Frank-Wolfe algorithms & constraint structure exploitation

Mathieu Besançon @matbesancon - besancon@zib.de,
Alejandro Carderera, Jan Macdonald, Deborah Hendrych, Hannah Troppens, Jonathan Eckstein, Zev Woodstock, Sebastian Pokutta

September 7, 2022

Zuse Institute Berlin - AIS^{2}T
TU Berlin, Georgia Tech, FU Berlin, Rutgers U.

Outline

Table of Contents

Considered problems for Frank-Wolfe

Problem class considered:

$$(P) : \min_{x} f(x)$$
$$\text{s.t. } x \in C$$

with:

- C: compact convex set,
- f: continuously differentiable on C.

Key requirement:
Optimizing a linear function over C much cheaper than (P) itself.
Linear Minimization Oracle (LMO):

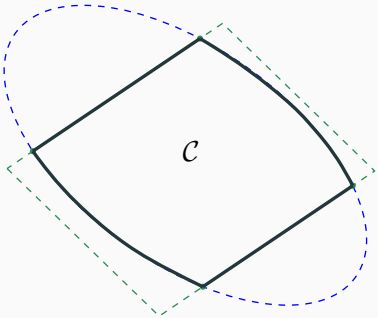
$$d \to v \in \arg\min_{y \in C} \langle y, d \rangle.$$

Frank-Wolfe structure and conic optimization

Frank-Wolfe:

$$(P) : \min_x f(x) \text{ s.t. } x \in C$$

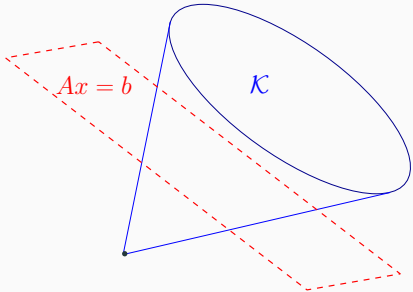
C compact convex (bounded).



Conic optimization:

$$(C) : \min_x \langle c, x \rangle \text{ s.t. } Ax = b, x \in \mathcal{K}$$

$\mathcal{K}$ proper cone.



Frank-Wolfe template

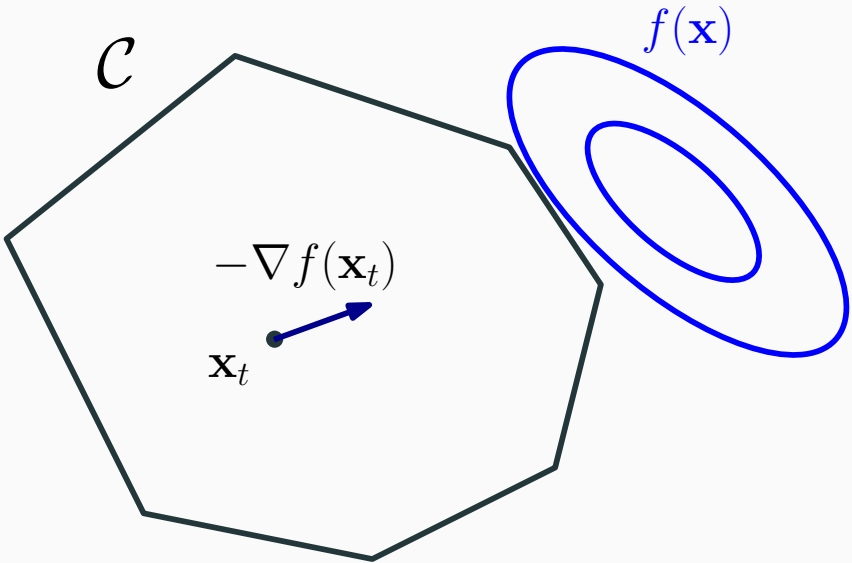
Algorithm 1.1 Frank-Wolfe algorithm

Require: Point $x_0 \in C$, function f.

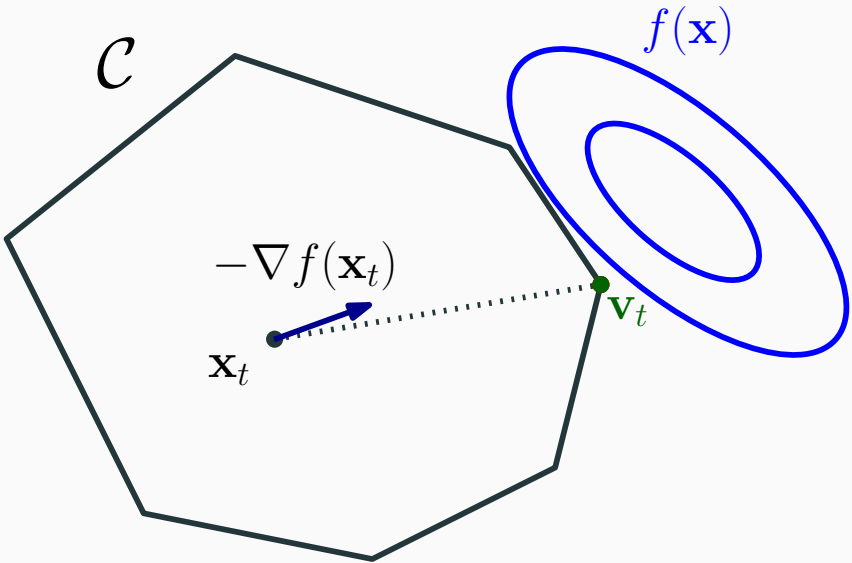
Ensure: Iterates $x_1, \cdots \in C$.

1: **for** $t = 0$ **to** $\ldots$ **do**
2: $\quad d_t \leftarrow \text{FirstOrderEstimate}(f, x_t)$
3: $\quad v_t \leftarrow \arg\min_{v \in C} \langle d_t, v \rangle$
4: $\quad \gamma_t \leftarrow \text{StepSizeStrategy}(x_t, t, d_t, v_t)$
5: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
6: **end for**

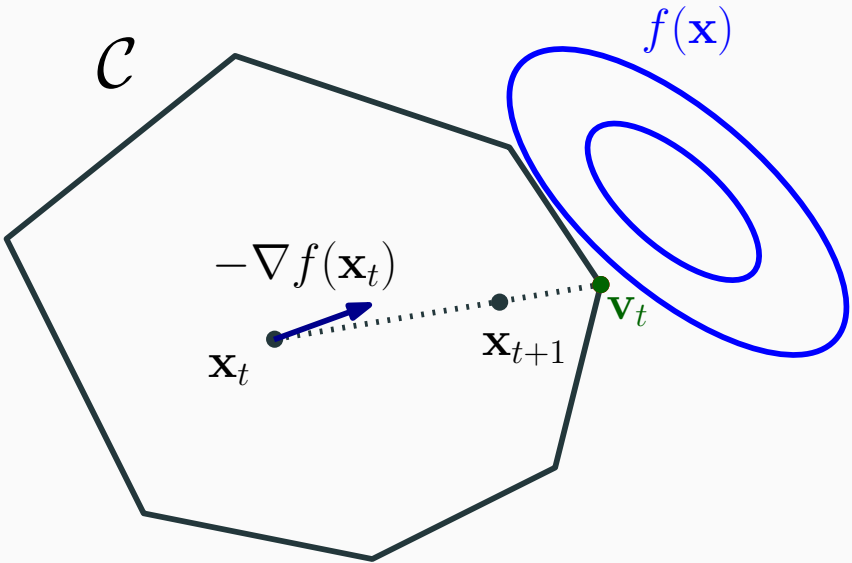
FW, visualized



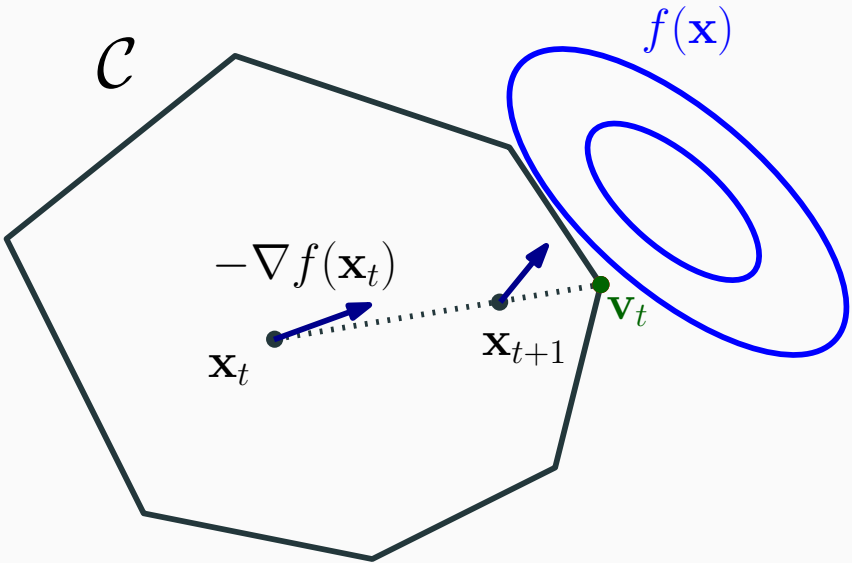
FW, visualized



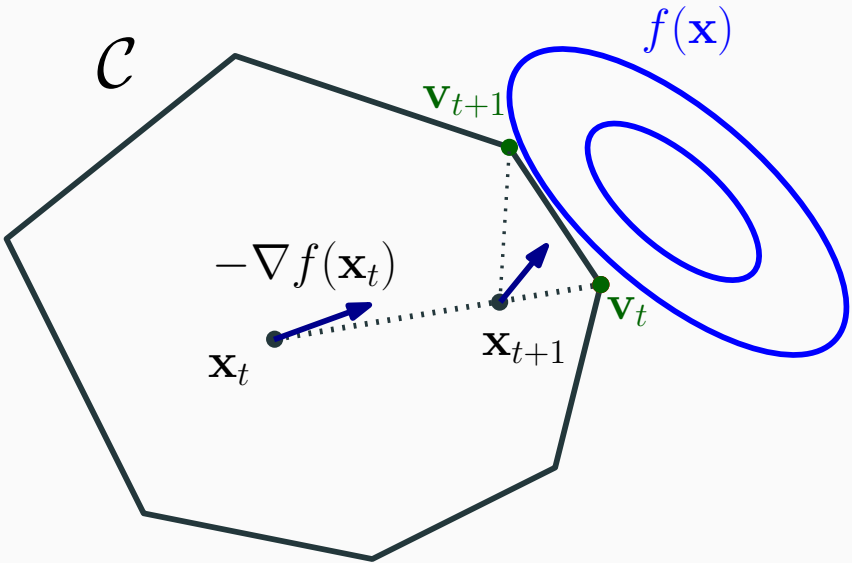
FW, visualized



FW, visualized

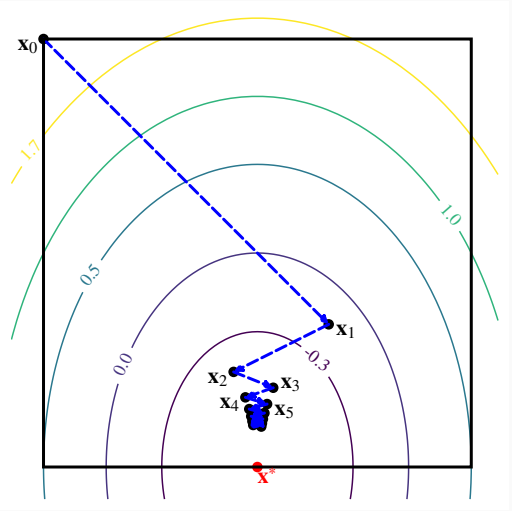


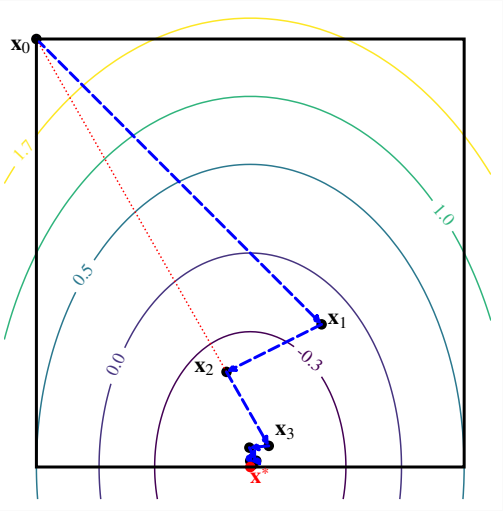
FW, visualized



Active set representations

Away FW, Blended (Pairwise) CG, ...





Linear Minimization Oracle

How hard is optimizing a linear function?

- Generic polytope $\Rightarrow$ LP methods: (primal) simplex, interior points
- Simple sets $\Rightarrow$ closed-form solutions
  $\ell_{1,2,\infty}$ norm balls, simplex...

Requirement: black box computation of primal value only
$\rightarrow$ e.g. convex hull of Mixed-Integer Problems.

Table of Contents

FrankWolfe.jl: yet another implementation?

FW algorithms: deceptive simplicity but no de-facto implementation.
Consequence: reinventing the wheel, potential bugs, performance variability, etc.
Goal: one central toolbox for:

1. **Practitioners** solving optimization problems fitting the form (P),
2. **Researchers** on Frank-Wolfe type algorithms developing new methods.

One-slide package summary:

Implemented in Julia:

- Compiled to native code, reaches C-like performance;
- Highly generic thanks to multiple dispatch;
- Generic numeric types: reduced (16, 32, 64 bits) and extended (128, GNU MP) precision, rationals;
- Memory-saving mode, in-place gradient computations;
- Switchable components - bring your own LMO / gradient / step size.
- MathOptInterface & JuMP compatibility, soon implementation

Main variants

Variant	Convergence		Sparsity	Numerical Stability	Active Set?	Lazy?
	Progress/iter.	Time/iter.				
FW	Low	Low	Low	High	No	Yes
Away FW	Medium	Medium-High	Medium	Medium-High	Yes	Yes
Stoch. FW	Low	Low	Low	High	No	No
Blended CG	High	Medium	High	Medium	Yes	By design
B-Pair. CG	Medium+	Low	High	High	Yes	By design

More variants

Bring your own component:

1: $d_t \leftarrow \text{FirstOrderEstimate}(f, x_t)$
2: $v_t \leftarrow \arg\min_{v \in C} \langle d_t, v \rangle$
3: $\gamma_t \leftarrow \text{StepSizeStrategy}(x_t, t, d_t, v_t)$

Each component has predefined types (simplex LMO, agnostic step size...) and a way to define your own.

Example usage

$$\min_{x \in \Delta} \frac{1}{2}\|x - y\|^2$$

```
using FrankWolfe
using LinearAlgebra
n = 1000
y = randn(n)
function f(x)
    return 1/2 * norm(x - y)^2
end
function grad!(storage, x)
    # perform entrywise without temporary allocation
    @. storage = x - y
end
```

Example usage (2)

```
lmo = FrankWolfe.ProbabilitySimplexOracle(1.0)
x0 = FrankWolfe.compute_extreme_point(lmo, zeros(n))
xfinal, vfinal, primal_value, _ = FrankWolfe.frank_wolfe(
    f, grad!, lmo, x0,
    max_iterations=1000, epsilon=10^-8, # other options
)
```

FrankWolfe.jl, how and where?

Registered on the official Julia package registry:

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

using FrankWolfe
```

Open-source license (MIT)
Available on https://github.com/ZIB-IOL/FrankWolfe.jl
Software paper: https://arxiv.org/abs/2104.06675

More complex feasible region, no closed-form solution?
Accepts problems defined through MathOptInterface / JuMP

Table of Contents

Frank-Wolfe and generalized self-concordance

TL;DR:

- Generalized self-concordance: useful class of functions, looser assumptions
- Frank-Wolfe converges with the usual $1/t$ rate
- Highlight that a simple twist on $2/(t+2)$ step size converges
- Works great (and robustly) on computational experiments.

Article (NeurIPS 2021):
Simple steps are all you need: Frank-Wolfe and generalized self-concordant functions
https://arxiv.org/abs/2105.13913

Assumptions for FW vs. self-concordance

- σ-smoothness - Lipschitz-continuous gradient
- μ-strong convexity

$$\mu I \preccurlyeq \nabla^2 f(x) \preccurlyeq \sigma I$$

(M, ν) generalized self-concordance: $f \in C^3$ (dom f), closed convex with dom $f \subseteq \mathbb{R}^n$ open.

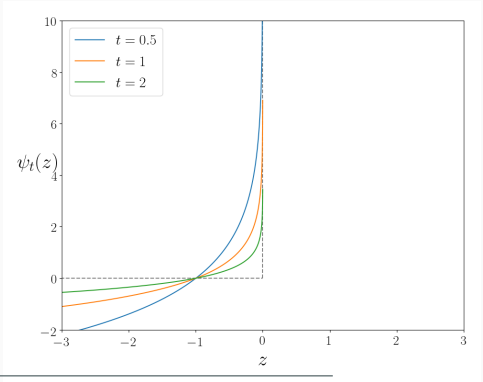
$$|\langle D^3 f(x)[w]u, u\rangle| \leq M\|u\|^2_{\nabla^2 f(x)} \|w\|^{\nu-2}_{\nabla^2 f(x)} \|w\|_2^{3-\nu} \quad \forall x \in \text{dom } f, u, w \in \mathbb{R}^n$$

Univariate case:

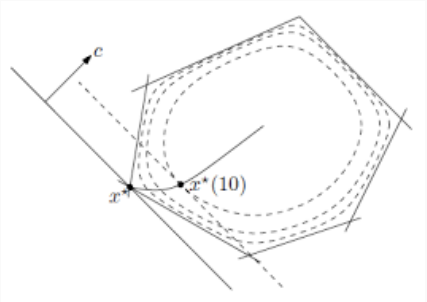
$$|\phi'''(t)| \leq M\phi''(t)^{\frac{\nu}{2}}$$

Examples?

- Logistic regression
- KL divergence between probability measures
- **Log-barrier functions**:



1



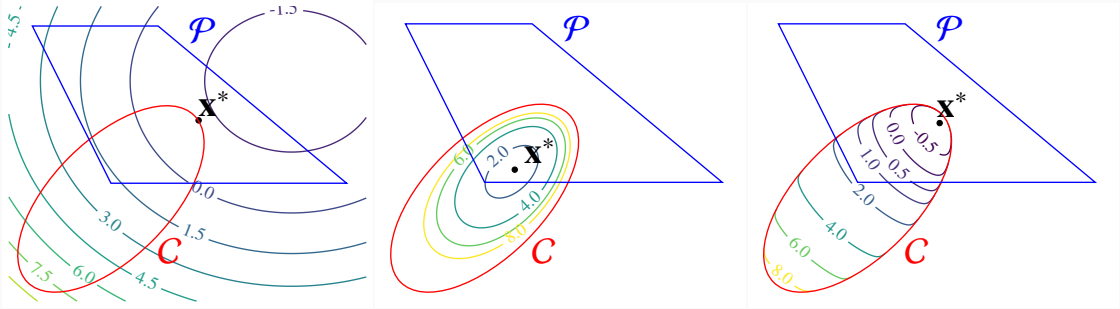
2

[1] Source: https://deepai.org/publication/log-barrier-constrained-cnns

[2] Source: http://www.stat.cmu.edu/ ryantibs/convexopt-S15

Exploiting barrier functions

Improving constraint structure through the objective



Prior work

- Focus on **second-order methods**
- (Generalized) self-concordant property designed for Newton-type algorithms [Sun, 2018, Sun and Tran-Dinh, 2019]
- First work on Frank-Wolfe but Hessian information required for step size [Dvurechensky et al., 2020, Liu et al., 2020].

Our contribution

Typical open loop step size strategy:

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

Monotonic variant:

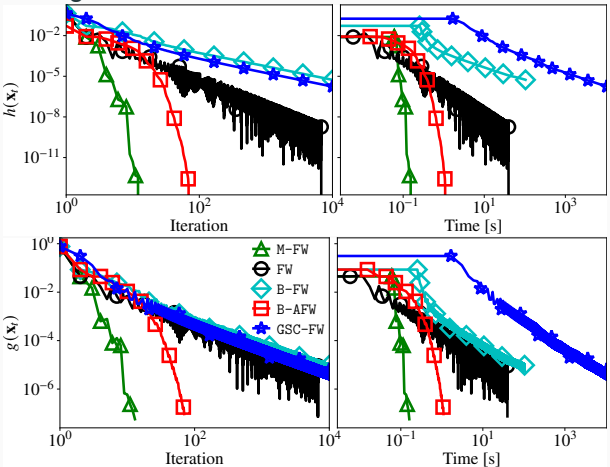
$$\gamma_t := \frac{2}{2+t}$$
$$\text{while } x_t + \gamma_t(v_t - x_t) \notin \operatorname{dom} f \text{ and } f(x_t) \geq f(x_t + \gamma_t(v_t - x_t)) : \gamma_t = \frac{\gamma_t}{2}$$

- Converges with $O(1/t)$, using a **Domain Oracle**.
- Improved convergence for uniformly convex sets.
- Convergence for Away-step FW.

Computational results

TL;DR: Away-step usually the fastest converging method, monotonic competitive, second-order too expensive and not much per-iteration progress.

Example on a logistic regression:

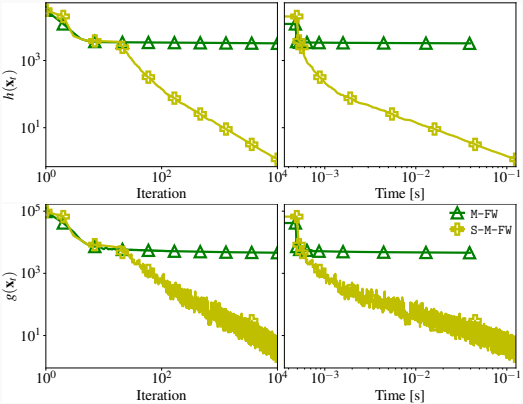


Stateless / restarted simple step

Bad directions early → smaller step size throughout all iterations

Potentially a problem for ill-conditioned instances.

$$x^T Qx + \langle b, x \rangle + \mu \varphi_{\mathbb{R}_+}(x)$$



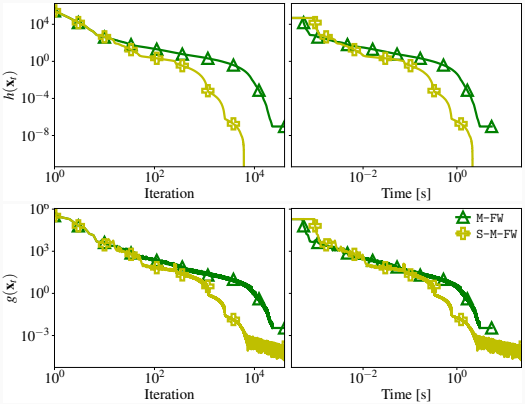


Table of Contents

Application to interpretable ML: Rate-Distortion Explanation

Interpretable Neural Networks with Frank-Wolfe:
Sparse Relevance Maps and Relevance Orderings, ICML 2022,
https://arxiv.org/abs/2110.08105.

Work spearheaded by Jan Macdonald, with Sebastian Pokutta,

Which input features of a prediction model matter (and which don't)?

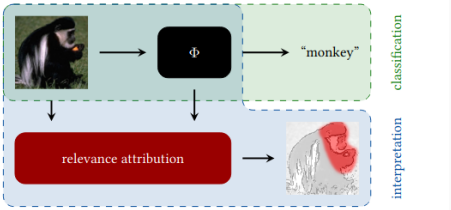
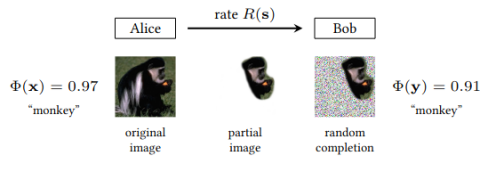


Figure 1: Relevance attribution methods aim at rendering black-box classifiers more interpretable by providing heatmaps of the input features that contribute most to an individual prediction.

Formulation



$$\min_{s} D(s)$$
$$\text{s.t. } \|s\|_1 \leq k,$$
$$s \in [0,1]^n$$

with $D(\cdot)$ the *distortion* of the prediction:
expected change when setting inactive feature to random noise.

RDE Formulation

$$\min_{s\in[0,1]^n} D(s) + \lambda\|s\|_1 \quad (L - RDE)$$

$$\Downarrow$$

$$\min_{s\in[0,1]^n} D(s) \text{ s.t. } \|s\|_1 \leq k \quad (RC - RDE)$$

Fixing a **rate** by an explicit constraint.

Both the formulation and the algorithm influence the solution structure:
FW favors sparsity by moving iterates in sparse subspaces (unlike prox methods).

Single to multirate RDE

(RC-RDE) requires a rate k: how many features do we maintain?
What about aggregating relevance maps s over multiple rates?

s alone has no meaning but provides priorities / ordering over features.
What about optimizing an ordering directly?

$$\min_{\Pi \in B_n} \sum_{k \in 2..n-1} D(\Pi p_k)$$

with B_n the set of doubly stochastic matrices (convex hull of permutation matrices), p_k selects the k first items.
B_n has a well-defined LMO but no efficient projection.

Single to multirate RDE

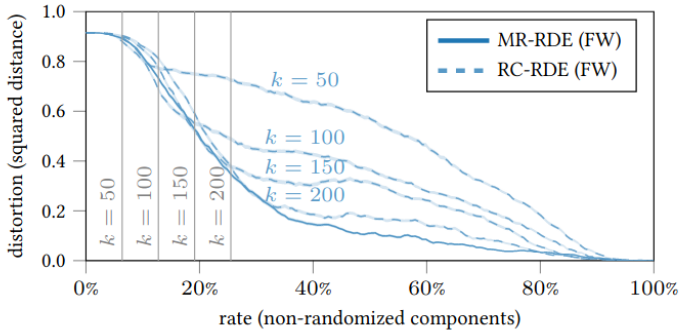


Figure 5: Relevance ordering test results for MNIST and (RC-RDE) at various rates. Vertical lines show the rates k at which the mappings were optimized. The combined (MR-RDE) solution approximates a lower envelope of the individual curves. An average result over 50 images from the test set (5 images per class) and 512 noise input samples per image is shown (shaded regions mark $\pm$ standard deviation).

Table of Contents

Branching over Frank-Wolfe

With Deborah Hendrych, Hannah Troppens (FU Berlin & ZIB), Sebastian Pokutta
`arxiv.org/abs/2208.11010`, *Convex integer optimization with FW methods*
Package available at `ZIB-IOL/Boscia.jl`
Applications in engineering, sparse prediction models, statistics.

Problem setting

Nonlinear objective + polyhedron & integer variables.

$$\begin{aligned} \min_{x} \; & f(x) \\ \text{s.t. } & x \in \mathcal{X} \\ & x_j \in \mathbb{Z} \; \forall j \in J \end{aligned}$$

Problem setting

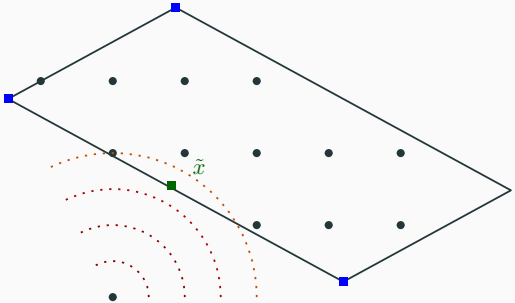
Nonlinear objective + polyhedron & integer variables.

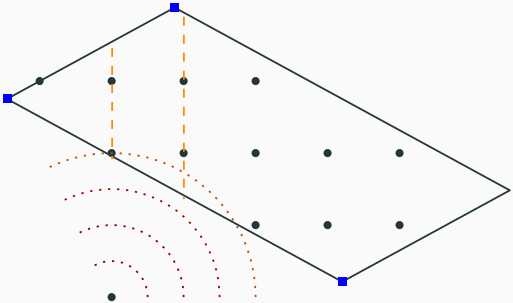
More generally:

$$\begin{aligned} \min_{x,y} \; & f(x,y) \\ \text{s.t. } & x \in \mathcal{X} \\ & x_j \in \mathbb{Z} \; \forall j \in J \\ & y \in \mathcal{Y} \end{aligned}$$

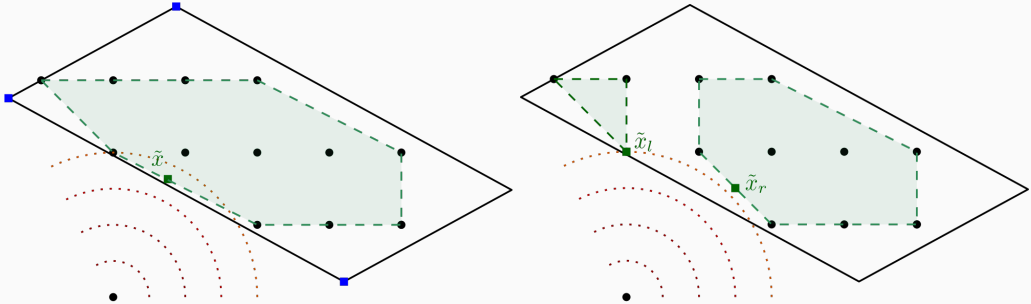
with LMO over $\mathcal{X} \cap$ bounds $\times \mathcal{Y}$

Branching Frank-Wolfe: continuous or convex hull relaxation





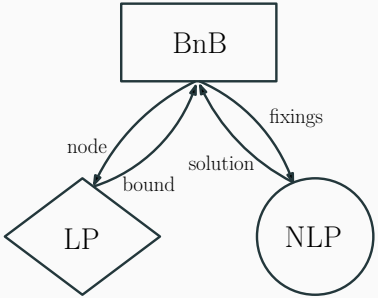
Branching Frank-Wolfe: continuous or convex hull relaxation

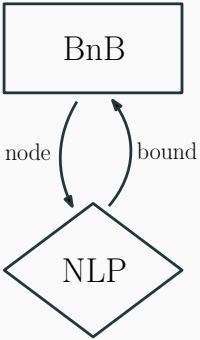


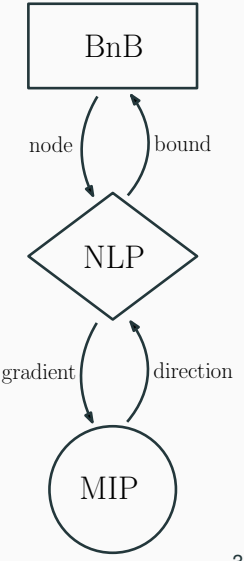
Open question:

Can we define adaptive criteria (geometry of the feasible set, conditioning of the function) to choose relaxation?

Branching Frank-Wolfe: computational model



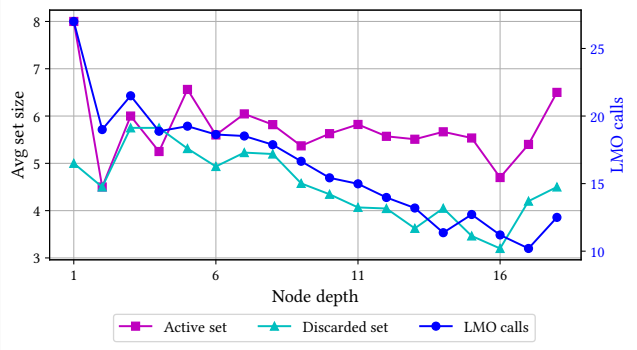




Reducing MIP oracle calls

Lazification techniques → fewer MIP calls

- Branching over the active set (all valid vertices)
- Branching over the discarded set, natural inclusion in the lazification



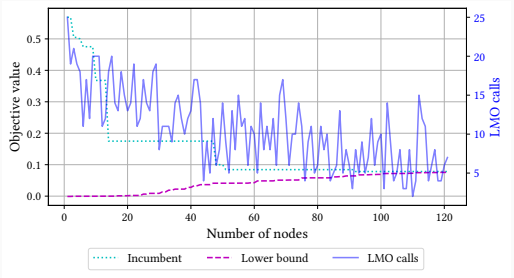
Reducing MIP cost

Reusing information across solves:

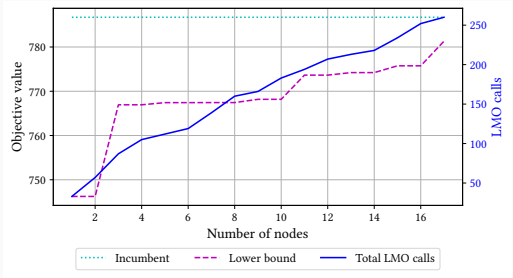
- MIP solver called with different objectives within node
- Identical polyhedron with updated bounds called across nodes.

What information should be maintained and/or transferred?
Reopens the question of MIP reoptimization [Gamrath et al., 2015].
→ which information should be (conditionally) transferred across instances?

Some computational results



Closest permutation matrix decomposition



Integer sparse regression $n = 40$

Conclusion

- Generalized self-concordance: alternative to strong convexity + smoothness requirements
- Frank-Wolfe algorithms Just Work^{TM} with such property
- FrankWolfe.jl: competitive platform for our and others' research

Branch-and-Bound:

- Novel branch-and-bound paradigm for a class of MINLPs
- No outer approximation $\rightarrow$ single polyhedron
- Leveraging an error-adaptive bounded convex subsolver
- Extra-lazification for cost reduction across the tree.

References i

Besançon, M., Carderera, A., and Pokutta, S. (2021).

FrankWolfe.jl: a high-performance and flexible toolbox for Frank-Wolfe algorithms and Conditional Gradients.

arXiv preprint arXiv:2104.06675.

Carderera, A., Besançon, M., and Pokutta, S. (2021).

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

arXiv preprint arXiv:2105.13913.

Dvurechensky, P., Safin, K., Shtern, S., and Staudigl, M. (2020).

Generalized self-concordant analysis of Frank-Wolfe algorithms.

arXiv preprint arXiv:2010.01009.

References ii

Gamrath, G., Hiller, B., and Witzig, J. (2015).

Reoptimization techniques for mip solvers.

In *International Symposium on Experimental Algorithms*, pages 181–192. Springer.

Liu, D., Cevher, V., and Tran-Dinh, Q. (2020).

A Newton Frank-Wolfe method for constrained self-concordant minimization.

arXiv preprint arXiv:2002.07003.

References iii

Sun, T. (2018).

Newton-type methods under generalized self-concordance and inexact oracles.

PhD thesis, University of North Carolina at Chapel Hill Graduate School.

Sun, T. and Tran-Dinh, Q. (2019).

Generalized self-concordant functions: a recipe for Newton-type methods.

Mathematical Programming, 178(1):145–213.

Future work

FW: classical setting

- Sparser and accelerated algorithms for FW methods with active set
  $\rightarrow$ as fast as BCG but numerically stable?
- Barrier functions: optimal decay policy?

Can we exploit a basis in barycentric coordinates to bound the active set cardinality?

Future work

Joint constraints:

$$\begin{aligned} \min_{x,y} \quad & f(x,y) \\ \text{s.t.} \quad & (x,y) \in (\mathcal{X}, \mathcal{Y}) \\ & G(x,y) \leq 0 \end{aligned}$$

G convex, differentiable or structured.

Given LMOs for $\mathcal{X}, \mathcal{Y}$, can we handle joint constraints?

Typical case: $G(x,y) = (x - y, y - x)$.

Current joint work with J. Eckstein, Z. Woodstock.

Inexact proximal approaches leveraging subproblem structures.

Error-adaptive first-order methods

Can approximate solves be cheaper / useful for the search tree or other components?

Beyond the convex MINLP case: solving diagonal QPs or LPs with PDHG (Chambolle-Pock / hybrid gradient).

Current work with Ambros Gleixner & several PhD researchers.
Primal heuristic based on approximate LP and fix-and-propagate.