SB1.2/SM2 Computational Statistics HT20

Lecturer: Geoff Nicholls

Lecture 7: B-Splines; Multivariate Regression

Notes and Problem sheets are available at

http://www.stats.ox.ac.uk/~nicholls/CompStats/(L1-7)

Lectures 8-13 will be given by Prof Windmeijer
frank.windmeijer@stats.ox.ac.uk
**B-splines** To construct solution, need basis for natural polynomial splines. The **B-spline** basis is convenient.

Splines: TL constant; TR linear; LL quadratic; LR cubic.
For knots $a = \xi_0 < \xi_1 < \xi_2 < \ldots, \xi_k \leq \xi_{k+1} = b$ in $[a, b]$, define new knots $\tau$ as (recall $M = 4$ for CS)

- $\tau_0 \leq \tau_1 \leq \ldots \leq \tau_M = \xi_0 = a$, ($M$ intervals below & above)
- $\tau_{j+M} = \xi_j$
- $b = \xi_{k+1} = \tau_{k+M+1} \leq \tau_{k+M+2} \leq \ldots \leq \tau_{k+2M+1}$

Define recursively. For $m = 1$ and $i = 1, \ldots, k + 2M$,

$$B_{i,1}(x) = \begin{cases} 1 & \tau_i \leq x < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

For $m \leq M$ and $i = 1, \ldots, k + 2M + 1 - m$.

$$B_{i,m}(x) = \frac{x - \tau_i}{\tau_{i+m-1} - \tau_i}B_{i,m-1}(x) + \frac{\tau_{i+m} - x}{\tau_{i+m} - \tau_{i+1}}B_{i+1,m-1}(x).$$

Support of $B_{i,m}(x)$ is $[\tau_i, \tau_{i+m}]$.

An implementation for R is `smooth.spline()` in the base distribution plus further functions in the `splines` package.
Fitting Splines

Take \( B_i(x) = B_{i,M}(x) \), with \( M = 4 \), we can write \( m \) in the B-spline basis as

\[
m(x) = \sum_{j=1}^{n+4} B_j(x)\beta_j
\]

Why \( n + 4 \)? Take knots at data \( a, x_1, \ldots, x_n, b \). The first spline non-zero in \( [a, b] \) is \( B_1(x) \) with support ending at \( x_1 \). The last one, \( B_{n+4} \), has support starting at \( x_n \).

The objective function, \( C(\beta) \) say,

\[
C(\beta) = \sum_{i=1}^{n} (y_i - m(x_i))^2 + \lambda \int (m''(x))^2 \, dx
\]

depends on \( \beta \) through \( m(x) \). Expanding,

\[
C(\beta) = \sum_{i=1}^{n} (y_i - \sum_{j=1}^{n+4} B_j(x_i)\beta_j)^2 + \lambda \sum_{j,j'=1}^{n+4} \beta_j \beta_{j'} \int B_j''(x)B_{j'}''(x) \, dx
\]

which we write in matrix form on the next slide.
\[ C(\beta) = (Y - B\beta)^T(Y - B\beta) + \lambda \beta^T\Omega\beta. \]

- \( Y \) is the \( n \times 1 \) vector \( Y = (y_1, \ldots, y_n) \).
- \( B \) is the \( n \times (n + 4) \)-matrix with entries \( B_{ij} = B_j(x_i) \) and
- \( \Omega \) is as \( (n + 4) \times (n + 4) \)-matrix with entries

\[
\Omega_{jk} = \int B''_j(x)B''_k(x) \, dx.
\]

Imposing \( \partial C/\partial \beta = 0 \) at \( \beta = \hat{\beta} \) gives the normal equations,

\[
(B^TB + \lambda\Omega)\hat{\beta} = B^TY.
\]

Setting \( \hat{m}(x) = \sum_{j=1}^{n+4} \hat{\beta}_j B_j(x) \), the fitted values are

\[
\hat{Y} = \hat{m}(x) = SY,
\]

with \( S = B(B^TB + \lambda\Omega)^{-1}B^T \), a linear smoother.
Solution depends on the regularization parameter $\lambda$.

- Value $\lambda$ non-intuitive
- Can find $\lambda = \lambda(df)$ for a given degrees of freedom $df$.

Here $df = \text{trace}(S)$ set to 10, 50, 150 corresponds to $\lambda$ approximately $2e - 3, 2e - 6$ and $2e - 20$ (!)
Choose \( \lambda \) by LOO-CV or GCV [see R-code in L6sp.R]
Multivariate smoothing

Consider smoothing functions $g(\cdot) : \mathbb{R}^p \mapsto \mathbb{R}$ with $p > 1$ now.

Can we just extend the methods and model functions $\mathbb{R}^p \mapsto \mathbb{R}$ nonparametrically?

Curse of Dimensionality

The usefulness of ‘local’ fitting is lost if $p \to \infty$ and $n$ constant. Say our data points $(x, y), y \in \mathbb{R},$

$$x = (x^{(1)}, x^{(2)}, \ldots, x^{(p)}) \in [0, 1]^p$$

are scattered uniformly in a unit cube.
A cube of side-length \(0 < \ell < 1\) occupies a fraction \(\ell^p\) of the volume \([0, 1]^p\). To capture say 5\% of the points we need \(\ell^p = 0.05\), or \(\ell = 0.05^{1/p}\),

<table>
<thead>
<tr>
<th>Dimension (p)</th>
<th>side length (\ell)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.05</td>
</tr>
<tr>
<td>2</td>
<td>0.22</td>
</tr>
<tr>
<td>5</td>
<td>0.54</td>
</tr>
<tr>
<td>10</td>
<td>0.74</td>
</tr>
<tr>
<td>1000</td>
<td>0.997</td>
</tr>
</tbody>
</table>

So ‘local’ is essentially the whole space.
Additive Models Treat this problem by imposing additive univariate structure. An additive model, $m_{add} : \mathbb{R}^p \mapsto \mathbb{R}$, takes

$$m_{add}(x) = \mu + \sum_{j=1}^{p} m_j(x^{(j)}), \quad \mu \in \mathbb{R}$$

where $m_j(\cdot) : \mathbb{R} \mapsto \mathbb{R}$ is the same sort of scalar nonparametric smoother we fit before and (for identifiability) we impose

$$\sum_{i=1}^{n} m_j(x^{(j)}_i) = 0 \quad \text{for each } j = 1, \ldots, p.$$ 

In fact $\mu$ may depend on other covariates $z = (z^{(1)}, \ldots, z^{(q)})$ in a simple parametric way, in which case

$$m_{add}(z, x) = z^T \theta + \sum_{j=1}^{p} m_j(x^{(j)}), \quad \theta \in \mathbb{R}^q$$

with the variables in $z$ distinct from those in $x$ for identifiability.
There are two settings for this sort of fitting. In the context of the course so far, we might assume

\[ Y = m_{\text{add}}(x) + \epsilon \]

with \( E(\epsilon) = 0 \) and choose \( m_{\text{add}}(x) \) to minimise the MSSE

\[
\sum_i \sum_j E((m_j(x_i^{(j)})) - \hat{m}_j(x_i^{(j)}))^2),
\]

for each \( j = 1, \ldots, p \), using, for example, LOO-CV.

There is also a GLM/spline setup, which makes explicit assumptions about the distribution of \( Y \). The linear predictor is

\[ \eta_i = z^T \theta + m_1(x_i^{(1)}) + m_2(x_i^{(2)}) + \ldots + m_p(x_i^{(p)}), \]

link function \( -1 g(\eta_i) = E(Y_i) \) and stochastic part \( Y_i \sim F_Y(\cdot; \eta_i) \). This is non-parametric, as the \( m_j(x^{(j)}) \) are smoothing splines.
These models are called **Generalised Additive Models** or GAM’s.

For eg., given binary data $Y_i \in \{0, 1\}$ and covariates $x_i \in \mathbb{R}^p$, we can avoid assumptions about the $x$-dependence by taking $p(\eta_i) = \exp(\eta_i)/(1 + \exp(\eta_i))$ and $Y_i \sim Bern(p(\eta_i))$, with

$$
\eta(x; \mu, m) = \mu + m_1(x^{(1)}) + m_2(x^{(2)}) + \ldots + m_p(x^{(p)}),
$$

We can estimate $p(x)$, the probability $Y = 1$ at $x \in \mathbb{R}^p$, leaving the $x$-dependence to be discovered by the spline smoother.

In this setting we have a likelihood, so we can fit by minimising the deviance $-2\ell(y; \mu, m) + \lambda J(m)$. This is a penalised likelihood method. We have many of the usual elements of testing, such as the AIC and Chi-Square tests, for model selection.
Backfitting

Given data $Y, \mathbf{x}, \ Y \in \mathbb{R}^n$, with $\mathbf{x} = [x_i^{(j)}]_{i=1:n}^{j=1:p}$ an $n \times p$ matrix, with column vectors $x^{(j)} = (x_1^{(j)}, \ldots, x_n^{(j)})$, $j = 1, \ldots, p$.

Take a smoothing spline and seek $m(x), \mu$ minimising

$$C(\mu, m) = \sum_i (Y_i - \mu - \sum_{j=1}^p m_j(x_i^{(j)}))^2 + \sum_j \lambda_j \int (m''_j(x))^2 dx$$

Let $\beta_j = (\beta_{j,1}, \ldots, \beta_{j,n+4})$ be spline parameters for $m_j$ so that

$$m_j(x) = \sum_{k=1}^{n+4} B_{j,k}(x) \beta_{j,k} \quad x \in \mathbb{R}.$$  

If $B_j$ is an $n \times (n + 4)$-matrix with entries $B_{jk} = B_{j,k}(x_i^{(j)})$, then $m_j(x^{(j)}) = B_j \beta_j$. We just put a $j$-index on each object.
Let $\Omega^{(j)}$ be the matrix with entries $\Omega^{(j)}_{ik} = \int B''_{j,i}(x)B''_{j,k}(x) \, dx$.

We can write

$$C(\mu, m) = |Y - \mu 1_n - \sum_{j=1}^{p} B_j \beta_j|^2 + \sum_{j} \lambda_j \beta_j^T \Omega_j \beta_j.$$ 

Now suppose we know $\hat{\mu}$ and $\hat{\beta}_j$ for all $j \neq d$. We can split off the unknown term,

$$C'(\hat{\mu}, \hat{m}) = |Y - \hat{\mu} 1_n - \sum_{j \neq d} B_j \hat{\beta}_j - B_d \hat{\beta}_d|^2 + \lambda_d \beta_d^T \Omega_d \beta_d + \ldots$$

$$= |\tilde{Y} - B_d \beta_d|^2 + \lambda_d \beta_d^T \Omega_d \beta_d + \text{const wrt } \beta_d$$

We choose $\hat{\beta}_d$ to minimise this, and this only reduces our loss function $C(\mu, m)$. We cycle through all $d = 1, \ldots, p$ repeating this until we get no improvement.
The backfitting algorithm

1. Set $\hat{m}_j \equiv 0$ for all $j = 1, \ldots, p$. Set $\hat{\mu} \leftarrow n^{-1} \sum_{i=1}^{n} Y_i$.
2. Cycle through the indices $d = 1, 2, \ldots, p, 1, 2, \ldots, p, 1, 2, \ldots$
   (a) Set
   $$\tilde{Y} = Y - \hat{\mu}1_n - \sum_{j \neq d} \hat{m}_j(x^{(j)}).$$
   (b) Compute a 1D smoother $\hat{m}_d(x^{(d)}) = B_d\hat{\beta}_d$ for $\tilde{Y}$,
   $$\hat{\beta}_d = \arg \min_{\beta_d} |\tilde{Y} - B_d\beta_d|^2 + \lambda_d \beta_d^T \Omega_d \beta_d$$
   and re-zero $\hat{m}_d(x^{(d)}) \leftarrow \hat{m}_d(x^{(d)}) - n^{-1} \sum_{i=1}^{n} \hat{m}_d(x_i^{(d)})$.
   (c) Update $\hat{\mu} \leftarrow n^{-1} \sum_{i=1}^{n} (Y_i - \sum_j \hat{m}_j(x_i^{(j)}))$.
   (d) Stop when the targeted loss ceases to improve (it gets smaller at each iteration but the gain becomes negligible).
3. Return $\hat{\mu}$ and $\hat{m}_j(x^{(j)}) = B_j\hat{\beta}_j$, $j = 1, \ldots, p$, the smoothers for each covariate and

$$
\hat{m}_{add}(x) \leftarrow \hat{\mu}1_n + \sum_{j=1}^{p} \hat{m}_j(x^{(j)}),
$$

the $n \times 1$ vector of fitted values $\hat{Y} = \hat{m}_{add}$. 
Example: Simple 2D function

\[ Y_i = 1 + (x_i^{(1)})^2 - (x_i^{(2)})^2 + \epsilon_i, \quad \epsilon_i \sim N(0, (0.2)^2) \]

observed on a 20 \times 20 grid in \([-1, 1]^2\).

\[
n=20; \quad N=n^2; \quad x1=\text{seq}(-1,1,\text{length.out}=n); \quad x2=x1 \\
X=\text{expand.grid}(x1,x2) \\
np=1+X[,1]^2-X[,2]^2 \quad \#\text{the truth!} \\
s=0.2; \quad y=np+s*\text{rnorm}(N) \quad \#\text{the data}
\]

Use backfitting with splines to fit

\[ y = \mu + s(x12[, 1]) - s(x12[, 2]) + \epsilon \]

See L7mv.R.
Truth

Fit and Data
Example: Daily ozone - levels in LA basin with 9 predictors.
The \texttt{R} function \texttt{gam()} (Generalized Additive Models) uses penalized regression splines, a variation of smoothing splines. The degrees of freedom for each variable are determined by generalized cross-validation.

See fit on final slide.

We can go on to do model selection etc much as we would for a NLM or GLM. See \texttt{L7mv.R} for the rest of this example.
> addmod <- gam(O3 ~ s(vdht)+s(wind)+s(humidity)+s(temp)+s(ibht)+
    s(dgpg)+s(ibtp)+s(vsty)+s(day), data=ozone)
> summary(addmod)
Family: gaussian
Link function: identity

Approximate significance of smooth terms:
        edf Ref.df  F p-value
s(vdht) 1.000 1.000 8.712 0.003402 **
s(wind) 1.000 1.000 4.271 0.039595 *
s(humidity) 3.631 4.517 2.763 0.018556 *
s(temp) 4.361 5.396 4.864 0.000182 ***
s(ibht) 3.043 3.725 1.292 0.356960
s(dgpg) 3.230 4.108 10.603 3.46e-08 ***
s(ibtp) 1.939 2.504 1.808 0.197988
s(vsty) 2.232 2.782 5.822 0.000890 ***
s(day) 4.021 5.055 15.817 3.58e-14 ***

R-sq.(adj) =  0.797    Deviance explained = 81.2%
GCV = 14.137  Scale est. = 13.046    n = 330