Modeling Data with Linear Combinations of Basis Functions

Read Chapter 3 in the text by Bishop

A Type of Supervised Learning Problem

We want to model data $(x_1, t_1), \ldots, (x_N, t_N)$, where x_i is a vector of D inputs (predictors) for case i, and t_i is the target (response) variable for case i, which is real-valued.

We are trying to predict t from x, for some future test case, but we are not trying to model the distribution of x.

Suppose also that we don't expect the best predictor for t to be a linear function of x, so ordinary linear regression on the original variables won't work well.

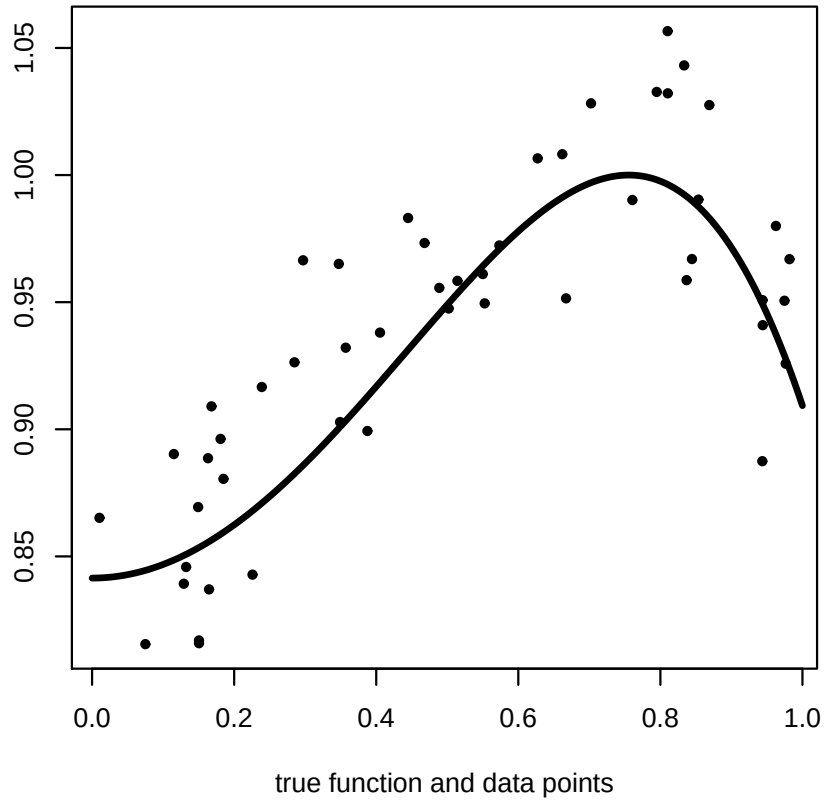
We need to allow for a non-linear function of x, but we don't have any theory that says what form this function should take. What to do?

An Example Problem

As an illustration, we can use the synthetic data set looked at last lecture — 50 points generated with x uniform from $(0, 1)$ and y set by the formula:

$$y \;=\; \sin(1 + x^2) \;+\; \text{noise}$$

where the noise has $N(0, 0.03^2)$ distribution.



The noise-free true function, $\sin(1 + x^2)$, is shown by the line.

This is simpler than real machine learning problems, but lets us look at plots...

Linear Basis Function Models

We earlier looked at fitting this data by least-squares linear regression, using not just x, but also x^2, x^3, etc., up to (say) x^4 as predictors.

This is an example of a *linear basis function model*.

In general, we do linear regression of t on $\phi_1(x)$, $\phi_2(x)$, ..., $\phi_{M-1}(x)$, where the ϕ_j are *basis functions*, that we have selected to allow for a non-linear function of x.

This gives the following model:

$$\begin{aligned} t &= y(x,w) + \text{noise} \\ y(x,w) &= w_0 + \sum_{j=1}^{M-1} w_j \phi_j(x) = w^T \phi(x) \end{aligned}$$

where w is the vector of all M regression coefficients (including the intercept, w_0) and $\phi(x)$ is the vector of all basis function values at input x, including $\phi_0(x) = 1$ for the intercept.

Maximum Likelihood Estimation

Suppose we assume that the noise in the regression model is Gaussian (normal) with mean zero and some variance σ^2. (The text uses β to denote $1/\sigma^2$.)

With this assumption, we can write down the *likelihood function* for the parameters w and σ, which is the joint probability density of all the targets in the training set as a function of w and σ:

$$\begin{aligned} L(w,\sigma) &= P(t_1,\ldots,t_N|x_1,\ldots,x_N,w,\sigma) \\ &= \prod_{i=1}^{N} N(t_i|w^T\phi(x_i),\sigma^2) \end{aligned}$$

where $N(t|\mu,\sigma^2)$ is the density for t under a normal distribution with mean μ and variance σ^2.

The maximum likelihood estimates for the parameters are the values of w and σ that maximize this likelihood. Equivalently, they maximize the log likelihood, which, ignoring terms that don't depend on w or σ, is

$$\log L(w,\sigma) = -N\log(\sigma) - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(t_i - w^T\phi(x_i))^2$$

Least Squares Estimation

From this log likelihood function, we see that regardless of what σ might be, the maximum likelihood estimate of w is the value that minimizes the sum of squared prediction errors over training cases.

Let's put the values of all basis functions in all training cases into a matrix Φ, with $\Phi_{ij} = \phi_j(x_i)$. Also, put the target values for all training cases into a vector t.

We can now write the sum of squared errors on training cases as

$$||t - \Phi w||^2 \quad = \quad (t - \Phi w)^T (t - \Phi w)$$

This is minimized for the value of w where its gradient is zero, which is where

$$-2\Phi^T (t - \Phi w) = 0$$

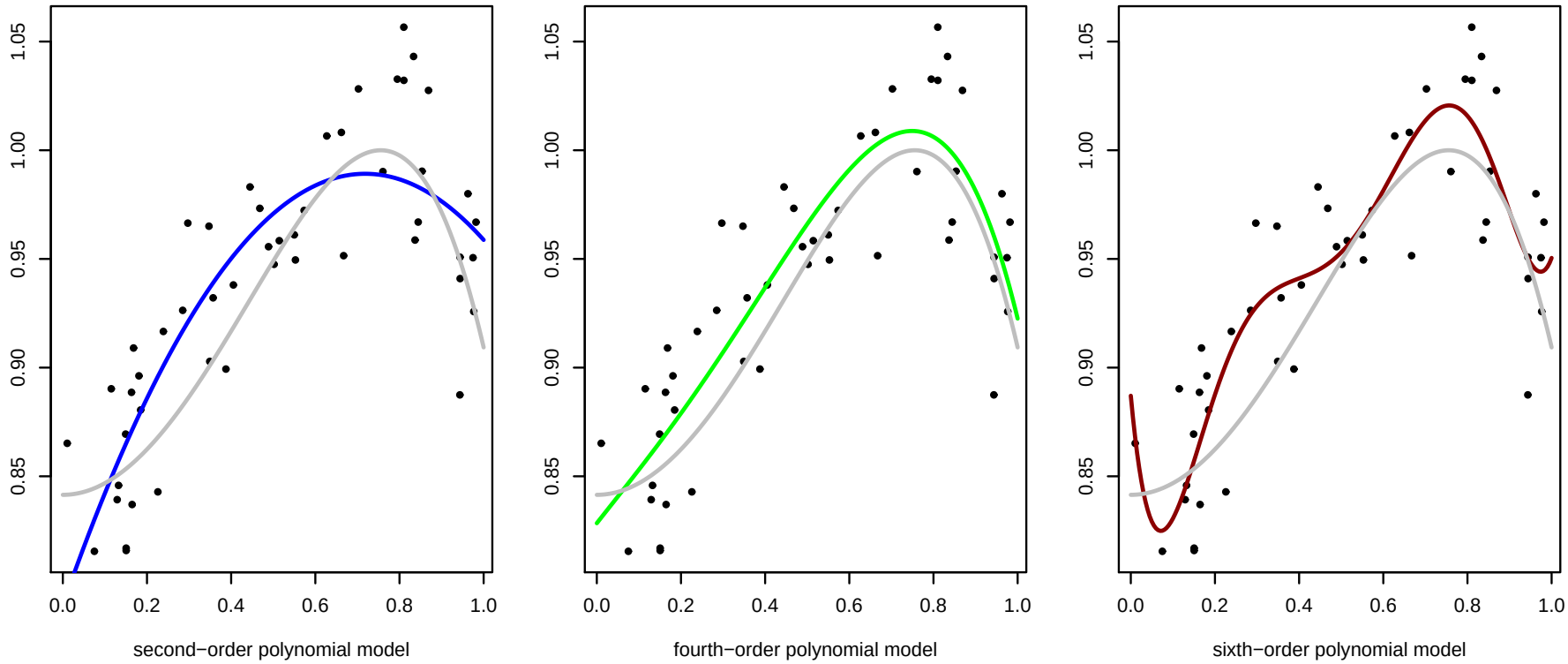
Solving this, the least squares estimate of w is

$$\hat{w} \quad = \quad (\Phi^T \Phi)^{-1} \Phi^T t$$

This assumes that $\Phi^T \Phi$ is non-singular, so that there is a unique solution.
When M is greater than N, this will not be the case!

Results with Polynomial Basis Functions

Recall we looked before at least-squares fits of polynomial models with increasing order, which can be viewed as basis function models with $\phi_j(x) = x^j$.



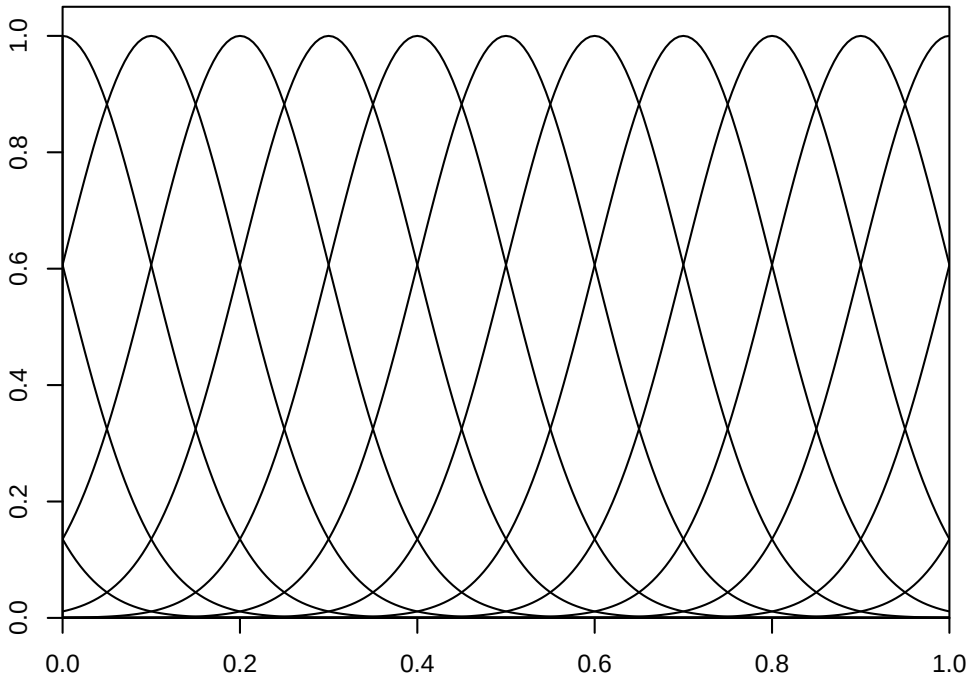
The gray line is the true noise-free function. We see that a second-order fit is too simple, but a sixth-order fit is too complex, producing "overfitting".

Gaussian Basis Functions

Polynomials are *global* basis functions, each affecting the prediction over the whole input space. Often, *local* basis functions are more appropriate. One possibility is to use functions proportional to Gaussian probability densities:

$$\phi_j(x) \quad = \quad \exp(-(x - \mu_j)^2/2s^2)$$

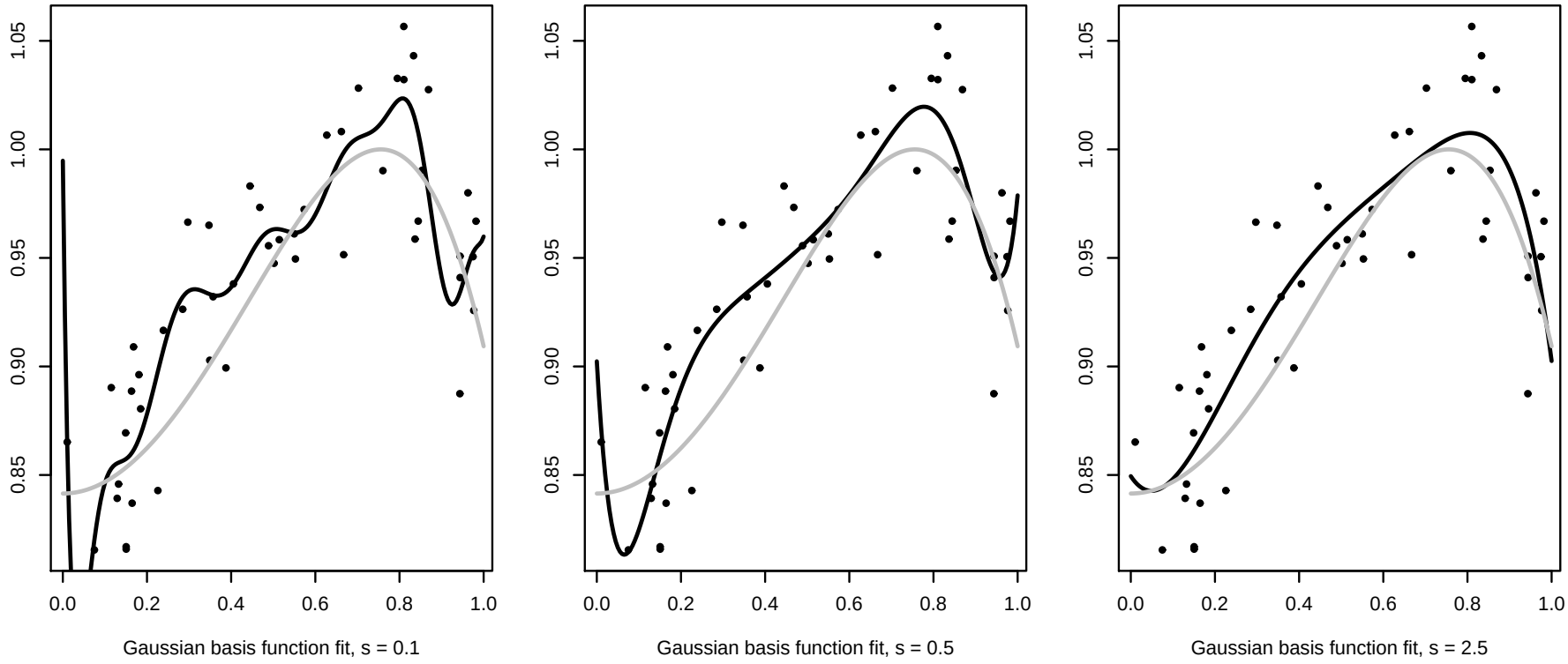
Here are these basis functions for $s = 0.1$, with the μ_j on a grid with spacing s:



Gaussian basis functions, s = 0.1

Results with Gaussian Basis Functions

Here are the results using Gaussian basis functions (plus $\phi_0(x) = 1$) on the example dataset, with varying width (and spacing) s:



The estimated values for the w_j are not what you might guess. For the middle model above, they are as follows:

6856.5 -3544.1 -2473.7 -2859.8 -2637.7 -2861.5 -2468.0 -3558.4