Basis Functions and p-dimensional Inputs: Tensor Product Bases

Stephen Vardeman

Analytics Iowa LLC

ISU Statistics and IMSE

Tensor product bases

If $p = 2$ ($\mathbf{x} = (x_1, x_2)$ takes values in $\Re^2$) one might proceed as follows. For $\{h_{11}, h_{12}, \ldots, h_{1M_1}\}$ a set of spline basis functions of x_1 and $\{h_{21}, h_{22}, \ldots, h_{2M_2}\}$ a set of spline basis functions of x_2 one might consider the set of $M_1 \cdot M_2$ basis functions based on $\mathbf{x}$ defined by

$$g_{jk}(\mathbf{x}) = h_{1j}(x_1)\, h_{2k}(x_2)$$

and corresponding forms for regression splines

$$f(\mathbf{x}) = \sum_{j,k} \beta_{jk} g_{jk}(\mathbf{x})$$

This potential method suffers the explosion in the size of a tensor product basis as p increases. For example, using K knots for cubic regression splines in each of p dimensions produces $(4 + K)^p$ basis functions for the p-dimensional problem.

Penalization and MARS

Some kind of shrinking of coefficients or forward selection algorithm is needed to produce any kind of workable fits with the huge numbers of basis functions in tensor product bases.

The multivariate smoothing routines provided in the `mgcv` `R` package of Wood allow for quadratic penalized (ridge-regression-type) fitting involving tensor product bases.

The following discussion of "MARS" concerns one kind of forward selection algorithm using (data-dependent) linear regression spline basis functions and products of them for building predictors

MARS and hockey stick functions

MARS (multivariate adaptive regression splines) is based on use of data-dependent "hockey-stick" or "hinge" functions (the kind of functions leading to piece-wise linear regression splines when $p = 1$) and their products as (data-dependent) "basis functions." That is, with input space $\Re^p$ consider *data-dependent* basis functions built on the Np pairs of functions

$$h_{ij1}(\mathbf{x}) = (x_j - x_{ij})_+ \quad \text{and} \quad h_{ij2}(\mathbf{x}) = (x_{ij} - x_j)_+ \tag{1}$$

(x_{ij} is the jth coordinate of the ith input training vector and both $h_{ij1}(\mathbf{x})$ and $h_{ij2}(\mathbf{x})$ depend on $\mathbf{x}$ only through the jth coordinate of $\mathbf{x}$). MARS builds predictors sequentially, making use of these "reflected pairs" represented on the next slide.

Hockey stick functions

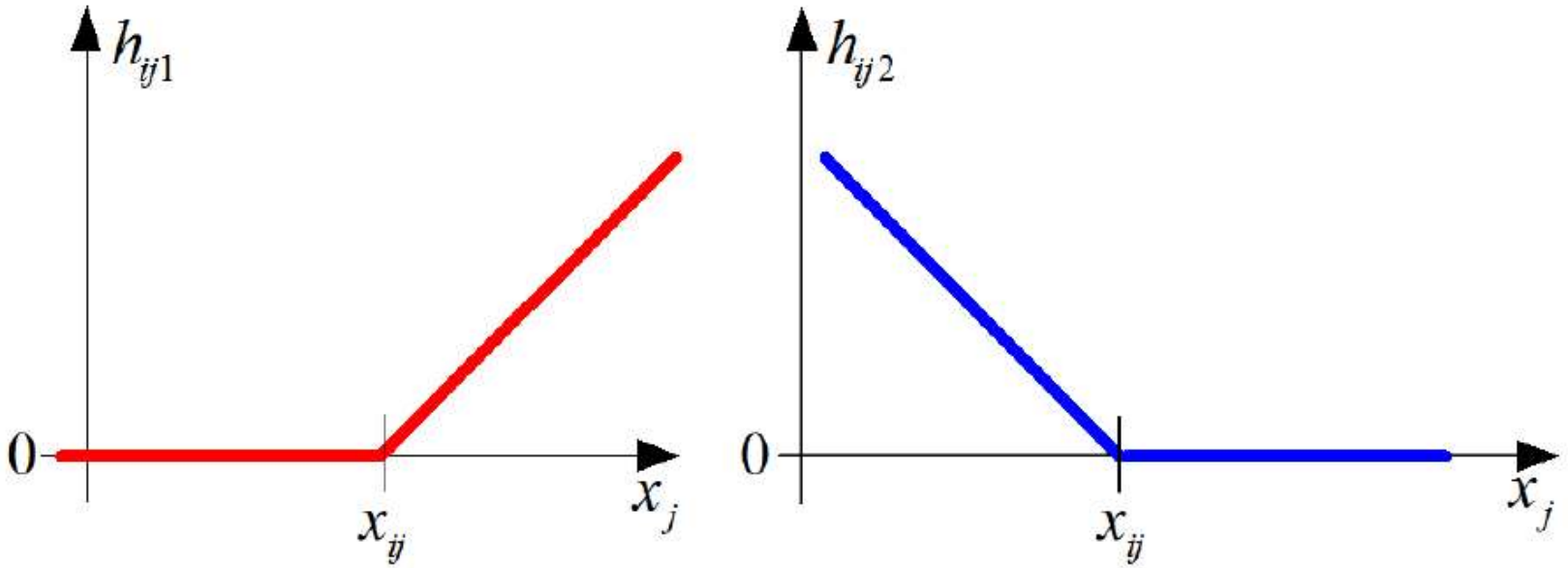

Figure: A "reflected pair" of "hockey-stick" functions represented as depending upon x_j (and constant in all other inputs).

MARS sequential model building

MARS goes roughly as follows:

1. Identify a pair (1) so that

$$\beta_0 + \beta_{11} h_{ij1}(\mathbf{x}) + \beta_{12} h_{ij2}(\mathbf{x})$$

has the best *SSE* possible. Call the selected functions

$$g_{11} = h_{ij1} \text{ and } g_{12} = h_{ij2}$$

and set

$$\hat{f}_1(\mathbf{x}) = \hat{\beta}_0 + \hat{\beta}_{11} g_{11}(\mathbf{x}) + \hat{\beta}_{12} g_{12}(\mathbf{x})$$

MARS sequential Model building cont.

2. At stage l of the predictor-building process, with predictor

$$\hat{f}_{l-1}(\mathbf{x}) = \hat{\beta}_0 + \sum_{m=1}^{l-1} \left(\hat{\beta}_{m1} g_{m1}(\mathbf{x}) + \hat{\beta}_{m2} g_{m2}(\mathbf{x}) \right)$$

in hand, consider for addition to the model, pairs of basis functions that are either of the basic form (1) or of the form

$$h_{ij1}(\mathbf{x})\, g_{m1}(\mathbf{x}) \text{ and } h_{ij2}(\mathbf{x})\, g_{m1}(\mathbf{x})$$

or of the form

$$h_{ij1}(\mathbf{x})\, g_{m2}(\mathbf{x}) \text{ and } h_{ij2}(\mathbf{x})\, g_{m2}(\mathbf{x})$$

for some $m < l$, subject to the constraints that no x_j appears in any candidate product more than once (maintaining the piece-wise linearity of sections of the predictor).

MARS sequential model building cont.

Additionally, one may decide to put an upper limit on the order of the products considered for inclusion in the predictor. The best candidate pair in terms of reducing *SSE* gets called, say, g_{l1} and g_{l2} and one sets

$$\hat{f}_l(\mathbf{x}) = \hat{\beta}_0 + \sum_{m=1}^{l} \left(\hat{\beta}_{m1} g_{m1}(\mathbf{x}) + \hat{\beta}_{m2} g_{m2}(\mathbf{x}) \right)$$

One might pick the complexity parameter l by cross-validation, but the standard implementation of MARS apparently uses instead a kind of generalized cross validation error.

MARS sequential model building cont.

The usual MARS generalized cross-validation error is

$$GCV(I) = \frac{\sum_{i=1}^{N} \left(y_i - \hat{f}_I\left(\mathbf{x}_i\right)\right)^2}{\left(1 - \frac{M(I)}{N}\right)^2}$$

where $M(I)$ is some kind of degrees of freedom figure. One must take account of both the fitting of the coefficients β in this and the fact that knots (values x_{ij}) have been chosen. The HTF recommendation is to use

$$M(I) = 2I + (2 \text{ or } 3) \cdot (\text{the number of different knots chosen})$$

(where presumably the knot count refers to different x_{ij} appearing in at least one $g_{m1}(\mathbf{x})$ or $g_{m2}(\mathbf{x})$).

MARS example

Below is a graphical portayal of a simple predictor (of, y, home sales price) of the kind a MARS algorithm can produce.