Data Science Practice

STATS 369 Coursebook: Week 5

Plan for this week

Regularised regression

- [L13] Motivation & Overview ;
- [L14] Regularised regression
  - LASSO regression
  - Ridge regression
- [L15] Examples

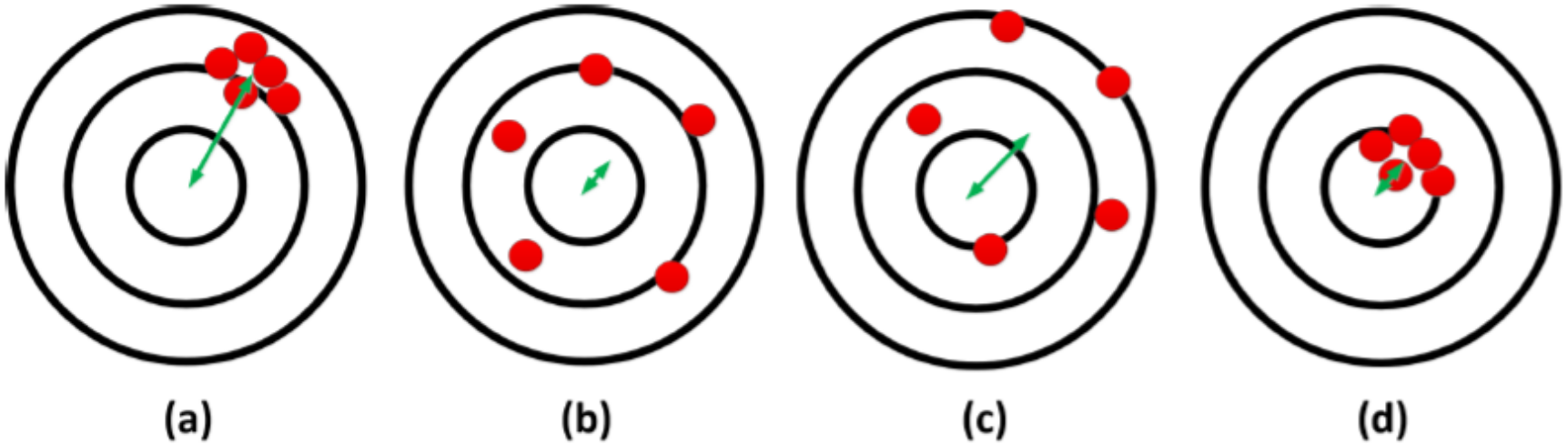
Bias-Variance Tradeoff (re-cap)

Bias-Variance Tradeoff (revisited)

Record that for a given test data point (x_0, y_0), the *expected MSPE* can be decomposed into three quantities, i.e.

$$\mathbb{E}(MSPE) = Bias^2 + Variance + Irreducible\ Error$$

Bias vs. Variance



The center of the 'dart board' represents the true value of a variable Y, and the red points are predictions.

a) high bias and low variance;

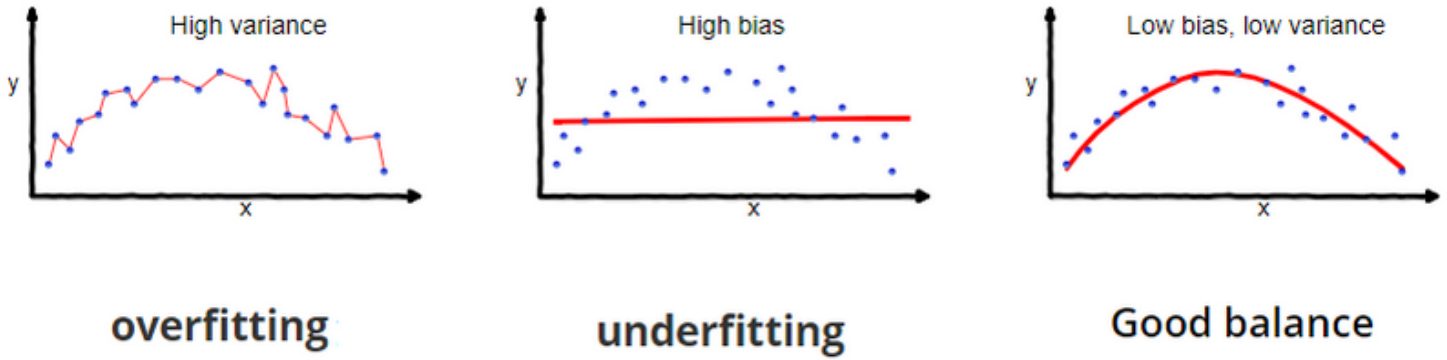
b) low bias and high variance; (e.g. OLS)

Image Credit

c) high bias and high variance; (**worse case scenario**)

d) low bias and low variance. (**best case scenario**)

Overfitting vs. Underfitting



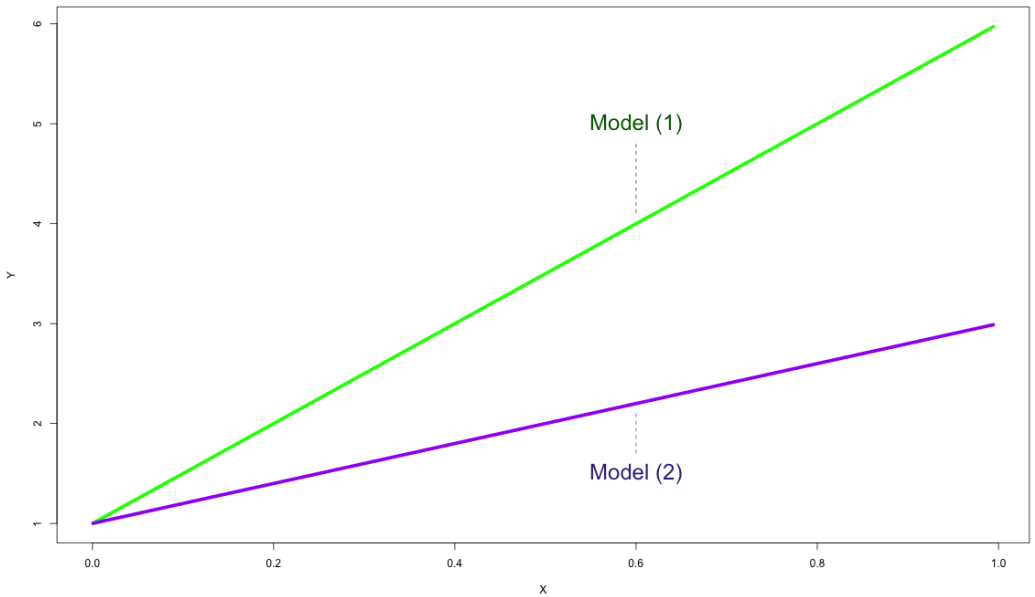
An overfitting model

- generally low on apparent error and fits the training data well, has **low bias** for the test data.
- sensitive to changes in training data, i.e. changes its shape a lot with new training set — **high variance**.

An underfitting model

- often high on apparent error and fits the training data poor, has **high bias** for the test data.
- insensitive to changes in training data, i.e. maintain its shape with new training set — **low variance**.

What drives the high variance?



- Two models are fitted to the same data with a same intercept $\beta_0 = 1$ and different slopes.
  - Model (1): $\hat{Y} = 1 + 5X$
  - Model (2): $\hat{Y} = 1 + 2X$
- One unit change in X, model (1) changes the prediction $\hat{Y}$ by 5 units; whereas model (2) only changes by 2 units. i.e., $\beta_1 = 5$ vs. $\beta_2 = 2$.
- Model (1) is *more sensitive* to the change in input data X — *higher* variance!

Observation: Given a same X, high coefficient value β drives the high variance in $\hat{Y}$.

A side note: scaling matters!

```
set.seed(123)
X <- runif(20)
Y <- rnorm(20)
f1 <- lm(Y~X)
f1$coefficients
```

```
## (Intercept)           X
##   0.6162818  -1.3148861
```

```
set.seed(123)
X <- 10 * runif(20)
Y <- rnorm(20)
f2 <- lm(Y~X)
f2$coefficients
```

```
## (Intercept)           X
##   0.6162818  -0.1314886
```

```
set.seed(123)
X <- 1000 * runif(20)
Y <- rnorm(20)
f3 <- lm(Y~X)
f3$coefficients
```

```
##  (Intercept)           X
##  0.616281841 -0.001314886
```

Observation: The increase in X scale does not affect the intercept β_0, but it decreases the slope β_1 proportionally.

Comment: Always check the scaling of your data set before building any models — it DOES matter!

Regularisation Overview

Regularisation

Regularisation is a process of introducing **additional small bias** to the model in order to gain significant reduction in **variance** and therefore, achieving a overall better modelling outcome.

Linear Regression with OLS

- The model

$$\hat{Y} = X\beta$$

- Aim to minimise the *loss function*

$$Loss = nlog(RSS).$$

Regularised Regression

- The model

$$\hat{Y} = X\beta + bias(\beta)$$

- Aim to minimise the *penalised loss function*

$$Penalised\ Loss = nlog(RSS) + \lambda * g(\beta)$$

Regularised regression is also called **penalised regression**, i.e., it penalises *large* estimates of β and tries to *shrink* them toward zero.

Best-subset is one form of regularisation

Let us re-write the penalised RSS as follows:

$$
\begin{aligned}
Penalised\ RSS &= nlog(RSS) + \lambda p \\
&= nlog(RSS) + \lambda \underbrace{\sum_{i}^{P} I(\beta_i \neq 0)}_{penalty\ term} \quad [i.e.\ g(\beta) = \sum I(\beta_i \neq 0)] \\
&= nlog(RSS) + \lambda ||\beta||_{m \to 0}
\end{aligned}
$$

- You can think of the AIC-type penalty as the limit of $\lambda||\beta||_m$ as m goes to zero. This is often called the l_0-form penalty.
- The term is not continuous, not convex, not even smooth! Hence, it is SLOW!
- The best-subset constraints some β to be exactly zero while letting others float freely.
- λ is a *penalty coefficient*. e.g., AIC has $\lambda = 2$, BIC has $\lambda = log(n)$. The **higher the λ**, the stronger the penalty, resulting in a **smaller model**.

Other forms of penalty

- l_0 form penalty — as in variable (subset) selection

$$Penalised\ RSS = nlog(RSS) + \lambda \sum_{i}^{P} I(\beta_i \neq 0) =: nlog(RSS) + \lambda ||\beta||_0$$

- l_1 form penalty — **LASSO** regression

$$Penalised\ RSS = nlog(RSS) + \lambda \sum_{i}^{P} |\beta_i| =: nlog(RSS) + \lambda ||\beta||_1$$

- l_2 form penalty — **Ridge** regression

$$Penalised\ RSS = nlog(RSS) + \lambda \sum_{i}^{P} (\beta_i)^2 =: nlog(RSS) + \lambda ||\beta||_2$$

An example

Let us use a simple example to see how penalised regression works.

Step 1: setup

```
set.seed(123)

n <- 300  # number of obs.
p <- 3 # number of predictors

s.mt <- matrix(rnorm(p^2), nrow = p)
sigma.mt <- t(s.mt) %*% s.mt
X <- MASS::mvrnorm(n, mu = rep(0,p), Sigma = sigma.mt)
Y <- 2 * X[,1] + 4 * X[,2] + 8 * X[,3] + rnorm(n, 0, 1.5)

X <- scale(X) # standandise X

lm(Y~X)$coefficients
```

```
## (Intercept)          X1          X2          X3
##   0.1396881   3.5023427   6.3886088  11.6807276
```

An example

Let us use a simple example to see how penalised regression works.

Step 2: define functions

```
# L1 norm penalty (LASSO)
pen_rss_L1.fn <- function(beta){
  n*log(sum((beta[1] + X%*%beta[2:4] - Y)^2)) + lambda * sum(abs(beta[2:4]))
}


# L2 norm penalty (Ridge)
pen_rss_L2.fn <- function(beta){
  n*log(sum((beta[1] + X%*%beta[2:4] - Y)^2)) + lambda * sum(beta[2:4]^2)
}
```

An example

Let us use a simple example to see how penalised regression works.

Step 3: run `optim()`

```
lambda = 2
# starting with all 4 betas = 0, i.e. NULL model, optim() finds the minimum to the penalised RSS.
optim(rep(0,4), pen_rss_L1.fn)$par
```

```
## [1]  0.1395602  3.4992756  6.3778264 11.6670674
```

```
optim(rep(0,4), pen_rss_L2.fn)$par
```

```
## [1]  0.1396593  3.6655437  6.0237727 11.4009910
```

```
lm(Y~X)$coefficients
```

```
## (Intercept)          X1          X2          X3
##   0.1396881   3.5023427   6.3886088  11.6807276
```

An example

Let us use a simple example to see how penalised regression works.

Step 4: now try larger λs

```
lambda = 20
optim(rep(0,4), pen_rss_L1.fn)$par
```

```
## [1]  0.1397644  3.4757644  6.2744118 11.5426295
```

```
optim(rep(0,4), pen_rss_L2.fn)$par
```

```
## [1] 0.1380733 0.5566516 0.5031452 0.9388332
```

```
lm(Y~X)$coefficients
```

```
## (Intercept)          X1          X2          X3
##   0.1396881   3.5023427   6.3886088  11.6807276
```

```
lambda = 100
optim(rep(0,4), pen_rss_L1.fn)$par
```

```
## [1]  1.314433e-01  4.689953e-07 -8.162618e-10  1.0
```

```
optim(rep(0,4), pen_rss_L2.fn)$par
```

```
## [1] 0.13722239 0.10726276 0.09707142 0.17132932
```

```
lm(Y~X)$coefficients
```

```
## (Intercept)          X1          X2          X3
##   0.1396881   3.5023427   6.3886088  11.6807276
```

Observations: large λ => smaller β; l_1 can shrink some β to zero; l_2 tends to shrink all β to a similar magnitude.

Regularisation as contrained minimisation

In regression, regularisation == constrained minimisation, i.e. $\underline{Penalised\ RSS = Loss + Constraint.}$ So...

- For best-subset selection, minimising

$$nlog(RSS) + \lambda p$$

  for specific λ is equivalent to minimising $nlog(RSS)$ subject to the *constraint* $p \leq p_\lambda$.

- For **lasso regression**, minimising

$$nlog(RSS) + \lambda ||\beta||_1$$

  for specific λ is equivalent to minimising $nlog(RSS)$ subject to the *constraint* $||\beta||_1 \leq c$ for some $c = c(\lambda)$.

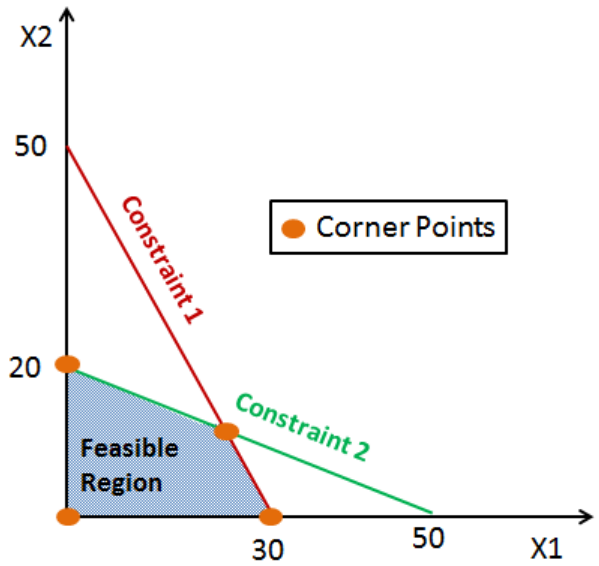
- For **ridge regression**, minimising

$$nlog(RSS) + \lambda ||\beta||_2$$

  for specific λ is equivalent to minimising $nlog(RSS)$ subject to the *constraint* $||\beta||_2 \leq c$ for some $c = c(\lambda)$.

A side note: basic optimisation concepts

Here are some of the basic terminologies from the optimisation practice.

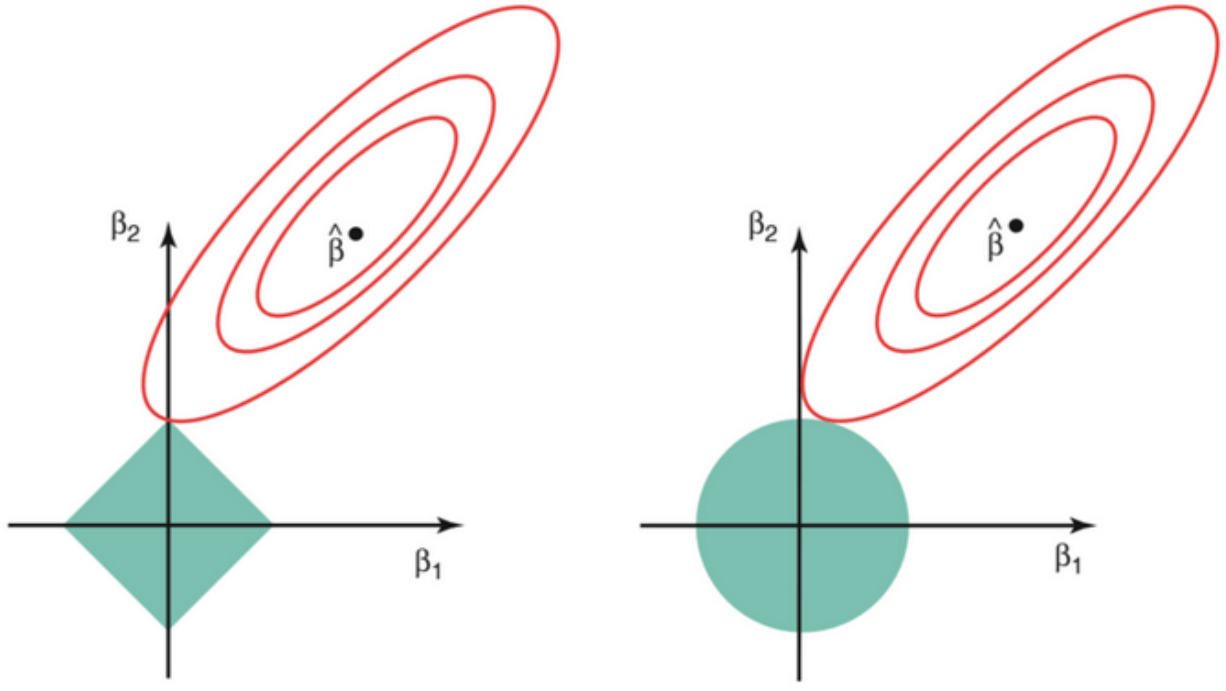


- A **constraint set** is a set of restrictions either in equality or inequality form. It restricts the values of *decision variables* (e.g. βs). The constraint sets forms a *decision boundary*, which — if well-behaved — is convex!
- The **objective function** is the function we want to optimise (e.g. minimise $nlog(RSS)$).
- The **feasible region** is a region that covers all possible values of decision variable that meets *all* the constraints.
- An (feasible) optimal solution occurs when the objective function and the feasible region meets.

Image Credit

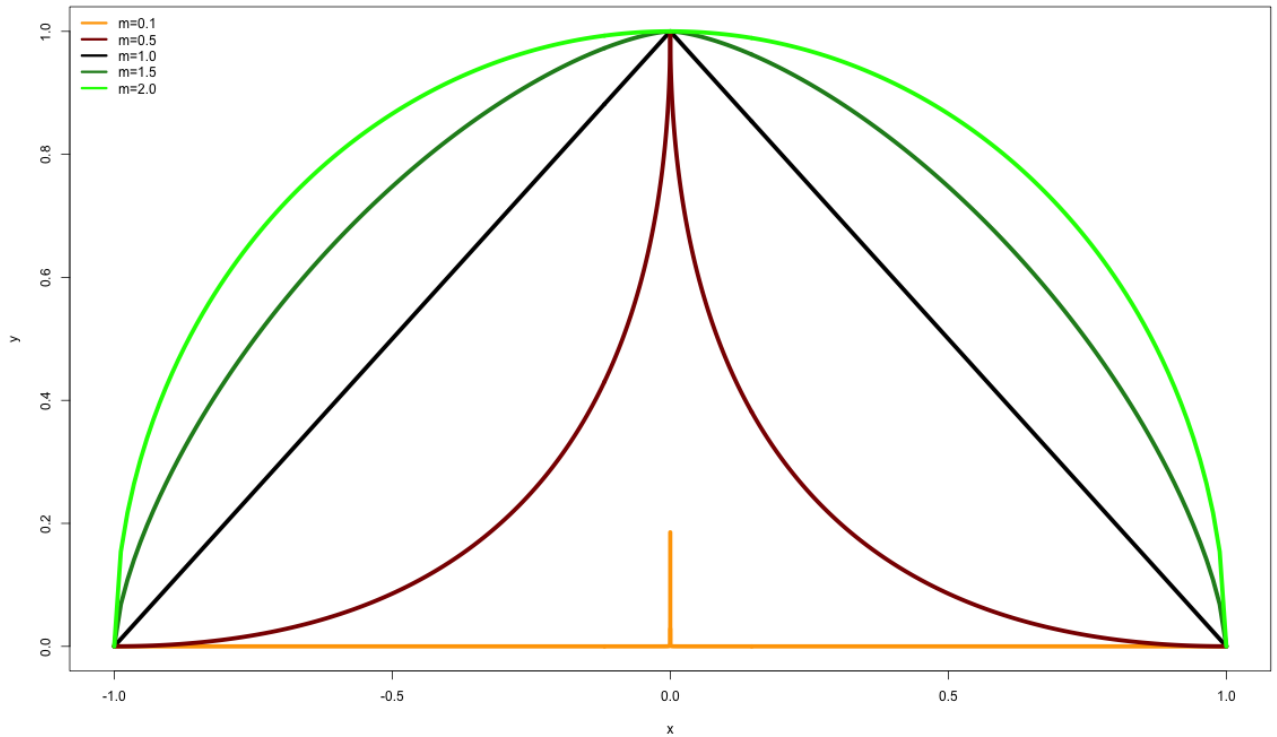
Regularisation as contrained minimisation

For regularised regression, the objective function remains the same, but the shape of the feasible regions differ.



A visual explanation

The general form of penalty term is $\lambda||\beta||_m$. The shape of the feasible region changes as m changes. As m increases, the decision boundary for the constraint set becomes less *pointy* and more smooth. e.g.



Regularisation in general

- In general, **regularisation** use various penalty terms as additional information to prevent **overfitting**.
  - Again, we do not try to minimise the apparent error in *training* data
  - Instead, we want to minimise MSPE for *future test* data.

- So, we minimise error with a **penalty** for model being too complicated. The penalty works well for regression. Other estimators have different ways to avoid overfitting too.
  - It is not just the number of variables; heuristically, large β is also considered as 'complex' model. The aim is to penalise for complex models — both in *quantity* and *magnitude*, i.e. Occam's razor principle
  - Variable selection only deals with *quantity* aspect of the model complexity; whereas LASSO and Ridge regression deal with both through **sparsity** and **shrinkage**. (see later)
- This may not work if the truth is really complex — but it would be hard to build a good model anyway!

LASSO and Ridge Regression

STATS369

Course develped by Thomas Lumley (Department of Statistics), with additional teaching and work by Lisa Chen (Statistics), David Welch (Computer Science), Yalu Wen (Statistics)
Faculty of Science, University of Auckland

Plan for this week

Regularised regression

- [L13] Motivation & Overview 👍
- [L14] Regularised regression
  - LASSO regression
  - Ridge regression
- [L15] Examples

LASSO Regression

Motivation

The ordinary least square (OLS) is one of the most commonly used (linear) model fitting methods. However, it may not perform well:

- if the number of observation n is similar to the number of predictors, p, i.e. $n \approx p$, there can be lots of variability in the OLS fit. Hence, cause **overfitting** and result in poor prediction on (unseen) future data.
- if $n < p$, there is no longer a unique OLS coefficient estimate $\hat{\beta}$; the variance is *infinite* so the method cannot be used at all.

An example

```
set.seed(123)
y <- rnorm(3); X <- matrix(rnorm(3 * 6), nrow = 3, ncol = 6)
summary(lm(y~X))
```

```
##
## Call:
## lm(formula = y ~ X)
##
## Residuals:
## ALL 3 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (4 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.5743        NaN     NaN      NaN
## X1            1.1832        NaN     NaN      NaN
## X2           -0.1511        NaN     NaN      NaN
## X3                NA         NA      NA       NA
## X4                NA         NA      NA       NA
## X5                NA         NA      NA       NA
## X6                NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,    Adjusted R-squared:    NaN
## F-statistic:   NaN on 2 and 0 DF,  p-value: NA
```

Lasso Regression

We can smooth the penalty term by choosing a different function over β.

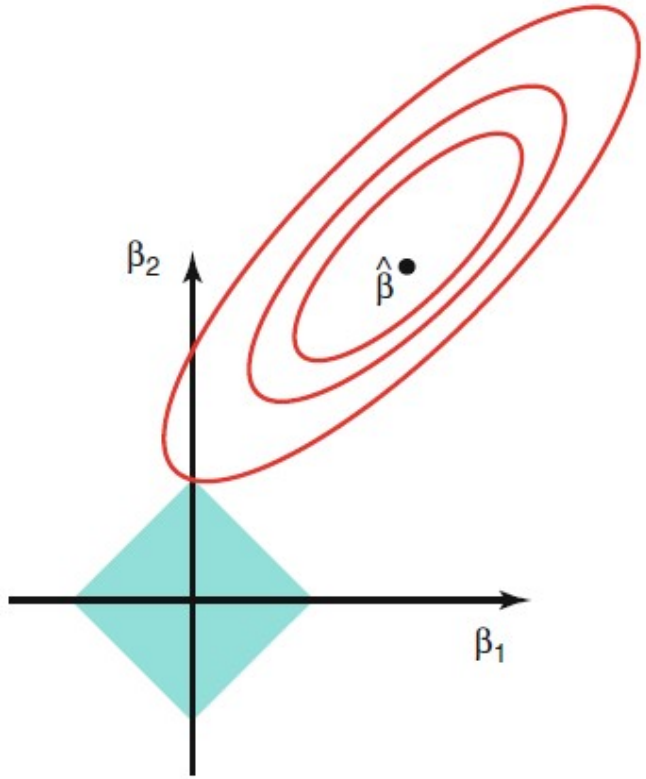
$$nlog(RSS) + \lambda \sum_{i}^{P} |\beta_i| = nlog(RSS) + \lambda ||\beta||_1$$

- The penalty term is called the **Lasso-type penalty**, or l_1-form penalty (i.e. corresponds to an l_1 constraint on β 's). The resulting model is a **Lasso regression** model[*].
- By using the absolute value, instead of indicator functions, it is now continuous and convex. That means
  - For a fixed λ, we can minimise the *objective function* by just heading 'downhill' from any given starting point.
  - Finding the truly optimal β_λ for one given λ is fast.

[*] Lasso stands for 'Least Absolute Shrinkage and Selection Operator'.

Geometric interpretation

The l_1 penalty in lasso means there are 'corners' in the constraint set. In 2D, this corresponds to a diamond shape.



- If the sum of squares (RSS) 'touches' one of these corners, the corresponding $\hat{\beta}$ is shrunk to 0.
- As number of variables p increases, the multi-dimensional diamond has increasing no. of corners, so it is more likely that some $\hat{\beta}$ will be set to 0 — Lasso performing shrinkage and effectively **variable selection**.
- In comparison with best-subset selection, Lasso uses *soft thresholding*: as the smoothing parameter is varied, the sample path of the estimates moves continuously to zero.

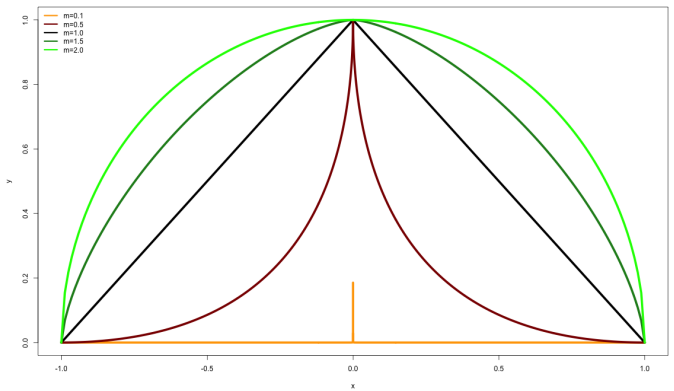
Variable selection with LASSO

Lasso performs variables selection by making β_λ equal to or reduced to zero.

- **Sparse**: many entries of β_λ equal to 0.
  - If λ is sufficiently large, lasso does variable selection by driving some coefficients to 0, leading to a *sparse* model.
  - Sparse model is convenient and often plausible.
- **Shrinkage**: values of β_λ get shrunk to (close to) 0.
  - In general, entries of β_λ are closer to zero if one fits the same predictions without the penalty.
  - Lasso does variable selection by shrinking coefficients towards 0.
  - Shrinkage is intermediate between a variable being *in* the model and *out* of the model. Such flexibility is useful.

Sparsity & Shrinkage

The general form of penalty term is $\lambda||\beta||_m$ for some value $m > 0$. As m increases, the decision boundary for the constraint set becomes less *pointy* and more smooth.



- AIC-type penalty
  - $m \to 0$. It is maximally pointy. $\hat{\beta}$ is sparse but **no shrinkage** (i.e. free value if not zero).
- Lasso-type penalty
  - $m = 1$, the constraint set is still point but is convex. The lasso is the only value of m such that it gives a convex constraint set and a sparse estimator. **There is sparsity AND shrinkage.**
- Ridge-type penalty
  - $m = 2$, the penalty is not pointy at all. The boundary is smooth, the constraint set is convex. **There is shrinkage but no sparsity.**

Compute β_λ

Varying λ

- For large enough λ, the optimal model has no predictors — penalised too heavily.
- If we know $\hat{\beta}_\lambda$ for one value of λ, it makes a good starting point for a slightly smaller λ. Hence, we can alternate
  - reducing λ slightly
  - optimise to find best β

  So, finding $\hat{\beta}_\lambda$ for all λs is *fast*.
- There is no problem with having more observations than variables, $n > p$ — we just need to stop decreasing λ before n variables are included in the best model.

Special structure

- In particular for linear regression, the path of β_λ with varying λ is made of straight lines, so it can be computed in no more time than fitting the largest model being considered.
- Similar penalised estimators for other regression models (e.g., logistic regression) may take slightly longer, but still fast.

Choosing λ

- As usual, we can choose the hyperparameter λ by **cross-validation** or validation in a separate data set (if available).
- Doing cross-validation for lasso is a much faster computation than doing the same for subset selection.

Ridge Regression

Motivation

When we fit a regression model $Y\ X\beta$ to a collection of predictors / features $X = (X_1, X_2, \ldots, X_p)$, if two or more predictors are highly correlated — a problem called **multicollinearity** arises, which can significantly impact the results of regression and limit the conclusions we can draw.

- The estimated coefficient $\hat{\beta}$ will depend on what other X variables are in the model.
- The standard error $se(\hat{\beta})$ increases as more X are added to the model.
- So the inference (e.g., p-value, confidence interval) based on the model is vastly inaccurate.
- It may still be okay to use model for prediction, but the variability of prediction will be big (again, depends on which X are in the model).

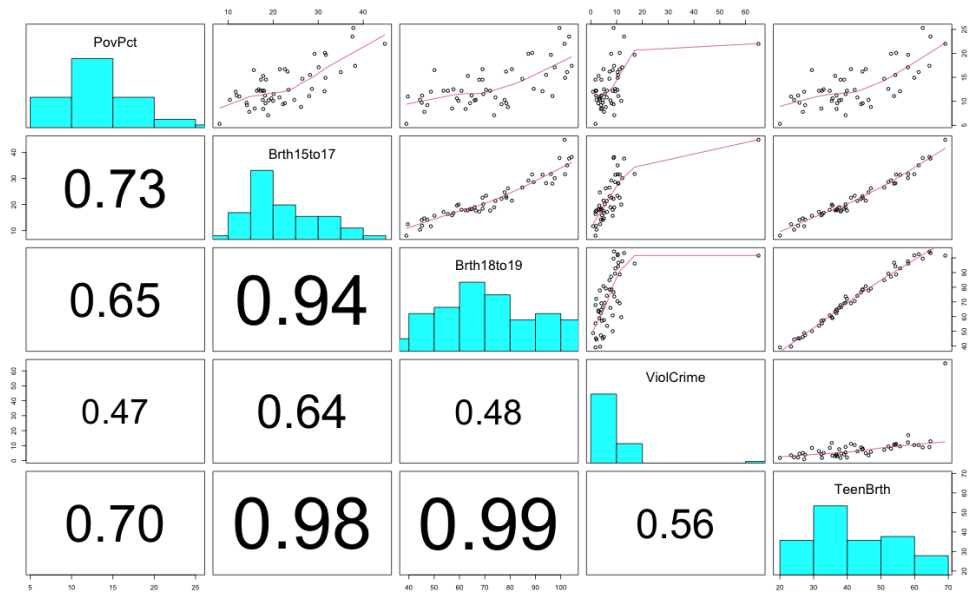
An example

We examine a data set about poverty and teen birth rate in US.

- Y = `PctPov`: percentage of each US state's population living under poverty condition
- X_1 = `Brth15to17`: birth rate for females 15-17 years old
- X_2 = `Brth18to19`: birth rate for females 18-19 years odd

```
poverty<-read.table(file.path(datasets.dir,
                              'index.txt'),
                    header = TRUE)
# glimpse(poverty)
round(cor(poverty[,c(2:4)]),2) # correlations
```

```
##             PovPct Brth15to17 Brth18to19
## PovPct        1.00       0.73       0.65
## Brth15to17    0.73       1.00       0.94
## Brth18to19    0.65       0.94       1.00
```



<!--]]]
Data Souced from *Mind on Statistics*, (3rd Edition), Utts and Heckard

-->

Y~X1

```
f1.lm <- lm(PovPct ~ Brth15to17,
            data=poverty)
summary(f1.lm)
```

```
##
## Call:
## lm(formula = PovPct ~ Brth15to17, data = poverty)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8390 -2.0532 -0.2401  2.4201  6.1562
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.46447    1.22799   3.636 0.000665 ***
## Brth15to17   0.38834    0.05189   7.483 1.19e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.952 on 49 degrees of freedom
## Multiple R-squared:  0.5333,    Adjusted R-squared:  0.5238
## F-statistic:    56 on 1 and 49 DF,  p-value: 1.188e-09
```

Y~X2

```
f2.lm <- lm(PovPct ~ Brth18to19,
            data=poverty)
summary(f2.lm)
```

```
##
## Call:
## lm(formula = PovPct ~ Brth18to19, data = poverty)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.2637 -2.4027 -0.4447  2.0789  8.1582
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.57124    1.82205   1.411    0.165
## Brth18to19   0.14644    0.02448   5.982  2.5e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.285 on 49 degrees of freedom
## Multiple R-squared:  0.4221,    Adjusted R-squared:  0.4103
## F-statistic: 35.78 on 1 and 49 DF,  p-value: 2.495e-07
```

Y~X1+X2

```
f3.lm <- lm(PovPct~Brth15to17+Brth18to19,
            data=poverty)
summary(f3.lm)
```

```
##
## Call:
## lm(formula = PovPct ~ Brth15to17 + Brth18to19, data = poverty)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.9659 -2.1686 -0.5964  2.4158  5.6097
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.21415    1.91423   3.246 0.002135 **
## Brth15to17   0.56141    0.15456   3.632 0.000682 ***
## Brth18to19  -0.07784    0.06552  -1.188 0.240654
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.939 on 48 degrees of freedom
## Multiple R-squared:  0.5467,    Adjusted R-squared:  0.5278
## F-statistic: 28.94 on 2 and 48 DF,  p-value: 5.678e-09
```

Observations:

- $\hat{\beta}$ value is different in multiple regression `f3.lm` as it is in the relevant simple regression, `f1.lm` and `f2.lm`.
- The R^2 for `f3.lm` is not even close to the sum of R^2 for two simple regression models. Adding `Brth18to19` does not make independent 'add-on' contribution to `f3.lm`.
- When `Brth15to17` is present in the model, `Brth18to19` seems to have a diminished effect and does not improve R^2 as much.
- The correlation between `Brth15to17` and `Brth18to19` have increased the standard errors of coefficients. In other words, the estimate of individual slopes are less precise.

Ridge Regression

In relation to Lasso, we can also use a penalty based on *squared* coefficients

$$nlog(RSS) + \lambda \sum_{i=1}^{p} |\beta_i|^2 = nlog(RSS) + \lambda ||\beta||_2.$$

This estimator is called **ridge regression**.

- This is another 'popular' regularised (or penalised) regression technique.
- The squared coefficients $|\beta_i|^2$ makes the penalty term even *smoother*.
- Such penalty term is often called a l_2 penalty.

A constrained minimsation view

Ridge regression, $\hat{\beta}_{ridge}$ is chosen to minimise the penalised RSS,

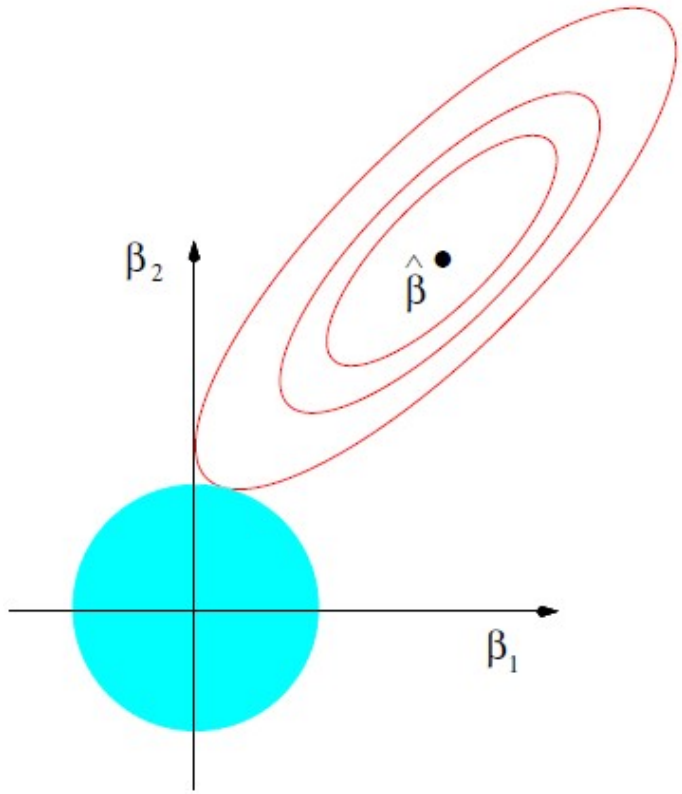
$$nlog(RSS) + \lambda||\beta||_2$$

This is equivalent to the minimisation of $nlog(RSS)$ subject to the *constraint* $||\beta||_2 = \sum |\beta_i|^2 < c(\lambda)$, for some $c(\lambda) > 0$.

- Ridge regression puts further constraints on parameter β_i's.
- Larger β_i's would take heavier penalty due to the l_2 operation.

Geometric Interpretation

In 2D space, i.e. $P = 2$, the l_2 penalty of ridge means the constraint region is a disk $\beta_1^2 + \beta_2^2 \leq c$.



- We are trying to minimise the ellipse size (contour of RSS) and disk (constraint set) simultaneously. The ridge estimate is marked by the point when the ellipse and the disk touches.
- Unlike the diamond shape for Lasso, with a disk region, it is less likely for the ellipses to 'touch' the axis exactly. So ridge regression performs *no variable selection* as few/no elements of $\hat{\beta}_{ridge}$ will be exactly zero.
- Ridge regression does **proportional shrinkage** — it shrinks *all* βs towards zero proportional to the amount of information each X holds.

Ridge Regression Estimator

Assume a *standardised model*, i.e. all Xs have mean 0 and variance 1, response Y also has 0 mean. Such (regression) model will have no intercept. We then can find the ridge estimator* by linear algebra (i.e. set derivative w.r.t. β = 0 and solve for β).

$$\hat{\beta}_{\lambda_{ridge}} = (X^T X + \lambda I)^{-1} X^T Y.$$

Compare to the ordinary least square (OLS) estimator

$$\hat{\beta}_{LS} = (X^T X)^{-1} X^T Y$$

Essentially, ridge regression estimator **adds a (small) value, λ,** to the *diagonal* elements of the matrix $X^T X$.

- $X^T X$ for a standardised model has ns on its diagonal.
- $X^T X + \lambda I$ then has $n + \lambda$ on its diagonal, but the same values off diagonal.

[*] This is where ridge regression gets its name since the diagonal of a symmetric (correlation) matrix may be thought of as a 'ridge'.

Why is this addition of λ to X^TX useful?

- The solution (to the matrix inverse operation) exists for all $\lambda > 0$ even when X^TX is singular — something OLS will definitely fail.

- When X^TX is nearly singular — OLS may fail too. Adding λ to the diagonal while keeping off diagonal the same makes variates less co-linear, so $\hat{\beta}_{ridge}$ regression is more robust against **multicollinearity**. Correlations between Xs have less impact on $\hat{\beta}_{ridge}$.

```
set.seed(123)
X <- matrix(rnorm(12), 3)
X.s <- apply(X, 2, scale) # standardised X
round(cor(X.s),2)  # correlation matrix
```

```
##       [,1]  [,2]  [,3]  [,4]
## [1,]  1.00  0.99 -0.33  0.12
## [2,]  0.99  1.00 -0.22  0.01
## [3,] -0.33 -0.22  1.00 -0.98
## [4,]  0.12  0.01 -0.98  1.00
```

```
ss <- t(X.s) %*% X.s # matrix X.transpose * X
lambda <- 1 # set lambda
diag(ss) <- diag(ss) + lambda # adding lambda
round(cov2cor(ss),2)  # reduced correlations
```

```
##       [,1]  [,2]  [,3]  [,4]
## [1,]  1.00  0.66 -0.22  0.08
## [2,]  0.66  1.00 -0.15  0.01
## [3,] -0.22 -0.15  1.00 -0.65
## [4,]  0.08  0.01 -0.65  1.00
```

Linear Algebra Trick (Not examinable)

Given a standardised matrix of X, the Singular Value Decomposition (SVD) gives

$$X = UDV^T$$

- $U = (u_1, u_2, \ldots, u_p)$ is a $N \times p$ orthogonal matrix. Each u_i has mean zero and length one and is uncorrelated with each other.
- $V = (v_1, v_2, \ldots, v_p)$ is a $p \times p$ orthogonal matrix. Each v_i has mean zero and length one and is uncorrelated with each other.
- $D = diag(d_1, d_2, \ldots, d_p)$, where $d_1 \geq d_2 \geq \ldots \geq d_p \geq 0$ are the eigenvalues of X.

The *principle component* (PC) of X are $z_i = d_i u_i, i = 1, 2, \ldots, p$.

- The first PC of X, z_1, has the largest sample variance

$$Var(z_1) = d_1^2/N$$

- Subsequent PCs z_j, $j > 1$ have (max) variance d_j^2/N, and is orthogonal to all earlier PCs.

What does it mean for ridge regression?

Apply $X = UDV^T$ to X^TX, this gives

$$\begin{aligned} X^TX &= (UDV^T)^T(UDV^T) \\ &= VDU^TUDV^T \\ &= VD^2V^T \end{aligned}$$

Consider a standardised X, the ridge fitted value is

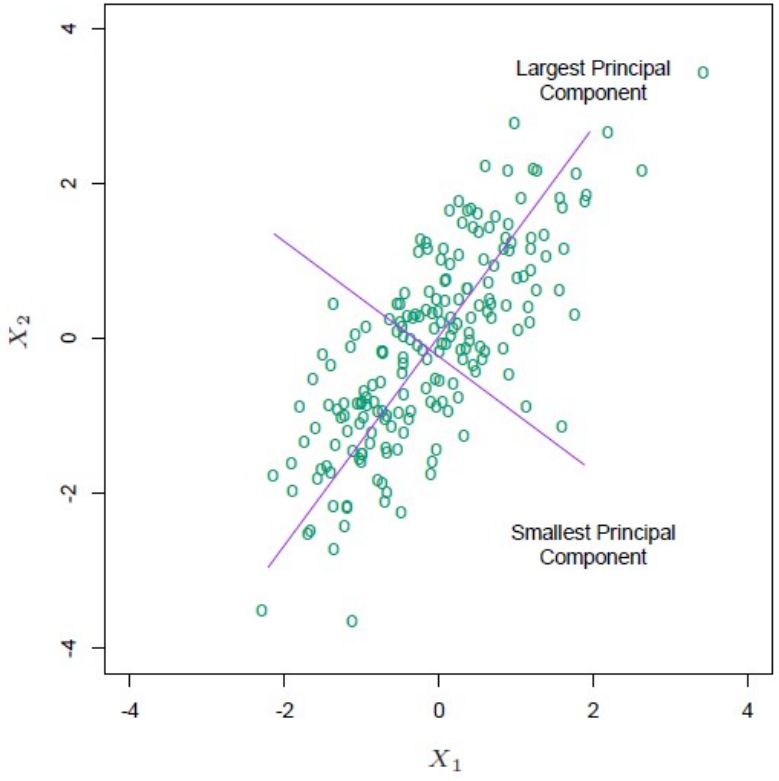
$$\begin{aligned} \hat{y}_{ridge} &= X\hat{\beta}_{ridge} \\ &= X(X^TX + \lambda I)^{-1}X^Ty \\ &= UDV^T(VD^2V^T + \lambda I)^{-1}(UDV^T)^Ty \\ &= UD(D^2 + \lambda I)^{-1}DU^Ty \\ &= \sum_1^p u_i \frac{d_i^2}{d_i^2 + \lambda} u_i^Ty \end{aligned}$$

where u_i is the standardised PC of X.

So the ridge estimate

$$\hat{\beta}_{i_{ridge}} = \frac{d_i^2}{d_i^2 + \lambda} u_i^T y$$

- Instead of using input X directly, ridge regression shrinks the coefficient w.r.t. **principle components**.
- The shrinking factor is $\frac{d_i^2}{d_i^2+\lambda}$. The larger the λ, the more shrinkage in the direction of u_i.
- Ridge regression shrinks the coefficients of the low variance components (i.e. with low information) more than the high-variance components.



So the ridge estimate

$$\hat{\beta}_{i_{ridge}} = \frac{d_i^2}{d_i^2 + \lambda} u_i^T y$$

- Instead of using input X directly, ridge regression shrinks the coefficient w.r.t. **principle components**.
- The shrinking factor is $\frac{d_i^2}{d_i^2+\lambda}$. The larger the λ, the more shrinkage in the direction of u_i.
- Ridge regression shrinks the coefficients of the low variance components (i.e. with low information) more than the high-variance components.

A numerical example:

Assuming that X contains two variables and penalty $\lambda = 1$. SVD gives 2 PCs with $d_1 = 10$ and $d_2 = 2$.

- Then the ridge shrinkage on $\hat{\beta}_1$ is $\frac{d_1^2}{d_1^2+\lambda} = \frac{10^2}{10^2+1} = 0.99$
- The shrinkage on $\hat{\beta}_2$ is $\frac{d_2^2}{d_2^2+\lambda} = \frac{2^2}{2^2+1} = 0.8$
- The shrinkage on more 'informative' PC1 is 1% (1-0.99) compared to that of 20% on (less variable) PC2.

Choosing λ

As usual, we choose the hyper-parameter λ by cross-validation or validation using a separate data set.

Computationally, ridge regression is much faster than subset selection, but it would be much slower than the Lasso if we use

$$\hat{\beta}_\lambda = (X^T X + \lambda I)^{-1} X^T Y$$

separately for every single λ — each needs $O(nP + P^3)$ time.

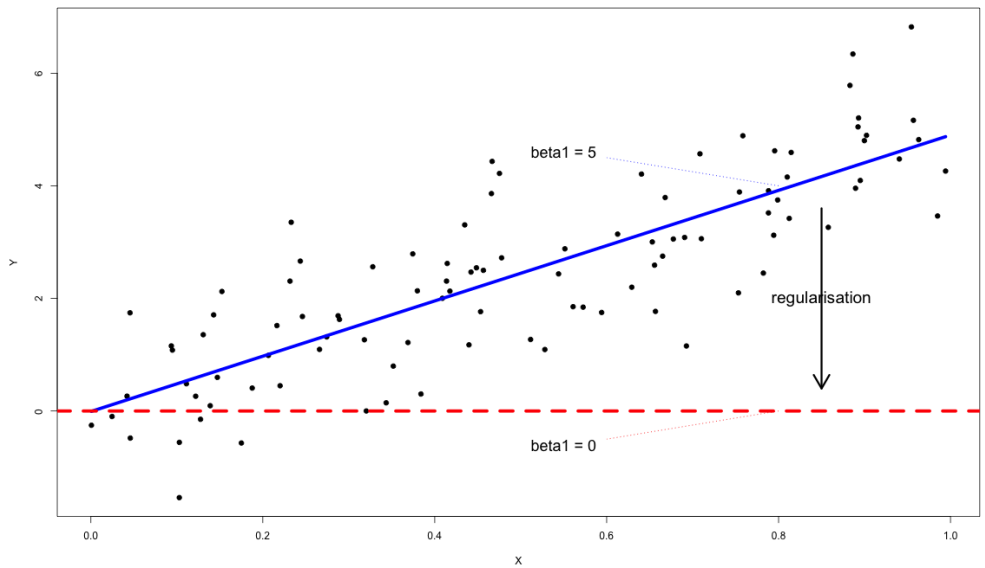
Why LASSO can do variable selection, but Ridge can't?

We use a simulated data sets and plots to illustrate how does $Penalised\ RSS$ behave w.r.t. β for different λs.

```
set.seed(123)
X <- runif(100)
Y <- X * 5 + rnorm(100)

fit.lm <- lm(Y~X)
fit.lm$coefficients
```

```
##  (Intercept)            X
## -0.008959851  4.910168641
```

```
b0 <- fit.lm$coefficients[1] # intercept
b1 <- fit.lm$coefficients[2] # slope
```



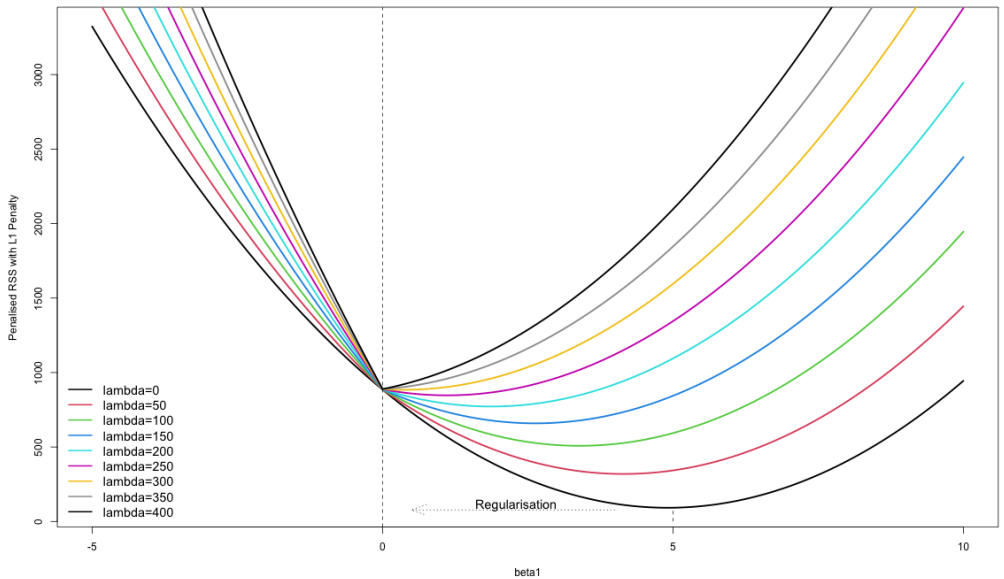
Compute $Penalised\ RSS(\lambda)$

```
# set up
betas.vt <- seq(-5, 10, by=0.1)
ii <- which(betas.vt==5)  # index for OLS beta1 (= 5)
lambdas <- seq(0,400, by=50)  # penalty coefficients

# L1 penalised RSS
pen_L1.fn <- function(beta,l){
  sum((b0 + X * beta - Y)^2) + l * (abs(beta) + abs(b0))
}

# L2 penalised RSS
pen_L2.fn <- function(beta,l){
  sum((b0 + X * beta - Y)^2) + l * (beta^2 + b0^2)
}

# lambda = 0 ==> OLS (no L1 penalty)
p1.vt <- betas.vt %>% map_dbl(~pen_L1.fn(., l=lambdas[1]))

# lambda = 0 ==> OLS (no L2 penalty)
p2.vt <- betas.vt %>% map_dbl(~pen_L2.fn(., l=lambdas[1]))

all(p1.vt == p2.vt) # TRUE as both are OLS (i.e. lambda = 0)
```

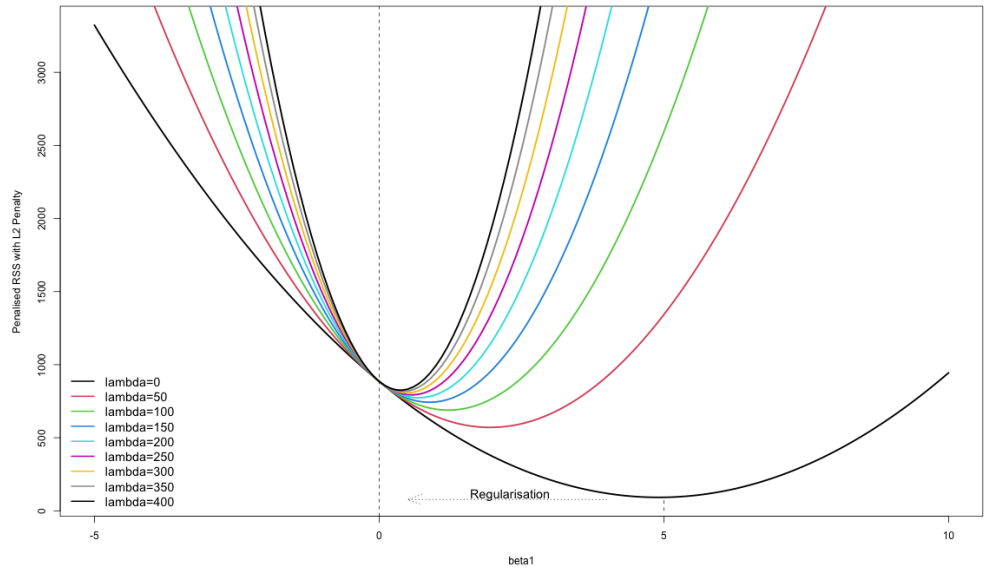
```
## [1] TRUE
```

Visualise $Penalised\ RSS(\lambda)$

LASSO Regression with l_1 penalty



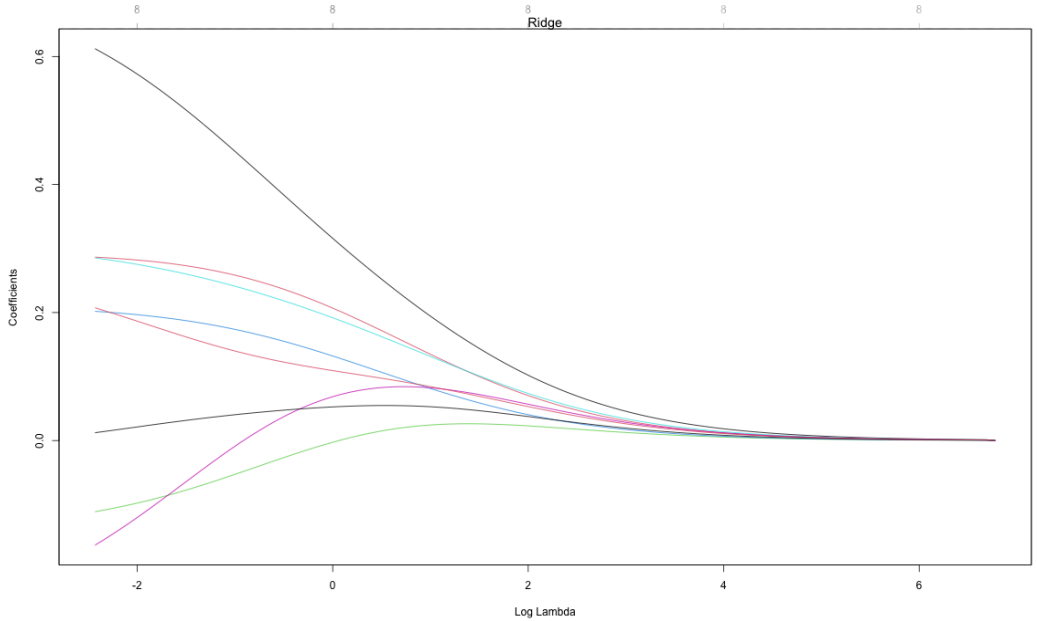
Ridge Regression with l_2 penalty

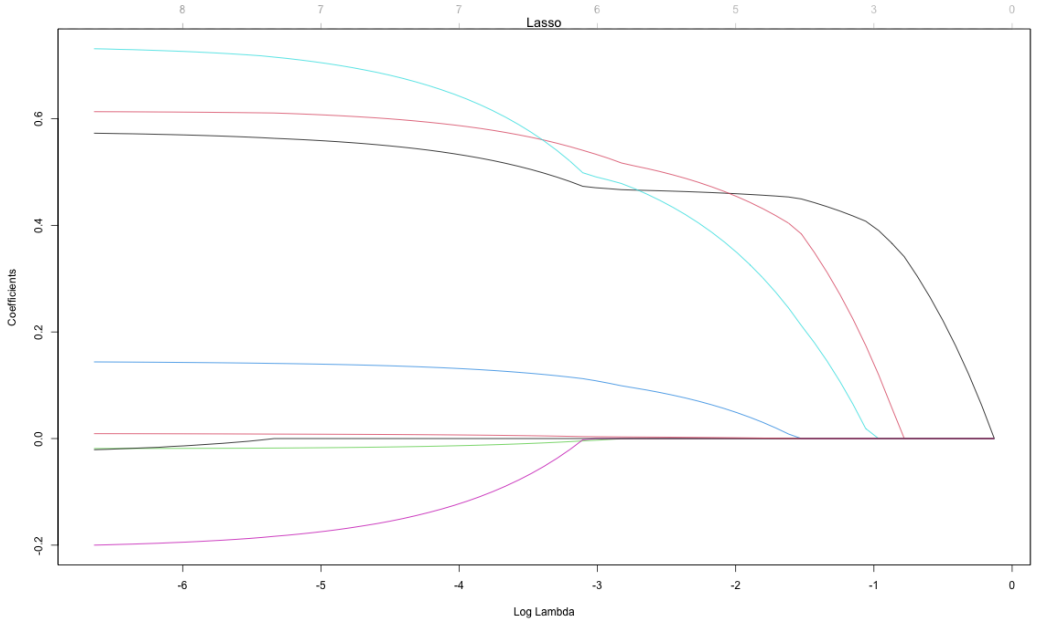


Comment:

- l_1 penalty exists a *'sharp' twist* at $\beta_1 = 0$. This *becomes* the minimum of $Penalised\ RSS$ as λ increases.
- l_2 penalty is always smooth and the minimum of $Penalised\ RSS$ *approaches to* zero as λ increases.
- Hence, LASSO can do variable selection (i.e. have a $\beta = 0$), but ridge can't.

Paths of Shrinkage





Subset selection, LASSO and Ridge regression

There is no general rules about which one is better.

- AIC-type subset selection: l_0 penalty; basically drop all variables with βs smaller than the M^{th} largest — a form of *hard thresholding*.

- Lasso regression: l_1 penalty; shrinks each β by a constant factor λ, truncating at zero — a form of *soft thresholding*. Can do **variable selection**.

- Ridge regression: l_2 penalty; *proportional shrinkage* and shrink all variables with more shrinkage along the low-variance principle components. Help dealing with **multicollinearity**.

These all regularise the ordinary least square (OLS) estimator to make it less **overfit**. The best prediction depends on whether the true best β looks like the output of subset selection, lasso or ridge regression.

Other variants

- Least Angle Regression (LAR)
  - closely related to Lasso, it is a 'boosting' method applied in regression.
  - provides a very efficient algorithms for computing the entire Lasso path.
- Elastic Net
  - penalty term

$$\lambda \sum_{1}^{p} (\alpha \beta_i^2 + (1-\alpha)|\beta_i|)$$

  - essentially a complement (and compromise) between ridge and lasso.
  - It select variables like the lasso, and shrinks all together the (correlated) variables like ridge.
- Others ...

Resources and references for regularisation

- Text book 'Introduction to Statistical Learning with Applications in R'
  - Section 2.2: The Bias-Variance Trade-off
  - Section 6.2: Shrinkage Methods
- The classical linear regression model is good. Why do we need regularization?
- 'Subset Selection in Regression' By Alan Miller (2nd, 2002)
  - Chapter 3 (section 3.10, 3.11, Appendix A): Finding subsets which fit well.
- *R for Statistical Learning* By David Dalpiaz
  - Section 24.1: Ridge Regression
  - Section 24.2: Lasso
- Penalized Regression Essentials: Ridge, Lasso & Elastic Net

Regularisation Regression Examples

STATS369

Course develped by Thomas Lumley (Department of Statistics), with additional teaching and work by Lisa Chen (Statistics), David Welch (Computer Science), Yalu Wen (Statistics)
Faculty of Science, University of Auckland

Plan for this week

Regularised regression

- [L13] Motivation & Overview 👍
- [L14] Regularised regression 👍
    - LASSO regression
    - Ridge regression
- [L15] Examples

Examples

Two examples in separate HTML files on Canvas...

1: the 'mushrooms' data

W05_ex_mushrooms_lasso.html
W05_ex_mushrooms_ridge.html

2: the 'marathons' data

W05_ex_marathons.html
(data uploaded on Canvas as well.)