Non-smooth optimization for the estimation of cellular immune components in a tumoral environment

Ph.D. defense

Quentin Klopfenstein

Institut de Mathématiques de Bourgogne

June 30, 2021

- Chloé-Agathe Azencott (Examinatrice)
- Hervé Cardot (Co-directeur)
- Jalal Fadili (Examinateur)
- Enrico Glaab (Examinateur)
- Sophie Lambert-Lacroix (Rapportrice)
- Jérôme Malick (Rapporteur)
- Nelly Pustelnik (Examinatrice)
- Samuel Vaiter (Directeur)

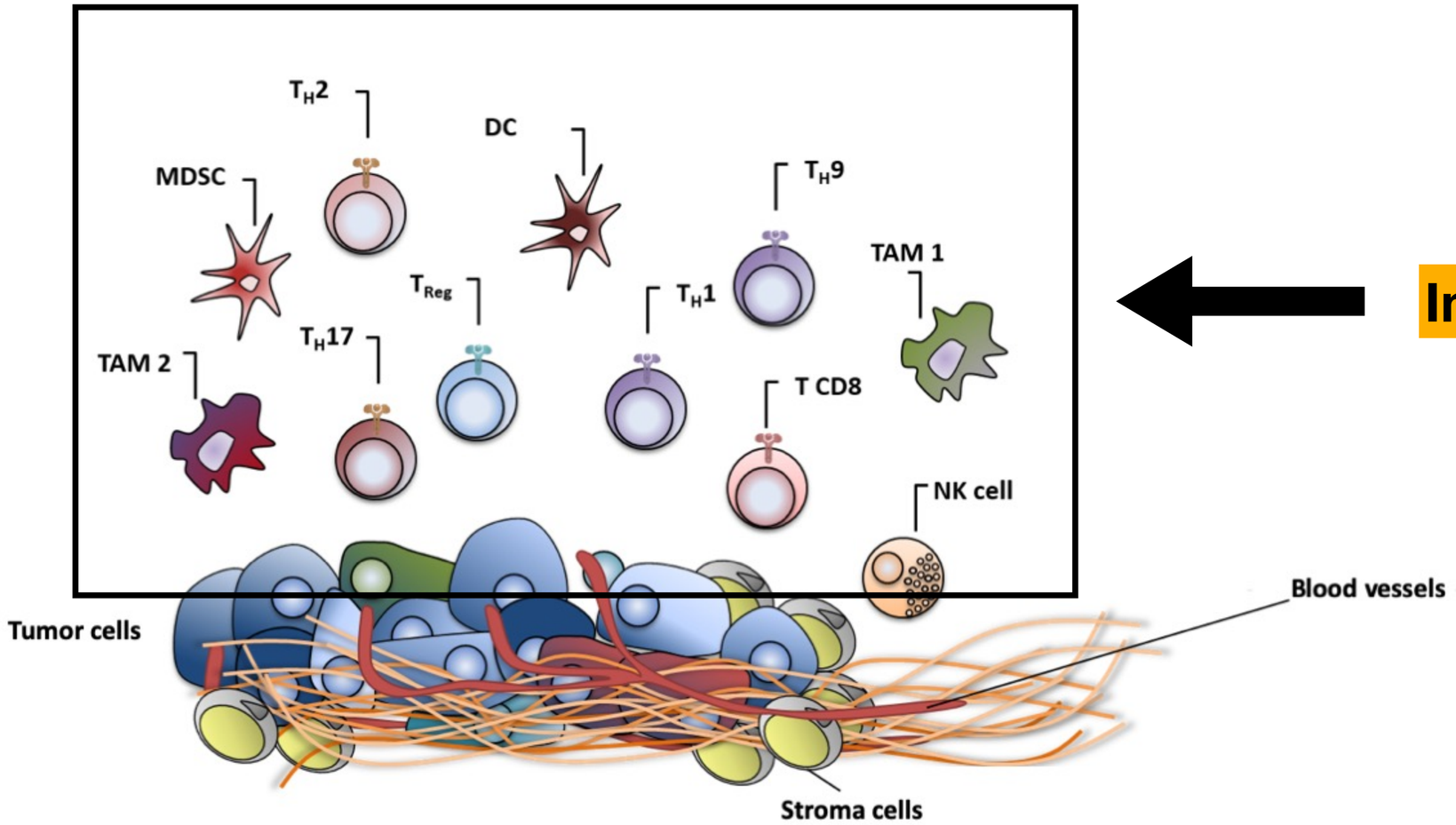
The importance of the immune system

A new cancer treatment: immunotherapy

[Couzin-Frankel ’13]

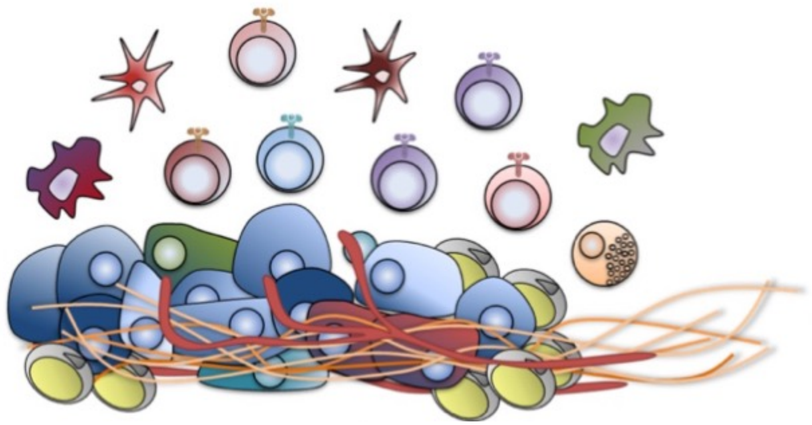
- First results seemed **promising**
- Unfortunately, only a **small proportions** of patients respond to immunotherapy
- Goal: find the patients that will respond and **understand the reasons**
- Information about the **tumoral environment** can be key [Glaire et al. ’19, Reichling et al. ’20]

The cellular composition of a tumor



Immune cells

A linear inverse problem



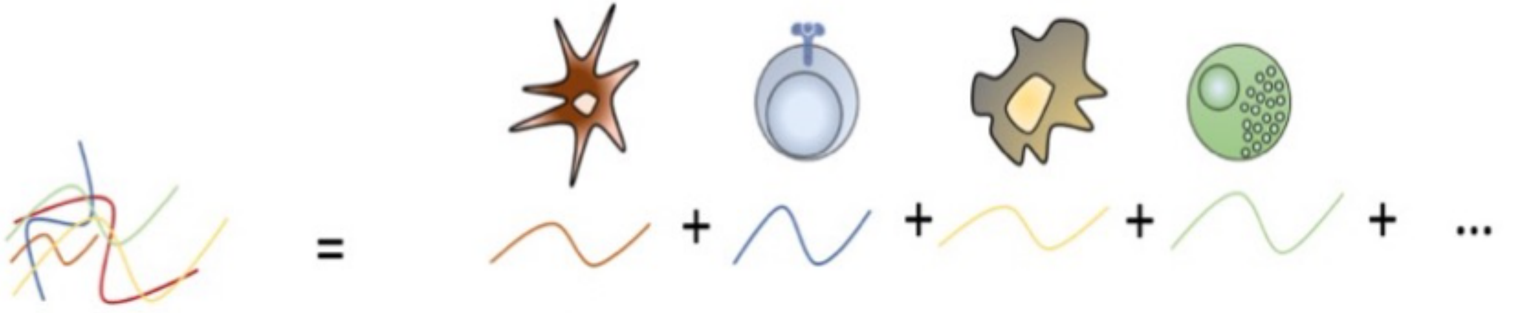
Quantity/Proportions of cells

$$\underset{n \approx 10000}{\begin{bmatrix} y \end{bmatrix}} = \underset{p \approx 25}{\begin{bmatrix} X \end{bmatrix}} \begin{bmatrix} \beta \end{bmatrix} + \epsilon$$

$$\beta \in \{x \in \mathbb{R}^p : x_j \geq 0\}$$ Positivity constraints

$$\beta \in \{x \in \mathbb{R}^p : x_j \geq 0, \sum_{j=1}^{p} x_j = 1\}$$ Simplex constraints

Extraction of tumor DNA/RNA

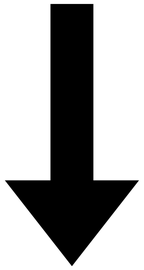


Mixed signal **Pure signals**

State-of-the-art

Authors	Date	Estimator	Constraints
Abbas et al.	2009	Ordinary Least Squares	None
Qiao et al.	2012	Ordinary Least Squares	Positivity
Gong et al.	2011	Ordinary Least Squares	Simplex
Newman et al.	2015	ν-SVR	None

Gold Standard



CIBERSORT

[*source*: cibersort.stanford.edu/]

1) Compute SVR estimator

2) Set negative coefficients to zero

3) Divide by the sum of the remaining coefficients

ν-Support Vector Regression [Schölkopf et al. '99]

Depends on 2 hyperparameters:

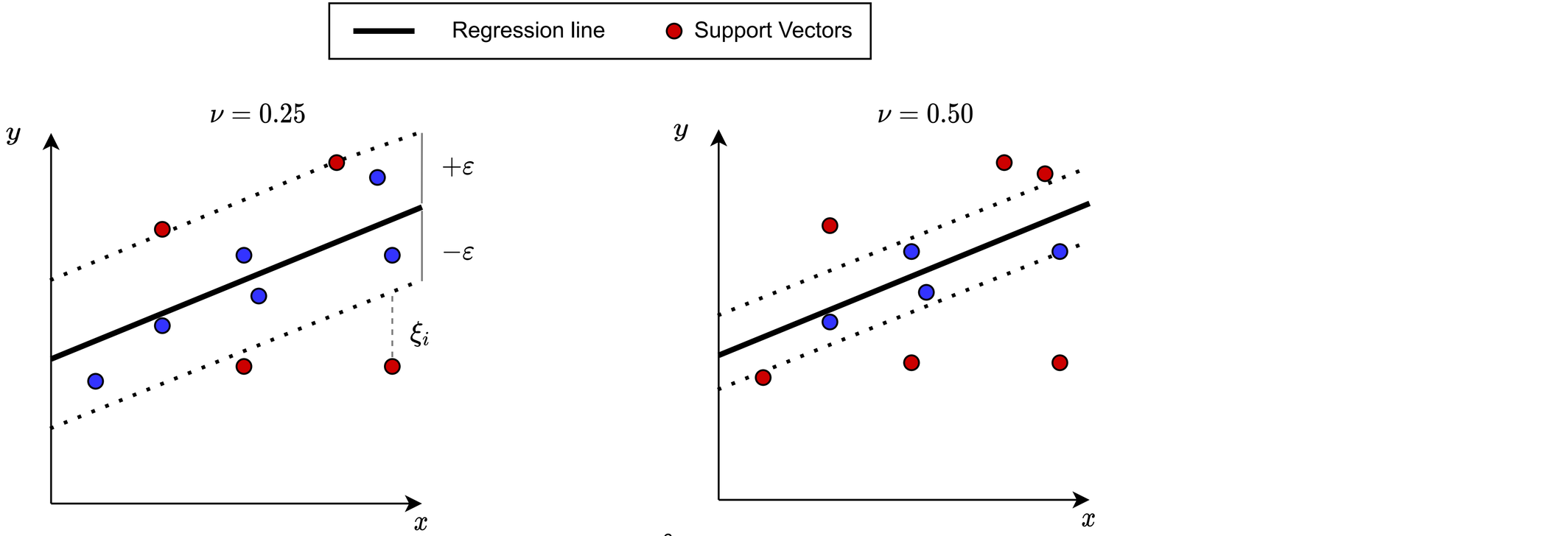
$$C > 0$$
$$\nu \in [0, 1]$$

$$\hat{\beta}_{\text{SVR}} \in \underset{\beta, \beta_0, \xi_i, \xi_i^*, \varepsilon}{\arg\min}$$

$$\frac{1}{2}||\beta||^2 + C(\nu\varepsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*))$$

$$\text{s.t.}$$

$$y_i - X_{i:}\beta - \beta_0 \leq \varepsilon + \xi_i$$
$$X_{i:}\beta + \beta_0 - y_i \leq \varepsilon + \xi_i^*$$
$$\xi_i, \xi_i^* \geq 0, \varepsilon \geq 0 \ .$$



Possible improvements?

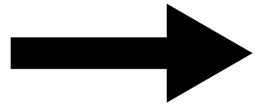
1) $$\hat{\beta}_{\text{SSVR}} \in \underset{\beta,\beta_0,\xi_i,\xi_i^*,\varepsilon}{\arg\min} \quad \frac{1}{2}||\beta||^2 + C(\nu\varepsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*))$$

$$\text{s.t.} \quad y_i - X_{i:}\beta - \beta_0 \leq \varepsilon + \xi_i$$

$$X_{i:}\beta + \beta_0 - y_i \leq \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0, \varepsilon \geq 0$$

Simplex constraints $\beta_j \geq 0, \sum_{j=1}^{p} \beta_j = 1$.



How to solve this optimization problem?

2) The hyperparameters are set as follows:

$$C = 1 \qquad \nu \in \{0.25, 0.5, 0.75\}$$

Is there a better way to find the best pair of hyperparameters than grid-search?

Fit the three models and keep the one that minimizes the prediction error

Composite minimization problem

$$\hat{\beta} \in \underset{\beta \in \mathbb{R}^p}{\arg\min} \; f(\beta) + \underbrace{\sum_{j=1}^{p} g_j(\beta_j)}_{:= g(\beta)}$$

$f : \mathbb{R}^p \to \mathbb{R}$

- Convex
- Differentiable
- Lipschitz Gradient (Smooth)

$g_j : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$

- Convex
- Proper $\quad \mathrm{dom}(f) \neq \emptyset$
- Closed

Examples:

- Lasso: $f(\beta) = \frac{1}{2}\|y - X\beta\|^2 \qquad g_j(\beta_j) = \lambda|\beta_j|$
- Dual SVM: $f(\beta) = \frac{1}{2}\beta^\top Q\beta - \mathbf{e}^\top\beta \qquad g_j(\beta_j) = \iota_{[0,C]}(\beta_j)$

Coordinate descent

$$\hat{\beta} \in \arg\min_{\beta \in \mathbb{R}^p} f(\beta) + \sum_{j=1}^{p} g_j(\beta_j)$$

Proximal operator of a convex function:

For any $\gamma > 0$ and $x \in \mathbb{R}^p$: $\text{prox}_{\gamma g}(x) = \arg\min_{y \in \mathbb{R}^p} \frac{1}{2\gamma}\|x - y\|^2 + g(y)$

Algorithm 1 Proximal coordinate descent

init : $\beta = 0_p$, $L \in \mathbb{R}^p$

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

  for $j = 1, \ldots, p$ **do**

    $z_j \leftarrow \beta_j - \frac{1}{L_j} \nabla_j f(\beta)$; // gradient step

    $\beta_j \leftarrow \text{prox}_{\lambda g_j / L_j}(z_j)$; // proximal step

return β

Local Lipschitz constant

Fixed point equation: $\hat{\beta}_j^{(\lambda)} = \text{prox}_{\gamma_j g_j}\left(\hat{\beta}_j^{(\lambda)} - \gamma_j \nabla_j f(\hat{\beta}^{(\lambda)})\right)$ [Combettes-Wajs '05]

Convergence results: [Tseng and Yun '09] [Razaviyayn et al. '13]

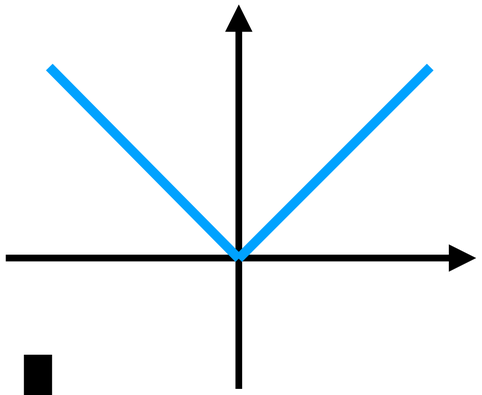
Structure in non-smooth optimization

Definition: Generalized support (e.g. Nutini et al. '19)

For a vector $x \in \mathbb{R}^p$, its generalized support $S_x \subseteq \{1, \dots, p\}$ is the set of indices $j \in \{1, \dots, p\}$ such that g_j is differentiable at x_j:

$$S_x := \{j \in \{1, \dots, p\} : \partial g_j(x_j) \text{ is a singleton}\}.$$

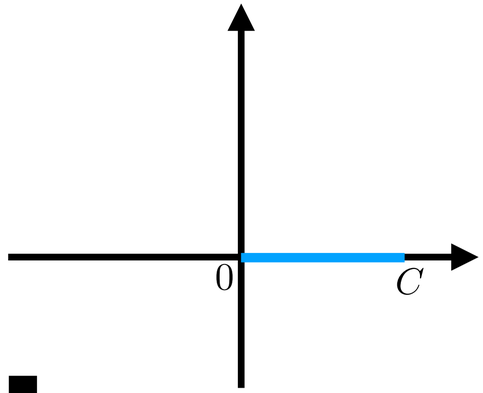
Example: ℓ_1-norm



$$g_j(x_j) = |x_j|$$

$$S_x = \{j \in [p] : x_j \neq 0\}$$

Example: $\iota_{[0,C]}$



$$g_j(x_j) = \begin{cases} 0, \text{ if } x_j \in [0, C] \\ +\infty \text{ otherwise} \end{cases}$$

$$S_x = \{j \in [p] : x_j \in]0, C[\}$$

Structure identification for coordinate descent

- Proximal gradient descent identifies the generalized support after a finite number of iterations [Liang et al. '15]
- Is it the case for coordinate descent? [Nutini '18]

Theorem: [Q.K. et al. '20]

Let $\hat{\beta} \in \arg\min_{\beta \in \mathbb{R}^p} f(\beta) + \sum_{j=1}^{p} g_j(\beta_j)$ and $S = S_{\hat{\beta}}$. Suppose

- **Local regularity**: For all $j \in S$, g_j is locally $\mathcal{C}^2$ around $\hat{\beta}_j$ and f is locally $\mathcal{C}^2$ around $\hat{\beta}$
- **Non-degeneracy**: $-\nabla f(\hat{\beta}) \in \mathrm{ri}(\partial g(\hat{\beta}))$
- **Convergence**: Coordinate algorithm converges to $\hat{\beta}$

Then, the coordinate descent algorithm identifies the support after a finite number of iterations.

Local linear convergence

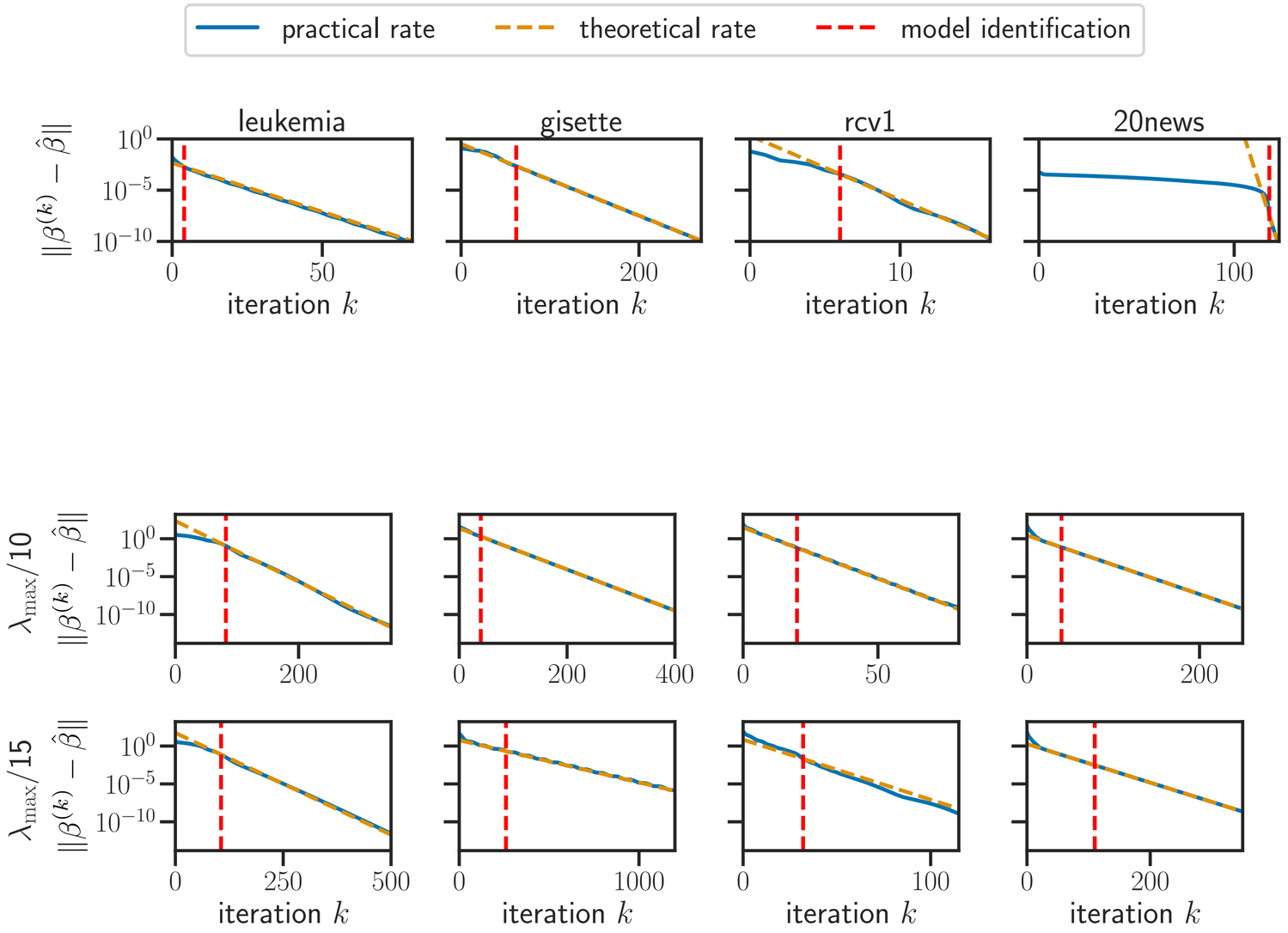
Theorem: [Q.K. et al. '20]

Let $\hat{\beta} \in \arg\min_{\beta \in \mathbb{R}^p} f(\beta) + \sum_{j=1}^{p} g_j(\beta_j)$ and $S = S_{\hat{\beta}}$. Suppose

- **Local regularity**: For all $j \in S$, g_j is locally $\mathcal{C}^2$ around $\hat{\beta}_j$ and f is locally $\mathcal{C}^2$ around $\hat{\beta}$
- **Non-degeneracy**: $-\nabla f(\hat{\beta}) \in \mathrm{ri}(\partial g(\hat{\beta}))$
- **Convergence**: Coordinate algorithm converges to $\hat{\beta}$
- **Restricted injectivity**: $\nabla^2_{S,S} f(\hat{\beta}) \succ 0$
- **Identification**: The support S has been identified at iteration $K > 0$

Then $(\beta^{(k)})_{k \geq K}$ converges linearly towards $\hat{\beta}$.

Illustration of identification and linear convergence



SVM:

$$\arg\min_{\alpha \in \mathbb{R}^n} \quad \frac{1}{2}\alpha^\top Q\alpha - \mathbf{e}^\top \alpha$$

$$\text{subject to} \quad 0 \leq \alpha_i \leq C$$

Lasso:

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

$$\lambda_{\max} = \frac{\|X^\top y\|_\infty}{n}$$

Linear constraints on Support Vector Regression

$$\min_{\beta,\beta_0,\xi_i,\xi_i^*,\varepsilon} \quad \frac{1}{2}||\beta||^2 + C(\nu\epsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*))$$

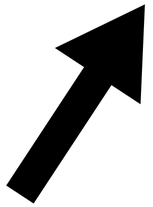
$$\text{subject to} \quad X_{i:}\beta + \beta_0 - y_i \leq \varepsilon + \xi_i$$

$$y_i - X_{i:}\beta - \beta_0 \leq \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0, \varepsilon \geq 0$$

$$A\beta \leq b$$

$$\Gamma\beta = d$$



Assumption: Non-empty feasible set

- Non-negative regression:

$$A = -\mathrm{Id}, \quad b = \mathbf{0}, \quad \Gamma = \mathbf{0}, \quad d = 0$$

- Simplex regression:

$$A = -\mathrm{Id}, \quad b = \mathbf{0}, \quad \Gamma = \mathbf{e}^\top, \quad d = 1$$

- Isotonic regression:

$$A = \begin{bmatrix} 1 & -1 & 0 & \dots & 0 \\ 0 & 1 & -1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 1 & -1 \end{bmatrix}$$

$$b = \mathbf{0}, \quad \Gamma = \mathbf{0}, \quad d = 0$$

Dual optimization problem

$$\min_{\alpha,\alpha^*,\gamma,\mu} \quad \frac{1}{2}\Big[(\alpha-\alpha^*)^\top Q(\alpha-\alpha^*) + \gamma^\top AA^\top\gamma + \mu^\top\Gamma\Gamma^\top\mu + 2\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)\gamma^\top AX_{i:}$$

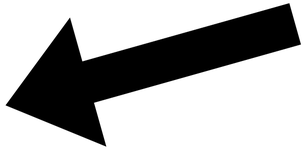
$$-2\sum_{i=1}^{n}(\alpha_i-\alpha_i^*)\mu^\top\Gamma X_{i:} - 2\gamma^\top A\Gamma^\top\mu\Big] + y^\top(\alpha-\alpha^*) + \gamma^\top b - \mu^\top d$$

$$\text{subject to} \quad 0 \le \alpha_i, \alpha_i^* \le \frac{C}{n}$$

$$\mathbf{e}^\top(\alpha+\alpha^*) = C\nu$$

$$\mathbf{e}^\top(\alpha-\alpha^*) = 0$$

$$\gamma_j \ge 0$$



Non-separable constraints
Not possible to use vanilla Coordinate descent

Classical SVR can be solved with Sequential Minimal Optimization algorithm [Platt '98]
It updates two coordinates at a time to keep the non-separable constraints verified at all time

Proposed algorithm and convergence result

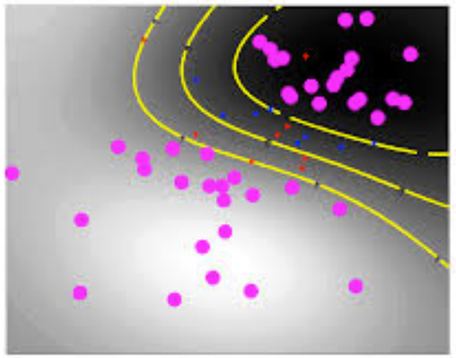
Generalized SMO :

- Perform SMO algorithm when updating the variables α or α^*
- Perform projected coordinate descent when updating the variables γ
- Perform smooth coordinate descent when updating the variables μ
- The block in which the update will happen is chosen based on the violation of the KKT conditions

Theorem: [Q.K. and Vaiter '21]

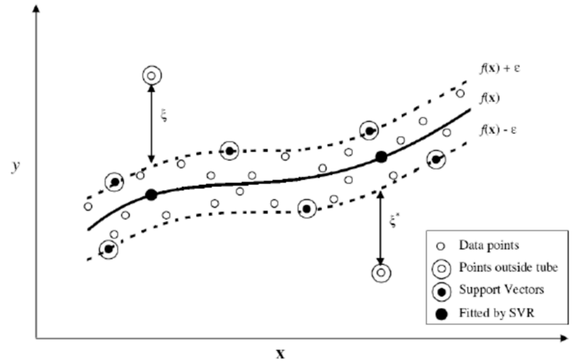
The sequence of iterates obtained by the generalized SMO converges towards a solution of the dual problem of the Linearly constrained SVR.

Hyperparameters in machine learning



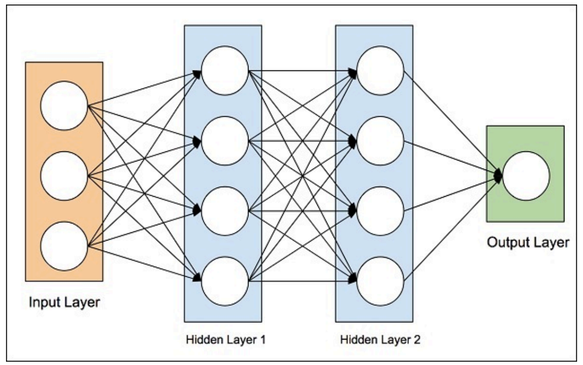
Support Vector Machine [Vapnik '92]

$C > 0$



ε-Support Vector Regression [Drucker et al. '96]

$C > 0, \varepsilon > 0$



Neural Networks

layers, learning rate

Regularized models:

Lasso [Tibshirani '96]

Ridge regression [Hoerl-Kennard '70]

Elastic net [Zou-Hastie '05]

Total variation [Rudin-Osher-Fatemi '92] …

Typically $\lambda > 0$

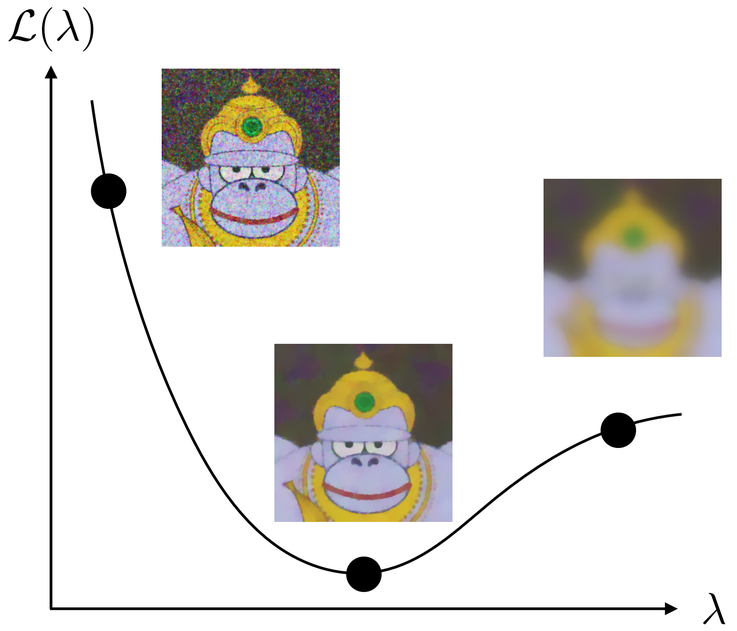
Selection criteria

Hyperparameter:

$$\lambda \in \mathbb{R}^r$$

Criterion:

$$\mathcal{L} : \mathbb{R}^r \to \mathbb{R}$$



Goal

Find:

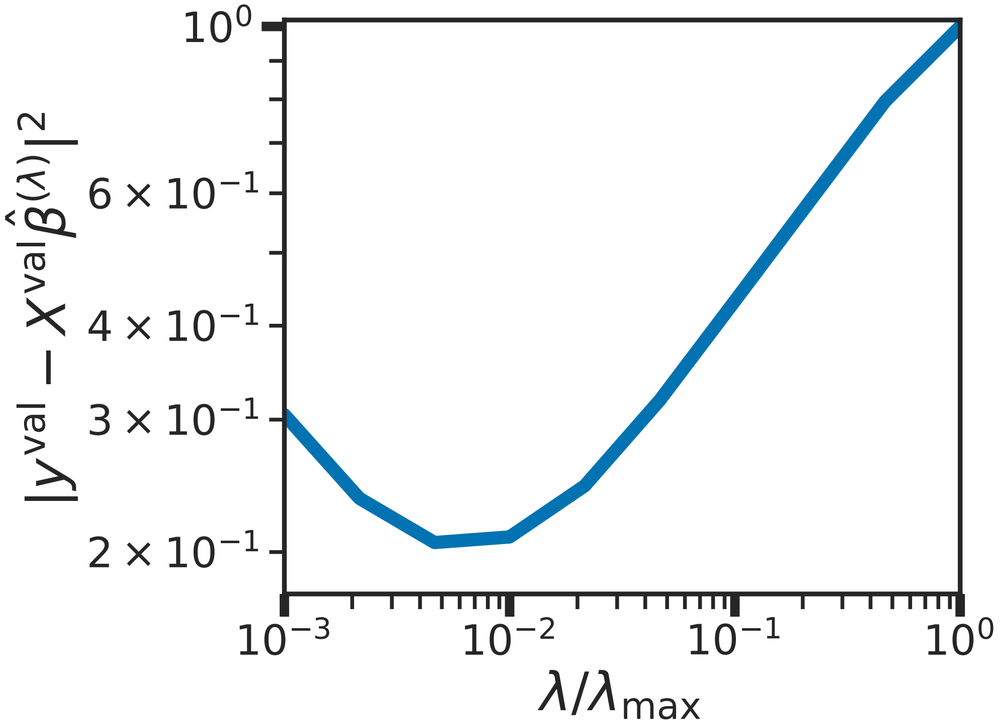
$$\lambda^\star \in \operatorname*{argmin}_{\lambda \in \mathbb{R}^r} \mathcal{L}(\lambda)$$

- Examples:
  cross-validation [Stone-Ramer ’65]
  AIC [Akaike ’74],
  BIC [Schwarz ’78],
  SURE [Stein ’81]

Hyperparameter optimization (HO)

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

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



Bilevel optimization problem

Grid-search as 0-order optimization method

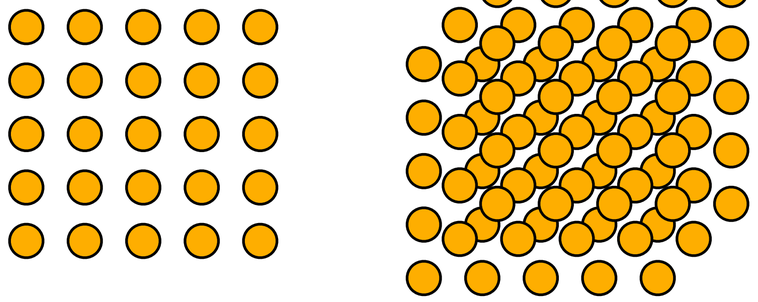
$$\underbrace{\arg\min_{\lambda \in \mathbb{R}} \left\{ \mathcal{L}(\lambda) := \|y^{\text{val}} - X^{\text{val}} \hat{\beta}^{(\lambda)}\|^2 \right\}}_{\text{outer optimization problem}}$$

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

Algorithm:

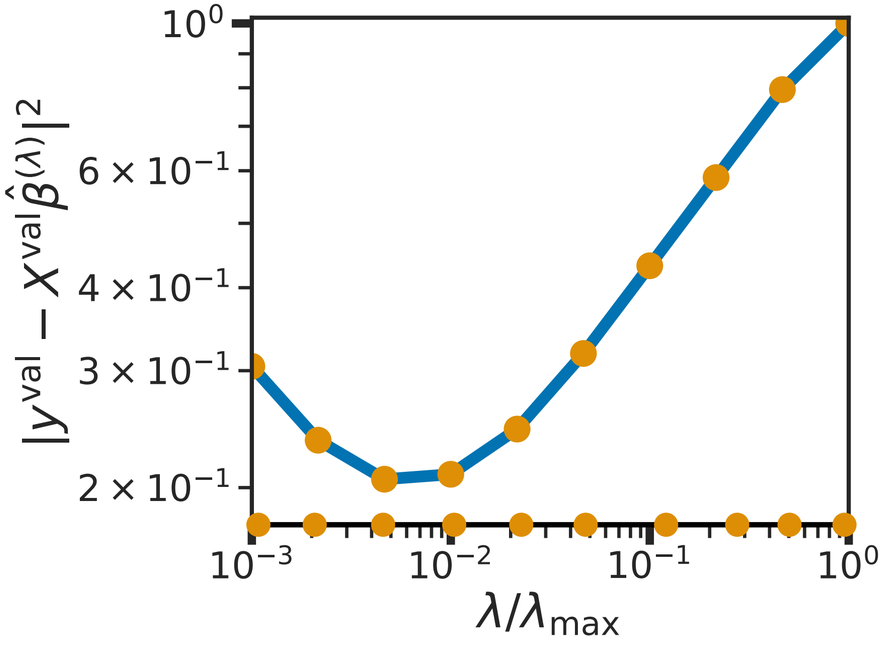
1. Choose a grid of values for λ
2. Evaluate $\mathcal{L}(\lambda)$ for each λ
3. Keep the best $\lambda^\star$

Dimension is a problem



2 parameters

3 parameters



Grid-search as 0-order optimization method

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

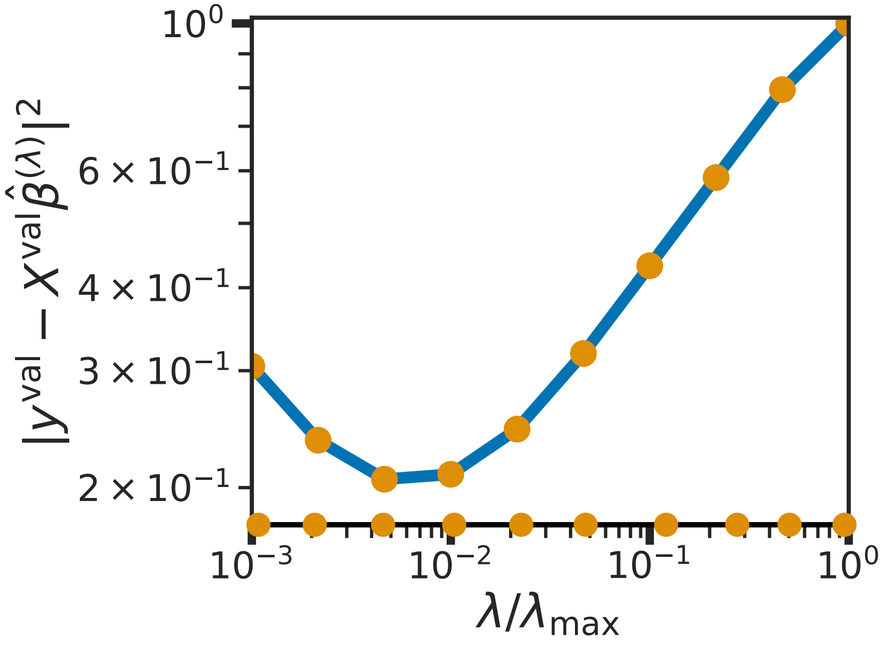
$$\text{s.t. } \underbrace{\hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} \|y^{\text{train}} - X^{\text{train}} \beta\|^2 + \lambda \|\beta\|_1}_{\text{inner optimization problem}}$$

Algorithm:
1. Choose a grid of values for λ
2. Evaluate $\mathcal{L}(\lambda)$ for each λ
3. Keep the best $\lambda^\star$

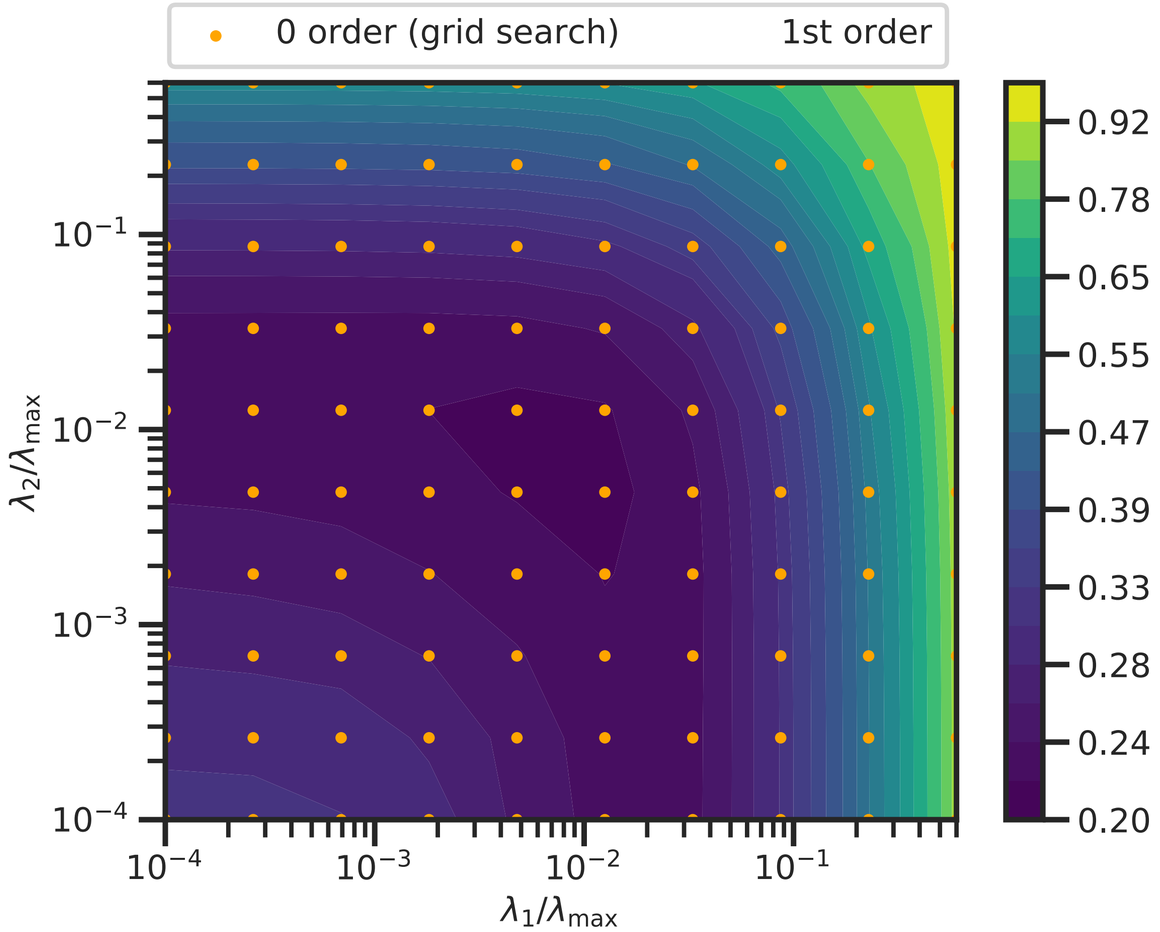
Idea:

If $\mathcal{L}$ is differentiable, use 1st order method

Algorithm:
1. Choose a starting point $\lambda^{(0)}$
2. Compute $\nabla \mathcal{L}(\lambda^{(t)})$
3. Update $\lambda^{(t+1)} = \lambda^{(t)} - \rho \nabla \mathcal{L}(\lambda^{(t)})$
4. Repeat 2/3 until convergence

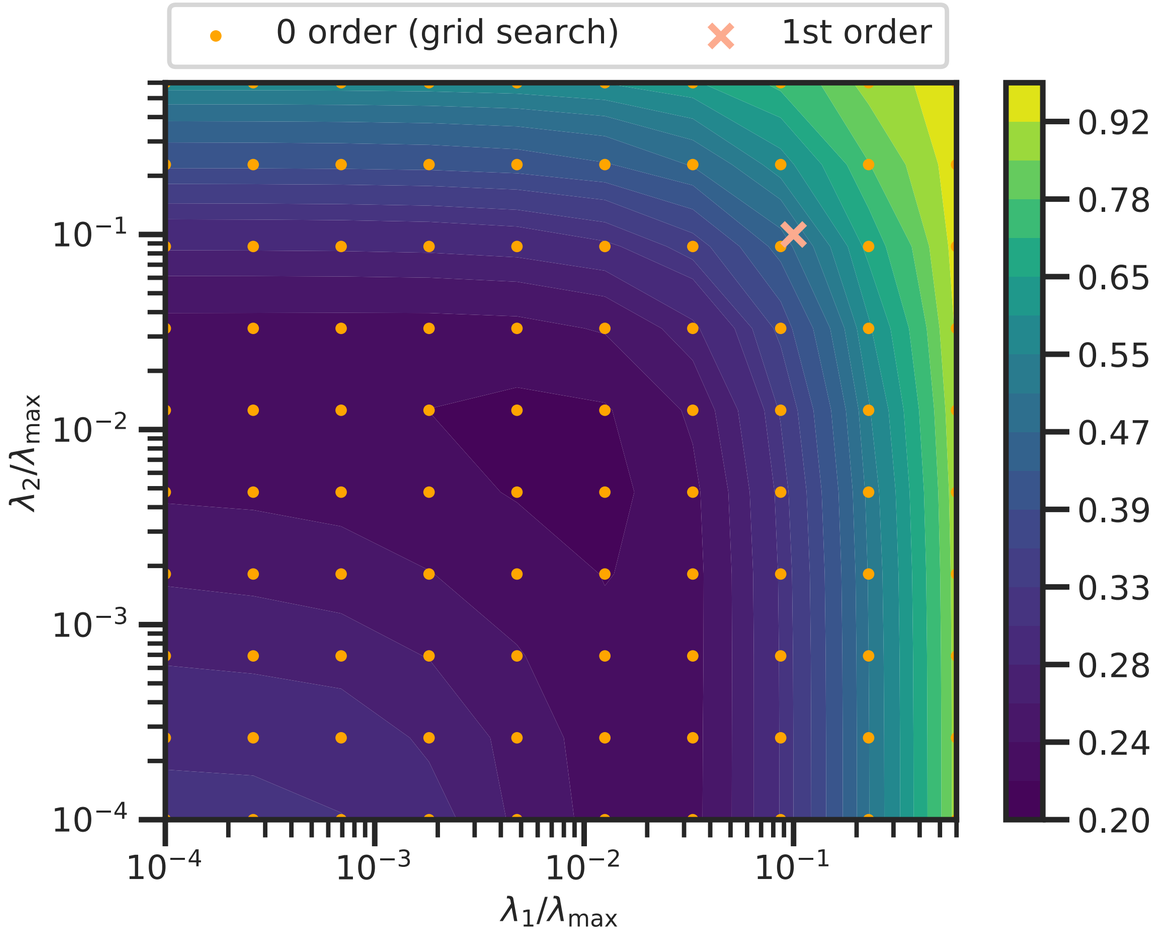


1st-order optimization method



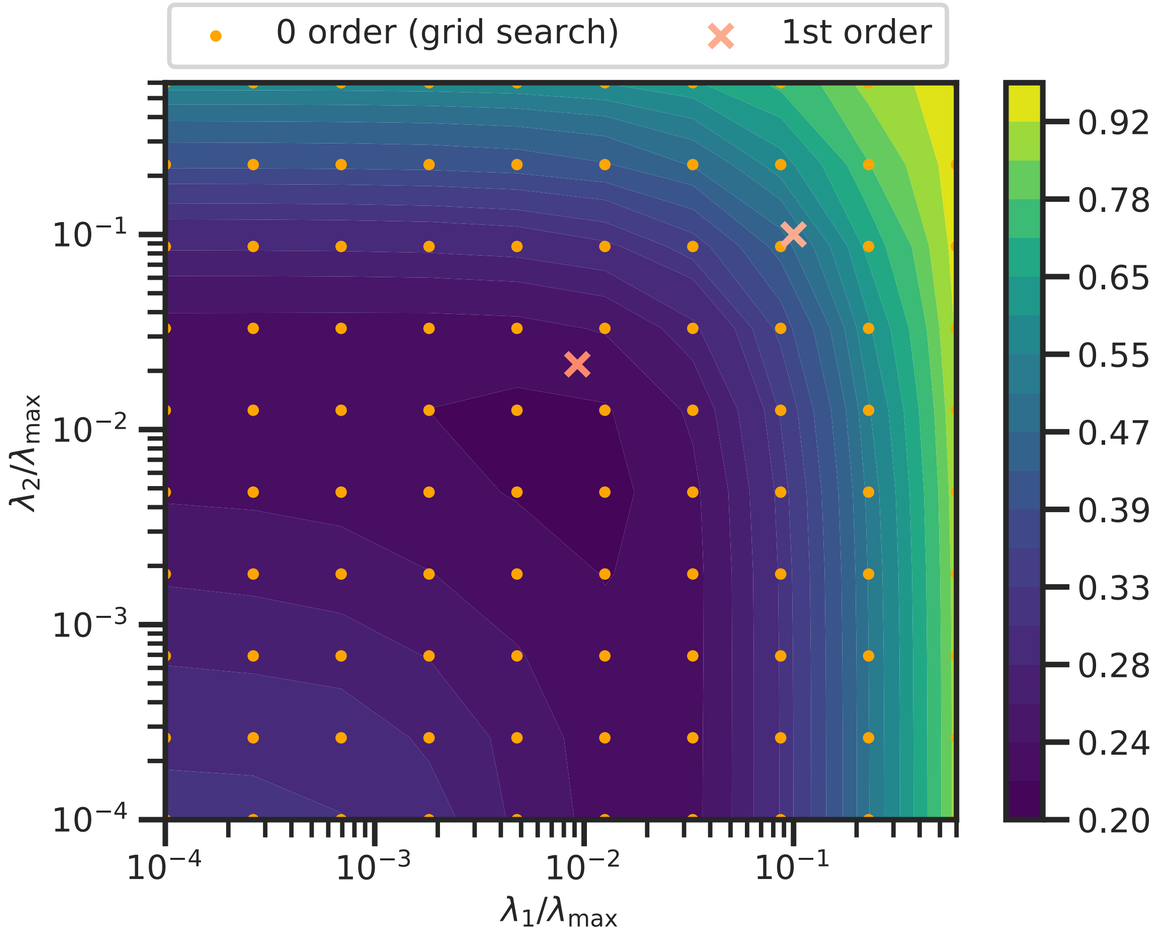
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



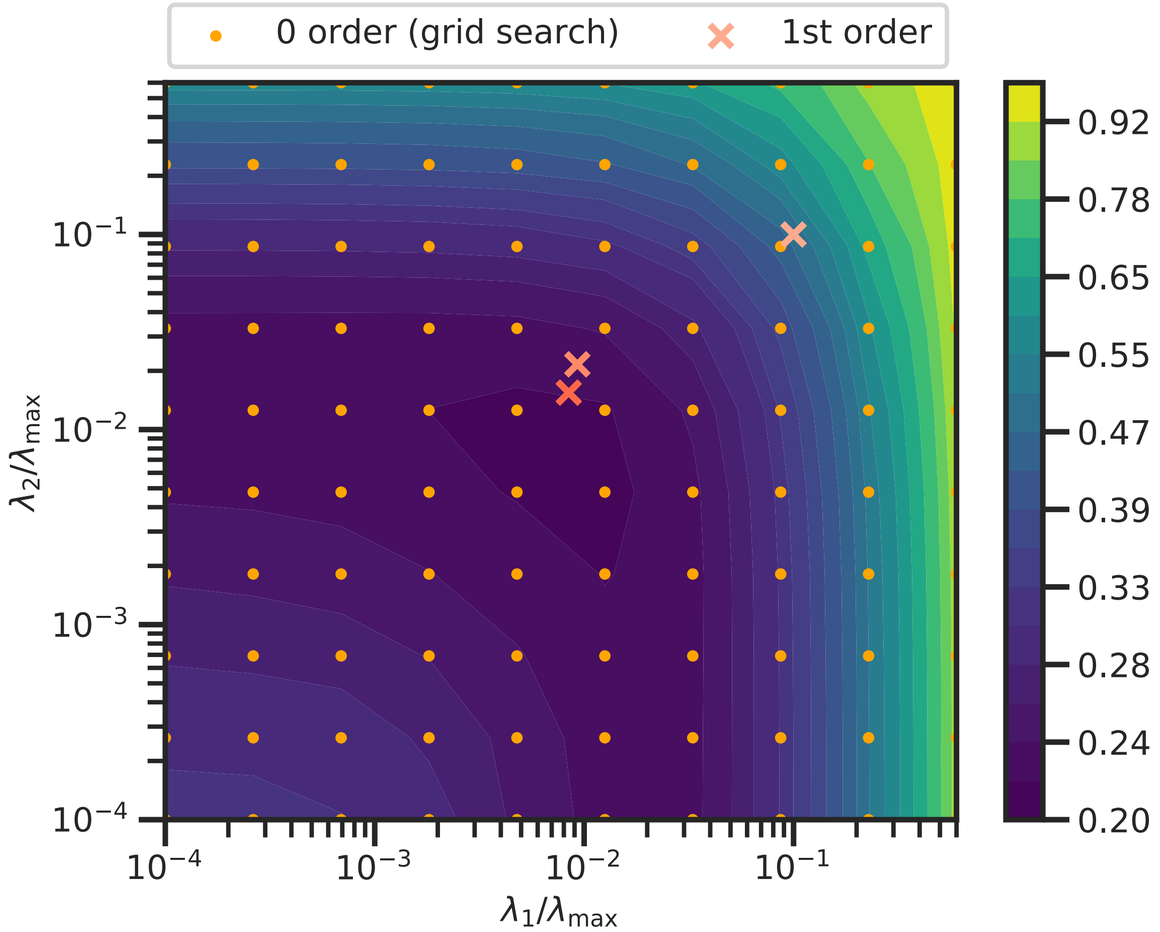
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



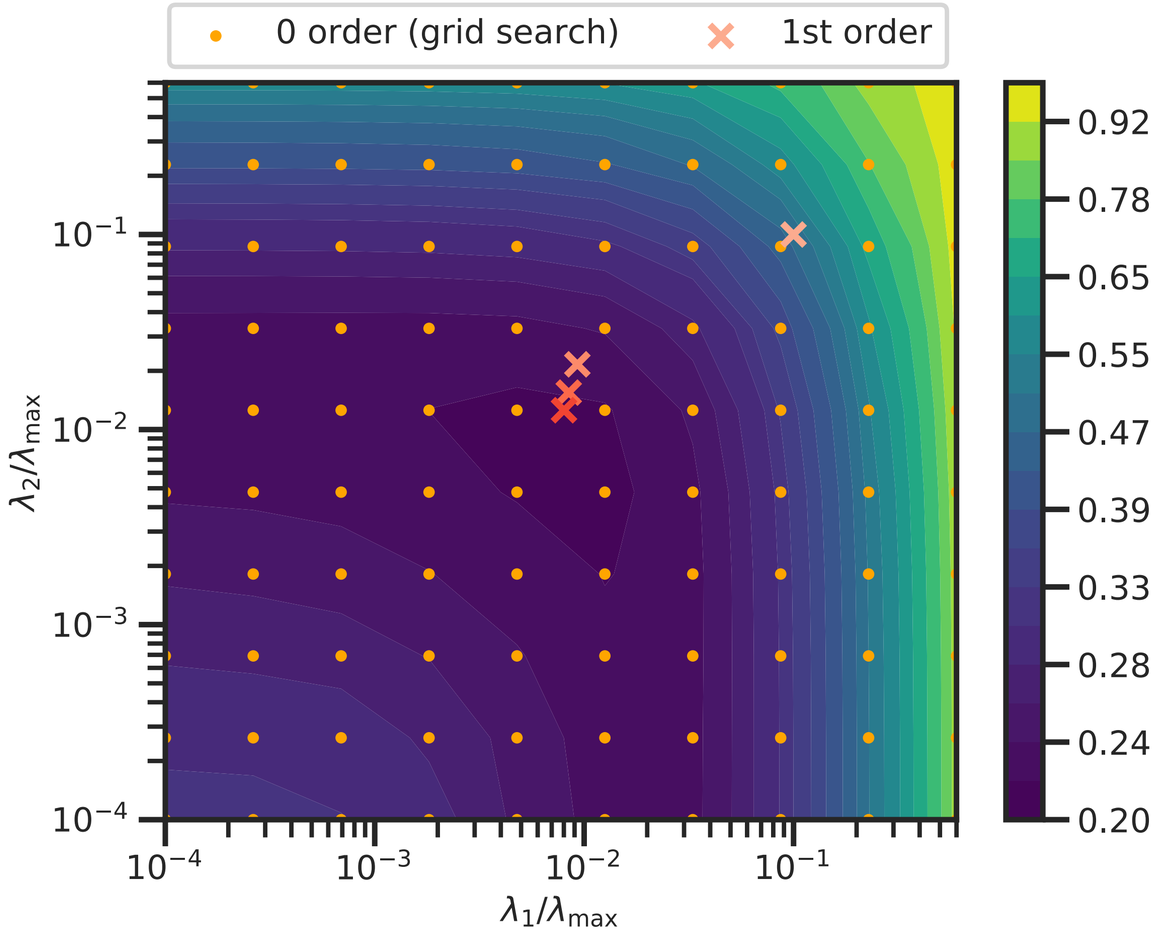
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



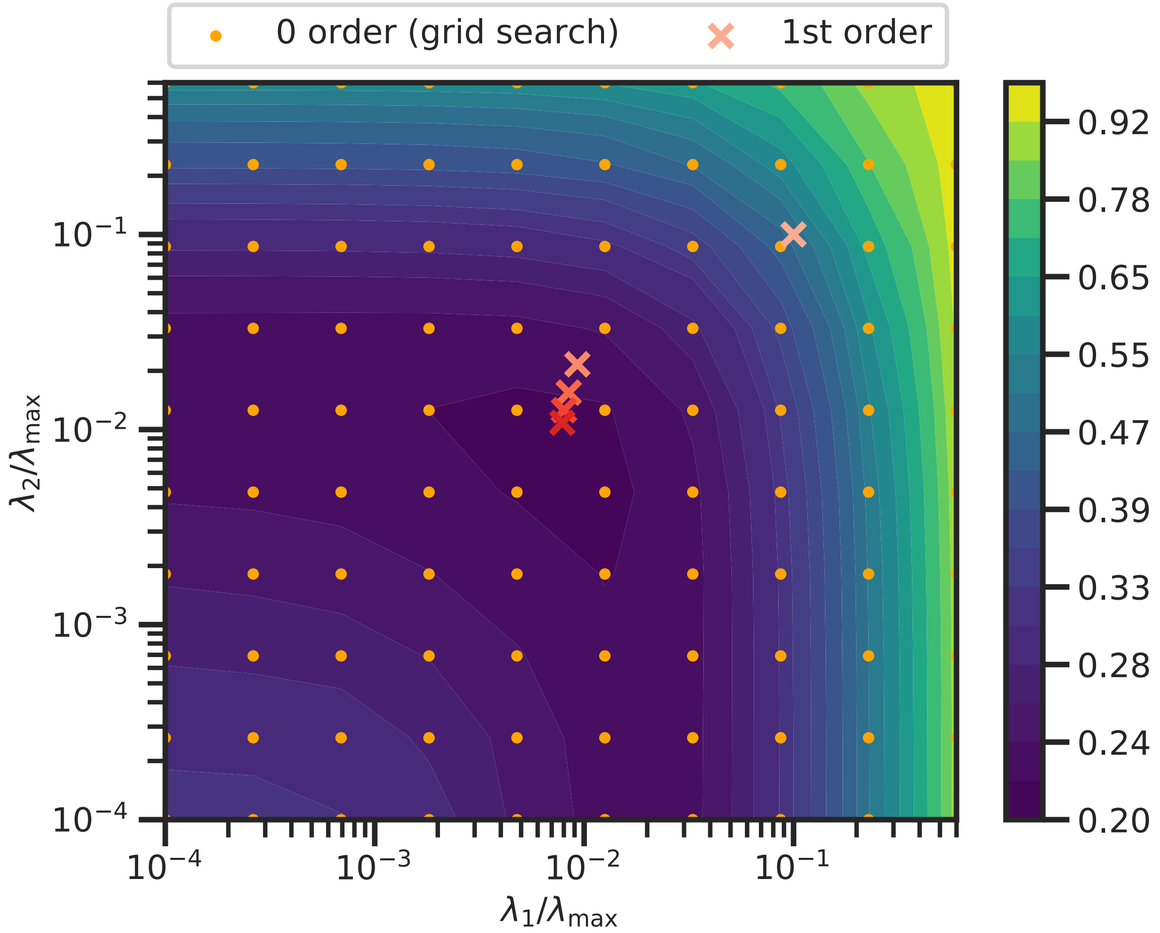
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



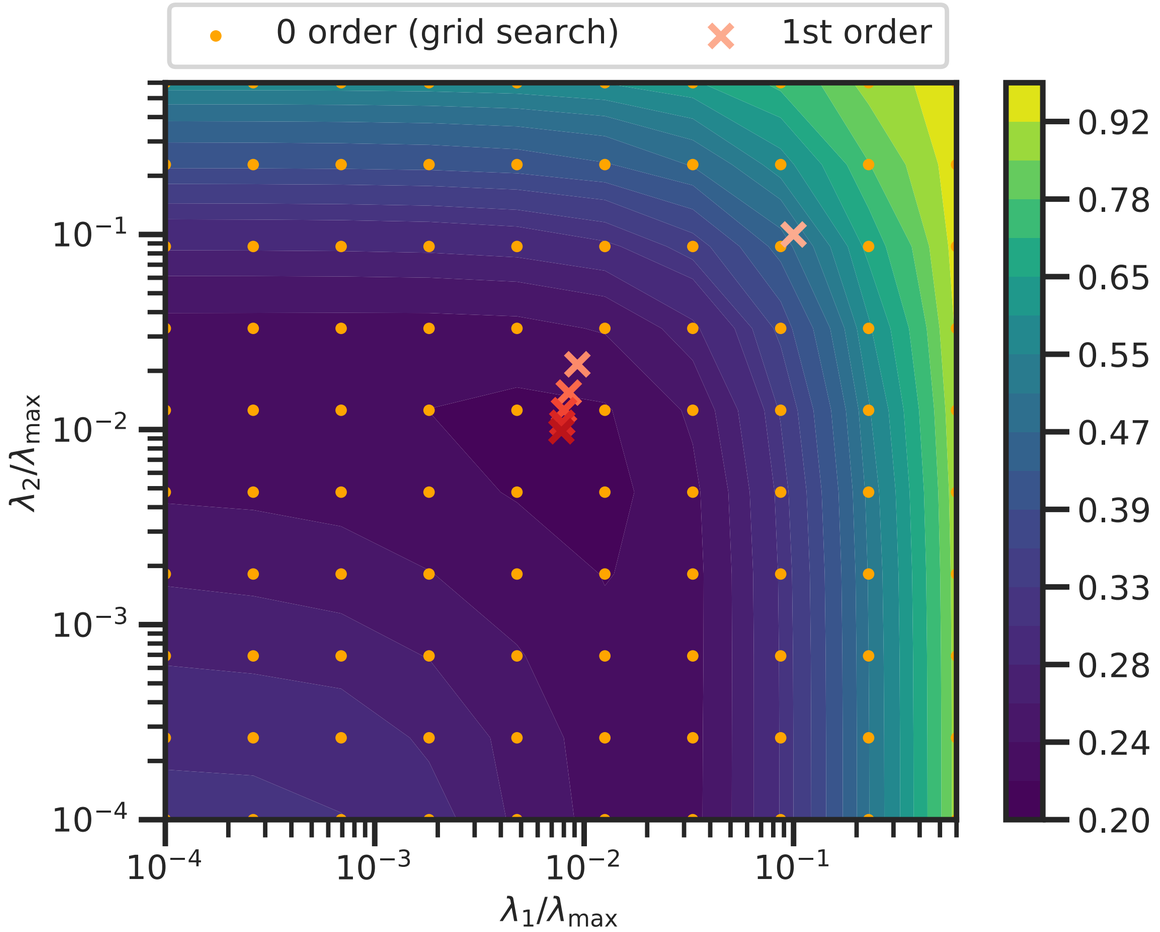
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



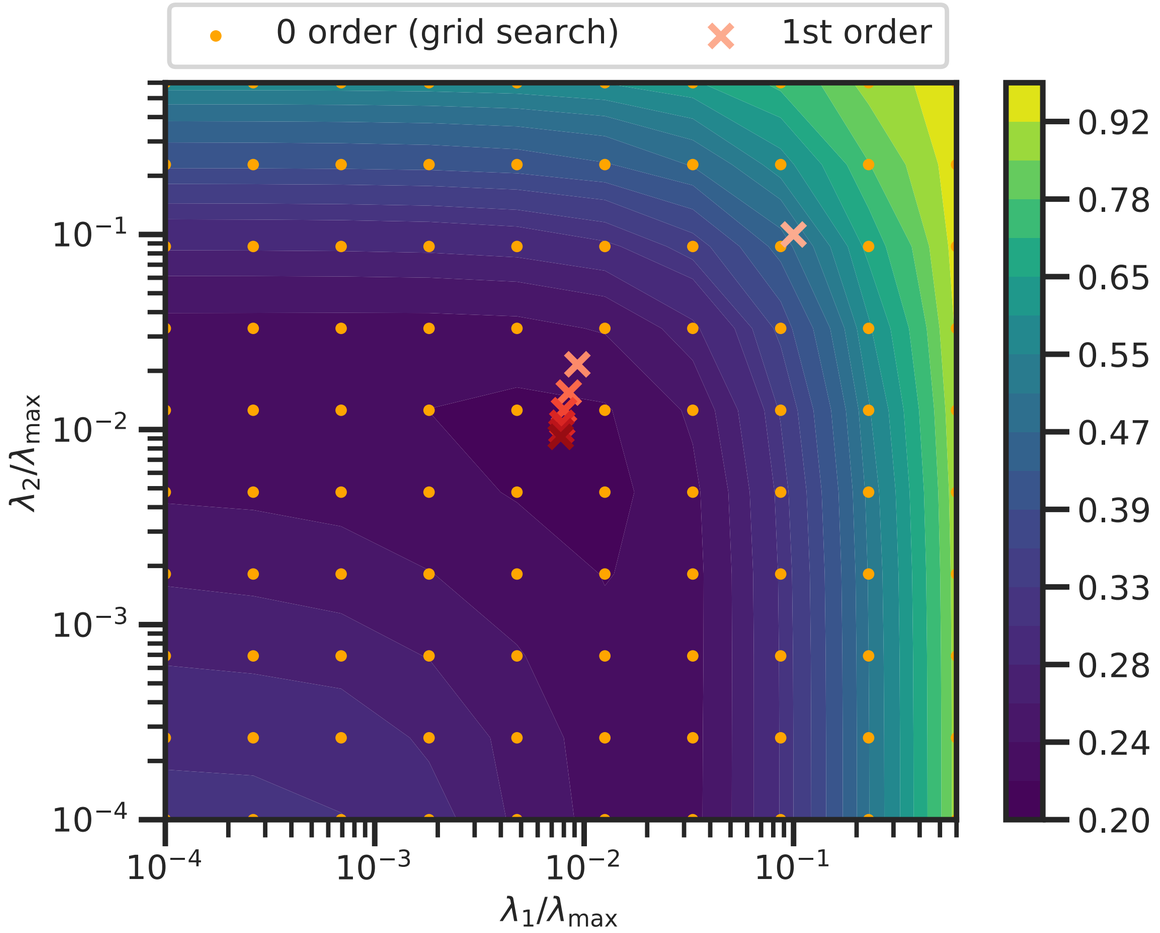
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



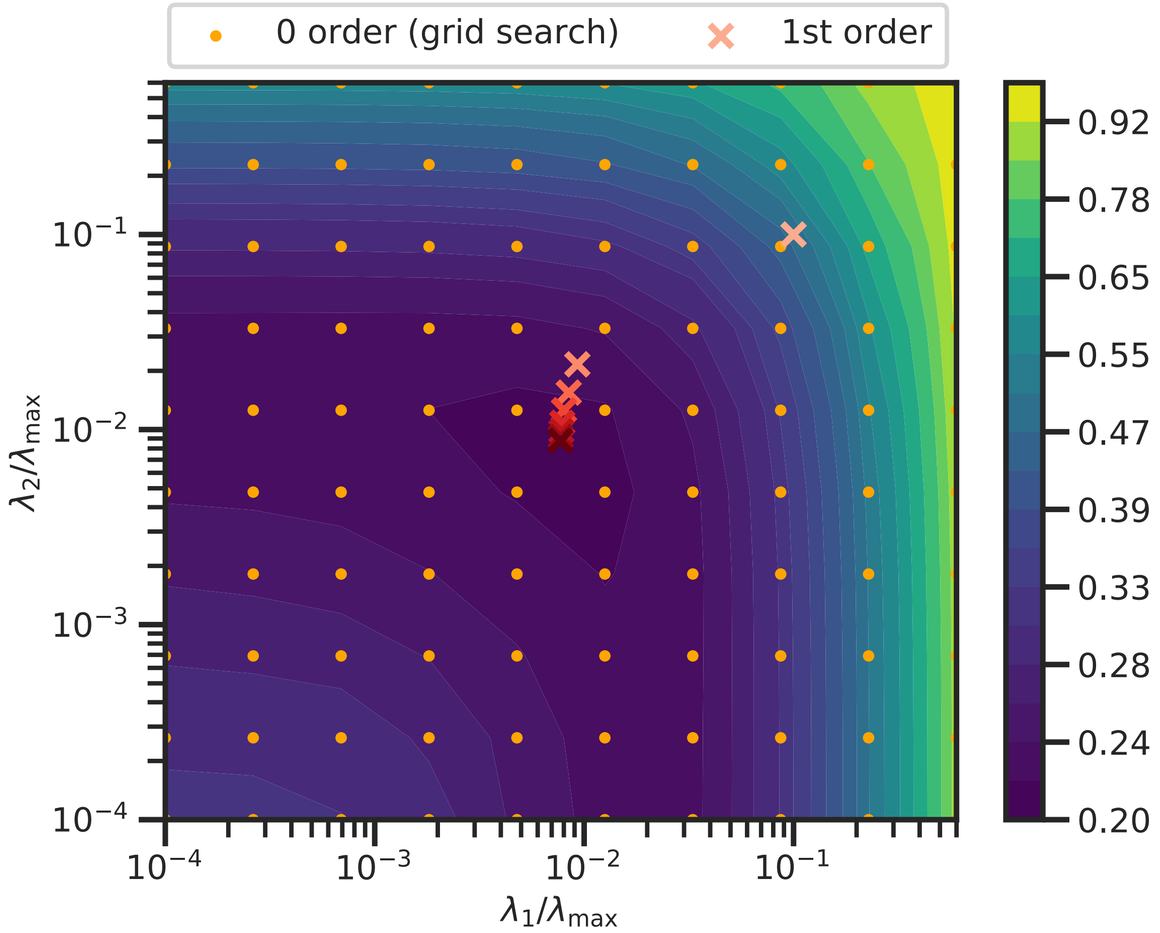
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



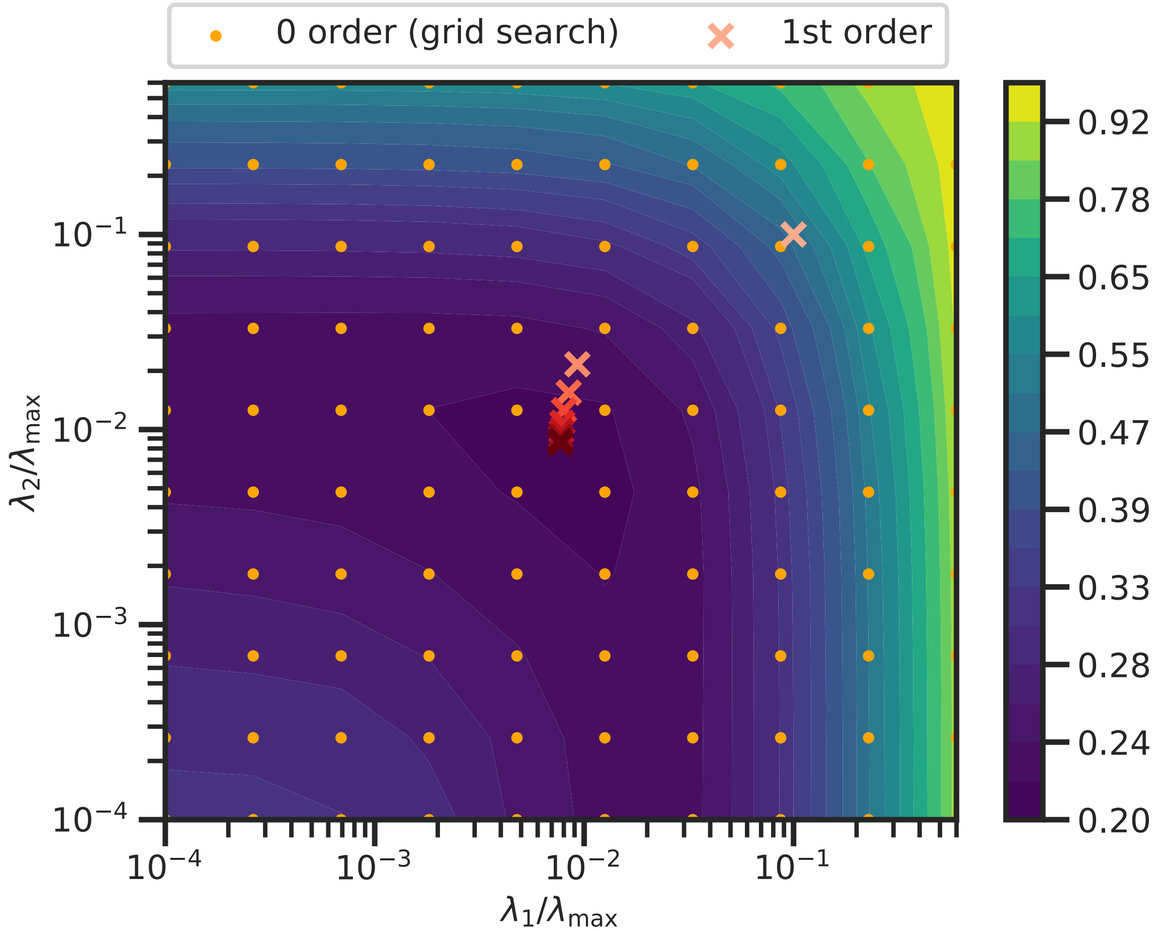
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



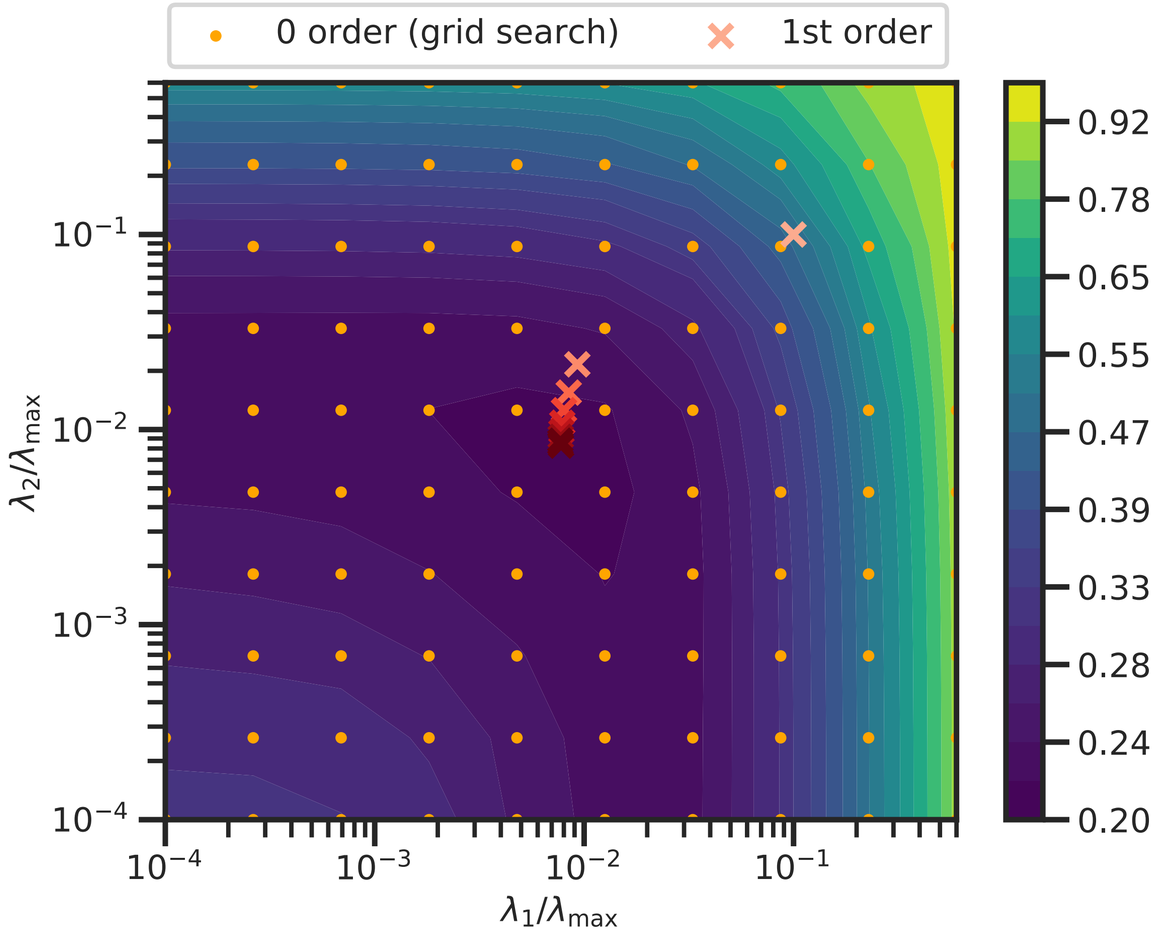
Elastic net: $\arg\min_\beta \frac{1}{2n}\|y^{\text{train}} - X^{\text{train}}\beta\|^2 + \lambda_1\|\beta\|_1 + \frac{\lambda_2}{2}\|\beta\|^2$

1st-order optimization method



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

1st-order optimization method



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

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

$$\arg\min_{\lambda \in \mathbb{R}^r} \left\{ \mathcal{L}(\lambda) := C(\hat{\beta}^{(\lambda)}) \right\}$$

$$\text{s.t. } \hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \Phi(\beta, \lambda)$$

The chain rule gives:

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

Related work: If inner problem is regular enough

- Implicit differentiation [Larsen et al. '96]-[Bengio '00]

$$\nabla_\beta \Phi(\hat{\beta}^{(\lambda)}, \lambda) = 0$$

Differentiating with respect to λ

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

$$\hat{\mathcal{J}}_{(\lambda)}^\top = -\nabla^2_{\beta,\lambda} \Phi\left(\hat{\beta}^{(\lambda)}, \lambda\right) \underbrace{\left(\nabla^2_\beta \Phi(\hat{\beta}^{(\lambda)}, \lambda)\right)^{-1}}_{\in \mathbb{R}^{p \times p}}$$

- Iterative differentiation:
  -Forward [Wengert '64]
  -Backward [Linnainmaa '70]

Implicit differentiation

$f(x, \lambda)$, $\partial_1 f$ derivative w.r.t. the first variable
and $\partial_2 f$ w.r.t. the second variable

$$\hat{\beta}^{(\lambda)} \in \underset{\beta \in \mathbb{R}^p}{\arg\min}\, \Phi(\beta, \lambda) := f(\beta) + \sum_{j=1}^{p} g_j(\beta_j, \lambda)$$

Theorem: *(Under a set of assumptions)* [Bertrand, Q.K. et al. '21]

Let $\hat{z} = \hat{\beta} - \gamma \nabla f(\hat{\beta})$, the Jacobian of the inner problem is obtained by solving the linear system:

$$A\hat{\mathcal{J}}_{\hat{S}} + B = 0$$

with $A := \mathrm{Id}_{|\hat{S}|} - \partial_1 \mathrm{prox} \gamma g(\hat{z})_{\hat{S}} \left(\mathrm{Id}_{\hat{S}} - \gamma \nabla^2_{\hat{S},\hat{S}} f(\hat{\beta}) \right)$

$$B := \partial_2 \mathrm{prox}_{\gamma g}(\hat{z})_{\hat{S}} - \gamma \partial_1 \mathrm{prox}_{\gamma g}(\hat{z})_{\hat{S}} \nabla^2_{\hat{S},\hat{S}^c} f(\hat{\beta}) \hat{\mathcal{J}}_{\hat{S}^c}$$

$$\hat{\mathcal{J}}_{\hat{S}^c} = \partial_2 \mathrm{prox}_{\gamma g}\left(\hat{z}\right)_{\hat{S}^c}$$

Implicit differentiation

[Gribonval and Nikolova '20]

Remarks:

- The proof relies on the differentiation of $\hat{\beta}^{(\lambda)} = \mathrm{prox}_{\gamma g}\left(\hat{\beta}^{(\lambda)} - \gamma \nabla f(\hat{\beta}^{(\lambda)})\right)$ [Bareilles et al. '20]

- The assumptions ensure that we can **differentiate the fixed point equation**

- To be usable, we need to **ensure support identification**, it is the case for proximal gradient decent [Liang et al. '15] and proximal coordinate descent [Q.K. et al. '20]

- The system to solve has size $|\hat{S}| \times |\hat{S}|$ instead of $p \times p$ (for sparse model) $p \gg |\hat{S}|$

- For the Lasso: [Dossal et al. '13]

$$\mathcal{J}_{\hat{S}} = n(X_{:\hat{S}}^\top X_{:\hat{S}})^{-1}\mathrm{sign}(\hat{\beta}_{\hat{S}})$$

$$\hat{\mathcal{J}}_{\hat{S}^c} = 0$$

Same sparsity as the solution

Proposed algorithm

Algorithm IMPLICIT FORWARD DIFF

input : $X \in \mathbb{R}^{n \times p}, y \in \mathbb{R}^n, \lambda \in \mathbb{R}, n_{\text{iter}} \in \mathbb{N}, n_{\text{jac}} \in \mathbb{N}$

$\beta = 0$; // potentially warm started

; // Compute solution of inner problem

Obtain solution $\hat{\beta} \in \arg\min_{\beta \in \mathbb{R}^p} f(\beta) + \sum_{i=1}^{p} g_j(\beta_j, \lambda)$ and its support $\hat{S}$

$\hat{z} = \hat{\beta} - \gamma \nabla f(\hat{\beta})$

; // Compute the Jacobian

$\mathcal{J}_{\hat{S}^c} = \partial_2 \text{prox}_{\gamma g}(\hat{z})_{\hat{S}^c}$

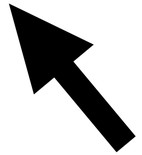
for $k = 1, \ldots, n_{jac}$ **do**

  for $j \in \hat{S}$ **do**

$$\mathcal{J}_j = \partial_1 \text{prox}_{\gamma_j g_j}(\hat{z}_j) \left(\mathcal{J}_j - \gamma_j \nabla^2_{j,\hat{S}} f(\hat{\beta}) \mathcal{J}_{\hat{S}} \right)$$

$$\mathcal{J}_j += \partial_2 \text{prox}_{\gamma_j g_j}(\hat{z}_j) - \gamma \partial_1 \text{prox}_{\gamma_j g_j}(\hat{z}_j) \nabla^2_{j,\hat{S}^c} f(X\hat{\beta}) \mathcal{J}_{\hat{S}^c}$$

return $\beta^{n_{iter}}, \mathcal{J}^{n_{jac}}$



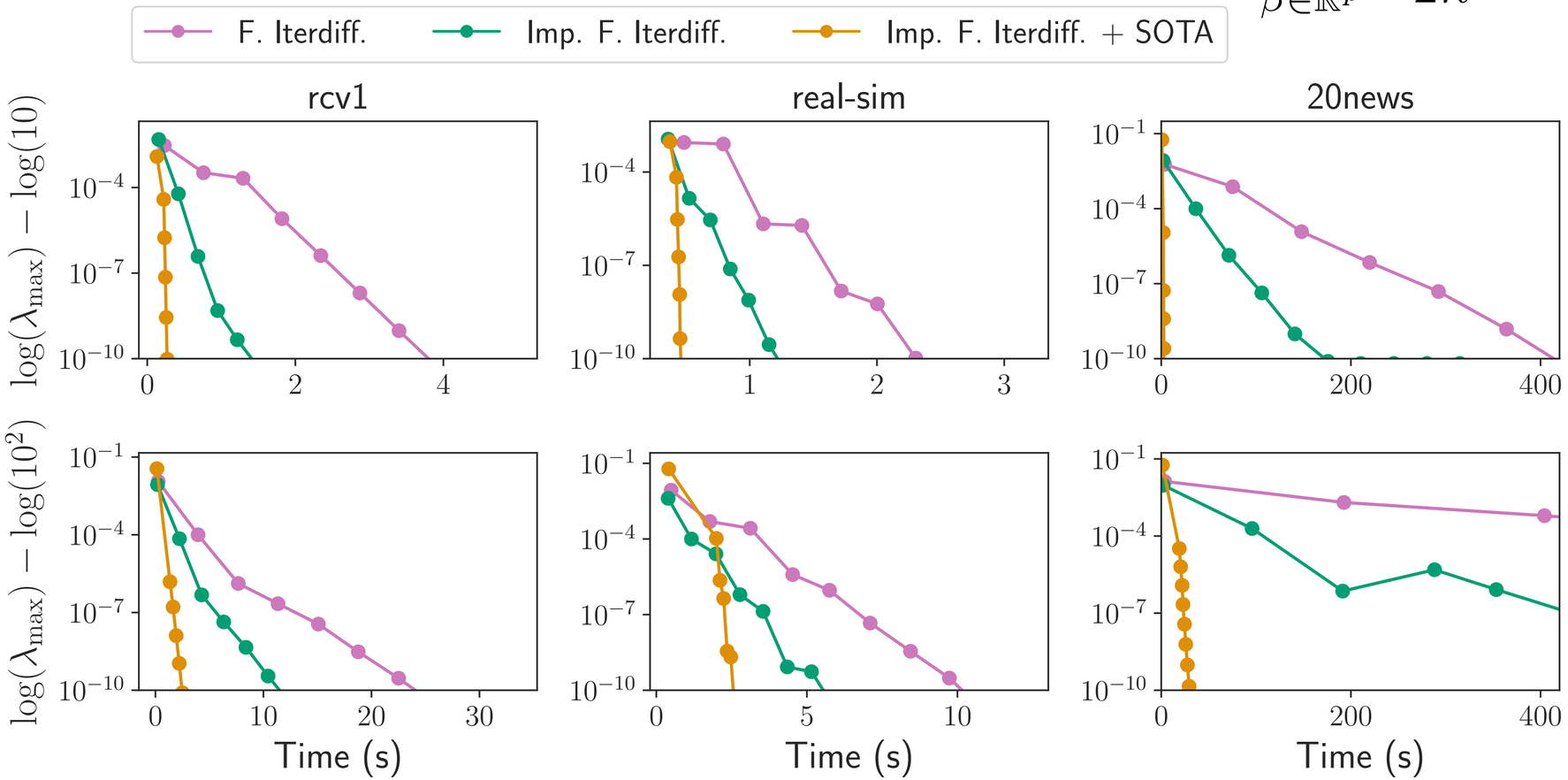
Possible to use solver of choice

”Hyper”-gradient computation time

Model: Lasso
Criterion: hold-out loss

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

$$\text{s.t } \hat{\beta}^{(\lambda)} \in \arg\min_{\beta \in \mathbb{R}^p} \frac{1}{2n} ||y^{\text{train}} - X^{\text{train}} \beta||_2^2 + \lambda ||\beta||_1$$



F. Iterdiff. : Forward differentiation

[Deledalle et al. ’14]

Imp. : Proposed algorithm with vanilla CD

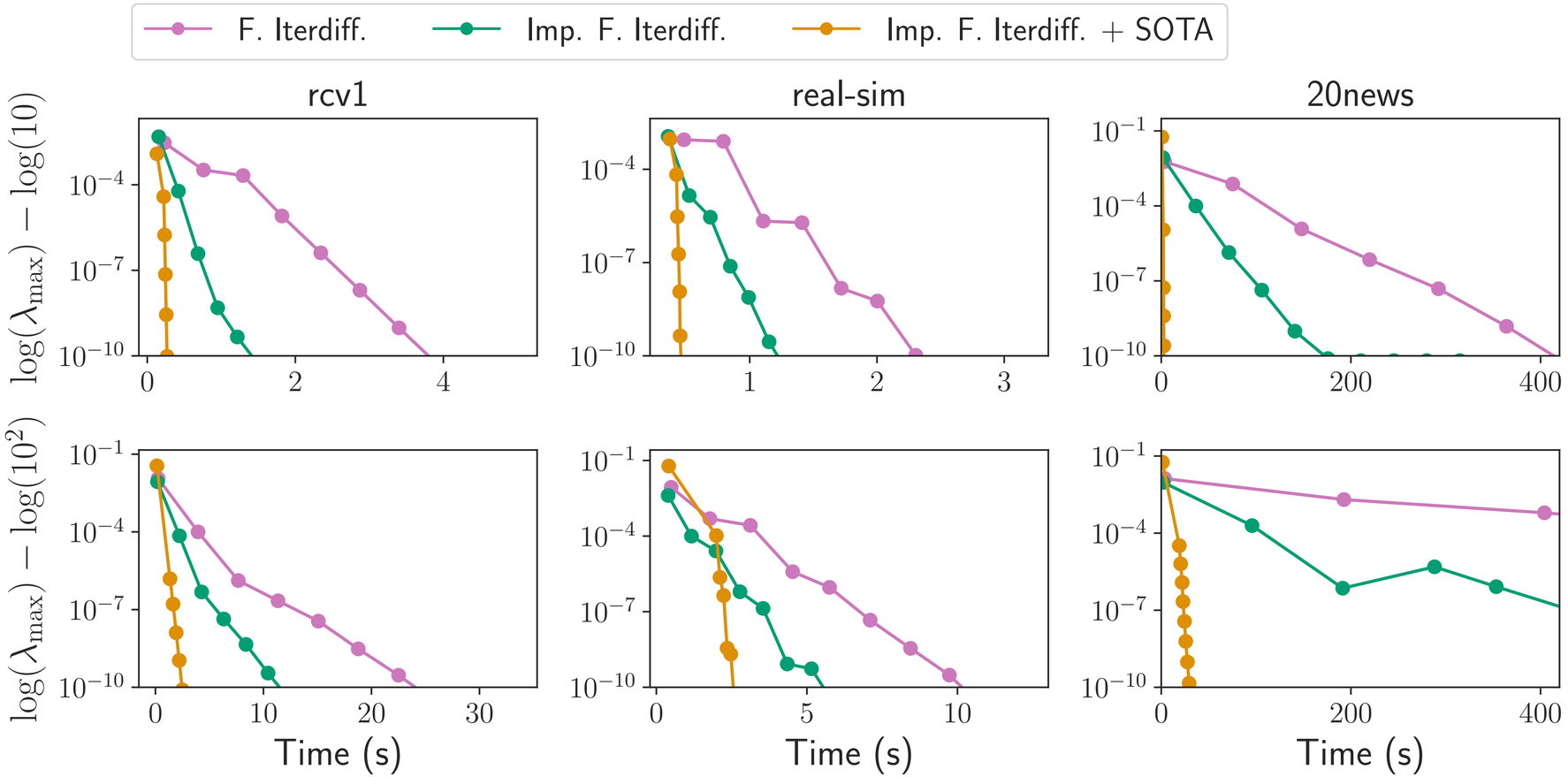
SOTA: Acceleration from Celer

[Massias et al. ’20]

”Hyper”-gradient computation time

Model: Lasso
Criterion: hold-out loss

name	# samples n	# features p
rcv1	$20,242$	$19,960$
real-sim	$72,309$	$20,958$
20news	$19,996$	$632,983$



F. Iterdiff. : Forward differentiation

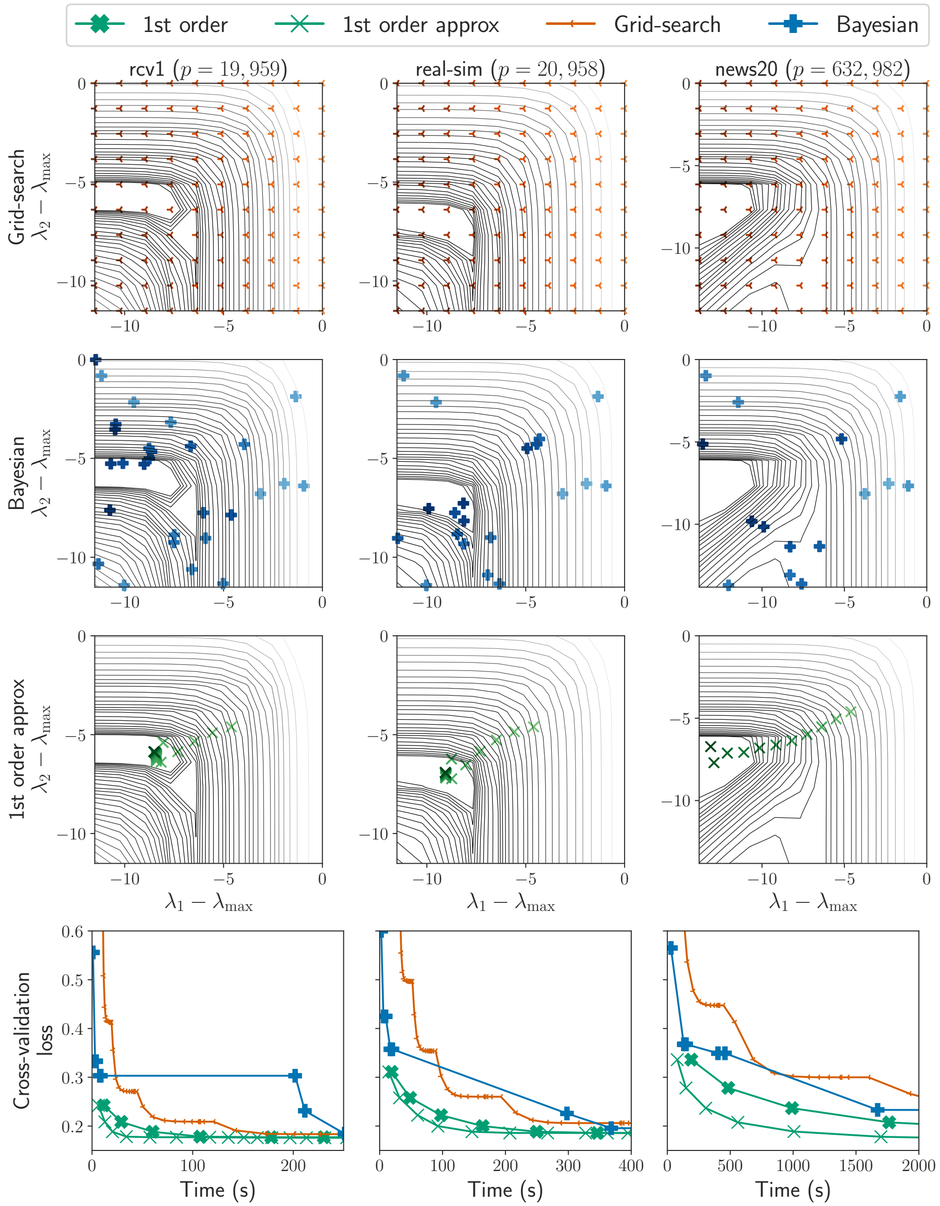
[Deledalle et al. ’14]

Imp. : Proposed algorithm with vanilla CD

SOTA: Acceleration from Celer

[Massias et al. ’20]

Hyperparameter selection: Elastic net



name	# samples n	# features p
rcv1	$20,242$	$19,960$
real-sim	$72,309$	$20,958$
20news	$19,996$	$632,983$

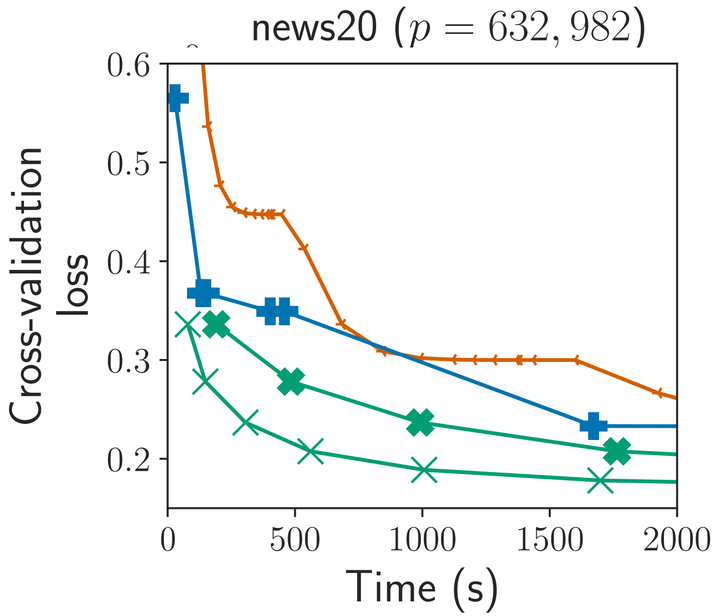
Model: Elastic net

Criterion: Cross validation 5-folds

$$\underset{\lambda=(\lambda_1,\lambda_2)\in\mathbb{R}^2}{\arg\min} \mathcal{L}(\lambda) = \frac{1}{n_{\text{fold}}} \sum_{i=1}^{n_{\text{fold}}} \|y^{\text{val}_i} - X^{\text{val}_i} \hat{\beta}^{(\lambda,i)}\|_2^2$$

$$\text{s.t.} \hat{\beta}^{(\lambda,i)} \in \underset{\beta\in\mathbb{R}^p}{\arg\min} \frac{1}{2n} \|y^{\text{train}_i} - X^{\text{train}_i} \beta\|_2^2 + \lambda_1 \|\beta\|_1 + \frac{1}{2}\lambda_2 \|\beta\|_2$$

Hyperparameter selection: Elastic net



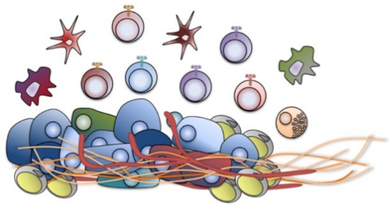
name	# samples n	# features p
rcv1	$20,242$	$19,960$
real-sim	$72,309$	$20,958$
20news	$19,996$	$632,983$

Model: Elastic net
Criterion: Cross validation 5-folds

$$\underset{\lambda=(\lambda_1,\lambda_2)\in\mathbb{R}^2}{\arg\min}\ \mathcal{L}(\lambda) = \frac{1}{n_{\text{fold}}}\sum_{i=1}^{n_{\text{fold}}} \|y^{\text{val}_i} - X^{\text{val}_i}\hat{\beta}^{(\lambda,i)}\|_2^2$$

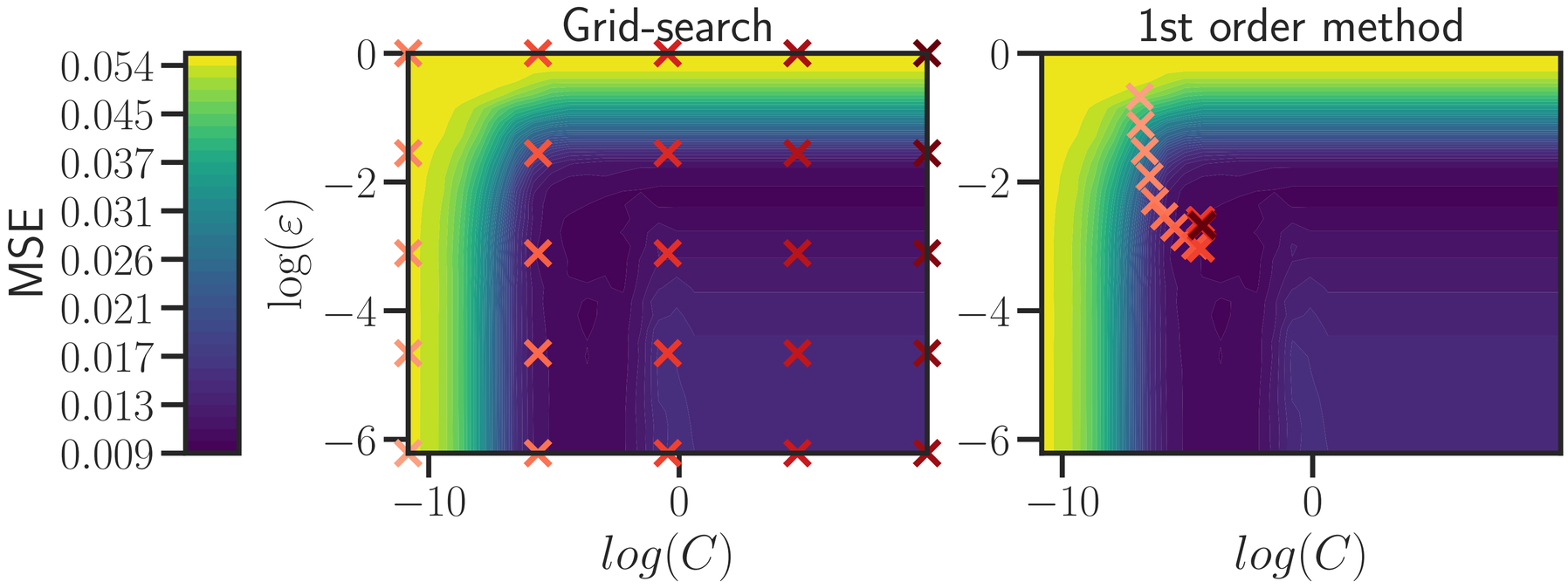
$$\text{s.t.}\ \hat{\beta}^{(\lambda,i)} \in \underset{\beta\in\mathbb{R}^p}{\arg\min}\ \frac{1}{2n}\|y^{\text{train}_i} - X^{\text{train}_i}\beta\|_2^2 + \lambda_1\|\beta\|_1 + \frac{1}{2}\lambda_2\|\beta\|_2$$

A new method for the estimation of cells

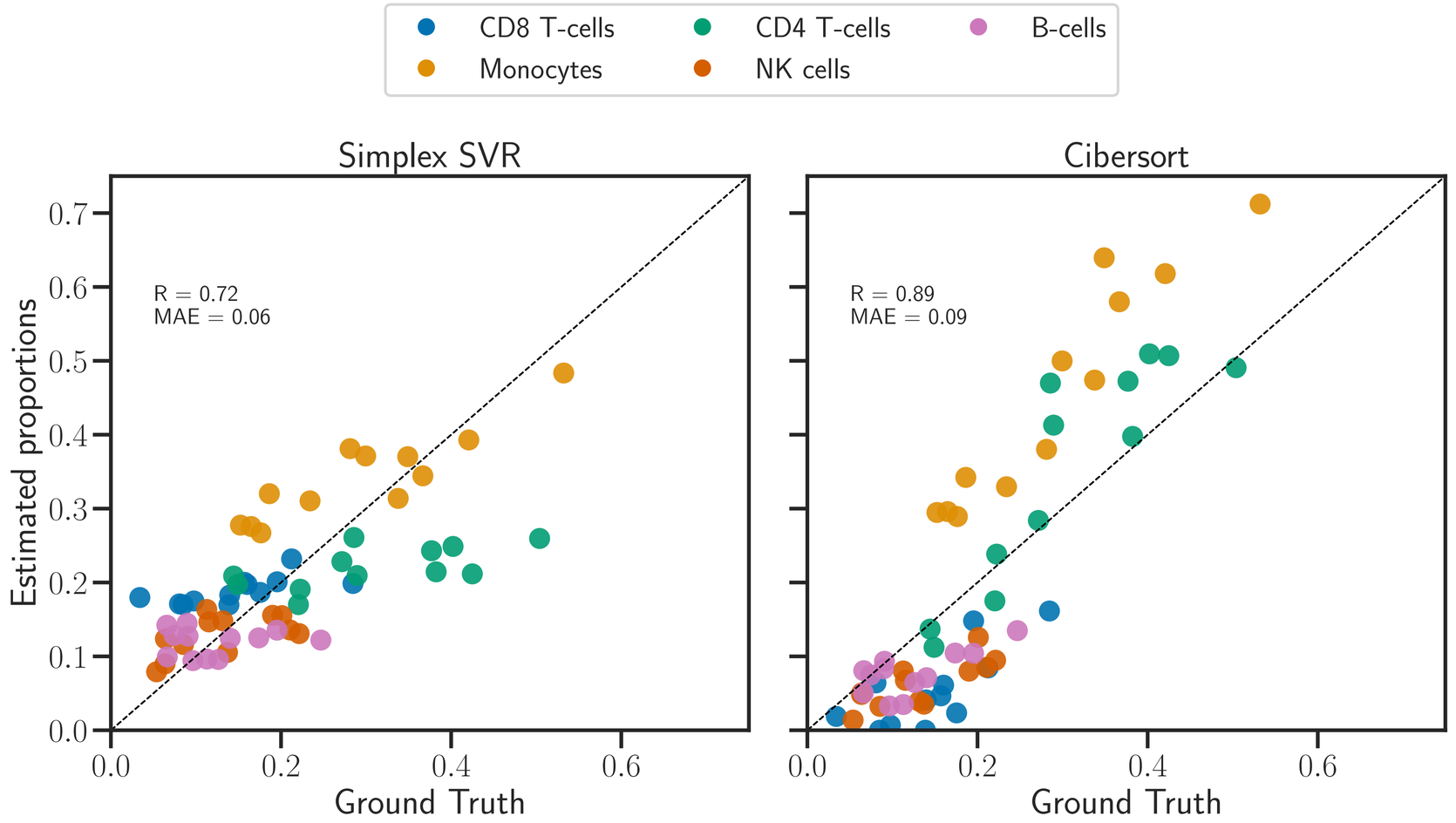


$$\underset{n \approx 10000}{\left[y \right]} = \underset{p \approx 25}{\left[X \right]} \left[\beta \right] + \epsilon$$

- Use Simplex ε - SVR without bias (it is separable) solved via coordinate descent
- Select optimal hyperparameters using implicit differentiation and Mean Squared Error as criterion

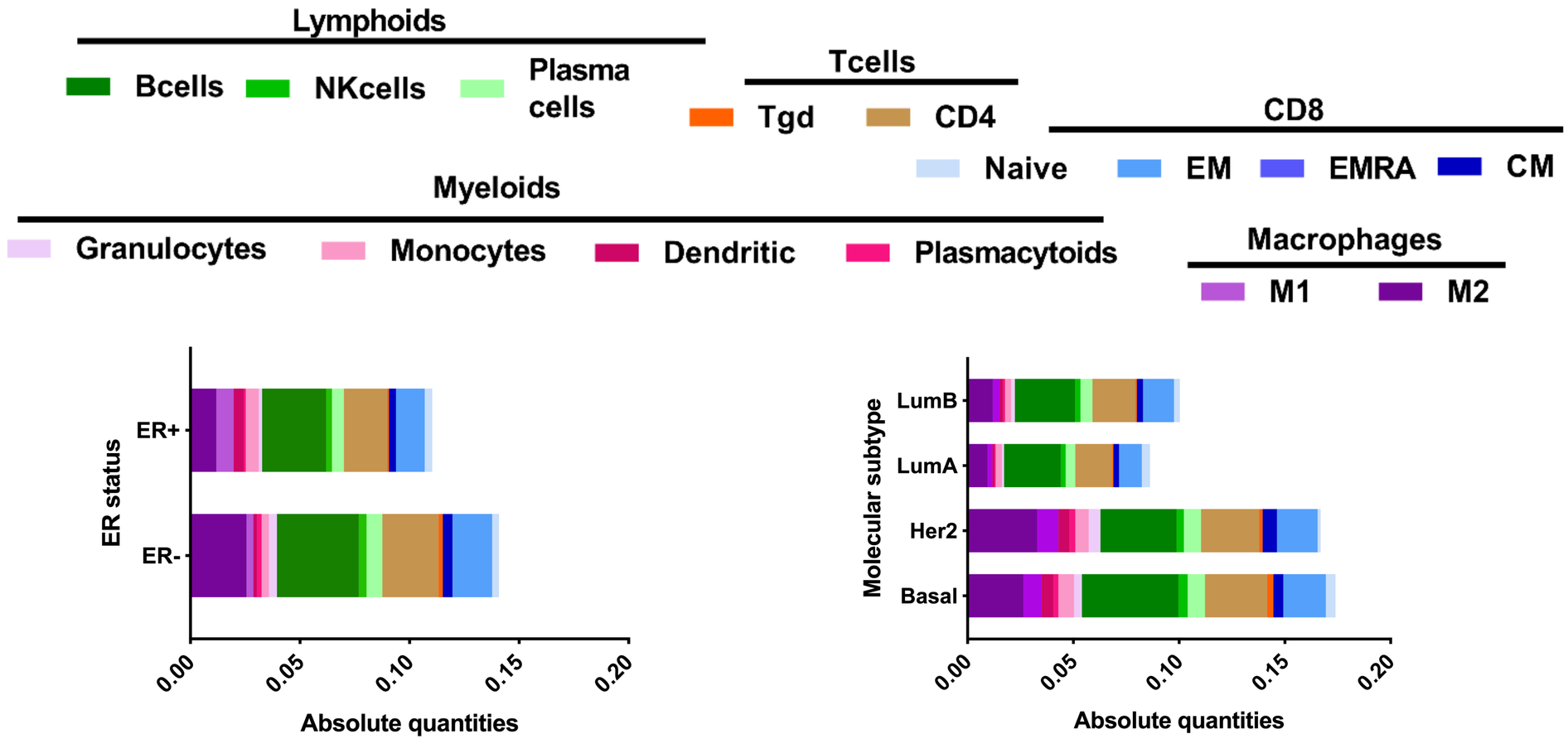


Comparison on sc-RNAseq/RNAseq



Peripheral blood mononuclear cell samples (GSE127813)

Immune cells in breast cancer

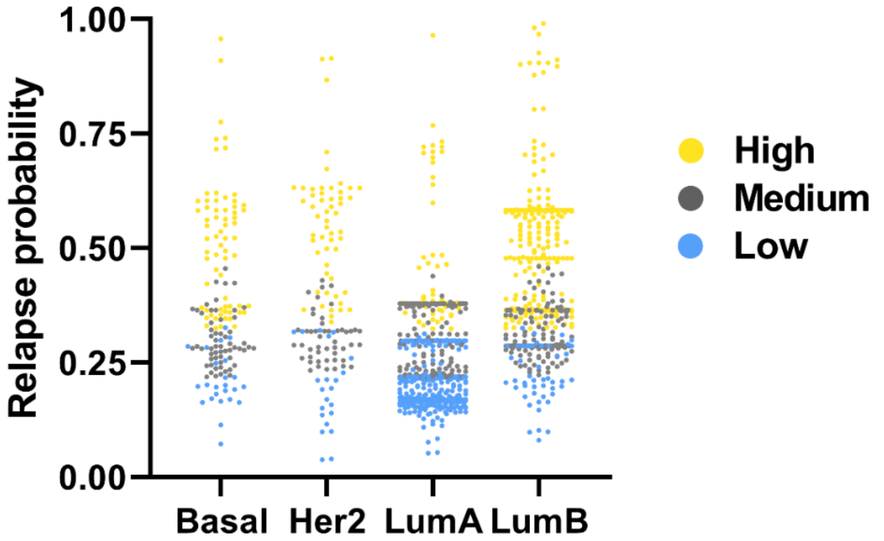


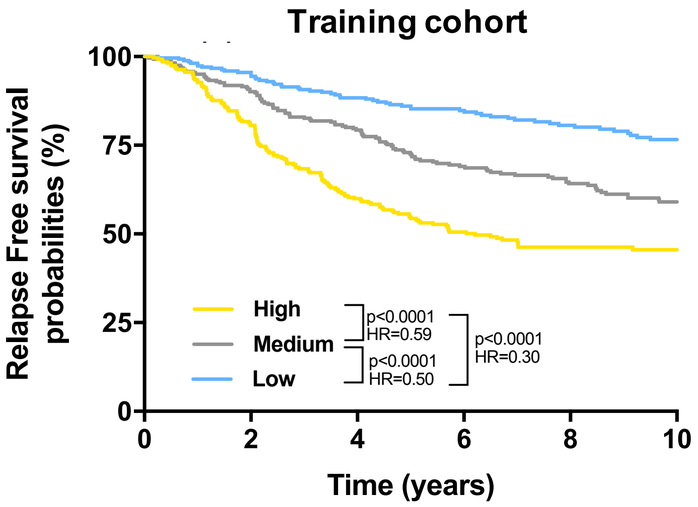
Average proportions of cells in different subgroups of patients suffering from breast cancer (2800 tumors analyzed)

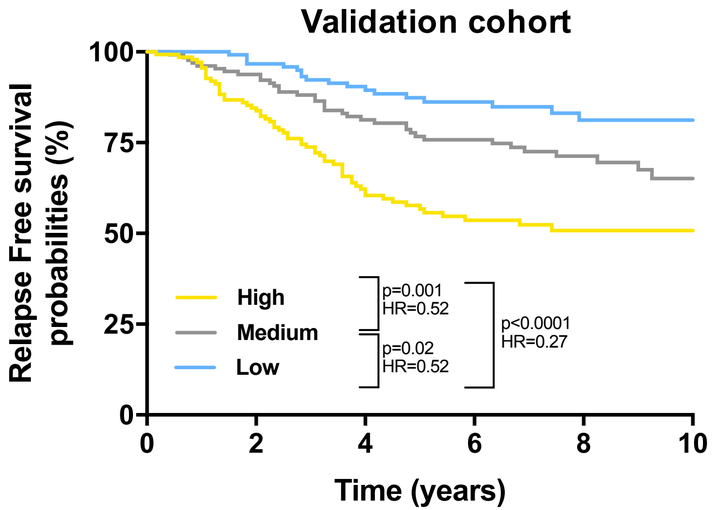
Immune cells in breast cancer

Construction of a Cox model to predict the risk of relapse including:

- Clinical variables
- Molecular classification of the tumor
- Immune cells proportions estimated







The estimation of immune cells brings valuable information in the prediction of the risk of relapse

Perspectives and future work

- Extend identification and local linear convergence rates for **block separable functions**
- Validation of this new estimation method on **data obtained at the cancer institute in Dijon**
- Consider **other types of cancer** (colon cancer) where immune cells could be a valuable information
- Perform an extensive **benchmark** against the large number of **estimation methods** proposed in the last years