Supplementary Information
Bipartite Tight Spectral Clustering (BiTSC) Algorithm for Identifying Conserved Gene Co-clusters in Two Species

Yidan Eden Sun, Heather J. Zhou and Jingyi Jessica Li

S1 Six possible variants of BiTSC

In the development of BiTSC, we considered six possible variants of its algorithm, and we compared them with BiTSC to justify our choice of BiTSC as the proposed algorithm. The performance comparison is in Section 3.1.

1. Bipartite spectral clustering with kernel enhancement (Spectral-kernel). This algorithm applies kernel enhancement followed by bipartite spectral clustering to the original bipartite network with node covariates. Comparing it with BiTSC would help us evaluate the effectiveness of the subsampling-and-aggregation approach taken by BiTSC.

2. Bipartite spectral clustering (Spectral). This algorithm removes the kernel enhancement step from Spectral-kernel. Comparing it with Spectral-kernel would help us evaluate whether kernel enhancement is useful.

3. BiTSC-1. This algorithm differs from BiTSC in terms of the timing of subsampling. Unlike BiTSC, BiTSC-1 performs subsampling after applying kernel enhancement and bipartite spectral clustering to the original bipartite network. The use of "1" in the algorithm name means that bipartite spectral clustering is only applied for once. In detail, BiTSC-1 first performs kernel enhancement on the original bipartite network with node covariates to obtain an enhanced bi-adjacency matrix $\mathbf{B}$. BiTSC-1 next applies bipartite spectral clustering, same as in BiTSC but without the last K-means clustering step, to $\mathbf{B}$ to obtain $\mathbf{V}$, an $(m+n) \times K_0$ matrix. Then BiTSC-1 applies the subsampling-and-aggregation approach to the $(m+n)$ rows of $\mathbf{V}$. Specifically, in the h-th run, $h = 1, \ldots, H$, BiTSC-1 has the following three steps: (1) it randomly samples without replacement $\tilde{m}$ rows from the first m rows of $\mathbf{V}$ and $\tilde{n}$ rows from the last n rows of $\mathbf{V}$; (2) it divides the subsampled $(\tilde{m}+\tilde{n})$ rows into K_0 initial co-clusters using the K-means algorithm with Euclidean distance; (3) it assigns the unsampled $(m-\tilde{m}+n-\tilde{n})$ nodes to the K_0 initial co-clusters based on node covariates, same as in BiTSC. The remaining steps of BiTSC-1, including the aggregation of these H sets of K_0 node co-clusters into a consensus matrix and the identification of tight node co-clusters, are the same as those of BiTSC. Comparing BiTSC-1 with BiTSC will help us evaluate the effect of the timing of subsampling, i.e., whether performing subsampling before kernel enhancement and bipartite spectral clustering aids the identification of tight node co-clusters.

4. BiTSC-1-nokernel. This algorithm removes the kernel enhancement step from BiTSC-1. Comparing it to BiTSC-1 would help us evaluate whether kernel enhancement is useful given the subsampling-and-aggregation approach.

5. BiTSC-1-NC. This algorithm modifies BiTSC-1 by changing how the unsampled nodes are assigned into the K_0 initial node co-clusters in each of the H runs. Specifically, BiTSC-1-NC only differs from BiTSC-1 in step (3) of the h-th run, $h = 1, \ldots, H$, where BiTSC-1-NC assigns the unsampled $(m-\tilde{m}+n-\tilde{n})$ nodes to the K_0 initial node co-clusters based on their corresponding rows in $\mathbf{V}$ instead of their covariates. In detail, BiTSC-1-NC calculates a mean vector for each initial node co-cluster as the average of its corresponding rows in $\mathbf{V}$; then BiTSC-1-NC assigns each unsampled node to the

initial co-cluster whose mean vector has the smallest Euclidean distance to the node's corresponding row in $\mathbf{V}$. Note that "NC" in the algorithm name means that "no covariates" is used in the assignment step. Comparing BiTSC-1-NC with BiTSC-1 would help us evaluate the effect of using node covariates in the assignment step.

6. BiTSC-1-NC-nokernel. This algorithm removes the kernel enhancement step from BiTSC-1-NC. When compared to BiTSC-1-NC, it can help us evaluate whether kernel enhancement is useful in the absence of node covariates in the assignment step.

In summary, the above six possible variants of BiTSC can help us evaluate the design of BiTSC. Three variants pose contrasts to BiTSC in three aspects: Spectral-kernel does not use the subsampling-and-aggregation approach; BiTSC-1 uses a different timing for subsampling; BiTSC-1-NC does not use node covariates in the assignment of unsampled nodes to initial node co-clusters. The other three variants, Spectral, BiTSC-1-nokernel, and BiTSC-1-NC-nokernel, remove the kernel enhancement from Spectral-kernel, BiTSC-1, and BiTSC-1-NC respectively to evaluate the effectiveness of kernel enhancement.

S2 Data generation in simulation studies

Here we describe how we generate bipartite networks with node covariates in simulation studies (Section 3.1). The generation process comprises three steps.

1. Network structure setup. To evaluate the capacity of BiTSC in detecting tight node co-clusters and leaving out outlier nodes, we generate a bipartite network including two types of nodes: clustered nodes in co-clusters and noise nodes that do not belong to any co-clusters. We consider K non-overlapping node co-clusters, each of which contains n_1 nodes from side 1 and n_2 nodes from side 2. Hence, there are Kn_1 and Kn_2 clustered nodes on side 1 and 2, respectively. We define θ as the ratio (# of noise nodes)/(# of clustered nodes). Then there are $\lfloor \theta K n_1 \rfloor$ and $\lfloor \theta K n_2 \rfloor$ noise nodes on side 1 and 2, respectively. Therefore, the bipartite network contains a total of $m = Kn_1 + \lfloor \theta K n_1 \rfloor$ nodes on side 1 and $n = Kn_2 + \lfloor \theta K n_2 \rfloor$ nodes on side 2.

2. Edge generation. We generate independent binary edges between nodes by following a stochastic block model [Nowicki and Snijders, 2001]: if two nodes belong to the same co-cluster, an edge between them is drawn from Bernoulli(p), where $p \in (0,1)$ is the within-cluster edge probability; otherwise, an edge is drawn from Bernoulli(q), where $q \in (0,p)$ is the not-within-cluster edge probability smaller than p. A larger q/p ratio would lead to a more obscure node co-cluster structure in the resulting bipartite network.

3. Covariate generation. We generate covariate vectors for clustered nodes and noise nodes separately. First, we assume that nodes in each co-cluster on each side have covariate vectors following a multivariate Gaussian distribution. In detail, for the n_1 nodes in the k-th co-cluster on side 1, we independently draw n_1 vectors of length p_1 from a p_1-dimensional Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{1k}, (\omega_1 k)^2 \mathbf{I}_{p_1})$, $k = 1, \ldots, K$. Note that the larger the co-cluster index k, the more spread out the n_1 covariate vectors are around the mean $\boldsymbol{\mu}_{1k}$. The K co-cluster mean vectors on side 1, $\boldsymbol{\mu}_{11}, \ldots, \boldsymbol{\mu}_{1K}$, are independently drawn from a p_1-dimensional Gaussian prior $\mathcal{N}(\mathbf{0}, \sigma_1^2 \mathbf{I}_{p_1})$. Nodes in co-clusters on side 2 are simulated similarly from K p_2-dimensional Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}_{2k}, (\omega_2 k)^2 \mathbf{I}_{p_2})$, with the co-cluster mean vectors on side 2, $\boldsymbol{\mu}_{21}, \ldots, \boldsymbol{\mu}_{2K}$, independently drawn from a p_2-dimensional Gaussian prior $\mathcal{N}(\mathbf{0}, \sigma_2^2 \mathbf{I}_{p_2})$. Second, for noise nodes that do not belong to any co-clusters on each side, we randomly generate their covariate vectors from uniform distributions defined by the ranges of the already-generated covariate vectors of clustered nodes. In detail, on side 1, we take the Kn_1 clustered nodes and define a range based on their values in each of the p_1 dimensions. Then we independently draw $\lfloor \theta K n_1 \rfloor$ scalars uniformly from the range of each dimension to construct $\lfloor \theta K n_1 \rfloor$ covariate vectors of length p_1. Similarly, we simulate covariate vectors of noise nodes on side 2.

In summary, the data generation process requires the following input parameters: K, the number of true co-clusters; n_r, the number of nodes in each co-cluster on side r; θ, the ratio of the number of noise nodes over the number of clustered nodes; p, the within-cluster edge probability; q, the not-within-cluster edge probability; p_r, the dimension of covariate vectors on side r; ω_r, the parameter for within-cluster variance on side r; σ^2, the variance parameter for generating co-cluster mean vectors on side r, $r = 1, 2$.

We simulated an example bipartite network using the approach described above, with $H = 50$, $n_1 = 50$, $n_2 = 70$, $p_1 = p_2 = 2$, $\sigma_1 = \sigma_2 = 10$, $\omega_1 = \omega_2 = 0.1$, $\theta = 0.5$, $q/p = 5$, and $q = 0.03$. Figure S5a-b illustrate the bi-adjacency matrix and the true co-membership matrix of this simulated network. Figure S5c-d show the node covariates.

S2.1 Average degree

Within each of the K true co-clusters, there are n_1 and n_2 nodes on side 1 and 2, respectively. The noise node ratio is denoted by θ; that is, there are $\lfloor n_1 K\theta \rfloor$ and $\lfloor n_2 K\theta \rfloor$ noise nodes on side 1 and 2. Hence, in total there are $m = n_1 K + \lfloor n_1 K\theta \rfloor$ nodes on side 1 and $n = n_2 K + \lfloor n_2 K\theta \rfloor$ nodes on side 2. Recall that $\mathbf{A} = (a_{ij})_{m\times n}$ is the bi-adjacency matrix. Let $\lambda_{1i} = \sum_{j=1}^{n} a_{ij}$ and $\lambda_{2j} = \sum_{i=1}^{m} a_{ij}$ be the degree of node i on side 1 and node j on side 2, respectively. Then, the average degree of the network is

$$\lambda = \frac{\sum_{i=1}^{m} \lambda_{1i} + \sum_{j=1}^{n} \lambda_{2j}}{m+n} = \frac{2\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}}{m+n}\,. \tag{S1}$$

S3 Evaluation metric of clustering result

In the simulation study (Section 3.1), we use the weighted Rand index [Thalamuthu et al., 2006], which the extension of the adjusted Rand index [Rand, 1971, Hubert and Arabie, 1985], to evaluate the clustering result by comparing the identified co-clusters to the true co-clusters. The weighted Rand index was developed for the case where noise nodes exist and should stay unclustered. We use $\mathcal{V} = \{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_K, \mathcal{V}_{K+1}\}$ to denote the true node cluster membership, where $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_K$ indicate the K true co-clusters and $\mathcal{V}_{K+1}$ indicates the set of noise nodes. We use $\tilde{\mathcal{V}} = \{\tilde{\mathcal{V}}_1, \tilde{\mathcal{V}}_2, \ldots, \tilde{\mathcal{V}}_C, \tilde{\mathcal{V}}_{C+1}\}$ to denote the clustering result, where $\tilde{\mathcal{V}}_1, \tilde{\mathcal{V}}_2, \ldots, \tilde{\mathcal{V}}_C$ indicate the C identified co-clusters and $\tilde{\mathcal{V}}_{C+1}$ represents the set of unclustered nodes.

Thalamuthu et al. proposed two types of adjusted Rand index. The first one, $\text{Rand}_1(\mathcal{V}, \tilde{\mathcal{V}})$, considers $\mathcal{V}_{K+1}$ and $\tilde{\mathcal{V}}_{C+1}$ as two regular clusters:

$$\text{Rand}_1(\mathcal{V}, \tilde{\mathcal{V}}) = \frac{\sum_{i=1}^{K+1}\sum_{j=1}^{C+1}\binom{N_{ij}}{2} - \sum_{i=1}^{K+1}\binom{N_{i\cdot}}{2}\sum_{j=1}^{C+1}\binom{N_{\cdot j}}{2}/\binom{N}{2}}{0.5\left[\sum_{i=1}^{K+1}\binom{N_{i\cdot}}{2} + \sum_{j=1}^{C+1}\binom{N_{\cdot j}}{2}\right] - \sum_{i=1}^{K+1}\binom{N_{i\cdot}}{2}\sum_{j=1}^{C+1}\binom{N_{\cdot j}}{2}/\binom{N}{2}}\,,$$

where $N_{ij} = |\mathcal{V}_i \cap \tilde{\mathcal{V}}_j|$, $N_{i\cdot} = \sum_{j=1}^{C+1} N_{ij}$, $N_{\cdot j} = \sum_{i=1}^{K+1} N_{ij}$, and $N = \sum_{i=1}^{K+1}\sum_{j=1}^{C+1} N_{ij}$. Note that N denotes the total number of nodes.

The second index $\text{Rand}_2(\mathcal{V}, \tilde{\mathcal{V}})$ ignores $\mathcal{V}_{K+1}$ and $\tilde{\mathcal{V}}_{C+1}$:

$$\text{Rand}_2(\mathcal{V}, \tilde{\mathcal{V}}) = \frac{\sum_{i=1}^{K}\sum_{j=1}^{C}\binom{N_{ij}}{2} - \sum_{i=1}^{K}\binom{\tilde{N}_{i\cdot}}{2}\sum_{j=1}^{C}\binom{\tilde{N}_{\cdot j}}{2}/\binom{\tilde{N}}{2}}{0.5\left[\sum_{i=1}^{K}\binom{\tilde{N}_{i\cdot}}{2} + \sum_{j=1}^{C}\binom{\tilde{N}_{\cdot j}}{2}\right] - \sum_{i=1}^{K}\binom{\tilde{N}_{i\cdot}}{2}\sum_{j=1}^{C}\binom{\tilde{N}_{\cdot j}}{2}/\binom{\tilde{N}}{2}}\,,$$

where $\tilde{N}_{i\cdot} = \sum_{j=1}^{C} N_{ij}$, $\tilde{N}_{\cdot j} = \sum_{i=1}^{K} N_{ij}$ and $\tilde{N} = \sum_{i=1}^{K}\sum_{j=1}^{C} N_{ij}$.

$\text{Rand}_1(\mathcal{V}, \tilde{\mathcal{V}})$ is biased against clustering methods that do not allow unclustered nodes, especially when the number of noise nodes is large. On the other hand, $\text{Rand}_2(\mathcal{V}, \tilde{\mathcal{V}})$ is biased against clustering methods that allow unclustered nodes. To balance the two, the weight Rand index was proposed as a weighted sum of the two indices [Thalamuthu et al., 2006]:

$$\text{Rand}(\mathcal{V}, \tilde{\mathcal{V}}) = \lambda \cdot \text{Rand}_1(\mathcal{V}, \tilde{\mathcal{V}}) + (1-\lambda)\cdot \text{Rand}_2(\mathcal{V}, \tilde{\mathcal{V}})\,, \tag{S2}$$

where $\lambda = |\mathcal{V}_{K+1} \cup \tilde{\mathcal{V}}_{C+1}|/N$.

We implemented the weighted Rand index by using the `adjusted_rand_score` function in the Python package `sklearn` [Pedregosa et al., 2011].

S4 Robustness analysis of input parameters

Figure S6 shows the robustness analysis of BiTSC and its three variants: Spectral-kernel, BiTSC-1, and BiTSC-1-NC, against input parameters K_0 (the number of co-clusters in each subsampling run), ρ (the subsampling proportion), and τ (the kernel enhancement parameter), in the simulation setting (Section S2). We observe that BiTSC is robust to the choice of K_0 when K_0 is larger than K, the number of true co-clusters. BiTSC outperforms the three variants when ρ is larger than 0.7, and BiTSC is robust to the choice of τ. Hence, we set $\rho = 0.8$ and $\tau = (1, 1)$ as the default input parameters in BiTSC.

Interestingly, BiTSC-1-NC has weighted Rand indices that are extremely close to those of Spectral-kernel and invariant to ρ in Figure S6b. Spectral-kernel does not use subsampling, so it is expected to have a constant weighted Rand index invariant to ρ. However, BiTSC-1-NC uses subsampling, so its weighted Rand index should depend on ρ. We investigated this phenomenon and found that BiTSC-1-NC has weighted Rand indices not exactly the same but very close in values:

ρ	Weighted Rand index
0.3	0.7708656319227645
0.4	0.7706005330861144
0.5	0.7710055546541685
0.6	0.7711873136856517
0.7	0.7712402375720288
0.8	0.7713976529947345
0.9	0.7715965812884181

This result suggests that BiTSC-1-NC, where we only do subsampling on $\mathbf{V}$ and do not use node covariates but only rows in $\mathbf{V}$ to assign unsampled nodes in each subsampling run (Section S1), is very robust to ρ and highly similar to Spectral-kernel. This result is consistent with what we observed in Figure 2.

We further verified the robustness of BiTSC against τ in simulated networks with various sparsity levels (Figure S7). The performance of BiTSC and the three variants consistently peak near $\tau = 0$ and decrease as τ increases. This is within our expectation: in these simulated bipartite networks, the covariate information is consistent with the edge information, i.e., the true co-clusters feature both dense edges between species and similar covariates within species (Section S2); therefore, a smaller τ means better performance (Section 2.2). We also observe that the smaller p is, i.e., the lower the edge density, the more quickly the performance of BiTSC and the three variants drops as τ increases, demonstrating the importance of incorporating covariate information in the kernel enhancement step especially when edge density is low.

Interestingly, as is the most obvious in Figures S7c–h, as τ keeps increasing, eventually the performance of the three variant algorithms surpass the performance of BiTSC. This is because in the three variant algorithms, kernel enhancement is performed prior to subsampling (or in the case of spectral-kernel, no subsampling is done) (Section S1). Hence, utilizing covariate information from all of the nodes in the bipartite network can make up for a large τ to a certain extent.

S5 Data processing in the real data application

In the real data application of BiTSC (Section 3.2), the *D. melanogaster* (fly) and *C. elegans* (worm) data set consists of two parts: gene expression data and gene orthology information. For gene expression data, we started with 15,095 fly protein-coding genes' expression levels across 30 developmental stages and 44,969 worm protein-coding genes' expression levels across 35 developmental stages. Note that the gene expression levels are in the FPKM (Fragments Per Kilobase of transcript per Million mapped reads) unit and were processed from RNA-seq data collected by the modENCODE Consortium [Gerstein et al., 2014, Li et al., 2014]. For gene orthology information, we started with 11,403 ortholog pairs between the above mentioned fly and worm genes (obtained and processed from the TreeFam database [Li et al., 2006, Li et al., 2014]). Then we removed all the fly and worm genes that have zero expression levels across all the developmental stages and have no orthologous genes, leaving us with 5,414 fly genes, 5,731 worm genes, and 10,975 ortholog pairs. After that, we performed the logarithmic transformation on the gene expression levels and standardized the transformed levels by subtracting the mean and dividing the standard deviation for every gene (Section

S5.1). Finally, we built a bipartite network of fly and worm genes by connecting orthologous genes, and we collected every gene's standardized expression levels into its node covariate vector. In this resulting bipartite network with node covariates (Table 2), $m = 5,414$, $n = 5,731$, $p_1 = 30$, and $p_2 = 35$. The average degree of this bipartite network is 1.97 (Section S2.1).

S5.1 Processing of gene expression levels

Gene expression values in the FPKM unit are typically highly skewed with the presence of extremely large values. Logarithmic transformation has been widely used to transform FPKM values to reduce the effects of outliers and to make the transformed values more normally distributed [Zwiener et al., 2014, Danielsson et al., 2015, Pertea et al., 2016]. Following the notations in Section 2.1, for gene i on side r, we constructed its j-th covariate as

$$x_{rij} = \frac{l_{rij} - \bar{l}_{ri\cdot}}{\sqrt{\sum_{j=1}^{p_r}(l_{rij} - \bar{l}_{ri\cdot})^2/(p_r - 1)}}\,, \tag{S3}$$

where $l_{rij} = \log_2(\text{FPKM}_{rij} + 1)$ and $\bar{l}_{ri\cdot} = \sum_{j=1}^{p_r} l_{rij}/p_r$. In other words, the gene covariates are standardized log-transformed FPKM values. We used these covariates in our fly-worm data analysis in Section 3.2.

S6 Choice of K_0 in the real data application

In our simulation study in Section S4, we observed that BiTSC is robust to K_0 values above K, the number of true co-clusters. However, in real data applications, we do not know K. Here we explain how we used the consensus distribution [Monti et al., 2003, Cowell, 2011] to choose $K_0 = 30$ in the real data application (Section 3.2).

The idea of using the consensus distribution to guide the choice of K_0 is the following: since the entries of the consensus matrix $\bar{\mathbf{M}}$ lie between 0 and 1, with each entry indicating how frequently two nodes are grouped together across multiple clustering results, a good K_0 should lead to many entries in the consensus matrix equal to 0 or 1 and few having fractional values in between. In other words, if we plot a histogram of the consensus matrix obtained for a good K_0, it would present a bimodal shape with two bins concentrated at 0 and 1. To better quantify this concentration of the consensus distribution, [Monti et al., 2003] used both the empirical cumulative distribution function (CDF) (Figure S8 (a)) of a consensus matrix entries and the area under CDF as the guidance to select K_0. In particular, the CDF corresponding to a good K_0 should present the shape with 2 increases (one increase at 0 and one increase at 1) and a horizontal line in between, and this bimodal shape would keep stable as K_0 past K (if known). In general, by inspecting the area under CDF curve (Figure S8b) to see where does the progression reach stable, users could decide an appropriate number of clusters as K_0.

Following the same sense, BiTSC provides an efficient way to select K_0 based on the idea of binary search (details in Algorithm S1). The rationale of Algorithm S1 is among a range of K_0's, we want to find the value of K_0 corresponding to which the CDF of the consensus matrix reaches stable at the earliest. In the real data analysis (Section 3.2), when we initialize K_0_min $= 10$ and K_0_max $= 50$ ($H = 48$, $\rho = 0.8$ and $\tau = (1, 1)$), Algorithm S1 selects 30 as the final input for K_0. To further verify the rationality in Algorithm S1, we applied BiTSC ($H = 48$, $\rho = 0.8$ and $\tau = (1, 1)$) to the fly-worm data, with K_0 ranging from 2 to 50. For each K_0 and its resulting consensus matrix, we plotted the empirical cumulative distribution function (CDF) of the matrix entries in Figure S8a. We also plotted the area under CDF curve as a function of K_0 in Figure S8b. From the result, we can observe that distribution of entries in the consensus matrix corresponding to $K_0 = 30$ concentrates at 0 and 1, and the area under the CDF curve plateaus after this point.

S7 Statistical analysis of identified gene co-clusters

Here we introduce three hypothesis tests, which are designed to analyze the gene co-clusters identified by BiTSC in the real data application (Section 3.2). The three tests are from [Li et al., 2014] and described in detail below. Note that we only include biological process (BP) gene ontology (GO) terms in the GO term enrichment test and the GO term overlap test for ease of interpretation. The GO terms are from the R package GO.db [Carlson, 2019].

S7.1 GO term enrichment test

Given a gene co-cluster, the GO term enrichment test is to check for each species, whether a GO term is enriched in this co-cluster relative to all the genes in that species. The top enriched GO terms would indicate the biological functions of this co-cluster in each species.

Suppose that there are u genes of species 1 in this co-cluster, v of which are annotated with a given GO term. Also suppose that species 1 has a total of U genes, V of which are annotated with the same GO term. The null hypothesis is that this GO term has the same enrichment level in the u genes as in the U genes, i.e., the u genes are randomly sampled from the U genes. The alternative hypothesis is that this GO term is more enriched in the u genes relative to the U genes. Under the null hypothesis, X, the number of genes that are annotated with this GO term among any u genes, follows a hypergeometric distribution with the following probability mass function

$$P(X = x) = \frac{\binom{V}{x}\binom{U-V}{u-x}}{\binom{U}{u}},\ x = 0, 1, \cdots, \min(V, u)\,.$$

Hence, this test has a P-value defined below and denoted by P_1.

$$\mathrm{P}_1 = P(X \geq v) = \sum_{x=v}^{\min(V,u)} \frac{\binom{V}{x}\binom{U-V}{u-x}}{\binom{U}{u}}\,.$$

We implemented this test, which is equivalent to the one-tail Fisher's exact test, using the R package topGO [Alexa and Rahnenfuhrer, 2019].

S7.2 GO term overlap test

Given a gene co-cluster, the GO term overlap test is to check whether the genes from the two species share similar biological functions, i.e., whether the two sets of genes have a significant overlap in their annotated GO terms.

Specifically, for this co-cluster, we denote by A and B the sets of GO terms associated with its genes in species 1 and 2, respectively. We define a population of N GO terms as the set of terms associated with any genes in species 1 or 2. The null hypothesis is that A and B are two independent samples from the population, i.e., the common GO terms shared by two species in this co-cluster is purely due to a random overlap. The alternative hypothesis is that A and B are positively dependent samples. Under the null hypothesis that two samples with sizes $|A|$ and $|B|$ are independently drawn from the population, Y, the number of common GO terms shared by the two samples, has the following probability mass function

$$P(Y = y) = \frac{\binom{N}{y}\binom{N-y}{|A|-y}\binom{N-|A|}{|B|-y}}{\binom{N}{|A|}\binom{N}{|B|}},\ y = 0, 1, \cdots, \min(|A|, |B|)\,.$$

Hence, this test has a P-value defined below and denoted by P_2.

$$\mathrm{P}_2 = P(Y \geq |A \cap B|) = \sum_{y=|A\cap B|}^{\min(|A|,|B|)} \frac{\binom{N}{y}\binom{N-y}{|A|-y}\binom{N-|A|}{|B|-y}}{\binom{N}{|A|}\binom{N}{|B|}}.$$

S7.3 Ortholog enrichment test

Given a gene co-cluster, the ortholog enrichment test is to check whether the genes from the two species are rich in orthologs. This test will work as a sanity check for BiTSC, which is expected to output co-clusters enriched with orthologs.

For example, given *D. melanogaster* (fly) and *C. elegans* (worm), the two species in our real data application (Section 3.2), we denote the population of ortholog pairs between fly and worm by $O = \{(f_1, w_1), \cdots, (f_M, w_M)\}$, where f_i and w_i are the fly gene and worm gene in the i-th ortholog pair. Note that $f_1, \ldots, f_M$ contain repetitive genes and so do $w_1, \cdots, w_M$. Given a co-cluster, F denotes its set of fly genes, and $F' = \{(f_i, w_i) : f_i \in F\}$ denotes the set of ortholog pairs whose fly genes are in F. Similarly, W denotes the set of worm genes in this co-cluster, and $W' = \{(f_i, w_i) : w_i \in W\}$ denotes the set of ortholog pairs whose worm genes are in W. Note that $F' \cap W'$ is the set of ortholog pairs between fly genes in F and worm genes in W. The null hypothesis is that F' and W' are two independent samples from O, i.e., their common ortholog pairs in $F' \cap W'$ are purely due to a random overlap. The alternative hypothesis is that F' and W' are positively dependent samples. Under the null hypothesis that two samples with sizes $|F'|$ and $|W'|$ are independently drawn from O, Z, the number of ortholog pairs shared by the two samples, has the following probability mass function

$$P(Z = z) = \frac{\binom{M}{z}\binom{M-z}{|F'|-z}\binom{M-|F'|}{|W'|-z}}{\binom{M}{|F'|}\binom{M}{|W'|}}, \; z = 0, 1, \cdots, \min(|F'|, |W'|)\,.$$

Hence, this test has a P-value defined below and denoted by P_3.

$$\mathrm{P}_3 = P(Z \geq |F' \cap W'|) = \sum_{z=|F' \cap W'|}^{\min(|F'|,|W'|)} \frac{\binom{M}{z}\binom{M-z}{|F'|-z}\binom{M-|F'|}{|W'|-z}}{\binom{M}{|F'|}\binom{M}{|W'|}}.$$

S8 Computational time

Under Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0−157−generic x86_64), the computational time for the real data analysis (Section 3.2) is: BiTSC with a single subsampling run took 161 seconds using a single core, and OrthoClust took 258 seconds.

S9 Supplementary Materials

- Folder `Code`: R code for reproducing the statistical analysis in Section 3.2
  - `P_1.R`: R code to perform the GO term enrichment test (Section S7.1) within each co-cluster. The results are in the folder $\mathtt{P}_1$
  - `P_2.R`: R code to perform the GO term overlap test (Section S7.2). The results are in `P_2.xls`
  - `P_3.R`: R code to perform the ortholog enrichment test (Section S7.3). The results are in `P_3.xls`
- Folder `FlyWorm_Result`:
  - Folder `P_1`
  - `P_2.xls`
  - `P_3.xls`
- Folder `Figures`:
  - `heatmap.pdf`: In the application of BiTSC to the fly-worm bipartite network with input parameters $H = 100$, $\rho = 0.8$, and $K_0 = 30$, we varied the value of α in the set

    $$\{1.0, 0.98, 0.96, 0.94, 0.92, 0.90, 0.88, 0.86, 0.84\}$$

    For each α value, we collected the nodes in the resulting tight co-clusters, which we required to have at least 10 nodes on each side, and plotted a heatmap of the submatrix of $\bar{\mathbf{M}}$ that corresponds to these nodes. Then we compared the number of visible blocks in the heatmap with the number of tight co-clusters. Our goal was to choose a large α value for which the two numbers are close, and we chose $\alpha = 0.9$ for our analysis in Section 3.2.

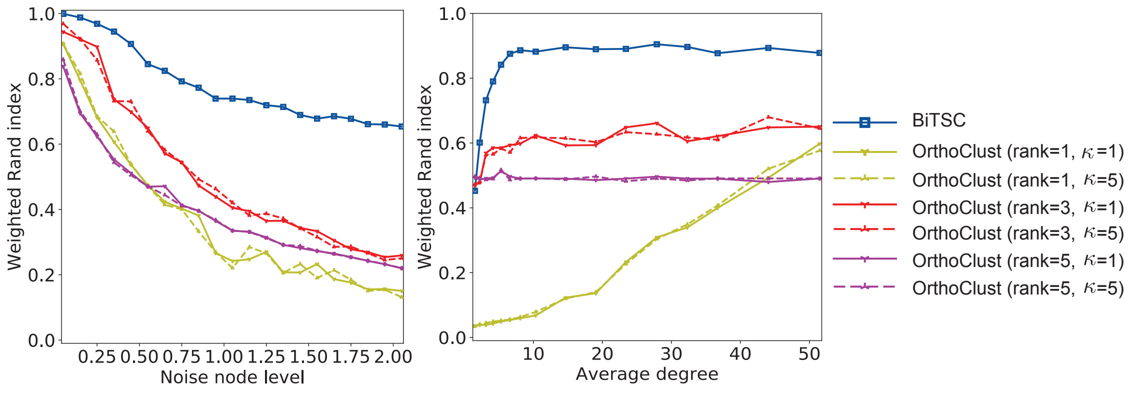


Figure S1: Performance of BiTSC vs. OrthoClust. This figure is an extension of Figure 2. The only difference is that in Figure 2, genes in identified co-clusters that only contain genes from one species are considered noise genes in the calculation of the weighted Rand index, just like unclustered genes. Here, we only consider unclustered genes as noise genes. BiTSC outperforms OrthoClust in a wide range of simulation settings.

Algorithm S1: Automatic selection of K_0 in BiTSC

```
Result: $K_0$
Input: $H$, $\rho$, and $\tau$ for BiTSC; candidate $K_0$'s in the interval $[a, b]$ (default $[5, 50]$)
Initialization:
$K_0\_\min = a$ and $K_0\_\max = b$, $K_0\_\text{mid} = \left\lceil \frac{K_0\_\min + K_0\_\max}{2} \right\rceil$;
Calculate AUC_$K_0$_min and AUC_$K_0$_max as follows:
  1. Run BiTSC with $H = H$, $\rho = \rho$, $\tau = \tau$, and $K_0 = K_0\_\min$ to obtain the consensus matrix;
  2. Plot the empirical cumulative distribution function (CDF) of the entries of the consensus matrix;
  3. Calculate the area under the CDF curve and define it as AUC_$K_0$_min;
  4. Obtain AUC_$K_0$_max similarly based on Step 1-3;
while True do
    if $\frac{|\text{AUC}\_K_0\_\min - \text{AUC}\_K_0\_\max|}{\text{AUC}\_K_0\_\min} \leq 0.05$ then
        Return $K_0$_mid as the selected $K_0$;
    else
        $K_0\_\text{mid} = \left\lceil \frac{K_0\_\min + K_0\_\max}{2} \right\rceil$, calculate AUC_$K_0$_mid based on Step 1-3;
        if | AUC_$K_0$_min − AUC_$K_0$_mid | $\geq$ | AUC_$K_0$_mid − AUC_$K_0$_max | then
            $K_0$_min $\leftarrow$ $K_0$_mid; $K_0$_max $\leftarrow$ $K_0$_max;
            AUC_$K_0$_min $\leftarrow$ AUC_$K_0$_mid; AUC_$K_0$_max $\leftarrow$ AUC_$K_0$_max;
        else
            $K_0$_min $\leftarrow$ $K_0$_min; $K_0$_max $\leftarrow$ $K_0$_mid;
            AUC_$K_0$_min $\leftarrow$ AUC_$K_0$_min; AUC_$K_0$_max $\leftarrow$ AUC_$K_0$_mid;
        end
    end
end
```

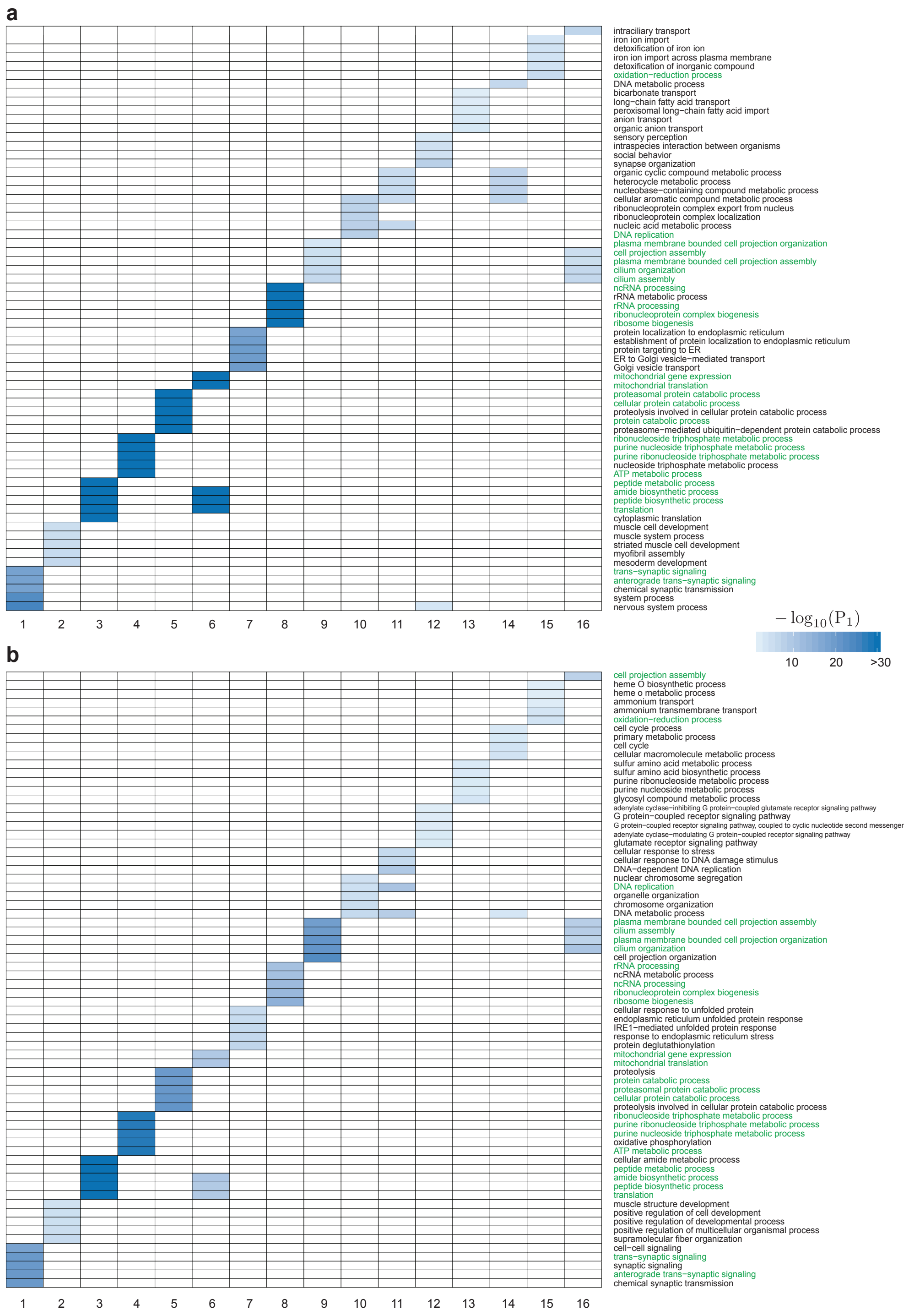


Figure S2: Top five enriched BP GO terms in each of the 16 fly-worm gene co-clusters identified by BiTSC (Section 3.2). (a) GO terms enriched in fly genes of each co-cluster. (b) GO terms enriched in worm genes of each co-cluster. The *P*-value (P_1 value) of each GO term is computed by the GO term enrichment test (Section S7.1). Smaller P_1 values are shown in darker colors to indicate stronger enrichment. For each co-cluster, the top enriched GO terms common to fly and worm are highlighted in green.

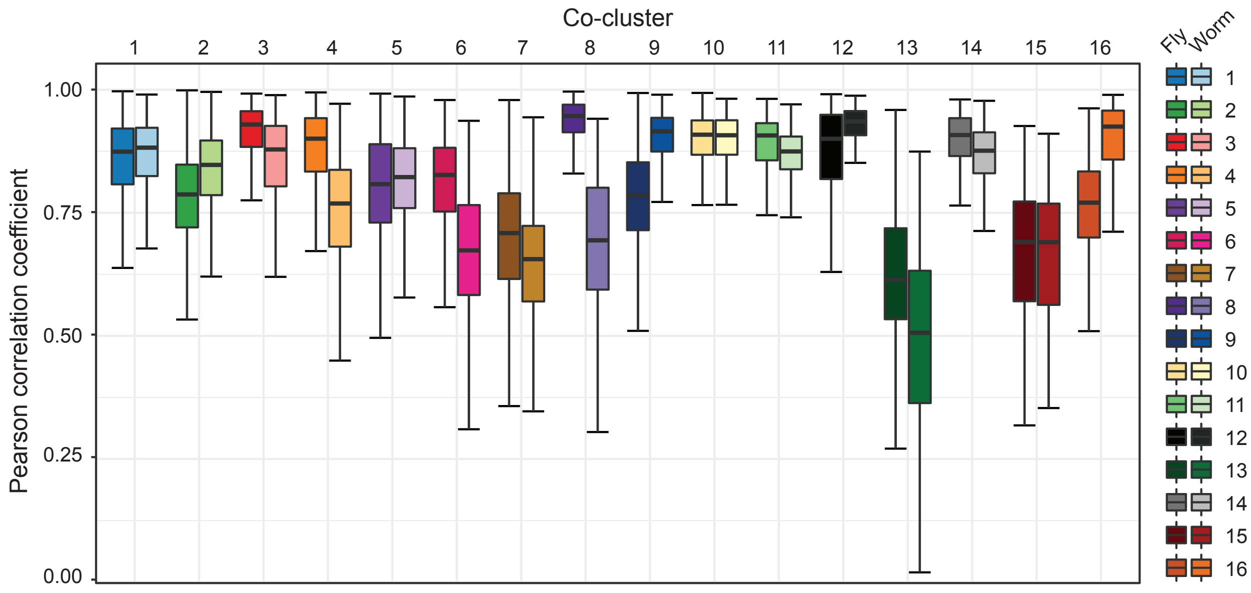


Figure S3: Boxplots of pairwise Pearson correlation coefficients in each species within each of the 16 fly-worm gene co-clusters identified by BiTSC (Section 3.2). In each co-cluster, the Pearson correlation coefficient is computed for every two genes of the same species based on their expression levels, i.e., covariate vectors.

Table S1: Fly-worm gene co-clusters identified by OrthoClust (Section 3.2)

Co-cluster[1]	# of genes Fly	# of genes Worm	P_2[2]	P_3[3]	Average PPC[4] Fly	Average PPC Worm	Average PPC[5] Total
1	216	4	2.54e-03	1.25e-07	0.52758649	0.27495886	0.52752971
2	296	5	4.39e-02	1.14e-08	0.11182023	-0.09538319	0.11177321
3	999	3	9.57e-01	6.71e-04	0.51022172	-0.22759466	0.51021819
4	449	3	3.95e-02	6.07e-05	0.54076835	0.30057179	0.54076212
5	646	5	2.14e-01	4.88e-05	0.06707985	0.20971478	0.06708681
6	530	3	3.47e-01	9.99e-05	0.25106598	0.03598257	0.25106156
7	8	502	4.07e-03	1.34e-11	0.77520991	0.61604135	0.61608481
8	3	442	6.34e-02	5.79e-05	0.46570141	0.52308973	0.52308803
9	2	610	4.44e-01	2.86e-03	0.51949823	0.38322655	0.38322734
10	265	286	1.47e-08	9.72e-110	0.78480461	0.33437622	0.58870568
11	79	105	1.18e-08	1.10e-179	0.90816313	0.62631328	0.7687559
12	5	675	3.37e-03	7.17e-07	0.35234373	0.60067804	0.60066888
13	5	864	2.27e-02	1.54e-04	0.56978234	0.40720803	0.40721284
14	4	467	2.58e-04	2.78e-06	0.13064446	0.51414297	0.51412525

[1] With $\kappa = 1$, OrthoClust outputs a total of 24 clusters, among which 14 co-clusters have ≥ 2 genes in each species.
[2] P-value of the co-cluster based on the GO term overlap test (Section S7.2).
[3] P-value of the co-cluster based on the ortholog enrichment test (Section S7.3).
[4] The average pairwise Pearson correlation (PPC) is computed for all fly gene pairs in each co-cluster.
[5] The average PPC is computed for all gene pairs of the same species in each co-cluster. Values in this column are shown in Figure 3b-c.

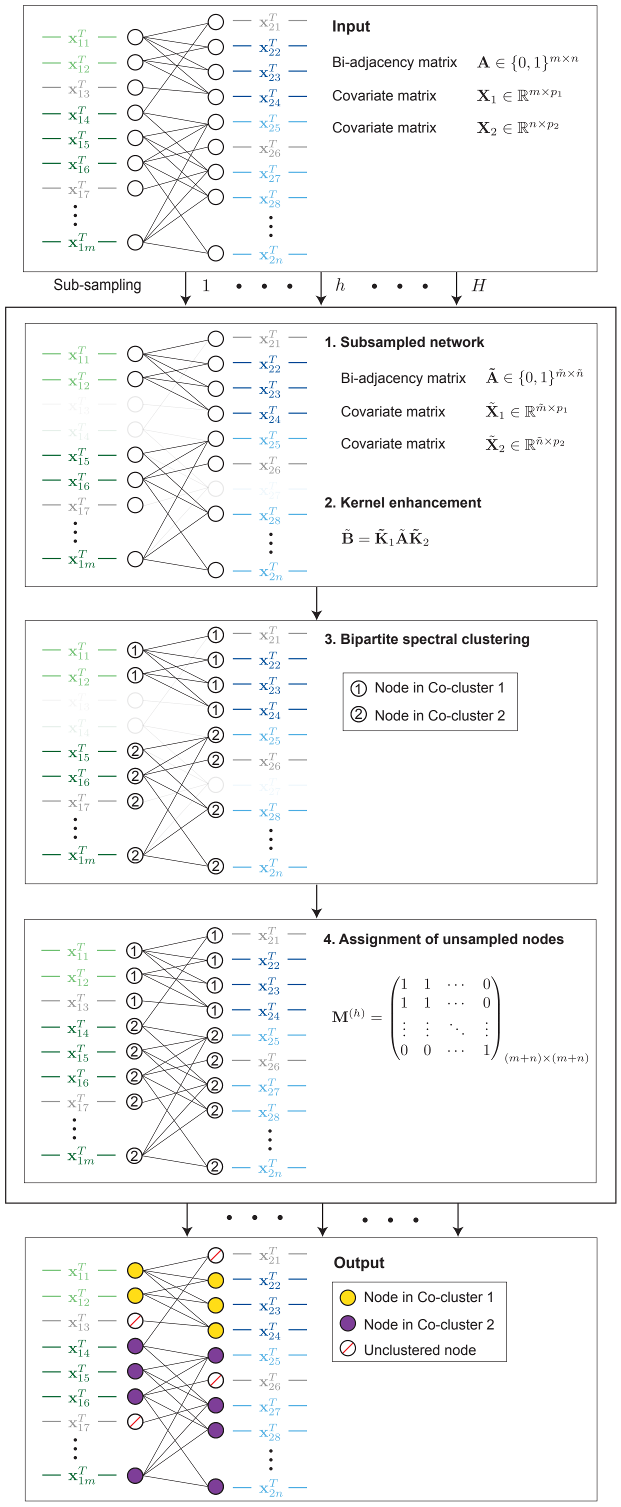

Input
Bi-adjacency matrix $\mathbf{A} \in \{0,1\}^{m\times n}$
Covariate matrix $\mathbf{X}_1 \in \mathbb{R}^{m\times p_1}$
Covariate matrix $\mathbf{X}_2 \in \mathbb{R}^{n\times p_2}$
Sub-sampling
1
h
H
1. Subsampled network
Bi-adjacency matrix $\tilde{\mathbf{A}} \in \{0,1\}^{\tilde{m}\times \tilde{n}}$
Covariate matrix $\tilde{\mathbf{X}}_1 \in \mathbb{R}^{\tilde{m}\times p_1}$
Covariate matrix $\tilde{\mathbf{X}}_2 \in \mathbb{R}^{\tilde{n}\times p_2}$
2. Kernel enhancement
$\tilde{\mathbf{B}} = \tilde{\mathbf{K}}_1\tilde{\mathbf{A}}\tilde{\mathbf{K}}_2$
3. Bipartite spectral clustering
Node in Co-cluster 1
Node in Co-cluster 2
4. Assignment of unsampled nodes
$\mathbf{M}^{(h)} = \begin{pmatrix} 1 & 1 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}_{(m+n)\times(m+n)}$
Output
Node in Co-cluster 1
Node in Co-cluster 2
Unclustered node
$\mathbf{x}_{11}^T$
$\mathbf{x}_{12}^T$
$\mathbf{x}_{13}^T$
$\mathbf{x}_{14}^T$
$\mathbf{x}_{15}^T$
$\mathbf{x}_{16}^T$
$\mathbf{x}_{17}^T$
$\mathbf{x}_{1m}^T$
$\mathbf{x}_{21}^T$
$\mathbf{x}_{22}^T$
$\mathbf{x}_{23}^T$
$\mathbf{x}_{24}^T$
$\mathbf{x}_{25}^T$
$\mathbf{x}_{26}^T$
$\mathbf{x}_{27}^T$
$\mathbf{x}_{28}^T$
$\mathbf{x}_{2n}^T$

Figure S4: Workflow of BiTSC (Section 2.2).

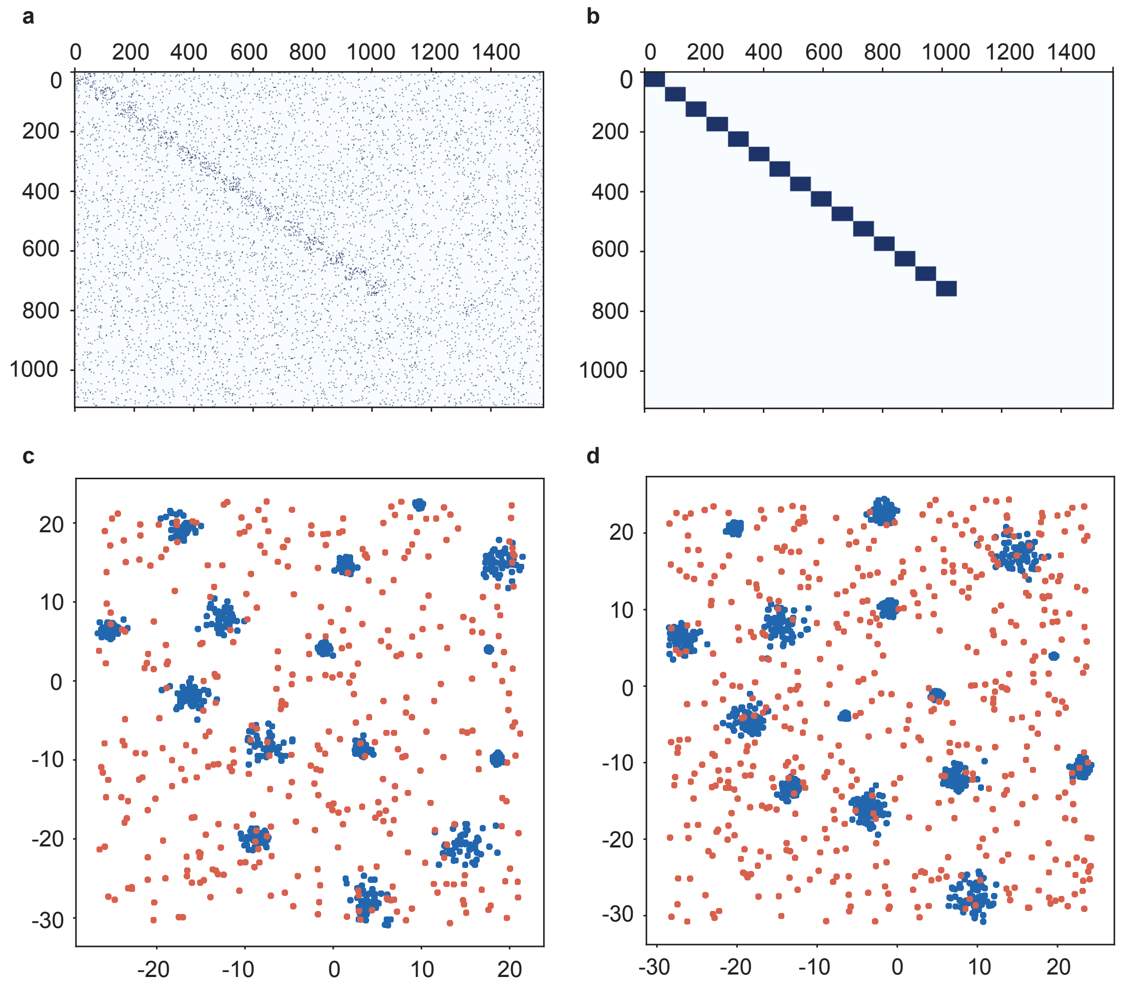


Figure S5: The simulated bipartite network described in Section S2. (a) The bi-adjacency matrix. (b) The true co-membership matrix. In both matrices, entries of ones are shown in blue colors. Nodes belonging to the same true co-cluster are ordered next to each other. (c) The node covariates on side 1. (d) The node covariates on side 2. Each point corresponds to one node, and the two axes represent the two dimensions of node covariates. Nodes in the 15 true co-clusters are marked in blue, and noise nodes not belonging to any co-clusters are marked in red.

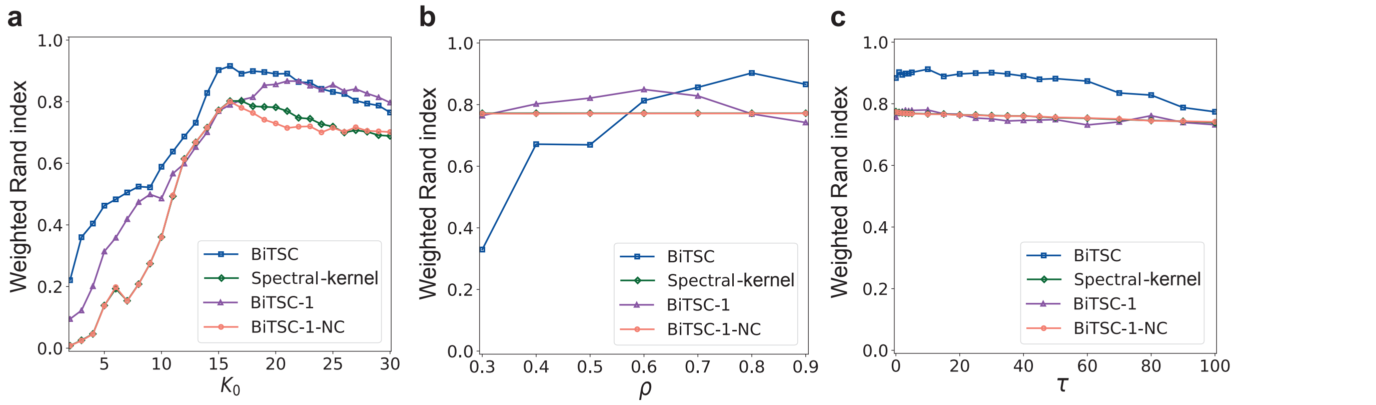


Figure S6: Clustering performance (weighted Rand index) of BiTSC and three variant algorithms (Section S1) as functions of (a) K_0, the number of clusters in each run, (b) the subsampling proportion ρ, and (c) the kernel enhancement parameter τ. A bipartite network was simulated as described in Section S2 using $K = 15$, $n_1 = 50$, $n_2 = 70$, $p_1 = p_2 = 2$, $\sigma_1 = \sigma_2 = 10$, $\omega_1 = \omega_2 = 0.1$, $\theta = 0.5$, $p = 0.15$, and $q = p/5$. We set $H = 50$ and $\alpha = 0.7$ whenever applicable. For (a), we set $\rho = 0.8$ and $\tau = (1, 1)$. For (b), we set $K_0 = 15$ and $\tau = (1, 1)$. For (c), we set $K_0 = 15$ and $\rho = 0.8$.

Table S2: Fly-worm gene co-clusters identified by OrthoClust (Section 3.2)

Co-cluster[1]	# of genes Fly	# of genes Worm	P_2[2]	P_3[3]	Average PPC[4] Fly	Average PPC Worm	Average PPC[5] Total
1	412	4	3.14e-01	1.68e-06	0.1812038	-0.05888426	0.1811872
2	417	2	1.00e-00	1.33e-03	0.24701688	-0.08194976	0.2470133
3	216	16	1.11e-06	3.51e-16	0.52421706	0.29469462	0.5231781
4	412	3	4.16e-02	4.68e-05	0.115830253	-0.05725930	0.1158242
5	900	5	8.00e-01	3.03e-06	0.51815480	0.08545191	0.518146
6	650	2	2.93e-01	3.24e-03	0.63868879	0.41859354	0.6386879
7	4	692	3.35e-05	1.35e-05	-0.13076364	0.5729412	0.572928
8	4	734	2.15e-01	1.70e-05	0.021641929	0.473630719	0.4736222
9	250	361	3.40e-04	1.20e-151	0.78019168	0.33264337	0.5172098
10	6	1067	2.32e-01	3.94e-05	0.49751377	0.42596621	0.4259682
11	2	427	2.72e-01	1.40e-03	-0.64244048	0.10860390	0.1085944
12	78	99	1.23e-06	3.40e-176	0.92014421	0.70566047	0.8179273
13	2	517	1.00e-00	2.05e-03	-0.796179190	0.240137690	0.2401283
14	2	692	1.14e-01	2.23e-04	0.296111471	0.50715018	0.5071478

[1] With $\kappa = 3$, OrthoClust outputs a total of 22 clusters, among which 14 co-clusters have ≥ 2 genes in each species.

[2] P-value of the co-cluster based on the GO term overlap test (Section S7.2).

[3] P-value of the co-cluster based on the ortholog enrichment test (Section S7.3).

[4] The average pairwise Pearson correlation (PPC) is computed for all fly gene pairs in each co-cluster.

[5] The average PPC is computed for all gene pairs of the same species in each co-cluster.

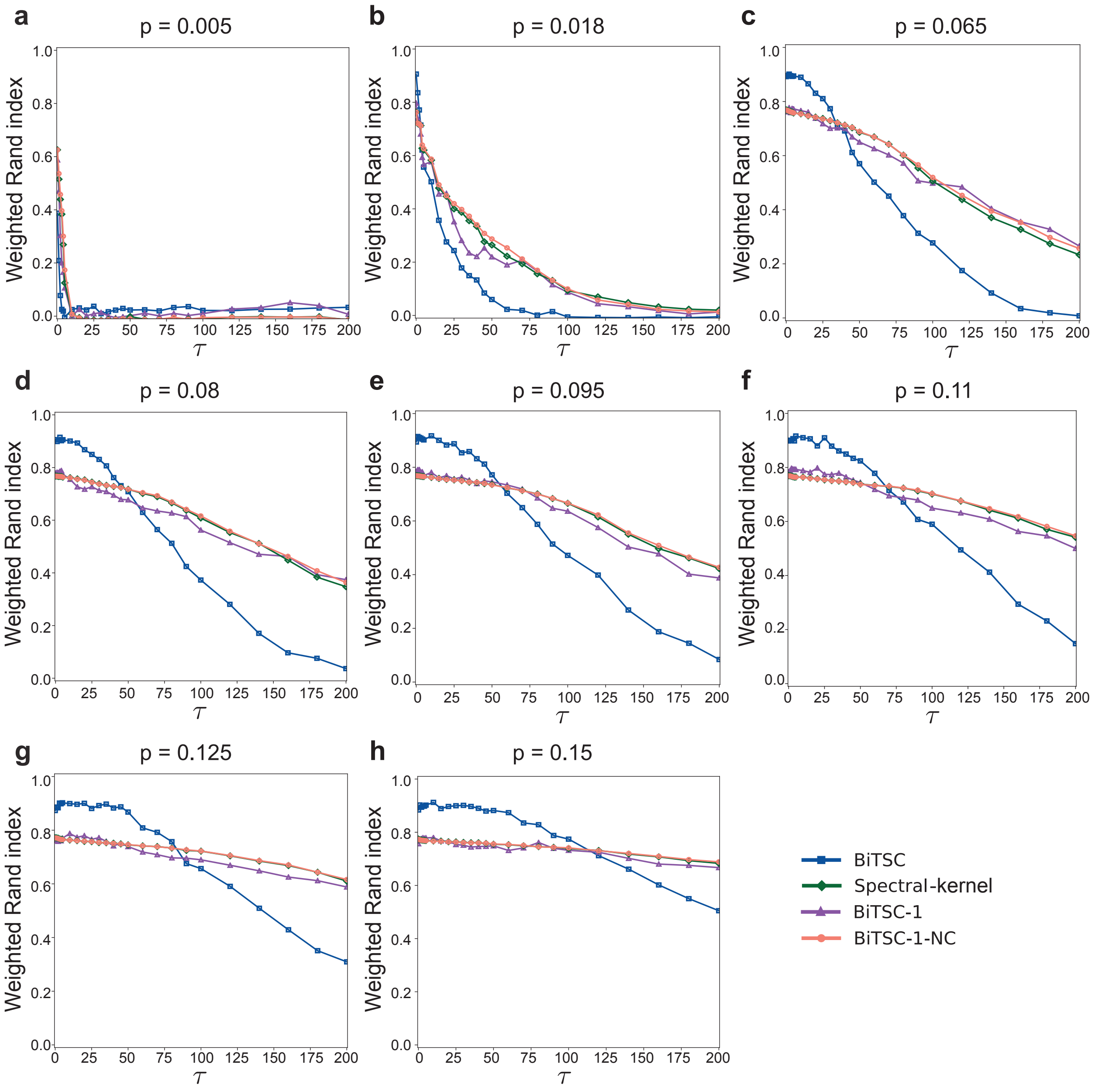


Figure S7: Clustering performance (weighted Rand index) of BiTSC and three variant algorithms (Section S1) as functions of τ ($\tau_1 = \tau_2$). For each of (a) through (h), a bipartite network was simulated following the procedure described in Section S2 using $K = 15$, $n_1 = 50$, $n_2 = 70$, $p_1 = p_2 = 2$, $\sigma_1 = \sigma_2 = 10$, $\omega_1 = \omega_2 = 0.1$, $\theta = 0.5$, and $q = p/5$. For the input parameters of the algorithms, we set $H = 50$, $\rho = 0.8$, $K_0 = K = 15$, and $\alpha = 0.7$ whenever applicable. Note that Figure S6c is a truncated version of panel (h) here.

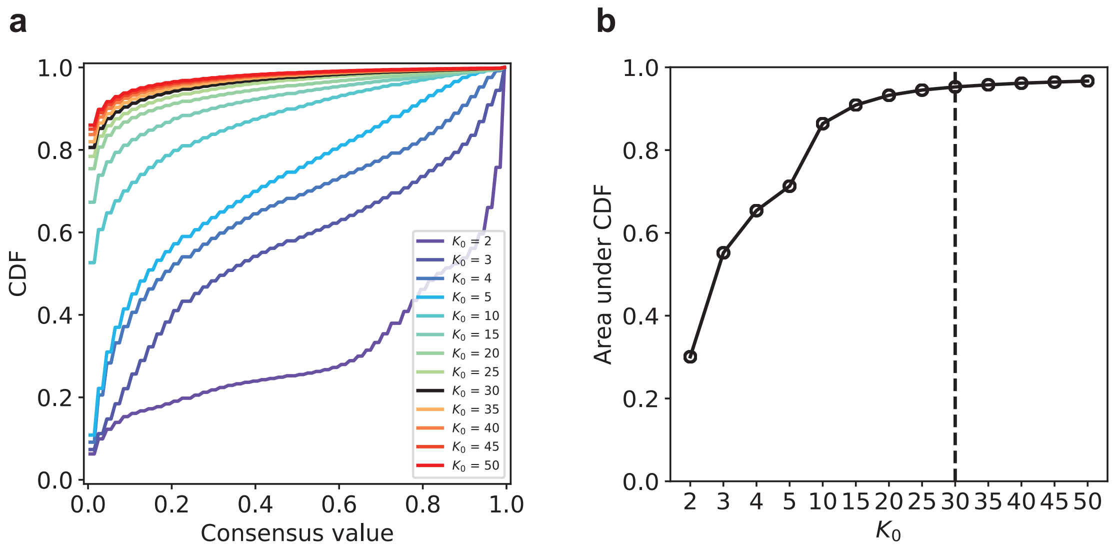


Figure S8: (a) The empirical CDFs of the entries of the consensus matrix for K_0 between 2 and 50 (in increments of 5 after K_0 passing 5) in the real data example (Section 3.2). (b) The area under CDF curve as a function of K_0. $K_0 = 30$ was chosen.

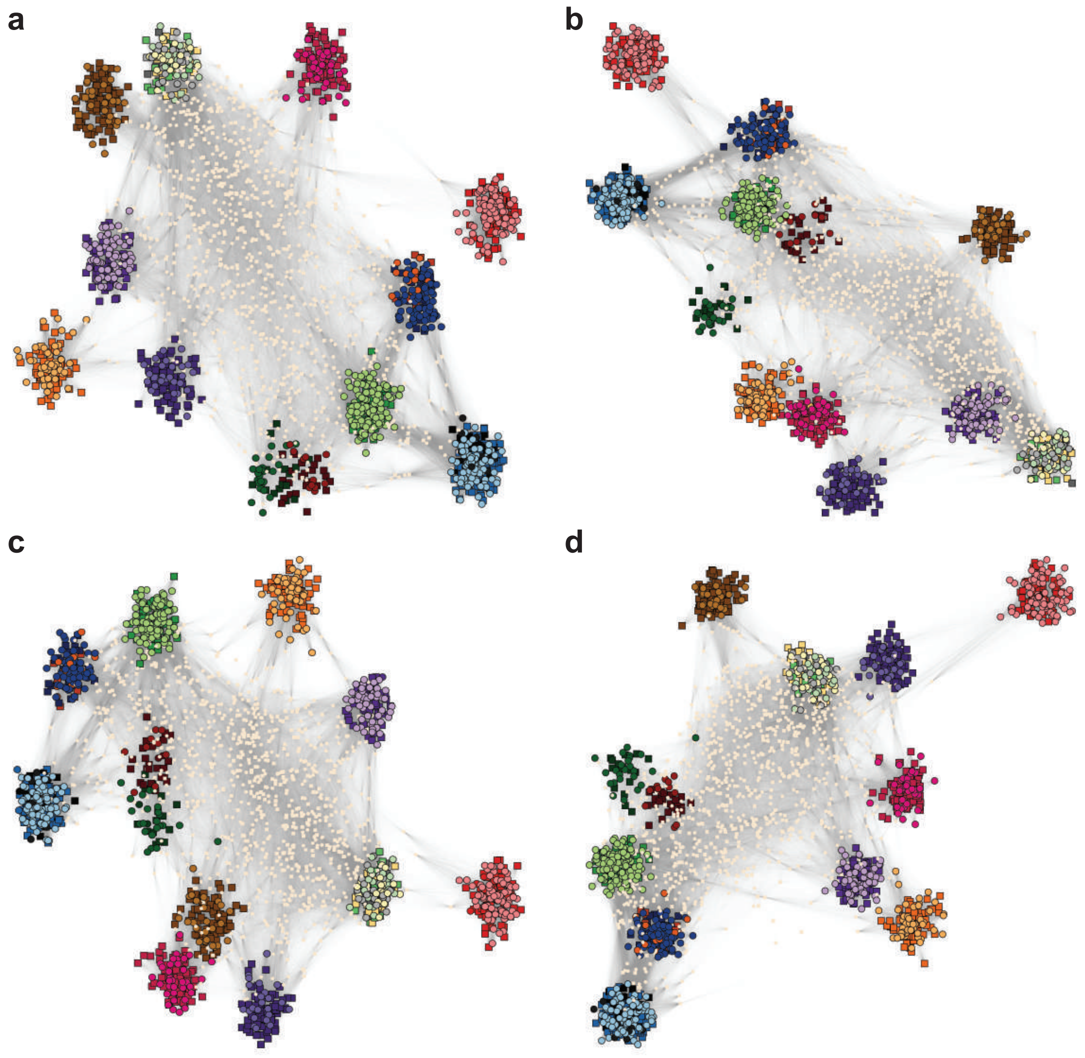
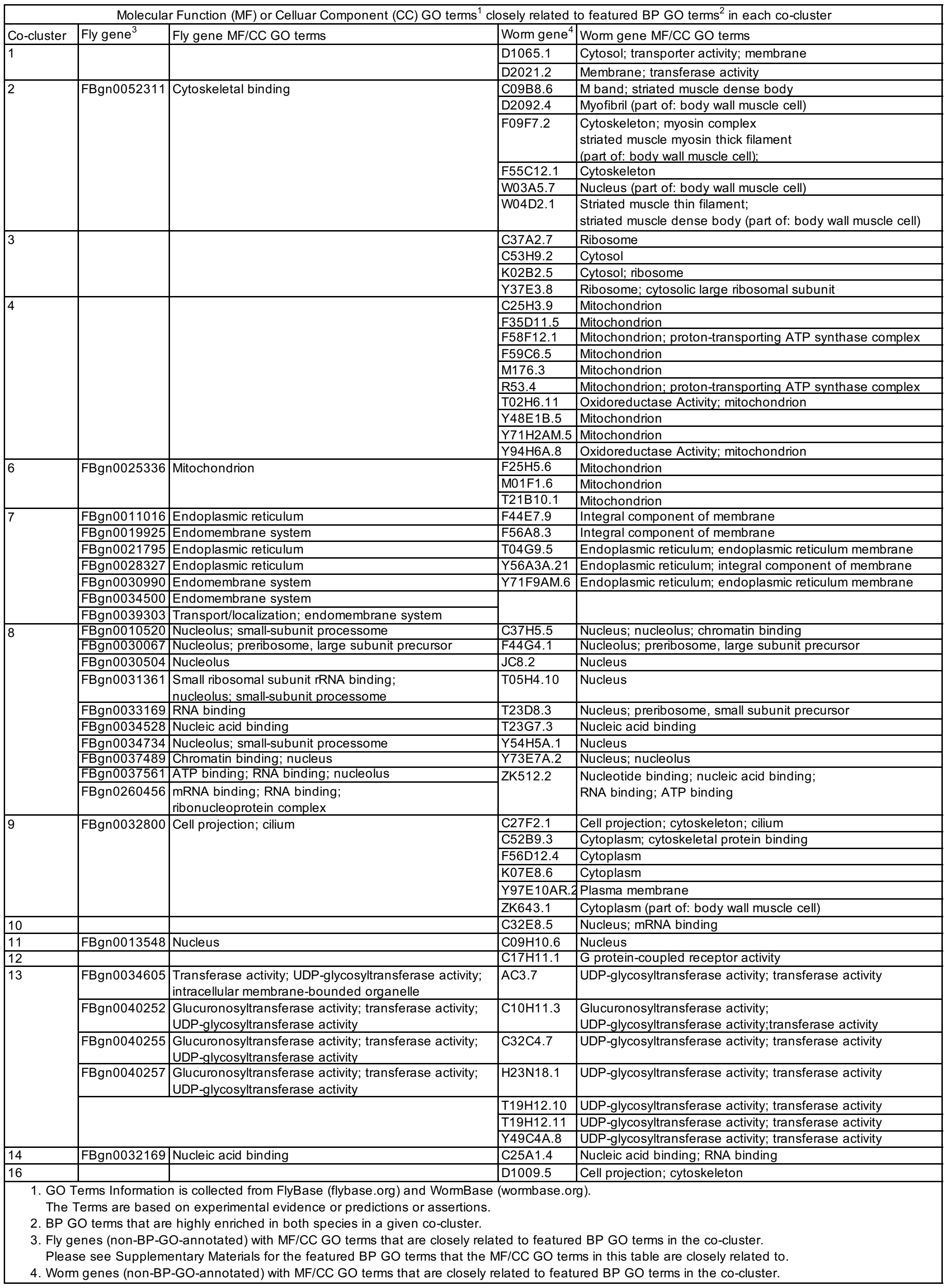


Figure S9: Robustness of the random selection of 1,000 unclustered genes in Figure 4. (a)-(d) Visualization of the 16 gene co-clusters found by BiTSC, same as in Figure 4 except that four different random seeds were used to sample the 1,000 unclustered genes.



Molecular Function (MF) or Celluar Component (CC) GO terms[1] closely related to featured BP GO terms[2] in each co-cluster				
Co-cluster	Fly gene[3]	Fly gene MF/CC GO terms	Worm gene[4]	Worm gene MF/CC GO terms
1			D1065.1	Cytosol; transporter activity; membrane
			D2021.2	Membrane; transferase activity
2	FBgn0052311	Cytoskeletal binding	C09B8.6	M band; striated muscle dense body
			D2092.4	Myofibril (part of: body wall muscle cell)
			F09F7.2	Cytoskeleton; myosin complex striated muscle myosin thick filament (part of: body wall muscle cell);
			F55C12.1	Cytoskeleton
			W03A5.7	Nucleus (part of: body wall muscle cell)
			W04D2.1	Striated muscle thin filament; striated muscle dense body (part of: body wall muscle cell)
3			C37A2.7	Ribosome
			C53H9.2	Cytosol
			K02B2.5	Cytosol; ribosome
			Y37E3.8	Ribosome; cytosolic large ribosomal subunit
4			C25H3.9	Mitochondrion
			F35D11.5	Mitochondrion
			F58F12.1	Mitochondrion; proton-transporting ATP synthase complex
			F59C6.5	Mitochondrion
			M176.3	Mitochondrion
			R53.4	Mitochondrion; proton-transporting ATP synthase complex
			T02H6.11	Oxidoreductase Activity; mitochondrion
			Y48E1B.5	Mitochondrion
			Y71H2AM.5	Mitochondrion
			Y94H6A.8	Oxidoreductase Activity; mitochondrion
6	FBgn0025336	Mitochondrion	F25H5.6	Mitochondrion
			M01F1.6	Mitochondrion
			T21B10.1	Mitochondrion
7	FBgn0011016	Endoplasmic reticulum	F44E7.9	Integral component of membrane
	FBgn0019925	Endomembrane system	F56A8.3	Integral component of membrane
	FBgn0021795	Endoplasmic reticulum	T04G9.5	Endoplasmic reticulum; endoplasmic reticulum membrane
	FBgn0028327	Endoplasmic reticulum	Y56A3A.21	Endoplasmic reticulum; integral component of membrane
	FBgn0030990	Endomembrane system	Y71F9AM.6	Endoplasmic reticulum; endoplasmic reticulum membrane
	FBgn0034500	Endomembrane system		
	FBgn0039303	Transport/localization; endomembrane system		
8	FBgn0010520	Nucleolus; small-subunit processome	C37H5.5	Nucleus; nucleolus; chromatin binding
	FBgn0030067	Nucleolus; preribosome, large subunit precursor	F44G4.1	Nucleolus; preribosome, large subunit precursor
	FBgn0030504	Nucleolus	JC8.2	Nucleus
	FBgn0031361	Small ribosomal subunit rRNA binding; nucleolus; small-subunit processome	T05H4.10	Nucleus
	FBgn0033169	RNA binding	T23D8.3	Nucleus; preribosome, small subunit precursor
	FBgn0034528	Nucleic acid binding	T23G7.3	Nucleic acid binding
	FBgn0034734	Nucleolus; small-subunit processome	Y54H5A.1	Nucleus
	FBgn0037489	Chromatin binding; nucleus	Y73E7A.2	Nucleus; nucleolus
	FBgn0037561	ATP binding; RNA binding; nucleolus	ZK512.2	Nucleotide binding; nucleic acid binding; RNA binding; ATP binding
	FBgn0260456	mRNA binding; RNA binding; ribonucleoprotein complex		
9	FBgn0032800	Cell projection; cilium	C27F2.1	Cell projection; cytoskeleton; cilium
			C52B9.3	Cytoplasm; cytoskeletal protein binding
			F56D12.4	Cytoplasm
			K07E8.6	Cytoplasm
			Y97E10AR.2	Plasma membrane
			ZK643.1	Cytoplasm (part of: body wall muscle cell)
10			C32E8.5	Nucleus; mRNA binding
11	FBgn0013548	Nucleus	C09H10.6	Nucleus
12			C17H11.1	G protein-coupled receptor activity
13	FBgn0034605	Transferase activity; UDP-glycosyltransferase activity; intracellular membrane-bounded organelle	AC3.7	UDP-glycosyltransferase activity; transferase activity
	FBgn0040252	Glucuronosyltransferase activity; transferase activity; UDP-glycosyltransferase activity	C10H11.3	Glucuronosyltransferase activity; UDP-glycosyltransferase activity;transferase activity
	FBgn0040255	Glucuronosyltransferase activity; transferase activity; UDP-glycosyltransferase activity	C32C4.7	UDP-glycosyltransferase activity; transferase activity
	FBgn0040257	Glucuronosyltransferase activity; transferase activity; UDP-glycosyltransferase activity	H23N18.1	UDP-glycosyltransferase activity; transferase activity
			T19H12.10	UDP-glycosyltransferase activity; transferase activity
			T19H12.11	UDP-glycosyltransferase activity; transferase activity
			Y49C4A.8	UDP-glycosyltransferase activity; transferase activity
14	FBgn0032169	Nucleic acid binding	C25A1.4	Nucleic acid binding; RNA binding
16			D1009.5	Cell projection; cytoskeleton

1. GO Terms Information is collected from FlyBase (flybase.org) and WormBase (wormbase.org). The Terms are based on experimental evidence or predictions or assertions.
2. BP GO terms that are highly enriched in both species in a given co-cluster.
3. Fly genes (non-BP-GO-annotated) with MF/CC GO terms that are closely related to featured BP GO terms in the co-cluster. Please see Supplementary Materials for the featured BP GO terms that the MF/CC GO terms in this table are closely related to.
4. Worm genes (non-BP-GO-annotated) with MF/CC GO terms that are closely related to featured BP GO terms in the co-cluster.

Figure S10: MF and CC GO terms of the genes without BP GO terms in the 16 gene co-clusters identified by BiTSC (Section 3.2).

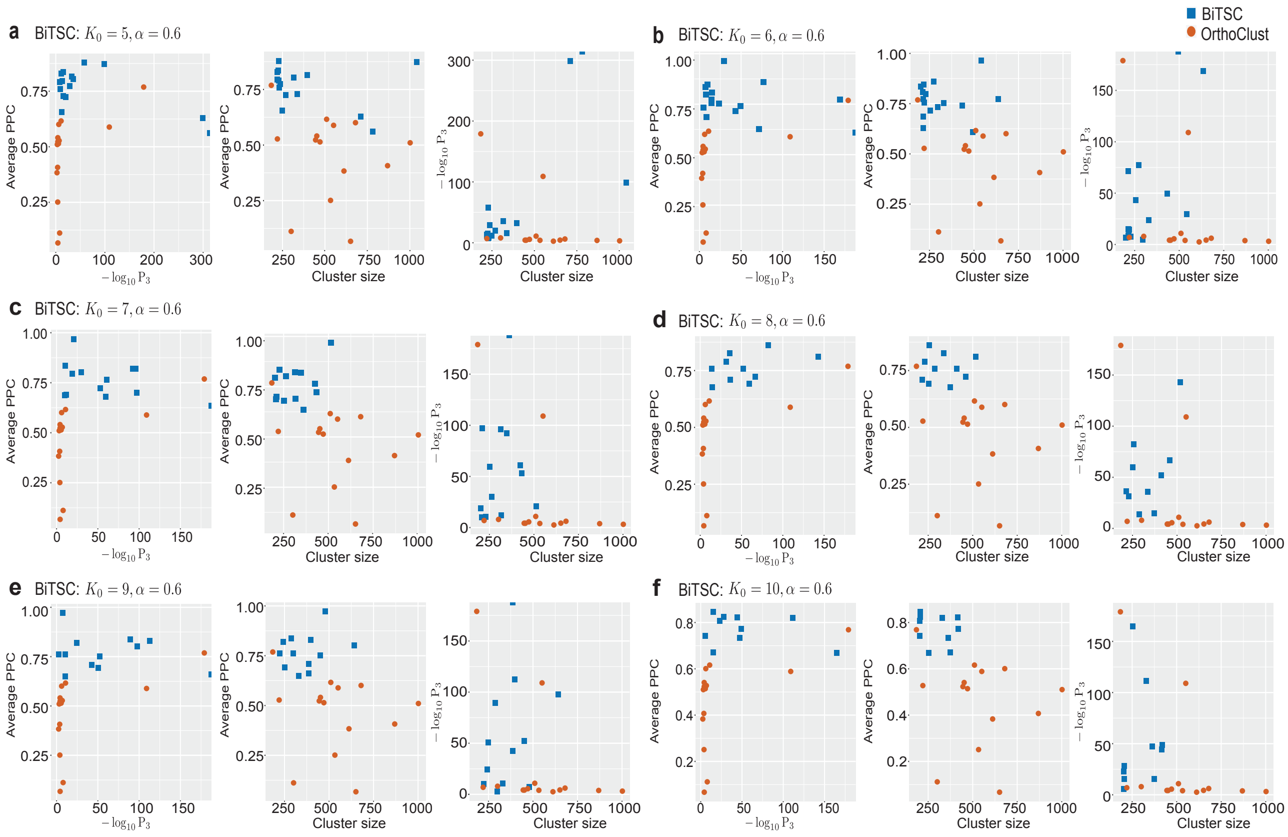


Figure S11: Comparison of BiTSC and OrthoClust in terms of their identified co-clusters (Section 3.2.1). This figure is an extension of Figure 3. After running the OrthoClust algorithm, we kept the 14 co-clusters with at least two genes in each species (Section 3.2.1; Table S1). The cluster sizes are between 184 and 1002. For BiTSC, we set $H = 100$, $\rho = 0.8$, and $\tau = (1, 1)$ as we did in Section 3.2.1. Then we varied K_0 between 5 and 10 and set $\alpha = 0.6$ so that BiTSC outputs co-clusters with sizes comparable to those identified by OrthoClust. For a given choice of K_0 and α, we kept the co-clusters that have at least 10 genes from each species and at least 200 genes total. Here, This figure shows that controlling for cluster size, the co-clusters identified by BiTSC exhibit higher gene expression similarity (second column of each panel) and greater enrichment of orthologs (third column of each panel).

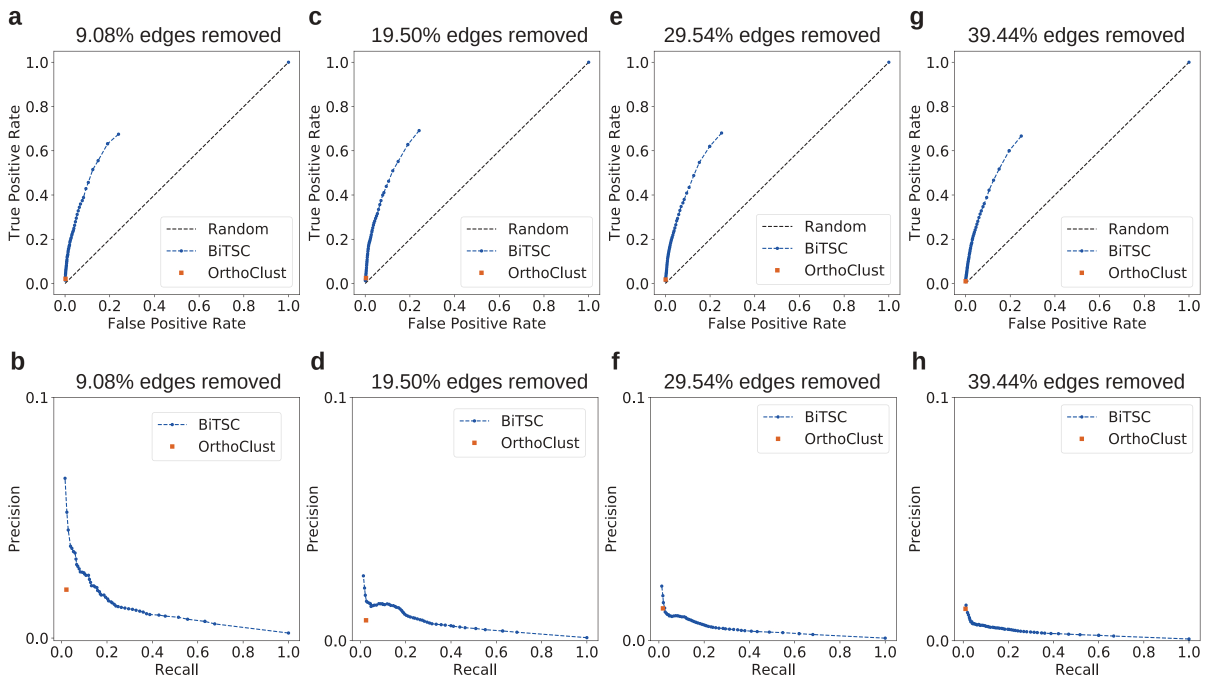


Figure S12: Comparison of BiTSC and OrthoClust in terms of link prediction performance on the fly-worm data set (Section 3.2). From the 5414 fly genes and the 5731 worm genes in the fly-worm data set, we sampled a proportion of fly genes and worm genes each, removed all edges among the sampled genes from the original network, and run BiTSC on the remaining network with $H = 100, \rho = 0.8, \tau = (1, 1)$, and $K_0 = 30$ (the same parameters as we used in Section 3.2) as well as OrthoClust. The number of removed edges divided by the number of edges in the original network $(10, 975)$ is shown in the title of each plot. For BiTSC, we performed link prediction by thresholding the consensus matrix with cutoffs between 0 and 1. For OrthoClust, we performed link prediction by connecting every pair of genes in the same identified co-cluster. Denote the set of sampled fly genes as F and the set of sampled worm genes as W. Let $F' \subset F$ be the set of fly genes in F that are orthologous to at least one gene in W, and similarly, let $W' \subset W$ be the set of worm genes in W that are orthologous to at least one gene in F. That is, we removed the "isolated" genes in F and W to obtain F' and W'. Then, the link prediction performance of BiTSC and OrthoClust was evaluated based on the set of all gene pairs between F' and W' and the ROC curves and precision-recall curves were obtained accordingly. We chose to evaluate BiTSC and OrthoClust based on all gene pairs between F' and W' instead of F and W due to the low edge density in the original network. If we considered all gene pairs between F and W, then the negatives would overwhelm the positives in the truth.

References

[Alexa and Rahnenfuhrer, 2019] Alexa, A. and Rahnenfuhrer, J. (2019). *topGO: Enrichment Analysis for Gene Ontology*. R package version 2.36.0.

[Carlson, 2019] Carlson, M. (2019). *GO.db: A set of annotation maps describing the entire Gene Ontology*. R package version 3.8.2.

[Cowell, 2011] Cowell, F. (2011). *Measuring Inequality*. Oxford University Press, 3 edition.

[Danielsson et al., 2015] Danielsson, F. et al. (2015). Assessing the consistency of public human tissue rna-seq data sets. *Briefings in Bioinformatics*, 16:941–949.

[Gerstein et al., 2014] Gerstein, M. et al. (2014). Comparative analysis of the transcriptome across distant species. *Nature*, 512(7515):445–448.

[Hubert and Arabie, 1985] Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2:193–218.

[Li et al., 2006] Li, H., Coghlan, A., et al. (2006). Treefam: a curated database of phylogenetic trees of animal gene families. *Nucleic Acids Research*, 34:572–580.

[Li et al., 2014] Li, J. J. et al. (2014). Comparison of d. melanogaster and c. elegans developmental stages, tissues, and cells by modencode rna-seq data. *Genome Research*.

[Monti et al., 2003] Monti, S., Tamayo, P., Mesirov, J., and Golub, T. (2003). Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning*, 52:91–118.

[Nowicki and Snijders, 2001] Nowicki, K. and Snijders, T. A. B. (2001). Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087.

[Pedregosa et al., 2011] Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Pertea et al., 2016] Pertea, M. et al. (2016). Transcript-level expression analysis of rna-seq experiments with hisat, stringtie and ballgown. *Nature Protocols*, 11(9):1605–1667.

[Rand, 1971] Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.

[Thalamuthu et al., 2006] Thalamuthu, A. et al. (2006). Evaluation and comparison of gene clustering methods in microarray analysis. *Bioinformatics*, 22(19):2405–2412.

[Zwiener et al., 2014] Zwiener, I., Frisch, B., and Binder, H. (2014). Transforming rna-seq data to improve the performance of prognostic gene signatures. *PLOS ONE*, 9(1):1–13.