Frank-Wolfe algorithms for SVMs: complexity and practical strategies

Emanuele Frandi

Department of Electrical Engineering
KU Leuven

emanuele.frandi@esat.kuleuven.be

Workshop on Data-driven Modeling Methods and Applications
July 14-15, 2014, Leuven

Introduction

The Frank-Wolfe algorithm [Frank and Wolfe (1956)] is a first-order method to maximize a concave differentiable function on a bounded, convex set Σ.

- Recent surge of interest in the context of Machine Learning, for its sparsity properties [Clarkson (2008), Jaggi (2013)].
- Well-suited for SVM training [F., Ñanculef *et al.* (2010, 2013, 2014), Ouyang and Gray (2010), Kumar and Yildirim (2011), Jaggi and Lacoste-Julien (2013)].

Introduction

L_2-SVM dual problem: maximization of a convex quadratic function

$$\max_{\alpha \in \Sigma} \; f(\alpha) := -\frac{1}{2}\alpha^T K \alpha \;,$$

where Σ is the unit simplex in $\mathbb{R}^m$:

$$\sum_{i=1}^{m} \alpha_i = 1 \,, \; \alpha \geq 0 \,.$$

- For classification we have $K_{i,j} = y_i y_j k(x_i, x_j) + y_i y_j + \delta_{i,j}/C$, where k is a kernel function and C is a regularization parameter.

Frank-Wolfe optimization

Given the current iterate $\alpha_k \in \Sigma$, a standard Frank-Wolfe iteration consists in the following steps.

1. Find a search direction d_k by maximizing a local first-order approximation:

$$y_k = \operatorname*{argmax}_{u \in \Sigma} \; f(\alpha_k) + \nabla f(\alpha_k)^T (u - \alpha_k), \quad d_k^{FW} = y_k - \alpha_k.$$

2. Perform a line-search $\lambda_k = \operatorname{argmax}_{\lambda \in [0,1]} f(\alpha_k + \lambda d_k^{FW})$.
3. Update the iterate by

$$\alpha_{k+1} = \alpha_k + \lambda_k d_k^{FW} = (1 - \lambda_k)\alpha_k + \lambda_k y_k \, .$$

Theoretical results (I)

Proposition (Sublinear convergence - FW)

Let α^* be an optimal solution. Then, for sufficiently large k, we have

$$f(\alpha^*) - f(\alpha_k) \leq \frac{D}{k+2}, \qquad D \text{ constant.}$$

- Primal gap is not a computable quantity ($f(\alpha^*)$ unknown)
- Stopping criterion on duality gap:

$$\Delta_d(\alpha_k) := \max_i \nabla f(\alpha_k)_i - \alpha_k^T \nabla f(\alpha_k) \leq \varepsilon \ ,$$

where $\varepsilon > 0$ is a given tolerance parameter.

Relation between primal and dual gap

- Dual gap is an upper bound of primal gap

$$0 \leq f(\alpha^*) - f(\alpha_k) \leq \Delta_d(\alpha_k).$$

- Primal gap is monotonic if a line-search is used. Dual gap is not, but has the same asymptotic bound [Jaggi (2013)].

Proposition

Let α^* be an optimal solution. Then, for sufficiently large k, we have

$$\Delta_d(\alpha_k) \leq \frac{B}{k+2}, \qquad B \text{ constant.}$$

Theoretical results (II)

Proposition (Bound on the total number of iterations)

The algorithm terminates after at most

$$K \leq Q + \frac{M}{\varepsilon}$$

iterations, where Q, M are constants independent of m and ε.

- Sparsity certificate: α_k has at most k nonzero coordinates after iteration k, and at most $\mathcal{O}(1/\epsilon)$ after stopping.
- Convergence rate and total number of iterations are independent from the problem size.

Complexity of the algorithm

- The computational cost lies almost entirely in the linear maximization (i.e. the computation of the FW vertex).
- Focusing on the SVM case, we have $y_k = e_{i^*}$, where

$$i^* \in \operatorname*{argmax}_{i=1,\dots,m} \nabla f(\alpha_k)_i = \operatorname*{argmax}_{i=1,\dots,m} -\sum_{j \in \mathcal{I}_k} K_{i,j}(\alpha_k)_j$$

- Cost of an iteration: $\mathcal{O}(m|\mathcal{I}_k|) \leq \mathcal{O}(mk)$, proportional to the number of examples.
- Since the algorithm stops in $\mathcal{O}(1/\epsilon)$ iterations, total complexity is $\lesssim \mathcal{O}(m/\epsilon^2)$

Clearly not feasible for large datasets...

Randomized Frank-Wolfe

- To avoid the dependence from m, we need to explore only a fixed number of vertices.
- Solution: extract a sample $S \subset \{1, \dots, m\}$ and solve

$$i_S \in \operatorname*{argmax}_{i \in S} \nabla f(\alpha_k)_i \, .$$

- The cost of an iteration is bounded by $\mathcal{O}(|S|k)$, and that of the overall procedure by $\mathcal{O}(|S|/\epsilon^2)$.

Randomized Frank-Wolfe

Justification for random sampling [Schölkopf and Smola (2002)]:

Theorem

Let $D := \{d_1, \ldots, d_m\} \subset \mathbb{R}$ be a set of cardinality m, and let $D' \subset D$ be a random subset of size r. Then the probability that $\max D'$ *is greater or equal than $\tilde{m}$ elements of D is at least $1 - (\frac{\tilde{m}}{m})^r$.*

- In our case, $D = \{\nabla f(\alpha_k)_1, \ldots, \nabla f(\alpha_k)_m\}$ and $D' = \{\nabla f(\alpha_k)_i \,|\, i \in S\}$.
- For example, if $|S| \approx 60$, then, with probability at least 0.95, $\nabla f(\alpha_k)_{i_s}$ is between the 5% largest gradient components.

Issues with a randomized step

- In general, S will not contain the "true" i^*. The randomized FW will compute an approximate index i_S.
- The true dual gap is not computable anymore. We have to use the approximate quantity

$$\Delta_S(\alpha_k) := \nabla f(\alpha_k)_{i_S} - \alpha_k^T \nabla f(\alpha_k).$$

- In general $\Delta_S(\alpha_k) \leq \Delta_d(\alpha_k)$. We can't be sure that $f(\alpha^*) - f(\alpha_k) \leq \Delta_S(\alpha_k)$.
- Tradeoff: computational cost against risk of early stopping.

Experimental results

Question: can the algorithm properties (convergence rate, number of iterations) be preserved in the stochastic case?

Time for some experiments:

- Experiments run on a 3.40 GHz Intel i7 machine with 16 GB RAM.
- Code written in Julia (version 0.3.0).
- Tolerance parameter: $\epsilon = 10^{-4}$. C selected on a logarithmic grid $[2^4, \ldots, 2^{12}]$. RBF kernel used for all the problems.
- Results averaged over 10 runs.

Experimental results

Dataset		Full sampling	240 points	120 points	60 points
heart	Acc	97.8%	97.8%	98.2%	97.8%
($m = 270$)	Time	11.40 s	10.48 s	5.56 s	2.69 s
	Iter	6926	7104	7071	6526
	SVs	134	132	134	133
breast	Acc	99.1%	99.1%	98.8%	97.2%
cancer	Time	11.83 s	4.62 s	0.66 s	0.02s
($m = 683$)	Iter	4759	4871	1181	96
	SVs	83	83	85	56
ionosphere	Acc	100%	100%	100%	99.4%
($m = 351$)	Time	9.79 s	7.16 s	3.72 s	0.31 s
	Iter	7568	7772	6837	962
	SVs	83	85	97	106
fourclass	Acc	99.5%	99.5%	99.2%	99.6%
($m = 862$)	Time	71.72 s	18.68 s	9.10 s	5.07 s
	Iter	7048	6358	5859	5520
	SVs	267	265	269	283

Experimental results

Some observations:

- SV set size and accuracy are fairly stable. Classification model very similar to that of the full sampling case.
- The number of samples needed to avoid the early stopping phenomenon is problem-dependent.

Dataset		240 points	120 points	60 points
heart	Iter (mean)	7104	7071	6526
	2σ-interval	[6854,7354]	[6705,7437]	[5956,7096]
breast cancer	Iter (mean)	4759	1181	96
	2σ-interval	[4075,5443]	[919,1443]	[30,162]

Experimental results

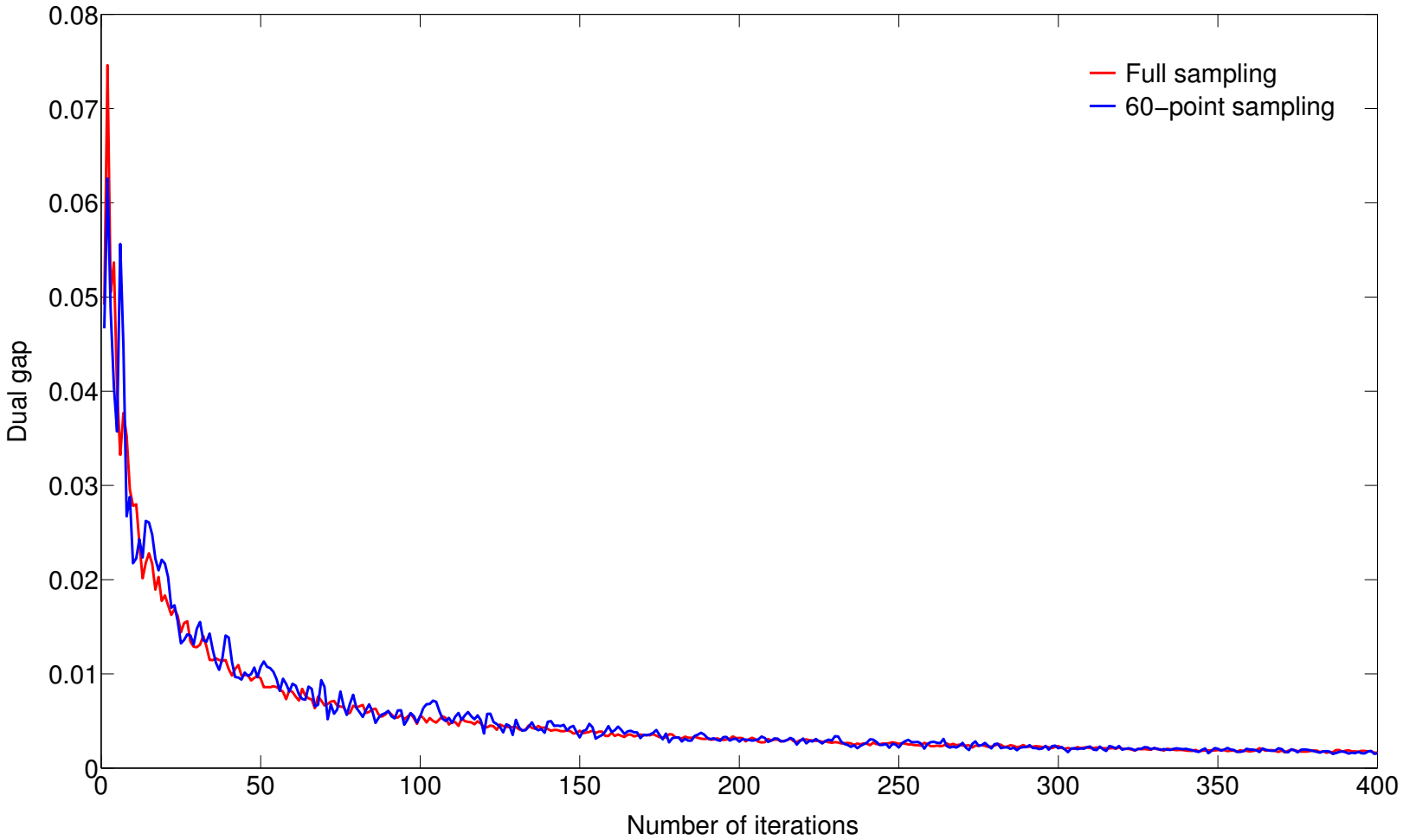


Figure : Behaviour of dual gap on the **heart** dataset (first 400 iterations)

Experimental results

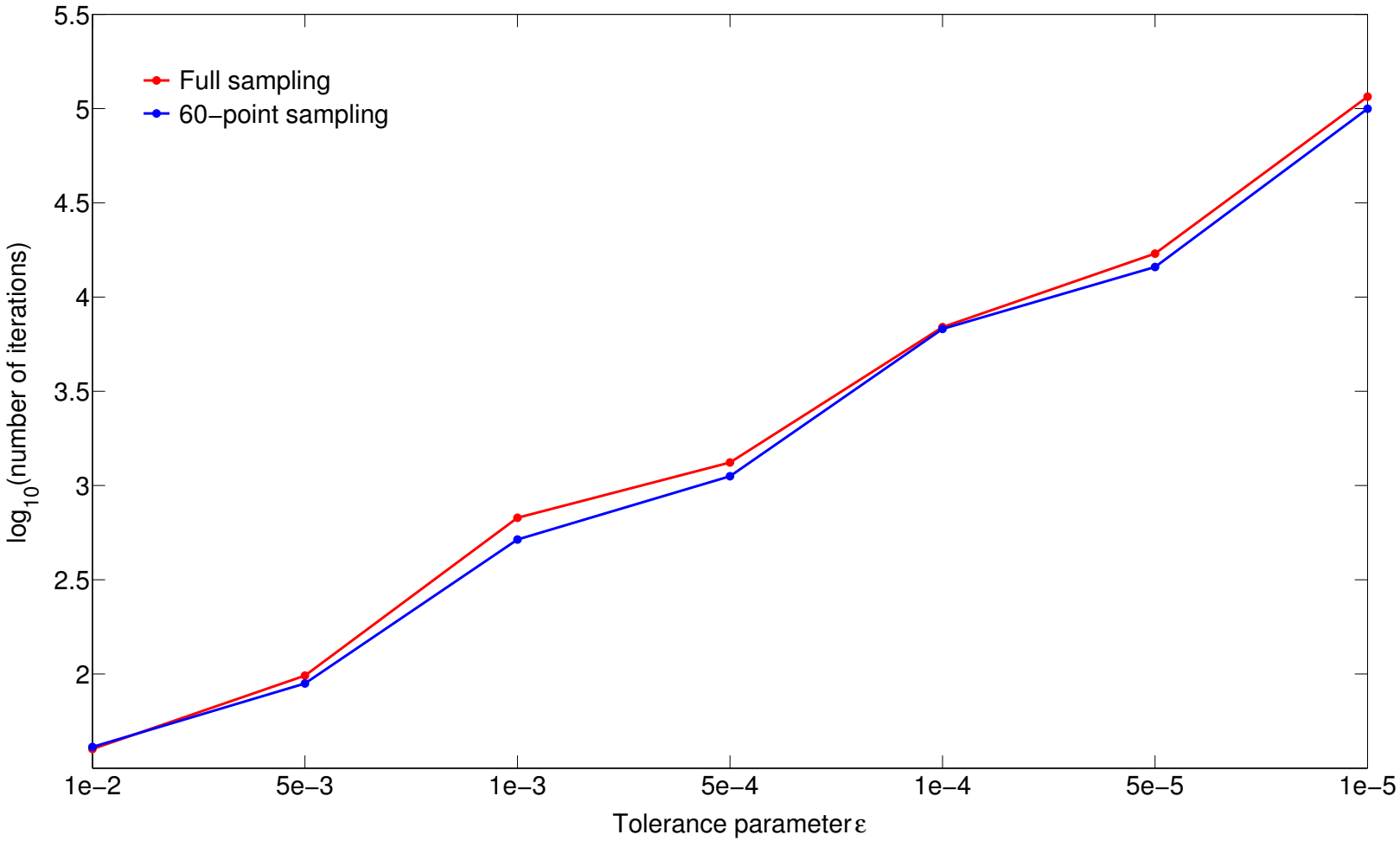


Figure : Tolerance vs. total iterations on the **heart** dataset

Experimental results

- Overall, experiments suggest that the stochastic variant works well.
- Conjecture:

Let α^* be an optimal solution. Then, for sufficiently large k, we have

$$f(\alpha^*) - \mathbb{E}(f(\alpha_k)) \leq \mathcal{O}\left(\frac{1}{k}\right),$$

where the expectation is taken w.r.t. the selected vertex i_S.

To do:

- Analyze the effect of the problem size: since our probability bounds are independent from m, we expect similar results as m increases.
- Find a sound way to determine $|S|$ as a parameter.

Concluding remarks

- Frank-Wolfe based algorithms are robust alternatives to several classical SVM solvers.
- Random sampling techniques are needed to make it feasible on large problems.
- Experiments show that this is possible without sacrificing the accuracy of the model.
- However, a rigorous convergence proof is still lacking.

Thank you very much!