HHS Public Access

Author manuscript

Stat Med. Author manuscript; available in PMC 2025 January 06.

Published in final edited form as:

Stat Med. 2023 June 15; 42(13): 2116–2133. doi:10.1002/sim.9714.

SpiderLearner: An ensemble approach to Gaussian graphical model estimation

Katherine H. Shutta[1,2,3], **Laura B. Balzer**[4], **Denise M. Scholtens**[5], **Raji Balasubramanian**[1]

[1]Department of Biostatistics and Epidemiology, University of Massachusetts–Amherst, Amherst, Massachusetts, USA

[2]Department of Biostatistics, Harvard School of Public Health, Boston, Massachusetts, USA

[3]Channing Division of Network Medicine, Department of Medicine, Brigham and Women's Hospital, Harvard Medical School, Boston, Massachusetts, USA

[4]Division of Biostatistics, University of California–Berkeley, Berkeley, California, USA

[5]Division of Biostatistics, Department of Preventive Medicine, Northwestern University Feinberg School of Medicine, Chicago, Illinois, USA

Abstract

Gaussian graphical models (GGMs) are a popular form of network model in which nodes represent features in multivariate normal data and edges reflect conditional dependencies between these features. GGM estimation is an active area of research. Currently available tools for GGM estimation require investigators to make several choices regarding algorithms, scoring criteria, and tuning parameters. An estimated GGM may be highly sensitive to these choices, and the accuracy of each method can vary based on structural characteristics of the network such as topology, degree distribution, and density. Because these characteristics are *a priori* unknown, it is not straightforward to establish universal guidelines for choosing a GGM estimation method. We address this problem by introducing SpiderLearner, an ensemble method that constructs a consensus network from multiple estimated GGMs. Given a set of candidate methods, SpiderLearner estimates the optimal convex combination of results from each method using a likelihood-based loss function. K-fold cross-validation is applied in this process, reducing the risk of overfitting. In simulations, SpiderLearner performs better than or comparably to the best candidate methods according to a variety of metrics, including relative Frobenius norm and out-of-sample likelihood. We apply SpiderLearner to publicly available ovarian cancer gene expression data including 2013 participants from 13 diverse studies, demonstrating our tool's potential to identify biomarkers of complex disease. SpiderLearner is implemented as flexible, extensible, open-source code in the R package `ensembleGGM` at https://github.com/katehoffshutta/ensembleGGM.

Correspondence Katherine H. Shutta, Department of Biostatistics, Harvard School of Public Health, Boston, MA, USA. kshutta@hsph.harvard.edu.

SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

Keywords

ensemble models; Gaussian graphical models; gene expression; networks; ovarian cancer; super learner

1 | INTRODUCTION

Gaussian graphical models (GGMs) provide a modeling framework for network-based analyses of multivariate normal data. A GGM is an undirected graph in which nodes correspond to variables and weighted edges correspond to the magnitude of the partial correlation between pairs of variables, that is, their correlation conditional on all of the other variables in the network.[1] In a GGM, the absence of an edge between nodes corresponds to zero partial correlation; if the data are indeed multivariate normal, this is equivalent to conditional independence between the two nodes, given the other nodes in the network. Under the assumption of multivariate normality, it can be shown that the pairwise partial correlations are functions of the corresponding elements of the precision (inverse covariance) matrix.[2] Thus, estimating a GGM is equivalent to estimating the corresponding precision matrix using the sampled multivariate normal data.

Precision matrix estimation is typically straightforward when the sample size n is much larger than the number of predictors p; in this setting, a maximum likelihood estimate can be found simply by inverting the sample covariance matrix. When n is close to or less than p or when variables are highly correlated, however, this inverse is undefined or numerically unstable. A popular approach for GGM estimation in these settings is to apply the graphical lasso, which estimates a sparse precision matrix by optimizing a penalized log likelihood function.[3-5] Several existing open-source software resources implement versions of the graphical lasso, including methods for selecting the tuning parameter λ. For example, the `glasso R` package implements the original graphical lasso algorithm developed in 2008 by Friedman et al,[3] augmented by computational advances developed in 2011 by Witten et al.[6] The `huge` R package incorporates the graphical lasso algorithm with several different options for scoring criteria.[7] The hub graphical lasso is an extension to the graphical lasso designed to better estimate networks with hub structure.[8] Approaches exist outside of the graphical lasso as well; for example, Cai et al (2011) present CLIME (A Constrained ℓ_1 Minimization Approach to Sparse Precision Matrix Estimation), which is based on estimating a sparse precision matrix subject to a constraint on the difference between the product of the sample covariance matrix with the estimated precision matrix and the identity matrix.[9] The `bootnet R` package includes a broad range of different network estimation methods, several of which are for GGM estimation, in a framework for bootstrap estimation of network accuracy.[10] For a detailed review of Gaussian graphical model theory and estimation, we refer the reader to Shutta et al (2022).[11]

Clearly, there is no shortage of options for estimating a GGM; this is both a blessing and a curse. Estimating a GGM using these packages requires several decisions with regard to tuning parameter selection, choice of scoring criteria for model selection, and selection of hyperparameters for these scoring criteria. The estimated GGM may be highly

sensitive to these choices, making it difficult to compare GGMs across studies and assess reproducibility.[12-14] Because it is impossible to know *a priori* which approach is best for a given problem, researcher bias toward use of a particular "favorite method" can have a large impact on the estimation and interpretation of a GGM.

Ensemble methods are a broad class of statistical approaches which follow the general principle of combining several different candidate models to generate a single ensemble model.[15,16] One such method is the Super Learner approach of van der Laan et al.[17] Super Learner uses an internal cross-validation scheme to estimate a convex combination of candidate algorithms ("learners") that minimizes a user-defined loss function. Large-sample properties of the Super Learner are established by comparison to the expected loss (i.e., risk) of an oracle model, which is the best model among all possible convex combinations given the true data generating process. Under mild conditions on the loss function and the set of candidate learners, the performance of the Super Learner ensemble model is asymptotically equivalent to that of the oracle model in the sense of risk minimization.[17-19]

Here we develop SpiderLearner, a network estimation tool that applies the Super Learner approach to GGM estimation by optimizing a likelihood-based loss function via cross-validation. Our approach improves GGM estimation by circumventing the complicated decision-making burden described above. SpiderLearner considers a library of candidate GGM estimation methods and constructs the optimal convex combination of their results, eliminating the need for the researcher to make arbitrary decisions in the estimation process. To the best of our knowledge, we are the first to propose, evaluate, and apply a loss-based ensemble learning method for GGM estimation. In particular, our novel approach applies a data-driven loss function to create an ensemble from candidate precision matrix estimators, integrating a cross-validation scheme to honestly evaluate performance and construct the optimal convex combination according to that loss function. SpiderLearner is implemented in a user-friendly R package. We used this package to evaluate the performance of SpiderLearner rigorously under a range of network settings, including four topologies, two densities, and four different dimensionalities (n / p ratios). Under a variety of metrics, including out-of-sample likelihood, SpiderLearner outperforms each of the candidates as well as a naive "simple mean" ensemble giving equal weight to each candidate. We demonstrate that SpiderLearner can discover meaningful biological insights in complex multivariate data by presenting an illustrative application to 13 publicly-available gene expression datasets.[20,21]

The remainder of this manuscript is organized as follows. In Section 2, we propose our novel approach to GGM estimation, including a tailored loss function and cross-validation scheme for combining candidate GGM learners. In Section 3, we describe our simulation studies, error metrics, and results. In Section 4, we present the ovarian cancer data application. We conclude with a brief discussion of alternative approaches, limitations, and areas of future work.

2 | MODEL FORMULATION

Let $\mathbf{X} \sim N_p(\mathbf{0}, \mathbf{\Sigma})$ be a centered p-dimensional multivariate normal random variable, with precision matrix $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$. Under the multivariate Gaussian assumption, a particularly useful relationship holds between the precision matrix $\mathbf{\Theta} = \mathbf{\Sigma}^{-1}$ and the matrix of partial correlations, $\{\rho_{X_i, X_j \mid X_{-i, -j}}\}_{1 \leq i \leq p; 1 \leq j \leq p}$.[2] Let θ_{ij} represent the i, j^{th} element of $\mathbf{\Theta}$; it can be shown that

$$\rho_{X_i, X_j \mid X_{-i, -j}} = -\frac{\theta_{ij}}{\sqrt{\theta_{ii}\theta_{jj}}} \tag{1}$$

Equation (1) shows that estimating a GGM of partial correlations is equivalent to estimating the precision matrix $\mathbf{\Theta}$. A common approach to this estimation problem in cases where the maximum likelihood estimate (MLE) is not well-defined or where sparsity is desired for the sake of interpretability is the graphical lasso (glasso).[3-5] The glasso estimates a sparse precision matrix by optimizing the penalized log likelihood function:

$$\ell(\mathbf{\Theta}) = \log \det \mathbf{\Theta} - \text{trace}(S\mathbf{\Theta}) - \lambda \|\mathbf{\Theta}\|_1 \tag{2}$$

Here S is the sample covariance matrix and $\lambda > 0$ is a tuning parameter, with higher values of λ leading to sparser estimates of $\mathbf{\Theta}$.

As described in the Introduction, several open-source software resources are available to estimate GGMs using the approach in Equation (2) (e.g., Zhao et al.,[7] Epskamp et al.[10]) or other methods (e.g., Cai et al.[9]). Here, based on the Super Learner framework of van der Laan et al.,[17] we present an ensemble approach that estimates a GGM by finding the optimal convex combination of a set of candidate GGM estimates obtained from tools such as these. The foundations for a Super Learner-type method are (i) the specification of a library of candidate algorithms, (ii) the specifation of a loss function, and (iii) the implemention of a cross-validation scheme to determine the optimal convex combination of the candidates.[22] We introduce the foundations of our method similarly, but focus first on (ii) and (iii). We address (i) when describing our simulation study design; in brief, we considered several methods based on the graphical lasso with different tuning parameter selection criteria, the MLE, and CLIME.[9]

To develop the loss function for SpiderLearner, we begin by supposing that we have a library of M different candidate methods for estimating GGMs. We use K-fold cross-validation to estimate a weighted combination of these M estimates, in the spirit of the Super Learner approach.[17] First, we partition the data X into K folds of approximately equal size $\sim n / K$. We next apply the precision matrix estimator K times for each of the M methods; each time, data from the k^{th} fold is withheld $(k = 1, \ \ldots \ , K)$ as the test set while the remaining $(K - 1)$ of the folds serve as the training set. Let X_k be the k^{th} fold of the dataset X, X_{-k} be the

remainder of the dataset X with the k^{th} fold withheld and let $\widehat{\boldsymbol{\Theta}}_m^{(-k)}$ be the precision matrix estimate for method m trained on X_{-k}. For the k^{th} fold, we define the estimator $\boldsymbol{\Theta}_{SL}^{(-k)}$, which is a function of weights $\boldsymbol{\alpha} = (\alpha_1, \ \ldots \ , \alpha_M)$, as:

$$\boldsymbol{\Theta}_{SL}^{(-k)}(\boldsymbol{\alpha}) = \alpha_1 \widehat{\boldsymbol{\Theta}}_1^{(-k)} + \alpha_2 \widehat{\boldsymbol{\Theta}}_2^{(-k)} + \cdots + \alpha_M \widehat{\boldsymbol{\Theta}}_M^{(-k)}; \ \sum_{m=1}^{M} \alpha_m = 1; \alpha_m \geq 0 \tag{3}$$

For simplicity of notation, we denote $\boldsymbol{\Theta}_{SL}^{(-k)}(\boldsymbol{\alpha})$ as $\boldsymbol{\Theta}_{SL}^{(-k)}$ below, keeping in mind that the dependence on $\boldsymbol{\alpha}$ is implied throughout.

Let n_k be the number of observations in the k^{th} fold and let $X_k^{(i)}$ be the i^{th} observation in fold k. We define the loss in the fold k as $Q_k(\boldsymbol{\alpha})$, the negative average log likelihood of $\boldsymbol{\Theta}_{SL}^{(-k)}$ evaluated on the withheld data X_k:

$$Q_k(\boldsymbol{\alpha}) = -\frac{1}{2} \log \left| \boldsymbol{\Theta}_{SL}^{(-k)} \right| + \frac{1}{2n_k} \sum_{i=1}^{n_k} (X_k^{(i)})^T \boldsymbol{\Theta}_{SL}^{(-k)} X_k^{(i)} \tag{4}$$

Note that minimizing the negative average log likelihood for $Q_k(\boldsymbol{\alpha})$ is equivalent to maximizing the total likelihood. Equation (4) is based on the the average rather than total log likelihood for numerical stability in optimization, and we use the negative log likelihood rather than the positive so that the problem is framed in the context of minimum loss as in Super Learner.[17]

We use these foundations to develop a loss function in terms of the coefficients $\boldsymbol{\alpha} = \alpha_1, \ \ldots \ , \alpha_M$. Let $\bar{Q}(\boldsymbol{\alpha})$ be the average loss across K folds:

$$\bar{Q}(\boldsymbol{\alpha}) = \frac{1}{K} \sum_{k=1}^{K} Q_k(\boldsymbol{\alpha}) \tag{5}$$

The K-fold cross-validated coefficient estimator $\widehat{\boldsymbol{\alpha}}$ is the value of $\boldsymbol{\alpha}$ that minimizes Equation (5), subject to the constraints of the convex combination:

$$\widehat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}: \sum_{m=1}^{M} \alpha_m = 1; \alpha_m \geq 0}{\operatorname{argmin}} \left\{ \frac{1}{K} \sum_{k=1}^{K} \left(-\frac{1}{2} \log(\left| \boldsymbol{\Theta}_{SL}^{(-k)} \right|) + \frac{1}{2n_k} \sum_{i=1}^{n_k} (X_k^{(i)})^T \boldsymbol{\Theta}_{SL}^{(-k)} X_k^{(i)} \right) \right\} \tag{6}$$

Standard constrained optimization algorithms such as those implemented in the `solnp` function from the R package `Rsolnp`[23] can be used to find the coefficients $\widehat{\boldsymbol{\alpha}}$ that solve

Equation (6). Once these coefficients have been found, we complete the process by running the original M candidate methods again using the full dataset X, obtaining estimates $\widehat{\boldsymbol{\Theta}}_1, \ldots, \widehat{\boldsymbol{\Theta}}_M$. We then use these estimates to construct the SpiderLearner estimate of the precision matrix as:

$$\widehat{\boldsymbol{\Theta}}_{SL} = \sum_{m=1}^{M} \hat{\alpha}_m \widehat{\boldsymbol{\Theta}}_m \tag{7}$$

A diagram of this workflow for $M = 4$ estimation methods and $K = 5$ cross-validation folds is shown in Figure 1. The choice of K may depend on a variety of factors including sample size and number of variables (i.e., dimensionality of the problem); in practice, $K = 5$ and $K = 10$ have demonstrated generally good balance in the bias-variance tradeoff.[24] We discuss the choice of K further in the Supporting Information (Supporting Appendix 4).

The large-sample properties of Super Learner derived by van der Laan et al.[17] require a bounded loss function. Our loss function $\bar{Q}(\boldsymbol{\alpha})$ (Equation 5) is not bounded; therefore, it is not clear if the large-sample oracle results of van der Laan et al.[17] apply with the log likelihood-based loss function evaluated on multivariate normal data. Polley et al.[18] note that oracle results also hold for certain types of unbounded loss functions as described in van der Vaart et al.;[25] however, it is not straightforward to formally show that $\bar{Q}(\boldsymbol{\alpha})$ meets the necessary criteria (Supporting Appendix 1). We note this as an area for future work, while observing that in practice the log likelihood is often used as a loss function for Super Learner estimation (e.g., Petersen et al.,[26] Balzer et al.[27]), and that the log likelihood loss is provided as part of the standard implementation of the `SuperLearner` R package.[28] We additionally explored a transformation of the log likelihood loss that permits the application of the oracle results of van der Laan et al.[17] We observed similar performance between the original and transformed loss functions in a simulation setting (Supporting Appendix 1). Finally, we note that the SpiderLearner estimator produces a positive definite precision matrix if each of the candidate methods produces a positive definite precision matrix; this point is discussed further in Supporting Appendix 2.

3 | SIMULATION

3.1 | Design of simulation study

To assess the performance of the SpiderLearner algorithm, we conducted several simulation studies with varying dimensionality (Table 1A). The dimension of a GGM is typically described in terms of the number of samples n and the number of variables p included in the model. Importantly, p is *not* the number of parameters in the model; the precision matrix corresponding to the GGM has $q = p * (p-1)/2 + p$ unique entries that need to be estimated. Dimensionality thus quickly becomes a major factor in estimation even if a GGM does not include very many predictors.

We explored four different network topologies in our simulations: random, small world, scale-free, and hub-and-spoke. Each topology has unique characteristics that may be relevant for biological data. Topologies are discussed in detail in Supporting Appendix 3. We explore each topology for different edge densities, where the edge density of a graph is defined as the number of edges divided by $p(p-1)/2$, the number of possible edges on p nodes. For each of the four topologies, we simulated networks with two different density levels (low density: approximately 6% dense, and high density: approximately 20% dense). A visualization of the simulated networks is shown in Supporting Figure 3. Topologies and densities considered are shown in Table 1B.

The simulation workflow, shown in Figure 2, consisted of (i) designing gold-standard networks corresponding to each topology and density, (ii) assigning edge weights to the network based on an observed distribution of partial correlations from a real biological dataset and converting the associated weighted adjacency matrices to valid precision matrices, (iii) sampling multivariate normal data based on the precision matrices from (ii), (iv) using various methods, including our proposed ensemble method, to estimate the original network from the sampled data, and (v) comparing the estimated network to the original gold standard used to generate the data. In step (i), the `igraph` R package[29] was used to simulate gold-standard networks (Table 1B). In step (ii), we created a realistic edge weight distribution by using metabolomics data from the CATHeterization GENetics (CATHGEN) biorepository as a starting point.[30] The CATHGEN biorepository consists of data from a prospectively-collected clinical study of ~ 10,000 participants undergoing cardiac catheterization with scheduled annual followup at Duke University Hospital; further details of the study population have previously been published.[30] Details on how we used these data to assign edge weights are provided in the Supporting Information (Supporting Appendix 3). In step (iii), to sample network data from the gold-standard networks, we inverted each gold standard precision matrix $\boldsymbol{\Theta}$ to find the corresponding covariance matrix $\boldsymbol{\Sigma}$, then simulated a sample of size n by drawing $X_1, \ldots, X_n \sim MVN(\mathbf{0}, \boldsymbol{\Sigma})$. In step (iv), we estimated precision matrices from this sample in three ways: (i) by applying each of 9 candidate methods individually (ii) by using a simple mean ensemble model in which each candidate is weighted equally, and (iii) by using SpiderLearner (Figure 2). In step (v), we compared estimated precision matrices to the original, data-generating gold-standard precision matrices in order to assess performance.

The 9 candidate methods are shown in Figure 2. The graphical lasso is the foundation of Candidate Methods 1, 2, 3, 6, and 7. These methods use the `huge` and `huge.select` functions from the R package `huge` with the `glasso` method, which corresponds to the original graphical lasso.[3,6,7] The difference between these methods is the choice of scoring criterion used to select the tuning parameter (λ in Equation 2). The first criterion is the extended Bayesian information criterion (eBIC), which optimizes a BIC-type quantity tuned by a hyperparameter γ, where $\gamma = 0$ corresponds to a standard BIC measure and $\gamma = 0.5$ is a typical default value for graphical modeling.[12,31] In the `huge.select` function, γ can be adjusted using the `ebic.gamma` argument. Candidate Methods 1 and 2 apply this criterion with `ebic.gamma` = 0 and `ebic.gamma` = 0.5, respectively. Candidate Method 3 applies a criterion called the rotation information criterion (RIC), which is based on a permutation

strategy that generates a null distribution for comparison.[7,12] Candidate Methods 6 and 7 use a criterion called the stability approach to regularization selection (StARS), which is a subsampling approach.[12,32] One of several hyperparameters that can be selected using StARS is a threshold β that relates to the amount of variability tolerated across subsamples.[32] In the `huge.select` function, β can be adjusted using the `stars.thres` argument. Candidate Method 6 applies the StARS criterion with `stars.thres` = 0.05 and Candidate Method 7 applies it it with `stars.thres` = 0.1. Candidate Method 4 is the hub graphical lasso, which is an extension of the original graphical lasso that can effectively model hub structures in networks and is implemented in the `hglasso` R package.[8] Candidate Method 5 is the MLE, that is, the inverse of the sample covariance as computed with the `cov` function in base R. Candidate Methods 8 and 9 are similar to Candidate Methods 1 and 2; they also use the original graphical lasso along with an eBIC scoring criterion, but are implemented in the `qgraph` R package.[33] A difference between the `qgraph` implementation and the `huge` implementation is in the default range of tuning parameters λ considered. Let λ^* be the smallest value of λ that creates an empty graph; `huge` uses a logarithmic sequence of ten candidate λ values between $0.1\lambda^*$ and λ^*, while `qgraph` uses a larger logarithmic sequence of length 100 between $0.01\lambda^*$ and λ^*.[7,33] While Candidate Method 5, the MLE, is typically well-defined in Simulations A-C (barring multicollinearity), it is not in Simulation D, where $n < p$. Therefore, Simulation D excludes Candidate Method 5.

We limited our investigation to these 9 candidates because they are common approaches[11] that are computationally efficient. We remind the reader that our goal in this work is to demonstrate the improved performance of SpiderLearner over a set of candidates, not to evaluate the individual performance of all candidates. However, as demonstrated in Supporting Appendix 5, SpiderLearner can easily accommodate other methods, such as CLIME. We also note that as a benchmark for SpiderLearner, we calculated a simple mean model as the average of the nine candidate methods, giving equal weight of 1/9 to each candidate.

We used several different metrics to compare estimated precision matrices to the gold standards and assess performance. Let $\widehat{\mathbf{\Theta}}$ be an estimate of the true $p \times p$ precision matrix $\mathbf{\Theta}$, and let $\hat{\theta}_{ij}$ and θ_{ij} represent the corresponding elements of each. We define the **error matrix** Δ as $\widehat{\mathbf{\Theta}} - \mathbf{\Theta}$, and refer to its i, j^{th} element as δ_{ij}. Although the true precision matrix is symmetric, the estimated matrix may not be; a notable example of possible asymmetry is in the graphical lasso algorithm.[34] Therefore, we considered every element of the error matrix Δ rather than just upper or lower triangular components.

One area of interest is to assess error in the estimated edge weights (partial correlations) in the GGM. Because these edge weights follow directly from the estimated precision matrix, we begin by focusing our efforts on quantifying error in the precision matrix itself. The first metric is based on the size of Δ as assessed by the Frobenius norm:

$$\|\Delta\|_F = \sqrt{\sum_{i=1}^{p}\sum_{j=1}^{p}\delta_{ij}^2} \tag{8}$$

To obtain a quantity that can be compared across topologies, we scale $\|\Delta\|_F$ by the Frobenius norm of the true precision matrix, $\|\boldsymbol{\Theta}\|_F$, defining the **relative Frobenius norm (RFN)** as

$$RFN = \frac{\|\Delta\|_F}{\|\boldsymbol{\Theta}\|_F} \tag{9}$$

We are interested in the generalizability of SpiderLearner to independent datasets. For this purpose, we assessed the **out-of-sample log likelihood** of each estimated precision matrix on a new, independent sample of the same size generated from the same gold-standard precision matrix.

We also considered a number of additional diagnostics including element-wise bias, mean squared error (MSE), overall sensitivity and specificity, and the **matrix RV coefficient**, a matrix analogue of a correlation coefficient implemented in the R package `MatrixCorrelation`.[35,36] Details are provided in Supporting Appendix 3.

3.2 | Results of simulation study

We conducted 100 iterations for each of the eight network topologies in each of Simulations A-D. Primary results for Simulation A (the lowest p / n ratio) and Simulation D (the highest p / n ratio) are presented here; additional results for Simulations A and D and full results for Simulation B and Simulation C are presented in the Supporting Information (Supporting Appendix 3; Supporting Figures 6-9).

In Simulation A, the sample size is much larger than number of features and parameters estimated ($n >> p, q$). The average ensemble weights estimated by SpiderLearner over the 100 iterations in this setting are shown in Table 2A. SpiderLearner selected at least three different methods to have nonzero weights for each topology, demonstrating that combining multiple candidate algorithms is indeed important from a likelihood-based loss perspective. For every topology, `qgraph-ebic-0` and the inverse sample covariance (ie, MLE) were included in the combination, although the weights varied broadly by topology, with `qgraph-ebic-0` weights ranging from 0.03 for the low-density hub-and-spoke topology to 0.4 for the low-density scale-free topology and the MLE weights ranging from 0.25 for the low-density scale-free graph topology to 0.59 for the high-density random graph topology. The `hglasso` was included in seven out of eight topologies (excluding the low-density scale-free topology), again with broadly varying weights (0.10–0.63). The `glasso-ebic-0` was selected for minor contributions in the low-density scale-free case (0.34) and the high-density scale-free case (0.07). The `glasso-ebic-0.5` also contributed slightly in the

high-density scale-free case (0.07). The `glasso-ric`, `glasso-stars-0.05`, `glasso-stars-0.1`, and `qgraph-ebic-0.5` methods were weighted zero for all cases.

The average value of the RFN in Simulation A is shown in Figure 3. The performance of each candidate method varied widely according to this metric, emphasizing the importance of our ensemble approach. SpiderLearner performed better than the individual candidates and better than the simple mean ensemble model across all settings considered. The performance as assessed by out-of-sample log likelihood is shown in Figure 3. Again, SpiderLearner performed well relative to each of the candidates and the simple mean of the candidates. Performance as assessed by bias, MSE, sensitivity and specificity, and matrix RV coefficient are shown in the Supporting Information (Supporting Figures 4 and 5, Supporting Tables 2 and 3). Generally, SpiderLearner is comparable or better than other methods in terms of these measures; a detailed analysis is presented in Supporting Appendix 3.

In Simulation D, the sample size is lower than the number of features, which is much lower than the number of parameters estimated ($n < < p, q$). The average value of the ensemble weights estimated by SpiderLearner in this setting are shown in Table 2B. SpiderLearner selected at least four of the candidate methods in every case. Interestingly, `glasso-stars-0.05` and `glasso-stars-0.1` both contributed to the ensemble in all eight cases in Simulation D, but they were not selected in any case in Simulation A. Similarly, the hub graphical lasso was a highly-weighted candidate in most cases in Simulation A, but not in Simulation D. These observations are further evidence of the importance of considering multiple methods when estimating a GGM; the performance (in the log-likelihood sense) of estimates from different methods varies broadly based on the characteristics of the true underlying network.

The average value of the RFN in Simulation D is shown in Figure 3. SpiderLearner performed comparably to the `qgraph-ebic-0` candidate method and `qgraph-ebic-0.5`, two methods which were highly weighted in the ensemble (Table 2b). The out-of-sample log likelihood performance is shown in Figure 3. SpiderLearner again performed well when compared to the remainder of the methods. The hub graphical lasso had notably lower out-of-sample log likelihood than the other candidates, suggesting overfitting in this setting. Performance as assessed by bias, MSE, sensitivity and specificity, and matrix RV coefficient are shown in the Supporting Information (Supporting Figures 10 and 11 and Supporting Tables 10 and 11). Generally, SpiderLearner again performed comparably to or better than the other candidates. In this setting, SpiderLearner was more sensitive but less specific than other methods (Supporting Tables 10 and 11). A detailed analysis is presented in Supporting Appendix 3.4.

Results for Simulations B and C led to similar conclusions as above. In particular, as the p / n ratio increased, we consistently observed that the SpiderLearner model resisted overfitting. In contrast, some other approaches, in particular the MLE, tended to overfit the data, resulting in strong in-sample performance but weaker out-of-sample performance (Supporting Figures 6-9). We attribute this advantage of SpiderLearner to the use of K-fold cross-validation and the incorporation of multiple candidate learners.

We also expanded our library of candidate estimators beyond the graphical lasso and MLE candidates to explore performance with CLIME.[9] As shown in Supporting Appendix 5, CLIME is a strong candidate learner that outperforms other candidates for low p / n ratios. However, as dimensionality increases, the performance of CLIME weakens relative to the other candidates and SpiderLearner. These results provide further evidence towards our motivation for developing SpiderLearner: the best choice of GGM estimation method varies broadly depending on the features of the true network, and ensembling a library of candidate methods gives the SpiderLearner algorithm the strengths of each algorithm without having to choose just one method *a priori*.

A note on the MLE as the precision matrix estimator: In simulation settings A,B, and C, the sample size n is larger than the number of predictors p in the model, meaning that the sample covariance matrix is non-singular, except in the case of multicollinearity. The sample covariance matrix is the MLE for the population covariance matrix, and because inversion of a non-singular matrix is a continuous function, the inverted sample covariance matrix is the MLE for the population precision matrix.[37,38] It is notable, then, that the likelihood-based SpiderLearner model gives weight to models other than the MLE, and that other individual regularized algorithms perform better than the MLE according to the relative Frobenius norm, matrix RV coefficient, and out-of-sample likelihood. We hypothesized that this phenomenon was related to the sparsity of the underlying network. To investigate, we ran the SpiderLearner algorithm on an Erdös-Renyí random graph with a variety of densities (0.05, 0.1, 0.25, 0.5, 0.75, 1) with 30 iterations for each density. As hypothesized, the weight of the MLE in the ensemble model increases with the density of the graph, as shown in Supporting Figure 12. These results suggest that even though the ensemble loss function does not incorporate a shrinkage penalty, it is still advantageous from the likelihood-based perspective to shrink estimates of small precision matrix entries to zero in the case where the population precision matrix is sparse. The takeaway is that shrinkage methods can improve out-of-sample performance even in low-dimensional cases, which is consistent with results observed in the original LASSO publication.[39]

3.2.1 | Practical questions in applying SpiderLearner—Finally, we refer the reader to the Supporting Information for detailed simulations that we have conducted regarding three practical questions in this methodology: (i) how to select K in the K-fold cross-validation (Supporting Appendix 4), (ii) how to select the library of candidates (Supporting Appendix 5), and (iii) the stability of `Rsolnp` in estimating the coefficients α and the overall ensemble model (Supporting Appendix 6). Code for all simulations is available at https://github.com/katehoffshutta/SpiderLearnerWorkflow.

4 | APPLICATION

We applied SpiderLearner to analyze 13 ovarian cancer gene expression datasets from the curatedOvarianData collection of Ganzfried et al.[21] One of these datasets ($N = 260$ participants) was used to estimate a precision matrix using SpiderLearner; twelve other diverse investigations including a total of 1753 participants were used as *independent* validation datasets to evaluate SpiderLearner's performance, as measured by the likelihood

of the observed data evaluated at the estimated precision matrix. All data are publicly available through the R package `curatedOvarianData` via Bioconductor.[21]

The training dataset ("Yoshihara dataset") consists of 260 late-stage ovarian cancer patients with gene expression data for 20106 genes, obtained via microarray experiments by Yoshihara et al.[20] Characteristics of this dataset have been previously described.[20] Briefly, participants with advanced stage high-grade serous ovarian cancer who underwent debulking surgery followed by chemotherapy were followed for up to ten years. Basic characteristics and references to original publications for all 13 datasets are shown in Table 3.

Yoshihara et al[20] present a 126-gene signature of high-risk ovarian cancer based on overall survival, defined as time from primary surgery to death due to ovarian cancer. 111 of these 126 genes were available across all 13 datasets; in this application, we constructed a GGM on these 111 genes. To do so, we applied SpiderLearner using $K = 10$ folds for model training and a library of the nine candidate GGM estimation methods described in Figure 2. The weights selected by SpiderLearner were 0.67 for `hglasso`, 0.19 for `glasso-ebic-0`, 0.08 for `qgraph-ebic-0`, 0.07 for the MLE, and zero for the remainder of the candidate algorithms.

We evaluated the out-of-sample performance of SpiderLearner and of each candidate algorithm in two ways. First, we used 10-fold internal cross-validation on the training dataset. We partitioned the Yoshihara dataset into ten folds, each of which was withheld in turn; a precision matrix was estimated on the remaining 9 folds and its likelihood evaluated on the withheld data. Boxplots of the cross-validated out-of-sample log likelihood for SpiderLearner and each of the nine individual GGM estimation methods are shown in Figure 4a. SpiderLearner performs better than, or comparably to, the best candidate models in its library according to this criterion.

Second, we used the 12 independent validation datasets to demonstrate the ability of SpiderLearner to outperform competitors across a diverse set of test data that includes multiple platforms, different patient characteristics, and varying severity of disease (Table 3). The precision matrices trained in the Yoshihara dataset by SpiderLearner and by each of the 9 candidate methods were applied to all 12 validation datasets in turn, with performance measured by evaluating the value of the log likelihood at the estimated precision matrix. Boxplots of the relative performance of each model in the 12 independent validation datasets are shown in Figure 4B,C. Performance is measured in terms of percent difference in log likelihood relative to the best performing model of the 9 candidates and SpiderLearner. Again, SpiderLearner performs better than, or comparably to, the best candidates.

Importantly, the fact that SpiderLearner does not substantially outperform the best candidate algorithms is not an argument against its utility. Indeed, this is exactly what SpiderLearner is designed to do: from a set of input candidate algorithms, it can demonstrably select the best in a data-driven fashion. For example, in this ovarian cancer dataset, the hub graphical lasso performs quite well; however, our simulation studies showed that in some cases it is prone to overfitting (see results for Simulation D, Figure 3). SpiderLearner is able to distinguish

between these situations, weighting the hub graphical lasso heavily in this application but assigning little weight in Simulation D (Table 2B).

Biological interpretation of the GGM:

Having assessed the statistical performance of SpiderLearner, we next turn to a biological interpretation of the estimated network. Networks are often interpreted in terms of communities, which are clusters of nodes that are highly connected to each other and weakly connected to the remainder of the graph.[40] Community detection is a useful way to identify meaningful patterns in graphical models. Earlier work in bipartite networks of single nucleotide polymorphisms (SNPs) and genes has demonstrated that hubs within communities ("local hubs") are enriched for disease-associated SNPs.[40] We hypothesized that local hubs in GGM communities might also be key players in the functional processes reflected by a network, inspiring us to investigate these genes in the SpiderLearner-estimated ovarian cancer GGM.

We began by using the `cluster_fast_greedy` community detection algorithm as implemented in the `igraph` R package[29,41] to detect communities in the SpiderLearner-estimated network from the Yoshihara dataset. Next, we identified local hubs, defined as the gene in each community with the highest hub score, by applying the `hub_score` function of the `igraph` R package to the adjacency matrix of each community.[29,42]

Figure 4d shows the community structure and local hubs of the network. Six genes were identified as local hubs: *N4BP2L2, NCKAP1L, PARVA, RAD17, RCOR3*, and *RPS21*. Notably, all six of these genes have important biological function. Previous literature links their expression levels to processes such as cell proliferation and immune system function, with implications for studying the development, progression, and treatment of cancer. *N4BP2L2* encodes a protein known as NEDD4 Binding Protein 2 Like 2.[43] There is evidence that N4BP2L2 is involved in neutrophil deficiency (neutropenia), participating in transcriptional regulation of a neutrophil production pathway.[43] *NCKAP1L* has recently been identified as a novel tumor microenvironment-related biomarker in luminal breast cancer, but has not been previously studied extensively in relation to ovarian cancer.[44] Deficiency in NCKAP1L has been reported to be associated with a novel syndrome involving immune system dysregulation in humans, and loss-of-function experiments in zebrafish models showed reduced *NCKAP1L* expression was associated with diminished neutrophil response to the site of tail fin injury.[45] *PARVA* is an oncogene that has been implicated in breast cancer, colorectal cancer, lung adenocarcinoma, and melanoma.[46] Alpha-parvin, the protein encoded by PARVA, forms a complex with integrin-linked kinase (ILK) and particularly interesting new Cys-His protein 1 (PINCH-1) that is an integral component of cell survival and is related to cell shape modulation, cell motility, and cell spreading.[47,48] *RAD17* encodes a protein that is related to checkpoint signaling in the cell cycle; RAD17 expression is oscillatory, and engineered stabilization of RAD17 results in disrupted checkpoint signalling and consequent diminished re-entry into the cell cycle.[49] *RCOR3* encodes a protein called CoREST/REST corepressor 3 which is a paralog of RCOR1, a protein which works together with lysine-specific demethylase 1 (LSD1) in epigenetic regulation of cell fates.[50] Upadhyay et al (2014) demonstrate that RCOR3 is

recruited to target genes by LSD1 along with a protein called growth factor independent 1B transcriptional repressor (GFI1B), decreasing histone demethylation and thus de-repressing target gene expression. It has been shown that LSD1 represses tumor suppressor gene expression in oncogenesis; increases in RCOR3 expression could attenuate this contribution to oncogenesis.[50] *RPS21* encodes a ribosomal protein that has been implicated as a diagnostic and prognostic biomarker for prostate cancer; *in vitro* studies of RPS21 along with another ribosomal protein (RPL22L1) showed that diminished expression of RPS21/RPL22L1 via shRNA knockdown results in decreased cell proliferation, migration, and invasion and increased apoptosis in prostate cancer cells.[51] More recently, RPS21 has also been shown to play a similar role in osteosarcoma via MAPK signaling.[52] Differential expression of RPS21 has also been reported as part of a larger study on MHC class 1 proteins involved in cisplatin resistance in human ovarian cancer cells.[53]

The importance of each of these genes and their downstream products suggests that GGMs estimated by SpiderLearner can be used in practice to provide additional insights into existing literature as well as to identify targets of interest for future study.

5 | DISCUSSION

In this work, we propose and evaluate a novel method for Gaussian graphical model (GGM) estimation: SpiderLearner, an ensemble method that builds a convex combination of precision matrix estimates from a library of candidate estimation methods. In a wide variety of simulation settings, SpiderLearner consistently performed comparably to or better than each of the candidate methods according to a variety of metrics including the relative Frobenius norm of the difference between the estimated and true matrices and the out-of-sample likelihood. Importantly, some of the individual candidate methods performed quite poorly; since common practice *a priori* is to simply choose one of the candidate methods at will, our ensemble method provides a considerable advantage for practical use. This is apparent in our application to ovarian cancer data, where SpiderLearner outperforms candidate methods in both internal cross-validation and external validation on independent datasets and is able to identify genes with biological relevance in cancer.

The superior performance of SpiderLearner is no coincidence; there are a number of features in the design of our method that contribute to its success. Under mild conditions, loss-based ensemble models such as SpiderLearner enjoy desirable large sample properties, including comparable performance to an oracle estimator.[17,25] Incorporating K-fold cross-validation in SpiderLearner reduces the risk of overfitting, an especially important point in biological applications where generalizability to external datasets is paramount. The data-driven nature of the SpiderLearner model selection process reduces the potential for human bias to interfere with honest network estimation. A researcher might inadvertently choose one of the many existing GGM estimation methods based on limited awareness of options, ease of implementation, a “favorite method” from previous use, or a method that produces desirable results for a publication. By developing a likelihood-based loss function tailored to this problem setting and applying it in SpiderLearner, we provide an alternative approach that circumvents these issues.

New methods for GGM estimation are being continually developed and assessed. For example, Lartigue et al.[54] conducted an extensive simulation study on GGM estimation for high-dimensional data with small sample sizes and presented a composite procedure that uses a likelihood criterion to select a GGM. Methods such as these that are specific to particular research settings are areas for further development. An advantage of SpiderLearner is that such methods, when developed, can be included as candidate models in the ensemble library.

Ongoing advances have been proposed to improve the applicability and reproducibility of network estimation methods. Steinley et al.[55] propose a Monte Carlo-based method for generating confidence intervals for network statistics, allowing a researcher to assess whether a network property such as edge presence or node centrality differs from that expected by random chance. Epskamp et al. (2018)[10] present a bootstrap-based approach which allows researchers to investigate the variability of an estimated network. Epskamp et al. (2020)[56] develop a network meta-analysis framework that permits integration of estimated networks across multiple studies. Again, each of these methods relies on the use of an initial estimation algorithm, remaining sensitive to the many choices that the researcher must make during the estimation process. Consequently, coupling advances such as these with SpiderLearner would contribute substantially to improved reproducibility and generalizability in GGM estimation and is an important area of future work.

One alternative to ensemble GGM estimation is to apply bootstrap resampling to estimate a collection of precision matrices which are then averaged to create an overall estimate. Variants of this concept have been proposed by, for example, Meinshausen and Bühlmann (2010),[57] Li et al. (2013),[58] and Cai et al. (2016).[59] A variation of this approach applied to differential network analysis can be found in Chen et al. (2022).[60] While both this bootstrap-based approach and SpiderLearner construct a precision matrix as a linear combination of candidate estimates, the two methods are fundamentally different in several key ways. First, the precision matrices comprising the SpiderLearner ensemble are each estimated with different candidate learners, whereas the bootstrap-based ensemble applies a single candidate learner. The bootstrap-based ensemble thus remains subject to the limitation that the user needs to pick a single GGM estimation method and that results are highly sensitive to this choice, as discussed in the Introduction. Second, a key tenet of SpiderLearner is the use of K-fold cross-validation to reduce overfitting. The bootstrap-based ensemble uses a simple average of bootstrap estimates constructed by resampling the entire dataset, increasing the risk of overfitting because no independent assessment of performance is used in model selection. Third, SpiderLearner uses loss-based learning to identify the optimal weights for each candidate precision matrix in the ensemble, whereas the bootstrap-based ensemble is a simple mean in which each bootstrapped precision matrix is equally weighted. Empirical comparison of SpiderLearner with this alternative bootstrap-based approach did, in fact, indicate superior performance by SpiderLearner (Supporting Appendix 7).

In a recent publication, Isvoranu and Epskamp reported the results of a comprehensive simulation study using 13 different GGM estimation methods, 60 different network measures, and a range of true network configurations similar to networks expected in psychology.[14] Isvoranu and Epskamp conclude that the research question should guide the

selection of estimation method, and provide practical guidance on this process. This work, while comprehensive, differs from ours in that the endpoint is still the choice of a single estimation method rather than an ensemble, and that the researcher must choose based on research questions and intuition. In contrast, SpiderLearner uses a data-driven loss-based learning principle to circumvent this choice. A promising area of future work would be to extend SpiderLearner by adding new loss functions that could optimize the ensemble with respect to the different network measures described in Isvoranu and Epskamp.[14]

One limitation of our approach is computational cost; for K-fold cross validation with M candidate models, the time cost of estimating the ensemble model is about $M(K+1)$ times the cost of estimating just one candidate model. Moreover, the number of model parameters to be estimated by each candidate model grows quadratically with the number of predictors included in the network, meaning that the computational cost of the ensemble model can quickly become substantial for larger predictor sets (Supporting Table 1). Because model fitting in each fold is independent, parallelization is a good solution to this problem when multiple cores are available. To address this limitation, we have implemented parallel processing in our `R` package. It is also worth nothing that in our experience, the majority of computing time is dedicated to estimating the candidates, and that the computing time involved in estimating the convex combination is trivial in comparison. We therefore recommend that users wishing to reduce computing time exercise care when selecting the candidate algorithms; for example, the runtime of CLIME was prohibitive of its inclusion in our high-dimensional Setting D ($n = 60$, $p = 100$; Supporting Appendix 5).

A second limitation is that our objective function (Equation 6) may not yield a unique solution. Concretely, this could occur if two candidates estimate the same precision matrix and, thus, have the same cross-validated risk. In Supporting Appendix 6, we explore the stability of the estimated weights $\boldsymbol{\alpha}$ as well as the overall ensemble when using the `Rsolnp` solver, and do not find excessive variability in either case. To allow the user to assess the stability of the numeric solver, the SpiderLearner code includes an option to set a seed for the random fold selection so that users can easily bootstrap their own dataset and assess variability. The SpiderLearner code is also designed such that the control parameters for the `Rsolnp::solnp` function can be applied as input. In this way, the user can tune key aspects of the optimization such as the number of iterations, step size, and tolerance.

A third limitation lies in the rigidity of the convex combination of precision matrices. The same coefficient is applied to every element of each precision matrix in the current ensemble model formulation. A more flexible extension could address this limitation by partitioning matrices into regions determined to be similar across methods (eg, the row and column corresponding to a hub node), fitting a convex combination within each partition, and combining these results to yield the ensemble precision matrix; this is an area of future work.

The past decade has shown numerous advances in GGM estimation, but the burden has still been left on the researcher to determine the specifics of the estimation process. SpiderLearner removes this barrier, enabling researchers to easily construct a likelihood-based optimal combination from a library of candidate methods.

Our simulation studies demonstrate that SpiderLearner outperforms existing approaches, both in terms of accuracy of matrix estimation and out-of-sample likelihood. When applied to ovarian cancer gene expression data, SpiderLearner had better out-of-sample likelihood in internal cross-validation and external validation against 12 independent datasets. Key genes in the SpiderLearner-estimated network have been previously linked to important biological functions related to cancer development and progression, highlighting the practical relevance of this ensemble method. SpiderLearner is available in the R package `ensembleGGM` at https://github.com/katehoffshutta/ensembleGGM, and the code for the simulation and application are available at https://github.com/katehoffshutta/SpiderLearnerWorkflow.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge Subhajit Naskar for his contributions in designing the simulation studies that inspired this work. We thank the participants of the CATHGEN study and of the 13 ovarian cancer studies used in the biological application of this manuscript. Research reported in this manuscript was supported by the National Library of Medicine of the National Institutes of Health under award number R01LM013444. K.H.S. is supported by the National Heart, Lung, and Blood Institute of the National Institutes of Health under award number T32HL007427. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

Funding information

U.S. National Library of Medicine, Grant/Award Number: R01LM013444; National Heart, Lung, and Blood Institute of the National Institutes of Health, Grant/Award Number: T32HL007427

DATA AVAILABILITY STATEMENT

The ovarian cancer gene expression data that support the findings of this study are openly available in the `curatedOvarianData` R package.[21] The CATHGEN data used to generate the distribution of partial correlations used in the simulation study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

ABBREVIATIONS:

CATHGEN	CATHeterization GENetics biorepository
eBIC	extended Bayesian information criterion
GGM	Gaussian graphical model
MLE	maximum likelihood estimate
MSE	mean squared error
RIC	rotation information criterion
RFN	relative Frobenius norm

StARS stability approach to regularization selection

REFERENCES

1. Uhler C Gaussian Graphical Models: An Algebraic and Geometric Perspective. 2017. https://arxiv.org/abs/1707.04345
2. Lauritzen SL. Graphical Models. 17. New York: Clarendon Press; 1996.
3. Friedman J, Hastie T, Tibshirani R. Sparse inverse covariance estimation with the graphical lasso. Biostatistics. 2008;9(3):432–441. [PubMed: 18079126]
4. Yuan M, Lin Y. Model selection and estimation in the gaussian graphical model. Biometrika. 2007;94:19–35.
5. Banerjee O, Ghaoui LE, d'Aspremont A. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. J Mach Learn Res. 2008;9:485–516.
6. Witten DM, Friedman JH, Simon N. New insights and faster computations for the graphical lasso. J Comput Graph Stat. 2011;20(4):892–900.
7. Zhao T, Liu H, Roeder K, Lafferty J, Wasserman L. The huge package for high-dimensional undirected graph estimation in R. J Mach Learn Res. 2012;13:1059–1062. [PubMed: 26834510]
8. Tan K, London P, Mohan K, Lee S, Fazel M, Witten D. Learning graphical models with hubs. J Mach Learn Res: JMLR. 2014;15:3297–3331. [PubMed: 25620891]
9. Cai T, Liu W, Luo X. A constrained l1 minimization approach to sparse precision matrix estimation. J Am Stat Assoc. 2011;106(494):594–607.
10. Epskamp S, Borsboom D, Fried EI. Estimating psychological networks and their accuracy: a tutorial paper. Behav Res Methods. 2018;50(1):195–212. [PubMed: 28342071]
11. Shutta KH, De Vito R, Scholtens DM, Balasubramanian R. Gaussian graphical models with applications to omics analyses. Stat Med. 2022;41(25):5150–5187. [PubMed: 36161666]
12. Wysocki AC, Rhemtulla M. On penalty parameter selection for estimating network models. Multivar Behav Res. 2021;56(2):288–302.
13. Shutta K, Naskar S, Rexrode K, Scholtens D, Balasubramanian R. Estimation of Metabolomic Networks with Gaussian Graphical Models. 2020 https://raji-lab.github.io/News/ENAR2020.pdf; ENAR 2020 Conference, Online
14. Isvoranu AM, Epskamp S. Which estimation method to choose in network psychometrics? Deriving guidelines for applied researchers. Psychol Methods. 2021. 10.1037/met0000439
15. Wolpert DH. Stacked generalization. Neural Netw. 1992;5(2):241–259.
16. Breiman L. Stacked regressions. Machine Learn. 1996;24(1):49–64.
17. van der Laan MJ, Polley EC, Hubbard AE. Super learner. Stat Appl Gene Mole Biol. 2007;6(1):25. https://www.degruyter.com/document/doi/10.2202/1544-6115.1309/html
18. Polley EC, van der Laan MJ. Super Learner in Prediction. 2010.
19. Polley EC, Rose S, van der Laan MJ. Super learning. 2011 43–66.
20. Yoshihara K, Tsunoda T, Shigemizu D, et al. High-risk ovarian cancer based on 126-gene expression signature is uniquely characterized by downregulation of antigen presentation pathway. Clin Cancer Res. 2012;18(5):1374–1385. [PubMed: 22241791]
21. Ganzfried BF, Riester M, Haibe-Kains B, et al. curatedOvarianData: clinically annotated data for the ovarian cancer transcriptome. Database. 2013;2013:bat013. https://academic.oup.com/database/article/doi/10.1093/database/bat013/330978 [PubMed: 23550061]
22. Naimi AI, Balzer LB. Stacked generalization: an introduction to super learning. Eur J Epidemiol. 2018;33(5):459–464. [PubMed: 29637384]
23. Ye Y. Interior Algorithms for Linear, Quadratic, and Linearly Constrained Non-Linear Programming. PhD Thesis. Stanford, CA: Department of ESS, Stanford University; 1987.
24. James G, Witten D, Hastie T, Tibshirani R. An Introduction to Statistical Learning. 112. New York: Springer; 2013.
25. van der Vaart AW, Dudoit S, van der Laan MJ. Oracle inequalities for multi-fold cross validation. Stat Decis. 2006;24(3):351–371.

26. Petersen ML, LeDell E, Schwab J, et al. Super learner analysis of electronic adherence data improves viral prediction and may provide strategies for selective HIV RNA monitoring. J Acquir Immune Defic Syn. 1999;69(1):109.
27. Balzer LB, Havlir DV, Kamya MR, et al. Machine learning to identify persons at high-risk of human immunodeficiency virus acquisition in rural Kenya and Uganda. Clin Infect Dis. 2020;71(9):2326–2333. [PubMed: 31697383]
28. Polley E, LeDell E, Kennedy C, Lendle S, Laan v dM. Package 'SuperLearner'. 2019.
29. Csardi G, Nepusz T. The igraph software package for complex network research. Int J Complex Syst. 2006;1695(5):1–9.
30. Kraus WE, Granger CB, Sketch MH, et al. A guide for a cardiovascular genomics biorepository: the CATHGEN experience. J Cardiovas Transl Res. 2015;8(8):449–457.
31. Foygel R, Drton M. Extended Bayesian information criteria for Gaussian graphical models. 2010 604–612.
32. Liu H, Roeder K, Wasserman L. Stability approach to regularization selection (stars) for high dimensional graphical models. 2010 1432–1440.
33. Epskamp S, Cramer AO, Waldorp LJ, Schmittmann VD, Borsboom D. qgraph: network visualizations of relationships in psychometric data. J Stat Softw. 2012;48(4):1–18.
34. Rolfs BT, Rajaratnam B. A note on the lack of symmetry in the graphical lasso. Comput Stat Data Anal. 2013;57(1):429–434.
35. Robert P, Escoufier Y. A unifying tool for linear multivariate statistical methods: the RV-coefficient. J R Stat Soc: Ser C (Appl Stat). 1976;25(3):257–265.
36. Indahl UG, Næs T, Liland KH. A similarity index for comparing coupled matrices. J Chemom. 2018;32(10):e3049.
37. Stewart G. On the continuity of the generalized inverse. SIAM J Appl Math. 1969;17(1):33–45.
38. Casella G, Berger RL. Statistical Inference. 2. CA: Duxbury Pacific Grove; 2002.
39. Tibshirani R. Regression shrinkage and selection via the lasso. J R Stat Soc: SerB (Methodol). 1996;58(1):267–288.
40. Platig J, Castaldi PJ, DeMeo D, Quackenbush J. Bipartite community structure of eQTLs. PLoS Comput Biol. 2016;12(9):e1005033. [PubMed: 27618581]
41. Clauset A, Newman ME, Moore C. Finding community structure in very large networks. Phys Rev E. 2004;70(6):066111.
42. Kleinberg JM. Authoritative sources in a hyperlinked environment. J ACM. 1999;46(5):604–632.
43. Salipante SJ, Rojas ME, Korkmaz B, et al. Contributions to neutropenia from PFAAP5 (N4BP2L2), a novel protein mediating transcriptional repressor cooperation between Gfi1 and neutrophil elastase. Mol Cell Biol. 2009;29(16):4394–4405. [PubMed: 19506020]
44. Wang Y, Zhu M, Guo F, Song Y, Fan X, Qin G. Identification of tumor microenvironment-related prognostic biomarkers in luminal breast cancer. Front Gene. 2020;11:555865.
45. Castro CN, Rosenzwajg M, Carapito R, et al. NCKAP1L defects lead to a novel syndrome combining immunodeficiency, lymphoproliferation, and hyperinflammation. J Experi Med. 2020;217(12):e20192275.
46. Velazquez-Torres G, Shoshan E, Ivan C, et al. A-to-I miR-378a-3p editing can prevent melanoma progression via regulation of PARVA expression. Nat Commun. 2018;9(1):1–7. [PubMed: 29317637]
47. Fukuda T, Chen K, Shi X, Wu C. PINCH-1 is an obligate partner of integrin-linked kinase (ILK) functioning in cell shape modulation, motility, and survival. J Biol Chem. 2003;278(51):51324–51333. [PubMed: 14551191]
48. Sepulveda J, Wu C. The parvins. Cell Mol Life Sci. 2006;63(1):25–35. [PubMed: 16314921]
49. Zhang L, Park CH, Wu J, et al. Proteolysis of Rad17 by Cdh1/APC regulates checkpoint termination and recovery from genotoxic stress. EMBO J. 2010;29(10):1726–1737. [PubMed: 20424596]
50. Upadhyay G, Chowdhury AH, Vaidyanathan B, Kim D, Saleque S. Antagonistic actions of Rcor proteins regulate LSD1 activity and cellular differentiation. Proc Natl Acad Sci. 2014;111(22):8071–8076. [PubMed: 24843136]

51. Liang Z, Mou Q, Pan Z, et al. Identification of candidate diagnostic and prognostic biomarkers for human prostate cancer: RPL22L1 and RPS21. Med Oncol. 2019;36(6):1–10.
52. Wang T, Wang ZY, Zeng LY, Gao YZ, Yan YX, Zhang Q. Down-regulation of ribosomal protein RPS21 inhibits invasive behavior of osteosarcoma cells through the inactivation of MAPK pathway. Cancer Manag Res. 2020;12:4949. [PubMed: 32612383]
53. Shetty V, Nickens Z, Testa J, Hafner J, Sinnathamby G, Philip R. Quantitative immunoproteomics analysis reveals novel MHC class I presented peptides in cisplatin-resistant ovarian cancer cells. J Proteom. 2012;75(11):3270–3290.
54. Lartigue T, Bottani S, Baron S, Colliot O, Durrleman S, Allassonnière S. Gaussian Graphical Model exploration and selection in high dimension low sample size setting. 2003.
55. Steinley D, Hoffman M, Brusco MJ, Sher KJ. A method for making inferences in network analysis: Comment on Forbes, Wright, Markon, and Krueger (2017). J Abnormal Psychol. 2017;126(7):1000–1010. 10.1037/abn0000308
56. Epskamp S, Isvoranu AM, Cheung MWL. Meta-analytic Gaussian network aggregation. Psychometrika. 2022:1–35.
57. Meinshausen N, Bühlmann P. Stability selection. J R Stat Soc: Ser B (Stat Methodol). 2010;72(4):417–473.
58. Li S, Hsu L, Peng J, Wang P. Bootstrap inference for network construction with an application to a breast cancer microarray study. Ann Appl Stat. 2013;7(1):391. [PubMed: 24563684]
59. Cai TT, Li H, Liu W, Xie J. Joint estimation of multiple high-dimensional precision matrices. Stat Sin. 2016;26(2):445. [PubMed: 28316451]
60. Chen H, Guo Y, He Y, et al. Simultaneous differential network analysis and classification for matrix-variate data with application to brain connectivity. Biostat. 2022;23(3):967–989.
61. Crijns AP, Fehrmann RS, Jong dS, et al. Survival-related profile, pathways, and transcription factors in ovarian cancer. PLoS Med. 2009;6(2):e1000024. [PubMed: 19192944]
62. Denkert C, Budczies J, Darb-Esfahani S, et al. A prognostic gene expression index in ovarian cancer—validation across different independent data sets. J Pathol: A J Pathol Soc Great Britain Ireland. 2009;218(2):273–280.
63. Yoshihara K, Tajima A, Yahata T, et al. Gene expression profile for predicting survival in advanced-stage serous ovarian cancer across two independent datasets. PloS one. 2010;5(3):e9615. [PubMed: 20300634]
64. Mok SC, Bonome T, Vathipadiekal V, et al. A gene signature predictive for outcome in advanced ovarian cancer identifies a survival factor: microfibril-associated glycoprotein 2. Cancer Cell. 2009;16(6):521–532. [PubMed: 19962670]
65. Konstantinopoulos PA, Spentzos D, Karlan BY, et al. Gene expression profile of BRCAness that correlates with responsiveness to chemotherapy and with outcome in patients with epithelial ovarian cancer. J Clin Oncol. 2010;28(22):3555. [PubMed: 20547991]
66. Bonome T, Levine DA, Shih J, et al. A gene signature predicting for survival in suboptimally debulked patients with ovarian cancer. Cancer Res. 2008;68(13):5478–5486. [PubMed: 18593951]
67. Ferriss JS, Kim Y, Duska L, et al. Multi-gene expression predictors of single drug responses to adjuvant chemotherapy in ovarian carcinoma: predicting platinum resistance. PloS one. 2012;7(2):e30550. [PubMed: 22348014]
68. Tothill RW, Tinker AV, George J, et al. Novel molecular subtypes of serous and endometrioid ovarian cancer linked to clinical outcome. Clin Cancer Res. 2008;14(16):5198–5208. [PubMed: 18698038]
69. Dressman HK, Berchuck A, Chan G, et al. An integrated genomic-based approach to individualized treatment of patients with advanced-stage ovarian cancer. J Clin Oncol. 2007;25(5):517–525. [PubMed: 17290060]
70. Berchuck A, Iversen ES, Luo J, et al. Microarray analysis of early stage serous ovarian cancers shows profiles predictive of favorable outcome. Clin Cancer Res. 2009;15(7):2448–2455. [PubMed: 19318476]
71. Cancer Genome Atlas Research Network. Integrated genomic analyses of ovarian carcinoma. Nature. 2011;474(7353):609. [PubMed: 21720365]

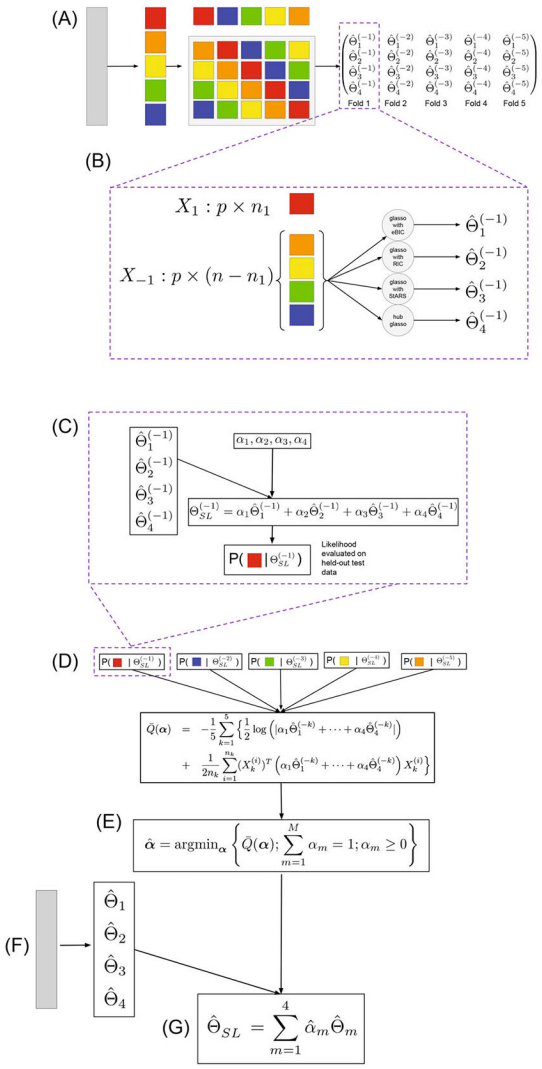

FIGURE 1.

(A) Data are partitioned into five folds. Each fold is left out from the model fitting process in turn. (B) Every candidate model is fit on the training data in each fold. This generates an $(M = 4) \times (K = 5)$ array of estimated matrices $\widehat{\boldsymbol{\Theta}}_m^{(-k)} : m = 1, \ldots, 4; k = 1, \ldots, 5$. (C) For each held-out dataset k and coefficient set $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_4)$, $\boldsymbol{\Theta}_{SL}^{(-k)}$ is calculated from the estimates in (B). The likelihood of the estimator given the held-out data is then calculated as a function of the unknown $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_4)$. (D) The process is repeated across all $K = 5$ folds and averaged to yield our loss function. (E) The loss function is minimized to yield the optimal coefficients $\widehat{\boldsymbol{\alpha}}$, subject to the constraints of the convex combination. (F) The $M = 4$ methods are used to fit $\widehat{\boldsymbol{\Theta}}_1, \ldots, \widehat{\boldsymbol{\Theta}}_4$ on the whole dataset. (G) The final SpiderLearner estimator $\widehat{\boldsymbol{\Theta}}_{SL}$ is calculated as the convex combination of the coefficients selected in (E) with the models fit in (F).

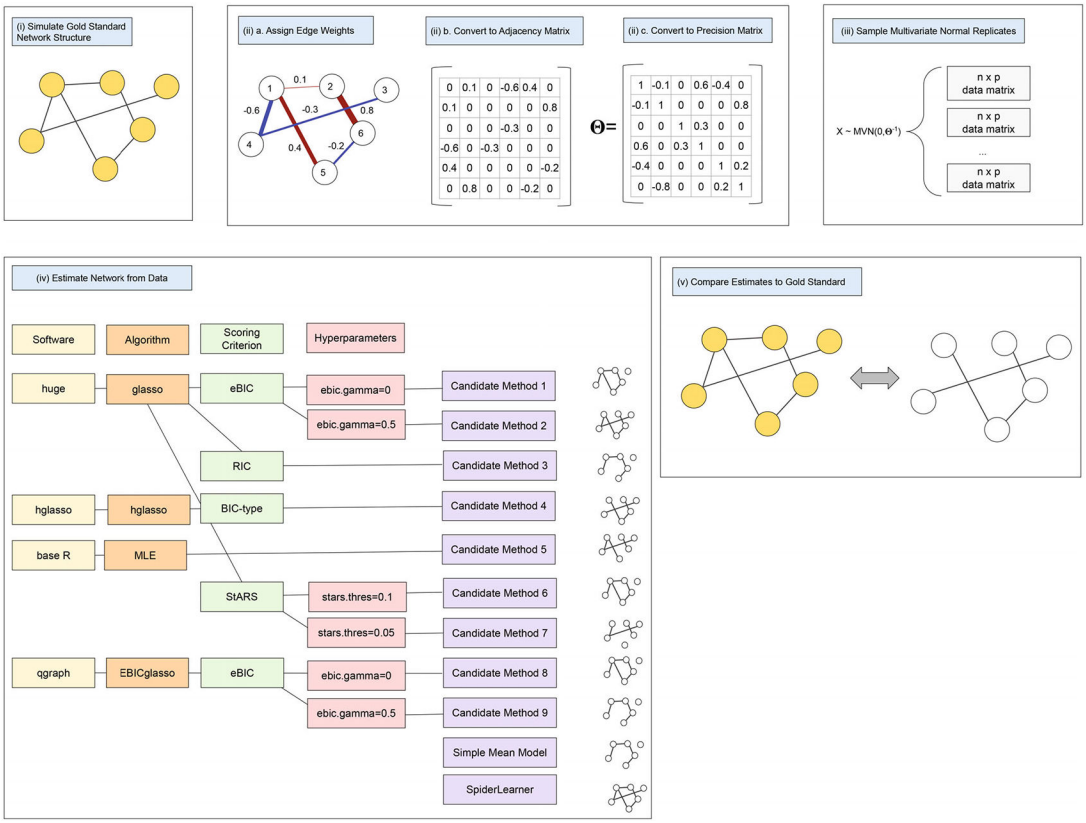

FIGURE 2.

Simulation study workflow. In (i) we design gold-standard networks. In (ii), we assign edge weights to the gold standards by sampling from the distribution of partial correlations observed in the CATHGEN dataset and convert the corresponding adjacency matrices to precision matrices. In (iii), we sample multivariate normal data based on the precision matrices from (ii). In (iv), we estimate the networks from the sampled data. In (v), we compare the estimated network to the gold standard.

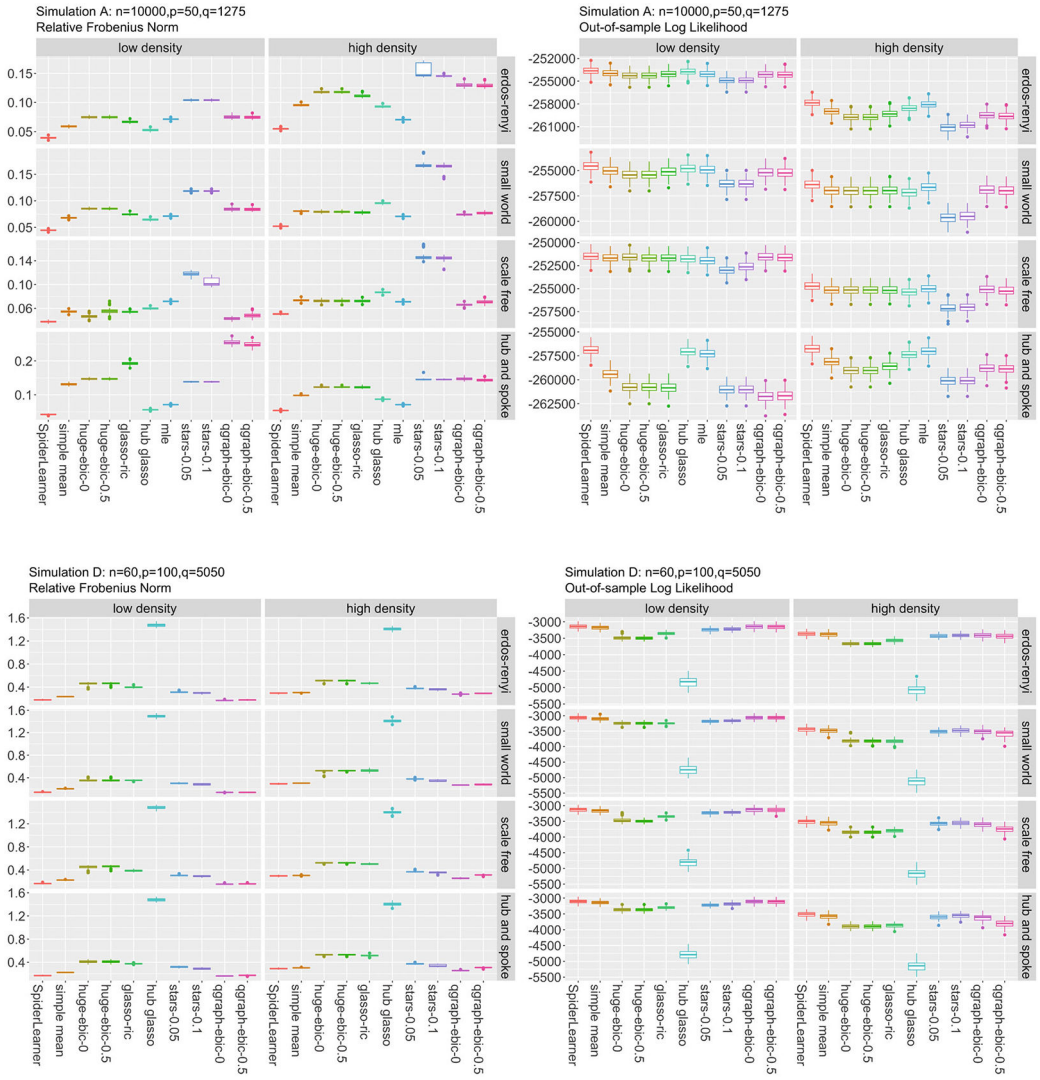

FIGURE 3.

Simulation A and D results comparing SpiderLearner to the simple mean model and to the eight candidate algorithms in the library. The variable p represents the number of nodes in the network, while q represents the number of parameters to be fit. Relative Frobenius norm results demonstrate the ability of the algorithm to accurately estimate the precision matrix entries; lower is better. Out-of-sample log likelihood shows that the estimated precision matrix is not overfit by the SpiderLearner; higher is better. Results show that the SpiderLearner ensemble model is able to outperform or match the performance of every other candidate included in the model and exceed the performance of a simple mean of the candidates.

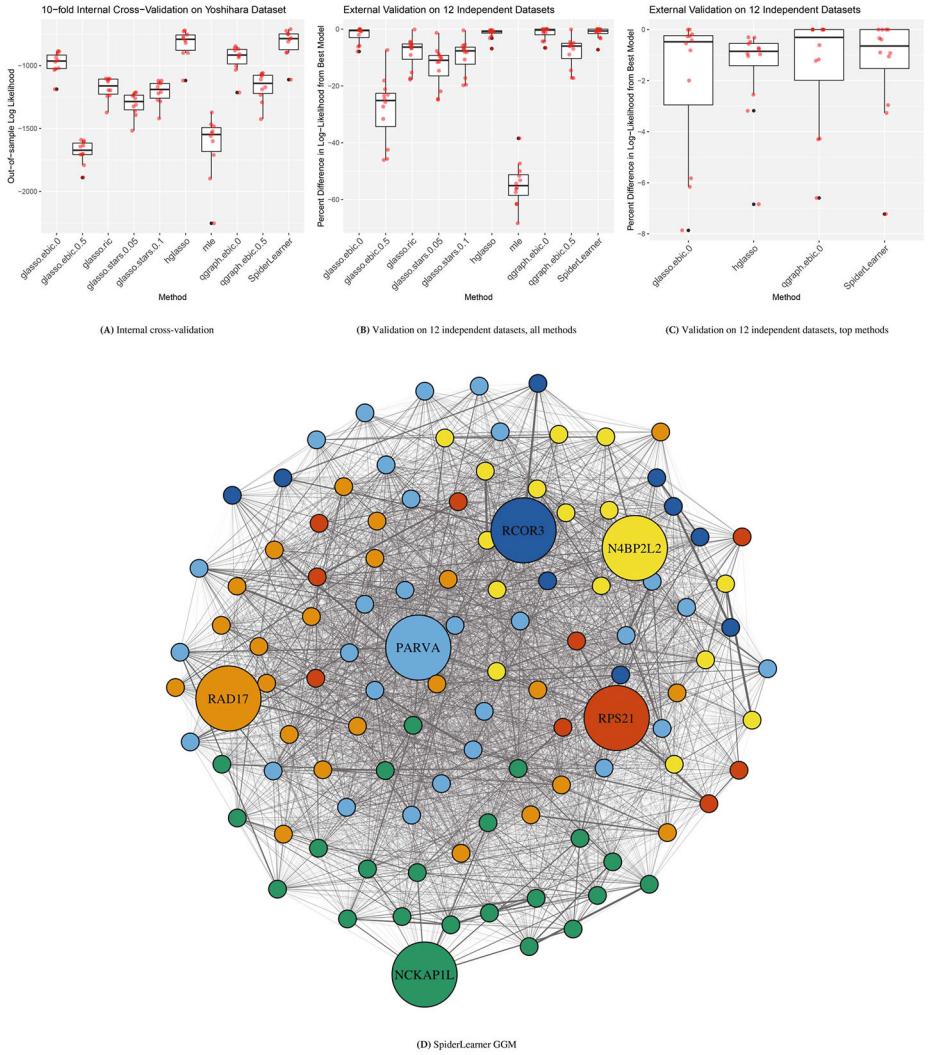

FIGURE 4.

(A) Performance of SpiderLearner and candidate methods in internal 10-fold cross-validation, measured in terms of out-of-sample log likelihood. (B,C) Performance of SpiderLearner and candidate methods on 12 independent validation datasets from the `curatedOvarianData` R package. Performance of each candidate algorithm is measured in terms of percent difference in log likelihood relative to the best performing model for that dataset; for example, a y-axis value of −10 means a model had a 10% lower log likelihood than the best model for that particular dataset. (D) SpiderLearner GGM on 111 genes associated with high-risk ovarian cancer. Node color corresponds to community membership. Large, labeled nodes indicate the six “local hubs”: genes with the highest hub score within each community.

TABLE 1

Details of the simulation study designed to evaluate the practical performance of SpiderLearner across a range of network dimensions and topologies. (A) Simulation study dimensionality. n represents the sample size; p, the number of predictors in the network; q, the number of parameters that need to be estimated in the model; $.9 * n / q$: the sample size-to-parameter ratio in each training set in the 10-fold cross-validation. (B) Gold-standard networks were constructed using a variety of functions from the `igraph` package. Graph density is a function of the parameters used in each function as well as the number of predictors in the graph, and cannot be exactly specified. Parameters used in this study were chosen to achieve approximately 6% dense graphs in the low-density cases and 20% dense graphs in the high-density cases.

(A)

Simulation	n	p	q	$.9 * n / q$
A	10,000	50	1275	7.06
B	1600	50	1275	1.13
C	100	50	1275	0.07
D	60	100	5050	0.01

(B)

Topology	Density	`igraph` Function	Simulated density (Simulations A,B,C)	Simulated density (Simulation D)
Random	Low	`sample_gnp`	0.053	0.061
Random	High	`sample_gnp`	0.219	0.194
Small world	Low	`sample_smallworld`	0.082	0.061
Small world	High	`sample_smallworld`	0.204	0.202
Scale-free	Low	`sample_pa`	0.079	0.059
Scale-free	High	`sample_pa`	0.192	0.191
Hub-and-spoke	Low	`sample_pa`	0.079	0.059
Hub-and-spoke	High	`sample_pa`	0.192	0.191

TABLE 2

Average weight for each method as selected by SpiderLearner in $N = 100$ simulations.

(A) Simulation A

Topology	glasso–ebic–0	glasso–ebic–0.5	glasso–ric	hglasso	mle	glasso–stars–0.05	glasso–stars–0.1	qgraph–ebic–0	qgraph–ebic–0.5
Random low	0	0	0	0.57	0.28	0	0	0.15	0
Random high	0	0	0	0.33	0.59	0	0	0.08	0
Small world low	0	0	0	0.46	0.39	0	0	0.15	0
Small world high	0	0	0	0.19	0.55	0	0	0.25	0
Scale-free low	0.34	0	0	0	0.25	0	0	0.4	0
Scale-free high	0.07	0.07	0	0.1	0.51	0	0	0.24	0
Hub-and-spoke low	0	0	0	0.63	0.34	0	0	0.03	0
Hub-and-spoke high	0	0	0	0.37	0.57	0	0	0.06	0

(B) Simulation D

Topology	glasso–ebic–0	glasso–ebic–0.5	glasso–ric	hglasso	glasso–stars–0.05	glasso–stars–0.1	qgraph–ebic–0	qgraph–ebic–0.5
Random low	0	0	0	0.03	0.08	0.24	0.63	0.01
Random high	0	0	0	0.07	0.12	0.62	0.2	0
Small world low	0.02	0.02	0.02	0.02	0.04	0.06	0.4	0.44
Small world high	0	0	0	0.08	0.23	0.6	0.08	0
Scale-free low	0	0	0	0.03	0.08	0.22	0.67	0
Scale-free high	0	0	0	0.09	0.15	0.75	0.01	0
Hub-and-spoke low	0	0	0	0.03	0.05	0.22	0.62	0.08
Hub-and-spoke high	0	0	0	0.1	0.18	0.72	0.01	0

TABLE 3

Basic characteristics and references for the 13 ovarian cancer datasets used in the SpiderLearner application. The Yoshihara dataset is GSE32062.GPL6480.

Dataset	Platform ID	*N*	Age: Mean(SD) *missing*	Tumor stage (% < 4) *missing*	Summary stage (% Late) *missing*	Summary grade high (%) *missing*
GSE32062.GPL6480[20]	hgug4112a	260	—	78	100	50
GSE13876[61]	OperonHumanV3	157	57.95(12.39)	—	100	54 *13*
GSE14764[62]	hgu133a	80	—	98	89	68
GSE17260[63]	hgug4112a	110	—	85	100	39
GSE18520[64]	hgu133plus2	63	—	100 *10*	84 *10*	84 *10*
GSE19829.GPL570[65]	hgu133plus2	28	—	—	—	—
GSE26712[66]	hgu133a	195	61.54(11.86) *13*	80 *13*	95 *10*	95 *10*
GSE30161[67]	hgu133plus2	58	62.57(10.61)	91	100	57 *4*
GSE32063[20]	hgug4112a	40	—	78	100	42
GSE9891[68]	hgu133plus2	285	59.62(10.59) *3*	92 *3*	84 *3*	57 *6*
PMID17290060[69]	hgu133a	117	—	85 *1*	98 *1*	49 *3*
PMID19318476[70]	hgu133a	42	61.46(10.61) *1*	76 *1*	93 *1*	57 *1*
TCGA[71]	hthgu133a	578	59.7(11.56) *10*	85 *15*	90 *15*	83 *23*