Frank-Wolfe algorithms & constraint structure exploitation

Mathieu Besançon @matbesancon - besancon@zib.de

September 13th, 2023

Zuse Institute Berlin - AIS²T.

Outline

Frank-Wolfe algorithms 101

`FrankWolfe.jl`: a Julia implementation

Application to interpretable ML

Differentiable optimization layers

A pivoting framework for active set cardinality control

Frank-Wolfe for convex mixed-integer optimization

Format

Slides available (soon) at: `https://matbesancon.xyz/slides/fw_tutorial.pdf`

Code available at:

- `https://github.com/ZIB-IOL/FrankWolfe.jl` for most of the talk
- `https://github.com/ZIB-IOL/Boscia.jl` for the last part

Interactive talk: ask questions / interrupt.

Table of Contents

Class of problems

Problem class considered:

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

with:

- $\mathcal{C}$: compact convex set,
- f: differentiable on $\mathcal{C}$, often Lipschitz-smooth.

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

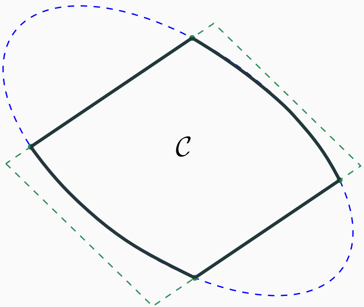
$$d \to v \in \underset{y \in \mathcal{C}}{\operatorname{argmin}} \langle y, d \rangle.$$

Frank-Wolfe vs. conic optimization as modelling paradigms

Frank-Wolfe:

$$\min_x \; f(x) \text{ s.t. } x \in \mathcal{C}$$

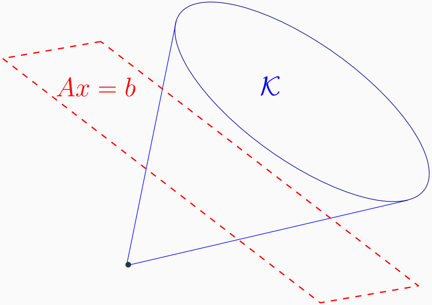
$\mathcal{C}$ compact convex (bounded).



Conic optimization:

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

$\mathcal{K}$ proper cone.



Frank-Wolfe template

Algorithm 1.1 Frank-Wolfe algorithm

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

Ensure: Iterates $x_1, \dots \in \mathcal{C}$.

for $t = 0$ **to** $\dots$ **do**

  $d_t \leftarrow \text{FirstOrderEstimate}(f, x_t)$

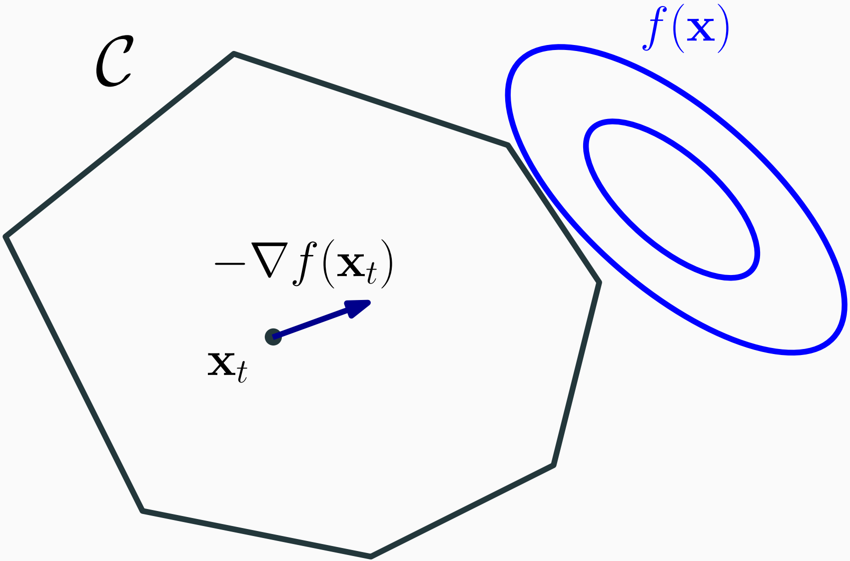
  $v_t \leftarrow \text{argmin}_{v \in \mathcal{C}} \langle d_t, v \rangle$

  $\gamma_t \leftarrow \text{StepSizeStrategy}(x_t, t, d_t, v_t)$

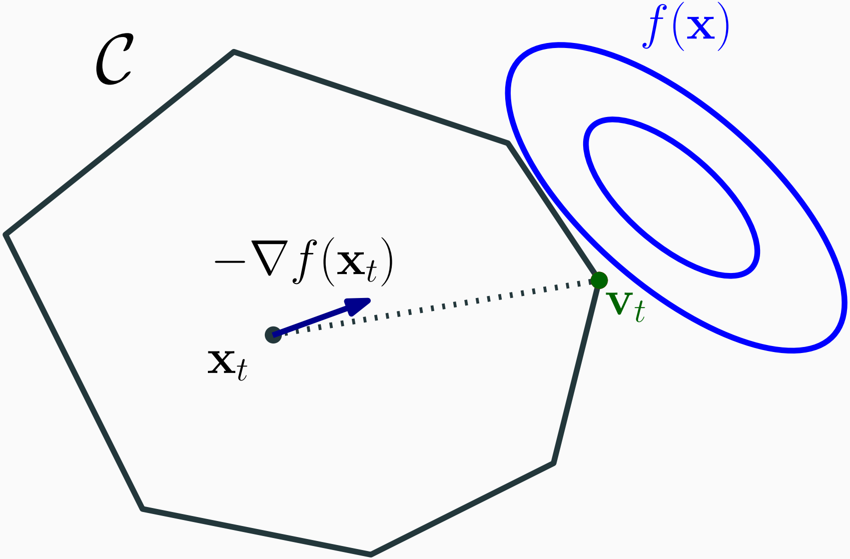
  $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$

end for

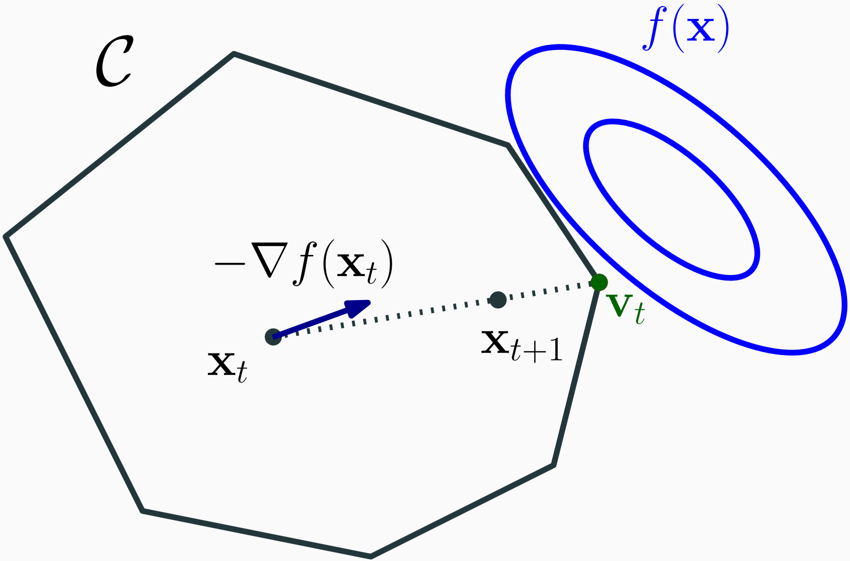
FW, visualized



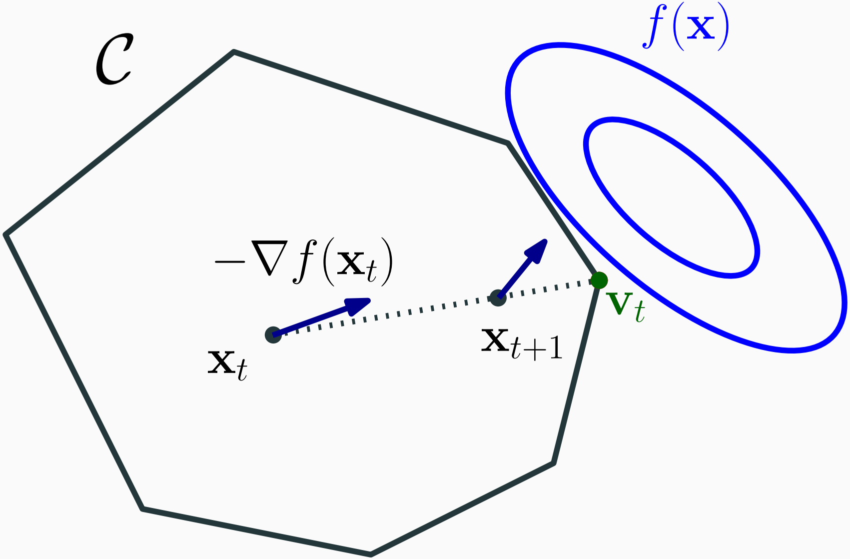
FW, visualized



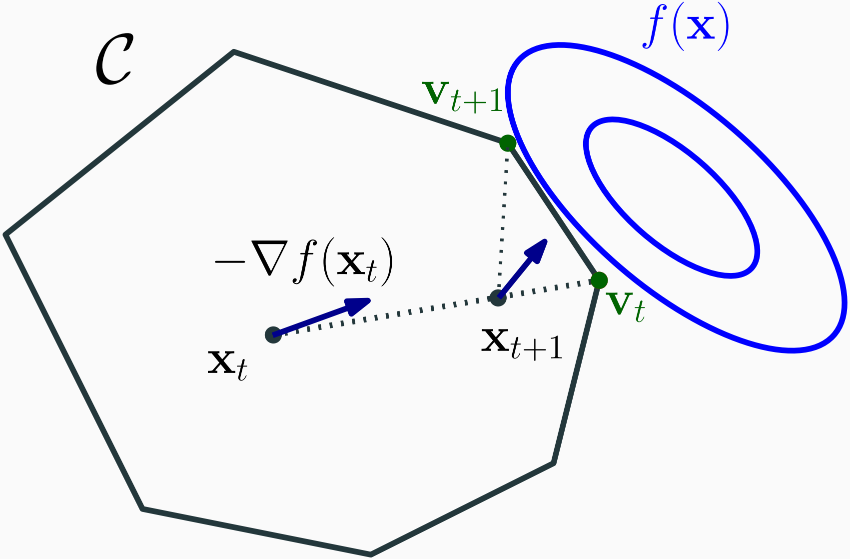
FW, visualized



FW, visualized



FW, visualized



Upper and lower bounds

All x_t are feasible by convexity. Furthermore:

$$\begin{aligned} f(x) - f(x^*) &\leq \langle \nabla f(x), x - x^* \rangle && \text{(convexity)} \\ &\leq \langle \nabla f(x), x - v \rangle + \langle \nabla f(x), v \rangle - \langle \nabla f(x), x^* \rangle \\ &\leq \langle \nabla f(x), x - v \rangle := \text{FW gap} && (\text{since } v \in \operatorname{argmin} \langle \nabla f(x), \cdot \rangle) \end{aligned}$$

Lower bound on optimality as a by-product at each iteration.

Convergence proof

Let f be L-smooth on $\mathcal{X}$ of diameter D, then FW converges with:

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

when using the agnostic step-size rule: $\gamma_t = \frac{2}{t+3}$.
Only proof in the talk.
Goal: providing **intuition** on the convergence.

Proof

$$
\begin{aligned}
f(x_{t+1}) - f(x_t) \leq & \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2} \left\| x_{t+1} - x_t \right\|^2 && \text{(L-smoothness)} && (1) \\
= & \gamma_t \langle \nabla f(x_t), v_t - x_t \rangle + \frac{L\gamma_t^2}{2} \| v_t - x_t \|^2 && && (2) \\
\leq & \gamma_t \langle \nabla f(x_t), x^* - x_t \rangle + \frac{L\gamma_t^2 D^2}{2} && && (3) \\
\leq & \gamma_t (f(x_t) - f(x^*)) + \frac{L\gamma_t^2 D^2}{2}. && && (4)
\end{aligned}
$$

- (1) → (2) using the FW step definition: $x_{t+1} - x_t = \gamma_t (v_t - x_t)$
- (2) → (3): $\langle \nabla f(x_t), v_t \rangle \leq \langle \nabla f(x_t), x^* \rangle$ by definition, and diameter bound.
- (3) → (4): $\langle \nabla f(x_t), x^* - x_t \rangle \leq f(x_t) - f(x^*)$ by convexity.

$$
\boxed{h_{t+1} \leq (1 - \gamma_t) h_t + \gamma_t^2 \frac{LD^2}{2}}
$$

Proof (2)

$$h_{t+1} \leq (1 - \gamma_t) h_t + \gamma_t^2 \frac{LD^2}{2}$$

For $t = 1$, and setting $\gamma_1 = 1$:

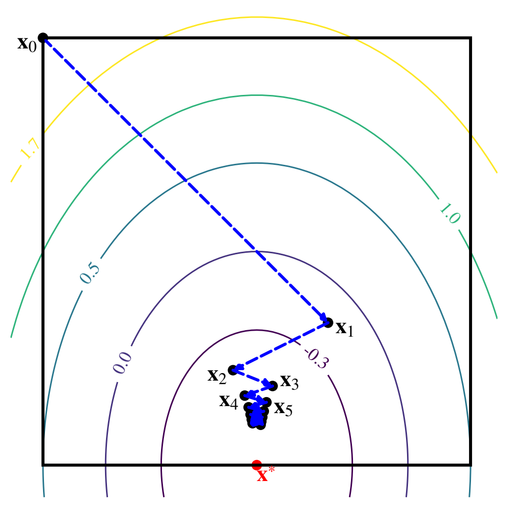
$$h_2 \leq \frac{LD^2}{2} = \frac{2LD^2}{4} = \frac{2LD^2}{t+3}.$$

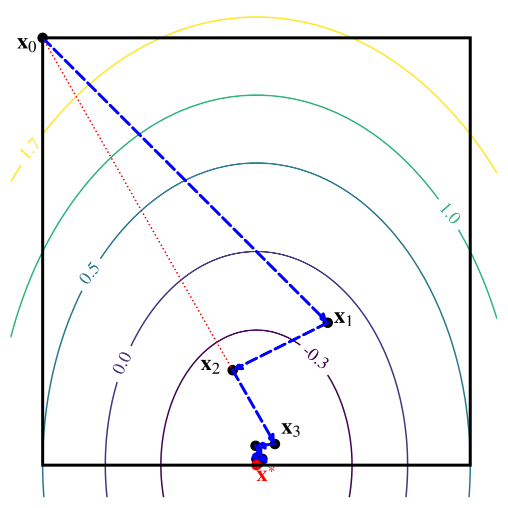
Pluging the step size rule $\gamma_t = \frac{2}{t+3}$ and the bound for h_t:

$$\begin{aligned} h_{t+1} &\leq (1 - \frac{2}{t+3}) \frac{2LD^2}{t+3} + \frac{4}{(t+3)^2} \frac{LD^2}{2} \\ &= \frac{(t+2)2LD^2}{(t+3)^2} \\ &\leq \frac{2LD^2}{t+4} = \frac{2LD^2}{(t+1)+3}. \quad \blacksquare \end{aligned}$$

Active set representations

Away FW, Blended (Pairwise) Conditional Gradients, ...





Linear Minimization Oracle

How hard is optimizing a linear function over C?

Problem	Feasible set C	Oracle
Generic (P)	Polytope	LP solver: p/d simplex
Sparse regression, discrete measures	$\ell_{1,2,\infty}$ norm balls, simplex	Closed form
Optimal Transport	Transportation polytope	Network simplex
Matching, Markov chains	Birkhoff polytope	Hungarian algorithm
Image segmentation, PCA	Nuclear norm	Greatest singular vector pair
Combinatorics, covariance estimation	Spectraplex	Greatest eigenvector
Discrete problems	$\mathrm{conv}(\{x_1 \ldots x_k\})$	Enumeration

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.

Main variants

Variant	Convergence Progress/iter.	 Time/iter.	Sparsity	Numerical Stability	Active Set?	Lazy?
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: $v_t \leftarrow \text{argmin}_{v \in C} \langle d_t, v \rangle$

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

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

```
using FrankWolfe as FW
using LinearAlgebra
y = randn(1000)

f(x) = 1/2 * norm(x - y)^2
function grad!(storage, x)
    # entrywise operation
    # without temporary allocation
    @. storage = x - y
end
```

```
# oracle of C
lmo = FW.ProbabilitySimplexOracle(1.0)
# starting point
x0 = FW.compute_extreme_point(
    lmo, ones(n)
)
xf, _ = FW.frank_wolfe(
    f, grad!, lmo, x0,
    max_iterations=1000, epsilon=1e-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

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

Work spearheaded by Jan Macdonald, joint with Sebastian Pokutta,

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

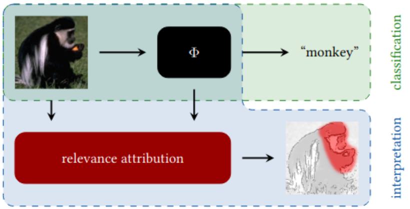
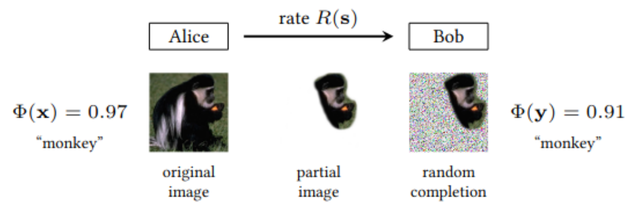


Figure 1: Relevance attribution methods aim at rendering black-box classifiers more interpretable by providing heatmaps of the input features that contribute most to an individual prediction.

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

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

$$\Downarrow$$

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

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 retain?
What about aggregating over multiple rates?

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

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

B_n 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

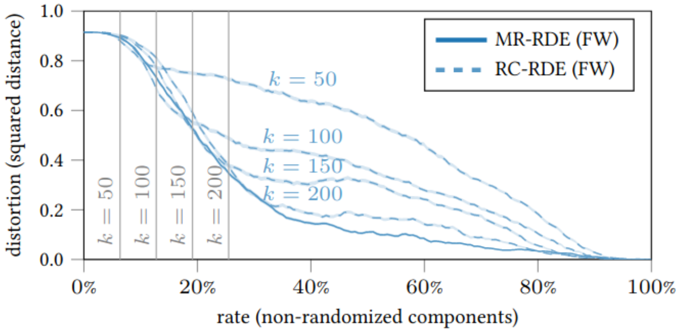


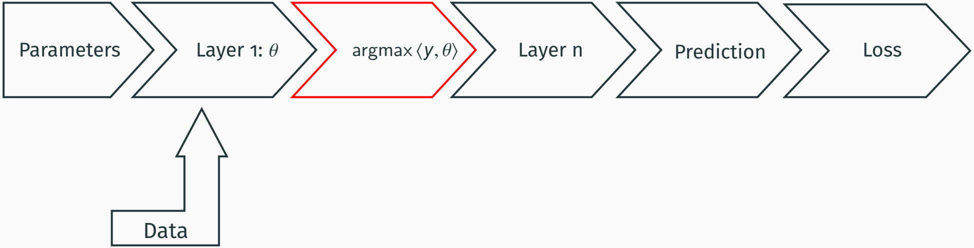
Figure 5: Relevance ordering test results for MNIST and (RC-RDE) at various rates. Vertical lines show the rates k at which the mappings were optimized. The combined (MR-RDE) solution approximates a lower envelope of the individual curves. An average result over 50 images from the test set (5 images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

Table of Contents

Consider a parameterized optimization problem:

$$\max_{y \in C} \langle y, \theta \rangle$$

used as a layer in a deep learning architecture



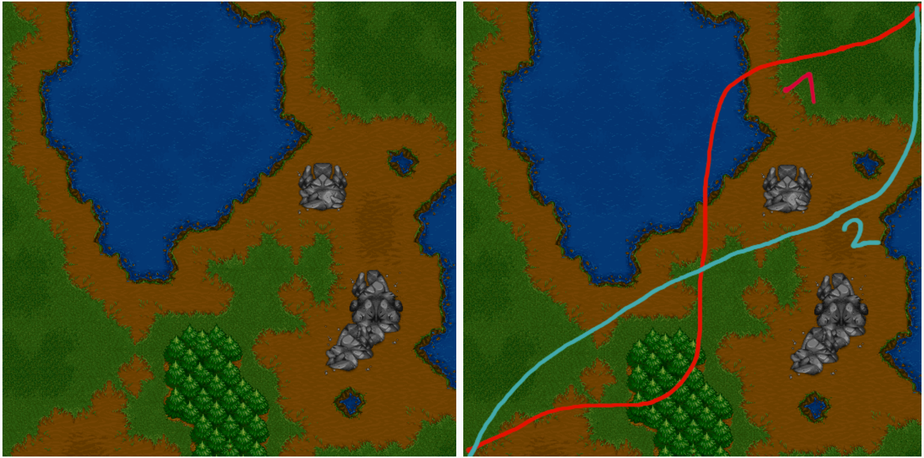
∂O allows **backpropagating** through the problem.

Modelling rich structures & constraints exactly with fewer parameters.

See B. Amos, Ph.D. thesis, 2019.

Example: Warcraft maps

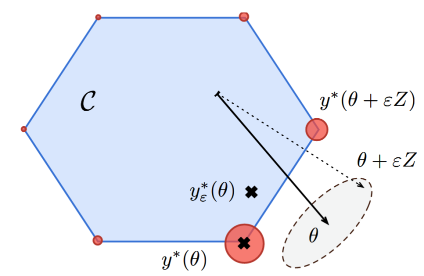
Learning paths on a map from images with (unknown) element costs.



Vlastelica et. al , Differentiation of blackbox combinatorial solvers, 2020.

Non-differentiable problems?

Differentiating through an argmax layer?
$\Rightarrow$ zero Jacobian almost everywhere, with discontinuities.



Solution of Berthet et al. 2020:
- Perturbation via sampling of θ.
- Corresponds to a distribution of the output y.
- Unbiased gradient estimate $\nabla_\theta y$ via MC sampling.
- Noise on $\theta \Leftrightarrow$ regularization of original problem.

Source: Berthet et al, Learning with Differentiable Perturbed Optimizers, 2020.

Alternative approach: explicit regularization

Approach by Dalle et al, 2022:
Replace sampling with an explicit regularizer:

$$\underset{y \in C}{\operatorname{argmax}} \langle y, \theta \rangle - \Omega(y)$$

with Ω convex, smooth.

Frank-Wolfe produces a convex combination of vertices
$\rightarrow$ Interpretable as a probability distribution.
Differentiation:
Interpret optimum as fixed-point of projected gradient onto the simplex of vertices.

Learning with Combinatorial Optimization Layers: a Probabilistic Approach

Table of Contents

A Pivoting Framework for Cardinality Control

Joint work with Elias Wirth, Sebastian Pokutta (TU Berlin & ZIB).

Away / Pairwise / Blended / Blended Pairwise Frank-Wolfe:
Collect vertices in an **active set** $\{v_1, v_2 \ldots v_k\}$ with weights $\lambda \in \Delta$.

TL;DR: sparser iterates i.e. smaller active set with scalable techniques.

Method: Prune the active set using linear optimization.

Active Set Cardinality Control

Carathéodory Theorem: $\mathcal{C} \subset \mathbb{R}^n$ compact convex.
$x \in \mathcal{C}$ can be represented as a convex combination of $n+1$ extreme points of $\mathcal{C}$.

Not exploited in FW algorithms, only bounds: # iterations, # vertices of $\mathcal{C}$!

Goal: Carathéodory-ensuring algorithm.

Formulation as optimizing over an extended weight polytope:

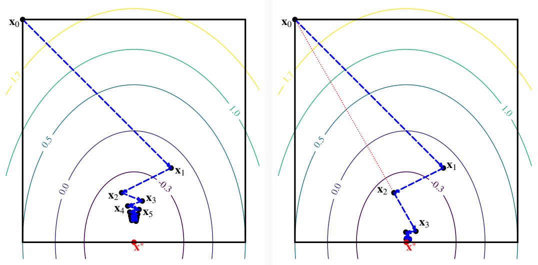
$$\begin{aligned} &V\lambda = x_t \\ &e^\top \lambda = 1 \\ &\lambda \geq 0. \end{aligned}$$

Method: linear system of size $(n+2) \times (n+2)$ finding the weights $\lambda \in \Delta$.

Remember the convergence bound?

$$h_t = f(x_t) - f(x^*) \leq \frac{2L{\color{red}D}^2}{t+3}$$

If D dependent on the dimension, arbitrarily bad constant!



Luckily:

Away and Blended Pairwise FW $\rightarrow$ Active Set Identification Property in finite time.

I. Bomze, F. Rinaldi, D. Zeffiro, 2020 for AFW, E. Wirth, M. Besançon, S. Pokutta, 2023 for BPFW.

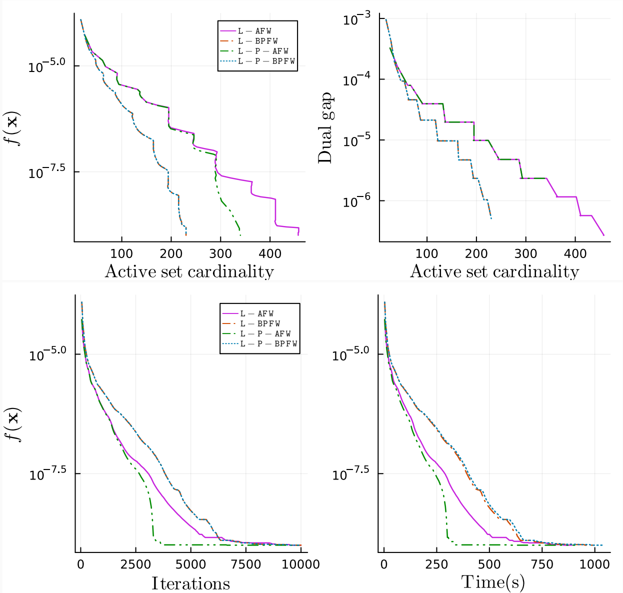
Active Set Identification

$$h_t = f(x_t) - f(x^*) \leq \frac{2L{D_F}^2}{t+3}$$

with D_F the diameter of the optimal face.

Theorem:
Assume minimizers in relative interior of a face $\mathcal{F}$ of the polytope. After a finite number of iterations, AFW and BPFW contain only vertices from $\mathcal{F}$.

Results: quadratic objective on Birkhoff polytope



Results: logistic regression on K-sparse polytope

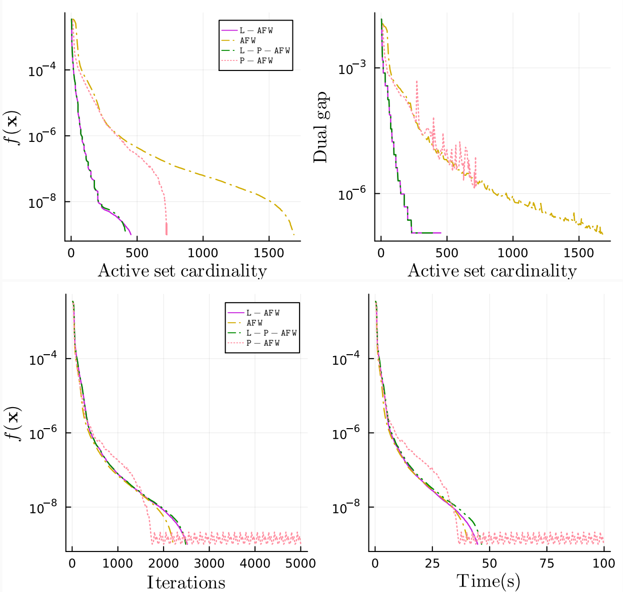


Table of Contents

Preprint: `arxiv.org/abs/2208.11010`,
Convex mixed-integer optimization with FW methods
Package available at `github.com/ZIB-IOL/Boscia.jl`

Applications in engineering, sparse prediction models, statistics, relaxation of combinatorial problems.

Problem setting

Nonlinear convex objective + polyhedron & integer variables
(& combinatorial constraints).

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

Problem setting

Nonlinear convex objective + polyhedron & integer variables (& combinatorial constraints).
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 $\mathcal{X} \cap \text{bounds} \times \mathcal{Y}$

Considered problems

(Group) cardinality-constrained {linear, logistic, Poisson} regression:

$$\min_{\beta} 1/2\|X\beta - y\|^2, \text{ s.t. } \|\beta_{i,i\in g}\|_0 \leq k_g \; \forall g \in \mathcal{G}$$

Integer and cardinality-constrained portfolio optimization & regression.

Considered problems (2)

Tail-cardinality constraints in {linear, logistic, Poisson} regression:

$$\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) + \lambda\|\max\{|\mathbf{x}_i| - \tau_i, 0\}_{i\in[n]}\|_0 + \mu\|\mathbf{x}\|_2^2$$

which can be reformulated using indicator constraints as:

$$\begin{aligned}
\min_{\mathbf{x},\mathbf{z},\mathbf{s}} \quad & f(\mathbf{x}) - \lambda\sum_i \mathbf{z}_i + \mu\|\mathbf{x}\|_2^2 \\
\text{s.t.} \quad & \mathbf{z}_i = 1 \Rightarrow \mathbf{s}_i \leq 0 \\
& \mathbf{s}_i \geq \mathbf{x}_i - \tau_i \\
& \mathbf{s}_i \geq -\mathbf{x}_i - \tau_i \\
& \mathbf{x} \in \mathcal{X}, \mathbf{z} \in \{0,1\}^n, \mathbf{s} \in \mathcal{X} \cap \mathbb{R}_+^n.
\end{aligned} \tag{TCMP}$$

Considered problems (3)

Closest combination of permutation matrices / cardinality-constrained projection on Birkhoff:

$$\min_{X \in P_n, \theta \in \Delta_k} \|\sum_{i \in [k]} \theta_i X_i - \hat{X}\|_F^2,$$

Sparse + low-rank decomposition: robust PCA / foreground-background segmentation:

$$\min_{X,Y} \|X + Y - D\|_F^2, \text{ s.t. } \|X\|_* \leq \tau, \|Y\|_0 \leq k$$

Tigher than convex relaxation, convexification of rank constraint only [Bertsimas et al., 2021].

Design of experiments

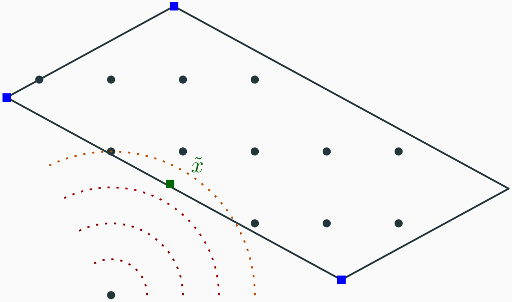
The D-Optimal Problem

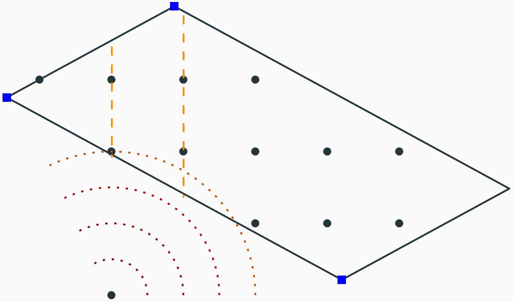
$$
\begin{aligned}
\max_x \ & \operatorname{logdet}\left(A^\top \operatorname{diag}(x) A\right) \\
\text{s.t.} \ & \sum_{i=1}^{m} x_i = N \\
& l \le x \le u \\
& x \in \mathbb{Z}_+^m.
\end{aligned}
$$

The A-Optimal Problem

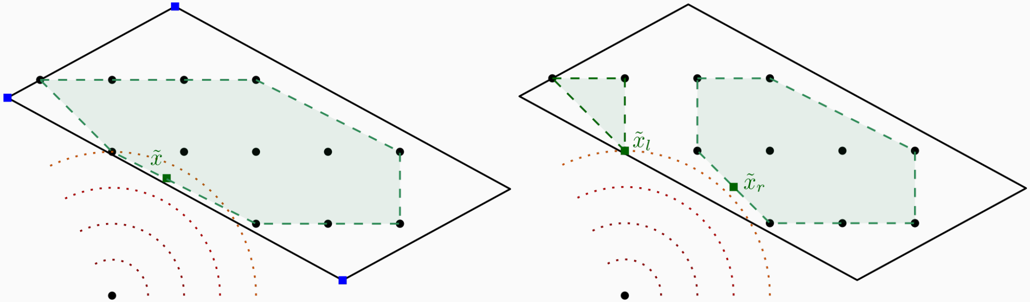
$$
\begin{aligned}
\max_x \ & -\operatorname{Tr}\left(\left(A^\top \operatorname{diag}(x) A\right)^{-1}\right) \\
\text{s.t.} \ & \sum_{i=1}^{m} x_i = N \\
& l \le x \le u \\
& x \in \mathbb{Z}_+^m.
\end{aligned}
$$

Branching Frank-Wolfe: continuous or convex hull relaxation





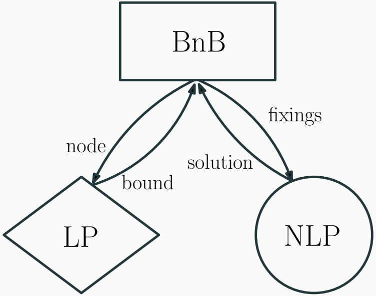
Branching Frank-Wolfe: continuous or convex hull relaxation

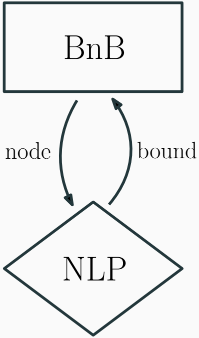


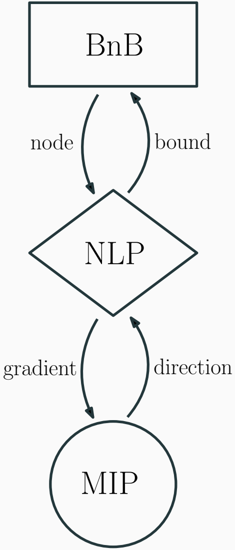
Open question:

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

Branching Frank-Wolfe: computational model



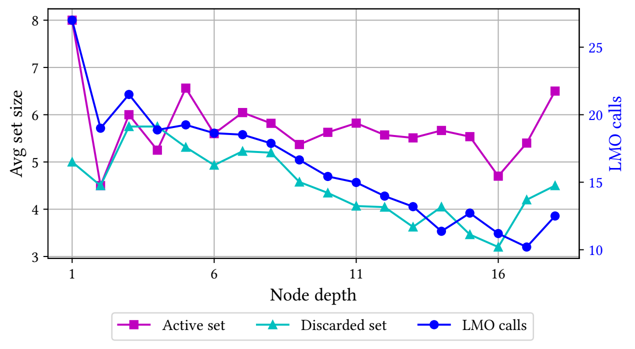




Reducing oracle calls

Lazification techniques using Blended-Pairwise [Tsuji et al., 2021] → fewer MIP calls.

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



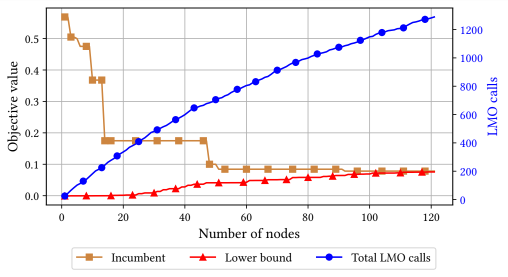
Reducing MIP cost

Reusing information across solves:

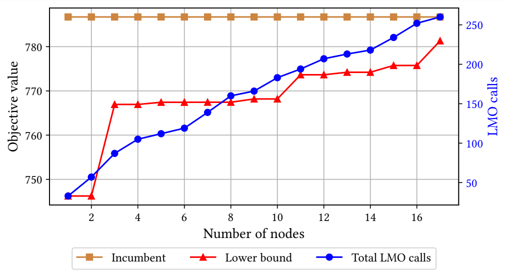
- MIP solver called with different objectives within node
- Identical polyhedron with updated bounds called across nodes.

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

Computational results

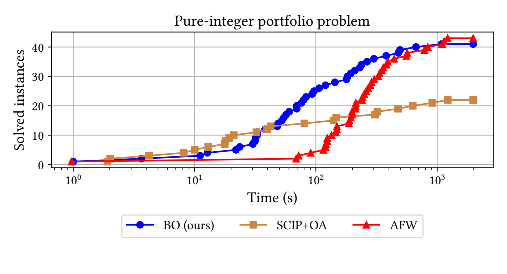


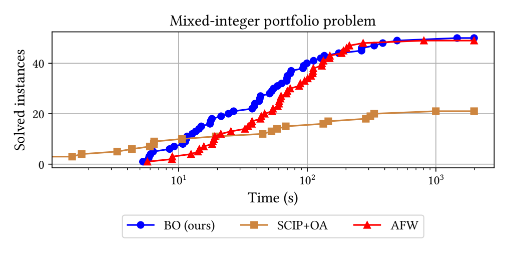
Closest permutation matrix decomposition



Integer sparse regression $n = 40$

Comparison with outer approximation and other convex solver

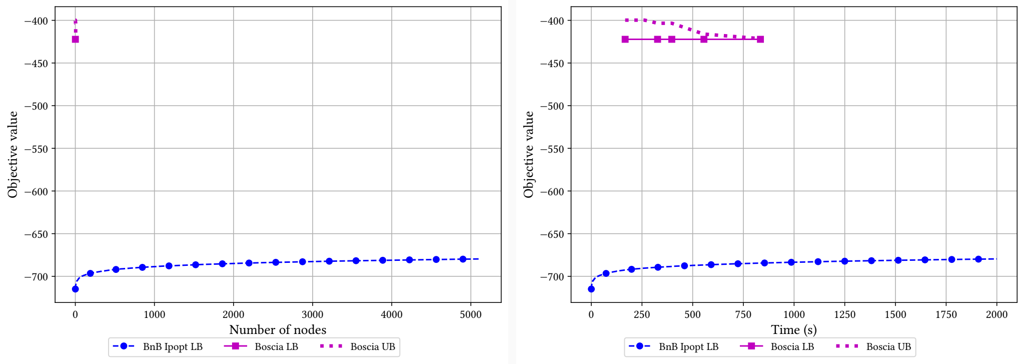




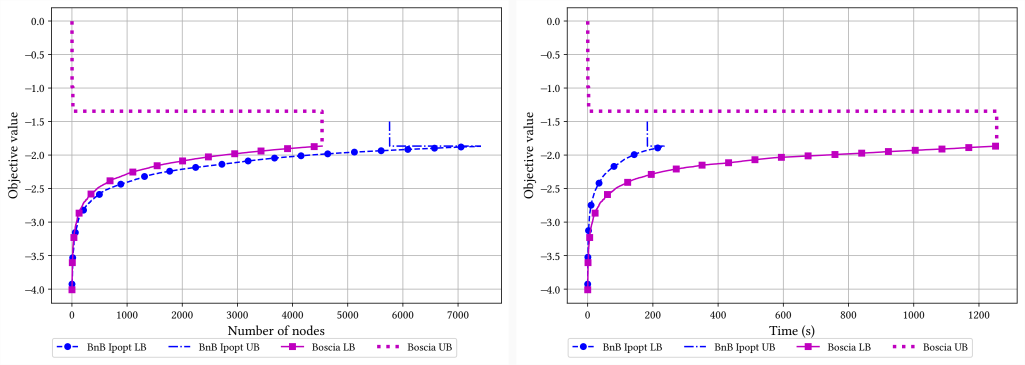
Outer Approximation with SCIP + gradient cuts (OA) performs well for highly constrained and integer problems.
Away Frank-Wolfe (AFW) operates on denser iterates and larger active sets.

Improvement over pure Branch-and-Bound - pg5: the good

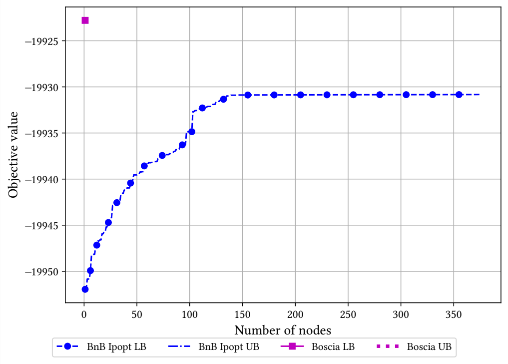
Comparison with pure BNB & Ipopt on MIPLib instances and quadratic objectives:

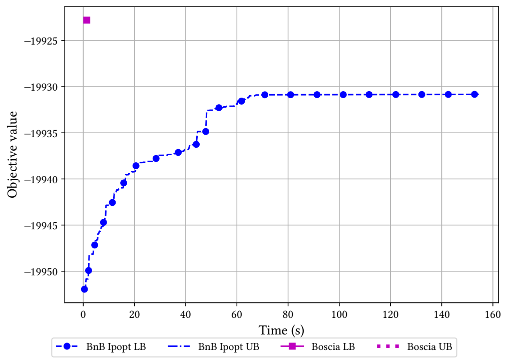


Improvement over Branch-and-Bound (2) - neos5: the bad



Improvement over Branch-and-Bound (3) - 22433: the lucky





Conclusion

More to Machine Learning training than gradient descent!
Frank-Wolfe methods allow structured constraints, sparsity-induction.

Boscia:

- 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.

Example

```
y = rand(Bool, n) * 0.6 .+ 0.3
o = SCIP.Optimizer()
x = MOI.add_variables(o, n)
MOI.add_constraint.(o, x, MOI.ZeroOne())
lmo = FrankWolfe.MathOptLMO(o)

f(x) = 0.5 * norm(x-y)^2
function grad!(storage, x)
    @. storage = x - y
end

x, _, result = Boscia.solve(f, grad!, lmo)
```

Bonus: handling nonconvexities in MINLP

$$\begin{aligned}\min_{x,y}\ & f(x) + g(y) \\ \text{s.t.}\ & (x,y) \in \mathcal{R} \\ & x \in \mathcal{X} \subseteq \{0,1\}\end{aligned}$$

f nonconvex, g convex, add convexifier:

$$\begin{aligned}\min_{x,y}\ & f(x) + g(y) + |\lambda_{min}| \sum_j (x_j^2 - x_j) \\ \text{s.t.}\ & (x,y) \in \mathcal{R} \\ & x \in \mathcal{X} \subseteq \{0,1\}\end{aligned}$$

Applications in QUBO, Max-cut, quadratic assignment...

References i

Bertsimas, D., Cory-Wright, R., and Johnson, N. A. (2021).

Sparse plus low rank matrix decomposition: A discrete optimization approach.

arXiv preprint arXiv:2109.12701.

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.

References ii

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.

Gamrath, G., Hiller, B., and Witzig, J. (2015).
Reoptimization techniques for mip solvers.
In *International Symposium on Experimental Algorithms*, pages 181–192. Springer.

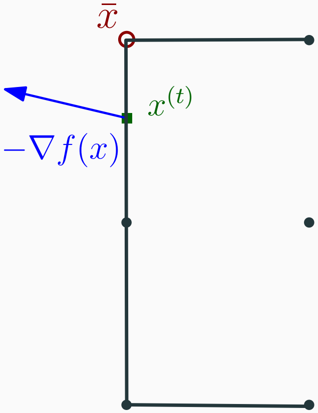
Tsuji, K., Tanaka, K., and Pokutta, S. (2021).
Sparser kernel herding with pairwise conditional gradients without swap steps.
arXiv preprint arXiv:2110.12650.

Dual tightening

Given bounds $[\mathbf{l}, \mathbf{u}]$, relaxed solution $\mathbf{x}^{(t)}$ and $j \in J$ such that $\mathbf{x}_j^{(t)} = \mathbf{l}_j$ and $\nabla f(\mathbf{x}^{(t)})_j \geq 0$.
Then, if $\exists M \in \{1, \ldots, \mathbf{u}_j - \mathbf{l}_j\}$, such that:

$$M \nabla f(\mathbf{x}^{(t)})_j > \mathrm{UB} - f(\mathbf{x}^{(t)}) + g(\mathbf{x}^{(t)})$$

$\mathbf{u}_j$ can be tightened to: $\mathbf{x}_j^{(t)} \leq \mathbf{l}_j + M - 1$.
$M = 1 \Rightarrow$ fixing $\mathbf{u}_j = \mathbf{l}_j$.

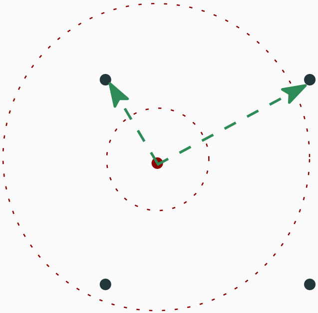


Proof idea: convexity and property of the LMO yield minimum condition for an optimum moving from $\mathbf{l}_j$ to $\mathbf{l}_j + M - 1$.

Strong convexity dual bounds

Current relaxation $\hat{\mathbf{x}}$, fractional variables $\hat{J} \subseteq J$, j variable to branch on, $\mathbf{x}_l^*$ solution after branching. Better lower bound for $f(\mathbf{x}_l^*)$?

$$
\begin{aligned}
f(\mathbf{x}_l^*) &\geq f(\hat{\mathbf{x}}) + \frac{\mu}{2}\left\|\mathbf{x}_l^* - \hat{\mathbf{x}}\right\|_2^2 + \left\langle \nabla f(\hat{\mathbf{x}}), \mathbf{x}_l^* - \hat{\mathbf{x}} \right\rangle \\
&\geq f(\hat{\mathbf{x}}) + \frac{\mu}{2}(\hat{\mathbf{x}}_j - \lfloor \hat{\mathbf{x}}_j \rfloor)^2 - \left\langle \nabla f(\hat{\mathbf{x}}), \hat{\mathbf{x}} - \mathbf{x}_l^* \right\rangle \\
&\geq f(\hat{\mathbf{x}}) + \frac{\mu}{2}(\hat{\mathbf{x}}_j - \lfloor \hat{\mathbf{x}}_j \rfloor)^2 - \max_{\mathbf{v} \in \mathcal{X}} \langle \nabla f(\hat{\mathbf{x}}), \hat{\mathbf{x}} - \mathbf{v} \rangle \\
&= f(\hat{\mathbf{x}}) + \frac{\mu}{2}(\hat{\mathbf{x}}_j - \lfloor \hat{\mathbf{x}}_j \rfloor)^2 - g(\hat{\mathbf{x}}).
\end{aligned}
$$



Usage:

Branching rule, pruning children, pruning node altogether.