Block-Coordinate Frank-Wolfe Optimization for Counting Objects in Images

Fei Xia*
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
xia.fei09@gmail.com

Shanghang Zhang*
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
shanghaz@andrew.cmu.edu

Abstract

We develop an optimization method to count objects in images. To avoid the detection of individual objects, which is computationally expensive and relies heavily on image quality, we model the object density as a linear transformation of each pixel feature, and obtain the object count by integrating density over the image. Learning such linear transformation is formulated as the minimization of a regularized quadratic function. Solving this optimization problem is highly nontrivial because it has exponentially large number of constraints. To cope with this challenge, inspired by the structural support vector machine (SVM), we explore the Block-Coordinate Frank-Wolfe (BCFW) algorithm, which is a state-of-the-art algorithm to solve structural SVM. However, BCFW cannot be directly applied to our problem. We derive the dual of our optimization problem and solve it by BCFW with modifications. Experiments show that BCFW solves our problem with lower iteration cost, faster convergence, and decent error rate.

1 Introduction

Learning to count objects in images is an important machine learning problem, forming an essential part of many real-world applications, including counting cells in microscopic images, monitoring pedestrian in surveillance system, and counting vehicles to estimate traffic flow. For objects counting, two types of approaches have been proposed in literature [7]: (1) Counting by detection [13, 2, 5], and (2) Counting by regression [6, 4, 8, 3]. The first approach attempts to detect individual object in the image and then count the number of detected objects. This approach is quite sensitive to the image quality and computationally expensive. To address these shortcomings, the second approach attempts to directly map the global image characteristics to the number of objects [18]. However, this approach has to discard object location information.

To overcome the limitation of existing methods, we model object density as a linear transformation of each pixel feature, and obtain the object count by integrating object density over the image [14]. Learning to infer such linear transformation is formulated as minimization of a regularized quadratic function. To reflect spatial information of image and avoid influence of zero-mean noise, we apply the Maximum Excess over SubArrays (MESA) distance measure [14]. Such distance induces exponentially large number of constraints, making the optimization problem very challenging to solve. Observing the similarity between our formulation and structural SVM, we explore BCFW [10] algorithm to solve our problem. BCFW is a state-of-the-art algorithm to solve structural SVM, but it cannot be directly applied to our problem. We derive the dual of our optimization problem and solve it by BCFW with modifications. Compared with Frank-Wolfe algorithm [1, 9, 16], BCFW only requires a single call of maximization oracle, with the same convergence rate $O(1/\varepsilon)$. Although the stochastic subgradient [17, 19] and BCFW have a similar iteration cost, BCFW has several important advantages: (1) the optimal step-size can be efficiently computed in closed-form; (2) the algorithm yields a duality gap guarantee, and the duality gap can be computed as a proper stopping criterion. (3) the convergence rate holds even when using approximate maximization oracles [10].

*Authors contributed equally to this work.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

2 Problem Formalization

We try to estimate a density function F so that its integral over the whole image is equal to the number of objects in it [14]. For an image I_i, each pixel p in an image can be represented by a feature vector x_p^i, so that we can model the density function as a linear transformation of x_p^i: $F_i(p) = w^\top x_p^i$. Given a set of training images, the parameter vector w is learnt in the regularized risk framework to match the estimated densities with the ground truth densities for the training images (under regularization of w). Each training image I_i is annotated with a set of 2D points $\mathcal{P}_i = \{P_1, P_2, ..., P_{c(i)}\}$, where P_k is the center of the object and $c(i)$ is the number of objects annotated by users. We define the ground truth density function for training image I_i to be a kernel density estimate based on the provided 2D points: $F_i^0(p) = \sum_{P \in \mathcal{P}_i} \mathcal{N}(p; P, \sigma^2 \mathbf{1}_{2\times 2})$, where $\mathcal{N}(p; P, \sigma^2 \mathbf{1}_{2\times 2})$ denotes a normalized Gaussian kernel evaluated at pixel p. One critical question is how to define the loss between the ground truth and estimated density. Here we apply MESA distance [14]: $\mathcal{D}_{\text{MESA}}(F_1, F_2) = \max_{B\in\mathcal{B}} |\sum_{p\in B} F_1(p) - \sum_{p\in B} F_2(p)|$, where $\mathcal{B}$ is the set of all box subarrays in I, and $\mathcal{D}_{\text{MESA}}$ is defined as the largest absolute difference between sums of $F_1(p)$ and $F_2(p)$ over all box subarrays. To learn w, we need to solve the following optimization problem:

$$\min_{w} \quad \frac{1}{2} w^\top w + C \sum_{i=1}^{N} \max_{B\in\mathcal{B}_i} |\sum_{p\in B} F_i^0(p) - \sum_{p\in B} w^\top x_p^i| \tag{1}$$

This is a convex quadratic problem. It encourages w to minimize the sum of mismatch between ground truth and estimated density under regularization. By introducing slack variables $\xi = \{\xi_1, \xi_2, ..., \xi_N\}$, (1) can be rewritten as:

$$\begin{aligned} \min_{w,\boldsymbol{\xi}} \quad & \frac{1}{2} w^\top w + C \sum_{i=1}^{N} \xi_i \\ \text{s.t.} \quad & \forall i, \forall B \in \mathcal{B}_i : \; \xi_i \geq \sum_{p\in B} (F_i^0(p) - w^\top x_p^i), \; \xi_i \geq \sum_{p\in B} (w^\top x_p^i - F_i^0(p)) \end{aligned} \tag{2}$$

The most challenging part of this problem is to tackle a large number of constraints. We give our analysis and solution in the next section.

3 Optimization

Problem (2) is actually not trivial to solve — $\mathcal{B}_i$ is the set of all sub-boxes in image i, so $|\mathcal{B}_i|$ is combinatorial and the number of constraints in (2) is very large. To solve this problem, we are inspired by the Structural SVM, which also has a large number of constraints. This observation leads us to explore methods or frameworks that can be used to solve Structural SVM. Specifically, we investigate Block-Coordinate Frank-Wolfe (BCFW), which is a state-of-the-art algorithm for Structural SVM, yet cannot be directly applied to our problem. To develop the BCFW algorithm for object counting, we need to introduce an important concept: Max Oracle, which is defined as $\hat{H}_{iB} = \max_{B\in\mathcal{B}_i} H_{iB}$, where $H_{iB} = \sum_{p\in B} (w^\top x_p^i - F_i^0(p))$. With this definition, the constraints in (2) can thus be written as $\xi_i \geq -\hat{H}_{iB}$ and $\xi_i \geq \hat{H}_{iB}$. Here in our problem, the Max Oracle is actually a 2D maximum subarray problem in an image and can be solved in polynomial time. Solving the Max Oracle is to find out the most violated constraints in (2).

3.1 Dual Problem

We now derive the dual of our optimization problem and show it can be solved with BCFW with some modifications. BCFW targets the problem with the following form:

$$\min_{\gamma\in\mathcal{M}^{(1)}\times ...\times \mathcal{M}^{(n)}} f(\gamma) \tag{3}$$

where the domain has structure of a Cartesian product $\mathcal{M} = \mathcal{M}^{(1)} \times ... \times \mathcal{M}^{(n)} \subseteq \mathbb{R}^m$ over $n \geq 1$ blocks. It is important that the constraint set can be decomposed, since the idea of BCFW is to randomly select a block $i \in \{1, ..., n\}$, and then perform an iteration of the Frank-Wolfe algorithm applied to just one of the blocks. The dual problem of (2) is derived as (derivation in appendix)

Algorithm 1 Block-Coordinate Primal-Dual Frank-Wolfe Algorithm for Problem (2)

Let $w^{(0)} = w_i^{(0)} = \bar{w} = 0, \ell^{(0)} = \ell_i^{(0)} = \bar{\ell} = 0$
for $k \leftarrow 0$ to K **do**
  Pick i at random in $\{1, ..., n\}$
  Solve $B_i^* = \text{argmax}_{B \in \mathcal{B}_i} \{H_{iB}, -H_{iB}\}$. Set $p = 1$ if the optimal value is $H_{iB_i^*}$, set $p = -1$ otherwise
  Let $w_s = p \cdot C \sum_{p \in B_i^*} x_p^i$ and $\ell_s = p \cdot C \sum_{p \in B_i^*} F_i^0(p)$
  Do line search $t = \frac{(w_i^{(k)} - w_s)^\top w^{(k)} - \ell_i^{(k)} + \ell_s}{\|w_i^{(k)} - w_s\|_2^2}$ and clip it to [0,1]
  Update $w_i^{(k+1)} = (1-t) w_i^{(k)} + t w_s, \;\; \ell_i^{(k+1)} = (1-t)\ell_i^{(k)} + t\ell_s$
  Update $w^{(k+1)} = w^{(k)} + w_i^{(k+1)} - w_i^{(k)}, \ell^{(k+1)} = \ell^{(k)} + \ell_i^{(k+1)} - \ell_i^{(k)}$
  (Optionally: Update $\bar{w}^{(k+1)} = \frac{k}{k+2}\bar{w}^{(k)} + \frac{2}{k+2} w^{(k+1)}$)

$$\begin{aligned} \min_{\alpha,\beta} \quad & \frac{1}{2}\|A(\alpha - \beta)\|_2^2 - b^\top(\alpha - \beta) \\ \text{s.t.} \quad & \forall i : \sum_{B \in \mathcal{B}_i} \alpha_{iB} + \beta_{iB} = 1 \qquad \alpha_{iB}, \beta_{iB} \geq 0 \end{aligned}$$

where $A \in \mathbb{R}^{d \times m}$ and $A = \{C \sum_{p \in B} x_p^i | i \in [n], B \in \mathcal{B}_i\}$, $b = (C \sum_{p \in B} x_p^i)_{i \in [n], B \in \mathcal{B}_i}$, $\alpha = (\alpha_{iB})_{i \in [n], B \in \mathcal{B}_i}$, $\beta = (\beta_{iB})_{i \in [n], B \in \mathcal{B}_i}$ and d is the feature dimension, $m = \sum_i^N |\mathcal{B}_i|$. To meet our target form (3), we further let $X = (A, -A), c = (b^\top, -b^\top)^\top, \gamma = (\alpha^\top, \beta^\top)^\top$, the optimization problem can be transformed into:

$$\begin{aligned} \min_{\gamma} \quad & \frac{1}{2}\|X\gamma\|_2^2 - c^\top \gamma \\ \text{s.t.} \quad & \forall i : \sum_{g=1}^{2|\mathcal{B}_i|} \gamma_{ig} = 1 \qquad \gamma_i \geq 0 \end{aligned} \tag{4}$$

where $\gamma_i := (\alpha_i^\top, \beta_i^\top)^\top$, note that γ_i takes nonconsecutive blocks from γ. This has the same form as (3). We further derive the specific BCFW solver for our problem in the following section.

3.2 A Block-Coordinate Frank-Wolfe (BCFW) Solver

Now we derive the BCFW solver. In Frank-Wolfe algorithm, the major step is to compute $s := \text{argmin}_{s' \in \mathcal{M}} \langle s', \nabla f(\gamma) \rangle$, where $\nabla f(\gamma)$ is the gradient of $f(\gamma)$. Our main observation is Lemma 1[2].

Lemma 1. *For each data point, solving* $s = \text{argmin}_{s' \in \mathcal{M}} \langle s', \nabla f(\gamma) \rangle$ *is equivalent to solving* $B_i^* = \text{argmax}_{B \in \mathcal{B}_i} \{H_{iB}, -H_{iB}\}$

With Lemma 1 in mind, we can get the Primal-Dual BCFW algorithm by only maintaining the corresponding primal iterates. Please refer to Alg. 1. The convergence rate is proved to be $O(1/\epsilon)$ [12]. Compared with Cutting Plane method, the most important advantage of BCFW is that in each iteration, it doesn't need to solve another quadratic programming problem, which saves a lot of time.

Maintaining Primal iterates. In the Frank-Wolfe Algorithm, the corresponding primal iterates $w^{(k)} = X\gamma^{(k)}$. With Frank-Wolfe search corner s, this yields $w_s = Xs$. Since only one element in s is 1, others are 0, w_s in Frank-Wolfe Algorithm can be written as:

$$w_s = \begin{cases} \sum_{i=1}^N C \sum_{p \in B_i^*} x_p^i, & \text{if } \max\{H_{iB}, -H_{iB}\} = H_{iB_i^*} \\ -\sum_{i=1}^N C \sum_{p \in B_i^*} x_p^i, & \text{if } \max\{H_{iB}, -H_{iB}\} = -H_{iB_i^*} \end{cases}$$

Then for BCFW,

$$w_s = \begin{cases} C \sum_{p \in B_i^*} x_p^i, & \text{if } \max\{H_{iB}, -H_{iB}\} = H_{iB_i^*} \\ -C \sum_{p \in B_i^*} x_p^i, & \text{if } \max\{H_{iB}, -H_{iB}\} = -H_{iB_i^*} \end{cases}$$

[2]Due to space limit, the proof is in the appendix.

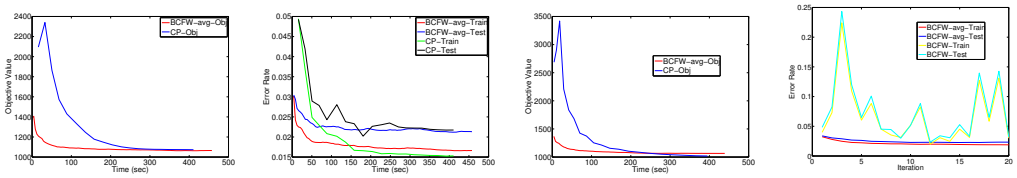

Figure 1: From left to right: (a) Objective value comparison between BCFW and CP with inner active set; (b) Error rate comparison between BCFW and CP with inner active set; (c) Objective value comparison between BCFW and CP with inner interior point; (d) Error rate comparison between BCFW with weighted average $\bar{w}$ and BCFW without weighted average w.

Line Search. The optimal step-size can be obtained analytically because of the quadratic form of the dual problem. Specifically, $t_{opt} = \text{argmin}_{t\in[0,1]} f(\gamma + t(s - \gamma))$.

Weighted Averaged of w. Optionally, we maintain a weighted average version of w, i.e. $\bar{w}$. It turns out that $\bar{w}$ is more stable than w in the optimization process. We'll show experiments on this later.

4 Experiments

In this section, we evaluate the proposed algorithm on a cell counting task on microscopy images, and compare its performance with cutting plane method [11].

Experiment Setup. We have 200 microscopy images, each of which has the total number of cells varying between 74 and 317. We extract SIFT feature [15] of each pixel, and then use a codebook of $K = 256$ entries constructed via k-means on SIFT descriptors extracted from the images. We used 160 images for training, while another 40 images for testing. Each object is labeled as a dot in the training image. The ground truth density is computed as the normalized 2D Gaussian kernel estimate based on the labeled dot. The weight vector w which maps each pixel's feature vector to density distribution is learnt from the training images. Counting is then performed by summing the density of all pixels in the test images.

Experiment Results. We do three experiments to compare the performance of different solutions based on the above settings: 1) Compare the BCFW algorithm and the cutting plane method (CP) with active set algorithm as the inner algorithm. 2) Compare the BCFW and the CP with interior point algorithm as the inner algorithm. 3) Compare the BCFW with weighted average $\bar{w}$ and the BCFW without weighted average $\bar{w}$.

Fig. 1 (a) shows the objective value of BCFW and CP with inner active set algorithm over time. From the figure we can see that BCFW converges much faster than CP. Fig. 1 (b) shows the error rate of BCFW and CP with inner active set algorithm over time. A pretty low training and testing error rate can be achieved, which means the model here is decent. Also, BCFW and CP can finally get similar results, but BCFW is much faster in terms of wall clock time. Fig. 1 (c) shows the objective value of BCFW and CP with inner interior point algorithm over time. Similarly, we can observe that, in terms of wall clock time, the objective value of BCFW has lower objective value in the first 250 seconds, and then cutting plane achieves slightly lower objective value. Eventually, BCFW and CP reach very close objective values, but BCFW converges faster, especially in the first 100 seconds. Generally speaking, according to our experiments, BCFW performs better than CP.

The experiments above all use BCFW with weighted average. We also do experiemnts to compare the weighted-average version and non-weighted-average version, shown in Fig. 1 (d). Obviously, The former one is much more stable, so the weighted average is important in BCFW.

5 Conclusion

In this paper, we formulate the object counting problem as a convex optimization problem. We explored BCFW algorithm in solving this optimization problem with large number of constraints. Experiments show that BCFW performs better than Cutting Plane method in general. Several important properties of BCFW are also revealed: (1) BCFW converges faster than CP; (2) weighted average is a useful strategy to keep the BCFW algorithm stable.

Acknowledgement

We thank Prof. Ryan Tibshirani, Wei Yu, and Yuxiang Wang for helpful discussions and comments.

References

[1] Francis Bach, Simon Lacoste-Julien, and Guillaume Obozinski. On the equivalence between herding and conditional gradient algorithms. *arXiv preprint arXiv:1203.4523*, 2012.

[2] Olga Barinova, Victor Lempitsky, and Pushmeet Kholi. On detection of multiple object instances using hough transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1773–1784, 2012.

[3] Chen Change Loy, Shaogang Gong, and Tao Xiang. From semi-supervised to transfer counting of crowds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2256–2263, 2013.

[4] Ke Chen, Shaogang Gong, Tao Xiang, and Chen Change Loy. Cumulative attribute space for age and crowd density estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2467–2474, 2013.

[5] Chaitanya Desai, Deva Ramanan, and Charless C Fowlkes. Discriminative models for multi-class object layout. *International journal of computer vision*, 95(1):1–12, 2011.

[6] Wesley Nunes Gonçalves, Bruno Brandoli Machado, and Odemir Martinez Bruno. Spatiotemporal gabor filters: a new method for dynamic texture recognition. *arXiv preprint arXiv:1201.3612*, 2012.

[7] Tsuyoshi Ide, Takayuki Katsuki, and Tetsuro Morimura. Monitoring entire-city traffic using low-resolution web cameras. In *Proceedings of the 20th ITS World Congress*, 2013.

[8] Haroon Idrees, Imran Saleemi, Cody Seibert, and Mubarak Shah. Multi-source multi-scale counting in extremely dense crowd images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2547–2554, 2013.

[9] Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML (1)*, pages 427–435, 2013.

[10] Martin Jaggi, Simon Lacoste-Julien, Mark Schmidt, and Patrick Pletscher. Block-coordinate frank–wolfe for structural svms. In *NIPS Workshop on Optimization for Machine Learning, Lake Tahoe, NV, USA*. Citeseer, 2012.

[11] Thorsten Joachims, Thomas Finley, and Chun-Nam John Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, 2009.

[12] Simon Lacoste Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-coordinate frank-wolfe optimization for structural svms. In *arXiv preprint*, 2012.

[13] A. Kembhavi, D. Harwood, and L.S. Davis. Vehicle detection using partial least squares. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2011.

[14] Victor S. Lempitsky and Andrew Zisserman. Learning to count objects in images. In *NIPS*, 2010.

[15] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[16] Hua Ouyang and Alexander G Gray. Fast stochastic frank-wolfe algorithms for nonlinear svms. In *SDM*, pages 245–256. SIAM, 2010.

[17] Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. (online) subgradient methods for structured prediction. 2007.

[18] J. Wu S. Hu and L. Xu. Real-time traffic congestion detection based on video analysis. In *Journal of Information and Computational Science*, 2012.

[19] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

Appendix

Due to space limitation, we explain our algorithm detailedly in the appendix.

The Max Oracle

The Max Oracle is defined as follows:

$$\hat{H}_{iB} = \max_{B \in \mathcal{B}_i} \underbrace{\sum_{p \in B} (w^\top x_p^i - F_i^0(p))}_{= H_{iB}} \tag{5}$$

With this definition, the constraints in (2) can thus be written as $\xi_i \geq -\hat{H}_{iB}$ and $\xi_i \geq \hat{H}_{iB}$. Here in our problem, the Max Oracle problem is actually a 2D maximum subarray problem in an image and can be solved in polynomial time. Note that solving the Max Oracle problem is actually to find out the most violated constraints in (2).

Derivation of the Dual

Now let's derive the dual of our optimization problem and show it can be solved by BCFW with some modifications. Recall the primal problem in (2). The Lagrangian is:

$$\begin{aligned} L(w, \xi, \alpha, \beta) &= \frac{1}{2} w^\top w + C \sum_{i=1}^{N} \xi_i + \sum_{i=1}^{N} \sum_{B \in \mathcal{B}_i} \Big(\alpha_{iB} \big(\sum_{p \in B} (F_i^0(p) - w^\top x_p^i) - \xi_i \big) \\ &\quad + \beta_{iB} \big(\sum_{p \in B} (w^\top x_p^i - F_i^0(p)) - \xi_i \big) \Big) \\ &= \frac{1}{2} w^\top w + C \sum_{i=1}^{N} \xi_i \\ &\quad + \sum_{i=1}^{N} \sum_{B \in \mathcal{B}_i} \Big((\alpha_{iB} - \beta_{iB}) \sum_{p \in B} (F_i^0(p) - w^\top x_p^i) - (\alpha_{iB} + \beta_{iB}) \xi_i \Big) \end{aligned} \tag{6}$$

where α and β are lagrange multipliers. To calculate $\min_{w,\xi} L(w, \xi, \alpha, \beta)$, we take derivatives with respect to w and ξ_i and set them to 0:

$$\begin{aligned} \frac{\partial L}{\partial w} &= w + \sum_{i=1}^{N} \sum_{B \in \mathcal{B}_i} \Big((\beta_{iB} - \alpha_{iB}) \sum_{p \in B} x_p^i \Big) = 0 \\ \frac{\partial L}{\partial \xi_i} &= C + \sum_{B \in \mathcal{B}_i} (-\alpha_{iB} - \beta_{iB}) = 0 \end{aligned}$$

We get

$$\begin{aligned} w &= \sum_{i=1}^{N} \sum_{B \in \mathcal{B}_i} \Big((\alpha_{iB} - \beta_{iB}) \sum_{p \in B} x_p^i \Big) \\ C &= \sum_{B \in \mathcal{B}_i} \alpha_{iB} + \beta_{iB} \end{aligned} \tag{7}$$

Plugging (7) to (6) and do some scaling on α, β, the dual problem can be written as:

$$\begin{aligned} \min_{\alpha,\beta} \quad & \frac{1}{2} \| C \sum_{i=1}^{N} \sum_{B \in \mathcal{B}_i} \Big((\alpha_{iB} - \beta_{iB}) \sum_{p \in B} x_p^i \Big) \|_2^2 - C \sum_{i=1}^{N} \sum_{B \in \mathcal{B}_i} \Big((\alpha_{iB} - \beta_{iB}) \sum_{p \in B} F_i^0(p) \Big) \\ \text{s.t.} \quad & \forall i : \sum_{B \in \mathcal{B}_i} \alpha_{iB} + \beta_{iB} = 1 \qquad \alpha_{iB}, \beta_{iB} \geq 0 \end{aligned} \tag{8}$$

Simplify it into matrix form:

$$\begin{aligned} \min_{\alpha,\beta} \quad & \frac{1}{2}\|A(\alpha-\beta)\|_2^2 - b^\top(\alpha-\beta) \\ \text{s.t.} \quad & \forall i: \sum_{B\in\mathcal{B}_i} \alpha_{iB} + \beta_{iB} = 1 \qquad \alpha_{iB}, \beta_{iB} \geq 0 \end{aligned}$$

where $A \in \mathbb{R}^{d\times m}$ and $A = \{C\sum_{p\in B} x_p^i | i \in [n], B \in \mathcal{B}_i\}$, $b = (C\sum_{p\in B} x_p^i)_{i\in[n],B\in\mathcal{B}_i}$, $\alpha = (\alpha_{iB})_{i\in[n],B\in\mathcal{B}_i}$, $\beta = (\beta_{iB})_{i\in[n],B\in\mathcal{B}_i}$ and d is the feature dimension, $m = \sum_i^N |\mathcal{B}_i|$. To meet our target form (3), we further let $X = (A, -A), c = (b^\top, -b^\top)^\top, \gamma = (\alpha^\top, \beta^\top)^\top$, the optimization problem can be transformed into:

$$\begin{aligned} \min_{\gamma} \quad & \frac{1}{2}\|X\gamma\|_2^2 - c^\top\gamma \\ \text{s.t.} \quad & \forall i: \sum_{g=1}^{2|\mathcal{B}_i|} \gamma_{ig} = 1 \qquad \gamma_i \geq 0 \end{aligned}$$

where $\gamma_i := (\alpha_i^\top, \beta_i^\top)^\top$, note that γ_i takes nonconsecutive blocks from γ. This has the same form as (3). Next we derive the specific BCFW algorithm for our problem.

Derivation of the BCFW

To derive BCFW algorithm for our problem, we need to first review the Frank-Wolfe algorithm, shown in Alg. 2.

Algorithm 2 Frank-Wolfe Algorithm

1: Let $\gamma^0 \in \mathcal{M}$
2: **for** $k \leftarrow 0$ to K **do**
3: Compute $s := \text{argmin}_{s'\in\mathcal{M}} \langle s', \nabla f(\gamma)\rangle$
4: Let $t := \frac{2}{2+k}$, or optimize t by line-search
5: Update $\gamma^{k+1} = (1-t)\gamma^k + t\cdot s$

Line 3 in Alg. 2 is the core part of Frank-Wolfe algorithm. Our main observation is Lemma 1. The proof of Lemma 1 is as follows.

Proof.

$$\nabla f(\gamma) = X^\top X\gamma - c = X^\top w - c = \begin{pmatrix} A^\top w - b \\ -(A^\top w - b) \end{pmatrix}$$

Since $\gamma \in \mathcal{M}^{(1)} \times ... \times \mathcal{M}^{(n)}$ and $\gamma_i := (\alpha_i^\top, \beta_i^\top)^\top$, correspondingly, we need to define $\nabla_i f(\gamma)$ as

$$\nabla_i f(\gamma) = \begin{pmatrix} -C\cdot H_i \\ C\cdot H_i \end{pmatrix}$$

where $H_i = (H_{iB})_{B\in\mathcal{B}_i}$ is a vector with dimension $|\mathcal{B}_i|$. Then, the minimization $\min_{s'\in\mathcal{M}}\langle s', \nabla f(\gamma)\rangle$ decomposes as $\sum_i \min_{s_i\in\mathcal{M}^{(i)}}\langle s_i, \nabla_i f(\gamma)\rangle$. Note that $\mathcal{M}^{(i)}$ is a probability simplex, and the minimization of a linear function over a simplex is to search over its corners (only one component is 1, others are 0). Thus, for each $\min_{s_i\in\mathcal{M}^{(i)}}\langle s_i, \nabla_i f(\gamma)\rangle$, it is equivalent to finding the minimal component in $\nabla_i f(\gamma)$, i.e. $\min_{B\in\mathcal{B}_i}\{-C\cdot H_{iB}, C\cdot H_{iB}\}$, which is the same as $\max_{B\in\mathcal{B}_i}\{H_{iB}, -H_{iB}\}$. □

With Lemma 1 in mind, we can get the Primal-Dual BCFW algorithm by only maintaining the corresponding primal iterates. Please refer to Alg. 1 for details.