Block-Coordinate Frank-Wolfe Optimization for Structural SVMs

Simon Lacoste-Julien[*] INRIA - SIERRA project-team, École Normale Supérieure, Paris, France
Martin Jaggi[*] CMAP, École Polytechnique, Palaiseau, France
Mark Schmidt INRIA - SIERRA project-team, École Normale Supérieure, Paris, France
Patrick Pletscher Machine Learning Laboratory, ETH Zurich, Switzerland
[*] Both authors contributed equally.

Abstract

We propose a randomized block-coordinate variant of the classic Frank-Wolfe algorithm for convex optimization with block-separable constraints. Despite its lower iteration cost, we show that it achieves a similar convergence rate in duality gap as the full Frank-Wolfe algorithm. We also show that, when applied to the dual structural support vector machine (SVM) objective, this yields an online algorithm that has the same low iteration complexity as primal stochastic subgradient methods. However, unlike stochastic subgradient methods, the block-coordinate Frank-Wolfe algorithm allows us to compute the *optimal* step-size and yields a computable duality gap guarantee. Our experiments indicate that this simple algorithm outperforms competing structural SVM solvers.

1. Introduction

Binary SVMs are amongst the most popular classification methods, and this has motivated substantial interest in optimization solvers that are tailored to their specific problem structure. However, despite their wider applicability, there has been much less work on solving the optimization problem associated with *structural* SVMs, which are the generalization of SVMs to structured outputs like graphs and other combinatorial objects (Taskar et al., 2003; Tsochantaridis et al., 2005). This seems to be due to the difficulty of dealing with the exponential number of constraints in the primal problem, or the exponential number of variables in the dual problem. Indeed, because they achieve an $\tilde{O}(1/\varepsilon)$ convergence rate while only requiring a single call to the so-called *maximization oracle* on each iteration, basic stochastic subgradient methods are still widely used for training structural SVMs (Ratliff et al., 2007; Shalev-Shwartz et al., 2010a). However, these methods are often frustrating to use for practitioners, because their performance is very sensitive to the sequence of step sizes, and because it is difficult to decide when to terminate the iterations.

To solve the dual structural SVM problem, in this paper we consider the Frank-Wolfe (1956) algorithm, which has seen a recent surge of interest in machine learning and signal processing (Mangasarian, 1995; Clarkson, 2010; Jaggi, 2011; 2013; Bach et al., 2012), including in the context of binary SVMs (Gärtner & Jaggi, 2009; Ouyang & Gray, 2010). A key advantage of this algorithm is that the iterates are *sparse*, and we show that this allows us to efficiently apply it to the dual structural SVM objective even though there are an exponential number of variables. A second key advantage of this algorithm is that the iterations only require optimizing linear functions over the constrained domain, and we show that *this is equivalent to the maximization oracle* used by subgradient and cutting-plane methods (Joachims et al., 2009; Teo et al., 2010). Thus, the Frank-Wolfe algorithm has the same wide applicability as subgradient methods, and can be applied to problems such as low-treewidth graphical models (Taskar et al., 2003), graph matchings (Caetano et al., 2009), and associative Markov networks (Taskar, 2004). In contrast, other approaches must use more expensive (and potentially intractable) oracles such as computing marginals over labels (Collins et al., 2008; Zhang et al., 2011) or doing a Bregman projection onto the space of structures (Taskar et al., 2006). Interestingly, for structural SVMs we also show that existing batch subgradient and cutting-plane methods are *special cases* of Frank-Wolfe algorithms, and this leads to stronger and simpler $O(1/\varepsilon)$ convergence rate guarantees for these existing algorithms.

Proceedings of the 30^{th} *International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

As in other batch structural SVM solvers like cutting-plane methods (Joachims et al., 2009; Teo et al., 2010) and the excessive gap technique (Zhang et al., 2011) (see Table 1 at the end for an overview), each Frank-Wolfe iteration unfortunately requires calling the appropriate oracle once for *all* training examples, unlike the single oracle call needed by stochastic subgradient methods. This can be prohibitive for data sets with a large number of training examples. To reduce this cost, we propose a novel randomized block-coordinate version of the Frank-Wolfe algorithm for problems with block-separable constraints. We show that this algorithm still achieves the $O(1/\varepsilon)$ convergence rate of the full Frank-Wolfe algorithm, and in the context of structural SVMs, it only requires a single call to the maximization oracle. Although the stochastic subgradient and the novel block-coordinate Frank-Wolfe algorithms have a similar iteration cost and theoretical convergence rate for solving the structural SVM problem, the new algorithm has several important advantages for practitioners:

- The *optimal* step-size can be efficiently computed in closed-form, hence no step-size needs to be selected.
- The algorithm yields a *duality gap* guarantee, and (at the cost of computing the primal objective) we can compute the duality gap as a proper stopping criterion.
- The convergence rate holds even when using *approximate* maximization oracles.

Further, our experimental results show that the optimal step-size leads to a significant advantage during the first few passes through the data, and a systematic (but smaller) advantage in later passes.

2. Structural Support Vector Machines

We first briefly review the standard convex optimization setup for structural SVMs (Taskar et al., 2003; Tsochantaridis et al., 2005). In structured prediction, the goal is to predict a structured object $\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})$ (such as a sequence of tags) for a given input $\boldsymbol{x} \in \mathcal{X}$. In the standard approach, a structured feature map $\boldsymbol{\phi} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ encodes the relevant information for input/output pairs, and a linear classifier with parameter $\boldsymbol{w}$ is defined by $h_{\boldsymbol{w}}(\boldsymbol{x}) = \operatorname{argmax}_{\boldsymbol{y}\in\mathcal{Y}(\boldsymbol{x})} \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y})\rangle$. Given a labeled training set $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$, $\boldsymbol{w}$ is estimated by solving

$$\begin{aligned} \min_{\boldsymbol{w},\,\boldsymbol{\xi}} \quad & \frac{\lambda}{2}\left\|\boldsymbol{w}\right\|^2 + \frac{1}{n}\sum_{i=1}^{n} \xi_i && (1)\\ \text{s.t.} \quad & \langle \boldsymbol{w}, \boldsymbol{\psi}_i(\boldsymbol{y})\rangle \geq L(\boldsymbol{y}_i, \boldsymbol{y}) - \xi_i \quad \forall i,\ \forall \boldsymbol{y} \in \overbrace{\mathcal{Y}(\boldsymbol{x}_i)}^{=:\mathcal{Y}_i}, \end{aligned}$$

where $\boldsymbol{\psi}_i(\boldsymbol{y}) := \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y})$, and $L_i(\boldsymbol{y}) := L(\boldsymbol{y}_i, \boldsymbol{y})$ denotes the task-dependent structured error of predicting output $\boldsymbol{y}$ instead of the observed output $\boldsymbol{y}_i$ (typically a Hamming distance between the two labels). The slack variable ξ_i measures the surrogate loss for the i-th datapoint and λ is the regularization parameter. The convex problem (1) is what Joachims et al. (2009, Optimization Problem 2) call the n-slack structural SVM with margin-rescaling. A variant with *slack-rescaling* was proposed by Tsochantaridis et al. (2005), which is equivalent to our setting if we replace all vectors $\boldsymbol{\psi}_i(\boldsymbol{y})$ by $L_i(\boldsymbol{y})\boldsymbol{\psi}_i(\boldsymbol{y})$.

Loss-Augmented Decoding. Unfortunately, the above problem can have an exponential number of constraints due to the combinatorial nature of $\mathcal{Y}$. We can replace the $\sum_i |\mathcal{Y}_i|$ linear constraints with n *piecewise-linear* ones by defining the structured hinge-loss:

$$\text{`max oracle'} \qquad \tilde{H}_i(\boldsymbol{w}) := \max_{\boldsymbol{y}\in\mathcal{Y}_i} \underbrace{L_i(\boldsymbol{y}) - \langle \boldsymbol{w}, \boldsymbol{\psi}_i(\boldsymbol{y})\rangle}_{=:\, H_i(\boldsymbol{y};\boldsymbol{w})}. \qquad (2)$$

The constraints in (1) can thus be replaced with the non-linear ones $\xi_i \geq \tilde{H}_i(\boldsymbol{w})$. The computation of the structured hinge-loss for each i amounts to finding the most 'violating' output $\boldsymbol{y}$ for a given input $\boldsymbol{x}_i$, a task which can be carried out efficiently in many structured prediction settings (see the introduction). This problem is called the *loss-augmented decoding* subproblem. In this paper, we only assume access to an efficient solver for this subproblem, and we call such a solver a *maximization oracle*. The equivalent non-smooth unconstrained formulation of (1) is:

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\left\|\boldsymbol{w}\right\|^2 + \frac{1}{n}\sum_{i=1}^{n} \tilde{H}_i(\boldsymbol{w}). \qquad (3)$$

Having a maximization oracle allows us to apply subgradient methods to this problem (Ratliff et al., 2007), as a subgradient of $\tilde{H}_i(\boldsymbol{w})$ with respect to $\boldsymbol{w}$ is $-\boldsymbol{\psi}_i(\boldsymbol{y}_i^*)$, where $\boldsymbol{y}_i^*$ is any maximizer of the loss-augmented decoding subproblem (2).

The Dual. The Lagrange dual of the above n-slack-formulation (1) has $m := \sum_i |\mathcal{Y}_i|$ variables or potential 'support vectors'. Writing $\alpha_i(\boldsymbol{y})$ for the dual variable associated with the training example i and potential output $\boldsymbol{y} \in \mathcal{Y}_i$, the dual problem is given by

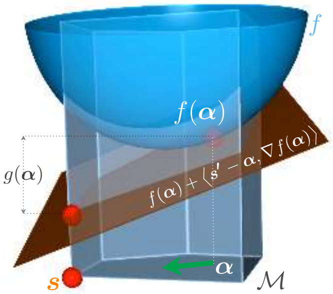
$$\begin{aligned} \min_{\substack{\boldsymbol{\alpha}\in\mathbb{R}^m \\ \boldsymbol{\alpha}\geq 0}} \quad & f(\boldsymbol{\alpha}) \;:=\; \frac{\lambda}{2}\left\|A\boldsymbol{\alpha}\right\|^2 - \boldsymbol{b}^T\boldsymbol{\alpha} && (4)\\ & \text{s.t.} \quad \textstyle\sum_{\boldsymbol{y}\in\mathcal{Y}_i} \alpha_i(\boldsymbol{y}) = 1 \quad \forall i \in [n]\ , \end{aligned}$$

where the matrix $A \in \mathbb{R}^{d\times m}$ consists of the m columns $A := \left\{ \frac{1}{\lambda n}\boldsymbol{\psi}_i(\boldsymbol{y}) \in \mathbb{R}^d \;\middle|\; i \in [n], \boldsymbol{y} \in \mathcal{Y}_i \right\}$, and the vector $\boldsymbol{b} \in \mathbb{R}^m$ is given by $\boldsymbol{b} := \left(\frac{1}{n}L_i(\boldsymbol{y})\right)_{i\in[n],\boldsymbol{y}\in\mathcal{Y}_i}$. Given a dual variable vector $\boldsymbol{\alpha}$, we can use the Karush-Kuhn-Tucker optimality conditions to obtain the corresponding primal variables $\boldsymbol{w} = A\boldsymbol{\alpha} = \sum_{i,\,\boldsymbol{y}\in\mathcal{Y}_i} \alpha_i(\boldsymbol{y}) \frac{\boldsymbol{\psi}_i(\boldsymbol{y})}{\lambda n}$, see Appendix E. The gradient of f then takes the simple form $\nabla f(\boldsymbol{\alpha}) = \lambda A^T A\boldsymbol{\alpha} - \boldsymbol{b} = \lambda A^T \boldsymbol{w} - \boldsymbol{b}$; its $(i,\boldsymbol{y})$-th component is $-\frac{1}{n} H_i(\boldsymbol{y};\boldsymbol{w})$, cf. (2). Finally, note that the domain $\mathcal{M} \subset \mathbb{R}^m$ of (4) is the product of n probability simplices, $\mathcal{M} := \Delta_{|\mathcal{Y}_1|} \times \ldots \times \Delta_{|\mathcal{Y}_n|}$.

3. The Frank-Wolfe Algorithm

We consider the convex optimization problem $\min_{\boldsymbol{\alpha}\in\mathcal{M}} f(\boldsymbol{\alpha})$, where the convex feasible set $\mathcal{M}$ is *compact* and the convex objective f is *continuously differentiable*. The Frank-Wolfe algorithm (1956) (shown in Algorithm 1) is an iterative optimization algorithm for such problems that only requires optimizing *linear* functions over $\mathcal{M}$, and thus has wider applicability than projected gradient algorithms, which require optimizing a quadratic function over $\mathcal{M}$. At every iteration, a feasible search corner $\boldsymbol{s}$ is first found by minimizing over $\mathcal{M}$ the *linearization* of f at the current iterate $\boldsymbol{\alpha}$ (see picture in inset). The next iterate is then obtained as a convex combination of $\boldsymbol{s}$ and the previous iterate, with step-size γ. These simple updates yield two interesting properties. First, every iterate $\boldsymbol{\alpha}^{(k)}$ can be written as a convex combination of the starting point $\boldsymbol{\alpha}^{(0)}$ and the search corners $\boldsymbol{s}$ found previously. The parameter $\boldsymbol{\alpha}^{(k)}$ thus has a sparse representation, which makes the algorithm suitable even for cases where the dimensionality of $\boldsymbol{\alpha}$ is exponential. Second, since f is convex, the minimum of the linearization of f over $\mathcal{M}$ immediately gives a lower bound on the value of the yet unknown optimal solution $f(\boldsymbol{\alpha}^*)$. Every step of the algorithm thus computes for free the following 'linearization duality gap' defined for any feasible point $\boldsymbol{\alpha} \in \mathcal{M}$ (which is in fact a special case of the Fenchel duality gap as explained in Appendix D):

$$g(\boldsymbol{\alpha}) := \max_{\boldsymbol{s}'\in\mathcal{M}} \langle \boldsymbol{\alpha} - \boldsymbol{s}', \nabla f(\boldsymbol{\alpha})\rangle = \langle \boldsymbol{\alpha} - \boldsymbol{s}, \nabla f(\boldsymbol{\alpha})\rangle. \quad (5)$$



Algorithm 1 Frank-Wolfe on a Compact Domain

```
Let $\boldsymbol{\alpha}^{(0)} \in \mathcal{M}$
for $k = 0 \ldots K$ do
    Compute $\boldsymbol{s} := \operatorname{argmin}_{\boldsymbol{s}'\in\mathcal{M}} \left\langle \boldsymbol{s}', \nabla f(\boldsymbol{\alpha}^{(k)}) \right\rangle$
    Let $\gamma := \frac{2}{k+2}$, or optimize $\gamma$ by line-search
    Update $\boldsymbol{\alpha}^{(k+1)} := (1-\gamma)\boldsymbol{\alpha}^{(k)} + \gamma \boldsymbol{s}$
```

As $g(\boldsymbol{\alpha}) \geq f(\boldsymbol{\alpha}) - f(\boldsymbol{\alpha}^*)$ by the above argument, $\boldsymbol{s}$ thus readily gives at each iteration the current duality gap as a *certificate* for the current approximation quality (Jaggi, 2011; 2013), allowing us to monitor the convergence, and more importantly to choose the theoretically sound stopping criterion $g(\boldsymbol{\alpha}^{(k)}) \leq \varepsilon$.

In terms of convergence, it is known that after $O(1/\varepsilon)$ iterations, Algorithm 1 obtains an ε-approximate solution (Frank & Wolfe, 1956; Dunn & Harshbarger, 1978) as well as a guaranteed ε-small duality gap (Clarkson, 2010; Jaggi, 2013), along with a certificate to (5). For the convergence results to hold, the internal linear subproblem does not need to be solved exactly, but only to some error. We review and generalize the convergence proof in Appendix C. The constant hidden in the $O(1/\varepsilon)$ notation is the *curvature constant* C_f, an affine invariant quantity measuring the maximum deviation of f from its linear approximation over $\mathcal{M}$ (it yields a weaker form of Lipschitz assumption on the gradient, see e.g. Appendix A for a formal definition).

4. Frank-Wolfe for Structural SVMs

Note that classical algorithms like the projected gradient method cannot be tractably applied to the dual of the structural SVM problem (4), due to the large number of dual variables. In this section, we explain how the Frank-Wolfe method (Algorithm 1) can be efficiently applied to this dual problem, and discuss its relationship to other algorithms. The main insight here is to notice that the linear subproblem employed by Frank-Wolfe is actually directly equivalent to the loss-augmented decoding subproblem (2) for each datapoint, which can be solved efficiently (see Appendix B.1 for details). Recall that the optimization domain for the dual variables $\boldsymbol{\alpha}$ is the product

Algorithm 2 Batch Primal-Dual Frank-Wolfe Algorithm for the Structural SVM

```
Let $\boldsymbol{w}^{(0)} := \mathbf{0}$,  $\ell^{(0)} := 0$
for $k = 0 \ldots K$ do
    for $i = 1 \ldots n$ do
        Solve $\boldsymbol{y}_i^* := \operatorname{argmax}_{\boldsymbol{y}\in\mathcal{Y}_i} H_i(\boldsymbol{y}; \boldsymbol{w}^{(k)})$ cf. (2)
    Let $\boldsymbol{w_s} := \sum_{i=1}^{n} \frac{1}{\lambda n} \boldsymbol{\psi}_i(\boldsymbol{y}_i^*)$ and $\ell_{\boldsymbol{s}} := \frac{1}{n} \sum_{i=1}^{n} L_i(\boldsymbol{y}_i^*)$
    Let $\gamma := \frac{\lambda(\boldsymbol{w}^{(k)} - \boldsymbol{w_s})^T \boldsymbol{w}^{(k)} - \ell^{(k)} + \ell_{\boldsymbol{s}}}{\lambda \|\boldsymbol{w}^{(k)} - \boldsymbol{w_s}\|^2}$ and clip to $[0,1]$
    Update $\boldsymbol{w}^{(k+1)} := (1-\gamma)\boldsymbol{w}^{(k)} + \gamma\,\boldsymbol{w_s}$
        and $\ell^{(k+1)} := (1-\gamma)\ell^{(k)} + \gamma\,\ell_{\boldsymbol{s}}$
```

of n simplices, $\mathcal{M} = \Delta_{|\mathcal{Y}_1|} \times \ldots \times \Delta_{|\mathcal{Y}_n|}$. Since each simplex consists of a potentially exponential number $|\mathcal{Y}_i|$ of dual variables, we cannot maintain a dense vector $\boldsymbol{\alpha}$ during the algorithm. However, as mentioned in Section 3, each iterate $\boldsymbol{\alpha}^{(k)}$ of the Frank-Wolfe algorithm is a sparse convex combination of the previously visited corners $\boldsymbol{s}$ and the starting point $\boldsymbol{\alpha}^{(0)}$, and so we only need to maintain the list of previously seen solutions to the loss-augmented decoding subproblems to keep track of the non-zero coordinates of $\boldsymbol{\alpha}$, avoiding the problem of its exponential size. Alternately, if we do not use kernels, we can avoid the quadratic explosion of the number of operations needed in the dual by *not* explicitly maintaining $\boldsymbol{\alpha}^{(k)}$, but instead maintaining the corresponding *primal* variable $\boldsymbol{w}^{(k)}$.

A Primal-Dual Frank-Wolfe Algorithm for the Structural SVM Dual. Applying Algorithm 1 with line search to the dual of the structural SVM (4), but only maintaining the corresponding *primal* primal iterates $\boldsymbol{w}^{(k)} := A\boldsymbol{\alpha}^{(k)}$, we obtain Algorithm 2. Note that the Frank-Wolfe search corner $\boldsymbol{s} = (\mathbf{e}^{\boldsymbol{y}_1^*}, \ldots, \mathbf{e}^{\boldsymbol{y}_n^*})$, which is obtained by solving the loss-augmented subproblems, yields the update $\boldsymbol{w}_{\boldsymbol{s}} = A\boldsymbol{s}$. We use the natural starting point $\boldsymbol{\alpha}^{(0)} := (\mathbf{e}^{\boldsymbol{y}_1}, \ldots, \mathbf{e}^{\boldsymbol{y}_n})$ which yields $\boldsymbol{w}^{(0)} = \mathbf{0}$ as $\boldsymbol{\psi}_i(\boldsymbol{y}_i) = \mathbf{0}\ \forall i$.

The Duality Gap. The duality gap (5) for our structural SVM dual formulation (4) is given by

$$\begin{aligned} g(\boldsymbol{\alpha}) := & \max_{\boldsymbol{s}' \in \mathcal{M}} \langle \boldsymbol{\alpha} - \boldsymbol{s}', \nabla f(\boldsymbol{\alpha}) \rangle \\ = & \ (\boldsymbol{\alpha} - \boldsymbol{s})^T (\lambda A^T A \boldsymbol{\alpha} - \boldsymbol{b}) \\ = & \ \lambda (\boldsymbol{w} - A\boldsymbol{s})^T \boldsymbol{w} - \boldsymbol{b}^T \boldsymbol{\alpha} + \boldsymbol{b}^T \boldsymbol{s} \ , \end{aligned}$$

where $\boldsymbol{s}$ is an *exact* minimizer of the linearized problem given at the point $\boldsymbol{\alpha}$. This (Fenchel) duality gap turns out to be the same as the Lagrangian duality gap here (see Appendix B.2), and gives a direct handle on the suboptimality of $\boldsymbol{w}^{(k)}$ for the primal problem (3). Using $\boldsymbol{w}_{\boldsymbol{s}} := A\boldsymbol{s}$ and $\ell_{\boldsymbol{s}} := \boldsymbol{b}^T\boldsymbol{s}$, we observe that the gap is efficient to compute given the *primal* variables $\boldsymbol{w} := A\boldsymbol{\alpha}$ and $\ell := \boldsymbol{b}^T\boldsymbol{\alpha}$, which are maintained during the run of Algorithm 2. Therefore, we can use the duality gap $g(\boldsymbol{\alpha}^{(k)}) \leq \varepsilon$ as a proper stopping criterion.

Implementing the Line-Search. Because the objective of the structural SVM dual (4) is simply a quadratic function in $\boldsymbol{\alpha}$, the optimal step-size for any given candidate search point $\boldsymbol{s} \in \mathcal{M}$ can be obtained *analytically*. Namely, $\gamma_{LS} := \operatorname{argmin}_{\gamma \in [0,1]} f\left(\boldsymbol{\alpha} + \gamma\left(\boldsymbol{s} - \boldsymbol{\alpha}\right)\right)$ is obtained by setting the derivative of this univariate quadratic function in γ to zero, which here (before restricting to $[0, 1]$) gives $\gamma_{opt} := \frac{\langle \boldsymbol{\alpha} - \boldsymbol{s}, \nabla f(\boldsymbol{\alpha}) \rangle}{\lambda \|A(\boldsymbol{\alpha} - \boldsymbol{s})\|^2} = \frac{g(\boldsymbol{\alpha})}{\lambda \|\boldsymbol{w} - \boldsymbol{w}_{\boldsymbol{s}}\|^2}$ (used in Algorithms 2 and 4).

Convergence Proof and Running Time. In the following, we write R for the maximal length of a difference feature vector, i.e. $R := \max_{i \in [n], \boldsymbol{y} \in \mathcal{Y}_i} \|\boldsymbol{\psi}_i(\boldsymbol{y})\|_2$, and we write the maximum error as $L_{\max} := \max_{i, \boldsymbol{y}} L_i(\boldsymbol{y})$. By bounding the curvature constant C_f for the dual SVM objective (4), we can now directly apply the known convergence results for the standard Frank-Wolfe algorithm to obtain the following *primal-dual* rate (proof in Appendix B.3):

Theorem 1. *Algorithm 2 obtains an ε-approximate solution to the structural SVM dual problem (4) and duality gap $g(\boldsymbol{\alpha}^{(k)}) \leq \varepsilon$ after at most $O\left(\frac{R^2}{\lambda\varepsilon}\right)$ iterations, where each iteration costs n oracle calls.*

Since we have proved that the duality gap is smaller than ε, this implies that the original SVM primal objective (3) is actually solved to accuracy ε as well.

Relationship with the Batch Subgradient Method in the Primal. Surprisingly, the batch Frank-Wolfe method (Algorithm 2) is equivalent to the batch subgradient method in the primal, though Frank-Wolfe allows a more clever choice of step-size, since line-search can be used in the dual. To see the equivalence, notice that a subgradient of (3) is given by $\boldsymbol{d}_{sub} = \lambda \boldsymbol{w} - \frac{1}{n}\sum_i \boldsymbol{\psi}_i(\boldsymbol{y}_i^*) = \lambda(\boldsymbol{w} - \boldsymbol{w}_{\boldsymbol{s}})$, where $\boldsymbol{y}_i^*$ and $\boldsymbol{w}_{\boldsymbol{s}}$ are as defined in Algorithm 2. Hence, for a step-size of β, the subgradient method update becomes $\boldsymbol{w}^{(k+1)} := \boldsymbol{w}^{(k)} - \beta \boldsymbol{d}_{sub} = \boldsymbol{w}^{(k)} - \beta\lambda(\boldsymbol{w}^{(k)} - \boldsymbol{w}_{\boldsymbol{s}}) = (1 - \beta\lambda)\boldsymbol{w}^{(k)} + \beta\lambda\boldsymbol{w}_{\boldsymbol{s}}$. Comparing this with Algorithm 2, we see that each Frank-Wolfe step on the dual problem (4) with step-size γ is equivalent to a batch subgradient step in the primal with a step-size of $\beta = \gamma/\lambda$, and thus our convergence results also apply to it. This seems to generalize the equivalence between Frank-Wolfe and the subgradient method for a quadratic objective with identity Hessian as observed by Bach et al. (2012, Section 4.1).

Relationship with Cutting Plane Algorithms. In each iteration, the cutting plane algorithm of Joachims et al. (2009) and the Frank-Wolfe method (Algorithm 2) solve the loss-augmented decoding problem for each datapoint, selecting the same new 'active' coordinates to add to the dual problem. The only difference is that instead of just moving towards the corner $\boldsymbol{s}$, as in classical Frank-Wolfe, the cutting plane algorithm re-optimizes over all the previously added 'active' dual variables (this task is a quadratic program). This shows that the method is exactly equivalent to the 'fully corrective' variant of Frank-Wolfe, which in each iteration re-optimizes over all previously visited corners (Clarkson, 2010; Shalev-Shwartz et al., 2010b). Note that the convergence results for the 'fully correc-

Algorithm 3 Block-Coordinate Frank-Wolfe Algorithm on Product Domain

```
Let $\boldsymbol{\alpha}^{(0)} \in \mathcal{M} = \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}$
for $k = 0 \ldots K$ do
  Pick $i$ at random in $\{1, \ldots, n\}$
  Find $\boldsymbol{s}_{(i)} := \underset{\boldsymbol{s}'_{(i)} \in \mathcal{M}^{(i)}}{\operatorname{argmin}} \left\langle \boldsymbol{s}'_{(i)}, \nabla_{(i)} f(\boldsymbol{\alpha}^{(k)}) \right\rangle$
  Let $\gamma := \frac{2n}{k+2n}$, or optimize $\gamma$ by line-search
  Update $\boldsymbol{\alpha}_{(i)}^{(k+1)} := \boldsymbol{\alpha}_{(i)}^{(k)} + \gamma\big(\boldsymbol{s}_{(i)} - \boldsymbol{\alpha}_{(i)}^{(k)}\big)$
```

tive' variant directly follow from the ones for Frank-Wolfe (by inclusion), thus our convergence results apply to the cutting plane algorithm of Joachims et al. (2009), significantly simplifying its analysis.

5. Faster Block-Coordinate Frank-Wolfe

A major disadvantage of the standard Frank-Wolfe algorithm when applied to the structural SVM problem is that each iteration requires a full pass through the data, resulting in n calls to the maximization oracle. In this section, we present the main new contribution of the paper: a *block-coordinate* generalization of the Frank-Wolfe algorithm that maintains all appealing properties of Frank-Wolfe, but yields much cheaper iterations, requiring only one call to the maximization oracle in the context of structural SVMs. The new method is given in Algorithm 3, and applies to any constrained convex optimization problem of the form

$$\min_{\boldsymbol{\alpha} \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}} f(\boldsymbol{\alpha}) \;, \tag{6}$$

where the domain has the structure of a Cartesian product $\mathcal{M} = \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)} \subseteq \mathbb{R}^m$ over $n \geq 1$ blocks. The main idea of the method is to perform cheaper update steps that only affect a single variable block $\mathcal{M}^{(i)}$, and not all of them simultaneously. This is motivated by coordinate descent methods, which have a very successful history when applied to large scale optimization. Here we assume that each factor $\mathcal{M}^{(i)} \subseteq \mathbb{R}^{m_i}$ is convex and *compact*, with $m = \sum_{i=1}^n m_i$. We will write $\boldsymbol{\alpha}_{(i)} \in \mathbb{R}^{m_i}$ for the i-th block of coordinates of a vector $\boldsymbol{\alpha} \in \mathbb{R}^m$. In each step, Algorithm 3 picks one of the n blocks uniformly at random, and leaves all other blocks unchanged. If there is only one block ($n = 1$), then Algorithm 3 becomes the standard Frank-Wolfe Algorithm 1. The algorithm can be interpreted as a simplification of Nesterov's 'huge-scale' uniform coordinate descent method (Nesterov, 2012, Section 4). Here, instead of computing a projection operator on a block (which is intractable for structural SVMs), we only need to solve one linear subproblem in each iteration, which for structural SVMs is equivalent to a call to the maximization oracle.

Algorithm 4 Block-Coordinate Primal-Dual Frank-Wolfe Algorithm for the Structural SVM

```
Let $\boldsymbol{w}^{(0)} := \boldsymbol{w}_i{}^{(0)} := \bar{\boldsymbol{w}}^{(0)} := \mathbf{0}$,  $\ell^{(0)} := \ell_i{}^{(0)} := 0$
for $k = 0 \ldots K$ do
  Pick $i$ at random in $\{1, \ldots, n\}$
  Solve $\boldsymbol{y}_i^* := \underset{\boldsymbol{y} \in \mathcal{Y}_i}{\operatorname{argmax}}\; H_i(\boldsymbol{y}; \boldsymbol{w}^{(k)})$ cf. (2)
  Let $\boldsymbol{w}_{\boldsymbol{s}} := \frac{1}{\lambda n} \boldsymbol{\psi}_i(\boldsymbol{y}_i^*)$ and $\ell_{\boldsymbol{s}} := \frac{1}{n} L_i(\boldsymbol{y}_i^*)$
  Let $\gamma := \frac{\lambda(\boldsymbol{w}_i^{(k)} - \boldsymbol{w}_{\boldsymbol{s}})^T \boldsymbol{w}^{(k)} - \ell_i^{(k)} + \ell_{\boldsymbol{s}}}{\lambda \|\boldsymbol{w}_i^{(k)} - \boldsymbol{w}_{\boldsymbol{s}}\|^2}$ and clip to $[0, 1]$
  Update $\boldsymbol{w}_i{}^{(k+1)} := (1-\gamma)\boldsymbol{w}_i{}^{(k)} + \gamma\, \boldsymbol{w}_{\boldsymbol{s}}$
      and $\ell_i{}^{(k+1)} := (1-\gamma)\ell_i{}^{(k)} + \gamma\, \ell_{\boldsymbol{s}}$
  Update $\boldsymbol{w}^{(k+1)} := \boldsymbol{w}^{(k)} + \boldsymbol{w}_i{}^{(k+1)} - \boldsymbol{w}_i{}^{(k)}$
      and $\ell^{(k+1)} := \ell^{(k)} + \ell_i{}^{(k+1)} - \ell_i{}^{(k)}$
  (Optionally: Update $\bar{\boldsymbol{w}}^{(k+1)} := \frac{k}{k+2} \bar{\boldsymbol{w}}^{(k)} + \frac{2}{k+2} \boldsymbol{w}^{(k+1)}$)
```

Convergence Results. The following main theorem shows that after $O(1/\varepsilon)$ many iterations, Algorithm 3 obtains an ε-approximate solution to (6), and guaranteed ε-small duality gap (proof in Appendix C). Here the constant $C_f^{\otimes} := \sum_{i=1}^n C_f^{(i)}$ is the sum of the (partial) curvature constants of f with respect to the individual domain block $\mathcal{M}^{(i)}$. We discuss this Lipschitz assumption on the gradient in more details in Appendix A, where we compute the constant precisely for the structural SVM and obtain $C_f^{\otimes} = C_f/n$, where C_f is the classical Frank-Wolfe curvature.

Theorem 2. *For each $k \geq 0$, the iterate $\boldsymbol{\alpha}^{(k)}$ of Algorithm 3 (either using the predefined step-sizes, or using line-search) satisfies $\mathrm{E}\big[f(\boldsymbol{\alpha}^{(k)})\big] - f(\boldsymbol{\alpha}^*) \leq \frac{2n}{k+2n}\big(C_f^{\otimes} + h_0\big)$, where $\boldsymbol{\alpha}^* \in \mathcal{M}$ is a solution to problem (6), $h_0 := f(\boldsymbol{\alpha}^{(0)}) - f(\boldsymbol{\alpha}^*)$ is the initial error at the starting point of the algorithm, and the expectation is over the random choice of the block i in the steps of the algorithm.*

Furthermore, if Algorithm 3 is run for $K \geq 0$ iterations, then it has an iterate $\boldsymbol{\alpha}^{(\hat{k})}$, $0 \leq \hat{k} \leq K$, with duality gap bounded by $\mathrm{E}\big[g(\boldsymbol{\alpha}^{(\hat{k})})\big] \leq \frac{6n}{K+1}\big(C_f^{\otimes} + h_0\big)$.

Application to the Structural SVM. Algorithm 4 applies the block-coordinate Frank-Wolfe algorithm with line-search to the structural SVM dual problem (4), maintaining only the primal variables $\boldsymbol{w}$. We see that Algorithm 4 is equivalent to Algorithm 3, by observing that the corresponding primal updates become $\boldsymbol{w}_{\boldsymbol{s}} = A\boldsymbol{s}_{[i]}$ and $\ell_{\boldsymbol{s}} = \boldsymbol{b}^T \boldsymbol{s}_{[i]}$. Here $\boldsymbol{s}_{[i]}$ is the zero-padding of $\boldsymbol{s}_{(i)} := \mathbf{e}^{\boldsymbol{y}_i^*} \in \mathcal{M}^{(i)}$ so that $\boldsymbol{s}_{[i]} \in \mathcal{M}$. Note that Algorithm 4 has a primal parameter vector $\boldsymbol{w}_i$ ($= A\boldsymbol{\alpha}_{[i]}$) for each datapoint i, but that this does not significantly increase the storage cost of the algorithm since each $\boldsymbol{w}_i$ has a sparsity pattern that is the union of the corresponding $\boldsymbol{\psi}_i(\boldsymbol{y}_i^*)$ vectors. If the feature vectors are not sparse, it might be more efficient to work directly in the dual instead (see the kernelized version below). The line-search is analogous to the batch Frank-Wolfe case discussed above, and formalized in Appendix B.4.

By applying Theorem 2 to the SVM case where $C_f^{\otimes} = C_f/n = 4R^2/\lambda n$ (in the worst case), we get that the number of iterations needed for our new block-wise Algorithm 4 to obtain a specific accuracy ε is the same as for the batch version in Algorithm 2 (under the assumption that the initial error h_0 is smaller than $4R^2/\lambda n$), even though each iteration takes n times fewer oracle calls. If $h_0 > 4R^2/\lambda n$, we can use the fact that Algorithm 4 is using line-search to get a weaker dependence on h_0 in the rate (Theorem C.4). We summarize the overall rate as follows (proof in Appendix B.3):

Theorem 3. *If $L_{max} \leq \frac{4R^2}{\lambda n}$ (so $h_0 \leq \frac{4R^2}{\lambda n}$), then Algorithm 4 obtains an ε-approximate solution to the structural SVM dual problem (4) and expected duality gap $\mathrm{E}[g(\boldsymbol{\alpha}^{(k)})] \leq \varepsilon$ after at most $O\left(\frac{R^2}{\lambda\varepsilon}\right)$ iterations, where each iteration costs a single oracle call.*

If $L_{max} > \frac{4R^2}{\lambda n}$, then it requires at most an additional (constant in ε) number of $O\left(n \log\left(\frac{\lambda n L_{max}}{R^2}\right)\right)$ steps to get the same error and duality gap guarantees.

In terms of ε, the $O(1/\varepsilon)$ convergence rate above is similar to existing stochastic subgradient and cutting-plane methods. However, unlike stochastic subgradient methods, the block-coordinate Frank-Wolfe method allows us to compute the optimal step-size at each iteration (while for an additional pass through the data we can evaluate the duality gap (5) to allow us to decide when to terminate the algorithm in practice). Further, unlike cutting-plane methods which require n oracle calls per iteration, this rate is achieved 'online', using only a single oracle call per iteration.

Approximate Subproblems and Decoding. Interestingly, we can show that all the convergence results presented in this paper also hold if only approximate minimizers of the linear subproblems are used instead of exact minimizers. If we are using an approximate oracle giving candidate directions $\boldsymbol{s}_{(i)}$ in Algorithm 3 (or $\boldsymbol{s}$ in Algorithm 1) with a *multiplicative* accuracy $\nu \in (0, 1]$ (with respect to the the duality gap (5) on the current block), then the above convergence bounds from Theorem 2 still apply. The only change is that the convergence is slowed by a factor of $1/\nu^2$. We prove this generalization in the Theorems of Appendix C. For structural SVMs, this significantly improves the applicability to large-scale problems, where *exact* decoding is often too costly but *approximate* loss-augmented decoding may be possible.

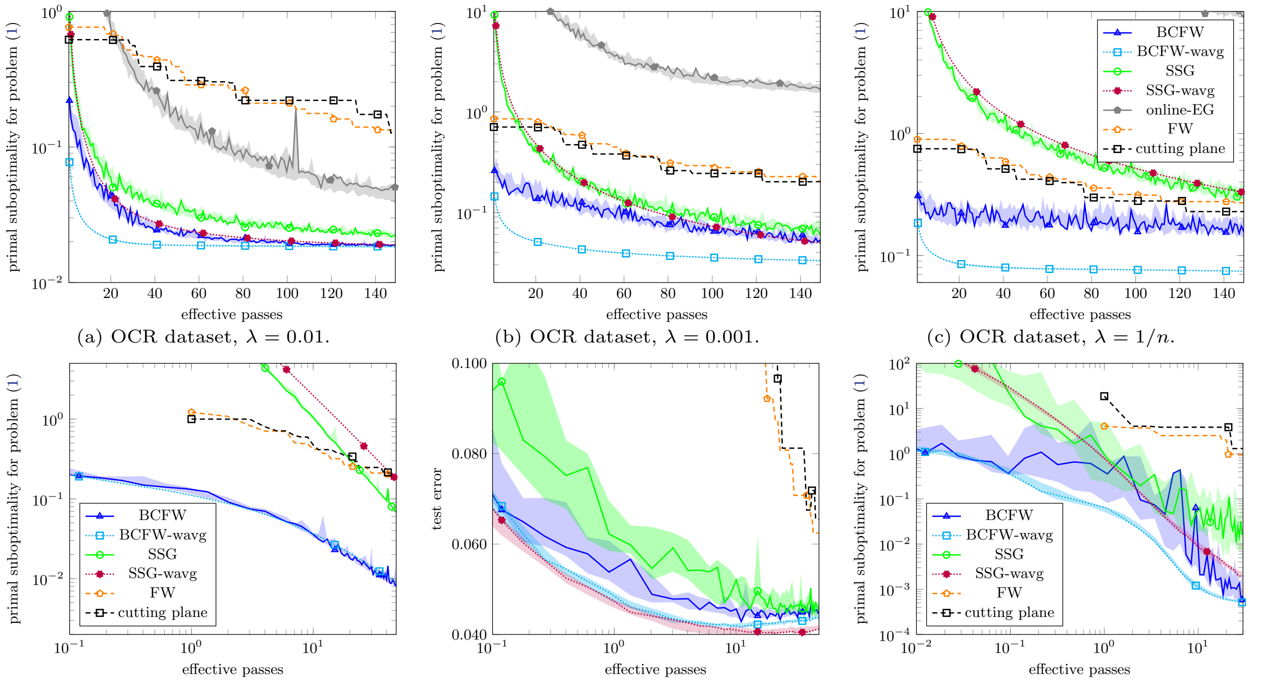
Kernelized Algorithms. Both Algorithms 2 and 4 can directly be used with kernels by maintaining the sparse dual variables $\boldsymbol{\alpha}^{(k)}$ instead of the primal variables $\boldsymbol{w}^{(k)}$. In this case, the classifier is only given implicitly as a sparse combination of the corresponding kernel functions, i.e. $\boldsymbol{w} = A\boldsymbol{\alpha}$. Using our Algorithm 4, we obtain the currently best known bound on the *number of support vectors*, i.e. a guaranteed ε-approximation with only $O(\frac{R^2}{\lambda\varepsilon})$ support vectors. In comparison, the standard cutting plane method (Joachims et al., 2009) adds n support vectors $\boldsymbol{\psi}_i(\boldsymbol{y}_i^*)$ at each iteration. More details on the kernelized variant of Algorithm 4 are discussed in Appendix B.5.

6. Experiments

We compare our novel Frank-Wolfe approach to existing algorithms for training structural SVMs on the OCR dataset ($n = 6251, d = 4028$) from Taskar et al. (2003) and the CoNLL dataset ($n = 8936, d = 1643026$) from Sang & Buchholz (2000). Both datasets are sequence labeling tasks, where the loss-augmented decoding problem can be solved exactly by the Viterbi algorithm. Our third application is a word alignment problem between sentences in different languages in the setting of Taskar et al. (2006) ($n = 5000, d = 82$). Here, the structured labels are bipartite matchings, for which computing marginals over labels as required by the methods of Collins et al. (2008); Zhang et al. (2011) is intractable, but loss-augmented decoding can be done efficiently by solving a min-cost flow problem.

We compare Algorithms 2 and 4, the batch Frank-Wolfe method (*FW*)[1] and our novel block-coordinate Frank-Wolfe method (*BCFW*), to the *cutting plane* algorithm implemented in SVMstruct (Joachims et al., 2009) with its default options, the online exponentiated gradient (*online-EG*) method of Collins et al. (2008), and the stochastic subgradient method (*SSG*) with step-size chosen as in the 'Pegasos' version of Shalev-Shwartz et al. (2010a). We also include the weighted average $\bar{\boldsymbol{w}}^{(k)} := \frac{2}{k(k+1)} \sum_{t=1}^{k} t\boldsymbol{w}^{(t)}$ of the iterates from *SSG* (called *SSG-wavg*) which was recently shown to converge at the faster rate of $O(1/k)$ instead of $O((\log k)/k)$ (Lacoste-Julien et al., 2012; Shamir & Zhang, 2013). Analogously, we average the iterates from *BCFW* the same way to obtain the *BCFW-wavg* method (implemented efficiently with the optional line in Algorithm 4), which also has a provable $O(1/k)$ convergence rate (Theorem C.3). The performance of the different algorithms according to several criteria is visualized in Figure 1. The results are discussed

[1] This is equivalent to the batch subgradient method with an adaptive step-size, as mentioned in Section 4.



(a) OCR dataset, $\lambda = 0.01$.

(b) OCR dataset, $\lambda = 0.001$.

(c) OCR dataset, $\lambda = 1/n$.

(d) CoNLL dataset, $\lambda = 1/n$.

(e) Test error for $\lambda = 1/n$ on CoNLL.

(f) Matching dataset, $\lambda = 0.001$.

Figure 1. The shaded areas for the stochastic methods (*BCFW*, *SSG* and *online-EG*) indicate the worst and best objective achieved in 10 randomized runs. The top row compares the suboptimality achieved by different solvers for different regularization parameters λ. For large λ (a), the stochastic algorithms (*BCFW* and *SSG*) perform considerably better than the batch solvers (*cutting plane* and *FW*). For a small λ (c), even the batch solvers achieve a lower objective earlier on than *SSG*. Our proposed *BCFW* algorithm achieves a low objective in both settings. (d) shows the convergence for CoNLL with the first passes in more details. Here *BCFW* already results in a low objective even after seeing only few datapoints. The advantage is less clear for the test error in (e) though, where *SSG-wavg* does surprisingly well. Finally, (f) compares the methods for the matching prediction task.

in the caption, while additional experiments can be found in Appendix F. In most of the experiments, the *BCFW-wavg* method dominates all competitors. The superiority is especially striking for the first few iterations, and when using a small regularization strength λ, which is often needed in practice. In term of test error, a peculiar observation is that the weighted average of the iterates seems to help both methods significantly: *SSG-wavg* sometimes slightly outperforms *BCFW-wavg* despite having the worst objective value amongst all methods. This phenomenon is worth further investigation.

7. Related Work

There has been substantial work on dual coordinate descent for SVMs, including the original sequential minimal optimization (SMO) algorithm. The SMO algorithm was generalized to structural SVMs (Taskar, 2004, Chapter 6), but its convergence rate scales badly with the size of the output space: it was estimated as $O\left(n|\mathcal{Y}|/\lambda\varepsilon\right)$ in Zhang et al. (2011). Further, this method requires an expectation oracle to work with its factored dual parameterization. As in our algorithm, Rousu et al. (2006) propose updating one training example at a time, but using multiple Frank-Wolfe updates to optimize along the subspace. However, they do not obtain any rate guarantees and their algorithm is less general because it again requires an expectation oracle. In the degenerate *binary* SVM case, our block-coordinate Frank-Wolfe algorithm is actually equivalent to the method of Hsieh et al. (2008), where because each datapoint has a unique dual variable, exact coordinate optimization can be accomplished by the line-search step of our algorithm. Hsieh et al. (2008) show a local linear convergence rate in the dual, and our results complement theirs by providing a global *primal* convergence guarantee for their algorithm of $O\left(1/\varepsilon\right)$. After our paper had appeared on arXiv, Shalev-Shwartz & Zhang (2012) have proposed a generalization of dual coordinate descent applicable to several regularized losses, including the structural SVM objective. Despite being motivated from a different perspective, a version of their algorithm (Option II of Figure 1) gives the exact same step-size and update direction as *BCFW* with line-search, and their Corol-

Table 1. Convergence rates given in the *number of calls to the oracles* for different optimization algorithms for the structural SVM objective (1) in the case of a Markov random field structure, to reach a specific accuracy ε measured for different types of gaps, in term of the number of training examples n, regularization parameter λ, size of the label space $|\mathcal{Y}|$, maximum feature norm $R := \max_{i,\boldsymbol{y}} \|\boldsymbol{\psi}_i(\boldsymbol{y})\|_2$ (some minor terms were ignored for succinctness). Table inspired from (Zhang et al., 2011). Notice that only stochastic subgradient and our proposed algorithm have rates independent of n.

Optimization algorithm	**Online**	**Primal/Dual**	**Type of guarantee**	**Oracle type**	**# Oracle calls**
dual extragradient (Taskar et al., 2006)	no	primal-'dual'	saddle point gap	Bregman projection	$O\left(\frac{nR\log\|\mathcal{Y}\|}{\lambda\varepsilon}\right)$
online exponentiated gradient (Collins et al., 2008)	yes	dual	expected dual error	expectation	$O\left(\frac{(n+\log\|\mathcal{Y}\|)R^2}{\lambda\varepsilon}\right)$
excessive gap reduction (Zhang et al., 2011)	no	primal-dual	duality gap	expectation	$O\left(nR\sqrt{\frac{\log\|\mathcal{Y}\|}{\lambda\varepsilon}}\right)$
BMRM (Teo et al., 2010)	no	primal	≥primal error	maximization	$O\left(\frac{nR^2}{\lambda\varepsilon}\right)$
1-slack SVM-Struct (Joachims et al., 2009)	no	primal-dual	duality gap	maximization	$O\left(\frac{nR^2}{\lambda\varepsilon}\right)$
stochastic subgradient (Shalev-Shwartz et al., 2010a)	yes	primal	primal error w.h.p.	maximization	$\tilde{O}\left(\frac{R^2}{\lambda\varepsilon}\right)$
this paper: block-coordinate Frank-Wolfe	yes	primal-dual	expected duality gap	maximization	$O\left(\frac{R^2}{\lambda\varepsilon}\right)$ Thm. 3

lary 3 gives a similar convergence rate as our Theorem 3. Balamurugan et al. (2011) propose to approximately solve a quadratic problem on each example using *SMO*, but they do not provide any rate guarantees. The *online-EG* method implements a variant of dual coordinate descent, but it requires an expectation oracle and Collins et al. (2008) estimate its primal convergence at only $O\left(1/\varepsilon^2\right)$.

Besides coordinate descent methods, a variety of other algorithms have been proposed for structural SVMs. We summarize a few of the most popular in Table 1, with their convergence rates quoted in number of oracle calls to reach an accuracy of ε. However, we note that almost no guarantees are given for the optimization of structural SVMs with approximate oracles. A regret analysis in the context of online optimization was considered by Ratliff et al. (2007), but they do not analyze the effect of this on solving the optimization problem. The cutting plane algorithm of Tsochantaridis et al. (2005) was considered with approximate maximization by Finley & Joachims (2008), though the dependence of the running time on the the approximation error was left unclear. In contrast, we provide guarantees for batch subgradient, cutting plane, and block-coordinate Frank-Wolfe, for achieving an ε-approximate solution as long as the error of the oracle is appropriately bounded.

8. Discussion

This work proposes a novel randomized block-coordinate generalization of the classic Frank-Wolfe algorithm for optimization with block-separable constraints. Despite its potentially much lower iteration cost, the new algorithm achieves a similar convergence rate in the duality gap as the full Frank-Wolfe method. For the dual structural SVM optimization problem, it leads to a simple online algorithm that yields a solution to an issue that is notoriously difficult to address for stochastic algorithms: no step-size sequence needs to be tuned since the optimal step-size can be efficiently computed in closed-form. Further, at the cost of an additional pass through the data (which could be done alongside a full Frank-Wolfe iteration), it allows us to compute a duality gap guarantee that can be used to decide when to terminate the algorithm. Our experiments indicate that empirically it converges faster than other stochastic algorithms for the structural SVM problem, especially in the realistic setting where only a few passes through the data are possible.

Although our structural SVM experiments use an exact maximization oracle, the duality gap guarantees, the optimal step-size, and a computable bound on the duality gap are all still available when only an appropriate approximate maximization oracle is used. Finally, although the structural SVM problem is what motivated this work, we expect that the block-coordinate Frank-Wolfe algorithm may be useful for other problems in machine learning where a complex objective with block-separable constraints arises.

Acknowledgements. We thank Francis Bach, Bernd Gärtner and Ronny Luss for helpful discussions, and Robert Carnecky for the 3D illustration. MJ is supported by the ERC Project SIPA, and by the Swiss National Science Foundation. SLJ and MS are partly supported by the ERC (SIERRA-ERC-239993). SLJ is supported by a Research in Paris fellowship. MS is supported by a NSERC postdoctoral fellowship.

References

Bach, F., Lacoste-Julien, S., and Obozinski, G. On the equivalence between herding and conditional gradient algorithms. In *ICML*, 2012.

Balamurugan, P., Shevade, S., Sundararajan, S., and Keerthi, S. A sequential dual method for structural SVMs. In *SDM*, 2011.

Caetano, T.S., McAuley, J.J., Cheng, Li, Le, Q.V., and Smola, A.J. Learning graph matching. *IEEE PAMI*, 31 (6):1048–1058, 2009.

Clarkson, K. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. *ACM Transactions on Algorithms*, 6(4):1–30, 2010.

Collins, M., Globerson, A., Koo, T., Carreras, X., and Bartlett, P. L. Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. *JMLR*, 9:1775–1822, 2008.

Dunn, J.C. and Harshbarger, S. Conditional gradient algorithms with open loop step size rules. *Journal of Mathematical Analysis and Applications*, 62(2):432–444, 1978.

Finley, T. and Joachims, T. Training structural SVMs when exact inference is intractable. In *ICML*, 2008.

Frank, M. and Wolfe, P. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

Gärtner, B. and Jaggi, M. Coresets for polytope distance. *ACM Symposium on Computational Geometry*, 2009.

Hsieh, C., Chang, K., Lin, C., Keerthi, S., and Sundararajan, S. A dual coordinate descent method for large-scale linear SVM. In *ICML*, pp. 408–415, 2008.

Jaggi, M. *Sparse convex optimization methods for machine learning*. PhD thesis, ETH Zürich, 2011.

Jaggi, M. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *ICML*, 2013.

Joachims, T., Finley, T., and Yu, C. Cutting-plane training of structural SVMs. *Machine Learn.*, 77(1):27–59, 2009.

Lacoste-Julien, S., Schmidt, M., and Bach, F. A simpler approach to obtaining an O(1/t) convergence rate for the projected stochastic subgradient method. Technical Report 1212.2002v2 [cs.LG], arXiv, December 2012.

Mangasarian, O.L. Machine learning via polyhedral concave minimization. Technical Report 95-20, University of Wisconsin, 1995.

Nesterov, Yurii. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Ouyang, H. and Gray, A. Fast stochastic Frank-Wolfe algorithms for nonlinear SVMs. *SDM*, 2010.

Rakhlin, A., Shamir, O., and Sridharan, K. Making gradient descent optimal for strongly convex stochastic optimization. In *ICML*, 2012.

Ratliff, N., Bagnell, J. A., and Zinkevich, M. (Online) subgradient methods for structured prediction. In *AISTATS*, 2007.

Rousu, J., Saunders, C., Szedmak, S., and Shawe-Taylor, J. Kernel-based learning of hierarchical multilabel classification models. *JMLR*, 2006.

Sang, E.F.T.K. and Buchholz, S. Introduction to the CoNLL-2000 shared task: Chunking, 2000.

Shalev-Shwartz, S. and Zhang, T. Proximal stochastic dual coordinate ascent. Technical Report 1211.2717v1 [stat.ML], arXiv, November 2012.

Shalev-Shwartz, S., Singer, Y., Srebro, N., and Cotter, A. Pegasos: primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127(1), 2010a.

Shalev-Shwartz, S., Srebro, N., and Zhang, T. Trading accuracy for sparsity in optimization problems with sparsity constraints. *SIAM Journal on Optimization*, 20: 2807–2832, 2010b.

Shamir, O. and Zhang, T. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *ICML*, 2013.

Taskar, B. *Learning structured prediction models: A large margin approach*. PhD thesis, Stanford, 2004.

Taskar, B., Guestrin, C., and Koller, D. Max-margin Markov networks. In *NIPS*, 2003.

Taskar, B., Lacoste-Julien, S., and Jordan, M. I. Structured prediction, dual extragradient and Bregman projections. *JMLR*, 7:1627–1653, 2006.

Teo, C.H., Vishwanathan, S.V.N., Smola, A.J., and Le, Q.V. Bundle methods for regularized risk minimization. *JMLR*, 11:311–365, 2010.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, 2005.

Zhang, X., Saha, A., and Vishwanathan, S. V. N. Accelerated training of max-margin Markov networks with kernels. In *ALT*, pp. 292–307. Springer, 2011.