Simple steps are all you need: Frank-Wolfe and generalized self-concordant functions

Alejandro Carderera[1], Mathieu Besançon[2], Sebastian Pokutta[2,3]

[1]Georgia Institute of Technology, [2]Zuse Institute Berlin, [3]Technische Universität Berlin

NEURAL INFORMATION PROCESSING SYSTEMS

Problem Setting

Minimization of a *generalized-self concordant* (GSC) function over a compact convex $\mathcal{X}$

$$\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

Informally, GSC functions are those whose third derivative is bounded by their second derivative

Problem Setting

Minimization of a *generalized-self concordant* (GSC) function over a compact convex $\mathcal{X}$

$$\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

Informally, GSC functions are those whose third derivative is bounded by their second derivative

These functions appear in:

1. Interior-point formulations with barrier functions
2. Marginal inference with concave maximization
3. Logistic regression for classification

NEURAL INFORMATION PROCESSING SYSTEMS

Focus on $\mathcal{X}$ for which projections are hard

For example, if $\mathcal{X} = \mathcal{P} \cap \mathcal{C}$, where $\mathcal{P}$ is a polytope, and $\mathcal{C}$ is a convex set for which we can easily build a barrier function $\Phi_{\mathcal{C}}(\mathbf{x})$. Projecting onto $\mathcal{X}$ can be expensive!

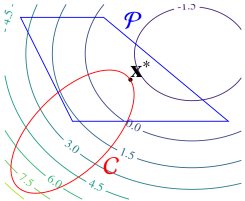

Figure: $f(x)$

Focus on $\mathcal{X}$ for which projections are hard

For example, if $\mathcal{X} = \mathcal{P} \cap \mathcal{C}$, where $\mathcal{P}$ is a polytope, and $\mathcal{C}$ is a convex set for which we can easily build a barrier function $\Phi_{\mathcal{C}}(\mathbf{x})$. Projecting onto $\mathcal{X}$ can be expensive!

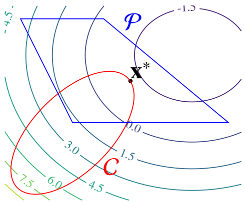

Figure: $f(x)$

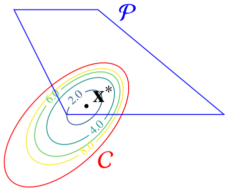

Figure: $f(x) + \mu' \Phi_{\mathcal{C}}(\mathbf{x})$

NEURAL INFORMATION PROCESSING SYSTEMS

Focus on $\mathcal{X}$ for which projections are hard

For example, if $\mathcal{X} = \mathcal{P} \cap \mathcal{C}$, where $\mathcal{P}$ is a polytope, and $\mathcal{C}$ is a convex set for which we can easily build a barrier function $\Phi_{\mathcal{C}}(\mathbf{x})$. Projecting onto $\mathcal{X}$ can be expensive!

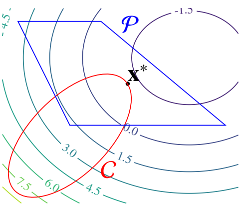

Figure: $f(x)$

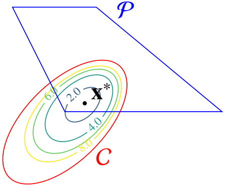

Figure: $f(x) + \mu' \Phi_{\mathcal{C}}(\mathbf{x})$

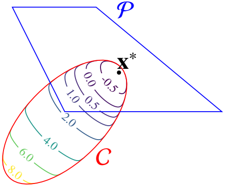

Figure: $f(x) + \mu \Phi_{\mathcal{C}}(\mathbf{x})$

Focus on $\mathcal{X}$ for which projections are hard

For example, if $\mathcal{X} = \mathcal{P} \cap \mathcal{C}$, where $\mathcal{P}$ is a polytope, and $\mathcal{C}$ is a convex set for which we can easily build a barrier function $\Phi_{\mathcal{C}}(\mathbf{x})$. Projecting onto $\mathcal{X}$ can be expensive!

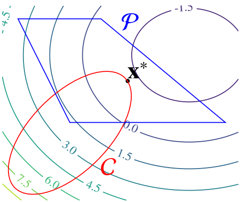

Figure: $f(x)$

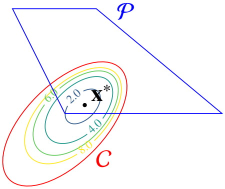

Figure: $f(x) + \mu' \Phi_{\mathcal{C}}(\mathbf{x})$

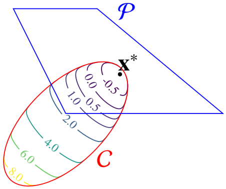

Figure: $f(x) + \mu \Phi_{\mathcal{C}}(\mathbf{x})$

A benefit from this approach is that the solution is expressed as a sparse convex combination of the vertices of $\mathcal{P}$, this might lead to better interpretability, or generalization capabilities

Main ingredients

Focus on the *Frank-Wolfe* (FW) algorithm [FW56; Pol74] using:

Main ingredients

Focus on the *Frank-Wolfe* (FW) algorithm [FW56; Pol74] using:
Domain Oracle (DO). Given $\mathbf{x} \in \mathbb{R}^n$, return:

$$\texttt{true} \text{ if } \mathbf{x} \in \operatorname{dom}(f), \texttt{false} \text{ otherwise}$$

Zeroth/First-Order Oracle (Z/FOO). Given $\mathbf{x} \in \operatorname{dom}(f)$ return:

$$\nabla f(\mathbf{x}) \in \mathbb{R}^n \text{ and } f(\mathbf{x}) \in \mathbb{R}$$

Linear Minimization Oracle (LMO). Given $\mathbf{v} \in \mathbb{R}^n$, return:

$$\operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} \langle \mathbf{v}, \mathbf{x} \rangle$$

NEURAL INFORMATION PROCESSING SYSTEMS

Main ingredients

Focus on the *Frank-Wolfe* (FW) algorithm [FW56; Pol74] using:

Domain Oracle (DO). Given $\mathbf{x} \in \mathbb{R}^n$, return:

$$\texttt{true} \text{ if } \mathbf{x} \in \mathrm{dom}(f), \ \texttt{false} \text{ otherwise}$$

Zeroth/First-Order Oracle (Z/FOO). Given $\mathbf{x} \in \mathrm{dom}(f)$ return:

$$\nabla f(\mathbf{x}) \in \mathbb{R}^n \text{ and } f(\mathbf{x}) \in \mathbb{R}$$

Linear Minimization Oracle (LMO). Given $\mathbf{v} \in \mathbb{R}^n$, return:

$$\operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} \langle \mathbf{v}, \mathbf{x} \rangle$$

Note that we do not assume access to a **second-order oracle** or a **backtracking line search!**

NEURAL INFORMATION PROCESSING SYSTEMS

Question: Why is **second-order** information (or the **backtracking line search** of [Ped+20]) used?

In order for the iterates to satisfy that $\mathbf{x}_t \in \mathrm{dom}(f)$*, and to use a smoothness-like inequality, obtaining algorithms with* $\mathcal{O}\left(1/t\right)$ *convergence rates in primal gap [Dvu+20]*

NEURAL INFORMATION PROCESSING SYSTEMS

Question: Why is **second-order** information (or the **backtracking line search** of [Ped+20]) used?

In order for the iterates to satisfy that $\mathbf{x}_t \in \mathrm{dom}(f)$, and to use a smoothness-like inequality, obtaining algorithms with $\mathcal{O}(1/t)$ convergence rates in primal gap [Dvu+20]

Question: Can we achieve the same rates without these two ingredients?

Yes! We can substitute second-order information for a domain oracle, and a backtracking line search for a $2/(2+t)$ step size

Contributions

Our contributions can be summarized as follows:

Contributions

Our contributions can be summarized as follows:

1. A simple parameter-free FW algorithm with a $2/(2+t)$ step size and a convergence rate of $\mathcal{O}(1/t)$ in **both** primal gap and Frank-Wolfe gap, without using second-order information or a backtracking line search

Contributions

Our contributions can be summarized as follows:

1. A simple parameter-free FW algorithm with a $2/(2+t)$ step size and a convergence rate of $\mathcal{O}(1/t)$ in **both** primal gap and Frank-Wolfe gap, without using second-order information or a backtracking line search
2. Improved convergence rates when the optimum is contained in the interior of $\mathcal{X} \cap \mathrm{dom}(f)$, or when the set $\mathcal{X}$ is uniformly or strongly convex, using the backtracking line search of [Ped+20]

Contributions

Our contributions can be summarized as follows:

1. A simple parameter-free FW algorithm with a $2/(2+t)$ step size and a convergence rate of $\mathcal{O}(1/t)$ in **both** primal gap and Frank-Wolfe gap, without using second-order information or a backtracking line search
2. Improved convergence rates when the optimum is contained in the interior of $\mathcal{X} \cap \mathrm{dom}(f)$, or when the set $\mathcal{X}$ is uniformly or strongly convex, using the backtracking line search of [Ped+20]
3. Numerical experiments that compare the performance of the algorithms on generalized self-concordant objectives to those in the existing literature

Monotonous Frank-Wolfe (`M-FW`)

for $t = 0$ to T **do**

$\quad \mathbf{v}_t \leftarrow \operatorname{argmin}_{\mathbf{x}\in\mathcal{X}} \langle \nabla f(\mathbf{x}_t), \mathbf{x}\rangle$, $\gamma_t \leftarrow 2/(2+t)$

$\quad \mathbf{x}_{t+1} = \mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)$

$\quad$ **if** $\mathbf{x}_{t+1} \notin \operatorname{dom}(f)$ **or** $f(\mathbf{x}_{t+1}) > f(\mathbf{x}_t)$ **then**

$\quad\quad \mathbf{x}_{t+1} = \mathbf{x}_t$

Monotonous Frank-Wolfe (`M-FW`)

for $t = 0$ to T **do**
  $\mathbf{v}_t \leftarrow \mathrm{argmin}_{\mathbf{x} \in \mathcal{X}} \langle \nabla f(\mathbf{x}_t), \mathbf{x} \rangle$, $\gamma_t \leftarrow 2/(2+t)$
  $\mathbf{x}_{t+1} = \mathbf{x}_t + \gamma_t (\mathbf{v}_t - \mathbf{x}_t)$
  if $\mathbf{x}_{t+1} \notin \mathrm{dom}(f)$ **or** $f(\mathbf{x}_{t+1}) > f(\mathbf{x}_t)$ **then**
    $\mathbf{x}_{t+1} = \mathbf{x}_t$

Theorem (Convergence rate of M-FW)

If f is a (M, ν) GSC function with $\nu \geq 2$. Then the `M-FW` satisfies:

$$f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq \frac{4(T_\nu + 1)}{t+1} \max\{h(\mathbf{x}_0), C\},$$

for $t \geq T_\nu$, where C and T_ν depends on the diameter of $\mathcal{X}$, ν, M, and the largest eigenvalue of the Hessian for points $\mathbf{y} \in \mathcal{X}$ with $f(\mathbf{y}) \leq f(\mathbf{x}_0)$. Otherwise it holds that $f(\mathbf{x}_t) \leq f(\mathbf{x}_0)$ for $t < T_\nu$.

Proof Sketch

For GSC functions, we have a *smoothness-like* inequality that holds locally around any given point $\mathbf{x}_t$. Denoting the primal gap by $h(\mathbf{x}_t)$ we have that:

$$h(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq h(\mathbf{x}_t)(1 - \gamma_t) + \gamma_t^2 C,$$

for $d(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t), \mathbf{x}_t) \leq 1/2$. But in order to compute d we would need knowledge of the Hessian!

Proof Sketch

For GSC functions, we have a *smoothness-like* inequality that holds locally around any given point $\mathbf{x}_t$. Denoting the primal gap by $h(\mathbf{x}_t)$ we have that:

$$h(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq h(\mathbf{x}_t)(1 - \gamma_t) + \gamma_t^2 C,$$

for $d(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t), \mathbf{x}_t) \leq 1/2$. But in order to compute d we would need knowledge of the Hessian!

However, there is a T_ν such that for $t \geq T_\nu$, due to the decreasing step size $\gamma_t = 2/(2+t)$, we know that $\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t) \in \mathrm{dom}(f)$ and $d(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t), \mathbf{x}_t) \leq 1/2$

Key Inequality

For $d(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t), \mathbf{x}_t) \leq 1/2$ *we have:*

$$h(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq h(\mathbf{x}_t)(1 - \gamma_t) + \gamma_t^2 C, \tag{1}$$

We need to ensure that $f(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq f(\mathbf{x}_t)$ in order to move, otherwise we set $\mathbf{x}_{t+1} = \mathbf{x}_t$. There are two scenarios:

Key Inequality

For $d(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t), \mathbf{x}_t) \leq 1/2$ *we have:*

$$h(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq h(\mathbf{x}_t)(1 - \gamma_t) + \gamma_t^2 C, \tag{1}$$

We need to ensure that $f(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq f(\mathbf{x}_t)$ in order to move, otherwise we set $\mathbf{x}_{t+1} = \mathbf{x}_t$. There are two scenarios:

Case $\gamma_t h(\mathbf{x}_t) - \gamma_t^2 C > 0$: Going to Eq. 1 we see that this means that $f(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq f(\mathbf{x}_t)$, so we take a non-zero step size! Using Equation 1 and induction we can prove the claim

Key Inequality

For $d(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t), \mathbf{x}_t) \leq 1/2$ *we have:*

$$h(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq h(\mathbf{x}_t)(1 - \gamma_t) + \gamma_t^2 C, \tag{1}$$

We need to ensure that $f(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq f(\mathbf{x}_t)$ in order to move, otherwise we set $\mathbf{x}_{t+1} = \mathbf{x}_t$. There are two scenarios:

Case $\gamma_t h(\mathbf{x}_t) - \gamma_t^2 C > 0$: Going to Eq. 1 we see that this means that $f(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq f(\mathbf{x}_t)$, so we take a non-zero step size! Using Equation 1 and induction we can prove the claim

Case $\gamma_t h(\mathbf{x}_t) - \gamma_t^2 C \leq 0$: Can't ensure $f(\mathbf{x}_t + \gamma_t(\mathbf{v}_t - \mathbf{x}_t)) \leq f(\mathbf{x}_t)$ using Eq. 1, however, we know that $f(\mathbf{x}_{t+1}) \leq f(\mathbf{x}_t)$ by monotonicity, and reordering $\gamma_t h(\mathbf{x}_t) - \gamma_t C \leq 0$ we have that $h(\mathbf{x}_t) \leq 2/(2+t)C$, which proves the claim.

Additional results

In addition, as stated before, our contributions include:

1. Proof of $\mathcal{O}(1/t)$ convergence in Frank-Wolfe gap for `M-FW`.
2. Improved convergence for variant using backtracking line search of [Ped+20] if $\mathbf{x}^* \in \operatorname{Int} \mathcal{X} \cap \operatorname{dom}(f)$, or if $\mathcal{X}$ is uniformly convex
3. Linearly convergent away-step variant for polytopes using backtracking line search of [Ped+20]

Computational Results

Portfolio Optimization over the probability simplex

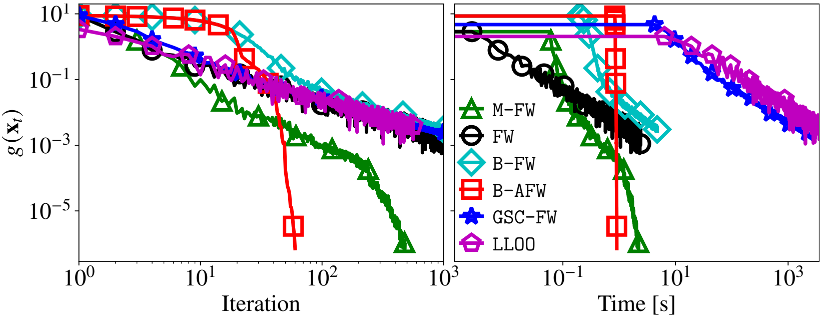

Figure: Frank-Wolfe gap vs. iteration (left) and time in seconds (right)

Computational Results

Logistic Regression over the ℓ_1 ball

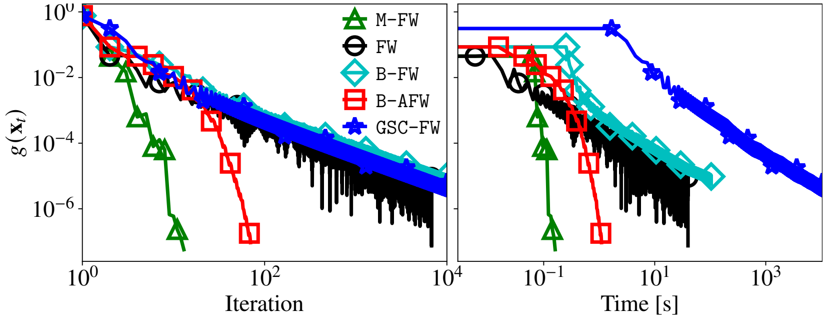

Figure: Frank-Wolfe gap vs. iteration (left) and time in seconds (right)

Computational Results

Matching over the Birkhoff polytope

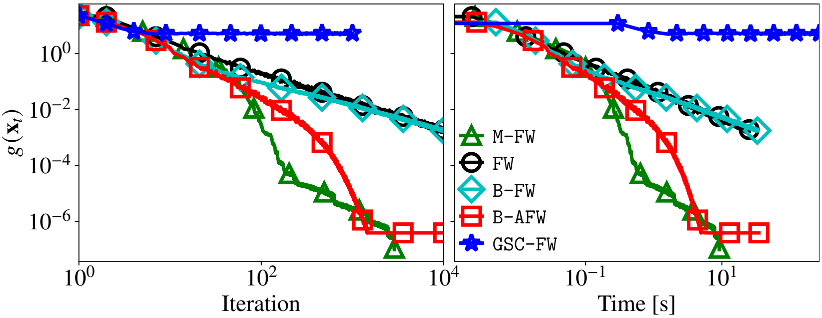

Figure: Frank-Wolfe gap vs. iteration (left) and time in seconds (right)

NEURAL INFORMATION PROCESSING SYSTEMS

Thank you
for your attention!

References I

[FW56] Marguerite Frank and Philip Wolfe. "An algorithm for quadratic programming". In: *Naval research logistics quarterly* 3.1-2 (1956), pp. 95–110.

[Pol74] Boris Teodorovich Polyak. "Minimization methods in the presence of constraints". In: *Itogi Nauki i Tekhniki. Seriya" Matematicheskii Analiz"* 12 (1974), pp. 147–197.

[Ped+20] Fabian Pedregosa, Geoffrey Negiar, Armin Askari, and Martin Jaggi. "Linearly Convergent Frank–Wolfe with Backtracking Line-Search". In: *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*. PMLR. 2020.

[Dvu+20] Pavel Dvurechensky, Kamil Safin, Shimrit Shtern, and Mathias Staudigl. "Generalized Self-Concordant Analysis of Frank-Wolfe algorithms". In: *arXiv preprint arXiv:2010.01009* (2020).