Une version polyatomique de l'algorithme Frank-Wolfe pour résoudre le problème LASSO en grandes dimensions

Adrian JARRET[1], Matthieu SIMEONI[2], Julien FAGEOT[1]

[1]Laboratoire des Communications AudioVisuelles
Bâtiment BC, Route Cantonale, 1015 Lausanne, Suisse

[2]EPFL Center for Imaging
Bâtiment BM, Station 17, 1015 Lausanne, Suisse

adrian.jarret@epfl.ch, matthieu.simeoni@epfl.ch, julien.fageot@epfl.ch

Résumé – Nous nous intéressons à la reconstruction parcimonieuse d'images à l'aide du problème d'optimisation régularisé LASSO. Dans de nombreuses applications pratiques, les grandes dimensions des objets à reconstruire limitent, voire empêchent, l'utilisation des méthodes de résolution proximales classiques. C'est le cas par exemple en radioastronomie. Nous détaillons dans cet article le fonctionnement de l'algorithme *Frank-Wolfe Polyatomique*, spécialement développé pour résoudre le problème LASSO dans ces contextes exigeants. Nous démontrons sa supériorité par rapport aux méthodes proximales dans des situations en grande dimension avec des mesures de Fourier, lors de la résolution de problèmes simulés inspirés de la radio-interférométrie.

Abstract – We consider the problem of recovering sparse images by means of the penalised optimisation problem LASSO. For various practical applications, it is impossible to rely on the proximal solvers commonly used for that purpose due to the size of the objects to recover, as it is the case for radio astronomy. In this article we explain the mechanisms of the *Polyatomic Frank-Wolfe algorithm*, specifically designed to minimise the LASSO problem in such challenging contexts. We demonstrate in simulated problems inspired from radio-interferometry the preeminence of this algorithm over the proximal methods for high dimensional images with Fourier measurements.

1 Introduction

La reconstruction parcimonieuse de signaux ou d'images présente de nombreux intérêts, que ce soit sur le plan pratique ou théorique. Entre autres, elle induit souvent une sélection de variables, ce qui permet d'obtenir des modèles interprétables avec un plus grand pouvoir de généralisation. Utiliser une hypothèse de parcimonie *a priori* peut également aider à modéliser certains phénomènes physiques. L'application qui nous intéresse dans cet article est la reconstruction de sources lumineuses ponctuelles sur des images du ciel en radio-interférométrie pour l'astronomie. Une méthode encore largement utilisée pour ce problème est donnée par l'algorithme CLEAN [1], qui met l'accent sur la parcimonie en produisant une solution par l'ajout successif de composants, appelés *atomes*, issus d'un dictionnaire de reconstruction.

Une autre méthode, désormais classique pour la reconstruction parcimonieuse en traitement du signal, consiste à résoudre un problème inverse sous la forme d'un problème d'optimisation pénalisé, dont le terme de pénalité induit de la parcimonie dans les solutions. Nous nous focalisons sur le problème suivant, connu sous le nom de LASSO ou *Basis Pursuit* :

$$\arg\min_{\mathbf{x}\in\mathbb{R}^N} \frac{1}{2}\left\|\mathbf{y}-\mathbf{G}\mathbf{x}\right\|_2^2 + \lambda\left\|\mathbf{x}\right\|_1, \tag{1}$$

dans lequel $\mathbf{G}$ représente la modélisation linéaire d'un système d'acquisition et $\mathbf{y} \in \mathbb{C}^L$ est supposé être le résultat bruité de la mesure par $\mathbf{G}$ d'un certain signal source $\mathbf{x}_0$, de telle sorte que $\mathbf{y} \approx \mathbf{G}\mathbf{x}_0$. Le paramètre λ présente l'avantage de pouvoir contrôler l'équilibre entre fidélité avec les données et parcimonie de la solution. En effet, avec ce choix de pénalité, il a été prouvé l'existence de solutions au plus K-parcimonieuses, avec K inférieur au nombre de mesures L [2, Théorème 6].

Cette dernière approche n'a cependant reçu que peu de considération en radio-interférométrie. En effet, les principales méthodes actuelles de résolution du LASSO, telles que APGD [3] ou FISTA [4], sont basées sur des algorithmes proximaux et manipulent des solutions intermédiaires denses. Les grandes dimensions des données manipulées en radioastronomie rendent ainsi prohibitif le coût en mémoire de l'application directe de ces méthodes. Des approche plus élaborées ont été proposées, notamment de la descente par blocs de coordonnées [5].

Dans cet article, nous présentons le fonctionnement de l'algorithme **Frank-Wolfe Polyatomique**, récemment proposé dans [6], pour résoudre le problème LASSO. En conservant une structure parcimonieuse pour ses solutions intermédiaires, cet algorithme est particulièrement adapté lorsque la taille des pro-

Ce travail a été financé par le Fonds national Suisse (FNS) à travers les bourses CRSII5_193826 AstroSignals (A. Jarret and M. Simeoni), 200 021 181 978/1, "SESAM - Sensing and Sampling : Theory and Algorithms" (M. Simeoni) et P400P2_194364 (J. Fageot).

blèmes limite l'usage des méthodes usuelles. Nous étendons les cadres d'application dans lesquels cet algorithme démontre sa supériorité, en le confrontant à des problèmes d'optimisation simulés mettant en jeu des mesures de Fourier. Ces situations, inspirées de la radioastronomie, sont numériquement exigeantes à la fois par leur dimension et la nature complexe des données manipulées.

2 Frank-Wolfe Polyatomique

Afin de présenter le fonctionnement de l'algorithme *Frank-Wolfe Polyatomique* (P-FW), nous détaillons d'abord la version classique, souvent appelée *Vanilla Frank-Wolfe (V-FW)*.

Frank-Wolfe, un algorithme glouton. L'algorithme de Frank-Wolfe (FW) [7], dans sa version originale, résout un problème d'optimisation de la forme

$$\arg\min_{\mathbf{x}\in\mathcal{D}} f(\mathbf{x}) \tag{2}$$

avec f une fonction de coût convexe et continûment différentiable et $\mathcal{D}$ un sous-ensemble compact convexe d'un espace de Banach.

Il est connu que le problème LASSO (1) peut être reformulé sous la forme (2), en le modifiant légèrement pour le rendre différentiable et en introduisant l'espace de recherche

$$\mathcal{D} = \{\mathbf{x} \in \mathbb{R}^N : \|\mathbf{x}\|_1 \leq M\}$$

avec $M = \|\mathbf{y}\|_2^2 / 2\lambda$ [8,9]. On obtient ainsi l'algorithme d'optimisation V-FW fourni en Algorithme 1. La quantité $\boldsymbol{\eta}_k = (1/\lambda)\mathbf{G}^*(\mathbf{y} - \mathbf{G}\mathbf{x}_k)$ est appelée *certificat dual empirique* et $\mathbf{e}_i$ représente le i-ème vecteur de la base canonique de $\mathbb{R}^N$.

Algorithme 1 : V-FW pour le LASSO

Initialisation : $\mathbf{x}_0 = \mathbf{0}$
pour $k = 1, 2 \cdots$ **faire**
1) Déterminer une direction d'avancement :
$\mathbf{s}_k = \pm M \mathbf{e}_{i_k} \in \arg\max_{\mathbf{s}\in\mathcal{D}} \langle \boldsymbol{\eta}_k, \mathbf{s} \rangle$
2.a) Taille du pas : $\gamma_k \leftarrow \frac{2}{k+2}$
2.b) Pondération : $\mathbf{x}_{k+1} \leftarrow (1-\gamma_k)\mathbf{x}_k + \gamma_k \mathbf{s}_k$
fin

Plus en détail, cet algorithme se compose de deux étapes, identifiées dans Algorithme 1 par 1), la recherche de nouvelle direction de descente, et 2.a)-b), l'estimation de la nouvelle solution intermédiaire, comme combinaison convexe de la solution actuelle et de cette direction de descente. On désigne ainsi les directions de descente $\mathbf{s}_k$ identifiées à l'étape 1) en tant qu'*atomes*. Le comportement *glouton* des algorithmes FW provient de la stratégie de sélection de ces atomes, en minimisant l'approximation au premier ordre de la fonction objectif évaluée en la position actuelle.

L'algorithme V-FW converge en terme de fonction objectif à la vitesse $\mathcal{O}(1/k)$, où k est le nombre d'itérations [10], ce qui est inférieur aux méthodes proximales actuelles. En contraste, APGD converge à la vitesse $\mathcal{O}(1/k^2)$ [3]. Les méthodes FW présentent cependant la propriété de conserver au cours des itérations des solutions intermédiaires parcimonieuses, comme combinaison linéaires d'atomes, et de fait peuvent se révéler performantes en grandes dimensions, quand les algorithmes proximaux sont ralentis par leurs solutions intermédiaires denses (besoins en mémoire plus importants, calculs plus longs).

Algorithme 2 : FW Polyatomique pour le LASSO

Initialisation : $\mathbf{x}_0 \leftarrow 0, \mathcal{S}_0 \leftarrow \emptyset$
pour $k = 1, 2 \cdots$ **faire**
$\gamma_k \leftarrow \frac{2}{k+2}$
1'.a) Exploration polyatomique :
$\mathcal{I}_k = \{1 \leq j \leq N : |\boldsymbol{\eta}_k[j]| \geq \|\boldsymbol{\eta}_k\|_\infty - \delta\gamma_k\}$
1'.b) Actualisation des indices actifs : $\mathcal{S}_k \leftarrow \mathcal{S}_{k-1} \cup \mathcal{I}_k$
2'.a) Réglage du seuil de précision : $\varepsilon_k = \varepsilon_0 \gamma_k$
2'.b) Évaluation des poids actifs :
$\mathbf{x}_{k+1} \leftarrow \arg\min \frac{1}{2}\|\mathbf{y} - \mathbf{G}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1$
avec la contrainte $\mathrm{Supp}(\mathbf{x}) \in \mathcal{S}_k$,
résolution interrompue au seuil de précision ε_k
fin

Résoudre le problème LASSO avec un variant polyatomique. Le variant *Frank-Wolfe Polyatomique* présenté dans [6] a été conçu pour tirer profit au maximum du comportement glouton de V-FW, et ainsi identifier rapidement les atomes importants dans la construction d'une solution au LASSO, tout en conservant des solutions intermédiaires parcimonieuses, pour rester applicable en grandes dimensions. La procédure numérique est présentée en Algorithme 2[1], avec $\delta > 0$ un paramètre de reconstruction à adapter aux données considérées.

Le coeur de l'algorithme P-FW réside dans l'étape 1'.a), au cours de laquelle, contrairement à FW original, il est possible de rajouter plusieurs atomes à la solution intermédiaire actuelle. Les atomes identifiés par P-FW ayant la même forme que ceux identifiés par V-FW, considérer plusieurs atomes candidats revient à autoriser la solution intermédiaire à accéder à de nouvelles coordonnées d'indices $\mathcal{I}_k$, non visitées jusque là. L'ensemble des coordonnées accessibles après cette mise-à-jour à l'étape k est noté $\mathcal{S}_k$ (étape 1'.b)).

Au cours de l'étape 2'.b), l'algorithme attribue un poids aux derniers atomes ajoutés et ajuste les poids des atomes précédents. Pour cela, il ré-optimise la fonction objectif du LASSO en ne considérant que les solutions dont le support est contraint à vivre parmi les coordonnées identifiées actives $\mathcal{S}_k$. Cela revient à résoudre un nouveau sous-problème LASSO, dont la dimension est extrêmement réduite par rapport au problème initial. Numériquement, nous utilisons l'algorithme d'optimisation ISTA [4], initialisé avec la solution intermédiaire actuelle. De plus, pour réduire le temps de calcul et tenir compte du fait que

[1] Une version plus détaillée et reproductible de l'algorithme est fournie dans [6].

la solution partielle évolue beaucoup d'une itération à l'autre aux cours des premières itérations, nous arrêtons cette procédure d'estimation des poids prématurément, à l'aide du critère d'arrêt ε_k[2]. Ce critère d'arrêt est décroissant pour permettre, à terme, une convergence fine de P-FW.

Convergence. En s'appuyant sur les résultats présentés dans l'article [10], nous sommes capables de garantir que la séquence des solutions intermédiaires produites par P-FW converge en terme de fonction objectif vers un minimum du problème LASSO. La vitesse de convergence théorique est du même ordre que pour l'algorithme FW classique [6, Théorème 1].

Theorem 1 (Convergence de P-FW) *Soit $\mathbf{x}_k$ la suite des solutions intermédiaires produites par P-FW. Notons $\mathcal{L}(\mathbf{x}) = \frac{1}{2}\|\mathbf{y}-\mathbf{G}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1$ la fonction objectif du problème LASSO et $\mathcal{L}^*$ sa valeur optimale. Nous avons alors que*

$$\mathcal{L}(\mathbf{x}_k) - \mathcal{L}^* = \mathcal{O}\left(\frac{1}{k}\right)$$

Dans nos expériences (Section 4), nous observons que la convergence est souvent plus efficace que APGD dans les régimes considérés.

3 Implémentation

Afin d'évaluer les performances de P-FW, nous le comparons à APGD pour résoudre le problème LASSO sur des données simulées. Les simulations sont réalisées en Python, à l'aide de la librairie d'optimisation `Pycsou` [11]. Cette dernière permet de résoudre facilement des problèmes inverses à l'aide d'algorithmes proximaux puissants. Le logiciel implémente de façon très modulaire les principaux éléments constituants de problèmes génériques d'optimisation convexe (fonctionnelles de coût, termes de pénalisation et opérateurs linéaires). En particulier, pour utiliser les algorithmes fournis par `Pycsou`, il est nécessaire de définir l'application directe et l'application de l'adjoint de l'opérateur de mesure, liant les données à l'inconnue du problème inverse.

Nous détaillons dans cette section les subtilités à prendre en compte pour l'implémentation.

Opérateur de mesure. Pour nos simulations, nous utilisons comme opérateur de mesure un sous-échantillonnage aléatoire de la transformée de Fourier discrète (2D) de l'image d'entrée. Ce choix nous place dans un cadre d'étude similaire à celui de la radio-interférométrie, dont l'opérateur est généralement modélisé par des mesures de Fourier (non uniformes en fréquence).

Soit une image $\mathbf{x} \in \mathbb{R}^N$ de taille $N = n \times n$ et L coordonnées fréquentielles $(u_\ell, v_\ell) \in [0, n-1]^2$, l'opérateur de mesure est donné par

$$\begin{aligned}(\mathbf{G}\mathbf{x})[\ell] &= \frac{1}{n}\sum_{p=0}^{n-1}\sum_{q=0}^{n-1}\mathbf{x}[p,q]\exp\left(-j\frac{2\pi}{n}(u_\ell p + v_\ell q)\right) \\ &= \mathrm{DFT}_{2\mathrm{D}}(\mathbf{x})[u_\ell, v_\ell].\end{aligned} \tag{3}$$

Par définition, cet opérateur est à valeurs complexes, sa signature est donnée par $\mathbf{G} : \mathbb{R}^N \to \mathbb{C}^L$. Il convient de définir l'adjoint $\mathbf{G}^* : \mathbb{C}^L \to \mathbb{R}^N$ par rapport au produit scalaire hermitien sur $\mathbb{C}^L$:

$$\forall \mathbf{a}, \mathbf{b} \in \mathbb{C}^L, \quad \langle \mathbf{a}, \mathbf{b}\rangle_{\mathbb{C}^L} = \Re\left(\overline{\mathbf{a}}^T\mathbf{b}\right)$$

où $\Re : \mathbb{C} \to \mathbb{R}$ désigne la partie réelle d'un nombre complexe. On obtient l'expression de l'opérateur adjoint à implémenter :

$$\forall \mathbf{z} \in \mathbb{C}^L, \quad \mathbf{G}^*\mathbf{z} = \Re\left(\overline{\mathbf{G}}^T\mathbf{z}\right) \tag{4}$$

Pour une meilleure lisibilité, les opérations ont ici été définies sur des images en deux dimensions, mais sont en pratique implémentées avec les versions aplaties et unidimensionnelles des mêmes images.

Correction des poids : résoudre un problème LASSO à support réduit avec Pycsou. Lors de l'étape 2'.b) de P-FW, les poids sont estimés en cherchant la solution optimale au problème LASSO en limitant l'espace de recherche à l'enveloppe convexe des atomes estimés jusqu'à l'itération actuelle (se référer à [6] pour plus de détails). On peut ainsi se ramener à résoudre le problème LASSO à support réduit suivant :

$$\underset{\widetilde{\mathbf{x}}\in\mathbb{R}^{\mathrm{Card}(\mathcal{S}_k)}}{\arg\min}\ \frac{1}{2}\|\mathbf{y} - \mathbf{G}_{\mathcal{S}_k}\widetilde{\mathbf{x}}\|_2^2 + \lambda\|\widetilde{\mathbf{x}}\|_1 \tag{5}$$

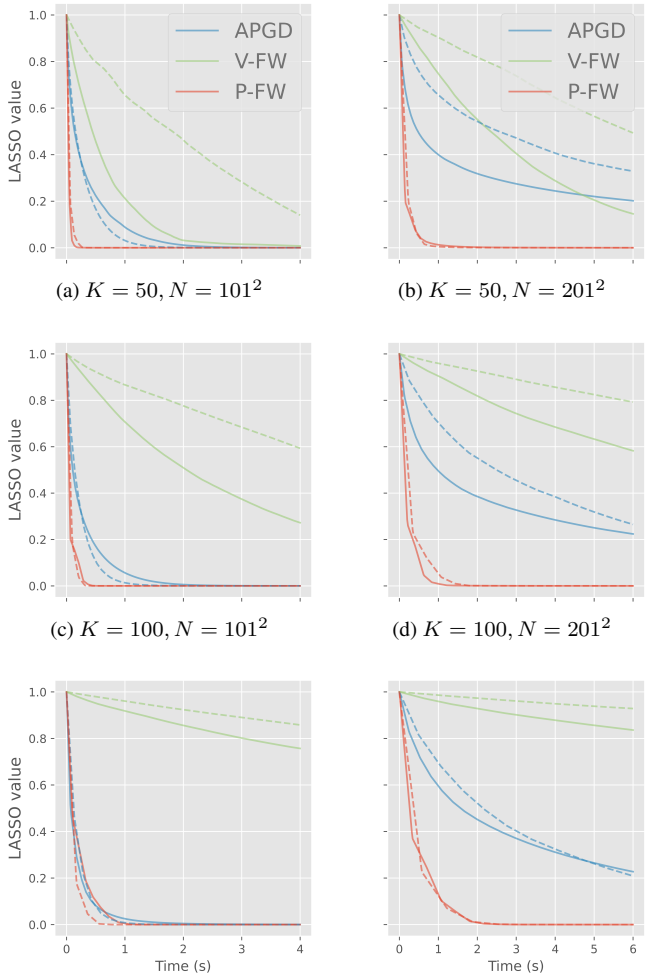
Numériquement, il suffit de récupérer les colonnes d'intérêt de la matrice $\mathbf{G}$, ainsi notée $\mathbf{G}_{\mathcal{S}_k}$, pour contraindre le support de la solution de ce sous-problème. On obtient un problème dont la taille est nettement inférieure au problème initial, peu gourmand en mémoire (en pratique $\mathrm{Card}(\mathcal{S}_k)$ est au maximum de l'ordre de la parcimonie des solutions, d'où $\mathrm{Card}(\mathcal{S}_k) \ll N$). Nous utilisons ensuite l'implémentation de ISTA fournie par `Pycsou` pour obtenir les poids de la prochaine solution intermédiaire.

4 Résultats

Nous regardons l'évolution de la valeur de la fonction objectif au fil du temps lors de la résolution du problème LASSO. À titre de comparaison, nous reportons les performances de V-FW et de APGD en parallèle de celles obtenues avec P-FW. Quelques corrections mineures ont été intégrées par rapport [6], le code pour reproduire les expériences est accessible en [12].

Le contexte expérimental est le suivant. Une fois la matrice de mesure $\mathbf{G}$ calculée selon la procédure détaillée en Section 3, les données sont obtenues selon l'équation $\mathbf{y} = \mathbf{G}\mathbf{x}_0 + \mathbf{w} \in \mathbb{C}^L$, où $\mathbf{x}_0 \in \mathbb{R}^N = \mathbb{R}^{n\times n}$ est une image aléatoire ne comportant que K pixels allumés (indice de parcimonie K) et $\mathbf{w}$ est la réalisation d'un vecteur aléatoire gaussien i.i.d. à valeurs complexes (PSNR

[2] Contrairement à FISTA, ISTA a l'avantage de garantir la décroissance de la fonction objectif à chaque itération, ce qui assure une amélioration de l'itérée même avec notre stratégie d'arrêt prématuré.



(a) $K = 50, N = 101^2$ (b) $K = 50, N = 201^2$

(c) $K = 100, N = 101^2$ (d) $K = 100, N = 201^2$

(e) $K = 200, N = 101^2$ (f) $K = 200, N = 201^2$

FIGURE 1 – Valeur normalisée du problème LASSO au cours du temps de résolution, pour les 3 algorithmes étudiés (APGD, V-FW et P-FW) et différentes tailles d'images N et valeurs de parcimonie sous-jacentes K. Les traits pleins et en pointillés correspondent respectivement à $\alpha = 8$ et $\alpha = 16$. Temps de calculs obtenus avec une machine ThinkPad T14, Intel Core i7 (4C/8T) @ 1.8GHz et 32GB RAM.

de 20dB). Le nombre de mesures est donné par $L = \alpha K$ avec $\alpha = 8$ ou $\alpha = 16$. Les résultats numériques sont présentés en Figure 1.

Sur tous les scénarios considérés, V-FW présente toujours les moins bonnes performances, et converge rarement dans les temps impartis. En revanche, la version polyatomique présente d'excellents résultats, et est systématiquement plus rapide que APGD (à l'exception du contexte e), dans lequel les performances sont comparables). Les situations (b) et (d) sont les plus remarquables, P-FW converge en moins de 1s, alors que APGD n'a atteint que 80% de sa marge de progression au bout de 6s.

Plus en détail, nous constatons que les performances de APGD dépendent peu de la parcimonie K de la source, mais plutôt de la dimension N du problème. Cet algorithme est beaucoup plus long à converger quand l'image d'entrée est de plus grande dimension ($N = 201 \times 201$) et quand le nombre de mesures augmente ($\alpha = 16$ ralentit la convergence en (b), (d) et (f)). À l'inverse, la vitesse de convergence de P-FW semble surtout dépendre de la parcimonie sous-jacente de la solution, l'algorithme étant plus efficace pour des petites valeurs de K. Cela peut s'expliquer par son comportement glouton, lui permettant d'identifier efficacement les degrés de liberté actifs de la solution.

5 Conclusion et perspectives

Dans ce travail, nous avons confronté l'algorithme Frank-Wolfe Polyatomique au problème LASSO dans un contexte de mesures fréquentielles de Fourier. Tout comme les expériences menées dans [6] pour de l'acquisition comprimée, les résultats obtenus démontrent la supériorité de l'algorithme par rapport aux méthodes proximales comme APGD dans les grandes dimensions étudiées.

Ces résultats encourageants représentent une étape intermédiaire dans le but d'appliquer l'algorithme Frank-Wolfe Polyatomique aux données réelles de radioastronomie. Dans cette idée, nous souhaitons développer des opérateurs de mesures de Fourier non uniformes plus rapides à l'aide de techniques de *Fast Fourier Trasnform (FFT)*. L'algorithme doit également être adapté en conséquence pour manipuler ce type d'opérateurs.

Références

[1] J. A. Högbom, "Aperture Synthesis with a Non-Regular Distribution of Interferometer Baselines," *Astronomy and Astrophysics Supplement Series*, vol. 15, p. 417, Jun 1974.

[2] M. Unser, J. Fageot, and H. Gupta, "Representer Theorems for Sparsity-Promoting ℓ_1 Regularization," *IEEE Transactions on Information Theory*, vol. 62, pp. 5167–5180, Sept. 2016.

[3] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[4] A. Beck and M. Teboulle, "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems," *SIAM Journal on Imaging Sciences*, vol. 2, pp. 183–202, Jan. 2009.

[5] P.-A. Thouvenin, A. Abdulaziz, A. Dabbech, A. Repetti, and Y. Wiaux, "Parallel faceted imaging in radio interferometry via proximal splitting (Faceted HyperSARA) : I. algorithm and simulations," *Monthly Notices of the Royal Astronomical Society*, Aug 2021.

[6] A. Jarret, J. Fageot, and M. Simeoni, "A Fast and Scalable Polyatomic Frank-Wolfe Algorithm for the LASSO," *IEEE Signal Processing Letters*, vol. 29, p. 637–641, 2022.

[7] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics Quarterly*, vol. 3, no. 1-2, pp. 95–110, 1956.

[8] 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, p. 014001, Jan. 2020.

[9] Z. Harchaoui, A. Juditsky, and A. Nemirovski, "Conditional gradient algorithms for machine learning," in *NIPS Workshop on Optimization for ML*, vol. 3, pp. 3–2, Citeseer, 2012.

[10] M. Jaggi, "Revisiting Frank-Wolfe : Projection-Free Sparse Convex Optimization," in *Proceedings of the 30th International Conference on Machine Learning*, pp. 427–435, PMLR, Feb. 2013.

[11] M. Simeoni, *matthieumeo/pycsou : Pycsou 1.0.6*. Zenodo, Apr 2021.

[12] `https://github.com/AdriaJ/PolyatomicFW_SPL`. Accessed : Mar 2022.