AGWAN: A Generative Model for Labelled, Weighted Graphs

Michael Davis, Weiru Liu, and Paul Miller

Centre for Secure Information Technologies (CSIT),
School of Electronics, Electrical Engineering and Computer Science,
Queen's University, Belfast, United Kingdom
{mdavis05, w.liu}@qub.ac.uk, p.miller@ecit.qub.ac.uk

Abstract. Real-world graphs or networks tend to exhibit a well-known set of properties, such as heavy-tailed degree distributions, clustering and community formation. Much effort has been directed into creating realistic and tractable models for unlabelled graphs, which has yielded insights into graph structure and evolution. Recently, attention has moved to creating models for labelled graphs: many real-world graphs are labelled with both discrete and numeric attributes. In this paper, we present AGWAN (Attribute Graphs: Weighted and Numeric), a generative model for random graphs with discrete labels and weighted edges. The model is easily generalised to edges labelled with an arbitrary number of numeric attributes. We include algorithms for fitting the parameters of the AGWAN model to real-world graphs and for generating random graphs from the model. Using the Enron "who communicates with whom" social graph, we compare our approach to state-of-the-art random labelled graph generators and draw conclusions about the contribution of discrete vertex labels and edge weights to the structure of real-world graphs.

Keywords: Network models, graph generators, random graphs, labelled graphs, weighted graphs, graph mining

1 Introduction

Network analysis is concerned with finding patterns and anomalies in real-world graphs, such as social networks, computer and communication networks, or biological and ecological processes. Real graphs exhibit a number of interesting structural and evolutionary properties, such as power-law or log-normal degree distribution, small diameter, shrinking diameter, and the Densification Power Law [4, 13, 15].

Besides discovering network properties, researchers are interested in the mechanisms of network formation. Generative graph models provide an abstraction of how graphs form; if the model is accurate, generated graphs will obey the same properties as real graphs. Generated graphs are also useful for simulation experiments, hypothesis testing and making predictions about graph evolution or missing graph elements. Most existing models are for unlabelled, unweighted graphs [4, 13]; a generative model for random graphs with discrete labels has recently been proposed [11].

In this paper, we present AGWAN, a generative model for labelled, weighted graphs. Weights are commonly used to represent the number of occurrences of each edge: the number of e-mails sent between individuals in a social network [1]; the number of calls to a subroutine in a software call graph [6]; or the number of people walking between a pair of door sensors in a building access control network [5]. In other applications, the edge weight may represent continuous values: donation amounts in a bipartite graph of donors and political candidates [1]; distance or speed in a transportation network [6]; or elapsed time between the sensors in the building network [5]. In some cases, the weight is a multi-dimensional feature vector [5, 6].

Our main motivation for this work is to create a model to better understand the laws governing the relationship between graph structure and numeric labels or weights. Furthermore, we want to be able to create realistic random, labelled, weighted graphs for large-scale simulation experiments for our pattern discovery algorithms [5]. Our experiments in §5 show the extent to which various graph properties are related to labels and weights, and measure exactly how "realistic" our random graphs are. Graphs generated with AGWAN are shown to have more realistic degree strength distributions and spectral properties than the comparative methods.

This paper is arranged as follows: §2 is an overview of generative graph models; §3 presents AGWAN, our generative model for weighted and numeric labelled graphs. We include a fitting algorithm to learn AGWAN's parameters from a real input graph, and an algorithm to generate random graphs from the model. §4 gives an overview of the Enron "who communicates with whom" social graph that we use in the experiments, and outlines the statistical measures and tests that we use to evaluate the generated graphs. The experiments in §5 demonstrate that the vertex labels and edge weights of a graph can predict the graph structure with high accuracy. Conclusions are in §6.

2 Related Work

Our understanding of the mathematical properties of graph structure was pioneered by Paul Erdős and Alfréd Rényi [7]. Graph formation is modelled as a Bernoulli process, parameterised by the number of vertices and a wiring probability between each vertex pair. While it has been essential to our understanding of component sizes and expected diameter, the Erdős-Rényi model does not explain other important properties of real-world graphs such as degree distribution, transitivity and clustering [4, 15].

Barabási and Albert's Preferential Attachment model [2] uses the "rich get richer" principle to grow graphs from a few vertices up to the desired size. The probability of an edge is proportional to the number of edges already connected to a vertex. This generates graphs with power-law degree distributions. A number of variants of Preferential Attachment have been proposed [4, 15]; notably, the Forest Fire model [14] which exhibits the Densification Power Law and shrinking diameter effect. Still, Preferential Attachment models lack some desired properties, such as community structure.

The RMat algorithm [4] solves the community structure problem with its recursive matrix approach. RMat graphs consist of 2^n vertices and E edges, with four probabilities a, b, c, d to determine in which quadrant of the adjacency matrix each edge falls. These parameters allow the specification of power-law or log-normal degree distributions; if $a = b = c = d$, the result will be an Erdős-Rényi graph. However, RMat does not model the Densification Power Law and shrinking diameter effect.

Kronecker Graphs [13] are a generalisation of RMat graphs which fulfil all the properties mentioned above. The model starts with an initiator matrix. Kronecker (Tensor) multipication is recursively applied to yield the final adjacency matrix of the desired size. This work synthesises the previous work in random graphs in a very elegant way and proves that RMat graphs are a special case of Stochastic Kronecker graphs. While they fulfil the desired structural properties, Kronecker Graphs are unlabelled.

The Multiplicative Attribute Graph (MAG) model [11] is a generative model for labelled graphs. MAG is parameterised by the number of vertices, a set of prior probabilities for vertex label values and a set of *affinity matrices* specifying the probability of an edge conditioned on the vertex labels. The affinity matrices can be learned from real graphs using Maximum Likelihood Estimation [10]. [11] proves that Kronecker Graphs are a special case of MAG graphs, and that suitably-parameterized MAG graphs fulfil all the desired properties: log-normal or power-law degree distribution, small diameter, the existence of a unique giant component and the Densification Power Law. The MAG model considers discrete vertex labels only. We believe that our method, described in the next section, is the first generative model to include numeric labels or weights.

3 AGWAN: A Generative Model for Labelled, Weighted Graphs

In this section, we present our generative model, AGWAN (Attribute Graph: Weighted and Numeric). The model is illustrated in Fig. 1 for the Enron graph described in §4.

Consider a graph $G = (V, E)$ with discrete vertex label values drawn from a set L. In Fig. 1, $u, v \in V$ are vertices and $w_{uv}, w_{vu} \in \mathbb{R}$ are edge weights. Edges $e \in E$ are specified as a 3-tuple $\langle u, v, w_{uv} \rangle$. In the discussion which follows, we restrict ourselves to a single label on each vertex; we outline how this can be extended to multiple labels in §3.3.

We assume that the edge weights $W^{ij} = \{w_{ij}\}$ follow an arbitrary probability distribution. W^{ij} may be Gaussian, if it is drawn from an independent and identically distributed random variable such as speed, elapsed time, or amounts of money. If W^{ij} represents the number of events occurring in a fixed interval of time—number of e-mails sent or number of people moving between sensors—it is expected to follow a Poisson distribution. Other quantities—frequency of word use or number of papers written by scientists—follow a power-law distribution [15]. In many cases, W^{ij} arises from multiple processes. As the underlying distribution is unknown, we model W^{ij} using a Gaussian Mixture Model (GMM) with an arbitrary number of mixtures. This provides a reasonable approximation to any general probability distribution. We avoid the problem of knowing the "correct" number of mixtures by assuming that W^{ij} consists of an infinite number of mixtures and using variational inference to determine the optimal number for our model [3].

The AGWAN model is parameterised by μ, a set of prior probabilities over L; and Θ, a set of edge weight mixture parameters: $\Theta = \{\Omega^{ij} | i, j \in L\}$. For directed graphs, $|\Theta| = |L|^2$ and we need to generate both w_{uv} and w_{vu} (see Fig. 1). For undirected graphs, $\Omega^{ij} = \Omega^{ji}$, so $|\Theta| = O(|L|^2/2)$ and $w_{vu} = w_{uv}$.

For each combination of vertex attributes $\langle i, j \rangle$, the corresponding mixture model Ω^{ij} parameterises the distribution of edge weights (with an edge weight of 0 indicating no edge). Ω^{ij} is a GMM with M mixtures:

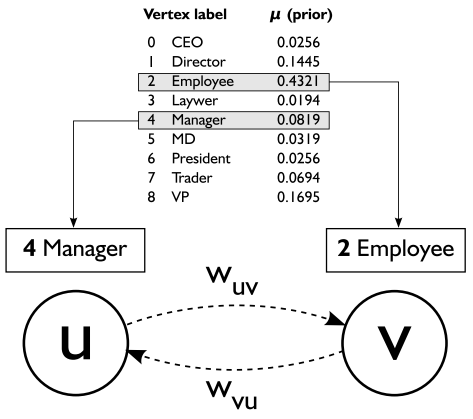


Fig. 1: AGWAN parameters. Vertex labels are selected according to prior probability μ. Edge weight w_{uv} is selected from mixture model Ω^{42} and w_{vu} is selected from mixture model Ω^{24}.

$$\Omega^{ij} = \sum_{m=0}^{M-1} \omega_m^{ij} \cdot \eta(\mu_m^{ij}, (\sigma^2)_m^{ij}) \tag{1}$$

where ω_m^{ij} is the weight of each mixture and $\eta(\mu_m^{ij}, (\sigma^2)_m^{ij})$ is the Gaussian PDF with mean μ_m^{ij} and variance $(\sigma^2)_m^{ij}$. The mixture weights form a probability distribution over the mixtures: $\sum_{m=0}^{M-1} \omega_m^{ij} = 1$. We can specify Ω^{ij} such that the first mixture encodes the probability of no edge: $\omega_0^{ij} = 1 - P(e_{ij})$, where $P(e_{ij})$ is the probability of an edge between pairs of vertices with labels $\langle i, j\rangle$. The model degenerates to an unweighted graph if there are two mixtures, $\eta_0(0,0)$ and $\eta_1(1,0)$. Furthermore, if the weights ω_m^{ij} are the same for all $\langle i, j\rangle$, the model degenerates to an Erdős-Rényi random graph.

3.1 Graph Generation

Algorithm 1 describes how to generate a random graph using AGWAN(N, L, μ, Θ). The number of vertices in the generated graph is specified by N. After assigning discrete label values to each vertex (lines 2–3, *cf.* Fig. 1), the algorithm checks each vertex pair $\langle u, v\rangle$ for the occurrence of an edge (lines 4–7). If $m = 0$, $\langle u, v\rangle$ is not an edge (line 7). If there is an edge, we assign its weight from mixture m (lines 8–9). The generated graph is returned as $G = (V, E)$.

3.2 Parameter Fitting

To create realistic random graphs, we need to learn the parameters μ, Θ from a real-world input graph G. Let W^{ij} be the set of edge weights between pairs of vertices with labels $\langle i, j\rangle$. During parameter fitting, we want to create a model Ω^{ij} for each W^{ij} in G. Each GMM Ω^{ij} has a finite number of mixtures M. If M is known, Ω^{ij} can be estimated using Expectation Maximisation [9]. However, not only is M unknown, but we expect that it will be different for each Ω^{ij} within a given graph model [5].

We solve this problem by modelling Ω^{ij} as a non-parametric mixture model with an unbounded number of mixtures: a Dirichlet Process Gaussian Mixture Model [3]

Algorithm 1 AGWAN Graph Generation

```
Require: $N$ (no. of vertices), $L$ (set of discrete label values), $\mu$ (prior distribution over $L$),
    $\Theta = \left\{\Omega^{ij}\right\}$ (set of mixture models)
1: Create vertex set $V$ of cardinality $N$, edge set $E = \emptyset$
2: for all $u \in V$ do
3:     Assign discrete label $l_u \in L$ from prior $\mu$
4: for all $u, v \in V : u \neq v$ do
5:     $i = l_u, j = l_v$
6:     Select Gaussian $m$ uniformly at random from $\Omega^{ij}$
7:     if $m \neq 0$ then
8:         Assign edge weight $w_{uv}$ uniformly at random from $\eta(\mu_m^{ij}, (\sigma^2)_m^{ij})$
9:         Create edge $e = \langle u, v, w_{uv} \rangle, E = E \cup \{e\}$
   return $G = (V, E)$
```

(DPGMM). "Non-parametric" does not mean that the model has no parameters; rather, the number of parameters is allowed to grow as more data are observed. In essence, the DPGMM is a probability distribution over the probability distributions of the model.

The Dirichlet Process (DP) over edge weights W^{ij} is a stochastic process $DP(\alpha, H_0)$, where α is a positive scaling parameter and H_0 is a finite measure on W^{ij}. If we draw a sample from $DP(\alpha, H_0)$, the result is a random distribution over values drawn from H_0. This distribution H is discrete, represented as an infinite sum of atomic measures. If H_0 is continuous, then the infinite set of probabilities corresponding to the frequency of each possible value that H can return are distributed according to a *stick-breaking process*. The stick-breaking representation of H is given as:

$$\omega_m^{ij}(\mathbf{x}) \prod_{n=1}^{m-1} (1 - \omega_n^{ij}) \qquad\qquad H = \sum_{n=1}^{\infty} \omega_n^{ij}(\mathbf{x}) \delta_{\eta_m^*} \tag{2}$$

where $\{\eta_1^*, \eta_2^*, \ldots\}$ are the atoms representing the mixture components. We learn the mixture parameters using the variational inference algorithm for generating Dirichlet Process Mixtures described in [3]. The weights of each component are generated one-at-a-time by the stick-breaking process, which tends to return the components with the largest weights first. In our experiments, the first 3–5 mixtures accounted for over 99% of the data. Mixtures with weights summing to less than 0.01 are dropped from the model, and the remaining weights $\{\omega_m^{ij}\}$ are normalised.

Algorithm 2 is the algorithm for AGWAN parameter fitting. First, we estimate the vertex priors (lines 1–3). Next, we sample the edge weights for each possible combination of vertex label values, with no edge counting as a weight of zero (lines 4–7). Finally, we estimate the GMMs Ω^{ij} from the appropriate set of samples W^{ij} using the the stick-breaking process described above.

3.3 Extending AGWAN to multiple attributes

We have presented AGWAN for a single discrete vertex label and a single numeric edge label (the weight). Many graphs have multiple labels on vertices and edges. AGWAN can be extended to multiple numeric edge labels by generalising the concept of edge weight to k dimensions. In this case, the mean of each mixture becomes a k-dimensional vector

Algorithm 2 AGWAN Parameter Fitting

```
Require: Input graph $G = (V, E)$
 1: $L = \{\text{discrete vertex label values}\},\ \ d = |L|$
 2: Calculate vertex label priors, apply Laplace smoothing $\forall l \in L : P(l) = \frac{count(l)+\alpha}{N+\alpha d}$
 3: $\mu$ = the normalised probability distribution over $L$ such that $\sum_{i=1}^{d} P(l_i) = 1$
 4: $\forall i, j \in L : W^{ij} = \emptyset$
 5: for all $u, v \in V : u \neq v$ do
 6:     $i = l_u, j = l_v$
 7:     $W^{ij} = W^{ij} \cup \{w_{uv}\}$            ▷ If $\langle u, v \rangle$ is not an edge, then $w_{uv}$ has value zero
 8: for all $i, j \in L$ do
 9:     estimate $\Omega^{ij}$ from $W^{ij}$ using variational inference
10: $\Theta = \{\Omega^{ij}\}$
          return $\mu, \Theta$
```

and the variance $(\sigma_m^{ij})^2$ is replaced with the $k \times k$ covariance matrix Σ_m^{ij}. The variational algorithm can be accelerated for higher-dimensional data using a kd-tree [12] and has been demonstrated to work efficiently on datasets of hundreds of dimensions.

A more difficult question is how to extend the model to multiple discrete vertex labels. With even a small number of labels, modelling the full joint probability across all possible combinations of label values becomes a complex combinatorial problem with hundreds or thousands of parameters. The MAG model reduces this complexity by assuming that vertex labels are independent, so edge probabilities can be computed as the product of the probabilities from each label [11]. For latent attributes, MAGFIT enforces independendence by regularising the variational parameters using mutual information [10]. However, the MAG model has not solved this problem for real attributes, where independence cannot be assumed. Furthermore, multiplying the probabilities sets an upper limit (proportional to $\log N$) on the number of attributes which can be used in the model. In our experiments (§5), using three latent attributes created a less accurate model than using two.

An alternative to multiplying independent probabilities is to calculate the GMM for each edge as the weighted summation of the GMM for each individual attribute; it is likely that some attributes have a large influence on graph structure while others affect it little or not at all. This problem remains a topic for further research; see §6.

4 Experiments

We evaluate our approach by comparing AGWAN with the state-of-the-art in labelled graph generation, represented by the MAG model [10, 11]. We learn the AGWAN and MAG model parameters from a real-world graph. We then generate random graphs from each model and calculate a series of statistics on each graph. These statistics are used to compare how closely the model maps to the input graph.

For our input graph, we use the "who communicates with whom" graph of the Enron e-mail corpus [1] (Fig. 2). The graph has 159 vertices representing Enron's senior employees and 2667 edges representing e-mail communications sent between them. Vertices have a single discrete label which describes the job role of the employee.

The edges are weighted according to the number of e-mails sent between each pair of employees. As e-mail communications are not symmetric, the edges are directed.

We evaluated AGWAN against the following models:

Erdős-Rényi random graph (ER): The ER model $G(n,p)$ has two parameters. We set the number of vertices $n = 159$ and the edge probability $p = 0.106$ to match the input graph as closely as possible. We do not expect a very close fit, but the ER model provides a useful baseline.

MAG with real attributes (MAG-R1): The MAG model with one real attribute is similar to AGWAN with one real attribute, with the difference that $\Theta = \{\Omega^{ij}\}$ is replaced with a set of binary edge probabilities, $\Theta = \{p^{ij}\}$.

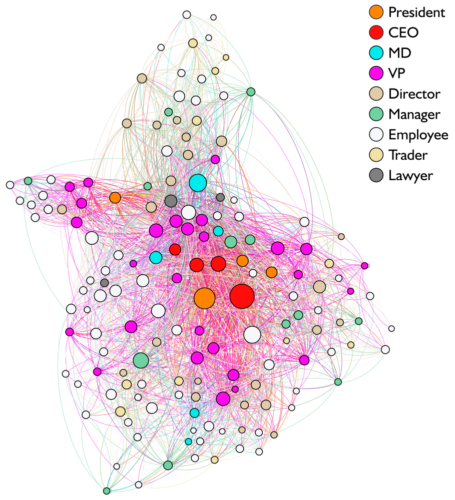


Fig. 2: Social graph of who communicates with whom in the Enron corpus

MAG with latent attributes (MAG-L1, MAG-L2, MAG-L3): The MAG model allows for modelling the graph structure using latent attributes. These models ignore the discrete label provided in the input graph and instead learn a set of latent binary attributes to describe the graph structure. Parameters for the latent attributes were learned using MAGFIT [10]. The model is sensitive to the number of latent attributes, so we repreated the experiments with 1, 2 and 3 attributes.

As these models do not generate weighted graphs, we set the weight of the edges in the generated graphs to the mean edge weight in the input graph (25.892). This ensures that statistics such as average degree strength are not skewed by unweighted edges.

To evaluate the closeness of fit of each model, we use the following statistics:

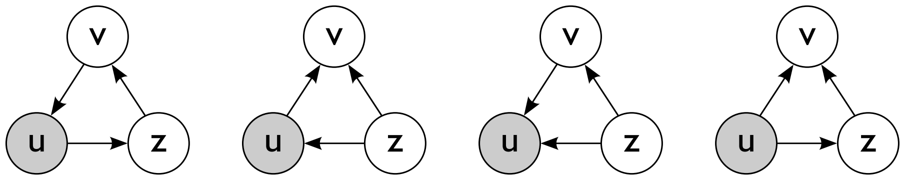
Degree Strength: For an unweighted graph, one of the most important measures is the degree distribution (the number of in-edges and out-edges of each vertex). Real-world graphs tend to have heavy-tailed power-law or log-normal degree distributions [4, 15]. For a weighted graph, we generalise the concept of vertex degree to vertex strength [8]:

$$s_u = \sum_{v \neq u} w_{uv} \tag{3}$$

For each generated graph, we plot a CDF of the in-strength, out-strength and total strength of each vertex.

Spectral Properties: We use Singular Value Decomposition (SVD) to calculate the singular values and singular vectors of the graph's adjacency matrix, which act as a signature of the most important features of the graph structure. In an unweighted graph, the adjacency matrix contains binary values, for "edge" or "no edge". In a weighted graph, the adjacency matrix contains the edge weights (with 0 indicating no edge). For SVD $U\Sigma V$, we plot CDFs of the singular values Σ and the components of the left singular vector U corresponding to the highest singular value.

Clustering Coefficients: the clustering coefficient C is an important measure of community structure. It measures the density of triangles in the graph, or the probability



$\mathbb{W}^{cycle}_{uvz} = w_{uz}^{\frac{1}{3}} \cdot w_{zv}^{\frac{1}{3}} \cdot w_{vu}^{\frac{1}{3}}$

(a) Cycle

$\mathbb{W}^{middleman}_{uvz} = w_{uv}^{\frac{1}{3}} \cdot w_{zu}^{\frac{1}{3}} \cdot w_{zv}^{\frac{1}{3}}$

(b) Middleman

$\mathbb{W}^{in}_{uvz} = w_{zu}^{\frac{1}{3}} \cdot w_{zv}^{\frac{1}{3}} \cdot w_{vu}^{\frac{1}{3}}$

(c) In

$\mathbb{W}^{out}_{uvz} = w_{uv}^{\frac{1}{3}} \cdot w_{uz}^{\frac{1}{3}} \cdot w_{zv}^{\frac{1}{3}}$

(d) Out

Fig. 3: Triad Patterns in a Directed Graph

that two neighbours of a vertex are themselves neighbours [15]. We extend the notion of clustering coefficients to weighted, directed graphs using the equation in [8]:

$$C_u = \frac{[\mathbf{W}_u^{[\frac{1}{3}]} + (\mathbf{W}_u^T)^{[\frac{1}{3}]}]^3_{uu}}{2[d_u^{tot}(d_u^{tot} - 1) - 2d_u^{\leftrightarrow}]} \tag{4}$$

where C_u is the weighted clustering coefficient for vertex u, $\mathbf{W}_u$ is the weighted adjacency matrix for u and its neighbours, $\mathbf{W}^T$ is the transpose of $\mathbf{W}$, d_u^{tot} is the total degree of a vertex (the sum of its in- and out-degrees) and $d_u^{\leftrightarrow}$ is the number of bilateral edges in u (the number of neighbours of u which have both an in-edge and an out-edge between themselves and u).

Triad Participation: Closely related to the clustering coefficient is the concept of triangle or triad participation. The number of triangles that a vertex is connected to is a measure of transitivity [15]. In a directed graph, the triangles have a different interpretation depending on the edge directions. There are four types of triangle pattern [8], as shown in Fig. 3. To generalise the concept of triad participation to weighted, directed graphs, we consider each of the four triangle types separately, and sum the total strength of the edges in each triad:

$$t_u^y = \sum_{v,z \in \mathbf{W}_u \setminus u} \mathbb{W}^y_{uvz} \tag{5}$$

where $y = \{cycle, middleman, in, out\}$ is the triangle type and $\mathbb{W}^y_{uvz}$ is calculated as shown in Fig. 3 for each triangle type y.

To give a more objective measure of the closeness of fit between the generated graphs and the input graph, we use a Kolmogorov-Smirnov (KS) test and the L2 (Euclidean) distance between the CDFs for each statistic. As the CDFs are for heavy-tailed distributions, we use the logarithmic variants of these measures [10]. The KS and L2 statistics are calculated as:

$$KS(D_1, D_2) = max_x |\log D_1(x) - \log D_2(x)| \tag{6}$$

$$L2(D_1, D_2) = \sqrt{\frac{1}{\log b - \log a} \sum_{x=a}^{b} (\log D_1(x) - \log D_2(x))^2} \tag{7}$$

where $[a, b]$ is the support of distributions D_1 and D_2.

The model that generates graphs with the lowest KS and L2 values for each of the statistics discussed above has the closest fit to the real-world graph. The results are presented in the next section.

5 Results

For each of the six models (AGWAN, MAG-R1, MAG-L1, MAG-L2, MAG-L3, ER), we generated 10 random graphs and calculated statistics for each. The plots of the averaged CDFs of the 10 graphs for each model are shown in Fig.s 4–7. The closeness of fit of each CDF (KS and L2 statistics) are shown in Tables 1–2.

Vertex Strength (Fig. 4): AGWAN successfully models the in-strength and out-strength of the input graph with high accuracy, significantly better than any of the other approaches. AGWAN successfully predicts the heavy-tailed vertex strength distribution (small number of high-strength vertices and large number of low-strength vertices).

Spectral Properties (Fig. 5): Fig. 5a shows that while AGWAN's singular values are not absolutely the closest to the values from the real graph, the shape of the distribution follows the real graph more closely than any of the other approaches. Fig. 5b shows that the components of the primary left singular vector map very closely to the real graph. Thus the spectral properties of the AGWAN graphs are very realistic, while the other models have singular vectors not much closer than random.

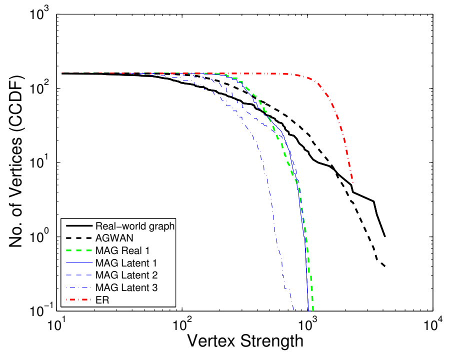
Clustering Coefficients (Fig. 6): The clustering coefficients map very closely to those from MAG-R1, outperforming it by only a small margin. Both approaches are outperformed by MAG-L1 for the In-Coefficient and MAG-L2 for the Out-Coefficient. It is interesting that there is no clear winner overall for clustering properties.

Triad Participation (Fig. 7): Triad participation is closely related to clustering, so it is not suprising to find similar results: the results for AGWAN and MAG-R1 are very similar, but MAG-L2 appears to model the real-world triad participation most closely.

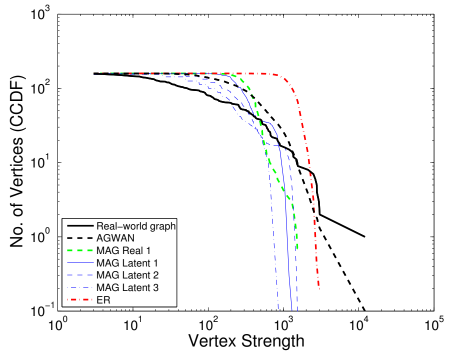
Our results demonstrate that AGWAN produces an accurate model of the properties of a weighted real-world graph, significantly outperforming the unweighted approach (MAG-R1) across the statistics measured. It is interesting to note that for vertex strength and spectral qualities, AGWAN performs much better than the MAG latent approaches, whereas for clustering and triad participation, the latent approaches perform better. It is likely that the attributes in MAG-L2 and MAG-L3 are modelling different aspects of the graph structure (*e.g.* homophily and core-periphery). One of the findings in [10] was that "Simplified MAG" (where all attributes are the same) could not model the clustering property, implying that clustering cannot be accurately modelled using a single attribute. We propose to extend our model to more than one attribute as outlined in §3.3 to investigate whether this produces a more accurate model of clustering.

6 Conclusions

We presented AGWAN, a model for random graphs with discrete labels and weighted edges. We proposed a fitting algorithm to learn a model of graph edge weights from real-world data, and a generative algorithm to generate random graphs with similar characteristics to the real-world graph. We measured the closeness of fit of our generated graphs to the input graph over a range of graph statistics, and compared our

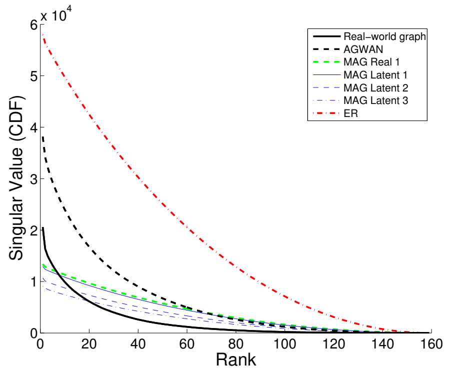


(a) In-Strength

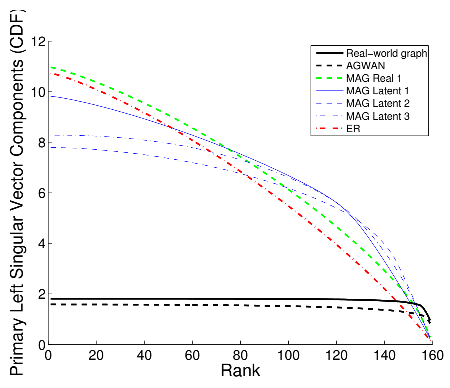


(b) Out-Strength

Fig. 4: Vertex Strength Distribution

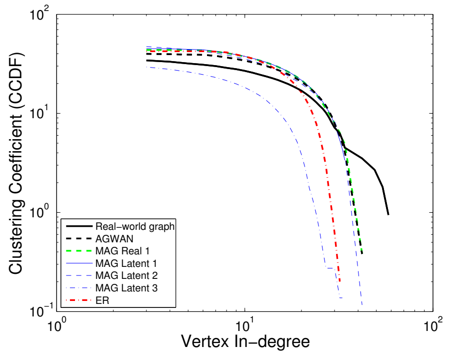


(a) Singular Values

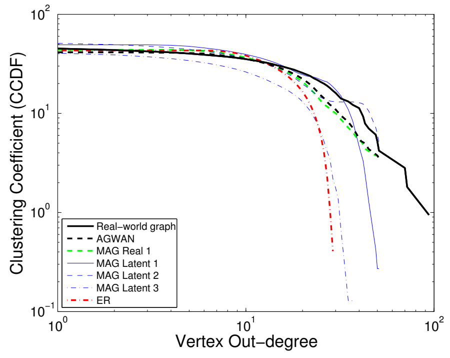


(b) Primary Left Singular Vector

Fig. 5: Spectral Properties

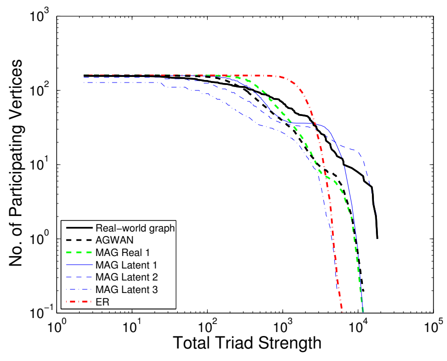


(a) In-edges

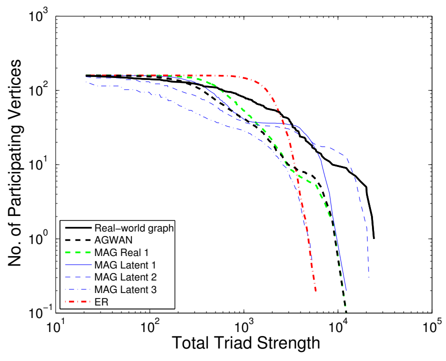


(b) Out-edges

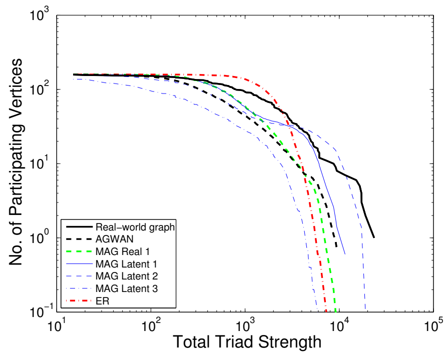
Fig. 6: Clustering Coefficients



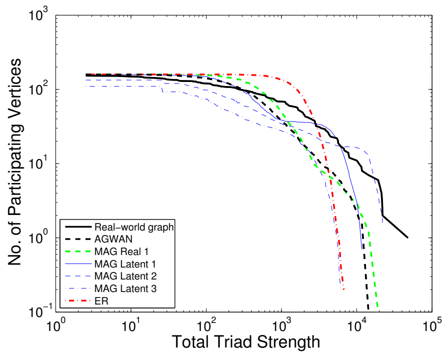
(a) Cycles



(b) Middlemen



(c) Ins



(d) Outs

Fig. 7: Triad Participation

	AGWAN	MAG-R1	MAG-L1	MAG-L2	MAG-L3	ER
In-Strength	**1.455**	4.700	4.942	4.942	5.438	2.469
Out-Strength	**2.303**	2.659	4.942	4.605	5.247	2.708
Singular Values	34.894	35.752	35.838	34.666	**30.049**	37.235
Singular Vector	**0.282**	1.801	1.691	1.460	1.520	1.915
Clustering Coefficient (In)	2.220	2.208	**0.348**	3.406	3.821	3.444
Clustering Coefficient (Out)	0.702	0.769	2.956	**0.411**	4.628	3.728
Triad Participation (Cycles)	3.555	4.248	4.248	**0.698**	4.174	4.787
Triad Participation (Middlemen)	4.500	4.500	3.807	**2.303**	3.584	4.382
Triad Participation (Ins)	**2.436**	4.500	2.457	2.996	5.075	4.700
Triad Participation (Outs)	4.248	4.094	2.436	**0.729**	4.443	4.382

Table 1: Kolmogorov-Smirnoff statistic for CDFs in Fig.s 4–7

	AGWAN	MAG-R1	MAG-L1	MAG-L2	MAG-L3	ER
In-Strength	**1.816**	4.912	4.015	4.744	10.452	5.679
Out-Strength	**2.117**	3.534	3.252	3.420	5.188	5.100
Singular Values	18.360	19.546	19.405	**17.931**	19.262	25.044
Singular Vector	**0.988**	7.587	7.609	6.973	7.273	7.316
Clustering Coefficient (In)	1.528	1.607	**0.958**	2.226	7.175	3.528
Clustering Coefficient (Out)	1.002	1.191	2.802	**0.690**	4.716	3.145
Triad Participation (Cycles)	3.101	3.000	2.053	**1.625**	5.180	3.823
Triad Participation (Middlemen)	4.207	4.178	**2.406**	2.465	6.592	5.144
Triad Participation (Ins)	4.332	4.826	**2.166**	2.524	8.922	4.630
Triad Participation (Outs)	3.203	3.295	1.819	**1.791**	4.864	3.727

Table 2: L2 statistic for CDFs in Fig.s 4–7

approach to the state-of-the-art in random graph generative algorithms. AGWAN was found to model vertex strength distributions and singular vectors much more closely than the other approaches.

The measures of clustering and triad participation are much better than random and better than MAG with real attributes, but not as good as MAG with two latent attributes. This suggests that edge weights make some contribution to clustering, but clustering cannot be accurately modelled with a single vertex label. The MAG latent approach can assign different structural processes (homophily, core-periphery, *etc.*) to each label, and the clustering property arises from the combination of labels. We propose to extend AGWAN to multiple vertex labels to investigate the effect on clustering.

As discussed in §3.3, MAG's method of combining multiple vertex attributes is unsatisfactory when applied to real attributes, due to the assumption of independence and the limit on the number of attributes which can be modelled. We have proposed a future line of research based on a weighted summation of the GMM for each edge. The fitting algorithm would need to regularise the individual contributions of each edge to take account of dependencies. The complexity of modelling the full joint distribution could be reduced with an approach based on Markov Random Fields or Factor Graphs.

References

1. Akoglu, L., McGlohon, M., Faloutsos, C.: OddBall: Spotting anomalies in weighted graphs. In: PAKDD 2010, Hyderabad, India (2010)
2. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science 286(5439), 509–512 (1999)
3. Blei, D.M., Jordan, M.I.: Variational inference for dirichlet process mixtures. Bayesian Analysis 1, 121–144 (2005)
4. Chakrabarti, D., Faloutsos, C.: Graph Mining: Laws, Tools, and Case Studies. Synthesis Lectures on Data Mining and Knowledge Discovery, Morgan & Claypool Publishers (2012)
5. Davis, M., Liu, W., Miller, P.: Finding the most descriptive substructures in graphs with discrete and numeric labels. In: NFMCP, pp. 138–154. Springer (2013)
6. Eichinger, F., Huber, M., Böhm, K.: On the usefulness of weight-based constraints in frequent subgraph mining. In: ICAI 2010. pp. 65–78. BCS SGAI (Dec 2010)
7. Erdős, P., Rényi, A.: On the evolution of random graphs. Publication of the Mathematical Institute of the Hungarian Academy of Sciences 5, 17–61 (1960)
8. Fagiolo, G.: Clustering in complex directed networks. Physical Review E 76(2) (Aug 2007)
9. Figueiredo, M.A., Jain, A.: Unsupervised learning of finite mixture models. Pattern Analysis and Machine Intelligence, IEEE Transactions on 24(3), 381–396 (2002)
10. Kim, M., Leskovec, J.: Modeling social networks with node attributes using the Multiplicative Attribute Graph model. In: UAI 2011, Barcelona, Spain. pp. 400–409 (2011)
11. Kim, M., Leskovec, J.: Multiplicative Attribute Graph model of real-world networks. Internet Mathematics 8(1–2), 113–160 (2012)
12. Kurihara, K., Welling, M., Vlassis, N.: Accelerated variational dirichlet process mixtures. In: NIPS (2006)
13. Leskovec, J., Chakrabarti, D., Kleinberg, J.M., Faloutsos, C., Ghahramani, Z.: Kronecker graphs: An approach to modeling networks. JMLR 11, 985–1042 (2010)
14. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graph evolution: Densification and shrinking diameters. ACM Trans. Knowl. Discov. Data 1(1) (Mar 2007)
15. Newman, M.: Networks: An Introduction. OUP, New York, NY, USA (2010)