Generalized additive models for large data sets

Author(s): Simon N. Wood, Yannig Goude and Simon Shaw

Source: *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, JANUARY 2015, Vol. 64, No. 1 (JANUARY 2015), pp. 139-155

Published by: Wiley for the Royal Statistical Society

Stable URL: https://www.jstor.org/stable/24771867

Appl. Statist. (2015)
64, *Part* 1, *pp.* 139–155

Generalized additive models for large data sets

Simon N. Wood,

University of Bath, UK

Yannig Goude

Electricité de France, Clamart, France

and Simon Shaw

University of Bath, UK

[Received July 2012. Final revision February 2014]

Summary. We consider an application in electricity grid load prediction, where generalized additive models are appropriate, but where the data set's size can make their use practically intractable with existing methods. We therefore develop practical generalized additive model fitting methods for large data sets in the case in which the smooth terms in the model are represented by using penalized regression splines. The methods use iterative update schemes to obtain factors of the model matrix while requiring only subblocks of the model matrix to be computed at any one time. We show that efficient smoothing parameter estimation can be carried out in a well-justified manner. The grid load prediction problem requires updates of the model fit, as new data become available, and some means for dealing with residual auto-correlation in grid load. Methods are provided for these problems and parallel implementation is covered. The methods allow estimation of generalized additive models for large data sets by using modest computer hardware, and the grid load prediction problem illustrates the utility of reduced rank spline smoothing methods for dealing with complex modelling problems.

Keywords: Correlated additive model; Electricity load prediction; Generalized additive model estimation

1. Introduction

Regression problems containing from tens of thousands to millions of response observations are now commonplace. Sometimes such large data sets require completely new modelling approaches, but sometimes existing model classes are appropriate, provided that they can be made computationally feasible. This paper considers the problem of making generalized additive model (GAM) estimation feasible for large data sets, using modest computer hardware, and in the context in which smoothing parameters must be estimated as part of model fitting.

We were motivated by one application in particular. Fig. 1 shows the gigawatt load on the French national electricity grid at half-hourly intervals, starting on September 1st, 2002. The French energy company Electricité de France (EDF) have had considerable success in using GAMs for short-term load prediction, based on a number of covariates, and especially the load 24 h earlier. However, with existing GAM fitting methods it was not computationally feasible

Address for correspondence: Simon N. Wood, Department of Mathematical Sciences, University of Bath, Claverton Down, Bath, BA2 7AY, UK.
E-mail: s.wood@bath.ac.uk

 0035–9254/15/64139
Published by John Wiley & Sons Ltd on behalf of the Royal Statistical Society.

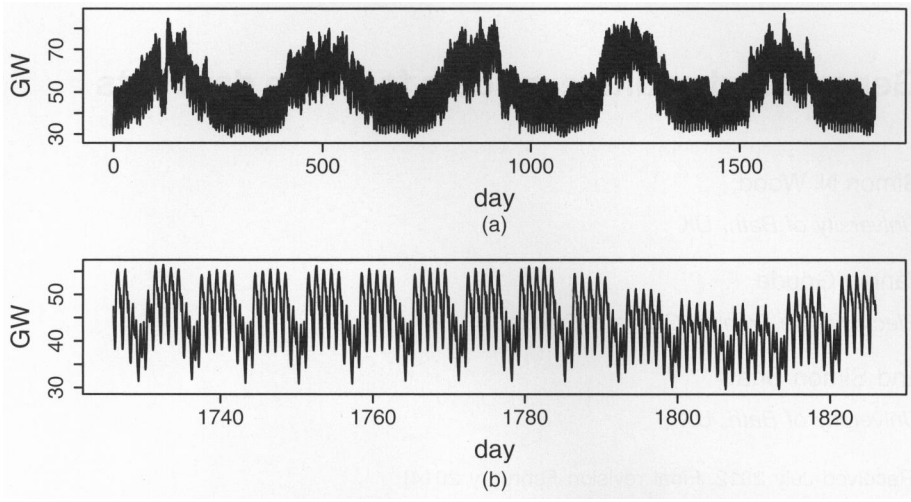

Fig. 1. (a) Load on the French national grid, half hourly in gigawatts against day since September 1st, 2002, and (b) half-hourly gigawatt load for the final 100 days of the data

to fit the whole data set at once, and instead 48 separate models had to be fitted for each half-hour of the day (Pierrot and Goude, 2011). It would clearly be preferable to use a single model.

Although challenging for existing GAM estimation methods, our motivating example is relatively modest by current standards. For example, much statistical modelling effort has been devoted to elucidating the relationships between air pollution and respiratory mortality. One approach is to use GAMs to decompose mortality rates into a background component, varying smoothly in time, and pollutant effects (smooth or linear), while treating the observed death counts as Poisson distributed. Peng and Welty (2004) have assembled daily pollution mortality data, broken into three age groups, for 108 US cities for a period of about 5000 days. An analysis in Wood (2006), section 5.3, for Chicago alone, suggested a very strong ozone temperature interaction effect, which would be important if repeatable. However, the effect is sensitive to rather few days of data, so it should really be tested on the remaining US cities in Peng and Welty's data set. Ideally we would like to fit a GAM to all 1.2 million observations in the data set, simultaneously. Such a fit is well beyond the reach of existing GAM estimation methods but is feasible on modest computing hardware with the methods that are developed below, as the on-line supporting material shows.

Current fitting methods for GAMs and related models (e.g. Wood (2011)) are reasonably efficient and robust when applied to data sets containing up to a few tens of thousands of observations, but they tend to become too memory intensive much beyond this point, so larger data sets, containing hundreds of thousands or millions of data, are out of reach. The difficulty is simply that the model matrix for the model can become too big: if n and p are respectively the number of rows and columns of the model matrix, and M is the number of smoothing parameters, then the memory requirements of GAM fitting methods are typically $O(Mnp^2)$, which can become too large to handle. Here we shall show how simple strategies for updating a model matrix factorization can be used to avoid formation of the whole model matrix in the GAM context. Most importantly, we show how to adapt smoothing parameter estimation methods in this setting.

The general class of models that we consider can be written as

$$g\{E(y_i)\} = \mathbf{A}_i\boldsymbol{\theta} + \sum_j L_{ij} f_j \tag{1}$$

where y_i is one of n observations of a univariate response variable from an exponential family distribution (or at least with mean–variance relationship known to within a scale parameter), g is a known smooth monotonic link function, $\mathbf{A}$ is an n-row model matrix, $\boldsymbol{\theta}$ is a vector of unknown parameters, L_{ij} is a known linear functional and f_j an unknown smooth function of one or more variables, with an unknown degree of smoothness. Associated with each f_j is some measure of departure from smoothness $J_j(f)$.

The most common example of equation (1) is a GAM (Hastie and Tibshirani, 1986, 1990) which occurs when the L_{ij} are evaluation functionals, but other examples are varying-coefficient models, functional generalized linear models (e.g. Marx and Eilers (1999)) and structured additive regression models (e.g. Fahrmeir *et al.* (2004)). There are a variety of methods for estimating particular members of the model class. We shall focus on the case in which the f_j are represented by intermediate rank penalized regression splines (e.g. Parker and Rice (1985) and Eilers and Marx (1996)). In this case quite efficient computational methods can be obtained for the whole model class (e.g. Wood (2000)), with fitting performed by penalized iteratively reweighted least squares, and smoothness selection by generalized cross-validation (GCV), restricted maximum likelihood (REML) or similar (see Wood (2011)).

2. Gaussian identity link case

First consider the case in which the y_i are independently normally distributed with variance ϕ, and g is the identity function. The f_j are each represented by using a linear basis expansion (e.g. a B-spline or thin plate regression spline basis), and J_j is chosen to be quadratic in the basis coefficients. In this case the model for the expected response can be rewritten as

$$E(\mathbf{y}) = \mathbf{X}\boldsymbol{\beta} \tag{2}$$

where $n \times p$ model matrix $\mathbf{X}$ contains $\mathbf{A}$ and the evaluated basis functions, and $\boldsymbol{\beta}$ contains $\boldsymbol{\theta}$ and all the basis coefficients. We assume $p < n$, since the methods that are presented here are only practically interesting in this case, and we estimate $\boldsymbol{\beta}$ by minimization of

$$\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_j \lambda_j \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta} \tag{3}$$

where $\mathbf{S}_j$ is a matrix of known coefficients, such that $J_j(f_j) = \boldsymbol{\beta}^{\mathrm{T}}\mathbf{S}_j\boldsymbol{\beta}$ ($\mathbf{S}_j$ is $p \times p$, but its non-zero block is usually smaller than this), and λ_j is a smoothing parameter controlling the fit–smoothness trade-off for f_j. (The notation is slightly sloppy here, as there may be several smoothing parameters associated with one f_j.) Given $\boldsymbol{\lambda}$, expression (3) may readily be minimized to give the coefficient estimates $\hat{\boldsymbol{\beta}}_\lambda$.

The estimation of $\boldsymbol{\lambda}$ is more awkward. One approach, based on trying to minimize prediction error, is GCV, which seeks to minimize

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

with respect to the smoothing parameters, where $\mathrm{tr}(\mathbf{F}_\lambda)$ is the effective degrees of freedom of the model, $\mathbf{F}_\lambda = (\mathbf{X}^{\mathrm{T}}\mathbf{X} + \mathbf{S}_\lambda)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{X}$ and $\mathbf{S}_\lambda = \Sigma_j \lambda_j \mathbf{S}_j$. Other possibilities are REML, $\mathcal{V}_r$, or Mallows's C_p which are covered in Appendix A. A Newton method is usually used to optimize $\mathcal{V}_g$ with respect to $\log(\boldsymbol{\lambda})$, with the $\hat{\boldsymbol{\beta}}_\lambda$ being obtained by direct minimization of expression (3) for each trial $\boldsymbol{\lambda}$: Wood (2004, 2011) provide details of computationally stable numerical methods.

Now suppose that the model matrix is first QR decomposed into a column orthogonal $n \times p$ factor $\mathbf{Q}$ and an upper triangular $p \times p$ factor $\mathbf{R}$ so that $\mathbf{X} = \mathbf{QR}$. If we also form $\mathbf{f} = \mathbf{Q}^{\mathrm{T}}\mathbf{y}$ and $\|\mathbf{r}\|^2 = \|\mathbf{y}\|^2 - \|\mathbf{f}\|^2$ then expression (3) becomes

$$\|\mathbf{f} - \mathbf{R}\beta\|^2 + \|\mathbf{r}\|^2 + \sum_j \lambda_j \beta^{\mathrm{T}} \mathbf{S}_j \beta \tag{4}$$

and fairly routine calculation shows that

$$\mathcal{V}_g(\boldsymbol{\lambda}) = \frac{n\|\mathbf{f} - \mathbf{R}\hat{\beta}_\lambda\|^2 + \|\mathbf{r}\|^2}{\{n - \mathrm{tr}(\mathbf{F}_\lambda)\}^2}$$

and $\mathbf{F}_\lambda = (\mathbf{R}^{\mathrm{T}}\mathbf{R} + \mathbf{S}_\lambda)^{-1}\mathbf{R}^{\mathrm{T}}\mathbf{R}$.

The point here is that once we have $\mathbf{R}$, $\mathbf{f}$ and $\|\mathbf{r}\|^2$ then we have everything that is needed for fitting, and $\mathbf{X}$ plays no further part. Hence, if we can obtain these quantities without forming $\mathbf{X}$ as a whole, then we can estimate the models without incurring high computer memory costs. Appendix A shows that the same is true when using Mallows's C_p and REML, and also discusses some potential alternative approaches.

In fact $\mathbf{R}$, $\mathbf{f}$ and $\|\mathbf{r}\|^2$ can be computed in a way that requires only small subblocks of $\mathbf{X}$ to be formed at any one time, using methods based on iterative updating of a QR-decomposition, or less stably using a Choleski decomposition method. Appendix B provides the full details and also shows that the approach leads naturally to an efficient on-line updating method for large additive models.

2.1. Correlated errors

The short-term load prediction problem that is discussed in Section 4 also requires that we model residual auto-correlation, and a simple AR(p) correlation structure is quite easy to deal with. First modify the Gaussian identity link model (2) to $\mathbf{y} = \mathbf{X}\beta + \mathbf{e}$, where the covariance matrix of $\mathbf{e}$ is $\phi\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}$ is an auto-regressive AR(p) correlation matrix. Then the Choleski factor $\mathbf{C}$ of $\boldsymbol{\Sigma}^{-1}$ is banded and $\boldsymbol{\varepsilon} = \mathbf{Ce}$ are independent and identically distributed $N(0, \phi)$. In consequence if $\tilde{\mathbf{y}} = \mathbf{Cy}$ and $\tilde{\mathbf{X}} = \mathbf{CX}$, we have

$$\tilde{\mathbf{y}} = \tilde{\mathbf{X}}\beta + \boldsymbol{\varepsilon}, \tag{5}$$

which is in the form (2), so the methods of the previous sections can be used to estimate β. The only modification is that, if REML is used to estimate ρ itself, then the log-REML score must be corrected for the transformation by $\mathbf{C}$, but given that $\mathbf{C}$ is triangular the required log-determinant is easily obtained. Computationally, a simple one-dimensional search can be used for ρ, with each ρ-value requiring the model to be refitted. Note that the banded structure of $\mathbf{C}$ and the fact that it can be obtained without forming $\boldsymbol{\Sigma}^{-1}$ are what make the AR(p) model computationally feasible: the formation of $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{y}}$ involves a computationally cheap weighted differencing of adjacent rows of $\mathbf{X}$ and $\mathbf{y}$, rather than an expensive full matrix multiplication.

3. Generalized additive model fitting

In the generalized case the unknown functions and their penalties are represented exactly as in the simple Gaussian identity link case. All that changes is that the model becomes an over-parameterized generalized linear model,

$$g\{E(y_i)\} = \mathbf{X}_i\beta,$$

to be estimated by penalized likelihood maximization, in place of penalized least squares (see for example Green and Silverman (1994)). The algorithm that is used to maximize the penalized likelihood is penalized iteratively reweighted least squares (PIRLS) which proceeds as follows, where V is the function such that $\text{var}(y_i) = \phi V(\mu_i)$ and $\mu_i = E(y_i)$.

First initialize $\hat{\mu}_i = y_i + \xi_i$ and $\hat{\eta}_i = g(\hat{\mu}_i)$ where ξ_i is a small quantity (often 0) added to ensure that $g(\hat{\mu}_i)$ exists. Then iterate the following steps to convergence.

Step 1: form $z_i = g'(\hat{\mu}_i)(y_i - \hat{\mu}_i) + \hat{\eta}_i$ and $w_i = V(\hat{\mu}_i)^{-1/2} g'(\hat{\mu}_i)^{-1}$.

Step 2: putting the w_i in a diagonal matrix $\mathbf{W}$, minimize the weighted version of expression (3),

$$\|\mathbf{W}\mathbf{z} - \mathbf{W}\mathbf{X}\boldsymbol{\beta}\|^2 + \sum_j \lambda_j \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta},$$

with respect to $\boldsymbol{\beta}$ to obtain $\hat{\boldsymbol{\beta}}$ and the updates $\hat{\boldsymbol{\eta}} = \mathbf{X}\hat{\boldsymbol{\beta}}$, and $\hat{\mu}_i = g^{-1}(\hat{\eta}_i)$.

For moderate-sized data sets it is most reliable to iterate the PIRLS algorithm to convergence for each trial $\boldsymbol{\lambda}$, and to estimate $\boldsymbol{\lambda}$ by using generalized versions of GCV, C_p or a Laplace approximate REML (see Wood (2008, 2011)). In the large data set case this approach carries the disadvantage of requiring several times the storage of $\mathbf{X}$ to calculate derivatives of the smoothness selection criterion efficiently. To avoid such high storage cost we can instead return to an earlier approach, originally due to Gu (1992). This simply uses GCV, C_p or REML to select the smoothing parameters of the working linear model, at each step of the PIRLS algorithm. Gu (1992, 2002) termed this 'performance-oriented iteration', and it is quite similar to Breslow and Clayton's (1993) penalized quasi-likelihood. The method usually converges and, although convergence is not guaranteed, the kind of ill conditioning that promotes convergence problems tends to decrease with increasing n.

3.1. Performance-oriented iteration for large data sets

Performance-oriented iteration for large data sets can be implemented by using the QR-update approach of Appendix B on the matrix $\mathbf{WX}$, at each step of the PIRLS algorithm. This means that the calculations that are required to form the (submatrices of the) model matrix must be repeated at each step, but these computations are $O(np)$ rather than the $O(np^2)$ of the QR-decomposition, so for many smoothing bases the operations cost is not important. Formally, the algorithm is as follows.

3.1.1. Initialization

Let $\mathbf{x}_i$ denote the vector of covariates that are associated with response variable y_i where $i = 1, \ldots, n$, and divide the integers from 1 to n into M non-overlapping subsets $\gamma_1, \ldots, \gamma_M$ of approximately equal size (so $\cup_i \gamma_i = \{1, \ldots, n\}$ and $\gamma_j \cap \gamma_i = \emptyset$ for all $i \neq j$). M is chosen to avoid running out of computer memory. Let $\bar{\eta}_i = g(y_i + \xi_i)$ (with ξ_i as defined in the previous section). Set the PIRLS iteration index $q = 0$ and $D = 0$ (or any constant, in fact). Perform any initialization steps that are necessary to set up the bases for the smooth terms.

3.1.2. Iteration

Step 1: set $D_{\text{old}} = D$, $\mathbf{R}$ to be a $0 \times p$ matrix, $\mathbf{f}$ a 0-vector, $D = 0$ and $r = 0$.

Step 2: repeat the following steps (a)–(f) for $k = 1, \ldots, M$.

(a) Set $\mathbf{f}_0 = \mathbf{f}$ and $\mathbf{R}_0 = \mathbf{R}$.

(b) Form the model submatrix $\mathbf{X}_k$ for the covariate set $\{\mathbf{x}_i : i \in \gamma_k\}$.

(c) If $q>0$ form $\hat{\boldsymbol{\eta}}=\mathbf{X}_k\hat{\boldsymbol{\beta}}_\lambda$; otherwise $\hat{\boldsymbol{\eta}}=\bar{\boldsymbol{\eta}}_{\gamma_k}$.
(d) Form $\hat{\mu}_i=g^{-1}(\hat{\eta}_i)$, $z_i=g'(\hat{\mu}_i)(y_i-\hat{\mu}_i)+\hat{\eta}_i$ and $w_i=V(\hat{\mu}_i)^{-1/2}g'(\hat{\mu}_i)^{-1}\ \forall i\in\gamma_k$. Let $\mathbf{z}$ be the vector containing these z_i-values and $\mathbf{W}$ be the diagonal matrix of corresponding w_i-values.
(e) Set $r\leftarrow r+\|\mathbf{Wz}\|^2$, calculate the deviance residuals for the current subset of data and add the sum of squares of these to D.
(f) Form

$$\mathbf{QR}=\begin{pmatrix}\mathbf{R}_0\\ \mathbf{WX}_k\end{pmatrix}$$

and

$$\mathbf{f}=\mathbf{Q}^{\mathrm{T}}\begin{pmatrix}\mathbf{f}_0\\ \mathbf{Wz}\end{pmatrix}$$

and discard $\mathbf{Q}$.

Step 3: set $\|\mathbf{r}\|^2=r-\|\mathbf{f}\|^2$.
Step 4: if $q>0$ test for convergence by comparing the current deviance D with the previous deviance D_{old}. Stop if convergence has been reached (or q has exceeded some predetermined limit suggesting failure).
Step 5: estimate $\boldsymbol{\lambda}$ by optimizing $\mathcal{V}_{r/g}$ or C_p exactly as in Section 2. This also yields $\hat{\boldsymbol{\beta}}_\lambda$.
Step 6: $q\leftarrow q+1$.

At convergence the final $\hat{\boldsymbol{\beta}}_\lambda$ and $\boldsymbol{\lambda}$ are the coefficient and smoothing parameter estimates. Further inference is most usefully based on the Bayesian approximation

$$\boldsymbol{\beta}\sim N\{\hat{\boldsymbol{\beta}}_\lambda,(\mathbf{R}^{\mathrm{T}}\mathbf{R}+\mathbf{S}_\lambda)^{-1}\phi\},$$

where any ϕ-estimate required is obtained as part of REML optimization or as

$$\hat{\phi}=\frac{\|\mathbf{f}-\mathbf{R}\hat{\boldsymbol{\beta}}_\lambda\|^2+\|\mathbf{r}\|^2}{n-\mathrm{tr}(\mathbf{F}_\lambda)}$$

(see for example Wood (2006) for further details).

Note that step 2 can be parallelized as described in Appendix B. The γ_i are grouped into equally sized non-overlapping sets to be allocated to different processors. Step 2 is run on each processor, with its set of γ_i, resulting in an $\mathbf{R}_i$ and $\mathbf{f}_i$ from each processor. Application of expression (7) from Appendix B then yields the required $\mathbf{R}$ and $\mathbf{f}$. Furthermore at step 5 the operations count for the smoothing parameter optimization method of Wood (2011) can drop substantially as a result of $\mathcal{V}$ being unweighted in the smoothing parameter optimization.

A Choleski-based alternative simply replaces three steps of the iteration as follows.

Step 1: set $D_{\mathrm{old}}=D$, $\mathbf{R}$ to be a $p\times p$ matrix of 0s, $\mathbf{f}$ a p-vector of 0s, $D=0$ and $r=0$.
Step 2(f): set $\mathbf{R}=\mathbf{R}_0+\mathbf{X}_k^{\mathrm{T}}\mathbf{WX}_k$ and $\mathbf{f}=\mathbf{f}_0+\mathbf{X}_k^{\mathrm{T}}\mathbf{Wz}$.
Step 3: replace $\mathbf{R}$ (which really contains $\mathbf{X}^{\mathrm{T}}\mathbf{WX}$) by its Choleski decomposition, and then replace $\mathbf{f}$ by $\mathbf{R}^{-1}\mathbf{f}$. Set $\|\mathbf{r}\|^2=r-\|\mathbf{f}\|^2$.

Again, step 2 is easily parallelized in this case.

3.1.3. *Subsampling for starting values*

It is computationally wasteful to run the early steps of the PIRLS algorithm using the full data set, since this amounts to wasting effort exactly fitting a working model that is known to

be wrong. For this reason it is often sensible first to estimate the model on a 5–10% random subsample of the data, and then to use the resulting β and λ-estimates as starting values for fitting the full data. In practice this trick usually saves one or two steps of the PIRLS algorithm when fitting the full data.

3.2. *Justifying the smoothness selection step*

No special justification is required to apply GCV or C_p to the working model, at each step of the PIRLS iteration: the assumptions that are required for these criteria hold for the working model.

REML (or ML) is less straightforward as the working data z_i may be far from the normality that is required to derive $\mathcal{V}_r$. However, for large data sets with $n \gg p$ the central limit theorem implies that $\mathbf{f} = \mathbf{Q}^{\mathrm{T}}\mathbf{z}$ (where $\mathbf{z}$ now refers to the whole working data n-vector) will tend to an $N(\mathbf{R}\beta, \mathbf{I}\phi)$ distribution. The REML score based on the density of $\mathbf{f}$ is then

$$\mathcal{V}_r^*(\boldsymbol{\lambda}) = \frac{\|\mathbf{f} - \mathbf{R}\hat{\beta}_\lambda\|^2 + \hat{\beta}_\lambda^{\mathrm{T}}\mathbf{S}_\lambda\hat{\beta}_\lambda}{2\phi} + \frac{p - M_p}{2}\log(2\pi\phi) + \frac{\log|\mathbf{R}^{\mathrm{T}}\mathbf{R} + \mathbf{S}_\lambda| - \log|\mathbf{S}_\lambda|_+}{2}$$

where $|\mathbf{S}_\lambda|_+$ is the product of the positive eigenvalues of $\mathbf{S}_\lambda$ which has M_p zero eigenvalues. So $\mathcal{V}_r^*$ has exactly the form of $\mathcal{V}_r$ from Appendix A, but with $\|\mathbf{r}\|^2 \equiv 0$ and n set to p. For any fixed ϕ, $\mathcal{V}_r$ and $\mathcal{V}_r^*$ are obviously minimized by the same $\boldsymbol{\lambda}$. However, if ϕ is unknown then it must be estimated somehow. Optimizing $\mathcal{V}_r^*$ with respect to ϕ is clearly not a good option, since the absent term $\|\mathbf{r}\|^2$ carries information about ϕ. However,

$$\hat{\phi} = \frac{\|\mathbf{f} - \mathbf{R}\hat{\beta}_\lambda\|^2 + \|\mathbf{r}\|^2 + \hat{\beta}_\lambda^{\mathrm{T}}\mathbf{S}_\lambda\hat{\beta}_\lambda}{n - M_p}$$

is an estimator of ϕ that can be motivated either by analogy with REML estimation of ϕ or as a simple moment estimator (in general the numerator of the estimator is the Pearson statistic plus the smoothing penalty). Now, since $\hat{\phi}$ is readily seen to be the minimizer of $\mathcal{V}_r$, whereas $\mathcal{V}_r$ and $\mathcal{V}_r^*$ are minimized by the same $\boldsymbol{\lambda}$, then minimizing $\mathcal{V}_r^*$ with respect to $\boldsymbol{\lambda}$ while using $\hat{\phi}$ as the estimator of ϕ is the same as finding $\hat{\boldsymbol{\lambda}}$ and $\hat{\phi}$ by minimizing $\mathcal{V}_r$ (with $\mathbf{R}$, $\mathbf{f}$ and $\|\mathbf{r}\|^2$ computed by PIRLS as in Section 3.1).

Recent work by Reiss and Ogden (2009) suggests that $\mathcal{V}_r$ is less prone to multiple local minima than $\mathcal{V}_g$ (or presumably the closely related C_p). In a performance-oriented iteration context this suggests that use of REML is likely to promote iteration convergence, since there is less scope for the iteration to cycle between multiple optima.

3.3. *The $p < n$ assumption*

When using GAM models for such large data sets, computational feasibility rests on the ability to use reduced rank smooths so that p is substantially less than n. But one obvious question is whether it is reasonable to suppose that p will grow much more slowly than n, as the approach given here does, implicitly. Clearly the assumption is not reasonable if smooths are being used as random fields to mop up residual correlation, nor for a smooth of time when additional data extend the time axis. Otherwise all the theoretical evidence is that smoothing bases need to grow only rather slowly with sample size: as an example consider the case of a cubic regression spline with evenly spaced knots. It is known that the average squared bias for a cubic spline is $O(h^8) = O(k^{-8})$ where h is the knot spacing and k the number of knots (e.g. de Boor (1978)). From basic regression theory the average variance of the spline is $O(k/n)$ where n is the number of data.

To avoid either bias or variance dominating the other, and giving suboptimal mean-square error as $n \to \infty$, we should choose k to equate the orders of the squared bias and variance, i.e. $k \propto n^{1/9}$. This rate can also be used under penalization, since at any finite sample size we choose the degree of penalization to decrease the mean-square error relatively to pure regression, although this argument itself does not say that an alternative rate might not be optimal under penalization. Indeed, under penalization Gu and Kim (2002) suggested that the basis dimension should scale as $n^{2/9}$, i.e. 1 million data should require only around five times the number of coefficients that 1000 required.

4. Short-term grid load prediction

As discussed in Section 1, Fig. 1 shows the load in gigawatts on the French national grid. EDF have built successful 1-day-ahead grid load prediction models based on splitting the data up, by half-hour of the day and fitting GAMs to each of the 48 resulting subsets. Although it makes estimation feasible with existing methods, using 48 separate models suffers from three practical disadvantages.

(a) It fails to use information efficiently, since the correlation between data in neighbouring half-hour periods is not exploited.
(b) It suffers from interpretational difficulties, since model continuity between half-hour periods is not enforced, which is somewhat unphysical.
(c) Operational forecast models *must* be statistically stable, while at the same time the predictive purpose of the model suggests that smoothness estimation for such models should use a prediction error criterion such as GCV. The difficulty then arises that GCV and related criteria are known to produce a small proportion of substantial overfits, with increasing problems as the sample size decreases (see for example Reiss and Ogden (2009)). Fitting separate models to 48 subsets of the data exacerbates exposure to this problem, thereby detracting from model stability, and burdening the operational forecaster with a very substantial model checking task, each time that model estimates are updated. Fitting one model to all the data substantially reduces the scope for overfitting, while reducing the associated model checking task to manageable proportions.

The primary motivation for the development of the methods that are presented in this paper was to allow a single model to be used in place of the 48 separate models, and one version of such a model is presented here.

The Fig. 1 data are from a data set that was assembled by EDF, which also includes meteorological data (1-day-ahead forecasts of temperature in degrees Celsius and cloud cover in eighths of sky covered, from MeteoFrance), calendar and tariff information. Although the determinants of load are relatively complex, so forecasting must be done statistically, EDF adopts the sensible precaution of insisting that forecasting models must have interpretable effects that make physical sense. In part this is necessary to help the process of operationally forecasting exceptional events, outside the conditions that normally apply in the model fitting data set.

Previous forecasting experience and exploratory data analysis suggest that good predictive performance can be achieved on the basis of grid load 24 h earlier, forecast temperature, actual temperature 24 and 48 h earlier, forecast cloud cover and the time of year. Broadly speaking, load is lowest around 15 °C, with an increase below that as heating demand increases with decreasing temperature, and a less pronounced increase above 15 °C that is attributable to air-conditioning. The temperature that is used for forecasting is an average over France, weighted by the use of electricity. Comparison of days that are similar except for cloud cover suggests an

important additional direct effect of cloud cover, which is probably attributable to the effects of passive solar heating of buildings and lighting effects.

A further important effect is that EDF manages expected days of exceptionally high demand in winter via demand reducing financial incentives to large users. 1 day in advance, EDF Services produce a forecast of the expected demand reduction at 7 p.m. for the 'Special tariff days' on which such incentives apply, and this forecast is also included as a predictor. Based on these considerations, exploratory analysis, experience with separate models for each half-hour of the day and after some model selection, the following model is proposed:

$$\begin{aligned} L_i = {} & \gamma_{j(i)} + f_{k(i)}(I_i, L_{i-48}) + g_1(t_i) + g_2(I_i, \text{toy}_i) + g_3(T_i, I_i) + g_4(\text{T.24}_i, \text{T.48}_i) \\ & + g_5(\text{cloud}_i) + \text{ST}_i h(I_i) + e_i \end{aligned} \tag{6}$$

if observation i is from day of the week j, and *day class* k. The notation $j(i)$ indicates that index j is itself a function of index i. The error term is modelled as AR(1): $e_i = \rho e_{i-1} + \varepsilon_i$ and $\varepsilon_i \sim N(0, \sigma^2)$. Here L_i is the grid load in megawatts at the ith half-hour period; I_i is the half-hour period of the day (an integer from 1 to 48; henceforth the 'instant'); t is time since September 1st, 2002; toy is the time of year in hours running from 0 to 8760; cloud is an index of cloud cover over France; T is temperature with T.24 and T.48 temperature measurements lagged by 24 and 48 h respectively. Day class is one of hh, hw, ww or hw, depending on whether the day in question is a holiday following a holiday, a workday following a holiday, and so on. The idea is that the way in which one day's load depends on the previous day's load is strongly dependent on whether the days concerned are workdays or holidays. Weekends are holidays in this model. ST is the predicted special tariff load reduction, which is 0 on normal days, and a single number for the whole of each special tariff day. The g_k and f_j are all smooth functions represented as penalized regression splines. The four f_k are each rank 150 tensor products of cubic regression splines (cyclic in I_i); g_2 and g_3 are similar tensor products, but each of rank 120; g_4 is a rank 45 tensor product spline; g_1 and g_5 are cubic splines and h is an unpenalized function equivalent to a 48-level factor variable for half-hour period of the day.

The individual terms in model (6) are largely based on effects that are expected on the basis of EDF's experience and in some cases, such as the temperature effects, on *a priori* grounds; however, there are some substantial structural assumptions that are not obvious *a priori*. One consideration is whether the effects would better be treated as additive or multiplicative, which can be addressed by estimating the model with and without a log-link. When this was done, assuming that $\rho = 0$, the estimated r^2 was the same to three significant figures for both models, with no detectable difference in predictive performance. We therefore decided to use the additive version, which makes the handling of correlation easier than it would otherwise be. Another issue is the way in which seasonality is handled. We used cyclic effects in the time of year for this, but another elegant approach to seasonality is the varying-coefficient approach of Eilers *et al.* (2008), in which seasonality is handled via a truncated Fourier series of the form

$$\sum_{k=1}^{K} f_{2k-1}(t, I)\sin(2k\pi/T) + f_{2k}(t, I)\cos(2k\pi/T)$$

where the $f_j(t, I)$ terms are smoothly varying functions of time and instant of day, which control the phase and amplitude of a period T cycle. We experimented with models in which seasonality was handled by such terms, but for these data we could produce only slightly worse performance than model (6) in terms of R^2 and prediction error performance.

As an illustration of model efficacy, the model was estimated by using data until August 31st,

2008, and then used to make 1-day-ahead predictions for the following year, using the on-line updating approach of Appendix B to update the model estimates with each new day's data. In operational forecasting bank holidays are handled *ad hoc* and are generally excluded from routine automatic forecasting, so we have also done this here. Given the predictive nature of the modelling, GCV was used for smoothing parameter and AR parameter estimation. Note that when performing the initial model fit it is necessary to set up the bases in a way that allows $g_1(t)$ to have a domain running up to September 2009, so that the basis continues to be appropriate until the end of the prediction period. This is unproblematic to set up, but it could cause numerical problems if we were to use the Choleski-based updating scheme as the initial $\mathbf{X}^T\mathbf{X}$ could then be rank deficient, or nearly so.

Fig. 2 shows residual plots for the initial fitted data, indicating that the major errors are in the prediction of Monday mornings, that big outliers are largely in the daytime and that residuals tend to be larger in winter, when the load is higher. Fig. 2(d) shows 1-day-ahead prediction residuals for the final year of data. The residual plots for prediction appear similar to those from the initial fitting. The AR parameter is estimated as 0.98, and Fig. 2 also shows prediction residuals for a simplified model with the parameter set to 0. The slightly worse appearance of the residuals when residual correlation is ignored is confirmed by the mean absolute percentage error MAPE and root-mean-square error RMSE figures for the models (Table 1). Fig. 3(a) directly shows the fitted load from the model overlaid on the observed load for the period shown in Fig. 1(b).

Prediction error is lower for the AR model, and the mismatch between the fitted set and prediction set is much lower as well. The larger mismatch for $\rho = 0$ almost certainly results from overfitting when correlation is ignored: the effective degrees of freedom of the model with correlation is 83% of the equivalent for the model without correlation. This emphasizes the practical importance of the method that was developed in Section 2.1. Figs 3(b) and 3(c) show the auto-correlation functions for the model residuals with $\rho = 0$ and for the standardized residuals when $\rho = 0.98$. Clearly the AR(1) model leads to substantial improvement, but there is some room for further refinement. The performance of model (6) is competitive with that of Pierrot and Goude (2011) but has the three practical advantages of single-model fitting that were listed above, while also easing the process of updating model estimates as new data become available. To appreciate the latter advantage, we compared computation times by using a machine with a 3.1-GHz Intel i3 540 processor and 3.7 Gbytes of random-access memory, running LINUX (i.e. a personal computer retailing at less than US $600). The 48 separate models equivalent to model (6), without AR residuals, take around 1 h to estimate by using `gam` from package `mgcv`, whereas using this paper's methods initial estimation of model (6) takes under half an hour, including searching for ρ. A subsequent update, using Appendix B, then takes just under 2 min, whereas previously a complete refit of all 48 models would have been required.

An obvious question is whether a daily update is necessary, when the model has already been fitted to such a lengthy data set. The reason for doing so is that there are many combinations of conditions that may not be well sampled in even quite a long run of data, and the predictor variables are themselves highly correlated. This means that, if conditions are unusual over a period of a few days, the last few days of information may constitute a non-negligible proportion of the information about load under these conditions and may therefore have a non-negligible influence on the estimates of the smooth functions in the vicinity of these unusual covariates. For this reason it is usually undesirable to exclude the most recent data in model fitting for prediction.

Although we developed the methods that are described here in direct response to difficulties in using existing GAM estimation methods for this problem, model (6) can be estimated by the method of Wood (2011) given about 8 Gbytes of memory. However, doing so is less than

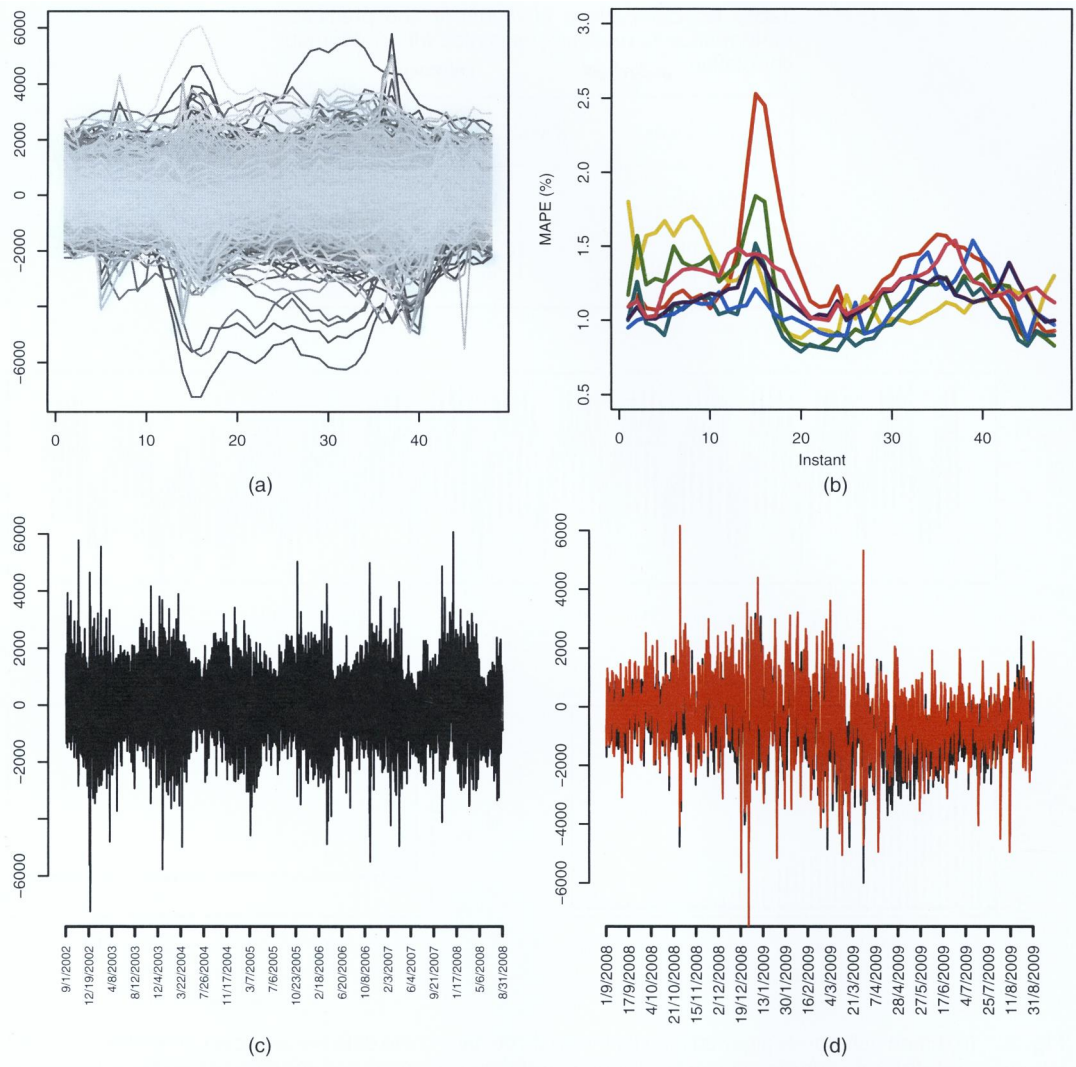

Fig. 2. (a) Daily residuals against half-hour 'instant' of the day, (b) residual mean absolute percentage error MAPE against instant (——, Monday; ——, Tuesday; ——, Wednesday; ——, Thursday; ——, Friday; ——, Saturday; ——, Sunday), (c) residuals against time and (d) predictive residuals against time (——, model ignoring residual auto-correction; ——, full model with AR(1) residuals)

a tenth of the speed of the method that is proposed here and an update is only possible by full refit. For some of the more complex models that were considered during the model development, using larger bases and more complicated classifications of day types, substantially more memory would be required, and the future objective of modelling some effects by region is completely out of reach without the new methods. The on-line supporting material analyses the air pollution data that were discussed in Section 1 and provides an example that is substantially beyond standard personal computing hardware without the methods that are considered here.

To illustrate interpretability, the estimates of the f_k are shown in Fig. 4. Note the relatively linear relationship of one workday's load to the previous workday's load at all times of day, with some regression to the mean evident. Holidays show a different pattern, with low power con-

Table 1. Comparison of model fit and predictive performance for model (6) estimated with and without correlation

Model variant	*RMSE (Mw)*	*MAPE (%)*
$\rho = 0$ fit data	831	1.17
$\rho = 0$ prediction	1220	1.87
$\rho = 0.98$ fit	1024	1.46
$\rho = 0.98$ prediction	1156	1.62

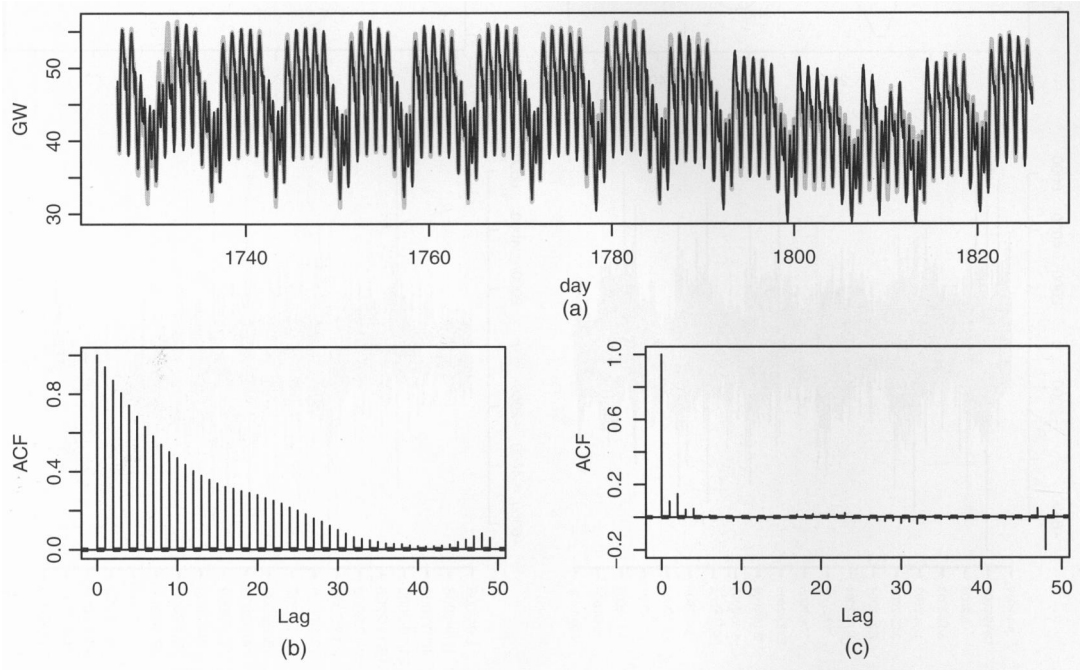

Fig. 3. (a) Fitted half-hourly gigawatt load for the final 100 days of the data (——) overlaid on the observed load (——), (b) auto-correlation function when correlation is ignored and (c) auto-correlation function for standardized residuals from the model with AR(0.98) errors

sumption on the preceding day not being very predictive, but a linear relationship evident above 50 Gw. Workdays following holidays, which are dominated by Mondays, show the opposite shape to that of the holidays, with the strongest effect evident at low loads.

In summary, the approach that is developed in this paper has allowed us to improve the stability and interpretability of the EDF forecasting model, by fitting all available data simultaneously in a manner that allows modelling of auto-correlation and efficient estimation updates as new data become available.

5. Conclusions

Motivated by the practical need to improve the speed and stability of forecasting models that are used by EDF, we have shown how GAMs can be applied to much larger data sets than

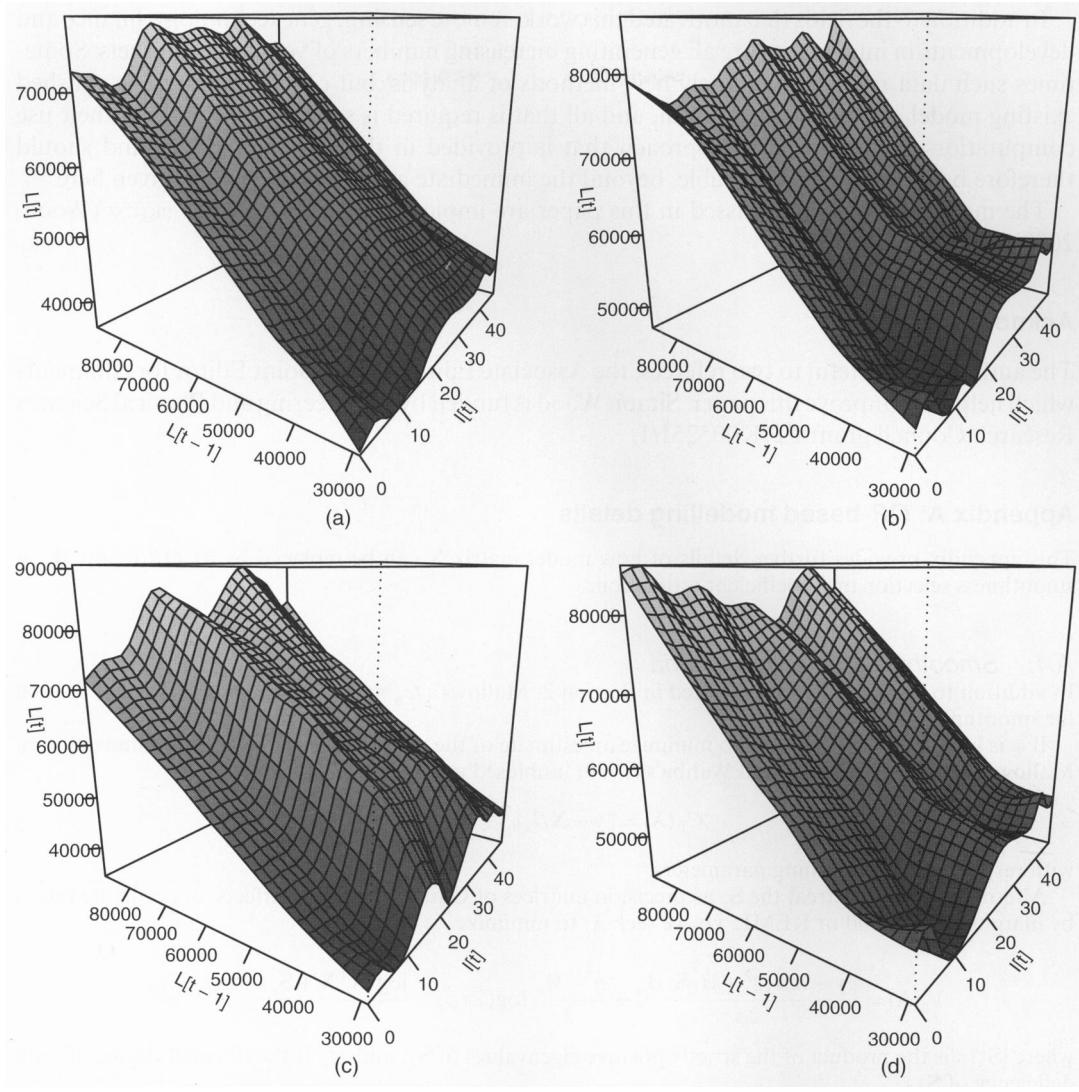

Fig. 4. Lagged load effect against time of day $l(t)$, for (a) a workday–workday, (b) a workday–holiday, (c) a holiday–workday and (d) a holiday–workday transition

have hitherto been generally possible. A particular advantage of our approach is that it can be implemented by relatively straightforward extension of existing methods, while delivering very substantial improvements both in the size of data set that can be modelled and in some cases the speed of fitting. The smooth Poisson regression air pollution example, which was introduced in Section 1 and is developed further in the on-line supporting material, provides a stark example of the practical improvements. In that case the model matrix alone would require over 7 Gbytes of storage if formed in one go, but we can fit the model by using less than 1 Gbyte of storage. We know of no other publicly available approach that could fit a model of broadly the structure that we used to the air pollution data set. Estimation (which took less than 12 min on the cheap computer that was described in Section 4) was also about 100 times faster than existing methods would be predicted to manage, if storage was no problem.

In addition to the fields that motivated this work, remote sensing, gene technology, finance and developments in informatics are all generating increasing numbers of very large data sets. Sometimes such data require completely new methods of analysis, but other times well-established existing model classes are also useful, and all that is required is some way of making their use computationally feasible. The approach that is provided in this paper does this and should therefore be quite widely applicable, beyond the immediate applications that are given here.

The methods that are discussed in this paper are implemented in R package `mgcv` (Wood, 2009) as function `bam`.

Acknowledgements

The authors are grateful to two referees, the Associate Editor and the Joint Editor for comments which helped to improve this paper. Simon Wood is funded by Engineering and Physical Sciences Research Council grant EP/K005251/1.

Appendix A: *QR*-based modelling details

This appendix provides further details of how model matrix $\mathbf{X}$ can be replaced by its QR-factor $\mathbf{R}$ for smoothness selection and coefficient estimation.

A.1. Smoothness selection criteria

In addition to GCV, which was covered in Section 2, Mallows's C_p and REML criteria can also be used for smoothness selection.

If ϕ is known then attempting to minimize an estimate of the prediction error leads to minimization of Mallows's (1973) C_p (Craven and Wahba's (1979) 'unbiased risk estimate')

$$C_p(\boldsymbol{\lambda}) = \|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}_\lambda\|^2 + 2\phi \operatorname{tr}(\mathbf{F}_\lambda)$$

with respect to the smoothing parameters.

Alternatively we can treat the $\mathbf{S}_j$ as precision matrices of Gaussian random effects and estimate the λ_j by marginal likelihood or REML, i.e. we seek λ_j to minimize

$$\mathcal{V}_r(\boldsymbol{\lambda}) = \frac{\|\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}_\lambda\|^2 + \hat{\boldsymbol{\beta}}_\lambda^{\mathrm{T}}\mathbf{S}_\lambda\hat{\boldsymbol{\beta}}_\lambda}{2\phi} + \frac{n - M_p}{2}\log(2\pi\phi) + \frac{\log|\mathbf{X}^{\mathrm{T}}\mathbf{X} + \mathbf{S}_\lambda| - \log|\mathbf{S}_\lambda|_+}{2}$$

where $|\mathbf{S}_\lambda|_+$ is the product of the strictly positive eigenvalues of $\mathbf{S}_\lambda$, and M_p is the (formal) degree of rank deficiency of $\mathbf{S}_\lambda$.

Both Mallows's C_p and the REML criterion can be re-expressed in terms of $\mathbf{f}$, $\|\mathbf{r}\|^2$ and $\mathbf{R}$ from Section 2 as follows:

$$C_p(\boldsymbol{\lambda}) = \|\mathbf{f} - \mathbf{R}\hat{\boldsymbol{\beta}}_\lambda\|^2 + \|\mathbf{r}\|^2 + 2\phi \operatorname{tr}(\mathbf{F}_\lambda)$$

and

$$\mathcal{V}_r(\boldsymbol{\lambda}) = \frac{\|\mathbf{f} - \mathbf{R}\hat{\boldsymbol{\beta}}_\lambda\|^2 + \|\mathbf{r}\|^2 + \hat{\boldsymbol{\beta}}_\lambda^{\mathrm{T}}\mathbf{S}_\lambda\hat{\boldsymbol{\beta}}_\lambda}{2\phi} + \frac{n - M_p}{2}\log(2\pi\phi) + \frac{\log|\mathbf{R}^{\mathrm{T}}\mathbf{R} + \mathbf{S}_\lambda| - \log|\mathbf{S}_\lambda|_+}{2}.$$

A.2. Alternative schemes

Two alternatives to the approach that was suggested in Section 2 deserve comment.

(a) In principle we could base all inference on the working linear model $\mathbf{f} = \mathbf{R}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} \sim N(\mathbf{0}, \mathbf{I}\phi)$, in which case $\|\mathbf{r}\|^2$ would play no further part. If ϕ is known, then inference is invariant to such a

change, but if ϕ is unknown then dropping $\|\mathbf{r}\|^2$ typically leads to poor results, since $\|\mathbf{r}\|^2$ contains substantial information about ϕ.

(b) A superficially attractive alternative for avoiding high storage costs is to use sparse representations for the component f_j, so that $\mathbf{X}$ is a sparse matrix, incurring low storage costs (e.g. by using the `Matrix` library of Bates and Maechler (2013)). However, in tests, the overheads that are associated with sparse computation meant that we could not produce a practical advantage for this approach in terms of either speed or storage requirements, and it substantially restricts the range of smoothers that can be employed. In any case the sparsity of $\mathbf{X}$ does not carry over to $\mathbf{R}$ ($\mathbf{R}$ is also the Choleski factor of $\mathbf{X}^T\mathbf{X}$, which usually is nearly dense when there is more than one f_j).

Appendix B: Obtaining R, f and $\|r\|^2$ without forming X

For a large data set and reasonably flexible model, $\mathbf{X}$ can become too large to fit into computer memory. Indeed, any fitting method with a memory footprint that is a multiple of the size of $\mathbf{X}$ will run out of memory for substantially smaller data set sizes. For example Wood (2011) required storage of an $n \times p$ matrix for each smoothing parameter λ_j. From Section 2 it is clear that such problems can be avoided if $\mathbf{R}$, $\mathbf{f}$ and $\|\mathbf{r}\|^2$ can be obtained without having to form $\mathbf{X}$ in one go. Two approaches are possible.

B.1. QR-updating

Consider constructing a QR-decomposition of a partitioned $\mathbf{X}$. Suppose that

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_0 \\ \mathbf{X}_1 \end{pmatrix},$$

and similarly

$$\mathbf{y} = \begin{pmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \end{pmatrix}.$$

$\mathbf{X}_0$ and $\mathbf{y}_0$ both have n_0 rows, whereas $\mathbf{X}_1$ and $\mathbf{y}_1$ both have n_1 rows. $n_0 + n_1 = n$. Now form QR-decompositions $\mathbf{X}_0 = \mathbf{Q}_0\mathbf{R}_0$ and

$$\begin{pmatrix} \mathbf{R}_0 \\ \mathbf{X}_1 \end{pmatrix} = \mathbf{Q}_1\mathbf{R}.$$

It is routine to check that $\mathbf{X} = \mathbf{QR}$ where

$$\mathbf{Q} = \begin{pmatrix} \mathbf{Q}_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \mathbf{Q}_1$$

($\mathbf{I}$ is $n_1 \times n_1$ here) and

$$\mathbf{Q}^T\mathbf{y} = \mathbf{Q}_1^T \begin{pmatrix} \mathbf{Q}_0^T\mathbf{y}_0 \\ \mathbf{y}_1 \end{pmatrix}.$$

Repeated application of such a construction enables $\mathbf{R}$ and $\mathbf{f}$ to be obtained by considering only one subblock of $\mathbf{X}$ at a time. With enough blocks, the memory footprint can be reduced to a small proportion of what would be necessary if $\mathbf{X}$ were formed whole. If $\mathbf{X}$ is $n \times p$ and there are M blocks it is readily seen that the operations count for this approach is $O(np^2 + Mp^3)$ as opposed to $O(np^2)$ for explicit formation of $\mathbf{X}$, i.e. when n is substantially larger than p the overhead is minor. $\|\mathbf{r}\|^2 = \|\mathbf{y}\|^2 - \|\mathbf{f}\|^2$ provides the remaining ingredient that is required for estimation. Various types of QR-updating are quite widely used, including in ordinary regression modelling: see Golub and van Loan (1996), section 12.5, for some discussion of QR-updating methods.

An advantage of this simple approach is that most of the work is 'embarrassingly parallel'. The data can be divided between m processors, each of which accumulates $\mathbf{R}_i$ and $\mathbf{f}_i$ for its data subset. At the end the required $\mathbf{R}$ and $\mathbf{f}$ are derived from one further QR-decomposition:

$$\begin{pmatrix} \mathbf{R}_1 \\ \vdots \\ \mathbf{R}_m \end{pmatrix} = \tilde{\mathbf{Q}}\mathbf{R}, \qquad \mathbf{f} = \tilde{\mathbf{Q}}^{\mathrm{T}} \begin{pmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_m \end{pmatrix}. \tag{7}$$

B.2. Cholesky and $\mathbf{X}^{\mathrm{T}}\mathbf{X}$-updating

If $\mathbf{X} = \mathbf{QR}$ then clearly $\mathbf{X}^{\mathrm{T}}\mathbf{X} = \mathbf{R}^{\mathrm{T}}\mathbf{R}$, so $\mathbf{R}$ is the Choleski factor of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$. If $\mathbf{X}$ is partitioned rowwise into blocks $\mathbf{X}_1, \mathbf{X}_2, \ldots$ then

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

can be used to accumulate $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ without having to form $\mathbf{X}$ whole. Once $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ is available, $\mathbf{R}$ can be computed by Choleski decomposition, whereas $\mathbf{f} = \mathbf{R}^{-\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{y}$ and, as before, $\|\mathbf{r}\|^2 = \|\mathbf{y}\|^2 - \|\mathbf{f}\|^2$. $\mathbf{X}^{\mathrm{T}}\mathbf{y}$ can be accumulated at the same time as $\mathbf{X}^{\mathrm{T}}\mathbf{X}$. Again most of the process is embarrassingly parallel. The advantage of this approach is that it approximately halves the leading order operations count, relatively to QR-updating. The disadvantage is decreased numerical stability, relatively to QR-updating (see for example Wood (2004)), and the need for regularization in the event of rank deficiency of $\mathbf{X}$.

B.3. On-line updating

For large models that are used for prediction, such as the electricity grid load prediction model of Section 4, the QR-updating approach provides an obvious way of updating models as new data become available. Provided that the new data do not force a change in any of the bases that are used to represent the smooth terms, then the following process is possible.

Step 1: use the new data to update $\mathbf{R}$, $\mathbf{f}$ and $\|\mathbf{r}\|^2$, exactly as shown above when processing a newly computed block of the model matrix.

Step 2: re-estimate the smoothing parameters $\boldsymbol{\lambda}$ and coefficients $\boldsymbol{\beta}_\lambda$ basing the smoothness selection criterion on the updated terms from step 1, and using the smoothing parameters and ϕ-estimates from the original fit as starting values for this optimization.

References

Bates, D. and Maechler, M. (2013) Matrix: sparse and dense matrix classes and methods. *R Package Version 1.0-12*. (Available from `http://CRAN.R-project.org/package=Matrix`.)

de Boor, C. (1978) *A Practical Guide to Splines*. New York: Springer.

Breslow, N. E. and Clayton, D. G. (1993) Approximate inference in generalized linear mixed models. *J. Am. Statist. Ass.*, **88**, 9–25.

Craven, P. and Wahba, G. (1979) Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross validation. *Numer. Math.*, **31**, 377–403.

Eilers, P. H. C., Gampe, J., Marx, B. D. and Rau, R. (2008) Modulation models for seasonal time series and incidence tables. *Statist. Med.*, **27**, 3430–3441.

Eilers, P. H. C. and Marx, B. D. (1996) Flexible smoothing with B-splines and penalties. *Statist. Sci.*, **11**, 89–121.

Fahrmeir, L., Kneib, T. and Lang, S. (2004) Penalized structured additive regression for space-time data: a Bayesian perspective. *Statist. Sin.*, **14**, 731–761.

Golub, G. H. and van Loan, C. F. (1996) *Matrix Computations*, 3rd edn. Baltimore: Johns Hopkins University Press.

Green, P. J. and Silverman, B. W. (1994) *Nonparametric Regression and Generalized Linear Models*. London: Chapman and Hall.

Gu, C. (1992) Cross-validating non-Gaussian data. *J. Computnl Graph. Statist.*, **1**, 169–179.

Gu, C. (2002) *Smoothing Spline ANOVA Models*. New York: Springer.

Gu, C. and Kim, Y.-J. (2002) Penalized likelihood regression: general formulation and efficient approximation. *Can. J. Statist.*, **30**, 619–628.

Hastie, T. and Tibshirani, R. (1986) Generalized additive models (with discussion). *Statist. Sci.*, **1**, 297–318.

Hastie, T. and Tibshirani, R. (1990) *Generalized Additive Models*. London: Chapman and Hall.

Mallows, C. L. (1973) Some comments on C_p. *Technometrics*, **15**, 661–675.
Marx, B. D. and Eilers, P. H. (1999) Generalized linear regression on sampled signals and curves: a P-spline approach. *Technometrics*, **41**, 1–13.
Parker, R. and Rice, J. (1985) Discussion on 'Some aspects of the spline smoothing approach to non-parametric regression curve fitting' (by B. W. Silverman). *J. R. Statist Soc.* B, **47**, 40–42.
Peng, R. D. and Welty, L. J. (2004) The NMMAPSdata package. *R News*, **4**, 10–14.
Pierrot, A. and Goude, Y. (2011) Short term electricity load forecasting with generalized additive models. In *Proc. 16th Int. Conf. Intelligent System Applications to Power Systems*, pp. 410–415. New York: Institute of Electrical and Electronics Engineers.
Reiss, P. T. and Ogden, R. T. (2009) Smoothing parameter selection for a class of semiparametric linear models. *J. R. Statist. Soc.* B, **71**, 505–523.
Wood, S. N. (2000) Modelling and smoothing parameter estimation with multiple quadratic penalties. *J. R. Statist. Soc.* B, **62**, 413–428.
Wood, S. N. (2004) Stable and efficient multiple smoothing parameter estimation for generalized additive models. *J. Am. Statist. Ass.*, **99**, 673–686.
Wood, S. N. (2006) *Generalized Additive Models: an Introduction with R*. Boca Raton: Chapman and Hall–CRC.
Wood, S. N. (2009) mgcv. *R Package Version 1.6-0*. (Available from `http://CRAN.R-project.org/package=mgcv`.)
Wood, S. N. (2011) Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *J. R. Statist. Soc.* B, **73**, 3–36.

Supporting information

Additional 'supporting information' may be found in the on-line version of this article:

'Supporting material for: Generalized additive models for large datasets'.