

Conditional Gradients in Machine Learning

Sebastian Pokutta

Technische Universität Berlin
and
Zuse Institute Berlin

pokutta@math.tu-berlin.de
[@spokutta](https://twitter.com/espokutta)

Workshop on Geometry and Machine Learning

November 14, 2023 · Leipzig, Germany



Berlin Mathematics Research Center



What is this talk about?

Introduction

A very versatile and simple optimization method for projection-free optimization that promotes sparsity.

What is this talk about?

Introduction

A very versatile and simple optimization method for projection-free optimization that promotes sparsity.

Why? Constraints and Sparsity help interpretability and explainability.

What is this talk about?

Introduction

A very versatile and simple optimization method for projection-free optimization that promotes sparsity.

Why? Constraints and Sparsity help interpretability and explainability.

Today: A brief overview of recent developments in *conditional gradient methods*.

What is this talk about?

Introduction

A very versatile and simple optimization method for projection-free optimization that promotes sparsity.

Why? Constraints and Sparsity help interpretability and explainability.

Today: A brief overview of recent developments in *conditional gradient methods*.

Outline

- The basics: Conditional Gradients a.k.a. the Frank-Wolfe algorithm
- Several examples:
 - Recovering Dynamics from Noisy Data
 - Deep Learning
 - Robust Rate-Distortion Explanation
- High-performance Julia Package: FrankWolfe.jl

(Hyperlinked) References are not exhaustive; check references contained therein.



Conditional Gradients a.k.a. the Frank-Wolfe algorithm

—The Basics—

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Given a smooth and convex function f and a polytope P , solve **optimization problem**:

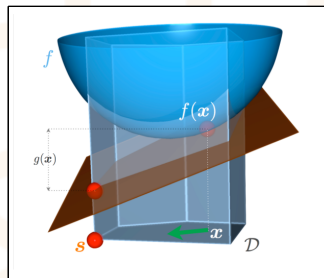
The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Given a smooth and convex function f and a polytope P , solve **optimization problem**:

$$\min_{x \in P} f(x)$$

(baseProblem)



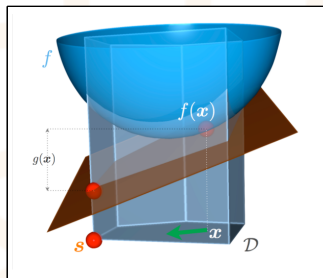
Source: [Jaggi, 2013]

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Given a smooth and convex function f and a polytope P , solve **optimization problem**:

$$\min_{x \in P} f(x) \quad (\text{baseProblem})$$



Source: [Jaggi, 2013]

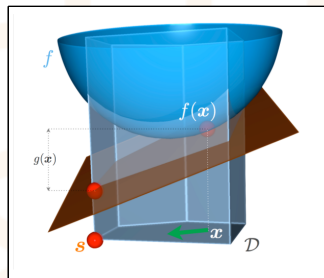
1. Very **versatile** model
2. Can use various types of **information about both f and P**
3. Works very well in (continuous) **real-world applications**
4. At the core of many (all?) **learning algorithms** (albeit mostly non-convex case)

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Given a smooth and convex function f and a polytope P , solve **optimization problem**:

$$\min_{x \in P} f(x) \quad (\text{baseProblem})$$



Source: [Jaggi, 2013]

Our setup.

1. Access to P . **Linear Minimization Oracle (LMO)**: Given linear objective c return

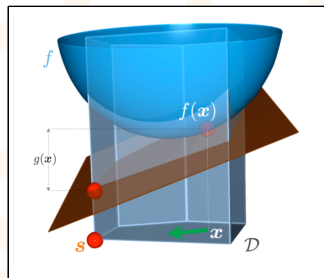
$$x \leftarrow \arg \min_{v \in P} c^T v.$$

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Given a smooth and convex function f and a polytope P , solve **optimization problem**:

$$\min_{x \in P} f(x) \quad (\text{baseProblem})$$



Source: [Jaggi, 2013]

Our setup.

1. Access to P . **Linear Minimization Oracle (LMO)**: Given linear objective c return

$$x \leftarrow \arg \min_{v \in P} c^T v.$$

2. Access to f . **First-Order Oracle (FO)**: Given x return

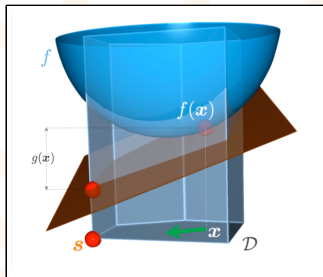
$$\nabla f(x) \quad \text{and} \quad f(x).$$

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Given a smooth and convex function f and a polytope P , solve **optimization problem**:

$$\min_{x \in P} f(x) \quad (\text{baseProblem})$$



Source: [Jaggi, 2013]

Our setup.

1. Access to P . **Linear Minimization Oracle (LMO)**: Given linear objective c return

$$x \leftarrow \arg \min_{v \in P} c^T v.$$

2. Access to f . **First-Order Oracle (FO)**: Given x return

$$\nabla f(x) \quad \text{and} \quad f(x).$$

\Rightarrow *Complexity of convex optimization relative to LO/FO oracle*

Interlude: why LMOs?

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

LMO model has many advantages.

1. Includes explicit formulation via constraints
2. Some problems do not possess 'small' formulations but have efficient LMOs.
Example: Matching Polytope [Rothvoss, 2014, Braun and Pokutta, 2015a,b, Braun et al., 2015, 2017b]
3. Allows modeling of compact convex constraints as long as we have an LMO.
Example: SDP cone
4. Often much faster than projection.
Example: nuclear norm. Largest singular vector (Lanczos method) vs. full SVD
5. LMO is a black box for the algorithms
6. For many LMOs of interest close form solutions available.
Example: ℓ_1 -ball for LASSO regression.

For an overview see: [Combettes and Pokutta, 2021]

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Basic notions. Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a differentiable function.

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Basic notions. Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a differentiable function.

Definition (Convexity)

For all x, y it holds:

$$f(y) - f(x) \geq \langle \nabla f(x), y - x \rangle .$$

In particular, all local minima are global minima.

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Basic notions. Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a differentiable function.

Definition (Convexity)

For all x, y it holds:

$$f(y) - f(x) \geq \langle \nabla f(x), y - x \rangle .$$

In particular, all local minima are global minima.

Definition (L -Smoothness)

For all x, y it holds:

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

The basic problem

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Basic notions. Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a differentiable function.

Definition (Convexity)

For all x, y it holds:

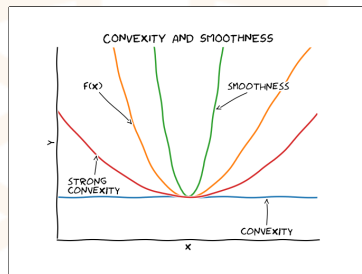
$$f(y) - f(x) \geq \langle \nabla f(x), y - x \rangle.$$

In particular, all local minima are global minima.

Definition (L -Smoothness)

For all x, y it holds:

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

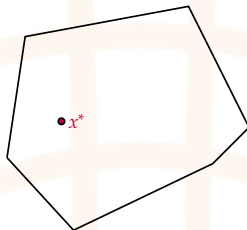


The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



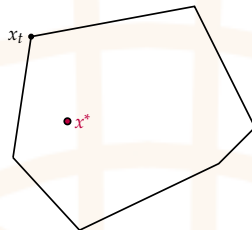
[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



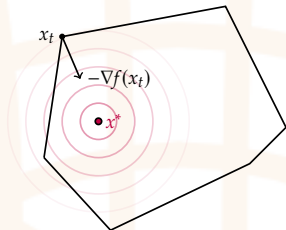
[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



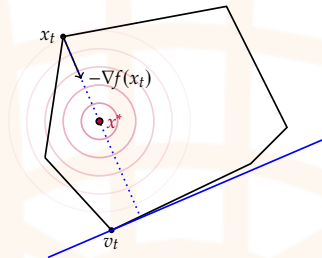
[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



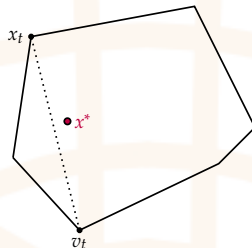
[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



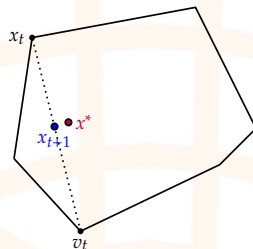
[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



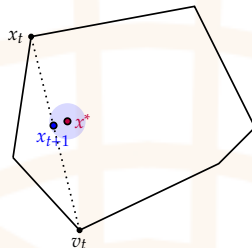
[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



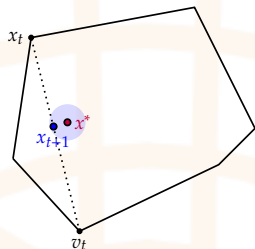
[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

Advantages:

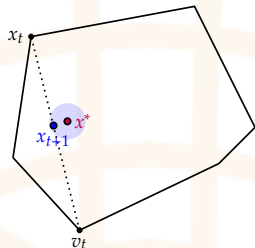
- **Extremely simple and robust:** no complicated data structures to maintain
- **Easy to implement:** requires only the two oracles
- **Projection-free:** feasibility convex combination and LO oracle
- **Sparsity:** optimal solution is a convex combination of (usually) vertices
- **Affine invariance:** no rescaling etc required
- **Parameter-free:** does not require any knowledge about the function or feasible region

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

Advantages:

- **Extremely simple and robust:** no complicated data structures to maintain
- **Easy to implement:** requires only the two oracles
- **Projection-free:** feasibility convex combination and LO oracle
- **Sparsity:** optimal solution is a convex combination of (usually) vertices
- **Affine invariance:** no rescaling etc required
- **Parameter-free:** does not require any knowledge about the function or feasible region

Disadvantages:

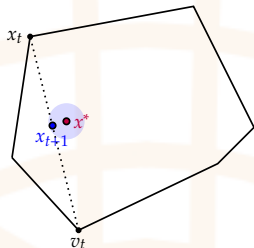
- Suboptimal convergence rate of $O(1/T)$
- No iterate convergence in the classical sense

The Frank-Wolfe Algorithm

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Algorithm Frank-Wolfe Algorithm (FW)

- 1: $x_0 \in P$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
 - 5: **end for**
-



[Frank and Wolfe, 1956, Levitin and Polyak, 1966]

Advantages:

- **Extremely simple and robust:** no complicated data structures to maintain
- **Easy to implement:** requires only the two oracles
- **Projection-free:** feasibility convex combination and LO oracle
- **Sparsity:** optimal solution is a convex combination of (usually) vertices
- **Affine invariance:** no rescaling etc required
- **Parameter-free:** does not require any knowledge about the function or feasible region

Disadvantages:

- Suboptimal convergence rate of $O(1/T)$
- No iterate convergence in the classical sense

\Rightarrow *Despite (theoretically) suboptimal rate heavily used in applications due to simplicity.*

Simple Convergence Proof

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Theorem (Convergence rate of the vanilla Frank-Wolfe Algorithm)

Let f be L -smooth convex, P be polytope with diameter D . With choice $\gamma_t \doteq \frac{2}{t+3}$:

$$f(x_t) - f(x^*) \leq \frac{2LD^2}{t+3}.$$

Simple Convergence Proof

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Theorem (Convergence rate of the vanilla Frank-Wolfe Algorithm)

Let f be L -smooth convex, P be polytope with diameter D . With choice $\gamma_t \doteq \frac{2}{t+3}$:

$$f(x_t) - f(x^*) \leq \frac{2LD^2}{t+3}.$$

Proof Sketch.

By smoothness:

$$f(x_{t+1}) - f(x_t) \leq \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2} \|x_{t+1} - x_t\|^2 = \gamma_t \langle \nabla f(x_t), v_t - x_t \rangle + \frac{L\gamma_t^2}{2} \|v_t - x_t\|^2.$$

Simple Convergence Proof

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Theorem (Convergence rate of the vanilla Frank-Wolfe Algorithm)

Let f be L -smooth convex, P be polytope with diameter D . With choice $\gamma_t \doteq \frac{2}{t+3}$:

$$f(x_t) - f(x^*) \leq \frac{2LD^2}{t+3}.$$

Proof Sketch.

By smoothness:

$$f(x_{t+1}) - f(x_t) \leq \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2} \|x_{t+1} - x_t\|^2 = \gamma_t \langle \nabla f(x_t), v_t - x_t \rangle + \frac{L\gamma_t^2}{2} \|v_t - x_t\|^2.$$

LP maximality and convexity: $\langle \nabla f(x_t), v_t - x_t \rangle \leq \langle \nabla f(x_t), x^* - x_t \rangle \leq f(x^*) - f(x_t)$. Moreover, $\|v_t - x_t\| \leq D$.

Simple Convergence Proof

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Theorem (Convergence rate of the vanilla Frank-Wolfe Algorithm)

Let f be L -smooth convex, P be polytope with diameter D . With choice $\gamma_t \doteq \frac{2}{t+3}$:

$$f(x_t) - f(x^*) \leq \frac{2LD^2}{t+3}.$$

Proof Sketch.

By smoothness:

$$f(x_{t+1}) - f(x_t) \leq \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2} \|x_{t+1} - x_t\|^2 = \gamma_t \langle \nabla f(x_t), v_t - x_t \rangle + \frac{L\gamma_t^2}{2} \|v_t - x_t\|^2.$$

LP maximality and convexity: $\langle \nabla f(x_t), v_t - x_t \rangle \leq \langle \nabla f(x_t), x^* - x_t \rangle \leq f(x^*) - f(x_t)$. Moreover, $\|v_t - x_t\| \leq D$.

Thus:

$$f(x_{t+1}) - f(x^*) \leq (1 - \gamma_t)(f(x_t) - f(x^*)) + \gamma_t^2 \frac{LD^2}{2}.$$

Simple Convergence Proof

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Theorem (Convergence rate of the vanilla Frank-Wolfe Algorithm)

Let f be L -smooth convex, P be polytope with diameter D . With choice $\gamma_t \doteq \frac{2}{t+3}$:

$$f(x_t) - f(x^*) \leq \frac{2LD^2}{t+3}.$$

Proof Sketch.

By smoothness:

$$f(x_{t+1}) - f(x_t) \leq \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2} \|x_{t+1} - x_t\|^2 = \gamma_t \langle \nabla f(x_t), v_t - x_t \rangle + \frac{L\gamma_t^2}{2} \|v_t - x_t\|^2.$$

LP maximality and convexity: $\langle \nabla f(x_t), v_t - x_t \rangle \leq \langle \nabla f(x_t), x^* - x_t \rangle \leq f(x^*) - f(x_t)$. Moreover, $\|v_t - x_t\| \leq D$.

Thus:

$$f(x_{t+1}) - f(x^*) \leq (1 - \gamma_t)(f(x_t) - f(x^*)) + \gamma_t^2 \frac{LD^2}{2}.$$

By Induction (plugging in the guarantee + definition of γ_t):

$$f(x_{t+1}) - f(x^*) \leq \left(1 - \frac{2}{t+3}\right) \frac{2LD^2}{t+3} + \frac{4}{(t+3)^2} \cdot \frac{LD^2}{2} = \frac{2LD^2(t+2)}{(t+3)^2} \leq \frac{2LD^2}{t+4},$$

by $(t+2)(t+4) \leq (t+3)^2$.

A matching lower bound

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Consider $P = \text{conv}(\{e_1, \dots, e_n\})$ the probability simplex and $f = \|x\|^2$.

A matching lower bound

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Consider $P = \text{conv}(\{e_1, \dots, e_n\})$ the probability simplex and $f = \|x\|^2$.

Clearly. $\arg \min_{x \in P} f(x) = x^* \doteq \frac{1}{n}e$ with $f(x^*) = \frac{1}{n}$.

A matching lower bound

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Consider $P = \text{conv}(\{e_1, \dots, e_n\})$ the probability simplex and $f = \|x\|^2$.

Clearly. $\arg \min_{x \in P} f(x) = x^* \doteq \frac{1}{n}e$ with $f(x^*) = \frac{1}{n}$.

Observe. Starting from any vertex e_i after $t < n$ iterations we picked up at most t vertices of P .

A matching lower bound

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Consider $P = \text{conv}(\{e_1, \dots, e_n\})$ the probability simplex and $f = \|x\|^2$.

Clearly. $\arg \min_{x \in P} f(x) = x^* \doteq \frac{1}{n}e$ with $f(x^*) = \frac{1}{n}$.

Observe. Starting from any vertex e_i after $t < n$ iterations we picked up at most t vertices of P .

Easy to see. For any iterate x_t :

$$f(x_t) \geq \min_{\substack{x \in \text{conv}(S) \\ S \subseteq \{e_1, \dots, e_n\} \\ |S| \leq t}} f(x) = 1/t,$$

A matching lower bound

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Consider $P = \text{conv}(\{e_1, \dots, e_n\})$ the probability simplex and $f = \|x\|^2$.

Clearly. $\arg \min_{x \in P} f(x) = x^* \doteq \frac{1}{n}e$ with $f(x^*) = \frac{1}{n}$.

Observe. Starting from any vertex e_i after $t < n$ iterations we picked up at most t vertices of P .

Easy to see. For any iterate x_t :

$$f(x_t) \geq \min_{\substack{x \in \text{conv}(S) \\ S \subseteq \{e_1, \dots, e_n\} \\ |S| \leq t}} f(x) = 1/t,$$

Thus lower bound. $f(x_t) - f(x^*) \geq \frac{1}{t} - \frac{1}{n}$

\Rightarrow Any LP method converges no faster than $O(1/t)$.

Note: Strong consequences for strongly convex case and also provides a sparsity vs. optimality trade-off.

see also for non-smooth variants: [Braun et al., 2017a]

A matching lower bound

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

Consider $P = \text{conv}(\{e_1, \dots, e_n\})$ the probability simplex and $f = \|x\|^2$.

Clearly. $\arg \min_{x \in P} f(x) = x^* \doteq \frac{1}{n}e$ with $f(x^*) = \frac{1}{n}$.

Observe. Starting from any vertex e_i after $t < n$ iterations we picked up at most t vertices of P .

Easy to see. For any iterate x_t :

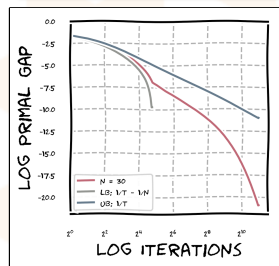
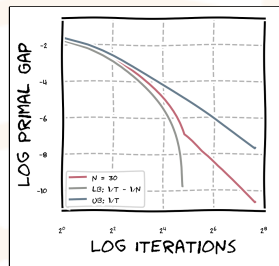
$$f(x_t) \geq \min_{\substack{x \in \text{conv}(S) \\ S \subseteq \{e_1, \dots, e_n\} \\ |S| \leq t}} f(x) = 1/t,$$

Thus lower bound. $f(x_t) - f(x^*) \geq \frac{1}{t} - \frac{1}{n}$

\Rightarrow Any LP method converges no faster than $O(1/t)$.

Note: Strong consequences for strongly convex case and also provides a sparsity vs. optimality trade-off.

see also for non-smooth variants: [Braun et al., 2017a]



Significant progress over the recent years (incomplete list)

Conditional Gradients a.k.a. the Frank-Wolfe algorithm

1. Strongly convex case [Garber and Hazan, 2013, Lacoste-Julien and Jaggi, 2015, Lan and Zhou, 2016, Garber and Meshi, 2016]
2. Non-convex case [Lacoste-Julien, 2016]
3. Online case [Hazan and Kale, 2012]
4. Stochastic variants and adaptive gradients [Hazan and Luo, 2016, Reddi et al., 2016, Combettes et al., 2020]
5. Sharp functions and sharp regions [Kerdreux et al., 2019, 2021a,b]
6. Acceleration [Diakonikolas et al., 2020, Bach, 2020, Carderera et al., 2021a]
7. Specialized variants [Freund et al., 2017, Braun et al., 2017c, 2019b,a]

Conditional Gradients very competitive: simple, robust, real-world performance.

For more background etc see our survey!

[Braun et al., 2022]

**Conditional Gradient-based Identification
of Nonlinear Dynamics (CINDy)**
—Recovering Dynamics from Noisy Data—

joint work with Alejandro Carderera, Christof Schütte, Martin Weiser

[Carderera et al., 2021b]

Physical Systems via ODEs

CINDy: Recovering Dynamics from Noisy Data

Physical systems described by ordinary differential equation.

$$\dot{x}(t) = F(x(t)),$$

where $x(t) \in \mathbb{R}^d$ denotes the state of the system at time t .

Physical Systems via ODEs

CINDy: Recovering Dynamics from Noisy Data

Physical systems described by ordinary differential equation.

$$\dot{x}(t) = F(x(t)),$$

where $x(t) \in \mathbb{R}^d$ denotes the state of the system at time t .

Usually. $F : \mathbb{R}^d \rightarrow \mathbb{R}^d$ (usually) linear combination of simpler *ansatz functions* $\mathcal{D} = \{\psi_i \mid i \in \llbracket 1, n \rrbracket\}$ with $\psi_i : \mathbb{R}^d \rightarrow \mathbb{R}$:

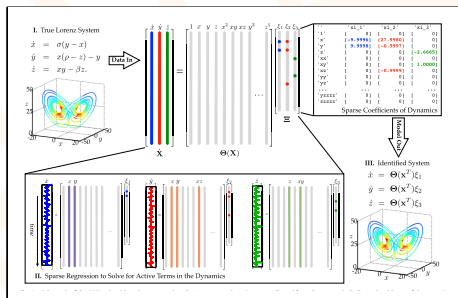
$$\dot{x}(t) = F(x(t)) = \Xi^T \psi(x(t)) = \begin{bmatrix} - & \xi_1 & - \\ & \vdots & \\ - & \xi_d & - \end{bmatrix} \begin{bmatrix} \psi_1(x(t)) \\ \vdots \\ \psi_n(x(t)) \end{bmatrix},$$

where $\Xi \in \mathbb{R}^{n \times d}$ is a typically sparse matrix and $\psi(x(t)) = [\psi_1(x(t)), \dots, \psi_n(x(t))]^T \in \mathbb{R}^n$.

Sparse Identification of Nonlinear Dynamics (SINDy)

CINDy: Recovering Dynamics from Noisy Data

[Brunton et al., 2016]



Focus on component-wise formulation of **sparse recovery problem**, and solve a relaxation of:

$$\min_{\xi_j \in \mathbb{R}^d} \sum_{i=1}^m \|\dot{x}_i - \xi_j^T \psi(x_i)\|_2^2 + \alpha \|\xi_j\|_0,$$

for each $j \in \llbracket 1, d \rrbracket$ for a suitably chosen $\alpha \geq 0$.

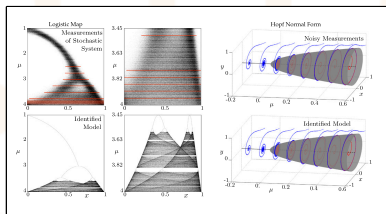
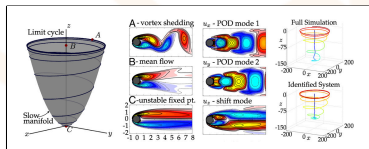
Note. Earlier approach via Gröbner/Border Bases for homogeneous.

[Heldt et al., 2009]

Sparse Identification of Nonlinear Dynamics (SINDy)

CINDy: Recovering Dynamics from Noisy Data

[Brunton et al., 2016]



Characteristics of SINDy.

1. Works on a **very wide** variety of dynamics
2. Recovers **sparse** dynamics very well in the noise-free case
3. **However** when data is noisy, picks up many auxiliary terms to *explain* noise.

The Fully-Corrective Frank-Wolfe Algorithm

CINDy: Recovering Dynamics from Noisy Data

Algorithm Fully-Corrective FW Algorithm (FCFW)

- 1: $x_0 \in P, \mathcal{S}_0 \leftarrow \{x_0\}$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $\mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
 - 5: $x_{t+1} \leftarrow \arg \min_{x \in \text{conv}(\mathcal{S}_{t+1})} f(x)$
 - 6: **end for**
-

[Holloway, 1974]

The Fully-Corrective Frank-Wolfe Algorithm

CINDy: Recovering Dynamics from Noisy Data

Algorithm Fully-Corrective FW Algorithm (FCFW)

- 1: $x_0 \in P, \mathcal{S}_0 \leftarrow \{x_0\}$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $\mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
 - 5: $x_{t+1} \leftarrow \arg \min_{x \in \text{conv}(\mathcal{S}_{t+1})} f(x)$
 - 6: **end for**
-

[Holloway, 1974]

The Fully-Corrective Frank-Wolfe Algorithm

CINDy: Recovering Dynamics from Noisy Data

Algorithm Fully-Corrective FW Algorithm (FCFW)

- 1: $x_0 \in P, \mathcal{S}_0 \leftarrow \{x_0\}$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $\mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
 - 5: $x_{t+1} \leftarrow \arg \min_{x \in \text{conv}(\mathcal{S}_{t+1})} f(x)$
 - 6: **end for**
-

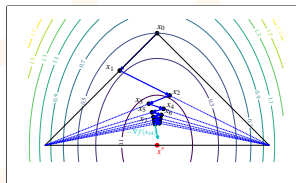
[Holloway, 1974]

The Fully-Corrective Frank-Wolfe Algorithm

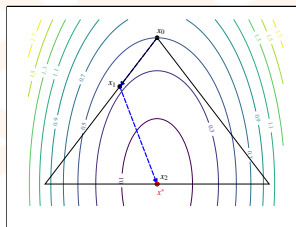
CINDy: Recovering Dynamics from Noisy Data

Algorithm Fully-Corrective FW Algorithm (FCFW)

- 1: $x_0 \in P, \mathcal{S}_0 \leftarrow \{x_0\}$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $\mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
 - 5: $x_{t+1} \leftarrow \arg \min_{x \in \text{conv}(\mathcal{S}_{t+1})} f(x)$
 - 6: **end for**
-



Vanilla FW Algorithm



Fully-Corrective FW Algorithm

[Holloway, 1974]

The Fully-Corrective Frank-Wolfe Algorithm

CINDy: Recovering Dynamics from Noisy Data

Algorithm Fully-Corrective FW Algorithm (FCFW)

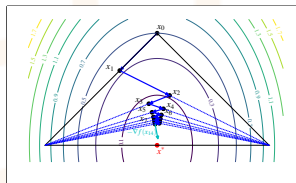
- 1: $x_0 \in P, \mathcal{S}_0 \leftarrow \{x_0\}$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: $v_t \leftarrow \arg \min_{v \in P} \langle \nabla f(x_t), v \rangle$
 - 4: $\mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
 - 5: $x_{t+1} \leftarrow \arg \min_{x \in \text{conv}(\mathcal{S}_{t+1})} f(x)$
 - 6: **end for**
-

[Holloway, 1974]

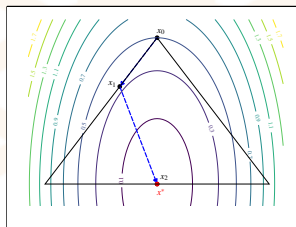
- **Sparsity:** FCFW offers much higher sparsity
- **Speed:** Convergence speed is (much) higher but iterations *very* costly
- **Projection-free:** While still projection-free requires solver for subproblems

\Rightarrow While expensive can be useful if sheer speed is not a priority but sparsity is.

Note. Sparsity *not only* a function of formulation but also *algorithm and its trajectory*.



Vanilla FW Algorithm

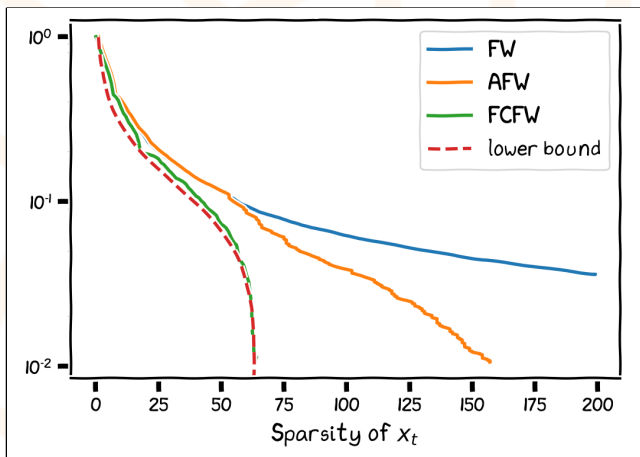


Fully-Corrective FW Algorithm

Sparsity of different FW variants

CINDy: Recovering Dynamics from Noisy Data

Example. Recovery of a signal x^* in $\|\cdot\|_7$ norm, i.e., objective $\|x - x^*\|_7^2$.



CINDy vs SINDy: comparison of the methods

CINDy: Recovering Dynamics from Noisy Data

SINDy.

Solves approximation of

$$\min_{\xi_j \in \mathbb{R}^d} \sum_{i=1}^m \|\dot{x}_i - \xi_j^T \psi(x_i)\|_2^2 + \alpha \|\xi_j\|_0,$$

via Least-Squares Step + Thresholding

CINDy.

Solves

$$\min_{\substack{\|\Xi\|_{1,1} \leq \alpha \\ \Xi \in \mathbb{R}^{n \times d}}} \|\dot{X} - \Xi^T \Psi(X)\|_F^2$$

via Fully-Corrective Frank-Wolfe (or similar)

CINDy vs SINDy: comparison of the methods

CINDy: Recovering Dynamics from Noisy Data

SINDy.

Solves approximation of

$$\min_{\xi_j \in \mathbb{R}^d} \sum_{i=1}^m \|\dot{x}_i - \xi_j^T \psi(x_i)\|_2^2 + \alpha \|\xi_j\|_0,$$

via Least-Squares Step + Thresholding

CINDy.

Solves

$$\min_{\substack{\|\Xi\|_{1,1} \leq \alpha \\ \Xi \in \mathbb{R}^{n \times d}}} \|\dot{X} - \Xi^T \Psi(X)\|_F^2$$

via Fully-Corrective Frank-Wolfe (or similar)

Advantages of CINDy.

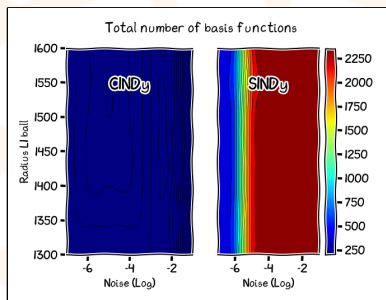
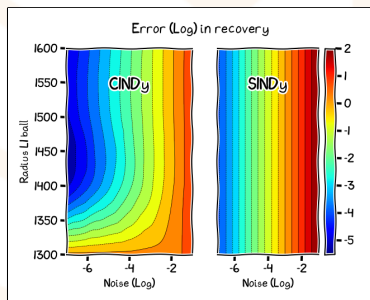
1. Better sparsity
2. Better noise tolerance
3. Allows for inclusion of additional constraints (e.g., conversation laws etc)
4. Control of coefficients due to simple ball with some radius α

CINDy vs SINDy: a recovery example

CINDy: Recovering Dynamics from Noisy Data

Kuramoto model. $d = 10$ weakly-coupled identical oscillators. For oscillator i :

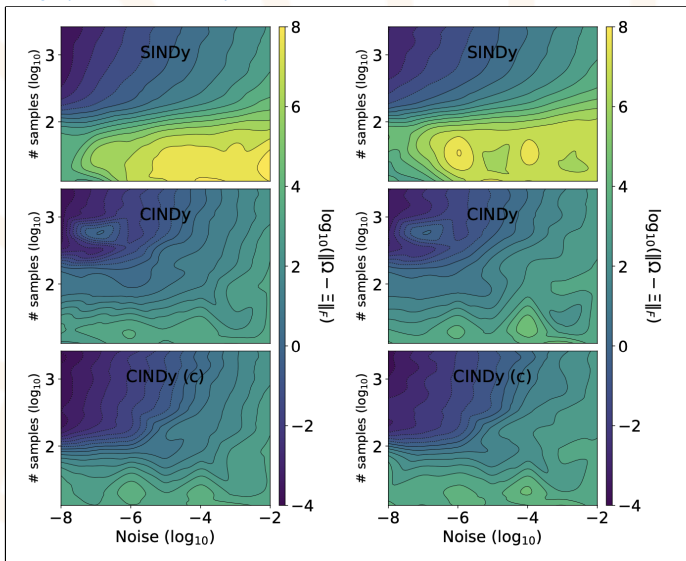
$$\dot{x}_i = \omega_i + \frac{K}{d} \sum_{j=1}^d \sin(x_j - x_i) + h \sin(x_i)$$



Number of data points. 3000 generated from 100 experiments (30 per experiment with additive random noise of 1.0^{-3}).

Sample Efficiency: Fermi-Pasta-Ulam-Tsingou model

CINDy: Recovering Dynamics from Noisy Data



Left: differential formulation / Right: integral formulation.

CINDy vs SINDy: sparsity matters - the most parsimonious model

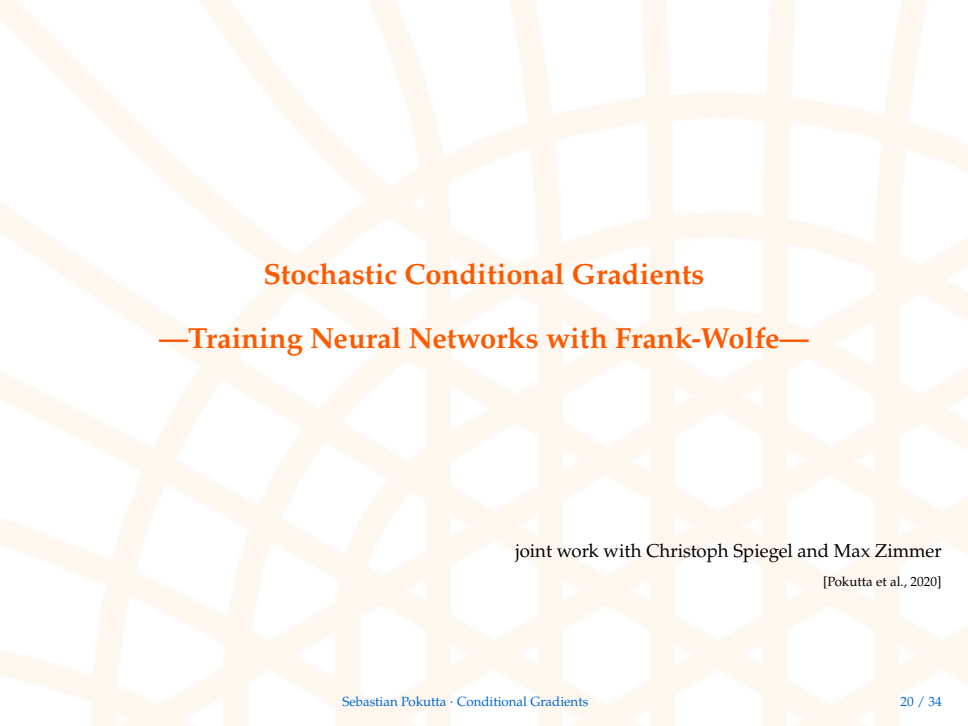
CINDy: Recovering Dynamics from Noisy Data

Fermi-Pasta-Ulam-Tsingou model.



Kuramoto model.





Stochastic Conditional Gradients

—Training Neural Networks with Frank-Wolfe—

joint work with Christoph Spiegel and Max Zimmer

[Pokutta et al., 2020]

The Stochastic Frank-Wolfe Algorithm (with Momentum)

Training Neural Networks with Conditional Gradients

Algorithm Stochastic FW Algorithm (SFW)

- 1: $m_0 \leftarrow 0$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: uniformly sample i.i.d. $i_1, \dots, i_{b_t} \sim \llbracket 1, m \rrbracket$
 - 4: $\tilde{\nabla}L(\theta_t) \leftarrow \frac{1}{b_t} \sum_{j=1}^{b_t} \nabla \ell_{i_j}(\theta_t)$
 - 5: $m_t \leftarrow (1 - \rho_t) m_{t-1} + \rho_t \tilde{\nabla}L(\theta_t)$
 - 6: $v_t \leftarrow \arg \min_{v \in P} \langle m_t, v \rangle$
 - 7: $\theta_{t+1} \leftarrow \theta_t + \alpha_t (v_t - \theta_t)$
 - 8: **end for**
-

e.g., [Reddi et al., 2016]

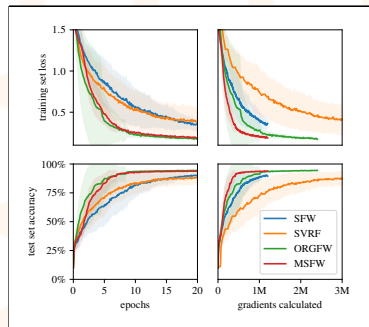
The Stochastic Frank-Wolfe Algorithm (with Momentum)

Training Neural Networks with Conditional Gradients

Algorithm Stochastic FW Algorithm (SFW)

- 1: $m_0 \leftarrow 0$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: uniformly sample i.i.d. $i_1, \dots, i_{b_t} \sim \llbracket 1, m \rrbracket$
 - 4: $\tilde{\nabla}L(\theta_t) \leftarrow \frac{1}{b_t} \sum_{j=1}^{b_t} \nabla \ell_{i_j}(\theta_t)$
 - 5: $m_t \leftarrow (1 - \rho_t) m_{t-1} + \rho_t \tilde{\nabla}L(\theta_t)$
 - 6: $v_t \leftarrow \arg \min_{v \in P} \langle m_t, v \rangle$
 - 7: $\theta_{t+1} \leftarrow \theta_t + \alpha_t (v_t - \theta_t)$
 - 8: **end for**
-

e.g., [Reddi et al., 2016]



SFW variants

The Stochastic Frank-Wolfe Algorithm (with Momentum)

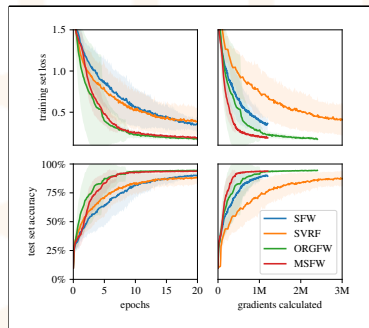
Training Neural Networks with Conditional Gradients

Algorithm Stochastic FW Algorithm (SFW)

- 1: $m_0 \leftarrow 0$
 - 2: **for** $t = 0$ **to** $T - 1$ **do**
 - 3: uniformly sample i.i.d. $i_1, \dots, i_{b_t} \sim \llbracket 1, m \rrbracket$
 - 4: $\tilde{\nabla}L(\theta_t) \leftarrow \frac{1}{b_t} \sum_{j=1}^{b_t} \nabla \ell_{i_j}(\theta_t)$
 - 5: $m_t \leftarrow (1 - \rho_t) m_{t-1} + \rho_t \tilde{\nabla}L(\theta_t)$
 - 6: $v_t \leftarrow \arg \min_{v \in P} \langle m_t, v \rangle$
 - 7: $\theta_{t+1} \leftarrow \theta_t + \alpha_t (v_t - \theta_t)$
 - 8: **end for**
-

e.g., [Reddi et al., 2016]

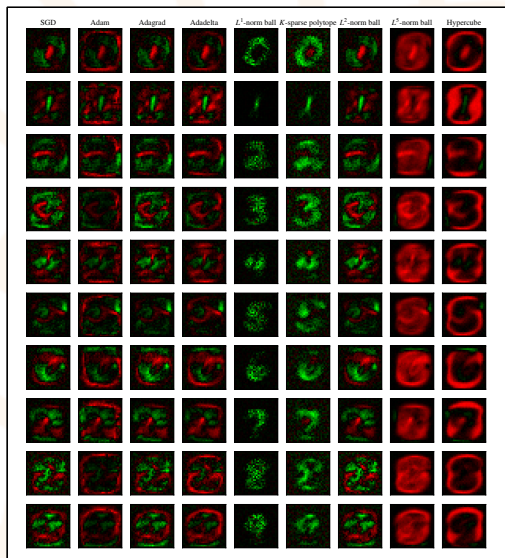
- **Convergence rate:** In the non-convex stochastic smooth case $O(1/\sqrt{T})$ -rate
- **Speed:** Works well for very large data sets due to mini-batched gradients
- **Projection-free:** Remains projection-free and allows for constraints



SFW variants

Relevance maps under different optimizers / feasible regions

Training Neural Networks with Conditional Gradients



Robust Rate-Distortion Explanations via Conditional Gradients

joint work with Mathieu Besançon and Jan Macdonald

[Macdonald et al., 2022]

Rate-Distortion Explanation: the problem formulation

Discrete Optimization in Machine Learning

[Macdonald et al., 2019]

Expected Distortion of S .

$$D(S) := D(S, \Phi, x, V) := \underbrace{\mathbb{E}_{y \sim V} \left[\frac{1}{2} (\Phi(x) - \Phi(y))^2 \right]}_{\text{Stability of } \Phi \text{ when varying outside of } S}$$

Rate-Distortion Explanation: the problem formulation

Discrete Optimization in Machine Learning

[Macdonald et al., 2019]

Expected Distortion of S .

$$D(S) := D(S, \Phi, x, V) := \underbrace{\mathbb{E}_{y \sim V} \left[\frac{1}{2} (\Phi(x) - \Phi(y))^2 \right]}_{\text{Stability of } \Phi \text{ when varying outside of } S}$$



Rate-Distortion Explanation: the problem formulation

Discrete Optimization in Machine Learning

[Macdonald et al., 2019]

Expected Distortion of S .

$$D(S) := D(S, \Phi, x, V) := \underbrace{\mathbb{E}_{y \sim V} \left[\frac{1}{2} (\Phi(x) - \Phi(y))^2 \right]}_{\text{Stability of } \Phi \text{ when varying outside of } S}$$



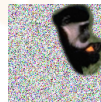
Rate-Distortion Explanation: the problem formulation

Discrete Optimization in Machine Learning

[Macdonald et al., 2019]

Expected Distortion of S .

$$D(S) := D(S, \Phi, x, V) := \underbrace{\mathbb{E}_{y \sim V} \left[\frac{1}{2} (\Phi(x) - \Phi(y))^2 \right]}_{\text{Stability of } \Phi \text{ when varying outside of } S}$$



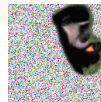
Rate-Distortion Explanation: the problem formulation

Discrete Optimization in Machine Learning

[Macdonald et al., 2019]

Expected Distortion of S .

$$D(S) := D(S, \Phi, x, V) := \underbrace{\mathbb{E}_{y \sim V} \left[\frac{1}{2} (\Phi(x) - \Phi(y))^2 \right]}_{\text{Stability of } \Phi \text{ when varying outside of } S}$$



Rate-Distortion function.

$$R(\varepsilon) := \min \{ \underbrace{\text{card}(S) : D(S) \leq \varepsilon}_{\text{smallest set of fixings } S} \}$$

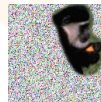
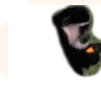
Rate-Distortion Explanation: the problem formulation

Discrete Optimization in Machine Learning

[Macdonald et al., 2019]

Expected Distortion of S .

$$D(S) := D(S, \Phi, x, V) := \underbrace{\mathbb{E}_{y \sim V} \left[\frac{1}{2} (\Phi(x) - \Phi(y))^2 \right]}_{\text{Stability of } \Phi \text{ when varying outside of } S}$$



Rate-Distortion function.

$$R(\varepsilon) := \min \{ \underbrace{\text{card}(S) : D(S) \leq \varepsilon}_{\text{smallest set of fixings } S} \}$$

After convex relaxation (original problem is hard).

$$\underbrace{\min \{ D(s) : \|s\|_1 \leq \lambda \}}$$

given budget λ find s with lowest distortion aka most relevant pixels

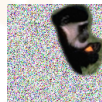
Rate-Distortion Explanation: the problem formulation

Discrete Optimization in Machine Learning

[Macdonald et al., 2019]

Expected Distortion of S .

$$D(S) := D(S, \Phi, x, V) := \underbrace{\mathbb{E}_{y \sim V} \left[\frac{1}{2} (\Phi(x) - \Phi(y))^2 \right]}_{\text{Stability of } \Phi \text{ when varying outside of } S}$$



Rate-Distortion function.

$$R(\varepsilon) := \min \{ \underbrace{\text{card}(S) : D(S) \leq \varepsilon}_{\text{smallest set of fixings } S} \}$$

After convex relaxation (original problem is hard).

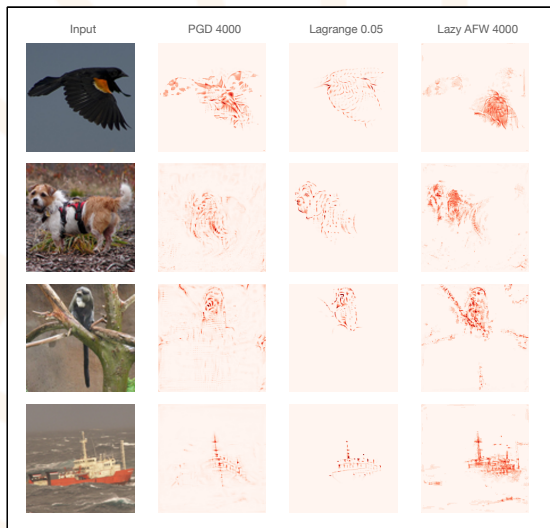
$$\underbrace{\min \{ D(s) : \|s\|_1 \leq \lambda \}}$$

given budget λ find s with lowest distortion aka most relevant pixels

⇒ Structured optimization problem over ℓ_1 -ball.

Rate-Distortion Explanation: Examples

Discrete Optimization in Machine Learning



All methods had the same budget for picking relevant pixels. However, sparser solutions of Conditional Gradients focus weight on most relevant pixels rather than spreading out.

Rate-Distortion Explanation: Ordered Relevance

Discrete Optimization in Machine Learning

While a good first step, often not sufficient.

Obtain Ordered Relevance. Solve structured problem over Birkhoff polytope to obtain ordered relevance.

[Macdonald et al., 2022]

Rate-Distortion Explanation: Ordered Relevance

Discrete Optimization in Machine Learning

While a good first step, often not sufficient.

Obtain Ordered Relevance. Solve structured problem over Birkhoff polytope to obtain ordered relevance. [Macdonald et al., 2022]

Note. Works only(!) for Frank-Wolfe variant as explicit ℓ_1 -constraint.

Rate-Distortion Explanation: Ordered Relevance

Discrete Optimization in Machine Learning

While a good first step, often not sufficient.

Obtain Ordered Relevance. Solve structured problem over Birkhoff polytope to obtain ordered relevance.

[Macdonald et al., 2022]

Note. Works only(!) for Frank-Wolfe variant as explicit ℓ_1 -constraint.



Rate-Distortion Explanation: Ordered Relevance

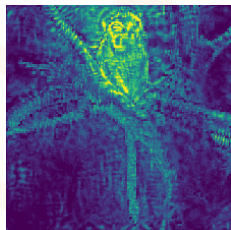
Discrete Optimization in Machine Learning

While a good first step, often not sufficient.

Obtain Ordered Relevance. Solve structured problem over Birkhoff polytope to obtain ordered relevance.

[Macdonald et al., 2022]

Note. Works only(!) for Frank-Wolfe variant as explicit ℓ_1 -constraint.



Rate-Distortion Explanation: Ordered Relevance

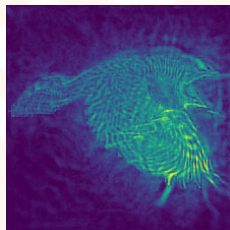
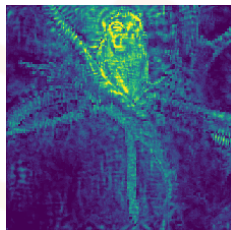
Discrete Optimization in Machine Learning

While a good first step, often not sufficient.

Obtain Ordered Relevance. Solve structured problem over Birkhoff polytope to obtain ordered relevance.

[Macdonald et al., 2022]

Note. Works only(!) for Frank-Wolfe variant as explicit ℓ_1 -constraint.



Rate-Distortion Explanation: Ordered Relevance

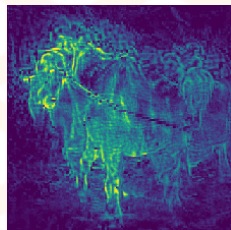
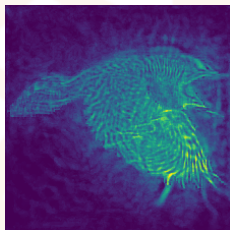
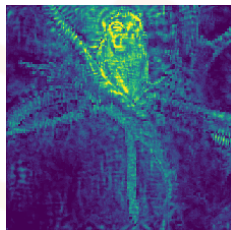
Discrete Optimization in Machine Learning

While a good first step, often not sufficient.

Obtain Ordered Relevance. Solve structured problem over Birkhoff polytope to obtain ordered relevance.

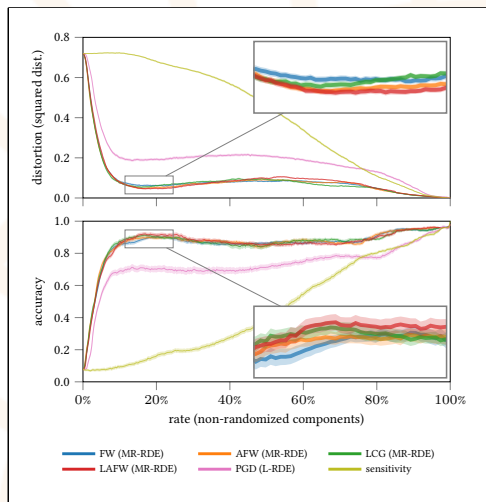
[Macdonald et al., 2022]

Note. Works only(!) for Frank-Wolfe variant as explicit ℓ_1 -constraint.



Rate-Distortion Explanation: Ordered Relevance Test

Discrete Optimization in Machine Learning



Relevance ordering test results for STL-10. An average result over 50 images from the test set (5 images per class) and 64 noise input samples per image is shown (shaded regions mark \pm standard deviation).

FrankWolfe.jl

**a high-performance Julia package
for Conditional Gradients**

joint work with Mathieu Besançon and Alejandro Carderera

[Besançon et al., 2022]

Overview and Features

FrankWolfe.jl

In a nutshell.

- Implemented in Julia
- Open source under MIT License
- Generic numeric types: reduced (16, 32, 64 bits) and extended (128 bits, GNU MP) precision, rationals
- Memory-saving mode, in-place gradient computations
- Scales well (solved some problems with 1B variables)
- Switch components - bring your own LMO / ∇f / step size.

Give it a try.

```
using Pkg
Pkg.add("FrankWolfe")
```

Example

FrankWolfe.jl

```
using LinearAlgebra
using FrankWolfe

n = 1000
xp = rand(n)

f(x) = norm(x - xp)^2

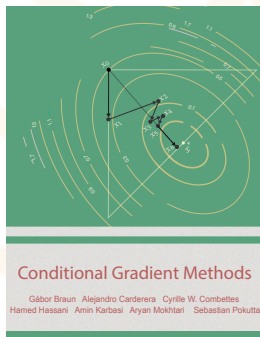
function grad!(storage, x)
    @. storage = 2 * (x - xp)
    return nothing
end

# create a L_1-norm ball of radius 2.5
lmo_radius = 2.5
lmo = FrankWolfe.LpNormLMO{Float64,1}(lmo_radius)

x0 = FrankWolfe.compute_extreme_point(lmo, zeros(n))

x_sol, _ = frank_wolfe(f, grad!, lmo, x0)
```

Thank you!



Conditional Gradient Methods

Gábor Braun, Alejandro Carderera, Cyrille W Combettes, Hamed Hassani, Amin Karbasi, Aryan Mokhtari, and Sebastian Pokutta

<https://conditional-gradients.org/>
<https://arxiv.org/abs/2211.14103>

References I

- F. Bach. On the effectiveness of Richardson extrapolation in machine learning. *arXiv preprint 2002.02835v3*, July 2020.
- M. Besançon, A. Carderera, and S. Pokutta. FrankWolfe.jl: A high-performance and flexible toolbox for Frank–Wolfe algorithms and conditional gradients. *INFORMS Journal on Computing*, 2 2022. URL <https://pubsonline.informs.org/doi/abs/10.1287/ijoc.2022.1191>. [slides].
- G. Braun and S. Pokutta. The matching polytope does not admit fully-polynomial size relaxation schemes. In *Proceedings of SODA*, 2015a.
- G. Braun and S. Pokutta. The matching polytope does not admit fully-polynomial size relaxation schemes. *IEEE Transactions on Information Theory*, 61(10): 5754–5764, 2015b. doi: 10.1109/TIT.2015.2465864.
- G. Braun, S. Pokutta, and D. Zink. Inapproximability of combinatorial problems via small LPs and SDPs. *Proceedings of STOC*, 2015.
- G. Braun, C. Guzmán, and S. Pokutta. Unifying Lower Bounds on the Oracle Complexity of Nonsmooth Convex Optimization. *IEEE Transactions of Information Theory*, 63(7):4709–4724, 2017a.
- G. Braun, R. Jain, T. Lee, and S. Pokutta. Information-theoretic approximations of the nonnegative rank. *Computational Complexity*, 26(1):147–197, 2017b.
- G. Braun, S. Pokutta, and D. Zink. Lazifying conditional gradient algorithms. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 566–575, 2017c. [slides].
- G. Braun, S. Pokutta, D. Tu, and S. Wright. Blended conditional gradients: The unconditioning of conditional gradients. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, pages 735–743, 2019a. URL <http://proceedings.mlr.press/v97/braun19a/braun19a.pdf>. [slides].
- G. Braun, S. Pokutta, and D. Zink. Lazifying Conditional Gradient Algorithms. *Journal of Machine Learning Research (JMLR)*, 20(71):1–42, 2019b. [slides].
- G. Braun, A. Carderera, C. W. Combettes, H. Hassani, A. Karbasi, A. Mokthari, and S. Pokutta. Conditional gradient methods. *preprint available at <https://arxiv.org/abs/2211.14103>*, 11 2022.
- S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.
- A. Carderera, J. Diakonikolas, C. Y. Lin, and S. Pokutta. Parameter-free locally accelerated conditional gradients. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, pages 1283–1293, 2 2021a. [slides].
- A. Carderera, S. Pokutta, C. Schütte, and M. Weiser. CINDy: Conditional gradient-based identification of non-linear dynamics – noise-robust recovery, 1 2021b. Preprint.
- C. W. Combettes and S. Pokutta. Complexity of linear minimization and projection on some sets. *Operations Research Letters*, 49, 7 2021.
- C. W. Combettes, C. Spiegel, and S. Pokutta. Projection-Free Adaptive Gradients for Large-Scale Optimization. *preprint*, 10 2020.
- J. Diakonikolas, A. Carderera, and S. Pokutta. Locally Accelerated Conditional Gradients. In S. Chiappa and R. Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1737–1747, 2020. URL <http://proceedings.mlr.press/v108/diakonikolas20a/diakonikolas20a.pdf>. [slides].

References II

- M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956.
- R. M. Freund, P. Grigas, and R. Mazumder. An extended Frank-Wolfe method with “in-face” directions, and its application to low-rank matrix completion. *SIAM Journal on Optimization*, 27(1):319–346, 2017.
- D. Garber and E. Hazan. Playing non-linear games with linear oracles. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 420–428. IEEE, 2013.
- D. Garber and O. Meshi. Linear-memory and decomposition-invariant linearly convergent conditional gradient algorithm for structured polytopes. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1001–1009. Curran Associates, Inc., 2016. URL <http://papers.nips.cc/paper/6115-linear-memory-and-decomposition-invariant-linearly-convergent-conditional-gradient-algorithm-for-structured-pdf>.
- E. Hazan and S. Kale. Projection-free online learning. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.
- E. Hazan and H. Luo. Variance-reduced and projection-free stochastic optimization. In *International Conference on Machine Learning*, pages 1263–1271, 2016.
- D. Heldt, M. Kreuzer, S. Pokutta, and H. Poulisse. Approximate computation of zero-dimensional polynomial ideals. *Journal of Symbolic Computation*, 44: 1566–1591, 2009.
- C. A. Holloway. An extension of the Frank and Wolfe method of feasible directions. *Mathematical Programming*, 6:14–27, Dec. 1974. doi: 10.1007/BF01580219.
- M. Jaggi. Revisiting Frank-Wolfe: projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning*, pages 427–435, 2013.
- T. Kerdreux, A. d’Aspremont, and S. Pokutta. Restarting Frank-Wolfe. In K. Chaudhuri and M. Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 1275–1283, 2019. URL <http://proceedings.mlr.press/v89/kerdreux19a/kerdreux19a.pdf>. [slides].
- T. Kerdreux, A. d’Aspremont, and S. Pokutta. Projection-free optimization on uniformly convex sets. In A. Banerjee and K. Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 19–27, 1 2021a. [slides].
- T. Kerdreux, A. d’Aspremont, and S. Pokutta. Local and Global Uniform Convexity Conditions. *preprint*, 2 2021b.
- S. Lacoste-Julien. Convergence rate of Frank-Wolfe for non-convex objectives. *arXiv preprint arXiv:1607.00345*, 2016.
- S. Lacoste-Julien and M. Jaggi. On the global linear convergence of Frank-Wolfe optimization variants. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 496–504. Curran Associates, Inc., 2015. URL <http://papers.nips.cc/paper/5925-on-the-global-linear-convergence-of-frank-wolfe-optimization-variants.pdf>.
- G. Lan and Y. Zhou. Conditional gradient sliding for convex optimization. *SIAM Journal on Optimization*, 26(2):1379–1409, 2016.
- E. S. Levitin and B. T. Polyak. Constrained minimization methods. *USSR Computational Mathematics and Mathematical Physics*, 6(5):1–50, 1966.

References III

- J. Macdonald, S. Wäldchen, S. Hauch, and G. Kutyniok. A rate-distortion framework for explaining neural network decisions. *arXiv preprint arXiv:1905.11092*, 2019.
- J. Macdonald, M. Besançon, and S. Pokutta. Interpretable neural networks with Frank-Wolfe: Sparse relevance maps and relevance orderings, 5 2022. To appear in *Proceedings of ICML*.
- S. Pokutta, C. Spiegel, and M. Zimmer. Deep Neural Network Training with Frank-Wolfe. *preprint*, 10 2020.
- S. J. Reddi, S. Sra, B. Póczos, and A. Smola. Stochastic Frank-Wolfe methods for nonconvex optimization. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1244–1251. IEEE, 2016.
- T. Rothvoss. The matching polytope has exponential extension complexity. In *Symposium on Theory of Computing*, pages 263–272, 2014.