Frank-Wolfe Splitting via Augmented Lagrangian Method

Gauthier Gidel\textsuperscript{1}    Fabian Pedregosa\textsuperscript{2}    Simon Lacoste-Julien\textsuperscript{1}

\textsuperscript{1}MILA, DIRO Université de Montréal    \textsuperscript{2}UC Berkeley & ETH Zurich

April 2018
Why Frank-Wolfe is wonderful.

- Constrained optimization algorithm:

\[
\min_{x \in \mathcal{C}} f(x)
\]

\(f\) convex, \(\mathcal{C}\) convex compact.

- Interesting for highly structured constraint sets:

  - Alignment constraint: [Alayrac et al., 2016]
  - Permutahedron: [Lancia and Serafini, 2018] [Evangelopoulos et al., 2017]
Why Frank-Wolfe is wonderful.

- Constrained optimization algorithm:
  \[
  \min_{x \in C} f(x)
  \]
  \(f\) convex, \(C\) convex compact.

- Interesting for highly structured constraint sets:
  
  Alignment constraint:

  Permutahedron: [Lancia and Serafini, 2018] [Evangelopoulos et al., 2017]

  [Alayrac et al., 2016]
Why Frank-Wolfe is wonderful.

- Constrained optimization algorithm:
  \[
  \min_{x \in C} f(x)
  \]
  \(f\) convex, \(C\) convex compact.

- Interesting for highly structured constraint sets:

  **Alignment constraint:**
  \[C_{1,1}, C_{T,1}, C_{1,K}, C_{T,K}\]
  [Alayrac et al., 2016]

  **Permutahedron:**
  [Lancia and Serafini, 2018]
  [Evangelopoulos et al., 2017]
Why Frank-Wolfe is wonderful.

- Constrained optimization algorithm:
  \[
  \min_{x \in C} f(x)
  \]
  
  $f$ convex, $C$ convex compact.

- Interesting when projection is **not practical**:

  **Projection Linear Minimization Oracle**

- When projection is practical **better use** projected gradient method.
Why Frank-Wolfe sometimes is not enough.

- FW requires linear minimization (LMO) over these set.
  \[ \text{LMO}(d) := \arg \min_{x \in C} \langle d, x \rangle \]

- Intersection of constraint sets: \( C_1 \cap C_2 \).
- \( \text{LMO}_{C_1 \cap C_2}(d) \) may be too expensive.
- FW-AL just requires \( \text{LMO}_{C_1}(d) \) and \( \text{LMO}_{C_2}(d) \).
Why Frank-Wolfe sometimes is not enough.

- FW requires \textit{linear minimization} (LMO) over these set.

\[
\text{LMO}(d) := \arg \min_{x \in C} \langle d, x \rangle
\]

- Intersection of constraint sets: \(C_1 \cap C_2\).
- \(\text{LMO}_{C_1 \cap C_2}(d)\) may be too expensive.

- FW-AL just requires \(\text{LMO}_{C_1}(d)\) and \(\text{LMO}_{C_2}(d)\).
Why Frank-Wolfe sometimes is not enough.

- FW requires \textit{linear minimization} (LMO) over these set.
  \[
  \text{LMO}(d) := \arg \min_{x \in C} \langle d, x \rangle
  \]

- Intersection of constraint sets: \(C_1 \cap C_2\).
- \(\text{LMO}_{C_1 \cap C_2}(d)\) may be too expensive.

- FW-AL just requires \(\text{LMO}_{C_1}(d)\) and \(\text{LMO}_{C_2}(d)\).
Simultaneously sparse and low rank matrix recovery

Proposed by Richard et al. [2012]:

\[
\min_{S \succeq 0, \|S\|_1 \leq \beta_1, \|S\|_* \leq \beta_2} \|S - \hat{\Sigma}\|_2^2.
\]

- Sparcity constraint: \(C_1 := \{S \succeq 0, \|S\|_1 \leq \beta_1\}\),
  \(\text{LMO}_{C_1}(D) = \text{Largest coefficient of the matrix: } O(d^2)\)

- Low rank constraint: \(C_2 := \{S \succeq 0, \|S\|_* \leq \beta_2\}\).
  \(\text{LMO}_{C_2}(D) = \text{Largest eigenvector: } O(d^2/\sqrt{\epsilon})\)
Simultaneously sparse and low rank matrix recovery

Proposed by Richard et al. [2012]:

$$\begin{align*}
\min_{S \succeq 0, \|S\|_1 \leq \beta_1, \|S\|_* \leq \beta_2} & \quad \|S - \hat{\Sigma}\|_2^2.
\end{align*}$$

- **Sparcity constraint:** $C_1 := \{S \succeq 0, \|S\|_1 \leq \beta_1\}$,
  
  $$\text{LMO}_{C_1}(D) = \text{Largest coefficient of the matrix: } O(d^2)$$

- **Low rank constraint:** $C_2 := \{S \succeq 0, \|S\|_* \leq \beta_2\}$.
  
  $$\text{LMO}_{C_2}(D) = \text{Largest eigenvector: } O(d^2/\sqrt{\epsilon})$$
Simultaneously sparse and low rank matrix recovery

Proposed by Richard et al. [2012]:

\[
\min_{S \succeq 0, \|S\|_1 \leq \beta_1, \|S\|_* \leq \beta_2} \|S - \hat{\Sigma}\|^2_2.
\]

- **Sparcity constraint:** \( C_1 := \{ S \succeq 0, \|S\|_1 \leq \beta_1 \}, \)
  
  \( \text{LMO}_{C_1}(D) = \text{Largest coefficient of the matrix: } O(d^2) \)

- **Low rank constraint:** \( C_2 := \{ S \succeq 0, \|S\|_* \leq \beta_2 \}. \)
  
  \( \text{LMO}_{C_2}(D) = \text{Largest eigenvector: } O(d^2/\sqrt{\epsilon}) \)
Multiple sequence alignment

Proposed by Yen et al. [2016a]:

\[ \min_{W \in A \cap P} \langle W, D \rangle \]

- \( W \): alignment the sequences. \( D \): cost matrix.
- \( A \): alignment constraint. Each alignment with the consensus sequence is valid.
- \( P \): consensus constraint. Alignments consistent between each other.
Multiple sequence alignment

Proposed by Yen et al. [2016a]:

\[ \min_{W \in A \cap P} \langle W, D \rangle \]

- \( W \): alignment the sequences. \( D \): cost matrix.
- \( A \): alignment constraint. Each alignment with the consensus sequence is valid.
- \( P \): consensus constraint. Alignments consistent between each other.
Multiple sequence alignment

Proposed by Yen et al. [2016a]:

\[
\min_{W \in A \cap P} \langle W, D \rangle
\]

- \( W \): alignment the sequences. \( D \): cost matrix.
- \( A \): alignment constraint. Each alignment with the consensus sequence is valid.
- \( P \): consensus constraint. Alignments consistent between each other.
Multiple sequence alignment

Proposed by Yen et al. [2016a]:

$$\min_{W \in A \cap P} \langle W, D \rangle$$

- $A$: alignment constraint. Each alignment with the consensus sequence is valid.
- $P$: consensus constraint. Alignments consistent between each other.
Structured SVM

Proposed by Yen et al. [2016b]:

\[
\text{dual problem: } \min_{\alpha_f \in \Delta^{|\mathcal{Y}|}} \frac{1}{2} \sum_{F \in \mathcal{T}} \|A_F \alpha\|^2_2 - \sum_{j \in \mathcal{V}} \delta_j^\top \alpha_j \\
\text{s.t. } M_{fi} \alpha_f = \alpha_i, \quad f \in F, \ F \in \mathcal{T}, \ i \in \mathcal{N}(f).
\]

\(\mathcal{V}\): Variables. \(\mathcal{T}\): Factor templates. \(\mathcal{N}(f)\): neighbors of \(f\).

- Consistency constraint: \(M_{11} x^{(1)} = \alpha_1, M_{12} x^{(1)} = \alpha_2, \ldots\)

\[
\begin{array}{c}
\alpha_1 \quad x^{(1)} \quad \alpha_2 \quad x^{(2)} \quad \alpha_3
\end{array}
\]
Structured SVM

Proposed by Yen et al. [2016b]:

\[
\text{dual problem: } \min_{\alpha_f \in \Delta |\mathcal{Y}|} \frac{1}{2} \sum_{F \in \mathcal{T}} \|A_F \alpha\|_2^2 - \sum_{j \in \mathcal{V}} \delta_j^\top \alpha_j \\
\text{s.t. } M_{f_i} \alpha_f = \alpha_i, \ f \in F, F \in \mathcal{T}, i \in \mathcal{N}(f).
\]

\(\mathcal{V}\) : Variables. \(\mathcal{T}\) : Factor templates. \(\mathcal{N}(f)\): neighbors of \(f\).

Consistency constraint: \(M_{11} x^{(1)} = \alpha_1, M_{12} x^{(1)} = \alpha_2, \ldots\)
Proposed by Yen et al. [2016b]:

**dual problem:**  
\[
\min_{\alpha_f \in \Delta^{|\mathcal{Y}_f|}} \frac{1}{2} \sum_{F \in \mathcal{T}} \|A_F \alpha\|_2^2 - \sum_{j \in \mathcal{V}} \delta_j^\top \alpha_j
\]

s.t. \( M_{fi} \alpha_f = \alpha_i, \quad f \in F, \ F \in \mathcal{T}, \ i \in \mathcal{N}(f) \).

▶ \( \mathcal{V} \): Variables. \( \mathcal{T} \): Factor templates. \( \mathcal{N}(f) \): neighbors of \( f \).

▶ Consistency constraint: \( M_{11} x^{(1)} = \alpha_1, M_{12} x^{(1)} = \alpha_2, \ldots \)

\[\begin{align*}
\alpha_1 & \quad x^{(1)} \quad \alpha_2 & \quad x^{(2)} \quad \alpha_3
\end{align*}\]
General Formulation

\[
\begin{align*}
\text{minimize} \quad & f(x^{(1)}, \ldots, x^{(k)}) , \\
\text{subject to} \quad & x^{(k)} \in C_k, \quad k \in [K], \quad \sum_{k=1}^{K} A_k x^{(k)} = 0.
\end{align*}
\]

- $f$ is convex and smooth (gradient Lipschitz).
- $C_k, \ k \in \{1, \ldots, K\}$ are convex compact.
Augmented Lagrangian Method

- Augmented Lagrangian trick to get rid of $\sum_{k=1}^{K} A_k \mathbf{x}^{(k)} = 0$.

- $M$ s.t. $M \mathbf{x} = 0 \iff \sum_{k=1}^{K} A_k \mathbf{x}^{(k)} = 0$ and the functions,

  $$L(\mathbf{x}, \mathbf{y}) := f(\mathbf{x}) + \langle \mathbf{y}, M \mathbf{x} \rangle + \frac{\lambda}{2} \|M \mathbf{x}\|^2.$$ 

  $$p(\mathbf{x}) := \max_{\mathbf{y} \in \mathbb{R}^d} L(\mathbf{x}, \mathbf{y}) = \begin{cases} f(\mathbf{x}) & \text{if } M \mathbf{x} = 0, \\ +\infty & \text{otherwise}. \end{cases}$$

- Augmented Lagrangian formulation of our problem,

  $$\begin{align*}
  \text{minimize} & \quad \max_{\mathbf{x}} \max_{\mathbf{y} \in \mathbb{R}^d} L(\mathbf{x}, \mathbf{y}) \\
  \text{s.t.} & \quad \mathbf{x} \in \mathcal{X} := \times_{k=1}^{K} \mathcal{C}_k.
  \end{align*}$$
Augmented Lagrangian Method

- Augmented Lagrangian trick to get rid of $\sum_{k=1}^{K} A_k x^{(k)} = 0$.
- $M$ s.t. $Mx = 0 \iff \sum_{k=1}^{K} A_k x^{(k)} = 0$ and the functions,

$$L(x, y) := f(x) + \langle y, Mx \rangle + \frac{\lambda}{2} \|Mx\|^2. \quad p(x) := \max_{y \in \mathbb{R}^d} L(x, y) = \begin{cases} f(x) & \text{if } Mx = 0, \\ +\infty & \text{otherwise.} \end{cases}$$

- Augmented Lagrangian formulation of our problem,

$$\text{minimize } \max_{x} \max_{y \in \mathbb{R}^d} L(x, y) \quad \text{s.t. } x \in X := \times_{k=1}^{K} C_k.$$
Augmented Lagrangian Method

▶ Augmented Lagrangian trick to get rid of $\sum_{k=1}^{K} A_k x^{(k)} = 0$.

▶ $M$ s.t. $M x = 0 \iff \sum_{k=1}^{K} A_k x^{(k)} = 0$ and the functions,

$$\mathcal{L}(x, y) := f(x) + \langle y, M x \rangle + \frac{\lambda}{2} \|M x\|^2.$$ 

$$p(x) := \max_{y \in \mathbb{R}^d} \mathcal{L}(x, y) = \begin{cases} f(x) & \text{if } M x = 0, \\ +\infty & \text{otherwise.} \end{cases}$$

▶ Augmented Lagrangian formulation of our problem,

$$\minimize_{x} \max_{y \in \mathbb{R}^d} \mathcal{L}(x, y) \quad \text{s.t. } x \in \mathcal{X} := \times_{k=1}^{K} C_k .$$
FW-AL algorithm

\[
\begin{align*}
\text{minimize } & \max_{x, y \in \mathbb{R}^d} \mathcal{L}(x, y) \\
\text{s.t. } & x \in \mathcal{X} := \times_{k=1}^K \mathcal{C}_k.
\end{align*}
\]

- **Standard AL method:**
  
  \[
  \begin{cases}
  x_{t+1} = \arg\min_{x \in \mathcal{X}} \mathcal{L}(x, y) & \text{(argmin step)} , \\
  y_{t+1} = y_t + \eta_t M x_{t+1} & \text{(Gradient ascent step)}.
  \end{cases}
  \]

- **Replace arg min steps by FW steps. FW-AL:**
  
  \[
  \begin{cases}
  x_{t+1} = \text{FW}(x_t; \mathcal{L}(\cdot, y_t)) & \text{(Frank-Wolfe step)} , \\
  y_{t+1} = y_t + \eta_t M x_{t+1} & \text{(Gradient ascent step)}.
  \end{cases}
  \]
FW-AL algorithm

\[
\begin{align*}
\text{minimize } & \max_{x, y \in \mathbb{R}^d} \mathcal{L}(x, y) \\
\text{s.t. } & x \in \mathcal{X} := \times_{k=1}^K \mathcal{C}_k.
\end{align*}
\]

- Standard AL method:

\[
\begin{align*}
\{ x_{t+1} &= \arg\min_{x \in \mathcal{X}} \mathcal{L}(x, y_t) \} \quad \text{(argmin step)}, \\
y_{t+1} &= y_t + \eta_t M x_{t+1} \quad \text{(Gradient ascent step)}.
\end{align*}
\]

- Replace \( \arg\min \) steps by FW steps. FW-AL:

\[
\begin{align*}
\{ x_{t+1} &= \text{FW}(x_t; \mathcal{L}(\cdot, y_t)) \} \quad \text{(Frank-Wolfe step)}, \\
y_{t+1} &= y_t + \eta_t M x_{t+1} \quad \text{(Gradient ascent step)}.
\end{align*}
\]
The FW algorithm

Algorithm 1 One Frank-Wolfe step

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

**Algorithm 2 One Frank-Wolfe step**

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

Algorithm 3 One Frank-Wolfe step

1: Let $x^{(t)} \in \mathcal{M}$
2: Compute $r^{(t)} = \nabla f(x^{(t)})$
3: Compute $s^{(t)} \in \arg\min_{s \in \mathcal{C}} \langle s, r^{(t)} \rangle$
4: Compute $g_t := \langle x^{(t)} - s^{(t)}, r^{(t)} \rangle$
5: if $g_t \leq \epsilon$ then return $x^{(t)}$
6: Let $\gamma = \frac{2}{2 + t}$ (or do line-search)
7: Update $x^{(t+1)} := (1 - \gamma)x^{(t)} + \gamma s^{(t)}$
The FW algorithm

**Algorithm 4 One Frank-Wolfe step**

1: Let $x^{(t)} \in \mathcal{M}$
2: Compute $r^{(t)} = \nabla f(x^{(t)})$
3: Compute $s^{(t)} \in \text{argmin}_{s \in \mathcal{C}} \langle s, r^{(t)} \rangle$
4: Compute $g_t := \langle x^{(t)} - s^{(t)}, r^{(t)} \rangle$
5: if $g_t \leq \epsilon$ then return $x^{(t)}$
6: Let $\gamma = \frac{2}{2+t}$ (or do line-search)
7: Update $x^{(t+1)} := (1 - \gamma)x^{(t)} + \gamma s^{(t)}$
Related work: GDMM

- Replace arg min step by a FW step initially proposed by Yen et al. [2016a] to solve MSA problem.

- Afterwards used for Structured SVM [Yen et al., 2016b] and MAP inference [Huang et al., 2017].

- Restricted to polytopes and simple (linear and quadratic) functions.

Contributions:

- Extension of GDMM for general convex sets. (e.g. Trace norm ball)

- Fix a crucial missing part in the previous proofs.
Related work: GDMM

- Replace arg min step by a FW step initially proposed by Yen et al. [2016a] to solve MSA problem.

- Afterwards used for Structured SVM [Yen et al., 2016b] and MAP inference [Huang et al., 2017].

- Restricted to polytopes and simple (linear and quadratic) functions.

Contributions:

- Extension of GDMM for general convex sets. (e.g. Trace norm ball)

- Fix a crucial missing part in the previous proofs.
Related work: GDMM

- Replace arg min step by a FW step initially proposed by Yen et al. [2016a] to solve MSA problem.

- Afterwards used for Structured SVM [Yen et al., 2016b] and MAP inference [Huang et al., 2017].

- Restricted to polytopes and simple (linear and quadratic) functions.

Contributions:

- Extension of GDMM for general convex sets. (e.g. Trace norm ball)

- Fix a crucial missing part in the previous proofs.
Related work: GDMM

- Replace arg min step by a FW step initially proposed by Yen et al. [2016a] to solve MSA problem.
- Afterwards used for Structured SVM [Yen et al., 2016b] and MAP inference [Huang et al., 2017].
- Restricted to polytopes and simple (linear and quadratic) functions.

Contributions:

- Extension of GDMM for general convex sets. (e.g. Trace norm ball)
- Fix a crucial missing part in the previous proofs.
**Theoretical contribution**

**Additional assumption:**

Slater’s condition: \( \exists \mathbf{x}^{(k)} \in \text{relint}(C_k), \ k \in [K] \) s.t. \( \sum_{k=1}^{K} A_k \mathbf{x}^{(k)} = 0 \).

**New lemma:**

Let \( d \) be the augmented dual function,

\[ d(y) := \min_{x \in \mathcal{X}} \mathcal{L}(x, y). \]

There exist a constant \( \alpha > 0 \) such that close enough to \( \mathcal{Y}^* \),

\[ d^* - d(y) \geq \alpha \text{dist}(y, \mathcal{Y}^*)^2. \]
Theoretical contribution

Additional assumption:

Slater’s condition: \( \exists \mathbf{x}^{(k)} \in \text{relint}(\mathcal{C}_k), k \in [K] \) s.t. \( \sum_{k=1}^{K} A_k \mathbf{x}^{(k)} = 0 \).

New lemma:

Let \( d \) be the augmented dual function,

\[
d(y) := \min_{x \in \mathcal{X}} \mathcal{L}(x, y).
\]

There exist a constant \( \alpha > 0 \) such that close enough to \( \mathcal{Y}^* \),

\[
d^* - d(y) \geq \alpha \text{dist}(y, \mathcal{Y}^*)^2.
\]
Convergence results

- **For general convex sets:**
  
  With decreasing step size \( \eta_t := O\left(\frac{1}{t+1}\right) \),
  
  subopt: \( \Delta_t \leq \frac{O(1)}{t} \),  
  feasibility: \( \min_{t_0 \leq s \leq t} \| Mx_s \|^2 \leq \frac{O(1)}{t} \).

- **For \( \mathcal{X} \) a polytope:**
  
  With small enough constant step size \( \eta_t \):
  
  \[ \Delta_t \leq \frac{\Delta_{t_0}}{(1 + \rho)^{t-t_0}}, \quad \| Mx_{t+1} \|^2 \leq \frac{O(1)}{(1 + \rho)^{t-t_0}}. \]

  Only holds for generalized strongly convex function and uses a variant of FW with away-step.

- Standard splitting algorithms have faster rate per iteration in practice.

- Advantage only comes from the cheaper oracle!
Convergence results

- **For general convex sets:**
  With decreasing step size $\eta_t := O\left(\frac{1}{t+1}\right)$,

  subopt: $\Delta_t \leq \frac{O(1)}{t}$, feasibility: $\min_{t_0 \leq s \leq t} \|Mx_s\|^2 \leq \frac{O(1)}{t}$.

- **For $\mathcal{X}$ a polytope:**
  With small enough constant step size $\eta_t$:

  $\Delta_t \leq \frac{\Delta_{t_0}}{(1 + \rho)^{t-t_0}}$, $\|Mx_{t+1}\|^2 \leq \frac{O(1)}{(1 + \rho)^{t-t_0}}$.

  Only holds for generalized strongly convex function and uses a variant of FW with away-step.

- Standard splitting algorithms have faster rate **per iteration** in practice.

- Advantage only comes from the **cheaper oracle**!
Simultaneously sparse and low rank matrix recovery:

$$\min_{S \succeq 0, \|S\|_1 \leq \beta_1, \|S\|_* \leq \beta_2} \|S - \hat{\Sigma}\|_2^2.$$ 

- **Sparcity constraint:** $C_1 := \{S \succeq 0, \|S\|_1 \leq \beta_1\}$,

  $$\text{LMO}_{C_1}(D) = \text{Largest coefficient of the matrix: } O(d^2)$$

- **Low rank constraint:** $C_2 := \{S \succeq 0, \|S\|_* \leq \beta_2\}$.

  $$\text{LMO}_{C_2}(D) = \text{Largest eigenvector: } O(d^2/\sqrt{\epsilon})$$
Experiments

LMO vs. projection for trace norm ball:

![Graph showing comparison between FW-AL Vs. Baseline, Linear oracle on $B_*$, and Projection on $B_*$ over different dimensions. The x-axis represents dimension, and the y-axis represents time (in s). The graph demonstrates how the time increases with the dimension for all methods, with Projection on $B_*$ having the lowest time across all dimensions.]
Support recovered by FW-AL and the generalized forward backward algorithm as a function of time:
Task: Minimize a function over an intersection of convex sets.

Problem:
- Projections or linear minimization oracle (LMO) over the intersection is expensive.
- Projection onto each individual set is expensive.

Our solution:
- Requires linear minimization oracles over individual constraints.
- Based on the Augmented Lagrangian Method.

Contributions:
- Extension of GDMM for general convex sets.
- Fix a missing part of the previous proofs.
Conclusion

Task: Minimize a function over an intersection of convex sets.

Problem:

▶ Projections or linear minimization oracle (LMO) over the intersection is expensive.
▶ Projection onto each individual set is expensive.

Our solution:

▶ Requires linear minimization oracles over individual constraints.
▶ Based on the Augmented Lagrangian Method.

Contributions:

▶ Extension of GDMM for general convex sets.
▶ Fix a missing part of the previous proofs.
Conclusion

**Task:** Minimize a function over an *intersection* of convex sets.

**Problem:**
- Projections or linear minimization oracle (LMO) over the intersection is **expensive**.
- Projection onto each individual set is **expensive**.

**Our solution:**
- Requires linear minimization oracles over individual constraints.
- Based on the Augmented Lagrangian Method.

**Contributions:**
- Extension of GDMM for general convex sets.
- Fix a missing part of the previous proofs.
Conclusion

**Task:** Minimize a function over an intersection of convex sets.

**Problem:**
- Projections or linear minimization oracle (LMO) over the intersection is **expensive**.
- Projection onto each individual set is **expensive**.

**Our solution:**
- Requires linear minimization oracles over individual constraints.
- Based on the Augmented Lagrangian Method.

**Contributions:**
- Extension of GDMM for general convex sets.
- Fix a missing part of the previous proofs.
Conclusion

**Task:** Minimize a function over an *intersection* of convex sets.

**Problem:**
- Projections or linear minimization oracle (LMO) over the intersection is *expensive*.
- Projection onto each individual set is *expensive*.

**Our solution:**
- Requires *linear minimization oracles* over *individual constraints*.
- Based on the **Augmented Lagrangian Method**.

**Contributions:**
- Extension of GDMM for general convex sets.
- Fix a missing part of the previous proofs.
Conclusion

**Task:** Minimize a function over an *intersection* of convex sets.

**Problem:**
- Projections or linear minimization oracle (LMO) over the intersection is *expensive*.
- Projection onto each individual set is *expensive*.

**Our solution:**
- Requires *linear minimization oracles* over individual constraints.
- Based on the **Augmented Lagrangian Method**.

**Contributions:**
- Extension of GDMM for general convex sets.
- Fix a missing part of the previous proofs.
Conclusion

**Task:** Minimize a function over an *intersection* of convex sets.

**Problem:**
- Projections or linear minimization oracle (LMO) over the intersection is expensive.
- Projection onto each individual set is expensive.

**Our solution:**
- Requires linear minimization oracles over individual constraints.
- Based on the Augmented Lagrangian Method.

**Contributions:**
- Extension of GDMM for general convex sets.
- Fix a missing part of the previous proofs.
Thank You!


