Frank-Wolfe Algorithms for Saddle Point problems

Gauthier Gidel\textsuperscript{1,3} \hspace{1cm} Tony Jebara\textsuperscript{2} \hspace{1cm} Simon Lacoste-Julien\textsuperscript{3}

\textsuperscript{1}INRIA Paris, Sierra Team \hspace{1cm} \textsuperscript{2}Department of CS, Columbia University \hspace{1cm} \textsuperscript{3}Department of CS & OR (DIRO) Université de Montréal

25th May 2017
Overview

- Frank-Wolfe algorithm (FW) gained in popularity in the last couple of years.
- Main advantage: FW only needs LMO.
- Extend FW properties to solve saddle point problem\textsuperscript{1}.
- **Straightforward** extension but **Non trivial** analysis.

Frank-Wolfe algorithm (FW) gained in popularity in the last couple of years.

Main advantage: FW only needs LMO.

Extend FW properties to solve saddle point problem\(^1\).

**Straightforward** extension but **Non trivial** analysis.

---

Saddle point and link with variational inequalities

Let $\mathcal{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, where $\mathcal{X}$ and $\mathcal{Y}$ are convex and compact.

Saddle point problem: solve $\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathcal{L}(x, y)$

A solution $(x^*, y^*)$ is called a **Saddle Point**.
Saddle point and link with variational inequalities

Let $\mathcal{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, where $\mathcal{X}$ and $\mathcal{Y}$ are convex and compact.

Saddle point problem: solve $\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathcal{L}(x, y)$

A solution $(x^*, y^*)$ is called a Saddle Point.

- **Necessary stationary conditions:**

  \[ \langle x - x^*, \nabla_x \mathcal{L}(x^*, y^*) \rangle \geq 0 \]
Saddle point and link with variational inequalities

Let $\mathcal{L} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$, where $\mathcal{X}$ and $\mathcal{Y}$ are convex and compact.

| Saddle point problem: solve $\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathcal{L}(x, y)$ |

A solution $(x^*, y^*)$ is called a **Saddle Point**.

- **Necessary stationary conditions:**
  
  $\langle x - x^*, \nabla_x \mathcal{L}(x^*, y^*) \rangle \geq 0$
  $\langle y - y^*, -\nabla_y \mathcal{L}(x^*, y^*) \rangle \geq 0$
Saddle point and link with variational inequalities

Let $L : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$, where $\mathcal{X}$ and $\mathcal{Y}$ are convex and compact.

Saddle point problem: solve $\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} L(x, y)$

A solution $(x^*, y^*)$ is called a Saddle Point.

- **Necessary stationary conditions:**

  \[
  \langle x - x^*, \nabla_x L(x^*, y^*) \rangle \geq 0 \\
  \langle y - y^*, -\nabla_y L(x^*, y^*) \rangle \geq 0
  \]

- **Variational inequality:**

  \[
  \forall z \in \mathcal{X} \times \mathcal{Y} \quad \langle z - z^*, g(z^*) \rangle \geq 0
  \]

  where $(x^*, y^*) = z^*$ and $g(z) = (\nabla_x L(z), -\nabla_y L(z))$
Saddle point and link with variational inequalities

Let $\mathcal{L} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, where $\mathcal{X}$ and $\mathcal{Y}$ are convex and compact.

Saddle point problem: solve $\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \mathcal{L}(x, y)$

A solution $(x^*, y^*)$ is called a Saddle Point.

▶ **Necessary stationary conditions:**

\[
\langle x - x^*, \nabla_x \mathcal{L}(x^*, y^*) \rangle \geq 0 \\
\langle y - y^*, -\nabla_y \mathcal{L}(x^*, y^*) \rangle \geq 0
\]

▶ **Variational inequality:**

$$
\forall z \in \mathcal{X} \times \mathcal{Y} \quad \langle z - z^*, g(z^*) \rangle \geq 0
$$

where $(x^*, y^*) = z^*$ and $g(z) = (\nabla_x \mathcal{L}(z), -\nabla_y \mathcal{L}(z))$

▶ **Sufficient condition:** Global solution if $\mathcal{L}$
convex-concave. $\forall (x, y) \in \mathcal{X} \times \mathcal{Y}$

\[x' \mapsto \mathcal{L}(x', y)\] is convex and $y' \mapsto \mathcal{L}(x, y')$ is concave.
Motivations: games and robust learning

- **Zero-sum games with two players:**

\[
\min_{x \in \Delta(I)} \max_{y \in \Delta(J)} x^\top My
\]

Robust learning:

\[
\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(f_{\theta}(x_i), y_i) + \lambda \Omega(\theta)
\]

Minimize the worst case \( \rightarrow \) gives robustness

---

Motivations: games and robust learning

- **Zero-sum games with two players:**

  \[
  \min_{x \in \Delta(I)} \max_{y \in \Delta(J)} x^\top My
  \]

- **Robust learning:**

  \[
  \min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(f_{\theta}(x_i), y_i) + \lambda \Omega(\theta)
  \]

  with an uncertainty regarding the data:

  \[
  \min_{\theta \in \Theta} \max_{w \in \Delta_n} \sum_{i=1}^{n} \omega_i \ell(f_{\theta}(x_i), y_i) + \lambda \Omega(\theta)
  \]

  Minimize the worst case \(\rightarrow\) gives robustness

---

Problem with Hard projection

The \textit{structured SVM}:

\[
\min_{\omega \in \mathbb{R}^d} \lambda \Omega(\omega) + \frac{1}{n} \sum_{i=1}^{n} \max_{y \in \mathcal{Y}_i} (L_i(y) - \langle \omega, \phi_i(y) \rangle)
\]

structured empirical loss
Problem with Hard projection

The *structured SVM*:

\[
\min_{\omega \in \mathbb{R}^d} \lambda \Omega(\omega) + \frac{1}{n} \sum_{i=1}^{n} \max_{y \in Y_i} (L_i(y) - \langle \omega, \phi_i(y) \rangle)
\]

structured empirical loss

Regularization: penalized $\rightarrow$ constrained.

\[
\min_{\Omega(\omega) \leq \beta} \max_{\alpha \in \Delta(|Y|)} b^T \alpha - \omega^T M \alpha
\]
Problem with Hard projection

The \textit{structured SVM}:

$$\min_{\omega \in \mathbb{R}^d} \lambda \Omega(\omega) + \frac{1}{n} \sum_{i=1}^{n} \max_{y \in Y_i} \left( L_i(y) - \langle \omega, \phi_i(y) \rangle \right)$$

structured empirical loss

Regularization: penalized $\rightarrow$ constrained.

$$\min_{\Omega(\omega) \leq \beta} \max_{\alpha \in \Delta(|Y|)} b^T \alpha - \omega^T M \alpha$$
Problem with Hard projection

The **structured SVM**:

\[
\min_{\omega \in \mathbb{R}^d} \lambda \Omega(\omega) + \frac{1}{n} \sum_{i=1}^{n} \max_{y \in \mathcal{Y}_i} \left( L_i(y) - \langle \omega, \phi_i(y) \rangle \right)
\]

structured empirical loss

Regularization: penalized $\rightarrow$ constrained.

\[
\min \max_{\Omega(\omega) \leq \beta} b^T \alpha - \omega^T M \alpha
\]
Problem with Hard projection

The *structured SVM*:

\[
\min_{\omega \in \mathbb{R}^d} \lambda \Omega(\omega) + \frac{1}{n} \sum_{i=1}^{n} \max_{y \in \mathcal{Y}_i} (L_i(y) - \langle \omega, \phi_i(y) \rangle)
\]

structured empirical loss

Regularization: penalized $\rightarrow$ constrained.

\[
\min_{\Omega(\omega) \leq \beta} \max_{\alpha \in \Delta(|\mathcal{Y}|)} b^T \alpha - \omega^T M \alpha
\]

Difficult to project when:

▶ Structured sparsity norm (group lasso norm).
▶ The output $\mathcal{Y}$ is structured: exponential size.
Problem with Hard projection

The structured SVM:

$$\min_{\omega \in \mathbb{R}^d} \lambda \Omega(\omega) + \frac{1}{n} \sum_{i=1}^{n} \max_{y \in \mathcal{Y}_i} \left( L_i(y) - \langle \omega, \phi_i(y) \rangle \right)$$

structured empirical loss

Regularization: penalized $\rightarrow$ constrained.

$$\min_{\Omega(\omega) \leq \beta} \max_{\alpha \in \Delta(|\mathcal{Y}|)} b^T \alpha - \omega^T M \alpha$$

Difficult to project when:

- **Structured sparsity** norm (group lasso norm).
Problem with Hard projection

The \textit{structured SVM}:

\[
\min_{\omega \in \mathbb{R}^d} \lambda \Omega(\omega) + \frac{1}{n} \sum_{i=1}^{n} \max_{y \in \mathcal{Y}_i} (L_i(y) - \langle \omega, \phi_i(y) \rangle)
\]

structured empirical loss

Regularization: penalized $\rightarrow$ constrained.

\[
\min \max_{\Omega(\omega) \leq \beta, \alpha \in \Delta(|\mathcal{Y}|)} b^T \alpha - \omega^T M \alpha
\]

Difficult to project when:

- \textit{Structured sparsity} norm (group lasso norm).
- The output $\mathcal{Y}$ is structured: \textit{exponential} size.
Standard approaches in literature

- Projected gradient algorithm.

\[
\begin{align*}
\mathbf{x}^{(t+1)} &= P_X(\mathbf{x}^{(t)} - \eta \nabla_x \mathcal{L}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})) \\
\mathbf{y}^{(t+1)} &= P_Y(\mathbf{y}^{(t)} + \eta \nabla_y \mathcal{L}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}))
\end{align*}
\]

Intuition: lookahead move: look at what your opponent would do before deciding your move.

\[
\begin{align*}
\mathbf{x}^{(t+1)} &= P_X(\mathbf{x}^{(t)} - \eta \nabla_x \mathcal{L}(\bar{\mathbf{x}}^{(t+1)}, \bar{\mathbf{y}}^{(t+1)})) \\
\mathbf{y}^{(t+1)} &= P_Y(\mathbf{y}^{(t)} + \eta \nabla_y \mathcal{L}(\bar{\mathbf{x}}^{(t+1)}, \bar{\mathbf{y}}^{(t+1)}))
\end{align*}
\]

Prevents oscillations for non strongly convex objective.

---

Standard approaches in literature

- Projected gradient algorithm.

\[
\begin{align*}
x^{(t+1)} &= P_X(x^{(t)} - \eta \nabla_x \mathcal{L}(x^{(t)}, y^{(t)})) \\
y^{(t+1)} &= P_Y(y^{(t)} + \eta \nabla_y \mathcal{L}(x^{(t)}, y^{(t)}))
\end{align*}
\]

- Projected extra-gradient\(^3\).

\[
\begin{align*}
\bar{x}^{(t+1)} &= P_X(x^{(t)} - \eta \nabla_x \mathcal{L}(x^{(t)}, y^{(t)})) \\
\bar{y}^{(t+1)} &= P_Y(y^{(t)} + \eta \nabla_y \mathcal{L}(x^{(t)}, y^{(t)}))
\end{align*}
\]

Intuition: **lookahead move**: look at what your opponent would do before deciding your move.

\[
\begin{align*}
x^{(t+1)} &= P_X(x^{(t)} - \eta \nabla_x \mathcal{L}(\bar{x}^{(t+1)}, \bar{y}^{(t+1)})) \\
y^{(t+1)} &= P_Y(y^{(t)} + \eta \nabla_y \mathcal{L}(\bar{x}^{(t+1)}, \bar{y}^{(t+1)}))
\end{align*}
\]

Prevents oscillations for non strongly convex objective.

Standard approaches in literature

- Gradient method works for non-smooth optimization, but

\[
\frac{1}{T} \sum_{t=1}^{T} (x^{(t)}, y^{(t)}) \xrightarrow{T \to \infty} (x^*, y^*)
\]

Even when projections are expensive:

Can use LMO to compute approximate projections.

---

Standard approaches in literature

- Gradient method works for non-smooth optimization, but
  \[ \frac{1}{T} \sum_{t=1}^{T} \left( x^{(t)}, y^{(t)} \right) \xrightarrow{T \to \infty} (x^*, y^*) \]

- Extragradient method works for smooth optimization,
  \( (x^{(t)}, y^{(t)}) \to (x^*, y^*) \)

---

\(^4\)N. He and Z. Harchaoui. “Semi-proximal Mirror-Prox for Nonsmooth Composite Minimization”. In: NIPS. 2015.
Standard approaches in literature

- Gradient method works for non-smooth optimization, but
  \[
  \frac{1}{T} \sum_{t=1}^{T} (x^{(t)}, y^{(t)}) \underset{T \to \infty}{\longrightarrow} (x^*, y^*)
  \]

- Extragradient method works for smooth optimization,
  \[
  (x^{(t)}, y^{(t)}) \to (x^*, y^*)
  \]

Even when projections are expensive:

Can use LMO to compute approximate projections\(^4\).

The FW algorithm

Algorithm Frank-Wolfe algorithm

1: Let $x^{(0)} \in \mathcal{X}$
2: for $t = 0 \ldots T$ do
3: Compute $r^{(t)} = \nabla f(x^{(t)})$
4: Compute $s^{(t)} \in \text{argmin}_{s \in \mathcal{X}} \langle s, r^{(t)} \rangle$
5: Compute $g_t := \langle x^{(t)} - s^{(t)}, r^{(t)} \rangle$
6: if $g_t \leq \epsilon$ then return $x^{(t)}$
7: Let $\gamma = \frac{2}{2+t}$ (or do line-search)
8: Update $x^{(t+1)} := (1-\gamma)x^{(t)} + \gamma s^{(t)}$
9: end for
The FW algorithm

Algorithm Frank-Wolfe algorithm

1: Let $x^{(0)} \in \mathcal{X}$
2: for $t = 0 \ldots T$ do
3: Compute $r^{(t)} = \nabla f(x^{(t)})$
4: Compute $s^{(t)} \in \operatorname{argmin}_{s \in \mathcal{X}} \langle s, r^{(t)} \rangle$
5: Compute $g_t := \langle x^{(t)} - s^{(t)}, r^{(t)} \rangle$
6: if $g_t \leq \epsilon$ then return $x^{(t)}$
7: Let $\gamma = \frac{2}{2 + t}$ (or do line-search)
8: Update $x^{(t+1)} := (1 - \gamma)x^{(t)} + \gamma s^{(t)}$
9: end for
The FW algorithm

**Algorithm** Frank-Wolfe algorithm

1: Let $x^{(0)} \in \mathcal{X}$
2: for $t = 0 \ldots T$ do
3: Compute $r^{(t)} = \nabla f(x^{(t)})$
4: Compute $s^{(t)} \in \arg\min_{s \in \mathcal{X}} \langle s, r^{(t)} \rangle$
5: Compute $g_t := \langle x^{(t)} - s^{(t)}, r^{(t)} \rangle$
6: if $g_t \leq \epsilon$ then return $x^{(t)}$
7: Let $\gamma = \frac{2}{2 + t}$ (or do line-search)
8: Update $x^{(t+1)} := (1 - \gamma)x^{(t)} + \gamma s^{(t)}$
9: end for

---

Gauthier Gidel
Frank-Wolfe Algorithms for SP
25th May 2017
The FW algorithm

**Algorithm** Frank-Wolfe algorithm

1: Let $x^{(0)} \in \mathcal{X}$
2: **for** $t = 0 \ldots T$ **do**
3: Compute $r^{(t)} = \nabla f(x^{(t)})$
4: Compute $s^{(t)} \in \arg\min_{s \in \mathcal{X}} \langle s, r^{(t)} \rangle$
5: Compute $g_t := \langle x^{(t)} - s^{(t)}, r^{(t)} \rangle$
6: **if** $g_t \leq \epsilon$ **then** return $x^{(t)}$
7: Let $\gamma = \frac{2}{2+t}$ (or do line-search)
8: Update $x^{(t+1)} := (1 - \gamma)x^{(t)} + \gamma s^{(t)}$
9: **end for**
Algorithm  Saddle point FW algorithm

1: for $t = 0 \ldots T$ do
2:   Compute $\mathbf{r}(t) := \left( \nabla_x \mathcal{L}(\mathbf{x}(t), \mathbf{y}(t)) \right. \left. - \nabla_y \mathcal{L}(\mathbf{x}(t), \mathbf{y}(t)) \right)$
3:   Compute $\mathbf{s}(t) \in \text{argmin}_{z \in \mathcal{X} \times \mathcal{Y}} \langle z, \mathbf{r}(t) \rangle$
4:   Compute $g_t := \langle \mathbf{z}(t) - \mathbf{s}(t), \mathbf{r}(t) \rangle$
5:   if $g_t \leq \epsilon$ then return $\mathbf{z}(t)$
6:   Let $\gamma = \min \left( 1, \frac{\nu}{C} g_t \right)$ or $\gamma = \frac{2}{2+t}$
7:   Update $\mathbf{z}(t+1) := (1 - \gamma)\mathbf{z}(t) + \gamma\mathbf{s}(t)$
8: end for

Originally proposed by Hammond\(^4\) with $\gamma_t = 1/(t + 1)$.

---

Algorithm  Saddle point FW algorithm

1:  \textbf{for} $t = 0 \ldots T$ \textbf{do} \\
2: \hspace{1em} Compute $r^{(t)} := \left( \nabla_x \mathcal{L}(x^{(t)}, y^{(t)}) \right) \left( \begin{array}{c} -\nabla_y \mathcal{L}(x^{(t)}, y^{(t)}) \end{array} \right)$ \\
3: \hspace{1em} Compute $s^{(t)} \in \arg\min_{z \in X \times Y} \langle z, r^{(t)} \rangle$ \\
4: \hspace{1em} Compute $g_t := \langle z^{(t)} - s^{(t)}, r^{(t)} \rangle$ \\
5: \hspace{1em} \textbf{if} $g_t \leq \epsilon$ \textbf{then return} $z^{(t)}$ \\
6: \hspace{1em} Let $\gamma = \min \left( 1, \frac{\nu}{C} g_t \right)$ or $\gamma = \frac{2}{2+t}$ \\
7: \hspace{1em} Update $z^{(t+1)} := (1 - \gamma) z^{(t)} + \gamma s^{(t)}$ \\
8: \hspace{1em} \textbf{end for}

\begin{itemize}
  \item Originally proposed by Hammond\textsuperscript{4} with $\gamma_t = 1/(t + 1)$.
\end{itemize}

---

SP-FW

Algorithm  Saddle point FW algorithm

1: for $t = 0 \ldots T$ do
2:  Compute $r(t) := \begin{pmatrix} \nabla_x L(x(t), y(t)) \\ -\nabla_y L(x(t), y(t)) \end{pmatrix}$
3:  Compute $s(t) \in \text{argmin}_{z \in X \times Y} \langle z, r(t) \rangle$
4:  Compute $g_t := \langle z(t) - s(t), r(t) \rangle$
5:  if $g_t \leq \epsilon$ then return $z(t)$
6:  Let $\gamma = \min(1, \frac{\nu}{C} g_t)$ or $\gamma = \frac{2}{2+t}$
7:  Update $z(t+1) := (1 - \gamma)z(t) + \gamma s(t)$
8: end for

Originally proposed by Hammond\textsuperscript{4} with $\gamma_t = 1/(t + 1)$.

---

Algorithm Saddle point FW algorithm

1: for \( t = 0 \ldots T \) do
2: \hspace{1em} Compute \( r(t) := \left( \nabla_x L(x(t), y(t)), -\nabla_y L(x(t), y(t)) \right) \)
3: \hspace{1em} Compute \( s(t) \in \text{argmin}_{z \in \mathcal{X} \times \mathcal{Y}} \langle z, r(t) \rangle \)
4: \hspace{1em} Compute \( g_t := \langle z(t) - s(t), r(t) \rangle \)
5: \hspace{1em} if \( g_t \leq \epsilon \) then return \( z(t) \)
6: \hspace{1em} Let \( \gamma = \min \left( 1, \frac{\nu}{C} g_t \right) \) or \( \gamma = \frac{2}{2 + t} \)
7: \hspace{1em} Update \( z(t+1) := (1 - \gamma)z(t) + \gamma s(t) \)
8: end for

Originally proposed by Hammond\(^4\) with
\( \gamma_t = 1/(t + 1) \).

Algorithm Saddle point FW algorithm

1: for $t = 0 \ldots T$ do
2: \hspace{1em} Compute $r^{(t)} := \begin{pmatrix} \nabla_x \mathcal{L}(x^{(t)}, y^{(t)}) \\ -\nabla_y \mathcal{L}(x^{(t)}, y^{(t)}) \end{pmatrix}$
3: \hspace{1em} Compute $s^{(t)} \in \arg\min_{z \in X \times Y} \langle z, r^{(t)} \rangle$
4: \hspace{1em} Compute $g_t := \langle z^{(t)} - s^{(t)}, r^{(t)} \rangle$
5: \hspace{1em} if $g_t \leq \epsilon$ then return $z^{(t)}$
6: \hspace{1em} Let $\gamma = \min\left(1, \frac{\nu}{C} g_t\right)$ or $\gamma = \frac{2}{2+t}$
7: \hspace{1em} Update $z^{(t+1)} := (1 - \gamma)z^{(t)} + \gamma s^{(t)}$
8: end for

Originally proposed by Hammond\(^4\) with $\gamma_t = 1/(t + 1)$.

Algorithm Saddle point FW algorithm

1: for $t = 0 \ldots T$ do 
2: Compute $\mathbf{r}^{(t)} := \left( \nabla_x \mathcal{L}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \right.$
3: Compute $\mathbf{s}^{(t)} \in \mathop{\arg\min}_{\mathbf{z} \in X \times Y} \langle \mathbf{z}, \mathbf{r}^{(t)} \rangle$
4: Compute $g_t := \langle \mathbf{z}^{(t)} - \mathbf{s}^{(t)}, \mathbf{r}^{(t)} \rangle$
5: if $g_t \leq \epsilon$ then return $\mathbf{z}^{(t)}$
6: Let $\gamma = \min \left( 1, \frac{\nu}{C} g_t \right)$ or $\gamma = \frac{2}{2 + t}$
7: Update $\mathbf{z}^{(t+1)} := (1 - \gamma) \mathbf{z}^{(t)} + \gamma \mathbf{s}^{(t)}$
8: end for

- Originally proposed by Hammond\textsuperscript{4} with $\gamma_t = 1/(t + 1)$.
- One can define FW extension with away step.

Algorithm  Saddle point FW algorithm

1: for $t = 0 \ldots T$ do
2:  Compute $r^{(t)} := \left( \nabla_x \mathcal{L}(x^{(t)}, y^{(t)}) \right)$
3:  Compute $s^{(t)} \in \arg\min_{z \in X \times Y} \langle z, r^{(t)} \rangle$
4:  Compute $g_t := \langle z^{(t)} - s^{(t)}, r^{(t)} \rangle$
5:  if $g_t \leq \epsilon$ then return $z^{(t)}$
6:  Let $\gamma = \min \left( 1, \frac{\nu}{C} g_t \right)$ or $\gamma = \frac{2}{2+t}$
7:  Update $z^{(t+1)} := (1 - \gamma)z^{(t)} + \gamma s^{(t)}$
8: end for

Originally proposed by Hammond\(^4\) with $\gamma_t = 1/(t + 1)$.

One can define FW extension with \texttt{away} step.

Crucial for our linear convergence results.

---

Algorithm Saddle point FW algorithm

1: for $t = 0 \ldots T$ do
2: Compute $\mathbf{r}^{(t)} := \left( \nabla_x \mathcal{L}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \right)$
3: Compute $\mathbf{s}^{(t)} \in \arg\min_{\mathbf{z} \in \mathcal{X} \times \mathcal{Y}} \langle \mathbf{z}, \mathbf{r}^{(t)} \rangle$
4: Compute $g_t := \langle \mathbf{z}^{(t)} - \mathbf{s}^{(t)}, \mathbf{r}^{(t)} \rangle$
5: if $g_t \leq \epsilon$ then return $\mathbf{z}^{(t)}$
6: Let $\gamma = \min \left( 1, \frac{\nu}{C} g_t \right)$ or $\gamma = \frac{2}{2+t}$
7: Update $\mathbf{z}^{(t+1)} := (1 - \gamma) \mathbf{z}^{(t)} + \gamma \mathbf{s}^{(t)}$
8: end for

- Originally proposed by Hammond\textsuperscript{4} with $\gamma_t = 1/(t + 1)$.
- One can define FW extension with \textit{away} step.
- Crucial for our linear convergence results.

\[ \gamma_t = \frac{1}{1+t} \Rightarrow \mathbf{z}^{(t)} = \frac{1}{t} \sum_{i=0}^{t} \mathbf{s}^{(i)}. \]

**Algorithm**  Saddle point FW algorithm

1: for $t = 0 \ldots T$ do  
2:  Compute $\mathbf{r}(t) := \left( \nabla_x L(x(t), y(t)), -\nabla_y L(x(t), y(t)) \right)$  
3:  Compute $\mathbf{s}(t) \in \text{argmin}_{z \in \mathcal{X} \times \mathcal{Y}} \langle z, \mathbf{r}(t) \rangle$  
4:  Compute $g_t := \langle z(t) - s(t), \mathbf{r}(t) \rangle$  
5:  if $g_t \leq \epsilon$ then return $z(t)$  
6:  Let $\gamma = \min \left( 1, \frac{\nu}{C} g_t \right)$ or $\gamma = \frac{2}{2+t}$  
7:  Update $z(t+1) := (1 - \gamma) z(t) + \gamma \mathbf{s}(t)$  
8: end for

- Originally proposed by Hammond\(^4\) with $\gamma_t = 1/(t + 1)$.
- One can define FW extension with *away* step.
- Crucial for our linear convergence results.

\[\gamma_t = \frac{1}{1+t} \Rightarrow z(t) = \frac{1}{t} \sum_{i=0}^{t} \mathbf{s}(i).\]

\[\left(\gamma_t = \frac{1}{1+t}\right) + \text{Bilinear objective} \leftrightarrow \text{fictitious play algorithm}.\]

Advantages of SP-FW

Same main property as FW:

Only LMO (linear minimization oracle).

Main difference with FW:

No line-search.

When constraint set is a "complicated" structured polytope: projection is difficult whereas LMO is tractable.
Advantages of SP-FW

Same main property as FW:

Only LMO (linear minimization oracle).

Same other *advantages* as FW:

- Convergence certificate $g_t$ for free.
Advantages of SP-FW

Same main property as FW:

- Only LMO (linear minimization oracle).

Same other advantages as FW:

- Convergence certificate $g_t$ for free.
- Affine invariance of the algorithm.
Advantages of SP-FW

Same main property as FW:

Only LMO (linear minimization oracle).

Same other advantages as FW:

- Convergence certificate $g_t$ for free.
- Affine invariance of the algorithm.
- **Sparsity** of the iterates.
Advantages of SP-FW

Same main property as FW:

- Only LMO (linear minimization oracle).

Same other **advantages** as FW:

- Convergence certificate $g_t$ for free.
- Affine invariance of the algorithm.
- **Sparsity** of the iterates.
- Universal step size $\gamma_t := \frac{2}{2+t}$, adaptive step size $\gamma_t := \frac{\nu}{C} g_t$. 
Advantages of SP-FW

Same main property as FW:

- Only LMO (linear minimization oracle).

Same other advantages as FW:

- Convergence certificate $g_t$ for free.
- Affine invariance of the algorithm.
- Sparsity of the iterates.
- Universal step size $\gamma_t := \frac{2}{2+t}$, adaptive step size $\gamma_t := \frac{\nu}{C} g_t$.

Main difference with FW:

- No line-search.
Advantages of SP-FW

Same main property as FW:

- Only LMO (linear minimization oracle).

Same other advantages as FW:

- Convergence certificate $g_t$ for free.
- Affine invariance of the algorithm.
- Sparsity of the iterates.
- Universal step size $\gamma_t := \frac{2}{2 + t}$, adaptive step size $\gamma_t := \nu g_t$.

Main difference with FW:

- No line-search.

When constraint set is a “complicated” structured polytope: projection is difficult whereas LMO is tractable.
Hypothesis

Similar hypothesis as AFW:

- $\mathcal{L}$ is $L$-smooth and $\mu$-strongly convex-concave.
- $\mathcal{X}$ and $\mathcal{Y}$ polytopes.
Similar hypothesis as AFW:

- $\mathcal{L}$ is $L$-smooth and $\mu$-strongly convex-concave.
- $\mathcal{X}$ and $\mathcal{Y}$ polytopes.
- Additional assumption on bilinearity:

$$\mathcal{L}(x, y) = f(x) + x^\top M y - h(y)$$

Roughly, $\|M\|$ smaller than the strong convexity constant.

$$\nu := \frac{1}{2} - \frac{\sqrt{2} \|M\|}{\mu} \frac{D}{\delta} > 0$$

$$D := \max\{\text{diam}(\mathcal{X}), \text{diam}(\mathcal{Y})\}, \quad \delta := \min\{\text{PWidth}(\mathcal{X}), \text{PWidth}(\mathcal{Y})\}$$
Theoretical contribution

SP extension of FW with \textit{away step}\textsuperscript{6}:

\textbf{Linear} rate with \textit{adaptive} step size $\gamma_t := \frac{\nu}{LD^2} g_t$.

$$\min_{s \leq t} g_s \leq O(1) \left(1 - \nu^2 \frac{\delta^2}{D^2} \frac{\mu}{2L}\right)^{k(t)}$$

\textbf{Sublinear} rate with \textit{universal} step size $\gamma_t := \frac{2}{2+k(t)}$.

$$\min_{s \leq t} g_s \leq O \left(\frac{1}{t}\right)$$

\begin{itemize}
  \item $k(t)$: number of non drop steps, $k(t) \geq \frac{t}{3}$.
\end{itemize}

Theoretical contribution

SP extension of FW with \textit{away step}\textsuperscript{6}:

\textbf{Linear} rate with \textit{adaptive} step size $\gamma_t := \frac{\nu}{LD^2} g_t$.  

$$\min_{s \leq t} g_s \leq O(1) \left(1 - \nu^2 \frac{\delta^2}{D^2} \frac{\mu}{2L}\right)^{k(t)}$$

\textbf{Sublinear} rate with \textit{universal} step size $\gamma_t := \frac{2}{2+k(t)}$.

$$\min_{s \leq t} g_s \leq O \left( \frac{1}{t} \right)$$

\begin{itemize}
  \item $k(t)$: number of non drop steps, $k(t) \geq t/3$.
  \item Proof use recent advances on AFW $\rightarrow$ growth condition.
\end{itemize}

Theoretical contribution

SP extension of FW with away step\(^7\):

**Linear** rate with *adaptive* step size \( \gamma_t := \frac{\nu}{L^2} g_t \).

\[
\min \{ g_s \mid s \leq t \} \leq O(1) \left( 1 - \nu^2 \frac{\delta^2}{D^2} \frac{\mu}{2L} \right)^{k(t)}
\]

**Sublinear** rate with *universal* step size \( \gamma_t := \frac{2}{2+k(t)} \).

\[
\min \{ g_s \mid s \leq t \} \leq O \left( \frac{1}{t} \right)
\]

- \( k(t) \) : number of non drop steps, \( k(t) \geq t/3 \).
- Proof use recent advances on AFW → growth condition.
- Partially answering a *30 years old conjecture*\(^8\).
  - strongly monotone obj with step size \( \frac{1}{t+1} \) over polytope.

---


Growth Condition: Pairwise Frank Wolfe Gap

\[ s_t := \arg \min_{s \in X} \langle \nabla f(x(t)), s \rangle. \]

\[ v_t := \arg \max_{v \in S(t)} \langle \nabla f(x(t)), s \rangle \]

\[ g_t^{PW} := \left\langle \nabla f(x(t)), v_t - s_t \right\rangle \]
Growth Condition

Key quantity, independent of any algorithm⁹:

- If $\mathcal{X}$ is a polytope and $f$ strongly convex,

$$f(\mathbf{x}^{(t)}) - f^* \leq \frac{(g_t^{PW})^2}{2\mu_{FW}}.$$
Growth Condition

Key quantity, independent of any algorithm\(^9\):

- If \( \mathcal{X} \) is a polytope and \( f \) strongly convex,
  \[
  f(x^{(t)}) - f^* \leq \frac{(g_t^{PW})^2}{2\mu_{FW}}.
  \]

- In the unconstrained case, analog of:
  \[
  f(x^{(t)}) - f^* \leq \frac{\|\nabla f(x^{(t)})\|^2}{2\mu}.
  \]

---

Growth Condition

Key quantity, independent of any algorithm\(^9\):

- If \( \mathcal{X} \) is a polytope and \( f \) strongly convex,

\[
f(x^{(t)}) - f^* \leq \frac{(g_t^{\text{PW}})^2}{2\mu_{FW}}.\]

- In the unconstrained case, analog of:

\[
f(x^{(t)}) - f^* \leq \frac{\|\nabla f(x^{(t)})\|^2}{2\mu}.
\]

- Can extend this growth condition to SP.

---

Difficulties for saddle point

Usual descent Lemma:

\[ h_{t+1} \leq h_t - \gamma_t g_t + \gamma_t^2 \frac{L \| d(t) \|^2}{2} \geq 0 \]

With \( \gamma_t \) small enough the sequence decreases.
Difficulties for saddle point

Usual descent Lemma:

\[ h_{t+1} \leq h_t - \gamma_t g_t + \gamma_t^2 \frac{L \| d^{(t)} \|^2}{2} \geq 0 \]

With \( \gamma_t \) small enough the sequence decreases.

For saddle point problem the Lipschitz gradient property gives

\[ \mathcal{L}_{t+1} - \mathcal{L}^* \leq \mathcal{L}_t - \mathcal{L}^* - \gamma_t \left( \frac{g_t(x) - g_t(y)}{2} \right) + \gamma_t^2 \frac{L \| d^{(t)} \|^2}{2} . \]

- Cannot control the oscillation of the sequence.
- Must introduce other quantities to establish convergence.
Difficulties for saddle point

Standard merit functions: *primal* + *dual gaps*

\[ h_t := \max_{y \in \mathcal{Y}} \mathcal{L}(x^{(t)}, y) - \min_{x \in \mathcal{X}} \mathcal{L}(x, y^{(t)}) \geq 0. \]

---

\[ 0 \leq w_t \leq h_t \leq g_t \]

In general, \( w_t \) can be zero even if we have not reached a solution. But for strongly convex-concave function

Difficulties for saddle point

Standard merit functions: *primal + dual gaps*

\[ h_t := \max_{y \in \mathcal{Y}} \mathcal{L}(x^{(t)}, y) - \min_{x \in \mathcal{X}} \mathcal{L}(x, y^{(t)}) \geq 0. \]

Problem: \( \hat{y}^{(t)} := \arg \max_{y \in \mathcal{Y}} \mathcal{L}(x^{(t)}, y) \) depends on \( t \).

---

Difficulties for saddle point

Standard merit functions: *primal + dual gaps*

\[ h_t := \max_{y \in Y} \mathcal{L}(x^{(t)}, y) - \min_{x \in X} \mathcal{L}(x, y^{(t)}) \geq 0. \]

Problem: \( \hat{y}^{(t)} := \arg\max_{y \in Y} \mathcal{L}(x^{(t)}, y) \) depends on \( t \).

\[ w_t := \underbrace{\mathcal{L}(x^{(t)}, y^*) - \mathcal{L}^*}_{:=w_t^{(x)}} + \underbrace{\mathcal{L}^* - \mathcal{L}(x^*, y^{(t)})}_{:=w_t^{(y)}}. \]

We have,

\[ 0 \leq w_t \leq h_t \leq g_t \]

In general, \( w_t \) can be zero even if we have not reached a solution. But for strongly convex-concave function\(^\text{10}\)

\[ h_t \leq Cte\sqrt{w_t} \]

Toy Experiments

- SP-AFW with theoretical step-size.

\[ \gamma_t = \nu \frac{g_t}{C} \]

\[ \mathcal{L}(x, y) := \frac{\mu}{2} \| x - x^* \|_2^2 + (x - x^*)^\top M(y - y^*) - \frac{\mu}{2} \| y - y^* \|_2^2 \]

- \( \mathcal{X} = \mathcal{Y} := [0, 1]^d \)
- \( d = 30 \)
- \( C := 2LD^2 \)
- \( L = \mu \)
Toy Experiments

- SP-AFW vs. Extragradient with approximate projection.

Theoretical step-size

\[ \gamma_t = \frac{\nu}{C} g_t. \]

EG : [He & Harchaoui NIPS 2015]

\[ \mathcal{L}(x, y) := \frac{\mu}{2} \| x - x^* \|^2_2 + (x - x^*)^\top M (y - y^*) - \frac{\mu}{2} \| y - y^* \|^2_2 \]

- \( \mathcal{X} = \mathcal{Y} := [0, 1]^d \)
- \( d = 30 \)
- \( C := 2LD^2 \)
- \( L = \mu \)
Toy Experiments

- SP-AFW with heuristic step-size. (When $\nu < 0$)

Heuristic step-size.

$$\gamma_t = \frac{g_t}{C + 2\|M\|^2D^2\|\mu\|}$$

Recall: theoretical step-size

$$\gamma_t = \frac{\nu}{C}g_t.$$ 

$$\mathcal{L}(x, y) := \frac{\mu}{2}\|x - x^*\|^2 + (x - x^*)^\top M(y - y^*) - \frac{\mu}{2}\|y - y^*\|^2$$

- $\mathcal{X} = \mathcal{Y} := [0, 1]^d$
- $d = 30$
- $C := 2LD^2$
- $L = \mu$
Toy Experiments

- SP-AFW with heuristic step-size. (When $\nu < 0$)

Heuristic step-size.

$$\gamma_t = \frac{g_t}{C + 2 \frac{\|M\|_2 D^2}{\mu}}$$

Recall: theoretical step-size

$$\gamma_t = \frac{\nu}{C} g_t.$$
Toy Experiments

- SP-AFW with heuristic step-size. (When $\nu < 0$)

Heuristic step-size.

$$\gamma_t = \frac{g_t}{C + 2\|M\|^2D^2}$$

Recall: theoretical step-size

$$\gamma_t = \frac{\nu}{C} g_t.$$
Conclusion

- SP-FW one of the first SP solver only working with LMO.
Conclusion

▶ SP-FW one of the first SP solver only working with LMO.
▶ FW resurgence lead to new structured problems.
Conclusion

- SP-FW one of the first SP solver only working with LMO.
- FW resurgence lead to new structured problems.
- Same hope as FW for SP-FW
Conclusion

- SP-FW one of the first SP solver only working with LMO.
- FW resurgence lead to new *structured* problems.
- Same hope as FW for SP-FW

Call for applications!
Conclusion

- SP-FW one of the first SP solver only working with LMO.
- FW resurgence lead to new *structured* problems.
- Same hope as FW for SP-FW ↵

**Call for applications !**

- With a bilinear objective this algorithm is *highly related* to the *fictitious play algorithm*. 
Conclusion

- SP-FW one of the first SP solver only working with LMO.
- FW resurgence lead to new *structured* problems.
- Same hope as FW for SP-FW

Call for applications!

- With a bilinear objective this algorithm is *highly related* to the *fictitious play algorithm*.
- Rich interplay tapping into this game theory literature.
Conclusion

- SP-FW one of the first SP solver only working with LMO.
- FW resurgence lead to new *structured* problems.
- Same hope as FW for SP-FW

Call for applications!

- With a bilinear objective this algorithm is *highly related* to the *fictitious play algorithm*.
- Rich interplay tapping into this game theory literature.
- Still many theoretical opened questions.
  - Karlin’s conjecture.\textsuperscript{11}
  - Convergence without assumption on the bilinearity.

Thank You!

Slides available on www.di.ens.fr/~gidel.
Problems with difficult projection

University game:

1. Game between two universities ($A$ and $B$).
2. Admitting $d$ students and have to assign pairs of students into dorms.
3. The game has a payoff matrix $M$ belonging to $\mathbb{R}^{(d(d-1)/2)^2}$.
4. $M_{ij,kl}$ is the expected tuition that $B$ gets (or $A$ gives up) if $A$ pairs student $i$ with $j$ and $B$ pairs student $k$ with $l$.
5. Here the actions are both in the marginal polytope of all perfect unipartite matchings.

Hard to project on this polytope whereas the LMO can be solved efficiently with the blossom algorithm\(^\text{12}\).

Experiments

Sublinear convergence rate (faster than expected $O(t^{-2})$)

Figure: SP-FW on the University game.
Experiments

- Sublinear convergence rate (faster than expected $O(t^{-2})$)
- Best theoretical rate proved: $O(t^{-1/d})$

Figure: SP-FW on the University game.
Experiments

Figure: SP-FW on the University game.

- Sublinear convergence rate (faster than expected $O(t^{-2})$)
- Best theoretical rate proved: $O(t^{-1/d})$
- Scale well with dimension.

Figure: SP-FW on the University game.