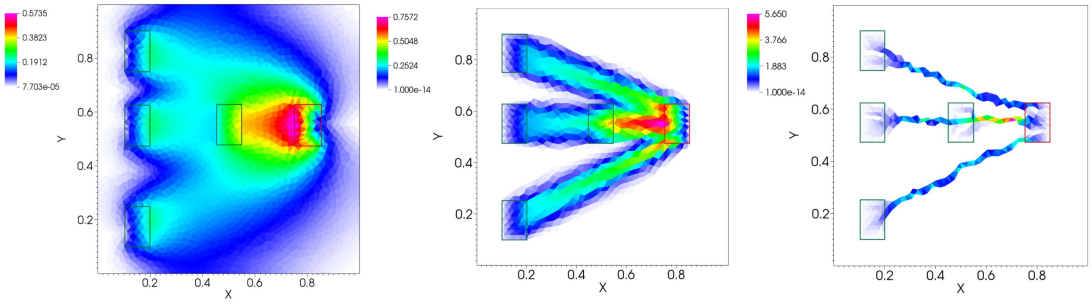


Fig. 3.1: Different values of β in Equation (2.4b) lead to different settings of a routing optimization problem. Colors denote different intensities of conductivity μ as described by the color bar on the left. **Left:** $\beta < 1$ enforces avoiding mass congestion; **Center:** $\beta = 1$ is shortest path-like, the mass goes straight from source to sink; **Right:** $\beta > 1$ encourages traffic consolidation. Red rectangle denotes the sink, green ones the four sources.

Under appropriate regularity assumptions, it can be shown [19, 27] that the equilibrium solution of the above problem $(\mu^*(x), u^*(x))$ is a minimizer of the following functional:

$$\mathcal{L}(\mu, u) = \frac{1}{2}\int_\Omega \mu|\nabla u|^2 dx + \int_\Omega \frac{\mu^{P(\beta)}}{P(\beta)}dx, \tag{3.1}$$

where $P(\beta) = (2-\beta)/\beta$. In words, this functional is the sum of the total energy dissipated during transport (the first term is the Dirichlet energy corresponding to the solution of the first

PDE) plus a nonlinear (sub-additive) function of the total capacity of the system at equilibrium. In terms of costs, this functional can be interpreted as the cost of transport, assumed to be proportional to the total dissipated energy, and the cost of building the transport infrastructure, assumed to be a nonlinear function (with power $2-\beta$) of the total transport capacity of the system.

We exploit the robustness of this numerical solver to extract the solutions of DMK equations corresponding to various routing optimization problems. We here focus on the case $\beta \geq 1$, where the approximate support of μ^* displays a network-like structure. This is the first step of our extraction pipeline, which we denote as DMK-Solver. The numerical solution of these equations does not allow for a straightforward network representation. Indeed, depending on various numerical details related to the spatial discretization and other parameters, one usually obtains a visually well-defined network structure (see Section 3.2) whose rendering as a graph object is however uncertain and non-unique. This in turns can hinder a proper investigation of the topological properties associated to optimal routing optimization problems, motivating the main contribution of our work: the proposal of a graph extraction pipeline to automatically and robustly extract network topologies from the solutions' output of DMK-Solver. We reinforce that our contribution is not limited to this application, but is also able to extract network-like shapes from any kind of image where a color or greyscale thresholds can be used to identify the sought structure.

Our extraction pipeline then proceeds with two main steps: *pre-extraction* and *graph filtering*. The first one tackles the problem of translating a solution from the continuous scenario into a graph structure, while the second one addresses the problem of removing redundant graph structure resulting from the previous step. A pseudo-code of the overall pipeline is provided in Algorithm Line 1[72].

Algorithm 1: Generating optimal networks from solutions of dynamical systems

Input: parameters to set up network extraction problem:
i) Set the space Ω: $\{T_i\}_i$ grid triangulation and mesh-related parameters;
ii) Set up routing optimization problem: $\beta \geq 1$, μ_0, f^+, f^-, and threshold δ;
iii) Set up the *discrete* routing optimization (*graph filtering*): β_d, δ_d, and τ_{BC};

Output: $G(V, E, W)$ final network

1 *Run* DMK-Solver: outputs $(\mu^*,\ u^*)$
2 *Graph pre-extraction*: outputs $G(V, E, W)$ with possible redundancies
3 *Graph filtering*: removes redundancies from $G(V, E, W)$

Our final goal is to translate the solution pair (μ^*, u^*) into a proper network structure using several techniques from graph theory. With these networks at hand, a practitioner is then able to investigate topologies associated with this novel representation of routing optimization solutions.

3.3 Graph preliminary extraction

In this section, we expand on the *graph pre-extraction* step: extracting a network representation from the numerical solution output of the DMK-Solver. This involves a combination of numerical methods for discretizing the space and translating the values of μ^*, and u^* into

edge weights of an auxiliary network, which we denote as $G = (V, E, W)$, where V is the set of nodes, E the set of edges and W the set of weights.

The DMKsolver outputs the solution on a triangulation of the domain Ω (here also named *grid*) and denoted as $\Delta_\Omega = \{T_i\}_i$, with $\cup T_i = \Omega$. The numerical solution, piecewise constant on each triangle T_i, is considered assigned to the triangle barycenter (center of gravity) at position $\mathbf{b}_i = (x_i, y_i) \in \Omega$ [73]. This means that the result is a set of pairs $\{(\mu^*(\mathbf{b}_i), u^*(\mathbf{b}_i))_i\}$. We can track any function of these two quantities. For simplicity, we use μ^* (see Figure 3.1 for various examples), but one could use u^* or a function of these two. This choice does not affect the procedure, although the resulting network might be different.

We neglect information on the triangles where the solution is smaller than a user-specified threshold $\delta \in \mathbb{R}_{\geq 0}$, in order to work only with the most relevant information. Formally, we only keep the information on T_i such that $\mu^*(\mathbf{b}_i) \geq \delta$. We observed empirically that in many cases, several triangles contain a value of μ^* that is orders of magnitude smaller than others, see for instance the scale of Figure 3.1. Since we want to build a network that connects these barycenters, we remark that this procedure depends on the choice of the threshold δ: if $\delta_1 < \delta_2$, then $G(\delta_2) \subset G(\delta_1)$. On one hand, the smaller δ, the more likely G is to be connected, but at the cost of containing many possibly loop-forming edges and nodes (the extreme case $\delta = 0$ uses the whole grid to build the final network); on the other hand, the higher δ, the smaller the final network is (both in terms of the number of nodes and edges). Thus one needs to tune the parameter δ such that resulting paths from sources to sinks are connected while avoiding the inclusion of redundant information.

The set of relevant triangles does not correspond to a straightforward meaningful network structure, i.e. a set of nodes and edges connecting neighboring nodes. In fact, we want to remove as much as possible the biases introduced by the underlying triangulation and thus we start by connecting the triangle barycenters. For this, we need rules for defining nodes, edges and weights on the edges. Here, we propose three methods for defining the graph nodes and edges and two functions to assign the weights. The overall *graph pre-extraction* routine is given by choosing one of the former and one of the latter, and it can be applied also to more general inputs beyond solutions of the DMK-Solver.

3.3.1 Rules for selecting nodes and edges

Selecting V and E requires defining the neighborhood $\sigma(T_i)$ of a triangle in the original triangulation Δ_Ω (for i such that $\mu^*(\mathbf{b}_i) \geq \delta$). We consider three different procedures:

(I) Edge-or-node sharing: $\sigma(T_i)$ is the set of triangles that either share a grid edge or a grid node with T_i.

(II) Edge-only sharing: $\sigma(T_i)$ is the set of triangles that share a grid edge with T_i. Note that $|\sigma(T_i)| \leq 3, \quad \forall i$.

(III) Original triangulation: let v, w, s be the grid nodes of T_i ; then add v, w, s to V and $(v, w), (w, s), (s, v)$ to E.

It is worth mentioning that since the grid Δ_Ω is non uniform and μ^* is not constant, we cannot control *a priori* the degree d_i of a node i in the graph G generated for a particular threshold δ. We give examples of networks resulting from these three definitions in Figure 3.2 and a pseudo-code for the first two in Line 2.

Algorithm 2: Graph extraction

Input: (μ, u) solution of the DMK-Solver, δ threshold, $\{T_i\}_i$ grid triangulation

Initialize: $V, E = \emptyset$
for $i \;\; s.t. \;\; \mu(\mathbf{b}_i) \geq \delta$ **do**
  $V \leftarrow V \cup \{i\}$
  for $T_j \in \sigma(T_i) \;\; s.t. \;\; \mu(\mathbf{b}_j) \geq \delta$ **do**
    $V \leftarrow V \cup \{j\}$
    $E \leftarrow E \cup e_{ij} := (i, j)$
    $w_{ij} = f\left(\mu(\mathbf{b}_i), \mu(\mathbf{b}_j)\right)$
  end
end

Output: a $G(V, E, W)$ network

3.3.2 Rules for selecting weights

The weights w_{ij} to be assigned to edges $e_{ij} := (i, j) \in E$ should be a function of $\mu(\mathbf{b}_i)$ and $\mu(\mathbf{b}_j)$, the density contained in the original triangles. We consider two possibilities for this function:

(i) Average (AVG): $w_{ij} = \frac{\mu(\mathbf{b}_i)+\mu(\mathbf{b}_j)}{2}$.

(ii) Effective reweighing (ER): $w_{ij} = \frac{\mu(\mathbf{b}_i)}{d_i} + \frac{\mu(\mathbf{b}_j)}{d_j}$.

While using the average as in Item (i) captures the intuition, it may overestimate the contribution of a triangle when this has more than one neighbor in G with the risk of calculating a total density larger than the original output of the DMK-Solver. To avoid this issue, we consider an *effective* reweighing as in Item (ii), where each triangle contribution by the degree $d_i = |\sigma_i|$ of a node $i \in V$ is reweighted, with σ_i the set of neighbors of i. This guarantees the recovery of the density obtained from DMK-Solver, since

$$\frac{1}{2}\sum_{i,j} w_{ij} = \frac{1}{2}\sum_i \left[\mu(\mathbf{b}_i) + \sum_{j\in\sigma_i} \frac{\mu(\mathbf{b}_j)}{d_j}\right] = \sum_i \mu(\mathbf{b}_i),$$

where in the sum we neglected isolated nodes, i.e. i s.t. $d_i = 0$. Note that in the case of choosing the original triangulation for node and edge selection (case Item (III) above), the ER rule does not apply; in that case, we use AVG.

3.4 Graph filtering

The output of the graph extraction step is a network closer to our expectation of obtaining an optimal network topology resulting from a routing optimization problem. However, this network may contain redundant structures like dangling nodes or small irrelevant loops (see Figure 3.2). These are not related to any intrinsic property of optimality, but rather are a feature of the discretization procedure resulting from the graph pre-extraction step. It is thus important to filter the network by removing these redundant parts. However, how to perform this removal in an automated and principled way is not an obvious task. One has to be careful in removing enough structure, while not compromising the core optimality properties of the

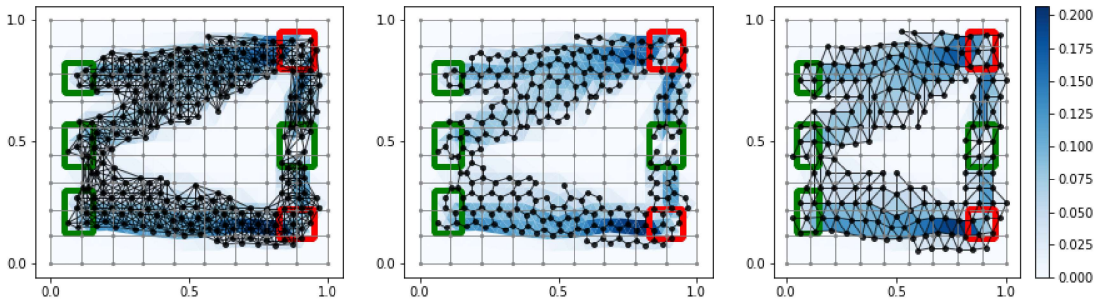


Fig. 3.2: *Graph pre-extraction* rules. Left): edge-or-node sharing Item (I); center): edge-only sharing Item (II); right): original triangulation Item (III). We monitor the conductivity μ and use parameters $\mu_0 = 1$, $\beta = 1.02$, $\delta = 0.0001$. Weights w_{ij} are chosen with AVG Item (i), f is chosen such that sources and sinks are inside green and red rectangles respectively.

network. This removal is then a problem in-and-of-itself, we name it *graph filtering* step. We now proceed by explaining how we tackle it in a principled way and discuss its quantitative interpretation in terms of minimizing a cost function interpolating between an operating and an infrastructural cost.

3.4.1 The Discrete DMK-Solver

Going beyond heuristics and inspired by the problem presented in Section 3.2, we consider as a solution for the graph filtering step, the implementation of a second routing optimization algorithm to the network G output of the pre-extraction step, i.e. in discrete space. Several choices for this could be drawn, for instance, from routing optimization literature [74], but we need to make sure that this second optimization step does not modify any of the intrinsic properties related to optimality resulting from the DMK-Solver.

We thus propose to use a *discrete* version of the DMK-Solver (*discrete-DMK-Solver*). This was proven to be related to the *Basis Pursuit* (BP) optimization problem [29]. In fact, BP is related[75] to the PP dynamical problem in discrete space and the *discrete-DMK-Solver* gives a solution to the PP in discrete space[29]. The discretization results in a reduction of the computational costs for solutions of BP problems, compared to standard combinatorial optimization approaches[29]. Being an adaptation to discrete settings of our original optimization problem, it is a natural candidate for a graph filtering step, preserving the solution's properties. This dynamics was already introduced in Section 2.3, including the interpretation of the discrete Lyapunov in Equation (2.14).

Similarly to the graph pre-extraction step, the filtering is also valid beyond networks related to solutions of the DMK-Solver. It applies to more general inputs if defined on a discrete space, for instance, images. Finally, notice that the filter generates a graph with a new set of nodes and edges, both subsets of the corresponding ones in G, result of the pre-extraction. The weights of the final graph can then be assigned with same rules as in Section 3.3.1; in addition, one can consider as weights the values of μ_e resulting from the BP problem (we named this weighing method "BPW"). Alternatively, one can ignore the weights of BP and keep (for the edges remaining after the filter) the weights as in the previous pre-extraction step (labeled as "IBP"). Figure 3.3 shows an example of three filtering settings on the same input.

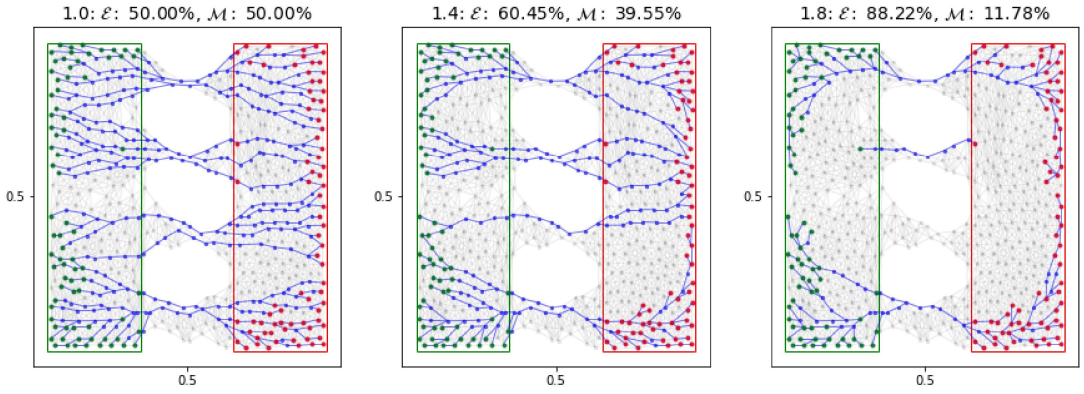


Fig. 3.3: *Graph filtering* rules. Left): $\beta_d = 1.0$; center): $\beta_d = 1.4$; right): $\beta_d = 1.8$. The number on top denote the contributions of operating and infrastructural cost to the energy. Green and red dots represent sources and sinks respectively ($\tau_{BC} = 10^{-1}$); blue edges are those e with $\mu_e^* \geq 10^{-3}$. The filtering input is generated from DMK-Solver with $\beta = 1.05$. The apparent lack of symmetry of the network's branches is due to numerical discretization of the domain, solver and threshold δ. As the relative size of the terminal set decreases compared to the size of remaining part of the domain, this lack of symmetry becomes negligible.

3.4.2 Selecting sources and sinks

The *discrete-DMK-Solver* requires in input a set of source and sink nodes (S^+ and S^-) that identify the support of the forcing vector $\mathbf{f}$ introduced in Section 3.4.1. However, the graph pre-extraction output G might contain redundant nodes (or edges) as mentioned before. In principle, among the nodes $i \in V$, all of those contained in the support of $f(x = \mathbf{b}_i)$, i.e. contained in the supports of sources and sinks of the original routing optimization problem in Section 2.3, are *eligible* to be treated as sources or sink in the resulting network. However, several paths connecting source and sink nodes may be redundant and clearly not compatible with an optimal routing network (see Figure 7.2 for such example). Therefore, it is important to select "representatives" for sources and sinks, such that the final network is heuristically closer to optimality. Here we propose a criterion to select source and sink nodes from the eligible ones in each of the connected components $\{C_m\}_m$ of G, using a combination of two network properties. Starting from the complete graph formed by all the nodes characterized by a significant (above the threshold) density, source and sink nodes and rates are defined as follows.

A node $i \in S^+$, i.e. is a source $f_i > 0$, if either i) is in the convex hull of the set of eligible sources or ii) its betweenness centrality is smaller than a given threshold τ_{BC}. Similarly for sink nodes in S^-. This is because, on one side, nodes in the convex hull capture the outer shape structure of the source and sink sets defined in the continuous problem; on the other side, nodes with *small* values of the betweenness centrality capture the end-points of G inside the source and sink sets, analogously to leaves (i.e., degree-one nodes) [76]. We present more details in Section 7.2.

Once we have identified the sets of source and sink vertices, we need to assign a proper value f_i such that Kirchhoff law is satisfied in each of the different connected components C_m. It is reasonable to assume that each connected component is "closed", i.e. $\sum_{i \in C_m} f_i = 0, \forall C_m$. Denoting with $|S|$ the number of elements in a set S and $V(C_m)$ the set of nodes in C_m, we then distribute the mass-fluxes uniformly by setting $f_i = \frac{1}{|S^+ \cap V(C_m)|}$ for $i \in S^+$, and $f_i = -\frac{1}{|S^- \cap V(C_m)|}$ for $i \in S^-$ sinks ($f_i = 0$ otherwise) so that the total original source and sink

flux is assigned to the overall source/sink nodes of all C_m. Note that this procedure maintains the overall system and each connected component "closed", as stated above.

3.5 Model validation

Our extraction pipeline proceeds by compressing routing information in the raw output of the DMK-Solver (although what follows is not restricted to this case) on a lean network structure. This might lead us to lose relevant information in the process. Hence, we need to devise *a posteriori* estimates that provide quantitative guidance on the "leanness" and information loss of the final network. Here we propose metrics to evaluate the compression performance of the various graph pre-extraction and filtering protocols. The raw information is made of a set of weights $w(T_i)$ representing the values (μ^*, u^*) on each of the triangles $T_i \in \Omega$. We consider as the truth benchmark the distribution of w, or any other quantity of interest, supported on the subgrid $\Delta_\Omega^\delta \subset \Delta_\Omega$ formed by all triangles where w is larger than the threshold value δ, i.e., $\Delta_\Omega^\delta := \{T_i \in \Delta_\Omega : w(T_i) \geq \delta\}$. We expect that a good compression scheme should preserve both the total *amount* of the weights from the original solution in Δ_Ω^δ and the information of *where* these weights are located inside the domain Ω. Also, we want this compression to be parsimonious, i.e. to store the least amount of information as possible. We test against these two requirements by proposing two metrics that measure: i) an information difference between the raw output of the DMK-Solver and the network extracted using our procedure, capturing the information of where the weights are located in space; ii) the amount of information needed to store the network.

Our first proposed metric relies in partitioning Ω in several subsets and then calculating the difference in the extracted network weights and the uncompressed output, *locally* within each subset. More precisely, we partition Ω into P non intersecting subsets $C_\alpha \subset \Omega$, with $\alpha = 1, \dots, P$ and $\cup_1^P C_\alpha = \Omega$. For example, we define $C_\alpha = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$, for x_i, x_{i+1}, y_j and y_{j+1}, consecutive elements of N-regular partitions of $[0, 1]$, and $P = (N-1)^2$. Denote with $w_\delta(T_i)$ the weight on the triangle $T_i \in \Delta_\Omega^\delta$, resulting from the DMK-Solver (usually a function of μ^* and u^*). If we denote the *local weight* of Δ_Ω^δ inside C_α as $w_\alpha = \sum_{i:\mathbf{b}_i \in C_\alpha} w_\delta(T_i)$, then we propose the following evaluation metric:

$$\hat{w}_q(G) \quad := \quad \frac{1}{P} \left[\sum_{\alpha=1}^{P} \left(\sum_{e \in E} | \, \mathbb{I}_\alpha(e) \; w_e - w_\alpha \, | \right)^q \right]^{\frac{1}{q}}, \tag{3.2}$$

where $\mathbb{I}_\alpha(e)$ is an indicator of whether an edge $e = (i, j) \in E$ is inside an element C_α of the partition, i.e. $\mathbb{I}_\alpha(e) = 1, 0, 1/2$ if both $\mathbf{b}_i, \mathbf{b}_j$ are in C_α, none of them are, or only one of them is, respectively.

In words, $\hat{w}_q(G)$ is a distance between the weights of the network extracted by our procedure and the original weights output of the DMK-Solver, over each of the local subsets C_α. This metric penalizes networks that either place large-weight edges where they were not present in the original triangulation, or low-weight ones where they were instead present originally. In this work we consider the Euclidean distance, i.e. $q = 2$, but other choices are also possible.

Note that $\hat{w}_q(G)$ does not say anything about how much information was required to store the processed network. If we want to encourage parsimonious networks, i.e. networks with few redundant structures, then we should include in the evaluation the monitoring of $L(G) = \sum_{e \in E} \ell_e$, the total path length of the compressed network, where the edge length ℓ_e can be specified based on the application. Standard choices are uniform $\ell_e = 1, \forall e$ or the Euclidean distance between $\mathbf{b}_i$ and $\mathbf{b}_j$. Intuitively, networks with small values of both $\hat{w}_q(G)$

and $L(G)$ are both accurate and parsimonious representations of the original DMK solutions defined on the triangulation.

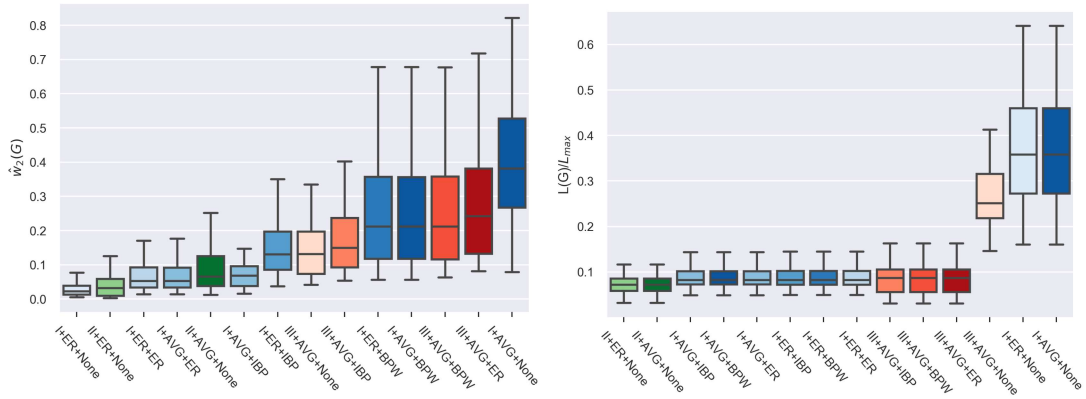


Fig. 3.4: Graph extraction performance evaluation. We plot results for different combinations of the graph extraction rule in terms of: (left) the metric $\hat{w}_q(G)$ of Eq. (3.2); (right) total network length $L(G)$ normalized by L_{max}, the max length over the 170 networks. Each bar denotes a possible combination as follows: roman numbers denote one of the three rules I-III (3.3.1); first label after the number denotes one of the rules to assign weights i-ii 3.3.2), which is applied to the output of the first step; the second (and last) label denotes the same rule but applied after the filtering step, "None" means that nothing is done, i.e. no filter applied, "IBP" means filter applied but with no reweighing, i.e. when an edge is removed by the filter we simply lose information without relocating its weight. Bars are color-coded so that rules I-III have three different primary colors and their corresponding routines have different shades of that main color. Here, we keep track of the conductivity μ and show medians and quartiles of a distribution over 170 networks generated with $\beta \in \{1.1, 1.2, 1.3\}$, $\beta_d = 1.1$ and $\delta = 0.01$.

We evaluate numerous graph extraction pipelines in terms of these two metrics on various routing optimization problem settings and parameters. In Figure 3.4 we show the main results for a distribution of 170 networks obtained with $\beta \in \{1.1, 1.2, 1.3\}$ and $\beta_d = 1.1$. Similar results were obtained for other parameter settings. Networks are generated as follows: first, we choose a set of 5 different initial transport densities μ_0, grouped in parabola-like, delta-like and uniform distributions, and a set of 12 different configurations for sources/sinks (mainly rectangles placed in different positions along the domain, see Section 7.3 for more details). Then, for each of these setup, we run our procedure: i) first the DMK-Solver calculates the solution of the continuous problems; ii) then we apply the *graph pre-extraction* procedure according the rules of Section 3.3.1 and weights as in Section 3.3.2; iii) finally, we run the *graph filtering* step and consider various weight functions, as described in Figure 3.4.

We observe that not applying the final filtering step and considering rule I with ER to build the graph (I-ER-None), the values of $\hat{w}_2(G)$ are smaller than other cases. This is expected as by filtering we remove information and thus achieve better performance with this metric when compared to no filtering. However, we pay a price in terms of total relative length as $L(G)/L_{max}$ is larger for this case. When working with rule II, we notice the appearance of many non-optimal small disconnected components and this effect deteriorates if filtering is activated. Corresponding statistics show low values for both $\hat{w}_2(G)$ and $L(G)/L_{max}$.

We argue that this is because rule II produces, by construction, fewer redundant objects than rule I in the initial phase. This might have a similar effect as a filter but is done *a priori* during the pre-extraction, because rule II produces in this phase a limited number of effective neighbors. However, this comes at a price of higher variability with the sampled networks, as the variance of $\hat{w}_2(G)$ is higher than for the other combinations. Among the possibilities with filtering applied, we observe that rule I performs better than rule III, while

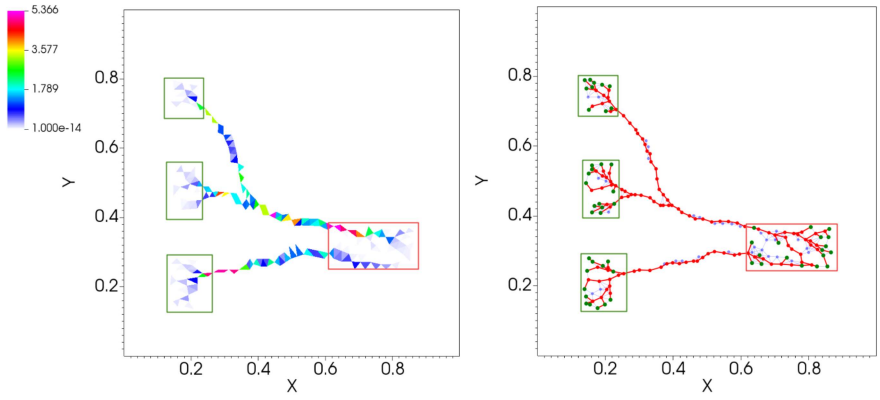


Fig. 3.5: Network extraction example. We show a network generated from a routing optimization problem with parameters $\beta_c = 1.4$, $\beta_d = 1.3$ and $\delta = 0.001$; (left) raw output of the DMK-Solver; (right) final network extracted using the routine I-ER-ER.

all the weighting rules give a similar performance in terms of both metrics. Any combination involving rule I plus filtering has a similar performance as rule II in terms of both metrics but with smaller variability. Finally, these combinations perform differently in terms of the number of disconnected components (not shown here), with rule II producing more spurious splittings, as already mentioned. Depending on the application at hand, a practitioner should select one of these combinations based on their properties as discussed in this section. We give an example of a network generated with I-ER-ER in Figure 3.5.

3.6 Application: network analysis of a vein network

We demonstrate our protocol on a biological network of fungi foraging for resources in space. The network structure corresponds to the fungi response to food cues while foraging [77]. Edges are veins or venules and connect adjacent nodes. This and those of other types of fungi are well known networks typically studied using image segmentation methods [55, 56, 57, 58]. It is thus interesting to compare results found by these techniques and by our approach, under the conjecture that the underlying dynamic driving the network structure could be the same as the optimality principles guiding our extraction pipeline.

In particular, we are interested in analyzing the distribution $P(\ell)$ of the veins lengths, i.e. the network edges. The benchmark $P(\ell)$ distribution obtained by Baumgarten and Hauser [55] using image processing techniques is an exponential of the type $P(\ell) = P_0\, e^{-\gamma \ell}$. Accordingly, as shown in Fig. 3.6, we find that an exponential fit (with values $P_0 = 234.00, \gamma = 36.32$) well captures the left part of the distribution, i.e. short edges. Differences between fit and observed data can be seen in the right-most tail of the graph, corresponding to longer path lengths, where the data decay faster than the fit. However, we find that the exponential fit is nevertheless better than other distributions, such as the gamma and log-normal proposed in Dirnberger and Mehlhorn [78] for the *P. polycephalum*. Drawing definite quantitative conclusions is beyond the scope of our work, as this example aims at a qualitative illustration of possible applications that can be addressed with our model. In general, however, it seems not possible to choose a single distribution that well fits both center and tails of the distribution for various datasets of this type [78].

To conclude, we demonstrate the flexibility of our graph extraction method on a more general input than the one extracted from DMK-Solver. Specifically, we consider as example an image

of *P. polycephalum* taken from data publicly available in the Slime Mold Graph Repository (SMGR) repository [79]. We first downsample an image of the SMGR's *KIST Europe data set*, using *OpenCV* (left) and a color scale defined on the pixels as an artificial μ^* function. We build a graph using the *graph pre-extraction* and *graph filtering* steps as shown in Fig. 3.7.

Our protocol in its standard settings with filtering can only generate tree-like structures. Therefore, if we want to obtain a network with loops as we did in Fig. 3.7, we should consider a modification of our routine, which can be done in a fully automatized way, as detailed in Section 7.4. In short, after the *graph pre-extraction* step, where loops are still present, we extract a tree-like structure close to the original loopy graph and give this in input to the filtering. We can then add *a posteriori* edges that connect terminals that were close by in the graph obtained from the pre-extraction step but removed by the filter, thus recovering loops. In case obtaining loops is not required, our routine can be used with no modifications. Adapting our filtering model to allow for loopy structures in a principled way, analogously to what done in Sec. 3, will be subject of future work.

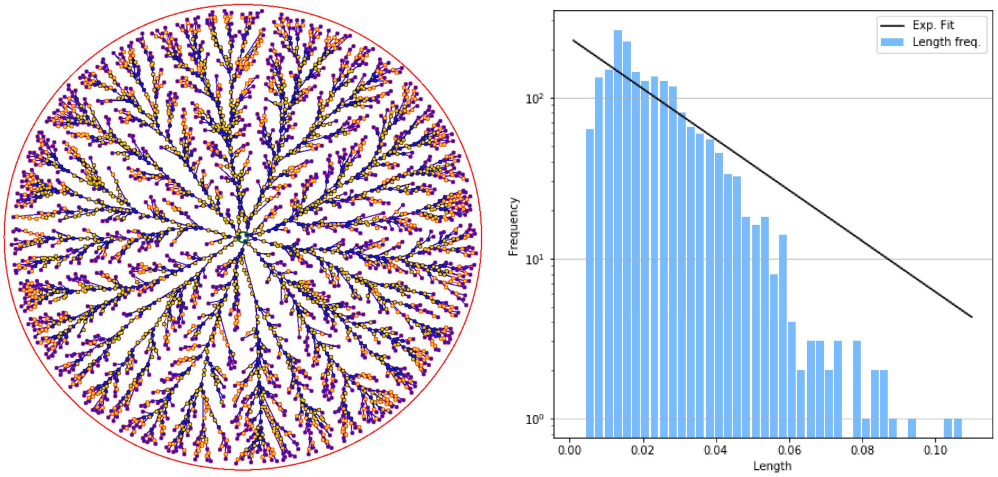


Fig. 3.6: Application to fungi network. We generate a synthetic network similar to the image of Fig. 1a reported in Boddy et al. [58] and Fig. 4a in Obara et al. [56] for the of *Phanerochaete velutina* fungus [77] and Fig. 1 in their supplementary for the *Coprinus picaceus*. Fitted parameters are: $P_0 = 234.00, \gamma = 36.32$. Here $f^+(x,y) = 1$, if $(x-0.5)^2 + (y-0.5)^2 \leq 0.01$; $f^+(x,y) = 0$, otherwise; $f^-(x,y) = -1$, if $0.01 < (x-0.5)^2 + (y-0.5)^2 \leq 0.45$; $f^-(x,y) = 0$, otherwise. The network on the left corresponds to the filtered graph. Yellow nodes are degree-2 nodes that we omitted when computing the length distribution. Green and red outlines are used to denote nodes in S^+ and S^-, respectively.

3.7 Discussion

We propose a graph extraction method for processing raw solutions of routing optimization problems in continuous space into interpretable network topologies. The goal is to provide a valuable tool to help practitioners bridging the gap between abstract mathematical principles behind optimal transport theory and more interpretable and concrete principles of network theory. While the underlying routing optimization scheme behind the first step of our routine uses recent advances of optimal transport theory, our tool enables automatic graph extraction without requiring expert knowledge.

We purposely provide a flexible routine for graph extraction so that it can be easily adapted to serve the specific needs of practitioners from a wider interdisciplinary audience. We thus

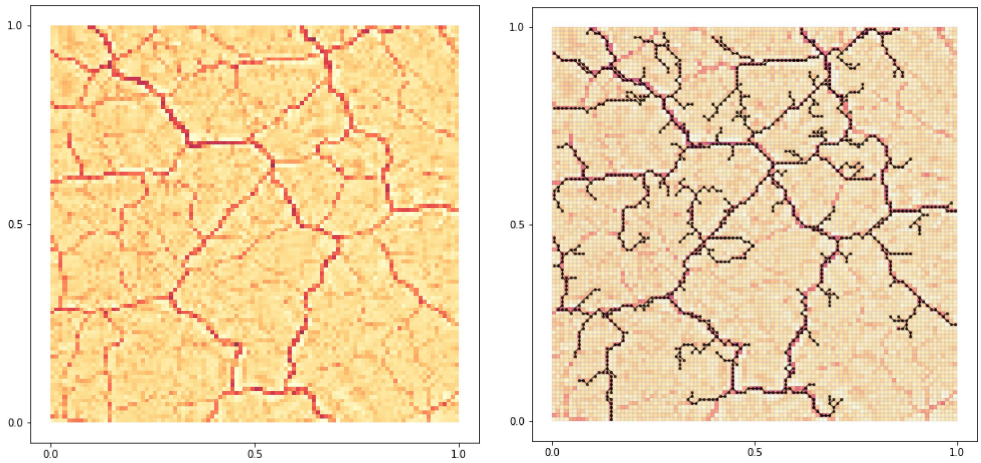


Fig. 3.7: Application to images. We take an image of the *P. Polycephalum* from the SMGR repository [79].The picture used is a 1200x1200-pixel section of an original image of size 5184x3456 pixels (see Section 7.4 for details) and extract a network with step 2 and 3 of our protocol. As a pre-processing step, we downsample the image using *OpenCV* (left) and use the color scale defined on the pixels as an artificial μ^* function. Using this information and the grid structure associated to the image's pixels, we first build a graph G with the *graph pre-extraction* step described in Sec. 3.3; then, we obtain a graph G_f (right) using the *graph filtering* step of Sec. 3.4, for an appropriate selection of sources and sinks, and adding a correction to retrieve loops.

encourage users to choose the parameters and details of the subroutines to suitably customize the protocol based on the application of interest. To help guiding this choice, we provide several examples here and in the Chapter 7.

We anticipate that this work will find applications beyond that of automating graph extraction from routing optimization problems. We remark that two of the three steps of our protocol apply to inputs that might not necessarily come from solutions of routing optimization. Indeed, the pipeline can be applied to any image setting where an underlying network needs to be extracted.

The advantage of our setting with respect to more conventional machine learning methods is that the final structure extracted with our approach minimizes a clearly defined energy functional, that can be interpreted as the combination of the total dissipated energy during transport and the cost of building the transport infrastructure.

We foresee that this minimizing interpretation together with the simplification of the pipeline from abstract modeling to final concrete network outputs will foster cross-breeding between fields as our tool will inform network science with optimal transport principles and vice-versa. In addition, we expect to advance the field of network science by promoting the creation of new network databases related to routing optimization problems.

4 Community Detection in networks by Dynamical Optimal Transport Formulation

Detecting communities in networks is important in various domains of applications. While a variety of methods exist to perform this task, recent efforts propose Optimal Transport (OT) principles combined with the geometric notion of Ollivier-Ricci curvature to classify nodes into groups by rigorously comparing the information encoded into nodes' neighborhoods. In this chapter, we present an OT-based approach that exploits recent advances in OT theory to allow tuning between different transportation regimes. This allows for better control of the information shared between nodes' neighborhoods. As a result, our model can flexibly capture different types of network structures and thus increase performance accuracy in recovering communities, compared to standard OT-based formulations. We test the performance of our algorithm on both synthetic and real networks, achieving a comparable or better performance than other OT-based methods in the former case while finding communities that better represent node metadata in real data. This pushes further our understanding of geometric approaches in their ability to capture patterns in complex networks.

4.1 Introduction

Complex networks are ubiquitous, hence modeling interactions between pairs of individuals is a relevant problem in many disciplines [21, 80]. Among the variety of analyses that can be performed on them, community detection [81, 82, 83, 84] is a popular application that involves finding groups (or communities) of nodes that share similar properties. The detected communities may reveal important structural properties of the underlying system. Community detection has been used in diverse areas including, discovering potential friends on social networks [85], evaluating social networks [86], personalized recommendation of item to user [87], detecting potential terrorist activities on social platforms [88], fraud detection in finance [89], study epidemic spreading process [90] and so on.

Several algorithms have been proposed to tackle this problem which utilize different approaches, such as statistical inference [91, 92], graph modularity [93], statistical physics [94], information theory [95] and multifractal topological analysis [96]. Here, instead, we adopt a recent approach connecting community detection with geometry, where communities are detected using geometric methods like the Ollivier-Ricci curvature (ORC) and we exploit a dynamical approach of optimal transport theory to calculate this efficiently and flexibly across various transportation regimes.

In Riemannian geometry, the sign of the curvature quantifies how geodesic paths converge or diverge. In networks, the ORC plays a similar role: edges with negative curvature are traffic bottlenecks, whereas positively curved ones allow mass to flow more easily along the network. Defining communities as structures that allow robust transport of information, we could cluster edges based on their curvature: those with positive curvature can be clustered together, while those with negative curvature may be seen as "bridges" connecting different communities. The

idea of using Ricci curvature to find communities on networks was first proposed by Jost and Liu [97] and then further explored in subsequent works [98, 99, 100, 101]. Our work follows a similar approach as in [100, 101] to calculate the OR curvature, but generalizes it for the cases of branched [41, 102] and congested [103] optimal transport problems, building from recent results [104, 105]. Specifically, our algorithm allows to efficiently tune the sensitivity to detecting communities in a network, through a parameter that controls the flow of information shared between nodes. We perform a comprehensive comparison between the proposed algorithm and existing ones on synthetic and real data. Our algorithm, named ORC-Nextrout, detects communities in synthetic networks with similar or higher accuracy compared to other OT-based methods in the regime where inference is not trivial, i.e. the inference problem is neither too easy nor too difficult to solve, and thus communities are only partially retrieved. This is also observed in a variety of real networks, where the ability to tune between different transportation regimes allows finding at least one result that outperforms other methods, including approaches based on statistical inference, and modularity.

4.1.1 Related work

The idea of exploring the geometrical properties of a graph, and in particular curvature, has been explored in different branches of network science, ranging from biological[106] to communication [107] networks. Intuitively, the Ricci curvature can be seen as the amount of volume through which a geodesic ball in a curved Riemannian manifold deviates to the standard ball in Euclidean spaces[108]. When defined in graphs, it indicates whether edges (those with positive values for the curvature) connect nodes inside a cluster, or if they rather bond different clusters together (those with negative values for the curvature).

Previous works [109, 110, 111, 112] extended the idea of the OR curvature. In [109], the authors introduced the concept of "resistance curvature" for both nodes and edges. Taking inspiration from electrical circuits, this approach assigns a resistance being applied by the whole network from a current that flows between any two edges and correlates this to known concepts of OR discrete curvature. The resistance curvature provides a natural way to define the Ricci flow. In [110] the authors proposed a *dynamical* version of the OR curvature, where a continuous-time diffusion process is defined for every node, at different time scales. In this context, the dynamical perspective is used to frame probability masses at nodes in terms of diffusion processes, e.g. those deployed in random walks.

In our work instead, the dynamics enters to solve efficiently the underlying optimization problem required to compute the OR curvature. Regardless of the choice of the distribution that characterizes mass on nodes, this quantity is then used to define the curvature of the edges of the graph. Previous works have typically defined the OR curvature in terms of the 1-Wasserstein distance. In contrast, we take a more general approach and explore the usage of the β-Wasserstein, where $\beta \in (0, 2]$, to account for a variety of OT problems, ranging from branched to congested transportation.

Other discrete graph curvature approaches include the Ollivier-Ricci (OR) curvature based on the Optimal Transport theory introduced by Ollivier [113, 114], and Forman-Ricci curvature introduced by Forman[115]. While the graph Laplacian-based Forman curvature is computationally fast and less geometrical, we focus on the OT-based approach due to its more geometric nature. Some applications of the Ollivier-Ricci curvature include network alignment [116] and community detection [101, 100, 117].

On the other hand, community detection in networks is a fundamental area of network science, with a wide range of approaches proposed for this task [81, 82, 22]. These include methods based on statistical mechanical models [95, 94, 118], probabilistic generative models [91, 20,

119, 120], nonnegative matrix factorization[121], spectral methods [122, 123], multifractal topological analysis [96] and modularity optimization [124, 125, 93]. In contrast, our work is inspired by recent OT-based methods [101, 100] for community detection. These methods consider the OR curvature to sequentially identify and prune negatively curved edges from a network to identify communities. While our approach also considers OR curvature to prune edges, it controls the flow of information exchanged between nodes by means of a traffic-penalization parameter, making the edge pruning completely dynamic. This is detailed in Section 4.2.

4.2 β-Wasserstein Community Detection Algorithm

In this section, we describe how our approach solves the community detection problem. As previously stated, we rely on optimal transport principles to find the communities. To solve the optimal transport problem applied in our analysis we use the discrete *Dynamic Monge-Kantorovich* model (*DMK*), as proposed by Facca et al. [126, 28] to solve transportation problems on networks.

We denote a weighted undirected graph as $G = (V, E, W)$, where V, E, W are the set of nodes, edges, and weights, respectively. We use the information of the neighborhood of a node i, $\mathcal{N}(i) = \{j \in V | (i, j) \in E\}$, to decide whether a node belongs to a given community. We do this by comparing a distribution defined on $\mathcal{N}(i)$ with those defined on other nodes close to i. There are several choices that can be made for this. For instance, one could frame this in the context of diffusion processes on networks and relate the distribution to random walkers traveling along the network with a certain jump probability[110]. Here we follow previous work[116] and assign it as m_i^α, where $m_i^\alpha(k) := \alpha$ if $k = i$ and $m_i^\alpha(k) := (1 - \alpha)/|\mathcal{N}(i)|$ if $k \in \mathcal{N}(i)$. Intuitively, the distribution m assigns a unit of mass to i and its connections: α controls how much weight node i should have, and once this is assigned, its neighbors receive the remaining mass in an even way. We use $\alpha = 0$ in all the experiments reported in this manuscript, i.e. the mass is equally distributed on the neighbors. This corresponds to a one-step transition probability for a random walker in the context of diffusion processes.

The next step is to compare the distribution m_i^α of node i to that of its neighbors. Consider an edge $(i, j) \in E$ and m_j, the distribution defined on node j, neighbor of i. We assume that if i and j belong to the same community, then both nodes may have several neighbors in common and, therefore, m_i and m_j should be similar. Note that this is valid for both assortative and disassortative community structures. In the former case, nodes are more likely to interact within the same community, while in the latter case we have the opposite, nodes are more likely to interact across communities [80, 82]. When there is a consistent community pattern for all groups (e.g. all communities are assortative), this idea of comparing the distributions m_i^α may be appropriate to detect communities. On the contrary, it may be difficult to perform this task in networks with mixed connectivity patterns, where some communities are assortative and others are disassortative. This makes it difficult to detect communities as edges within an assortative community are shortened, likewise edges between a node in a disassortative and a node in an assortative one. This may confuse the algorithm, as both types of edges are shortened. A careful treatment of these cases is an interesting direction for future work.

To estimate the similarity between m_i and m_j we use OT principles. Specifically, we compute the cost of transforming one distribution into the other. This is related to the cost of moving the mass from one neighborhood to the other, and it is assumed to be the weighted shortest-path distance between nodes belonging to $\mathcal{N}(i)$ and $\mathcal{N}(j)$. A schematic representation of the

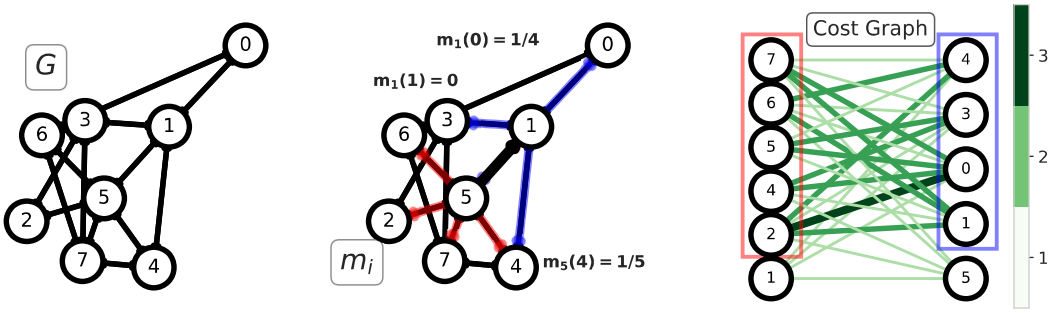


Fig. 4.1: Left): an example graph G where edges have unitary weights. Center): the edge $(1, 5)$ (bold black line) is selected to define the OT problem between m_1, m_5; neighborhoods of nodes 1 and 5 are highlighted with blue and red edges and are used to build the corresponding distributions m_1, m_5. Right): The complete bipartite graph B_{15} where the OT problem is defined. The color intensity of the edges represents the distance between the associated nodes on the graph G, as shown by the color bar. m_1 and m_5 are both defined for $\alpha = 0$, i.e. no mass is left in 1 and 5.

algorithm can be seen in Figure 4.1. The OT problem is solved in an auxiliary graph, the complete bipartite network $B_{ij} = (V_{ij}, E_{ij}, \omega_{ij})$ where $V_{ij} := (V_i, V_j) := (\mathcal{N}(i) \cup \{i\}, \mathcal{N}(j) \cup \{j\})$, E_{ij} is made of all the possible edges between V_i and V_j. The weights of the edges are given by the weighted shortest path distance d between two nodes measured on the input network G.

The similarity between m_i and m_j is the Wasserstein cost $\mathcal{W}(m_i, m_j, \omega_{ij})$ of the solution of the transportation problem. In its standard version, this number is the inner product between the solution Q, a vector of flows defined on edges, and the cost ω_{ij}. In our case, since the DMK model allows to control the flow of information through a hyperparameter $\beta \in (0, 2]$, we define the *β-Wasserstein cost*, $\mathcal{W}_\beta(m_i, m_j, \omega_{ij})$, as the inner product of the solution $Q = Q(\beta)$ of the DMK model and the cost ω_{ij}. For $\beta = 1$ we compute the $1-$Wasserstein distance between m_i and m_j, while for $\beta \neq 1$ the influence of β in the solution of the transportation problem can be seen in Figure 4.2. When $\beta < 1$, more edges of B tend to be used to transport the mass, thus we observe congested transportation [103]. When $\beta > 1$ fewer edges are used, hence we observe branched transportation, and the *β-Wasserstein cost* coincides with a branched transport distance [127, 102]. The idea of tuning β to interpolate between various transportation regimes has been used in several works and engineering applications [104, 30, 128, 129, 130, 131, 132].

Calculating the Wasserstein cost is necessary to determine our main quantity of interest, the discrete Olliver-Ricci curvature, defined as

$$\kappa_\beta(i, j) := 1 - \frac{\mathcal{W}_\beta(m_i, m_j, \omega_{ij})}{d_{ij}} \quad , \tag{4.1}$$

where d_{ij} is the weighted shortest path distance between i and j as measured in G. Intuitively, if i and j are in the same communities, several $k \in V_i$ and $\ell \in V_j$ will be also directly connected. Thus, the Wasserstein distance between m_i and m_j will be shorter than d_{ij}, yielding a positive $\kappa_\beta(i, j)$. Instead, when i and j are in different communities, their respective neighbors will be unlikely connected, hence $d_{ij} < \mathcal{W}_\beta(m_i, m_j, \omega_{ij})$, yielding a negative $\kappa_\beta(i, j)$.

The Ricci flow algorithm on a network is defined by iteratively updating the weights of the graph G [100, 101]. These are updated by combining the curvature and shortest path

distance information [113]. We redefine these updates using our proposal for the Ollivier-Ricci curvature:

$$w_{ij}^{(t+1)} := d_{ij}^{(t)} - \kappa_{\beta}^{(t)}(i,j) \cdot d_{ij}^{(t)}, \tag{4.2}$$

where $w_{ij}^{(t+1)}$ is the weight of edge (i,j) at time t, $w_{ij}^{(0)} = d_{ij}^{(0)}$, and $d_{ij}^{(t)}$ is the shortest path distance between nodes i and j at iteration t. At every time step t, the weights are normalized by their total sum.

The algorithm ORC-Nextrout dynamically changes the weights of the graph G to isolate communities: intra-community edges will be shortened, while inter-community ones will be enlarged. These changes are reached after a different number of iterations of the whole routine depending on the input data. To find the communities, we apply a *network surgery* criterion on the edges based on the stabilization of the modularity of the network, as proposed by Ni et al.[100]. Notice that our algorithm does not need prior information about the number of communities: edges will be enlarged or shortened depending on the optimal transport principles, agnostic to community labeling. The computational complexity of the algorithm is dominated by that of solving the DMK model, which takes $O(|E|^{2.36})$ (estimated numerically) and by computing weighted shortest path distances d_{ij}, which costs $O(|V|^2 \log |V| + |V||E|)$[133]. A pseudo-code of the implementation is shown in Algorithm 3.

Algorithm 3: ORC-Nextrout

Input: $G = (V, E, W)$, traffic rate β, *MaxIterNum*$\in \mathbb{N}$
Output: updated W

Initialize: edge weights $\mathbf{w}^0 = W$; neighborhood distributions $\mathbf{m}$;
for $t \in range(MaxIterNum)$ **do**
  Compute all-pair-shortest-path matrix d^t;
  for $e = (i,j) \in E$ **do**
    Build B_{ij};
    Get $Q(\beta) \in \mathbb{R}^{|E_{ij}|}$, $Q(\beta) = DMK(B_{ij}, m_i, m_j, \beta, d^t)$;
    Compute $\kappa_\beta(e)$ using $d^t(e)$ and $Q(\beta)$;
    Compute $\mathbf{w}_\beta(e)$ using $\kappa_\beta(e)$ and $d^t(e)$;
  end
  Update $\mathbf{w}^t = \mathbf{w}_\beta$;
end

4.3 Results on Community Detection problems

4.3.1 Synthetic Networks

To investigate the accuracy of our model in detecting communities, we consider synthetic networks generated using the *Lancichinetti–Fortunato–Radicchi* (LFR) benchmark [134] and the *Stochastic Block Model* (SBM) [135]. Both models provide community labels used as *ground-truth* information during the classification tasks.

Lancichinetti–Fortunato–Radicchi benchmark: this benchmark generates undirected unweighted networks G with disjoint communities. It samples node degrees and community sizes from power law distributions, see Figure 4.4 for an example. One of its advantages is that it generates networks with heterogeneous distributions of degrees and community sizes. The

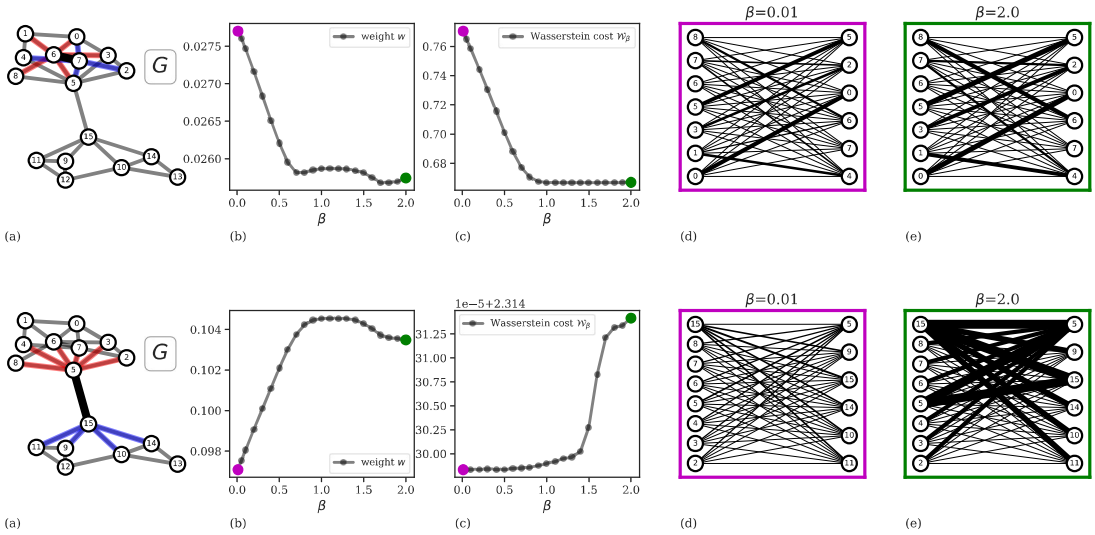


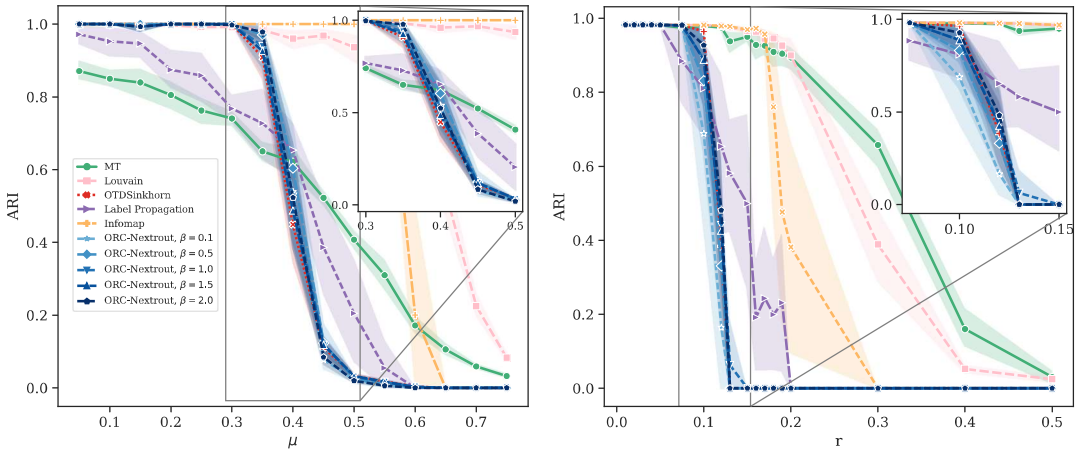
Fig. 4.2: Visualization of how β impacts intra-community and inter-community edge weights. (a) Examples of intra-community (top panel) and inter-community (bottom panel) structures between nodes 6 and 7, and nodes 5 and 15, respectively. (b) The weight of edge $(6, 7)$ decreases when $0 < \beta < 0.6$, while for $0.5 < \beta < 2.0$ it reaches a minimum, and then slightly increases again. Similar but opposite pattern is observed for the edge $(5, 15)$. (c) The β-Wasserstein cost: for intra-community edges, $\beta > 1$ consolidates traffic in the network as the Wasserstein cost stabilizes, making it minimum for the extreme value $\beta = 2$, whereas it is maximized in the case of the inter-community edge. (d-e) Example cost graphs B_{67} (top) and B_{515} (bottom) with fluxes solution of the OT problem (edge thickness is proportional to the amount of flux) in the regimes of small (d) and high (e) values of β.

main parameters in input are the number of nodes N, two exponents τ_1 and τ_2 for the power law distributions of the node degree and community size respectively, the expected degree d of the nodes, the maximum number of communities on the network K_{max} and a fraction μ of inter-community edges incident to each node. To test the performance of our algorithm, we use the set of LFR networks used and provided by the authors of [100]. We set $\tau_1 = 2$, $\tau_2 = 1$, $d = 20$, $K_{max} = 50$ and $\mu \in [0.05, 0.75]$.

Stochastic Block Model: this model probabilistically generates networks with non-overlapping communities. One specifies the number of nodes N and the number of communities K, together with the expected degree d of a node and a ratio $r \in [0, 1]$. Networks are generated by connecting nodes with a probability $r * p_{intra}$ if they belong to different communities; p_{intra} if they are part of the same community, where $p_{intra} = d \times K/N$. Notice that the smaller the ratio r is, the fewer inter-community connections would exist, which leads to networks with a more distinct community structure. We set $N = 500$, $K = 3$, $d = 15$ and $r \in [0.01, 0.5]$ and generate 10 random networks per value of r.

Results

To evaluate the performance of our method in recovering the communities, we use the *Adjusted Rand Index* (ARI) [136]. ARI compares the community partition obtained with the *ground truth* clustering. It takes values ranging from 0 to 1, where ARI $= 0$ is equivalent to random community assignment, and ARI $= 1$ denotes perfect matching with the ground truth



(a) LFR **(b)** SBM

Fig. 4.3: Results on LFR and SBM synthetic data. Performance in detecting ground-truth communities is measured by the ARI score. Markers and shadows are the averages and standard deviations over 10 network realisations with the same value of the parameter used in generation. Markers' shape denote different algorithms. a) LFR graph with $N = 500$ nodes and different values of K ranging from $(17, 22)$. b) SBM with $N = 500$ nodes, $K = 3$ communities and average degree $d = 15$. The parameter r is the ratio of inter-community with intra-community edges. The inset on each plot zooms in the central parts of the plots.

communities, hence the higher this value, the better the recovery of communities. A more detailed presentation of this metric is given in Chapter 8.

We test our algorithm for different types of information spreading in our OT-based model, as controlled by the parameter β, using the software[1] developed in [137]. We used $\beta = 1$, i.e., standard Wasserstein distance; $\beta \in \{0.1, 0.5\}$ for congested transportation, enforcing broad spreading across neighbors; and $\beta \in \{1.5, 2\}$ to favor branching schemes, where fewer edges are used to decide which community a node should belong to. For OT-based algorithms where we update the weights in eq. (4.2) for 15 times ($MaxIterNum = 15$ in Algorithm 3). Since in some cases the ARI score does not consistently increase with the number of iterations, we show results only for the iteration that maximizes the score.

The results in Figure 4.3 show the performance on both LFR and SBM benchmarks with OT-based methods, our method for various β and one based on the Sinkhorn algorithms (OTDSinkhorn) [138, 139]. Our main goal is to assess the impact of tuning between different transportation regimes (as done by β) in terms of community detection via OT principles. Nevertheless, to better contextualize the performance of OT-based algorithms in the wide spectrum of community detection methods, we also include comparisons with algorithms that are not OT-based. Namely, we consider a probabilistic model with latent variables (MT)[91], two modularity-based algorithms, Label Propagation[92] and Louvain [125], and with a flow-based algorithm, Infomap[95].

Our algorithm outperforms OTDSinkhorn for various values of β in an intermediate regime where OT-based inference is not trivial, i.e. detecting communities is neither too easy nor too difficult. This occurs in both the LFR and SBM benchmark, as shown in Figure 4.3. For lower and higher values of the parameters, performance is similar and close to the two extremes of ARI $= 0$ and 1. OT-based methods have a similar sharp decay in performance from the regime where inference is easy to the more difficult one, as also observed in [100]. The other

[1] Source code at https://gitlab.com/enrico_facca/dmk_solver

community detection methods have smoother decay, but with lower performance in the regime where OT-based approaches strive, except for Label Propagation and MT, which are more robust in this sense.

In the intermediate regime where inference is not trivial (i.e. along the sharp decay of OT-based methods), we observe that different values of β give higher performance than OTDSinkhornin most cases. For SBM the highest performance is consistently achieved for high $\beta = 2$, while for LFR the best β varies with μ. A qualitative example where ORC-Nextrout is performing better than OTDSinkhorn, in an instance of LFR of this intermediate regime, is shown in Figure 4.4. Note that in this case, ORC-Nextrout perfectly recovers the 21 communities described by the ground-truth network, whereas OTDSinkhorn merges three of the central communities into one group, therefore recovering only 19 groups. These results suggest that practitioners may choose the β that gives the best performance in detecting communities, e.g. the one that maximizes ARI or other relevant metrics depending on the application at hand. We show examples of this on real data in Figure 4.3.

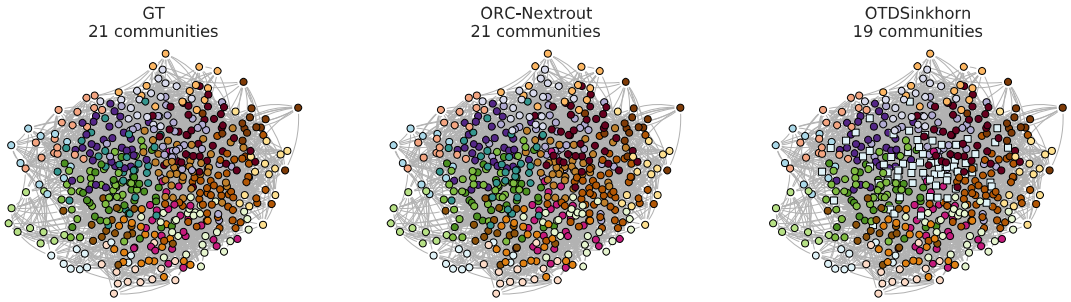


Fig. 4.4: Example of community structure on a synthetic LFR network. The rightmost panel shows the ground-truth community structures to be predicted in an LFR network generated using $\mu = 0.35$. This network is one sample of the synthetic data used in Fig. 4.3. Square-shaped markers denote nodes that are assigned to communities different from those in ground-truth. In the middle and last panels, ORC-Nextrout with $\beta = 2$ perfectly retrieves the 21 communities, while OTDSinkhorn predicts only 19 communities with an ARI score of 0.73, wrongly assigning ground-truth dark green and light brown (square-shaped) nodes to the light blue community.

4.3.2 Real networks

Next, we evaluate our model on various real datasets[140] containing node metadata that can be used to assess community recovery. While failing to recover communities that align well with node metadata should not be automatically interpreted as a model's failure [141] (e.g. the inferred communities and the chosen node metadata may capture different aspect of the data), having a reference community structure to compare against allows one to inspect quantitatively difference between models. These real networks differ on structural features like number of nodes, average degree, number of communities, and other standard network properties as detailed in Table 4.1.

Specifically, we consider i) a network of co-appearances of characters in the novel *Les Misérables* [142] (Les Miserables). Edges are built between characters that encounter each other. ii) A network of 62 bottlenose dolphins in a community living off Doubtful Sound, in New Zealand [143] (Dolphins). The nodes represent dolphins, and the edges indicate frequent associations between them. This network is clustered into four groups, conjectured as clustered from one

population and three sub-populations based on the interactions between dolphins of different sex and ages [144]. The dolphins were observed between 1994 and 2001. iii) A network of Division I matches of American Football during a regular season in the fall of 2000 [124] (American football). Nodes represent teams, and edges are games between teams. Teams can be clustered according to their football college conference memberships. iv) A network of books on US politics published around the 2004 presidential election and sold by an online bookseller [145] (Political books). Nodes represent the books, and the edges between books are frequent co-purchasing of books by the same buyers. Books are clustered based on their political spectrum as neural, liberal, or conservative.

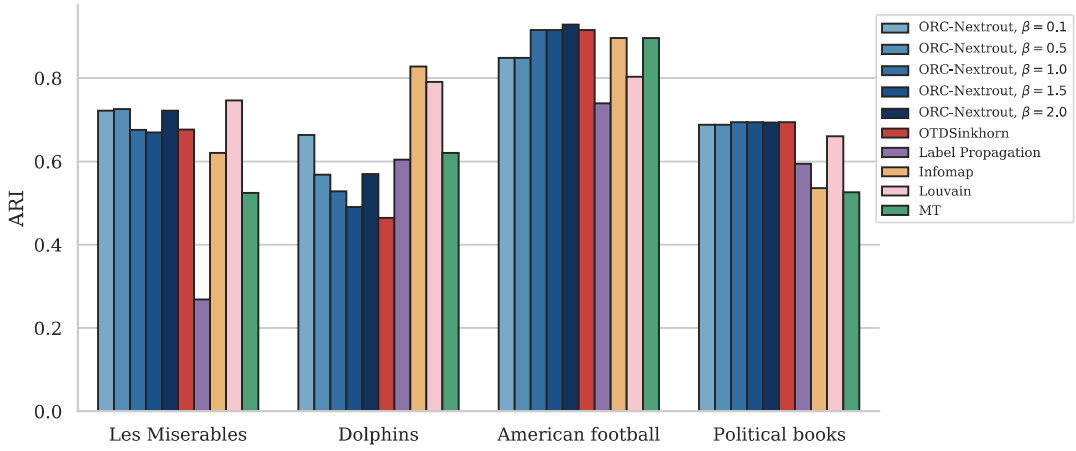


Fig. 4.5: Results on real data. Performance in terms of recovering communities using metadata information is calculated in terms of the ARI score. ORC-Nextrout shows competing results against all methods with different optimal β across datasets.

OT-based algorithms outperform other community detection algorithms in detecting communities aligned with node metadata for two of the four studied datasets, as shown in Figure 4.5. In particular, ORC-Nextrout has the highest accuracy performance considering the best performing β in these cases. The impact of tuning this parameter is noticeable from these plots, as the best-performing value varies across datasets. In Les Miserables and Dolphins networks, $\beta < 1$ has better performance, while in American Football the best performing value is for $\beta > 1$. Performance is similar across OT-based methods in the Political books network.

In Figure 4.6 we show the communities detected by the best-performing ORC-Nextrout version together with OTDSinkhorn and Infomap in Les Miserables and Political books (see Chapter 8 for the remaining datasets). Focusing on Les Miserables, we see how ORC-Nextrout successfully detects three characters in the green communities, in particular a highly connected node in the center of the figure (in dark green). Notice that these are placed in the same community (pink or black) by OTDSinkhorn. Thus ORC-Nextrout achieves a higher ARI than OTDSinkhorn. Both OT-based approaches retrieve well communities exhibiting clustering patterns with many connections within the community. Instead, they both divide the communities with a hub and spokes structure due to the lack of common connections within the group.

The communities detected in both datasets highlight the tendency of OT-based methods to extract a larger number of communities than those observed from node metadata. Among these extra communities, some are made of a few nodes (e.g. the light-blue and violet), while others are made of one isolated node each (highlighted in black). This is related to the fact that OT-based methods perform particularly well for networks with internally densely connected community structures, but may be weaker for community structures that are sparsely connected [101]. One could potentially assign these nodes to larger communities,

for instance, by preferential attachment as done in [101], thus in practice reducing the number of communities. Devising a principled method or criterion to do this automatically is an interesting topic for future work.

Dataset	N	E	K	**AvgDeg**	**AvgBtw**	**AvgClust**
Les Miserables	77	254	11	6.6	0.0219	0.5731
Dolphins	62	159	4	5.1	0.0393	0.2590
American football	115	613	12	10.7	0.0133	0.4032
Political books	105	441	3	8.4	0.0202	0.4875

Tab. 4.1: **Real networks description.** We report statistics for the real networks used in our experiments. N and E denote the number of nodes and edges, respectively. K is the number of communities in the ground truth data. AvgDeg, AvgBtw and AvgClust are the average degree, betweenness centrality and average clustering coefficient, respectively.

This tendency is further corroborated by the fact that OT-based algorithms recover robustly the two communities that are mostly assortative (blue and brown in the figure) in the Political books network, while they struggle to recover the disassortative community depicted in the center (violet). This community has several connections with nodes in the other two communities and has been separated into smaller groups by OT-based approaches, as described above. This also highlights the need for methods that are robust against situations where mixed connectivity patterns arise, i.e. a combination of assortative and disassortative communities coexisting in a network.

4.3.3 Two tests on semi-synthetic networks

To further investigate the different performance gaps between the various approaches, we expand the comparison between the OT-based methods and Infomap on two semi-synthetic scenarios generated from Les-Miserables (Figure 4.6a) and Dolphins datasets, where the largest ARI differences were observed. Specifically, we add random noise to the existing set of connections to understand if the performance gap can also be observed in more challenging scenarios. We add noise to the real data in two different ways (more details can be found in Section 8.3):

1. *Flipping entries:* from a given network, we generate a new one by flipping R entries of its adjacency matrix A uniformly at random. This means that if $A_{ij} = 1$, this is changed to $A_{ij} = 0$, and vice versa. The flipping of an entry A_{ij} occurs with probability $p = 0.1$.

2. *Removing intra-community edges:* from a given network, we build a new one with the same inter-community structure but modified intra-community one by removing R within-community edges uniformly at random, based on the ground truth communities. To avoid generating disconnected networks, we only sample edges that are not connected to any leaves.

Both of these procedures make inference harder, but they act differently. The first process is meant to add random noise independently of the community structure (flips are made uniformly at random), while the second aims at targeting the community structure by weakening the assortative structure. We choose R to be $r \times |V|^2$ for the first test and $r \times |E|$ for the second, where we vary $r \in [0, 1]$ to study the impact of these parameters on inferring the communities

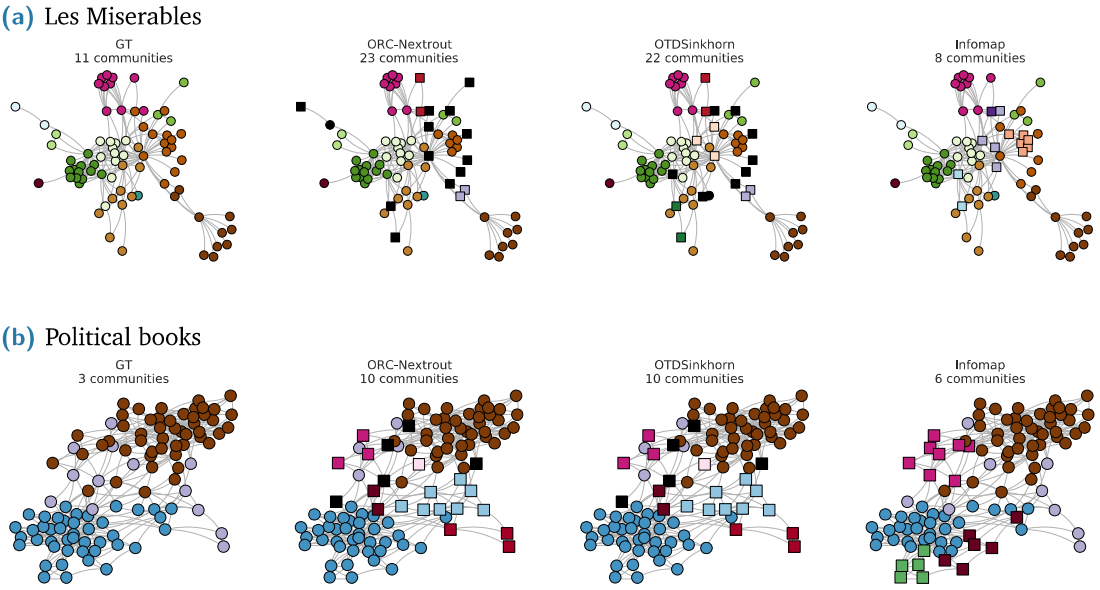


Fig. 4.6: Communities in real networks. We show the communities inferred for Les Miserables (a) and Political books (b) by ORC-Nextrout ($\beta = 0.5, 0.1$ for top and bottom rows respectively), OTDSinkhorn and Infomap and compare against those extracted using node attributes (GT). The visualization layout is given by the *Fruchterman-Reingold force-directed* algorithm [146], therefore, groups of well-connected nodes are located close to each other. Dark nodes represent individual nodes who are assigned to isolated communities by OT-based methods. Square-shaped markers denote nodes assigned to communities different from those obtained from node metadata.

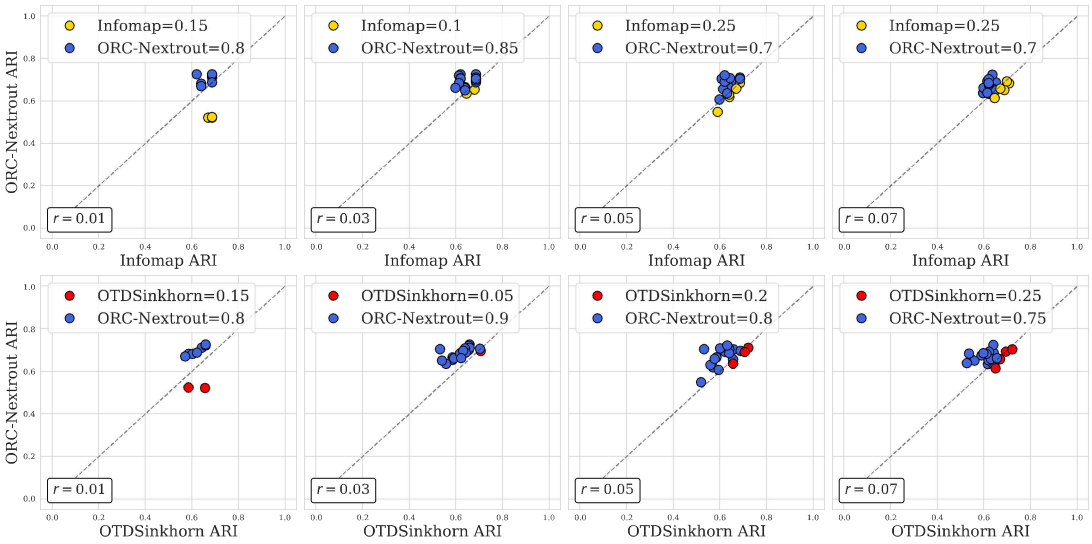


Fig. 4.7: Removing intra-community edges test on Les Miserables data. Markers correspond to 20 instances of semi-synthetic networks generated from real data. Their (x, y) coordinates are ARI scores of the indicated method on the axes. Colors are given by the best performing algorithm, e.g. if $x > y$, the color of the associated method to x is chosen. The legend shows the percentage of times that the corresponding method outperforms the other. The parameter r describes the proportion of entries for the adjacency matrix A that have been changed. This increases from left to right.

as measured by the ARI score. We generate 20 samples for each of these two mechanisms built using the Les-Miserables dataset ($|V| = 77$, $|E| = 254$) shown in Figure 4.6a.

We show the results obtained in the test of removing intra-community edges as scatter plots in Figure 4.7. We use these plots to compare the algorithms on trial-by-trial community detection tasks: a point on each plot is an instance of a semi-synthetic network with (x, y) coordinates being the ARI scores of ORC-Nextrout (y) and either OTDSinkhorn or Infomap (x). If $y > x$, then ORC-Nextrout outperforms the other method in this particular dataset (blue markers), and vice versa if $x < y$. We then compute the percentage of times that ORC-Nextrout outperforms the other (as indicated in the legend). We find that ORC-Nextrout outperforms both OTDSinkhorn and Infomap clearly and consistently across different values of r ranging from $r = 0.01$ ($R \approx 3$) to $r = 0.07$ ($R \approx 20$). In at least 70% of the cases, ORC-Nextrout gives more accurate results than the other two algorithms. This suggests that ORC-Nextrout is more robust against perturbations of the community structure. Similar patterns are seen in the case where edges are removed at random and in semi-synthetic networks generated from the Dolphins dataset, see Section 8.3.

4.4 Discussion

Community detection on networks is a relevant and challenging open area of research. Several methods have been proposed to tackle this issue, with no "best algorithm" that fits well for every type of data. We focused here on a recent line of work that exploits principles from Optimal Transport theory combined with the geometric concept of Ollivier-Ricci curvature applied to discrete graphs. Our method is flexible in that it tunes between different transportation regimes to extract the information necessary to compute the OR curvature on edges. On synthetic data, our model is able to identify communities more robustly than other OT-based

methods based on the standard Wasserstein distance in the regime where inference is not trivial. On real data, our model shows better or comparable performance in recovering community structure aligned with node metadata compared to other approaches, thanks to the ability to tune the parameter β.

A relevant advantage of OT-based methods is that the number of communities is automatically learned from data, contrarily to other approaches that need this as an input parameter. In this respect, our model has the tendency to overestimate this number, similarly to other OT-based methods. Understanding how to properly incorporate small-size communities into larger ones in a principled and automatic way is an interesting topic for future work. Similarly, it would be interesting to quantify the extent to which various β capture different network topologies. To address this, one could, for instance, use methods to calculate the structural distance between networks [147] and correlate this against the values of the best performing β. Similarly, as our approach allows obtaining different sets of weights on edges, depending on β, it would be interesting to investigate how different values of this parameter impact network properties that are governed by the weight distribution, such as multi-fractality [148].

An interesting research direction from this method is to account for the multilayer case. Expanding our model to accommodate different traffic regimes for each edge type, similarly to what proposed in [132, 149], could enhance the accuracy of the detected communities. While in chapter 4 our focus has been on the flexibility of solving various transportation regimes to compute the OR curvature, alternative insights could be gleaned by varying the input mass distributions across nodes' neighborhoods, as proposed in [110].

4.5 Methods

4.5.1 Optimal Transport Formulation (Wasserstein Distance)

Consider the probability distributions q that take pairs of vertices and also satisfy the constraints $\sum_i q_{ij} = m_j, \sum_j q_{ij} = m_i$. In other words, these are the joint distributions whose marginals are m_i and m_j. We call these distributions *transport plans* between m_i and m_j. The Optimal Transport problem we are interested in is that of finding a transport plan q^* that minimizes the quantity $\sum_{i\sim j} q_{ij} d_{ij}$, where $i \sim j$ means that the nodes i and j are neighbors and d_{ij} is the cost of transporting mass from i to j, e.g. the distance between these two nodes. The quantity $\mathcal{W}_\beta(m_i, m_j, d) := \sum_{i,j} q^*_{ij} d_{ij}$, defined for this optimal q^*, is the *Wasserstein distance* between m_i and m_j.

4.5.2 Ollivier-Ricci curvature

It was recently proved [126, 28] that solutions of the optimal transport problem previously stated can be found by turning that problem into a system of differential equations. This section is dedicated to describe this dynamical formulation.

Let $G = (V, E, W)$ be a weighted graph, with N the number of nodes and E the number of edges in G. Let $\mathbf{B}$ be the *signed incidence* matrix of G. Let f^+ and f^- be two N-dimensional discrete distributions such that $\sum_{i\in V} f_i = 0$ for $f = f^+ - f^-$; let $\mu(t) \in \mathbb{R}^E$ and $u(t) \in \mathbb{R}^N$ be two time-dependent functions defined on edges and nodes, respectively. The discrete *Dynamical Monge-Kantorovich model* can be written as:

$$f_i = \sum_e B_{ie} \frac{\mu_e(t)}{w_e} \sum_j B_{ej}\, u_j(t)\,, \tag{4.3}$$

$$\mu_e'(t) = \left[\frac{\mu_e(t)}{w_e} \left| \sum_j B_{ej}\, u_j(t) \right| \right]^{\beta} - \mu_e(t)\,, \tag{4.4}$$

$$\mu_e(0) > 0\,, \tag{4.5}$$

where $|\cdot|$ is the absolute value element-wise. Equation (4.3) corresponds to Kirchhoff's law, Eq. (4.4) is the discrete dynamics with β a traffic rate controlling the different routing optimization mechanisms; Eq. (4.5) is the initial distribution for the edge conductivities.

For $\beta = 1$ the dynamical system described by Eqs. (4.3)-(4.5) is known to reach a steady state, i.e., the updates of μ_e and u_e converge to stationary functions μ^* and u^* as t increases. The flux function q defined as $q_e^* := \mu_e^* |u_i^* - u_j^*| / w_e$ is the solution of the optimal transport problem presented in the previous section. Notice that μ and u depend on the chosen traffic rate β, and thus, so does $q = q(\beta)$. Therefore, we can introduce a generalized version of the distance $\mathcal{W}$:

$$\mathcal{W}_\beta(m_i, m_j, w) := \sum_{i,j} q_{ij}^*(\beta)\; w_{ij}.$$

We then redefine the proposed Ollivier-Ricci curvature as:

$$\kappa_\beta(i,j) := 1 - \frac{\mathcal{W}_\beta(m_i, m_j, w)}{d_{ij}}.$$

Probability distributions on neighborhoods

ORC-Nextrout takes in input a graph and a forcing term. While the graph encapsulates the neighborhood information provided by the nodes i and j, the forcing function is related to the distributions that one needs to transport. Analogously to what was proposed by [100], we define this graph to be the weighted *complete bipartite* $B_{ij} = (V_{ij}, E_{ij}, \omega_{ij})$. The weights in ω_{ij} change iteratively based on the curvature. Notice that a bipartite graph must satisfy $\mathcal{N}(i) \cap \mathcal{N}(j) = \varnothing$, which does not hold true if i and j have common neighbors (this is always the case since $i \in \mathcal{N}(j)$). Nonetheless, this condition does not have great repercussions in the solution of the optimal transport problem since the weights corresponding to these edges (of the form (i,i)) are equal to 0. As for the forcing function, we define it to be $f := f^+ - f^- = m_i - m_j$.

4.5.3 Other methods

To evaluate the performance of ORC-Nextrout, we compare with some of the well-established community detection algorithms including: Infomap[95], MULTITENSOR[91] (MT), discrete Ricci flow[100] (OTDSinkhorn), Label propagation[92] and Louvain[125]. We briefly describe each of these algorithms as follows;

- The *Discrete Ricci flow* (here addressed as OTDSinkhorn) [100] is an iterative node clustering algorithm that deforms edge weights as time progresses, by shrinking sparsely traveled links and stretching heavily traveled edges. These edge weights are iteratively updated based on neighborhood transportation Wasserstein costs, similarly to what is proposed in this manuscript. After a predefined number of iterations, heavily traveled

links are removed from the graph. Communities are then obtained as the connected components of this modified network.

- MULTITENSOR (MT) [91] is an algorithm to find communities in multilayer networks. It is a probabilistic model with latent variables regulating community structure and runs with a complexity of $O(EK)$ with assortative structure (as we consider here), where K is the number of communities. This model assumes that the nodes inside the communities can belong to multiple groups (mixed-membership). In this implementation we use their validity for single layer networks (a particular case of a multilayer network).
- *Infomap*[95] employs information theoretic approach for community detection. This method uses the map equation to attend patterns of flow on a network. This flow is simulated using random walkers' traversed paths. Based on the theoretic description of these paths, nodes with quick information flow are then clustered into the same groups. The algorithm runs in $O(E)$. In the experiments, we fix the number of random initialization of the random walkers to be equal to 10. The inferred partition is then the one minimizing the entropy.
- *Label propagation*[92] assigns each node to the same community as the majority of its neighbors. Its working principle start by initializing each node with a distinct label and converges when every node has same label as the majority of its neighboring node. The algorithm has a complexity scaling as $O(E)$.
- *Louvain*[125] is a fast algorithm used to find communities on networks by maximizing the modularity of the associated partitions. It consists of two phases. First, it assigns every node on the network into a different community. Then, it aggregates nodes and neighbors based on gains of modularity. This last step is repeated until no further improvement can be achieved.

5 Urban transportation networks and optimal transport-based infrastructures: similarity and economy of scale

Designing and optimizing the structure of urban transportation networks is a challenging task. In this chapter, we propose a method inspired by optimal transport theory and the principle of economy of scale that uses little information in input to generate structures that are similar to those of public transportation networks. Contrarily to standard approaches, we do not assume any initial backbone network infrastructure but rather extract this directly from a continuous space using only a few origin and destination points. Analyzing a set of urban train, tram and subway networks, we find a noteworthy degree of similarity in several of the studied cases between simulated and real infrastructures. By tuning one parameter, our method can simulate a range of different subway, tram and train networks that can be further used to suggest possible improvements in terms of relevant transportation properties. Similar to what we presented in Chapter 4, outputs of our algorithm provide naturally a principled quantitative measure of similarity between two networks that can be used to automatize the selection of similar simulated networks.

5.1 Introduction & Motivation

Transportation networks are a fundamental part of a city's infrastructure. Their design impacts the efficiency with which the system is operated, hence, they should follow optimal principles while being constrained by limitations like budget or physical obstacles. Existing approaches for studying the quality of network design often rely on the analysis of the topological network properties, and relate them to optimal features like transportation cost, efficiency or robustness. These analyses are usually made *a posteriori*, only once the network has been constructed, and thus only resulting properties can be analyzed [150, 151]. A different approach is that of posing *a priori* a principled optimization setup, where one defines a cost function that a network should minimize under a set of constraints, and then searches for optimal solutions in terms of network topologies.

Numerous studies have explored this approach in biological networks, transportation networks, etc. [152, 153]. However, most of these methods rely on an existing backbone of a network infrastructure that can be optimized in terms of traffic distribution [154, 155] but do not consider the possibility of building the network from scratch, starting from a limited set of nodes. Alternatively, as optimizing over all possible topologies is difficult, one can investigate only various simple shapes from a predetermined set of possible geometries [156, 157, 158] or rely on heuristics [159, 160]. Recently, Kay et al. [161] presented a two-step agent-based model that replicates biologically-grown networks and proposes them as a template for urban design. Further, the lack of a principled metric to measure the similarities between an observed network and a simulated one poses the problem of making this evaluation effectively.

In this chapter, as another important application of Nextrout, we show that urban transportation systems can exhibit underlying network topologies similar to those that follow optimality principles as defined in optimal transport theory. Specifically, we propose a model to characterize real transportation networks based on a simple optimal transport framework, similar to what is observed in biological systems like the slime mold *Physarum polycephalum*, which adapts its network structure to reach food patches in an optimal way. A previous study by Tero et al. [162] shows how this mold forms networks with comparable optimal transportation properties, e.g. efficiency and cost, to those of the Tokyo rail system, but provided no rigorous quantitative definition of network similarity beyond measuring these properties. We empirically validate our approach with a systematic characterization of the structure of several urban transportation networks and propose a rigorous definition of similarity in terms of optimal transport theory.

Urban transportation networks often exhibit different network structures based on the goals of network designers. For instance, some networks focus on connecting people living in the outer layers of the city to the city core, while others prefer to develop a robust infrastructure servicing the core [163].

Several studies have focused on analyzing properties like scaling laws and network connectivity [164, 165, 166], which are indications of an underlying optimality mechanism that these networks might follow to make a city efficient, both in reduced infrastructure costs per capita and in increased productivity. However, our understanding of what optimality principles are captured in real transportation networks is incomplete. In fact, studying network properties could only partially explain the underlying mechanisms regulating network design, as each property captures a different aspect.

Here, we take a different approach and build the network from scratch while comparing it with the real ones observed from data, starting with only a few shared nodes in input. Specifically, we model network structures observed in urban transportation networks by adapting a classical optimal transport framework to simulate a network-design problem dependent on realistic travel demand settings and using little information in input. We then compare the resulting networks with those observed from real data and assess their similarity.

Importantly, the model can simulate different optimal strategies by tuning a parameter β, which interpolates between minimizing infrastructural and operating costs, in a similar fashion as in the principle of economy of scale, a fundamental concept in economy that establishes the relationship between growth and production costs. This principle affirms that, as the quantity of produced units rises, the average cost per unit of production declines [167]. This impacts the balance between the costs of producing and that of maintaining and operating the network. On one hand, this allows to simulate optimal networks that resemble those observed in real transportation systems more closely, as we tune β. On the other hand, by comparing the networks resulting for various values of β with those observed from real data, we can also assess the impact of the two types of cost in the design of various urban infrastructures.

We use this model to analyze several transportation networks from 18 cities and a national rail network [168, 169, 170]. Despite the complex nature of the mechanisms driving the design of transportation networks, we observe that multiple of the studied urban transportation networks follow a surprisingly similar topological pattern, as noticed in biological systems. We observed that in several cases, the optimal networks obtained with our approach have similar cost and performance to those observed in real ones.

5.2 Modeling network design in transportation networks

Consider an urban area where a set of points of interest are located in certain positions in space. These may correspond to a combination of central and peripheral points where people work and live. The goal is to connect them by building a transportation network under the perspectives of an optimality criterion, based on the minimization of a cost-based energy functional. At this point, we do not observe any network, but are rather free to use the whole space where the urban area is located, i.e. a 2D surface. From this, we need to select a set of points and edges connecting them, in other words, a network. In Figure 5.1a–b we illustrate the problem setup for the subway network in Rome, where green and red markers denote an example set of such reference points and the lines denote edges in the observed metro network infrastructure. In the same figure, we show how existing stations are placed across urban areas with different population densities, as the evolution of subway networks often reflects the evolution of population and activity densities [171].

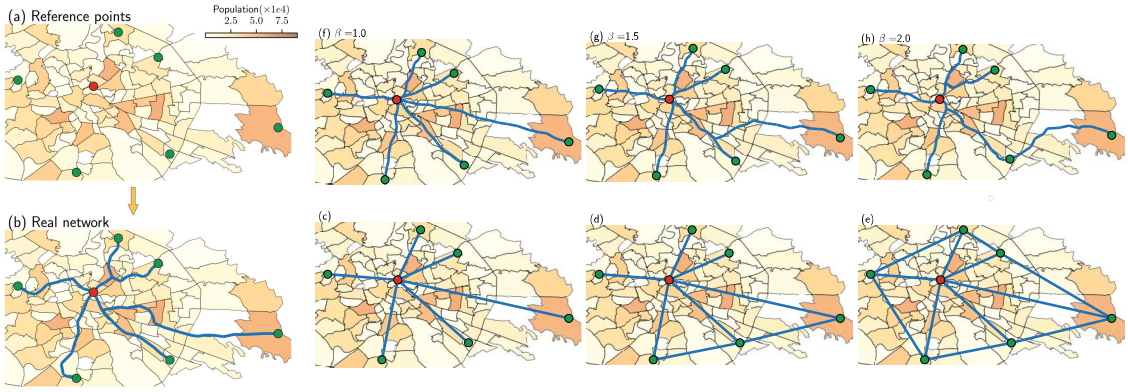


Fig. 5.1: Problem setup for the subway network of Rome. (a) Given a set of real latitude-longitude coordinates denoting origins (green) and destinations (red), we aim to build a network structure that resembles well the observed public transportation network connecting those points, as in (b). (c)-(e) Intuitive ways to build a network structure by connecting origins and destinations, versus networks extracted with our optimal transport-based method in (f)-(h). The only known information is the set of six origins (O) and one destination (D). We capture different optimization mechanisms by tuning the β parameter: in (f), the network is the shortest path-like structure, while in (g) and (h) we show examples of branched transportation schemes. This information is added to the population density across multiple urban areas (2019) [172], where darker (lighter) colors indicate higher (lower) densities.

In general, there are many choices for designing the network. For instance, in Figure 5.1c–e we show three examples of intuitive shortest-path-like minimization solutions for the settings shown in Figure 5.1a. These are however quite different from the observed network in Figure 5.1b. The question we address is what network design principle is producing simulated networks that are more similar to those observed in real urban networks.

Optimality could be defined in various ways depending on the network engineers' and designers' goals, but generally, it is not known what principles they used when building the network. Instead, we want to assess this by observing real data of transportation networks and fitting them with a flexible and computationally efficient optimization setup guided by optimal transport theory. In this context, a well-defined cost-based functional combines aspects that are critical for a transportation network: the cost of building the infrastructure and that of operating the network, in terms of power dissipation. This is relevant in scenarios where we expect infrastructures to be regulated by energy-saving requirements and the principle

of economy of scale, where it is more convenient to consolidate traffic into fewer and larger edges. We expect this to be a reasonable assumption in urban transportation networks.

As with any other natural or urban system, we do not know *a priori* what (if any) is the functional being optimized in the network under study. In fact, many of these systems (e.g., metro and tram) are built in phases [163], where the design of an initial backbone structure is followed by several expansion steps, which may lead to suboptimal structures. However, our model allows considering, among the many possible choices, a simple but yet principled mechanism of optimality.

By measuring the degree of similarity of networks that follow these principles with existing urban transportation networks, we can assess if the observed ones can be explained by this simple mechanism. And if not, we can point out alternative infrastructures that can be better in terms of certain relevant network properties, e.g. total path length. Our setting is simple because we consider only a limited input (few nodes that need to exist, e.g. main origin and destination stations), but otherwise do not consider any other constraint, besides main physical laws such as conservation of mass, and let the model select nodes and edges from a two-dimensional space where it can be optimal to drive passengers, tuning only one parameter.

For this, we adopt the formalism recently developed by Facca et al. [18, 19, 27] that generalizes to a continuous space the original idea of Tero et al. [14]. In particular, this allows starting with only a set of relatively few origin and destination nodes in input, and then designing a network by exploring the 2D surface, i.e. without the need of an initial existing backbone. The idea is inspired by the behavior of the slime mold *Physarum polycephalum*, which dynamically builds a network-like body shape when foraging. One can thus consider a dynamics for the two main quantities involved, flows and conductivities, that implements this mechanism at any point in space. The stationary solution of this dynamics corresponds to the minimizer of a Lyapunov cost in a standard optimization setup, which has a nice interpretation in terms of infrastructure and operating transportation costs.

From these solutions, one can then automatically extract optimal network structures using the approach presented in [173]. From now onwards, we refer to the algorithmic implementation of this approach as “Nextrout”.

Having introduced the main problem and ideas, we now briefly refer to the model. Consider a surface in 2D and a set of points on it. Specifically, we denote a set of origins and destinations as f^+ and f^-, respectively. These contain the reference points where people enter and exit the transportation network. By defining $f = f^+ - f^-$, mass conservation can be enforced with the constraint $\int f dx = 0$. The two main quantities of interest are denoted with $\mu(x,t)$, the transport density (or conductivity), and $u(x,t)$ the transport potential. The former can be seen as a quantity proportional to the size of a network edge, while the latter determines the fluxes traveling along them. The dynamical equations in this continuous setting

$$-\nabla \cdot (\mu(t,x)\nabla u(t,x)) = f, \tag{5.1}$$

$$\frac{\partial \mu(t,x)}{\partial t} = (\mu(t,x)\nabla u(t,x))^{\beta} - \mu(t,x), \tag{5.2}$$

$$\mu(0,x) = \mu_0(x) > 0. \tag{5.3}$$

Equation (1) determines the spatial balance of the flux, assumed to be governed by the Fick-Poiseuille flux as $q = -\mu\nabla u$; Equation (5.2) enforces optimal solutions, and represents the *Physarum polycephalum* dynamics in the continuous domain; Equation (5.3) is the initial

condition. The parameter β captures different optimization mechanisms: $\beta < 1$ enforces congested transportation, $\beta = 1$ is the shortest path-like and $\beta > 1$ is branched transportation. In Figure 5.1f–h we show examples of different optimal configurations, with $\beta = 1$, $\beta = 1.5$ and $\beta = 2.0$. Here, we consider the cases $1 < \beta \leq 2$, where the approximate support of the conductivity μ displays a network-like structure. Under the lenses of a network, the conductivities can be viewed as the traffic capacities on the edges, hence Equation (5.3) defines how the initial traffic capacities are distributed along the network, while Eq. Equation (5.2) describes how these capacities evolve in response to the fluxes. As time evolves (i.e., $\lim_{t\to\infty}$), the equilibrium solution pair (μ^*, u^*) is reached. In [28, 27] the authors show that under certain assumptions, this equilibrium solution is a minimizer of the functional

$$\mathcal{L}(\mu, u) = \frac{1}{2}\int \mu|\nabla u|^2 dx + \int \frac{\beta}{2-\beta}\mu^{\frac{2-\beta}{\beta}}. \tag{5.4}$$

This can be interpreted as the network transportation cost, where the first term is a network operating cost (or power dissipation, it is the Dirichlet energy to the solution of the first PDE), while the second is a non-linear cost to build the infrastructure. When $\beta > 1$, this second term corresponds to a principle of economy of scale, where it is more convenient to consolidate traffic into fewer (but larger) edges. This is the scenario we consider here. By changing β, one can tune their relative contribution to the total transportation cost, thus tuning the impact of the principle of economy of scale and how much concentrated path trajectories are. Besides being relevant for urban transportation, this strategy seems to be a fundamental mechanism in various natural systems, e.g., tree branches and roots, blood vessels or river networks [174, 175].

Alternative approaches can be considered to design a network infrastructure from simple mechanisms. Examples are cost-benefit analysis [176], maximizing for efficiency [177] accounting for paths and flows of passengers, or minimizing the total length, as in the Euclidean minimum spanning tree problem. In discrete settings, when an initial network backbone is given, the cost in Equation (5.4) has been shown to be implicitly related to the total path length minimization accounting for the passengers' trajectories [129]. One main difference between ours and these types of approaches is that we focus on a continuous space (as opposed to discrete settings) where the only necessary input is a set of origins and destinations, but otherwise no initial backbone network is given. This enables the design of a network from scratch, simulating where nodes and edges should be located in space to minimize the cost.

Once the optimal (μ^*, u^*) are obtained, one can then use the model described in [173] to extract a final network structure, i.e. a set of nodes, a set of edges connecting them and their weights proportional to the conductivities. This can then be compared with the one observed from real data and repeated for various values of β. An example is shown in Figure 5.5a–d, where, as we increase β, one can notice how the network infrastructure evolves from a shortest path-like ($\beta = 1$) to a branching topology, where two branches ($\beta = 1.5$) are created to lower the cost of building the infrastructure. In particular, the southern branch in Figure 5.5b resembles an analogous one observed in the real subway network of Rome. As β increases to 2, this branch disappears to build a unique path that connects two destination points in the southeast side of the city, further lowering the cost to build the infrastructure. However, at this extreme value, the network is now less similar to the real one.

It is important to remark that in our setting, besides the parameter β, the other input quantities that need to be specified are origins and destinations via the function f. By imposing non-zero entries to this function, a user automatically selects a set of nodes that will be necessarily present in the output network. Otherwise, no other set of nodes or edges needs to be given,

but is rather automatically learned by solving the optimization problem described above. This implies that similarity between simulated and observed networks trivially increases as we add more non-zero terms in f, as shown in Figure 5.5d. Here we consider the non-trivial scenario where we fix only a small number of origins and destinations, as described in more detail below.

5.2.1 Selecting Origin and Destination points

As we aim at extracting a network, our problem starts by defining a set of origins and destinations (OD) points in the space with coordinates (x, y), where passengers might enter or exit. This choice necessarily impacts the output network, as the optimization problem depends on it. Ideally, one could reframe it by including OD pairs as variables to be optimized along with conductivities and fluxes. But this becomes a different and more complex problem and it is not clear how to solve it. Here instead, we focus on the optimization setting introduced above and treat OD pairs as fixed in input. While we limit selection to a small number of points, specifically one destination and few origins, it is important to decide where to place these input nodes in space.

There are no universal criteria to define what are the most relevant points where city planners should add a stop to accommodate traffic when designing a network. However, existing infrastructures often evolve reflecting needs such as population increase or land usage [178]. With this in mind, we can assume that at least some of the existing nodes have already been placed in positions relevant to transportation needs. Hence, we select OD pairs from important nodes as observed in existing urban networks.

Specifically, we use centrality measures obtained from the original network: nodes with the smallest and highest centrality are assigned as origin and destination nodes, respectively. These measures might reflect the choice on where to place new stations that are often made by urban planners or transportation engineers, usually based on a variety of factors, such as population density, land use patterns or available funding [179]. For instance, in Figure 5.2a we observe higher population density in peripheral regions, where the stations with lower centrality are located, whereas those with higher centrality are located towards the center, with lower population density. In the same figure, we show node sizes as proportional to the annual traffic in each station, as measured in 2019 [180]. In this example, the highest traffic corresponds to the station with the highest degree centrality, thus reinforcing the choice of that node as a destination.

Another possibility is to incorporate urban features by using some measure of attractiveness, which takes into account the densities of Points of Interest (POI) and population in a given urban area, an approach also used in other works [177]. However, this strategy may not be scalable, as it requires the integration of several datasets, whereas using network centralities can be done automatically from the observed network data at no additional cost. We show an example of this for the metro network of Rome, to assess the extent to which these two strategies align. We notice that the high centrality nodes found by the two criteria are located nearby geographically and yield similar simulated networks, see Figure 5.2. Hence, we adopt the centrality criteria as a good approximation for attractiveness to select origins and destinations in all the networks investigated here.

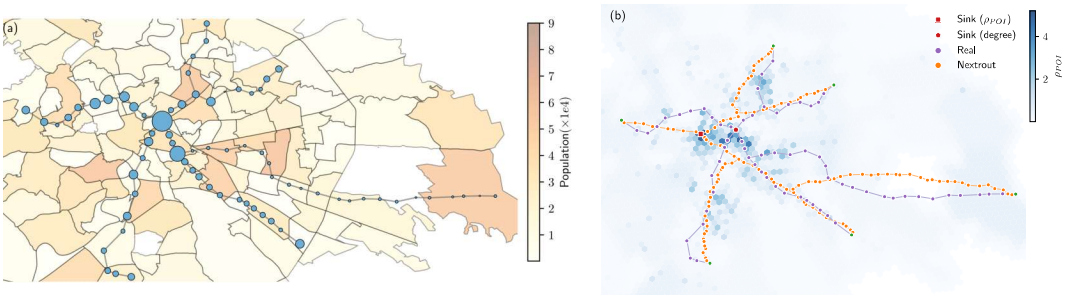


Fig. 5.2: Comparing criteria to select origin and destination nodes. We compare two criteria to select the input nodes that we give to our algorithm, one based on a topological property (centrality) and one based on population and point-of-interest densities (ρ_{POI}). (a) Stations with the highest and lowest annual traffic (2019), proportional to the node sizes, over the population distribution in Rome (in multiples of 10000)). The degree centrality of nodes in the observed is related to the traffic at stations. In particular, the central node with the highest degree (destination), has also the highest annual traffic. (b) Density accounting for population and distribution of points-of-interest (ρ_{POI}), darker colors mean higher values, i.e. regions of higher relevance for transportation. When accounting for the Points of Interest (POI), we notice that the city center presents higher density (ρ_{POI}) compared to the peripheral areas, therefore using centralities to select destinations is an approximation to real demands.

5.3 Investigating the similarity of optimal simulated networks and the observed transportation systems

We apply the proposed dynamics to empirical data collected from 18 different cities in multiple geographical regions around the world. For each city, we selected various available types of public transportation systems, such as rail, subway and tram, keeping the largest connected component. The networks considered in this chapter have a few loops, as the dynamics can only retrieve loopless structures in the regime where network extraction is meaningful [173]. These networks could be seen as phase I in the classification of [163], i.e. the initial phase where a backbone infrastructure is built, before a later evolution where further additional links are added through time. We expect these to be more likely to follow a global optimization criteria as the one formulated in our model (as opposed to other greedy heuristics for later extension phases). We thus measure the loop ratio $L_{ratio} = n_L/E$ as the number of loops divided by the number of edges and select networks with a low ratio, i.e. with $L_{ratio} < 0.2$ (see Methods for more details). One could in principle recover loopy structures by employing numerical schemes, e.g. superposition of different outputs [30], but this is not the main focus of this work. Instead, we point towards directions on the loop recover perspective, presented later in this chapter, by exploring an example of a more complex network structure as the New York subway.

The applied Optimal Transport (OT) dynamics successfully describe the transportation network structures observed in different cities at a macroscopic level. While the selected transportation networks have different topologies and include multiple transportation modes, the networks reconstructed by Nextrout with only little information in input show a significant degree of similarity with the real ones (see Figure 5.3a–c) for several of the studied cases, as evidenced by different similarity measures that we calculated to compare the topology of the simulated networks against the real ones, see Figure 5.4 and next sections for details. This suggests the existence of simple universal optimality rules captured by the so-called Dynamic Monge-Kantorovich (DMK) dynamics for the modeling of urban transportation rail networks, similar to what has been observed for the behavior of the slime mold *Physarum polycephalum*. Notice

that in principle one can increase this similarity further, by simply adding more information in input in terms of origin and destinations (see Figure 5.5d for an example with multiple origins and destinations). However, here we are interested in recovering macroscopic structure in a more challenging scenario where input information is strictly limited to one central destination and few peripheral origins.

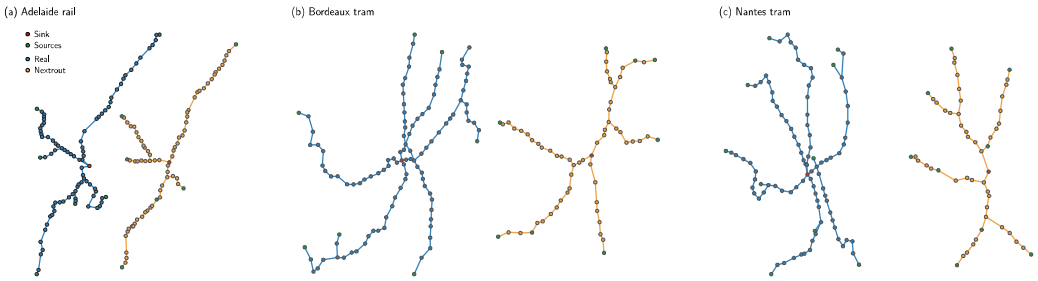


Fig. 5.3: Example of different network topologies generated by Nextrout (yellow), plotted against the corresponding real network (blue). Green nodes represent those chosen as origins (O), whilst red nodes correspond to the destinations (D). (a) Adelaide rail network, with N=28 nodes, $O = 6$ and $D = 1$. (b) Bordeaux tram network, with $N = 108$ nodes, $O = 8$ and $D = 1$. (c) Nantes tram network with N=97 nodes, O=10 and D=1.

Beyond a qualitative visual comparison, we explore how our simulated networks score in terms of core network properties relevant for transportation compared to the real networks. For this, we consider the cost, the total path length l, the distribution of traffic and the density of branching points (or bifurcations; here we use the two terms interchangeably), for both extracted and original networks. Similar to Tero [162], we define the cost as the total length of the network (TL), i.e. the total number of edges.

Passengers may not always take the shortest path, but may rather consolidate on fewer main arteries (e.g. to minimize the number of stops or connections) [149], a behavior that can be captured by a DMK discrete dynamics (built-in the filtering step of nextrout) by varying β and extracting the flows u_e on edges, quantities proportional to the number of passengers using an edge. Hence, we consider an alternative measure of total path length as $l := \sum_{e \in E_i} \mathrm{l_e} |u_e|$, where $\mathrm{l_e}$ is the Euclidean distance. This takes into account u_e, the flow of passengers on an edge e, and its absolute value $|u_e|$ is proportional to the number of passengers traveling on an edge e, i.e. how traffic is distributed, assuming that passengers follow optimality principles to consolidate paths. This is a reasonable assumption in rail networks as the ones studied here, where the cost to build the infrastructure can be high (and thus should be minimized) and minimizing traffic congestion is not as relevant as in, e.g., road networks.

In our experiments, we extract optimal flows u_e by running the discrete DMK dynamics on the extracted and real networks, using the same sets of origins and destinations as used in the original network extraction problem, setting $\beta = 1.5$.

This information can also be used to measure the macroscopic behavior of traffic on edges, which can be measured using the Gini coefficient [181] (Gini(T_e)) on the traffic $T_e = |u_e|$. This coefficient ranges from $[0, 1]$, where the closer to 1, the more unequal is the traffic distribution on the network. Finally, we calculate the percentage of bifurcations (D_{BP}) as the fraction of nodes with degree equal to 3. In several cases, the simulated networks display similar properties as those observed on the real ones, as shown in Figure 5.4. While similarity differs depending on the property and datasets vary in their range values, we notice that most of the datasets have at least a pair of properties that have a close value between simulated and

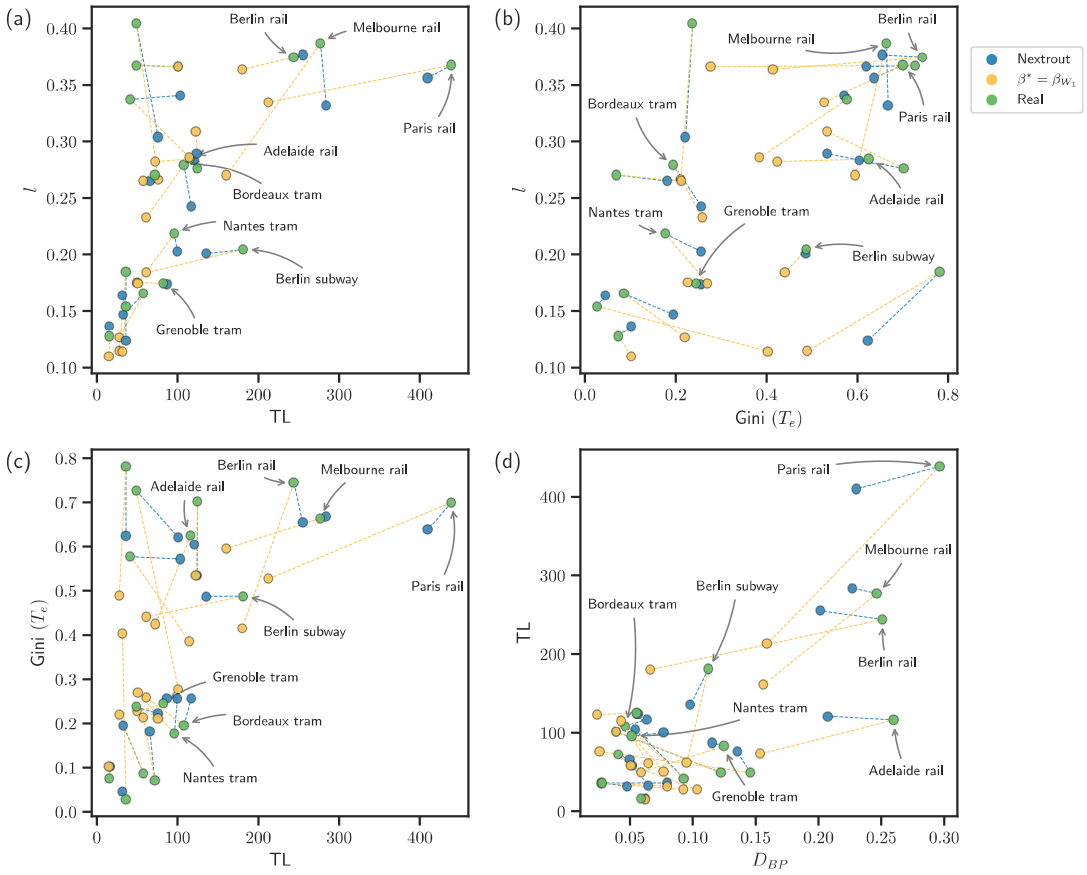


Fig. 5.4: Performance measures for real and simulated networks. Each dataset is assigned to a different color, market shapes distinguish real and simulated networks. Simulated networks are further distinguished based on the one generated via nextrout that gives the closest point in terms of the metrics plotted in the figure (circle) or the one corresponding to the best Wasserstein measure (square). (a) Cost (TL) measured in both simulated and real networks, plotted against the total path length. (b) Gini coefficient as a measure of traffic distribution, versus the total path length. (c) Traffic distribution in terms of the Cost. (d) Density of bifurcations plotted against the cost.

observed networks. For instance, in Figure 5.3a we notice that Adelaide rail has an intuitively similar path length, which is confirmed in Figure 5.4a. Furthermore, the same network shows comparable results for traffic and cost. As for the tram networks of Bordeaux and Nantes in Figure 5.3b-c, we observe similar behavior for cost and traffic.

5.3.1 Automatic selection of similar simulated networks

Our method allows extracting various simulated networks by varying the parameter β. One can select the one that more closely resembles the observed one in terms of a particular metric of interest, as shown in the previous section. However, different metrics may lead to different most similar simulated networks (i.e. different β), which may not be ideal for a practitioner willing to consider an individual simulated network that resembles well the observed one in terms of all metrics. Hence, the need for a principle automatic selection criteria for choosing the value of β.

The formalism introduced in the previous section suggests a natural way to tackle this problem by considering the Wasserstein similarity measure, a main quantity in optimal transport theory [43]. Given two graphs that need to be compared, intuitively, this measure captures the minimum "effort" required to move a certain distribution of mass from one to the other. Similar ideas based on optimal transport to measure similarity between graphs have also been proposed in recent works [182, 183].

Here, we describe our proposal for a similarity measure and thus automatic selection of β in detail. Denote the observed network with $G_1(V_1, E_1)$ and the one obtained from the model introduced in previous sections with $G_2(V_2, E_2)$, where V_i, E_i denote the set of nodes and edges, respectively, $i = 1, 2$. We consider the union graph $G_U(V_U, E_U)$, with sets of nodes $V_U = V_1 \cup V_2$ and edges $E_U = E_1 \cup E_2$. One can further assign weights $w_e \in W_U$ to the edges $e \in E$, for instance using the Euclidean distance ℓ_e between the nodes $i, j \in V_U$, where $e = (i, j)$, or simply binary values $\{0, 1\}$. Notice that the observed network G_1 may contain nodes that do not correspond exactly to nodes in G_2, because in this continuous setting the model uses all the 2D space where the original network is embedded. Only the input origin and destination nodes are guaranteed to be present in both graphs, as they are given in input to the model.

Working on this union network, we then exploit a similar setting as the one already introduced with the model to obtain a Wasserstein-based similarity measure between G_1 and G_2. Specifically, we denote with B the unsigned incidence matrix of G_U with entries $B_{ie} = +1$ if node i is a start or end point of the edge e, and 0 otherwise. Defining q_i as an indicator vector for the edges in G_U that are also in G_i, i.e. $q_{ie} = 1$ if $e \in E_i$, and $q_{ie} = 0$ otherwise, $\forall\, e \in E_U$ and $i = 1, 2$, we can set the origin and destination vectors $f = f^+ - f^-$, such that $\boldsymbol{f}^+ = B\, q_1$ and $\boldsymbol{f}^- = B\, q_2$, so that G_1 contributes to f^+ and G_2 to f^-. By running a discrete dynamics analogous to the continuous one described in Equations (5.1) to (5.3), which can be done using nextrout [173], one naturally obtains our Wasserstein similarity measure defined as:

$$W_1(G_1, G_2) = \sum_{e \in E_U} w_e\, \mu_e\,, \tag{5.5}$$

where μ_e are the optimal solutions for the conductivities on G_U and w_e is the weight of edge $e = (i, j)$. Here, we fix this to be the Euclidean distance between nodes i and j. Examples of how the Wasserstein measure changes depending on the different output networks are shown in Figure 5.5i, where we show simulated networks and report their Wasserstein measure from the observed network of the Grenoble tram ($N = 80$ nodes). Intuitively, the Wasserstein similarity captures how much “cost” is “paid” to move information between G_1 and G_2. This means that the more similar these networks are, the lower the Wasserstein is, i.e., when $G_1 = G_2$, $W_1 = 0$, and if there are no nodes connecting them, $W_1 \to \infty$. In Figure 5.5b we show an example where W_1 is higher simply because there is a disruption in one of the branches of the network, thus increasing the cost to move from the real network to $G_2 = G_{\beta=1.6}$. Notice that if similarity is chosen to be defined in terms of the cost, the closest network to the real one would be that with $\beta = 1.4$, as shown in Figure 5.5e.

We further validate this measure by comparing it with other selection criteria based on the topological properties described above and found that the simulated graph selected with the Wasserstein measure has, on average, higher similarity with the real networks compared to the other selection criteria, across various properties. In other words, it shows transportation properties that are consistently more aligned to those behold by the observed network, see Section 9.1.

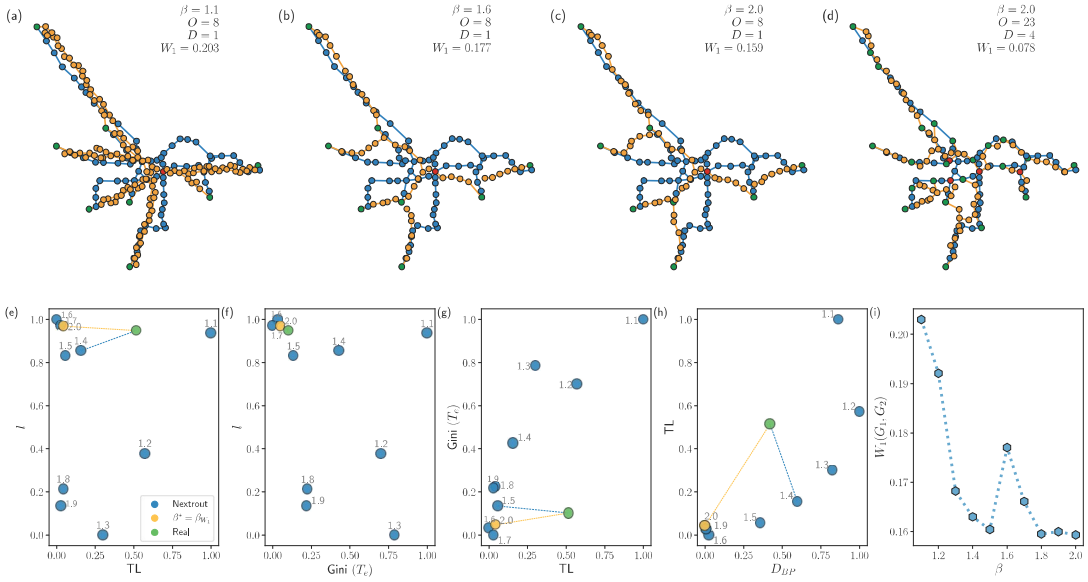


Fig. 5.5: Wasserstein similarity measure between graphs for automatic selection of β. (a-c): we select the origin nodes based on those with the smallest degree, and a unique destination as the one with the highest degree. In (d) we show how the same network changes as we set more stations as initial input (O $= 23$ origins and D $= 4$ destinations), resulting in smaller Wasserstein, at the cost of a higher amount of information given in input. (e-h): we compare several network properties as measured in the observed and simulated networks. (e) The cost (TL) against the total path length (l), highlighting the different β for each obtained network. (f) Gini(T_e) against l. The optimal network is equivalent to the one with minimal Wasserstein, i.e. $\beta = 2.0$. (g) TL against the Gini coefficient of traffic on edges (Gini(T_e)). In this case, the optimal network corresponds to the one with $\beta = 1.5$. (h) TL against the density of branching nodes. The closest network in terms of the number of bifurcations for $\beta = 1.4$). (i) Wasserstein similarity measure for the simulated Grenoble tram networks as a function of β, in the setting of eight origins and one destination. The most similar network in terms of this measure is at $\beta = 2$, when $W(G_1, G_2)$ is minimum. The peak at $\beta = 1.6$ is due to the absence of a few edges in the rightmost part of the network that results in disconnecting a small branch, thus causing the distance to increase.

5.3.2 The New York subway system: a look into more complex structures

The New York subway is one of the largest and busiest transportation systems in the world. Due to its size and complexity, navigating through such network might be difficult for humans [184], but understanding its properties and structure could be indicative of improvement to city planners and urban designers.

In the scope of recovering such a complex structure, the strategy of selecting only a small number of origins and destinations, as done for the studied networks so far, might produce networks that are far from similar to the real one, especially given the high complexity of this particular system (see Figure 9.9). We thus use a different approach that could be a pointer towards recovering structures from major urban transportation systems. Each line that comprises the subway infrastructure of New York could be seen as an independent network itself. With this assumption, we selected four major lines (red, green, orange and yellow), extracting the nodes with lower degree as origins and one common destinationfor all of them, corresponding to the point with higher density of POI (see Figure 5.6).

We notice that in terms of cost (TL), our simulated networks have similar values in all cases, with equal performance for the red line, and a small difference for all the other

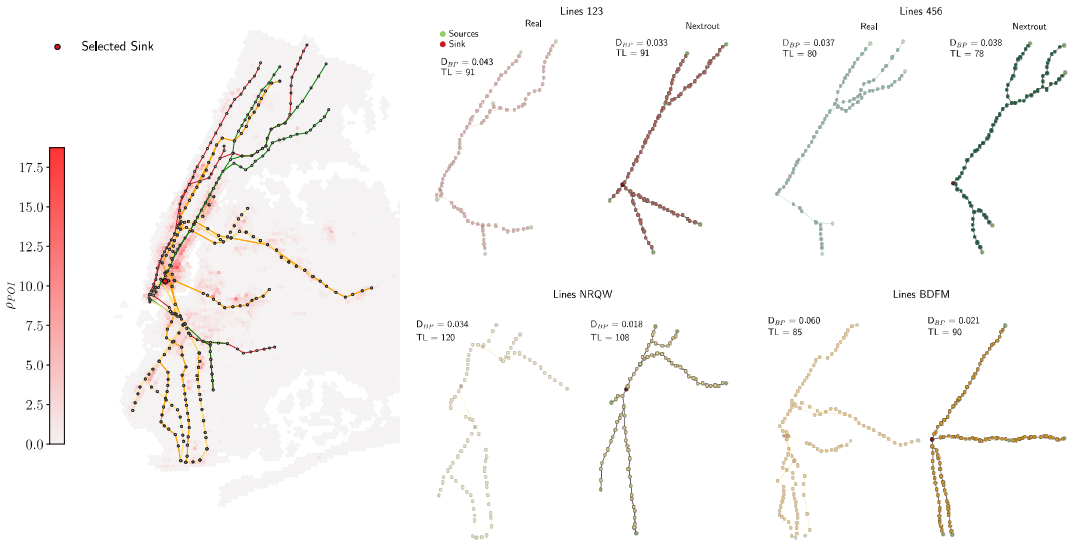


Fig. 5.6: Comparison for real and simulated networks for major lines of the subway system in New York, with distribution of POI. We select the four major lines (left panel), here shown along with the distribution of the density of POI (density varies as in the colorbar). We report the cost (TL) and density of branching points (D_{BP}) for both real and simulated networks. Here we set $\beta = 1.1$ for the orange line, $\beta = 1.5$ for both red and green lines, and $\beta = 1.6$ for the yellow line.

lines. For the density of branching points (D_{BP}) - besides an intuitive visual similarity - we observe comparable results. For instance, the green line (456) has similar branches both on the north and south sides, and similar considerations apply to the red line (123) and the north side of the yellow line (NRQW). One can also investigate how main differences are distributed in the orange line (BDFM), where there is not much similarity between observed and simulated infrastructures. This is because two destinations on the east side are traversed by a unique branch in our simulated network, while the real one splits them into two branches. Furthermore, the southern part of the network also shows a distinct behavior, where our simulated network has two branches, the real one splits into a more complex pattern that cannot be explained by our optimality principles. While it is not clear whether these differences are due to different underlying optimization rules or a lack of optimality in the observed network, our method enables practitioners to identify key insights on principled alternative designs where optimality is clearly defined in terms of network operating and infrastructural costs.

5.3.3 Initial network development: the French Railway in the 1850s

Our model builds a network backbone from scratch, with a global optimization that follows a principle of economy of scale. This could be particularly suited to study the initial development of a rail network infrastructure, as opposed to later stages where the network is gradually extended. We thus study a real scenario of the French railway system where we have access to historical information about network development in time [169], focusing on an initial phase around the year 1850. At that stage, the network topology contains multiple connected components but no loops - which only appear in a later stage, around several years later (see Figure 9.10).

We focus on the four biggest components with more interesting topologies, as the remaining components are either too small or simple straight lines, and select the node with the highest degree as a destination. In the biggest component, for instance, this corresponds to Paris. In Figure 5.7 we show examples of real and simulated networks highlighting the D_{BP} and the total length (L), the latter measured given the longitude and latitude coordinates mapped into a $[0, 1]$ system of coordinates. We note various degrees of similarity in the different components between observed and simulated network. For instance, the biggest component has similar D_{BP} but the observed network has a larger total length, mainly due to branches taking slightly longer detours to reach the sinks and the two small branches south-west of Paris being split from Paris onward into two in the observed network, while they are only later split in the simulated one. A similar behavior is observed in component 3 (north-est France), where we see the real network splitting earlier on than what simulated, thus causing a longer total length. The other two components have higher similarity in terms of both metrics, in particular, component 2 (south France) has a main branching point est of Nîmes similarly located in the observed and simulated network. The higher path length in this case is due to a longer detour of the southern branch. These types of detours could be caused by geographical obstacles that are not included in our more coarse-grained model.

Understanding the underlying optimization principles that drive how transportation networks changed and evolved over time might point towards creating better transportation systems. Our approach is a pointer towards comprehending the initial stages of such evolution, particularly suited for systems that follow the principle of economy of scale.

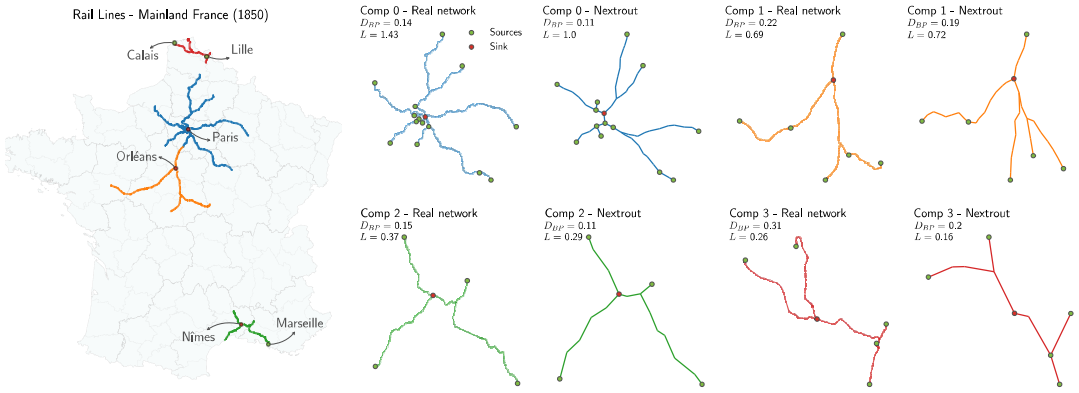


Fig. 5.7: Simulating the initial French Railways in the year 1850. Left: the original observed network, with connected components represented in different colors. Right: the networks simulated with our model for each component, given a set of origins and one destination. On top we report the density of branching points D_{BP} and the total Euclidean length L.

5.4 Improving network properties

Simulating networks that follow optimality principles and resemble well those observed in real datasets can be used to assess how urban transportation networks perform in terms of main transportation properties. This can guide network managers towards potential measures directed at improving certain properties. This possibility is conveniently enabled by our approach, as by continuously tuning the parameter β we can simulate various transportation scenarios, thus assessing how a network can increase or decrease a certain property.

In Figure 5.8 we show the main properties in all the networks simulated with our model and compare with those observed in real data, aiming to compare their trade-offs between

various performance metrics. In general, we notice how simulated networks cover a wider range of values than the real ones for the four transportation properties investigated in this work. This allows obtaining, for instance, networks that have shorter total path length ℓ with a comparable cost, as shown in Figure 5.8a where many simulated networks are located in the regime $0 < \text{TL} < 200$ with ℓ sharply dropping towards 0.1, while many real networks have $\ell > 0.1$. A similar behavior is observed also for the traffic against TL in Figure 5.8c, where simulated networks cover areas of the plot where traffic is less congested (smaller Gini(T_e)), in contrast to several real networks. Among the simulated networks, those selected according to the best Wasserstein measure tend to have lower cost and a smaller percentage of bifurcations D_{BL}, indicating that this measure encourages not only the usage of a lower amount of edges, but also nodes with low degree.

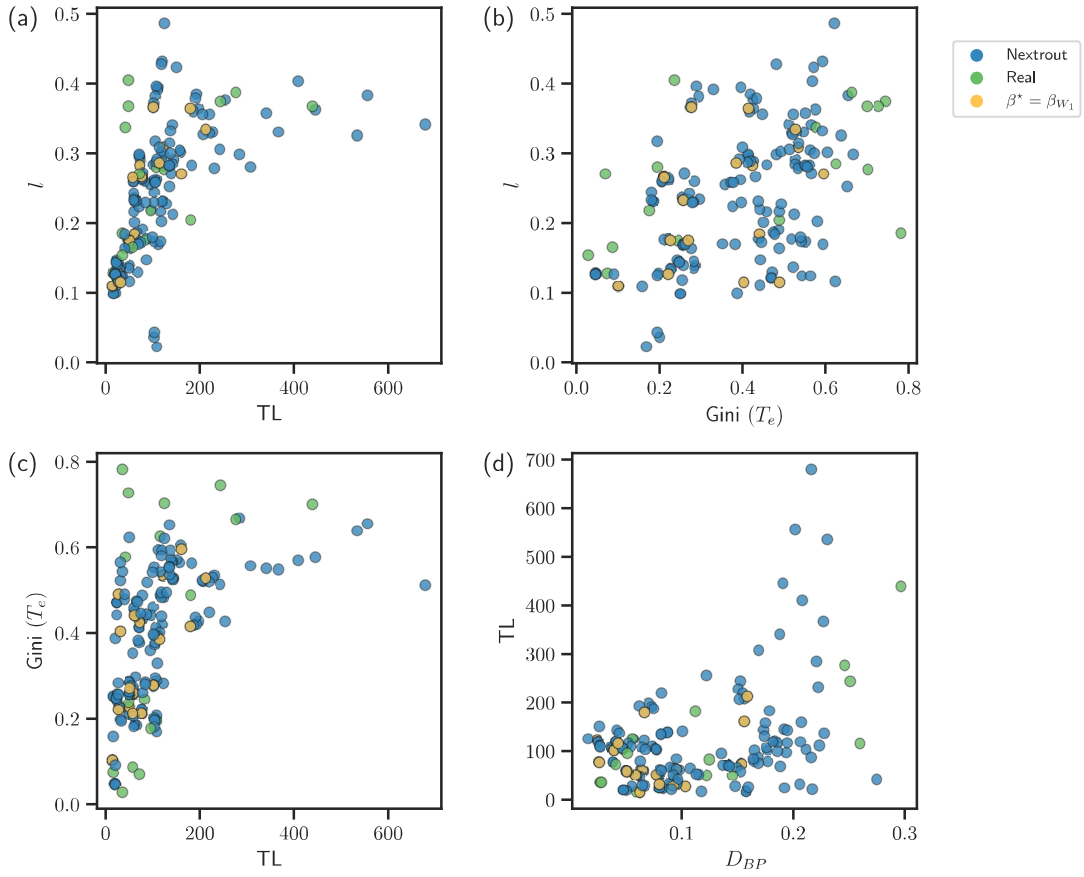


Fig. 5.8: Comparison of simulated and observed networks. We show the values of the main transportation properties investigated in this work for real and simulated networks. Simulated networks cover a wider range of properties' values, thus allowing in particular to select network that have lower or comparable values of these properties than those observed in the corresponding real networks.

5.5 Discussion

Planning transportation systems in a city is a challenging task. This study has shown that some urban transportation systems with a small number of loops can be simulated by simple principles based on optimal transport theory and economy of scale. Using empirical data from

one national railway and multiple rail transportation types across 18 cities, our model provides simulated networks obtained with little information in input and no *a priori* backbone network structure, exhibiting properties that resemble those observed in real transportation systems to various extents. Our model interpolates between various transportation regimes by tuning a single parameter, while allowing for a natural definition of a similarity measure to compare the simulated networks with those observed in real systems. We observed how the selected networks with this criterion can exhibit transportation properties that on average resemble the corresponding real networks well or point towards alternative infrastructures that improve relevant topological properties.

A limitation of this study is that our OT-based model does not capture infrastructures with loops, thus limiting its applicability to rail networks, or subway and rail networks with a very small density of loops. Possible extensions of the formalism presented in this work to account for loops are an interesting direction for future work. Besides simple heuristics and beyond the example we presented for the New York subway, one can make other interesting modeling choices to effectively tackle this problem. For instance, one could generalize our approach to situations where travel demands are treated stochastically [32, 185, 42] or change in time [131], scenarios where in certain regimes an OT-based approach can naturally lead to the formation of loops. Similarly, loops could emerge in multicommodity settings where fluxes of passengers are distinguished by their origin and destination stations, using an OT-based multicommodity framework as the one proposed in [129, 130, 186]. Both directions, provided they could be generalized to a continuous setting as the one studied here, can potentially result in optimal simulated network infrastructures capturing properties that differ from the ones analyzed here, e.g. robustness to disruptions.

Another limitation is the assumption that networks are static, i.e. do not change in time. It would be interesting to compare how differences between simulated and observed networks may arise because different network branches may have been developed in different time periods. This could be used to study the evolution of transportation properties in time, as done in [171, 187]. Similarly, the networks studied in this work are often one layer of a multimode network. Integrating other transportation modes into a multilayer formalism and suitably adapting our OT-based approach, e.g. borrowing ideas from [132], could give us a deeper understanding of optimal network design in interconnected urban systems.

Our model takes a few inputs (origins and destinations), but it does not consider any geographical obstacle. This can result in fine-grained mismatches between simulated and observed topologies due to longer detours in the real infrastructures to avoid these physical obstacles. In principle, this could be incorporated into our model by properly adding extra terms in the dynamical equations that drive flows and conductivities differently based on the location in space. However, this would require fine-grain details to be specified in input, an information that may not be easily available.

Here, we focus on designing a network from scratch. This is relevant in cases where infrastructures are relatively new or did not change considerably compared to their initial design, as in the cases we investigated here. However, this may be limited to studying infrastructures that have evolved over time. For these scenarios, it would be appropriate to consider how our model can be adapted to study network growth, where an initial backbone is further extended with new branches. This problem is related to the interplay between an urban transportation network and the distribution of its underlying population, as there could be a co-evolution between the two that should be taken into account [188].

In summary, there are many factors contributing to the development of urban transportation networks. Our simple optimization scheme provides a principled and computationally efficient

benchmark for comparison with real-world networks. By interpolating between different transportation regimes, we can vary the degree of similarity between the networks simulated by optimal transport principles and those observed in real systems. In particular, measuring relevant topological properties on simulated network resulting from different parameters' values against those observed in real networks can give us indications on how to improve transportation performance when taking into account principles of optimal transport and economy of scale.

5.6 Methods

Data collection and analysis

We collected network data from various public transportation networks from 18 different cities [168] and one national railway [169]. Network statistics are detailed in Table 5.1. Each city had one or multiple transportation modes available. Node ids were associated with the longitude and latitude coordinates of real stations for multiple means of transportation, as well as possible connections between them. Our main goal was to analyze each network individually, therefore we did not address the multilayer case where connections among the different means of transportation exist. For instance, if rail and subway stops have the same coordinates, they are treated as distinct in each network.

To avoid possible redundancies or inconsistencies in the data, such as duplicated nodes or edges that looked too long, we performed a preprocessing step. Specifically, we considered a threshold τ that corresponds to the minimum distance in kilometers between the pairs of nodes that had the same node ids and no connections between them, matching features stored in the node metadata such as names of the real stations. If the Euclidean distance $d(i, j)$ between nodes i and j was smaller than this threshold, we collapsed the two nodes into one, i.e. $i = j$. This was used to avoid those entries and exits of each station would be counted as two distinct nodes in the same network, and possibly affecting the selection of origins and destinations.

To match the latitude and longitude coordinates of the datasets with those in the 2-dimensional plane that nextrout uses to solve the continuous problem, we re-scaled every pair (lon, lat) to a $(0, 1)$ system of coordinates. Starting with a total of 64 data points (networks), we extracted the individual disconnected components and the number of loops for each of them. Network extraction was performed on the biggest components only.

We extract networks using Nextrout [173], selecting $1 < \beta \leq 2$ such that for every pair $(origins, destinations)$ we simulate 10 different networks. Since the extracted networks may contain redundancies, we remove them using the graph filtering step from Nextrout. Outputs of this step have less redundant structures and are closer to the optimal topologies.

Selecting origins and destinations with points of interest.

For the 16 studied networks, we used origins and destinations based on the degree and betweenness centrality measures. The degree centrality d_i of a given node is defined as the number of edges connected to it. The betweenness centrality is defined as the frequency with which a node is on the shortest path between all other nodes,

$$B_i = \sum_{i \neq j \neq k} \frac{\sigma_{ik}(j)}{\sigma_{ik}},$$

City	Trans. mode	N	E	# Comp.	L_{ratio}	O	$D_{\{d_i,B_i\}}$
Adelaide	rail	88	116	1	0.25	6	1
Berlin	rail	203	258	1	0.21	19	1
Berlin	subway	169	181	1	0.072	14	1
Bordeaux	tram	110	110	1	0.009	8	1
Brisbane	rail	297	367	1	0.193	10	1
Dublin	rail	59	73	1	0.03	10	1
Grenoble	tram	80	83	1	0.048	8	1
Helsinki	subway	17	16	1	0.0	4	1
Lisbon	rail	48	49	2	0.041	9	1
Luxembourg	rail	43	56	1	0.025	11	1
Melbourne	rail	219	290	1	0.248	22	1
Nantes	tram	97	96	1	0.0	10	1
New York	subway	423	506	1	0.17	22	1
Paris	rail	337	445	1	0.244	24	1
Prague	subway	24	23	3	0.0	6	1
Rome	subway	73	72	1	0.0	6	1
Toulouse	subway	37	36	1	0.0	4	1
Venice	tram	37	38	1	0.053	4	1

Tab. 5.1: Description of real networks considered. We report the main network statistics as the number of nodes N, the number of edges E, the number of components # Comp, the number of origins O, the number of destinations D, and the loops ratio L_{ratio} defined as the number of loops divided by the number of edges, and selecting networks with $L_{\text{ratio}} < 0.2$, as higher values would require the recovery of loops in the extracted networks.

where σ_{ik} is the total number of shortest paths from node i to node k and $\sigma_{ik}(j)$ those shortest paths passing through j.

Terminals were chosen as follows: nodes with $d_i \leq 1$ are assigned as origins, while those with $d_i = \max_n \cup d_n$ or $B_i = \max_n \cup B_n$ as destinations. We selected this set {origins,destinations} to be small, in order to use the least amount of information in input. In multiple cases the set of origins was equivalent in both centralities, with the difference being on the location of the destinations, thus the output networks were different. In terms of final networks properties, the results are comparable for both studied centralities (see SI for more details).

In order to account for other measures of attractiveness for the destinations and investigate the impact of including realistic information about urban areas, we explored a different approach for the cities of Rome and New York, by looking at a measure that accounts for both population distribution and land usage. To do that, we collected both population and distribution of POI from Open Street Map (OSM) data, and mapped them to an H3 tiling discretization of the space. We then defined the density of POI as $\rho_{\text{POI}} = P_{ij} W_{ij}$, where P_{ij} is the population density and W_{ij} is the number of POI for the corresponding ij cell.

Our hypothesis is that stations with higher centralities correspond to higher density cells. The destinationis then assigned based on the center of this H3 cell with highest ρ_{POI}. In Figure 5.2 (b) we show how this new criterion generates a configuration that leads to a different optimal network but still preserves similarity compared with the real network. We also notice that the highest centrality node is connected to regions with higher densities compared to the peripheral areas, where the origins are placed.

Conclusions and Future work

6

In this chapter, we provide an overview of the main research directions explored throughout this thesis. We summarize the main findings and challenges for each direction, discussing future work and applications that go beyond those already presented in the previous chapters. In Section 6.1 we summarize the importance of our network extraction method introduced in Chapter 3. In Section 6.2 we explore the implications of having a geometric-based community detection method, summarizing what we introduced in Chapter 4 and its implications. Finally, we highlighting the contributions of Chapter 5 in Section 6.3, discussing possible extensions of the model and how it can help practitioners to designing more efficient transportation systems.

6.1 Network topologies from routing optimization problems

In this thesis, we investigated optimal network topologies by combining recent advances from Optimal Transport and network theory. Specifically, in Chapter 3 we introduced a method to extract network topologies from routing optimization problems in continuous spaces. We described the three main steps of our method: extracting an optimal trajectory by solving a dynamical system, pre-extracting a network, and filtering out potential redundancies. We provided an alternative to manual graph extraction techniques, such as image processing, where the problems might not necessarily be related to routing optimization, therefore final output networks might contain information that is not relevant to the given application. It is important to highlight that the optimality of the generated networks from our method is given from the minimization of the well-defined Lyapunov functional, that can be interpreted as the costs of building and maintaining the network. This also implies that we expect this method to be valid to model network-like structures where this type of cost is appropriate. It may not be valid when we expect the underline shape to follow very different rules from the one captured by our formalism, e.g. random and noisy topologies that do not follow optimality.

Our method provides a flexible routine to extract optimal networks that can be adapted for different applications in a wider range of disciplines. For instance, in the climate sciences domain we can think of extending our method to extracting and understanding the dynamics of natural transportation networks like rivers or plant roots, as suggested in [26]. River networks have refined branching structures with significant biological implications for the stability of their populations [189, 190, 191], so understanding the optimality principles behind its evolution is fundamental for predicting environmental processes affecting them [192]. We can thus use information from river sources and deltas as inputs to our graph extraction pipeline and generate several optimal networks to compare with existing structures.

Besides only automating the graph extraction from routing optimization problems, the discrete dynamic used in the filtering step and introduced in Section 2.3 is independent of the network extraction step, allowing practitioners to use inputs that might not necessarily come from solutions of routing optimization problems. For example, filtering redundancies in Internet Protocol (IP) networks is crucial to avoid congestion and inefficiency [193, 194], as loops might cause the data packets to circulate indefinitely. Another direct application to our filtering

step is the use of the Wasserstein distance for graph comparison, an application explored in Chapter 4 and Chapter 5.

We remark that the current implementation the filtering step only accounts for a single type of flow in the network. Possible extensions of it would also account for multi-commodities or time-dependent forcing, such as the works in [128, 130, 131]. Last, we also expect that our method will create new network databases related to routing optimization problems, which could be a resource for testing and validating new routing algorithms.

6.2 Community Detection from a Geometrical Perspective

Many real-world networks often exhibit intrinsic community structures. In general, they are defined as a group of nodes that are more densely connected compared to the rest of the network [20, 195, 21, 91]. Despite the existence of multiple algorithms to find such groups, community detection still remains a critical and complex area of study, lacking an overall "best" algorithm due to the large diversity of data types [196, 110], as networks can vary significantly in size, density, and nature of the connections between nodes, thus influencing how communities form and evolve.

Among the several directions to consider when detecting communities in networks, an interesting one is that of considering the network as a geometrical object, and use the geometric concept of curvature to measure the shared information across and within communities. In Chapter 4, inspired by the filtering step of our network extraction pipeline proposed in Chapter 3, we presented an algorithm that allows us to understand how combining OT principles and geometrical methods, as the Ollivier-Ricci curvature (ORC), can effectively detect communities in networks by controlling the flow of information shared across nodes' neighborhoods. A significant advantage of our method (and of OT-based approaches in general) is that it automatically extracts the number of communities from the data, without requiring this to be a preset input parameter, unlike statistical or combinatorial methods.

We looked into different transportation regimes in order to efficiently measure the ORC on edges in Sections 4.2 to 4.4. In comparison with existing OT-based methods, our model has shown more robustness in the regimes where inference is not trivial, for both synthetic and semi-synthetic data. We performed an extensive analysis to show better or comparable performance in recovering community structures than other geometric-based approaches.

A limitation of our model is that it tends to overestimate the number of communities, as OT-based methods have shown better performance for networks with more densely connected community structures [196, 101]. Future extensions of it should then focus on the case of more sparsely connected community structures, which are characteristic of many real-world networks [197, 198]. An additional extension to be explored is that of understanding how the parameter β captures the various network topologies, for instance, by measuring structural distances between networks and correlate these with the best-performing β values. Other research directions include accounting for the multilayer case, where communities reveal the relationships of nodes within and across layers, or incorporating a node attributes for a more informed community detection, inspired in the works of [199, 20].

6.3 Designing Optimal Transport Urban Infrastructures

Designing optimal transportation networks in urban environments is essential for not only moving people and resources within a city efficiently, but also for saving on new construction costs and reducing their environmental impacts. As cities tend to change over time, an optimal transportation network should be also adaptable to compress these changes, like the different demands. Despite the existence of several methods for studying the quality of network design, they often depend on measuring topological properties only once the network has already been built. One can instead look at the opposite case, where the network has not yet been constructed. The problem then shifts to that of finding a principled optimization setup with a defined cost function that the network should minimize under a set of constraints, and then finding optimal solutions in terms of network topologies.

In Chapter 5 we introduced a method that does not assume an initial network structure, but instead extracts this from a continuous space, taking advantage of the network extraction setup presented in Chapter 3. We also explore the principle of economy of scale, that establishes how the costs of producing and that of maintaining and operating the network should balance, showing how our method could lead to more cost-effective designed urban transportation networks. For each of the studied cities, we start with only a set of points that represent regions to be connected, like the outskirts and city center, which are regions that we assume people will need to daily commute between where they live and work. We generate optimal network topologies and propose a method to compare their similarity with existing structures, also measuring several network properties like average path length or traffic distribution on the network.

By comparing different optimal networks, our method could guide urban planners and policymakers by providing an evaluation for alternative infrastructures, and select the most effective strategies for improving transportation systems. Our method is particularly suitable for networks with small density of loops, so an extension of the model would include such cases, which are often observed in more complex transportation infrastructures. Another research direction would be generalizing our model to the multilayer case, where multiple modes of transportation are integrated. A challenging scenario for future research includes adding real traffic data to understand how our proposed networks would re-distribute passengers in case of disruptions, and compare it with existing strategies. This could be useful to understand and improve the resilience and vulnerability of transportation infrastructures.

Bibliography

[1] Gaspard Monge. “Mémoire sur la théorie des déblais et des remblais”. In: *Mem. Math. Phys. Acad. Royale Sci.* (1781), pp. 666–704.

[2] Leonid V Kantorovich. “On the translocation of masses”. In: *Dokl. Akad. Nauk. USSR (NS)*. Vol. 37. 1942, pp. 199–201.

[3] Eduardo Fernandes Montesuma, Fred Ngole Mboula, and Antoine Souloumiac. “Recent advances in optimal transport for machine learning”. In: *arXiv preprint arXiv:2306.16156* (2023).

[4] Luis Caicedo Torres, Luiz Manella Pereira, and M Hadi Amini. “A survey on optimal transport for machine learning: Theory and applications”. In: *arXiv preprint arXiv:2106.01963* (2021).

[5] Filippo Santambrogio. “Models and applications of optimal transport in economics, traffic and urban planning”. In: *arXiv preprint arXiv:1009.3857* (2010).

[6] Geert-Jan Huizing, Gabriel Peyré, and Laura Cantini. “Optimal transport improves cell–cell similarity inference in single-cell omics data”. In: *Bioinformatics* 38.8 (2022), pp. 2169–2177.

[7] Antoine Diez and Jean Feydy. “An optimal transport model for dynamical shapes, collective motion and cellular aggregates”. In: *arXiv preprint arXiv:2402.17086* (2024).

[8] Sarah Tepler Drobnitch, Kaare H Jensen, Paige Prentice, and Jarmila Pittermann. “Convergent evolution of vascular optimization in kelp (Laminariales)”. In: *Proceedings of the Royal Society B: Biological Sciences* 282.1816 (2015), p. 20151667.

[9] Andrea Perna and Tanya Latty. “Animal transportation networks”. In: *Journal of The Royal Society Interface* 11.100 (2014), p. 20140334.

[10] Oliver Stein, Jan Oldenburg, and Wolfgang Marquardt. “Continuous reformulations of discrete–continuous optimization problems”. In: *Computers & chemical engineering* 28.10 (2004), pp. 1951–1966.

[11] Valentin Hartmann and Dominic Schuhmacher. “Semi-discrete optimal transport: a solution procedure for the unsquared Euclidean distance case”. In: *Mathematical Methods of Operations Research* 92.1 (2020), pp. 133–163.

[12] Kent Quanrud. “Approximating optimal transport with linear programs”. In: *arXiv preprint arXiv:1810.05957* (2018).

[13] Nazarii Tupitsa, Pavel Dvurechensky, Darina Dvinskikh, and Alexander Gasnikov. “Numerical methods for large-scale optimal transport”. In: *arXiv preprint arXiv:2210.11368* (2022).

[14] Atsushi Tero, Ryo Kobayashi, and Toshiyuki Nakagaki. “A mathematical model for adaptive transport network in path finding by true slime mold”. In: *Journal of theoretical biology* 244.4 (2007), pp. 553–564.

[15] Toshiyuki Nakagaki, Hiroyasu Yamada, and Ágota Tóth. “Maze-solving by an amoeboid organism”. In: *Nature* 407.6803 (2000), pp. 470–470.

[16] Audrey Dussutour, Tanya Latty, Madeleine Beekman, and Stephen J Simpson. “Amoeboid organism solves complex nutritional challenges”. In: *Proceedings of the National Academy of Sciences* 107.10 (2010), pp. 4607–4611.

[17] Liping Zhu, Song-Ju Kim, Masahiko Hara, and Masashi Aono. “Remarkable problem-solving ability of unicellular amoeboid organism and its mechanism”. In: *Royal Society Open Science* 5.12 (2018), p. 180396.

[18] Enrico Facca, Franco Cardin, and Mario Putti. “Towards a Stationary Monge–Kantorovich Dynamics: The Physarum Polycephalum Experience”. In: *SIAM Journal on Applied Mathematics* 78.2 (2018), pp. 651–676.

[19] Enrico Facca, Sara Daneri, Franco Cardin, and Mario Putti. “Numerical Solution of Monge–Kantorovich Equations via a Dynamic Formulation”. In: *J Sci Comput* 82.68 (Feb. 2020), pp. 1–26.

[20] Martina Contisciani, Eleanor A Power, and Caterina De Bacco. “Community detection with node attributes in multilayer networks”. In: *Scientific reports* 10.1 (2020), pp. 1–16.

[21] Xinyu Huang, Dongming Chen, Tao Ren, and Dongqi Wang. “A survey of community detection methods in multilayer networks”. In: *Data Mining and Knowledge Discovery* 35.1 (2021), pp. 1–45.

[22] Andrea Lancichinetti and Santo Fortunato. “Community detection algorithms: a comparative analysis”. In: *Physical review E* 80.5 (2009), p. 056117.

[23] Daniela Leite and Caterina De Bacco. “Similarity and economy of scale in urban transportation networks and optimal transport-based infrastructures”. In: *Nature Communications* 15.1 (2024), p. 7981.

[24] Thilo Gross and Hiroki Sayama. *Adaptive networks*. Springer, 2009.

[25] Vincenzo Bonifaci, Kurt Mehlhorn, and Girish Varma. “Physarum can compute shortest paths”. In: *Journal of theoretical biology* 309 (2012), pp. 121–133.

[26] Enrico Facca et al. “Biologically inspired formulation of Optimal Transport Problems”. In: (2018).

[27] Enrico Facca and Franco Cardin. “Branching structures emerging from a continuous optimal transport model”. In: *J Comput Phys* Submitted (2020).

[28] Enrico Facca, Sara Daneri, Franco Cardin, and Mario Putti. “Numerical solution of Monge–Kantorovich equations via a dynamic formulation”. In: *Journal of Scientific Computing* 82.3 (2020), pp. 1–26.

[29] Enrico Facca, Franco Cardin, and Mario Putti. “Physarum Dynamics and Optimal Transport for Basis Pursuit”. In: *arXiv preprint arXiv:1812.11782* (2018).

[30] Diego Baptista and Caterina De Bacco. “Principled network extraction from images”. In: *Royal Society Open Science* 8.7 (2021), p. 210025.

[31] Jayanth R Banavar, Francesca Colaiori, Alessandro Flammini, Amos Maritan, and Andrea Rinaldo. “Topology of the fittest transportation network”. In: *Physical Review Letters* 84.20 (2000), p. 4745.

[32] Francis Corson. “Fluctuations and redundancy in optimal transport networks”. In: *Physical Review Letters* 104.4 (2010), p. 048703.

[33] G Li, SDS Reis, AA Moreira, et al. “Towards design principles for optimal transport networks”. In: *Physical review letters* 104.1 (2010), p. 018701.

[34] Chi Ho Yeung, David Saad, and KY Michael Wong. “From the physics of interacting polymers to optimizing routes on the London Underground”. In: *Proceedings of the National Academy of Sciences* 110.34 (2013), pp. 13717–13722.

[35] Roger Guimerà, Albert Díaz-Guilera, Fernando Vega-Redondo, Antonio Cabrales, and Alex Arenas. “Optimal network topologies for local search with congestion”. In: *Physical review letters* 89.24 (2002), p. 248701.

[36] Luca Donetti, Pablo I Hurtado, and Miguel A Munoz. “Entangled networks, synchronization, and optimal network topology”. In: *Physical Review Letters* 95.18 (2005), p. 188701.

[37] Henrik Ronellenfitsch, Jörn Dunkel, and Michael Wilczek. “Optimal noise-canceling networks”. In: *Physical review letters* 121.20 (2018), p. 208301.

[38] Yuval Gazit, David A Berk, Michael Leunig, Laurence T Baxter, and Rakesh K Jain. “Scale-invariant behavior and vascular network formation in normal and tumor tissue”. In: *Physical review letters* 75.12 (1995), p. 2428.

[39] Diego Garlaschelli, Guido Caldarelli, and Luciano Pietronero. “Universal scaling relations in food webs”. In: *Nature* 423.6936 (2003), p. 165.

[40] Paul Balister, József Balogh, Enrico Bertuzzo, et al. “River landscapes and optimal channel networks”. In: *Proceedings of the National Academy of Sciences* 115.26 (2018), pp. 6548–6553.

[41] Filippo Santambrogio. “Optimal channel networks, landscape function and branched transport”. In: *Interfaces and Free Boundaries* 9.1 (2007), pp. 149–169.

[42] Eleni Katifori, Gergely J Szöllősi, and Marcelo O Magnasco. “Damage and fluctuations induce loops in optimal transport networks”. In: *Physical Review Letters* 104.4 (2010), p. 048704.

[43] Filippo Santambrogio. “Optimal transport for applied mathematicians”. In: *Birkäuser, NY* 55.58-63 (2015), p. 94.

[44] Susanna M Messinger, Keith A Mott, and David Peak. “Task-performing dynamics in irregular, biomimetic networks”. In: *Complexity* 12.6 (2007), pp. 14–21.

[45] Marcus Fruttiger. “Development of the mouse retinal vasculature: angiogenesis versus vasculogenesis”. In: *Investigative ophthalmology & visual science* 43.2 (2002), pp. 522–527.

[46] Chris B Schaffer, Beth Friedman, Nozomi Nishimura, et al. “Two-photon imaging of cortical surface microvessels reveals a robust redistribution in blood flow after vascular occlusion”. In: *PLoS biology* 4.2 (2006), e22.

[47] Fabrizio Altarelli, Alfredo Braunstein, Luca Dall’Asta, Caterina De Bacco, and Silvio Franz. “The edge-disjoint path problem on random graphs by message-passing”. In: *PloS one* 10.12 (2015), e0145222.

[48] M Bayati, C Borgs, Alfredo Braunstein, et al. “Statistical mechanics of steiner trees”. In: *Physical review letters* 101.3 (2008), p. 037208.

[49] Caterina De Bacco, Silvio Franz, David Saad, and C H Yeung. “Shortest node-disjoint paths on random graphs”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2014.7 (2014), P07009.

[50] Alfredo Braunstein and Anna Paola Muntoni. “The cavity approach for Steiner trees packing problems”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2018.12 (2018), p. 123401.

[51] Henrik Ronellenfitsch and Eleni Katifori. “Global Optimization, Local Adaptation, and the Role of Growth in Distribution Networks”. In: *Phys Rev Lett* 117.13 (Sept. 2016), H364–5.

[52] Alessio Brancolini and Sergio Solimini. “Fractal regularity results on optimal irrigation patterns”. In: *Journal de Mathématiques Pures et Appliquées* 102.5 (Nov. 2014), pp. 854–890.

[53] Qinglan Xia. “Motivations, ideas and applications of ramified optimal transportation”. In: *ESAIM Math. Model. Numer. Anal.* 49.6 (2015), pp. 1791–1832.

[54] Paul Pegon, Filippo Santambrogio, and Qinglan Xia. “A fractal shape optimization problem in branched transport”. In: *Journal de Mathématiques Pures et Appliquées* 123 (Mar. 2019), pp. 244–269.

[55] Werner Baumgarten and MJ Hauser. “Detection, extraction, and analysis of the vein network”. In: *Journal of Computational Interdisciplinary Sciences* 1.3 (2010), pp. 241–249.

[56] Boguslaw Obara, Vicente Grau, and Mark D Fricker. “A bioimage informatics approach to automatically extract complex fungal networks”. In: *Bioinformatics* 28.18 (2012), pp. 2374–2381.

[57] Daniel P Bebber, Juliet Hynes, Peter R Darrah, Lynne Boddy, and Mark D Fricker. “Biological solutions to transport network design”. In: *Proceedings of the Royal Society B: Biological Sciences* 274.1623 (2007), pp. 2307–2315.

[58] Lynne Boddy, Jonathan Wood, Emily Redman, Juliet Hynes, and Mark D Fricker. “Fungal network responses to grazing”. In: *Fungal Genetics and Biology* 47.6 (2010), pp. 522–530.

[59] Maryam Taghizadeh Dehkordi, Saeed Sadri, and Alimohamad Doosthoseini. “A review of coronary vessel segmentation algorithms”. In: *Journal of medical signals and sensors* 1.1 (2011), p. 49.

[60] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. " O’Reilly Media, Inc.", 2008.

[61] Michael Dirnberger, Tim Kehl, and Adrian Neumann. “NEFI: Network extraction from images”. In: *Scientific reports* 5 (2015), p. 15669.

[62] Dengfeng Chai, Wolfgang Forstner, and Florent Lafarge. “Recovering line-networks in images by junction-point processes”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 1894–1901.

[63] Jun Wang, Jingwei Song, Mingquan Chen, and Zhi Yang. “Road network extraction: A neural-dynamic framework based on deep learning and a finite state machine”. In: *International Journal of Remote Sensing* 36.12 (2015), pp. 3144–3169.

[64] Jan Dirk Wegner, Javier Alexander Montoya-Zegarra, and Konrad Schindler. “Road networks as collections of minimum cost paths”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 108 (2015), pp. 128–137.

[65] Aric Hagberg, Pieter Swart, and Daniel S Chult. *Exploring network structure, dynamics, and function using NetworkX*. Tech. rep. Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[66] Kaikuo Xu, Changjie Tang, Rong Tang, Ghulam Ali, and Jun Zhu. “A comparative study of six software packages for complex network research”. In: *2010 Second International Conference on Communication Software and Networks*. IEEE. 2010, pp. 350–354.

[67] Vladimir Batagelj and Andrej Mrvar. “Pajek-program for large network analysis”. In: *Connections* 21.2 (1998), pp. 47–57.

[68] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. “Gephi: an open source software for exploring and manipulating networks”. In: *Third international AAAI conference on weblogs and social media*. 2009.

[69] Lawrence C Evans and Wilfrid Gangbo. *Differential equations methods for the Monge-Kantorovich mass transfer problem*. 653. American Mathematical Soc., 1999.

[70] Qinglan Xia. “On landscape functions associated with transport paths”. In: *Discrete Contin. Dyn. Syst* 34.4 (2014), pp. 1683–1700.

[71] *With hydraulic interpretation, we can think of the edges as pipes, small cylindrical channels where the mass is passing through, and the capacity is proportional to the size of the pipe diameter.*

[72] *Mesh-related parameters specify how the mesh for the discretization of space is built. Specifically, we could specify ndiv, the number of divisions in the x axis and nref, the number of refinements, i.e. the number of times each triangle on the grid generated by a specific ndiv is subdivided into 4 triangles.*

[73] *In this work we focus on a 2D space, but the procedure can be generalized to 3D.*

[74] Carlos AS Oliveira and Panos M Pardalos. *Mathematical aspects of network routing optimization*. Springer, 2011.

[75] Damian Straszak and Nisheeth K Vishnoi. “IRLS and slime mold: Equivalence and convergence”. In: *arXiv preprint arXiv:1601.02712* (2016).

[76] *Due to the high graph connectivity, degree centrality is not appropriate for selecting these ending parts.*

[77] M Fricker, L Boddy, and D Bebber. “Network organisation of mycelial fungi”. In: *Biology of the fungal cell*. Springer, 2007, pp. 309–330.

[78] Michael Dirnberger and Kurt Mehlhorn. “Characterizing networks formed by P. polycephalum”. In: *Journal of Physics D: Applied Physics* 50.22 (2017), p. 224002.

[79] Michael Dirnberger, Kurt Mehlhorn, and Tim Mehlhorn. “Introducing the slime mold graph repository”. In: *Journal of Physics D: Applied Physics* 50.26 (2017), p. 264001.

[80] Mark Newman. *Networks*. Oxford university press, 2018.

[81] Santo Fortunato. “Community detection in graphs”. In: *Physics reports* 486.3-5 (2010), pp. 75–174.

[82] Santo Fortunato and Darko Hric. “Community detection in networks: A user guide”. In: *Physics reports* 659 (2016), pp. 1–44.

[83] Melanie Weber, Jürgen Jost, and Emil Saucan. “Detecting the coarse geometry of networks”. In: *In NeurIPS 2018 Workshop* (2018).

[84] Areejit Samal, RP Sreejith, Jiao Gu, et al. “Comparative analysis of two discretizations of Ricci curvature for complex networks”. In: *Scientific reports* 8.1 (2018), pp. 1–16.

[85] Jiang Zhu, Bai Wang, Bin Wu, and Weiyu Zhang. “Emotional community detection in social network”. In: *IEICE Transactions on Information and Systems* 100.10 (2017), pp. 2515–2525.

[86] Dong Wang, Jiexun Li, Kaiquan Xu, and Yizhen Wu. “Sentiment community detection: exploring sentiments and relationships in social networks”. In: *Electronic Commerce Research* 17.1 (2017), pp. 103–132.

[87] Chaoyi Li and Yangsen Zhang. “A personalized recommendation algorithm based on large-scale real micro-blog data”. In: *Neural Computing and Applications* 32.15 (2020), pp. 11245–11252.

[88] Todd Waskiewicz. “Friend of a friend influence in terrorist social networks”. In: *Proceedings on the international conference on artificial intelligence (ICAI)*. The Steering Committee of The World Congress in Computer Science, Computer . . . 2012, p. 1.

[89] Carlos Andrë Reis Pinheiro. “Community detection to identify fraud events in telecommunications networks”. In: *SAS SUGI proceedings: customer intelligence* (2012).

[90] Jiancong Chen, Huiling Zhang, Zhi-Hong Guan, and Tao Li. “Epidemic spreading on networks with overlapping community structure”. In: *Physica A: Statistical Mechanics and its Applications* 391.4 (2012), pp. 1848–1854.

[91] Caterina De Bacco, Eleanor A Power, Daniel B Larremore, and Cristopher Moore. “Community detection, link prediction, and layer interdependence in multilayer networks”. In: *Physical Review E* 95.4 (2017), p. 042317.

[92] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. “Near linear time algorithm to detect community structures in large-scale networks”. In: *Physical review E* 76.3 (2007), p. 036106.

[93] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. “Finding community structure in very large networks”. In: *Physical review E* 70.6 (2004), p. 066111.

[94] Jörg Reichardt and Stefan Bornholdt. “Statistical mechanics of community detection”. In: *Physical review E* 74.1 (2006), p. 016110.

[95] Martin Rosvall and Carl T Bergstrom. “Maps of random walks on complex networks reveal community structure”. In: *Proceedings of the national academy of sciences* 105.4 (2008), pp. 1118–1123.

[96] Ruochen Yang, Frederic Sala, and Paul Bogdan. “Hidden network generating rules from partially observed complex networks”. In: *Communications Physics* 4.1 (2021), pp. 1–12.

[97] Jürgen Jost and Shiping Liu. “Ollivier’s Ricci curvature, local clustering and curvature-dimension inequalities on graphs”. In: *Discrete & Computational Geometry* 51.2 (2014), pp. 300–322.

[98] Melanie Weber, Jürgen Jost, and Emil Saucan. “Forman-Ricci flow for change detection in large dynamic data sets”. In: *Axioms* 5.4 (2016), p. 26.

[99] Melanie Weber, Emil Saucan, and Jürgen Jost. “Characterizing complex networks with Forman-Ricci curvature and associated geometric flows”. In: *Journal of Complex Networks* 5.4 (2017), pp. 527–550.

[100] Chien-Chun Ni, Yu-Yao Lin, Feng Luo, and Jie Gao. “Community detection on networks with ricci flow”. In: *Scientific reports* 9.1 (2019), pp. 1–12.

[101] Jayson Sia, Edmond Jonckheere, and Paul Bogdan. “Ollivier-ricci curvature-based method to community detection in complex networks”. In: *Scientific reports* 9.1 (2019), pp. 1–12.

[102] Enrico Facca, Franco Cardin, and Mario Putti. “Branching structures emerging from a continuous optimal transport model”. In: *Journal of Computational Physics* 447 (2021), p. 110700.

[103] Lorenzo Brasco, Guillaume Carlier, and Filippo Santambrogio. “Congested traffic dynamics, weak flows and very degenerate elliptic equations”. In: *Journal de mathématiques pures et appliquées* 93.6 (2010), pp. 652–671.

[104] Diego Baptista, Daniela Leite, Enrico Facca, Mario Putti, and Caterina De Bacco. “Network extraction by routing optimization”. In: *Scientific reports* 10.1 (2020), pp. 1–13.

[105] Enrico Facca, Andreas Karrenbauer, Pavel Kolev, and Kurt Mehlhorn. “Convergence of the non-uniform directed Physarum model”. In: *Theoretical Computer Science* 816 (2020), pp. 184–194.

[106] Romeil Sandhu, Tryphon Georgiou, Ed Reznik, et al. “Graph curvature for differentiating cancer networks”. In: *Scientific reports* 5.1 (2015), pp. 1–13.

[107] Chi Wang, Edmond Jonckheere, and Reza Banirazi. “Interference constrained network control based on curvature”. In: *2016 American Control Conference (ACC)*. IEEE. 2016, pp. 6036–6041.

[108] Chien-Chun Ni, Yu-Yao Lin, Jie Gao, Xianfeng David Gu, and Emil Saucan. “Ricci curvature of the internet topology”. In: *2015 IEEE conference on computer communications (INFOCOM)*. IEEE. 2015, pp. 2758–2766.

[109] Karel Devriendt and Renaud Lambiotte. “Discrete curvature on graphs from the effective resistance”. In: *Journal of Physics: Complexity* (2022).

[110] Adam Gosztolai and Alexis Arnaudon. “Unfolding the multiscale structure of networks with dynamical Ollivier-Ricci curvature”. In: *Nature Communications* 12.1 (2021), pp. 1–11.

[111] Daniel Paulin. “Mixing and concentration by Ricci curvature”. In: *Journal of Functional Analysis* 270.5 (2016), pp. 1623–1662.

[112] Laurent Veysseire. “Coarse Ricci curvature for continuous-time Markov processes”. In: *arXiv preprint arXiv:1202.0420* (2012).

[113] Yann Ollivier. “Ricci curvature of Markov chains on metric spaces”. In: *Journal of Functional Analysis* 256.3 (2009), pp. 810–864.

[114] Yann Ollivier. “A survey of Ricci curvature for metric spaces and Markov chains”. In: *Probabilistic approach to geometry*. Mathematical Society of Japan, 2010, pp. 343–381.

[115] Robin Forman. “Bochner’s method for cell complexes and combinatorial Ricci curvature”. In: *Discrete and Computational Geometry* 29.3 (2003), pp. 323–374.

[116] Chien-Chun Ni, Yu-Yao Lin, Jie Gao, and Xianfeng Gu. “Network alignment by discrete Ollivier-Ricci flow”. In: *International Symposium on Graph Drawing and Network Visualization*. Springer. 2018, pp. 447–462.

[117] Ze Ye, Kin Sum Liu, Tengfei Ma, Jie Gao, and Chao Chen. “Curvature graph network”. In: *International Conference on Learning Representations*. 2019.

[118] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. “Inference and phase transitions in the detection of modules in sparse networks”. In: *Physical Review Letters* 107.6 (2011), p. 065701.

[119] Brian Ball, Brian Karrer, and Mark EJ Newman. “Efficient and principled method for detecting communities in networks”. In: *Physical Review E* 84.3 (2011), p. 036103.

[120] Tiago P Peixoto. “Bayesian stochastic blockmodeling”. In: *Advances in network clustering and blockmodeling* (2019), pp. 289–332.

[121] Jaewon Yang and Jure Leskovec. “Overlapping community detection at scale: a nonnegative matrix factorization approach”. In: *Proceedings of the sixth ACM international conference on Web search and data mining*. 2013, pp. 587–596.

[122] Mark EJ Newman. “Spectral methods for community detection and graph partitioning”. In: *Physical Review E* 88.4 (2013), p. 042822.

[123] Daniel L Sussman, Minh Tang, Donniell E Fishkind, and Carey E Priebe. “A consistent adjacency spectral embedding for stochastic blockmodel graphs”. In: *Journal of the American Statistical Association* 107.499 (2012), pp. 1119–1128.

[124] Michelle Girvan and Mark EJ Newman. “Community structure in social and biological networks”. In: *Proceedings of the national academy of sciences* 99.12 (2002), pp. 7821–7826.

[125] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. “Fast unfolding of communities in large networks”. In: *Journal of statistical mechanics: theory and experiment* 2008.10 (2008), P10008.

[126] Enrico Facca, Franco Cardin, and Mario Putti. “Towards a Stationary Monge–Kantorovich Dynamics: The Physarum Polycephalum Experience”. In: *SIAM Journal on Applied Mathematics* 78.2 (2018), pp. 651–676.

[127] Qinglan Xia. “Optimal paths related to transport problems”. In: *commcont* 5.02 (2003), pp. 251–279.

[128] Diego Baptista and Caterina De Bacco. “Convergence properties of optimal transport-based temporal networks”. In: *International Conference on Complex Networks and Their Applications*. Springer. 2021, pp. 578–592.

[129] Alessandro Lonardi, Enrico Facca, Mario Putti, and Caterina De Bacco. “Designing optimal networks for multicommodity transport problem”. In: *Physical Review Research* 3.4 (2021), p. 043010.

[130] Alessandro Lonardi, Mario Putti, and Caterina De Bacco. “Multicommodity routing optimization for engineering networks”. In: *Scientific Reports* 12.1 (2022), pp. 1–11.

[131] Alessandro Lonardi, Enrico Facca, Mario Putti, and Caterina De Bacco. “Infrastructure adaptation and emergence of loops in network routing with time-dependent loads”. In: *arXiv preprint arXiv:2112.10620* (2021).

[132] Abdullahi Adinoyi Ibrahim, Alessandro Lonardi, and Caterina De Bacco. “Optimal transport in multilayer networks for traffic flow optimization”. In: *Algorithms* 14.7 (2021), p. 189.

[133] Michael L Fredman and Robert Endre Tarjan. “Fibonacci heaps and their uses in improved network optimization algorithms”. In: *Journal of the ACM (JACM)* 34.3 (1987), pp. 596–615.

[134] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. “Benchmark graphs for testing community detection algorithms”. In: *Physical review E* 78.4 (2008), p. 046110.

[135] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. “Stochastic blockmodels: First steps”. In: *Social networks* 5.2 (1983), pp. 109–137.

[136] Lawrence Hubert and Phipps Arabie. “Comparing partitions”. In: *Journal of classification* 2.1 (1985), pp. 193–218.

[137] Enrico Facca and Michele Benzi. “Fast Iterative Solution of the Optimal Transport Problem on Graphs”. In: *SIAM Journal on Scientific Computing* 43.3 (2021), A2295–A2319.

[138] Marco Cuturi. “Sinkhorn distances: Lightspeed computation of optimal transport”. In: *Advances in neural information processing systems* 26 (2013).

[139] Rémi Flamary, Nicolas Courty, Alexandre Gramfort, et al. “Pot: Python optimal transport”. In: *Journal of Machine Learning Research* 22.78 (2021), pp. 1–8.

[141] Leto Peel, Daniel B Larremore, and Aaron Clauset. “The ground truth about metadata and community detection in networks”. In: *Science advances* 3.5 (2017), e1602548.

[142] Donald Ervin Knuth. *The Stanford GraphBase: a platform for combinatorial computing*. Vol. 1. AcM Press New York, 1993.

[143] David Lusseau, Karsten Schneider, Oliver J Boisseau, et al. “The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations”. In: *Behavioral Ecology and Sociobiology* 54.4 (2003), pp. 396–405.

[144] David Lusseau and Mark EJ Newman. “Identifying the role that animals play in their social networks”. In: *Proceedings of the Royal Society of London. Series B: Biological Sciences* 271.suppl_-6 (2004), S477–S481.

[146] Thomas MJ Fruchterman and Edward M Reingold. “Graph drawing by force-directed placement”. In: *Software: Practice and experience* 21.11 (1991), pp. 1129–1164.

[147] Xiongye Xiao, Hanlong Chen, and Paul Bogdan. “Deciphering the generating rules and functionalities of complex networks”. In: *Scientific reports* 11.1 (2021), pp. 1–15.

[148] Yuankun Xue and Paul Bogdan. “Reliable multi-fractal characterization of weighted complex networks: algorithms and implications”. In: *Scientific reports* 7.1 (2017), pp. 1–22.

[149] Abdullahi Adinoyi Ibrahim, Daniela Leite, and Caterina De Bacco. “Sustainable optimal transport in multilayer networks”. In: *Physical Review E* 105.6 (2022), p. 064302.

[150] Jianhua Zhang, Mingwei Zhao, Haikuan Liu, and Xiaoming Xu. “Networked characteristics of the urban rail transit networks”. In: *Physica A: Statistical Mechanics and its Applications* 392.6 (2013), pp. 1538–1546.

[151] Rui Ding, Norsidah Ujang, Hussain Bin Hamid, et al. “Application of complex networks theory in urban traffic network researches”. In: *Networks and Spatial Economics* 19.4 (2019), pp. 1281–1317.

[152] Saket Navlakha and Ziv Bar-Joseph. “Algorithms in nature: the convergence of systems biology and computational thinking”. In: *Molecular systems biology* 7.1 (2011), p. 546.

[153] David Easley, Jon Kleinberg, et al. *Networks, crowds, and markets*. Vol. 8. Cambridge university press Cambridge, 2010.

[154] Shin Watanabe, Atsushi Tero, Atsuko Takamatsu, and Toshiyuki Nakagaki. “Traffic optimization in railroad networks using an algorithm mimicking an amoeba-like organism, Physarum plasmodium”. In: *Biosystems* 105.3 (2011), pp. 225–232.

[155] Liang Liu, Yuning Song, Haiyang Zhang, Huadong Ma, and Athanasios V Vasilakos. “Physarum optimization: A biology-inspired algorithm for the steiner tree problem in networks”. In: *IEEE Transactions on Computers* 64.3 (2013), pp. 818–831.

[156] David Aldous and Marc Barthelemy. “Optimal geometry of transportation networks”. In: *Physical Review E* 99.5 (2019), p. 052303.

[157] Michael Mc Gettrick. “The role of city geometry in determining the utility of a small urban light rail/tram system”. In: *Public Transport* 12.1 (2020), pp. 233–259.

[158] Marc Barthelemy. “Optimal Transportation Networks and Network Design”. In: *Spatial Networks*. Springer, 2022, pp. 373–405.

[159] Giulio Erberto Cantarella, G Pavone, and Antonino Vitetta. “Heuristics for urban road network design: lane layout and signal settings”. In: *European Journal of Operational Research* 175.3 (2006), pp. 1682–1695.

[160] Gilbert Laporte, JA Mesa, FA Ortega, and F Perea. “Planning rapid transit networks”. In: *Socio-Economic Planning Sciences* 45.3 (2011), pp. 95–104.

[161] Raphael Kay, Anthony Mattacchione, Charlie Katrycz, and Benjamin D Hatton. “Stepwise slime mould growth as a template for urban design”. In: *Scientific reports* 12.1 (2022), pp. 1–15.

[162] Atsushi Tero, Seiji Takagi, Tetsu Saigusa, et al. “Rules for biologically inspired adaptive network design”. In: *Science* 327.5964 (2010), pp. 439–442.

[163] Sybil Derrible and Christopher Kennedy. “Characterizing metro networks: state, form, and structure”. In: *Transportation* 37.2 (2010), pp. 275–297.

[164] David Levinson. “Network structure and city size”. In: *PloS one* 7.1 (2012), e29721.

[165] Sybil Derrible and Christopher Kennedy. “The complexity and robustness of metro networks”. In: *Physica A: Statistical Mechanics and its Applications* 389.17 (2010), pp. 3678–3691.

[166] Jingyi Lin and Yifang Ban. “Complex network topology of transportation systems”. In: *Transport reviews* 33.6 (2013), pp. 658–685.

[167] Aubrey Silberston. “Economies of scale in theory and practice”. In: *The Economic Journal* 82.325s (1972), pp. 369–391.

[168] Rainer Kujala, Christoffer Weckström, Richard K Darst, Miloš N Mladenović, and Jari Saramäki. “A collection of public transport network data sets for 25 cities”. In: *Scientific data* 5.1 (2018), pp. 1–14.

[169] Alexandre Litvine, Isabelle Séguy, Thibaut Thévenin, et al. *French Historical GIS, 1700-2020. Administrative Units, Populations, Transports, Economy*. Forthcoming. 2024.

[171] Aihui Pei, Feng Xiao, Senbin Yu, and Lili Li. “Efficiency in the evolution of metro networks”. In: *Scientific Reports* 12.1 (2022), pp. 1–10.

[173] Diego Baptista, Daniela Leite, Enrico Facca, Mario Putti, and Caterina De Bacco. “Network extraction by routing optimization”. In: *Scientific reports* 10.1 (2020), pp. 1–13.

[174] Jayanth R Banavar, Amos Maritan, and Andrea Rinaldo. “Size and form in efficient transportation networks”. In: *Nature* 399.6732 (1999), pp. 130–132.

[175] Jayanth R Banavar, Todd J Cooke, Andrea Rinaldo, and Amos Maritan. “Form, function, and evolution of living organisms”. In: *Proceedings of the National Academy of Sciences* 111.9 (2014), pp. 3332–3337.

[176] Rémi Louf, Pablo Jensen, and Marc Barthelemy. “Emergence of hierarchy in cost-driven growth of spatial networks”. In: *Proceedings of the National Academy of Sciences* 110.22 (2013), pp. 8824–8829.

[177] Sebastiano Bontorin, Giulia Cencetti, Riccardo Gallotti, Bruno Lepri, and Manlio De Domenico. “Emergence of complex network topologies from flow-weighted optimization of network efficiency”. In: *arXiv preprint arXiv:2301.08661* (2023).

[178] Liu Yang, Koen H van Dam, Arnab Majumdar, et al. “Integrated design of transport infrastructure and public spaces considering human behavior: A review of state-of-the-art methods and tools”. In: *Frontiers of Architectural Research* 8.4 (2019), pp. 429–453.

[179] Rémi Louf, Camille Roth, and Marc Barthelemy. “Scaling in transportation networks”. In: *PLoS One* 9.7 (2014), e102007.

[181] Philip M Dixon, Jacob Weiner, Thomas Mitchell-Olds, and Robert Woodley. “Bootstrapping the Gini coefficient of inequality”. In: *Ecology* 68.5 (1987), pp. 1548–1551.

[182] Hermina Petric Maretic, Mireille El Gheche, Giovanni Chierchia, and Pascal Frossard. “GOT: An Optimal Transport framework for Graph comparison”. In: *Advances in Neural Information Processing Systems 32*.

[183] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin Duke. “Gromov-wasserstein learning for graph matching and node embedding”. In: *International conference on machine learning*. PMLR. 2019, pp. 6932–6941.

[184] Riccardo Gallotti, Mason A Porter, and Marc Barthelemy. “Lost in transportation: Information measures and cognitive limits in multilayer navigation”. In: *Science advances* 2.2 (2016), e1500445.

[185] Dan Hu and David Cai. “Adaptation and optimization of biological transport networks”. In: *Physical review letters* 111.13 (2013), p. 138701.

[186] Vincenzo Bonifaci, Enrico Facca, Frederic Folz, et al. “Physarum-inspired multi-commodity flow dynamics”. In: *Theoretical Computer Science* (2022).

[187] Oded Cats. “Topological evolution of a metropolitan rail transport network: The case of Stockholm”. In: *Journal of Transport Geography* 62 (2017), pp. 172–183.

[188] Christa Brelsford, Taylor Martin, Joe Hand, and Luís MA Bettencourt. “Toward cities without slums: Topology and the spatial evolution of neighborhoods”. In: *Science advances* 4.8 (2018), eaar4644.

[189] Finnbar Lee, Kevin S Simon, and George LW Perry. “River networks: An analysis of simulating algorithms and graph metrics used to quantify topology”. In: *Methods in Ecology and Evolution* 13.7 (2022), pp. 1374–1387.

[190] Luca Carraro and Florian Altermatt. “Optimal Channel Networks accurately model ecologically-relevant geomorphological features of branching river networks”. In: *Communications Earth & Environment* 3.1 (2022), p. 125.

[191] Armaghan Abed-Elmdoust, Arvind Singh, and Zong-Liang Yang. “Emergent spectral properties of river network topology: An optimal channel network approach”. In: *Scientific reports* 7.1 (2017), p. 11486.

[192] Ilya Zaliapin, Efi Foufoula-Georgiou, and Michael Ghil. “Transport on river networks: A dynamic tree approach”. In: *Journal of Geophysical Research: Earth Surface* 115.F2 (2010).

[193] Charles E Perkins and David B Johnson. “Route optimization for mobile IP”. In: *cluster computing* 1 (1998), pp. 161–176.

[194] Jennifer Rexford. “Route optimization in IP networks”. In: *Handbook of Optimization in Telecommunications* (2006), pp. 679–700.

[195] Muhammad Aqib Javed, Muhammad Shahzad Younis, Siddique Latif, Junaid Qadir, and Adeel Baig. “Community detection in networks: A multidisciplinary review”. In: *Journal of Network and Computer Applications* 108 (2018), pp. 87–111.

[196] Daniela Leite, Diego Baptista, Abdullahi A Ibrahim, Enrico Facca, and Caterina De Bacco. “Community detection in networks by dynamical optimal transport formulation”. In: *Scientific Reports* 12.1 (2022), p. 16811.

[197] Mark EJ Newman. “The structure and function of complex networks”. In: *SIAM review* 45.2 (2003), pp. 167–256.

[198] Mark D Humphries and Kevin Gurney. “Network ‘small-world-ness’: a quantitative method for determining canonical network equivalence”. In: *PloS one* 3.4 (2008), e0002051.

[199] Mark EJ Newman and Aaron Clauset. “Structure and inference in annotated networks”. In: *Nature communications* 7 (2016), p. 11863.

List of Figures

List of Tables

7 Additional Material for Chapter 3

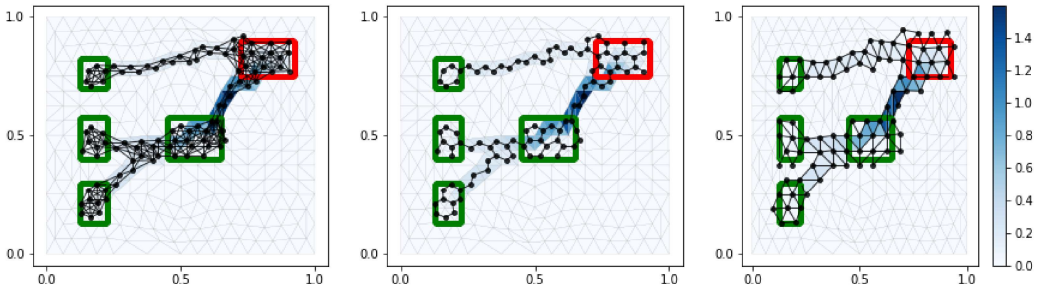
Here we provide additional material for Chapter 3.

- Section 7.1 and Section 7.2 show alternative choices for selecting the graphs on the pre-extraction step, and several examples of resulting networks with distinct setups, respectively.
- Section 7.3 details the different criteria to select sources and sinks.
- Section 7.4 describes the pipeline for graph extraction from images.

7.1 Examples of graph pre-extraction routines

We provide here several examples of networks resulting from different routing optimization setup for each of the three graph definitions.

(a) $\beta = 1.1$: traffic consolidation



(b) $\beta = 1.0$: shortest-path optimization

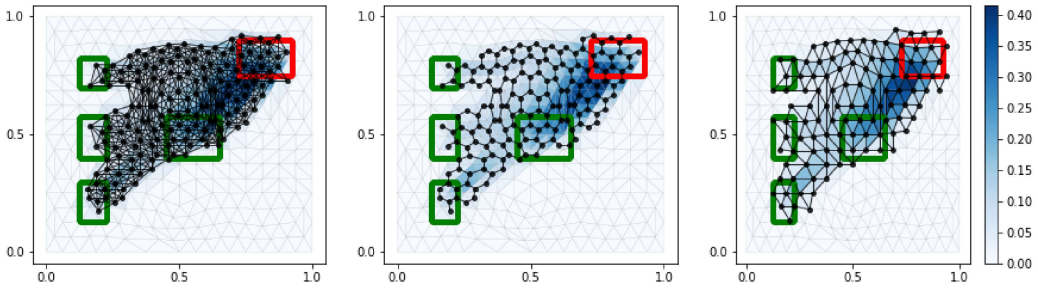


Fig. 7.1: *Graph pre-extraction* rules for three transportation optimization problems. Left: edge-or-node sharing; center: edge-only sharing; right: original triangulation. We monitor the density μ and use parameters: $\mu_0 = 1$, $f = 5$rch, $\delta = 0.0001$, $w_{ij} =$ ER.

7.2 Additional networks and routing optimization scenarios

We show several examples of networks resulting from different routing optimization setups for different configuration choices of our proposed routines. This should serve as an example guideline on what parameters to choose based on the application. More specifically, we show three different setups given in input as initial routing optimization problems to the DMK-Solver. We consider: i) different locations of sources and sinks to show how this impacts the formation of branches and their symmetry; ii) different values of $\beta \in \{1.2, 1.3, 1.5\}$ to show how traffic consolidates into fewer edges as β grows; iii) different initial $\mu_0(x, y)$ (parabolic, delta-like and uniform) to show variability in the initial transport density.

(a) Parabolic, $\beta = 1.2$ **(b)** Delta, $\beta = 1.3$ **(c)** Uniform, $\beta = 1.5$

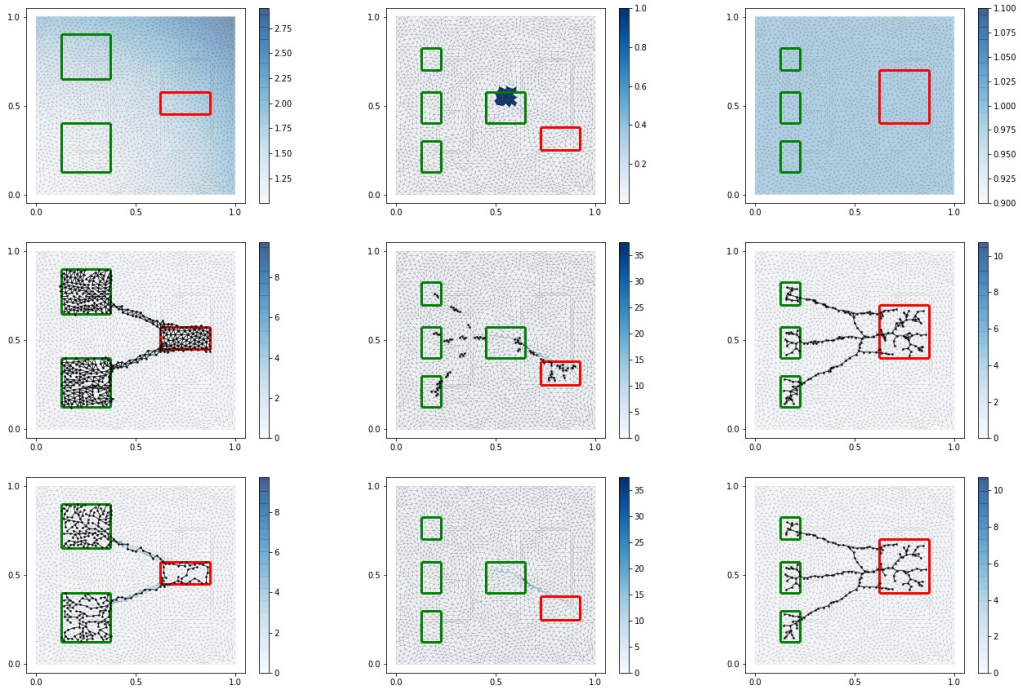


Fig. 7.2: Network extraction examples for branched transportation optimization. Columns denote a particular setup based on source/sink locations and values of β and μ_0 as described in the sub-captions. Each column represents different initial transport densities (parabolic, delta and uniform,respectively), while rows represent a step of our protocol: (top) initial transport density $\mu_0(x, y)$ given in input to DMK-Solver; (center) graph *pre-extraction* using III-AVG (left), II-AVG (center) and I-AVG (right); (bottom) graph *filtering* with weight assigned with rule IBP (left and right) and no simplification (center). The color scheme refers to the value of μ. Sources and sinks are points inside the green and red rectangles, respectively. Note that pre-extraction rule II tends to break the network into disconnected components. This is also the reason why this rule scores well in terms of total path length (see Section 3.5 and Section 3.5). Since the path is already broken and sources cannot reach the sinks, it does not make sense to apply the filtering, hence we left the center-bottom plot empty.

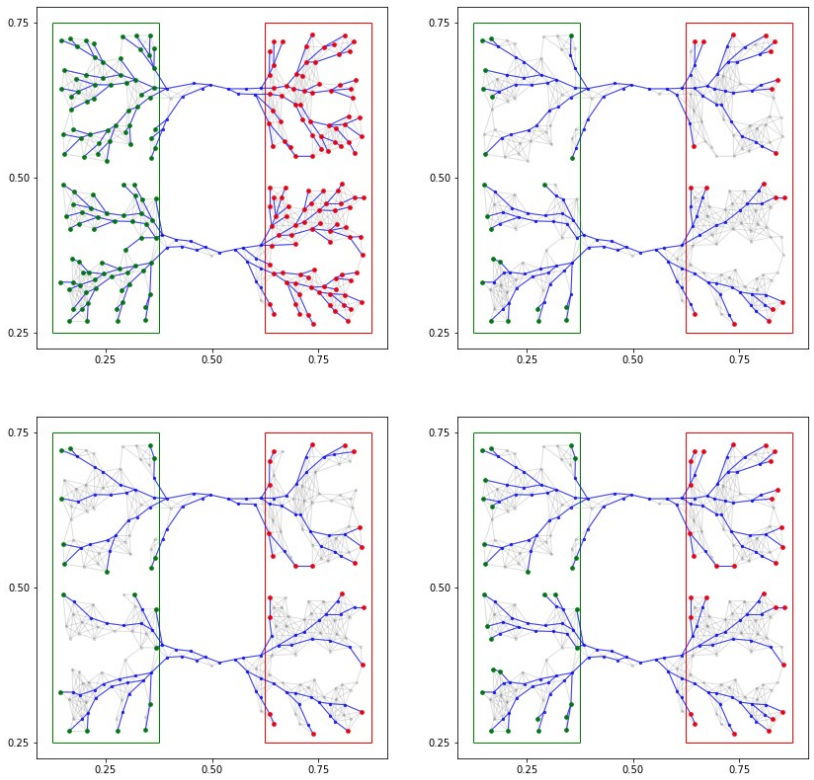


Fig. 7.3: Choosing sources and sinks. Top-left) no selection of sources and sink ($\tau_{BC} = 1$); top-right) selection using betweenness centrality only ($\tau_{BC} = 0.01$) ; bottom-left) selection using convex hull-only ($\tau_{BC} = -1$); bottom-right) selection with convex-hull and betweenness centrality criteria ($\tau_{BC} = 0.01$).

7.3 Source and sink selection

Selecting source and sink is an important task during the graph filtering step as explained in Section 3.4.2. Here we give examples showing why this is the case. Specifically, for a particular problem setup, we consider four cases: i) no selection of sources and sinks. This means that we simply consider sources and sinks *all* the points inside the support of the forcing function $f(x)$, input of the DMK-Solver. In this case, the final network structure is dominated by the many branches connecting the sources and sinks within the support of $f(x)$, thus hindering the contribution of the bulk of the network, the one connecting sources and sinks; ii) convex hull-only: we set as source (or sink) all the points inside the convex hull of the set of eligible sources (or sinks); iii) betweenness centrality-only: we set as source (or sink) all the eligible sources (or sinks) that have betweenness centrality smaller than a threshold $0 \leq \tau_{BC} \leq 1$. These two criteria miss possibly relevant sub-branches, as shown in Figure 7.3 top-right and bottom-left; iv) selection using both convex-hull and betweenness centrality criteria. In this case, we capture the relevant sub-branches from both criteria, as shown in the bottom-right of Figure 7.3. In Chapter 3 we always consider this final criterion as it seems to capture all the relevant branches inside the support of $f(x)$. However, we encourage the final user to choose what criteria to choose based on the application. Figure 7.3 should help driving this decision.

7.4 Network extraction from images

We describe here the steps followed to get the network shown in Section 3.6 in Chapter 3[1]. The original figure contains loops, however our protocol in its standard settings with filtering can only generate tree-like structures. Therefore, if we want to obtain a network with loops, we should consider a minor modification of our routine, as explained in detail below. In short, we need to find first a tree-like network close to the original image with loops and then give this one in input to our routine. Then, after applying our protocol, we can obtain loops by adding edges that properly *close* the loops in the tree. This can all be done in an automatized way as explained below. Otherwise, if obtaining loops is not required, our routine can be used with no modifications. The whole procedure can be divided into 6 parts: image selection and approximation, graph pre-extraction, tree reduction, terminal identification, graph filtering and edge correction.

1. Image selection and approximation: this consists in choosing an image and, if desired, mapping it into a reduced version of it. This reduction is useful when dealing with high-resolution images. We use the *resize* function included in *OpenCV*. The pixel values for the reduction image can be computed using different interpolation methods. We use linear interpolation (*cv.INTER_NEAREST*). The original image's dimensions are 1200x1200. The reduced ones are 100x100.

2. Graph pre-extraction: the RGB values of the pixels are mapped into an integer in a one-to-one way, and then they are used to define the function μ^* on each pixel. The graph G is obtained using rule I-ER as defined in Section 3.3.1 of Chapter 3.

3. Tree reduction: the filtering could be applied directly on G after choosing some sources and sinks, but this will generate a filtered tree-like network, i.e., no loops from the image will be captured. Thus, we propose to change G by a tree graph whose structure could then be easily "corrected" (after the filtering step) to get the mentioned loops. This tree is taken to be the *Breadth First Search* graph (G_{BFS}) of G from a random root.

4. Terminal identification: terminals (the union of sources and sinks) are defined using *filters*, i.e., squared pieces of the image. We place terminals in the graph by counting how many intersections are between the image and the boundary of the filter: if just one intersection is found, then a terminal is placed in the node with the lowest closeness centrality; if three or more intersections are found, then a terminal is placed in the node with the highest closeness centrality. We cover the whole image by disjoint filters, thus we do not miss the relevant parts of the image. Notice that if some parts of the image do not have a representative in one of the two source or sink sets, then they will not be part of the filtered graph.

5. Graph filtering: G_{BFS} is filtered as in Section 3.4 of the Chapter 3 (to obtain a graph G_f) by defining the source set (S^+) to be a random element on the set of terminals and the sink set (S^-) to be the remaining ones; $\beta_d = 1$ in Fig 7.

6. Edge correction: to *close* the loops we use the following rule. For each pair of terminals $u, v \in G_f$, if the shortest path P^{uv}_{BFS} on G_{BFS} is *long* but the shortest path P^{uv} on G is *short,* then add P^{uv} to G_f. The notions of *short* and *long* paths are precisely defined by introducing two parameters L_{BFS} and L_G, respectively. We say that a path P is *long*, if

[1] We used image "IMG_0379.jpg" stored at KIST_Europe_data _set/raw_images/ motion16/ inside the SMGR repository [79]. The dimension of the original one is 5184x3456 pixels. We used a 1200x1200-pixel section in Fig. 7. The coordinates of the bottom-left vertex of the box are $(2000, 2000)$ (assuming $(1, 1)$ to be the bottom-left pixel of the original image).

$l(P) > L_{BFS}$; and it is *short,* if $l(P) < L_G$, where l is the number of nodes in the path P; the values of L_{BFS} and L_G can be tuned based on the system size at hand.

8 Additional Material for Chapter 4

Here we provide additional material for chapter Chapter 4.

- Section 8.1 has additional examples for communities obtained from real networks, detected with the multiple community detection algorithms presented in Chapter 4.
- Section 8.2 details the ARI score, main quantity used to evaluate the performances of the community detection algorithms.
- Section 8.3 describes additional experiments made with random networks generated from real network structures.

8.1 Community Detection in additional Real networks

This section presents additional plots for real networks on communities detected by some algorithms studied in Chapter 4. Figure 8.1a and Figure 8.1b exhibit the detected communities for American football and Political books, respectively. For each dataset we compare three algorithms with the ground truth, and with ORC-Nextrout having a single setting of β.

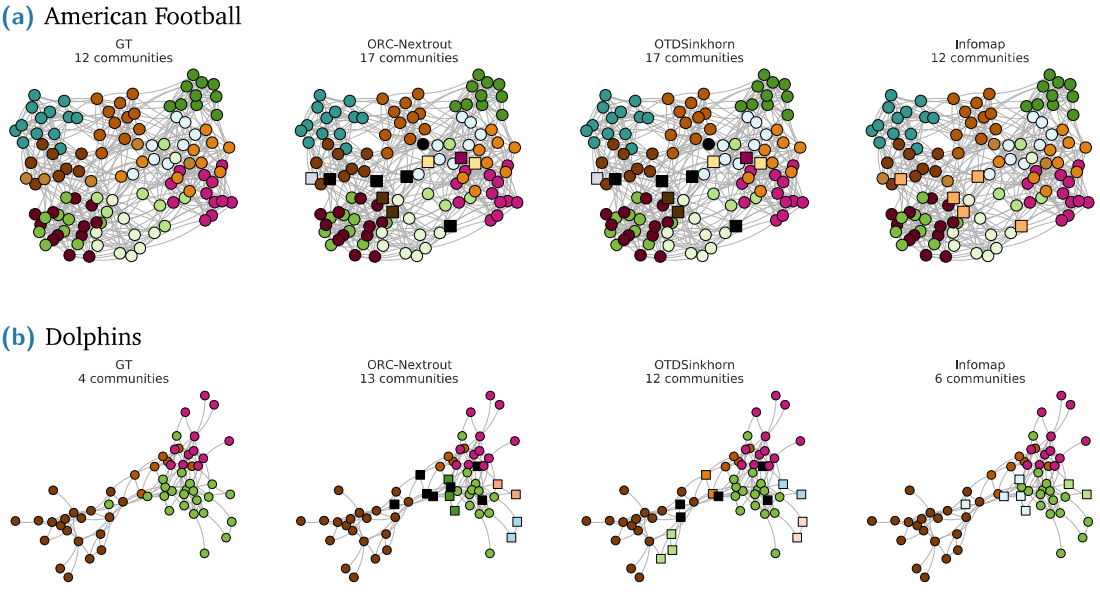


Fig. 8.1: Communities in real networks. We show the communities for American football (a) and Dolphins (b) inferred by ORC-Nextrout ($\beta = 2.0, 1.5$ for top and bottom rows respectively), OTDSinkhorn and Infomap and compare against those extracted using node attributes (GT). Dark nodes represent individual nodes who are assigned to isolated communities by OT-based methods. Square-shaped markers denote nodes assigned to communities different than those obtained from node metadata.

8.2 ARI score

The *Rand Index* [136] (RI) is a measure of the similarity between two sets of clusters. Let $S = \{x_1, x_2, ..., x_n\}$ be a set of n different elements, and let $X = \{X_1, X_2, .., X_m\}$, $\hat{X} = \{\hat{X}_1, \hat{X}_2, .., \hat{X}_p\}$, be two partitions of S into m and p subsets, respectively. Intuitively, X can thought of as the "correct" separation of S into classes, while $\hat{X}$ would be a "prediction" of it. We would like to understand the quality of the prediction $\hat{X}$ in terms of the ground truth community information X. Consider

- TP (*true positive*) is the number of times that a pair of elements (x_i, x_j) belonging to the **same** class X_k gets assigned to the **same** class $\hat{X}_l$.
- TN (*true negative*) is the number of times that a pair of elements (x_i, x_j) belonging to **different** classses in X gets assigned to **different** classes in $\hat{X}$.
- FP (*false positive*) is the number of times that a pair of elements (x_i, x_j) belonging to **same** class X_k is (*falsely*) assigned to **different** classes $\hat{X}_l$ and $\hat{X}_m$.
- FN (*false negative*) is the number of times that a pair of elements (x_i, x_j) belonging to **different** classes X_k and X_l is (*falsely*) assigned to the **same** class $\hat{X}_m$.

One can think of the the words *positive* and *negative* referring to whether two elements in S belong to the same or to different classes, respectively. The words *true* and *false* would then judge the performance of the prediction: if it maches the nature of the elements under inspection, then the word *true* is associated to it; *negative*, otherwise. The Rand Index is then computed using the formula:

$$RI(X, \hat{X}) = \frac{TP + TN}{TP + FP + FN + TN}.$$

The *Adjusted Rand Index* (ARI) is the *corrected for chance* version of the RI:

$$ARI(X, \hat{X}) = \frac{RI - \mathbb{E}[RI]}{1 - \mathbb{E}[RI]},$$

where $\mathbb{E}[RI]$ is the expected value of the RI under the assumptions that the partitions X and $\hat{X}$ are sampled from the generalized hypergeometric distribution. Closed forms for the terms shown in the ARI formula can be computed. See [136] for a more detailed presentation of the RI and ARI scores.

8.3 Tests on random networks generated from real structures

The pseudo-code of the random processes defined in Section 4.3.3 is shown in the Algorithm 4 and line 4. We show two examples of the outputs of these algorithms on Figure 8.2c together with the adjacency matrix that was used to originally build them. Notice that the matrix in panel (b) is different from that in panel (a) both in terms of intra and inter-community blocks, whereas that in (c) only differs along the within-community entries. This indicates that the first method alters the overall configuration of the edges in the network by adding random noise, whereas the second changes the original network only by reducing the intra-community relationships.

Algorithm 4: Flipping entries of the adjacency matrix

Input: $G = (V, E, W)$, flipping proportion $r \in [0, 1]$, flipping probability $p \in [0, 1]$

Build adjacency matrix of G: A
Make a copy of A: A'
Compute number of nodes in G: N
for $t \in range(r * N^2)$ **do**
  for $i \in range(N)$ **do**
    for $j \in range(i + 1, N)$ **do**
      Assume $\mathbb{P}_{ij}(A[i][j]) = p$
      Sample $A'[i][j]$ from $\mathbb{P}$
    end
  end
end
Symmetrize A'
Build G' using the connections between nodes described by A'.

Output: $G' = (V, E')$

Algorithm 5: Removing intra-community edges

Input: $G = (V, E, W)$, removal proportion $r \in [0, 1]$,

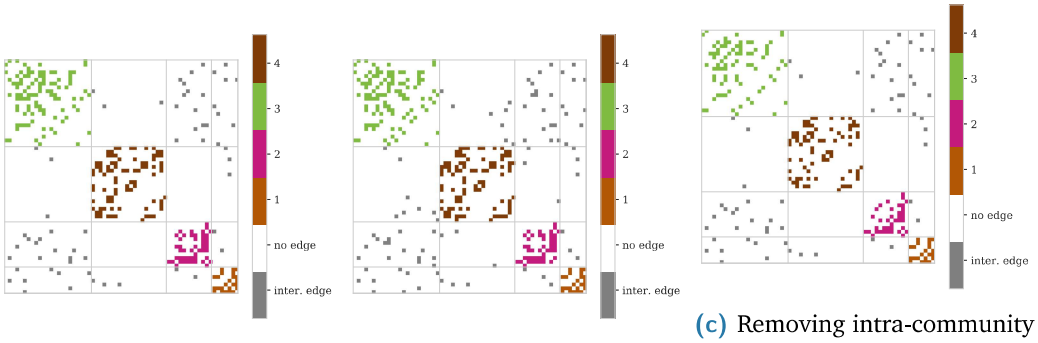
Build list of edges of G: $E(G)$
Make a copy of E: $E'(G)$
Compute number of edges in G: M
Remove from $E'(G)$ all the edges $e = (i, j)$ such that either i or j is a leaf in G
Remove $r * M$ elements from $E'(G)$ uniformly at random
Define G' using V and $E'(G)$

Output: $G' = (V, E')$

Results for Les-Miserables dataset

We show results obtained on the Les-Miserables dataset for the test where we flip at random the entries of the adjacency matrix A (see Figure 8.3). In this case, ORC-Nextrout outperforms Infomap only in the case where $r = 0.005$, i.e. when 5% of elements of A are changed. As r increases, Infomap increases its accuracy. On the other hand, ORC-Nextrout shows a better performance than OTDSinkhorn consistently across values of r.

Lastly, in Figure 8.4. we show the results obtained for both tests on the Dolphins dataset. It can be seen that ORC-Nextrout outperforms OTDSinkhorn consistently across values of r in both cases. On the other hand, Infomap has a higher accuracy in both tests, as expected given the results shown in Figure 4.5 of the Chapter 4.



(a) Original adjacency matrix (b) Flipping entries (c) Removing intra-community edges

Fig. 8.2: Adjacency matrices for original and perturbed Dolphins network. We show the adjacency matrix of (a) the original dataset, (ii) a perturbed network built from the previous one by flipping entries at random and (c) a perturbed graph obtained by removing intra-community edges. We used $r = 0.05$ and $r = 0.15$ to generate the matrices shown in (b) and (c), respectively. Nodes are grouped by communities to highlight the block structure. Gray lines denote the boundaries of these blocks. Diagonal blocks represent the communities. Off-diagonal blocks show connections between communities. Colored entries are in agreement with those of community layout shown in Figure 8.1b. Inter-community connections are highlighted in gray.

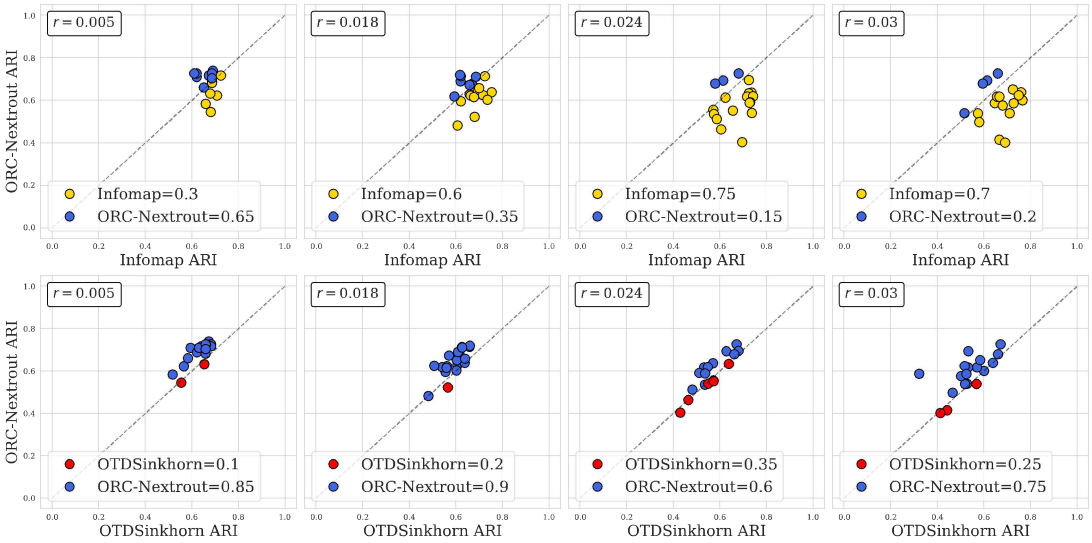


Fig. 8.3: Flipping-entries test on Les Miserables data. Markers correspond to 20 instances of semi-synthetic networks generated from real data. Their (x, y) coordinates are the ARI scores of the method indicated on axes. Colors are given by the best performing algorithm, e.g. if $x > y$, the color of the method associated to x is chosen. The legend shows the percentage of times that the corresponding method outperforms the other. The parameter r describes the proportion of entries of the adjacency matrix A that have been changed. This increases from left to right.

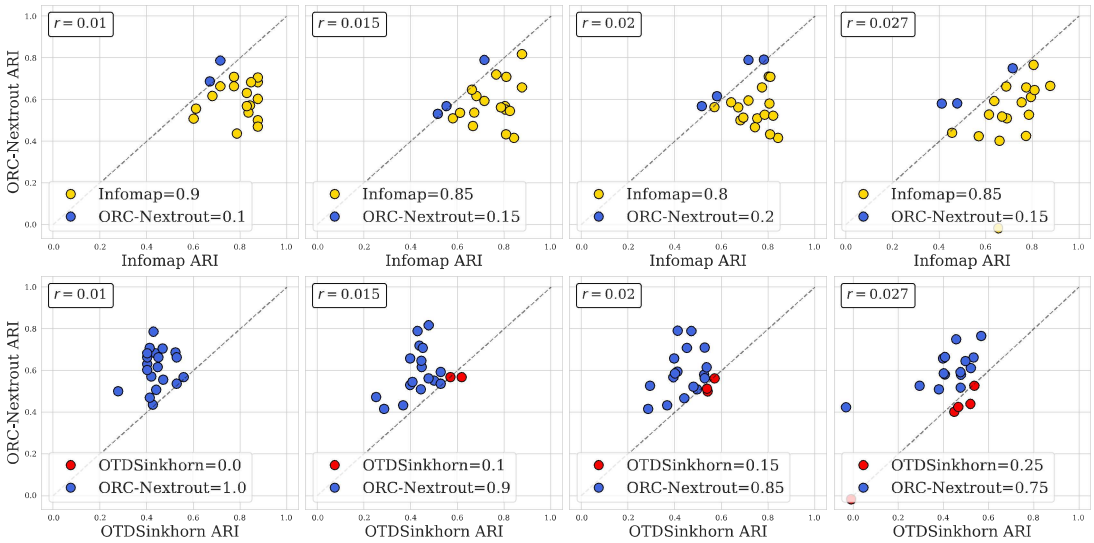


Fig. 8.4: Flipping-entries test on Dolphins data. Markers' description is similar as in Figure 8.3.

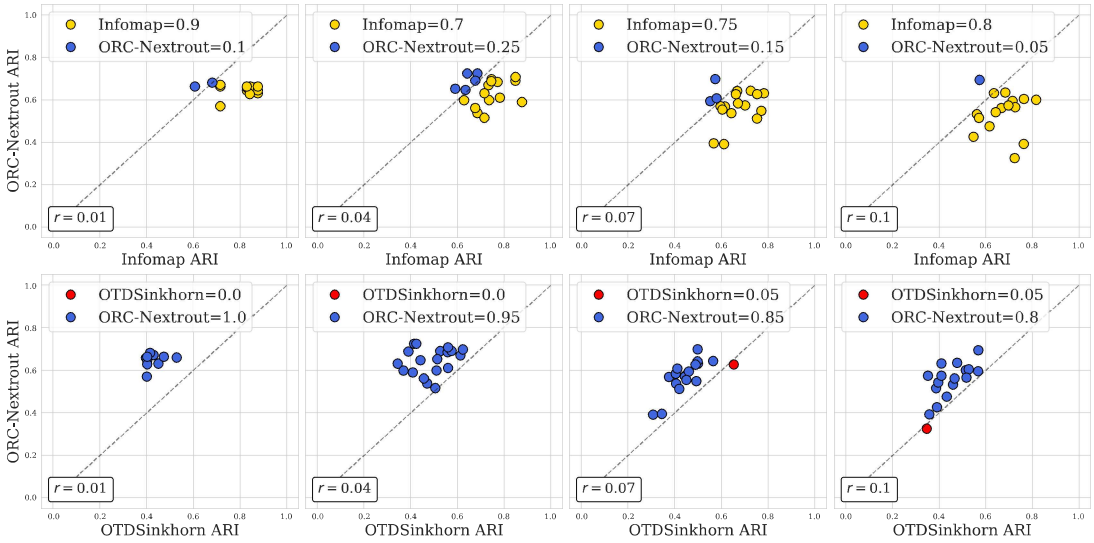


Fig. 8.5: Removing intra-community edges test on Dolphins data. Markers' description is similar as in Figure 8.3.

9 Additional Material for Chapter 5

Here we provide additional material for chapter Chapter 5.

- Section 9.1 evaluates the Wasserstein-based metrics to automatically the select networks presented in Chapter 5.
- Section 9.2 analyses how the different weights change the final network selection.
- Section 9.3 highlights the importance of selecting the sink with different centrality criteria, thus complementing what was presented in Chapter 5, Section 5.2.1.
- Section 9.4 shows traffic distribution for the different values of β
- Section 9.5 and Section 9.6 highlights two important examples: the New York subway and the evolution of the French railway system in a period of around 40 years along the 1800's. We also highlight some limitations of the approach described in Chapter 5.

9.1 Validating the Wasserstein Metric

To evaluate how the simulated networks selected with our Wasserstein-based metric perform in terms of the main topological properties defined in Chapter 5, we define a similarity ratio r_p for each of them. Given a particular property p, we compute the ratio between the value measured on both simulated and observed networks. Thus, when $r_p = 1$, the simulated network extracted by Nextrout matches perfectly with the observed one in terms of property p. For instance, when p is the cost, then $r_{TL} = TL_{\text{Nextrout}}/TL_{\text{real}}$.

In Figure 9.1 and Figure 9.2 we show the ratio values for the multiple topological properties, comparing them across different automatic selections of β. Specifically, we obtain for each property p the value of β_p that leads to a simulated network closer to the observed one in terms of property p, i.e. corresponding to a r_p closer to 1. By definition, the simulated graph obtained with β_p has the best r_p for property p, but this may not be true for other properties measured on that same graph. For instance, the graph selected with β_{TL} (second row of Figure 9.1) as $r_{TL} \approx 1$, thus reproducing well the number of edges, but it tends to largely overestimate both $Gini(T_e)$ and the number of bifurcations, with $r_p > 2$ for these two properties (see Figure 9.2). Instead, we observe that the simulated graph selected with the Wasserstein measure (first row of Figure 9.1) has on average r_p closer to 1 across various properties. In other words, it shows transportation properties that are consistently more aligned to those behold by the observed network.

To validate the consistency of the Wasserstein measure, we compute how each data point in Figure 9.1 and Figure 9.2 differs from the perfect match $r_p = 1$. Specifically, we define the minimum mean displacement $\mathcal{D}_d = \frac{1}{|p|}\sum_p \left(1/n \sum_{i=1}^{n} |r_{p_i} - 1|^d\right)^{\frac{1}{d}}$, where p refers to each considered metric ratio ($|p| = 4$), $n = 17$ is the number of data points and $d = 1, 2$. The closer to 0, the less the considered measure deviates from the perfect score. We found that the Wasserstein measure deviates with $\mathcal{D}_{d=1} = 0.5$, whilst the cost has $\mathcal{D}_{d=1} = 0.61$, the total length $\mathcal{D}_{d=1} = 0.69$, the traffic $\mathcal{D}_{d=1} = 0.35$ and the density of bifurcation points has $\mathcal{D}_{d=1} = 0.32$. Similar results are found with the square displacement $d = 2$. The traffic and

number of bifurcations have the lowest displacements, with the Wasserstein following. As the Wasserstein tends to select networks with higher β (not shown here), this measure encourages smaller cost (TL). The fact that we see a higher displacement than the one of traffic and density of bifurcations is a sign that real networks could be further optimized in terms of TL. Nevertheless, the gap is small and beyond problem-specific properties (as the traffic or density of bifurcations). These results confirm the benefit of using our Wasserstein-based similarity measure to automatically select a simulated network across different values of β in our Optimal Transport-based setting. More generally, it provides a robust and meaningful measure to compare network structures that can be used in applications beyond the one considered in this work.

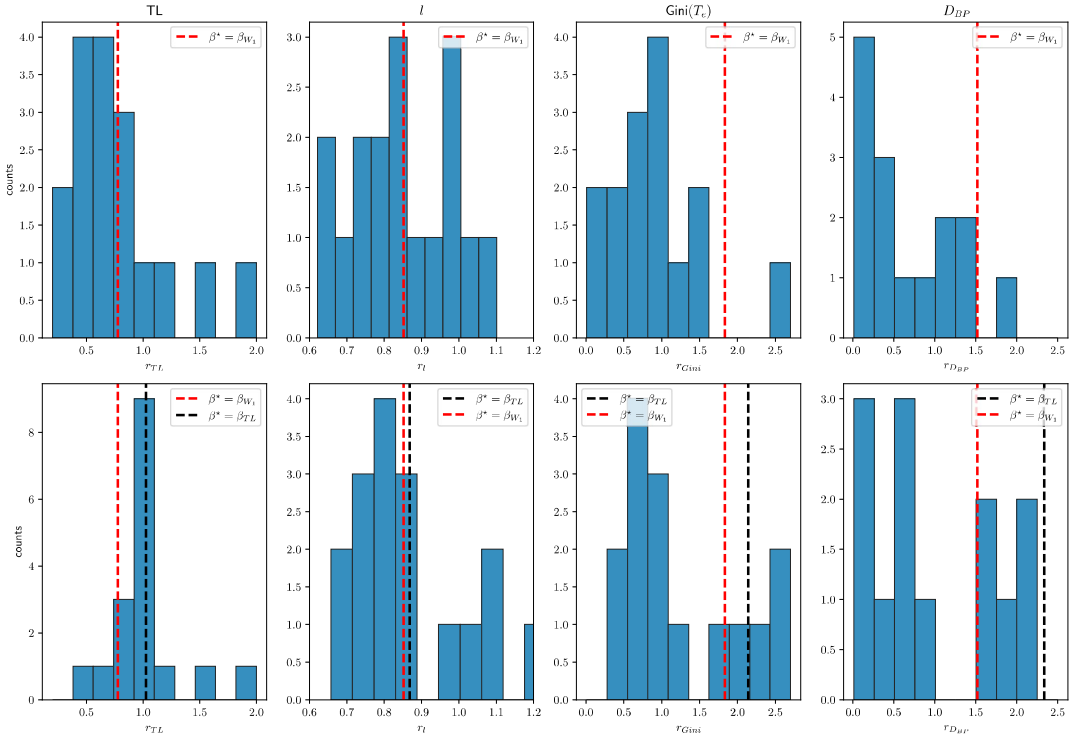


Fig. 9.1: Counts for the metric ratios of the Wasserstein and cost (TL). Given a particular metric (e.g. Wasserstein, first row), we select the optimal β for each data point and measure the ratio across the multiple considered metrics. The red lines highlight the average for the optimal β given by the Wasserstein measure. We then repeat this procedure for the TL and highlight how the average of $\beta^\star = \beta_{Wass}$ changes given other optimal β.

As the final optimal network changes depending on specific selected transportation properties, a natural question is how distinct these networks are and how do such properties favor different optimal β. In Section 9.1 we show an example for the subway network of Berlin to illustrate this difference. While $\beta^\star = \beta_{Gini}$ favors a network with more branches and wider coverage, for $\beta^\star = \beta_{D_{BP}}$ the optimal network is that with fewer branches. This difference is thus evaluated across multiple properties using the metric ratios. Overall, we notice that the Wasserstein encourages smaller path length, but higher traffic and density of bifurcation points (which is confirmed by the minimum mean displacement).

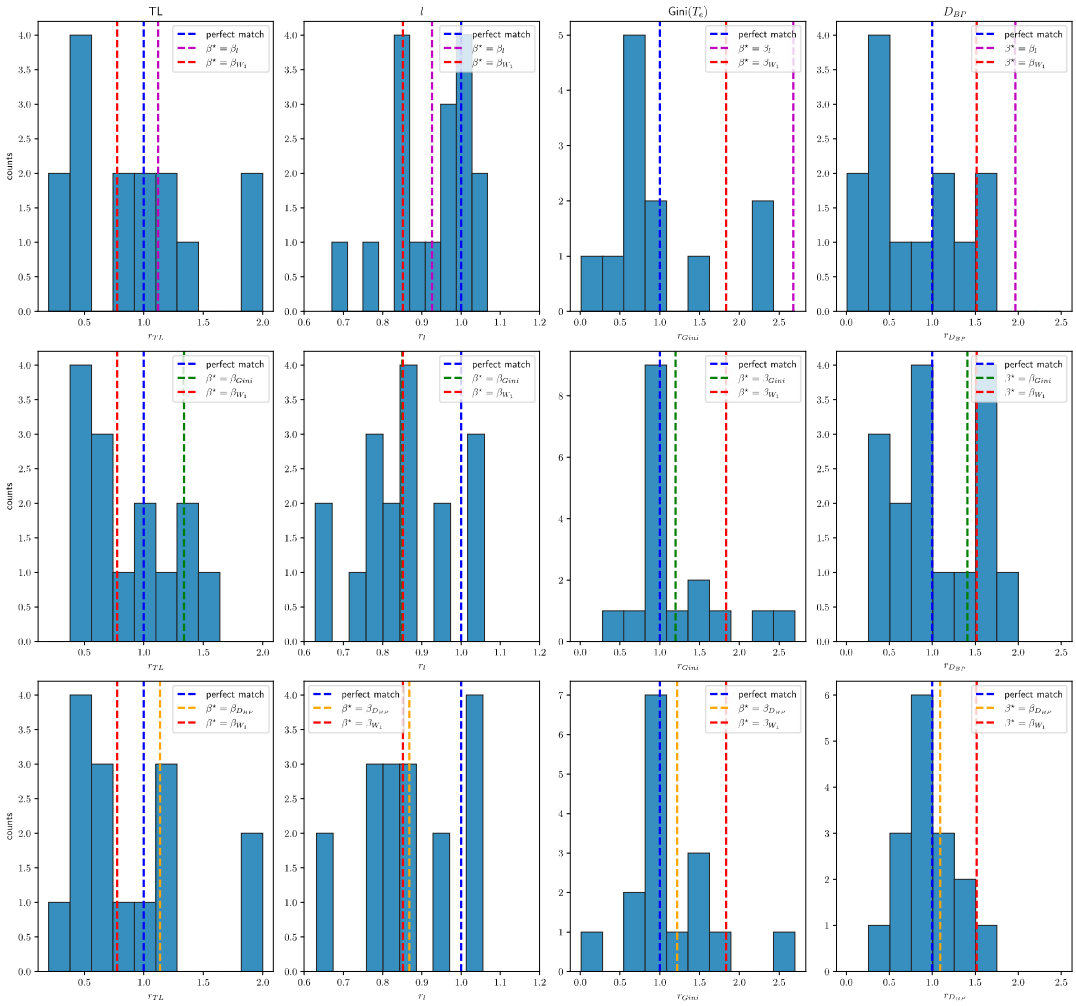


Fig. 9.2: Metric ratios for the path length l, Gini and density of branching points. We measure the ratio r_p across the three metrics, highlighting how the average of $\beta^\star = \beta_{Wass}$ - given by the red lines - changes given other optimal β, and how it deviates from the '*perfect match*', i.e, when $r_p = 1$.

9.2 Wasserstein weights

The Wasserstein similarity measure (W_1) is designed to capture the amount of 'effort' necessary to move information between the original network and the one extracted using our pipeline. Since it is defined as the weighted sum of the optimal fluxes, the choice of such weights might influence the final optimal network. We compared the differences of selecting unitary or euclidean weights in Figure 9.4b.

9.3 Centrality criteria for selecting origins and destinations

The first step of our pipeline consists in defining the set of Origin-Destination pairs (OD) taken from a real network. We perform a preprocessing step to remove possible redundancies found in the original spatial data collected from [168].

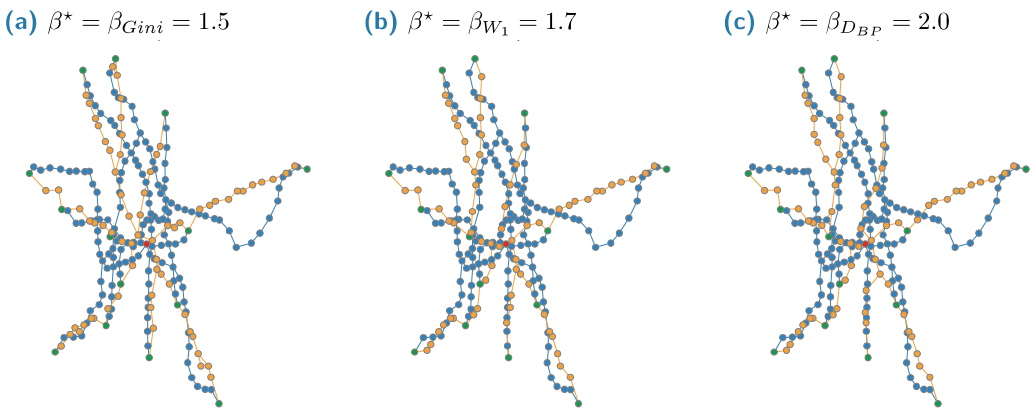


Fig. 9.3: Optimal networks for the subway network of Berlin. In (a) the optimal β is that given by the $Gini$, which tends to favor the networks with more branches and wider coverage to distribute the traffic; in (b) we show the optimal network as selected by the Wasserstein measure, whilst in (c) we show the one given by the density of branching points, which tends to favor networks with fewer branches, so higher values of β.

We map each stop, which consists in a pair longitude-latitude, to a node in a $[0, 1]$ system of coordinates as a starting point. We then perform the preprocessing to obtain the original network mapping, and compute different network centralities to define the set of OD points: nodes with lowest values of degree, betweenness and closenness centrality are set to be origins and those with the highest values of these properties are set as destinations. In nearly all networks we noticed that the selected nodes for betweenness and closenness were equivalent, therefore the Nextrout generated networks would have the same set of nodes and edges (hence the measured properties would be equivalent), so we find it enough to show the results for networks generated using only degree and betweenness centrality.

As we have already shown results obtained for the networks with OD points selected based on the degree in Chapter 5, we now present the network properties obtained based on the betweenness centrality. Notice that despite the absolute number of destinations being the same, they still differ in terms of location, which will impact the final networks generated using our pipeline. The final betweenness properties are shown in Figures 9.5 to 9.6.

9.4 Traffic distribution

We also show how traffic is distributed along the network. We measure traffic on edges (T_e) by setting the same origins and destination nodes used on the first step of our extraction pipeline, and running the discrete DMK-dynamics with $\beta = 1.5$. Figure 9.7 shows that with this setting, both extracted and real networks tend to distribute traffic towards the more central edges, which is usually observed in real transportation systems, where stations with more connections register higher flow of passengers.

9.5 Limitations of our approach

In order to recover the entire structure of the New York subway system, given its size and complexity, we first selected multiple sources (nodes with smallest degree) with one randomly

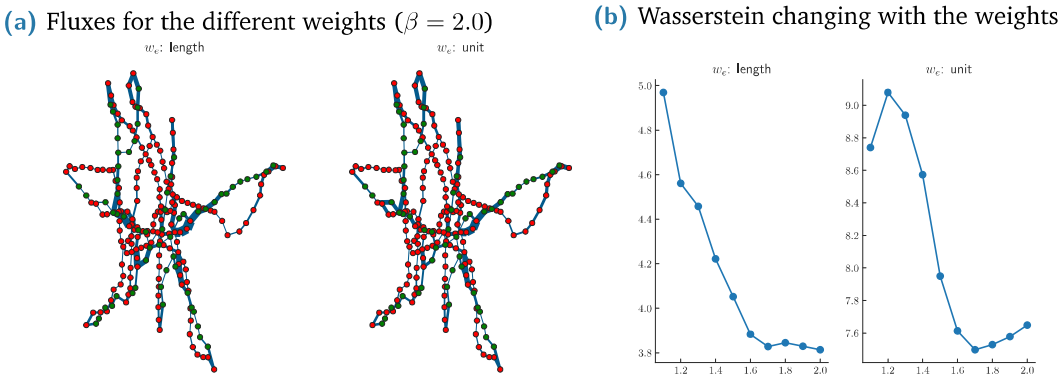


Fig. 9.4: Selecting weights for the Wasserstein similarity measure. (a) Given the source graph (red nodes, real network) and the sink graph (green nodes, extracted network), we observe how the optimal flux distributes along the union graph. The euclidean length (left) distributes the flux along the more central nodes, thus capturing more realistic scenarios, whilst unitary weights distribute more equally along the network. We then compare in (b) how the Wasserstein measure changes across different values of β. We design the Wasserstein measure to capture more realistic scenarios, thus for smaller values of β, the networks are expected to have more redundant nodes, so we expect this measure to be higher, and the opposite as β increases. This pattern seems to be better captured by the euclidean weights.

selected sink among the nodes with highest degree. This strategy fails as shown in Figure 9.8, where the generated topology does not show clear similarity with the original network. This evidences a limitation of our approach for selecting sources and sinks in more complex, bigger structures.

The original topology contains multiple lines that can be individually seen as an independent network itself, generating loops when superimposed. As already presented in Section 5.3.2, each obtained network shows a higher similarity with the correspondent original line. A comparison of the superimposed lines is shown in Figure 9.9. This strategy is particularly suited for understanding more complex structures, specially for loop recovery, and it can be further explored in future work.

9.6 The evolution of the French railway system

We show in Figure 9.10 how the French railways changed in a period of around 40 years. It justifies the usage of a network from 1850 in our experiments, where the topology does not yet contain multiple loops and it is in its initial stage.

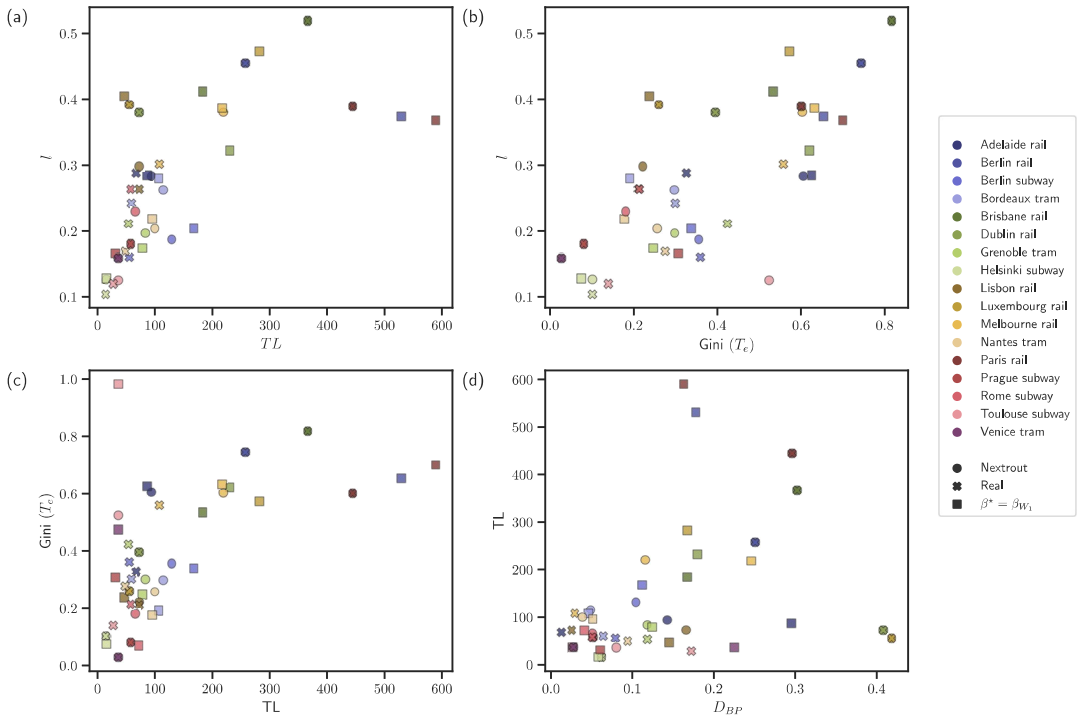


Fig. 9.5: Measures for the studied networks. Each dataset is assigned to a different color, market shapes distinguish real and simulated networks. The ladder are further distinguished based on the one generated via Nextrout (having betweenness as the destinations selection criteria) that gives the closest point in terms of the metrics plotted in the figure (circle) or the one corresponding to the best Wasserstein measure (square). (a) We measure Cost (TL) for both simulated and real networks, plotted against the total path length l. (b) Gini coefficient as a measure of traffic distribution, versus the total path length. (c) Traffic distribution in terms of the Cost. (d) Density of bifurcations plotted against the cost.

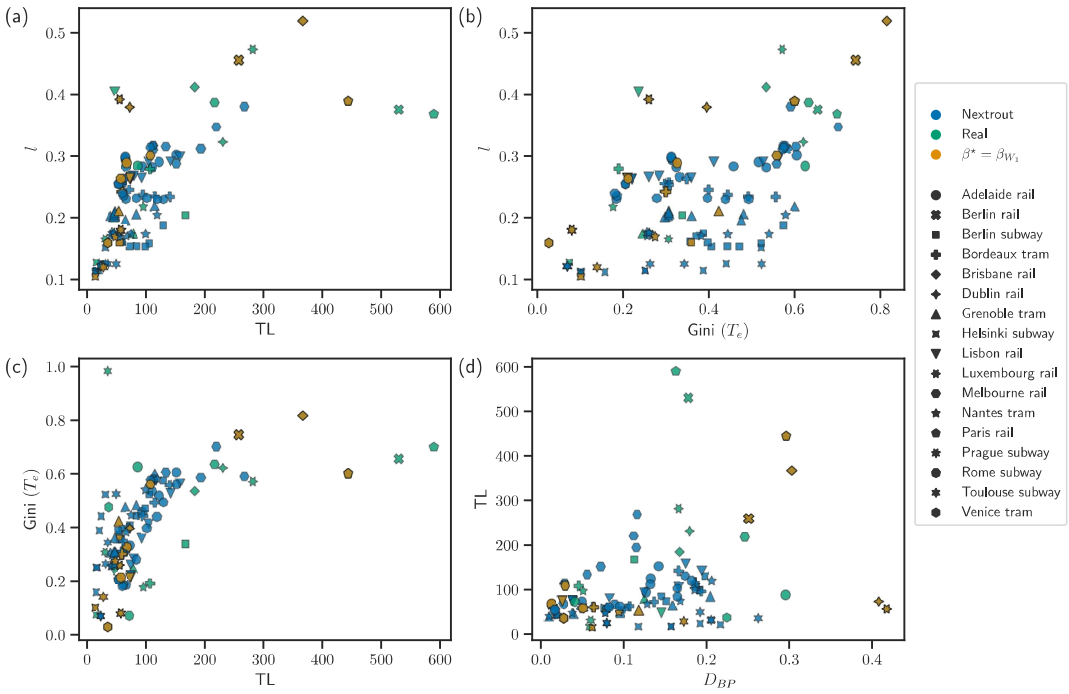


Fig. 9.6: Comparison of simulated networks using the betweenness criteria and the one from observed data. We show the values of the main transportation properties investigated in this work for real and simulated networks. Simulated networks cover a wider range of properties' values, thus allowing in particular to select network that have lower or comparable values of these properties than those observed in the corresponding real networks.

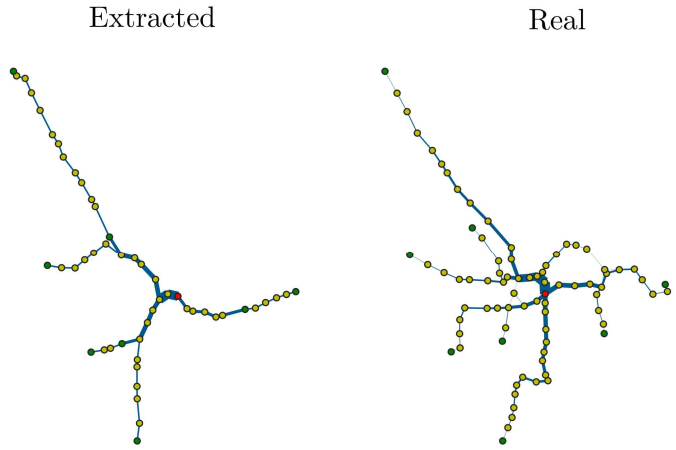


Fig. 9.7: Traffic in extracted and real networks. Using the same set of origins and destinations we observe a higher distribution of traffic towards more central edges in both cases.

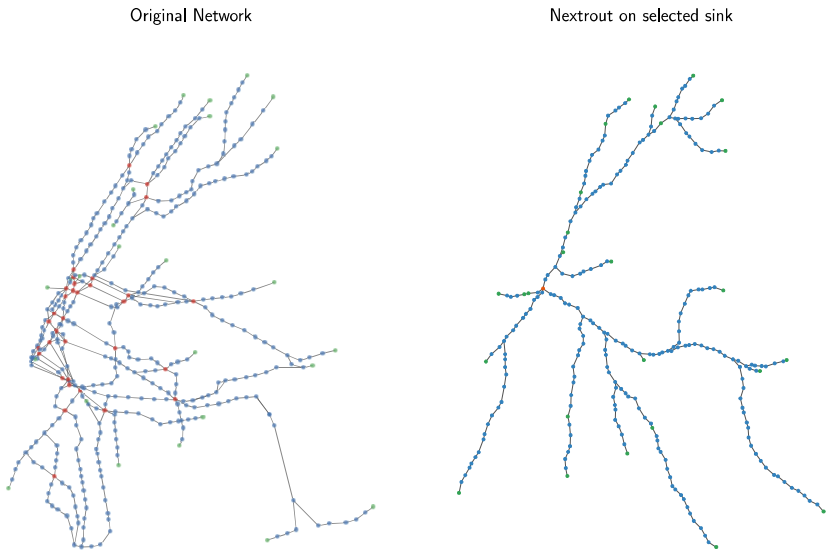


Fig. 9.8: Left: Original New York subway network highlighting nodes with lower (green) and higher degree (red). We select one node from the higher degree ones as a sink. On the right we show the simulated network obtained with $\beta = 1.9$. Notice that despite some branches cover similar areas, such as the north-left branch, the similarity along the network is not very clear, which evidences a limitation of our approach for selecting a single sink in larger structures.

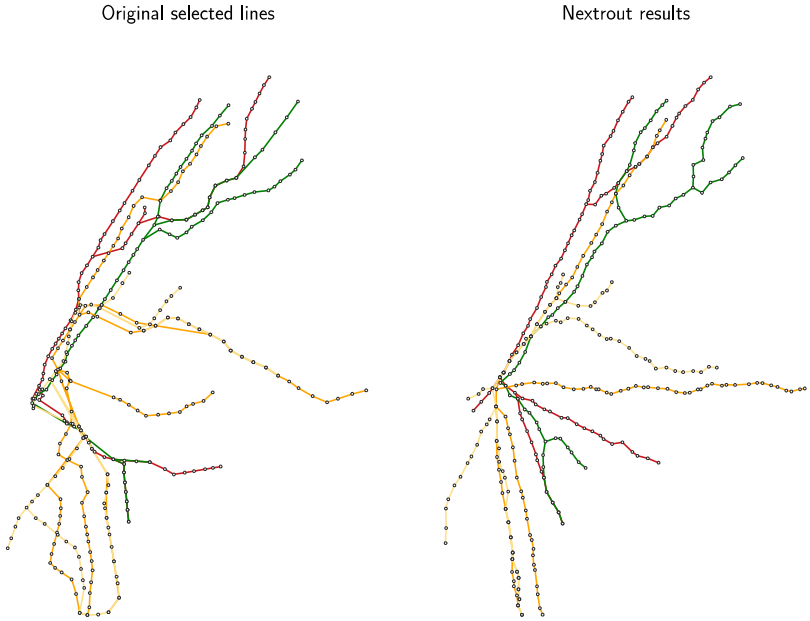


Fig. 9.9: Overlapping selected main lines from the New York subway. We have already shown in Section 5.3.2 the network properties for each individual line. We now build the union of these lines, noticing that this strategy might lead to the appearance of a few loops.

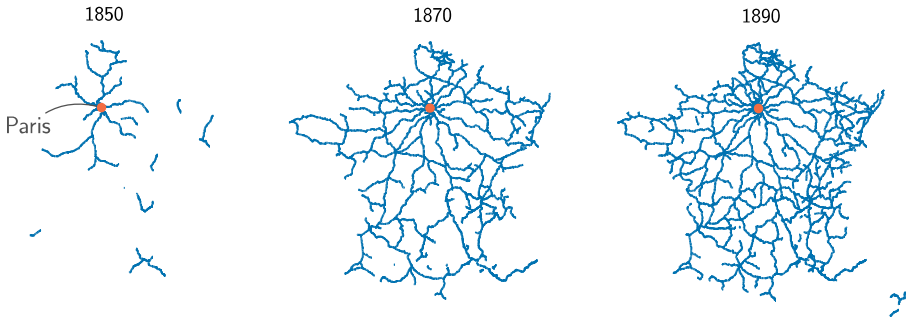


Fig. 9.10: Evolution of the French railway network over time. We show how the network topology changes in a range of 40 years along the nineteenth century, from the selected as an example of network evolution in the main text (1850), until the emergence of multiple loops around 40 years later [169].