Model Selection

Richard L. Sweeney

based on slides by Chris Conlon

Empirical Methods
Spring 2019

Overview

How many components should we include in our model?

- Too few: under-fitting and large residuals.
- Too many: over-fitting and poor out of sample prediction.

How do we choose?

- X variables.
- Instrumental Variables.

In these notes we'll talk about these in the context of linear regression. (ESL Ch 3)

When do we have too much data?

- On the internet!
- Hedonics: What really determines the price of your house?
- Prediction: What really determines loan defaults?
- Consideration Sets: How many products do consumers really choose among on the shelf?
- What elements of financial filings really matter?

What we teach undergrads

Traditional ways to select the number of components (p) in a model:

$$\begin{aligned}
\overline{R}^2 &= 1 - SSR(p)/TSS - SSR(p)/TSS \cdot \frac{p}{N-p-1} \\
AIC(p) &= \ln\left(\frac{SSR(p)}{N}\right) + (p+1)\frac{2}{N} \\
BIC(p) &= \ln\left(\frac{SSR(p)}{N}\right) + (p+1)\frac{\ln N}{N}
\end{aligned}$$

- Commonly employed in macroeconometric or time-series context for things like selecting lags of an autoregression

$$y_t = \alpha_0 + \sum_{k=1}^{p} \alpha_k y_{t-k} + \varepsilon_t$$

- These are designed for strictly nested models.

These aren't all that helpful in applied micro

- Rely on LRT and a bunch of assumptions (see backup). No clear "best" method.
- Only make sense for "nested" models
  - Should you include x_1 and x_2, or just x_1?
  - Not should you include x_2 *or* x_3?
- Some tests (Cox, Vuong) can handle non-nested case, but rarely used

Back to the real world...

- We have some theoretical benchmark which lets us discern which of two model we prefer (under certain assumptions).
- In practice we often start with a functional form like: $y_i = \beta_0 + \sum_{k=1}^{p} \beta_k x_{i,k} + \varepsilon_i$
- Which x's do we include?
- Which x's do we leave out?
- It is not clear that BIC/AIC or Vuong test tells us what we should do in practice.
- Helpful to review what we're really getting out of multivariate regression. [Much of this section taken from ESL]

Univariate Regression

- Consider a univarite model (with no intercept)

$$Y = X'\beta + \epsilon$$

- Let $\langle x, y \rangle = \sum_i^N x_i y_i$ be the inner product between these two variables in our sample.
- Then the least squares estimate and residuals are

$$\hat{\beta} = \frac{\langle x, y \rangle}{\langle x, x \rangle}$$

$$r = y - x\hat{\beta}$$

What about the case of multiple regression?

- Suppose we had inputs $x1, x2, ..., x_p$
- If they are all **orthogonal**, ie $\langle x_j, x_k \rangle = 0$, we can show that estimates can be recovered using the exact same univariate procedure.
- Since they're orthogonal, including them has no impact.
- Outside of controlled experiments, this is not going to be applicable.

OLS with correlated inputs

- Imagine if we include an intercept along with our x of interest.
- Then the least squares estimate becomes

$$\hat{\beta} = \frac{\langle x - \bar{x}, y \rangle}{\langle x - \bar{x}, x - \bar{x} \rangle}$$

- Note that $x - \bar{x}$ are just the residuals from a projection x on to our $x_0 = 1$

Getting back to orthogonality

- This suggests a simple procdure
  1. Orthogonalize x_1 by projecting it onto x_0
  2. Regress y onto this residual z_1 to get $\hat{\beta}_1$
- This generalizes to the case of p inputs, where for each x_p you project onto the residuals from the previous $p-1$ x's.
- Unless we are running regressions by hand this doesn't seem tremendously helpful.
- However, in practice this is often what your software does!

Gram-Schmidt/QR Decomposition

1. Let $x_0 = z_0 = 1$
2. For $j = 1, 2, \ldots p$: Regress x_j on $z_0, z_1, \ldots, z_{j-1}$ to give you $\hat{\gamma_{jl}} = \langle z_l, x_j \rangle / \langle z_l, x_l \rangle$ and residual $z_j = x_j - \sum_{k=0}^{j-1} \hat{\gamma_{kj}} z_k$.
3. With your transformed orthogonal basis $\mathbf{z}$ you can now regress y on z_p one by one to obtain $\hat{\beta}_p$.

Gram-Schmidt/QR Decomposition II

What does this do?

- The resulting vector $\hat{\beta}$ has been adjusted to deliver the marginal contribution of x_j on y after adjusting for all x_{-j}.
- If x_j is highly correlated with other x_k's then the residual z_j will be close to zero and the coefficient will be unstable.
- This will be true for any variables x_l within a set of correlated variables.
- We can delete any one of them to "resolve" this issue.
- However we should pause to appreciate that our estimate, and its stability, is mechanically linked to the other x_s we do or don't include.

QR Decomposition (Technical Details)

QR Decomposition has a matrix form which regression software uses:

$$\begin{aligned}
\mathbf{X} &= \mathbf{Z\Gamma} \\
&= \underbrace{ZD^{-1}}_{\mathbf{Q}}\underbrace{D\Gamma}_{\mathbf{R}} \\
\hat{\beta} &= \mathbf{R^{-1}Q'y} \\
\hat{\mathbf{y}} &= \mathbf{QQ'y}
\end{aligned}$$

- Z is the matrix of the orthogonalized residuals z_j's.
- Γ is upper triangular matrix with entries $\hat{\gamma}_{kj}$
- D is diagonal matrix with entries $||z_j||$.
- Q is $N \times ((p+1)$ orthogonal matrix $Q'Q = I$
- R is $(p+1) \times (p+1)$ upper triangular matrix.

What happens in practice?

What are people likely doing in practice:

- Start with a single x variable and then slowly add more until additional x's were insignificant
- Start with all possible x variables and drop those where t-statistics were insignificant.
- These procedures actually make some sense if the columns of X are linearly independent or orthogonal.
- In practice our regressors are often correlated (sometimes highly so).

Subset selection

- Least squares have some undesirable properties
  - Poor prediction: They are low bias but high variance. By tempering some of the large coefficients, might improve prediction
  - Interpretation: Sometimes we just want to know the main factors explaining variation in the data.
- A natural alternative is to restrict models to subsets of the full x's.
- If you have K potential regressors you could consider all 2^K possible regressions.
- Or you could could consider all $\binom{K}{P}$ possible combinations with p parameters.

Minimizing SSR: all possible regressions

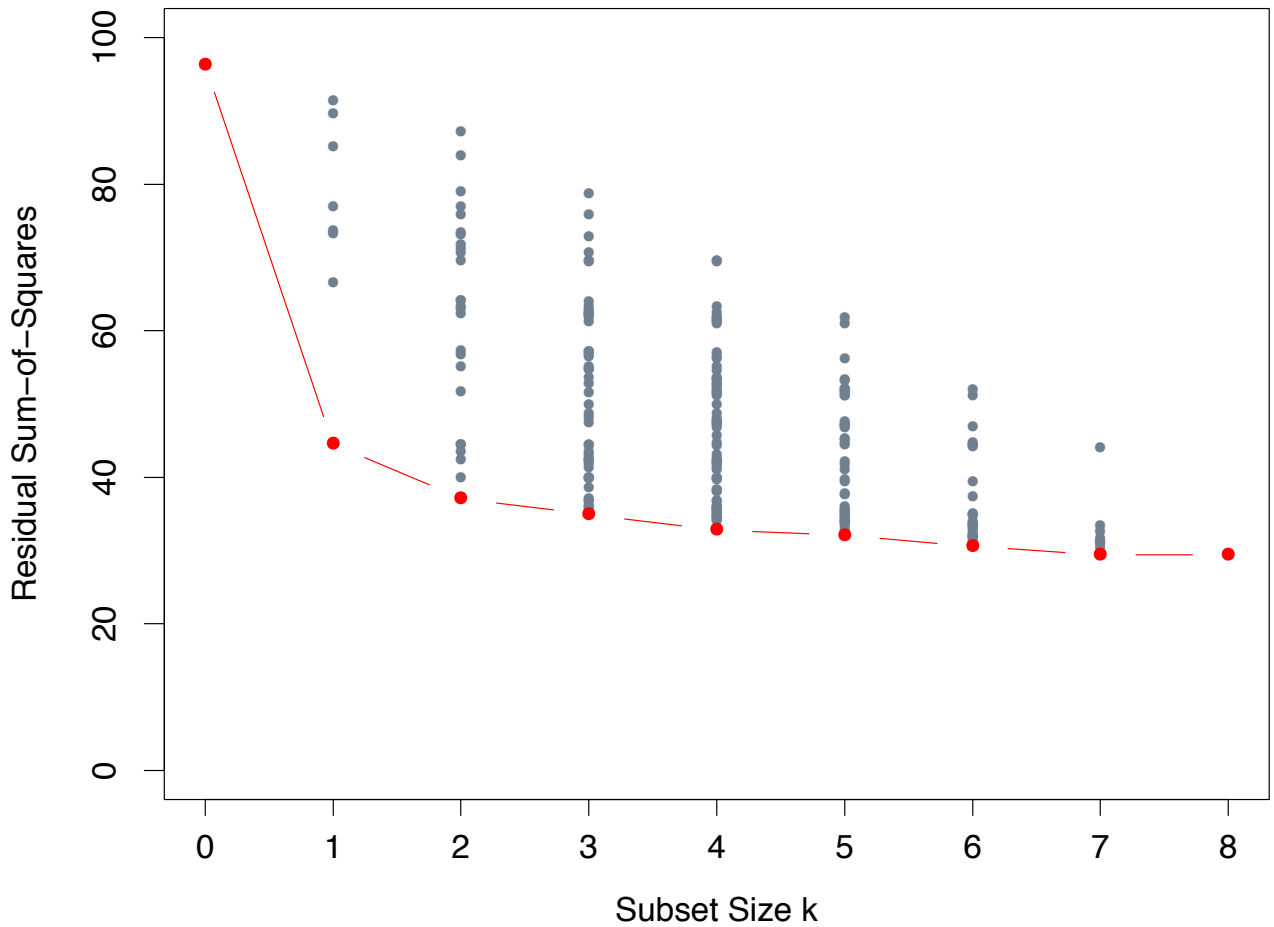


FIGURE 3.5. *All possible subset models for the prostate cancer example. At each subset size is shown the residual sum-of-squares for each model of that size.*

Forward Stepwise Regression

Consider the following greedy algorithm

1. Start with an empty model and add a constant $\overline{y}$.
2. Then run K single-variable regressions, choose the x_k with the highest t-statistic call this $x^{(1)}$.
3. Now run $K-1$ two variable regressions where the constant and $x^{(1)}$ and choose $x^{(2)}$ as regression where x_k has the highest t-statistic.
4. Now run $K-2$ three variable regressions where the constant and $x^{(1)}, x^{(2)}$
5. You get the idea!

We stop when the x_k with the highest t-statistic is below some threshold (often 20% significance).

Backwards Stepwise Regression

1. Start with an full model.
2. Remove the x variable with the lowest t-statistic. Call this $x^{(k)}$.
3. Re-run the regression without $x^{(k)}$.
4. Repeat until the smallest t-statistic exceeds some threshold.

Comparison

- Backwards and fowards stepwise regression tend to give similar choices (but not always).
- Everything is trivial if X's columns are orthogonal (computer has some tricks otherwise- QR).
- Forward stepwise works when we have more regressors than observations $K > N$.
- I proposed the t-stat here but some packages use AIC/BIC as the criteria.
- We should also be careful to group dummy variables together as a single regressor.
- These are implemented in `step` in R and `stepwise` in Stata.
- We probably want to adjust our standard errors for the fact that we have run many regressions in sequence before arriving at our model. In practice not enough people do this!

(Incremental) Forward Stagewise Regression

As an alternative consider:

1. Start with $r = y$ and $(\beta_1, \ldots, \beta_p) = 0$.
2. Find the predictor x_j most correlated with r.
3. Update $\beta_j \leftarrow \beta_j + \delta_j$ where $\delta_j = \epsilon \cdot sgn\langle r, x_j \rangle$.
4. Update $r \leftarrow r - \delta_j \cdot x_j$ and repeat for S steps.

- Alternative $\delta_j = \langle r, x_j \rangle$
- We can continue until no regressors have correlation with residuals
- This is very slow (it takes many many S).
- Sometimes slowness can be good – in high dimensions to avoid overfitting.

Stepwise selection proedures

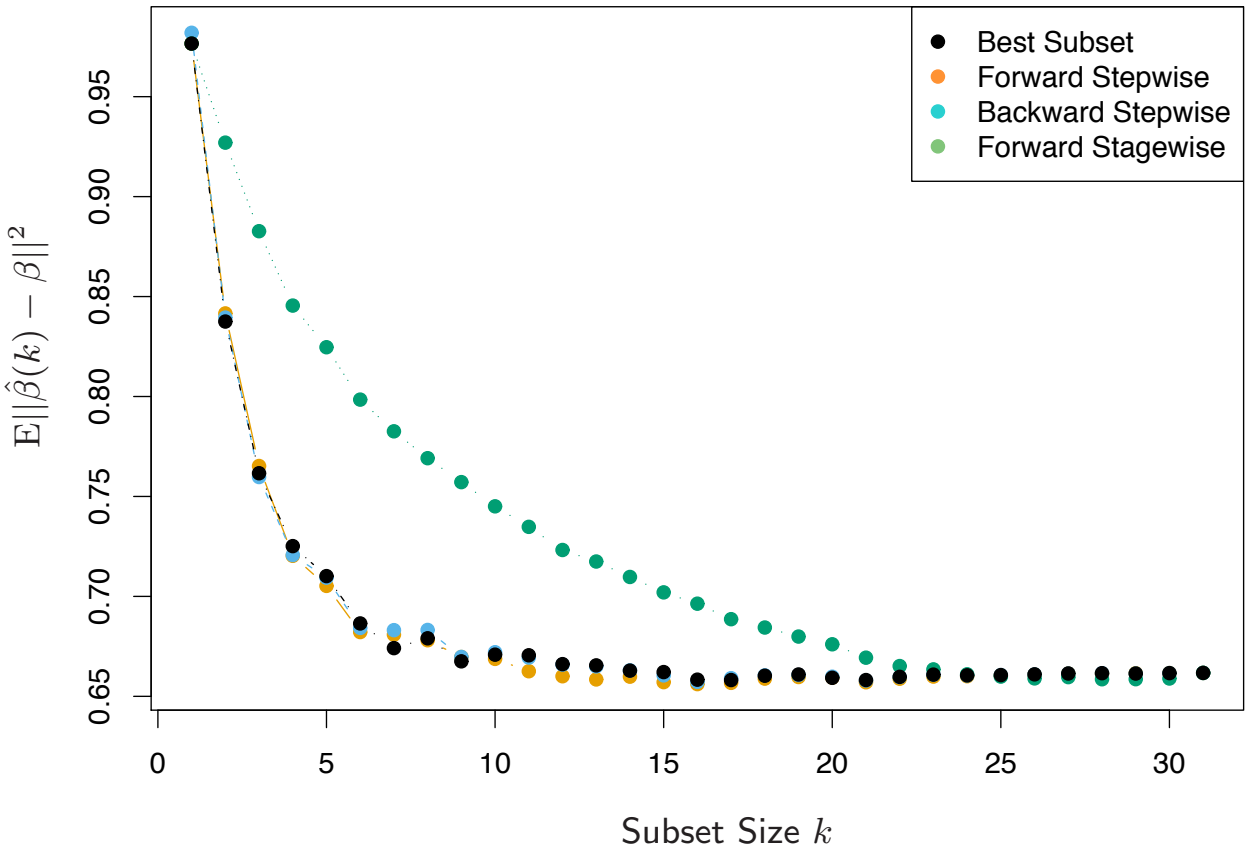


FIGURE 3.6. *Comparison of four subset-selection techniques on a simulated linear regression problem* $Y = X^T\beta + \varepsilon$. *There are* $N = 300$ *observations on* $p = 31$ *standard Gaussian variables, with pairwise correlations all equal to* 0.85. *For* 10 *of the variables, the coefficients are drawn at random from a* $N(0, 0.4)$ *distribution; the rest are zero. The noise* $\varepsilon \sim N(0, 6.25)$, *resulting in a signal-to-noise ratio of* 0.64. *Results are averaged over* 50 *simulations. Shown is the mean-squared error of the estimated coefficient* $\hat{\beta}(k)$ *at each step from the true* β.

Penalized Regression

Suppose we fit a regression model and penalized extra variables all in one go, what would that look like?

$$\hat{\beta} = \arg\min_{\beta} \left[\frac{1}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|^q \right]$$

- We call λ the regularization parameter.
- The penalty term $\lambda \sum_{j=1}^{p} |\beta_j|^q$ as penalizes models where β gets further away from zero.
- Similar to placing a prior distribution on β_j centered at 0.
- **Important:** We definitely want to standardize our inputs before using penalized regression methods.

What Penalty Function?

- Usually you fix q and then look at how estimates respond to γ.
- There are two famous cases $q = 1$ (Lasso) and $q = 2$ (Ridge) though in practice there are many possibilities.
- We can choose λ in a way that minimizes expected prediction error (EPE).
- Recall $EPE(\lambda) = E_x E_{y|x}([Y - g(X, \lambda)]^2 | X)$.
- In practice most people look at out of sample prediction error rate on a cross validated sample.

Ridge Regression

Another popular alternative is the $q = 2$ case

$$\hat{\beta}^{Ridge} = \arg\min_{\beta} \left[\frac{1}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{K} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{K} |\beta_j|^2 \right]$$

- Penalty is L_2 norm on β.
- Can re-write as a constriant $\sum_{j=1}^{K} |\beta_j|^2 \leq s$
- $\hat{\beta}^{Ridge} = (X'X + \lambda I)^{-1} X'Y$.
- If X is orthonormal then $\hat{\beta}_j^{Ridge} = \hat{\beta}_j / (1 + \lambda)$
- In words: everything gets dampened by a constant factor λ (we don't get zeros).
- Adding a constant to the diagonal of $(X'X)$ ensures that the matrix will be invertible even when we have multicollinearity.

What is Ridge Doing?

Just like we can diagonalize (some) square matrices, we can take the singular value decomposition (SVD) of any matrix $\mathbf{X}$ that is $N \times p$

$$\mathbf{X} = \mathbf{UDV}'$$

- $\mathbf{U}, \mathbf{V}$ are $N \times p$ and $p \times p$ orthonormal matrices ($\mathbf{U}$ spans the column space, and $\mathbf{V}$ spans the row space of X.)
- $\mathbf{D}$ is a diagonal matrix $p \times p$ with elements corresponding to the singular values of $\mathbf{X}$.
- If $\mathbf{X}$ is a square, diagonalizable matrix the singular values are equal to the eigenvalues.

OLS vs Ridge

Can write the least squares solution as

$$\mathbf{X}\hat{\beta}^{ols} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} = UU'\mathbf{y}.$$

- $\mathbf{U}'\mathbf{y}$ maps $\mathbf{y}$ into the orthonormal basis $\mathbf{U}$.
- This looks a lot like $\mathbf{QR}$ except that we have chosen a different basis.

OLS vs Ridge

Can write the least squares solution as

$$\mathbf{X}\hat{\beta}^{ols} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} = UU'\mathbf{y}.$$

- $\mathbf{U}'\mathbf{y}$ maps $\mathbf{y}$ into the orthonormal basis $\mathbf{U}$.
- This looks a lot like $\mathbf{QR}$ except that we have chosen a different basis.

And the Ridge solution is:

$$\begin{aligned}\mathbf{X}\hat{\beta}^{ridge} = \mathbf{X}(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}'\mathbf{y} &= \mathbf{U}\mathbf{D}(\mathbf{D}^2 + \lambda\mathbf{I})^{-1}\mathbf{D}\mathbf{U}'\mathbf{y} \\ &= \sum_{j=1}^{p} \mathbf{u}_j \frac{d_j^2}{d_j^2 + \lambda}\mathbf{u}_j'\mathbf{y}\end{aligned}$$

- Same change of basis. Now we shrink each component by $d_j^2/(d_j^2 + \lambda)$.

What is Ridge Doing? II

- Ridge is shrinking the OLS cordinates of y by $d_j^2/(d_j^2 + \lambda)$
- This will have the biggest impact on small d_j. What does that mean?
- Consider the sample variance $\mathbf{S} = \mathbf{X^T X/N}$
- It has eigen decomposition $\mathbf{X^T X = V D^2 V^T}$, and the eigenvectors v_j are the principal components directions of X.
- First component (largest eigenvalue) is $\mathbf{z_1} = \mathbf{X}v_1 = \mathbf{u_1}d_1$.
- Variance is $Var(\mathbf{z_1}) = Var(\mathbf{X}v_1) = \frac{d_1^2}{N}$ ($\mathbf{z_1}$ is first principal component of $\mathbf{X}$).
- So this is telling use the ridge is placing more weight on directions in the columnspace of $\mathbf{X}$ that have larger sample variance

Principal Components

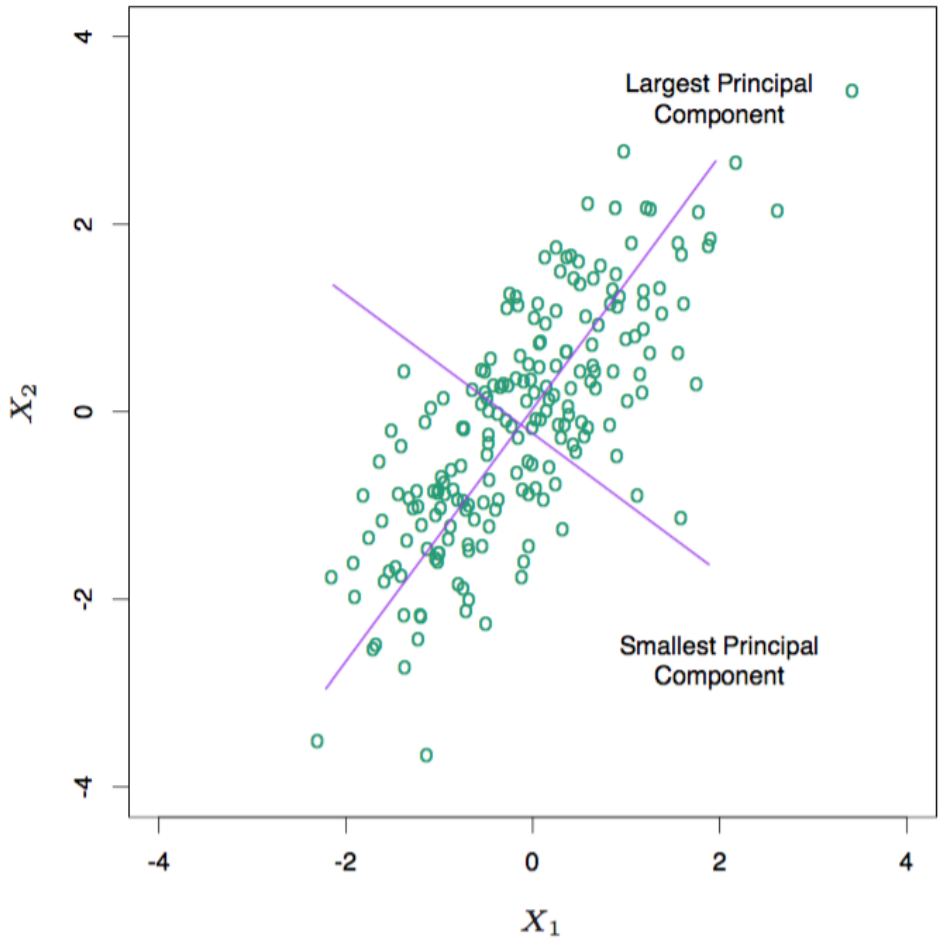
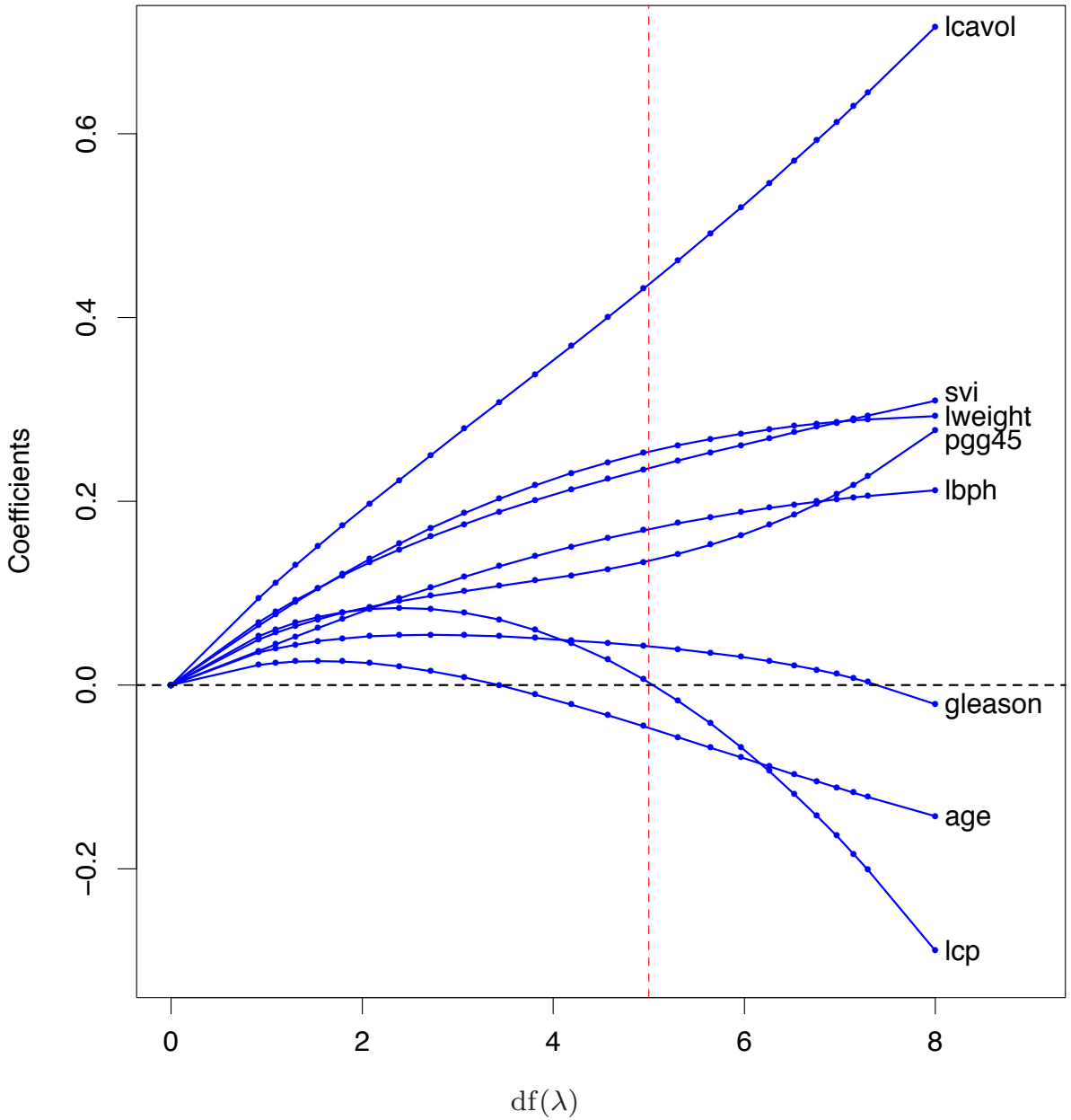


FIGURE 3.9. *Principal components of some input data points. The largest principal component is the direction that maximizes the variance of the projected data, and the smallest principal component minimizes that variance. Ridge regression projects* $\mathbf{y}$ *onto these components, and then shrinks the coefficients of the low–variance components more than the high-variance components.*

Ridge Path


lcavol
svi
lweight
pgg45
lbph
gleason
age
lcp
Coefficients
0.6
0.4
0.2
0.0
−0.2
0
2
4
6
8
df(λ)

LASSO Regression

$$\hat{\beta}^{LASSO} = \arg\min_{\beta} \left[\frac{1}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{K} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{K} |\beta_j| \right]$$

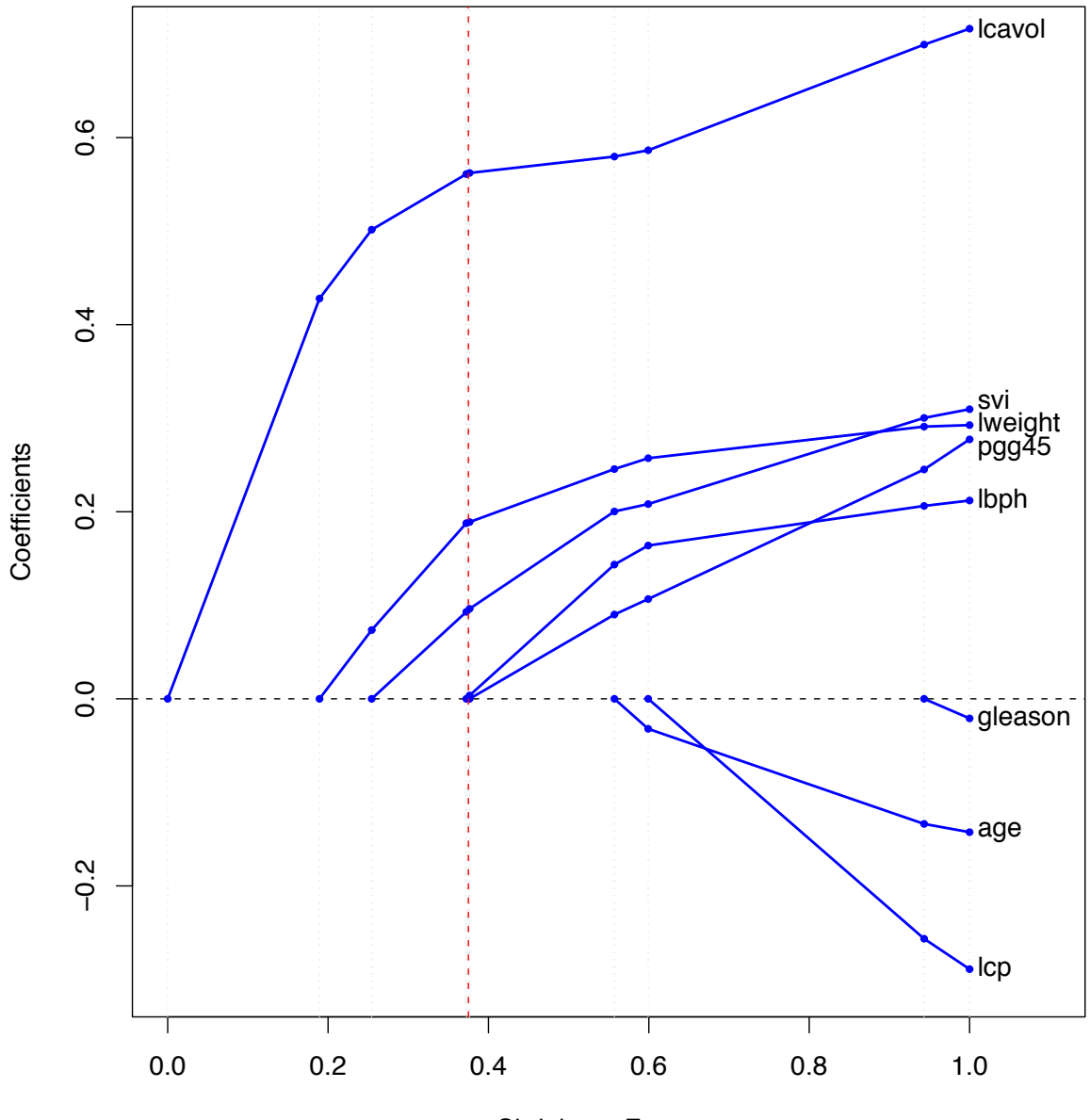
- Penalty is L_1 norm on β.
- Can re-write as a constriant $\sum_{j=1}^{K} |\beta_j| \leq s$
- If X is orthonormal then $\hat{\beta}_j^{LASSO} = sign(\hat{\beta}_j) \cdot (|\hat{\beta}_j| - \lambda)_+$
- In words: we get coefficients that are closer to zero by λ, but coefficients within λ of zero are shrunk to zero. It thus produces models that are sparse.

LASSO vs Subset Selection

Instead of a discrete parameter such as the number of lags p we can continuously penalize additional complexity with λ.
But... is choosing λ any easier than choosing p?

- We call λ the regularization parameter.
- We can choose λ in a way that minimizes expected prediction error (EPE).
- Recall $EPE(\lambda) = E_x E_{y|x}([Y - g(X, \lambda)]^2|X)$.
- In practice most people look at out of sample prediction error rate on a cross validated sample.

LASSO Path


lcavol
svi
lweight
pgg45
lbph
gleason
age
lcp
Coefficients
0.6
0.4
0.2
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0

LASSO vs Ridge

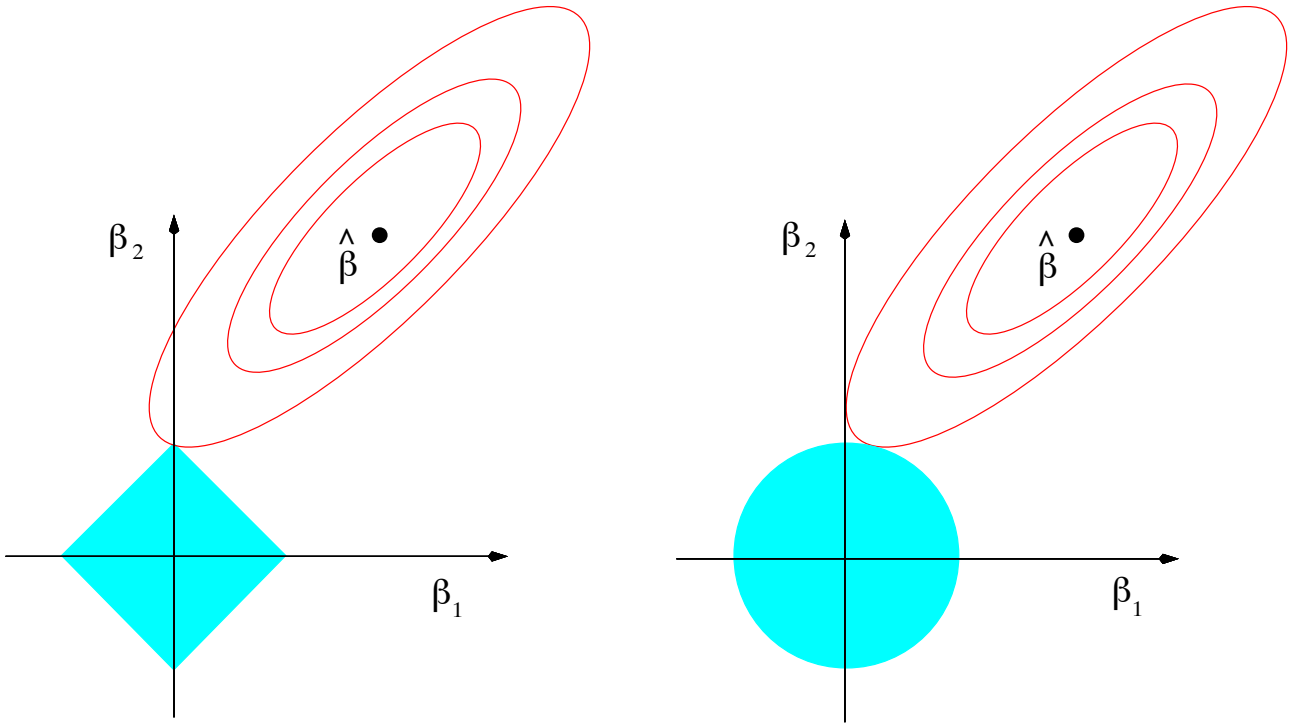


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \leq t$ *and* $\beta_1^2 + \beta_2^2 \leq t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

What is the point?

Ridge:

- Ridge doesn't provide sparsity which can be a good thing.
- It is most helpful (relative to OLS) when X's are highly correlated with one another.
- OLS can set large but imprecise coefficients when it cannot disentangle effects.

LASSO:

- LASSO is useful for variable/feature selection.
- LASSO does not generally posses the oracle property though variants such as adaptive LASSO may.
- People sometimes use LASSO to choose components and then OLS for unbiased coefficient estimated.

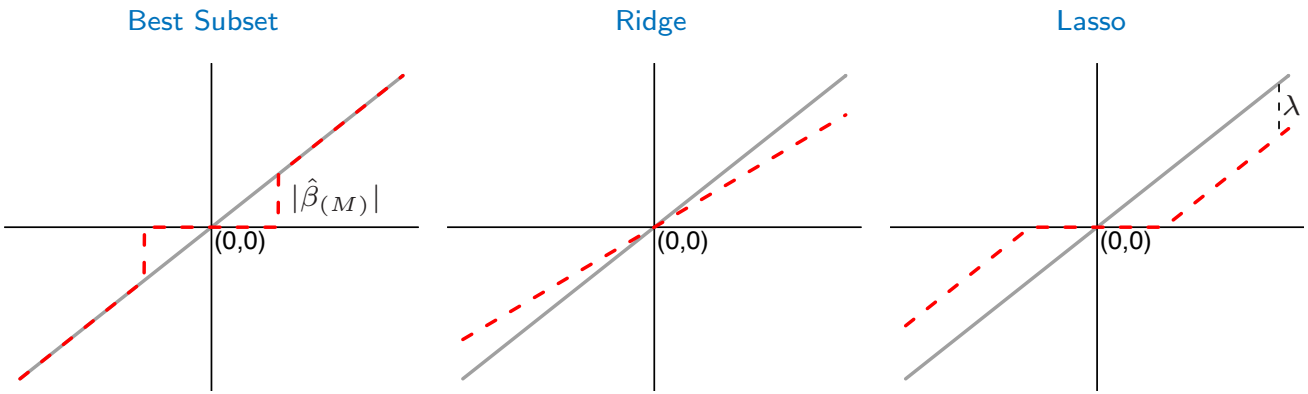
We can actually combine them using elastic net regression:

$$P(\lambda_1, \lambda_2, \beta) = \lambda_1 \sum_{j=1}^{K} |\beta_j| + \lambda_2 \sum_{j=1}^{K} |\beta_j|^2$$

LASSO vs. Ridge

TABLE 3.4. *Estimators of β_j in the case of orthonormal columns of $\mathbf{X}$. M and λ are constants chosen by the corresponding techniques;* sign *denotes the sign of its argument (± 1), and x_+ denotes "positive part" of x. Below the table, estimators are shown by broken red lines. The 45° line in gray shows the unrestricted estimate for reference.*

Estimator	Formula
Best subset (size M)	$\hat{\beta}_j \cdot I(\lvert\hat{\beta}_j\rvert \geq \lvert\hat{\beta}_{(M)}\rvert)$
Ridge	$\hat{\beta}_j/(1+\lambda)$
Lasso	$\text{sign}(\hat{\beta}_j)(\lvert\hat{\beta}_j\rvert - \lambda)_+$



Oracle Property

- An important question with LASSO is whether or not it produces consistent parameter estimates (Generally no).
- We think of asymptotics as taking both $N, p \to \infty$.
- Donohu (2006) shows that for $p > N$ case, when the true model is sparse, LASSO identified correct predictors with high probability (with certain assumptions on $\mathbf{X}$) as we slowly relax the penalty.
- Condition looks like ("good" variables are not too correlated with "bad" variables).

$$\max_{j \in S^c} ||x_j' X_S (X_S' X_S)^{-1}||_1 \leq (1 - \epsilon)$$

- Where X_S are columns corresponding to nonzero coefficients, and S^c are set of columns with zero coefficients (at true value).

Other Extensions

- *Grouped LASSO* for penalzing groups of coefficients at once (like fixed effects)
- *Relaxed LASSO* run LASSO to select coefficients and then run a non-penalized subset regression or LASSO with a less stringent penalty on the subset. (Here CV tends to pick a less strong penalty term λ leading to less shrinkage and bias).
- *SCAD: Smoothly Clipped Absolute Deviation*: do less shrinkage on big coefficients but preserve sparsity

$$\frac{dJ_a(|beta, \lambda)}{d\beta} = \lambda \cdot sgn(\beta) \left[I(|\beta| \leq \lambda) + \frac{(a\lambda - |\beta|)_+}{(a-1)\lambda} I(|\beta| >$$

- *Adaptive LASSO* uses a weighted penalty of the form $\sum_{j=1}^{p} w_j |\beta_j|$ where $W_j = 1/|\hat{\beta}_j|^\nu$ using the OLS estimates as weights. This yields consistent estimates of parameters while retaining the convexity property.

Penalty Comparisons

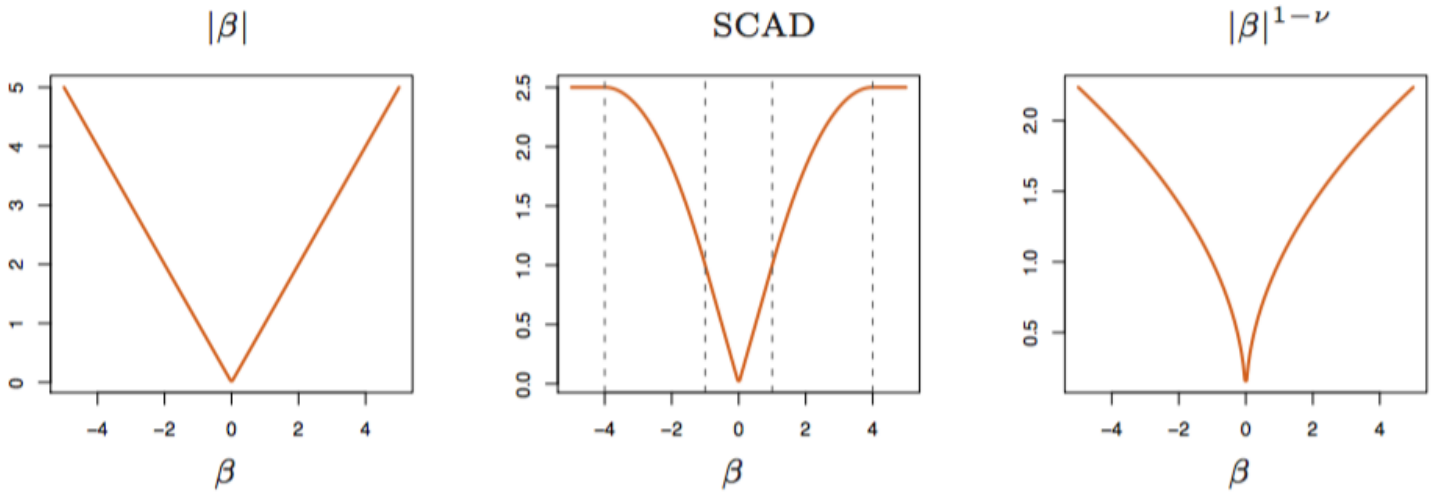


FIGURE 3.20. *The lasso and two alternative non-convex penalties designed to penalize large coefficients less. For SCAD we use* $\lambda = 1$ *and* $a = 4$*, and* $\nu = \frac{1}{2}$ *in the last panel.*

LAR: Least Angle Regression

Consider this alternative to Forward Stagewise Regression:

1. Start with $r = y - \overline{y}$ and $(\beta_1, \ldots, \beta_p) = 0$. (Standardize first!)
2. Find the predictor x_j most correlated with r.
3. SLOWLY move β_j from 0 to its least-squares estimate
4. Update $r \leftarrow r - \delta_j \cdot x_j$.
5. Keep moving x_j in same direction until x_k has as much correlation with updated r,
6. Continue updating (β_j, β_k) in direction of joint least-squares coefficients until some other competitor x_l has as much correlation with r.
7. Continue until all p predictors have entered. After $\min[N-1, p]$ steps we arrive at full OLS solution.

- Optional: If a current least-squares estimate hits zero drop it from the active set and re-estimate the joint least squares direction without it.

LAR: Least Angle Regression

Why do we need LAR?

- It turns out that with the optional step from the previous slide: LAR gives us an easy algorithm to compute the LASSO estimate.
- Actually it does even better – it gives us the full path of LASSO estimates for all values of λ!
- This is actually a relatively new result.

LASSO vs LAR

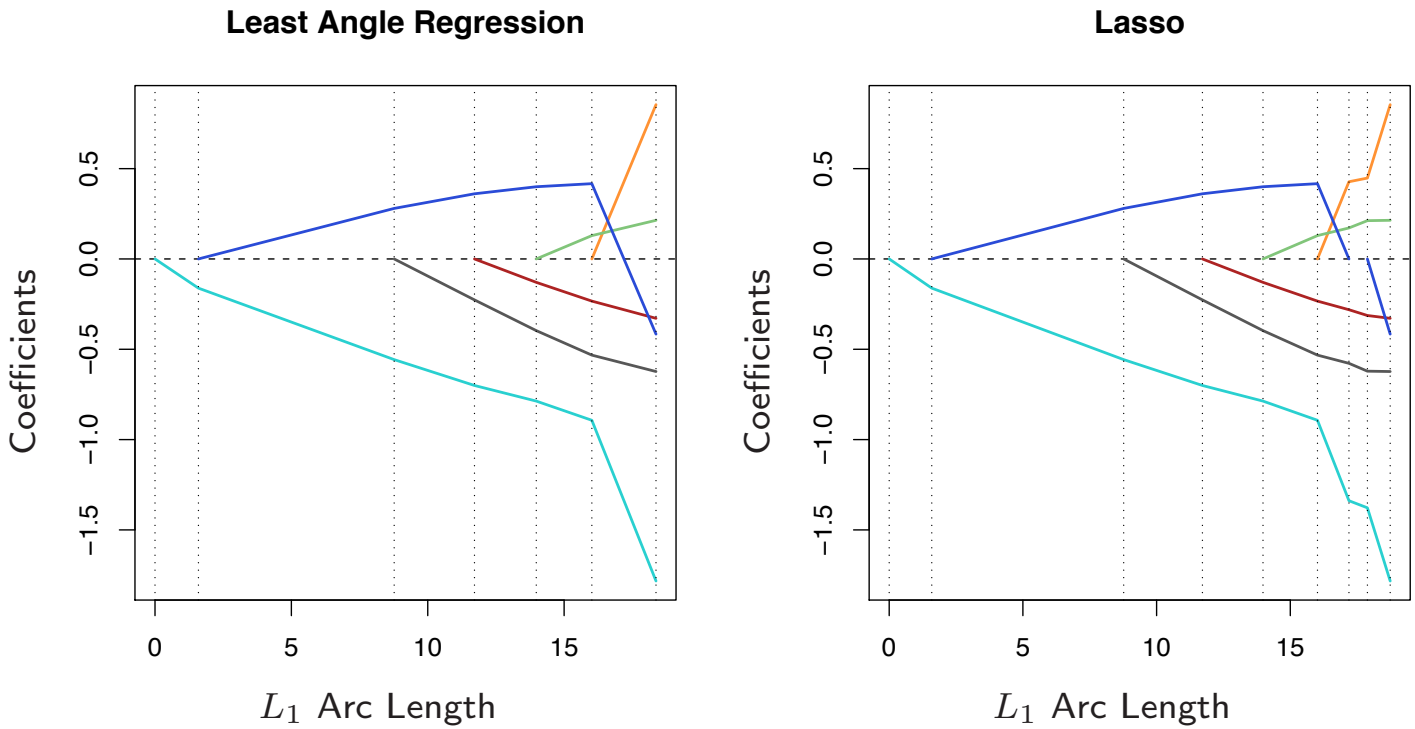


FIGURE 3.15. *Left panel shows the LAR coefficient profiles on the simulated data, as a function of the L_1 arc length. The right panel shows the Lasso profile. They are identical until the dark-blue coefficient crosses zero at an arc length of about* 18.

Overall Comparison

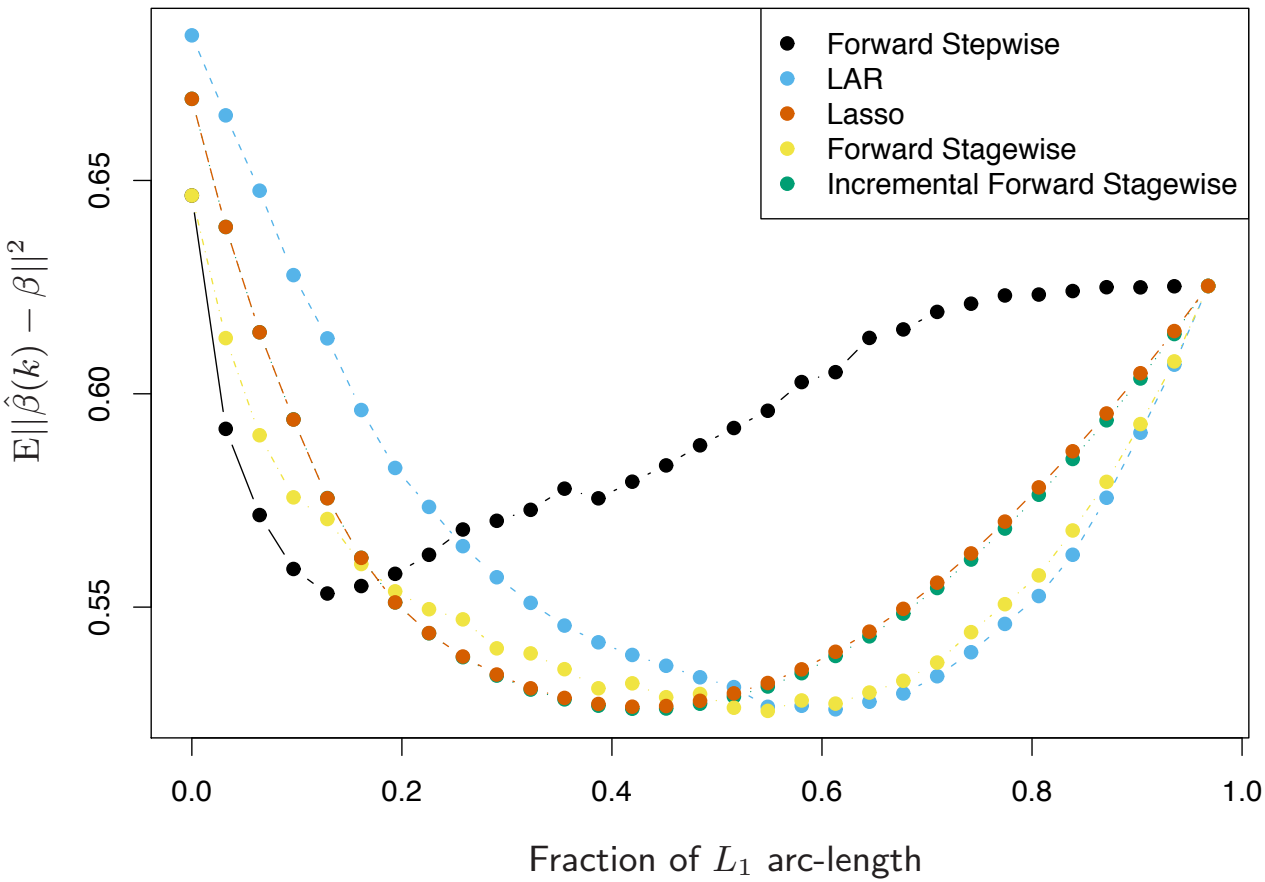


FIGURE 3.16. *Comparison of LAR and lasso with forward stepwise, forward stagewise (FS) and incremental forward stagewise (FS_0) regression. The setup is the same as in Figure 3.6, except $N = 100$ here rather than 300. Here the slower FS regression ultimately outperforms forward stepwise. LAR and lasso show similar behavior to FS and FS_0. Since the procedures take different numbers of steps (across simulation replicates and methods), we plot the MSE as a function of the fraction of total L_1 arc-length toward the least-squares fit.*

Other Data Reduction Techniques

We have other data reduction techniques with a long history in Econometrics

- Principal Components
- Factor Analysis
- Partial Least Squares

Principal Components

Suppose we have a very high dimensional X where we have a high degree of correlation among the components x_j.

- We can start by computing the appropriate correlation matrix $C = E[\tilde{X}'\tilde{X}]$ where $\tilde{X}$ denotes we have standardized each column x_j to have mean zero and variance 1.
- Diagonlize C via the eigen-decomposition $V^{-1}CV = D$ where D is the diagonal matrix of eigenvalues.
- Sort D and the corresponding columns of V in decreasing order of the eigenvalues d_j.
- Choose a subset of $m < K$ eigenvalues and eigenvectors and call that V_m and λ_m
- Compute transformed data: $Z_m = V_m\tilde{X}$ which is of dimension $N \times m$ instead of $N \times K$.

Principal Components

- If $m \ll K$ then we can substantially reduce the dimension of the data.
- The idea is to choose m so that $(Z'Z)$ spans approximately the same space that $(X'X)$ does.
- This works because we use the principal eigenvectors (those with the largest eigenvalues).
- The first eigenvector explains most of the variation in the data, the second the most of the remaining variation, and so on.
- You may also recall that eigenvectors form an orthonormal basis, so each dimension is linearly independent of the others.
- As eigenvalues decline, it means they explain less of the variance.

Principal Components

Output from software will inlcude

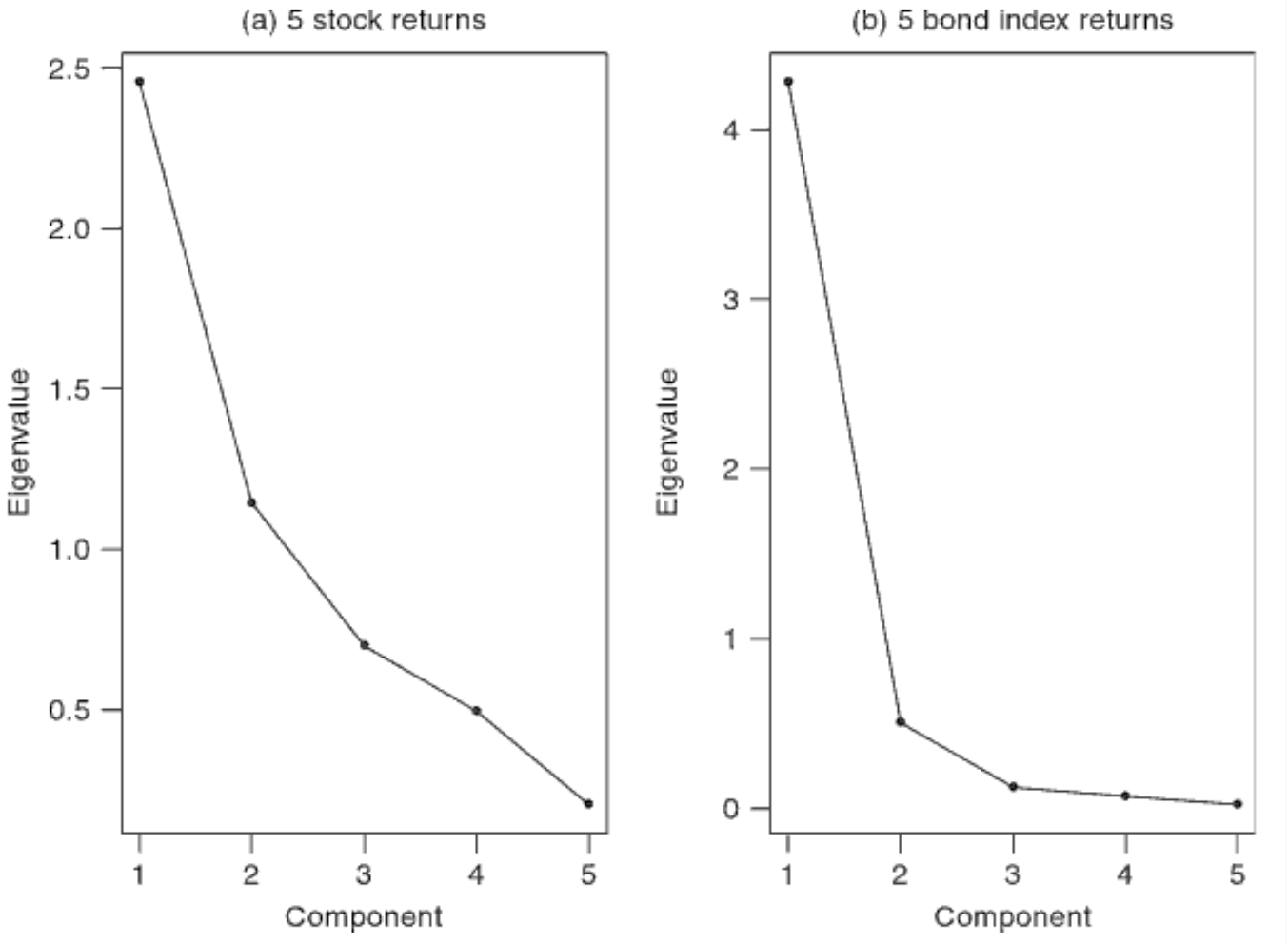
- Coefficients: these transform from $X \to Z$
- Score: these are the transformed Z's
- Latent/Eigenvalue: the corresponding Eigenvalue
- Explained/Cumulative: cumulative explained variance $\sum_{j=1}^{m}(\lambda_j / \sum_k \lambda_k)$
- Stata: `pca`, Matlab: `pca`, R/stats: `princomp`.

Principal Components

How many components?

- Choose # of components by the eigenvalues or % of variance explained
- Common cutoffs are 90-95% of variance.
- Eigenvalue based cutoff rules (only take eigenvalues > 1).
- Most common method is to eyeball the scree plot.

Principal Components



Principal Components

Table 1 Principal Components

Principal Component	Eigenvalue	Proportion of Variance	Cumulative Variance
1	75	26.95%	26.95%
2	43	15.45%	42.40%
3	30	10.78%	53.18%
4	21	7.55%	60.73%
5	19	6.83%	67.55%
6	18	6.47%	74.02%
7	17	6.11%	80.13%
8	11	3.95%	84.08%
9	10	3.59%	87.67%
10	10	3.41%	91.09%
11	9	3.16%	94.25%
12	5	1.80%	96.05%
13	4	1.58%	97.63%

Principal Components

- We can run a regression on principal components Z's and then recover the betas of the X's

$$\hat{y}^{pcr}_{(M)} = \overline{y}\mathbf{1} + \sum_{m=1}^{M} \hat{\theta}_m \mathbf{z}_m$$

- Because principal components are orthogonal we can find coefficients using univariate regression $\hat{\theta}_m = \langle \mathbf{z_m}, \mathbf{y} \rangle / \langle \mathbf{z_m}, \mathbf{z_m} \rangle$.
- We can recover the x coefficients because the PCA is a linear transformation:

$$\hat{\beta}^{PCR} = \sum_{m=1}^{M} \hat{\theta}_m v_m$$

- If $M = P$ (we use all components) then PCR = OLS.
- If $M < P$ then we discard the $p - M$ smallest eigenvalue components
- This is similar to ridge which shrinks β's for components with small eigenvalues.

Principal Components

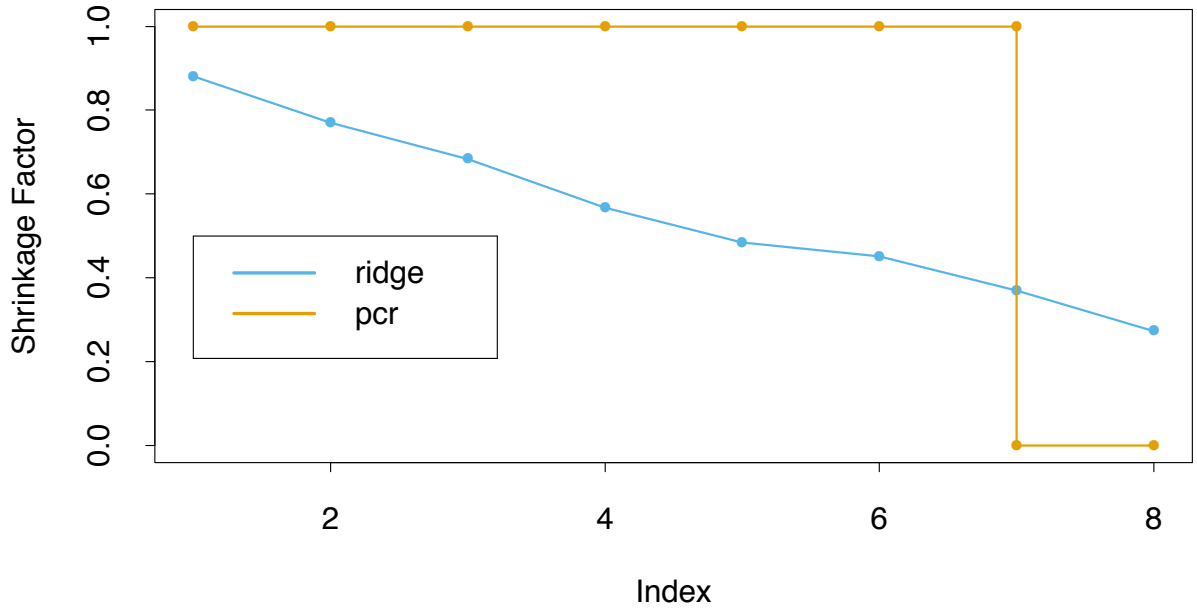


FIGURE 3.17. *Ridge regression shrinks the regression coefficients of the principal components, using shrinkage factors $d_j^2/(d_j^2 + \lambda)$ as in (3.47). Principal component regression truncates them. Shown are the shrinkage and truncation patterns corresponding to Figure 3.7, as a function of the principal component index.*

Hansen Singleton (1982)

- This is the original GMM example, though it comes from macro-finance not microeconometrics

$$\begin{aligned} \max_{c_{t+i}, A_{t+i}} \quad & E_t \sum_{i=0}^{\infty} \frac{U(c_{t+i})}{(1+\delta)^i} \quad \text{subject to} \\ A_{t+i} &= (1+r)A_{t+i-1} + y_{t+i} - c_{t+1} \\ 0 &= \lim_{i \to \infty} E_t A_{t+i} (1+r)^{-i} \end{aligned}$$

- A_t are your investment assets with return r and discount factor δ
- y_t is your income in period t, c_t is your consumption

Hansen Singleton (1982)

- Assume CRRA utility with risk aversion γ

$$U(c_{t+i}) = \frac{c_{t+i}^{1-\gamma}}{1-\gamma}$$

- We can take the first-order/Euler conditions and get:

$$E_t \left(\underbrace{\frac{1+r}{1+\delta} c_{t+1}^{-\gamma} - c_t^{-\gamma}}_{g(x_{t+i},\theta)} \right) = 0$$

- We want to estimate the "deep parameters" $\theta \equiv (\gamma, \delta)$.

Hansen Singleton (1982)

- We can solve for θ without actually solving the dynamic programming problem!
- We just need some instruments z_t that are conditionally independent/orthogonal to $g(x_{t+i}, \theta)$ so that

$$E_t[g(x_{t+i}, \theta)|z_t] = 0 \Rightarrow E_t[g(x_{t+i}, \theta)z_t] = 0$$

- This is nonlinear GMM. We need a matrix of instruments z_t with dimension $N \times Q$ where $Q \geq \dim(\theta)$.
- Where do we get z_t? $\rightarrow$ Economic Theory!

Hansen Singleton (1982)

Consider $E_t[g(x_{t+i}, \theta)|z_t] = 0$.

- The error arises from the error in the Euler equation: deviations between observed behavior and expected behavior.
- If the model is true this optimization error should be independent of anything known to the agent at the time the decision was made.
- We often write: $E_t[g(x_{t+i}, \theta)|z_t, \Omega_t] = 0$ where Ω_t is everything known by the agent up until time t (including the full history).
- If we have some potential instrument z_t and use the full history then $z_{t-1}, z_{t-2}, \ldots$ are all valid instruments
- If we use the conditional moment restriction $E_t[g(x_{t+i}, \theta)|z_t, \Omega_t] = 0$ then any nonlinear function of z_t is also an instrument

$$E_t[g(x_{t+i}, \theta) f(z_{t,t-1,\ldots,0})] = 0$$

Hansen Singleton (1982)

- We have literally infinite possibilities to construct instruments $z_t, z_t^2, z_t^3, z_t \cdot z_{t-1}, \ldots$
- But our instruments could be weak even though we have many of them.
- And they might be highly correlated with each other.
- Carrasco (2012) suggests regularization on the instruments first.
- One possibility for $f(z_t, z_{t-1}, z_{t-2}, \ldots)$ is to take several higher order interactions and take the first Q principal components.
- Even though we might have 100 instruments, after running PCA we might find that 99% of our variation is only in 6 components. In that case we should not try and identify more than 6 parameters!
- Conlon (2014) suggests this as a test of non-identification in nonlinear BLP-type GMM problems.

Factor Models: Take Macroeconometrics!

- Related to PCA is the factor model
- These are frequently used in finance for asset pricing.

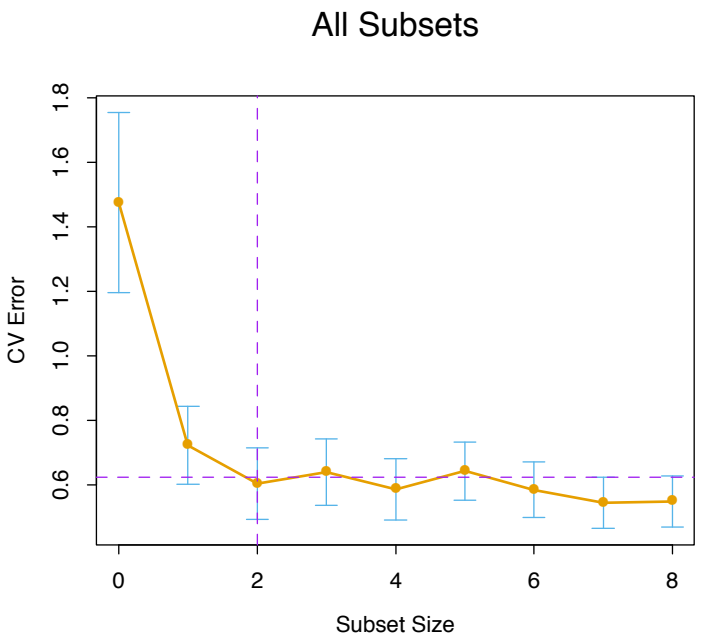
$$r_i = b_0 + b_1 f_1 + b_2 f_2 + \dots b_p f_p + e_i$$

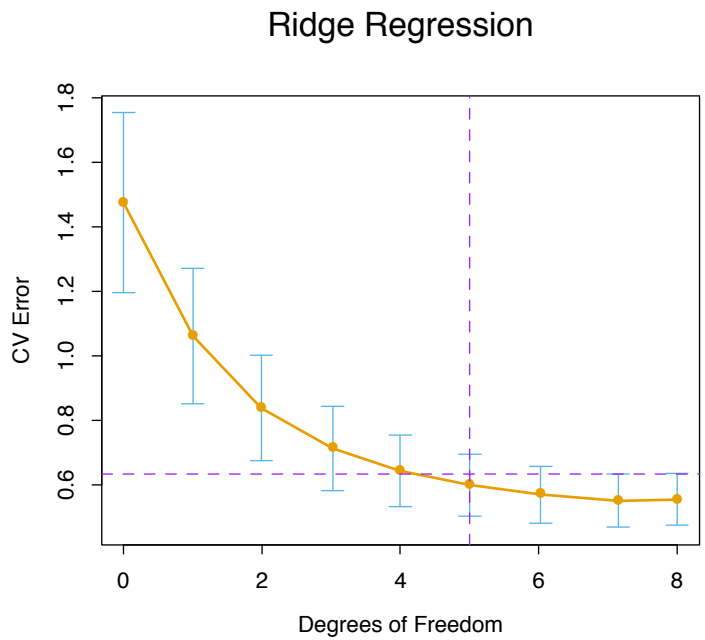
- Typically we choose factors so that $E[f_i] = 0$ and $E[f_i e_i] = 0$ and that $Cov(f_i, f_j) = 0$ for $i \neq j$.
- That is we choose scaled factors to form an orthogonal basis (which makes pricing assets easier).
- Instead of choosing f to best explain $X'X$ we choose it to best explain r by taking linear combinations of our X's.
- CAPM is a single factor model (where the factor is the market return).
- Fama-French have expanded to a 5 factor model (book-to-market, market-cap, profitability, momentum)
- Ross's APT is another form of a factor model.

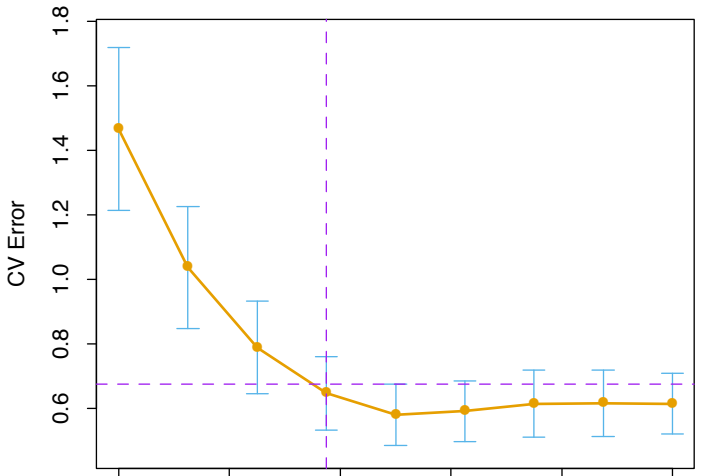
Factor Models: Other Examples

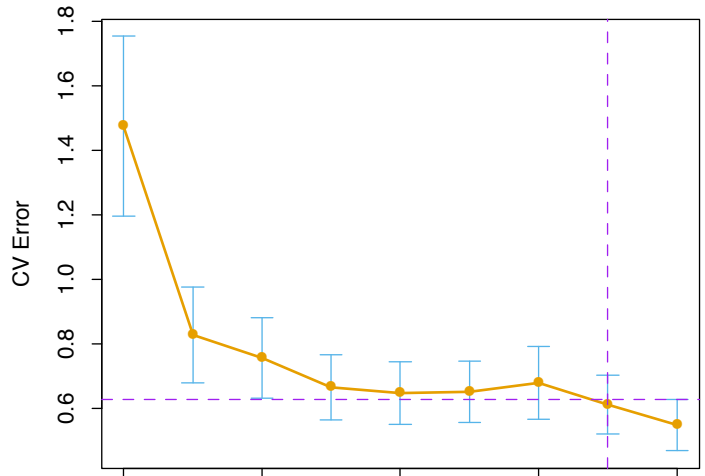
- *Eigenfaces* reduces your face to the first few eigenvalues – this is how face detection works!
- In psychometrics they use data from multiple tests to measure different forms of intelligence (mathematical reasoning, verbal, logical, spatial, etc.)
  - An old literature searched for general intelligence factor g
  - Nobody can tell what the GMAT measured!
- In marketing PCA/factor analyses are used in the construction of *perceptual maps*
- Marketing practitioners use FA/PCA more than academics these days (guess: maybe?)

Overall Comparison







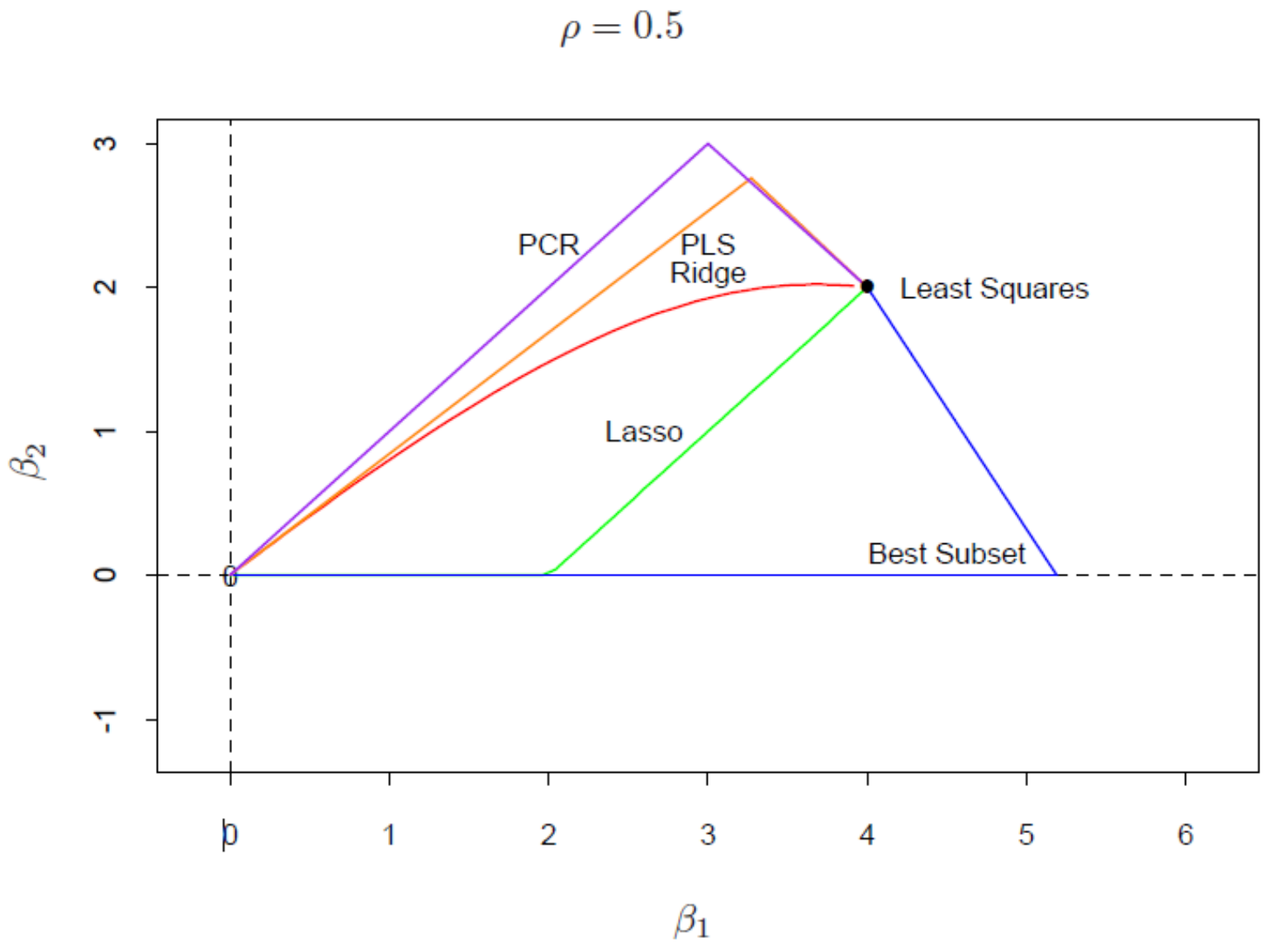


Overall Comparison

TABLE 3.3. *Estimated coefficients and test error results, for different subset and shrinkage methods applied to the prostate data. The blank entries correspond to variables omitted.*

Term	LS	Best Subset	Ridge	Lasso	PCR	PLS
Intercept	2.465	2.477	2.452	2.468	2.497	2.452
lcavol	0.680	0.740	0.420	0.533	0.543	0.419
lweight	0.263	0.316	0.238	0.169	0.289	0.344
age	−0.141		−0.046		−0.152	−0.026
lbph	0.210		0.162	0.002	0.214	0.220
svi	0.305		0.227	0.094	0.315	0.243
lcp	−0.288		0.000		−0.051	0.079
gleason	−0.021		0.040		0.232	0.011
pgg45	0.267		0.133		−0.056	0.084
Test Error	0.521	0.492	0.492	0.479	0.449	0.528
Std Error	0.179	0.143	0.165	0.164	0.105	0.152

Paths


$\rho = 0.5$
PCR
PLS
Ridge
Least Squares
Lasso
Best Subset
β_2
β_1
3
2
1
0
-1
0
1
2
3
4
5
6

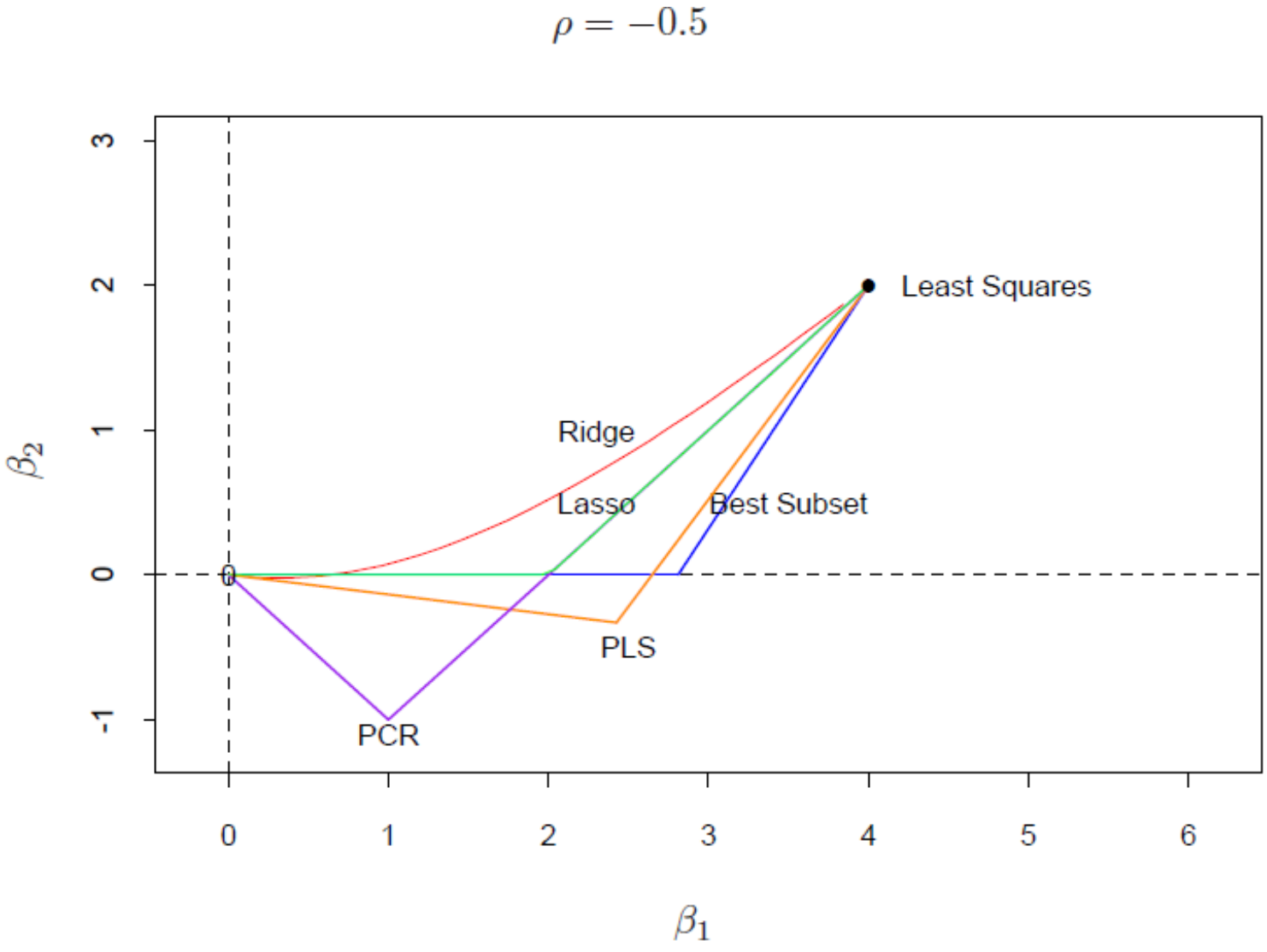


FIGURE 3.18. *Coefficient profiles from different methods for a simple problem: two inputs with correlation* ± 0.5*, and the true regression coefficients* $\beta = (4, 2)$*.*

Implementation

- Routines are highly specialized: there are lots of tricks
- No hope of coding this up on your own!
- In R you can use `glmnet` or `lars`.
- In Python you can use `scikit-learn`
- In most recent Matlab in Stats toolbox you have `lasso` and `ridge`.
- In STATA you can download `.ado` files from Christian Hansen's website (Chicago).

Review AIC/BIC

- AIC tends to select larger models than BIC since it penalizes the number of parameters less heavily.
- These usually depend on ordering potential models by p the number of components and then sequentially fitting them.
- AIC is not consistent: as $N \to \infty$ it may still select too many parameters.
- BIC is consistent: as $N \to \infty$ it will select the correct number of parameters.
- Of course for finite-sample $N < \infty$ anything can happen.

Where does it come from?

How do we come up with these penalized regressions?

- AIC/BIC arise from considering the likelihood ratio test (LRT) of a maximum likelihood estimator and making a lot of assumptions.
- AIC arises from minimizing the Expected KLIC.

$$\begin{aligned} KLIC(f,g) &= \int f(y)\log(f(y))\partial y - \int f(y)\log(g(y))\partial y \\ &= C_f - E\log(g(y)) \end{aligned}$$

- Low values of KLIC mean the models are similar.

Where does it come from?

How do we come up with these penalized regressions?

- Recall that OLS is a ML estimator in the case where ε is normally distributed.

$$D = -2\ln\left(\frac{\text{Likelihood } H_0}{\text{Likelihood } H_a}\right) = -2\ln\underbrace{\left(\frac{(\sup L(\theta|x) : \theta \in \Theta_0)}{(\sup L(\theta|x) : \theta \in \Theta)}\right.}_{\Lambda(x)}$$

- If the models are nested then $\Theta_0 \subset \Theta$ and $\dim(\Theta) - \dim(\Theta_0) = q$ then as $N \to \infty$ we have that $D \to^d \chi^2(q)$.

Non-nested cases

Many cases we are interested in are not strictly nested

- Should I include x_2 OR x_3 in my regression? (partially overlapping)
- Is the correct distribution $f(y|x, \theta)$ normal or log-normal? (non-overlapping)

Non-nested cases

- Cox (1961) suggested the following (often infeasible solution) by assuming that F_θ is the true model.

$$LR(\hat{\theta}, \hat{\gamma}) = L_f(\hat{\theta}) - L_g(\hat{\gamma}) = \sum_{i=1}^{N} \ln \frac{f(y_i|x_i, \hat{\theta})}{g(y_i|x_i, \hat{\gamma})}$$

- Depending on which the true model is you could reject F_θ for G_γ and vice versa!
- Deriving the test statistic is hard (and specific to F_θ) because we must obtain $E_f[\ln \frac{f(y_i|x_i,\hat{\theta})}{g(y_i|x_i,\hat{\gamma})}]$.
- Similar to AIC in that we are minimizing KLIC over F_θ.

Vuong Test

$$H_0 : E_{h(y|x)}\left[\frac{f(y|x,\theta)}{g(y|x,\gamma)}\right] = 0$$
$$\rightarrow E_h[\ln(h/g)] - E_h[\ln(h/f)] = 0$$

- Instead of taking expectation with respect to one of two distributions, we take it with respect to $h(y|x)$ the unknown but true distribution.
- Same as testing whether two densities (f, g) have same KLIC.
- The main result is that (details in 8.5 of CT):

$$\frac{1}{\sqrt{N}} LR(\hat{\theta}, \hat{\gamma}) \rightarrow^d N[0, \omega_*^2]$$

$$\omega_*^2 = V_0\left[\ln \frac{f(y|x,\hat{\theta})}{g(y|x,\hat{\gamma})}\right]$$

Non-nested cases

Many cases we are interested in are not strictly nested

- Should I include x_2 OR x_3 in my regression? (partially overlapping)
- Is the correct distribution $f(y|x, \theta)$ normal or log-normal? (non-overlapping)

Non-nested cases

- Cox (1961) suggested the following (often infeasible solution) by assuming that F_θ is the true model.

$$LR(\hat{\theta}, \hat{\gamma}) = L_f(\hat{\theta}) - L_g(\hat{\gamma}) = \sum_{i=1}^{N} \ln \frac{f(y_i|x_i, \hat{\theta})}{g(y_i|x_i, \hat{\gamma})}$$

- Depending on which the true model is you could reject F_θ for G_γ and vice versa!
- Deriving the test statistic is hard (and specific to F_θ) because we must obtain $E_f[\ln \frac{f(y_i|x_i,\hat{\theta})}{g(y_i|x_i,\hat{\gamma})}]$.
- Similar to AIC in that we are minimizing KLIC over F_θ.

Vuong Test

$$H_0 : E_{h(y|x)}\left[\frac{f(y|x,\theta)}{g(y|x,\gamma)}\right] = 0$$
$$\rightarrow E_h[\ln(h/g)] - E_h[\ln(h/f)] = 0$$

- Instead of taking expectation with respect to one of two distributions, we take it with respect to $h(y|x)$ the unknown but true distribution.
- Same as testing whether two densities (f, g) have same KLIC.
- The main result is that (details in 8.5 of CT):

$$\frac{1}{\sqrt{N}} LR(\hat{\theta}, \hat{\gamma}) \rightarrow^d N[0, \omega_*^2]$$

$$\omega_*^2 = V_0\left[\ln \frac{f(y|x,\hat{\theta})}{g(y|x,\hat{\gamma})}\right]$$