

A SNN-Walktrap: Motivation and Overview

SNN-Walktrap is the workhorse of our algorithm. SNN can be broken down into three steps (Main Paper: Figure 1 , red box):

Step (a) : Construct SNN Network Construction of shared-nearest-neighbor is the corner stone of SNN-Cliq [Xu and Su (2015)], the method that BiSNN-Walk expands on. SNN network defines the notion of “distance” between two nodes within the context of a local neighborhood as opposed to a distance quantified by an global measure (e.g Euclidean distance). This localization is desirable for high dimensional data, where the high dimensionality renders global measures like Euclidean norm less useful as a proxy for distance [Aggarwal et al. (2001)]. Please find an overview of the construction of a SNN network and the rationale behind its use in Appendix C.

Step (b) : Walktrap Clustering After a network is constructed using SNN, we use Walktrap [Pons and Latapy (2005)] to perform cell clustering. The Walktrap algorithm is an agglomerative hierarchical clustering scheme akin to a complete-linkage hierarchical clustering. The distance between node i and node j is related to the difference in the behaviors of two random walks starting at the two nodes. A very important feature of Walktrap is that one does not need specify a priori the number of clusters. Cutting threshold of the tree is set automatically and is related to the distance measure. The intuitiveness of the cutting threshold was a major reason for choosing Walktrap as our clustering algorithm. Please refer to Appendix D for an overview of the algorithm.

Step (c) : Select Candidate Cluster Walktrap, being a clustering algorithm, will identify several cell clusters; our purpose here, however, is to find the best one. To define “best”, we use a heuristic involving three common clustering metrics: conductance, transitivity, and the Jaccard score. Please refer to Appendix H for details.

B Finding Characteristic Genes

B.1 Selecting Characteristic Genes Selection

Let C denote a cluster, and Q denote the quantile matrix, i.e Q is a $g \times n$ matrix such that Q_{ij} = quantile of gene i 's expression level for cell j . Quantizing the raw expression level is a form of normalization that makes the expression across cells more comparable. Define the contrast of gene i on cluster C as

$$z_{i,C} = \text{median}(Q_{ij} : j \in C) - 75^{\text{th}}\text{Quantile}(Q_{ij} : j \notin C) \quad (1)$$

We call gene i “characteristic to cluster C ” if $z_{i,C} > 0$. In other words, the characteristic genes are those that are generally more highly expressed in C than the rest of the cells. The characteristic genes will be ranked according to their contrast—genes with higher contrast are more representative of the cluster. We could, of course, replace $75^{\text{th}}\text{Quantile}$ with Median and the Max in equation (1), but our concern is that the median would result in too liberal a list for genes to be called cluster-specific “characteristic genes”, while the max would return too conservative a list and would thus remove potentially useful clustering information; $75^{\text{th}}\text{Quantile}$, therefore, was chosen as compromise.

B.2 On Using Characteristic Genes for Subsequent Analysis

As noted in the flowchart (Main Paper: Figure 1, ③), the selected genes will be used to subset the gene expression matrix, which will be fed into the SNN-Walktrap procedure. The rationale behind using only the characteristic genes as our next input is to rid of impurities in the cluster. Assuming our initial cluster mostly contains cells of one state, then it’s reasonable to believe that the characteristic genes associated with this cluster will most likely be most relevant to that state. Thus we will see these cells forming a tighter group in a SNN network constructed using only the characteristic genes, thus removing cells of a foreign state. Main Paper: Figure 2 demonstrates this “purification” step at work.

C Construction of SNN Network

Let's demonstrate the edge-weight calculation using a simple example. Let X be a 2×8 matrix, i.e we have 2 genes and 8 cells with fabricated gene expressions, shown in Figure 1.

$$X = \begin{bmatrix} 0 & 0 & 0.25 & 0.25 & 1.5 & 1.75 & 1.5 & 1.75 \\ 0 & 0.25 & 0.25 & 0 & 1.5 & 1.5 & 1.75 & 1.75 \end{bmatrix}$$

Appendix Figure 1

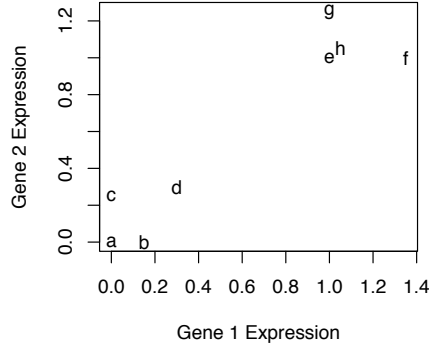


Figure 1: *Positions of cells in gene expression space.* This shows the relative positions of the cells in the gene expression space in our example. It is apparent that our cells should be grouped into two clusters $\{a, b, c, d\}$ and $\{e, f, g, h\}$.

Let's calculate the edge weight between a and b . First find the list of neighbors ranked in order of proximity (using Euclidean distance), in this case a 's neighborhood is $\{a, b, c, d, e, h, g, f\}$ and that of b is $\{b, a, c, d, e, h, g, f\}$. Then define an integer k so that we only look at the top k neighbors in each list (this is why sometimes shared-nearest-neighbor is also called the k -nearest-neighbor). Let $k = 3$, then the neighbor list we actually use are $\{a, b, c\}$ for a and $\{b, a, c\}$ for b . Searching through the pair of listings, we find the highest positions of their common neighbors, in this case, a , who is ranked 0 in a 's neighborhood and 1 in b 's (or b , who is ranked 1 in a 's neighborhood and 0 in b 's). Note that even though c is a common neighbor, it is ranked lower than the other common neighbors (i.e a and b) in the list, so it is not used to calculate proximity. The average rank of the highest common neighbor in this case is $\frac{0+1}{2} = 0.5$, and the edge weight is therefore $k - 0.5 = 3 - 0.5 = 2.5$. Take another example, suppose we want to create an edge between a and e , with $k = 3$, then the corresponding neighborhood lists are $\{a, b, c\}$ and $\{e, h, g\}$. Since no common neighbor exists, the an edge will not be drawn between a and e in the final graph. Using the procedure described above, Figure 2 shows the networks create using SNN constructor with $k = 2$ and $k = 3$, respectively.

In general, let n be the number of genes and m be the number of cells, then the $n \times m$ gene expression represents m points in \mathbb{R}^n , the algorithm constructs a network with nodes being cells and edge weights between two cells being pseudo-measure of their proximity in \mathbb{R}^n . Let c denote a cell, define a positive integer k to be the neighborhood size such that $V_k(c)$ is an ordered list of k cells who are c 's closest neighbor measured in Euclidean distance, with the first element of $V_k(c)$ being the closest to c . Define $rank_k(a, c)$ to be the position of cell a in $V_k(c)$. Then the weight of the edge between cells a and c is defined as

$$w(a, c) \triangleq \begin{cases} \max \left\{ k - \frac{rank_k(b, a) + rank_k(b, c)}{2} \right\} & \text{if } c \in V(a) \cup V(c) \\ 0 & \text{o.w} \end{cases}$$

Appendix Figure 2

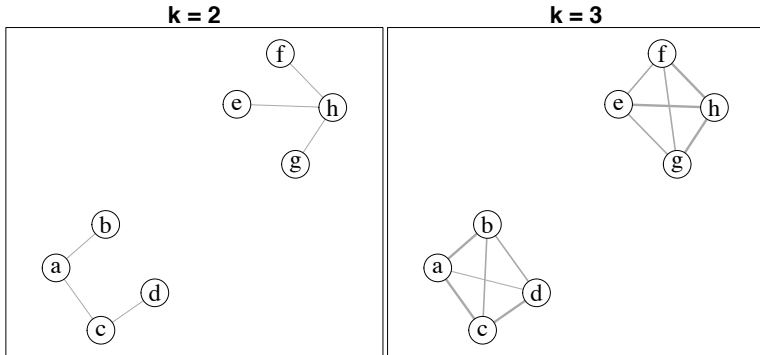


Figure 2: Constructed SNN network using example gene expression matrix with $k = 2$ (left) and $k = 3$ (right). Notice that the $k = 3$ network contains more edges, because lower k will yield a more sparse network by construction.

In short, the closer two nodes are to their shared neighbor, the more weight will be assigned to the edge that connects them. In BiSNN-Walk k is hard-coded to be $\lceil \log(n) \rceil$. This is because BiSNN-Walk contains a self-correcting scheme, so it does not require a refined selection of k , and we found that $\lceil \log(n) \rceil$ is reasonable under most scenarios.

We chose to construct SNN instead of directly using similarity matrices for two reasons. First, in SNN networks, the notion of “distance” between two nodes is established in the context of a local neighborhood instead of quantified by an absolute measure, such as the Euclidean distance. This localization of distance is especially desirable for high dimensional data, where the absolute distance measure like Euclidean distance becomes less and less useful for gauging proximity with higher dimensions [Aggarwal et al. (2001)]. Secondly, the edge weights between two nodes in an SNN network implicitly carries information about the similarity between the two neighborhoods of the two nodes, whereas in a similarity matrix, the similarity score between two cells only carries information about the two nodes themselves. Through its edge construction, SNN creates a filter that condenses informative neighborhood characteristics, which are otherwise lost in similarity matrices.

D Walktrap Clustering

Walktrap Clustering was proposed by Pascal Pons and Matthieu Latapy in [Pons and Latapy (2005)]. The method uses an agglomerative hierarchical clustering to cluster the the nodes, and the paper also suggests a method of cutting the resulting tree. Notation here will follow the paper as closely as possible.

Let $G(V, E)$ be an undirected graph with vertices V and edges E , where $|V| = n$, and $|E| = m$. Let A be the adjacency or weight matrix, and $D = \text{diag}(d_1, \dots, d_n)$ is the $n \times n$ diagonal matrix containing the corresponding degree (sum of weights if graph is weighted) of each node. $P = [P_{ij}] = \left[\frac{A_{ij}}{d_i} \right]$ be the corresponding $n \times n$ transition matrix.

The most important piece of any hierarchical clustering algorithm is the definition of distance between nodes. Here the distance between nodes is defined as

Definition 1. The distance between node i and j is

$$r_{ij}(t) = \sqrt{\sum_{k=1}^n \frac{([P^t]_{ik} - [P^t]_{jk})^2}{d_k}} \quad (2)$$

$$= \|D^{-\frac{1}{2}} [P^t]_i - D^{-\frac{1}{2}} [P^t]_j\| \quad (3)$$

where t is some predefined time.

Since

$$[P^t]_{ij} = \mathbb{P}(\text{a walk starting at node } i \text{ will end up at } j \text{ at time } t)$$

The vector $[P^t]_i$ can be thought of a visiting ‘‘profile’’ of a walk starting at i , at time t , then if i and j share many neighbors, then their visiting profile should be similar, thus the corresponding distance $r_{ij}(t)$ should be small. The $D^{-\frac{1}{2}}$ factor is just a normalizing factor that down weights the effect of nodes with large degrees, whom, by the nature of the transition matrix, will be visited more no matter the starting position .

$r_{ij}(t)$ is closely associated with the spectral properties of the transition matrix P . Let $\{\lambda_\alpha : 1 \leq \alpha \leq n\}$ and $\{v_\alpha : 1 \leq \alpha \leq n\}$ be the eigenvalues and eigenvectors of P , then

$$r_{ij}^2(t) = \sum_{\alpha=1}^n \lambda_\alpha^{2t} (v_\alpha(i) - v_\alpha(j))^2$$

where $v_\alpha(i)$ is the i^{th} element of the vector v_α .

Since $r_{ij}(t)$ is the ‘‘distance’’ between node i and j only at time t , it’s a better idea to examine at the entire history of the walk over all t , which leads to the generalized distance \hat{r}_{ij} , which defined as

Definition 2. Let $\{c_k : k = 1, \dots, \infty, c_k \geq 0 \forall k, \sum c_k = 1\}$ be a set of predefined weights. Let $\hat{P}_i = \sum_{k=1}^{\infty} c_k P_i^k$. The generalized distance

$$\hat{r}_{ij}^2 = \sum_{\alpha=1}^n f^2(\lambda_\alpha) (v_\alpha(i) - v_\alpha(j))^2 \quad (4)$$

$$= \|D^{-\frac{1}{2}} \hat{P}_i - D^{-\frac{1}{2}} \hat{P}_j\| \quad (5)$$

where $f(x) = \sum_{k=1}^{\infty} c_k x^k$ is a power series function dictated by $\{c_k\}$.

Example 1. If we consider the continuous parallel of the random walk defined by P , i.e in the continuous random walk, the probability of a walk starting in node i and ending up in node j after time t is

$$\left[e^{(P-I)t} \right]_{ij}$$

Then the associated generalized distance with this transition matrix is

$$\hat{r}_{ij}^2 = \sum_{\alpha=1}^n e^{2t(\lambda_\alpha - 1)} (v_\alpha(i) - v_\alpha(j))^2$$

with $c_k = \frac{t^k}{k!} e^{-t}$.

Since computing \hat{r}_{ij}^2 exactly require us to know all the eigenvectors, it is entirely possible when P is small, but becomes quite expensive when P is large ($O(n^3)$), so in most cases we will use the form \square and approximate \hat{P}_i . To approximate $\sum_{k=1}^{\infty} c_k P_i^k$ notice that since $\sum_j [P^k]_{ij} = 1$ and $[P^k]_{ij} \geq 0 \forall i, j$, and $\sum_k c_k = \sum_k \frac{t^k}{k!} e^{-t} = 1$, then for any $\epsilon > 0$, there exists an integer r such that $\|\sum_{k=r+1}^{\infty} c_k P_i^k\| < \epsilon$ by Cauchy-Schwartz. We can approximate \hat{P}_i with $\sum_{k=1}^r c_k P_i^k$ with some predefined r .

So far we only talked about node-to-node distance, in order for the distance to be used in a heirarchical setting, we'll need to extend this notion to cluster-cluster and cluster-node setting. Let C be a cluster, the average probability of a walk starting at any of the members in C to reach node j is

$$\hat{P}_{Cj} = \frac{1}{|C|} \sum_{i \in C} \hat{P}_{ij}$$

then the corresponding generalized distance between two clusters is

$$\hat{r}_{C_1 C_2} = \|D^{-\frac{1}{2}} \hat{P}_{C_1} - D^{-\frac{1}{2}} \hat{P}_{C_2}\|$$

Therefore, to build the tree, we will start with every node being its own cluster, call this clustering \mathcal{P}_1 . And in the next step, like the regular hierarchical clustering, we will merge two of the clusters (nodes) in \mathcal{P}_1 to obtain the next clustering \mathcal{P}_2 . For each step k clusters from the clustering \mathcal{P}_{k-1} , and all nodes will be merged into a single cluster by step $n - 1$, which will be the root of the tree.

At each step we will merge clusters by minimizing the following quantity

$$\sigma_k = \frac{1}{n} \sum_{C \in \mathcal{P}_k} \sum_{i \in C} \hat{r}_{iC}^2$$

Which is the average squared distance of a node to the cluster it belongs to. Minimizing this quantity directly at each step is computationally intensive, and requiring $O(|\mathcal{P}_k|^2)$ computation time for each k , instead we try to find, let $C_1, C_2 \in \mathcal{P}_k$, and $C_3 = C_1 \cup C_2$, then

$$\Delta\sigma(C_1, C_2) = \frac{1}{n} \left(\sum_{i \in C_3} \hat{r}_{iC_3} - \sum_{i \in C_1} \hat{r}_{iC_1} - \sum_{i \in C_2} \hat{r}_{iC_2} \right)$$

Which relates to $\hat{r}_{C_1C_2}^2$ like

$$\Delta\sigma(C_1, C_2) = \frac{1}{n} \frac{|C_1||C_2|}{|C_1| + |C_2|} \hat{r}_{C_1C_2}$$

So as long as we know $\hat{r}_{C_1C_2}$, $\Delta\sigma(C_1, C_2)$ can be calculated in linear time.

To cut the tree, we use the quantity

$$\eta_k = \frac{\Delta\sigma_k}{\Delta\sigma_{k-1}} = \frac{\sigma_{k+1} - \sigma_k}{\sigma_k - \sigma_{k-1}}$$

Intuitively, the idea is that when two very distant communities are merged, we would see a large $\Delta\sigma$, so the preferable clustering \mathcal{P}_k should contain distant clusters so that further merging of \mathcal{P}_{k+1} would greatly increase σ_k , but previous clustering \mathcal{P}_{k-1} still contain similar clusters such that \mathcal{P}_{k-1} to \mathcal{P}_k does not increase σ_k significantly, that is, we cut the tree at $k = \operatorname{argmax}_k \eta_k$.

E Adjusted Rand Index

The Rand index is developed by William M. Rand for the purpose of quantifying the agreement between two clustering results in his seminal paper “Objective criteria for the evaluation of clustering methods” [Rand (1971)]. The method assumes that the clusters do not overlap, i.e each item belongs to only one cluster. In our case, let $U = \{U_1, \dots, U_M\}$ and $V = \{V_1, \dots, V_N\}$ be two sets of clusters on cells $1, \dots, n$. Define the following quantities, as mentioned in the main text,

$$a = \text{pairs belong to the same cluster in } U \text{ as well as } V \quad (6)$$

$$b = \text{pairs belong to the same cluster in } U \text{ but different clusters in } V \quad (7)$$

$$c = \text{pairs belong to different clusters in } U \text{ but the same cluster in } V \quad (8)$$

$$d = \text{pairs belong to different clusters in } U \text{ as well as } V \quad (9)$$

a is a set of nodes, but if there is no confusion we will also use a to denote its cardinality. The Rand index of the clustering U, V is

$$RI(U, V) = \frac{a + d}{a + b + c + d} = \frac{a + d}{\binom{n}{2}}$$

The Rand Index is bounded between $[0, 1]$. Suppose we break down the cluster memberships in a different way as shown in Table 1.

	V_1	V_2	\dots	V_N	total
U_1	n_{11}	n_{12}	\dots	n_{1N}	$n_{1\cdot}$
U_2	n_{21}	n_{22}	\dots	n_{2N}	$n_{2\cdot}$
\vdots	\vdots		\ddots	\vdots	\vdots
U_M	n_{M1}	n_{M2}	\dots	n_{MN}	$n_{M\cdot}$
total	$n_{\cdot 1}$	$n_{\cdot 2}$	\dots	$n_{\cdot N}$	n

Table 1: Contingency table showing the break-down of membership assignment of node-pairs. Here n_{ij} = number of nodes that are simultaneously assigned to clusters U_i and V_j .

Table 1 allows us to easily calculate a few important quantities, e.g

$$\binom{n_{ij}}{2} = \text{total number of possible node pairs that are assigned to } U_i \text{ and } V_j$$

$$\binom{n_{i\cdot}}{2} = \text{total number of possible node pairs that are assigned to } U_i$$

$$\binom{n}{2} = \text{total number of possible node pairs}$$

Using these quantities we can calculate the probability of a node pair belonging to $a + d$:

$$\begin{aligned}
 a + d &= \left\{ \underbrace{\binom{n}{2}}_{\text{total \# pairs}} - \underbrace{\left[\sum_{i=1}^M \binom{n_{i\cdot}}{2} + \sum_{i=1}^N \binom{n_{\cdot j}}{2} - \sum_{i=1}^M \sum_{j=1}^N \binom{n_{ij}}{2} \right]}_{\text{total \# pairs clustered together in at least one of the partitions}} \right\} + \\
 &\quad \underbrace{\sum_{i=1}^M \sum_{j=1}^N \binom{n_{ij}}{2}}_{\text{total \# pairs that were clustered to the same cluster in both partitions}} \\
 &= \binom{n}{2} + 2 \sum_{i=1}^M \sum_{j=1}^N \binom{n_{ij}}{2} - \left[\sum_{i=1}^M \binom{n_{i\cdot}}{2} + \sum_{i=1}^N \binom{n_{\cdot j}}{2} \right]
 \end{aligned}$$

One issue with the Rand Index is that, suppose $T = \{T_l, l = 1 \dots L\}$ is the ground truth partition, then $RI(U, T)$ and $RI(V, T)$ are not comparable, that is, even if $RI(U, T) > RI(V, T)$ it is not necessarily the case that U is a better partition than V with respect to T because there is no consistent baseline measure. In other words, comparing $RI(U, T)$ and $RI(V, T)$ is akin to comparing realizations of $X \sim N(\mu_x, \sigma^2)$ and $Y \sim N(\mu_y, \sigma^2)$ without actually knowing what μ_x and μ_y are. This fact severely limits the usefulness of the Rand Index; therefore, Hubert and Arabie proposed to frame the problem in terms of a hypergeometric model [Hubert and Arabie (1985)], in which we assume that the number of elements in $n_{i\cdot}$'s and $n_{\cdot j}$'s are fixed, and N_{ij} 's are random variables. Then, the probability of a node pair to belong to U_i and V_j is

$$E \left[\frac{\binom{N_{ij}}{2}}{\binom{n}{2}} \right] = \frac{\binom{n_{i\cdot}}{2} \binom{n_{\cdot j}}{2}}{\binom{n}{2} \binom{n}{2}} \tag{10}$$

Thus

$$E \left[\binom{N_{ij}}{2} \right] = \frac{\binom{n_{i\cdot}}{2} \binom{n_{\cdot j}}{2}}{\binom{n}{2}}$$

Then, with some simple algebra

$$E[RI(U, V)] = \frac{E \left[\binom{n}{2} + 2 \sum_{i=1}^M \sum_{j=1}^N \binom{N_{ij}}{2} - \left[\sum_{i=1}^M \binom{n_{i\cdot}}{2} + \sum_{i=1}^N \binom{n_{\cdot j}}{2} \right] \right]}{\binom{n}{2}} \quad (11)$$

$$= 1 + 2 \sum_{i=1}^M \sum_{j=1}^N \frac{\binom{n_{i\cdot}}{2} \binom{n_{\cdot j}}{2}}{\binom{n}{2}^2} - \left[\sum_{i=1}^M \frac{\binom{n_{i\cdot}}{2}}{\binom{n}{2}} + \sum_{i=1}^N \frac{\binom{n_{\cdot j}}{2}}{\binom{n}{2}} \right] \quad (12)$$

Using the chance-corrected form of an index: $\frac{index - E[index]}{\max[index] - E[index]}$ and noting Rand Index is bounded above by 1, then

$$ARI(U, V) = \frac{RI(U, V) - E[RI(U, V)]}{1 - E[RI(U, V)]} \quad (13)$$

$$= \frac{\sum_{i=1}^M \sum_{j=1}^N \binom{N_{ij}}{2} - \frac{\sum_{i=1}^M \binom{n_{i\cdot}}{2} \sum_{j=1}^N \binom{n_{\cdot j}}{2}}{\binom{n}{2}}}{\frac{1}{2} \left[\sum_{i=1}^M \binom{n_{i\cdot}}{2} + \sum_{i=1}^N \binom{n_{\cdot j}}{2} \right] - \frac{\sum_{i=1}^M \binom{n_{i\cdot}}{2} \sum_{j=1}^N \binom{n_{\cdot j}}{2}}{\binom{n}{2}}} \quad (14)$$

F Irreducible Discovery Rate

One of the most pertinent question in high throughput sequencing is whether the signal we see in the data are real, or true positives. For instance, suppose we were to conduct a Chip-seq experiment to find binding sites (peaks) of a transcription factor, suppose we were to repeat the experiment under identical settings many times, the peaks that show up as significant across experiments would be considered “reproducible”. In practice, however, we usually an experiment is only replicated twice due to budget and time constraints, and irreducible discovery rate was introduced to quantify the “reproducibility” of the signals in the replicate experiments. Other measures of reproducibility also exist before the introduction of IDR, the most prominent of which include Spearman’s correlation and rank correlation; however, the idea that set IDR apart from its predecessors is that it makes a lot more sense to measure reproducibility using the signals that are actually reproducible; in other words, one should not use the entire experiment to measure reproducibility of replicate experiments. Also, there was a lack of measure that quantify local reproducibility, i.e using the previous Chip-seq example, it is also worthwhile to know the reproducibility of individual peaks.

The main idea of IDR is that it separates the pairs into two groups (remember, we have two replicates of every experiment, thus the data is a $n \times 2$ matrix, i.e n pairs), a reproducible group, and a non-reproducible group. Let $(x_{i,1}, x_{i,2})$ denote the pair of observations, assume $(x_{i,1}, x_{i,2}) \sim F^1(\cdot, \cdot)$ if the pair belongs to the reproducible group, $\sim F^0(\cdot, \cdot)$ otherwise. Suppose the proportion of genuine signals is π_1 and that of spurious signals is $\pi_0 = 1 - \pi_1$, then $(x_{i,1}, x_{i,2}) \sim F(\cdot, \cdot) = \pi_1 F^1(\cdot, \cdot) + \pi_0 F^0(\cdot, \cdot)$. Let $F_1(\cdot) \equiv$ marginal distribution of the first coordinate, and $F_2(\cdot)$ similarly defined.

Now let’s define the dependence structure within each group. Let $(z_{i,1}, z_{i,2}) \sim BN\left(\begin{pmatrix} \mu \\ \mu \end{pmatrix}, \begin{pmatrix} \rho\sigma^2 & \sigma^2 \\ \sigma^2 & \rho\sigma^2 \end{pmatrix}\right)$, $\mu_1 > 0, \rho > 0$ if the pair are drawn from the genuine group, otherwise $(z_{i,1}, z_{i,2}) \sim SBN$. Here ρ gauges the overall reproducibility between two experiments, and is the notion of IDR we use in our study. Let G denote the marginal distributions of $z_{i,j}$, then

$$G(\cdot) = \frac{\pi_1}{\sigma} \Phi\left(\frac{\cdot - \mu}{\sigma}\right) + \pi_0 \Phi(\cdot)$$

In our model, $(z_{i,1}, z_{i,2})$ are the unobserved latent variables that induces $(x_{i,1}, x_{i,2})$ according to the following relationship

$$x_{i,1} = F_1^{-1}(G(z_{i,1})) \tag{15}$$

with $x_{i,2}$ similarly defined. In other words, the drawing of $(z_{i,1}, z_{i,2})$ (thus G) gives $(x_{i,1}, x_{i,2})$ their dependence structure, while F dictates the actual value they will take. In the paper, this is called the copula mixture model.

According to equation $\square()$, $(z_{i,1}, z_{i,2}) = (G^{-1}(F_1(x_{i,1})), G^{-1}(F_2(x_{i,2})))$. Assume all pairs are independent and identically distributed, i.e they are all induced by their respective i.i.d $(z_{i,1}, z_{i,2})$ ’s, then the semi-parameterized likelihood function is parameterized by $\theta = (\pi_1, \mu, \rho, \sigma)$ and (F_1, F_2)

and can be written as

$$L(\theta) = \prod_{i=1}^n [\pi_0 h_0(G^{-1}(F_1(x_{i,1})), G^{-1}(F_2(x_{i,2}))) + \quad (16)$$

$$\pi_1 h_1(G^{-1}(F_1(x_{i,1})), G^{-1}(F_2(x_{i,2}))) \quad (17)$$

Where h_1 is the density of $BN\left(\begin{pmatrix} \mu_1 \\ \mu_1 \end{pmatrix}, \begin{pmatrix} \rho\sigma^2 & \sigma^2 \\ \sigma^2 & \rho\sigma^2 \end{pmatrix}\right)$ and h_0 is the density of SBN.

EM algorithm is used to fit $L(\theta)$ using the following steps []:

1. First compute the marginal empirical distribution $\hat{F}_1(x_{i,1}) = \frac{r_{i,1}}{n}$, where $r_{i,1}$ = rank of $x_{i,1}$ in experiment 1. $\hat{F}_2(x_{i,2})$ similarly defined.
2. Let $u_{i,1} \equiv \frac{n-1}{n} \hat{F}_1(x_{i,1})$ be the empirical quantile of $x_{i,1}$. The factor $\frac{n-1}{n}$ is applied to avoid unboundedness of G^{-1} at 1. Obtain $u_{i,2}$ with similar fashion
3. Initialize θ , denote it $\theta^{(0)} = (\pi_1^{(0)}, \rho^{(0)}, \mu^{(0)}, \sigma^{(0)})$
4. Compute pseudo data $z_{i,1} = G^{-1}(u_{i,1})$, and $z_{i,2}$.
5. Apply EM algorithm on the likelihood function of the augmented dataset $Y_i = (\mathbf{z}_i, K_i)$, where $\mathbf{z}_i = (z_{i,1}, z_{i,2})$ and the latent variable

$$K_i = \begin{cases} 1 & \mathbf{z}_i \in \text{reproducible group} \\ 0 & \text{o.w} \end{cases}$$

The corresponding likelihood is

$$l(\theta) \equiv \sum_{i=1}^n \{K_i (\log \pi_1 + \log h_1(\mathbf{z}_i)) + (1 - K_i) (\log \pi_0 + \log h_0(\mathbf{z}_i))\}$$

For the Expectation step, we need to find the expectation of $l(\theta)$:

$$\begin{aligned} Q(\theta|\theta^{(0)}) &= E_{K|Z, \theta^{(0)}} l(\theta) \quad (18) \\ &= \sum_{i=1}^n \left\{ E_{K|Z, \theta^{(0)}} [K_i] (\log \pi_1 + \log h_1(\mathbf{z}_i)) + \left[1 - E_{K|Z, \theta^{(0)}} (K_i) \right] (\log \pi_0 + \log h_0(\mathbf{z}_i)) \right\} \end{aligned}$$

where

$$E_{K|Z, \theta^{(0)}} [K_i] = P(K_i = 1 | \mathbf{z}_i, \theta^{(0)}) \quad (20)$$

$$= \frac{P(K_i = 1, \mathbf{z}_i | \theta^{(0)})}{P(\mathbf{z}_i | \theta^{(0)})} \quad (21)$$

$$= \frac{\pi_1^{(0)} h_1(\mathbf{z}_i)}{\pi_1^{(0)} h_1(\mathbf{z}_i) + (1 - \pi_1^{(0)}) h_0(\mathbf{z}_i)} \quad (22)$$

For the Maximization step, we need to maximize $Q(\theta|\theta^{(0)})$ which involves fairly straightforward calculus steps. After convergence, set the resulting θ as $\theta^{(1)}$.

6. If convergence criterion is not met, e.g. $\|\theta^{(0)} - \theta^{(1)}\| < \epsilon$ for some predefined ϵ , set $\theta^{(1)} \mapsto \theta^{(0)}$ and return to step 4.

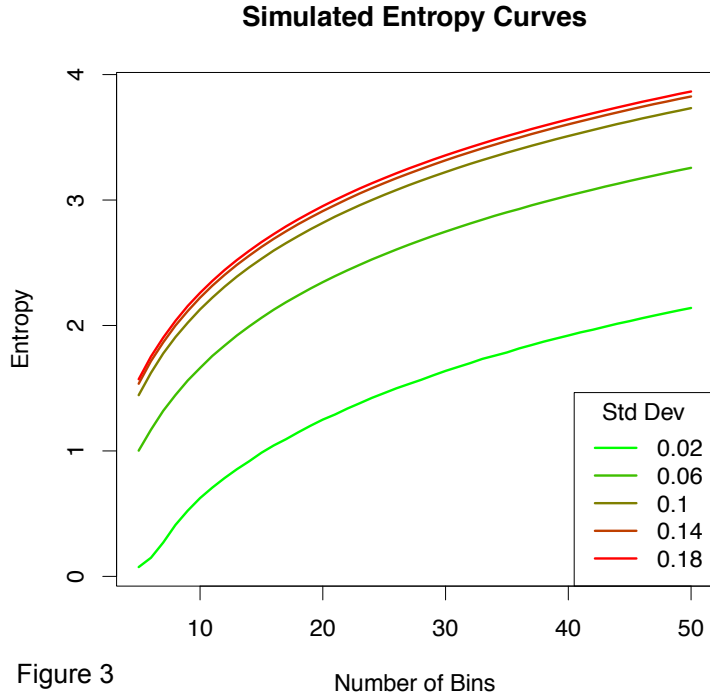


Figure 3

Figure 3: *Entropy curves from toy simulation.* We generate a similarity matrix for 10 clusters where the within-cluster similarity is randomly drawn from $\mathcal{N}(0.7, \sigma^2)$, and out-of-cluster similarity is randomly drawn from $\mathcal{N}(0.3, \sigma^2)$. Each curve represents a different value of σ , varying from 0.02 to 0.18. As one can see, entropy increase monotonically with σ across all numbers of bins.

G Exploration of Entropy Measure via Simulation and Real Data

To explore the behavior of the entropy curves, we did a toy simulation where we generated 10 clusters, with size of each cluster uniformly chosen between 4 and 20. We assume the similarity (correlation) within a cluster is $N(0.7, \sigma^2)$, and similarity between cluster is $N(0.3, \sigma^2)$, where σ varies from 0.02 to 0.18. As one can see from the resulting curves in Figure 3, the curves line up according to σ , with the curve corresponding to $\sigma = 0.18$ having the highest entropy while the curve corresponding to $\sigma = 0.02$ having the lowest. When we applied this measure to real data sets, we found it can either help yield the best result among different choices of initial similarity matrices or generate results that are quite comparable to the best result available (See Table 2 for more details).

In real data set, however, using entropy to choose initial matrix does not necessarily produce the highest quality final clusters. For instance, for Human Embryo data set, we should choose Spearman correlation as our similarity matrix (Figure 4), but Table 2 shows us that Euclidean matrix produces the best final result. This, however, should not detract from the idea of using entropy as a measure of clustering potential. First, the entropy based measure is used to select a good starting point for the algorithm, but a good starting point does not guarantee best result. Second, even though entropy-selected matrix does not yield the best result, they are often quite comparable to the best result available (Table 2). Therefore, we believe entropy-based measure of clustering-potential is a promising direction, and its theoretical properties will be explored in depth in a future paper.

Figure 4: Entropy curves of initial similarity matrices from the three scRNA-Seq data sets

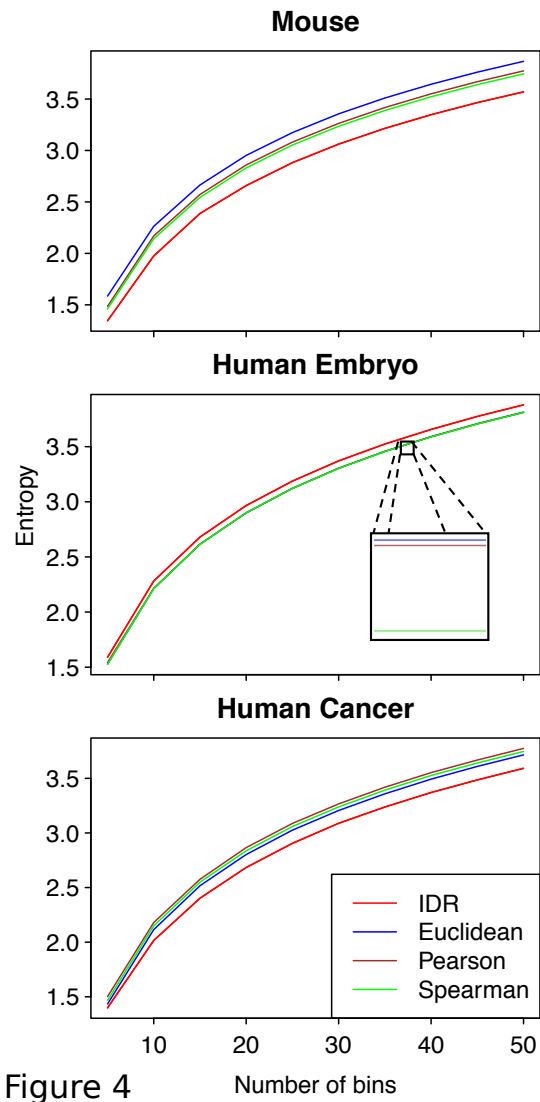


Figure 4

Table 2: Comparison between ARIs of final clusters using BiSNN-Walk with four different initial similarity matrices

	Mouse Embryo	Human Embryo	Human Cancer
IDR	0.472 (310/317)	0.600 (124/124)	0.883 (83/86)
Euclidean	0.447 (314/317)	0.798 (124/124)	0.873 (82/86)
Pearson	0.481 (297/317)	0.677 (124/124)	0.834 (70/86)
Spearman	0.467 (307/317)	0.776 (124/124)	0.880 (86/86)

Number in parentheses is (number of cells clustered / total number of cells)
 Bold font indicates the matrix with lowest entropy

H Selecting the Best Cell Cluster using Transitivity, Conductance, and Jaccard Score

H.1 Selection Heuristic

Here we present how to select the “best” cell cluster among ones identified by SNN-Walktrap using transitivity, conductance, and the Jaccard score. Please find the mathematical definition of the measures in the Definition (Section H.2) section below. We may encounter two situations here:

1. When the inner loop is first called (at the very beginning of the algorithm or by the outer loop), we need to initialize a candidate cluster. We select a single candidate from the Walktrap clusters according to two well-known network clustering measures: conductance and transitivity. Conductance measures how well separated a cluster is from rest of the network, and transitivity measures the connectedness of nodes within a cluster. To select a single one from the Walktrap clusters, we first rank them with respect to conductivity, and break ties using transitivity.
2. On subsequent iterations of the inner loop, when there already exists a candidate cluster, we want to improve upon the existing one. We calculate the Jaccard score of each Walktrap cluster with the candidate cluster as a measure of agreement, and select the Walktrap cluster of the highest agreement with the candidate cluster as the new candidate. However, if all Walktrap clusters that overlaps with the original cluster are of sizes 2 or less, then we will stop improving upon the old candidate cluster and choose a new initial cluster from the Walktrap clusters using conductance and transitivity as described previously. The rationale behind this heuristic is that Walktrap will sometimes return clusters of the same transitivity and conductance (e.g isolated perfect cliques), so the Jaccard measure serves as another tie breaker as well as ensuring a sense of continuity among the iterations’ candidate clusters.

H.2 Mathematical Definition of Transitivity and Conductance

Let $G(V, E)$ denote a network, with $V = \{v_i : i = 1, \dots, n\}$ denoting a set of nodes and $E = \{e_{i,j} \in \mathbb{R}^+ : i, j = 1, \dots, n\}$ denote a set of edges. Let C denote a cluster, and let $S(C) = \{e_{i,j} : v_i \in C, v_j \notin C\}$ be the edges that are connected to outside the cluster, and $I(C) = \{e_{i,j} : v_i, v_j \in C\}$ be the edges

that are connected within the cluster, then conductance is defined as

$$\begin{aligned} \text{conductance}(C) &= \frac{|S(C)|}{\min\{|I(C)|, |I(V \setminus C)|\}} \\ &= \frac{\text{\# of edges connected to other clusters}}{\min\{\text{\# of edges in the cluster}, \text{\# of edges outside the cluster}\}} \end{aligned}$$

Conductance is bounded below by 0, where zero-conductance implies cluster C is isolated from rest of the network, i.e lower conductivity indicates a better isolated cluster. Note if the cluster is the entire node set, $C = V$, then we force $\text{conductance}(C) = \text{conductance}(V) = 0$.

Transitivity, or clustering coefficient, is easier described in words

$$\begin{aligned} \text{transitivity}(C) &= \frac{3 \times \text{\# of triangles in the cluster}}{\text{\# of connected triplet of vertices, or V shapes}} \\ &= \frac{\text{\# of triangles in the cluster}}{\text{\# of total possible triangles if all nodes are connected}} \end{aligned}$$

Transitivity is bounded between $[0,1]$, where 0 means all members of the cluster are isolated points, and 1 indicates a perfect clique.

I scRNA-Seq Data Sets

Mouse Embryonic Cells. The size of the gene expression matrix is 41,128 genes \times 317 cells. All mouse embryonic cells are crossed between CAST female mated to C57 male cell lines. Embryonic cells were collected during 10 developmental stages from zygote to blastocysts. Somatic cells (liver and fibroblast) are also collected. For consistency’s sake, somatic cells are obtained from either C57 x CAST or CAST x C57 offsprings. See Main Paper: Table 2 for details. Cells are sequence using either Smart-seq or Smart-seq2 technology [Deng et al. (2014)]. This data set will be here on referred to as “mouse”.

Human Embryonic Cells. The size of the gene expression matrix is 60,483 genes \times 124 cells. In this work, Yan et al [Yan et al. (2013)] investigates the genetic markers involved in the derivation of human embryonic stem cells (hESC) by examining the development of human embryo through its developmental stages. 124 embryonic cells were obtained from in-vitro fertilization patients. Patients are aged controlled to be within 25-35 years old with mean age of 30. The cells were categorized into into 8 categories, details show in Main Paper: Table 3. The sequencing technology introduced and used in this study is called “single cell RNA-Seq”. This dataset will here on be referred to as “human embryo”.

Human Cancer Cells The size of the gene expression matrix is 60,483 genes \times 86 cells. In this work the sequencing technique “Smart-seq” was introduced by Ramsköld et al. [Ramsköld et al. (2012)] and applied to various low frequency cancer cells such as circulating tumor cells or somatic cells that are difficult to obtain in mass quantities, such as brain cells. A total of 86 human cells are reported by the study¹, details about cell species are listed in Main Paper: Table 4. Though the study contains both cancer cells and various somatic cells, we will dub this data set “human cancer” for short.

¹The study also contained 2 white blood cells, but the sequencing quality was extremely poor, and is confirmed by the author to be unusable through email correspondence. The study also contained reports on mouse cells, but are not used in our data because they cannot be compared directly to human cells.

J Comparison with Selected Algorithms

J.1 Introduction

We applied four popular biclustering algorithm to compare our method with, they are Plaid [Lazzeroni and Owen (2002)], Cheng & Church [Cheng and Church (2000)], Xmotifs [Voggenliter et al. (2012)], and BiMax [Madeira and Oliveira (2004)]. We also applied a recently published clustering algorithm, GiniClust [Jiang et al. (2016)], designed to handle scRNA-Seq data.

Plaid, Cheng & Church. Plaid and CC are frequently used as benchmarks. Both Plaid and CC assumes that gene expression can be expressed in an additive fashion of the form $\mu + a_i + b_j$, where μ is the background constant, a_i/b_j are row/column specific constants, respectively. The μ , a_i , and b_j are treated as parameters in the Plaid model whereas they are set as row, column and overall means in CC. The chosen bicluster would have expression values that fall most consistent around a_i , b_j , and μ . For Plaid model, after a bicluster is found, its values are then subtracted, and the residual expression matrix is used to find the subsequent biclusters. For CC, after a bicluster is found, random gene expression values are used to replace that of the true bicluster, and the resulting scrambled “gene expression matrix” is then used for subsequent runs.

BiMax. Devised as a benchmark algorithm, BiMax is the simplest of the benchmarks. It operates on binary matrices, and uses a divide and conquer approach to find all maximal completely bipartite graphs (maximal submatrices containing all 1s).

Xmotifs. For any random cell cluster C we define a gene's expression on the cells in C as interesting or not interesting, which is called a gene's *state* on C . We call a gene-cell bicluster (G, C) an Xmotif if for every cell in C , all of the genes in G are in the same state. Further more, an Xmotif is maximal if every genes not in the Xmotif has less than β (some user defined percentage) states in common with genes in G . The algorithm randomly generates a user-defined number of seed-Xmotifs and attempt to grow them into maximal Xmotifs.

GiniClust. We first use a modified Gini index to isolate genes of interest. The submatrix with rows being the selected genes and columns being the cells are then passed to the clustering algorithm DBSCAN to obtain cell clusters. Therefore the algorithm will return cell clusters, but only one cluster of genes, so it's in fact more an algorithm for clustering than biclustering.

J.2 Results

From Figures 5, 6, and 7, Plaid obtained the best cell clusters compared to the other three biclustering algorithms; however, it did not have the sensitivity to obtain clusters nearly as homogenous as the ones BiSNN-Walk discovered.

The failure of the four biclustering algorithms stems from requiring inputs, i.e a gene exhibits high expression and one exhibiting zero expression is equally informative. For RNAseq data, for a given cell the majority of the genes will exhibit zero expression, the biclustering algorithms in consideration thus were not designed to process this kind of input. Another failure point for these algorithm is the fact that they all consider both overlapping cell and gene clusters, though this formulation is more general, it also works to the disadvantage to the algorithms because they cannot take advantage of the simpler cluster structure of the RNAseq data. We'd be remiss if we did not

mention that the algorithm may need expert tuning to achieve maximum performance.

From Figure 8, we see that BiSNN-Walk clusters are cleaner. The reason, as mentioned in the main paper, is because GiniClust is designed to find smaller tight-knit clusters of rare cancer cells rather than general purpose biclustering.

References

- Aggarwal, C. C., Hinneburg, A. and Keim, D. A. 2001., *On the surprising behavior of distance metrics in high dimensional space*, Springer.
- Cheng, Y. and Church, G. M. 2000., Biclustering of expression data., *in* ‘Ismb’, Vol. 8, pp. 93–103.
- Deng, Q., Ramsköld, D., Reinius, B. and Sandberg, R. 2014., ‘Single-cell rna-seq reveals dynamic, random monoallelic gene expression in mammalian cells’, *Science* **343**(6167), 193–196.
- Hubert, L. and Arabie, P. 1985., ‘Comparing partitions’, *Journal of classification* **2**(1), 193–218.
- Jiang, L., Chen, H., Pinello, L. and Yuan, G.-C. 2016., ‘Giniclust: detecting rare cell types from single-cell gene expression data with gini index’, *Genome Biology* **17**(1), 144.
- Lazzeroni, L. and Owen, A. 2002., ‘Plaid models for gene expression data’, *Statistica sinica* pp. 61–86.
- Madeira, S. C. and Oliveira, A. L. 2004., ‘Biclustering algorithms for biological data analysis: a survey’, *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* **1**(1), 24–45.
- Pons, P. and Latapy, M. 2005., Computing communities in large networks using random walks, *in* ‘Computer and Information Sciences-ISCIS 2005’, Springer, pp. 284–293.
- Ramsköld, D., Luo, S., Wang, Y.-C., Li, R., Deng, Q., Faridani, O. R., Daniels, G. A., Khrebtukova, I., Loring, J. F., Laurent, L. C. et al. 2012., ‘Full-length mrna-seq from single-cell levels of rna and individual circulating tumor cells’, *Nature biotechnology* **30**(8), 777–782.
- Rand, W. M. 1971., ‘Objective criteria for the evaluation of clustering methods’, *Journal of the American Statistical association* **66**(336), 846–850.
- Voggenreiter, O., Bleuler, S., Gruissem, W. et al. 2012., ‘Exact biclustering algorithm for the analysis of large gene expression data sets.’, *BMC Bioinformatics* **13**(S-18), A10.
- Xu, C. and Su, Z. 2015., ‘Identification of cell types from single-cell transcriptomes using a novel clustering method’, *Bioinformatics* p. btv088.
- Yan, L., Yang, M., Guo, H., Yang, L., Wu, J., Li, R., Liu, P., Lian, Y., Zheng, X., Yan, J. et al. 2013., ‘Single-cell rna-seq profiling of human preimplantation embryos and embryonic stem cells’, *Nature structural & molecular biology* **20**(9), 1131–1139.

Figure 5: Cell-clusters found by selected biclustering algorithm compared to ground truth for the mouse dataset. *x*-axis are the cell-clusters found by indicated algorithm, ordered roughly by developmental stage. *y*-axis is ground truth ordered by developmental stage. The value in each grid represents the percentage of ground truth cluster that is in each cell cluster.

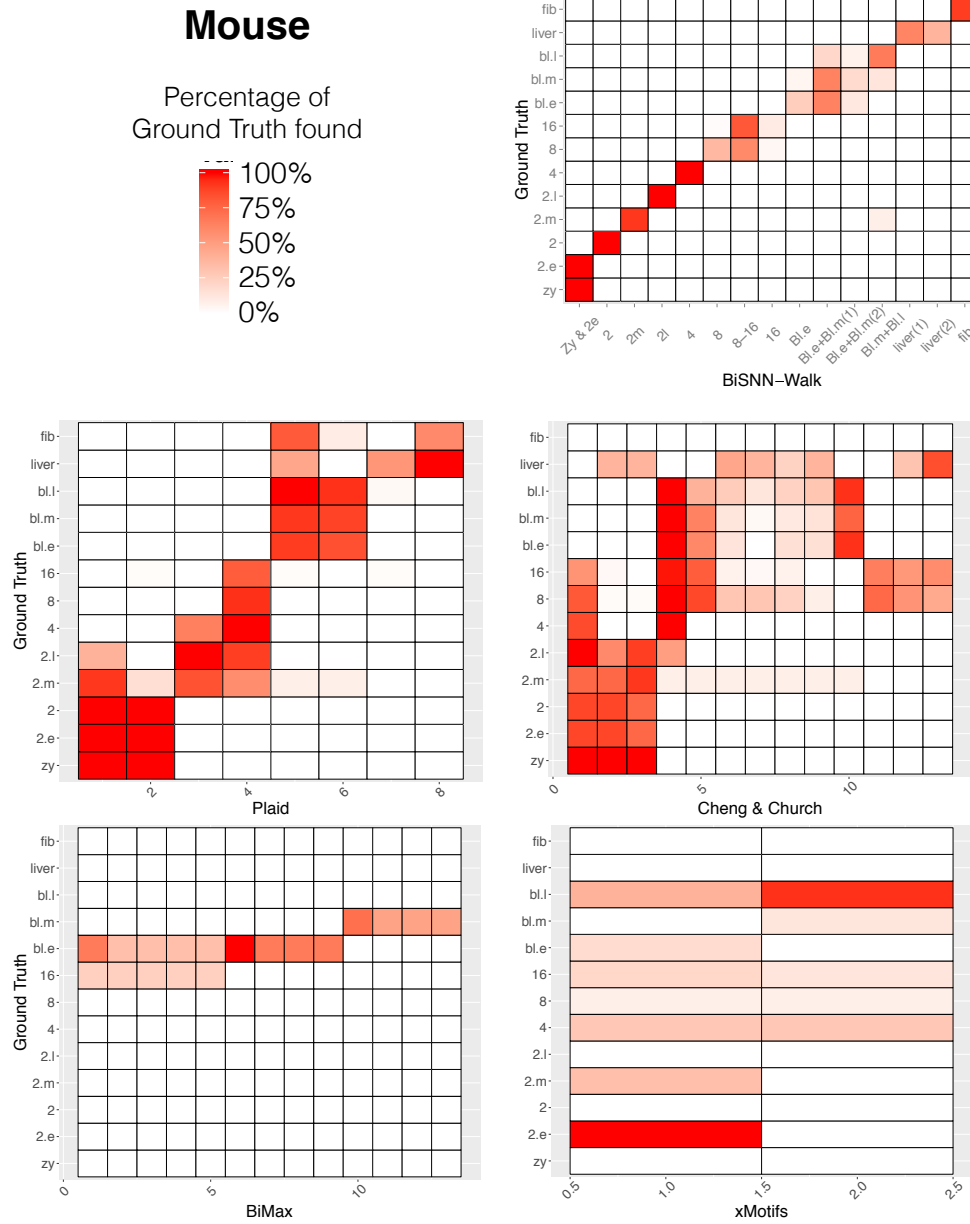


Figure 5

Figure 6: Cell-clusters found by selected biclustering algorithm compared to ground truth for the human embryo dataset. *x*-axis are the cell-clusters found by indicated algorithm, ordered roughly by developmental stage. *y*-axis is ground truth ordered by developmental stage. The value in each grid represents the percentage of ground truth cluster that is in each cell cluster.

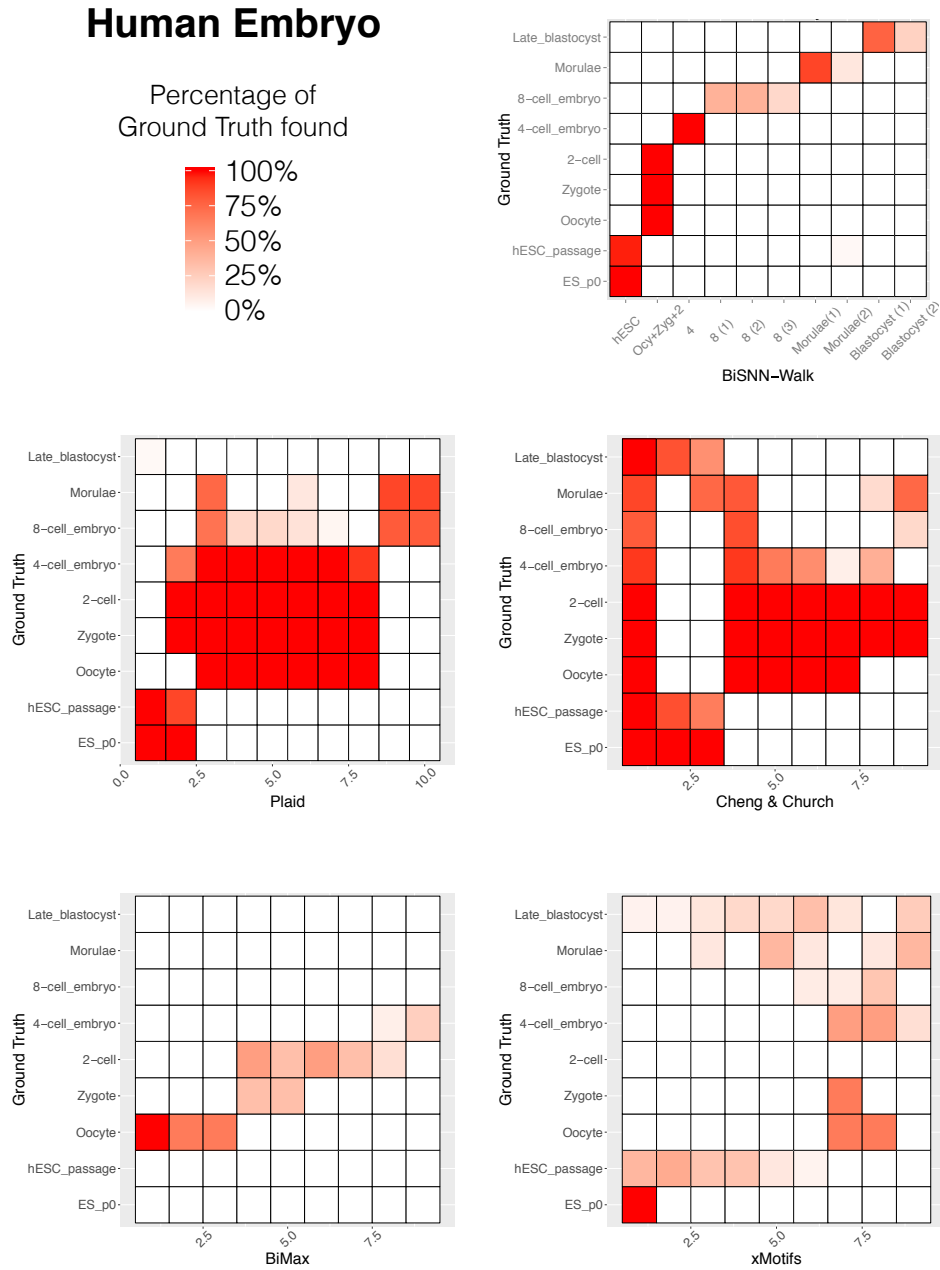


Figure 6

Figure 7: Cell-clusters found by selected biclustering algorithm compared to ground truth for the human cancer dataset. *x*-axis are the cell-clusters found by indicated algorithm, *y*-axis is ground truth. The value in each grid represents the percentage of ground truth cluster that is in each cell cluster.

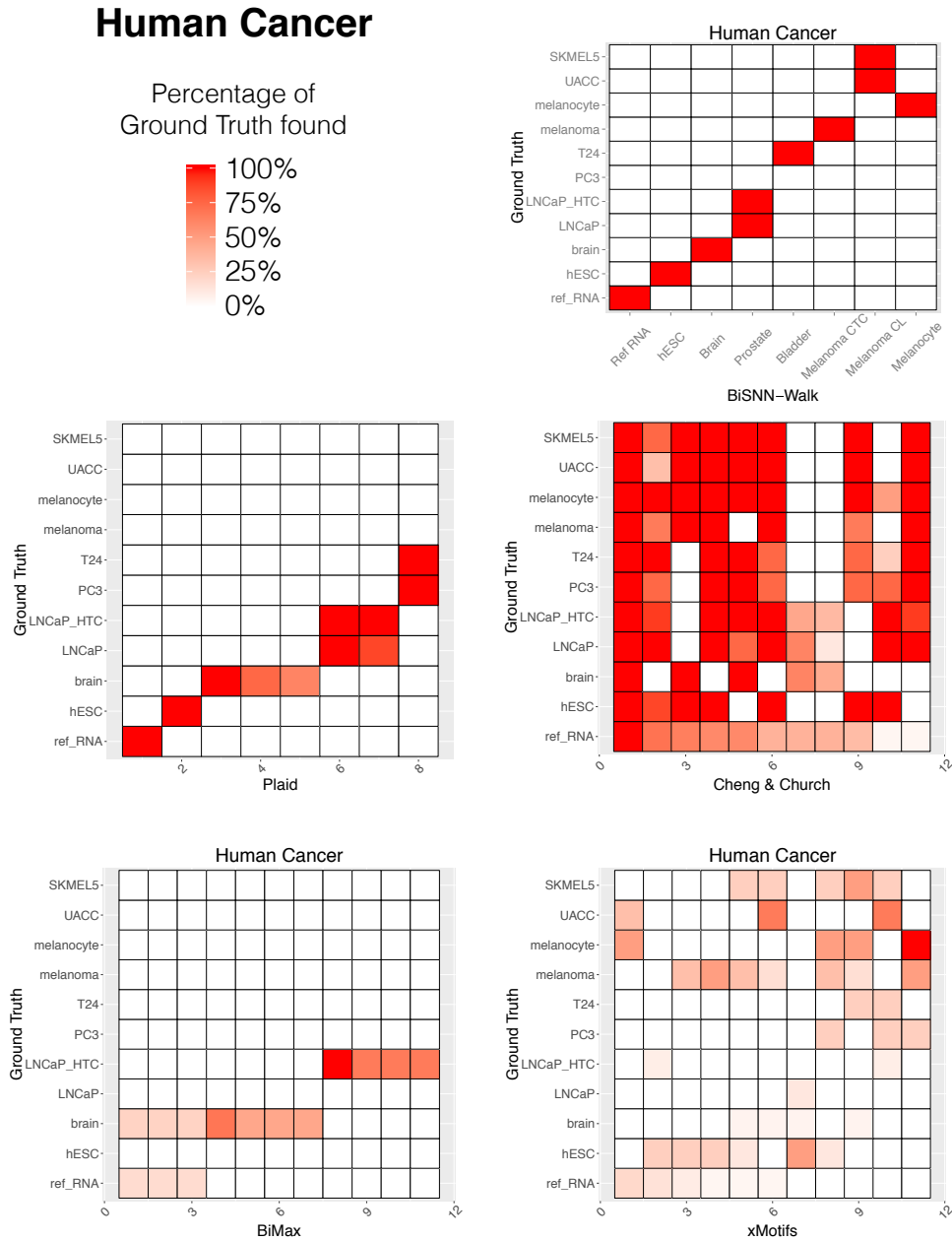


Figure 7

Figure 8: Heatmap of *BiSNN-Walk* (left) and *GiniClust* (right) cell clusters plotted against ground truth. *x*-axis are the cell-clusters found by indicated algorithm, *y*-axis is ground truth. The value in each grid represents the percentage of ground truth cluster that is in each cell cluster.

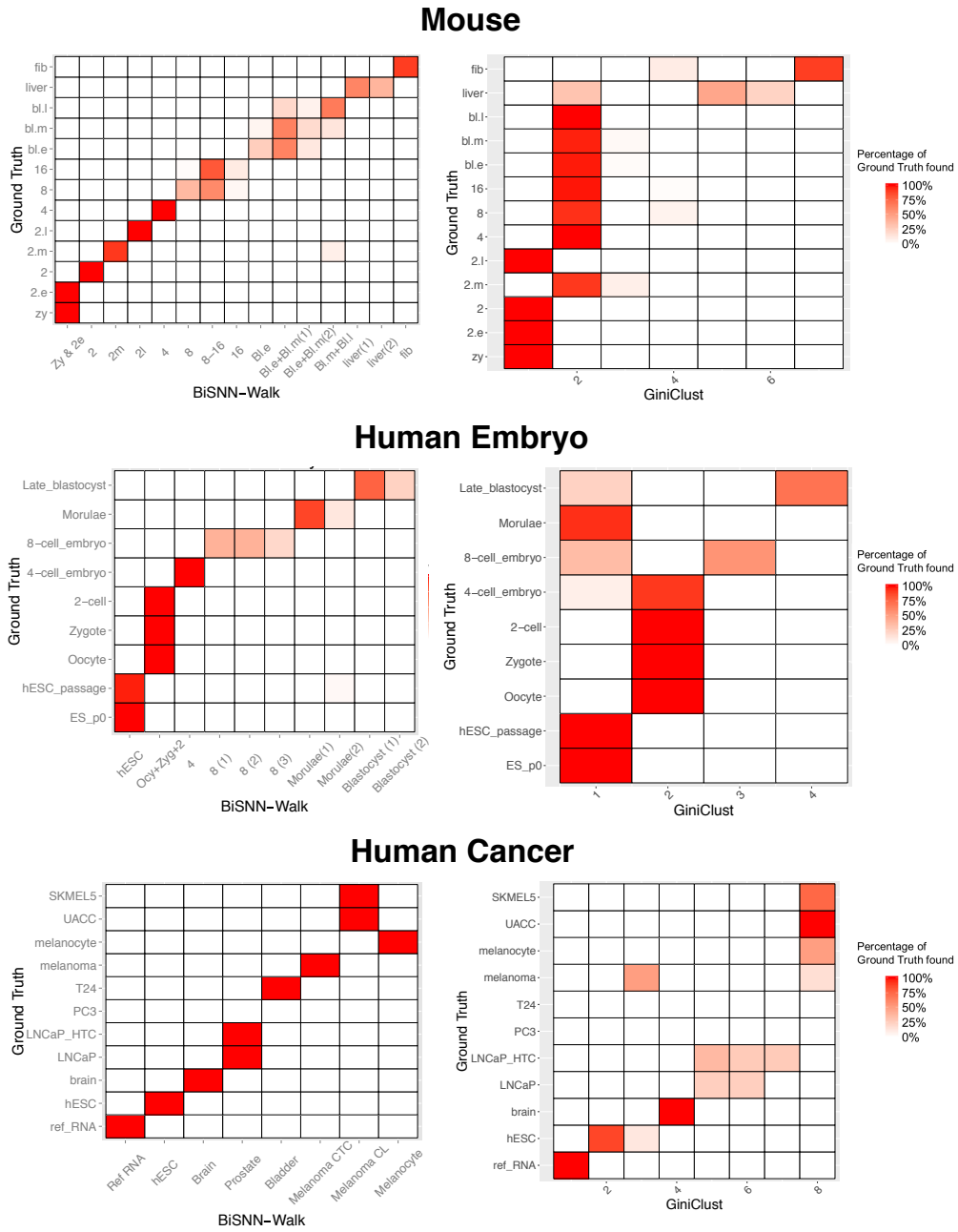


Figure 8