Machine Learning - Waseda University
Linear Regression

AD

June 2011

Regression

- Assume you are given some training data $\{\mathbf{x}_n, t_n\}_{n=1}^{N}$ where $\mathbf{x}_n \in \mathbb{R}^d$ and $t_n \in \mathbb{R}^c$.
- Given an input test data $\mathbf{x}$, you want to predict/estimate the output t associated to $\mathbf{x}$.
- Applications:
  - $\mathbf{x}$: location, t sensor reading.
  - $\mathbf{x}$: stock at time $k-d, k-d+1, ..., k-1$, t: stock at time k.
  - $\mathbf{x}$: facebook activity, number of friends etc., t: budget spent on entertainment.

Formalization

- We want to learn a mapping/predictor based on $\left\{\mathbf{x}_n, t_n\right\}_{n=1}^{N}$:

$$\widehat{t}\left(\mathbf{x}\right) : \mathbb{R}^d \rightarrow \mathbb{R}^c$$

  which allows us to predict the response t given a new input $\mathbf{x}$.
- Linear regression is the simplest approach to build such a mapping and is ubiquitous in applied science.

Linear Regression: The Simplest Case

- For sake of simplicity, consider the simplest case $c = 1$ and $d = 1$.
- Linear regression assumes a model

$$\widehat{t}(x) = w_0 + w_1 x$$

  where $w_1, w_0 \in \mathbb{R}$.
- Given only 2 training data, we can solve for w_1 and w_0 but the result would be dependent of the 2 training data selected and very sensitive to the noise in the training responses t_n.
- A more sensible approach is to minimize the residual errors ε^i over the N training data

$$\begin{aligned} \epsilon_n &= t_n - \widehat{t}(x_n) \\ &= t_n - (w_0 + w_1 x_n) \end{aligned}$$

Least square Regression

- We select (w_0, w_1) as the coefficients minimizing the sum of the squared residual errors

$$E(w_0, w_1) = \sum_{n=1}^{N} \left(t_n - \widehat{t}(x_n)\right)^2$$

- Clearly as $N \longrightarrow \infty$

$$\lim_{N \to \infty} \frac{1}{N} E(w_0, w_1) = \int \left(t - \widehat{t}(x)\right)^2 p(x, t)\, dxdt$$

where $p(x, t)$ is the unknown density of the training data $\{x_n, t_n\}_{n=1}^{N}$.

Least square Regression

- Rewritten in vector-matrix form, we have

$$\underbrace{\mathbf{t}}_{N\times 1} = \widehat{\mathbf{t}} + \boldsymbol{\epsilon} = \underbrace{\boldsymbol{\phi}}_{N\times 2}\underbrace{\mathbf{w}}_{2\times 1} + \underbrace{\boldsymbol{\epsilon}}_{N\times 1}$$

  where $\mathbf{t} = (t_1, ..., t_N)^{\mathsf{T}}$, $\boldsymbol{\epsilon} = (\epsilon_1, ..., \epsilon_N)^{\mathsf{T}}$, $[\boldsymbol{\phi}]_{n,1} = 1$, $[\boldsymbol{\phi}]_{n,2} = x_n$ for $n = 1, ..., N$ and $\mathbf{w} = (w_0 \quad w_1)^{\mathsf{T}}$.

- With this notation, we have

$$E(w_0, w_1) = E(\mathbf{w}) = (\mathbf{t} - \boldsymbol{\phi}\mathbf{w})^{\mathsf{T}} (\mathbf{t} - \boldsymbol{\phi}\mathbf{w})$$

- We can easily minimize $E(\mathbf{w})$ w.r.t $\mathbf{w}$ to obtain

$$\mathbf{w}_{\mathsf{LS}} = \left(\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}\right)^{-1} \boldsymbol{\phi}^{\mathsf{T}}\mathbf{t}.$$

Linear Regression: A More General Case

- Consider now the case where $\mathbf{x} = \left(x^0, x^1, ..., x^{M-1}\right) \in \mathbb{R}^M$ where $x^0 = 1$ by convention (i.e. before we considered $M = 2$) and $t \in \mathbb{R}$ then we can use the model

$$\widehat{t}\left(\mathbf{x}\right) = \mathbf{w}^{\mathsf{T}}\mathbf{x} = w_0 + \sum_{j=1}^{M-1} w_j x^j$$

  where $\mathbf{w}$ is to be determined.
- Similarly, in a matrix-vector form

$$\underbrace{\mathbf{t}}_{N\times 1} = \widehat{\mathbf{t}} + \boldsymbol{\epsilon} = \underbrace{\boldsymbol{\phi}}_{N\times M}\underbrace{\mathbf{w}}_{M\times 1} + \underbrace{\boldsymbol{\epsilon}}_{N\times 1}$$

  where $\left[\boldsymbol{\phi}\right]_{n,k} = x_n^{k-1}$ is the so-called design matrix.
- We can select $\mathbf{w}$ minimizing

$$E\left(\mathbf{w}\right) = \left(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}\right)^{\mathsf{T}}\left(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}\right)$$

  and again obtain

$$\mathbf{w}_{\mathsf{LS}} = \left(\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}\right)^{-1}\boldsymbol{\phi}^{\mathsf{T}}\mathbf{t}.$$

Nonlinear Regression using Basis Functions

- Consider again $\mathbf{x} = \left(x^1, ..., x^d\right) \in \mathbb{R}^d$ and $t \in \mathbb{R}$ then you can consider the nonlinear model

$$\widehat{t}\left(\mathbf{x}\right) = \mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

  where $\phi_0(\mathbf{x}) = 1$ by convention, $\phi_j(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$ is an arbitrary "basis" function for $j = 1, ..., M-1$ and $\mathbf{w}$ is a $M-$dimensional vector to be determined.
- In this case, in a matrix-vector form we have

$$\mathbf{t} = \widehat{\mathbf{t}} + \boldsymbol{\epsilon} = \boldsymbol{\phi}\mathbf{w} + \boldsymbol{\epsilon}$$

  where

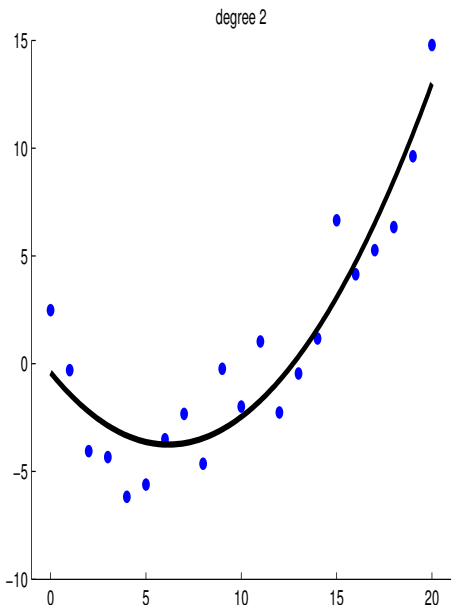
$$\left[\boldsymbol{\phi}\right]_{n,k} = \phi_{k-1}(\mathbf{x}_n).$$

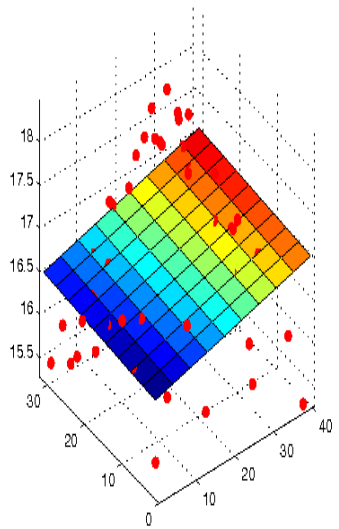
- Once more, if we minimized the sum of squared errors $E\left(\mathbf{w}\right) = \left(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}\right)^{\mathsf{T}} \left(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}\right)$ then

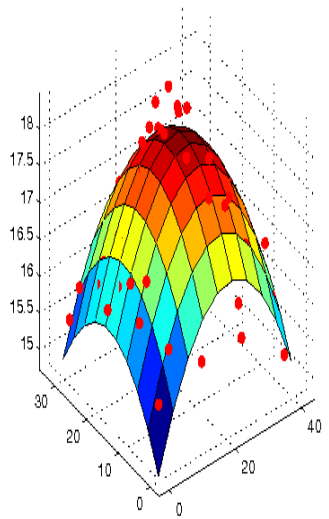
$$\mathbf{w}_{\mathsf{LS}} = \left(\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}\right)^{-1} \boldsymbol{\phi}^{\mathsf{T}}\mathbf{t}.$$

Nonlinear Regression using Basis Functions

- Example: $\Phi(\mathbf{x}) = \left(1, x, x^2\right)^{\mathsf{T}}$ $(d = 1, M = 3)$; $\Phi(\mathbf{x}) = \mathbf{x} = \left(1, x^1, x^2\right)$ $(d = 2, M = 3)$; $\Phi(\mathbf{x}) = \left(1, x^1, x^2, \left(x^1\right)^2, \left(x^2\right)^2\right)$ $(d = 2, M = 5)$.







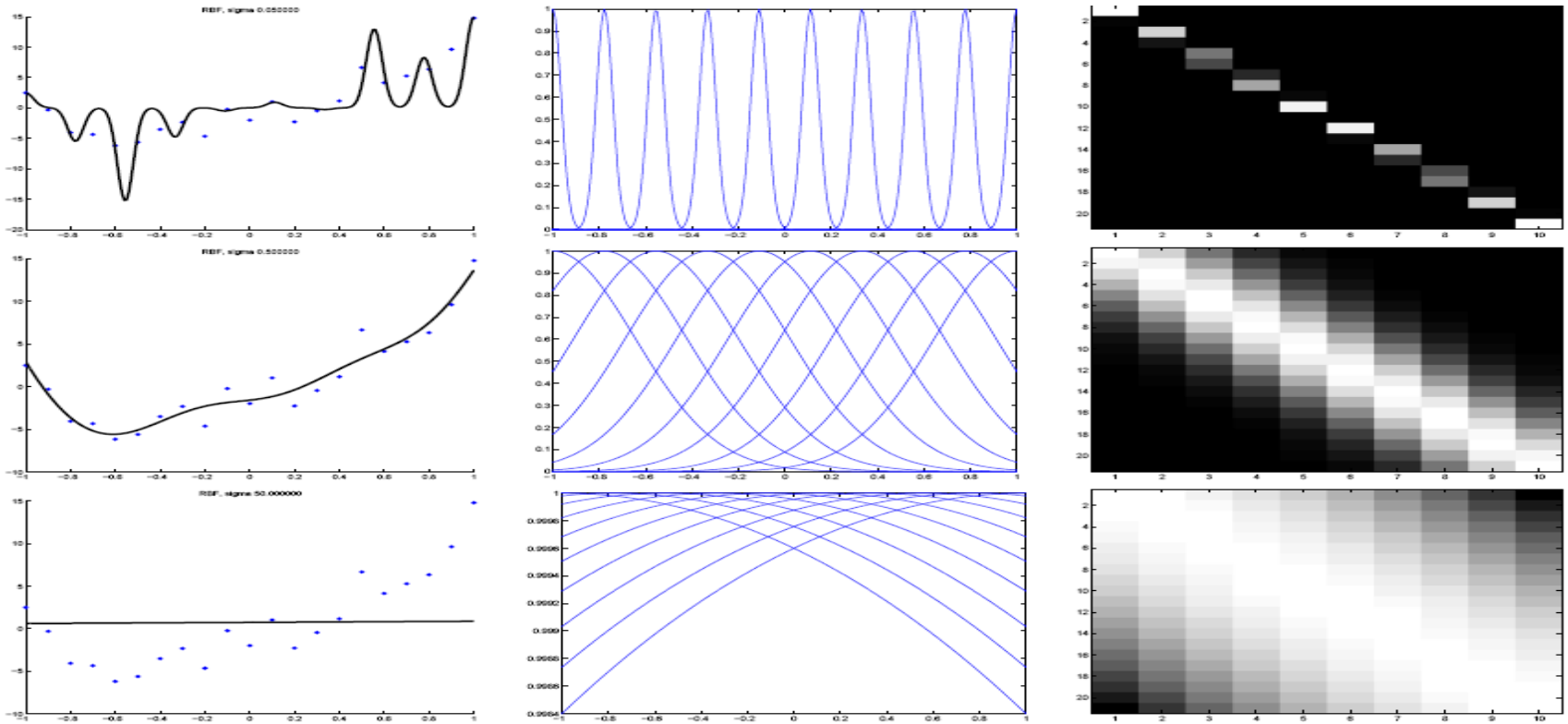
Nonlinear Regression using Basis Functions: Kernel Regression

- A standard way to perform nonlinear regression is to use kernels to define the basis functions $\Phi(\mathbf{x}) = \left(1, K(\mathbf{x}, \boldsymbol{\mu}_1), ..., K\left(\mathbf{x}, \boldsymbol{\mu}_{M-1}\right)\right)$ where

$$K(\mathbf{x}, \boldsymbol{\mu}) = \exp\left(-\frac{(\mathbf{x}-\boldsymbol{\mu})^{\mathsf{T}}(\mathbf{x}-\boldsymbol{\mu})}{2\sigma^2}\right).$$

- Alternatively we can use any function: wavelets, curvelets, splines etc.
- Selecting $\left(\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_m, \sigma^2\right)$ can be difficult.
- How to select $\boldsymbol{\mu}$: 1) place the centers unformly spaced in the region containing the data, 2) place one center at each data point, 3) cluster the data and use one center for each cluster, 4) use CV, MLE or Bayesian approach.
- How to select σ^2 : 1) use average squared distances to neighboring centers (scaled by a constant), 2) use CV, MLE or Bayesian approach.

Kernel Regression



(Left) RBF basis in 1d. (Middle) Basis functions evaluated on a grid. (Right) Design matrix.

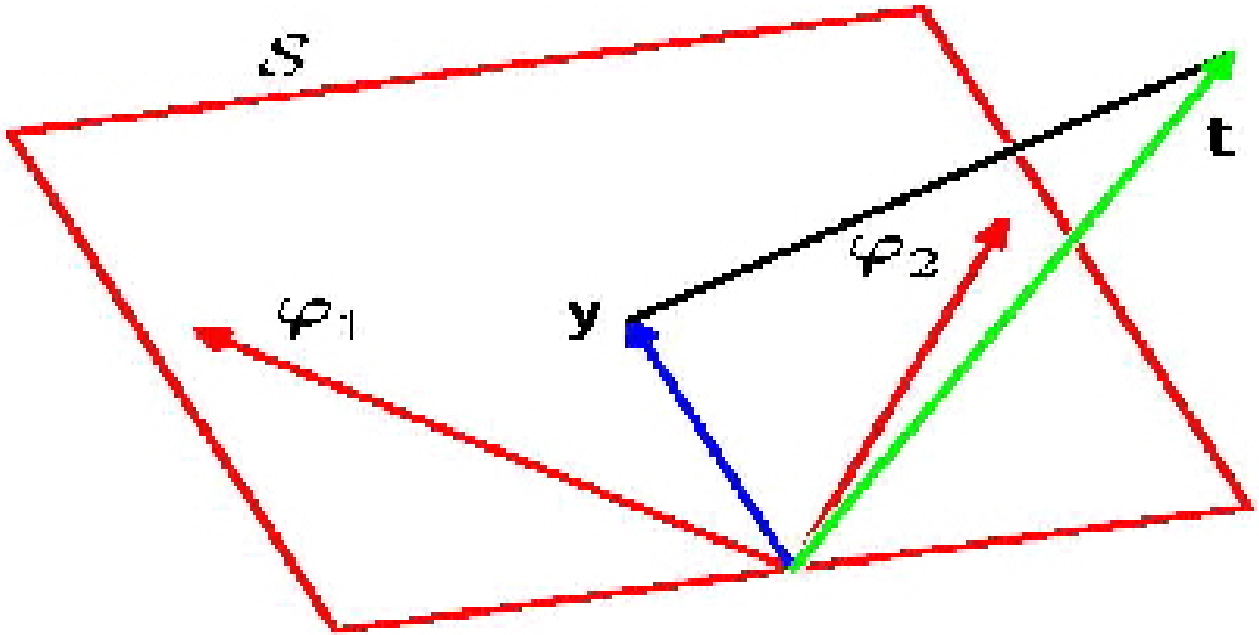
Geometric Interpretation of Least Squares

- We want to model the data $\mathbf{t} \in \mathbb{R}^N$.
- Consider the M vectors $\varphi_j = \left(\phi_{j-1}(\mathbf{x}_1), ..., \phi_{j-1}(\mathbf{x}_N)\right)^T \in \mathbb{R}^N$ for $j = 1, ..., M$ and

$$\mathbf{y} = \boldsymbol{\phi}\mathbf{w}_{\text{LS}}$$

- Then whatever being $\mathbf{w}_{\text{LS}}$ then $\mathbf{y}$ lies in the space spanned by $\left\{\varphi_j\right\}_{j=1}^{M}$.
- If $M \geq N$ (and if the vectors $\left\{\varphi_j\right\}_{j=1}^{M}$ are not colinear) then the space spanned by $\left\{\varphi_j\right\}_{j=1}^{M}$ is $\mathbb{R}^N$ so we will be able to find potentially an infinity of $\mathbf{w} \in \mathbb{R}^N$ such that $\widehat{\mathbf{t}} = \boldsymbol{\phi}\mathbf{w} = \mathbf{t}$.
- If $M < N$ then the vectors $\left\{\varphi_j\right\}_{j=1}^{M}$ span a M-dimensional subspace of $\mathbb{R}^N$ and the $\mathbf{y} = \boldsymbol{\phi}\mathbf{w}_{LS}$ is the point in this subspace that is the closest to $\mathbf{t}$ (in sense of the square error).

Geometric Interpretation



- $\mathbf{y} = \boldsymbol{\phi}\mathbf{w}_{LS}$ is the orthogonal projection of $\mathbf{t}$ on the subspace spanned by $\left\{\varphi_j\right\}_{j=1}^{M}$.

Geometric Interpretation

- To check that this is is the orthogonal projection, we need to establish that

$$\boldsymbol{\phi}^{\mathsf{T}}\left(\mathbf{t}-\boldsymbol{\phi}\mathbf{w}_{LS}\right)=\mathbf{0};$$

that is the residual is orthogonal to all the vectors $\left\{\varphi_j\right\}_{j=1}^{M}$.

- We have

$$\mathbf{t}-\boldsymbol{\phi}\mathbf{w}_{LS}=\mathbf{t}-\boldsymbol{\phi}\left(\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}\right)^{-1}\boldsymbol{\phi}^{\mathsf{T}}\mathbf{t}$$

so

$$\boldsymbol{\phi}^{\mathsf{T}}\left(\mathbf{t}-\boldsymbol{\phi}\mathbf{w}_{LS}\right)=\boldsymbol{\phi}^{\mathsf{T}}\mathbf{t}-\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}\left(\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}\right)^{-1}\boldsymbol{\phi}^{\mathsf{T}}\mathbf{t}=\mathbf{0}$$

Least Squares: A Probabilistic Interpretation

- Consider the following probabilistic regression model

$$p\left(t|\,\mathbf{x},\mathbf{w},\beta\right)=\mathcal{N}\left(t;\widehat{t}\left(\mathbf{x}\right),\beta^{-1}\right)$$

where $\widehat{t}\left(\mathbf{x}\right)=\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x})$; i.e.

$$t=\widehat{t}\left(\mathbf{x}\right)+\epsilon$$

with $\epsilon\sim\mathcal{N}\left(0,\beta^{-1}\right)$.

- Given the training set $D=\left\{\mathbf{x}_n,t_n\right\}_{n=1}^{N}$, the ML estimates of $(\mathbf{w},\beta)$ is given by

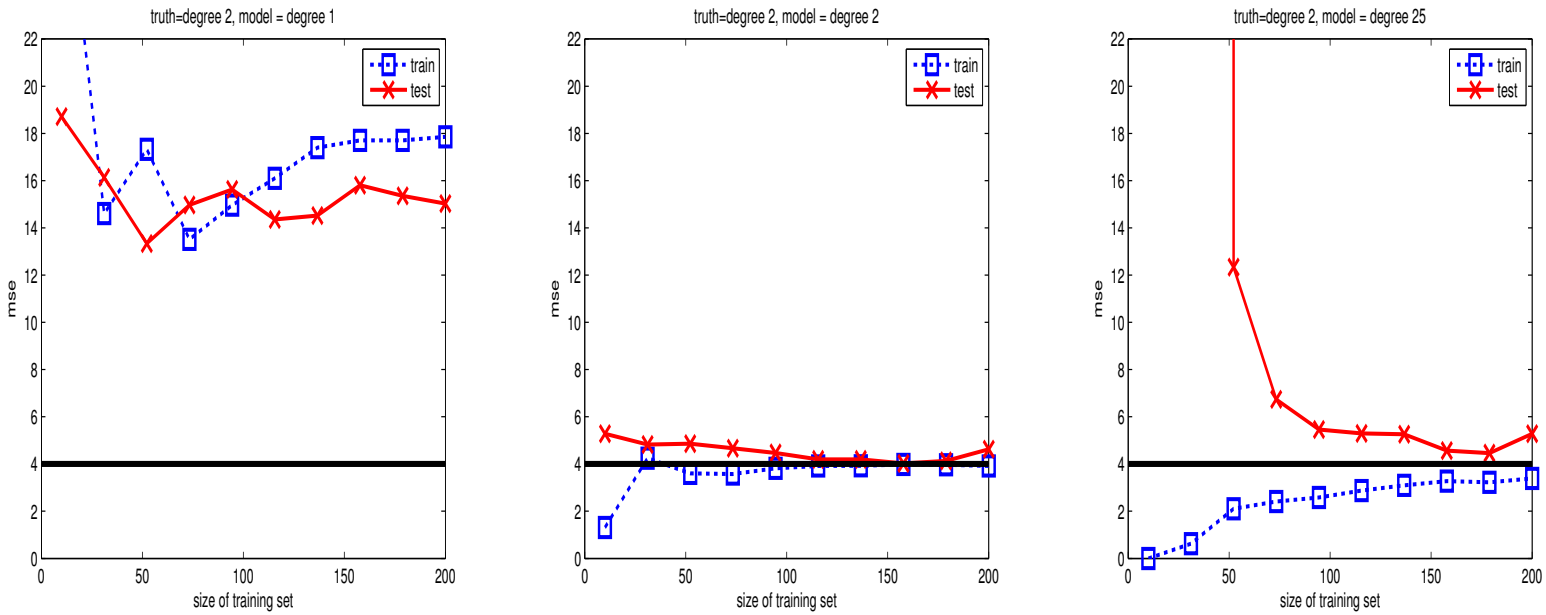
$$\left(\mathbf{w}_{ML},\beta_{ML}\right)=\underset{(\mathbf{w},\beta)}{\arg\max}\sum_{i=1}^{N}\log p\left(t_n|\,\mathbf{x}_n,\mathbf{w},\beta^{-1}\right)$$

where

$$\sum_{n=1}^{N}\log p\left(t_n|\,\mathbf{x}_n,\mathbf{w},\beta\right)=-\frac{N}{2}\log\left(2\pi\beta^{-1}\right)-\frac{\beta E\left(\mathbf{w}\right)}{2}$$

- We obtain $\mathbf{w}_{ML}=\mathbf{w}_{LS}$ and $\beta_{ML}^{-1}=E\left(\mathbf{w}_{LS}\right)/N$.

Examples: Mean Square Errors on Training and Test Sets



Data generated from a degree 2 poly. with Gaussian noise of var. 4. We fit polynomial models. For N small, test error of the degree 25 poly. is higher than that of the degree 2 poly., due to overfitting, but this difference vanishes once as N increases. Note also that the degree 1 poly. is too simple and has high test error even given large N.

Limitations of Least-Square Regression and MLE

- Remember that our estimate is given by

$$\mathbf{w}_{MLE} = \mathbf{w}_{LS} = \left(\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}\right)^{-1}\boldsymbol{\phi}^{\mathsf{T}}\mathbf{t}.$$

- This assumes that the $M \times M$ matrix $\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}$ is invertible.
- We have $\text{rank}\left(\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}\right) = \text{rank}(\boldsymbol{\phi}) \leq \min\left(N, M\right)$. Hence in common scenarios where $N < M$, we can never invert $\boldsymbol{\phi}^{\mathsf{T}}\boldsymbol{\phi}$!
- We have also problems when columns/rows of $\boldsymbol{\phi}$ are almost linearly dependent (collinearity).

The Need for Regularization

- When we have a small number of data, we want to be able to regularize the solution and limit overfitting.
- When fitting a polynomial regression model, "wiggly" functions will have large weights $\mathbf{w}$.
- For example for the 14 polynomial model fitted previously, we have 11 coefficients w_k such that $|\widehat{w}_{k,LS}| > 100$!

Regularized Least Squares - Penalized Likelihood

- Two problems with MLE: overfitting and $\mathbf{w}_{ML}$ might not exist.
- To control this problem we are going to add a regularization term; e.g. instead of minimizing

$$E(\mathbf{w}) = (\mathbf{t} - \boldsymbol{\phi}\mathbf{w})^{\mathsf{T}} (\mathbf{t} - \boldsymbol{\phi}\mathbf{w})$$

we minimize

$$E'(\mathbf{w}) = \frac{E(\mathbf{w})}{2} + \lambda E_W(\mathbf{w})$$

where $\lambda > 0$ is a penalization term and $E_W(\mathbf{w}) \to \infty$ as $\|\mathbf{w}\| \to \infty$.

Bayesian MAP Interpretation

- If we set a prior distribution on $\mathbf{w}$ of the form

$$p\left(\mathbf{w}\middle|\,\beta,\lambda\right) \propto \exp\left(-\beta\lambda E_W\left(\mathbf{w}\right)\right)$$

  then minimizing $E'\left(\mathbf{w}\right)$ is equivalent to compute the Maximum A Posteriori estimate of $\mathbf{w}$ defined by

$$\mathbf{w}_{\mathrm{MAP}} = \arg\max_{\mathbf{w}} p\left(\mathbf{w}\middle|\,D,\beta,\lambda\right)$$

- Similarly most regularized least squares procedures have a Bayesian interpretation.

L2 Penalization - Ridge Regression

- Consider the case where

$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{w} \Leftrightarrow p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}; \mathbf{0}, (\beta\boldsymbol{\lambda})^{-1}\mathbf{I}\right)$$

- In this case, we minimize

$$E'(\mathbf{w}) = \frac{1}{2}(\mathbf{t} - \boldsymbol{\phi}\mathbf{w})^\mathsf{T}(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^\mathsf{T}\mathbf{w}$$

  and the closed-form solution is

$$\mathbf{w}_{MAP} = \left(\boldsymbol{\phi}^\mathsf{T}\boldsymbol{\phi} + \boldsymbol{\lambda}\mathbf{I}\right)^{-1}\boldsymbol{\phi}^\mathsf{T}\mathbf{t}$$

- This is a very popular approach as $\left(\boldsymbol{\phi}^\mathsf{T}\boldsymbol{\phi} + \boldsymbol{\lambda}\mathbf{I}\right)^{-1}$ always exists.

Ridge Regression as A Constrained Optimization Problem

- Minimizing

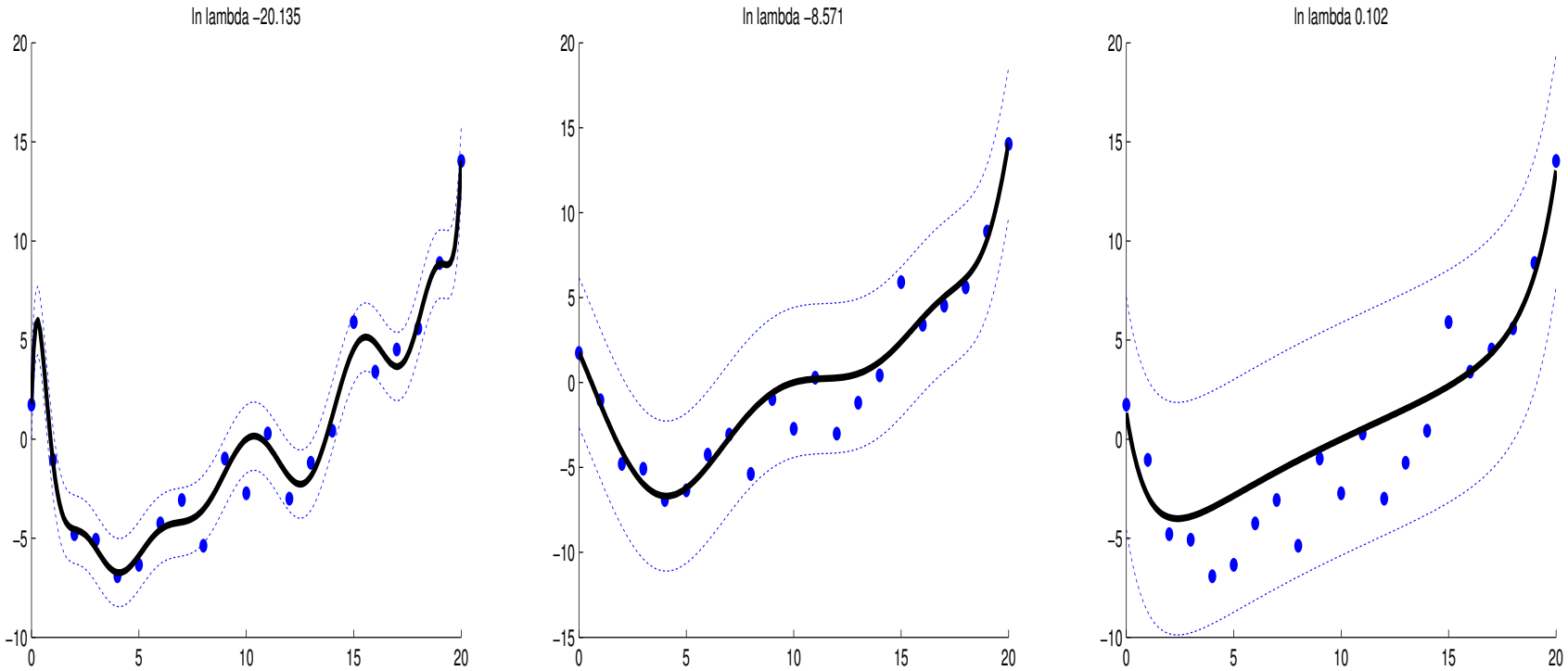
$$E'(\mathbf{w}) = \frac{1}{2}(\mathbf{t} - \boldsymbol{\phi}\mathbf{w})^{\mathsf{T}}(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^{\mathsf{T}}\mathbf{w}$$

  is equivalent to minimize

$$(\mathbf{t} - \boldsymbol{\phi}\mathbf{w})^{\mathsf{T}}(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}) \quad \text{s.t. } \mathbf{w}^{\mathsf{T}}\mathbf{w} \leq t(\lambda)$$

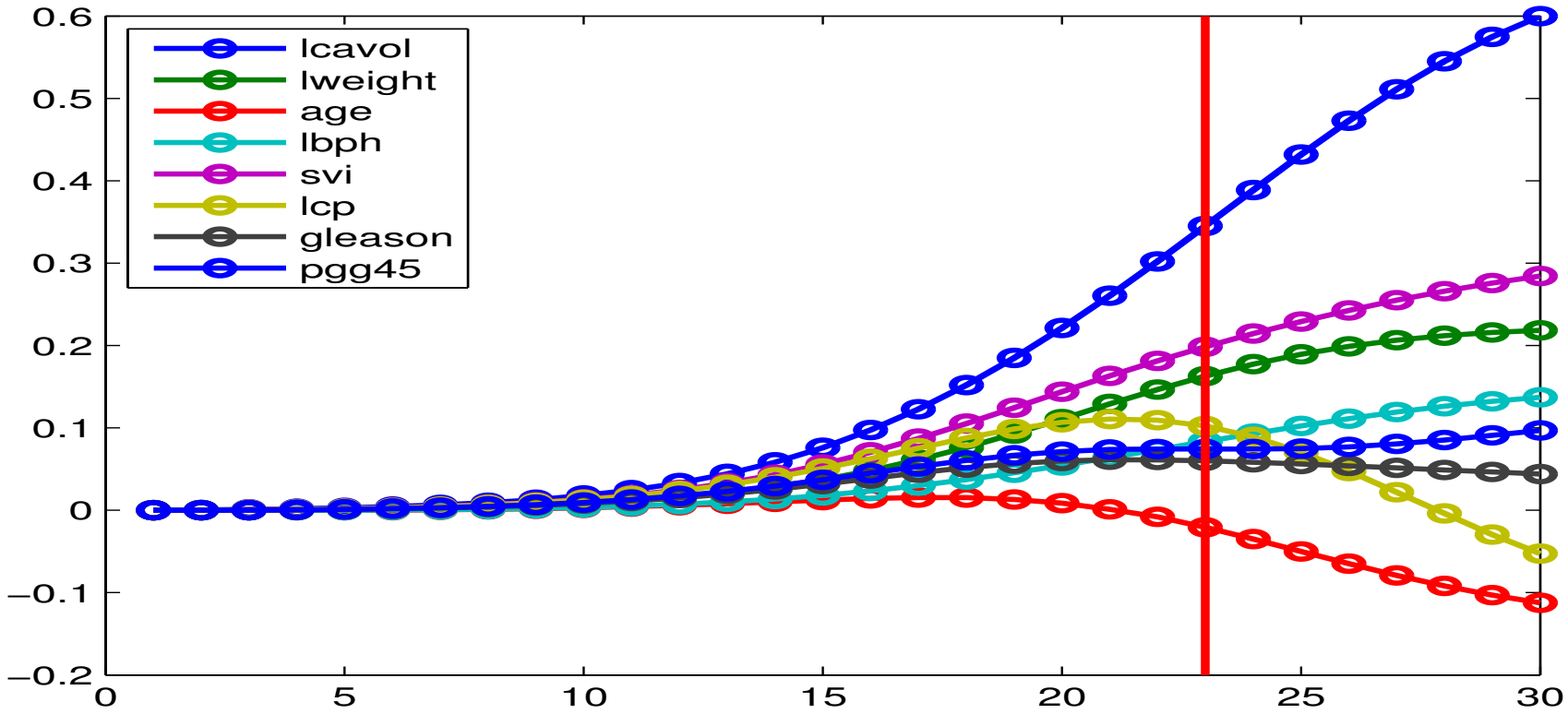
  where $t(\lambda) \longrightarrow 0$ as $\lambda \longrightarrow \infty$.
- This shrinks the value of $\mathbf{w}_{MAP}$ towards zeros.

Example of Ridge Regression



Degree 14 polynomial fit to $N = 21$ data points with increasing λ. The errors bars, represents the standard deviation $\sqrt{\widehat{\beta}_{MAP}^{-1}}$.

Regularization Path for Ridge Regression



Profile of Ridge coefficients for an example on real data where $M = 8$ vs bound on $\mathbf{w}^{\mathsf{T}}\mathbf{w}$, i.e. small $t\,(\lambda)$ means large λ.

L1 Penalization - Lasso

- Consider the case where

$$E_W(\mathbf{w}) = \frac{1}{2}\sum_{j=0}^{M-1} |w_j|$$

then

$$p(\mathbf{w}|\beta,\lambda) = \prod_{j=0}^{M-1} p(w_j|\beta,\lambda)$$

where

$$p(w_j|\beta,\lambda) = \frac{\lambda\beta}{2}\exp(-\lambda\beta|w_j|)$$

- This prior is a so-called double exponential.
- There is no closed-form expression for the minimizer of $E(\mathbf{w})$.
- The optimization problem is convex and can be solved using standard quadratic programming algorithms.

Lasso as A Constrained Optimization Problem

- We minimize in this case

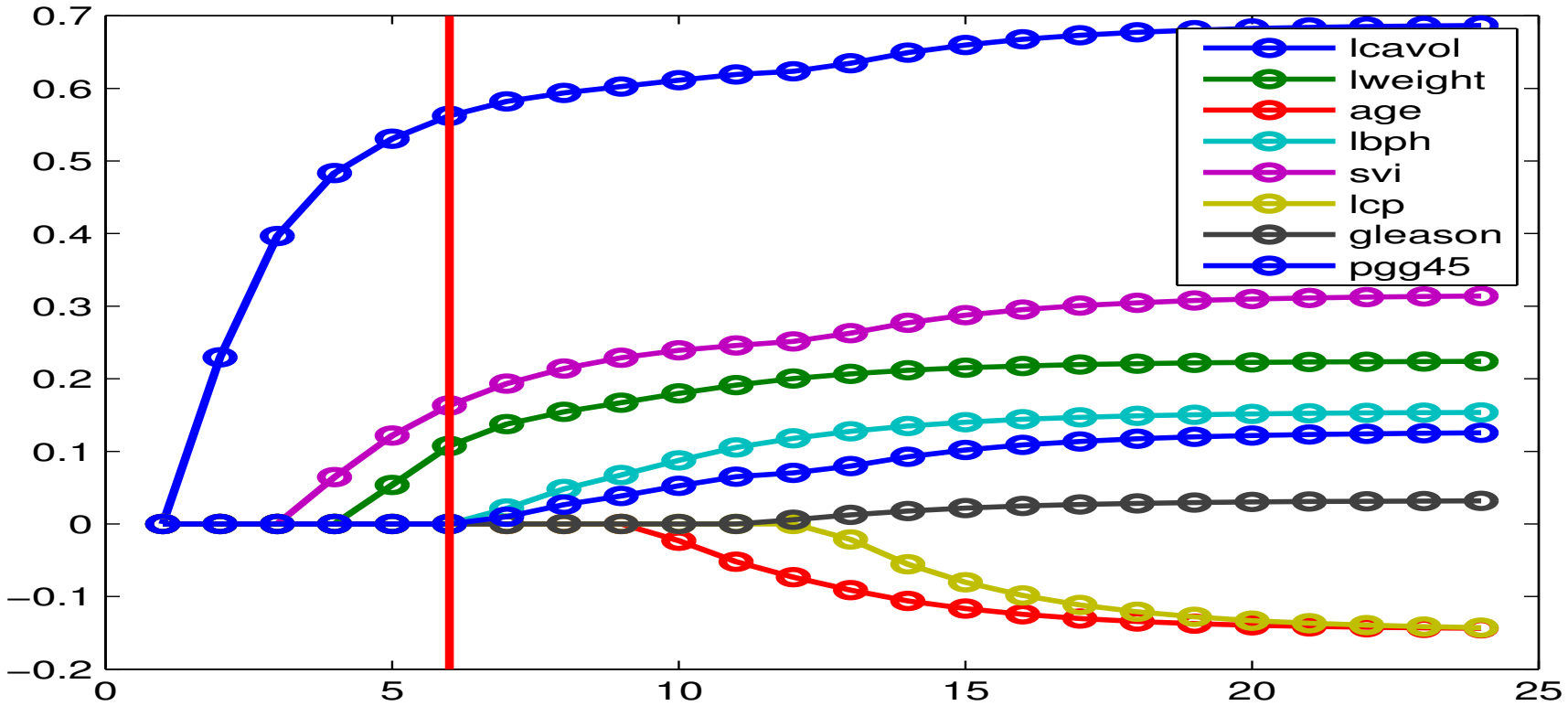
$$E'(\mathbf{w}) = \frac{1}{2}(\mathbf{t} - \boldsymbol{\phi}\mathbf{w})^{\mathsf{T}}(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}) + \frac{\lambda}{2}\sum_{j=0}^{M-1}|w_j|$$

- It can be shown that this is equivalent to minimize

$$(\mathbf{t} - \boldsymbol{\phi}\mathbf{w})^{\mathsf{T}}(\mathbf{t} - \boldsymbol{\phi}\mathbf{w}) \quad \text{s.t. } \mathbf{w}^{\mathsf{T}}\mathbf{w} \leq t(\lambda)$$

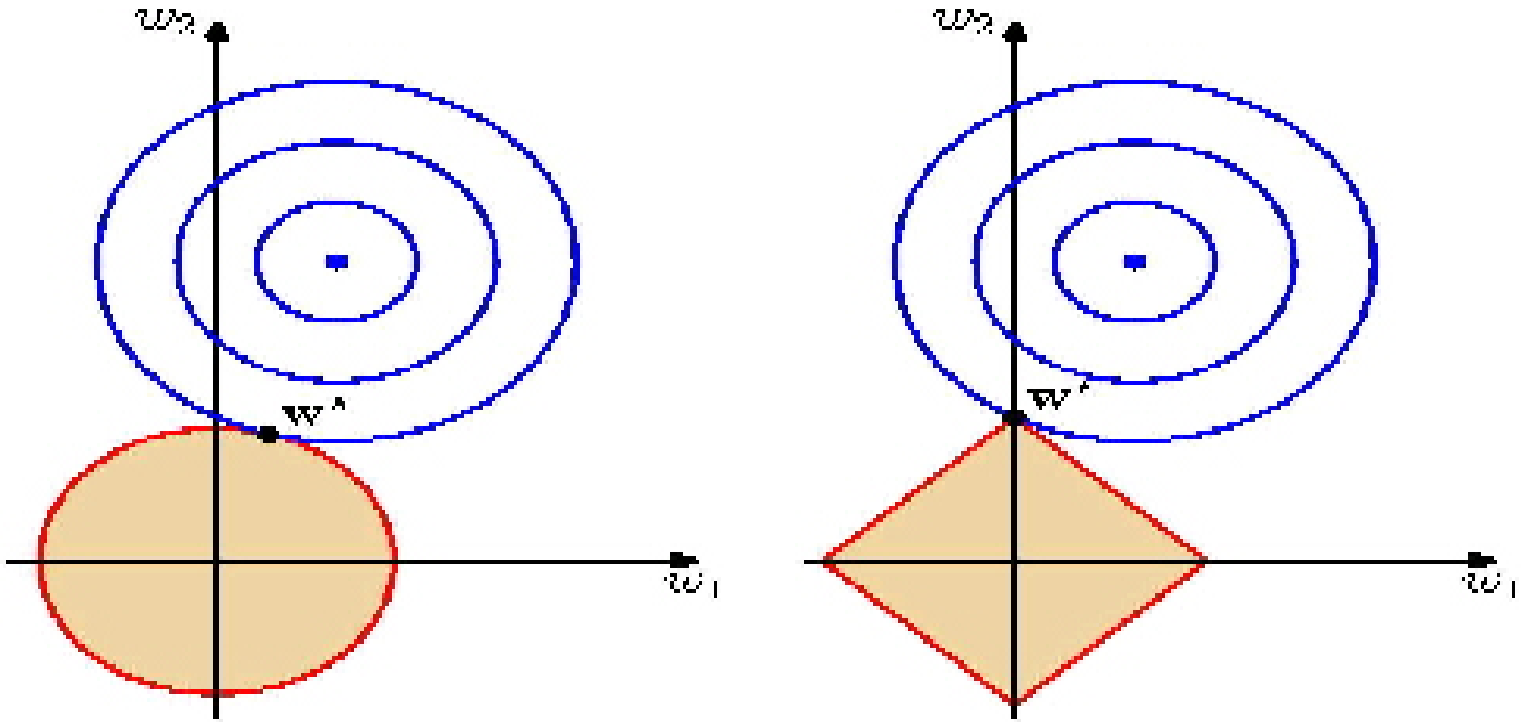
where $t(\lambda) \longrightarrow 0$ as $\longrightarrow \infty$.

Regularization Path for Lasso



Profile of Lasso coefficients for an example on real data where $M = 8$ vs bound on $\sum_{j=0}^{M-1} |w_j|$, i.e. small $t(\lambda)$ means large λ.

L1 versus L2 penalization

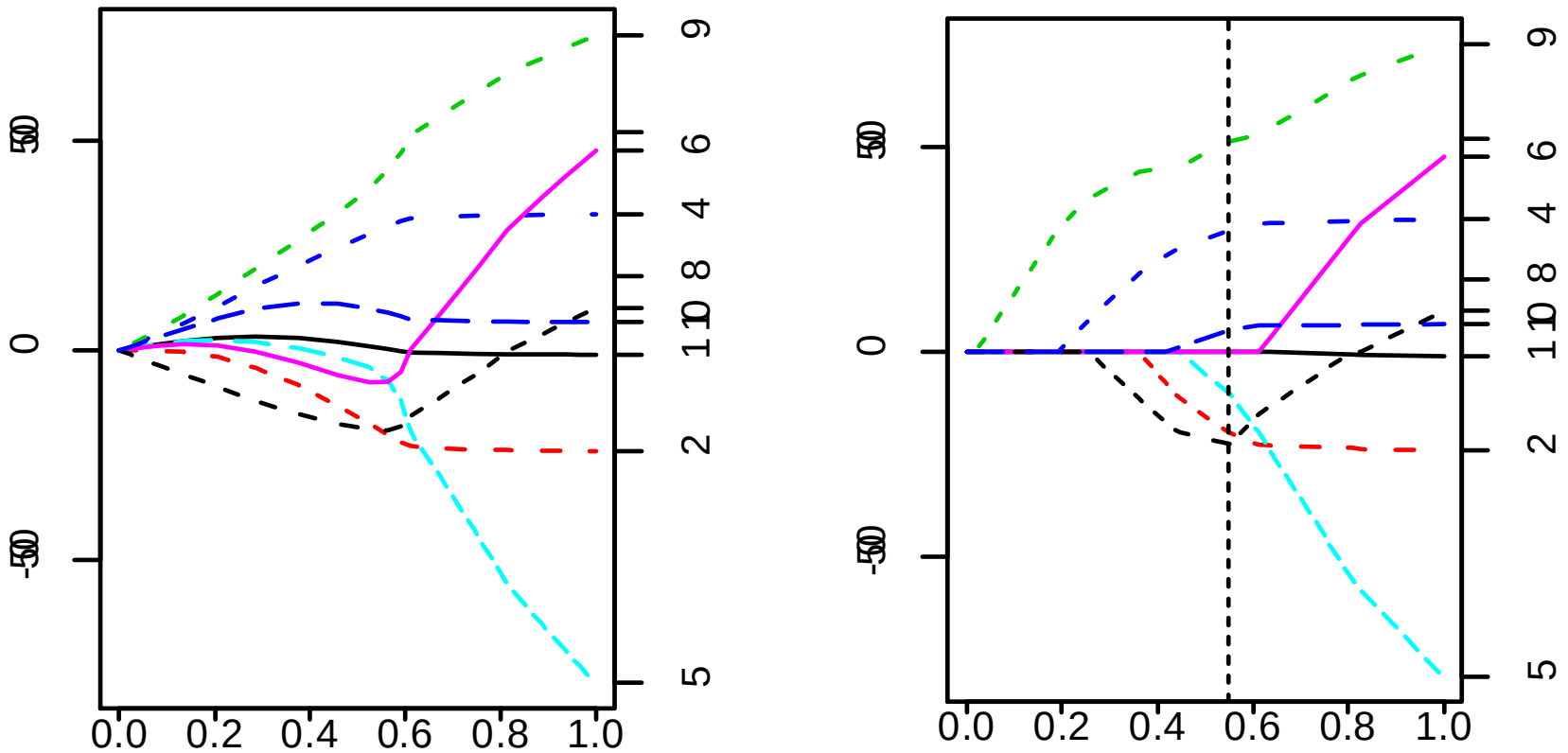


Plot of the contours of the unregularized error function (blue) for ridge regression (left) and Lasso (right)

- The lasso give a sparse solution as $w_1^* = 0$. The corners of the simplex is more likely to intersect the ellipse than one of the sides as they "stick out" more.

Diabetes Data

- This is a data set where $N = 442$ and $M = 10$.
- We compute the solution for various regularization parameters λ



Regularization paths for Ridge Regression (left) and Lasso (Right)
We display w_j as λ goes from ∞ to 0

Pros and Cons of Lasso

- Pros
  - Lasso has remarkable properties. For a properly selected penalization, it can find the 'right' variables to include in the model (asymptotically): no need to variable selection model...
  - Efficient packages have been developed to implement it.
  - You can find the regularization path by running it only once.
- Cons
  - Finite sample properties are less clear.
  - The posterior distribution associated to Lasso is multimodal.
- Lasso is becoming prevalent in machine learning and statistics.

Summary

- MLE has bad properties.
- Penalized MLE has nicer properties and can be reinterpreted as Bayesian MAP estimate.
- For non-quadratic penalizations, the posterior might not admit a closed-form but can still lead to tractable optimization problems.
- We have yet to address key problems
  - How to set the free parameters of the regression model? i.e. regularization parameter λ
  - How to select the number of basis M to include?
  - Given different models (e.g. radial basis functions, wavelets), which one should we pick?
- Various approaches are possible: cross-validation, full Bayesian analysis.

Cross-Validation: Ridge Regression Example

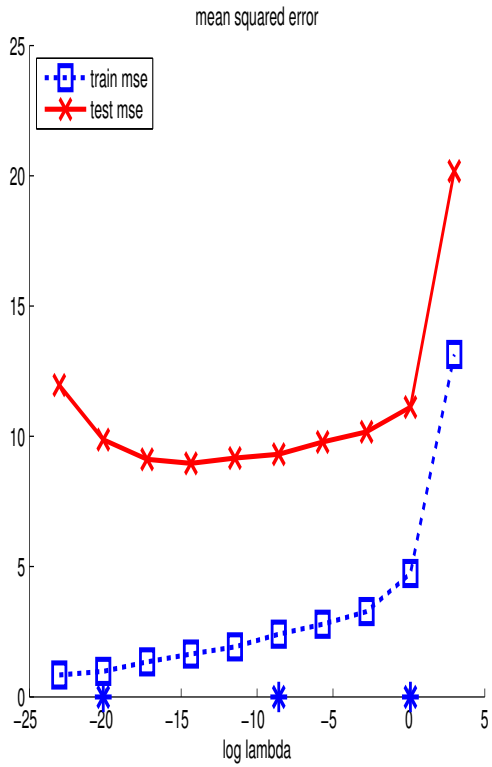
- In practice, we need to adjust parameters of the model; e.g. when performing ridge regression, you need to set λ.
- We want procedures to do perform well on training data but also to perform well on unseen 'test' data.
- However, in real-world applications, we cannot evaluate our prediction error on the test set!
- A simple idea to evaluate the error rate consists of splitting the training data into two blocks: a block used as training data and the other block known as validation set.
- **Example**: Assume you are given $\{\mathbf{x}_n, t_n\}_{n=1}^{N}$ training data, then only $N_{train} < N$ data, say $\{\mathbf{x}_n, t_n\}_{n=1}^{N_{train}}$ are used as training data whereas the remaining $N_{valid} = N - N_{train}$ data $\{\mathbf{x}_n, t_n\}_{n=N_{train}+1}^{N}$ are used to assess the performance of our regression function using say $MSE = \frac{1}{N_{valid}} \sum_{n=N_{train}+1}^{N} \left(t_n - y\left(\mathbf{x}_n\right)\right)^2$ where $y\left(\mathbf{x}_n\right) = \mathbf{w}_{MAP}^{\mathsf{T}} \boldsymbol{\phi}\left(\mathbf{x}_n\right)$ for ridge regression.
- Compute MSE for various λ and select the one which minimizes Err.

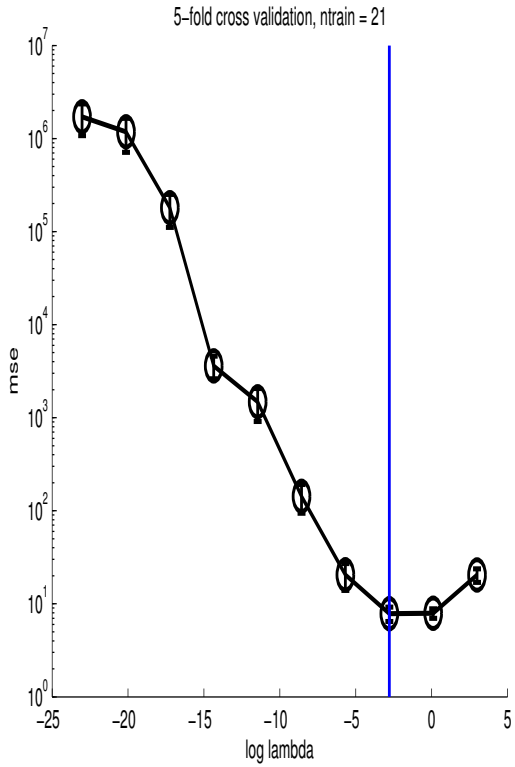
Cross-Validation

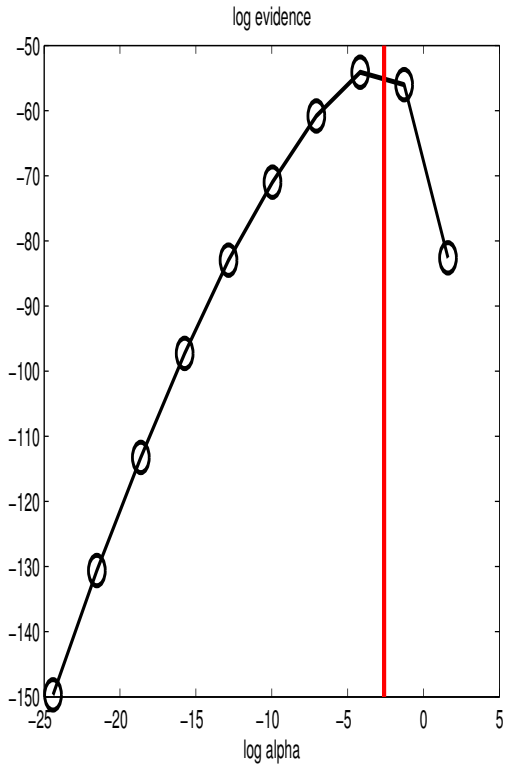
- If N is small, this technique is unreliable as the model won't have enough data to train on, and we won't have enough data to make a reliable estimate of the future performance.
- A simple and popular solution to this is K-fold **cross validation** (CV). We split the training data into K folds then, for each fold $k \in \{1, 2, ..., K\}$, we train on all the folds but the k'th, and test on the k'th, in a round-robin fashion to estimate $MSE = \frac{1}{K} \sum_{k=1}^{K} MSE_k$. N-fold CV is called leave-one-out CV.

Figure: 5-fold cross validation

Cross-Validation for Ridge Regression







(left) Training and test error for a degree 14 poly. with increasing λ. (center) Estimate of test MSE produced by 5-fold CV. (right) Log-marginal likelihood vs $\log(\alpha)$ where $\alpha = \lambda\beta$.

Cross-Validation

- Cross-validation is extremely general.
- It can be used to pick the number of basis M for your regression model: you just need to compute the MSE on validation sets for various $M \in \{1, 2, ..., M_{\max}\}$.
- It can also be used for classification as described later.
- Main problem of Cross-Validation: It can be very computationally intensive...

A (Semi)-Bayesian Approach: Using the Evidence

- Assume you are interested in performing ridge regression, you need to set λ.
- Using the Bayesian interpretation of ridge regression, you have a Gaussian prior $p(\mathbf{w}|\beta,\lambda) = \mathcal{N}\left(\mathbf{w};\mathbf{0},(\beta\boldsymbol{\lambda})^{-1}\mathbf{I}\right)$. (Assume β is known here for sake of simplicity).
- Combined to the likelihood of the data, we have the posterior

$$p(\mathbf{w}|D,\beta,\lambda) = \frac{p(D|\mathbf{w},\beta,\lambda)\,p(\mathbf{w}|\beta,\lambda)}{p(D|\beta,\lambda)}$$

but also the marginal likelihood, known as the evidence, given by

$$p(D|\beta,\lambda) = \int p(D|\mathbf{w},\beta,\lambda)\,p(\mathbf{w}|\beta,\lambda)\,d\mathbf{w}$$

which can be computed exactly.

- It is sensible to set λ as the value which maximizes $p(D|\beta,\lambda)$.

A (Semi)-Bayesian Approach: Using the Evidence

- Assume you are interested in performing lasso regression, you need to set λ.
- Using the Bayesian interpretation of lasso regression, you have

$$p\left(\mathbf{w}\middle|\,\beta,\lambda\right)=\prod_{j=0}^{M-1}\frac{\lambda\beta}{2}\exp\left(-\lambda\beta\left|w_j\right|\right)$$

- Combined to the likelihood of the data, we have the posterior

$$p\left(\mathbf{w}\middle|\,D,\beta,\lambda\right)=\frac{p\left(D\middle|\,\mathbf{w},\beta,\lambda\right)p\left(\mathbf{w}\middle|\,\beta,\lambda\right)}{p\left(D\middle|\,\beta,\lambda\right)}$$

but also the marginal likelihood, known as the evidence, given by

$$p\left(D\middle|\,\beta,\lambda\right)=\int p\left(D\middle|\,\mathbf{w},\beta,\lambda\right)p\left(\mathbf{w}\middle|\,\beta,\lambda\right)d\mathbf{w}$$

which can NOT be computed exactly. Approximations are required.

- It is sensible to set λ as the value which maximizes $p\left(D\middle|\,\beta,\lambda\right)$.