A Fast and Scalable Polyatomic Frank-Wolfe Algorithm for the LASSO Problem

Adrian Jarret, Julien Fageot, Matthieu Simeoni

Laboratory of AudioVisual Communications - EPFL Center for Imaging
École Polytechnique Fédérale de Lausanne (EPFL)

Abstract- We propose a fast and scalable ***Polyatomic Frank-Wolfe (P-FW)*** algorithm for the resolution of high-dimensional LASSO regression problems. This algorithm improves upon traditional Frank-Wolfe methods by considering generalized greedy steps with ***polyatomic*** updates, hence allowing for a more efficient exploration of the search space. To preserve ***sparsity of the intermediate iterates***, we re-optimize the LASSO problem over the set of selected atoms at each iteration. For efficiency reasons, the accuracy of this re-optimization step is relatively low for early iterations and gradually increases with the iteration count. We provide ***convergence guarantees*** for our algorithm and validate it in simulated compressed sensing setups. Our experiments reveal that P-FW outperforms state-of-the-art methods in terms of runtime, both for FW methods and optimal first-order proximal gradient methods such as the *Fast Iterative Soft-Thresholding Algorithm (FISTA)*.
The code is accessible on GitHub: *AdriaJ/PolyatomicFW SPL*

1. The LASSO Problem

Context: The LASSO optimization problem is commonly used to solve linear inverse problems with a sparsity prior, such as sparse reconstruction in signal processing or variable selection in statistics.

$$\arg\min_{\mathbf{x}\in\mathbb{R}^N} \quad \frac{1}{2}\|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2+\lambda\|\mathbf{x}\|_1$$

Measurements vector	$\mathbf{y}\in\mathbb{R}^L$
Measurement operator	$\mathbf{H}\in\mathbb{R}^{L\times N}$
Regularization parameter	$\lambda>0$

Motivation: Scalability

This problem does not admit closed-form solution and thus requires to be solved numerically. We currently benefit from fast algorithms, such as FISTA, that are able to reach a convergence rate of $\mathcal{O}(1/k^2)$.
In some high dimensional applications, it becomes difficult, sometimes impossible, to apply these methods, mainly for memory requirements.
We study the Frank-Wolfe algorithm, an optimization method with a greedy behavior, in order to circumvent this limitation.

Selected references:
- M. Jaggi, "Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization," in *Proceedings of the 30th International Conference on Machine Learning*, Feb. 2013, pp. 427–435.
- Q. Denoyelle, V. Duval, G. Peyré, and E. Soubies, "The sliding Frank–Wolfe algorithm and its application to super-resolution microscopy," *Inverse Problems*, vol. 36, no. 1, p. 014001, Jan. 2020.
- M. Simeoni et al. (2021). GitHub: matthieumeo/pycsou: Pycsou 1.0.6

2. The Vanilla Frank-Wolfe Algorithm (FW)

The standard algorithm: A greedy behavior

$$\min_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x})$$

Algorithm 1: Vanilla Frank-Wolfe Algorithm (V-FW)

Initialize $\mathbf{x}_0\in\mathcal{D}$
for $k=1,2\cdots$ **do**
① 1) Find an update direction: "Atom"
$\mathbf{s}_k\in\arg\min_{\mathbf{s}\in\mathcal{D}}\langle\nabla f(\mathbf{x}_k),\mathbf{s}\rangle$
② 2.a) Step size: $\gamma_k\leftarrow\frac{2}{k+2}$
2.b) Reweight: $\mathbf{x}_{k+1}\leftarrow(1-\gamma_k)\mathbf{x}_k+\gamma_k\mathbf{s}_k$
end

① Atom creation ② Weights estimation

Sparse iterates: Scalability $\quad \mathbf{x}_{k+1}=\sum_{i=0}^{k}\alpha_{i,k}\mathbf{s}_i$

FW for the LASSO: Shape of the atoms

When applying FW to the LASSO, the atoms created at step 1 are canonical basis vectors of $\mathbb{R}^N$ (scaled with a multiplicative signed factor): at each iteration, the algorithm updates a single coordinate.

$$\mathbf{s}_k=\pm M\mathbf{e}_{i_k}$$

Limitations: A lot of coordinates are visited many times, leading to redundancy and oscillations, and thus slowing down the convergence.

4. Comparison

Vanilla FW	Polyatomic FW
① Only one atom per iteration $\mathbf{s}_k$	① Allow several atoms $\mathcal{I}_k=\{\mathbf{s}_1^{(k)},\mathbf{s}_2^{(k)},\dots\}$
② Convex combination of the weights $\mathbf{x}_{k+1}\leftarrow(1-\gamma_k)\mathbf{x}_k+\gamma_k\mathbf{s}_k$	② Re-estimation over all the weights $\mathbf{x}_{k+1}\leftarrow\arg\min_{\mathrm{Supp}(\mathbf{x})\subset\mathcal{S}_k} f(\mathbf{x})$

Partial correction: During this step of PFW, we run an instance of an optimization algorithm (ISTA) to re-evaluate all the weights. We enforce an early stopping criterion to keep the iterations quick, that gets more precise along the iterations to obtain a fine convergence.

3. A Polyatomic Variant (PFW)

Our algorithm: faster convergence, no oscillation

Algorithm 5: Polyatomic FW (P-FW) of quality $\delta>0$

Initialize: $\mathbf{x}_0\leftarrow 0,\mathcal{S}_0\leftarrow\emptyset$
for $k=1,2\cdots$ **do**
$\gamma_k\leftarrow 2/(k+2)$
① 1".a) Polyatomic exploration:
$\mathcal{I}_k:\ \langle\nabla f(\mathbf{x}_k),\mathbf{s}_k\rangle\in\left[\min_{\mathbf{s}\in\mathcal{D}}\langle\nabla f(\mathbf{x}_k),\mathbf{s}\rangle,\min_{\mathbf{s}\in\mathcal{D}}\langle\nabla f(\mathbf{x}_k),\mathbf{s}\rangle+\delta\gamma_k\right]$
1".b) Update active indices: $\mathcal{S}_k\leftarrow\mathcal{S}_{k-1}\cup\mathcal{I}_k$
② 2".a) Set accuracy threshold: $\varepsilon_k=\varepsilon_0\gamma_k$
2".b) Update active weights:
$\mathbf{x}_{k+1/2}\leftarrow(1-\gamma_k)\mathbf{x}_k+\gamma_k\mathbf{s}_k$
$\mathbf{x}_{k+1}\leftarrow\texttt{partial\_correction}(\mathbf{x}_{k+1/2},\mathcal{S}_k,\varepsilon_k)$
$k\leftarrow k+1$
end

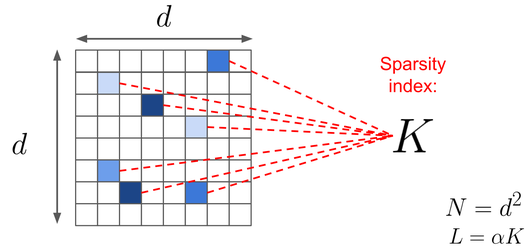
Convergence guarantee:

Theorem 1: *(Convergence of Polyatomic Frank-Wolfe)*

- Let : $\begin{cases} f(\mathbf{x})=\frac{1}{2}\|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2+\lambda\|\mathbf{x}\|_1 \\ \mathbf{x}^*\in\arg\min f(\mathbf{x}) \end{cases}$
- Then : $f(\mathbf{x}_k)-f(\mathbf{x}^*)\leq\frac{2}{k+2}(C_f+2\delta)$

5. Simulation setup

Random sparse image: $\mathbf{x}_0$



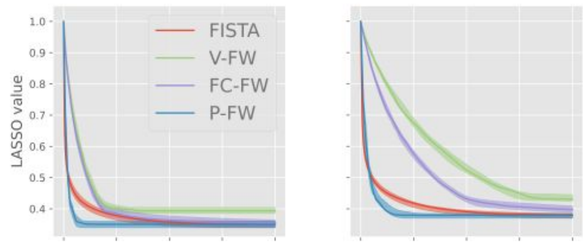
$N=d^2$
$L=\alpha K$

Random measurement matrix: $\mathbf{H}\in\mathbb{R}^{L\times N}$

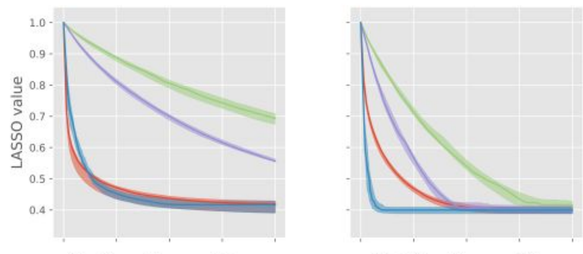
Simulated measurements: $\mathbf{y}=\mathbf{H}\mathbf{x}_0+\mathbf{w}$ ← Gaussian noise

6. Numerical results

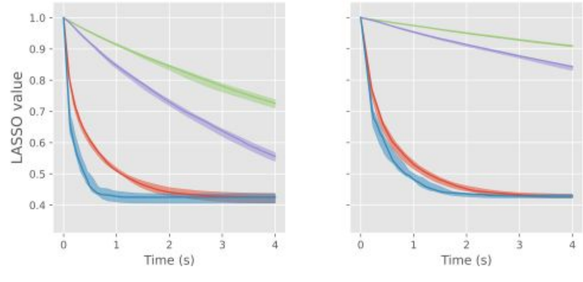
Experiences: We report the minimum value of the LASSO objective function obtained with four different algorithms with respect to running time. We repeat the experiences for different ground truth sparsity index K and different number of measurements L.



(a) $K=32,\alpha=16$ (b) $K=32,\alpha=64$



(c) $K=64,\alpha=16$ (d) $K=64,\alpha=64$



(e) $K=128,\alpha=16$ (f) $K=128,\alpha=64$

Conclusion: *PFW* performs better than the other FW algorithms (*Vanilla* et *Fully-Corrective* variants). The performances are equivalent to FISTA and usually better in the context of sparse solutions.
We also report that *PFW* is more likely to scale better than proximal methods like FISTA in large dimension due to its sparse iterates.