Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 27, no. 5, pp. 341–404 (2023)
DOI: 10.7155/jgaa.00626

Connected-closeness: A Visual Quantification of Distances in Network Layouts

Mathieu Jacomy[1]

[1]Tantlab,
Aalborg University, Copenhagen

Submitted:	January 2021	Reviewed:	August 2021	Revised:	January 2022
Reviewed:	June 2022	Revised:	October 2022	Accepted:	June 2023
Final:	June 2023	Published:	July 2023		
Article type:	Regular paper	Communicated by:	S. G. Kobourov		

Abstract. This paper proposes a network-visualization metric, connected-closeness, designed to provide a quantified statement about the mediation of the topology by the node placement. It allows stating the percentage of connected nodes that are closer than a certain characteristic distance, computed on the basis of the layout, and pictured in the visualization. This statement, and others it provides, are intended to help non-experts interpreting network visualizations visually. Connected-closeness allows assessing a layout's validity from the specific angle of bringing connected nodes closer. A benchmark finds that force-directed layouts are indeed good at bringing connected nodes closer, but the metric also detects situations and layouts where it fails. It allows comparing different layouts for a given network and different networks for a given layout, and provides quantified evidence that force-driven placements consistently capture an aspect of the topological structure of networks. The calculations allow assessing visual distances as a statistical measure of edge presence in terms or precision and recall, and show that in practice, layout algorithms prioritize recall over precision. The paper provides the definition of different indicators, their underlying rationale, visual examples, a simple optimization, implementation remarks, and a benchmark of 14 network generators and 7 node-placement algorithms rendered 100 times each, for a total of 9800 network visualizations.

1 Introduction

Network maps are a popular graph visualization technique [5, 12] and force-driven layouts are a common node placement technique [10, 39] notably for large graphs [13]. The images produced

E-mail address: mathieu.jacomy@gmail.com (Mathieu Jacomy)

by force-driven layouts or similar approaches (e.g., stress-majorization techniques [9, 16]) are generally assessed on the basis of aesthetic criteria (e.g., minimizing edge crossings) [2, 31, 32] and experimental settings where specific tasks are performed (e.g., finding shortest paths) [30, 34] but rarely on the basis of explicit statements such as "the closer two nodes, the higher their chances to be connected". Network maps are notoriously difficult to interpret [20, 22, 38] and explicit statements about how the image mediates the network's topology would help contextualize them and prevent misinterpretations.

This paper proposes the "maximum connected-closeness", a metric providing a contextual statement about a given network map, i.e. a graph drawing where nodes are represented as dots and placed by a force-directed or similar algorithm, and edges are represented as lines. It aims at quantifying the intuition that "most connected nodes are very close". It defines a distance (Δ_{max}) that captures the best this notion (figure 1).

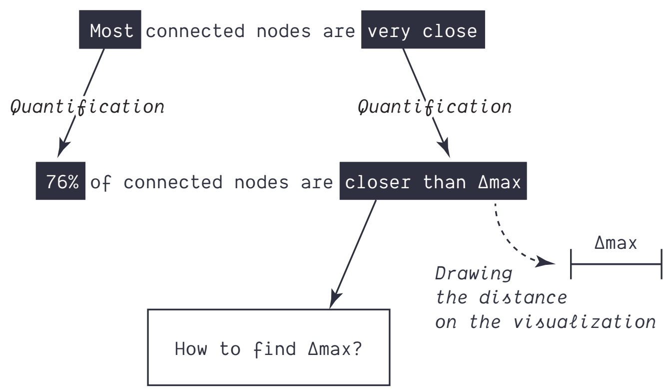


Figure 1: The goal is to quantify an intuitive statement about network maps. The distance Δ_{max} is computed by maximizing connected-closeness, a metric presented below. The "76%" number corresponds to the case of figure 5.

The connected-closeness C is the metric used to find a special distance Δ_{max} that relates to how close the connected nodes are in the layout. In other words, it characterizes overall edge shortness. More precisely, C accounts for how many edges are shorter than a given distance Δ, discounting the number of edges that would be shorter anyways (i.e., if all edges were redistributed at random). When the function $C(\Delta)$ reaches its maximum, we define Δ_{max} and $C_{max} = C(\Delta_{max})$ the maximum connected-closeness. By definition Δ_{max} is the distance for which the layout captures the most unexpectedly close connected pairs; here "close" means closer than Δ_{max}, and "unexpectedly" means compared to a random rewiring of the same network (cf. the dedicated section for a formal definition).

In this paper I contend that C_{max} is relevant to characterizing layouts as well as networks. Beyond helping scholars to interpret network maps, it can be used as a quality metric to compare different layouts. After the related works section I will expose my motivation, formalize the problem of finding the maximum connected-closeness, and provide formal definitions. Then I will propose a graphic design for network maps and the rationale for my choices. I will present a practical implementation of the metric, and highlight the main takeaways of a benchmark I conducted on various networks and layouts, also presented in detail as an appendix. Finally, I will discuss the flaws of the metric and argue that despite those, it takes us one step closer to formalizing a stable ground for visual network interpretation.

2 Related Work

The work presented here is novel in that it is designed to be embedded in a network map: connected-closeness is at the same time a visualization technique and a statistical metric, and either part needs the other to help the person interpreting their network visually. As a visualization technique, it relates to the evaluation of graph drawing, and as a statistical metric it relates to neighborhood preservation. These two issues being generally considered separately in the literature, I will look at them one after the other.

The evaluation of graph drawing, as a field, features two distinct approaches that one can refer to as "diagrammatic" and "topological" [14, 38]. The *diagrammatic* mode of interpretation is the most often cited, and focuses on the legibility of relatively small networks, evaluating tasks like path retrieval [31, 32, 40]. It offers quality metrics often called "aesthetic criteria" such as minimizing edge crossings or getting homogeneous edge lengths. The work presented here departs from this approach and relates instead to the *topological* mode of interpretation, where the node placement *mediates* the topology of the network [28, 34, 35]. In this context, graph drawing evaluation measures the cognitive ability to infer information about the graph structure by looking at the layout, for instance retrieving clusters visually [19], which is critical to understanding large graphs [10]. It is worth noting that, as Andreas Noack remarks [25], *"layouts that group densely connected nodes and separate sparsely connected nodes ... often violate aesthetic criteria like small edge lengths or uniformly distributed nodes,"* hence the incompatibility between the two approaches to evaluating layouts. Connected-closeness aims at assessing the mediation of the topology, that is the correlation between the layout (visually close nodes, notably visual groupings) and the graph structure (connected nodes, notably clusters); and it does it with a statistical metric.

Statistical metrics are often used to compare the node placement to the graph structure, but typically as a benchmark or as an optimization goal for new visualization techniques, and not as a way to inform a network map reader. For instance, stress functions that some layout algorithms aim to minimize [3] or majorize [8, 16], structural embeddedness [24, 27], or neighborhood preservation [1, 36]. These can be seen as layout faithfulness measures, as they compare a node placement to a given feature of the graph structure. However, contrary to connected-closeness, these metrics are not designed to be understandable to an audience untrained in graph theory, like most humanities scholars or data journalists. That being said, connected-closeness is similar to neighborhood preservation: while stress functions compare the layout distances between *all* pairs of nodes to a chosen graph distance (e.g., shortest path length), neighborhood preservation only compares a small subset of node pairs (neighborhoods in the graph space or the layout space). Connected-closeness similarly looks for neighborhoods in the layout space, although it compares it to the simplest graph distance: being connected or not. Like neighborhood preservation, it quantifies the general public's expectation that visual neighborhoods indicate structural clusters [19], but it pushes the logic further by finding the visual distance where the correlation is the strongest, and displaying it visually.

This focus on neighborhoods is also at the core of nonlinear projection methods [7, 18, 36], where it is hypothesized that "pure distance preservation might not be the ideal aim of a good graph layout method. Rather, having groups of nodes topologically close in the graph be also geometrically close in the layout allows reading the drawing well" [18]. Connected-closeness is nonlinear in the exact same way, ignoring the structure beyond neighborhoods. I will return to this question with more insights after I have exposed the details of the metric.

3 Motivation

Most node-placement (embedding) techniques use edge length as a criterion. Stress majorization strategies aim at specially defined distances [9, 16], while force-driven algorithms aim at the shortest distances possible [6, 11, 15, 25]. From the standpoint of a scholar interpreting the layout visually, this suggests that node distances can be interpreted directly, following the intuition that visually close nodes are topology close ; unfortunately this assumption is most often false, which leads to misinterpretations: visual distances correlate poorly with graph distances such as shortest path length or mean commuting time [38].

Even though a force-driven placement algorithm tries to minimize edge lengths, the constraints are generally too strong for it to succeed. Some edges have to remain long, while many disconnected nodes have to be packed together. This is for instance the case of large star networks in two dimensions (figure 2). Complex networks having a heavy-tailed degree distribution are subject to the same constraints: their highly-connected nodes raise the same issue as the star of figure 2.

As a consequence, the simple interpretations that come to mind are not necessarily true. It is not true in general that all connected nodes are close, or that disconnected nodes are distant. Two close nodes are not necessarily connected, and two distant nodes are not necessarily disconnected. Yet we expect connected nodes to be closer, on average, than disconnected nodes. Can we build a quantifiable (and true) statement that captures the effect of the layout?

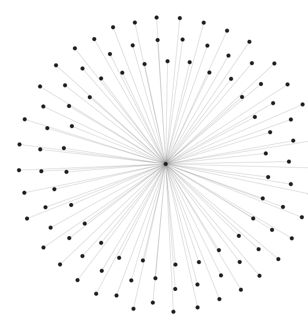


Figure 2: A star network with 100 nodes. Layout: Force Atlas 2 [15]. Connected node pairs are less close than many disconnected pairs.

This work aims at providing a quantified and explicit statement about node distances and network topology, in order to help scholars formulate reproducible interpretations. I do not claim that it is the only possible interpretation, let alone the best, but at least it is quantified and fits most force-directed placements. This statement is, in short, “most connected nodes are very close.” The “most” and the “very close” are quantified, and the validity of the statement is precisely assessed (figure 1). It is a visual statement insofar as “very close” is expressed as a distance that must be plotted. The quantification depends on the layout and has to be computed.

As we will see, this statement captures two features of force-driven placement algorithms: the nonlinearity of their neighborhood preservation, and their priorization of recall (minimizing the number of long edges) over precision (minimizing the number of close but disconnected node pairs).

4 Problem formulation

The goal of this endeavor is to find a quantified version of the statement “most connected nodes are very close” for a given network and layout. Let us have a network and a node placement in 2 dimensions. I refer to the distances in this arbitrary space as “Euclidean distances” to differentiate them from other distances such as the geodesic distance (length of a shortest path). For a given

Euclidean distance Δ, we can find the exact set of edges (or node pairs) whose Euclidean distance is shorter (figure 3).

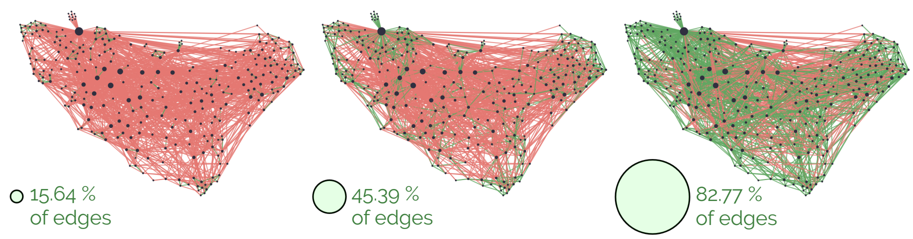


Figure 3: Network: C. Elegans [41]. Layout: LinLog [25] (Force Atlas 2 implementation [15]). A given part of edges are smaller than a given selection distance Δ. The diameter of the circle represents the selection distance Δ. Shorter edges are in green, longer edges in red.

We compute different indicators for each selection distance Δ (see figure 4), from zero to the size of the network map, or if you prefer, the layout distance separating the most distant nodes. The percentage of edges shorter than Δ is noted $E_{\%}(\Delta)$ and plotted in black. It ranges from 0% (unless some nodes perfectly overlap) to 100% (because at some point Δ is bigger than the whole layout). The percentage of node pairs closer than Δ (including the disconnected ones) is noted $p_{\%}(\Delta)$ and plotted in blue. By definition both quantities are strictly increasing, and in practice $E_{\%}(\Delta)$ is always higher than $p_{\%}(\Delta)$ (black over blue) because force-driven algorithms try to make edges shorter (but we could build an anti-force-driven layout that aims at making edges longer and the black curve would be under the blue curve). In other terms, an expected effect of the node placement algorithm (but not guaranteed) is to overrepresent connected pairs captured at any selection distance Δ, due to the attraction force between connected nodes.

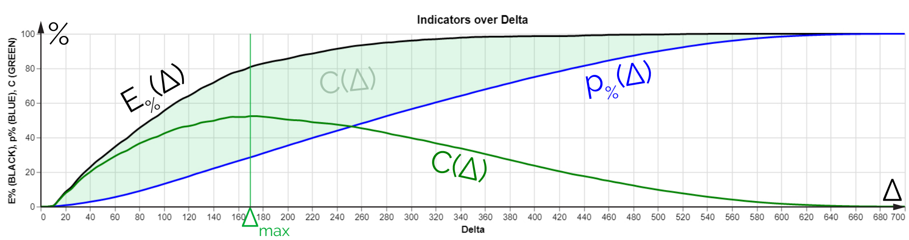


Figure 4: Indicators computed over the selection distance Δ for the network and layout of figure 3. Black curve: $E_{\%}(\Delta)$ the percentage of edges shorter than Δ. Blue curve: $p_{\%}(\Delta)$ the percentage of node pairs closer than Δ. Green area: $C(\Delta) = E_{\%}(\Delta) - p_{\%}(\Delta)$ the connected-closeness. Green curve: idem. Green bar: Δ_{max} the distance of maximum connected-closeness.

If the edges had no influence on the node placement (e.g., if the network was rewired randomly after the node placement) then the blue curve $p_{\%}(\Delta)$ and the black curve $E_{\%}(\Delta)$ would resemble each other, because the distance of *connected* pairs (in black) would have nothing special compared to *any* pair (in blue). As $p_{\%}(\Delta)$ (in blue) is a natural point of comparison, the situation presents an opportunity. Indeed, the share of node pairs $p_{\%}(\Delta)$ can be understood as the share of edges *if they were distributed at random*, if the network were rewired at random after the layout was computed.

Let us call this the "expected share of edges shorter than Δ", where "expected" means "in a randomly rewired network" (for the same layout). We can then compare the *actual* proportion of edges (in black) to the *expected* proportion (in blue). The difference between the black and the blue curve, highlighted as a greenish area and also plotted as the green curve, is therefore an important metric: the proportion of edges shorter than Δ above expectations (a formal definition is given in the next section).

The green curve is null at both ends (because $E_{\%}$ and $p_{\%}$ both range from 0% to 100%), so it has to reach a maximum in between (plotted as a vertical green line). This point is remarkable. First, the higher the green curve, the more "unexpected" edges are captured by the layout (this is the most dramatic statement we can make); "unexpected" meaning here "compared to the same layout but with edges rewired at random". Second, it provides the precise distance where the layout is the most efficient, which is precious practical information about the visualization. Identifying this point allows forging a quantitative and informative statement such as "X% of edges are *unexpectedly* shorter than Δ", where X is as high as possible. I propose to call the quantity represented by the green curve "connected-closeness" and turn its maximum into a network layout metric. In practice, the curve may have a plateau, which challenges the existence of a single maximum. I address this issue in section 6 "Implementation".

5 Definitions

5.1 Naming

- $C(\Delta)$ is called the *connected-closeness* for a given Euclidean distance Δ. It measures the *percentage of unexpectedly-close connected nodes*, where close means closer than Δ, and connected means adjacent.
- Δ_{max} is called the *distance of maximum connected-closeness.*
- $C_{max} = C(\Delta_{max})$ is called the *maximum connected-closeness.*

5.2 Definition of connected-closeness

$$C(\Delta) = \frac{E(\Delta) - E_{expected}(\Delta)}{E(\infty)} \tag{1}$$

- $E(\Delta)$ is the number of edges shorter than the Euclidean distance Δ
- $E(\infty) = e$ is the total number of edges (i.e., the graph size e)
- $E_{expected}(\Delta) = E(\infty) \times \dfrac{p(\Delta)}{p(\infty)}$ is the number of *expected* edges shorter than Δ
- $p(\Delta)$ is the number of node pairs closer than the Euclidean distance Δ
- $p(\infty) = n \times (n-1)$ is the total number of node pairs (n is the graph order, its number of nodes)

Remark that by definition $\frac{p(\Delta)}{p(\infty)}$ represents the probability that a node pair picked at random gets placed closer than Δ, but also the probability that an edge redistributed at random gets placed closer than Δ. Indeed, edges redistributed at random have the same probability to appear for all node pairs. As a consequence, $E_{expected}(\Delta)$ represents *the average number of edges that would be placed closer than* Δ *if all edges were redistributed at random.* This is what "expected" means in this context.

5.3 Other useful quantities

- $E_{\%}(\Delta) = \dfrac{E(\Delta)}{E(\infty)}$ is the share of edges shorter than Δ.
- $p_{\%}(\Delta) = \dfrac{p(\Delta)}{p(\infty)}$ is the share of node pairs placed closer than Δ
- $P_{edge}(\Delta) = \dfrac{E(\Delta)}{p(\Delta)}$ is the probability for two nodes closer than Δ to have an edge.

Using these quantities we can rewrite connected-closeness as follows:

$$C(\Delta) = \frac{E(\Delta)}{E(\infty)} - \frac{p(\Delta)}{p(\infty)} = E_{\%}(\Delta) - p_{\%}(\Delta) \tag{2}$$

The two characterizations of connected-closeness (1) and (2) are trivially equivalent (by definition of $E_{expected}$) but they do not read the same. The first reads as "the unexpected number of edges divided by the total number of edges" which corresponds to the intuition we aim at quantifying, while the second reads as "the black curve minus the blue curve" which corresponds to how the indicators behave in figure 4.

5.4 Definition of the maximum connected-closeness

C_{max} is defined as the maximum of $C(\Delta)$ for all Δ. Δ_{max} is the Euclidean distance where $C(\Delta)$ is maximum. In case of ties, the smallest Δ is retained (see section 6 for additional details).

Note: $C(\Delta)$ is better when it is higher, while Δ is better when it is smaller. The "max" in Δ_{max} corresponds to the maximization of C (the connected closeness).

5.5 Quantified mediation

Our target intuition (figure 1) is quantified by $E_{\%}(\Delta_{max})$. The resulting statement "$E_{\%}(\Delta_{max})$ of connected nodes are closer than Δ_{max}" quantifies how the layout mediates the topology under a number of assumptions. It assumes that we compute $E_{\%}(\Delta_{max})$ in practice, which requires finding Δ_{max} in the first place. It also assumes that Δ_{max} is represented visually or rendered meaningful another way. And it assumes that C_{max} is significantly greater than zero, else its C is basically flat and the indicators are degenerate. I address the questions raised by these assumptions in the next two sections.

It is worth noting that all the metrics defined above, including C_{max} and Δ_{max}, are fully determined by the node placement. They cannot be different for the same set of node coordinates. However, the optimization proposed below introduces a non-deterministic element via random sampling, as a trade-off for a better performance. In this situation, C_{max} will better preserve its true value than Δ_{max}, for the reasons that I will develop when it comes to degenerate cases. In short, notwithstanding optimization, the layout distance Δ_{max} characterizes the node placement. Furthermore, as the benchmark will show, it is also remarkably stable through multiple re-runs of the same non-deterministic layout algorithm: in that sense it also characterizes the application of a given algorithm to a given network. In other words, connected-closeness quantifies an aspect of how the layout mediates the graph structure, that is by containing a certain amount of edges ($E_{\%}(\Delta_{max})$) under a certain visual distance (Δ_{max}).

6 Design and visualization

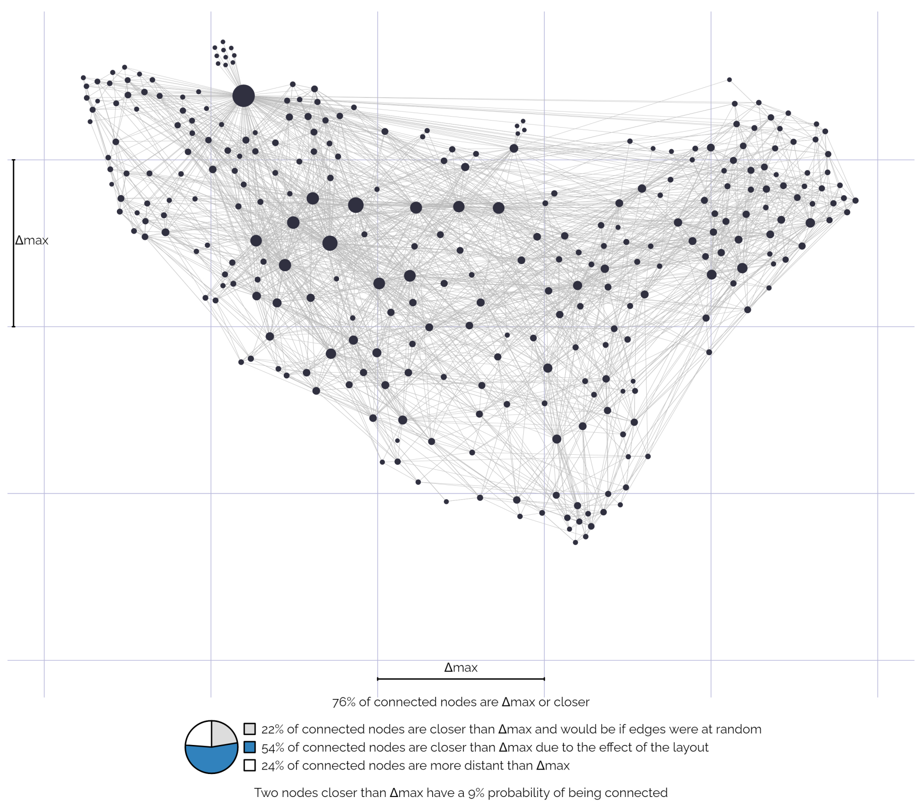


Figure 5: C. Elegans [41] (Layout: LinLog [25]) visualized using Δ_{max} as a grid. The legend provides some necessary context.

As the quantification is in part visual (Δ_{max} directly relates to the visualization) the graphic design matters. I offer a few recommendations to make the most out of connected-closeness. These guidelines are implemented in figure 5.

- Δ_{max} should be represented visually. I suggest using a grid, as the repetitiveness of the cells makes it easier to estimate visually whether a given pair of nodes is shorter than Δ_{max}.

- The most directly useful statement with connected-closeness is $E_{\%}(\Delta_{max})$ because it quantifies the intuition about the layout as a mediation. It should be prioritized as the first information to communicate (along with Δ_{max}). In the context of figure 5 it reads as "76% of connected nodes are Δ_{max} or closer".

- The readers of the network map cannot be expected to know about connected-closeness, thus the meaning of the indicators should be stated in plain English (or any language). The formalism is meaningless to most readers and should be dropped, but as Δ_{max} has to be mentioned, its original notation may be retained. I made that choice in figure 5 but an explicit name such as "characteristic distance" might be more efficient at communicating the idea.

- The second priority is to provide a justification about the choice of Δ_{max}. The reader cannot infer from $E_{\%}(\Delta_{max})$ what makes Δ_{max} special, because Δ_{max} does not maximize $E_{\%}$ but the connected-closeness C. Thus the second priority is to state the meaning of C_{max} in plain English to provide a ground for the choice of Δ_{max}; for instance, in the context of figure 5: "54% of connected nodes are closer than Δ_{max} due to the effect of the layout".

- Once C has been stated, it is possible to provide a meaning for Δ_{max}. For instance "Δ_{max} is the distance at which the effect of the layout is maximum". In figure 5 I did not state the meaning of Δ_{max} explicitly because it is less informative, but that choice is debatable. I chose instead to clarify the meaning of connected-closeness with a pie chart featuring $E_{expected}(\Delta_{max})/E(\infty)$ and C_{max}, and I hinted at the connection between C_{max} and Δ_{max} by using the same color for the grid and the slice of the pie chart. In the context of a publication, the most complicated details should probably feature in the caption of the main text.

- Last but not least, a common but regrettable misunderstanding should be prevented. The probability of A given B is often confused with the probability of B given A. We should make it clear that the probability for two connected nodes to be close does NOT equate with the probability for two close nodes to be connected. Even in cases where the layout succeeded at making most edges quite short, one expects most close nodes to be disconnected. We should avoid the misunderstanding that closeness implies connection. To address this issue I propose to simply state the probability P_{edge} in plain English for the sake of clarity, even though it is rarely informative in practice. In the context of figure 5: "two nodes closer than Δ_{max} have 9% chances of being connected". Remark that it is much lower than the 54% of C_{max}.

6.1 Resistance to misuses

Beginners and non-experts might be inclined to put too much trust into indicators whose validity conditions they do not fully master [20]. It is thus important to push back against docility and proactively implement protections against the most preventable misinterpretations.

The statement proposed by connected-closeness only makes sense insofar as Δ_{max} captures a significant share of edges. When C remains too low its curve is basically flat and finding Δ_{max} and its related indicators face a degeneracy issue. In addition, Δ_{max} is not really characteristic of the layout because connected-closeness fails to capture any effect of the layout. For instance in the case of the random layout, Δ never captures more edges than if they were redistributed at random (see the benchmark in the appendix). Displaying Δ_{max} while C_{max} is close to zero would imply that the layout has a characteristic distance while it has not, even though we can always compute Δ_{max} in practice.

For this reason, as a protection against misuses, I propose to display Δ_{max} if C_{max} is above a threshold that I fixed at 10% because it provided satisfying results in my limited testing (it can be adapted and should be debated). Below that threshold, Δ_{max} and all indicators except C_{max} are declared not applicable, in plain English (figure 6).

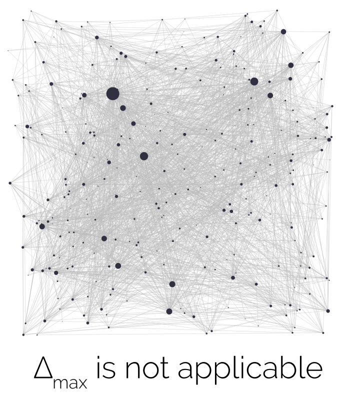


Figure 6: C. Elegans [41], random layout, visualized using Δ_{max}. As C_{max} is below 10%, Δ_{max} is declared non-applicable.

7 Implementation

7.1 Fixing degenerate cases

Certain realizations of $C(\Delta)$ have a plateau on top, which makes finding C_{max} problematic (figure 7). This does not happen only when C is very low. It may also happen when it is very high and thus, intuitively, meaningful (degeneracy is discussed in subsection 9.1). In such cases we should use the smallest Δ as a tie-breaker, as smaller distances are more informative (a smaller Δ_{max} is more selective, see figure 3). However, in some cases the plateau has tiny fluctuations that do not provide the expected solution (what appears to be the leftmost point of the flat part).

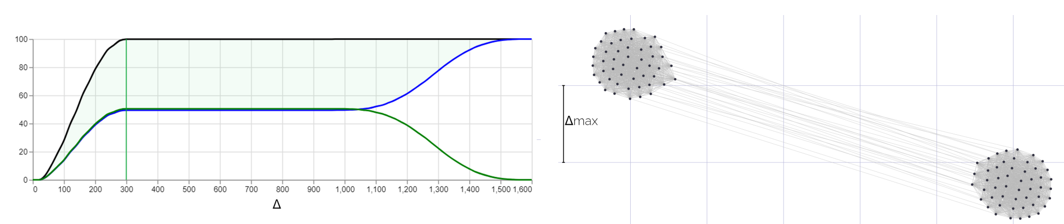


Figure 7: A planted partition model with $P_{in} = 99\%$. Layout: Force Atlas 2 [15]. The function $C(\Delta)$ (green) has a plateau, corresponding to the gap between the two clusters. The smallest distance is preferred, as it is the most informative.

I propose as a basic solution to use a tolerance parameter ϵ. The definition of connected-closeness is adapted to allow picking the smallest distance that fits the maximum of $C(\Delta)$ *within a tolerance of* ϵ. In my limited testing I settled on a value of $\epsilon = 0.03$ (i.e., 3%).

7.2 Optimization

The steps of a naive implementation are:

1. Compute the distances of all pairs of nodes
2. Sample the distances to filter out those that are too similar
3. Compute indicators for each remaining distance
4. Pick the max of $C(\Delta)$ to define Δ_{max} and relevant indicators

Computing and storing the distances between all node pairs is expensive. Then the sampling of distances is relatively straightforward, for instance using regular intervals up to the longest distance between node pairs. Yet if the network is sparse, the necessary precision may require small intervals. Both aspects can be optimized.

The distribution of $E_{\%}(\Delta)$ and $p_{\%}(\Delta)$ can be estimated by a simple random sampling of node pairs. I found a 10% sampling satisfying and lower percentages remained acceptable. As the connected-closeness curve itself is less important than its maximum, the errors on C_{max} and Δ_{max} are lower than the average error on $C(\Delta)$. But there is another easy improvement: adding the list of edges to the sampling. Indeed, many networks are sparse and edge lengths need to be computed anyway. To keep it simple, I propose to sample as many pairs as there are edges. In the case of C. Elegans [41], it means a sampling of 2.7% node pairs, which leads to an error of 3.7% on C_{max} and 3.3% on Δ_{max}.

Finding C_{max} can be optimized by a grid search strategy. We split the range of distances into a number of equal parts, and we pick the part that is the most promising (where $C(\Delta)$ is the highest). Then we iterate until the precision is satisfying. We leverage here the empirical observation that $C(\Delta)$ is a relatively smooth curve. The steps are as follows:

1. Initialize the search with a range from 0 to the largest distance Δ, and set a grid size s.
2. Compute $C(\Delta)$ in s equally separated distances across the whole range.
3. Pick the highest $C(\Delta)$ among the data points, and use the range from the previous to the next data point for the next iteration.
4. Iterate over 2. and 3. until C_{max} is not improved significantly (or at all).

A reference implementation of this optimized algorithm, including the ϵ tolerance modification, is available online in a public Javascript notebook[1].

7.3 Running time

On a consumer laptop, for the network of figure 5, the optimized implementation computes in 10 ms to 20 ms, versus 350 ms to 400 ms for the naive implementation. As the number of edges exceeds 10,000, the optimized implementation takes seconds to compute, while the naive implementation takes minutes. The table below shows how the computation time scales for square lattices.

[1] https://observablehq.com/@jacomyma/efficient-implementation-of-connected-closeness

Network	# nodes	# edges	Time OPTIM.	Time NAIVE
Square lattice 2x2	4	4	1 ms	1 ms
Square lattice 4x4	16	24	1 ms	2 ms
Square lattice 8x8	64	112	2 ms	16 ms
Square lattice 16x16	256	480	5 ms	238 ms
Square lattice 32x32	1,024	1,984	27 ms	8,238 ms
Square lattice 64x64	4,096	8,064	326 ms	259,645 ms
Square lattice 128x128	16,384	32,512	4,542 ms	

Table 1: Computation time for the optimized and naive implementations of connected-closeness on square lattices of various sizes, on a consumer laptop.

8 Benchmark (highlights)

I conducted a benchmark on connected-closeness using 14 network generators (e.g., stochastic block model with different settings) and 7 node-placement algorithms available in Javascript libraries (Javascript is popular for online data visualizations). For each *(generator, layout)* pair I generated and visualized 100 networks of 100 nodes, and computed the main indicators on the resulting layout: Δ_{max}, C_{max}, $E_{\%}(\Delta_{max})$, $p_{\%}(\Delta_{max})$, and $P_{edge}(\Delta_{max})$. The benchmark data and visualizations are detailed in appendices. Here I only highlight the main takeaways.

Connected-closeness is a remarkably stable metric. Force-directed placement algorithms are non-deterministic, and resulting node coordinates have a high variance. Yet we know that the patterns produced (e.g., clusters, center-periphery relations) can be stable. Connected-closeness captures this aspect very well.

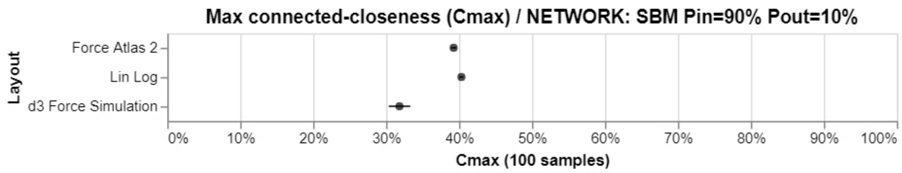
Perhaps not so surprisingly, the C_{max} of a same network with a community structure (e.g., two cliques linked by a bridge) visualized by a force-directed layout (e.g., LinLog [25]) has a standard variation as low as 0.00819% for an average C_{max} (over multiple renderings of that same layout) of 50.5%.

More surprisingly, C_{max} is also stable for different networks with similar properties. For instance a stochastic block model with two blocks, an internal link probability of 90% and an external link probability of 10%. 100 different networks rendered with LinLog give an average C_{max} of 40% with a standard deviation of 0.49%. The benchmark shows consistently similar results for other network models and other force-directed layout algorithms.

Force-directed layouts perform well in the presence of a community structure. This is expected considering that, as Noack wrote, "Modularity clustering is force-directed layout" [26]. Connected-closeness confirms it: C_{max} is consistently high for such networks (figure 8).

As the community structure decreases, so does connected-closeness. A stochastic block model allows quantifying the influence of the community structure (figure 9). As the internal link probability P_{in} drops closer to 50%, the community structure disappears. C_{max} drops accordingly (figures 9 and 10).

Connected-closeness is low for random layouts, as expected. Indeed, by definition they do not even try to bring connected nodes closer; they put all nodes randomly close regardless of



Layout	C_{max} Avg.	C_{max} S.D.
Force Atlas 2	39%	0.70%
Lin Log	40%	0.49%
d3 Force Simulation	32%	2.90%

Figure 8: C_{max} for 100 renditions of a stochastic block model with $P_{in} = 90\%$ (probability of an edge between two nodes of the same block) and $P_{out} = 10\%$ (between two nodes of different blocks), for three different force-directed layouts. The dots represent the average, the error bars the standard deviation.

whether or not they are connected. Their connected-closeness is close to zero (figure 11).

In the same perspective, "bad" layouts perform worse than "good" layouts on the same network. The benchmark features a "bad" parametrization of Force Atlas 2: the strong gravity and the low resolution of the simulation introduce randomness, and the layout is stopped after too few steps. The resulting layouts are visibly glitchy (figure 12 at the center). Following intuition, this layout performs half-way between the default settings of Force Atlas 2 [15], and a random layout (figure 13). Remark that even for bad and random layouts, the standard deviation of C_{max} is very low.

Force-directed layouts perform better on sparse networks. Bringing connected nodes closer is less constrained in networks with a low density. Force-directed algorithms are very good at producing a high C_{max} in this context (figures 14 and 15).

Layouts are pointless on cliques. This follows intuition: as every node is connected to every other node, all are topologically equivalent, and the node placement is meaningless unless all nodes are stacked on the same coordinates. By definition C_{max} is always null for cliques, as rewiring edges makes no difference whatsoever. The same argument is true for stables (networks with no links).

The denser the network, the lower C_{max} is achieved by force-directed layouts. This intuitively follows the case of cliques. We can see it experimentally by comparing random networks with different densities (figure 16).

More communities increases C_{max}. Indeed, a higher count of dense clusters sensibly decreases $p_{\%}(\Delta_{max})$, the share of node pairs closer than Δ_{max}, without decreasing much $E_{\%}(\Delta_{max})$, the share of *edges* shorter than Δ_{max}. Let us compare networks of 100 nodes with groups internally connected with a 99.9% chance against 0.01% from one group to another, and using the Force Atlas 2 layout. For 5 groups we obtain $C_{max} = 81\%$ (figure 20 in appendix), against $C_{max} = 50\%$ for only 2 groups (figure 21 in appendix). $E_{\%}(\Delta_{max})$ is essentially the same (97% and 99% respectively) while $p_{\%}(\Delta_{max})$ is very different (16% and 49% respectively).

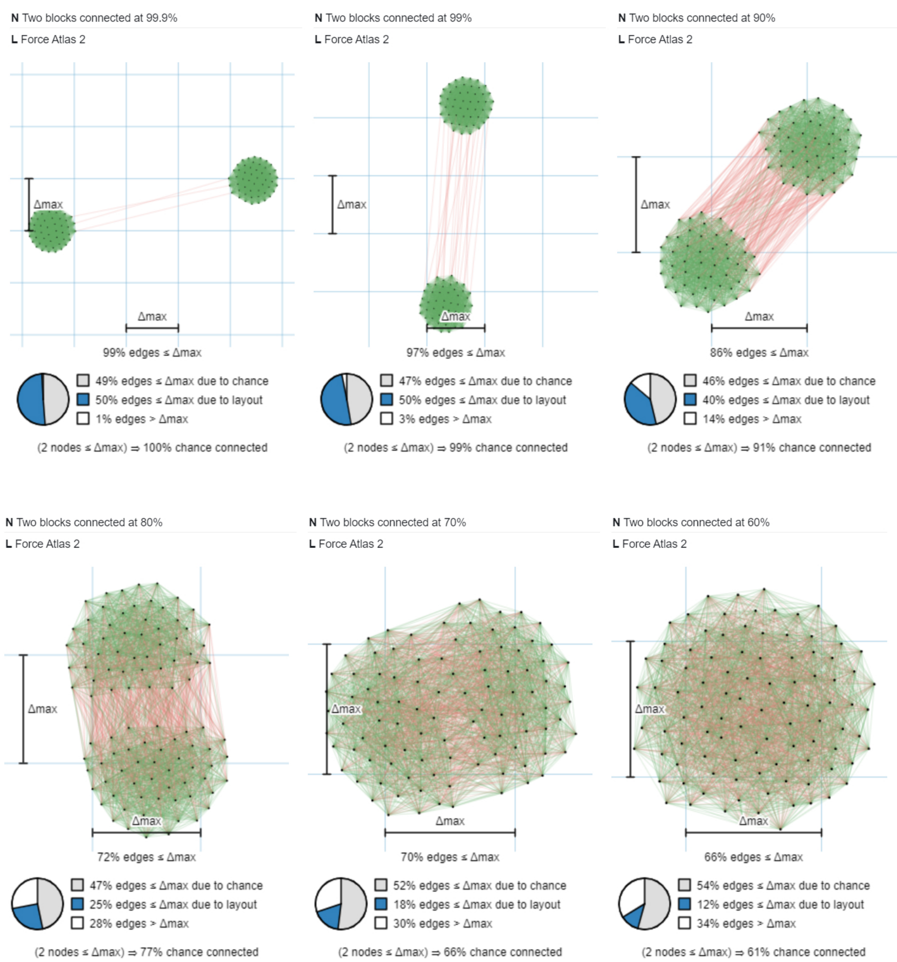
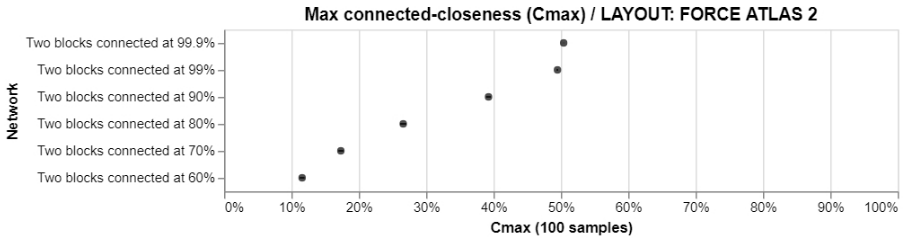
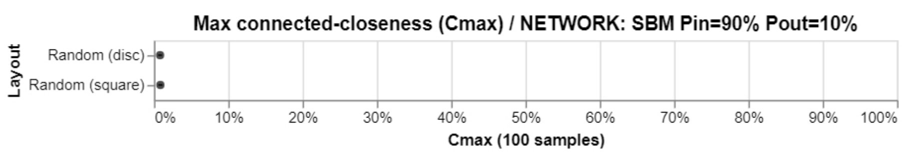


Figure 9: Samples of stochastic block model networks with a decreasing community structure. Layout: Force Atlas 2 [15]. P_{in} in reading order: 99.9%; 99%; 90%; 80%, 70% and 60%. C_{max} in reading order: 50%, 50%, 40%, 25%, 18% and 12%



Network	C_{max} Avg.	C_{max} S.D.
Two blocks connected at 99.9%	50%	0.06%
Two blocks connected at 99%	49%	0.21%
Two blocks connected at 90%	39%	0.70%
Two blocks connected at 80%	27%	0.92%
Two blocks connected at 70%	17%	0.70%
Two blocks connected at 60%	12%	0.67%

Figure 10: C_{max} for 100 renditions of stochastic block models with decreasing P_{in}. With force-driven layouts, C_{max} is high in presence of a strong community structure. Layout: Force Atlas 2. The dots represent the average, the error bars the standard deviation.



Layout	C_{max} Avg.	C_{max} S.D.
Random (disc)	1%	0.50%
Random (square)	1%	0.54%

Figure 11: C_{max} for 100 renditions of a network with a community structure with random layouts. C_{max} is close to zero for random layouts regardless of the network. The dots represent the average, the error bars the standard deviation.

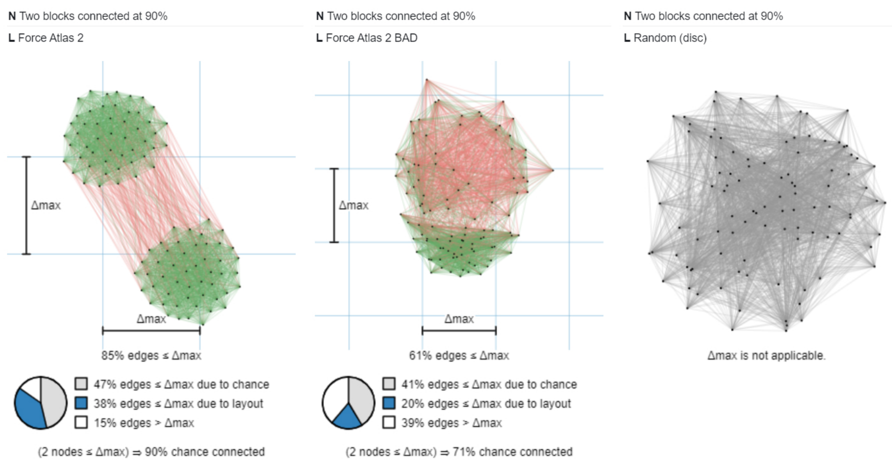
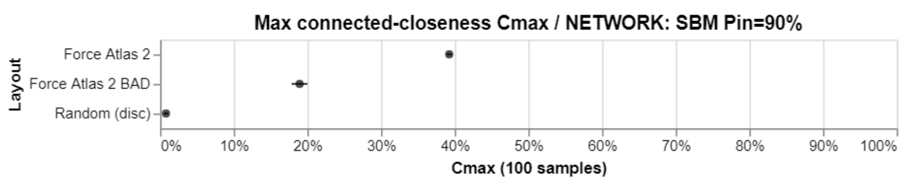
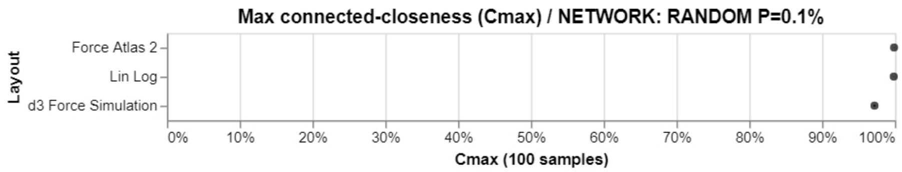


Figure 12: The same network with a community structure, rendered with three layouts. Left to right: Force Atlas 2 with default settings, Force Atlas 2 with intentionally bad settings, random layout.



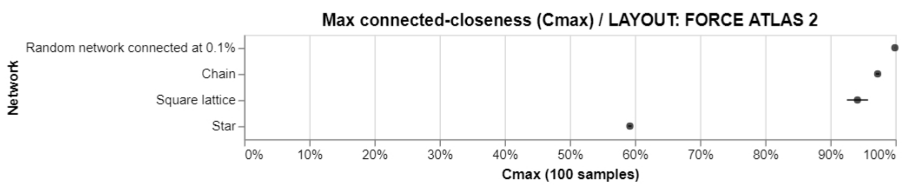
Layout	C_{max} Avg.	C_{max} S.D.
Force Atlas 2	39%	0.70%
Force Atlas 2 BAD	19%	2.12%
Random (disc)	1%	0.50%

Figure 13: C_{max} for 100 renditions of a network with a community structure with three layouts, from "good" to "bad". C_{max} is only as high as the layout is "good" (in presence of a strong community structure). The dots represent the average, the error bars the standard deviation.



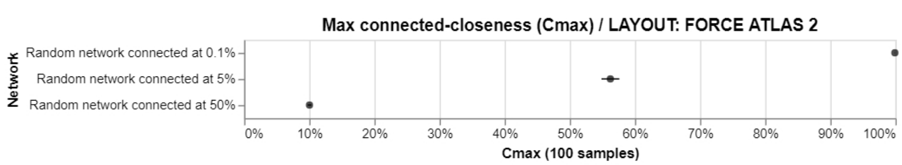
Layout	C_{max} Avg.	C_{max} S.D.
Force Atlas 2	100%	0.04%
Lin Log	100%	0.05%
d3 Force Simulation	97%	0.28%

Figure 14: C_{max} for 100 renditions of a sparse random network (link probability 0.1%) with three different force-directed networks. C_{max} is high for sparse networks visualized with any force-driven layout. The dots represent the average, the error bars the standard deviation.



Network	C_{max} Avg.	C_{max} S.D.
Random network connected at 0.1%	100%	0.04%
Chain	97%	0.52%
Square lattice	94%	3.19%
Star	59%	0.47%

Figure 15: C_{max} for 100 renditions of a chain network with three different force-directed networks. C_{max} is high for most but not all sparse networks (it is lower for star-shaped networks) with force-driven layouts. The dots represent the average, the error bars the standard deviation.



Network	C_{max} Avg.	C_{max} S.D.
Random network connected at 0.1%	100%	0.04%
Random network connected at 5%	56%	2.72%
Random network connected at 50%	10%	0.52%

Figure 16: C_{max} for 100 renditions of a random networks with three different densities. Sparse random network visualized with a force-driven layout achieve a high C_{max}. Layout: Force Atlas 2 [15]. The dots represent the average, the error bars the standard deviation.

9 Discussion

9.1 Degenerate cases

I encountered only three distinct cases of degeneracy.

Case 1: the network consists of well-delineated clusters (figures 7, 19-22(a)(c) and 23(c)). The nodeless spaces between the clusters generate plateaux in $C(\Delta)$, because any distance bigger than one cluster but smaller than the inter-cluster gap is evaluated the same.

Case 2: the layout adds no information, either because it is random (figures 19-32(d)(e)), or because the network is too random and dense (figure 25), a clique (figure 18) or a stable. In this case, $E_{\%} \approx p_{\%}$ and $C \approx 0$. C is pointless, Δ_{max} has no relevant meaning, and the assumption that distances mediate connectedness is wrong.

Case 3: the C curve is low but not quite null, and flat but not totally. In such case the degeneracy is low but still present in the sense that Δ_{max} gets into chaotic regime: minor changes in node positions have a low impact on C_{max} but a big impact on Δ_{max}. This happens either because the layout is bad and adds randomness (figures 19-32(b)), or because the clusters are too close and entangled (figures 25-26(a)(c)).

9.2 Design space

My rationale for design being stated, let me acknowledge that my recommendation should now be tested. There are alternatives to the grid (hexagons, circles...). It might be more productive to color edges depending on how their length compares to Δ_{max}. The metric might be better understood to certain publics framed as a confusion matrix with precision and recall (see next subsection), etc. Trying and assessing these options is beyond the scope of this paper.

9.3 How connected-closeness relates to precision and recall

Precision and recall are performance metrics commonly used in pattern recognition, information retrieval and machine learning. They come up when testing the accuracy of a retrieval task, where one tries to find which elements of a given set have a certain feature. Precision measures how many retrieved elements have the feature, while recall measures how many elements with the feature were retrieved. False positives lower precision, while false negatives lower recall.

If we assume that the visualization carries out the task of retrieving connected nodes by selecting the node pairs closer than Δ_{max}, then by definition:

- **precision** is $P_{edge}(\Delta_{max})$, the probability for two nodes closer than Δ_{max} to be connected;
- **recall** is $E_{\%}(\Delta_{max})$, the share of edges shorter than Δ_{max}.

I argued in section 6 that the recall $E_{\%}(\Delta_{max})$ is directly useful to the reader, while, on the contrary, the precision $P_{edge}(\Delta_{max})$ only serves as a protection against misinterpretation. Indeed, as figure 2 illustrates, even simple networks may be impossible to arrange so that *only* connected nodes are close. This effect is particularly severe in networks with a heavy-tailed degree distribution, and for many empirical networks, a high precision is unattainable in practice. In that case, maximizing recall is more productive. Force-driven node placement algorithms typically do so, as in their design, edges influence the attraction force but not the repulsion force [15, 25]. They aim to bring connected nodes closer even if it also brings disconnected nodes closer; they minimize false negative and ignore the production of false positives, they seek recall and disregard precision.

The literature about dimensionality reduction calls precision "trustworthiness" and recall "continuity", and identifies the aforementioned tradeoff between them [37]. Networks are not strictly equivalent to multidimensional datasets but the similarity is sufficient to make the connection. We can then say network maps do not offer a trustworthy neighborhood preservation in general, even when it has a good continuity.

For a star-shaped network of 100 nodes, force-driven layouts typically obtain a recall above 95% for a precision below 10% (data in appendix A.7). Yet high precision can be attained for certain networks. For two cliques bridged by a single edge, force-driven layouts find a recall *and a precision* above 90% (see appendices A.4 and A.6). However, such extreme community structures are rare in practice, as empirical networks often feature a heavy-tailed degree distribution, like the network from figure 5, with a recall of 76% for a precision of 9%.

When facing a network map, one should *not* expect that distances mediate connectedness with a good precision, although recall could be good. In general, one should expect many disconnected nodes to be close. Network layouts are flawed as neighborhood preservation techniques; but who said it was their goal?

The precision issue is not relative to the layout *per se*, but to the assumption that distances directly measure connectedness. We have at least three reasons to consider this assumption wrong. First, as a methodological commitment, such assumption should be considered wrong by default, until we have a reason to hold it for true. Second, the precision issue precisely tells us that this interpretation is not always appropriate. Third, the graph drawing literature does not explicitly support this interpretation. Noack, for instance, argues that force-driven layouts make the community structure visible, which is a sensibly different question [26]. The precision issue tells us that we cannot take for granted that distances accurately mediate connections, but it may sometimes have a good recall and a bad precision, and in rare situations, a good recall and a good precision. Connected-closeness tells us how valid is this interpretation for a given network map; and as we have seen, it does not work in every situation.

9.4 Beyond the assumption that distances measure connectedness

If the assumption that distances measure connectedness is wrong in general, then why not pick a better one? My answer is twofold. First, although it is not true in general, it *can* be true, and when it is, it is useful. Second, I did not find a simple enough and truer alternative. Simple, because interpretations are only as useful as the audience understands them; and at least true in more situations: more solid, more grounded. The geodesic distance is a candidate, by it poorly correlates with the Euclidean distance [38], and the interpretation is less straightforward, more technical. Mean commuting time is another candidate, but it does not correlate at all [38]. As it is even more complicated to explain than the geodesic distance, there is little hope to obtain a useful result in practice. Other distances could make candidates (e.g., resistance distance [17]), but we have no reason to believe that they correlate with layout distance.

However, we do have reasons to consider the assumptions suggested by graph drawing algorithms. Considering that force-driven [6, 11, 15, 25] and stress majorization [9, 16] algorithms explicitly aim at minimizing edge length, the first assumption we should test is that connected nodes are close, which is precisely the assumption of connected-closeness. But another ambition is frequently stated: manifesting the community structure. Following Noack [26] we may consider the assumption that layout distances measure the belonging to a same community. Noack refers to modularity clustering [23], but there are other approaches to community detection [33]. These techniques share an operationalization problem: as all community detection methods "exhibit at least some degree of degeneracy" [29], they do not offer a stable ground to build a practical measure. There are multiple, equally valid ways to partition a given network in communities. As a result, the measure would not only depend on the layout, but also on the partition found. This added dependency causes two major issues: for a given network, the value would not be characteristic of the layout, and the interpretation would now require understanding the topic of community detection (which is far from simple).

I acknowledge, however, the intuition that force-driven layouts are better at putting groups together than producing short edges. I hypothesize that a continuous distance that reproduces the community structure without requiring to compute well-demarcated communities would correlate with the Euclidean distance and provide a better assumption for interpreting network maps in favorable cases. This endeavor is for a future work, and I hope that connected-closeness can bring us one step closer to an more grounded interpretation framework for network maps despite its limitations.

9.5 The nonlinear problem

The nonlinearity of neighborhood preservation in networks roots in two unrelated issues. The first is the lack of space in the Euclidean plane, which prevents high-degree nodes from having all their neighbors close, as seen in the star of figure 2. Hyperbolic spaces, nonlinear, can be used to solve this issue [21].

The second issue is the empirical criterion of cluster separation, the goal to "group densely connected nodes and separate sparsely connected nodes" [25]. To achieve this separation, a layout must necessarily make the bridging edges long (see figure 17). Manifesting the community structure is not the only possible goal for a layout, but it is common in practice, and it is incompatible with homogeneous edge lengths, which prevents linearity.

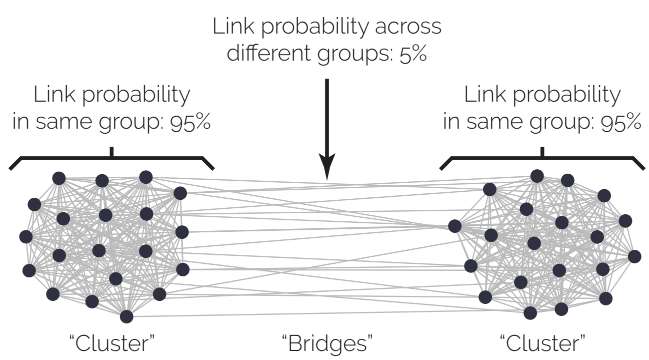


Figure 17: Neighborhood preservation in network layouts has to be nonlinear because cluster separation requires long edges (the bridges).

Connected-closeness is nonlinear in at least two ways because these two roots are reflected in its design. The first issue is reflected in the highlighting the recall $E_{\%}(\Delta_{max})$ and the downplaying the precision $P_{edge}(\Delta_{max})$. The overrepresentation of false positives (close but disconnected nodes) captures this nonlinearity. The second issue is reflected in the use of a threshold distance (Δ_{max})

for quantification. Not all edges can be short because of cluster separability, and intuitively, Δ_{max} relates to the size of communities.

Any alternative to connected-closeness will have to deal with the two same sources of nonlinearity.

10 Conclusion

In this paper I presented the connected-closeness, a metric consisting of two distinct parts: the maximum connected-closeness $C_{max} = C(\Delta_{max})$ and the distance of maximum connected-closeness Δ_{max}. Δ_{max} is characteristic of the layout, and is intended to be drawn on the visualization as a contextual element. C_{max} is characteristic of both the network and the layout, and is only high when, intuitively, the distances in the layout can be interpreted meaningfully. It allows a quantified statement about how the node placement mediates the topology of the network, such as "76% of connected nodes are Δ_{max} or closer". In practice, as the benchmark illustrates, connected-closeness offers us different opportunities:

1. It allows quantitative statements about the placement of nodes in the visualization. Different indicators provide different statements, some being simple enough to be directly useful to a non-expert audience.

2. Maximum connected-closeness C_{max} assesses the layout's validity (from a specific angle). Different layouts have different goals, and this metric accounts for one of them (bringing connected nodes closer). It states a game at which the layout is presumably good (following the arguments of algorithm designers [6, 11, 15, 25]), and checks that it actually is. The benchmark shows that force-directed layouts are indeed good at the game of bringing connected nodes closer, but also detects situations and layouts that are bad at it.

3. It allows comparability. As C_{max} is comparable on different layouts for a given network, it can be used as a layout quality metric to determine which algorithm performs better. It even allows comparability across network generators and across multiple renditions of the same non-deterministic layout. Optimization aside, the metric is deterministic and the benchmark has shown that its standard deviation is, in most situations, very low.

4. It shows that layouts capture something of the topological structure of networks. As force-driven placement algorithms are non-deterministic, one might conclude that they are not reliable. Practice tells otherwise, but it is not easy to quantify how similar are two renderings of the same network, by the same layout algorithm, while all node positions are different. Maximum connected-closeness C_{max}, as a highly consistent measure of network layouts, proves that some of them (e.g., force-directed algorithms) consistently capture an aspect of the topological structure of the network they represent.

5. It shows that interpreting layout distances as a proxy for connectedness cannot be taken for granted. The calculations provide the precision and recall of visual distances as a statistical measure of edge presence. Our empirical results show that even though this measure *may* have a good precision and recall (it depends on the network and the layout), it often has a good recall but a *bad* precision, or in the most random cases a bad precision and recall.

References

[1] R. Ahmed, F. De Luca, S. Devkota, S. Kobourov, and M. Li. Graph drawing via gradient descent, gd2. 2020.

[2] C. Bennett, J. Ryall, L. Spalteholz, and A. Gooch. The aesthetics of graph visualization. *Proceedings of the 2007 Computational Aesthetics in Graphics, Visualization, and Imaging*, pages 57–64, 2007. doi:10.2312/COMPAESTH/COMPAESTH07/057-064.

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

[4] A. Condon and R. M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Structures & Algorithms*, 18(2):116–140, 2001.

[5] C. D. Correa and K.-L. Ma. Visualizing Social Networks. In *Social Network Data Analytics*, pages 307–326. Springer US, Boston, MA, 2011. doi:10.1007/978-1-4419-8462-3_11.

[6] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, nov 1991. doi:10.1002/spe.4380211102.

[7] S. F. Galán and O. J. Mengshoel. Neighborhood beautification: Graph layout through message passing. *Journal of Visual Languages & Computing*, 44:72–88, feb 2018. URL: https://www.sciencedirect.com/science/article/pii/S1045926X16300477, doi:10.1016/J.JVLC.2017.11.008.

[8] E. R. Gansner, Y. Hu, and S. North. A maxent-stress model for graph layout. *IEEE transactions on visualization and computer graphics*, 19(6):927–940, 2012.

[9] E. R. Gansner, Y. Koren, and S. North. Graph drawing by stress majorization. In *Lecture Notes in Computer Science*, volume 3383, pages 239–250, 2004. doi:10.1007/978-3-540-31843-9_25.

[10] H. Gibson, J. Faith, and P. Vickers. A survey of two-dimensional graph layout techniques for information visualisation. *Information Visualization*, 12(3-4):324–357, 2012. doi:10.1177/1473871612455749.

[11] S. Hachul and M. Junger. Large-Graph Layout with the Fast Multipole Multilevel Method. *ACM Transactions on Graphics*, V(December):1–27, 2005. doi:10.7155/jgaa.00150.

[12] I. Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: a survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, 2000. doi:10.1109/2945.841119.

[13] Y. Hu and L. Shi. Visualizing large graphs. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7:115–136, 2015. doi:10.1002/wics.1343.

[14] M. Jacomy. *Situating Visual Network Analysis.* PhD thesis, 2021. PhD supervisor: Associate Prof. Anders Kristian Munk, Aalborg University Assistant PhD supervisor: Associate Prof. Anders Koed Madsen, Aalborg University.

[15] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian. ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software. *PLoS ONE*, 9(6):1–18, 2014. doi:10.1371/journal.pone.0098679.

[16] M. Khoury, Y. Hu, S. Krishnan, and C. Scheidegger. Drawing Large Graphs by Low-Rank Stress Majorization. *Computer Graphics Forum*, 31(3pt1):975–984, jun 2012. doi:10.1111/j.1467-8659.2012.03090.x.

[17] D. J. Klein and M. Randić. Resistance distance. *Journal of mathematical chemistry*, 12(1):81–95, 1993.

[18] J. F. Kruiger, P. E. Rauber, R. M. Martins, A. Kerren, S. Kobourov, and A. C. Telea. Graph layouts by t-sne. In *Computer Graphics Forum*, volume 36, pages 283–294. Wiley Online Library, 2017.

[19] C. McGrath, J. Blythe, and D. Krackhardt. Seeing groups in graph layouts. *Connections*, 19(2):22–29, 1996.

[20] A. K. Munk, A. K. Madsen, and M. Jacomy. Thinking through the databody: Sprints as experimental situations. In Å. Mäkitalo, T. E. Nicewonger, and M. Elam, editors, *Designs for Experimentation and Inquiry: Approaching Learning and Knowing in Digital Transformation*, chapter 6, pages 110–127. Routledge, 2019.

[21] T. Munzner. *Interactive Visualization of Large Graphs and Networks*. PhD thesis, 2000. URL: papers2://publication/uuid/D241CECC-6C6B-46F5-87CA-1412547FF972.

[22] T. Munzner. *Visualization Analysis and Design*. CRC Press, 2014. doi:10.1201/b17511.

[23] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of ...*, 103(23):8577–8582, 2006. arXiv:0602124, doi:10.1073/pnas.0601602103.

[24] B. Nick, C. Lee, P. Cunningham, and U. Brandes. Simmelian backbones: Amplifying hidden homophily in Facebook networks. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2013*, pages 525–532, 2013. doi:10.1145/2492517.2492569.

[25] A. Noack. Energy models for graph clustering. *Journal of Graph Algorithms and Applications*, 11(2):453–480, 2007. doi:10.7155/jgaa.00154.

[26] A. Noack. Modularity clustering is force-directed layout. *Physical Review E*, 79(2):026102, 2009.

[27] A. Nocaj, M. Ortmann, and U. Brandes. Untangling the Hairballs of Multi-Centered, Small-World Online Social Media Networks. *Journal of Graph Algorithms and Applications*, 19(2):595–618, 2015. URL: http://jgaa.info/getPaper?id=370, doi:10.7155/jgaa.00370.

[28] M. Okoe, R. Jianu, and S. Kobourov. Node-link or adjacency matrices: Old question, New Insights. *IEEE Transactions on Visualization and Computer Graphics*, 25(10):2940–2952, 2019. doi:10.1109/TVCG.2018.2865940.

[29] T. P. Peixoto. Revealing consensus and dissensus between network partitions. *Physical Review X*, 11(2):021003, 2021.

[30] H. Purchase. Which aesthetic has the greatest effect on human understanding? In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 1353, pages 248–261. Springer Verlag, 1997. doi:10.1007/3-540-63938-1_67.

[31] H. C. Purchase. Metrics for graph drawing aesthetics. *Journal of Visual Languages and Computing*, 13(5):501–516, 2002. doi:10.1016/S1045-926X(02)90232-6.

[32] H. C. Purchase, R. F. Cohen, and M. James. Validating Graph Drawing Aesthetics. *Graph Drawing – Symposium on Graph Drawing, GD '95 Passau, Germany, September 20–22, 1995 Proceedings*, 1027:435–446, 1995. doi:10.1007/BFb0021827.

[33] M. T. Schaub, J. C. Delvenne, M. Rosvall, and R. Lambiotte. The many facets of community detection in complex networks. *Applied Network Science*, 2(1):4, 2017. arXiv:1611.07769, doi:10.1007/s41109-017-0023-6.

[34] U. Soni, Y. Lu, B. Hansen, H. C. Purchase, S. Kobourov, and R. Maciejewski. The Perception of Graph Properties in Graph Layouts. *Computer Graphics Forum*, 37(3):169–181, jun 2018. doi:10.1111/cgf.13410.

[35] F. Van Ham and B. Rogowitz. Perceptual organization in user-generated graph layouts. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1333–1339, 2008.

[36] J. Venna and S. Kaski. Neighborhood preservation in nonlinear projection methods: An experimental study. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2130:485–491, 2001. doi:10.1007/3-540-44668-0_68.

[37] J. Venna and S. Kaski. Local multidimensional scaling with controlled tradeoff between trustworthiness and continuity. In *WSOM 2005 - 5th Workshop on Self-Organizing Maps*, pages 695–702, 2005.

[38] T. Venturini, M. Jacomy, and P. Jensen. What do we see when we look at networks: Visual network analysis, relational ambiguity, and force-directed layouts. *Big Data & Society*, 8(1):20539517211018488, 2021.

[39] T. Von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. J. van Wijk, J.-D. Fekete, and D. W. Fellner. Visual analysis of large graphs: state-of-the-art and future research challenges. *Computer graphics forum*, 30(6):1719–1749, 2011. doi:10.1111/j.1467-8659.2011.01898.x.

[40] C. Ware, H. Purchase, L. Colpoys, and M. McGill. Cognitive measurements of graph aesthetics. *Information visualization*, 1(2):103–110, 2002.

[41] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

A Appendix: Benchmark (details)

This appendix presents a benchmark on connected-closeness. I used 14 network generators (e.g., stochastic block model with different settings) and 7 node-placement algorithms available in Javascript. The choice of Javascript is justified by its importance for interactive data visualization. One of the algorithms used is the default choice offered by the *D3.js* library. For each *(generator, layout)* pair I generated and visualized 100 networks of 100 nodes, and computed the following indicators: Δ_{max}, C_{max}, $E_{\%}(\Delta_{max})$, $p_{\%}(\Delta_{max})$, and $P_{edge}(\Delta_{max})$ (as previously defined in the paper). The benchmark is available as an online notebook[2] as well as its analysis including the data[3]. First I will provide visual examples and second I will present the statistical results of each indicator one after the other.

The data of the benchmark are presented in extenso in appendix A.7.

Network generators. 14 different strategies were used to generate networks. Note that some of them generate a different network every time (e.g., the random-network generator) and others do not (e.g., the clique generator). The stochastic-block-model generator is set to a planted-partitions model [4].

- **Clique.** All nodes are connected.
- **Two bridged cliques.** 2 cliques (groups of fully connected nodes) connected by 1 edge.
- **Two blocks connected at 99.9%.** Network generated by stochastic block model. It has 2 blocks. Two nodes in the same block have a 99.9% probability of being connected. Two nodes in a different block have a 0.1% probability of being connected.
- **Two blocks connected at 99%.** Network generated by stochastic block model. It has 2 blocks. Two nodes in the same block have a 99% probability of being connected. Two nodes in a different block have a 1% probability of being connected.
- **Two blocks connected at 90%.** Network generated by stochastic block model. It has 2 blocks. Two nodes in the same block have a 90% probability of being connected. Two nodes in a different block have a 10% probability of being connected.
- **Two blocks connected at 80%.** Network generated by stochastic block model. It has 2 blocks. Two nodes in the same block have a 80% probability of being connected. Two nodes in a different block have a 20% probability of being connected.
- **Two blocks connected at 70%.** Network generated by stochastic block model. It has 2 blocks. Two nodes in the same block have a 70% probability of being connected. Two nodes in a different block have a 30% probability of being connected.
- **Two blocks connected at 60%.** Network generated by stochastic block model. It has 2 blocks. Two nodes in the same block have a 60% probability of being connected. Two nodes in a different block have a 40% probability of being connected.
- **Random network connected at 50%.** Random network: two nodes have a 50% probability of being connected.

[2]https://observablehq.com/@jacomyma/evaluating-the-connected-closeness-metric-post-design
[3]https://observablehq.com/@jacomyma/highlights-on-connected-closeness

- **Random network connected at 5%.** Random network: two nodes have a 5% probability of being connected.
- **Random network connected at 0.1%.** Random network: two nodes have a 0.1% probability of being connected.
- **Chain.** A chain of nodes.
- **Star.** Network with a central node, and the other nodes are only connected to that one.
- **Square lattice.** This network is a square grid of nodes.

Layouts. 7 different algorithms were used to generate networks.

- **Force Atlas 2.** The Force Atlas 2 layout with standard settings [15].
- **Force Atlas 2 BAD.** The Force Atlas 2 layout with bad settings: too much gravity, not enough iterations.
- **Lin Log.** The Force Atlas 2 layout with the *LinLog* energy model. [25]
- **Random (in a disc).** Nodes are placed at random in a disc.
- **Random (Graphology).** Nodes are placed at random in a square, using the method from the Graphology library.
- **Circular layout.** Nodes are placed around a circle. Uses the method from the Graphology library. Note: this algorithm uses the order of nodes. For most generators (bridged cliques, block models, chain, star and lattice) the nodes are ordered in a relevant way, which produces remarkable patterns. For the random network generators, the node order is irrelevant and the circular layout will behave similarly to a random layout.
- **D3 Force Simulation.** The force-directed layout algorithm implemented in the library *D3.js*, with default settings.

A.1 Visual examples

You will find here one rendition for each $(networkgenerator, layout)$ pair. For each of these 14×7 cases, that same network is visualized and accompanied by a plot. The network visualization has colored edges if $C_{max} > 10\%$, with edges shorter than Δ_{max} in green, and longer edges in red. The plot features the main indicators over the Euclidean distance Δ. The share of edges shorter than Δ, i.e. $E_{\%}(\Delta)$, is plotted in black. The share of node pairs closer than Δ, i.e. $p_{\%}(\Delta)$, is plotted in blue. The connected-closeness $C(\Delta)$ is plotted in green. Its maximum point $C_{max} = C(\Delta_{max})$ is highlighted by a vertical green bar.

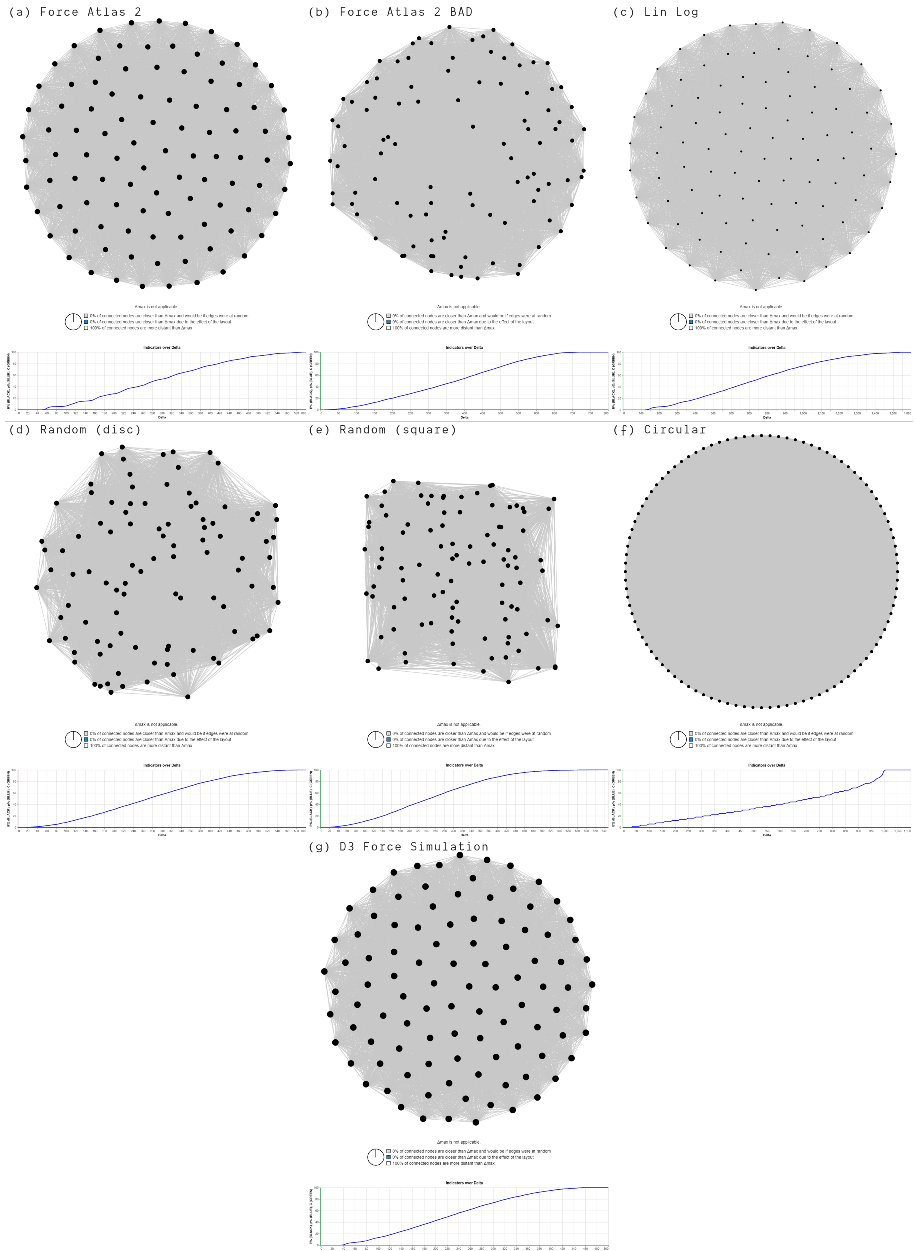


Figure 18: Clique

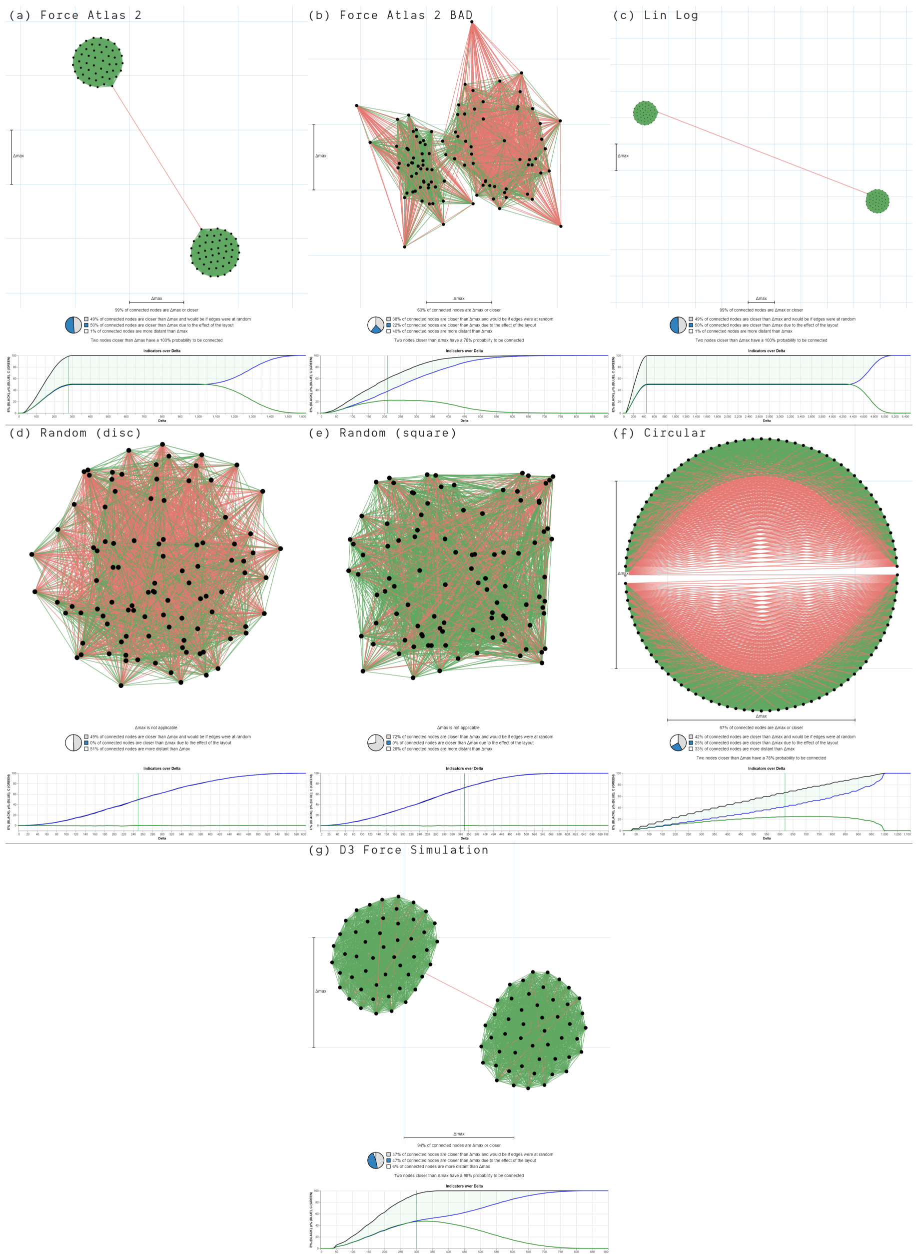


Figure 19: Two cliques connected by one bridge

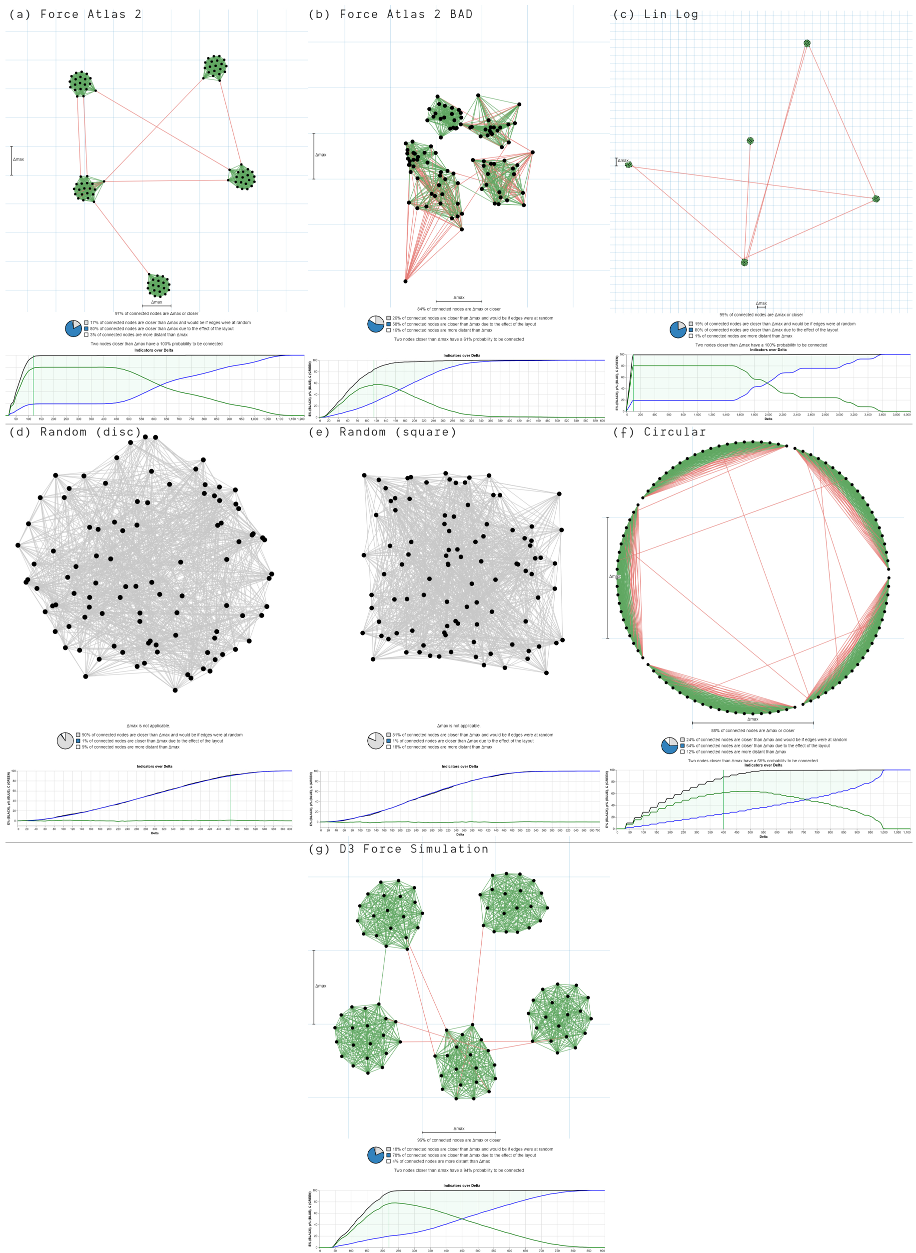


Figure 20: Five blocks internally connected with a probability of 99.9% (and 0.1% in-between)

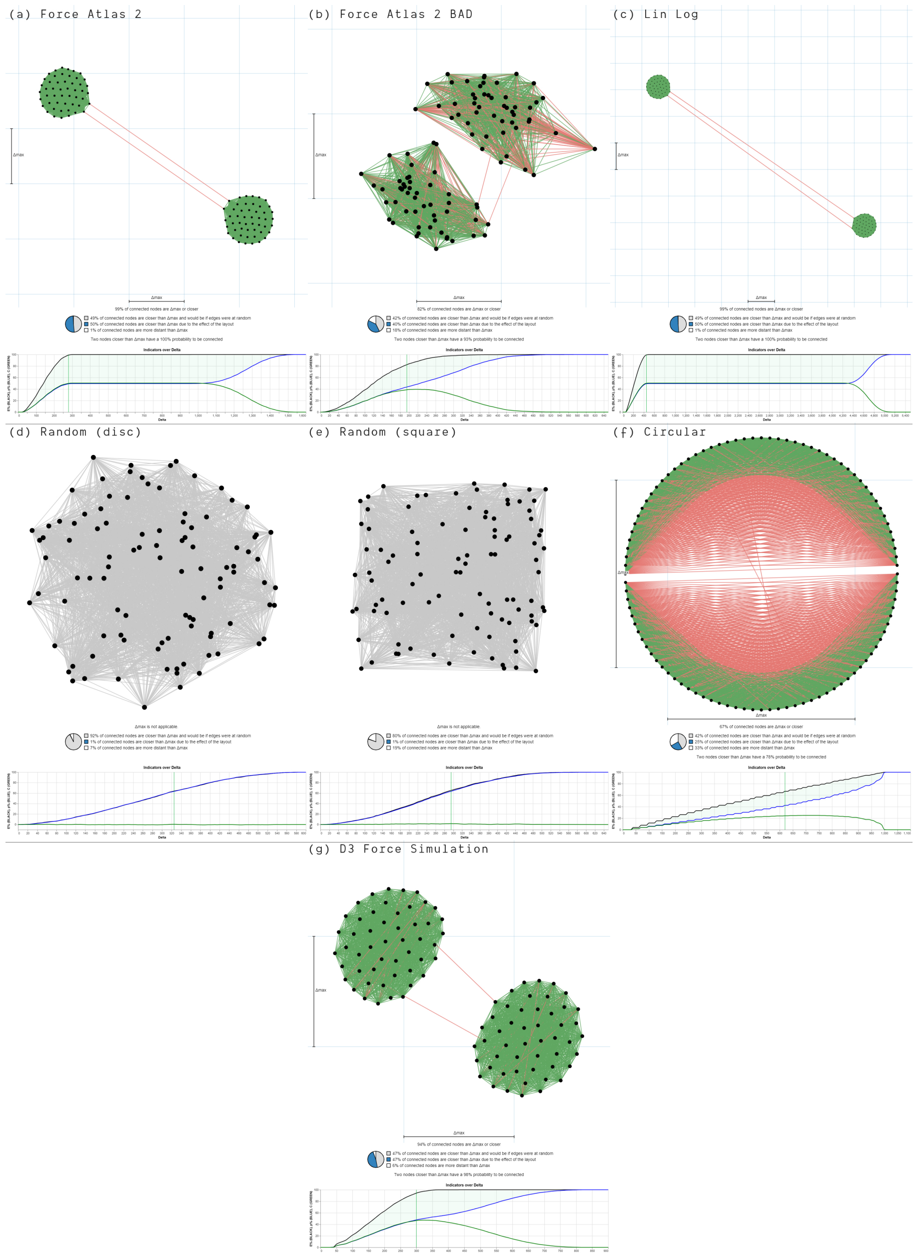


Figure 21: Two blocks internally connected with a probability of 99.9% (and 0.1% in-between)

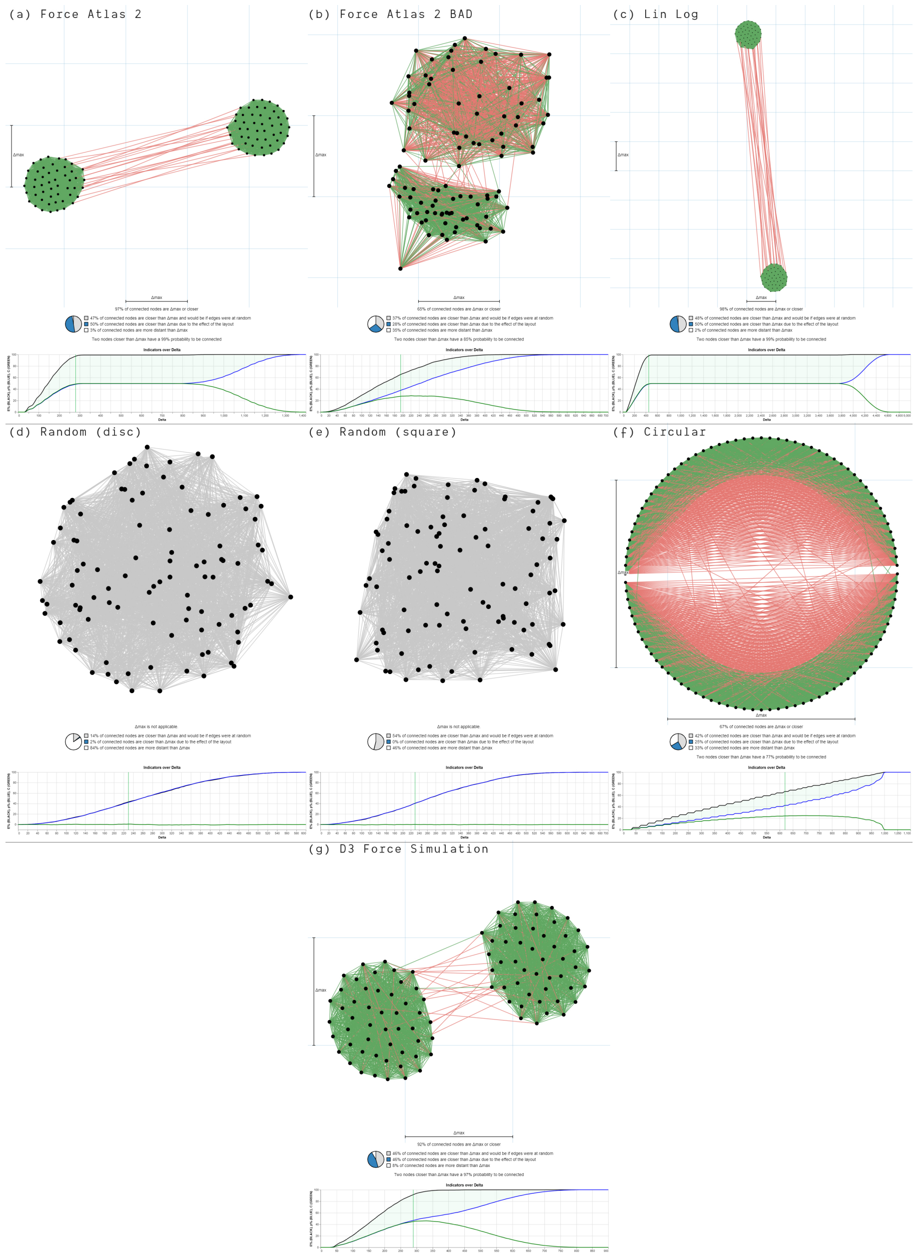


Figure 22: Two blocks internally connected with a probability of 99% (and 1% in-between)

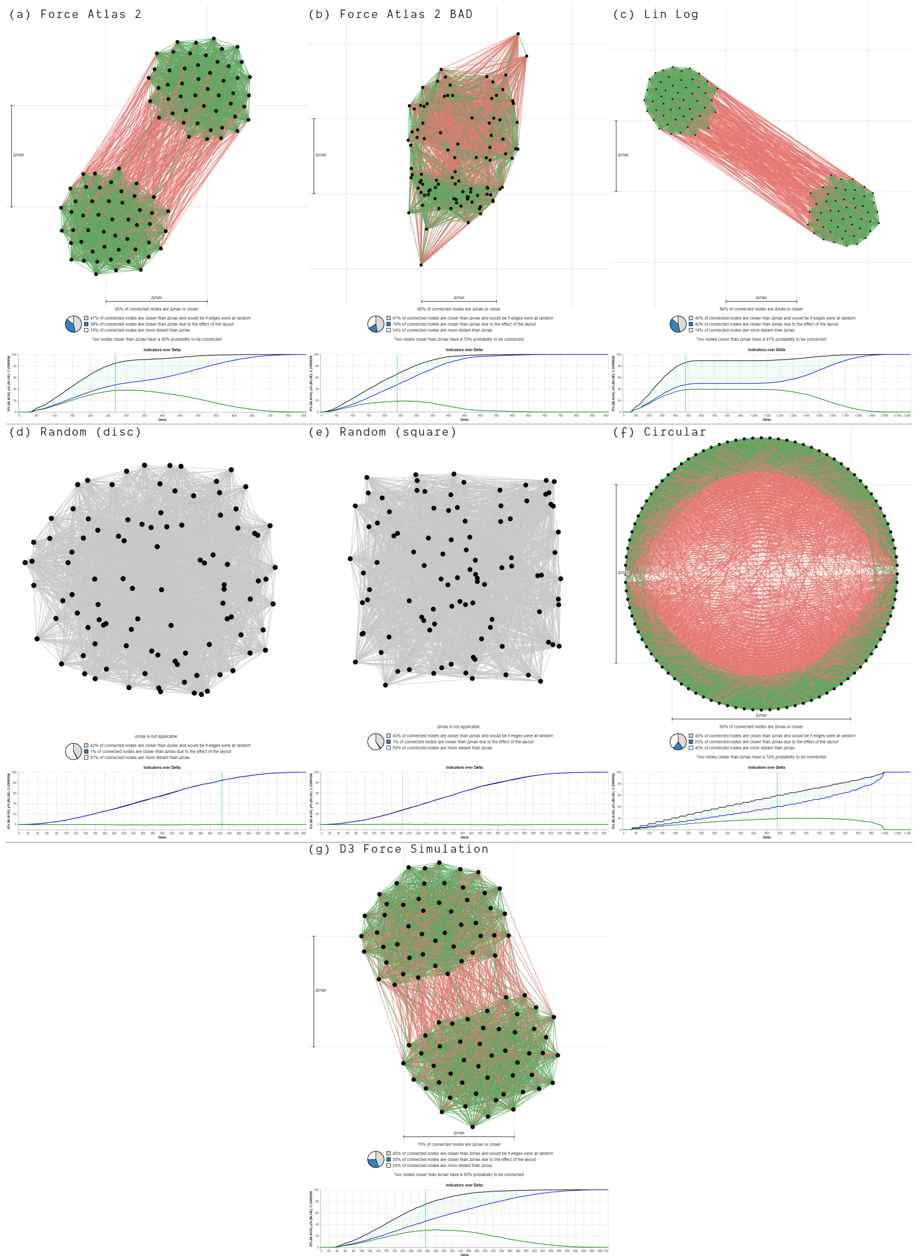


Figure 23: Two blocks internally connected with a probability of 90% (and 10% in-between)

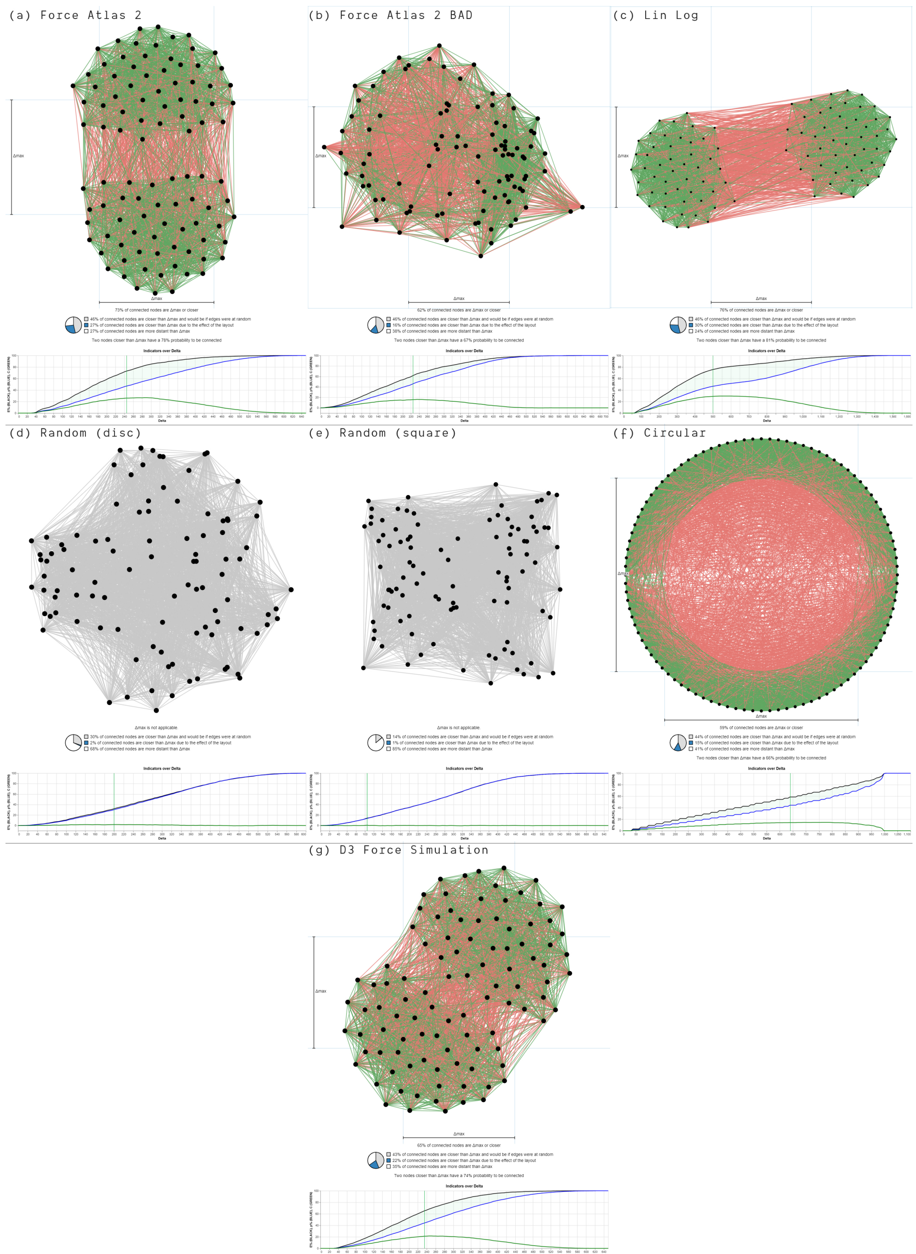


Figure 24: Two blocks internally connected with a probability of 80% (and 20% in-between)

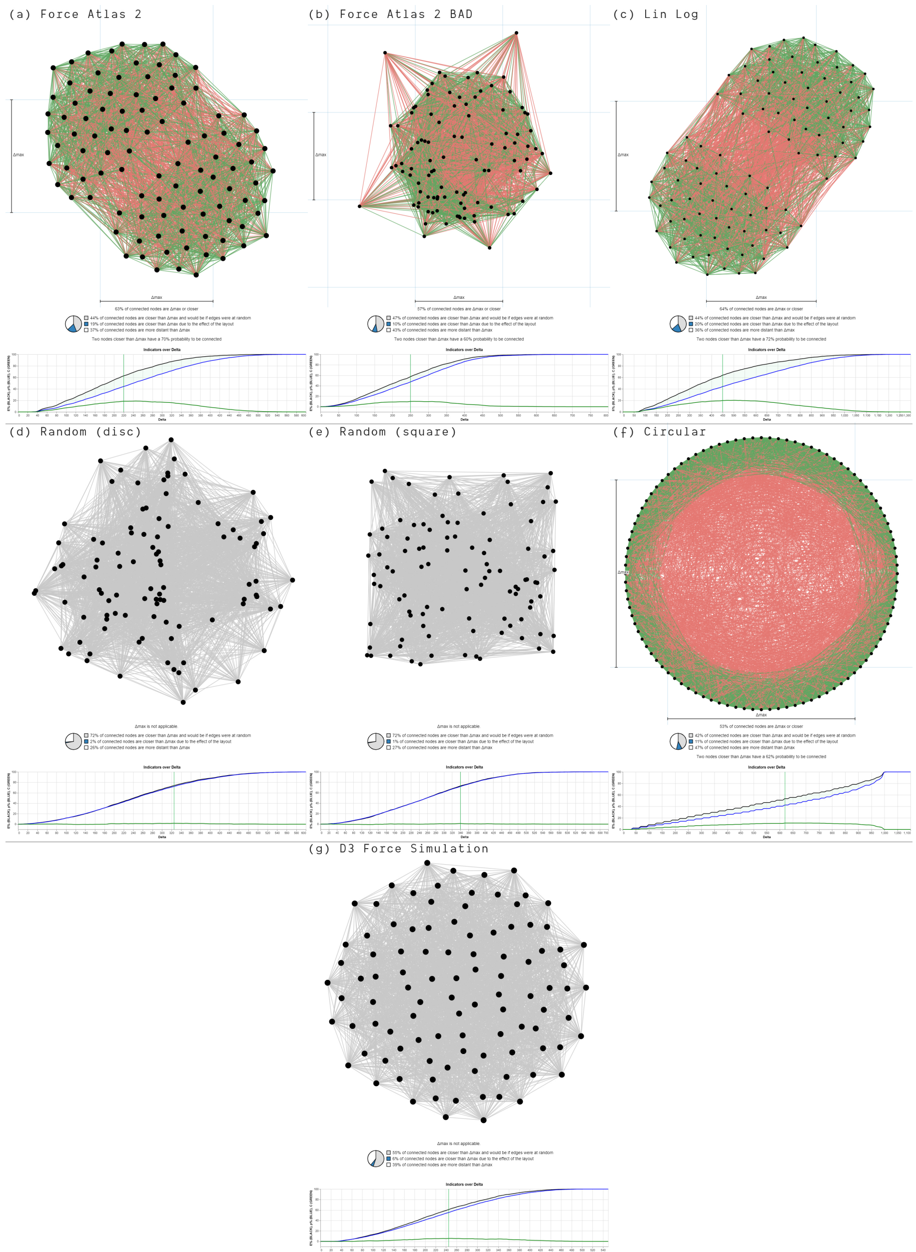


Figure 25: Two blocks internally connected with a probability of 70% (and 30% in-between)

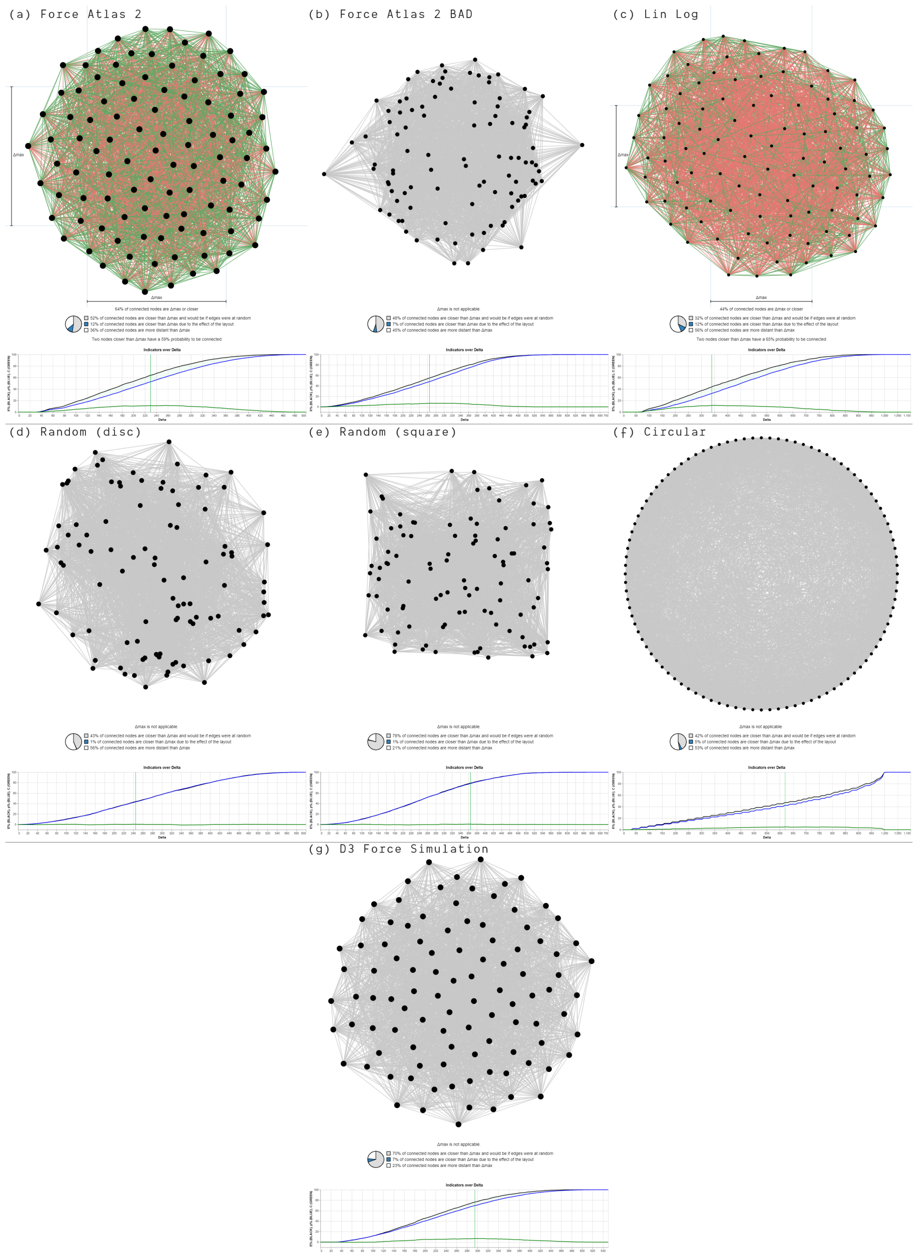


Figure 26: Two blocks internally connected with a probability of 60% (and 40% in-between)

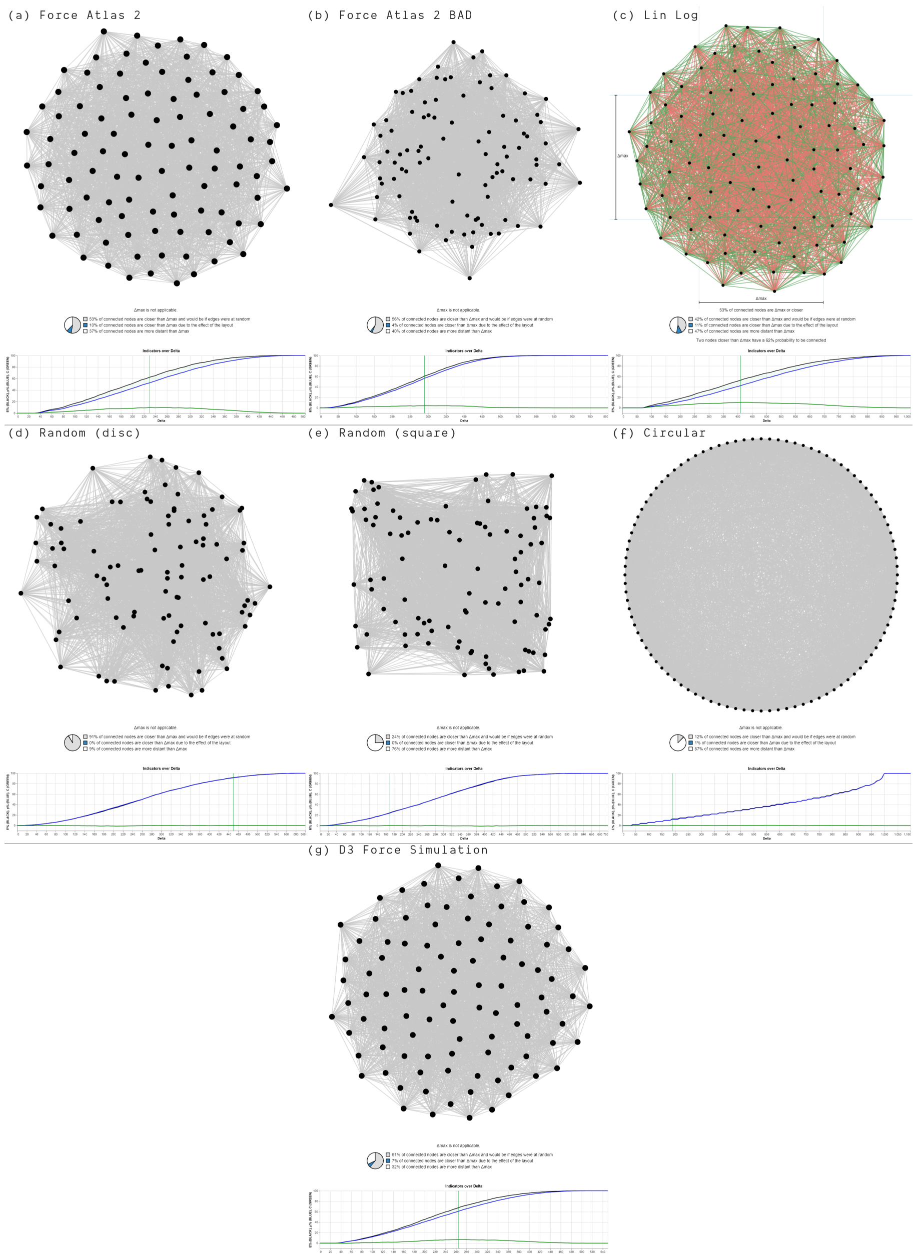


Figure 27: Random network with a connection probability of 50%

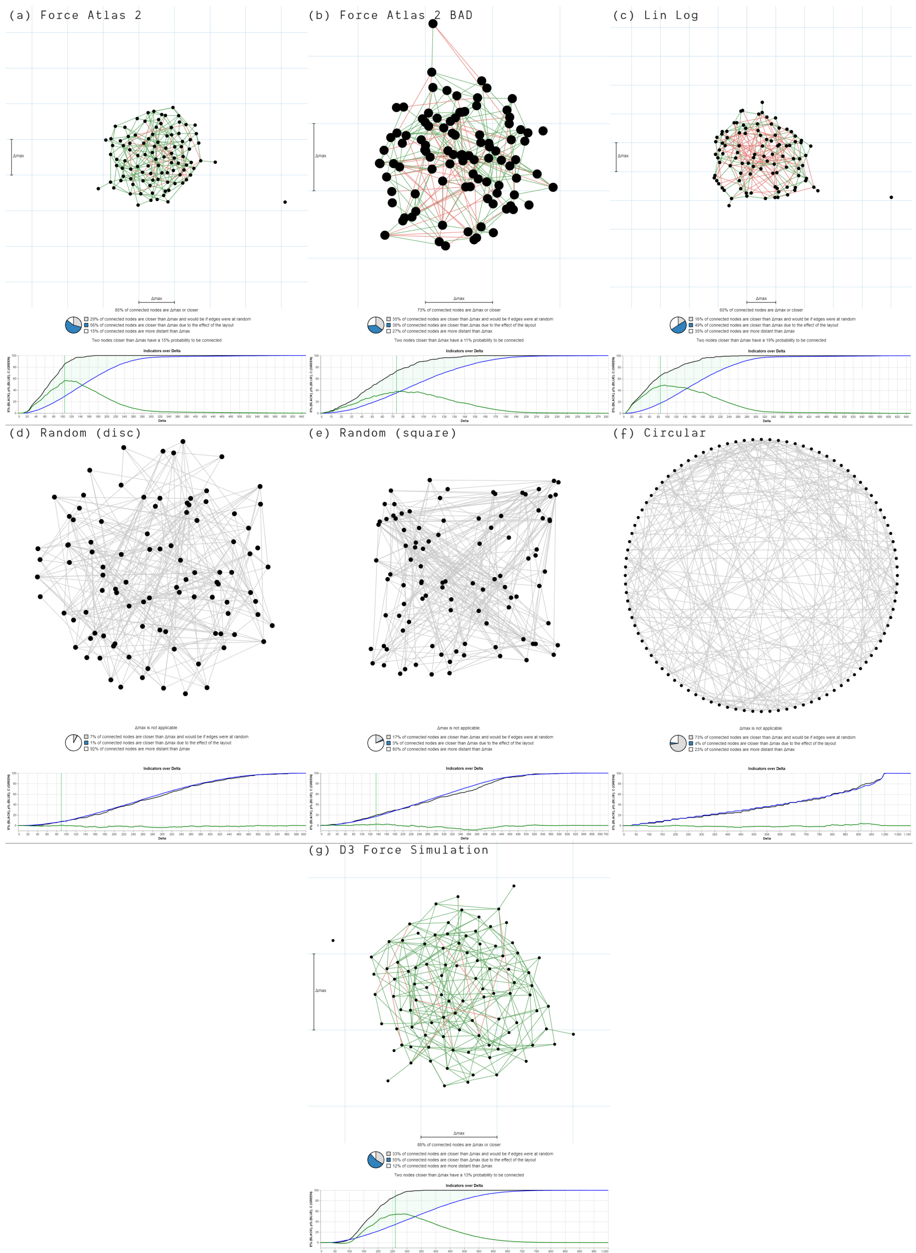


Figure 28: Random network with a connection probability of 5%

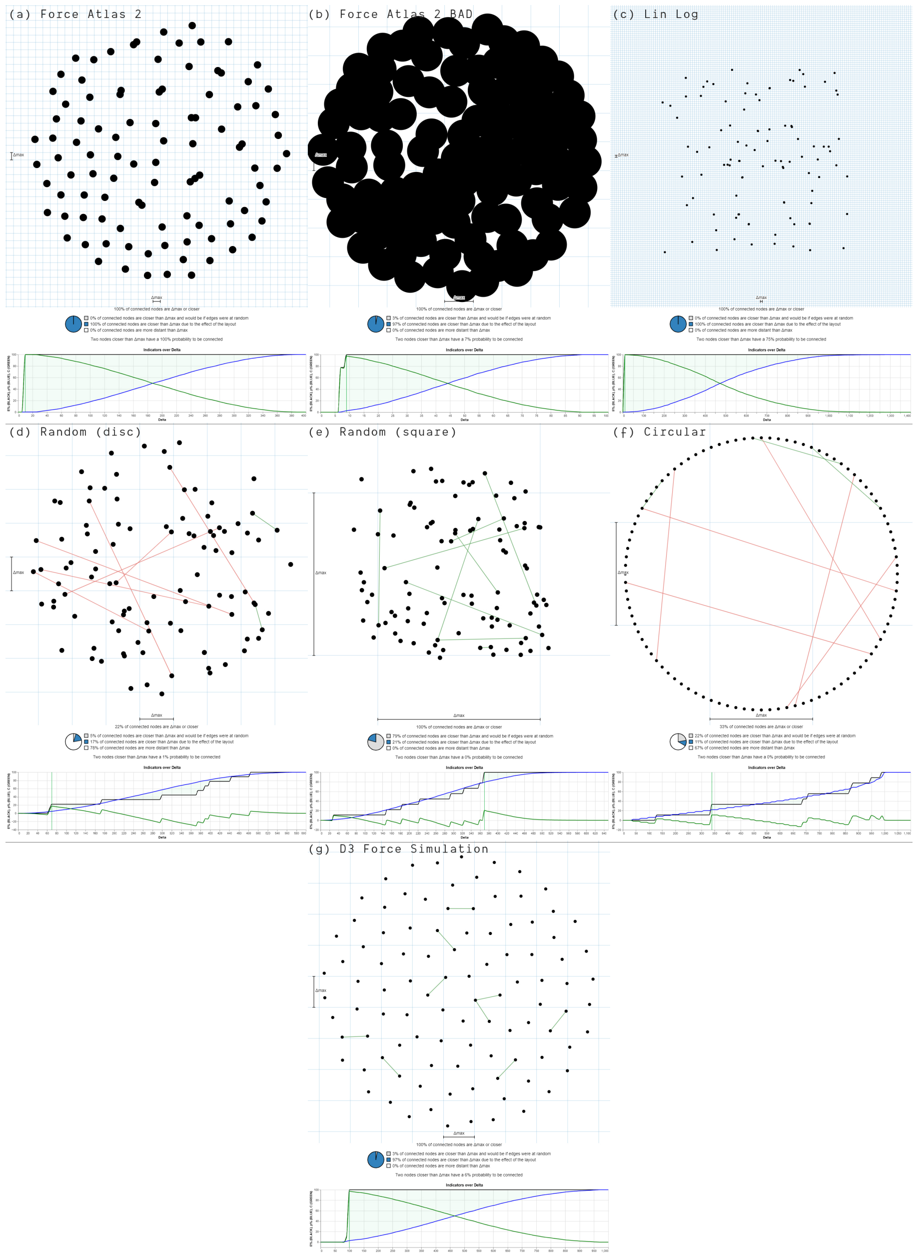


Figure 29: Random network with a connection probability of 0.1%

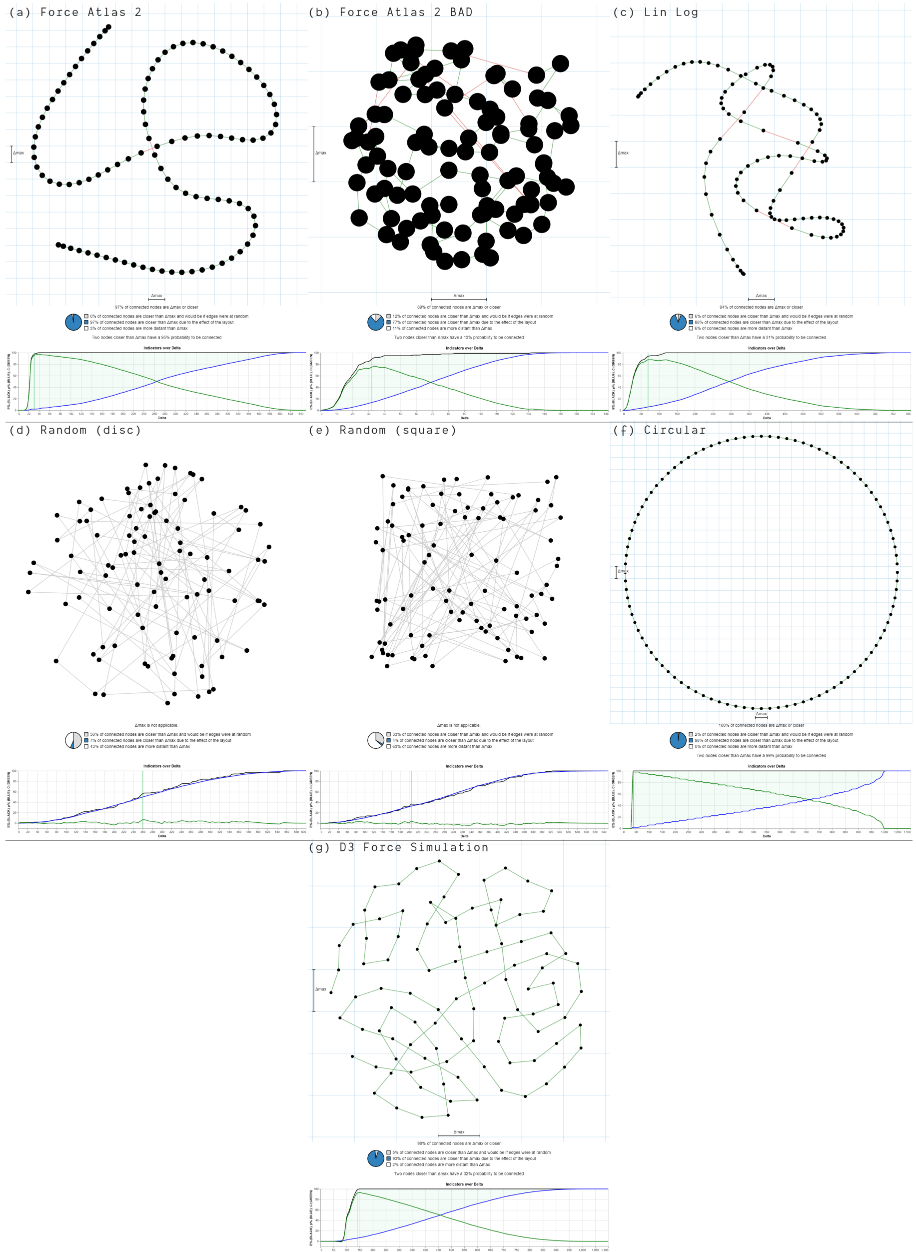


Figure 30: Chain

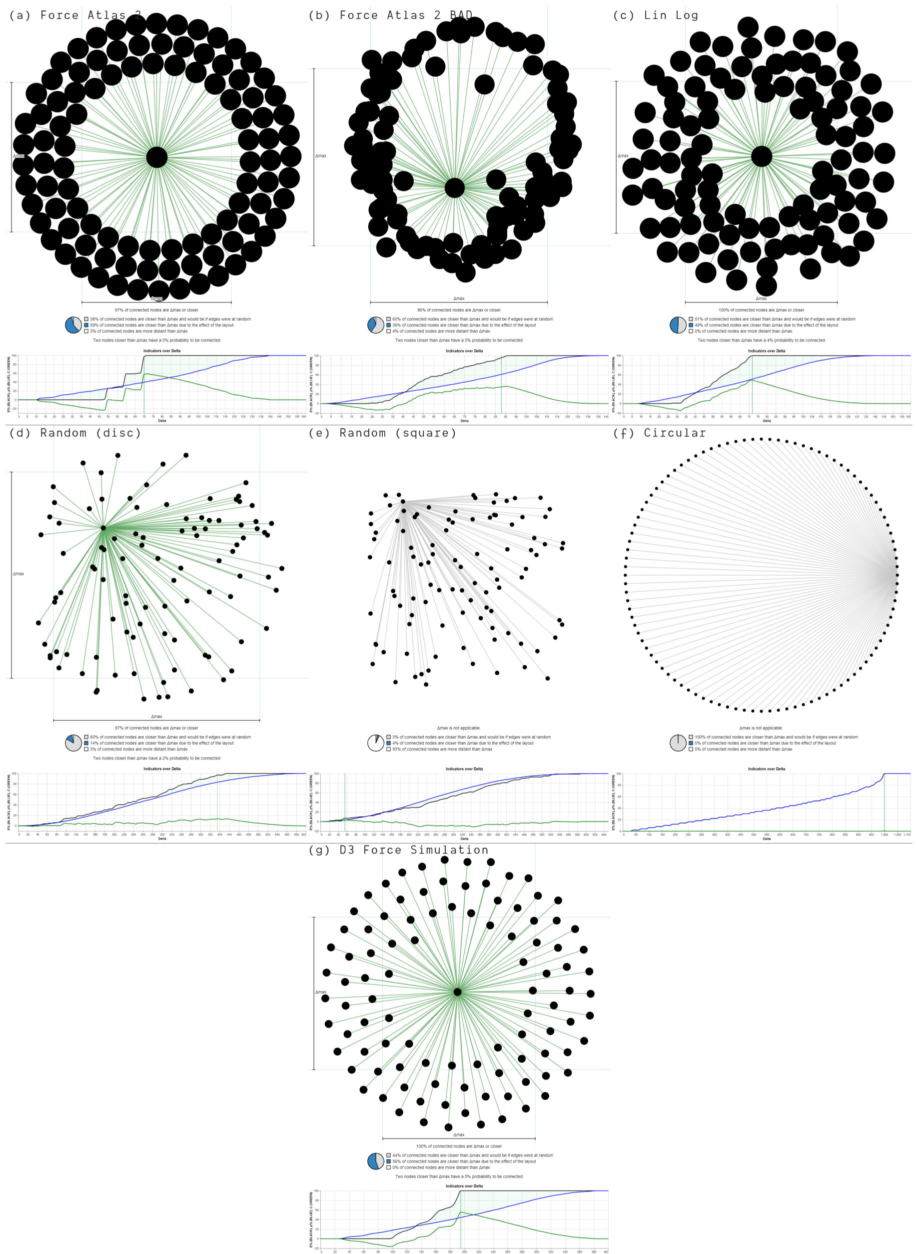


Figure 31: Star

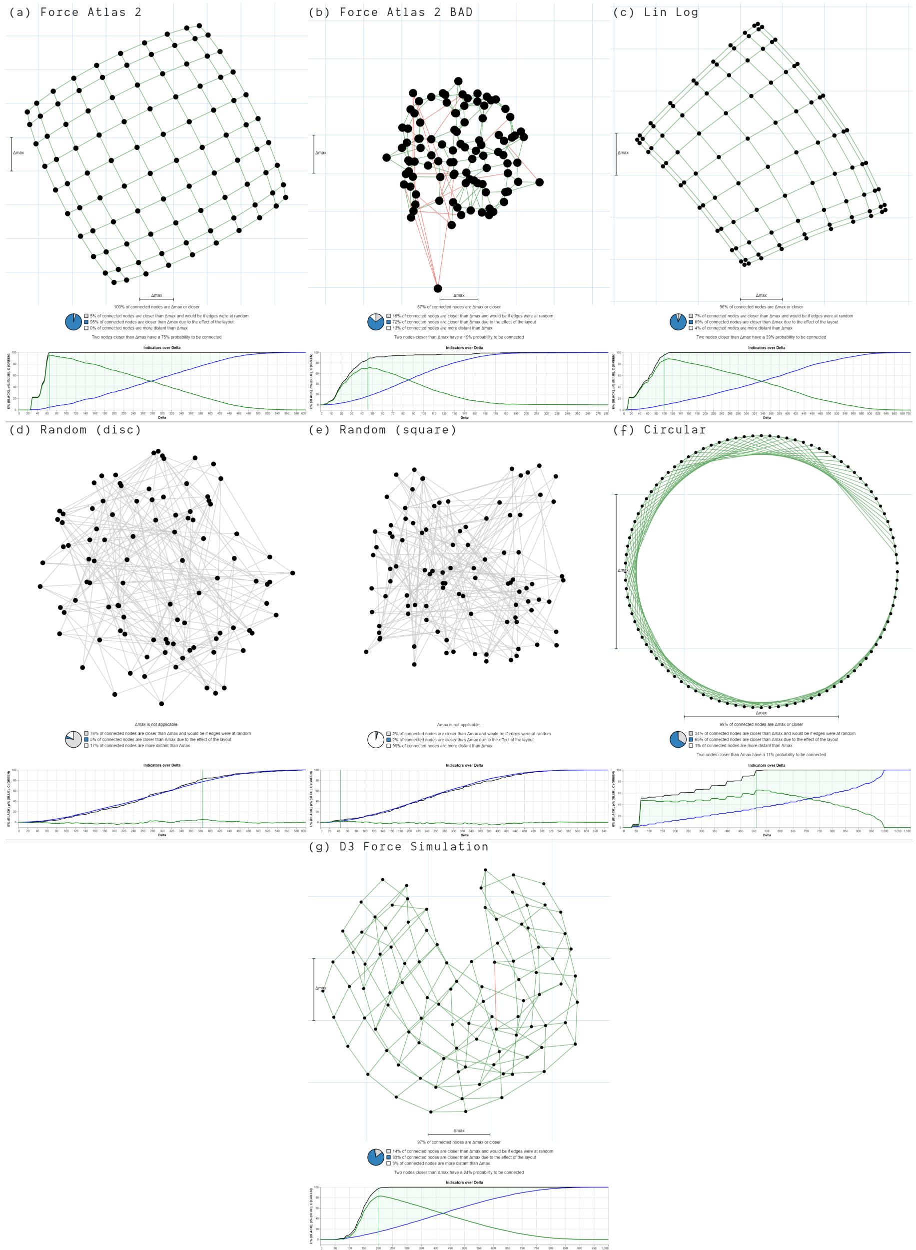
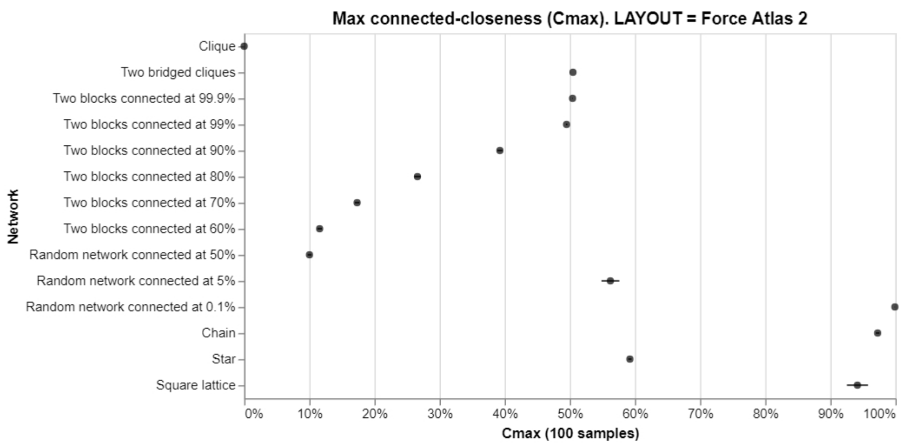


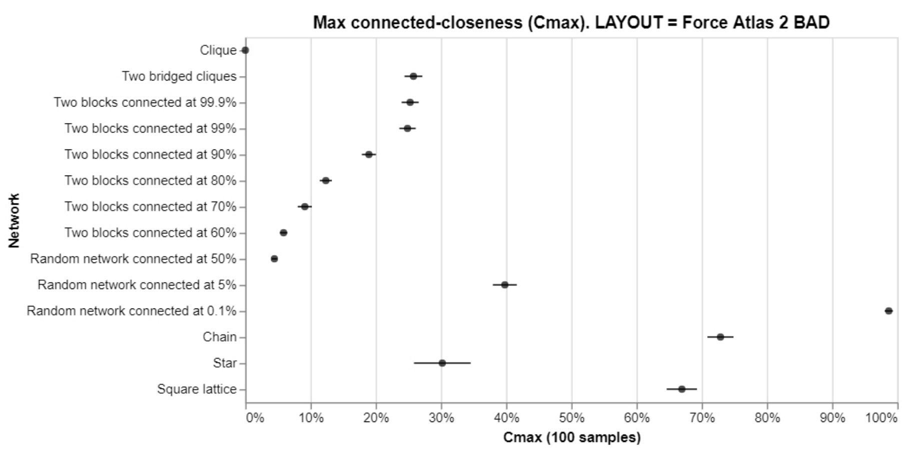
Figure 32: Square lattice

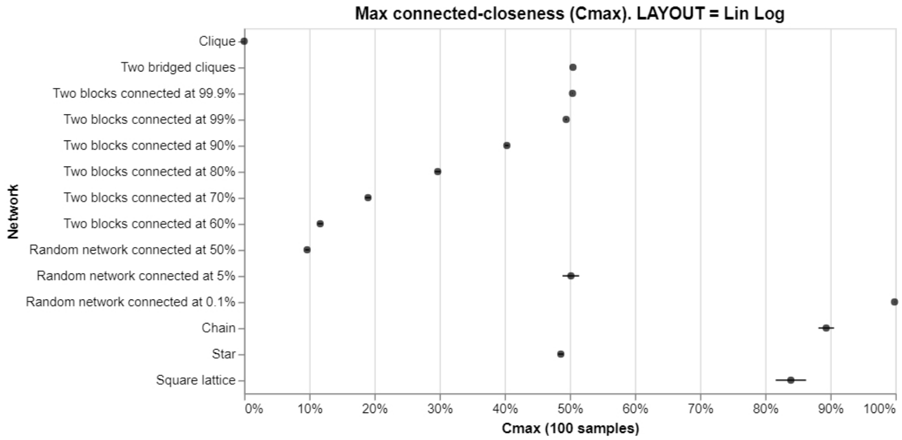
A.2 Maximum connected-closeness $C_{max} = C(\Delta_{max})$

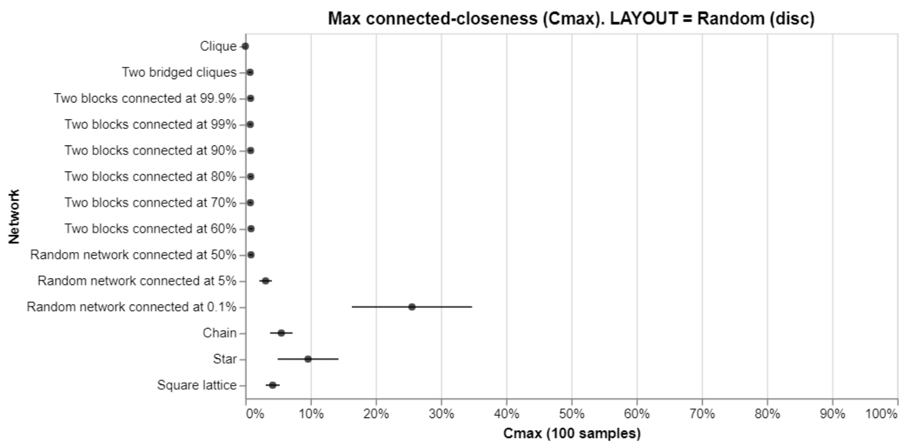
C_{max} is the maximum percentage of unexpectedly-close connected nodes. For a given network, it allows comparing different layouts; and for a layout, it allows comparing different networks. Higher is better, as it means that the layout captures more unexpected edges under distance Δ_{max}. As we have seen, "unexpected" means here "discounting what we would observe on average with edges redistributed at random".

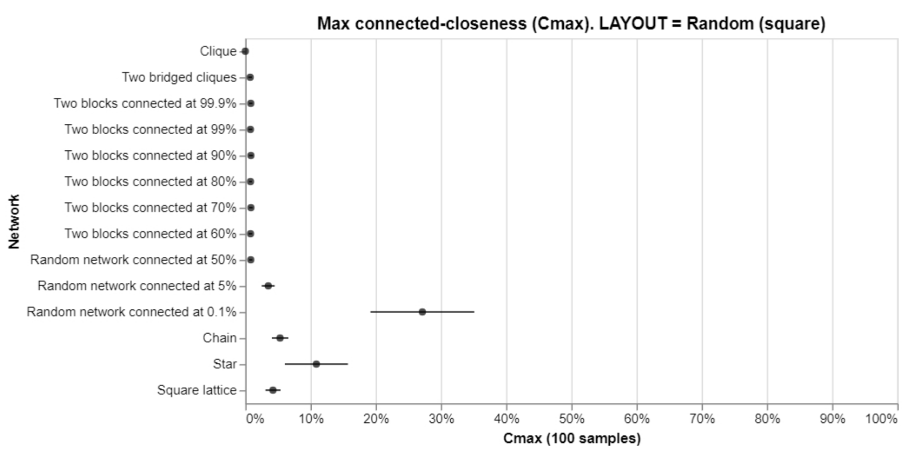
For each *(network generator, layout)* pair, the values for the 100 renditions are represented as a dot (mean) with an error bar (standard deviation). Noteworthy: the standard deviation may be so low that the bar is not even visible (which is remarkable).

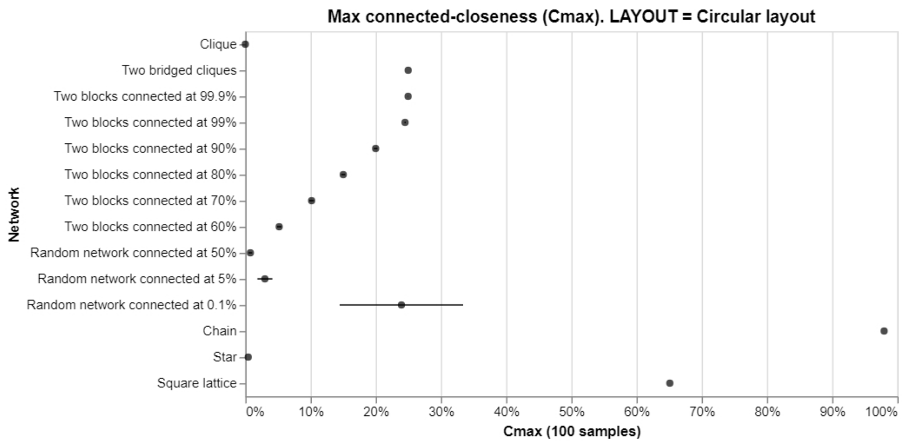




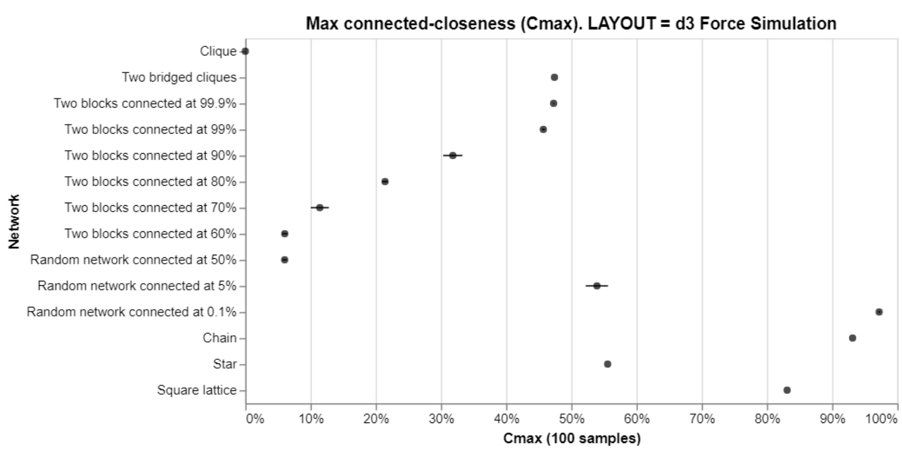

Max connected-closeness (Cmax). LAYOUT = Lin Log
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
Cmax (100 samples)


Max connected-closeness (Cmax). LAYOUT = Random (disc)
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
Cmax (100 samples)


Max connected-closeness (Cmax). LAYOUT = Random (square)
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
Cmax (100 samples)



Max connected-closeness (Cmax). LAYOUT = Circular layout
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
Cmax (100 samples)

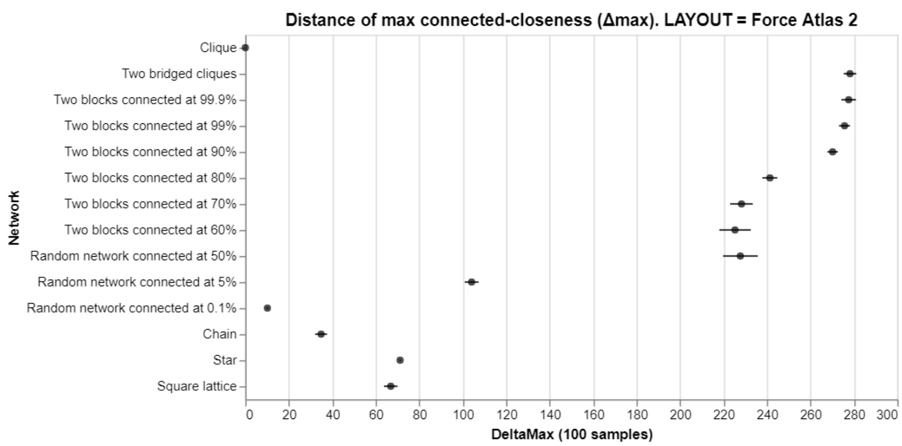


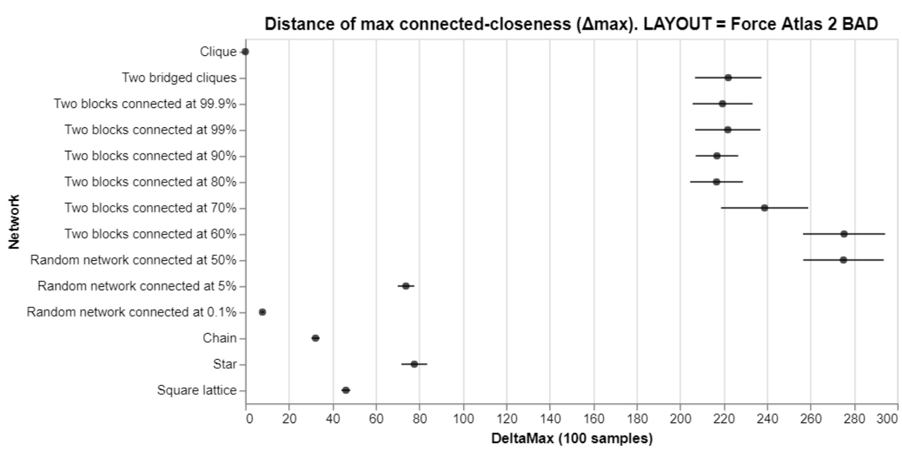
Max connected-closeness (Cmax). LAYOUT = d3 Force Simulation
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
Cmax (100 samples)

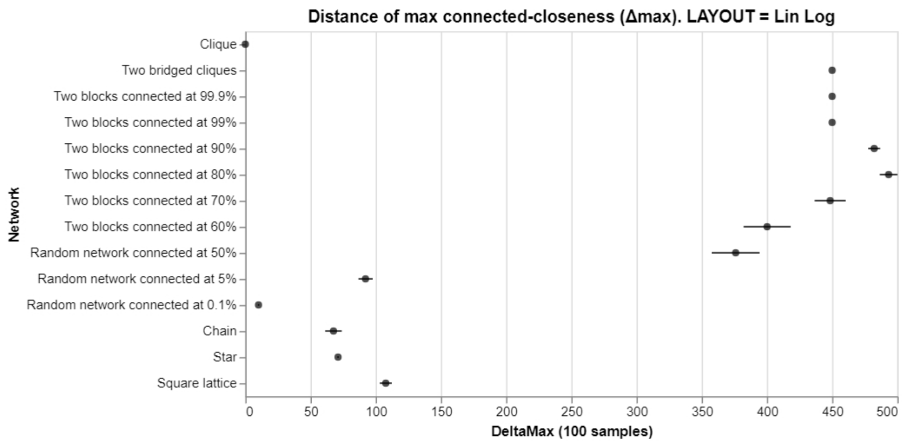
A.3 Distance of maximum connected-closeness Δ_{max}

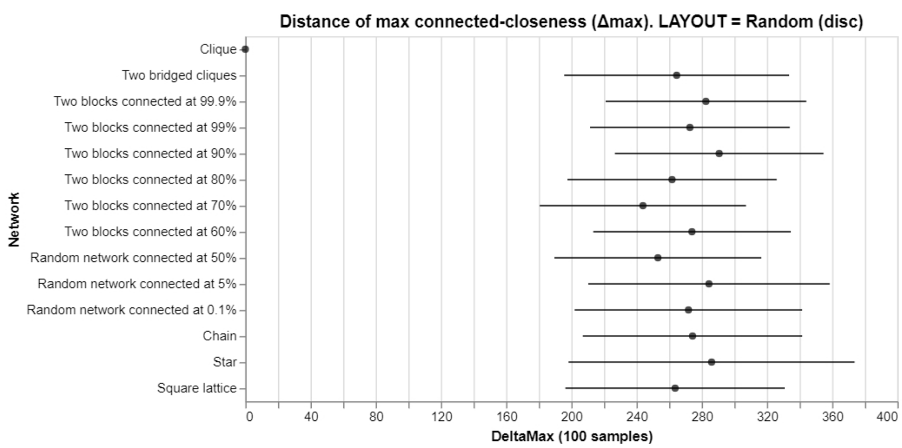
Δ_{max} is the Euclidean distance defining the maximum percentage of unexpectedly-close connected nodes. Its unit is the arbitrary unit generated by the node coordinates provided by the layout (it is not normalized). For a given layout, it allows comparing different networks. It does not compare from one layout to another. Lower is better, but only insofar as $C(\Delta_{max})$ is high. Δ_{max} by itself does not tell much about a network map, as its purpose is to be drawn on the visualization to provide context. It is however useful to look at how if behaves in different situations, and notably its consistency (standard deviation).

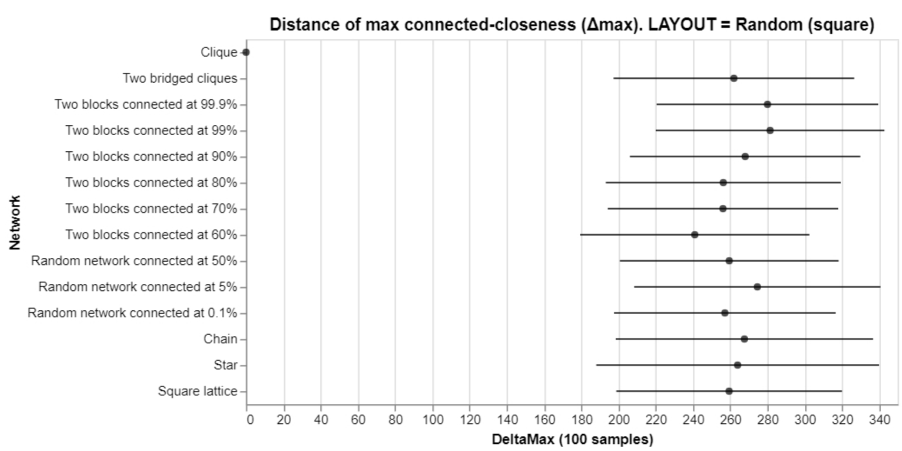
For each *(network generator, layout)* pair, the values for the 100 renditions are represented as a dot (mean) with an error bar (standard deviation). Noteworthy: the standard deviation may be so low that the bar is not even visible (which is remarkable).

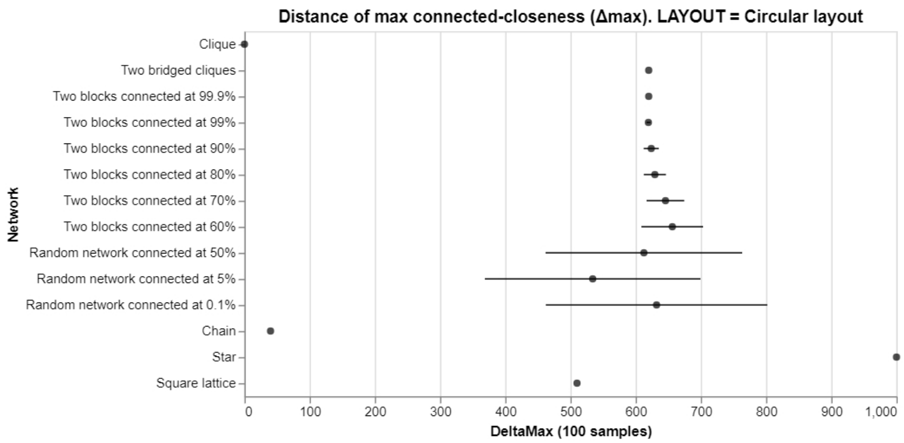


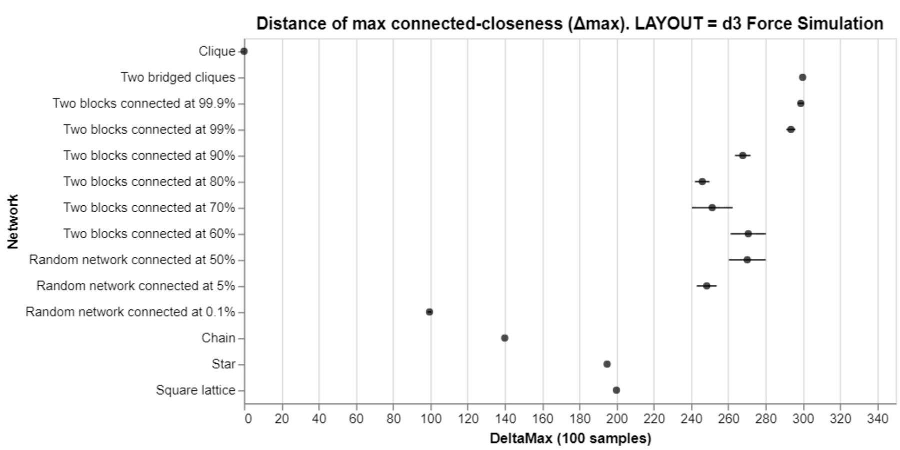



Distance of max connected-closeness (Δmax). LAYOUT = Lin Log
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0 50 100 150 200 250 300 350 400 450 500
DeltaMax (100 samples)


Distance of max connected-closeness (Δmax). LAYOUT = Random (disc)
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0 40 80 120 160 200 240 280 320 360 400
DeltaMax (100 samples)


Distance of max connected-closeness (Δmax). LAYOUT = Random (square)
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0 20 40 60 80 100 120 140 160 180 200 220 240 260 280 300 320 340
DeltaMax (100 samples)


Distance of max connected-closeness (Δmax). LAYOUT = Circular layout
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0 100 200 300 400 500 600 700 800 900 1,000
DeltaMax (100 samples)

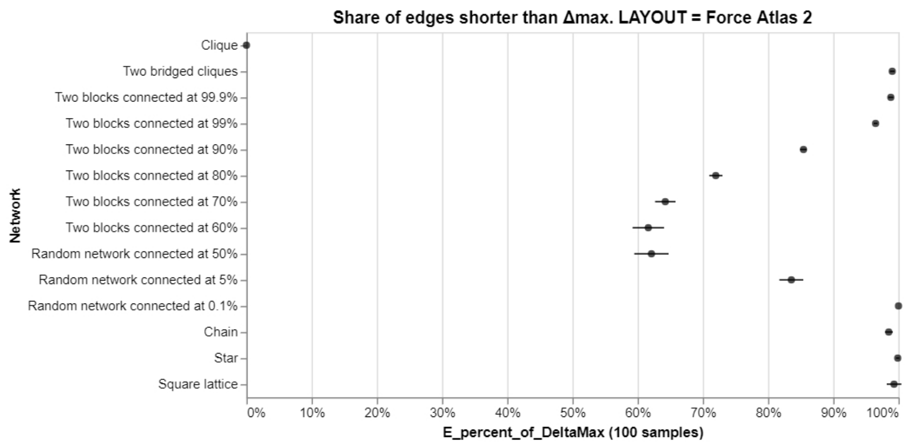

Distance of max connected-closeness (Δmax). LAYOUT = d3 Force Simulation
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0 20 40 60 80 100 120 140 160 180 200 220 240 260 280 300 320 340
DeltaMax (100 samples)

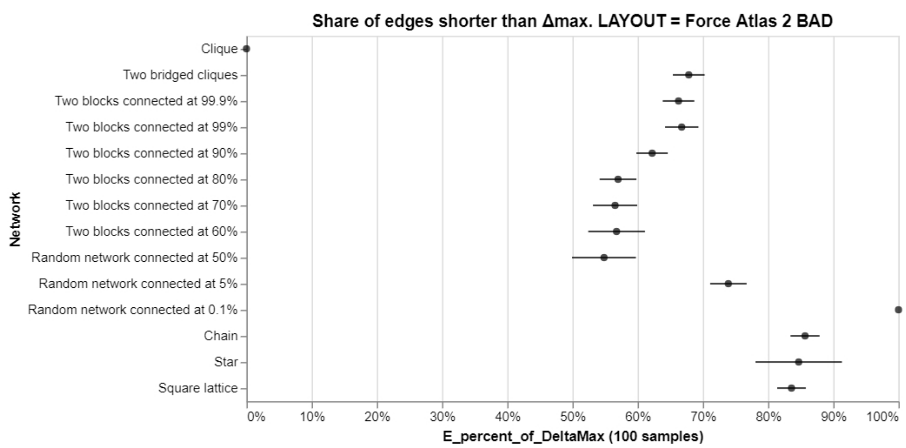
A.4 $E_{\%}(\Delta_{max})$, the share of edges shorter than Δ_{max} (*recall*).

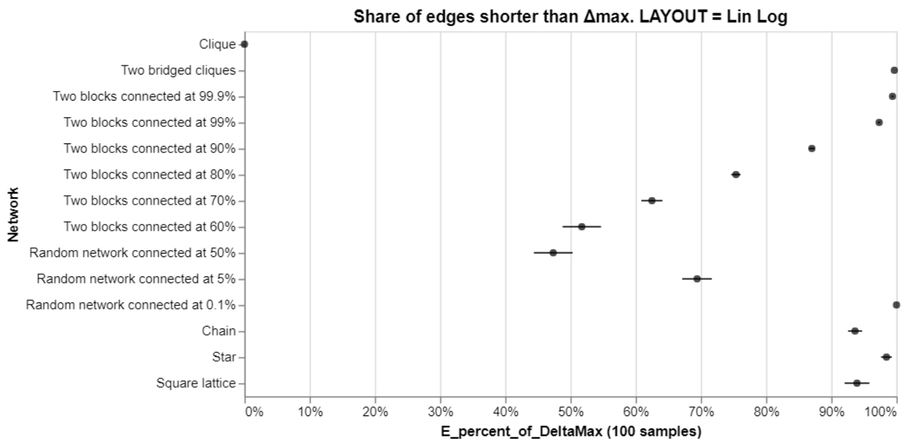
$E_{\%}(\Delta_{max})$ is a percentage of edges. For a given network, it allows comparing different layouts, and for a layout, it allows comparing different networks. It also compares to $p_{\%}(\Delta_{max})$.

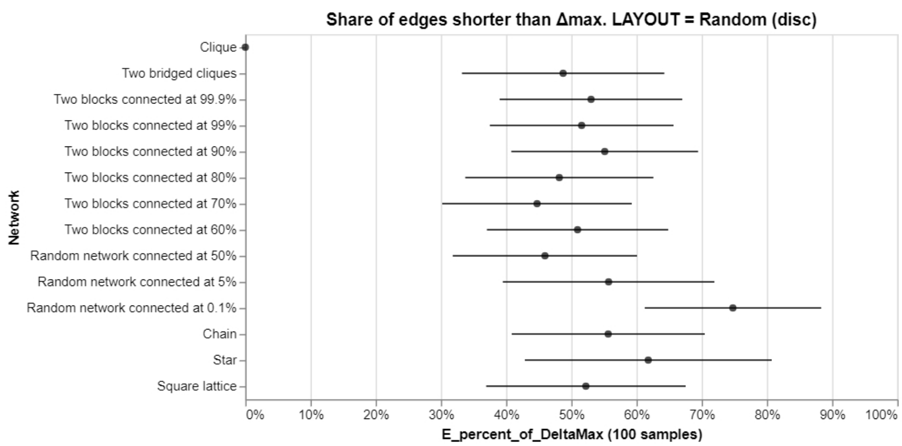
$E_{\%}(\Delta_{max})$ is also the recall of measuring connectedness with the distance Δ_{max}.

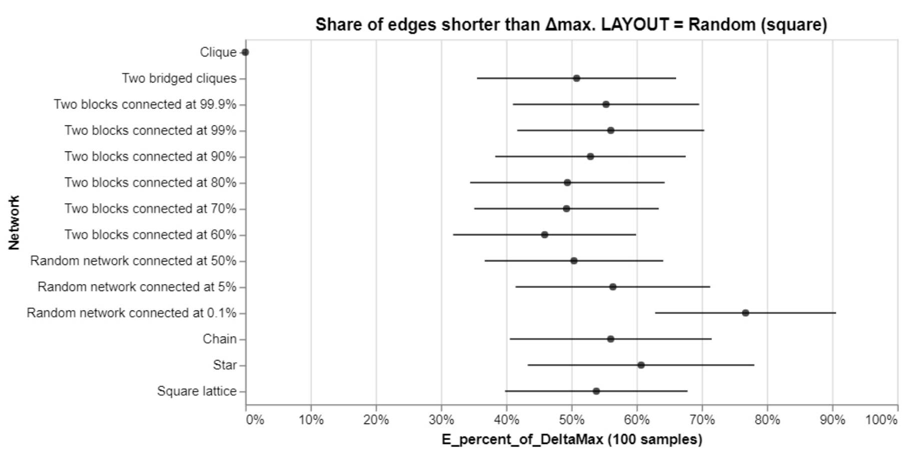
For each *(network generator, layout)* pair, the values for the 100 renditions are represented as a dot (mean) with an error bar (standard deviation). Noteworthy: the standard deviation may be so low that the bar is not even visible (which is remarkable).

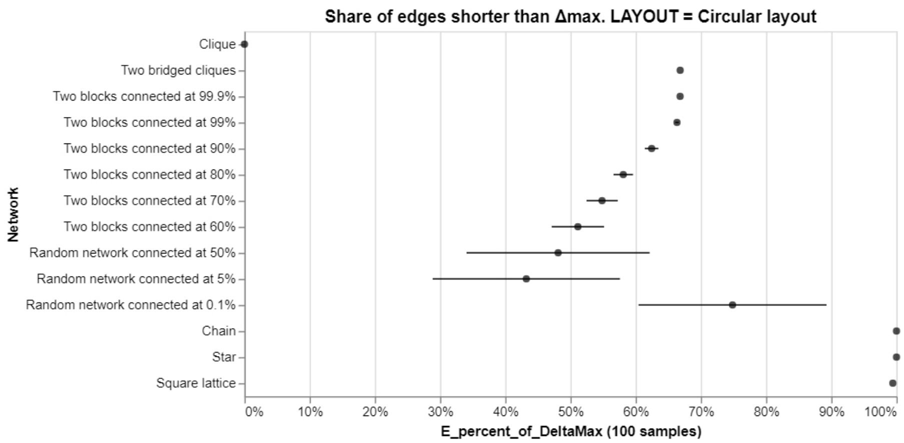


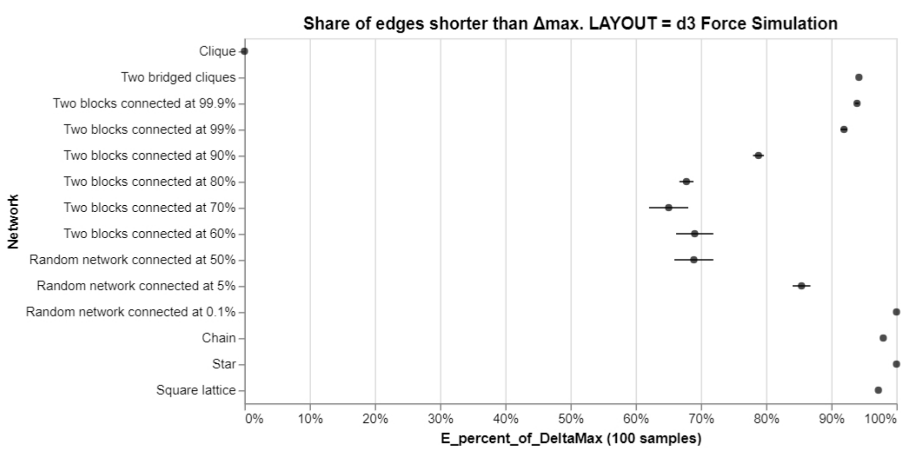



Share of edges shorter than Δmax. LAYOUT = Lin Log
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
E_percent_of_DeltaMax (100 samples)


Share of edges shorter than Δmax. LAYOUT = Random (disc)
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
E_percent_of_DeltaMax (100 samples)


Share of edges shorter than Δmax. LAYOUT = Random (square)
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
E_percent_of_DeltaMax (100 samples)

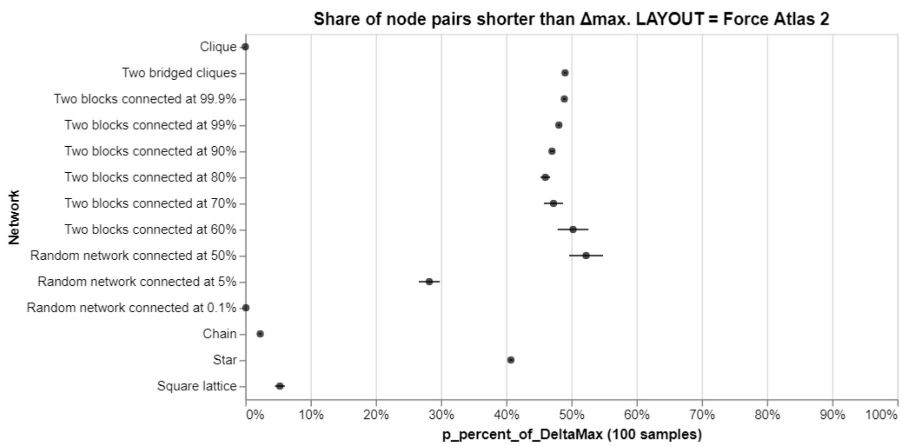

Share of edges shorter than Δmax. LAYOUT = Circular layout
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
E_percent_of_DeltaMax (100 samples)

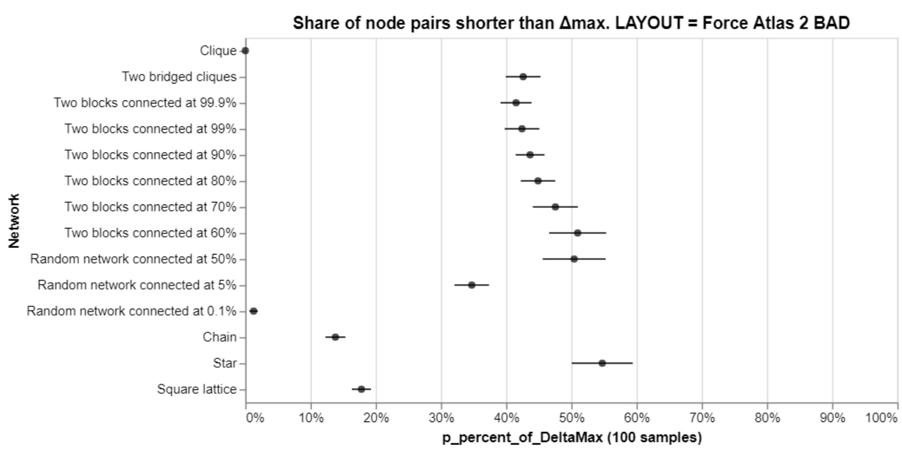

Share of edges shorter than Δmax. LAYOUT = d3 Force Simulation
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
E_percent_of_DeltaMax (100 samples)

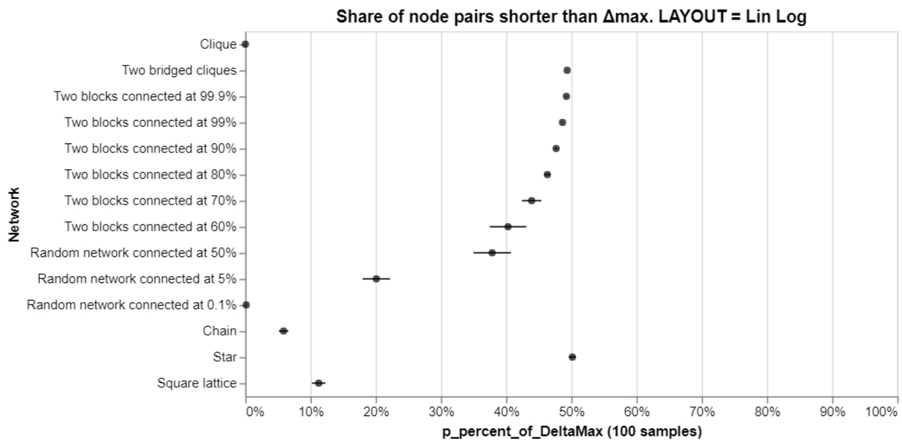
A.5 $p_{\%}(\Delta_{max})$, the share of node pairs closer than Δ_{max}.

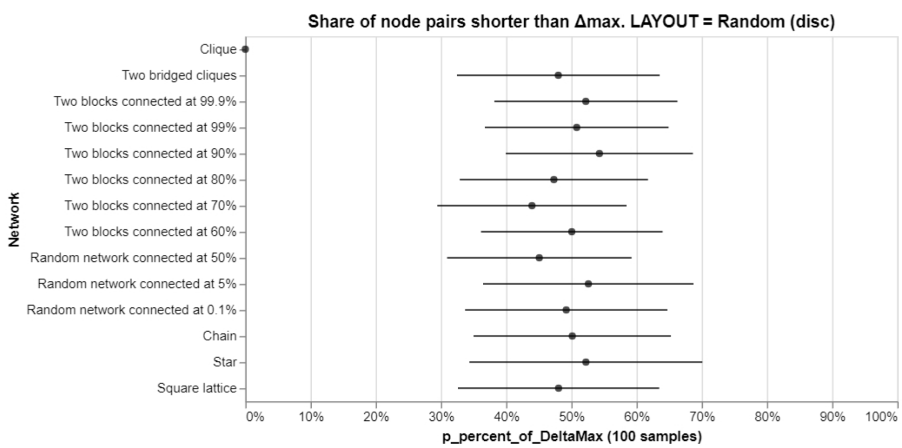
$p_{\%}(\Delta_{max})$ is a percentage of node pairs. For a given network, it allows comparing different layouts, and for a layout, it allows comparing different networks. It also compares to $E_{\%}(\Delta_{max})$.

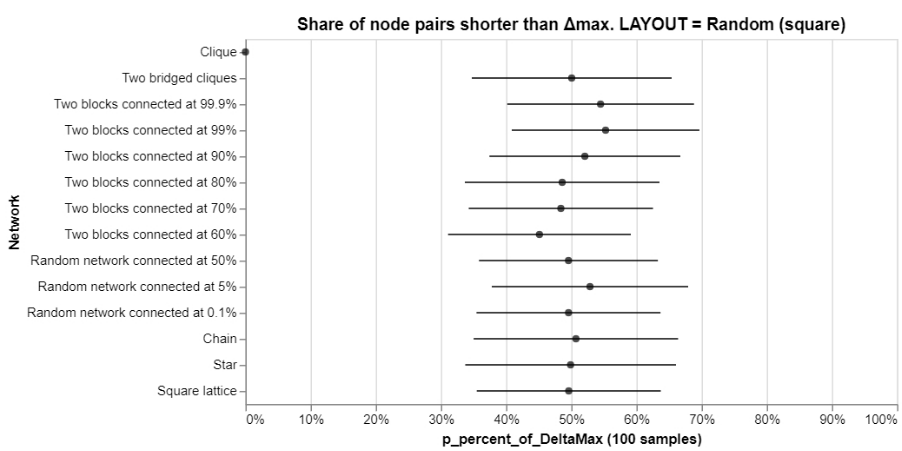
For each *(network generator, layout)* pair, the values for the 100 renditions are represented as a dot (mean) with an error bar (standard deviation). Noteworthy: the standard deviation may be so low that the bar is not even visible (which is remarkable).

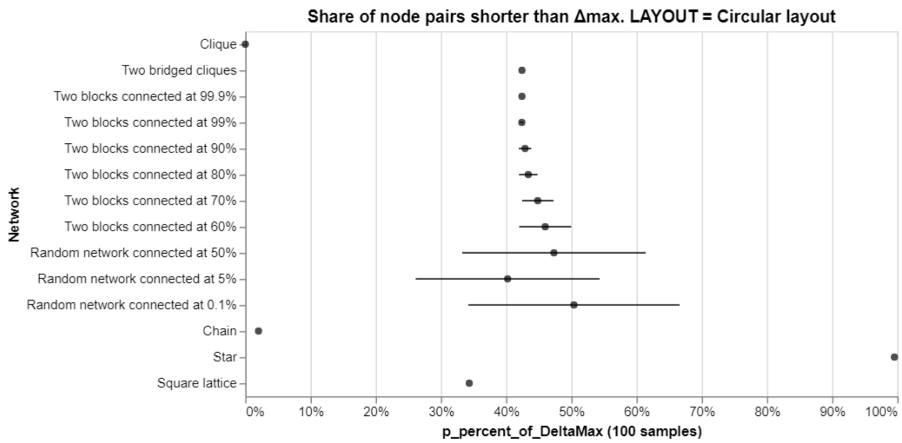


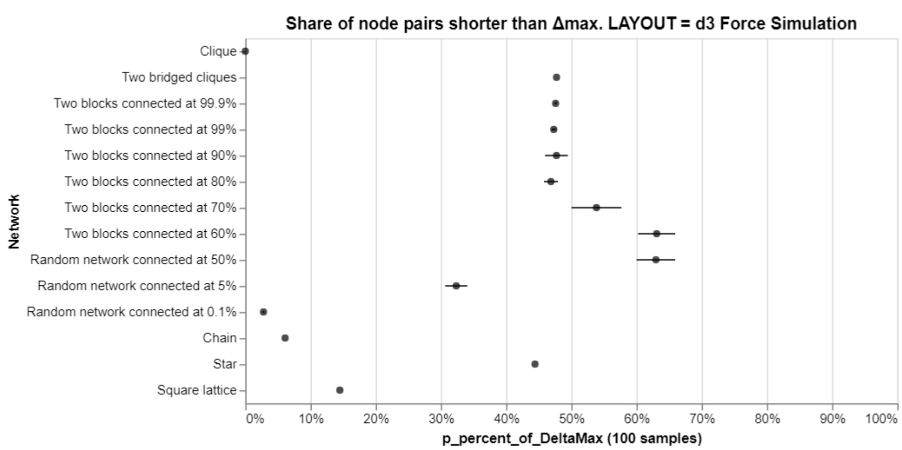



Share of node pairs shorter than Δmax. LAYOUT = Lin Log
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
p_percent_of_DeltaMax (100 samples)


Share of node pairs shorter than Δmax. LAYOUT = Random (disc)
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
p_percent_of_DeltaMax (100 samples)


Share of node pairs shorter than Δmax. LAYOUT = Random (square)
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
p_percent_of_DeltaMax (100 samples)


Share of node pairs shorter than Δmax. LAYOUT = Circular layout
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
p_percent_of_DeltaMax (100 samples)

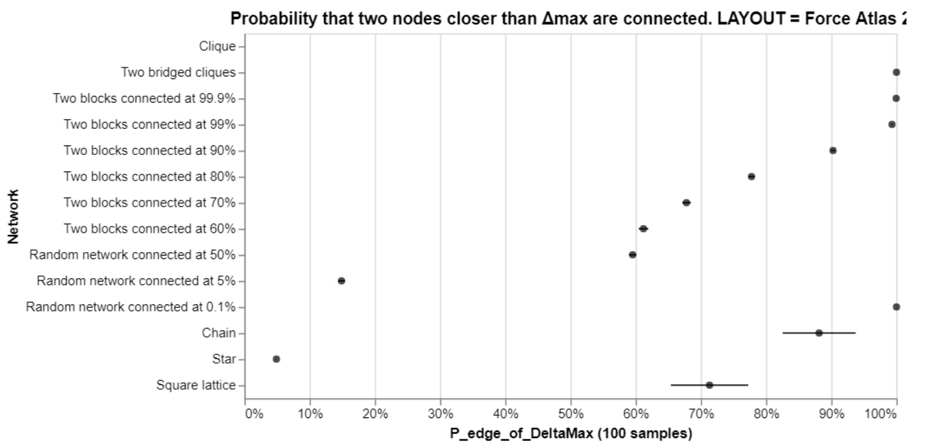

Share of node pairs shorter than Δmax. LAYOUT = d3 Force Simulation
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
p_percent_of_DeltaMax (100 samples)

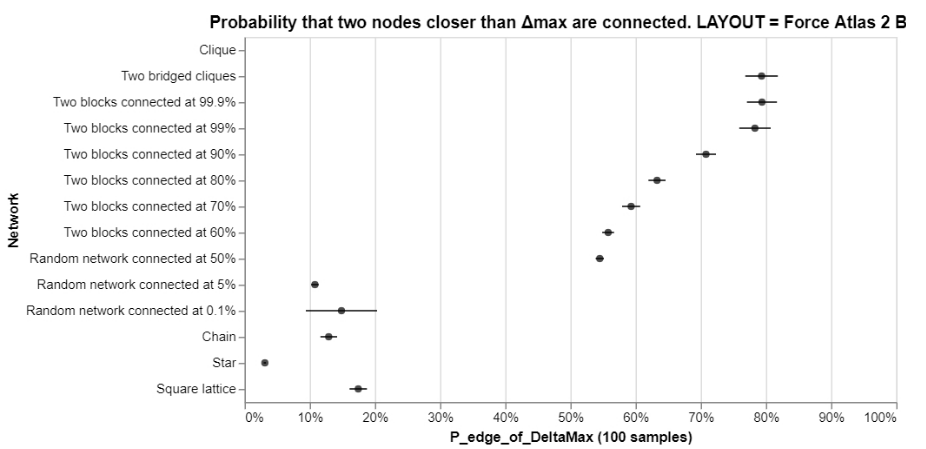
A.6 $P_{edge}(\Delta_{max})$, the probability that a node pair closer than Δ_{max} is connected (*precision*).

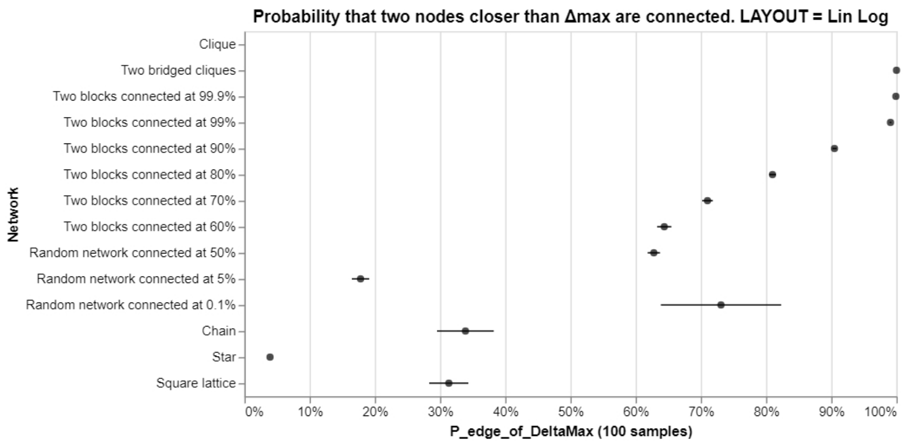
$P_{edge}(\Delta_{max})$ is a probability expressed as a percentage. Reminder for the purpose of clarity: the probability for two nodes to be connected given that they are close (i.e., $P_{edge}(\Delta_{max})$) is not the same as the probability for two nodes to be close given that they are connected (i.e., $E_{\%}(\Delta_{max})$). For a given network, $P_{edge}(\Delta_{max})$ allows comparing different layouts, and for a layout, it allows comparing different networks.

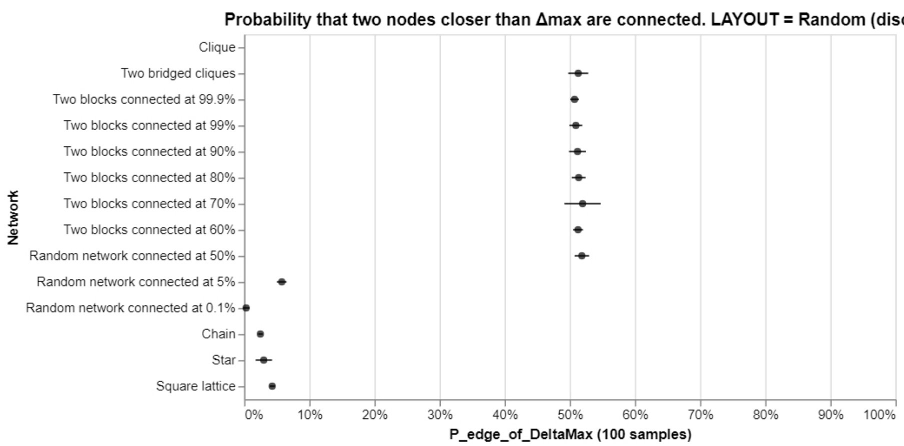
$P_{edge}(\Delta_{max})$ is also the precision of measuring connectedness with the distance Δ_{max}.

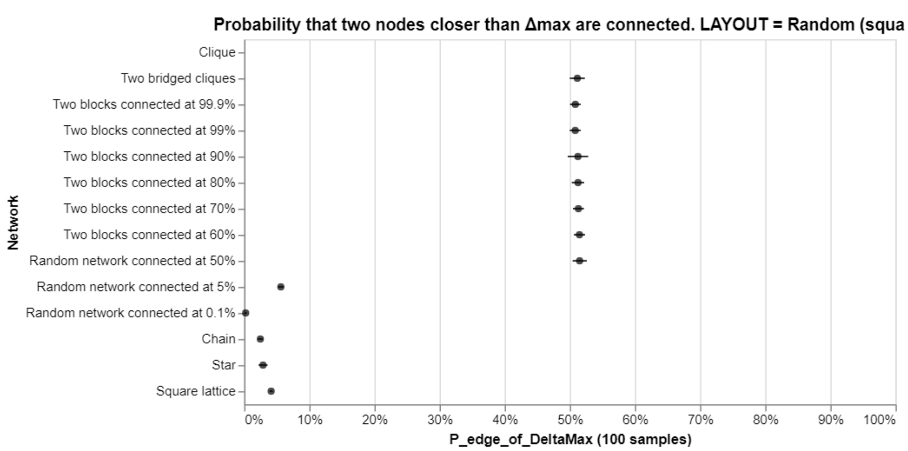
For each *(network generator, layout)* pair, the values for the 100 renditions are represented as a dot (mean) with an error bar (standard deviation). Noteworthy: the standard deviation may be so low that the bar is not even visible (which is remarkable).

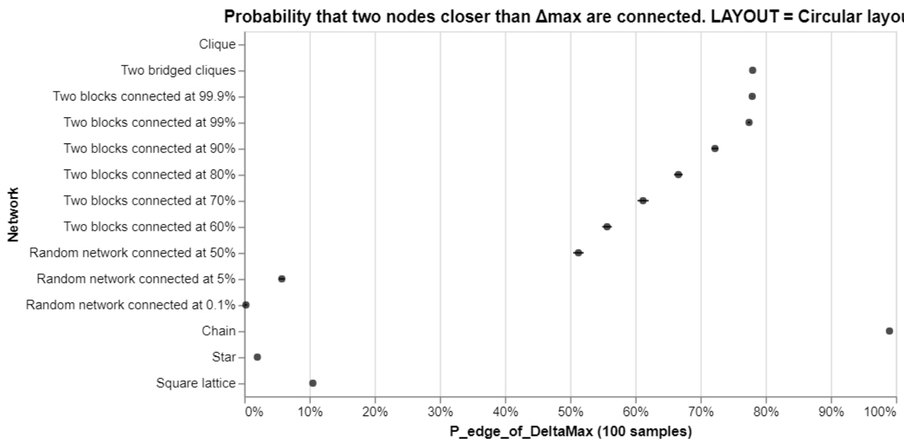




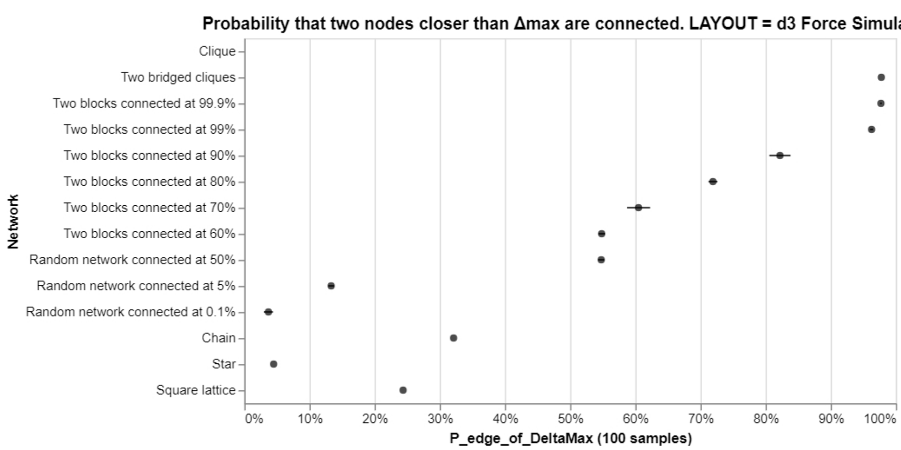

Probability that two nodes closer than Δmax are connected. LAYOUT = Lin Log
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
P_edge_of_DeltaMax (100 samples)


Probability that two nodes closer than Δmax are connected. LAYOUT = Random (disc
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
P_edge_of_DeltaMax (100 samples)


Probability that two nodes closer than Δmax are connected. LAYOUT = Random (squa
Network
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
P_edge_of_DeltaMax (100 samples)



Probability that two nodes closer than Δmax are connected. LAYOUT = Circular layou
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
Network
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
P_edge_of_DeltaMax (100 samples)



Probability that two nodes closer than Δmax are connected. LAYOUT = d3 Force Simula
Clique
Two bridged cliques
Two blocks connected at 99.9%
Two blocks connected at 99%
Two blocks connected at 90%
Two blocks connected at 80%
Two blocks connected at 70%
Two blocks connected at 60%
Random network connected at 50%
Random network connected at 5%
Random network connected at 0.1%
Chain
Star
Square lattice
Network
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
P_edge_of_DeltaMax (100 samples)

A.7 Benchmark data (tabular)

Network	Layout	C_{max} Avg.	C_{max} S.D.	Δ_{max} Avg.	Δ_{max} S.D.
clique	fa2	0.0%	0.00%	0	0.0
clique	fa2-bad	0.0%	0.00%	0	0.0
clique	linlog	0.0%	0.00%	0	0.0
clique	random-round	0.0%	0.00%	0	0.0
clique	random-graphology	0.0%	0.00%	0	0.0
clique	circular	0.0%	0.00%	0	0.0
clique	d3	0.0%	0.00%	0	0.0
bri-cli-2-1	fa2	50.5%	0.00%	278	5.8
bri-cli-2-1	fa2-bad	25.8%	2.72%	222	30.5
bri-cli-2-1	linlog	50.5%	0.01%	450	0.0
bri-cli-2-1	random-round	0.7%	0.46%	265	137.8
bri-cli-2-1	random-graphology	0.7%	0.50%	262	129.1
bri-cli-2-1	circular	25.0%	0.00%	620	0.0
bri-cli-2-1	d3	47.4%	0.00%	300	0.0
sbm-999	fa2	50.4%	0.06%	278	6.5
sbm-999	fa2-bad	25.3%	2.56%	220	27.5
sbm-999	linlog	50.4%	0.07%	450	0.0
sbm-999	random-round	0.8%	0.63%	283	122.9
sbm-999	random-graphology	0.8%	0.46%	280	118.9
sbm-999	circular	25.0%	0.05%	620	0.0
sbm-999	d3	47.3%	0.12%	299	3.0
sbm-99	fa2	49.5%	0.21%	276	5.0
sbm-99	fa2-bad	24.9%	2.48%	222	29.9
sbm-99	linlog	49.4%	0.20%	450	0.0
sbm-99	random-round	0.8%	0.43%	273	122.4
sbm-99	random-graphology	0.8%	0.51%	281	122.8
sbm-99	circular	24.5%	0.15%	620	5.9
sbm-99	d3	45.7%	0.38%	294	4.9
sbm-90	fa2	39.3%	0.70%	270	4.7
sbm-90	fa2-bad	18.9%	2.12%	217	19.5
sbm-90	linlog	40.3%	0.49%	482	8.9
sbm-90	random-round	0.8%	0.50%	291	127.9
sbm-90	random-graphology	0.9%	0.54%	268	123.6
sbm-90	circular	20.0%	0.52%	624	23.1
sbm-90	d3	31.8%	2.90%	268	8.2
sbm-80	fa2	26.6%	0.92%	241	6.8
sbm-80	fa2-bad	12.3%	1.79%	217	24.1
sbm-80	linlog	29.7%	0.82%	493	13.3
sbm-80	random-round	0.8%	0.44%	262	128.2
sbm-80	random-graphology	0.8%	0.42%	256	126.1
sbm-80	circular	15.0%	0.56%	629	33.8
sbm-80	d3	21.4%	0.79%	246	7.8
sbm-70	fa2	17.3%	0.70%	228	10.3
sbm-70	fa2-bad	9.1%	2.15%	239	40.0
sbm-70	linlog	19.0%	0.73%	449	23.6
sbm-70	random-round	0.8%	0.45%	244	126.4
sbm-70	random-graphology	0.9%	0.45%	256	123.7
sbm-70	circular	10.2%	0.67%	646	57.3
sbm-70	d3	11.4%	2.74%	251	21.7

Network	Layout	C_{max} Avg.	C_{max} S.D.	Δ_{max} Avg.	Δ_{max} S.D.
sbm-60	fa2	11.6%	0.67%	225	14.4
sbm-60	fa2-bad	5.9%	1.09%	276	37.6
sbm-60	linlog	11.7%	0.69%	400	35.9
sbm-60	random-round	0.9%	0.50%	274	121.0
sbm-60	random-graphology	0.8%	0.49%	241	122.9
sbm-60	circular	5.2%	0.63%	656	94.2
sbm-60	d3	6.0%	0.71%	271	18.9
rand-50	fa2	10.0%	0.52%	228	15.8
rand-50	fa2-bad	4.5%	0.82%	275	36.8
rand-50	linlog	9.7%	0.58%	376	36.5
rand-50	random-round	0.9%	0.48%	253	126.9
rand-50	random-graphology	0.8%	0.44%	260	117.4
rand-50	circular	0.8%	0.51%	613	301.5
rand-50	d3	6.0%	0.65%	270	19.6
rand-5	fa2	56.2%	2.72%	104	6.3
rand-5	fa2-bad	39.8%	3.62%	74	7.5
rand-5	linlog	50.2%	2.51%	92	10.7
rand-5	random-round	3.1%	1.89%	284	148.0
rand-5	random-graphology	3.5%	1.95%	275	132.1
rand-5	circular	3.0%	2.25%	534	330.5
rand-5	d3	53.9%	3.36%	249	10.6
rand-01	fa2	99.9%	0.04%	10	0.9
rand-01	fa2-bad	98.7%	1.24%	8	1.2
rand-01	linlog	99.9%	0.05%	10	1.0
rand-01	random-round	25.6%	18.42%	272	139.5
rand-01	random-graphology	27.2%	15.87%	257	119.0
rand-01	circular	23.9%	18.89%	632	339.1
rand-01	d3	97.2%	0.28%	100	2.0
chain	fa2	97.3%	0.52%	35	5.4
chain	fa2-bad	72.9%	3.95%	32	3.7
chain	linlog	89.4%	2.36%	68	12.5
chain	random-round	5.5%	3.38%	274	134.6
chain	random-graphology	5.3%	2.51%	268	138.2
chain	circular	98.0%	0.00%	40	0.0
chain	d3	93.1%	0.00%	140	0.0
star	fa2	59.2%	0.47%	71	0.6
star	fa2-bad	30.2%	8.65%	78	11.7
star	linlog	48.6%	0.72%	71	1.0
star	random-round	9.6%	9.29%	286	175.4
star	random-graphology	10.9%	9.65%	264	151.7
star	circular	0.4%	0.00%	1000	0.0
star	d3	55.6%	0.00%	195	0.0
sq-latt	fa2	94.2%	3.19%	67	6.1
sq-latt	fa2-bad	67.0%	4.60%	46	4.0
sq-latt	linlog	83.9%	4.63%	108	8.9
sq-latt	random-round	4.2%	2.13%	264	134.5
sq-latt	random-graphology	4.2%	2.29%	259	121.1
sq-latt	circular	65.1%	0.00%	510	0.0
sq-latt	d3	83.1%	0.00%	200	0.0

Network	Layout	$E_{\%}(\Delta_{max})$ Avg.	$E_{\%}(\Delta_{max})$ S.D.	$p_{\%}(\Delta_{max})$ Avg.	$p_{\%}(\Delta_{max})$ S.D.
clique	fa2	0.0%	0.00%	0.0%	0.00%
clique	fa2-bad	0.0%	0.00%	0.0%	0.00%
clique	linlog	0.0%	0.00%	0.0%	0.00%
clique	random-round	0.0%	0.00%	0.0%	0.00%
clique	random-graphology	0.0%	0.00%	0.0%	0.00%
clique	circular	0.0%	0.00%	0.0%	0.00%
clique	d3	0.0%	0.00%	0.0%	0.00%
bri-cli-2-1	fa2	99.0%	0.69%	49.0%	0.34%
bri-cli-2-1	fa2-bad	67.8%	4.83%	42.6%	5.27%
bri-cli-2-1	linlog	99.7%	0.05%	49.4%	0.02%
bri-cli-2-1	random-round	48.7%	31.01%	48.0%	31.08%
bri-cli-2-1	random-graphology	50.8%	30.50%	50.1%	30.64%
bri-cli-2-1	circular	66.8%	0.00%	42.4%	0.00%
bri-cli-2-1	d3	94.2%	0.00%	47.7%	0.00%
sbm-999	fa2	98.8%	0.75%	48.9%	0.38%
sbm-999	fa2-bad	66.3%	4.80%	41.5%	4.70%
sbm-999	linlog	99.4%	0.30%	49.2%	0.12%
sbm-999	random-round	53.0%	27.97%	52.2%	28.06%
sbm-999	random-graphology	55.3%	28.52%	54.5%	28.63%
sbm-999	circular	66.8%	0.05%	42.4%	0.00%
sbm-999	d3	93.9%	0.53%	47.6%	0.34%
sbm-99	fa2	96.5%	0.55%	48.1%	0.31%
sbm-99	fa2-bad	66.8%	5.10%	42.4%	5.29%
sbm-99	linlog	97.3%	0.38%	48.6%	0.11%
sbm-99	random-round	51.6%	28.14%	50.8%	28.16%
sbm-99	random-graphology	56.1%	28.64%	55.3%	28.73%
sbm-99	circular	66.3%	0.50%	42.4%	0.45%
sbm-99	d3	92.0%	1.03%	47.3%	0.65%
sbm-90	fa2	85.4%	1.00%	47.0%	0.55%
sbm-90	fa2-bad	62.2%	4.78%	43.7%	4.38%
sbm-90	linlog	87.0%	0.75%	47.7%	0.36%
sbm-90	random-round	55.1%	28.62%	54.3%	28.65%
sbm-90	random-graphology	52.9%	29.21%	52.1%	29.24%
sbm-90	circular	62.4%	2.03%	42.9%	1.86%
sbm-90	d3	78.8%	1.65%	47.7%	3.46%
sbm-80	fa2	72.0%	1.99%	46.0%	1.46%
sbm-80	fa2-bad	57.0%	5.63%	44.9%	5.25%
sbm-80	linlog	75.4%	1.40%	46.3%	0.92%
sbm-80	random-round	48.1%	28.80%	47.3%	28.82%
sbm-80	random-graphology	49.4%	29.77%	48.6%	29.81%
sbm-80	circular	58.1%	2.96%	43.4%	2.81%
sbm-80	d3	67.8%	2.13%	46.9%	2.12%
sbm-70	fa2	64.2%	3.09%	47.3%	2.94%
sbm-70	fa2-bad	56.5%	6.72%	47.6%	6.86%
sbm-70	linlog	62.5%	3.20%	43.9%	2.96%
sbm-70	random-round	44.7%	29.05%	44.0%	29.00%
sbm-70	random-graphology	49.3%	28.30%	48.4%	28.24%
sbm-70	circular	54.8%	4.76%	44.8%	4.79%
sbm-70	d3	65.1%	6.01%	53.9%	7.64%

Network	Layout	$E_{\%}(\Delta_{max})$ Avg.	$E_{\%}(\Delta_{max})$ S.D.	$p_{\%}(\Delta_{max})$ Avg.	$p_{\%}(\Delta_{max})$ S.D.
sbm-60	fa2	61.6%	4.79%	50.3%	4.71%
sbm-60	fa2-bad	56.8%	8.66%	51.0%	8.73%
sbm-60	linlog	51.7%	5.85%	40.3%	5.58%
sbm-60	random-round	50.9%	27.75%	50.1%	27.82%
sbm-60	random-graphology	45.9%	28.04%	45.1%	28.02%
sbm-60	circular	51.1%	8.02%	46.0%	7.96%
sbm-60	d3	69.1%	5.66%	63.1%	5.64%
rand-50	fa2	62.1%	5.24%	52.3%	5.14%
rand-50	fa2-bad	54.8%	9.75%	50.4%	9.62%
rand-50	linlog	47.4%	5.89%	37.9%	5.71%
rand-50	random-round	46.0%	28.22%	45.1%	28.25%
rand-50	random-graphology	50.4%	27.38%	49.6%	27.42%
rand-50	circular	48.1%	28.03%	47.3%	28.09%
rand-50	d3	68.9%	5.92%	63.0%	5.87%
rand-5	fa2	83.6%	3.64%	28.2%	3.18%
rand-5	fa2-bad	73.9%	5.56%	34.7%	5.24%
rand-5	linlog	69.4%	4.51%	20.1%	4.19%
rand-5	random-round	55.7%	32.44%	52.6%	32.24%
rand-5	random-graphology	56.4%	29.82%	52.9%	30.09%
rand-5	circular	43.2%	28.69%	40.2%	28.17%
rand-5	d3	85.4%	2.70%	32.3%	3.35%
rand-01	fa2	100.0%	0.00%	0.1%	0.04%
rand-01	fa2-bad	100.0%	0.00%	1.3%	1.24%
rand-01	linlog	100.0%	0.00%	0.1%	0.05%
rand-01	random-round	74.8%	27.03%	49.2%	30.98%
rand-01	random-graphology	76.7%	27.73%	49.6%	28.21%
rand-01	circular	74.9%	28.81%	50.4%	32.37%
rand-01	d3	100.0%	0.00%	2.8%	0.28%
chain	fa2	98.5%	1.16%	2.3%	0.38%
chain	fa2-bad	85.7%	4.41%	13.8%	3.03%
chain	linlog	93.6%	2.11%	5.9%	1.45%
chain	random-round	55.7%	29.59%	50.1%	30.23%
chain	random-graphology	56.0%	30.92%	50.7%	31.33%
chain	circular	100.0%	0.00%	2.0%	0.00%
chain	d3	98.0%	0.00%	6.1%	0.00%
star	fa2	99.9%	0.51%	40.7%	0.52%
star	fa2-bad	84.7%	13.24%	54.7%	9.35%
star	linlog	98.5%	1.63%	50.2%	1.03%
star	random-round	61.8%	37.82%	52.2%	35.70%
star	random-graphology	60.7%	34.73%	49.9%	32.28%
star	circular	100.0%	0.00%	99.6%	0.00%
star	d3	100.0%	0.00%	44.4%	0.00%
sq-latt	fa2	99.3%	2.19%	5.3%	1.47%
sq-latt	fa2-bad	83.6%	4.34%	17.8%	2.88%
sq-latt	linlog	93.9%	3.79%	11.3%	2.01%
sq-latt	random-round	52.2%	30.54%	48.0%	30.83%
sq-latt	random-graphology	53.8%	27.97%	49.6%	28.21%
sq-latt	circular	99.4%	0.00%	34.3%	0.00%
sq-latt	d3	97.2%	0.00%	14.5%	0.00%

Network	Layout	$P_{edge}(\Delta_{max})$ Avg.	$P_{edge}(\Delta_{max})$ S.D.
clique	fa2		
clique	fa2-bad		
clique	linlog		
clique	random-round		
clique	random-graphology		
clique	circular		
clique	d3		
bri-cli-2-1	fa2	100.0%	0.00%
bri-cli-2-1	fa2-bad	79.3%	4.98%
bri-cli-2-1	linlog	100.0%	0.00%
bri-cli-2-1	random-round	51.3%	3.03%
bri-cli-2-1	random-graphology	51.1%	2.26%
bri-cli-2-1	circular	78.0%	0.00%
bri-cli-2-1	d3	97.8%	0.00%
sbm-999	fa2	100.0%	0.05%
sbm-999	fa2-bad	79.4%	4.59%
sbm-999	linlog	99.9%	0.06%
sbm-999	random-round	50.7%	1.27%
sbm-999	random-graphology	50.8%	1.59%
sbm-999	circular	77.9%	0.07%
sbm-999	d3	97.7%	0.21%
sbm-99	fa2	99.3%	0.19%
sbm-99	fa2-bad	78.3%	4.79%
sbm-99	linlog	99.1%	0.22%
sbm-99	random-round	50.9%	1.92%
sbm-99	random-graphology	50.8%	1.65%
sbm-99	circular	77.5%	0.37%
sbm-99	d3	96.3%	0.45%
sbm-90	fa2	90.3%	0.63%
sbm-90	fa2-bad	70.8%	3.03%
sbm-90	linlog	90.5%	0.61%
sbm-90	random-round	51.2%	2.58%
sbm-90	random-graphology	51.2%	3.12%
sbm-90	circular	72.2%	1.01%
sbm-90	d3	82.2%	3.24%
sbm-80	fa2	77.8%	0.93%
sbm-80	fa2-bad	63.3%	2.60%
sbm-80	linlog	81.0%	0.99%
sbm-80	random-round	51.4%	2.09%
sbm-80	random-graphology	51.2%	1.89%
sbm-80	circular	66.6%	1.29%
sbm-80	d3	71.9%	1.34%
sbm-70	fa2	67.8%	1.30%
sbm-70	fa2-bad	59.3%	2.73%
sbm-70	linlog	71.0%	1.62%
sbm-70	random-round	52.0%	5.54%
sbm-70	random-graphology	51.3%	1.65%
sbm-70	circular	61.2%	1.68%
sbm-70	d3	60.5%	3.52%

Network	Layout	$P_{edge}(\Delta_{max})$ Avg.	$P_{edge}(\Delta_{max})$ S.D.
sbm-60	fa2	61.2%	1.49%
sbm-60	fa2-bad	55.8%	1.76%
sbm-60	linlog	64.4%	2.13%
sbm-60	random-round	51.3%	1.47%
sbm-60	random-graphology	51.5%	1.65%
sbm-60	circular	55.7%	1.43%
sbm-60	d3	54.8%	1.00%
rand-50	fa2	59.5%	1.13%
rand-50	fa2-bad	54.5%	1.24%
rand-50	linlog	62.8%	1.85%
rand-50	random-round	51.8%	2.17%
rand-50	random-graphology	51.5%	2.13%
rand-50	circular	51.3%	1.57%
rand-50	d3	54.8%	0.95%
rand-5	fa2	14.9%	1.05%
rand-5	fa2-bad	10.8%	1.09%
rand-5	linlog	17.8%	2.66%
rand-5	random-round	5.8%	1.45%
rand-5	random-graphology	5.6%	0.81%
rand-5	circular	5.7%	0.91%
rand-5	d3	13.3%	0.84%
rand-01	fa2	100.0%	0.00%
rand-01	fa2-bad	14.9%	10.87%
rand-01	linlog	73.1%	18.41%
rand-01	random-round	0.3%	0.99%
rand-01	random-graphology	0.2%	0.32%
rand-01	circular	0.2%	0.34%
rand-01	d3	3.7%	1.36%
chain	fa2	88.1%	11.16%
chain	fa2-bad	12.9%	2.53%
chain	linlog	33.9%	8.67%
chain	random-round	2.5%	0.66%
chain	random-graphology	2.5%	0.72%
chain	circular	99.0%	0.00%
chain	d3	32.1%	0.00%
star	fa2	4.9%	0.06%
star	fa2-bad	3.1%	0.34%
star	linlog	3.9%	0.04%
star	random-round	3.0%	2.50%
star	random-graphology	2.9%	1.36%
star	circular	2.0%	0.00%
star	d3	4.5%	0.00%
sq-latt	fa2	71.3%	11.84%
sq-latt	fa2-bad	17.4%	2.63%
sq-latt	linlog	31.3%	5.97%
sq-latt	random-round	4.3%	0.76%
sq-latt	random-graphology	4.1%	0.46%
sq-latt	circular	10.5%	0.00%
sq-latt	d3	24.4%	0.00%