# Frank Wolfe Algorithm

Authors

Dylan Nico Ambrosi [578586]
Emiliano Quaranta [580667]

Emails: d.ambrosi1@studenti.unipi.it, e.quaranta1@studenti.unipi.it

December 2025

## 1 Introduction

In this project, we solve a quadratic nonseparable knapsack problem of the form

$$min\{x^TQx + Q^Tx : \sum_i a_ix_i \geq b, l \leq x \leq u\},$$

where $x, q, a \in \mathbb{R}^n,\ 0 \leq l < u,\ b \in \mathbb{R},\ Q \in \mathbb{R}^{n\times n}$ Positive Semidefinite, with a Frank Wolfe type algorithm.

## 2 Prerequisites

First, we have to verify some matching prerequisites for the function $f(x) = x^TQx + q^Tx$. This is a polynomial function with $Q \succeq 0$, so it's continuously differentiable. The gradient is computed as follow:

$$\nabla f(x) = 2Qx + q.$$

Moreover, we present two important results:

**Proposition 1** (f convex)**.** *Since $Q \succ 0$, then the function $f$ is strongly convex*

**Proposition 2** (L-smoothness)**.** *The function $f$ is L-Smooth with $L = 2\lambda_{max}(Q)$*

*Proof.* A function is L-Smooth if $||\nabla f(x) - \nabla f(y)|| \leq L||x - y||$. We have that the gradient is $\nabla f(x) = 2Qx + q$ and so $||2Q(x - y)|| \leq 2||Q|| \times ||x - y||$. If we wrap up everything we obtain that $L = 2||Q|| = 2\lambda_{max}(Q)$ □

The fact that $f$ is strongly convex assures that the only stationary point will be the global optimum. L-Smoothness, on the other hand, helps us to derive some interesting results on the convergence and will be useful later on.
An important requirement of the Frank–Wolfe algorithm concerns also the feasible set $D$, which must be convex and compact.

**Definition 1** (Convex set)**.** *A set $D$ is convex if for every $x, y \in D$ and for every $\theta \in [0,1]$, the point $\theta x + (1-\theta)y$ also belongs to $D$.*

In our case, the feasible set is defined by a linear constraint $a^T x \geq b$, which defines a half space, together with the box constraint $l \leq x \leq u$. Both the half–space and the box are convex sets, and their intersection is therefore convex. Moreover, the box constraint makes the set bounded, so $D$ is also compact. This property is essential for the Frank–Wolfe method: the algorithm minimizes a linear approximation of $f$ at each iteration, and a linear function has minimum only if it is bounded. Furthermore, since each update is a convex combination of feasible points, the convexity of $D$ guarantees that every new iterate produced by Frank–Wolfe remains inside the box.

## 3 Considerations about the step size

We decided to consider the most common value for the step size, widely described in literature. This value "Typically improves practical convergence" (Jaggi, 2015) and it is preferred because it computes the best possible step along the tomography (Frank Wolfe paper, 1956). We show the computation using the Exact Line Search over the tomography of the function $f$. We minimize the univariate quadratic function

$$\begin{aligned}
\phi(\alpha) &= f(x + \alpha d) && (1)\\
&= (x+\alpha d)^T Q (x + \alpha d) + q^T (x + \alpha d) && (2)\\
&= x^T Q x + 2\alpha x^T Q d + \alpha^2 d^T Q d + q^T x + \alpha q^T d && (3)\\
&= (d^T Q d)\alpha^2 + (2x^T Q d + q^T d)\alpha + \text{const.} && (4)
\end{aligned}$$

The form is $\phi(\alpha) = a\alpha^2 + b\alpha + c$. If we set the derivative $\phi'(\alpha) = 2a\alpha + b = 0$, we obtain

$$\boxed{\alpha^* = -\frac{x^T Q d + \frac{1}{2} q^T d}{d^T Q d}}. \qquad (5)$$

Since $Q \succeq 0$, in the actual implementation we have

$$\alpha^* = min\Big(1, max\big(0, -\frac{x^T Q d + \frac{1}{2} q^T d}{d^T Q d}\big)\Big),$$

covering the case in which the denominator is equal to zero and the line search reduces to minizing a linear function over the interval $[0,1]$. In such scenario, since the direction $d$ is always a descent direction such that $\nabla f(x)^T d < 0$, the minimum is always at $\alpha^* = 1$.
We can now present the Frank Wolfe Algorithm with a pseudocode:

**Algorithm 1** Frank–Wolfe Algorithm

```
1: Input: initial point $x_0 \in D$, maximum iterations $max\_iter$
2: for $k = 0, 1, 2, \dots, T$ do
3:     Compute gradient $g_k = \nabla f(x_k)$
4:     Solve linear minimization oracle:
           $s_k = \arg\min_{s \in D} g_k^\top s$
5:     Set Frank–Wolfe direction $d_k = s_k - x_k$
6:     Compute step size $\alpha_k \in [0, 1]$ (e.g., exact line search)
7:     Update iterate:
           $x_{k+1} = x_k + \alpha_k d_k$
8: end for
9: Return $x_T$
```

## 4 Linear problem

In this section, we describe the linear solver for Frank Wolfe Algorithm, implemented by the function **solveLP()**. This is a simple knapsack problem of the form

$$s_k = argmin(g_k^T s), \quad a^T \times s \geq b, \quad l \leq s \leq u$$

where $g$ is the cost of a variable and $a$ is the value. This is solved by first choosing a feasible $s$ in the box, sorting efficient variables, where efficiency is defined as $\rho_i = a_i/g_i$, and finally increasing them by a partial or total factor, in order to reach the linear constraint $a^T * s = b$ with minimum cost. First, we check that it gives a descent direction:

**Proposition 3** (Descent direction)**.** *If $s_k$ is a solution of the linear minimization problem $\min_{s \in D} g_k^T s$, then the direction $d_k = s_k - x_k$ satisfies $g_k^T d_k \leq 0$. Moreover, equality holds iif $x_k$ is the optimim of $\min_{x \in D} f(x)$.*

*Proof.* Since $x_k \in D$ and $s_k$ is a minimizer of the linear function $s \mapsto g_k^\top s$ over the set $D$, we have

$$g_k^\top s_k \leq g_k^\top x_k.$$

Subtracting $g_k^\top x_k$ from both sides yields

$$g_k^\top (s_k - x_k) \leq 0.$$

By definition of the Frank–Wolfe direction $d_k = s_k - x_k$, this implies

$$g_k^\top d_k \leq 0.$$

If the inequality holds we have that

$$g_k^\top (s_k - x_k) = 0,$$

then

$$g_k^\top s \geq g_k^\top x_k = g_k^T (s - x_k) \geq 0 \qquad \forall s \in D,$$

which is the first order optimality condition for the convex optimization problem $\min_{x \in D} f(x)$. Hence, equality holds if and only if $x_k$ is an optimal solution. □

Since the linear subproblem has the structure of a continuous knapsack problem with box constraints, the routine solves it exactly using a greedy strategy, and thus returns the true minimizer with a descent direction. An important aspect of the Frank–Wolfe algorithm is the efficiency of the linear minimization oracle, since it is invoked at every iteration. In our implementation, the linear subproblem is solved by exploiting the structure of the feasible set, which consists of box constraints and a single linear inequality. The routine first computes the minimizer over the box constraints using a simple loop over the variables, and then, if necessary, restores feasibility with respect to the linear constraint by a greedy procedure. This procedure involves sorting the variables according to an efficiency ratio, which requires $O(n \log n)$ operations, while all remaining steps are linear in $n$. Therefore, the overall computational cost of the linear oracle is $O(n \log n)$ per iteration.

## 5 Convergence

In this section, we present some results on convergence. First, we introduce some necessary theorems and definitions:

**Definition 2** (Duality gap)**.** *Let $f : \mathcal{D} \to \mathbb{R}$ be convex and differentiable, and let $\mathcal{D} \subset \mathbb{R}^n$ be a nonempty compact convex set. The* Frank–Wolfe duality gap *is*

$$g(x) \; := \; \max_{s \in \mathcal{D}} \langle \nabla f(x), \, x - s \rangle \; .$$

**Theorem 1** (Upper bound)**.** *Under the assumptions of the previous definition, for every $x \in \mathcal{D}$ and any optimal solution $x^\star$, the Frank–Wolfe duality gap satisfies*

$$f(x) - f(x^\star) \; \leq \; g(x).$$

This, as stated in [1], is a certificate of convergence and so can be used in our algorithm.

**Theorem 2** (Primal convergence, Sublinear convergence)**.** *Let $f$ be convex and differentiable in a $D$ nonempty compact convex set, it holds*

$$f(x) - f(x^\star) \; \leq \; \frac{2C_f}{k}(1 - \delta),$$

*with $\delta$ precision of Linear Problem ($\delta = 0$ in our case since it is exact) and $C_f$ curvature constant [1].*

In addition, if $f$ is L-smooth, we can use the Lemma in [1], which says that $C_f \leq \frac{1}{2}LD^2$ with $D = Diam(D)$, or also the Theorem1 from [2], to obtain a convergence of the type

$$O(\frac{LD^2}{k}).$$

Since we know that the duality gap is a certificate for optimality we can use the primal convergence theorem to imply sublinear convergence of our FW algorithm with the formula written above.

## 6 Local Linear Convergence for standard Frank-Wolfe

There is a very special case in which we can have local linear convergence. Theorem 2 from [2] gives this result:

**Theorem 3** (Local linear convergence)**.** *Suppose assumptions 1 and 2 are satisfied and let the optimal solution $x^*$ be in the relative interior of $D$. Then the sequence $\{x^k\}_k$ generated by FW converges geometrically to $x^*$.*

An important catch is given by the proof, in which we can understand that: if the optimal solution $x^*$ lies in the relative interior of the feasible set $D$, then there exists a neighborhood of $x^*$ that does not intersect the boundary of $D$. Once the iterates $x^k$ enter this region (for $k$ sufficiently large, $k >> K$), the optimization problem becomes locally equivalent to an unconstrained problem. In this local regime, the Frank–Wolfe directions remain uniformly well aligned with the gradient, and the algorithm achieves a linear (geometric) rate of convergence over those iterations. An important note is that this convergence result is stated in terms of the primal objective error $f(x^k) - f(x^*)$.

## 7 Experiments

We wrote a function **generate_test_case()** that generates three main cases for which the algorithm deserves to be analyzed. In such cases, the matrix $Q$ is set to be Symmetric Positive Definite, and the dimension is set to $n = 10$. The precision (i.e. the stopping condition for the duality gap) is set to $10^{-6}$, as we can see from common experiments in famous articles [1] [3]. The initial starting point $x_0$ is chosen at random. Whenever this point does not fall inside the box boundaries, we snap it back to a feasible position. A pseudocode used to generate such tests is shown below.

**Algorithm 2** Common values

```
Require: DIMENSION $n$
1: $l \leftarrow 0_n, \quad u \leftarrow 1_n$
2: $a \leftarrow rand(n,1) + 0.1$
3: $\kappa \leftarrow 10^2, \quad \lambda_{\min} \leftarrow 1, \quad \lambda_{\max} \leftarrow \kappa\lambda_{\min}$
4: $\lambda \leftarrow \text{logspace}(\log_{10}\lambda_{\min}, \log_{10}\lambda_{\max}, n)$
5: $U \leftarrow \text{qr}(randn(n))$
6: $Q \leftarrow U\,\text{diag}(\lambda)\,U^\top$
7: $x_0 \leftarrow rand(n,1)$
8: $\varepsilon \leftarrow 10^{-6}$
```

**Algorithm 3** Interior test

```
1: $x_\star \leftarrow 0.3 + 0.4 \cdot rand(n,1)$
2: $q \leftarrow -2Qx_\star$
3: $b \leftarrow 0.1$
```

**Algorithm 4** Box boundary test

```
1: $x_\star \leftarrow \mathbf{1}_n$
2: $x_\star(1) \leftarrow 0.8$
3: $q \leftarrow -2Qx_\star$
4: $b \leftarrow 0.1$
```

**Algorithm 5** Active linear test

```
1: $x_\star \leftarrow rand(n,1)$
2: $b \leftarrow a^\top x_\star$
3: $q \leftarrow -2Qx_\star$
```

The first notable case is the "interior" one, where the optimum lies inside the box, with the linear constraint inactive. As expected from Theorem 3, the algorithm achieves geometric convergence in the primal error, with about 700 iterations and a precision of $10^{-6}$. This is a very special case. In general, linear convergence is not guaranteed with the basic Frank-Wolfe algorithm, while it is achievable with some variants. The second test, for which the optimum lies on the boundary of the box constraint, converges sublinearly with number of iterations that reaches the maximum $10^5$, with precision of an order of $10^{-6}$. It is less stable with respect to the first one: the behavior of the dual gap in this case is also expected because of the nature of the Linear Problem solved by our routine. The solution $s$ always points to a vertex, and the zigzag pattern becomes less stable if the box constraint is active. In particular, as we get close to the optimum on a face of the box, the directions $d = s - x$ becomes more and more orthogonal to the gradient, resulting in a poor sublinear convergence.

This is also stated in [2], which gives a lower bound on the convergence with the theorem

**Theorem 4.** *Assuming:*

- *$f$ L-smooth and strongly convex,*
- *the (unique) solution $x^\star$ lies on the boundary of $D$,*
- *$x^\star$ is not on a vertex,*
- *$x^k \notin T^\star$ for some index $k$, where $T^\star$ denotes the smallest face of $S$ containing $x^\star$.*

*Then, for any constant $\delta > 0$, the inequality*

$$f(x^k) - f(x^\star) \geq \frac{1}{k^{1+\delta}}$$

*holds for infinitely many indices $k$.*

The third test is done with optimum on the active linear constraint. When the optimal point lies on a face defined by a linear constraint $a^T x = b$, this face is typically oblique with respect to the coordinate axes. Such a face is usually long and does not correspond to a simple box edge. FW operates by repeatedly moving toward a vertex of the polytope. On an oblique face, none of the vertices lie close to the optimal point, and FW cannot move along them efficiently. This happens to be the worst case in our tests, as the number of iterations reaches the maximum and the precision suffers, resulting in an order of $10^{-4} < \epsilon = 10^{-6}$. This is obviously due to the fact that we have sublinear convergence and the gap decreases very slowly. We can easily see it with this algebra: Suppose $n = 10$, so roughly $\epsilon = 1/k$, with $k$ number of iteration. We have that $f(x_k) - f_\star = 1/k = \epsilon$. The stopping condition is $gap_k \leq \epsilon$, but $f(x_k) - f_\star \leq gap_k$ (the gap is un upper bound), so to have at least a precision of $10^{-6}$, we need 1 million iterations. The situation worsens, of course, as we increase the dimension $n$.

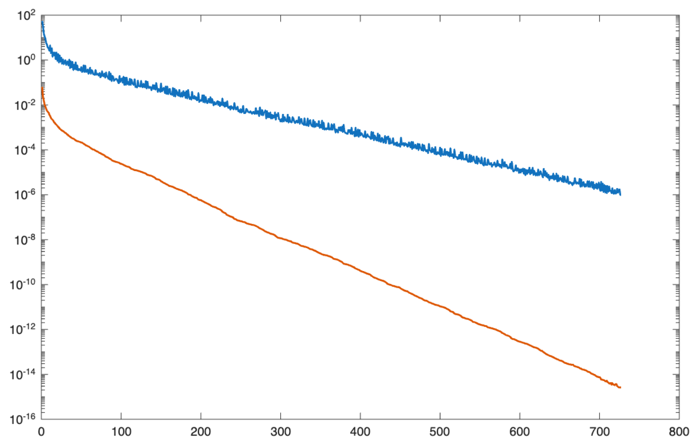

Figure 1: Frank Wolfe with interior optimum

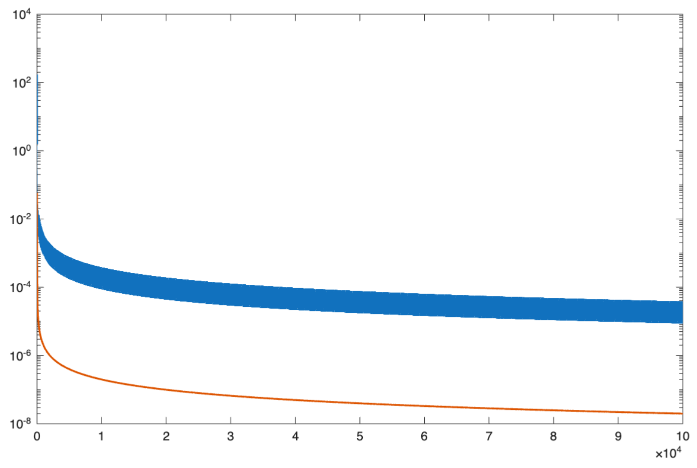

Figure 2: Frank Wolfe with optimum on the boundary of the box

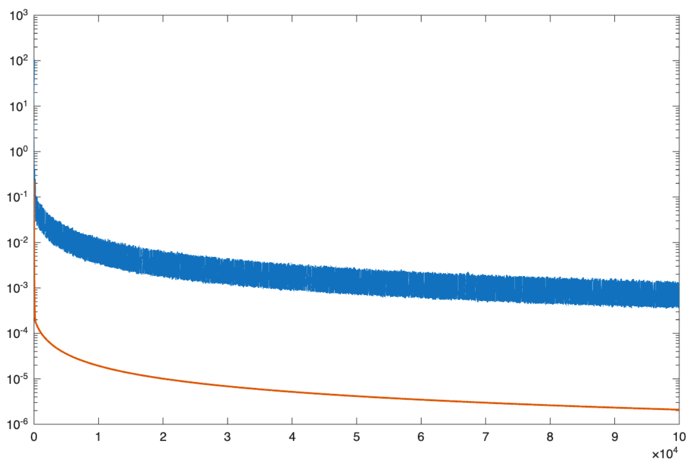

Figure 3: Frank Wolfe with optimum on the linear constraint

## 8 How interior vs box boundary can influence the zigzag patter

Understanding a concept often means visualizing it. We plotted two examples in 2 dimensions to effectively see how an optimum on the box boundary can affect the stability of the algorithm, as we described in the previous Section. We plotted the level set of a simple two-dimensional function. As we can see from the images below, the algorithm performs more efficiently when having a minimum in the box, and the zigzag pattern is less evident.

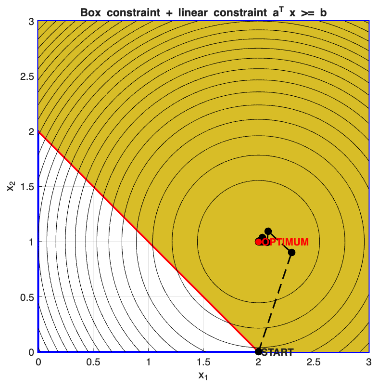

Figure 4: Frank Wolfe with interior optimum 2D

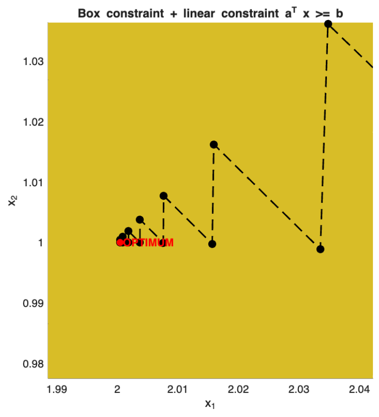

Figure 5: Frank Wolfe zoomed with interior optimum 2D

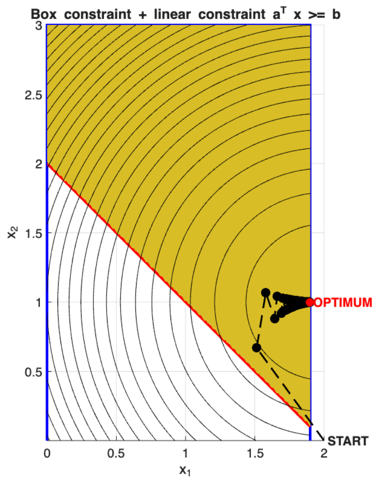

Figure 6: Frank Wolfe with boundary optimum 2D

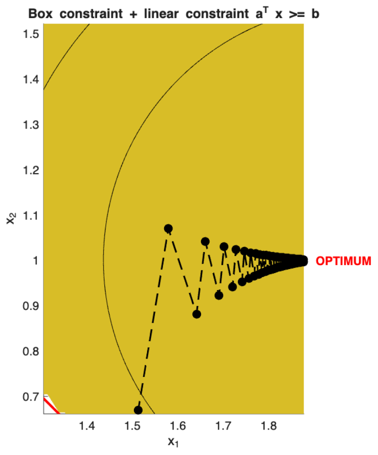

Figure 7: Frank Wolfe zoomed with boundary optimum 2D

## 9 Conditioning tests

Here, we discuss the case of ill-conditioned matrixes with respect to Frank Wolfe. In particular, we considered only the simplest case of minimum on the interior of the box, with linear constraint inactive. The test is generated with this pseudocode, in which we tried discrete values of the condition number ranged from $10^2$ to $10^6$:

**Algorithm 6** Test: conditioning vs iterations (interior)

```
Require: DIMENSION $n$
 1: $l \leftarrow 0_n, \quad u \leftarrow 1_n$
 2: $a \leftarrow rand(n,1) + 0.1$
 3: $\kappa_{\text{list}} \leftarrow [10^2, 10^3, 10^4, 10^5, 10^6]$
 4: $\lambda_{\min} \leftarrow 1, \quad \varepsilon \leftarrow 10^{-3}$
 5: $x_\star \leftarrow 0.3 + 0.4 \cdot (n,1)$
 6: $x_0 \leftarrow rand(n,1)$
 7: $b \leftarrow 0.1$
 8: for $\kappa \in \kappa_{\text{list}}$ do
 9:     $\lambda_{\max} \leftarrow \kappa\, \lambda_{\min}$
10:     $\lambda \leftarrow$ LOGSPACE($\log_{10} \lambda_{\min}, \log_{10} \lambda_{\max}, n$)
11:     $U \leftarrow$ QR($rand(n)$)
12:     $Q \leftarrow U\, \text{DIAG}(\lambda)\, U^\top$ {SPD WITH COND $\approx \kappa$}
13:     $q \leftarrow -2Qx_\star$
14:     $(x_{\text{FW}}, f_{\text{FW}}, f_\star, \text{gaps}) \leftarrow$ FRANK_WOLFE($Q, q, x_0, a, b, l, u, \varepsilon$)
15:     iters($\kappa$) $\leftarrow$ LENGTH(gaps)
16:     final_gap($\kappa$) $\leftarrow$ gaps(end)
17: end for
```

As we increase the condition number, the iterations will grow. This actually depends only on one eigenvalue, in particular $\lambda_{max}$. This is because, in the quadratic case and for Proposition 2, we have that $L = \lambda_{max}(Q)$. We also explain another point of view using this proposition:

**Proposition 4.** *Let $f$ be a L-smooth convex quadratic function, for any direction $d$ it holds*

$$d^T Q d \leq \lambda_{max} ||d||^2$$

*Proof.* Since $f$ is quadratic, its second-order Taylor expansion is exact:

$$f(x+d) = f(x) + \langle \nabla f(x), d \rangle + d^\top Q d.$$

If $f$ is $L$-smooth, then the standard smoothness inequality yields

$$f(x+d) \leq f(x) + \langle \nabla f(x), d \rangle + L|d\|^2.$$

Comparing the two expressions and canceling common terms gives

$$d^\top Q d \leq L\|d\|^2.$$

□

Using the result above, we can also understand why larger values of $\lambda_{\max}$ may force the algorithm to take smaller steps. Indeed, the term $d^\top Q d$ appears in the denominator of the exact line-search formula, and since $d^\top Q d \leq \lambda_{\max}\|d\|^2$, a large $\lambda_{\max}$ can make the optimal step size $\alpha^\star$ arbitrarily small (provided that, in some cases, we have some components along the direction of maximum eigenvalue). In the following, Figure 9 shows the number of iterations versus the conditioning number.

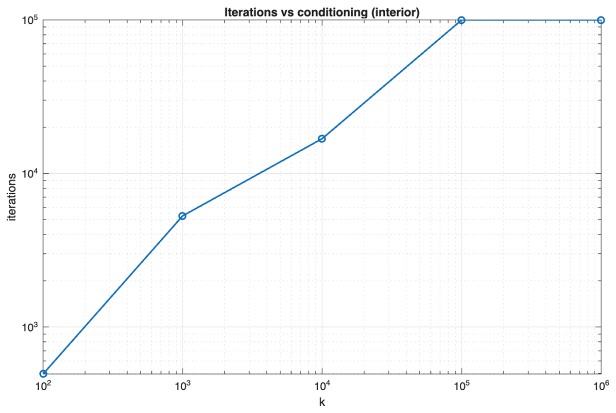

Figure 8: Iterations vs conditioning number

## References

[1] M. Jaggi, "Revisiting frank–wolfe: Projection-free sparse convex optimization," *Journal of Machine Learning Research*, vol. 14, pp. 427–486, 2013.

[2] J. Guelat and P. Marcotte, "Some comments on wolfe's method for convex programming," *Mathematical Programming*, vol. 35, no. 1, pp. 110–119, 1986.

[3] M. Jaggi, "An overview of frank–wolfe optimization," *Foundations and Trends in Machine Learning*, vol. 8, no. 3–4, pp. 231–364, 2015. arXiv:1511.05932.