Science Advances
AAAS

advances.sciencemag.org/cgi/content/full/4/7/eaar8260/DC1

Supplementary Materials for

A physical model for efficient ranking in networks

Caterina De Bacco*, Daniel B. Larremore*, Cristopher Moore*

*Corresponding author. Email: cdebacco@santafe.edu (C.D.B.); daniel.larremore@colorado.edu (D.B.L.); moore@santafe.edu (C.M.)

Published 20 July 2018, *Sci. Adv.* 4, eaar8260 (2018)
DOI: 10.1126/sciadv.aar8260

This PDF file includes:

Supporting Information (SI)

Section S1. Deriving the linear system minimizing the Hamiltonian

The SpringRank Hamiltonian Eq. (2) is convex in s and we set its gradient $\nabla H(s) = 0$ to obtain the global minimum

$$\frac{\partial H}{\partial s_i} = \sum_j \left[A_{ij}\left(s_i - s_j - 1\right) - A_{ji}\left(s_j - s_i - 1\right)\right] = 0 \tag{S1}$$

Let the weighted out-degree and in-degree be $d_i^{\text{out}} = \sum_j A_{ij}$ and $d_i^{\text{in}} = \sum_j A_{ji}$, respectively. Then Eq. (S1) can be written as

$$\left(d_i^{\text{out}} + d_i^{\text{in}}\right) s_i - \left(d_i^{\text{out}} - d_i^{\text{in}}\right) - \sum_j \left[A_{ij} + A_{ji}\right] s_j = 0 \tag{S2}$$

We now write the system of N equations together by introducing the following matrix notation. Let $D^{\text{out}} = \text{diag}(d_1^{\text{out}}, \ldots, d_N^{\text{out}})$ and $D^{\text{in}} = \text{diag}(d_1^{\text{in}}, \ldots, d_N^{\text{in}})$ be diagonal matrices, let $\mathbf{1}$ be the N-dimensional vector of all ones. Then Eq. (S2) becomes

$$\left[D^{\text{out}} + D^{\text{in}} - \left(A + A^T\right)\right] s = \left[D^{\text{out}} - D^{\text{in}}\right] \mathbf{1} \tag{S3}$$

This is a linear system of the type $B\,s = b$, where $B = \left[D^{\text{out}} + D^{\text{in}} - \left(A + A^T\right)\right]$ and $b = \left[D^{\text{out}} - D^{\text{in}}\right] \mathbf{1}$. The rank of B is at most $N-1$ and more generally, if the network represented by A consists of C disconnected components, B will have rank $N - C$. In fact, B has an eigenvalue 0 with multiplicity C, and the eigenvector $\mathbf{1}$ is in the nullspace. B is not invertible, but we can only invert in the $N - C$-dimensional subspace orthogonal to the nullspace of B. The family of translation-invariant solutions s^* is therefore defined by

$$s^* = \left[D^{\text{out}} + D^{\text{in}} - \left(A + A^T\right)\right]^{-1} \left[D^{\text{out}} - D^{\text{in}}\right] \mathbf{1} \tag{S4}$$

in which the notation $[\cdot]^{-1}$ should be taken as the Moore-Penrose pseudoinverse.

In practice, rather than constructing the pseudo-inverse, it will be more computationally efficient (and for large systems, more accurate) to solve the linear system in an iterative fashion. Since we know that solutions may be translated up or down by an arbitrary constant, the system can be made full-rank by fixing the position of an arbitrary node 0. Without loss of generality, let $s_N = 0$. In this case, terms that involve s_N can be dropped from Eq. (S2), yielding

$$\left(d_i^{\text{out}} + d_i^{\text{in}}\right) s_i - \left(d_i^{\text{out}} - d_i^{\text{in}}\right) - \sum_{j=1}^{N-1} \left[A_{ij} + A_{ji}\right] s_j = 0\,, \qquad i \neq N \tag{S5}$$

$$- \left(d_N^{\text{out}} - d_N^{\text{in}}\right) - \sum_{j=1}^{N-1} \left[A_{Nj} + A_{jN}\right] s_j = 0 \tag{S6}$$

Adding Eq. (S5) to Eq. (S6) yields

$$\left(d_i^{\text{out}} + d_i^{\text{in}}\right) s_i - \left(d_i^{\text{out}} + d_N^{\text{out}} - d_i^{\text{in}} - d_N^{\text{in}}\right) - \sum_{j=1}^{N-1} \left[A_{ij} + A_{Nj} + A_{ji} + A_{jN}\right] s_j = 0 \tag{S7}$$

which can be written in matrix notation as

$$\left[D^{\text{out}} + D^{\text{in}} - \mathring{A}\right] s = \left[D^{\text{out}} - D^{\text{in}}\right] \mathbf{1} + \left(d_N^{\text{out}} - d_N^{\text{in}}\right) \mathbf{1} \tag{S8}$$

where

$$\mathring{A}_{ij} = A_{ij} + A_{Nj} + A_{ji} + A_{jN} \tag{S9}$$

In this formulation, Eq. (S8) can be solved to arbitrary precision using iterative methods that take advantage of the sparsity of $\mathring{A}$. The resulting solution may then be translated by an arbitrary amount as desired.

Section S2. Poisson generative model

The expected number of edges from node i to node j is $c\exp\left[-\frac{\beta}{2}(s_i - s_j - 1)^2\right]$ and therefore the likelihood of observing a network A, given parameters β, s, and c is

$$P(A \,|\, s,\beta,c\,) = \prod_{i,j} \frac{\left[c\ \mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2}\right]^{A_{ij}}}{A_{ij}!}\ \exp\left[-c\,\mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2}\right] \tag{S10}$$

Taking logs yields

$$\log P(A \,|\, s,\beta,c\,) = \sum_{i,j} A_{ij}\log c - \frac{\beta}{2}A_{ij}\,(s_i - s_j - 1)^2 - \log\left[A_{ij}!\right] - c\mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2} \tag{S11}$$

Discarding the constant term $\log\left[A_{ij}!\right]$, and recognizing the appearance of the SpringRank Hamiltonian $H(s)$, yields

$$\mathcal{L}(A \,|\, s,\beta,c\,) = -\beta H(s) + \sum_{i,j} A_{ij}\log c - \sum_{i,j} c\mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2} \tag{S12}$$

Taking $\partial\mathcal{L}/\partial c$ and setting it equal to zero yields

$$\hat{c} = \frac{\sum_{i,j} A_{ij}}{\sum_{i,j} \mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2}} \tag{S13}$$

which has the straightforward interpretation of being the ratio between the number of observed edges and the expected number of edges created in the generative process for $c = 1$. Substituting in this solution and letting $M = \sum_{i,j} A_{ij}$ yields

$$\begin{aligned}\mathcal{L}(A \,|\, s,\beta\,) &= -\beta H(s) + M\log\hat{c} - \sum_{i,j}\hat{c}\,\mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2} \\ &= -\beta H(s) + M\log M - M\log\left[\sum_{i,j}\mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2}\right] - M\end{aligned} \tag{S14}$$

The terms $M\log M$ and M may be neglected since they do not depend on the parameters, and we divide by β, yielding a log-likelihood of

$$\mathcal{L}(A \,|\, s,\beta\,) = -H(s) - \frac{M}{\beta}\log\left[\sum_{i,j}\mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2}\right] \tag{S15}$$

Note that the SpringRank Hamiltonian may be rewritten as $H(s) = M\left[\frac{1}{2}\langle A_{ij}(s_i - s_j - 1)^2\rangle_E\right]$ where $\langle\cdot\rangle_E$ denotes the average over elements in the edge set E. In other words, $H(s)$ scales with M and the square of the average spring length. This substitution for $H(s)$ allows us to analyze the behavior of the log-likelihood

$$\mathcal{L}(A \,|\, s,\beta\,) = -M\left\{\frac{1}{2}\left\langle A_{ij}\ (s_i - s_j - 1)^2\right\rangle_E + \frac{1}{\beta}\log\left[\sum_{i,j}\mathrm{e}^{-\frac{\beta}{2}(s_i - s_j - 1)^2}\right]\right\} \tag{S16}$$

Inside the logarithm there are N^2 terms of finite value, so that the logarithm term is of order $O(\frac{\log N}{\beta})$. Thus, for well-resolved hierarchies, i.e. when β is large enough that the sampled edges consistently agree with the score difference between nodes, the maximum likelihood ranks $\hat{s}$ approach the ranks s^* found by minimizing the Hamiltonian. In practice, exactly maximizing the likelihood would require extensive computation, e.g. by using local search heuristic or Markov chain Monte Carlo sampling.

Section S3. Rewriting the energy

The Hamiltonian Eq. (2) can be rewritten as

$$\begin{aligned}
2\,H(s) &= \sum_{i,j=1}^{N} A_{ij}\,\left(s_i - s_j - 1\right)^2 \\
&= \sum_{i,j=1}^{N} A_{ij}\,\left(s_i^2 + s_j^2 - 2s_i s_j + 1 - 2s_i + 2s_j\right) \\
&= \sum_{i}^{N} s_i^2 \sum_{j=1}^{N} A_{ij} + \sum_{j}^{N} s_j^2 \sum_{i=1}^{N} A_{ij} - 2\sum_{i} s_i \sum_{j} A_{ij} s_j + M \\
&\quad -2\sum_{i=1}^{N} s_i \sum_{j=1}^{N} A_{ij} + 2\sum_{j=1}^{N} s_j \sum_{i=1}^{N} A_{ij} \\
&= \sum_{i}^{N} s_i^2\left(d_i^{\text{out}} + d_i^{\text{in}}\right) - 2\sum_{i} s_i\left(d_i^{\text{out}} - d_i^{\text{in}}\right) + M - 2\sum_{i} s_i \sum_{j} A_{ij} s_j
\end{aligned} \tag{S17}$$

From Eq. (S2) we have

$$\sum_{j} s_i^2\left(d_i^{\text{out}} + d_i^{\text{in}}\right) - \sum_{i} s_i\left(d_i^{\text{out}} - d_i^{\text{in}}\right) = \sum_{i} s_i \sum_{j}\left[A_{ij} + A_{ji}\right] s_j \tag{S18}$$

We can substitute this into Eq. (S17)

$$\begin{aligned}
2H(s) &= \sum_{i} s_i \sum_{j}\left[A_{ij} + A_{ji}\right] s_j - \sum_{i} s_i\left(d_i^{\text{out}} - d_i^{\text{in}}\right) + M - 2\sum_{i} s_i \sum_{j} A_{ij} s_j \\
&= \sum_{i} s_i\left(d_i^{\text{in}} - d_i^{\text{out}}\right) + M \\
&= \sum_{i} h_i s_i + M
\end{aligned} \tag{S19}$$

where $h_i \equiv d_i^{\text{in}} - d_i^{\text{out}}$.

Section S4. Ranks distributed as a multivariate Gaussian distribution

Assuming that the ranks are random variables distributed as a multivariate Gaussian distribution of average $\bar{s}$ and covariance matrixΣ, we have

$$P(s) \propto \exp\left(-\frac{1}{2}(s-\bar{s})^{\intercal}\Sigma^{-1}(s-\bar{s})\right) \tag{S20}$$

We can obtain this formulation by considering a Boltzman distribution with the Hamiltonian Eq. (2) as the energy term and inverse temperature β so that

$$P(s) \propto \exp\left(-\frac{\beta}{2}\sum_{i,j=1}^{N} A_{ij}\left(s_i - s_j - 1\right)^2\right) \tag{S21}$$

Manipulating the exponent of Eq. (S20) yields

$$\frac{1}{2}(s-\bar{s})^T\Sigma^{-1}(s-\bar{s}) = \frac{1}{2}\left(s^T\Sigma^{-1}s - 2s^T\Sigma^{-1}\bar{s} + \bar{s}^T\Sigma^{-1}\bar{s}\right) \tag{S22}$$

whereas the parallel manipulation of Eq. (S21) yields

$$\frac{\beta}{2}\sum_{i,j=1}^{N} A_{ij}\left(s_i - s_j - 1\right)^2 = \frac{\beta}{2}\left[s^T\left(D^{\text{out}} + D^{\text{in}} - A^T - A\right)s + 2\;s^T(D^{\text{in}} - D^{\text{out}})\mathbf{1} + M\right] \tag{S23}$$

where $\mathbf{1}$ is a vector of ones and D^{in} are diagonal matrices whose entries are the in- and out-degrees, $D^{\text{out}}_{ii} = \sum_j A_{ij}$ and $D^{\text{in}}_{ii} = \sum_j A_{ji}$ and $M = \sum_{i,j} A_{ij}$. Comparing these last two expressions and removing terms that do not depend on s because irrelevant when accounting for normalization, we obtain

$$\Sigma = \tfrac{1}{\beta}\left(D^{\text{out}} + D^{\text{in}} - A^T - A\right)^{-1} \quad \text{and} \quad \bar{s} = \beta\,\Sigma\left(D^{\text{out}} - D^{\text{in}}\right)\mathbf{1} = s^* \tag{S24}$$

Section S5. Bayesian SpringRank

Adopting a Bayesian approach with a factorized Gaussian prior for the ranks, we obtain that the s that maximizes the posterior distribution is the one that minimizes the regularized SpringRank Hamiltonian Eq. (4), i.e. the s that solves the linear system Eq. (5). In fact, defining

$$P(s) = Z^{-1}(\beta,\alpha)\prod_{i\in V} \mathrm{e}^{-\beta\,\frac{\alpha}{2}(s_i-1)^2} = Z^{-1}(\beta,\alpha)\prod_{i\in V} \mathrm{e}^{-\beta\,\alpha\,H_0(s_i)} \tag{S25}$$

where $Z(\beta,\alpha) = \left[\frac{2\pi}{\beta\,\alpha}\right]^{N/2}$ is a normalization constant that depends on α and β, and following the same steps as before we get

$$\begin{aligned}\log P(s\,|\,A) &= \sum_{i,j}\log P(A_{ij}\,|\,s) - \beta\alpha\sum_{i\in V} H_0(s_i) + \log\left(Z\left(\beta,\alpha\right)\right)\\ &= -\,\beta\left[H(s) + \alpha\sum_{i\in V} H_0(s_i)\right] + C\end{aligned} \tag{S26}$$

where C is a constant that does not depend on the parameters, and thus may be ignored when maximizing $\log P(s\,|\,A)$.

Section S6. Fixing c to control for sparsity

The parameter c included in the generative model (7) controls for network's sparsity. We can indeed fix it so to obtain a network with a desired expected number of edges $\langle M\rangle$ as follows

$$\langle M\rangle \equiv \sum_{i,j}\langle A_{ij}\rangle = c\sum_{i,j}\mathrm{e}^{-\frac{\beta}{2}(s_i-s_j-1)^2}$$

For a given vector of ranks s and inverse temperature β, the c realizing the desired sparsity will then be

$$c = \frac{\langle M\rangle}{\sum_{i,j}\mathrm{e}^{-\frac{\beta}{2}(s_i-s_j-1)^2}} = \frac{\langle k\rangle N}{\sum_{i,j}\mathrm{e}^{-\frac{\beta}{2}\,(s_i-s_j-1)^2}} \tag{S27}$$

where $\langle k\rangle$ is the expected node degree $\langle k\rangle = \sum_{i=1}^{N}\left[d^{\text{in}}_i + d^{\text{out}}_i\right]$. Similar arguments apply when considering a generative model with Bernoulli distribution.

Section S7. Comparing optimal β for predicting edge directions

In the main text, (12) and Eq. (13) define the accuracy of edge prediction, in terms of the number of edges predicted correctly in each direction and the log-likelihood conditioned on the undirected graph. Here we compute the optimal values of β for both notions of accuracy. In both computations that follow, the following two facts will be used

$$P'_{ij}(\beta) = 2\,(s_i - s_j)\;\mathrm{e}^{-2\beta(s_i-s_j)}\,P^2_{ij}(\beta) \tag{S28}$$

and

$$1 = \frac{P_{ij}(\beta)}{1-P_{ij}(\beta)}\,\mathrm{e}^{-2\beta(s_i-s_j)} \tag{S29}$$

A. Choosing β to optimize edge direction accuracy

We take the derivative of Eq. (12) with respect to β, set it equal to zero, and solve as follows

$$0 \equiv \frac{\partial \sigma_a(\beta)}{\partial \beta} = \frac{\partial}{\partial \beta}\left[1 - \frac{1}{2m}\sum_{i,j}\left|A_{ij} - (A_{ij} + A_{ji})\ P_{ij}(\beta)\right|\right] \tag{S30}$$

In preparation to take the derivatives above, note that $P'_{ij}(\beta) = -P'_{ji}(\beta)$ and that whenever the (i,j) term of $\sigma_a(\beta)$ takes one sign, the (j,i) term takes the opposite sign

$$A_{ij} - (A_{ij} + A_{ji})\, P_{ij}(\beta) = -\left[A_{ji} - (A_{ij} + A_{ji})\, P_{ji}(\beta)\right] \tag{S31}$$

Without loss of generality, assume that the (i,j) term is positive and the (j,i) term is negative. This implies that

$$\frac{\partial}{\partial \beta}\left|A_{ij} - (A_{ij} + A_{ji})\, P_{ij}(\beta)\right| = -\left(A_{ij} + A_{ji}\right) P'_{ij}(\beta) \tag{S32}$$

and

$$\frac{\partial}{\partial \beta}\left|A_{ji} - (A_{ij} + A_{ji})\, P_{ji}(\beta)\right| = -\left(A_{ij} + A_{ji}\right) P'_{ij}(\beta) \tag{S33}$$

In other words, the derivatives of the (i,j) and (j,i) terms are identical, and the sign of both depends on whether the quantity $[A_{ij} - (A_{ij} + A_{ji})P_{ij}(\beta)]$ is positive or negative. We can make this more precise by directly including the sign of the (i,j) term, and by using Eq. (S28), to find that

$$\frac{\partial}{\partial \beta}\left|A_{ij} - (A_{ij} + A_{ji})\, P_{ij}(\beta)\right| = -2\left(A_{ij} + A_{ji}\right)(s_i - s_j)\ \mathrm{e}^{-2\beta(s_i - s_j)}\ P_{ij}^2 \times \mathrm{sign}\big\{A_{ij} - (A_{ij} + A_{ji})\, P_{ij}(\beta)\big\} \tag{S34}$$

Expanding P_{ij}^2 and reorganizing yields

$$\frac{\partial}{\partial \beta}\left|A_{ij} - (A_{ij} + A_{ji})\, P_{ij}(\beta)\right| = -2\frac{\left(A_{ij} + A_{ji}\right)(s_i - s_j)}{2\cosh\left[2\beta\,(s_i - s_j)\right] + 2} \times \mathrm{sign}\big\{A_{ij} - (A_{ij} + A_{ji})\, P_{ij}(\beta)\big\} \tag{S35}$$

Combining terms (i,j) and (j,i), the optimal inverse temperature for local accuracy $\hat{\beta}_a$ is that which satisfies

$$0 = \sum_{(i,j)\in U(E)}^{N} \frac{\left(A_{ij} + A_{ji}\right)(s_i - s_j)}{\cosh\left[2\hat{\beta}_a\,(s_i - s_j)\right] + 1} \times \mathrm{sign}\big\{A_{ij} - (A_{ij} + A_{ji})\, P_{ij}(\hat{\beta}_a)\big\} \tag{S36}$$

which may be found using standard root-finding methods.

B. Choosing β to optimize the conditional log likelihood

We take the derivative of Eq. (13) with respect to β, set it equal to zero, and partially solve as follows

$$0 \equiv \frac{\partial \sigma_L(\beta)}{\partial \beta} = \frac{\partial}{\partial \beta}\left[\sum_{i,j}\log\binom{A_{ij} + A_{ji}}{A_{ij}} + \log\left[P_{ij}(\beta)^{A_{ij}}\ \left[1 - P_{ij}(\beta)\right]^{A_{ji}}\right]\right] \tag{S37}$$

Combining the (i,j) and (j,i) terms, we get

$$\begin{aligned}
0 \equiv & \frac{\partial}{\partial \beta}\sum_{(i,j)\in U(E)}\log\binom{A_{ij} + A_{ji}}{A_{ij}} + \log\binom{A_{ij} + A_{ji}}{A_{ji}} + \left[\,A_{ij}\log P_{ij}(\beta)\ + A_{ji}\log\left[1 - P_{ij}(\beta)\right]\,\right] \\
= & \sum_{(i,j)\in U(E)}\left[\frac{A_{ij}}{P_{ij}(\beta)} - \frac{A_{ji}}{1 - P_{ij}(\beta)}\right]\frac{\partial P_{ij}(\beta)}{\partial \beta} \\
= & \sum_{(i,j)\in U(E)} 2\,(s_i - s_j)\ \left[A_{ij} - (A_{ij} + A_{ji})\ P_{ij}(\beta)\right]\frac{P_{ij}(\beta)}{1 - P_{ij}(\beta)}\,\mathrm{e}^{-2\beta(s_i - s_j)}
\end{aligned} \tag{S38}$$

Applying both Eq. (S28) and Eq. (S29), the optimal inverse temperature for the conditional log likelihood $\hat{\beta}_L$ is that which satisfies

$$0 = \sum_{(i,j)\in U(E)} 2\,(s_i - s_j)\,\left[A_{ij} - (A_{ij} + A_{ji})\,P_{ij}(\hat{\beta}_L)\right] \tag{S39}$$

which, like Eq. (S36) may be found using standard root-finding methods. Comparing equations Eq. (S36) and Eq. (S39), we can see that the values of β that maximize the two measures may, in general, be different. Table S2 shows for optimal values for $\hat{\beta}_L$ and $\hat{\beta}_a$ for various real-world datasets.

Section S8. Bitwise accuracy σ_b

Some methods provide rankings but do not provide a model to estimate P_{ij}, meaning that Eq. (12) and Eq. (13) cannot be used. Nevertheless, such methods still estimate one bit of information about each pair (i, j): whether the majority of the edges are from i to j or vice versa. This motivates the use of a bitwise version of σ_a, which we call σ_b,

$$\sigma_b = 1 - \frac{1}{N^2 - t}\sum_{i,j}\Theta\,(s_i - s_j)\,\Theta\,(A_{ji} - A_{ij}) \tag{S40}$$

where$\Theta(\ x) = 1$ if $x > 0$ and$\Theta(\ x) = 0$ otherwise, and N is the number of nodes and t is the number of instances in which $A_{ij} = A_{ji}$; there are $N^2 - t$ total bits to predict. Results in terms of this measure on the networks considered in the main text are shown in Figure S4. In the special case that the network is unweighted (A is a binary adjacency matrix) and there are no bi-directional edges (if $A_{ij} = 1$, then $A_{ji} = 0$), then $1 - \sigma_b$ is the fraction of edges that violate the rankings in s. In other words, for this particular type of network, $1 - \sigma_b$ is the minimum violations rank penalty normalized by the total number of edges in the network, i.e., $\frac{1}{M}\sum_{i,j}\Theta(s_i - s_j)\,A_{ji}$.

Section S9. Performance metrics

When evaluating the performance of a ranking algorithm in general one could consider a variety of different measures. One possibility is to focus on the ranks themselves, rather than the outcomes of pairwise interactions, and calculate correlation coefficients as in Fig. 1; this is a valid strategy when using synthetic data thanks to the presence of ground truth ranks, but can only assess the performance with respect of the specific generative process used to generate the pairwise comparisons, as we point out in the main text. This strategy can also be applied for comparisons with *observed* real world ranks, as we did in Table S11 and it has been done for instance in [19, 20] to compare the ranks with those observed in real data in sports. However, the observed ranks might have been derived from a different process than the one implied by the ranking algorithm considered. For instance, in the faculty hiring networks, popular ranking methods proposed by domain experts for evaluating the prestige of universities do not consider interactions between institutions, but instead rely on a combination of performance indicators such as first-year student retention or graduation rates. The correlation between observed and inferred ranks should thus be treated as a qualitative indicator of how well the two capture similar features of the system, such as prestige, but should not be used to evaluate the performance of a ranking algorithm.

Alternatively, one can look at the outcomes of the pairwise comparisons and relate them to the rankings of the nodes involved as in Eqs. (12) and (13) for testing prediction performance. A popular metric of this type is the number of violations (also called upsets), i.e., outcomes where a higher ranked node is defeated by a lower ranked one. This is very similar to the bitwise accuracy defined in (S40), indeed when there are no ties and two nodes are compared only once, then they are equivalent. These can be seen as low-resolution or coarse-grained measures of performance: for each comparison predict a winner, but do not distinguish between cases where the winner is easy to predict and cases where there is almost a tie. In particular, an upset between two nodes ranked nearby counts as much as an upset between two nodes that are far away in the ranking. The latter case signals a much less likely scenario. In order to distinguish these two situations, one can penalize each upset by the nodes' rank difference elevated to a certain power d. This is what the *agony* function does [18] with the exponent d treated as a parameter to tune based on the application. When $d = 0$ we recover the standard number of unweighted upsets.

Note that optimization of agony is often used as a non-parametric approach to detect hierarchies [21], in particular for ordinal ranks. For ordinal ranks, rank differences are integer-valued and equal to one for adjacent-ranked nodes, yet for real-valued scores this is not the case. Therefore the result of the agony minimization problem can vary

widely between ordinal and real valued ranking algorithms. (We note that the SpringRank objective function, i.e., the Hamiltonian in Eq. (2), can be considered a kind of agony. However, since we assume that nearby pairs are more likely to interact, it is large for a edge from i to j if i is ranked far above or far below j, and more specifically whenever s_i is far from $s_j + 1$.)

In contrast to the coarse prediction above—which competitor is more likely to win?—we require, when possible, more precise predictions in Eqs. (12) and (13), which ask *how much more likely* is one competitor to win? This, however, requires the ranking algorithm to provide an estimate of P_{ij}, the probability that i wins over j, which is provided only by BTL and SpringRank; all other methods compared in this study provide orderings or embeddings without probabilistic predictions.

The conditional log-likelihood σ_L as defined in Eq. (13) can be seen as a *Log Loss* often used as a classification loss function [46] in statistical learning. This type of function heavily penalizes ranking algorithms that are very confident about an incorrect outcome, e.g. when the predicted P_{ij} is close to 1, i very likely to win over j, but the observed outcome is that j wins over i. For this reason, this metric is more sensitive to outliers, as when in sports a very strong team loses against one at the bottom of the league. The accuracy σ_b defined in Eq. (12) focuses instead in predicting the correct proportions of wins/losses between two nodes that are matched in several comparisons. This is less sensitive to outliers, and in fact if P_{ij} is close but not exactly equal to 1, for a large number of comparisons between i and j, we would expect that j should indeed win few times, e.g. if $P_{ij} = 0.99$ and i, j are compared 100 times, σ_a is maximized when i wins 99 times and j wins once.

Section S10. Parameters used for regularizing ranking methods

When comparing SpringRank to other methods, we need to deal with the fact that certain network structures cause other methods to fail to return any output. Eigenvector Centrality cannot, for example, be applied to directed trees, yet this is precisely the sort of structure that one might expect when hierarchy becomes extreme.

More generally, many spectral techniques fail on networks that are not *strongly connected*, i.e., where it is not the case that one can reach any node from any other by moving along a path consistent with the edge directions, since in that case the adjacency matrix is not irreducible and the Perron-Frobenius theorem does not apply. In particular, nodes with zero out-degree—sometimes called "dangling nodes" in the literature [13]—cause issues for many spectral methods since the adjacency matrix annihilates any vector supported on such nodes. In contrast, the SpringRank optimum given by Eq. (3) is unique up to translation whenever the network is connected in the undirected sense, i.e., whenever we can reach any node from any other by moving with or against directed edges.

A different issue occurs in the case of SyncRank. When edges are reciprocal in the sense that an equal number of edges point in each direction, they effectively cancel out. That is, if $A_{ij} = A_{ji}$, the corresponding entries in the SyncRank comparison matrix will be zero, $C_{ij} = C_{ji} = 0$, as if i and j were never compared at all. As a result, there can be nodes i such that $C_{ij} = C_{ji} = 0$ for all j. While rare, these pathological cases exist in real data and during cross-validation tests, causing the output of SyncRank to be undefined.

In all these cases, regularization is required. Our regularized implementations of five ranking methods are described below:

- **Regularized Bradley-Terry-Luce (BTL).** If there exist dangling nodes, the Minimization-Maximization algorithm to fit the BTL model to real data proposed in [38] requires a regularization. In this case we set the total number of out-edges $d_i^{\text{out}} = 10^{-6}$ for nodes that would have $d_i = 0$ otherwise. This corresponds to W_i in Eq.(3) of [38].
- **Regularized PageRank.** If there exist dangling nodes, we add an edge of weight $1/N$ from each dangling node to every other node in the network. For each dataset we tried three different values of the teleportation parameter, $\alpha \in \{0.4, 0.6, 0.8\}$, and reported the best results of these three.
- **Regularized Rank Centrality.** If there exist dangling nodes, we use the regularized version of the algorithm presented in Eq. (5) of [14] with $\epsilon = 1$.
- **Regularized SyncRank.** If there are nodes whose entries in the comparison matrix C are zero, we add a small constant $\epsilon = 0.001$ to the entries of H in Eq. (13) of Ref. [20], so that D is invertible.
- **Regularized Eigenvector Centrality.** If the network is not strongly connected, we add a weight of $1/N$ to every entry in A and then diagonalize.

Section S11. Supplementary Tables

Table S1. Pearson correlation coefficients between various rankings of faculty hiring networks. All coefficients are statistically significant ($p < 10^{-9}$). SpringRank is most highly correlated with Minimum Violations Ranks across all three faculty hiring networks. Among *US News* and NRC rankings, SpringRank is more similar to *US News*. Values for *US News* and NRC were drawn from Ref. [3] for comparison to the ranks available at the same time that the faculty hiring data were collected. The NRC does not rank business departments.

Comp. Sci.	SpringRank	MVR	*US News*	NRC	Eig. C.	PageRank
SpringRank	-	0.96	0.80	0.72	0.84	0.57
MVR	0.96	-	0.81	0.73	0.80	0.48
US News	0.80	0.81	-	0.73	0.69	0.41
NRC	0.72	0.73	0.73	-	0.68	0.41
Eig. C.	0.84	0.80	0.69	0.68	-	0.74
PageRank	0.57	0.48	0.41	0.41	0.74	-

Business	SpringRank	MVR	*US News*	NRC	Eig. C.	PageRank
SpringRank	-	0.98	0.74	-	0.92	0.75
MVR	0.98	-	0.72	-	0.92	0.69
US News	0.74	0.72	-	-	0.68	0.60
NRC	-	-	-	-	-	-
Eig. C.	0.92	0.92	0.68	-	-	0.72
PageRank	0.75	0.69	0.60	-	0.72	-

History	SpringRank	MVR	*US News*	NRC	Eig. C.	PageRank
SpringRank	-	0.95	0.86	0.66	0.86	0.69
MVR	0.95	-	0.86	0.65	0.77	0.57
US News	0.86	0.86	-	0.66	0.72	0.51
NRC	0.66	0.65	0.66	-	0.59	0.44
Eig. C.	0.86	0.77	0.72	0.59	-	0.88
PageRank	0.69	0.57	0.51	0.44	0.88	-

Table S2. **Statistics for SpringRank applied to real-world networks.** Column details are as follows: N is the number of nodes; M is the number of edges; H/m is the ground state energy per edge; Accuracy σ_a refers to accuracy in 5-fold cross-validation tests using temperature $\hat{\beta}_a$; $\hat{\beta}_L$ and $\hat{\beta}_a$ are temperatures optimizing edge prediction accuracies σ_L and σ_a respectively; Violations refers to the number of edges that violate the direction of the hierarchy as a number, as a percentage of all edges, with a lower bound provided for reference, computed as the number of unavoidable violations due to reciprocated edges; Weighted violations are the sum of each violation weighted by the difference in ranks between the offending nodes; Depth is $s_{\max} - s_{\min}$; p-value refers to the null model described in the Materials and Methods. Relevant performance statistics for NCAA datasets (53 networks) are reported elsewhere; see Fig. S3.

DataSet	Type	N	M	H/M	Acc. σ_a	$\hat{\beta}_L$	$\hat{\beta}_a$	Viol. (%) / Bound	Wt. viol. (per viol.)	Depth	p-value
Parakeet G1 [5]	Anim. Dom.	21	838	0.174	0.930	2.70	6.03	76 (9.1%) / 42	0.008 (0.089)	2.604	$< 10^{-4}$
Parakeet G2 [5]	Anim. Dom.	19	961	0.193	0.932	2.78	18.12	75 (7.8%) / 36	0.011 (0.139)	1.879	$< 10^{-4}$
Asian Elephants [37]	Anim. Dom.	20	23	0.078	0.923	2.33	3.44	2 (8.7%) / 0	0.001 (0.040)	3.000	0.4466
Business [3]	Fac. Hiring	112	7353	0.251	0.881	2.04	3.14	1171 (15.9%) / 808	0.019 (0.119)	2.125	$< 10^{-4}$
Computer Science [3]	Fac. Hiring	205	4033	0.220	0.882	2.23	8.74	516 (12.8%) / 255	0.013 (0.105)	2.423	$< 10^{-4}$
History [3]	Fac. Hiring	144	3921	0.186	0.909	2.39	5.74	397 (10.1%) / 227	0.012 (0.119)	2.234	$< 10^{-4}$
Alakāpuram [2]	Soc. Support	415	2497	0.222	0.867	1.98	7.95	347 (13.9%) / 120	0.011 (0.079)	3.618	$< 10^{-4}$
Tenpaṭṭi [2]	Soc. Support	361	1809	0.241	0.858	1.89	8.20	262 (14.5%) / 120	0.012 (0.082)	3.749	$< 10^{-4}$

Section S12. Supplementary Figures

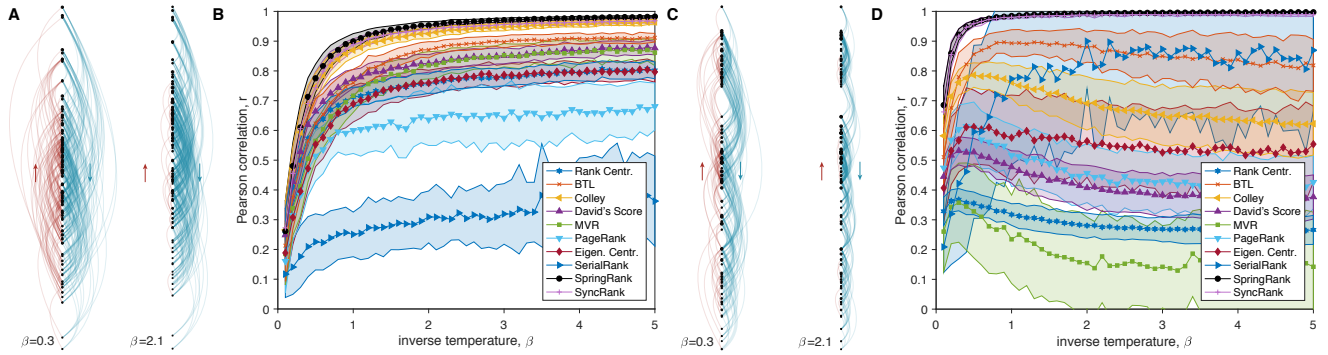
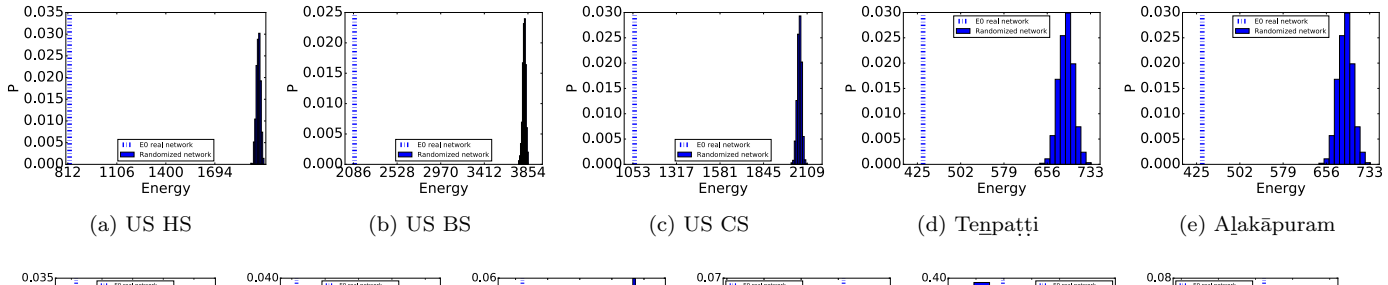
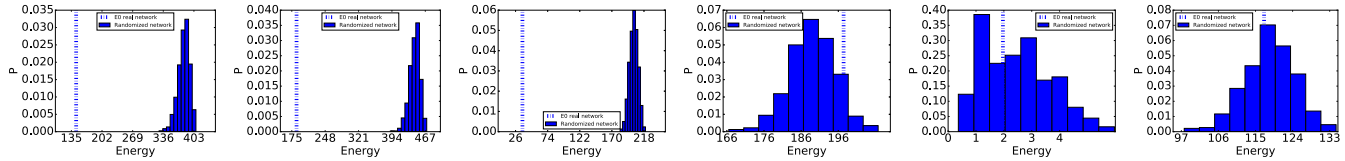


Fig. S1. **Performance (Pearson correlation) on synthetic data.** Tests were performed as in Fig. 1, but here performance is measured using Pearson correlation. This favors algorithms like SpringRank and BTL, that produce real-valued ranks, over ordinal ranking schemes like Minimum Violation Ranking which are not expected to recover latent positions. (A) Linear hierarchy diagrams show latent ranks s_{planted} of 100 nodes, drawn from a standard normal distribution, with edges drawn via the generative model Eq. (7) for indicated β (noise) values. Blue edges point down the hierarchy and red edges point up, indicated by arrows. (B) Mean accuracies $\pm$ one standard deviation (symbols $\pm$ shading) are measured as the Pearson correlation between method output and s_{planted} for 100 replicates. (C, D) Identical to A and B but for hierarchies of $N = 102$ nodes divided into three tiers. All plots have mean degree 5; see Fig. 1 for performance curves for Spearman correlation r. See Materials and Methods for synthetic network generation.



(a) US HS (b) US BS (c) US CS (d) Tenpaṭṭi (e) Aḷakāpuram



(f) parakeet G1 (g) parakeet G2 (h) planted $\beta = 5.0$ (i) planted $\beta = 0.1$ (j) Asian elephant (k) ER $N = 100, \langle k \rangle = 3$

Fig. S2. **Statistical significance testing using the null model distribution of energies.** Results are 1000 realizations of the null model where edge directions are randomized while keeping the total number of interactions between each pair fixed, for real and synthetic networks: a-c) US History (HS), Business (BS) and Computer Science (CS) faculty hiring networks [3]; d-e) social support networks of two Indian villages [2] considering 5 types of interactions (see main manuscript); f,g) aggression network of parakeet Group 1 and 2 (as in [5]); h,i) planted network using SpringRank generative model with $N = 100$ and mean degree $\langle k \rangle = 5$, Gaussian prior for the ranks with average $\mu = 0.5$ and variance 1 ($\alpha = 1/\beta$) and two noise levels $\beta = 5.0$ and $\beta = 0.1$; j) dominance network of asian elephants [37]; k) Erdős-Rényi directed random network with $N = 100$ and $\langle k \rangle = 3$. The vertical line is the energy obtained on the real network. In all but the last two cases we reject the null hypothesis that edge directions are independent of the ranks, and conclude that the hierarchy is statistically significant.

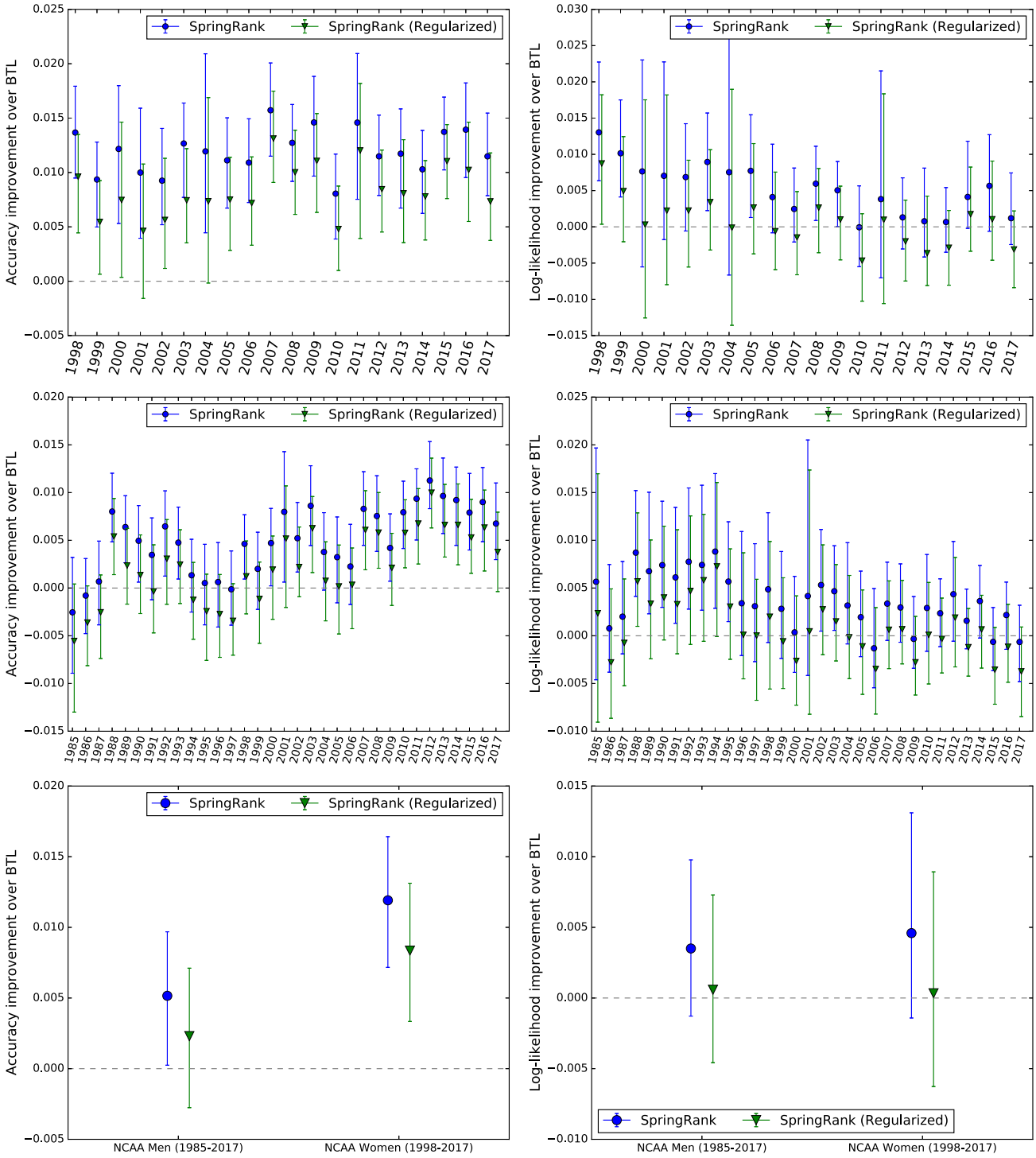


Fig. S3. **Edge prediction accuracy over BTL for NCAA basketball data sets**. Distribution of differences in performance of edge prediction of SpringRank compared to BTL on NCAA College Basketball regular season matches for (top) Women and (middle) Men, defined as (left) the probabilistic edge-prediction accuracy σ_a Eq. (12) and (right) the conditional log-likelihood σ_L Eq. (13). Error bars indicate quartiles and markers show medians, corresponding to 50 independent trials of 5-fold cross-validation, for a total of 250 test sets for each dataset. The bottom plot is obtained by considering the distributions over all the seasons together. In terms of number of correctly predicted outcomes, SpringRank correctly predicts on average 8 to 16 more outcomes than BTL for each of the 20 Women NCAA seasons and up to 12 more outcomes for each of the 33 Men NCAA seasons; for the latter dataset, BTL has an average better prediction in 3 out of the 33 seasons. The number of matches played per season in the test set varies from the past to the most recents years from 747 to 1079.

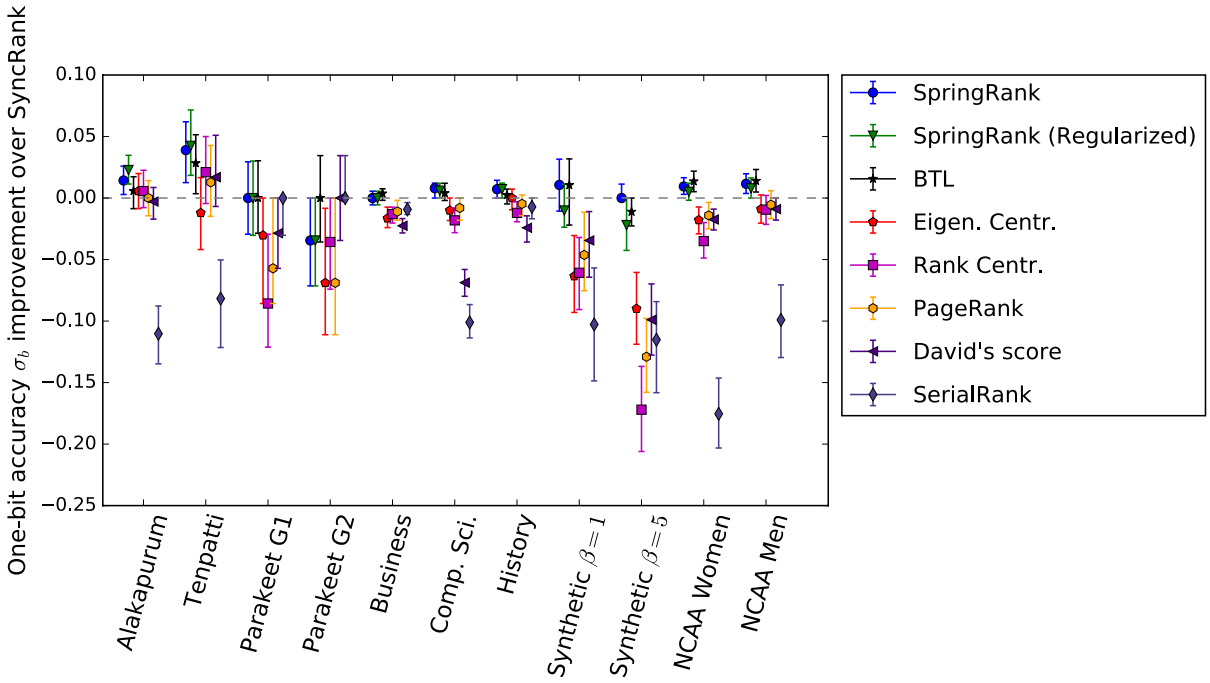


Fig. S4. **Bitwise edge direction prediction**. Symbols show medians of bitwise edge prediction accuracies Eq. (S40) over 50 realization of 5-fold cross-validation (for a total of 250 trials) compared with the median accuracy for SyncRank; error bars indicate quartiles. Thus, points above the dashed line at zero indicate better predictions than SyncRank, while values below indicate that SyncRank performed better.

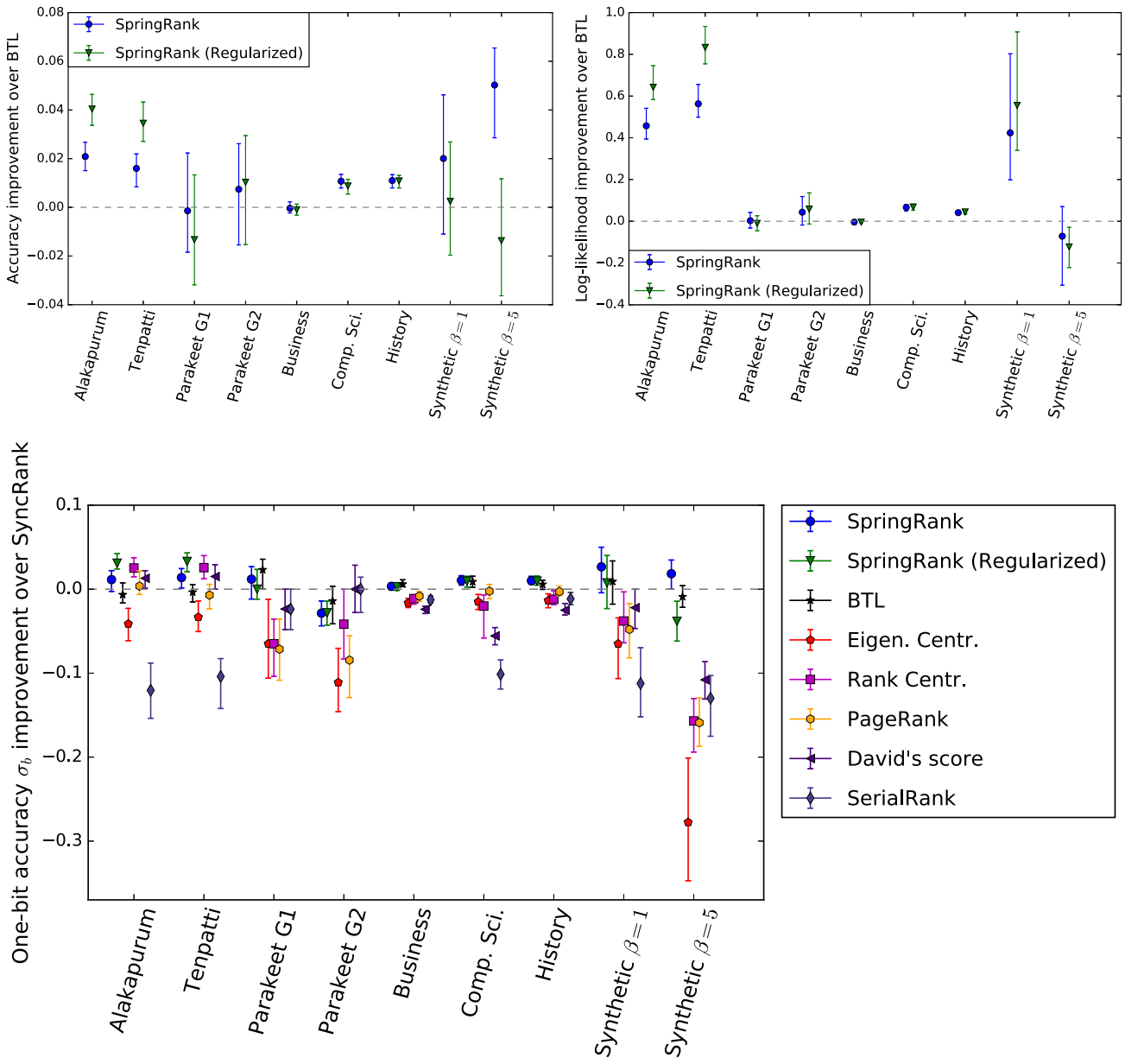


Fig. S5. **Edge prediction accuracy with twofold cross-validation**. Top: the accuracy of probabilistic edge prediction of SpringRank compared to the median accuracy of BTL on real and synthetic networks defined as (top left) edge-prediction accuracy σ_a Eq. (12) and (top right) the conditional log-likelihood σ_L Eq. (13); (bottom) bitwise edge prediction accuracies σ_b Eq. (S40) of SpringRank and other algorithms compared with the median accuracy of SyncRank. Error bars indicate quartiles and markers show medians, corresponding to 50 independent trials of 2-fold cross-validation, for a total of 100 test sets for each network. The two synthetic networks are generated with $N = 100$, average degree 5, and Gaussian-distributed ranks as in Fig. 1A, with inverse temperatures $\beta = 1$ and $\beta = 5$. Notice that these results are similar those of Fig. 3, obtained using 5-fold cross-validation.

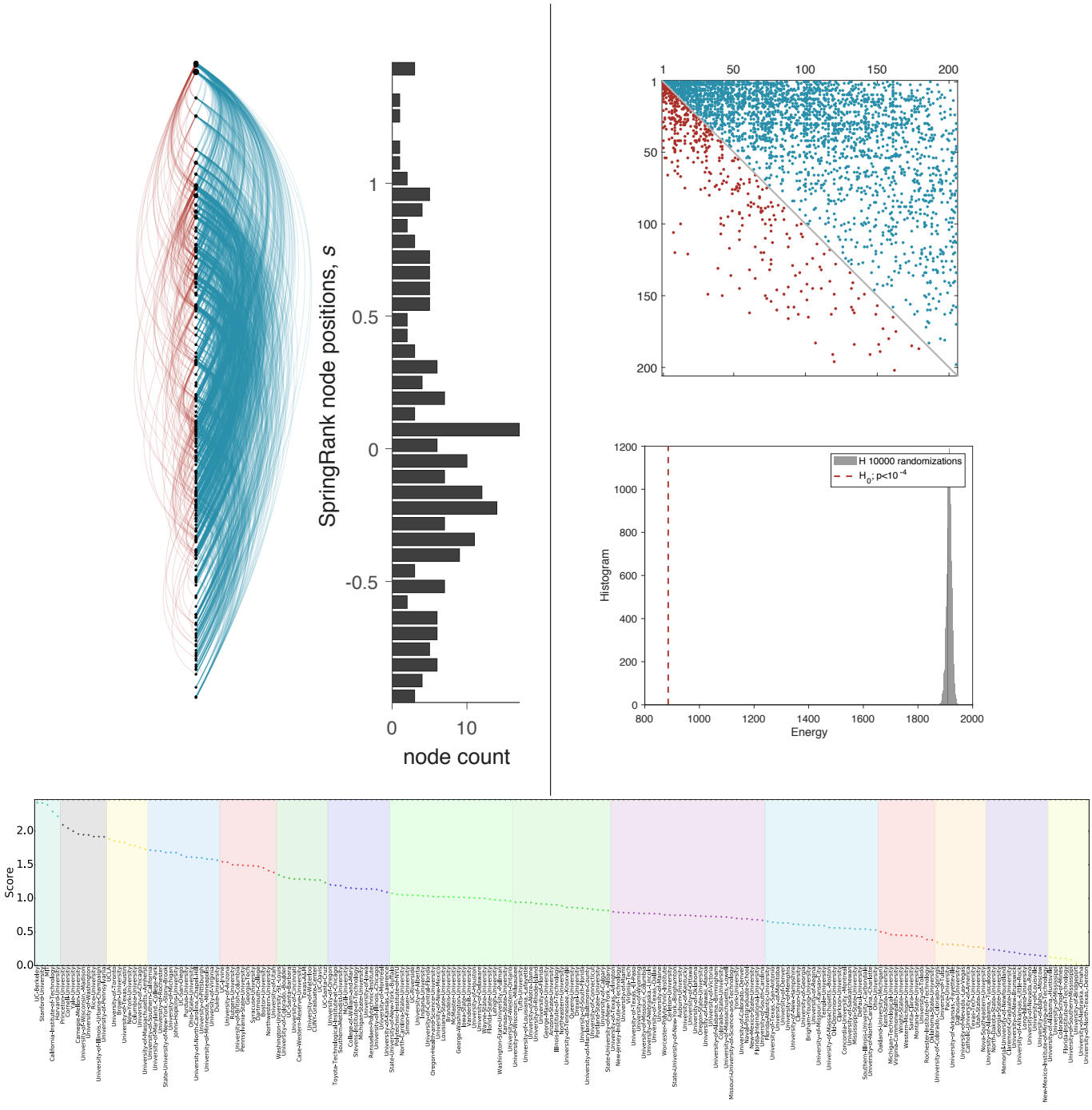


Fig. S6. **Summary of SpringRank applied to computer science faculty hiring network.** (top-left) A linear hierarchy diagram showing inferred SpringRank scores. Circles correspond to nodes; blue edges point down the hierarchy and red edges point up. (top-middle) A histogram shows the empirical distribution of ranks: the vertical axis is the rank s_i (binned) and the horizontal axis is the count of nodes having a rank in that bin. (top-right) A sparsity plot of rank-ordered adjacency matrix; blue and red dots represent non-zero entries going down and up the hierarchy, respectively. (middle-right) Results of statistical significance test with randomized edge directions. The histogram represents the energies obtained in the randomized samples: the dotted line is the ground state energy obtained on the observed real network. (bottom) Nodes' ranks are plotted, ordered by rank, from top rank (left) to bottom rank (right), and shaded by tier. The tiers are calculated by the k-means algorithm.

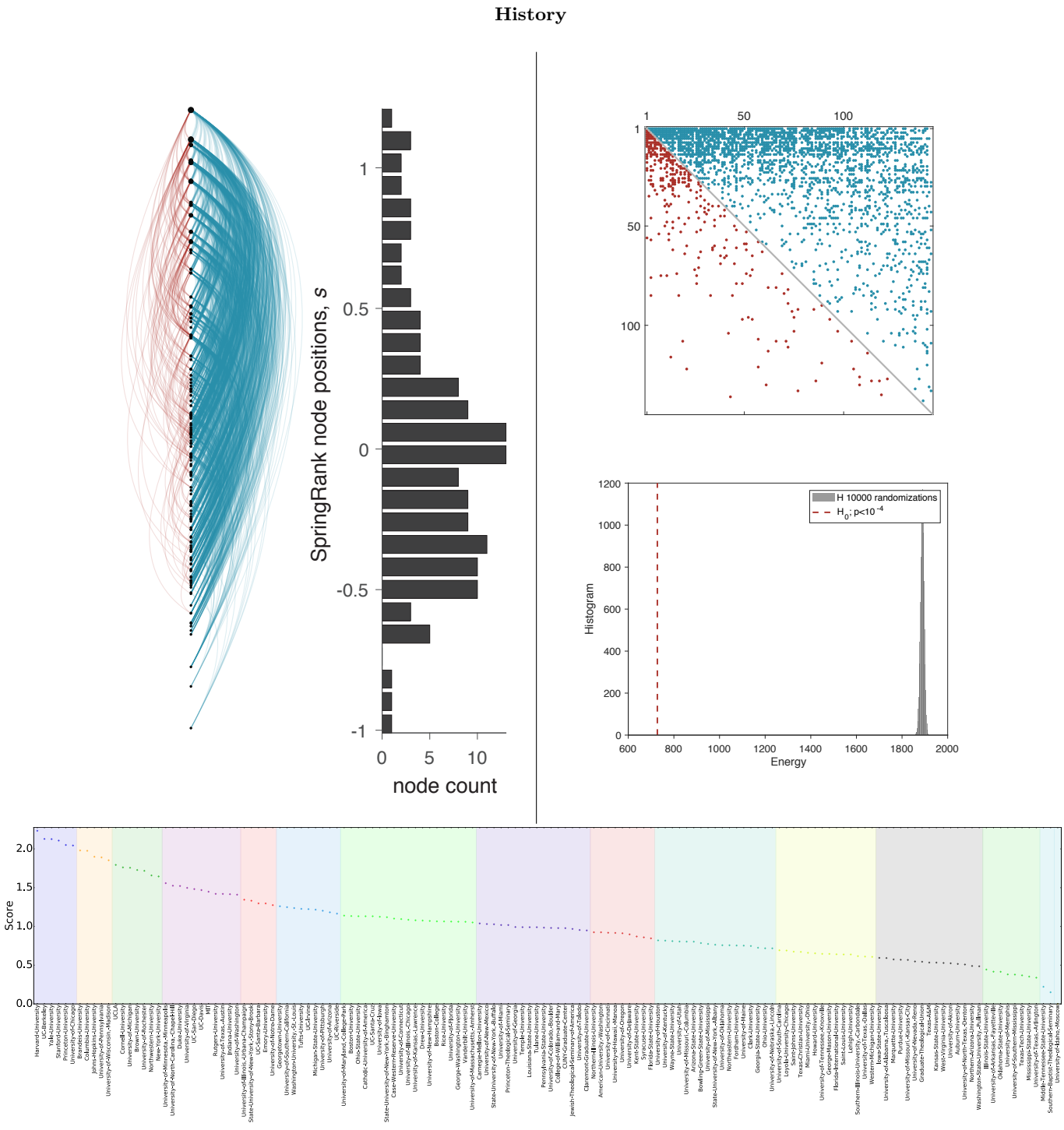


Fig. S7. **Summary of SpringRank applied to history faculty hiring network.** (top-left) A linear hierarchy diagram showing inferred SpringRank scores. Circles correspond to nodes; blue edges point down the hierarchy and red edges point up. (top-middle) A histogram shows the empirical distribution of ranks: the vertical axis is the rank s_i (binned) and the horizontal axis is the count of nodes having a rank in that bin. (top-right) A sparsity plot of rank-ordered adjacency matrix; blue and red dots represent non-zero entries going down and up the hierarchy, respectively. (middle-right) Results of statistical significance test with randomized edge directions. The histogram represents the energies obtained in the randomized samples: the dotted line is the ground state energy obtained on the observed real network. (bottom) Nodes' ranks are plotted, ordered by rank, from top rank (left) to bottom rank (right), and shaded by tier. The tiers are calculated by the k-means algorithm.

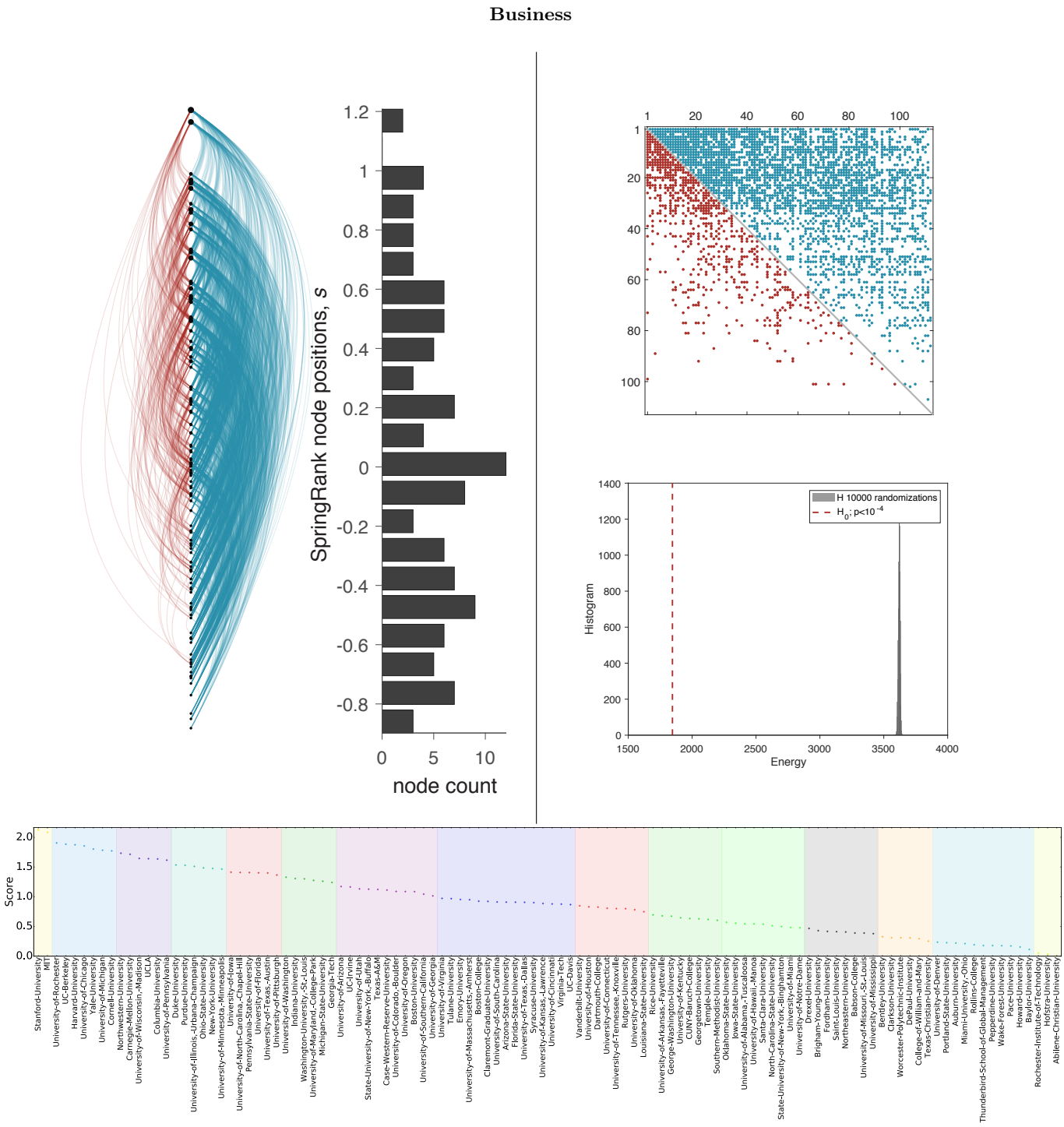


Fig. S8. **Summary of SpringRank applied to business faculty hiring network.** (top-left) A linear hierarchy diagram showing inferred SpringRank scores. Circles correspond to nodes; blue edges point down the hierarchy and red edges point up. (top-middle) A histogram shows the empirical distribution of ranks: the vertical axis is the rank s_i (binned) and the horizontal axis is the count of nodes having a rank in that bin. (top-right) A sparsity plot of rank-ordered adjacency matrix; blue and red dots represent non-zero entries going down and up the hierarchy, respectively. (middle-right) Results of statistical significance test with randomized edge directions. The histogram represents the energies obtained in the randomized samples: the dotted line is the ground state energy obtained on the observed real network. (bottom) Nodes' ranks are plotted, ordered by rank, from top rank (left) to bottom rank (right), and shaded by tier. The tiers are calculated by the k-means algorithm.

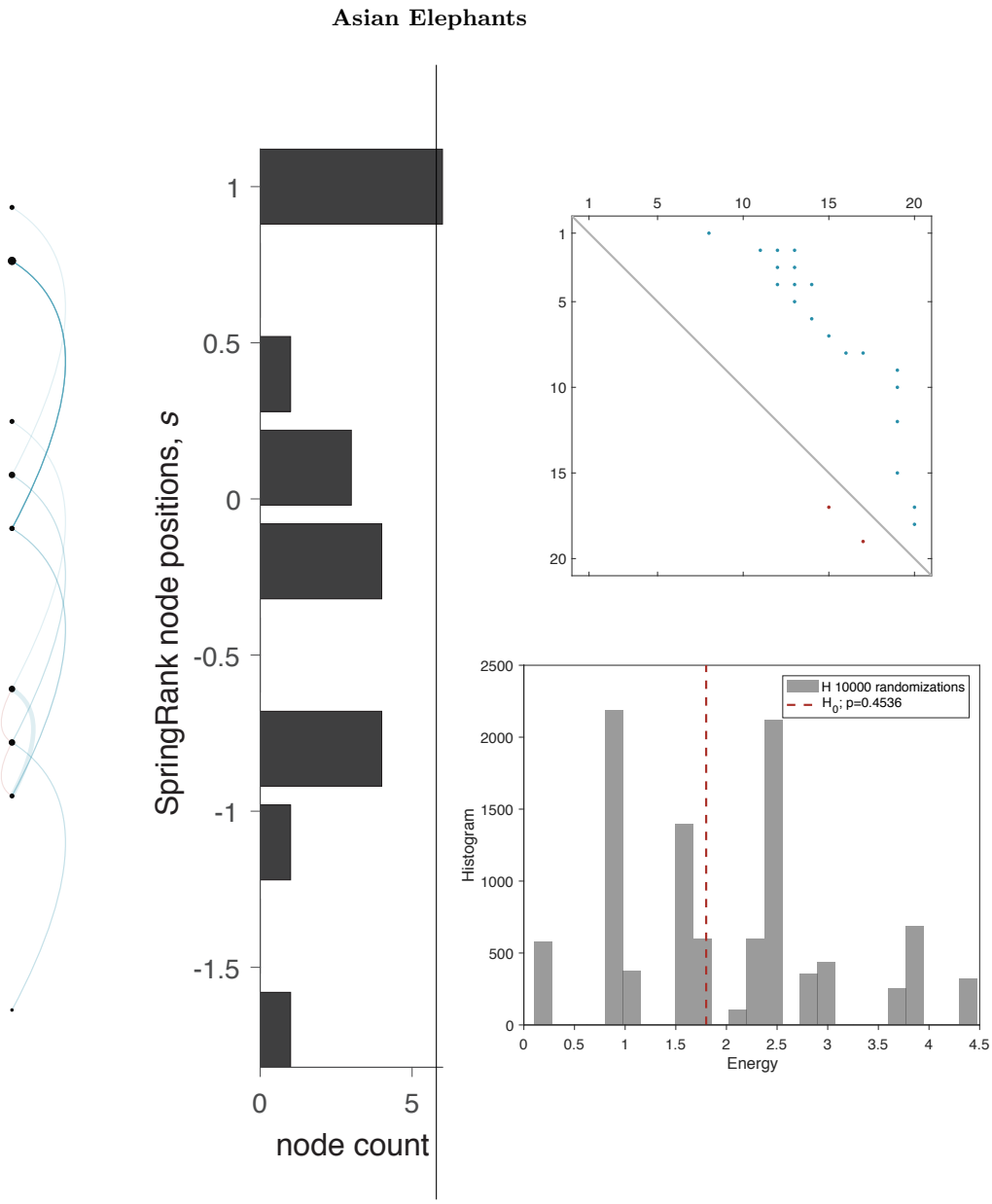


Fig. S9. **Summary of SpringRank applied to Asian Elephants network.** (top-left) A linear hierarchy diagram showing inferred SpringRank scores. Circles correspond to nodes; blue edges point down the hierarchy and red edges point up. (top-middle) A histogram shows the empirical distribution of ranks: the vertical axis is the rank s_i (binned) and the horizontal axis is the count of nodes having a rank in that bin. (top-right) A sparsity plot of rank-ordered adjacency matrix; blue and red dots represent non-zero entries going down and up the hierarchy, respectively. (middle-right) Results of statistical significance test with randomized edge directions. The histogram represents the energies obtained in the randomized samples: the dotted line is the ground state energy obtained on the observed real network.

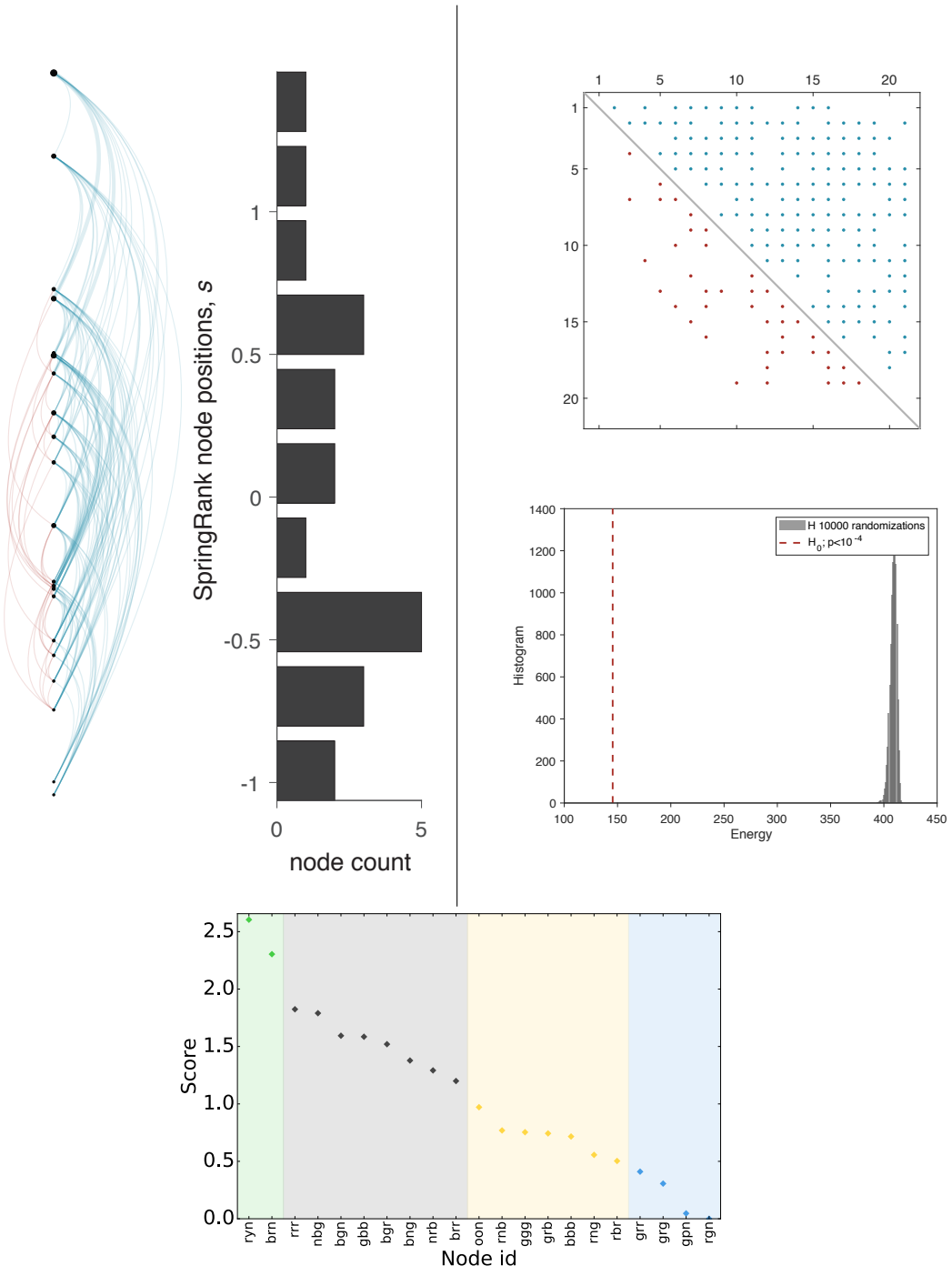


Fig. S10. **Summary of SpringRank applied to parakeet G1 network.** (top-left) A linear hierarchy diagram showing inferred SpringRank scores. Circles correspond to nodes; blue edges point down the hierarchy and red edges point up. (top-middle) A histogram shows the empirical distribution of ranks: the vertical axis is the rank s_i (binned) and the horizontal axis is the count of nodes having a rank in that bin. (top-right) A sparsity plot of rank-ordered adjacency matrix; blue and red dots represent non-zero entries going down and up the hierarchy, respectively. (middle-right) Results of statistical significance test with randomized edge directions. The histogram represents the energies obtained in the randomized samples: the dotted line is the ground state energy obtained on the observed real network. (bottom) Nodes' ranks are plotted, ordered by rank, from top rank (left) to bottom rank (right), and shaded by tier. The tiers are calculated by the k-means algorithm.

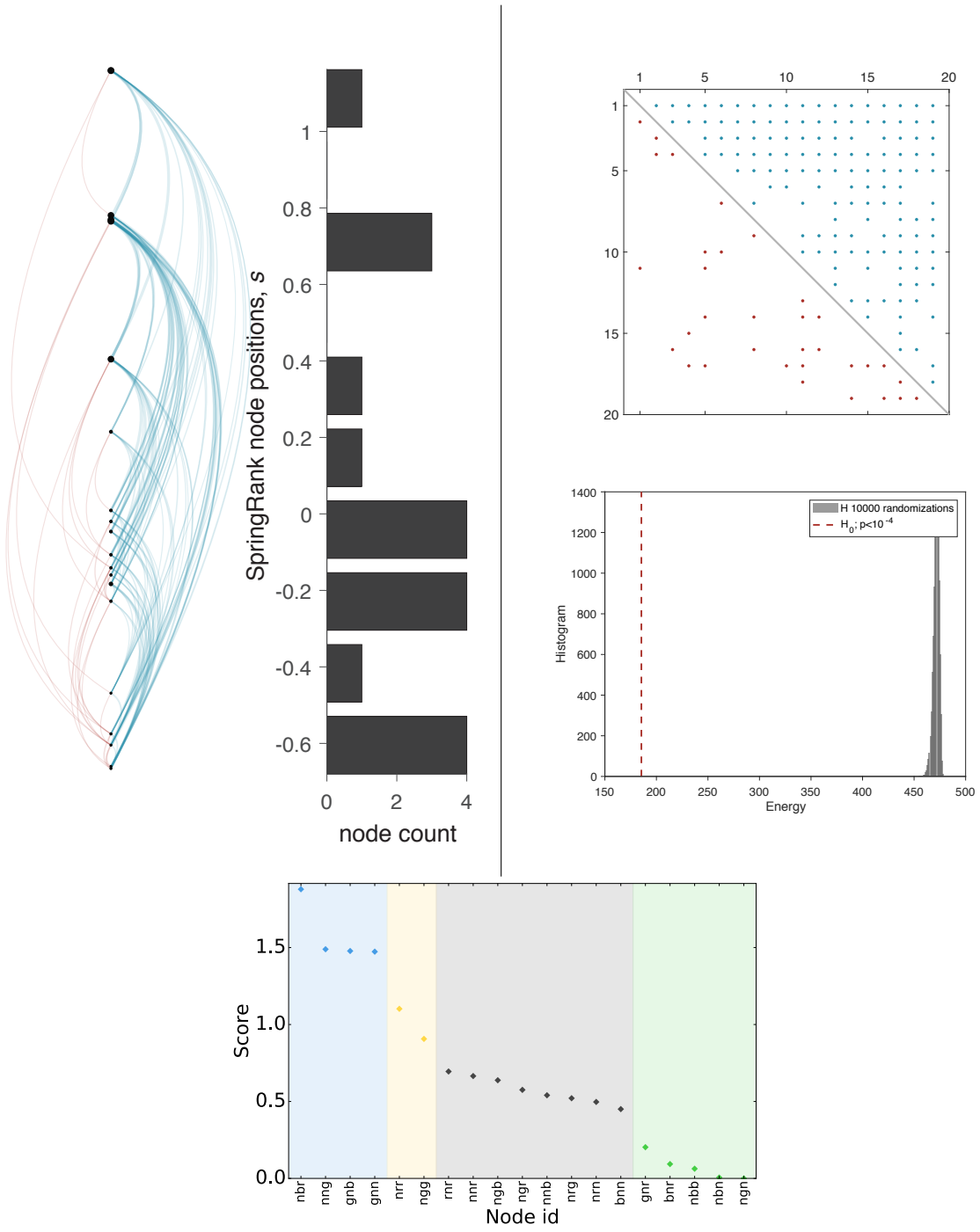


Fig. S11. **Summary of SpringRank applied to parakeet G2 network.** (top-left) A linear hierarchy diagram showing inferred SpringRank scores. Circles correspond to nodes; blue edges point down the hierarchy and red edges point up. (top-middle) A histogram shows the empirical distribution of ranks: the vertical axis is the rank s_i (binned) and the horizontal axis is the count of nodes having a rank in that bin. (top-right) A sparsity plot of rank-ordered adjacency matrix; blue and red dots represent non-zero entries going down and up the hierarchy, respectively. (middle-right) Results of statistical significance test with randomized edge directions. The histogram represents the energies obtained in the randomized samples: the dotted line is the ground state energy obtained on the observed real network. (bottom) Nodes' ranks are plotted, ordered by rank, from top rank (left) to bottom rank (right), and shaded by tier. The tiers are calculated by the k-means algorithm.

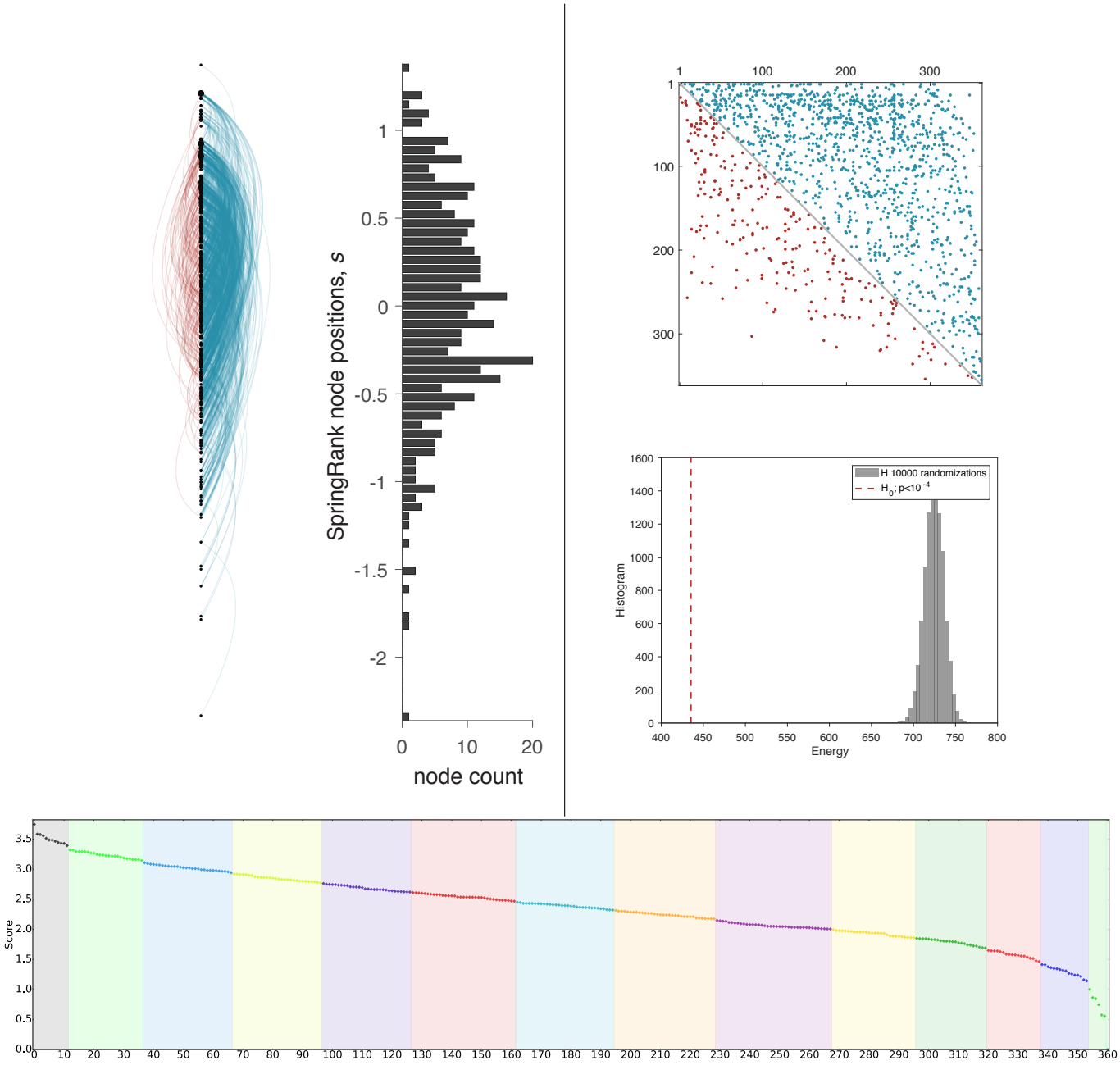


Fig. S12. **Summary of SpringRank applied to Tenpaṭṭi social support network.** (top-left) A linear hierarchy diagram showing inferred SpringRank scores. Circles correspond to nodes; blue edges point down the hierarchy and red edges point up. (top-middle) A histogram shows the empirical distribution of ranks: the vertical axis is the rank s_i (binned) and the horizontal axis is the count of nodes having a rank in that bin. (top-right) A sparsity plot of rank-ordered adjacency matrix; blue and red dots represent non-zero entries going down and up the hierarchy, respectively. (middle-right) Results of statistical significance test with randomized edge directions. The histogram represents the energies obtained in the randomized samples: the dotted line is the ground state energy obtained on the observed real network. (bottom) Nodes' ranks are plotted, ordered by rank, from top rank (left) to bottom rank (right), and shaded by tier. The tiers are calculated by the k-means algorithm.

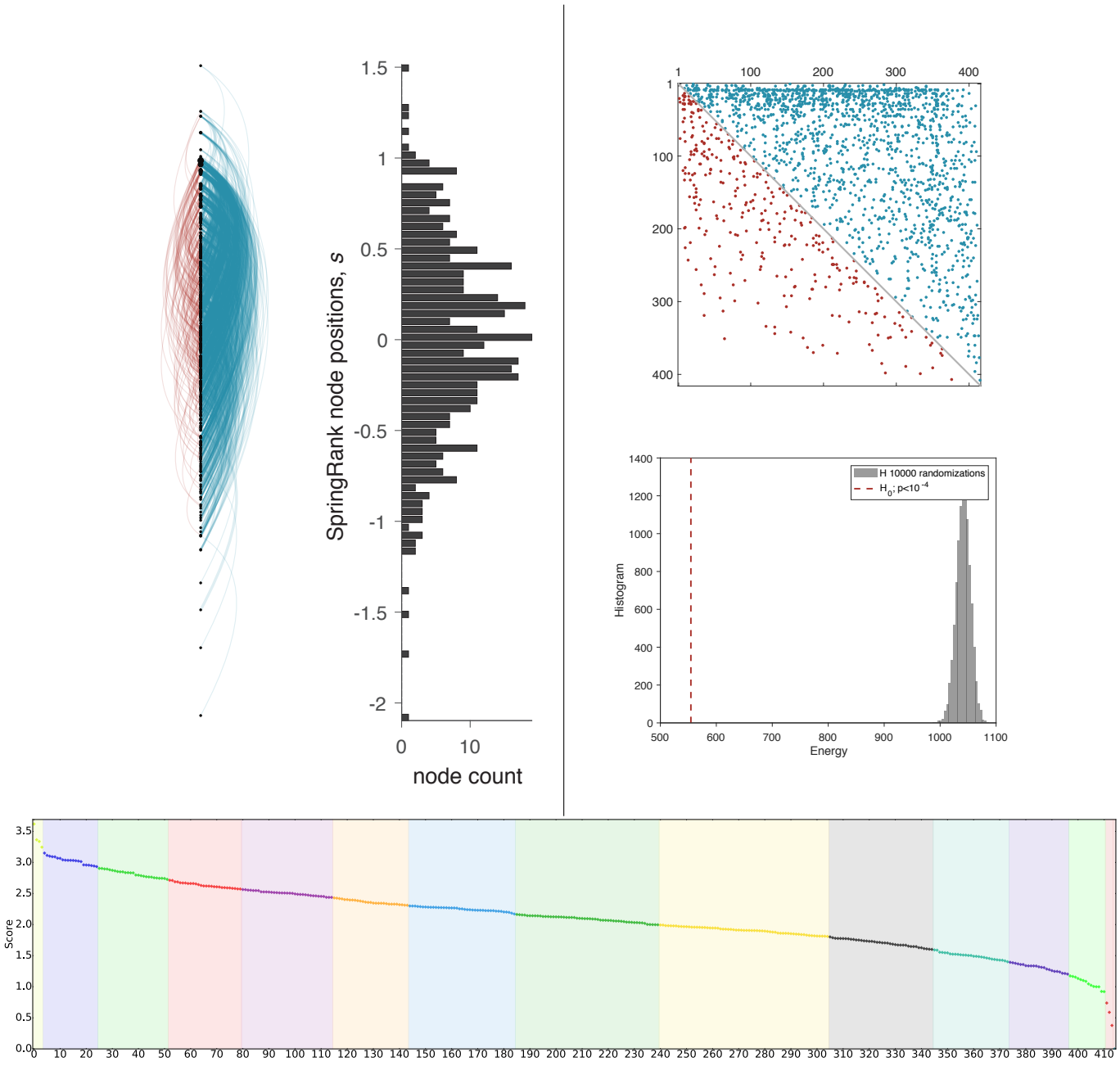


Fig. S13. **Summary of SpringRank applied to Alakāpuram social support network.** (top-left) A linear hierarchy diagram showing inferred SpringRank scores. Circles correspond to nodes; blue edges point down the hierarchy and red edges point up. (top-middle) A histogram shows the empirical distribution of ranks: the vertical axis is the rank s_i (binned) and the horizontal axis is the count of nodes having a rank in that bin. (top-right) A sparsity plot of rank-ordered adjacency matrix; blue and red dots represent non-zero entries going down and up the hierarchy, respectively. (middle-right) Results of statistical significance test with randomized edge directions. The histogram represents the energies obtained in the randomized samples: the dotted line is the ground state energy obtained on the observed real network. (bottom) Nodes' ranks are plotted, ordered by rank, from top rank (left) to bottom rank (right), and shaded by tier. The tiers are calculated by the k-means algorithm.