Sparsity in Learning with the LASSO 2

Binh Nguyen – Telecom Paris

M2DS Alternants Research Seminar Course; 14/04/2022

Outline

Outline

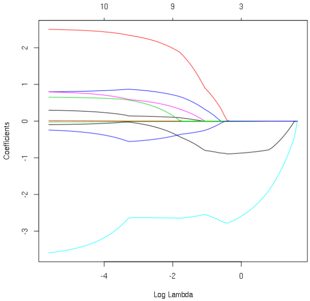
Reminder

Variants of Lasso

Hyperparameter Optimization

Previously...

Lasso: Least Absolute Shrinkage and Selection Operator



$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1$$

where $\lambda > 0$ controls the sparsity of the solution

$\longrightarrow$ Promote sparsity: there is a threshold λ_{max} such that $\lambda > \lambda_{max}$ implies $\beta_{lasso} = 0$

Outline

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1$$

- Only one λ that dictates sparsity degree of all $\boldsymbol{\beta}_j$
- What if we want a scheme that is adaptive: coefficients with large magnitude (absolute value) receive smaller sparse penalty?

Zou H. (2006), 'The adaptive lasso and its oracle properties', Journal of the American Statistical Association 101(476), 1418–1429.

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1$$

- Only one λ that dictates sparsity degree of all $\boldsymbol{\beta}_j$
- What if we want a scheme that is adaptive: coefficients with large magnitude (absolute value) receive smaller sparse penalty?

  $\longrightarrow$ Lasso with adaptive weights on ℓ_1-regularization

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{j=1}^{p} w_j |\boldsymbol{\beta}_j|$$

where $w_j \in [0, 1)$.

Zou H. (2006), 'The adaptive lasso and its oracle properties', Journal of the American Statistical Association 101(476), 1418–1429.

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1$$

- Only one λ that dictates sparsity degree of all $\boldsymbol{\beta}_j$
- What if we want a scheme that is adaptive: coefficients with large magnitude (absolute value) receive smaller sparse penalty?

$\longrightarrow$ Lasso with adaptive weights on ℓ_1-regularization

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{j=1}^{p} w_j |\boldsymbol{\beta}_j|$$

where $w_j \in [0, 1)$.
$\longrightarrow$ Optimization problem is still convex in $\boldsymbol{\beta}$

Zou H. (2006), 'The adaptive lasso and its oracle properties', Journal of the American Statistical Association 101(476), 1418–1429.

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{j=1}^{p} w_j |\boldsymbol{\beta}_j|$$

- Typically w_j are initialized as

$$w_j = \begin{cases} \frac{1}{|\beta_j^{\text{init}}|} & \text{if } \beta_j^{\text{init}} \neq 0 \\ 0 & \text{if } \beta_j^{\text{init}} = 0 \end{cases}$$

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{j=1}^{p} w_j |\boldsymbol{\beta}_j|$$

- Typically w_j are initialized as

$$w_j = \begin{cases} \dfrac{1}{|\beta_j^{\text{init}}|} & \text{if } \beta_j^{\text{init}} \neq 0 \\ 0 & \text{if } \beta_j^{\text{init}} = 0 \end{cases}$$

- But what is this $\boldsymbol{\beta}^{\text{init}}$?

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{j=1}^{p} w_j |\boldsymbol{\beta}_j|$$

- Typically w_j are initialized as

$$w_j = \begin{cases} \dfrac{1}{|\beta_j^{\text{init}}|} & \text{if } \beta_j^{\text{init}} \neq 0 \\ 0 \quad \text{if } \beta_j^{\text{init}} = 0 \end{cases}$$

- But what is this $\boldsymbol{\beta}^{\text{init}}$?
- Just put a standard lasso for finding $\boldsymbol{\beta}^{\text{init}}$ (called screening operation)

Adaptive (weighted) Lasso

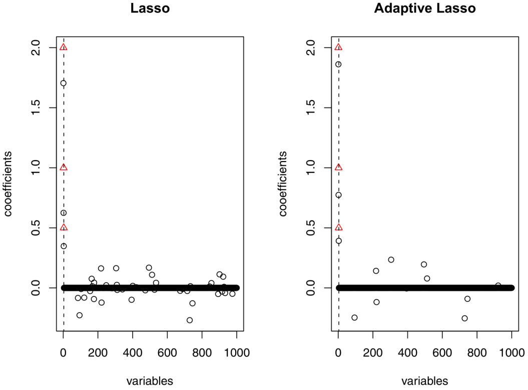


Fig. 2.4 Estimated regression coefficients in the linear model with $p = 1000$ and $n = 50$. Left: Lasso. Right: Adaptive Lasso with Lasso as initial estimator. The 3 true regression coefficients are indicated with triangles. Both methods used with tuning parameters selected from 10-fold cross-validation.

Bühlmann, P., & Geer, S. A. van de. (2011). Statistics for high-dimensional data: Methods, theory and applications. Springer.

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{j=1}^{p} |w_j \boldsymbol{\beta}_j|$$

- Optimization problem is still convex in $\boldsymbol{\beta}$, but how to solve now that there is multiple value of λ possible?

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{j=1}^{p} |w_j \boldsymbol{\beta}_j|$$

- Optimization problem is still convex in $\boldsymbol{\beta}$, but how to solve now that there is multiple value of λ possible?
- Question: can we reformulate the adaptive lasso back to the original lasso?

Adaptive (weighted) Lasso

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda \sum_{j=1}^{p} |w_j \boldsymbol{\beta}_j|$$

- Optimization problem is still convex in $\boldsymbol{\beta}$, but how to solve now that there is multiple value of λ possible?
- Question: can we reformulate the adaptive lasso back to the original lasso?
- Hint: use the change of variable $\tilde{\boldsymbol{\beta}}$ as some form of w and $\boldsymbol{\beta}$
- To the whiteboard...

Adaptive (weighted) Lasso

$$\tilde{\boldsymbol{\beta}}_{lasso} \stackrel{\text{def.}}{=} \underset{\tilde{\boldsymbol{\beta}}}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \tilde{\mathrm{X}}\tilde{\boldsymbol{\beta}}\|^2 + \lambda|\tilde{\boldsymbol{\beta}}|_1$$

and this means

$$\tilde{\boldsymbol{\beta}}_{lasso} = \mathrm{W}^{-1}\boldsymbol{\beta}_{lasso}$$

i.e. the solution of the adaptive Lasso is just a rescaling of the solution of original Lasso
$\longrightarrow$ enjoys theoretical guarantee (consistency, asymptotic normality) from the Lasso without additional assumptions

Adaptive (weighted) Lasso

In `sklearn.linear_model.Lasso`

fit(*X*, *y*, *sample_weight=None*, *check_input=True*) [source]

Fit model with coordinate descent.

Parameters:	**X : *{ndarray, sparse matrix} of (n_samples, n_features)*** Data. **y : *{ndarray, sparse matrix} of shape (n_samples,) or (n_samples, n_targets)*** Target. Will be cast to X's dtype if necessary. **sample_weight : *float or array-like of shape (n_samples,), default=None*** Sample weights. Internally, the `sample_weight` vector will be rescaled to sum to `n_samples`. *New in version 0.23.* **check_input : *bool, default=True*** Allow to bypass several input checking. Don't use this parameter unless you know what you do.

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html

A cousin of Lasso: Elastic-Net

- A problem with Lasso: when there are high-correlations between variables, *e.g.* $\mathrm{X}_{*,i}$ and $\mathrm{X}_{*,j}$ empirically Lasso select one but not both...
- At most n variables will be selected by the lasso, so problematic when $n \ll p$
- A solution: adding ℓ_2 norm to the lasso optimization problem: elastic net

---

Zou, Hui; Hastie, Trevor (2005). "Regularization and Variable Selection via the Elastic Net". Journal of the Royal Statistical Society, Series B. 67 (2): 301–320.

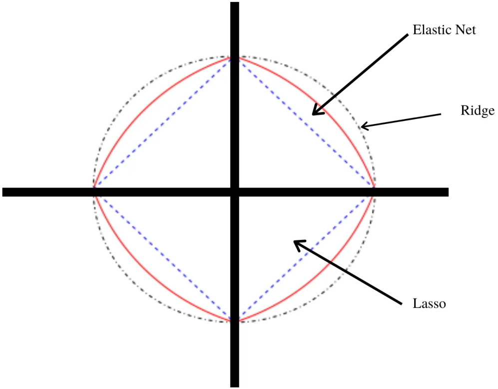
A cousin of Lasso: Elastic-Net

- A problem with Lasso: when there are high-correlations between variables, *e.g.* $X_{*,i}$ and $X_{*,j}$ empirically Lasso select one but not both...
- At most n variables will be selected by the lasso, so problematic when $n \ll p$
- A solution: adding ℓ_2 norm to the lasso optimization problem: elastic net

---

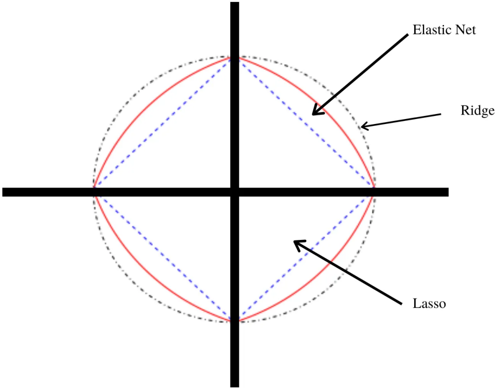
Zou, Hui; Hastie, Trevor (2005). "Regularization and Variable Selection via the Elastic Net". Journal of the Royal Statistical Society, Series B. 67 (2): 301–320.

Elastic-Net



$$\boldsymbol{\beta}_{enet} \overset{\text{def.}}{=} \frac{1}{2n} \operatorname*{argmin}_{\beta} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

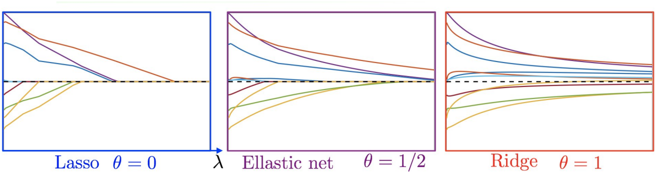
Elastic-Net



$$\boldsymbol{\beta}_{enet} \overset{\text{def.}}{=} \frac{1}{2n} \underset{\beta}{\operatorname{argmin}} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

but now we have two hyper-parameters λ_1 and λ_2?

Elastic-Net



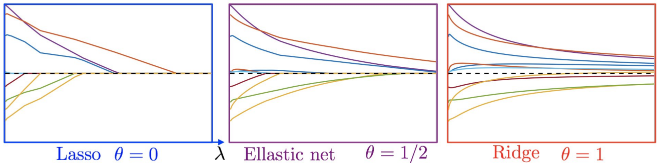
we can just set $\theta = \dfrac{\lambda_2}{\lambda_1 + \lambda_2} \in [0, 1]$, then the equivalent problem is

$$\boldsymbol{\beta}_{enet} \stackrel{\text{def.}}{=} \underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + (1 - \theta)\|\boldsymbol{\beta}\|_1 + \frac{\theta}{2}\|\boldsymbol{\beta}\|_2^2$$

$\longrightarrow$ enet-path interpolates between Lasso and Ridge regression path

Image from Gabriel Peyré's twitter: https://twitter.com/gabrielpeyre/status/1318054267685621761

Elastic-Net



- Elastic-net solutions: interpolates between Lasso and Ridge regression solutions
- Question: this gives hint on finding the solution of Enet? (remember how we find solution for Lasso and for Ridge?)

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \stackrel{\text{def.}}{=} \underset{\beta}{\operatorname{argmin}} \frac{1}{2n}\|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1\|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2$$

$$\text{in the case } \frac{1}{n}\mathrm{X}^\top\mathrm{X} = \mathrm{Id}, \text{ then } \hat{\boldsymbol{\beta}}^{LS} = 1/n(\mathrm{X}^\top\mathrm{X})^{-1}\mathrm{X}^\top\mathrm{y} = \mathrm{X}^\top\mathrm{y}/n$$

so for the first term:

$$\underset{\beta}{\operatorname{argmin}} \frac{1}{2n}\|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2$$

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \stackrel{\text{def.}}{=} \underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

$$\text{in the case } \frac{1}{n}\mathrm{X}^\top \mathrm{X} = \mathrm{Id}, \text{ then } \hat{\boldsymbol{\beta}}^{LS} = 1/n(\mathrm{X}^\top \mathrm{X})^{-1}\mathrm{X}^\top \mathrm{y} = \mathrm{X}^\top \mathrm{y}/n$$

so for the first term:

$$\underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2$$

$$= \underset{\beta}{\operatorname{argmin}} \; \frac{1}{2n} \left\{ \mathrm{y}^\top \mathrm{y} + \boldsymbol{\beta}^\top \boldsymbol{\beta} - 2\mathrm{y}^\top \mathrm{X}\boldsymbol{\beta} \right\}$$

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \overset{\text{def.}}{=} \operatorname*{argmin}_{\beta} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

in the case $\frac{1}{n}\mathrm{X}^\top \mathrm{X} = \mathrm{Id}$, then $\hat{\boldsymbol{\beta}}^{LS} = 1/n(\mathrm{X}^\top \mathrm{X})^{-1}\mathrm{X}^\top \mathrm{y} = \mathrm{X}^\top \mathrm{y}/n$

so for the first term:

$$\operatorname*{argmin}_{\beta} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2$$

$$= \operatorname*{argmin}_{\beta} \; \frac{1}{2n} \left\{ \mathrm{y}^\top \mathrm{y} + \boldsymbol{\beta}^\top \boldsymbol{\beta} - 2\mathrm{y}^\top \mathrm{X}\boldsymbol{\beta} \right\}$$

$$= \operatorname*{argmin}_{\beta} \; \frac{1}{2n} \left\{ \mathrm{y}^\top \mathrm{y} + n\boldsymbol{\beta}^\top \boldsymbol{\beta} + 2n\boldsymbol{\beta}^\top \hat{\boldsymbol{\beta}}^{LS} + n(\hat{\boldsymbol{\beta}}^{LS})^\top \hat{\boldsymbol{\beta}}^{LS} - n(\hat{\boldsymbol{\beta}}^{LS})^\top \hat{\boldsymbol{\beta}}^{LS} \right\}$$

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \overset{\text{def.}}{=} \underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

in the case $\frac{1}{n}\mathrm{X}^\top \mathrm{X} = \mathrm{Id}$, then $\hat{\boldsymbol{\beta}}^{LS} = 1/n(\mathrm{X}^\top \mathrm{X})^{-1}\mathrm{X}^\top \mathrm{y} = \mathrm{X}^\top \mathrm{y}/n$

so for the first term:

$$\underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2$$

$$= \underset{\beta}{\operatorname{argmin}} \; \frac{1}{2n} \left\{ \mathrm{y}^\top \mathrm{y} + \boldsymbol{\beta}^\top \boldsymbol{\beta} - 2\mathrm{y}^\top \mathrm{X}\boldsymbol{\beta} \right\}$$

$$= \underset{\beta}{\operatorname{argmin}} \; \frac{1}{2n} \left\{ \mathrm{y}^\top \mathrm{y} + n\boldsymbol{\beta}^\top \boldsymbol{\beta} + 2n\boldsymbol{\beta}^\top \hat{\boldsymbol{\beta}}^{LS} + n(\hat{\boldsymbol{\beta}}^{LS})^\top \hat{\boldsymbol{\beta}}^{LS} - n(\hat{\boldsymbol{\beta}}^{LS})^\top \hat{\boldsymbol{\beta}}^{LS} \right\}$$

$$= \underset{\beta}{\operatorname{argmin}} \; \frac{1}{2n} \left\{ (\hat{\boldsymbol{\beta}}^{LS} - \boldsymbol{\beta})^\top (n\hat{\boldsymbol{\beta}}^{LS} - \boldsymbol{\beta}) + \mathrm{y}^\top (\mathrm{Id} - \mathrm{X}^\top \mathrm{X})\mathrm{y} \right\}$$

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \stackrel{\text{def.}}{=} \underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

in the case $\frac{1}{n}\mathrm{X}^\top \mathrm{X} = \mathrm{Id}$, then $\hat{\boldsymbol{\beta}}^{LS} = 1/n(\mathrm{X}^\top \mathrm{X})^{-1}\mathrm{X}^\top \mathrm{y} = \mathrm{X}^\top \mathrm{y}/n$

so for the first term:

$$\underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2$$

$$= \underset{\beta}{\operatorname{argmin}} \; \frac{1}{2n} \left\{\mathrm{y}^\top \mathrm{y} + \boldsymbol{\beta}^\top \boldsymbol{\beta} - 2\mathrm{y}^\top \mathrm{X}\boldsymbol{\beta}\right\}$$

$$= \underset{\beta}{\operatorname{argmin}} \; \frac{1}{2n} \left\{\mathrm{y}^\top \mathrm{y} + n\boldsymbol{\beta}^\top \boldsymbol{\beta} + 2n\boldsymbol{\beta}^\top \hat{\boldsymbol{\beta}}^{LS} + n(\hat{\boldsymbol{\beta}}^{LS})^\top \hat{\boldsymbol{\beta}}^{LS} - n(\hat{\boldsymbol{\beta}}^{LS})^\top \hat{\boldsymbol{\beta}}^{LS}\right\}$$

$$= \underset{\beta}{\operatorname{argmin}} \; \frac{1}{2n} \left\{(\hat{\boldsymbol{\beta}}^{LS} - \boldsymbol{\beta})^\top (n\hat{\boldsymbol{\beta}}^{LS} - \boldsymbol{\beta}) + \mathrm{y}^\top (\mathrm{Id} - \mathrm{X}^\top \mathrm{X})\mathrm{y}\right\}$$

$$= \underset{\beta}{\operatorname{argmin}} \; \frac{1}{2} \|\hat{\boldsymbol{\beta}}^{LS} - \boldsymbol{\beta}\|_2^2$$

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \overset{\text{def.}}{=} \operatorname*{argmin}_{\beta} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

$$\text{in the case } \frac{1}{n}\mathrm{X}^\top \mathrm{X} = \mathrm{Id}$$

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \stackrel{\text{def.}}{=} \operatorname*{argmin}_{\beta} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

$$\text{in the case } \frac{1}{n} \mathrm{X}^\top \mathrm{X} = \mathrm{Id}$$

- This means

$$\boldsymbol{\beta}_{enet} = \operatorname*{argmin}_{\beta} \frac{1}{2} \sum_{j=1}^{p} (\hat{\beta}_j^{LS} - \beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \frac{\lambda_2}{2} \sum_{j=1}^{p} \beta_j^2$$

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \overset{\text{def.}}{=} \underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

$$\text{in the case } \frac{1}{n} \mathrm{X}^\top \mathrm{X} = \mathrm{Id}$$

- This means

$$\boldsymbol{\beta}_{enet} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2} \sum_{j=1}^{p} (\hat{\beta}_j^{LS} - \beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \frac{\lambda_2}{2} \sum_{j=1}^{p} \beta_j^2$$

- The problem is separable: for each j

$$\beta_j^{enet} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2} (\hat{\beta}_j^{LS} - \beta_j)^2 + \lambda_1 |\beta_j| + \frac{\lambda_2}{2} \beta_j^2$$

Elastic-Net, in Orthogonal Design settings

$$\boldsymbol{\beta}_{enet} \overset{\text{def.}}{=} \underset{\beta}{\operatorname{argmin}} \frac{1}{2n} \|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|_2^2 + \lambda_1 \|\boldsymbol{\beta}\|_1 + \frac{\lambda_2}{2} \|\boldsymbol{\beta}\|_2^2$$

$$\text{in the case } \frac{1}{n}\mathrm{X}^\top \mathrm{X} = \mathrm{Id}$$

- This means

$$\boldsymbol{\beta}_{enet} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2} \sum_{j=1}^{p} (\hat{\beta}_j^{LS} - \beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \frac{\lambda_2}{2} \sum_{j=1}^{p} \beta_j^2$$

- The problem is separable: for each j

$$\begin{aligned}
\beta_j^{enet} &= \underset{\beta}{\operatorname{argmin}} \frac{1}{2} (\hat{\beta}_j^{LS} - \beta_j)^2 + \lambda_1 |\beta_j| + \frac{\lambda_2}{2} \beta_j^2 \\
&= \underset{\beta}{\operatorname{argmin}} \frac{1}{2} \left(\beta_j - \frac{\hat{\beta}_j^{LS}}{1+\lambda_2} \right)^2 + \frac{\lambda_1}{1+\lambda_2} |\beta_j| \\
&\overset{\text{def.}}{=} \operatorname{prox}_{\|\cdot\|_1} \left(\beta_j - \frac{\hat{\beta}_j^{LS}}{1+\lambda_2}, \frac{\lambda_1}{1+\lambda_2} \right)
\end{aligned}$$

Elastic-Net

This means: in general settings, we can find solution of Enet with iterative optimization algorithm (from last session):

- ISTA, FISTA
- Coordinate descent (implemented in `sklearn`)

Other Variants

- Group lasso
- Lasso for data matrix with missing elements
- Debiased Lasso

Other Variants

- Group lasso
- Lasso for data matrix with missing elements
- Debiased Lasso

  ...which we will wait for presentations next week :-)

Reminder

Variants of Lasso

Hyperparameter Optimization

Previously...

Lasso: Least Absolute Shrinkage and Selection Operator

$$\boldsymbol{\beta}_{lasso} \stackrel{\text{def.}}{=} \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{2}\|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_1$$

where $\lambda > 0$ controls the sparsity of the solution

- Choose λ based $\lambda_{max} = \|\mathrm{X}^\top \boldsymbol{y}\|_\infty$
- Reminder: when $\lambda > \lambda_{max}$ all β_j will shrink to zero
- But λ to select? – cross-validation/Information Criterion

Hyperparameter selection, the popular way

- Cross validation
- Criterion (AIC/BIC) that control model complexity

Hyperparameter selection, the popular way

- Cross validation
- Criterion (AIC/BIC) that control model complexity
- Formalization: for Lasso

$$\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{2n} \| \mathtt{y}^{\mathtt{train}} - \mathrm{X}^{\mathtt{train}} \boldsymbol{\beta} \|^2 + \lambda \| \boldsymbol{\beta} \|_1$$

- Subject to:

$$\mathcal{L}(\lambda) = \min_{\lambda} \| \mathtt{y}^{\mathtt{val}} - \mathrm{X}^{\mathtt{val}} \hat{\boldsymbol{\beta}}^{(\lambda)} \|^2$$

Hyperparameter selection, the popular way

- Cross validation
- Criterion (AIC/BIC) that control model complexity
- Formalization: for Lasso

$$\hat{\boldsymbol{\beta}}^{(\lambda)} \in \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{2n} \|\mathtt{y}^{\mathtt{train}} - \mathtt{X}^{\mathtt{train}} \boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1$$

- Subject to:

$$\mathcal{L}(\lambda) = \min_{\lambda} \|\mathtt{y}^{\mathtt{val}} - \mathtt{X}^{\mathtt{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2$$

$\longrightarrow$ Today: hyper-parameter selection with bi-level optimization

Hyperparameter Selection: Bilevel Optimization?

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) = \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathrm{val}} - \mathrm{X}^{\mathrm{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \text{ s.t. } \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} h(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

Caveat: for the moment we deviate from Lasso, and assume the case h is at least twice-differentiable

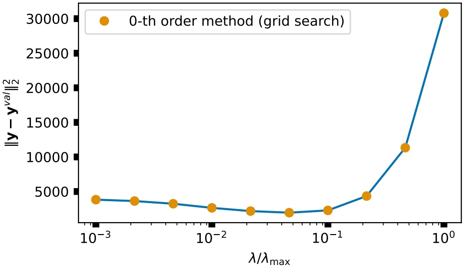
Grid-search as a zero-order optimization method

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \;=\; \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathtt{val}} - \mathrm{X}^{\mathtt{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \;\text{s.t.}\; \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} h(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

Grid-search with cross-validation (assume 1-fold CV):

- Defines a range of values for λ
- For each λ, solves the inner problem, then calculate the outer loss
- Choose $\lambda \in \text{grid}(\lambda)$ that that minimizes the outer loss

Grid-search as a zero-order optimization method



Grid-search with cross-validation (assume 1-fold CV):

- Defines a range of values for λ
- For each λ, solves the inner problem, then calculate the outer loss
- Choose $\lambda \in \text{grid}(\lambda)$ that that minimizes the outer loss

Example from: `https://qb3.github.io/sparse-ho/index.html`

First-order hyperparameter-optimization?

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \quad = \quad \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathtt{val}} - \mathrm{X}^{\mathtt{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \text{ s.t. } \quad \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \boldsymbol{h}(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

- Idea: gradient descent?

$$\lambda^{(t+1)} = \lambda^{(t)} - \eta \nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda^{(t)})$$

First-order hyperparameter-optimization?

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \quad = \quad \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathtt{val}} - \mathrm{X}^{\mathtt{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \ \text{s.t.} \quad \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \boldsymbol{h}(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

- Idea: gradient descent?

$$\lambda^{(t+1)} = \lambda^{(t)} - \eta \nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda^{(t)})$$

First-order hyperparameter-optimization

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \quad = \quad \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathrm{val}} - \mathrm{X}^{\mathrm{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \ \text{s.t.} \quad \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} h(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

- Previous calculus classes tell us that

$$\nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) = \partial_\lambda \hat{\boldsymbol{\beta}}^{(\lambda)\top} \nabla_1 \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) + \nabla_2 \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda)$$

First-order hyperparameter-optimization

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \quad = \quad \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathrm{val}} - \mathrm{X}^{\mathrm{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \ \text{s.t.} \quad \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} h(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

- Previous calculus classes tell us that

$$\nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) = \partial_\lambda \hat{\boldsymbol{\beta}}^{(\lambda)\top} \nabla_1 \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) + \nabla_2 \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda)$$

- Question: what is problematic in computation of this gradient?

First-order hyperparameter-optimization

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \quad = \quad \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathrm{val}} - \mathrm{X}^{\mathrm{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \text{ s.t. } \quad \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} h(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

- Previous calculus classes tell us that

$$\nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) = \partial_\lambda \hat{\boldsymbol{\beta}}^{(\lambda)\top} \nabla_1 \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) + \nabla_2 \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda)$$

- Question: what is problematic in computation of this gradient?
- $\hat{\boldsymbol{\beta}}^{(\lambda)}$ is the solution of another optimization problem...

Implicit Function Theorem to the rescue

Remember the inner problem:

$$\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \boldsymbol{h}(\boldsymbol{\beta}, \lambda)$$

Implicit Function Theorem to the rescue

Remember the inner problem:

$$\hat{\boldsymbol{\beta}}^{(\lambda)} \in \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \, h(\boldsymbol{\beta}, \lambda)$$

- $\hat{\boldsymbol{\beta}}^{(\lambda)}$ is an implicit function of λ, characterized by

$$\nabla_1 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) = 0$$

Implicit Function Theorem to the rescue

Remember the inner problem:

$$\hat{\boldsymbol{\beta}}^{(\lambda)} \in \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \, h(\boldsymbol{\beta}, \lambda)$$

- $\hat{\boldsymbol{\beta}}^{(\lambda)}$ is an implicit function of λ, characterized by

$$\nabla_1 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) = 0$$

- Implicit Function Theorem: if $\mathcal{L}$ and h are continuously differentiable, then there exists a unique $\hat{\boldsymbol{\beta}}^{(\lambda)}$, and we have

$$\begin{aligned} \partial_\lambda \hat{\boldsymbol{\beta}}^{(\lambda)} &= -[\nabla_1^2 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda)]^{-1} \nabla_{1,2}^2 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \\ &= -[H_{\beta,h}]^{-1} \; \nabla_{1,2}^2 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \end{aligned}$$

Implicit Function Theorem to the rescue

Remember the inner problem:

$$\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} h(\boldsymbol{\beta}, \lambda)$$

- $\hat{\boldsymbol{\beta}}^{(\lambda)}$ is an implicit function of λ, characterized by

$$\nabla_1 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) = 0$$

- Implicit Function Theorem: if $\mathcal{L}$ and h are continuously differentiable, then there exists a unique $\hat{\boldsymbol{\beta}}^{(\lambda)}$, and we have

$$\begin{aligned} \partial_\lambda \hat{\boldsymbol{\beta}}^{(\lambda)} &= -[\nabla_1^2 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda)]^{-1} \nabla_{1,2}^2 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \\ &= -[H_{\beta,h}]^{-1} \ \nabla_{1,2}^2 h(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \end{aligned}$$

- Question: where does this equation come from?

First-order hyperparameter-optimization

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \;=\; \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathrm{val}} - \mathrm{X}^{\mathrm{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \;\text{s.t.}\; \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \boldsymbol{h}(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

- So:

$$\begin{aligned} \nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) &= \partial_\lambda \hat{\boldsymbol{\beta}}^{(\lambda)\top} \nabla_1 \mathcal{L} + \nabla_2 \mathcal{L} \\ &= -[\nabla^2_{1,2} h]^\top [H_{\beta,h}]^{-1} \nabla_1 \mathcal{L} + \nabla_2 \mathcal{L} \end{aligned}$$

Y. Bengio. Gradient-based optimization of hyperparameters. Neural computation, 12(8):1889–1900, 2000.

First-order hyperparameter-optimization

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \quad = \quad \underbrace{\min_{\lambda} \| \mathrm{y}^{\mathrm{val}} - \mathrm{X}^{\mathrm{val}} \hat{\boldsymbol{\beta}}^{(\lambda)} \|^2}_{\text{outer optimization problem}} \text{ s.t. } \quad \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \boldsymbol{h}(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

- So:

$$\begin{aligned} \nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) &= \partial_\lambda \hat{\boldsymbol{\beta}}^{(\lambda)\top} \nabla_1 \mathcal{L} + \nabla_2 \mathcal{L} \\ &= -[\nabla^2_{1,2} h]^\top [H_{\beta,h}]^{-1} \nabla_1 \mathcal{L} + \nabla_2 \mathcal{L} \end{aligned}$$

- But: any problem remains?

Y. Bengio. Gradient-based optimization of hyperparameters. Neural computation, 12(8):1889–1900, 2000.

First-order hyperparameter-optimization

$$\mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) \quad = \quad \underbrace{\min_{\lambda} \|\mathrm{y}^{\mathrm{val}} - \mathrm{X}^{\mathrm{val}} \hat{\boldsymbol{\beta}}^{(\lambda)}\|^2}_{\text{outer optimization problem}} \text{ s.t. } \quad \underbrace{\hat{\boldsymbol{\beta}}^{(\lambda)} \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} h(\boldsymbol{\beta}, \lambda)}_{\text{inner optimization problem}}$$

- So:

$$\begin{aligned} \nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) &= \partial_\lambda \hat{\boldsymbol{\beta}}^{(\lambda)\top} \nabla_1 \mathcal{L} + \nabla_2 \mathcal{L} \\ &= -[\nabla^2_{1,2} h]^\top [H_{\beta,h}]^{-1} \nabla_1 \mathcal{L} + \nabla_2 \mathcal{L} \end{aligned}$$

- But: any problem remains?
- Inverting Hessian is generally very costly, and not possible when $n < p$...

Y. Bengio. Gradient-based optimization of hyperparameters. Neural computation, 12(8):1889–1900, 2000.

First-order hyperparameter-optimization

$$\nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda) = -[\nabla^2_{1,2} h]^\top [H_{\beta,h}]^{-1} \nabla_1 \mathcal{L} + \nabla_2 \mathcal{L}$$

Pedregosa (2016): at iteration k we have a tolerance ϵ_k small enough

1. With λ_k, solve the inner optimization problem, obtain $\hat{\boldsymbol{\beta}}^{\lambda_k}$
2. Approximate $[H_{\beta,h}]^{-1} \nabla_1 \mathcal{L}$ by solving for q_k s.t

$$\| H_{\hat{\boldsymbol{\beta}}^{\lambda_k}, h} q_k - \nabla_1 \mathcal{L} \| \leq \epsilon_k$$

3. Approximate $\nabla \mathcal{L}(\hat{\boldsymbol{\beta}}^{(\lambda)}, \lambda)$ with

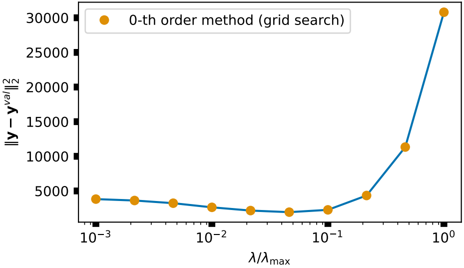
$$p_k = -[\nabla^2_{1,2} h]^\top q_k + \nabla_2 \mathcal{L}(\hat{\beta}^{\lambda_k}, \lambda_k)$$

4. Update $\lambda_{k+1} = \mathrm{ProjGD}(\lambda_k, p_k, \eta)$

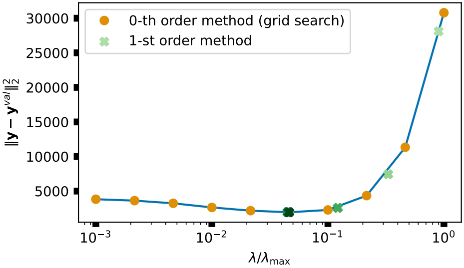
$\longrightarrow$ no inversion of the Hessian

Pedregosa, F. (2016). Hyperparameter optmimization with approximate gradient. In International conference on machine learning (pp. 737-746). PMLR.

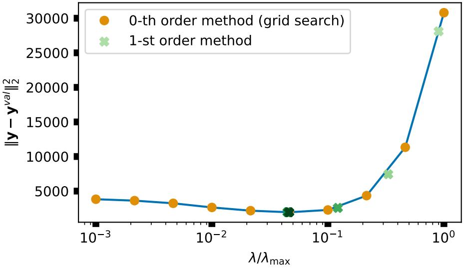
First-order hyperparameter-optimization



First-order hyperparameter-optimization



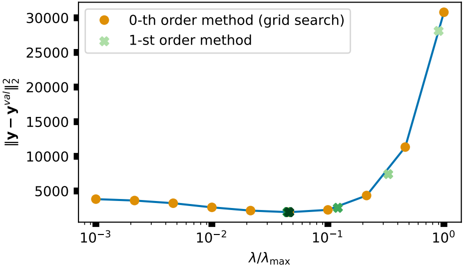
First-order hyperparameter-optimization



- Still: we requires h to be smooth
- But what about the case for Lasso?

$$h(\boldsymbol{\beta}, \lambda) = \frac{1}{2}\|\mathrm{y} - \mathrm{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_1$$

First-order hyperparameter-optimization



$\longrightarrow$ Check the work of Bertrand et al. (2020)

- Also leverage the sparsity induced by the Lasso for the computation
- Faster than implicit forward differentiation methods

Bertrand, Q., Klopfenstein, Q., et al. (2020). Implicit differentiation of Lasso-type models for hyperparameter optimization. Proceedings of the 37th International Conference on Machine Learning