Computationally Tractable Methods for High-Dimensional Data

Peter Bühlmann

Seminar für Statistik, ETH Zürich

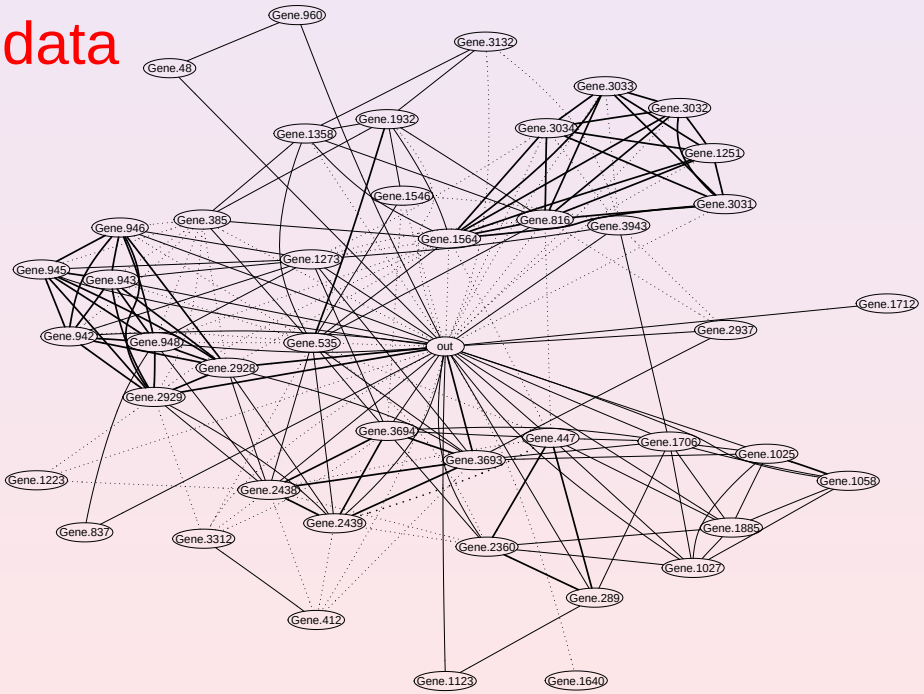
August 2008

Riboflavin production in Bacillus Subtilis

in collaboration with DSM (former Roche Vitamines)

response variables $Y \in \mathbb{R}$: riboflavin production rate
covariates $X \in \mathbb{R}^p$: expressions from $p = 4088$ genes
sample size $n = 72$ from a "homogeneous" population of genetically engineered mutants of Bacillus Subtilis

$p \gg n$ and
high quality data



goal: improve riboflavin production rate of Bacillus Subtilis

statistical goal:

quantify importance of genes/variables in terms of association (i.e. regression)

$\rightsquigarrow$ new interesting genes
which we should knock-down or enhance

my primary interest:

variable selection / variable importance

but many of the concepts work also for the easier problem of prediction

my primary interest:

variable selection / variable importance

but many of the concepts work also for the easier problem of prediction

High-dimensional data

$(X_1, Y_1), \ldots, (X_n, Y_n)$ i.i.d. or stationary

X_i p-dimensional predictor variable

Y_i response variable, e.g. $Y_i \in \mathbb{R}$ or $Y_i \in \{0, 1\}$

high-dimensional: $p \gg n$

areas of application:
biology, astronomy, marketing research, text classification, econometrics, ...

High-dimensional data

$(X_1, Y_1), \ldots, (X_n, Y_n)$ i.i.d. or stationary

X_i p-dimensional predictor variable

Y_i response variable, e.g. $Y_i \in \mathbb{R}$ or $Y_i \in \{0, 1\}$

high-dimensional: $p \gg n$

areas of application:
biology, astronomy, marketing research, text classification, econometrics, ...

High-dimensional linear and generalized linear models

$$Y_i = (\beta_0 +) \sum_{j=1}^{p} \beta_j X_i^{(j)} + \epsilon_i, \; i = 1, \ldots, n, \; p \gg n$$

in short: $Y = X\beta + \epsilon$

Y_i independent, $\mathbb{E}[Y_i | X_i = x] = \mu(x)$,

$$\eta(x) = g(\mu(x)) = (\beta_0 +) \sum_{j=1}^{p} \beta_j x^{(j)}, \; p \gg n$$

goal: estimation of β

- variable selection: $\mathcal{A}_{true} = \{j; \; \beta_j \neq 0\}$
- prediction: e.g. $\beta^T X_{new}$

We need to regularize

if true β_{true} is sparse w.r.t.

- $\|\beta_{\text{true}}\|_0 =$ number of non-zero coefficients
  $\rightsquigarrow$ penalize with the $\|\cdot\|_0$-norm:
  $\text{argmin}_\beta(-2 \text{ log-likelihood}(\beta) + \lambda\|\beta\|_0)$, e.g. AIC, BIC
  $\rightsquigarrow$ computationally infeasible if p is large (2^p sub-models)
- $\|\beta_{\text{true}}\|_1 = \sum_{j=1}^p |\beta_{\text{true},j}|$
  $\rightsquigarrow$ penalize with the $\|\cdot\|_1$-norm, i.e. Lasso:
  $\text{argmin}_\beta(-2 \text{ log-likelihood}(\beta) + \lambda\|\beta\|_1)$
  $\rightsquigarrow$ convex optimization: computationally feasible for large p

alternative approaches include:
Bayesian methods for regularization
$\rightsquigarrow$ computationally hard (and computation is approximate)

We need to regularize

if true β_{true} is sparse w.r.t.

- $\|\beta_{\text{true}}\|_0 =$ number of non-zero coefficients
  $\rightsquigarrow$ penalize with the $\|\cdot\|_0$-norm:
  $\text{argmin}_\beta(-2 \text{ log-likelihood}(\beta) + \lambda\|\beta\|_0)$, e.g. AIC, BIC
  $\rightsquigarrow$ computationally infeasible if p is large (2^p sub-models)
- $\|\beta_{\text{true}}\|_1 = \sum_{j=1}^p |\beta_{\text{true},j}|$
  $\rightsquigarrow$ penalize with the $\|\cdot\|_1$-norm, i.e. Lasso:
  $\text{argmin}_\beta(-2 \text{ log-likelihood}(\beta) + \lambda\|\beta\|_1)$
  $\rightsquigarrow$ convex optimization: computationally feasible for large p

alternative approaches include:
Bayesian methods for regularization
$\rightsquigarrow$ computationally hard (and computation is approximate)

Short review on Lasso

for linear models; analogous results for GLM's

Lasso for linear models (Tibshirani, 1996)

$$\hat{\beta}(\lambda) = \text{argmin}_\beta(n^{-1}\|Y - X\beta\|^2 + \underbrace{\lambda}_{\geq 0} \underbrace{\|\beta\|_1}_{\sum_{j=1}^p |\beta_j|})$$

$\rightsquigarrow$ convex optimization problem

- Lasso does variable selection
  some of the $\hat{\beta}_j(\lambda) = 0$
  (because of "ℓ^1-geometry")
- $\hat{\beta}(\lambda)$ is (typically) a shrunken LS-estimate

Lasso for variable selection:

$$\hat{\mathcal{A}}(\lambda) = \{j;\ \hat{\beta}_j(\lambda) \neq 0\}$$

no significance testing involved

computationally tractable (convex optimization only)

whereas $\|\cdot\|_0$-norm penalty methods (AIC, BIC) are computationally infeasible (2^p sub-models)

Why the Lasso/ℓ^1-hype?

among other things (which will be discussed later)

ℓ^1-penalty approach approximates $\underbrace{\ell^0\text{-penalty problem}}_{\text{what we usually want}}$

consider underdetermined system of linear equations:

$$A_{p\times p}\beta_{p\times 1} = b_{p\times 1}, \quad \operatorname{rank}(A) = m < p$$

ℓ^0-penalty-problem: solve for β which is sparsest w.r.t. $\|\beta\|_0$
i.e. "Occam's razor"

Donoho & Elad (2002), ...: if A is not too ill-conditioned (in the sense of linear dependence of sub-matrices)

$$\begin{aligned} & \text{sparsest solution } \beta \text{ w.r.t. } \|\cdot\|_0\text{-norm} \\ = \; & \underbrace{\text{sparsest solution } \beta \text{ w.r.t. } \|\cdot\|_1\text{-norm}}_{\text{amounts to a convex optimization}} \end{aligned}$$

Why the Lasso/ℓ^1-hype?

among other things (which will be discussed later)

ℓ^1-penalty approach approximates $\underbrace{\ell^0\text{-penalty problem}}_{\text{what we usually want}}$

consider underdetermined system of linear equations:

$$A_{p\times p}\beta_{p\times 1} = b_{p\times 1}, \quad \text{rank}(A) = m < p$$

ℓ^0-penalty-problem: solve for β which is sparsest w.r.t. $\|\beta\|_0$
i.e. "Occam's razor"

Donoho & Elad (2002), ...: if A is not too ill-conditioned (in the sense of linear dependence of sub-matrices)

$$\begin{aligned} & \text{sparsest solution } \beta \text{ w.r.t. } \|\cdot\|_0\text{-norm} \\ = \; & \underbrace{\text{sparsest solution } \beta \text{ w.r.t. } \|\cdot\|_1\text{-norm}}_{\text{amounts to a convex optimization}} \end{aligned}$$

and also Boosting $\approx$ Lasso-type methods will be useful

What else do we know from theory?

assumptions: linear model $Y = X\beta + \varepsilon$ (or GLM)

- $p = p_n = O(n^\alpha)$ for some $\alpha < \infty$ (high-dimensional)
- $\|\beta\|_0 =$ no. of non-zero β_j's $= o(n)$ (sparse)
- conditions on the design matrix X
  ensuring that design matrix doesn't exhibit "strong linear dependence"

rate-optimality up to $\log(p)$-term:
under "coherence conditions" for the design matrix,
and for suitable λ

$$\mathbb{E}[\|\hat{\beta}(\lambda) - \beta\|_2^2] \leq C\sigma^2 \frac{\|\beta\|_0 \log(p_n)}{n}$$

(e.g. Meinshausen & Yu, 2007)

note: for classical situation with $p = \|\beta\|_0 < n$

$$\mathbb{E}[\|\hat{\beta}_{\mathrm{OLS}} - \beta\|_2^2] = \sigma^2 \frac{p}{n} = \sigma^2 \frac{\|\beta\|_0}{n}$$

consistent variable selection:
under restrictive design conditions (i.e. “neighborhood stability”), and for suitable λ,

$$\mathbb{P}[\hat{\mathcal{A}}(\lambda) = \mathcal{A}_{true}] = 1 - O(\exp(-Cn^{1-\delta}))$$

(Meinshausen & PB, 2006)

variable screening property:
under “coherence conditions” for the design matrix (weaker than neighborhood stability), and for suitable λ

$$\mathbb{P}[\hat{\mathcal{A}}(\lambda) \supseteq \mathcal{A}_{true}] \to 1 \ (n \to \infty)$$

(Meinshausen & Yu, 2007;...)

in addition: for prediction-optimal λ^* (and nice designs)

Lasso yields too large models

$$\mathbb{P}[\underbrace{\hat{\mathcal{A}}(\lambda^*)}_{|\hat{\mathcal{A}}| \leq O(\min(n,p))} \supseteq \mathcal{A}_{\text{true}}] \to 1 \ (n \to \infty)$$

$\rightsquigarrow$ Lasso as an

excellent filter/screening procedure for variable selection

i.e. true model is contained in selected models from Lasso

the Lasso filter is $\underbrace{\text{easy to use,}}_{\text{prediction optimal tuning}}$

$\underbrace{\text{"computationally efficient"}}_{O(np\min(n,p))}$ and statistically accurate

in addition: for prediction-optimal λ^* (and nice designs)

Lasso yields too large models

$$\mathbb{P}[\underbrace{\hat{\mathcal{A}}(\lambda^*)}_{|\hat{\mathcal{A}}| \leq O(\min(n,p))} \supseteq \mathcal{A}_{\text{true}}] \to 1 \ (n \to \infty)$$

$\rightsquigarrow$ Lasso as an

excellent filter/screening procedure for variable selection

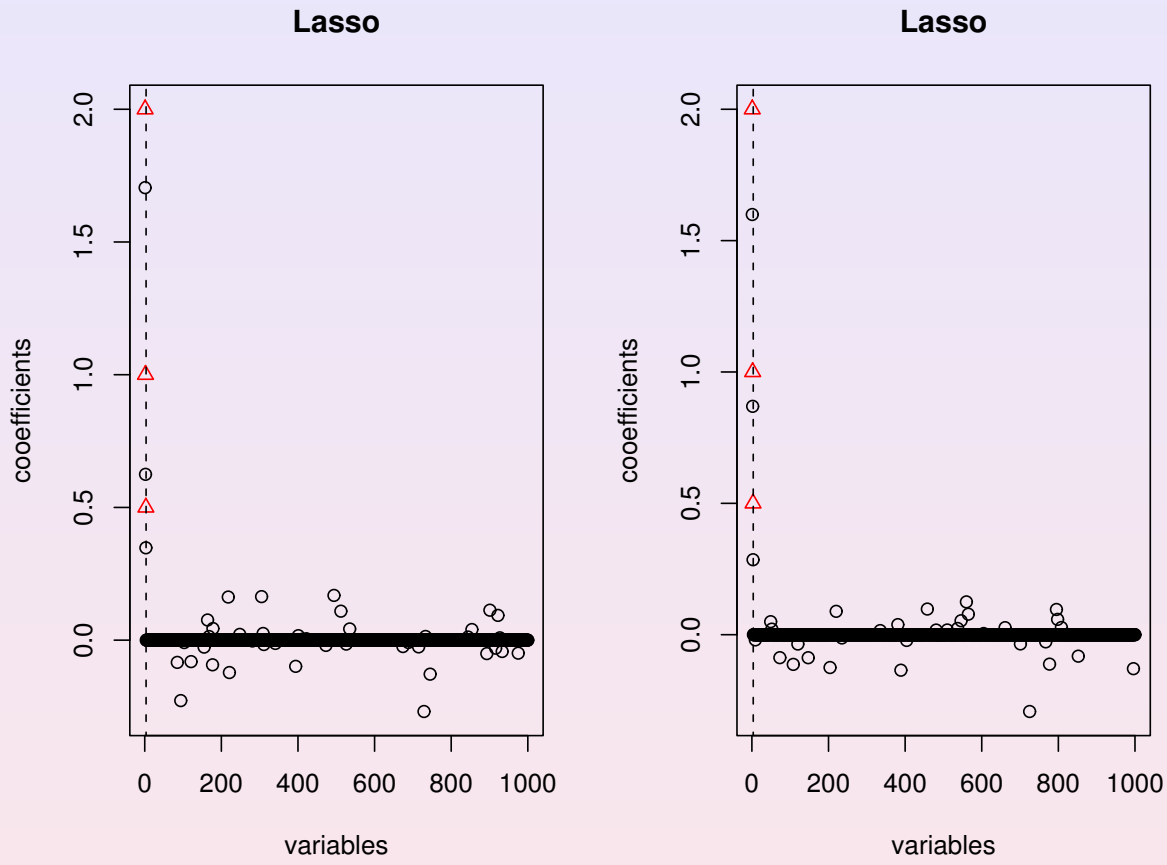
i.e. true model is contained in selected models from Lasso

the Lasso filter is $\underbrace{\text{easy to use,}}$

prediction optimal tuning

$\underbrace{\text{"computationally efficient"}}_{O(np\min(n,p))}$ and statistically accurate

$p_{eff} = 3,\ p = 1'000,\ n = 50$; 2 independent realizations



prediction-optimal tuning

44 selected variables

36 selected variables

deletion of variables with small coefficients:

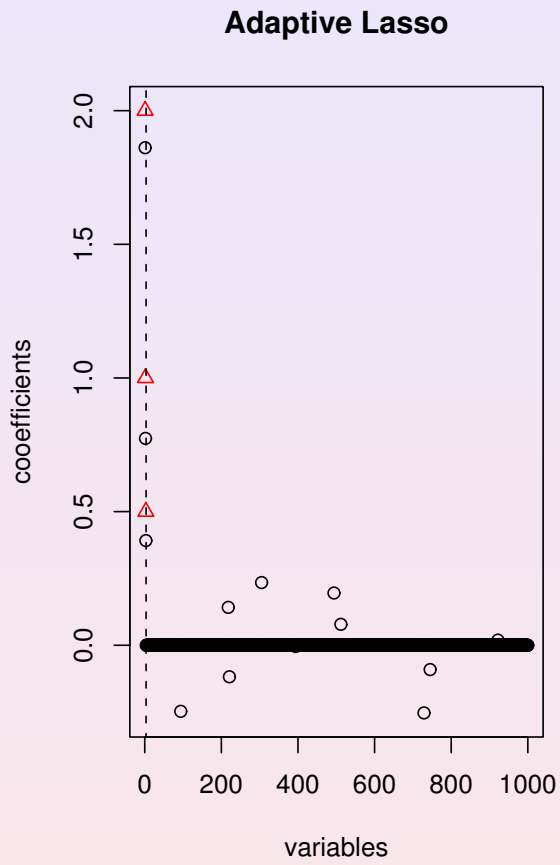
Adaptive Lasso (Zou, 2006): re-weighting the penalty function

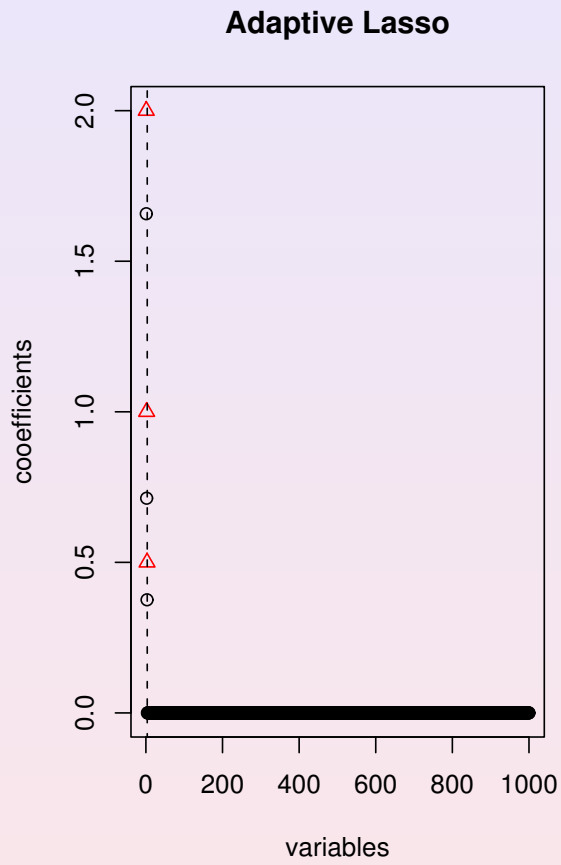
$$\hat{\beta} = \text{argmin}_{\beta} \sum_{i=1}^{n} (Y_i - (X\beta)_i)^2 + \lambda \sum_{j=1}^{p} \frac{|\beta_j|}{|\hat{\beta}_{init,j}|},$$

$\hat{\beta}_{init,j}$ from Lasso in first stage $\underbrace{\text{(or OLS if } p < n)}_{\text{Zou (2006)}}$

$\rightsquigarrow$ adaptive amount of shrinkage
reduces bias of the original Lasso procedure

$p_{eff} = 3,\ p = 1'000,\ n = 50$
same 2 independent realizations from before





13 selected variables
(Lasso: 44 sel. var.)

3 selected variables
(Lasso: 36 sel. var.)

adaptive Lasso (with prediction-optimal penalty) always yields sparser model fits than Lasso

Motif regression for transcription factor binding sites in DNA sequences

$n = 1300,\ p = 660$

	Lasso	Adaptive Lasso	Adaptive Lasso twice
no. select. variables	91	42	28
$\mathbb{E}[(\hat{Y}_{new} - Y_{new})^2]$	0.6193	0.6230	0.6226

(similar prediction performance might be due to high noise)

Computation of Adaptive Lasso

$$\hat{\beta} = \text{argmin}_\beta \sum_{i=1}^{n} (Y_i - (X\beta)_i)^2 + \lambda \sum_{j=1}^{p} \frac{|\beta_j|}{|\hat{\beta}_{init,j}|},$$

$\rightsquigarrow$ use linear transformation and Lasso computation

- transform $X^{(j)} \Longrightarrow \tilde{X}^{(j)} = X^{(j)} \cdot \hat{\beta}_{init,j} \quad (j = 1, \ldots, p)$
  $\beta_j \Longrightarrow \tilde{\beta}_j = \frac{\beta_j}{\hat{\beta}_{init,j}} \quad (j = 1, \ldots, p)$
- use Lasso-computation $\rightsquigarrow \quad \hat{\tilde{\beta}}$
- back-transform $\hat{\tilde{\beta}} \Longrightarrow \hat{\beta}_j = \hat{\tilde{\beta}}_j \cdot \hat{\beta}_{init,j} \quad (j = 1, \ldots, p)$

What we sometimes need/want in addition

- allowing for group-structure
  $\rightsquigarrow$ categorical covariates
  additive modeling (and functional data analysis, etc.)
- penalty for sparsity and smoothness
  $\rightsquigarrow$ "flexible" additive modeling
- scalable computation $\rightsquigarrow$ will be able to deal with $p \approx 10^6$

The Group Lasso (Yuan & Lin, 2006)

high-dimensional parameter vector is structured into q groups or partitions (known a-priori):

$$\mathcal{G}_1, \ldots, \mathcal{G}_q \subseteq \{1, \ldots, p\}, \text{ disjoint and } \cup_g \mathcal{G}_g = \{1, \ldots, p\}$$

corresponding coefficients: $\beta_{\mathcal{G}} = \{\beta_j;\ j \in \mathcal{G}\}$

Example: categorical covariates

$X^{(1)}, \ldots, X^{(p)}$ are factors (categorical variables)
each with 4 levels (e.g. “letters” from DNA)

for encoding a main effect: 3 parameters
for encoding a first-order interaction: 9 parameters
and so on ...

parameterization (e.g. sum contrasts) is structured as follows:

- intercept: no penalty
- main effect of $X^{(1)}$: group $\mathcal{G}_1$ with $df = 3$
- main effect of $X^{(2)}$: group $\mathcal{G}_2$ with $df = 3$
- ...
- first-order interaction of $X^{(1)}$ and $X^{(2)}$: $\mathcal{G}_{p+1}$ with $df = 9$
- ...

often, we want sparsity on the group-level
either all parameters of an effect are zero or not

often, we want sparsity on the group-level
either all parameters of an effect are zero or not

this can be achieved with the Group-Lasso penalty

$$\lambda \sum_{g=1}^{q} s(df_g) \underbrace{\|\beta_{\mathcal{G}_g}\|_2}_{\sqrt{\|\cdot\|_2^2}}$$

typically $s(df_{\mathcal{G}_g}) = \sqrt{df_{\mathcal{G}_g}}$ so that $s(df_{\mathcal{G}_g})\|\beta_{\mathcal{G}_g}\|_2 = O(df_g)$

properties of Group-Lasso penalty

- for group-sizes $|\mathcal{G}_g| \equiv 1 \rightsquigarrow$ standard Lasso-penalty
- convex penalty $\rightsquigarrow$ convex optimization for standard likelihoods (exponential family models)
- either $(\hat{\beta}_{\mathcal{G}}(\lambda))_j = 0$ or $\neq 0$ for all $j \in \mathcal{G}$
- penalty is invariant under orthonormal transformation
  e.g. invariant when requiring orthonormal parameterization for factors

asymptotically:
the Group Lasso has optimal convergence rates (for prediction);
and some variable screening properties hold as well
(Meier, van de Geer & PB, 2008)

main assumptions:

- generalized linear model with convex negative likelihood function
- $p = p_n$ with $\log(p_n)/n \to 0$ (high-dimensional)
- bounded group sizes: $\max_g df(\mathcal{G}_g) \leq C < \infty$
- number of non-zero group-effects $\leq D < \infty$ (sparsity)
  can be generalized...

$$\mathbb{E}_X |\eta_{\hat{\beta}}(X) - \eta_\beta(X)|^2 = O_P(\frac{\log(q_n)}{n}) = O_P(\frac{\log(p_n)}{n})$$

Computation: exact and approximate solution paths

see also useR!2006 (Hastie) $\rightsquigarrow$ nowadays, "we" do it differently

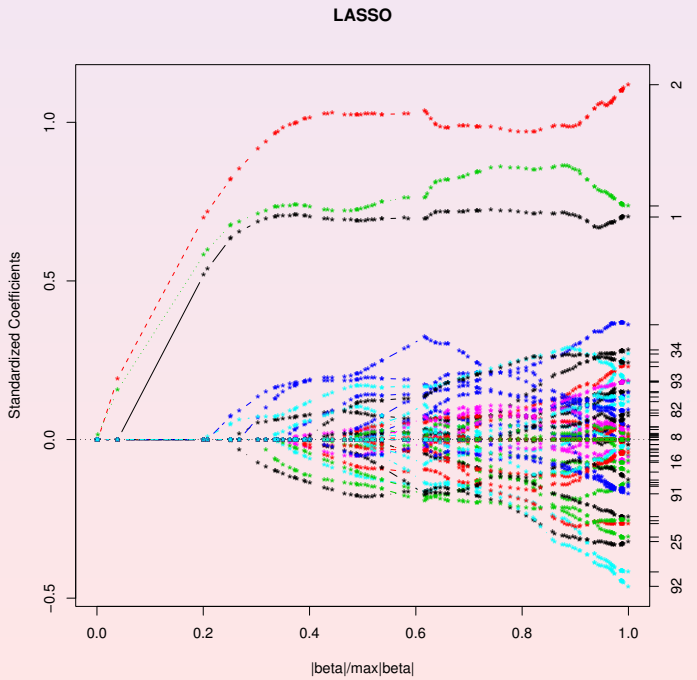
LARS algorithm (homotopy method) from Efron et al. (2004) became very popular for computing Lasso in linear model

$$\hat{\beta}(\lambda) = \text{argmin}_\beta \|Y - X\beta\|_2^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

piecewise linear solution path for $\{\hat{\beta}(\lambda);\ \lambda \in \mathbb{R}^+\}$

$$\hat{\beta}(\lambda) = \hat{\beta}(\underbrace{\lambda_k}_{\text{kink-points}}) + (\lambda - \lambda_k)\underbrace{\gamma_k}_{\in \mathbb{R}^p}$$

for $\lambda_k \le \lambda \le \lambda_{k+1}$



what we need to compute:

- kink-points: $\lambda_0 = 0 < \lambda_1 < \lambda_2 < \ldots \lambda_{\max}$
- linear coefficients $\in \mathbb{R}^p$: $\gamma_1, \ldots, \gamma_{\max}$

number of different λ_k's,γ_k's is $O(n)$

the LARS algorithm computes all these quantities in

$O(np \min(n, p))$ essential operations

i.e. linear in p if $p \gg n$

no exact piecewise linear regularization path anymore for

- Group-Lasso penalty
- non-Gaussian likelihood with Lasso (or Group-Lasso) penalty

LARS-algorithm cannot handle these problems exactly
$\rightsquigarrow$ approximate LARS-type algorithms
e.g. `pathglm` (Park and Hastie)

even more: if p is very large (e.g. $p \approx 10^6$), LARS is slow for Lasso in linear models

other algorithms are needed...

no exact piecewise linear regularization path anymore for

- Group-Lasso penalty
- non-Gaussian likelihood with Lasso (or Group-Lasso) penalty

LARS-algorithm cannot handle these problems exactly
$\rightsquigarrow$ approximate LARS-type algorithms
e.g. `pathglm` (Park and Hastie)

even more: if p is very large (e.g. $p \approx 10^6$), LARS is slow for Lasso in linear models

other algorithms are needed...

Fast computation: coordinatewise descent
R packages `grplasso` and `glmnet`

coordinatewise approaches ("Gauss-Seidel") are re-discovered as efficient tools for Lasso-type convex optimization problems

Fu (1998) called it the "shooting algorithm"

note: "coordinatewise" because e.g. no gradient is available

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$) keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

$$\text{Lasso: } (\beta_1, \beta_2 = \beta_2^{(0)}, \ldots, \beta_j = \beta_j^{(0)}, \ldots, \beta_p = \beta_p^{(0)})$$

$$\uparrow$$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$) keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

$$\text{Lasso: } (\beta_1 = \beta_1^{(1)}, \beta_2, \ldots, \beta_j = \beta_j^{(0)}, \ldots, \beta_p = \beta_p^{(0)})$$

$$\uparrow$$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$) keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

$$\text{Lasso: } (\beta_1 = \beta_1^{(1)}, \beta_2 = \beta_2^{(1)}, \ldots, \beta_j, \ldots, \beta_p = \beta_p^{(0)})$$

$$\uparrow$$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$) keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

$$\text{Lasso: } (\beta_1 = \beta_1^{(1)}, \beta_2 = \beta_2^{(1)}, \ldots, \beta_j = \beta_j^{(1)}, \ldots, \beta_p)$$

$$\uparrow$$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$) keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

$$\text{Lasso: } (\beta_1, \beta_2 = \beta_2^{(1)}, \ldots, \beta_j = \beta_j^{(1)}, \ldots, \beta_p = \beta_p^{(1)})$$

$$\uparrow$$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$)
  keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

Group Lasso: $(\beta_{\mathcal{G}_1}, \beta_{\mathcal{G}_2} = \beta^{(0)}_{\mathcal{G}_2}, \ldots, \beta_{\mathcal{G}_j} = \beta^{(0)}_{\mathcal{G}_j}, \ldots, \beta_{\mathcal{G}_q} = \beta^{(0)}_{\mathcal{G}_q})$

$\uparrow$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$)
  keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

Group Lasso: $(\beta_{\mathcal{G}_1} = \beta^{(1)}_{\mathcal{G}1}, \beta_{\mathcal{G}_2}, \ldots, \beta_{\mathcal{G}_j} = \beta^{(0)}_{\mathcal{G}_j}, \ldots, \beta_{\mathcal{G}_q} = \beta^{(0)}_{\mathcal{G}_q})$

$\uparrow$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$)
  keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

Group Lasso: $(\beta_{\mathcal{G}_1} = \beta^{(1)}_{\mathcal{G}1}, \beta_{\mathcal{G}_2} = \beta^{(1)}_{\mathcal{G}_2}, \ldots, \beta_{\mathcal{G}_j}, \ldots, \beta_{\mathcal{G}_q} = \beta^{(0)}_{\mathcal{G}_q})$

$\uparrow$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$)
  keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

Group Lasso: $(\beta_{\mathcal{G}_1} = \beta^{(1)}_{\mathcal{G}1}, \beta_{\mathcal{G}_2} = \beta^{(1)}_{\mathcal{G}_2}, \ldots, \beta_{\mathcal{G}_j} = \beta^{(1)}_{\mathcal{G}_j}, \ldots, \beta_{\mathcal{G}_q})$

coordinatewise descent: a generic description

for both, Lasso or Group-Lasso problems

- cycle through all coordinates $j = 1, \ldots, p, 1, 2, \ldots$
  or $j = 1, \ldots, q, 1, 2, \ldots$
- optimize the penalized log-likelihood w.r.t. β_j (or $\beta_{\mathcal{G}_j}$)
  keeping all other coefficients $\beta_k, k \neq j$ (or $k \neq \mathcal{G}_j$) fixed

Group Lasso: $(\beta_{\mathcal{G}_1}, \beta_{\mathcal{G}_2} = \beta^{(1)}_{\mathcal{G}_2}, \ldots, \beta_{\mathcal{G}_j} = \beta^{(1)}_{\mathcal{G}_j}, \ldots, \beta_{\mathcal{G}_q} = \beta^{(1)}_{\mathcal{G}_q})$

$\uparrow$

coordinatewise descent for Gaussian likelihood (squared error loss)

- coordinatewise up-dates are easy: closed-form solutions exist
- numerical convergence can be easily proved using theory from Tseng (2001)

Coordinatewise descent for generalized linear models

(with non-Gaussian, convex negative log-likelihood)

difficulty:
coordinatewise/groupwise up-dates: no closed-form solution exists

strategy which is fast: improve every coordinate/group numerically, but not until numerical convergence

- use quadratic approximation of log-likelihood function for improving/optimization of a single coordinate
- theory from Tseng & Yun (2007) $\rightsquigarrow$ numerical convergence can be proved

further tricks (Meier, van de Geer & PB, 2008)

- after a few runs, cycle only around the active set (where coefficient is non-zero) and visit the remaining variables only from time to time (e.g. every 10th time)
  $\rightsquigarrow$ very fast algorithm for sparse problems
- don't up-date the quadratic approximation at each step
  a rough approximation will do it
  in fact: can work with quadratic approximation from previous λ value
- for all grid-values of penalty parameters
  $\lambda_1 < \lambda_2 < \ldots < \lambda_m = \lambda_{\max}$
  warm-starts: $\hat{\beta}(\lambda_k)$ is used as initial value in the optimization for $\hat{\beta}(\lambda_{k-1})$

all these "tricks" are mathematically justifiable: can still prove numerical convergence (Meier, van de Geer & PB, 2008)

Software in R

for fast coordinatewise descent

- `grplasso` (Meier, 2006) for Group-Lasso problems
  statistical and algorithmic theory in
  Meier, van de Geer & PB (2008)
- `glmnet` (Friedman, Hastie & Tibshirani, 2007) for Lasso and Elastic net
  using exactly the building blocks from our approach...

other software

Madigan and co-workers:

Bayesian Logistic Regression (BBR, BMR, BXR)

`http://www.bayesianregression.org/`

How fast?

logistic case: $p = 10^6$, $n = 100$
group-size = 20, sparsity: 2 active groups = 40 parameters

for 10 different λ-values
CPU using `grplasso`: 203.16 seconds $\approx$ 3.5 minutes
(dual core processor with 2.6 GHz and 32 GB RAM)

we can easily deal today with predictors in the Mega's
i.e. $p \approx 10^6 - 10^7$

How fast?

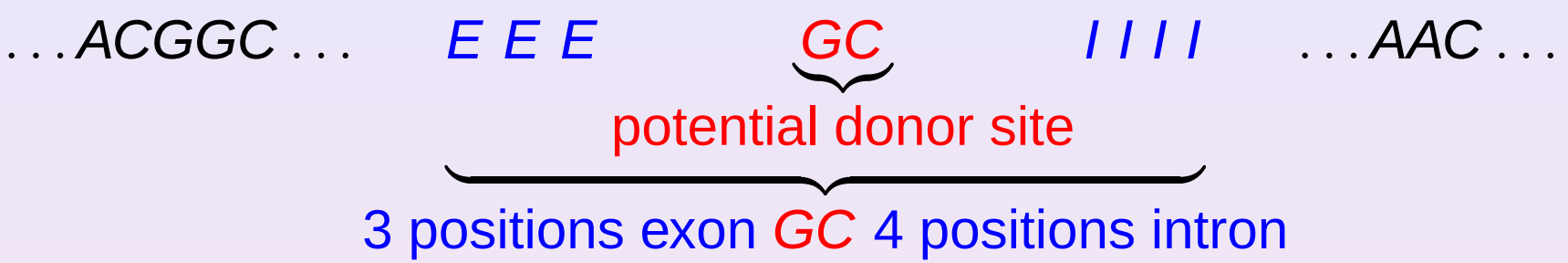
logistic case: $p = 10^6$, $n = 100$
group-size = 20, sparsity: 2 active groups = 40 parameters

for 10 different λ-values
CPU using `grplasso`: 203.16 seconds $\approx$ 3.5 minutes
(dual core processor with 2.6 GHz and 32 GB RAM)

we can easily deal today with predictors in the Mega's
i.e. $p \approx 10^6 - 10^7$

DNA splice site detection: (mainly) prediction problem

DNA sequence



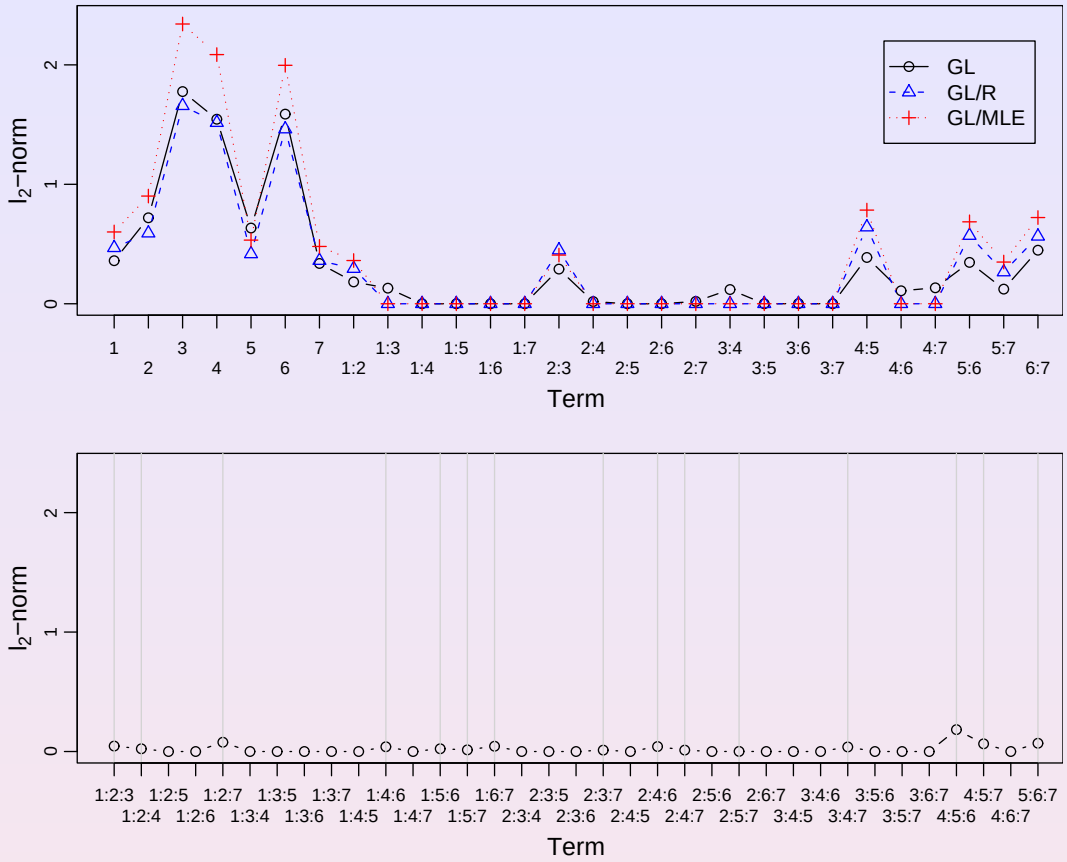
response $Y \in \{0, 1\}$: splice or non-splice site
predictor variables: 7 factors each having 4 levels
(full dimension: $4^7 = 16'384$)
data:

training:	$5'610$ true splice sites
	$5'610$ non-splice sites
	plus an unbalanced validation set
test data:	$4'208$ true splice sites
	$89'717$ non-splice sites

logistic regression:

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \text{ main effects} + \text{first order interactions} + \ldots$$

use the Group-Lasso which selects whole terms



- mainly neighboring DNA positions show interactions (has been “known” and “debated”)
- no interaction among exons and introns (with Group Lasso method)
- no second-order interactions (with Group Lasso method)

predictive power:
competitive with “state to the art” maximum entropy modeling from Yeo and Burge (2004)

correlation between true and predicted class

Logistic Group Lasso	0.6593
max. entropy (Yeo and Burge)	0.6589

- our model (not necessarily the method/algorithm) is simple and has clear interpretation
- it is as good or better than many of the complicated non-Markovian stochastic process models (e.g. Zhao, Huang and Speed (2004))

The sparsity-smoothness penalty (SSP)

(whose corresponding optimization becomes again a Group-Lasso problem...)

for additive modeling in high dimensions

$$Y_i = \sum_{j=1}^{p} f_j(x_i^{(j)}) + \varepsilon_i \ (i = 1, \ldots, n)$$

$f_j : \mathbb{R} \to \mathbb{R}$ smooth univariate functions

$p \gg n$

in principle: basis expansion for every $f_j(\cdot)$ with basis functions

$$B_{1,j}, \ldots, B_{m,j} \text{ where } m = O(n) \quad (\text{or e.g. } m = O(m^{1/2}))$$
$$j = 1, \ldots, p$$

$\rightsquigarrow$ represent

$$\sum_{j=1}^{p} f_j(x^{(j)}) = \sum_{j=1}^{p} \sum_{k=1}^{m} \beta_{k,j} B_{k,j}(x^{(j)})$$

$\rightsquigarrow$ high-dimensional parametric problem

and use the Group-Lasso penalty to ensure sparsity of whole functions

$$\lambda \sum_{g=1}^{p} \| \underbrace{\beta_{\mathcal{G}_j}}_{(\beta_{1,j},\ldots,\beta_{m,j})^T} \|_2$$

drawback:
if different additive functions $f_j(\cdot)$ have very different complexity

this naive approach will not be flexible enough
and this applies also to L_2Boosting (PB & Yu, 2003)
R-package `mboost` (Hothorn et al.)

when using a large number of basis functions (large m) for achieving a high degree of flexibility
$\rightsquigarrow$ need additional control for smoothness

Sparsity-Smoothness Penalty (SSP)

(Meier, van de Geer & PB, 2008)

$$\lambda_1 \sum_{j=1}^{p} \sqrt{\|f_j\|_2^2 + \lambda_2 I^2(f_j)}$$

$$I^2(f_j) = \int (f_j''(x))^2 dx$$

where $f_j = (f_j(X_1^{(j)}), \ldots, f_j(X_n^{(j)}))^T$

$\rightsquigarrow$ SSP-penalty does variable selection ($\hat{f}_j \equiv 0$ for some j)

SSP-penalty is between COSSO (Lin & Zhang, 2006) and the SpAM approach (Ravikumar et al., 2007)

but our SSP penalty is asymptotically oracle optimal
(while this fact is unclear for other proposals)

for additive modeling:

$$\hat{f}_1, \ldots, \hat{f}_p = \text{argmin}_{f_1,\ldots,f_p} \| Y - \sum_{j=1}^{p} f_j \|_2^2 + \lambda_1 \sum_{j=1}^{p} \sqrt{\|f_j\|_2^2 + \lambda_2 I^2(f_j)}$$

or for GAM:

$$\hat{f}_1, \ldots, \hat{f}_p = \text{argmin}_{f_1,\ldots,f_p} - 2\ell(f_1, \ldots f_p) + \lambda_1 \sum_{j=1}^{p} \sqrt{\|f_j\|_2^2 + \lambda_2 I^2(f_j)}$$

assuming f_j is twice differentiable

$\rightsquigarrow$ solution is a natural cubic spline with knots at $X_i^{(j)}$

$\rightsquigarrow$ finite-dimensional parameterization with e.g. B-splines:

$$f = \sum_{j=1}^{p} f_j, \quad f_j = \underbrace{B_j}_{n \times m} \underbrace{\beta_j}_{m \times 1}$$

penalty becomes:

$$\lambda_1 \sum_{j=1}^{p} \sqrt{\|f_j\|_2^2 + \lambda_2 I^2(f_j)}$$

$$= \lambda_1 \sum_{j=1}^{p} \sqrt{\beta_j^T \underbrace{B_j^T B_j}_{\Sigma_j} \beta_j + \lambda_2 \beta_j^T \underbrace{\Omega_j}_{\text{integ. 2nd derivatives}} \beta_j}$$

$$= \lambda_1 \sum_{j=1}^{p} \sqrt{\beta_j^T \underbrace{(\Sigma_j + \lambda_2 \Omega_j)}_{A_j = A_j(\lambda_2)} \beta}$$

$\rightsquigarrow$ re-parameterize $\tilde{\beta}_j = \tilde{\beta}_j(\lambda_2) = R_j \beta_j, \qquad R_j^T R_j = A_j = A_j(\lambda_2)$ (Choleski)

penalty becomes

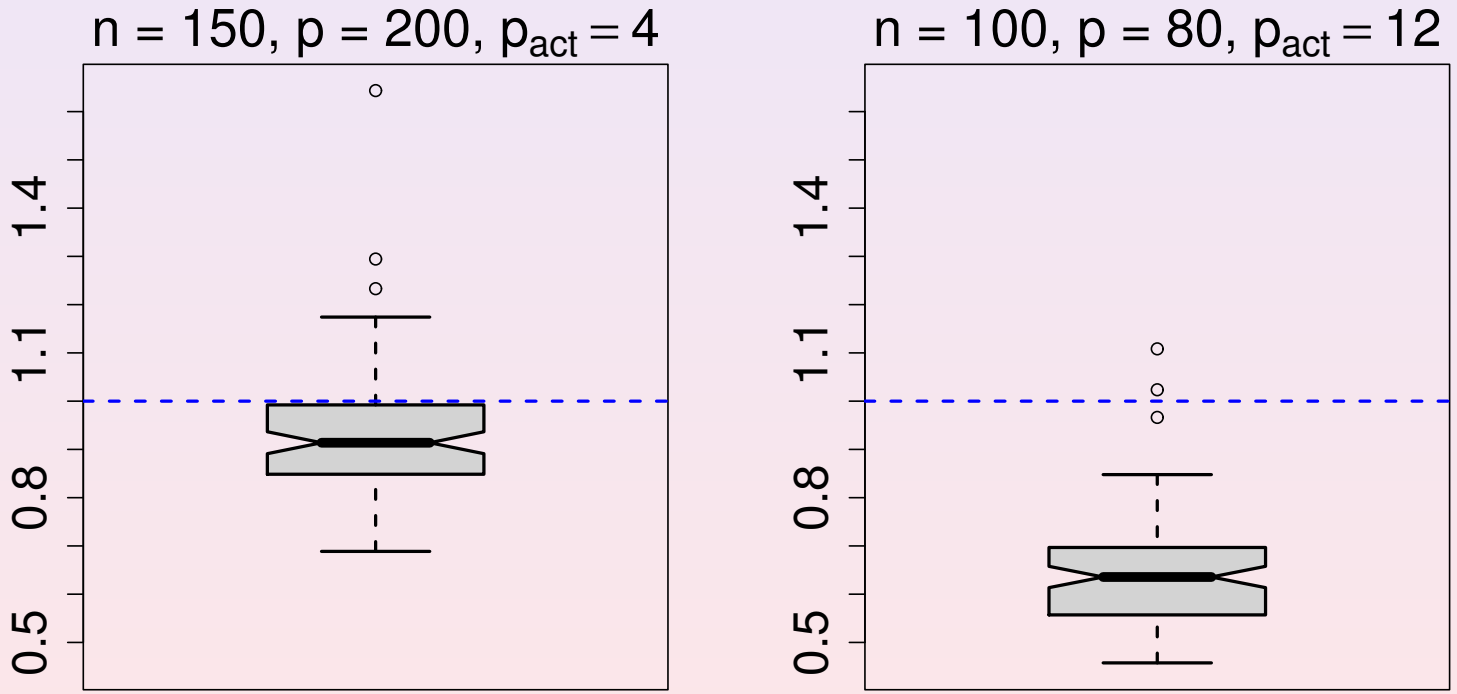
$$\lambda_1 \sum_{j=1}^{p} \underbrace{\|\tilde{\beta}_j\|_2}_{\text{depending on } \lambda_2}$$

i.e., a Group-Lasso penalty

Small simulation study

comparison with L_2Boosting with splines for additive modeling
R-package `mboost` (Hothorn et al.)

ratio of $(\hat{f}(X_{\text{new}}) - f(X_{\text{new}}))^2$: $\frac{\text{spars.-smooth. pen. (SSP)}}{\text{boosting (mboost)}}$



right: true functions have very different degree of complexity

Meatspec: real data-set

`meatspec` data-set, available in `R` package `faraway`

$p = 100,\ n = 215$

highly correlated covariates (channel spectrum measurements)

samples of finely chopped pure meat
Y: fat content
X: 100 channel measurements of absorbances

goal: predict fat content of new samples using 100 absorbances which can be measured more easily

50 random splits in training and test data $\rightsquigarrow (\hat{Y}_{\text{new}} - Y_{\text{new}})^2$:

$$\mathbb{E}[\frac{\text{prediction error SSP}}{\text{prediction error boosting}}] = 0.86$$

i.e. 14% better performance using SSP

Further improvements: Adaptive SSP penalty

straightforward to do and implement... for reducing bias

new penalty:

$$\sqrt{w_{1,j}\|f_j\|_2^2 + \lambda_2 w_{2,j} I^2(f_j)},$$

$$w_{1,j} = 1/\|\hat{f}_{\text{init},j}\|_2,\ w_{2,j} = 1/I(\hat{f}_{\text{init},j})$$

performance ratio: $\mathbb{E}[\frac{\text{squared error adaptive SSP}}{\text{squared error SSP}}]$:

model	performance ratio
$n = 150,\ p = 200,\ p_{\text{eff}} = 4$	0.47
$n = 100,\ p = 80,\ p_{\text{eff}} = 12$	0.77

$\rightsquigarrow$ substantial additional performance gains

effect of adaptivity seems even more pronounced than for linear models

Further improvements: Adaptive SSP penalty

straightforward to do and implement... for reducing bias
new penalty:

$$\sqrt{w_{1,j}\|f_j\|_2^2 + \lambda_2 w_{2,j} I^2(f_j)},$$
$$w_{1,j} = 1/\|\hat{f}_{\text{init},j}\|_2,\ w_{2,j} = 1/I(\hat{f}_{\text{init},j})$$

performance ratio: $\mathbb{E}[\frac{\text{squared error adaptive SSP}}{\text{squared error SSP}}]$:

model	performance ratio
$n = 150,\ p = 200,\ p_{\text{eff}} = 4$	0.47
$n = 100,\ p = 80,\ p_{\text{eff}} = 12$	0.77

$\rightsquigarrow$ substantial additional performance gains
effect of adaptivity seems even more pronounced
than for linear models

The general (new) Group Lasso penalty

what we used (with provable properties) for

- categorical data
- flexible additive modeling
- ... and many more problems

the general Group Lasso penalty:

$$\lambda \sum_{j=1}^{q} \sqrt{\beta_{\mathcal{G}_j}^T \underbrace{A_j}_{\text{pos. definite}} \beta_{\mathcal{G}_j}}$$

A_j may be of the form $A_j = A_j(\lambda_2)$

HIF1α motif additive regression

for finding HIF1α transcription factor binding sites on DNA sequences

$n = 287$, $p = 196$: data from liver cell lines
Y_i: binding intensity of HIF1α to a DNA-region i
(from CHIP-chip experiments)

many candidate motifs from de-novo computational algorithms (MDScan)
e.g. *ACCGTTAC*, *GAGGTTCAG*, ...
$X_i^{(j)}$: score of abundance of candidate motif j in region i

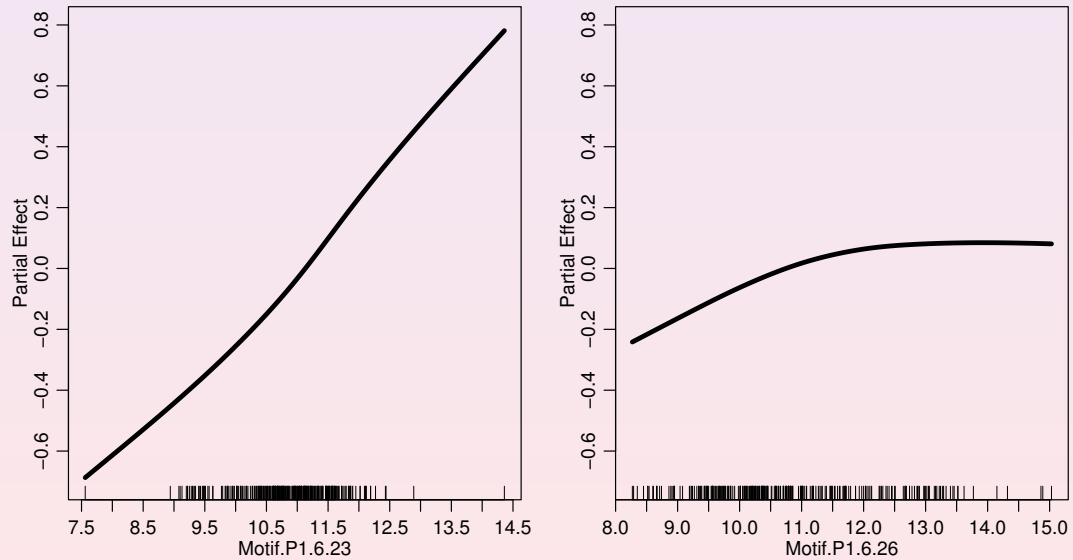
goal: find the relevant variables (the relevant motifs) which explain the binding intensity of HIF1α in an additive model

$$
\begin{aligned}
Y_i &= \text{binding intensity in DNA region } i \\
&= \sum_{j=1}^{p} f_j(\text{abundance of candidate motif } j \text{ in region } i) + \text{error} \\
&\quad (i = 1, \ldots, n)
\end{aligned}
$$

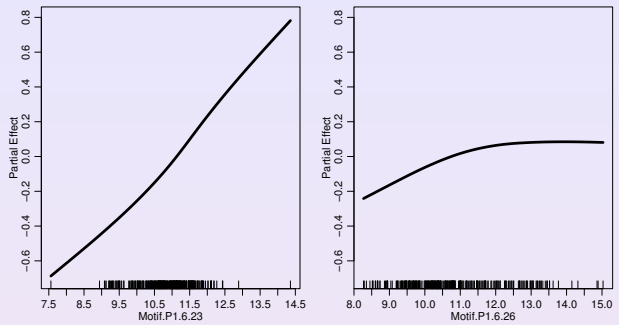
5 fold CV for prediction optimal tuning

additive model with SSP has $\approx$ 20% better prediction performance than linear model with Lasso

SSP: 28 active functions (selected variables)
bootstrap stability analysis: select the variables (functions) which have occurred at least in 50% among all bootstrap runs
$\rightsquigarrow$ only 2 stable variables /candidate motifs remain



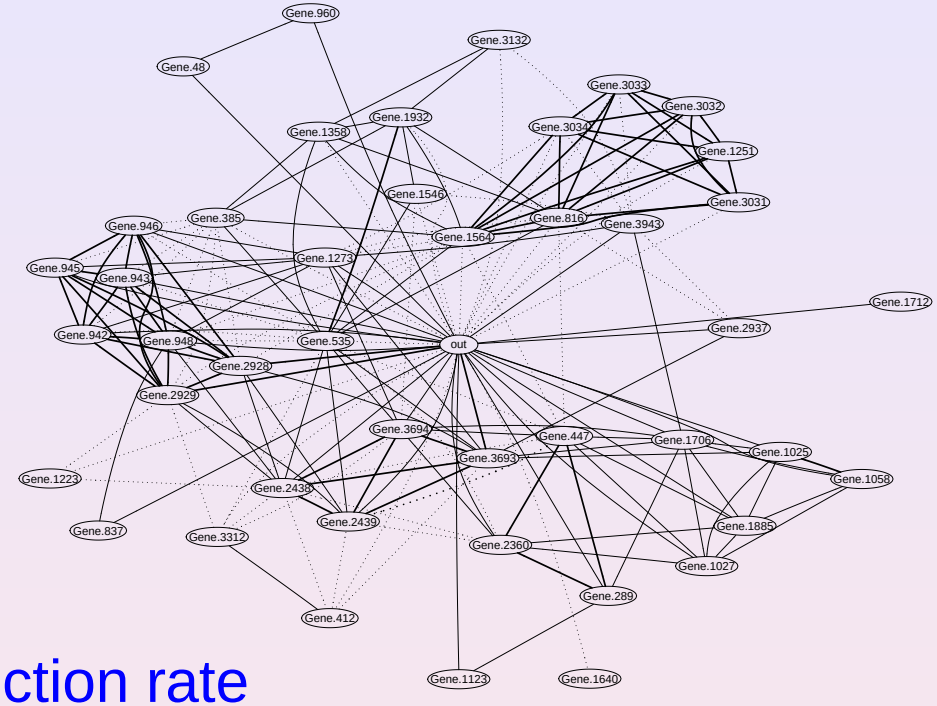
right panel: indication for nonlinearity



right panel: variable corresponds to a true, known motif

variable/motif corresponding to left panel:
good additional support for relevance (nearness to transcriptional start-site of important genes, ...)
ongoing validation with Ricci and Krek labs, ETH Zurich

Riboflavin production with Bacillus Subtilis



Y: riboflavin production rate

covariates $X \in \mathbb{R}^p$: expressions from $p = 4088$ genes

sample size $n = 72$ from a "homogeneous" population of genetically engineered mutants of Bacillus Subtilis

goal: find variables / genes which are relevant for
riboflavin production rate and
which have not been modified so far

5-fold CV for prediction optimal tuning:
additive model (with SSP) and linear model with Lasso have essentially the same prediction error

and estimated additive functions look very linear

but we can tell this only ex post having fitted an additive model with $p = 4088$

SSP for additive model: 44 selected genes
Lasso for linear models: 50 selected genes

overlap: 40 genes selected by both methods/models

one interesting gene “XYZ” which

- is selected by both methods (after bootstrap stability analysis)
- is biologically “plausible”
- has not been modified so far

High-dimensional data analysis and software in R

there are many things you can do...

mathematically well understood methods having “optimality” properties

- ℓ^1-type (Lasso-type) penalization and versions thereof
  `grplasso`: Fitting user specified models with Group Lasso penalty
  `glmnet`: Lasso and elastic-net regularized generalized linear models
  `glasso`: Graphical lasso- estimation of Gaussian graphical models

  `relaxo`: Relaxed Lasso; `lars`: Least Angle Regression, Lasso and Forward Stagewise; `penalized`: L1 (lasso) and L2 (ridge) penalized estimation in GLMs and in the Cox model; `lasso2`: L1 constrained estimation aka ’lasso’; `elasticnet`: Elastic-Net for Sparse Estimation and Sparse PCA
- kernel methods
  (less understood in terms of variable selection)
  `kernlab`: Kernel Methods Lab

mathematically less exploited but also very useful
(in particular for mixed data-types)

- Boosting:
  `gbm`: Generalized Boosted Regression Models
  `mboost`: Model-Based Boosting

  `CoxBoost`: Cox survival models by likelihood based boosting; `GAMBoost`: Generalized additive models by likelihood based boosting
- Random Forest:
  `randomForest`: Breiman and Cutler's random forests for classification and regression

  `randomSurvivalForest`: Ishwaran and Kogalur's Random Survival Forest
- ...

Conclusions

1. ℓ^1-type (Lasso-type) penalty methods
   - are computationally tractactable for $p \gg n$
   - have provable properties with respect to:
     numerical convergence
     and statistical asymptotic "optimality" (or consistency)
   - have contributed to successful modeling in practice
2. the generalized Group-Lasso penalty

$$\lambda \sum_{j=1}^{q} \sqrt{\beta_{\mathcal{G}_j}^T A_j \beta_{\mathcal{G}_j}}$$

   is for a broad range of high-dimensional problems
3. coordinatewise optimization of convex (non-smooth) objective functions is fast and scales nicely in p
4. "new" R-software which is fast for $p \gg n$:
   `grplasso` (Meier) for generalized Group Lasso
   `glmnet` (Hastie, Friedman & Tibshirani) for Lasso
   `penGAM` (Meier, in preparation) for flexible additive modeling