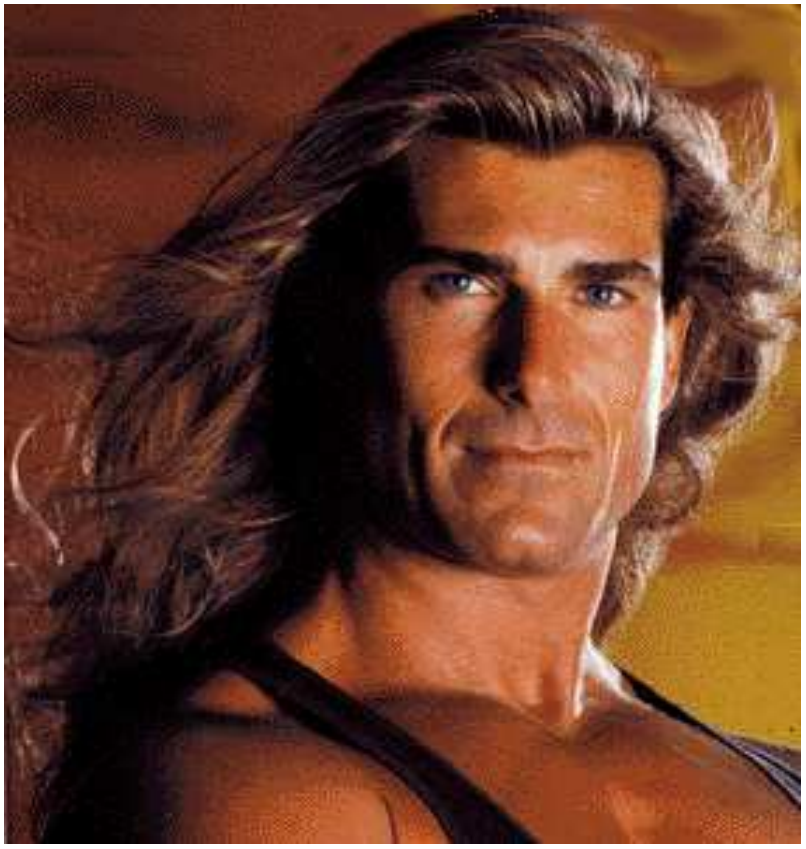
Pathwise coordinate optimization

Jerome Friedman, Trevor Hastie, Holger Hoefling, Robert Tibshirani
Stanford University

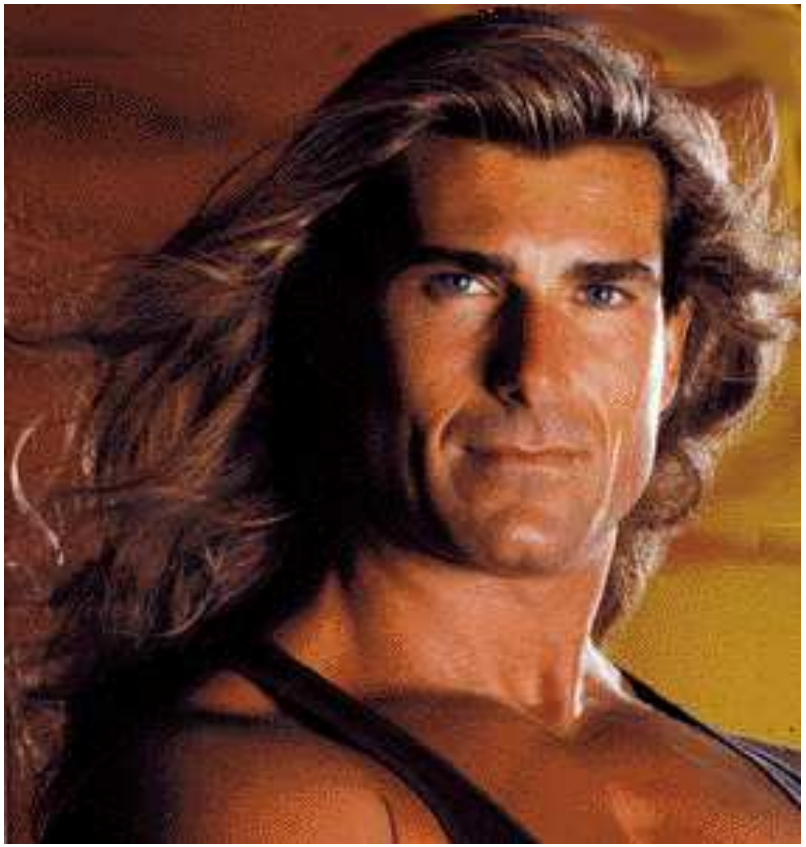
Acknowledgements: Thanks to **Stephen Boyd**, Michael Saunders, Guenther Walther.

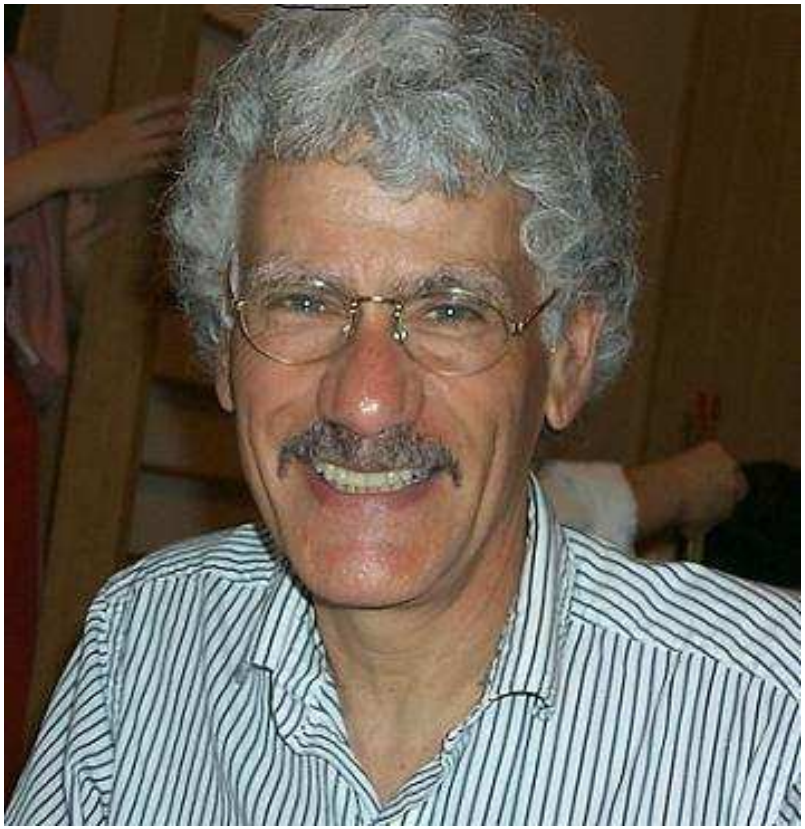
`Email:tibs@stat.stanford.edu`
`http://www-stat.stanford.edu/~tibs`



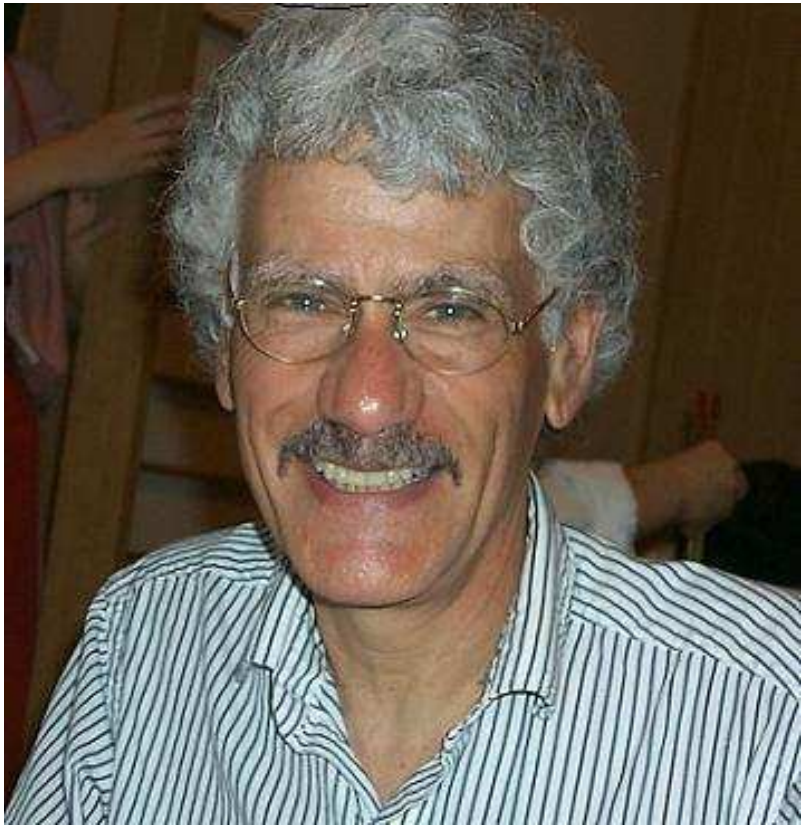
Jerome Friedman

Trevor Hastie





From MyHeritage.Com



Jerome Friedman

Trevor Hastie

Today's topic

Convex optimization for two problems:

- Lasso

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \cdot \sum_{j=1}^{p}|\beta_j|.$$

(assume features are standardized and y_i has mean zero; Tibshirani, 1996; also "Basis Pursuit"- Chen, Donoho and Saunders 1997)

- Fused lasso

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda_1 \sum_{j=1}^{p}|\beta_j| + \lambda_2 \sum_{j=2}^{p}|\beta_j - \beta_{j-1}|.$$

(Tibshirani, Saunders, Rosset, Zhu and Knight, 2004)

and other related problems

Approaches

- Many good methods available, including LARS (homotopy algorithm), and sophisticated convex optimization procedures
- *Today's topic:* an extremely simple-minded approach.

  Minimize over one parameter at a time, keeping all others fixed. "Coordinate descent".
- works for lasso, and many other related methods (elastic net, grouped lasso)
- coordinate-wise descent seems to have been virtually ignored by Statistics and CS researchers
- doesn't work for fused lasso, but a modified version does work.

A brief history of coordinate descent for the lasso

- 1997: Tibshirani's student Wenjiang Fu at University of Toronto develops the "shooting algorithm" for the lasso. Tibshirani doesn't fully appreciate it
- 2002 Ingrid Daubechies gives a talk at Stanford, describes a one-at-a-time algorithm for the lasso. Hastie implements it, makes an error, and Hastie +Tibshirani conclude that the method doesn't work
- 2006: Friedman is the external examiner at the PhD oral of Anita van der Kooij (Leiden) who uses the coordinate descent idea for the Elastic net. Friedman wonders whether it works for the lasso. Friedman, Hastie + Tibshirani start working on this problem

Outline

- pathwise coordinate descent for the lasso
- application to other separable models
- one-dimensional fused lasso for signals
- two-dimensional fused lasso for images

Pathwise coordinate descent for the lasso

- For a single predictor, solution is given by soft-thresholding of the least squares estimate: $\text{sign}(\hat{\beta})(|\hat{\beta}| - \gamma)_+$. Same thing for multiple orthogonal predictors.
- For general multiple regression, coordinate descent algorithm is soft-thresholding of a partial residual:

$$\tilde{\beta}_j(\lambda) \quad \leftarrow \quad S\Big(\sum_{i=1}^{n} x_{ij}(y_i - \tilde{y}_i^{(j)}), \lambda\Big) \tag{1}$$

  where $S(t, \lambda) = \text{sign}(t)(|t| - \lambda)_+$, $\tilde{y}_i^{(j)} = \sum_{k \neq j} x_{ik}\tilde{\beta}_k(\lambda)$.
- start with a large value of λ. Run procedure until convergence. Then decrease λ using previous solution as a warm start. Key point: quantities in above equation can be quickly updated a $j = 1, 2, \ldots p, 1, 2, \ldots$

- *Basic coordinate descent code for Lasso is just 73 lines of Fortran!*
- We also iterate on the *active set* of predictors to speed things up. *glmnet* package is the R implementation

Movie of Pathwise Coordinate algorithm

[show movie]

Comparison to LARS algorithm

- LARS (Efron, Hastie, Johnstone, Tibshirani 2002) gives exact solution path. Faster than general purpose algorithms for this problem. Closely related to the "homotopy " procedure (Osborne, Presnell, Turlach 2000)
- Pathwise coordinate optimization gives solution on a grid of λ values
- Speed comparisons: pathwise coordinate optimization (Fortran + R calling function), LARS-R (mostly R, with matrix stuff in Fortran), LARS-Fort (Fortran with R calling function) lasso2 (C + R calling function). Exponentially decreasing coefficients

Method	Population correlation between features					
	$n = 100, p = 1000$					
	0	0.1	0.2	0.5	0.9	0.95
coord-Fort	0.31	0.33	0.40	0.57	1.20	1.45
lars-R	2.18	2.46	2.14	2.45	2.37	2.10
lars-Fort	2.01	2.09	2.12	1.95	2.50	2.22
lasso2-C	2.42	2.16	2.39	2.18	2.01	2.71
	$n = 100, p = 20,000$					
coord-Fort	7.03	9.34	8.83	10.62	27.46	40.37
lars-R	116.26	122.39	121.48	104.17	100.30	107.29
lars-Fort would not run						
lasso2-C would not run						

Method	Population correlation between features					
	$n = 1000, p = 100$					
	0	0.1	0.2	0.5	0.9	0.95
coord-Fort	0.03	0.04	0.04	0.04	0.06	0.08
lars-R	0.42	0.41	0.40	0.40	0.40	0.40
lars-Fort	0.30	0.24	0.22	0.23	0.23	0.28
lasso2-C	0.73	0.66	0.69	0.68	0.69	0.70
	$n = 5000, p = 100$					
	0	0.1	0.2	0.5	0.9	0.95
coord-Fort	0.16	0.15	0.14	0.16	0.15	0.16
lars-R	1.02	1.03	1.02	1.04	1.02	1.03
lars-Fort	1.07	1.09	1.10	1.10	1.10	1.08
lasso2-C	2.91	2.90	3.00	2.95	2.95	2.92

Extension to related models

- Coordinate descent works whenever penalty is separable- a sum of functions of each parameter. See next slide.
- Examples: elastic net (Zhu and Hastie), garotte (Breiman), grouped lasso (Yuan), Berhu (Owen), least absolute deviation regression (LAD), LAD-lasso,
- Also - logistic regression with L1 penalty- use same approach as in lasso, with iteratively reweighted least squares. See next slide.

Logistic regression

- Outcome $Y = 0$ or 1; Logistic regression model

$$\log(\frac{Pr(Y=1)}{1-Pr(Y=1)}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \ldots$$

- Criterion is binomial log-likelihood +absolute value penalty
- Example: sparse data. $N = 50,000$, $p = 700,000$.
- State-of-the-art interior point algorithm (Stephen Boyd, Stanford), exploiting sparsity of features : 3.5 hours for 100 values along path
- Pathwise coordinate descent (*glmnet*): 1 minute
- 75 vs 7500 lines of code

Recent generalizations of lasso

- *The Elastic net* (Zou, Hastie 2005)

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{n}(y_i - \sum_{j=1}^{n} x_{ij}\beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} \beta_j^2/2 \qquad (2)$$

- *Grouped Lasso* (Yuan and Lin 2005). Here the variables occur in groups (such as dummy variables for multi-level factors). Suppose X_j is an $N \times p_j$ orthonormal matrix that represents the jth group of p_j variables, $j = 1, \ldots, m$, and β_j the corresponding coefficient vector. The grouped lasso solves

$$\min_{\beta} ||y - \sum_{j=1}^{m} X_j\beta_j||_2^2 + \sum_{j=1}^{m} \lambda_j ||\beta_j||_2, \qquad (3)$$

  where $\lambda_j = \lambda\sqrt{p_j}$.

Can apply coordinate descent in both models!

Fused lasso

- general form

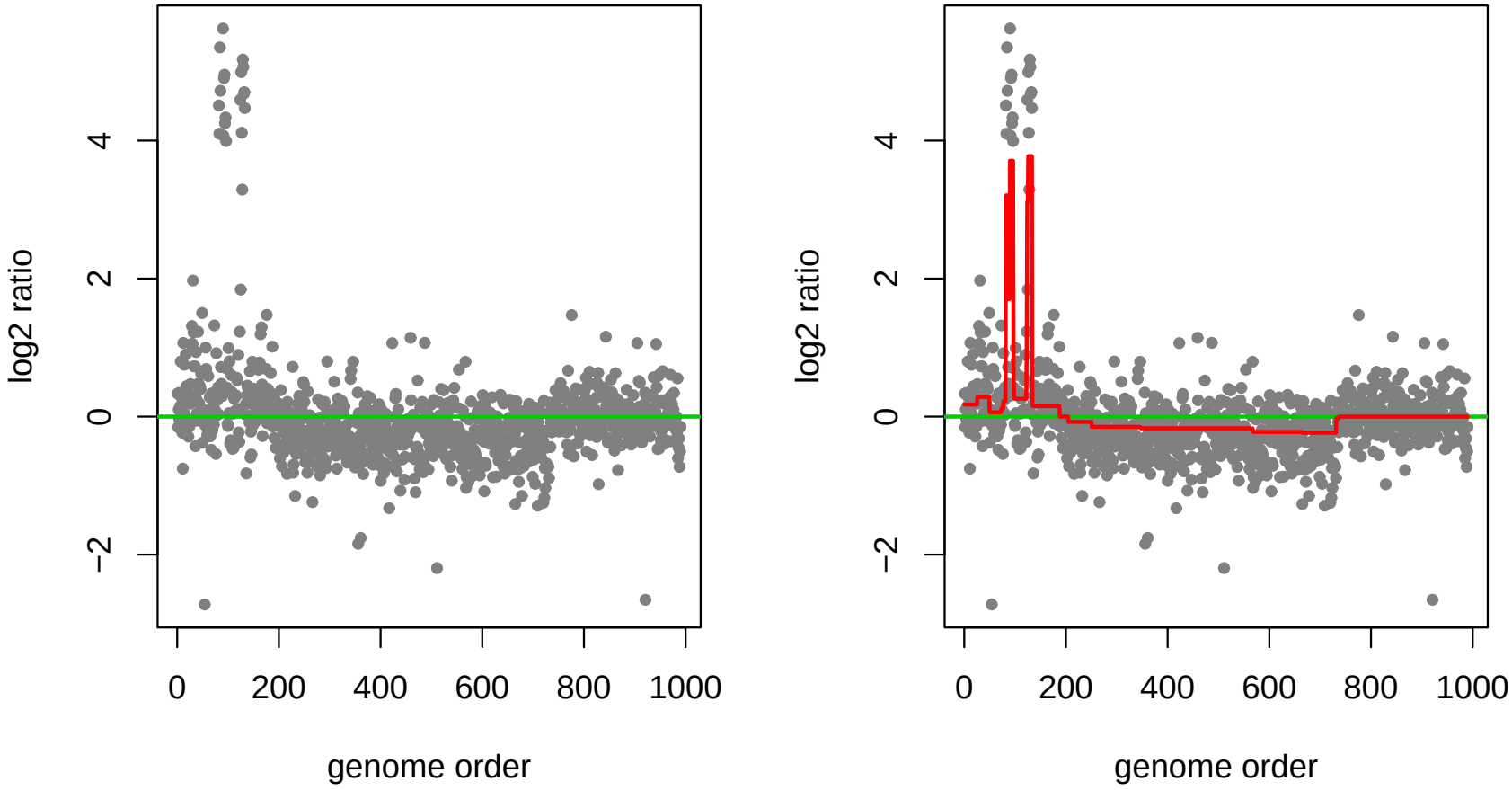
$$\min_{\beta} \frac{1}{2} \sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=2}^{p} |\beta_j - \beta_{j-1}|.$$

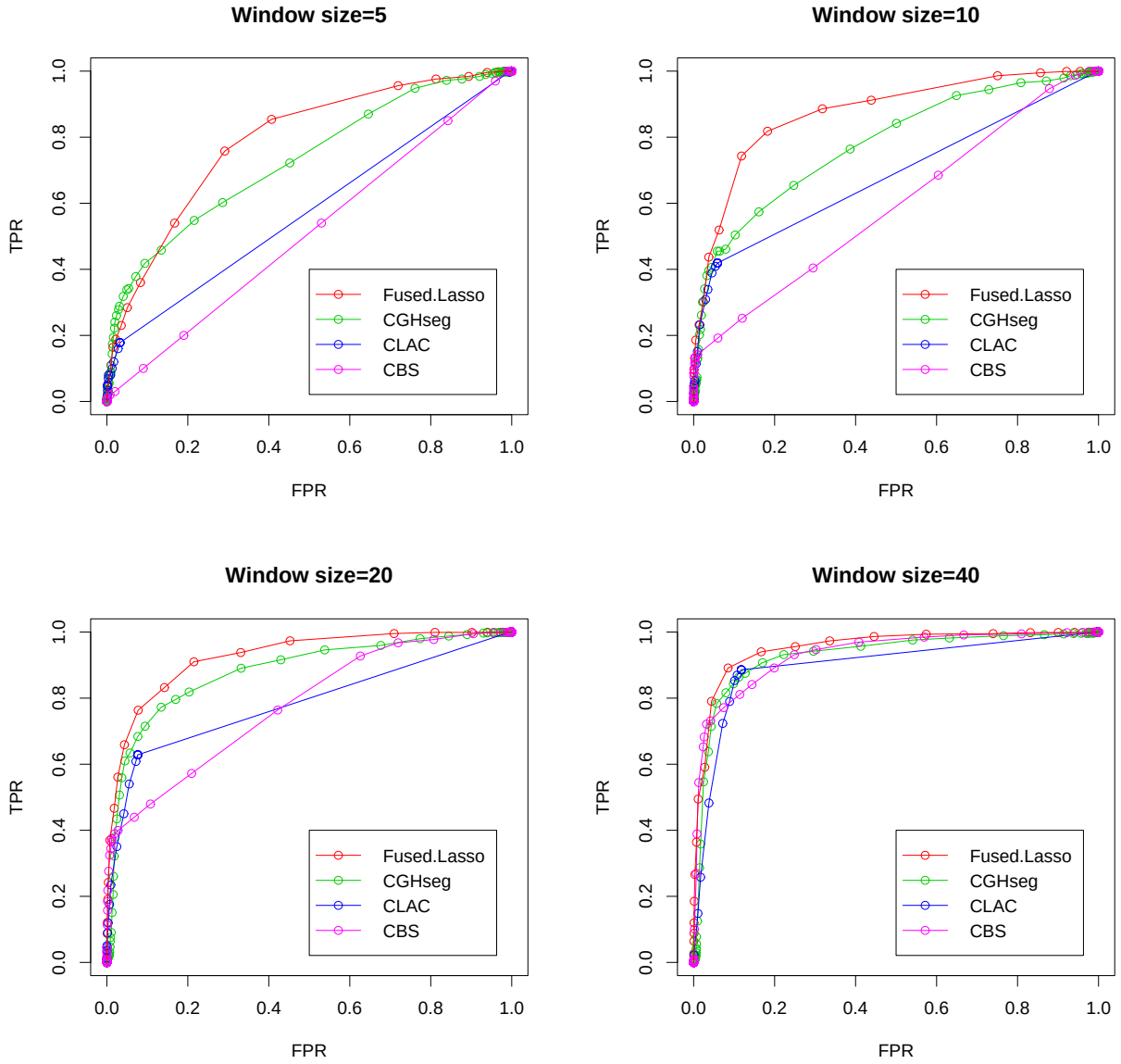
- Will focus on diagonal fused lasso

$$\min_{\beta} \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda_1 \sum_{i=1}^{n} |\beta_i| + \lambda_2 \sum_{i=2}^{n} |\beta_i - \beta_{i-1}|.$$

- useful for approximating signals and images

Comparative Genomic Hybridization (CGH) data


log2 ratio
4
2
0
−2
0
200
400
600
800
1000
genome order
log2 ratio
4
2
0
−2
0
200
400
600
800
1000
genome order


Window size=5
Window size=10
Window size=20
Window size=40
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0
Fused.Lasso
CGHseg
CLAC
CBS

Surprise!

- loss function is strictly convex, but coordinate descent doesn't work!
- loss function is not differentiable $->$ algorithm can get stuck

When does coordinate descent work?

Paul Tseng (1988), (2001)

If

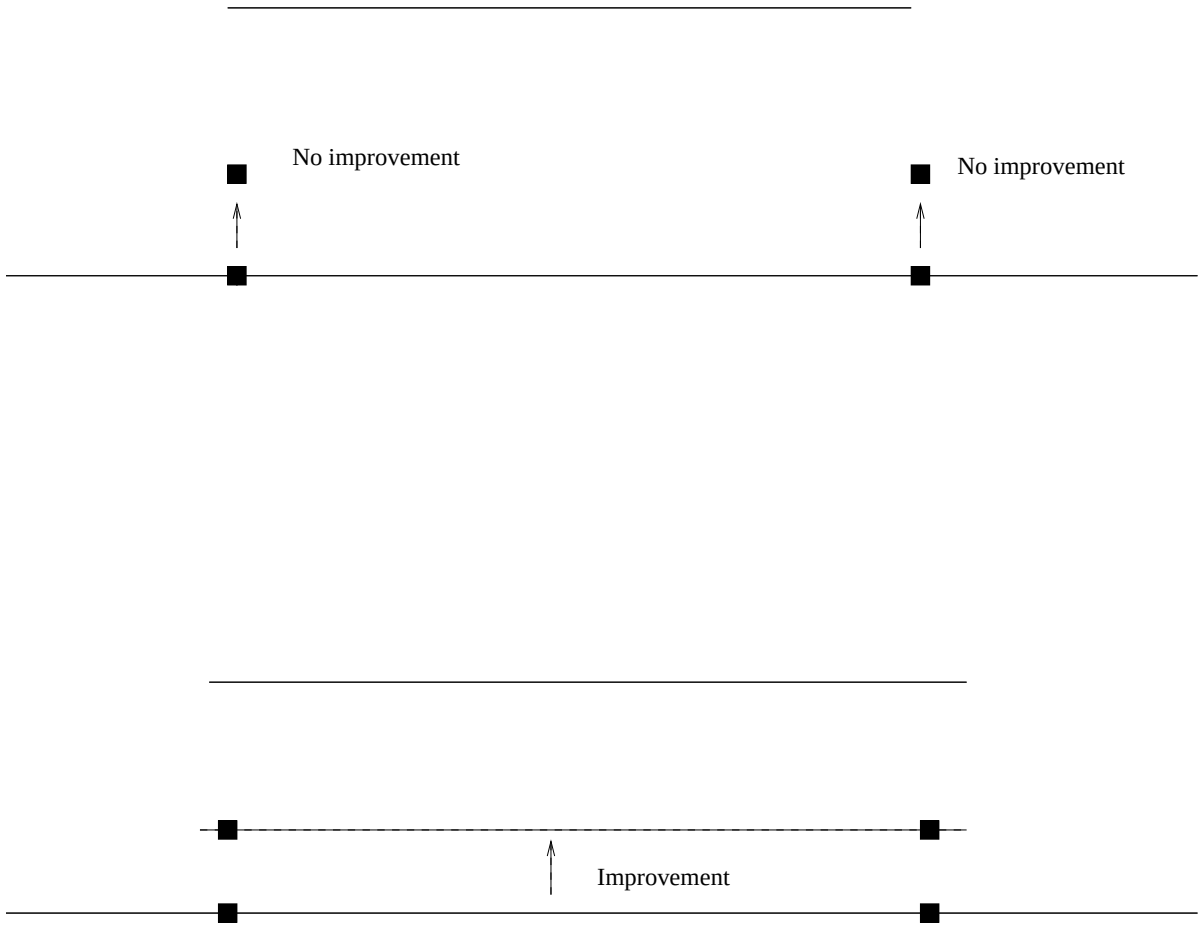
$$f(\beta_1 \dots \beta_p) = g(\beta_1 \dots \beta_p) + \sum h_j(\beta_j)$$

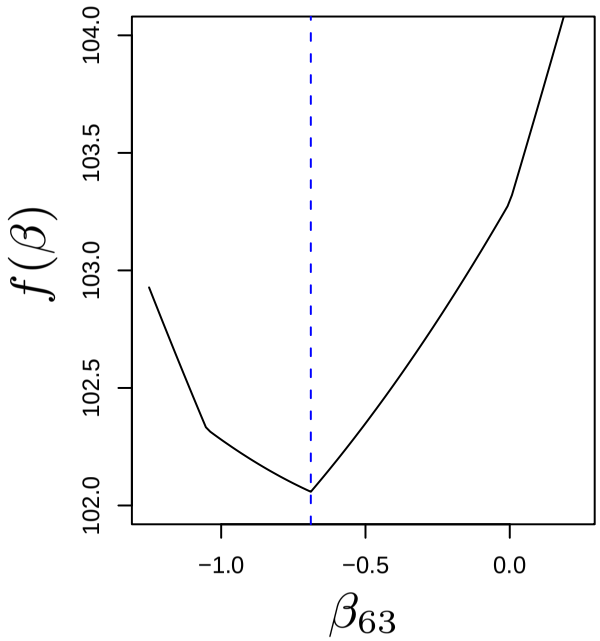
where $g(\cdot)$ is convex and differentiable, and $h_j(\cdot)$ is convex, then coordinate descent converges to a minimizer of f.

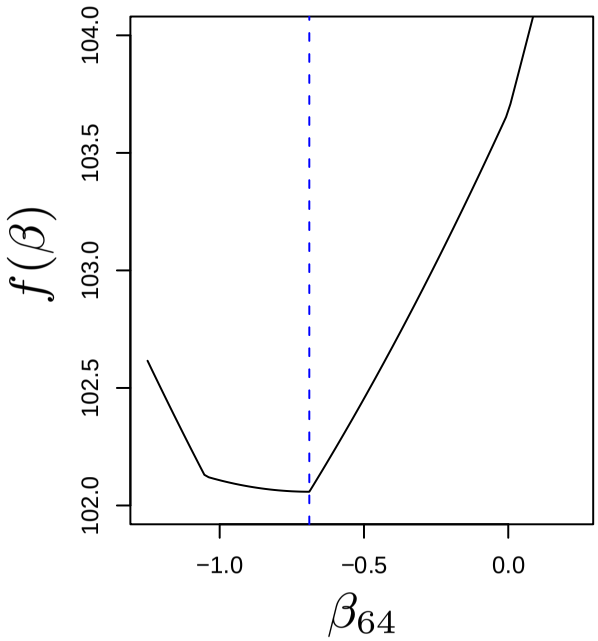
What goes wrong

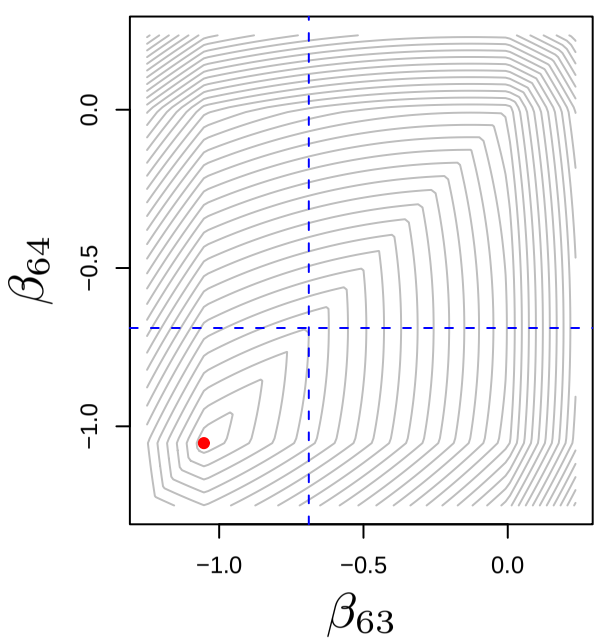
- can have a situation where no move of an individual parameter helps, but setting two (or more) adjacent parameters equal, and moving their common value, does help

The problem




$f(\beta)$
102.0
102.5
103.0
103.5
104.0
−1.0
−0.5
0.0
β_{63}

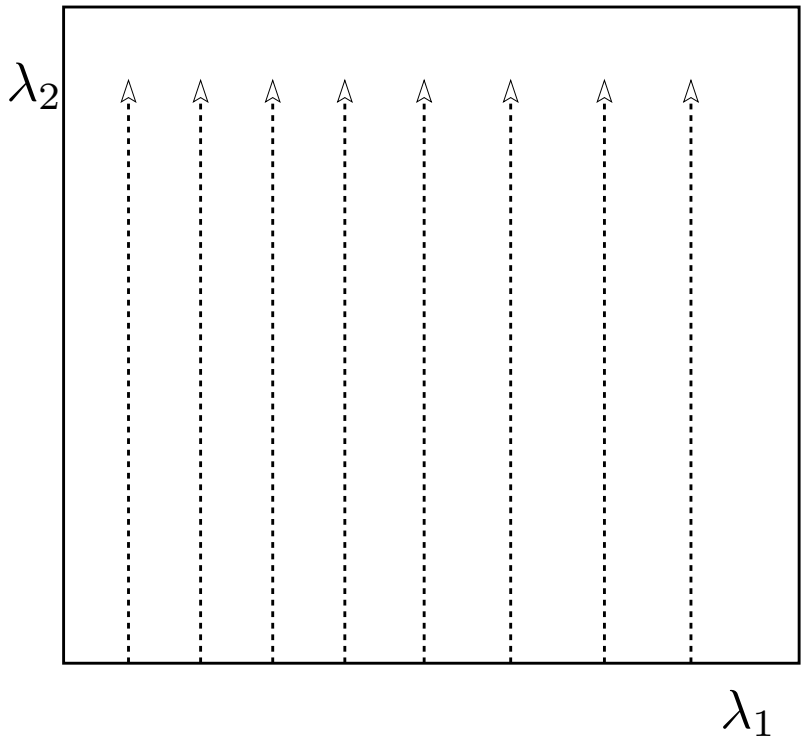

$f(\beta)$
102.0
102.5
103.0
103.5
104.0
−1.0
−0.5
0.0
β_{64}


β_{64}
0.0
−0.5
−1.0
−1.0
−0.5
0.0
β_{63}

Our solution

- try moving not only individual parameters, but also fusing together pairs and varying their common value
- What if no fusion of a pair helps, but moving three or more together helps?
- Solution- *don't let this happen*
- fix λ_1 , start with $\lambda_2 = 0$ and solve the problem for incremental values of λ_2.

Strategy for fused lasso



Theorem

- any two non-zero adjacent parameters that are fused for value λ_2, are also fused for $\lambda_2 > \lambda_2'$.
- seems obvious, but surprisingly difficult to prove
- thanks to Stephen Boyd for showing us the subgradient representation.

Sub-gradient representations

Necessary and sufficient conditions for solution: see Bertsekas (1999)

Lasso

$$-X^T(y - X\beta) + \lambda \sum s_j = 0$$

where $s_j \in \text{sign}(\beta_j)$, that is $s_j = \text{sign}(\beta_j)$ if $\beta_j \neq 0$, and $s_j \in [-1, 1]$ otherwise.

Fused lasso

$$\begin{aligned} -(y_1 - \beta_1) + \lambda_1 s_1 - \lambda_2 t_2 &= 0 \\ -(y_j - \beta_j) + \lambda_1 s_j + \lambda_2(t_j - t_{j+1}) &= 0, \; j = 2, \ldots n \end{aligned} \tag{4}$$

with $s_j = \text{sign}(\beta_j)$ if $\beta_j \neq 0$ and $s_j \in [-1, 1]$ if $\beta_j = 0$. Similarly, $t_j = \text{sign}(\beta_j - \beta_{j-1})$ if $\beta_j \neq \beta_{j-1}$ and $t_j \in [-1, 1]$ if $\beta_j = \beta_{j-1}$.

Algorithm for fused lasso

- Fix λ_1 and start λ_2 at zero
- for each parameter, try coordinate descent. (This is fast-function is piecewise linear).
- If coordinate descent doesn't help, try fusing it with the parameter to its left and moving their common value
- when algorithm has converged for the current value of λ_2, fuse together equal adjacent non-zero values forever. Collapse data and form a weighted problem.
- increment λ_2 and repeat

Movie of fused lasso algorithm

[Show movie of 1D fused lasso]

Speed comparisons

generalized pathwise coordinate optimization vs two-phase active set algorithm `sqopt` of Gill, Murray, and Saunders

$p = 1000$

λ_1	λ_2	# Active	Path Coord	Standard
0.01	0.01	450	0.039	1.933
0.01	2.00	948	0.017	0.989
1.00	0.01	824	0.022	1.519
1.00	2.00	981	0.023	1.404
2.00	0.01	861	0.023	1.499
2.00	2.00	991	0.018	1.407

$p = 5000$

λ_1	λ_2	# Active	Path Coord	Standard
0.002	0.002	4044	0.219	19.157
0.002	0.400	3855	0.133	27.030
0.200	0.002	4305	0.150	41.105
0.200	0.400	4701	0.129	45.136
0.400	0.002	4301	0.108	41.062
0.400	0.400	4722	0.119	38.896

Two-dimensional fused lasso

- problem has the form

$$\min_{\beta} \frac{1}{2} \sum_{i=1}^{p} \sum_{i'=1}^{p} (y_{ii'} - \beta_{ii'})^2$$

$$\text{subject to} \quad \sum_{i=1}^{p} \sum_{i'=1}^{p} |\beta_{ii'}| \leq s_1,$$

$$\sum_{i=1}^{p} \sum_{i'=2}^{p} |\beta_{i,i'} - \beta_{i,i'-1}| \leq s_2,$$

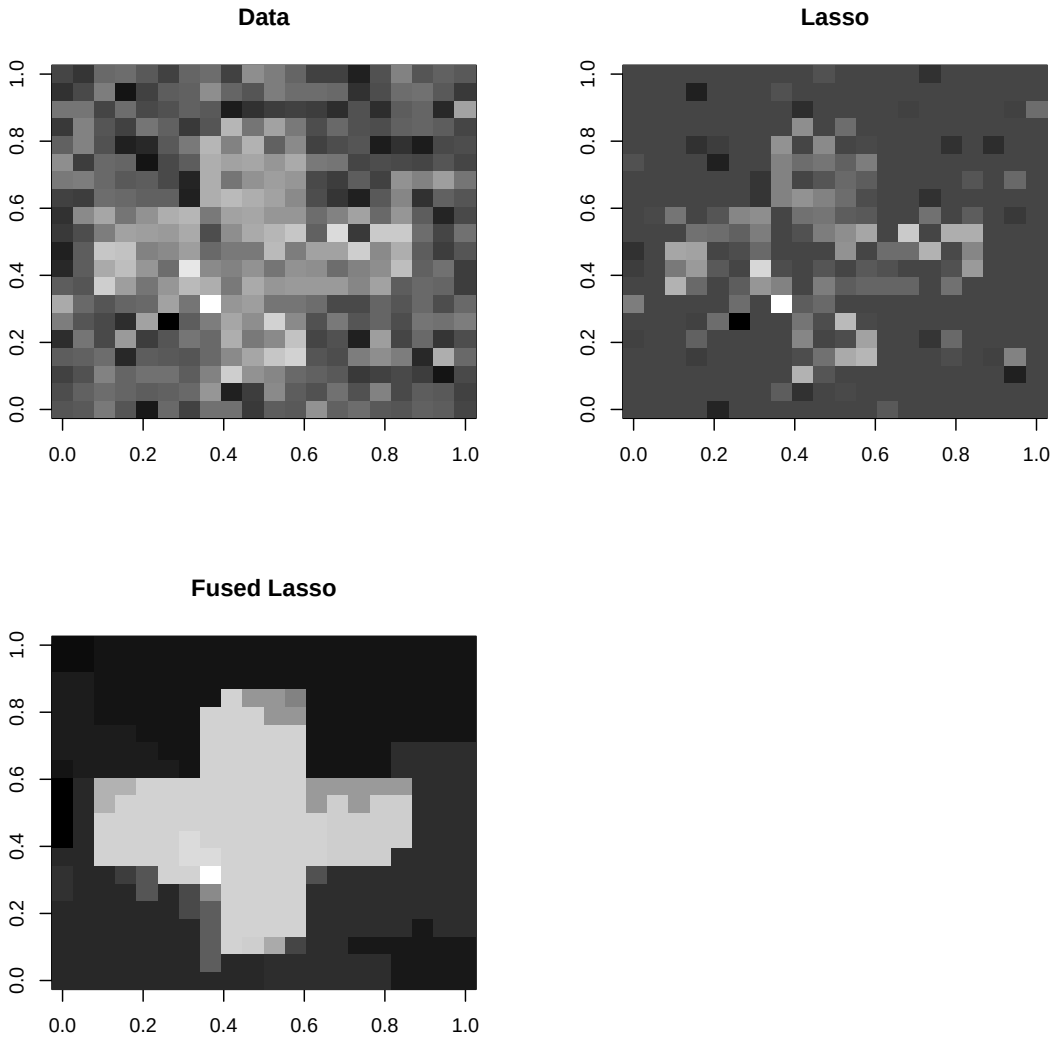
$$\sum_{i=2}^{p} \sum_{i'=1}^{p} |\beta_{i,i'} - \beta_{i-1,i'}| \leq s_3.$$

- we consider fusions of pixels one unit away in horizontal or vertical directions.
- after a number of fusions, fused regions can become very irregular in shape. Required some impressive programming by Friedman.

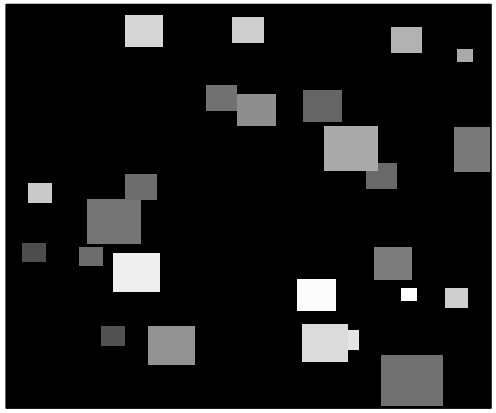
- algorithm is not exact- fused groups can break up as λ_2 increases

Examples

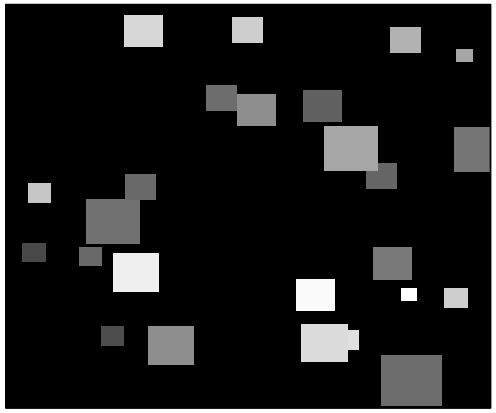
Two-fold validation (even and odd pixels) was used to estimate λ_1, λ_2.


Data
Lasso
Fused Lasso
0.0
0.2
0.4
0.6
0.8
1.0

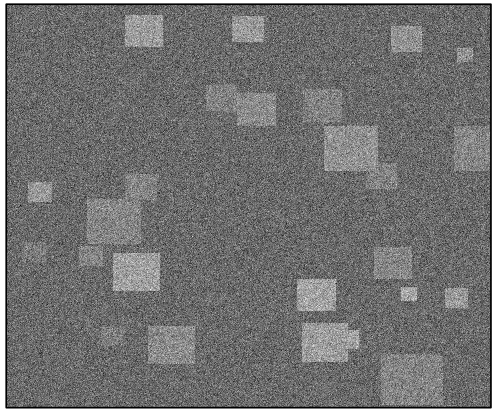
Original image



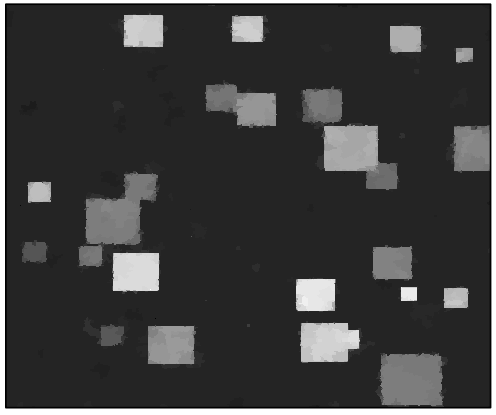
Fused lasso reconstruction



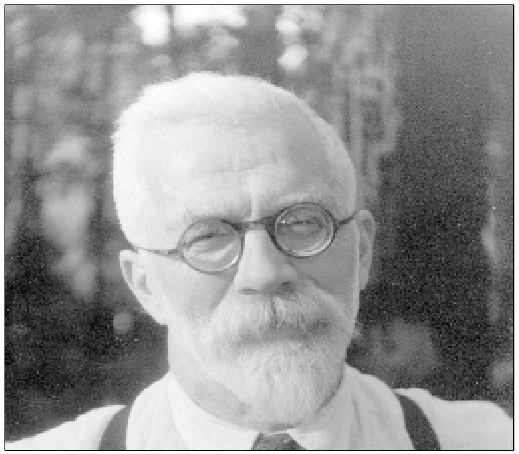
Noisy image



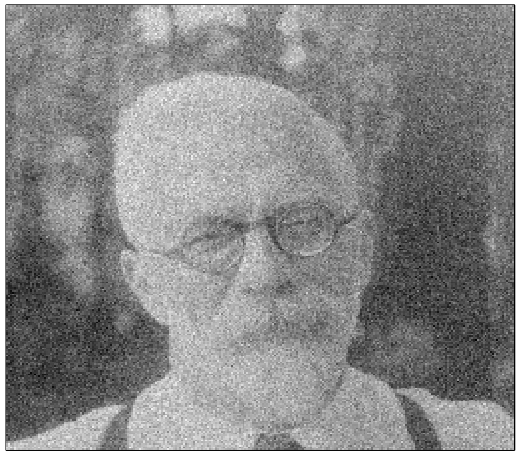
Fused lasso reconstruction



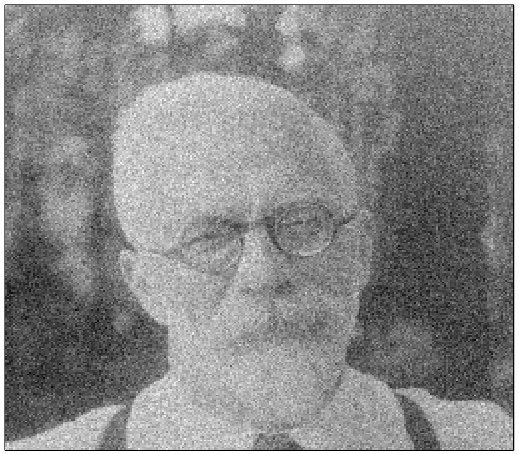
original image



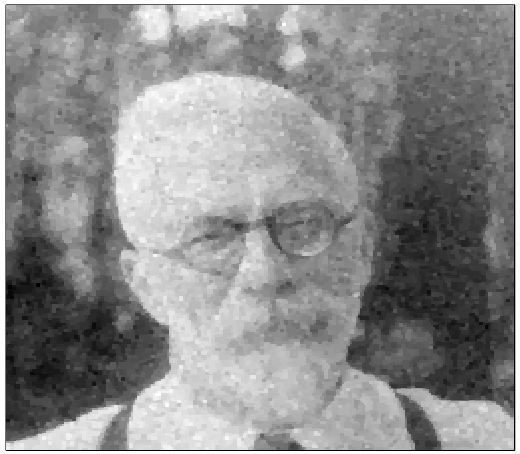
noisy image



lasso denoising



fusion denoising



movie

[Show movie of 2D fused lasso]

Discussion

- Pathwise coordinate optimization is simple but extremely useful
- We are thinking about/ working on further applications of pathwise coordinate optimization:
  - generalizations of the fused lasso to higher (> 2) dimensions
  - applications to big, sparse problems
  - Hoefling has derived an exact path (LARS-like) algorithm for the fused lasso, using maximal flow ideas