Smooth effect types & Big Data methods

Matteo Fasiolo (University of Bristol, UK)

matteo.fasiolo@bristol.ac.uk

June 27, 2018

Smooth effect types & Big Data methods

Structure:

1. GAM model fitting
2. Types of smooth effects
3. Big Data methods

Structure of the talk

Structure:

1. **GAM model fitting**
2. Types of smooth effects
3. Big Data methods

GAM model fitting

Recall the GAM model structure:

$$y|\mathbf{x} \sim \text{Distr}\{y|\mu(\mathbf{x}), \boldsymbol{\theta}\}$$

where $\mu(\mathbf{x}) = \mathbb{E}(y|\mathbf{x}) = g^{-1}\big\{\sum_{j=1}^{m} f_j(\mathbf{x})\big\}$.

The f_j's can be

- parametric e.g. $f_j(\mathbf{x}) = \beta_1 x_j + \beta_2 x_j^2$
- random effects
- spline-based smooths such as

$$f_j(x_j) = \sum_{i=1}^{r} \beta_{ji} b_{ji}(x_j)$$

  where β_{ji} are coefficients and $b_{ji}(x_j)$ are known spline basis functions.

NB: we call $\sum_{j=1}^{m} f_j(\mathbf{x})$ **linear predictor** because it is linear in $\boldsymbol{\beta}$.

GAM model fitting

$\hat{\boldsymbol{\beta}}$ is the maximizer of **penalized** log-likelihood

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\text{argmax}}\ \text{PenLogLik}(\boldsymbol{\beta}|\boldsymbol{\gamma}) = \underset{\boldsymbol{\beta}}{\text{argmax}} \Big\{ \overbrace{L_y(\boldsymbol{\beta})}^{\text{goodness of fit}} - \underbrace{\text{Pen}(\boldsymbol{\beta}|\boldsymbol{\gamma})}_{\text{penalize complexity}} \Big\}$$

where:

- $L_y(\boldsymbol{\beta}) = \sum_i \log p(y_i|\boldsymbol{\beta})$ is log-likelihood
- $\text{Pen}(\boldsymbol{\beta}|\boldsymbol{\gamma})$ penalizes the complexity of the f_j's
- $\boldsymbol{\gamma} > 0$ smoothing parameters ($\uparrow \boldsymbol{\gamma}$ $\uparrow$smoothness)

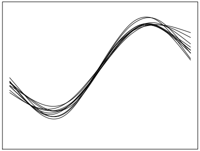
GAM model fitting

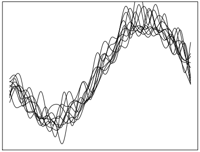
We use a hierarchical framework (Wood, 2011):

1. Select γ determine smoothness

$$\hat{\gamma} = \underset{\gamma}{\text{argmax}} \, \text{LAML}(\gamma)$$

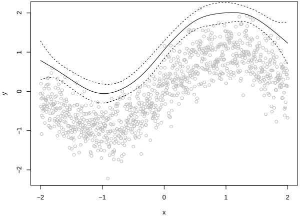
where $\text{LAML}(\gamma) \approx p(y|\gamma) = \int p(y, \boldsymbol{\beta}|\gamma) d\boldsymbol{\beta}$.





2. For fixed γ, estimate $\boldsymbol{\beta}$ to determine actual fit

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\text{argmax}} \, \text{PenLogLik}(\boldsymbol{\beta}|\gamma).$$



GAM model fitting

Alternatives to Laplace Approximate Marginal Likelihood (LAML) for γ selection:

- Generalized Cross-Validation (GCV)
- Akaike Infomation Criterion (AIC)

but LAML is most widely applicable in `mgcv`.

Variance parameters of random effects can be included in γ and estimated by LAML.

Extra parameters $\boldsymbol{\theta}$ of $y|\mathbf{x} \sim \text{Distr}\{y|\mu(\mathbf{x}), \boldsymbol{\theta}\}$ handled similarly.

Structure of the talk

Structure:

1. GAM model fitting
2. **Types of smooth effects**
3. Big Data methods

Types of smooths

`mgcv` offers a wide variety of smooths (see `?smooth.terms`).

Univariate types:

- `s(x) = s(x, bs = "tp")` thin-plate-splines
- `s(x, bs = "cr")` cubic regression spline
- `s(x, bs = "ad")` adaptive smooth

Multivariate type:

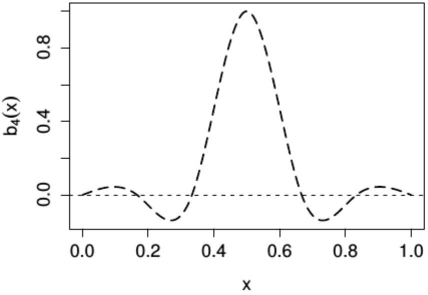
- `s(x1, x2) = s(x1, x2, bs = "tp")` thin-plate-splines (isotropic)
- `te(x1, x2)` tensor-product-smooth (anisotropic)
- `s(x, y, bs = "sos")` smooth on sphere

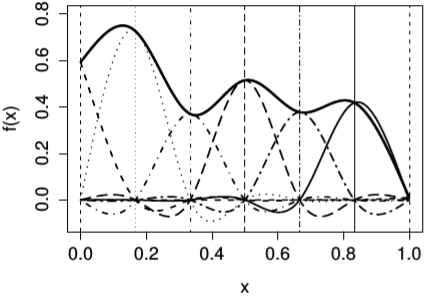
They can depends on factors:

- `s(x, by = Subject)`
- `s(x, Subject, bs = "fs")`

Types of smooths

```
s(x, bs = "cr", k = 20)
```



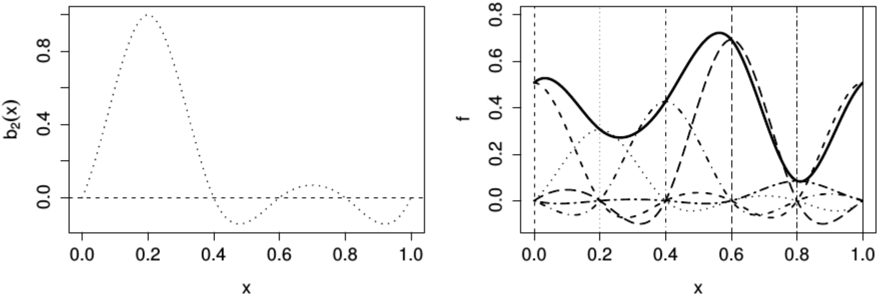


Cubic regression splines are related to the optimal solution to

$$\sum_{i=1}^{n}\{y_i - f(x_i)\}^2 + \gamma \int f''(x)^2 \, dx.$$

Types of smooths

```
s(x, bs = "cc")
```

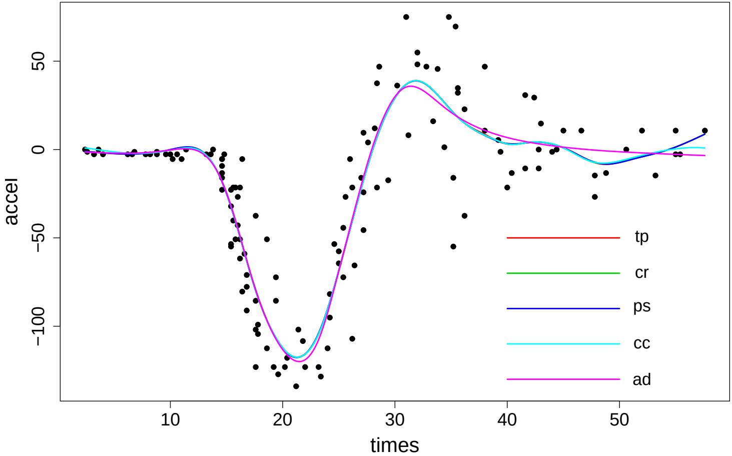


Cyclic cubic regression splines make so that

- $f(x_{min}) = f(x_{max})$
- $f'(x_{min}) = f'(x_{max})$

Types of smooths

`s(x, bs = "ad")`



The wiggliness or smoothness of $f(x)$ depends on x.

Types of smooths

`s(x1, x2)`, `s(x1, x2, x3)`, ...

Based on thin plate regression splines basis.

Related to optimal solution to:

$$\sum_i \{y_i - f(x_i, z_i)\}^2 + \gamma \int f_{xx}^2 + 2f_{xz}^2 + f_{zz}^2 dxdz$$

A single smoothing parameter γ.

Isotropic: same smoothness along x_1, x_2, ...

Types of smooths

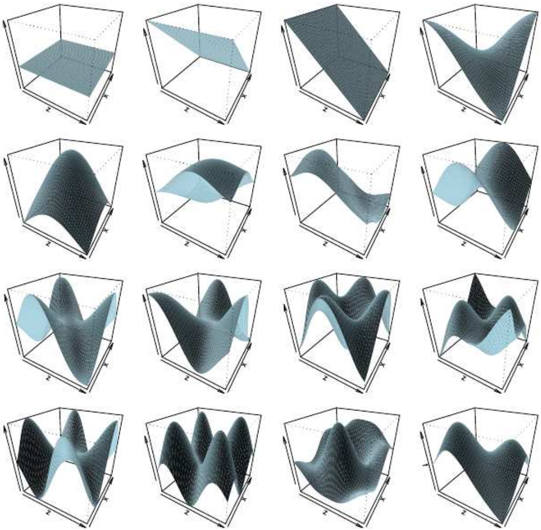


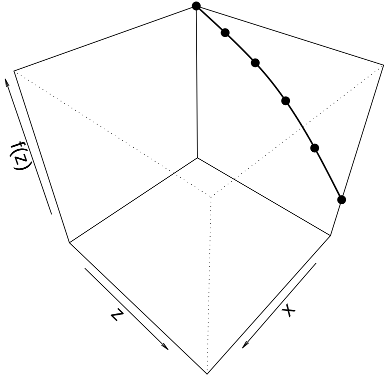
Figure : Rank 17 2D TPRS basis. Courtesy of Simon Wood.

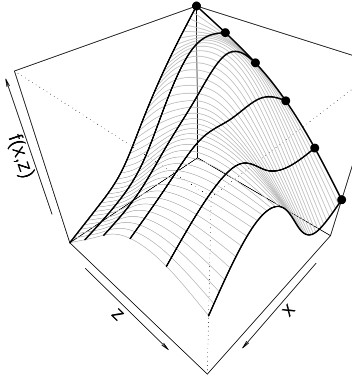
Types of smooths

Isotropic effect of x_1, x_2 are in same unit (e.g. Km).

If different units better use tensor product smooths `te(x1, x2)`.

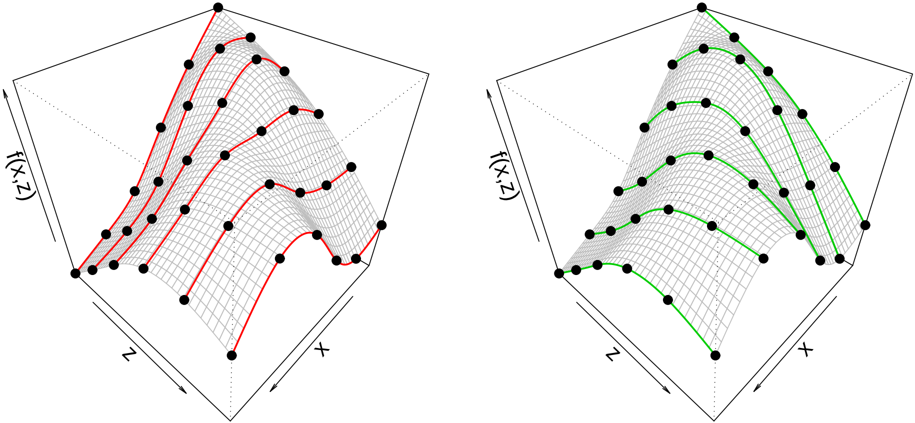
Construction: make a spline $f_z(z)$ a function of x by letting its coefficients vary smoothly with x





Types of smooths

- x-penalty: average wiggliness of red curves
- z-penalty: average wiggliness of green curves



Types of smooths

Can use (almost) any kind of marginal:

- `te(x1, x2, x3)` product of 3 cubic regression splines bases
- `te(x1, x2, bs = c("cc", "cr"), k = c(10, 6))`
- `te(LO, LA, t, d=c(2,1), k=c(20,10), bs=c("tp","cc"))`

Basis of `te` contains functions of the form $f(x_1)$ and $f(x_2)$.

To fit $f(x_1) + f(x_2) + f(x_1, x_2)$ separately use:

```
y ~ ti(x1) + ti(x2) + ti(x1, x2)
```

Types of smooths

By-factor smooths

Approach (1) is `s(x, by = subject)`, which means

- $\mu(x) = f_1(x) + \dots \qquad$ if subject $= 1$
- $\mu(x) = f_2(x) + \dots \qquad$ if subject $= 2$
- ...

Approach (2) is `s(x, subject, bs = "fs")`, which means

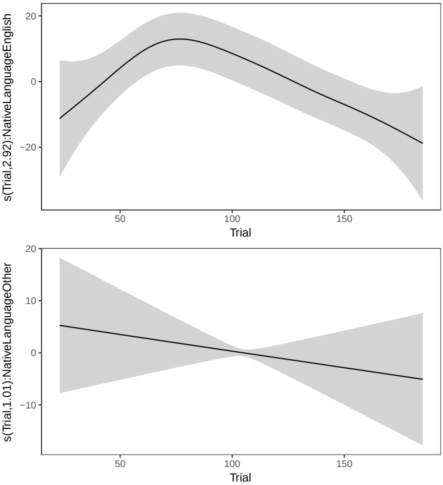
- $\mu(x) = b_1 + f_1(x) + \dots \qquad$ if subject $= 1$
- $\mu(x) = b_2 + f_2(x) + \dots \qquad$ if subject $= 2$
- ...

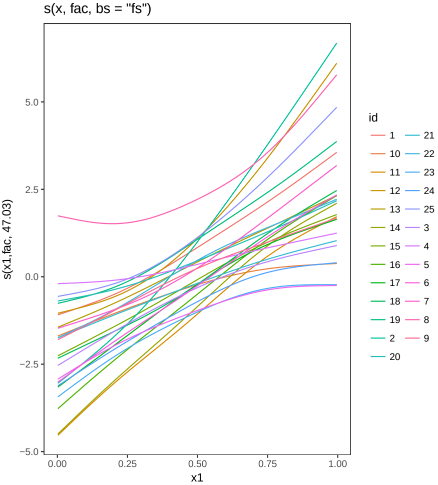
where $b_1, b_2, \dots \sim N(0, \gamma_{\mathbf{b}}\mathbf{I})$ are random effects.

In (1) each f_j has its own smoothing parameter.

In (2) all f_j's have the same smoothing parameter.

Types of smooths





Structure of the talk

Structure:

1. GAM model fitting
2. Types of smooth effects
3. **Big Data methods**

Big Data methods

Recall the GAM model structure

$$\mu(\mathbf{x}) = \mathbb{E}(y|\mathbf{x}) = g^{-1}\Big\{ \sum_{j=1}^{m} f_j(\mathbf{x}) \Big\}$$

Here $\mu(\mathbf{x}_i)$ can be written as $g^{-1}(\mathbf{X}_i\boldsymbol{\beta})$, where $\mathbf{X}_i$ row of matrix $\mathbf{X}$ having has n rows and

$$d = p + k_1 + \cdots + k_j + \cdots + k_m$$

columns.

Big Data methods

Bottom line: $\mathbf{X}$ can get very big, which causes problems:

- storing $\mathbf{X}$ takes too much memory
- computing things involving $\mathbf{X}$ (e.g. $\mathbf{X}^\mathsf{T}\mathbf{X}$) takes time

Solution implemented in `mgcv::bam` function:

- do not create $\mathbf{X}$ but only sub-blocks:

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} \\ \mathbf{X}_{21} & \mathbf{X}_{22} \end{bmatrix}$$

  do not store them either, but create them when needed;
- any computation involving $\mathbf{X}$ is based on the blocks;
- use parallelization when possible;

Big Data methods

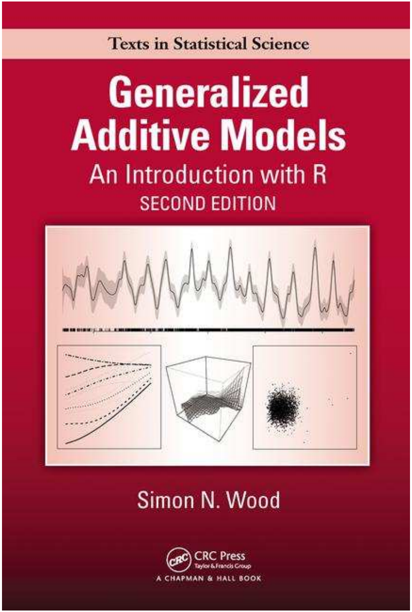
Further acceleration and memory savings by discretization.

Instead of having n unique rows of $\mathbf{X}$ discretize to $b << n$ rows.

In `mgcv`:

```
fit <- bam(y ~ s(x),
           discrete = TRUE,
           nthreads = 2,
           ...)
```

Further reading



References I

Wood, S. N. (2011). Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 73*(1), 3–36.