

# STATS 211: Statistical Methods II

## Lecture 4: Basis Expansions

Babak Shahbaba

UCI, Winter 2012

## Moving beyond linearity

- In this lecture, we discuss some strategies for relaxing the linearity assumption for the relationship between  $y$  and  $x$  in regression models.
- The main idea is to augment/replace the original input space  $x$  with a set of  $h_m(x)$ , for  $m = 1, \dots, M$ , where  $h_m(x) : \mathbb{R}^p \rightarrow \mathbb{R}$  is a transformation of  $x$ .
- This is referred to as *basis expansion* in  $x$ , and  $h_m(x)$  is called a *basis function*.
- Then, we model  $y$  as a linear function of  $h_1(x), \dots, h_M(x)$ ,

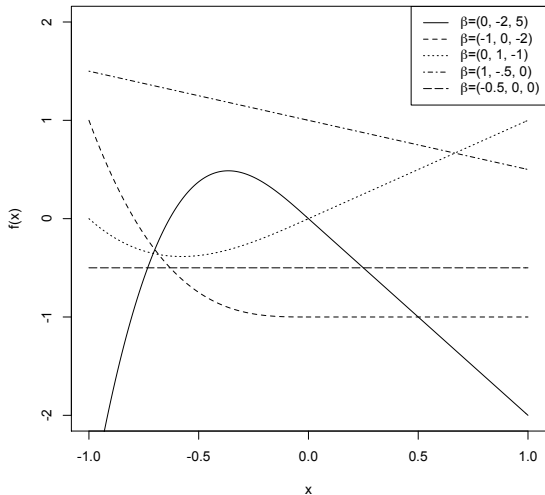
$$f(x) = \sum_{m=1}^M \beta_m h_m(x)$$

- Assignment: Show that  $h_1(x), \dots, h_M(x)$  form a vector space in  $\mathbb{R}^M$ . (That is why we use the term “basis”).

## Moving beyond linearity

- While the above model is in general nonlinear in  $x$ , it is linear in  $\beta$ . Therefore, we can still estimate model parameters using linear regression techniques after replacing the original variables,  $x_1, \dots, x_p$ , with the new set of variables,  $h_1(x), \dots, h_M(x)$ .
- In this approach, we can use any basis functions we want.
- For example, the following graph shows the plot of  $f(x)$  with  $h_1(x) = 1$ ,  $h_2(x) = x$ , and  $h_3(x) = \min(0, x^3)$  and different  $\beta$ 's.

## Moving beyond linearity



## Moving beyond linearity

- We can create very flexible models using the above approach. In practice, however, we impose some constraints on the class of functions in order to control model complexity.
- For example, we can restrict the basis functions such that each basis function depends on one variable only. This results in *additive models*:

$$\begin{aligned} f(x) &= \sum_{j=1}^p f_j(x_j) \\ &= \sum_{j=1}^p \sum_{m=1}^M \beta_{jm} h_{jm}(x_j) \end{aligned}$$

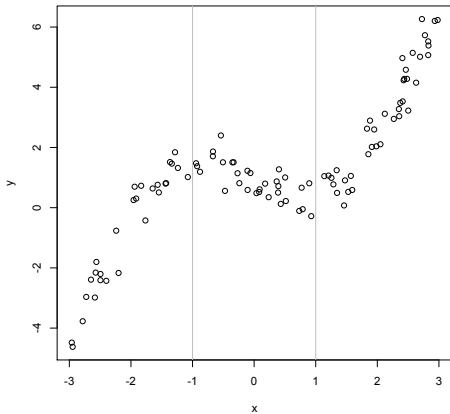
## Moving beyond linearity

- We can restrict  $f(x)$  to be *piecewise constant*, where each function is constant in some region and zero outside the region.
- Or *piecewise linear*, where each function is linear in some region and zero outside the region.
- Or *piecewise polynomial*, where each function is a polynomial of degree  $q$  in some region and zero outside the region.
- We start by discussing these models for univariate cases.

## Piecewise constant model

- We divide the domain of  $x$  into  $K + 1$  regions by specifying  $K$  *knots*,  $\xi_1, \dots, \xi_K$ .
- For each region, we specify a function that is constant within the region and zero outside the region.
- For the following example, we set  $\xi_1 = -1$  and  $\xi_2 = 1$ .

# Piecewise constant model



## Piecewise constant model

- We define the following three functions:

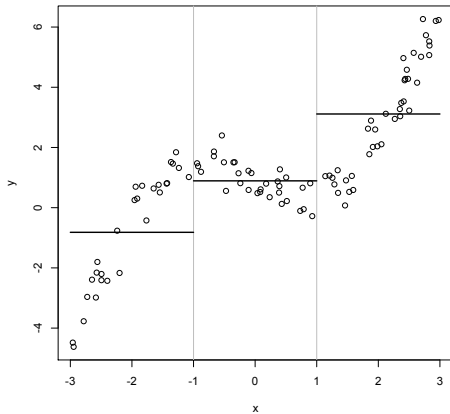
$$h_1(x) = I(x < \xi_1), \quad h_2(x) = I(\xi_1 \leq x < \xi_2), \quad h_3(x) = I(\xi_2 \leq x)$$

- The piecewise constant function model is then specified as follows:

$$f(x) = \sum_{m=1}^3 \beta_m h_m(x)$$

- In this case,  $\hat{\beta}_m = \hat{y}_m$ , i.e., the mean of the observed response values in the  $m^{th}$  region

# Piecewise constant model

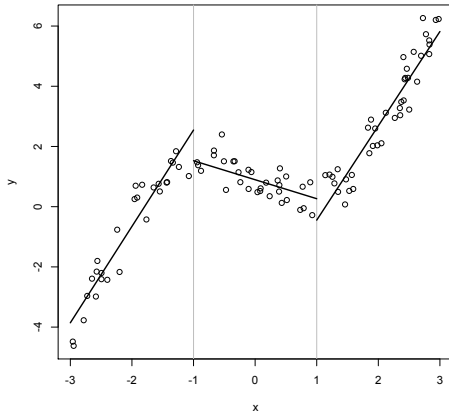


## Piecewise linear model

- Alternatively, we can allow the functions to be linear in each region.
- For this, we include the following three functions along with the previous three functions,  $h_1(x)$ ,  $h_2(x)$ , and  $h_3(x)$ :

$$h_4(x) = h_1(x)x, \quad h_5(x) = h_2(x)x, \quad h_6(x) = h_3(x)x$$

# Piecewise linear model



## Piecewise linear model

- While the model seems to fit the data well, its discontinuity at the knots should be addressed. We usually prefer models that are continuous, since we don't believe there would be a drastic change in  $y$  as we move, for example, from  $x = \xi_1^-$  to  $x = \xi_1^+$ .
- To make the above model continuous, we need to impose the following two constraints:

$$\beta_1 h_1(\xi_1) + \beta_4 h_4(\xi_1) = \beta_2 h_2(\xi_1) + \beta_5 h_5(\xi_1)$$

$$\beta_2 h_2(\xi_2) + \beta_5 h_5(\xi_2) = \beta_3 h_3(\xi_2) + \beta_6 h_6(\xi_2)$$

which means,

$$\beta_1 + \beta_4 \xi_1 = \beta_2 + \beta_5 \xi_1$$

$$\beta_2 + \beta_5 \xi_2 = \beta_3 + \beta_6 \xi_2$$

## Piecewise linear model

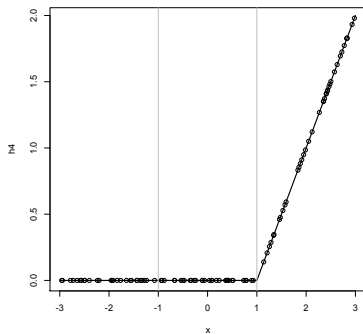
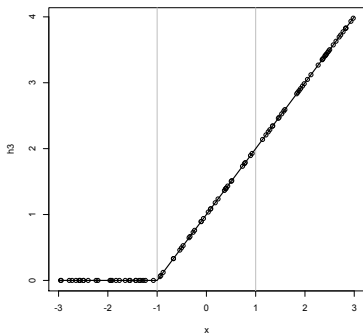
- Imposing the above two constraints reduces the dimensionality from 6 to 4.
- Alternatively, we can use the following four functions:

$$h_1(x) = 1, \quad h_2(x) = x, \quad h_3(x) = (x - \xi_1)_+, \quad h_4(x) = (x - \xi_2)_+$$

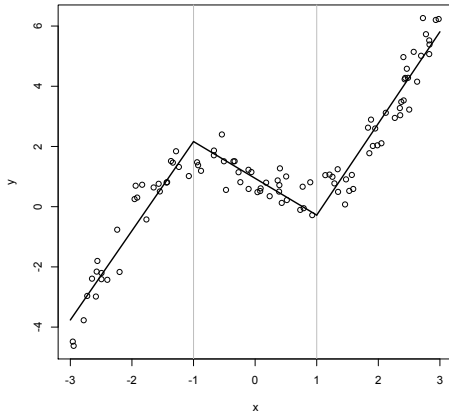
where  $(x - \xi)_+$  means  $\max(0, x - \xi)$ , i.e., the positive part of the function only.

- Next graph shows  $h_3$  and  $h_4$  functions with  $\xi_1 = -1$  and  $\xi_2 = 1$ .

# Piecewise linear model



# Piecewise linear model



## Piecewise polynomial model

- To make the model more flexible, we can include functions with higher degree.

$$h_7(x) = h_1(x)x^2, \quad h_8(x) = h_2(x)x^2, \quad h_9(x) = h_3(x)x^2$$

- As before, we impose two constraints so the function becomes continuous at the two knots.

$$\begin{aligned}\beta_1 h_1(\xi_1) + \beta_4 h_4(\xi_1) + \beta_7 h_7(\xi_1) &= \beta_2 h_2(\xi_1) + \beta_5 h_5(\xi_1) + \beta_8 h_8(\xi_1) \\ \beta_2 h_2(\xi_2) + \beta_5 h_5(\xi_2) + \beta_8 h_8(\xi_2) &= \beta_3 h_3(\xi_2) + \beta_6 h_6(\xi_2) + \beta_9 h_9(\xi_2)\end{aligned}$$

- Additionally, we can create *smoothness* by making the first derivatives continuous at the knots.

$$\begin{aligned}\beta_1 h'_1(\xi_1) + \beta_4 h'_4(\xi_1) + \beta_7 h'_7(\xi_1) &= \beta_2 h'_2(\xi_1) + \beta_5 h'_5(\xi_1) + \beta_8 h'_8(\xi_1) \\ \beta_2 h'_2(\xi_2) + \beta_5 h'_5(\xi_2) + \beta_8 h'_8(\xi_2) &= \beta_3 h'_3(\xi_2) + \beta_6 h'_6(\xi_2) + \beta_9 h'_9(\xi_2)\end{aligned}$$

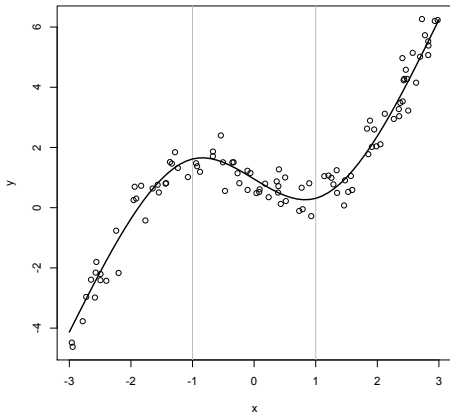
# Splines

- In general, we can use a piecewise polynomial of order  $M$  (with degree  $M - 1$ ) that is continuous and has continuous derivatives up to order  $M - 2$ . We refer to these models as *order- $M$  splines*.
- The continuous piecewise linear model discussed above is in fact an order-2 spline.
- In practice, we rarely need to go beyond order-4 splines, which are known as *cubic splines*.
- For cubic splines, the function, its first derivative, and its second derivative are all continuous at the knots.

## Natural cubic splines

- As shown in Fig. 5.3 of the book by Hastie et. al. (2010), the behavior of splines are erratic (high variance) near and beyond the boundaries.
- To address this issue, we use *natural cubic spline*, which forces the function to be linear beyond the boundary knots.
- Therefore, the second and third derivatives are zero at the boundary knots.

# Natural cubic splines



## Natural cubic splines

- In general, piecewise polynomials of order  $M$  and  $K$  knots and no constraint have  $M(K + 1)$  basis functions.
- Natural cubic splines with  $K$  knots have  $K$  basis functions:  $4(K + 1) - 3K - 4$ . (Three constraints at each knot and two additional constraints at each boundary knot.)
- Stone (1986) has found that in general, the location of knots is not that crucial for natural cubic splines. We can put the knots at equally spaced quantiles for example. (See Harrell, 2001).
- A more important factor is the number of knots:  $K$ .

## Natural cubic splines

- How should we decide the number of knots  $K$ ?
- For large  $K$ , the variance is high, and small values of  $K$  might result in a large bias.
- We could of course find a balance between bias and variance by finding a good value for  $K$  using, for example, cross validation or AIC.

## Smoothing splines

- Alternatively, we could use *smoothing splines*, where model complexity is controlled by regularization similar to ridge regression.
- For these models, we minimize the following penalized residual sum of squares:

$$PRSS(f, \lambda) = \sum_{i=1}^n [y_i - f(x_i)]^2 + \lambda \int [f''(t)]^2 dt$$

where  $f(x)$  is any function with second derivatives, and  $\lambda$  is a fixed smoothing parameter.

- Setting  $\lambda = \infty$  results in a simple linear regression model with least squares estimates.
- Setting  $\lambda = 0$  will force the model to pass through all the observed data points.
- We can choose the right  $\lambda$  using cross validation.

## Smoothing splines

- It turns out that the natural cubic spline with knots at unique observed values of  $x$  (i.e., with  $K = n$  knots) is the unique minimizer of the above function (Green and Silverman, 1994).
- Therefore, the solution can be written as follows:

$$f(x) = \sum_{j=1}^n N_j(x)\theta_j$$

where  $N_j(x)$  are an  $n$ -dimensional basis function.

- Analogous to ridge regression, the estimate of  $\theta$  is (see the book for details)

$$\hat{\theta} = (N^T N + \lambda \Omega_N)^{-1} N^T y$$

where  $\{\Omega_N\}_{jk} = \int N_j''(t) N_k''(t) dt$

## Smoothing splines

- Using the above parameter estimates, we predict the value of the response variable for future observations as follows:

$$\hat{f}(x) = \sum_{j=1}^n N_j(x) \hat{\theta}_j$$

- For the training cases, the fitted values are

$$\begin{aligned}\hat{f} &= N(N^T N + \lambda \Omega_N)^{-1} N^T y \\ &= S_\lambda y\end{aligned}$$

- We refer to  $S_\lambda$  as the *smoother matrix*, which is analogous to the hat matrix,  $H$ , for linear regression models.

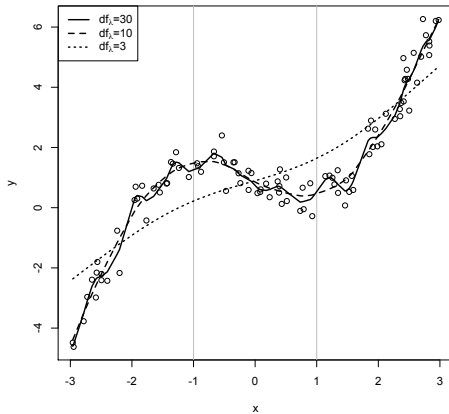
## Smoothing splines

- For linear regression models, where  $x$  is an  $n \times p$  matrix,  $\text{trace}(H) = p$  gives the number of model parameters (including the intercept).
- By analogy, the *effective degrees of freedom* for a smoothing spline is defined as

$$\text{df}_\lambda = \text{trace}(S_\lambda)$$

- Therefore, instead of working with  $\lambda$  to control complexity, it is more intuitive to work with  $\text{df}_\lambda$ .
- The following graph shows three smoothing splines with  $\text{df}_\lambda = 30$ ,  $\text{df}_\lambda = 10$ , and  $\text{df}_\lambda = 3$ .

# Smoothing splines



## Multidimensional splines

- When we have more than one variable, we can restrict our model to be additive,

$$\begin{aligned} f(x) &= \beta_0 + \sum_{j=1}^p f_j(x_j) \\ &= \sum_{j=1}^p \sum_{m=1}^{M_j} \beta_{jm} h_{jm}(x_j) \end{aligned}$$

- Note that we only have one intercept,  $\beta_0$ , in the model.
- To fit this model, we can create one big matrix of basis functions such that the first column is all ones, the next  $M_1$  columns are the basis functions for the first variable, the next  $M_2$  columns are the basis functions for the second variable, and so forth.
- To include interaction terms, we can use *tensor product basis* functions instead (see Section 5.7).