

Bilevel Optimization of Group-wise Penalties: Application to Structure Discovery

Jordan Frecon

Equipe App, LITIS, INSA Rouen



slides & more available at jordan-frecon.com

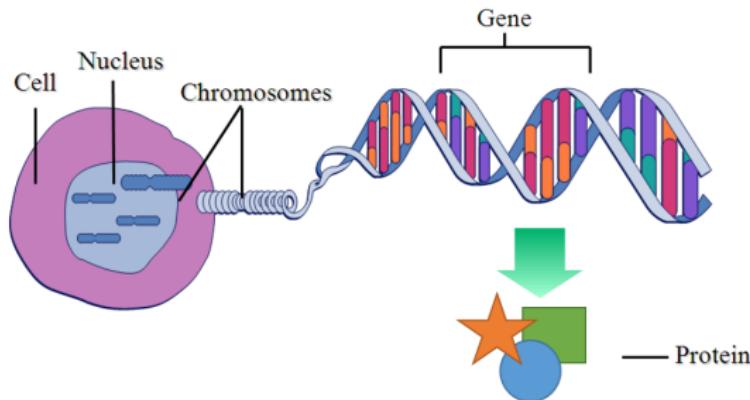
Example 1: Structured linear regression

Setting presented in
[Frecon et al. "Bilevel learning of the group Lasso structure". NeurIPS (2018)]

Motivation: genes expression analysis

Goal : Predict the function of proteins from regulatory patterns

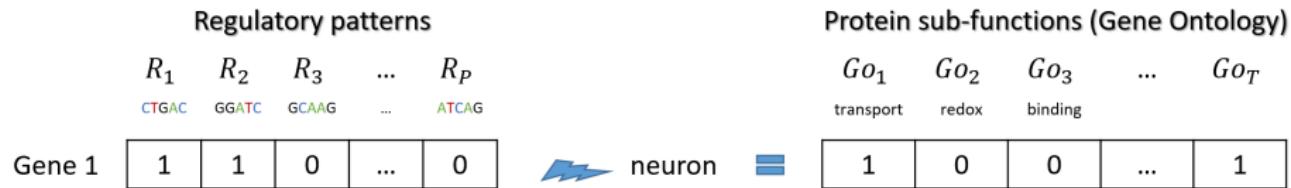
Collaboration with Giorgio Valentini (Universita degli Studi di Milano)



Gene = long sequence with regulatory patterns (dictate gene expression)

Proteins perform various functions (transport, redox, binding ...)

Motivation: genes expression analysis



Sequences R_1 and R_2 are present in Gene 1

Gene 1 produces neurons

Neurons perform transport of electrons

Motivation: genes expression analysis

Regulatory patterns						Protein sub-functions (Gene Ontology)					
	R_1	R_2	R_3	...	R_P		Go_1	Go_2	Go_3	...	Go_T
	CTGAC	GGATC	GCAAG	...	ATCAG						
Gene 1	1	1	0	...	0		transport	redox	binding		
Gene 2	1	1	1	...	0		neuron	=		1	
							hormone	=		0	

Sequences R_1 , R_2 and R_3 are present in Gene 2

Gene 2 produces hormones

Hormones perform transport of particles and reduction–oxidation

Motivation: genes expression analysis

Regulatory patterns

	R_1	R_2	R_3	...	R_P
	CTGAC	GGATC	GCAAG	...	ATCAG

Gene 1	1	1	0	...	0
Gene 2	1	1	1	...	0
Gene 3	0	0	0	...	1

Protein sub-functions (Gene Ontology)

	Go_1	Go_2	Go_3	...	Go_T
transport					

- neuron
- hormone
- antibody

1	0	0	...	1
1	1	0	...	0
0	0	1	...	0

Sequence R_P is present in Gene 3

Gene 3 produces antibodies

Antibodies perform binding of particles

Motivation: genes expression analysis

Regulatory patterns

$R_1 \quad R_2 \quad R_3 \quad \dots \quad R_P$

CTGAC GGATC GCAAG ... ATCAG

	R_1	R_2	R_3	...	R_P
Gene 1	1	1	0	...	0
Gene 2	1	1	1	...	0
Gene 3	0	0	0	...	1
⋮	⋮	⋮	⋮	⋮	⋮
Gene N	1	1	0	...	1

$$X \in \{0,1\}^{N \times P}$$

Protein sub-functions (Gene Ontology)

$Go_1 \quad Go_2 \quad Go_3 \quad \dots \quad Go_T$

transport redox binding

- neuron
- hormone
- antibody

	Go_1	Go_2	Go_3	...	Go_T
Gene 1	1	0	0	...	1
Gene 2	1	1	0	...	0
Gene 3	0	0	1	...	0
⋮	⋮	⋮	⋮	⋮	⋮
Gene N	0	0	1	...	1

$$y = [y_1 \cdots y_T] \in \{0,1\}^{N \times T}$$

Motivation: genes expression analysis

Regulatory patterns

$R_1 \quad R_2 \quad R_3 \quad \dots \quad R_P$

CTGAC GGATC GCAAG ... ATCAG

	R_1	R_2	R_3	...	R_P
Gene 1	1	1	0	...	0
Gene 2	1	1	1	...	0
Gene 3	0	0	0	...	1
⋮	⋮	⋮	⋮	⋮	⋮
Gene N	1	1	0	...	1

$$X \in \{0,1\}^{N \times P}$$

Protein sub-functions (Gene Ontology)

$Go_1 \quad Go_2 \quad Go_3 \quad \dots \quad Go_T$

transport redox binding

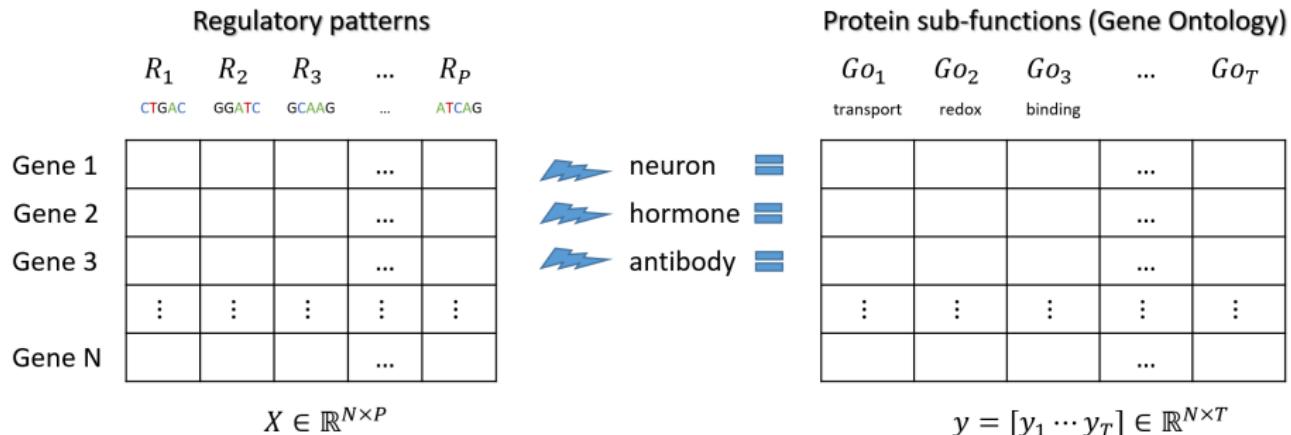
- neuron
- hormone
- antibody

	Go_1	Go_2	Go_3	...	Go_T
Gene 1	1	0	0	...	1
Gene 2	1	1	0	...	0
Gene 3	0	0	1	...	0
⋮	⋮	⋮	⋮	⋮	⋮
Gene N	0	0	1	...	1

$$y = [y_1 \cdots y_T] \in \{0,1\}^{N \times T}$$

Goal 1: Predict each y_t from X

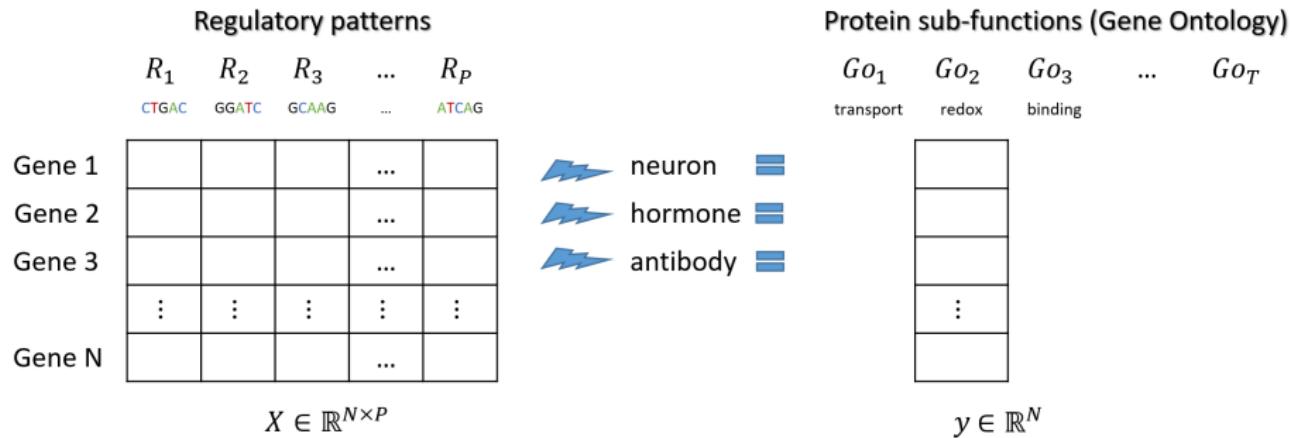
Motivation: genes expression analysis



Goal 1: Predict each y_t from X

→ to generalize: X and y are not only made of 0's and 1's

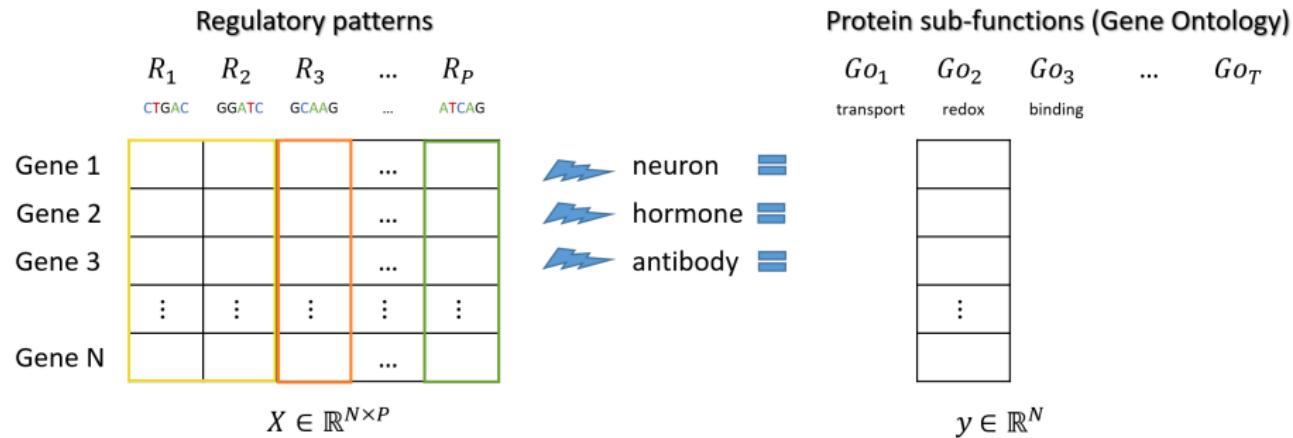
Motivation: genes expression analysis



Goal 1: Predict y from X

- to generalize: X and y are not only made of 0's and 1's
- to simplify: we first assume that $T = 1$ and omit the index t

Motivation: genes expression analysis



Goal 1: Predict y from X

- to generalize: X and y are not only made of 0's and 1's
- to simplify: we first assume that $T = 1$ and omit the index t

Goal 2: Discover if there exist some groups in X

ex: R_1 and R_2 are both equally relevant to predict y

Assumptions

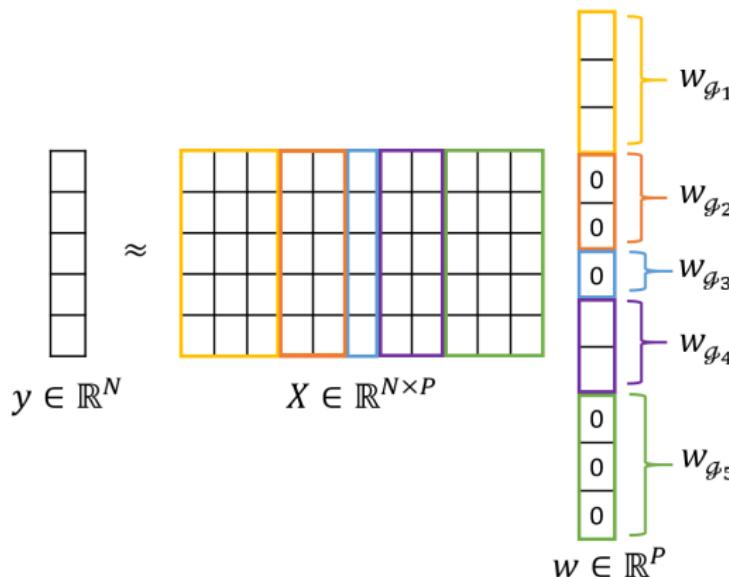
Model the observations

⇒ linear model + Gaussian distribution: there exists w such that $y \sim \mathcal{N}(Xw, \sigma)$

Model the group structure

few groups of features in X are relevant to predict y

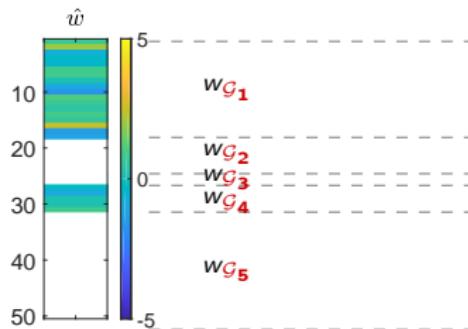
⇒ group sparsity: some groups of variables in w are zero while others are non-zero



Optimization problem

Goal 1: Predict y from X Group Lasso [Yuan and Lin (2006)]

$$\hat{w}(\mathcal{G}_1, \dots, \mathcal{G}_L) = \underset{w \in \mathbb{R}^P}{\operatorname{argmin}} \underbrace{\frac{1}{2} \|y - Xw\|^2}_{\propto -\log p(y|Xw)} + \lambda \sum_{l=1}^L \|\mathbf{w}_{\mathcal{G}_l}\|_2 \quad \text{enforces structure}$$



L partitions $\mathcal{G}_1, \dots, \mathcal{G}_L$ of P features

$$\mathcal{G}_l \subseteq \{1, \dots, P\}$$

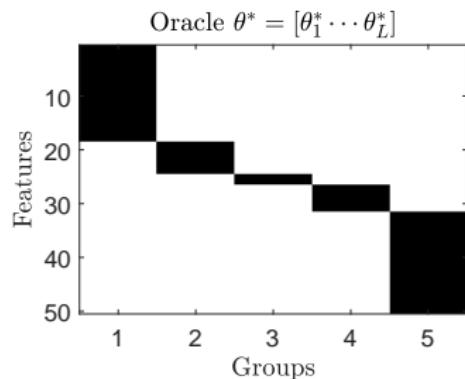
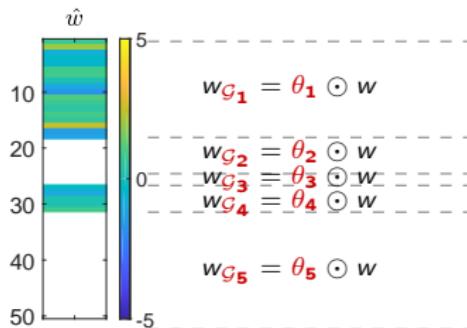
$$\mathcal{G}_l \cap \mathcal{G}_{l'} = \emptyset \text{ if } l \neq l'$$

$$\cup_{l=1}^L \mathcal{G}_l = \{1, \dots, P\}$$

Optimization problem

Goal 1: Predict y from X Group Lasso [Yuan and Lin (2006)]

$$\hat{w}(\mathcal{G}_1, \dots, \mathcal{G}_L) = \underset{w \in \mathbb{R}^P}{\operatorname{argmin}} \underbrace{\frac{1}{2} \|y - Xw\|^2}_{\propto -\log p(y|Xw)} + \lambda \sum_{l=1}^L \|\mathbf{w}_{\mathcal{G}_l}\|_2 \quad \text{enforces structure}$$



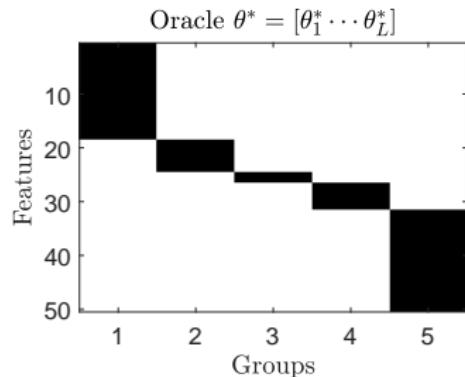
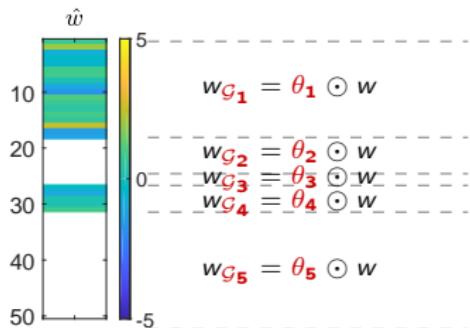
Mask $\theta_l = \{0, 1\}^P$ of the l -th group

element-wise multiplication $\theta_l \odot w = [\theta_{l,1} w_1, \theta_{l,2} w_2, \dots, \theta_{l,P} w_P]^\top$

Optimization problem

Goal 1: Predict y from X Group Lasso [Yuan and Lin (2006)]

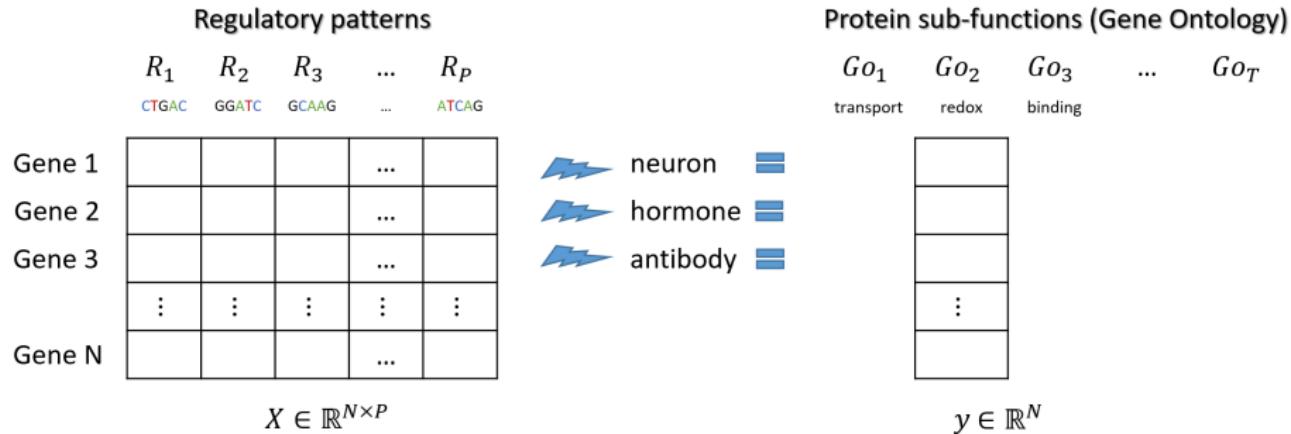
$$\hat{w}(\mathcal{G}_1, \dots, \mathcal{G}_L) = \underset{w \in \mathbb{R}^P}{\operatorname{argmin}} \underbrace{\frac{1}{2} \|y - Xw\|^2}_{\propto -\log p(y|Xw)} + \lambda \sum_{l=1}^L \|\mathbf{w}_{\mathcal{G}_l}\|_2 \quad \text{enforces structure}$$



Goal 2: Discover the structure of X

\implies finding $\{\mathcal{G}_1, \dots, \mathcal{G}_L\} \iff$ learning the hyperparameter $\theta = [\theta_1 \cdots \theta_L] \in \{0, 1\}^{P \times L}$

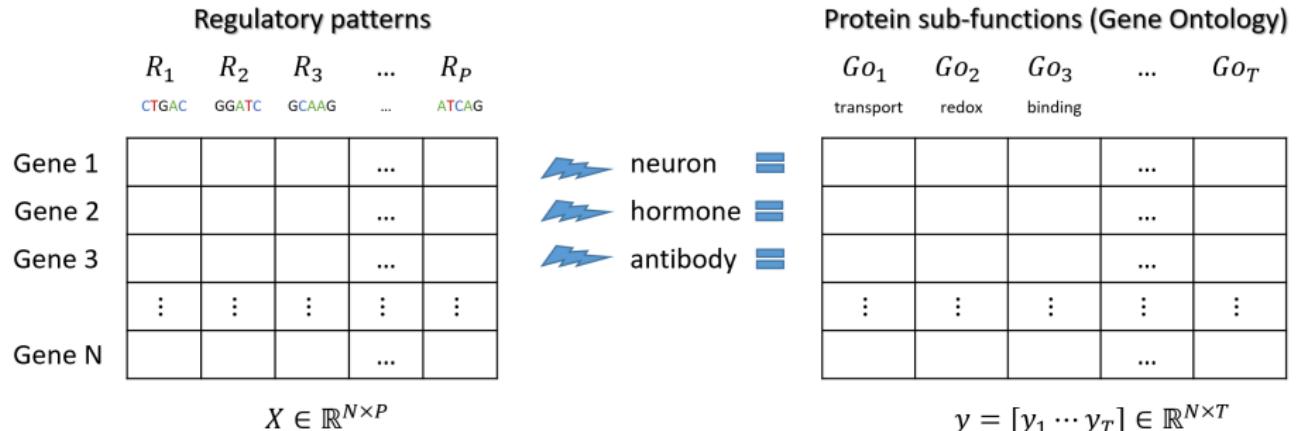
Reminder



Goal 1: Predict y from X

- to generalize: X and y are not only made of 0's and 1's
- to simplify: we first assume that $T = 1$ and omit the index t

Reminder



Goal 1: Predict y from X

→ to generalize: X and y are not only made of 0's and 1's



Now we consider all T tasks

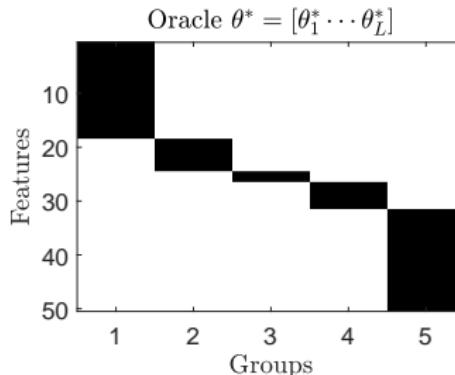
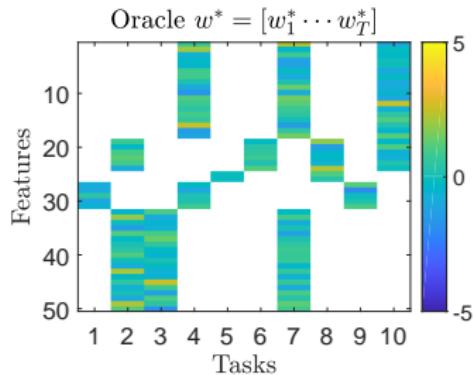
$$y \in \mathbb{R}^N \rightarrow y \in \mathbb{R}^{N \times T}$$

$$w \in \mathbb{R}^P \rightarrow w \in \mathbb{R}^{P \times T}$$

Single task $w \in \mathbb{R}^P \longrightarrow$ Multi-task $[w_1 \cdots w_T] \in \mathbb{R}^{P \times T}$

Multi-task setting : T tasks sharing the same group structure

$$(\forall t \in \{1, \dots, T\}) \quad \hat{w}_t(\theta) \in \operatorname{argmin}_{w_t \in \mathbb{R}^P} \frac{1}{2} \|y_t - Xw_t\|^2 + \lambda \sum_{l=1}^L \|\theta_l \odot w_t\|_2,$$



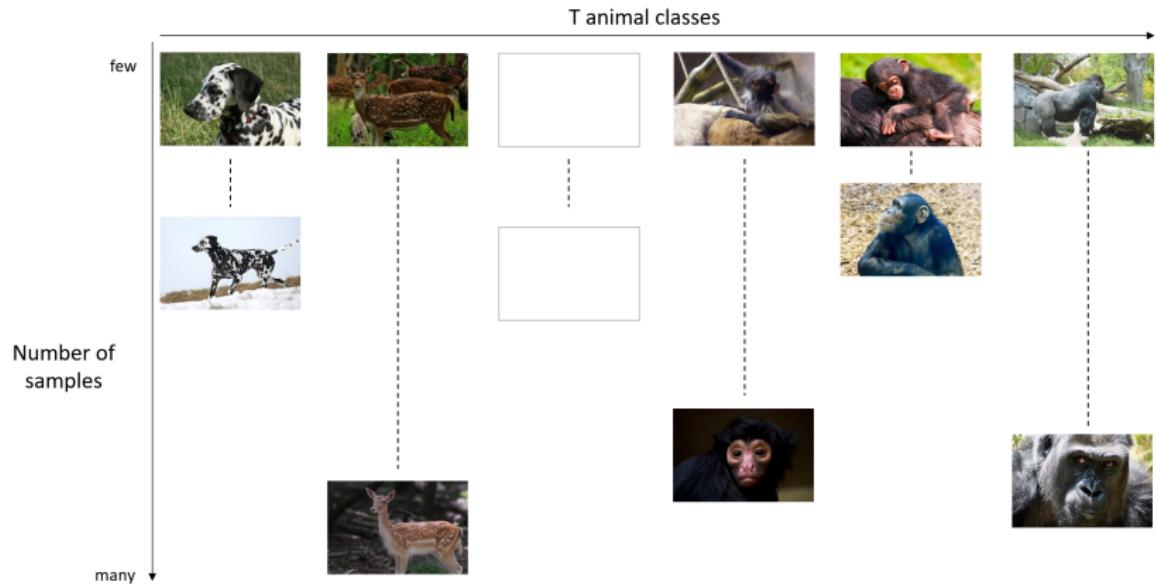
Some groups are relevant (non-zero) for some tasks and irrelevant (zero) for others

Example 2: Multi-task classification

Setting presented in

[Frecon et al. "Unveiling groups of related tasks in multi-task learning". ICPR (2020)]

Motivation: animal recognition

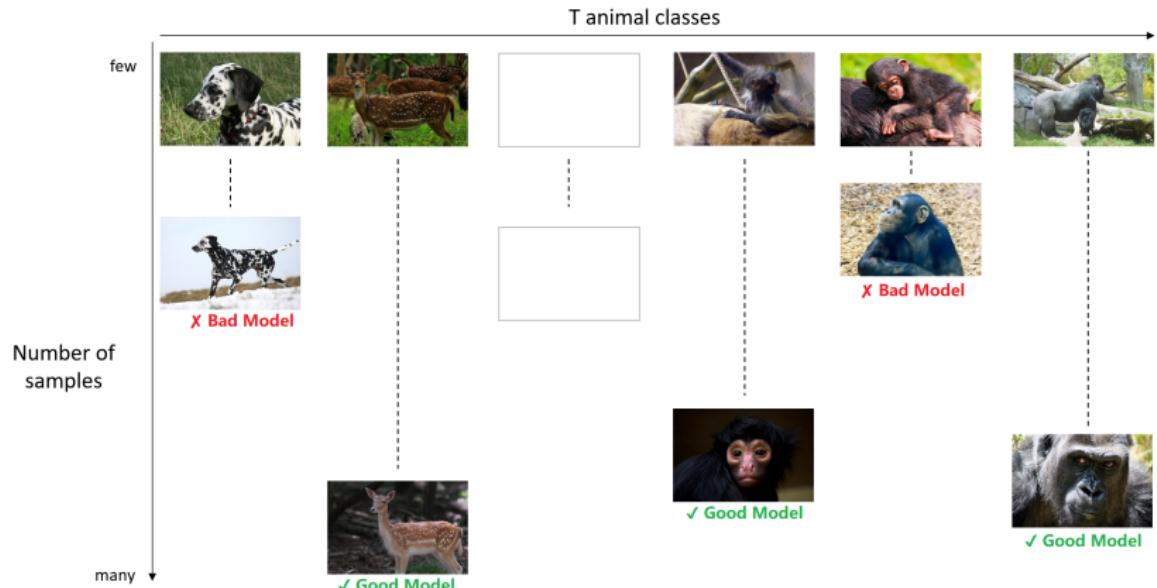


$X \in \mathbb{R}^{N \times P}$ made of N vectorized image of P pixels

$y \in \mathbb{R}^{N \times T}$ such that $y_{i,t} = 1$ if X_i belongs to the t -th animal class, 0 otherwise.

Issue: some classes of animals with very few samples (e.g., dalmatians & chimpanzees)

Motivation: animal recognition

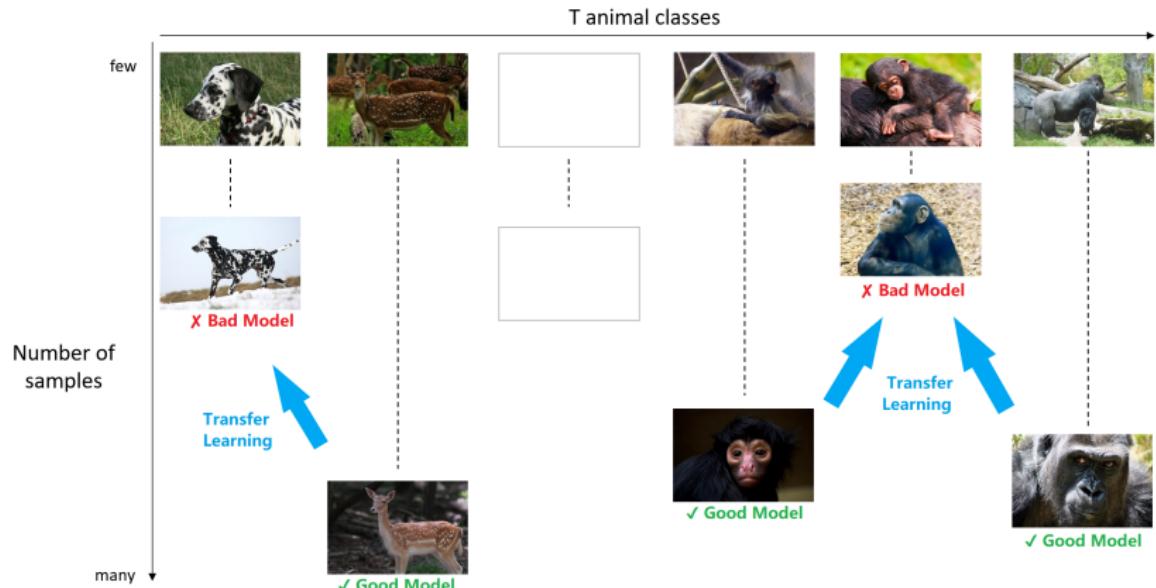


Naive idea: T binary classification tasks of one type of animal vs. all

logistic model: $(\forall t \in \{1, \dots, T\}), \exists w_t \in \mathbb{R}^P \mid y_t \sim \text{Bernouilli}(p_t)$ with $p_t = \frac{1}{1+\exp(-Xw_t)}$

estimate $w_1, \dots, w_T \Rightarrow T$ independent binary logistic regressions

Motivation: animal recognition



Proposed idea: learn classifiers of similar animals jointly

logistic model: $(\forall t \in \{1, \dots, T\}), \exists w_t \in \mathbb{R}^P \mid y_t \sim \text{Bernouilli}(p_t)$ with $p_t = \frac{1}{1+\exp(-Xw_t)}$

estimate $w = [w_1 \cdots w_T] \Rightarrow$ Multi-task binary logistic regression

How to transfer learning between similar tasks

Model the observations

$(\forall t \in \{1, \dots, T\}), \exists w_t \in \mathbb{R}^P \mid y_t \sim \text{Bernoulli}(p_t) \text{ with } p_t = 1/(1 + \exp(-Xw_t))$

Model task-relatedness

dalmatian (white + black spots) \approx deer (brown + white spots)

\Rightarrow find $w_{\text{dalmatian}} \propto w_{\text{deer}}$ or more generally $[w_{\text{dalmatian}} \, w_{\text{deer}}]$ low-rank

Task 1



W_1

Task 2



W_2

Task 3



W_3

Task 4



W_4

Task 5



W_5

$[W_1 \, W_2]$ low rank

Group 1: "4 legs + spots"

$[W_3 \, W_4 \, W_5]$ low rank

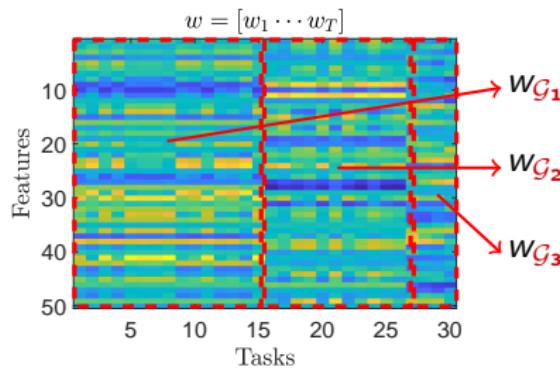
Group 2: primates

Optimization problem

Multi-task logistic regression: [Pong et al. (2010)] → here extended to L groups

$$\hat{w}(\mathcal{G}_1, \dots, \mathcal{G}_L) = \underset{w=[w_1 \cdots w_T]}{\operatorname{argmin}} \sum_{t=1}^T \underbrace{\log(1 - y_t \cdot Xw_t)}_{\propto -\log p(y_t | Xw_t)} + \lambda \sum_{l=1}^L \underbrace{\|w_{\mathcal{G}_l}\|_{\text{tr}}}_{\text{enforces structure}}$$

Trace norm $\|\cdot\|_{\text{tr}} = \text{sum of singular values} \Rightarrow \text{enforces low-rank}$



L partitions $\mathcal{G}_1, \dots, \mathcal{G}_L$ of T tasks

$$\mathcal{G}_l \subseteq \{1, \dots, T\}$$

$$\mathcal{G}_l \cap \mathcal{G}_{l'} = \emptyset \text{ if } l \neq l'$$

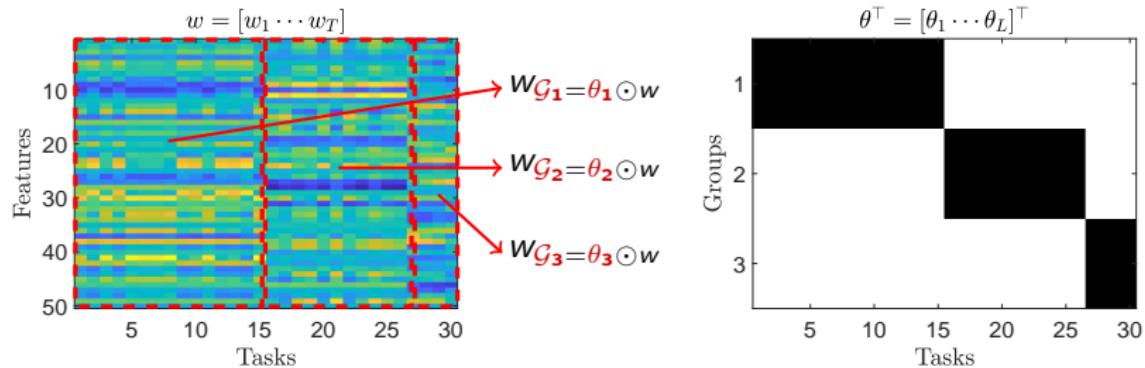
$$\cup_{l=1}^L \mathcal{G}_l = \{1, \dots, T\}$$

Optimization problem

Multi-task logistic regression: [Pong et al. (2010)] → here extended to L groups

$$\hat{w}(\mathcal{G}_1, \dots, \mathcal{G}_L) = \underset{w=[w_1 \cdots w_T]}{\operatorname{argmin}} \sum_{t=1}^T \underbrace{\log(1 - y_t \cdot Xw_t)}_{\propto -\log p(y_t | Xw_t)} + \lambda \sum_{l=1}^L \underbrace{\|w_{\mathcal{G}_l}\|_{\text{tr}}}_{\text{enforces structure}}$$

Trace norm $\|\cdot\|_{\text{tr}} = \text{sum of singular values} \Rightarrow \text{enforces low-rank}$



Mask $\theta_l = \{0, 1\}^T$ of the l -th group

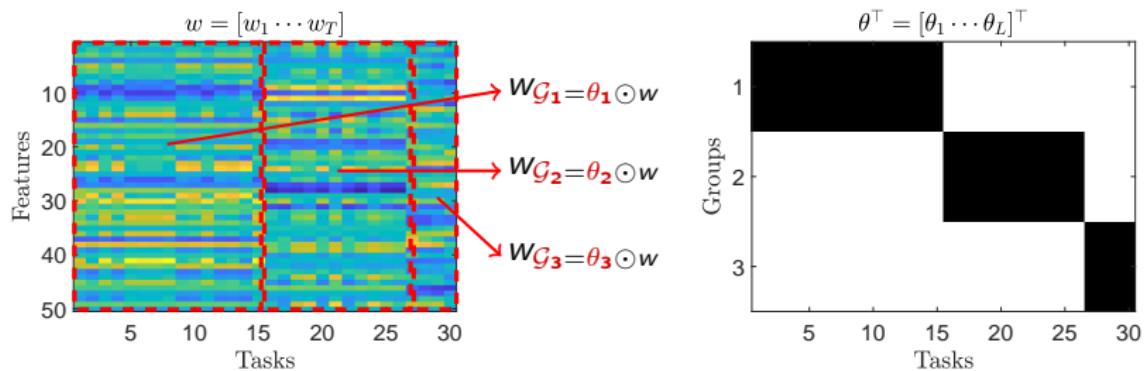
tasks-wise multiplication $\theta_l \odot w = [\theta_{l,1}w_1, \theta_{l,2}w_2, \dots, \theta_{l,T}w_T]^\top$

Optimization problem

Multi-task logistic regression: [Pong et al. (2010)] → here extended to L groups

$$\hat{w}(\mathcal{G}_1, \dots, \mathcal{G}_L) = \underset{w=[w_1 \cdots w_T]}{\operatorname{argmin}} \sum_{t=1}^T \underbrace{\log(1 - y_t \cdot Xw_t)}_{\propto -\log p(y_t | Xw_t)} + \lambda \sum_{l=1}^L \underbrace{\|w_{\mathcal{G}_l}\|_{\text{tr}}}_{\text{enforces structure}}$$

Trace norm $\|\cdot\|_{\text{tr}} = \text{sum of singular values} \Rightarrow \text{enforces low-rank}$



When the optimal groups are unknown

⇒ find $\{\mathcal{G}_1, \dots, \mathcal{G}_L\} \iff \text{learn the hyperparameter } \theta = [\theta_1 \cdots \theta_L] \in \{0, 1\}^{T \times L}$

Disclaimer

These are **motivating** examples to introduce the optimization problems
In practice, it is more complex ...

See the work of practitioners:

- Gene expressions [Higuera et al., (2015)]
- Animals images [Lambert et al. (2009)]
- Brain signals [Sabbagh et al. (2019)]

Proposed Framework

Setting presented in
[Frecon et al. "Bilevel optimization of group-wise penalties". Upcoming (2022)]

group-wise regularized optimization problem

In both examples, the prediction phase requires to solve

$$\hat{w}(\theta_1, \dots, \theta_L) = \underset{w}{\operatorname{argmin}} \left\{ \mathcal{L}(w; \theta) \triangleq \underbrace{\ell(y, \langle X, w \rangle)}_{\propto -\log p(y|Xw)} + \underbrace{\sum_{l=1}^L \rho_l(\theta_l \odot w)}_{\text{enforces structure}} \right\}$$

The structure is :

- encapsulated into $\theta = [\theta_1 \cdots \theta_L]$
- applied by the bilinear mapping \odot
- enforced by the norms ρ_l

Given the parameter matrix $w \in \mathbb{R}^{P \times T}$ made of P features and T tasks

- Grouping features (1st example) $\theta_l \in \{0, 1\}^P$ and $\sum_{l=1}^L \theta_l = \mathbb{1}_P$
- Grouping tasks (2nd example) $\theta_l \in \{0, 1\}^T$ and $\sum_{l=1}^L \theta_l = \mathbb{1}_T$

Learning the group structure θ

In many scenarios, the group structure θ is unknown or partly known



Learning θ to improve results

Issue: difficult combinatorial problem

the number of possible partitions grows exponentially with the dimension
⇒ trying them all is out of reach

Idea: relax and optimize

$$\begin{array}{ll} \text{indicators/masks } \theta = [\theta_1 \dots \theta_L] & \xrightarrow{\text{relaxation}} \\ \theta_I \text{ mask of group } I & \text{probabilities } \theta = [\theta_1 \dots \theta_L] \in \Theta \text{ simplex} \\ \theta_{I,i} \in \{0,1\} & \theta_I \text{ probability to belong to group } I \\ & \theta_{I,i} \in [0,1] \end{array}$$

Learning the group structure θ

In many scenarios, the group structure θ is unknown or partly known



Learning θ to improve results

Issue: difficult combinatorial problem

the number of possible partitions grows exponentially with the dimension
⇒ trying them all is out of reach

Idea: relax and optimize

$$\begin{array}{ccc} \text{indicators/masks } \theta = [\theta_1 \dots \theta_L] & \xrightarrow{\text{relaxation}} & \text{probabilities } \theta = [\theta_1 \dots \theta_L] \in \Theta \text{ simplex} \\ \theta_I \text{ mask of group } I & & \theta_I \text{ probability to belong to group } I \\ \theta_{I,i} \in \{0, 1\} & & \theta_{I,i} \in [0, 1] \end{array}$$

Learning the group structure θ

In many scenarios, the group structure θ is unknown or partly known



Learning θ to improve results

Issue: difficult combinatorial problem

the number of possible partitions grows exponentially with the dimension
⇒ trying them all is out of reach

Idea: relax and optimize

$$\begin{array}{l} \text{indicators/masks } \theta = [\theta_1 \dots \theta_L] \\ \theta_I \text{ mask of group } I \\ \theta_{I,i} \in \{0, 1\} \end{array} \xrightarrow{\text{relaxation}} \begin{array}{l} \text{probabilities } \theta = [\theta_1 \dots \theta_L] \in \Theta \text{ simplex} \\ \theta_I \text{ probability to belong to group } I \\ \theta_{I,i} \in [0, 1] \end{array}$$

Optimizing the probabilities θ

We would like to find the groups $\theta \in \Theta$ such that the structured predictor

$$\hat{w}(\theta) = \operatorname{argmin}_w \left\{ \mathcal{L}(w; \theta) \triangleq \ell(y, \langle X, w \rangle) + \sum_{l=1}^L \rho_l(\theta_l \odot w) \right\}$$

generalizes well to unseen data

Idea: Find θ such that $\hat{w}(\theta)$ minimizes the validation error $\mathcal{E}(\hat{w}(\theta)) = \ell(y^{\text{val}}, \langle X^{\text{val}}, \hat{w}(\theta) \rangle)$
⇒ type of continuous cross-validation

Bilevel Problem

$$\min_{\theta \in \Theta} \mathcal{E}(\hat{w}(\theta)) \quad \text{s.t.} \quad \hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$$

Optimizing the probabilities θ

We would like to find the groups $\theta \in \Theta$ such that the structured predictor

$$\hat{w}(\theta) = \operatorname{argmin}_w \left\{ \mathcal{L}(w; \theta) \triangleq \underbrace{\ell(y, \langle X, w \rangle)}_{\text{training error}} + \sum_{l=1}^L \rho_l(\theta_l \odot w) \right\}$$

generalizes well to unseen data

Idea: Find θ such that $\hat{w}(\theta)$ minimizes the validation error $\mathcal{E}(\hat{w}(\theta)) = \ell(y^{\text{val}}, \langle X^{\text{val}}, \hat{w}(\theta) \rangle)$
⇒ type of continuous cross-validation

Bilevel Problem

$$\min_{\theta \in \Theta} \mathcal{E}(\hat{w}(\theta)) \quad \text{s.t.} \quad \hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$$

Optimizing the probabilities θ

We would like to find the groups $\theta \in \Theta$ such that the structured predictor

$$\hat{w}(\theta) = \operatorname{argmin}_w \left\{ \mathcal{L}(w; \theta) \triangleq \underbrace{\ell(y, \langle X, w \rangle)}_{\text{training error}} + \sum_{l=1}^L \rho_l(\theta_l \odot w) \right\}$$

generalizes well to unseen data

Idea: Find θ such that $\hat{w}(\theta)$ minimizes the validation error $\mathcal{E}(\hat{w}(\theta)) = \ell(y^{\text{val}}, \langle X^{\text{val}}, \hat{w}(\theta) \rangle)$
⇒ type of continuous cross-validation

Bilevel Problem

$$\min_{\theta \in \Theta} \mathcal{E}(\hat{w}(\theta)) \quad \text{s.t.} \quad \hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$$

Exact Problem

$$\min_{\theta \in \Theta} \left\{ \mathcal{U}(\theta) \triangleq \mathcal{E}(\hat{w}(\theta)) \right\}$$

$$\text{s.t. } \hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$$

$\hat{w}(\theta)$ without closed form

Exact Problem

$$\min_{\theta \in \Theta} \left\{ \mathcal{U}(\theta) \triangleq \mathcal{E}(\hat{w}(\theta)) \right\}$$

s.t. $\hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$

$\hat{w}(\theta)$ without closed form

Approximate Problem

$$\min_{\theta \in \Theta} \left\{ \mathcal{U}^{(k)}(\theta) \triangleq \mathcal{E}(w^{(k)}(\theta)) \right\}$$

s.t.

- $w^{(0)}(\theta)$ chosen arbitrarily
- for $i = 0, \dots, k - 1$
- $| \quad w^{(i+1)}(\theta) = \mathcal{A}(w^{(i)}(\theta))$
- $w^{(k)}(\theta) \rightarrow \hat{w}(\theta)$

$\mathcal{U}^{(k)}$ smooth if \mathcal{A} smooth



choice of \mathcal{A} discussed next

Exact Problem

$$\min_{\theta \in \Theta} \left\{ \mathcal{U}(\theta) \triangleq \mathcal{E}(\hat{w}(\theta)) \right\}$$

s.t. $\hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$

$\hat{w}(\theta)$ without closed form

Approximate Problem

$$\min_{\theta \in \Theta} \left\{ \mathcal{U}^{(k)}(\theta) \triangleq \mathcal{E}(w^{(k)}(\theta)) \right\}$$

w⁽⁰⁾(θ) chosen arbitrarily
 for $i = 0, \dots, k - 1$
 s.t. $\lfloor w^{(i+1)}(\theta) = \mathcal{A}(w^{(i)}(\theta))$
 $w^{(k)}(\theta) \rightarrow \hat{w}(\theta)$

$\mathcal{U}^{(k)}$ smooth if \mathcal{A} smooth

$\theta^{(0)}$ chosen arbitrarily

for $n = 0, 1, \dots$

$w^{(0)}(\theta^{(n)})$ chosen arbitrarily
 for $i = 0, \dots, k - 1$ (inner algorithm)
 $\lfloor w^{(i+1)}(\theta^{(n)}) = \mathcal{A}(w^{(i)}(\theta^{(n)}))$
 $\theta^{(n+1)} = \operatorname{Proj}_{\Theta}(\theta^{(n)} - \gamma \nabla \mathcal{U}^{(k)}(\theta^{(n)}))$ where $\mathcal{U}^{(k)}(\theta^{(n)}) = \mathcal{E}(w^{(k)}(\theta^{(n)}))$

Algorithmic Solution

Optimization problem

$$\underset{w \in \mathbb{R}^P}{\text{minimize}} \quad \underbrace{\frac{1}{2} \|y - Xw\|_2^2}_{f(w) \text{ differentiable}} + \underbrace{\lambda \sum_{l=1}^L \|\theta_l \odot w\|_2}_{g(A_\theta w) \text{ non differentiable}}$$

where $A_\theta : w \in \mathbb{R}^P \mapsto (\theta_1 \odot w, \dots, \theta_L \odot w) \in \mathbb{R}^{P \times L}$

Forward-backward algorithm [Combettes and Wajs (2005)]

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left[\begin{array}{l} w^{(i+1)}(\theta) = \text{prox}_{\beta g \circ A_\theta}(w^{(i)}(\theta) - \beta \nabla f(w^{(i)}(\theta))) \end{array} \right] \end{array} \right.$$

proximity operator (prox) ? → see next slide

Group Lasso solver \mathcal{A}

Optimization problem

$$\underset{w \in \mathbb{R}^P}{\text{minimize}} \quad \underbrace{\frac{1}{2} \|y - Xw\|_2^2}_{f(w) \text{ differentiable}} + \underbrace{\lambda \sum_{l=1}^L \|\theta_l \odot w\|_2}_{g(A_\theta w) \text{ non differentiable}}$$

where $A_\theta : w \in \mathbb{R}^P \mapsto (\theta_1 \odot w, \dots, \theta_L \odot w) \in \mathbb{R}^{P \times L}$

Forward-backward algorithm [Combettes and Wajs (2005)]

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left[\begin{array}{l} w^{(i+1)}(\theta) = \text{prox}_{\beta g \circ A_\theta}(w^{(i)}(\theta) - \beta \nabla f(w^{(i)}(\theta))) \end{array} \right] \end{array} \right.$$

proximity operator (prox) ? → see next slide

Proximity operator

In the 1960s, [Moreau (1962)] proposed an extension of the notion of projection operator to any convex function h , leading to the so-called proximity operator

$$\begin{aligned}\text{Proj}_{\mathcal{C}}(v) &= \underset{w \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{2} \|w - v\|_2^2 \\ &= \underset{w \in \mathbb{R}^P}{\operatorname{argmin}} \iota_{\mathcal{C}}(w) + \frac{1}{2} \|w - v\|_2^2 \quad \text{where} \quad \iota_{\mathcal{C}}(w) = \begin{cases} 0 & \text{if } w \in \mathcal{C} \\ +\infty & \text{otherwise} \end{cases}\end{aligned}$$

$$\text{prox}_h(v) = \underset{w \in \mathbb{R}^P}{\operatorname{argmin}} h(w) + \frac{1}{2} \|w - v\|_2^2$$

Group Lasso solver \mathcal{A}

Optimization problem

$$\underset{w \in \mathbb{R}^P}{\text{minimize}} \quad \underbrace{\frac{1}{2} \|y - Xw\|_2^2}_{f(w) \text{ differentiable}} + \underbrace{\lambda \sum_{l=1}^L \|\theta_l \odot w\|_2}_{g(A_\theta w) \text{ non differentiable}}$$

where $A_\theta : w \in \mathbb{R}^P \mapsto (\theta_1 \odot w, \dots, \theta_L \odot w) \in \mathbb{R}^{P \times L}$

Forward-backward algorithm [Combettes and Wajs (2005)]

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left[\begin{array}{l} w^{(i+1)}(\theta) = \text{prox}_{\beta g \circ A_\theta}(w^{(i)}(\theta) - \beta \nabla f(w^{(i)}(\theta))) \end{array} \right] \end{array} \right.$$

*generalization of projected gradient descent
projection \rightarrow proximity operator*

$$\text{prox}_{\beta g \circ A_\theta}(v) = \underset{w \in \mathbb{R}^P}{\operatorname{argmin}} \beta g(A_\theta w) + \frac{1}{2} \|w - v\|_2^2 \quad \triangleq \text{without closed form}$$

Group Lasso solver \mathcal{A}

Optimization problem

$$\underset{w \in \mathbb{R}^P}{\text{minimize}} \quad \underbrace{\frac{1}{2} \|y - Xw\|_2^2}_{f(w) \text{ differentiable}} + \underbrace{\lambda \sum_{l=1}^L \|\theta_l \odot w\|_2}_{g(A_\theta w) \text{ non differentiable}}$$

where $A_\theta : w \in \mathbb{R}^P \mapsto (\theta_1 \odot w, \dots, \theta_L \odot w) \in \mathbb{R}^{P \times L}$

Forward-backward algorithm [Combettes and Wajs (2005)]

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left[\begin{array}{l} w^{(i+1)}(\theta) = \text{prox}_{\beta g \circ A_\theta}(w^{(i)}(\theta) - \beta \nabla f(w^{(i)}(\theta))) \end{array} \right] \end{array} \right.$$

*generalization of projected gradient descent
projection → proximity operator*

$$\text{prox}_{\beta g \circ A_\theta}(v) = \underset{w \in \mathbb{R}^P}{\operatorname{argmin}} \beta g(A_\theta w) + \frac{1}{2} \|w - v\|_2^2 \quad \text{⚠ without closed form}$$

Duality in convex optimization

The ideas of duality and transforms are ubiquitous in mathematics

- Harmonics analysis → Fourier transform
- Convex analysis → Fenchel conjugate: $h^*(x) = \sup_w \langle w, x \rangle - h(w)$

[Rockafellar (1970)]

Example: $h: w \mapsto \|w\|_2 \implies h^*: x \mapsto \iota_{B(1)}(x) = \begin{cases} 0 & \text{if } \|x\|_2 \leq 1 \\ +\infty & \text{otherwise} \end{cases}$

Duality in convex optimization

The ideas of duality and transforms are ubiquitous in mathematics

- Harmonics analysis → Fourier transform
- Convex analysis → Fenchel conjugate: $h^*(x) = \sup_w \langle w, x \rangle - h(w)$

[Rockafellar (1970)]

Example: $h: w \mapsto \lambda \|w\|_2 \quad \Rightarrow \quad h^*: x \mapsto \iota_{B(\lambda)}(x) = \begin{cases} 0 & \text{if } \|x\|_2 \leq \lambda \\ +\infty & \text{otherwise} \end{cases}$

Duality in convex optimization

The ideas of duality and transforms are ubiquitous in mathematics

- Harmonics analysis → Fourier transform
- Convex analysis → Fenchel conjugate: $h^*(x) = \sup_w \langle w, x \rangle - h(w)$

[Rockafellar (1970)]

Primal problem

↔

Dual problem

$$\underset{w \in \mathbb{R}^P}{\text{minimize}} f(w) + g(\mathbf{A}_\theta w)$$

$$\mathbf{A}_\theta : \mathbb{R}^P \rightarrow \mathbb{R}^{P \times L}$$

$$g(v_1 \dots v_L) = \sum_{l=1}^L \underbrace{\lambda \|v_l\|_2}_{\text{norm}}$$

$$\underset{u \in \mathbb{R}^{P \times L}}{\text{minimize}} f^*(-\mathbf{A}_\theta^\top u) + g^*(u)$$

$$\mathbf{A}_\theta^\top : \mathbb{R}^{P \times L} \rightarrow \mathbb{R}^P$$

$$g^*(u_1 \dots u_L) = \sum_{l=1}^L \underbrace{\iota_{\mathcal{B}(\lambda)}(u_l)}_{\substack{\text{indicator} \\ \text{dual norm ball}}}$$

Link

$$w = \nabla f^*(-\mathbf{A}_\theta^\top u)$$

$\text{prox}_{g \circ \mathbf{A}_\theta}$ without closed form \Rightarrow solve dual problem to move \mathbf{A}_θ in smooth part

Dual problem

$$\underset{u \in \mathbb{R}^{P \times L}}{\text{minimize}} \underbrace{f^*(-A_\theta^\top u)}_{\text{differentiable}} + \underbrace{g^*(u)}_{\text{non differentiable}}$$

Dual forward-backward algorithm

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*}(u^{(i)}(\theta) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta))) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta^* u^{(k)}(\theta)) \quad (\text{link}) \end{array} \right. \end{array} \right.$$

Dual problem

$$\underset{u \in \mathbb{R}^{P \times L}}{\text{minimize}} \underbrace{f^*(-A_\theta^\top u)}_{\text{differentiable}} + \underbrace{g^*(u)}_{\text{non differentiable}}$$

Dual forward-backward algorithm

$$\begin{cases} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*}(u^{(i)}(\theta) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta))) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta^* u^{(k)}(\theta)) \quad (\text{link}) \end{array} \right. \end{cases}$$

where the proximal operator reads:

$$\begin{aligned} \text{prox}_{\beta g^*}(v) &= \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \beta g^*(u) + \frac{1}{2} \|u - v\|^2 \\ &= \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \beta \sum_{l=1}^L \iota_{\mathcal{B}(\lambda)}(u_l) + \frac{1}{2} \|u - v\|^2 \\ &= \text{Proj}_{\mathcal{B}(\lambda)^L}(v) \quad \text{⚠️ not differentiable} \end{aligned}$$

Reminder: why differentiability is important

$\theta^{(0)}$ chosen arbitrarily

for $n = 0, 1, \dots$

(inner algorithm = dual forward-backward)

$w^{(0)}(\theta^{(n)})$ chosen arbitrarily	for $i = 0, \dots, k - 1$	$w^{(i+1)}(\theta^{(n)}) = \mathcal{A}(w^{(i)}(\theta^{(n)}))$
$\theta^{(n+1)} = \text{Proj}_{\Theta}(\theta^{(n)} - \gamma \nabla \mathcal{U}^{(k)}(\theta^{(n)}))$ where $\mathcal{U}^{(k)}(\theta^{(n)}) = \mathcal{E}(w^{(k)}(\theta^{(n)}))$		

We want a **differentiable dual forward-backward** algorithm
because it inside a bilevel algorithm !

Group Lasso solver \mathcal{A} : dual approach

Dual forward-backward algorithm

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*} \left(u^{(i)}(\theta) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta)) \right) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta u^{(k)}(\theta)). \end{array} \right. \end{array} \right.$$

where

$$\begin{aligned} \text{prox}_{\beta g^*}(v) &= \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \beta g^*(u) + \frac{1}{2} \|u - v\|^2 \\ &= \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \beta g^*(u) + \frac{1}{2} \|u\|^2 - \langle u, v \rangle + \text{cst} \end{aligned}$$

Dual forward-backward algorithm

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*} \left(u^{(i)}(\theta) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta)) \right) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta u^{(k)}(\theta)). \end{array} \right. \end{array} \right.$$

where

$$\text{prox}_{\beta g^*}(v) = \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \beta g^*(u) + \frac{1}{2} \|u\|^2 - \langle u, v \rangle + \text{cst}$$

Group Lasso solver \mathcal{A} : dual approach

Dual forward-backward algorithm **with Bregman distances** [Bauschke et al. (2016)]

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*}^\Phi(\nabla \Phi(u^{(i)}(\theta)) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta))) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta u^{(k)}(\theta)). \end{array} \right. \end{array} \right.$$

where the Bregman proximal operator associated to Φ :

$$\text{prox}_{\beta g^*}^\Phi(v) = \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \beta g^*(u) + \Phi(u) - \langle u, v \rangle$$

Choice of Φ to smooth the updates

Dual forward-backward algorithm with Bregman distances

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*}^\Phi (\nabla \Phi(u^{(i)}(\theta)) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta))) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta u^{(k)}(\theta)). \end{array} \right. \end{array} \right.$$

where

$$\text{prox}_{\beta g^*}^\Phi(v) = \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \beta g^*(u) + \Phi(u) - \langle u, v \rangle$$

Choice of Φ to smooth the updates

Dual forward-backward algorithm with Bregman distances

$$\left\{ \begin{array}{l} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*}^\Phi(\nabla \Phi(u^{(i)}(\theta)) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta))) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta u^{(k)}(\theta)). \end{array} \right. \end{array} \right.$$

where

$$\text{prox}_{\beta g^*}^\Phi(v) = \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \sum_{l=1}^L \varphi_{B(\lambda)}(u_l) + \Phi(u) - \langle u, v \rangle$$

Choice of Φ to smooth the updates

Dual forward-backward algorithm with Bregman distances

$$\begin{cases} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*}^\Phi (\nabla \Phi(u^{(i)}(\theta)) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta))) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta u^{(k)}(\theta)). \end{array} \right. \end{cases}$$

where

$$\text{prox}_{\beta g^*}^\Phi(v) = \underset{u \in \mathbb{R}^P \times L}{\operatorname{argmin}} \sum_{l=1}^L (\varphi_{\mathcal{B}(\lambda)}(u_l) + \phi(u_l) - \langle u_l, v_l \rangle)$$

for $\Phi(u) = \sum_{l=1}^L \phi(u_l)$

Choice of Φ to smooth the updates

Dual forward-backward algorithm with Bregman distances

$$\begin{cases} \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \text{prox}_{\beta g^*}^\Phi(\nabla \Phi(u^{(i)}(\theta)) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta))) \\ w^{(k)}(\theta) = \nabla f^*(-A_\theta u^{(k)}(\theta)). \end{array} \right. \end{cases}$$

where

$$\text{prox}_{\beta g^*}^\Phi(v) = \underset{u \in \mathbb{R}^{P \times L}}{\operatorname{argmin}} \sum_{l=1}^L \left(\iota_{\mathcal{B}(\lambda)}(u_l) - \sqrt{\lambda^2 - \|u_l\|^2} - \langle u_l, v_l \rangle \right)$$

for $\phi(u_l) = -\sqrt{\lambda^2 - \|u_l\|^2} \Rightarrow \text{dom } \phi = \mathcal{B}(\lambda)$
 $\Rightarrow \iota_{\mathcal{B}(\lambda)}(u_l)$ always equal to 0 !

⚠️ trick for a differentiable algorithm

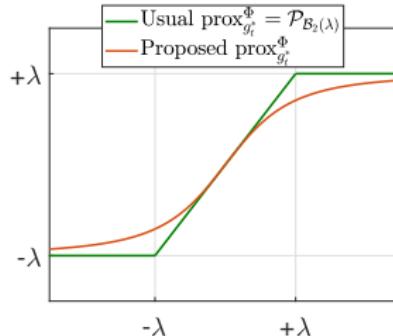
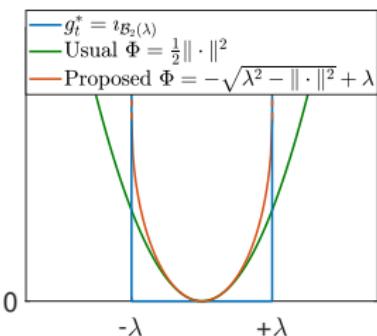
Choice of Φ to smooth the updates

Dual forward-backward algorithm with Bregman distances

$$\begin{cases} \text{for } i = 0, \dots, k-1 \\ \quad \left[u^{(i+1)}(\theta) = \text{prox}_{\beta g^*}^\Phi (\nabla \Phi(u^{(i)}(\theta)) + \beta A_\theta \nabla f^*(-A_\theta^\top u^{(i)}(\theta))) \right. \\ \quad \left. w^{(k)}(\theta) = \nabla f^*(-A_\theta u^{(k)}(\theta)). \right. \end{cases}$$

where

$$\text{prox}_{\beta g^*}^\Phi(v) = \left(\frac{\lambda v_l}{\sqrt{1 + \|v\|_2^2}} \right)_{l=1, \dots, L}$$



Convergence $w^{(k)}(\theta) \rightarrow \hat{w}(\theta)$

Theorem 1: For every $\theta \in \Theta$, $\|w^{(k)}(\theta) - \hat{w}(\theta)\|^2 \leq \frac{\text{Const}}{k}$

Convergence " $\mathcal{U}^{(k)}(\theta) \rightarrow \mathcal{U}(\theta)$ "

Theorem 2: Assume that Θ is a non-empty compact subset of $\mathbb{R}_+^{P \times L}$. If the iterates $\{w^{(k)}(\theta)\}_{k \in \mathbb{N}}$ converge to $\hat{w}(\theta)$ uniformly in Θ when $k \rightarrow +\infty$, then

$$\inf_{\theta \in \Theta} \mathcal{U}^{(k)}(\theta) \xrightarrow[k \rightarrow +\infty]{} \inf_{\theta \in \Theta} \mathcal{U}(\theta) \quad \text{and} \quad \operatorname{argmin}_{\theta \in \Theta} \mathcal{U}^{(k)}(\theta) \xrightarrow[k \rightarrow +\infty]{} \operatorname{argmin}_{\theta \in \Theta} \mathcal{U}(\theta)$$

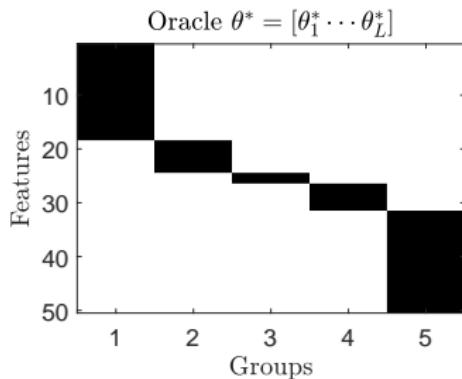
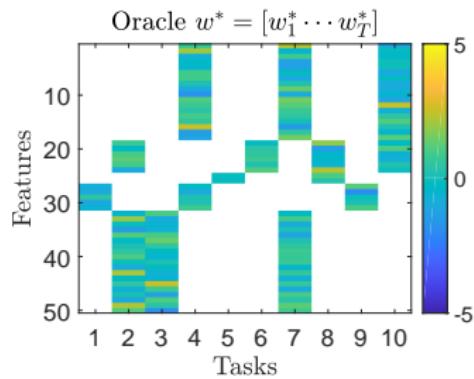
Reminder : $\begin{cases} \mathcal{U}(\theta) &= \mathcal{E}(\hat{w}(\theta)) \\ \mathcal{U}^{(k)}(\theta) &= \mathcal{E}(\hat{w}^{(k)}(\theta)) \end{cases}$

Numerical Experiments

Setting

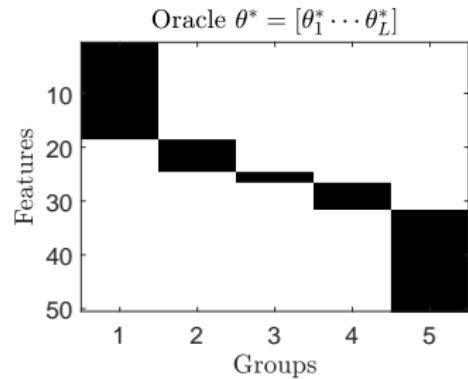
Setting: $T = 500$ tasks, $N = 25$ noisy observations, $P = 50$ parameters.

Goal: Estimate and group the parameters



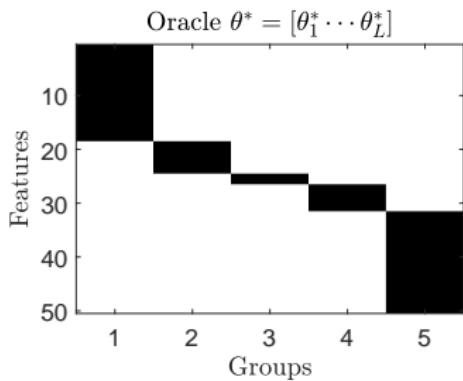
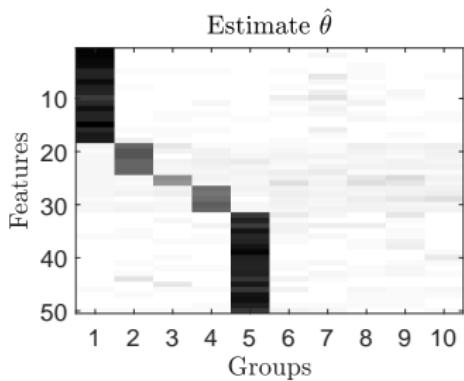
$$y_t = X_t w_t^* + \epsilon_t \quad \text{where} \quad \epsilon_t \sim \mathcal{N}(0, \sigma = 0.1)$$

Result



Recover the correct groups ! (just different ordering)

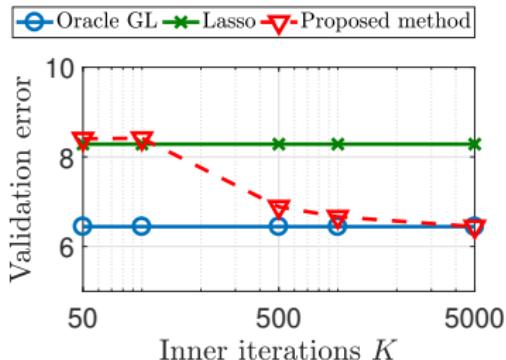
When the number of groups is unknown



Works even when the number of groups is unknown !

Phase transition

Reminder : $[w_1^{(k)}(\theta) \cdots w_T^{(k)}(\theta)] \rightarrow [\hat{w}_1(\theta) \cdots \hat{w}_T(\theta)]$



(Oracle GL) group Lasso with oracle groups
(Lasso) \Leftrightarrow GL with one group per variable
(BiGL) proposed method

Conclusion

1. Define structured predictor with **groups** θ

$$\hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$$

2. **Ideal:** find groups θ such that $\hat{w}(\theta)$ minimizes the **validation error**

$$\min_{\theta \in \Theta} \mathcal{E}(\hat{w}(\theta)) \quad \text{s.t.} \quad \hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$$

3. **Practice:** solve a **differentiable** bilevel problem

$$\begin{aligned} & w^{(0)}(\theta) \text{ chosen arbitrarily} \\ \min_{\theta \in \Theta} \mathcal{E}(w^{(k)}(\theta)) \quad & \text{s.t.} \\ & \quad \text{for } i = 0, \dots, k-1 \\ & \quad \quad \quad \lfloor w^{(i+1)}(\theta) = \mathcal{A}(w^{(i)}(\theta)) \quad \text{with } \mathcal{A} \text{ differentiable} \end{aligned}$$

What is next

1. More complex structures (overlapping, hierarchical, ...)
2. New multi-task models to transfer learning
3. Theoretical guarantees for bilevel optimization (global minima, convergence rate, ...)

Thank you

- "H. H. Bauschke, J. Bolte, and M. Teboulle. [A descent lemma beyond Lipschitz gradient continuity: first-order methods revisited and applications](#). Mathematics of Operations Research, 2016."
- "P. L Combettes and V. R. Wajs. [Signal recovery by proximal forward-backward splitting](#). In SIAM Multiscale Modeling & Simulation. 2005"
- "J. Frecon, S. Salzo, and M. Pontil. [Bilevel learning of the group Lasso structure](#). In Advances in Neural Information Processing Systems (NeurIPS). 2018"
- "J. Frecon, S. Salzo, and M. Pontil. [Unveiling groups of related tasks in multi-task learning](#). In Proceedings of the International Conference on Pattern Recognition (ICPR). 2020"
- "C. Higuera, K.J. Gardiner, K.J. Cios. [Self-organizing feature maps identify proteins critical to learning in a mouse model of down syndrome](#). In PLoS ONE. 2015"
- "Z. Kang, K. Grauman and F. Sha. [Learning with whom to share in multi-task feature learning](#). In Proceedings of the International Conference on Machine Learning (ICML), 2011."
- "M. Kshirsagar, E. Yang, A.C. Lozano. [Learning task clusters via sparsity grouped multitask learning](#). In Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), 2017."
- "C. H. Lampert, H. Nickisch, and S. Harmeling. [Learning to detect unseen object classes by between-class attribute transfer](#). In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2009"
- "J. J. Moreau. [Fonctions convexes duales et points proximaux dans un espace hilbertien](#). Comptes Rendus de l'Académie des Sciences de Paris. 1962."
- "T. K. Pong, P. Tseng, S. Ji, and J. Ye. [Trace norm regularization: reformulations, algorithms, and multi-task learning](#). In SIAM Journal of Optimization. 2010"
- "R.T. Rockafellar. [Convex analysis](#). Princeton university press. 1970"
- "D. Sabbagh, P. Ablin, G. Varoquaux, A. Gramfort, and D. Engemann. [Manifold-regression to predict from MEG/EEG brain signals without source modeling](#) In Advances in Neural Information Processing Systems. 2019"
- "M. Yuan and Y. Lin. [Model selection and estimation in regression with grouped variables](#). In Journal of the Royal Statistical Society: Series B (Statistical Methodology). 2006"

What is next (theoretical guarantees)

Exact problem

$$\min_{\theta \in \Theta} \left\{ \mathcal{U}(\theta) \triangleq \mathcal{E}(\hat{w}(\theta)) \right\}$$

s.t. $\hat{w}(\theta) = \operatorname{argmin}_{w \in \mathcal{W}} \mathcal{L}(w; \theta)$

Approximate problem

$$\min_{\theta \in \Theta} \left\{ \mathcal{U}^{(k)}(\theta) \triangleq \mathcal{E}(w^{(k)}(\theta)) \right\}$$

$w^{(0)}(\theta)$ chosen arbitrarily
for $i = 0, \dots, k - 1$
s.t. $\begin{cases} w^{(i+1)}(\theta) = \mathcal{A}(w^{(i)}(\theta)) \\ w^{(k)}(\theta) \rightarrow \hat{w}(\theta) \end{cases}$

$$\theta^{(n+1)} = \operatorname{Proj}_{\Theta}(\theta^{(n)} - \mu \nabla \mathcal{U}^{(k)}(\theta^{(n)}))$$

- $\inf_{\theta \in \Theta} \mathcal{U}^{(k)}(\theta) \xrightarrow{k \rightarrow +\infty} \inf_{\theta \in \Theta} \mathcal{U}(\theta)$ ✓
- $\operatorname{argmin}_{\theta \in \Theta} \mathcal{U}^{(k)}(\theta) \xrightarrow{k \rightarrow +\infty} \operatorname{argmin}_{\theta \in \Theta} \mathcal{U}(\theta)$ ✓
- Efficient computation of $\nabla \mathcal{U}^{(k)}$ ✓

- Impact of warm-restart on $w^{(0)}$?
- $\lim_{k \rightarrow \infty} \nabla \mathcal{U}^{(k)}(\theta) \in \partial \mathcal{U}(\theta)$?
- $\lim_{n \rightarrow \infty} \theta^{(n)} \in \partial \mathcal{U}^{-1}(0)$?

Convergence to a stationary point

Theorem 3: For \bar{n} uniformly sampled in $\{1, \dots, n_{\max}\}$:

$$\mathbb{E} \left[\|G_\gamma(\theta^{(\bar{n})})\|^2 \right] \leq \frac{\text{Const}}{n_{\max}},$$

where G_γ with step-size γ

$$G_\gamma(\theta) = \frac{1}{\gamma} (\theta - \mathcal{P}_\Theta(\theta - \gamma \nabla \mathcal{U}^{(k)}(\theta)))$$

Intuition : Without the projection, $G_\gamma(\theta) = \nabla \mathcal{U}^{(k)}(\theta)$

$$\left\{ \begin{array}{l} u^{(0)}(\theta) \in \mathcal{H} \\ \text{for } i = 0, \dots, k-1 \\ \quad \left| \begin{array}{l} u^{(i+1)}(\theta) = \mathcal{A}(u^{(i)}(\theta), \theta) \\ w^{(k)}(\theta) = \mathcal{B}(u^{(k)}(\theta), \theta), \end{array} \right. \end{array} \right. \quad (1)$$

we get

$$\nabla \mathcal{U}^{(k)}(\theta) = (u^{(k)})'(\theta)^\top \partial_1 \mathcal{B}(u^{(k)}(\theta), \theta)^\top \nabla C(w^{(k)}(\theta)) + \partial_2 \mathcal{B}(u^{(k)}(\theta), \theta)^\top \nabla C(w^{(k)}(\theta)). \quad (2)$$

Moreover, using the updating rule for $u^{(i)}(\theta)$ in (1) we have

$$(u^{(i+1)})'(\theta) = \partial_1 \mathcal{A}(u^{(i)}(\theta), \theta) (u^{(i)})'(\theta) + \partial_2 \mathcal{A}(u^{(i)}(\theta), \theta). \quad (3)$$

Setting $A_1^{(i)}(\theta) = \partial_1 \mathcal{A}(u^{(i)}(\theta), \theta)$ and $A_2^{(i)}(\theta) = \partial_2 \mathcal{A}(u^{(i)}(\theta), \theta)$, we have

$$(u^{(i+1)})'(\theta)^\top = (u^{(i)})'(\theta)^\top A_1^{(i)}(\theta)^\top + A_2^{(i)}(\theta)^\top. \quad (4)$$

Hypergradient Computation

Then, by combining the two equations above we have

$$\begin{aligned}\nabla \mathcal{U}^{(k)}(\theta) &= (u^{(k)})'(\theta)^\top \partial_1 \mathcal{B}(u^{(k)}(\theta), \theta)^\top \nabla C(w^{(k)}(\theta)) + \partial_2 \mathcal{B}(u^{(k)}(\theta), \theta)^\top \nabla C(w^{(k)}(\theta)) \\ &= (u^{(k-1)})'(\theta)^\top A_1^{(k-1)}(\theta)^\top \underbrace{\partial_1 \mathcal{B}(u^{(k)}(\theta), \theta)^\top \nabla C(w^{(k)}(\theta))}_{a_Q} \\ &\quad + A_2^{(k-1)}(\theta)^\top \underbrace{\partial_1 \mathcal{B}(u^{(k)}(\theta), \theta)^\top \nabla C(w^{(k)}(\theta))}_{a_k} + \underbrace{\partial_2 \mathcal{B}(u^{(k)}(\theta), \theta)^\top \nabla C(w^{(k)}(\theta))}_{b_k} \\ &= (u^{(k-1)})'(\theta)^\top \underbrace{A_1^{(k-1)}(\theta)^\top a_k}_{a_{k-1}} + \underbrace{A_2^{(k-1)}(\theta)^\top a_k + b_k}_{b_{k-1}} \\ &= (u^{(k-2)})'(\theta)^\top \underbrace{A_1^{(k-2)}(\theta)^\top a_{k-1}}_{a_{k-2}} + \underbrace{A_2^{(k-2)}(\theta)^\top a_{k-1} + b_{k-1}}_{b_{k-2}} \\ &= \dots \\ &= \underbrace{A_2^{(0)}(\theta)^\top a_1 + b_1}_{b_0},\end{aligned}$$

where in the last line we used that $u^{(0)}(\theta)$ is constant.

Comparison with state-of-the-art

State-of-the-art: joint optimization [Kang et al. (2011), Kshirsagar et al. (2017).]

$$(\hat{w}, \hat{\theta}) = \operatorname{argmin}_{w, \theta \in \Theta} \left\{ \mathcal{L}(w; \theta) \triangleq \ell(y, \langle X, w \rangle) + \sum_{l=1}^L \rho_l(\theta_l \odot w) \right\}$$

issues: some trivial undesired minima
unclear interpretation of the solution

Proposed method: bilevel optimization [Frécon et al. (2018), Frécon et al. (2020).]

$$\min_{\theta \in \Theta} \mathcal{E}(\hat{w}(\theta)) \quad \text{s.t.} \quad \hat{w}(\theta) = \operatorname{argmin}_w \mathcal{L}(w; \theta)$$

idea: find θ such that $\hat{w}(\theta)$ generalizes well to unseen data
→ choose \mathcal{E} as the validation error