Visualizing Uncertainty in Probabilistic Graphs with Network Hypothetical Outcome Plots (NetHOPs)

Dongping Zhang, Eytan Adar, and Jessica Hullman

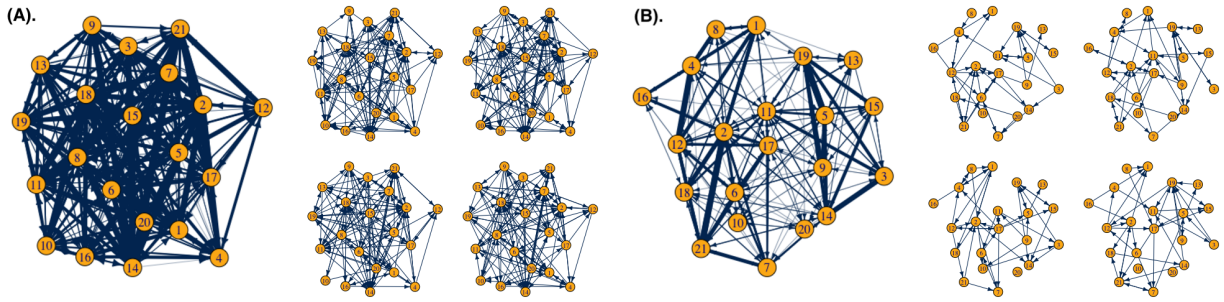

Fig. 1. (A) and (B) are comparisons between static probabilistic networks and sampled network realizations from a random graph model defined by probabilistically weighted edges. Both (A) and (B) consist of 21 vertices with corresponding density values of 0.9 and 0.5.

Abstract— Probabilistic graphs are challenging to visualize using the traditional node-link diagram. Encoding edge probability using visual variables like width or fuzziness makes it difficult for users of static network visualizations to estimate network statistics like densities, isolates, path lengths, or clustering under uncertainty. We introduce Network Hypothetical Outcome Plots (NetHOPs), a visualization technique that animates a sequence of network realizations sampled from a network distribution defined by probabilistic edges. NetHOPs employ an aggregation and anchoring algorithm used in dynamic and longitudinal graph drawing to parameterize layout stability for uncertainty estimation. We present a community matching algorithm to enable visualizing the uncertainty of cluster membership and community occurrence. We describe the results of a study in which 51 network experts used NetHOPs to complete a set of common visual analysis tasks and reported how they perceived network structures and properties subject to uncertainty. Participants' estimates fell, on average, within 11% of the ground truth statistics, suggesting NetHOPs can be a reasonable approach for enabling network analysts to reason about multiple properties under uncertainty. Participants appeared to articulate the distribution of network statistics slightly more accurately when they could manipulate the layout anchoring and the animation speed. Based on these findings, we synthesize design recommendations for developing and using animated visualizations for probabilistic networks.

Index Terms— Network, Uncertainty, Application

1 Introduction

Network data are prone to uncertainty. It is often unclear whether all relevant entities are included in a graph and if observed interactions are representative or occur merely by chance (e.g., [45, 92]). For example, social network data are frequently collected through surveys, but it is well-known in the Social Network Analysis (SNA) community that network surveys are problematic due to selection bias, response bias, and missing responses (e.g., [5, 90]). While technological affordances of online platforms reduce data uncertainty by making user interactions visible and persistent [27, 53, 54], uncertainty remains an issue when analysts binarize or predict social relations through statistical models based on the frequency of communication (e.g., [1]).

Uncertainty in the network analysis pipeline is sometimes addressed by imposing probabilities on edges as weights, resulting in a probabilistic graph. Although edge uncertainty can be visualized in a node-link diagram through visual encodings such as width, fuzziness, or grain [31, 58], the rendered graph is typically difficult to visually assess for practical use. For example, an analyst will likely find it challenging to respond to common graph analysis tasks, such as, "What is the most likely shortest path length between node 16 and node 9?", or "What is the expected density?" using either of the static visualizations shown in Fig. 1. In such cases, analysts must rely on a hard-to-decode visual channel not only to gain probability information about any single edge, which might be difficult to see due to the high density of a probabilistic graph (i.e., showing all edges with non-zero weights), but also to simultaneously integrate and process the joint probability from multiple edges for certain statistics (e.g., path lengths, isolates, and densities).

Similarly, although there are algorithms to identify clusters for static weighted networks (e.g., [55]), the uncertainty of community occurrence or cluster membership is difficult to visualize in a static diagram because traditional encodings (e.g., node coloring or convex hulls) are used to show deterministic community membership. These reasons may contribute to the relative lack of techniques available for visualizing probabilistic graphs to support exploratory network analysis.

To provide a solution, we introduce Network Hypothetical Outcome Plots (NetHOPs), a frequency-based uncertainty visualization technique that applies animated hypothetical outcomes [39] to probabilistic graphs. NetHOPs dynamically visualize a set of realizations sampled from a network distribution defined by probabilistic edges. By showing independent possible realizations of a network, NetHOPs avoid the challenges of supporting judgments about network properties under uncertainty with static encodings. Instead, structures and properties can be estimated by attending to the temporal frequency of occurrence from a set of independent but equally representative realizations.

A challenge arises when presenting network uncertainty as a se-

- *Dongping Zhang is with Northwestern University. E-mail: dzhang@u.northwestern.edu.*
- *Eytan Adar is with the University of Michigan. E-mail: eadar@umich.edu.*
- *Jessica Hullman is with Northwestern University. E-mail: jhullman@northwestern.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

quence of node-link diagrams: graph layout algorithms are intentionally optimized based on a static network. This induces a trade-off between more optimal visualization of each sampled realization and analysts' ability to preserve their mental maps [61] of how vertices and edges relate across realizations. NetHOPs address this trade-off using an offline graph drawing approach and applying an aggregation and anchoring algorithm [8] to enable analysts to control layout stability and readability [11]. To address the challenges associated with visualizing community membership under uncertainty, NetHOPs employ community detection algorithms designed for static unweighted networks, for which we develop a community matching and coloring algorithm.

To demonstrate the use of NetHOPs in a realistic analysis setting and explore how well the technique supports uncertainty perception on networks, we contribute a user study in which 51 network experts used NetHOPs to complete a set of common visual analysis tasks, chosen based on network task taxonomies [2,52]. The experts provided probability estimates and used distribution builders [82] to sketch their perceptions of uncertainty in network properties (e.g., density, shortest path, clusters). We find that NetHOPs allowed experts to visually assess the distributions of network structures fairly accurately: responses were within 11% of the ground truth distributions (defined on the full set of visualized realizations) across tasks. When participants were instructed to adjust NetHOP parameters like the amount of layout stability, the animation speed, and other visual properties, we see some evidence of a small (7%) additional average improvement in performance. We present an exploratory analysis of our results that we use to reflect on how different visual network estimation tasks under uncertainty can be best supported. Based on our findings, we synthesize design recommendations for future development of NetHOPs and similar techniques as a solution to address network uncertainty.

2 Background

2.1 Uncertainty Visualization

Prior work demonstrates that people can make better decisions when uncertainty is effectively represented [19,43,44,63], which makes uncertainty communication an important practice for scientific research [84]. Empirical evidence suggests that framing probabilities as frequencies (e.g., 3/10 rather than 30%) can make them easier to reason with [28,34]. Among other frequency-based uncertainty visualization techniques (e.g., [20, 26, 48]), hypothetical outcome plots (HOPs) [39, 47] display a finite set of samples from a distribution as a series of animated frames. Some studies of simple 2D visualizations find that HOPs can lead to better estimates than error bars [35, 39, 47] and other static techniques like static ensembles and violin plots [47].

We note several properties of HOPs that make them potentially interesting as a technique for displaying probabilistic graphs. First, HOPs are a natural choice for visualizing uncertainty when baseline visual encodings are already complex and hard to read, such that the addition of another encoding (e.g., width) or glyph (e.g., error bars), may not be effective. By using temporal frequency encoding, which perceptual research has found can be processed automatically without requiring counting [32], HOPs support intuitive estimation of event probabilities, which in a network application might include probabilities associated with structures of interest (e.g., edges, clusters, cliques). Finally, by depicting multivariate samples, HOPs make it possible for a single visualization to show joint probabilities, which typically requires adding additional views. While our network model does not require joint probability depiction, other probabilistic networks can (e.g., [40]).

2.2 Graph Visualization and Animation

Many network properties (e.g., centrality, clustering, density) have visual signatures, which make network visualization an important tool for network analysis. When well laid out, a node-link diagram can provide an effective bird's eye view of the network, making these diagrams a ubiquitous representation of graphs. Analysts use node-link diagrams to identify overall patterns such as cores and peripheries, communities or neighborhoods, and isolates and components, among others. While many structural configurations (e.g., reciprocity, triadic closure, or alternating path) can be summarized by network statistics or distributions [57], analysts typically begin the analysis by examining structural dependencies and correlations from a comprehensive overview of the graph, which visualizations can help to provide.

Graph layout algorithms, like force-directed (spring embedder) layouts [17], stress minimization [10, 25], and the stochastic gradient descent approach [93] can help to achieve aesthetic goals like minimizing edge crossing or overlapping nodes to support low-level visual tasks such as tracing each edge from source to target and seeing clusters [83].

Animated node-link diagrams are commonly used to visualize dynamic and longitudinal graphs [13], where each frame in the animation represents the graph at different states. It is possible but often impractical to show all graphs in a static visualization (e.g., small multiples [3]) due to large numbers of graphs. When designing network animations, it is insufficient to apply a layout algorithm to each state in a time-varying graph because the dynamic layout algorithm can produce inconsistent results when repeatedly applied to the same network data. Instead, layouts must be carefully engineered to help analysts preserve their mental maps of how the nodes and edges change over time [61, 69, 79].

Layout computation for dynamic network visualizations can be categorized into offline computation based on the whole sequence of graphs [9] and an online approach [8] that computes the graph layout one transition at a time [24, 66]. Layout stability can be addressed by anchoring vertices to fixed positions while animating edges [62] or by connecting vertices from different instances with external edges, which is conceptually similar to parallel-coordinates [15, 16, 18].

Animating random draws from a probabilistic graph model as applied in NetHOPs also requires addressing layout stability. However, instead of helping an analyst detect graph evolution over time, the goal of layout stability when animating a probabilistic graph is to facilitate more accurate probability estimates for network structures and make these entities of interest recognizable from frame to frame. NetHOPs' layout computation is based on the aggregation and anchoring approach, developed by Brandes et al. [8], which is an instance of the offline drawing scenario where stress minimization is used to compute a reference layout based on all graphs in the sequence.

2.3 Visualizing Probabilistic Graphs

Probabilistic networks have a fixed vertex set with immutable vertex attributes. Edges are weighted, and weights are typically the probability of edge occurrence. Edge weights can be dependent by conditional probability or independent based on network context or network models used. From an analytical perspective, probabilistic networks have constrained usability because some common graph analysis tasks, such as cluster detection, are difficult to implement when edges are weighted by probability. Although there are existing algorithms to identify clusters and communities for a static weighted network (e.g., [42, 55, 56]), the uncertainty of community occurrence or cluster membership is impossible to visualize in a static network representation.

Although visual variables [6] (e.g., width or fuzziness) can encode edge probability, this approach is unsuitable for denser probabilistic networks [31, 58]. This is because probabilistic graphs tend to have maximal connectivity, that is, all edges with non-zero weights must be present in the graph. Even if analysts can find relevant network configurations (e.g., path or alternating stars), they must decode and integrate the meaning of visual properties from multiple edges, which are often limited for supporting accurate visual judgments [39, 50, 59].

One common strategy to simplify weighted graphs is to truncate or subset the edges by an analyst-defined threshold [23, 72]. However, it is difficult to identify an optimal threshold that can maintain the structure of the networks without information loss. Perhaps because of these challenges, visualizing probabilistic networks using the node-link diagram is a relatively unexplored topic. One exception is Schulz et al. [81], who developed a layout algorithm for probabilistic networks that can blend sampled realizations from a probabilistic graph model into a static visualization, shown in Fig. 2. Their approach anchors and aligns different network realizations [9] such that the same vertex can form a vertex cloud through node splatting [36]. Edges are splatted and bundled [36] to improve readability. However, improved readability of

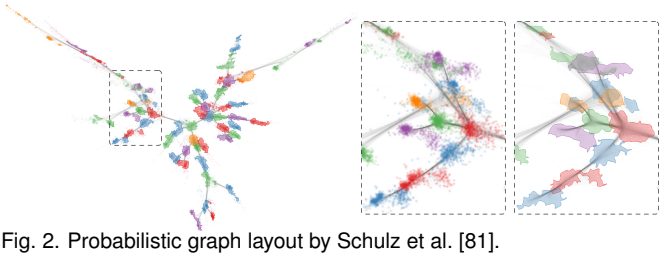

Fig. 2. Probabilistic graph layout by Schulz et al. [81].

the graph as a whole comes at the cost of concealing edge uncertainty, such that it is difficult to imagine analysts estimating the distribution of properties like edge occurrence or path lengths. Conceptually, probabilistic graph layout and NetHOPs share many similarities, but the major difference is that probabilistic graph layouts aim to create a static visual representation through 2D graph embedding, such that many of the limitations for decoding distributional information from a static depiction of a probabilistic graph remain.

2.4 Community Detection

Graph clustering is an important topic in many network domains (e.g., [4, 41, 49, 60]). Network task taxonomies describe clustering and community identification as a class of topology-based tasks that is fundamental to network analysis and visualization [2, 52].

Prior work proposes several robust community detection algorithms on static graphs based on modularity optimization (e.g., [29, 64]), Markov simulations [85], random walk [68], and cluster structures [74, 75], to name a few. Many of these techniques are unsupervised or semi-supervised, so cluster boundaries and node memberships can be uncertain. Some interactive visualization tools communicate this uncertainty by showing crisp and overlapping community structures [86, 87] to address fuzzy overlapping communities with different granularity.

When graphs are made dynamic through a temporal dimension, topology can change as vertices and edges appear and vanish. Hence, detected communities can emerge, merge, split, and disappear [67, 73]. Most dynamic community detection approaches can be classified as instant optimal, temporal trade-off, or cross-time [73]. The instant optimal approach searches communities individually using each graph in the sequence, and matches communities detected throughout the sequence (e.g., [37, 76, 88]). We develop a variant of the instant optimal approach for NetHOPs so our technique can support cluster detection and community membership for probabilistic graphs.

3 Network Hypothetical Outcome Plots (NetHOPs)

NetHOPs animate a set of hypothetical outcomes sampled from a probabilistic graph model, designed to address the challenges associated with visualizing probabilistic networks. This representation removes the requirement that users visually integrate distributional information across static encodings of probability. With NetHOPs, analysts can perceive deterministic network statistics from each sampled realization (see Fig. 1), and identify edge probabilities, and probabilities of higher-level network structures, by integrating across the animated samples.

At a high level, the NetHOPs creation pipeline starts with formulating a probabilistic random graph model based on a given network dataset. The model provides a network data generating process enabling us to sample a sequence of different network realizations via a Monte Carlo process. We apply our instant-optimal community detection and matching algorithms (see Sect. 3.2) to the network sequence so each individual realization is supplemented with additional measures that capture community structure across the set. We pass the network sequence to the visualization functions, which compute the layouts and use the additional community structure measures to color communities.

3.1 Probabilistic Network Data

NetHOPs are agnostic to how the probabilistic graph model is inferred as long as the graph model describes a network distribution and network sampling is enabled through a Monte Carlo process. In Sect. 3.4, we demonstrate the construction of a random graph model by treating edge occurrences as following a Bernoulli distribution parameterized by their corresponding probabilistic weights without dependency assumption.

Our example is simple and aimed to convey the idea of probabilistic graphs. More rigorous network modeling approaches are available but are not the focus of our work. For example, efforts have been made to statistically quantify and model uncertainty in network edges by reconstructing the network through Bayesian inference [91, 92]. The exponential-family random graph models (i.e., p^* models) can also place conditional probabilities on user-specified structural configurations [14, 57, 70, 71]. Additionally, probabilistic networks can be easily created based on network context, such as the well-known Cognitive Social Structures (CSS) [51] we use for demonstration.

Given a probabilistic graph model (e.g., a posterior distribution in a Bayesian network reconstruction pipeline [91, 92]), we can sample realizations through a Monte Carlo process, which generates a set G of N graphs where $G^N = \{G^1 = (V, E^1), \cdots, G^n = (V, E^n)\}$. We can then supplement the sequence of realizations with additional information, including a stability score and community label for each vertex.

3.2 Community Matching

NetHOPs are designed to support probabilistic versions of common visual network analysis tasks, such as estimating distributions of network statistics that are normally deterministic (e.g., shortest path, density, edge occurrence, isolates). Many of these can be supported primarily through layout engineering to preserve stability (see Sect. 4.1).

However, other properties, such as assessing the number of communities or the stability of community membership, require further optimizations. Cluster structures and membership can vary for each network realization that comprises NetHOPs due to small differences in configuration across realizations [73]. This problem is similar to the "Ship of Theseus" thought experiment [80] in which it is challenging to identify communities when their corresponding vertex memberships change partially or completely in different network realizations.

To demonstrate, the three static network visualizations in the first row of Fig. 3 are network realizations sampled from a probabilistic graph derived from Krackardt's advice-seeking CSS dataset (see Sect. 3.4 where we describe the data) positioned using the reference layout computed by aggregation. We apply a modularity-based community detection algorithm [7] to each of the network realizations, and colored vertices based on community membership without any analysis of relationships across realizations. All three realizations have three distinct communities, with similar vertex memberships. However, because community detection algorithms are typically not capable of leveraging cluster information across a set of related network realizations, communities will appear to change despite the fact that the clustering results are consistent. For example, the blue community in realization 2 has the same vertex membership as the orange community in realization 3, but we cannot identify these two communities as the same.

To address this problem, we devise a two-step community matching algorithm. Given a sequence of N graphs G^N where each $G^n = (V, E^n)$ with vertex set V and edge set $E^n = \{(i, j) : i \neq j \text{ and } i, j \in V\}$, a chosen community detection algorithm can assign a community label to each vertex as a vertex attribute Y where $Y^N = \{Y_i^1, \cdots, Y_i^n | i \in V\}$.

We then use the community-labeled network realizations as inputs to a matching process based on the degree of vertices' community co-occurrences. This process starts with constructing a weighted full graph (which we call the "co-community graph") $G_F = (V, E_F)$. The weight matrix W where $w_{i,j} = |\{Y_i^n = Y_j^n | n \in N\}|$ for G_F represents the total number of times two vertices belong to the same community summed across all networks in the sequence. In short, the "co-community graph" records the relationships between community status across realizations. To assess vertex community stability in the graph sequence, we set a threshold T where $t = \{1, \cdots, n\}$. As t increments, we remove edge (i, j) from G_F if $t > w_{ij}$, so G_F decomposes from a giant component and isolates emerge. Whenever a vertex becomes an isolate, we assign t as a "stability score" attribute to this vertex. Intuitively, the larger the stability score, the more stable the vertex. When each vertex has a stability score, we can identify the most stable vertices in each community and use these for coloring communities (Sect. 3.3.2).

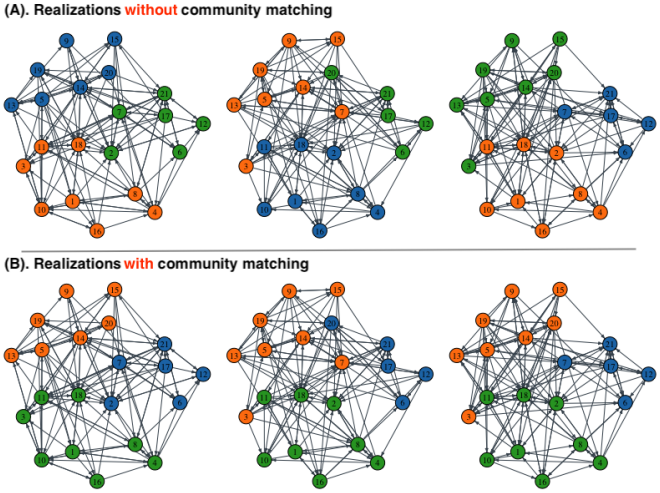

Fig. 3. (A) Probabilistic network realizations with community structures detected by Brandes et al. [7]. Vertices are colored to indicate community membership. (B) Same as (A) but with a community matching algorithm applied as described in Sect. 3.2

Our approach is designed specifically for sampled network realizations from a network distribution with a fixed vertex set. The Bernoulli sampling scheme causes network configurations to follow an approximately normal distribution by the central limit theorem, such that sampled realizations tend to have consistent structures and hence community memberships. However, this approach can be generalizable to other similar network sequences that follow a probability distribution.

3.3 Visualizing the Network

3.3.1 Layout Engineering

When arranging layouts for a graph sequence, too much emphasis on node stability across realizations diminishes readability, while optimizing individual layouts compromises analysts' ability to relate entities across states. NetHOPs address the trade-off between layout stability and readability [11] by drawing on an aggregation and anchoring technique developed by [8]. This dynamic layout algorithm is based on the stress minimization approach and leverages the offline drawing property that all realizations in a sequence are known in advance.

Given a graph $G = (V,E)$ defined by a vertex set V and an edge set E, stress minimization computes a layout P for G using Equation 1. To determine the positions p_i for $i \in V$, δ_{ij} measures the dissimilarity, or the shortest path, between (i, j) where $i, j \in V$ and $\|\cdot\|$ denotes the Euclidean norm. The weight matrix $W = \omega_{ij}$ determines the contribution of each (i, j) to the layout arrangement.

$$\text{stress}(P) = \sum_{i<j} \omega_{ij}(\delta_{ij} - \|p_i - p_j\|)^2 \tag{1}$$

Given a graph sequence G^N, we can compute a single reference layout P through aggregation by Equation 2. The reference layout has the maximum stability as it places each vertex to a fixed position for each realization in G^N (also called a "flip-book" approach [62]). In Equation 2, $\bar{\delta}_{ij}$ is the mean shortest path for (i, j) where $i, j \in V$. The weights ω_{ij} consider the variance of $\bar{\delta}_{ij}$, which places more importance on dyads having more stable shortest path length.

$$\begin{aligned} &\text{stress}(P) = \sum_{i<j} \omega_{ij}(\bar{\delta}_{ij} - \|p_i - p_j\|)^2 \\ &\text{where } \bar{\delta}_{ij} = \frac{1}{N}\sum_{n=1}^{N} \delta_{ij}^n \quad \text{and} \quad \omega_{ij} = \frac{1}{\bar{\delta}_{ij}^2} \cdot \frac{1}{1 + \text{Var}(\delta_{ij})} \end{aligned} \tag{2}$$

The reference layout, $P = p_i$ for $i \in V$, can be used as a benchmark to trade-off between layout stability and readability by Equation 3.

$$\begin{aligned} \text{stress}_\alpha(P^n) = &\underbrace{(1-\alpha) \cdot \sum_{i<j} \omega_{ij}^n (\delta_{ij}^n - \|p_i^n - p_j^n\|)^2}_{\textit{(characterizes individual layout difference)}} + \\ &\underbrace{\alpha \cdot \sum_i \|p_i^n - p_i\|^2}_{\textit{(reference stability)}} \end{aligned} \tag{3}$$

Therefore, the balance between stability and readability is parameterized by α. When $\alpha = 1$, all network realizations are positioned using P obtained by Equation 2. On the other hand, setting $\alpha = 0$ removes the stability control term, so P^n is computed by Equation 1.

3.3.2 Community Coloring

Node color is perhaps the most common way to display community structure in node-link diagrams. Thus, we use the stability score (see Sect. 3.2) and community labels to assign colors to communities. We reiterate the thresholding process used to compute the stability scores, and whenever we see a new component emerge, we assign a unique color from the Tableau 10 palette to the vertex with the highest stability score in the component if it does not yet have a color. When all stable vertices have colors assigned, we match the same colors to other vertices belonging to the same community in each network realization. The result of our matching algorithm is presented in Fig. 3 (B). Additional marks, such as convex hulls, can be added to reinforce boundaries.

3.4 Application of NetHOPs

We describe an end-to-end application of NetHOPs used to create an interactive web-based prototype visualization system for study.

Probabilistic Graph Model Krackhardt's CSS data [51] is a three-dimensional data structure used to measure individual perceptions of social relations within a network [12, 51]. Following the notation used in [51], if $\mathscr{R}$ denotes a collection of adjacency matrices measuring a relation of interest, CSS can be represented as $\mathscr{R}_{i,j,k}$, where i is the sender of a relation, j is the receiver, and k is the perceiver. For example, if $\mathscr{R}$ measures friendship, $\mathscr{R}_{1,2,3} = 1$ indicates person 3 perceives person 1 as a friend of person 2. We build our NetHOPs prototypes using two CSS datasets, which measure the advice-seeking and friendship relations among 21 managers in a high-tech firm. Therefore, our data has a dimension of $R \times N \times N \times N$ where $R = 2$ and $N = 21$.

Traditionally, perceptions $\mathscr{R}'_{i,j} = f(\mathscr{R}_{i,j,k_1}, \cdots, \mathscr{R}_{i,j,k_n})$ are used to build networks called *consensus structures* through a dimension-reduction technique by Equation 4 because the ground truth relations can be predicted from a weighted average of individual perceptions [23, 72]. However, Equation 4 is essentially the thresholding approach to create networks (see Sect. 2.3) because it adds an edge (i, j) to the network if a certain proportion of perceivers claim $\mathscr{R}'_{i,j} = 1$.

$$\mathscr{R}'_{i,j} = \begin{cases} 1 & \text{if } \frac{1}{N}\sum_k \mathscr{R}'_{i,j,k} \geq \text{ Threshold} \\ 0 & \text{Otherwise} \end{cases} \tag{4}$$

To avoid the thresholding approach, we create a weighted full graph $G = (V,E)$. A weight matrix $W = w_{i,j}$ where $i \neq j$ and $i, j \in V$ holds the aggregated perception of (i, j) given by k perceivers computed by $w_{i,j} = \frac{1}{N}\sum_k \mathscr{R}'_{i,j,k}$. In reality, individuals can have varying degrees of perception accuracy depending on their experiences, observations, or acquaintances with each other. For demonstration purposes, we treat the quality of perceptions homogeneously, thus placing an equal weight when computing edge probabilities.

Since the CSS data is a collection of perceptions provided independently by all individuals in the network, we treat each edge as an independent Bernoulli random variable. The edge variable $X_{i,j}$ can be directly modeled by its dyadic covariate $w_{i,j}$, and $Pr(X_{i,j} = 1) = E(X_{i,j}) = w_{i,j}$, which makes network sampling possible. Each sampled realization in the sequence is a possible "version of reality" given the

real-world social relations among the 21 managers. We repeatedly sample 150 network realizations for both the advice-seeking and friendship relations, and use these to create two sequences of 150 hypothetical outcomes, where the order in the sequence is not meaningful.

Community Membership For each sequence, we apply our community matching algorithm and recorded community membership.

NetHOPs Rendering We then compute a total of 11 sets of layouts using Equation 3 by setting the α parameter from zero to one with an increment of 0.1. Because of the offline drawing approach, users of the prototype can adjust layout stability and readability in the interface by tuning the amount of anchoring through a slider and receive immediate feedback. We then create another slider that enables users to adjust the animation speed in the unit of second per network, which ranges from 0.1 to 2. The interface also has four switches that allow users to control graph-specific visual elements, such as adjusting edge opacity between 0.2 and 1, or turn on or off visual aids like convex hulls, node color, and node labels designed specifically for community detection tasks.

4 An Exploratory Study on NetHOPs

We sought to investigate how well social network experts could use NetHOPs to conduct a broad range of common network analysis tasks, including density, path lengths, community detection, edge occurrence, and node attributes. We aimed to understand how well participants could process the uncertainty inherent in probabilistic graphs when estimating network properties or statistics. We used Krackhardt's CSS data (described in Sect. 3.4), which are relatively easy to describe for a study and include two graphs of varying connectivities. All experimental materials are included in the supplemental material.

4.1 Tasks

Our tasks were inspired by the task taxonomies from Lee et al. [52] and Ahn et al. [2]. We extended these tasks to focus on participants' ability to perceive the uncertainty of network statistics and properties. The available graph-specific objects in the CSS data imply tasks are centered around nodes, links, paths, and clusters (communities). Table 1 presents the complete list of tasks we used, labeled according to [52]. We omitted the isolate question on the advice-seeking CSS network because no isolates were detected after many sampling runs.

Naturally, the extent to which we design custom parameterizations of the visualization to support specific tasks will affect performance. Table 1 also lists the set of default visualization parameters and graphical elements we chose to provide for each task. We identified these parameter choices based on the target of the task. For graphical elements, this meant turning on visual features, like colors and convex hulls, helpful for community detection. Our goal was to provide minimal support so as to investigate how well analysts could do with a fairly stable visualization with extra visual features that would be easy to turn on or off (e.g., node or edge highlighting). For anchoring and animation speed, we used self-experimentation to choose parameter values that seemed most beneficial for the tasks. However, some of these choices can be arbitrary (e.g., node label, dark edges), so we designed a portion of our study to explore how analysts tuned the parameters themselves.

4.2 Study Interface

The study interface contained two columns, a visualization panel containing NetHOPs on the left, and a task panel showing questions on the right. The default view coupled each NetHOPs with an animation control panel, which allowed participants to pause, resume, forward play, or backward play realizations. When paused, participants could inspect NetHOPs' realizations one by one. A slider above the buttons indicated the progress of the animation, with a reset button next to the progress bar, which participants could click to reset to the first realization. The additional controls for NetHOPs' parameters mentioned in Sect. 3.4 were available for participants to use for the second part of the study, as we describe below.

4.3 Response Elicitation

Using a frequency-based distribution sketching tool called the distribution builder, our interface recorded probability estimates for node-attribute and edge-attribute tasks and elicited participants' uncertainty perception for topology, overview, and browsing tasks [82].

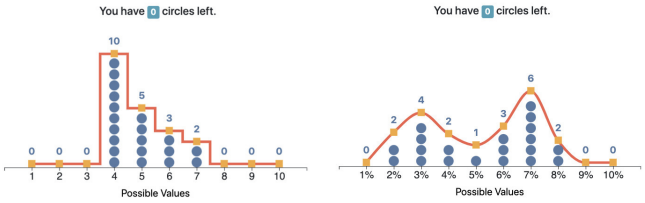

Fig. 4. Modified distribution builder adapted from [2, 38] with kernel density curve used to elicit uncertainty perception.

Distribution builders allowed participants to place a set number of balls (e.g., 20, 50, 100) in bins to express their beliefs about a parameter distribution. Prior work suggests this method leads to less noisy elicited beliefs than common approaches (e.g., asking for fractiles [30] or sketching continuous density functions [38]).

For our distribution elicitation tasks, we requested participants first make deterministic estimates of the upper and lower bound of a network statistic. The interface then used the range produced by these estimates as the x-axis scale of the distribution builder. Participants could then drag different yellow markers above all possible discrete values that fall within the range, and allocate a total of 20 balls to each discrete "bucket" to approximate their perception of a distribution.

Our study tasks required eliciting both discrete (e.g., path length) and continuous (i.e., network density) distributions. To help participants distinguish the difference, we created a modified distribution builder adapted from [38] by adding a kernel density curve as shown in Fig. 4. This instant feedback on the overall shape is important for continuous distributions because there are different ways a participant might try to use the interface. For example, if a participant wants to sketch a long-tailed distribution, she might forget to place balls in every other bin near the tail, which, unintentionally, results in a multi-modal distribution.

4.4 Study Procedure

Participants were directed to our web interface and instructed to complete the study in one session using Google Chrome. Participants were first required to pass a qualification test consisting of seven questions designed to check that they understood basic network statistics. The test questions were based on a simple static network diagram, and participants had to count or compute vertices, edges, isolates, communities, path length, and density, resembling the actual task questions shown in Table 1. To ensure participants were familiar with the use of distribution builders before tasks, we provided an exercise at the end of the qualification test. A bell-shaped normal distribution was given, and participants were asked to drag the yellow markers to mimic the distribution as much as possible. Participants passed the assessment if they made one or fewer errors, excluding the distribution builder exercise. If an error was detected, they were directed to a test results page, in which the system would show which question was wrong with detailed explanations of why and how to get the correct answer.

Participants who passed the qualification test reviewed some background information, which described the CSS data structures, illustrated the NetHOP's data generating process, and reviewed community detection. At the end of the background section, participants were given detailed instructions on how to use the NetHOP's application interface and watched a short video visually demonstrating functionalities with transcribed text shown on the same page.

Participants began the tasks ordered by taxonomy shown in Table 1. They first completed the task on the advice-seeking NetHOPs and then proceeded to complete the same task using the friendship NetHOPs. Participants completed all tasks two times. In the first iteration, they completed the tasks using the default visualization parameters and graphical elements (Table 1). In the second iteration, they completed the same set of tasks but were given the freedom to tune the visualization parameters and control graphical elements.

To ensure our participants fully understood how to use the visualization control panels to tune NetHOPs' displays, we created an instructional page describing the features of visualization parameters

Table 1. NetHOPs user study tasks listed by task taxonomy with default visualization parameter value and graphical elements add-ons.

Taxonomy	Task	Response Type	Visualization Parameter		Graphical Elements			
			Anchoring	Animation Speed	Dark Edges	Convex Hull	Node Color	Node Label
Topology	Estimate the number of distinct communities	Distribution	0.8	1	✓	✓	✓	✓
Overview	Estimate the number of isolates	Distribution	0.8	1	✓	✗	✗	✓
	Estimate the graph density	Distribution	0	1	✓	✗	✗	✗
Browsing	Estimate the highlighted shortest path lengths	Distribution	0.8	1	✓	✗	✗	✓
Vertex-attribute	Estimate the vertex community stability	Probability	1	0.5	✗	✗	✓	✓
Edge-attribute	Estimate the occurrence of highlighted edges	Probability	0.8	1	✗	✗	✗	✓

and graphical visual elements before proceeding to the second iteration. Another short video was created to provide a better visual aid explaining this newly added feature panel. In the second iteration, we reminded participants of each response they had previously provided to make it easier for them to tell if they thought the tuning could be improved. We instructed them to change their responses only if they were reasonably confident that tuning the visualization parameters and changing the graph elements could help them answer the task questions.

4.5 Participants

We recruited participants familiar with SNA and graph theory by sending a recruitment email to a large SNA listserv and encouraged recipients to forward the study to relevant personnel. We also recruited participants from a private institution in the midwest, mainly from departments related to perception and networks (e.g., psychology, computer science, industrial engineering, and sociology). Participants who successfully completed the study received a $15 dollar Amazon Gift Card or an equivalent for their time and effort completing the study.

4.6 Analysis Approach

We computed the ground truth probabilities and distributions for all tasks using the 150 network realizations that comprised NetHOPs.

For tasks asking for deterministic probability estimates (node-attribute and edge-attribute tasks), we computed the differences between participants' guesses and the ground truth probability summarized from all realizations to assess how much they were off and in what direction. For distribution elicitation tasks (overview, browsing, topology tasks), we discretized the ground truth distributions using the quantile dot plot (QDP) algorithm with pre-specified and constant bin width [48]. QDP is non-parametric and visualizes the predictive quantiles in Wilkinsonian dotplot [89], which allows us to match the ground truth distribution with the elicited distribution from the participant.

To assess the differences between participants' elicited distributions and the discretized ground truth distributions, we used Earth Mover's Distance (EMD), which computes the minimum amount of work needed to shift the distribution of interest to match the target distribution of truth [77, 78]. QDP enables us to one-to-one point match the elicited distribution with the ground truth distribution formed by 20 balls; hence, the input distributions have the integer solution property, and each ball has the same weights of one [33].

Given a user input distribution $U = \{(u_1, w_1), \cdots, (u_i, w_i)\}$ and the ground truth distribution $T = \{(t_1, w_1), \cdots, (t_j, w_j)\}$, $|U| = |T| = N$ and $w_i = w_j = 1\ \forall i, j \in N$. Let $F(U,T)$ be a set of all feasible flow to match balls in U to T, and so the distance or cost to perform such movements is the Manhattan distances, $L_M(u_i, t_j)$, summed across all pair of balls, which can be expressed by $Cost(F,u,t) = \sum_i \sum_j f_{ij} L_M(u_i, t_j)$. Therefore, the EMD score can be computed by Equation 5, which is an optimization aiming to minimize the cost of flows. EMD score can be interpreted as the average distance balls in the user distribution must move to match those of the target distribution.

$$EMD(U,T) = \frac{min_{F=(f_{ij}) \in F(u,t)} Cost(F,U,T)}{N} \tag{5}$$

Below, we report mean estimates with bootstrapped 95% confidence intervals (CIs) of participants' responses and EMDs for each task.

5 Results

5.1 Preliminaries

Of 173 participants recruited to our study website, 32% (56) passed the qualification test. The high exclusion rate indicates our stringent requirement on the expertise of the participants. We removed five entries due to incomplete data, leaving 51 responses for analysis. The majority of our participants are graduate-level network researchers (98%) who have taken at least one course in SNA or graph theory (90%) and use network analysis in their daily work (86%).

We provide an overview of performance in Fig. 5, with responses superimposed against the ground truth distribution (rows A and B) or probability (row C). Empirical cumulative distributions functions (A and B right) are plotted for all distribution elicitation tasks. Barplots and density curves differentiate the discrete (A left) and continuous (B left) responses. We present results from the first block of trials where visualization parameters were given separately from those for the second block where tuning was allowed.

In reviewing overall performance, we discovered a data anomaly for the browsing task of the friendship network in Fig. 5. About half of participants (22) incorrectly assumed some realizations had a shortest path length of zero (Figure 6 A, left, second column) when theoretically it should be infinity and therefore not counted when sketching the distributions. This conceptual misunderstanding led to more divergence (i.e., higher EMD scores) in response quality for this task.

5.2 Performance without Tuning

We evaluated individual task performance by computing the bootstrapped mean EMD scores for distribution elicitation tasks and the bootstrapped mean estimation error for the probability estimation tasks. The blue and orange 95% CIs in Fig. 6 indicate tasks completed with or without tuning correspondingly.

Browsing Task Participants performed consistently well when tracking the shortest path lengths between selected vertices on both the advice-seeking and the friendship networks, despite the smaller variance for the ground truth distribution in the advice-seeking network than that of the friendship network shown in Fig. 5 (A). Participants were more likely to overestimate the shortest path length for the advice-seeking network, and, as described above, underestimate for friendship.

Despite these small differences in apparent bias in the results, the two blue CIs in Fig. 6 (A) for the two browsing tasks almost completely overlap. The mean EMD for advice-seeking has a CI $[1.69, 2.49]$ compared to a CI $[1.67, 2.5]$ for the mean EMD of friendship.

Overview Tasks: Most participants excelled at the isolate identification task, with the response distributions closely resembling the ground truth (Fig. 5 A, middle columns). The blue CIs for the isolate task in the middle of Fig. 6 (A), top, come closest to zero EMD. Network density estimation, on the other hand, was the most difficult task for participants. Density plots and CDFs in Fig. 5 (B) show lots of dispersion in elicited distributions, with some centered roughly 40% points away from the ground truth distribution's location. The mean EMD scores for the density estimation tasks are much higher than the rest of the distribution-elicitation tasks in Fig. 6 (B). Participants, on average, perceived the density distribution more accurately on the sparse friendship network (CI$[7.3, 13.9]$) than that of the advice-seeking (CI$[13.9, 22.2]$). The difficulty of density estimation was also reflected in comments that participants left at the end of the study (e.g., "Density seems near impossible to intuitively estimate" and "I was able to count for the first few questions but that is clearly not practical for the density ones.").

Topology Task Results for the topology tasks counting distinct communities suggest participants perceived the distributions more accurately in the advice-seeking network. The elicited distributions are clearly biased toward underestimation for the sparse friendship network (Fig. 5 A, last column) but less clearly biased and closer to the ground truth for advice-seeking. The blue CIs on EMD in Fig. 6 (row A) do not overlap for these tasks (advice-seeking: $[1.47, 2.14]$, friendship:

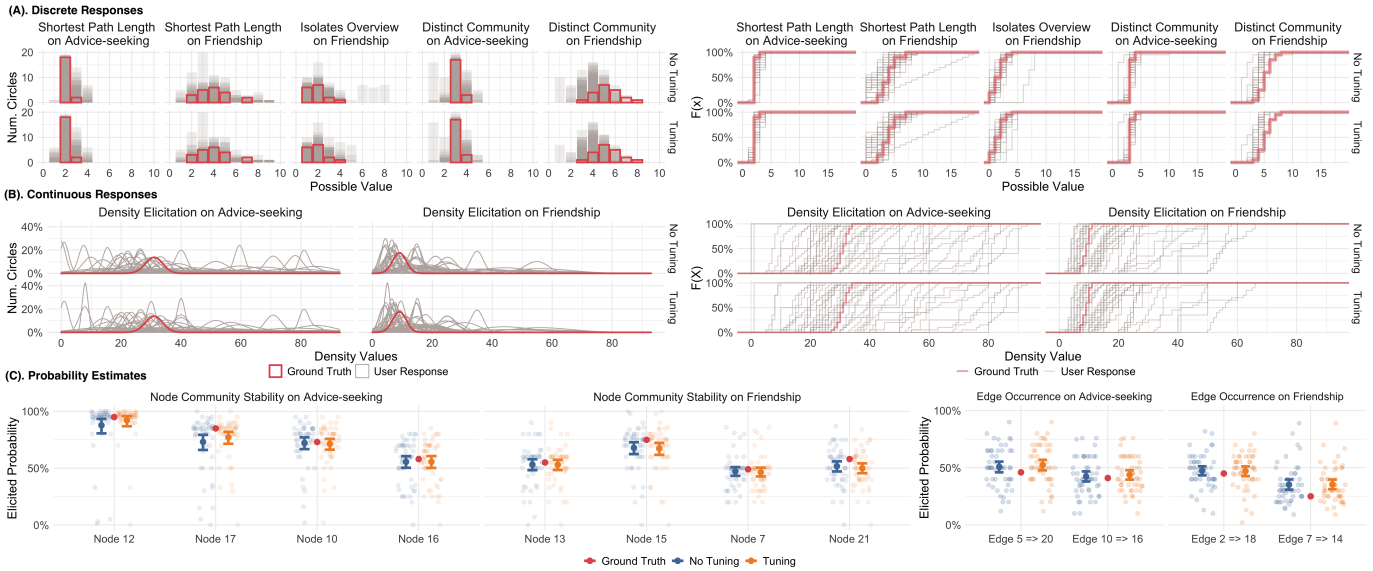

Fig. 5. (A). Left: Barplots display participants' discrete responses for browsing, topology, and overview (isolate) tasks stacked on top of each other against the ground truth distribution in red. Right: Empirical cumulative distribution functions based on participants' elicited discrete responses shown in the barplots on the left. (B). Left: the density plot shows participants' continuous responses for the network density task stacked on top of each other against the ground truth distribution in red. Right: Empirical cumulative distribution functions based on participants' elicited network density estimations are shown in the density plot on the left. (C). Strip plots present participants' deterministic probability estimates with 95% bootstrapped CIs for each attribute-based task from the study. The ground truth probabilities for each task are shown as red points between the CIs. We elicited four probability estimates for node-attribute tasks and two for edge-attribute tasks from both advice-seeking and friendship networks.

CI$[2.75, 3.36]$). This outcome is unsurprising because the ground truth distribution for the advice-seeking networks shown in Fig. 5 (A) has a smaller variance compared with that of the friendship network.

Attribute-based Tasks Fig. 6 (C) and (D) display how much participants' probability estimates were off from the ground truth for the attribute-based tasks. Participants tended to underestimate probabilities for node-attribute tasks and overestimate probabilities for edge-attribute tasks. However, on average, participants' probability estimates were not too far from the ground truth. As suggested by the blue CIs in Fig. 6 (C) and (D), all participants' deterministic probability guesses were off within 20 percentage points of the ground truth probability.

5.3 Performance with Tuning

Several participants commented that the ability to control NetHOP rendering helped them complete the tasks. We summed each participants' EMD scores for all tasks completed with and without tuning and computed the average amount of improvement. We found that the ability to control NetHOPs rendering could improve participants' distribution elicitation by 4% on average, though this was not a reliable difference given our sample size (CI $[-4.2\%, 12\%]$). We computed a similar statistic by aggregating the total absolute error for probability estimation tasks, and found tuning visualization parameters did not improve probability estimation (CI$[-14.2\%, 13.3\%]$).

Recall that we instructed participants to update their answers only if they were confident that tuning could help them achieve better results. Participants performed reasonably well with the default parameters, which makes this result not terribly surprising. Many may not have felt they could improve the prior parameters, or tweaked them only by a small amount. This is reflected in that 22% (11) of our participants provided the same answers in the second iteration of tasks. Nonetheless, we provide an exploratory analysis of tuning strategies, since how participants changed parameters can shed light on how visualization parameters may support different tasks in slightly different ways.

5.3.1 Tuning Strategies

The distributions of the two performance metrics indicate that tuning did help some participants. To investigate the dynamics between visualization parameters and accuracy, we ranked all participants by their performance with tuning and grouped top-performers from the first quartile and bottom-performers from the fourth quartile, with each

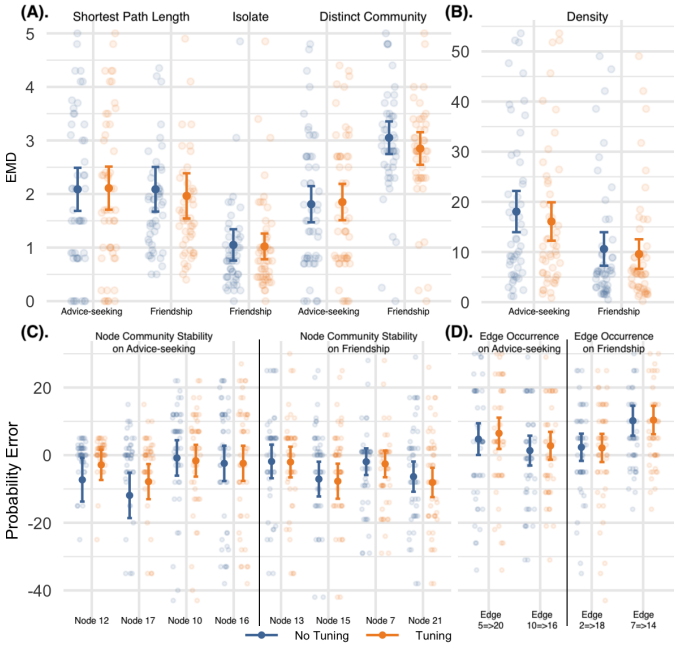

Fig. 6. (A) and (B) display the bootstrapped mean EMD scores with 95% CI for browsing (shortest path length), overview (isolate and network density), and topology (distinct community) tasks. (C) and (D) display the bootstrapped mean of probability estimation errors for each attribute-based task on node community stability and edge occurrence.

group consisting of 13 participants. Top-performers had a pooled mean EMD score of 3.1 (Min.: 1.7, Median: 2.8, SD: 1.2, Max.: 5.3), and their probability estimates were, on average, 1.9% off from the truth (Min.: 0.1, Median: 1.3, SD: 1.6, Max.: 5). On the other hand, bottom-performers had a pooled mean EMD score of 7.8 (Min.: 3.1, Median: 6.3, SD: 4.7, Max.: 15.4), and their probability estimates were, on average, 14.3% off from the truth (Min.: 1.3, Median: 12.6, SD: 12.1, Max.: 44.9). We focused on how anchoring and animation speed were used by each group, and present the bootstrapped mean with 95% CIs in Fig. 7 to assess how the parameters related to response quality.

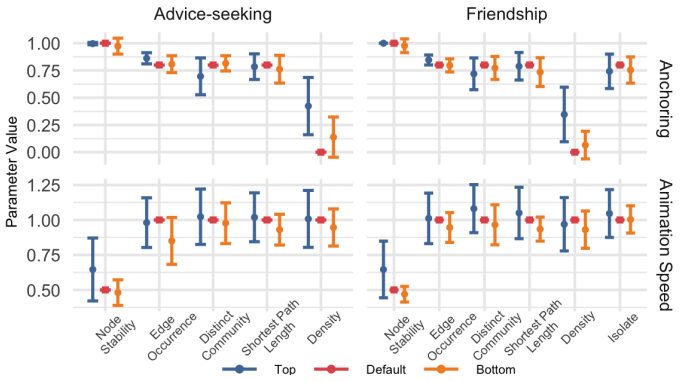

Fig. 7. Bootstrapped means with 95% CIs for anchoring (A) and frame rate (B) parameters used by participants.

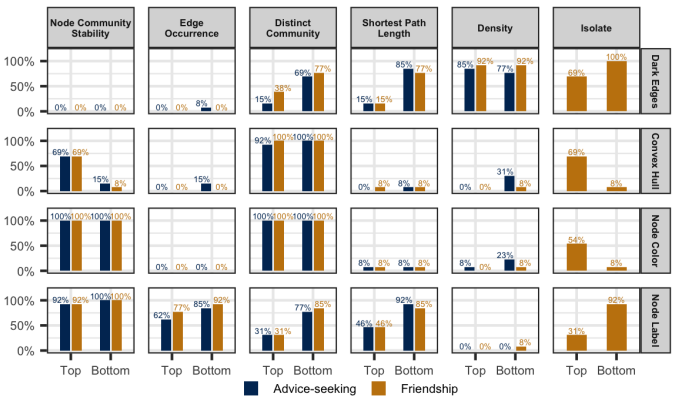

Fig. 8. Participants' use of graphical visual aids.

Tuning on Layout Stability Fig. 7 row (A) shows participants' anchoring tuning did not deviate too much from the default value we provided for most tasks, except for density estimation. For this overview task, participants preferred layout stability over the general readability the layout algorithm optimizes for (e.g., by reducing overlapping edges). This goes against our expectation that the need to estimate density would trump the need to keep nodes in consistent positions, which had led us to set a default anchoring value of zero.

Layout stability appears more important for the two attribute-based tasks. Participants preferred higher average anchoring for node-attribute tasks regardless of performance, and nearly all top-performers chose the most stable layout for node-attribute tasks.

The topology tasks (detecting distinct communities) and the isolate overview task were the only two tasks where top-performers on average preferred lower anchoring values than the bottom-performers. Anchoring preferences were similar between networks for the browsing tasks. Both groups lowered the anchoring slightly from the default of 0.8, but the bottom-performers, on average, lowered anchoring a bit more.

We pooled all anchoring parameters used by top-performers and bottom-performers for each task and found that top-performers preferred layout stability more than bottom-performers. Top-performers tended to set a slightly higher average anchoring of 0.81 (CI $[0.73, 0.89]$) compared with the average 0.71 (CI $[0.63, 0.79]$) set by the bottom-performers. Between CSS datasets, participants preferred more layout stability when working on the denser advice-seeking network. Top-performers used an anchoring of 0.83 (CI $[0.75, 0.90]$) on average for the advice-seeking and 0.78 (CI $[0.70, 0.86]$) for the friendship. This penchant is more noticeable from bottom-performers as the averaged anchoring used on the advice-seeking network is 0.72 (CI$[0.63, 0.81]$), and the friendship network is 0.67 (CI$[0.59, 0.75]$).

Tuning on Animation Speed One consistent pattern observable from Fig. 6 (B) is that top-performers generally chose to set the animation speed slower than the bottom-performers. Top-performers, on average, preferred to render each network for 1 second (CI $[0.9, 1.13]$), compared to the average 0.82 (CI $[0.75, 0.9]$) of bottom-performers.

When viewing the denser advice-seeking networks, all participants from both groups preferred faster animation speed on average, displaying each network realization on the screen for 0.77 seconds on average (CI $[0.66, 0.88]$), compared to 0.84 seconds (CI $[0.75, 0.92]$) for the friendship network. For top-performers, the averaged animation speed when completing tasks on the advice-seeking network was 0.97 seconds per network (CI$[0.85, 1.1]$), and 1.02 seconds per network (CI$[0.91, 1.13]$) for the friendship network. Similarly, bottom-performers chose to display each network realization on the screen for 0.77 seconds (CI$[0.68, 0.86]$) for the advice-seeking network and 0.83 (CI$[0.76, 0.90]$) seconds for the friendship network.

Graphical Elements Edge opacity, node color, convex hulls, and node labels are network-related graphical attributes or elements that can be highly task-specific. In analysis, we found it hard to generalize much from looking at each individually, but analyzing combinations of visual parameters was more meaningful.

We computed the percentage of participants who used each graphical element by task in Fig. 8 and found some combinations were very different from the default visualization that we thought would work best on a given task. The majority of top-performers chose to make targeted or relevant network objects and elements more salient by deactivating irrelevant visual elements. For example, when detecting the number of communities, they chose to make edges less salient and turn off node labels to better emphasize node color and convex hulls. When browsing shortest paths, 75% of top-performers preferred less salient edges to further emphasize the red highlighted edges connecting the nodes. To estimate network densities, top-performers preferred more salient edges while minimizing other visual elements by turning off convex hulls, node color, and node labels.

We also observed some creative use of visual features. For example, when identifying the number of isolates, roughly 70% of top-performers turned on convex hulls, 50% removed node color, and 30% preferred less salient edges, even though convex hulls and node colors were seemingly irrelevant visual elements to the task. We note that convex hulls do not include isolated nodes, and our coloring algorithm would consistently assign a distinct color to isolated nodes. With these two parameters, edge salience becomes less important and could be turned off to accentuate convex hulls and node color.

5.4 Precision of Inference & Time-Accuracy Correlation

Two forms of error can impact the precision of inferences about network statistics made with NetHOPs: perceptual and cognitive errors related to how accurately analysts can estimate probabilities from the visualization (which applies to any visualizations of distributions), and approximation error introduced by sampling from the network model.

First, we quantify the sampling error introduced when taking a set of random draws from the graph model. While we cannot compute approximation error against the ground truth model, we can infer it by re-sampling N sets of network realizations from the model and using the distributions of network statistics from each re-sampled set to compute EMD scores against those from the set of NetHOPs. We can then quantify the sampling error of the distribution for each network statistic in the unit of EMD by constructing a confidence interval, as shown in Fig. 9. Therefore, if a participant perfectly perceived and sketched a distribution and received an EMD score of zero using NetHOPs, her perception could be off by an $\mu_{EMD} \pm SE_{EMD}$ amount. This approach is generalizable and can be easily applied to compute the sampling error of any probability estimation task.

Naturally, participants may not view all NetHOPs' realizations. While we did not log exact realizations viewed, we can use time spent on tasks to infer approximate viewing. For our study, we found participants, on average, spent 55 minutes to complete all tasks (Min.: 9, Median: 49, SD: 32, Max.: 176), after removing one outlier of 549 minutes. This participant paused on one task page for approximately 8.5 hours, suggesting this participant might have left the browser open and walked away. Recall that the average of 55 minutes describes participants' time to complete a total of 70 questions spread across 22 task screens. For the distribution elicitation tasks, 75% of participants completed the tasks within approximately five minutes, which includes

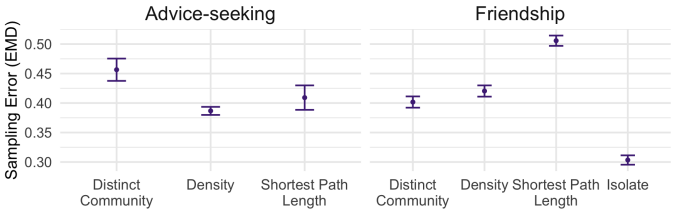

Fig. 9. Sampling error for the distribution elicitation tasks in EMD based on 500 samples of 150 network realizations.

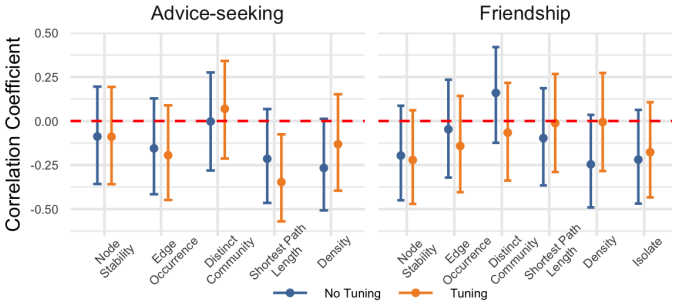

Fig. 10. Pearson's Correlation with 95% CIs between task completion time (mins.) and perception accuracy.

the time spent on identifying the lower and upper bounds and sketching the distributions. For probability elicitation tasks, 75% of our participants were able to complete all four sub-tasks for node stability within six minutes and two sub-tasks for edge occurrence within three minutes. These estimates suggest that many participants viewed all 150 frames, though we cannot be sure how they divided their attention between questions and watching the animation.

To assess whether spending more time on a task correlated with performance, we conducted a correlation analysis between task completion time and response quality by computing Pearson's correlation coefficient for each task, then by using Fisher's Z-Transformation [21, 22] to meet the normality assumption and compute CIs shown in Fig. 10. We observed negative correlation coefficients in the range of $[-0.3, 0]$ for the majority of the tasks, implying that longer time spent on tasks can slightly reduce the amount of perception error. However, almost all CIs include zero, which makes this effect unreliable except for the task on the shortest path length of the advice-seeking network with tuning. It is worth mentioning that participants left comments such as, "I had to think a lot about each question and pause to take breaks. There were a lot of questions," which may bias these correlations.

6 Discussion

Our user study demonstrates NetHOPs' potential to support exploratory analysis on probabilistic networks. We summarize our findings, provide design recommendations, and discuss future work.

Overall Performance When using NetHOPs, we discovered analysts were better at identifying isolates, tracking path lengths, and detecting attribute-based changes on both nodes and edges, while they struggled more with estimating properties like density and distinct communities when realizations had overlapping communities. We speculate the reason our participants excelled at identifying isolates and tracing paths is that people tend to be visually sensitive to a lack of continuity between nodes, perhaps because it can be recognized pre-attentively when layouts are stable.

We found that participants' estimates of network properties fell on average within 11% of the statistics computed on the realizations comprising NetHOPs. Naturally, a question arises of whether this amount of error, in combination with the sampling error discussed in Sect. 5.4, is tolerable. We suspect that the error induced by NetHOPs is acceptable in many cases where visual analysis aims to help analysts to get a rough sense of many network properties to build intuition. We also note that whenever visualization is used for network exploration, a rough sense of the properties is more likely to be the goal, since an analyst might otherwise use more exact methods or modeling. However, future work might better contextualize whether the approximations allowed by NetHOPs seem sufficient to network analysis.

Layout Stability Our participants appeared to tune NetHOPs differently based on network density. Stable layouts seemed generally beneficial for all tasks, even for tasks like density estimation that do not require stability. A highly stable layout may prevent change-blindness and make it easier to detect important differences across realizations. Interestingly, our study participants did not prefer a completely stable layout except for attribute-based tasks on nodes. This suggests that estimating distribution from animated network realizations requires its own balance of stability versus within-realization optimization as analysts appear sensitive to both.

We also found that analysts can tolerate more node movements for sparse networks. We speculate the reason behind this may be related to the overall shape of the networks and the amount of visual clutter in the visualizations caused by density. A sparse network is more visually trackable because of fewer graphical elements, so stability can be compromised in favor of readability. For denser networks, we think methods, such as [65], can be incorporated into the layout engineering when certain network properties (e.g., small-worldness) are known to make layout more comprehensible before aggregation and anchoring.

Animation Speed We found top-performers tended to use slower animation speeds, which is unsurprising as a slower speed allows a more thorough review of each network realization. However, a faster animation speed (and a high degree of anchoring) appears to work well for node-attribute tasks where change detection is straightforward. The chosen animation speeds for the two NetHOPs in our user study are relatively slower compared to the chosen default speeds for simpler 2D HOPs in prior work [39, 46, 47]. This is because network realizations convey more information than simpler charts and require more cognitive load for analysts to process. Participants may have needed a longer time to register what relationships were present in a realization.

Graphical Elements The salience of relevant network objects appears to be the most important driver of participants' choices. Therefore, graphical aids like edge opacity, node color, convex hulls, and node labels are useful in producing more accurate uncertainty depictions. Highlighting or emphasizing relevant network objects can also help analysts identify targeted graphical elements and capture changes with minimum effort. We think these add-ons can greatly reduce the risks of change-blindness, and suggest any practical implementation of NetHOPs would allow for such visual tuning.

Limitations and Future Work NetHOPs are subject to errors from (1) approximating a distribution with samples to create the animation, and (2) analysts not watching full realizations. As we described in Sect. 5.4, errors can be quantified by comparing network statistic distributions across sets of realizations of a particular size. However, future work might aim to better understand how many network realizations analysts tend to watch before they feel confident estimating a particular statistic, providing more insight into how much time is required and for how much accuracy gain, in relation to static graphs. Our study also involved two relatively small networks. Beyond testing larger networks, researchers could explore the utility of NetHOPs for bipartite or multiplex networks, as well as network models with edge dependencies, which NetHOPs are well suited to address.

7 Conclusion

NetHOPs aim to facilitate uncertainty communication for probabilistic networks. We summarized the design of NetHOPs and illustrated the computation of its controllable dynamic layout. We designed a community matching algorithm so our technique can support uncertainty detection of topology-based tasks such as clustering identification. The results of our user study suggested that the technique can support visual exploratory analysis, at least of small networks. Our results point to directions for future work around optimizing visualization parameters and better understanding perception of network properties via animation.

Acknowledgments

This work was supported by NSF IIS-1907941, NSF IIS-1815760, and Microsoft.

REFERENCES

[1] L. A. Adamic and E. Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

[2] J.-w. Ahn, C. Plaisant, and B. Shneiderman. A task taxonomy for network evolution analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):365–376, 2013.

[3] D. Archambault, H. Purchase, and B. Pinaud. Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):539–552, 2010.

[4] P. Bedi and C. Sharma. Community detection in social networks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 6(3):115–135, 2016.

[5] D. C. Bell, B. Belli-McQueen, and A. Haider. Partner naming and forgetting: recall of network members. *Social networks*, 29(2):279–299, 2007.

[6] J. Bertin. Semiology of graphics; diagrams networks maps. Technical report, 1983.

[7] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188, 2007.

[8] U. Brandes, N. Indlekofer, and M. Mader. Visualization methods for longitudinal social networks and stochastic actor-oriented modeling. *Social Networks*, 34(3):291–308, 2012.

[9] U. Brandes and M. Mader. A quantitative comparison of stress-minimization approaches for offline dynamic graph drawing. In *International Symposium on Graph Drawing*, pp. 99–110. Springer, 2011.

[10] U. Brandes and C. Pich. An experimental study on distance-based graph drawing. In *International Symposium on Graph Drawing*, pp. 218–229. Springer, 2008.

[11] U. Brandes and D. Wagner. A bayesian paradigm for dynamic graph layout. In *International Symposium on Graph Drawing*, pp. 236–247. Springer, 1997.

[12] R. A. Brands. Cognitive social structures in social network research: A review. *Journal of Organizational Behavior*, 34(S1):S82–S103, 2013.

[13] J. Branke. Dynamic graph drawing. In *Drawing Graphs*, pp. 228–246. Springer, 2001.

[14] A. Caimo and N. Friel. Bayesian inference for exponential random graph models. *Social Networks*, 33(1):41–55, 2011.

[15] T. Dwyer and D. R. Gallagher. Visualising changes in fund manager holdings in two and a half-dimensions. *Information Visualization*, 3(4):227–244, 2004.

[16] T. Dwyer, S.-H. Hong, D. Koschützki, F. Schreiber, and K. Xu. Visual analysis of network centralities. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation-Volume 60*, pp. 189–197, 2006.

[17] P. Eades. A heuristic for graph drawing. *Congressus numerantium*, 42:149–160, 1984.

[18] C. Erten, S. G. Kobourov, V. Le, and A. Navabi. Simultaneous graph drawing: Layout algorithms and visualization schemes. In *International Symposium on Graph Drawing*, pp. 437–449. Springer, 2003.

[19] B. J. Evans. Dynamic display of spatial data-reliability: Does it benefit the map user? *Computers & Geosciences*, 23(4):409–422, 1997.

[20] M. Fernandes, L. Walls, S. Munson, J. Hullman, and M. Kay. Uncertainty displays using quantile dotplots or cdfs improve transit decision-making. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2018.

[21] R. A. Fisher. Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. *Biometrika*, 10(4):507–521, 1915.

[22] R. A. Fisher. On the'probable error'of a coefficient of correlation deduced from a small sample. *Metron*, 1:1–32, 1921.

[23] L. C. Freeman and A. K. Romney. Words, deeds and social structure: A preliminary study of the reliability of informants. *Human Organization*, 46(4):330–334, 1987.

[24] Y. Frishman and A. Tal. Online dynamic graph drawing. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):727–740, 2008.

[25] E. R. Gansner, Y. Koren, and S. North. Graph drawing by stress majorization. In *International Symposium on Graph Drawing*, pp. 239–250. Springer, 2004.

[26] R. Garcia-Retamero, M. Galesic, and G. Gigerenzer. Do icon arrays help reduce denominator neglect? *Medical Decision Making*, 30(6):672–684, 2010.

[27] W. W. Gaver. Technology affordances. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 79–84, 1991.

[28] G. Gigerenzer and U. Hoffrage. How to improve bayesian reasoning without instruction: frequency formats. *Psychological Review*, 102(4):684, 1995.

[29] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[30] D. G. Goldstein and D. Rothschild. Lay understanding of probability distributions. *Judgment & Decision Making*, 9(1), 2014.

[31] H. Guo, J. Huang, and D. H. Laidlaw. Representing uncertainty in graph edges: An evaluation of paired visual variables. *IEEE Transactions on Visualization and Computer Graphics*, 21(10):1173–1186, 2015.

[32] L. Hasher and R. T. Zacks. Automatic processing of fundamental information: the case of frequency of occurrence. *American Psychologist*, 39(12):1372, 1984.

[33] F. S. Hillier and G. J. Lieberman. *Introduction to mathematical programming*. McGraw-Hill, 1995.

[34] U. Hoffrage and G. Gigerenzer. Using natural frequencies to improve diagnostic inferences. *Academic Medicine*, 73(5):538–540, 1998.

[35] J. M. Hofman, D. G. Goldstein, and J. Hullman. How visualizing inferential uncertainty can mislead readers about treatment effects in scientific results. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2020.

[36] D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006.

[37] J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Tracking evolving communities in large linked networks. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5249–5253, 2004.

[38] J. Hullman, M. Kay, Y.-S. Kim, and S. Shrestha. Imagining replications: Graphical prediction & discrete visualizations improve recall & estimation of effect uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):446–456, 2017.

[39] J. Hullman, P. Resnick, and E. Adar. Hypothetical outcome plots outperform error bars and violin plots for inferences about reliability of variable ordering. *PloS one*, 10(11):e0142444, 2015.

[40] D. R. Hunter, S. M. Goodreau, and M. S. Handcock. Goodness of fit of social network models. *Journal of the American Statistical Association*, 103(481):248–258, 2008.

[41] G. Jia, Z. Cai, M. Musolesi, Y. Wang, D. A. Tennant, R. J. Weber, J. K. Heath, and S. He. Community detection in social and biological networks using differential evolution. In *International Conference on Learning and Intelligent Optimization*, pp. 71–85. Springer, 2012.

[42] J. Jin, L. Pan, C. Wang, and J. Xie. A center-based community detection method in weighted networks. In *2011 IEEE 23rd International Conference on Tools with Artificial Intelligence*, pp. 513–518. IEEE, 2011.

[43] S. L. Joslyn and J. E. LeClerc. Uncertainty forecasts improve weather-related decisions and attenuate the effects of forecast error. *Journal of Experimental Psychology: Applied*, 18(1):126, 2012.

[44] M. F. Jung, D. Sirkin, T. M. Gür, and M. Steinert. Displayed uncertainty improves driving experience and behavior: The case of range anxiety in an electric car. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 2201–2210, 2015.

[45] C. N. Kaiser-Bunbury, J. Mougal, A. E. Whittington, T. Valentin, R. Gabriel, J. M. Olesen, and N. Blüthgen. Ecosystem restoration strengthens pollination network resilience and function. *Nature*, 542(7640):223–227, 2017.

[46] A. Kale, M. Kay, and J. Hullman. Visual reasoning strategies for effect size judgments and decisions. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):272–282, 2020.

[47] A. Kale, F. Nguyen, M. Kay, and J. Hullman. Hypothetical outcome plots help untrained observers judge trends in ambiguous data. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):892–902, 2018.

[48] M. Kay, T. Kola, J. R. Hullman, and S. A. Munson. When (ish) is my bus? user-centered visualizations of uncertainty in everyday, mobile predictive systems. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 5092–5103, 2016.

[49] K. Kim, C.-H. Jun, and J. Lee. Improved churn prediction in telecommunication industry by analyzing a large network. *Expert Systems with Applications*, 41(15):6575–6584, 2014.

[50] R. Kosara, S. Miksch, H. Hauser, J. Schrammel, V. Giller, and M. Tscheligi.

Useful properties of semantic depth of field for better f+ c visualization. In *ACM International Conference Proceeding Series*, vol. 22, pp. 205–210, 2002.

[51] D. Krackhardt. Cognitive social structures. *Social Networks*, 9(2):109–134, 1987.

[52] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *Proceedings of the 2006 AVI Workshop on Beyond Time and Errors: Novel Evaluation Methods for Information Visualization*, pp. 1–5, 2006.

[53] P. Leonardi and N. Contractor. Better people analytics. *Harvard Business Review*, 96(6):70–81, 2018.

[54] P. M. Leonardi, M. Huysman, and C. Steinfield. Enterprise social media: Definition, history, and prospects for the study of social technologies in organizations. *Journal of Computer-Mediated Communication*, 19(1):1–19, 2013.

[55] Z. Lu, X. Sun, Y. Wen, G. Cao, and T. La Porta. Algorithms and applications for community detection in weighted networks. *IEEE Transactions on Parallel and Distributed Systems*, 26(11):2916–2926, 2014.

[56] Z. Lu, Y. Wen, and G. Cao. Community detection in weighted networks: Algorithms and applications. In *2013 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 179–184. IEEE, 2013.

[57] D. Lusher, J. Koskinen, and G. Robins. *Exponential random graph models for social networks: Theory, methods, and applications*, vol. 35. Cambridge University Press, 2013.

[58] A. M. MacEachren, R. E. Roth, J. O'Brien, B. Li, D. Swingley, and M. Gahegan. Visual semiotics & uncertainty visualization: An empirical study. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2496–2505, 2012.

[59] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions On Graphics (Tog)*, 5(2):110–141, 1986.

[60] L.-E. Martinet, M. Kramer, W. Viles, L. Perkins, E. Spencer, C. Chu, S. Cash, and E. Kolaczyk. Robust dynamic community detection with applications to human brain functional networks. *Nature Communications*, 11(1):1–13, 2020.

[61] K. Misue, P. Eades, W. Lai, and K. Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages & Computing*, 6(2):183–210, 1995.

[62] J. Moody, D. McFarland, and S. Bender-deMoll. Dynamic network visualization. *American Journal of Sociology*, 110(4):1206–1241, 2005.

[63] L. Nadav-Greenberg and S. L. Joslyn. Uncertainty forecasts improve decision making among nonexperts. *Journal of Cognitive Engineering and Decision Making*, 3(3):209–227, 2009.

[64] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, 2004.

[65] A. Nocaj, M. Ortmann, and U. Brandes. Untangling hairballs. In *International Symposium on Graph Drawing*, pp. 101–112. Springer, 2014.

[66] S. C. North. Incremental layout in dynadag. In *International Symposium on Graph Drawing*, pp. 409–418. Springer, 1995.

[67] G. Palla, A.-L. Barabási, and T. Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664–667, 2007.

[68] P. Pons and M. Latapy. Computing communities in large networks using random walks. In *International Symposium on Computer and Information Sciences*, pp. 284–293. Springer, 2005.

[69] H. C. Purchase and A. Samra. Extremes are better: Investigating mental map preservation in dynamic graphs. In *International Conference on Theory and Application of Diagrams*, pp. 60–73. Springer, 2008.

[70] G. Robins, P. Pattison, Y. Kalish, and D. Lusher. An introduction to exponential random graph (p*) models for social networks. *Social Networks*, 29(2):173–191, 2007.

[71] G. Robins, T. Snijders, P. Wang, M. Handcock, and P. Pattison. Recent developments in exponential random graph (p*) models for social networks. *Social Networks*, 29(2):192–215, 2007.

[72] A. K. Romney, S. C. Weller, and W. H. Batchelder. Culture as consensus: A theory of culture and informant accuracy. *American Anthropologist*, 88(2):313–338, 1986.

[73] G. Rossetti and R. Cazabet. Community discovery in dynamic networks: a survey. *ACM Computing Surveys (CSUR)*, 51(2):1–37, 2018.

[74] M. Rosvall and C. T. Bergstrom. An information-theoretic framework for resolving community structure in complex networks. *Proceedings of the National Academy of Sciences*, 104(18):7327–7331, 2007.

[75] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[76] M. Rosvall and C. T. Bergstrom. Mapping change in large networks. *PloS one*, 5(1):e8694, 2010.

[77] Y. Rubner, L. J. Guibas, and C. Tomasi. The earth mover's distance, multi-dimensional scaling, and color-based image retrieval. In *Proceedings of the ARPA Image Understanding Workshop*, vol. 661, p. 668, 1997.

[78] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pp. 59–66. IEEE, 1998.

[79] P. Saffrey and H. Purchase. The "mental map" versus "static aesthetic" compromise in dynamic graphs: a user study. In *Proceedings of the ninth conference on Australasian user interface-Volume 76*, pp. 85–93, 2008.

[80] T. Scaltsas. The ship of theseus. *Analysis*, 40(3):152–157, 1980.

[81] C. Schulz, A. Nocaj, J. Goertler, O. Deussen, U. Brandes, and D. Weiskopf. Probabilistic graph layout for uncertain network visualization. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):531–540, 2016.

[82] W. F. Sharpe, D. G. Goldstein, and P. W. Blythe. The distribution builder: A tool for inferring investor preferences. *preprint*, 2000.

[83] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, 2006.

[84] B. N. Taylor and C. E. Kuyatt. Guidelines for evaluating and expressing the uncertainty of nist measurement results. 1994.

[85] S. M. Van Dongen. *Graph clustering by flow simulation*. PhD thesis, 2000.

[86] C. Vehlow, T. Reinhardt, and D. Weiskopf. Visualizing fuzzy overlapping communities in networks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2486–2495, 2013.

[87] Y. Wang, Q. Shen, D. Archambault, Z. Zhou, M. Zhu, S. Yang, and H. Qu. Ambiguityvis: Visualization of ambiguity in graph layouts. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):359–368, 2015.

[88] Y. Wang, B. Wu, and X. Pei. Commtracker: A core-based algorithm of tracking community evolution. In *International Conference on Advanced Data Mining and Applications*, pp. 229–240. Springer, 2008.

[89] L. Wilkinson. Dot plots. *The American Statistician*, 53(3):276–281, 1999.

[90] E. R. Wright and B. A. Pescosolido. "sorry, i forgot": the role of recall error in longitudinal personal network studies. In *Social Networks and Health*. Emerald Group Publishing Limited, 2002.

[91] J.-G. Young, G. T. Cantwell, and M. Newman. Bayesian inference of network structure from unreliable data. *Journal of Complex Networks*, 8(6):cnaa046, 2020.

[92] J.-G. Young, F. S. Valdovinos, and M. Newman. Reconstruction of plant–pollinator networks from observational data. *bioRxiv*, p. 754077, 2019.

[93] J. X. Zheng, S. Pawar, and D. F. Goodman. Graph drawing by stochastic gradient descent. *IEEE Transactions on Visualization and Computer Graphics*, 25(9):2738–2748, 2018.