Basis Expansions and Regularization

Yongkai Chen, Yilin Li

Oct 16, 2018

Introduction

In regression problems, $f(X) = E(Y|X)$ will typically be nonlinear and nonadditive in X, and representing $f(X)$ by a linear model is usually a convenient, and sometimes a necessary, approximation.

Linear basis expansion

Denote by $h_m(X) : \mathbb{R}^p \to \mathbb{R}$ the mth transformation of X.We model

$$f(X) = \Sigma_{m=1}^{M} \beta_m h_m(X)$$

Example 1.1 (Basis functions)

- *Linear* $h_m(X) = X$
- *Degree-2 polynomial* $h_m(X) = X_j^2$ and $h_m(X) = X_i X_j$
- *Nonlinear* $h_m(X) = log(X_j)$ or $h_m(X) = \sqrt{X_j}$
- *Indicate* $h_m(X) = I(L_m < X_k < U_m)$

Piecewise polynomials

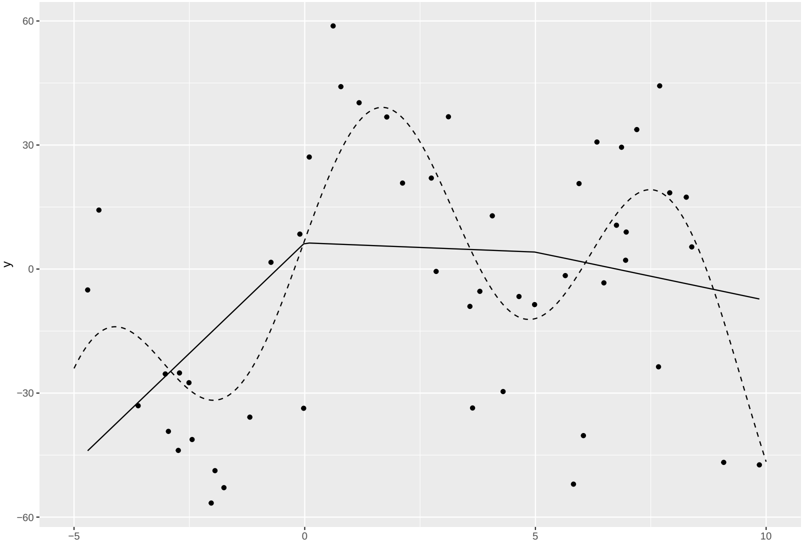
Piecewise polynomials

A **piecewise polynomial function** $f(X)$ is obtained by dividing the domain of $\mathcal{X}$ into contiguous intervals, and representing f by a separate polynomial in each interval.

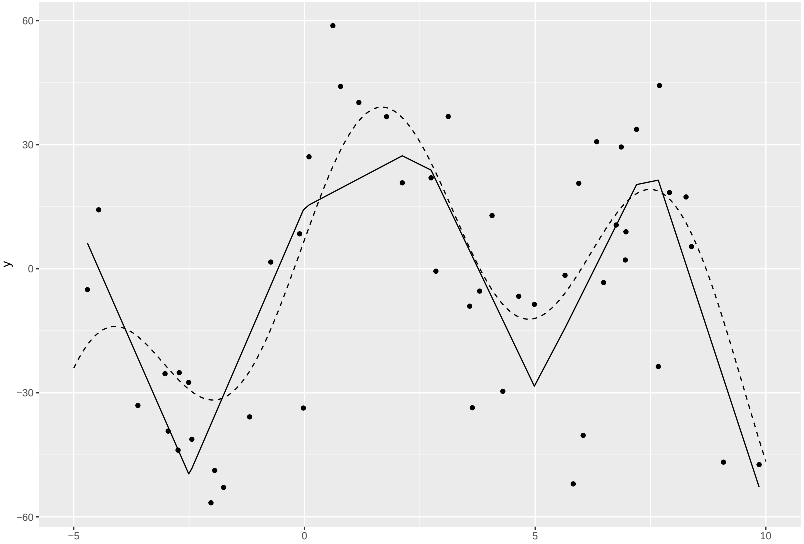
Example 2.1 (Piecewise polynomials by R)

```
library(ggplot2)
library(splines)
x<-runif(50,-5,10)
#Generate the underlying function
f<-function(x){25*sin(x)-x^2+5*x+x^1+7}
#Create the samples
y<-f(x)+rnorm(50,0,20)
x0<-seq(-5,10,by=0.01)
```

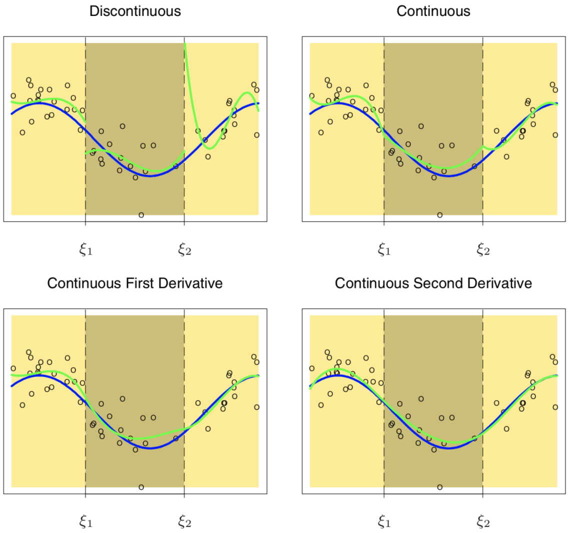
```r
m1 = lm(y~bs(x,degree=1,df=1,knots=c(0,5)))
qplot(x, y)+geom_line(aes(x,fitted(m1)))+
geom_line(aes(x0,f(x0)),linetype=2)
```



```r
m3 = lm(y~bs(x,degree=1,knots=c(-2.5,0,2.5,5,7.5)))
qplot(x, y)+geom_line(aes(x,fitted(m3)))+
geom_line(aes(x0,f(x0)),linetype=2)
```

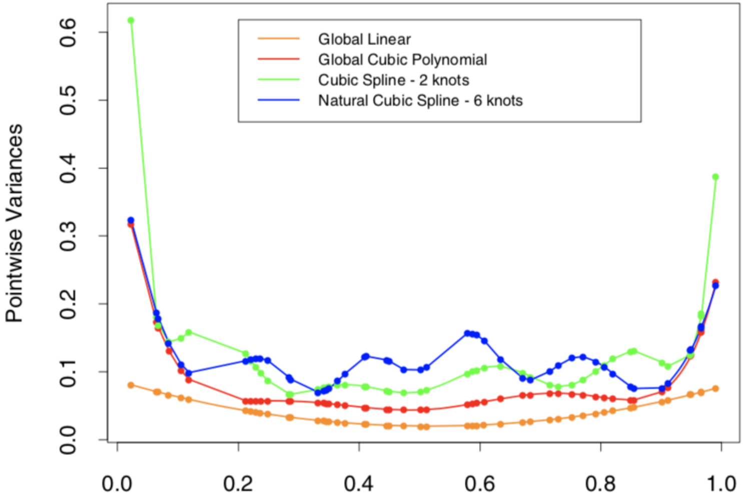


Piece-wise cubic Polynomials



Natural Cubic Splines

The behavior of polynomials fit to data tends to be erratically near the boundaries. The **pointwise variance** curves are shown below



A natural cubic spline adds additional constraints, namely that the function is **linear beyond the boundary knots**,and the spline with K knots is represented by K basis functions,which are

$$N_1(X) = 1, N_2(X) = X, N_{k+2} = d_k(X) - d_{K-1}(X),$$

where

$$d_k(X) = \frac{(X - \xi_k)^3_+ - (X - \xi_K)^3_+}{\xi_K - \xi_k}, k = 1, 2, ..., K - 1$$

Each of these basis functions can be seen to have zero second and third derivative for $X \geq \xi_K$.

The method for generating natural cubic spline

The natural cubic spline can be written as **truncated power series representation**

$$f(X) = \sum_{j=0}^{3} \beta_j X^j + \sum_{k=1}^{K} \theta_k (X - \xi_k)_+^3$$

The linear condition at left boundary shows $\beta_2 = 0, \beta_3 = 0$

As well as the linear condition at right boundary indicates $\sum_{k=1}^{K} \theta_k = 0, \sum_{k=1}^{K} \xi_k \theta_k = 0$

Smoothing splines

Smoothing splines

Among all functions $f(x)$ with two continuous derivatives, find one that minimizes the penalized residual sum of squares

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

where $\lambda \in (0, \infty)$ is a fixed *smoothing parameter*.

- $\lambda = 0$: f can be any function that interpolates the data.
- $\lambda = \infty$: the simple least squares line fit, since no second derivative can be tolerated.

A minimizer of smoothing splines: natural cubic splines

Derivation of smoothing splines

Suppose that $N \geq 2$, and that g is the natural cubic spline interpolant to the pairs $\{x_i, z_i\}_1^N$, with $a < x_1 < \cdots < x_N < b$. Prove that g is the unique one which minimizes RSS.

Proof. Let $\widetilde{g}(x)$ be any other differentiable function on $[a, b]$ that interpolates the N pairs. The first is equal for g and $\widetilde{g}(x)$.

Let $h(x) = \hat{g}(x) - g(x)$. Use integration by parts and the fact that g is a natural cubic spline to show that

$$\begin{aligned}
\int_a^b g^{''}(x)h^{''}(x)dx &= \sum_{i=1}^{N-1} \int_{x_i}^{x_{i+1}} g^{''}(x)h^{''}(x)dx \\
&= \sum_{i=1}^{N-1} (h^{'}(X)g^{''}(x)|_{x_i}^{x_{i+1}} - \int_{x_i}^{x_{i+1}} g^{''\prime}(x)h^{'}(x)dx) \\
&= 0 - \int_{x_i}^{x_{i+1}} g^{''\prime}(x_i^{+})h^{'}(x)dx \\
&= -\sum_{i=1}^{N-1} g^{''\prime}(x_i^{+})(h(x_{i+1}) - h(x_i))
\end{aligned}$$

For$g(x_i) = \widetilde{g}(x_i)$, then$h(x_i) = 0$. Hence $\int_a^b g^{"}(x)h^{"}(x)dx = 0$.

$$\begin{aligned}\int_a^b g^{"}(t)^2 dt &= \int_a^b g^{"}(t)\widetilde{g}(t)dt \\ &\leqslant (\int_a^b g^{"}(t)^2 dt \int_a^b \widetilde{g}^{"}(t)^2 dt)^{\frac{1}{2}}\end{aligned}$$

which means $\int_a^b g^{"}(t)^2 dt \leqslant \int_a^b \widetilde{g}^{"}(t)^2 dt$ and that equality can only hold if h is identically zero in $[a, b]$. #

With all above, the smoothing spline is given by

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

The minimizer can be rewritten as

$$RSS(\theta, \lambda) = (y - N\theta)^T (y - N\theta) + \lambda\theta^T \Omega_N \theta$$

Denote by $\hat{f}$ the N-vector of fitted values $f(xi)$ at the training predictors x_i. Then

$$\widehat{f} = N(N^T N + \lambda\Omega_N)^{-1} N^T y = S_\lambda y$$

Again the fit is linear in y, and the finite linear operator S_λ is known as the *smoother matrix*.

Suppose B_ξ is a $N \times M$ matrix of M cubic-spline basis functions evaluated at the N training points x_i, with knot sequence ξ, and $M << N$. Then the vector of fitted spline values is given by least square error

$$\widehat{f}(x) = B_\xi (B_\xi^T B_\xi)^{-1} B_\xi^T y = H_\xi y$$

There are some important similarities and differences between H_ξ and S_λ

- Both are symmetric, positive semidefinite matrices.
- $H_\xi H_\xi = H_\xi$ (idempotent), while $S_\lambda S_\lambda \ll S_\lambda$.
- H_ξ has rank M, while S_λ has rank N.

The expression $M = trace(H_\xi)$ gives the dimension of the projection space, which is also the number of basis functions, and hence the number of parameters involved in the fit.

By analogy we define the effective degrees offreedom of a smoothing spline to be

$$df_\lambda = trace(S_\lambda)$$

.

The **Reinsch form** of S_λ is

$$S_\lambda = (I + \lambda K)^{-1}$$

where $K = (N^{-1})^T \Omega_N (N^{-1})$.

For $\widehat{f} = S_\lambda y$ is the solution to

$$\min_f (y - f)^T (y - f) + \lambda f^T K f.$$

where K is known as the *penalty matrix*

The eigen-decomposition of S_λ is

$$S_\lambda = \sum_{k=1}^{N} \rho_k(\lambda) u_k u_k^T$$

with

$$\rho_k(\lambda) = \frac{1}{1 + \lambda d_k}$$

d_k the corresponding eigenvalue of K.

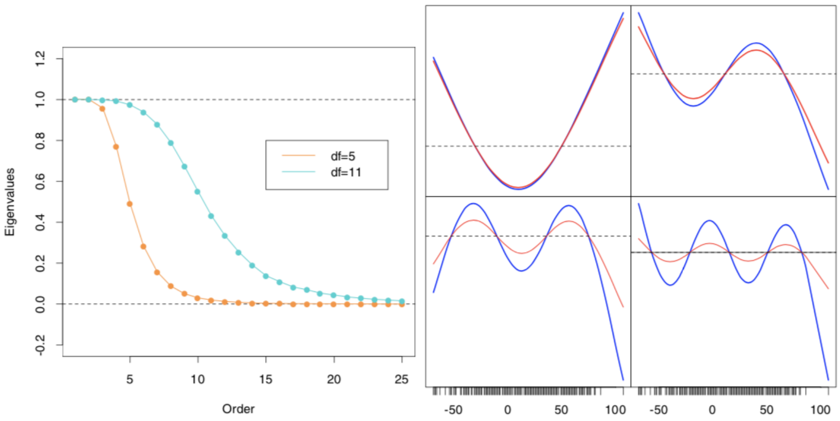


Figure 1:

(Left:) First 25 eigenvalues for the two smoothing-spline matrices.

(Right:) Third to sixth eigenvectors of the spline smoother matrices.

Automatic Selection of the Smoothing Parameters

The **integrated squared prediction error (EPE)** combines both bias and variance in a single summary:

$$\begin{aligned} EPE(\widehat{f}(\lambda)) &= E(Y - \widehat{f}_\lambda(X)) \\ &= Var(Y) + E[Bias^2(\widehat{f}_\lambda(X)) + Var(\widehat{f}_\lambda(x))] \\ &= \sigma^2 + MSE(\widehat{f}_\lambda) \end{aligned}$$

where σ^2 cannot be controlled, so it is equivilant to finding λ which minimizes $MSE(\widehat{f}(\lambda))$.

However $f(x)$ is unknown, so we use N-fold cross-validation to estimate EPE

$$CV(\widehat{f}_\lambda) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \widehat{f}_\lambda^{-i}(x_i))^2$$

Reproducing Kernel Hilbert Space(RKHS)

Kernel

Let $x, y \in \mathbb{R}^p$,for any function $K(x, y)$ satisfies

- positive definite: For any $f(x) \in L^2$,

$$\int\int f(x)K(x,y)f(y)dxdy \geq 0$$

- Symmetry: $K(x, y) = K(y, x)$

assume that K has an eigen-expansion

$$K(x,y) = \sum_{i=1}^{\infty} \lambda_i \phi_i(x)\phi_i(y)$$

with $\lambda_i \geq 0, \sum_{i=1}^{\infty} \lambda_i^2 < \infty$.(Mercer's theorem)

Every positive definite function $K(x, y)$ can span a RKHS $\mathcal{H}$, which is the space of functions generated by the linear span of $\phi_i(x)$ or $\{K(\cdot, y), y \in \mathbb{R}^p)\}$.

i.e. Elements of $\mathcal{H}_K$ have an expansion in terms of these eigen-functions

$$f(x) = \sum_{i=1}^{\infty} c_i \phi_i(x)$$

Then define the scalar product in RKHS to be:

$$< \sum_{i=1}^{\infty} c_i \phi_i(x), \sum_{i=1}^{\infty} d_i \phi_i(x) >_{\mathcal{H}} := \sum_{i=1}^{\infty} \frac{c_i d_i}{\lambda_i}$$

Thus, the norm in RKHS is

$$||f||_{\mathcal{K}}^2 = \sum_{i=1}^{\infty} \frac{c_i^2}{\lambda_i}$$

Several properties of RKHS

$K(x, y)$ is a Kernel and $f(x) \in \mathcal{H}_{\mathcal{K}}$

- $< K(\cdot, x_i), f >_{\mathcal{H}} = f(x_i)$
- $< K(\cdot, x_i), K(\cdot, x_j) >_{\mathcal{H}} = K(x_i, x_j)$(the reproducing property of $\mathcal{H}$)

Regularization and RKHS

Consider

$$min_{f\in\mathcal{H}}[\sum_{i=1}^{N} L(y_i, f(x_i)) + \lambda J(f)]$$

The second term is called **stabilizer**. Here we consider a special class of stabilizers, that is the norm $J(f) = ||f||^2_{\mathcal{K}}$ in a RKHS, which includes most of the usual regularization schemes.

Equivalently

$$min_{f\in\mathcal{H}}[\sum_{i=1}^{N} L(y_i, \sum_{i=1}^{\infty} c_i\phi_i(x)) + \lambda \sum_{i=1}^{\infty} \frac{c_i^2}{\lambda_i}]$$

The solution is finite-dimensional with the form

$$f(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i)$$

Proof.(c) If $g(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i)$, then

$$J(g) = ||g||_{\mathcal{K}}^2 = ||\sum_{i=1}^{N} \alpha_i K(x, x_i)||_{\mathcal{K}}^2 =< \sum_{i=1}^{N} \alpha_i K(x, x_i), \sum_{j=1}^{N} \alpha_j K(x, x_j) >_{\mathcal{H}}$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j < K(x, x_i), K(x, x_j) >= \sum_{i=1}^{N} \sum_{j=1}^{N} K(x_i, x_j) \alpha_i \alpha_j$$

Suppose that $h(x) = g(x) + \rho(x)$, with$\rho(x) \in \mathcal{H}_K$ and orthogonal in $\mathcal{H}_K$ to each of $K(x, x_i), i = 1, ..., N$.

Show that

$$\sum_{i=1}^{N} L(y_i, h(x_i)) + \lambda J(h) = \sum_{i=1}^{N} L(y_i, h(x_i)) + \lambda \| |g + \rho||_{\mathcal{K}}^2$$

$$= \sum_{i=1}^{N} L(y_i, h(x_i)) + \lambda \| |g||_{\mathcal{K}}^2 + \lambda ||\rho||_{\mathcal{K}}^2 \geq \sum_{i=1}^{N} L(y_i, h(x_i)) + \lambda \| |g||_{\mathcal{K}}^2$$

In addition

$$h(x_i) =< K(\cdot, x_i), h >_{\mathcal{H}} =< K(\cdot, x_i), g + \rho >_{\mathcal{H}} =< K(\cdot, x_i), g >_{\mathcal{H}} = g(x_i)$$

#

From all above, it can be reduced to a finite-dimensional criterion

$$min_{\alpha} L(\mathbf{y}, K\alpha) + \lambda \alpha^T K\alpha$$

In this way, several numerical algorithms can be used to optimize it.

Example(Genrealized ridge regression)

$$min_{\alpha}(\mathbf{y} - K\alpha)^T(\mathbf{y} - K\alpha) + \lambda \alpha^T K\alpha$$

The solution for α is obtained simple as

$$\hat{\alpha} = (K + \lambda I)^{-1}\mathbf{y}$$

The vector of N fitted values is given by

$$\widehat{\mathbf{f}} = K\hat{\alpha} = (I + \lambda K^{-1})^{-1}\mathbf{y}$$

Functional Data Analysis with R & Matlab

We build functions in two stages:

1. Firstly, we define a set of functional building blocks $\phi(x)_k$ called basis functions.

Example(Generate a B-spline basis)

Follow these steps:

1. Decide on the range, such as perhaps [0,1].
2. Choose an order, such as four.
3. Specify the number of basis functions. The more you specify, the more variability you can achieve in the function. As a first choice, 23 might be reasonable; for order four splines, this places by default knots at 0,0.05,0.10,. . . , 0.90,0.95 and 1 over [0,1].
4. Plot the basis to see how it looks using the plot command.

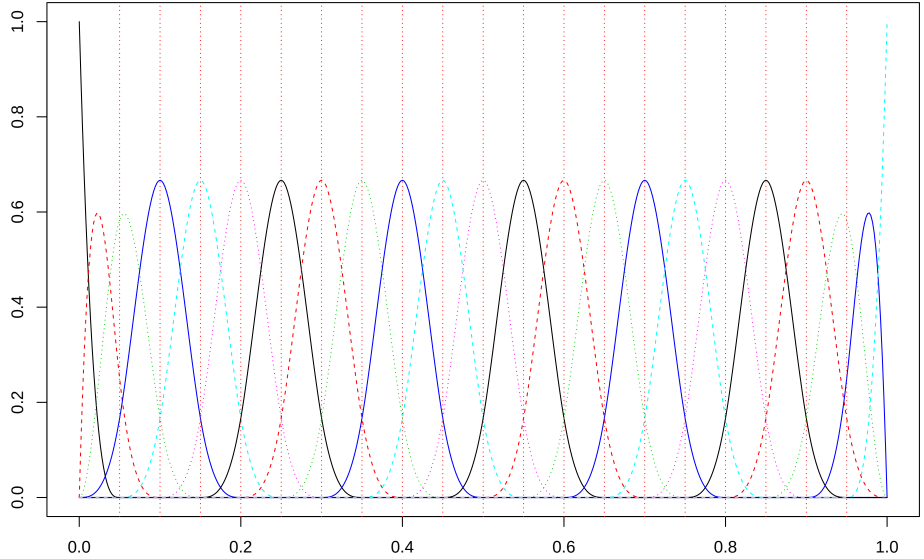
```r
library(fda)
```

```
## Loading required package: Matrix

##
## Attaching package: 'fda'

## The following object is masked from 'package:graphics':
##
##     matplot
```

```r
splinebasis=create.bspline.basis(c(0,1),nbasis=23,norder=4)
plot(splinebasis)
```

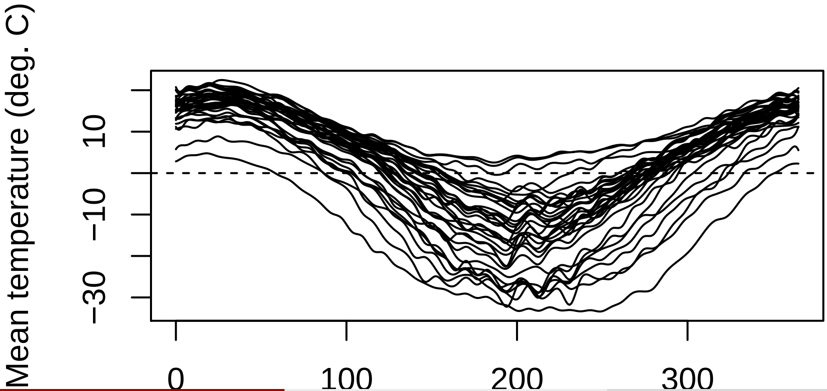


2. Then we set up a vector, matrix, or array of coefficients c to define the function as a linear combination of these basis functions.

$$x(t) = \sum_{k=1}^{K} c_k \phi_k(t) = \mathbf{c}^T \phi(t)$$

Example on the dataset CanadianWeather in package ‘fda’

```
daytime = (1:365)-0.5;JJindex = c(182:365, 1:181)
tempmat = daily$tempav[JJindex,]
tempbasis = create.bspline.basis(c(0,365),63)
tempfd = smooth.basis(daytime,tempmat,tempbasis)$fd
tempfd$fdnames = list("Day (July 2 to June 30)","Weather Stati
par(pin=c(3.5,1.3)); plot(tempfd, col=1, lty=1)
```


Mean temperature (deg. C)