Future extensions may provide further simulation algorithms for Gaussian and non-Gaussian random fields, and a basic toolbox for the analysis of geostatistical and spatial extreme value data.

Use `help(RandomFields)` to obtain the main man page. To start with, the examples in `help(GaussRF)` are recommended.

Acknowledgement. The work has been supported by the EU TMR network ERB-FMRX-CT96-0095 on "Computational and statistical methods for the analysis of spatial data" and the German Federal Ministry of Research and Technology (BMFT) grant PT BEO 51-0339476C. The author is thankful to Tilmann Gneiting, Martin Mächler, and Paulo Ribeiro for hints and discussions.

Bibliography

[1] M. Schlather. An introduction to positive definite functions and to unconditional simulation of random fields. Technical Report ST-99-10, Lancaster University, 1999. 19

Martin Schlather
University of Bayreuth, Germany
Martin.Schlather@uni-bayreuth.de

mgcv: GAMs and Generalized Ridge Regression for R

by Simon N. Wood

Generalized Additive Models (GAMs) have become quite popular as a result of the work of Wahba (1990) and co-workers and Hastie & Tibshirani (1990). Package **mgcv** provides tools for GAMs and other generalized ridge regression. This article describes how GAMs are implemented in **mgcv**: in particular the innovative features intended to improve the GAM approach. The package aims to provide the convenience of GAM modelling in S-PLUS, combined with much improved model selection methodology. Specifically, the degrees of freedom for each smooth term in the model are chosen simultaneously as part of model fitting by minimizing the Generalized Cross Validation (GCV) score of the whole model (not just component wise scores). At present **mgcv** only provides one dimensional smooths, but multi-dimensional smooths will be available from version 0.6, and future releases will include anisotropic smooths. GAMs as implemented in **mgcv** can be viewed as low rank approximations to (some of) the generalized spline models implemented in **gss** — the idea is to preserve most of the practical advantages with which elegant underlying theory endows the generalized smoothing spline approach, but without the formidable computational burden that accompanies full gss models of moderately large data sets.

GAMs in mgcv

GAMs are represented in **mgcv** as penalized generalized linear models (GLMs), where each smooth term of a GAM is represented using an appropriate set of basis functions and has an associated penalty measuring its wiggliness: the weight given to each penalty in the penalized likelihood is determined by its "smoothing parameter". Models are fitted by the usual iteratively re-weighted least squares scheme for GLMs, except that the least squares problem at each iterate is replaced by a penalized least squares problem, in which the set of smoothing parameters must be estimated alongside the other model parameters: the smoothing parameters are chosen by GCV. This section will sketch how this is done in a little more detail.

A GLM relating a univariate response variable y to a set of explanatory variables $x_1, x_2, \ldots$, has the general form:

$$g(\mu_i) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots \qquad (9.1)$$

where $E(y_i) \equiv \mu_i$ and the y_i are independent observations on r.v.s all from the same member of the exponential family. g is a smooth monotonic "link-function" that allows a useful degree of non-linearity into the model structure. The β_i are model parameters: likelihood theory provides the means for estimation and inference about them. The r.h.s. of (9.1) is the "linear predictor" of the GLM, and much of the statistician's modelling effort goes into finding an appropriate form for this.

The wide applicability of GLMs in part relates to the generality of the form of the of the linear predictor: the modeller is not restricted to including explanatory variables in their original form, but can include transformations of explanatory variables and dummy variables in whatever combinations are appropriate. Hence the class of models is very rich, including, for example, polynomial regression models and models for designed experiments. However the

standard methods for generalized linear modelling can become unwieldy as models become more complex. In particular, it is sometimes the case that prior beliefs about appropriate model structure might best be summarized as something like:

$$g(\mu_i) = \beta_0 + s_1(x_{1i}) + s_2(x_{2i}) + \cdots \tag{9.2}$$

i.e., the linear predictor should be given by a constant plus a smooth function of x_1 plus another smooth function of x_2 and so on (with some side conditions on the s_i to ensure identifiability). It is *possible* to build this sort of model structure directly within the GLM framework using, e.g. polynomials or more stable bases to represent the smooth terms: but such an approach becomes troublesome as the number of smooths and their complexity increases. The two main problems are that model selection becomes rather cumbersome (many models may need to be compared, and it is not always easy to keep them nested), and that the basis selected can have a rather strong influence on the fitted model (e.g. regression splines tend to be rather dependent on knot placement, while polynomials can be very unstable).

An alternative approach for working with models like (9.2) represents the smooth functions using linear smoothers, and performs estimation by backfitting (Hastie & Tibshirani, 1990) — this has the advantage that a very wide range of smoothers can be used, but the disadvantage that model selection (choosing the amount of smoothing to perform) is still difficult.

In **mgcv**, smooth terms in models like (9.2) are represented using penalized regression splines. That is, the smooth functions are re-written using a suitably chosen set of basis functions, and each has an associated penalty which enables its effective degrees of freedom to be controlled through a single smoothing parameter. How this works is best seen through an example, so consider a model with one linear term and a couple of smooth terms:

$$g(\mu_i) = \beta_0 + \beta_1 x_{1i} + s_1(x_{2i}) + s_2(x_{3i}) \tag{9.3}$$

The s_i can be re-written in terms of basis functions thus:

$$s_1(x) = \sum_{j=1}^{k_1} \beta_{j+1} b_{1j}(x) \quad s_2(x) = \sum_{j=1}^{k_2} \beta_{j+1+k_1} b_{2j}(x)$$

where k_i is the number of basis functions used for s_i, the β_j are parameters to be estimated and the b_{ji} are basis functions. For example, a suitable set of spline-like basis functions might be:

$$b_{j1}(x) = x \text{ and } b_{ji}(x) = |x - x^*_{ji}|^3 \text{ for } i > 1$$

where the x^*_{ji} are a set of "knots" spread "nicely" throughout the relevant range of explanatory variable values. So (9.3) now becomes:

$$g(\mu_i) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 |x_{2i} - x^*_{22}|^3 + \cdots$$

... a GLM. If we write the vector of values of $g(\mu_i)$ as $\boldsymbol{\eta}$ then it's pretty clear that the previous equation written out for all i can be written as $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$, where the model matrix $\mathbf{X}$ follows in an obvious way from the above equation. (Note that **mgcv** actually uses a different (but equivalent) regression spline basis based on cubic Hermite polynomials: its parameters are usefully interpretable and it is computationally convenient, but rather long winded to write out.) So far the degrees of freedom associated with each smooth term are determined *entirely* by the k_i so that model selection will have all the difficulties alluded to above and the fitted model will tend to show characteristics dependent on knot locations. To avoid these difficulties **mgcv** uses a relatively high value for each k_i and controls the smoothness (and hence degrees of freedom) for each term through a set of penalties applied to the likelihood of the GLM. The penalties measure the wiggliness of each s_i as:

$$\int [s_i''(x)]^2 dx$$

Since $s_1''(x) = \sum_{j=1}^{k_1} \beta_{j+1} b_{1j}''(x)$, it's not hard to see that it is possible to write:

$$\int [s_1''(x)]^2 dx = \boldsymbol{\beta}^T \mathbf{S}_1 \boldsymbol{\beta}$$

where $\boldsymbol{\beta}$ is the parameter vector, and $\mathbf{S}_1$ is a positive semi-definite matrix depending only on the basis functions. A similar result applies to s_2. So the model β_i's can be estimated by minimizing:

$$-l(\boldsymbol{\beta}) + \sum_{i=1}^{2} \lambda_i \boldsymbol{\beta}^T \mathbf{S}_i \boldsymbol{\beta}$$

where l is the log-likelihood for $\boldsymbol{\beta}$, and the λ_i's control the relative weight given to the conflicting goals of good fit and model smoothness. *Given* λ_i it is straightforward to solve this problem by (penalized) IRLS, but the λ_i need to be estimated, and this is not so straightforward.

Recall that the IRLS method for a GLM consists of iterating the following steps to convergence:

1. The current estimate of $\boldsymbol{\beta}$, $\boldsymbol{\beta}^{[k]}$, yields estimates of $\boldsymbol{\mu}$ and the variance of each y_i: V_i. Hence using the current estimates, calculate the following: (i) the diagonal weight matrix $\mathbf{W}$ where:

   $$W_{ii} = [g'(\mu_i)^2 V_i]^{-1}$$

   and (ii) the vector of pseudodata:

   $$\mathbf{z} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\Gamma}(\mathbf{y} - \boldsymbol{\mu})$$

   where $\boldsymbol{\Gamma}$ is a diagonal matrix and $\Gamma_{ii} = [g'(\mu_i)]^{-1}$.

2. Minimize:

$$\|\mathbf{W}^{1/2}(\mathbf{z}-\mathbf{X}\boldsymbol{\beta})\|^2$$

w.r.t. $\boldsymbol{\beta}$ to get $\boldsymbol{\beta}^{[k+1]}$.

mgcv fits GAMs by replacing step 2. with the following:

2. Find the λ_i minimizing:

$$\frac{\|\mathbf{W}^{1/2}(\mathbf{z}-\mathbf{X}\boldsymbol{\beta})\|^2}{[\mathrm{tr}(\mathbf{I}-\mathbf{A})]^2} \tag{9.4}$$

where $\boldsymbol{\beta}$ is the solution to the problem of minimizing:

$$\|\mathbf{W}^{1/2}(\mathbf{z}-\mathbf{X}\boldsymbol{\beta})\|^2 + \sum \lambda_j \boldsymbol{\beta}^T \mathbf{S}_j \boldsymbol{\beta}$$

w.r.t. $\boldsymbol{\beta}$, and $\mathbf{A}$ is the "influence" or "hat" matrix: $\mathbf{X}(\mathbf{X}^T\mathbf{W}\mathbf{X} + \sum \lambda_j \boldsymbol{\beta}^T \mathbf{S}_j \boldsymbol{\beta})^{-1}\mathbf{X}^T\mathbf{W}$, whose trace gives the estimated degrees of freedom for the model.

(9.4) is the GCV score for the model and its efficient minimization is the key to the approach used in **mgcv**: the method for doing this is based on a generalization of the method developed by Gu & Wahba (1991) for generalized smoothing spline models, and is described in Wood (2000). The computational burden is cubic in the dimension of $\boldsymbol{\beta}$ — which is usually much less than the computational burden of using Gu and Wahba's method for gss models, which is necessarily cubic in the number of data.

Note that direct minimization of (9.4) is not the same as minimizing separate GCV scores as part of each back-fitting iteration — the latter approach is very difficult to justify on other than ad hoc grounds.

A simple Bayesian argument yields a covariance matrix estimate for $\hat{\boldsymbol{\beta}}$ and hence estimated Bayesian confidence intervals for the components of the GAM (Wood, 2000; Hastie & Tibshirani, 1990). These are similar in spirit to the intervals for smoothing splines in Wahba (1983).

Note then, that the key point about **mgcv** is that the selection of degrees of freedom for the components of a fitted GAM is an integral part of model fitting. Furthermore the manner in which it is integrated is designed to make inclusion of multi-dimensional and anisotropic smooths quite straightforward. In addition it should be clear that in principle any smooth constructed using a basis and quadratic penalty could be incorporated into **mgcv**'s GAM modelling tools.

Practical GAMs

Because of the way in which estimation of degrees of freedom is integrated into model fitting, the `gam()` function provided by **mgcv** is not an exact clone of what is described in the white book and implemented in S-PLUS. This section describes what is implemented and how to use it.

`gam()`

mgcv's `gam()` function fits a GAM specified by a model formula and family, to univariate response data. A simple example of its use is:

```
> gam(y ~ s(x))
```

which will cause the model:

$$y_i \sim f(x_i) + \epsilon_i, \quad \epsilon_i \text{ i.i.d. } N(0,\sigma^2)$$

to be estimated, where f is a smooth function.

A more complicated example, illustrating a few more features is:

```
> gam(y^0.5 ~ -1 + x + s(z,5|f) + s(w) + s(v,20),
    data = mydata, family = gamma(link=I))
```

In this case the response is $\sqrt{y}$, and the linear predictor is made up of a linear term in x plus smooth functions of z, w and v, with no intercept term. The data are assumed to follow a gamma distribution, and the link is the identity function. The 3 different forms of `s()` each relate to a different representation of the smooth concerned. `s(z,5|f)` indicates that the smooth function of z is to be represented as a pure un-penalized regression spline, with 5 knots — under the representation used by **mgcv** this corresponds to exactly 4 degrees of freedom ('`|f`' indicates *f*ixed degrees of freedom). `s(w)` indicates that the smooth function of w is to be represented using a default 10 knot penalized regression spline: corresponding to maximum degrees of freedom of 9 for this term. Finally `s(v,20)` indicates that the smooth of v is to be represented by a 20 knot penalized regression spline: this term is being allowed a maximum of 19 degrees of freedom — presumably the relationship between $\sqrt{y}$ and v is expected to be fairly complicated.

The choice of the number of knots is not very critical, but should be somewhat larger than the estimated degrees of freedom plus 1, otherwise: (i) the choice of knot locations will begin to have a visible effect on the shape of the estimated function, and (ii) it is *possible* that if the GCV function is a strange shape then the optimization path to the GCV minimum may pass beyond the upper boundary on degrees of freedom, so that the smoothing parameter estimates become incorrectly stuck at that boundary. In practice I usually start with the default 10 knot splines, but increase the number of knots for any terms that are then estimated to have close to the corresponding maximum 9 degrees of freedom.

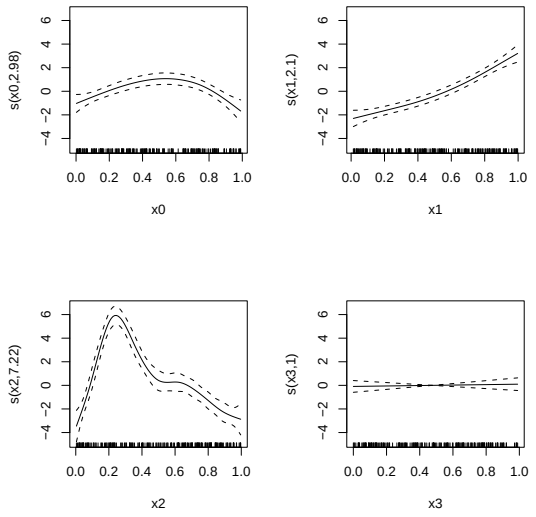
Other arguments to `gam()` will be familiar to users of `lm()` and `glm()`. The exception is the argument `scale`. If the scale parameter for the model distribution is known then there is an argument

for replacing GCV with the unbiased risk estimator (UBRE) given e.g. in Wahba (1990). Hence if `scale` is a positive number it is assumed to be the scale parameter and UBRE is used. If `scale` is zero (the default) then UBRE is used for the Poisson and binomial distributions (scale parameter 1), but GCV is used otherwise. A negative value for the scale parameter forces use of GCV in all cases. Note that GCV is appropriate if over-dispersion is suspected.

Other GAM functions

Package **mgcv** provides versions of `print.gam()`, `predict.gam()` and `plot.gam()`. `plot.gam()` differs most from what is available in S-PLUS — interactive use is missing, for example. Here is an example of its use to plot the results of fitting a simple 4 term model:

```
> gam.model <- gam(y ~ s(x0)+s(x1)+s(x2)+s(x3))
> plot(gam.model, pages = 1)
```



By default the same y axis range is used for all the plots, while the `pages=1` option provides automatic layout on a single page. The rug plots at the foot of each plot show the observed values of each explanatory variable. Each y-axis label indicates what the smooth is a function of and how many degrees of freedom the term has. The solid line in each plot is the estimate of the smooth function, while the dashed lines are at 2 standard errors above and below the estimate — roughly 95% confidence limits. This example used simulated data and `x3` is in fact unrelated to the response: notice how the smooth for `x3` is estimated to have just one degree of freedom, and how its "confidence band" comfortably includes zero everywhere.

It's also of some interest to print the `gam.model`:

```
> gam.model

Family: gaussian
Link function: identity

Formula:
y ~ s(x0) + s(x1) + s(x2) + s(x3)

Estimated degrees of freedom:
 2.982494 2.096610 7.219753 1.000005
 total =  14.29886

GCV score:  4.326104
```

The estimated degrees of freedom for each smooth term are given in the order in which the smooth terms are specified in the model formula — note that the total degrees of freedom includes those associated with purely parametric model components. The GCV score is useful for comparing models with and without particular terms included.

Dropping model terms

While **mgcv** selects the degrees of freedom for each term automatically, the nature of the estimation algorithm means that it can not automatically decide whether to drop a term altogether or not. The reason for this is that a zero term and a straight line have the same zero penalty — hence once a term has become a straight line, increasing its smoothing parameter further can have no effect on its degrees of freedom, and certainly won't force it to become zero. Hence the modeller must remove unwanted terms "by hand". Deciding which terms to remove is straightforward, and should be guided by the answers to 3 questions:

1. Is the estimated degrees of freedom for the term close to 1?

2. Does the plotted confidence band for the term include zero everywhere?

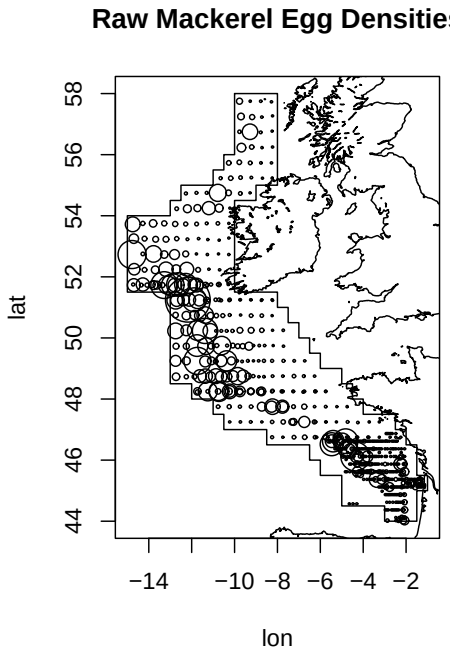
3. Does the GCV score drop when the term is dropped?

If the answer to all three questions is "yes" then the term should be dropped. If the e.d.f. is close to one but the answer to the other 2 questions is "no" then you might as well replace the smooth with a parametric linear term. Other cases will require judgement: for example, very small increases in GCV score shouldn't prevent a term from being dropped. Because of correlations between explanatory variable, terms should only be dropped one at a time: it makes sense to start with the term for which the zero line is most comfortably within the confidence band.

In the plot shown above it's clear that `x3` should be dropped from the model: the example in the next section provides a second illustration.

A fisheries example

As an example of use of GAMs with **mgcv**, consider a set of data originally analysed by Borchers *et al.* (1997) as part of the stock assessment process for the European mackerel fishery.



The data are mackerel egg densities (per m^2 of sea surface) obtained from net hauls undertaken from research boats in 1992 (Bowman & Azzalini, 1997, analyse a subset of these data, with slightly different pre-processing, using a loess model — their subset is available in package **sm**). The plotted symbol sizes are proportional to the egg density. Candidate explanatory variables in this case are longitude, latitude, sea bed depth, sea surface temperature and distance from the 200 metre sea bed depth contour. If a variance stabilizing transformation is employed then a Gaussian error model is not too bad. So, a first model attempt might be:

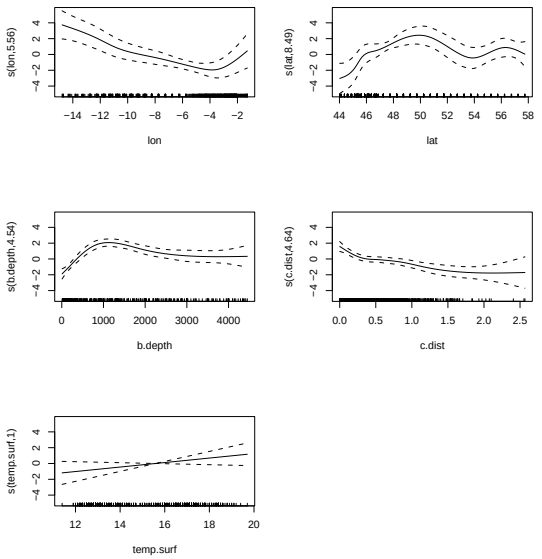
```
> mack.fit <-
+ gam(egg.dens^0.4 ~ s(lon) + s(lat) + s(b.depth)
+       + s(c.dist) + s(temp.surf),
+     data = mack)
```

(data frame `mack` contains the data for the 634 sampling stations). Here are the first results (the fitting took a few seconds on a Pentium II):

```
> mack.fit

Family: gaussian
Link function: identity

Formula:
egg.dens^0.4 ~ s(lon) + s(lat) +
s(b.depth) + s(c.dist) + s(temp.surf)

Estimated degrees of freedom:
 5.55544 8.494712 4.536643 4.63995
 1.000001   total =  25.22675

GCV score:  3.71533
```

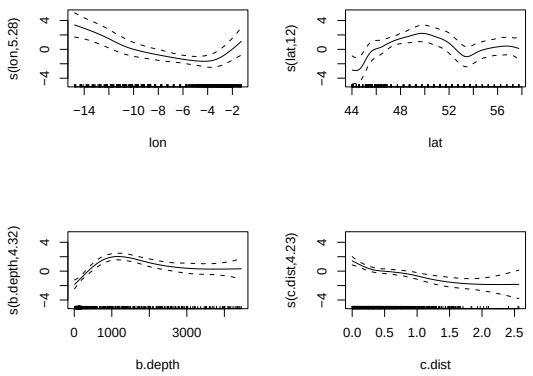
clearly surface temperature is a candidate for removal or replacement by a linear term: the next thing to do is to plot the estimated terms:

```
> plot(mack.fit)
```



So, surface temperature looks like a candidate for removal (and at the same time it's wise to increase the number knots for the latitude term).

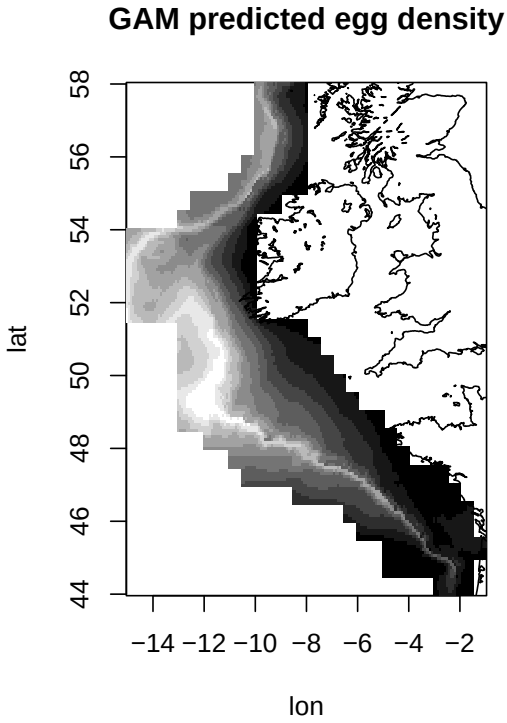
```
> mack.fit2 <-
+ gam(egg.dens^0.4 ~ s(lon) + s(lat, 20)
+       + s(b.depth) + s(c.dist),
+     data = mack)
> mack.fit2

Family: gaussian
Link function: identity

Formula:
egg.dens^0.4 ~ s(lon) + s(lat, 20) +
s(b.depth) + s(c.dist)

Estimated degrees of freedom:
 5.276965 12.00392 4.323457 4.234603
 total =  26.83895

GCV score:  3.709722

> plot(mack.fit2, pages = 1)
```



The GCV score has decreased, supporting the decision to remove the surface temperature term. There being no further terms to delete, the model can be used for prediction. Data frame `mackp` contained the explanatory variables on a fine grid over the survey area. To get a predicted (transformed) density for each point on this grid I used `predict.gam()`:

```
> mack.pred <- predict.gam(mack.fit2, mackp)
```

A certain amount of tedious manipulation was needed to copy this into a matrix suitable for contouring and plotting, but an image plot of the final predicted (transformed) densities looks like this:



Summary and the future

mgcv offers GAM modelling tools with much (but not all) of the functionality of their S-PLUS equivalents, plus the substantial advantage of well founded and efficient methods for selecting the degrees of freedom for each smooth term (and for selecting which terms to include at all). The package currently offers a more limited range of models than the **gss** package, but typically at much lower computational cost, and with slightly greater ease of use for those familiar with GAMs in S-PLUS. Future developments will provide further basic methods, and two further substantial enhancements. Version 0.6 will include multi-dimensional smooth terms, while in the longer term anisotropic smooth terms will be included.

References

Borchers, D. L., S. T. Buckland, I. G. Priede and S. Ahmadi (1997). Improving the precision of the daily egg production method using generalized additive models. *Canadian Journal of Fisheries and Aquatic Science*, **54**, 2727–2742.

Bowman, A. W. and A. Azzalini (1997). *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations*. Clarendon Press.

Gu C. and G. Wahba (1991) Minimizing GCV/GML scores with multiple smoothing parameters via the newton method. *SIAM Journal on Scientific and Statistical Computation*, **12**, 383–398.

Hastie T. J. and R. J. Tibshirani (1990). *Generalized Additive Models*. Chapman & Hall.

Wahba, G. (1983). Bayesian confidence intervals for the cross validated smoothing spline. *Journal of the Royal Statistical Society (B)*, **45**, 133–150.

Wahba, G. (1990). *Spline Models for Observational Data*. SIAM, Philadelphia.

Wood, S. N. (2000). Modelling and smoothing parameter estimation with multiple quadratic penalties. *Journal of the Royal Statistical Society (B)*, **62**, 413–428.

Simon N. Wood
University of St Andrews, Fife, UK
snw@mcs.st-and.ac.uk