Lecture 3: More on regularization. Bayesian vs maximum likelihood learning

- L2 and L1 regularization for linear estimators
- A Bayesian interpretation of regularization
- Bayesian vs maximum likelihood fitting more generally

Recall: Regularization

- Remember the intuition: complicated hypotheses lead to overfitting
- Idea: change the error function to *penalize hypothesis complexity*:

$$J(\mathbf{w}) = J_D(\mathbf{w}) + \lambda J_{pen}(\mathbf{w})$$

  This is called *regularization* in machine learning and *shrinkage* in statistics
- λ is called *regularization coefficient* and controls how much we value fitting the data well, vs. a simple hypothesis

Recall: What L_2 regularization for linear models does

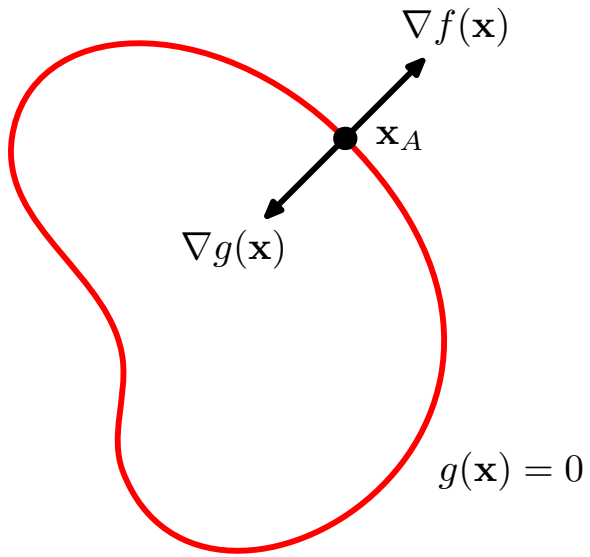
$$\arg\min_{\mathbf{w}} \frac{1}{2}(\mathbf{\Phi w} - \mathbf{y})^T(\mathbf{\Phi w} - \mathbf{y}) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} = (\mathbf{\Phi}^T\mathbf{\Phi} + \lambda I)^{-1}\mathbf{\Phi}^T\mathbf{y}$$

- If $\lambda = 0$, the solution is the same as in regular least-squares linear regression
- If $\lambda \to \infty$, the solution $\mathbf{w} \to 0$
- Positive λ will cause the magnitude of the weights to be smaller than in the usual linear solution
- This is also called *ridge regression*, and it is a special case of Tikhonov regularization (more on that later)
- A different view of regularization: we want to optimize the error while keeping the L_2 norm of the weights, $\mathbf{w}^T\mathbf{w}$, bounded.

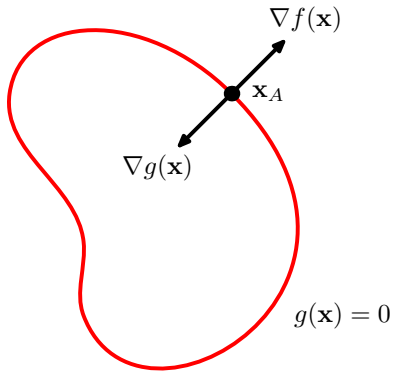
Detour: Constrained optimization

Suppose we want to find

$$\min_{\mathbf{w}} f(\mathbf{w})$$
$$\text{such that } g(\mathbf{w}) = 0$$



Detour: Lagrange multipliers

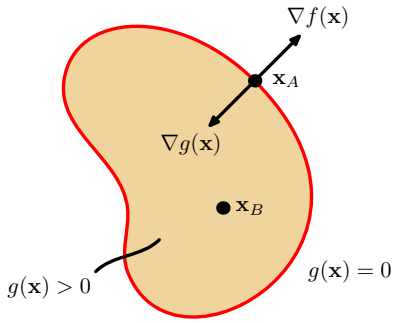


- ∇g has to be orthogonal to the constraint surface (red curve)
- At the optimum, ∇f and ∇g have to be parallel (in same or opposite direction)
- Hence, there must exist some $\lambda \in \mathbb{R}$ such that $\nabla f + \lambda \nabla g = 0$
- *Lagrangian function*: $L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$
  λ is called *Lagrange multiplier*
- We obtain the solution to our optimization problem by setting both $\nabla_{\mathbf{x}} L = 0$ and $\frac{\partial L}{\partial \lambda} = 0$

Detour: Inequality constraints

- Suppose we want to find

$$\min_{\mathbf{w}} f(\mathbf{w})$$
$$\text{such that } g(\mathbf{w}) \geq 0$$



- In the interior ($g(\mathbf{x} > 0)$) - simply find $\nabla f(\mathbf{x}) = 0$
- On the boundary ($g(\mathbf{x} = 0)$) - same situation as before, but the sign matters this time
  For minimization, we want ∇f pointing in the same direction as ∇g

Detour: KKT conditions

- Based on the previous observations, let the Lagrangian be $L(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda g(\mathbf{x})$
- We minimize L wrt $\mathbf{x}$ subject to the following constraints:

$$\begin{aligned} \lambda &\geq 0 \\ g(\mathbf{x}) &\geq 0 \\ \lambda g(\mathbf{x}) &= 0 \end{aligned}$$

- These are called *Karush-Kuhn-Tucker (KKT) conditions*

L_2 Regularization for linear models revisited

- Optimization problem: minimize error while keeping norm of the weights bounded

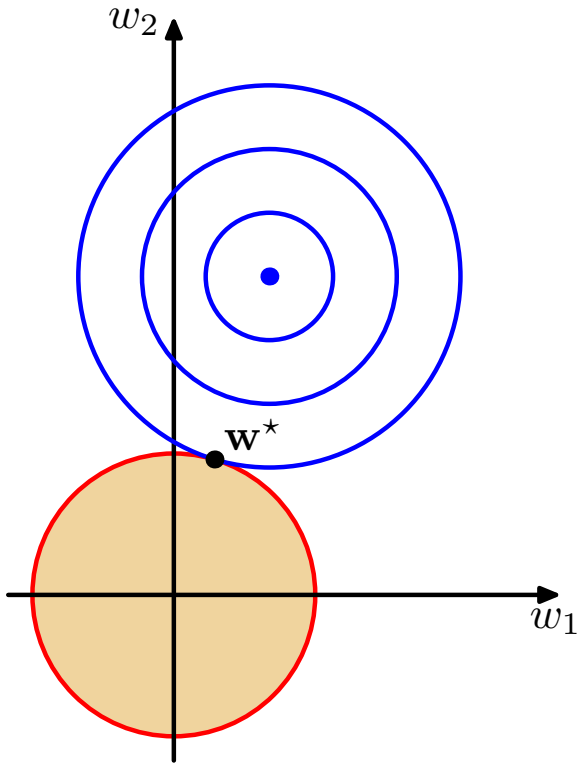
$$\begin{aligned} \min_{\mathbf{w}} J_D(\mathbf{w}) &= \min_{\mathbf{w}} (\mathbf{\Phi w} - \mathbf{y})^T (\mathbf{\Phi w} - \mathbf{y}) \\ \text{such that } \mathbf{w}^T \mathbf{w} &\leq \eta \end{aligned}$$

- The Lagrangian is:

$$L(\mathbf{w}, \lambda) = J_D(\mathbf{w}) - \lambda(\eta - \mathbf{w}^T \mathbf{w}) = (\mathbf{\Phi w} - \mathbf{y})^T (\mathbf{\Phi w} - \mathbf{y}) + \lambda \mathbf{w}^T \mathbf{w} - \lambda \eta$$

- For a fixed λ, and $\eta = \lambda^{-1}$, the best $\mathbf{w}$ is the same as obtained by weight decay

Visualizing regularization (2 parameters)



$$\mathbf{w}^* = (\mathbf{\Phi}^T\mathbf{\Phi} + \lambda I)^{-1}\mathbf{\Phi}\mathbf{y}$$

Pros and cons of L_2 regularization

- If λ is at a "good" value, regularization helps to avoid overfitting
- Choosing λ may be hard: cross-validation is often used
- If there are irrelevant features in the input (i.e. features that do not affect the output), L_2 will give them small, but non-zero weights.
- Ideally, irrelevant input should have weights exactly equal to 0.

L_1 Regularization for linear models

- Instead of requiring the L_2 norm of the weight vector to be bounded, make the requirement on the L_1 norm:

$$\begin{aligned} \min_{\mathbf{w}} J_D(\mathbf{w}) &= \min_{\mathbf{w}} (\mathbf{\Phi w} - \mathbf{y})^T (\mathbf{\Phi w} - \mathbf{y}) \\ \text{such that } \sum_{i=1}^{n} |w_i| &\leq \eta \end{aligned}$$

- This yields an algorithm called Lasso (Tibshirani, 1996)

Solving L_1 regularization

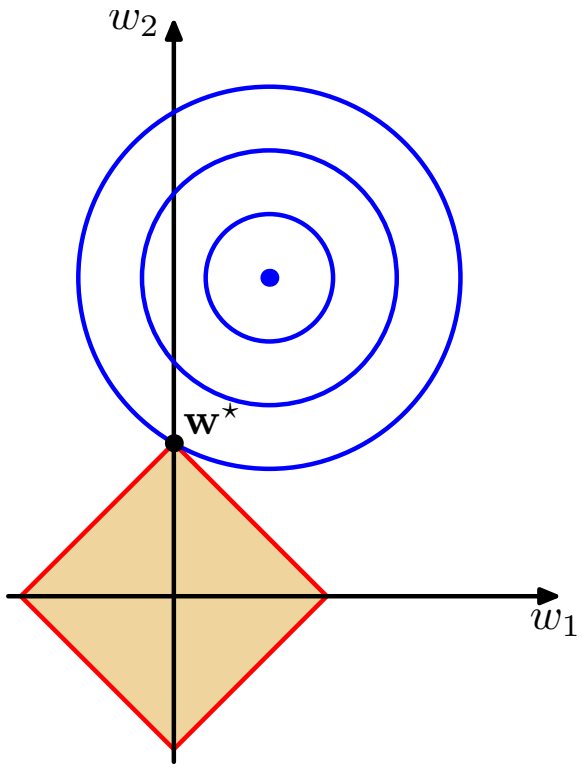
- The optimization problem is a quadratic program
- There is one constraint for each possible sign of the weights (2^n constraints for n weights)
- For example, with two weights:

$$\min_{w_1,w_2} \quad \sum_{j=1}^{m}(y_j - w_1x_1 - w_2x_2)^2$$

$$\begin{aligned} \text{such that } w_1 + w_2 &\leq \eta \\ w_1 - w_2 &\leq \eta \\ -w_1 + w_2 &\leq \eta \\ -w_1 - w_2 &\leq \eta \end{aligned}$$

- Solving this program directly can be done for problems with a small number of inputs

Visualizing L_1 regularization

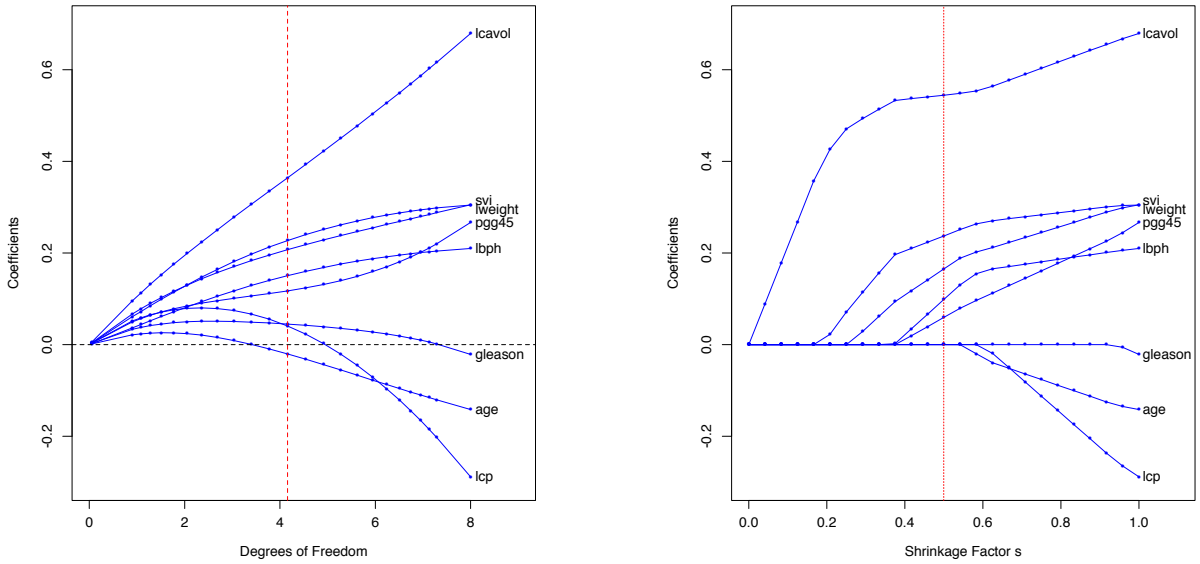


- If λ is big enough, the circle is very likely to intersect the diamond at one of the corners
- This makes L_1 regularization much more likely to make some weights *exactly* 0

Pros and cons of L_1 regularization

- If there are irrelevant input features, Lasso is likely to make their weights 0, while L_2 is likely to just make all weights small
- Lasso is biased towards providing *sparse solutions* in general
- Lasso optimization is computationally more expensive than L_2
- More efficient solution methods have to be used for large numbers of inputs (e.g. least-angle regression, 2003).
- L_1 methods of various types are very popular

Example of L1 vs L2 effect



- Note the sparsity in the coefficients induces by L_1
- Lasso is an efficient way of performing the L_1 optimization

More generally: statistical parameter fitting

- Given instances $\mathbf{x}_1, \ldots \mathbf{x}_m$ that are **i.i.d.** (this may or may not include the class label):
- Find a set of parameters θ such that the data can be summarized by a probability $P(\mathbf{x}|\theta)$
- θ depends on the family of probability distributions we consider (e.g. multinomial, Gaussian etc.)
- For regression and supervised methods, e have special target variables and we are interested in $P(y|\mathbf{x}, \mathbf{w})$

Maximum likelihood fitting

- Let D be the data set (all the instances)
- The *likelihood of parameter set θ given dataset D* is defined as:

$$L(\theta|D) = P(D|\theta)$$

- We derived this in lecture 1 from Bayes theorem, *assuming a uniform prior over instances*
- If the instances are i.i.d., we have:

$$L(\theta|D) = P(D|\theta) = \prod_{j=1}^{m} P(\mathbf{x}_j|\theta)$$

- E.g. in coin tossing, the likelihood of a parameter θ given the sequence $D = H, T, H, T, T$ is:

$$L(\theta|D) = \theta(1-\theta)\theta(1-\theta)(1-\theta) = \theta^{N_H}(1-\theta)^{N_T}$$

- Standard trick: maximize $\log L(\theta|D)$ instead!

$$\log L(\theta|D) = \sum_{i=1}^{m} \log P(x_j|\theta)$$

- To maximize, we take the derivatives of this function with respect to θ and set them to 0

Sufficient statistics

- To compute the likelihood in the coin tossing example, we only need to know N_H and N_T (number of heads and tails)
- We say that N_H and N_T are *sufficient statistics* for the binomial distribution
- In general, a sufficient statistic of the data is a function of the data that summarizes enough information to compute the likelihood
- Formally, $s(D)$ is a sufficient statistic if, for any two data sets D and D',

$$s(D) = s(D') \Rightarrow L(\theta|D) = L(\theta|D')$$

MLE applied to the binomial data

- The likelihood is:

$$L(\theta|D) = \theta^{N_H}(1-\theta)^{N_T}$$

- The log likelihood is:

$$\log L(\theta|D) = N_H \log \theta + N_T \log(1-\theta)$$

- Take the derivative of the log likelihood and set it to 0:

$$\frac{\partial}{\partial\theta} \log L(\theta|D) = \frac{N_H}{\theta} + \frac{N_T}{1-\theta}(-1) = 0$$

- Solving this gives

$$\theta = \frac{N_H}{N_H + N_T}$$

- This is intuitively appealing!

MLE for multinomial distribution

- Suppose that instead of tossing a coin, we roll a K-faced die
- The set of parameters in this case is $P(k) = \theta_k, k = 1, \ldots K$
- We have the additional constraint that $\sum_{k=1}^{K} \theta_k = 1$
- What is the log likelihood in this case?

$$\log L(\theta|D) = \sum_k N_k \log \theta_k$$

  where N_k is the number of times value k appears in the data
- We want to maximize the likelihood, but now this is a *constrained* optimization problem

Lagrange multipliers at work

- We can re-write our problem as maximizing:

$$\sum_k N_k \log \theta_k + \lambda \left(1 - \sum_k \theta_k\right)$$

- By taking the derivatives wrt θ_k and setting them to 0 we get $N_k = \lambda \theta_k$
- By summing over k and imposing the condition that $\sum_k \theta_k = 1$ we get $\lambda = \sum_k N_k$
- Hence, the best parameters are given by the "empirical frequencies":

$$\hat{\theta}_k = \frac{N_k}{\sum_k N_k}$$

Consistency of MLE

- For any estimator, we would like the parameters to converge to the "best possible" values as the number of examples grows

  We need to define "best possible" for probability distributions

- Let p and q be two probability distributions over X. The *Kullback-Leibler divergence* between p and q is defined as:

$$KL(p,q) = \sum_x P(x) \log \frac{P(x)}{q(x)}$$

A very brief detour into information theory

- Suppose I want to send some data over a noisy channel
- I have 4 possible values that I could send (e.g. A,C,G,T) and I want to encode them into bits such as to have short messages.
- Suppose that all values are equally likely. What is the best encoding?

A very brief detour into information theory (2)

- Now suppose I know A occurs with probability 0.5, C and G with probability 0.25 and T with probability 0.125. What is the best encoding?
- What is the expected length of the message I have to send?

Optimal encoding

- Suppose that I am receiving messages from an alphabet of m letters, and letter j has probability p_j
- The optimal encoding (by Shannon's theorem) will give $-\log_2 p_j$ bits to letter j
- So the expected message length if I used the optimal encoding will be equal to the **entropy** of p:

$$-\sum_j p_j \log_2 p_j$$

Interpretation of KL divergence

- Suppose now that letters would be coming from p but I don't know this. Instead, I believe letters are coming from q, and I use q to make the optimal encoding.
- The expected length of my messages will be $-\sum_j p_j \log_2 q_j$
- The amount of bits I waste with this encoding is:

$$-\sum_j p_j \log_2 q_j + \sum_j p_j \log_2 p_j = \sum_j p_j \log_2 \frac{p_j}{q_j} = KL(p, q)$$

Properties of MLE

- MLE is a **consistent estimator**, in the sense that (under a set of standard assumptions), w.p.1, we have:

$$\lim_{|D|\to\infty} \theta = \theta^*,$$

  where θ^* is the "best" set of parameters:

$$\theta^* = \arg\min_{\theta} KL(p^*(X), P(X|\theta))$$

  (p^* is the true distribution)
- With a small amount of data, the variance may be high (what happens if we observe just one coin toss?)

Prediction as inference

$$\begin{aligned} P(x_{n+1}|x_1, \ldots x_n) &= \int P(x_{n+1}|\theta, x_1, \ldots x_n) P(\theta|x_1, \ldots x_n) d\theta \\ &= \int P(x_{n+1}|\theta) P(\theta|x_1, \ldots x_n) d\theta, \end{aligned}$$

where

$$P(\theta|x_1, \ldots x_n) = \frac{P(x_1, \ldots x_n|\theta) P(\theta)}{P(x_1 \ldots x_n)}$$

Note that $P(x_1 \ldots x_n)$ is just a normalizing factor and $P(x_1, \ldots x_n|\theta) = L(\theta|D)$.

Example: Binomial data

- Suppose we observe 1 toss, $x_1 = H$. What would the MLE be?
- In the Bayesian approach,

$$P(\theta|x_1, \ldots x_n) \propto P(x_1, \ldots x_n|\theta)P(\theta)$$

- Assume we have a uniform prior for $\theta \in [0, 1]$, so $P(\theta) = 1$ (remember that θ is a continuous variable!)
- Then we have:

$$\begin{aligned} P(x_2 = H|x_1 = H) &\propto \int_0^1 P(x_1 = H|\theta)P(\theta)P(x_2 = H|\theta)d\theta \\ &= \int_0^1 \theta \cdot 1 \cdot \theta = \frac{1}{3} \end{aligned}$$

Example (continued)

- Likewise, we have:

$$\begin{aligned} P(x_2 = T|x_1 = H) &\propto \int_0^1 P(x_1 = H|\theta)P(\theta)P(x_2 = T|\theta)d\theta \\ &= \int_0^1 \theta \cdot 1 \cdot (1-\theta) = \frac{1}{6} \end{aligned}$$

- By normalizing we get:

$$\begin{aligned} P(x_2 = H|x_1 = H) &= \frac{\frac{1}{3}}{\frac{1}{3} + \frac{1}{6}} = \frac{2}{3} \\ P(x_2 = T|x_1 = H) &= \frac{1}{3} \end{aligned}$$

- It is as if we had our original data, plus two more tosses! (one heads, one tails)

Prior knowledge

- The prior incorporates prior knowledge or beliefs about the parameters
- As data is gathered, these beliefs do not play a significant role anymore
- More specifically, if the prior is well-behaved (does not assign 0 probability to feasible parameter values), MLE and Bayesian approach both give consistent estimators, so they converge in the limit to the same answer
- But the MLE and Bayesian predictions typically differ after fixed amounts of data. But in the short run, the prior can impact the **speed** of learning!

Multinomial distribution

- Suppose that instead of a coin toss, we have a discrete random variable with $k > 2$ possible values. We want to learn parameters $\theta_1, \ldots \theta_k$.
- The number of times each outcome is observed, $N_1, \ldots N_k$ represent sufficient statistics, and the likelihood function is:

$$L(\theta|D) = \prod_{i=1}^{k} \theta_i^{N_i}$$

- The MLE is, as expected,

$$\theta_i = \frac{N_i}{N_1 + \cdots + N_k}, \forall i = 1, \ldots k$$

Dirichlet priors

- A **Dirichlet prior** with parameters $\beta_1, \ldots \beta_k$ is defined as:

$$P(\theta) = \alpha \prod \theta_i^{\beta_i - 1}$$

- Then the posterior will have the same form, with parameter $\beta_i + N_i$:

$$P(\theta|D) = P(\theta)P(D|\theta) = \alpha \prod \theta_i^{\beta_i - 1 + N_i}$$

- We can compute the prediction of a new event in closed form:

$$P(x_{n+1} = k|D) = \frac{\beta_k + N_k}{\sum(\beta_i + N_i)}$$

Conjugate families

- The property that the posterior distribution follows the same parametric form as the prior is called **conjugacy**

  E.g. the Dirichlet prior is a conjugate family for the multinomial likelihood

- **Conjugate families** are useful because:
  - They can be represented in closed form
  - Often we can do on-line, incremental updates to the parameters as data is gathered
  - Often there is a closed-form solution for the prediction problem

Prior knowledge and Dirichlet priors

- The parameters β_i can be thought of a "imaginary counts" from prior experience
- The **equivalent sample size** is $\beta_1 + \cdots + \beta_k$
- The magnitude of the equivalent sample size indicates how confident we are in your priors
- The larger the equivalent sample size, the more real data items it will take to wash out the effect of the prior knowledge

The anatomy of the error of an estimator

- Suppose we have examples $\langle \mathbf{x}, y \rangle$ where $y = f(\mathbf{x}) + \epsilon$ and ϵ is Gaussian noise with zero mean and standard deviation σ
- We fit a linear hypothesis $h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$, such as to minimize sum-squared error over the training data:

$$\sum_{i=1}^{m} (y_i - h(\mathbf{x}_i))^2$$

- Because of the hypothesis class that we chose (hypotheses linear in the parameters) for some target functions f we will have a *systematic prediction error*
- Even if f were truly from the hypothesis class we picked, depending on the data set we have, the parameters $\mathbf{w}$ that we find may be different; this *variability* due to the specific data set on hand is a different source of error

Bias-variance analysis

- Given a new data point $\mathbf{x}$, what is the *expected prediction error*?
- Assume that the data points are drawn *independently and identically distributed (i.i.d.)* from a unique underlying probability distribution $P(\langle \mathbf{x}, y \rangle) = P(\mathbf{x})P(y|\mathbf{x})$
- The goal of the analysis is to compute, for an arbitrary given point $\mathbf{x}$,

$$E_P\left[(y - h(\mathbf{x}))^2 | \mathbf{x}\right]$$

  where y is the value of $\mathbf{x}$ in a data set, and the expectation is over all training sets of a given size, drawn according to P
- For a given hypothesis class, we can also compute the *true error*, which is the expected error over the input distribution:

$$\sum_{\mathbf{x}} E_P\left[(y - h(\mathbf{x}))^2 | \mathbf{x}\right] P(\mathbf{x})$$

  (if $\mathbf{x}$ continuous, sum becomes integral with appropriate conditions).
- We will decompose this expectation into three components

Recall: Statistics 101

- Let X be a random variable with possible values $x_i, i = 1 \ldots n$ and with probability distribution $P(X)$
- The *expected value* or *mean* of X is:

$$E[X] = \sum_{i=1}^{n} x_i P(x_i)$$

- If X is continuous, roughly speaking, the sum is replaced by an integral, and the distribution by a density function
- The *variance* of X is:

$$\begin{aligned} Var[X] &= E[(X - E(X))^2] \\ &= E[X^2] - (E[X])^2 \end{aligned}$$

The variance lemma

$$
\begin{aligned}
Var[X] &= E[(X - E[X])^2] \\
&= \sum_{i=1}^{n} (x_i - E[X])^2 P(x_i) \\
&= \sum_{i=1}^{n} (x_i^2 - 2x_i E[X] + (E[X])^2) P(x_i) \\
&= \sum_{i=1}^{n} x_i^2 P(x_i) - 2E[X] \sum_{i=1}^{n} x_i P(x_i) + (E[X])^2 \sum_{i=1}^{n} P(x_i) \\
&= E[X^2] - 2E[X]E[X] + (E[X])^2 \cdot 1 \\
&= E[X^2] - (E[X])^2
\end{aligned}
$$

We will use the form:

$$E[X^2] = (E[X])^2 + Var[X]$$

Bias-variance decomposition

- Simple algebra:

$$\begin{aligned} E_P\left[(y - h(\mathbf{x}))^2 | \mathbf{x}\right] &= E_P\left[(h(\mathbf{x}))^2 - 2yh(\mathbf{x}) + y^2 | \mathbf{x}\right] \\ &= E_P\left[(h(\mathbf{x}))^2 | \mathbf{x}\right] + E_P\left[y^2 | \mathbf{x}\right] - 2E_P[y|\mathbf{x}]E_P\left[h(\mathbf{x})|\mathbf{x}\right] \end{aligned}$$

- Let $\bar{h}(\mathbf{x}) = E_P[h(\mathbf{x})|\mathbf{x}]$ denote the *mean prediction* of the hypothesis at $\mathbf{x}$, when h is trained with data drawn from P
- For the first term, using the variance lemma, we have:

$$E_P[(h(\mathbf{x}))^2|\mathbf{x}] = E_P[(h(\mathbf{x}) - \bar{h}(\mathbf{x}))^2|\mathbf{x}] + (\bar{h}(\mathbf{x}))^2$$

- Note that $E_P[y|\mathbf{x}] = E_P[f(\mathbf{x}) + \epsilon|\mathbf{x}] = f(\mathbf{x})$ (because of linearity of expectation and the assumption on $\epsilon \sim \mathcal{N}(0, \sigma)$)
- For the second term, using the variance lemma, we have:

$$E[y^2|\mathbf{x}] = E[(y - f(\mathbf{x}))^2|\mathbf{x}] + (f(\mathbf{x}))^2$$

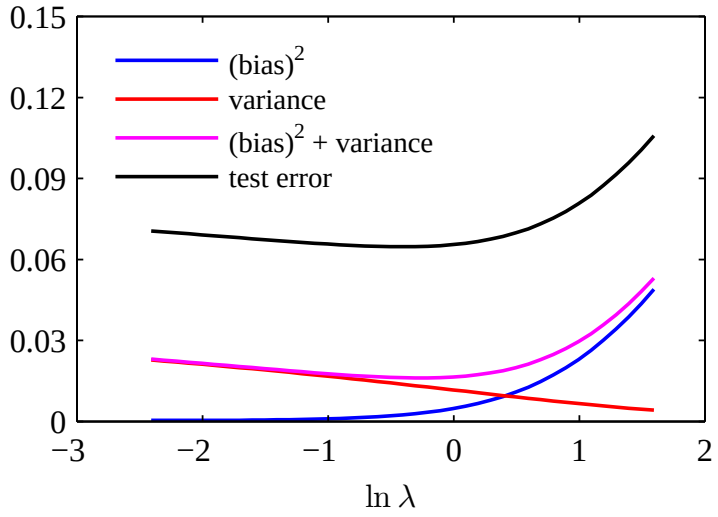
Bias-variance decomposition (2)

- Putting everything together, we have:

$$\begin{aligned} E_P\left[(y - h(\mathbf{x}))^2 | \mathbf{x}\right] &= E_P[(h(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 | \mathbf{x}] + (\bar{h}(\mathbf{x}))^2 - 2f(\mathbf{x})\bar{h}(\mathbf{x}) \\ &+ E_P[(y - f(\mathbf{x}))^2 | \mathbf{x}] + (f(\mathbf{x}))^2 \\ &= E_P[(h(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 | \mathbf{x}] + (f(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 \\ &+ E[(y - f(\mathbf{x}))^2 | \mathbf{x}] \end{aligned}$$

- The first term, $E_P[(h(\mathbf{x}) - \bar{h}(\mathbf{x}))^2 | \mathbf{x}]$, is the *variance* of the hypothesis h at $\mathbf{x}$, when trained with finite data sets sampled randomly from P
- The second term, $(f(\mathbf{x}) - \bar{h}(\mathbf{x}))^2$, is the *squared bias* (or systematic error) which is associated with the class of hypotheses we are considering
- The last term, $E[(y - f(\mathbf{x}))^2 | \mathbf{x}]$ is the *noise*, which is due to the problem at hand, and cannot be avoided

Error decomposition



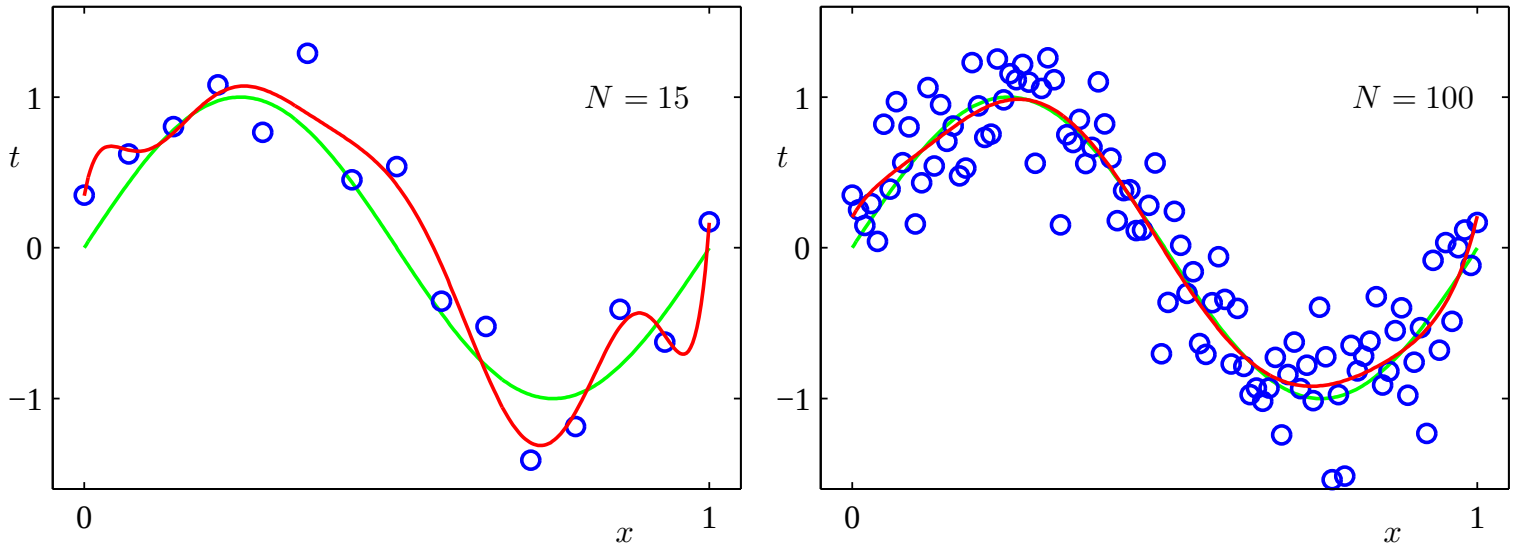
- The bias-variance sum approximates well the test error over a set of 1000 points
- x-axis measures the hypothesis complexity (decreasing left-to-right)
- Simple hypotheses usually have high bias (bias will be high at many points, so it will likely be high for many possible input distributions)
- Complex hypotheses have high variance: the hypothesis is very dependent on the data set on which it was trained.

Bias-variance trade-off

- Typically, bias comes from not having good hypotheses in the considered class
- Variance results from the hypothesis class containing "too many" hypotheses
- MLE estimation is typically unbiased, but has high variance
- Bayesian estimation is biased, but typically has lower variance
- Hence, we are faced with a *trade-off*: choose a more expressive class of hypotheses, which will generate higher variance, or a less expressive class, which will generate higher bias
- Making the trade-off has to depend on the amount of data available to fit the parameters (data usually mitigates the variance problem)

More on overfitting

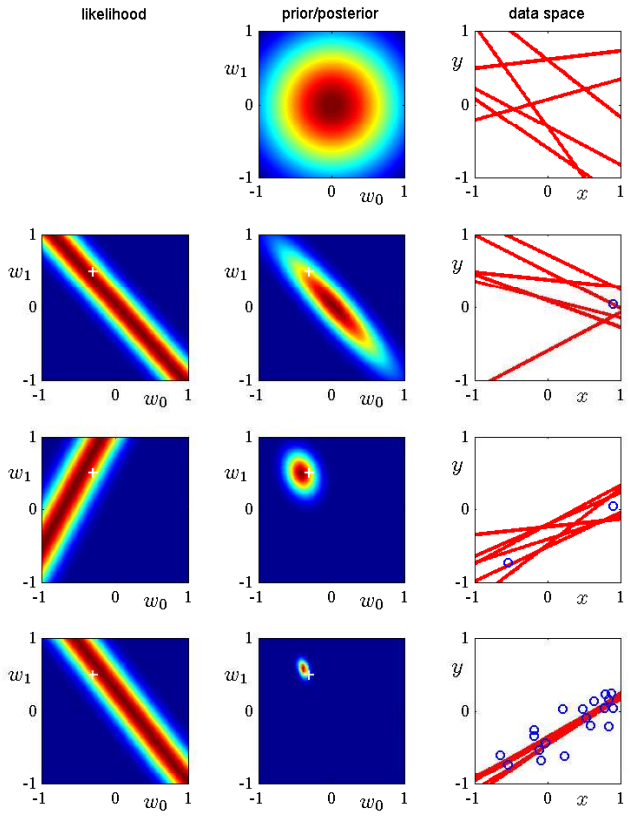
- Overfitting depends on the amount of data, relative to the complexity of the hypothesis
- With more data, we can explore more complex hypotheses spaces, and still find a good solution



Bayesian view of regularization

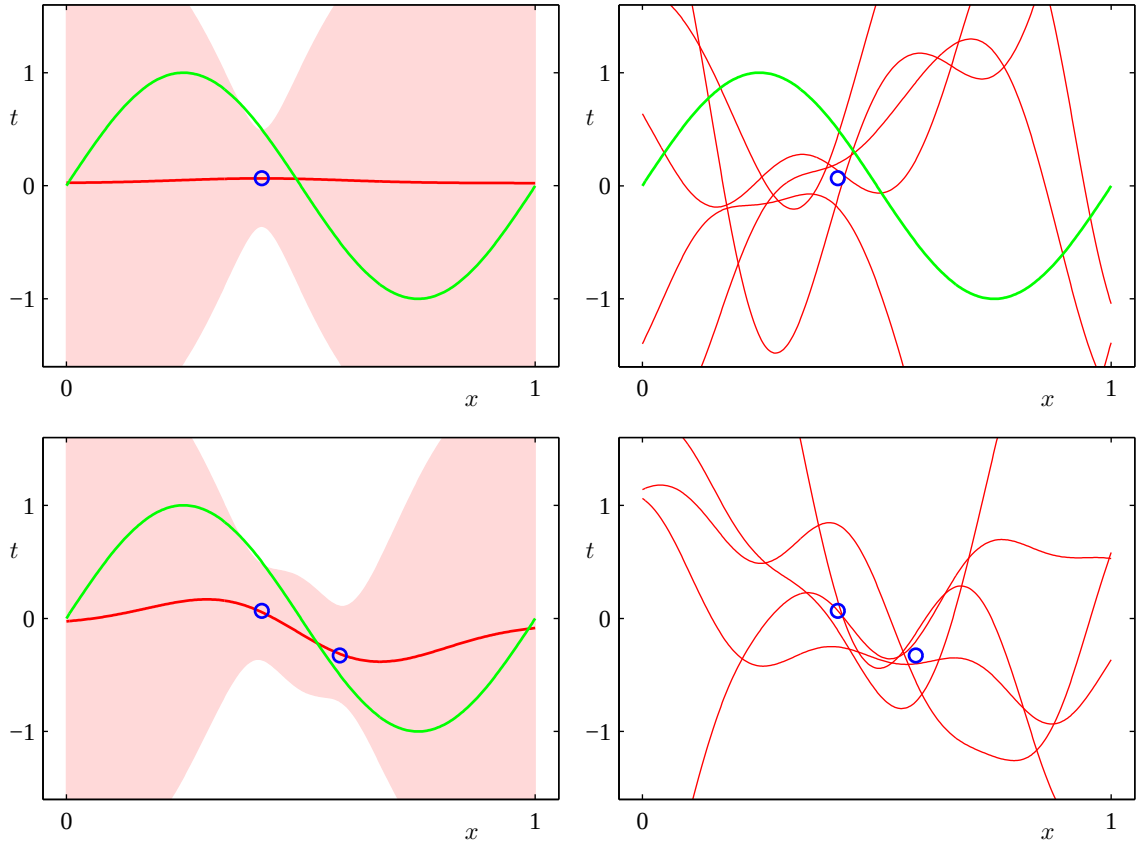
- Start with a *prior distribution* over hypotheses
- As data comes in, compute a *posterior distribution*
- We often work with *conjugate priors*, which means that when combining the prior with the likelihood of the data, one obtains the posterior in the same form as the prior
- Regularization can be obtained from particular types of prior (usually, priors that put more probability on simple hypotheses)
- E.g. L_2 regularization can be obtained using a circular Gaussian prior for the weights, and the posterior will also be Gaussian
- E.g. L_1 regularization uses double-exponential prior (see (Tibshirani, 1996))

Bayesian view of regularization



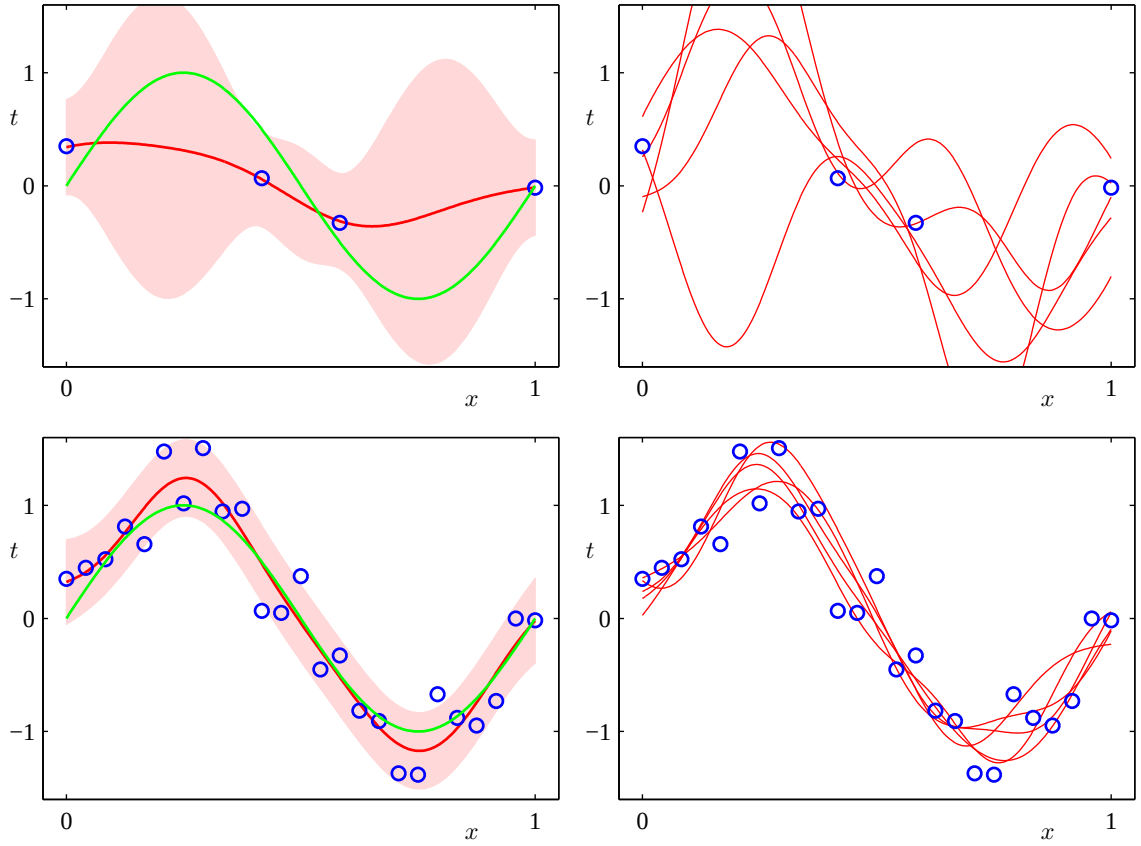
- Prior is round Gaussian
- Posterior will be skewed by the data

What does the Bayesian view give us?



- Circles are data points
- Green is the true function
- Red lines on right are drawn from the posterior distribution

What does the Bayesian view give us?



- Functions drawn from the posterior can be very different
- Uncertainty decreases where there are data points

What does the Bayesian view give us?

- Uncertainty estimates, i.e. how sure we are of the value of the function
- These can be used to guide active learning: ask about inputs for which the uncertainty in the value of the function is very high
- In the limit, Bayesian and maximum likelihood learning converge to the same answer
- In the short term, one needs a good prior to get good estimates of the parameters
- Sometimes the prior is overwhelmed by the data likelihood too early.
- Using the Bayesian approach does NOT eliminate the need to do cross-validation in general
- More on this later...