# Frank-Wolfe Methods for Optimization and Machine Learning

Cyrille W. Combettes

School of Industrial and Systems Engineering
Georgia Institute of Technology

April 16, 2021

# Outline

# Introduction

We consider

$$\begin{aligned} &\min f(x) \\ &\text{s.t. } x \in \mathcal{C} \end{aligned}$$

where

- $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set
- $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth convex function

# Introduction

We consider

$$\begin{aligned} &\min f(x) \\ &\text{s.t. } x \in \mathcal{C} \end{aligned}$$

where

- $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set
- $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth convex function

## Example

- Sparse logistic regression

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} \ln(1 + \exp(-y_i \langle a_i, x \rangle)) \\ &\text{s.t. } \|x\|_1 \leqslant \tau \end{aligned}$$

- Low-rank matrix completion

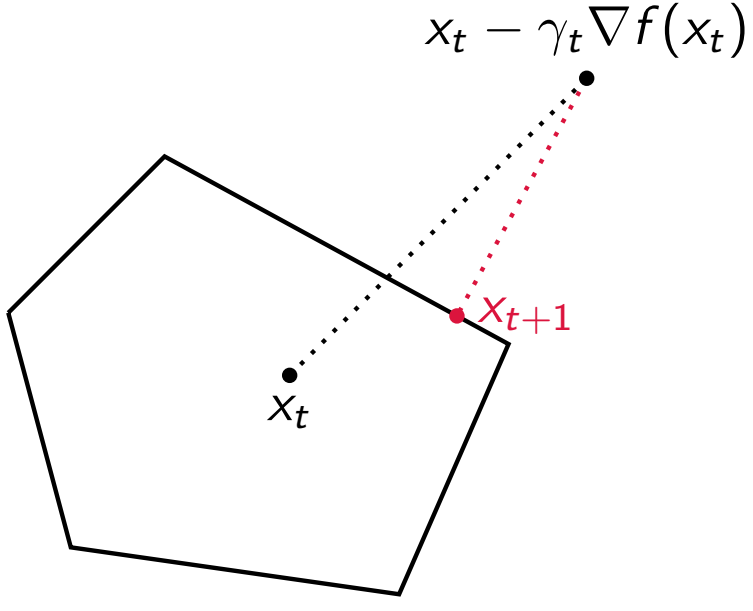
$$\begin{aligned} &\min_{X \in \mathbb{R}^{m \times n}} \frac{1}{2|\mathcal{I}|} \sum_{(i,j) \in \mathcal{I}} (Y_{i,j} - X_{i,j})^2 \\ &\text{s.t. } \|X\|_{\text{nuc}} \leqslant \tau \end{aligned}$$

# Introduction

- A natural approach is to use any efficient method and add projections back onto $\mathcal{C}$ to ensure feasibility

# Introduction

- A natural approach is to use any efficient method and add projections back onto $\mathcal{C}$ to ensure feasibility



# Introduction

- A natural approach is to use any efficient method and add projections back onto $\mathcal{C}$ to ensure feasibility
- However, in many situations projections onto $\mathcal{C}$ are very expensive

# Introduction

- A natural approach is to use any efficient method and add projections back onto $\mathcal{C}$ to ensure feasibility
- However, in many situations projections onto $\mathcal{C}$ are very expensive
- This is an issue with the method of projections, not necessarily with the geometry of $\mathcal{C}$: linear minimizations over $\mathcal{C}$ can still be relatively cheap

# Introduction

- A natural approach is to use any efficient method and add projections back onto $\mathcal{C}$ to ensure feasibility
- However, in many situations projections onto $\mathcal{C}$ are very expensive
- This is an issue with the method of projections, not necessarily with the geometry of $\mathcal{C}$: linear minimizations over $\mathcal{C}$ can still be relatively cheap
- We compare

$$\arg\min_{x \in \mathcal{C}} \langle x, y \rangle \quad \text{and} \quad \arg\min_{x \in \mathcal{C}} \|x - y\|$$

on several sets commonly used in optimization

# Complexity of linear minimization and projection

| **Set $\mathcal{C}$** | **Linear minimization** | **Projection** |
|---|---|---|
| $\ell_1/\ell_2/\ell_\infty$-ball | $\mathcal{O}(n)$ | $\mathcal{O}(n)$ |
| $\ell_p$-ball, $p \in ]1, \infty[ \setminus \{2\}$ | $\mathcal{O}(n)$ | $\mathcal{O}(n\rho^2 \|y - x^*\|_2^2/\varepsilon^2)$ |
| Nuclear norm-ball | $\mathcal{O}(\nu \ln(m+n)\sqrt{\sigma_1}/\sqrt{\varepsilon})$ | $\mathcal{O}(mn \min\{m, n\})$ |
| Flow polytope | $\mathcal{O}(m+n)$ | $\tilde{\mathcal{O}}(m^3 n + n^2)$ |
| Birkhoff polytope | $\mathcal{O}(n^3)$ | $\mathcal{O}(n^2 d_z^2/\varepsilon^2)$ |
| Permutahedron | $\mathcal{O}(n \ln(n))$ | $\mathcal{O}(n \ln(n) + n)$ |

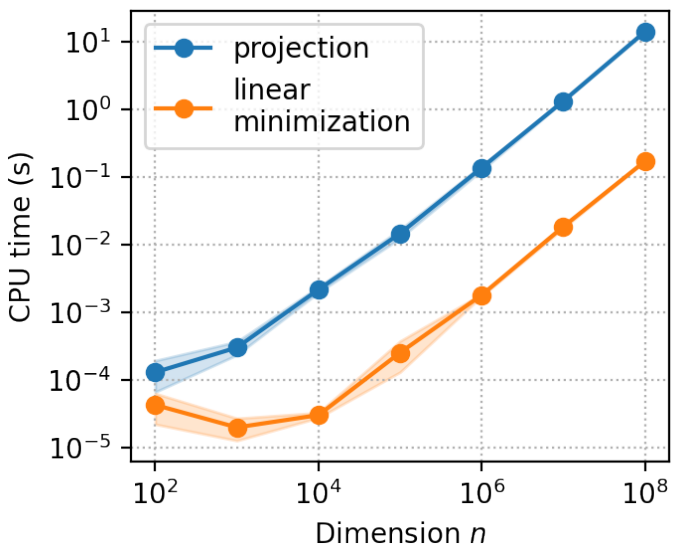
# Complexity of linear minimization and projection

| **Set $\mathcal{C}$** | **Linear minimization** | **Projection** |
|---|---|---|
| $\ell_1/\ell_2/\ell_\infty$-ball | $\mathcal{O}(n)$ | $\mathcal{O}(n)$ |
| $\ell_p$-ball, $p \in ]1, \infty[ \setminus \{2\}$ | $\mathcal{O}(n)$ | $\mathcal{O}(n\rho^2 \|y - x^*\|_2^2/\varepsilon^2)$ |
| Nuclear norm-ball | $\mathcal{O}(\nu \ln(m+n)\sqrt{\sigma_1}/\sqrt{\varepsilon})$ | $\mathcal{O}(mn \min\{m, n\})$ |
| Flow polytope | $\mathcal{O}(m+n)$ | $\tilde{\mathcal{O}}(m^3 n + n^2)$ |
| Birkhoff polytope | $\mathcal{O}(n^3)$ | $\mathcal{O}(n^2 d_z^2/\varepsilon^2)$ |
| Permutahedron | $\mathcal{O}(n \ln(n))$ | $\mathcal{O}(n \ln(n) + n)$ |

# Complexity of linear minimization and projection

| **Set $\mathcal{C}$** | **Linear minimization** | **Projection** |
|---|---|---|
| $\ell_1/\ell_2/\ell_\infty$-ball | $\mathcal{O}(n)$ | $\mathcal{O}(n)$ |
| $\ell_p$-ball, $p \in ]1, \infty[ \setminus \{2\}$ | $\mathcal{O}(n)$ | $\mathcal{O}(n\rho^2 \|y - x^*\|_2^2/\varepsilon^2)$ |
| Nuclear norm-ball | $\mathcal{O}(\nu \ln(m+n)\sqrt{\sigma_1}/\sqrt{\varepsilon})$ | $\mathcal{O}(mn \min\{m, n\})$ |
| Flow polytope | $\mathcal{O}(m+n)$ | $\tilde{\mathcal{O}}(m^3 n + n^2)$ |
| Birkhoff polytope | $\mathcal{O}(n^3)$ | $\mathcal{O}(n^2 d_z^2/\varepsilon^2)$ |
| Permutahedron | $\mathcal{O}(n \ln(n))$ | $\mathcal{O}(n \ln(n) + n)$ |

Example: the $\ell_1$-ball



# Complexity of linear minimization and projection

| **Set $\mathcal{C}$** | **Linear minimization** | **Projection** |
|---|---|---|
| $\ell_1/\ell_2/\ell_\infty$-ball | $\mathcal{O}(n)$ | $\mathcal{O}(n)$ |
| $\ell_p$-ball, $p \in ]1, \infty[ \setminus \{2\}$ | $\mathcal{O}(n)$ | $\mathcal{O}(n\rho^2 \|y - x^*\|_2^2/\varepsilon^2)$ |
| Nuclear norm-ball | $\mathcal{O}(\nu \ln(m+n)\sqrt{\sigma_1}/\sqrt{\varepsilon})$ | $\mathcal{O}(mn \min\{m, n\})$ |
| Flow polytope | $\mathcal{O}(m+n)$ | $\tilde{\mathcal{O}}(m^3 n + n^2)$ |
| Birkhoff polytope | $\mathcal{O}(n^3)$ | $\mathcal{O}(n^2 d_z^2/\varepsilon^2)$ |
| Permutahedron | $\mathcal{O}(n \ln(n))$ | $\mathcal{O}(n \ln(n) + n)$ |

# Complexity of linear minimization and projection

| **Set $\mathcal{C}$** | **Linear minimization** | **Projection** |
|---|---|---|
| $\ell_1/\ell_2/\ell_\infty$-ball | $\mathcal{O}(n)$ | $\mathcal{O}(n)$ |
| $\ell_p$-ball, $p \in ]1, \infty[ \setminus \{2\}$ | $\mathcal{O}(n)$ | $\mathcal{O}(n\rho^2 \|y - x^*\|_2^2/\varepsilon^2)$ |
| Nuclear norm-ball | $\mathcal{O}(\nu \ln(m+n)\sqrt{\sigma_1}/\sqrt{\varepsilon})$ | $\mathcal{O}(mn \min\{m, n\})$ |
| Flow polytope | $\mathcal{O}(m+n)$ | $\tilde{\mathcal{O}}(m^3 n + n^2)$ |
| Birkhoff polytope | $\mathcal{O}(n^3)$ | $\mathcal{O}(n^2 d_z^2/\varepsilon^2)$ |
| Permutahedron | $\mathcal{O}(n \ln(n))$ | $\mathcal{O}(n \ln(n) + n)$ |

# Complexity of linear minimization and projection

| **Set $\mathcal{C}$** | **Linear minimization** | **Projection** |
|---|---|---|
| $\ell_1/\ell_2/\ell_\infty$-ball | $\mathcal{O}(n)$ | $\mathcal{O}(n)$ |
| $\ell_p$-ball, $p \in ]1, \infty[ \setminus \{2\}$ | $\mathcal{O}(n)$ | $\mathcal{O}(n\rho^2 \|y - x^*\|_2^2/\varepsilon^2)$ |
| Nuclear norm-ball | $\mathcal{O}(\nu \ln(m+n)\sqrt{\sigma_1}/\sqrt{\varepsilon})$ | $\mathcal{O}(mn \min\{m, n\})$ |
| Flow polytope | $\mathcal{O}(m+n)$ | $\tilde{\mathcal{O}}(m^3 n + n^2)$ |
| Birkhoff polytope | $\mathcal{O}(n^3)$ | $\mathcal{O}(n^2 d_z^2/\varepsilon^2)$ |
| Permutahedron | $\mathcal{O}(n \ln(n))$ | $\mathcal{O}(n \ln(n) + n)$ |

- Can we avoid projections?

# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

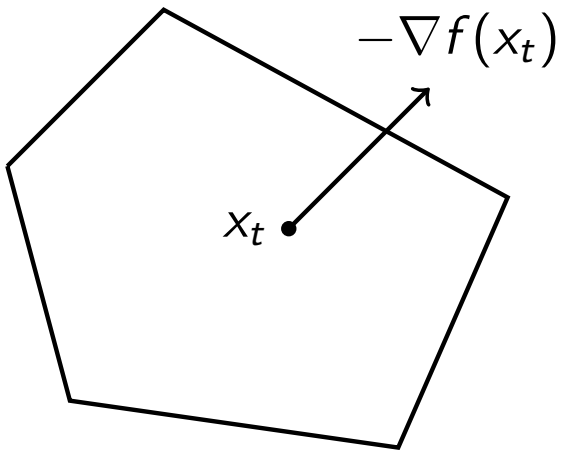
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

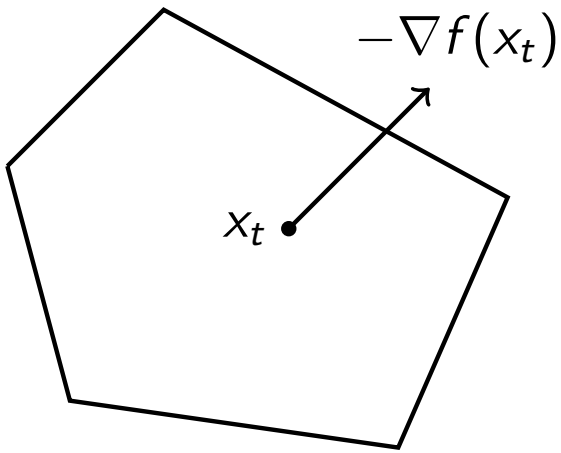
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

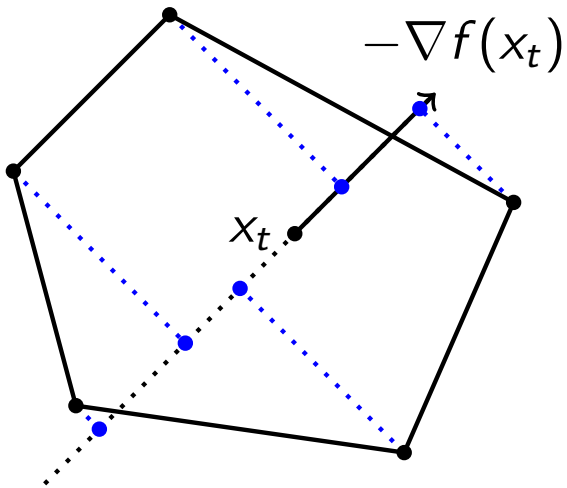
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

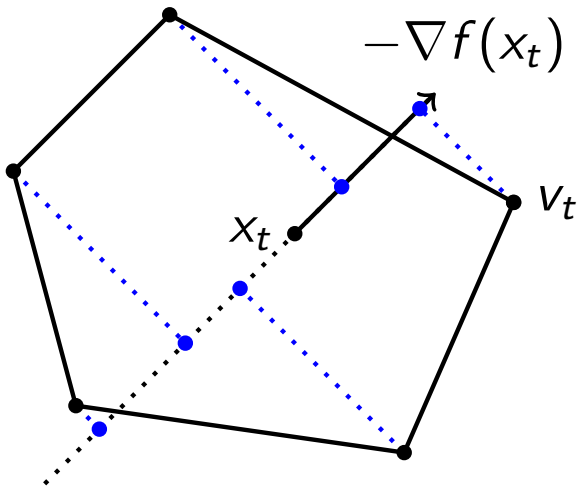
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

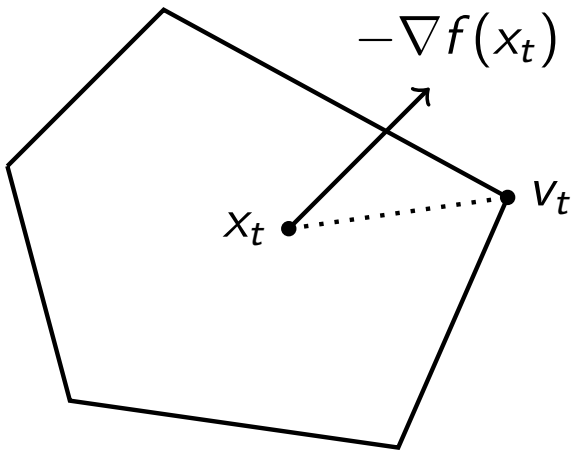
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

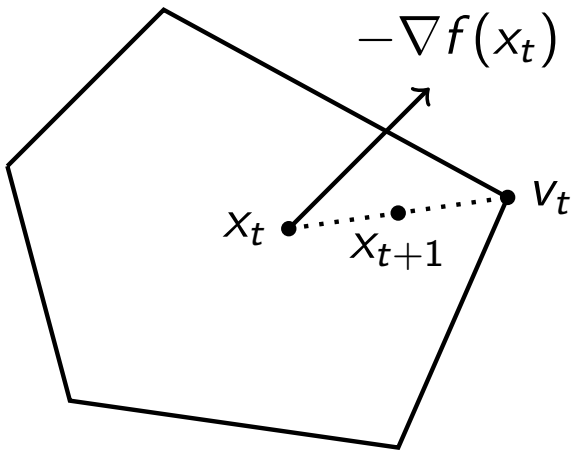
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

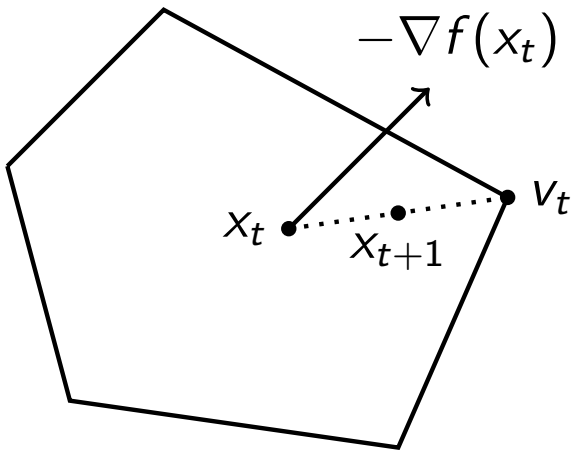
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$




- $x_{t+1}$ is obtained by convex combination of $x_t \in \mathcal{C}$ and $v_t \in \mathcal{C}$, thus $x_{t+1} \in \mathcal{C}$

# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

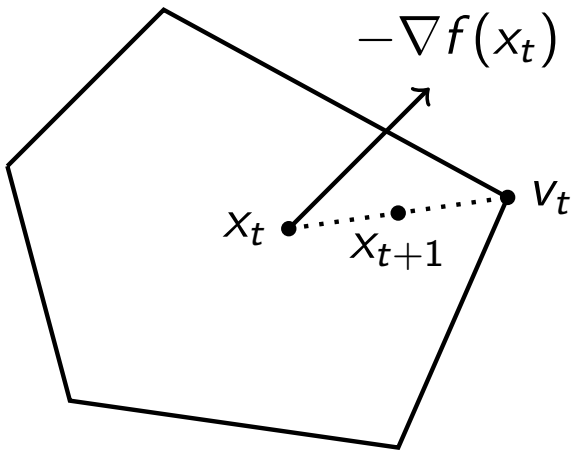
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$




- $x_{t+1}$ is obtained by convex combination of $x_t \in \mathcal{C}$ and $v_t \in \mathcal{C}$, thus $x_{t+1} \in \mathcal{C}$
- FW uses linear minimizations (the "FW oracle") instead of projections

# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

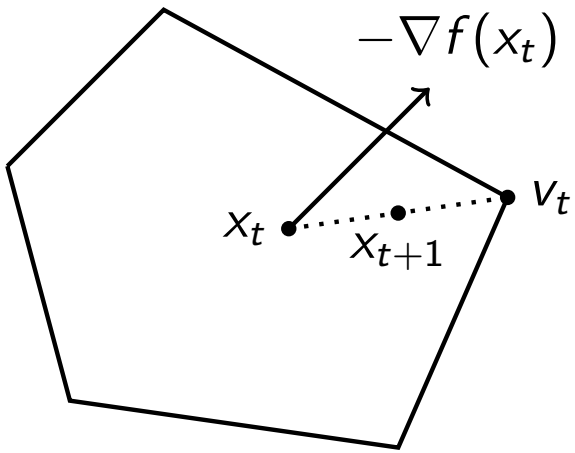
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$




- $x_{t+1}$ is obtained by convex combination of $x_t \in \mathcal{C}$ and $v_t \in \mathcal{C}$, thus $x_{t+1} \in \mathcal{C}$
- FW uses linear minimizations (the "FW oracle") instead of projections
- FW = pick a vertex (using gradient information) and move in that direction

# The Frank-Wolfe algorithm

The Frank-Wolfe algorithm (Frank & Wolfe, 1956) a.k.a. conditional gradient algorithm (Levitin & Polyak, 1966):

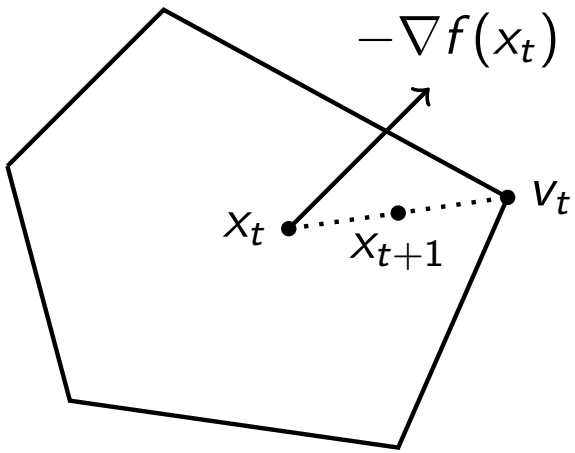
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$




- $x_{t+1}$ is obtained by convex combination of $x_t \in \mathcal{C}$ and $v_t \in \mathcal{C}$, thus $x_{t+1} \in \mathcal{C}$
- FW uses linear minimizations (the "FW oracle") instead of projections
- FW = pick a vertex (using gradient information) and move in that direction
- Applications: traffic assignment, computer vision, optimal transport, adversarial learning, etc.

# Step-size strategies

- The first strategy considered historically (Frank & Wolfe 1956; Levitin & Polyak, 1966; Demyanov & Rubinov, 1970) is

$$\gamma_t \leftarrow \min\left\{\frac{\langle x_t - v_t, \nabla f(x_t)\rangle}{L\|x_t - v_t\|^2}, 1\right\}$$

and is obtained from the smoothness upper bound:

$$\gamma_t = \underset{\gamma\in[0,1]}{\arg\min}\, f(x_t) + \gamma\langle v_t - x_t, \nabla f(x_t)\rangle + \frac{L}{2}\gamma^2\|v_t - x_t\|_2^2$$

# Step-size strategies

- The first strategy considered historically (Frank & Wolfe 1956; Levitin & Polyak, 1966; Demyanov & Rubinov, 1970) is

$$\gamma_t \leftarrow \min\left\{\frac{\langle x_t - v_t, \nabla f(x_t)\rangle}{L\|x_t - v_t\|^2}, 1\right\}$$

and is obtained from the smoothness upper bound:

$$\gamma_t = \underset{\gamma\in[0,1]}{\arg\min}\, f(x_t) + \gamma\langle v_t - x_t, \nabla f(x_t)\rangle + \frac{L}{2}\gamma^2\|v_t - x_t\|_2^2$$

# Step-size strategies

- The first strategy considered historically (Frank & Wolfe 1956; Levitin & Polyak, 1966; Demyanov & Rubinov, 1970) is

$$\gamma_t \leftarrow \min\left\{\frac{\langle x_t - v_t, \nabla f(x_t)\rangle}{L\|x_t - v_t\|^2}, 1\right\}$$

and is obtained from the smoothness upper bound:

$$\gamma_t = \underset{\gamma\in[0,1]}{\arg\min}\, f(x_t) + \gamma\langle v_t - x_t, \nabla f(x_t)\rangle + \frac{L}{2}\gamma^2\|v_t - x_t\|_2^2$$

# Step-size strategies

- The first strategy considered historically (Frank & Wolfe 1956; Levitin & Polyak, 1966; Demyanov & Rubinov, 1970) is

$$\gamma_t \leftarrow \min\left\{\frac{\langle x_t - v_t, \nabla f(x_t)\rangle}{L\|x_t - v_t\|^2}, 1\right\}$$

  and is obtained from the smoothness upper bound:

$$\gamma_t = \underset{\gamma\in[0,1]}{\arg\min}\, f(x_t) + \gamma\langle v_t - x_t, \nabla f(x_t)\rangle + \frac{L}{2}\gamma^2\|v_t - x_t\|_2^2$$

- Later on, Dunn & Harshbarger (1978) proposed *open-loop* strategies in the form $\gamma_t \sim 1/t$. The one popularized by Jaggi (2013) is

$$\gamma_t \leftarrow \frac{2}{t+2}$$

# Convergence analysis

**Theorem (Frank & Wolfe, 1956; Levitin & Polyak, 1966; Jaggi, 2013)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter $D$ and $f : \mathbb{R}^n \to \mathbb{R}$ be an $L$-smooth convex function. Then*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{4LD^2}{t+2}$$

# Convergence analysis

**Theorem (Frank & Wolfe, 1956; Levitin & Polyak, 1966; Jaggi, 2013)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter $D$ and $f : \mathbb{R}^n \to \mathbb{R}$ be an $L$-smooth convex function. Then*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{4LD^2}{t+2}$$

- The convergence rate cannot be improved (Canon & Cullum, 1968; Jaggi, 2013; Lan, 2013)

# Convergence analysis

**Theorem (Frank & Wolfe, 1956; Levitin & Polyak, 1966; Jaggi, 2013)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter $D$ and $f : \mathbb{R}^n \to \mathbb{R}$ be an $L$-smooth convex function. Then*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{4LD^2}{t+2}$$

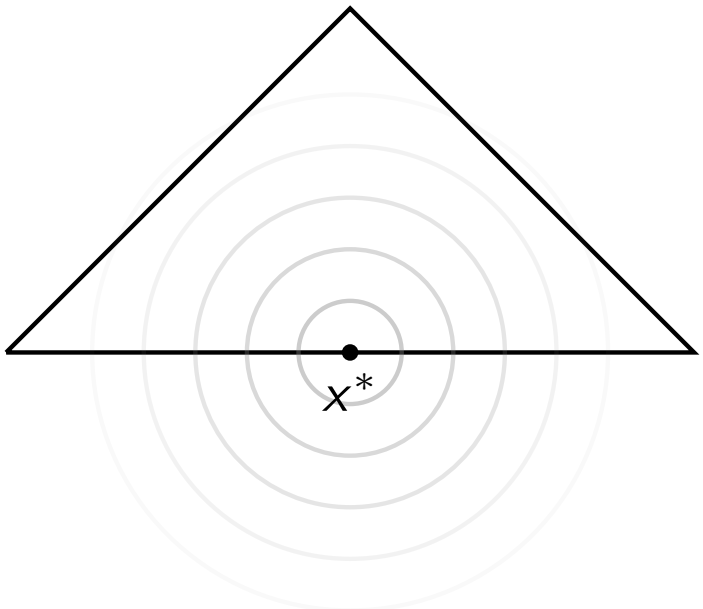
- The convergence rate cannot be improved (Canon & Cullum, 1968; Jaggi, 2013; Lan, 2013)
- The iterates can follow a naive and slow trajectory

# The zigzagging phenomenon

Consider the simple problem

$$\begin{aligned} &\min \frac{1}{2}\|x\|_2^2 \\ &\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\} \end{aligned}$$

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

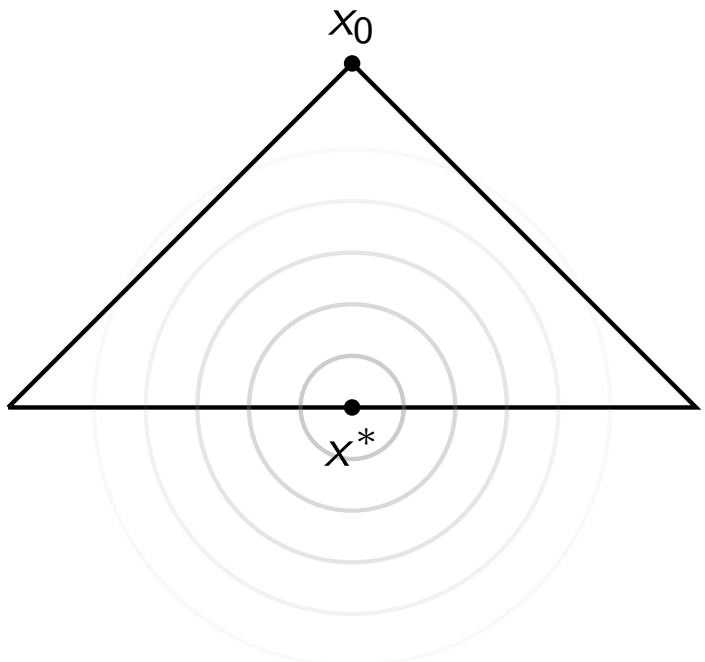


# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

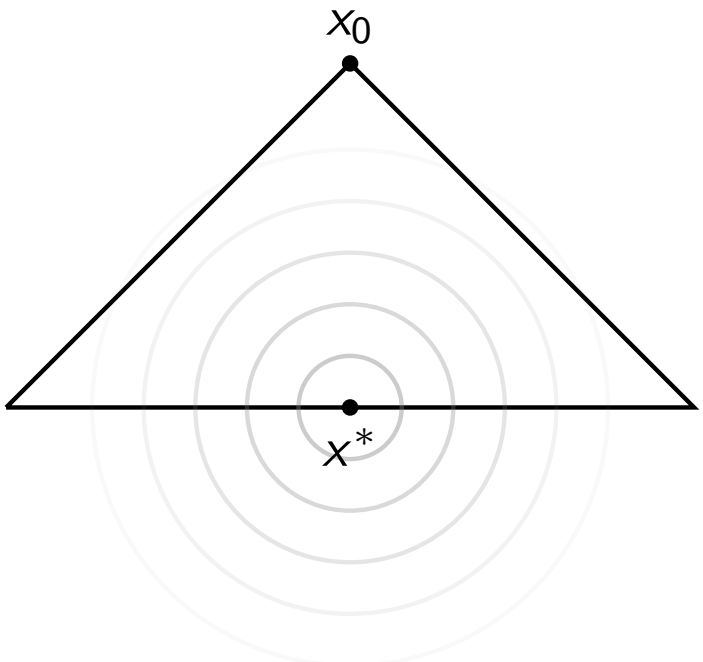



- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$



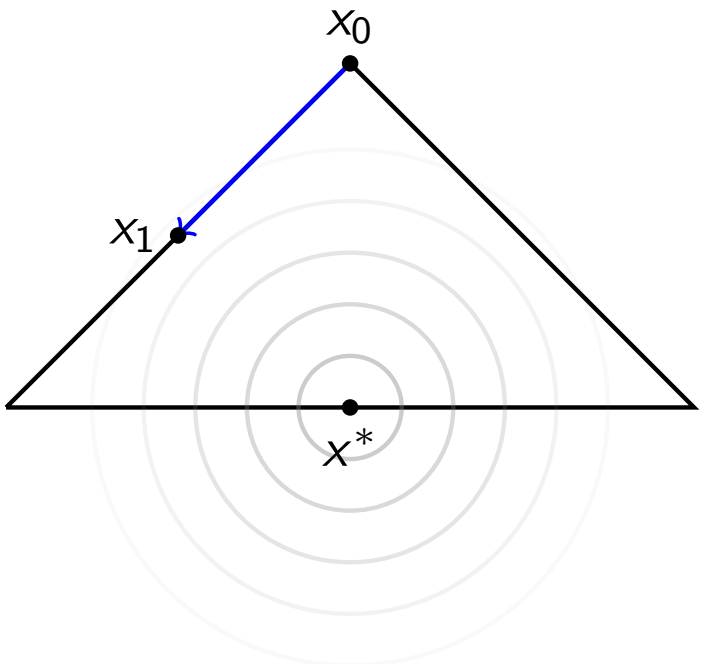

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \text{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$



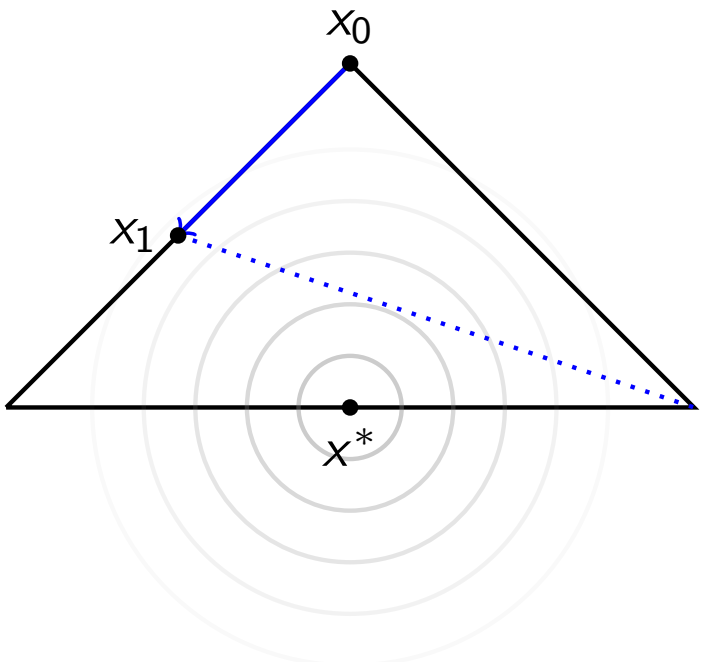

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$



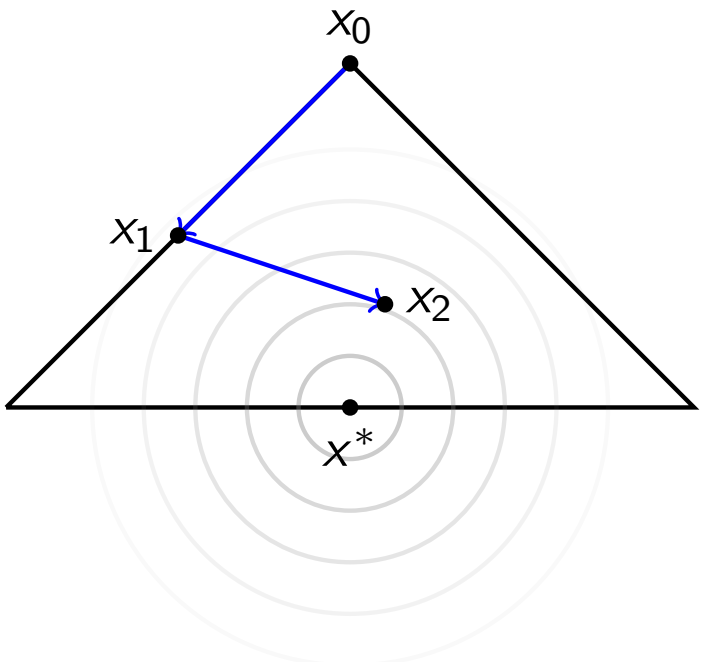

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$



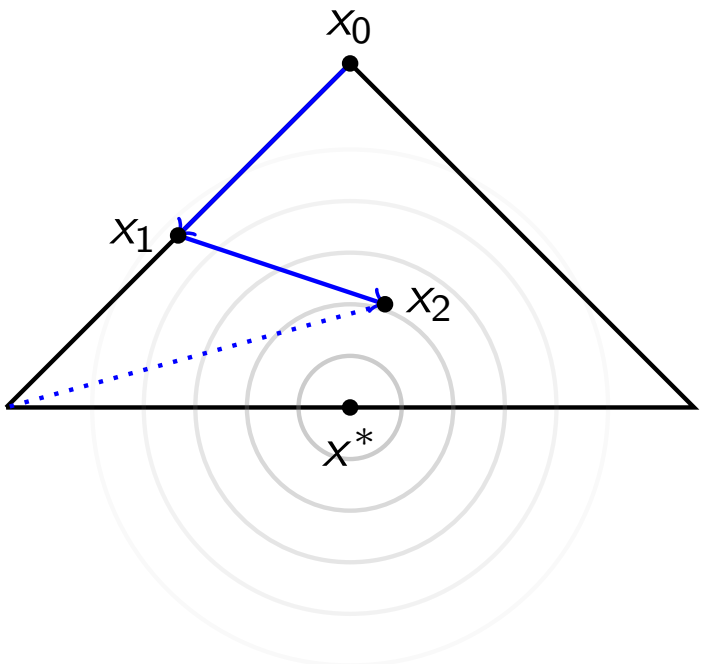

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$



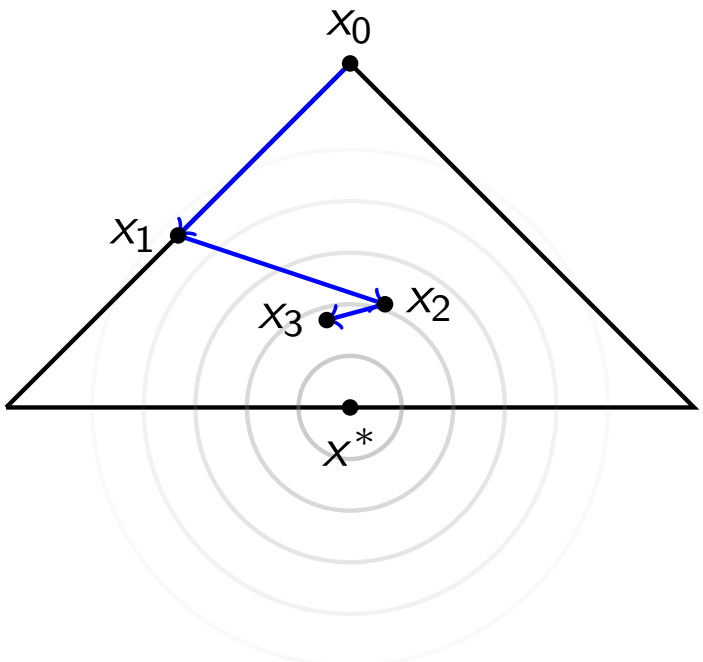

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$



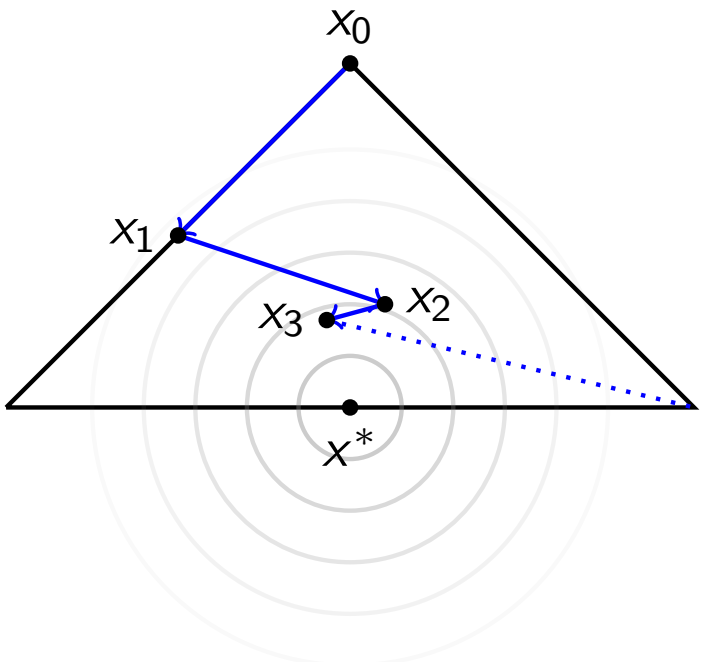

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$



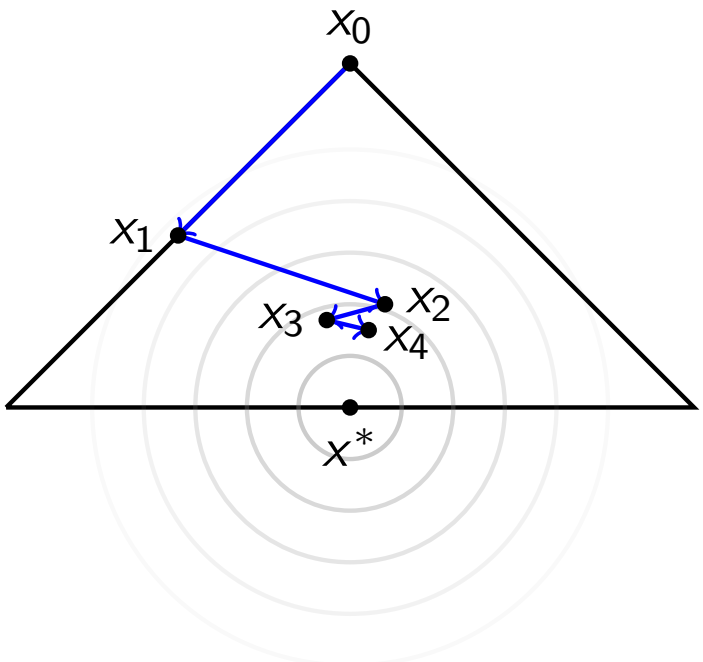

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$



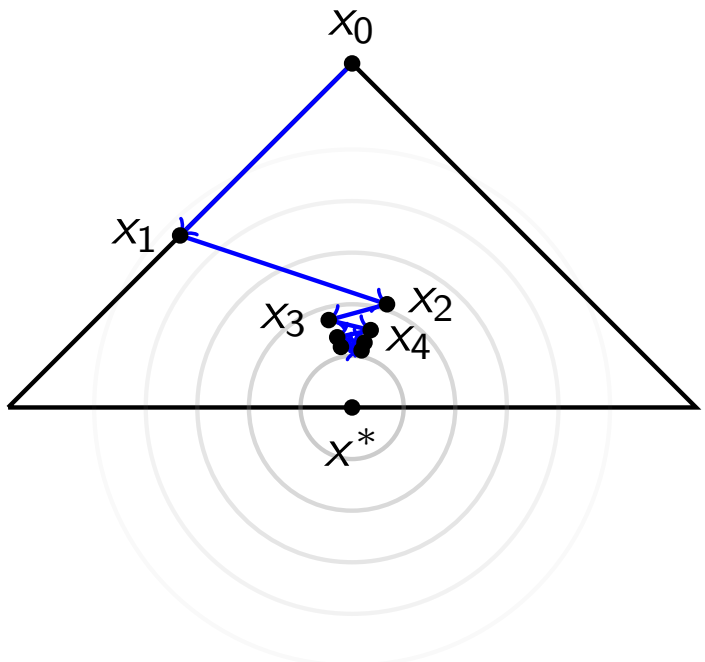

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$




and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$

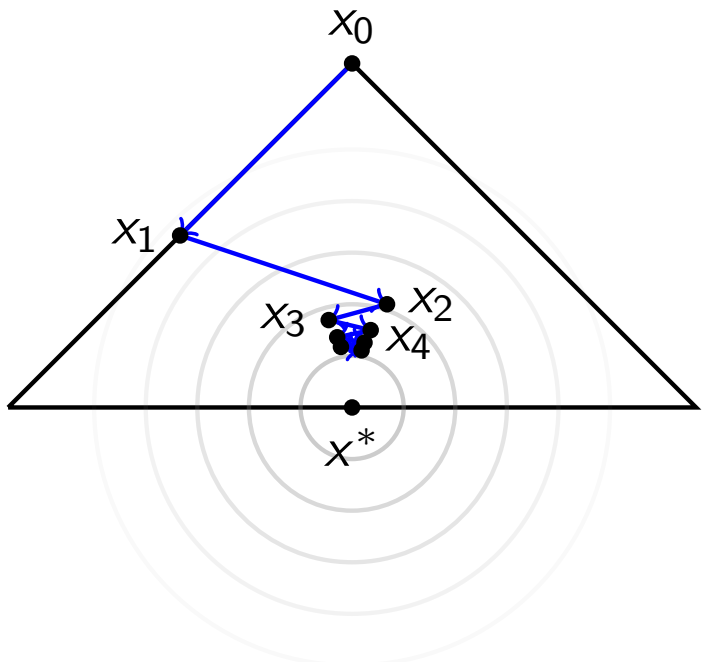
- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices

# The zigzagging phenomenon

Consider the simple problem

$$\min \frac{1}{2}\|x\|_2^2$$
$$\text{s.t. } x \in \operatorname{conv}\left\{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}-1\\0\end{pmatrix}, \begin{pmatrix}1\\0\end{pmatrix}\right\}$$

and $x^* = \begin{pmatrix}0\\0\end{pmatrix}$



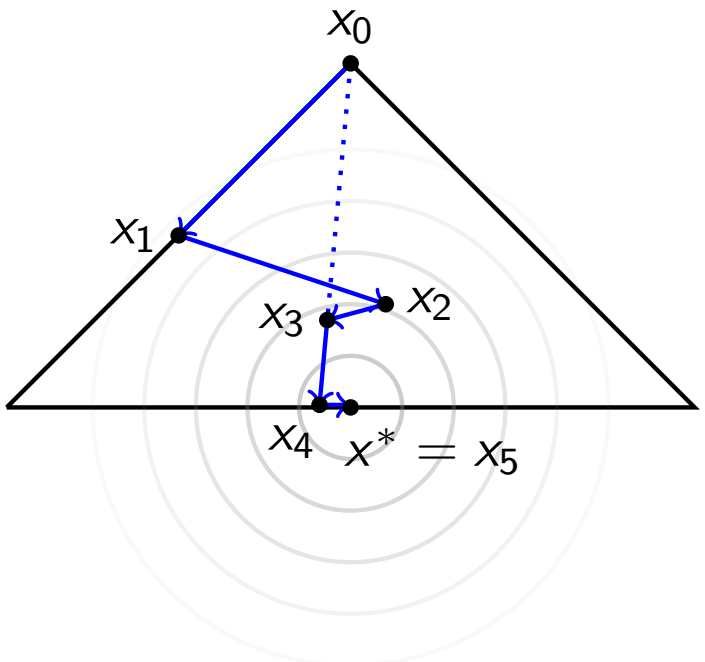

- Let $x_0 = \begin{pmatrix}0\\1\end{pmatrix}$
- FW tries to reach $x^*$ by moving towards vertices
- This yields an inefficient zig-zagging trajectory

# Faster variants of Frank-Wolfe

- Away-Step Frank-Wolfe (AFW) (Wolfe, 1970; Guélat & Marcotte, 1986; Lacoste-Julien & Jaggi, 2015): enhances FW by allowing to also move away from vertices

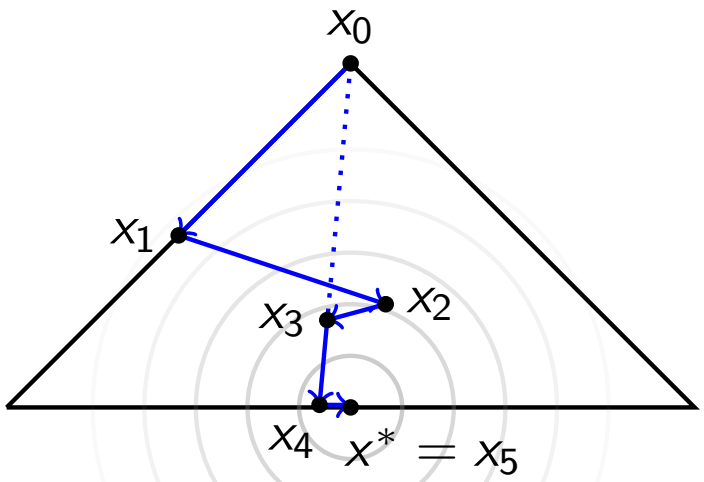
# Faster variants of Frank-Wolfe

- Away-Step Frank-Wolfe (AFW) (Wolfe, 1970; Guélat & Marcotte, 1986; Lacoste-Julien & Jaggi, 2015): enhances FW by allowing to also move away from vertices



# Faster variants of Frank-Wolfe

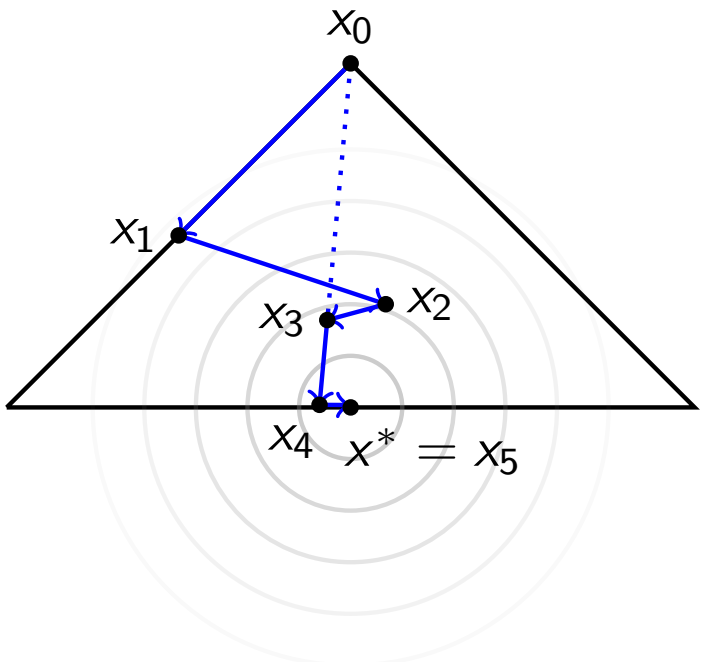
- Away-Step Frank-Wolfe (AFW) (Wolfe, 1970; Guélat & Marcotte, 1986; Lacoste-Julien & Jaggi, 2015): enhances FW by allowing to also move away from vertices




- Decomposition-Invariant Pairwise Conditional Gradient (DICG) (Garber & Meshi, 2016): memory-free variant of AFW

# Faster variants of Frank-Wolfe

- Away-Step Frank-Wolfe (AFW) (Wolfe, 1970; Guélat & Marcotte, 1986; Lacoste-Julien & Jaggi, 2015): enhances FW by allowing to also move away from vertices




- Decomposition-Invariant Pairwise Conditional Gradient (DICG) (Garber & Meshi, 2016): memory-free variant of AFW
- Blended Conditional Gradients (BCG) (Braun et al., 2019): blends FCFW and FW

# Boosting Frank-Wolfe

- Can we speed up FW in a simple way?

# Boosting Frank-Wolfe

- Can we speed up FW in a simple way?
- Rule of thumb in optimization: follow the steepest direction

# Boosting Frank-Wolfe

- Can we speed up FW in a simple way?
- Rule of thumb in optimization: follow the steepest direction

Idea:

- Speed up FW by moving in a direction better aligned with $-\nabla f(x_t)$

# Boosting Frank-Wolfe

- Can we speed up FW in a simple way?
- Rule of thumb in optimization: follow the steepest direction

Idea:

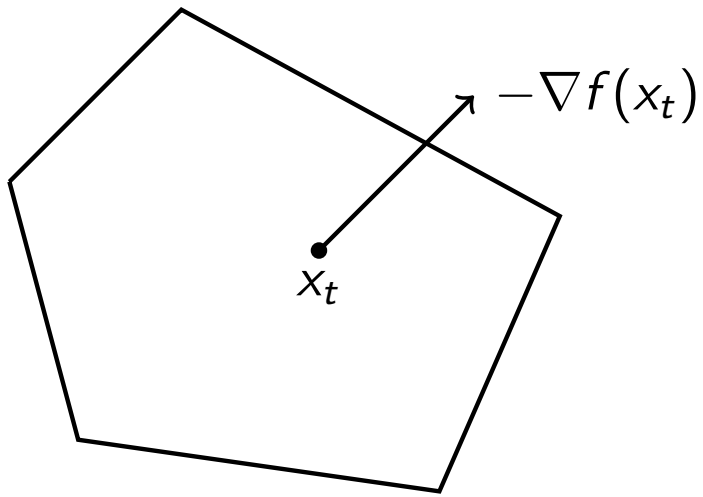
- Speed up FW by moving in a direction better aligned with $-\nabla f(x_t)$
- Build this direction by using $\mathcal{C}$ to maintain the projection-free property

# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

# Intuition

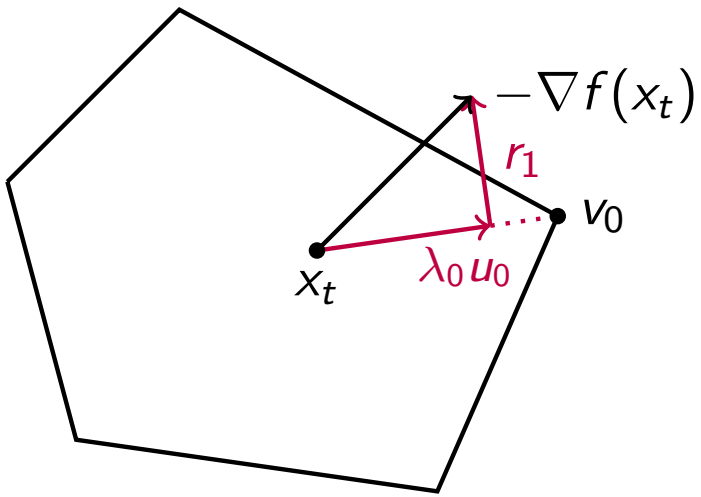
- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?



# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

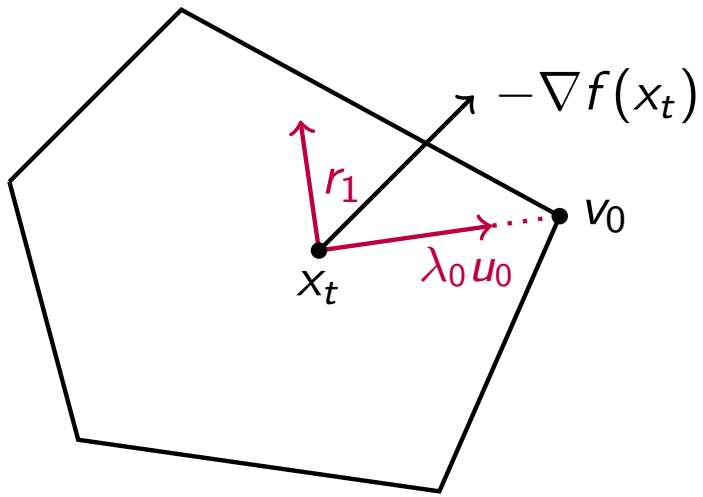
- $v_0 \in \arg\max_{v \in \mathcal{C}} \langle v, -\nabla f(x_t) \rangle$
  $\lambda_0 u_0 = \frac{\langle v_0 - x_t, -\nabla f(x_t) \rangle}{\|v_0 - x_t\|^2} (v_0 - x_t)$
  $r_1 = -\nabla f(x_t) - \lambda_0 u_0$



# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

- $v_0 \in \arg\max_{v \in \mathcal{C}} \langle v, -\nabla f(x_t) \rangle$
  $\lambda_0 u_0 = \frac{\langle v_0 - x_t, -\nabla f(x_t) \rangle}{\|v_0 - x_t\|^2} (v_0 - x_t)$
  $r_1 = -\nabla f(x_t) - \lambda_0 u_0$

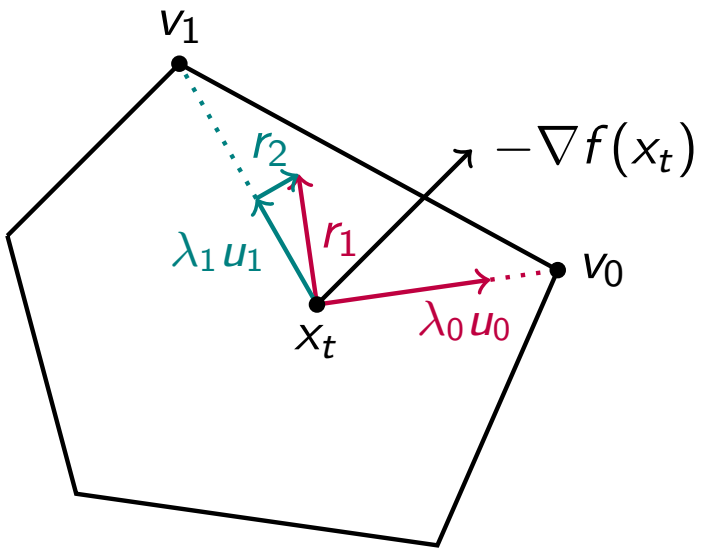


# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

- $v_0 \in \arg\max_{v \in \mathcal{C}} \langle v, -\nabla f(x_t) \rangle$
  $\lambda_0 u_0 = \frac{\langle v_0 - x_t, -\nabla f(x_t) \rangle}{\|v_0 - x_t\|^2}(v_0 - x_t)$
  $r_1 = -\nabla f(x_t) - \lambda_0 u_0$

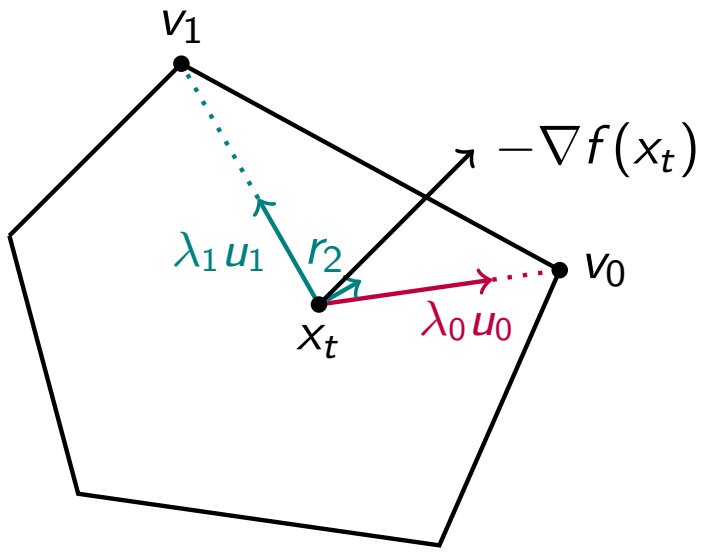
- $v_1 \in \arg\max_{v \in \mathcal{C}} \langle v, r_1 \rangle$
  $\lambda_1 u_1 = \frac{\langle v_1 - x_t, r_1 \rangle}{\|v_1 - x_t\|^2}(v_1 - x_t)$
  $r_2 = r_1 - \lambda_1 u_1$



# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

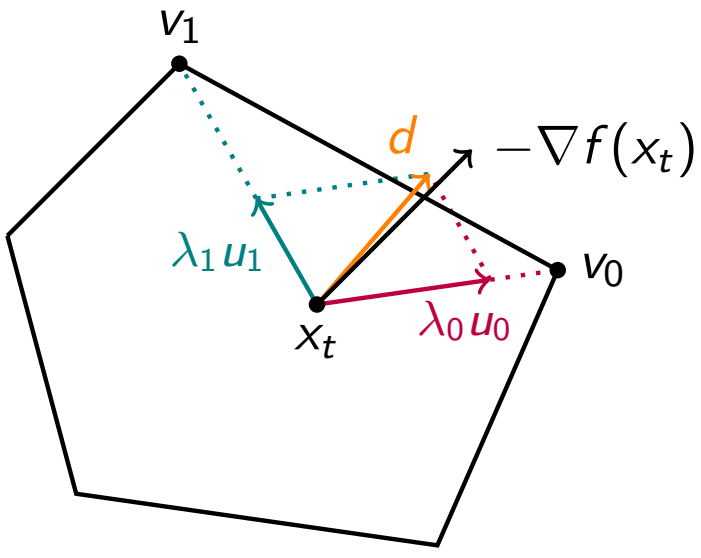
- $v_0 \in \arg\max_{v \in \mathcal{C}} \langle v, -\nabla f(x_t) \rangle$
  $\lambda_0 u_0 = \frac{\langle v_0 - x_t, -\nabla f(x_t) \rangle}{\|v_0 - x_t\|^2}(v_0 - x_t)$
  $r_1 = -\nabla f(x_t) - \lambda_0 u_0$
- $v_1 \in \arg\max_{v \in \mathcal{C}} \langle v, r_1 \rangle$
  $\lambda_1 u_1 = \frac{\langle v_1 - x_t, r_1 \rangle}{\|v_1 - x_t\|^2}(v_1 - x_t)$
  $r_2 = r_1 - \lambda_1 u_1$
- We could continue:
  $v_2 \in \arg\max_{v \in \mathcal{C}} \langle v, r_2 \rangle$



# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

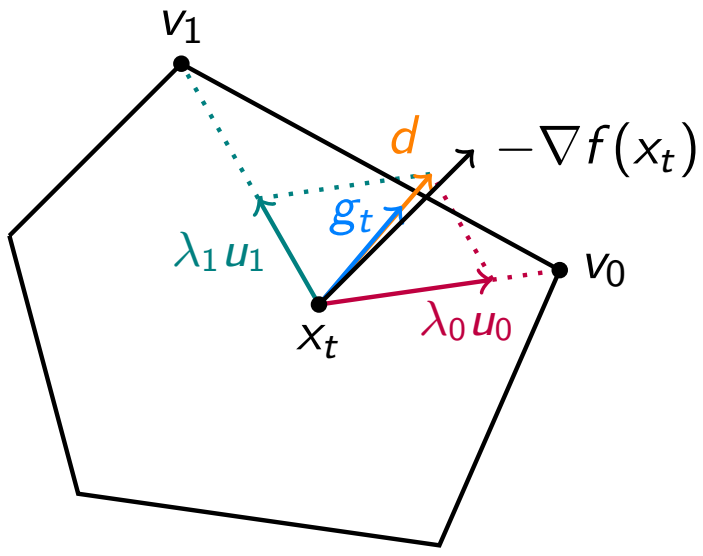
- $v_0 \in \arg\max_{v \in \mathcal{C}} \langle v, -\nabla f(x_t) \rangle$
  $\lambda_0 u_0 = \frac{\langle v_0 - x_t, -\nabla f(x_t) \rangle}{\|v_0 - x_t\|^2} (v_0 - x_t)$
  $r_1 = -\nabla f(x_t) - \lambda_0 u_0$
- $v_1 \in \arg\max_{v \in \mathcal{C}} \langle v, r_1 \rangle$
  $\lambda_1 u_1 = \frac{\langle v_1 - x_t, r_1 \rangle}{\|v_1 - x_t\|^2} (v_1 - x_t)$
  $r_2 = r_1 - \lambda_1 u_1$
- We could continue:
  $v_2 \in \arg\max_{v \in \mathcal{C}} \langle v, r_2 \rangle$
- $d = \lambda_0 u_0 + \lambda_1 u_1$



# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

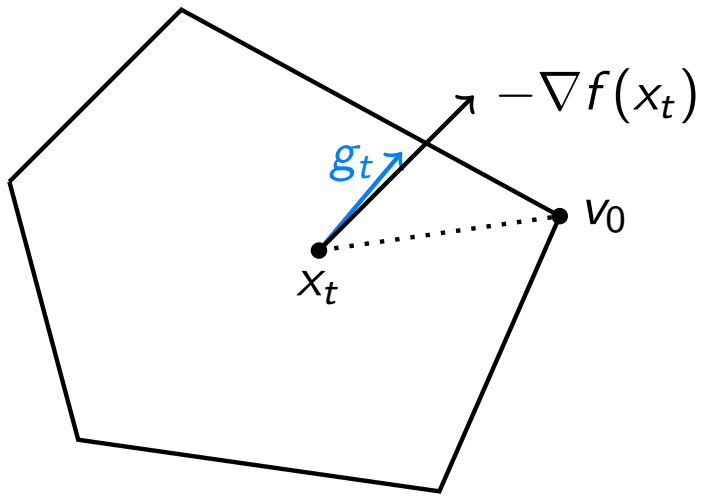
- $v_0 \in \arg\max_{v \in \mathcal{C}} \langle v, -\nabla f(x_t) \rangle$
  $\lambda_0 u_0 = \frac{\langle v_0 - x_t, -\nabla f(x_t) \rangle}{\|v_0 - x_t\|^2} (v_0 - x_t)$
  $r_1 = -\nabla f(x_t) - \lambda_0 u_0$
- $v_1 \in \arg\max_{v \in \mathcal{C}} \langle v, r_1 \rangle$
  $\lambda_1 u_1 = \frac{\langle v_1 - x_t, r_1 \rangle}{\|v_1 - x_t\|^2} (v_1 - x_t)$
  $r_2 = r_1 - \lambda_1 u_1$
- We could continue:
  $v_2 \in \arg\max_{v \in \mathcal{C}} \langle v, r_2 \rangle$
- $d = \lambda_0 u_0 + \lambda_1 u_1$
- $g_t = d/(\lambda_0 + \lambda_1)$



# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

- $v_0 \in \arg\max_{v \in \mathcal{C}} \langle v, -\nabla f(x_t) \rangle$
  $\lambda_0 u_0 = \frac{\langle v_0 - x_t, -\nabla f(x_t) \rangle}{\|v_0 - x_t\|^2}(v_0 - x_t)$
  $r_1 = -\nabla f(x_t) - \lambda_0 u_0$
- $v_1 \in \arg\max_{v \in \mathcal{C}} \langle v, r_1 \rangle$
  $\lambda_1 u_1 = \frac{\langle v_1 - x_t, r_1 \rangle}{\|v_1 - x_t\|^2}(v_1 - x_t)$
  $r_2 = r_1 - \lambda_1 u_1$
- We could continue:
  $v_2 \in \arg\max_{v \in \mathcal{C}} \langle v, r_2 \rangle$
- $d = \lambda_0 u_0 + \lambda_1 u_1$
- $g_t = d/(\lambda_0 + \lambda_1)$

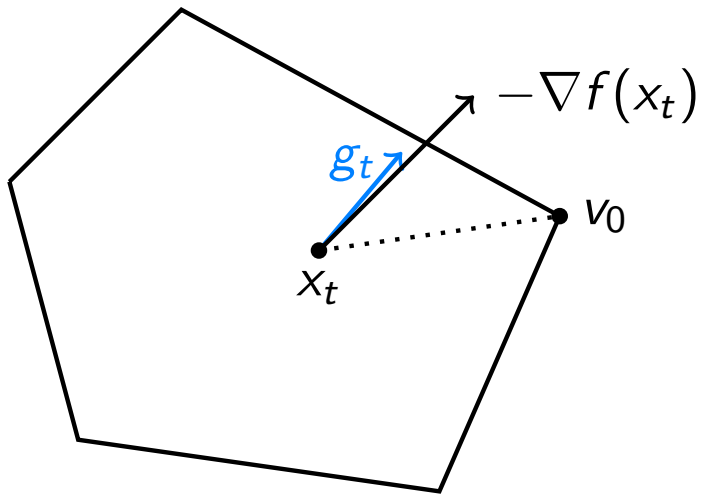



- The boosted direction $g_t$ is better aligned with $-\nabla f(x_t)$ than is the FW direction $v_0 - x_t$

# Intuition

- How can we build a direction better aligned with $-\nabla f(x_t)$ and that allows to update $x_{t+1}$ without projection?

- $v_0 \in \arg\max_{v \in \mathcal{C}} \langle v, -\nabla f(x_t) \rangle$
  $\lambda_0 u_0 = \frac{\langle v_0 - x_t, -\nabla f(x_t) \rangle}{\|v_0 - x_t\|^2}(v_0 - x_t)$
  $r_1 = -\nabla f(x_t) - \lambda_0 u_0$
- $v_1 \in \arg\max_{v \in \mathcal{C}} \langle v, r_1 \rangle$
  $\lambda_1 u_1 = \frac{\langle v_1 - x_t, r_1 \rangle}{\|v_1 - x_t\|^2}(v_1 - x_t)$
  $r_2 = r_1 - \lambda_1 u_1$
- We could continue:
  $v_2 \in \arg\max_{v \in \mathcal{C}} \langle v, r_2 \rangle$
- $d = \lambda_0 u_0 + \lambda_1 u_1$
- $g_t = d/(\lambda_0 + \lambda_1)$




- The boosted direction $g_t$ is better aligned with $-\nabla f(x_t)$ than is the FW direction $v_0 - x_t$ and satisfies $[x_t, x_t + g_t] \subset \mathcal{C}$ so we can update

$$x_{t+1} = x_t + \gamma_t g_t \quad \text{for all } \gamma_t \in [0, 1]$$

# A generic boosting procedure

**Algorithm** Boosting procedure Boost($\mathbf{d}, \mathbf{z}, K, \delta$)

**Input:** $\mathbf{d} \neq 0$, $\mathbf{z} \in \mathcal{C}$, $K \in \mathbb{N}\backslash\{0\}$, $\delta \in ]0,1[$

$d_0 \leftarrow 0$, $\Lambda \leftarrow 0$
**for** $k = 0$ **to** $K - 1$ **do**
  $r_k \leftarrow \mathbf{d} - d_k$ ▷ $k$-th residual
  $v_k \leftarrow \arg\max_{v \in \mathcal{C}} \langle v, r_k \rangle$ ▷ FW oracle
  $u_k \leftarrow v_k - \mathbf{z}$
  $\lambda_k \leftarrow \langle u_k, r_k \rangle / \|u_k\|_2^2$
  $d'_k \leftarrow d_k + \lambda_k u_k$
  **if** $\cos(d'_k, \mathbf{d}) - \cos(d_k, \mathbf{d}) \geqslant \delta$ **then**
    $d_{k+1} \leftarrow d'_k$
    $\Lambda \leftarrow \Lambda + \lambda_k$
  **else**
    **break** ▷ exit $k$-loop
$g \leftarrow d_k / \Lambda$ ▷ normalization

- $\cos(\hat{d}, \mathbf{d}) = \frac{\langle \hat{d}, \mathbf{d} \rangle}{\|\hat{d}\|_2 \|\mathbf{d}\|_2}$ if $\hat{d} \neq 0$, else $-1$ if $\hat{d} = 0$

# A generic boosting procedure

**Algorithm** Boosting procedure Boost($\mathbf{d}, \mathbf{z}, K, \delta$)

**Input:** $\mathbf{d} \neq 0$, $\mathbf{z} \in \mathcal{C}$, $K \in \mathbb{N}\backslash\{0\}$, $\delta \in ]0, 1[$

$d_0 \leftarrow 0$, $\Lambda \leftarrow 0$
**for** $k = 0$ **to** $K - 1$ **do**
  $r_k \leftarrow \mathbf{d} - d_k$ ▷ $k$-th residual
  $v_k \leftarrow \arg\max_{v \in \mathcal{C}} \langle v, r_k \rangle$ ▷ FW oracle
  $u_k \leftarrow v_k - \mathbf{z}$
  $\lambda_k \leftarrow \langle u_k, r_k \rangle / \|u_k\|_2^2$
  $d'_k \leftarrow d_k + \lambda_k u_k$
  **if** $\cos(d'_k, \mathbf{d}) - \cos(d_k, \mathbf{d}) \geqslant \delta$ **then**
    $d_{k+1} \leftarrow d'_k$
    $\Lambda \leftarrow \Lambda + \lambda_k$
  **else**
    **break** ▷ exit $k$-loop
$g \leftarrow d_k / \Lambda$ ▷ normalization

- $\cos(\hat{d}, \mathbf{d}) = \frac{\langle \hat{d}, \mathbf{d} \rangle}{\|\hat{d}\|_2 \|\mathbf{d}\|_2}$ if $\hat{d} \neq 0$, else $-1$ if $\hat{d} = 0$
- The stopping criterion is an alignment improvement condition (typically $\delta \leftarrow 10^{-3}$ and $K \leftarrow +\infty$)

# A generic boosting procedure

**Algorithm** Boosting procedure Boost($\mathbf{d}, \mathbf{z}, K, \delta$)

**Input:** $\mathbf{d} \neq 0$, $\mathbf{z} \in \mathcal{C}$, $K \in \mathbb{N}\backslash\{0\}$, $\delta \in ]0, 1[$

$d_0 \leftarrow 0$, $\Lambda \leftarrow 0$
**for** $k = 0$ **to** $K - 1$ **do**
  $r_k \leftarrow \mathbf{d} - d_k$ ▷ $k$-th residual
  $v_k \leftarrow \arg\max_{v \in \mathcal{C}} \langle v, r_k \rangle$ ▷ FW oracle
  $u_k \leftarrow v_k - \mathbf{z}$
  $\lambda_k \leftarrow \langle u_k, r_k \rangle / \|u_k\|_2^2$
  $d'_k \leftarrow d_k + \lambda_k u_k$
  **if** $\cos(d'_k, \mathbf{d}) - \cos(d_k, \mathbf{d}) \geqslant \delta$ **then**
    $d_{k+1} \leftarrow d'_k$
    $\Lambda \leftarrow \Lambda + \lambda_k$
  **else**
    **break** ▷ exit $k$-loop
$g \leftarrow d_k / \Lambda$ ▷ normalization

- $\cos(\hat{d}, \mathbf{d}) = \frac{\langle \hat{d}, \mathbf{d} \rangle}{\|\hat{d}\|_2 \|\mathbf{d}\|_2}$ if $\hat{d} \neq 0$, else $-1$ if $\hat{d} = 0$
- The stopping criterion is an alignment improvement condition (typically $\delta \leftarrow 10^{-3}$ and $K \leftarrow +\infty$)

# The Boosted Frank-Wolfe algorithm

**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla f(x_t), v \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

**Algorithm** Boosted Frank-Wolfe (BoostFW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$, $K \in \mathbb{N} \backslash \{0\}$, $\delta \in ]0, 1[$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), x_t, K, \delta)$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$

# The Boosted Frank-Wolfe algorithm

**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0,1]$

1: **for** $t = 0$ **to** $T-1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla f(x_t), v \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

**Algorithm** Boosted Frank-Wolfe (BoostFW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0,1]$, $K \in \mathbb{N} \backslash \{0\}$, $\delta \in ]0,1[$

1: **for** $t = 0$ **to** $T-1$ **do**

2: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), x_t, K, \delta)$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$

# The Boosted Frank-Wolfe algorithm

**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla f(x_t), v \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

**Algorithm** Boosted Frank-Wolfe (BoostFW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$, $K \in \mathbb{N} \backslash \{0\}$, $\delta \in ]0, 1[$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), x_t, K, \delta)$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$

# The Boosted Frank-Wolfe algorithm

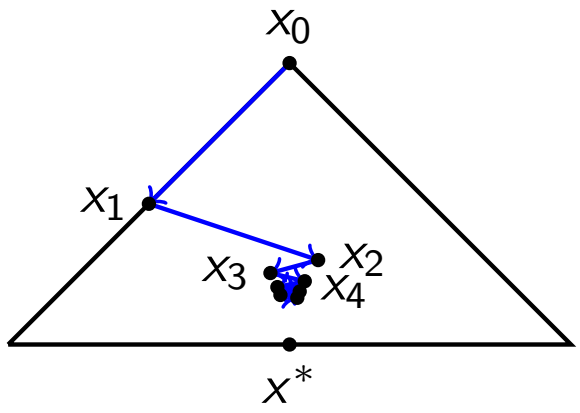
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0,1]$

1: **for** $t = 0$ **to** $T-1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla f(x_t), v \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



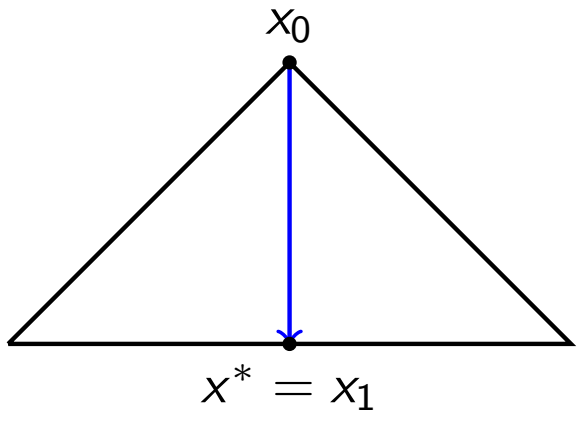

**Algorithm** Boosted Frank-Wolfe (BoostFW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0,1]$, $K \in \mathbb{N}\backslash\{0\}$, $\delta \in ]0,1[$

1: **for** $t = 0$ **to** $T-1$ **do**

2: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), x_t, K, \delta)$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$



# The Boosted Frank-Wolfe algorithm

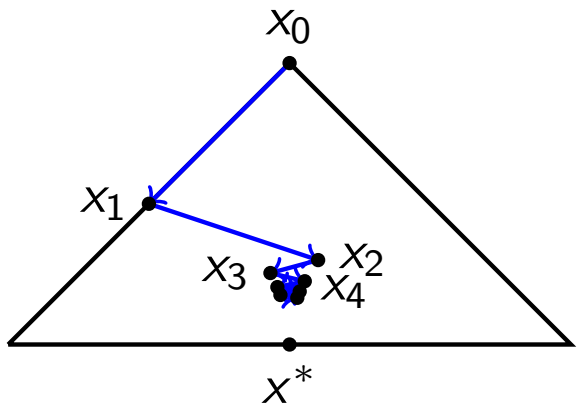
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0,1]$

1: **for** $t = 0$ **to** $T-1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla f(x_t), v \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



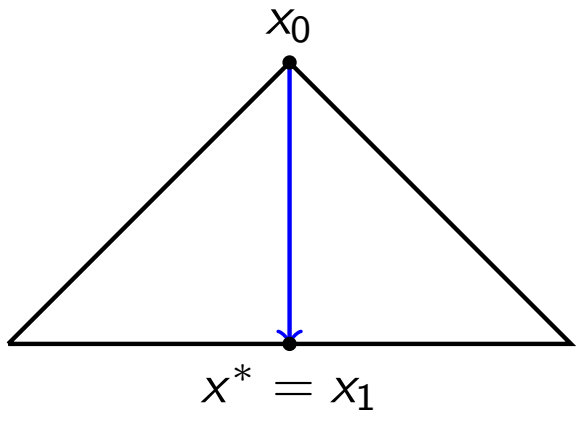

**Algorithm** Boosted Frank-Wolfe (BoostFW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0,1]$, $K \in \mathbb{N}\backslash\{0\}$, $\delta \in ]0,1[$

1: **for** $t = 0$ **to** $T-1$ **do**

2: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), x_t, K, \delta)$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$




- What is the convergence rate of BoostFW?

# The Boosted Frank-Wolfe algorithm

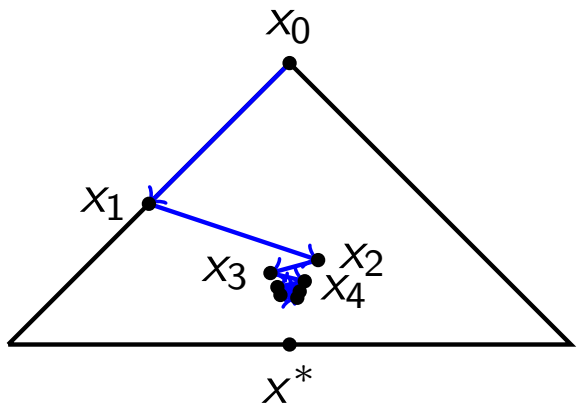
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0,1]$

1: **for** $t = 0$ **to** $T-1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla f(x_t), v \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



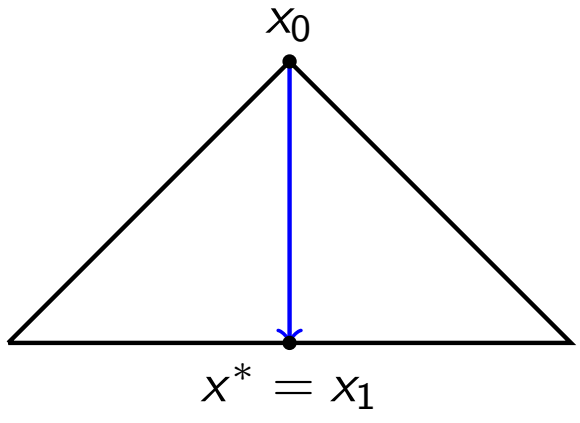

**Algorithm** Boosted Frank-Wolfe (BoostFW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0,1]$, $K \in \mathbb{N}\backslash\{0\}$, $\delta \in ]0,1[$

1: **for** $t = 0$ **to** $T-1$ **do**

2: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), x_t, K, \delta)$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$




- What is the convergence rate of BoostFW?
- Is BoostFW expensive in practice?

# The Boosted Frank-Wolfe algorithm

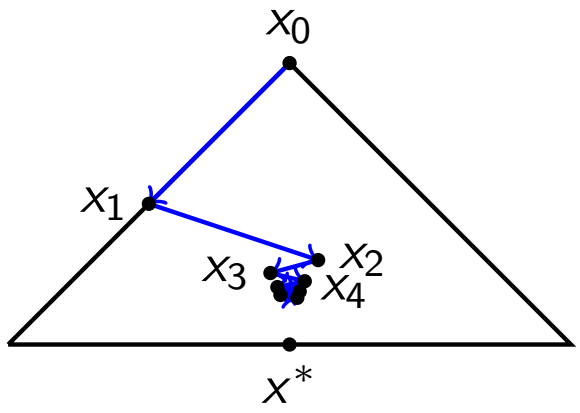
**Algorithm** Frank-Wolfe (FW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla f(x_t), v \rangle$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$



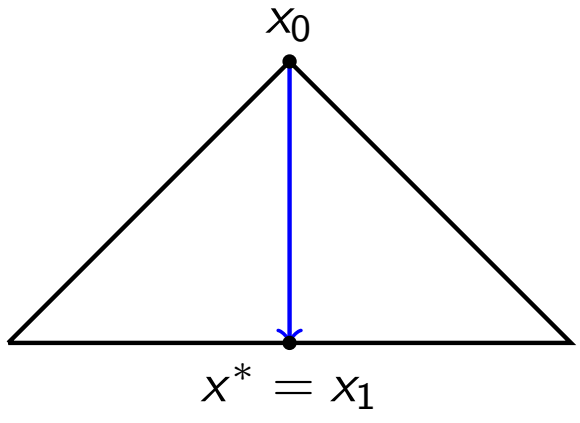

**Algorithm** Boosted Frank-Wolfe (BoostFW)

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$, $K \in \mathbb{N} \backslash \{0\}$, $\delta \in ]0, 1[$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), x_t, K, \delta)$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$




- What is the convergence rate of BoostFW?
- Is BoostFW expensive in practice?
- How does it compare to the state of the art?

# Convergence analysis

- Let $N_t$ be the number of iterations up to $t$ for which at least 2 rounds of alignment were performed (FW $=$ always 1 round) with a step-size $< 1$

# Convergence analysis

- Let $N_t$ be the number of iterations up to $t$ for which at least 2 rounds of alignment were performed (FW $=$ always 1 round) with a step-size $< 1$

**Theorem**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter $D$ and $f : \mathbb{R}^n \to \mathbb{R}$ be an $L$-smooth, convex, and $\mu$-gradient dominated function, and let $x_0 \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(y) \rangle$ for some $y \in \mathcal{C}$ and $\gamma_t \leftarrow \min\left\{ \frac{\langle g_t, -\nabla f(x_t) \rangle}{L \|g_t\|_2^2}, 1 \right\}$. Suppose that $N_t \geqslant \omega t$ where $\omega > 0$. Then*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2} \exp\left( -\delta^2 \frac{\mu}{L} \omega t \right)$$

# Convergence analysis

- Let $N_t$ be the number of iterations up to $t$ for which at least 2 rounds of alignment were performed (FW $=$ always 1 round) with a step-size $< 1$

**Theorem**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter $D$ and $f : \mathbb{R}^n \to \mathbb{R}$ be an $L$-smooth, convex, and $\mu$-gradient dominated function, and let $x_0 \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(y) \rangle$ for some $y \in \mathcal{C}$ and $\gamma_t \leftarrow \min\left\{ \frac{\langle g_t, -\nabla f(x_t) \rangle}{L \|g_t\|_2^2}, 1 \right\}$. Suppose that $N_t \geqslant \omega t$ where $\omega > 0$. Then*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2} \exp\left( -\delta^2 \frac{\mu}{L} \omega t \right)$$

- The assumption $N_t \geqslant \omega t$ simply states that $N_t$ is nonnegligeable, i.e., that the boosting procedure is active

# Convergence analysis

- Let $N_t$ be the number of iterations up to $t$ for which at least 2 rounds of alignment were performed (FW = always 1 round) with a step-size $< 1$

**Theorem**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter $D$ and $f : \mathbb{R}^n \to \mathbb{R}$ be an $L$-smooth, convex, and $\mu$-gradient dominated function, and let $x_0 \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(y) \rangle$ for some $y \in \mathcal{C}$ and $\gamma_t \leftarrow \min \left\{ \frac{\langle g_t, -\nabla f(x_t) \rangle}{L \|g_t\|_2^2}, 1 \right\}$. Suppose that $N_t \geqslant \omega t$ where $\omega > 0$. Then*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2} \exp\left(-\delta^2 \frac{\mu}{L} \omega t\right)$$

- The assumption $N_t \geqslant \omega t$ simply states that $N_t$ is nonnegligeable, i.e., that the boosting procedure is active
- Else, BoostFW reduces to FW and the convergence rate is $\frac{4LD^2}{t+2}$

# Convergence analysis

- Let $N_t$ be the number of iterations up to $t$ for which at least 2 rounds of alignment were performed (FW = always 1 round) with a step-size $< 1$

**Theorem**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter $D$ and $f : \mathbb{R}^n \to \mathbb{R}$ be an $L$-smooth, convex, and $\mu$-gradient dominated function, and let $x_0 \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(y) \rangle$ for some $y \in \mathcal{C}$ and $\gamma_t \leftarrow \min\left\{ \frac{\langle g_t, -\nabla f(x_t) \rangle}{L \|g_t\|_2^2}, 1 \right\}$. Suppose that $N_t \geqslant \omega t$ where $\omega > 0$. Then*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2} \exp\left( -\delta^2 \frac{\mu}{L} \omega t \right)$$

- The assumption $N_t \geqslant \omega t$ simply states that $N_t$ is nonnegligeable, i.e., that the boosting procedure is active
- Else, BoostFW reduces to FW and the convergence rate is $\frac{4LD^2}{t+2}$
- In practice, $N_t \approx t$ (so $\omega \lesssim 1$)

# Computational experiments

- We compare BoostFW to AFW, BCG, and DICG on a series of experiments involving various objective functions and feasible regions

# Computational experiments

- We compare BoostFW to AFW, BCG, and DICG on a series of experiments involving various objective functions and feasible regions

$$\min_{x \in \mathbb{R}^n} \|y - Ax\|_2^2 \quad \text{s.t. } \|x\|_1 \leqslant \tau$$

$$\min_{x \in \mathbb{R}^{|\mathcal{A}|}} \sum_{a \in \mathcal{A}} \tau_a x_a \left(1 + 0.03 \left(\frac{x_a}{c_a}\right)^4\right)$$
$$\text{s.t. } x_a = \sum_{r \in \mathcal{R}} \mathbb{1}_{\{a \in r\}} y_r \qquad a \in \mathcal{A}$$
$$\sum_{r \in \mathcal{R}_{i,j}} y_r = d_{i,j} \qquad (i,j) \in \mathcal{S}$$
$$y_r \geqslant 0 \qquad r \in \mathcal{R}_{i,j},\ (i,j) \in \mathcal{S}$$

$$\min_{x \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} \ln(1 + \exp(-y_i \langle a_i, x \rangle)) \quad \text{s.t. } \|x\|_1 \leqslant \tau$$

$$\min_{X \in \mathbb{R}^{m \times n}} \frac{1}{|\mathcal{I}|} \sum_{(i,j) \in \mathcal{I}} h_\rho(Y_{i,j} - X_{i,j}) \quad \text{s.t. } \|X\|_{\text{nuc}} \leqslant \tau$$

# Computational experiments

- We compare BoostFW to AFW, BCG, and DICG on a series of experiments involving various objective functions and feasible regions

$$\min_{x \in \mathbb{R}^n} \|y - Ax\|_2^2 \quad \text{s.t. } \|x\|_1 \leqslant \tau$$

$$\min_{x \in \mathbb{R}^{|\mathcal{A}|}} \sum_{a \in \mathcal{A}} \tau_a x_a \left(1 + 0.03 \left(\frac{x_a}{c_a}\right)^4\right)$$
$$\text{s.t. } x_a = \sum_{r \in \mathcal{R}} \mathbb{1}_{\{a \in r\}} y_r \qquad a \in \mathcal{A}$$
$$\sum_{r \in \mathcal{R}_{i,j}} y_r = d_{i,j} \qquad (i,j) \in \mathcal{S}$$
$$y_r \geqslant 0 \qquad r \in \mathcal{R}_{i,j},\ (i,j) \in \mathcal{S}$$

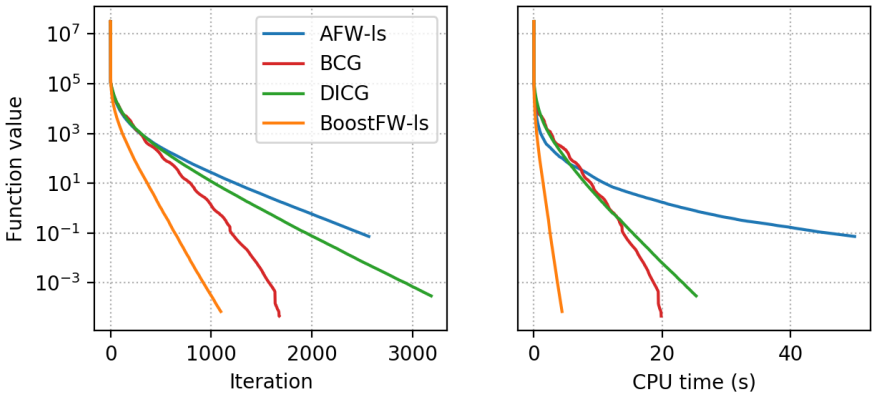
$$\min_{x \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} \ln(1 + \exp(-y_i \langle a_i, x \rangle)) \quad \text{s.t. } \|x\|_1 \leqslant \tau$$

$$\min_{X \in \mathbb{R}^{m \times n}} \frac{1}{|\mathcal{I}|} \sum_{(i,j) \in \mathcal{I}} h_\rho(Y_{i,j} - X_{i,j}) \quad \text{s.t. } \|X\|_{\text{nuc}} \leqslant \tau$$

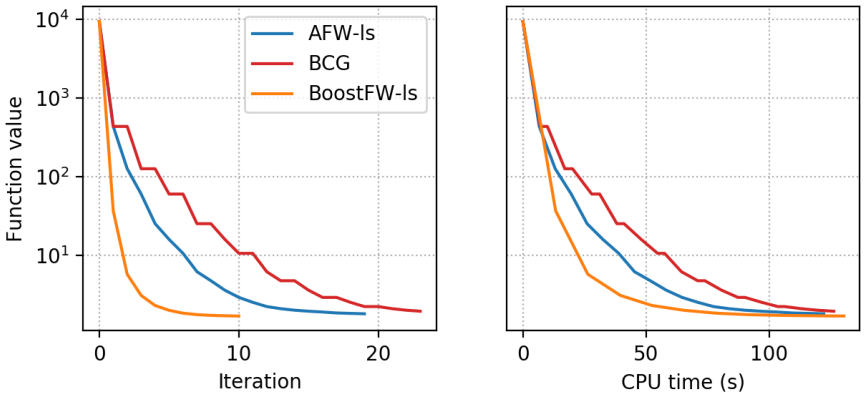
- For BoostFW and AFW we also run the line search-free strategies and label them with an "L"

# Computational experiments

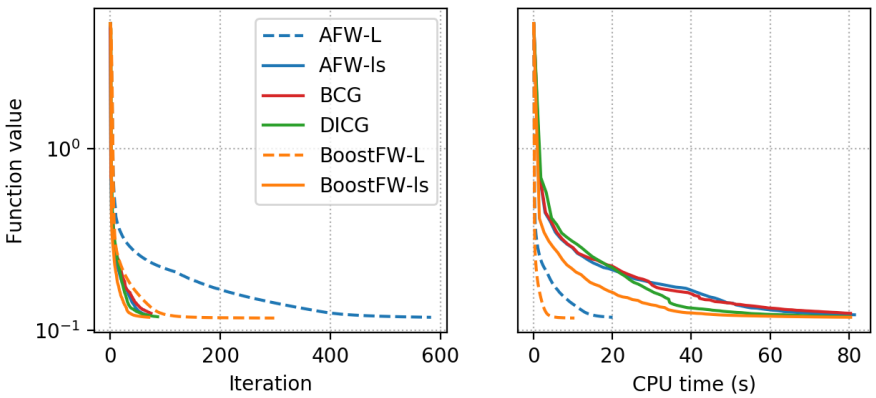
- Sparse signal recovery



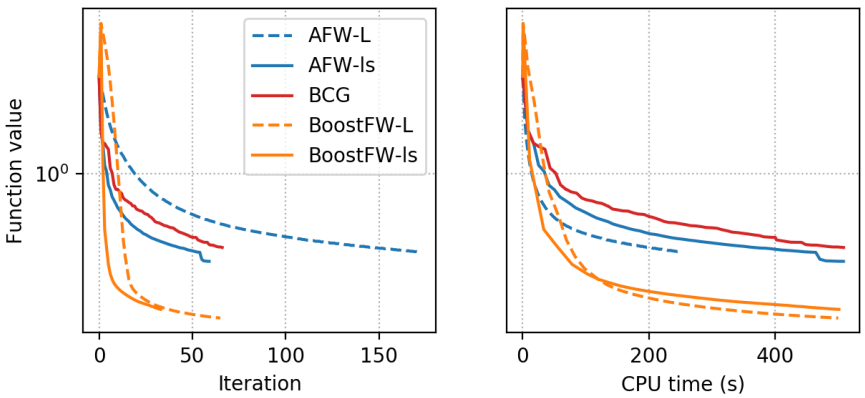

- Traffic assignment




- Sparse logistic regression on the Gisette dataset




- Collaborative filtering on the MovieLens 100k dataset

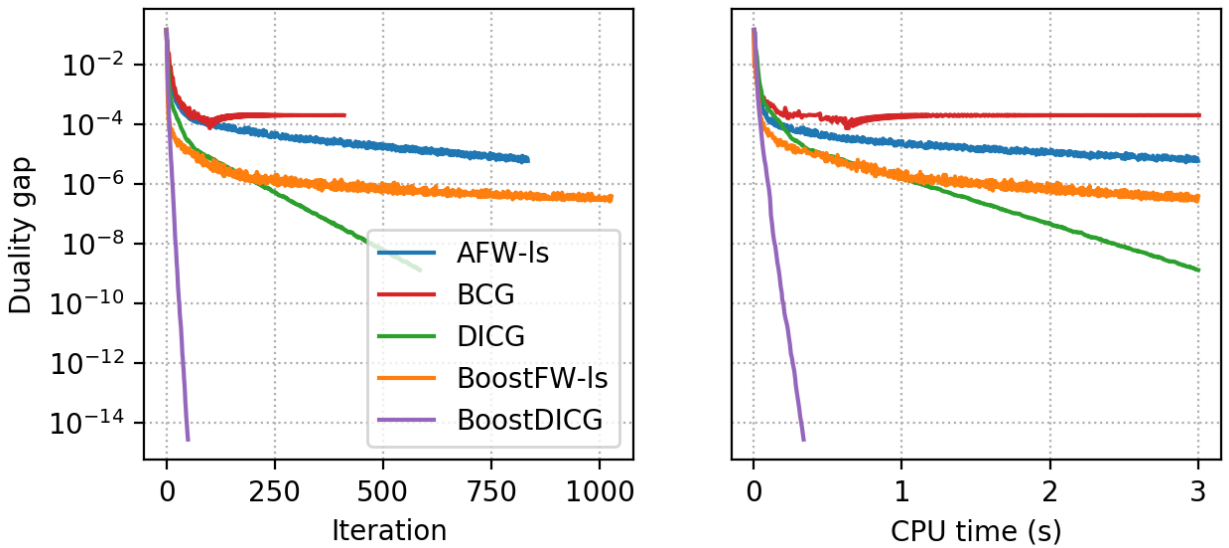


# Boosting DICG

- DICG is known to perform particularly well on the video co-localization experiment (YouTube-Objects dataset)

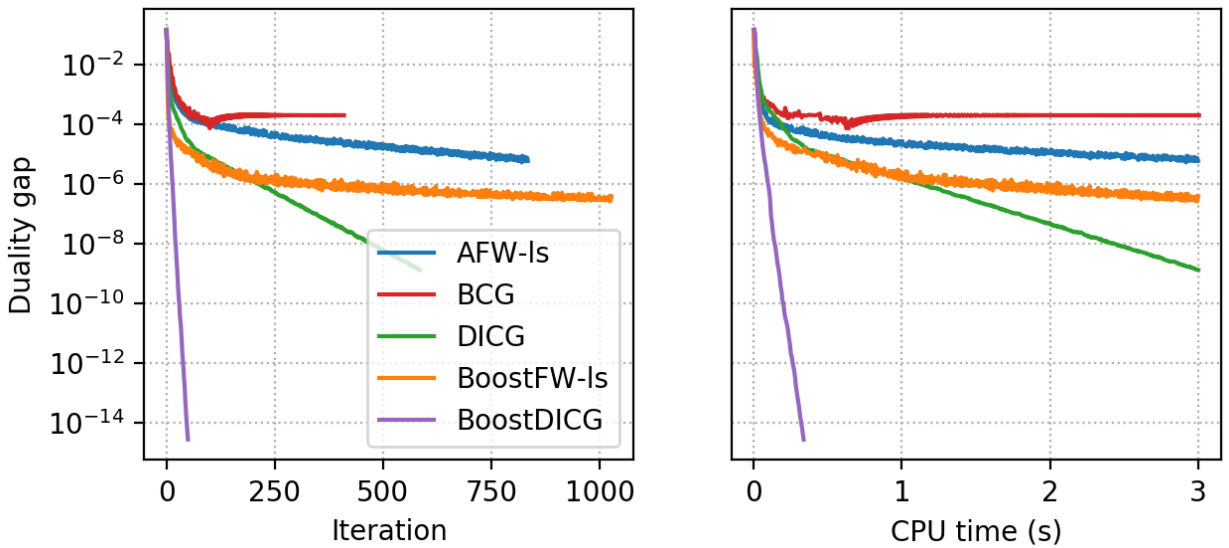
# Boosting DICG

- DICG is known to perform particularly well on the video co-localization experiment (YouTube-Objects dataset)
- BoostDICG: application of our method to DICG



# Boosting DICG

- DICG is known to perform particularly well on the video co-localization experiment (YouTube-Objects dataset)
- BoostDICG: application of our method to DICG




- (*details*)

DICG

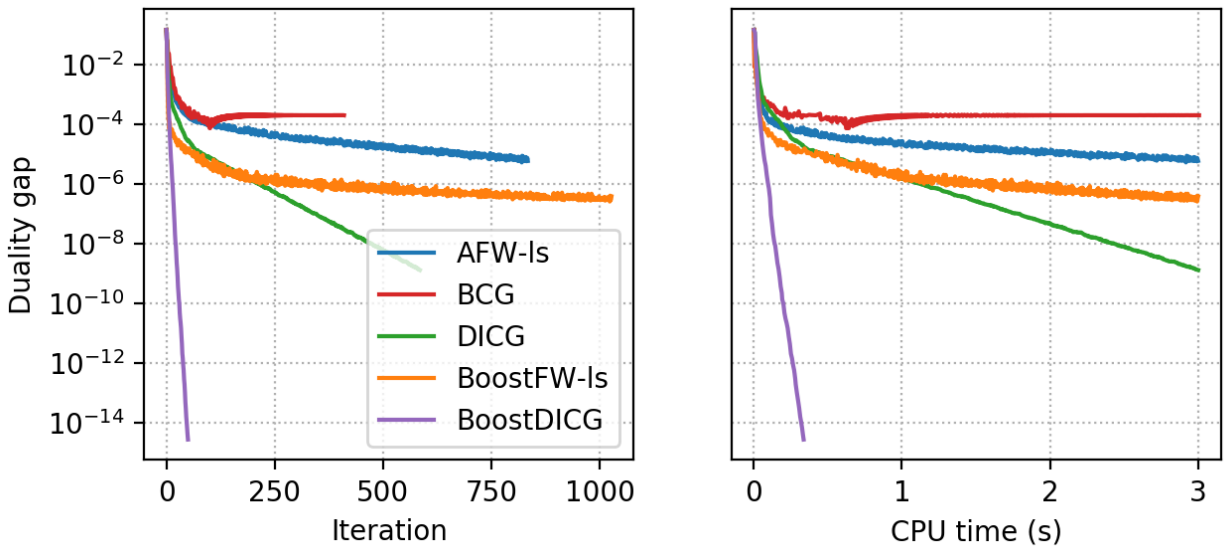
$a_t \leftarrow$ away vertex

$v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

$x_{t+1} \leftarrow x_t + \gamma_t (v_t - a_t)$

# Boosting DICG

- DICG is known to perform particularly well on the video co-localization experiment (YouTube-Objects dataset)
- BoostDICG: application of our method to DICG




- (*details*)

DICG

$a_t \leftarrow$ away vertex

$v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

$x_{t+1} \leftarrow x_t + \gamma_t (v_t - a_t)$

BoostDICG

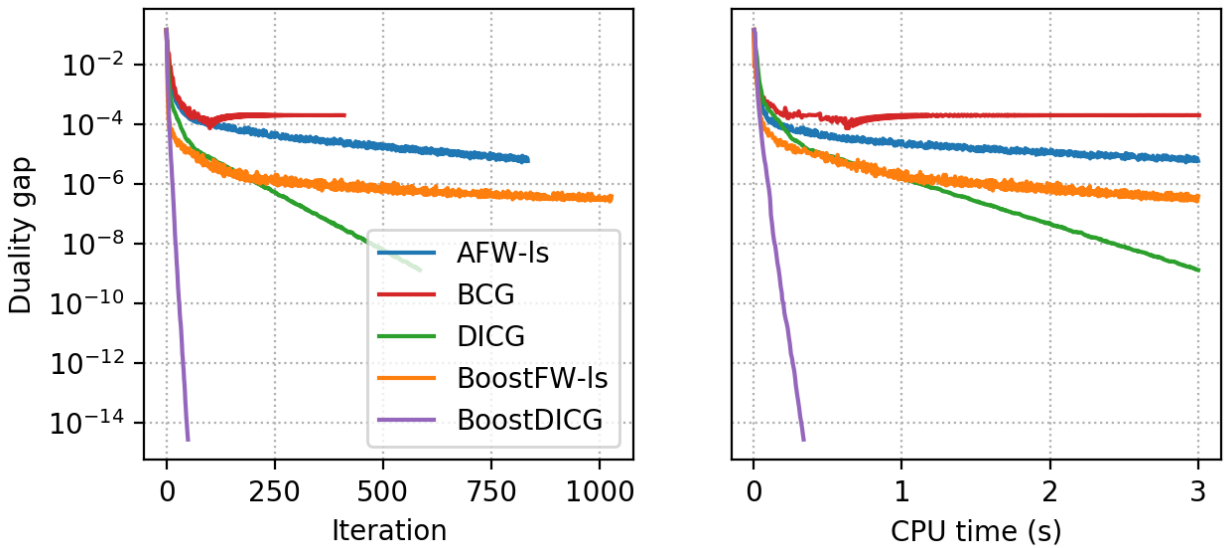
$a_t \leftarrow$ away vertex

$g_t \leftarrow \text{Boost}(-\nabla f(x_t), a_t, K, \delta)$

$x_{t+1} \leftarrow x_t + \gamma_t g_t$

# Boosting DICG

- DICG is known to perform particularly well on the video co-localization experiment (YouTube-Objects dataset)
- BoostDICG: application of our method to DICG




- (*details*)

DICG

$a_t \leftarrow$ away vertex

$v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

$x_{t+1} \leftarrow x_t + \gamma_t (v_t - a_t)$

BoostDICG

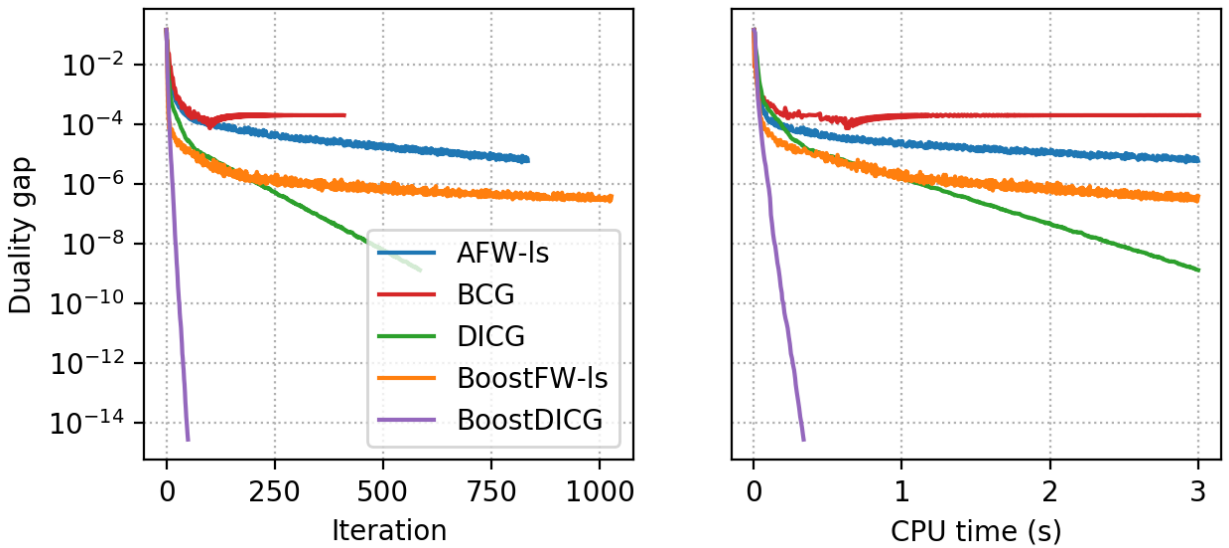
$a_t \leftarrow$ away vertex

$g_t \leftarrow \text{Boost}(-\nabla f(x_t), a_t, K, \delta)$

$x_{t+1} \leftarrow x_t + \gamma_t g_t$

# Boosting DICG

- DICG is known to perform particularly well on the video co-localization experiment (YouTube-Objects dataset)
- BoostDICG: application of our method to DICG




- (*details*)

DICG

$a_t \leftarrow$ away vertex

$v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$

$x_{t+1} \leftarrow x_t + \gamma_t (v_t - a_t)$

BoostDICG

$a_t \leftarrow$ away vertex

$g_t \leftarrow \text{Boost}(-\nabla f(x_t), a_t, K, \delta)$

$x_{t+1} \leftarrow x_t + \gamma_t g_t$

# Large-scale optimization

Consider

$$\min \left\{ f(x) := \frac{1}{m} \sum_{i=1}^{m} f_i(x) \right\}$$
$$\text{s.t. } x \in \mathcal{C}$$

where

- $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set
- $f_1, \ldots, f_m \colon \mathbb{R}^n \to \mathbb{R}$ are smooth (non)convex functions
- $m \ggg 1$ is very large

# Large-scale optimization

Consider

$$\min \left\{ f(x) := \frac{1}{m} \sum_{i=1}^{m} f_i(x) \right\}$$
$$\text{s.t. } x \in \mathcal{C}$$

where

- $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set
- $f_1, \ldots, f_m \colon \mathbb{R}^n \to \mathbb{R}$ are smooth (non)convex functions
- $m \ggg 1$ is very large

Computing $f(x)$ or $\nabla f(x)$ is too expensive

- Cannot use line search
- More efficient to use an estimator $\tilde{\nabla} f(x)$ to get approximate (but cheap) gradient information

# Stochastic Frank-Wolfe algorithms

**Template** Stochastic Frank-Wolfe

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \tilde{\nabla} f(x_t) \rangle$

4: $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

# Stochastic Frank-Wolfe algorithms

**Template** Stochastic Frank-Wolfe

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \tilde{\nabla} f(x_t) \rangle$

4: $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

- Use a stochastic estimator $\tilde{\nabla} f(x_t)$

# Stochastic Frank-Wolfe algorithms

**Template** Stochastic Frank-Wolfe

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: Update the gradient estimator $\tilde{\nabla} f(x_t)$
3: $v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \tilde{\nabla} f(x_t) \rangle$
4: $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

- Use a stochastic estimator $\tilde{\nabla} f(x_t)$
- The *vanilla* Stochastic Frank-Wolfe algorithm (SFW) estimates the gradient by averaging over a minibatch of size $b_t$:

$$\tilde{\nabla} f(x_t) \leftarrow \frac{1}{b_t} \sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) \quad \text{where} \quad i_1, \ldots, i_{b_t} \overset{\text{i.i.d.}}{\sim} \mathcal{U}(\llbracket 1, m \rrbracket)$$

# Stochastic Frank-Wolfe algorithms

**Template** Stochastic Frank-Wolfe

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: Update the gradient estimator $\tilde{\nabla} f(x_t)$
3: $v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \tilde{\nabla} f(x_t) \rangle$
4: $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

- Use a stochastic estimator $\tilde{\nabla} f(x_t)$
- The *vanilla* Stochastic Frank-Wolfe algorithm (SFW) estimates the gradient by averaging over a minibatch of size $b_t$:

$$\tilde{\nabla} f(x_t) \leftarrow \frac{1}{b_t} \sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) \quad \text{where} \quad i_1, \ldots, i_{b_t} \overset{\text{i.i.d.}}{\sim} \mathcal{U}(\llbracket 1, m \rrbracket)$$

- SFW converges with rate $\mathcal{O}(1/t)$ when $b_t \sim t^2$ (Hazan & Luo, 2016)

# Stochastic Frank-Wolfe algorithms

**Template** Stochastic Frank-Wolfe

**Input:** $x_0 \in \mathcal{C}$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: Update the gradient estimator $\tilde{\nabla} f(x_t)$
3: $v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \tilde{\nabla} f(x_t) \rangle$
4: $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

- Use a stochastic estimator $\tilde{\nabla} f(x_t)$
- The *vanilla* Stochastic Frank-Wolfe algorithm (SFW) estimates the gradient by averaging over a minibatch of size $b_t$:

$$\tilde{\nabla} f(x_t) \leftarrow \frac{1}{b_t} \sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) \quad \text{where} \quad i_1, \ldots, i_{b_t} \overset{\text{i.i.d.}}{\sim} \mathcal{U}(\llbracket 1, m \rrbracket)$$

- SFW converges with rate $\mathcal{O}(1/t)$ when $b_t \sim t^2$ (Hazan & Luo, 2016)
- New variants require only $b_t \sim t$ or $b_t \sim 1$

# The Adaptive Gradient algorithm

Simultaneously proposed by Duchi et al. (2011) and McMahan & Streeter (2010):

---

**Algorithm** Adaptive Gradient (AdaGrad)

---

**Input:** $x_0 \in \mathcal{C}$, $\delta > 0$, $\eta > 0$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $H_t \leftarrow \operatorname{diag}\left(\delta 1 + \sqrt{\sum_{s=0}^{t} \tilde{\nabla} f(x_s)^2}\right)$

4: $x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$

---

# The Adaptive Gradient algorithm

Simultaneously proposed by Duchi et al. (2011) and McMahan & Streeter (2010):

---

**Algorithm** Adaptive Gradient (AdaGrad)

---

**Input:** $x_0 \in \mathcal{C}$, $\delta > 0$, $\eta > 0$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $$H_t \leftarrow \operatorname{diag}\left(\delta 1 + \sqrt{\sum_{s=0}^{t} \tilde{\nabla} f(x_s)^2}\right)$$

4: $$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$$

---

# The Adaptive Gradient algorithm

Simultaneously proposed by Duchi et al. (2011) and McMahan & Streeter (2010):

**Algorithm** Adaptive Gradient (AdaGrad)

**Input:** $x_0 \in \mathcal{C}$, $\delta > 0$, $\eta > 0$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $H_t \leftarrow \operatorname{diag}\left(\delta 1 + \sqrt{\sum_{s=0}^{t} \tilde{\nabla} f(x_s)^2}\right)$

4: $x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$

# The Adaptive Gradient algorithm

Simultaneously proposed by Duchi et al. (2011) and McMahan & Streeter (2010):

---

**Algorithm** Adaptive Gradient (AdaGrad)

---

**Input:** $x_0 \in \mathcal{C}$, $\delta > 0$, $\eta > 0$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $H_t \leftarrow \operatorname{diag}\left(\delta 1 + \sqrt{\sum_{s=0}^{t} \tilde{\nabla} f(x_s)^2}\right)$

4: $x_{t+1} \leftarrow \underset{x \in \mathcal{C}}{\arg\min}\, \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$

---

# The Adaptive Gradient algorithm

Simultaneously proposed by Duchi et al. (2011) and McMahan & Streeter (2010):

---

**Algorithm** Adaptive Gradient (AdaGrad)

---

**Input:** $x_0 \in \mathcal{C}$, $\delta > 0$, $\eta > 0$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $H_t \leftarrow \operatorname{diag}\left(\delta 1 + \sqrt{\sum_{s=0}^{t} \tilde{\nabla} f(x_s)^2}\right)$

4: $x_{t+1} \leftarrow \underset{x \in \mathcal{C}}{\arg\min}\, \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$

---

- We denote $\|u\|_{H_t}^2 = \langle u, H_t u \rangle$

# The Adaptive Gradient algorithm

Simultaneously proposed by Duchi et al. (2011) and McMahan & Streeter (2010):

---

**Algorithm** Adaptive Gradient (AdaGrad)

---

**Input:** $x_0 \in \mathcal{C}$, $\delta > 0$, $\eta > 0$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $H_t \leftarrow \operatorname{diag}\left(\delta 1 + \sqrt{\sum_{s=0}^{t} \tilde{\nabla} f(x_s)^2}\right)$

4: $x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$

---

- We denote $\|u\|_{H_t}^2 = \langle u, H_t u \rangle$
- The default value for the offset is $\delta \leftarrow 10^{-8}$

# The Adaptive Gradient algorithm

Simultaneously proposed by Duchi et al. (2011) and McMahan & Streeter (2010):

---

**Algorithm** Adaptive Gradient (AdaGrad)

---

**Input:** $x_0 \in \mathcal{C}$, $\delta > 0$, $\eta > 0$

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $H_t \leftarrow \operatorname{diag}\left(\delta 1 + \sqrt{\sum_{s=0}^{t} \tilde{\nabla} f(x_s)^2}\right)$

4: $x_{t+1} \leftarrow \underset{x \in \mathcal{C}}{\arg\min}\, \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$

---

- We denote $\|u\|_{H_t}^2 = \langle u, H_t u \rangle$
- The default value for the offset is $\delta \leftarrow 10^{-8}$

# The Adaptive Gradient algorithm

We can rewrite

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \|x - (x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t))\|_{H_t}$$

# The Adaptive Gradient algorithm

We can rewrite

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \|x - (x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t))\|_{H_t}$$

Ignoring the constraint set $\mathcal{C}$ for ease of exposition, we obtain

$$x_{t+1} \leftarrow x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t)$$

# The Adaptive Gradient algorithm

We can rewrite

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \|x - (x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t))\|_{H_t}$$

Ignoring the constraint set $\mathcal{C}$ for ease of exposition, we obtain

$$x_{t+1} \leftarrow x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t)$$

i.e., for every feature $i \in [\![1, n]\!]$,

$$[x_{t+1}]_i \leftarrow [x_t]_i - \frac{\eta [\tilde{\nabla} f(x_t)]_i}{\delta + \sqrt{\sum_{s=0}^{t} [\tilde{\nabla} f(x_s)]_i^2}}$$

# The Adaptive Gradient algorithm

We can rewrite

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \|x - (x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t))\|_{H_t}$$

Ignoring the constraint set $\mathcal{C}$ for ease of exposition, we obtain

$$x_{t+1} \leftarrow x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t)$$

i.e., for every feature $i \in [\![1, n]\!]$,

$$[x_{t+1}]_i \leftarrow [x_t]_i - \frac{\eta [\tilde{\nabla} f(x_t)]_i}{\delta + \sqrt{\sum_{s=0}^{t} [\tilde{\nabla} f(x_s)]_i^2}}$$

- The offset $\delta$ prevents from dividing by zero

# The Adaptive Gradient algorithm

We can rewrite

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \|x - (x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t))\|_{H_t}$$

Ignoring the constraint set $\mathcal{C}$ for ease of exposition, we obtain

$$x_{t+1} \leftarrow x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t)$$

i.e., for every feature $i \in [\![1, n]\!]$,

$$[x_{t+1}]_i \leftarrow [x_t]_i - \frac{\eta [\tilde{\nabla} f(x_t)]_i}{\delta + \sqrt{\sum_{s=0}^{t} [\tilde{\nabla} f(x_s)]_i^2}}$$

- The offset $\delta$ prevents from dividing by zero
- The step-size automatically adjusts to the geometry of the problem

# The Adaptive Gradient algorithm

We can rewrite

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \|x - (x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t))\|_{H_t}$$

Ignoring the constraint set $\mathcal{C}$ for ease of exposition, we obtain

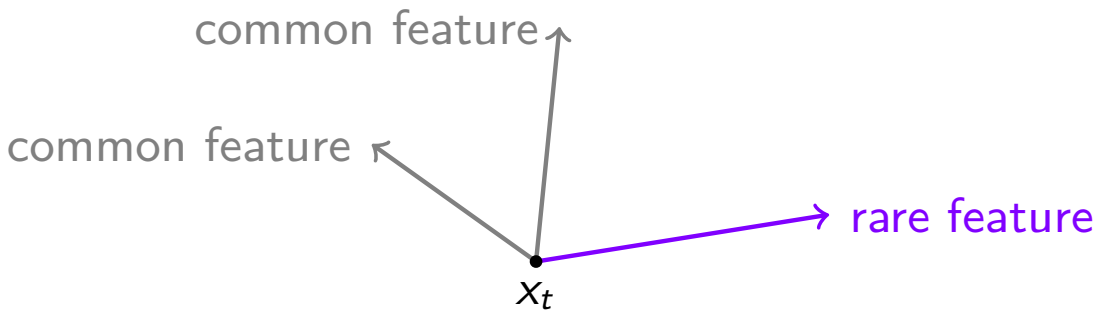
$$x_{t+1} \leftarrow x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t)$$

i.e., for every feature $i \in [\![1, n]\!]$,

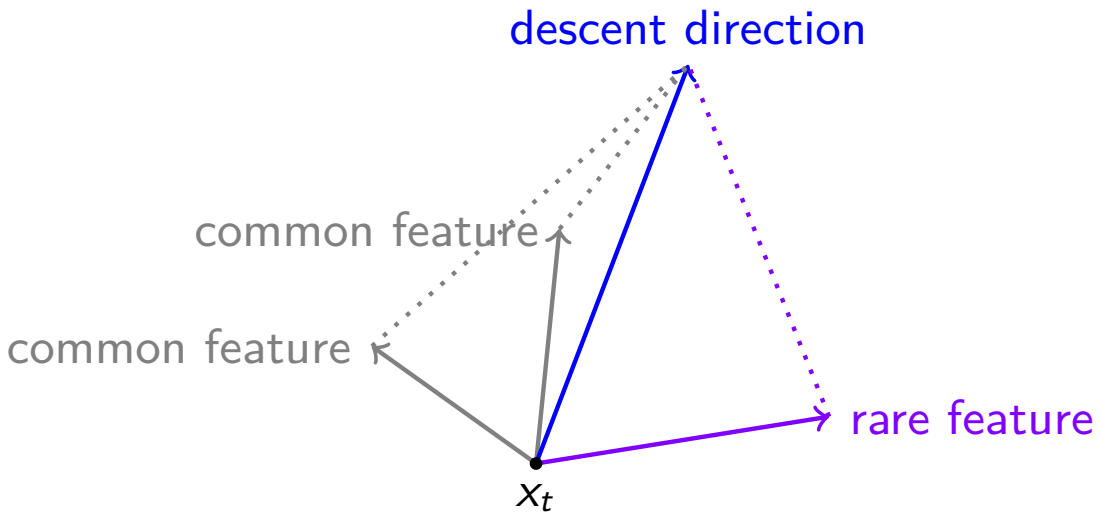
$$[x_{t+1}]_i \leftarrow [x_t]_i - \frac{\eta [\tilde{\nabla} f(x_t)]_i}{\delta + \sqrt{\sum_{s=0}^{t} [\tilde{\nabla} f(x_s)]_i^2}}$$

- The offset $\delta$ prevents from dividing by zero
- The step-size automatically adjusts to the geometry of the problem
- Larger step-sizes are given to infrequent (but potentially very informative) features whenever they appear so that they do not go unnoticed

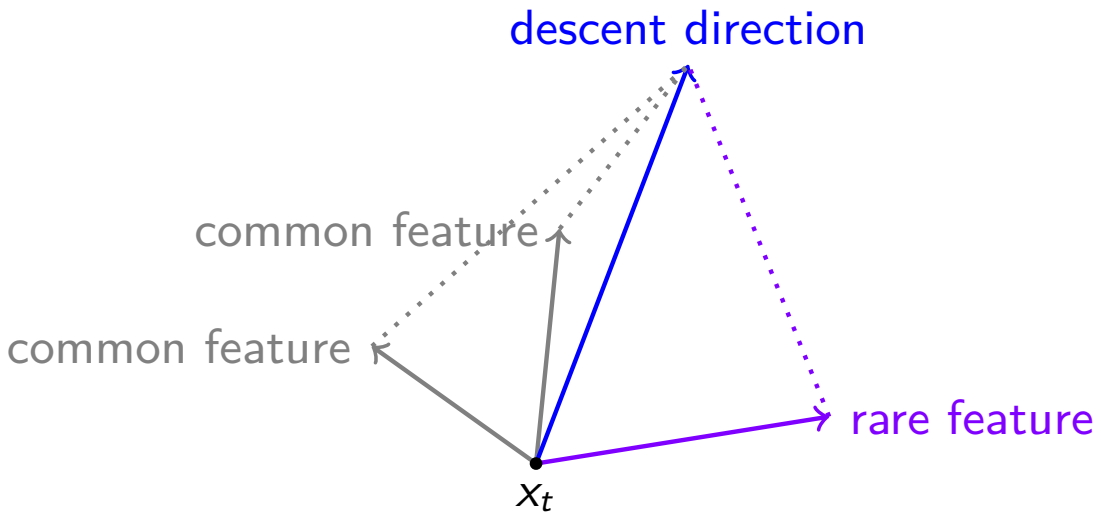
# The Adaptive Gradient algorithm



# The Adaptive Gradient algorithm

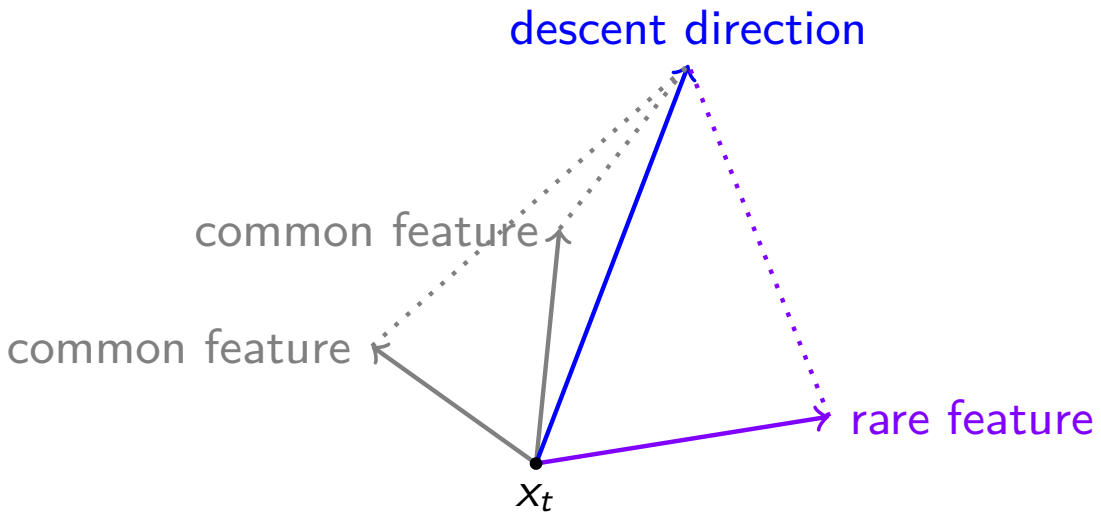


# The Adaptive Gradient algorithm

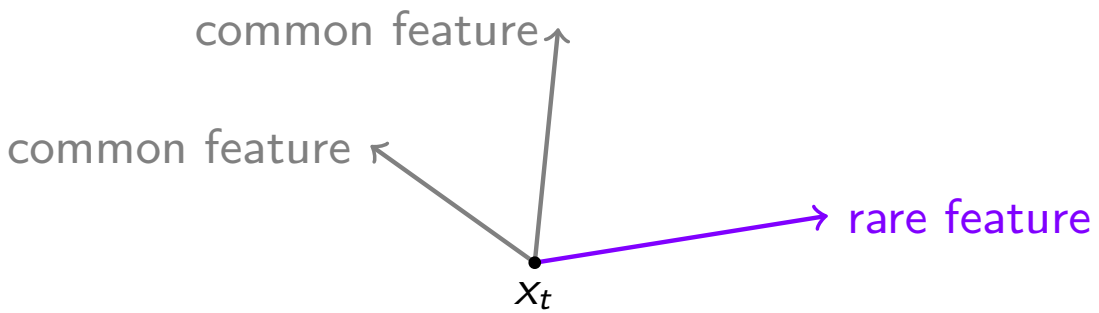



Adaptive step-sizes reduce the magnitude of the common features and increase those of the rare features:

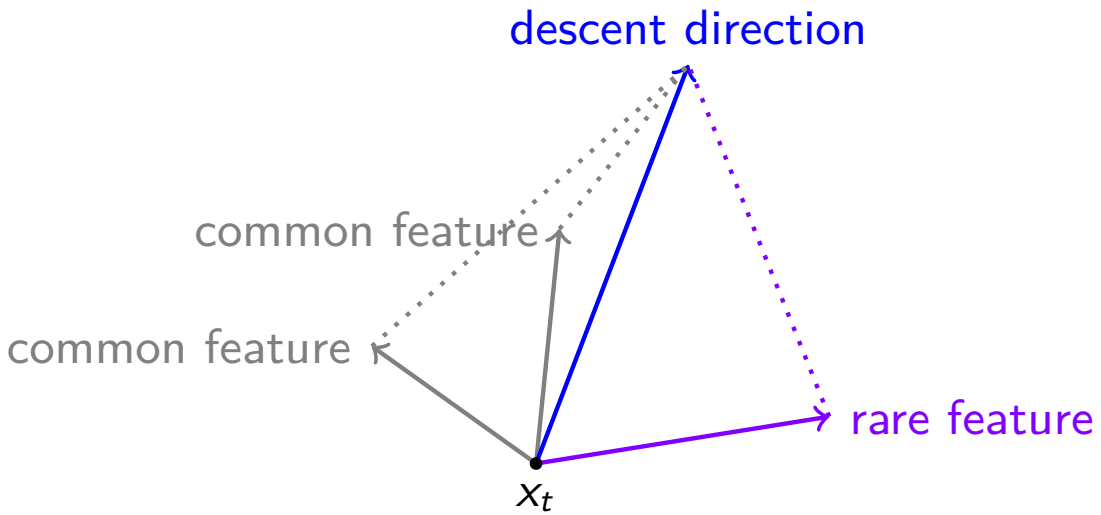
# The Adaptive Gradient algorithm



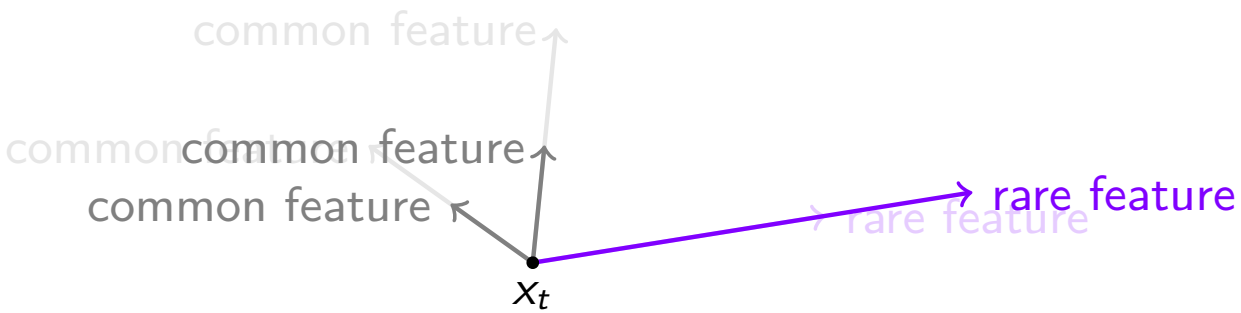

Adaptive step-sizes reduce the magnitude of the common features and increase those of the rare features:



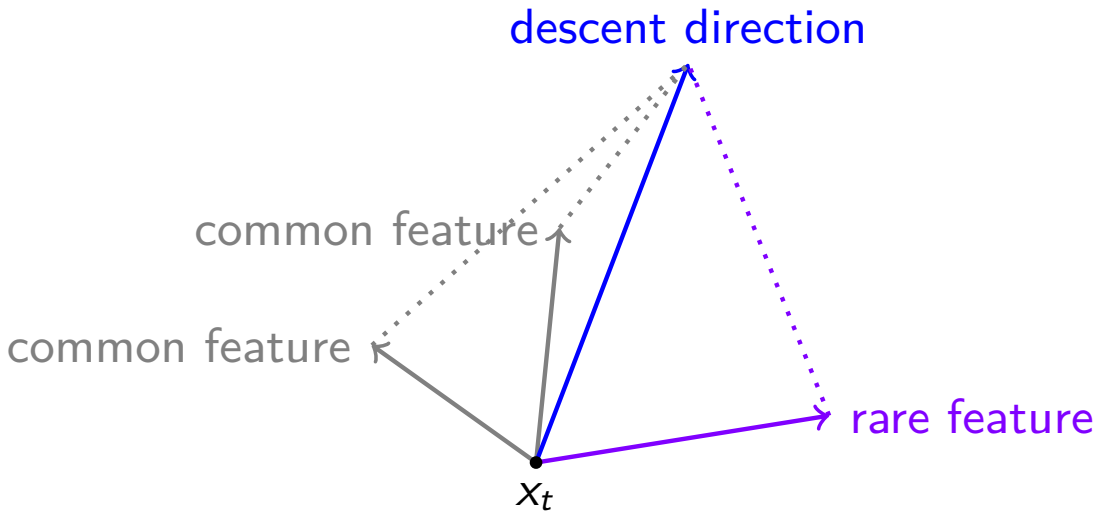
# The Adaptive Gradient algorithm



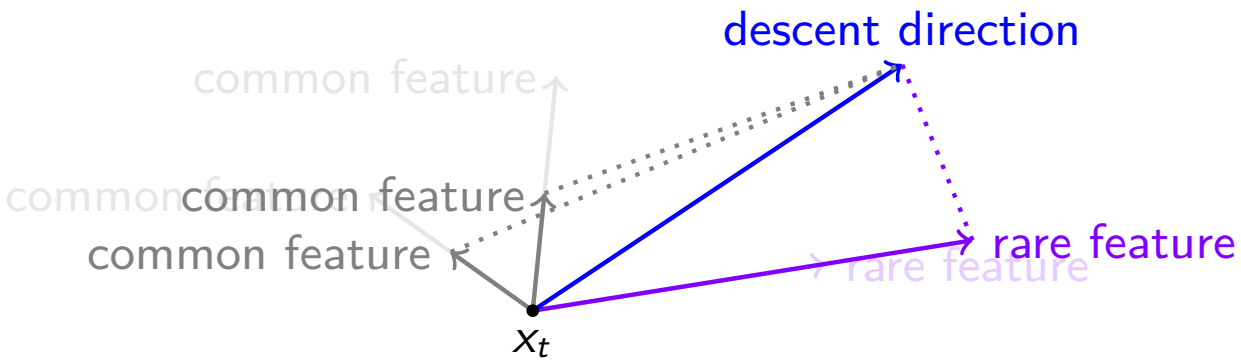

Adaptive step-sizes reduce the magnitude of the common features and increase those of the rare features:



# The Adaptive Gradient algorithm




Adaptive step-sizes reduce the magnitude of the common features and increase those of the rare features:



# Frank-Wolfe with adaptive gradients

- How can we use adaptive gradients in FW?

# Frank-Wolfe with adaptive gradients

- How can we use adaptive gradients in FW?
- Let $G_t = H_t^{-1}\tilde{\nabla} f(x_t)$, then unconstrained AdaGrad is

$$x_{t+1} \leftarrow x_t - \eta G_t$$

# Frank-Wolfe with adaptive gradients

- How can we use adaptive gradients in FW?
- Let $G_t = H_t^{-1}\tilde{\nabla} f(x_t)$, then unconstrained AdaGrad is

$$x_{t+1} \leftarrow x_t - \eta G_t$$

Could we do

$$v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, G_t \rangle$$
$$x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$$

as did FW for unconstrained gradient descent (for which $G_t = \nabla f(x_t)$)?

# Frank-Wolfe with adaptive gradients

- How can we use adaptive gradients in FW?
- Let $G_t = H_t^{-1}\tilde{\nabla} f(x_t)$, then unconstrained AdaGrad is

$$x_{t+1} \leftarrow x_t - \eta G_t$$

Could we do

$$v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, G_t \rangle$$
$$x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$$

as did FW for unconstrained gradient descent (for which $G_t = \nabla f(x_t)$)?

- We would likely lose the precious properties of the descent directions of AdaGrad

# Frank-Wolfe with adaptive gradients via sliding

- Instead, consider the constrained subproblem occurring at every iteration:

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$$

# Frank-Wolfe with adaptive gradients via sliding

- Instead, consider the constrained subproblem occurring at every iteration:

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$$

- This can become quite expensive and inefficient overall (AdaGrad is usually used for unconstrained optimization)

# Frank-Wolfe with adaptive gradients via sliding

- Instead, consider the constrained subproblem occurring at every iteration:

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$$

- This can become quite expensive and inefficient overall (AdaGrad is usually used for unconstrained optimization)

Idea:

- Solve the subproblem using FW (*sliding* technique of Lan & Zhou (2016))

# Frank-Wolfe with adaptive gradients via sliding

- Instead, consider the constrained subproblem occurring at every iteration:

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$$

- This can become quite expensive and inefficient overall (AdaGrad is usually used for unconstrained optimization)

Idea:

- Solve the subproblem using FW (*sliding* technique of Lan & Zhou (2016))
- Run only a small and fixed number $K$ of iterations of FW ($K \sim 5$)

# Frank-Wolfe with adaptive gradients via sliding

- Instead, consider the constrained subproblem occurring at every iteration:

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$$

- This can become quite expensive and inefficient overall (AdaGrad is usually used for unconstrained optimization)

Idea:

- Solve the subproblem using FW (*sliding* technique of Lan & Zhou (2016))
- Run only a small and fixed number $K$ of iterations of FW ($K \sim 5$)
- That is, we claim that leveraging just a small amount of information from the adaptive metric $H_t$ is enough

# The method

**Template** Frank-Wolfe with adaptive gradients

**Input:** $x_0 \in \mathcal{C}$, $0 < \lambda_t^- \leqslant \lambda_{t+1}^- \leqslant \lambda_{t+1}^+ \leqslant \lambda_t^+$, $K \in \mathbb{N}\backslash\{0\}$, $\eta > 0$, $\gamma_t \in [0, 1]$

**for** $t = 0$ **to** $T - 1$ **do**
  Update the gradient estimator $\tilde{\nabla} f(x_t)$
  Update the diagonal matrix $H_t$ and clip its entries to $[\lambda_t^-, \lambda_t^+]$
  $y_0^{(t)} \leftarrow x_t$
  **for** $k = 0$ **to** $K - 1$ **do**
    $\nabla Q_t(y_k^{(t)}) \leftarrow \tilde{\nabla} f(x_t) + \frac{1}{\eta_t} H_t(y_k^{(t)} - x_t)$
    $v_k^{(t)} \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla Q_t(y_k^{(t)}), v \rangle$
    $\gamma_k^{(t)} \leftarrow \min \left\{ \eta_t \frac{\langle \nabla Q_t(y_k^{(t)}), y_k^{(t)} - v_k^{(t)} \rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}, \gamma_t \right\}$
    $y_{k+1} \leftarrow y_k^{(t)} + \gamma_k^{(t)}(v_k^{(t)} - y_k^{(t)})$
  $x_{t+1} \leftarrow y_K^{(t)}$

# The method

**Template** Frank-Wolfe with adaptive gradients

**Input:** $x_0 \in \mathcal{C}$, $0 < \lambda_t^- \leqslant \lambda_{t+1}^- \leqslant \lambda_{t+1}^+ \leqslant \lambda_t^+$, $K \in \mathbb{N}\backslash\{0\}$, $\eta > 0$, $\gamma_t \in [0,1]$

**for** $t = 0$ **to** $T-1$ **do**
  Update the gradient estimator $\tilde{\nabla} f(x_t)$
  Update the diagonal matrix $H_t$ and clip its entries to $[\lambda_t^-, \lambda_t^+]$
  $y_0^{(t)} \leftarrow x_t$
  **for** $k = 0$ **to** $K-1$ **do**
    $\nabla Q_t(y_k^{(t)}) \leftarrow \tilde{\nabla} f(x_t) + \frac{1}{\eta_t} H_t(y_k^{(t)} - x_t)$
    $v_k^{(t)} \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla Q_t(y_k^{(t)}), v \rangle$
    $\gamma_k^{(t)} \leftarrow \min\left\{ \eta_t \frac{\langle \nabla Q_t(y_k^{(t)}), y_k^{(t)} - v_k^{(t)} \rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}, \gamma_t \right\}$
    $y_{k+1} \leftarrow y_k^{(t)} + \gamma_k^{(t)}(v_k^{(t)} - y_k^{(t)})$
  $x_{t+1} \leftarrow y_K^{(t)}$

# The method

**Template** Frank-Wolfe with adaptive gradients

**Input:** $x_0 \in \mathcal{C}$, $0 < \lambda_t^- \leqslant \lambda_{t+1}^- \leqslant \lambda_{t+1}^+ \leqslant \lambda_t^+$, $K \in \mathbb{N}\backslash\{0\}$, $\eta > 0$, $\gamma_t \in [0,1]$

**for** $t = 0$ **to** $T-1$ **do**
  Update the gradient estimator $\tilde{\nabla} f(x_t)$
  Update the diagonal matrix $H_t$ and clip its entries to $[\lambda_t^-, \lambda_t^+]$
  $y_0^{(t)} \leftarrow x_t$
  **for** $k = 0$ **to** $K-1$ **do**
    $\nabla Q_t(y_k^{(t)}) \leftarrow \tilde{\nabla} f(x_t) + \dfrac{1}{\eta_t} H_t(y_k^{(t)} - x_t)$
    $v_k^{(t)} \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla Q_t(y_k^{(t)}), v \rangle$
    $\gamma_k^{(t)} \leftarrow \min \left\{ \eta_t \dfrac{\langle \nabla Q_t(y_k^{(t)}), y_k^{(t)} - v_k^{(t)} \rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}, \gamma_t \right\}$
    $y_{k+1} \leftarrow y_k^{(t)} + \gamma_k^{(t)}(v_k^{(t)} - y_k^{(t)})$
  $x_{t+1} \leftarrow y_K^{(t)}$

# The method

**Template** Frank-Wolfe with adaptive gradients

**Input:** $x_0 \in \mathcal{C}$, $0 < \lambda_t^- \leqslant \lambda_{t+1}^- \leqslant \lambda_{t+1}^+ \leqslant \lambda_t^+$, $K \in \mathbb{N}\backslash\{0\}$, $\eta > 0$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: Update the gradient estimator $\tilde{\nabla} f(x_t)$
3: Update the diagonal matrix $H_t$ and clip its entries to $[\lambda_t^-, \lambda_t^+]$
4: $y_0^{(t)} \leftarrow x_t$
5: **for** $k = 0$ **to** $K - 1$ **do**
6: $\nabla Q_t(y_k^{(t)}) \leftarrow \tilde{\nabla} f(x_t) + \frac{1}{\eta_t} H_t(y_k^{(t)} - x_t)$
7: $v_k^{(t)} \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla Q_t(y_k^{(t)}), v \rangle$
8: $\gamma_k^{(t)} \leftarrow \min \left\{ \eta_t \frac{\langle \nabla Q_t(y_k^{(t)}), y_k^{(t)} - v_k^{(t)} \rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}, \gamma_t \right\}$
9: $y_{k+1} \leftarrow y_k^{(t)} + \gamma_k^{(t)}(v_k^{(t)} - y_k^{(t)})$
10: $x_{t+1} \leftarrow y_K^{(t)}$

- Lines 4–9 apply $K$ iterations of FW to $\min_{x \in \mathcal{C}} \left\{ Q_t(x) := f(x_t) + \langle x - x_t, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2\eta_t} \|x - x_t\|_{H_t}^2 \right\}$

# The method

**Template** Frank-Wolfe with adaptive gradients

**Input:** $x_0 \in \mathcal{C}$, $0 < \lambda_t^- \leqslant \lambda_{t+1}^- \leqslant \lambda_{t+1}^+ \leqslant \lambda_t^+$, $K \in \mathbb{N}\backslash\{0\}$, $\eta > 0$, $\gamma_t \in [0,1]$

1: **for** $t = 0$ **to** $T-1$ **do**
2: Update the gradient estimator $\tilde{\nabla} f(x_t)$
3: Update the diagonal matrix $H_t$ and clip its entries to $[\lambda_t^-, \lambda_t^+]$
4: $y_0^{(t)} \leftarrow x_t$
5: **for** $k = 0$ **to** $K-1$ **do**
6: $\nabla Q_t(y_k^{(t)}) \leftarrow \tilde{\nabla} f(x_t) + \frac{1}{\eta_t} H_t(y_k^{(t)} - x_t)$
7: $v_k^{(t)} \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla Q_t(y_k^{(t)}), v \rangle$
8: $\gamma_k^{(t)} \leftarrow \min\left\{ \eta_t \frac{\langle \nabla Q_t(y_k^{(t)}), y_k^{(t)} - v_k^{(t)} \rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}, \gamma_t \right\}$
9: $y_{k+1} \leftarrow y_k^{(t)} + \gamma_k^{(t)}(v_k^{(t)} - y_k^{(t)})$
10: $x_{t+1} \leftarrow y_K^{(t)}$

- Lines 4–9 apply $K$ iterations of FW to $\min_{x \in \mathcal{C}} \left\{ Q_t(x) := f(x_t) + \langle x - x_t, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2\eta_t} \|x - x_t\|_{H_t}^2 \right\}$

# The method

**Template** Frank-Wolfe with adaptive gradients

**Input:** $x_0 \in \mathcal{C}$, $0 < \lambda_t^- \leqslant \lambda_{t+1}^- \leqslant \lambda_{t+1}^+ \leqslant \lambda_t^+$, $K \in \mathbb{N}\backslash\{0\}$, $\eta > 0$, $\gamma_t \in [0, 1]$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: Update the gradient estimator $\tilde{\nabla} f(x_t)$
3: Update the diagonal matrix $H_t$ and clip its entries to $[\lambda_t^-, \lambda_t^+]$
4: $y_0^{(t)} \leftarrow x_t$
5: **for** $k = 0$ **to** $K - 1$ **do**
6: $\nabla Q_t(y_k^{(t)}) \leftarrow \tilde{\nabla} f(x_t) + \frac{1}{\eta_t} H_t(y_k^{(t)} - x_t)$
7: $v_k^{(t)} \leftarrow \arg\min_{v \in \mathcal{C}} \langle \nabla Q_t(y_k^{(t)}), v \rangle$
8: $\gamma_k^{(t)} \leftarrow \min \left\{ \eta_t \frac{\langle \nabla Q_t(y_k^{(t)}), y_k^{(t)} - v_k^{(t)} \rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}, \gamma_t \right\}$
9: $y_{k+1} \leftarrow y_k^{(t)} + \gamma_k^{(t)}(v_k^{(t)} - y_k^{(t)})$
10: $x_{t+1} \leftarrow y_K^{(t)}$

- Lines 4–9 apply $K$ iterations of FW to $\min_{x \in \mathcal{C}} \left\{ Q_t(x) := f(x_t) + \langle x - x_t, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2\eta_t} \|x - x_t\|_{H_t}^2 \right\}$
- Define AdaX depending on the strategy for $\tilde{\nabla} f(x_t)$: AdaSFW, AdaSVRF, etc.

# Convergence analysis

**Theorem**

*Let $f_1, \ldots, f_m \colon \mathbb{R}^n \to \mathbb{R}$ be $L$-smooth convex functions with $G = \max_{\mathcal{C}} \|\nabla f_i\|_2$. Then AdaSFW with $b_t \leftarrow \big(G(t+2)/(LD)\big)^2$, $\eta_t \leftarrow \lambda_t^- / L$, and $\gamma_t \leftarrow 2/(t+2)$ satisfies*

$$\mathbb{E}[f(x_t)] - \min_{\mathcal{C}} f \leqslant \frac{2LD^2(K+1+\kappa)}{t+1}$$

*where $\kappa = \lambda_0^+ / \lambda_0^-$*

# Convergence analysis

**Theorem**

*Let $f_1, \dots, f_m \colon \mathbb{R}^n \to \mathbb{R}$ be $L$-smooth convex functions with $G = \max_{\mathcal{C}} \|\nabla f_i\|_2$. Then AdaSFW with $b_t \leftarrow \left(G(t+2)/(LD)\right)^2$, $\eta_t \leftarrow \lambda_t^- / L$, and $\gamma_t \leftarrow 2/(t+2)$ satisfies*

$$\mathbb{E}[f(x_t)] - \min_{\mathcal{C}} f \leqslant \frac{2LD^2(K+1+\kappa)}{t+1}$$

*where $\kappa = \lambda_0^+ / \lambda_0^-$*

- In practice, no need to know $G, L, D$ and simply set $b_t = \Theta(t^2)$

# Convergence analysis

**Theorem**

*Let $f_1, \dots, f_m \colon \mathbb{R}^n \to \mathbb{R}$ be $L$-smooth convex functions with $G = \max_{\mathcal{C}} \|\nabla f_i\|_2$. Then AdaSFW with $b_t \leftarrow \left(G(t+2)/(LD)\right)^2$, $\eta_t \leftarrow \lambda_t^- / L$, and $\gamma_t \leftarrow 2/(t+2)$ satisfies*

$$\mathbb{E}[f(x_t)] - \min_{\mathcal{C}} f \leqslant \frac{2LD^2(K+1+\kappa)}{t+1}$$

*where $\kappa = \lambda_0^+ / \lambda_0^-$*

- In practice, no need to know $G, L, D$ and simply set $b_t = \Theta(t^2)$
- AdaSVRF and AdaCSFW also converge at a rate $\mathcal{O}(1/t)$ with $b_t \sim t$ and $b_t \sim 1$ respectively

# Convergence analysis

**Theorem**

*Let $f_1, \dots, f_m \colon \mathbb{R}^n \to \mathbb{R}$ be $L$-smooth convex functions with $G = \max_{\mathcal{C}} \|\nabla f_i\|_2$. Then AdaSFW with $b_t \leftarrow \left(G(t+2)/(LD)\right)^2$, $\eta_t \leftarrow \lambda_t^- / L$, and $\gamma_t \leftarrow 2/(t+2)$ satisfies*

$$\mathbb{E}[f(x_t)] - \min_{\mathcal{C}} f \leqslant \frac{2LD^2(K+1+\kappa)}{t+1}$$

*where $\kappa = \lambda_0^+ / \lambda_0^-$*

- In practice, no need to know $G, L, D$ and simply set $b_t = \Theta(t^2)$
- AdaSVRF and AdaCSFW also converge at a rate $\mathcal{O}(1/t)$ with $b_t \sim t$ and $b_t \sim 1$ respectively
- If $f_1, \dots, f_m$ are nonconvex, then AdaSFW converges to a stationary point at a rate $\mathcal{O}(1/t^{1/2-\nu})$ with $b_t \sim t$ and $\gamma_t \leftarrow 1/(t+1)^{1/2+\nu}$, $0 < \nu < 1/2$

# Computational experiments

- We compare our method to SFW, SVRF, SPIDER-FW, ORGFW, and CSFW, as well as AdaGrad and AMSGrad on a wide range of experiments

# Computational experiments

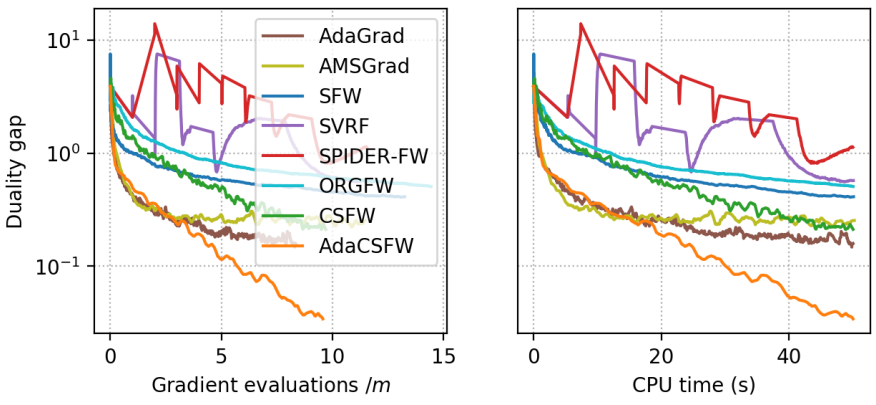
- We compare our method to SFW, SVRF, SPIDER-FW, ORGFW, and CSFW, as well as AdaGrad and AMSGrad on a wide range of experiments
- In addition, we run AdamSFW, a variant of AdaSFW with momentum inspired by Kingma & Ba (2015); Reddi et al. (2018)

# Computational experiments

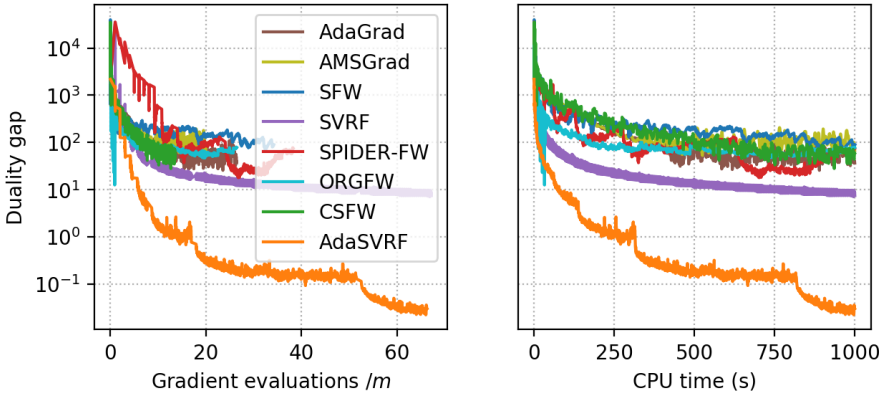
- We compare our method to SFW, SVRF, SPIDER-FW, ORGFW, and CSFW, as well as AdaGrad and AMSGrad on a wide range of experiments
- In addition, we run AdamSFW, a variant of AdaSFW with momentum inspired by Kingma & Ba (2015); Reddi et al. (2018)
- We set $K \sim 5$

# Convex objectives

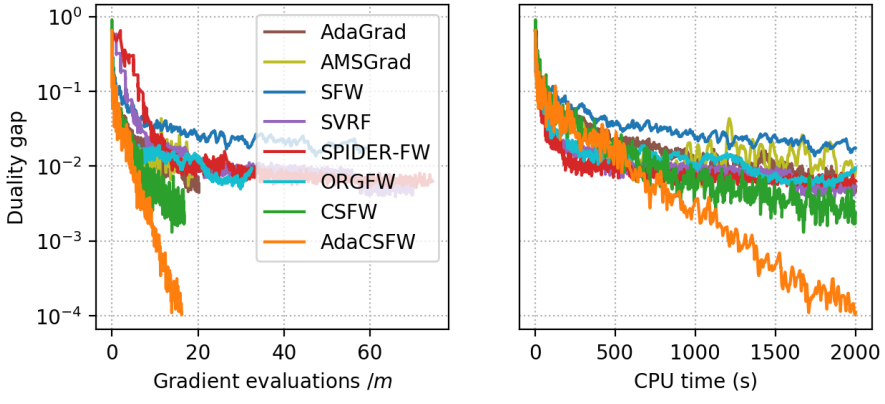
- Support vector classification on a synthetic dataset




- Linear regression on the YearPredictionMSD dataset

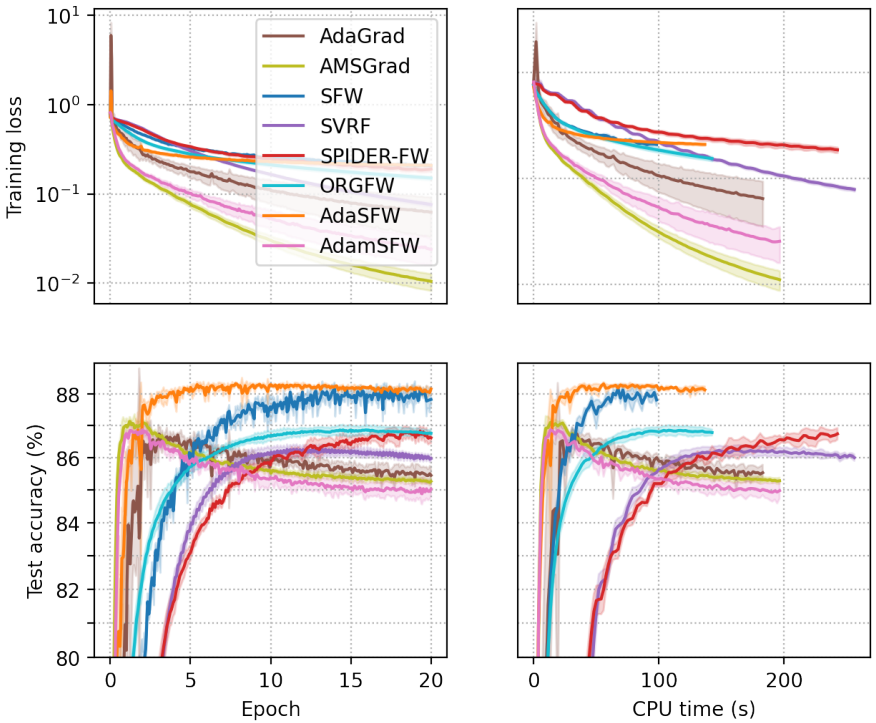



- Logistic regression on the RCV1 dataset

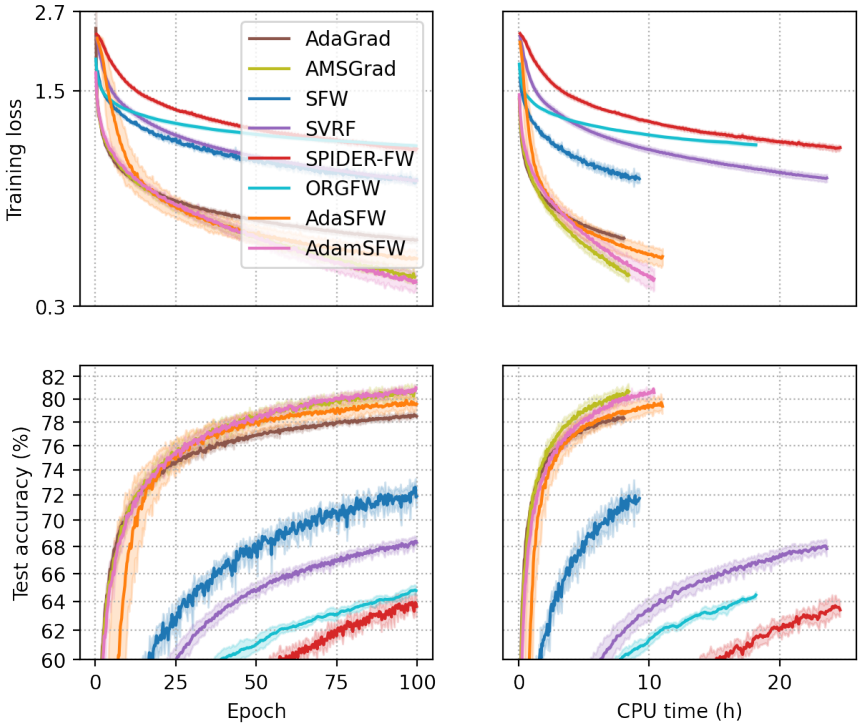


# Nonconvex objectives

- Neural network with one hidden layer on the IMDB dataset




- Convolutional network on the CIFAR-10 dataset




- Each layer is constrained into an $\ell_\infty$-ball

# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

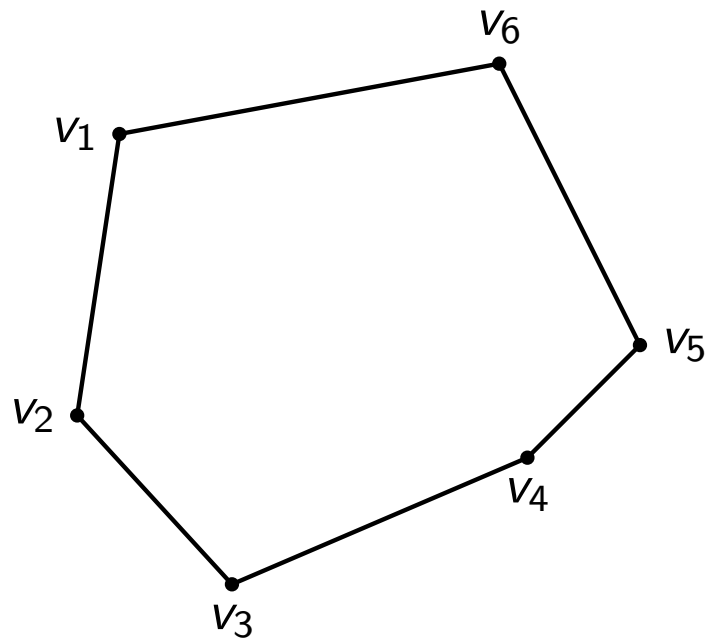
*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

- In $\mathbb{R}^2$, every point in $\mathcal{C}$ is a convex combination of at most 3 vertices

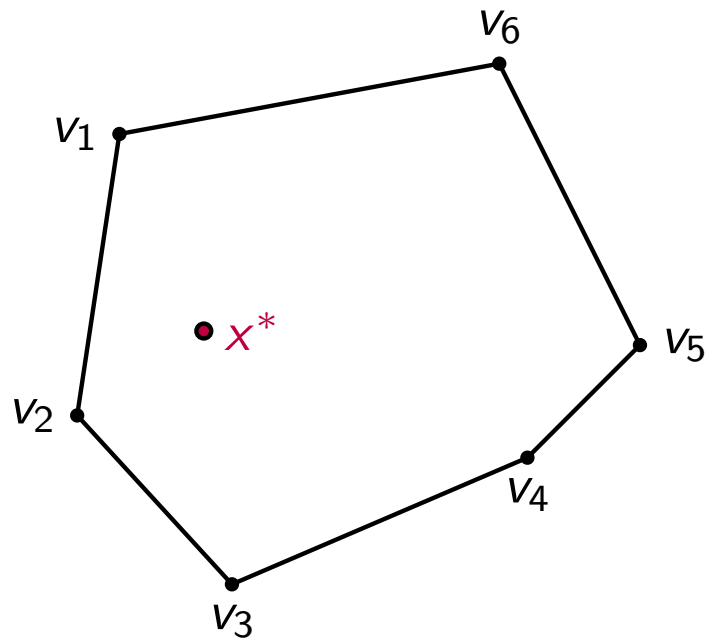


# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

- In $\mathbb{R}^2$, every point in $\mathcal{C}$ is a convex combination of at most 3 vertices

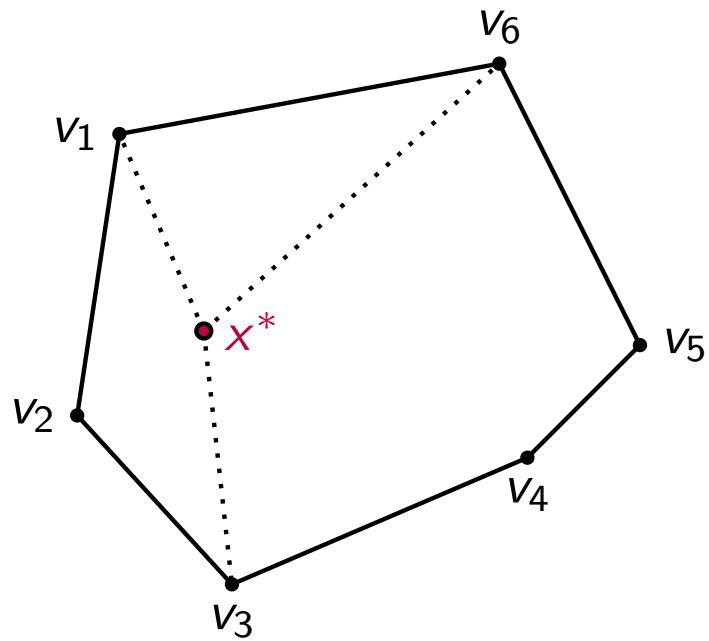


# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

- In $\mathbb{R}^2$, every point in $\mathcal{C}$ is a convex combination of at most 3 vertices

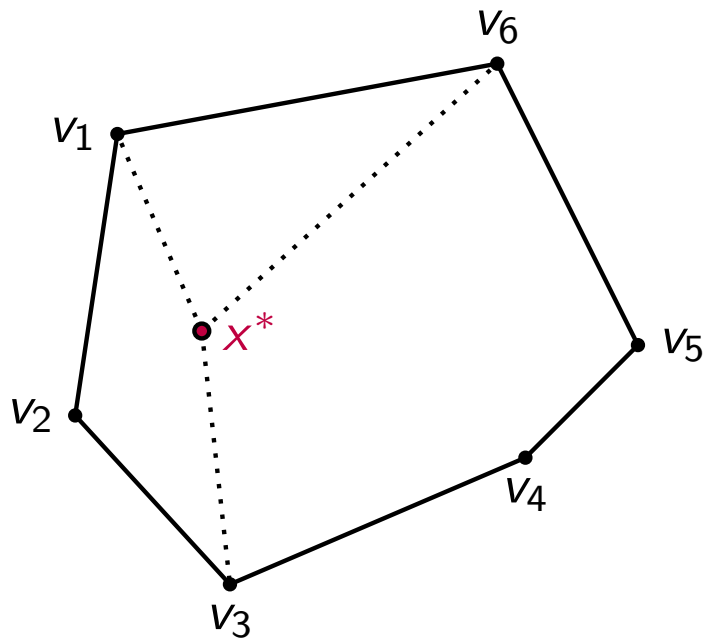


# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

- In $\mathbb{R}^2$, every point in $\mathcal{C}$ is a convex combination of at most 3 vertices
- Can we reduce $n+1$ when we can afford an $\varepsilon$-approximation?

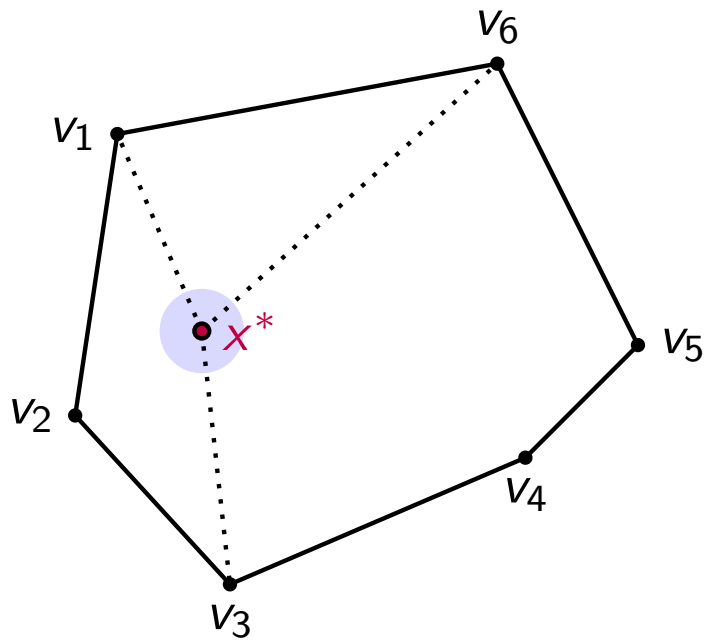


# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

- In $\mathbb{R}^2$, every point in $\mathcal{C}$ is a convex combination of at most 3 vertices
- Can we reduce $n+1$ when we can afford an $\varepsilon$-approximation?

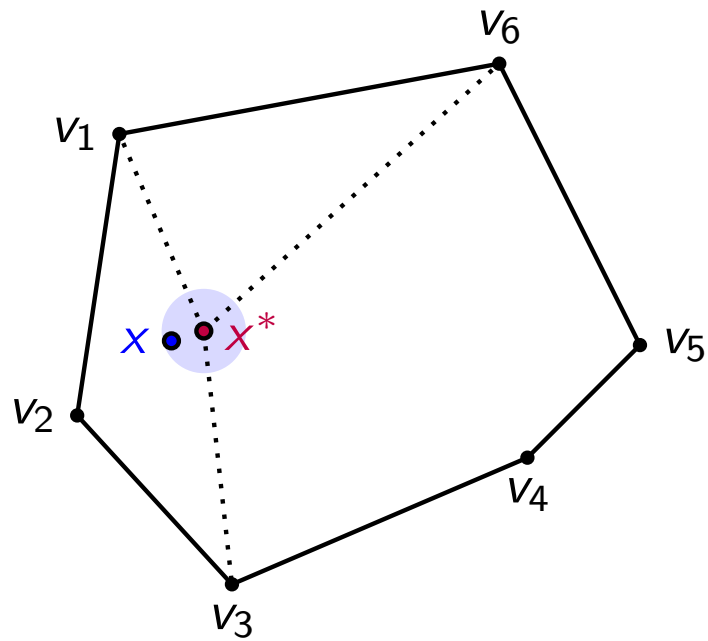


# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

- In $\mathbb{R}^2$, every point in $\mathcal{C}$ is a convex combination of at most 3 vertices
- Can we reduce $n+1$ when we can afford an $\varepsilon$-approximation?

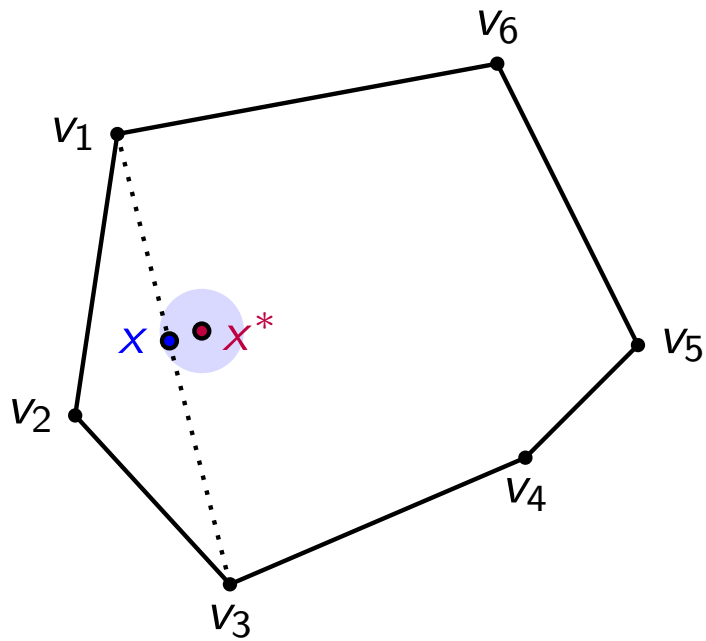


# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

- In $\mathbb{R}^2$, every point in $\mathcal{C}$ is a convex combination of at most 3 vertices
- Can we reduce $n+1$ when we can afford an $\varepsilon$-approximation?

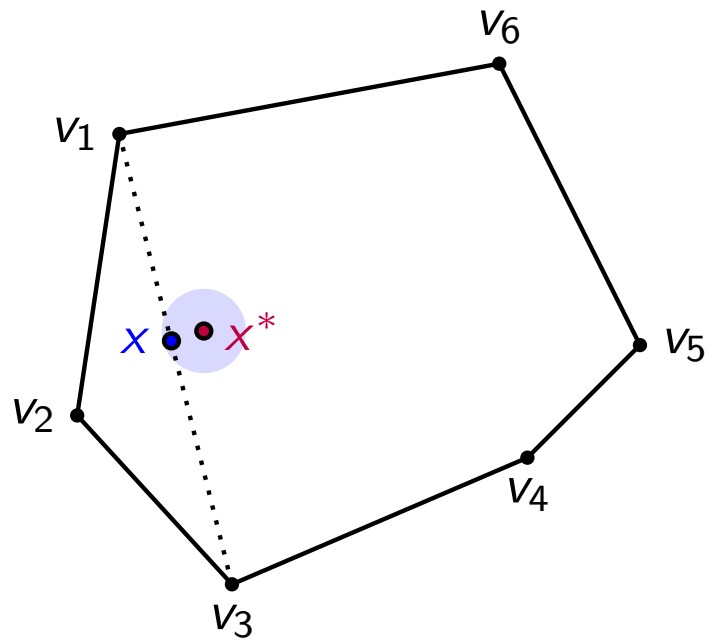


# The approximate Carathéodory problem

**Theorem (Carathéodory, 1907)**

*Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Then $x^*$ can be represented as a convex combination of at most $n+1$ vertices of $\mathcal{C}$.*

- In $\mathbb{R}^2$, every point in $\mathcal{C}$ is a convex combination of at most 3 vertices
- Can we reduce $n+1$ when we can afford an $\varepsilon$-approximation?
- Define the *cardinality* of $x \in \mathcal{C}$ as the number of vertices in a given convex decomposition of $x$



# The approximate Carathéodory problem

**Problem**

Find $x \in \mathcal{C}$ with low cardinality satisfying $\|x - x^*\|_p \leqslant \varepsilon$.

# The approximate Carathéodory problem

**Problem**

Find $x \in \mathcal{C}$ with low cardinality satisfying $\|x - x^*\|_p \leqslant \varepsilon$.

- Applications in game theory, combinatorial optimization, etc.

# The approximate Carathéodory problem

**Problem**

Find $x \in \mathcal{C}$ with low cardinality satisfying $\|x - x^*\|_p \leqslant \varepsilon$.

- Applications in game theory, combinatorial optimization, etc.

**Theorem (Barman, 2015)**

*Let $p \in [2, +\infty[$. Then there exists $x \in \mathcal{C}$ with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$, where $D_p$ is the diameter of $\mathcal{C}$ in the $\ell_p$-norm.*

# The approximate Carathéodory problem

**Problem**

Find $x \in \mathcal{C}$ with low cardinality satisfying $\|x - x^*\|_p \leqslant \varepsilon$.

- Applications in game theory, combinatorial optimization, etc.

**Theorem (Barman, 2015)**

*Let $p \in [2, +\infty[$. Then there exists $x \in \mathcal{C}$ with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$, where $D_p$ is the diameter of $\mathcal{C}$ in the $\ell_p$-norm.*

- This result is independent of the ambient dimension $n$

# The approximate Carathéodory problem

**Problem**

Find $x \in \mathcal{C}$ with low cardinality satisfying $\|x - x^*\|_p \leqslant \varepsilon$.

- Applications in game theory, combinatorial optimization, etc.

**Theorem (Barman, 2015)**

*Let $p \in [2, +\infty[$. Then there exists $x \in \mathcal{C}$ with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$, where $D_p$ is the diameter of $\mathcal{C}$ in the $\ell_p$-norm.*

- This result is independent of the ambient dimension $n$
- The bound is tight (Mirrokni et al., 2017)

# The approximate Carathéodory problem

**Problem**

Find $x \in \mathcal{C}$ with low cardinality satisfying $\|x - x^*\|_p \leqslant \varepsilon$.

- Applications in game theory, combinatorial optimization, etc.

**Theorem (Barman, 2015)**

*Let $p \in [2, +\infty[$. Then there exists $x \in \mathcal{C}$ with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$, where $D_p$ is the diameter of $\mathcal{C}$ in the $\ell_p$-norm.*

- This result is independent of the ambient dimension $n$
- The bound is tight (Mirrokni et al., 2017)
- Probabilistic proofs by Pisier (1981); Barman (2015)

# The approximate Carathéodory problem

**Problem**

Find $x \in \mathcal{C}$ with low cardinality satisfying $\|x - x^*\|_p \leqslant \varepsilon$.

- Applications in game theory, combinatorial optimization, etc.

**Theorem (Barman, 2015)**

*Let $p \in [2, +\infty[$. Then there exists $x \in \mathcal{C}$ with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$, where $D_p$ is the diameter of $\mathcal{C}$ in the $\ell_p$-norm.*

- This result is independent of the ambient dimension $n$
- The bound is tight (Mirrokni et al., 2017)
- Probabilistic proofs by Pisier (1981); Barman (2015)
- Deterministic proof by Mirrokni et al. (2017) using mirror descent (Nemirovsky & Yudin, 1983) on the dual problem

# The approximate Carathéodory problem

**Problem**

Find $x \in \mathcal{C}$ with low cardinality satisfying $\|x - x^*\|_p \leqslant \varepsilon$.

- Applications in game theory, combinatorial optimization, etc.

**Theorem (Barman, 2015)**

*Let $p \in [2, +\infty[$. Then there exists $x \in \mathcal{C}$ with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$, where $D_p$ is the diameter of $\mathcal{C}$ in the $\ell_p$-norm.*

- This result is independent of the ambient dimension $n$
- The bound is tight (Mirrokni et al., 2017)
- Probabilistic proofs by Pisier (1981); Barman (2015)
- Deterministic proof by Mirrokni et al. (2017) using mirror descent (Nemirovsky & Yudin, 1983) on the dual problem
- Can we solve $\min_{x \in \mathcal{C}} \|x - x^*\|_p$ by sequentially picking up vertices?

# Solving via Frank-Wolfe

**Lemma**

*Let $p \in [2, +\infty[$ and $f(x) = \frac{1}{2}\|x - x^*\|_p^2$. Then $f$ is convex, $(p-1)$-smooth, and 1-gradient dominated w.r.t. the $\ell_p$-norm.*

# Solving via Frank-Wolfe

**Lemma**

*Let $p \in [2, +\infty[$ and $f(x) = \frac{1}{2}\|x - x^*\|_p^2$. Then $f$ is convex, $(p-1)$-smooth, and 1-gradient dominated w.r.t. the $\ell_p$-norm.*

- If $p \in [2, +\infty[$, run FW on $\min_{x \in \mathcal{C}} \frac{1}{2}\|x - x^*\|_p^2$ and count the number of iterations to reach $\varepsilon^2/2$-convergence

# Solving via Frank-Wolfe

**Lemma**

*Let $p \in [2, +\infty[$ and $f(x) = \frac{1}{2}\|x - x^*\|_p^2$. Then $f$ is convex, $(p-1)$-smooth, and 1-gradient dominated w.r.t. the $\ell_p$-norm.*

- If $p \in [2, +\infty[$, run FW on $\min_{x \in \mathcal{C}} \frac{1}{2}\|x - x^*\|_p^2$ and count the number of iterations to reach $\varepsilon^2/2$-convergence

**Lemma**

*Let $p \in [1, 2[ \cup \{+\infty\}$ and $f(x) = \|x - x^*\|_p$. Then $f$ is convex and Lipschitz continuous w.r.t. the $\ell_2$-norm, with constant $n^{1/p-1/2}$ if $p \in [1, 2[$, else 1 if $p = +\infty$.*

# Solving via Frank-Wolfe

**Lemma**

*Let $p \in [2, +\infty[$ and $f(x) = \frac{1}{2}\|x - x^*\|_p^2$. Then $f$ is convex, $(p-1)$-smooth, and 1-gradient dominated w.r.t. the $\ell_p$-norm.*

- If $p \in [2, +\infty[$, run FW on $\min_{x \in \mathcal{C}} \frac{1}{2}\|x - x^*\|_p^2$ and count the number of iterations to reach $\varepsilon^2/2$-convergence

**Lemma**

*Let $p \in [1, 2[ \cup \{+\infty\}$ and $f(x) = \|x - x^*\|_p$. Then $f$ is convex and Lipschitz continuous w.r.t. the $\ell_2$-norm, with constant $n^{1/p-1/2}$ if $p \in [1, 2[$, else 1 if $p = +\infty$.*

- If $p \in [1, 2[ \cup \{+\infty\}$, run HCGS on $\min_{x \in \mathcal{C}} \|x - x^*\|_p$ and count the number of iterations to reach $\varepsilon$-convergence

# The Hybrid Conditional Gradient-Smoothing algorithm

Smoothen the problem (Argyriou et al., 2014) by taking the Moreau envelope (Moreau, 1965):

# The Hybrid Conditional Gradient-Smoothing algorithm

Smoothen the problem (Argyriou et al., 2014) by taking the Moreau envelope (Moreau, 1965):

$$f_\beta(x) = \min_{y \in \mathbb{R}^n} f(y) + \frac{1}{2\beta}\|x - y\|_2^2$$

# The Hybrid Conditional Gradient-Smoothing algorithm

Smoothen the problem (Argyriou et al., 2014) by taking the Moreau envelope (Moreau, 1965):

$$f_\beta(x) = \min_{y \in \mathbb{R}^n} f(y) + \frac{1}{2\beta}\|x - y\|_2^2 \quad \text{and} \quad \nabla f_\beta(x) = \frac{1}{\beta}(x - \text{prox}_{\beta f}(x))$$

# The Hybrid Conditional Gradient-Smoothing algorithm

Smoothen the problem (Argyriou et al., 2014) by taking the Moreau envelope (Moreau, 1965):

$$f_\beta(x) = \min_{y \in \mathbb{R}^n} f(y) + \frac{1}{2\beta}\|x - y\|_2^2 \quad \text{and} \quad \nabla f_\beta(x) = \frac{1}{\beta}(x - \text{prox}_{\beta f}(x))$$

**Algorithm** Hybrid Conditional Gradient-Smoothing (HCGS)

**Input:** $x_0 \in \mathcal{C}$, $G_2, D_2$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad \beta_t \leftarrow 2(D_2/G_2)/\sqrt{t+2}$
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f_{\beta_t}(x_t) \rangle$
4: $\quad \gamma_t \leftarrow 2/(t+2)$
5: $\quad x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$

# The Hybrid Conditional Gradient-Smoothing algorithm

Smoothen the problem (Argyriou et al., 2014) by taking the Moreau envelope (Moreau, 1965):

$$f_\beta(x) = \min_{y \in \mathbb{R}^n} f(y) + \frac{1}{2\beta}\|x - y\|_2^2 \quad \text{and} \quad \nabla f_\beta(x) = \frac{1}{\beta}(x - \text{prox}_{\beta f}(x))$$

---

**Algorithm** Hybrid Conditional Gradient-Smoothing (HCGS)

---

**Input:** $x_0 \in \mathcal{C}$, $G_2, D_2$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad \beta_t \leftarrow 2(D_2/G_2)/\sqrt{t+2}$
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f_{\beta_t}(x_t) \rangle$
4: $\quad \gamma_t \leftarrow 2/(t+2)$
5: $\quad x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$

---

# The Hybrid Conditional Gradient-Smoothing algorithm

Smoothen the problem (Argyriou et al., 2014) by taking the Moreau envelope (Moreau, 1965):

$$f_\beta(x) = \min_{y \in \mathbb{R}^n} f(y) + \frac{1}{2\beta}\|x - y\|_2^2 \quad \text{and} \quad \nabla f_\beta(x) = \frac{1}{\beta}(x - \text{prox}_{\beta f}(x))$$

---

**Algorithm** Hybrid Conditional Gradient-Smoothing (HCGS)

---

**Input:** $x_0 \in \mathcal{C}$, $G_2, D_2$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad \beta_t \leftarrow 2(D_2/G_2)/\sqrt{t+2}$
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f_{\beta_t}(x_t) \rangle$
4: $\quad \gamma_t \leftarrow 2/(t+2)$
5: $\quad x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$

---

# The Hybrid Conditional Gradient-Smoothing algorithm

Smoothen the problem (Argyriou et al., 2014) by taking the Moreau envelope (Moreau, 1965):

$$f_\beta(x) = \min_{y \in \mathbb{R}^n} f(y) + \frac{1}{2\beta}\|x - y\|_2^2 \quad \text{and} \quad \nabla f_\beta(x) = \frac{1}{\beta}(x - \operatorname{prox}_{\beta f}(x))$$

**Algorithm** Hybrid Conditional Gradient-Smoothing (HCGS)

**Input:** $x_0 \in \mathcal{C}$, $G_2, D_2$

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad \beta_t \leftarrow 2(D_2/G_2)/\sqrt{t+2}$
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f_{\beta_t}(x_t) \rangle$
4: $\quad \gamma_t \leftarrow 2/(t+2)$
5: $\quad x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$

**Lemma (Argyriou et al., 2014)**

*For all $\beta \geqslant \beta' > 0$, $f_\beta \leqslant f \leqslant f_\beta + \beta G_2^2/2$ and $f_\beta \leqslant f_{\beta'} \leqslant f_\beta + (\beta - \beta')G_2^2/2$.*

# The Hybrid Conditional Gradient-Smoothing algorithm

**Theorem (Argyriou et al., 2014)**

*Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex $G_2$-Lipschitz continuous function w.r.t. the $\ell_2$-norm. Then*

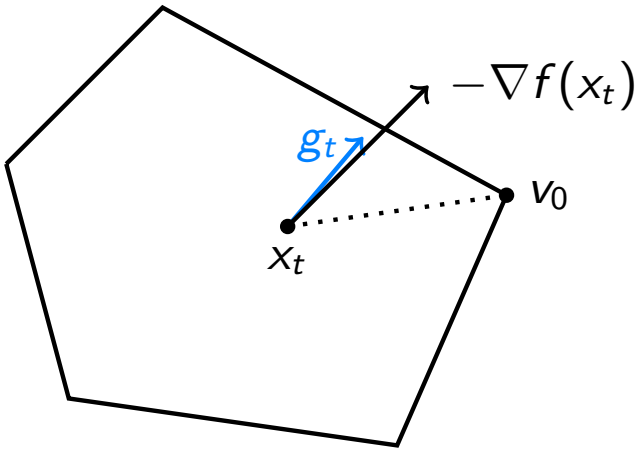
$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{4 G_2 D_2}{\sqrt{t+1}}.$$

# Cardinality bounds

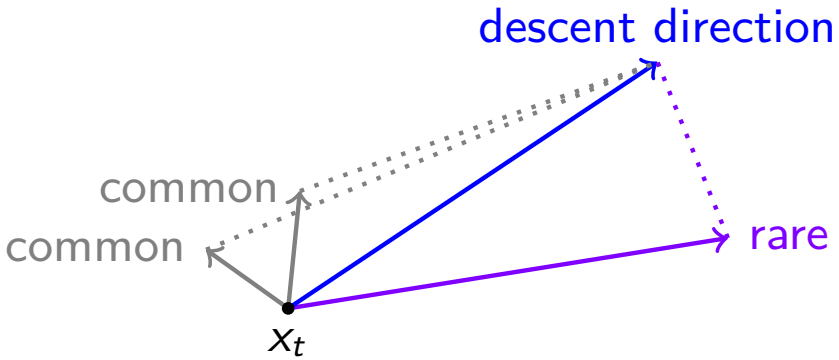
| $p$ | **Assumption** | **Cardinality bound** | |
|---|---|---|---|
| | | Via Frank-Wolfe | Related work |
| $[2, +\infty[$ | – | $\mathcal{O}\left(\frac{pD_p^2}{\varepsilon^2}\right)$ | $\mathcal{O}\left(\frac{pD_p^2}{\varepsilon^2}\right)$ |
| | $x^* \in \text{int}(\mathcal{C})$ | $\mathcal{O}\left(p\left(\frac{D_p}{r_p}\right)^2 \ln\left(\frac{1}{\varepsilon}\right)\right)$ | $\mathcal{O}\left(p\left(\frac{D_p}{r_p}\right)^2 \ln\left(\frac{1}{\varepsilon}\right)\right)$ |
| | $\mathcal{C}$ strongly convex | $\mathcal{O}\left(\frac{\sqrt{p}D_p + p/\alpha_p}{\varepsilon}\right)$ | – |
| $]1, 2[$ | – | $\mathcal{O}\left(\frac{n^{(2-p)/p}D_2^2}{\varepsilon^2}\right)$ | $\mathcal{O}\left(\frac{D_p^{p/(p-1)}}{p^{1/(p-1)}\varepsilon^{p/(p-1)}}\right)$ |
| $1$ | – | $\mathcal{O}\left(\frac{nD_2^2}{\varepsilon^2}\right)$ | – |
| $+\infty$ | – | $\mathcal{O}\left(\frac{D_2^2}{\varepsilon^2}\right)$ | $\mathcal{O}\left(\frac{\ln(n)D_\infty^2}{\varepsilon^2}\right)$ |

# Conclusion

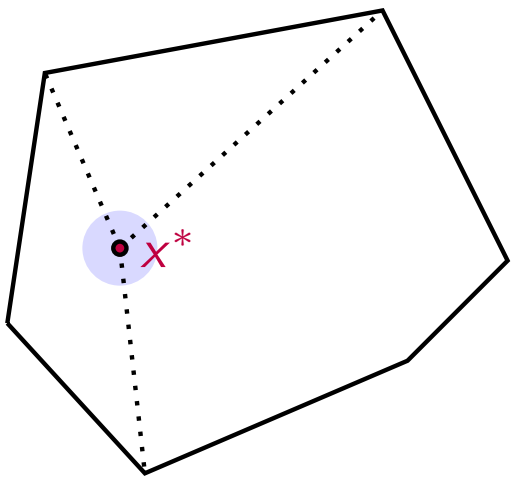
**Boosted Frank-Wolfe**




**Frank-Wolfe & adaptive gradients**




**Approximate Carathéodory**



# References (1/4)


A. Argyriou, M. Signoretto, and J. A. K. Suykens. Hybrid conditional gradient-smoothing algorithms with applications to sparse and low rank regularization. Chapman & Hall/CRC, 2014

S. Barman. Approximating Nash equilibria and dense bipartite subgraphs via an approximate version of Carathéodory's theorem. *STOC*, 2015

G. Braun, S. Pokutta, D. Tu, and S. Wright. Blended conditional gradients: the unconditioning of conditional gradients. *ICML*, 2019

M. D. Canon and C. D. Cullum. A tight upper bound on the rate of convergence of Frank-Wolfe algorithm. *SIAM J. Control*, 1968

**C. W. Combettes and S. Pokutta. Blended matching pursuit. *NeurIPS*, 2019**

**C. W. Combettes and S. Pokutta. Revisiting the approximate Carathéodory problem via the Frank-Wolfe algorithm. Minor revision at *Math. Program.*, 2021**

**C. W. Combettes and S. Pokutta. Boosting Frank-Wolfe by chasing gradients. *ICML*, 2020**

**C. W. Combettes and S. Pokutta. Complexity of linear minimization and projection on some sets. Minor revision at *Oper. Res. Lett.*, 2021**

**C. W. Combettes, C. Spiegel, and S. Pokutta. Projection-free adaptive gradients for large-scale optimization. *Submitted*, 2021**

L. Condat. Fast projection onto the simplex and the $\ell_1$ ball. *Math. Program.*, 2016

# References (2/4)


V. F. Demyanov and A. M. Rubinov. *Approximate Methods in Optimization Problems*. Elsevier, 1970

J. C. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 2011

J. C. Dunn and S. Harshbarger. Conditional gradient algorithms with open loop step size rules. *J. Math. Anal. Appl.*, 1978

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Res. Logist. Q.*, 1956

D. Garber and O. Meshi. Linear-memory and decomposition-invariant linearly convergent conditional gradient algorithm for structured polytopes. *NIPS*, 2016

E. Hazan and H. Luo. Variance-reduced and projection-free stochastic optimization. *ICML*, 2016

J. Guélat and P. Marcotte. Some comments on Wolfe's 'away step'. *Math. Program.*, 1986

Y. Haugazeau. *Sur les Inéquations Variationnelles et la Minimisation de Fonctionnelles Convexes*. Thèse de doctorat, Univ. Paris, 1968

M. Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. *ICML*, 2013

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015

S. Lacoste-Julien and M. Jaggi. On the global linear convergence of Frank-Wolfe optimization variants. *NIPS*, 2015

# References (3/4)


G. Lan. The complexity of large-scale convex programming under a linear optimization oracle. *arXiv*, 2013

G. Lan and Y. Zhou. Conditional gradient sliding for convex optimization. *SIAM J. Optim.*, 2016

E. S. Levitin and B. T. Polyak. Constrained minimization methods. *USSR Comp. Math. Math. Phys.*, 1966

F. Locatello, M. Tschannen, G. Rätsch, and M. Jaggi. Greedy algorithms for cone constrained optimization with convergence guarantees. *NIPS*, 2017

H. B. McMahan and M. Streeter. Adaptive bound optimization for online convex optimization. *COLT*, 2010

V. Mirrokni, R. Paes Leme, A. Vladu, and S. C.-W. Wong. Tight bounds for approximate Carathéodory and beyond. *ICML*, 2017

J. J. Moreau. Proximité et dualité dans un espace hilbertien. *Bull. Soc. Math. France*, 1965

G. Négiar, G. Dresdner, A. Y.-T. Tsai, L. El Ghaoui, F. Locatello, R. M. Freund, and F. Pedregosa. Stochastic Frank-Wolfe for constrained finite-sum minimization. *ICML*, 2020

A. S. Nemirovsky and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley, 1983

G. Pisier. Remarques sur un résultat non publié de B. Maurey. *Ec. polytech.*, 1981

S. J. Reddi, S. Kale, and S. Kumar. On the convergence of Adam and beyond. *ICLR*, 2018

# References (4/4)


Z. Shen, C. Fang, P. Zhao, J. Huang, and H. Qian. Complexities in projection-free stochastic non-convex minimization. *AISTATS*, 2019

P. Wolfe. Convergence theory in nonlinear programming. *Integer and Nonlinear Programming*. North-Holland, 1970

J. Xie, Z. Shen, C. Zhang, H. Qian, and B. Wang. Efficient projection-free online methods with stochastic recursive gradient. *AAAI*, 2020

A. Yurtsever, S. Sra, and V. Cevher. Conditional gradient methods via stochastic path-integrated differential estimator. *ICML*, 2019

# Stochastic Frank-Wolfe algorithms

| **Algorithm** | **Update** $\tilde{\nabla} f(x_t)$ |
|---|---|
| SFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$ |
| SVRF | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))$ |
| SPIDER-FW | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(x_{t-1}))$ |
| ORGFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) + (1-\rho_t)\left(\tilde{\nabla} f(x_{t-1}) - \frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_{t-1})\right)$ |
| CSFW | $\tilde{\nabla} f(x_{t-1}) + \sum_{j=1}^{b_t}\left(\frac{1}{m} f'_{i_j}(\langle a_{i_j}, x_t\rangle) - [\alpha_{t-1}]_{i_j}\right) a_{i_j}$ |
| | and $[\alpha_t]_{i_j} \leftarrow \begin{cases} (1/m) f'_{i_j}(\langle a_{i_j}, x_t\rangle) & \text{if } i_j \in \{i_1, \ldots, i_{b_t}\} \\ [\alpha_{t-1}]_{i_j} & \text{else} \end{cases}$ |

# Stochastic Frank-Wolfe algorithms

| **Algorithm** | **Update** $\tilde{\nabla} f(x_t)$ |
|---|---|
| SFW | $\frac{1}{b_t}\sum_{j=1}^{b_t}\nabla f_{i_j}(x_t)$ |
| SVRF | $\nabla f(\tilde{x}_t)+\frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t)-\nabla f_{i_j}(\tilde{x}_t))$ |
| SPIDER-FW | $\nabla f(\tilde{x}_t)+\frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t)-\nabla f_{i_j}(x_{t-1}))$ |
| ORGFW | $\frac{1}{b_t}\sum_{j=1}^{b_t}\nabla f_{i_j}(x_t)+(1-\rho_t)\left(\tilde{\nabla} f(x_{t-1})-\frac{1}{b_t}\sum_{j=1}^{b_t}\nabla f_{i_j}(x_{t-1})\right)$ |
| CSFW | $\tilde{\nabla} f(x_{t-1})+\sum_{j=1}^{b_t}\left(\frac{1}{m}f_{i_j}'(\langle a_{i_j},x_t\rangle)-[\alpha_{t-1}]_{i_j}\right)a_{i_j}$ |
| | and $[\alpha_t]_{i_j}\leftarrow\begin{cases}(1/m)f_{i_j}'(\langle a_{i_j},x_t\rangle) & \text{if } i_j\in\{i_1,\dots,i_{b_t}\}\\ [\alpha_{t-1}]_{i_j} & \text{else}\end{cases}$ |

# Stochastic Frank-Wolfe algorithms

| **Algorithm** | **Update** $\tilde{\nabla} f(x_t)$ |
|---|---|
| SFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$ |
| SVRF | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))$ |
| SPIDER-FW | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(x_{t-1}))$ |
| ORGFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) + (1-\rho_t)\left(\tilde{\nabla} f(x_{t-1}) - \frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_{t-1})\right)$ |
| CSFW | $\tilde{\nabla} f(x_{t-1}) + \sum_{j=1}^{b_t}\left(\frac{1}{m} f'_{i_j}(\langle a_{i_j}, x_t\rangle) - [\alpha_{t-1}]_{i_j}\right) a_{i_j}$ |
| | and $[\alpha_t]_{i_j} \leftarrow \begin{cases} (1/m) f'_{i_j}(\langle a_{i_j}, x_t\rangle) & \text{if } i_j \in \{i_1, \ldots, i_{b_t}\} \\ [\alpha_{t-1}]_{i_j} & \text{else} \end{cases}$ |

# Stochastic Frank-Wolfe algorithms

| **Algorithm** | **Update** $\tilde{\nabla} f(x_t)$ |
|---|---|
| SFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$ |
| SVRF | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))$ |
| SPIDER-FW | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(x_{t-1}))$ |
| ORGFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) + (1-\rho_t)\left(\tilde{\nabla} f(x_{t-1}) - \frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_{t-1})\right)$ |
| CSFW | $\tilde{\nabla} f(x_{t-1}) + \sum_{j=1}^{b_t}\left(\frac{1}{m} f'_{i_j}(\langle a_{i_j}, x_t\rangle) - [\alpha_{t-1}]_{i_j}\right) a_{i_j}$ |
| | and $[\alpha_t]_{i_j} \leftarrow \begin{cases} (1/m) f'_{i_j}(\langle a_{i_j}, x_t\rangle) & \text{if } i_j \in \{i_1, \ldots, i_{b_t}\} \\ [\alpha_{t-1}]_{i_j} & \text{else} \end{cases}$ |

# Stochastic Frank-Wolfe algorithms

| **Algorithm** | **Update** $\tilde{\nabla} f(x_t)$ |
|---|---|
| SFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$ |
| SVRF | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))$ |
| SPIDER-FW | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(x_{t-1}))$ |
| ORGFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) + (1-\rho_t)\left(\tilde{\nabla} f(x_{t-1}) - \frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_{t-1})\right)$ |
| CSFW | $\tilde{\nabla} f(x_{t-1}) + \sum_{j=1}^{b_t}\left(\frac{1}{m} f'_{i_j}(\langle a_{i_j}, x_t\rangle) - [\alpha_{t-1}]_{i_j}\right) a_{i_j}$ |
| | and $[\alpha_t]_{i_j} \leftarrow \begin{cases} (1/m) f'_{i_j}(\langle a_{i_j}, x_t\rangle) & \text{if } i_j \in \{i_1, \ldots, i_{b_t}\} \\ [\alpha_{t-1}]_{i_j} & \text{else} \end{cases}$ |

# Stochastic Frank-Wolfe algorithms

| **Algorithm** | **Update** $\tilde{\nabla} f(x_t)$ |
|---|---|
| SFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$ |
| SVRF | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))$ |
| SPIDER-FW | $\nabla f(\tilde{x}_t) + \frac{1}{b_t}\sum_{j=1}^{b_t}(\nabla f_{i_j}(x_t) - \nabla f_{i_j}(x_{t-1}))$ |
| ORGFW | $\frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) + (1-\rho_t)\left(\tilde{\nabla} f(x_{t-1}) - \frac{1}{b_t}\sum_{j=1}^{b_t} \nabla f_{i_j}(x_{t-1})\right)$ |
| CSFW | $\tilde{\nabla} f(x_{t-1}) + \sum_{j=1}^{b_t}\left(\frac{1}{m} f'_{i_j}(\langle a_{i_j}, x_t\rangle) - [\alpha_{t-1}]_{i_j}\right) a_{i_j}$ |
| | and $[\alpha_t]_{i_j} \leftarrow \begin{cases} (1/m) f'_{i_j}(\langle a_{i_j}, x_t\rangle) & \text{if } i_j \in \{i_1, \ldots, i_{b_t}\} \\ [\alpha_{t-1}]_{i_j} & \text{else} \end{cases}$ |

# A lower bound when $p \in [2, +\infty[$

**Theorem (Mirrokni et al., 2017)**

*Let $p \in [2, +\infty[$, $H_n \in \mathbb{R}^{n \times n}$ be the Hadamard matrix of dimension $n$, $\mathcal{C} = \mathrm{conv}(H_n / n^{1/p})$ be the convex hull of the $\ell_p$-normalized columns of $H_n$, and $x^* = e_1 / n^{1/p} \in \mathcal{C}$. Then for all $x \in \mathcal{C}$,*

$$\|x - x^*\|_p \leqslant \varepsilon \Rightarrow \mathrm{card}(x) \geqslant \frac{1}{\varepsilon^2 + 1/n}.$$

# A lower bound when $p \in [2, +\infty[$

**Theorem (Mirrokni et al., 2017)**

*Let $p \in [2, +\infty[$, $H_n \in \mathbb{R}^{n \times n}$ be the Hadamard matrix of dimension $n$, $\mathcal{C} = \text{conv}(H_n / n^{1/p})$ be the convex hull of the $\ell_p$-normalized columns of $H_n$, and $x^* = e_1 / n^{1/p} \in \mathcal{C}$. Then for all $x \in \mathcal{C}$,*

$$\|x - x^*\|_p \leqslant \varepsilon \Rightarrow \text{card}(x) \geqslant \frac{1}{\varepsilon^2 + 1/n}.$$

- $H_n \in \mathbb{R}^{n \times n}$ is a Hadamard matrix if $H_n \in \{\pm 1\}^{n \times n}$ and $H_n^\top H_n = n I_n$

# A lower bound when $p \in [2, +\infty[$

**Theorem (Mirrokni et al., 2017)**

*Let $p \in [2, +\infty[$, $H_n \in \mathbb{R}^{n \times n}$ be the Hadamard matrix of dimension $n$, $\mathcal{C} = \mathrm{conv}(H_n / n^{1/p})$ be the convex hull of the $\ell_p$-normalized columns of $H_n$, and $x^* = e_1 / n^{1/p} \in \mathcal{C}$. Then for all $x \in \mathcal{C}$,*

$$\|x - x^*\|_p \leqslant \varepsilon \Rightarrow \mathrm{card}(x) \geqslant \frac{1}{\varepsilon^2 + 1/n}.$$

- $H_n \in \mathbb{R}^{n \times n}$ is a Hadamard matrix if $H_n \in \{\pm 1\}^{n \times n}$ and $H_n^\top H_n = n I_n$
- Sylvester's construction:

$$H_{2n} = \begin{pmatrix} H_n & H_n \\ H_n & -H_n \end{pmatrix}$$

# A lower bound when $p \in [2, +\infty[$

**Theorem (Mirrokni et al., 2017)**

*Let $p \in [2, +\infty[$, $H_n \in \mathbb{R}^{n \times n}$ be the Hadamard matrix of dimension $n$, $\mathcal{C} = \mathrm{conv}(H_n / n^{1/p})$ be the convex hull of the $\ell_p$-normalized columns of $H_n$, and $x^* = e_1 / n^{1/p} \in \mathcal{C}$. Then for all $x \in \mathcal{C}$,*

$$\|x - x^*\|_p \leqslant \varepsilon \Rightarrow \mathrm{card}(x) \geqslant \frac{1}{\varepsilon^2 + 1/n}.$$

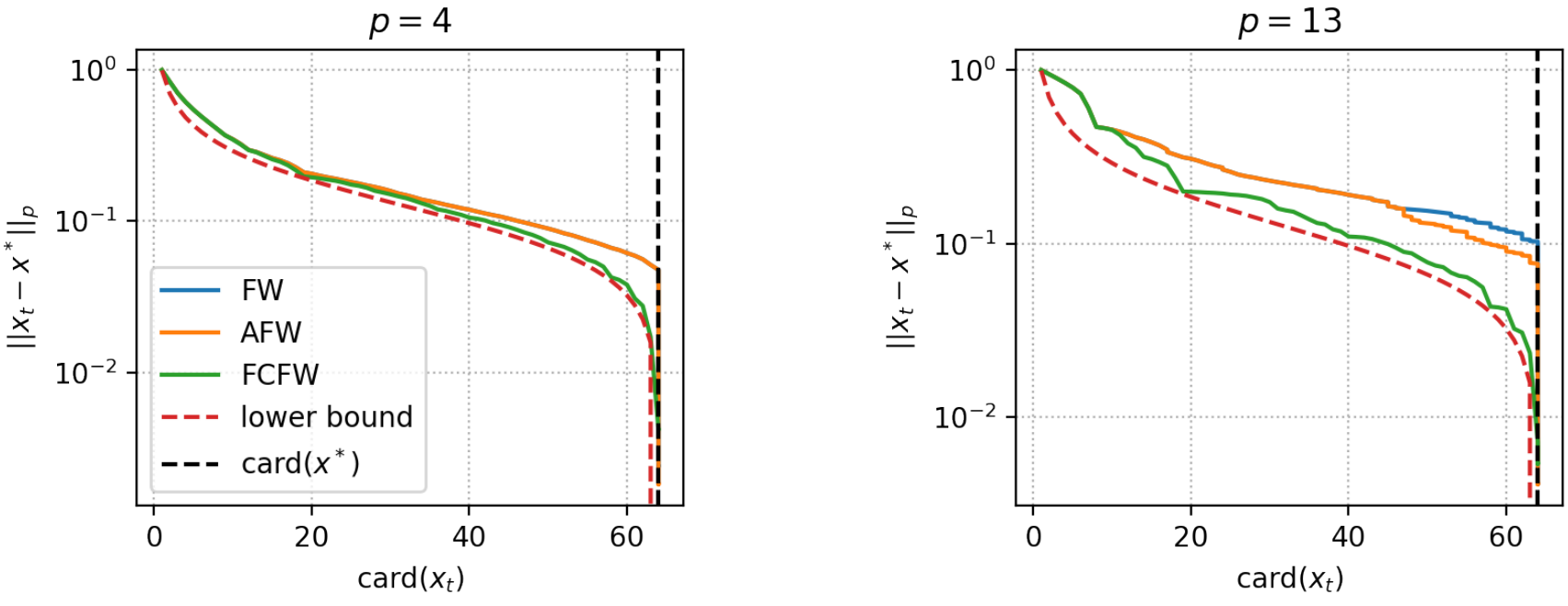
- $H_n \in \mathbb{R}^{n \times n}$ is a Hadamard matrix if $H_n \in \{\pm 1\}^{n \times n}$ and $H_n^\top H_n = n I_n$
- Sylvester's construction:

$$H_{2n} = \begin{pmatrix} H_n & H_n \\ H_n & -H_n \end{pmatrix}$$

gives

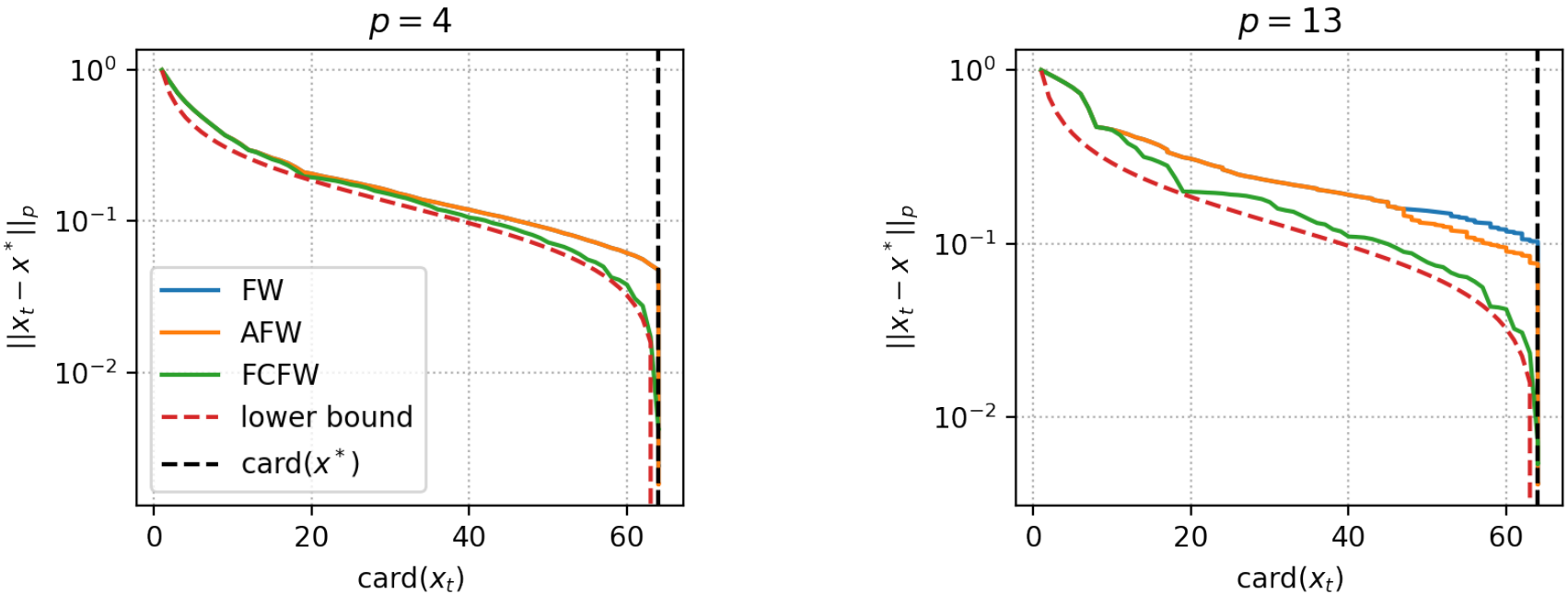
$$H_1 = (1), \quad H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad H_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

# A lower bound when $p \in [2, +\infty[$




- FCFW almost matches the lower bound

# A lower bound when $p \in [2, +\infty[$




- FCFW almost matches the lower bound
- There is no precise analysis of FCFW: the current analysis is transferred from that of AFW (Lacoste-Julien & Jaggi, 2015) and holds only for smooth strongly convex functions