Sparsity and smoothness via the Fused Lasso

with applications to protein mass spec and microarray studies

Robert Tibshirani

Stanford Univ

Joint work with Michael Saunders, Saharon Rosset, Ji Zhu

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

Outline

- Review lasso and least angle regression
- Introduce fused lasso, with application to protein mass spectroscopy and gene expression data

General comments

Applied genomics needs:

- New statistical methodology
- Careful application of existing methodology- eg design of experiments, cross-validation
- Well-designed, free software with an easy-to-use interface.

  #1 criterion for choosing between statistical methods: *availability in a convenient package.*

An example

Cross-validation, the wrong and right way.

Consider a simple classifier for microarrays:

1. Starting with all genes (say 5000), find the 200 genes having the largest correlation with the class labels
2. Carry about nearest-centroid classification using only these 200 genes

How do we estimate the test set performance of this classifier?

- *Wrong:* Apply cross-validation in step 2.
- *Right:* Apply cross-validation to steps 1 and 2.

It is easy to simulate realistic data with the class labels independent of the outcome, — so that true test error =50%— but "Wrong" CV error estimate is zero!

I have seen this error made in 4 high profile papers in the couple of years. See Ambroise and McLachlan PNAS 2002 for a nice discussion of this point.

MicroArray Example

- Expression data for 38 Leukemia patients ("Golub" data).
- X matrix with 38 samples and 7129 variables (genes)
- Response y is dichotomous ALL (27) vs AML (11)
- Idea: fit a linear model $y_i = \beta_0 + \sum_i x_{ij}\beta_j$, and use this to predict class label y_i.
- Have also a separate test set of 34 patients

Linear regression via the Lasso (Tibshirani, 1995)

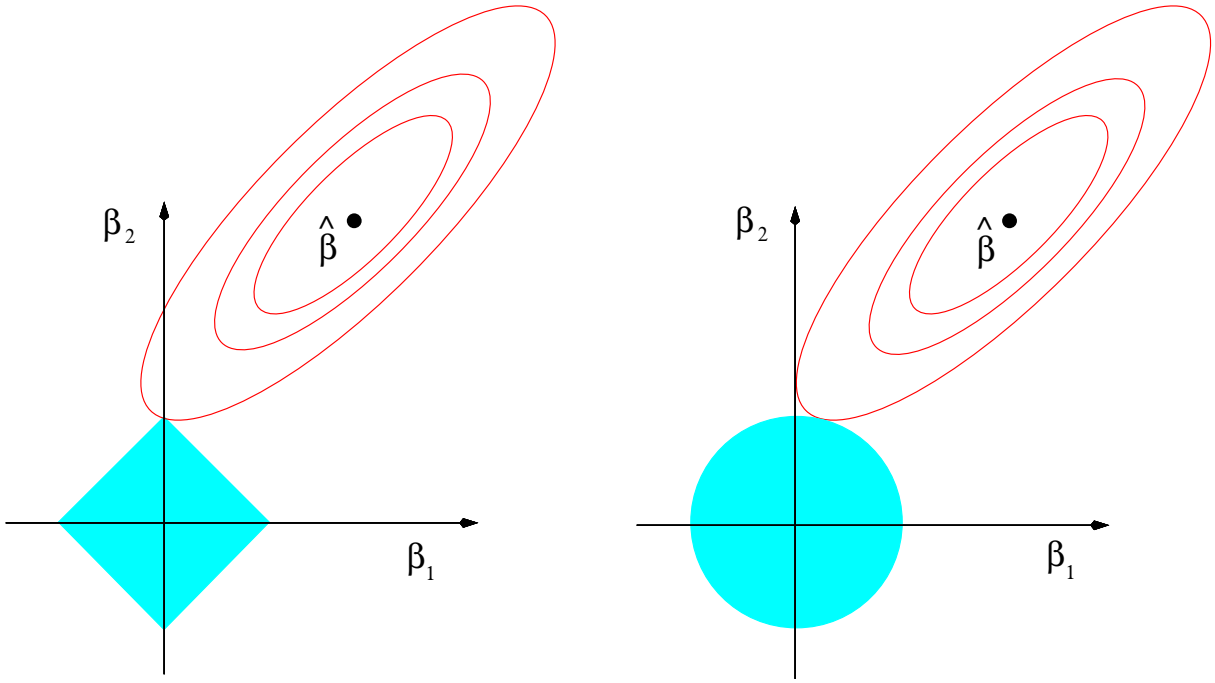
- Data consists of N observations data on p predictors $x_1, x_2, \ldots x_p$ and an outcome measurement y. Trying to predict y from $x_1, x_2, \ldots x_p$.
- Assume $\bar{y} = 0$, $\bar{x}_j = 0$, $\mathrm{Var}(x_j) = 1$ for all j.
- Minimize $\sum_i (y_i - \sum_j x_{ij}\beta_j)^2$ subject to $\sum_j |\beta_j| \leq s$
- With orthogonal predictors, solutions are soft thresholded version of least squares coefficients:

$$\mathrm{sign}(\hat{\beta}_j)(|\hat{\beta}_j| - \gamma)_+$$

(γ is a function of s)

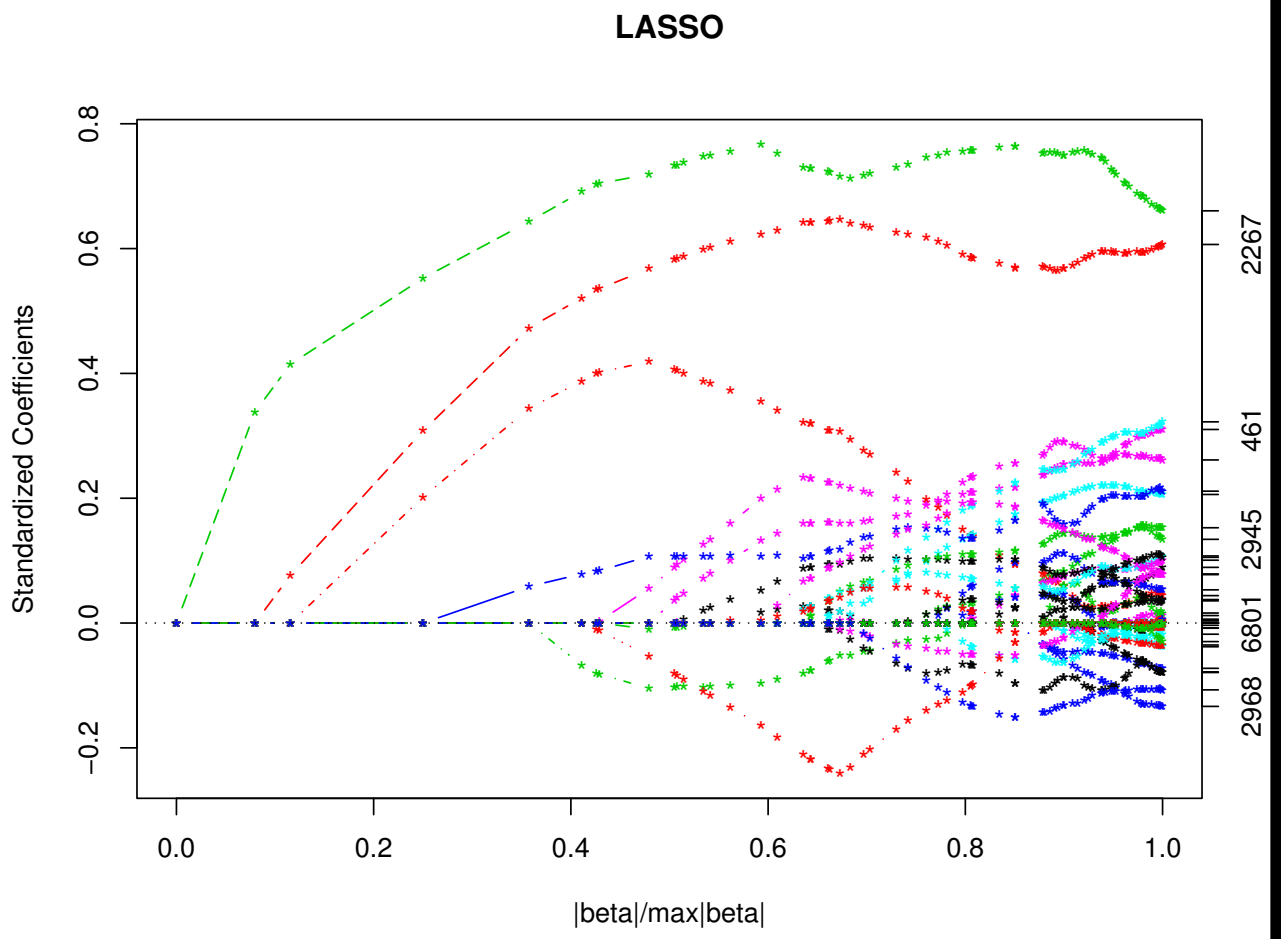
- For small values of the bound s, Lasso does variable selection. See pictures.
- Ridge regression- closely related to the lasso, uses a penalty $\sum_j \beta_j^2 \leq s$

Lasso and Ridge regression



More on Lasso

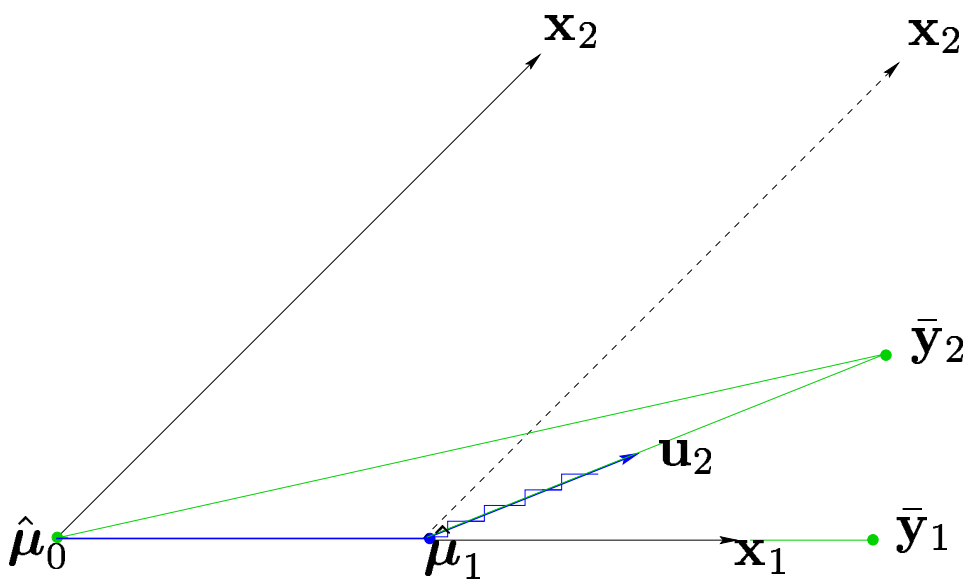
- Current implementations use quadratic programming to compute solutions
- Can be applied when $p > n$. In that case, number of non-zero coefficients is at most n (by convex duality)
- interesting consequences for applications, eg microarray data


LASSO
Standardized Coefficients
0.8
0.6
0.4
0.2
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
|beta|/max|beta|
2267
461
2945
6801
2968

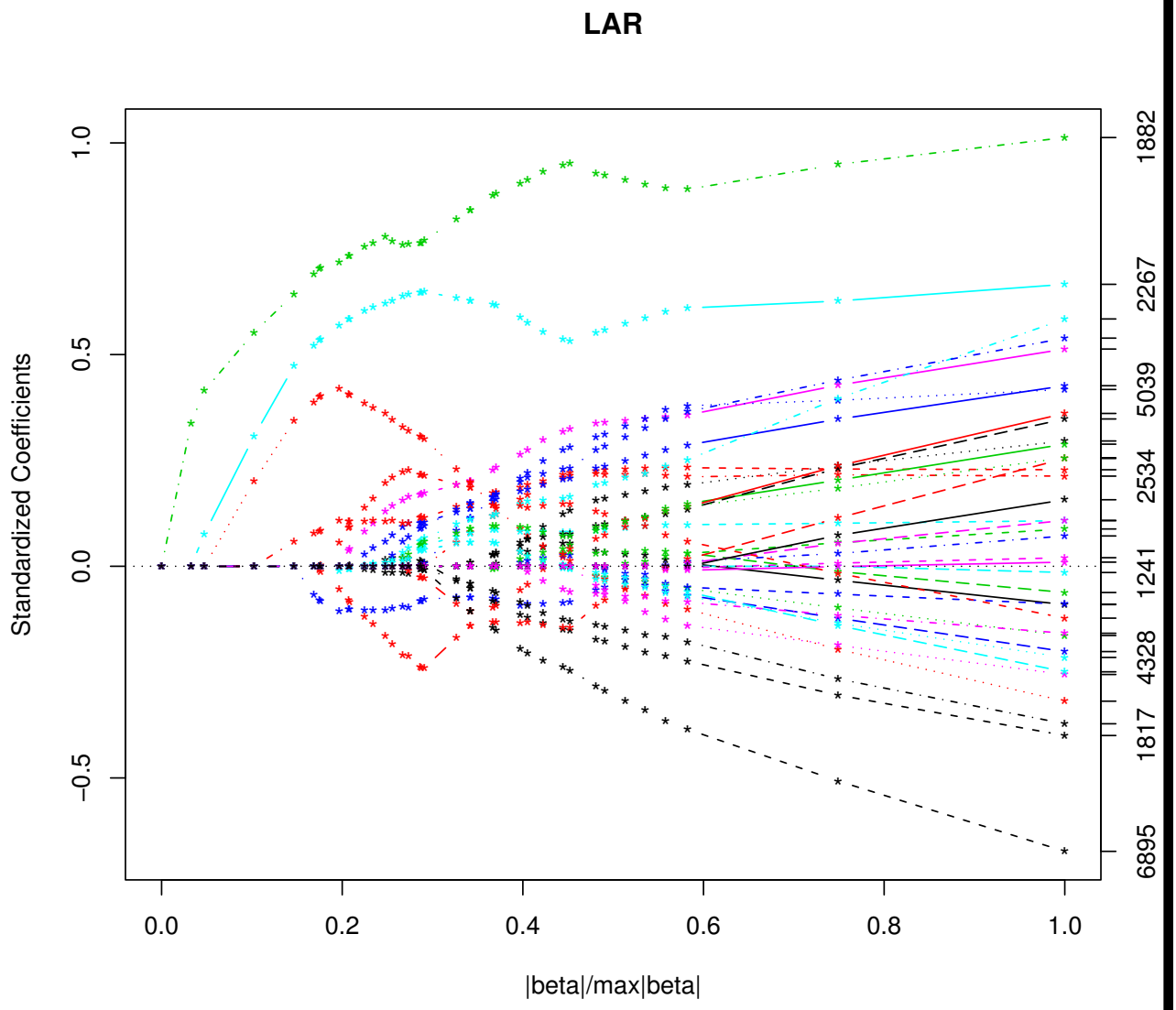
Least Angle Regression — LAR

Like a "more democratic" version of forward stepwise regression.

1. Start with $r = y$, $\hat{\beta}_1, \hat{\beta}_2, \ldots \hat{\beta}_p = 0$. Assume x_j standardized.
2. Find predictor x_j most correlated with r.
3. Increase β_j in the direction of $\text{sign}(\text{corr}(r, x_j))$ until some other competitor x_k has as much correlation with current residual as does x_j.
4. Move $(\hat{\beta}_j, \hat{\beta}_k)$ in the joint least squares direction for (x_j, x_k) until some other competitor x_ℓ has as much correlation with the current residual
5. Continue in this way until all predictors have been entered. Stop when $\text{corr}(r, x_j) = 0 \; \forall \; j$, i.e. OLS solution.



The LAR direction $\mathbf{u}_2$ at step 2 makes an equal angle with $\mathbf{x}_1$ and $\mathbf{x}_2$.


LAR
Standardized Coefficients
1.0
0.5
0.0
−0.5
0.0
0.2
0.4
0.6
0.8
1.0
|beta|/max|beta|
1882
2267
5039
2534
1241
4328
1817
6895

LAR gives the lasso path

- Start with LAR. If a coefficient crosses zero, stop. Drop that predictor, recompute the best direction and continue. This gives the Lasso path

Software for R and Splus

`lars()` function fits all three models: `lasso`, `lar` or `forward.stagewise`. Methods for prediction, plotting, and cross-validation. Detailed documentation provided. Visit
`www-stat.stanford.edu/~hastie/Papers/#LARS`

Main computations involve least squares fitting using the *active set* of variables. Computations managed by updating the Choleski R matrix (and frequent downdating for lasso and forward stagewise).

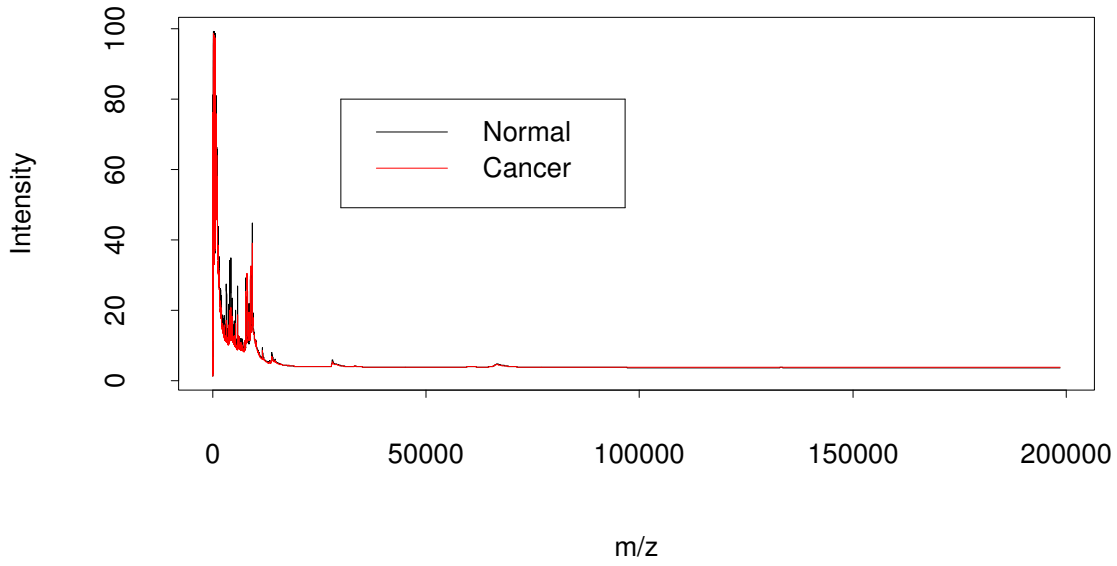
The fused lasso

- Note that when $p > N$, at most N lasso coefficients can be non-zero. Doesn't seem reasonable- too sparse!
- If there are many correlated features, Lasso gives only one of them a non-zero coefficient
- Maybe correlated featured should have similar coefficients

Protein mass spectroscopy

Blood serum samples from 157 healthy patients and 167 with cancer;

Intensity measurements at 48,538 m/z (mass/charge) sites.



Fused lasso

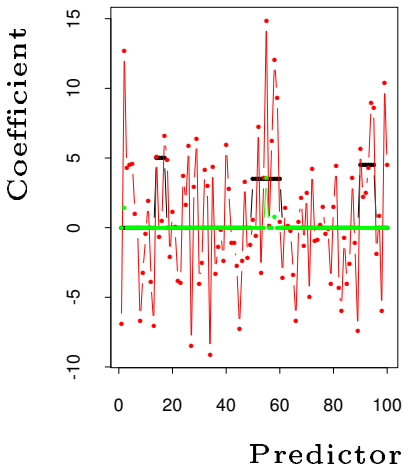
$$\hat{\beta} = \text{argmin} \sum_i (y_i - \sum_j x_{ij}\beta_j)^2$$

$$\text{subject to} \sum_{j=1}^{p} |\beta_j| \leq s_1$$

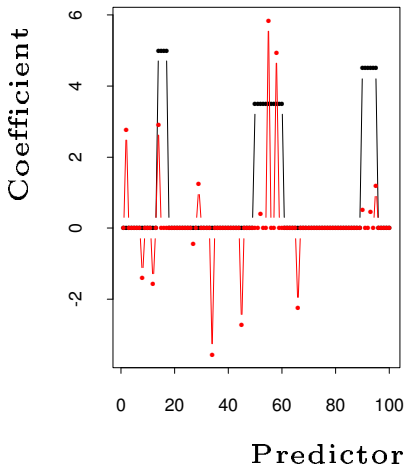
$$\text{and} \sum_{j=2}^{p} |\beta_j - \beta_{j-1}| \leq s_2$$

Example: $p = 100,\ N = 20$

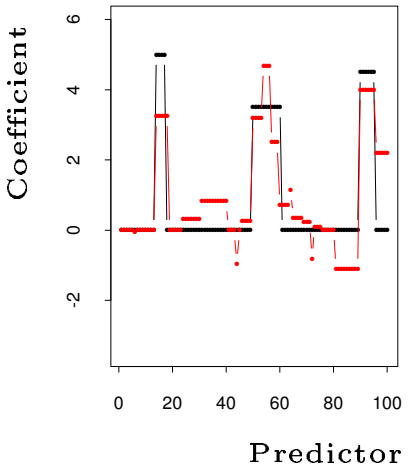
Univariate and thresholded



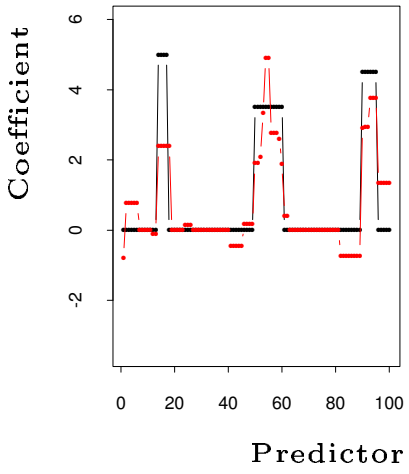
Lasso, $s_1 = 35.6$



Fusion, $s_2 = 27.7$

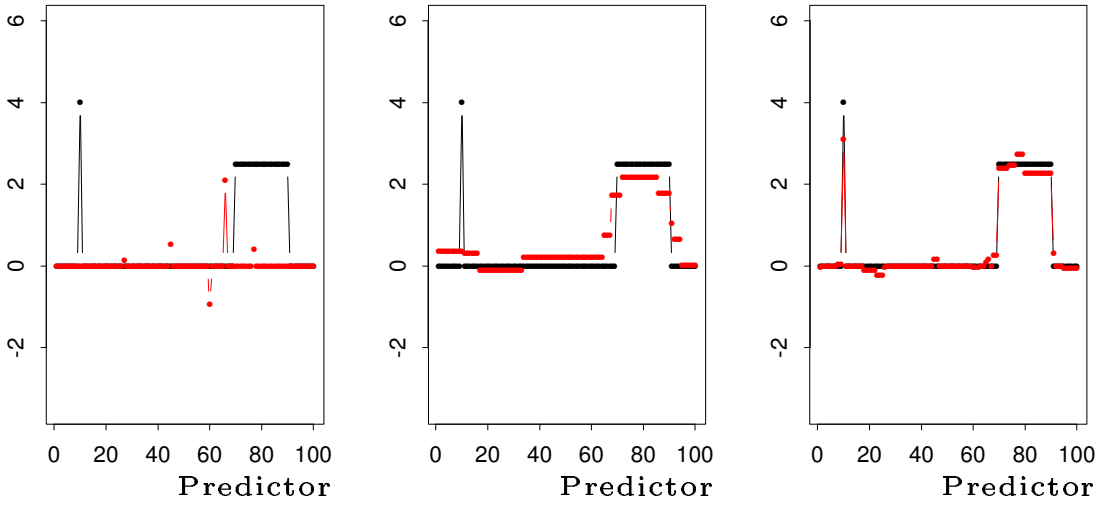


Fused lasso, $s_1 = 85.5, s_2 = 26.0$



Another example

Lasso $s_1 = 4.2$ Fusion $s_2 = 5.2$ Fused lasso $s_1 = 56.5, s_2 = .3$



Computational approach

- For Fixed s_1, s_2, fused lasso criterion leads to a quadratic programming problem.
- For large p, the problem is difficult to solve and special care must be taken to avoid the use of p^2 storage elements. We use the two-phase active set algorithm `sqopt` of Gill et al. (1999), which is designed for quadratic programming problems with sparse linear constraints.
- We have not yet been able to derive a LAR-like algorithm for generating paths of solutions.
- We have an adhoc method for search through the feasible (s_1, s_2) region.
- Speed is currently a limiting factor. Can solve confortably for $p = 2000$ but not $p = 20,000$.

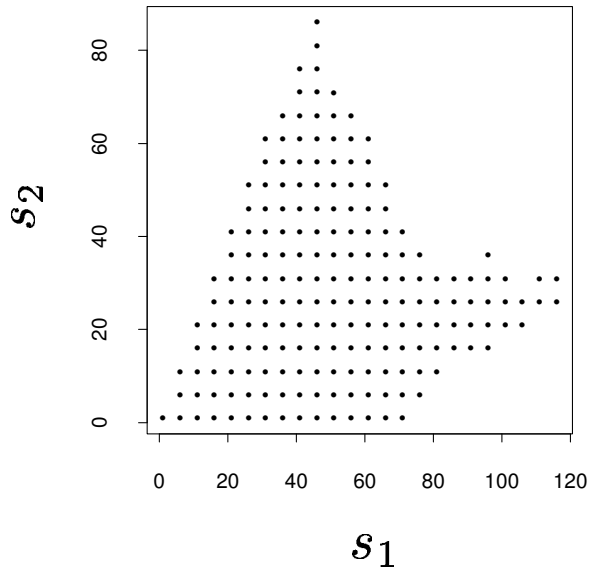
Quadratic programming formulation

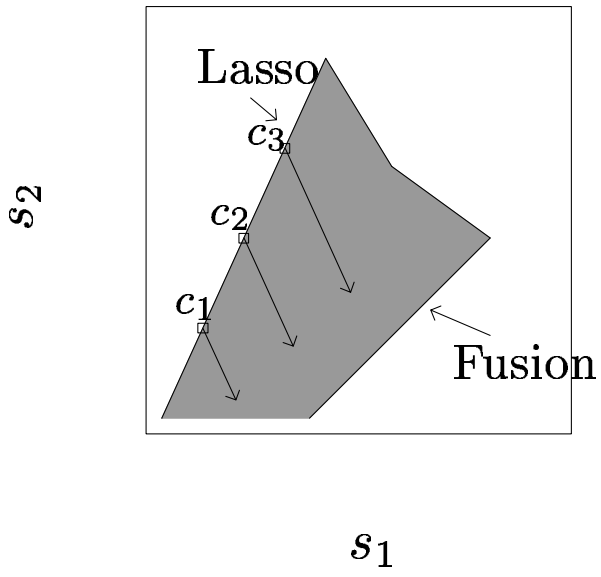
Let $\beta_j = \beta_j^+ - \beta_j^-$, with $\beta_j^+, \beta_j^- \geq 0$. Define $\theta_j = \beta_j - \beta_{j-1}$ for $j > 1$ and $\theta_1 = \beta_1$. Let $\theta_j = \theta_j^+ - \theta_j^-$ with $\theta_j^+, \theta_j^- \geq 0$. Let L be a $p \times p$ matrix with $L_{ii} = 1$ and $L_{i+1,i} = -1$, and $L_{ij} = 0$ otherwise.

$$\hat{\beta} = \text{argmin}(y - X\beta)^T(y - X\beta) \text{ subject to}$$

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \leq \begin{pmatrix} L & 0 & 0 & -I & I \\ I & -I & I & 0 & 0 \\ 0 & e^T & e^T & 0 & 0 \\ 0 & 0 & 0 & e^T & e^T \end{pmatrix} \begin{pmatrix} \beta \\ \beta^+ \\ \beta^- \\ \theta^+ \\ \theta^- \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ s_1 \\ s_2 \end{pmatrix},$$

in addition to the non-negativity constraints $\beta_j^+, \beta_j^-, \theta_j^+, \theta_j^- \geq 0$.

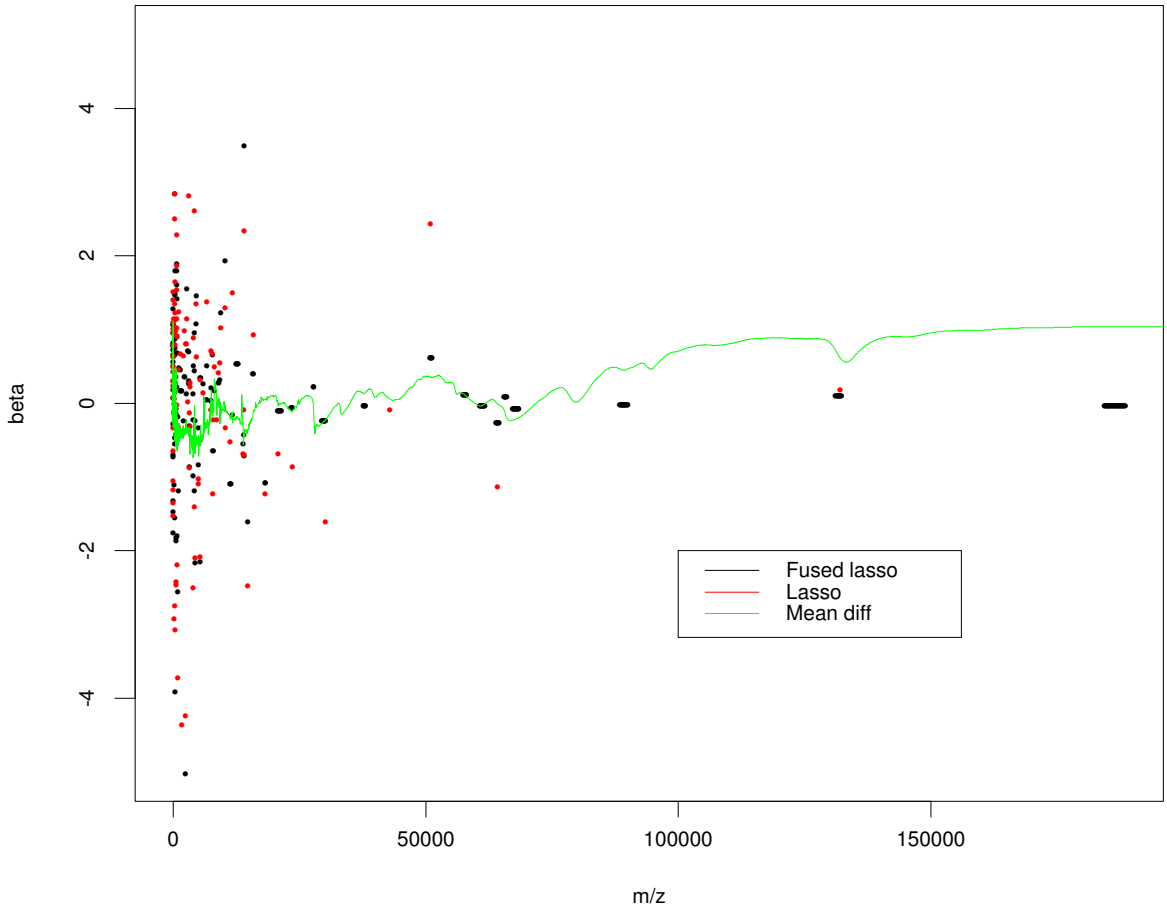

s_2
0
20
40
60
80
0
20
40
60
80
100
120
s_1


s_2
Lasso
c_3
c_2
c_1
Fusion
s_1

Degrees of freedom

- Defined as $\text{df} = \sum \text{cov}(y_i, \hat{y}_i)/\sigma^2$.
- For Lasso, df $\approx$ number of non-zero coefficients $\leq \min(N, p)$
- For fused lasso. df $\approx$ number of non-zero plateaus.

Prostate data results



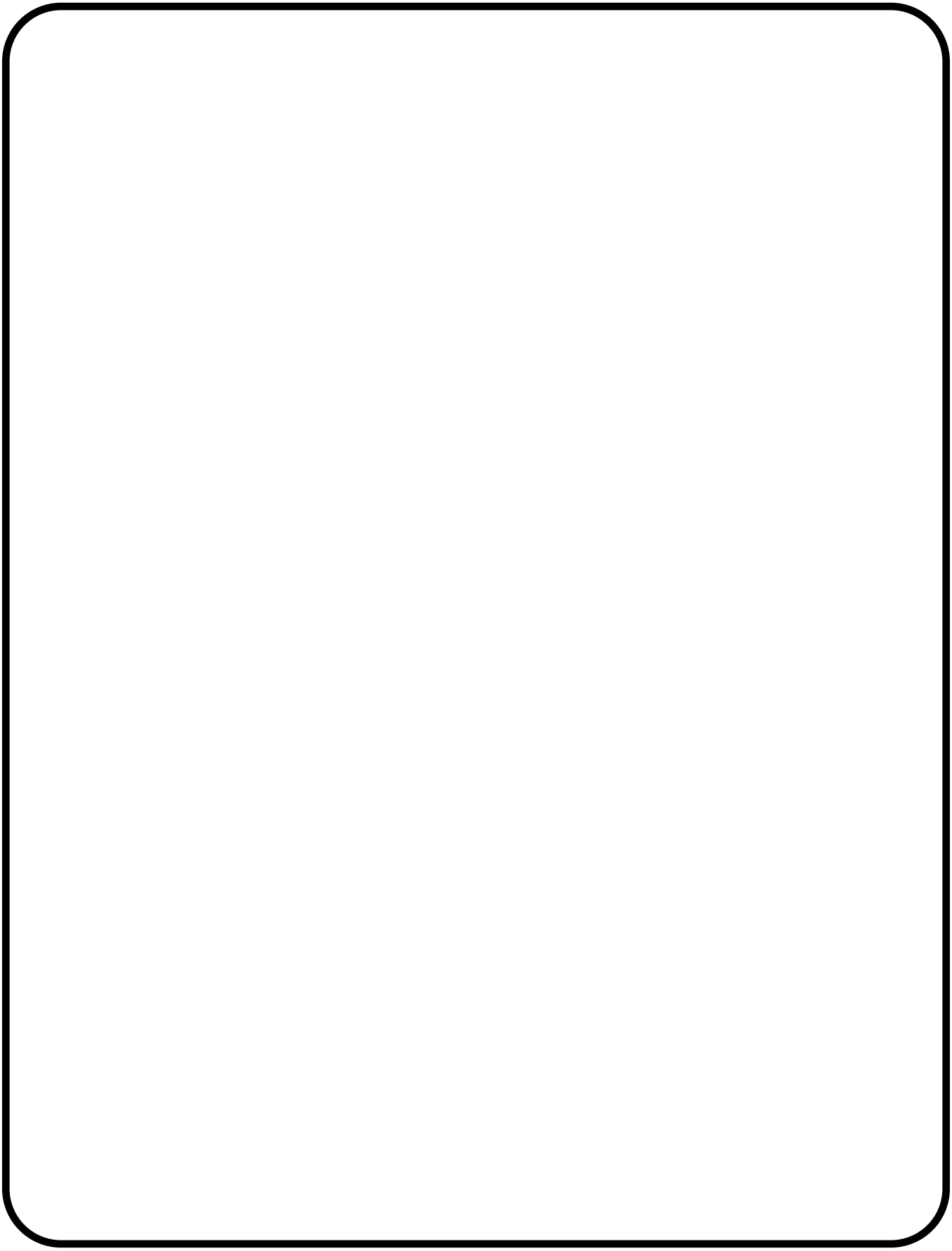
Method	Test err/108	df	# sites	s_1	s_2
Lasso	6	116	116	144	262
Fusion	19	168	171	175	200
Fused Lasso	6	122	344	184	222

Application to microarray data

- Apply hierarchical clustering to genes, to estimate an ordering for the genes.
- Use this ordering for fused lasso

Results for leukemia microarray example

Method	10-fold CV error	Test Error	# genes
Golub (50 genes)	3/38	4/34	50
Lasso 37 df	1/38	1/34	37
Fused lasso 38 df	1/38	2/34	135
Fused lasso 20 df	1/38	4/34	737



Gene #	Las	Fus Las	Gene #	Las	Fus Las	Gene #
9	0.00000	0.00203	421	-0.08874	-0.02506	765
10	0.00000	0.00495	422	0.00000	-0.00110	766
11	0.00000	0.00495				767
12	0.00000	0.00495	475	-0.01734	0.00000	768
13	0.00000	0.00495				769
14	0.00000	0.00495	522	0.00000	-0.00907	770
15	0.00000	0.00495	523	0.00000	-0.00907	771
			524	0.00000	-0.00907	772
22	0.01923	0.00745	525	0.00000	-0.00907	
23	0.00000	0.00745	526	0.00000	-0.00907	788
24	0.00000	0.00745	527	0.00000	-0.00907	
25	0.00000	0.00745	528	0.00000	-0.00907	798
26	0.00000	0.00745				799
27	0.01157	0.00294	530	0.01062	0.00000	800
31	-0.00227	0.00000	563	0.00000	-0.02018	815
			564	0.00000	-0.02018	
39	-0.00992	0.00000	565	0.00000	-0.02018	835
			566	0.00000	-0.02018	836
44	-0.00181	0.00000	567	0.00000	-0.02018	837
						838
54	0.00000	-0.00830	570	0.00000	-0.00005	839
55	-0.02027	-0.00830	571	0.00000	-0.00005	840
			572	-0.00764	-0.00005	
59	-0.01383	0.00000				881
			579	0.00000	-0.00119	882
74	-0.00014	0.00000	580	0.00000	-0.00119	883
75	0.00000	-0.00537	581	0.00000	-0.00119	884
76	0.00000	-0.00537	582	0.00000	-0.00119	885
			583	-0.00220	-0.00119	
143	0.03479	0.01008				906
144	0.00000	0.01008	616	0.01102	0.01743	907
145	0.00000	0.01008	617	0.00495	0.01743	
146	0.00000	0.01008				926
147	0.00000	0.01008	670	-0.00496	0.00000	927
						928
171	0.00000	0.00664	675	0.00000	0.00000	929
172	0.00057	0.00664	676	0.00000	0.00000	930
173	0.00000	0.00664	677	0.00000	0.00000	931

Future directions

- Develop faster algorithm and friendly interface
- get more experience with real data

References

Gill, P., Murray, W. & Saunders, M. (1999), Users guide for sqopt 5.3: a fortran package for large-scale linear and quadratic programming., Technical report, Stanford University.