A Non–Convex Optimization Approach to Correlation Clustering

Erik Thiel, Morteza Haghir Chehreghani, Devdatt Dubhashi
Chalmers University of Technology
{erikthi, morteza.chehreghani, dubhashi}@chalmers.se

Abstract

We develop a non-convex optimization approach to correlation clustering using the Frank-Wolfe (FW) framework. We show that the basic approach leads to a simple and natural local search algorithm with guaranteed convergence. This algorithm already beats alternative algorithms by substantial margins in both running time and quality of the clustering. Using ideas from FW algorithms, we develop subsampling and variance reduction paradigms for this approach. This yields both a practical improvement of the algorithm and some interesting further directions to investigate. We demonstrate the performance on both synthetic and real world data sets.

Introduction

Clustering is a central task in exploratory data analytics with applications in many domains including image processing, compression, knowledge management, social media analytics, etc. Clustering models usually assume a set of n objects (for instance, a set of documents, images or users) and the corresponding measurements in form of, e.g., the feature vectors or the pairwise similarities. The latter form, considers a (weighted) graph where the objects are the nodes and the pairwise similarities constitute the edge weights. Most of graph clustering models, e.g., *normalized cut* (Shi and Malik 2000), *spectral clustering* (Ng, Jordan, and Weiss 2001; von Luxburg 2007), power iteration clustering (Lin and Cohen 2010) and *dominant set clustering* (Pavan and Pelillo 2007; Haghir Chehreghani 2016), assume nonnegative pairwise similarities.

However, a particular clustering model, called *correlation clustering* (Bansal, Blum, and Chawla 2004; Demaine et al. 2006; Williamson and Shmoys 2011) is a very useful and flexible framework for unsupervised learning in the setting when one has both similarity as well as dissimilarity judgments between a set of objects. The problem has been extensively studied in the literature on approximation algorithms using linear programming (LP) and semi–definite programming (SDP) techniques.

Here we introduce a new approach to the problem based on a natural non–convex relaxation. We show that this relaxation is tight, in contrast to LP or SDP based relaxations. We then observe that our formulation is very well suited to the classical Frank-Wolfe (FW) or conditional gradient optimization algorithm and develop new algorithms for correlation clustering. There have been many recent advances in the FW method that renders it very suitable for large scale optimization and we take advantage of the block coordinate version of the FW algorithm. Also, we leverage recent advances in acceleration of first order optimization methods to develop several variants of our algorithms for correlation clustering ranging from a very simple combinatorial local search algorithm to various accelerated and randomized methods which are highly scalable for "Big Data" settings. In all these versions, we can either specify the number of clusters or we can have the algorithm discover the number of clusters in a data driven fashion. Building on recent results on the FW algorithm, we show that for multilinear functions (not necessarily concave/convex), the block FW algorithm has fast convergence properties. This translates into rigorous guarantees for the running time of our combinatorial correlation clustering algorithms. We also show that all our algorithms can be implemented very efficiently to scale to large scale systems (unlike previous methods based on LP and SDP relaxations. We also show that the quality of results obtained are vastly superior to those obtained from the methods from the approximation algorithms literature.

Background on Correlation Clustering

In the correlation clustering problem, we are given a graph $G = (V, E)$ with weights $w_{i,j}$ on the edges which may be *both positive and negative*. In the *maximizing agreements* version of the problem, we are required to partition the vertices into clusters so that the sum of the weights of edges within a cluster is maximized. We start with the exact formulation similar to (6.7) from (Williamson and Shmoys 2011):

$$\max \sum_{(i,j)\in E} w_{i,j} \boldsymbol{v}_i \cdot \boldsymbol{v}_j z$$
$$\boldsymbol{v}_i \in \{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_k\}, i = 1 \cdots n \tag{1}$$

(where $\boldsymbol{e}_i, i = 1 \cdots k$ are the basis vectors). In (Williamson and Shmoys 2011) and in much approximation algorithm literature, often the SDP relaxation of this formulation is considered. In this high dimensional relaxation, the number of clusters k disappears (which could be both a strength and a weakness). A rounding step (typically randomized) is needed to recover the clusters from the optimal solution to the SDP relaxation.

FW Algorithms

The natural continuous relaxation of (1) is

$$\max \sum_{(i,j)\in E} w_{i,j} \boldsymbol{v}_i \cdot \boldsymbol{v}_j$$
$$\boldsymbol{v}_i \in \text{ConvexHull}\left\{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_k\right\}, i = 1 \cdots n. \tag{2}$$

which is a non–convex (quadratic) optimization problem. Here the number of clusters k is retained in the relaxation and the optimal solution to the relaxation immediately translates into a clustering (with a simple rounding step if needed). If k is unknown, then we can set it to a known upper bound (or to n as in the SDP relaxation approaches) and we will show that the natural sparsity properties of the FW algorithm will recover the true number of clusters.

Tightness of Relaxation: Randomized Rounding

The natural randomized rounding algorithm based on the optimal solution $(v_i^*, i = 1 \cdots n)$ of (2) is:

Randomized Rounding For $i = 1 \cdots n$ independently, generate a distribution on $0/1$ vectors with exactly one 1 (i.e. a distribution on $\boldsymbol{e}_1, \cdots \boldsymbol{e}_k$) with marginals $v_{i,j}^*$, and let $\hat{\boldsymbol{v}}_i = \boldsymbol{e}_k$ with this distribution. A simple way to do this is as follows: divide the unit interval into k parts with the the jth part having size $v_{i,j}^*$. Then choose U uniformly at random in $[0, 1]$. If U falls into the jth interval, set $\hat{\boldsymbol{v}}_i = \boldsymbol{e}_j$.

Note that with this rule,

$$\begin{aligned} P[\hat{\boldsymbol{v}}_i = \hat{\boldsymbol{v}}_j] &= \sum_s P[\hat{\boldsymbol{v}}_i = \boldsymbol{e}_s = \hat{\boldsymbol{v}}_j] \\ &= \sum_s P[\hat{\boldsymbol{v}}_i = e_s] P[\hat{\boldsymbol{v}}_j = \boldsymbol{e}_s] \\ &= \sum_s v_{i,s}^* \cdot v_{j,s}^* \\ &= \boldsymbol{v}_i^* \cdot \boldsymbol{v}_j^*. \end{aligned}$$

and the expected value of the resulting solution is:

$$\begin{aligned} E[\sum_{(i,j)\in E} 1[\hat{\boldsymbol{v}}_i = \hat{\boldsymbol{v}}_j] w_{i,j}] &= \sum_{(i,j)\in E} P[\hat{\boldsymbol{v}}_i = \hat{\boldsymbol{v}}_j] w_{i,j} \\ &= \sum_{(i,j)\in E} \boldsymbol{v}_i^* \cdot \boldsymbol{v}_j^* w_{i,j} \end{aligned}$$

which is the optimal solution value of the relaxation (2). By the *probabilistic method*, there must exist at least one $0/1$ solution with this value, so this shows that

Theorem 1. *The continuous relaxation (2) of (1) is* **tight**.

This should be contrasted with the SDP relaxation which is *not* tight and known to have an *integrality gap* of approximately 0.87 (Feige and Schechtman 2002).

Block coordinate FW for non-convex functions

The classical Frank–Wolfe (1956) algorithm is one of the oldest methods for nonlinear constrained optimization and has seen an impressive revival recently in machine learning and signal processing to its low memory requirement and projection-free iterations (Jaggi 2013). This algorithm is perfectly tailored to solve the non–convex optimization problem in (2). In its classical form, the Frank-Wolfe algorithm can solve problems of the form $\max_{x\in D} f(x)$ where f is differentiable with L-Lipschitz gradient (a very standard assumption in optimization, which can be intuitively interpreted as requiring that the objective function must be "smooth") and the domain D is a convex and compact set.

Here we use the block coordinate version of the Frank–Wolfe algorithm from (Lacoste-Julien et al. 2013). This method applies to problem of the form

$$\max_{v\in\mathcal{M}^{(1)}\times\mathcal{M}^{(2)}\times\ldots\times\mathcal{M}^{(n)}} f(v)$$

where f is a concave function over $\mathcal{M}^{(1)}\times\mathcal{M}^{(2)}\times\ldots\times\mathcal{M}^{(n)}$ and $\mathcal{M}^{(i)} \subseteq \mathbb{R}^{n_i}$ is convex and compact for $i = 1, ..., n$. This method works well when certain subproblems are computationally cheap to solve when only considering the variables in one variable block $\mathcal{M}^{(i)}$ at a time. The method is described in Algorithm 1. Here we use $\nabla_i f$ to denote the i:th block of gradient of f. For the relaxed Problem 2 the natural block structure is to let each decision vector v_i belong to its own variable block. This way, updating a block simply means updating the cluster assignment of one node. We will show that this arrangement leads to very computationally simple iterations. Before that, we observe that the

Algorithm 1 Block-coordinate Frank-Wolfe Algorithm on a Product Domain

```
Let the initial solution $v^{(0)} \in \mathcal{M}^{(1)} \times \mathcal{M}^{(2)} \times ... \times \mathcal{M}^{(n)}$
for t = 1,...,T do
    Pick object $i$ at random in $\{1, ..., n\}$
    Find $s_{(i)} := \text{argmax}_{s_i' \in \mathcal{M}^{(i)}} s_{(i)}' \cdot \nabla_i f(v^{(t)})$
    Let $\gamma = \frac{2k}{t+2k}$ or optimize $\gamma$ by line-search
    Update $v_{(i)}^{(t+1)} := (1-\gamma) v_{(i)}^{(t)} + \gamma s_{(i)}$
```

block–coordinate version of the algorithm can be shown to converge for non–convex functions extending results of (Lacoste-Julien et al. 2013; Reddi et al. 2016). The convergence of Frank-Wolfe algorithms for non-concave problems can be measured by the Frank-Wolfe duality gap

$$d(v) = \max_{s\in\mathcal{M}} (s - v) \cdot \nabla f(v)$$

which is zero if and only v is a stationary point.

Theorem 2. *Let $\tilde{d}_t := \min_{0\le s\le t-1} d(v^{(t)})$. For f continuously differentiable (but not necessarily concave) with finite curvature constant C over $\mathcal{M}$, we have that the Frank-Wolfe iterates with line search satisfy*

$$E[\tilde{d}_t] \le \frac{Cn}{\sqrt{t}}$$

If f is multilinear in the variable blocks,

$$E[\tilde{d}_t] \leq \frac{(f^\star - f(v^0))n}{t}. \tag{3}$$

This convergence rate of $\mathcal{O}(\frac{1}{t})$ (in expectation) for block-coordinate Frank-Wolfe should be compared to the (deterministic) convergence of the duality gap of $\mathcal{O}(\frac{1}{\sqrt{t}})$ for general non-concave functions with ordinary Frank-Wolfe developed in (Reddi et al. 2016).

Proof. We prove the second part of the theorem since this is simple and applies to correlation clustering. The general result follows by combining the ideas in (Reddi et al. 2016; Lacoste-Julien et al. 2013). Duality in block-gap is closely related to gap over all variables. Following (Lacoste-Julien et al. 2013) we have

$$d(v) = \sum_{i=1}^{n} \max_{s_i \in \mathcal{M}_i} (s_i - v_i) \cdot \nabla_i f.$$

Let $d_i(v) := \max_{s_i \in \mathcal{M}^{(i)}} (s_i - v_i) \cdot \nabla_i f$. We have $E[d_i(v^{(t)})|v^{(t)}] = \frac{1}{n} d(v^{(t)})$ when choosing i u.a.r from $[n]$. Taking the expectation of this w.r.t to all randomly chosen blocks prior, we have $E[d_i(v^{(t)})] = \frac{1}{n} E[d(v^{(t)})]$. Since f is linear when fixing all variables except v_i, the optimal step length is $\gamma = 1$ and for this choice of step length we have $f(v^{(t+1)}) - f(v^{(t)}) = d_i(v^{(t)})$ (see equation 4). Taking the expectation and summing over t yields

$$\begin{aligned} \frac{1}{n} \sum_{t=0}^{T-1} E[d_i(v^{(t)})] &= \sum_{t=0}^{T-1} E[f(v^{(t+1)}) - f(v^{(t)})] \\ &= E[f(v^{(T)})] - f(v^{(0)}) \\ &\leq f^\star - f(v^{(0)}) \end{aligned}$$

Using $\frac{1}{n} \sum_{t=0}^{T-1} E[d_i(v^{(t)})] \geq \frac{T}{n} \tilde{d}_T$ yields the result. □

Combinatorial Local Search

Using the line search version of the FW algorithm, we observe that it simplifies to the combinatorial local search algorithm in Algorithm 2. To see why this holds, consider the so called linear programming oracle from algorithm 1

$$s_{(i)} := \underset{s'_i \in \mathcal{M}^{(i)}}{\operatorname{argmax}}\, s'_i \cdot \nabla_i f(v^{(t)}). \tag{4}$$

Here $\nabla_i f(v^{(t)}) = \sum_j w_{i,j} v_j^{(t)}$. The k:th element of this vector is (assuming all v_i integer) the cost associated with assigning node i to cluster k. It's easy to see that a solution to (4) is given by $s_i = e_{j^\star}$ where $j^\star = argmax_{j \in [k]} (\nabla_i f)_j$. Therefore, the LPO returns a 0/1 vector which has the highest objective when all other variables remain fixed.

This particular instantiation of our FW framework is similar to the greedy method in (Elsner and Schudy 2009), called BOEM. However, our approach enables us to analyze this local search method and very importantly to obtain a rigorous convergence rate. In addition, this framework allows us to utilize further improvements such as subsampling (stochastic FW) and variance reduction.[1]

[1]Moreover, BOEM at each step finds the object that yields a maximal improvement, which can be computationally expensive.

Algorithm 2 Local search for correlation clustering

```
Initialize $v_i \in \{e_1, ..., e_k\}$ randomly for $i = 1, ..., n$
while not converged do
    Select $i$ uniformly at random from $[n]$
    Assign $i$ to a cluster which maximizes the correlation clustering objective
```

Using the result in Theorem 2, we get a rigorous convergence result for this simple local search algorithm:

Theorem 3. *The local search algorithm converges (i.e., the expected duality gap $E[\tilde{d}_t]$ becomes ϵ-close to zero) at most in $n \sum_{i,j} |w_{i,j}|/\epsilon$ steps.*

This should be compared to the PTAS result of (Giotis and Guruswami 2006). Our algorithm is very efficient in practice unlike theirs.

Sparsity for data driven number of clusters For the typical clustering application, the number of clusters is not known. The FW algorithm is known to have favorable sparsity properties (Jaggi 2013) and we exploit this to discover the optimal number of clusters. In our experiments we have found that using a loose upper bound on k often yields very good results. We show empirically that we can even use $k = n$ in high noise settings and still retrieve the optimal number of clusters.

Algorithm 3 Stochastic Block-coordinate Frank-Wolfe

```
Initialize $v_i \in ConvexHull(e_1, ..., e_k)$ for $i = 1, ..., n$
for $t = 1, ..., T$ do
    Select $i$ uniformly at random from $[n]$
    Let $S$ be a random subset of $[n]$ (of fixed size)
    Let $\nabla_i f := \sum_{j \in S} w_{i,j} v_j$
    Let $j^\star := \operatorname{argmax}_{j \in [k]} (\nabla_i f)_j$
    Update $v_i \leftarrow (1 - \gamma_t) v_i + \gamma_t e_{j^\star}$
```

Stochastic Block FW

It is a common theme in machine learning to use randomness to speed up computation. The non-convex stochastic FW has been investigated in (Reddi et al. 2016). Algorithm 4 shows the pseudo code for a block-coordinate version of this algorithm. The non-block version of this algorithm is known to have inferior theoretical convergence properties, however, in our case one can show, using concentration of measure that if, for example, the ratio $\frac{n}{k}$ is large enough, taking only a subsample of the gradient will with high probability yield an exact update. This is a property of the block algorithm which might hold for other problems which has a natural block structure. Thus this algorithm is easy to implement and highly scalable and works very well in practice. We found that using $\gamma = 1$ works well in practice.

Variance reduced Block FW

Using stochastic steps in the optimization yields cheap iterations at the cost of inducing noise. An important technique in stochastic optimization to reduce the noise is the use of variance reduction techniques on the stochastic gradients. A number of recent advances have resulted in variance reduction techniques for acceleration of first order methods in convex optimization, in particular the SVRG and SAGA methods (Bottou, Curtis, and Nocedal 2018). Recently (Reddi et al. 2016) showed that the main ideas from SVRG and SAGA can be adapted to improve performance of FW algorithms in the non-convex setting. SAGA minimizes a sum $\frac{1}{n}\sum_i^n f_i(v)$ similar to stochastic gradient descent. However, in the general setup with SAGA, instead of moving in the direction of $-\nabla f_i(v)$ for a randomly chosen function f_i, we follow the average direction as

$$\nabla f_i(v) - \nabla f_i(\phi_i) + \frac{1}{n}\sum_{j=1}^{n}\nabla f_i(\phi_j)$$

where ϕ_i is the latest value of the variable v_i in which ∇f_i was evaluated. This yields an unbiased estimate of the gradient with typically lower variance. This allows for the use of larger step sizes. We adapt the SAGA based algorithm from (Reddi et al. 2016) to our framework. The result is described in Algorithm 4.

Algorithm 4 Stochastic FW with SAGA variance reduction

```
Initialize $v_i^0 \in ConvexHull(e_1, ..., e_k)$ for $i = 1, ..., n$
Let $g_{i,j} := w_{ij} v_j^{(0)}$                ▷ Memory of gradient
Let $g_i := \sum_j g_{i,j}$ for $i = 1, ..., n$
for $t = 1, ..., T$ do
    Select $i$ uniformly at random from $[n]$
    Let $I$ and $J$ be random subsets of $[n]$ (sampling with replacement) of size $b$
    Let $\nabla_i f := \sum_{i \in I} w_{i,j} v_j^{(t)}$
    Let $\hat{g}_i := \frac{n}{b}(\nabla_i f - \sum_{i \in I} g_i^{(t)}) + g_i^{(t)}$
    Let $s^{(t)} \in \text{argmax}_{s \in \mathcal{M}^{(i)}} (s - v_i^{(t)}) \cdot \hat{g}_i$
    Update $v_i^{(t+1)} \leftarrow (1 - \gamma_t) v_i^{(t)} + \gamma_t s^{(t)}$
    Update $g_{i,j}^{(t+1)} \leftarrow w_{i,j} v_j^{(t+1)}$ for $j \in J$
    Update $g_i^{(t+1)} \leftarrow g_i^{(t)} - \sum_{j \in J} g_{i,j}^{(t)} + \sum_{j \in J} g_{i,j}^{(t+1)}$
```

Experiments

In this section, we describe several experimental scenarios to investigate the different aspects of our algorithms and comparisons to other algorithms.

Robustness and quality

We investigate the quality, robustness and scalability of different algorithms. For this, we follow a generative approach to produce graphs with positive and negative edge weights, as follows. We fix the number of cluster $k = 5$. Then for a selected n, we assign each object randomly to one of the five different clusters. The weight between two nodes (objects) is a standard normal random variable. With probability $1 - p$ the sign of the weight is fixed such that the objects in the same cluster have positive edge weights and the objects in different clusters have negative edge weights. With probability p, the sign of a weight is random, i.e., p indicates the noise level. We repeat each setup 10 times and report the average results.

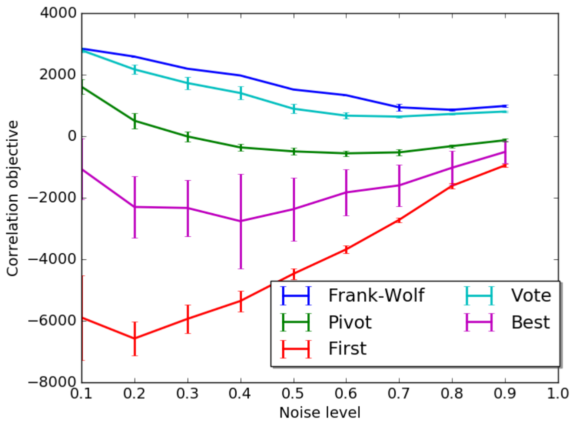

Figure 1: Comparison of FW and the other algorithms with respect to robustness to the noise p. FW yields more robust solutions with higher quality.

We first study the robustness of the different algorithms with respect to noise. For this, we fix the number of objects to $n = 2000$ and vary the noise level p from 0.0 to 0.9. The alternative methods are: PIVOT (Ailon, Charikar, and Newman 2008), and the three other heuristics BEST, FIRST and VOTE proposed in (Elsner and Schudy 2009). Figure 1 shows the objective of the clustering solutions computed by different methods. We observe that FW consistently yields the highest objectives for all noise levels, and for all problem instances. When the noise is low or intermediate, FW's superiority is more significant. As the noise level increases, the different methods perform more similarly, even though FW is still the best option.

We then study the scalability of different methods and demonstrate how they perform as the data size increases. Here, we fix $k = 5$ and $p = 0.3$, and vary n, the number of objects. Figure 2 shows the performance of different methods as function of n. We observe that FW again outperforms the alternative methods. For small n, the differences are less visible and as we increase n, they become more significant. This is due to the fact that the objective is quadratic with respect to n and thus grows faster. We realize the quadratic form of results for FW, consistent with the form of the objective function. It is important to notice that consistent to the results in (Elsner and Schudy 2009), we observe that the algorithms based on SDP or integer linear programming are very slow in practice, in fact they are applicable to only a few hundred objects. Moreover, they often yield worse solutions compared to the alternatives in terms of quality (for example, for $n = 200$ the SDP relaxation takes about 15 hours, whereas our method takes only a few seconds).

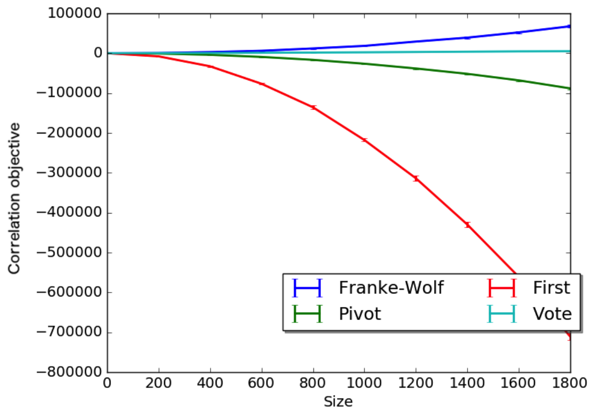

Figure 2: Comparison of the performance of different algorithms with respect to the problem size n. The y-axis shows the computed objective.

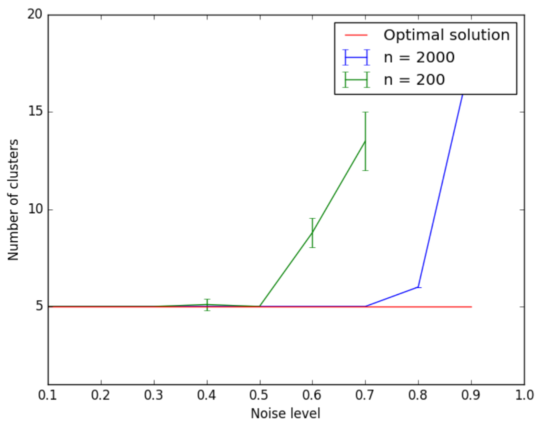

Figure 3: Number of clusters found by the FW (local search) algorithm at different noise levels. FW is often able to obtain the precise number of clusters even it is not fixed in advance.

Model order selection

A unique property of correlation clustering is that increasing the number of clusters does not always yield a solution with a lower cost (higher objective), rather, the objective might be optimal for a finite number of clusters. For example, consider a graph whose all edge weights are positive. Then a solution with only one cluster would have the highest objective, whereas increasing the effective (nonempty) number of clusters yields cutting some positive edges and thus decreases the objective. However, for example, the k-means optimal objective (negative cost function) improves as we increase the number of clusters (up to n clusters).

In this section, we investigate the ability of FW to identify the correct number of clusters (model order selection) in correlation clustering.

We use the aforementioned generative approach to produce data with different noise rates p, where k is fixed to 5. We investigate two cases where $n = 200$ and $n = 2000$. Figure 3 shows the number of clusters estimated by FW, respectively when $n = 200$ and $n = 2000$ (FW is initialized by $\hat{k} = n$ clusters, and we report the average results on 10 different problem instantiations). We observe that in both settings, FW finds the optimal number of clusters correctly up to a high noise rate. However, as we increase n, the model becomes more robust to noise as the number of objects per cluster and per p increases too.

Note that in a similar way, our method can identify the outliers, as they establish negative edges with many other objects and occur in singleton clusters.

Subsampling

The local search FW algorithm is efficient and produces clusters of higher quality compared to the alternatives. However, the runtime might be still too high in particular on large-scale data. Hence, we have proposed subsampling (Algorithm 4) in order to reduce even more the computational burden.

In Figure 4, we study subsampling for different values of the batch size, on a subset of 20 newsgroup data collection. Given the ground-truth with $k = 5$ clusters, the weight between two articles is 1 if they come from the same newsgroup, and -1 otherwise. With probability p the weight is corrupted to a random uniform value in $[-1, 1]$. In this experiment, we study the correlation clustering objective (the measure of quality) versus the effective number of passes through the FW blocks (as a measure of complexity). We observe that with a small subsampling ratio (i.e., $q = |S|/n = 0.1$, corresponding to a batch size of $b = 20$) we attain the maximal objective with a minimal numbers of passes. As we increase the batch size or q, we essentially require more computation in overall. on the other hand, a too small batch size (e.g., 10) leads to very unstable and noisy estimation of the gradient and, therefore, the optimization might fluctuate and fail (see Figure 5).

Variance reduction

Subsampling typically works well as long as the batch size is sufficiently large compared to $\frac{n}{k}$. In Figure 4 we see that decreasing the subsampling ratio q down to 0.1 yields computational efficiency. For subsampling ratios larger than 0.1, it is likely that the stochastic gradient provides very accurate updates and thus we do not observe a significant improvement if we use the variance reduction method. However, the other end of this spectrum is using very small batch sizes, where the gradient estimations are very poor and thus variance reduction could be more useful. Figure 5 illustrates the impact of variance reduction in such a setting, where we show the trajectory of the objective for subsampling with (SAGA) and without variance reduction. The parameters are: $n = 300$, $k = 10$, $p = 0.3$, and the batch size $b = 10$ (i.e., $q = 0.03$). Even though at the very early iterations the two methods perform rather similarly, variance reduction very quickly starts out performing the other. After a sufficient number of iterations, it yields a significant objective

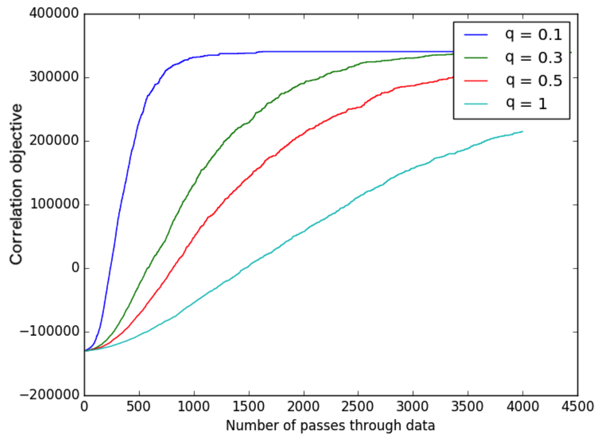

Figure 4: Frank-Wolfe with subsampling. The x-axis shows the effective number of passes through the blocks and y-axis is the correlation clustering objective. Subsampling helps to significantly reduce the computations.

(because of more informative estimation of the gradients), whereas the subsampling without variance reduction always yields very low objectives. Note that other variance reduction methods such as SVRG can be employed as well.

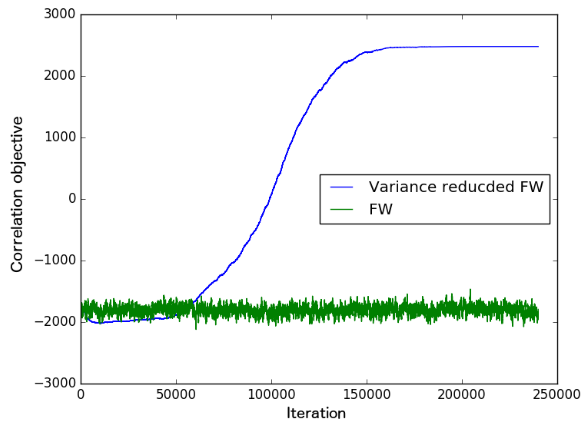

Figure 5: Correlation clustering objective for stochastic FW (SFW) with and without variance reduction, where both of the variants use batch size $= 10$. Due to small batch size, variance reduction is useful to compute more informative gradients.

Partitions in World Color Survey

The nature of color categories in the world's languages is contested. One major view holds that color categories are organized around universal focal colors, whereas an opposing view holds instead that categories are defined at their boundaries by linguistic convention. An influential proposal from cognitive scientists posits that color naming across languages reflects optimal or near-optimal partitions of an irregularly shaped perceptual color space (Regier, Kay, and Khetarpal 2007). We formalize this idea using the World Color Survey dataset consisting of human labellings of 330 color tiles in different languages. Psychologists have developed an embedding of these color tiles in 3-dimensional euclidean space. We use this embedding to calculate a weight between two color tiles with embeddings x and y in the following way

$$w(x, y) = \exp(-c||x - y||^2) - \frac{1}{2}$$

with $c = 0.001$. We apply our local search correlation clustering algorithm to cluster the color tiles based on these weights. Using $k = n$ our solution uses 32 different clusters. We also fix k for several different small values. It is interesting to see if the clusters we find correspond to color clusters which naturally emerge in human language. To this end we use the World color survey to construct human clustering of the color tiles. We compare the rand index and cluster cost of the clusters and compare it against color-naming data from a broad range of languages that use a similar amount of clusters. The result shows that it accounts for some universal tendencies in color naming while also accommodating some observed cross-language variation. The results are reported in Table 1 and in Figure 6.

k	7	9	11
FW	-321	76	283
Human	-2350	-1820	-886
normalized Rand	0.3	0.37	0.44

Table 1: The results of the color partitioning task.

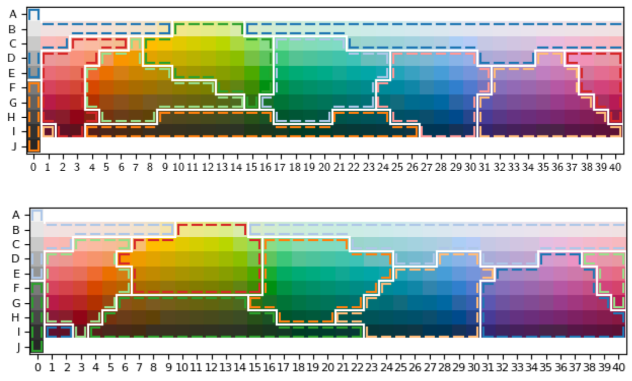

Figure 6: Clustering of color tiles. Top plot is a clustering based on majority vote by humans.

Twenty newsgroup dataset

In this section, we study the performance of different algorithms on several subsets of 20 newsgroup data collection.
1. *data1*: the documents in categories 'rec.sport.baseball, soc.religion.christian, rec.autos, talk.politics.mideast,

Algorithm	CC objective	Rand	NMI	$\hat{k}$
FW	179456.94	0.31	0.49	7.4
PIVOT	79681.22	0.09	0.14	10.6
FIRST	43722.62	0	0.01	3.4
BEST	88304.08	0.11	0.18	3
VOTE	171163.20	0.27	0.39	6.2

Table 2: Performance on *data1*.

Algorithm	CC objective	Rand	NMI	$\hat{k}$
FW	163563.83	0.25	0.39	6.2
PIVOT	95860.46	0.06	0.12	10.2
FIRST	45206.78	0	0.01	2.6
BEST	67322.08	0.04	0.08	2.4
VOTE	156492.40	0.19	0.32	6.6

Table 3: Performance on *data2*.

Algorithm	CC objective	Rand	NMI	$\hat{k}$
FW	194525.12	0.41	0.56	5.6
PIVOT	91475.30	0.11	0.17	10
FIRST	48596.36	0	0.01	2.6
BEST	105004.22	0.13	0.23	2.6
VOTE	188895.60	0.36	0.51	5.6

Table 4: Performance on *data3*.

Algorithm	CC objective	Rand	NMI	$\hat{k}$
FW	154102.30	0.24	0.38	5.2
PIVOT	82156.26	0.09	0.12	9.8
FIRST	48596.36	0	0.01	2
BEST	60007.92	0.03	0.09	2.4
VOTE	147882	0.21	0.33	6.6

Table 5: Performance on *data4*.

misc.forsale'.
2. *data2*: the documents in categories 'rec.autos, comp.sys.mac.hardware, misc.forsale, talk.politics.mideast, sci.electronics'.
3. *data3*: the documents in categories 'talk.politics.mideast, rec.motorcycles, rec.sport.hockey, soc.religion.christian, comp.sys.mac.hardware'.
4. *data4*: the documents in categories 'misc.forsale, sci.space, comp.sys.ibm.pc.hardware, talk.politics.misc, rec.motorcycles'.
5. *data5*: the documents in categories 'rec.sport.hockey, soc.religion.christian, talk.politics.guns, rec.motorcycles, sci.space'.

Each dataset consists of 5 clusters corresponding to the randomly selected categories. For each dataset, we compute the PCA of the TF-IDF vectors using 40 principal vectors.[2] The similarity between two documents is defined as the cosine similarity between the PCA vectors, which is a number in $[-1, +1]$.

In Tables 2-6, we compare the results of different algorithms with respect to the correlation objective, normalized Rand score and normalized mutual information. For each dataset, we run the algorithms 10 times and take the average results. According to the results, FW performs significantly better than the other algorithms in terms of all the three evaluations criteria. As mentioned, correlation clustering has the property to obtain the number of clusters. In the case of FW, we initialize the solution by a large number, e.g. 20, compared to the true number of clusters, and at the end only choose the non-empty clusters. Other methods compute the number of clusters automatically. We observe that FW also provides to compute more correctly the true number of clusters.

In Tables 2-6, we also report the estimated number of clusters by each method (we run FW with 20 initial clusters). We again observe that FW provides very precise estimation of the correct number of clusters. Except *data1*, FW's estimations are the most accurate numbers, and even on *data5* it gives exactly the correct number of clusters.

[2]Before we calculate TF-IDF we remove all headers and footers from the documents. Our results are consistent among different reasonable number of principal components.

Related Work

Correlation clustering was first introduced in (Bansal, Blum, and Chawla 2004) where the model and the approximation algorithms assume a complete graph with edge weights in $\{-1, +1\}$. Then, several methods (e.g., (Demaine et al. 2006; Charikar, Guruswami, and Wirth 2003; Ailon, Charikar, and Newman 2008)) developed approximation algorithms for general graphs in different setups, i.e, for maximizing the agreements or minimizing the disagreements. However, while minimizing the disagreements is even APX-hard (Demaine et al. 2006), for the maximum agreements variant polynomial-time approximation scheme (PTAS) results have been proposed (Bansal, Blum, and Chawla 2004; Giotis and Guruswami 2006). (Charikar, Guruswami, and Wirth 2003) prove that even finding a PTAS is APX-hard for minimizing the disagreements.

In practice, such approximation algorithms are often inefficient and/or the approximation gap is huge leading to low-quality solutions in terms of objective (Elsner and Schudy 2009). Semidefinite programming (SDP) relaxation of correlation clustering (e.g., (Charikar, Guruswami, and Wirth 2003; Mathieu and Schudy 2010; Williamson and Shmoys 2011)) has shown to yield a tighter bound on the quality

Algorithm	CC objective	Rand	NMI	$\hat{k}$
FW	179930.28	0.33	0.51	5.0
PIVOT	84201.86	0.09	0.13	10.6
FIRST	41136.74	0	0.01	1.8
BEST	66779.60	0.06	0.10	2.2
VOTE	171585.60	0.26	0.42	5.8

Table 6: Performance on *data5*.

and a more efficient optimization algorithm. However, it is still slow and impractical on fairly large datasets (Elsner and Schudy 2009). Methods like PIVOT (Ailon, Charikar, and Newman 2008), and the several heuristics proposed in (Elsner and Schudy 2009) (e.g., BEST, FIRST and VOTE) are efficient but yield poor results, as demonstrated by our numerical studies. Recently, (Haghir Chehreghani 2017) has developed an alternative regularization of the k-way min cut cost function and a local search optimization for that, which corresponds to correlation clustering in some cases.

Conclusion

We proposed an efficient optimization framework for correlation clustering based on Frank-Wolfe (FW) optimization techniques. We showed that the well-known block-coordinate FW with the line search step size becomes a natural local search algorithm with guaranteed convergence. This algorithm outperforms the alternatives in terms of quality and robustness, and is applicable to large datasets. Moreover, we developed subsampling (stochastic) and variance reduction to improve even further the scalability of the method. Finally, we performed several numerical studies to investigate the different aspects of the framework.

References

Ailon, N.; Charikar, M.; and Newman, A. 2008. Aggregating inconsistent information: Ranking and clustering. *J. ACM* 55(5):23:1–23:27.

Bansal, N.; Blum, A.; and Chawla, S. 2004. Correlation clustering. *Machine Learning* 56(1-3):89–113.

Bottou, L.; Curtis, F. E.; and Nocedal, J. 2018. Optimization methods for large-scale machine learning. *SIAM Review* 60(2):223–311.

Charikar, M.; Guruswami, V.; and Wirth, A. 2003. Clustering with qualitative information. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, 524–533.

Demaine, E. D.; Emanuel, D.; Fiat, A.; and Immorlica, N. 2006. Correlation clustering in general weighted graphs. *Theor. Comput. Sci.* 361(2-3):172–187.

Elsner, M., and Schudy, W. 2009. Bounding and comparing methods for correlation clustering beyond ilp. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, ILP '09, 19–27.

Feige, U., and Schechtman, G. 2002. On the optimality of the random hyperplane rounding technique for MAX CUT. *Random Struct. Algorithms* 20(3):403–440.

Giotis, I., and Guruswami, V. 2006. Correlation clustering with a fixed number of clusters. *Theory of Computing* 2(13):249–266.

Haghir Chehreghani, M. 2016. Adaptive trajectory analysis of replicator dynamics for data clustering. *Machine Learning* 104(2-3):271–289.

Haghir Chehreghani, M. 2017. Clustering by shift. In *2017 IEEE International Conference on Data Mining, ICDM*, 793–798.

Jaggi, M. 2013. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, 427–435.

Lacoste-Julien, S.; Jaggi, M.; Schmidt, M. W.; and Pletscher, P. 2013. Block-coordinate frank-wolfe optimization for structural svms. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, 53–61.

Lin, F., and Cohen, W. W. 2010. Power iteration clustering. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, 655–662.

Mathieu, C., and Schudy, W. 2010. Correlation clustering with noisy input. In *Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, 712–728.

Ng, A. Y.; Jordan, M. I.; and Weiss, Y. 2001. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, 849–856.

Pavan, M., and Pelillo, M. 2007. Dominant sets and pairwise clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* 29(1):167–172.

Reddi, S. J.; Sra, S.; Póczos, B.; and Smola, A. J. 2016. Stochastic frank-wolfe methods for nonconvex optimization. In *54th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2016, Monticello, IL, USA, September 27-30, 2016*, 1244–1251.

Regier, T.; Kay, P.; and Khetarpal, N. 2007. Color naming reflects optimal partitions of color space. *Proceedings of the National Academy of Sciences* 104(4):1436–1441.

Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 22(8):888–905.

von Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and Computing* 17(4):395–416.

Williamson, D. P., and Shmoys, D. B. 2011. *The Design of Approximation Algorithms*. Cambridge University Press.