Supplementary Material: The Emergence of Organizing Structure in Conceptual Representation

Brenden M. Lake,[1,2] Neil D. Lawrence,[3] Joshua B. Tenenbaum,[4,5]

[1]Center for Data Science, New York University

[2]Department of Psychology, New York University

[3]Department of Computer Science, University of Sheffield

[4]Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology

[5]Center for Brains Minds and Machines

1 Generating data from structure

A data set is a matrix $D = \{f^{(1)}, ..., f^{(m)}\}$ ($D \in \mathbb{R}^{n_x \times m}$), where the rows correspond to objects and the columns are features $f^{(i)}$. A data set is generated by a structure, parameterized by a symmetric matrix $S \in \mathbb{R}^{n_t \times n_t}$. Each row/column in S corresponds to a node in the graph. The set of object nodes is called X and cluster nodes is called Z, where n_x and n_z are their cardinalities, combining for a total of $n_t = n_x + n_z$ nodes in the graph.

A key property of the structural sparsity model is that S is sparse, meaning that most of its entries are equal to zero ($s_{ij} = 0$). For the remainder, their values are positive ($s_{ij} = s_{ji} > 0$). The sparsity pattern defines the adjacency matrix of the undirected graph, where a non-zero value for s_{ij} means that nodes i and j share an edge. Given that each object node $x \in X$ connects to only one cluster node, then $s_{xz} > 0$ for exactly one $z \in Z$. There are no self-edges, so the diagonal of S is also zero.

Following Zhu, Lafferty, and Ghahramani (2003) and the setup for the structural forms model (Kemp & Tenenbaum, 2008), we introduce the graph Laplacian Δ. Its off-diagonal elements are

given by $-S$ and its diagonal elements are

$$\Delta_{ii} = \sum_j s_{ij}.$$

A generative model for features can then be written as

$$\begin{aligned} P(f^{(k)}|S) &\propto \exp\Big(-\frac{1}{4}\sum_{i,j} s_{ij}(f_i^{(k)} - f_j^{(k)})^2\Big) \\ &= \exp\Big(-\frac{1}{2} f^{(k)\top} \Delta f^{(k)}\Big). \end{aligned}$$

This equation highlights why the model favors features that are smooth across the graph. Features are probable when connected objects i and j ($s_{ij} > 0$) have a similar value. The stronger the connection s_{ij} (meaning the larger its value), the more important it is for the feature values to match. As pointed out in Zhu et al. (2003), this distribution is not proper, since adding a constant value to the feature vector does not change its probability. Therefore following Zhu et al. (2003), we define the matrix $J = \Delta + \frac{1}{\sigma^2} I$ and use

$$P(f^{(k)}|S, \sigma^2) \propto \exp\Big(-\frac{1}{2} f^{(k)\top} J f^{(k)}\Big), \tag{1}$$

which results in a proper density. This distribution is an n_t dimensional Gaussian with zero mean

$$\begin{aligned} P(f^{(k)}|S, \sigma^2) &= N\Big(f^{(k)}|0, J^{-1}\Big) \\ &= \frac{1}{(2\pi)^{n_t/2}|J|^{-1/2}} \exp\Big(-\frac{1}{2} f^{(k)\top} J f^{(k)}\Big). \end{aligned}$$

This generative model for features can also be derived from a maximum entropy formulation, as shown in Lawrence (2011, 2012). This distribution over features is nearly the same as in the structural forms model, except the forms model adds the diagonal term $\frac{1}{\sigma^2}$ only to the observed nodes.

This distribution is also known as a Gaussian Markov Random Field (Koller & Friedman, 2009). The undirected graph S (and equivalently the precision matrix J) instantiates a set of conditional

independence relationships between the variables in the graph

$$s_{ij} = 0 \text{ if and only if } \Big(f_i^{(k)} \perp f_j^{(k)} | f_{\setminus\{i,j\}}^{(k)} \Big).$$

If nodes i and j do not share an edge, their feature values are conditionally independent when the rest of the feature vector is observed (or equivalently, their partial correlation is 0 when controlling for all other nodes). Intuitively, a shared edge is a path of direct dependence.

Given that each object connects to exactly one cluster node (and no other nodes), there are no paths of direct dependence between objects. Instead, covariation between objects is represented through the connections between their cluster nodes. Moreover, two objects assigned to the same cluster node are conditionally independent, if the feature values of their shared cluster node could be observed.

2 Prior distribution on structure

The prior on structures S has a very simple form

$$P(S, \sigma^2) \propto \exp\Big(-\beta(\#S) \Big),$$

where $\#S$ is the number of edges in a structure. Equivalently, $\#S = \frac{1}{2}||S||_0$, where $||S||_0$ known as the ℓ_0-norm, counts the number of non-zero entires in a matrix. The prior has support on the set of all possible graphs in which the following conditions are met: each object has only one connection (which is to a cluster node), there are no empty cluster nodes, each matrix entry is non-negative ($s_{ij} \geq 0$), and $\sigma^2 > 0$.

This prior is improper because the values in S have no upper bound. Given that this paper aims to just find a single good structure, this issue can be ignored and the prior becomes a penalty $-\beta(\#S)$ in the model's log-score (Main article; Eq. 3). If it was desirable (and tractable) to compute a posterior distribution $P(S|D)$, then this improper prior could be extended to form a proper prior. Following the structural forms model, S could be decomposed into its sparsity pattern S_{pat} (which values are non-zeros) and the edge values S_{val}. The forms model places a prior on each entry in S_{val}, and uses Laplace's method (MacKay, 2003) to approximate the integral over the

parameter values

$$P(D|S_{pat}) = \int P(D|S_{pat}, S_{val}, \sigma^2) P(S_{val}|S_{pat}) P(\sigma^2) dS_{val} d\sigma^2.$$

This strategy is entirely consistent with the structural sparsity model. In fact, approximating the integral would create an additional force for sparsity, known as the Bayesian Occam's razor (MacKay, 2003). But our current approach was chosen for simplicity, since we found it unnecessary to have two simultaneous forces driving for sparsity. There are some more formal reasons to support this point. As the number of features grows, the Laplace approximation for the integral above asymptotes to a model score known as the Bayesian Information Criterion (BIC) (Koller & Friedman, 2009; Schwarz, 1978). The BIC score maximizes the likelihood while penalizing the number of parameters, just like the penalization arising from our prior on S, although it differs in a scaling factor. Thus the more complex model, which combines these two sources of sparsity, asymptotes to the simpler model with just a larger value of the sparsity parameter β, soaking up both forces that promote sparsity.

3 Model implementation

Computing a posterior distribution over structures $P(S|D)$ is very difficult, and thus we aim to find a single good structure S that maximizes the posterior score (Main article; Eq. 3). Our approach to this optimization problem consists of two main routines. The outer-routine searches for the best clustering pattern (partition) of objects into cluster nodes, called *cluster search*. The inner-routine searches for the best graph S without changing the clustering pattern, called *connection search*.

3.1 Cluster search

The strategy for cluster search is related to standard methods for structure learning in graphical models (Koller & Friedman, 2009), which evaluate small changes to a graph and often greedily select the best change. Given a hypothesis structure and its cluster pattern, cluster search considers a set of local proposals to split or merge cluster nodes. The best scoring move, whether it is a split or a merge, is chosen greedily at each step. In order to assess the quality of each move, the edges must be re-optimized with the connection search routine which is computationally expensive. But

multiple possible moves can be evaluated in parallel. To help mitigate the problem of local optima, search does not terminate immediately when the best local move decreases the score. Instead, the algorithm terminates after the score decreases several times (we used 5). Search also keeps a tabu list of previously visited cluster partitions that it does not revisit. The best evaluated structure S is returned as the solution.

Initialization. Choosing a good initialization can save a lot of computation, and in some cases, lead to better results. Given that cluster search operates over cluster partitions, we use standard clustering methods to propose a set of reasonable candidates for initialization. We use k-means clustering to choose a set of k clusters. Multiple values of k are tried, where the values of k are evenly spaced on a logarithmic scale from 1 to the number of objects n_x. Each candidate is evaluated and scored with connection search. After determining the best value for k in this initial coarse sampling, the algorithm attempts to narrow in on a better value. Picking the closest, previously-tried values of k above and below the current k, k is further optimized in this range by Fibonacci search. Assuming the score is a unimodal function of k within these bounds, Fibonacci search will find the optimum in this range.

Splitting clusters. To split a cluster node, the objects currently assigned to that node must be divided between the two new clusters. Following the general splitting strategy used in the structural forms algorithm (Kemp & Tenenbaum, 2008), the sparsity algorithm chooses two seed objects at random, assigns one to each cluster, and stochastically distributes the remaining objects by picking whichever seed object is closer in feature space. There is also a low probability of choosing the opposite seed object. Rather than evaluating just one split per node, the algorithm tries several (3) randomly chosen seed objects. Each split must then be optimized with connection search and scored by the objective function (Main article; Eq. 3). For each step during search, the algorithm is limited in the total number of splits it will consider across all of the cluster nodes (30 for most data sets, but 8 for large data sets like the Senate and 200 Objects).

Merging clusters. To merge two cluster nodes, they must combine their attached objects to form a single cluster node. Rather than trying every combination of merges, each cluster node stochastically, but intelligently, picks another to combine with. To help select the merges, the algorithm first calculates the expected value of the latent features for all cluster nodes, and the probability of merging two nodes decreases with the distances in this feature space. The algorithm

also limits the number of merges it evaluates (30 for most data sets, but 8 for the Senate and 200 objects).

Swapping assignments. In addition to splits and merges, the algorithm also tries to swap the cluster assignments of objects (Kemp & Tenenbaum, 2008). These moves do not compete with splitting and merging during each greedy step of the algorithm. Instead, at regular intervals throughout search (we used every 3 moves), each object tries to change its cluster assignment while leaving all others fixed. It tries all possibilities, but it does not re-learn the sparsity pattern for each possible assignment, which is an expensive computation. Instead, it just re-optimizes the existing edge strengths of each candidate proposal S. If a new parent leads to a better score, then S is re-optimized with the full connection search.

3.2 Connection search

Connection search is the sub-routine that searches for the best sparse connectivity pattern, given a assignment of objects to cluster nodes. Defining a function $c(S)$ that extracts the cluster assignment, connection search must solve

$$\underset{\sigma^2,\{S:c(S)=w\}}{\operatorname{argmax}} \sum_{i=1}^{m} \log P(f_X^{(i)}|S,\sigma^2) - \beta(\#S), \tag{2}$$

where the current cluster assignment is denoted as w. Given that features are only observed for object nodes, and even those features can be missing, we use the Structural Expectation-Maximization algorithm (Structural EM or SEM) for learning in the presence of missing data (Friedman, 1997).

Structural Expectation-Maximization. SEM reduces the missing data problem to a sequence of simpler structure learning problems with complete data. Rather than maximizing Eq. 2 directly, the structure at iteration $r+1$ of SEM maximizes

$$S^{[r+1]},\sigma^{2[r+1]} \leftarrow \underset{\sigma^2,\{S:c(S)=w\}}{\operatorname{argmax}} \sum_{i=1}^{m} E_{Q_i(f_Z^{(i)},S^{[r]},\sigma^{2[r]})}\Big[\log P(f_X^{(i)},f_Z^{(i)}|S,\sigma^2)\Big] - \beta(\#S), \tag{3}$$

where the expectation is over the conditional distribution of the missing or latent features ($f_Z^{(i)}$)

given the observed features ($f_X^{(i)}$) and the current structure at iteration r,

$$Q_i(f_Z^{(i)}, S^{[r]}, \sigma^{2[r]}) = P(f_Z^{(i)} | f_X^{(i)}, S^{[r]}, \sigma^{2[r]}).$$

Each iteration of the Structural EM algorithm is guaranteed to improve the original objective, or the marginal probability of the observed features Eq. 2. For the structural sparsity model, each iteration can be decomposed into an E-step (Step 3 below) which computes expected sufficient statistics. This is followed by a Structural M-step that re-optimizes the structure to fit the new sufficient statistics (Step 4). The algorithm is shown below.

1: Initialize $S^{[0]}$ and $\sigma^{2[0]}$

2: **for** $r = 0, 1, 2, ...$ until convergence **do**

3: $\quad H \leftarrow \frac{1}{m} \sum_{i=1}^{m} E_{Q_i(f_Z^{(i)}, S^{[r]}, \sigma^{2[r]})}[f^{(i)} f^{(i)T}]$

4: $\quad \{S^{[r+1]}, \sigma^{2[r+1]}\} \leftarrow \underset{\sigma^2, \{S: c(S)=w\}}{\operatorname{argmax}} \log |J| - \operatorname{tr}(HJ) - \frac{\beta}{m} ||S||_0$

5: **end for**

The precision matrix J is implicitly a function of S and σ^2. Step 4 is intractable, as is the case with most structure learning problems, and we describe how we approximately solve it in the next section. Also, we run traditional EM to convergence at the end of each iteration r of Structural EM (see Alternating SEM-EM Friedman, 1997). Traditional EM is the same algorithm as SEM but with $\beta = 0$ and a fixed sparsity pattern for S across iterations.

Optimization with ℓ_1 heuristic. Exactly solving Step 4 is intractable, since all possible sparsity patterns need to be considered and the number of different sparsity patterns for k clusters is $2^{(k^2-k)/2}$. Instead we use a convex relaxation of the optimization which replaces the ℓ_0 norm with the ℓ_1 norm. This heuristic leads to a sparse solution (Banerjee, El Ghaoui, & D'Aspremont, 2008). The ℓ_1 relaxation penalizes a sum of the parameters instead of their cardinality, such that $||S||_0 \approx \lambda \sum_{ij} s_{ij}$ for some constant λ. This leads to a convex optimization problem which can be

solved exactly (Lake & Tenenbaum, 2010):

$$\underset{J,S,\sigma^2}{\text{argmax}} \log |J| - \text{tr}(HJ) - \frac{\beta\lambda}{m}\sum_{ij} s_{ij}$$

subject to

$$J = \text{diag}\Big(\sum_j s_{ij}\Big) - S + \frac{1}{\sigma^2}I$$

$$s_{ij} = 0,\ \{i,j\} \notin \mathcal{E}$$

$$s_{ij} \geq 0,\ \{i,j\} \in \mathcal{E}$$

$$\sigma^2 > 0.$$

The set of allowable edges is denoted as $\mathcal{E}$. Given that J can be defined as an affine mapping from the variables S and σ^2, we can make the first equality constraint implicit in the objective function and remove J as a variable (Boyd & Vandenberghem, 2004). By setting the missing edges to be zero and removing them as variables, we have a convex optimization problem with just a positivity constraint on the parameters. Efficient methods for solving a related sparse Gaussian problem with ℓ_1 have been developed (Banerjee et al., 2008; Schmidt, Van Den Berg, Friedlander, & Murphy, 2009), although this problem does not have a positivity constraint or the graph Laplacian interpretation. Algorithms for this related problem typically solve the dual optimization problem, although we found that solving the primal problem was more efficient for the variant defined here. To solve it, we used a projected quasi-Newton algorithm and code developed by Schmidt et al. (2009). For a given solution, many values are zero but others are just small. The final sparsity pattern is chosen by thresholding to approximately maximize the complete-data objective in Step 4 of the Structural EM algorithm. The algorithm repeats the optimization and thresholding a total of three times, at $\lambda = \{.5, 1, 2\}$, and the best is picked according to the same objective. For the largest data sets, $\lambda = 1$.

Derivation of Structural EM. More details of the Structural EM derivation are now described. We unpack the SEM objective function in Eq. 3 and derive the algorithm in the section above. Here, the distribution Q_i implicitly depends on the current hypothesis S, although we drop the dependence in the notation.

$$
\begin{aligned}
&\underset{\sigma^2,\{S:c(S)=w\}}{\operatorname{argmax}} \log P(S,\sigma^2) + \textstyle\sum_{i=1}^{m} E_{Q_i(f_Z^{(i)})}[\log\, p(f_X^{(i)}, f_Z^{(i)}; S, \sigma^2)] \\
&= \underset{\sigma^2,\{S:c(S)=w\}}{\operatorname{argmax}} \log P(S,\sigma^2) - \tfrac{m}{2}\log\,|J^{-1}|\; - \tfrac{1}{2}\textstyle\sum_{i=1}^{m} E_{Q_i(f_Z^{(i)})}[f^{(i)T} J f^{(i)}] \\
&= \log P(S,\sigma^2) + \tfrac{m}{2}\log\,|J| - \tfrac{1}{2}\textstyle\sum_{i=1}^{m}(\sum_{j=1}^{n}\sum_{k=1}^{n} E_{Q_i(f_Z^{(i)})}[f_j^{(i)} f_k^{(i)}] J_{jk}) \\
&= \log P(S,\sigma^2) + \tfrac{m}{2}\log\,|J|\; - \tfrac{1}{2}\textstyle\sum_{i=1}^{m} \operatorname{tr}(E_{Q_i(f_Z^{(i)})}[f^{(i)} f^{(i)T}] J) \\
&= \log P(S,\sigma^2) + \tfrac{m}{2}\log\,|J| - \tfrac{m}{2}\operatorname{tr}(HJ) \\
&\propto \log\,|J| - \operatorname{tr}(HJ) - \tfrac{\beta}{m}||S||_0
\end{aligned}
$$

Above, J is the precision matrix (Eq. 1) and H is the expectation of the empirical covariance,

$$H \leftarrow \frac{1}{m}\sum_{i=1}^{m} E_{Q_i(f_Z^{(i)})}[f^{(i)} f^{(i)\top}].$$

The E-step (Step 3 in the SEM algorithm) uses this formula for computing H. Afterwards, we can approximately solve the optimization problem shown in the last line of the derivation, which forms the M-step (Step 4 in the SEM algorithm).

References

Banerjee, O., El Ghaoui, L., & D'Aspremont, A. (2008). Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data. *Journal of Machine Learning Research*, *9*, 485–516.

Boyd, S., & Vandenberghem, L. (2004). *Convex Optimization*. Cambridge University Press.

Friedman, N. (1997). Learning Belief Networks in the Presence of Missing Values and Hidden Variables. *Fourteenth International Conference on Machine Learning (ICML)*.

Kemp, C., & Tenenbaum, J. B. (2008). The discovery of structural form. *Proceedings of the National Academy of Sciences*, *105*(31), 10687–92.

Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models*. MIT Press.

Lake, B. M., & Tenenbaum, J. B. (2010). Discovering Structure by Learning Sparse Graphs. *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*.

Lawrence, N. D. (2011). Spectral Dimensionality Reduction via Maximum Entropy. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 51–59.

Lawrence, N. D. (2012). A Unifying Probabilistic Perspective for Spectral Dimensionality Reduction: Insights and New Models. *Journal of Machine Learning Research*, *13*, 1609–1638.

MacKay, D. J. (2003). *Information Theory, Inference, and Learning Algorithms*. Cambridge, UK: Cambridge University Press.

Schmidt, M., Van Den Berg, E., Friedlander, M. P., & Murphy, K. (2009). Optimizing Costly Functions with Simple Constraints: A Limited-Memory Projected Quasi-Newton Algorithm. *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Schwarz, G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, *6*(2), 461–464.

Zhu, X., Lafferty, J., & Ghahramani, Z. (2003). *Semi-supervised learning: From Gaussian fields to Gaussian processes* (Tech. Rep. No. CMU-CS-03-175).