Hyperbolae Are No Hyperbole: Modelling Communities That Are Not Cliques

Saskia Metzler
Max Planck Institute for Informatics
Saarland Informatics Campus, Germany
smetzler@mpi-inf.mpg.de

Stephan Günnemann
Dept. of Informatics & Inst. for Advanced Study
Technical University of Munich, Germany
guennemann@in.tum.de

Pauli Miettinen
Max Planck Institute for Informatics
Saarland Informatics Campus, Germany
pmiettin@mpi-inf.mpg.de

***Abstract*—Cliques are frequently used to model communities: a community is a set of nodes where each pair is equally likely to be connected. But studying real-world communities reveals that they have more structure than that. In particular, the nodes can be ordered in such a way that (almost) all edges in the community lie below a hyperbola. In this paper we present three new models for communities that capture this phenomenon. Our models explain the structure of the communities differently, but we also prove that they are identical in their expressive power. Our models fit to real-world data much better than traditional block models or previously-proposed hyperbolic models, both of which are a special case of our model. Our models also allow for intuitive interpretation of the parameters, enabling us to summarize the shapes of the communities in graphs effectively.**

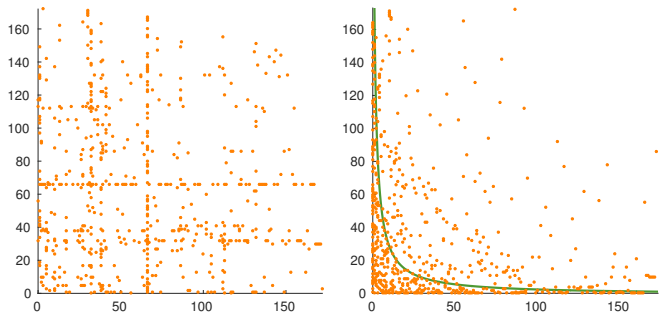
I. Introduction

Community detection in graphs has gathered significant research interest in recent years. So far, most approaches have (explicitly or implicitly) modelled communities as (quasi-) cliques, i.e. sets of nodes where every node is connected to (almost) every other node, that is, every edge within the community is equally likely.

We argue that a clique is often not a realistic model for real-world communities. Consider the adjacency matrix of a community from the YouTube data from the Stanford Large Network Dataset collection [10] in Figure 1a. In its original ordering, the community does look like a quasi-clique. But if we order the nodes by their degree (Figure 1b), we see immediately that the community takes the shape of a hyperbola: there is a clear curve such that it is much more likely to see an edge below this curve than above it. Hence, *not every edge between a pair of nodes in a community is equally likely*; a phenomenon that has been observed in multiple real-world graphs [3].

While [3] introduces an initial model to address this aspect, the proposed model is very restricted, capturing only certain shapes of communities and ignoring the inherent sparsity of real world networks (see Sections II and III-E for a detailed discussion). To overcome these limitations, we propose a novel model for communities that explicitly captures the shape of the edge distribution and that incorporates sparsity in its probabilistic formulation. Most importantly, our model contains (quasi-) cliques as well as the model of [3] as a special case.

In fact, we present three different models that allow us to capture different features of the communities and apply different optimization techniques when fitting the model in



(a) Unordered. (b) Degree ordered.

Figure 1. Adjacency matrix of a community from the YouTube data unordered and ordered by induced degree with a model fitted.

Section III. We will also prove that these three models are equivalent in the sense that a community in one model can be easily transformed to an equivalent community in another model. Yet, the models are not redundant, as the different views they provide allow for more in-depth understanding of the communities (and the models, as well) – and they show how our approach generalizes existing models.

In addition to modelling individual communities, we also present a model for the full graph given a set of communities in Section IV. In Section V we study the computational complexity of some problems related to the modelling and in Section VI we present our algorithms for fitting the models to individual communities and the full graph alike.

Our experimental evaluation, in Section VII, shows that we can efficiently model real-world graphs, yielding significantly more likely models than what we can get by modelling the communities as quasi-cliques or existing hyperbolic models. We will also demonstrate how our models will allow us to gain more understanding on the structure of the communities.

Our main contributions in this paper are

- We present three different but equivalent models for capturing non-uniform edge distributions inside communities.
- We show how to fit the models to the communities, and how to fit a model for the full graph to a graph with a set of communities.
- We show that our approach explains real graphs much better than traditional quasi-clique based models and also improves the model of [3].

II. Related Work

Nodes in real-world networks – including social networks and gene-regulatory networks – often organize into communities or clusters. A variety of methods have been introduced in the literature for finding the communities [1]. Even though the proposed approaches seem highly diverse, the vast amount of previous community detection methods have been either explicitly or implicitly aimed at detecting *block-shaped* areas of *uniform density* in the adjacency matrix. This includes prominent techniques such as stochastic block-models [2], [16], affiliation network models [20], pattern based techniques (e.g. detection of quasi-cliques [4], [8]), or cross-associations [5].

In this paper, following the observations of [3], we argue that communities in real networks do not show such a density profile. As illustrated in Figure 1, they are better represented by using a hyperbolic model. Our model captures that the density within the group is not uniformly distributed – some nodes show stronger connectivity among its peers.

Overlapping community detection: Unequal connectivity structure has also been considered in the area of overlapping community detection. Here, in contrast to classical partitioning approaches, each node might belong to multiple groups. As, e.g., noticed in [20], nodes participating in multiple communities lead to areas of higher density. Accordingly, one might conclude that these approaches can handle our scenario of hyperbolic communities. This conclusion, however, is incorrect.

Given a certain community (i.e. a set of nodes), models following the idea of affiliation networks [20], generate edges following the same probability; thus, leading to block-like structures in the adjacency matrix. Similarly, techniques such as mixed-membership block models [2] assume uniform density per community. Indeed, all these methods are highly related to the principle of Boolean Matrix factorization [14] – and its goal of finding overlapping *blocks* in binary matrices. Clearly, handling overlap and handling hyperbolic structure are two different aspects. As we will discuss in Section IV, for hyperbolic communities itself, three different types of (non-) overlap can be considered.

Hyperbolic community detection: The aspect of non-block communities has been discussed in [3]. The proposed model, called HyCom, still does not capture real scenarios well due to two reasons: First, the modeled hyperbolic communities are restricted in their possible shapes. That is, not all patterns appearing in real data can be well represented by the model. Indeed, as we will show formally in Section III-E, our model contains [3] as a special case. Second, [3] assumes a density of 100% inside a community, thus violating the general property of sparsity. In contrast, our model allows varying density among the different communities. The benefits of our model are also clearly confirmed in the experimental analysis. Nested matrices, of which hyperbolic graphs are a special case, have nonnegative rounding rank 1 [15], suggesting that rounding rank decompositions could provide an approach for finding hyperbolic communities. Yet, the connection to nested matrices does not generalize to higher rounding ranks.

Other graph patterns: For the purpose of graph compression, other graph patterns going beyond quasi-cliques have been considered. In [11], graphs are considered as a collection of hubs connecting to spokes. These hubs are recursively connected to super-hubs and so on. Extending this idea, the work [9] compresses graphs by using patterns such as stars or bipartite cores. None of these works exploits the idea of hyperbolic community structure.

III. Models for Communities

Our goal is to model the aforementioned structure in communities and to that end we present three different models. It should be noted that all these models contain the (quasi-) cliques and the model of [3] as special cases (see Section III-E). We will also show that these three different models – with different intuitions behind them – are equivalent representations besides having different parameterizations. These different models not only give three different interpretations but also enable ways to easily compare with other approaches and to efficiently compute their fit to given data.

A. General modelling decisions

Our input is an undirected graph $G = (V, E)$ with n nodes and m edges. We will assign a number from $\{0, \dots, n-1\}$ to the vertices and use (i, j) to denote both a pair of vertices and the (potential) undirected edge between them. We will represent the graph using its *adjacency matrix* $\boldsymbol{A} = (a_{ij}) \in \{0,1\}^{n \times n}$.

A *community* C is a tuple (V_C, π_C, Θ_C). The set $V_C \subseteq V$ contains the nodes in the community, and we write $n_C = |V_C|$. The permutation $\pi : V_C \to \{0, \dots, n_C - 1\}$ orders the nodes. In general, we assume the nodes to be ordered according to their degrees inside the community. The crucial part of our model is the following: not every edge between the nodes in V_C is necessarily part of our community – that assumption would make all of our communities quasi-cliques. Rather, our community models are defined using functions $f : \{0, \dots, n_C - 1\} \times \{0, \dots, n_C - 1\} \times \Theta \to \{0, 1\}$ operating on a set of parameters Θ and deciding for any pair of vertices $(i, j) \in \{0, \dots, n_C - 1\} \times \{0, \dots, n_C - 1\}$ if an edge between i and j is part of the community or not. We will define these functions in the subsequent sections.

Notice that the function f only gets the indices relative to the subgraph, not to the full graph, that is, to test a pair $(i, j) \in V_C \times V_C$, we need to compute $f(\pi_C(i), \pi_C(j), \Theta_C)$. For brevity, we will often omit the permutation and will simply assume that $i, j \in \{0, \dots, n_C - 1\}$.

Every community is associated with two sets of edges: the *area* of the community, A_C, defined as $A_C = \{(i, j) \in V_C \times V_C : f(i, j, \Theta_C) = 1\}$, and the *edges* of it, $E_C = E \cap A_C$. For notational convenience, we also define their complements (with respect to the community and the area, respectively): $\overline{A_C} = (V_C \times V_C) \setminus A_C$ and $\overline{E_C} = A_C \setminus E$.

Probabilistic model for a community: In practice the communities are rarely, if ever, exact. That is, some edges $(i, j) \in A_C$ are not in E_C, and some edges $(i, j) \in E$ that go between the nodes of the community are not in A_C. To

model these imperfect communities, we consider a *probabilistic model* of the community. Given a community C, we assume that edges $(i, j) \in V_C \times V_C$ are drawn from a Bernoulli distribution, $a_{i,j} \sim \text{Bernoulli}(p_*)$, where $\boldsymbol{A} = (a_{ij})$ is the adjacency matrix of the graph, and p_* is the *density* of the area that the edge belongs to. For a single community, we have two kinds of areas: the area of the community A_C and its complement $\overline{A_C}$. We denote the density of the area of the community by

$$d_C = |E_C| \,/\, |A_C| \ , \tag{1}$$

and the density of the area outside the community by

$$d_O = |E \cap \overline{A_C}| / \left|\overline{A_C}\right| \ . \tag{2}$$

These densities correspond to the maximum-likelihood solutions of the variables p_* for the edges that are inside or outside of the community. We can now consider the likelihood of the subgraph induced by the community $G|_{V_C}$ given the community C, $L(G|_{V_C} \mid C)$. The likelihood of an edge that is in community C is d_C; the likelihood of a pair (i, j) that is in the area of C but that is not an edge of G is $1 - d_C$; the likelihood of an edge that is not in the community is d_O; and the likelihood of a pair (i, j) that is not in the community's area and is not an edge of G is $1 - d_O$. This gives us

$$\begin{aligned} \log L(G|_{V_C} \mid C) = {} & |E_C| \log(d_C) + |\overline{E_C}| \log(1 - d_C) \\ & + |E \cap \overline{A_C}| \log(d_O) \\ & + |\overline{A_C} \setminus E| \log(1 - d_O) \ . \end{aligned} \tag{3}$$

B. Area restricted under a hyperbola

For our first model, we can notice from Figure 1 that when the nodes of a community are ordered in the induced degree order, the edges lie under a hyperbolic curve. To define the hyperbola we identify the vertex indices of the community as points in x and y axes. We use i and j instead of x and y to emphasize this connection. We will only consider the area $[0, n_C - 1] \times [0, n_C - 1]$ from the non-negative quadrant, as that is where the values important to our community are. The equation for a (rectangular) hyperbola is

$$(i + p)(j + p) = \theta \ , \tag{4}$$

with the centre at $(-p, -p)$. Figure 2 illustrates one hyperbola. Following the model, an edge (i, j) is considered to be in the community if $(i + p)(j + p) \leq \theta$. We call this model $\texttt{hyperbolic}(p, \theta)$ and write $(i, j) \in \texttt{hyperbolic}(p, \theta)$ if $(i + p)(j + p) \leq \theta$.

From Figure 2 we can gain some intuition to the parameters p and θ: different values of p will yield different shapes of the gradient (the coloured background in Figure 2), while different values of θ will move the line away from the origin.

Valid range of parameters: The values (4) assigns to elements (i, j) attain their minimum at the centre $(-p, -p)$. To make sure that all elements $(i, j) \in \mathbb{N} \times \mathbb{N}$ that are under the curve (4) are always in the community, we must bound p. A simple boundary is to enforce that $p \geq 0$, though in the next section we will derive a more relaxed boundary. Other than this boundary, p and θ can be any values.

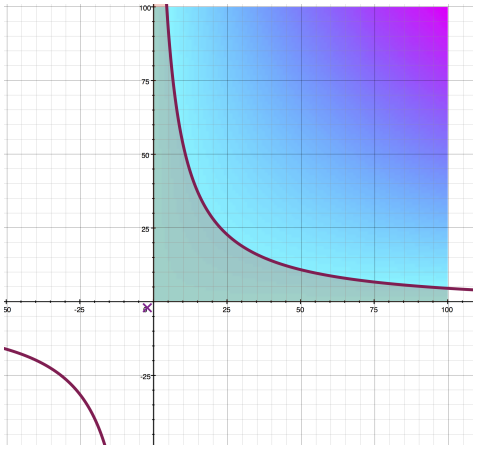


Figure 2. A hyperbola with $p = 2.06$ and $\theta = 673$ (dark red lines). The centre $(-p, -p)$ is marked with a cross. The colours in the background of the nonnegative quadrant indicate the values of $(i + p)(j + p)$ for $i, j \in [0, 99]$, with higher values moving from cyan to magenta. The area of the community is the solid-coloured area under the curve.

C. Fixed points in the curve

The shape of the hyperbola is not easy to interpret from (4), and hence it is not easy to say, by just looking at the parameters p and θ, whether the community is 'fat' or 'skinny'. To make the model parameters more interpretable, we can consider two points in the curve: the point at which it crosses the diagonal (i.e. when $i = j$), and the point at which the hyperbola exits the community (i.e. j for which $i = n_C$ or vice versa). We call the former γ and the latter H, and we can consider them as two values that define some p and θ such that

$$(\gamma + p)(\gamma + p) = \theta \tag{5}$$

$$(H + p)(n_C - 1 + p) = \theta \ . \tag{6}$$

To interpret the parameters, it is helpful to divide every community into two parts: *core* and *tail*. The core consists of nodes that form a (quasi-) clique, while the tail consists of nodes that are mainly connected to the core. This is illustrated in Figure 3 where the core is shaded in dark blue and the tail in light red. The parameter γ is the size of the core (minus 1 to account for zero-based indexing) – the larger γ, the larger clique the community has – while the parameter H tells how 'fat' the tails are. A (quasi-) clique would have large γ and H, while a star would have $\gamma = H = 0$. We will call this model $\texttt{fixed}(\gamma, H)$.

Equivalence: Given equations (5) and (6), it is hardly surprising that `fixed` is equivalent to `hyperbolic`. Formally, we can solve p and θ given γ, H, and n_C (the size of the community) as follows:

$$p = \frac{\gamma^2 - (n_C - 1)H}{(n_C - 1) + H - 2\gamma} \tag{7}$$

$$\theta = \frac{(\gamma - H)^2(\gamma - n_C - 1)^2}{(n_C - 1 + H - 2\gamma)^2} \ . \tag{8}$$

Similarly, we can easily work out the equations for γ and H given p and θ (and n_C). Notice that these equations also

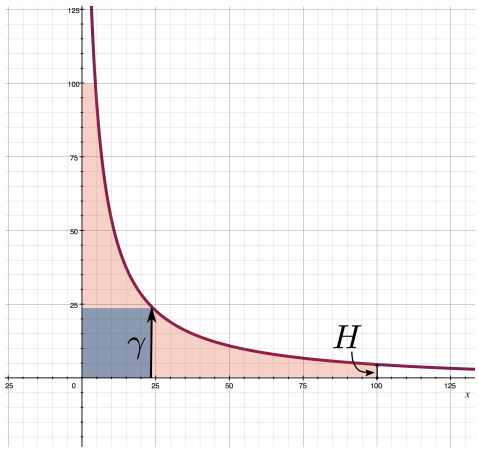


Figure 3. The parameter γ explains the size of the core of the community (dark-shaded box), while H explains the height of the tail at the end of the community.

give us a direct way to evaluate whether an edge (i, j) is in a community $\texttt{fixed}(\gamma, H)$: we only need to solve p and θ and evaluate if $(i + p)(j + p) \leq \theta$.

Constraints of parameters: Not every γ and H yield valid communities in (7) and (8), so we have to pay particular attention on the constraints of these parameters. Clearly both γ and H need to be nonnegative. As the hyperbola is monotonic, it must be that $H \leq \gamma$, and as the hyperbola is convex, it must be that $\gamma \leq (n_C - 1 + H)/2$.

Recall that in Section III-B we restricted p to be nonnegative so that it cannot happen that $(0+p)(0+p) > \theta$ if $(i+p)(j+p) \leq \theta$ for some $(i, j) \in \mathbb{N} \times \mathbb{N}$. The motivation for this was that we want element $(0, 0)$ to be included in every non-empty community. But we can relax the constraint $p \geq 0$ to

$$p^2 \leq \theta \Leftrightarrow p^2 \leq (\gamma + p)^2 \Leftrightarrow p \geq -\frac{\gamma}{2} , \tag{9}$$

assuming $\gamma > 0$. Here, the first equivalence follows by substituting (5) to θ. Notice that the p and γ in this inequality are bound together via (7). This constraint also implies the above constraint that $H \leq \gamma$.

D. A mixture of a line and a hyperbola

Given the understanding of the previous two models, we now introduce a third equivalent model. Here, we consider a mixture of two restricted models: a simple hyperbola where an edge (i, j) is in the community if $i \cdot j \leq \Sigma'$ for some $\Sigma' \in \mathbb{R}$, and a simple linear model $i + j \leq \Sigma''$, with $\Sigma'' \in \mathbb{R}$. Notice that unlike above, here the hyperbola's centre is fixed to the origin. Alone neither of these models is very expressive, but a mixture of these two is much more powerful: an edge (i, j) is in the community if

$$(1 - x)(i \cdot j) + x(i + j) \leq \Sigma$$

for some $x \in [0, 1]$ and $\Sigma \in \mathbb{R}$. Indeed, to allow even more flexibility, one can also consider a second mixture, which combines the hyperbola with a negative linear model

$$(1 - x)(i \cdot j) + x(-i - j) \leq \Sigma$$

for some $x \in [0, 1]$. Combining both of the above equations into a single model leads to our final definition: an edge (i, j) is in the community if for some $x \in [-1, 1]$ and $\Sigma \in \mathbb{R}$

$$(1 - |x|)(i \cdot j) + x(i + j) \leq \Sigma . \tag{10}$$

The meaning of the parameters here is slightly different to the model `hyperbolic`: the parameter Σ behaves similarly to θ in (4), moving the line (or the hyperbola) further from the origin, while the mixture parameter x dictates how much the model looks like a line and how much like a hyperbola centered to the origin. We call this model $\texttt{mixture}(x, \Sigma)$.

Equivalence: We will now turn our attention to the equivalence between `hyperbolic` and `mixture`.

Proposition 1. *For any pair (p, θ) of valid parameters of the `hyperbolic` model, there is a pair (x, Σ) of valid parameters of the `mixture` model that yield exactly the same community, and vice versa.*

Proof. We have

$$\begin{aligned} &(i + p)(j + p) \leq \theta \\ \Leftrightarrow\ &\frac{(i + j)p}{1 + |p|} + \frac{ij}{1 + |p|} \leq \frac{\theta - p^2}{1 + |p|} \\ \Leftrightarrow\ &(i + j)\frac{p}{1 + |p|} + ij\left(1 - \left|\frac{p}{1 + |p|}\right|\right) \leq \frac{\theta - p^2}{1 + |p|} , \end{aligned} \tag{11}$$

where the first equivalence is by expanding the left-hand side and re-arranging the terms and the second is by noting that

$$\frac{1}{1 + |p|} = \frac{1 + |p| - |p|}{1 + |p|} = 1 - \frac{|p|}{1 + |p|} = 1 - \left|\frac{p}{1 + |p|}\right|$$

If we now write $x = p/(1 + |p|)$ and $\Sigma = (\theta - p^2)/(1 + |p|)$, we get

$$(i + j)x + ij(1 - |x|) \leq \Sigma , \tag{12}$$

concluding the proof. □

E. Generalization of Quasi-Cliques and HyCom

An important consideration in our model(s) is that we want to be able to generalize existing models.

Quasi-cliques: Clearly, quasi-cliques are a special case of our model. Using the `fixed` model, we can simply set H and γ to n_C.

HyCom: In [3], a community has been defined as follows: Given $\alpha < 0$ and $\tau \in \mathbb{R}$, all edges (u, v) that fulfill

$$u^\alpha \cdot v^\alpha \geq \tau \tag{13}$$

are part of the community. Note that in [3], one-based indexing is used, i.e. the indices of nodes u, v start with 1. Thus, using our zero-based indexing w.r.t. i, j, (13) is equivalent to

$$(i+1)^\alpha \cdot (j+1)^\alpha \geq \tau \Leftrightarrow 0.5 \cdot (i \cdot j) + 0.5 \cdot (i + j) \leq \tau' . \tag{14}$$

Here, we set $\tau' = 0.5 \cdot (\tau^{1/\alpha} - 1)$ and exploited that $\alpha < 0$.

Using our `mixture` model, it is now obvious that [3] is a special case: it corresponds to $\texttt{mixture}(0.5, \tau')$. Indeed, while at first sight (13) seems to have two degrees of freedom, it only has one. The parameter τ' (in our notation: Σ) can be adapted, the parameter x is fixed to 0.5.

This restriction in the HyCom model limits the possible shapes of the communities significantly since communities with $x \neq 0.5$ can not be represented. Indeed, as we will show in the experiments, many real world datasets contain communities with x not close to 0.5 (see Figure 5c); thus, they are much better represented by our model. Furthermore, the HyCom model does not give us intuitive interpretation of the shape of the community as the exponent α can be switched to any other exponent by adjusting τ accordingly.

IV. Model for the Full Graph

While the previous section focused on modelling individual communities, we now introduce a principle for describing the full graph containing multiple communities. When multiple communities are present, we might observe overlapping groups. When communities are modelled as quasi-cliques, there is only one type of overlap we need to consider: if the nodes of two communities overlap, so do their (implicit) edges. With our community models, however, we have to distinguish three types of overlapping behaviour: two communities C and D are *node-disjoint* if they do not share any nodes ($V_C \cap V_D = \emptyset$); *area-disjoint* (but *node-overlapping*) if they do share nodes but no (implicit) edges ($V_C \cap V_D \neq \emptyset$ but $A_C \cap A_D = \emptyset$); and *area-overlapping* (or *overlapping* for short) if also their (implicit) edges overlap ($A_C \cap A_D \neq \emptyset$).

Area-overlapping communities present a particular challenge to the modelling as we have to assign a likelihood to every (implicit) edge. In this work we assign each edge to at most one community and the likelihood of the edge is calculated using only that community.

For defining the quality of a set of communities, we refer to a probabilistic approach, i.e. we aim to find the set of models $\mathcal{C}$ leading to the highest *likelihood* of the input graph G. For this purpose, notice that in real graphs the communities rarely have full density (i.e. $|E_C| \neq |A_C|$), and that the density varies between the communities.

We use the model from Section III-A as the basis for modelling a community. That is, the density of a community C, d_C, is defined as in (1), with every community having its own density. To define the outside density d_O, we have to consider not only the 'outside area' of a single community, but the whole area of the graph that does not belong to any community. If we let $A_{\mathcal{C}} = \cup_{C \in \mathcal{C}} A_C$ be the area that belongs to the communities and let $\overline{A_{\mathcal{C}}} = (V \times V) \setminus A_{\mathcal{C}}$ be its complement, then

$$d_O = |E \setminus A_{\mathcal{C}}| \,/\, \left|\overline{A_{\mathcal{C}}}\right| \ .$$

We can now model the full graph similarly to how we modelled a single community to obtain the overall likelihood $L(G \mid \mathcal{C})$ of a graph G given the set of communities $\mathcal{C}$.

Definition 1. Given a graph G and a collection of its communities $\mathcal{C}$, where every edge belongs to at most one community, the *log-likelihood* $\log L(G \mid \mathcal{C})$ is defined as

$$\begin{aligned}\log L(G \mid \mathcal{C}) = \sum_{C \in \mathcal{C}} \big(&|E_C| \log(d_C) + |\overline{E_C}| \log(1 - d_C)\big) \\ &+ |E \setminus A_{\mathcal{C}}| \log(d_O) \\ &+ \left(\left|\overline{A_{\mathcal{C}}}\right| - |E \setminus A_{\mathcal{C}}|\right) \log(1 - d_O) \ .\end{aligned} \tag{15}$$

V. Computational Complexity

Before we present our algorithms, let us briefly study the computational complexity of the problems related to the modelling. Instead of dealing directly with the likelihood, in this section our target is to minimize the number of non-edges inside the communities while simultaneously maximizing the number of non-edges outside (i.e. to maximize the community density while minimizing the outside-area density). This is a natural surrogate for the likelihood that allows us to avoid some issues in the analysis caused by the likelihood function (e.g. that the likelihood model is oblivious to the 'inside' and 'outside': very sparse communities with dense outside area are also good models in likelihood's sense).

We will first consider problems involving only a single community, showing that finding the node sets for communities is hard in our model:

Proposition 2. *Given a graph $G = (V, E)$ and a pair of parameters (p, θ), it is* NP*-hard to find*

- *the largest set of nodes $V_C \subset V$ and a permutation $\pi_C : V_C \to \{0, \ldots, |V_C| - 1\}$ such that the area A_C defined by V_C, π_C, and `hyperbolic`(p, θ) is* exact*, that is $A_C = E_C$;*
- *the set of nodes $V_C \subset V$ and a permutation $\pi_C : V_C \to \{0, \ldots, |V_C| - 1\}$ such that $d_C \geq c$ for some given constant $c \in (0, 1)$ and d_O is minimized.*

Proof. These results follow from the fact that the clique is a special case of our model. Hence, if (p, θ) are set so that they encode a clique, the first case is equivalent to the well-known NP-hard problem of finding the largest clique [7, Problem GT19], while the second is equivalent to the problem of finding the maximum c-quasi-clique, which is also NP-hard [17]. □

Let us now turn our attention to the case where we are already given a collection $\mathcal{C}$ of communities (with fitted models), and we want to find a subcollection $\mathcal{S} \subseteq \mathcal{C}$ that minimizes the number of edges in the outside area plus the number of non-edges inside the communities. That is, we want to minimize

$$\left|E \cap \overline{A_{\mathcal{S}}}\right| + |A_{\mathcal{S}} \setminus E| \ . \tag{16}$$

For these results, we use the general framework of Miettinen [13]. First note, that our communities are (symmetric) generalized rank-1 matrices in the sense of Miettinen: the functions f of our models define the outer product in Definition 1 of [13], while the adjacency matrix is the data matrix. For this problem, we only care about the union of the areas of the communities, and consequently, we take the element-wise disjunction of the matrices representing their areas. Propositions 6 and 10 of [13] directly provide the following results:

Proposition 3. *Given a graph $G = (V, E)$ and a collection $\mathcal{C}$ of communities of G,*

- *it is* NP*-hard to find the subcollection $\mathcal{S} \subseteq \mathcal{C}$ that minimizes* (16)*;*
- *it is* NP*-hard to approximate the error* (16) *to within a factor of $\Omega\left(2^{\log^{1-\varepsilon}|V|}\right)$ and quasi-*NP*-hard to approximate it within $\Omega\left(2^{(4\log|\mathcal{C}|)^{1-\varepsilon}}\right)$ for any $\varepsilon > 0$;*
- *the error* (16) *can be approximated to within a factor of $2\sqrt{(|\mathcal{C}| + |V|)\log|V|}$ in polynomial time.*

The situation of Proposition 3 can easily arise as the consequence of the following simple idea of finding the communities: first, find many subset of nodes (e.g. by sampling or by enumerating all dense subgraphs), then fit the community models to them, and then select the final subset of communities from the potentially highly redundant set of communities. As Proposition 3 shows, the last step of this approach is computationally hard and hence we do not use it.

VI. Algorithms

Next we present an algorithm for fitting our model to a graph. We assume the input is the graph and an initial collection of sets of nodes that represent initial communities. These initial node sets can be found using any existing community-detection algorithm, e.g. HyCom [3]. We will first present the algorithm to fit the model to a single community, and then explain how to use that to fit the model for the full graph.

A. Modelling a single community

To model a single community, we use the `fixed` model with *integer* parameters γ and H. Given the intuition from Figure 3, this restriction is natural. We will also not lose too much, as the following proposition demonstrates:

Proposition 4. *Let $C = \texttt{fixed}(\gamma, H)$ be a community of n_C nodes defined by $\gamma \in \mathbb{R}$ and $H \in \mathbb{N}$, and let A_C be its area. Then there exists integer γ' such that if $D = \texttt{fixed}(\gamma', H)$ is the community defined by γ' and H, and A_D is the respective area, then $|A_C - A_D| \in \Theta(\gamma \ln(n_C))$.*

In other words, the difference in area between integer and non-integer γ grows only linearly with γ and logarithmically with the number of nodes in the community. The proof of Proposition 4 is postponed to the extended version [12].

Constraining ourselves to integer parameters would not alone solve much, as there still are $O(n_C^2)$ parameter configurations to study. Many of these configurations, however, can be pruned out as they would lead to infeasible communities and the pruned search space is small enough for exhaustive search.

To gain intuition on how much of the search space the constraints remove, let us consider Figure 4. The area of the plot is the area of all possible combinations of γ and H for some community size n_C. The yellow area can be ignored as in that area $H > \gamma$. But also both of the green areas can be ignored, as in those areas, the constraint $p \geq -\gamma/2$ is violated (see (9)). This pruning significantly reduces the different parameter configurations we need to test in the exhaustive search.

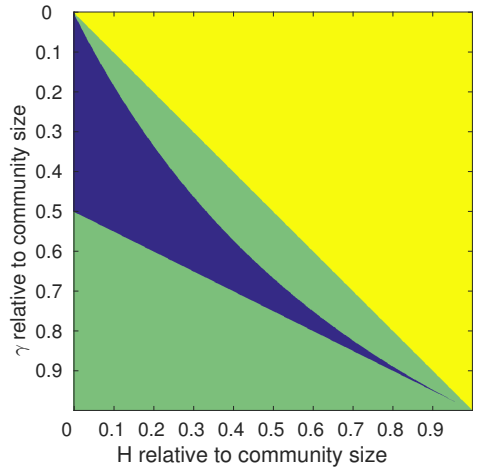


Figure 4. The blue area shows feasible values for parameters γ and H relative to a given community size. The green and yellow area are infeasible. The yellow part violates the trivial requirement of $H \leq \gamma$ and the green areas violate the condition $p \geq -\frac{\gamma}{2}$, where p is given by Equation (7).

Likelihood-Computation: Given a pair of parameters (γ, H), we need to evaluate its fit by computing the log-likelihood of the resulting model. According to the model proposed in Section III-A, this requires to determine the area inside and outside the current community as well as the corresponding number of edges (and non-edges) in these areas. Obviously, testing each position in the community is not a practical solution since it would lead to a running time quadratic in the community's size; instead we derive a solution which is linear in the number of edges.

The computation of the area can be done in time linear in the number of nodes by referring to the functional form of the hyperbola, i.e. evaluating $i = \frac{\theta}{j+p} - p$ for each column j. Here we can compute p and θ from γ and H using (7) and (8). Alternatively, we can approximate the area in constant time by taking the integral of this function from 0 to $n_C - 1$. Counting how many edges are inside the community requires a pass over the edges. Thus, this step dominates the time complexity.

It is easy to optimize this procedure further: First, we can compute the area faster by noticing that at the bottom we have a rectangle of size n_C-by-H. Second, when we test a succession of parameter values, we can re-use part of the information about the edges: by increasing the values of H or γ only edges previously outside the community need to be evaluated. All remaining edges will still be located within the community.

B. Modelling a full graph

To obtain a model for a graph consisting of multiple potentially overlapping communities, we aim to optimize the log-likelihood $L(G \mid \mathcal{C})$ for the full graph (i.e. (15)). Our main problem is to determine how to deal with overlapping communities; indeed, if there are no node-overlapping communities, we can simply optimize every community separately using the above approach. If the communities do overlap, however, we do need to decide the order of the communities so that we can assign every edge to at most one community.

As computing the log-likelihood for the full graph for each possible order of the communities is infeasible, we optimize the log-likelihoods of each community individually following an alternating optimization strategy. When optimizing one community, we keep track of the area that was already covered by other communities, and ignore that area in the computations of the subsequent communities. In concordance with the log-likelihood we want to optimize for, we consider the global density for the whole outside community area (see Section IV) in the log-likelihood computation of the model, instead of just the local outside density, as in Section III-A.

The algorithm, shown as Algorithm 1, comprises an initialization phase and an update phase. In the initialization phase of the algorithm (lines 1–5), we compute an individual model for each community, leaving out those edges that have already been covered in a previous step by another community. Note that during this step, each community uses its individual outside density. Next, we order the obtained models by their log-likelihood starting with the best and we compute the global outside density d_O for the further updates.

Now that we have established an order of the communities, the alternating optimization starts (line 7): Each time a community C_i is selected and a new model is fit to it – now not only excluding edges already covered in previous communities but also using the global outside density to determine the true log-likelihood for each community. After fitting the new model, we update the global outside density (line 13) if the new model is different to the old one.

All communities that have node overlap with C_i are marked: due to the update of C_i also the parameters of the overlapping communities might change. Thus, we mark the communities that overlap with C_i for a re-update (line 14). We iterate over this process of updating the community models until there are no more communities to be updated.

The output of this algorithm is a list of models for all communities, ordered by their log-likelihoods.

Algorithm 1 Algorithm to fit the `fixed` model to a graph.

Input: Undirected graph $G = (V, E)$, a collection of sets of nodes $\mathcal{V} = \{V_i \subset V\}$ describing the initial communities
Output: Ordered set of communities $\mathcal{C}$

1: **for** every set $V_i \in \mathcal{V}$ **do**
2: $\quad C \leftarrow$ best model describing $G|_{V_i}$
3: $\quad \mathcal{C} \leftarrow \mathcal{C} \cup \{C\}$
4: Order $\mathcal{C}$ based on the likelihoods of the communities
5: Compute the global outside density d_O
6: $\mathcal{F} \leftarrow \mathcal{C}$
7: **repeat**
8: $\quad \mathcal{T} \leftarrow \mathcal{F}$; $\mathcal{F} \leftarrow \emptyset$; $M \leftarrow \emptyset$
9: $\quad$ **for all** $C \in \mathcal{T}$ **do** ▷ In decreasing likelihood
10: $\quad\quad$ Update the model of C ignoring areas in M
11: $\quad\quad M \leftarrow M \cup A_C$
12: $\quad\quad$ **if** the likelihood of C improved **then**
13: $\quad\quad\quad$ Update d_O
14: $\quad\quad\quad \mathcal{F} \leftarrow \mathcal{F} \cup \{D \in \mathcal{C} : V_D \cap V_C \neq \emptyset\}$
15: $\quad\quad\quad$ Update the position of C in $\mathcal{C}$
16: **until** $\mathcal{F} = \emptyset$
17: **return** $\mathcal{C}$

Table I
SIZES OF THE DATASETS USED FOR EVALUATION, THEIR TOTAL NUMBER OF COMMUNITIES AS WELL AS THE NUMBER OF COMMUNITIES OF SIZE 100 TO 1000, AND THE TIME IT TOOK TO DETERMINE A HYPERBOLIC MODEL FOR A SAMPLE OF SIZE 500.

	nodes	edges	communities		time
			all	100–1000	(h)
Amazon	334,863	925,872	75,149	1,380	0.6
DBLP	317,080	1,049,866	13,477	805	27.0
Friendster	65,608,366	1,806,067,135	957,154	19,763	12.3
LiveJournal	3,997,962	34,681,189	287,512	8,769	11.3
Orkut	3,072,441	117,185,083	6,288,363	80,251	3.1
YouTube	1,134,890	2,987,624	8,385	129	0.8

VII. EXPERIMENTS

We divide our experiments into two groups. In the first group (Section VII-A and VII-B), we use graphs with human-annotated communities and our goal is to study the shape of these communities, and the differences between the block/HyCom models and our model. In the second group (Section VII-C) of experiments, we use existing community-detection algorithms to find the communities, and then apply our model to the found communities. The source code for our programs and the scripts to run the experiments are available online.[1]

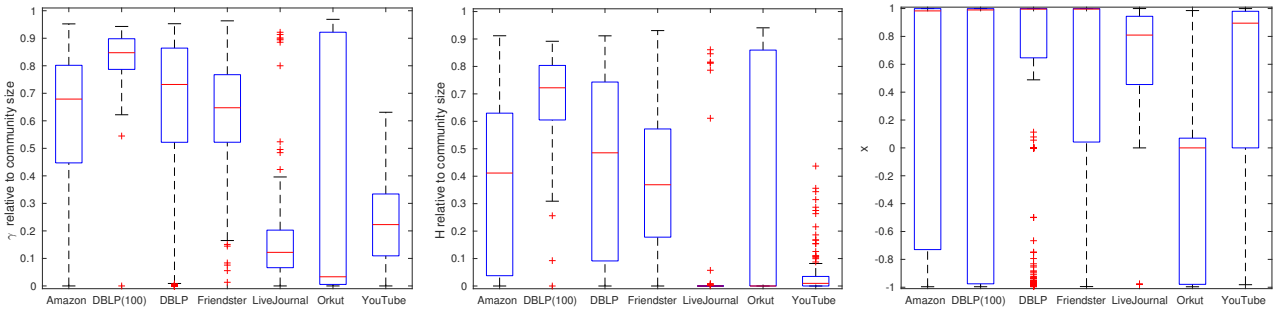
A. Obtained models

We start by studying the results of fitting our model to graphs where ground-truth communities are given. We used the Stanford Large Network Dataset collection [10], which offers datasets with known communities. Table I provides a summary of the employed datasets. As neither very small nor very large communities are particularly interesting, we restrict our analysis to communities with between 100 and 1000 nodes (inclusive). We use a sample of 500 communities from each of these data sets (for DBLP we also use a sample of 100 communities, see Section VII-B). Notice that the ground-truth information for the communities is only provided for the nodes and not for the edges.

Figure 5 summarizes the distributions of the parameters γ, H, and x for different data sets, respectively. The latter parameter we infer by converting the results into the `mixture` parametrization. For all box plots, the median is indicated as the central red mark and the edges of each box are the 25 and 75 percentiles. The whiskers extend to the most extreme data points not considered outliers.

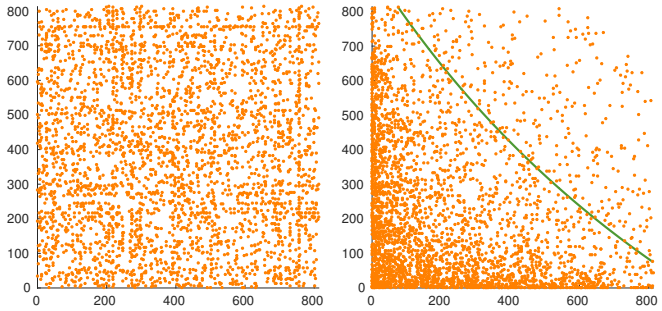
The box plots reveal a characteristic shape for each data set: Amazon, DBLP, and Friendster show rather thick communities with γ on median being more than 50% of the community size. LiveJournal, Orkut, and YouTube communities mostly have thin cores. While Orkut also has communities with big cores, LiveJournal is at the other extreme and its communities mostly exhibit a star-like shape. These observations are in line

[1] http://people.mpi-inf.mpg.de/~pmiettin/hybobo/



(a) Distribution of γ. (b) Distribution of H. (c) Distribution of x.

Figure 5. Distribution of selected parameters after fitting a hyperbolic model to sampled communities of size 100 to 1000 from each dataset. The sample size was 500 for all datasets. For DBLP, the result is shown also for a sample of 100 communities.



(a) Before. (b) After fitting the model.

Figure 6. Example of a community obtained from Amazon data.

with the intuitive idea about some of the data sets. DBLP, for instance, is the network of co-authorships and therefore many communities are almost quasi-cliques.

Furthermore, we observe that the obtained models for all datasets differ significantly from the HyCom model that would require $x = 0.5$ (see Section III-E). As Figure 5c shows, none of the medians is near $x = 0.5$. To gain intuition on how these communities look like, we give a few examples in Figures 1 and 6, for the YouTube and Amazon data, and further examples in the extended version [12].

Our algorithm is implemented in Matlab and C. It took between half an hour and a full day to compute the models on a machine with four Intel Xeon E7-4860 10-core CPUs running at 2.27 GHz and 256 GB of main memory. The exact running times are given in Table I. Notice that the time depends not only on the size of the graphs, but to a great extend on the amount of overlap between the communities. Overlapping communities have to be computed in a sequential manner as the model of one community has impact on the next. Additionally, an update to one model causes the re-computation of all communities that overlap with it. Hence, the computation for DBLP takes more than twice as much time than that for Friendster although the Friendster graph is orders of magnitude larger.

B. Comparison to block models and HyCom models

In Section III-E, we describe two special cases of our model: In one case, every community is assumed to be a quasi-clique yielding exactly one possible parameter configuration per community, i.e. $H = \gamma = n_C$. In the other case, the HyCom model, only the threshold parameter Σ may vary and the other is fixed to $x = 0.5$. In this section, we assess the benefit of the additional flexibility of our model by comparing to these restricted versions. To do so, we compare the log-likelihood, computed as in (15), of the hyperbolic models obtained as in Section VII-A to the log-likelihoods of the respective HyCom models and block models. The HyCom models of every data set were obtained by running Algorithm 1 but only admitting those (H, γ) combinations that yield $x = 0.5$. For the block models, no parameter search is necessary as there is only one admitted configuration per community.

To compare the log-likelihoods, we use the likelihood ratio test [19, Ch. 10.6]. In case of the block model, we test the null hypothesis H_0 that all parameters, i.e. all H and γ of all communities, are fixed to create the blocks versus the alternative hypothesis H_1 that the parameters are not fixed; for HyCom, we assume one free parameter. The likelihood ratio test statistics are given by $\lambda = 2\log\big(L(\text{our model})/L(\text{block model})\big)$ and $\lambda = 2\log\big(L(\text{our model})/L(\text{HyCom model})\big)$ for block and HyCom models, respectively. The results are shown in Table II.

Block model: For the block model, the derived p-values are with one exception always essentially zero and confirm that the hyperbolic model is statistically significantly better than the block model. The exception is the 500 sample of the DBLP data. For this data set, the block model gives a better likelihood than ours. While cliques are a special case of our model, and hence we can always model each community as a clique, the iterative method to update the model parameters (Section VI-B) is based on a greedy heuristic. In case of the DBLP data, the greedy heuristic has reached a local optimum that is less good than what could be obtained with pure block models. On one hand, this is partially because DBLP contains block-like communities, and on the other hand, the large overlaps between these communities might have lead the optimization astray. To test the latter hypothesis, we also sampled just 100 communities from DBLP: this reduced the amount of overlap between the communities (from 33% to 18%) and also significantly improved our algorithm's results.

Table II
TEST STATISTIC OF THE LIKELIHOOD RATIO TEST BETWEEN OUR MODELS AND BLOCK MODELS, AND OUR MODELS AND HYCOM. FOR ALL DATASETS 500 COMMUNITIES WERE SAMPLED. ADDITIONALLY, WE DISPLAY THE RESULT FOR SAMPLING 100 COMMUNITIES FROM THE DBLP DATA.

	LL ratio	
	block model	HyCom
Amazon	26450.6	30997.1
DBLP (100)	3148.5	-788.0
DBLP	-264974.7	17958.1
Friendster	200627.6	17811.7
LiveJournal	154982.4	22705.8
Orkut	11945.3	1598.5
YouTube	75689.6	12660.0

Table III
DATASETS USED FOR THE COMMUNITY FINDING EXPERIMENTS.

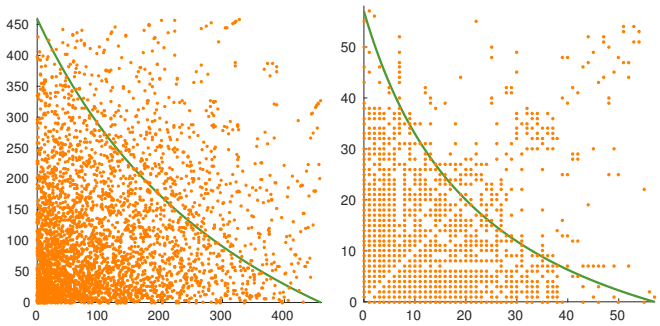
	nodes	non-zeros	content
Email	1,133	10,902	University email network
Erdős	472	2,628	Erdős collaboration network
Jazz	198	5,484.6	Network of Jazz musicians
PolBooks	105	882	Books about US politics

HyCom: For the comparison to the HyCom model, we obtain a similar result: With one exception, our hyperbolic model describes the data statistically significantly better than the HyCom model. The better solution for the 100 sample of DBLP found with the parameter space restricted to HyCom models is also a valid solution within our more general modelling framework. The greedy algorithm we propose, however, gives no guarantee to converge to the globally best solution and this result indicates that it depends on the data whether additional freedom in the parameter space is a benefit or hindrance *for the algorithm* to find a good solution. More importantly, as we will see in the next section, starting with HyCom as initialization, our model is always significantly better.

C. Finding communities

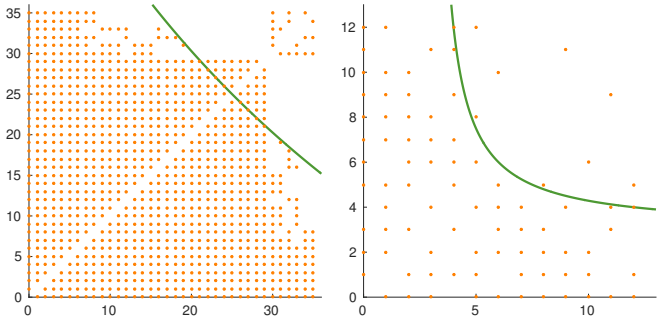
Next, we demonstrate that our model improves the description of communities returned by existing community-finding methods. We used spectral clustering, Boolean matrix factorization, and HyCom to find the communities; the first two approaches look for clique-like communities while HyCom looks for hyperbolic shapes (see Section III-E). We used various real-world data sets from the University of Florida Sparse Matrix Collection [6], summarized in Table III. They are significantly smaller than the data sets we examined in the previous experiments to allow the community detection algorithms to work efficiently. As we aim to assess the quality of the models and do not propose an own method for finding communities, no ground-truth community information was employed for the evaluation.

Spectral clustering: We first used spectral clustering with the normalized Laplacian [18] to cluster the nodes of the graph. The resulting communities are non-overlapping. We notice a significant benefit of modelling the obtained result by means of hyperbolic models, as the log-likelihood ratio test confirms for all examined datasets (see Table IV). These results yielded p-values that were essentially zero, confirming that the results are statistically significant. We have used $k = 10$ clusters for Email, $k = 8$ for Erdős, $k = 5$ for Jazz, and $k = 6$ for PolBooks. We display examples of modelled communities in Figure 7. These example communities show relatively large cores but thin tails, with most edges being in the lower triangular area. Our models clearly capture this phenomenon.



(a) Community from Email data. (b) Community from Jazz data.

Figure 7. Examples of communities fitted by our model. The initial communities were obtained using spectral clustering on the respective data.



(a) Community from Jazz data. (b) Community from PolBooks data.

Figure 8. Examples of communities fitted by our model. The initial communities were obtained using Boolean matrix factorization.

Boolean matrix factorization: To find overlapping communities, we used Boolean matrix factorization (BMF) [14]. We used the Asso algorithm [14] with the same number of communities as with spectral clustering. We set the threshold parameter τ of Asso to 0.6 and the weight w to 10. We again find that our models resemble the data significantly better than the corresponding block models (see Table IV). Figure 8 shows example communities from the PolBooks and Jazz data.

HyCom algorithm: We ran the HyCom algorithm [3] on each of the data sets stopping after we found k communities, with k specified as above. As the likelihood ratio test confirms (Table IV), our hyperbolic model improves the result of the HyCom algorithm.

D. Discussion

Our experiments have verified our intuition that the communities in real-world graphs are better modelled using our

Table IV
STATISTIC OF THE LIKELIHOOD RATIO TEST BETWEEN OUR AND BLOCK MODELS, AS WELL AS OUR AND HYCOM MODELS. THE COMMUNITIES ARE FOUND USING HYCOM, SPECTRAL CLUSTERING, AND BMF.

	LL ratio		
	spectral clustering	BMF	HyCom
Email	10895.8	3552.0	250.1
Erdős	1797.0	949.0	256.3
Jazz	3003.8	4435.0	3718.5
PolBooks	648.0	303.3	228.2

models than the traditional quasi-clique models, and that our models are an improvement over the previously proposed HyCom model [3]. This holds true for a variety of data sets, both with ground-truth communities, and with communities detected with existing methods. It is important to notice that the existing methods, especially the BMF, aim at finding clique-like communities. Thus, since our algorithm uses their results as the initial community candidates, any weaknesses of these algorithms will also affect our result. Still, our experiments show that our models provide statistically significantly better fit, even when we take into account the increased number of free parameters for our model.

Not only is our model a better fit for the data, it also provides interesting insights to the shape of the communities. The easy interpretability of the parameters γ and H means that we can simply study a summary of their distributions to gain an understanding on how the communities in a data look like, whether the cores are small or big and whether the tails are fat or skinny. This allows a data analyst to obtain a fast general understanding about the data without having to look at any particular community.

Finally, our experiments also demonstrate the scalability of our method. It had no problem of handling even the largest graph, Friendster, with approximately 65.6 million nodes and 1.8 billion edges.

VIII. CONCLUSIONS

We have proposed three novel models to describe hyperbolic communities. Based on the observation that communities in real-world graphs do not correspond to blocks of uniform density, our models capture the density distributions per community more accurately. We have shown that all models have the same expressive power, and we proposed an algorithm to fit these models to a given community – and likewise, to fit multiple, potentially overlapping, communities to represent the full graph.

In our experimental study, we have analysed a large variety of real-world datasets leading to interesting insights about the data's inherent community structure – showing variations from star-like data to data with patterns similar to quasi-cliques. Our hyperbolic model captures all these scenarios as special cases. Last, comparing the likelihood obtained by our model w.r.t. block-modelling approaches and existing hyperbolic models clearly shows the superiority of our hyperbolic community model for real-world data.

Future work: While our current model allows node overlapping communities, we have restricted edges to be part of at most one community. Extending our models and algorithms to handle edge overlaps is an important research direction we aim to investigate. Moreover, while this work focused on modeling a set of communities, we aim to investigate community detection algorithms able to detect hyperbolic structures directly. To that end, the aforementioned nonnegative rounding rank decompositions [15] provide an interesting approach.

ACKNOWLEDGEMENTS

This research was supported by the German Research Foundation (DFG), Emmy Noether grant GU 1409/2-1, and by the Technical University of Munich - Institute for Advanced Study, funded by the German Excellence Initiative and the European Union Seventh Framework Programme under grant agreement no 291763, co-funded by the European Union.

REFERENCES

[1] C. C. Aggarwal and H. Wang, editors. *Managing and mining graph data*. Springer, 2010.
[2] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. In *NIPS*, pages 33–40, 2009.
[3] M. Araujo, S. Günnemann, G. Mateos, and C. Faloutsos. Beyond blocks: Hyperbolic community detection. In *ECMLPKDD*, pages 50–65, 2014.
[4] B. Boden, S. Günnemann, H. Hoffmann, and T. Seidl. Mining coherent subgraphs in multi-layer graphs with edge labels. In *KDD*, pages 1258–1266, 2012.
[5] D. Chakrabarti, S. Papadimitriou, D. S. Modha, and C. Faloutsos. Fully automatic cross-associations. In *KDD*, pages 79–88, 2004.
[6] T. A. Davis and Y. Hu. The university of Florida sparse matrix collection. *ACM Trans. Math. Softw.*, 38(1):1–25, 2011.
[7] M. R. Garey and D. S. Johnson. *Computers and intractability: A guide to the theory of NP-Completeness*. W. H. Freeman, New York, 1979.
[8] S. Günnemann, I. Färber, B. Boden, and T. Seidl. Gamer: a synthesis of subspace clustering and dense subgraph mining. *Knowl. Inf. Syst.*, 40(2):243–278, 2014.
[9] D. Koutra, U. Kang, J. Vreeken, and C. Faloutsos. Summarizing and understanding large graphs. *Stat. Anal. Data Min.*, 8(3):183–202, 2015.
[10] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, 2014. Accessed 11 Feb 2016.
[11] Y. Lim, U. Kang, and C. Faloutsos. Slashburn: Graph compression and mining beyond caveman communities. *IEEE Trans. Knowl. Data Eng.*, 26(12):3077–3089, 2014.
[12] S. Metzler, S. Günnemann, and P. Miettinen. Hyperbolae are no hyperbole: Modelling communities that are not cliques. Technical Report arXiv:1602.0465, 2016.
[13] P. Miettinen. Generalized matrix factorizations as a unifying framework for pattern set mining: Complexity beyond blocks. In *ECMLPKDD*, pages 36–52, 2015.
[14] P. Miettinen, T. Mielikäinen, A. Gionis, G. Das, and H. Mannila. The discrete basis problem. *IEEE Trans. Knowl. Data Eng.*, 20(10):1348–1362, 2008.
[15] S. Neumann, R. Gemulla, and P. Miettinen. What you will gain by rounding: Theory and algorithms for rounding rank. In *ICDM*, 2016.
[16] K. Nowicki and T. A. B. Snijders. Estimation and prediction for stochastic blockstructures. *J. Amer. Statist. Assoc.*, 96(455):1077–1087, 2001.
[17] J. Pattillo, A. Veremyev, S. Butenko, and V. Boginski. On the maximum quasi-clique problem. *Discrete Appl. Math.*, 161(1-2):244–257, 2013.
[18] U. von Luxburg. A tutorial on spectral clustering. *Stat. Comput.*, 17(4):395–416, 2007.
[19] L. Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer, 2010.
[20] J. Yang and J. Leskovec. Community-affiliation graph model for overlapping network community detection. In *ICDM*, pages 1170–1175, 2012.