Wang and Qian *EURASIP Journal on Bioinformatics and Systems Biology* (2016) 2016:9
DOI 10.1186/s13637-016-0041-1

EURASIP Journal on Bioinformatics and Systems Biology

RESEARCH Open Access

Stochastic block coordinate Frank-Wolfe algorithm for large-scale biological network alignment

Yijie Wang* and Xiaoning Qian

Abstract

With increasingly "big" data available in biomedical research, deriving accurate and reproducible biology knowledge from such big data imposes enormous computational challenges. In this paper, motivated by recently developed stochastic block coordinate algorithms, we propose a highly scalable randomized block coordinate Frank-Wolfe algorithm for convex optimization with general compact convex constraints, which has diverse applications in analyzing biomedical data for better understanding cellular and disease mechanisms. We focus on implementing the derived stochastic block coordinate algorithm to align protein-protein interaction networks for identifying conserved functional pathways based on the IsoRank framework. Our derived stochastic block coordinate Frank-Wolfe (SBCFW) algorithm has the convergence guarantee and naturally leads to the decreased computational cost (time and space) for each iteration. Our experiments for querying conserved functional protein complexes in yeast networks confirm the effectiveness of this technique for analyzing large-scale biological networks.

Keywords: Network alignment, IsoRank, Stochastic optimizatoin, Frank-Wolfe algorithm

1 Introduction

First-order methods in convex optimization have attracted significant attention in statistical learning in recent years. They are appealing to many learning problems, such as LASSO regression and matrix completion, which have diverse applications in analyzing large-scale biological systems and high-dimensional biomedical measurement profiles [1, 2]. These first-order optimization methods scale well with the current "big" data in many biomedical applications due to their advantages that they have low computation burden per iteration and they are easy to be implemented on parallel computational resources.

In this paper, we focus on the Frank-Wolfe algorithm, which is also known as the conditional gradient method. One of its advantages is that at each iteration step, it decomposes the complex constrained optimization problem into subproblems that are easier to solve. Additionally, it is a projection-free algorithm, which avoids solving the projection problem for constrained optimization as done in many other algorithms. The original Frank-Wolfe algorithm, developed for smooth convex optimization on a polytope, dates back to Frank and Wolfe [3]. Dunn and Harshbarger [4, 5] have generalized the algorithm to solve the optimization for more general smooth convex objective functions over bounded convex feasible regions. Recently, researchers [6] have proposed stochastic optimization ideas to scale up the original Frank-Wolfe algorithm.

Based on these previous seminal efforts, our main contribution in this paper is that we generalize the stochastic block coordinate Frank-Wolfe algorithm proposed in [6], previously with *block separable constraints*, to solve more general optimization problems with *any convex compact constraints*, including the problems with block inseparable constraints. Such a generalized algorithm has a broader range of biomedical applications, including biological network alignment. We prove the convergence of our generalized stochastic block coordinate Frank-Wolfe algorithm and evaluate the algorithm performance for querying conserved functional protein complexes in real-world protein-protein interaction (PPI) networks.

*Correspondence: wyj.jackson@gmail.com
Department of Electrical and Computer Engineering, Texas A&M University, College Station, USA

In the following sections, we first describe the model formulation of the optimization problems that we are generally interested. Specifically, to address potential difficulty from more general convex compact constraints, we derive a new stochastic block coordinate Frank-Wolfe algorithm and provide the convergence proof. Then, we formulate the IsoRank problem for network alignment [7] as a convex programming problem and develop a stochastic block coordinate Frank-Wolfe (SBCFW)-IsoRank algorithm based on our new stochastic block coordinate Frank-Wolfe algorithm. At last, in our experiments, we show the efficiency and effectiveness of our algorithm for solving the PPI network query problem.

2 Stochastic block coordinate descent Frank-Wolfe algorithm

Consider the minimization problem:

$$\begin{aligned} &\min\colon f(\mathbf{x}) \\ &\quad s.t.\ \mathbf{x} \in \mathfrak{D}, \end{aligned} \tag{1}$$

where the objective function $f(\mathbf{x})$ is convex and differentiable on $\mathbf{R}^N$ and the domain $\mathfrak{D}$ is a compact convex subset of any vector space. We assume that the optimal solution $\mathbf{x}^*$ to the above problem is non-empty and bounded without loss of generality.

Assume that we can decompose the solution space $\mathbf{R}^N$ into n equal-size subspaces:

$$\mathbf{R}^N = \bigoplus_{i=1}^{n} \mathbf{R}^{N_i},\ N = \sum_{i=1}^{n} N_i, \tag{2}$$

where $N_1 = \ldots = N_i = \ldots, N_n$ and $\mathbf{R}^{N_i}$ denotes the ith equal-size subspace along the corresponding coordinates. This decomposition enables scalable stochastic optimization algorithms. Based on this decomposition, we introduce matrices U_i, who sum up to an identity matrix $I_N = \sum_{i=1}^{n} U_i$, and U_i is a $N \times N$ matrix with $U_i(t,t) = 1, t \in R^{N_i}$ on its diagonal and the other entries being equal to zero. In typical stochastic optimization algorithms [8, 9], instead of computing the gradient $\nabla f(\mathbf{x})$ at each iteration, the *partial gradient* of $f(\mathbf{x})$ on a randomly selected subspace $\mathbf{R}^{N_i}$ is used:

$$\nabla_i f(\mathbf{x}) = U_i \nabla f(\mathbf{x}). \tag{3}$$

Now, we generalize the previous stochastic block coordinate Frank-Wolfe algorithm derived in [6] to solve more general optimization problems with any compact convex constraints $\mathfrak{D}$. The new generalized SBCFW algorithm is illustrated in Algorithm 1. In the pseudo code, the operation $i = \mathfrak{C}_k$ randomly selects one of the n equal-size subspaces to update the partial gradient at each iteration with the same probability. In addition, $U_j \times \mathbf{s} = U_j \times \mathbf{x}^k$ denotes the condition that the elements of the jth block of $\mathbf{s}$ equal to the elements of the jth block of $\mathbf{x}^k$.

Algorithm 1 Generalized SBCFW algorithm

1 Let $\mathbf{x}^0 \in \mathfrak{D}$, $k = 0$.
2 **While** stopping criteria are not satisfied, **do**
3 Randomly divide $\mathbf{R}^N$ into n blocks $\mathbf{R}^N = \bigoplus_{i=1}^{n} \mathbf{R}^{N_i}$;
4 Choose $i = \mathfrak{C}_k$;
5 Find $\mathbf{s}_i^k$ such that
6 $\mathbf{s}_i^k := arg \min_{U_j \times \mathbf{s} = U_j \times \mathbf{x}^k, \forall j \neq i;\ \mathbf{s} \in \mathfrak{D}} \nabla_i f(\mathbf{x}^k)^T (\mathbf{s} - \mathbf{x}^k)$;
7 Determine the step size γ
8 $\gamma := arg \min_{\gamma \in [0,1]} f\left((1-\gamma)\mathbf{x}^k + \gamma \mathbf{s}_i^k\right)$;
9 Update $\mathbf{x}^{k+1} := (1-\gamma)\mathbf{x}^k + \gamma \mathbf{s}_i^k$;
10 $k = k + 1$;
11 **EndWhile**

Note that our generalized SBCFW algorithm is similar to the algorithm in [6], which aims to solve optimization problems with block separable constraints and has the sublinear convergence property. However, our algorithm provides a more generalized framework, which can manipulate any convex and compact constraints no matter whether they are block separable or not. Because the setup of our algorithm is more general without any specific structure, it is difficult to obtain theoretical convergence rate guarantees. In this paper, we only provide the proof that our SBCFW converges to the global optimum. The convergence guarantee of the generalized SBCFW algorithm is provided by Theorem 1 below, which is based on.

Lemma 1. At each iteration of the SBCFW algorithm, the following inequality holds

$$\nabla f\left(\mathbf{x}^k\right)^T \left(E_i\left[\mathbf{s}_i^k\right] - \mathbf{x}^k\right) \leq 0, \tag{4}$$

where $E_i\left[\mathbf{s}_i^k\right]$ is the expectation of $\mathbf{s}_i^k$ with respect to the random selection of the ith coordinate block to the corresponding subspace.

Proof. Assuming at the kth iteration, we solve the following optimization problem:

$$\begin{aligned} &\min\colon Z_k^i(\mathbf{s}) := \nabla_i f\left(\mathbf{x}^k\right)^T \left(\mathbf{s} - \mathbf{x}^k\right) \\ &\quad s.t.\ U_j \times \mathbf{s} = U_j \times \mathbf{x}^k,\ \forall j \neq i, \\ &\qquad \mathbf{s} \in \mathfrak{D}. \end{aligned} \tag{5}$$

The solution to (5) is $\mathbf{s}_i^k$. With $\mathbf{s}_i^k$ achieving the minimum of (5), we have

$$Z_k^i\left(\mathbf{s}_i^k\right) \leq Z_k^i\left(\mathbf{x}^k\right) = \nabla_i f\left(\mathbf{x}^k\right)^T \left(\mathbf{x}^k - \mathbf{x}^k\right) = 0. \tag{6}$$

Therefore,

$$Z_k^i\left(\mathbf{s}_i^k\right) = \nabla_i f\left(\mathbf{x}^k\right)^T\left(\mathbf{s}_i^k - \mathbf{x}^k\right) \le 0. \tag{7}$$

Taking expectation on both sides of the above inequality with respect to random blocks, we obtain

$$\begin{aligned} & E_i\left[\nabla_i f\left(\mathbf{x}^k\right)^T\left(\mathbf{s}_i^k - \mathbf{x}^k\right)\right] && \le 0 \\ \Rightarrow \quad & \frac{1}{n}\sum_i \nabla_i f\left(\mathbf{x}^k\right)^T\left(\mathbf{s}_i^k - \mathbf{x}^k\right) && \le 0 \\ \Rightarrow \quad & \left(\sum_i \nabla_i f\left(\mathbf{x}^k\right)\right)^T \frac{1}{n}\left(\sum_i\left(\mathbf{s}_i^k - \mathbf{x}^k\right)\right) && \le 0 \\ \Rightarrow \quad & \left(\sum_i \nabla_i f\left(\mathbf{x}^k\right)\right)^T\left(\frac{1}{n}\sum_i \mathbf{s}_i^k - \mathbf{x}^k\right) && \le 0 \\ \Rightarrow \quad & \nabla f\left(\mathbf{x}^k\right)^T\left(E_i\left[\mathbf{s}_i^k\right] - \mathbf{x}^k\right) && \le 0. \end{aligned} \tag{8}$$

The inequality in the third line can be derived based on the fact that $\mathbf{s}_i^k - \mathbf{x}^k$ is a vector with only its ith coordinate block having non-zero values and the other parts being all zeros. With that, the summation in the second line can be written as the inner product between vectors $\sum_i \nabla_i f\left(\mathbf{x}^k\right)$ and $\sum_i\left(\mathbf{s}_i^k - \mathbf{x}^k\right)$. □

We now analyze the convergence of the new SBCFW algorithm based on Lemma 1 from two cases. The first case is when

$$\nabla f\left(\mathbf{x}^k\right)^T\left(E_i\left[\mathbf{s}_i^k\right] - \mathbf{x}^k\right) = 0. \tag{9}$$

This simply means that $\mathbf{x}^k$ is a stationary point. Because the original objective function $f(\mathbf{x})$ is convex, we can conclude that $\mathbf{x}^k$ is the global minimum. Another case is when

$$\nabla f\left(\mathbf{x}^k\right)^T\left(E_i\left[\mathbf{s}_i^k\right] - \mathbf{x}^k\right) < 0, \tag{10}$$

indicating that $E_i\left[\mathbf{s}_i^k\right] - \mathbf{x}^k$ is a descent direction based on the definition [10]. Hence, $E_i\left[\mathbf{s}_i^k\right] - \mathbf{x}^k$ can move along the direction to get closer to the global minimum in expectation. Furthermore, we compute the optimal step size at each iteration; therefore, the objective function values are guaranteed to be non-increasing. With that, we present Theorem 1 as follows:

Theorem 1. The sequence $\left\{f\left(\mathbf{x}^1\right), f\left(\mathbf{x}^2\right), \ldots, f\left(\mathbf{x}^k\right), \ldots\right\}$ generated by the SBCFW algorithm is non-increasing

$$f\left(\mathbf{x}^1\right) \ge f\left(\mathbf{x}^2\right) \ge \ldots \ge f\left(\mathbf{x}^k\right) \ge f\left(\mathbf{x}^{k+1}\right), \quad k \to \infty. \tag{11}$$

3 Biological network alignment

3.1 Optimization model formulation

In this section, we re-formulate the involved optimization problem for the network alignment algorithm—IsoRank [7] to address the potential computational challenges of aligning multiple large-scale networks. The new formulation has the same mathematical programming structure as the problem (1).

Let G_a and G_b be two biological networks to align. Two networks has N_a and N_b vertices, respectively. We define $B \in \mathbf{R}^{(N_a \times N_b)\times(N_a \times N_b)}$ as the Cartesian product network from G_a and G_b: $B = G_a \otimes G_b$. Denote the all-one vector $\mathbf{1} \in \mathbf{R}^{N_a \times N_b}$ and

$$\bar{B} = B \times \mathrm{Diag}(B\mathbf{1})^{-1}, \tag{12}$$

where $\mathrm{Diag}(B\mathbf{1})$ can be considered as a degree matrix with $B\mathbf{1}$ on its diagonal and all the other entries equal to zero. $\bar{B}$ contains the transition probabilities for the underlying Markov random walk in IsoRank [7]. It is well known that if G_a and G_b are connected networks and neither of them is bipartite graph, then the corresponding Markov chain represented by $\bar{B}$ is irreducible and ergodic, and there exists a unique stationary distribution for the underlying state transition probability matrix $\bar{B}$. The goal of the IsoRank algorithm is to find the maximal right eigenvector of the matrix $\bar{B}$: $\bar{B}\mathbf{x} = \mathbf{x}$ and $\mathbf{1}^T\mathbf{x} = 1$, $\mathbf{x} \ge 0$, which corresponds to the best correspondence relationships between vertices across two networks. When two networks are of reasonable size, spectral methods as well as power methods can be implemented to solve the IsoRank problem [7]. However, with large-scale networks, the transition probability matrix $\bar{B}$ can be extremely large (quadratic with $N_a \times N_b$) and spectral and power methods can be computationally prohibitive. In this paper, we re-formulate this problem of searching for maximal right eigenvector as a constrained optimization problem:

$$\begin{aligned} & \min: f(\mathbf{x}) := \frac{1}{2}\left\|\bar{B}\mathbf{x} - \mathbf{x}\right\|^2 \\ & \quad s.t.\ \mathbf{1}^T\mathbf{x} = 1,\ \mathbf{x} \ge 0. \qquad (\mathfrak{H}) \end{aligned} \tag{13}$$

After expanding the objective function, we obtain $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T M\mathbf{x}$, where $M = \bar{B}^T\bar{B} - \bar{B} - \bar{B}^T + I$. Therefore, the equivalent optimization problem is

$$\begin{aligned} & \min: f(\mathbf{x}) := \frac{1}{2}\mathbf{x}^T M\mathbf{x} \\ & \quad s.t.\ \mathbf{1}^T\mathbf{x} = 1,\ \mathbf{x} \ge 0. \qquad (\mathfrak{H}) \end{aligned} \tag{14}$$

The gradient of $f(\mathbf{x})$ can be easily computed $\nabla f(\mathbf{x}) = M\mathbf{x}$. Furthermore, we find that the Hessian matrix of $f(\mathbf{x})$ is M, which is a positive semi-definite matrix proven by Lemma 2:

Lemma 2. $M = \bar{B}^T\bar{B} - \bar{B} - \bar{B}^T + I$ is positive semi-definite.

Proof. M can be written as $M = (\bar{B} - I)^T(\bar{B} - I)$, which proves the lemma. □

With Lemma 2, it is obvious that the objective function $f(\mathbf{x})$ is convex. Also, the constraint set $\mathfrak{H} = \{\mathbf{x}|\mathbf{x}^T\mathbf{1} = 1, \mathbf{x} \geq 0\}$ is a unit simplex, which is convex and compact. Hence, the IsoRank problem (13) has the same problem structure as (1) and our generalized SBCFW algorithm can be used to solve (14) with much better scalability and efficiency due to the efficiency of the randomized partial gradient computation at each iteration. Similarly as in [7], in addition to network topology, we can incorporate other information in the formulation for more biologically significant alignment results by replacing $\bar{B}$ with $\hat{B} = \alpha\bar{B}+(1-\alpha)\bar{S}\mathbf{1}^T$, $\alpha \in [0, 1]$. Here, $\bar{S} = S/|S|$ is a normalized similarity vector with size $N_a \times N_b$, cancatenated from the doubly indexed similarity estimates $S([u, v])$ based on the sequence or function similarity between vertices u in G_a and v in G_b.

3.2 SBCFW-IsoRank algorithm

As shown in Section 3.1, $f(\mathbf{x})$ in (13) is convex and the constraint set $\mathfrak{H}$ in (14) is a convex compact set. Therefore, we can apply the generalized SBCFW algorithm proposed in Section 2 to solve the corresponding optimization problem (14). The detailed algorithm is illustrated in Algorithm 2. We define $E = \bar{B} - I$. From Lemma 2, we know $M = (\bar{B} - I)^T(\bar{B} - I)$, and therefore, we can write $M = E^TE$. Here, we want to emphasize that, in each iteration of our SBCFW-IsoRank algorithm, both the time and space complexity are $O\left(\frac{N^2}{n}\right)$, which is achieved through tracking the vectors of $\mathbf{p}_k = E\mathbf{x}^k$ and $\mathbf{q}_k = E\mathbf{s}_i^k$ at steps 2 and 10 of each iteration in Algorithm 2, respectively. The stopping criterion is $\left\|\bar{B}\mathbf{x} - \mathbf{x}\right\| \leq \xi \|\mathbf{x}\|$, which can be efficiently estimated by

$$\left\|\bar{B}\mathbf{x} - \mathbf{x}\right\| = \mathbf{x}^TM\mathbf{x} = (E\mathbf{x})^TE\mathbf{x} = \mathbf{p}_k^T\mathbf{p}_k, \tag{15}$$

which is taken in line 11 in the SBCFW-IsoRank algorithm.

3.3 Initialization

In order to guarantee both the time and space complexity to be $O\left(\frac{N^2}{n}\right)$ at each iteration, we cannot initialize the algorithm with randomly generated $\mathbf{x}^0$ to avoid a multiplication of a matrix of size $N \times N$ and a vector of size N, whose time and space complexity would be $O(N^2)$. We propose to initialize $\mathbf{x}^0$ in the following way: First, randomly divide $\mathbf{R}^N$ into n parts with equal sizes and randomly pick the ith part. Then, we initialize every elements in the ith part with $\frac{n}{N}$, which makes $\mathbf{x}^0$ in the feasible space defined by the constraint set $\mathfrak{H}$. Using the above initialization strategy, the time and space complexity for computating $\nabla_i f(\mathbf{x}^0)$, $\mathbf{p}_0 = E\mathbf{x}^0$, and $\mathbf{q}_0 = E\mathbf{s}^0$ are all under $O\left(\frac{N^2}{n}\right)$, which is easy to verify.

Algorithm 2 SBCFW-IsoRank algorithm

Input: ξ, n and E
1 **For** $k = 0, ..., \infty$ **do**
2 Randomly divide $\mathbf{R}^N$ into n equal-size parts
3 Choose $i = \mathfrak{C}_k$
4 **If**($k == 0$)
5 Initialize the ith block of $\mathbf{x}^0$ with $\frac{n}{N}$
6 **EndIf**
7 Compute $\mathbf{p}_k = E\mathbf{x}^k$ and $\nabla_i f(\mathbf{x}^k) = [E^T]_i \mathbf{p}_k$
8 Solve the sub-problem:
9 $\mathbf{s}_i^k := arg \min_{U_j \times \mathbf{s} = U_j \times \mathbf{x}^k, \forall j \neq i; \mathbf{s} \in \mathfrak{H},} \nabla_i f(\mathbf{x}^k)^T(\mathbf{s} - \mathbf{x}^k)$
10 Compute $\mathbf{q}_k = E\mathbf{s}_i^k$
11 **If** $\mathbf{p}_k^T\mathbf{p}_k < \xi \|\mathbf{x}\|$
12 **Break**;
13 **EndIf**
14 Compute the step size γ_k^*:
15 $\gamma_k^* = \begin{cases} \min\{\hat{\gamma}, 1\} & \hat{\gamma} > 0, \hat{\gamma} = \frac{\mathbf{p}_k^T\mathbf{p}_k - \mathbf{p}_k^T\mathbf{q}_k}{\mathbf{p}_k^T\mathbf{p}_k - 2\mathbf{p}_k^T\mathbf{q}_k + \mathbf{q}_k^T\mathbf{q}_k} \\ 0 & o.w. \end{cases}$
16 $\mathbf{x}^{k+1} = \mathbf{x}^k + \gamma^k(\mathbf{s}_i^k - \mathbf{x}^k)$
17 **EndFor**
Output: $\mathbf{x}^k$

3.4 Algorithm to solve the subproblem

As shown in the SBCFW-IsoRank algorithm, at each iteration, we need to solve a subproblem. Fortunately, the subproblem can be solved in a straightforward manner for the optimization problem (14). For the following subproblem at iteration k:

$$\begin{aligned} &\min: \nabla_i f\left(\mathbf{x}^k\right)^T\left(\mathbf{s} - \mathbf{x}^k\right) \\ &s.t.\ \mathbf{s} \in \mathfrak{H}, \\ &\quad U_j \times \mathbf{s} = U_j \times \mathbf{x}^k,\ \forall j \neq i, \end{aligned} \tag{16}$$

the optimal solution is $\mathbf{s}^* = \mathbf{x}^k - U_i\mathbf{x}^k + L\mathbf{e}_j$, where $\mathbf{e}_j$ is an all-zero vector except that the jth element is 1 and $L = \sum_{l \in R^{N_i}} \mathbf{x}^k(l)$. Here, j is the index of the coordinate with the smallest value in the ith block of $\nabla_i f\left(\mathbf{x}^k\right)$:

$$j = \underset{l \in \mathbf{R}^{N_i}}{\operatorname{argmin}}: \left[\nabla_i f\left(\mathbf{x}^k\right)\right](l). \tag{17}$$

3.5 Optimal step size

To obtain the optimal step size at each iteration, we need to solve the following optimization problem:

$$\begin{aligned} &\min: \left(\mathbf{x}^k + \gamma\left(\mathbf{s}^k - \mathbf{x}^k\right)\right)^T M\left(\mathbf{x}^k + \gamma\left(\mathbf{s}^k - \mathbf{x}^k\right)\right) \\ &s.t.\ 0 \leq \gamma \leq 1, \end{aligned} \tag{18}$$

which is the classic quadratic form with respect to γ. If $\hat{\gamma} = \frac{\mathbf{p}_k^T \mathbf{p}_k - \mathbf{p}_k^T \mathbf{q}_k}{\mathbf{p}_k^T \mathbf{p}_k - 2\mathbf{p}_k^T \mathbf{q}_k + \mathbf{q}_k^T \mathbf{q}_k} > 0$, which is the solution to (18) without any constraints, the optimal solution γ^* is the minimum value between 1 and $\hat{\gamma}$, otherwise $\gamma^* = 0$. The definitions of $\mathbf{p}_k$ and $\mathbf{q}_k$ are given in lines 7 and 10 in Algorithm 2.

3.6 Time and space complexity

At each iteration, the most computationally expensive operations are the updates of $\mathbf{p}_k$ and $\mathbf{q}_k$ (lines 7 and 10 of SBCFW-IsoRank) and the calculation of the partial gradient $\nabla_i f\left(\mathbf{x}^k\right)$ (line 7 of SBCFW-IsoRank).

The calculation of $\mathbf{p}_k$ and $\mathbf{q}_k$ are similar. From line 10 of Algorithm 2, we know

$$\begin{aligned} \mathbf{p}_k &= E\mathbf{x}^k \\ &= E\left(\mathbf{x}^{k-1} + \gamma^{k-1}\left(\mathbf{s}^{k-1} - \mathbf{x}^{k-1}\right)\right) \\ &= \mathbf{p}_{k-1} + \gamma^{k-1} E\left(\mathbf{s}^{k-1} - \mathbf{x}^{k-1}\right). \end{aligned} \tag{19}$$

The second equation is derived by replacing $\mathbf{x}^k$ with the equation in line 16 of our SBCFW-IsoRank algorithm. Because we keep tracking $\mathbf{p}_k$ at each iteration, we do not need to recompute $\mathbf{p}_{k-1}$. Therefore, we only need to compute $E\left(\mathbf{s}^{k-1} - \mathbf{x}^{k-1}\right)$, which takes $O\left(\frac{N^2}{n}\right)$ operations because $\left(\mathbf{s}^{k-1} - \mathbf{x}^{k-1}\right)$ is a vector, with only its ith block being non-zeros and all the other parts are zeros. Additionally, the memory consumption is also $O\left(\frac{N^2}{n}\right)$ by the similar argument. Similarly, we can compute $\mathbf{q}_k$:

$$\begin{aligned} \mathbf{q}_k &= E\mathbf{s}^k \\ &= E\left(\mathbf{x}^k + \left(L\mathbf{e}_j - U_i\mathbf{x}^k\right)\right) \\ &= \mathbf{p}_k + E\left(L\mathbf{e}_j - U_i\mathbf{x}^k\right), \end{aligned} \tag{20}$$

where $\left(L\mathbf{e}_j - U_i\mathbf{x}^k\right)$ is also a vector with only the ith block having non-zero values. Therefore, the computation of $\mathbf{q}_k$ also takes $O\left(\frac{N^2}{n}\right)$ operations and consumes $O\left(\frac{N^2}{n}\right)$ memory.

The equation of calculating $\nabla_i f\left(\mathbf{x}^k\right)$ is as follows:

$$\nabla_i f\left(\mathbf{x}^k\right) = \left[E^T\right]_i \mathbf{p}_k, \tag{21}$$

where the operator $[\,\cdot]_i$ is to get the rows of the matrix corresponding to the ith coordinate block. Hence, it is easy to verify that the time complexity and space complexity of computing $\nabla_i f\left(\mathbf{x}^k\right)$ are $O\left(\frac{N^2}{n}\right)$.

In summary, based on the above analyses, both the time complexity and space complexity of our SBCFW-IsoRank at each iteration are $O\left(\frac{N^2}{n}\right)$.

4 Experiments

In this section, we apply our SBCFW-IsoRank algorithm to two network query problems. For the first set of experiments, we take a known protein complex in an archived yeast PPI network in one database [11] as the query to search for the subnetwork in another yeast PPI network [12] with different archived interactions. We call this yeast-yeast network query problem. The goal of this set of experiments is to check the correctness of our algorithm as we have the ground truth for the target subnetwork. With that, we aim to test the convergence property of our algorithm with different partitions and the relationship between the number of iteration steps and number of partitions. The second experiment is to query a large-scale yeast PPI network in IntAct [13] to find similar subnetworks of proteins with similar cellular functionalities for a known protein complex in human PPI network. The aim of this experiment is to show that our new algorithm can help transfer biology knowledge from model organisms to study potential functionalities of molecules in different organisms. Our Matlab implementation of SBCFW-IsoRank runs on a MacBook Pro notebook with 8 GB RAM. We do not compare with IsoRank algorithm in terms of computational and biological performance. Computationally, it is not fair to compare our algorithm (implemented in matlab) with IsoRank (implemented in C). Biologically, they should generate the same results if our algorithm converges, so the proof of convergence is sufficient to guarantee the equivalence of the biological performance.

4.1 Yeast-yeast network query problem

We test our SBCFW-IsoRank algorithm on the yeast-yeast PPI network query problem by solving the optimization problem introduced in the previous section. We take a subnetwork with six proteins (Fig. 1a) from Krogan's yeast PPI network [11] as the query example to search for the conserved functional complex in a target network, which is Collins' network [12] with 1622 proteins and 9074 interactions. The query subnetwork is the transcription factor TFIIIC complex in Krogan's network and we are interested in testing whether we can find the same subnetwork in Collins' network. The dimension of our optimization problem is $6 \times 1622 = 9732$. We run this preliminary example so that we can compare our stochastic optimization results with the results from the power method, which is typically done in the original IsoRank algorithm [7]. Theoretically, the time and space complexity of our SBCFW-IsoRank at each iteration are both $O(N^2/n)$ based on the analysis in Section 3.6. Compared to $O(N^2)$ time and space complexity for the power method by IsoRank [7], our SBCFW-IsoRank algorithm can scale better with the properly selected number of partitions n at each iteraction.

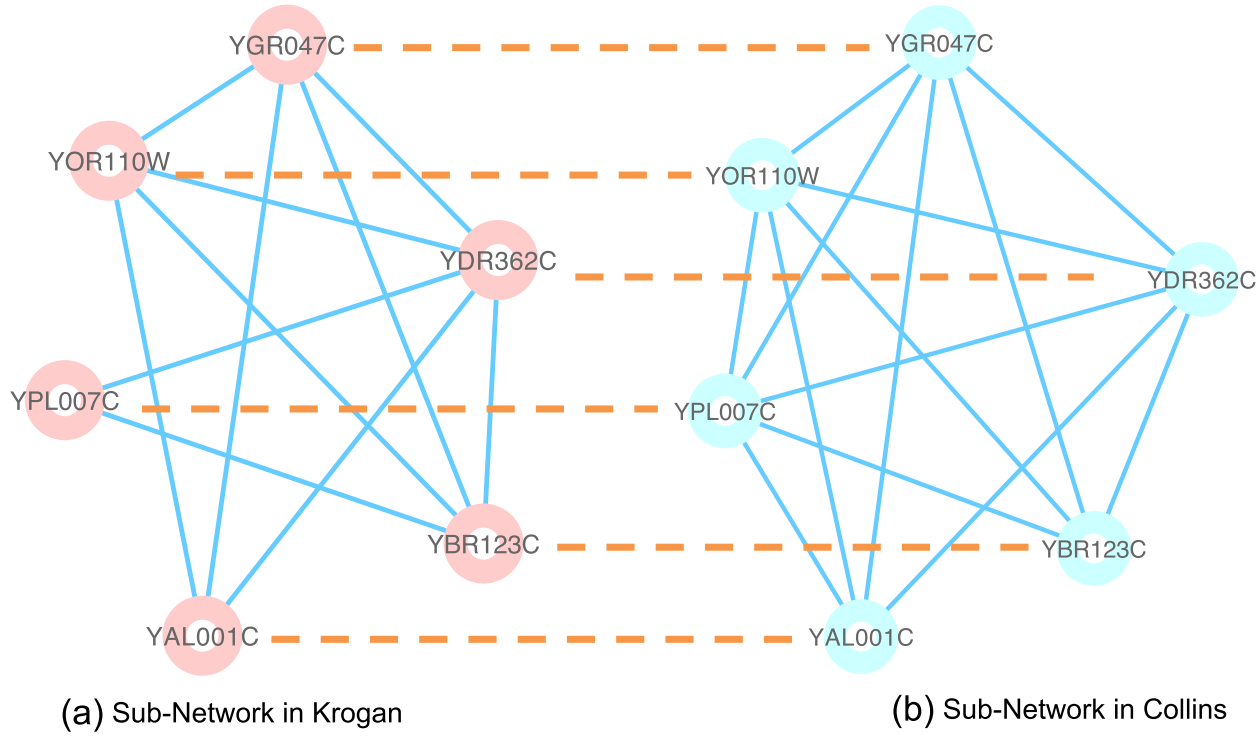

Fig. 1 Query subnetwork and its aligned result in the target network. **a** The query subnetwork in Krogan's yeast PPI network [11]. **b** The aligned result in Collins' yeast PPI network [12]

As both the query example and the target network contain interactions among proteins from the same organism—yeast, we can easily check the correctness of the query result. We define the accuracy as the number of corrected aligned proteins divided by the total number of proteins in the query subnetwork. We implement the SBCFW-IsoRank algorithm for different numbers of partitions n but use the same stopping criterion: $\left\|\hat{B}\mathbf{x}-\mathbf{x}\right\| \leq \xi \|\mathbf{x}\|, \xi = 0.1$ [9].

Figure 2 shows the changes of the objective function values with respect to the increasing number of iterations. As illustrated in Fig. 2, our algorithm converges for all different n values. Additionally, we find that, the larger the number of partitions n is, the larger the number of

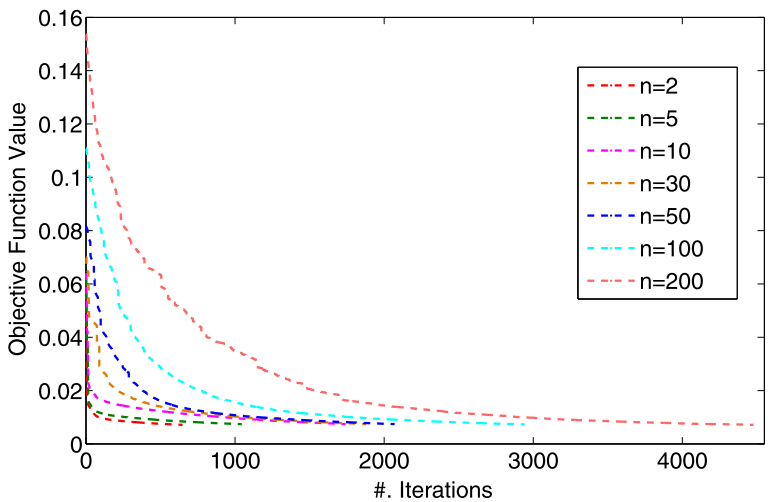

Fig. 2 The change of the objective function values with the increasing number of iterations with different numbers of partitions

iterations we need to have the algorithm converge to the global optimum with the same stopping criterion. This clearly demonstrates the tradeoff between the efficiency and scalability of the stochastic optimization algorithms. Interestingly, we notice that for $n = 10, 30$, and 50, the number of iterations does not increase much, which indicates that we may achieve fast computation with a reasonably large n because our algorithm is more efficient for larger n values at each iteration.

To further investigate the performance with different n values, we run our algorithm 10 times for $n = [2, 10, 20, 30, \ldots, 200]$ and show the average and standard deviation of the computational time as well as the average and standard deviation of the number of iterations in Fig. 3.

We find that for all different n values, our algorithm can obtain 100 % accuracy, which again demonstrates the effectiveness and convergence of our generalized SBCFW algorithm. Also, we notice that with the increasing n, the number of iterations increase; however, the computational time is first reducing then increasing. For example, when $n = 2$, our algorithm converges with the smallest number of iterations, but the computational time is not the best because at each iteration, the algorithm takes $O\left(\frac{N^2}{2}\right)$ operations. In contrast, when $n = 30$, the number of iterations is larger; but it reaches the global optimum with the least computation time, which is indeed twice faster than $n = 2$. The trend of the computational time implies that there may exist the best number of partitions n^*. Empirically, when $n < n^*$, the computational time decreases while the computational time can increase when $n > n^*$. However, it is difficult to provide a theoretical proof for this observed phenomenon. Finally, for the scalability of the algorithm, we always prefer larger n to make the memory requirement as low as possible.

4.2 Human-yeast network query problem

We further study the biological signficance of network query results by our SBCFW-IsoRank algorithm. We extract a subnetwork as a query example from a human PPI network archived in IntAct [13]. The query subnetwork is the proteasome core complex, with induced interactions among the corresponding proteins from IntAct [13]. The proteasome core complex in human consists of 14 proteins in total, as shown in Fig. 4a. The target network is the yeast PPI network, also obtained from IntAct [13], which has 6392 proteins and 77,065 interactions. Our goal is to find the most similar subnetwork to the human proteasome core complex in the target yeast PPI network, based on both the interaction topology and the protein sequence similarity, which is computed by BLAST [14].

We first construct the alignment network, which has $N = 14 \times 6,392 = 89,488$ vertices. By our SBCFW-IsoRank algorithm with $n = 300$, at each iteration, instead of operating a matrix of size $89,488 \times 89,488$ by the power method, we only need to handle a matrix of size $298 \times 89,488$. The computational time as well as the memory requirement are reduced 300 times. Our Matlab implementation of SBCFW-IsoRank on the MacBook Pro notebook with 8GB RAM takes only around 750 s to converge by reaching the stopping criteria (0.1).

The identified subnetwork in the target yeast PPI network by our algorithm is illustrated in Fig. 4b. To evaluate the biological significance of the obtained subnetwork, we check the p value based on Gene Ontology (GO) enrichment analysis using GOTerm Finder [15].

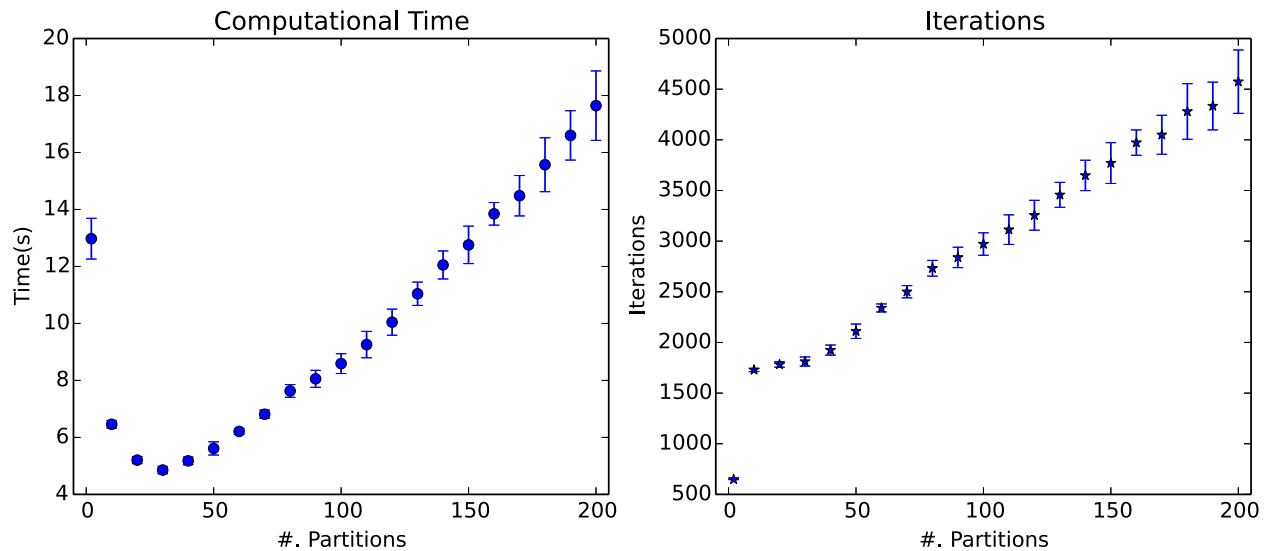

Fig. 3 Mean and standard deviation of the computational time in seconds and the number of iterations for different partition numbers

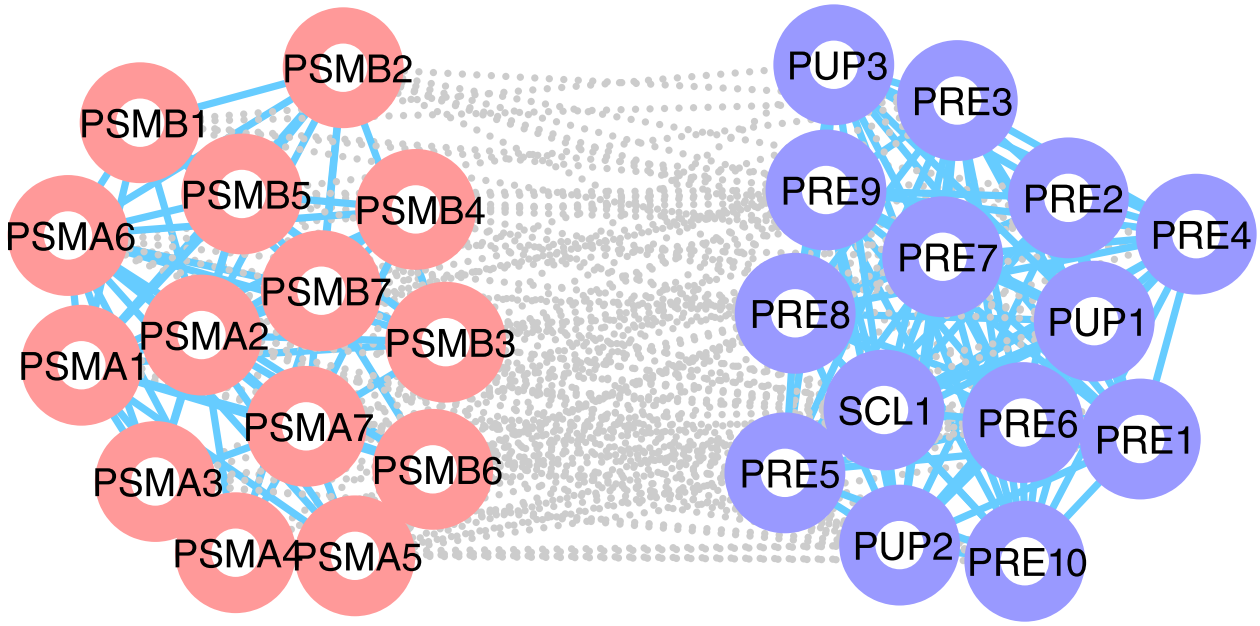

Fig. 4 Querying human protein complex in a yeast PPI network. The proteins are annotated by their gene names. The *solid lines* are protein interactions and the *dashed lines* denote orthologous relationships based on protein sequence similarity by BLAST between the proteins in different organisms. **a** Human proteasome core complex. **b** The aligned proteasome core complex in yeast found by SBCFW-IsoRank

The identified subnetwork is significantly enriched in GO term GO:0005839, which is in fact the same proteasome core complex, with p value $9.552e-36$. This experiment demonstrates that our algorithm can find the biologically consistent groups of proteins with the same cellular functionalities as the proteins in the query subnetwork, hence with the capability of transferring existing biology knowledge in model organisms (yeast for example) to less studied organisms when the group of proteins in the query subnetwork require better understanding of their cellular functionalities.

5 Conclusions

In this paper, we generalize the block coordinate Frank-Wolfe algorithm to solve general convex optimization problems with any convex and compact constraint set. Our generalized SBCFW algorithm has the convergence guarantee. We re-formulate the IsoRank problem to such a convex programming problem and solve the biological network alignment problem by our SBCFW-IsoRank algorithm, which scales better with the size of networks under study. The scalability, efficiency, and effectiveness of our algorithm on solving IsoRank are demonstrated for real-world PPI network query problems. In our future work, we will consider the derivation of the optimal partition number for better tradeoff between computational efficiency and scalability and generalize the derived SBCFW algorithm to solve other optimization problems when analyzing biological networks.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

YW and XQ conceived the algorithm. YW implemented the algorithm and performed the experiments. YW and XQ analyzed the results. YW and XQ wrote the paper. Both authors read and approved the final manuscript.

Acknowledgements

The authors would like to thank Simon Lacoste-Julien for pointing out the error in the original conference paper. This work was partially supported by Awards #1447235 and #1244068 from the National Science Foundation, as well as Award R21DK092845 from the National Institute Of Diabetes And Digestive And Kidney Diseases, National Institutes of Health.

Received: 24 August 2015 Accepted: 17 March 2016
Published online: 08 April 2016

References

1. M Zaslavskiy, F Bach, J Vert, Global alignment of protein-protein interaction networks by graph matching methods. Bioinformatics. **25**, 259–267 (2009)
2. G Klau, A new graph-based method for pairwise global network alignment. BMC Bioinformatics. **10**(Suppl 1), 59 (2009)
3. M Frank, P Wolfe, An algorithm for quadratic programming. Nav. Res. Logist Q. **3**, 95–110 (1956)

4. J Dunn, Convergence rates for conditional gradient sequences generated by implicit step length rules. SIAM J. Control Optim. **5**, 473–487 (1980)
5. J Dunn, S Harshbarger, Conditional gradient algorithms with open loop step size rules. J. Math. Analy. Appl. **62**, 432–444 (1978)
6. S Lacoste-Julien, M Jaggi, M Schmidt, P Pletscher, in *International Conference on Machine Learning*. Block-coordinate Frank-Wolfe optimization for structural SVMs, (2013)
7. R Singh, J Xu, B Berger, Global alignment of multiple protein interaction networks with application to functional orthology detection. Proc. Natl Acad. Sci. **105**, 12763–12768 (2008)
8. S Uryasev, PM Pardalos (eds.), *Stochastic optimization: algorithm and application vol. 54* (Springer, University of Florida, 2001)
9. Y Nesterov, Efficiency of coordinate descent methods on huge-scale optimization problems. SIAM J. Optim. **22**(2), 341–362 (2012)
10. JM Ortega, WC Rheinbold, *Iterative solution of nonlinear equations in several variables*. (Society for Industrial and Applied Mathematics, 1970)
11. N Krogan, et al., Global landscape of protein complexes in the yeast Saccharomyces cerevisiae. Nature. **440**, 4412–4415 (2006)
12. J Hasty, D McMillen, F Issacs, J Collins, Computational studies of gene regulatory networks: in numero molecular biology. Nat. Rev. Genet. **2**, 268–279 (2001)
13. S Kerrien, B Aranda, L Breuza, et al., The intact molecular interaction database in 2012. Nucleic Acids Res. **40**(D1), 841–846 (2012)
14. Z Zhang, S Schwartz, L Wagner, W Miller, A greedy algorithm for aligning DNA sequences. J. Comput. Biol. **7**(1–2), 203–214 (2000)
15. E Boyle, I Elizabeth, S Weng, J Gollub, H Jin, D Botstein, JM Cherry, G Sherlock, GO::TermFinder—open source software for accessing Gene Ontology information and finding significantly enriched Gene Ontology terms associated with a list of genes. Bioinformatics. **20**, 3710–3715 (2004)