Linear Regression
Linear Regression with Shrinkage

Some slides are due to Tommi Jaakkola, MIT AI Lab

Introduction

- The goal of regression is to make quantitative (real valued) predictions on the basis of a (vector of) features or attributes.
- Examples: house prices, stock values, survival time, fuel efficiency of cars, etc.

Predicting vehicle fuel efficiency (mpg) from 8 attributes:

y	x				
	cyls	disp	hp	weight	. . .
18.0	8	307.0	130.00	3504	. . .
26.0	4	97.00	46.00	1835	. . .
33.5	4	98.00	83.00	2075	. . .
. . .					

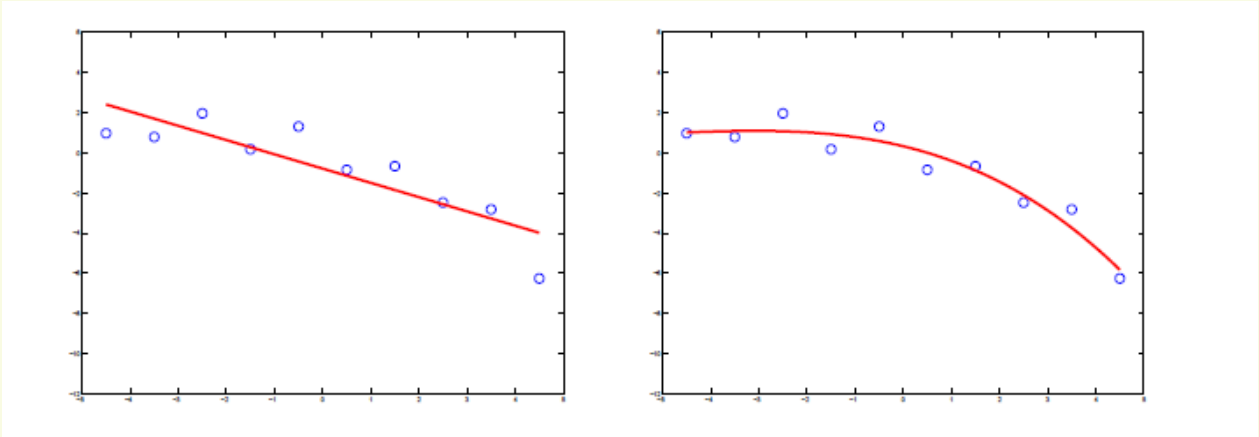
A generic regression problem

- The input attributes are given as fixed length vectors $\mathrm{x} \in \Re^M$ that could come from different sources: inputs, transformation of inputs (log, square root, etc…) or basis functions.
- The outputs are assumed to be real valued $\mathrm{y} \in \Re$ (with some possible restrictions).
- Given *n* iid training samples $\mathrm{D} = \{(\mathrm{x}_1, y_1)...(\mathrm{x}_n, y_n)\}$ from unknown distribution $\mathrm{P}(\mathrm{x}, y)$, the goal is to minimize the prediction error (loss) on new examples (x, y) drawn at random from the same $\mathrm{P}(\mathrm{x}, y)$.
- An a example of a loss function:

$$L(f(\mathrm{x}), y) = (f(\mathrm{x}) - y)^2$$ squared loss

our prediction for x

Regression Function



- We need to define a class of functions (types of predictions we make).

Linear prediction:

$$f(x; w_0, w_1) = w_0 + w_1 x$$

where w_0, w_1 are the parameters we need estimate.

Linear Regression

Typically we have a set of training data $(x_1, y_1)...(x_n, y_n)$ from which we estimate the parameters w_0, w_1 .

Each $x_i = (x_{i1},.., x_{iM})$ is a vector of measurements for ith case.

or

$h(x_i) = \{h_0(x_i),..., h_M(x_i)\}$ a basis expansion of x_i

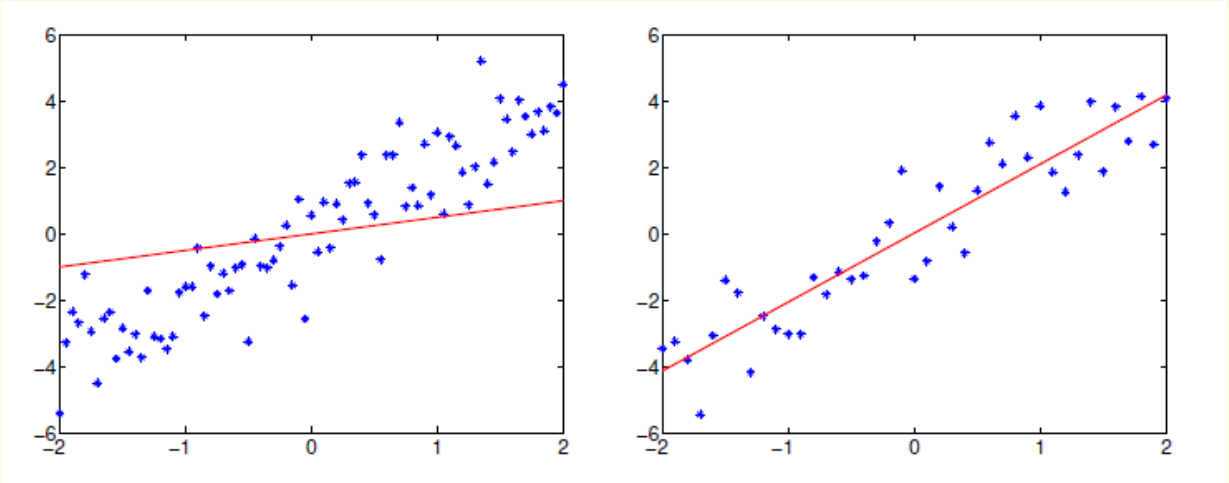
$$y(x) \approx f(x; w) = w_0 + \sum_{j=1}^{M} w_j h_j(x) = w^t h(x)$$

(define $h_0(x) = 1$)

Basis Functions

- There are many basis functions we can use e.g.
  - Polynomial $h_j(x) = x^{j-1}$
  - Radial basis functions $h_j(x) = \exp\left(-\frac{(x-\mu_j)^2}{2s^2}\right)$
  - Sigmoidal $h_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$
  - Splines, Fourier, Wavelets, etc

Estimation Criterion



- We need a fitting/estimation criterion to select appropriate values for the parameters w_0, w_1 based on the training set $D = \{(x_1, y_1)...(x_n, y_n)\}$
- For example, we can use the empirical loss:

$$J_n(w_1, w_0) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(x_i; w_1, w_0)\right)^2$$

(note: the loss is the same as in evaluation)

Empirical loss: motivation

- Ideally, we would like to find the parameters w_0, w_1 , that minimize the expected loss (unlimited training data):

$$J(w_0, w_1) = E_{(x,y)\sim P}\left(y_i - f(x_i; w_0, w_1)\right)^2$$

where the expectation is over samples from $P(x,y)$.

- When the number of training examples n is large:

$$\underbrace{E_{(x,y)\sim P}\left(y_i - f(x_i; w_0, w_1)\right)^2}_{\text{Expected loss}} \approx \underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(x_i; w_0, w_1)\right)^2}_{\text{Empirical loss}}$$

Linear Regression Estimation

- Minimize the empirical squared loss

$$J_n(w_0, w_1) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(x_i; w_0, w_1)\right)^2$$

$$= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - w_0 - w_1 x_i\right)^2$$

- By setting the derivatives with respect to w_1, w_0 to zero we get necessary conditions for the "optimal" parameter values.

$$\frac{\partial}{\partial w_1} J_n(w_0, w_1) = \frac{2}{n}\sum_{i=1}^{n}\left(y_i - w_0 - w_1 x_i\right)(-x_i) = 0$$

$$\frac{\partial}{\partial w_0} J_n(w_0, w_1) = \frac{2}{n}\sum_{i=1}^{n}\left(y_i - w_0 - w_1 x_i\right)(-1) = 0$$

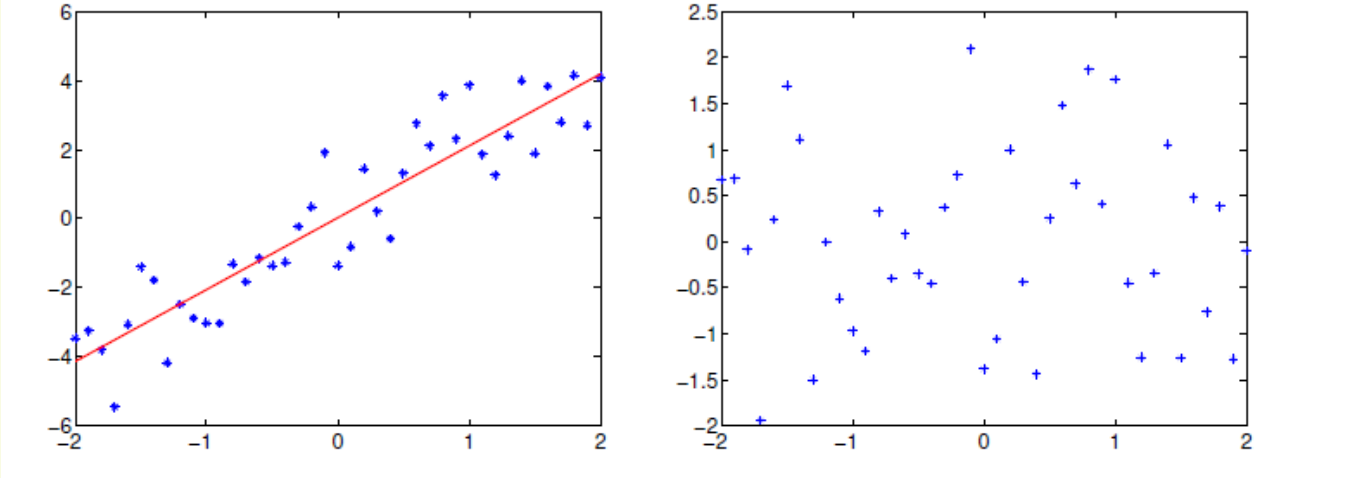
Interpretation

The optimality conditions

$$\frac{2}{n}\sum_{i=1}^{n}\left(y_i - w_0 - w_1 x_i\right)(-x_i) = 0$$

$$\frac{2}{n}\sum_{i=1}^{n}\left(y_i - w_0 - w_1 x_i\right)(-1) = 0$$

ensure that the prediction error $\varepsilon_i = \left(y - w_0 - w_1 x_i\right)$ is decorrelated with any linear function of the inputs.



Linear Regression: Matrix Form

$$\mathrm{X} = \begin{bmatrix} 1 & -x_1- \\ \dots & \dots \\ 1 & -x_n- \end{bmatrix} \begin{matrix} \text{n samples} \\ \\ \\ \text{M+1} \end{matrix} \qquad \mathrm{w} = \begin{bmatrix} w_0 \\ | \\ w_1 \\ | \end{bmatrix} \text{M+1} \qquad \mathrm{y} = \begin{bmatrix} y_1 \\ \dots \\ y_n \end{bmatrix}$$

$$J_n(w_0, w_1) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - w_0 - w_1 x_i \right)^2 = \left(\mathrm{Xw} - \mathrm{y} \right)^t \left(\mathrm{Xw} - \mathrm{y} \right)$$

Linear Regression Solution

- By setting the derivatives of $(\mathrm{Xw}-\mathrm{y})^t(\mathrm{Xw}-\mathrm{y})$ to zero,

$$\frac{2}{n}\left(\mathrm{X}^{\mathrm{t}}\mathrm{y}-\mathrm{X}^{\mathrm{t}}\mathrm{Xw}\right)=0$$

we get the solution:

$$\hat{w}=\left(\mathrm{X}^{\mathrm{t}}\mathrm{X}\right)^{-1}\mathrm{X}^{\mathrm{t}}\mathrm{y}$$

The solution is a linear function of the outputs y.

Statistical view of linear regression

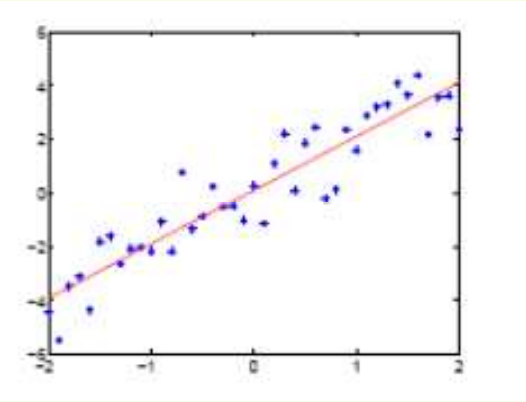
- In a statistical regression model we model both the function and noise

Observed output = function + noise

$$y(x) = f(x; w) + \varepsilon$$

where, e.g., $\varepsilon \sim N(0, \sigma^2)$

- Whatever we cannot capture with our chosen family of functions will be interpreted as noise

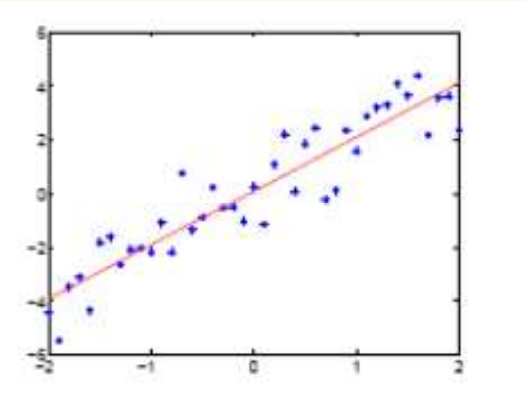


Statistical view of linear regression

- $f(x;w)$ is trying to capture the mean of the observations y given the input x:

$$E[y \mid x] = E[f(x;w) + \varepsilon \mid x] = f(x;w)$$

- where $E[y \mid x]$ is the conditional expectation of y given x, evaluated according to the model (not according to the underlying distribution of X)



Statistical view of linear regression

- According to our statistical model

$$y(x) = f(x;w) + \varepsilon, \;\; \varepsilon \sim N(0, \sigma^2)$$

the outputs y given x are normally distributed with mean $f(x;w)$ and variance σ^2:

$$p(y \mid x, w, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2\sigma^2} (y - f(x;w))^2 \right]$$

(we model the uncertainty in the predictions, not just the mean)

Maximum likelihood estimation

- Given observations $D = \{(x_1, y_1), ..., (x_n, y_n)\}$ we find the parameters w that maximize the likelihood of the outputs:

$$L(w, \sigma^2) = \prod_{i=1}^{n} p(y_i \mid x_i, w, \sigma^2)$$

$$= \left(\frac{1}{\sqrt{2\pi\sigma^2}} \right)^n \exp\left\{ -\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(y_k - f\left(x_k ; w \right) \right)^2 \right\}$$

- Maximize log-likelihood

$$\log L(w, \sigma^2) = \log \left(\frac{1}{\sqrt{2\pi\sigma^2}} \right)^n - \left(\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(y_k - f\left(x_k ; w \right) \right)^2 \right)$$

minimize

Maximum likelihood estimation

- Thus

$$w_{MLE} = \arg\min_{w} \sum_{i=1}^{n} \left(y_i - f\left(x_i; w\right)\right)^2$$

- But the empirical squared loss is

$$J_n(w) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - f\left(x_i; w\right)\right)^2$$

Least-squares Linear Regression is MLE for Gaussian noise !!!

Linear Regression (is it "good"?)

- Simple model
- Straightforward solution

BUT

- **MLS is not a good estimator for prediction error**
- **The matrix $X^T X$ could be ill conditioned**
  - **Inputs are correlated**
  - **Input dimension is large**
  - **Training set is small**

Linear Regression - Example

In this example the output is generated by the model (where ε is a small white Gaussian noise):

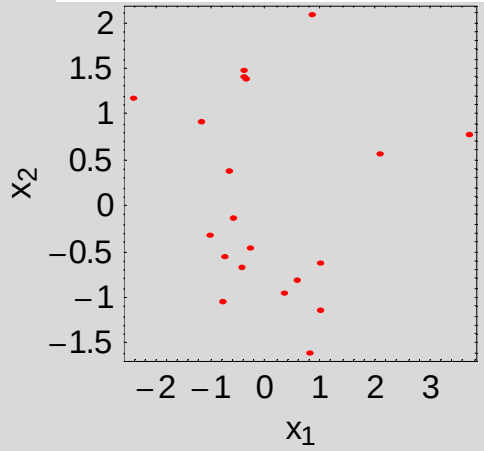
$$y = 0.8x_1 + 0.2x_2 + \varepsilon$$

Three training sets with different correlations between the two inputs were randomly chosen, and the linear regression solution was applied.

Linear Regression – Example

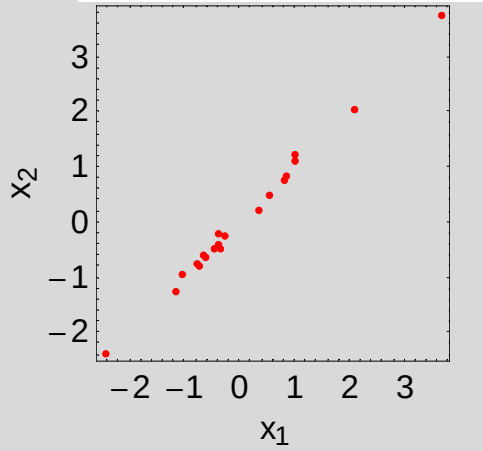
And the results are…

RSS=0.29
w=0.82, 0.209



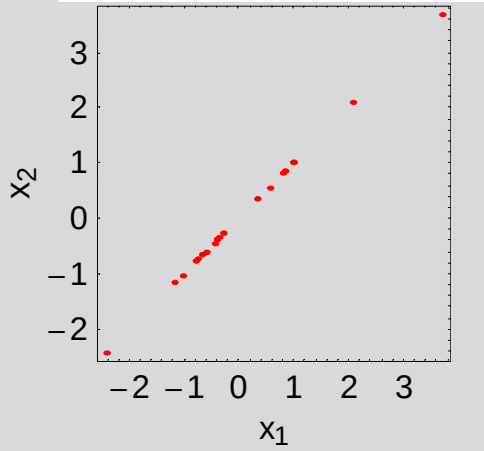
x_1,x_2 uncorrelated

RSS=0.204
w=0.42, 0.56



x_1,x_2 correlated

RSS=0.47
w=30.36, -29.21



x_1,x_2 strongly correlated

Strong correlation can cause the coefficients to be very large, and they will cancel each other to achieve a good *RSS*.

Linear Regression

What can be done?

Shrinkage methods

- Ridge Regression
- Lasso
- PCR (Principal Components Regression)
- PLS (Partial Least Squares)

Shrinkage methods

Before we proceed:

- Since the following methods are not invariant under input scale, we will assume the input is normalized (mean 0, variance 1):

$$x'_{ij} \leftarrow x_{ij} - \bar{x}_j \qquad\qquad x_{ij} \leftarrow \frac{x'_{ij}}{\sigma_j}$$

- The offset w_0 is always estimated as $w_0 = (\sum_i y_i) / N$ and we will work with centered y (meaning $y \leftarrow y\text{-}w_0$)

Ridge Regression

- Ridge regression shrinks the regression coefficients by imposing a penalty on their size (also called weight decay)
- In ridge regression, we add a quadratic penalty on the weights:

$$J(w) = \sum_{i=1}^{N} \left(y_i - w_0 - \sum_{j=1}^{M} x_{ij} w_j \right)^2 + \lambda \sum_{j=1}^{M} w_j^2$$

where $\lambda \geq 0$ is a tuning parameter that controls the amount of shrinkage.

- The size constraint prevents the phenomenon of wildly large coefficients that cancel each other from occurring.

Ridge Regression Solution

- Ridge regression in matrix form:

$$J(w) = (y - Xw)^t (y - Xw) + \lambda w^t w$$

- The solution is

$$\hat{w}^{ridge} = (X^t X + \lambda I_M)^{-1} X^t y$$

$$\hat{w}^{LS} = (\mathrm{X}^t \mathrm{X})^{-1} \mathrm{X}^t y$$

- The solution adds a positive constant to the diagonal of $X^T X$ before inversion. This makes the problem non singular even if X does not have full column rank.
- For orthogonal inputs the ridge estimates are the scaled version of least squares estimates:

$$\hat{w}^{ridge} = \gamma \, \hat{w}^{LS} \quad 0 \le \gamma \le 1$$

Ridge Regression (insights)

The matrix X can be represented by it's SVD:

$$X = UDV^T$$

- U is a $\mathrm{N}*\mathrm{M}$ matrix, it's columns span the column space of X
- D is a $\mathrm{M}*\mathrm{M}$ diagonal matrix of singular values
- V is a $\mathrm{M}*\mathrm{M}$ matrix, it's columns span the row space of X

Lets see how this method looks in the Principal Components coordinates of X

Ridge Regression (insights)

$$X' = XV$$

$$Xw = (XV)(V^T w) = X'w'$$

The least squares solution is given by: $\hat{w}^{ls} = (X^t X)^{-1} X^t y$

$$\hat{w}'^{ls} = V^T \hat{w}^{ls} = V^T \left(VDU^T UDV^T\right)^{-1} VDU^T y = D^{-1} U^T y$$

The Ridge Regression is similarly given by:

$$\hat{w}'^{ridge} = V^T \hat{w}^{ridge} = V^T \left(VDU^T UDV^T + \lambda I\right)^{-1} VDU^T y =$$

$$= \left(D^2 + \lambda I\right)^{-1} DU^T y = \left(D^2 + \lambda I\right)^{-1} D^2 \hat{w}'^{ls}$$

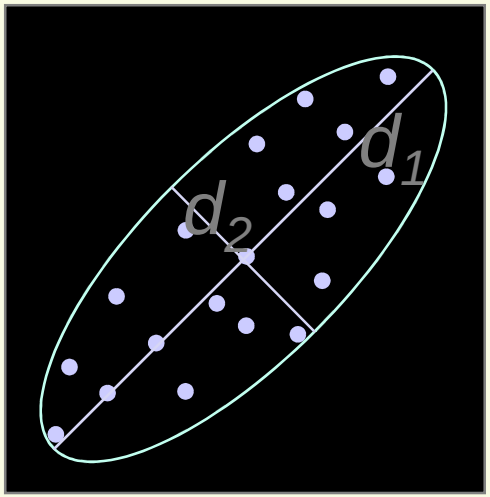
Diagonal

Ridge Regression (insights)

$$\hat{w}'^{ridge}_j = \frac{d_j^2}{d_j^2 + \lambda} \hat{w}'^{ls}_j$$

In the PCA axes, the ridge coefficients are just scaled LS coefficients!

The coefficients that correspond to smaller input variance directions are scaled down more.



Ridge Regression

In the following simulations, quantities are plotted versus the quantity

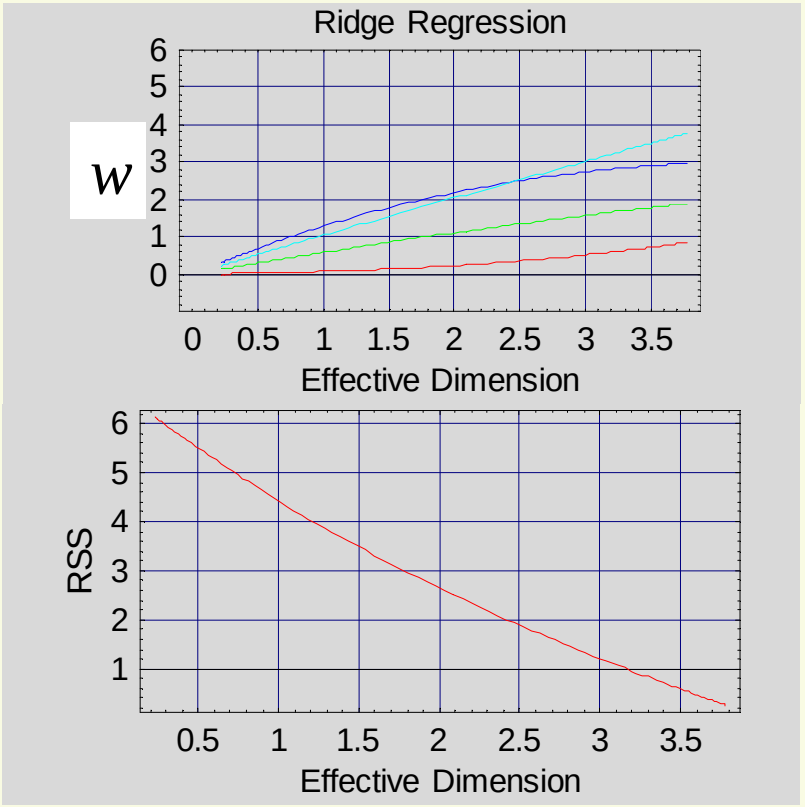
$$df(\lambda) = \sum_{j=1}^{M} \frac{d_j^2}{d_j^2 + \lambda}$$

This monotonic decreasing function is the *effective degrees of freedom* of the ridge regression fit.

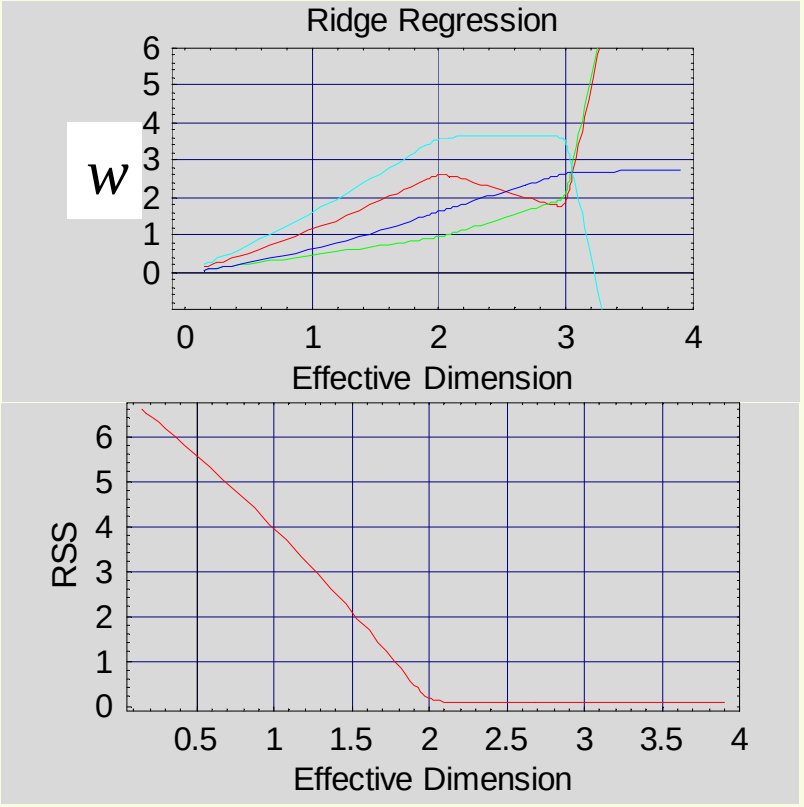
Ridge Regression

Simulation results: 4 dimensions (w=1,2,3,4)

No correlation



Strong correlation



Ridge regression is MAP with Gaussian prior

$$J(w) = -\log P(D \mid w) P(w)$$

$$= -\log\left[\prod_{i=1}^{n} N(y_i \mid w^t x_i, \sigma^2) N(w \mid 0, \tau^2)\right]$$

$$= \frac{1}{2\sigma^2}(y - Xw)^t (y - Xw) + \frac{1}{2\tau^2} w^t w + const$$

This is the same objective function that ridge solves, using $\lambda = \sigma^2 / \tau^2$

Ridge: $J(w) = (y - Xw)^t (y - Xw) + \lambda w^t w$

Lasso

Lasso is a shrinkage method like ridge, with a subtle but important difference. It is defined by

$$\hat{w}^{lasso} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \sum_{j=1}^{M} x_{ij} w_j \right)^2 \right\}$$

$$\text{subject to } \sum_{j=1}^{M} \left| w_j \right| \leq t$$

There is a similarity to the ridge regression problem: the L_2 ridge regression penalty is replaced by the L_1 lasso penalty.

The L_1 penalty makes the solution non-linear in y and requires a quadratic programming algorithm to compute it.

Lasso

If t is chosen to be larger then t_0: $t \geq t_0 \equiv \sum_{j=1}^{M} \left| \hat{w}^{ls} \right|$

then the lasso estimation is identical to the least squares. On the other hand, for say $t=t_0/2$, the least squares coefficients are shrunk by about 50% on average.

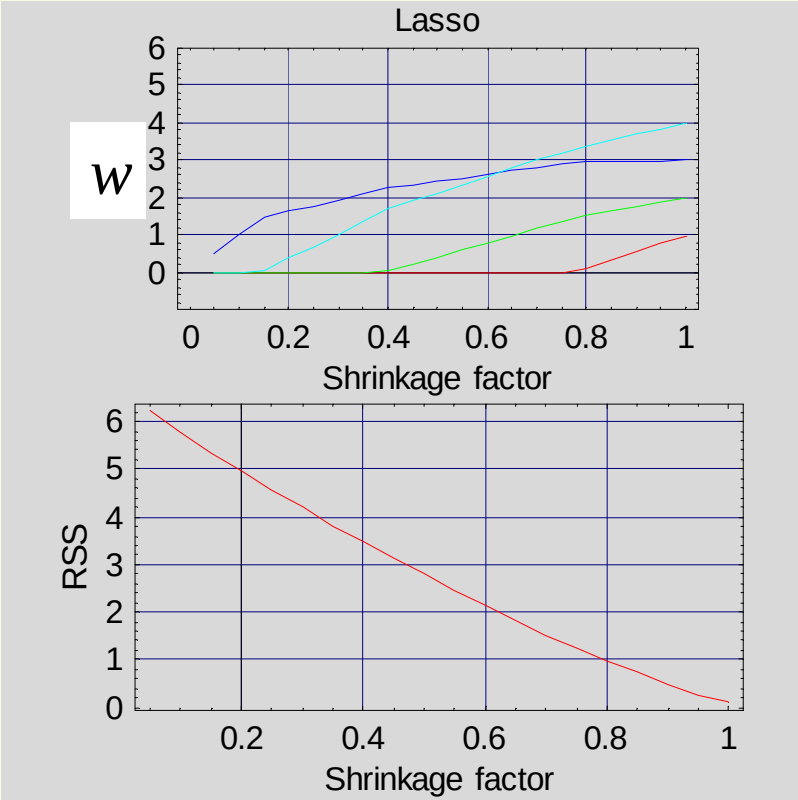
For convenience we will plot the simulation results versus shrinkage factor s:

$$s \equiv \frac{t}{t_0} = \frac{t}{\sum_{j=1}^{M} \left| \hat{w}^{ls} \right|}$$

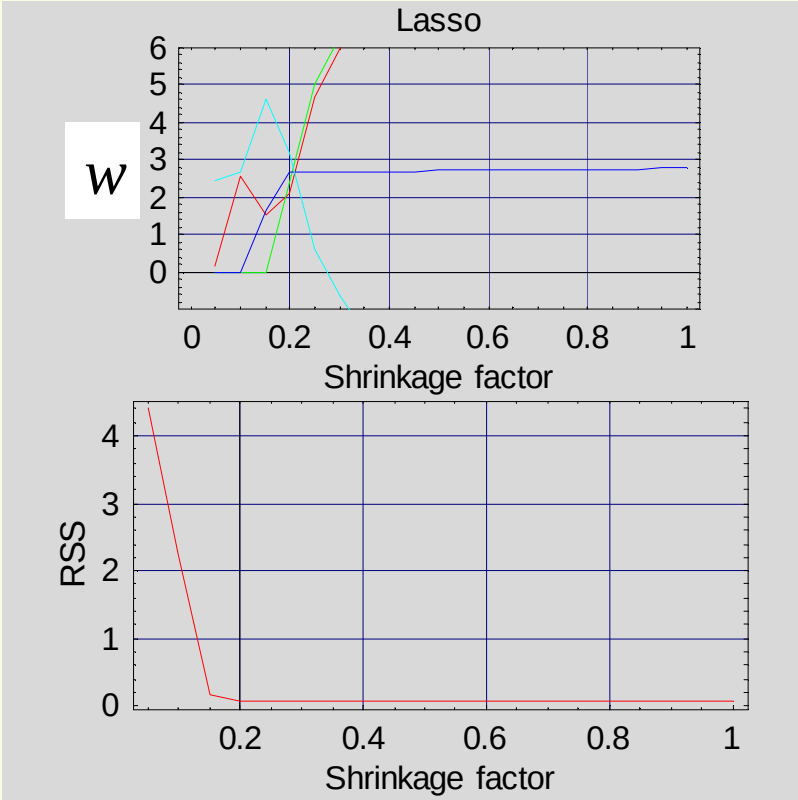
Lasso

Simulation results:

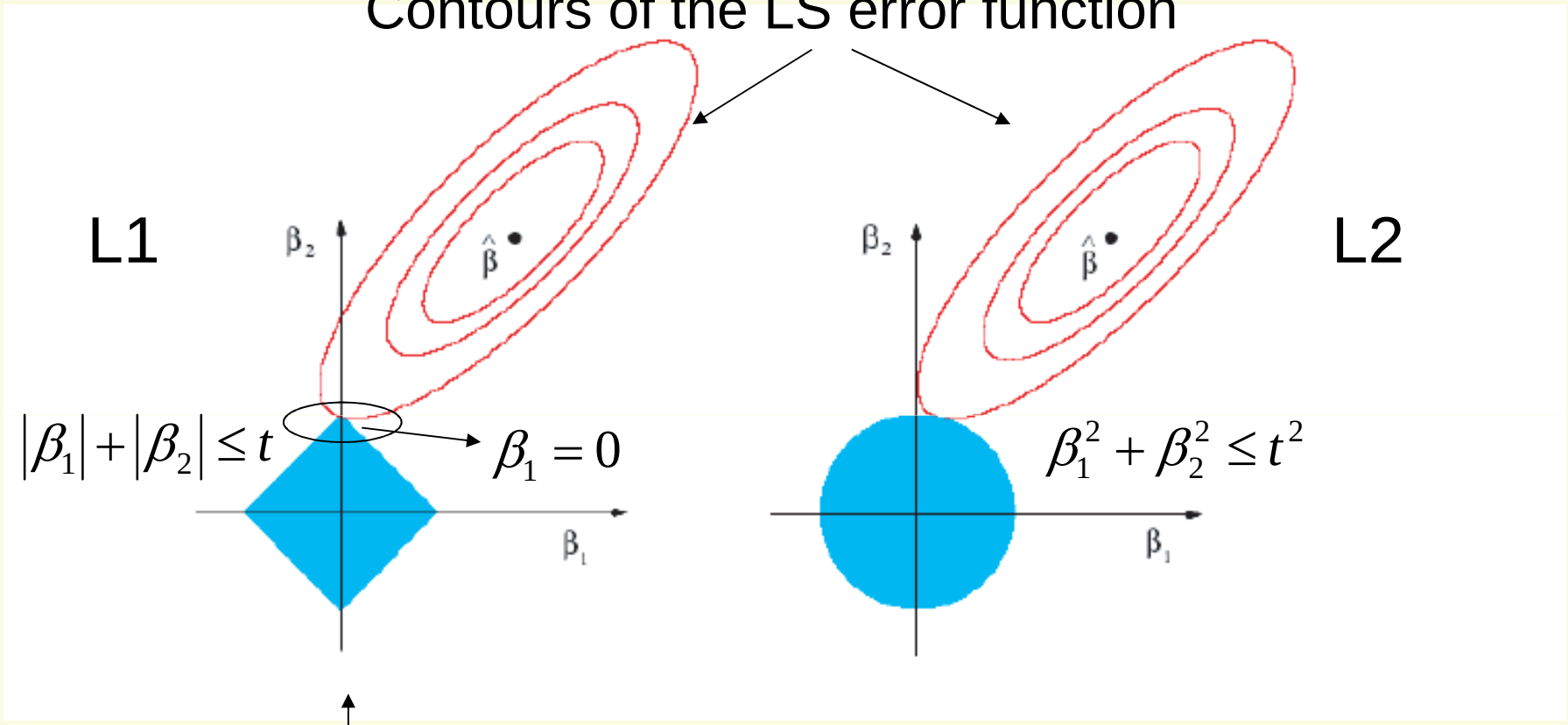
No correlation



Strong correlation



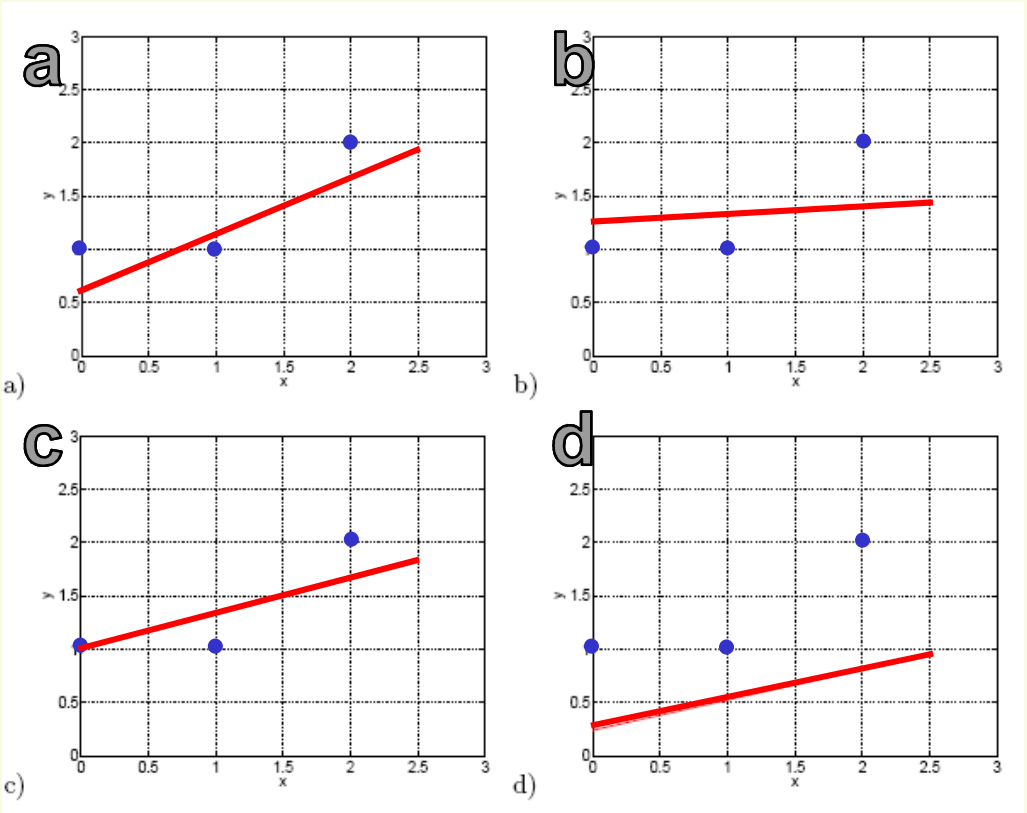
L2 vs L1 penalties



In Lasso the constraint region has corners; when the solution hits a corner the corresponding coefficients becomes 0 (when M>2 more than one).

Problem:

Figure 1 plots linear regression results on the basis of only three data points. We used various types of regularization to obtain the plots (see below) but got confused about which plot corresponds to which regularization method. Please assign each plot to one (and only one) of the following regularization method.



c

$$\sum_{t=1}^{3}(y_t - \theta x_t - \theta_0)^2 + \lambda\theta^2 \text{ where } \lambda = 1$$

b

$$\sum_{t=1}^{3}(y_t - \theta x_t - \theta_0)^2 + \lambda\theta^2 \text{ where } \lambda = 10$$

a

$$\sum_{t=1}^{3}(y_t - \theta x_t - \theta_0)^2 + \lambda(\theta^2 + \theta_0^2) \text{ where } \lambda = 1$$

d

$$\sum_{t=1}^{3}(y_t - \theta x_t - \theta_0)^2 + \lambda(\theta^2 + \theta_0^2) \text{ where } \lambda = 10$$