Lasso: Case studies, Bayesian interpretation

Patrick Breheny

February 23

Introduction

- Now that we have covered the basic ideas behind the lasso, we will use it to analyze data from two high-dimensional studies, and in the process:
    - Witness issues that arise in high-dimensional data where $p > n$
    - Deal with some complexities that arise in the analysis of complex real data
    - Become more familiar with the `R` package `glmnet` for fitting lasso models
- Lastly, we will explore the meaning of the lasso penalty as a Bayesian prior

Breast cancer: Study design

- Our first case study consists of breast cancer data from The Cancer Genome Atlas (TCGA) project
- The response variable in our analysis is expression of BRCA1, the first gene identified to increase the risk of early onset breast cancer
- In the dataset, expression measurements of 17,322 additional genes from 536 patients are available (and measured on the log scale)
- Because BRCA1 is likely to interact with many other genes, including tumor suppressors and regulators of the cell division cycle, it is of interest to find genes with expression levels related to that of BRCA1

Analysis

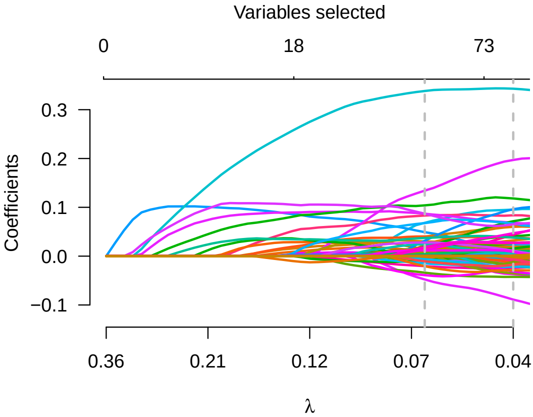
Let's start by fitting, and then plotting, the lasso solution path together with 10-fold cross validation results:

```
cvfit <- cv.glmnet(X, y)
fit <- cvfit$glmnet.fit
```

Note that the complete-data lasso path is included with the output of `cv.glmnet`; it is not necessary to call `glmnet` to obtain it.

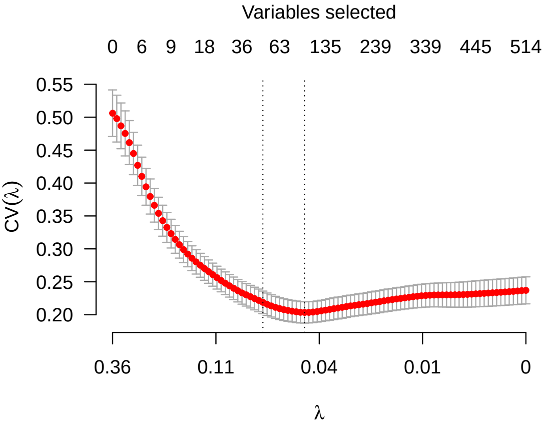
Coefficient path

```
xlim <- log(c(fit$lambda[1], cvfit$lambda.min))
plot(fit, xlim=xlim, xvar="lambda")
```



CV plot

```
plot(cvfit)
```



Remarks

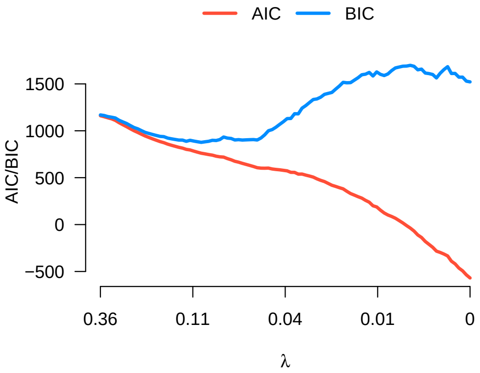
- By the sharp drop in CV error between $\lambda = 0.36$ and $\hat{\lambda} = 0.045$, we can see that the model successfully explains a substantial fraction of the variability in BRCA1 expression
- Specifically, the maximum R^2 of the model is 0.60:

```
max(1-cvfit$cvm/var(y))
[1] 0.6041819
```

- It is also fairly clear that lowering λ past 0.045 results in progressively worse predictions

AIC and BIC

```
fit <- ncvreg(X, y, penalty='lasso')
AIC(fit); BIC(fit)
```



Remarks: AIC

- Unlike the low-dimensional pollution data example, in this high-dimensional problem AIC gives drastically different results from cross-validation
- In particular, AIC offers no protection against overfitting, and is minimized at the (unidentifiable) unpenalized model
- The cross-validation results indicate that this estimate of prediction error is almost certainly wrong

Remarks: BIC

- BIC gives more reasonable results here, suggesting, like cross-validation, a regularization parameter somewhere in the range $0.05 < \lambda < 0.10$
- However, BIC begins to break down as $\lambda \to 0$, with the problem becoming more and more extreme the further we continue along the path
- In general, while BIC can be helpful in selecting λ, it is a good idea to plot it like this rather than blindly selecting the λ value that minimizes BIC

coef

- Like many R modeling functions, glmnet offers coef and predict methods to interact with the fitted model
- For example, from the coefficient path we can see that one gene stands out as being particularly significant; obviously, it would be of interest to know the identity of that gene
- Using glmnet's coef operator, we learn that this gene is named NBR2:

```
> b <- coef(cvfit)
> b[which(b > 0.15),,drop=FALSE]
NBR2 0.3334144
```

Remarks

- NBR2 is adjacent to BRCA1 on chromosome 17, and recent experimental evidence indicates that the two genes share a promoter, so its appearance in the plot makes perfect sense
- It is worth noting that NBR2 was not the first gene to be included in the lasso path – i.e., it was not the gene with the highest marginal association with BRCA1
- This illustrates the power of a regression-based approach over single gene association test to identify the most important biological factors from a large volume of noisy data

More on `coef`

- By default, the `coef` method for `cv.glmnet` returns $\widehat{\beta}(\hat{\lambda}_{1\text{SE}})$ and the `coef` method for `glmnet` returns a matrix of $\widehat{\beta}$ values for the entire grid, but one can obtain $\widehat{\beta}(\lambda)$ for any λ value of interest
- For example, we can see that at $\lambda = 0.2$, there are eight nonzero gene coefficients (plus the intercept):

```
> b <- coef(fit, s=0.2)
> sum(b != 0)
[1] 9
```

predict

- Finally, we illustrate the use of `predict` to obtain predictions of BRCA1 expression levels given expression levels for the other genes with nonzero coefficients in the model
- For example, to obtain the predicted BRCA1 level for subject 85,

```
> predict(cvfit, X[85,,drop=FALSE])
[1,] -0.4495948
```

The range of BRCA1 expression in this study ranged from -3.9 to 0.5, so this actually represents a fairly high expected value

Carbotax study: Introduction

- We now turn our attention to a second case study, involving gene expression changes in ovarian cancer, which brings up some issues we have not encountered previously
- The current standard treatment for ovarian cancer consists of surgery, followed by either carboplatin and paclitaxel or carboplatin alone
- This approach, however, is not effective for all patients
- The goal of this study was to identify genes and pathways associated with drug response

Experimental design

- To identify such genes, the investigators implanted ovarian cell lines into adult mice and allowed the tumors to grow for 2 months, at which point one of three treatments (carboplatin, carboplatin + paclitaxel, or control) was administered to each mouse
- At various time points ranging from 0 to 14 days following the initiation of treatment, the mice were sacrificed, at which point the investigators measured the size of the tumor as well as gene expression in the cancerous tissue

Experimental design (cont'd)

- Our analysis here concentrates on relative tumor volume (RTV) as the outcome variable
- We take a log base 2 transformation of RTV so that $y = 1$ means that the tumor has doubled in size since baseline and, by definition, $y = 0$ for all samples taken at day 0
- For this study, there were 34,694 features with expression data and a sample size of 101 mice

Analysis considerations

- Our goal is to assess the relationship between gene expression and tumor growth
- However, it is important to adjust for treatment group and time of collection in analyzing these data, both of which have significant effects on tumor size
- The lasso model is easily extended to allow for such an analysis

Coefficient-specific λ values

- Up to this point, we have kept λ the same across all variables, but all of our derivations can be easily modified to allow variable j to have its own regularization parameter, λ_j
- In particular, it is trivial to modify the soft-thresholding step of the coordinate descent algorithm so that the update is $S(\tilde{z}_j|\lambda_j)$

glmnet parameterization

- This straightforward extension of the basic lasso model is implemented in the glmnet (and ncvreg) package, albeit with a slight reparameterization
- Th glmnet package allows one to modify the penalty applied to individual covariates through the use of a weighting factor: $\lambda_j = \lambda w_j$, where w_j is the multiplicative factor applied to term j
- The idea here is that w_j scales the baseline regularization factor λ up or down for certain covariates

Remarks

- In general, one could envision carefully choosing a unique w_j for each coefficient based on the likelihood that the feature will play a role in determining the outcome
- For example, we might use a model in which genes that have been implicated in past cancer studies receive less penalization than other genes
- Our goal here is more simple: by assigning $w_j = 0$ for the treatment group and time of collection variables, we can include them in the model as unpenalized covariates
- The rationale for penalizing the gene expression variables is that we expect most genes to have no effect on relative tumor volume, but it does not make sense to extend that assumption to treatment group and time of collection

R code: Setup

- Let's construct a 2 degree of freedom spline to represent the effect of day of collection and allow for an interaction between day of collection and treatment group (here, X is the matrix of gene expression data and Z contains the clinical covariates):

```
y <- log2(sampleData$RTV)
library(splines)
sDay <- ns(Z$Day, df=2)
X0 <- model.matrix(~ Treatment*sDay, Z)[,-1]
w <- rep(0:1, c(ncol(X0), ncol(X)))
XX <- cbind(X0, X)
```

- One advantage of penalized regression is that, by preserving the basic structure of regression, building relatively complex models such as this is as straightforward as it in ordinary linear modeling

R code: Analysis

We can then carry out the analysis in glmnet:

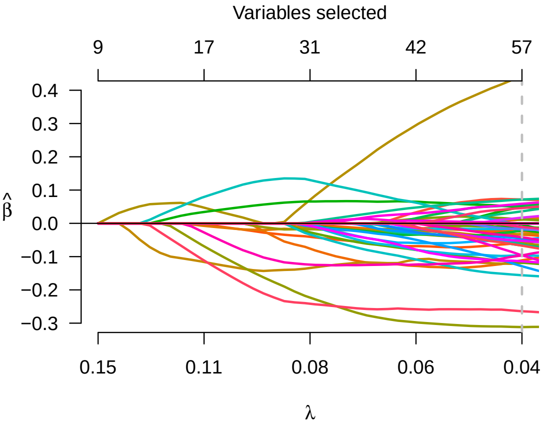
```
cvfit <- cv.glmnet(XX, y, penalty.factor=w)
fit <- cvfit$fit
```

or ncvreg:

```
cvfit <- cv.ncvreg(XX, y, penalty.factor=w, penalty='lasso')
fit <- cvfit$fit
```

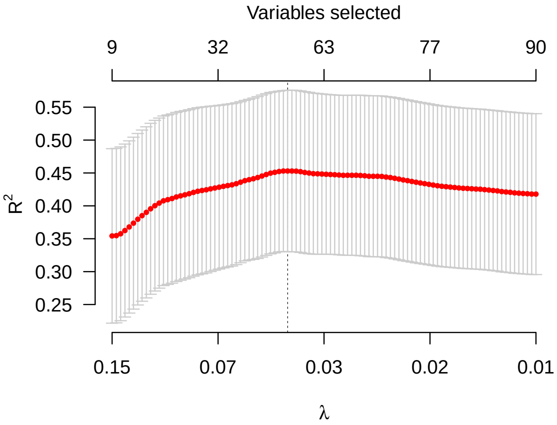
Carbotax: Coefficient path

```
plot(fit)
```



Carbotax: R^2

```
plot(cvfit, type='rsq')  # Only available in ncvreg
```



Remarks

- In this example, R^2 is not zero even at $\lambda_{\max}$; treatment group and day of collection (which, along with their interaction, consists of 8 covariates) alone explain 36% of the variability in RTV
- Nevertheless, the gene expression data provides additional predictive benefit beyond that of treatment group and day of collection: by including the gene expression variables, we can increase the R^2 to 45%

Double-exponential prior

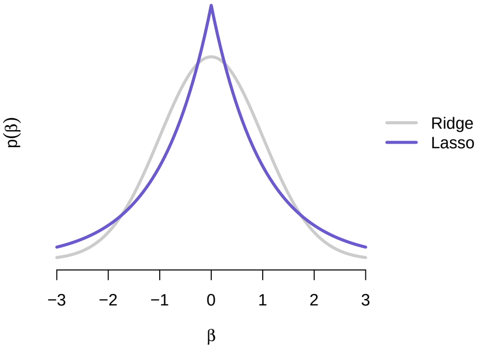
- As with ridge regression, the lasso objective function can be viewed as coming from a Bayesian formulation of the regression model
- Here, the prior on the regression coefficients is a Laplace, or double-exponential, distribution as opposed to a normal distribution:

$$\begin{aligned} p(\boldsymbol{\beta}) &= \prod_{j=1}^{p} \frac{\gamma}{2} \exp(-\gamma|\beta_j|) \\ &= \left(\frac{\gamma}{2}\right)^p \exp(-\gamma\|\boldsymbol{\beta}\|_1), \end{aligned}$$

where $\gamma > 0$

Ridge and lasso priors

- With this prior distribution on $\boldsymbol{\beta}$, we have that the lasso estimate $\widehat{\boldsymbol{\beta}}(\lambda)$ is the posterior mode of $\boldsymbol{\beta}$, where, as in the ridge regression case, $\lambda = \gamma\sigma^2/n$
- Comparison of ridge and lasso priors:



Remarks

- Note that the lasso prior is "pointy" at 0, so there is a chance that the posterior mode will be identically zero
- Note also that the lasso prior has thicker tails than the ridge prior, which explains why lasso solutions exhibit greater separation between small and large coefficients
- Interestingly, since the Laplace distribution can be written as a scale mixture of normals, it is also possible to write the lasso prior as

$$\begin{aligned} \beta_j|\tau_j^2 &\overset{\perp\!\!\!\perp}{\sim} \mathrm{N}(0, \tau_j^2) \\ \tau_j^2 &\overset{\perp\!\!\!\perp}{\sim} \frac{\gamma^2}{2}\exp(-\tfrac{1}{2}\gamma^2\tau_j^2) \end{aligned}$$