Modèles additifs

Yannig Goude [1]

[1] Laboratoire de Mathématiques d'Orsay
EDF R&D, EDF Lab Saclay

M2 MDA-statML, 2023

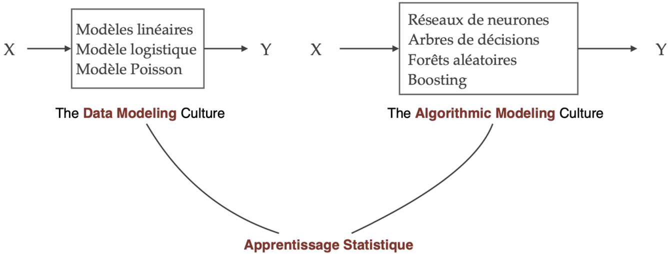
Sommaire

Introduction

Statistical Science
2001, Vol. 16, No. 3, 199–231

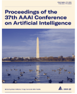
Statistical Modeling: The Two Cultures

Leo Breiman



Introduction

DEEP LEARNING BASED FORECASTS (TRANSFORMERS) ARE STILL IN DEBATE



The Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)

Are Transformers Effective for Time Series Forecasting?

Ailing Zeng[1,2]*, **Muxi Chen**[1]*, **Lei Zhang**[2], **Qiang Xu**[1]
[1]The Chinese University of Hong Kong
[2]International Digital Economy Academy
{zengailing, leizhang}@idea.edu.cn,{mxchen21, qxu}@cse.cuhk.edu.hk

« Surprisingly, our results show that LTSF-Linear outperforms existing complex Transformerbased models in all cases, and often by a large margin (20%, 50%). »

NeurIPS | 2022
Thirty-sixth Conference on Neural Information Processing Systems

Why do tree-based models still outperform deep learning on tabular data?

Léo Grinsztajn Soda, Inria Saclay leo.grinsztajn@inria.fr
Edouard Oyallon ISIR, CNRS, Sorbonne University
Gaël Varoquaux Soda, Inria Saclay

« On medium size tabular data tree-based models more easily yield good predictions, with much less computational cost. »

Parametric regression

Consider a regression context, where y is a dependent variable with conditional distribution $p(y|\boldsymbol{x})$, $\boldsymbol{x}$ being a d-dimensional vector of covariates.
In distributional regression, we are typically interested in modelling $p(y|\boldsymbol{x})$ via a parametric model : $p(y|\boldsymbol{\theta}, \boldsymbol{x})$ which is parametrized by the m-dimensional vector of parameters $\boldsymbol{\theta}$.

- the elements of $\boldsymbol{\theta}$ control various characteristic of the response distribution, such as location, scale and shape.
- in a standard regression modelling context, we allow only one of the elements of $\boldsymbol{\theta}$ to depend on $\boldsymbol{x}$.
- in the following, we call such parameter $\mu = \mu(\boldsymbol{x})$ and we use $\boldsymbol{\theta}$ to refer to the remaining parameters.

Parametric regression

$\mu(\boldsymbol{x})$ is typically a location parameter, which controls the conditional mean of the response, $\mathbb{E}(y|\boldsymbol{x})$.

- Gaussian regression : assume that $y \sim \mathrm{N}\{\mu(\boldsymbol{x}), \sigma^2\}$ and parameter μ acts exclusively on the conditional mean, while the scale is controlled by σ.
- Poisson regression : assume that $y \sim \mathrm{Poi}\{\mu(\boldsymbol{x})\}$, where $\mathbb{E}(y|\boldsymbol{x}) = \mathrm{var}(y|\boldsymbol{x})$, hence modelling the rate $\mu(\boldsymbol{x})$ results in both the mean and the variance being dependent on the covariates.

GAMs
Introduction

In GAM models, μ has a semi-parametric additive structure, that is

$$g\{\mu(\boldsymbol{x})\} = \boldsymbol{z}^{\mathsf{T}}\beta^0 + \sum_{j=1}^{J} f_j(\boldsymbol{x}), \tag{1}$$

where g is a known monotonic function, $\boldsymbol{z} = \boldsymbol{z}(\boldsymbol{x})$ is d-dimensional vector whose value depends on the covariates $\boldsymbol{x}$ and the f_j's are smooth effects. Hence $\boldsymbol{z}^{\mathsf{T}}\beta^0$ represents the parametric part of the model, with unknown regression coefficients β^0.

GAM

Additive model

The f_j's are built using spline bases expansions, so the j-th effect can be written

$$f_j(\boldsymbol{x}) = \boldsymbol{b}_j^\mathsf{T}\boldsymbol{\beta}^j = \sum_{k=1}^{K_j} b_j^k(\boldsymbol{x})\beta_k^j,$$

where

- $\boldsymbol{b}_j = \{b_j^1, \ldots, b_j^{K_j}\}$ are the spline basis functions used to built the j-th effect
- $\boldsymbol{\beta}^j = \{\beta_1^j, \ldots, \beta_{K_j}^j\}$ are the corresponding regression coefficients.

The basis functions are known and fixed, while the regression coefficients must be estimated. The dependence of μ on $\boldsymbol{\beta}$ is linear, in fact we can write

$$g\{\mu(\boldsymbol{x})\} = \mathbf{x}^\mathsf{T}\boldsymbol{\beta},$$

where $\mathbf{x} = \{\boldsymbol{z}, \boldsymbol{b}_1, \cdots, \boldsymbol{b}_J\}$ and $\boldsymbol{\beta} = \{\boldsymbol{\beta}^0, \boldsymbol{\beta}^1, \ldots, \boldsymbol{\beta}^J\}$

GAMs

Spline basis expansion

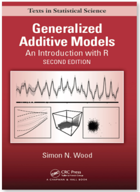
$$g\{\mu(\boldsymbol{x})\} = \boldsymbol{z}^{\mathsf{T}}\boldsymbol{\beta}^0 + \sum_{j=1}^{J} f_j(\boldsymbol{x}), \qquad (1)$$

- a GAM is a GLM where the linear predictor depends on smooth functions of covariates.
- the r.h.s. of (1) is generally called the “linear predictor”. While μ depends on both $\boldsymbol{x}$ and $\boldsymbol{\beta}$, here we refer to μ using either $\mu(\boldsymbol{x})$ or $\mu(\boldsymbol{\beta})$, depending on context.

GAMs

History of GAM

- Grace Wahba [Wah80] introduce penalized regression splines.
- Trevor Hastie and Robert Tibshirani invented GAMs [HT86] and GAM were originally fitted using the backfitting algorithm.
- Paul Eilers [EM96] improved the work of Wahba and apply it to GAMs in 1998.
- Simon Wood proposed thin plate regression splines [Woo03] and a global/powerful implementation in the R package mgcv. His book [Woo17] is a reference on the subject.



Why do we need spline bases expansions ?

Now we focus on a simple univariate problem :

$$y = f(x) + \varepsilon$$

where f is a smooth function.

- A common approach to dealing with nonlinear relationship like that is to consider polynomial(of a given order) transformation of x in a linear regression model. This "global" parametric regression model is limited, too restrictive for f to be correctly estimated, leading to systematic bias.
- Thinking more locally, make more qualitative hypothesis on f (like "f is smooth") without imposing a specific structure on f is the objective on non-parametric regression.
- These methods are more flexible and let the data speak themselves. They can uncover some structure in the data that would be missed by parametric regression.

Theoretical justification

For the regression problem :

$$y = f(x) + \varepsilon$$

we now precise the smoothness of f. We assume that x lies in $[a, b]$ and $f \in W_2^m[a, b]$ the *Sobolev space* :

$$W_2^m[a,b] = \{f : f, f', ..., f^{(m-1)} \text{are absolutely continuous}, \int_a^b (f^{(m)})^2 \, dx < \infty\}$$

then, for any $x \in [a, b]$, the Taylor's theorem states that :

$$f(x) = \underbrace{\sum_{k=0}^{m-1} \frac{f^{(k)}(a)}{k!}(x-a)^k}_{\text{polynomial of order m}} + \underbrace{\int_a^x \frac{(x-u)^{m-1}}{(m-1)!} f^{(m)}(u) \, du}_{\text{remainder term : Rem(x)}}$$

We see here that the regression model only include the first term, neglecting the Rem(x) term. The idea of regression spline is to let the data decide how large Rem(x) should be (see [Wan11]).

Penalized splines

Penalized splines aim at minimising the adjustment to the data while having a certain smoothness (red curve).

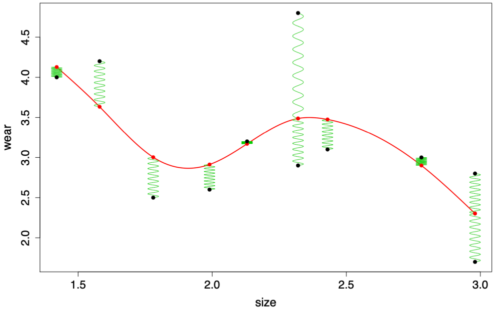


Figure – Original splines idea, source : Simon Wood.

An example : electricity consumption data

For now just assume that splines are piecewise polynomial of a fixed degree (usually 3) with some constraints at some points called the knots.
An example :

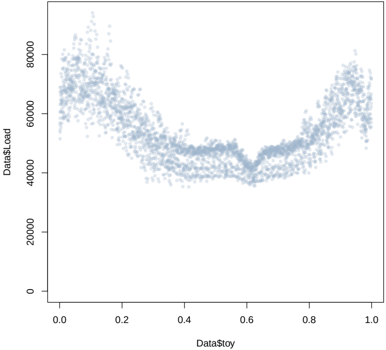


Figure – French load data.

An example : electricity consumption data

For now just assume that splines are piecewise polynomial of a fixed degree (usually 3) with some constraints at some points called the knots.
An example :

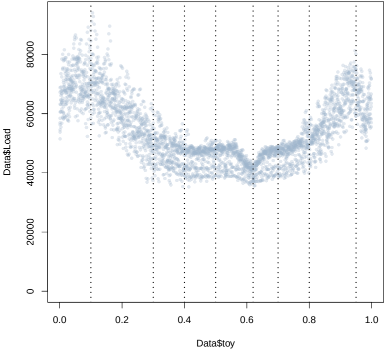


Figure – French load data.

An example : electricity consumption data

For now just assume that splines are piecewise polynomial of a fixed degree (usually 3) with some constraints at some points called the knots.
An example :

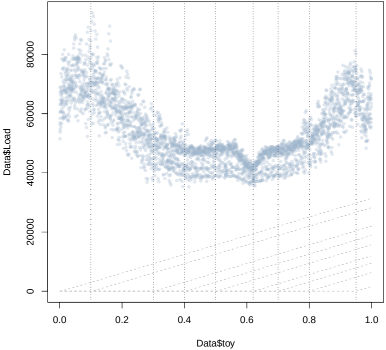


Figure – French load data.

An example : electricity consumption data

For now just assume that splines are piecewise polynomial of a fixed degree (usually 3) with some constraints at some points called the knots.
An example :

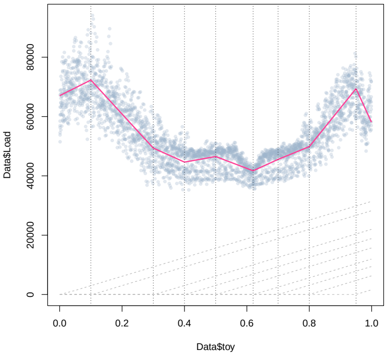


Figure – French load data.

An example : electricity consumption data

For now just assume that splines are piecewise polynomial of a fixed degree (usually 3) with some constraints at some points called the knots.
An example :

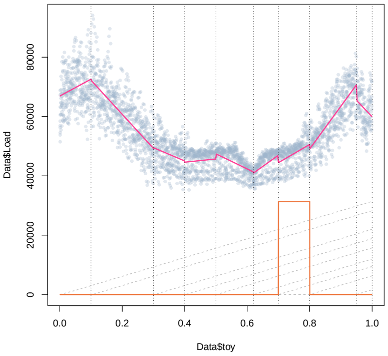


Figure – French load data.

An example : electricity consumption data

For now just assume that splines are piecewise polynomial of a fixed degree (usually 3) with some constraints at some points called the knots.
An example :

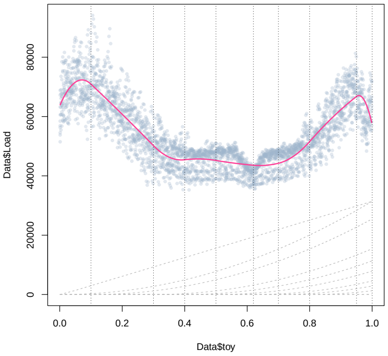


Figure – French load data.

Truncated power functions

In the previous example, we used truncated power functions, a very simple spline basis. Truncated power functions of order d with knots (a, b), for a covariate x are obtained with the following formulas :

- Polynomial part : $b_1(x) = 1$, $b_2(x) = x$,..., $b_d(x) = x^d$
- Piecewise polynomial part : $b_{d+1}(x) = (x - a)_+^d$, $b_{d+2}(x) = (x - b)_+^d$

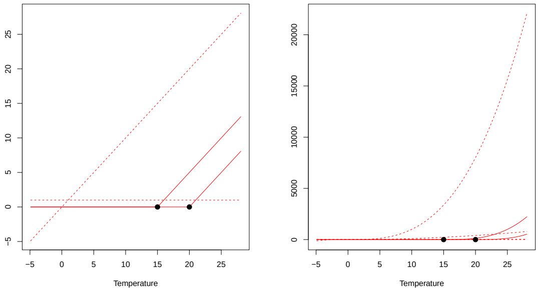


Figure – Truncated power functions regression.

Truncated power functions

In the previous example, we used truncated power functions, a very simple spline basis. Truncated power functions of order d with knots (a, b), for a covariate x are obtained with the following formulas :

- Polynomial part : $b_1(x) = 1$, $b_2(x) = x$,..., $b_d(x) = x^d$
- Piecewise polynomial part : $b_{d+1}(x) = (x - a)_+^d$, $b_{d+2}(x) = (x - b)_+^d$

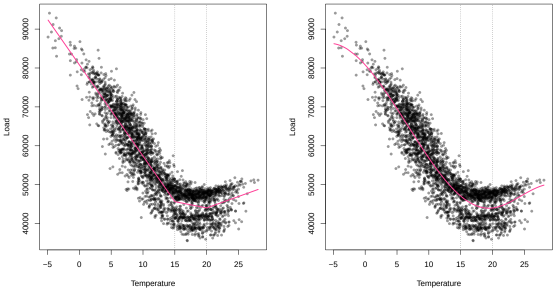


Figure – Truncated power functions regression.

B-splines

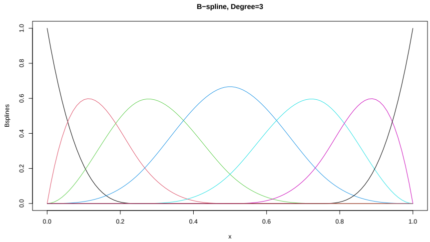


Figure – B-splines.

B-spline :

- a commonly used spline basis
- local support : high numerical stability
- efficient computation (recursive algorithm)

Natural splines

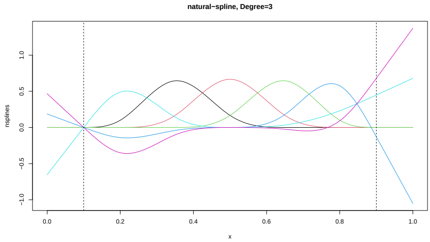


Figure – Natural splines.

Natural spline :

- splines can be erratic at the boundaries of the data
- natural splines are cubic splines + additional constraints that they are linear in the tails of the boundary knots ($f'' = 0$)

Cyclic splines

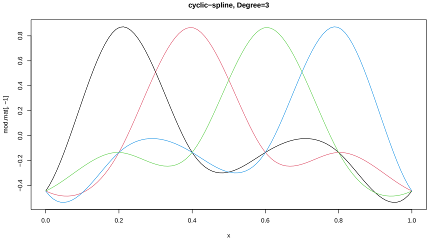


Figure – Cyclic splines.

Cyclic spline :

- penalized cubic regression splines whose ends match, up to second derivative
- useful to model periodic effects

GAM

We consider now the simplified univariate Gaussian model :

$$y_i = f(x_i) + \varepsilon_i$$

where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, and f is smooth (belong to the previous sobolev space $W_2^m[a,b]$). We suppose that we observe a sample of observations $(x_i, y_i)_{i=1,\dots,n}$.

The trade-off between a good fit of the data and the smoothness of f is achieved by solving the following penalized least square pb :

$$\min_f \underbrace{\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2}_{\text{goodness of fit}} + \lambda \underbrace{\int_a^b f^{(m)}(x)^2 dx}_{\text{roughness}}$$

where $\lambda > 0$, the smoothing parameter, controls the trade off between goodness of fit and roughness.

- cubic spline is a special case with $m = 2$
- no penalty for polynomials of order less than or equal to m
- for a given λ this problem as a unique minimizer in the space of natural polynomial spline of order m with knots $(x_1, ..., x_n)$, see [Wan11].

GAM

Practically, we choose a spline basis (and associated knots), then the pb reduces, for a given λ to a ridge regression problem :

$$\|\boldsymbol{Y} - \boldsymbol{X}\beta\|^2 + \lambda\beta^T S\beta$$

- $\boldsymbol{Y}$ vector of observation
- $\boldsymbol{X}$ the matrix with splines basis (columns), evaluate on x_i (lines)
- as f is linear in the parameters, β_i, $\int_a^b f^{(m)}(x)^2 dx$ could be written $\beta^T \boldsymbol{S}\beta$

Thus leading to the following estimator of β :

$$\widehat{\beta}_\lambda = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{S})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$$

GAM

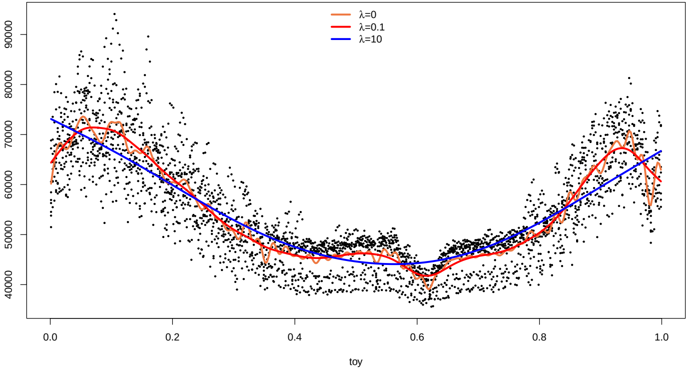


Figure – the smoothing parameter λ controls the trade off between goodness of fit and roughness.

GAM

- λ is a crucial parameter, our objective is to minimize a generalization error.
- To do that we need to find a criteria that could be minimized in order to respect a bias-variance trade-off :
  - AIC (Akaike Informattion Criteria) : $\text{RSS} + 2\text{df}/n$
  - BIC (Bayesian Information Criteria) : $\text{RSS} + \log(n)\text{df}/n$
  - Mallows' Cp : $\text{RSS} + 2\sigma^2\text{df}/n$
  - CV (cross validation) : $\frac{1}{n}\sum_{i=1}^{n}\frac{(y_i - f(x_i))^2}{(1-H_{i,i})^2}$
  - GCV (Generalized Cross Validation) : $\frac{1}{n}\sum_{i=1}^{n}\frac{(y_i - f(x_i))^2}{(1-tr(H)/n)^2}$

  where $\text{RSS} = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$ (could be generalized using the deviance, 2 times the log-likelihood ratio of the full model compared to the reduced model) and $\boldsymbol{H} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{S})^{-1}\boldsymbol{X}^T$.

  For all these criteria, the notion of **degrees of freedom** (effective number of parameter) is crucial as it allows to penalize complex model relative to simple ones (Occam's razor : *the model that fits observations sufficiently well in the least complex way should be prefered*).

Degrees of freedom

Let consider a linear Gaussian model :

$$\boldsymbol{Y} = \boldsymbol{X}\beta + \varepsilon, \varepsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$$

Using the least square method : $\widehat{\boldsymbol{Y}} = \boldsymbol{HY}$ and its i^e component $\widehat{y}_i = \sum_{j=1}^{n} h_{i,j} y_j$.

The degrees of freedom in regression = the number of parameters in the model, but can also be expressed as :

$$p = tr(\boldsymbol{H}) = \sum_i h_{i,i} = \sum_i \frac{\partial \widehat{y}_i}{\partial y_i}$$

the **degrees of freedom** are the sum of **sensitivities** of the fitted values $\widehat{y}_i$ with respect to observation y_i.
Generalizing to any linear smoothing estimation strategy where $\boldsymbol{H} = \boldsymbol{X}(\boldsymbol{X}^T \boldsymbol{X} + \lambda \boldsymbol{S})^{-1} \boldsymbol{X}^T$ depends on λ.

Degrees of freedom :

Another way to explain it Intuitively :

- estimate of $f$without penalization ($\lambda = 0$) : $\widehat{f}(0) = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$
- estimate of f with penalization ($\lambda > 0$) :$\widehat{f}(\lambda) = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{S})^{-1}\boldsymbol{X}^T\widehat{f}(0)$

entails that

$$\widehat{f}(\lambda) = \boldsymbol{H}\widehat{f}(0)$$

df is thus the dimension of the subspace spanned by $\boldsymbol{H}$ (linear operator of the penalized regression).

2 dimensional smoothing

Suppose now that $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2})$ is 2-dimensional, penalised spline regression can be performed :

$$\min_f \sum_{i=1}^{n} (y_i - f(\boldsymbol{x})^2 + \lambda \text{pen}(f)$$

where f can be represented using a tensor product basis :

$$\alpha_{j,k}(x) = a_j(x_1) b_k(x_2), j = 1, ..., J, k = 1, ..., K$$

and, for cubic splines :

$$\text{pen}(f) = \int \int \frac{\partial^2 f(x)}{\partial x_1^2}^2 + 2(\frac{\partial^2 f(x)}{\partial x_1 \partial x_2})^2 + \frac{\partial^2 f(x)}{\partial x_2^2}^2 dx_1 dx_2$$

GAM

Now we have the tools to consider a Gaussian GAM :

$$y_i = \boldsymbol{X}_i\beta + f_1(x_{1,i}) + f_2(x_{2,i}) + f_3(x_{3,i}, x_{4,i}) + ... + \varepsilon_i$$

- $\boldsymbol{X}_i\beta$ linear part of the model
- f_j are smooth functions
- ε_i are iid $\mathcal{N}(0, \sigma^2)$
- identifiability constraint has to be imposed otherwise each functional effects are estimable up to an additive constant.

mgcv basics

In mgcv, GAMs can be built and fitted using the gam function, an example call being

```
fit <- gam(formula = y ~ x1 + s(x2, k = 15, bs = "cr") + s(x3, x4, k=50),
family = Poisson(link = log), data = SomeData)
```

- first argument : model formula, where we are using a linear effect for covariate x1, a smooth effect for x2 and a bivariate smooth effect for the interaction(x2, x3) .
- arguments bs and k of the smooth effect specifier (default is thin plate), s, determine the type and number of basis functions used.
- last argument determines the response distribution to be used, here a Poisson distribution where the linear predictor is modelling $\log \mu(x_1, x_2) = \log \mathbb{E}(y|x_1, x_2)$. Under such model, one reason for using the log-link, $g = \log$, is to ensure the positivity of $\mu(x_1, x_2)$.
- an alternative to gam is bam for big additive models (multicore optimization of GAM) [WGS15].

mgcv basics

The function s has different arguments

```
fit <- gam( y ~ s(x, k = 15, bs = "cr")
```

- k : dimension of the basis used to represent the smooth term, more precisely the maximum number of df
- bs indicated the smoothing basis
  - bs="tp", thin platte regression splines
  - bs="ds", Duchon splines
  - bs="cr", cubic regression splines
  - bs="cc", cyclic cubic regression splines
  - bs="ps", P-splines (B-spline with a discrete penalty on the basis coefficient)
  - bs="ad" adaptive smooth (λ depends on x)

```
fit1 <- gam( y ~  s(x3, x4, k=50))
```

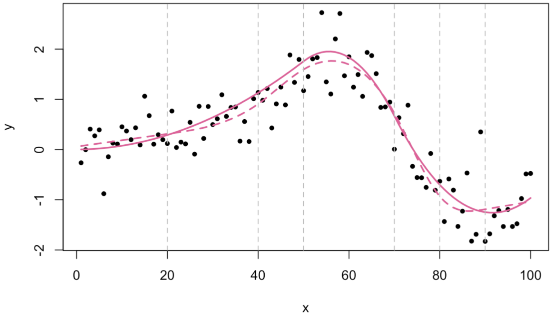
```
fit2 <- gam( y ~  te(x3, x4, k=c(5, 10)))
```

- bivariate effects can be entered either with the syntax s (one smoothness parameter, one df) or te (two smoothness parameters and dfs)

mgcv basics

Setting the position of knots (by defaut a regular partition of the quantiles) :

```
knots <- c(20, 40, 50, 70, 80, 90)
g <- gam(y ~ s(x, k = 15, bs = "cr"), knots=list(x=knot), sp=0)
```



mgcv basics

The `by` option :

- interaction with qualitative variable, a smooth effect per level is fitted :

  ```
  g <- gam(y ~ s(x, by=u)+u)
  ```

- functional GLM model of the form : $y_i = \int v_i(t) f(t) dt + \varepsilon_t$ can be estimated by :

  ```
  g <- gam(y ~ s(T, by=V))
  ```

  where T and V are matrices , discretized observations of $v_i(t)$ at $(t_1, ..., t_K)$ is the i^{th} row of V. Each row of T is a replicate of the (time) observations vector $(t_1, ..., t_K)$.

mgcv basics

Model summary

```
g <- gam(y ~ s(x, k = 10, bs = "cr") + s(z, k=10, bs='cr'), ...)
summary(g)
```

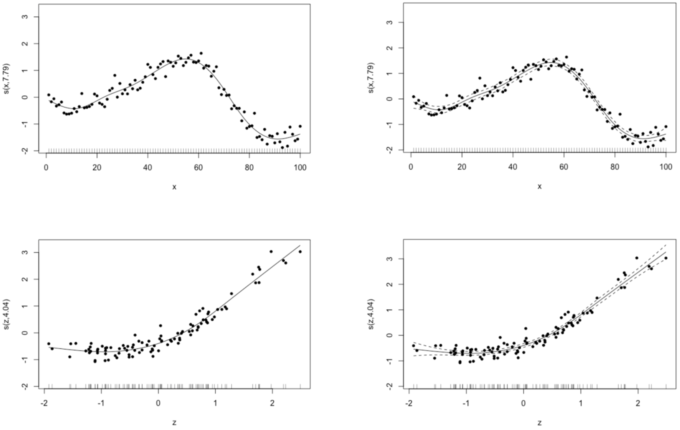
Linear terms →

Smooth terms →

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## y ~ s(x, k = 10, bs = "cr") + s(z, k = 10, bs = "cr")
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.18684    0.02436   48.71   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##        edf Ref.df     F p-value
## s(x) 7.254  8.258 204.3  <2e-16 ***
## s(z) 8.555  8.920 178.4  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =   0.97   Deviance explained = 97.5%
## GCV = 0.071359  Scale est. = 0.059364  n = 100
```

mgcv basics

```
g <- gam(y ~ s(x, k = 10, bs = "cr") + s(z, k=10, bs='cr'), ...)
plot(g, residuals=T, rug=T, se=F, pch=20)
```



mgcv basics

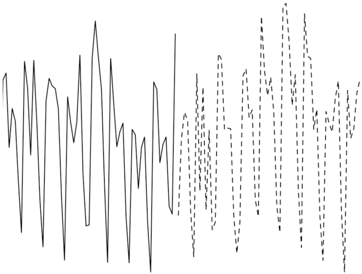
Forecasting :

- global forecast

```
g <- gam(...)
g.forecast <- predict(g, newdata=data1)
```

- per effect :

```
g.forecastt <- predict(g, newdata=data1, type='terms')
```



Auto-correlated data

For time series data the assumption of ε_i being iid is not satisfied.
Different options are then possible :

- weighted least square to fit a GAM with an AR(1) structure of the residuals
- lags as covariate : $y_t = f(y_{t-1}) + \varepsilon_t$ (use with care if you want to keep things interpretable)
- a two-step procedure : fit a GAM, than an ARIMA (or other time series models) model on the residuals

Time varying GAM

In practice, data often evolves with time : distribution shiftt, structural breaks...
Adaptation of the GAM over time is driven by a trade-off reactivity to a change/complexity of the model.
Re-estimated a full GAM often involves too much df to perform well (necessitate a too long history of data). To reduce the dimension of the adaptation problem, a strategy is to freeze the nonlinear effects, and to correct these effects by a multiplicative factor :

- we define $f(\boldsymbol{x}_t) = (1, \overline{f}_1(x_{t,1}), ..., \overline{f}_d(x_{t,d}))^\top$ where $\overline{f}_j$ is a normalized version of f_j obtained by subtracting the mean on the train set and dividing by the standard deviation.
- then we adaptively estimate a vector $\boldsymbol{\theta}_t$ such that

$$\mathbb{E}[y_t \mid \boldsymbol{x}_t] = \boldsymbol{\theta}_t^\top f(\boldsymbol{x}_t)\,.$$

We can then use different *online* linear strategies to update θ_t optimally.

exp-LS

Exponential weighted Least-Squares (exp-LS) :

- we solve at each step a least-squares problem with weight decreasing exponentially with the time difference :

$$\hat{\boldsymbol{\theta}}_t \in \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \sum_{s=1}^{t-1} e^{-\mu(t-s)} \left(y_s - \boldsymbol{\theta}^\top f(\boldsymbol{x}_s) \right)^2 ,$$

- we predict $\hat{y}_t = \hat{\boldsymbol{\theta}}_t^\top f(\boldsymbol{x}_t)$.

This formalisation leads to a single parameter, the exponential forgetting factor μ.
The forgetting factor μ is determined by minimizing the RMSE on a validation set (e.g. the last year of the train set) then we keep the same μ for the GAM trained on the whole train set.
Previous work has been done on estimating this parameter online, but leads to computational issues and potential instability of the model (see [Ba+12]).

Kalman Filter

We consider a state-space model approach, the setting of Kalman filtering [KO60]

$$y_t = \boldsymbol{\theta}_t^\top f(\boldsymbol{x}_t) + \varepsilon_t \,,$$
$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \boldsymbol{\eta}_t \,,$$

where

- (ε_t) and $(\boldsymbol{\eta}_t)$ are gaussian white noises of respective variance / covariance σ^2 and Q.
- the recursive formulae of Kalman provides the expectation and covariance of the state $\boldsymbol{\theta}_t$ given the past observations.
- these estimators yield the mean and variance of y_t given the past.

rq : the exp-LS method has a very similar recursive form. Its simplicity stands in a single scalar parameter e^μ as multiplicative factor for the update of P_t, whereas Kalman Filter needs a matrix parameter Q added in the recursion.

Kalman Filter

initialization : the prior $\boldsymbol{\theta}_1 \sim \mathcal{N}(\widehat{\boldsymbol{\theta}}_1, P_1)$ where $P_1 \in \mathbb{R}^{d\times d}$ is positive definite and $\widehat{\boldsymbol{\theta}}_1 \in \mathbb{R}^d$;

Recursion : at each time step $t = 1, 2, \ldots$

1. Prediction :

$$\mathbb{E}\left[y_t \mid (\boldsymbol{x}_s, y_s)_{s<t}, \boldsymbol{x}_t\right] = \widehat{\boldsymbol{\theta}}_t^\top f(\boldsymbol{x}_t)\,,$$
$$Var\left[y_t \mid (\boldsymbol{x}_s, y_s)_{s<t}, \boldsymbol{x}_t\right] = \sigma^2 + f(\boldsymbol{x}_t)^\top P_t f(\boldsymbol{x}_t)\,.$$

2. Estimation :

$$\widehat{\boldsymbol{\theta}}_{t+1} = \widehat{\boldsymbol{\theta}}_t + \frac{P_t f(\boldsymbol{x}_t)}{f(\boldsymbol{x}_t)^\top P_t f(\boldsymbol{x}_t) + \sigma^2}(y_t - \widehat{\boldsymbol{\theta}}_t^\top f(\boldsymbol{x}_t))\,,$$
$$P_{t+1} = P_t - \frac{P_t f(\boldsymbol{x}_t) f(\boldsymbol{x}_t)^\top P_t}{f(\boldsymbol{x}_t)^\top P_t f(\boldsymbol{x}_t) + \sigma^2} + Q\,.$$

GAM for load consumption

GAM is currenttly in use in operation at EDF for load consumption forecasting. Operational models are of the form :

$$\mathrm{Load}_t = \sum_{i=1}^{7}\sum_{j=0}^{1} \alpha_{i,j}\mathbb{1}_{\mathrm{DayType}_t=i}\mathbb{1}_{\mathrm{DLS}_t=j} + f_1(t) + f_2(\mathrm{ToY}_t)$$
$$+ \sum_{i=1}^{7} \beta_i \mathrm{Load1D}_t \mathbb{1}_{\mathrm{DayType}_t=i} + \gamma \mathrm{Load1W}_t$$
$$+ f_3(t, \mathrm{Temp}_t) + f_4(\mathrm{Temp95}_t) f_5(\mathrm{Temp99}_t) + f_6(\mathrm{TempMin99}_t, \mathrm{TempMax99}_t) + \varepsilon_t$$

where at each day t :

- $\mathrm{DayType}_t$ is a categorical variable indicating the type of the day of the week,
- DLS_t is a binary variable indicating whether t is in summer hour or winter hour,
- Load1D and Load1W are the load of the day before and the load of the week before,
- ToY_t is the time of year whose value grows linearly from 0 on the 1^{st} of January 00h00 to 1 on the 31^{st} of December 23h30,
- Temp_t is the national average temperature,
- $\mathrm{Temp95}_t$ and $\mathrm{Temp99}_t$ are exponentially smoothed temperatures of factor $\alpha = 0.95$ and 0.99. E.g. for $\alpha = 0.95$ at a given instant i,
  $\mathrm{Temp95}_i = \alpha \mathrm{Temp95}_{i-1} + (1-\alpha)\mathrm{Temp}_i$,
- $\mathrm{TempMin99}_t$ and $\mathrm{TempMax99}_t$ are the minimal and maximal value of $\mathrm{Temp99}_t$ at the current day.

mgcv basics

This model can be fitted with `mgcv`, using tthe following formula :

```
fit <- gam(formula = y ~ x1 + s(x2, k = 15, bs = "cr"),
family = Poisson(link = log), data = SomeData)
```

- first argument : model formula, where we are using a linear effect for covariate `x1` and a smooth effect for `x2`.
- arguments `bs` and `k` of the smooth effect specifier, `s`, determine the type and number of basis functions used (see Section **??** for more details).
- last argument determines the response distribution to be used, here a Poisson distribution where the linear predictor is modelling $\log \mu(x_1, x_2) = \log \mathbb{E}(y|x_1, x_2)$. Under such model, one reason for using the log-link, $g = \log$, is to ensure the positivity of $\mu(x_1, x_2)$.

References I

[Wah80] Grace Wahba. **Spline bases, regularization, and generalized cross validation for solving approximation problems with large quantities of noisy data**. University of WISCONSIN, 1980.

[HT86] Trevor Hastie and Robert Tibshirani. « Generalized Additive Models ». In: **Statistical Science** 1.3 (1986), pp. 297–318.

[EM96] Paul H. C. Eilers and Brian D. Marx. « Flexible smoothing with B-splines and penalties ». In: **Statistical Science** 11.2 (1996), pp. 89 –121. DOI: 10.1214/ss/1038425655. URL: https://doi.org/10.1214/ss/1038425655.

[Woo03] Simon N Wood. « Thin plate regression splines ». In: **Journal of the Royal Statistical Society: Series B (Statistical Methodology)** 65.1 (2003), pp. 95–114.

[Woo17] Simon N. Wood. **Generalized Additive Models: an Introduction with R**. 2nd ed. Boca Raton: Chapman & Hall/CRC, 2017.

[Wan11] Yuedong Wang. **Smoothing splines: methods and applications**. CRC press, 2011.

[WGS15] Simon N Wood, Yannig Goude, and Simon Shaw. « Generalized additive models for large data sets ». In: **Journal of the Royal Statistical Society: Series C (Applied Statistics)** 64.1 (2015), pp. 139–155.

[Ba+12] Amadou Ba et al. « Adaptive learning of smoothing functions: Application to electricity load forecasting ». In: **Advances in neural information processing systems**. 2012, pp. 2510–2518.

[KO60] Rudolph Emil Kalman and Others. « A new approach to linear filtering and prediction problems ». In: **Journal of basic Engineering** 82.1 (1960), pp. 35–45.