l_1-norm regularization

Ji Zhu (Michigan), Saharon Rosset (IBM T. J. Watson),

Trevor Hastie (Stanford), Rob Tibshirani (Stanford)

Agenda

- Regularized optimization problems
- l_1-norm penalty
- Motivations:
  - Statistical advantage: sparsity (feature selection)
  - Computational advantage: piecewise linear solution paths

Prediction problems

- Training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$
- Input $\mathbf{x}_i \in \mathcal{R}^p$
- Output y_i
  - Regression: $y_i \in \mathcal{R}$
  - Two class classification: $y_i \in \{1, -1\}$
- High-dimensional data modeling: $p \gg n$
- Wish to find a prediction model for future data

$$\mathbf{x} \in \mathcal{R}^p \rightarrow y \in \mathcal{R} \text{ or } \{1, -1\}$$

The regularized optimization problem

$$\hat{\beta}(\lambda) = \arg\min_{\beta} L\left(\mathbf{y}, X\beta\right) + \lambda J(\beta)$$

Where

- $L(\cdot,\cdot)$: $\mathcal{R}^p \times \mathcal{R}^p \rightarrow \mathcal{R}$ is a convex non-negative loss functional.
- J: $\mathcal{R}^p \rightarrow \mathcal{R}$ is a convex non-negative penalty functional. Almost exclusively use $J(\beta) = \|\beta\|_q^q, q \geq 1$.
- $\lambda \geq 0$ is a tuning parameter
  - As $\lambda \rightarrow 0$, we get non-regularized models.
  - As $\lambda \rightarrow \infty$, we get that $\hat{\beta}(\lambda) \rightarrow 0$.

Examples

- Traditional methods
  - Ridge regression: $L(\mathbf{y}, X\beta) = \sum_i (y_i - \beta' \mathbf{x}_i)^2$, $J(\beta) = \|\beta\|_2^2$
  - Penalized logistic regression: $L(\mathbf{y}, X\beta) = \sum_i \log(1 + e^{-y_i \beta' \mathbf{x}_i})$, $J(\beta) = \|\beta\|_q^q$
- Modern methods
  - Support vector machines: $L(\mathbf{y}, X\beta) = \sum_i (1 - y_i \beta' \mathbf{x}_i)_+$, $J(\beta) = \|\beta\|_2^2$
  - AdaBoost: approximately $L(\mathbf{y}, X\beta) = \sum_i e^{-y_i \beta' \mathbf{x}_i}$, $J(\beta) = \|\beta\|_1$.

    See Rosset, Zhu and Hastie 2003.

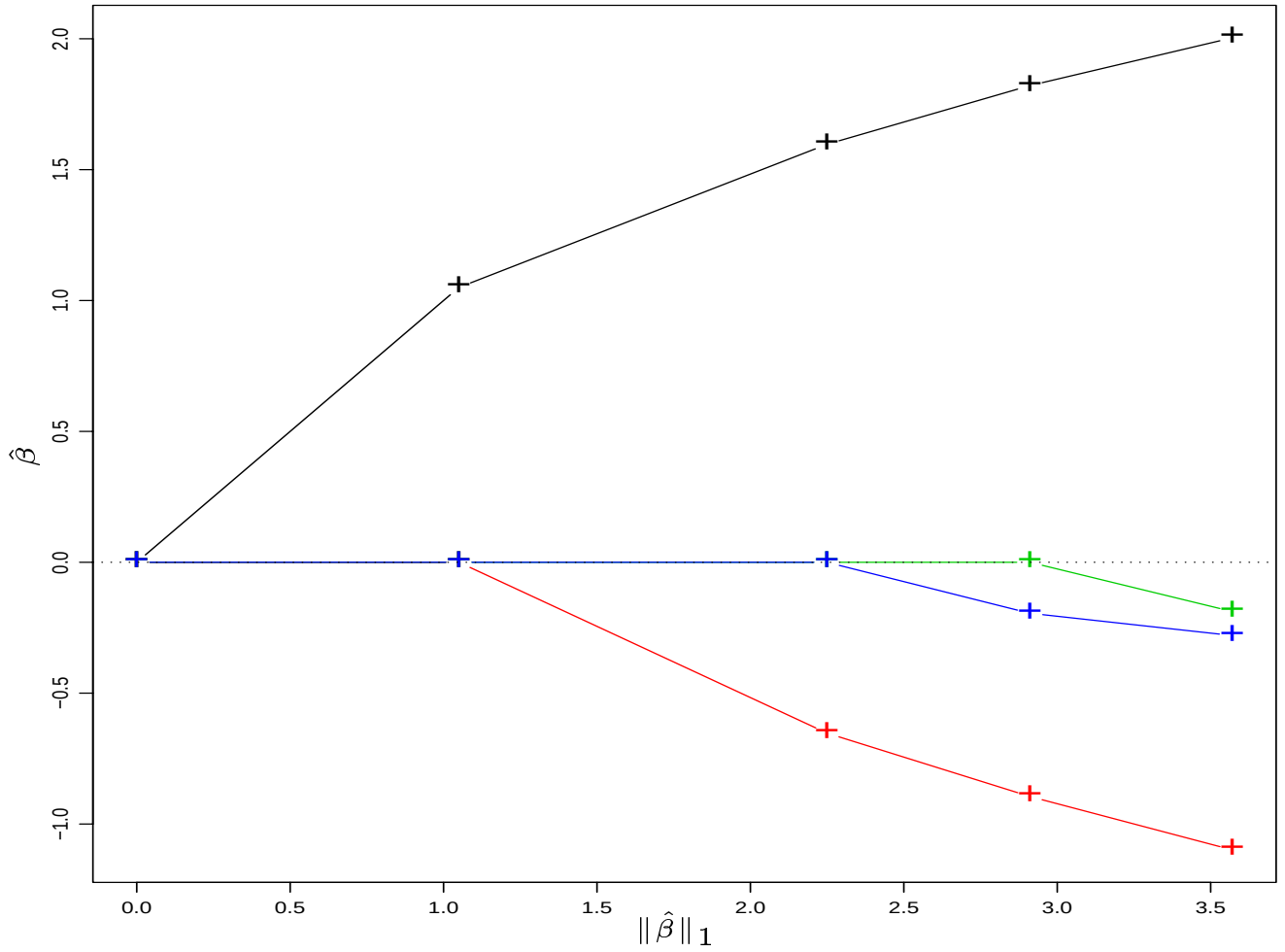
The l_1-norm penalty

A canonical example:

- Lasso (Tibshirani 1996, Efron, Hastie, Johnstone and Tibshirani 2002)

$$\hat{\beta}(\lambda) = \arg\min_{\beta} \sum_i (y_i - \beta' \mathbf{x}_i)^2 + \lambda \|\beta\|_1$$

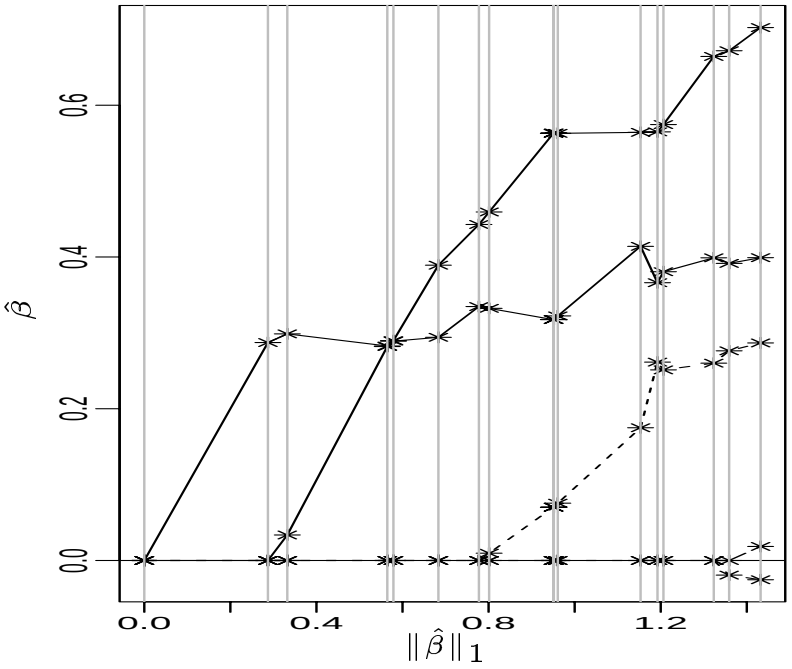
- Two properties:
  - Sparse solution (feature selection)
  - Piecewise linear coefficient paths


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
$\hat{\beta}$
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
$\|\hat{\beta}\|_1$

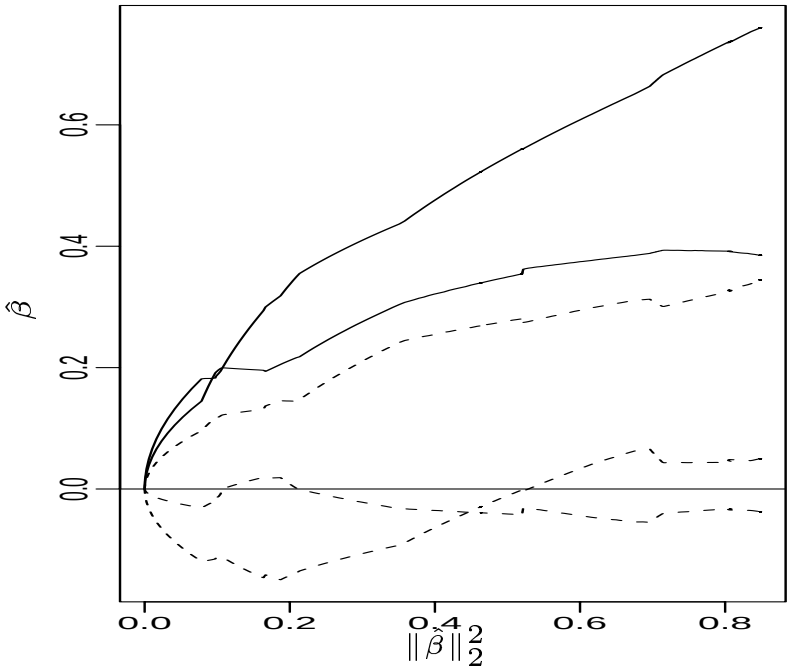
Sparsity

- l_1-norm penalty causes some of the coefficients $\hat{\beta}_j$'s to be exactly zero.
- l_1-norm penalty allows continuous feature selection as λ changes.

1-norm SVM



2-norm SVM



Existence and uniqueness of the sparse solution

$$\hat{\beta}(\lambda) = \arg\min_{\beta} L\left(\mathbf{y}, X\beta\right) + \lambda\|\beta\|_1$$

- There exists a solution which has at most n non-zero coefficients
- Under mild conditions, the sparse solution is unique
- Rosset, Zhu and Hastie 2003

Bet on sparsity

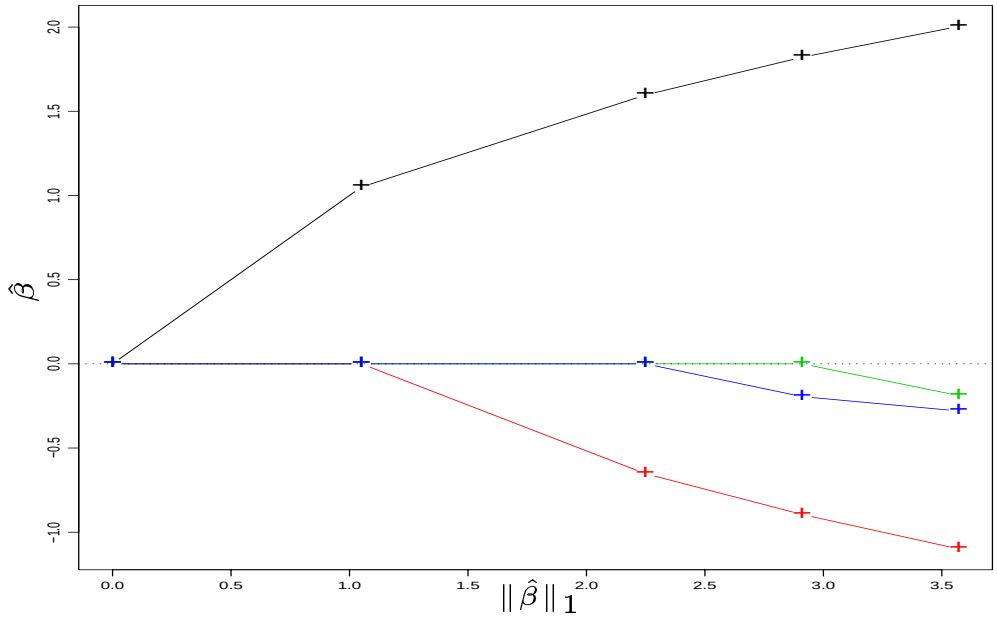
- $p \gg n$
- Sparse scenario: only a small number of true coefficients β_j's are non-zero.
- In the sparse scenario, the l_1-norm penalty works better than the l_2-norm penalty.
- In the non-sparse scenario, neither the l_1-norm penalty nor the l_2-norm penalty works well.
- Friedman, Hastie, Rosset, Tibshirani and Zhu 2003

Bet on sparsity simulation

- Regression: $n = 50, p = 300$
- Sparse scenario: $\beta_j \sim N(0,1),\ j = 1, \ldots, 10$ or 30, other $\beta_j = 0$
- Non-sparse scenario: $\beta_j \sim N(0,1),\ j = 1, \ldots, 300$

Computational advantage of the l_1-norm penalty

- When L is piecewise quadratic as a function of β, the solution path $\hat{\beta}(\lambda)$ is piecewise linear as a function of λ.
- Consequence:
  - Efficient algorithm to compute the exact whole solution path $\{\hat{\beta}(\lambda), 0 \leq \lambda \leq \infty\}$
  - Facilitate the selection of the tuning parameter λ



Examples

$$L(\mathbf{y}, X\beta) = \sum_i l(y_i, \beta' \mathbf{x}_i)$$

Examples:

- Regression (residual $r = y - \beta' x$)
  - Squared error loss: $l(r) = r^2$
  - Huber's loss with a fixed knot δ (more robust):

$$l(r) = \begin{cases} r^2 & \text{if } |r| \le \delta \\ 2\delta|r| - \delta^2 & \text{otherwise.} \end{cases}$$

  - Absolute value loss: $l(r) = |r|$ (non-differentiable at $r = 0$)

- Classification (margin $r = y\beta' x$)
  - Squared hinge loss: $l(r) = (1-r)_+^2$
  - Huberized squared hinge loss (more robust):

$$l(r) = \begin{cases} (1-\delta)^2 + 2(1-\delta)(\delta - r) & \text{if } r \leq \delta \\ (1-r)^2 & \text{if } \delta < r \leq 1 \\ 0 & \text{otherwise.} \end{cases}$$

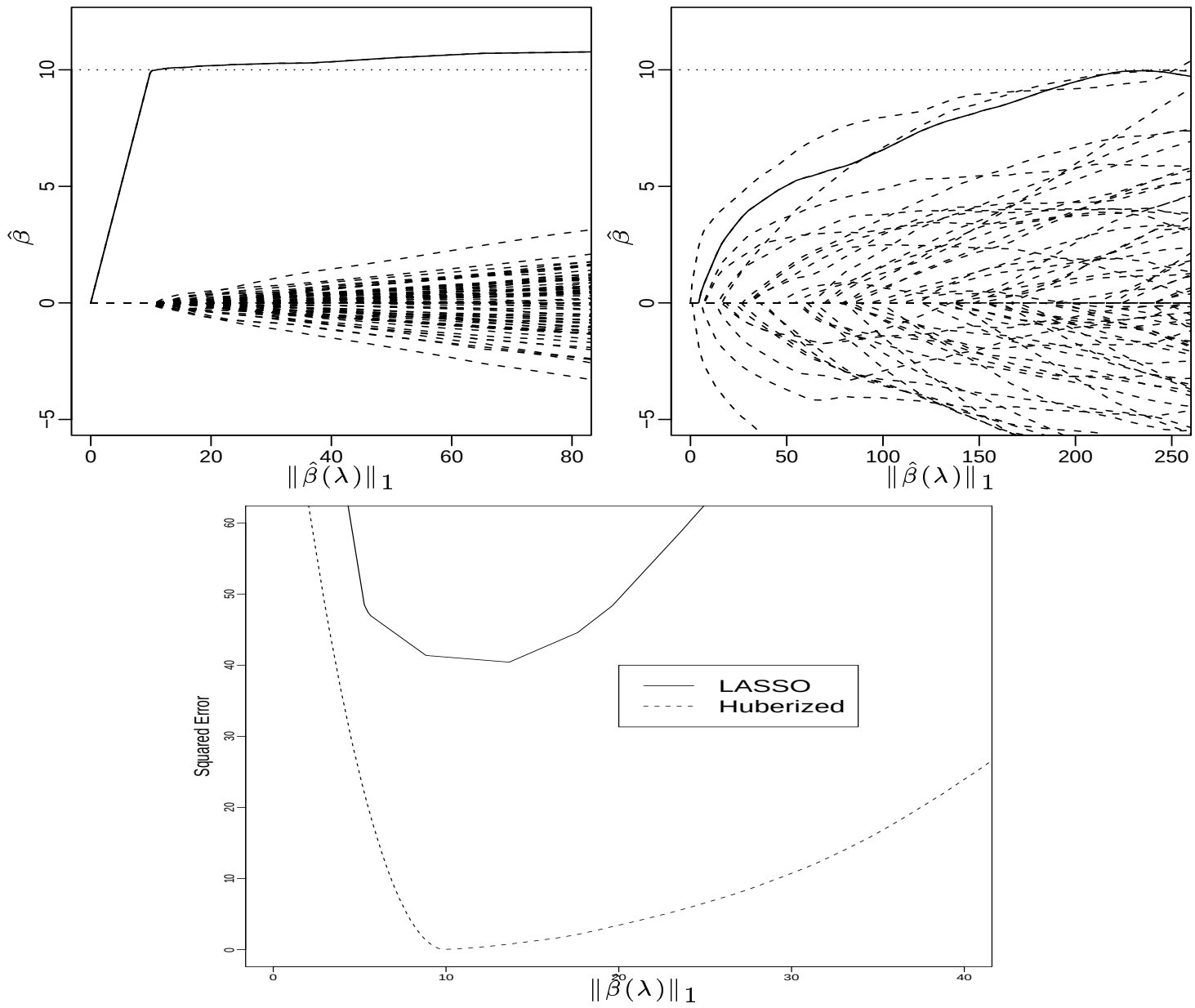
  - Hinge loss: $l(r) = (1-r)_+$ (non-differentiable at $r = 1$)

Illustration: regression

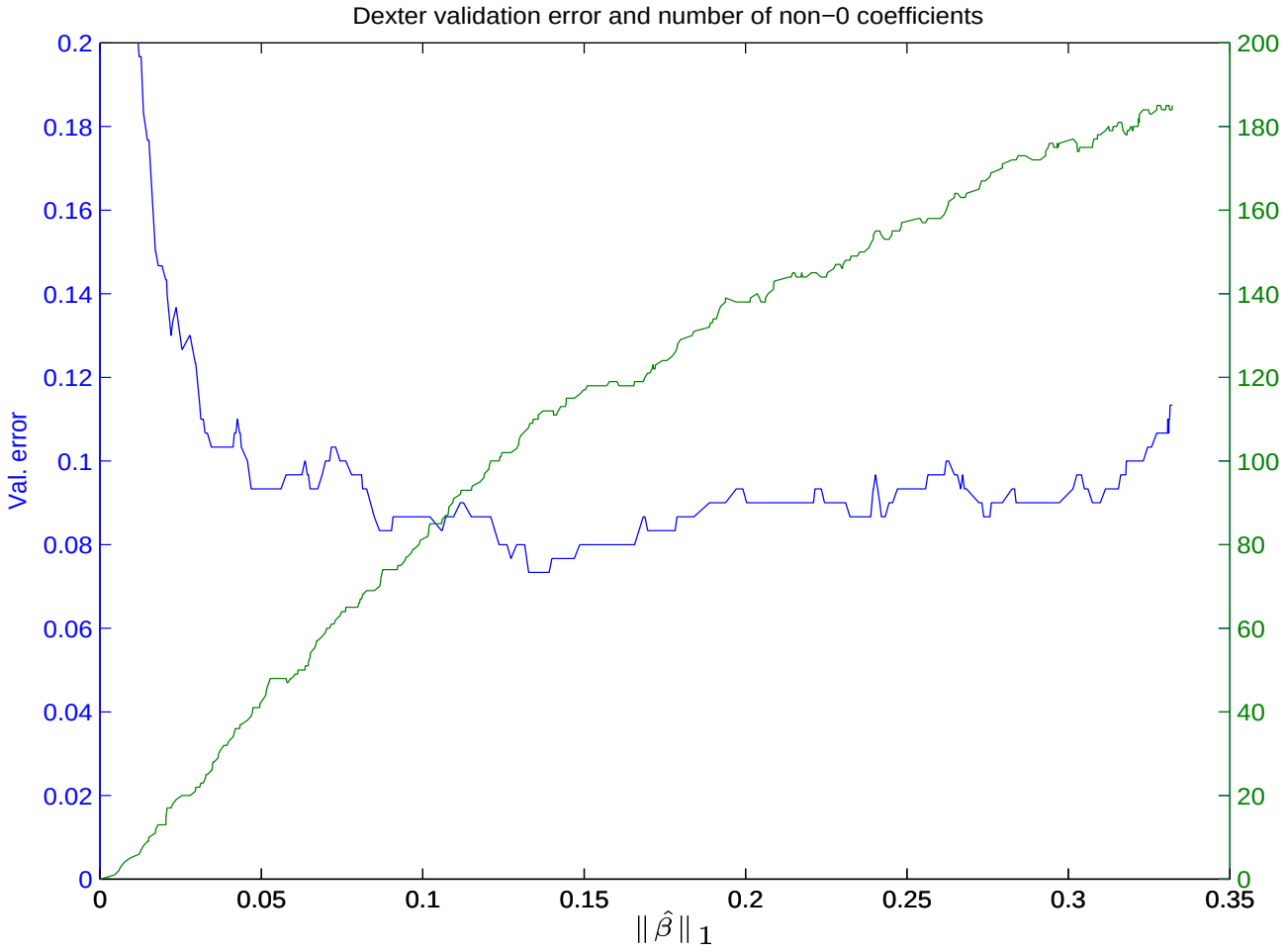
- $n = 100, p = 80.$
- All x_{ij} are i.i.d $N(0,1)$ and the true model is:

$$\begin{aligned} y_i &= 10 \cdot x_{i1} + \epsilon_i \\ \epsilon_i &\overset{iid}{\sim} 0.9 \cdot N(0,1) + 0.1 \cdot N(0,100) \end{aligned}$$

- Compare Huber's loss and squared error loss $+ l_1$-norm penalty


$\hat{\beta}$
$\|\hat{\beta}(\lambda)\|_1$
Squared Error
LASSO
Huberized

Dexter demonstration



Computational cost

Efficient algorithms available to compute the exact whole solution path $\{\hat{\beta}(\lambda), 0 \leq \lambda \leq \infty\}$. See Rosset and Zhu 2003.

- Approximate estimate of the computational cost
- Assume the number of joints is $O(n+p)$
- Total cost is $O(n^2 p)$
- Linear in p even when $p \gg n$

Summary

- What are good (loss L, penalty J) pairs? How should we determine the value of the tuning parameter λ?
- We use statistical motivations of robustness and sparsity to select interesting (loss L, penalty J) pairs.
- The resulting methods are adaptable (because we can choose an optimal tuning parameter), efficient (because we can generate the whole regularized path efficiently) and robust (because we choose to use robust loss functions).