Linear Convergence of the Frank-Wolfe Algorithm over Product Polytopes

Gabriele Iommazzo
Zuse Institute Berlin

David Martínez-Rubio
IMDEA Software Institute

Francisco Criado
CUNEF Universidad

Elias Wirth
Berlin Institute of Technology

Sebastian Pokutta
TU Berlin and Zuse Institute Berlin

Abstract

We study the linear convergence of Frank-Wolfe algorithms over product polytopes. We analyze two condition numbers for the product polytope, namely the *pyramidal width* and the *vertex-facet distance*, based on the condition numbers of individual polytope components. As a result, for convex objectives that are μ-Polyak-Łojasiewicz, we show linear convergence rates quantified in terms of the resulting condition numbers. We apply our results to the problem of approximately finding a feasible point in a polytope intersection in high-dimensions, and demonstrate the practical efficiency of our algorithms through empirical results.

1 INTRODUCTION

Frank-Wolfe algorithms (FW), also known as Conditional Gradient methods, are widely used first-order optimization algorithms. They are particularly suited for constrained convex problems $\min_{x\in\mathcal{X}} f(x)$, for a convex compact set $\mathcal{X}$, where linear minimization over $\mathcal{X}$ is used to keep the iterates feasible. The simplicity of this family of methods, along with its projection-free nature, make it especially appealing in large-scale settings, where projections can be costly. This is often the case when the feasible set is a polytope (Braun, Carderera, et al., 2022).

More concretely, FW methods compute iterates as combinations of extreme points in the feasible set $\mathcal{X}$ returned by a linear minimization oracle (LMO):

$$g \mapsto \arg\min_{x\in\mathcal{X}} \langle g, x\rangle \tag{LMO}$$

on $\mathcal{X}$, where g is some vector, usually the current gradient of the objective function or an approximation thereof. After computing (LMO), the standard FW method takes a step from the current iterate in the direction of the LMO solution. In many scenarios, (LMO) is a cheaper alternative to the quadratic optimization problem that projections consist of, cf. (Jaggi, 2013), and can be computed even when $\mathcal{X}$ is defined implicitly, such as a convex hull of a set of points.

The convergence behavior of FW methods is strongly influenced by the geometry of the constraint set, that of the function, and the location of the unconstrained minimizer. While sublinear rates are typical in general convex and smooth settings, it is known that linear convergence can be achieved under additional assumptions, such as global optimizers being in the interior of $\mathcal{X}$ (Wolfe, 1970; Guélat and Marcotte, 1986a), strong convexity of the feasible set along with all global optimizers being outside of it (Levitin and Polyak, 1966), or strong convexity of f for $\mathcal{X}$ being a polytope having some positive condition number. This last assumption holds for one of several different notions of conditioning that have been defined (Lacoste-Julien and Jaggi, 2013; Garber and Hazan, 2015a; Lacoste-Julien and Jaggi, 2015; Beck and Shtern, 2015; Peña, Rodríguez, and Soheili, 2016; Peña and Rodríguez, 2018), among which there is the pyramidal width and the vertex-facet distance that we use in this paper, cf. Section 4.

A few works extend the applicability of the algorithms to larger classes of functions such as the ones satisfying the Polyak-Łojasiewicz (PL) condition, as opposed to strong convexity (Beck and Shtern, 2015; Braun, Carderera, et al., 2022). We quantify the implications

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

of our condition number results in terms of linear convergence rates for our applications.

Importantly, we note that much of the aforementioned theory remains confined to the setting of single polytopes, with limited understanding of how favorable properties like good condition numbers might extend and compose in more structured domains. In this work, we focus on optimization over *product polytopes*.

This setting arises naturally in machine learning applications, specially via the feasibility problem for the intersection of several feasible sets, such as in image representation in computerized tomography (Censor et al., 2012), compressed sensing (Carmi, Censor, and Gurfil, 2012), and including signal processing and statistical estimation (Stark, Yang, and Yang, 1998; Combettes and Pesquet, 2011), image recovery (Combettes, 1996), absolute value equations (Alcantara, Chen, and Tam, 2023), matrix completion (Richard, Savalle, and Vayatis, 2012) and robust PCA (Ma and Aybat, 2018), SVMs (Lacoste-Julien, Jaggi, et al., 2013) and clustering (Bomze, Rinaldi, and Zeffiro, 2024).

A natural question is whether the good conditioning of individual polytopes can be lifted to the product domain in a way that still allows for linear convergence guarantees. We provide a surprising positive answer as the main contribution of this work: *the product of well-conditioned polytopes inherits a form of good conditioning*, that enables linear convergence of FW algorithms for smooth, convex, PL-objectives by using either the pyramidal width or the vertex-facet distance.

Positive polytope condition numbers, such as the pyramidal width or the vertex-facet distance, typically capture how well the directions found by the LMO in the polytope align with the idealized direction $x^* - x_t$ (from an iterate x_t towards a minimizer x^*) relative to the direction of the gradient that was used for the LMO, cf. Section A.4. Our findings imply that this alignment does not degrade much when considering a product polytope, with respect to the individual components.

We apply our results to the aforementioned important primitive in large-scale optimization: approximately finding a feasible point in the intersection of several polytopes or certifying that such intersection is empty. In other words, the problem of interest is, given $k \geq 2$ polytopes $\mathcal{P}_i$,

$$\text{find } x \in \bigcap_{i=1}^{k} \mathcal{P}_i \,, \tag{P}$$

if it exists, or determine that the intersection is empty. An *ε-approximate* solution of (P) consists of finding a point whose Euclidean distance to the intersection is less than ε, or determining that no such point exists. Finding an approximate feasible point in the intersection of k polytopes can be turned into an approximate solver for linear programming over such intersection as feasible set, by performing a binary search objective value, see, e.g., (Dadush, Végh, and Zambelli, 2022) and references therein.

For the above task, we consider the Away Frank-Wolfe (AFW) algorithm (Guélat and Marcotte, 1986b) (see Algorithm 1 in Section B) over the Cartesian product, where we have a variable for each polytope and minimize an objective that penalizes each of the pairwise distances between the k variables, cf. (4). Our objective is convex, smooth and satisfies the PL inequality. Naturally, this approach uses the LMO of the resulting product polytope, that is, the product of the corresponding LMO outputs for each polytope component, which can be computed in parallel.

Our technical contributions are the following.

Contributions

- Regarding polytope condition numbers, we show that, if the *pyramidal widths* of two polytopes $\mathcal{P}_1$ and $\mathcal{P}_2$ are $\delta_{\mathcal{P}_1}$ and $\delta_{\mathcal{P}_2}$, that of the product is exactly $\delta_{\mathcal{P}_1}\delta_{\mathcal{P}_2}/\sqrt{\delta_{\mathcal{P}_1}^2 + \delta_{\mathcal{P}_2}^2}$. This yields a pyramidal width $\delta_{\mathcal{P}} = \Omega(\frac{1}{\sqrt{k}} \min_{i\in[k]} \delta_{\mathcal{P}_i})$ for the product $\mathcal{P} = \prod_{i\in[k]} \mathcal{P}_i$ of k polytopes.

  Our proof uses the *affine pyramidal width*, which we introduce as a new equivalent characterization of the pyramidal width that is easier to work with in our proofs. Similarly, for the *vertex-facet distance*, we show $\mathsf{vf}_{\mathcal{P}} = \min_{i\in[k]} \mathsf{vf}_{\mathcal{P}_i}$.

- As a consequence of studying these two polytope condition numbers, we can quantify linear rates of convergence for the AFW algorithm optimizing convex, smooth functions satisfying the PL inequality over a product polytope, which inherits its conditioning from the components.

- We apply our results to the problem of approximately finding a feasible point in the intersection of several polytopes, by minimizing (4), a convex, smooth and PL objective.

- We substantiate our theoretical findings with empirical evidence, demonstrating the algorithm's applicability across large-scale optimization tasks.

The novel characterization of the pyramidal width as defined by the affine pyramidal width is of independent interest, and we believe that it can offer new insights into the optimization over polyhedral sets.

2 OTHER RELATED WORK

Polytope Condition Numbers A line of research has established nontrivial lower bounds for polytope

condition numbers in key domains (Wirth, Pokutta, et al., 2023; Chakrabarti, Farina, and Kroer, 2024). Recent works have further refined affine-invariant convergence analyses by exploiting facial distance to obtain improved Frank–Wolfe rates (Wirth, Pena, and Pokutta, 2025; Pena, 2023).

Convex Feasibility via Projections. For the convex feasibility problem, classical methods make use of projections as opposed to linear minimization oracles, including von Neumann's alternating projections and its extensions (Neumann, 1949; Ginat, 2018), and solve the intersection problem by repeatedly projecting onto each set. In general these approaches achieve only sublinear convergence (Badea and Seifert, 2016; Bauschke and Combettes, 2011; Braun, Pokutta, and Weismantel, 2022), though linear rates can be recovered under additional regularity or transversality assumptions on the intersection (Gubin, Polyak, and Raik, 1967; Bauschke and Borwein, 1993; Bauschke and Borwein, 1996; Beck and Teboulle, 2003; Drusvyatskiy, Ioffe, and Lewis, 2015; Bui, Loxton, and Moeini, 2021; Behling, Bello-Cruz, and Santos, 2021).

Frank–Wolfe for Feasibility and Block-coordinate Variants. Several works applied FW-type methods to the convex feasibility problem. Early distance-minimization schemes by (Willner, 1968; Wolfe, 1976) use FW methods without providing convergence rates. More recent block-coordinate FW algorithms (Beck, Pauwels, and Sabach, 2015; Bomze, Rinaldi, and Zeffiro, 2024) exploit product-structure by cycling or randomly selecting blocks, but without global linear rates unless combined with costly subroutines. An augmented-Lagrangian FW approach under a PL condition was proposed in (Gidel, Pedregosa, and Lacoste-Julien, 2018) for two blocks, though it requires stringent feasibility assumptions and does not easily extend to larger k. Garber, Livney, and Sabach (2023, Appendix A) point out that the proof in (Gidel, Pedregosa, and Lacoste-Julien, 2018) is incomplete. A full corrected analysis appears in (Silveti-Falls, Molinari, and Fadili, 2020, Section 6.2).

FW Convergence and Fast Convergence Rates. At the core of Frank–Wolfe research is the study of fast (Garber and Hazan, 2015b; Kerdreux, d'Aspremont, and Pokutta, 2020; Wirth, Kerdreux, and Pokutta, 2023; Wirth, Pena, and Pokutta, 2025) and affine-invariant linear rates (Lacoste-Julien and Jaggi, 2015; Tsuji, Tanaka, and Pokutta, 2022; Pena, 2023; Wirth, Pena, and Pokutta, 2024). To derive linear rates, three key premises (Peña and Rodríguez, 2018) appear throughout the literature.

The first premise involves a positive *condition number of the polytope*. The condition number is used to establish an inequality comparing (A) the alignment of the gradient with the direction taken by the algorithm, obtained by the LMO with the gradient as direction, and (B) the alignment of the gradient with the idealized direction $x_t - x^*$, cf. Theorem A.4.

The second premise is an *objective-function growth* condition from the minimizers. The standard assumption in this case is strong convexity (Guélat and Marcotte, 1986b; Lacoste-Julien and Jaggi, 2015; Garber and Hazan, 2015a; Tsuji, Tanaka, and Pokutta, 2022), although weaker conditions have been successfully applied (see, e.g., (Beck and Shtern, 2015; Lacoste-Julien and Jaggi, 2015; Wirth, Pena, and Pokutta, 2024)).

The third premise is a condition that guarantees we can obtain descent on the objective, such as smoothness. Our work also makes sure of these three premises in order to achieve linear convergence rates.

3 NOTATION AND PRELIMINARIES

We denote the set of vertices of an n-dimensional (bounded) polytope $\mathcal{P}$ by $\mathcal{V}_\mathcal{P}$, the convex hull and affine hull of its vertices by $\mathsf{conv}(\mathcal{P})$ and $\mathsf{aff}(\mathcal{P})$, the set of its proper faces by $\mathsf{faces}^*(\mathcal{P})$, and the set of its facets by $\mathsf{facets}(\mathcal{P})$. A face is *proper* if it is neither $\mathcal{P}$ itself nor the empty set. Facets are proper "maximal" faces, in the sense that they are not contained in any other face, so they have dimension $n-1$.

Throughout, we let $\|\cdot\|$ be the Euclidean norm and $\mathsf{dist}(\mathcal{P},\mathcal{Q}) \stackrel{\text{def}}{=} \min_{x\in\mathcal{P},y\in\mathcal{Q}} \|x-y\|$ for nonempty polytopes $\mathcal{P},\mathcal{Q} \subseteq \mathbb{R}^n$. Further, we use the shorthand $D_\mathcal{P} \stackrel{\text{def}}{=} \max_{v,w\in\mathcal{P}} \|v-w\|$ to denote the *diameter* of a polytope $\mathcal{P}$. We also denote the Kronecker product of two matrices A, B by $A \otimes B$.

We address the problem $\min_{x\in\mathcal{P}} f(x)$ of minimizing a convex, smooth and PL function over a polytope via FW methods. We denote its solution set by $\Omega^* \stackrel{\text{def}}{=} \arg\min_{x\in\mathcal{X}} f(x)$ and an arbitrary point in Ω^* by x^*.

Further, let $[k] \stackrel{\text{def}}{=} \{1,2,\dots,k\}$. We denote the $m\times m$ identity matrix by I_m and the m-dimensional vector of ones by $\mathbb{1}_m$. We make use of the Cartesian product of k n-dimensional polytopes $\Pi_{i\in[k]}\mathcal{P}_i \stackrel{\text{def}}{=} \mathcal{P}_1 \times \cdots \times \mathcal{P}_k \subset \mathbb{R}^{n\times k}$ denoting its components by upper indices $x = (x^1,\dots,x^k)$, where each $x^i \in \mathcal{P}_i$ is said to be a *block variable*. In the following, we will consider blocks $I_t = \{i\}$, for some $i \in [k]$, and the partial gradients $\nabla^i f(x) \stackrel{\text{def}}{=} (\nabla f(x))^i$ of a differentiable function f over block $i \in [k]$. A face is a product of faces in this setting.

We will work with the following function regularity conditions, where the differentiability assumption over

non-open sets $\mathcal{X}$ is to be understood as differentiability in an open set containing $\mathcal{X}$.

Definition 3.1 (**Convexity and L-smoothness**). *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a convex set and $f : \mathbb{R}^n \to \mathbb{R}$ differentiable in $\mathcal{X}$. The function f is convex (resp. L-smooth) in $\mathcal{X}$ if ① holds for all $x, y \in \mathcal{X}$ (resp. ②):*

$$0 \overset{①}{\leq} f(x) - f(y) - \langle \nabla f(y), x - y \rangle \overset{②}{\leq} \frac{L}{2} \|x - y\|^2 .$$

Under convexity, L-smoothness is equivalent to $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$, cf. (Beck, 2017).

Definition 3.2 (**Polyak-Łojasiewicz (PL) condition**). *Let $\mathcal{X} \subseteq \mathbb{R}^n$ be a set over which we minimize a function $f : \mathbb{R}^n \to \mathbb{R}$, differentiable in $\mathcal{X}$. Assume the solution set Ω^* exists and is nonempty, and let the minimum function value be $f^* \overset{\text{def}}{=} f(x^*)$, for $x^* \in \Omega^*$. We say that f is μ-PL in $\mathcal{X}$ if, for all $x \in \mathcal{X}$,*

$$\frac{1}{2}\|\nabla f(x)\|^2 \geq \mu(f(x) - f^*).$$

A well-known fact is that, under convexity, μ-PL is equivalent to the so-called μ-quadratic growth condition (QG), which is $f(x) - f^* \geq \frac{\mu}{2} \min_{x^* \in \Omega^*} \|x - x^*\|^2$ for every $x \in \mathbb{R}^n$, cf. (Karimi, Nutini, and Schmidt, 2016).

4 CONDITIONING OF PRODUCT POLYTOPES AND CONVERGENCE

In this section, we develop our theory providing bounds for how our polytope condition numbers of interest behave when considering product polytopes. At the end of this section we quantify the linear rates of convergence that ensue from our results.

We start by studying the pyramidal width and show that, if each subpolytope $\mathcal{P}_i$ is well-conditioned in terms of pyramidal width lower bounds, then the product polytope $\mathcal{P}$ is also well-conditioned. Note that several equivalent definitions of the pyramidal width can be found in the literature. We make use of the one given by (Peña and Rodríguez, 2018), where it is termed *facial distance*, while the original notions can be found in (Lacoste-Julien and Jaggi, 2013). See also (Beck and Shtern, 2015; Lacoste-Julien and Jaggi, 2015).

For a polytope $\mathcal{P}$ and one of its faces f, we write $P^f \overset{\text{def}}{=} \mathsf{conv}\,(\mathcal{V}_\mathcal{P} \setminus \mathcal{V}_f)$.

Definition 4.1 (**Pyramidal width (Peña and Rodríguez, 2018)**). *Let $\mathcal{P}$ be a polytope, with at least two vertices and let f be a face of $\mathcal{P}$. We define the* pyramidal width in f *$\Delta_f(\mathcal{P})$ of $\mathcal{P}$ as the distance between f and $\mathcal{P}^f$. The* pyramidal width *$\delta_\mathcal{P}$ of $\mathcal{P}$ is the minimum of $\Delta_f(\mathcal{P})$ over all faces f of $\mathcal{P}$. That is,*

$$\delta_\mathcal{P} \overset{\text{def}}{=} \min_{f \in \mathsf{faces}^*(\mathcal{P})} \mathsf{dist}\,(f, \mathsf{conv}\,(\mathcal{V}_\mathcal{P} \setminus f)) . \tag{1}$$

In order to achieve linear rates with FW algorithms, it is enough to obtain lower bounds for the pyramidal width $\delta_\mathcal{P}$ of the domain $\mathcal{P}$. For a product polytope, $\mathcal{P} = \prod_{i \in [k]} \mathcal{P}_i$, we trivially have that $\delta_\mathcal{P} \leq \min_{i \in [k]} \{\delta_{\mathcal{P}_i}\}$ from the definition. In the sequel, we will show a lower bound that is only a minor polynomial factor away from this upper bound, depending on k. To that end, we introduce the following variant of the pyramidal width, and show it is equivalent to the definition above.

Definition 4.2 (**Affine pyramidal width**). *Let $\mathcal{P}$ be a polytope, and let f be a face of $\mathcal{P}$. The* affine pyramidal width *of $\mathcal{P}$ in f is the distance between $\mathsf{aff}\,(f)$ and $\mathcal{P}^f$. We denote it by $\bar{\Delta}_f(\mathcal{P})$. The* affine pyramidal width *$\bar{\delta}_\mathcal{P}$ of $\mathcal{P}$ is the minimum of $\bar{\Delta}_f(\mathcal{P})$ over all faces f of $\mathcal{P}$. That is,*

$$\bar{\delta}_\mathcal{P} \overset{\text{def}}{=} \min_{f \in \mathsf{faces}^*(\mathcal{P})} \mathsf{dist}\,(\mathsf{aff}\,(f), \mathsf{conv}\,(\mathcal{V}_\mathcal{P} \setminus f)) . \tag{2}$$

Observe that, in these two definitions, one never considers the trivial faces $\emptyset$ and $\mathcal{P}$. Allowing either would yield an infinite pyramidal width under the convention that the distance between a set and the empty set is infinity. Since $\mathcal{P}$ is at least 1-dimensional, a nontrivial face exists so the pyramidal width is finite.

We note that $\bar{\Delta}_f(\mathcal{P})$ is not always equal to $\Delta_f(\mathcal{P})$. Our claim only applies to $\delta_\mathcal{P} = \bar{\delta}_\mathcal{P}$, that is, to the minimum over all faces for each respective quantity.

The next Lemma 4.3 states a key structural property of a polytope, i.e., that the point realizing the affine pyramidal width lies in the relative interior of the corresponding face. This immediately implies that the affine variant is equivalent to the classical pyramidal width, as stated by Corollary 4.4. However, The affine pyramidal width is also better suited for our compositional arguments in product polytopes: in fact, this equivalence allows us to work with the affine pyramidal width in the proof of Theorem 4.5 while still obtaining the classical pyramidal width for the product.

Lemma 4.3. [↓] *Let $\mathcal{P}$ be a full-dimensional polytope in $\mathbb{R}^n$ for $n \geq 1$, and let f be a face of $\mathcal{P}$ that achieves the minimum affine pyramidal width. Let p be a point of $\mathsf{aff}\,(f)$ that realizes the minimum distance. Then, p is in the relative interior of f.*

Corollary 4.4 (**Widths equivalence**). *Let $\mathcal{P}$ be a polytope. Then,*

$$\delta_{\mathcal{P}} = \bar{\delta}_{\mathcal{P}}\,.$$

Proof For every face f of $\mathcal{P}$, we directly have $\bar{\Delta}_f(\mathcal{P}) \leq \Delta_f(\mathcal{P})$ from the definition,so $\bar{\delta}_{\mathcal{P}} \leq \delta_{\mathcal{P}}$. Now, let f be a face of $\mathcal{P}$ with minimum affine pyramidal width. Then, by Lemma 4.3, $\Delta_f(\mathcal{P}) = \bar{\Delta}_f(\mathcal{P})$. Hence, $\delta_{\mathcal{P}} = \bar{\delta}_{\mathcal{P}}$.

■

The pyramidal width $\delta_{\mathcal{P},\mathcal{Q}}$ of the Cartesian product $\mathcal{P} \times \mathcal{Q}$ of two polytopes is characterized as follows. In the proof, we find specific faces that realize this distance and prove its optimality, yielding an equality.

Theorem 4.5 (**Pyramidal width of the product**). [↓] *Let $\delta_{\mathcal{P}}$ and $\delta_{\mathcal{Q}}$ be the pyramidal widths of polytopes $\mathcal{P}, \mathcal{Q} \subseteq \mathbb{R}^n$. Then,*

$$\delta_{\mathcal{P}\times\mathcal{Q}} = \sqrt{\frac{\delta_{\mathcal{P}}^2\,\delta_{\mathcal{Q}}^2}{\delta_{\mathcal{P}}^2 + \delta_{\mathcal{Q}}^2}}\,. \tag{3}$$

Now that Theorem 4.5 directly implies a bound $\delta_{\mathcal{P}} = \Omega(\frac{1}{\sqrt{k}} \min_{i\in[k]} \delta_{\mathcal{P}_i})$ for the pyramidal width of the product polytope $\mathcal{P} = \prod_{i\in[k]} \mathcal{P}_i$ if $\delta_{\mathcal{P}_i}$ is the pyramidal width of $\mathcal{P}_i$, see Corollary A.3. Compare this lower bound with the trivial upper bound $\delta_{\mathcal{P}} \leq \min_{i\in[k]}\{\delta_{\mathcal{P}i}\}$: the upper bound is essentially tight when one pyramidal width is much smaller than the others.

We also derive rates that rely on the vertex-facet distance, which is defined as in the following, where now the vertices of the polytope play a crucial role.

Definition 4.6 (**Vertex-facet distance (Peña and Rodríguez, 2018)**). *Let $\mathcal{P}$ be a nonempty polytope with at least two vertices. We define the vertex-facet distance of $\mathcal{P}$ as*

$$\mathsf{vf}_{\mathcal{P}} \stackrel{\text{def}}{=} \min_{F_{\mathcal{P}} \in \mathsf{facets}(\mathcal{P})} \mathsf{dist}\left(\mathsf{aff}\left(F_{\mathcal{P}}\right), \mathcal{V}_{\mathcal{P}} \setminus F_{\mathcal{P}}\right),$$

where $\mathcal{V}_{\mathcal{P}}$ is the set of vertices of $\mathcal{P}$.

Since this polytope condition number only depends on distances between facets and vertices, it behaves better on product polytopes. Notably, unlike the pyramidal width, it does not suffer a polynomial-factor decrease relative to the minimum component condition number.

Proposition 4.7 (**Vertex-facet distance of the product**). [↓] *Let $\mathcal{P} \stackrel{\text{def}}{=} \mathcal{P}_1 \times \cdots \times \mathcal{P}_k$ be a product polytope. We have $\mathsf{vf}_{\mathcal{P}} = \min_{i\in[k]}\{\mathsf{vf}_{\mathcal{P}_i}\}$.*

An interesting fact, proven in (Rademacher and Shu, 2022, Theorem 2.22), is that for a polytope $\mathcal{P}$ with at least two vertices, the following holds: $\mathsf{vf}_{\mathcal{P}} \geq \delta_{\mathcal{P}}$.

4.1 Linear Convergence

In this section, we examine the convergence behavior of the AFW algorithm, when it is run to minimize a PL objective over a product polytope. The following quantifies the linear rates of convergence that are obtained depending on the pyramidal widths or the vertex-facet distances of the individual polytopes in the product.

Proposition 4.8 (**Linear rates via the product's pyramidal width**). [↓] *For $i \in [k]$, let $\mathcal{P}_i$ be a polytope with pyramidal width $\delta_{\mathcal{P}_i}$. The AFW algorithm obtains an ε-minimizer of a convex, L-smooth and μ-PL objective over $\Pi_{i\in[k]}\mathcal{P}_i$ in at most*

$$t = O\left(\frac{LD_{\mathcal{P}}^2}{\mu\alpha_\delta^2} \log\left(\frac{LD_{\mathcal{P}}^2}{\varepsilon}\right)\right)$$

iterations, where $D_{\mathcal{P}}^2$ is the diameter of $\Pi_{i\in[k]}\mathcal{P}_i$, and $\alpha_\delta \stackrel{\text{def}}{=} \frac{1}{\sqrt{2^{\lceil \log_2(k+1)\rceil}}} \min_{i\in[k]}\{\delta_{\mathcal{P}_i}\}$.

We quantify linear convergence rates by using the composition of vertex-facet distances in the product by making use of the analysis in (Beck and Shtern, 2015). The authors did not state their results for PL functions, but their proofs work in this case.

Proposition 4.9 (**Linear rates via the product's vertex-facet distance**). [↓] *For $i \in [k]$, let $\mathcal{P}_i$ be a polytope with vertex-facet distance $\mathsf{vf}_{\mathcal{P}_i}$. The AFW algorithm obtains an ε-minimizer of a convex, L-smooth, μ-PL objective $f(x)$ over $\Pi_{i\in[k]}\mathcal{P}_i$ in at most*

$$t = O\left(\min\left\{\frac{32LD_{\mathcal{P}}^2N^2}{\mu\alpha_{\mathsf{vf}}^2}, 4\right\} \log\left(\frac{f - f(x^*)}{\varepsilon}\right)\right)$$

iterations where N is a constant associated with a procedure performed at the end of each iteration of the AFW algorithm, described in Section B.

5 THE APPROXIMATE FEASIBILITY PROBLEM FOR A POLYTOPE INTERSECTION

In this section, we present the objective that we can optimize in order to approximately solve the feasibility problem (P) for polytopes. We have a variable for each polytope component and we penalize the variables being far from each other:

$$f(x) \stackrel{\text{def}}{=} \frac{1}{2k} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \|x^i - x^j\|^2 \text{ for } x \in \mathcal{P}, \tag{4}$$

where $\mathcal{P} \overset{\text{def}}{=} \Pi_{i\in[k]}\mathcal{P}_i$, and each $\mathcal{P}_i$ is a polytope of dimension n so $x = (x^1, x^2, \dots, x^k) \in \mathbb{R}^{nk}$. Note that $f(x) = \frac{1}{2k}\langle M_k x, x\rangle$, with $M_k \overset{\text{def}}{=} (kI_k - \mathbb{1}_k\mathbb{1}_k^T) \otimes I_n$.

Now we establish several properties of the objective function of Eq. (4) that show that we can apply the theory in Section 4 to determine linear convergence rates for our application.

Proposition 5.1 (Convexity, smoothness, and PL of (4)). [↓] *For $i \in [k]$, let $\mathcal{P}_i \subseteq \mathbb{R}^n$ be convex sets and $\mathcal{P} \overset{\text{def}}{=} \Pi_{i\in[k]}\mathcal{P}_i$. The function $f(x) \overset{\text{def}}{=} \frac{1}{2k}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\|x^i - x^j\|^2$ in (4) is convex, 1-smooth, and 1-PL over $\mathcal{P}$.*

We note that, due to the 1-PL property and convexity, our objective also satisfies the 1-quadratic growth condition. Thus, if the intersection of polytopes is non-empty, the optimum value for the objective in (4) is 0, and reaching an $\frac{\varepsilon}{2k}$ minimizer of f implies that each of our k variables is an ε-approximate feasible point. Note that this factor appears in a logarithmic factor in the convergence rates. On the other hand, if the intersection is empty, we can run the algorithm until we observe that the function value is larger than a primal-dual gap that is computed by the algorithm and is decreasing with linear rates, which guarantees that the optimal value is strictly positive, cf. Section B.

6 EXPERIMENTS

In this section, we evaluate the practical performance of several algorithms for solving the problem in Eq. (4) with different product polytopes.

Each instance is constructed by generating k polytopes. Here we report results for settings with $k = 2, 10$ polytopes in $\mathbb{R}^n$, with up to $n = 20\,000$. We present more plots, corresponding to additional setups in Section C. We generate two kinds of problems: one where the polytopes do not intersect (cf. Figures 1 and 2) and another where they do (cf. Figures 3 and 4). Moreover, we test two polytope setups, that we call "point-cloud" and "vertex-facet" throughout. In the point-cloud setup (cf. Figures 1, 2 and 4), each polytope is generated as the convex hull of a list of points that are sampled uniformly at random from mutually disjoint intervals for every coordinate $i \in [n]$. For each polytope, we draw the number of points uniformly at random in $[n, 2n]$. This choice keeps each polytope "reasonably challenging" to optimize over. In Section C, we also report an example with two disjoint polytopes in $\mathbb{R}^{10\,000}$ built from 100 000 vertices, an order of magnitude larger: the convergence behavior of the tested algorithms in that instance is essentially unchanged, and only the absolute time to convergence increases, reflecting a higher cost for the LMO. In the vertex-facet setup (cf. Figure 3), we only consider the case $k = 2$, and the polytopes are generalizations of an ℓ_∞-ball and of an ℓ_1-ball, respectively. Then, we shift the aforementioned polytopes so they have non-empty intersection. Concretely, in the point-cloud setup we shift $\mathcal{P}_2$ until one of its vertices coincides with a vertex v of $\mathcal{P}_1$; we then move $\mathcal{P}_2$ towards the barycentre of $\mathcal{P}_1$ (the average of its vertices); finally, we translate all remaining polytopes so they share v. Instead, in vertex-facet intersections only a single vertex of the generalized ℓ_1-ball touches the interior of a facet of the generalized ℓ_∞-ball.

Our experiments compare the FW and AFW algorithms, cf. Section B, with different heuristic variants. We implement two types of algorithms: "full" (F-FW and F-AFW), that at each iteration computes a single LMO over the entire product polytope $\mathcal{P}$ and maintains a single *active set* in the product - i.e., a set of previously computed extreme points stored such that the current iterate can be written as a convex combination of such points - and "cyclic block-coordinate" (C-BC), where at each iteration we only compute the LMO of one component and update the corresponding variable and active set only. At each iteration t, F-AFW optimizes over the active set to find an away vertex, and updates the set, see Algorithm 1. An away vertex a_t allows the algorithm to choose to move in the direction guaranteeing more progress, namely, taking an *away step* from x_t in the direction $x_t - a_t$, or taking a *FW step* from x_t towards the LMO solution.

All algorithms were run with a standard line search, see Section B. For each experimental setup, we ran every FW variant until the FW gap dropped below 10^{-7} or until a setup-specific iteration budget was reached, whichever occurred first.

In the plots, we report the following metrics to quantify algorithmic performance over iterations: the primal gap $f(x_t) - \min_{x\in\mathcal{P}} f(x)$, and also the so-called FW gap $\max_{x\in\mathcal{P}}\langle\nabla f(x_t), x_t - x\rangle$ (Braun, Carderera, et al., 2022). While the former cannot be calculated in general, since one does not usually know the minimum objective value a priori, the latter is a standard measure for FW algorithms, since it is an upper bound on the former while at the same time being computable. Therefore, the FW gap results in a stopping criterion that can be used to guarantee ε-optimality has been reached. We thus used it as a stopping criterion in our experiments, if they already converged, cf. (Jaggi, 2013). Both metrics in the figures below are plotted against the elapsed wall-clock time, in seconds, measured within the corresponding FW routine, using logarithmic scales on both axes.

Note that the FW gap is not guaranteed to decrease monotonically. Furthermore, we remark that small

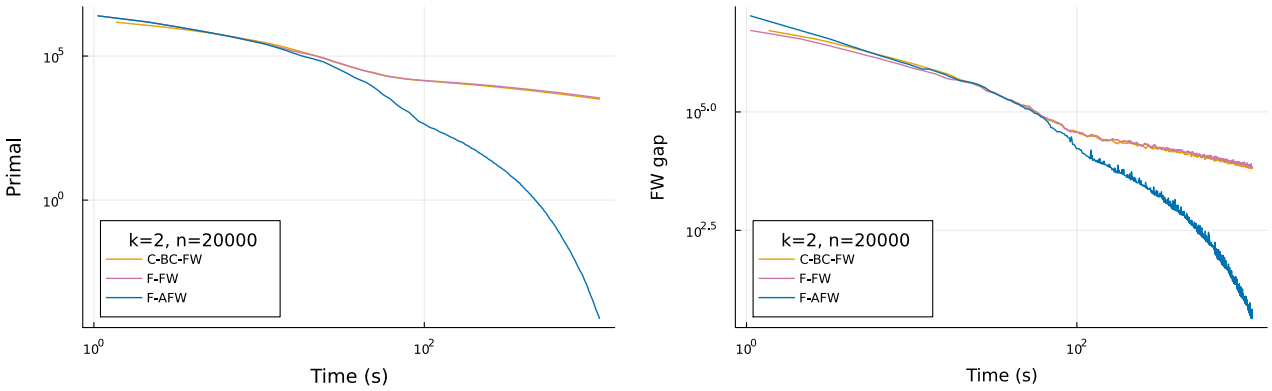

Figure 1: FW variants, run for up to 1 000 iterations on two non-intersecting point-cloud polytopes in $\mathbb{R}^{20\,000}$.

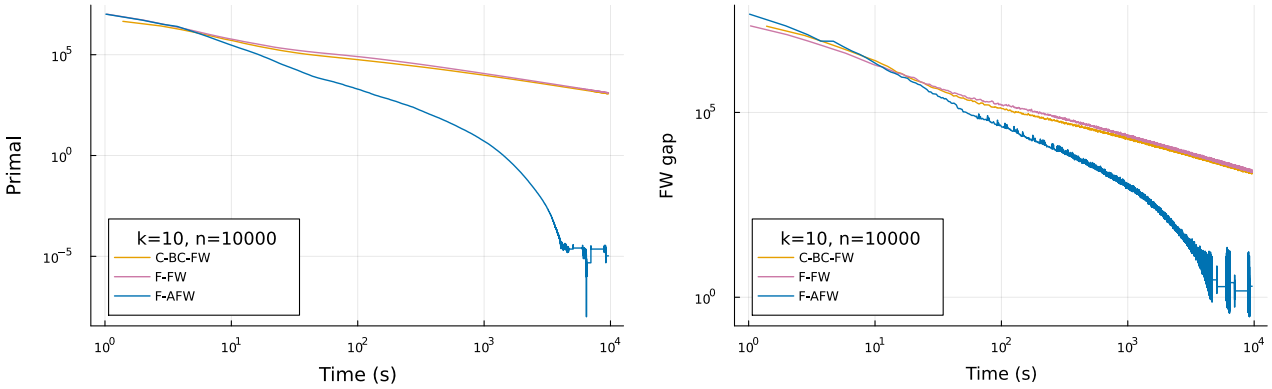

Figure 2: FW variants, run for up to 10 000 iterations on ten non-intersecting point-cloud polytopes in $\mathbb{R}^{10\,000}$.

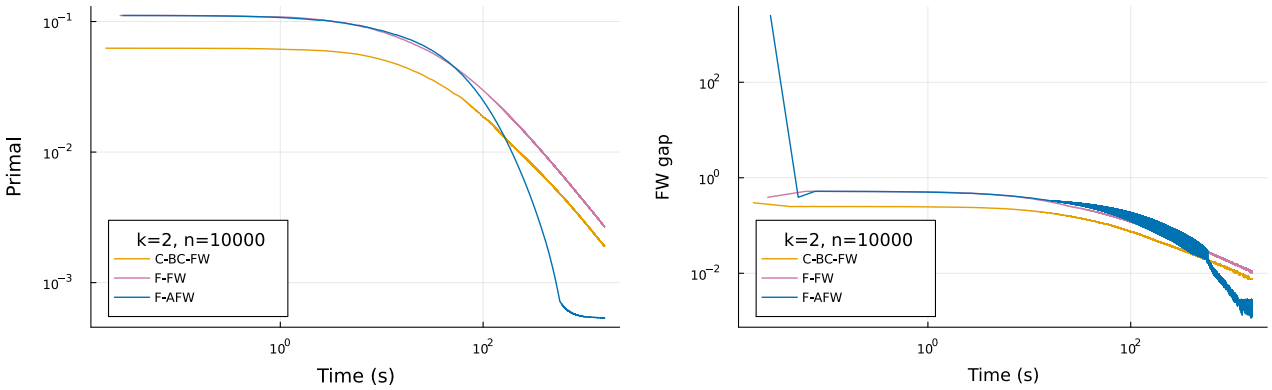

Figure 3: FW variants, run for up to 80 000 iterations on two intersecting vertex-facet polytopes in $\mathbb{R}^{10\,000}$.

non-monotonic fluctuations of the primal gap in the tail of Figure 2 are numerical artifacts caused by finite-precision effects and the approximate computation of the reference optimum.

The implementation was carried out fully in Julia (Version 1.9.4) and, in particular, we relied on the `FrankWolfe.jl` package (Version 0.5.0, MIT License) for the algorithmic part. All runs were executed on Slurm compute nodes equipped with Intel Xeon Gold 6 342 processors, each providing 96 logical cores, and approximately 512GiB RAM, interconnected via In-

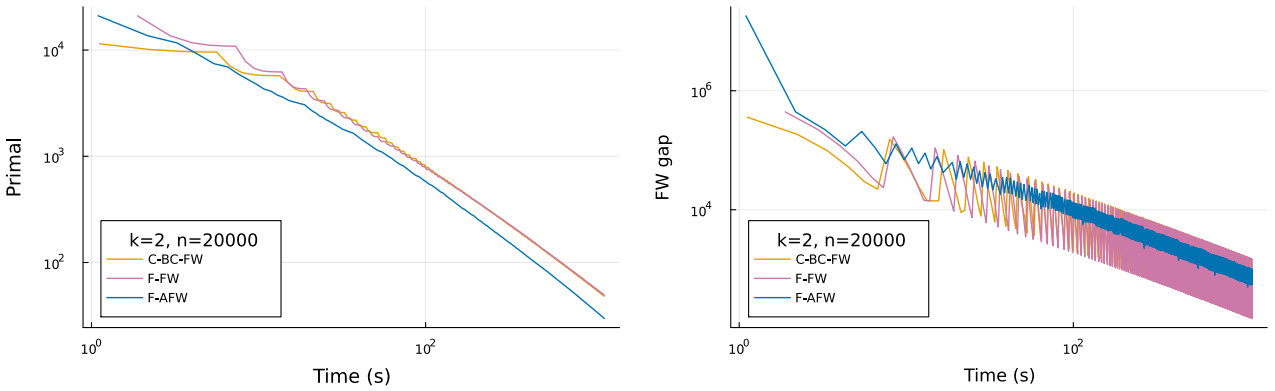

Figure 4: FW variants, run for up to 1 000 iterations on the two polytopes from Figure 1, translated to intersect.

finiband. We requested k cores per task in Slurm via `#SBATCH -cpus-per-task=k`, and then set `export JULIA_NUM_THREADS=$SLURM_CPUS_PER_TASK` so that Julia's FW routines used exactly k threads. The code for reproducing the results in this section is available at https://github.com/giommazz/ProductPolytopesFW.jl.

All in all, we found a consistent hierarchy across all experimental settings. We observe that the full AFW variant enjoys the best performance among the tested algorithms: away steps not only accelerate the algorithm in practice in the high-dimensional product-polytope setting, but are key for our linear convergence analysis, as the plots show. Even as k and n grow, AFW maintains a clear edge over the other variants, although a larger iteration budget is needed for the linear regime to emerge, which is expected. Instead, the algorithmic performance of the tested FW-based variants F-FW and C-BC-FW is very similar, and it yields worse convergence than F-AFW. Surprisingly, on our instances, the cyclic block-coordinate FW strategy offers no gain over the full FW variant, even with large k.

For the non-intersecting point-cloud instances, the empirical separation among the algorithms appears to be more pronounced than in the intersecting case. In particular, comparing Figure 4, reporting algorithmic performances on the same polytopes used in Figure 1 but translated so as to create a non-empty intersection, we observe that the curves in the intersecting case are closer to each other, suggesting a smaller performance gap. In these instances, a high-dimensional intersection yields a high-dimensional optimal set containing many points. This can make optimal-face identification less effective in practice. It is also consistent with less favorable local conditioning near the attained optimum, say on the optimal face, which plausibly reduces the empirical advantage of AFW over FW.

By contrast, in the vertex-facet construction the intersection is a single point by design, so the optimal set is isolated and the associated optimal face is geometrically much "sharper". In this setting, a larger face-dependent pyramidal-width-type constant is consistent with the clearer linear regime observed for AFW.

Additional examples showing similar behavior are reported in Section C. We also note that the convergence happens to be faster across all algorithms for the non-intersecting setting.

We note that we also experimented with a stochastic block-coordinate FW variant, which differs from C-BC-FW by selecting blocks in random order instead of cyclically. In our tests, its primal and FW gap trajectories were almost exactly the same as those of C-BC-FW, so we decided to omit it from our figures.

The plots showcase the superiority of employing AFW variants in speeding up convergence. Most importantly, our computational results provide empirical confirmation of our theoretical linear rates for AFW in several scenarios, as this algorithm converged at least as fast, and often consistently faster, than F-FW and C-BC-FW in all settings.

7 CONCLUSION

In this work, we have shown that the product of well-conditioned polytopes inherits a robust notion of conditioning, both in terms of pyramidal width and vertex–facet distance, which in turn guarantees linear convergence of the Away Frank-Wolfe method under standard assumptions.

Our analysis yields explicit bounds on the resulting condition numbers of the product domain, and thus on the convergence rate, as a function of the individual polytope components.

We then applied this theory to the classical convex-

feasibility problem over intersections of polytopes by minimizing a smooth, convex, Polyak-Łojasiewicz objective over the Cartesian product. We provided empirical results that strongly corroborate our findings.

Acknowledgements

The research in this paper was partially supported through the Research Campus Modal funded by the German Federal Ministry of Education and Research (fund numbers 05M14ZAM,05M20ZBM) and the Deutsche Forschungsgemeinschaft (DFG) through the DFG Cluster of Excellence MATH+. Francisco Criado was supported by grant PID2022-137283NB-C21. David Martínez-Rubio was partially funded by the Spanish Ministry of Science, Innovation, and Universities and by the State Research Agency (MICIU/AEI/10.13039/501100011033/) under grant PID2024-160448NA-I00 and by La Caixa Junior Leader Fellowship 2025.

References

Alcantara, Jan Harold, Jein-Shan Chen, and Matthew K Tam (2023). Method of alternating projections for the general absolute value equation. In: *Journal of Fixed Point Theory and Applications* 25.1, p. 39 (cit. on p. 2).

Badea, Catalin and David Seifert (2016). Ritt Operators And Convergence In The Method Of Alternating Projections. In: *Journal of Approximation Theory* 205, pp. 133–148. ISSN: 0021-9045 (cit. on p. 3).

Bauschke, Heinz and Jonathan Borwein (June 1993). On The Convergence Of Von Neumann's Alternating Projection Algorithm For Two Sets. In: *Set-Valued Analysis* 1, pp. 185–212 (cit. on p. 3).

Bauschke, Heinz H. and Jonathan M. Borwein (1996). On Projection Algorithms For Solving Convex Feasibility Problems. In: *SIAM Review* 38.3, pp. 367–426 (cit. on p. 3).

Bauschke, Heinz H. and Patrick L. Combettes (2011). *Convex Analysis And Monotone Operator Theory In Hilbert Spaces*. 1st. Springer Publishing Company, Incorporated. ISBN: 1441994661 (cit. on p. 3).

Beck, Amir (2017). *First-order methods in optimization*. SIAM (cit. on p. 4).

Beck, Amir, Edouard Pauwels, and Shoham Sabach (2015). The Cyclic Block Conditional Gradient Method for Convex Optimization Problems. In: *SIAM J. Optim.* 25.4, pp. 2024–2049 (cit. on pp. 3, 23).

Beck, Amir and Shimrit Shtern (2015). *Linearly Convergent Away-Step Conditional Gradient For Non-Strongly Convex Functions* (cit. on pp. 1, 3–5, 21, 22).

Beck, Amir and Marc Teboulle (Aug. 2003). Convergence Rate Analysis And Error Bounds For Projection Algorithms In Convex Feasibility Problems. In: *Optimization Methods And Software* 18, pp. 377–394 (cit. on p. 3).

Behling, Roger, Yunier Bello-Cruz, and Luiz-Rafael Santos (2021). Infeasibility And Error Bound Imply Finite Convergence Of Alternating Projections. In: *SIAM Journal on Optimization* 31.4, pp. 2863–2892. ISSN: 1095-7189 (cit. on p. 3).

Besançon, Mathieu, Sebastian Pokutta, and Elias Samuel Wirth (2025). The Pivoting Framework: Frank-Wolfe Algorithms with Active Set Size Control. In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 271–279 (cit. on p. 22).

Bomze, Immanuel, Francesco Rinaldi, and Damiano Zeffiro (2024). Projection free methods on product domains. In: *Computational Optimization and Applications*, pp. 1–30 (cit. on pp. 2, 3).

Braun, Gábor, Alejandro Carderera, Cyrille W Combettes, Hamed Hassani, Amin Karbasi, Aryan Mokhtari, and Sebastian Pokutta (2022). Conditional gradient methods. In: *arXiv preprint arXiv:2211.14103* (cit. on pp. 1, 6, 19–23).

Braun, Gábor, Sebastian Pokutta, and Robert Weismantel (2022). Alternating Linear Minimization: Revisiting von Neumann's alternating projections. In: *arXiv preprint arXiv:2212.02933* (cit. on pp. 3, 23).

Bui, Hoa T., Ryan C. Loxton, and Asghar Moeini (2021). A note on the finite convergence of alternating projections. In: *Oper. Res. Lett.* 49.3, pp. 431–438 (cit. on p. 3).

Carmi, Avishy, Yair Censor, and Pini Gurfil (2012). Convex feasibility modeling and projection methods for sparse signal recovery. In: *J. Comput. Appl. Math.* 236.17, pp. 4318–4335 (cit. on p. 2).

Censor, Yair, Wei Chen, Patrick L. Combettes, Ran Davidi, and Gabor T. Herman (2012). On the effectiveness of projection methods for convex feasibility problems with linear inequality constraints. In: *Comput. Optim. Appl.* 51.3, pp. 1065–1088 (cit. on p. 2).

Chakrabarti, Darshan, Gabriele Farina, and Christian Kroer (2024). Efficient Learning in Polyhedral Games via Best-Response Oracles. In: *Thirty-Eighth AAAI Conference on Artificial Intelligence,AAAI 2024*. Ed. by Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan. AAAI Press, pp. 9564–9572 (cit. on p. 3).

Combettes, Patrick L. (1996). The Convex Feasibility Problem In Image Recovery. In: ed. by Peter W. Hawkes. Vol. 95. Advances in Imaging and Electron Physics. Elsevier, pp. 155–270 (cit. on p. 2).

Combettes, Patrick L. and Jean-Christophe Pesquet (2011). Proximal Splitting Methods in Signal Processing. In: *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Ed. by Heinz H. Bauschke, Regina Sandra Burachik, Patrick L. Combettes, Veit Elser, D. Russell Luke, and Henry Wolkowicz. Vol. 49. Springer Optimization and Its Applications. Springer, pp. 185–212 (cit. on p. 2).

Dadush, Daniel, László A. Végh, and Giacomo Zambelli (2022). On finding exact solutions of linear programs in the oracle model. In: *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*. Ed. by Joseph (Seffi) Naor and Niv Buchbinder. SIAM, pp. 2700–2722 (cit. on p. 2).

Drusvyatskiy, Dmitriy, Alexander D. Ioffe, and Adrian S. Lewis (2015). Transversality and Alternating Projections for Nonconvex Sets. In: *Found. Comput. Math.* 15.6, pp. 1637–1651 (cit. on p. 3).

Garber, Dan and Elad Hazan (2015a). *A Linearly Convergent Conditional Gradient Algorithm With Applications To Online And Stochastic Optimization* (cit. on pp. 1, 3).

– (2015b). Faster rates for the Frank-Wolfe method over strongly-convex sets. In: *International Confer-*

ence on Machine Learning. PMLR, pp. 541–549 (cit. on p. 3).

Garber, Dan, Orna Livney, and Shoham Sabach (2023). Faster Projection-Free Augmented Lagrangian Methods via Weak Proximal Oracle. In: *International Conference on Artificial Intelligence and Statistics*. Ed. by Francisco Ruiz and Jennifer Dy. Vol. 206. Proceedings of Machine Learning Research. PMLR, pp. 553–573 (cit. on p. 3).

Gidel, Gauthier, Fabian Pedregosa, and Simon Lacoste-Julien (2018). *Frank-Wolfe Splitting via Augmented Lagrangian Method*. Ed. by Amos J. Storkey and Fernando Pérez-Cruz (cit. on p. 3).

Ginat, Omer (2018). *The Method Of Alternating Projections* (cit. on p. 3).

Gubin, L.G., B.T. Polyak, and E.V. Raik (1967). The Method Of Projections For Finding The Common Point Of Convex Sets. In: *USSR Computational Mathematics And Mathematical Physics* 7.6, pp. 1–24. ISSN: 0041-5553 (cit. on p. 3).

Guélat, Jacques and Patrice Marcotte (1986a). Some comments on Wolfe's 'away step'. In: *Mathematical Programming* 35.1, pp. 110–119 (cit. on p. 1).

– (1986b). Some Comments On Wolfe's "Away Step". In: *Mathematical Programming* 35.1, pp. 110–119 (cit. on pp. 2, 3).

Jaggi, Martin (2013). Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. In: *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*. Vol. 28. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 427–435 (cit. on pp. 1, 6).

Karimi, Hamed, Julie Nutini, and Mark Schmidt (2016). Linear Convergence of Gradient and Proximal-Gradient Methods Under the Polyak-Łojasiewicz Condition. In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I*. Ed. by Paolo Frasconi, Niels Landwehr, Giuseppe Manco, and Jilles Vreeken. Vol. 9851. Lecture Notes in Computer Science. Springer, pp. 795–811 (cit. on p. 4).

Kerdreux, Thomas, Alexandre d'Aspremont, and Sebastian Pokutta (2020). *Projection-Free Optimization On Uniformly Convex Sets* (cit. on p. 3).

– (2021). *Restarting Frank-Wolfe: Faster Rates Under Hölderian Error Bounds* (cit. on p. 19).

Lacoste-Julien, Simon and Martin Jaggi (2013). An affine invariant linear convergence analysis for Frank-Wolfe algorithms. In: *arXiv preprint arXiv:1312.7864* (cit. on pp. 1, 4).

– (2015). On the Global Linear Convergence of Frank-Wolfe Optimization Variants. In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. Ed. by Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, pp. 496–504 (cit. on pp. 1, 3, 4, 20).

Lacoste-Julien, Simon, Martin Jaggi, Mark Schmidt, and Patrick Pletscher (2013). Block-Coordinate Frank-Wolfe Optimization for Structural SVMs. In: *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*. Vol. 28. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 53–61 (cit. on p. 2).

Levitin, Evgeny S and Boris T Polyak (1966). Constrained minimization methods. In: *USSR Computational mathematics and mathematical physics* 6.5, pp. 1–50 (cit. on p. 1).

Ma, Shiqian and Necdet S. Aybat (2018). Efficient Optimization Algorithms For Robust Principal Component Analysis And Its Variants. In: *Proceedings of the IEEE* 106.8, pp. 1411–1426 (cit. on p. 2).

Martínez-Rubio, David and Sebastian Pokutta (2025). Beyond Short Steps in Frank-Wolfe Algorithms. In: *arXiv preprint arXiv:2501.18773* (cit. on p. 23).

Neumann, John Von (1949). On Rings Of Operators. Reduction Theory. In: *Annals of Mathematics* 50.2, pp. 401–485. ISSN: 0003486X (cit. on p. 3).

Pena, Javier F (2023). Affine invariant convergence rates of the conditional gradient method. In: *SIAM Journal on Optimization* 33.4, pp. 2654–2674 (cit. on p. 3).

Peña, Javier and Daniel Rodríguez (2018). Polytope Conditioning And Linear Convergence Of The Frank–Wolfe Algorithm. In: *Mathematics of Operations Research* 44.1, pp. 1–18 (cit. on pp. 1, 3–5).

Peña, Javier, Daniel Rodríguez, and Negar Soheili (2016). On the von Neumann and Frank-Wolfe Algorithms with Away Steps. In: *SIAM J. Optim.* 26.1, pp. 499–512 (cit. on p. 1).

Rademacher, Luis and Chang Shu (2022). The smoothed complexity of Frank–Wolfe methods via conditioning of random matrices and polytopes. In: *Mathematical Statistics and Learning* 5.3, pp. 273–310 (cit. on p. 5).

Richard, Emile, Pierre-André Savalle, and Nicolas Vayatis (2012). Estimation of simultaneously sparse and low rank matrices. In: *arXiv preprint arXiv:1206.6474* (cit. on p. 2).

Silveti-Falls, A., C. Molinari, and J. Fadili (2020). Generalized Conditional Gradient with Augmented Lagrangian for Composite Minimization. In: *SIAM Journal on Optimization* 30.4, pp. 2687–2725 (cit. on p. 3).

Stark, Henry, Yongi Yang, and Yongyi Yang (1998). *Vector Space Projections: A Numerical Approach To*

Signal And Image Processing, Neural Nets, And Optics. USA: John Wiley & Sons, Inc. ISBN: 0471241407 (cit. on p. 2).

Tsuji, Kazuma, Ken'ichiro Tanaka, and Sebastian Pokutta (2022). Pairwise Conditional Gradients without Swap Steps and Sparser Kernel Herding. In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Ed. by Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato. Vol. 162. Proceedings of Machine Learning Research. PMLR, pp. 21864–21883 (cit. on p. 3).

Willner, Leopold B. (1968). On The Distance Between Polytopes. In: *Quarterly of Applied Mathematics* 26.2, pp. 207–212. ISSN: 0033569X, 15524485 (cit. on p. 3).

Wirth, Elias, Thomas Kerdreux, and Sebastian Pokutta (2023). Acceleration of Frank-Wolfe algorithms with open-loop step-sizes. In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 77–100 (cit. on p. 3).

Wirth, Elias, Javier Pena, and Sebastian Pokutta (2024). Fast Convergence of Frank-Wolfe algorithms on polytopes. In: *arXiv preprint arXiv:2406.18789* (cit. on p. 3).

– (2025). Accelerated affine-invariant convergence rates of the Frank–Wolfe algorithm with open-loop step-sizes. In: *Mathematical Programming*, pp. 1–45 (cit. on p. 3).

Wirth, Elias, Sebastian Pokutta, et al. (2023). Approximate Vanishing Ideal Computations at Scale. In: *Proceedings of International Conference on Learning Representations* (cit. on p. 3).

Wolfe, Philip (1970). Convergence theory in nonlinear programming. In: *Integer and nonlinear programming*, pp. 1–36 (cit. on p. 1).

– (1976). Finding The Nearest Point In A Polytope. In: *Mathematical Programming* 11.1, pp. 128–149 (cit. on p. 3).

Checklist

1. For all models and algorithms presented, check if you include:

   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model.

   [Yes] Notation is described in Section 3. All assumptions on convexity, smoothness and the Polyak–Łojasiewicz condition, listed in Section 3, are stated in our linear convergence theorems appearing in Section 4.1. The minimal hypotheses on the polytopes whose condition numbers we examine appear in Section 4. All of the full proofs are given in Section A, except for the very short one of Corollary 4.4, that is located in the main paper. The examined Frank–Wolfe variants are described in Section B.

   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm.

   [Yes] We analyze linear convergence rates for Away-FW over product polytopes under PL, with rates expressed via the polytope condition numbers known as pyramidal width and the vertex–facet distance. Notably, we quantify these condition numbers for product polytopes and connect them to iteration complexity in Section 4, Section 5, Section A.4, and Section A.5.

   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries.

   [Yes] The data and code is available in the supplementary material. We will publish all code, data, and logs in a publicly accessible GitHub repository, including (i) the complete `Julia` package source for generating instances and running the experiments (including `Project.toml` and `Manifest.toml` files pinning all dependency versions), fully commented to assist both expert and beginner users in understanding the code, (ii) all `.CSV` logs used to generate the paper's plots, alongside scripts for parsing and plotting, (iii) comprehensive `README` files with step-by-step instructions for installing the package, reproducing the experiments both locally or on a SLURM-managed server, and re-generating all plots.

2. For any theoretical claim, check if you include:

   (a) Statements of the full set of assumptions of all theoretical results.

   [Yes] Assumptions on convexity, smoothness, PL condition, and polytope structure are stated with each result; see Section 3, Section 4, and Section 5

   (b) Complete proofs of all theoretical results.

   [Yes] Full proofs appear in Section A. The short proof of Corollary 4.4 is kept in the main text.

   (c) Clear explanations of any assumptions.

   [Yes] Assumptions are motivated and discussed alongside the main theorems and their proofs.

3. For all figures and tables that present empirical results, check if you include:

   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL).

   [Yes]

   Section 6 and Section C describe dataset generation, parameter settings (k, n, maximum number of iterations for the tested algorithms, their step size strategy and termination criterion), and the hardware and software environment. In particular, all datasets are synthetically generated with the procedure described in the paper, and more documentation can be found directly in the code. The full code to reproduce these experiments will be made available upon publication and is attached in the supplementary material. The manuscript also includes pseudocode in Section B.

   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen).

   [Yes] We report the experimental-configuration details that serve as training details for optimization studies: (i) instance-generation protocols for intersecting and disjoint product polytopes, with all parameter values and random seeds; (ii) algorithmic hyperparameters for each Frank–Wolfe variant, including step-size policy and away-step selection; (iii) stopping criteria (maximum iterations and Frank–Wolfe gap threshold); and (iv) software/hardware environment. Because instances are synthetically generated, train/validation/test splits are not applicable; comparability is ensured via fixed seeds and replicated runs. See Section 6 and Section C.

   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times).

   [Yes] We quantify run-to-run variability by including additional plots, pertaining to instances generated with different random seeds, in Section C. These supplementary figures closely mirror the behavior of the tested algorithms that is already shown in the main paper, and confirm our theoretical findings. This approach, namely, providing additional figures which plot performance curves across multiple seeds, serves the purpose of confirming the statistical significance of our claims.

   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider).

   [Yes] Experiments were run on SLURM-managed nodes with Intel Xeon Gold 6 342 CPUs (96 logical cores) and ∼512 GiB RAM over InfiniBand; we specify threading and SLURM settings (`cpus-per-task` and `JULIA_NUM_THREADS`) in Section 6.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

   (a) Citations of the creator If your work uses existing assets.

   [Yes] We cite `FrankWolfe.jl` and all prior theoretical results underpinning our analysis.

   (b) The license information of the assets, if applicable.

   [Yes] We state that the `FrankWolfe.jl` is MIT license and include license details for any external packages used.

   (c) New assets either in the supplemental material or as a URL, if applicable.

   [Yes] We release an anonymized code archive in the supplementary material. A public repository will be posted on GitHub upon publication with code, logs, and scripts.

   (d) Information about consent from data providers/curators.

   [Not Applicable] All data are synthetic and generated as described. No external providers or human data are involved.

   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content.

   [Not Applicable] The work contains no sensitive content or personally identifiable information. Synthetic instances contain only randomly generated numeric data.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

   (a) The full text of instructions given to participants and screenshots.

   [Not Applicable] The paper involves no crowdsourcing or human subjects.

   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB)

approvals if applicable.

[Not Applicable] No human studies were conducted.

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation.

[Not Applicable] No participants were recruited.

Supplementary Material

A PROOFS

A.1 Pyramidal Width

Lemma A.1 contains some simple known facts about polytopes, that serve as a warm-up and will be useful in the sequel.

Lemma A.1 (Cartesian product of k polytopes). *The Cartesian product $\mathcal{P} = \Pi_{i\in[k]}\mathcal{P}_i$ of k polytopes, each in $\mathbb{R}^n$ for some $n \in \mathbb{N}_{>0}$, is a polytope of dimension kn. In particular, $\mathcal{V}_\mathcal{P} = \Pi_{i\in[k]}\mathcal{V}_{\mathcal{P}_i}$. Further, if $F_\mathcal{P} \in \mathsf{faces}^*(\mathcal{P})$, then $F_\mathcal{P}$ is the product of faces of all the individual polytopes, i.e., $\forall i \in [k]$ there exist $F_{\mathcal{P}_i} \in \mathsf{faces}^*(\mathcal{P}_i)$ such that $F_\mathcal{P} = \Pi_{i\in[k]}F_{\mathcal{P}_i}$.*

Proof For all $i \in [k]$, we denote the i-th polytope by $\mathcal{P}_i \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n : A_{\mathcal{P}_i}x \le b_{\mathcal{P}_i}\}$, where $A_{\mathcal{P}_i} \in \mathbb{R}^{m_{\mathcal{P}_i}\times n}$ and $b_{\mathcal{P}_i} \in \mathbb{R}^{m_{\mathcal{P}_i}}$. Compactness and convexity are inherently preserved in the Cartesian product, so

$$\begin{aligned}\Pi_{i\in[k]} &= \{(x^1,\dots,x^k) \in \mathbb{R}^n \times \cdots \times \mathbb{R}^n : x^1 \in \mathcal{P}_1,\dots,x^k \in \mathcal{P}_k\}\\ &= \{(x \in \mathbb{R}^{kn} : \forall i \in [k]\,,\, A_{\mathcal{P}_i}x^i \le b_{\mathcal{P}_i}\}\,.\end{aligned}$$

By linearity of the constraints and independence of each block $i \in [k]$, the product is still a polytope described by a constraint matrix in $\mathbb{R}^{(m_{\mathcal{P}_1}+\cdots+m_{\mathcal{P}_k})\times kn}$ with diagonal blocks $A_{\mathcal{P}_i}$, for $i \in [k]$, and right-hand side coefficients $b_{\mathcal{P}_i} \in \mathbb{R}^{m_{\mathcal{P}_1}+\cdots+m_{\mathcal{P}_k}}$. Moreover, since $\forall i \in [k]$, $\mathcal{P}_i = \mathsf{conv}(\mathcal{P}_i)$ and $\mathsf{conv}\left(\Pi_{i\in[k]}\mathcal{P}_i\right) = \Pi_{i\in[k]}\mathsf{conv}(\mathcal{P}_i)$, then $\mathcal{V}_\mathcal{P} = \Pi_{i\in[k]}\mathcal{V}_{\mathcal{P}_i}$. Finally, a polytope face is the set of points optimizing some linear functional over the polytope. Consider the linear functional defined by $n = (n^1,\dots,n^k) \in \mathbb{R}^k$. A point $(p^1,\dots,p^k) \in \mathcal{P}$ optimizes the linear functional defined by n if and only if, each p^i optimizes the linear functional defined by the corresponding n^i over $\mathcal{P}_i$, as a linear program in the product is solved in each component independently. If, for all $i \in [k]$, we denote the faces defined by that condition as $F_i \subseteq \mathcal{P}_i$, the set of points optimizing n,is exactly $F_1 \times F_2 \times \cdots \times F_k \subset \mathcal{P}_1 \times \mathcal{P}_2 \times \cdots \times \mathcal{P}_k$, i.e., a face $F_\mathcal{P}$ of $\mathcal{P}$. ■

Proof of Lemma 4.3.

Suppose that p is not in the relative interior of f. Then p might be in the exterior or in the boundary of f. Choose one facet g of f such that $\mathsf{aff}(g)$ separates p from the interior of f (p could lie exactly in $\mathsf{aff}(g)$ if p is in the relative boundary of f). We will show that $\bar{\Delta}_g(\mathcal{P}) < \bar{\Delta}_f(\mathcal{P})$, which would be a contradiction.

Since g is a face of f, there is some vertex v of f that is not a vertex of g, which means $v \notin \mathsf{aff}(g)$. Let p' be the intersection of the segment $[p, v]$ with the affine subspace $\mathsf{aff}(g)$. This intersection exists and is unique because $\mathsf{aff}(g)$ separates p and v, $[p, v]$ is 1-dimensional, and $\mathsf{aff}(g)$ is 1-codimensional as a subspace of $\mathsf{aff}(f)$.

Let q be the point of $\mathcal{P}^f$ that is closest to p. Finally, let q' be the point of the segment $[v, q]$ that is closest to $\mathsf{aff}(g)$. By construction, $q' \in \mathcal{P}^g$. Observe from Figure 5 that, in the right triangle v, p, q, the segment corresponding to $\mathsf{dist}(p', q')$ is at most as long as its height $\mathsf{dist}(p, q)$ over the hypotenuse, giving the key inequality:

$$\bar{\Delta}_g(\mathcal{P}) \le \mathsf{dist}(\mathsf{aff}(g), q') \le \mathsf{dist}(p', q') < \mathsf{dist}(p, q) = \bar{\Delta}_f(\mathcal{P}).$$

■

In the sequel we will use $\delta_\mathcal{P}$ to denote both the affine pyramidal width and the pyramidal width. However, we will use the definition of affine pyramidal width to prove the following theorem.

Proof of Theorem 4.5. We will first prove the upper bound by explicitly constructing the face of the product that achieves the affine pyramidal width we claim, as well as the points that achieve the minimum distance. In

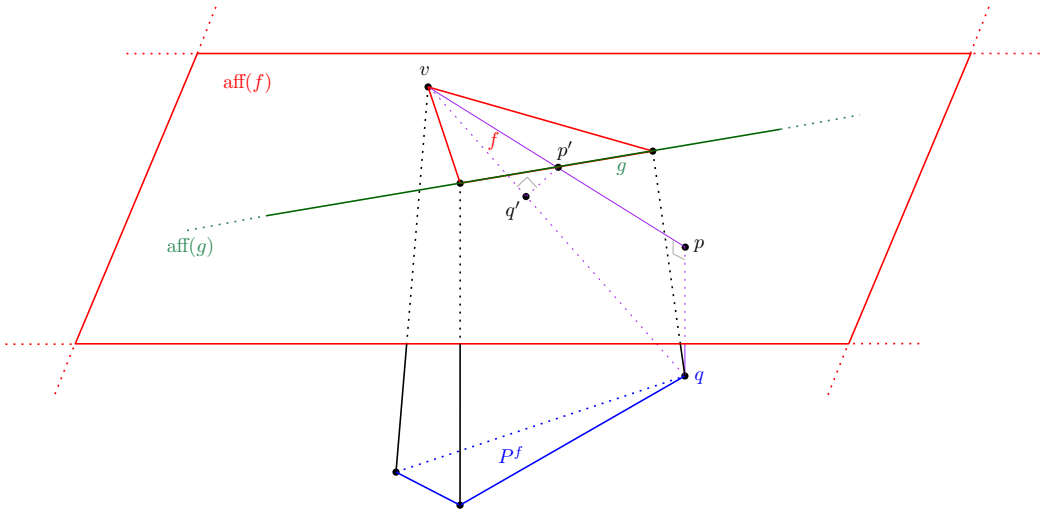

Figure 5: To prove that $\mathsf{dist}\left(\mathsf{aff}\left(g\right),[q,v]\right) < \mathsf{dist}\left(\mathsf{aff}\left(f\right),q\right)$, we look at the right triangle $\triangle vpq$ and compare $\mathsf{dist}\left(p',q'\right)$ with $\mathsf{dist}\left(p,q\right)$.

the second part of the proof, we will prove that no other face of the product can have a smaller affine pyramidal width.

Let f_1 be a face of $\mathcal{P}_1$ such that $\delta_{\mathcal{P}_1} = \bar{\Delta}_{f_1}(\mathcal{P}_1)$. Let $p_1 \in f_1$, $q_1 \in \mathcal{P}_1^{f_1}$, and $\delta_1 \in \mathbb{R}$ be such that $\mathsf{dist}\left(p_1,q_1\right) = \delta_1 = \delta_{\mathcal{P}_1}$. Note that we know that $p_1 \in f_1$ and not just $p \in \mathsf{aff}\left(f_1\right)$ because of Lemma 4.3 and the optimality of f_1. We define f_2, p_2, q_2, and δ_2 analogously.

Let

$$\begin{aligned} p &= (p_1, p_2), \\ q &= (p_1, q_2)\frac{\delta_1^2}{\delta_1^2+\delta_2^2} + (q_1, p_2)\frac{\delta_2^2}{\delta_1^2+\delta_2^2}. \end{aligned}$$

We have that $p \in f_1 \times f_2$ by construction. Now, we also have $q \in (\mathcal{P}_1 \times \mathcal{P}_2)^{f_1\times f_2}$. Indeed, q is a convex combination of the points (p_1,q_2) and (q_1,p_2). The point (p_1,q_2) is in $f_1 \times \mathcal{P}_2^{f_2}$, which is a subset of $(\mathcal{P}_1 \times \mathcal{P}_2)^{f_1\times f_2}$. Analogously, (q_1,p_2) is also in $(\mathcal{P}_1 \times \mathcal{P}_2)^{f_1\times f_2}$. Thus, their convex combination is also in $(\mathcal{P}_1 \times \mathcal{P}_2)^{f_1\times f_2}$.

Now, let us compute $\mathsf{dist}\left(p,q\right)$, which is an upper bound for $\delta_{\mathcal{P}_1\times\mathcal{P}_2}$:

$$\begin{aligned} \mathsf{dist}\left(p,q\right) &= \left\| (p_1,p_2) - \left((p_1,q_2)\frac{\delta_1^2}{\delta_1^2+\delta_2^2} + (q_1,p_2)\frac{\delta_2^2}{\delta_1^2+\delta_2^2} \right) \right\| \\ &= \left\| \left((p_1-q_1)\,\frac{\delta_2^2}{\delta_1^2+\delta_2^2}, (p_2-q_2)\,\frac{\delta_1^2}{\delta_1^2+\delta_2^2} \right) \right\| \\ &= \sqrt{\frac{\left(\delta_2^2+\delta_1^2\right)\delta_1^2\delta_2^2}{\left(\delta_1^2+\delta_2^2\right)^2}} \\ &= \frac{1}{\sqrt{\frac{1}{\delta_1^2}+\frac{1}{\delta_2^2}}}. \end{aligned}$$

Now let us prove the lower bound. This time, let f_1 and f_2 be *arbitrary* faces of $\mathcal{P}_1$ and $\mathcal{P}_2$ and denote $\bar{\delta}_1 \stackrel{\text{def}}{=} \bar{\Delta}_{f_1}(\mathcal{P}_1)$ and $\bar{\delta}_2 \stackrel{\text{def}}{=} \bar{\Delta}_{f_2}(\mathcal{P}_2)$.

Since $\mathsf{dist}\left(\mathsf{aff}\left(f_1\right),\mathcal{P}_1^{f_1}\right) = \bar{\delta}_1$ and $\mathsf{dist}\left(\mathsf{aff}\left(f_2\right),\mathcal{P}_2^{f_2}\right) = \bar{\delta}_2$, by the Hyperplane Separation Theorem there are linear functionals:

$$\pi_1 : \mathbb{R}^n \to \mathbb{R}, \quad \pi_1(x) = \langle u_1, x \rangle + c_1,$$
$$\pi_2 : \mathbb{R}^m \to \mathbb{R}, \quad \pi_2(y) = \langle u_2, y \rangle + c_2,$$

such that $\pi_1(x) = \bar{\delta}_1$, $\pi_2(y) = \bar{\delta}_2$ for all $x \in \mathsf{aff}(f_1)$ and $y \in \mathsf{aff}(f_2)$, and $\pi_1(x) \leq 0$, $\pi_2(y) \leq 0$ for all $x \in \mathcal{P}_1^{f_1}$, $y \in \mathcal{P}_2^{f_2}$. Furthermore $|u_1| = |u_2| = 1$. Observe that π_1, π_2 are constant over $\mathsf{aff}(f_1)$ and $\mathsf{aff}(f_2)$ respectively. Indeed, for example if π_1 were not constant over $\mathsf{aff}(f_1)$ then $\pi_1(\mathsf{aff}(f_1)) = \mathbb{R}$ so π_1 would not be a separating functional.

We define the linear functional:

$$\pi : \mathbb{R}^{n+m} \to \mathbb{R}, \quad \pi(x,y) = \frac{\frac{\pi_1(x)}{\bar{\delta}_1} + \frac{\pi_2(y)}{\bar{\delta}_2} - 1}{\sqrt{\frac{1}{\bar{\delta}_1^2} + \frac{1}{\bar{\delta}_2^2}}} = \left\langle \frac{\left(\frac{u_1}{\bar{\delta}_1}, \frac{u_2}{\bar{\delta}_2}\right)}{\sqrt{\frac{1}{\bar{\delta}_1^2} + \frac{1}{\bar{\delta}_2^2}}}, (x,y) \right\rangle + c.$$

Observe that the gradient of π is unitary by construction. Also, $\pi(x,y) = \frac{1}{\sqrt{\frac{1}{\bar{\delta}_1^2} + \frac{1}{\bar{\delta}_2^2}}}$ for all $(x,y) \in \mathsf{aff}(f_1) \times \mathsf{aff}(f_2)$.

Now let us study $\pi(x,y)$ over the polytope $(\mathcal{P}_1 \times \mathcal{P}_2)^{f_1 \times f_2}$.

Since $(\mathcal{P}_1 \times \mathcal{P}_2)^{f_1 \times f_2}$ is a polytope, π is maximized at a vertex of $(\mathcal{P}_1 \times \mathcal{P}_2)^{f_1 \times f_2}$. The vertices of this polytope are of the form (v_1, v_2) where $v_1 \in \mathcal{V}_{\mathcal{P}_1}$ and $v_2 \in \mathcal{V}_{\mathcal{P}_2}$ but removing the vertices of $f_1 \times f_2$. In other words, $v_1 \notin \mathsf{aff}(f_1)$ or $v_2 \notin \mathsf{aff}(f_2)$.

Without loss of generality we consider the case where $v_2 \in \mathcal{V}_{\mathcal{P}_2}$. Then, $\pi_1(v_1) \leq \bar{\delta}_1$ and $\pi_2(v_2) \leq 0$, which implies $\pi(v_1, v_2) \leq 0$. Therefore, π is a separating functional for $(\mathcal{P}_1 \times \mathcal{P}_2)^{f_1 \times f_2}$ and $\mathsf{aff}(f_1) \times \mathsf{aff}(f_2)$.

Recall that π is unitary, so the distance between $(\mathcal{P}_1 \times \mathcal{P}_2)^{f_1 \times f_2}$ and $\mathsf{aff}(f_1) \times \mathsf{aff}(f_2)$ is at least $\frac{1}{\sqrt{\frac{1}{\bar{\delta}_1^2} + \frac{1}{\bar{\delta}_2^2}}}$. In other words, $\bar{\Delta}_{f_1 \times f_2}(\mathcal{P}_1 \times \mathcal{P}_2) \geq \frac{1}{\sqrt{\frac{1}{\bar{\delta}_1^2} + \frac{1}{\bar{\delta}_2^2}}}$.

By repeating this argument for every pair of faces of $\mathcal{P}_1$ and $\mathcal{P}_2$:

$$\begin{aligned}
\delta_{\mathcal{P}_1 \times \mathcal{P}_2} &= \min_{\substack{f_1 \in F(\mathcal{P}_1) \\ f_2 \in F(\mathcal{P}_2)}} \bar{\Delta}_{f_1 \times f_2}(\mathcal{P}_1 \times \mathcal{P}_2) \\
&\geq \min_{\substack{f_1 \in F(\mathcal{P}_1) \\ f_2 \in F(\mathcal{P}_2)}} \left(\frac{1}{\sqrt{\frac{1}{\left(\bar{\Delta}_{f_1}(\mathcal{P}_1)\right)^2} + \frac{1}{\left(\bar{\Delta}_{f_2}(\mathcal{P}_2)\right)^2}}} \right) \\
&= \frac{1}{\sqrt{\frac{1}{\left(\min_{f_1 \in F(\mathcal{P}_1)} \bar{\Delta}_{f_1}(\mathcal{P}_1)\right)^2} + \frac{1}{\left(\min_{f_2 \in F(\mathcal{P}_2)} \bar{\Delta}_{f_2}(\mathcal{P}_2)\right)^2}}} \\
&= \frac{1}{\sqrt{\frac{1}{\delta_{\mathcal{P}_1}^2} + \frac{1}{\delta_{\mathcal{P}_2}^2}}}.
\end{aligned}$$

This concludes the proof. ■

Lemma A.2. *Let $a, b > 0$. Then*

$$\frac{1}{\sqrt{2}} \min\{a,b\} \leq \sqrt{\frac{a^2 b^2}{a^2 + b^2}} < \min\{a,b\} \tag{5}$$

and $\sqrt{\frac{a^2 b^2}{a^2+b^2}}$ is monotonically nondecreasing in both a and b over $\mathbb{R}_{++}$.

Proof Assume without loss of generality that $a = \min\{a, b\}$. Then, we have $\sqrt{\frac{a^2b^2}{a^2+b^2}} \geq \sqrt{\frac{a^2b^2}{2b^2}} = \frac{a}{\sqrt{2}}$, which corresponds to the lower bound. The upper bound follows from the fact that $\frac{b^2}{a^2+b^2} < 1$. Monotonicity is verified by examining the partial derivatives, which are always positive. ■

Corollary A.3. *Let $\mathcal{P} \stackrel{\text{def}}{=} \prod_{i\in[k]} \mathcal{P}_i$ be a product of polytopes. Then,*

$$\delta_{\mathcal{P}} \geq \frac{1}{\sqrt{2}^{\lceil \log_2(k+1)\rceil}} \min_{i\in[k]} \delta_{\mathcal{P}_i} = \Omega\left(\frac{1}{\sqrt{k}} \min_{i\in[k]} \delta_{\mathcal{P}_i}\right).$$

Proof We use induction and Lemma A.1 to encode the facial distance of the k-Cartesian product. For the lower bound, the base case with $k = 2$ is implied by Theorem 4.5 and Lemma A.2. For the induction step, we assume that the lower bound holds for the Cartesian product of $m \leq \lceil k/2 \rceil$ polytopes and consider the two polytopes $\mathcal{P}_1 \times \cdots \times \mathcal{P}_{\lceil k/2\rceil}$ and $\mathcal{P}_{\lceil k/2\rceil+1} \times \cdots \times \mathcal{P}_k$. Then,

$$\begin{aligned}
\delta_{1:k} &\geq \tfrac{1}{\sqrt{2}} \min\left\{\delta_{\mathcal{P}_1 \cdots \mathcal{P}_{\lceil k/2\rceil}}, \delta_{\mathcal{P}_{\lceil k/2\rceil+1} \cdots \mathcal{P}_k}\right\} \\
&\geq \tfrac{1}{\sqrt{2}} \min\left\{ \frac{1}{\sqrt{2}^{\lceil \log_2 \lceil k/2\rceil\rceil}} \min_{i\in\{1,\ldots,\lceil k/2\rceil\}} \delta_{\mathcal{P}_i}\,,\ \frac{1}{\sqrt{2}^{\lceil \log_2 \lceil k/2\rceil\rceil}} \min_{j\in\{\lceil k/2\rceil+1,\ldots,k\}} \delta_{\mathcal{P}_j} \right\} \\
&\geq \frac{1}{\sqrt{2}^{1+\lceil \log_2 \lceil k/2\rceil\rceil}} \min_{i\in[k]} \delta_{\mathcal{P}_i} \\
&\geq \frac{1}{\sqrt{2}^{\lceil \log_2(k+1)\rceil}} \min_{i\in[k]} \delta_{\mathcal{P}_i} = \Omega\left(\frac{1}{\sqrt{k}} \min_{i\in[k]} \delta_{\mathcal{P}_i}\right),
\end{aligned}$$

where the first inequality is the base case with two polytopes, the second one applies the induction step, the third one uses $\sqrt{2}^{\lceil \log_2 \lceil k/2\rceil\rceil} \geq \sqrt{2}^{\lfloor \log_2 \lfloor k/2\rfloor\rfloor}$ and computes the inner min.

■

A.2 Vertex-Facet Distance

In order to prove Proposition 4.7, we define the *inner vertex-facet distance* of $\mathcal{P}$ with respect to $F_{\mathcal{P}} \in \mathsf{facets}(\mathcal{P})$ as

$$\mathsf{vf}_{\mathcal{P},F_{\mathcal{P}}} \stackrel{\text{def}}{=} \mathsf{dist}\left(\mathsf{aff}\left(F_{\mathcal{P}}\right), \mathcal{V}_{\mathcal{P}} \setminus \mathcal{V}_{F_{\mathcal{P}}}\right),$$

so that $\mathsf{vf}_{\mathcal{P}} = \min_{F_{\mathcal{P}}\in\mathsf{facets}(\mathcal{P})} \mathsf{vf}_{\mathcal{P},F_{\mathcal{P}}}$.

Proof of Proposition 4.7. It is enough to show the statement for the Cartesian product of two polytopes $\mathcal{P}$ and $\mathcal{Q}$. By Definition 4.6,

$$\mathsf{vf}_{\mathcal{P}\times\mathcal{Q}} = \min_{F\in\mathsf{facets}(\mathcal{P}\times\mathcal{Q})} \mathsf{vf}_{\mathcal{P}\times\mathcal{Q},F} = \min_{F\in\mathsf{facets}(\mathcal{P}\times\mathcal{Q})} \mathsf{dist}\left(\mathsf{aff}\left(F\right), \mathcal{V}_{\mathcal{P}\times\mathcal{Q}} \setminus \mathcal{V}_F\right). \tag{6}$$

All product facets F are of the form $F_{\mathcal{P}} \times \mathcal{Q}$, where $F_{\mathcal{P}}$ is a facet of $\mathcal{P}$, or $\mathcal{P} \times F_{\mathcal{Q}}$, where $F_{\mathcal{Q}}$ is a facet of $\mathcal{Q}$. Given $F_{\mathcal{P}} \times \mathcal{Q} \in \mathsf{facets}(\mathcal{P} \times \mathcal{Q})$, its vertices are

$$\begin{aligned}
\mathcal{V}_{\mathcal{P}\times\mathcal{Q}} \setminus \mathcal{V}_{F_{\mathcal{P}}\times\mathcal{Q}} &= (\mathcal{V}_{\mathcal{P}} \times \mathcal{V}_{\mathcal{Q}}) \setminus (\mathcal{V}_{F_{\mathcal{P}}} \times \mathcal{V}_{\mathcal{Q}}) \\
&= (\mathcal{V}_{\mathcal{P}} \setminus \mathcal{V}_{F_{\mathcal{P}}}) \times \mathcal{V}_{\mathcal{Q}}\,.
\end{aligned} \tag{7}$$

so

$$\begin{aligned}
\mathsf{vf}_{\mathcal{P}\times\mathcal{Q},F_{\mathcal{P}}\times\mathcal{Q}} &= \mathsf{dist}\left(\mathsf{aff}\left(F_{\mathcal{P}} \times \mathcal{Q}\right), (\mathcal{V}_{\mathcal{P}} \setminus \mathcal{V}_{F_{\mathcal{P}}}) \times \mathcal{V}_{\mathcal{Q}}\right) \\
&= \min_{(x,y)\in\mathsf{aff}(F_{\mathcal{P}}\times\mathcal{Q})} \min_{(z_1,z_2)\in(\mathcal{V}_{\mathcal{P}}\setminus\mathcal{V}_{F_{\mathcal{P}}})\times\mathcal{V}_{\mathcal{Q}}} \|(x,y) - (z_1,z_2)\| \\
&= \min_{\substack{x\in\mathsf{aff}(F_{\mathcal{P}}) \\ y\in\mathsf{aff}(\mathcal{Q})}} \min_{\substack{z_1\in\mathcal{V}_{\mathcal{P}}\setminus\mathcal{V}_{F_{\mathcal{P}}} \\ z_2\in\mathcal{V}_{\mathcal{Q}}}} \sqrt{\|x - z_1\|^2 + \|y - z_2\|^2} \\
&= \mathsf{vf}_{\mathcal{P},F_{\mathcal{P}}}\,.
\end{aligned}$$

The first equality is the definition of inner vertex-facet distance (see Definition 4.6). In the second equality, we apply the fact that $\mathsf{aff}\left(F \times \mathcal{Q}\right) = \mathsf{aff}\left(F\right) \times \mathsf{aff}\left(\mathcal{Q}\right)$ and Eq. (7). The third equality is just a reformulation over

the individual sets in $\mathcal{P}$ and in $\mathcal{Q}$. To obtain the last equality, we minimize the first summand by choosing x, z_1 achieving $\mathsf{vf}_{\mathcal{P},F_\mathcal{P}}$ and the second summand by selecting $y = z_2$ so that $\|y - z_2\|^2 = 0$, which we can do as $\mathcal{V}_\mathcal{Q} \subset \mathsf{aff}(\mathcal{Q})$. Similarly, the vertices of facets of the form $\mathcal{P} \times F_\mathcal{Q}$ are $\mathcal{V}_{\mathcal{P}\times\mathcal{Q}} \setminus \mathcal{V}_{\mathcal{P}\times F_\mathcal{Q}} = \mathcal{V}_\mathcal{P} \times (\mathcal{V}_\mathcal{Q} \setminus \mathcal{V}_{F_\mathcal{Q}})$, thus $\mathsf{vf}_{\mathcal{P}\times\mathcal{Q},\mathcal{P}\times F_\mathcal{Q}} = \mathsf{vf}_{\mathcal{Q},F_\mathcal{Q}}$.

Plugging the above equalities in Eq. (6) yields:

$$\begin{aligned}\mathsf{vf}_{\mathcal{P}\times\mathcal{Q}} &= \min_{F\in\mathsf{facets}(\mathcal{P}\times\mathcal{Q})} \mathsf{vf}_{\mathcal{P}\times\mathcal{Q},F} \\ &= \min\left\{\min_{F_\mathcal{P}\in\mathsf{facets}(\mathcal{P})} \mathsf{vf}_{\mathcal{P},F_\mathcal{P}}, \min_{F_\mathcal{Q}\in\mathsf{facets}(\mathcal{Q})} \mathsf{vf}_{\mathcal{Q},F_\mathcal{Q}}\right\} \\ &= \min\{\mathsf{vf}_\mathcal{P}, \mathsf{vf}_\mathcal{P}\},\end{aligned}$$

where the last equality follows from the definition of inner vertex-facet distance. ■

A.3 Properties of the Objective Function in Eq. (4)

Proof of Proposition 5.1. We establish the properties of the objective function appearing in Eq. (4). We recall that $f(x) = \frac{1}{2k}\langle M_k x, x\rangle$, with $M_k \overset{\text{def}}{=} (kI_{k\times k} - \mathbb{1}_k\mathbb{1}_k^T) \otimes I_n \in \mathbb{R}^{nk\times nk}$. In the following, we use the shorthand $A_k \overset{\text{def}}{=} (kI_{k\times k} - \mathbb{1}_k\mathbb{1}_k^T)$. As $f(x)$ is twice differentiable on $\mathbb{R}^{nk}$, we can compute its Hessian $\nabla^2 f(x) = \frac{1}{k}M_k$. Moreover, $f(x)$ is a weighted sum of squared Euclidean norms with positive weights, so $\langle M_k x, x\rangle \geq 0$ for all $x \in \mathbb{R}^{nk}$. Thus, M_k is positive semidefinite and f is convex.

Next, we characterize the eigenvalues of the Hessian $\frac{1}{k}M_k$.

First we note that, as M_k is symmetric, and thus its eigenvalues are real. Then, since f is convex and twice differentiable, the L-smoothness condition in Definition 3.1 is equivalent to the eigenvalues of the Hessian being upper bounded by L, i.e., whether it holds that $\frac{1}{k}\mu(M_k) \leq L$, for all $\mu(M_k) \in \mathbb{R}$. As the eigenvalues of a Kronecker product are the product of the eigenvalues of the involved matrices, counting algebraic multiplicities, the eigenvalues of the Hessian are of the form $\frac{1}{k}\mu(A_k)$, with $\mu(A_k)$ being the eigenvalues of A_k. Notice that $\mathbb{1}_k\mathbb{1}_k^T$ is a rank-1 matrix, so it has exactly one nonzero eigenvalue and all other ones are zero. Exhibiting the eigenpair $(\mathbb{1}_k\mathbb{1}_k^T)\mathbb{1}_k = k\mathbb{1}_k$ shows that its single nonzero eigenvalue is its trace k, with multiplicity 1. Since the other matrix is a multiple of the identity, it follows that $\frac{1}{k}\mu(A_k)$ consists of k eigenvalues with value 1, and one that is 0. Therefore, the smoothness constant is $L = 1$.

Now we prove the PL property. Since $f(x)$ is convex and differentiable over the compact set $\mathcal{P} \subset \mathbb{R}^{nk}$, we have that Ω^* is nonempty and a minimum exists. Next, we observe that

$$A_k^2 = (kI_k - \mathbb{1}_k\mathbb{1}_k^T)^2 = k^2 I_k + k\mathbb{1}_k\mathbb{1}_k^T - 2k\mathbb{1}_k\mathbb{1}_k^T = k(kI_k - \mathbb{1}_k\mathbb{1}_k^T) = kA_k\,,$$

so that $M_k^2 = (A_k \otimes I_n)^2 = A_k^2 \otimes I_n = k(A_k \otimes I_n) = kM_k$. Therefore, $\|\nabla f(x)\|^2 = \frac{1}{k^2}\|M_k x\|^2 = \frac{1}{k^2}\langle M_k^2 x, x\rangle = \frac{1}{k}\langle M_k x, x\rangle = 2f(x)$. We then verify that f is 1-PL, because $\frac{1}{2}\|\nabla f(x)\|^2 = f(x) \geq (f(x) - f^*)$ since the minimum f^* is clearly nonnegative. ■

A.4 Linear Rates in the Product in Terms of the Pyramidal Width

The goal of this section, and of the following Section A.5, is to quantify linear convergence rates of the AFW method for optimizing PL objectives over product polytopes. In this work, we achieve this by applying the novel bounds we found on the pyramidal width and the vertex-facet distance of the product polytope, presented in Theorem 4.5 and Proposition 4.7.

Linear convergence of certain FW variants using Hölder error bounds (HEB), or *sharpness* conditions, which generalize the PL condition, is known and has been established in previous works (see, e.g., (Braun, Carderera, et al., 2022; Kerdreux, d'Aspremont, and Pokutta, 2021)). These results leverage a combination of the scaling inequality from Theorem A.4, based on a positive condition number of the polytope, and an HEB instead of strong convexity. Our convergence results in Lemma A.5 and Proposition 4.8 follow ideas similar to those in, say, (Braun, Carderera, et al., 2022, Theorem 2.29, Lemma 3.32, Corollary 3.33). For completeness, here we include the full analysis that accounts for the specific quantities in our setting and yield slightly different convergence rates.

Given a_t, w_t defined at lines 3-5 of Algorithm 1, we first recall (Lacoste-Julien and Jaggi, 2015, Theorem 3), that we write conveniently with our notation:

Theorem A.4 ((Lacoste-Julien and Jaggi, 2015, Theorem 3)). *Let $\mathcal{P}$ be a polytope, $x_t \in \mathcal{P}$ be a suboptimal point and $\mathcal{A}_t$ be an active set for x_t. Let x^* be an optimal point and corresponding error direction $\hat{e}_t = \frac{x_t - x^*}{\|x_t - x^*\|}$, and gradient $\nabla f(x_t)$ (and so $\langle \nabla f(x_t), \hat{e}_t \rangle > 0$). Let $\tilde{d}_t = a_t - w_t$ be the pairwise FW direction obtained over $\mathcal{A}_t$ and $\mathcal{P}$, respectively, with gradient $\nabla f(x_t)$. Then $\frac{\langle \nabla f(x_t), \tilde{d}_t \rangle}{\langle \nabla f(x_t), \hat{e}_t \rangle} \geq \delta_{\mathcal{X}}$*

To quantify rates of AFW for solving Eq. (4) and, more generally, for optimizing a PL objective over a product polytope, we rely on the following lemmas:

Lemma A.5 ((Braun, Carderera, et al., 2022, Lemma 3.32)). *Let $\mathcal{P}$ be a polytope with pyramidal width $\delta_{\mathcal{P}} > 0$ and let f be a convex and μ-PL function. Then, it holds:*

$$\langle \nabla f(x_t), a_t - w_t \rangle^2 \geq \frac{\delta_{\mathcal{P}}^2 \, \mu}{2} h_t \,, \tag{8}$$

with $a_t \in \arg\max_{v \in \mathcal{A}_t} \langle \nabla f(x_t), x \rangle$, $w_t \in \arg\min_{v \in \mathcal{P}} \langle \nabla f(x_t), x \rangle$ and $h_t \stackrel{\text{def}}{=} f(x_t) - f(x^*)$.

Proof In order to prove the lemma, we rely on the notion of quadratic growth (QG), which is known to be equivalent to the PL condition under convexity, see Definition 3.2. This holds, in particular, in the case of our objective in Eq. (4). We then get the following inequalities:

$$\begin{aligned} \langle \nabla f(x_t), a_t - w_t \rangle^2 &\geq \delta_{\mathcal{X}}^2 \tfrac{\langle \nabla f(x_t), x_t - x^* \rangle^2}{\|x_t - x^*\|^2} \\ &\geq \delta_{\mathcal{X}}^2 \tfrac{h_t^2}{\|x_t - x^*\|^2} \\ &\geq \delta_{\mathcal{X}}^2 \tfrac{h_t}{\|x_t - x^*\|^2} \tfrac{\mu}{2} \|x_t - x^*\|^2 \\ &= \tfrac{\delta_{\mathcal{X}}^2 \mu}{2} h_t \,, \end{aligned} \tag{9}$$

the first one by Theorem A.4 and picking $x^* \in \arg\min_{z \in \Omega^*} \|x_t - z\|^2$, the second one by convexity of f, and the last one because f is μ-QG. ∎

Lemma A.6. *For each step t of Algorithm 1, the following holds:*

$$2 \langle \nabla f(x_t), d_t \rangle \geq \langle \nabla f(x_t), a_t - w_t \rangle \,,$$

with $w_t \in \arg\min_{v \in \mathcal{P}} \langle \nabla f(x_t), x \rangle$.

Proof If Algorithm 1 takes a FW step (Line 6), $d_t = x_t - w_t$ and $\langle \nabla f(x_t), x_t - w_t \rangle \geq \langle \nabla f(x_t), a_t - x_t \rangle$, and then $\langle \nabla f(x_t), 2(x_t - w_t) \rangle \geq \langle \nabla f(x_t), x_t - w_t + a_t - x_t \rangle$. If, instead, Algorithm 1 takes an away step (Line 11), $\langle \nabla f(x_t), a_t - x_t \rangle \geq \langle \nabla f(x_t), x_t - w_t \rangle$, and then $\langle \nabla f(x_t), 2(a_t - x_t) \rangle \geq \langle \nabla f(x_t), a_t - x_t + x_t - w_t \rangle$. ∎

Lemma A.7 ((Braun, Carderera, et al., 2022, Lemma 1.5)). *Let f be an L-smooth function, and $y \stackrel{\text{def}}{=} x - \lambda d$, where d is an arbitrary vector (i.e., a direction) and $\lambda > 0$. Then, if $y \in \mathsf{dom} f$;*

$$f(x) - f(y) \geq \lambda \frac{\langle \nabla f(x), d \rangle}{2} \qquad \text{for } 0 \leq \lambda \leq \frac{\langle \nabla f(x), d \rangle}{L \|d\|^2} \,.$$

Proof of Proposition 4.8. In the first part of the proof, we show linear convergence of Algorithm 1 over $\mathcal{P}$ under the PL inequality. Our proof adapts the one of (Braun, Carderera, et al., 2022, Theorem 2.29), by exploiting the PL condition rather than strong convexity. This is a routinary analysis and not one of the main contribution of our work. However, after finishing this analysis, we can use the values of our condition numbers in the product polytope to obtain a quantified linear convergence rate to see the impact of our study.

By Lemma A.5 and Lemma A.6, it holds:

$$h_t \leq \frac{2 \langle \nabla f(x_t), a_t - w_t \rangle^2}{\delta_{\mathcal{P}}^2 \, \mu} \leq \frac{8 \langle \nabla f(x_t), d_t \rangle^2}{\delta_{\mathcal{P}}^2 \, \mu}. \tag{10}$$

for $h_t \stackrel{\text{def}}{=} f(x_t) - f(x^*)$. For every step of Algorithm 1, by first applying convexity and then either a FW step or an away step, we have

$$h_t \leq \langle \nabla f(x_t), x_t - w_t \rangle \leq \langle \nabla f(x_t), d_t \rangle \,, \tag{11}$$

where we first applied convexity of f and linear minimization, and then either a FW step or an away step. Recall line 14 of Algorithm 1: $\lambda_t \stackrel{\text{def}}{=} \min\left\{\Lambda_t, \ \frac{\langle \nabla f(x_t), d_t \rangle}{L \, \|d_t\|^2}\right\}$, it holds:

$$\begin{aligned} f(x_t) - f(x_{t+1}) &= h_t - h_{t+1} \\ &\geq \frac{\langle \nabla f(x_t), d_t \rangle}{2} \min\left\{\Lambda_t, \frac{\langle \nabla f(x_t), d_t \rangle}{L\|d_t\|^2}\right\} \quad \text{[Lemma A.7, line 14 Algorithm 1]} \\ &\geq \min\left\{\Lambda_t \frac{\langle \nabla f(x_t), d_t \rangle}{2}, \frac{\langle \nabla f(x_t), d_t \rangle^2}{2LD_{\mathcal{P}}^2}\right\} \\ &\geq \min\left\{\Lambda_t \frac{h_t}{2}, \frac{\delta_{\mathcal{P}}^2 \, \mu}{16LD_{\mathcal{P}}^2} h_t\right\} , \qquad \text{[Eq. (10) and Eq. (11)]} \end{aligned} \tag{12}$$

so

$$h_{t+1} \leq \left(1 - \min\left\{\frac{\Lambda_t}{2}, \frac{\delta_{\mathcal{P}}^2 \, \mu}{16LD_{\mathcal{P}}^2}\right\}\right) h_t \,. \tag{13}$$

We now proceed to show $h_{t+1} \leq \left(1 - \frac{\mu \, \delta_{\mathcal{P}}^2}{16LD_{\mathcal{P}}^2}\right) h_t$ for any steps and at least half of the iterations, making sure that Λ_t is bounded away from 0 to get linear rates.

Recall line 12 of Algorithm 1: $\lambda_t \stackrel{\text{def}}{=} \min\left\{\Lambda_t, \ \frac{\langle \nabla f(x_t), d_t \rangle}{L \, \|d_t\|^2}\right\}$. We now proceed to show that $h_{t+1} \leq \left(1 - \frac{\mu \, \delta_{\mathcal{P}}^2}{8LD_{\mathcal{P}}^2}\right) h_t$ for both FW and away steps, for at least half of the iterations.

For both types of steps, if $\lambda_t < \Lambda_t$, then the minimum in the first line of Eq. (12) is achieved by the second term, which immediately yields the desired bound.

Next, we consider the cases where $\lambda_t = \Lambda_t$. For FW steps, AFW sets $\Lambda_t = 1$ (see line 8 of Algorithm 1), hence the bound follows by applying Eq. (13). In fact, since $\mu \leq L$ and $\delta_{\mathcal{P}} \leq D_{\mathcal{P}}$, we have $\frac{\mu \, \delta_{\mathcal{P}}^2}{8L \, D_{\mathcal{P}}^2} \leq \frac{1}{8} \leq \frac{1}{2}$. For away steps, AFW sets $\Lambda_t = \frac{\gamma_{t,a_t}}{1-\gamma_{t,a_t}}$. When $\lambda_t = \Lambda_t$, a drop step is performed (see line 24), which discards the away vertex a_t. Since away vertices can only be removed if the active set $\mathcal{A}_t$ is nonempty, and each iteration of AFW adds at most one vertex to $\mathcal{A}_t$, drop steps can only occur in at most half of the iterations. $h_1 \leq \frac{L \, D_{\mathcal{P}}^2}{2} = C$, see also (Braun, Carderera, et al., 2022, Theorem 2.2).

To conclude the proof, the progress estimate appearing in the theorem statement follows directly from Corollary A.3. Finally, the linear rates in the form $t = O(\log\left(\frac{1}{\varepsilon}\right))$ are obtained by using $(1-x)^r \leq \exp(-rx)$, $\forall r > 0, x \in \mathbb{R}$, and solving for t the last inequality below, as follows:

$$h_t \leq C\left(1 - \frac{\mu}{16LD_{\mathcal{P}}^2}\alpha_\delta^2\right)^{\lceil (t-1)/2 \rceil} \leq C \exp\left(-\frac{\mu}{16LD_{\mathcal{P}}^2}\left\lceil \frac{t-1}{2} \right\rceil\right) \leq \varepsilon \,.$$

■

A.5 Linear Rates in the Product in Terms of the Vertex-Facet Distance

Similarly to Section A.4, here we use our bounds on the vertex-facet distance of the product polytope to quantify the linear convergence rates achieved by AFW when optimizing PL objectives over it.

Proof of Proposition 4.9.

For convergence analysis, the rates in Proposition 4.9 are quantified by instantiating (Beck and Shtern, 2015, Theorem 3.1) with the parameters of our setting. More precisely, we first identify the constants appearing in their geometric decay bound, namely Ω_X, ρ, D, N, κ, and C and then translate that decay into an explicit bound on the number of iterations t. This last step is carried out exactly as in the proof of Proposition 4.8.

Concretely, (Beck and Shtern, 2015, Theorem 3.1) yields $f(x_t) - f^* \leq C\left(1 - \alpha^\dagger\right)^{(t-1)/2}$, where

$$\alpha^\dagger = \min\left\{\frac{\Omega_X^2}{8\rho\kappa D^2 N^2}, \frac{1}{2}\right\} .$$

In our notation, Beck and Shtern's Ω_X is the vertex-facet distance $\mathsf{vf}_{\mathcal{P}}$ of the domain polytope $\mathcal{P}$, while their ρ is the smoothness constant, that we denote by L. The quantity D is the diameter $D_{\mathcal{P}}$, and N is the active-set bound discussed in our Section B; for the product polytope, one may take $N \in \{|\mathcal{V}_{\mathcal{P}}|, nk+1\}$. In (Beck and Shtern, 2015), κ is defined via their Lemmas 2.2, 2.4 and 2.5 as the optimal value of an auxiliary optimization problem, whose explicit computation may be challenging. However, we note that their Lemma 2.5 is essentially a quadratic-growth bound showing the QG inequality. Thus, for problem (4) in our setting and more generally for optimizing convex functions satisfying the PL condition over a polytope, we can determine κ in a more straightforward way than in (Beck and Shtern, 2015). Since f is μ-PL and μ-QG, see Definition 3.2 and Lemma A.5, then (Beck and Shtern, 2015, Lemma 2.5) applies directly with $\kappa = 2/\mu$ to our objective $f(x)$. Finaly, the constant C appearing in (Beck and Shtern, 2015, Theorem 3.1) can be any upper bound on the primal gap $f(x_t) - f(x^*)$, see their Lemma 2.4; in particular, we set C here to the primal gap itself, but the coarser bound $C = \frac{LD_{\mathcal{P}}^2}{2}$ would also be valid by (Braun, Carderera, et al., 2022, Theorem 2.2).

Substituting these quantities into $\alpha^\dagger$ gives

$$\alpha^\dagger = \min\left\{\frac{\mu\,\mathsf{vf}_{\mathcal{P}}^2}{16LD_{\mathcal{P}}^2N^2}, \frac{1}{2}\right\}.$$

Applying now Proposition 4.7, we obtain $\mathsf{vf}_{\mathcal{P}} = \min_{i\in[k]} \mathsf{vf}_{\mathcal{P}_i} = \alpha_{\mathsf{vf}}$, and hence the progress estimate stated in the proposition. Lastly, as in the proof of Proposition 4.8, we use $(1-x)^r \le \exp(-rx)$ for $r > 0$ and $x \in (0,1)$ to obtain

$$f(x_t) - f(x^*) \le C\exp\left(-\frac{\mu\,\alpha_{\mathsf{vf}}^2}{32LD_{\mathcal{P}}^2N^2}(t-1)\right).$$

Solving $f(x_t) - f(x^*) \le \varepsilon$ for t gives exactly the iteration complexity stated in Proposition 4.9. ■

B ALGORITHMS

The pseudocode for the FW algorithm can be found in, e.g., (Braun, Carderera, et al., 2022, Algorithm 2.1). We then present the pseudocode of the AFW algorithm.

The *active set* $\mathcal{A}_t$ of a polytope $\mathcal{P}$ at iteration t is defined as follows:

$$\mathcal{A}_t \overset{\text{def}}{=} \{v \in \mathcal{V}_{\mathcal{P}} \,:\, x_t = \sum_{v\in\mathcal{V}_{\mathcal{P}}} \gamma_v v,\ \gamma > 0,\ \sum_{v\in\mathcal{V}_{\mathcal{P}}} \gamma_v = 1\},$$

and we use $\gamma_{t,v}$ to denote the active set weight of a vertex $v \in \mathcal{A}_t$ to represent x_t.

An essential component of any FW variant is the choice of step size. One rule is the *line search*, in which at iteration t one approximately or exactly finds $\arg\min_{\lambda\in[0,\Lambda_t]} f(x_t + \lambda(v_t - x_t))$, where v_t is the LMO vertex and Λ_t maintains feasibility. Although this rule maximally decreases the current objective, it requires solving an additional subproblem and can be expensive. An alternative, appearing on line 14 of Algorithm 1, is the *short step* rule, which is obtained by minimizing a quadratic upper bound on the objective guaranteed by its smoothness. In fact, the short step rule is a cheaper approximation of the line search one, although it comes at the expense of knowing or estimating the smoothness constant. Both step size rules ensure monotonic decrease of f (Braun, Carderera, et al., 2022), leading to linear rates under standard assumptions. In fact, any convergence rate proved for the short step variant carries over to the line search version, which makes larger progress per iteration, while the short step remains the preferred choice for clean theoretical analysis due to its simple form.

We note that Algorithm 1 incorporates a procedure $\mathcal{R}$, described at lines 16-25, which refines the active set $\mathcal{A}_t$ and the active weight vector λ_t at each iteration. In particular, after line 21 of the algorithm, $\mathcal{R}$ updates the representation of the current iterate x_t to a more efficient one, using a reduced active set $\tilde{\mathcal{A}}_t \subseteq \mathcal{A}_t$ with $|\tilde{\mathcal{A}}_t| \le |\mathcal{V}_{\mathcal{P}}|$ as in Algorithm 1. We note that $\mathcal{R}$ can also be designed to implement the Carathéodory theorem, in which case $|\tilde{\mathcal{A}}_t| \le n+1$, where n is the dimension of $\mathcal{P}$, cf. (Beck and Shtern, 2015, Appendix A), (Besançon, Pokutta, and Wirth, 2025) and references therein. This variant is advantageous when the number of vertices of $\mathcal{P}$ is significantly larger than its dimension. The notation N employed in the proof of Proposition 4.9 refers to these two upper bounds on $|\tilde{\mathcal{A}}_t|$, i.e., since the product polytope has dimension nk, $N \in \{|\mathcal{V}_{\mathcal{P}}|, nk+1\}$.

Algorithm 1: Away Frank–Wolfe (Braun, Carderera, et al., 2022, Algorithm 2.2)

Data: Convex and smooth function f, start vertex $x_0 \in \mathcal{V}_{\mathcal{P}}$, LMO over $\mathcal{P}$

Initialize active set: $\mathcal{A}_0 \leftarrow \{x_0\}$
Initialize active set weights: $\gamma_{0,x_0} \leftarrow 1$
for $t = 0, \dots, T-1$ **do**
  $w_t \leftarrow \arg\min_{v \in \mathcal{P}} \langle \nabla f(x_t), v \rangle$; ▷ FW vertex
  $a_t \leftarrow \arg\max_{v \in \mathcal{A}_t} \langle \nabla f(x_t), v \rangle$; ▷ away vertex in $\mathcal{A}_t$
  if $\langle \nabla f(x_t), x_t - w_t \rangle \geq \langle \nabla f(x_t), a_t - x_t \rangle$ **then**
    $d_t \leftarrow x_t - w_t$; ▷ FW step
    $\Lambda_t \leftarrow 1$
  end
  else
    $d_t \leftarrow a_t - x_t$; ▷ away step
    $\Lambda_t \leftarrow \frac{\gamma_{t,a_t}}{1-\gamma_{t,a_t}}$
  end
  $\lambda_t \leftarrow \min\left\{\Lambda_t, \frac{\langle \nabla f(x_t), d_t \rangle}{L\|d_t\|^2}\right\}$
  $x_{t+1} \leftarrow x_t - \lambda_t d_t$
  if $\langle \nabla f(x_t), x_t - w_t \rangle \geq \langle \nabla f(x_t), a_t - x_t \rangle$ **then**
    $\gamma_{t+1,v} \leftarrow (1-\lambda_t)\gamma_{t,v} \quad \forall v \in \mathcal{A}_t \setminus \{w_t\}$; ▷ perform $\mathcal{R}$: update $\mathcal{A}_t$ and weights
    $\gamma_{t+1,w_t} \leftarrow \begin{cases} \lambda_t & \text{if } w_t \in \mathcal{A}_t \\ (1-\lambda_t)\gamma_{t,w_t} + \lambda_t & \text{otherwise} \end{cases}$
    $\mathcal{A}_{t+1} \leftarrow \begin{cases} \mathcal{A}_t \cup \{w_t\} & \text{if } \lambda_t < 1 \\ \{w_t\} & \text{if } \lambda_t = 1 \end{cases}$
  end
  else
    $\gamma_{t+1,v} \leftarrow (1+\lambda_t)\gamma_{t,v} \quad \forall v \in \mathcal{A}_t \setminus \{a_t\}$
    $\gamma_{t+1,a_t} \leftarrow (1+\lambda_t)\gamma_{t,a_t} - \lambda_t$
    $\mathcal{A}_{t+1} \leftarrow \begin{cases} \mathcal{A}_t \setminus \{a_t\} & \text{if } \gamma_{t+1,a_t} = 0; \text{ ▷ drop step} \\ \mathcal{A}_t & \text{if } \gamma_{t+1,a_t} > 0 \end{cases}$
  end
end

Algorithm 2: Alternating linear minimization

Data: L-smooth and convex f, $(x_0, y_0) \in \mathcal{P} \times \mathcal{Q}$, $\lambda_t, \gamma_t \in [0,1]$, $t = 0, \dots, T-1$

for $t = 0$ **to** $T-1$ **do**
  $w_t \leftarrow \arg\min_{x \in \mathcal{P}} \langle \nabla_x f(x_t, y_t), x \rangle$
  $x_{t+1} \leftarrow x_t - \lambda_t(x_t - w_t)$
  $z_t \leftarrow \arg\min_{y \in \mathcal{Q}} \langle \nabla_x f(x_{t+1}, y_t), y \rangle$
  $y_{t+1} \leftarrow x_t - \gamma_t(y_t - z_t)$
end

We note that one can modify the analysis of the AFW algorithm to obtain similar convergence rates for a primal-dual computable gap, similarly as in (Martínez-Rubio and Pokutta, 2025, Appendix D.1.2), which shows the linear convergence for AFW exploiting the pyramidal width and strong convexity of the objective.

Next, we provide the pseudocode for the alternating linear minimization (ALM) algorithm proposed in (Braun, Pokutta, and Weismantel, 2022), an adaptation of the cyclic block-coordinate FW (C-BC-FW) algorithm from (Beck, Pauwels, and Sabach, 2015) to the set-intersection problem. In our computational experiments in Section 6, we use ALM as a benchmark in solving problem Eq. (4).

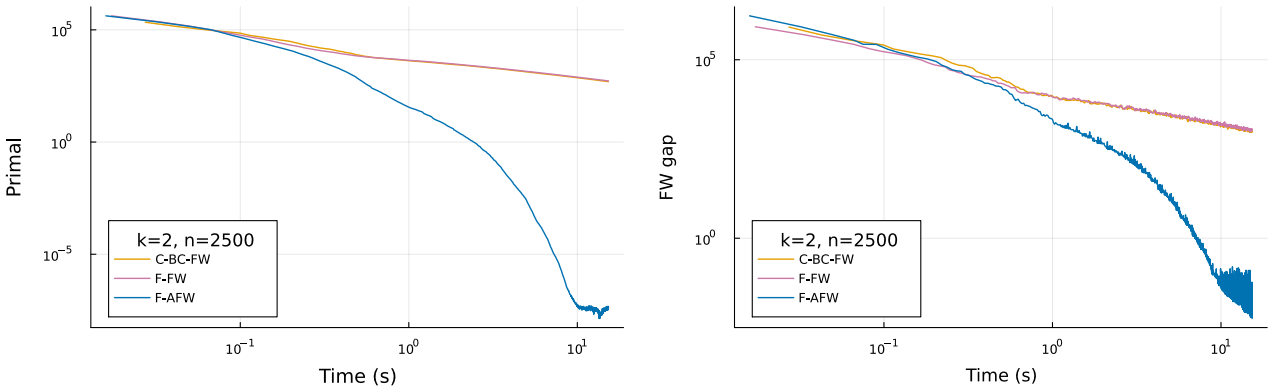

Figure 6: FW variants, run for up to 1 000 iterations on two non-intersecting point-cloud polytopes in $\mathbb{R}^{2\,500}$.

C ADDITIONAL EXPERIMENTS

In this section, we present additional plots to further validate the practical behavior of FW, C-BC-FW and AFW when solving both point-cloud and vertex-facet instances. As in Section 6, the plots below display the primal and FW gap of the examined algorithms, plotted against the elapsed computational time, in seconds, using logarithmic scales on both axes. The software and hardware settings are the same as the ones used to obtain the figures in Section 6.

The point-cloud plots below show the outcome of running the experiments of Section 6 with different configurations and seeds. In both the non-intersecting cases (Figure 6, Figure 7, Figure 8, Figure 9) and the intersecting ones (Figure 10, Figure 11), we observe the same relative ordering among the algorithms as in the main text.

In particular, AFW consistently outperforms the FW-based baselines in non-intersecting runs, clearly showing its linear convergence in the plots. Even the larger non-intersecting point-cloud instances in Figures 8, 9 and 11 to 13, where each polytope is generated from denser convex-hull constructions with a large amount of points (up to an order of magnitude larger than the example in Figure 1), make the linear regime of AFW clearly visible. Even in these cases, AFW continues to outperform the other methods, underscoring its robustness even when individual LMO calls become substantially more expensive. In fact, we note that, for each polytope, we always compute LMOs that perform a single linear pass over the vertex list that was used to generate it, evaluating one dot product per vertex. The same implementation was used in Section 6. Because of this, the time to convergence in these examples is longer, in absolute terms, than on instances with polytopes generated from a list of between n and $2n$ vertices (see Section 6). Clearly, however, the relative performance of all the tested algorithms is virtually unchanged.

At the same time, the intersecting point-cloud cases still display a smaller empirical separation among the methods, in agreement with the discussion in Section 6. Overall, these plots confirm that the runs presented in Section 6 are representative of algorithmic behavior and not statistical outliers.

We also include new vertex-facet experiments in Figures 14 to 16. Figure 14 complements the $n = 10\,000$ intersecting vertex-facet instance from the main text by showing the corresponding non-intersecting construction. Figures 15 and 16 further extend this family to dimension 20 000, in both the non-intersecting and intersecting settings. Across these additional vertex-facet plots, AFW again exhibits the clearest linear regime and maintains a marked advantage over the FW-based baselines, which is consistent with the sharper local geometry induced by the vertex-facet construction.

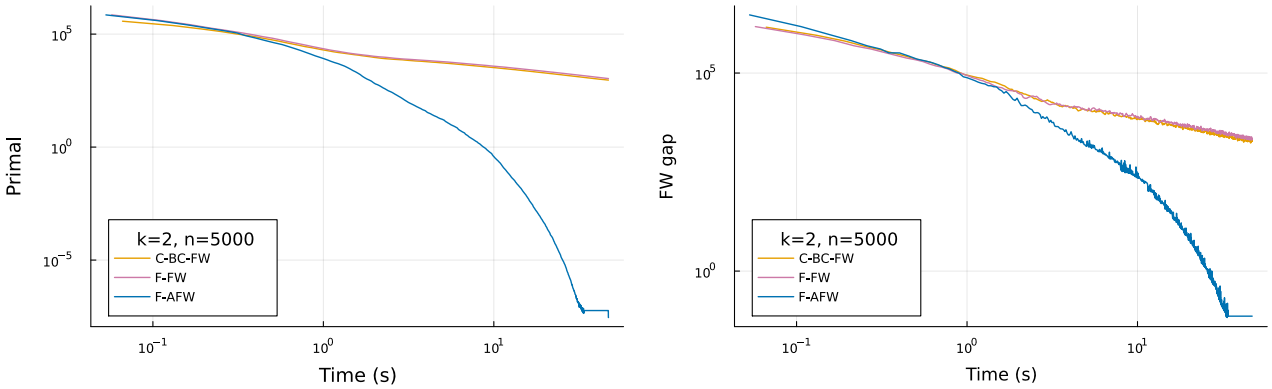

Figure 7: FW variants, run for up to 1 000 iterations on two non-intersecting point-cloud polytopes in $\mathbb{R}^{5\,000}$.

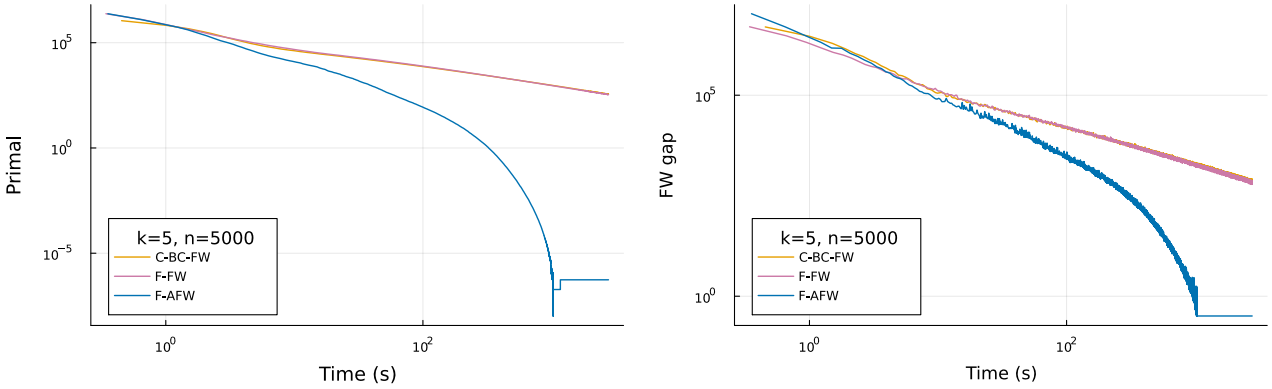

Figure 8: FW variants, run for up to 10 000 iterations on five non-intersecting point-cloud polytopes in $\mathbb{R}^{5\,000}$. For each polytope, we draw the number of points uniformly at random in $[n, 5n]$.

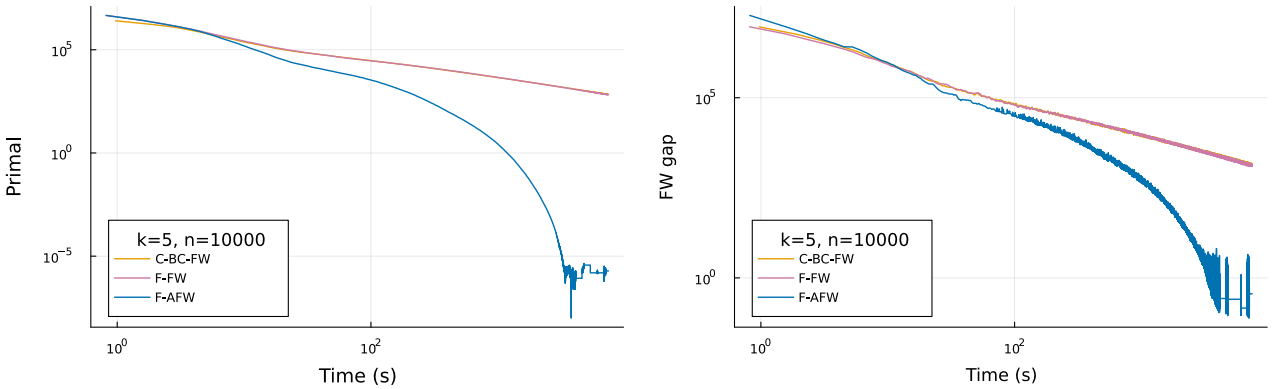

Figure 9: FW variants, run for up to 10 000 iterations on five non-intersecting point-cloud polytopes in $\mathbb{R}^{10\,000}$. For each polytope, we draw the number of points uniformly at random in $[n, 5n]$.

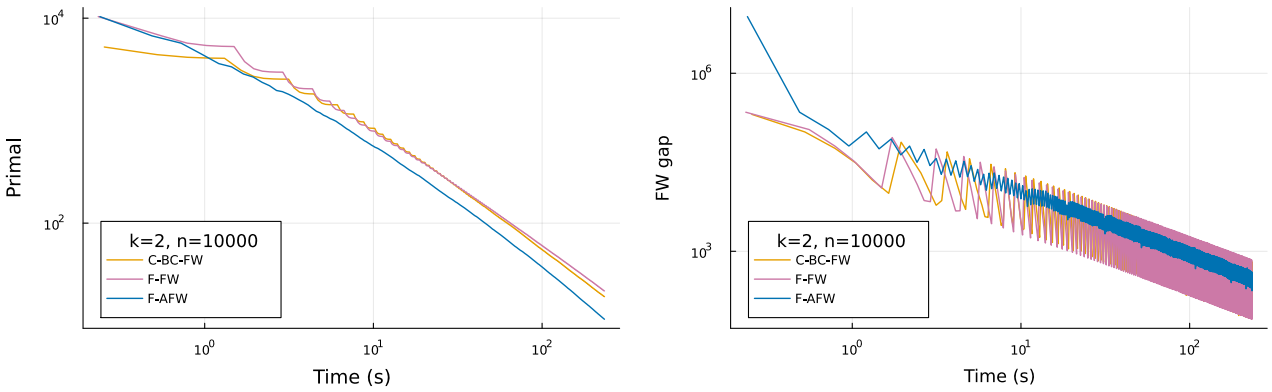

Figure 10: FW variants, run for up to 1 000 iterations on two intersecting point-cloud polytopes in $\mathbb{R}^{10\,000}$.

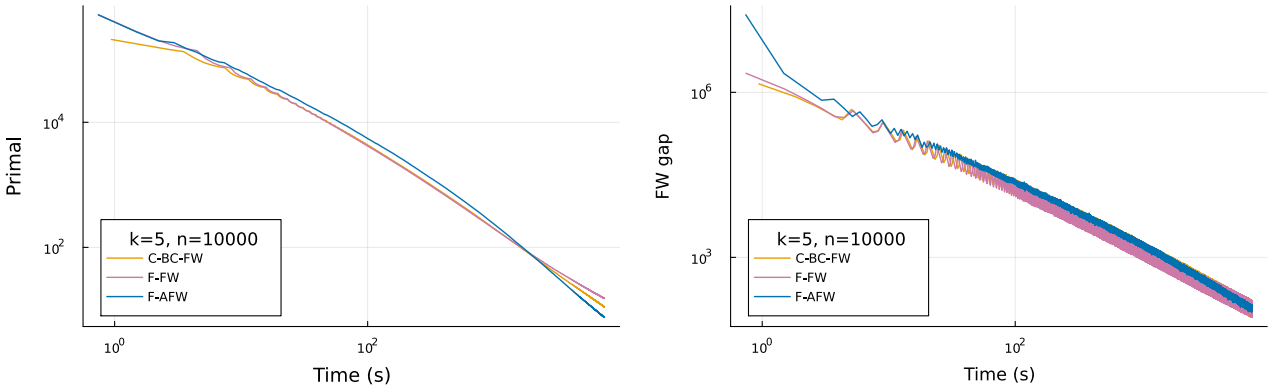

Figure 11: FW variants, run for up to 10 000 iterations on the five intersecting point-cloud polytopes from Figure 9, here translated to intersect. For each polytope, we draw the number of points uniformly at random in $[n, 5n]$.

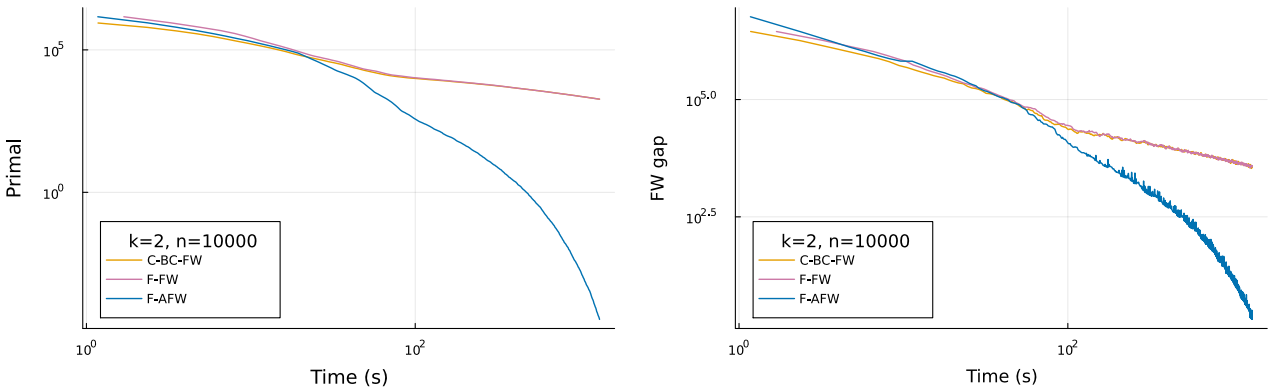

Figure 12: FW variants, run for up to 1 000 iterations on two non-intersecting point-cloud polytopes in $\mathbb{R}^{10\,000}$, generated as the convex hull of 100 000 vertices.

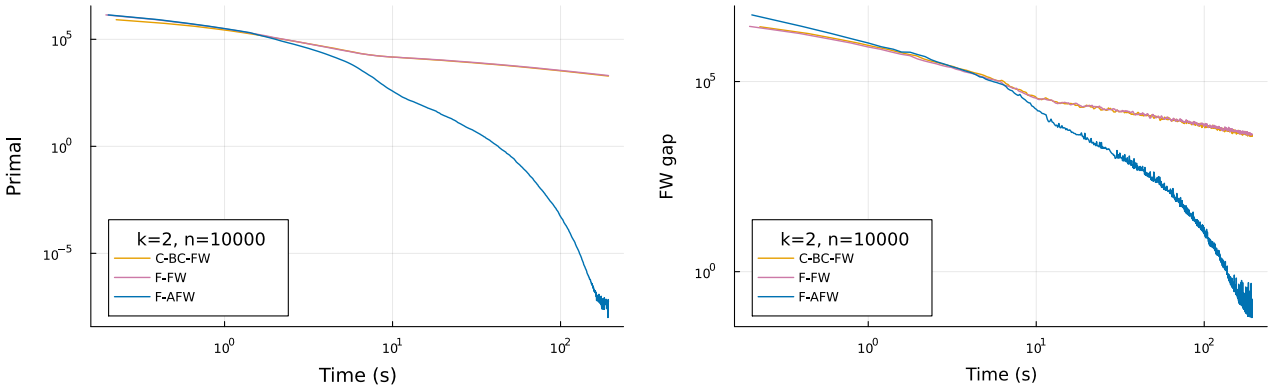

Figure 13: FW variants, run for up to 1 000 iterations on two non-intersecting point-cloud polytopes in $\mathbb{R}^{10\,000}$, generated as the convex hull of 15 000 vertices.

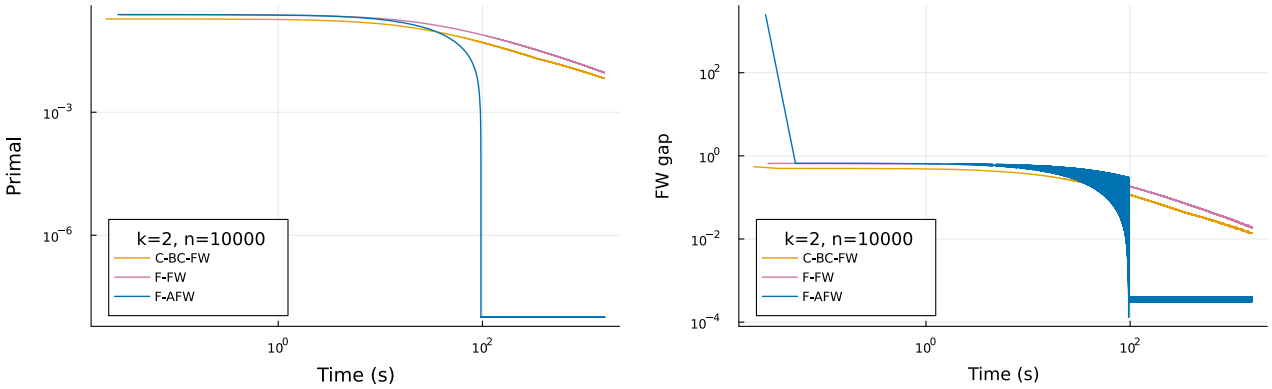

Figure 14: FW variants, run for up to 80 000 iterations on the two vertex-facet polytopes from Figure 3, not intersecting here.

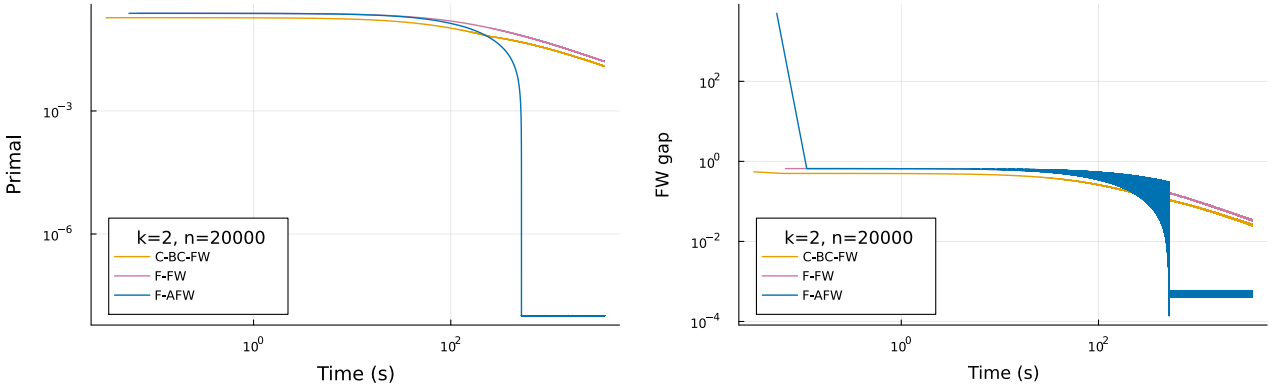

Figure 15: FW variants, run for up to 80 000 iterations on two non-intersecting vertex-facet polytopes in $\mathbb{R}^{20\,000}$.

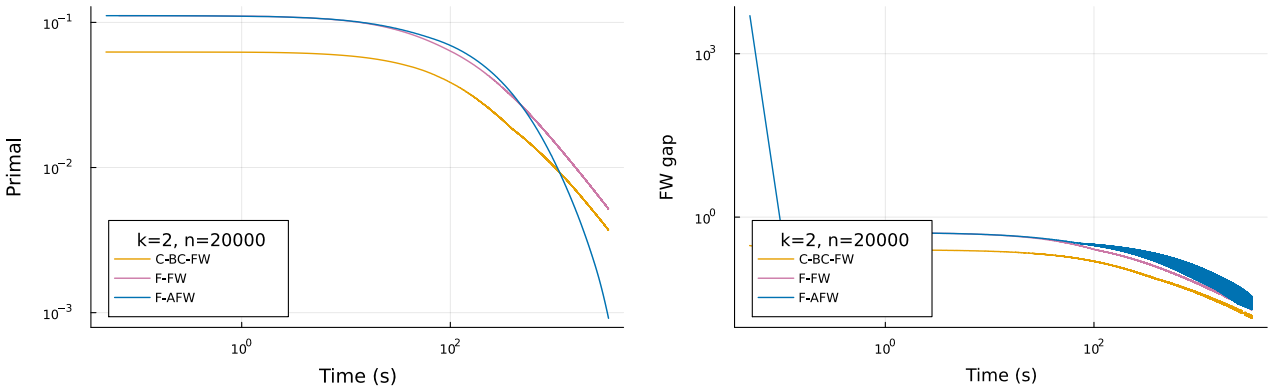

Figure 16: FW variants, run for up to 80 000 iterations on the two vertex-facet polytopes from Figure 15, here translated to intersect.