Minding the Gaps for Block Frank-Wolfe Optimization of Structured SVMs

Anton Osokin*, Jean-Baptiste Alayrac*, Isabella Lukasewitz, Puneet K.Dokania, Simon Lacoste-Julien

INRIA/ENS, Paris, France

*- equal contribution

SUMMARY

Block-Coordinate Frank-Wolfe (**BC-FW**) [1] is a popular algorithm for constrained optimization over **block-separable** domains.
Examples: dual of **Structured SVM**, multiple-sequence alignment, etc.

Key insight: Frank-Wolfe (FW) block gaps indicate **suboptimality** on blocks and we use them to design **adaptive** algorithms.

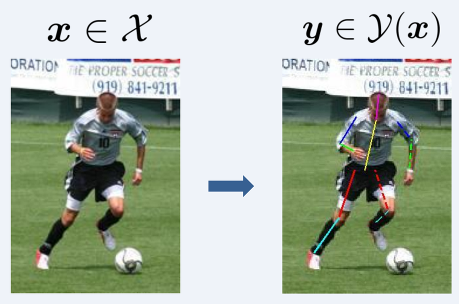
Contributions:

i. **Adaptive sampling** of blocks using FW gaps
ii. **Caching** the oracle calls with gap-based criterion
iii. **Pairwise** and **away** steps in BC setting
iv. **Regularization path** for Structured SVM

(i–iii) **Improving BC-FW**

Structured SVM

Structured prediction



Given a joint "**structured**" feature map $\phi(x, y) \in \mathbb{R}^d$, construct a linear classifier
$h_w(x) = \operatorname{argmax}_{y \in \mathcal{Y}(x)} \langle w, \phi(x, y) \rangle$
to predict well w.r.t. loss $L(y, y')$

Structured SVM problem

Primal

$$\min_{w} \quad \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^{n} \overbrace{\underbrace{\max_{y \in \mathcal{Y}_i}}_{\text{max oracle}} L_i(y) - \langle w, \underbrace{\phi(x_i, y_i) - \phi(x_i, y)}_{=: \psi_i(y)} \rangle}^{\text{structured hinge loss}}$$

Primal-dual link

$$w^* = A\alpha^*$$

Dual

$$\min_{\substack{\alpha \in \mathbb{R}^m \\ \alpha \succcurlyeq 0}} \quad f(\alpha) := \frac{\lambda}{2}\|A\alpha\|^2 - b^\mathsf{T}\alpha \qquad \text{s.t.} \quad \textstyle\sum_{y \in \mathcal{Y}_i} \alpha_i(y) = 1 \quad \forall i \in [n]$$

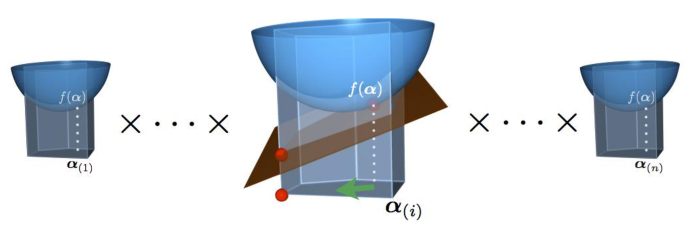
$A := \{\frac{1}{\lambda n}\psi_i(y) \in \mathbb{R}^d \mid i \in [n], y \in \mathcal{Y}_i\}$; $b := (\frac{1}{n}L_i(y))_{i \in [n], y \in \mathcal{Y}_i}$

⚠ **Exponential number of variables!**

BC-FW

General setting

$$\min \quad f(\alpha) \qquad \text{s.t.} \quad \alpha \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(1)}$$



Frank-Wolfe gap:

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

Algorithm

- Select object
- Linear FW oracle
- Select step size

Advantages $\mathcal{O}(1/T)$ **rate!**

+ Sparse iterates
+ Dual gap guarantee
+ Online

⚠ **Not only limited to SSVM!**

Applied to SSVM

Nice properties

+ Analytic line search
+ Dual gap matches FW gap

Key property

$$\max_{s_{(i)} \in \mathcal{M}^{(i)}} \langle s_{(i)}, -\nabla_{(i)} f(\alpha^{(k)}) \rangle = \frac{1}{n} \max_{y \in \mathcal{Y}_i} L_i(y) - \langle w^{(k)}, \psi_i(y) \rangle.$$

Linear FW oracle $\Leftrightarrow$ **Max oracle**

I) Gap sampling

Key property: $g(\alpha) = \sum_{i=1}^{n} g_i(\alpha)$ **Separable FW gap!**

Motivation

Use **block gaps** to **pick an object** at each iteration.

Sample objects proportional to block gaps

Adaptive scheme: sampling depends on past computation.

Theory

When using **exact** block gaps, **convergence rate** $\mathcal{O}(1/T)$ is multiplied by:

$$\chi^{\otimes} := \max_{k} \mathbf{E}\left[\frac{\chi(C_f^{(:)})}{\chi(g_:(\alpha^{(k)}))^3}\right]$$

+ Lower bound n times faster!

$x \in \mathbb{R}^n$, $p := \frac{x}{\|x\|_1}$, $\chi(x) := \sqrt{1 + n^2 \operatorname{Var}[p]} \in [1, \sqrt{n}]$

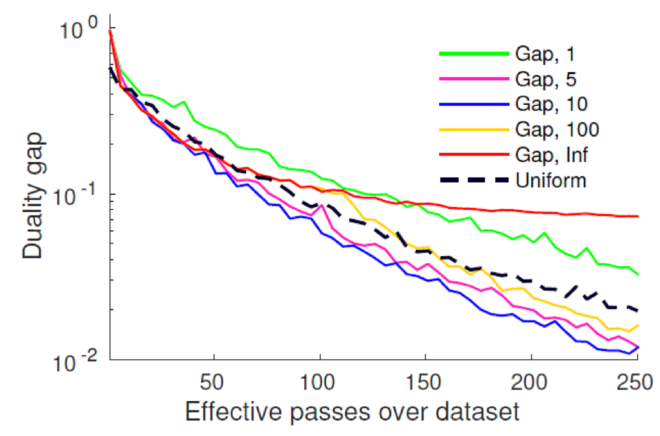
Practice

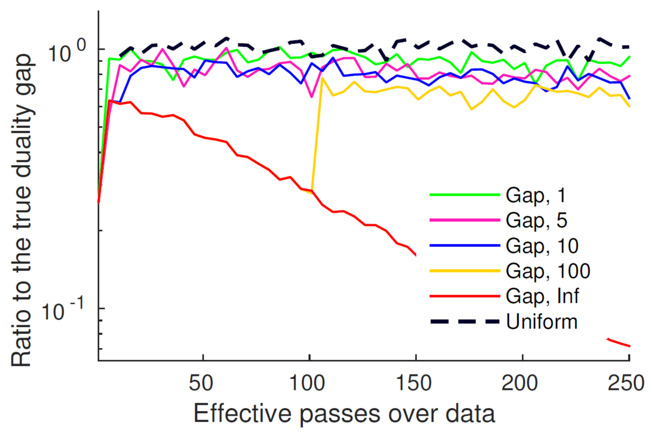
Problem: Maintaining all exact gaps at all time is too costly (O(n) per iteration).

Current solution: update **one gap at a time**. Update **all gaps** every 10 passes with a batch pass.

Consequence: **Exploitation** versus **staleness** trade-off.

Open problem: how to **analyze** the staleness effect?





II) Caching oracle

Motivation

Problem: The **oracle** is often **bottleneck**

Solution: cache the outputs of the oracle and reuse them

Cache oracle: $y_i^c := \operatorname{argmax}_{y \in \mathcal{C}_i} L_i(y) - \langle w, \psi_i(y) \rangle$

$\mathcal{C}_i \subset \mathcal{Y}_i \qquad |\mathcal{C}_i| \ll |\mathcal{Y}_i|$

Possible improvement:
$g(\alpha) := \langle \alpha - s(y_i^c), \nabla f(\alpha) \rangle$

Overheads for maintaining cache

Safety rate of $\mathcal{O}(1/T)$

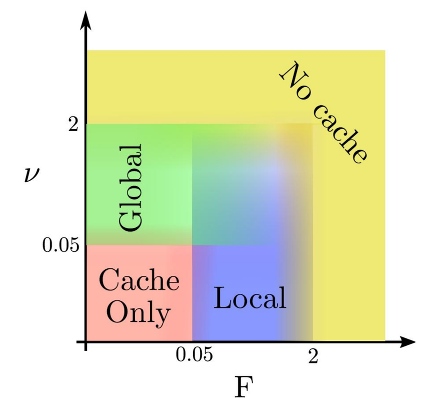
Open question: add cache cost in analysis

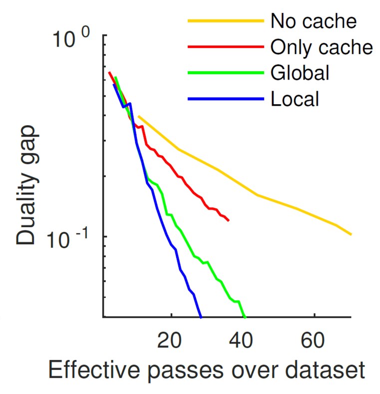
Adaptive criterion

When to use the cache?

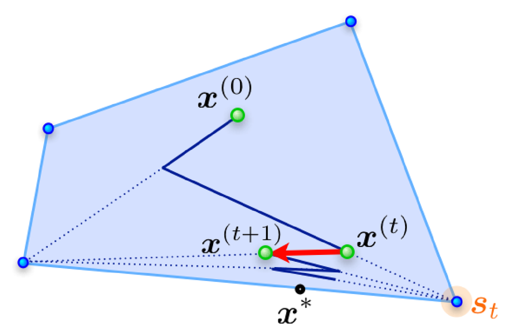
$$\hat{g}_i^{(k)} \geq \max\left(F g_i^{(k_i)}, \frac{\nu}{n} g^{(k_0)}\right)$$

Cache gap — **Local** criterion based on last oracle call — **Safety** criterion (convergence)

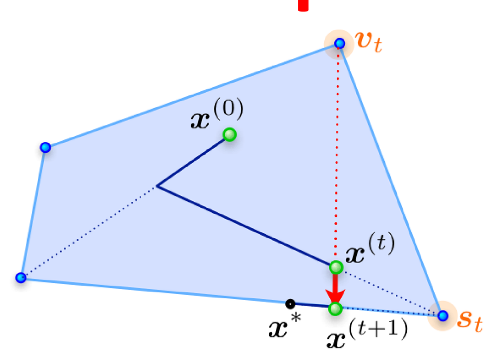




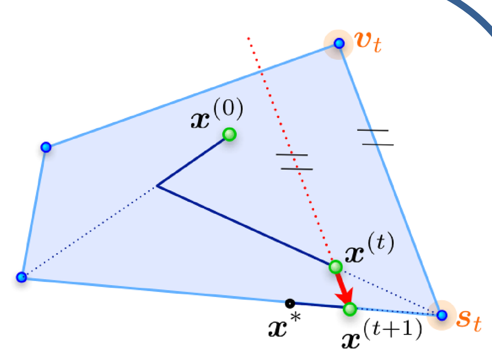
III) Pairwise steps for BC-FW



Zig-zagging problem of FW



Away steps (**A-FW**)
See [2] for review



Pairwise steps (**P-FW**)

Motivation

Slow convergence of FW

FW variants (away and pairwise steps) enjoy a **linear** convergence in the **batch case.**

We proposed **BC-A-FW** and **BC-P-FW**.

Overheads are shared with cache!

Theory vs. Practice

Bad news: linear convergence can be proven only under uncheckable assumptions, need a new proof technique.

Good news: observe empirical linear convergence behavior.

Experimental results

Extensive benchmark

- Comparison of **8** methods
- Tested on **4** structured prediction datasets:

$\underbrace{\textbf{OCR / CoNLL}}_{\text{NLP}}$ / $\underbrace{\textbf{HorseSeg / LSP}}_{\text{Vision}}$

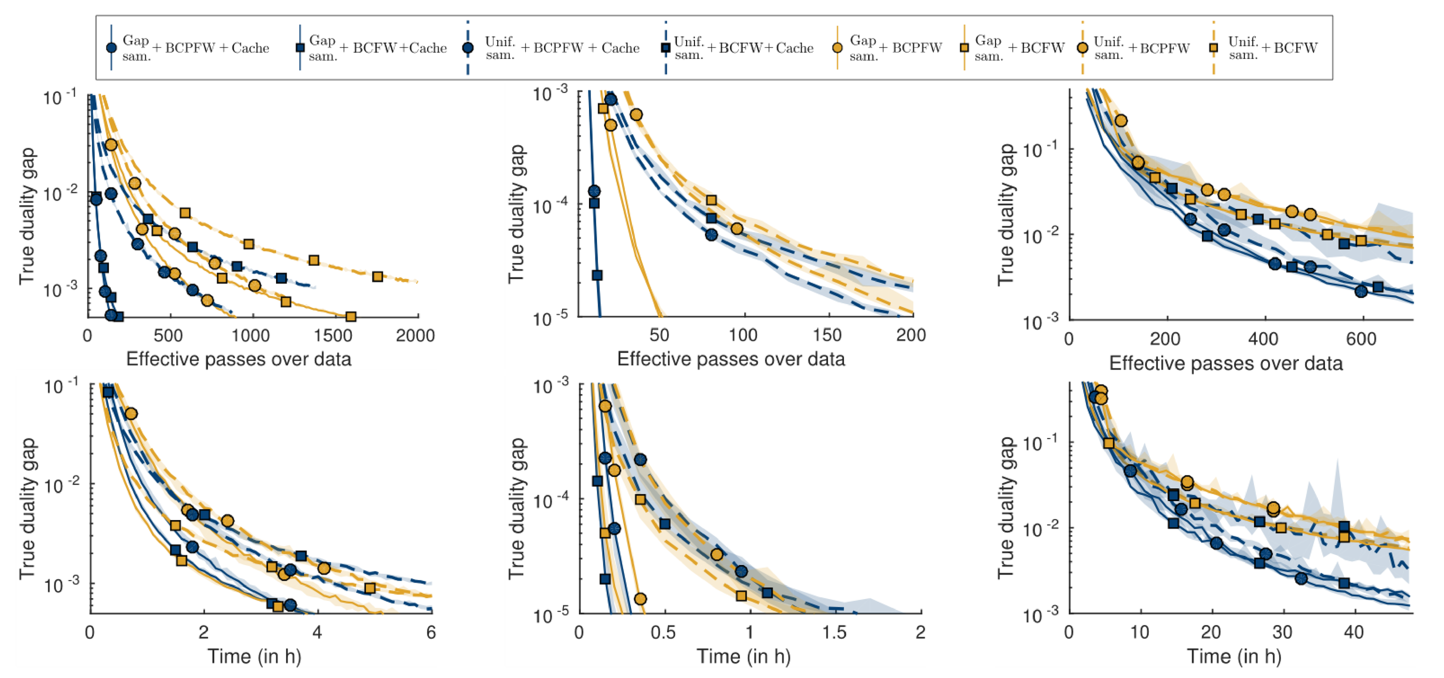
Conclusions

1. **Gap sampling** always helps!
2. **Caching** really helps if **slow oracle** (cost of our naïve implementation)
3. **BC-P-FW** helps for **high accuracy** and **stronger** when more strongly convex!

RECOMMANDATIONS

(a) BC-P-FW + gap sampling + caching (when **slow oracle**)

(b) BC-FW + gap sampling (when **fast oracle**)



(a) OCR-large, $\lambda = 0.001$, $n = 6251, d = 4082$ (b) HorseSeg-medium, $\lambda = 10$, $n = 6121, d = 1969$ (c) LSP-small, $\lambda = 100$, $n = 100, d = 2676$

IV) Regularization path

Motivation

What is it? Get $w^*(\lambda)$ for all λ

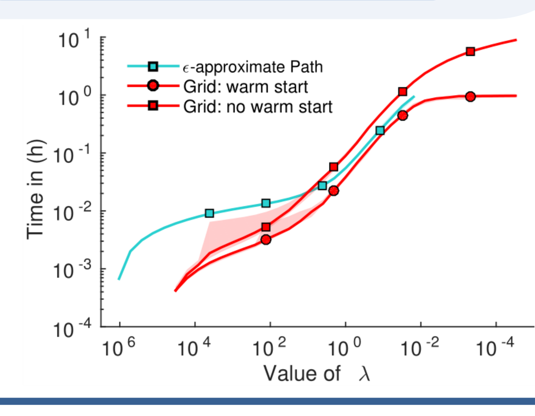
Why? Better than grid search, but usually expensive.

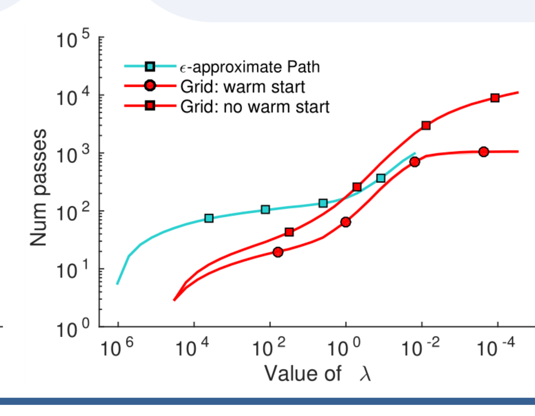
Our approach: use **piecewise constant** ϵ-approximate path

Key insight: can control gap change when decreasing λ while keeping parameter w constant: $w = A(\lambda)\alpha(\lambda)$

Algorithm

1. **Initialization:** find smallest breakpoint λ_1 s.t. $\frac{\lambda_1}{\lambda} w^1$ is ϵ-approximate for $\lambda \geq \lambda_1$
2. **Iterate:**
   i. At a breakpoint, automatically pick the next one such that the gap stays smaller than ϵ for constant w.
   ii. Optimize for this λ with any solver to get gap of $\kappa\epsilon$ with $\kappa < 1$.





Project webpage

References

[1] Lacoste-Julien, Jaggi, Schmidt, Pletscher. Block-coordinate Frank-Wolfe optimization for structural SVMS, *ICML 2013.*

[2] Lacoste-Julien, Jaggi. On the global linear convergence of Frank-Wolfe optimization variants, *NIPS 2015.*