A Scalable Approach for General Correlation Clustering

Yubo Wang, Linli Xu, Yucheng Chen, and Hao Wang

School of Computer Science and Technology,
University of Science and Technology of China, Anhui, China
linlixu@ustc.edu.cn,
{wybang,ycchen,xdwangh}@mail.ustc.edu.cn

Abstract. We focus on the problem of correlation clustering, which is to partition data points into clusters so that the repulsion within one cluster and the attraction between clusters could be as small as possible without predefining the number of clusters k. Finding the optimal solution to the problem is proven to be NP-hard, and various algorithms have been proposed to solve the problem approximately. Unfortunately, most of them are incapable of handling large-scale data. In this paper, we relax the problem by decoupling the affinity matrix and cluster indicator matrix, and propose a pseudo-EM optimization method to improve the scalability. Experimental results on synthetic data and real world problems including image segmentation and community detection show that our technique achieves state of the art performance in terms of both accuracy and scalability.

Keywords: Correlation clustering, Unsupervised learning, Large scale, Pseudo-EM algorithm

1 Introduction

Clustering is one of the most fundamental problems in machine learning, with the goal to partition data points into groups such that the points within clusters are more similar to each other than those in different clusters. The clustering problem has received a significant amount of attention during the past few decades, and numerous methods have been proposed to solve it. However, most of them need the number of clusters k as a priori. *Correlation Clustering* [1] makes an exception, which is able to select k automatically. Moreover, this "model selection" property can be theoretically justified with a probabilistic interpretation [2], and theoretical analysis has been conducted for correlation clustering with error bounds derived [3].

Correlation clustering is a graph-based problem, where vertices correspond to the data points, and each edge (u,v) is labeled either "+" or "-" depending on whether vertices u and v are similar or not. Given this complete binary affinity graph, the task of correlation clustering is to minimize the "-" edges (repulsion) within clusters and "+" edges (attraction) between clusters, which is also known as minimizing disagreements. An equivalent optimization problem is to maximize agreements — maximize the "+" edges within clusters and "-" edges between clusters. The correlation clustering problem is proven to be NP-complete [1], and the majority of efforts are then devoted to solving it approximately [1, 4–7]. Among them, convex continuous relaxation is

frequently applied. A linear programming (LP) formulation in [4] results in a factor 4-approximation algorithm for minimizing disagreements. For maximizing agreements, several relaxations based on semi-definite programming (SDP) are achieved [4, 6]. To make the problem more flexible, [4, 5] extend the binary graphs to general weighted graphs, which contain both positive and negative edges.

Despite the large amount of theoretical analysis conducted on correlation clustering, most of the existing algorithms are impractical for real-world applications which are relatively large-scale [8, 9]. For example, there are $O(n^3)$ constraints in the LP relaxation for minimizing disagreements, while the SDP formulation with $O(n^2)$ variables for maximizing agreements is known to be computationally expensive and hard to scale up. Some recent work has focused on the computational issue and tried to address it. In [10], a more effective relaxation is proposed by exploiting the special problem domain. Discrete energy minimization algorithms are adopted in [2] to scale up the computational procedure. Although these approaches do improve the efficiency of correlation clustering, they are still insufficient for real-world problems.

In this paper, we reformulate correlation clustering with a new perspective of decoupling the affinity matrix and cluster indicator matrix. A pseudo-EM optimization method is proposed by relaxing the new formulation. Beyond that, to further improve the performance, we adopt online updates and extend the algorithm to adapt to sparse data by appending a sparsity factor. Experiments are performed on synthetic data and practical tasks including pixel-level image segmentation and community detection in real information networks, and convincing results are achieved to demonstrate the advantages of our proposed technique in terms of both accuracy and scalability.

The remainder of the paper is organized as follows. In the next section we give a brief introduction to the correlation clustering problem. In Section 3 we show how to reformulate and solve it with effective alternating minimization routine. Experimental results are presented to demonstrate the effectiveness of the proposed algorithms in Section 4 and in Section 5 we conclude the paper with possible directions for future work.

2 Correlation Clustering

Correlation clustering is defined on a complete graph $G = (V, E)$, with n vertices corresponding to the data points to be clustered and an edge between every pair of nodes. Each edge is assigned with a label $e(u, v) \in \{+, -\}$ where $e(u, v) = +$ if u and v are similar, $e(u, v) = -$ otherwise. In this paper, we will focus on the minimizing disagreements objective of correlation clustering.

Assuming cluster assignments can be represented with natural numbers, the goal of correlation clustering is to find a cluster assignment $\mathcal{C} : V \to \mathbb{N}$ by solving the following problem: $\min_{\mathcal{C}} \sum_{e(u,v)=+} 1[\mathcal{C}(u) \neq \mathcal{C}(v)] + \sum_{e(u,v)=-} 1[\mathcal{C}(u) = \mathcal{C}(v)]$. One can further extend the complete graph with binary affinity to a general graph. This graph can be described with an affinity matrix $W \in \mathbb{R}^{n \times n}$:

$$W \begin{cases} > 0 : u \text{ and } v \text{ attract each other by } |W_{uv}| \\ < 0 : u \text{ and } v \text{ repel each other by } |W_{uv}| \\ = 0 : \text{the relation between } u \text{ and } v \text{ is uncertain} \end{cases},$$

and the clustering objective for the general graph can be written as

$$\min_{\mathcal{C}} \sum_{W_{uv}>0} 1[\mathcal{C}(u) \neq \mathcal{C}(v)]W_{uv} - \sum_{W_{uv}<0} 1[\mathcal{C}(u) = \mathcal{C}(v)]W_{uv} \ .$$

By introducing a matrix D in which $D_{uv} = 1$ if u and v are in the same cluster and $D_{uv} = -1$ otherwise, we can notice that D encodes an equivalence relation, namely that it is transitive, reflexive and symmetric. Thus the minimizing disagreements problem can be rewritten as

$$\begin{aligned} \min_{D} \quad & -\Big(\sum_{\substack{W_{uv}>0 \\ D_{uv}<0}} W_{uv}D_{uv} + \sum_{\substack{W_{uv}<0 \\ D_{uv}>0}} W_{uv}D_{uv}\Big) \\ \text{s.t.} \quad & D_{uv} \in \{-1, 1\}, \forall u, v; \ D_{uu} = 1, \forall u; \\ & D_{uv} = D_{vu}, \forall u, v; \ D_{uv} + D_{vs} \leq D_{us} + 1, \forall u, v, s \end{aligned} \ . \tag{1}$$

One should notice that any feasible D matrix in (1) corresponds to an equivalence relation, and therefore it is straightforward to recover a clustering from the solution to (1).

Correlation Clustering Optimization Solving (1) exactly is NP-complete [1]. A natural way out is then to relax the hard equivalence relation constraints on D. Actually, simply by relaxing the discrete constraints $D \in \{-1, 1\}^{n\times n}$ to be continuous, the problem can be reformulated as a linear program with $O(n^3)$ linear constraints [4, 5]. Due to the cubic number of constraints, the time complexity of the LP formulation grows rapidly with the problem size.

Another relevant piece of work is to start with maximizing agreements problem with similar objective and constraints, and apply semi-definite relaxation to a linear transformation of the D variable. As a consequence, a semi-definite optimization problem with an $n \times n$ matrix variable and $O(n^2)$ linear constraints can be formulated. As a convex optimization problem, it can be solved with polynomial time. However the time complexity of solving an SDP with a matrix variable of size p and q constraints is up to $O(q^2p^{2.5})$ [11], which is prohibitive for even medium-size data.

From the discussion above, we can conclude that although convex relaxations have the nice property that global optimum can be found for the relaxed problems in polynomial time, unfortunately they are not practical for large-scale problems.

Another possible direction is to trade convexity for scalability. Recently, [2] takes a new perspective and treats correlation clustering optimization as a special discrete energy minimization problem without unary terms. Based on techniques in energy minimization [12, 13], several algorithms are proposed including *Expand-and-Explore*, *Swap-and-Explore* and *Adaptive-label ICM*. Compared to the continuous convex relaxations discussed above, significant improvements in scalability are achieved in this framework, which are chosen as the rival algorithms in our experiments.

3 Pseudo-EM Algorithm

Instead of the indirect routine of solving correlation clustering by first computing the relaxed cluster equivalence relation matrix D and then recovering the cluster assign-

ments based on D, we take a more intuitive and straightforward perspective, which is to fixate on the cluster label assignments directly.

We first define a *clustering indicator matrix* L which describes the cluster assignments of the vertices. Specifically, $L_{iu} = 1$ means that vertex u is in cluster i, $L_{iu} = -1$ otherwise. L encodes a valid clustering if it satisfies the following: a data point belongs to one and only one cluster; each cluster contains at least one data point. That is,

$$\mathcal{Q} : \{L \in \{-1, 1\}^{k\times n}; \sum_{i=1}^{k} L_{iu} = 2 - k, \forall u; \sum_{u=1}^{n} L_{iu} > -n, \forall i\} \ , \tag{2}$$

where k is the number of clusters and not predefined in correlation clustering.

Proposition 1. *The correlation indicator vector $D_{u,:}$ is the same as the cluster indicator vector $L_{\mathcal{C}(u),:}$ for any vertex u, where $\mathcal{C}(u)$ denotes the cluster assignment of u.*

Based on Proposition 1, it is possible to replace D with L in (1) and solve for L directly. Moreover, W and L can be treated as general variables without any inherent physical meaning in the new objective. As a result, a new reformulation and corresponding relaxation by decoupling W and L are derived with a two-step effective alternating optimization method, which is similar to expectation-maximization (EM) algorithm: compute a latent variable $\mathcal{C}$ based on L in the first step and optimize L according to $\mathcal{C}$ in the following step.

3.1 The Basic Pseudo-EM Routine

Consider the correlation clustering problem in a general graph (1). According to Proposition 1, the following relations are true:

$$\begin{aligned}
&D_{uv}, \forall u, v \quad &\Longleftrightarrow \quad &L_{\mathcal{C}(u)v}, \forall \mathcal{C}(u) \in \{1, 2, ..., k\}, \forall v;\\
&D_{uu} = 1, \forall u \quad &\Longleftrightarrow \quad &L_{\mathcal{C}(u)u} = 1, \forall u;\\
&\{D_{uu} = 1, \forall u;\ D_{uv} = D_{vu}, \forall u, v;\ D_{uv} + D_{vs} \leq D_{us} + 1, \forall u, v, s\} \Longleftrightarrow \quad &\mathcal{Q} \ .
\end{aligned}$$

Thus we can replace D with L according to the equivalence mentioned above and rewrite (1) as a summation of loss produced by each row of W and L, which can be expressed as

$$\begin{aligned}
\min_{L,\mathcal{C},k} \quad & -\sum_{u} \sum_{v \atop W_{uv}L_{\mathcal{C}(u)v}<0} W_{uv} L_{\mathcal{C}(u)v} \\
\text{s.t.} \quad & \mathcal{C}(u) \in \{1, 2, ..., k\}, \forall u;\ \mathcal{Q};\ L_{\mathcal{C}(u)u} = 1, \forall u
\end{aligned} \ . \tag{3}$$

Notice in the optimization problem above, apart from the variables including the cluster indicator matrix L and the number of clusters k, we introduce an auxiliary variable $\mathcal{C}$ which corresponds to the cluster assignment. These variables are coupled with each other through the constraints. Despite the redundancy of the variables, we benefit from treating them separately in the optimization procedure as we will see shortly.

The reformulated problem is still computationally hard to solve. However, we can first relax the constraint $L_{\mathcal{C}(u)u} = 1$ (which means u is in cluster $\mathcal{C}(u)$ physically) and fix L, leaving $\mathcal{C}$ the only variable to be optimized. In this way, we can minimize the objective in an iterative manner: assume at iteration t, we have a feasible candidate for L: $L^t = [\ell_1, \ell_2, ..., \ell_k]$. By fixing L^t, we can optimize over $\mathcal{C}$, where the best $\mathcal{C}^{t+1}(u)$ for vertex u can be simply computed by enumerating all possible class assignments for u, and solve the following optimization problem:

$$\mathcal{C}^{t+1}(u)^* = \underset{\mathcal{C}(u)\in\{1,2,...,k\}}{\arg\min} \sum_{v, W_{uv}L^t_{\mathcal{C}(u)v}<0} -W_{uv}L^t_{\mathcal{C}(u)v} \ . \tag{4}$$

Once the cluster assignment for all the vertices $\mathcal{C}$ is completed, it is straightforward to update L accordingly: we first set k to the number of unique clusters in $\mathcal{C}$ and reassign $\mathcal{C}$ to take values from $\{1, ..., k\}$. L can then be updated by starting from a $k \times n$ matrix of all -1's and setting $L_{\mathcal{C}(u)u} = 1$. This optimization routine works like expectation-maximization (EM) algorithm in the sense that it computes the latent variable $\mathcal{C}$ in the first step and optimizes L in the second step, therefore we call it pseudo-EM. Notice that it is possible some clusters contain no vertices at some iteration, and they would disappear after the iteration, which makes the number of clusters be selected to adapt to a lower loss. To improve the convergence rate and optimization quality, this procedure can be conducted in an online way, that means when deciding the cluster assignments for new vertices, the existing assignments are already in effect. This iterative procedure is shown in Algorithm 1.

Given a candidate for L, Algorithm 1 chooses the best cluster assignment for every u in each iteration to minimize the objective, and therefore the loss will decrease until convergence. Due to the property of selecting k automatically in Algorithm 1, there is no need to predefine the number of clusters. As we can observe from (4), vertices with similar affinity vectors will be likely in the same cluster. This implies that although we remove the constraint $L_{\mathcal{C}(u)u} = 1$, which is equivalent to the strong equivalence relation encoded in D, the nature of correlation clustering is preserved due to the intuitive optimization procedure.

Theorem 1. *Algorithm 1 is guaranteed to converge.*

Proof. First notice that in each iteration $L^t = [\ell_1, \ell_2, ..., \ell_k]$ corresponds to a clustering of all the vertices $\mathcal{C}^t$, and the optimization problem (4) is equivalent to

$$\mathcal{C}^{t+1}(u)^* = \underset{\mathcal{C}(u)\in\{1,2,...,k\}}{\arg\min} \sum_{W_{uv}>0} 1[\mathcal{C}(u) \neq \mathcal{C}^t(v)]W_{uv} - \sum_{W_{uv}<0} 1[\mathcal{C}(u) = \mathcal{C}^t(v)]W_{uv} \ ,$$

where the objective can be treated as a measurement of diversity between two cluster assignments $\mathcal{C}$ and $\mathcal{C}^t$. More specifically, define a function

$$f(\mathcal{C}_1, \mathcal{C}_2) = \sum_u \left[\sum_{W_{uv}>0} 1[\mathcal{C}_1(u) \neq \mathcal{C}_2(v)]W_{uv} - \sum_{W_{uv}<0} 1[\mathcal{C}_1(u) = \mathcal{C}_2(v)]W_{uv} \right] \ ,$$

it is easy to prove that f is non-negative: $f(\mathcal{C}_1, \mathcal{C}_2) \geq 0$ and f is symmetric: $f(\mathcal{C}_1, \mathcal{C}_2) = f(\mathcal{C}_2, \mathcal{C}_1)$. By the end of each iteration, the clustering of all the vertices is updated to

Algorithm 1 Basic Pseudo-EM Routine

Input : Affinity matrix $W^{n\times n}$, initial L^0, `Online`
Output: Cluster assignments of all the vertices

$t = 0$;
repeat
  for $u \in \{1, ..., n\}$ **do**
    compute optimal $\mathcal{C}^{t+1}(u)$ according to (4);
    if `Online` == `True` **then**
      $L^t(\mathcal{C}^t(u), u) = -1$;$L^t(\mathcal{C}^{t+1}(u), u) = 1$;
    end if
  end for
  Update L^{t+1} determined by $\mathcal{C}^{t+1}$;
  $t = t + 1$;
until the partition determined by $\mathcal{C}$ does not change
return $\mathcal{C}$

$\mathcal{C}^{t+1} = \arg\min_{\mathcal{C}} f(\mathcal{C}, \mathcal{C}^t)$, therefore, we have $0 \leq f(\mathcal{C}^{t+2}, \mathcal{C}^{t+1}) = \min_{\mathcal{C}} f(\mathcal{C}, \mathcal{C}^{t+1}) \leq f(\mathcal{C}^t, \mathcal{C}^{t+1}) = f(\mathcal{C}^{t+1}, \mathcal{C}^t)$, which guarantees that the function value of f or the summation of the objective value in (4) monotonically decreases while being lower-bounded by 0. As a consequence, Algorithm 1 is guaranteed to converge. □

3.2 Discussion

In the above, we are motivated by solving for the cluster label assignments directly, and propose the basic pseudo-EM routine for correlation clustering optimization. However, the proposed algorithm works by starting with an initial L. In this section we will introduce how to initialize L. Apart from that, we will also look into the sparsity issue in data set. At last, we will analyse the computational complexity.

In principle, any L which satisfies (2) is legal. An example is to initialize L with $2I^{n\times n} - 1$. On one hand, it is not difficult to find that the number of clusters decreases along with iterations in our method, therefore one may want to set the number of rows of initial L (i.e. the initial number of clusters) to a relatively large value. On the other hand, as the number of rows of initial L grows, the time complexity of the algorithm increases and the speed of convergence decreases. Here we describe a heuristic initialization of L based on the positive degree of vertices. First, all vertices are sorted in a list by the positive degree in descending order. Starting from the first vertex u in the list, a cluster indicator vector l is constructed by setting $l(v) = 1$ if vertex v has a positive relation with u and is currently in the list, $l(v) = -1$ otherwise, then the vertex u and v that $l(v) = 1$ are removed from the list. These steps are repeated until the list is empty. Then we get a initial L.

However, the sparsity of the affinity matrix leads to ineffectiveness when merging cluster indicator vectors, which will result in a relatively large number of clusters. To address this problem caused by sparse data, we append a sparsity factor of $\mathtt{sum}(L^t_{\mathcal{C}(u),:} + \mathbf{1})/2$ (the size of current cluster $\mathcal{C}(u)$) to the objective to discourage small clusters when optimizing for $\mathcal{C}(u)^*$ in (4). In our experiments when dealing with sparse data, we follow this routine.

Algorithm 2 Initialize L

Input : Affinity matrix $W^{n\times n}$
Output: Cluster indicator matrix L
$L = \emptyset$; $\forall i, j, G_{ij} = 2(W_{ij} > 0) - 1$; $\forall i, G_{ii} = 1$;
$pdegree =$ **sum**$(G, 2)$; $list =$ **sort**$(pdegree, \texttt{descend})$;
while $list$ is not empty **do**
  $u = list$.**pop**(); $\ell^{1\times n} = -1$;
  for $v \in list$ **do**
    if $G(u, v) == 1$ **then**
      $\ell_v = 1$; $list$.**remove**(v);
    end if
  end for
  $L = L \cup \ell$;
end while
return L

The computational complexity of our algorithm is linear in each iteration as we will see. Let n, a and r denote the number of vertices, the number of attraction edges and the number of repulsion edges respectively. The cost of initializing L is $O(a+nlogn)$, while the proposed algorithm involves complexity of $O(a + r + n)$ in each iteration, which is linear with the sum of the number of edges and the number of vertices. Obviously, it can be seen that the more sparse W is, the less time complexity it will achieve. This is a very useful property to be exploited, especially for large-scale problems in real applications. The number of iterations taken to convergence is relatively small from experience.

A related algorithm to pseudo-EM is the LocalSearch method proposed in the Clustering Aggregation framework [14]. The two algorithms are similar in the sense that they both can be used to optimize correlation clustering. However, the fundamental difference is that LocalSearch is based on a greedy vertex-wise manner, while ours works like EM algorithm.

4 Experiments

In this section, we evaluate the performance of our proposed algorithm pseudo-EM using both synthetic and real data. To investigate the numerical performance, we first conduct comparison with the optimal solution to the correlation clustering problem (1) and the SDP relaxation [6] on toy data. We then compare them to the algorithms including Swap-and-Explore, Expand-and-Explore and Adaptive-label ICM proposed in [2], which aims at large-scale correlation clustering. In synthetic experiments, we also compare with k-means which is the representative of traditional clustering methods. To generate general affinity matrices and the ground truth of clustering, we follow the recipe in [2]. In terms of real data, we conduct experiments on image segmentation and community detection tasks. To evaluate the quality of clustering with the ground truth, we use F_1-measure and recovery levels of k. F_1-measure takes value from $[0, 1]$, and a larger value implies higher quality of clustering; while the recovery level of k is the difference ratio of the selected k to the ground truth, where smaller values indicate better recovery of the number of clusters.

4.1 Numerical Comparison

To investigate the numerical performance of the proposed optimization methods, we randomly generate a problem with 20 vertices and 2 clusters. We first find the exact solution to problem (1) by searching over all possible clustering assignments, which adds up to $\sum_{i=1}^{20} C_{20}^{i} = 1,048,575$ clusterings, and choosing the one with the minimum objective value. We then compare the SDP relaxation and the proposed methods with the optimal solution. SDP is implemented with the semi-definite programming package SeDuMi [15] and the YALMIP toolbox[1]. A subtlety here is that the SDP objective is to maximize agreements, therefore a postprocessing step is taken following the SDP procedure to convert the result to the minimal disagreements solution.

Table 1 summarizes the comparison in terms of the loss objectives given the discretized solutions and time costs of optimization. We can first observe that the SDP relaxation is more effective than solving the problem exactly. However, the loss objective based on the discretized solution of the relaxed continuous result produced by SDP is greater than the exact loss. In the meantime, our proposed methods including pseudo-EM (P-EM) and online pseudo-EM (OP-EM) find solutions with loss objectives closer to the optimal one and significant improvements in time complexity.

4.2 Synthetic Data

To generate synthetic affinity matrix W, n vertices are assigned to c clusters with different sizes (size ratio between the largest and smallest clusters is 100) randomly. For each vertex, we sample the same number of neighbors with nonzero affinities. Then the adjacency matrix of ground truth is corrupted with noise both on the signs and values to get a real affinity matrix.

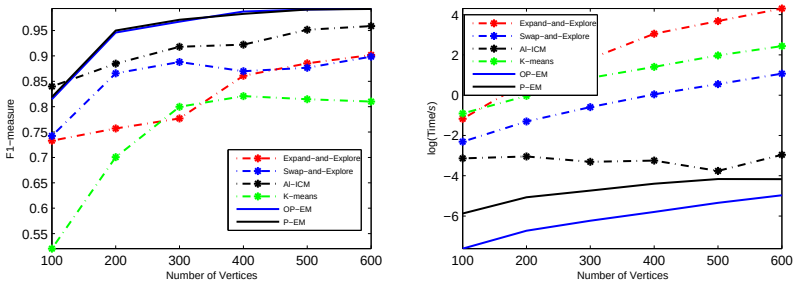
Due to the high computational complexity of Expand-and-Explore, Swap-and-Explore and k-means, we only compare with them as well as adaptive-label ICM on small scale data with the following parameters setting: $\#Clusters = 30$, $\#Neighbors = 20$, $Balance = 0.5$, $Noise = 0.1$, where $Balance$ is the ratio of the number of inter and intra cluster neighbors. The parameter k for k-means is set as the ground truth. Fig. 1 shows the F_1-measure and running time of different algorithms as the number of vertices increases. It can be seen that our methods produce more accurate clustering than the competing algorithms. On the other hand, although being provided with the true number of clusters, the clustering quality produced by k-means is not comparable to the rest of the algorithms. Another observation is that as the number of vertices grows, the clustering quality increases, which makes sense in that one can obtain better knowledge of the underlying distribution given more data. In addition, as mentioned above, the computational costs of Expand-and-Explore, Swap-and-Explore and k-means are high, therefore we will not include them in the following comparison on real data sets.

We further investigate the ability of different algorithms to automatically select the number of clusters k, which is illustrated in Fig. 2. As can be seen, the proposed methods are shown to be significantly more effective at selecting k. Similarly, we can observe that more data makes the estimate of k more accurate; while the difficulty of selecting k increases with the number of clusters given the same amount of data.

[1] http://users.isy.liu.se/johanl/yalmip/

Table 1. Loss comparison of exact optimum and SDP/P-EM/OP-EM

	Exact solution	SDP	P-EM	OP-EM
Loss	9.3588	78.2503	35.9834	35.9834
Time/s	858.7878	35.3129	6.1241e-4	5.6880e-4



(a) F_1-measure **(b)** Time

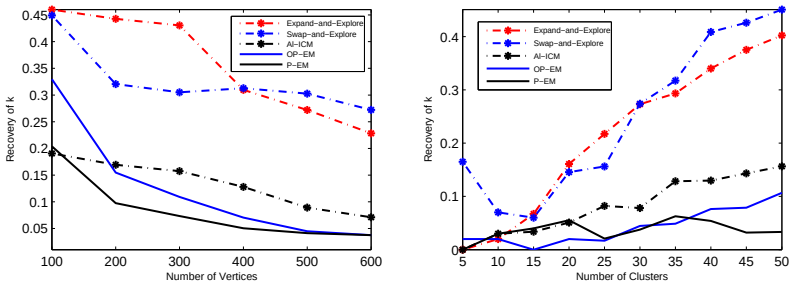
Fig. 1. Small scale data, #Clusters=30, #Neighbors=20, Balance=0.5, Noise=0.1. *(a) F_1-measure. (b) Running time.*

4.3 Image Data

To further demonstrate the scalability and quality of clustering of the proposed algorithms, we conduct experiments on the pixel-level image segmentation task. We take the image data in [16] and rescale the images to 134×200 for illustration. We use the classical normalized cut [17] as comparison.

Before running the algorithms, one should notice that the affinity matrix for correlation clustering consists of both positive and negative entries, which is different from normalized cut. Therefore, to construct the sparse affinity matrix, we first take the affinity matrix computed for normalized cut, followed by a nonlinear transformation to convert the nonnegative affinities to real valued affinities. The nonlinear transformation function is modified from $y = log\frac{x}{1-x}$ proposed by [1]. To avoid the problem occurred when $x = 1$, here we use the transformation $y = log\frac{1+(x-\delta)}{1-(x-\delta)}$ with the following properties: $y > 0$ when $x > \delta$; $y < 0$ when $x < \delta$ and $y = 0$ when $x = \delta$, where δ is a super parameter and is fixed to 0.05 in our experiments. Another subtlety regarding the comparison is the number of clusters k has to be predefined for normalized cut. In our experiments we set $k = 5$, while our methods automatically select k.

The comparison of image segmentation results is shown in Fig. 3, where the segmentation results of normalized cut are shown in the left column, while the results of our methods including OP-EM and OP-EM-S (OP-EM with sparsity factor) are shown in the middle and right columns respectively. From Fig. 3 we can observe that OP-EM-S is very effective at finding segments on images while OP-EM itself performs not as well, which justifies the idea that adding sparsity factor in (4) could be more effective for sparse data. On the other hand, the left column shows that for normalized cut, inappropriate value of k would result in improper segmentation, which is one of the



(a) Effect of number of vertices **(b)** Effect of number of clusters

Fig. 2. Recovery of k. *(a) #Clusters=30, #Neighbors=20, Balance=0.5, Noise=0.1. (b) #Vertices=500, #Neighbors=20, Balance=0.5, Noise=0.1.*

disadvantages of traditional clustering methods. As comparison, correlation clustering does not need to predefine the k value, which would alleviate the problems caused by unknown number of clusters in the data.

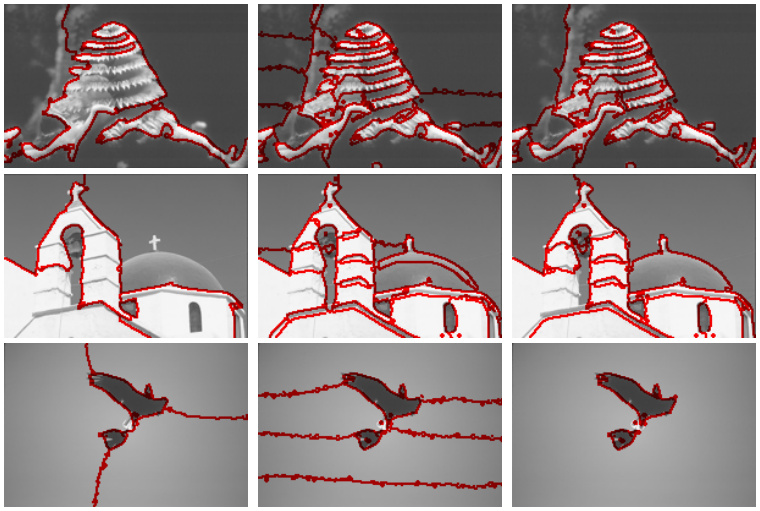


Fig. 3. Image segmentation on Berkeley Segmentation Dataset. *Left to right column: Normalized cut, online pseudo-EM (OP-EM) and online pseudo-EM with sparsity factor (OP-EM-S).*

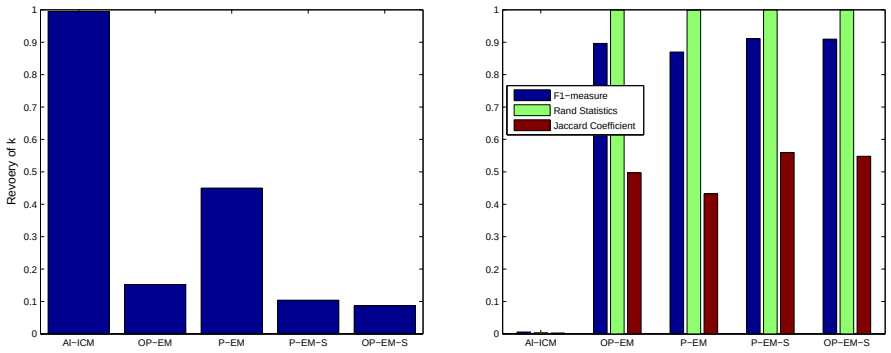
4.4 Social Network Data

Another natural application of clustering is community detection in networks. Here we conduct experiments on Amazon product co-purchasing network with a ground truth distribution of communities[2]. The original network contains 334,863 nodes and 925,872 edges. The number of communities in the network is 151,037 and average community

[2] http://snap.stanford.edu/data/com-Amazon.html

size is 19.78. Here we preprocess the data by keeping the top 5,000 communities and removing the redundant clusters and duplicate nodes. As a result we get a network with 16,685 nodes and 1,145 communities for our investigation.

We adopt the commonly used *Jaccard's coefficient* metric to measure the similarity between two nodes in the network [18], then convert the similarities to construct an affinity matrix as described in 4.3. Given the affinity matrix, we apply correlation clustering to detect the communities. We use the following metrics to evaluate the quality of the communities detected: F_1-measure, Rand Statistic and Jaccard Coefficient. All of these measures take value from $[0, 1]$, and larger values imply higher quality of clustering. Fig. 4 summarizes the comparison of the algorithms in terms of recovery of k and quality of clustering, which shows a clear advantage of our algorithms.



(a) Recovery of k **(b)** Clustering evaluation

Fig. 4. Community detection on Amazon network data.

5 Conclusion

In this paper we propose algorithms for solving general correlation clustering which could handle large-scale data from a different perspective by decoupling the affinity matrix and the cluster indicator matrix followed by a pseudo-EM optimization. To further improve the quality of optimization and alleviate the problem of local minimum, we adopt online udpates and append a sparsity factor for sparse data. Experimental results on both synthetic and real data demonstrate the effectiveness of the proposed techniques.

Correlation clustering is based on graphs with two types of edges: positive and negative, while in many real problems such as social networks or protein-protein interaction networks, the types of edges can be more diverse. Therefore, an interesting direction for future work will be to further generalize the proposed methodology to graphs with more diverse relations or interactions. Another important problem to look into is the theoretical analysis of the optimization quality of the proposed methods.

Acknowledgments. Research supported by the National Natural Science Foundation of China (No. 61003135) and the Fundamental Research Funds for the Central Universities (WK0110000022, WK0110000036).

References

1. Bansal, N., Blum, A., Chawla, S.: Correlation clustering. In: Proceedings of the 43rd Symposium on Foundations of Computer Science. FOCS '02, Washington, DC, USA, IEEE Computer Society (2002) 238
2. Bagon, S., Galun, M.: Large scale correlation clustering optimization. CoRR **abs/1112.2903** (2011)
3. Joachims, T., Hopcroft, J.: Error bounds for correlation clustering. In: Proceedings of the 22nd international conference on Machine learning. ICML '05, New York, NY, USA, ACM (2005) 385–392
4. Charikar, M., Guruswami, V., Wirth, A.: Clustering with qualitative information. In: Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on. (2003) 524–533
5. Demaine, E., Immorlica, N.: Correlation clustering with partial information. In Arora, S., Jansen, K., Rolim, J., Sahai, A., eds.: Approximation, Randomization, and Combinatorial Optimization.. Algorithms and Techniques. Volume 2764 of Lecture Notes in Computer Science., Springer Berlin Heidelberg (2003) 1–13
6. Swamy, C.: Correlation clustering: maximizing agreements via semidefinite programming. In: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms. SODA '04, Philadelphia, PA, USA, Society for Industrial and Applied Mathematics (2004) 526–527
7. Ailon, N., Charikar, M., Newman, A.: Aggregating inconsistent information: Ranking and clustering. J. ACM **55**(5) (November 2008) 23:1–23:27
8. Nowozin, S., Jegelka, S.: Solution stability in linear programming relaxations: graph partitioning and unsupervised learning. In: Proceedings of the 26th Annual International Conference on Machine Learning. ICML '09, New York, NY, USA, ACM (2009) 769–776
9. Glasner, D., Vitaladevuni, S.N., Basri, R.: Contour-based joint clustering of multiple segmentations. In: Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition. CVPR '11, Washington, DC, USA, IEEE Computer Society (2011) 2385–2392
10. Vitaladevuni, S., Basri, R.: Co-clustering of image segments using convex optimization applied to em neuronal reconstruction. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. (2010) 2203–2210
11. Yurii, N., Arkadii, N.: Interior-Point Polynomial Algorithms in Convex Programming. Society for Industrial and Applied Mathematics (1994)
12. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. Pattern Analysis and Machine Intelligence, IEEE Transactions on **23**(11) (2001) 1222–1239
13. Besag, J.: On the Statistical Analysis of Dirty Pictures. Journal of the Royal Statistical Society. Series B (Methodological) **48**(3) (1986) 259–302
14. Gionis, A., Mannila, H., Tsaparas, P.: Clustering aggregation. ACM Trans. Knowl. Discov. Data **1**(1) (March 2007)
15. Sturm, J.: Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. Optimization Methods and Software **11-12** (1999)
16. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on **33**(5) (2011) 898–916
17. Shi, J., Malik, J.: Normalized cuts and image segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on **22**(8) (2000) 888–905
18. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: Proceedings of the twelfth international conference on Information and knowledge management. CIKM '03, New York, NY, USA, ACM (2003) 556–559