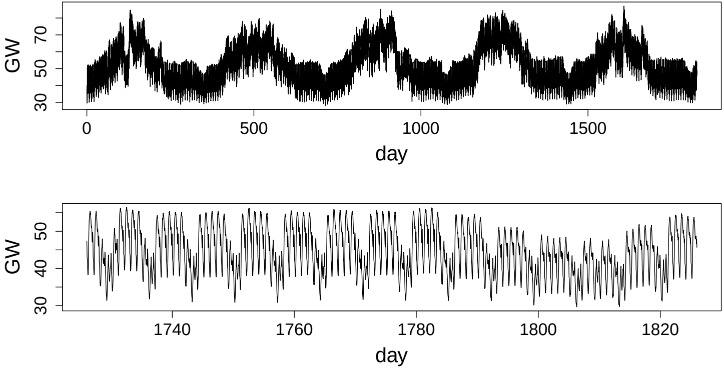
GAM for large datasets and load forecasting

Simon Wood, University of Bath, U.K.

in collaboration with

Yannig Goude, EDF

French Grid load



- Half hourly load data from 1st Sept 2002.
- Pierrot and Goude (2011) successfully predicted one day ahead using 48 separate models, one for each half hour period of the day.

Why 48 models?

- The 48 separate models each have a form something like

$$\texttt{MW}_t = f(\texttt{MW}_{t-48}) + \sum_j g_j(x_{jt}) + \epsilon_t$$

  where f and the g_j are a smooth functions to be estimated, and the x_j are other covariates.
- 48 Separate models has some disadvantages
  1. Continuity of effects across the day is not enforced.
  2. Information is wasted as the correlation between neighbouring time points is not utilised.
  3. Fitting 48 models, each to 1/48 of the data promotes estimate instability and creates a huge model checking task each time the model is updated (every day?)
- **But** existing methods for fitting such *smooth additive models* could not cope with fitting one model to all the data.

Smooth additive models

- The basic model is

$$y_i = \sum_j f_j(x_{ji}) + \epsilon_i$$

  where the f_j are smooth functions to be estimated.
- Need to estimate the f_j (including *how* smooth).
- Represent each f_j using a spline basis expansion $f_j(x_j) = \sum_k b_{jk}(x_j)\beta_{jk}$ where b_{jk} are basis functions and β_{jk} are coefficients (maybe 10-100 of each).
- So model becomes

$$y_i = \sum_j \sum_k b_{jk}(x_{ji})\beta_{jk} + \epsilon_i = \mathbf{X}_i \boldsymbol{\beta} + \epsilon_i$$

  ... a linear model. But it is much too flexible.

Penalizing overfit

- To avoid over fitting, we penalize function wiggliness (lack of smoothness) while fitting.
- In particular let

$$J_j(f_j) = \beta^{\mathrm{T}} \mathbf{S} \beta$$

  measure wiggliness of f_j.
- Estimate β to minimize

$$\sum_i (y_i - \mathbf{X}_i \beta)^2 + \sum_j \lambda_j \beta^{\mathrm{T}} \mathbf{S} \beta$$

  where λ_j are *smoothing parameters*.
- High $\lambda_j \Rightarrow$ smooth f_j. Low $\lambda_j \Rightarrow$ wiggly f_j.

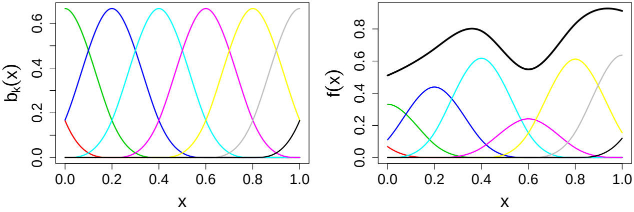
Example basis-penalty: P-splines

- Eilers and Marx have popularized the use of B-spline bases with discrete penalties.
  - If $b_k(x)$ is a B-spline and β_k an unknown coefficient, then

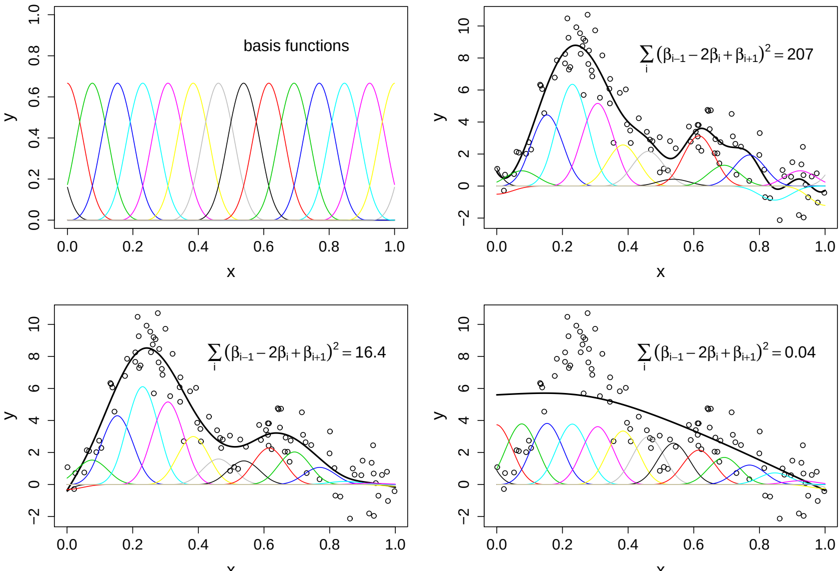
$$f(x) = \sum_k^K \beta_k b_k(x).$$

  - Wiggliness can be penalized by e.g.

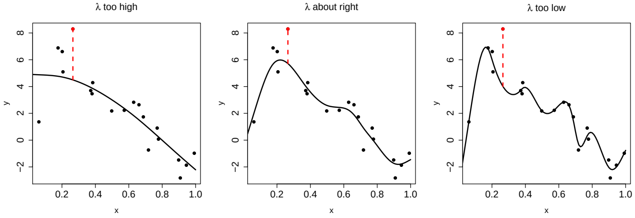
$$\mathcal{P} = \sum_{k=2}^{K-1} (\beta_{j-1} - 2\beta_j + \beta_{j+1})^2 = \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S} \boldsymbol{\beta}.$$



Example varying P-spline penalty



Choosing how much to smooth



- Choose λ_j to optimize model's ability to predict data it wasn't fitted too.
- Generalized cross validation chooses λ_j to minimize

$$\mathcal{V}(\boldsymbol{\lambda}) = \frac{\|\mathbf{y} - \mathbf{X}\beta\|^2}{[n - \mathrm{tr}(\mathbf{F})]^2}$$

where $\mathbf{F} = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \sum_j \lambda_j \mathbf{S}_j)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{X}$. $\mathrm{tr}(\mathbf{F})$ = effective degrees of freedom.

The main problem

- We estimate β by the minimizer of

$$\|\mathbf{y} - \mathbf{X}\beta\|^2 + \sum_j \lambda_j \beta^{\mathrm{T}} \mathbf{S} \beta$$

and the λ_j to minimise

$$\mathcal{V}(\boldsymbol{\lambda}) = \frac{\|\mathbf{y} - \mathbf{X}\beta\|^2}{[n - \mathrm{tr}(\mathbf{F})]^2}$$

- If $\mathbf{X}$ is too large, then we can easily run out of computer memory when trying to fit the model.
- A simple approach will let us fit the model without forming $\mathbf{X}$ whole.

A smaller equivalent fitting problem

- Let $\mathbf{X}$ be $n \times p$
- Consider the QR decomposition $\mathbf{X} = \mathbf{QR}$, where $\mathbf{R}$ is $p \times p$ and $\mathbf{Q}$ has orthogonal columns.
- Let $\mathbf{f} = \mathbf{Q}^{\mathrm{T}}\mathbf{y}$ and $\gamma = \|\mathbf{y}\|^2 - \|\mathbf{f}\|^2$.
- $\|\mathbf{y} - \mathbf{X}\beta\|^2 = \|\mathbf{f} - \mathbf{R}\beta\|^2 + \gamma$ and

$$\mathcal{V}(\boldsymbol{\lambda}) = \frac{\|\mathbf{y} - \mathbf{X}\beta\|^2}{[n - \mathrm{tr}(\mathbf{F})]^2} = \frac{\|\mathbf{f} - \mathbf{R}\beta\|^2 + \gamma}{[n - \mathrm{tr}(\mathbf{F})]^2}$$

  where $\mathbf{F} = (\mathbf{R}^{\mathrm{T}}\mathbf{R} + \sum_j \lambda_j \mathbf{S}_j)^{-1}\mathbf{R}^{\mathrm{T}}\mathbf{R}$.
- So once we have $\mathbf{R}$ and $\mathbf{f}$ we can estimate β and $\boldsymbol{\lambda}$ without $\mathbf{X}$...

QR updating for $\mathbf{R}$ and $\mathbf{f}$

- Let $\mathbf{X} = \begin{bmatrix} \mathbf{X}_0 \\ \mathbf{X}_1 \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \end{bmatrix}$
- Form $\mathbf{X}_0 = \mathbf{Q}_0\mathbf{R}_0$ & $\begin{bmatrix} \mathbf{R}_0 \\ \mathbf{X}_1 \end{bmatrix} = \mathbf{Q}_1\mathbf{R}$.
- Then $\mathbf{X} = \mathbf{Q}\mathbf{R}$ where $\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{Q}_1$.
- Also $\mathbf{f} = \mathbf{Q}^{\mathrm{T}}\mathbf{y} = \mathbf{Q}_1^{\mathrm{T}} \begin{bmatrix} \mathbf{Q}_0^{\mathrm{T}}\mathbf{y}_0 \\ \mathbf{y}_1 \end{bmatrix}$
- Applying these results recursively, $\mathbf{R}$ and $\mathbf{f}$ can be accumulated from sub-blocks of $\mathbf{X}$ without forming $\mathbf{X}$ whole.

$\mathbf{X}^{\mathrm{T}}\mathbf{X}$ updating and Choleski

- $\mathbf{R}$ is the Choleski factor of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$.
- Partitioning $\mathbf{X}$ row-wise into sub-matrices $\mathbf{X}_1, \mathbf{X}_2, \ldots$, we have

$$\mathbf{X}^{\mathrm{T}}\mathbf{X} = \sum_j \mathbf{X}_j^{\mathrm{T}}\mathbf{X}_j$$

  which can be used to accumulate $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ without needing to form $\mathbf{X}$ whole.
- At same time accumulate $\mathbf{X}^{\mathrm{T}}\mathbf{y} = \sum_j \mathbf{X}_j^{\mathrm{T}}\mathbf{y}$.
- Choleski decomposition gives $\mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{X}^{\mathrm{T}}\mathbf{X}$, and $\mathbf{f} = \mathbf{R}^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$.
- This is twice as fast as QR approach, but less numerically stable.

Generalizations

1. Generalized additive models (GAMs) allow distributions other than normal (and non-identity link functions).
2. GAMs can be estimated by iteratively estimating working linear models, using **R** and **f** accumulation tricks on each.
3. REML, AIC etc can also be used for λ_j estimation. REML can be made especially efficient in this context.
4. The accumulation methods are easy to parallelize.
5. Updating a model fit with new data is very easy.
6. AR1 residuals can also be handled easily in the non-generalized case.

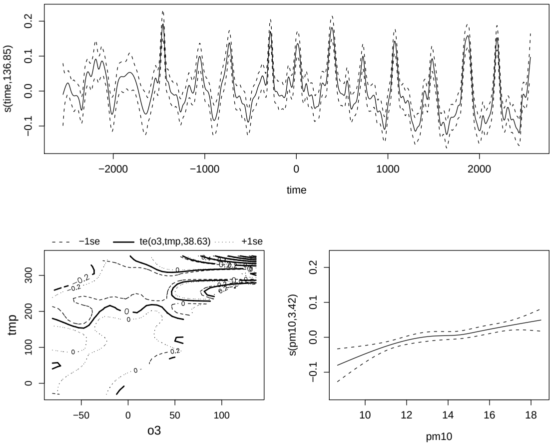
Air pollution example

- Around 5000 daily `death` rates, for Chicago, along with `time`, `ozone`, `pm10`, `tmp` (last 3 averaged over preceding 3 days). Peng and Welty (2004).
- Appropriate GAM is: $\texttt{death}_\texttt{i} \sim \text{Poi}$,

$$\log\{\mathbb{E}(\texttt{death}_i)\} = f_1(\texttt{time}_i) + f_2(\texttt{ozone}_i, \texttt{tmp}_i) + f_3(\texttt{pm10}_i).$$

- f_1 and f_3 penalized cubic regression splines, f_2 tensor product spline.
- Results suggest a *very strong* ozone - temperature interaction.

Air pollution Chicago results



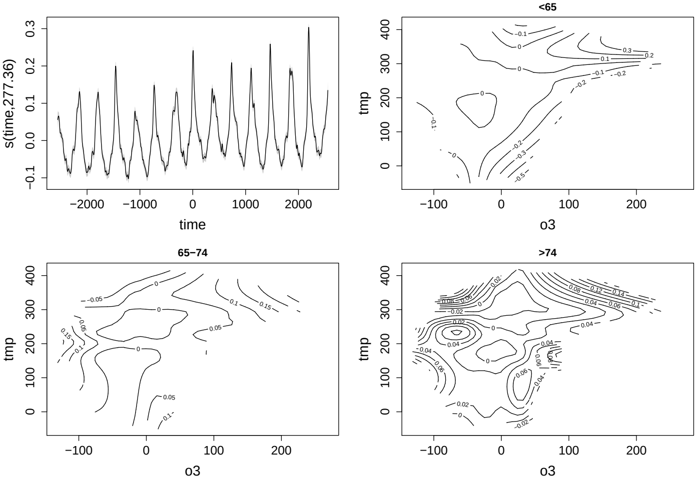
Air pollution Chicago results

- To test the truth of ozone-temp interaction it would be good to fit to the equivalent data for around 100 other cities, simultaneously.
- Model is

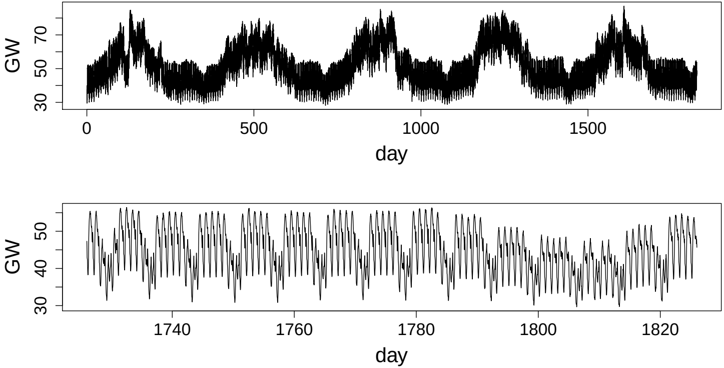
$$\log\{E(\texttt{death}_i)\} = \gamma_j + \alpha_k + f_k(\texttt{o3}_i, \texttt{temp}_i) + f_4(t_i)$$

  if observation i is from city j and age group k (there are 3 age groups recorded).
- Model has 802 coefs and is estimated from 1.2M data.
- Fitting takes 12.5 minutes using 4 cores of a $600 PC.
- Same model to just Chicago, takes 11.5 minutes by previous methods.

Air pollution all cities results



Back to grid load



- Now the 48 separate grid load model can be replaced by

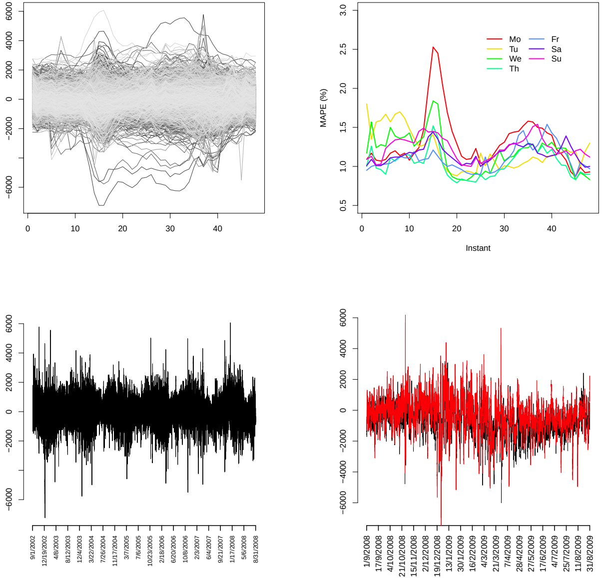
$$\begin{aligned}\mathtt{L}_i = \gamma_j + f_k(\mathtt{I}_i, \mathtt{L}_{i-48}) + g_1(\mathtt{t}_i) + g_2(\mathtt{I}_i, \mathtt{toy}_i) + g_3(\mathtt{T}_i, \mathtt{I}_i) \\ + g_4(\mathtt{T.24}_i, \mathtt{T.48}_i) + g_5(\mathtt{cloud}_i) + \mathtt{ST}_i h(\mathtt{I}_i) + e_i\end{aligned}$$

  if observation i is from day of the week j, and *day class* k.
- $e_i = \rho e_{i-1} + \epsilon_i$ and $\epsilon_i \sim N(0, \sigma^2)$ (AR1).

Fitting grid load model

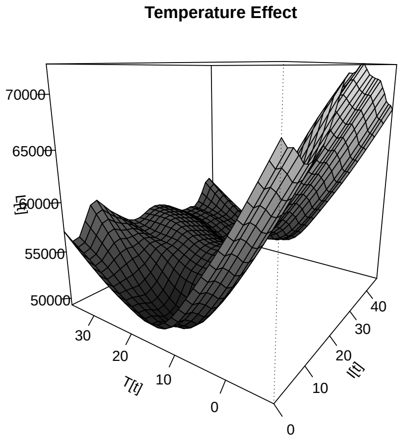
- Using QR update fitting takes under 30 minutes and we estimate $\rho = 0.98$.
- Update with new data then takes under 2 minutes.
- We fit up to 31 August 2008, and then predicted the next years data, one day at a time, with model update.
- Predictive RMSE was 1156MW (1024MW for fit). Predictive MAPE 1.62% (1.46% for fit).
- Setting $\rho = 0$ gives overfit, and worse predictive performance.

Residuals



MAPE (%)
Instant
Mo
Tu
We
Th
Fr
Sa
Su
9/1/2002
12/19/2002
4/8/2003
8/12/2003
12/4/2003
3/22/2004
7/26/2004
11/17/2004
3/7/2005
7/6/2005
10/23/2005
2/18/2006
6/20/2006
10/8/2006
2/3/2007
6/4/2007
9/21/2007
1/17/2008
5/6/2008
8/31/2008
1/9/2008
17/9/2008
4/10/2008
21/10/2008
15/11/2008
2/12/2008
19/12/2008
13/1/2009
30/1/2009
16/2/2009
4/3/2009
21/3/2009
7/4/2009
28/4/2009
27/5/2009
17/6/2009
4/7/2009
25/7/2009
11/8/2009
31/8/2009

Temperature effect



Lagged load Effects