



Relative entropy-regularized optimal transport on a graph: a new algorithm and an experimental comparison

Sylvain Courtain¹ · Guillaume Guex² · Ilkka Kivimäki³ · Marco Saerens^{4,5}

Received: 7 October 2021 / Accepted: 21 October 2022

© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract

The present work investigates a new relative entropy-regularized algorithm for solving the optimal transport on a graph problem within the randomized shortest paths formalism. More precisely, a unit flow is injected into a set of input nodes and collected from a set of output nodes with specified marginals, while minimizing the expected transportation cost, together with a paths-based relative entropy regularization term, providing a randomized routing policy. The main advantage of this new formulation is the fact that it can easily accommodate edge flow capacity constraints which commonly occur in real-world problems. The resulting optimal routing policy, i.e., the probability distribution of following an edge in each node, is Markovian and is computed after constraining the input and output flows to the prescribed marginal probabilities. In addition, experimental comparisons with other recently developed techniques show that the distance measure between nodes derived from the introduced model provides competitive results on semi-supervised classification tasks.

Keywords Link analysis · Network science · Optimal transport · Randomized shortest paths · Distances between nodes

1 Introduction

1.1 General introduction

This work studies a flow-constrained randomized shortest paths [1–3] formulation to the relative entropy-regularized

optimal transport problem on a directed graph with multiple inputs and outputs having fixed marginal probabilities (or margins, providing fixed unit input and output flows), studied in [4,5]. This last work extended the relative entropy-regularized optimal transport problem (see the recent work [6], but also [7–9]) to a graph structure. Intuitively, it aims to carry goods from some input nodes to output nodes with least expected cost while maintaining a prescribed level of relative entropy of the path probabilities connecting inputs to outputs. In this problem, the input flows (proportion of goods carried from input nodes) and output flows (proportion of goods carried to output nodes) are constrained to be equal to some predefined values (marginal probabilities), which are not present in the standard **randomized shortest paths** (RSP, [1–3]) and **bag-of-paths** (BoP, [10,11]) models. The introduced model will therefore be called the **margin-constrained bag-of-paths** model along the paper, in order to remain consistent with [5].

The introduced algorithm solving this problem provides an optimal randomized policy balancing exploitation and exploration through a simple iterative algorithm inspired by [12]. This resulting policy takes the form of a biased Markov chain guiding the agent from input nodes to output nodes with minimal expected cost for a given entropy

✉ Sylvain Courtain
sylvain.courtain@uclouvain.be

Guillaume Guex
guillaume.guex@unil.ch

Ilkka Kivimäki
ilkka.s.kivimaki@gmail.com

Marco Saerens
marco.saerens@uclouvain.be

¹ LouRIM, Université catholique de Louvain, Place de l'Université 1, 1348 Ottignies-Louvain-la-Neuve, Belgium

² SLI, Université de Lausanne, Unicentre, 1015 Lausanne, Switzerland

³ Department of Computer Science, Aalto University, Otakaari 24, 00076 Aalto, Finland

⁴ ICTEAM, Université catholique de Louvain, Place de l'Université 1, 1348 Ottignies-Louvain-la-Neuve, Belgium

⁵ IRIDIA, Université Libre de Bruxelles, Avenue Franklin Roosevelt 50, 1050 Bruxelles, Belgium

level, while satisfying the flow constraints on the input and output nodes.

Similarly to the standard RSP and BoP frameworks,¹ the model is monitored by a parameter θ in such a way that, when θ goes to infinity, it approximates the optimal, lowest-cost, solution to the transportation problem. Conversely, when θ is close to zero, the solution approaches a random walk behavior (provided a priori by the user) in terms of relative entropy (also called Kullback–Leibler divergence). Thus, when varying θ , the model interpolates between an optimal (exploitation) and a random (exploration) behavior.

The main idea, in comparison with the previous work (mainly [4,5]), is the following. The relative entropy-regularized optimal transport on a graph problem studied in [5] is based on a BoP formulation where the set of all paths between the source and target nodes is considered. On the contrary, in the present work, the problem is rephrased within a RSP formalism ([2]; inspired by [13]) only considering the set of paths from one single source supernode and one target supernode, both added to the original network.

This rephrasing into a source-target RSP problem allows us to easily define capacity constraints on edge flows, as presented in [12]. Flow capacity constraints are relevant in many graph-based applications; for instance, in the case of traffic or passenger volume constraints on transportation networks [14]. This other, previous, work [12] allows us to reformulate the margin-constrained BoP on a graph problem into a capacity-constrained RSP problem which has its own merits. As an application, the margin-constrained bag-of-paths surprisal distance measure between pairs of nodes [5] is derived within the new formalism.

To summarize, because the problem is transformed into a RSP single source-single target framework, it suffices to follow the procedure introduced in [12] for dealing with both the margin and the capacity constraints. The new algorithm therefore solves the relative entropy-regularized minimum expected cost flow with capacity constraints problem on a graph. It is, however, slower than the dedicated algorithm developed in [5] so that it should only be used when capacity constraints are indeed present.

The introduced algorithm could therefore be used to solve standard minimum cost flow problems [14,15] which regroup a wide variety of real-world problems such as 3D reconstruction [16], image segmentation [17], scheduling of tasks

[18], transportation problems [19,20] and so forth. Moreover, transportation problems are also subdivided into a large number of sub-problems which are categorized by transportation mode such as military tanker fleet scheduling problem [21], air traffic flow management problem [22], truck loading problem [23] and many others (see [19,20] for details and the references in these papers). See also the recent work on entropy-regularized optimal transport and its applications reported in [24].

In addition to the introduction of this new algorithm, we also conducted an extensive experimental comparison between the newly introduced model and other state-of-the-art models on graph-based semi-supervised classification problems. This experimental comparison was left for future work in [5]. The results of the comparison suggest that the distance measure derived from the margin-constrained BoP model is clearly competitive with respect to the other investigated distance measures.

In order to avoid redundancy, the present paper will focus on the derivation of the new model and the experimental comparison. For a more comprehensive discussion and related work concerning the margin-constrained BoP problem on a graph, as well as some illustrative examples, see [4,5].

1.2 Related work

As already mentioned, the present work is a follow-up of [4,5] which investigated the entropy-regularized optimal transport on a graph problem, respectively from a flow model and a path-based RSP perspective—see this last work and references therein for information about the RSP.

Moreover, as discussed in [5], the optimal transport on a graph has been studied extensively in the operations research field, where it is known as the minimum cost flow in a network or the minimum cost circulation problem (see, e.g., [14,25,26]). It admits a linear programming formulation, as will be shown later in this paper (see Eq. (10)), but several alternative, including more efficient, algorithms have been proposed in the literature. See the above references as well as the related work in [5] as a starting point.

The entropy-regularized extension of this problem on a bipartite graph has already been studied in the field of spatial geography and economics [7,9,27] where it is known as the gravity model; see also [28] for a recent account. It corresponds to a regularized optimal matching problem on this bipartite graph with fixed marginals on the nodes. This formulation was revisited and extended in the machine learning field by Cuturi [6], who showed that using Sinkhorn's matrix scaling algorithm is more efficient than standard linear programming solvers. Cuturi also observed that the derived

¹ The main difference between the BoP and the RSP formalism is that, for the BoP, all possible paths in the network are considered [10,11], whereas only source-target paths connecting two nodes of interest are considered in the RSP [1–3]. The RSP therefore avoids the need for defining prior distributions on source and on target nodes because there is only one single source and target.

entropy-regularized distance improves upon classic optimal transport distances on supervised classification tasks. A large amount of subsequent related work was published, providing more in-depth analysis and extensions; see for instance [24].

The entropy-regularized optimal transport on a general directed graph, which is the main focus of this work, was also studied in [29,30], where the authors develop optimal transportation policies along bounded paths of finite length. Interestingly, they show that this problem is closely related to the discrete counterpart of the Schrödinger bridge problem (see, e.g., [31,32]). The optimal randomized policy, which is time-dependent in general, is derived. In parallel to this work, [4,5] investigated transportation models on a graph considering all possible paths of unbounded length, leading to a Markovian policy.

In addition, and very recently, [33] proposed an algorithm solving the same entropy-regularized optimal transport problem on a graph while, in addition, considering capacity constraints on the edges and storage capacities on the nodes. The developed procedure uses Dykstra's iterative projection algorithm to solve the problem. Notice that this last work is therefore closely related to the work presented in this paper, the differences being that (1) we do not consider storage capacities on the nodes, (2) our algorithm repeatedly uses the RSP framework solving a system of linear equations for computing the optimal policy, and (3) the obtained policy is of the form of a Markov chain, instead of edge flows.

1.3 Main contributions and content

The main contributions of this work are

- The introduction of a new algorithm for solving the relative entropy-regularized optimal transport (in terms of minimum expected cost flow) on a graph problem by considering edge flow constraints.
- An experimental comparison between the distance measure derived from the studied model and other state-of-the-art distance measures between nodes on a graph on semi-supervised classification problems.

The next section presents the notation and some preliminaries. Section 3 develops the new RSP-based algorithm relying on flow constraints. Then, Sect. 4 introduces how this algorithm can be used to solve transportation problems. Section 5 details the experimental comparison and its results. Finally, Sect. 6 presents the conclusions of the work.

2 Notation, problem statement, and preliminaries

2.1 Notation

Assume a directed, strongly connected and weighted, graph $G = \{\mathcal{V}, \mathcal{E}\}$ containing $(n - 2)$ nodes,² in which we have to carry goods from a predefined set of input nodes $\mathcal{In}$ to a set of output nodes $\mathcal{Out}$. The user specifies the proportions of non-negative, continuous, **flow** σ_i^{in} coming from each input $i \in \mathcal{In}$ as well as the proportion of flow delivered to each output σ_j^{out} , $j \in \mathcal{Out}$. For all other nodes $i \notin \mathcal{In}$, we set $\sigma_i^{\text{in}} = 0$. Symmetrically, for all nodes $j \notin \mathcal{Out}$, $\sigma_j^{\text{out}} = 0$. In practice, we almost always consider that input nodes are different from output nodes, $\mathcal{In} \cap \mathcal{Out} = \emptyset$, but this assumption is not strictly needed in the model. Let us further assume

$$\begin{cases} \sum_{i \in \mathcal{In}} \sigma_i^{\text{in}} = 1 & \text{with all } \sigma_i^{\text{in}} \geq 0 \\ \sum_{j \in \mathcal{Out}} \sigma_j^{\text{out}} = 1 & \text{with all } \sigma_j^{\text{out}} \geq 0 \end{cases} \quad (1)$$

In other words, a unit flow is considered. If we have to transport a non-unitary flow, we simply find the solution for a unit flow and then multiply the quantities by the total flow. The $n \times 1$ column vector containing the input flow of each node is denoted by σ_{in} whereas the vector containing the output flows is σ_{out} .

Moreover, it is assumed that a non-negative **local cost** c_{ij} is associated with each edge (i, j) , reflecting the penalty of following this edge (it can be a distance, a cost, a travel time, an energy consumption, etc). The **cost matrix** $\mathbf{C}$ contains the individual costs c_{ij} as elements. When there is no edge from node i to node j we consider the cost c_{ij} to be infinite. The graph G is also associated with an **adjacency matrix** $\mathbf{A}$ containing local affinities, or weights, $\{a_{ij}\}$, between nodes. When there is no direct edge between two nodes i, j , $a_{ij} = 0$. Matrices $\mathbf{C}$ and $\mathbf{A}$ are given and depend on the problem at hand. The **transition probability matrix** associated with the natural random walk on this graph G is $\mathbf{P}$, with elements

$$[\mathbf{P}]_{ij} = p_{ij} = \frac{a_{ij}}{\sum_{j' \in \text{Succ}(i)} a_{ij'}} = \frac{a_{ij}}{\sum_{j'=1}^n a_{ij'}} \quad (2)$$

where $\text{Succ}(i)$ is the set of successor nodes of node i . The third equality is valid because the elements on the i th row of the adjacency matrix are equal to 0 for the missing edges $j \in \mathcal{E} \setminus \text{Succ}(i)$.

² See later for the justification; an extended graph with two additional nodes, and thus n nodes in total, will be defined later in Sect. 2.3.

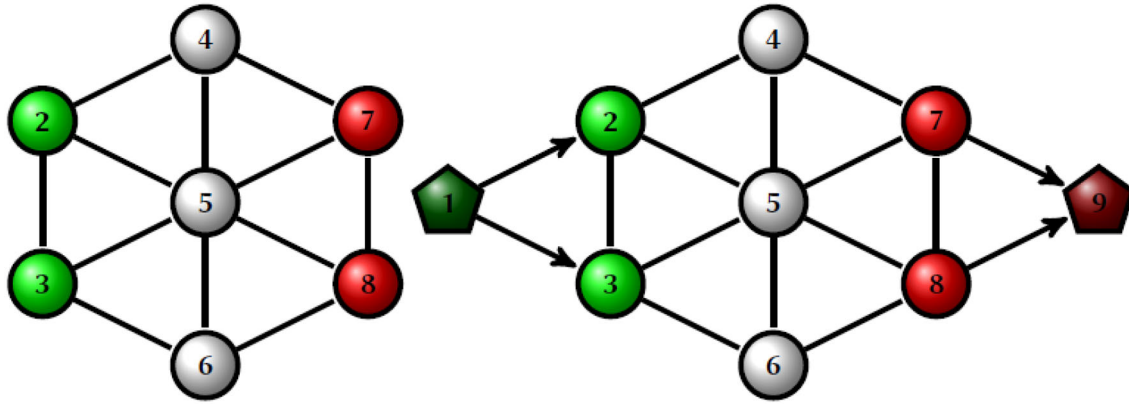


Fig. 1 On the left, a small undirected graph G with two input nodes $\mathcal{In} = \{2, 3\}$ (in green) and two output nodes $\mathcal{Out} = \{7, 8\}$ (in red). On the right, the extended graph G_{ext} of this small undirected graph G with one source supernode 1 (in dark green) connected to all the input

nodes and one target supernode $n = 9$ (in dark red) connected to all the output nodes. Therefore, on this extended graph, the source supernode is node 1 and the target supernode is node n (the total number of nodes in G_{ext}). The figures are generated with Tikz

2.2 Problem statement

Given this context, the problem is to find the best policy, which takes the form of path probabilities, for carrying the input flow from the source nodes to the target nodes, minimizing the expected cost along the paths connecting sources to targets, while

1. keeping a given level of exploration quantified by the Kullback–Leibler divergence between path probabilities and completely random paths provided by the natural random walk (Eq. (2)), and
2. satisfying the flow constraints stating that input flows and output flows are fixed to σ_i^{in} for each node $i \in \mathcal{In}$ and σ_j^{out} for each node $j \in \mathcal{Out}$.

This problem will be solved in Sect. 3; before, let us introduce an extended graph G_{ext} by adding two supernodes to G .

2.3 Definition of an extended, single-source and target, graph

For convenience, we now transform the original graph G into a new, equivalent, single-source single-target graph $G_{\text{ext}} = \{\mathcal{V}_{\text{ext}}, \mathcal{E}_{\text{ext}}\}$ with n nodes in a standard way [14,25].³ Two nodes—one **source node** (a supernode indexed as node 1) and one **target node** (an absorbing supernode (no outgoing edge) indexed as node n , where n is the total number of nodes in this new, extended, graph) are added to the graph. All the other nodes in $\mathcal{V}$ (nodes 2 to $(n - 1)$) and their connections $\mathcal{E}$ remain the same as in G (G is a subgraph of G_{ext}); therefore, $\mathcal{V}_{\text{ext}} = \mathcal{V} \cup \{1, n\}$. All nodes with label $i \in \mathcal{V}$ keep the same index numbering in $\mathcal{V}_{\text{ext}}$. Moreover, the new source

node (node 1) is only connected by directed edges to the input nodes with a zero cost, and the output nodes are only connected to the new target node (node n) by a directed edge, also with zero cost (no penalty when following these edges). In order to be equivalent to the original graph G , the source node generates a unit flow while the target node is made killing and absorbing (a cemetery or sink node), and thus absorbs this unit flow. This new graph will be called the **extended graph**. Illustrative examples of this concept are presented in Fig. 1 and Sect. 4.

2.3.1 The adjacency matrix of the extended graph

We might now ask ourselves which weights (values of corresponding elements of the adjacency matrix) should be associated with the new edges incident to node 1 and node n . First of all, as explained at the beginning of this subsection, node 1 is only connected by directed edges to the input nodes and the output nodes are only connected to node n by a directed edge. Therefore, the weights associated with the directed edges between the input nodes and node 1 and between node n and the output nodes are zero because these edges do not exist.

Before defining the weights associated with the existing edges incident to node 1 and node n , let us first recall two fundamental Markov chain quantities defined on G_{ext} . These quantities are the expected number of visits to node $i \in \mathcal{V}$, denoted as $\bar{n}_i$, and the expected number of passages (also called the flow) through edge (i, j) , $\bar{n}_{ij}$. Moreover, note that, by conservation of flows, we have $\bar{n}_j = \sum_{i \in \text{Pred}(j)} \bar{n}_{ij}$ and $\bar{n}_{ij} = \bar{n}_i p_{ij}^{\text{ext}}$ for any $j \in \mathcal{V}$.

Now, regarding the directed edges coming out of node 1, a natural requirement would be that the weights of these edges are proportional to the input flows, in such a way that the corresponding transition probabilities are exactly equal

³ This kind of trick is common in network flow algorithms design.

to these input flows σ_{in} . Indeed, because node 1 generates a unit flow, $\bar{n}_1 = 1$, it is necessary that the transition probabilities $p_{1i}^{ext} = \sigma_i^{in}$ to ensure that $\bar{n}_{1i} = \bar{n}_1 p_{1i}^{ext} = \sigma_i^{in}$ by conservation of flows for any $i \in \mathcal{I}n$.

For the directed edges incident to node n , the choice of the weights is slightly more difficult. These weights, denoted as $\mathbf{w}$, can either be provided explicitly by the user or computed in such a way that the corresponding transition probabilities guarantee, by conservation of flows, that $\bar{n}_{jn} = \bar{n}_j p_{jn}^{ext} = \sigma_j^{out}$ for any $j \in \mathcal{O}ut$. In other words, the $\mathbf{w}$ should be chosen in order to obtain σ_{out} as output distribution of the flows. However, compared to the previous situation, the flow of each node $j \in \mathcal{O}ut$, $\bar{n}_j$, is not equal to 1, and thus needs to be computed, which slightly complicates the determination of $\mathbf{w}$. The choice of $\mathbf{w}$ is further below.

Therefore, the adjacency matrix of the extended graph, $\mathbf{A}_{ext}$, can be written as

$$\mathbf{A}_{ext} = \begin{matrix} & \begin{matrix} 1 & \{2, \dots, (n-1)\} = \mathcal{V} & n \end{matrix} \\ \begin{matrix} 1 \\ \{2, \dots, (n-1)\} = \mathcal{V} \\ n \end{matrix} & \begin{bmatrix} 0 & \sigma_{in}^T & 0 \\ \mathbf{0} & \mathbf{A} & \mathbf{w} \\ 0 & \mathbf{0}^T & 0 \end{bmatrix} \end{matrix} \quad (3)$$

where $\mathbf{0}$ is the null column vector full of 0s. Similarly, the cost matrix becomes

$$\mathbf{C}_{ext} = \begin{matrix} & \begin{matrix} 1 & \{2, \dots, (n-1)\} = \mathcal{V} & n \end{matrix} \\ \begin{matrix} 1 \\ \{2, \dots, (n-1)\} = \mathcal{V} \\ n \end{matrix} & \begin{bmatrix} 0 & \mathbf{0}^T & 0 \\ \mathbf{0} & \mathbf{C} & \mathbf{0} \\ 0 & \mathbf{0}^T & 0 \end{bmatrix} \end{matrix} \quad (4)$$

If the weights $\mathbf{w}$ are set by the user, the transition matrix $\mathbf{P}_{ext} = (p_{ij}^{ext})$ of the natural random walk on this extended graph G_{ext} can easily be computed by Eq. (2) from $\mathbf{A}_{ext}$ instead of $\mathbf{A}$. In that case, our model will compute a policy (transition probabilities followed for carrying the goods) interpolating between the optimal expected lowest-cost policy and the one closest (in terms of Kullback–Leibler divergence) to the natural random walk. But it will in general, when $\theta \rightarrow 0$, *not be exactly equal* to this natural random walk transition matrix because of the flow constraints in Eq. (1) which are *not satisfied in general* for edges entering node n in the extended graph G_{ext} .

It is, however, possible to find sets of values of the weights $\mathbf{w}$ such that the net flow in each of these edges is exactly equal to σ_j^{out} when considering a natural random walk on the extended graph [4]. Indeed, in Appendix A, the transition matrix on the extended graph leading to flows satisfying exactly the constraints in Eq. (1) for $j \in \mathcal{O}ut$ is computed in closed form by stating a simple consistency argument (largely inspired by [4,5]),

$$\mathbf{P}_{ext} = \begin{matrix} & \begin{matrix} 1 & \{2, \dots, (n-1)\} = \mathcal{V} & n \end{matrix} \\ \begin{matrix} 1 \\ \{2, \dots, (n-1)\} = \mathcal{V} \\ n \end{matrix} & \begin{bmatrix} 0 & \sigma_{in}^T & 0 \\ \mathbf{0} & (\mathbf{I} - \text{Diag}(\alpha))\mathbf{P} & \alpha \\ 0 & \mathbf{0}^T & 0 \end{bmatrix} \end{matrix} \quad (5)$$

where the quantity α is computed from Eqs. (A7)–(A9) (see Appendix A). Interestingly, for nodes not belonging to $\mathcal{I}n$, these transition probabilities remain exactly the same as before, as long as the random walker is not absorbed. The vector $\mathbf{w}$ is then obtained from α through Eq. (A3). Notice that only the output nodes $j \in \mathcal{O}ut$ can possibly have a positive weight w_j (and thus also a positive α_j); the weight of all other nodes is set to 0. If we do not have a good reason for choosing the weights $\mathbf{w}$, it seems reasonable to compute the consistent transition probabilities because it removes the arbitrariness associated with the choice of these weights.

3 Optimal transport on a graph in a constrained randomized shortest paths framework

Our formulation of the problem is based on the RSP framework defining dissimilarity measures interpolating between the shortest-path distance and the commute-cost distance [1–3]. This formalism is based on full paths instead of standard “local” flows [14], and was initially inspired by a model developed in transportation science [13].

For completeness, we therefore start with a brief description (closely following [2,12,34]) of the RSP formalism before defining the problem and then deriving the algorithm for solving the regularized multi-inputs multi-outputs transport problem on the graph G_{ext} .

3.1 The standard randomized shortest paths formalism

Let us start with a short reminder about the RSP model. For the sake of simplicity, in this section, all quantities are discussed in the context of a graph G with a single source node 1 and a single target node n (an absorbing node) in order to avoid more cumbersome notations.

In its basic form, the RSP provides a model of movement on a graph from a source node to a target node interpolating between an optimal and a random behavior. The optimal behavior corresponds to the set of shortest paths whereas the random behavior is provided by the natural random walk on the graph with transition probabilities given by Eq. (2).

In this context, the movement between source node to target node is represented by trajectories, or paths, connecting these two nodes of interest. The set of all these paths is denoted as $\mathcal{P}_{1n}$ and one particular path as $\wp \in \mathcal{P}_{1n}$. Thus, $\mathcal{P}_{1n}$ can be considered as a “bag of paths” containing all

paths from node 1 to n . Each of these paths has a cost provided by the total **cumulated cost** c_{ij} along the path,⁴ $\tilde{c}(\varphi)$, considering costs as additive. In addition, the **random walk likelihood** of the path is the product of the p_{ij} along the path and denoted as $\tilde{\pi}(\varphi)$.

Next, a probability mass is independently assigned on this set of paths, providing the probability of choosing a path for moving from node 1 to node n within the bag of paths. This is done by solving the following optimisation problem⁵

$$\begin{cases} \text{minimize} & \phi(\mathbf{P}) = \sum_{\varphi \in \mathcal{P}_{1n}} \mathbf{P}(\varphi) \tilde{c}(\varphi) + T \sum_{\varphi \in \mathcal{P}_{1n}} \mathbf{P}(\varphi) \log \left(\frac{\mathbf{P}(\varphi)}{\tilde{\pi}(\varphi)} \right) \\ \text{subject to} & \sum_{\varphi \in \mathcal{P}_{1n}} \mathbf{P}(\varphi) = 1 \end{cases} \quad (6)$$

which is nothing else than the minimization of the free energy potential, widely used in statistical physics [35–37], of the whole system.

This objective function aims at minimizing the expected cost of moving from 1 to n (first term) while at the same time not deviating too much from a pure random walk behavior (second term). Indeed, the second term is the relative entropy, or Kullback–Leibler divergence, between the path probability distribution and the random walk path likelihood distribution (introducing randomness). The two terms are balanced by a temperature parameter T monitoring the degree of randomness of the trajectories. When the temperature T is low, shorter paths are favored but, when T is large, paths are mainly chosen according to their likelihood in a natural random walk on the graph G . Interestingly this model therefore interpolates between an optimal and a purely random behavior.

After defining $\theta = 1/T$ (the inverse temperature parameter), the minimization of the objective function leads to a Gibbs–Boltzmann distribution on the set of paths (see, e.g., [10] for a detailed derivation),

$$\begin{aligned} \mathbf{P}^*(\varphi) &= \frac{\tilde{\pi}(\varphi) \exp[-\theta \tilde{c}(\varphi)]}{\sum_{\varphi' \in \mathcal{P}_{1n}} \tilde{\pi}(\varphi') \exp[-\theta \tilde{c}(\varphi')]} \\ &= \frac{\tilde{\pi}(\varphi) \exp[-\theta \tilde{c}(\varphi)]}{\mathcal{Z}} \end{aligned} \quad (7)$$

and the denominator $\mathcal{Z} = \sum_{\varphi \in \mathcal{P}_{1n}} \tilde{\pi}(\varphi) \exp[-\theta \tilde{c}(\varphi)]$ is usually called the **partition function** of the system. One of

the key points is that this set of path probabilities (the optimal randomized policy of choosing a trajectory in terms of full paths) is exactly generated by, and thus equivalent to, a **Markov chain** with biased transition probabilities (the optimal randomized policy, but now in terms of local transitions in G) favoring shorter paths, depending on the temperature T (see Eq. (B19) in the Appendix B, or [2]). This is called the optimal local policy, defining a transition probability matrix $\mathbf{P}^*$ guiding the random walker towards target node n from any node of the graph.

The main objective of this work is to compute such an optimal local policy, but considering now additional margin constraints in a multiple-sources and multiple-targets setting (an optimal transport problem on G), instead of a single source-target setting for the standard RSP.

The Appendix B provides a brief summary of the most important quantities that can be derived from the standard RSP model. Some of these quantities are also reported in Table 1 for convenience. As an example, an important quantity that will be needed in the algorithm is the **fundamental matrix** $\mathbf{Z}$ which is computed thanks to Eq. (B12).

3.2 Statement of the problem

The objective now is to compute the randomized shortest paths solution on G_{ext} satisfying the source/target flow constraints⁶ stated in Sect. 2, and recalled here for convenience,

$$\begin{cases} \bar{n}_{1i} = \sigma_i^{\text{in}} & \text{for each node } i \in \mathcal{I}n \\ \bar{n}_{jn} = \sigma_j^{\text{out}} & \text{for each node } j \in \mathcal{O}ut \end{cases} \quad (8)$$

where $\bar{n}_{ij}$ is the flow (expected number of passages) in edge (i, j) (see Eq. (B16)).

As already mentioned in the introduction, the margin-constrained BoP problem on a graph has recently been studied in [4] and [5]. The former paper is based on an entropy regularization at the flow level while the later one adopts a bag-of-paths approach. Considering now multiple input and output nodes, we want to solve the problem⁷

$$\begin{cases} \text{minimize} & \phi(\mathbf{P}) = \sum_{\varphi \in \mathcal{P}} \mathbf{P}(\varphi) \tilde{c}(\varphi) + T \sum_{\varphi \in \mathcal{P}} \mathbf{P}(\varphi) \log \left(\frac{\mathbf{P}(\varphi)}{\tilde{\pi}(\varphi)} \right) \\ \text{subject to} & \sum_{j \in \mathcal{V}} \sum_{\varphi_{ij} \in \mathcal{P}_{ij}} \mathbf{P}(\varphi_{ij}) = \sigma_i^{\text{in}} & \forall i \in \mathcal{I}n \\ & \sum_{i \in \mathcal{V}} \sum_{\varphi_{ij} \in \mathcal{P}_{ij}} \mathbf{P}(\varphi_{ij}) = \sigma_j^{\text{out}} & \forall j \in \mathcal{O}ut \end{cases} \quad (9)$$

where the set of considered paths is $\mathcal{P} = \cup_{i \in \mathcal{I}n} \cup_{j \in \mathcal{O}ut} \mathcal{P}_{ij}$. It was shown [5] that the solution can be obtained by iterative

⁴ The basic quantities that are defined on whole paths φ will be denoted with a tilde in order to distinguish them from the same local quantities defined on edges.

⁵ Note that we should normally add non-negativity constraints but this is not necessary as the resulting probabilities will automatically be non-negative.

⁶ Required input flow σ_i^{in} is set to 0 for nodes i not in $\mathcal{I}n$ and output flow σ_j^{out} is set to 0 for nodes j not in $\mathcal{O}ut$.

⁷ Observe the difference with the standard RSP formulation of Eq. (6).

Table 1 Summary of notation for the RSP transport model on a graph

Notation	Meaning
$\mathcal{V}, \mathcal{V}_{\text{ext}}$	Set of vertices, or nodes, of graph G and extended graph G_{ext}
$\mathcal{E}, \mathcal{E}_{\text{ext}}$	Set of edges of graph G and extended graph G_{ext}
$\text{Succ}(i), \text{Pred}(i)$	Set of successor and predecessor nodes of node i
$\mathbf{A} = (a_{ij})$	Adjacency matrix of graph G
$\mathbf{P} = (p_{ij})$	Transition probability matrix of the natural random walk on graph G
$\mathbf{C} = (c_{ij})$	Cost matrix of graph G
$\mathbf{A}_{\text{ext}} = (a_{ij}^{\text{ext}})$	Adjacency matrix of the extended graph G_{ext}
$\mathbf{P}_{\text{ext}} = (p_{ij}^{\text{ext}})$	Transition probability matrix of the extended graph G_{ext}
$\mathbf{C}_{\text{ext}} = (c_{ij}^{\text{ext}})$	Cost matrix of the extended graph G_{ext}
$\sigma_{\text{in}} = (\sigma_i^{\text{in}})$	Input flows column vector
$\sigma_{\text{out}} = (\sigma_i^{\text{out}})$	Output flows column vector
$\bar{n}_i, \bar{n}_{ij}$	Expected number of visits to node i and edge (i, j)
$\mathcal{P}_{ij}$	Set of all paths connecting node i to node j (a bag of paths)
$\wp \in \mathcal{P}_{1n}$	A particular path $\wp$ starting in node 1 and ending in node n
$P(\wp)$	The probability of choosing path $\wp$ from the bag of paths
$\tilde{c}(\wp), \tilde{\pi}(\wp)$	Cumulated cost and likelihood (natural, reference, probability) along path $\wp$
$\ell(\wp), \wp(\tau)$	Length of path $\wp$ and node appearing at position τ on path $\wp$
$\mathbf{W} = (w_{ij})$	The auxiliary matrix containing elements $w_{ij} = p_{ij} \exp[-\theta c_{ij}]$
$\mathbf{Z} = (z_{ij}) = (\mathbf{I} - \mathbf{W})^{-1}$	The fundamental matrix of the randomized shortest paths system
ϕ_{ij}	The free energy of the system of paths connecting node i to node j
$\mathbf{\Gamma} = (\gamma_{ij})$	The coupling matrix

proportional fitting (also called matrix balancing or biproportional scaling), as for the standard, relaxed, optimal transport problem with entropy regularization (see for instance [6–9]).

In the present paper, we adopt a different point of view (in comparison with [5]) for solving this problem by designing a new algorithm based on the constraints imposed on flows in the graph G_{ext} (see Eq. (8)), inspired by [12]. It is important to note that this formulation of the problem can readily integrate additional capacity constraints as well; see [12] for details. This is what makes the present formulation interesting for practical problems: it extends the scope of the optimal transport on a graph procedure introduced in [4,5] and solving Eq. (9) to problems with capacity constraints.

Notice that when the temperature becomes close to 0 (no entropy regularization, and thus no randomness), we recover the optimal transport on a graph problem, also called the network minimum cost flow problem in the literature, which can be solved by linear programming [14,25]. Indeed, denoting the non-negative edge flows matrix to be found by $\mathbf{X} = (x_{ij})$, the problem becomes

$$\begin{aligned} & \underset{\mathbf{X}}{\text{minimize}} \quad \mathbf{e}^T (\mathbf{X} \circ \mathbf{C}) \mathbf{e} \\ & \text{subject to} \quad (\mathbf{X} - \mathbf{X}^T) \mathbf{e} = \sigma_{\text{in}} - \sigma_{\text{out}} \\ & \quad \quad \quad x_{ij} \geq 0 \quad \forall i, j \in \mathcal{V} \end{aligned} \quad (10)$$

where $\mathbf{e}$ is a column vector of 1's and $\mathbf{C}$ is the cost matrix containing non-negative costs on the edges, with infinite values set to 0 (missing edges). This equation says that we have to minimize the total cost of flows while satisfying the input and output constraints. The solution $\mathbf{X}^*$ of this problem corresponds to the matrix containing directed flows on the edges [5].

3.3 The margin-constrained randomized shortest path algorithm

The algorithm solving the relative entropy-regularized optimal transport on a graph problem is derived from results obtained in [12] by exploiting its Lagrange formulation and Lagrangian duality. It provides the optimal **randomized policy** taking the form of the transition matrix of a **biased random walk** on G_{ext} , as in the case of the standard RSP problem (see Eq. (B19)). It corresponds to a Markov chain on the extended graph biasing the random walk towards the output nodes while satisfying the input and output flow constraints of Eq. (8). Note that, for convenience, most of the more technical results are reported in the Appendix C.

3.3.1 The Lagrange function

The equality constraints (Eq. (8)) can be expressed in the following Lagrange function defined on the extended graph G_{ext} with $\mathcal{P}_{1n}$ being the set of all possible paths from source node 1 to target node n ,

$$\begin{aligned} \mathcal{L}(\mathbf{P}, \boldsymbol{\lambda}) = & \underbrace{\sum_{\wp \in \mathcal{P}_{1n}} \mathbf{P}(\wp) \tilde{c}(\wp) + T \sum_{\wp \in \mathcal{P}_{1n}} \mathbf{P}(\wp) \log \left(\frac{\mathbf{P}(\wp)}{\tilde{\pi}(\wp)} \right)}_{\text{free energy, } \phi(\mathbf{P})} \\ & + \mu \left(\sum_{\wp \in \mathcal{P}_{1n}} \mathbf{P}(\wp) - 1 \right) + \sum_{i \in \mathcal{I}n} \lambda_i^{\text{in}} (\bar{n}_{1i} - \sigma_i^{\text{in}}) \\ & + \sum_{j \in \mathcal{O}ut} \lambda_j^{\text{out}} (\bar{n}_{jn} - \sigma_j^{\text{out}}) \end{aligned} \quad (11)$$

where vector $\boldsymbol{\lambda}$ contains the Lagrange parameters $\{\lambda_i^{\text{in}}\}$ and $\{\lambda_j^{\text{out}}\}$. There is a Lagrange parameter associated with each input node ($\mathcal{I}n$) and each output node ($\mathcal{O}ut$). Note that because we are working on the extended graph, $\tilde{\pi}(\wp)$ is the product of the p_{ij}^{ext} (defined in Eq. (5)) along path $\wp$.

3.3.2 Exploiting Lagrangian duality

Following the same reasoning as in [12], we will exploit the fact that, in this formulation of the problem, the Lagrange dual function and its gradient are easy to compute.⁸ Moreover, because the objective function is convex and all the equality constraints are linear, there is only one global minimum and the duality gap is zero [39–41]. We therefore use a common optimization procedure, the Arrow–Hurwicz–Uzawa algorithm [42]), which sequentially solves the primal and increases the dual (which is concave) until convergence. In our context, this provides the following steps, which are iterated until convergence,

$$\begin{cases} \mathcal{L}(\mathbf{P}^*, \boldsymbol{\lambda}) = \min_{\{\mathbf{P}(\wp)\}_{\wp \in \mathcal{P}_{1n}}} \mathcal{L}(\mathbf{P}, \boldsymbol{\lambda}) & \text{(compute the dual function)} \\ \text{subject to } \sum_{\wp \in \mathcal{P}_{1n}} \mathbf{P}(\wp) = 1 \\ \mathcal{L}(\mathbf{P}^*, \boldsymbol{\lambda}^*) = \max_{\boldsymbol{\lambda}} \mathcal{L}(\mathbf{P}^*, \boldsymbol{\lambda}) & \text{(maximize the dual function)} \end{cases} \quad (12)$$

where we set $\mathcal{L}(\mathbf{P}, \boldsymbol{\lambda}) = \mathcal{L}(\mathbf{P}^*, \boldsymbol{\lambda}^*)$ at the end of each iteration. In short, the dual function is computed analytically and then maximized through a block coordinate ascend in terms of the Lagrange parameters $\boldsymbol{\lambda}$. It is shown in the Appendix C that the dual function is

⁸ This is actually a standard result related to maximum entropy problems (see for instance [38]).

$$\mathcal{L}(\mathbf{P}^*, \boldsymbol{\lambda}) = -T \log \mathcal{Z}' - \sum_{i \in \mathcal{I}n} \lambda_i^{\text{in}} \sigma_i^{\text{in}} - \sum_{j \in \mathcal{O}ut} \lambda_j^{\text{out}} \sigma_j^{\text{out}} \quad (13)$$

where $\mathcal{Z}' = \sum_{\wp \in \mathcal{P}_{1n}} \tilde{\pi}(\wp) \exp[-\theta \tilde{c}'(\wp)]$ is the partition function (Eq. (B13)) computed from the so-called **augmented costs** c'_{ij} on G_{ext} , which depend on the Lagrange multipliers,

$$c'_{ij} = \begin{cases} c_{ij}^{\text{ext}} + \lambda_j^{\text{in}} & \text{when } i = 1 \text{ and } j \in \mathcal{I}n \\ c_{ij}^{\text{ext}} + \lambda_i^{\text{out}} & \text{when } i \in \mathcal{O}ut \text{ and } j = n \\ c_{ij}^{\text{ext}} & \text{otherwise} \end{cases} \quad (14)$$

Moreover, as further shown in the Appendix C, the maximization of the dual function provides the following Lagrange parameters updates at each iteration,

$$\begin{cases} \lambda_k^{\text{in}} = \frac{1}{\theta} \left(\log z'_{kn} - \sum_{l \in \mathcal{I}n} \sigma_l^{\text{in}} \log z'_{ln} \right) & \text{for each } k \in \mathcal{I}n \\ \lambda_l^{\text{out}} = \frac{1}{\theta} \left(\log z'_{ln} - \log \left(\frac{\sigma_l^{\text{out}}}{p_{ln}^{\text{ext}}} \right) - \sum_{k \in \mathcal{O}ut} \sigma_k^{\text{out}} \left[\log z'_{lk} - \log \left(\frac{\sigma_k^{\text{out}}}{p_{kn}^{\text{ext}}} \right) \right] \right) & \text{for each } l \in \mathcal{O}ut \end{cases} \quad (15)$$

where the z'_{kn} (element k, n of the fundamental matrix) are computed thanks to Eq. (B12) in terms of the augmented costs and the natural random walk transition probabilities (p_{kl}^{ext}) on the extended graph G_{ext} . Notice that the Lagrange parameters are undefined up to a translation. In the present case, they are shifted in order to satisfy $\sum_{k \in \mathcal{I}n} \lambda_k^{\text{in}} \sigma_k^{\text{in}} = 0$ and $\sum_{l \in \mathcal{O}ut} \lambda_l^{\text{out}} \sigma_l^{\text{out}} = 0$, but other translations could be used instead.

3.3.3 The resulting algorithm

The resulting algorithm is presented in Algorithm 1. The different steps of the procedure are the following:

1. Compute the extended graph G_{ext} (its edge costs and transition probabilities matrices) from the original graph G as described in Sect. 2.3 and Eqs. (3), (4) and (5). We now work on this extended graph.
2. Initialize the Lagrange parameters to 0.
3. Iterate the following steps until convergence, first to update the quantities associated with the source nodes, and then to update the quantities associated with the target nodes:
 - The necessary elements of the fundamental matrix are computed from the current augmented costs (Eq. (B12)) on G_{ext} .
 - The Lagrange parameters are updated (Eq. (15)).
 - The augmented costs are updated (Eq. (14)).

4. Compute the optimal policy (transition probabilities) from the augmented costs (depending on the Lagrange parameters, see Eq. (14)) on G_{ext} thanks to Eq. (B19).

The time complexity of the algorithm is dominated by the two systems of linear equations that need to be solved at each iteration. Therefore, it is of order $2k.O(n^3)$ where n is the number of nodes and k is the number of required iterations. Note that, because both algorithms are based on Lagrangian duality for dealing with flow capacity constraints, Algorithm 1 is closely related to Algorithm 2 presented in [12], page 351. It presents another way of solving the optimal transport on a graph problem already studied in [4, 5]. Let us now introduce a dissimilarity measure based on this model.

3.4 A distance measure between nodes

In this subsection, we will derive two important quantities from the margin-constrained bag-of-paths model. These quantities are the coupling matrix and the surprisal distance measure between nodes of the graph, defined from the coupling matrix. For more details concerning these two quantities, see [5].

3.4.1 The coupling matrix

Let us first remind the definition of the **coupling matrix** [43, 44], which was used in [5] to define a distance measure between the nodes of the graph (see next subsection). But, first, let us define $\ell(\varphi)$ (or simply ℓ) as the length of path φ , that is, the total number of nodes (including repeated nodes) in G_{ext} visited by path φ . In addition, $\varphi(\tau)$ refers to the node at position τ on path φ , starting at position 0 (initial node).

The coupling matrix is denoted by $\mathbf{0} = (\gamma_{ij})$ and is defined as the joint probability of starting the walk in node $i \in \mathcal{In}$ (reaching input node $S = i$ from supernode 1 at time step 1) and ending the walk in node $j \in \mathcal{Out}$ (visiting output node $T = j$ at time step $\ell(\varphi) - 1$ and then immediately transiting to supernode n) when walking according to the optimal path probabilities defined in Eqs. (7) and (B11) and using the augmented costs after convergence in order to satisfy the constraints,

$$\begin{aligned} \gamma_{ij} &\triangleq \mathbf{P}^*(S=i, T=j) = \mathbf{P}^*(\varphi(1)=i, \varphi(\ell-1)=j | \varphi(0)=1, \varphi(\ell)=n) \\ &= \frac{w'_{1i} (\sum_{\varphi' \in \mathcal{P}_{1n}} \tilde{w}'(\varphi'_{ij})) w'_{jn}}{\sum_{\varphi' \in \mathcal{P}_{1n}} \tilde{w}'(\varphi')} = \frac{w'_{1i} z'_{ij} w'_{jn}}{\sum_{\varphi' \in \mathcal{P}_{1n}} \tilde{w}'(\varphi')} \\ &= \frac{w'_{1i} z'_{ij} w'_{jn}}{\sum_{i' \in \mathcal{In}} \sum_{j' \in \mathcal{Out}} w'_{1i'} z'_{i'j'} w'_{j'n}} \quad \text{with } i \in \mathcal{In} \text{ and } j \in \mathcal{Out} \quad (16) \end{aligned}$$

where $\mathcal{P}_{ij}$ is the set of paths starting in input node $i \in \mathcal{In}$ and ending in output node $j \in \mathcal{Out}$. It also holds that $\sum_{i \in \mathcal{In}} \gamma_{ij} = \sigma_j^{\text{out}}$ and $\sum_{j \in \mathcal{Out}} \gamma_{ij} = \sigma_i^{\text{in}}$. Notably, the

Algorithm 1 Solving the relative entropy-regularized optimal transport on a graph problem with multiple sources and targets, called the margin-constrained bag-of-paths model.

Require:

- A weighted directed graph G_{ext} containing n nodes, derived from a strongly connected graph G as detailed in Sect. 2.3. Node 1 is the source supernode and node n the absorbing, target, supernode. The indegree of node 1 and the outdegree of node n are both equal to 0.
- The set of input nodes $\mathcal{In}$ (only connected to the source supernode 1) and output nodes $\mathcal{Out}$ (only connected to the target supernode n).
- The $n \times n$ transition matrix $\mathbf{R}_{\text{ext}}$ associated with G_{ext} . See Eq. (5) and Algorithm 1 in [5] for the computation of $\mathbf{R}_{\text{ext}}$ and α .
- The $n \times n$ non-negative cost matrix $\mathbf{C}_{\text{ext}}$ associated with G_{ext} (see Eq. (4)). These original costs are equal to zero for edges starting in node 1 and ending in $\mathcal{In}$ as well as edges starting in $\mathcal{Out}$ and ending in node n .
- The $n \times 1$ vectors of input flows, σ_{in} , and output flows, σ_{out} , both non-negative and summing to 1.
- The inverse temperature parameter θ .

Ensure:

- The $n \times n$ randomized policy defined by the transition matrix $\mathbf{P}^*$.

```

1:  $\lambda_{\text{in}} \leftarrow \mathbf{0}; \lambda_{\text{out}} \leftarrow \mathbf{0}$  ▷ initialize  $n \times 1$  Lagrange parameter vectors
2:  $\mathbf{C}' \leftarrow \mathbf{C}_{\text{ext}}$  ▷ initialize the augmented costs matrix
3: repeat [main iteration loop]
4:    $\mathbf{W}' \leftarrow \mathbf{R}_{\text{ext}} \circ \exp[-\theta \mathbf{C}']$  ▷ compute  $\mathbf{W}'$  matrix (elementwise exponential and multiplication  $\circ$ )
5:   Solve  $(\mathbf{I} - \mathbf{W}') \mathbf{z}'_n = \mathbf{e}_n$  ▷ backward variables (column  $n$  of the fundamental matrix  $\mathbf{Z}'$ ) with elements  $z'_{kn}$  ( $n$  is fixed)
6:   for all  $k \in \mathcal{In}$  do ▷ initialize Lagrange parameters associated with source nodes
7:      $\lambda_k^{\text{in}} \leftarrow \frac{1}{\theta} \log z'_{kn}$  ▷ compute Lagrange parameters
8:   end for
9:   for all  $k \in \mathcal{In}$  do ▷ update Lagrange parameters and costs associated with source nodes
10:     $\lambda_k^{\text{in}} \leftarrow \lambda_k^{\text{in}} - \sum_{k' \in \mathcal{In}} \sigma_{k'}^{\text{in}} \lambda_{k'}^{\text{in}}$  ▷ normalize Lagrange parameters
11:     $c'_{1k} \leftarrow \lambda_k^{\text{in}}$  ▷ update augmented costs (recall that  $c_{1k}^{\text{ext}} = 0$  for all  $k \in \mathcal{In}$ )
12:   end for
13:    $\mathbf{W}' \leftarrow \mathbf{R}_{\text{ext}} \circ \exp[-\theta \mathbf{C}']$  ▷ update  $\mathbf{W}'$  matrix
14:   Solve  $(\mathbf{I} - \mathbf{W}'^T) \mathbf{z}'_1 = \mathbf{e}_1$  ▷ forward variables (row 1 of the fundamental matrix  $\mathbf{Z}'$ ) with elements  $z'_{lk}$  ( $l$  is fixed)
15:   for all  $l \in \mathcal{Out}$  do ▷ initialize Lagrange parameters associated with target nodes
16:      $\lambda_l^{\text{out}} \leftarrow \frac{1}{\theta} \log z'_{1l} - \frac{1}{\theta} \log \left( \frac{\sigma_l^{\text{out}}}{p_{\text{ext}}^{\text{out}}} \right)$  ▷ compute Lagrange parameters
17:   end for
18:   for all  $l \in \mathcal{Out}$  do ▷ update Lagrange parameters and costs associated with target nodes
19:      $\lambda_l^{\text{out}} \leftarrow \lambda_l^{\text{out}} - \sum_{l' \in \mathcal{Out}} \sigma_{l'}^{\text{out}} \lambda_{l'}^{\text{out}}$  ▷ normalize Lagrange parameters
20:      $c'_{ln} \leftarrow \lambda_l^{\text{out}}$  ▷ update augmented costs (recall that  $c_{ln}^{\text{ext}} = 0$  for all  $l \in \mathcal{Out}$ )
21:   end for
22: until convergence of  $\lambda_{\text{in}}, \lambda_{\text{out}}$ 
23:  $\mathbf{P}^* \leftarrow (\text{Diag}(\mathbf{z}'_n))^{-1} \mathbf{W}' \text{Diag}(\mathbf{z}'_1)$  ▷ compute optimal policy
24: return  $\mathbf{P}^*$ 
    
```

elements w'_{ij} are computed from the augmented costs after convergence of Algorithm 1. Further note that the rows and the columns associated with the two supernodes indexed as nodes 1 and n should be removed to obtain the coupling matrix associated with the original graph.

3.4.2 The margin-constrained bag-of-paths surprisal distance

We can now compute the **margin-constrained bag-of-paths surprisal distance** introduced in [5] (called the margin-constrained bag-of-paths surprisal distance in this paper, and itself inspired by [10]) as

$$\Delta_{ij}^{\text{sur}} = \begin{cases} -\frac{1}{2} (\log(\gamma_{ij}) + \log(\gamma_{ji})) & \text{if } i \neq j, \\ 0 & \text{if } i = j \end{cases} \quad (17)$$

where γ_{ij} is the element i, j of the coupling matrix $\mathbf{0}$ defined in Eq. (16). This distance is a generalization of the surprisal distance [1, 10] where positive weights $\mathbf{v} = (v_i)$, summing to 1, can be attached to each node through σ_{in} and σ_{out} .

Intuitively, the distance (Eq. (17)) quantifies the “surprise” generated by the event $(S = i) \wedge (T = j)$, that is, sampling a path having input node $i \in \mathcal{In}$ and output node $j \in \mathcal{Out}$ from the bag of paths $\mathcal{P}_{1n}$ defined on G_{ext} , with probability distribution provided by Eq. (7) and using augmented costs in order to satisfy the constraints (Eq. (8)).

In this work, we consider that each node acts as both input and output, with $\sigma_{\text{in}} = \sigma_{\text{out}} = \mathbf{v}$. This choice is inspired from the well-known PageRank [45,46] and the random walk with restart [47,48] models in which, at each time step, the random walker has a chance of leaving the current node (which is then similar to a sink node) for restarting in some nodes of the network (which are then similar to source nodes). This choice, although somewhat counter-intuitive,⁹ provides quite competitive results as will be shown in the next section.

Note that the authors of [5] also propose a hitting path version of this distance which is not definable in our present framework. This distance based on hitting paths will nevertheless be investigated in our experiments.

4 An illustrative example

In this section, we show how the algorithm introduced in Sect. 3.3.3 can be applied to solve a small transportation problem. We also take the opportunity to present numerical examples of the matrices introduced in Sect. 2.3.

Let us consider that a food wholesaler is planning to allocate goods from three warehouses to three retailers. The transportation costs from each warehouse (shipping source) to each retailer (shipping destination) are reported in Table 2.

The first step to apply our algorithm to this example is to transform the data of Table 2 into a bipartite graph. Then, we extend this graph, as explained in Sect. 2.3, by adding one source node 1 connected to all the warehouses and one target node 8 connected to all the retailers. Finally, the flow constraints σ_{in} and σ_{out} should be defined. The input flow σ_i^{in} of each warehouse is fixed to its supply divided by the total supply ($\sigma_{\text{in}} = [1/4; 7/24; 11/24]$). Similarly, the output flow σ_j^{out} of each retailer is fixed to its demand divided by the total demand ($\sigma_{\text{out}} = [1/3; 1/6; 1/2]$). All the matrices presented in Sect. 2 related to these graph and extended graph, as well as the coupling matrix $\mathbf{\Gamma}$, are reported in Eq. (18) for $\theta = 1$ and $\mu = 1$. In the cost matrices, ∞ represents a large positive value.

⁹ Because we inject and remove the same quantity of flow in each node, this setting is only meaningful when the parameter θ is not too large. Indeed, when θ increases, the transportation is more and more optimal and the input-output flows tend to neutralize each other, resulting in a coupling matrix that converges to the identity matrix. This is, however, not the case for intermediate measures of θ for which the coupling matrix is able to capture the similarity between nodes (in terms of proximity and high connectivity).

Table 2 Table representation of a small example of transportation problem

Shipping destination $\rightarrow$:	Retailer 1	Retailer 2	Retailer 3	Supply
Shipping source $\downarrow$:				
Warehouse 1	6	8	10	150
Warehouse 2	7	11	11	175
Warehouse 3	4	5	12	275
Demand	200	100	300	600

$$\begin{aligned}
 \mathbf{A} &= \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \\
 \mathbf{A}_{\text{ext}} &= \begin{bmatrix} 0 & 1/4 & 7/24 & 11/24 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 3 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\
 \mathbf{P} &= \begin{bmatrix} 0 & 0 & 0 & 1/3 & 1/3 & 1/3 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 \\ 0 & 0 & 0 & 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \end{bmatrix} \\
 \mathbf{P}_{\text{ext}} &= \begin{bmatrix} 0 & 1/4 & 7/24 & 11/24 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \\ 0 & 1/6 & 1/6 & 1/6 & 0 & 0 & 0 & 1/2 \\ 0 & 1/4 & 1/4 & 1/4 & 0 & 0 & 0 & 1/4 \\ 0 & 1/12 & 1/12 & 1/12 & 0 & 0 & 0 & 3/4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\
 \mathbf{C} &= \begin{bmatrix} \infty & \infty & \infty & 6 & 8 & 10 \\ \infty & \infty & \infty & 7 & 11 & 11 \\ \infty & \infty & \infty & 4 & 5 & 12 \\ \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty \end{bmatrix} \\
 \mathbf{C}_{\text{ext}} &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \infty & \infty & \infty & 6 & 8 & 10 & 0 \\ 0 & \infty & \infty & \infty & 7 & 11 & 11 & 0 \\ 0 & \infty & \infty & \infty & 4 & 5 & 12 & 0 \\ 0 & \infty & \infty & \infty & \infty & \infty & \infty & 0 \\ 0 & \infty & \infty & \infty & \infty & \infty & \infty & 0 \\ 0 & \infty & \infty & \infty & \infty & \infty & \infty & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\
 \mathbf{\Gamma} &= \begin{bmatrix} 0 & 0 & 0 & 0.0307 & 0.0068 & 0.2125 \\ 0 & 0 & 0 & 0.0367 & 0.0011 & 0.2538 \\ 0 & 0 & 0 & 0.2659 & 0.1588 & 0.0337 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (18)
 \end{aligned}$$

The transportation solutions provided by the Algorithm 1 for different θ values are represented in Fig. 2. From Fig. 2d, we can observe that the optimal total transportation cost of 4525 is reached when $\theta = 10$. However, from Fig. 2a–c, we can see that the total transportation cost moves further and further away from the optimal solution as θ approaches zero. As already mentioned in Sect. 3.1, when θ is low (close to zero), the paths are chosen according to their likelihood in the natural random walk, and therefore no longer consider costs (see Eq. (6)). For that reason, some goods are allocated to sub-optimal roads in terms of cost, which increases the total transportation cost.

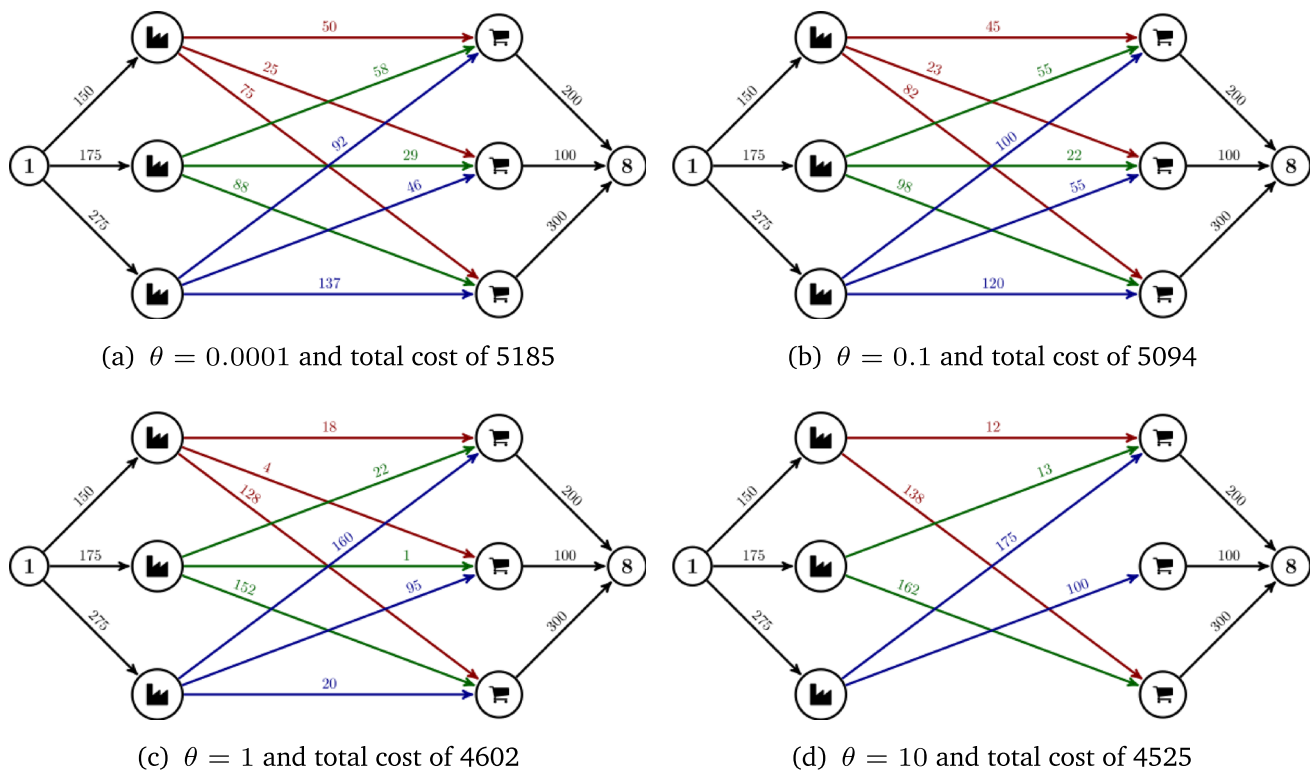


Fig. 2 Representation of the solutions of the transportation problem presented in Table 2 provided by the Algorithm 1 for different θ values. The optimal total transportation cost of 4525 is reached for $\theta = 10$. The figures are generated with Tikz

5 Experimental comparison on semi-supervised classification tasks

In this section, the introduced method of Sect. 3.4 and its equivalent on hitting paths [5] will be compared in terms of classification accuracy on semi-supervised classification tasks with the other methods defined in the bag-of-paths framework. The goal of this experiment is to highlight the best methods within the bag-of-paths framework rather than propose an extended comparison with a large number of state-of-the-art techniques. Indeed, the methods defined in the bag-of-paths framework, such as the free energy distance, have already demonstrated their competitiveness with state-of-the-art techniques in pattern recognition tasks [10,49,50].

The section is organized as follows. First, the set of investigated methods is presented in Sect. 5.1. Then, Sect. 5.2 provides details on the experimental design largely inspired by [10,12] and already partly used in [34,49]. Finally, Sect. 5.3 presents and discusses the results of the experiments.

5.1 Investigated methods

For our experimental comparisons, we have selected seven methods defined in the RSP/BoP framework, introduced in Sect. 3.1. Recall that the main difference between these two models is that the randomized shortest paths model is defined

based on the set of all (usually hitting) paths $\mathcal{P}_{st}^h$ from a unique node s to a unique target node t , and not based on the set of all paths $\mathcal{P}$ in the graph, as the bag-of-paths model does. Nevertheless, most of the methods introduced hereafter could be defined in both frameworks.

Moreover, as already stated in other terms in Sect. 2.2, the RSP framework interpolates between an optimal (exploitation) and a random (exploration) behavior based on a monitoring parameter θ . This framework therefore allowed us to define dissimilarity measures interpolating between the shortest path distance when θ is large (exploitation) and the commute time distance (up to a scaling factor) when $\theta \rightarrow 0^+$ (exploration). Therefore, it seemed clear to us to use these two boundary methods as baselines in our experiments. Finally, the seven methods retained for the experiments are:

- The *shortest path distance* (SP) between two nodes i and j is defined as the cost of the path having the smallest cost between these two nodes, derived from the cost matrix $\mathbf{C}$. This method is the most standard distance and has no hyperparameter.
- The *commute time kernel* (CT) [51,52] simply corresponds to the Moore-Penrose pseudoinverse of the Laplacian matrix $\mathbf{L}^+$ [53]. This method has no hyperparameter.

- The *free energy distance* (FE) [1,10] is a distance build on the symmetrization of the directed free energy distance presented in Eq. (B15). This method has one hyperparameter θ .
- The regular *surprisal distance* (Sup) [10] is a distance quantifying the “surprise” generated by the event $(S = i) \wedge (T = j)$ (see Sect. 3.4.2). This method has one hyperparameter θ .
- The *randomized shortest paths dissimilarity* (RSP) [1,2] is obtained by symmetrization of the expected cost of Eq. (B18). This method has one hyperparameter θ .
- The *margin-constrained bag-of-paths surprisal distance* (cBoP) is the distance introduced in [5] and re-derived in this paper (Sect. 3.4.2, Eq. (17)) from another point of view. This method has two hyperparameters θ and the non-negative weights vector $\mathbf{v}$.
- The *margin-constrained bag-of-hitting-paths surprisal distance* (cBoPH) [5] is the counterpart of the previous method in terms of hitting paths. This method has two hyperparameters θ and the non-negative weights vector $\mathbf{v}$. We computed the quantity by following [5], Algorithm 3, page 108.

5.2 Experimental design

As mentioned earlier, our experimental methodology is closely related to the one used in [10,12] and our description is largely inspired by these previous works as well as from [34,49], in which a similar methodology was used. Notice that this design was itself initially inspired by [54–58], leading to interesting results. Therefore, the following section will only summarize the procedure¹⁰ and emphasize the main differences.

5.2.1 Datasets

A collection of 14 well-known network datasets, already used in previous experimental comparisons, has been selected to evaluate the performance of the different methods. The collection includes the 4 WebKB datasets [59], the IMDB dataset [59], and 9 extracted from the 20 Newsgroup datasets [60,61]. All these datasets are described by an adjacency matrix $\mathbf{A}$ and a class label vector $\mathbf{y}$. Note that we consider that each graph is undirected and we assert it by using $\mathbf{A} = (\mathbf{A} + \mathbf{A}^T)/2$. Furthermore, the elements of the cost matrix $\mathbf{C}$ are defined as $c_{ij} = 1/a_{ij}$ as for electrical networks [10]. A summary of the main characteristics of each dataset can be found in Table 3. The datasets used during the current study are available from the corresponding author on reasonable request.

5.2.2 Experimental methodology

The graph-based semi-supervised classification methodology is divided into two parts. The first part consists of extracting {5, 10, 20%} of the dominant eigenvectors of a kernel matrix to use them as node features in a linear support vector machine (SVM).¹¹ These extracted features contain condensed information about the graph structure.

The second part consists of directly feeding the kernel matrix into a kernel SVM.¹² We will refer to the first part as 5, 10, and 20%F and the second part as Ker. The main objective is to determine to which extent the different methods can deal with partial information about the graph structure, contained in only a few dimensions (5, 10 and 20%—node features extraction), as well as with the full information contained in the kernel matrix.

The kernel matrices $\mathbf{K}$ are derived from the dissimilarity matrices by using both classical multidimensional scaling (MDS) [63] and Gaussian transformation (Gauss) [64]. We also considered centering the Gaussian kernels (GaussCenter)¹³ by applying the following transformation $\mathbf{K} = \mathbf{H}\mathbf{K}\mathbf{H}$ where $\mathbf{H} = \mathbf{I} - \mathbf{e}\mathbf{e}^T/n$ is the centering matrix, $\mathbf{e}$ is a column vector full of 1's and n is the number of nodes.

For the first part of the classification method, we also try two different options for extracting the node features. The first option is to weight the dominant eigenvectors by the square root of their corresponding eigenvalues before concatenating them into the data matrix $\mathbf{X}$. The matrix $\mathbf{X}$ contains the features of the nodes on its rows and is used as the input of the SVM. This option is equivalent to a multidimensional scaling limited to a reduced number of dimensions and it is denoted as SD (spectral decomposition). The second option corresponds to directly concatenating the dominant eigenvectors into the matrix $\mathbf{X}$ and to normalize each row, in such a way that the resulting node feature vectors are of unit length. This normalization corresponds to a projection of the rows of $\mathbf{X}$ on the unit radius sphere centered at the origin that removes the effect of the size of the feature vectors (only the direction is relevant). We will refer to this second option as NSD (normalized spectral decomposition). For conciseness, we only report for each method the results of the best kernel transformation and feature extraction options according to the Nemenyi tests rankings [65]. The best combination for each method is reported in Table 4.

The performance of the different methods will be evaluated in terms of classification accuracy. To avoid large variance in the results, all the methods are assessed by repeating 10 runs of a standard 5×5 nested cross-validation

¹⁰ The interested reader will find more details in Section 7 of [10].

¹¹ We use the LIBSVM library [62] with the options ‘-s 0’ and ‘-t 0’.

¹² We use the LIBSVM library [62] with the options ‘-s 0’ and ‘-t 4’.

¹³ We did not apply this transformation to the MDS kernels as they are centered by construction.

Table 3 Main characteristics of the datasets used in our experiments

Dataset name	#Labels	#Nodes	#Edges	Prior of the majority class
webKB-cornell (DB1)	6	346	13416	41.91%
webKB-texas (DB2)	6	334	16494	48.80%
webKB-washington (DB3)	6	434	15231	39.17%
webKB-wisconsin (DB4)	6	348	16625	44.54%
imdb (DB5)	2	1126	20282	50.18%
news-2cl-1 (DB6)	2	400	33854	50.00%
news-2cl-2 (DB7)	2	398	21480	50.25%
news-2cl-3 (DB8)	2	399	36527	50.13%
news-3cl-1 (DB9)	3	600	70591	33.34%
news-3cl-2 (DB10)	3	598	68201	33.44%
news-3cl-3 (DB11)	3	595	64169	33.61%
news-5cl-1 (DB12)	5	998	176962	20.04%
news-5cl-2 (DB13)	5	999	164452	20.02%
news-5cl-3 (DB14)	5	997	155618	20.06%

Table 4 The best combination of kernel transformation and feature extraction options for each method according to Nemenyi tests

Feature set →: Method ↓:	5%	10%	20%	Ker
CT	NSD	NSD	NSD	—
cBoP	GaussCenterSD	GaussCenterSD	GaussCenterSD	GaussCenter
cBoPH	GaussCenterSD	GaussSD	GaussCenterSD	Gauss
FE	GaussCenterSD	GaussCenterSD	GaussSD	Gauss
RSP	GaussCenterSD	GaussSD	GaussCenterSD	Gauss
SP	GaussSD	GaussSD	GaussCenterSD	GaussCenter
Sup	GaussSD	GaussSD	GaussCenterSD	Gauss

methodology with different folds of labeled/unlabeled nodes. In each external 5-folds cross-validation, methods are trained on one fold containing 20% of the labels, and the remaining 80% of the labels are hidden. The parameters of each method are tuned on the training fold by performing an internal 5-fold cross-validation with a labeling rate of 80%. In each run, external and internal folds are kept identical on all methods. The final accuracy and standard deviation are obtained by averaging the 50 results of the external cross-validation folds.

Concerning the parameters, the θ of the bag-of-paths-type methods are tuned among values of $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$, and the margin parameter c of the SVM is tuned on the set of values $\{10^{-2}, 10^{-1}, 1, 10, 100\}$. For the margin-constrained bag-of-paths methods introduced in [5] and reinterpreted in Sect. 3.4.2, we tested the three following different types of positive weights $\mathbf{v}$:

- Uniform weights: $\mathbf{e}/n$;
- L1-normalized degree weights: $\mathbf{d}/(\mathbf{e}^T \mathbf{d})$ with $\mathbf{d} = \mathbf{A}\mathbf{e}$;
- L1-normalized inverse degree weights: $(\mathbf{e} \div \mathbf{d})/(\mathbf{e}^T (\mathbf{e} \div \mathbf{d}))$.

where $\div$ is the elementwise division. To avoid redundancy in the comparisons, we only present the best results over the three (positive sum-to-one) weightings for each of the two margin-constrained bag-of-paths methods according to Nemenyi tests rankings [65]. We observed that, for every feature extractions sets, the weight achieving the best result is the normalized degree for both hitting and non-hitting margin-constrained bag-of-paths surprisal distances.

5.3 Results and discussion

The classification accuracy and standard deviation averaged over the 50 runs are reported in Table 5 for the 14 datasets and the four extracted feature sets. The best performing method is highlighted in boldface for each dataset and each feature set. BoldItalic values indicate the best performing method overall (across all feature sets) for each dataset.

Table 5 Classification accuracy in percent $\pm$ standard deviation for the various classification methods, obtained on the different datasets

Method $\rightarrow$: Dataset $\downarrow$:	CT	cBoP	cBoPH	FE	RSP	SP	Sup
5%F							
webKB-cornell	54.10 $\pm$ 3.91	59.41$\pm$3.88	58.82 $\pm$ 2.82	59.07 $\pm$ 2.85	58.82 $\pm$ 2.90	47.65 $\pm$ 3.43	58.67 $\pm$ 3.15
webKB-texas	67.24 $\pm$ 3.40	78.62$\pm$2.34	77.44 $\pm$ 3.05	77.87 $\pm$ 3.13	76.28 $\pm$ 3.52	66.89 $\pm$ 2.83	77.14 $\pm$ 2.94
webKB-washington	65.59 $\pm$ 2.10	71.10$\pm$3.02	69.65 $\pm$ 2.74	69.50 $\pm$ 2.83	69.49 $\pm$ 2.87	62.75 $\pm$ 2.69	68.98 $\pm$ 2.61
webKB-wisconsin	71.88 $\pm$ 3.53	79.43$\pm$1.99	78.53 $\pm$ 2.07	78.16 $\pm$ 1.87	76.62 $\pm$ 2.23	64.14 $\pm$ 3.12	77.67 $\pm$ 2.24
imdb	78.90$\pm$1.31	77.61 $\pm$ 1.20	78.29 $\pm$ 1.74	78.34 $\pm$ 1.76	78.09 $\pm$ 1.45	77.58 $\pm$ 1.57	78.28 $\pm$ 1.42
news-2cl-1	96.64$\pm$0.73	95.50 $\pm$ 1.30	95.84 $\pm$ 1.04	95.66 $\pm$ 1.18	95.56 $\pm$ 1.74	92.82 $\pm$ 1.30	95.59 $\pm$ 1.30
news-2cl-2	91.17 $\pm$ 1.34	91.64 $\pm$ 2.22	93.29 $\pm$ 1.50	93.22 $\pm$ 1.41	92.54 $\pm$ 1.96	91.61 $\pm$ 1.81	93.47$\pm$1.40
news-2cl-3	94.08 $\pm$ 0.68	96.18 $\pm$ 1.01	96.47 $\pm$ 0.81	96.39 $\pm$ 0.90	96.19 $\pm$ 1.26	96.59$\pm$0.86	96.40 $\pm$ 1.02
news-3cl-1	90.52 $\pm$ 1.32	92.66 $\pm$ 1.28	93.05$\pm$1.43	92.82 $\pm$ 1.48	92.75 $\pm$ 1.48	92.87 $\pm$ 1.18	92.91 $\pm$ 1.38
news-3cl-2	89.92 $\pm$ 1.23	92.00 $\pm$ 1.51	92.50$\pm$1.38	92.29 $\pm$ 1.39	92.07 $\pm$ 1.43	89.61 $\pm$ 1.46	92.37 $\pm$ 1.66
news-3cl-3	89.63 $\pm$ 1.21	92.14 $\pm$ 1.65	92.91$\pm$1.36	92.47 $\pm$ 1.61	92.06 $\pm$ 1.81	91.42 $\pm$ 1.00	92.71 $\pm$ 1.40
news-5cl-1	78.50 $\pm$ 1.55	88.49 $\pm$ 0.95	88.79$\pm$1.05	88.49 $\pm$ 1.17	88.43 $\pm$ 1.05	87.93 $\pm$ 1.09	88.49 $\pm$ 1.37
news-5cl-2	76.04 $\pm$ 1.61	82.76 $\pm$ 1.42	83.23 $\pm$ 1.33	83.33 $\pm$ 1.40	82.58 $\pm$ 1.76	81.80 $\pm$ 1.24	83.35$\pm$1.37
news-5cl-3	75.25 $\pm$ 1.78	82.19 $\pm$ 1.36	82.51 $\pm$ 1.73	82.81$\pm$1.80	82.31 $\pm$ 1.53	77.17 $\pm$ 1.77	82.49 $\pm$ 1.62
10%F							
webKB-cornell	53.51 $\pm$ 3.28	58.08$\pm$3.25	57.98 $\pm$ 3.19	57.79 $\pm$ 3.10	57.95 $\pm$ 3.95	48.29 $\pm$ 3.27	57.00 $\pm$ 3.89
webKB-texas	70.26 $\pm$ 2.71	78.05$\pm$3.14	76.82 $\pm$ 3.03	76.81 $\pm$ 3.27	76.48 $\pm$ 2.49	65.86 $\pm$ 3.25	76.04 $\pm$ 3.46
webKB-washington	63.53 $\pm$ 2.56	70.46$\pm$2.58	68.29 $\pm$ 2.46	68.63 $\pm$ 2.15	69.59 $\pm$ 2.78	62.40 $\pm$ 2.69	67.75 $\pm$ 2.54
webKB-wisconsin	73.00 $\pm$ 1.36	79.61$\pm$2.01	77.67 $\pm$ 2.17	77.59 $\pm$ 1.97	77.06 $\pm$ 2.05	63.79 $\pm$ 2.59	76.52 $\pm$ 2.11
imdb	77.35 $\pm$ 1.31	77.76 $\pm$ 1.31	78.95 $\pm$ 1.42	79.10$\pm$1.38	78.83 $\pm$ 1.36	77.57 $\pm$ 1.56	78.63 $\pm$ 1.55
news-2cl-1	93.30 $\pm$ 1.64	95.00 $\pm$ 1.06	95.08 $\pm$ 1.69	95.06 $\pm$ 1.30	95.17$\pm$1.43	93.15 $\pm$ 1.19	94.90 $\pm$ 1.73
news-2cl-2	91.79 $\pm$ 1.79	91.76 $\pm$ 2.04	92.36 $\pm$ 1.69	92.32 $\pm$ 1.64	91.87 $\pm$ 1.73	90.78 $\pm$ 2.17	92.41$\pm$1.52
news-2cl-3	93.53 $\pm$ 1.17	96.22 $\pm$ 0.83	96.17 $\pm$ 1.13	96.29 $\pm$ 1.02	96.36 $\pm$ 1.03	96.72$\pm$0.74	96.28 $\pm$ 1.14
news-3cl-1	89.84 $\pm$ 1.26	92.78 $\pm$ 1.14	93.02 $\pm$ 1.66	93.08$\pm$1.25	92.63 $\pm$ 1.53	92.83 $\pm$ 1.09	92.69 $\pm$ 1.46
news-3cl-2	90.28 $\pm$ 1.56	92.23 $\pm$ 1.53	92.72$\pm$1.39	92.23 $\pm$ 1.46	92.10 $\pm$ 1.56	89.72 $\pm$ 1.55	92.25 $\pm$ 1.67
news-3cl-3	90.87 $\pm$ 1.33	91.97 $\pm$ 1.64	92.97$\pm$1.47	92.23 $\pm$ 1.48	92.21 $\pm$ 1.27	91.72 $\pm$ 1.12	92.76 $\pm$ 1.46
news-5cl-1	83.95 $\pm$ 1.16	88.56 $\pm$ 1.03	88.79 $\pm$ 1.26	88.71 $\pm$ 0.96	88.45 $\pm$ 1.30	88.01 $\pm$ 0.81	88.89$\pm$1.07
news-5cl-2	77.20 $\pm$ 1.74	82.70 $\pm$ 1.86	83.56$\pm$1.52	83.18 $\pm$ 2.02	82.42 $\pm$ 1.96	81.55 $\pm$ 1.35	83.36 $\pm$ 1.88
news-5cl-3	80.50 $\pm$ 1.82	82.16 $\pm$ 1.82	82.79$\pm$1.66	82.70 $\pm$ 1.96	82.58 $\pm$ 1.63	77.52 $\pm$ 1.80	82.59 $\pm$ 1.71
20%F							
webKB-cornell	56.19 $\pm$ 2.87	57.72 $\pm$ 3.44	57.59 $\pm$ 2.75	57.81 $\pm$ 3.64	58.13$\pm$3.71	48.46 $\pm$ 3.62	56.54 $\pm$ 3.32
webKB-texas	75.18 $\pm$ 1.91	77.66$\pm$3.48	75.91 $\pm$ 3.95	76.72 $\pm$ 3.60	76.45 $\pm$ 3.22	65.06 $\pm$ 3.32	76.12 $\pm$ 3.57
webKB-washington	65.59 $\pm$ 1.66	70.41$\pm$2.31	68.95 $\pm$ 2.23	69.36 $\pm$ 2.04	69.83 $\pm$ 2.80	63.12 $\pm$ 2.31	67.61 $\pm$ 2.31
webKB-wisconsin	74.48 $\pm$ 1.71	79.21$\pm$2.12	76.98 $\pm$ 2.15	78.09 $\pm$ 2.69	77.00 $\pm$ 2.28	63.04 $\pm$ 2.83	75.57 $\pm$ 2.46
imdb	78.38 $\pm$ 1.48	77.75 $\pm$ 1.47	78.83 $\pm$ 1.84	79.06$\pm$1.51	78.71 $\pm$ 1.66	77.43 $\pm$ 1.69	78.42 $\pm$ 1.54
news-2cl-1	87.15 $\pm$ 2.29	94.76 $\pm$ 1.44	95.16$\pm$1.37	95.00 $\pm$ 1.53	95.00 $\pm$ 2.07	93.31 $\pm$ 1.13	94.95 $\pm$ 1.67
news-2cl-2	86.75 $\pm$ 2.48	91.31 $\pm$ 2.14	91.92 $\pm$ 1.88	92.01$\pm$1.41	91.89 $\pm$ 1.53	90.94 $\pm$ 1.81	91.63 $\pm$ 2.17
news-2cl-3	90.46 $\pm$ 1.53	96.29 $\pm$ 0.88	96.35 $\pm$ 0.93	96.30 $\pm$ 0.98	96.29 $\pm$ 1.10	96.64$\pm$1.00	96.38 $\pm$ 0.97
news-3cl-1	84.03 $\pm$ 1.95	92.69 $\pm$ 1.39	93.00 $\pm$ 1.25	93.01$\pm$1.40	92.50 $\pm$ 1.43	92.82 $\pm$ 0.98	92.99 $\pm$ 1.27
news-3cl-2	85.74 $\pm$ 1.89	92.32 $\pm$ 1.21	92.99$\pm$1.22	92.69 $\pm$ 1.32	92.69 $\pm$ 1.08	89.57 $\pm$ 1.39	92.81 $\pm$ 1.41
news-3cl-3	87.42 $\pm$ 2.06	92.15 $\pm$ 1.44	92.71 $\pm$ 1.19	92.80$\pm$1.10	92.38 $\pm$ 0.94	91.82 $\pm$ 1.14	92.80 $\pm$ 1.16
news-5cl-1	81.21 $\pm$ 1.66	88.55 $\pm$ 1.12	88.91$\pm$1.07	88.75 $\pm$ 0.98	88.53 $\pm$ 1.15	88.09 $\pm$ 0.93	88.84 $\pm$ 1.12
news-5cl-2	75.69 $\pm$ 1.82	82.85 $\pm$ 1.48	83.48$\pm$1.52	83.25 $\pm$ 1.45	82.27 $\pm$ 1.92	81.69 $\pm$ 1.46	83.44 $\pm$ 1.41

Table 5 continued

Method →: Dataset ↓:	CT	cBoP	cBoPH	FE	RSP	SP	Sup
news-5cl-3	75.81±1.84	82.42±1.57	82.81±1.80	82.94±1.61	81.98±1.92	77.57±1.49	82.86±1.80
	Ker						
webKB-cornell	42.05±0.40	59.45±2.99	58.79±3.38	58.51±2.81	58.30±3.76	48.58±3.68	58.71±2.75
webKB-texas	50.13±2.04	78.87±2.81	76.68±3.26	77.11±3.17	76.88±2.93	64.50±3.57	76.91±3.05
webKB-washington	44.44±5.54	71.84±2.04	70.12±2.19	69.46±2.02	69.78±3.14	63.08±2.55	69.05±2.20
webKB-wisconsin	53.28±4.26	78.35±1.97	76.42±2.42	77.26±2.24	75.69±2.32	63.96±2.93	76.25±2.32
imdb	79.15±1.08	77.86±1.57	79.24±1.48	78.95±1.51	78.99±1.21	77.57±1.97	79.58±1.52
news-2cl-1	90.21±7.25	94.98±1.28	95.63±0.88	95.40±1.21	95.66±1.50	93.41±1.08	95.45±0.88
news-2cl-2	92.08±2.03	91.56±1.54	92.59±1.60	92.27±1.61	92.04±1.60	91.25±1.56	92.32±1.97
news-2cl-3	87.54±9.94	96.34±1.06	96.62±0.74	96.62±0.78	96.47±0.96	96.79±0.78	96.72±0.84
news-3cl-1	63.96±15.90	92.85±1.25	93.33±1.00	93.24±0.92	92.80±1.32	92.84±1.00	93.29±1.02
news-3cl-2	54.96±15.96	92.56±1.14	93.23±1.00	92.84±1.05	92.65±1.16	89.75±1.28	93.26±0.97
news-3cl-3	52.11±12.71	92.30±1.16	93.26±1.02	92.78±1.16	92.42±1.27	91.89±0.79	92.98±1.21
news-5cl-1	27.06±7.41	88.54±1.24	88.94±0.89	88.78±1.02	88.57±1.03	88.16±0.91	88.87±0.96
news-5cl-2	30.23±8.69	82.73±1.47	83.94±1.32	83.26±1.20	82.66±1.64	81.66±1.38	83.70±1.27
news-5cl-3	25.76±7.35	82.28±1.61	83.54±1.32	83.17±1.17	82.43±1.75	77.62±1.46	83.52±1.20

Results are reported for the four feature sets (5, 10, 20%, and Ker). For each dataset and method, the final accuracy and standard deviation are obtained by averaging over 10 runs of a standard cross-validation procedure. Each run consists of a nested cross-validation with 5 external folds (test sets, for validation) on which the accuracy and the standard deviation of the classifier are averaged, and 5 internal folds (for parameter tuning). The best performing method is highlighted in boldface for each dataset and each feature set. BoldItalic values indicate the best performance overall for each dataset, across all feature sets

5.3.1 Comparison of the different methods

The classification results of the seven different methods are now compared for each feature set, and then across all feature sets.

From the raw results of Table 5, it can be seen that 11 of the best results over 14 datasets are obtained by directly feeding the kernel to the SVM. Furthermore, 9 out of these 14 results are obtained by the two introduced constrained optimal transport methods (cBoP and cBoPH). Across all the feature sets, we observe that the cBoP seems to outperform all the methods on DB1 to DB4 except in 20%F where the RSP prevails on DB1. The results on DB5 are more contrasted: the FE performs the best for 10 and 20%F, whereas the CT and the Sup are respectively the highlighted methods for 5%F and Ker. For the newsgroup datasets (DB6-DB14), the best method is dataset-dependent and feature set-dependent except for the DB8 where the SP dominates. Nevertheless, we can underline that the cBoPH appears 18 times among the 32 highlighted results of the eight other newsgroup datasets. The remaining methods only appear respectively six times for the FE, five times for the Sup, two times for the RSP, and one time for the CT. Another observation is that the CT obtains slightly lower results with Ker compared to its performances in the other feature sets. The additional information provided by the kernel is therefore not optimally exploited by this method.

In order to have a more general overview of the results, a Borda ranking of the methods is performed for each feature set and reported in Table 6. The Borda ranking starts by sorting all the methods in ascending order of classification accuracy for each dataset. Then, the score of each method is computed by adding its ranks over all datasets. Therefore, the best method is the one with the highest Borda score reflecting a higher global accuracy across all the datasets.

From Table 6, we observe that the ranking of the methods does not change much across the feature sets. The top three methods are always the cBoPH, the FE, and the Sup. For 5 and 10%F, the cBoPH is first followed by the FE and thereafter by the Sup. The cBoPH only exchanges its first rank for the second one with the FE for 20%F, whereas the Sup takes the second position of the FE for Ker. The fourth and the fifth positions of the ranking are respectively taken by the cBoP and the RSP for 5%F and Ker, although they exchange their ranks for 20%F. The two methods on the bottom of the ranking are the SP and the CT. The CT is at the last position except for 5%F where it is the SP. Globally, the cBoPH appears in the first position, followed in order by the FE, the Sup, the cBoP, the RSP, the SP, and the CT baselines. Furthermore, by observing the scores, we notice that the SP and the CT obtain much worse results in comparison to the other methods.

Table 6 Overall position of the different classification techniques for the four feature sets (5, 10, 20% and kernel), and overall, according to Borda's method performed across all datasets (the higher the score, the better)

Feature set →: Method ↓:	5%		10%		20%		Ker		Overall	
	Score	Position	Score	Position	Score	Position	Score	Position	Score	Position
cBoPH	83	1	82	1	76	2	84	1	325	1
FE	73	2	74	2	83	1	67	3	297	2
Sup	69	3	63	3	66	3	78	2	276	3
cBoP	59	4	62	4	58	5	57	4	236	4
RSP	48	5	60	5	60	4	53	5	221	5
SP	29	7	28	6	30	6	32	6	119	6
CT	31	6	23	7	20	7	21	7	95	7

The next step of our analysis consists of comparing the different methods across all the 14 datasets through a Friedman test followed by a Nemenyi post-hoc test [65]. The Friedman test is a non-parametric equivalent of the repeated-measures ANOVA. The null hypothesis (H_0) of this test is that all the classifiers have the same average ranks. The p values of the Friedman tests are respectively 4.5×10^{-7} for 5%F, 3.7×10^{-8} for 10%F, 7.8×10^{-9} for 20%F, and 7.5×10^{-9} for Ker. All these p values are lower than the threshold α of 0.05, meaning that we can reject H_0 and that at least one classifier is significantly different from the others. As all the Friedman tests are positive, we can perform Nemenyi tests, which determine whether or not the performance of each method differs significantly from another. The results of these tests are reported in Fig. 3a–d. First of all, we can observe that the rankings provided by the Nemenyi tests are quite similar to those provided by the Borda ranking. The tests confirm that the cBoPH, the FE, and the Sup all provide competitive results, which are significantly superior to the results obtained by the SP and the CT for all feature sets. Moreover, the cBoPH also outperforms the RSP in 5%F. As regards the cBoP, it performs significantly better than the CT on all feature sets except for 5%F. Furthermore, we can further observe that the cBoP outperforms the SP for 10%F. The tests also show that the RSP obtains results significantly superior to those of the CT for 10 and 20%F.

We continue our analysis by performing multiple Wilcoxon signed-ranks tests for matched data [65] to potentially discover other significant pairwise differences between the methods. The Wilcoxon signed-ranks test is a non-parametric equivalent of the paired t test. The null hypothesis (H_0) of this test is that the two compared classifiers perform equally well. The results of these tests are presented in Table 7 for 5 and 10%F and in Table 8 for 20%F and Ker. All the p values lower than our threshold α of 0.05 are highlighted in boldface indicating that H_0 is rejected. Besides confirming the findings of the Friedman-Nemenyi tests, the Wilcoxon tests show that the SP is outperformed by all the methods except the CT on all feature sets. As regards the CT, it is as well outperformed

by all the methods except the SP for 5, 10, and 20%F. Moreover, the CT obtains results significantly inferior to those of all the methods in Ker. These findings tend to indicate that the techniques developed in the bag-of-paths framework can take advantage of both the SP and the CT to outperform them whatever the retained amount of information. The tests also highlight that the cBoPH performs significantly better than the Sup in 10 and 20%F and the RSP in 10%F and Ker. On its side, the RSP obtains results significantly inferior to those of the Sup for 5%F, and Ker as well as those of the FE on all feature sets, except for 10%F.

Finally, for information, we also analyze the results overall by concatenating the 56 results obtained across the 14 datasets and the four feature sets for each method (last drawing, Fig. 3e). Here, the assumption that the 56 datasets are independent of each other is certainly not fulfilled (they are partially overlapping), so that we cannot draw any statistical conclusion. However, we can notice that these results confirm the outputs of the Borda ranking.

In summary, the first part of the experiments showed that three methods stand out from the others: the cBoPH, the FE, and the Sup. These methods achieve to consistently outperform most of the methods through all the feature sets on the investigated datasets. Among these three, the cBoPH set itself apart by being the best method across three of the four feature sets and the second in the last one according to the Borda ranking (see Table 6).

5.3.2 Comparison of the impact of the different extracted feature sets

We now analyze the impact of the feature extraction technique (with a growing number of extracted features) on the classification results, limited to the three methods performing best in the first part of the experiments for conciseness. Nevertheless, we have performed the analysis on all the methods and have drawn similar conclusions except for the CT. As we already pointed out, the performances of the CT are lower in

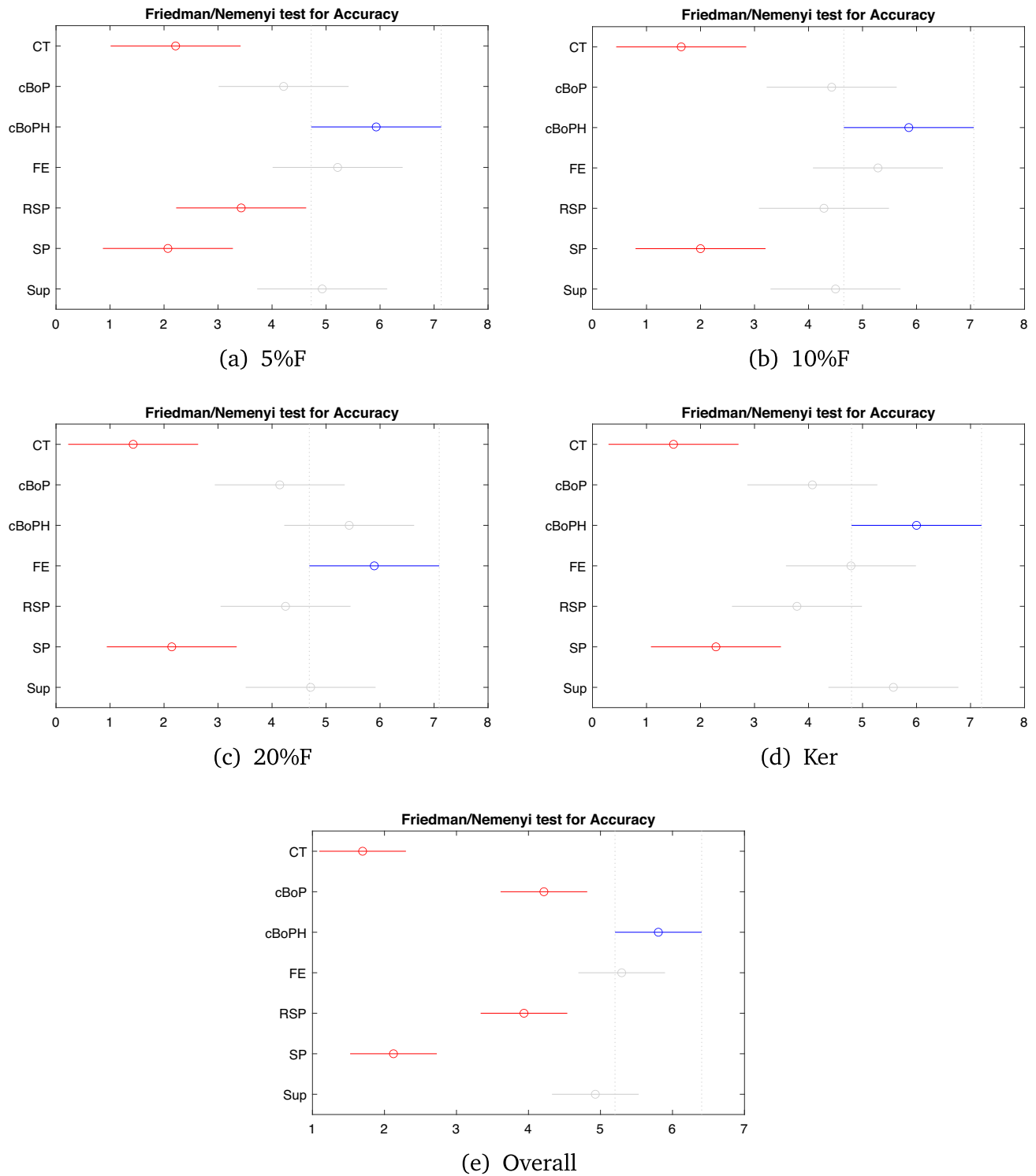


Fig. 3 Mean ranks and 95% Nemenyi confidence intervals for the 7 methods across the 14 datasets for feature sets 5%F (a), 10%F (b), 20%F (c), Ker (d) and overall (all feature sets) (e). Two methods are considered as significantly different if their confidence intervals do not

overlap. The axis-x unit is the average rank of the methods. The higher the rank, the best the method. The best method is highlighted. The figures are generated with Matlab

Table 7 The p values provided by a pairwise Wilcoxon signed-rank test, for 5%F in the upper right triangle and 10%F in the lower left

Method →: Method ↓:	CT	cBoP	cBoPH	FE	RSP	SP	Sup
CT	1.0000	0.0012	0.0006	0.0006	0.0006	0.9515	0.0006
cBoP	0.0002	1.0000	0.6257	0.6698	0.4631	0.0023	0.8077
cBoPH	0.0001	0.5830	1.0000	0.3910	0.0002	0.0002	0.0085
FE	0.0001	0.6257	0.2166	1.0000	0.0001	0.0006	0.3258
RSP	0.0001	0.5830	0.0419	0.0906	1.0000	0.0006	0.0085
SP	0.6257	0.0012	0.0004	0.0004	0.0006	1.0000	0.0004
Sup	0.0001	0.8077	0.0017	0.1353	0.7609	0.0006	1.0000

All the p values lower than our threshold α of 0.05 are highlighted

Table 8 The p values provided by a pairwise Wilcoxon signed-rank test, for 20%F in the upper right triangle and Ker in the lower left

Method →: Method ↓:	CT	cBoP	cBoPH	FE	RSP	SP	Sup
CT	1.0000	0.0002	0.0001	0.0001	0.0001	0.5416	0.0001
cBoP	0.0006	1.0000	0.5016	0.3258	0.5830	0.0012	0.9032
cBoPH	0.0001	0.6257	1.0000	0.3258	0.2958	0.0004	0.0494
FE	0.0004	0.8552	0.0906	1.0000	0.0327	0.0004	0.1040
RSP	0.0001	0.5830	0.0419	0.0906	1.0000	0.0006	0.8552
SP	0.0006	0.0017	0.0002	0.0002	0.0006	1.0000	0.0004
Sup	0.0001	0.7148	0.1726	0.2166	0.0134	0.0002	1.0000

All the p values lower than our threshold α of 0.05 are highlighted

Ker compared to the other feature sets which is not the case for the other methods.

To analyze the results of the three methods across all the feature sets (5, 10, 20% and Ker), we follow the same procedure as before. First, we perform Friedman tests to identify if there are some differences between the feature sets for each method. As all the p values of these tests are lower than our threshold α of 0.05,¹⁴ we pursue our analysis by performing Nemenyi tests [65] and report the results in Fig. 4a–c. From these figures, we can observe that the Ker feature extraction outperforms the 10 and the 20%F feature sets for the Sup and the cBoPH methods. We can also notice that the results obtained by the Ker feature set are significantly superior to those of the 10%F feature set for the FE method.

To refine our analysis, we again perform multiple Wilcoxon signed-ranks tests performing pairwise comparisons [65]. The Wilcoxon tests show that the Ker feature set outperforms (again at the $\alpha = 0.05$ level) the 20%F feature set for the FE (p value = 0.0494) method. Moreover, we can also observe that the 5%F feature set seems to outperform the 10%F feature set for the Sup (p value = 0.0494). Nevertheless, the tests do not show any significant difference between the Ker feature set and the 5%F feature set for the cBoPH (p value = 0.3910), the FE (p value = 0.8552) and the Sup (p value = 0.2412).

¹⁴ The p values of the Friedman tests are respectively 0.0080 for the cBoPH, 0.0370 for the FE, and 0.0071 for the Sup.

To conclude, the second part of the experiments highlighted that the results obtained by extracting 5% of dominant eigenvectors are not significantly different from the results obtained by using all the information contained in the kernel for all the methods except the CT. This finding seems to show that the techniques can perform well with only partial information, at least on the investigated datasets and following the stated methodology.

6 Conclusion

In this work, we introduced a new algorithm solving the relative entropy-regularized minimum expected cost flow problem on a graph with capacity constraint. This new formulation of the problem extends the previous ones [4,5] by potentially integrating flow capacity constraints, which frequently appear in real-world applications [14]. Therefore, this contribution expands the previous models to a larger range of applications.

Furthermore, the first part of the experimental comparisons on graph-based semi-supervised classification problems demonstrated that the margin-constrained bag-of-paths surprisal distance, and especially its hitting version, are competitive in comparison with other bag-of-paths methods and, consequently, with other state-of-the-art techniques [10,49,50].

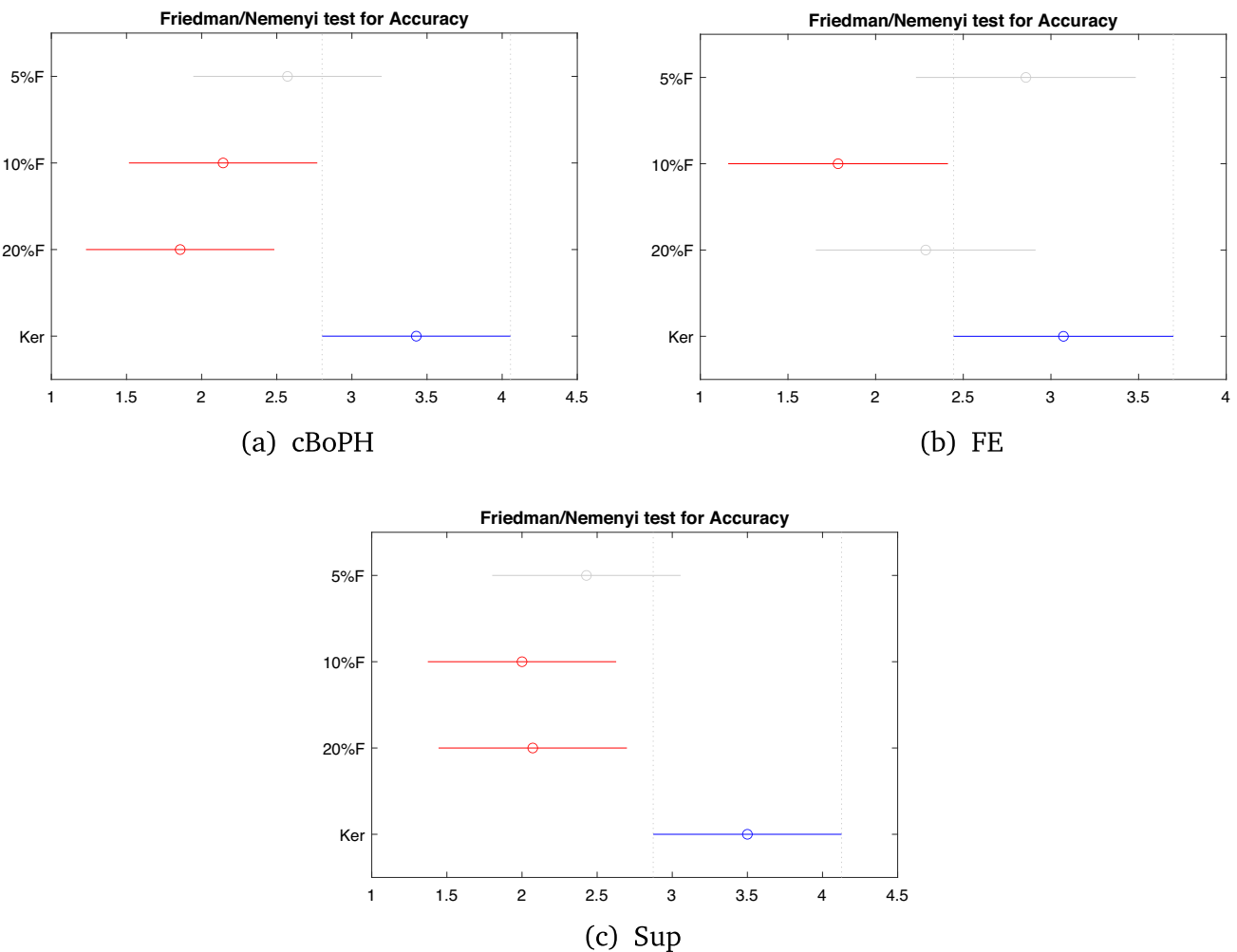


Fig. 4 Mean ranks and 95% Nemenyi confidence intervals for the three best methods across the four feature sets (5, 10, 20%F, and Ker). Two feature sets are considered as significantly different if their confidence intervals do not overlap. The axis-x unit is the average rank of the feature

sets: the higher the rank, the best the results obtained on the feature set. Each time, the best feature set is highlighted. The figures are generated with Matlab

In addition, the second part of the experiments shows that the performance of the two best bag-of-paths methods does not decrease significantly with partial information about the graph structure, in comparison with all the available information, in our semi-supervised classification tasks.

Future work will aim at developing a similar approach for a bag of hitting paths, instead of regular paths in the present work. Another interesting track would be to make the link between the present approach and electrical current in the case of an undirected graph [66].

Acknowledgements This work was partially supported by the Immediate and the Brufence projects funded by InnovIris (Brussels region), as well as former projects funded by the Walloon region, Belgium, and the Research Council of Norway. We thank these institutions for giving us the opportunity to conduct both fundamental and applied research.

Appendix

The appendices provide some derivations of relevant quantities introduced in the main text as well as basic quantities related to the standard randomized shortest paths model.

Appendix A Computing transition probabilities compatible with output flow

In this section, we derive the form of the natural transition matrix on the extended graph G_{ext} in such a way that output flow constraints are satisfied (see Eq. (1)) by using a consistency argument [4,5].

A.1 The transition matrix of the natural random walk on the extended graph

Let us deduce the form of the transition matrix $\mathbf{P}_{\text{ext}}$ of the natural random walk on the extended graph in addition to the value of $\mathbf{w}$ in Eq. (3). Clearly its first row (source node 1) is $[0 \ \sigma_{\text{in}}^T \ 0]$ and the last row (target node n) is full of zeros (killing, absorbing, node).

The row sums corresponding to the other nodes $i \in \mathcal{V}$ (belonging to the original graph G) of the new, extended, $n \times n$ matrix $\mathbf{A}_{\text{ext}} = (a_{ij}^{\text{ext}})$ (Eq. (3)) are equal to¹⁵ $[\mathbf{A}_{\text{ext}}]_{i\bullet} = a_{i\bullet}^{\text{ext}} = a_{i\bullet} + w_i$ with $a_{i\bullet} = \sum_{j \in \mathcal{V}} a_{ij}$. Thus, for $i, j \in \mathcal{V}$ (in other words, $i, j \notin \{1, n\}$), $p_{ij}^{\text{ext}} = a_{ij} / (a_{i\bullet} + w_i)$ and, for $j = n$ (last column), $p_{in}^{\text{ext}} = w_i / (a_{i\bullet} + w_i)$.

We now introduce the $(n-2) \times 1$ vector α containing the elements corresponding to nodes $i \in \mathcal{V}$ on the last column n of $\mathbf{P}_{\text{ext}}$ (see Eq. (5)),

$$\alpha_i \triangleq p_{in}^{\text{ext}} = \frac{w_i}{a_{i\bullet} + w_i} \text{ for } i \in \mathcal{V} \quad (\text{A1})$$

hence containing elements i, n of the $n \times n$ transition matrix of the extended graph. This quantity corresponds to the probability of being “killed” in node i during the next step of the random walk (transiting to the absorbing/cemetery node n). Moreover, it is easy to verify that $1 - \alpha_i = a_{i\bullet} / (a_{i\bullet} + w_i)$, which is the probability of surviving (continuing the walk). Therefore, we readily obtain, for $i, j \in \mathcal{V}$,

$$p_{ij}^{\text{ext}} = \frac{a_{ij}}{a_{i\bullet} + w_i} = \frac{a_{i\bullet}}{(a_{i\bullet} + w_i)} \times \frac{a_{ij}}{a_{i\bullet}} = (1 - \alpha_i) p_{ij} \quad (\text{A2})$$

where the p_{ij} are the transition probabilities of the natural random walk on the original graph G (see Eq. (2)). Therefore, the **transition matrix** representing the killed random walk on the extended graph is

$$\mathbf{P}_{\text{ext}} = \begin{matrix} & \begin{matrix} 1 & \{2, \dots, (n-1)\} = \mathcal{V} & n \end{matrix} \\ \begin{matrix} 1 \\ \{2, \dots, (n-1)\} = \mathcal{V} \\ n \end{matrix} & \begin{bmatrix} 0 & \sigma_{\text{in}}^T & 0 \\ \mathbf{0} \ (\mathbf{I} - \mathbf{Diag}(\alpha)) \mathbf{P} & \alpha \\ 0 & \mathbf{0}^T & 0 \end{bmatrix} \end{matrix}$$

where $\mathbf{Diag}(\alpha)$ is a diagonal matrix with α on its diagonal and the elements of the vector α still have to be determined—this problem is solved in the next subsection. Once α computed, the elements of the weight vector $\mathbf{w}$ appearing in the adjacency matrix of Eq. (3) can be obtained from Eq. (A1),

$$w_i = \frac{\alpha_i a_{i\bullet}}{1 - \alpha_i} \text{ or, in matrix form,} \\ \mathbf{w} = (\alpha \circ (\mathbf{Ae})) \div (\mathbf{e} - \alpha) \quad (\text{A3})$$

where $\circ$ and $\div$ are respectively the elementwise product and division.

A.2 Computing the values of α

In this subsection, the vector α is computed by following a reasoning similar to the one appearing in [4,5]. Let us first recall the two fundamental Markov chain quantities defined on G_{ext} , the expected number of visits to node $i \in \mathcal{V}$, denoted as $\bar{n}_i$, and the expected number of passages (also called the flow) through edge (i, j) , n_{ij} . Recall that we have $\bar{n}_j = \sum_{i \in \text{Pred}(j)} \bar{n}_{ij}$ and $\bar{n}_{ij} = \bar{n}_i p_{ij}^{\text{ext}}$ for any $j \in \mathcal{V}$.

The elements α_i with $i \in \mathcal{V}$ will be set such that the flows in the output edges (i, n) , $\bar{n}_{in}$, are equal to the prescribed output flows, $\bar{n}_{in} = \bar{n}_i p_{in}^{\text{ext}} = \bar{n}_i \alpha_i = \sigma_i^{\text{out}}$. Thus, $\alpha_i = \sigma_i^{\text{out}} / \bar{n}_i$ and we therefore have to find the $\bar{n}_i$.

To do so, let us start from the previous identity, valid for each $j \neq \{1, n\}$ (and thus $j \in \mathcal{V}$) on the extended graph,

$$\begin{aligned} \bar{n}_j &= \sum_{i \in \text{Pred}(j)} \bar{n}_{ij} = \sum_{i \in \text{Pred}(j)} \bar{n}_i p_{ij}^{\text{ext}} \\ &= \sum_{i=1}^n \bar{n}_i p_{ij}^{\text{ext}} = \sum_{i=2}^n \bar{n}_i p_{ij}^{\text{ext}} + \underbrace{\bar{n}_1 p_{1j}^{\text{ext}}}_{\sigma_j^{\text{in}}} \\ &= \sum_{i=2}^n \bar{n}_i p_{ij}^{\text{ext}} + \sigma_j^{\text{in}} = \sum_{i=2}^{n-1} \bar{n}_i p_{ij}^{\text{ext}} + \underbrace{\bar{n}_n p_{nj}^{\text{ext}}}_{0} + \sigma_j^{\text{in}} \\ &= \sum_{i \in \mathcal{V}} \bar{n}_i p_{ij}^{\text{ext}} + \sigma_j^{\text{in}} \text{ for each } j \in \mathcal{V} \end{aligned} \quad (\text{A4})$$

But we know from Eq. (5) that $p_{ij}^{\text{ext}} = (1 - \alpha_i) p_{ij}$ when $i, j \notin \{1, n\}$, which is injected in Eq. (A4). Recalling also that $\bar{n}_i \alpha_i = \sigma_i^{\text{out}}$, we obtain

$$\begin{aligned} \bar{n}_j &= \sum_{i \in \mathcal{V}} \bar{n}_i p_{ij}^{\text{ext}} + \sigma_j^{\text{in}} = \sum_{i \in \mathcal{V}} \bar{n}_i (p_{ij} - \alpha_i p_{ij}) + \sigma_j^{\text{in}} \\ &= \sum_{i \in \mathcal{V}} \bar{n}_i p_{ij} - \sum_{i \in \mathcal{V}} \underbrace{\bar{n}_i \alpha_i}_{\sigma_i^{\text{out}}} p_{ij} + \sigma_j^{\text{in}} \\ &= \sum_{i \in \mathcal{V}} \bar{n}_i p_{ij} - \sum_{i \in \mathcal{V}} \sigma_i^{\text{out}} p_{ij} + \sigma_j^{\text{in}} \\ &= \sum_{i \in \mathcal{V}} (\bar{n}_i - \sigma_i^{\text{out}}) p_{ij} + \sigma_j^{\text{in}} \text{ for each } j \in \mathcal{V} \end{aligned} \quad (\text{A5})$$

or, in matrix form, where $\bar{\mathbf{n}}$ is the $(n-2) \times 1$ column vector containing the $\bar{n}_i$,

$$\bar{\mathbf{n}} = \mathbf{P}^T \bar{\mathbf{n}} - \mathbf{P}^T \sigma_{\text{out}} + \sigma_{\text{in}}$$

We thus have to solve

$$(\mathbf{I} - \mathbf{P}^T) \bar{\mathbf{n}} = \sigma_{\text{in}} - \mathbf{P}^T \sigma_{\text{out}} \quad (\text{A6})$$

¹⁵ Recall that all nodes with label $i \in \mathcal{V}$ keep the same index in $\mathcal{V}_{\text{ext}}$.

and then compute α from

$$\alpha = \sigma_{\text{out}} \div \bar{\mathbf{n}} \quad (\text{A7})$$

where $\div$ denotes the elementwise division. It can be observed that only the output nodes $j \in \text{Out}$ are likely to have a positive killing probability α_j , and thus also a positive weight w_j (see Eq. (A3)). The probability (and weight) of all other nodes $\mathcal{V} \setminus \text{Out}$ is set to 0.

As $\mathbf{P}^T$ is rank-deficient, the solution to Eq. (A6) is the sum of a particular solution plus any vector in the null space of $(\mathbf{I} - \mathbf{P}^T)$ [67–69]. As the null space of $(\mathbf{I} - \mathbf{P}^T)$ is spanned by π , the equilibrium distribution of the Markov chain on G with transition matrix $\mathbf{P}$, we obtain

$$\bar{\mathbf{n}} = (\mathbf{I} - \mathbf{P}^T)^+ (\sigma_{\text{in}} - \mathbf{P}^T \sigma_{\text{out}}) + \mu \pi \quad (\text{A8})$$

where $+$ denotes the Moore–Penrose pseudoinverse and μ is the non-negative “persistence parameter” [4]. To ensure that $\bar{\mathbf{n}} \geq \sigma_{\text{out}}$ (so that there is a feasible solution), the parameter μ has to satisfy the following inequality [4],

$$\mu \geq \max_i \left\{ \frac{\sigma_i^{\text{out}} - \mathbf{e}_i^T (\mathbf{I} - \mathbf{P}^T)^+ (\sigma_{\text{in}} - \mathbf{P}^T \sigma_{\text{out}})}{\pi_i} \right\} \quad (\text{A9})$$

Any value of μ satisfying this constraint provides valid flows.

Therefore the transition matrix of the extended graph is provided by Eq. (5) where α is given by the previous Eqs. (A7)–(A8) and is pre-computed. Then, the value of $\mathbf{w}$ can be obtained from α by using Eq. (A3). The algorithm computing $\bar{\mathbf{n}}$ and α is detailed in Algorithm 1 of [5] (page 99).

Appendix B The standard RSP model: some quantities of interest

For the sake of completeness, this appendix summarizes some important quantities that are derived from the standard randomized shortest paths framework, and is largely inspired by [12,34] (see these references for a more complete summary). Interestingly, almost all the quantities of interest can be computed by taking the partial derivative of the partition function (see [1–3,10,53] for details).

Auxiliary matrix

Let us first introduce an important matrix. The elements of this **auxiliary matrix** $\mathbf{W}$ are

$$w_{ij} = p_{ij} \exp[-\theta c_{ij}] \quad \text{or, in matrix form,} \quad \mathbf{W} = \mathbf{P} \circ \exp[-\theta \mathbf{C}] \quad (\text{B10})$$

where we recall that $\mathbf{C}$ is the cost matrix and $\circ$ is the elementwise (Hadamard) product. Notably, the optimal path

probabilities (see Eq. (7)) can be expressed in terms of this matrix as

$$P^*(\wp) = \frac{\tilde{w}(\wp)}{\sum_{\wp' \in \mathcal{P}_{1n}} \tilde{w}(\wp')} \quad (\text{B11})$$

with $\tilde{w}(\wp) = \prod_{\tau=1}^{\ell} w_{\wp(\tau-1)\wp(\tau)} = \prod_{\tau=1}^{\ell} p_{\wp(\tau-1)\wp(\tau)} \exp[-\theta c_{\wp(\tau-1)\wp(\tau)}] = \tilde{\pi}(\wp) \exp[-\theta \tilde{c}(\wp)]$, that is, the product of the corresponding elements of the matrix along the edges of the path.

Fundamental matrix

It will be shown that the partition function $\mathcal{Z}$ can be computed in closed form from this auxiliary matrix (again, see, e.g., [1,2,10] for details). To this end, let us further introduce the **fundamental matrix** of the randomized shortest paths system,

$$\mathbf{Z} = \mathbf{I} + \mathbf{W} + \mathbf{W}^2 + \dots = (\mathbf{I} - \mathbf{W})^{-1} \quad (\text{B12})$$

This expression sums up contributions of different paths lengths, starting from zero-length paths ($\mathbf{I}$). Therefore, in the case of our extended graph, $z_{11} = 1$ because node 1 has no predecessor on the extended graph [10].

Partition function and forward/backward variables

Then, the **partition function** can be computed thanks to [1–3,10]

$$\mathcal{Z} = \sum_{\wp \in \mathcal{P}_{1n}} \tilde{\pi}(\wp) \exp[-\theta \tilde{c}(\wp)] = \sum_{\wp \in \mathcal{P}_{1n}} \tilde{w}(\wp) = [\mathbf{Z}]_{1n} = z_{1n} \quad (\text{B13})$$

More generally [70], the following **forward** and **backward** variables can be defined,

$$z_{1i} = \sum_{\wp \in \mathcal{P}_{1i}} \tilde{w}(\wp) \quad \text{and} \quad z_{jn} = \sum_{\wp \in \mathcal{P}_{jn}} \tilde{w}(\wp) \quad (\text{B14})$$

where $\mathcal{P}_{1i}$ is the set of paths starting in node 1 and ending in node i and $\mathcal{P}_{jn}$ is the set of paths starting in node j and ending in node n .

Free energy potential

It can be shown [1] that the **free energy potential** at the optimal probability distribution is obtained by replacing the probability distribution $P(\cdot)$ with the optimal distribution $P^*(\cdot)$ provided by Eq. (7) in the objective function (Eq. (6)). The result is

$$\phi_{1n} \triangleq \phi(P^*) = -T \log \mathcal{Z} = -\frac{1}{\theta} \log z_{1n} \quad (\text{B15})$$

More generally, we define the free energy potential from an arbitrary node i as $\phi_{in} = -\frac{1}{\theta} \log z_{in}$. This quantity is

an interesting distance measure between nodes which was investigated in [1, 10].

Edge flows

It further turns out that the directed **flow** in (i, j) can be obtained from Eq. (B12) by computing the partial derivative of the logarithm of the partition function with respect to the edge costs [2, 71],

$$\bar{n}_{ij} = -\frac{1}{\theta} \frac{\partial \log \mathcal{Z}}{\partial c_{ij}} = \frac{z_{1i} p_{ij}^{\text{ext}} \exp[-\theta c_{ij}] z_{jn}}{z_{1n}} = \frac{z_{1i} w_{ij} z_{jn}}{z_{1n}} \quad (\text{B16})$$

for any existing link (i, j) ; $\bar{n}_{ij} = 0$ on missing links. In addition, the matrix $\mathbf{N} = (\bar{n}_{ij})$ will contain these edge flows.

Moreover, since only the first row and the last column of $\mathbf{Z}$ are needed in our situation, two systems of linear equations can be solved instead of matrix inversion in Eq. (B12).

Number of visits to nodes

The **expected number of visits** to a node $j \neq 1$ can now be readily computed¹⁶ from the edge flows,

$$\bar{n}_j = \sum_{i=1}^n \bar{n}_{ij} = \frac{z_{1j} z_{jn}}{z_{1n}} \quad (\text{B17})$$

In addition, $\bar{n}_1 = 1$.

Expected cost.

The total **expected cost** until reaching the target [2] is

$$\langle \tilde{c} \rangle = \mathbf{e}^T (\mathbf{N} \circ \mathbf{C}) \mathbf{e} \quad (\text{B18})$$

where the missing links have a 0 contribution. Intuitively, it corresponds to the sum of the expected number of passages through each edge times the cost of following the edge.

Optimal local policy (optimal transition probabilities).

Finally, the optimal transition probability of following the edge (i, j) (the optimal, local, policy), minimizing the objective function (Eq. (6)), is

$$p_{ij}^* = \frac{\bar{n}_{ij}}{\bar{n}_i} = \frac{z_{jn}}{z_{in}} p_{ij} \exp[-\theta c_{ij}] \quad (\text{B19})$$

This equation is the counterpart of Eq. (7) at the node level, instead of path level. These transition probabilities define a biased random walk on the graph—the random walker is “attracted” by the target node n . Moreover, these transition probabilities do not depend on the source node and correspond to the optimal randomized policy minimizing free energy for the given temperature T .

¹⁶ Recall that node 1 has no predecessor and node n no successor on the extended graph.

Appendix C Derivation of the algorithm computing the optimal randomized policy

This appendix section derives the algorithm allowing to compute the randomized policy for the relative entropy-regularized optimal transport on the extended graph G_{ext} , and is directly inspired by [12]. The first part introduces the augmented costs whereas the second part details the iterative procedure allowing to solve the problem.

C.1 The Lagrange function

Given that $\bar{n}_{ij} = \sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \eta((i, j) \in \wp)$ with $\eta((i, j) \in \wp)$ being the number of times edge (i, j) appears on path $\wp$, the Lagrange function (Eq. (11)) defined on the extended graph G_{ext} becomes

$$\begin{aligned} \mathcal{L}(\mathbf{P}, \boldsymbol{\lambda}) = & \underbrace{\sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \tilde{c}(\wp)}_{\text{free energy, } \phi(\mathbf{P})} + T \sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \log \left(\frac{P(\wp)}{\tilde{\pi}(\wp)} \right) \\ & + \mu \left(\sum_{\wp \in \mathcal{P}_{1n}} P(\wp) - 1 \right) \\ & + \sum_{i \in \mathcal{I}_n} \lambda_i^{\text{in}} \left[\sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \eta((1, i) \in \wp) - \sigma_i^{\text{in}} \right] \\ & + \sum_{j \in \mathcal{O}_{ut}} \lambda_j^{\text{out}} \left[\sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \eta((j, n) \in \wp) - \sigma_j^{\text{out}} \right] \end{aligned} \quad (\text{C20})$$

Note that the objective function to be minimized is convex and the equality constraints are all linear. The Lagrange function can be rearranged as

$$\begin{aligned} \mathcal{L}(\mathbf{P}, \boldsymbol{\lambda}) = & \sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \left[\underbrace{\tilde{c}(\wp) + \sum_{i \in \mathcal{I}_n} \lambda_i^{\text{in}} \eta((1, i) \in \wp) + \sum_{j \in \mathcal{O}_{ut}} \lambda_j^{\text{out}} \eta((j, n) \in \wp)}_{\tilde{c}'(\wp)} \right] \\ & + T \sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \log \left(\frac{P(\wp)}{\tilde{\pi}(\wp)} \right) + \mu \left(\sum_{\wp \in \mathcal{P}_{1n}} P(\wp) - 1 \right) \\ & - \sum_{i \in \mathcal{I}_n} \lambda_i^{\text{in}} \sigma_i^{\text{in}} - \sum_{j \in \mathcal{O}_{ut}} \lambda_j^{\text{out}} \sigma_j^{\text{out}} \\ = & \underbrace{\sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \tilde{c}'(\wp)}_{\text{free energy } \phi'(\mathbf{P}) \text{ based on augmented costs}} + T \sum_{\wp \in \mathcal{P}_{1n}} P(\wp) \log \left(\frac{P(\wp)}{\tilde{\pi}(\wp)} \right) \\ & + \mu \left(\sum_{\wp \in \mathcal{P}_{1n}} P(\wp) - 1 \right) \\ & - \underbrace{\sum_{i \in \mathcal{I}_n} \lambda_i^{\text{in}} \sigma_i^{\text{in}} - \sum_{j \in \mathcal{O}_{ut}} \lambda_j^{\text{out}} \sigma_j^{\text{out}}}_{\text{does not depend on } \mathbf{P}(\cdot)} \end{aligned} \quad (\text{C21})$$

where $\tilde{c}'(\wp) = \sum_{\tau=1}^t c'_{s(\tau-1)s(\tau)}$ is the total cumulated augmented cost along path $\wp$ with the original costs c_{ij} being replaced by the *augmented costs* on G_{ext} ,

$$c'_{ij} = \begin{cases} c_{ij}^{\text{ext}} + \lambda_j^{\text{in}} & \text{when } i = 1 \text{ and } j \in \mathcal{In} \\ c_{ij}^{\text{ext}} + \lambda_i^{\text{out}} & \text{when } i \in \mathcal{Out} \text{ and } j = n \\ c_{ij}^{\text{ext}} & \text{otherwise} \end{cases}$$

because the initial costs are equal to zero for edges starting in node 1 and ending in $\mathcal{In}$. The same holds for edges starting in $\mathcal{Out}$ and ending in node n . $\mathbf{C}'$ will be the matrix containing these augmented costs. Besides, $\phi'(\mathbf{P})$ is the free energy depending on these augmented costs. This justifies Eq. (14).

Thus, in Eq. (C21), everything happens as if the costs have been redefined by taking into account the Lagrange parameters. These Lagrange parameters can therefore be interpreted as additional costs necessary to satisfy the equality constraints. We now have to find the Lagrange parameters λ by using Lagrangian duality.

C.2 Exploiting Lagrangian duality

From Eq. (12), we have to compute the dual function and then maximize the dual function in terms of the Lagrange parameters.

C.2.1 Computing the dual function

Let us first compute $\mathbf{P}^* = \arg \min_{\{\mathbf{P}(\wp)\}_{\wp \in \mathcal{P}_{1n}}} \mathcal{L}(\mathbf{P}, \lambda)$ satisfying the sum-to-one constraint. From Eq. (C21), as for the RSP (see Sect. 3.1), the primal can be solved easily and exactly; it provides a Gibbs-Boltzmann distribution (Eq. (7)), but this time in terms of the *augmented costs*, $\tilde{c}'(\wp)$, introduced in Eq. (14). It simply computes the new probability distribution $\mathbf{P}^*(\cdot)$ based on these augmented costs.

Once $\mathbf{P}^*$ has been computed, as shown in Sect. 3.1, Eq. (B16), $\phi'(\mathbf{P}^*) = -T \log \mathcal{Z}'$ on G_{ext} , and the corresponding dual function (Eq. (C21)) can be rewritten in terms of the partition function (Eq. (B13)) on G_{ext} as

$$\begin{aligned} \mathcal{L}(\mathbf{P}^*, \lambda) &= \phi'(\mathbf{P}^*) - \sum_{i \in \mathcal{In}} \lambda_i^{\text{in}} \sigma_i^{\text{in}} - \sum_{j \in \mathcal{Out}} \lambda_j^{\text{out}} \sigma_j^{\text{out}} \\ &= -T \log \mathcal{Z}' - \sum_{i \in \mathcal{In}} \lambda_i^{\text{in}} \sigma_i^{\text{in}} - \sum_{j \in \mathcal{Out}} \lambda_j^{\text{out}} \sigma_j^{\text{out}} \end{aligned}$$

which provides Eq. (13).

C.2.2 Maximizing the dual function

For computing the maximum of the dual function,¹⁷ we will use a block coordinate ascend [39,72] procedure optimizing sequentially with respect to the $\{\lambda_k^{\text{in}}\}_{k \in \mathcal{In}}$ and then with respect to the $\{\lambda_k^{\text{out}}\}_{k \in \mathcal{Out}}$. As the dual function is concave, the block coordinate ascend procedure converges to the global maximum.

Computing Lagrange parameters associated with input nodes.

Let us start with the $\{\lambda_k^{\text{in}}\}_{k \in \mathcal{In}}$. Recalling that $\partial_{c_{ij}}(-T \log \mathcal{Z}) = \bar{n}_{ij}$ (see Eq. (B16)), the gradient of the dual function with respect to λ_k^{in} can be computed for each node $k \in \mathcal{In}$,

$$\begin{aligned} \frac{\partial \mathcal{L}(\mathbf{P}^*, \lambda)}{\partial \lambda_k^{\text{in}}} &= \frac{\partial(-T \log \mathcal{Z}')}{\partial \lambda_k^{\text{in}}} - \sigma_k^{\text{in}} \\ &= \frac{\partial(-T \log \mathcal{Z}')}{\partial c'_{1k}} \frac{\partial c'_{1k}}{\partial \lambda_k^{\text{in}}} - \sigma_k^{\text{in}} = \bar{n}_{1k} - \sigma_k^{\text{in}} \end{aligned} \quad (\text{C22})$$

where we used the definition of the augmented costs in Eq. (14). Expressing the fact that the gradient must cancel at the optimum simply reduces to the equality constraint of Eq. (8), which is common in maximum-entropy problems [8,73] and was already observed in, e.g., [12],

$$\bar{n}_{1k} = \sigma_k^{\text{in}} \quad \text{for each } k \in \mathcal{In} \quad (\text{C23})$$

Now, from Eq. (B16), $\bar{n}_{1k} = z'_{11} p_{1k}^{\text{ext}} \exp[-\theta c'_{1k}] z'_{kn} / z'_{1n}$ and, as node 1 has no incoming edge, $z'_{11} = 1$ (see Eq. (B12)). Moreover, again from Eq. (B12), $(\mathbf{I} - \mathbf{W}')\mathbf{Z}' = \mathbf{Z}' - \mathbf{W}'\mathbf{Z}' = \mathbf{I}$, and taking element 1, n provides

$$\begin{aligned} z'_{1n} &= \sum_{k \in \text{Succ}(1)} w'_{1k} z'_{kn} = \sum_{k \in \text{Succ}(1)} p_{1k}^{\text{ext}} \exp[-\theta c'_{1k}] z'_{kn} \\ &= \sum_{k \in \mathcal{In}} p_{1k}^{\text{ext}} \exp[-\theta \lambda_k^{\text{in}}] z'_{kn} \end{aligned} \quad (\text{C24})$$

where $\text{Succ}(1)$ is the set of successor nodes of node 1, which corresponds to the set of input nodes $\mathcal{In}$. Thus, from the previous discussion and Eq. (B16),

$$\begin{aligned} \bar{n}_{1k} &= \frac{z'_{11} p_{1k}^{\text{ext}} \exp[-\theta c'_{1k}] z'_{kn}}{z'_{1n}} \\ &= \frac{p_{1k}^{\text{ext}} \exp[-\theta \lambda_k^{\text{in}}] z'_{kn}}{\sum_{k' \in \mathcal{In}} p_{1k'}^{\text{ext}} \exp[-\theta \lambda_{k'}^{\text{in}}] z'_{k'n}} \end{aligned} \quad (\text{C25})$$

because $z'_{11} = 1$. We observe that $\sum_{k=1}^n \bar{n}_{1k} = 1$, meaning that the input flow is 1, as it should be.

¹⁷ Recall that the dual function is concave; see for instance [41].

Note from Eq. (B14) that, as edge $(1, k)$ ($k \neq 1$) does not appear on any path in $\mathcal{P}_{kn}$ by construction of the extended graph, the backward variables z'_{kn} with $k > 1$ do not depend on the λ_k^{in} . Thus, from Eq. (C23), for updating the λ_k^{in} , we have to solve

$$\frac{p_{1k}^{\text{ext}} \exp[-\theta \lambda_k^{\text{in}} z'_{kn}]}{\sum_{k' \in \mathcal{I}_n} p_{1k'}^{\text{ext}} \exp[-\theta \lambda_{k'}^{\text{in}} z'_{k'n}]} = \sigma_k^{\text{in}} \quad \text{for all } k \in \mathcal{I}_n \quad (\text{C26})$$

These equations are invariant when adding a constant to the Lagrange parameters. Therefore, we will impose the condition $\sum_{k \in \mathcal{I}_n} \lambda_k^{\text{in}} \sigma_k^{\text{in}} = 0$, with the consequence that we have the following nice property: the expected augmented cost from source node 1 to target node n will be equal to the real expected cost ($\langle \tilde{c}' \rangle = \langle \tilde{c} \rangle$).

It was shown in the appendix of [74] that the closed-form solution¹⁸ of this system of logistic function equations is $\lambda_k^{\text{in}} = -\frac{1}{\theta} \log \left(\frac{\sigma_k^{\text{in}}}{p_{1k}^{\text{ext}} z'_{kn}} \right) - \sum_{l \in \mathcal{I}_n} \sigma_l^{\text{in}} \left[-\frac{1}{\theta} \log \left(\frac{\sigma_l^{\text{in}}}{p_{1l}^{\text{ext}} z'_{ln}} \right) \right]$ and because we know that $p_{1l}^{\text{ext}} = \sigma_l^{\text{in}}$ for $l \in \mathcal{I}_n$, we further obtain

$$\begin{aligned} \lambda_k^{\text{in}} &= \frac{1}{\theta} \left(\log z'_{kn} - \sum_{l \in \mathcal{I}_n} \sigma_l^{\text{in}} \log z'_{ln} \right) \\ &= \sum_{l \in \mathcal{I}_n} \sigma_l^{\text{in}} (\phi_l - \phi_k) \quad \text{for } k \in \mathcal{I}_n \end{aligned} \quad (\text{C27})$$

where we used Eq. (B15) and $\sum_{l \in \mathcal{I}_n} \sigma_l^{\text{in}} = 1$. It can be observed that the $\log z'_{kn}$ are centered with respect to their weighted mean (with weights provided by σ_k^{in}). This last expression proves the first part of Eq. (15).

Computing Lagrange parameters associated with output nodes.

Symmetrically to Eq. (C25), we have for the Lagrange parameters $\{\lambda_l^{\text{out}}\}_{l \in \mathcal{O}_{ut}}$ associated with the edges incident to the target node n ,

$$\begin{aligned} \bar{n}_{ln} &= \frac{z'_{1l} p_{ln}^{\text{ext}} \exp[-\theta c'_{ln} z'_{nn}]}{z'_{1n}} \\ &= \frac{z'_{1l} p_{ln}^{\text{ext}} \exp[-\theta \lambda_l^{\text{out}}]}{\sum_{l' \in \mathcal{O}_{ut}} z'_{1l'} p_{ln}^{\text{ext}} \exp[-\theta \lambda_{l'}^{\text{out}}]} \end{aligned} \quad (\text{C28})$$

because $z'_{nn} = 1$. As the target node is absorbing, $\sum_{l=1}^n \bar{n}_{ln} = 1$ is verified. Setting $\bar{n}_{ln} = \sigma_l^{\text{out}}$ and proceeding in the same way as before provides, for the Lagrange parameters associ-

ated with the nodes in $\mathcal{O}_{ut}$,

$$\begin{aligned} \lambda_l^{\text{out}} &= \frac{1}{\theta} \left(\log z'_{1l} - \log \left(\frac{\sigma_l^{\text{out}}}{p_{ln}^{\text{ext}}} \right) \right. \\ &\quad \left. - \sum_{k \in \mathcal{O}_{ut}} \sigma_k^{\text{out}} \left[\log z'_{1k} - \log \left(\frac{\sigma_k^{\text{out}}}{p_{kn}^{\text{ext}}} \right) \right] \right) \quad \text{for } l \in \mathcal{O}_{ut} \end{aligned} \quad (\text{C29})$$

We readily observe that $\sum_{l \in \mathcal{O}_{ut}} \lambda_l^{\text{out}} \sigma_l^{\text{out}} = 0$. This further proves the second part of Eq. (15).

These Lagrange parameters, for both input and output nodes, are recomputed in turn until convergence. They are then used in order to calculate the final augmented cost values and the corresponding fundamental matrix, allowing to obtain the quantities of interest, in particular, the optimal transition probabilities matrix (the policy).

References

1. Kivimäki I, Shimbo M, Saelens M (2014) Developments in the theory of randomized shortest paths with a comparison of graph node distances. *Phys A* 393:600–616. <https://doi.org/10.1016/j.physa.2013.09.016>
2. Saelens M, Achbany Y, Fouss F, Yen L (2009) Randomized shortest-path problems: two related models. *Neural Comput* 21(8):2363–2404. <https://doi.org/10.1162/neco.2009.11-07-643>
3. Yen L, Mantrach A, Shimbo M, Saelens M (2008) A family of dissimilarity measures between nodes generalizing both the shortest-path and the commute-time distances. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*, pp 785–793. <https://doi.org/10.1145/1401890.1401984>
4. Guex G (2016) Interpolating between random walks and optimal transportation routes: flow with multiple sources and targets. *Phys A* 450:264–277. <https://doi.org/10.1016/j.physa.2015.12.117>
5. Guex G, Kivimäki I, Saelens M (2019) Randomized optimal transport on a graph: framework and new distance measures. *Netw Sci* 7(1):88–122. <https://doi.org/10.1017/nws.2018.29>
6. Cuturi M (2013) Sinkhorn distances: lightspeed computation of optimal transport. In: *Advances in Neural Information Processing Systems 26: Proceedings of the NIPS '13 Conference*, pp. 2292–2300. MIT Press, Cambridge
7. Erlander S, Stewart N (1990) The gravity model in transportation analysis. Theory and extensions. VSP International Science Publishers, Utrecht
8. Kapur JN, Kesavan HK (1992) Entropy optimization principles with applications. Academic Press, Cambridge
9. Wilson AG (1970) Entropy in urban and regional modelling. Routledge, New York
10. Francoise K, Kivimäki I, Mantrach A, Rossi F, Saelens M (2017) A bag-of-paths framework for network data analysis. *Neural Netw* 90:90–111. <https://doi.org/10.1016/j.neunet.2017.03.010>
11. Mantrach A, Yen L, Callut J, Francoise K, Shimbo M, Saelens M (2010) The sum-over-paths covariance kernel: a novel covariance between nodes of a directed graph. *IEEE Trans Pattern Anal Mach Intell* 32(6):1112–1126. <https://doi.org/10.1109/TPAMI.2009.78>
12. Courtain S, Leleux P, Kivimäki I, Guex G, Saelens M (2020) Randomized shortest paths with net flows and capacity constraints. *Inform Sci*. <https://doi.org/10.1016/j.ins.2020.10.005>

¹⁸ It can easily be verified by injecting the solution into the system of equations.

13. Akamatsu T (1996) Cyclic flows, Markov process and stochastic traffic assignment. *Transp Res B* 30(5):369–386. [https://doi.org/10.1016/0191-2615\(96\)00003-3](https://doi.org/10.1016/0191-2615(96)00003-3)
14. Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows: theory, algorithms, and applications. Prentice Hall, Hoboken
15. Ford LR, Fulkerson DR (1956) Maximal flow through a network. *Can J Math* 8:399–404. <https://doi.org/10.4153/CJM-1956-045-5>
16. Lempitsky V, Boykov Y (2007) Global optimization for shape fitting. In: Proceedings of the 12th IEEE International Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '07), IEEE, pp 1–8
17. Yuan J, Bae E, Tai X-C (2010) A study on continuous max-flow and min-cut approaches. In: Proceedings of the 15th IEEE International Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '10), IEEE, pp 2217–2224
18. Kolisch R, Heimerl C (2012) An efficient metaheuristic for integrated scheduling and staffing it projects based on a generalized minimum cost flow network. *Nav Res Logist* 59(2):111–127. <https://doi.org/10.1002/nav.21476>
19. Díaz-Parra O, Bernábe Loranca JAR-VB, Fuentes-Penna A, Barrera-Cámara RA (2014) A survey of transportation problems. *J Appl Math*. <https://doi.org/10.1155/2014/848129>
20. Kacher Y, Singh P (2021) A comprehensive literature review on transportation problems. *Int J Appl Comput Math* 7(5):1–49. <https://doi.org/10.1007/s40819-021-01134-y>
21. Flood MM (1954) Application of transportation theory to scheduling a military tanker fleet. *J Oper Res Soc Am* 2(2):150–162. <https://doi.org/10.1287/opre.2.2.150>
22. Helme MP (1992) Reducing air traffic delay in a space-time network. In: Proceedings of the 4th IEEE International Conference on Systems, Man, and Cybernetics (SMC '92), pp 236–242
23. Yüceer U, Özakç A (2010) A truck loading problem. *Comput Indus Eng* 58(4):766–773. <https://doi.org/10.1016/j.cie.2010.02.008>
24. Peyré G, Cuturi M (2019) Computational optimal transport. *Found Trends Mach Learn* 11(5–6):355–607. <https://doi.org/10.1561/22000000073>
25. Jungnickel D (2013) Graphs, networks, and algorithms, 4th edn. Springer, Berlin
26. Williamson D (2019) Network flow algorithms. Cambridge University Press, Cambridge
27. Wilson AG (1967) A statistical theory of spatial distribution models. *Transp Res* 1(3):253–269. [https://doi.org/10.1016/0041-1647\(67\)90035-4](https://doi.org/10.1016/0041-1647(67)90035-4)
28. Reggiani A, Schintler L, Czamanski D, Patuelli R (2021) Handbook on entropy. Complexity and spatial dynamics. Edward Elgar Publishing, Cheltenham
29. Chen Y, Georgiou TT, Pavon M, Tannenbaum A (2017) Robust transport over networks. *IEEE Trans Autom Control* 62(9):4675–4682. <https://doi.org/10.1109/TAC.2016.2626796>
30. Chen Y, Georgiou TT, Pavon M, Tannenbaum A (2017) Efficient robust routing for single commodity network flows. *IEEE Trans Autom Control* 63(7):2287–2294. <https://doi.org/10.1109/TAC.2017.2763418>
31. Pavon M, Ticozzi F (2010) Discrete-time classical and quantum Markovian evolutions: maximum entropy problems on path space. *J Math Phys* 51(4):042104. <https://doi.org/10.1063/1.3372725>
32. Chen Y, Georgiou TT, Pavon M (2021) Stochastic control liaisons: Richard Sinkhorn meets Gaspard Monge on a Schrodinger bridge. *SIAM Rev* 63(2):249–313. <https://doi.org/10.1137/20M1339982>
33. Arqué F, Uribe CA, Ocampo-Martinez C (2022) Approximate Wasserstein attraction flows for dynamic mass transport over networks. *Automatica* 143:110432. <https://doi.org/10.1016/j.automatica.2022.110432>
34. Leleux P, Courtain S, Guex G, Saerens M (2021) Sparse randomized shortest paths routing with Tsallis divergence regularization. *Data Mining Knowledge Discov*. <https://doi.org/10.1007/s10618-021-00742-y>
35. Jaynes ET (1957) Information theory and statistical mechanics. *Phys Rev* 106:620–630. <https://doi.org/10.1103/PhysRev.106.620>
36. Peliti L (2011) Statistical mechanics in a nutshell. Princeton University Press, Princeton
37. Reichl LE (1998) A modern course in statistical physics, 2nd edn. Wiley, Hoboken
38. Jebara T (2012) Machine learning: discriminative and generative, vol 755. Springer, Berlin
39. Bertsekas DP (1999) Nonlinear Programm, 2nd edn. Athena Scientific, Nashua
40. Culioli J (2012) Introduction a L'optimisation. Ellipses, Paris
41. Griva I, Nash S, Sofer A (2008) Linear and nonlinear optimization, 2nd edn. SIAM, Philadelphia
42. Arrow K, Hurwicz L, Uzawa H (1958) Studies in linear and nonlinear programming. Stanford University Press, Palo Alto
43. Villani C (2003) Topics in optimal transportation. American Mathematical Society, Providence
44. Villani C (2008) Optimal transport: old and new. Springer, Berlin
45. Brin S, Page L (1998) The anatomy of a large-scale hypertextual Web search engine. *Computer Netw ISDN Syst* 30(1–7):107–117. [https://doi.org/10.1016/S0169-7552\(98\)00110-X](https://doi.org/10.1016/S0169-7552(98)00110-X)
46. Page L, Brin S, Motwani R, Winograd T (1998) The PageRank citation ranking: Bringing order to the web. Technical report 1999-66, Computer Science Department, Stanford InfoLab, Stanford University, USA. <http://ilpubs.stanford.edu:8090/422/>
47. Gori M, Pucci A (2007) ItemRank: a random-walk based scoring algorithm for recommender engines. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI '07), pp 2766–2771
48. Tong H, Faloutsos C, Pan J-Y (2008) Random walk with restart: fast solutions and applications. *Knowl Inf Syst* 14(3):327–346. <https://doi.org/10.1007/s10115-007-0094-2>
49. Guex G, Courtain S, Saerens M (2020) Covariance and correlation kernels on a graph in the generalized bag-of-paths formalism. *J Complex Netw*. <https://doi.org/10.1093/comnet/cnaa026>
50. Sommer F, Fouss F, Saerens M (2016) Comparison of graph node distances on clustering tasks. In: Proceedings of the International Conference on Artificial Neural Networks (ICANN 2016). Lecture Notes in Computer Science, vol 9886, Springer, New York, pp 192–201. https://doi.org/10.1007/978-3-319-44778-0_23
51. Fouss F, Pirotte A, Renders J-M, Saerens M (2007) Random-walk computation of similarities between nodes of a graph, with application to collaborative recommendation. *IEEE Trans Knowl Data Eng* 19(3):355–369. <https://doi.org/10.1109/TKDE.2007.46>
52. Saerens M, Fouss F, Yen L, Dupont P (2004) The principal components analysis of a graph, and its relationships to spectral clustering. In: Proceedings of the 15th European Conference on Machine Learning (ECML '04). Lecture Notes in Artificial Intelligence, vol. 3201, pp 371–383. Springer, Berlin. https://doi.org/10.1007/978-3-540-30115-8_35
53. Fouss F, Saerens M, Shimbo M (2016) Algorithms and models for network data and link analysis. Cambridge University Press, Cambridge
54. Tang L, Liu H (2009) Relational learning via latent social dimensions. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09), pp 817–826. <https://doi.org/10.1145/1557019.1557109>
55. Tang L, Liu H (2009) Scalable learning of collective behavior based on sparse social dimensions. In: Proceedings of the ACM Conference on Information and Knowledge Management (CIKM '09), pp 1107–1116
56. Tang L, Liu H (2010) Toward predicting collective behavior via social dimension extraction. *IEEE Intell Syst* 25(4):19–25. <https://doi.org/10.1109/MIS.2010.36>

57. Zhang D, Mao R (2008) Classifying networked entities with modularity kernels. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM 2008), ACM, pp 113–122. <https://doi.org/10.1145/1458082.1458100>
58. Zhang D, Mao R (2008) A new kernel for classification of networked entities. In: Proceedings of 6th International Workshop on Mining and Learning with Graphs, Helsinki, Finland
59. Macskassy SA, Provost F (2007) Classification in networked data: a toolkit and a univariate case study. *J Mach Learn Res* 8:935–983
60. Dua D, Graff C (2017) UCI machine learning repository. <http://archive.ics.uci.edu/ml>
61. Yen L, Fouss F, Decaestecker C, Francq P, Saelens M (2009) Graph nodes clustering with the sigmoid commute-time kernel: a comparative study. *Data Knowl Eng* 68(3):338–361. <https://doi.org/10.1016/j.datak.2008.10.006>
62. Chang C-C, Lin C-J (2011) LIBSVM: a library for support vector machines. *ACM Trans Intell Syst Technol* 2:27–12727. <https://doi.org/10.1145/1961189.1961199>
63. Borg I, Groenen P (1997) Modern multidimensional scaling: theory and applications. Springer, Berlin
64. Schölkopf B, Smola A (2002) Learning with kernels. MIT Press, Cambridge
65. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 7:1–30
66. Doyle PG, Snell JL (1984) Random walks and electric networks, 1st edn. The Mathematical Association of America
67. Schott J (2005) Matrix analysis for statistics, 2nd edn. Wiley, Hoboken
68. Searle S (1982) Matrix algebra useful for statistics. Wiley, Hoboken
69. Serre D (2002) Matrices. Theory and applications. Springer, Berlin
70. García-Díez S, Fouss F, Shimbo M, Saelens M (2011) A sum-over-paths extension of edit distances accounting for all sequence alignments. *Pattern Recogn* 44(6):1172–1182. <https://doi.org/10.1016/j.patcog.2010.11.020>
71. Kivimäki I, Leichot B, Saramäki J, Saelens M (2016) Two betweenness centrality measures based on randomized shortest paths. *Sci Rep* 6:19668
72. Lange K (2013) Optimization, 2nd edn. Springer, Berlin
73. Kapur JN (1989) Maximum-entropy models in science and engineering. Wiley, Hoboken
74. Leichot B, Guex G, Kivimäki I, Saelens M (2018) A constrained randomized shortest-paths framework for optimal exploration. ArXiv. <https://doi.org/10.48550/arXiv.1807.04551>

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.