

HYDRA: A METHOD FOR STRAIN-MINIMIZING HYPERBOLIC EMBEDDING OF NETWORK- AND DISTANCE-BASED DATA

MARTIN KELLER-RESSEL AND STEPHANIE NARGANG

ABSTRACT. We introduce *hydra* (hyperbolic distance recovery and approximation), a new method for embedding network- or distance-based data into hyperbolic space. We show mathematically that *hydra* satisfies a certain optimality guarantee: It minimizes the ‘hyperbolic strain’ between original and embedded data points. Moreover, it recovers points exactly, when they are located on a hyperbolic submanifold of the feature space. Testing on real network data we show that the embedding quality of *hydra* is competitive with existing hyperbolic embedding methods, but achieved at substantially shorter computation time. An extended method, termed *hydra+*, outperforms existing methods in both computation time and embedding quality.

1. INTRODUCTION

Embeddings of networks and distance-based data into hyperbolic geometry have received substantial interest in recent years. Such embeddings have been used for visualization [23], link prediction [19, 20] and community detection [20, 16]. They offer insight into the tradeoff between popularity and similarity effects in network growth [19] and have interesting implications for routing, network navigability [10, 1] and efficient computation of shortest network paths [25, 5]. Moreover, such embedding methods can be seen as alternatives to classic visualization and dimensionality reduction techniques based on Euclidean geometry, such as principal component analysis or multidimensional scaling. However, the hyperbolic embedding methods as yet proposed in the literature have either been based on specific assumptions about network growth (e.g. [20, 16]), or methods with strong theoretical properties, but requiring costly non-linear numerical optimization procedures (e.g. H-MDS of [23], *Rigel* of [25] and *HyPy* of [5]). Here, we introduce *hydra* (hyperbolic distance recovery and approximation), a novel method for embedding network or distance-based data into hyperbolic space, which has strong mathematical foundations and does not depend on specific assumptions on network growth or structure. At the same time, the method is computationally efficient and based on reduced matrix Eigendecomposition. We show mathematically, that when presented with mutual distances of data points located on a low-dimensional hyperbolic submanifold of the feature space, *hydra* will recover these points exactly. For general data, the method satisfies a certain optimality property, similar to the strain-minimizing property of multidimensional scaling. Finally, we introduce *hydra+*, an extension where the result of *hydra* is used as initial condition for hyperbolic embedding methods based on optimization, such as *Rigel*/*HyPy*,

INSTITUTE OF MATHEMATICAL STOCHASTICS, DEPARTMENT OF MATHEMATICS, TU DRESDEN, GERMANY

Date: September 4, 2019.

The authors would like to thank Carlo Vittorio Cannistraci and Alessandro Muscoloni for comments and discussions.

substantially improving their efficiency. When tested on real network data, `hydra` and its variants typically outperform existing hyperbolic embedding methods. All new methods introduced are available in the package `hydra` [9] for the statistical computing environment `R` [21].

2. EMBEDDINGS INTO HYPERBOLIC SPACE

2.1. Hyperbolic Space. We summarize the key features of the hyperboloid model of hyperbolic geometry (cf. [22, 3]) in dimension d . This will provide the mathematical framework in which we formulate our embedding method. To start, we define for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{d+1}$ the indefinite inner product

$$(1) \quad \mathbf{x} \circ \mathbf{y} := x_1 y_1 - (x_2 y_2 + \dots + x_{d+1} y_{d+1}),$$

also called *Lorentz product*. The real vector space \mathbb{R}^{d+1} equipped with this inner product is called *Lorentz space* and denoted by $\mathbb{R}^{1,d}$. As nested subsets, it contains the *positive Lorentz space* $\mathbb{R}_+^{1,d} = \{\mathbf{x} \in \mathbb{R}^{1,d} : x_1 > 0\}$ and the single-sheet hyperboloid

$$(2) \quad \mathcal{H}_d = \{\mathbf{x} \in \mathbb{R}^{1,d} : \mathbf{x} \circ \mathbf{x} = 1, x_1 > 0\}.$$

The *hyperboloid model* with curvature $-\kappa$, ($\kappa > 0$), consists of \mathcal{H}_d endowed with the hyperbolic distance

$$(3) \quad d_H^\kappa(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{\kappa}} \operatorname{arcosh}(\mathbf{x} \circ \mathbf{y}), \quad \mathbf{x}, \mathbf{y} \in \mathcal{H}_d.$$

The hyperbolic distance d_H^κ is a distance on \mathcal{H}_d in the usual mathematical sense; in particular it takes only positive values and satisfies the triangle inequality, cf. [22, §3.2]. In fact, it can be shown that \mathcal{H}_d endowed with the Riemannian metric tensor

$$ds^2 = \frac{1}{\kappa} (d\mathbf{x} \circ d\mathbf{x})$$

is a Riemannian manifold and $d_H^\kappa(\mathbf{x}, \mathbf{y})$ is the corresponding Riemannian distance.¹ The sectional curvature of this manifold is constant and equal to $-\kappa$, which explains the role of κ as curvature parameter. Just as Euclidean space is the canonical model of geometry with zero curvature, hyperbolic space is the canonical model of geometry with negative curvature.

2.2. The Poincaré ball Model. In addition to the hyperboloid model, we introduce the *Poincaré ball model* of hyperbolic geometry, which is more appealing for visualizations of hyperbolic space and hyperbolic embeddings. In \mathbb{R}^d , consider the open unit ball

$$\mathcal{B}_d := \{\mathbf{z} \in \mathbb{R}^d : |\mathbf{z}| < 1\},$$

where $|\mathbf{z}| = \sqrt{z_1^2 + \dots + z_d^2}$ is the usual Euclidean norm. The *stereographic projection* $\xi : \mathbb{R}^{1,d} \rightarrow \mathbb{R}^d$ defined by

$$(4) \quad \xi(\mathbf{x}) = \left(\frac{x_2}{1+x_1}, \dots, \frac{x_{d+1}}{1+x_1} \right)$$

restricts to a bijective mapping from the hyperboloid \mathcal{H}_d onto \mathcal{B}_d (cf. [22, §4.2]). It transfers the hyperbolic distance from \mathcal{H}_d to \mathcal{B}_d , by setting

$$d_B^\kappa(\xi(\mathbf{z}_1), \xi(\mathbf{z}_2)) = d_H^\kappa(\mathbf{z}_1, \mathbf{z}_2), \quad \mathbf{z}_1, \mathbf{z}_2 \in \mathcal{B}_d.$$

¹That is, $d_H^\kappa(\mathbf{x}, \mathbf{y})$ is the length of the shortest path from \mathbf{x} to \mathbf{y} , where lengths are measured using the length element $ds = \sqrt{ds^2}$.

Endowed with this distance, $(\mathcal{B}_d, d_B^\kappa)$ is isometric to $(\mathcal{H}_d, d_H^\kappa)$ and therefore an equivalent model of hyperbolic geometry.

It will be convenient to parameterize \mathcal{B}_d by the radial coordinate $r \in [0, 1)$ and the directional coordinate \mathbf{u} (a unit vector in \mathbb{R}^d), given by

$$r := \sqrt{z_1^2 + \cdots + z_d^2}, \quad \mathbf{u} := \frac{\mathbf{z}}{r}.$$

An easy calculation shows that the conversion from coordinates in \mathcal{H}_d is given by

$$(5) \quad r = \xi_r(x_1) := \sqrt{\frac{x_1 - 1}{x_1 + 1}} \quad \text{and}$$

$$(6) \quad \mathbf{u} = \xi_{\mathbf{u}}(x_2, \dots, x_{d+1}) := (x_2, \dots, x_{d+1}) / \sqrt{x_2^2 + \cdots + x_{d+1}^2}.$$

In dimension $d = 2$, the Poincaré ball becomes the *Poincaré disc*, and each of its points can be described by the radius r and the unique angle $\theta \in [0, 2\pi)$ such that

$$z_1 = r \cos \theta, \quad z_2 = r \sin \theta.$$

2.3. Embedding of Distances and Networks. To formulate the embedding problem, let a symmetric matrix $D = [d_{ij}] \in \mathbb{R}_{\geq 0}^{n \times n}$ with zero diagonal be given, which represents the pairwise dissimilarities between some objects $\mathbf{o}_1, \dots, \mathbf{o}_n$. The basic premise of hyperbolic embedding is that the matrix D can be approximated by a hyperbolic distance matrix $H = [d_H^\kappa(\mathbf{x}_i, \mathbf{x}_j)]$, i.e., that we can find points $\mathbf{x}_1, \dots, \mathbf{x}_n$ in low-dimensional hyperbolic space \mathcal{H}_d , such that

$$(7) \quad d_H^\kappa(\mathbf{x}_i, \mathbf{x}_j) \approx d_{ij}.$$

The points $\mathbf{x}_1, \dots, \mathbf{x}_n$ give a low-dimensional representation in hyperbolic space of the configuration of $\mathbf{o}_1, \dots, \mathbf{o}_n$ induced by their dissimilarities. In Euclidean space, such approximations are well studied and can be calculated e.g. by multidimensional scaling (MDS), see also Section A.2 and [2].

An important special case is the *network embedding problem*, where a (unweighted, undirected) graph $G = (V, E)$ is given and $D = [d_{ij}]$ is the graph distance matrix of G , i.e., d_{ij} is the length of the shortest path in G from vertex v_i to v_j . In particular for graphs with *locally tree-like structure* it can be expected that hyperbolic geometry gives a better representation than Euclidean geometry, see e.g. [10]. Instead of the shortest-path distance, other dissimilarity measures based on the structure of G can be used, such as the repulsion-attraction (RA) rule or edge-betweenness-centrality (EBC), cf. [16].

2.4. Connection to prior work and innovations. Most existing methods for hyperbolic embedding can be placed into one of two classes: Stress-based methods or network-specific methods.

- **Stress-based methods** aim to solve the embedding problem (7) by minimizing the *stress functional*

$$(8) \quad \text{Stress}(\mathbf{x}_1, \dots, \mathbf{x}_n)^2 := \sum_{i,j=1}^n (d_{ij} - d_H^\kappa(\mathbf{x}_i, \mathbf{x}_j))^2$$

over all $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{H}_d$. This minimization problem is a challenging high-dimensional non-convex optimization problem, and methods largely differ in their numerical approach to minimize (8). The H-MDS method proposed in [23] is a gradient descent scheme for minimizing (8) based on explicit calculation of the gradient. [4] propose

a neural-network-based approach to minimizing (8), while [25] and [5] develop so-called ‘landmark-based’ minimization algorithms (**Rigel** and **HyPy** respectively) based on iterative quasi-Newton minimization. Due to the ‘landmark’ heuristic, these methods are able to deal with large-scale instances of (8) and do not require full knowledge of D , see [5] for details.

- **Model-based methods** focus on the network embedding problem and rely on underlying assumptions on the generating mechanism of the graph G , see e.g. [19] for a model of ‘hyperbolic network growth’. In **HyperMap** of [20] and the *coalescent embedding* of [16], the radial coordinate r_i of the embedded points in the Poincaré ball model is determined directly from the degree of the vertices v_i , using the assumption of a power-law relationship. The directional component \mathbf{u}_i of the embedding is then determined by maximizing likelihood in an underlying probabilistic model (cf. [20]) or by applying existing nonlinear dimensionality reduction methods (such as Laplacian Eigenmapping or **ISOMAP**) to the underlying data (cf. [16]).

Here, our main innovation is to replace the stress functional (8) by the *strain functional*

$$(9) \quad \text{Strain}(\mathbf{x}_1, \dots, \mathbf{x}_n)^2 := \sum_{i,j=1}^n (\cosh(\sqrt{\kappa} d_{ij}) - \mathbf{x}_i \circ \mathbf{x}_j)^2,$$

which results from (8) when all distances are transformed by hyperbolic cosine. Furthermore, we introduce a highly efficient method for the minimization of hyperbolic strain, called **hydra** (hyperbolic distance recovery and approximation). Contrary to stress-minimization, **hydra** is based on matrix Eigendecomposition, similar to principal component analysis or classic multidimensional scaling.² In Theorems 3.1 and 3.2 we show that **hydra** satisfies important theoretical optimality properties, in particular, it returns a guaranteed global minimum of (9). For instances based on real data, the embedding results of **hydra** are comparable to those based on pure stress-minimization, even when embedding quality is measured by the stress functional (8); see Section 4 below. This shows, that even when minimization of stress is the final goal, the strain functional (9) is a valuable and useful proxy for stress, as it can be minimized in a highly efficient way. The best results in terms of stress are obtained when strain- and stress-minimization are combined. This is the basis of the **hydra+** method, introduced in Section 3.2, where the embedding result of **hydra** is used as initial condition for a stress-minimization run.

3. A NEW HYPERBOLIC EMBEDDING METHOD

3.1. The hydra algorithm. We introduce the **hydra** algorithm, displayed as Algorithm 1, which calculates an embedding into the Poincaré ball model of hyperbolic space by efficiently solving the strain-minimization problem

$$(10) \quad \min_{\mathbf{x}_i \in \mathbb{R}^{1,d}} \sum_{i,j} (\cosh(\sqrt{\kappa} d_{ij}) - (\mathbf{x}_i \circ \mathbf{x}_j))^2.$$

The algorithm proceeds as follows:

- In steps A1 and A2 the strain-minimization problem is solved by means of a matrix Eigendecomposition. These steps return a coordinate matrix $X = [x_{ij}]$, whose rows $\mathbf{x}_1, \dots, \mathbf{x}_n$ are elements of positive Lorentz space $\mathbb{R}_+^{1,d}$

²In fact, the relation between hyperbolic strain- and stress-minimization is similar to the relation between ‘classic’ and ‘metric’ multidimensional scaling in the Euclidean case, cf. [2].

and the optimizers of (10). The optimality of $\mathbf{x}_1, \dots, \mathbf{x}_n$ is the subject of Theorem 3.2 below.

- In steps B1 and B2, the points $\mathbf{x}_1, \dots, \mathbf{x}_n$ are projected onto the Poincaré ball \mathcal{B}_d using the stereographic projection (4) and converted to radial/directional coordinates (\mathbf{u}_i) using (5) and (6). No adjustment is necessary for the directional coordinates, which are computed in step B1.
- Due to (5), the radial coordinates (r_i) depend only on the first column (x_{11}, \dots, x_{n1}) of X and can be obtained by applying ξ_r elementwise. But $\xi_r(x_{1i})$ may be undefined for elements with $x_{1i} \in (0, 1)$.³ Therefore, r_i is calculated in step B2 as

$$r_i = \xi_r \left(\frac{x_{i1}}{x_{\min}} \right),$$

that is, after rescaling the first column of X by dividing by its smallest element x_{\min} .

Algorithm 1 $\text{hydra}(D, d, \kappa)$

Input:

- A symmetric matrix $D = [d_{ij}] \in \mathbb{R}_{\geq 0}^{n \times n}$ with zero diagonal
- Embedding dimension $d \leq n - 1$
- Hyperbolic curvature parameter $\kappa > 0$ (actual curvature: $-\kappa$)

Step A1: Set

$$(11) \quad A = [a_{ij}] := [\cosh(\sqrt{\kappa} d_{ij})]$$

and compute the Eigendecomposition

$$(12) \quad A = Q\Lambda Q^\top,$$

where Λ is the diagonal matrix of the Eigenvalues $\lambda_1 \geq \dots \geq \lambda_n$ and the columns of Q are the Eigenvectors $\mathbf{q}_1, \dots, \mathbf{q}_n$.

Step A2: Allocate the $n \times (d + 1)$ -matrix

$$(13) \quad X := \left[\sqrt{\lambda_1} \mathbf{q}_1 \quad \sqrt{(-\lambda_{n-d+1})^+} \mathbf{q}_{n-d+1} \quad \dots \quad \sqrt{(-\lambda_n)^+} \mathbf{q}_n \right],$$

where x^+ denotes the positive part $x^+ = \max(x, 0)$.

Step B1: ('Directional projection') For $i \in 1, \dots, n$ set

$$\mathbf{u}_i := \frac{(x_{i2}, \dots, x_{i(d+1)})}{\sqrt{x_{i2}^2 + \dots + x_{i(d+1)}^2}},$$

with x_{ij} the elements of X .

Step B2: ('Radial projection') For $i \in 1, \dots, n$ set

$$x_{\min} := \min(1, x_{11}, \dots, x_{n1})$$

and

$$r_i := \sqrt{\frac{x_{i1} - x_{\min}}{x_{i1} + x_{\min}}}$$

Return: Matrix X and embedding $(r_i, \mathbf{u}_i)_{i=1, \dots, n}$ as radial and directional coordinates in the Poincaré ball \mathcal{B}_d .

³By Theorem 3.2, steps A1 and A2 guarantee that all x_{1i} are positive, but not that they are larger than one.

The key theoretical properties of the `hydra` algorithm are summarized in the following theorems, whose proofs are given in Appendix A. The first theorem shows that `hydra` recovers any configuration of points in d -dimensional hyperbolic space up to isometry:

Theorem 3.1 (Exact Recovery). *Let $\mathbf{a}_1, \dots, \mathbf{a}_n$ be points in hyperbolic d -space \mathcal{H}_d , and let $D = [d_{ij}] = [d_H^\kappa(\mathbf{a}_i, \mathbf{a}_j)]$ be the matrix of their hyperbolic distances with curvature $-\kappa$. Then `hydra`(D, d, κ) recovers the points $\mathbf{a}_1, \dots, \mathbf{a}_n$ up to isometry. In particular, the rows $\mathbf{x}_1, \dots, \mathbf{x}_n$ of the matrix X and the points $(r_1, \mathbf{u}_1), \dots, (r_n, \mathbf{u}_n)$ returned by `hydra`(D, d, κ) satisfy*

$$d_B^\kappa\left((r_i, \mathbf{u}_i), (r_j, \mathbf{u}_j)\right) = d_H(\mathbf{x}_i, \mathbf{x}_j) = d_{ij}, \quad i, j = 1, \dots, n.$$

For applications to real data, exact recovery is an atypical situation. However, `hydra` enjoys an optimality guarantee for strain minimization, expressed in the following theorem:

Theorem 3.2 (Optimal Approximation). *The rows $\mathbf{x}_1, \dots, \mathbf{x}_n$ of the matrix X returned by `hydra`(D, d, κ) are the globally optimal solutions of the strain minimization problem (10). Moreover, the first column of X is strictly positive; equivalently, all \mathbf{x}_i are elements of positive Lorentz space $\mathbb{R}_+^{1,d}$.*

3.2. Practical guidelines and extensions. While the result of `hydra` satisfies the theoretical optimality guarantees in Theorem 3.1 and 3.2, it can still be advantageous to adjust the results in order to improve the attractiveness of visualization or the embedding quality in terms of stress (8) (as opposed to strain, which is globally minimal). The so-called *equiangular adjustment* has been introduced in [16] and can be applied to two-dimensional hyperbolic embeddings. Here we propose a slight modification, which allows to interpolate smoothly between no adjustment and full equiangular adjustment.

Equiangular adjustment: Let $\lambda \in [0, 1]$ be the adjustment parameter and define $\text{ark}(\theta_i)$ as the *angular rank* of \mathbf{x}_i , i.e. when the embedded points are ordered by increasing angular coordinate θ , then $\text{ark}(\theta_i)$ is defined as the rank (from 1 to n) of \mathbf{x}_i in this list. The adjusted angular coordinate is then set to

$$\theta'_i := \lambda\theta_i + (1 - \lambda)(\text{ark}(\theta_i) - 1)\frac{2\pi}{n}, \quad i = 1, \dots, n.$$

If $\lambda = 0$, no adjustment takes place. If $\lambda = 1$ then the angles θ'_i are regularly spaced ('equiangular') and only the ordering given by θ_i is retained.⁴ Values of $\lambda \in (0, 1)$ interpolate between these two extremes. We propose a values of $\lambda = 1/2$, which typically leads to improvements in both visual appeal and stress value of the embedding; see also method `hydra-equi` in Figure 1.

hydra+: If minimization of stress is the ultimate objective and strain is used only as a proxy, the result of `hydra` can be used as an initial condition for a direct minimization of the stress functional (8). This can be seen as a chaining of `hydra` and `HyPy/Rigel` [25, 5], where `hydra` substitutes the random initial condition of `HyPy/Rigel`. For the minimization of stress, efficient quasi-Newton routines, such as LBFGS [26] can be used and supplied with the explicit gradient of stress, given in [5, Eqs. (3.1),(3.2)].

In terms of efficiency, the following simple improvement can be made to `hydra`:

⁴This is the equiangular adjustment as proposed in [16].

Reduced Eigendecomposition: The numerically dominating part of `hydra` is the Eigendecomposition in (12). Note, however, that in (13) only the single first and the last d Eigenvalues and Eigenvectors of the matrix A are needed. There are efficient numerical routines (see e.g. [12]) to perform such a reduced Eigendecomposition without computing the full Eigendecomposition of A . These routines substantially improve efficiency if $n \gg d$ and are used in our implementation of `hydra`.

Using the reduced Eigendecomposition, we expect the time complexity of `hydra` to be $\mathcal{O}(n^\alpha)$ with α slightly above, but close, to 2, cf. [8, Ch. 55]. For `hydra+`, the time complexity is harder to estimate, since it is based on iterative minimization of a non-convex objective function. In a single step of LBFGS both the stress functional and its gradient have to be evaluated at a complexity of $\mathcal{O}(n^2)$. Depending on the number of steps to convergence, we thus also expect a complexity of $\mathcal{O}(n^\alpha)$, with α strictly larger than 2. Empirical estimates of α are given in Section 4 below.

3.3. Remarks on strain-minimizing graph embeddings. In the seminal paper [19] it has been argued that the inherent negative curvature in hyperbolic geometry resolves the trade-off between the conflicting attractive forces of popularity and similarity in network growth models. For this reason [19] have proposed to interpret the radial coordinate r in the Poincaré disc as dimension of ‘popularity’ and the angular coordinate θ as dimension of ‘similarity’. Interestingly, the strain minimization problem (10) and its solution by `hydra` gives additional mathematical support for this interpretation. More precisely, revisiting Algorithm 1 in the graph embedding context, we observe that:

- The **radial coordinates** r_i are determined only from the Perron-Frobenius Eigenvector⁵ \mathbf{q}_1 of the matrix A . This provides a remarkable connection to the *Eigenvector centralities* (corresponding to the popularity dimension) of the nodes v_i , which are determined from the Perron-Frobenius Eigenvector of their *adjacency matrix*.
- The **directional coordinates** \mathbf{u}_i are determined only through the Eigenvectors $\mathbf{q}_{n-d+1}, \dots, \mathbf{q}_n$ (and corresponding Eigenvalues) at the *low end* of the spectrum of A . This provides a remarkable connection to Cheeger’s inequality (cf. [6, Ch. 9]), which shows that the low end of the spectrum of the graph *Laplacian matrix* encodes the separability of the graph into sparsely connected ‘communities’ (corresponding to the similarity dimension).

We remark that while the connections described above are a first step towards a mathematization of the popularity-similarity paradigm in hyperbolic network geometry, the matrix $A = [\cosh(\sqrt{\kappa}d_{ij})]$ is in general neither identical to the adjacency nor to the Laplacian matrix of a given graph, and thus further research into the rigorous mathematical underpinning of these connections is warranted.

4. NUMERICAL RESULTS

4.1. Methods and Data. In our numerical experiments, we evaluate different variants of `hydra` and compare them to existing hyperbolic embedding methods, using stress as performance criterion. We focus on small to medium sized networks (see Table 1), for which it is still feasible to compute the full distance matrix as input to our methods. Edge weights (when available) were discarded, i.e., all networks were

⁵The Perron-Frobenius Eigenvector is the Eigenvector associated to the largest Eigenvalue of a positive matrix (i.e. a matrix consisting only of positive entries) and is itself a positive vector, cf. [8, Ch. 10].

treated as unweighted undirected graphs. This network data was used as input for the following methods:

hydra: The `hydra` method (without equi-angular adjustment) as described in Algorithm 1

hydra-equi: The `hydra` method with equiangular adjustment $\lambda = 0.5$, as described in Section 3.2

hydra+: The `hydra+` method as described in Section 3.2 and using the result of `hydra-equi` as initial condition.

HyPy/Rigel: The HyPy algorithm from [5], which is based on Rigel from [25]. Both methods are based on direct minimization of the stress functional (8). *Landmark selection*, as proposed in [5] was not implemented, since it serves to reduce runtime and memory use for large networks, but is not expected to improve embedding results. As in [5], the initial condition for minimization was chosen at random and we repeated the embedding 100 times.

CE-LE: The coalescent embedding (CE) using Laplacian Eigenmapping (LE) as dimension-reduction method, full equiangular adjustment and repulsion-attraction (RA) pre-weighting; see [16] for details. Among the methods developed in [16], this was the best performing method to invert the PSO generating model of [20] for hyperbolic networks.

For the methods `hydra`, `hydra-equi`, `hydra+` and `HyPy/Rigel` we used our own implementations in R, which are available in the R-package `hydra`, [9]. For `CE-LE` we used the MATLAB implementation of the methods of [16] available from github.⁶ The stress-optimization in `hydra+` and `HyPy/Rigel` was performed using the LBFGS method (see [26]) as implemented in the R-function `optim` and using the analytic form of the gradient of the stress functional (8) from [5]. Note that all methods except `CE-LE` use the shortest-path matrix as input dissimilarities; `CE-LE` uses repulsion-attraction (RA) weights as input dissimilarities, see [16]. For all methods hyperbolic curvature was fixed to $-\kappa = -1$ and we embed into dimension $d = 2$.

Network	Description	Source	# Nodes
karate	Social interaction network ('Zachary's karate club network') from [24]	<code>igraphdata</code> [7]	34
UKfaculty	Personal friendship network of a UK university faculty from [17]	<code>igraphdata</code> [7]	81
opsahl	One-node projection of message Exchange Network from [18]; two isolated nodes have been removed	<code>toreopsahl.com</code>	897
facebook	Facebook social circles network from [15]; combined edge sets	<code>snap.stanford.edu</code>	4039
collaboration	Co-authorship network from ArXiv submissions to category Hep-Ph (High Energy Physics); largest connected component. From [14]	<code>snap.stanford.edu</code>	8638
oregon	Autonomous systems peering information inferred from route-views in Oregon on March 26, 2001. From [13]	<code>snap.stanford.edu</code>	11174

TABLE 1. Networks used for numerical experiments

⁶https://github.com/biomedical-cybernetics/coalescent_embedding

4.2. Results and Discussion. Results on embedding quality (measured by stress) for all networks and methods (except CE-LE) are shown in Figure 1. Note that stress values are normalized by using the average result of HyPy/Rigel as a reference. This facilitates the comparison of results between different networks. As HyPy/Rigel depends on randomized initial conditions we indicate the 5%- and 95%-quantiles (over 100 runs) in addition to its average result. The stress-values of the embeddings produced by CE-LE were substantially larger (by a factor from 13 to 26) than the reference method and we have therefore excluded this method from the plot and from the further analysis of computation times.

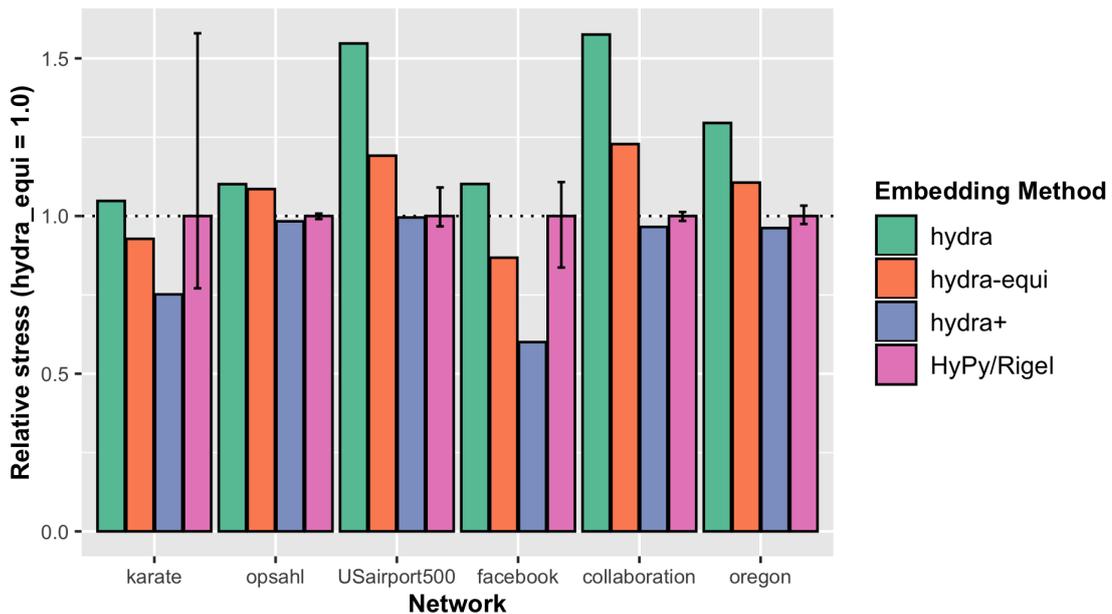


FIGURE 1. **Embedding performance on real network data.** Embedding quality (measured by stress (8), relative to the average result of HyPy/Rigel) of different hyperbolic embedding methods applied to the six networks listed in Table 1. For HyPy/Rigel a 5%–95% error bar is shown, corresponding to 100 runs with randomized initial condition.

Computation times of the different methods is shown in Figure 2 in doubly logarithmic coordinates. As should be expected, the methods split into two groups with computation times for `hydra` and `hydra-equi` being shorter than for `hydra+` and `HyPy/Rigel` by two orders of magnitude. The seemingly small gap between `hydra+` and `HyPy/Rigel` still corresponds to a difference of about 50% in runtime. Based on the discussion in Section 3.2 we have added regression lines to estimate the exponent α in the conjectured complexity $\mathcal{O}(n^\alpha)$. To avoid clutter, regression lines are only shown for `hydra-equi` and `hydra+`; the estimates for all methods are $\alpha \approx 2.0$ for `hydra`, $\alpha \approx 2.1$ for `hydra-equi` and $\alpha \approx 2.3$ for both `hydra+` and `HyPy/Rigel`. As

setup costs seem to dominate the computation times for the smallest network, we have excluded it from the regression analysis.

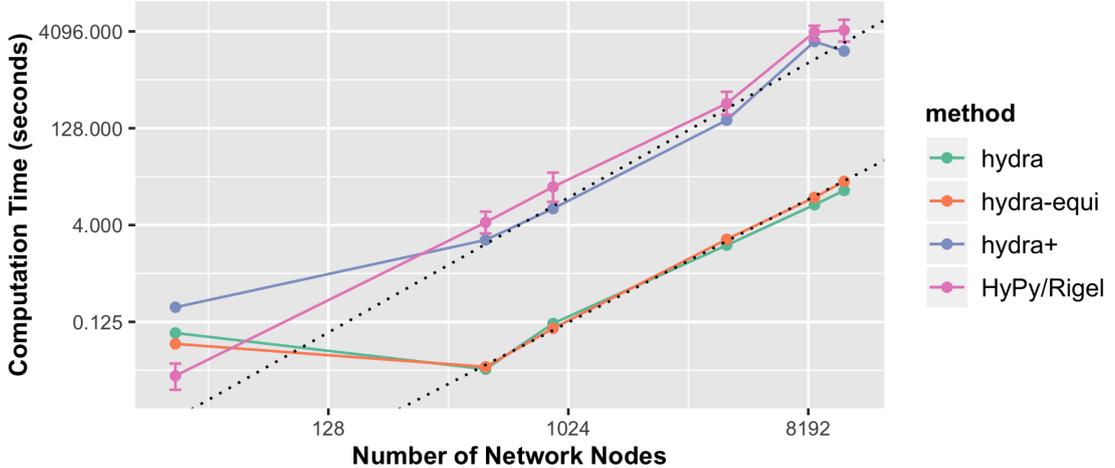


FIGURE 2. **Computation time of embedding methods.** Computation time (in seconds) of different hyperbolic embedding methods, in relation to the number of nodes in the six networks listed in Table 1. Coordinate axes are doubly logarithmic. For HyPy/Rigel, average computation time and a 5%–95% error bar is shown, corresponding to 100 runs with randomized initial condition. For hydra-equi and hydra+ dotted regression lines (excluding observations from the smallest network) are indicated.

Finally, an exemplary embedding result produced by the hydra-equi method for the facebook network is shown in Figure 3. Nodes are placed into the Poincaré disc model of hyperbolic geometry according to their embedding coordinates (the full disc is indicated in grey) and a random subsample of links is drawn as hyperbolic geodesics. Visually, the embedding conforms well with the popularity-vs-similarity paradigm of [19] for hyperbolic networks: Nodes with a function as hubs between communities or individuals (popularity dimension) are placed closer to the center of the hyperbolic disc. Communities are identified along the angular coordinate (similarity dimension) with the effective distance between communities indicated by angular separation.

Summarizing our numerical experiments, we conclude the following:

- In general, the strain-minimization performed by hydra seems to be a good proxy for stress-minimization, but is faster by a factor of 100 or more in comparison to stress-minimization from a random initial condition (HyPy/Rigel). Note that hydra also eliminated the uncertainty associated with the randomized nature of HyPy/Rigel, which can lead to large variations in embedding quality in some instances (e.g., the facebook network).
- The simple equi-angular adjustment performed in hydra-equi consistently improves embedding quality in terms of stress at negligible numerical costs. The returned embeddings outperform HyPy/Rigel for two networks (karate,

facebook) and are competitive for all others, with a largest observed difference of 23% in terms of stress.

- Using the results of **hydra** as a starting value for stress-minimization, instead of a random initial condition, i.e., replacing **HyPy/Rigel** by **hydra+** reduces computation time by approx. 30% - 50% and leads to better (average) embedding quality in all cases. The reduction in stress is considerable in the facebook network, where stress is reduced by approx. 40%.
- The **CE-LE** method, based on the PSO network growth model of [20], is not competitive with the other methods in terms of embedding quality. This suggests that the structure of the real networks that we have considered deviates from the theoretical growth model (PSO-model) of [20] upon which **CE-LE** is build.

As a next step, we plan to make strain-minimizing hyperbolic embedding methods feasible for large and very large networks. For such networks the computational complexity of $\mathcal{O}(n^\alpha)$ (with $\alpha > 2$) of the proposed methods, but also of the graph distance calculation itself, are prohibitive. For this reason, heuristics such as the landmark heuristic of [5] will have to be adapted to strain-minimizing embedding methods.

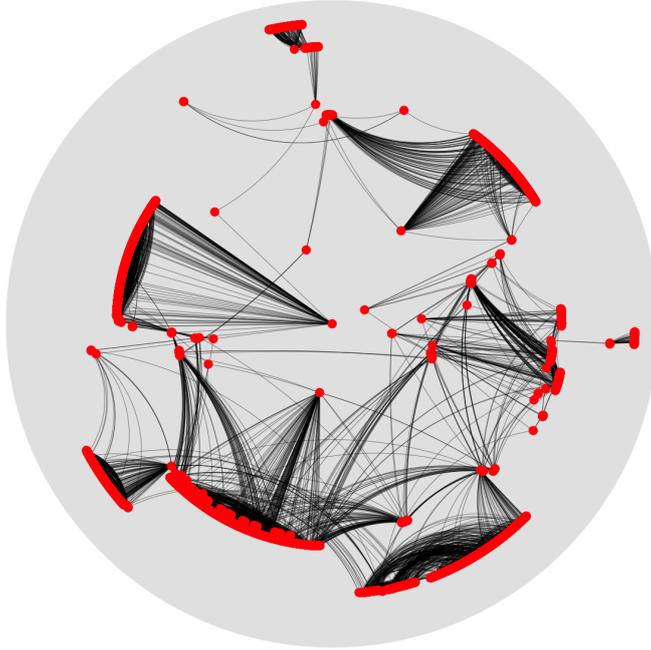


FIGURE 3. **Embedding example.** The hyperbolic embedding of the **facebook** network produced by the method **hydra-equi**. All 4039 network nodes are shown as red dots. A random subsample of the 88234 total edges are also shown and drawn as hyperbolic geodesics in black. The edge subsample was produced by randomly sampling two incident edges from each network node, allowing for repetitions.

APPENDIX A. THEORETICAL RESULTS

To prove the theoretical properties of the **hydra** method, it is convenient to reformulate the strain minimization problem (10) in matrix form. To this end, let $D = [d_{ij}]$ be the given dissimilarity matrix, set $A = [\cosh(\sqrt{\kappa} d_{ij})]$ and write

$$X = (\mathbf{x}_1, \dots, \mathbf{x}_n)^\top \in \mathbb{R}^{n \times (d+1)}$$

for the coordinate matrix of some points $\mathbf{x}_1, \dots, \mathbf{x}_n$ in \mathbb{R}^{d+1} . Finally, let J be the $(d+1) \times (d+1)$ diagonal matrix

$$(15) \quad J = \text{diag}(1, -1, \dots, -1),$$

cf. [22, §3.1]. The strain minimization problem (10) can now be written in compact form as

$$(16) \quad \min_{X \in \mathbb{R}^{n \times (d+1)}} \|A - X^\top J X\|_F^2,$$

where $\|\cdot\|_F$ denotes the Frobenius norm. Imposing the constraint that all \mathbf{x}_i are elements of the hyperboloid \mathcal{H}_d is equivalent to requiring that

$$\text{diag}(X^\top JX) = (1, \dots, 1) \quad \text{and} \quad X\mathbf{e}_1 > 0,$$

where \mathbf{e}_1 is the first standard unit vector. In particular, the first condition guarantees $\mathbf{x}_i \circ \mathbf{x}_i = 1$, and the second one selects the upper sheet of the two-sheet hyperboloid thus described.

A.1. Hyperbolic strain minimization and exact recovery. For a real symmetric matrix A , denote by $n_+(A)$ and $n_-(A)$ the number of positive and negative Eigenvalues of A . The following Lemma characterizes matrices that can be written as inner product matrices (‘Gram matrices’) with respect to the Lorentz product (1):

Lemma A.1. *Let $G = [g_{ij}] \in \mathbb{R}_{\geq 0}^{n \times n}$ be positive and symmetric, and let $d \leq n - 1$. The following are equivalent*

- a) G satisfies $n_+(G) = 1$ and $n_-(G) \leq d$.
- b) G is a ‘Lorentzian Gram matrix’, i.e., there exist $\mathbf{x}_1, \dots, \mathbf{x}_n$ in $\mathbb{R}^{1,d}$, such that

$$g_{ij} = \mathbf{x}_i \circ \mathbf{x}_j, \quad \forall i, j \in 1, \dots, n.$$

- c) There exists $X \in \mathbb{R}^{n \times (d+1)}$, such that

$$(17) \quad G = XJX^\top,$$

where J is given by (15).

In addition,

- The first column of X is positive if and only if $\mathbf{x}_1, \dots, \mathbf{x}_n$ are in the positive Lorentz space $\mathbb{R}_+^{1,d}$;
- The points $\mathbf{x}_1, \dots, \mathbf{x}_n$ are in \mathcal{H}_d if and only if $\text{diag}(G) = (1, \dots, 1)$ and the first column of X is positive.

Proof. The equivalence of (b) and (c) follows directly from the definition of the Lorentz product in (1). Next, we show that (c) implies (a): From [11, Ch. 10.3] it follows from (17) that $n_+(G) \leq n_+(J) = 1$ and $n_-(G) \leq n_-(J) = d$. But G is a positive matrix and Perron’s theorem (cf. [11, Ch. 16]) guarantees that its leading Eigenvalue is positive, i.e., $n_+(G) \geq 1$, and we conclude (a). To show that (a) implies (c), assume first that $n_-(G) = d$. By Sylvester’s law of inertia, there exists a decomposition

$$G = \hat{X}\hat{J}\hat{X}^\top, \quad \text{where} \quad \hat{J} = \text{diag}\left(+1, 0, \dots, 0, \underbrace{-1, \dots, -1}_{d \text{ times}}\right).$$

This decomposition can be reduced to (17), by simply dropping all rows and columns containing only zeroes from \hat{J} and by also dropping the corresponding columns from \hat{X} . If $n_-(G) = d' < d$, the same procedure yields a decomposition with X of dimension $n \times (d' + 1)$ and J of dimension $(d' + 1) \times (d' + 1)$. Padding X with zero columns and J ’s diagonal with -1 s, (17) also follows in this case.

The additional statements follow directly from the following observations: The first column of X contains exactly the first coordinate of all points $\mathbf{x}_1, \dots, \mathbf{x}_n$. If the first coordinate of a point \mathbf{x} is positive, it is an element of positive Lorentz space and vice versa. The diagonal of G contains the values $\mathbf{x}_i \circ \mathbf{x}_i, i = 1, \dots, n$. If $\mathbf{x}_i \circ \mathbf{x}_i = 1$ and $\mathbf{x}_i \in \mathbb{R}_+^{1,d}$ then \mathbf{x}_i is an element of the hyperboloid \mathcal{H}_d and vice versa. \square

Proof of Theorem 3.2. Let $A = [a_{ij}] = [\cosh(\sqrt{\kappa} d_{ij})]$ and let $B = [b_{ij}]$ be another symmetric matrix in $\mathbb{R}^{n \times n}$. Let $(\lambda_i(A))_{i=1, \dots, n}$ and $(\lambda_i(B))_{i=1, \dots, n}$ be their Eigenvalues in descending order, and denote by $\|\cdot\|_F$ the Frobenius norm. By a result of Wielandt-Hoffmann, cf. [11, Ch. 10, Thm. 18],

$$(18) \quad \sum_{i,j} (a_{ij} - b_{ij})^2 = \|A - B\|_F^2 \geq \sum_i^n (\lambda_i(A) - \lambda_i(B))^2.$$

Assume now that B is a ‘Lorentzian Gram matrix’ with elements given by

$$b_{ij} = \mathbf{b}_i \circ \mathbf{b}_j, \quad i, j = 1, \dots, n$$

for some $\mathbf{b}_1, \dots, \mathbf{b}_n \in \mathbb{R}^{1,d}$. By Lemma A.1 this implies that $n_+(B) = 1$ and $n_-(B) \leq d$. Hence all Eigenvalues of B with index $2, \dots, n-d$ are zero, and we obtain

$$\begin{aligned} \sum_{i,j} (a_{ij} - \mathbf{b}_i \circ \mathbf{b}_j)^2 &= \|A - B\|_F^2 \geq \\ &\geq (\lambda_1(A) - \lambda_1(B))^2 + \sum_{i=2}^{n-d} \lambda_i(A)^2 + \sum_{i=n-d+1}^n (\lambda_i(A) - \lambda_i(B))^2. \end{aligned}$$

For the first summand on the right hand side we have the trivial lower bound 0. In the last sum, all $\lambda_i(B)$ are negative or zero, and hence, for any $i = (n-d+1), \dots, n$, we can estimate

$$(\lambda_i(A) - \lambda_i(B))^2 \geq \begin{cases} 0 & \text{if } \lambda_i(A) \leq 0 \\ \lambda_i(A)^2 & \text{if } \lambda_i(A) > 0, \end{cases}$$

which is the same as $(\lambda_i(A)^+)^2$. Together, we obtain that

$$(19) \quad \sum_{i,j} (a_{ij} - \mathbf{b}_i \circ \mathbf{b}_j)^2 \geq \sum_{i=2}^{n-d} \lambda_i(A)^2 + \sum_{i=n-d+1}^n (\lambda_i(A)^+)^2.$$

Denote by $A = Q\Lambda_A Q^\top$ the Eigendecomposition of A with $\Lambda_A = \text{diag}(\lambda_1(A), \dots, \lambda_n(A))$. Let X be the matrix returned by `hydra`(D, d, κ) and $\mathbf{x}_1, \dots, \mathbf{x}_n$ the rows of X . By (13) the associated Lorentzian Gram matrix $G = XJX^\top$ has the Eigendecomposition $G = Q\Lambda_G Q^\top$ with

$$\Lambda_G = \text{diag}(\lambda_1(A), 0, \dots, 0, (-\lambda_{n-d+1}(A))^+, \dots, (-\lambda_n(A))^+).$$

Using the unitary invariance of the Frobenius norm and the trivial identity $x - (-x)^+ = x^+$, we obtain

$$(20) \quad \begin{aligned} \sum_{i,j} (a_{ij} - \mathbf{x}_i \circ \mathbf{x}_j)^2 &= \|Q\Lambda_A Q^\top - Q\Lambda_G Q^\top\|_F^2 = \|\Lambda_A - \Lambda_G\|_F^2 = \\ &= \sum_{i=2}^{n-d} \lambda_i(A)^2 + \sum_{i=n-d+1}^n (\lambda_i(A)^+)^2. \end{aligned}$$

This shows that setting $\mathbf{b}_i := \mathbf{x}_i$ for all $i \in 1, \dots, n$ achieves equality in (19) and hence that the points \mathbf{x}_i minimize (16). \square

Proof of Theorem 3.1. Let $D = [d_{ij}]$ be the hyperbolic distance matrix of $\mathbf{a}_1, \dots, \mathbf{a}_n$ in \mathcal{H}^d . Then $A = [a_{ij}] = [\cosh(\sqrt{\kappa} d_{ij})]$ is the associated Lorentzian Gram matrix with elements

$$a_{ij} = \mathbf{a}_i \circ \mathbf{a}_j.$$

By Lemma A.1 A satisfies $n_+(A) = 1$ and $n_-(A) \leq d$, i.e. the Eigenvalues of A satisfy $\lambda_i(A) = 0$ for $i = 2, \dots, n-d$ and $\lambda_i(A) \leq 0$ for $i = n-d+1, \dots, n$. Hence, it follows from (20) that $\sum_{i,j} (a_{ij} - \mathbf{x}_i \circ \mathbf{x}_j)^2 = 0$ or, equivalently, that

$$\mathbf{x}_i \circ \mathbf{x}_j = \mathbf{a}_i \circ \mathbf{a}_j$$

for all $i, j \in 1, \dots, n$. Applying $\cosh(\sqrt{\kappa} \cdot)$ to both sides, we see that

$$d_H^\kappa(\mathbf{x}_i, \mathbf{x}_j) = d_H^\kappa(\mathbf{a}_i, \mathbf{a}_j)$$

and hence that (\mathbf{x}_i) and (\mathbf{a}_i) are isometric. \square

A.2. Comparison to classic multidimensional scaling. In several aspects, the hydra method can be seen as the ‘hyperbolic analogue’ of classic multidimensional scaling (MDS), cf. [2], which is based on Euclidean geometry. Below, we summarize the classical MDS method and point out parallels to (and differences from) **hydra**. Classical MDS also takes a matrix $D = [d_{ij}] \in \mathbb{R}_{\geq 0}^{n \times n}$ with zero diagonal as input. Using the centering matrix $C = I - \frac{1}{n} \mathbf{1} \mathbf{1}^\top \in \mathbb{R}^{n \times n}$, where $\mathbf{1}$ denotes a matrix of ones of matching dimension, the ‘doubly centered’ matrix

$$A = -\frac{1}{2} C^\top D C \quad [\text{compare (11)}]$$

is derived from D , and its Eigendecomposition

$$A = Q \Lambda Q^\top \quad [\text{compare (12)}]$$

computed. Again, Λ is the diagonal matrix of the Eigenvalues $\lambda_1 \geq \dots \geq \lambda_n$ and the columns of Q are the Eigenvectors $\mathbf{q}_1, \dots, \mathbf{q}_n$. MDS then returns the (Euclidean) coordinate matrix

$$X = \left[\sqrt{\lambda_1} \mathbf{q}_1 \quad \sqrt{\lambda_2} \mathbf{q}_2 \quad \dots \quad \sqrt{\lambda_d} \mathbf{q}_d \right], \quad [\text{compare (13)}]$$

whose rows \mathbf{x}_i are interpreted as points in Euclidean space \mathbb{R}^d . This coordinate matrix X solves the strain minimization problem

$$\min_{X \in \mathbb{R}^{n \times d}} \|A - X^\top X\|_F^2, \quad [\text{compare (16)}]$$

cf. [2, Ch. 12]. Moreover, if the input matrix D is a matrix of *squared* Euclidean distances, i.e., $d_{ij} = |\mathbf{x}_i - \mathbf{x}_j|^2$ then MDS recovers the points \mathbf{x}_i exactly (up to Euclidean isometry). Note that $X^\top X$ appearing above is the Gram matrix of the points $\mathbf{x}_1, \dots, \mathbf{x}_n$, i.e. the matrix of their scalar products $\mathbf{x}_i^\top \mathbf{x}_j$, whereas the matrix $X^\top J X$ in (16) is the ‘Lorentzian Gram matrix’ of the Lorentz products $\mathbf{x}_i \circ \mathbf{x}_j$.

REFERENCES

- [1] Marian Boguna, Dmitri Krioukov, and Kimberly C Claffy. Navigability of complex networks. *Nature Physics*, 5(1):74, 2009.
- [2] Ingwer Borg and Patrick JF Groenen. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.
- [3] W. James Cannon, William J. Floyd, Richard Kenyon, and Walter R. Parry. Hyperbolic geometry. In Silvio Levy, editor, *Flavors of Geometry*, pages 59–115. 31 edition, 1997.
- [4] Benjamin Paul Chamberlain, James Clough, and Marc Peter Deisenroth. Neural embeddings of graphs in hyperbolic space. *arXiv:1705.10359*, 2017.
- [5] Kenny Chowdhary and Tamara G Kolda. An improved hyperbolic embedding algorithm. *Journal of Complex Networks*, 2017.
- [6] Fan RK Chung and Linyuan Lu. *Complex graphs and networks*. Number 107. American Mathematical Soc., 2006.
- [7] Gabor Csardi. *igraphdata: A Collection of Network Data Sets for the ‘igraph’ Package*, 2015. R package version 1.0.1.

- [8] Leslie Hogben. *Handbook of linear algebra*. CRC Press, 2006.
- [9] Martin Keller-Ressel. *hydra: Hyperbolic Embedding*, 2019. R package version 0.1.0.
- [10] Robert Kleinberg. Geographic routing using hyperbolic space. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications*, pages 1902–1909, 2007.
- [11] Peter Lax. *Linear Algebra and Its Applications*. Wiley, 2nd ed. edition, 2007.
- [12] Richard B Lehoucq, Danny C Sorensen, and Chao Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, 1998.
- [13] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 177–187. ACM, 2005.
- [14] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2, 2007.
- [15] Jure Leskovec and Julian J McAuley. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*, pages 539–547, 2012.
- [16] Alessandro Muscoloni, Josephine Maria Thomas, Sara Ciucci, Ginestra Bianconi, and Carlo Vittorio Cannistraci. Machine learning meets complex networks via coalescent embedding in the hyperbolic space. *Nature Communications*, 8(1):1615, 2017.
- [17] Tamás Nepusz, Andrea Petróczy, László Négyessy, and Fülöp Bazsó. Fuzzy communities and the concept of bridgeness in complex networks. *Physical Review E*, 77(1):016107, 2008.
- [18] Tore Opsahl. Triadic closure in two-mode networks: Redefining the global and local clustering coefficients. *Social Networks*, 35(2):159–167, 2013.
- [19] Fragkiskos Papadopoulos, Maksim Kitsak, M Ángeles Serrano, Marián Boguná, and Dmitri Krioukov. Popularity versus similarity in growing networks. *Nature*, 489(7417):537, 2012.
- [20] Fragkiskos Papadopoulos, Constantinos Psomas, and Dmitri Krioukov. Network mapping by replaying hyperbolic growth. *IEEE/ACM Transactions on Networking (TON)*, 23(1):198–211, 2015.
- [21] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016.
- [22] John Ratcliffe. *Foundations of hyperbolic manifolds*, volume 149. Springer Science & Business Media, 2006.
- [23] Jörg A Walter. H-mds: a new approach for interactive visualization with multidimensional scaling in the hyperbolic space. *Information systems*, 29(4):273–292, 2004.
- [24] Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33(4):452–473, 1977.
- [25] Xiaohan Zhao, Alessandra Sala, Haitao Zheng, and Ben Y Zhao. Fast and scalable analysis of massive social graphs. *arXiv preprint arXiv:1107.5114*, 2011.
- [26] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560, 1997.