1 Generating an candidate graph

We want to generate a set of candidate graphs (Gs) and see how likely each is to result in an observed X. The process for creating a potential graph involves creating a possible random walk Z of length m that results in X of length n, and simply drawing edges between each adjacent pair of nodes in Z; i.e., an edge between Z_{i-1} and Z_i for all i from $2...m$. Z_1 and Z_2 are constrained to X_1 and X_2, respectively. For $i = 3...m$, we set

$$Z_i = \begin{cases} \text{random element} \in \{X_1...X_{k-1}\} & \text{with probability } \theta \\ X_k & \text{with probability } 1-\theta \end{cases} \tag{1}$$

where k is the next unobserved element in X. This process is repeated until all elements in X have been observed, i.e. when $k = n$.

To construct a candidate graph from multiple X, simply construct a Z for each X using the same procedure.

2 Computing graph probability

Here is the original formulation for calculating graph probability from a single X:

$$P(G|X) = \sum_Z P(G, Z|X) \propto \sum_Z P(X|G, Z)P(Z|G)P(G) \tag{2}$$

Since X is deterministic from Z, the first term $P(X|G, Z)$ is either zero or one, i.e.:

$$P(X|G, Z) = 1_{\mathbb{Z}} Z \tag{3}$$

where $\mathbb{Z}$ is the set of all possible Zs on G that reduce to X and 1 is the indicator function. We enforce this by only only sampling over Zs that produce X. Currently, we are assuming all graphs are equally likely, and so $P(G)$ falls out. If you recall, I tried implementing a small-world prior that didn't help, but frankly I was not confident in my ability to implement this properly (i.e., how to calculate the prior probability of a particular graph with small-world coefficient S^{WS} (Humphries, 2008).

So the meat of the computation is computing $P(Z|G)$ as:

$$P(Z|G) = P(Z_1|G) \prod_{i=2}^{m} P(Z_i|Z_{1:i-1}, G) \tag{4}$$

I believe at the moment, $P(Z_1)$ is uniform and so since all graphs have the same number of nodes, this falls out. It should be the stationary probability of Z_1, but it still wouldn't matter for our purposes since the starting point is

fixed by X_1. To compute the likelihood of a graph with multiple X, we extend Equation 2 further:

$$P(G|X^1, X^2) \propto \sum_{Z^1 \in \mathbb{Z}^1} P(Z^1|G) \sum_{Z^2 \in \mathbb{Z}^2} P(Z^2|G) \tag{5}$$

and so forth for more X.

Since computing $P(Z|G)$ in Equation 4 typically results in underflow, we can approximate it using the log-sum-exp trick:

(6)

3 Generating a sample of Zs that result in X

Since the likelihood term contains an infinite sum over Z, we need to generate a representative sample of Zs to estimate this quantity. To generate a Z on graph G that results in X, we take a random walk on G starting from X_1, while constraining the walk such no path to X_j is taken before hitting X_i for all $j > i$. The walk is complete when we have a Z that results in X.

4 Incorporating inter-item response times

One problem with the approach above is that the procedure prefers sparse graphs. That is, it seems to prefer the *RW* solution used by Jun (submitted), which treats the censored list X as an uncensored list Z, and constructs a graph with edges between each pair of adjacent nodes. The intuitive reason for this is that the most sparse graph constructed from one X will result in shorter Z sequences on average,since the probability of directly transitioning between Z_i and Z_{i+1} is .5 for all recurrent nodes, the maximum possible for an undirected graph.

This problem is alleviated somewhat by using multiple X, but it does not appear to solve the problem entirely. By taking into account inter-item response times (IRTs), we can penalize short Zs for not conforming to the true pattern of IRTs. For example, in the most sparse graph constructed from one X, it is predicted that all IRTs have the same approximately normal distribution, but human data does not reflect this.

Let $I = \{I_1 ... I_{n-2}\}$ be a vector of observed inter-item response times. Let $H = \{H_1 ... H_{n-2}\}$ be a corresponding vector where each item denotes the number of hidden nodes between any two adjacent items (first hits) in X for a given Z. These vectors can instead be of length $n - 1$, however H_1 will always be 0 because there cannot be any hidden nodes between X_1 and X_2—therefore, its informative value is suspect. The vector I should correlate highly with the expected value of each item in H, i.e. $E(H) = \{E(H_1) ... E(H_{n-2})\}$.

I can think of at least two ways to approach this, though I am sure there are others and/or that these methods can be refined. I'll discuss both methods and then go over the pros and cons.

4.1 Method 1

Previously, we generated a sample of Zs on graph G that result in an observed X. From this sample, we also form a distribution of hidden nodes between any two adjacent nodes in X. That is, for any graph G we know not only $E(H)$ but also the full distribution of any H_i. Thus we can compute, for any two items $I_i < I_j$, the probability that G will generate a walk Z (that results in X) where $H_i < H_j$.

My formalism is not quite right, so I can't quite frame the full equation. However the gist is to estimate $P(I|G, X)$ as $P(ord(H) = ord(I)|G, X)$ where $ord(x)$ is the ordinal representation of x. This is done by:

$$P(I|G, X) = \prod_{i,j}^{n-2} P(H_i < H_j|G, X) \text{ for all } I_i < I_j \tag{7}$$

The notation needs some work but I think this makes sense conceptually.

4.2 Method 2

The second method is to look directly at the correlation between I and the H corresponding to a specific Z. The Z is more likely when the correlation indicates a better fit, i.e. when the standardized sum of the square residuals (SSR) is small. This is a little easier to formalize:

$$P(G|I, X) = \sum_{Z} P(I|Z, G, X)P(X|Z, G)P(Z|G)P(G) \tag{8}$$

Here, the new term that we are estimating using the SSR is $P(I|Z, G, X)$. Essentially, this term offsets the $P(Z|G)$ term so that we now don't necessarily prefer short Z sequences if the IRTs don't match I. How do we know what a good SSR is? That's tricky—one solution is to compare it to the SSRs generated from toy graphs, e.g. a subset of the animal semantic network with the same number of nodes. We can compute this beforehand and form a distribution of SSRs that are expected from reconstructing X.

4.3 Benefits and drawback of each method

There are several benefits to Method 1. One major benefit is that is does not require any sort of prior expectations about what the graph should look like. That is, it does not make use of the previously constructed animal semantic network. Another major benefit is that it can actually be solved analytically, rather that using random samples. Jun (submitted) shows in Theorem 3 that, given a graph G, it is possible to solve analytically for the probability of

transitioning between any X_i and X_{i+1} (Doyle & Snell, 1984). This allows computing $P(X|G)$ without generating a sample of Zs, which is by far the largest computational bottleneck in our procedure. A similar procedure can generate the expected length of a random walk before being absorbed. Thus, we can compute both $P(X|G)$ and $P(I|G)$ without doing expensive Monte Carlo simulations.

There are two primary drawbacks to using Method 1. First is that it considers only an ordinal representation of the IRTs. Whether the difference between two IRTs is 4s and 4.01s or 4s and 100s, the model will treat them the same. The other drawback is that it treats the IRTs as independent from the Zs that created them. That is, the model finds the likelihood of generating X on G and the likelihood of generating I on G, but the two are independent. I don't know how important this is.

The primary benefit of Method 2 is that it can use the raw IRT data, rather than simple ordinal comparisons, which may provide a better fit. Unlike Method 1, it also ties IRTs to the specific walk that created them. I suspect this is not a big factor, but I don't know for sure.

The drawbacks are that Method 2 relies on generating a computationally expensive sample of Zs as we have been doing, and that it relies on a prior animal semantic network to judge the likelihood. The good news, though, is that our simulations show the patten of IRTs is *roughly* similar for Erdos-Renyi and Watts-Strogatz graphs with equal number of nodes and edges. So even though we're making assumptions about the graph properties, I suspect the procedure will be fairly robust against it. Lastly, it's simply not as elegant; relying on SSRs is a bit of a hack, though it saves us from making assumptions about how hidden nodes map on to IRTs.

5 Estimating parameters of the graph

All of the above describes how to measure the probability of a random graph, but not how to optimize it (i.e., determine the optimal graph/link matrix). Using an analytical form of Method 1, we might be able to use the maximum likelihood solution used by Jun (submitted). To be honest, I didn't quite follow the procedure in Jun's section 2.2 for computing the optimal parameters, so I'm not entirely sure. The other solution, which we have partially implemented, is to use random mutations on an initial graph and follow some sort of simulated annealing heuristic to find the best fitting graph.

Side note: We can generate an initial graph using a single parameter θ, as before, though it may help to introduce an additional decay parameter λ which increases θ inverse-exponentially as a function of i in X. This may provide a better fit to the IRTs, and we could even compute the most efficient

starting parameters from training on the animal semantic network.