Implicit differentiation for fast hyperparameter selection in non-smooth convex learning

Quentin Bertrand (Inria)
https://QB3.github.io

Joint work with:
Quentin Klopfenstein (Univ. Bourgogne Franche-Comté)
Mathurin Massias (Univ. Genova)
Mathieu Blondel (Google)
Samuel Vaiter (CNRS)
Alexandre Gramfort (Inria)
Joseph Salmon (IMAG, Univ. Montpellier, CNRS)

Motivation

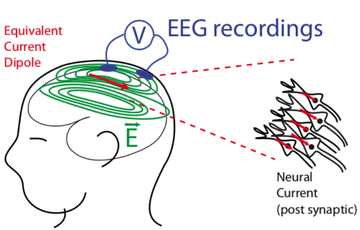
Hyperparameter optimization

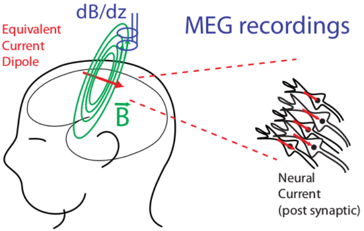
Hypergradient computation

Experiments

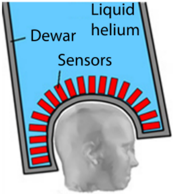
M/EEG inverse problem for brain imaging

- sensors: electric and magnetic fields during a cognitive task
- goal: which parts of the brain are responsible for the signals?
- applications: epilepsy treatment, brain aging, anesthesia risks



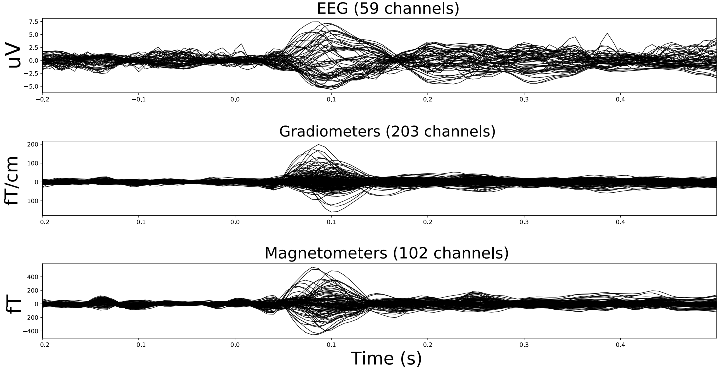


First EEG
recordings
in 1929
by H. Berger



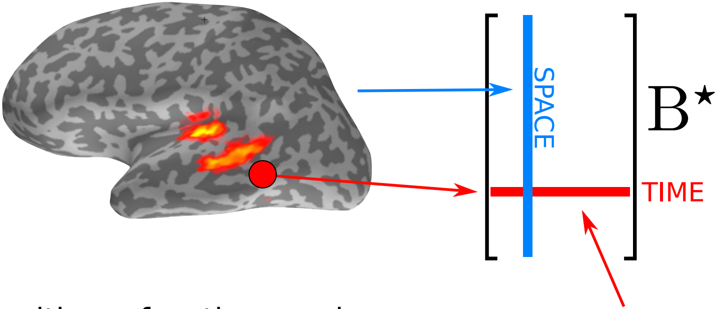
Hôpital La Timone
Marseille, France

M/EEG data

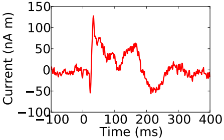


- 3 different types of sensor

Source modeling (discretization with voxels)

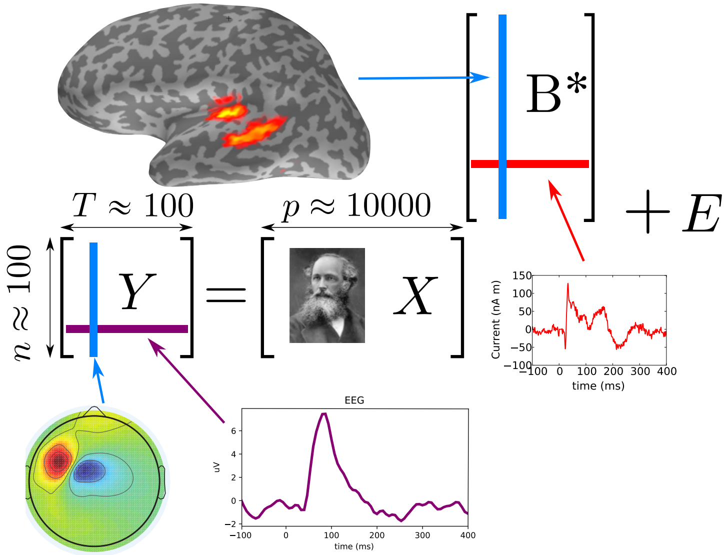


Position a few thousands candidate sources over the brain (*e.g.*, every 5mm)



$$\mathrm{B}^\star \in \mathbb{R}^{p \times T}$$

The M/EEG inverse problem: modeling

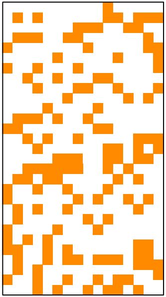


$$n \ll p$$

Multi-Task penalties[1]

Popular convex penalties considered:

$$\hat{\mathrm{B}} \in \underset{\mathrm{B} \in \mathbb{R}^{p \times T}}{\arg\min} \left(\frac{1}{2nT} \|Y - X\mathrm{B}\|_F^2 + \lambda \Omega(\mathrm{B}) \right)$$



Sparse support: no structure

Penalty: **Lasso type**

$$\Omega(\mathrm{B}) = \|\mathrm{B}\|_1 = \sum_{j=1}^{p} \sum_{k=1}^{T} |\mathrm{B}_{j,k}|$$

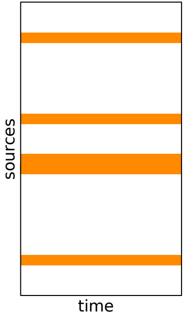
Parameter $\hat{\mathrm{B}} \in \mathbb{R}^{p \times T}$

[1] G. Obozinski, B. Taskar, and M. I. Jordan. "Joint covariate selection and joint subspace selection for multiple classification problems". In: *Statistics and Computing* 20.2 (2010), pp. 231–252.

Multi-Task penalties[1]

Popular convex penalties considered: Multi-Task Lasso (MTL)

$$\hat{\mathrm{B}} \in \underset{\mathrm{B} \in \mathbb{R}^{p \times T}}{\arg\min} \left(\frac{1}{2nT} \| Y - X\mathrm{B} \|_F^2 + \lambda \Omega(\mathrm{B}) \right)$$



Parameter $\hat{\mathrm{B}} \in \mathbb{R}^{p \times T}$

Sparse support: group structure ✓

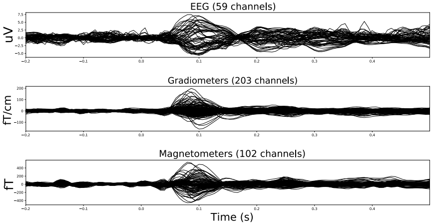
Penalty: **Group-Lasso type**

$$\Omega(\mathrm{B}) = \|\mathrm{B}\|_{2,1} = \sum_{j=1}^{p} \|\mathrm{B}_{j,:}\|_2$$

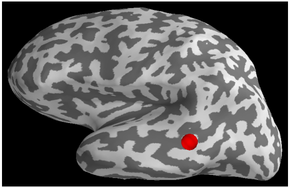
where $\mathrm{B}_{j,:}$ the j-th row of B

[1] G. Obozinski, B. Taskar, and M. I. Jordan. "Joint covariate selection and joint subspace selection for multiple classification problems". In: *Statistics and Computing* 20.2 (2010), pp. 231–252.

Summary



What you have: Y



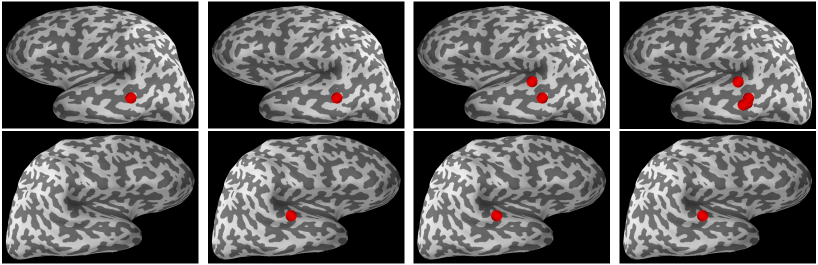
What you want B

This is typically done using optimization based estimators:

$$\hat{\mathrm{B}} \in \mathop{\arg\min}_{\mathrm{B} \in \mathbb{R}^{p \times T}} \left(\frac{1}{2nT} \| Y - X\mathrm{B} \|_F^2 + \lambda \Omega(\mathrm{B}) \right)$$

Which λ to pick?

$$\hat{\mathrm{B}} \in \underset{\mathrm{B} \in \mathbb{R}^{p \times T}}{\arg\min} \left(\frac{1}{2nT} \|Y - X\mathrm{B}\|_F^2 + \lambda \|\mathrm{B}\|_{2,1} \right)$$



$\lambda = 0.85\lambda_{\max}$ $\lambda = 0.82\lambda_{\max}$ $\lambda = 0.80\lambda_{\max}$ $\lambda = 0.75\lambda_{\max}$

Real MEEG data. Brain source reconstruction using multitask Lasso with multiple λ. Which λ to pick? How to *automatically* select λ?

- When $\lambda \geq \lambda_{\max}$, $\hat{\mathrm{B}} = 0$ no sources are recovered

Which λ to pick? A statistical persective[2] (i.i.d. case, Single-Task, $y = X\beta + \sigma^* \varepsilon$)

$$\hat{\beta} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$$

Theorem

- i.i.d. Gaussian noise
- $+$ X satisfying the "Restricted Eigenvalue" property
- $+$ $\lambda = 2\sigma_* \sqrt{\frac{2\log(p/\delta)}{n}}$
- $\Longrightarrow$ with probability $1 - \delta$:

$$\frac{1}{n}\|X\beta^* - X\hat{\beta}^{(\lambda)}\|^2 \leq \frac{18}{\kappa_{s^*}^2} \frac{\sigma_*^2 s^*}{n} \log\left(\frac{p}{\delta}\right)$$

BUT σ_* is unknown in practice !

[2] P. J. Bickel, Y. Ritov, and A. B. Tsybakov. "Simultaneous analysis of Lasso and Dantzig selector". In: *Ann. Statist.* 37.4 (2009), pp. 1705–1732.

Which λ to pick? A statistical persective[2] (i.i.d. case, Single-Task, $y = X\beta + \sigma^* \varepsilon$)

$$\hat{\beta} \in \mathop{\arg\min}_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$$

Theorem

- i.i.d. Gaussian noise
- $+\ X$ satisfying the "Restricted Eigenvalue" property
- $+\ \lambda = 2\sigma_* \sqrt{\frac{2\log(p/\delta)}{n}}$
- $\Longrightarrow$ with probability $1 - \delta$:

$$\frac{1}{n} \|X\beta^* - X\hat{\beta}^{(\lambda)}\|^2 \leq \frac{18}{\kappa_{s^*}^2} \frac{\sigma_*^2 s^*}{n} \log\left(\frac{p}{\delta}\right)$$

BUT σ_* is unknown in practice !

[2] P. J. Bickel, Y. Ritov, and A. B. Tsybakov. "Simultaneous analysis of Lasso and Dantzig selector". In: *Ann. Statist.* 37.4 (2009), pp. 1705–1732.

Which λ to pick? A statistical persective II[3] (i.i.d. case, Single-Task)

$$\hat{\beta} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{\sqrt{n}} \|y - X\beta\|_2 + \lambda \|\beta\|_1$$

Theorem

- i.i.d. Gaussian noise
- $+\ X$ satisfying the "Restricted Eigenvalue" property
- $+\ \lambda = 2\sqrt{\frac{2\log(p/\delta)}{n}}$
- $\implies$ with probability $1 - \delta$:

$$\frac{1}{n}\|X\beta^* - X\hat{\beta}^{(\lambda)}\|^2 \leq \frac{18}{\kappa_{s^*}^2} \frac{\sigma_*^2 s^*}{n} \log\left(\frac{p}{\delta}\right)$$

λ does not depend on σ_* anymore!

[3] A. Belloni, V. Chernozhukov, and L. Wang. "Square-root Lasso: pivotal recovery of sparse signals via conic programming". In: *Biometrika* 98.4 (2011), pp. 791–806.

Which λ to pick? A statistical persective II[3] (i.i.d. case, Single-Task)

$$\hat{\beta} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{\sqrt{n}} \|y - X\beta\|_2 + \lambda \|\beta\|_1$$

Theorem

- i.i.d. Gaussian noise
- $+\ X$ satisfying the "Restricted Eigenvalue" property
- $+\ \lambda = 2\sqrt{\frac{2\log(p/\delta)}{n}}$
- $\implies$ with probability $1-\delta$:

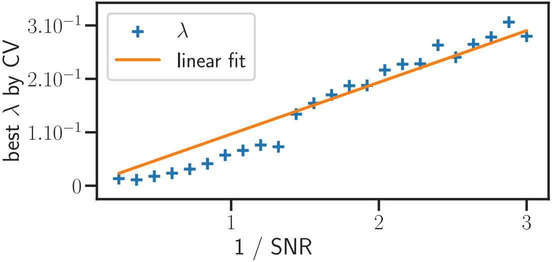
$$\frac{1}{n}\|X\beta^* - X\hat{\beta}^{(\lambda)}\|^2 \leq \frac{18}{\kappa_{s^*}^2} \frac{\sigma_*^2 s^*}{n} \log\left(\frac{p}{\delta}\right)$$

λ does not depend on σ_* anymore!

[3] A. Belloni, V. Chernozhukov, and L. Wang. "Square-root Lasso: pivotal recovery of sparse signals via conic programming". In: *Biometrika* 98.4 (2011), pp. 791–806.

Which λ to pick? A statistical persective III

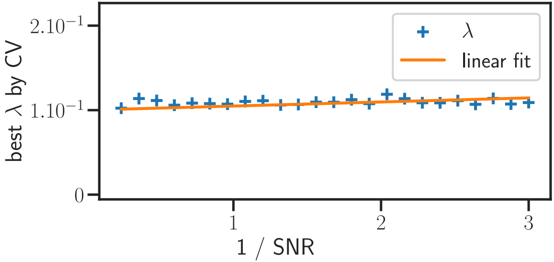
$$\boxed{\hat{\beta}_{\text{Lasso}} \in \arg\min_{\beta \in \mathbb{R}^p} \left(\frac{1}{2n} \|y - X\beta\|^2 + \lambda \|\beta\|_1 \right)}$$



Lasso

Which λ to pick? A statistical persective III

$$\boxed{\hat{\beta}_{\sqrt{\text{Lasso}}} \in \mathop{\arg\min}_{\beta \in \mathbb{R}^p} \left(\frac{1}{\sqrt{n}} \|y - X\beta\| + \lambda \|\beta\|_1 \right)}$$



Square-root Lasso

Which λ to pick? A statistical persective III

- $\lambda \sim \sigma^*$ and λ independent of σ^* confirmed in practice ✓
- Strong statistical assumptions, not verified in practice ✗
- Still unknown quantities in the closed-form formula for λ: still needs calibration in practice ✗

Hyperparameter optimization (HO)

Possible selection criterion:

- Good generalization[4] of $\hat{\beta}^{(\lambda)}$
- AIC/BIC,[5] SURE[6] that control model complexity

---

[4] L. R. A. Stone and J.C. Ramer. "Estimating WAIS IQ from Shipley Scale scores: Another cross-validation". In: *Journal of clinical psychology* 21.3 (1965), pp. 297–297.

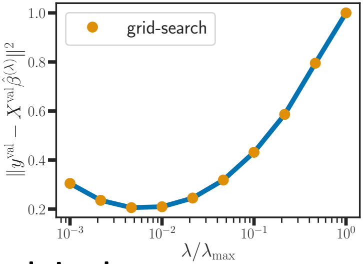
[5] W. Liu, Y. Yang, et al. "Parametric or nonparametric? A parametricness index for model selection". In: *Ann. Statist.* 39.4 (2011), pp. 2074–2102.

[6] C. M. Stein. "Estimation of the mean of a multivariate normal distribution". In: *Ann. Statist.* 9.6 (1981), pp. 1135–1151.

Hyperparameter optimization (HO)

Possible selection criterion:

- Good generalization[4] of $\hat{\beta}^{(\lambda)}$
- AIC/BIC,[5] SURE[6] that control model complexity



Real-sim dataset

Validation loss as a function of λ.

Example

Model: Lasso

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{\|y^{\text{train}} - X^{\text{train}}\beta\|^2}{2n} + \lambda\|\beta\|_1$$

Criterion: held-out loss

$$\arg\min_{\lambda} \|y^{\text{val}} - X^{\text{val}}\hat{\beta}^{(\lambda)}\|^2$$

---

(4) L. R. A. Stone and J.C. Ramer. "Estimating WAIS IQ from Shipley Scale scores: Another cross-validation". In: *Journal of clinical psychology* 21.3 (1965), pp. 297–297.

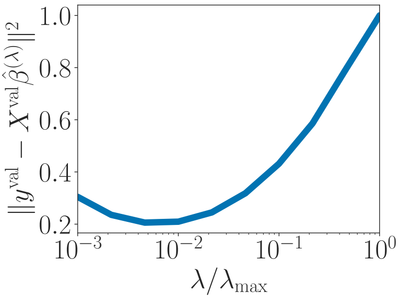
(5) W. Liu, Y. Yang, et al. "Parametric or nonparametric? A parametricness index for model selection". In: *Ann. Statist.* 39.4 (2011), pp. 2074–2102.

(6) C. M. Stein. "Estimation of the mean of a multivariate normal distribution". In: *Ann. Statist.* 9.6 (1981), pp. 1135–1151.

HO as a bilevel optimization problem(7)(8)

$$\underset{\lambda \in \mathbb{R}}{\arg\min} \overbrace{\left\{ \mathcal{L}(\lambda) := \| y^{\text{val}} - X^{\text{val}} \hat{\beta}^{(\lambda)} \|^2 \right\}}^{\text{outer optimization problem}}$$

$$\text{s.t. } \underbrace{\hat{\beta}^{(\lambda)} \in \underset{\beta \in \mathbb{R}^p}{\arg\min} \frac{1}{2n} \| y^{\text{train}} - X^{\text{train}} \beta \|^2 + \lambda \|\beta\|_1}_{\text{inner optimization problem}}$$



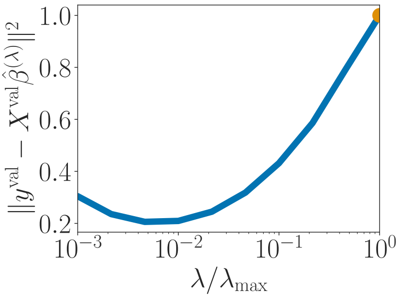
(7) P. Ochs et al. "Bilevel optimization with nonsmooth lower level problems". In: *SSVM*. 2015, pp. 654–665.

(8) F. Pedregosa. "Hyperparameter optimization with approximate gradient". In: *ICML*. vol. 48. 2016, pp. 737–746.

HO as a bilevel optimization problem[7][8]

$$\underset{\lambda \in \mathbb{R}}{\arg\min} \overbrace{\left\{ \mathcal{L}(\lambda) := \| y^{\text{val}} - X^{\text{val}} \hat{\beta}^{(\lambda)} \|^2 \right\}}^{\text{outer optimization problem}}$$

$$\text{s.t. } \underbrace{\hat{\beta}^{(\lambda)} \in \underset{\beta \in \mathbb{R}^p}{\arg\min} \frac{1}{2n} \| y^{\text{train}} - X^{\text{train}} \beta \|^2 + \lambda \|\beta\|_1}_{\text{inner optimization problem}}$$



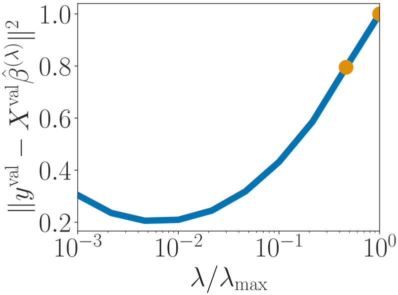
[7] P. Ochs et al. "Bilevel optimization with nonsmooth lower level problems". In: *SSVM*. 2015, pp. 654–665.

[8] F. Pedregosa. "Hyperparameter optimization with approximate gradient". In: *ICML*. vol. 48. 2016, pp. 737–746.

HO as a bilevel optimization problem[7][8]

$$\underbrace{\vphantom{X}}_{}\overbrace{\arg\min_{\lambda \in \mathbb{R}} \left\{ \mathcal{L}(\lambda) := \| y^{\text{val}} - X^{\text{val}} \hat{\beta}^{(\lambda)} \|^2 \right\}}^{\text{outer optimization problem}}$$

$$\text{s.t. } \underbrace{\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \| y^{\text{train}} - X^{\text{train}} \beta \|^2 + \lambda \|\beta\|_1}_{\text{inner optimization problem}}$$



---

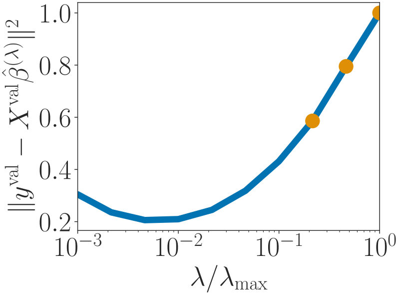
[7] P. Ochs et al. "Bilevel optimization with nonsmooth lower level problems". In: *SSVM*. 2015, pp. 654–665.

[8] F. Pedregosa. "Hyperparameter optimization with approximate gradient". In: *ICML*. vol. 48. 2016, pp. 737–746.

HO as a bilevel optimization problem[7][8]

$$\overbrace{\arg\min_{\lambda \in \mathbb{R}} \left\{ \mathcal{L}(\lambda) := \|y^{\text{val}} - X^{\text{val}} \hat{\beta}^{(\lambda)}\|^2 \right\}}^{\text{outer optimization problem}}$$

$$\text{s.t. } \underbrace{\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\text{train}} - X^{\text{train}} \beta\|^2 + \lambda \|\beta\|_1}_{\text{inner optimization problem}}$$



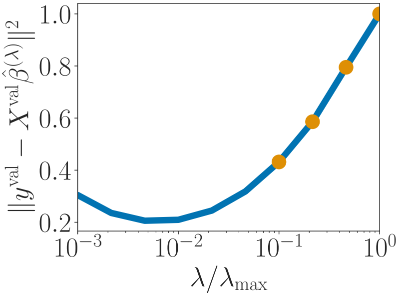
(7) P. Ochs et al. "Bilevel optimization with nonsmooth lower level problems". In: *SSVM*. 2015, pp. 654–665.

(8) F. Pedregosa. "Hyperparameter optimization with approximate gradient". In: *ICML*. vol. 48. 2016, pp. 737–746.

HO as a bilevel optimization problem[7][8]

$$\underset{\lambda\in\mathbb{R}}{\arg\min} \overbrace{\left\{\mathcal{L}(\lambda) := \|y^{\text{val}} - X^{\text{val}}\hat{\beta}^{(\lambda)}\|^2\right\}}^{\text{outer optimization problem}}$$

$$\text{s.t. } \underbrace{\hat{\beta}^{(\lambda)} \in \underset{\beta\in\mathbb{R}^p}{\arg\min} \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda\|\beta\|_1}_{\text{inner optimization problem}}$$



---

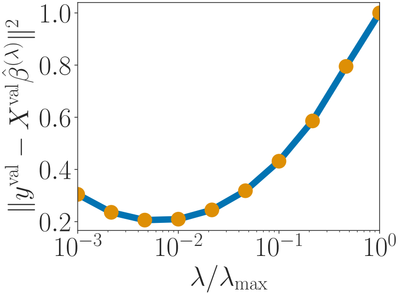
[7] P. Ochs et al. "Bilevel optimization with nonsmooth lower level problems". In: *SSVM*. 2015, pp. 654–665.

[8] F. Pedregosa. "Hyperparameter optimization with approximate gradient". In: *ICML*. vol. 48. 2016, pp. 737–746.

HO as a bilevel optimization problem[7][8]

$$\underset{\lambda \in \mathbb{R}}{\arg\min} \overbrace{\left\{ \mathcal{L}(\lambda) := \|y^{\text{val}} - X^{\text{val}} \hat{\beta}^{(\lambda)}\|^2 \right\}}^{\text{outer optimization problem}}$$

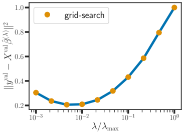
$$\text{s.t. } \underbrace{\hat{\beta}^{(\lambda)} \in \underset{\beta \in \mathbb{R}^p}{\arg\min} \frac{1}{2n} \|y^{\text{train}} - X^{\text{train}} \beta\|^2 + \lambda \|\beta\|_1}_{\text{inner optimization problem}}$$



[7] P. Ochs et al. "Bilevel optimization with nonsmooth lower level problems". In: *SSVM*. 2015, pp. 654–665.

[8] F. Pedregosa. "Hyperparameter optimization with approximate gradient". In: *ICML*. vol. 48. 2016, pp. 737–746.

Grid-search as a 0-order optimization method



$$\arg\min_{\lambda \in \mathbb{R}} \left\{ \mathcal{L}(\lambda) := \|y^{\text{val}} - X^{\text{val}} \hat{\beta}^{(\lambda)}\|^2 \right\}$$

$$\text{s.t. } \hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\text{train}} - X^{\text{train}} \beta\|^2 + \lambda \|\beta\|_1$$

- Grid-search, random-search,[9] SMBO[10]: 0-order methods to solve bilevel optimization problem
- **Idea:** if $\mathcal{L}$ is differentiable, use first-order optimization, *i.e.*, compute $\nabla_\lambda \mathcal{L}$
- Once $\nabla_\lambda \mathcal{L}(\lambda)$ is computed, use gradient descent[11]:
$$\lambda^{(t+1)} = \lambda^{(t)} - \rho \nabla_\lambda \mathcal{L}(\lambda^{(t)}) \quad \text{with suitable } \rho > 0$$

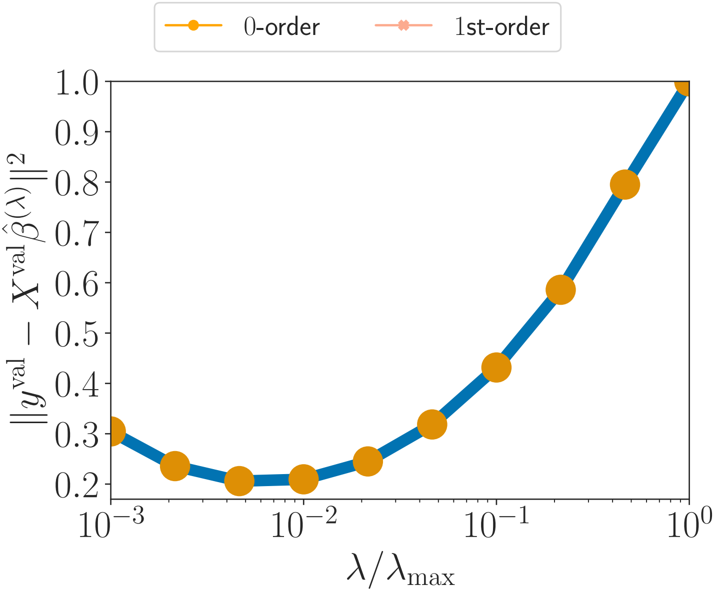
---

[9] J. Bergstra and Y. Bengio. "Random search for hyper-parameter optimization". In: *Journal of Machine Learning Research* 13.2 (2012).

[10] E. Brochu, V. M. Cora, and N. De Freitas. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning". In: (2010).

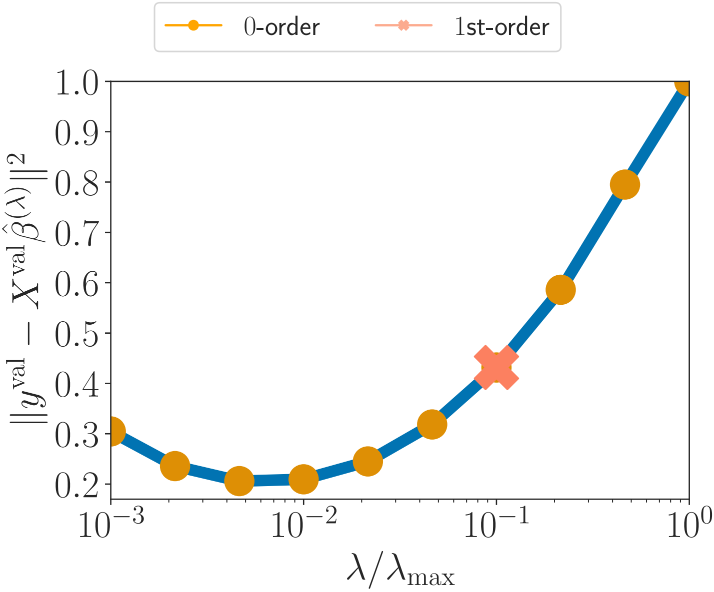
[11] F. Pedregosa. "Hyperparameter optimization with approximate gradient". In: *ICML*. vol. 48. 2016, pp. 737–746.

First-order optimization in λ, Lasso



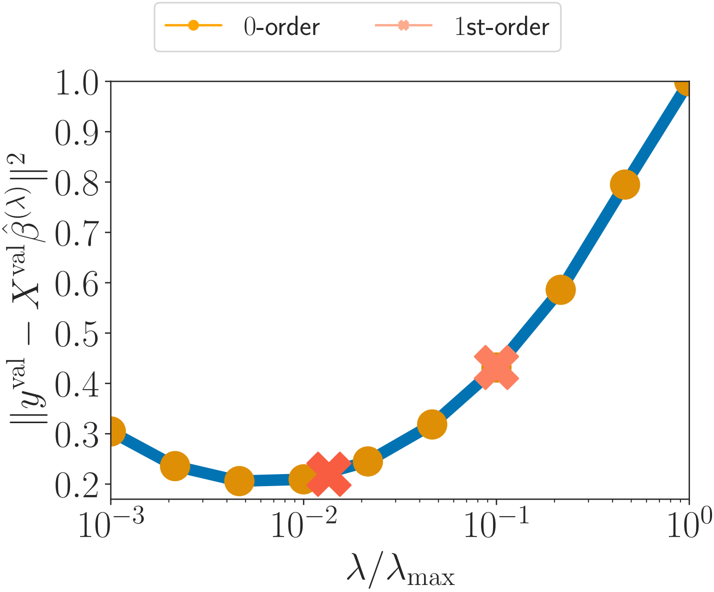
Real-sim dataset. Validation loss as a function of λ.

First-order optimization in λ, Lasso



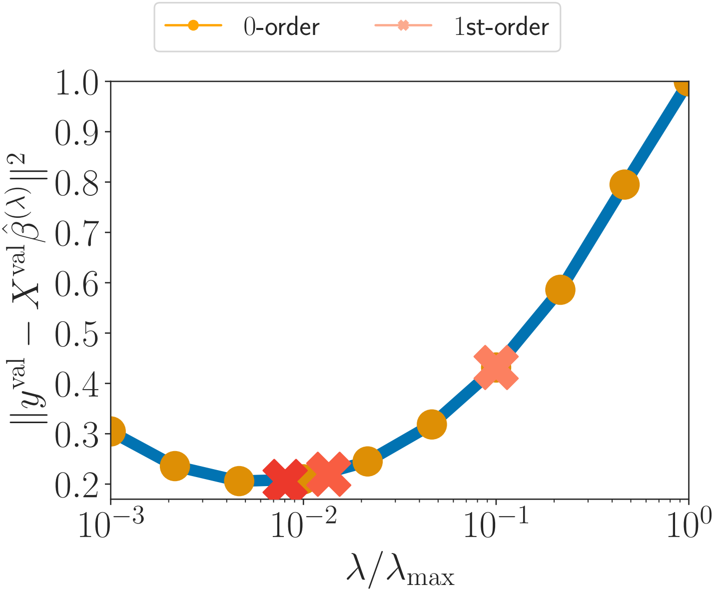
Real-sim dataset. Validation loss as a function of λ.

First-order optimization in λ, Lasso



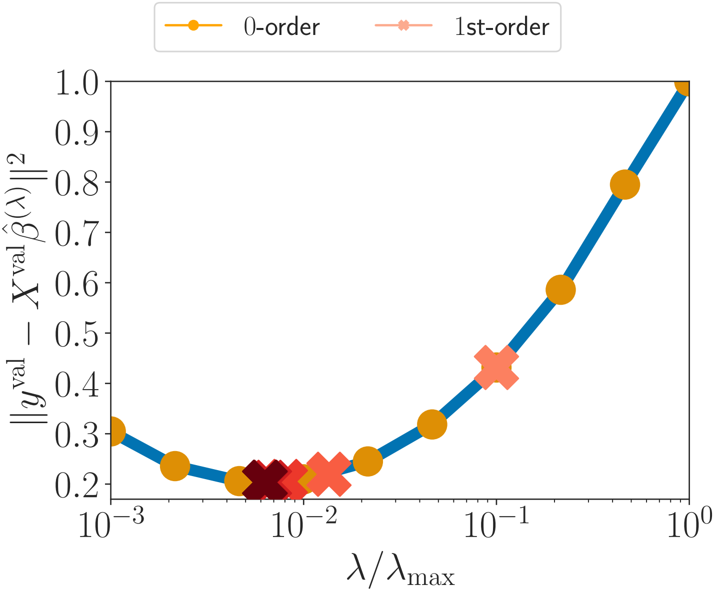
Real-sim dataset. Validation loss as a function of λ.

First-order optimization in λ, Lasso



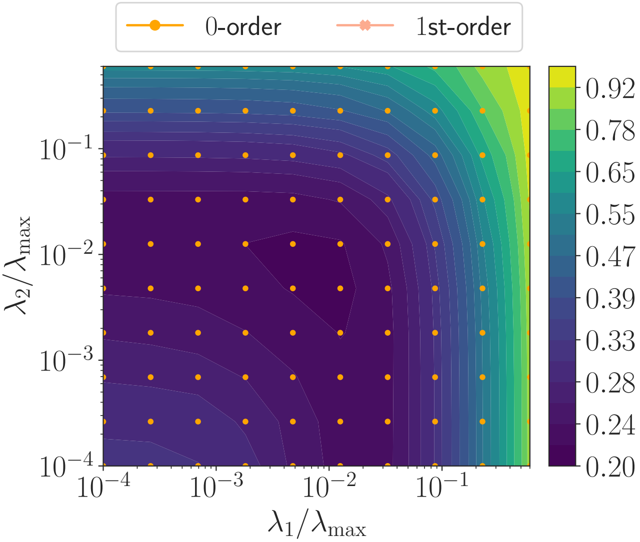
Real-sim dataset. Validation loss as a function of λ.

First-order optimization in λ, Lasso



Real-sim dataset. Validation loss as a function of λ.

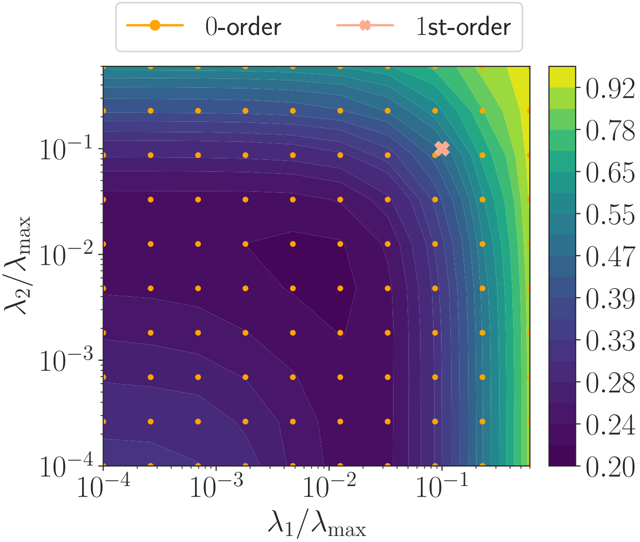
First-order optimization in λ, Enet



Real-sim dataset, level sets of the validation loss (hold-out)

$$\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$$

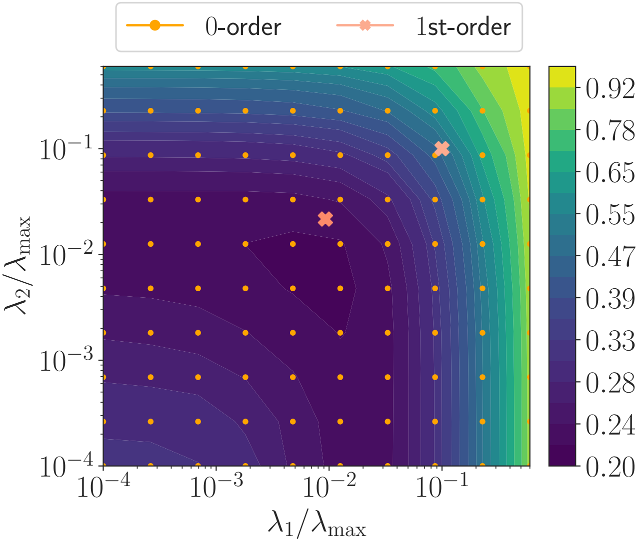
First-order optimization in λ, Enet



Real-sim dataset, level sets of the validation loss (hold-out)

$$\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$$

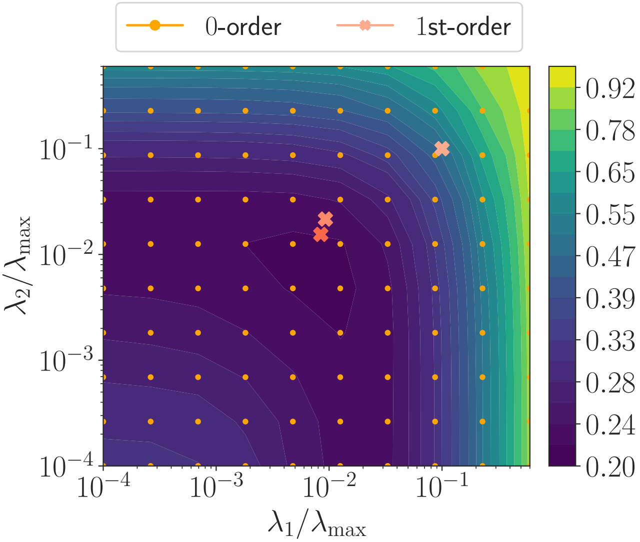
First-order optimization in λ, Enet



Real-sim dataset, level sets of the validation loss (hold-out)

$$\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$$

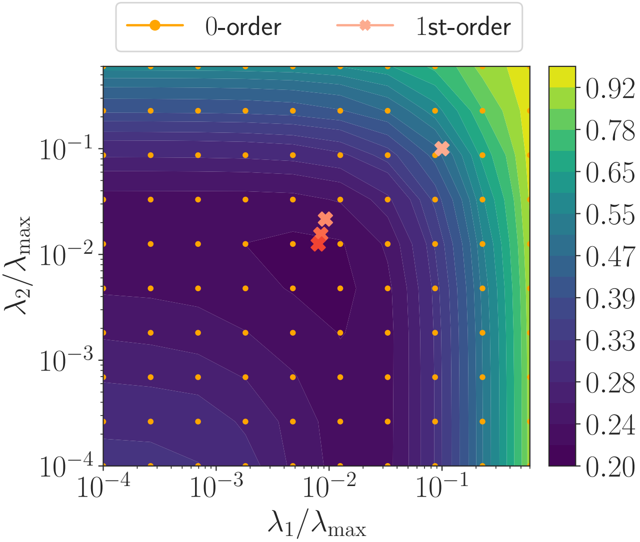
First-order optimization in λ, Enet



Real-sim dataset, level sets of the validation loss (hold-out)

$$\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$$

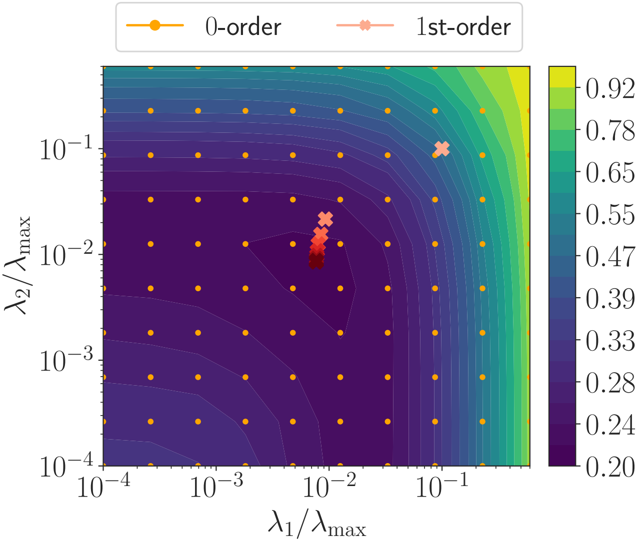
First-order optimization in λ, Enet



Real-sim dataset, level sets of the validation loss (hold-out)

$$\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$$

First-order optimization in λ, Enet



Real-sim dataset, level sets of the validation loss (hold-out)

$$\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$$

What's hard? Computing $\nabla_\lambda \mathcal{L}(\lambda)$

$$\operatorname*{arg\,min}_{\lambda \in \mathbb{R}} \left\{ \mathcal{L}(\lambda) := C(\hat\beta^{(\lambda)}) := \| y^{\mathsf{val}} - X^{\mathsf{val}} \hat\beta^{(\lambda)} \|^2 \right\}$$

$$\text{s.t. } \hat\beta^{(\lambda)} \in \operatorname*{arg\,min}_{\beta \in \mathbb{R}^p} \frac{1}{2n} \| y^{\mathsf{train}} - X^{\mathsf{train}} \beta \|^2 + \lambda \|\beta\|_1$$

Once $\nabla_\lambda \mathcal{L}(\lambda)$ is computed, let's pretend "life is easy":

- Line-search[12]
- LBFGS[13]
- Gradient descent

[12] J. Nocedal and S. J. Wright. *Numerical optimization*. Second. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006, Chap. 3.

[13] D. C. Liu and J. Nocedal. "On the limited memory BFGS method for large scale optimization". In: *Mathematical programming* 45.1-3 (1989), pp. 503–528.

What's hard? Computing $\nabla_\lambda \mathcal{L}(\lambda)$

$$\arg\min_{\lambda \in \mathbb{R}} \left\{ \mathcal{L}(\lambda) := C(\hat\beta^{(\lambda)}) := \|y^{\mathsf{val}} - X^{\mathsf{val}} \hat\beta^{(\lambda)}\|^2 \right\}$$

$$\text{s.t. } \hat\beta^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\mathsf{train}} - X^{\mathsf{train}} \beta\|^2 + \lambda \|\beta\|_1$$

Once $\nabla_\lambda \mathcal{L}(\lambda)$ is computed, let's pretend "life is easy":

- Line-search[12]
- LBFGS[13]
- Gradient descent

Main challenge: compute $\nabla_\lambda \mathcal{L}(\lambda)$ for a given λ

---

[12] J. Nocedal and S. J. Wright. *Numerical optimization*. Second. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006, Chap. 3.

[13] D. C. Liu and J. Nocedal. "On the limited memory BFGS method for large scale optimization". In: *Mathematical programming* 45.1-3 (1989), pp. 503–528.

What's hard? Computing $\nabla_\lambda \mathcal{L}(\lambda)$

$$\begin{aligned}\arg\min_{\lambda \in \mathbb{R}} &\left\{ \mathcal{L}(\lambda) := C(\hat{\beta}^{(\lambda)}) := \|y^{\mathsf{val}} - X^{\mathsf{val}} \hat{\beta}^{(\lambda)}\|^2 \right\} \\ \text{s.t. } \hat{\beta}^{(\lambda)} &\in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\mathsf{train}} - X^{\mathsf{train}} \beta\|^2 + \lambda \|\beta\|_1\end{aligned}$$

Once $\nabla_\lambda \mathcal{L}(\lambda)$ is computed, let's pretend "life is easy":

- Line-search[12]
- LBFGS[13]
- Gradient descent

Main challenge: compute $\nabla_\lambda \mathcal{L}(\lambda)$ for a given λ

---

(12) J. Nocedal and S. J. Wright. *Numerical optimization*. Second. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006, Chap. 3.

(13) D. C. Liu and J. Nocedal. "On the limited memory BFGS method for large scale optimization". In: *Mathematical programming* 45.1-3 (1989), pp. 503–528.

How to compute $\nabla_\lambda \mathcal{L}(\lambda)$?

$$\arg\min_{\lambda \in \mathbb{R}} \left\{ \mathcal{L}(\lambda) := C(\hat{\beta}^{(\lambda)}) := \|y^{\mathsf{val}} - X^{\mathsf{val}} \hat{\beta}^{(\lambda)}\|^2 \right\}$$

$$\text{s.t. } \hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\mathsf{train}} - X^{\mathsf{train}} \beta\|^2 + \lambda \|\beta\|_1$$

Chain rule and Jacobian:

$$\nabla_\lambda \mathcal{L}(\lambda) = \underbrace{\hat{\mathcal{J}}_{(\lambda)}^\top}_{\substack{:=(\nabla_\lambda \hat{\beta}_1^{(\lambda)}, \dots, \nabla_\lambda \hat{\beta}_p^{(\lambda)}) \\ \to \text{main challenge}}} \nabla_\beta C(\hat{\beta}^{(\lambda)})$$

- Boils down to:

how to compute the Jacobian $\hat{\mathcal{J}}_{(\lambda)}$ efficiently?

How to compute $\nabla_\lambda \mathcal{L}(\lambda)$?

$$\arg\min_{\lambda \in \mathbb{R}} \left\{ \mathcal{L}(\lambda) := C(\hat{\beta}^{(\lambda)}) := \|y^{\mathsf{val}} - X^{\mathsf{val}} \hat{\beta}^{(\lambda)}\|^2 \right\}$$

$$\text{s.t. } \hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\mathsf{train}} - X^{\mathsf{train}} \beta\|^2 + \lambda \|\beta\|_1$$

Chain rule and Jacobian:

$$\nabla_\lambda \mathcal{L}(\lambda) = \underbrace{\hat{\mathcal{J}}_{(\lambda)}^\top}_{\substack{:=(\nabla_\lambda \hat{\beta}_1^{(\lambda)}, \dots, \nabla_\lambda \hat{\beta}_p^{(\lambda)}) \\ \to \text{main challenge}}} \nabla_\beta C(\hat{\beta}^{(\lambda)})$$

- Boils down to:

  how to compute the Jacobian $\hat{\mathcal{J}}_{(\lambda)}$ efficiently?

How to compute $\hat{\mathcal{J}}_{(\lambda)} := (\nabla_\lambda \hat{\beta}_1^{(\lambda)}, \dots, \nabla_\lambda \hat{\beta}_p^{(\lambda)})^\top$?

$$\underbrace{\hat{\beta}^{(\lambda)} \in \operatorname*{arg\,min}_{\beta \in \mathbb{R}^p} \frac{1}{2n} \| y^{\text{train}} - X^{\text{train}} \beta \|^2 + \frac{\lambda}{2} \|\beta\|^2}_{\text{inner optimization problem}}$$

"Smooth" inner optimization problems, **well studied**:

- *Implicit differentiation* (**closed-form** formula)[14]: need to solve a $p \times p$ linear system ($p = \#$features)
- *Automatic differentiation*, *forward*[15] or *reverse*[16] mode

(14) J. Larsen et al. "Design and regularization of neural networks: the optimal use of a validation set". In: *Neural Networks for Signal Processing VI. Proceedings of the 1996 IEEE Signal Processing Society Workshop*. 1996; Y. Bengio. "Gradient-based optimization of hyperparameters". In: *Neural computation* 12.8 (2000), pp. 1889–1900.

(15) L. Franceschi et al. "Forward and reverse gradient-based hyperparameter optimization". In: *ICML*. 2017, pp. 1165–1173.

(16) J. Domke. "Generic methods for optimization-based modeling". In: *AISTATS*. vol. 22. 2012, pp. 318–326.

How to compute $\hat{\mathcal{J}}_{(\lambda)} := (\nabla_\lambda \hat{\beta}_1^{(\lambda)}, \dots, \nabla_\lambda \hat{\beta}_p^{(\lambda)})^\top$?

$$\underbrace{\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\text{train}} - X^{\text{train}} \beta\|^2 + \lambda \|\beta\|_1}_{\text{inner optimization problem}}$$

"Nonsmooth" inner optimization problems, **scarce literature**:

- *Smooth the nonsmooth term*[17]
- Use algorithms with differentiable updates[18][19] (Bregman)

Our contributions:

- Iterative differentiation can be applied on proximal algorithms
- $\hat{\mathcal{J}}_{(\lambda)} := (\nabla_\lambda \hat{\beta}_1^{(\lambda)}, \dots, \nabla_\lambda \hat{\beta}_p^{(\lambda)})^\top$ shares $\hat{\beta}^{(\lambda)}$'s **sparsity pattern**

(17) G. Peyré and J. M. Fadili. "Learning analysis sparsity priors". In: *Sampta*. 2011.

(18) P. Ochs et al. "Bilevel optimization with nonsmooth lower level problems". In: *SSVM*. 2015, pp. 654–665.

(19) J. Frecon, S. Salzo, and M. Pontil. "Bilevel learning of the group lasso structure". In: *NeurIPS*. 2018, pp. 8301–8311.

How to compute $\hat{\mathcal{J}}_{(\lambda)} := (\nabla_\lambda \hat{\beta}_1^{(\lambda)}, \dots, \nabla_\lambda \hat{\beta}_p^{(\lambda)})^\top$?

$$\underbrace{\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \| y^{\text{train}} - X^{\text{train}} \beta \|^2 + \lambda \|\beta\|_1}_{\text{inner optimization problem}}$$

"Nonsmooth" inner optimization problems, **scarce literature**:

- *Smooth the nonsmooth term*[17]
- Use algorithms with differentiable updates[18][19] (Bregman)

Our contributions:

- Iterative differentiation can be applied on proximal algorithms
- $\hat{\mathcal{J}}_{(\lambda)} := (\nabla_\lambda \hat{\beta}_1^{(\lambda)}, \dots, \nabla_\lambda \hat{\beta}_p^{(\lambda)})^\top$ shares $\hat{\beta}^{(\lambda)}$'s **sparsity pattern**

[17] G. Peyré and J. M. Fadili. "Learning analysis sparsity priors". In: *Sampta*. 2011.

[18] P. Ochs et al. "Bilevel optimization with nonsmooth lower level problems". In: *SSVM*. 2015, pp. 654–665.

[19] J. Frecon, S. Salzo, and M. Pontil. "Bilevel learning of the group lasso structure". In: *NeurIPS*. 2018, pp. 8301–8311.

Forward-mode differentiation of PGD

$$\hat{\beta}^{(\lambda)} \in \underset{\beta \in \mathbb{R}^p}{\arg\min} \overbrace{f}^{\text{smooth}}(\beta) + \lambda \overbrace{g}^{\text{non-smooth}}(\beta) \tag{1}$$

Algorithm: Proximal gradient descent PGD

init : $\beta = 0_p$, , L

for $\text{iter} = 1, \ldots,$ **do**

$z \leftarrow \beta - \frac{1}{L} \nabla f(\beta)$ `// gradient step`

$\beta \leftarrow \text{prox}_{\lambda g / L}(z)$ `// proximal step`

return β

Forward-mode differentiation of PGD

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \overbrace{f}^{\text{smooth}}(\beta) + \lambda \overbrace{g}^{\text{non-smooth}}(\beta) \tag{1}$$

Algorithm: Forward-mode differentiation of PGD

init : $\beta = 0_p, \quad , L$

for $\text{iter} = 1, \dots,$ **do**

$z \leftarrow \beta - \frac{1}{L} \nabla f(\beta)$ `// gradient step`

$dz \leftarrow \left(\mathrm{Id}_p - \frac{1}{L} \nabla^2 f(\beta)\right) \mathcal{J}$ `// diff w.r.t. λ: chain rule`

$\beta \leftarrow \mathrm{prox}_{\lambda g / L}(z)$ `// proximal step`

return β

Forward-mode differentiation of PGD

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \overbrace{f}^{\text{smooth}}(\beta) + \lambda \overbrace{g}^{\text{non-smooth}}(\beta) \tag{1}$$

Algorithm: Forward-mode differentiation of PGD

init : $\beta = 0_p$, $\mathcal{J} = 0_p$, L

for iter $= 1, \dots,$ **do**

$z \leftarrow \beta - \frac{1}{L} \nabla f(\beta)$ `// gradient step`

$dz \leftarrow \left(\mathrm{Id}_p - \frac{1}{L} \nabla^2 f(\beta) \right) \mathcal{J}$ `// diff w.r.t. λ: chain rule`

$\beta \leftarrow \mathrm{prox}_{\lambda g / L}(z)$ `// proximal step`

$\mathcal{J} \leftarrow \partial_z \mathrm{prox}_{\lambda g / L}(z) dz$ `// diff w.r.t. λ: chain rule`

return β, $\mathcal{J}$

Forward-mode differentiation of PGD

$$\hat{\beta}^{(\lambda)} \in \operatorname*{arg\,min}_{\beta \in \mathbb{R}^p} \overbrace{f}^{\text{smooth}}(\beta) + \lambda \overbrace{g}^{\text{non-smooth}}(\beta) \qquad (1)$$

Algorithm: Forward-mode differentiation of PGD

init : $\beta = 0_p$, $\mathcal{J} = 0_p$, L

for iter $= 1, \ldots,$ **do**

- $z \leftarrow \beta - \frac{1}{L} \nabla f(\beta)$ `// gradient step`
- $dz \leftarrow \left(\mathrm{Id}_p - \frac{1}{L} \nabla^2 f(\beta)\right) \mathcal{J}$ `// diff w.r.t. λ: chain rule`
- $\beta \leftarrow \mathrm{prox}_{\lambda g / L}(z)$ `// proximal step`
- $\mathcal{J} \leftarrow \partial_z \, \mathrm{prox}_{\lambda g/L}(z) dz$ `// diff w.r.t. λ: chain rule`
  $\quad + \partial_\lambda \, \mathrm{prox}_{\lambda g/L}(z)$ `// do not forget this term!`

return β, $\mathcal{J}$

Forward-mode differentiation of PCD

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\text{train}} - X^{\text{train}} \beta\|^2 + \lambda \|\beta\|_1 \tag{2}$$

- Forward-mode differentiation can be applied on **proximal coordinate descent** (PCD)
- **Convergence** of the Jacobian sequence $\mathcal{J}$?

Forward-mode differentiation of PCD

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\mathsf{train}} - X^{\mathsf{train}} \beta\|^2 + \lambda \|\beta\|_1 \qquad (2)$$

- Forward-mode differentiation can be applied on **proximal coordinate descent** (PCD)
- **Convergence** of the Jacobian sequence $\mathcal{J}$?

Contribution

- Proved Jacobian sequence convergence for PGD and PCD

Forward-mode differentiation of PCD

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\text{train}} - X^{\text{train}} \beta\|^2 + \lambda \|\beta\|_1 \tag{2}$$

- Forward-mode differentiation can be applied on **proximal coordinate descent** (PCD)
- **Convergence** of the Jacobian sequence $\mathcal{J}$?

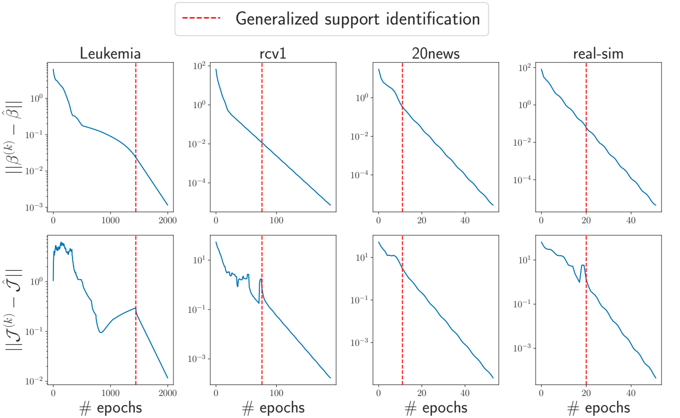
Contribution

- Proved Jacobian sequence convergence for PGD and PCD

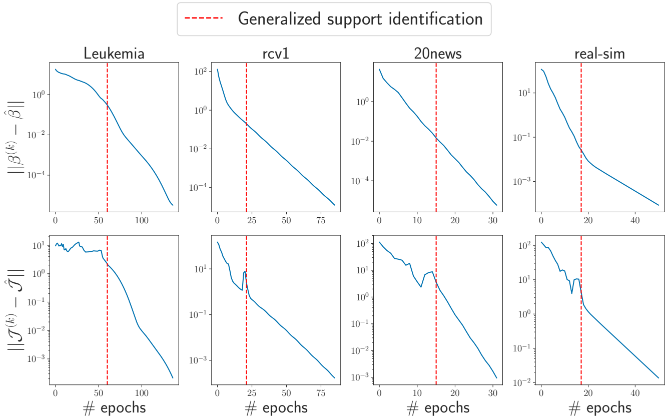
Local linear convergence of the Jacobian (I)

Proposition: forward diff. convergence (Lasso)

Assuming that the Lasso inner optimization has a unique minimizer, then the Jacobian sequence based on forward diff. of PCD converges to the true Jacobian. Once the support (*i.e.*, non-zeros coefs.) has been identified, convergence is linear.



Local linear convergence of the Jacobian (II)



Example: sparse logistic regression

$$\arg\min_{\beta \in \mathbb{R}^p} \frac{1}{n} \sum_i \log \frac{1}{1 + \exp(-y_i X_{i:} \beta)} + \lambda \|\beta\|_1$$

Implicit differentiation (smooth ψ)

$$\hat{\beta}^{(\lambda)} \in \underset{\beta \in \mathbb{R}^p}{\arg\min}\, \psi(\beta, \lambda)$$

$$\nabla_\beta \psi\left(\hat{\beta}^{(\lambda)}, \lambda\right) = 0$$

Implicit differentiation (smooth ψ)

$$\hat{\beta}^{(\lambda)} \in \underset{\beta \in \mathbb{R}^p}{\arg\min}\, \psi\left(\beta, \lambda\right)$$

$$\nabla_\beta \psi\left(\hat{\beta}^{(\lambda)}, \lambda\right) = 0$$

$$\nabla^2_{\beta,\lambda}\psi(\hat{\beta}^{(\lambda)}, \lambda) + \hat{\mathcal{J}}_{(\lambda)}^\top \nabla^2_\beta \psi(\hat{\beta}^{(\lambda)}, \lambda) = 0$$

Implicit differentiation (smooth ψ)

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \psi\left(\beta, \lambda\right)$$

$$\nabla_\beta \psi\left(\hat{\beta}^{(\lambda)}, \lambda\right) = 0$$

$$\nabla^2_{\beta,\lambda} \psi(\hat{\beta}^{(\lambda)}, \lambda) + \hat{\mathcal{J}}_{(\lambda)}^\top \nabla^2_\beta \psi(\hat{\beta}^{(\lambda)}, \lambda) = 0$$

$$\hat{\mathcal{J}}_{(\lambda)}^\top = -\nabla^2_{\beta,\lambda} \psi\left(\hat{\beta}^{(\lambda)}, \lambda\right) \underbrace{\left(\nabla^2_\beta \psi(\beta^{(\lambda)}, \lambda)\right)}_{p \times p}{}^{-1} \tag{3}$$

- Need to solve a linear **system of size** p

Implicit differentiation (smooth ψ)

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \psi\left(\beta, \lambda\right)$$

$$\nabla_\beta \psi\left(\hat{\beta}^{(\lambda)}, \lambda\right) = 0$$

$$\nabla^2_{\beta,\lambda}\psi(\hat{\beta}^{(\lambda)}, \lambda) + \hat{\mathcal{J}}^\top_{(\lambda)} \nabla^2_\beta \psi(\hat{\beta}^{(\lambda)}, \lambda) = 0$$

$$\hat{\mathcal{J}}^\top_{(\lambda)} = -\nabla^2_{\beta,\lambda}\psi\left(\hat{\beta}^{(\lambda)}, \lambda\right) \underbrace{\left(\nabla^2_\beta \psi(\beta^{(\lambda)}, \lambda)\right)^{-1}}_{p \times p} \tag{3}$$

- Need to solve a linear **system of size** p

Implicit differentiation ($f + \lambda \sum_j |\beta_j|$)

$$\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} f(\beta) + \lambda \sum_j |\beta_j|$$

$$\hat{\beta}^{(\lambda)} = \mathrm{ST}\left(\hat{\beta}^{(\lambda)} - \frac{1}{L} \nabla f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right)$$

Implicit differentiation ($f + \lambda \sum_j |\beta_j|$)

$$\hat{\beta}^{(\lambda)} \in \mathop{\arg\min}_{\beta \in \mathbb{R}^p} f(\beta) + \lambda \sum_j |\beta_j|$$

$$\hat{\beta}^{(\lambda)} = \mathrm{ST}\left(\hat{\beta}^{(\lambda)} - \frac{1}{L}\nabla f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right)$$

$$\hat{\mathcal{J}} = \partial_\beta \, \mathrm{ST}\left(\hat{\beta}^{(\lambda)} - \frac{1}{L}\nabla f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right)\left(\mathrm{Id} - \frac{\nabla^2 f}{L}\right)\hat{\mathcal{J}}$$
$$+ \partial_\lambda \, \mathrm{ST}\left(\hat{\beta}^{(\lambda)} - \frac{1}{L}\nabla f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right)$$

Implicit differentiation ($f + \lambda \sum_j |\beta_j|$)

$$\hat{\beta}^{(\lambda)} \in \mathop{\arg\min}_{\beta \in \mathbb{R}^p} f(\beta) + \lambda \sum_j |\beta_j|$$

$$\hat{\beta}^{(\lambda)} = \mathrm{ST}\left(\hat{\beta}^{(\lambda)} - \frac{1}{L}\nabla f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right)$$

$$\begin{aligned}\hat{\mathcal{J}} &= \partial_\beta \, \mathrm{ST}\left(\hat{\beta}^{(\lambda)} - \frac{1}{L}\nabla f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right)\left(\mathrm{Id} - \frac{\nabla^2 f}{L}\right)\hat{\mathcal{J}} \\ &+ \partial_\lambda \, \mathrm{ST}\left(\hat{\beta}^{(\lambda)} - \frac{1}{L}\nabla f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right)\end{aligned}$$

Key observation, if $\hat{\beta}_j^{(\lambda)} = 0$:

$$\partial_\beta \, \mathrm{ST}\left(\hat{\beta}_j^{(\lambda)} - \frac{1}{L}\nabla_j f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right) = 0 = \partial_\lambda \, \mathrm{ST}\left(\hat{\beta}_j^{(\lambda)} - \frac{1}{L}\nabla_j f(\hat{\beta}^{(\lambda)}), \frac{\lambda}{L}\right)$$

Implicit differentiation ($f + \lambda \sum_j |\beta_j|$)

$$\hat\beta^{(\lambda)} \in \operatorname*{arg\,min}_{\beta \in \mathbb{R}^p} f(\beta) + \lambda \sum_j |\beta_j|$$

$$\hat\beta^{(\lambda)} = \mathrm{ST}\left(\hat\beta^{(\lambda)} - \frac{1}{L}\nabla f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right)$$

$$\begin{aligned}\hat{\mathcal{J}} = {} & \partial_\beta \,\mathrm{ST}\left(\hat\beta^{(\lambda)} - \frac{1}{L}\nabla f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right)\left(\mathrm{Id} - \frac{\nabla^2 f}{L}\right)\hat{\mathcal{J}} \\ & + \partial_\lambda \,\mathrm{ST}\left(\hat\beta^{(\lambda)} - \frac{1}{L}\nabla f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right)\end{aligned}$$

Key observation, if $\hat\beta_j^{(\lambda)} = 0$:

$$\partial_\beta \,\mathrm{ST}\left(\hat\beta_j^{(\lambda)} - \frac{1}{L}\nabla_j f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right) = 0 = \partial_\lambda \,\mathrm{ST}\left(\hat\beta_j^{(\lambda)} - \frac{1}{L}\nabla_j f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right)$$

With $\mathcal{S} = \left\{j \in [p] \,:\, \hat\beta_j^{(\lambda)} = 0\right\}$ we have $\hat{\mathcal{J}}_{\mathcal{S}^c} = 0$

$$\hat{\mathcal{J}}_{\mathcal{S}} = \partial_\beta \,\mathrm{ST}(\hat\beta^{(\lambda)} - \tfrac{1}{L}\nabla f(\hat\beta^{(\lambda)}), \tfrac{\lambda}{L})_{\mathcal{S}} \hat{\mathcal{J}}_{\mathcal{S}} + \partial_\lambda \,\mathrm{ST}(\hat\beta_j^{(\lambda)} - \tfrac{1}{L}\nabla_j f(\hat\beta^{(\lambda)}), \tfrac{\lambda}{L})_{\mathcal{S}}$$

Implicit differentiation ($f + \lambda \sum_j |\beta_j|$)

$$\hat\beta^{(\lambda)} \in \operatorname*{arg\,min}_{\beta \in \mathbb{R}^p} f(\beta) + \lambda \sum_j |\beta_j|$$

$$\hat\beta^{(\lambda)} = \mathrm{ST}\left(\hat\beta^{(\lambda)} - \frac{1}{L}\nabla f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right)$$

$$\begin{aligned}\hat{\mathcal{J}} = {} & \partial_\beta \,\mathrm{ST}\left(\hat\beta^{(\lambda)} - \frac{1}{L}\nabla f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right)\left(\mathrm{Id} - \frac{\nabla^2 f}{L}\right)\hat{\mathcal{J}} \\ & + \partial_\lambda \,\mathrm{ST}\left(\hat\beta^{(\lambda)} - \frac{1}{L}\nabla f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right)\end{aligned}$$

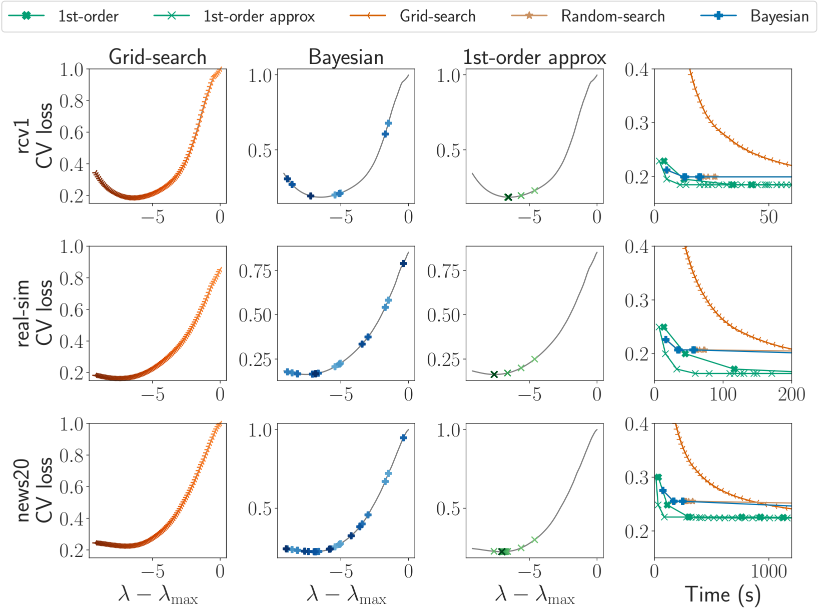
Key observation, if $\hat\beta_j^{(\lambda)} = 0$:

$$\partial_\beta \,\mathrm{ST}\left(\hat\beta_j^{(\lambda)} - \frac{1}{L}\nabla_j f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right) = 0 = \partial_\lambda \,\mathrm{ST}\left(\hat\beta_j^{(\lambda)} - \frac{1}{L}\nabla_j f(\hat\beta^{(\lambda)}), \frac{\lambda}{L}\right)$$

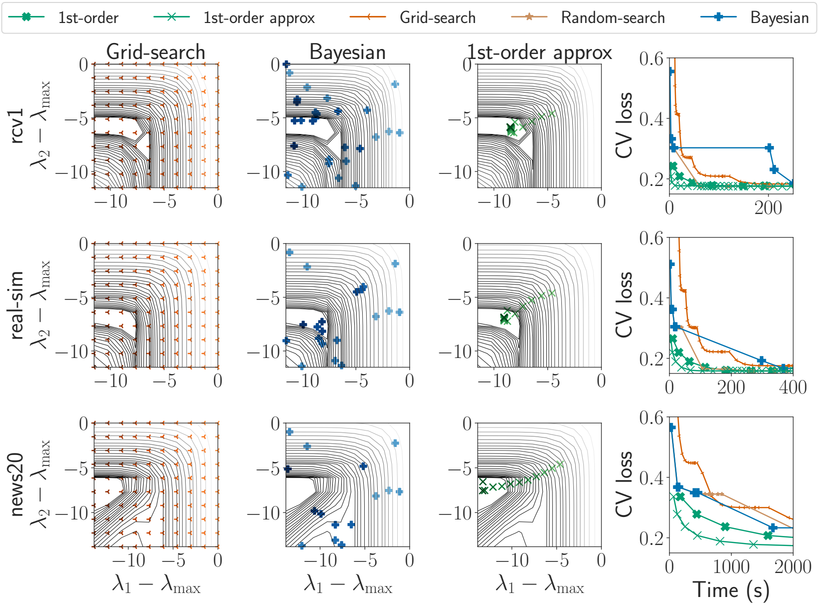
With $\mathcal{S} = \left\{ j \in [p] \,:\, \hat\beta_j^{(\lambda)} = 0 \right\}$ we have $\hat{\mathcal{J}}_{\mathcal{S}^c} = 0$

$$\hat{\mathcal{J}}_{\mathcal{S}} = \partial_\beta \,\mathrm{ST}(\hat\beta^{(\lambda)} - \tfrac{1}{L}\nabla f(\hat\beta^{(\lambda)}), \tfrac{\lambda}{L})_{\mathcal{S}} \hat{\mathcal{J}}_{\mathcal{S}} + \partial_\lambda \,\mathrm{ST}(\hat\beta_j^{(\lambda)} - \tfrac{1}{L}\nabla_j f(\hat\beta^{(\lambda)}), \tfrac{\lambda}{L})_{\mathcal{S}}$$

Experiments I - Lasso cross-validation

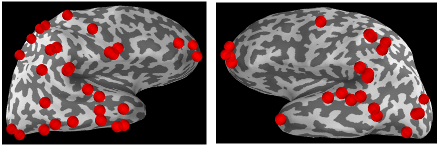


Experiments II - Enet cross-validation



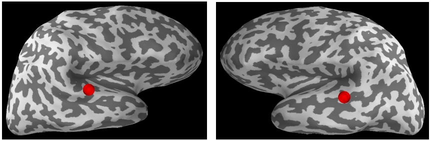
Experiments III - Real MEEG data

- **Outer criterion:** SURE
- **Inner problems:** the Lasso and weighted Lasso



Vanilla Lasso (1 parameter)

$$\arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} ||y - X\beta||_2^2 + \lambda \|\beta\|_1$$



Weighted Lasso (p parameters)

$$\arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} ||y - X\beta||_2^2 + \sum_{j=1}^{p} \lambda_j |\beta_j|$$

Limitations

- Specific parametrization e^{λ}
- Need a **differentiable criterion**: cannot use 0/1-loss
- Need a **continuous estimator** *w.r.t.* data and hyperparameters: does not apply yet to **non-convex** penalties[20] nor reweighted Lasso[21]
- Optimized function often **non-convex**: possibly multiple local minima
- Potentially slow and handy **outer solver**

[20] P. Breheny and J. Huang. "Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection". In: *Ann. Appl. Stat.* 5.1 (2011), p. 232.

[21] E. J. Candès, M. B. Wakin, and S. P. Boyd. "Enhancing Sparsity by Reweighted l_1 Minimization". In: *J. Fourier Anal. Applicat.* 14.5-6 (2008), pp. 877–905.

Conclusion

Contributions:

- 1st-order optimization with nonsmooth inner problem
- **Local linear convergence** of the Jacobian
- **Leverage sparsity** to speed up hypergradient computation

Future work:

- Convergence of the bilevel procedure
- Smarter outer solver

Conclusion

Contributions:

- 1st-order optimization with nonsmooth inner problem
- **Local linear convergence** of the Jacobian
- **Leverage sparsity** to speed up hypergradient computation

Future work:

- Convergence of the bilevel procedure
- Smarter outer solver

References:

- Paper `https://proceedings.icml.cc/paper/2020/file/e0ab531ec312161511493b002f9be2ee-Paper.pdf`
- Open source package `https://github.com/QB3/sparse-ho`

Conclusion

Contributions:

- 1st-order optimization with nonsmooth inner problem
- **Local linear convergence** of the Jacobian
- **Leverage sparsity** to speed up hypergradient computation

Future work:

- Convergence of the bilevel procedure
- Smarter outer solver

References:

- Paper https://proceedings.icml.cc/paper/2020/file/e0ab531ec312161511493b002f9be2ee-Paper.pdf
- Open source package https://github.com/QB3/sparse-ho

- Belloni, A., V. Chernozhukov, and L. Wang. "Square-root Lasso: pivotal recovery of sparse signals via conic programming". In: *Biometrika* 98.4 (2011), pp. 791–806.
- Bengio, Y. "Gradient-based optimization of hyperparameters". In: *Neural computation* 12.8 (2000), pp. 1889–1900.
- Bergstra, J. and Y. Bengio. "Random search for hyper-parameter optimization". In: *Journal of Machine Learning Research* 13.2 (2012).
- Bickel, P. J., Y. Ritov, and A. B. Tsybakov. "Simultaneous analysis of Lasso and Dantzig selector". In: *Ann. Statist.* 37.4 (2009), pp. 1705–1732.
- Breheny, P. and J. Huang. "Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection". In: *Ann. Appl. Stat.* 5.1 (2011), p. 232.

- Brochu, E., V. M. Cora, and N. De Freitas. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning". In: (2010).
- Candès, E. J., M. B. Wakin, and S. P. Boyd. "Enhancing Sparsity by Reweighted l_1 Minimization". In: *J. Fourier Anal. Applicat.* 14.5-6 (2008), pp. 877–905.
- Domke, J. "Generic methods for optimization-based modeling". In: *AISTATS*. Vol. 22. 2012, pp. 318–326.
- Franceschi, L. et al. "Forward and reverse gradient-based hyperparameter optimization". In: *ICML*. 2017, pp. 1165–1173.
- Frecon, J., S. Salzo, and M. Pontil. "Bilevel learning of the group lasso structure". In: *NeurIPS*. 2018, pp. 8301–8311.
- Larsen, J. et al. "Design and regularization of neural networks: the optimal use of a validation set". In: *Neural Networks for Signal Processing VI. Proceedings of the 1996 IEEE Signal Processing Society Workshop*. 1996.

- Liu, D. C. and J. Nocedal. “On the limited memory BFGS method for large scale optimization”. In: *Mathematical programming* 45.1-3 (1989), pp. 503–528.
- Liu, W., Y. Yang, et al. “Parametric or nonparametric? A parametricness index for model selection”. In: *Ann. Statist.* 39.4 (2011), pp. 2074–2102.
- Nocedal, J. and S. J. Wright. *Numerical optimization*. Second. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006.
- Obozinski, G., B. Taskar, and M. I. Jordan. “Joint covariate selection and joint subspace selection for multiple classification problems”. In: *Statistics and Computing* 20.2 (2010), pp. 231–252.
- Ochs, P. et al. “Bilevel optimization with nonsmooth lower level problems”. In: *SSVM*. 2015, pp. 654–665.
- Pedregosa, F. “Hyperparameter optimization with approximate gradient”. In: *ICML*. Vol. 48. 2016, pp. 737–746.

- Peyré, G. and J. M. Fadili. "Learning analysis sparsity priors". In: *Sampta*. 2011.
- Stein, C. M. "Estimation of the mean of a multivariate normal distribution". In: *Ann. Statist.* 9.6 (1981), pp. 1135–1151.
- Stone, L. R. A. and J.C. Ramer. "Estimating WAIS IQ from Shipley Scale scores: Another cross-validation". In: *Journal of clinical psychology* 21.3 (1965), pp. 297–297.