Generalized additive models for electricity demand forecasting

Matteo Fasiolo (University of Bristol, UK)

Joint work with:

Simon N. Wood (University of Bristol, UK)
Yannig Goude (EDF R&D)
Raphaël Nedellec (Talend, formerly EDF R&D)

matteo.fasiolo@bristol.ac.uk

February 28, 2019

Talk structure

1. GAMs for electricity load forecasting
2. More scalability: GAMs for Big Data
3. More flexibility: GAMLSS and quantile GAMs
4. Current and future work

Introduction to GAMs

Generalized **additive** model (GAM) (Hastie and Tibshirani, 1990):

$$\mathsf{Load}_i|\mathbf{x}_i \sim \mathsf{Distr}\{\mathsf{Load}_i|\theta_1 = \mu(\mathbf{x}), \theta_2, \dots, \theta_p\},$$

where

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

and g is the link function.

f_j's can be fixed (parametric) or smooth effects with coefficients $\boldsymbol{\beta}$.

$\theta_2, \dots, \theta_p$ control scale, shape of distribution.

Introduction to GAMs

Example: a Gaussian GAM for expected load is

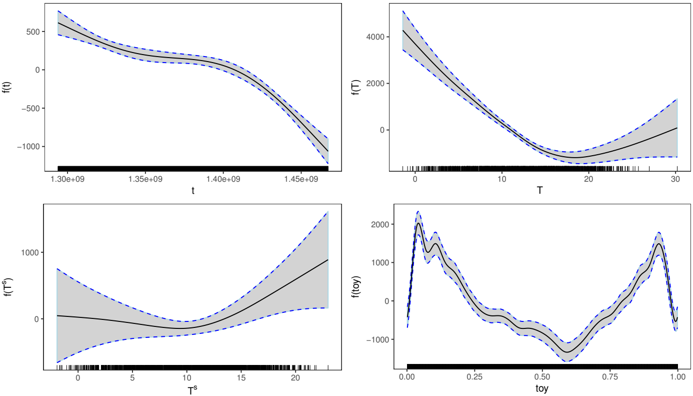
$$
\begin{aligned}
\mathbb{E}(\texttt{Load}_i) &= \sum_{j=1}^{7} \beta_j w_{d(i)}^{j} && \cdot \text{ Day-of-week factor} \\
&+ \beta_8 \texttt{Load}_{i-48} && \cdot \text{ Lagged load} \\
&+ f_1(t_i) && \cdot \text{ Long-term trend} \\
&+ f_2(T_i) && \cdot \text{ Temperature} \\
&+ f_3(T_i^s) && \cdot \text{ Smoothed temperature} \\
&+ f_4(\texttt{toy}_i), && \cdot \text{ Time-of-year}
\end{aligned}
$$

where $T_i^s = \alpha T_i + (1 - \alpha) T_{i-1}^s$, with $\alpha = 0.05$.

Introduction to GAMs

Using mgcv R package (Wood, 2001):

```
fit <- gam(load ~ dow + load48 + s(time) + s(temp) +
                  s(tempSmo) + s(toy),
           family = gaussian, data = UKload)
```



Introduction to GAMs

Recall model structure:

$$\mathbb{E}(\text{Load}|\mathbf{x}) = g^{-1}\Big\{ f_1(\mathbf{x}) + f_2(\mathbf{x}) + \cdots \Big\},$$

Smooth effects built using spline bases

$$f_j(\mathbf{x}) = \sum_{k=1}^{r} \beta_k b_k(\mathbf{x})$$

where b_k's are known, β_k's unknown.

To determine complexity of $f_j(\mathbf{x})$:

- the basis rank r is large enough for sufficient flexibility
- a complexity penalty on $\boldsymbol{\beta}$ controls the wiggliness of the effects

Introduction to GAMs

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

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

where:

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

Concrete example $\mathbb{E}(\text{Load}|\mathbf{x}) = f(x_1) + g(x_2, x_3)$:

$$\text{Pen}(\boldsymbol{\beta}|\boldsymbol{\gamma}) \approx \gamma_1 \int f_{x_1 x_1}^2 dx_1 + \gamma_2 \int g_{x_2 x_2}^2 + 2g_{x_2 x_3}^2 + g_{x_3 x_3}^2 dx_2 dx_3$$

mgcv uses a hierarchical fitting framework:

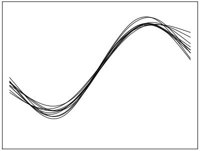
1. Select γ to determine smoothness

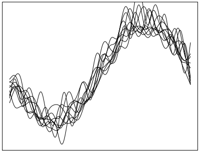
$$\hat{\gamma} = \underset{\gamma}{\text{argmax}}\ \text{LAML}(\gamma)$$

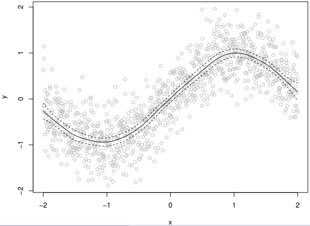
where $\text{LAML}(\gamma) \approx p(y|\gamma) = \int p(y, \boldsymbol{\beta}|\gamma) d\boldsymbol{\beta}$.

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

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







Introduction to GAMs

There are alternatives to `mgcv`, such as:

- `mboost` (Hothorn et al., 2010)
- `gamlss` (Rigby and Stasinopoulos, 2005)
- `brms` (Bürkner et al., 2017)
- `BayesX` (Brezger et al., 2003)

Each offers much flexibility (e.g. classes of smooth effects and distributions).

Practical advantages of `mgcv`'s GAM fitting methods:

1. little tuning needed (automatic smoothing selection parameter)
2. efficient and stable numerical implementation

Next we'll see how these can be scaled to large data sets.

Talk structure

1. GAMs for electricity load forecasting
2. More scalability: GAMs for Big Data
3. More flexibility: GAMLSS and quantile GAMs
4. Current and future work

GAMs for Big Data

Example: a Gaussian GAM for expected load is

$$
\begin{aligned}
\mathbb{E}(\texttt{Load}_i) &= \sum_{j=1}^{7} \beta_j w^j_{d(i)} && \cdot \text{ Day-of-week factor} \\
&+ \beta_8 \texttt{Load}_{i-48} && \cdot \text{ Lagged load} \\
&+ f_1(t_i) && \cdot \text{ Long-term trend} \\
&+ f_2(T_i) && \cdot \text{ Temperature} \\
&+ f_3(T^s_i) && \cdot \text{ Smoothed temperature} \\
&+ f_4(\texttt{toy}_i), && \cdot \text{ Time-of-year}
\end{aligned}
$$

where $T^s_i = \alpha T_i + (1-\alpha) T^s_{i-1}$, with $\alpha = 0.05$.

It is standard practice to model the 48 30min slots separately.

So we need to fit 48 models.

GAMs for Big Data

Example: a more ambitious model is

$$
\begin{aligned}
\mathbb{E}(\texttt{Load}_i) &= \sum_{j=1}^{7} \beta_j w^j_{d(i)} && \cdot \text{ Day-of-week factor} \\
&+ f(\text{tod}_i)\texttt{Load}_{i-48} && \cdot \text{ Lagged load} \\
&+ \text{te}_1(t_i, \text{tod}_i) && \cdot \text{ Long-term trend} \\
&+ \text{te}_2(T_i, \text{tod}_i) && \cdot \text{ Temperature} \\
&+ \text{te}_3(T^s_i, \text{tod}_i) && \cdot \text{ Smoothed temperature} \\
&+ \text{te}_4(\texttt{toy}_i, \text{tod}_i), && \cdot \text{ Time-of-year}
\end{aligned}
$$

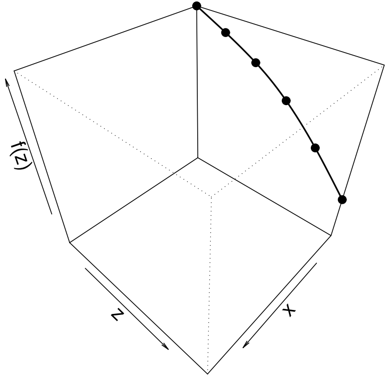
where

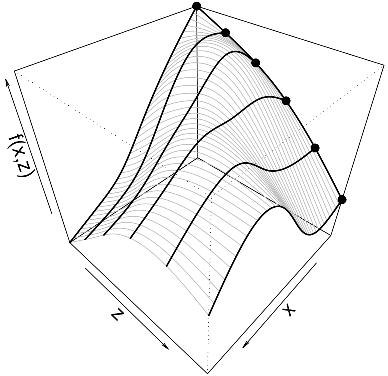
- `tod` is time of day $1, \dots, 48$
- `te`'s are 2D tensor product smooths
- $f(\text{tod}_i)\texttt{Load}_{i-48}$ is varying coefficient effect

GAMs for Big Data

Tensor products `te(x, z)` useful when `x` and `z` are different units (e.g. `x` $=$ temper, `y` $=$ time of day).

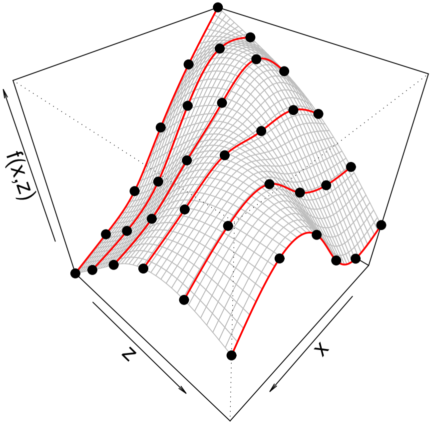
Construction: make a spline $f_z(z)$ a function of x by letting its coefficients vary smoothly with x

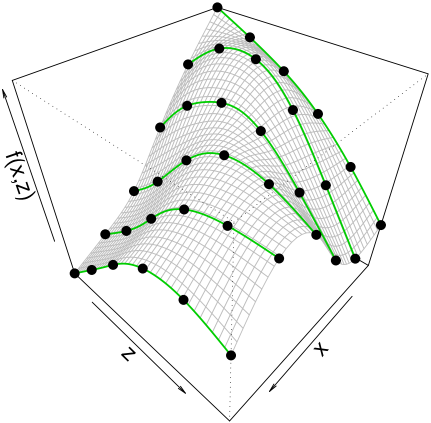




GAMs for Big Data

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

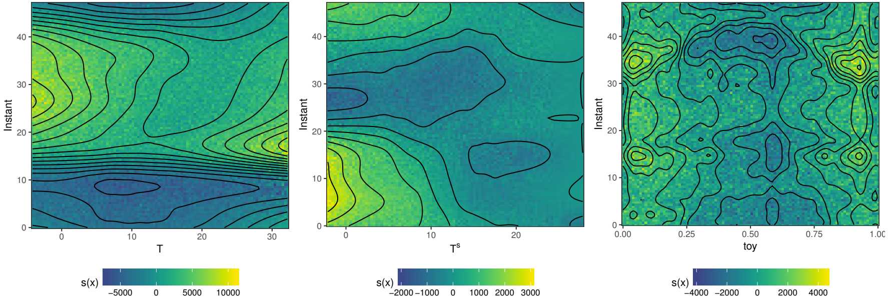




GAMs for Big Data

We can use mgcv's Big Data methods to model all 48 time slots jointly:

```
fit <- bam(load ~ dow + s(tod, by = load48) +
              te(time, tod) + te(toy, tod) +
              te(temp, tod) + te(tempSmo, tod),
          data = UKload,
          discrete = TRUE,
          nthreads = 16)
```



GAMs for Big Data

Why is this useful? Some answers:

- statistical efficiency $\rightarrow$ share information across time-of-day (a.k.a. instant)
- ease of use and interpretation

To see why we need Big Data methods notice that:

- n is 48 times bigger than a 30min model
- tensor product can have large number of basis functions

$$\text{te}(\text{T}, \text{tod}) = \sum_{j=1}^{J} \sum_{k=1}^{K} \beta_{ij} b_j(\text{T}) b_k(\text{tod}) = \sum_{j=1}^{J} \sum_{k=1}^{K} \beta_{ij} \tilde{b}_{jk}(\text{T}, \text{tod})$$

so tensor effect has $J \times K$ coefficients.

GAMs for Big Data

Recall that we are modelling $\mathbb{E}(\text{load}|\mathbf{x})$.

Here $\mathbb{E}(\text{load}|\mathbf{x}_i)$ can be written as $\mathbf{X}_i\boldsymbol{\beta}$, where $\mathbf{X}_i$ row of

$$\mathbf{X} = \left[\begin{array}{ccccccc} 1 & \mathbb{1}(\text{dow}_1 = \text{Mon}) & \cdots & b_{11}(\text{T}_1, \text{tod}_1) & \cdots & b_{JK}(\text{T}_1, \text{tod}_1) & \cdots \\ 1 & \mathbb{1}(\text{dow}_2 = \text{Mon}) & \cdots & b_{11}(\text{T}_2, \text{tod}_2) & \cdots & b_{JK}(\text{T}_2, \text{tod}_2) & \cdots \\ . & \cdots & . & . & \cdots & . & \cdots \\ . & . & \cdots & . & \cdots & . & \cdots \end{array} \right]$$

with n rows and

$$d = p + J \times K + \cdots,$$

columns.

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. $\mathbf{X}^\text{T}\mathbf{X}$ or $\text{QR}(\mathbf{X})$) takes time

GAMs for Big Data

`bam()` implements memory-saving methods of Wood et al. (2015):

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

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} \\ \mathbf{X}_{21} & \mathbf{X}_{22} \\ \vdots & \vdots \\ \mathbf{X}_{B1} & \mathbf{X}_{B2} \end{bmatrix}$$

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

Block-oriented methods can be used also to perform fast model updates:

```
fit <- bam.update(fit, data = newData, chunk.size = 1e4)
```

GAMs for Big Data

Faster computation and memory savings using Wood et al. (2017).

Simple observation is that many variables are discrete in nature:

- time of day (tod) $\in \{1, \ldots, 48\}$
- time of year (toy) $\in \{1, \ldots, 365\}$
- temperature (T) $\in \{\ldots, -0.1, 0, 0.1, 0.2, \ldots\}$

There is room for data compression, example:

- we have 10 year of data and 48×365 obs per year
- effect of toy is

$$s(\text{toy}) = \sum_{i=1}^{p} \beta_i b_i(\text{toy}).$$

  so model matrix part $\mathbf{X}$ of toy is $(10 * 48 * 365) \times p$
- compressed model matrix $\bar{\mathbf{X}}$ is $365 \times p$
- saving factor $\#elem(\mathbf{X})/\#elem(\bar{\mathbf{X}}) = 10 * 48 * p$

GAMs for Big Data

Discretization can be applied to variables that are not "naturally" discrete.

Sampling variability is $O(n^{-\frac{1}{2}})$, so discretizing in $m = O(n^{\frac{1}{2}})$ bins is ok.

Wood et al. (2017) use discretization to fit UK black smoke pollution data from 2000 stations, with $n = 10^8$ and $p = 10^4$.

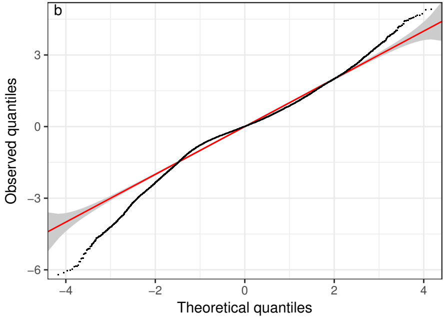
With latest `mgcv` version, the model

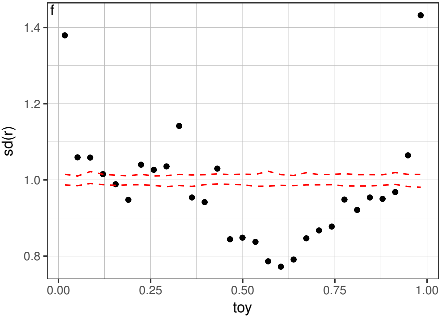
$$
\begin{aligned}
\log(\mathtt{bs}_i) &= f_1(\mathtt{y}_i) + f_2(\mathtt{doy}_i) + f_3(\mathtt{dow}_i) + f_4(\mathtt{y}_i, \mathtt{doy}_i) + f_5(\mathtt{y}_i, \mathtt{dow}_i) \\
&\quad + f_6(\mathtt{doy}_i, \mathtt{dow}_i) + f_7(\mathtt{n}_i, \mathtt{e}_i) + f_8(\mathtt{n}_i, \mathtt{e}_i, \mathtt{y}_i) + f_9(\mathtt{n}_i, \mathtt{e}_i, \mathtt{doy}_i) \\
&\quad + f_{10}(\mathtt{n}_i, \mathtt{e}_i, \mathtt{dow}_i) + f_{11}(\mathtt{h}_i) + f_{12}(\mathtt{T}_i^0, \mathtt{T}_i^1) + f_{13}(\bar{\mathtt{T}}\mathtt{1}_i, \bar{\mathtt{T}}\mathtt{2}_i) \\
&\quad + f_{14}(\mathtt{r}_i) + \alpha_{k(i)} + b_{\text{id}(i)} + e_i
\end{aligned}
$$

can be fitted in 5min on 8 cores.

GAMs for Big Data

```
fit <- bam(load ~ dow + s(tod, by = load48) +
              te(time, tod) + te(toy, tod) +
              te(temp, tod) + te(tempSmo, tod),
           data = UKload,
           discrete = TRUE,
           nthreads = 16)
```





Talk structure

1. GAMs for electricity load forecasting
2. More scalability: GAMs for Big Data
3. More flexibility: GAMLSS and quantile GAMs
4. Current and future work

From GAMs to GAMLSS

Generalized Additive Models for Location Scale and Shape (GAMLSS, Rigby and Stasinopoulos (2005)) let scale and shape change with **x**.

GAMLSS model structure:

$$\text{Load}|\mathbf{x} \sim \text{Distr}\{y|\theta_1 = \mu_1(\mathbf{x}), \theta_2 = \mu_2(\mathbf{x}), \dots, \theta_p = \mu_p(\mathbf{x})\},$$

where

$$\mu_1(\mathbf{x}) = g_1^{-1}\Big\{\sum_{j=1}^m f_j^1(\mathbf{x})\Big\},$$

$$\dots$$

$$\mu_p(\mathbf{x}) = g_p^{-1}\Big\{\sum_{j=1}^m f_j^p(\mathbf{x})\Big\},$$

and $g_1, \dots, g_p$ are link function.

From GAMs to GAMLSS

Example: Gaussian model for location and scale:

$$y|\mathbf{x} \sim N\{y|\mu(\mathbf{x}), \sigma(\mathbf{x})\}$$

where

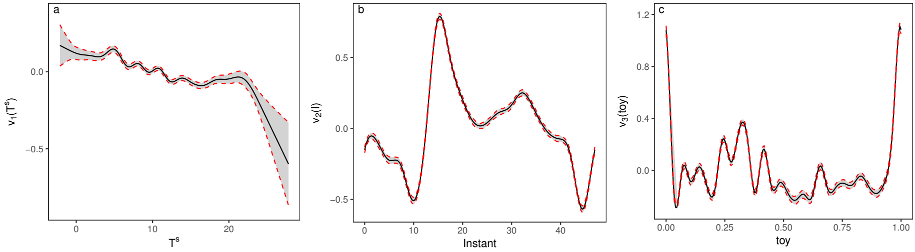
$$\mu(\mathbf{x}) = \sum_{j=1}^{m} f_j^1(\mathbf{x}), \quad \sigma(\mathbf{x}) = \exp\Big\{\sum_{j=1}^{m} f_j^2(\mathbf{x})\Big\}$$

and $g_2 = \log$ to guarantee $\sigma > 0$.

For electricity load forecasting:

```
fit <- gam(list(load ~ dow + te(time, instant) +
                       te(load48, instant) + ...,
                     ~ dow + s(temp) + s(instant) + s(toy)),
           family = gaulss, ...)
```

From GAMs to GAMLSS



GAMLSS models can be quite complex, eg. `shash` family from `mgcFam`:

```
fit <- gam(list(load ~ s(time) + ...,        # location
                     ~ s(temp) + ...,        # scale
                     ~ s(toy) + ...,         # skewness
                     ~ s(instant) + ...),    # kurtosis
           family = shash, ...)
```

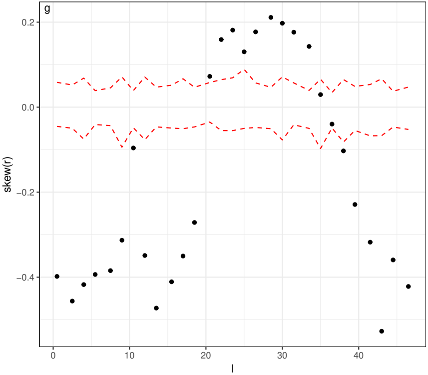
From GAMs to GAMLSS

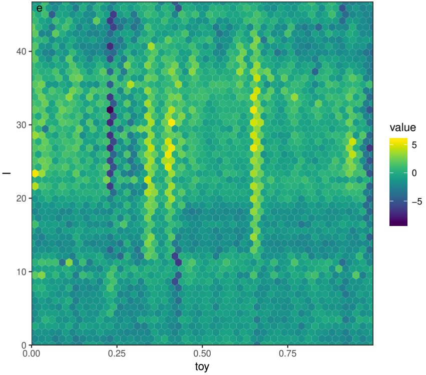
GAMLSS models are flexible but need to specify a model per parameter.

Exhaustive/automated model search very expensive.

Fasiolo et al. (2018) argue in favour of visual interactive model building.

New visual methods implemented by `mgcViz` R package.





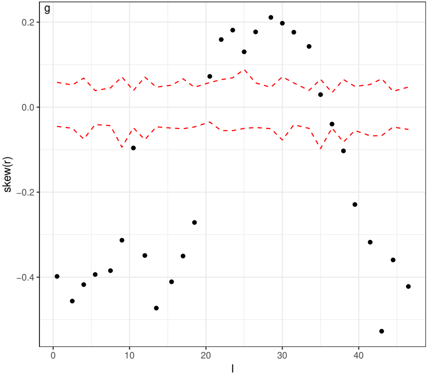
From GAMs to GAMLSS

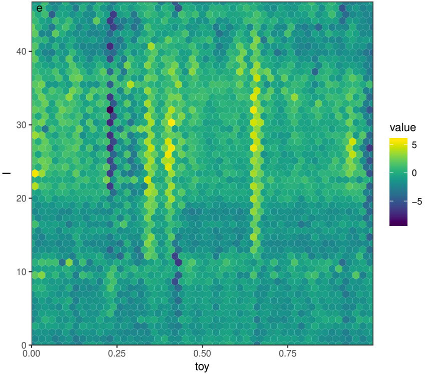
GAMLSS models are flexible but need to specify a model per parameter.

Exhaustive/automated model search very expensive.

Fasiolo et al. (2018) argue in favour of visual interactive model building.

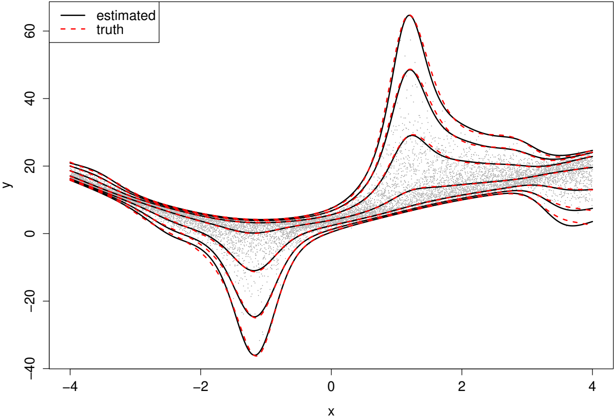
New visual methods implemented by `mgcViz` R package.





From GAMs to GAMLSS

GAMLSS useful to model whole distribution Load$|\mathbf{x}$ not just $\mathbb{E}(\text{Load}|\mathbf{x})$.



Still parametric assumption on Distr(load$|\mathbf{x}$).

In QGAMs we model quantiles individually.

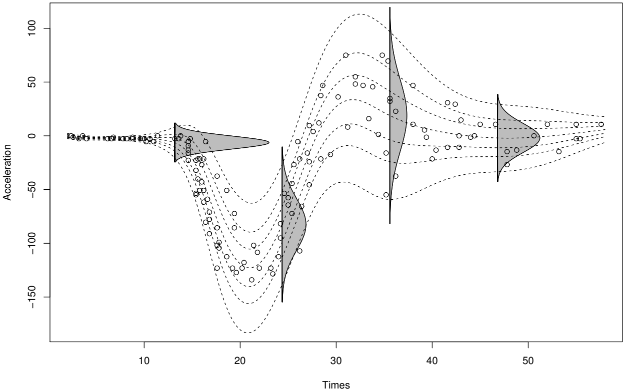
From GAMLSS to QGAM

Quantile regression models the τ-th quantiles of y, conditionally on $\mathbf{x}$.

Relevant for continuous y.

Define $F(y|\mathbf{x}) = \text{Prob}(Y \leq y|\mathbf{x})$.

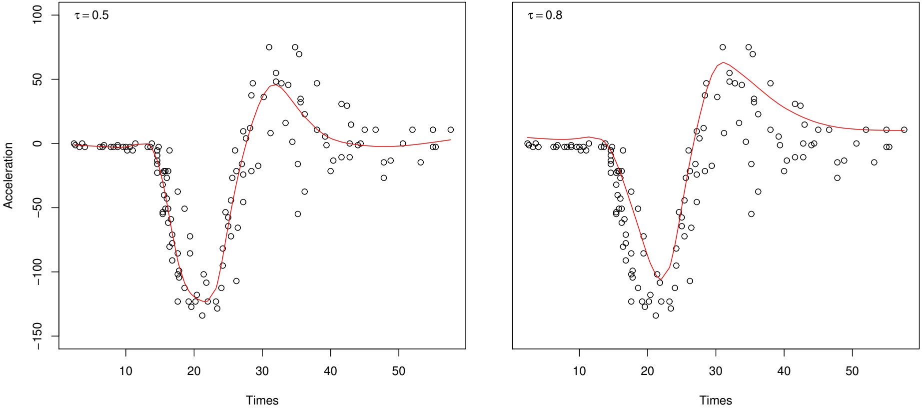
The τ-th ($\tau \in (0, 1)$) quantile is $\mu_\tau(\mathbf{x}) = F^{-1}(\tau|\mathbf{x})$.



From GAMLSS to QGAM

Quantile regression estimates conditional quantiles $\mu_\tau(\mathbf{x})$ directly.

No model for $p(y|\mathbf{x})$.



From GAMLSS to QGAM

The τ-th quantile is

$$\mu = F^{-1}(\tau|\mathbf{x}),$$

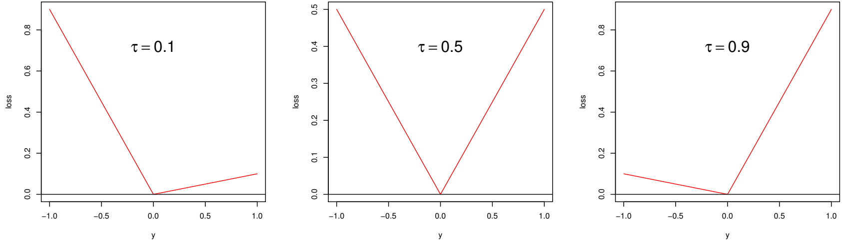
and it is the minimizer of

$$L(\mu|\mathbf{x}) = \mathbb{E}\{\ \rho_\tau(y-\mu)|\mathbf{x}\ \},$$

where

$$\rho_\tau(z) = (\tau - 1)z\mathbb{1}(z<0) + \tau z\mathbb{1}(z \geq 0),$$

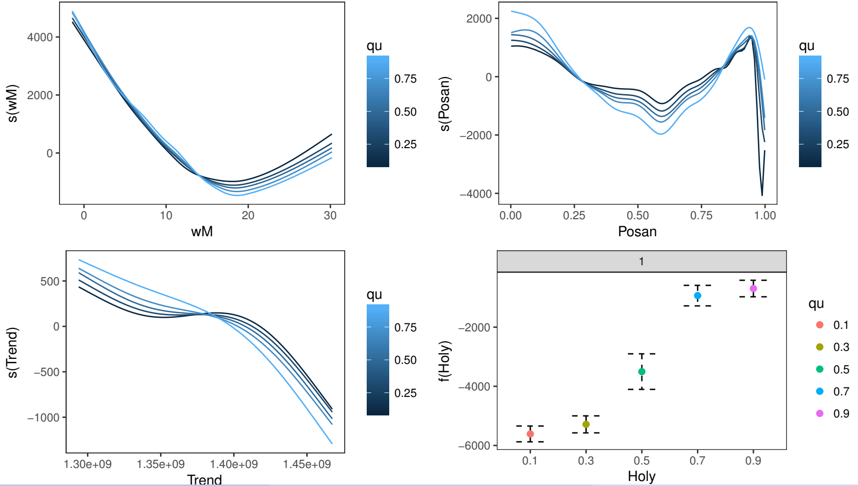
is the "pinball" loss.



From GAMLSS to QGAM

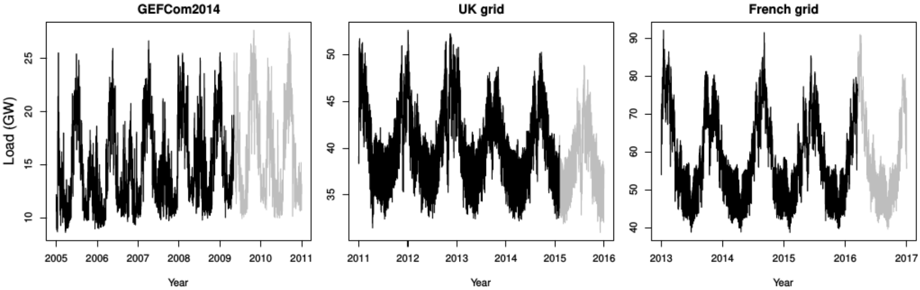
This is implemented by `qgam` R package (Fasiolo et al., 2018):

```
fit <- qgam(load ~ dow + s(load48) + ..., qu = 0.7)
```



From GAMLSS to QGAM

Consider three data sets and 11:30-12:00 daily slot.



Divided between training (black) and testing (grey).

Forecast 20 quantiles between $\tau = 0.05$ and $\tau = 0.95$.

Testing procedure, alternate the steps:

1. predict one week ahead and calculate pinball loss
2. refit using also data from that week

From GAMLSS to QGAM

Model is

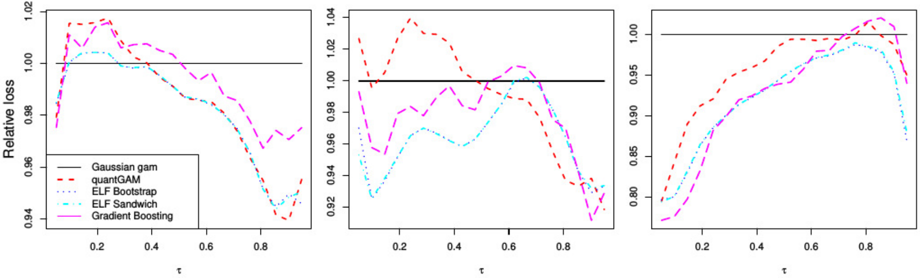
$$
\begin{aligned}
\mu_\tau(\texttt{Load}_i) &= \sum_{j=1}^{7} \beta_j w_{d(i)}^{j} && \cdot \text{ Day-of-week factor} \\
&+ s_1(\texttt{Load}_{i-48}) && \cdot \text{ Lagged load} \\
&+ s_2(t_i) && \cdot \text{ Long-term trend} \\
&+ s_3(T_i) && \cdot \text{ Temperature} \\
&+ s_4(T_i^s) && \cdot \text{ Smoothed temperature} \\
&+ s_5(\texttt{toy}_i), && \cdot \text{ Time-of-year}
\end{aligned}
$$

We consider:

- `qgam` with σ calibrated by bootstrapping (slow)
- `qgam` with σ calibrated by Bayesian sandwich (fast)
- quantGAM method of Gaillard et al. (2016)
- gradient boosting with `mboost` (Hothorn et al., 2010)

From GAMLSS to QGAM

Results on GEFCom, UK and French data



Nice that:

- bootstrap and sandwich QGAM give same results
- QGAM and quantGAM perform similarly on GEFCom data set

CPU times:

- sandwich: 1 to 4 seconds
- bootstrap: 15 to 30 seconds
- boosting: 90 to 700 seconds (but more in practice)

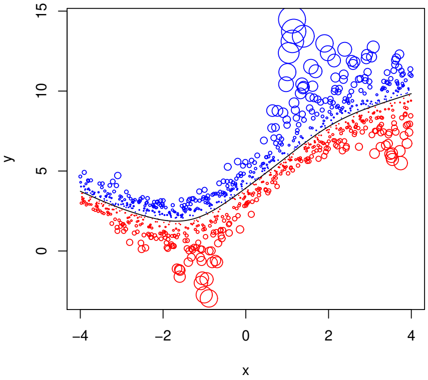
Talk structure

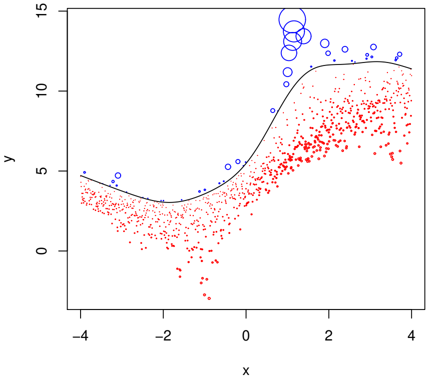
1. GAMs for electricity load forecasting
2. More scalability: GAMs for Big Data
3. More flexibility: GAMLSS and quantile GAMs
4. Current and future work

Extreme quantile regression

QGAMs are based on the pinball loss.

For extreme quantiles this is very asymmetric.





When predicting extreme demand, we expect high variance unless $n >> p$.

Extreme quantile regression

Distributional GAMs and GAMLSS have more bias, but less variance.

Could we improve predictions by mixing quantile and distributional GAMs?

Natural approach in GAM context is stacked regression (Breiman, 1996).

Let $\mu_j(\mathbf{x})$ be prediction for j-th model, fitted on training data.

Then we estimate the ensemble weights using fresh data:

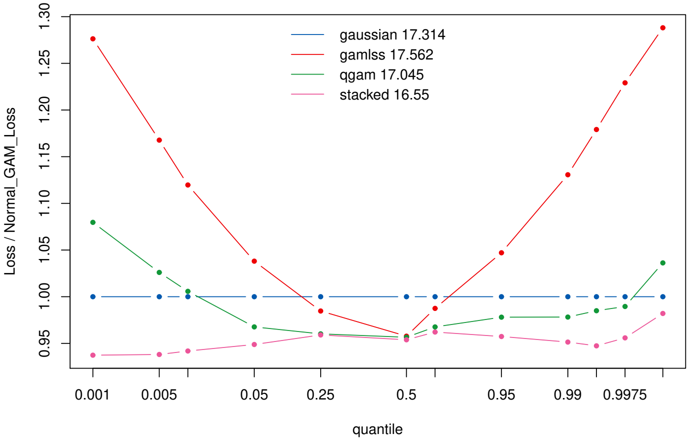
$$\boldsymbol{\alpha} = \underset{\boldsymbol{\alpha}}{\text{argmin}} \sum_{i=1}^{n} \rho_\tau \left\{ y_i - \sum_{j=1}^{m} \alpha_j \mu_j(\mathbf{x}_i) \right\}$$

where $\sum_{j=1}^{m} \alpha_j = 1$.

Difficulty: ρ_τ is highly skewed for $\tau \approx 1 \rightarrow$ high variance in stacking.

Candidate models for the stacked regression:

- QGAMs
- probabilistic GAMs
- Generalized Pareto GAMs (for extreme quantiles)

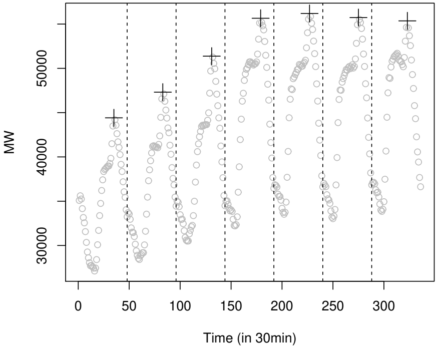


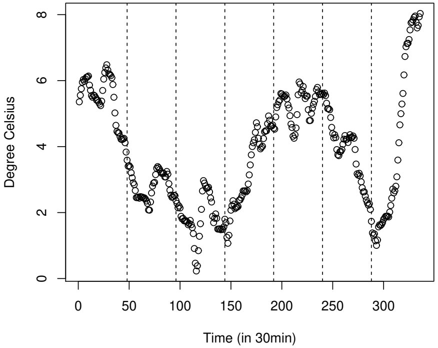
Multi-resolution GAMs for daily max

Related problem is modelling max demand over time horizon.

We have 30min electricity demand $L_{1:T}$, over n days.

We want to predict y_i, the maximal demand on the i-th day.





We need to deal with data at different resolutions.

Multi-resolution GAMs for daily max

Modelling approach:

- distribution for day max y_i is Generalized Extreme Value (GEV)
- capture information at 30min resolution using functional effects

1D functional effects:

- naive approach $\mathbb{E}(y_i) = \beta_1 \text{Temp}_1^i + \cdots + \beta_{48} \text{Temp}_{48}^i + \cdots$
- functional $\mathbb{E}(y_i) = \sum_{k=1}^{48} \beta(k) \text{Temp}_k^i + \cdots$ where $\beta(k)$ smooth

2D functional effects:

- naive approach $\mathbb{E}(y_i) = f_1(\text{Temp}_1^i) + \cdots + f_{48}(\text{Temp}_{48}^i) + \cdots$
- functional $\mathbb{E}(y_i) = \sum_{k=1}^{48} \text{te}(\text{Temp}_k^i, k) + \cdots$

Multi-resolution GAMs for daily max

Final model for daily max on UK data is

$$
\begin{aligned}
\mathbb{E}(y_i) = & \sum_{k=1}^{7} \beta_k \mathbb{I}(\mathsf{wd}_i = k) + s_1(\mathsf{toy}_i) + s_2(\mathsf{t}_i) \\
& + \sum_{k=1}^{48} \mathsf{te}_1(\mathsf{temp}_k^i, k) + \sum_{k=1}^{48} \mathsf{te}_2(\mathsf{tempS}_k^i, k) + \sum_{k=1}^{48} \mathsf{te}_3(\mathsf{L}_k^{i-1}, k),
\end{aligned}
$$

with $y_i \sim$ GEV.

RMSE on test set (last year of UK data):

- Multi-resolution: 14317 (best)
- Big model by-instant: 17889
- 48 models by-instant: 17228

Multi-resolution GAMs for daily max

Note y_i does not need to be daily max:

- total demand in a day ($y_i \sim$ Normal?)
- position of daily max ($y_i \in \{1, \dots, 48\}$, $y_i \sim$ OCAT?)

and functional structure stays the same.

We can be multi-resolution across space:

$$\mathbb{E}(W_i^{\text{tot}}) = \sum_k \text{te}(\text{lon}_k, \text{lat}_k, \hat{W}_k^i) + \cdots$$
$$\approx \int f\left\{\text{lon}, \text{lat}, \hat{W}(\text{lon}, \text{lat})\right\} d\text{lon}\, d\text{lat} + \cdots$$

Doable in `mgcv`, but see also `refund` package (Crainiceanu et al., 2012).

References I

Breiman, L. (1996). Stacked regressions. *Machine learning 24*(1), 49–64.

Brezger, A., T. Kneib, and S. Lang (2003). Bayesx: Analysing bayesian structured additive regression models. Technical report, Discussion paper//Sonderforschungsbereich 386 der Ludwig-Maximilians .

Bürkner, P. C. et al. (2017). brms: An r package for bayesian multilevel models using stan. *Journal of Statistical Software 80*(1), 1–28.

Crainiceanu, C., P. Reiss, J. Goldsmith, L. Huang, L. Huo, F. Scheipl, S. Greven, J. Harezlak, M. Kundu, and Y. Zhao (2012). refund: Regression with functional data. *R package version 0.1-6*.

Fasiolo, M., Y. Goude, R. Nedellec, and S. N. Wood (2018). Fast calibrated additive quantile regression. *arXiv preprint arXiv:1707.03307*.

Fasiolo, M., R. Nedellec, Y. Goude, and S. N. Wood (2018). Scalable visualisation methods for modern generalized additive models. *arXiv preprint arXiv:1809.10632*.

Gaillard, P., Y. Goude, and R. Nedellec (2016). Additive models and robust aggregation for gefcom2014 probabilistic electric load and electricity price forecasting. *International Journal of Forecasting*.

References II

Hastie, T. and R. Tibshirani (1990). *Generalized Additive Models*, Volume 43. CRC Press.

Hothorn, T., P. Bühlmann, T. Kneib, M. Schmid, and B. Hofner (2010). Model-based boosting 2.0. *The Journal of Machine Learning Research 11*, 2109–2113.

Rigby, R. A. and D. M. Stasinopoulos (2005). Generalized additive models for location, scale and shape. *Journal of the Royal Statistical Society: Series C (Applied Statistics) 54*(3), 507–554.

Wood, S. N. (2001). mgcv: Gams and generalized ridge regression for r. *R news 1*(2), 20–25.

Wood, S. N., Y. Goude, and S. Shaw (2015). Generalized additive models for large data sets. *Journal of the Royal Statistical Society: Series C (Applied Statistics) 64*(1), 139–155.

Wood, S. N., Z. Li, G. Shaddick, and N. H. Augustin (2017). Generalized additive models for gigadata: modeling the uk black smoke network daily data. *Journal of the American Statistical Association 112*(519), 1199–1210.