

# HyperLink algorithm description

Here we provide a detailed description of the embedding algorithm presented at the repository. A general outline of the algorithm and a discussion of its design is provided in the paper “[Link prediction with hyperbolic geometry](#)”.

## Algorithm input:

- path to network file in the edge list format;
- power-law exponent of the degree distribution  $\gamma$ ;
- temperature parameter of the  $\mathbb{H}^2$  model  $T$ ;
- seed for the random number generator;
- path to the output file with nodes’ coordinates;
- fraction of uniformly removed links  $1 - q$  from the network (set to 0 if the network is full);
- number of node layers  $m$ ;
- grid multiplier defining the resolution of angular inference search.

The algorithm consists of the following steps:

1. **Reading the network.** Input the network in the edge list format and extract its largest connected component in the `tmp_gcc.net` file.
2. **Computing network parameters.** Using the largest connected component, compute its observed size  $\tilde{N}$ , average degree  $\tilde{k}$ , maximum degree  $\tilde{k}_{max}$ .
3. **Inference of the  $\mathbb{H}^2$  model parameters.** Parameters  $R$ ,  $R_0$ ,  $N$ ,  $\bar{k}$  are inferred taking into account finite size effects, observed statistics of the network, and the fraction of uniformly retained links  $q$  in the case of link prediction, as explained in the Appendix F2 of the [paper](#).
4. **Inference of radial coordinates.** Radial coordinates of nodes are found using maximum-likelihood estimation (MLE). For each node  $i$ , its radial coordinate  $r_i$  is set to:
  - $r_i = R$ , if node’s degree  $k_i \leq \gamma T$ ;
  - $r_i = 2 \log \left[ \frac{q\bar{k}(1-e^{-(\gamma-2)(R-R_0)/2})}{k_i+(\gamma-1)T} \frac{\gamma-2}{\gamma-1} e^{R/2} \right]$ , otherwise.
5. **Construction of node layers and core subgraphs.**
  - All nodes are sorted by their degrees in the descending order.
  - For each entry  $N_l$  of the logarithmically-spaced sequence of integers from 1 to  $N$  of size  $m - 1$ , define the graph *core* consisting of the  $N_l$  highest-degree nodes, and the corresponding *layer of nodes* consisting of the nodes added between the  $N_{l-1}$ -th and  $N_l$ -th cores. The only exception is the first layer: its size is set to  $N_1 = 20$ . Nodes that have  $k_i = 1$  are collected into a separate  $m$ -th layer.
6. **Phase 1: embedding multiple replicas of highest-degree nodes.** Initialize  $n_{replica} = 20$  copies of the embedder, and for each of the replicas do:
  - For the first half of layers  $l = 1, \dots, \lfloor (m - 1)/2 \rfloor$ , do:
    - If  $l = 1$ , angles of all nodes are assigned randomly from the uniform distribution on  $[0, 2\pi]$ . The vector of node indices storing the *visiting order* in which the embedder visits nodes to infer their angular coordinates is populated with the nodes from the first layer.
    - Else, for each node  $i$  in the layer  $l$ , find the maximum-likelihood angle  $\theta_i$  as follows:
      - \* if the number of nodes in the current core subgraph  $G_l$  is less than 500, or the total number of nodes in the largest connected component  $G$  is less than 1,500, use the **full local log-likelihood** to estimate the optimal  $\theta_i$ . The full local log-likelihood of node  $i$  is defined as:

$$\log \mathcal{L}_i^{(full)} = \sum_{j \neq i \in G_l} [a_{ij} \log (qp_{ij}) + (1 - a_{ij}) \log (1 - qp_{ij})],$$

where  $a_{ij}$  is the entry of the original adjacency matrix,  $p_{ij}$  is the connection probability between nodes  $i$  and  $j$  in the  $\mathbb{H}^2$  model, and  $q$  is the fraction of retained links in the case of link prediction. The optimal angle  $\theta_i$  is found by first splitting the whole angular space  $[0, 2\pi]$  into  $N_l$  equal-sized regions of width  $\frac{2\pi}{N_l}$ , and then setting a *proposal angle*  $\theta_i^{(prop.)}$  to the lower boundary of that region. For each of the proposal angles, the full local log-likelihood is computed, and then the angle with the highest  $\log \mathcal{L}_i^{(full)}$  is selected as the optimal angle  $\theta_i$ .

- \* otherwise, use the **approximate local log-likelihood** to estimate the optimal  $\theta_i$ . The approximate log-likelihood of node  $i$  is defined as:

$$\log \mathcal{L}_i^{(approx.)} = \sum_{j \in \Gamma(i)} [a_{ij} \log(qp_{ij})],$$

where  $\Gamma(i)$  is the set of neighbors of node  $i$  present in the subgraph  $G_l$ . The approximate optimal angle  $\theta_i$  is found in the following two stages:

- the whole angular space  $[0, 2\pi]$  is split into  $N_l$  equal-sized regions of width  $\frac{2\pi}{N_l}$ , and then a *proposal angle*  $\theta_i^{(prop.)}$  is set to the lower boundary of each region. For each of the proposal angles, the approximate local log-likelihood is computed, and then the angle with the highest  $\log \mathcal{L}_i^{(approx.)}$  is selected as the approximate  $\theta_i^{(approx.)}$ ;
- given the approximate  $\theta_i^{(approx.)}$ , create 300  $\frac{2\pi}{N}$ -sized angular regions from each side of the  $\theta_i^{(approx.)}$ , and set a proposal angle  $\theta_i^{(prop.)}$  as the lower boundary of the region for each of them. Compute the *full* local log-likelihood for each of these proposal angles, compare it to the full local log-likelihood corresponding to the  $\theta_i^{(approx.)}$ , and select the angle that corresponds to the highest value of  $\log \mathcal{L}_i^{(full)}$ . This angle is selected as the optimal angle  $\theta_i$ .
- \* add the node to the visiting order vector.
- Shuffle the node visiting order vector.
- Perturb angular positions of nodes added to the subgraph  $G_l$ . First, set the *noise amplitude* to  $a = \frac{\pi}{4} \left(1 - \frac{l}{m}\right) + 0.01 \frac{l}{m}$ . Then, update each angle  $\theta_i$  as

$$\theta_i \leftarrow \theta_i + aX_i,$$

where  $X_i$  is sampled uniformly from  $(-\frac{\pi}{2}, \frac{\pi}{2})$ .

- Infer angular coordinates of nodes after the perturbation running the:
  - \* **full local log-likelihood angular inference**, as described above, if the number of nodes in the current core subgraph  $G_l$  is less than 500, or the total number of nodes in the largest connected component  $G$  is less than 1,500;
  - \* otherwise, **approximate local log-likelihood angular inference**, as described above.

Run angular inference for all nodes for 10 rounds or until the maximum angular displacement  $\Delta\theta$  for *any* node in  $G_l$  after each round does not exceed a pre-defined threshold  $\epsilon$ , i.e.,  $\Delta\theta < \epsilon$ , where  $\epsilon = 0.0001$  radians. During these rounds, the number of angular regions used to probe the local likelihood of each node  $i \in G_l$  is set to the number of nodes in  $G_l$ .

- After all nodes in the first  $\lfloor (m-1)/2 \rfloor$  layers are embedded, record the total log-likelihood and the resulting angular configuration.

Select the angular configuration corresponding to the replica with the highest total log-likelihood.

7. **Phase 2: embedding the rest of layers of nodes with  $k > 1$ .** Start with the angular configuration of the best replica from the previous phase. For the second half of layers  $l = \lfloor (m-1)/2 \rfloor + 1, \dots, m-1$ , do:

- for each node  $i$  in the layer  $l$ , find the maximum-likelihood angle  $\theta_i$  as follows:
  - if the number of nodes in the current core subgraph  $G_l$  is less than 500, or the total number of nodes in the largest connected component  $G$  is less than 1,500, use the **full local log-likelihood** to estimate the optimal  $\theta_i$ , as in Phase 1.
  - else, use the **approximate local log-likelihood** to estimate the optimal  $\theta_i$ , as in Phase 1.
  - add the node to the visiting order vector.
- Shuffle the node visiting order vector.
- Perturb angular positions of nodes added to the subgraph  $G_l$ . First, set the *noise amplitude* to  $a = \frac{\pi}{4} \left(1 - \frac{l}{m}\right) + 0.01 \frac{l}{m}$ . Then, update each angle  $\theta_i$  as

$$\theta_i \leftarrow \theta_i + aX_i,$$

where  $X_i$  is sampled uniformly from  $(-\frac{\pi}{2}, \frac{\pi}{2})$ .

- Infer angular coordinates of nodes after the perturbation running the:
  - **full local log-likelihood angular inference**, as described above, if the number of nodes in the current core subgraph  $G_l$  is less than 500, or the total number of nodes in the largest connected component  $G$  is less than 1,500;
  - otherwise, **approximate local log-likelihood angular inference**, as described above.

Run angular inference for all nodes for 10 rounds or until the maximum angular displacement  $\Delta\theta$  for *any* node in  $G_l$  after each round does not exceed a pre-defined threshold  $\epsilon$ , i.e.,  $\Delta\theta < \epsilon$ , where  $\epsilon = 0.0001$  radians. During these rounds, the number of angular regions used to probe the local likelihood of each node  $i \in G_l$  is set to the number of nodes in  $G_l$ .

8. **Phase 3: embedding the  $k = 1$  nodes.** For each of  $k = 1$  nodes  $i$ , run the **approximate local log-likelihood** angular inference for 1 round *without* angular noise perturbations, and add the node to the visiting order vector afterwards.
9. **Phase 4: “massaging” of nodes.** After all nodes are placed at angular positions, run the **approximate local log-likelihood** angular inference for all nodes for 20 rounds. Shuffle the node visiting order vector before each round. At this stage, no noise is added to angular positions of nodes.
10. **Small noise addition.** A small random noise of maximum amplitude of  $10^{-4}$  radians is added to all nodes to avoid the possibility of two nodes having exactly the same angular coordinates.
11. **Saving nodes’ coordinates.** The resulting radial and angular coordinates are saved to the output file. The output file contains the following information for each node: node’s label, node’s degree, node’s inferred radial coordinate, and node’s inferred angular coordinate.