Lecture 7: “*What the heck was that?*” thought Caroline as she got into the elevator with the fish in it

Upcoming events

Embedded
Tuesday, 3-4pm 6229 MS

Over the last decade, the Internet has been described as a democratizing force; a global distribution system providing instantaneous access to information and enabling new forms of collaboration. In short, the Internet connects people and a wide range of information sources and services. In this talk, I will consider a new trend in networking in which "embedded" information technologies create connections between people and the physical world. I will focus mainly on distributed networks of sensors and actuators, and various infomechanical devices, embedded in the physical world and operating semi-autonomously.

These systems present statisticians and other "data scientists" with new methodological challenges; from design and deployment, to fault detection and repair, to visualization, data analysis and modeling.

Last time

We attempted to do a final wrap up on the shrinkage material, ending with a grand flourish of practical applications

But alas, fate conspired against us and we didn't quite get there; several threads were left dangling

- Effective degrees of freedom for ridge regression
- Selecting the ridge parameter
- Working out what happens with a different penalty

Today

We will first talk about effective degrees of freedom and make that concept clear with a little simulation

We next take up the issue of selecting the penalty parameter; armed with degrees of freedom, we can try AIC/BIC or we use cross-validation

Finally, we end with the absolute value penalty, working out in a bit more detail what happens in the orthogonal regression case

This brings us to the end of our regression wrap up and plants you firmly in the mid 1990's in terms of linear modeling technology

Applying ridge regression

In practice, we do not want to penalize the intercept in our fit; to do so would make our procedure dependent on the origin chosen for the response

With this in mind, we instead write the penalized OLS criterion in the form

$$\sum_{i=1}^{n}[y_i - \beta_0 - \beta_1 x_{i1} - \ldots - \beta_p x_{ip}]^2 + \sum_{k=1}^{p} \beta_k^2$$

Applying ridge regression

We can show that minimizing this criterion over $\beta_0, \beta_1, \ldots, \beta_p$ is equivalent to a two-step procedure

1. We assume the predictors have been standardized

$$\frac{1}{n}\sum_i x_{ik} = 0 \quad \text{and} \quad \frac{1}{n}\sum_i x_{ik}^2 = 1$$

Applying ridge regression

We want to work with standardized predictors because the ridge solutions are sensitive do differences in scales

That is, we will get a different solution if we multiply one predictor by 10 (say, by choosing to measure one variable in millimeters rather than centimeters)

To avoid these difficulties, we want to fix the “scale” of each of the predictor variables in advance

Applying ridge regression

We can show that minimizing this criterion over $\beta_0, \beta_1, \ldots, \beta_p$ is equivalent to a two-step procedure

2. We define $\hat{\beta}_0 = \overline{y}$, and do not shrink it; we estimate the remaining coefficients $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ to minimize

$$\sum_{i=1}^{n} [y_i - \widehat{\beta}_0 + \beta_1 x_{i1} \cdots - \beta_p x_{ip}]^2 + \sum_{k=1}^{p} \beta_k^2$$

Applying ridge regression

Then, for any fixed value of λ , our ridge estimates are $\hat{\beta}_0 = \overline{y}$ and

$$\widetilde{\boldsymbol{\beta}} = (\boldsymbol{X}^t\boldsymbol{X} + \lambda \boldsymbol{I}_{p\times p})^{-1}\boldsymbol{X}^t\boldsymbol{y}$$

Example: Prostate cancer

Analysis of prostat-specific antigen and clinical measures among men who were about to have their prostates removed

`lcavol`	log(cancer volume)
`lweight`	log(prostate weight)
`age`	age
`lbph`	log(benign prostatic hyperplasia)
`svi`	seminal vesicle invasion
`lcp`	log(capsular penetration
`gleason`	Gleason score
`pgg45`	percent of Gleason scores 4 or 5
`lpsa`	log(prostate-specific antigen)

Example: Prostate cancer

In all, there are 97 cases which we divide randomly into two sets; 67 points will form our *training set* and the remaining 30 will be used as a *test set*

Let $y_1^*, \ldots, y_{30}^*$ denote the test data, and for each y_j^* let $x_{j1}^*, \ldots, x_{jp}^*$ denote the vector of predictors

After fitting a set of models using the training data, we can then assess how well the models perform using predictions on the test set

$$\frac{1}{30}\sum_{j=1}^{30}[y_j^* - \widehat{\beta}_0 - \widehat{\beta}_1 x_{j1}^* - \cdots - \widehat{\beta}_p x_{jp}^*]^2$$

Test set errors

With a test set, we are estimating the *generalization error* of our model; how well will it perform on new data?

Cross-validation is another way to approach this; when data are plentiful there is no reason not to hold out a test set

The test set error can be broken down in to three terms: One related to the squared bias of our estimate, one to the variance of our estimate, and one to the error variance

The algebra to derive this looks a lot like what we did in the third lecture when we considered MSE

Example: OLS full model

Fitting a simple linear model with all the predictors yields the following summary

```
> fit = lm(lpsa~.,data=prostate)
> summary(fit)

Call:
lm(formula = lpsa ~ ., data=prostate)

Residuals:
     Min       1Q   Median       3Q      Max
-1.64870 -0.34147 -0.05424  0.44941  1.48675

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 2.45235    0.08702  28.182  < 2e-16 ***
lcavol       0.71641    0.13350   5.366 1.47e-06 ***
lweight      0.29264    0.10638   2.751  0.00792 **
age         -0.14255    0.10212  -1.396  0.16806
lbph         0.21201    0.10312   2.056  0.04431 *
svi          0.30962    0.12539   2.469  0.01651 *
lcp         -0.28901    0.15480  -1.867  0.06697 .
gleason     -0.02091    0.14258  -0.147  0.88389
pgg45        0.27735    0.15959   1.738  0.08755 .
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

Residual standard error: 0.7123 on 58 degrees of freedom
Multiple R-Squared: 0.6944,     Adjusted R-squared: 0.6522
F-statistic: 16.47 on 8 and 58 DF,  p-value: 2.042e-12
```

Example: Subset selection

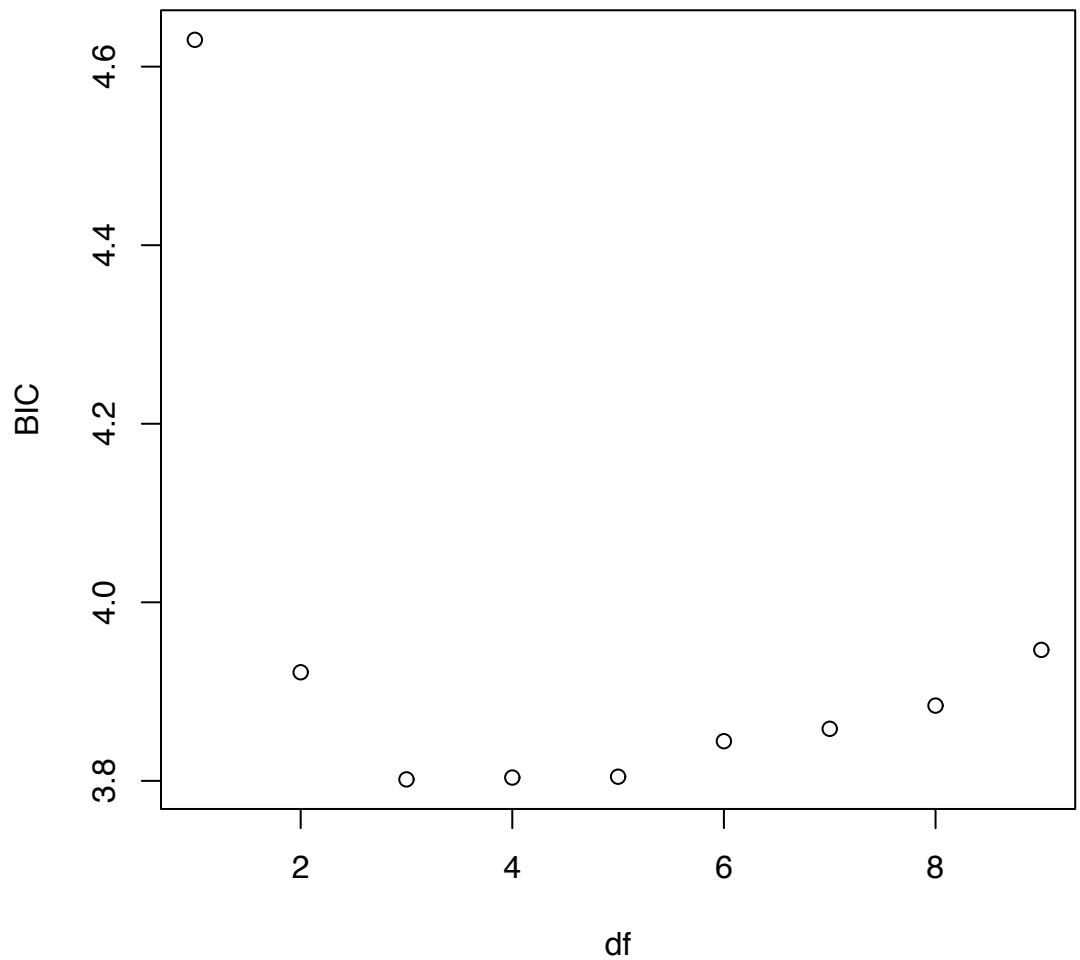
In R, we can perform a variety of subset selection techniques; here is the summary output from a simple forward search

```
> subs = regsubsets(lpsa~.,data=prostate,method="forward")
> summary(subs)

Subset selection object
Call: regsubsets.formula(lpsa ~ ., data = prostate, method = "forward")

8 Variables  (and intercept)
        Forced in Forced out
lcavol      FALSE      FALSE
lweight     FALSE      FALSE
age         FALSE      FALSE
lbph        FALSE      FALSE
svi         FALSE      FALSE
lcp         FALSE      FALSE
gleason     FALSE      FALSE
pgg45       FALSE      FALSE

1 subsets of each size up to 8

Selection Algorithm: forward
         lcavol lweight age lbph svi lcp gleason pgg45
1  ( 1 ) "*"    " "     " " " "  " " " " " "     " "
2  ( 1 ) "*"    "*"     " " " "  " " " " " "     " "
3  ( 1 ) "*"    "*"     " " " "  "*" " " " "     " "
4  ( 1 ) "*"    "*"     " " "*"  "*" " " " "     " "
5  ( 1 ) "*"    "*"     " " "*"  "*" " " " "     "*"
6  ( 1 ) "*"    "*"     " " "*"  "*" "*" " "     "*"
7  ( 1 ) "*"    "*"     "*" "*"  "*" "*" " "     "*"
8  ( 1 ) "*"    "*"     "*" "*"  "*" "*" "*"     "*"
```

Example: Subset selection

We can then compare the different fits with BIC or some other rule

```
> bic = log(subs$rss)+log(67)*(1:9)/67
> plot(bic)
```



Example: Subset selection

Choosing the model with just two predictors, we get the following summary

```
> fit = lm(lpsa~lcavol+lweight,data=pro)
> summary(fit)

Call:
lm(formula = lpsa ~ lcavol + lweight, data = pro)

Residuals:
     Min       1Q   Median       3Q      Max
-1.58852 -0.44174  0.01304  0.52613  1.93127

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.45235    0.09301  26.368  < 2e-16 ***
lcavol       0.77986    0.09824   7.938 4.14e-11 ***
lweight      0.35191    0.09824   3.582 0.000658 ***
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

Residual standard error: 0.7613 on 64 degrees of freedom
Multiple R-Squared: 0.6148,     Adjusted R-squared: 0.6027
F-statistic: 51.06 on 2 and 64 DF,  p-value: 5.54e-14
```

Example: Ridge regression

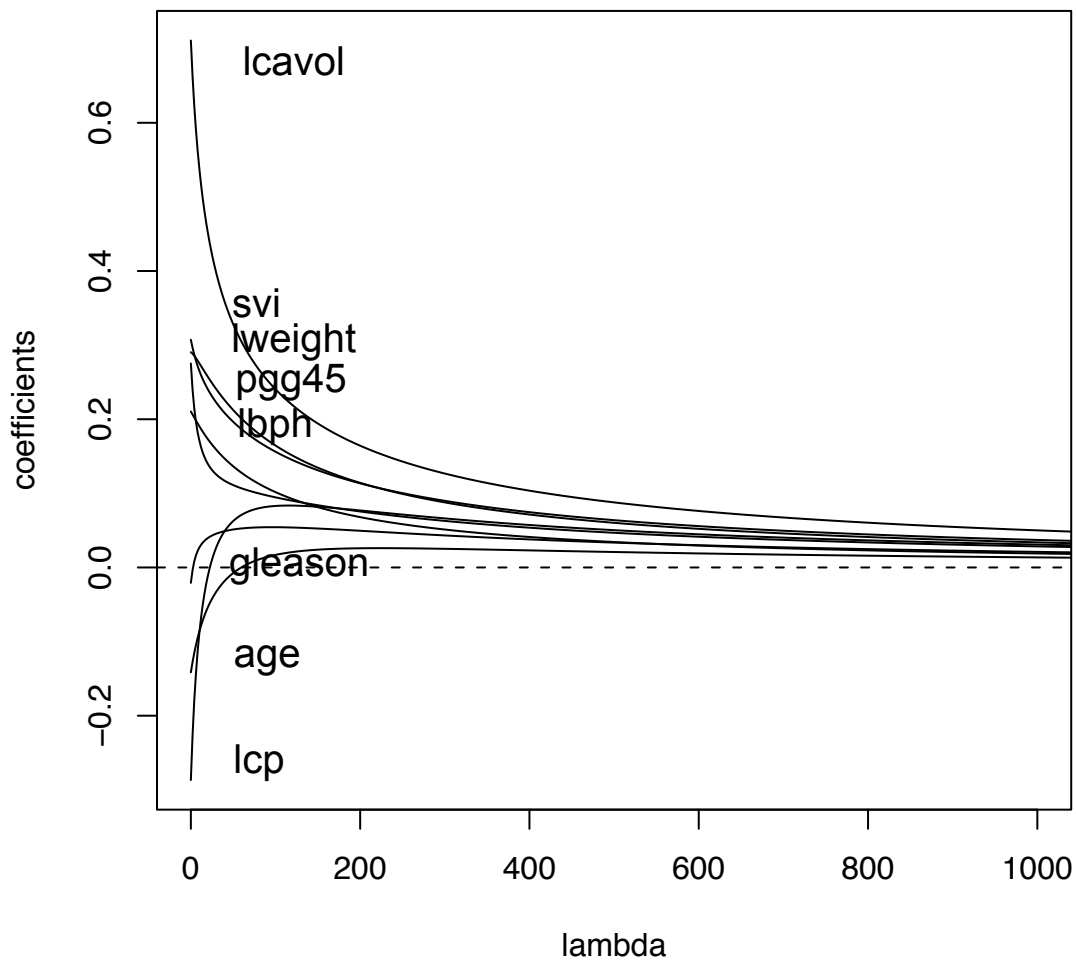
R can fit a ridge regression for a large number of choices of λ in one go

```
> lam = seq(0,10000,len=5000)
> ridgefits = lm.ridge(lpsa~.,data=prostate,lam=lam)
```

Example: Ridge regression

The code below generates the ridge traces for each predictor

```
> plot(range(lam), range(ridgefits$coef),type="n")
> for(i in 1:nrow(ridgefits$coef) lines(lam,ridgefits$coef[i,])
```

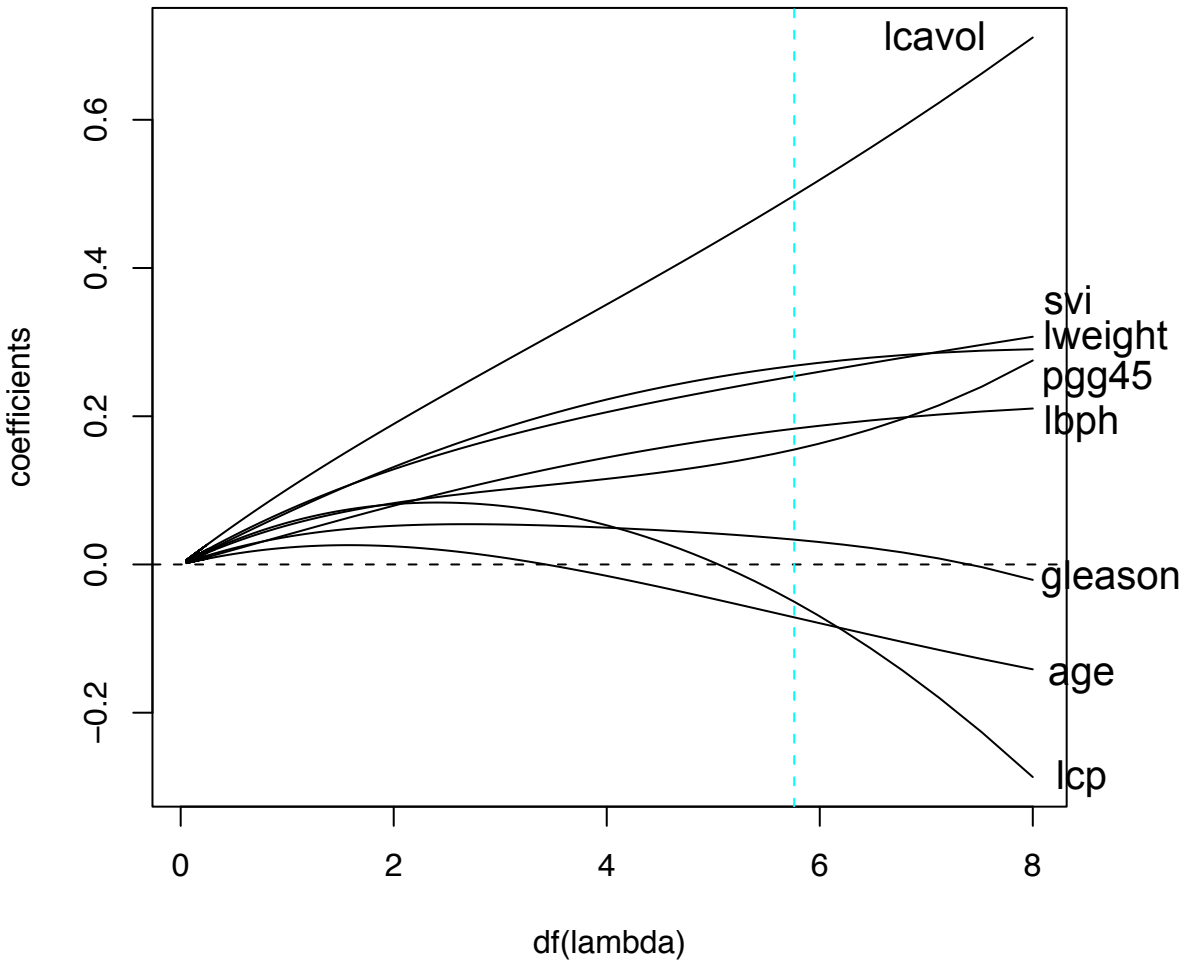


Example

The ridge parameter is not really interpretable between models

It is not on a natural scale and is not the most useful for understanding the amount of shrinkage taking place

Instead, the *effective degrees of freedom* allows us to interpret the impact of the penalty



Effective degrees of freedom

As we increase λ, our ridge estimates feel the pinch of the constraint

To measure flexibility in our fit, we define the effective degrees of freedom to be

$$\mathsf{df}(\lambda) = \mathsf{trace}[\boldsymbol{X}(\boldsymbol{X}^t\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^t]$$

Notice that this definition has the right properties; $\mathsf{df}(0) = p$ and as $\lambda \to \infty$, $\mathsf{df}(\lambda) \to 0$

Effective degrees of freedom

Where do we actually see the effects of the constraint? Consider the following simulation setup for a fixed λ

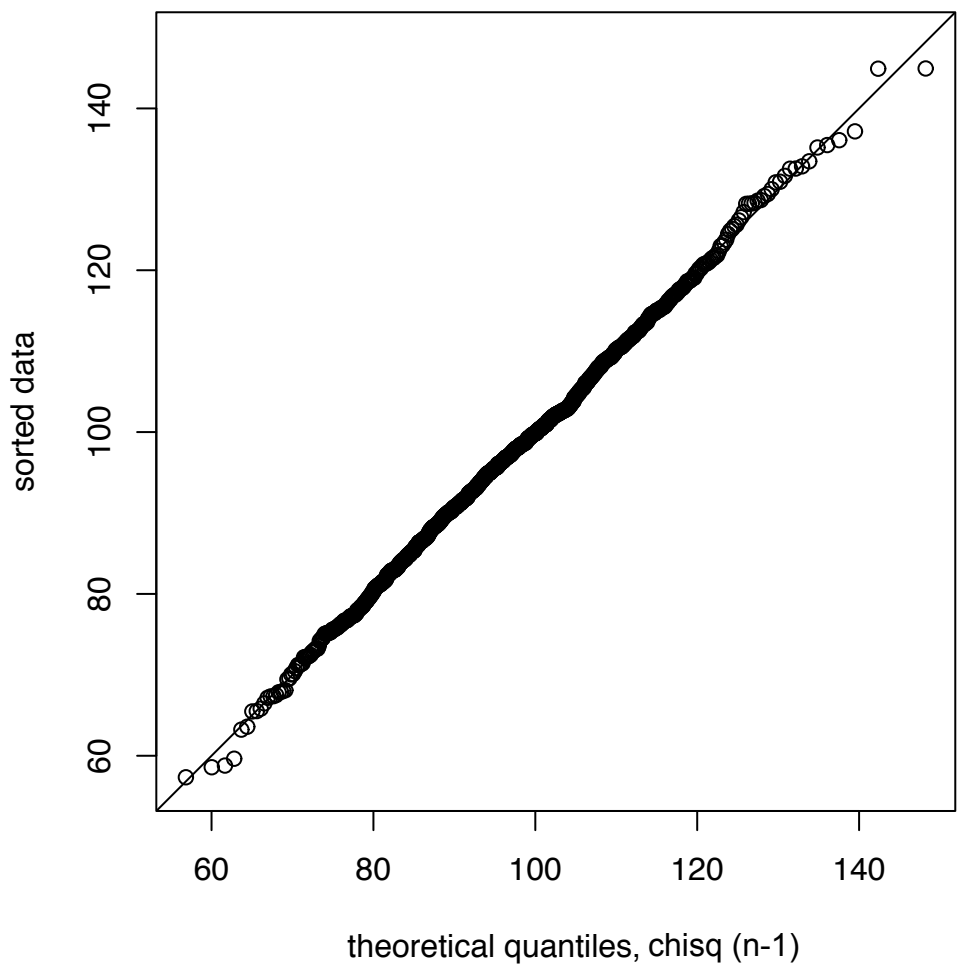
1. For each observation in our training sample, create a new observation $\widetilde{y}_i = \epsilon_i$ where the ϵ_i are independent, normal random variables with mean 0 and variance 0.5

2. Fit a ridge regression using the given value of λ and compute RSS/σ^2

3. Repeat steps 1. and 2. a total of 5,000 times, generating 5000 *RSS* numbers

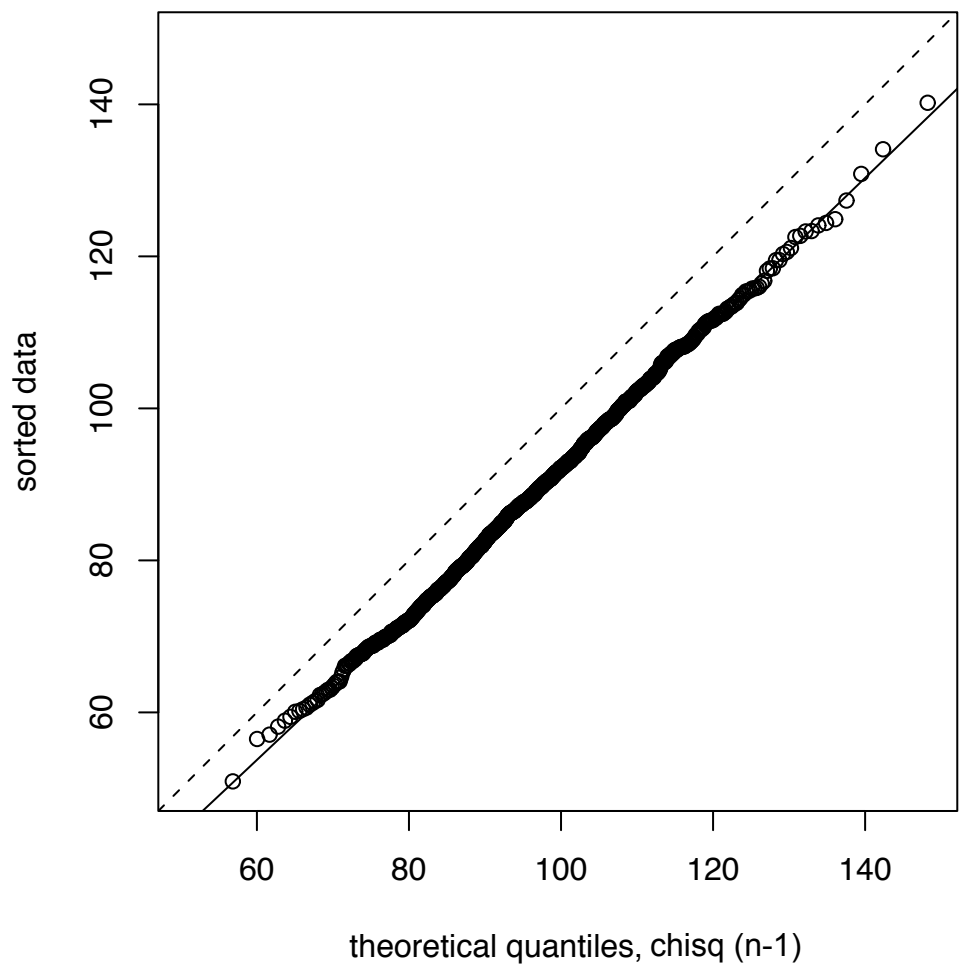
Effective degrees of freedom

Here we set $\lambda = 10,000$ and we have plotted our sorted data against the quantiles for a chi-square distribution with *n-1* degrees of freedom; the 1-1 line is included



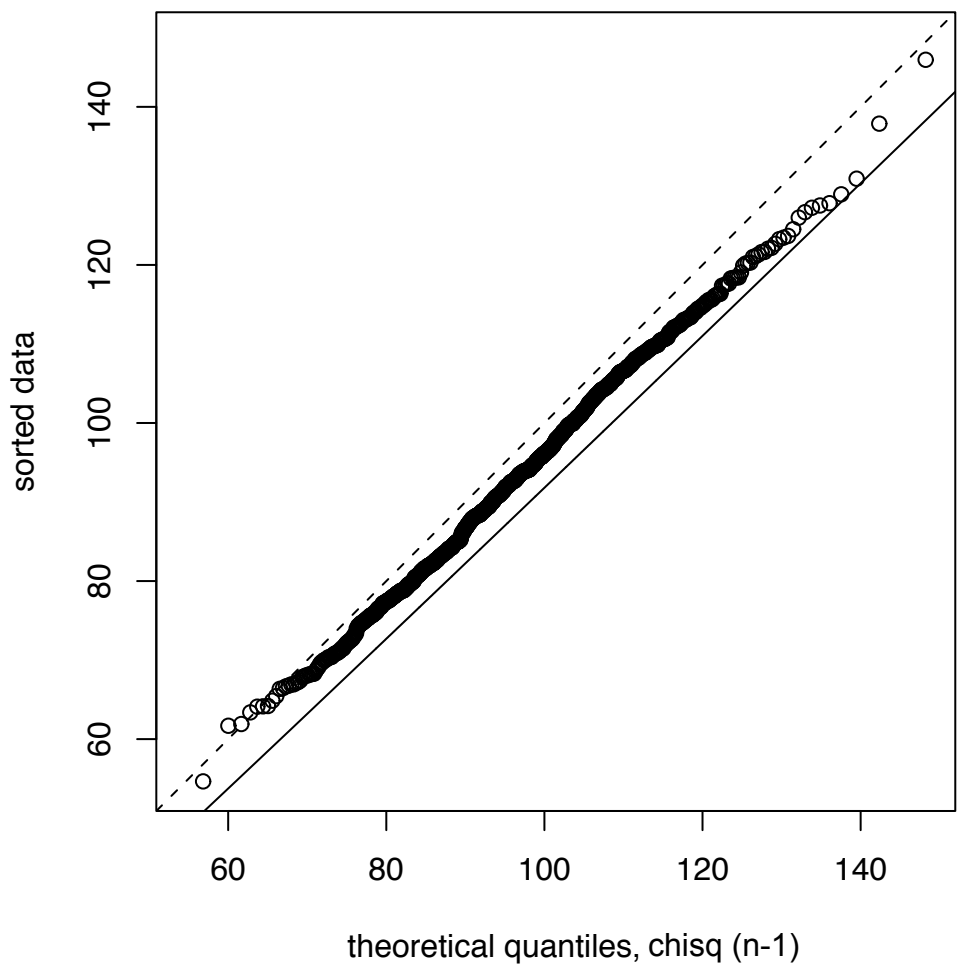
Effective degrees of freedom

Same plot, except that now we have taken $\lambda = 0$; the solid line corresponds to quantiles for a chi-square distribution with *n-9* degrees of freedom



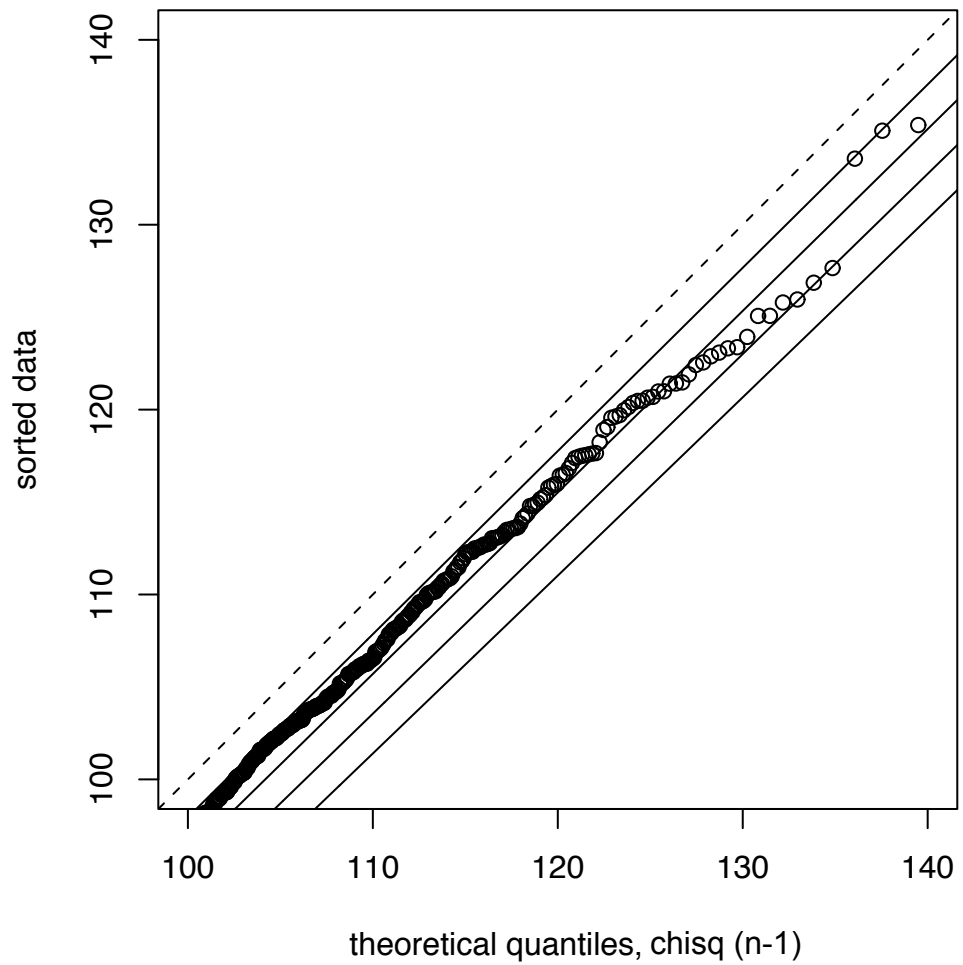
Effective degrees of freedom

Finally, we set $\lambda = 300$; the solid line corresponds to quantiles for a chi-square distribution with *n-9* degrees of freedom, the dashed *n-1* degrees of freedom



Effective degrees of freedom

Here is a close-up of the previous plot ($\lambda = 300$); the lines now correspond to quantiles of a chi-square distribution with *n-9, n-7, n-5, n-3* and *n-1* (moving from lower right to upper left)



Chi-square distribution

While this distribution was probably presented to you in terms of sums of squared, independent normals, the degrees of freedom can in fact be continuous

Here is the density

$$f(x) = \frac{e^{-x/2}\ x^{\nu/2-1}}{2^{\nu/2}\ \Gamma(\nu/2)}$$

where we recall the definition of the Gamma function

$$\Gamma(a) = \int_0^\infty t^{a-1} e^{-t} dt$$

Putting it to use

Once we have something like an effective degrees of freedom, we can use it in selection criterion like AIC/BIC

For example, we recall that BIC is given by

$$\mathsf{BIC}(\lambda) = RSS(\lambda) + \mathsf{df}(\lambda)\sigma^2 \log n$$

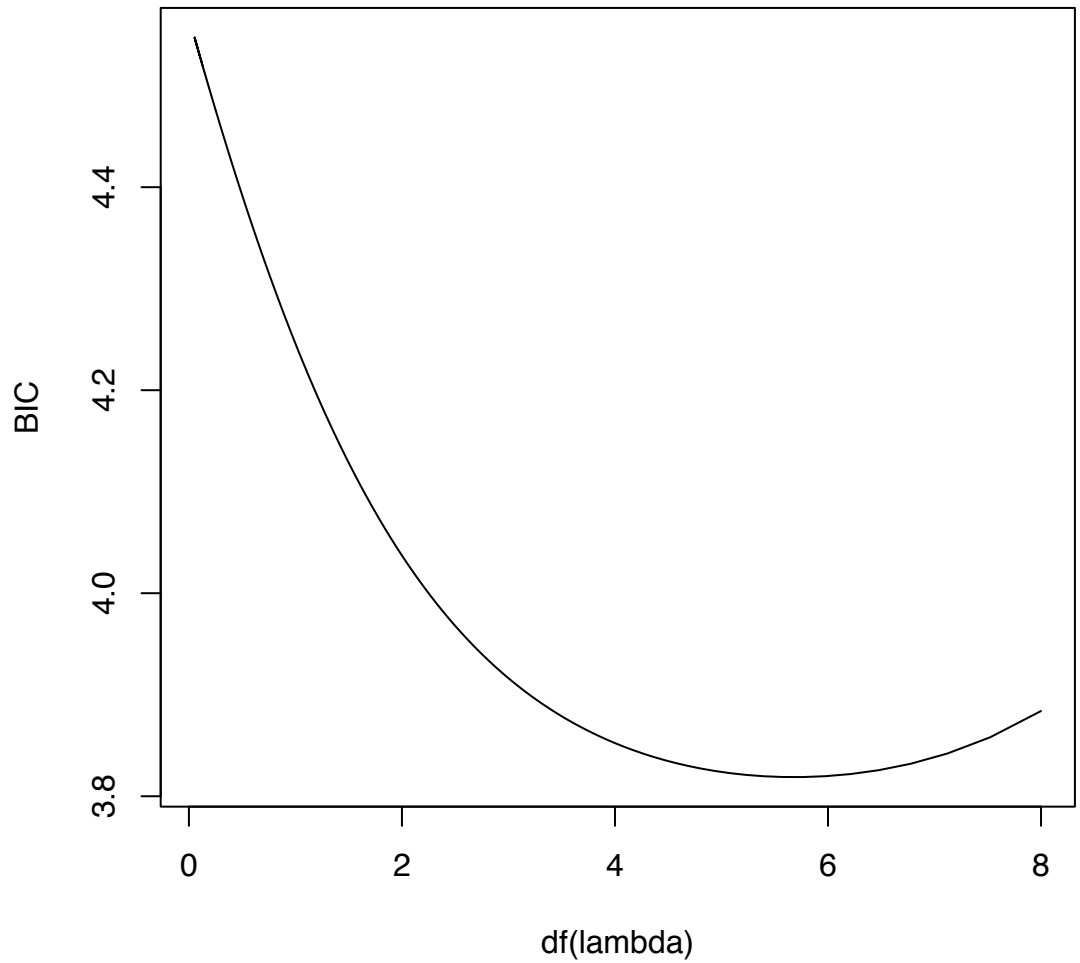
when σ^2 is known (or can be independently estimated) and by

$$\mathsf{BIC}(\lambda) = \log RSS(\lambda) + \frac{\mathsf{df}(\lambda)}{n} \log n$$

when it is unknown

Putting it to use

For the ridge regression on the prostate data set, we obtain the following BIC curve (error variance unkown)

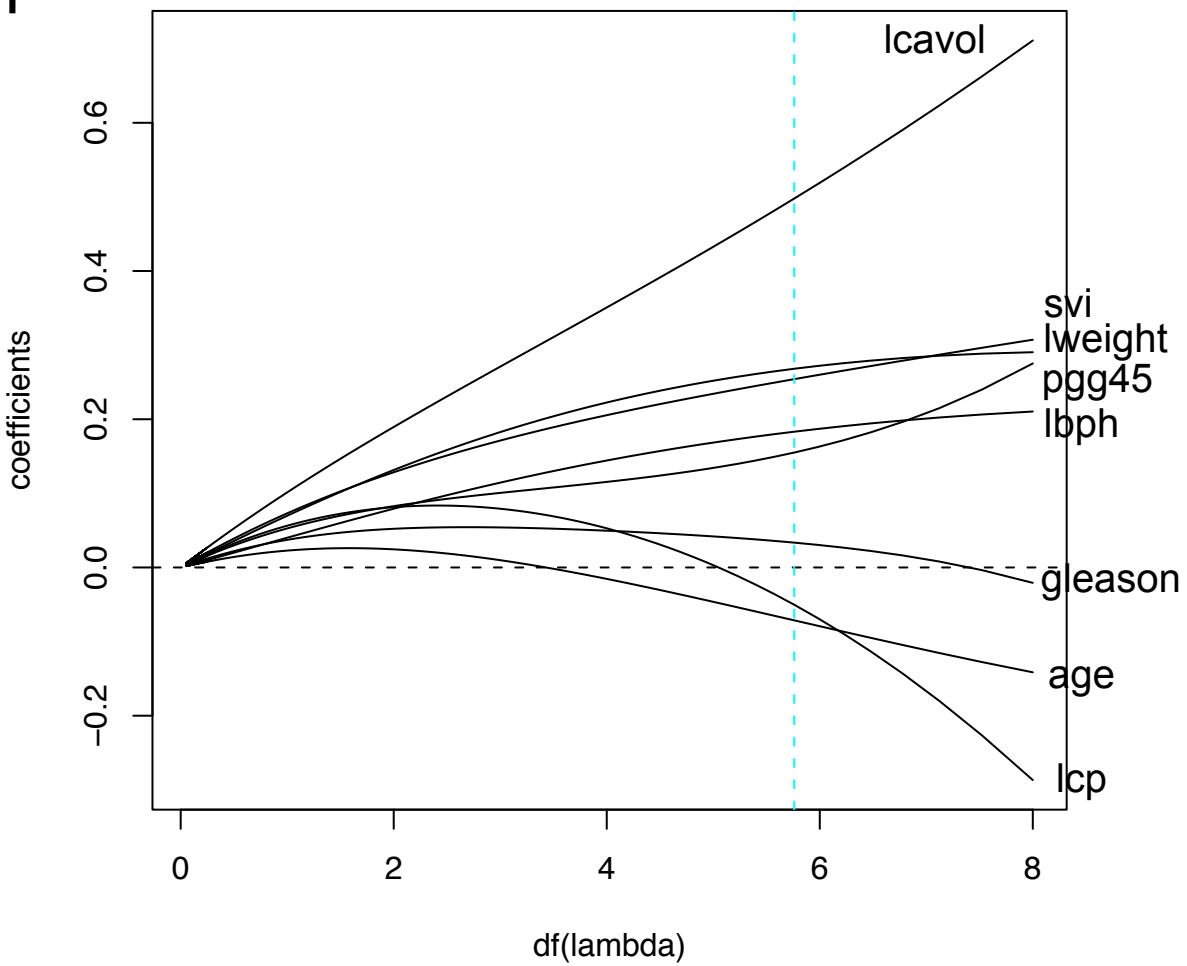


Example: Ridge regression

This now gives us an easy way to select the ridge parameter

Using the BIC choice, we select the coefficient values that intersect the dashed vertical line

It is also common to use cross-validation to choose the ridge parameter



A bit more general

Predictions from a ridge regression can be written in the following form

$$\widehat{\boldsymbol{y}} = \boldsymbol{X}(\boldsymbol{X}^t\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}^t\boldsymbol{y}$$

In general, there are many kinds of estimates that can be expressed as simple linear combinations of the response; we can write them abstractly as

$$\widehat{\boldsymbol{y}} = \boldsymbol{S}(\lambda)\boldsymbol{y}$$

Setting $\mathsf{df}(\lambda) = \mathsf{trace}[\boldsymbol{S}(\lambda)]$ is a common approximation for the degrees of freedom in all these models

Test set errors

We now compare our test set errors using the three different estimation procedures

$$\frac{1}{30}\sum_{j=1}^{30}[y_j^* - \widehat{\beta}_0 - \widehat{\beta}_1 x_{j1}^* - \cdots - \widehat{\beta}_p x_{jp}^*]^2$$

OLS, full model	0.586
OLS, subset of two predictors	0.574
Ridge regression	0.540

Test set errors

OLS with the full model will have a larger error the data set contains unnecessary predictors (added variance)

Subset selection can do better, providing we can correctly identify the unnecessary predictors; if not, the error will be larger (since it is contributing to the bias)

Ridge can do better than OLS on the full model if we don’t over-shrink (which would add to the bias)

Ridge regression

By way of motivation, minimizing the penalized OLS criterion

$$\sum_{i=1}^{n}[y_i - \beta_1 x_{i1} - \cdots - \beta_p x_{ip}]^2 + \lambda \sum_{k=1}^{p} \beta_k^2$$

is equivalent to minimizing the ordinary OLS criterion

$$\sum_{i=1}^{n}[y_i - \beta_1 x_{i1} - \cdots - \beta_p x_{ip}]^2$$

subject to the constraint that

$$\sum_{k=1}^{p} \beta_k^2 \leq s$$

for some value of *s*

Alternative penalties

The most recent innovations in regression technology involve a different penalty

$$\sum_{i=1}^{n}[y_i - \beta_1 x_{i1} - \cdots - \beta_p x_{ip}]^2 + \lambda \sum_{k=1}^{p} |\beta_k|$$

which is equivalent to minimizing the ordinary OLS criterion

$$\sum_{i=1}^{n}[y_i - \beta_1 x_{i1} - \cdots - \beta_p x_{ip}]^2$$

subject to the constraint that

$$\sum_{k=1}^{p} |\beta_k| < s$$

for some value of *s*

The lasso

Unlike the ridge regression penalty, the lasso solutions have the property that the coefficients can shrink to zero for finite values of the penalty parameter

This means that the lasso is somewhere between subset selection and ridge regression

The lasso

Whereas we found it much easier to work with λ for ridge regression, for the lasso, it is more natural to work directly with *s*

In fact, most implementations take s to be some multiple of the sum of the absolute values of the OLS coefficients

$$s = b \sum_{k=1}^{p} |\widehat{\beta}_k|$$

For *b*=1, we have our OLS solution, for *b*=0 we have the intercept-only model

The lasso

In R, we can use the following

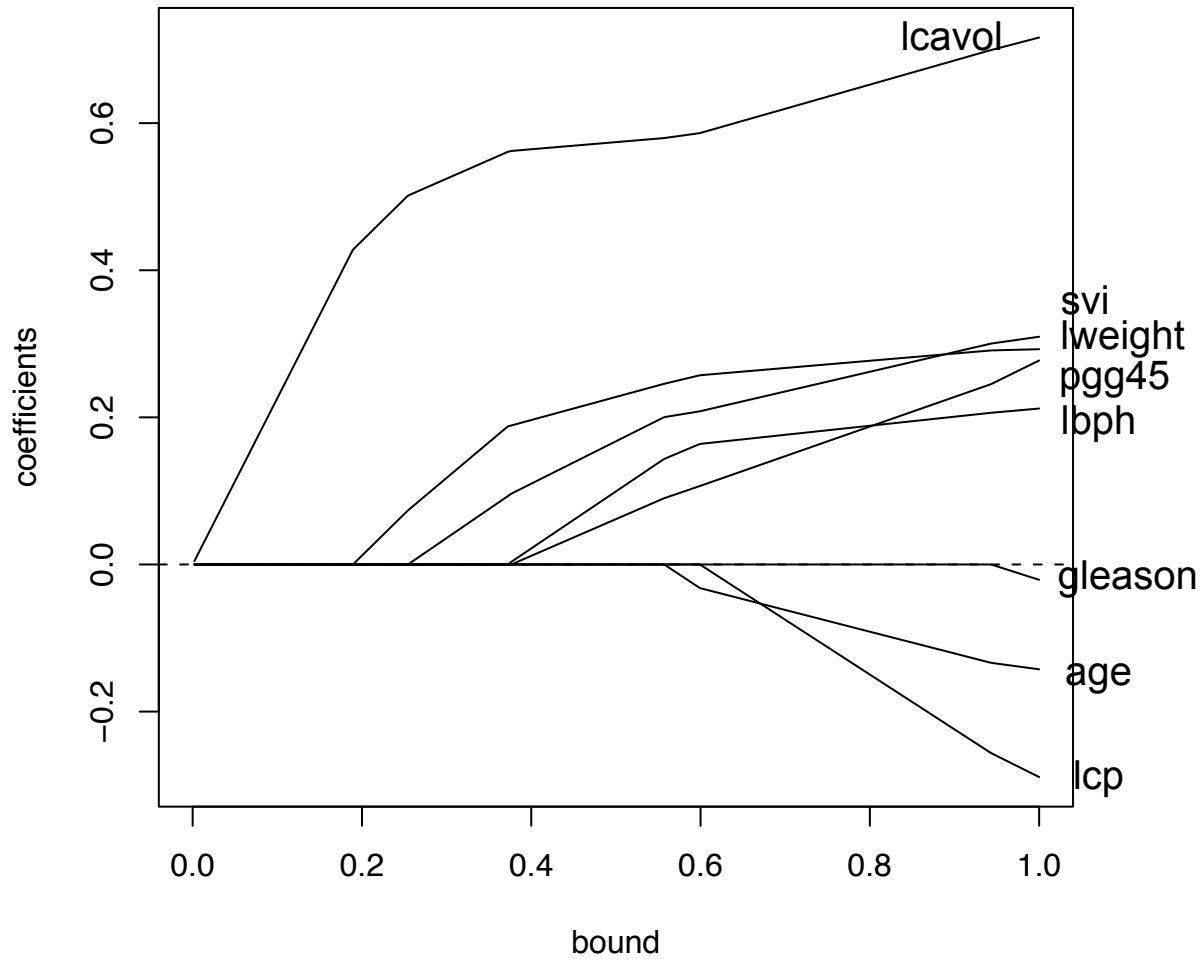
```
> lfit = l1ce(lpsa~.,data=prostate,bound=(1:500)/500)
> summary(lfit[[10]])

Call:
 l1ce(formula = lpsa ~ ., data = prostate, bound = (1:500)/500)

Residuals:
     Min       1Q   Median       3Q      Max
-2.81424 -0.78144  0.08825  0.86691  2.96808

Coefficients:
            Value   Std. Error Z score Pr(>|Z|)
(Intercept)  2.4523  0.1519    16.1460  0.0000
lcavol       0.0452  0.2074     0.2179  0.8275
lweight      0.0000  0.1798     0.0000  1.0000
age          0.0000  0.1768     0.0000  1.0000
lbph         0.0000  0.1768     0.0000  1.0000
svi          0.0000  0.2133     0.0000  1.0000
lcp          0.0000  0.2663     0.0000  1.0000
gleason      0.0000  0.2488     0.0000  1.0000
pgg45        0.0000  0.2751     0.0000  1.0000

Residual standard error: 1.243 on 58.96 degrees of freedom
The relative L1 bound was      : 0.02
The absolute L1 bound was      : 0.04520983
The Lagrangian for the bound is:  55.46005
```

Example: The lasso

To compare a number of different bounds, we can create a plot that is similar to the one we generated for ridge regression

Instead of degrees of freedom, we plot the coefficients against the bound to indicate the amount of smoothing taking place

We can again apply model selection criterion to select the bound



Lasso for orthonormal problems

When our predictor variables are orthonormal, then we can rewrite the penalized criterion as

$$\sum y_i^2 + \beta_1^2 + \cdots + \beta_p^2$$
$$-2\beta_1 \sum y_i x_{1i} - \cdots - 2\beta_p \sum y_i x_{pi}$$
$$+\lambda|\beta_1| + \cdots + \lambda|\beta_p|$$

Lasso for orthonormal problems

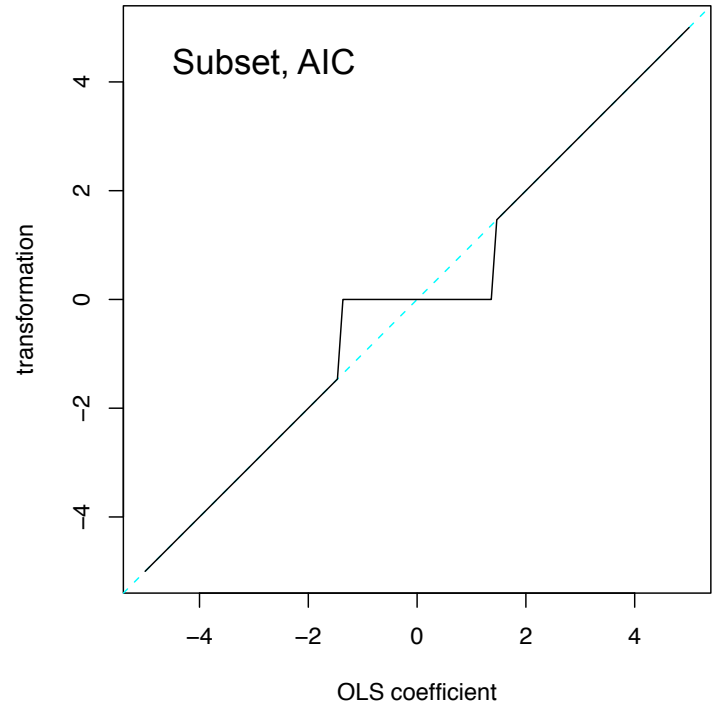
By working with this expression, we find that the lasso takes the OLS estimate and applies "soft-thresholding"

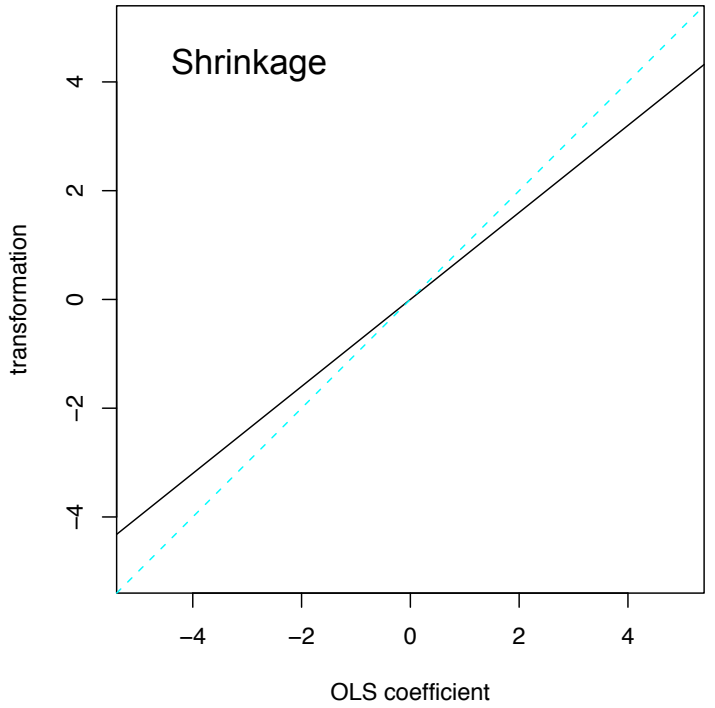
$$\widetilde{\beta_k} = \mathsf{sign}(\widehat{\beta}_k)(|\widehat{\beta}_k| - \lambda/2)^+$$

A short comparison

So far, we have seen the following strategies

Subset selection	$\widetilde{\beta_k} = \widehat{\beta}_k$ if $\|\widehat{\beta}_k\| > \sigma\sqrt{\lambda}$
Shrinkage	$\widetilde{\beta_k} = \dfrac{1}{1+\lambda}\ \widehat{\beta}_k$
Lasso	$\widetilde{\beta_k} = \mathsf{sign}(\widehat{\beta}_k)(\|\widehat{\beta}_k\| - \lambda/2)^+$

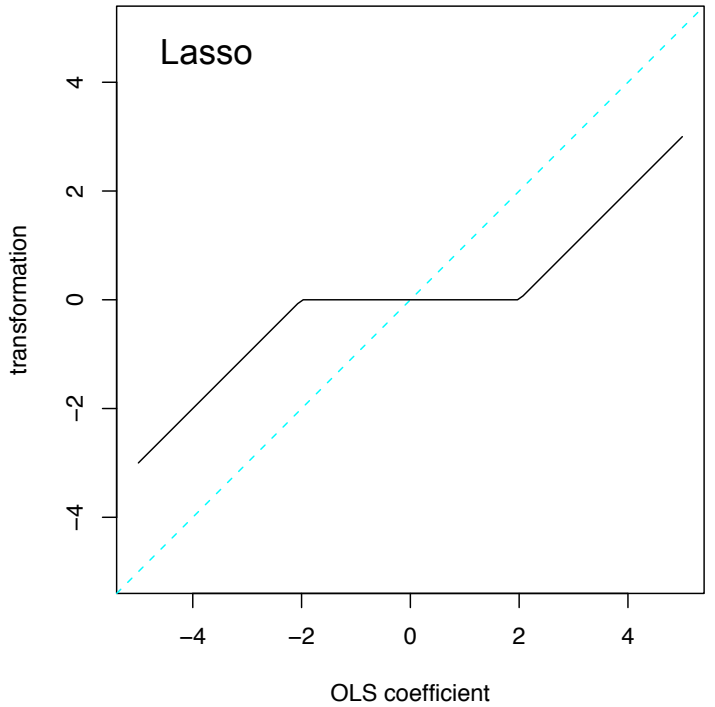




A short comparison

These graphs should make the formulas clearer

By considering these approaches in the context of orthogonal regression, we gain some insights into what each is doing



Next time

We start on the case when our response is either binary or a binomial count

We will start by exploring what breaks down (or doesn't) if we use OLS and try to see how people commonly fix the problems

This will lead us briefly to discrimination and then ultimately to logistic regression