GAMs — semi-parametric GLMs

Simon Wood
Mathematical Sciences, University of Bath, U.K.

Generalized linear models, GLM

1. A GLM models a univariate response, y_i as

$$g\{\mathbb{E}(y_i)\} = \mathbf{X}_i\boldsymbol{\beta} \text{ where } y_i \sim \text{Exponential family}$$

2. g is a known, smooth monotonic *link function*.
3. $\mathbf{X}_i$ is the ith row of a known *model matrix*, which depends on measured predictor variables (covariates).
4. $\boldsymbol{\beta}$ is an **unknown** parameter vector, estimated by MLE.
5. $\mathbf{X}\boldsymbol{\beta}$ $(= \boldsymbol{\eta})$ is the *linear predictor* of the model.
6. Class includes log-linear models, general linear regression, logistic regression,...
7. `glm` in R implements this class.

Generalized additive models, GAM

1. A GAM (semi-parametric GLM) is a GLM where the linear predictor depends linearly on *unknown smooth functions*.
2. In general

$$g\{\mathbb{E}(y_i)\} = \mathbf{A}_i\boldsymbol{\theta} + \sum_j L_{ij}f_j \text{ where } y_i \sim \text{Exponential family}$$

3. Parameters, $\boldsymbol{\theta}$, and smooth functions, f_j, are **unknown**.
4. Parametric model matrix, $\mathbf{A}$, and linear functionals, L_{ij}, depend on predictor variables.
5. Examples of $L_{ij}f_j$: $f_1(x_i)$, $f_2(z_i)w_i$, $\int k_i(v)f_j(v)dv$, ...

Specifying GAMs in R with mgcv

- `library(mgcv)` loads a semi-parametric GLM package.
- `gam(formula,family)` is quite like `glm`.
- The `family` argument specifies the distribution and link function. e.g. `Gamma(log)`.
- The response variable and linear predictor structure are specified in the model `formula`.
- Response and parametric terms exactly as for `lm` or `glm`.
- Smooth functions are specified by `s` or `te` terms. e.g.

$$\log\{\mathbb{E}(y_i)\} = \alpha + \beta x_i + f(z_i), \;\; y_i \sim \text{Gamma},$$

is specified by...

```
gam(y ~ x + s(z),family=Gamma(link=log))
```

More specification: `by` variables

- Smooth terms can accept a `by` variable argument, which allows $L_{ij} f_j$ terms other than just $f_j(x_i)$.
- e.g. $\mathbb{E}(y_i) = f_1(z_i) w_i + f_2(v_i)$, $y_i \sim \text{Gaussian}$, becomes

  ```
  gam(y ~ s(z,by=w) + f(v))
  ```

  i.e. $f_1(x_i)$ is multiplied *by* w_i in the linear predictor.
- e.g. $\mathbb{E}(y_i) = f_j(x_i, z_i)$ if factor g_i is of level j, becomes

  ```
  gam(y ~ s(x,z,by=g) + g)
  ```

  i.e. there is one smooth function for each level of factor variable `g`, with each y_i depending on just one of these functions.

Yet more specification: a summation convention

- `s` and `te` smooth terms accept *matrix* arguments and `by` variables to implement general $L_{ij} f_j$ terms.
- If **X** and **L** are $n \times p$ matrices then

  ```
  s(X,by=L)
  ```

  evaluates $L_{ij} f_j = \sum_k f(X_{ik}) L_{ik}$ for all i.
- For example, consider data $y_i \sim \text{Poi}$ where

  $$\log\{\mathbb{E}(y_i)\} = \int k_i(x) f(x) dx \simeq \frac{1}{h} \sum_{k=1}^{p} k_i(x_k) f(x_k)$$

  (the x_k are evenly spaced points).
- Let $X_{ik} = x_k \ \forall \ i$ and $L_{ik} = k_i(x_k)/h$. The model is fit by

  ```
  gam(y ~ s(X,by=L),poisson)
  ```

How to estimate GAMs?

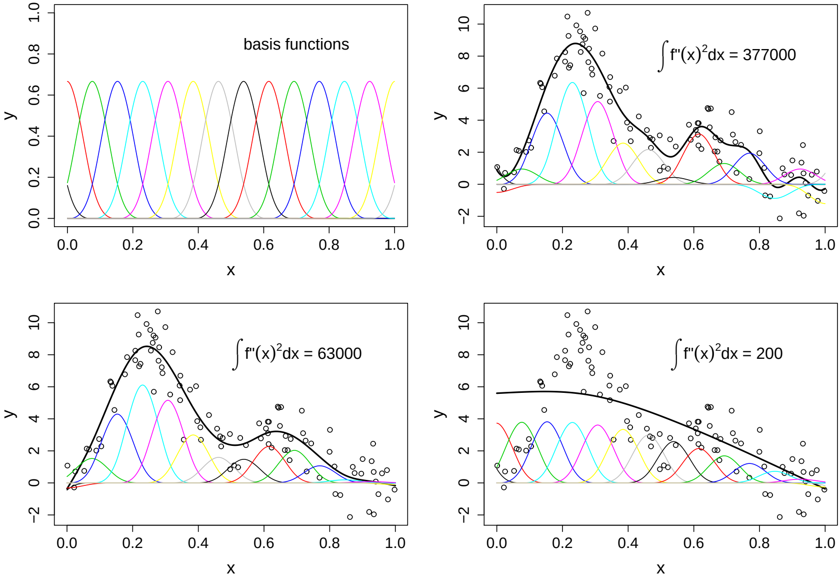
- The `gam` calls on the previous slides would also *estimate* the specified model — it's useful to have some idea how.
- We need to estimate the parameters, $\boldsymbol{\theta}$, and the smooth functions, f_j.
- This includes estimating *how smooth* the f_j are.
- To begin with we need decide on two things
  1. How to represent the f_j by something computable.
  2. How to formalize what is meant by *smooth*.
- ... here we'll discuss only the approach based on representing the f_j with penalized regression splines (as in R package, `mgcv`).

Bases and penalties for the f_j

- Represent each f as a weighted sum of known basis functions, b_k,
$$f(x) = \sum_{k=1}^{K} \gamma_k b_k(x)$$
Now only the γ_k are unknown.
- *Spline function* theory supplies good choices for the b_k.
- K is chosen large enough to be sure that $\text{bias}(\hat{f}) \ll \text{var}(\hat{f})$.
- Choose a measure of function *wiggliness* e.g.

$$\int f''(x)^2 dx = \boldsymbol{\gamma}^{\mathrm{T}} \boldsymbol{\mathcal{S}} \boldsymbol{\gamma}, \quad \text{where } \mathcal{S}_{ij} = \int b_i''(x) b_j''(x) dx.$$

Basis & penalty example



A computable GAM

- Representing each f_j with basis functions b_{jk} we have,

$$g\{\mathbb{E}(y_i)\} = \mathbf{A}_i\boldsymbol{\theta} + \sum_j L_{ij} f_j = \mathbf{A}_i\boldsymbol{\theta} + \sum_j \sum_k \gamma_{jk} L_{ij} b_{jk} = \mathbf{X}\boldsymbol{\beta}$$

  where $\mathbf{X}$ contains $\mathbf{A}$ and the $L_{ij}b_{jk}$, while $\boldsymbol{\beta}$ contains $\boldsymbol{\theta}$ and the γ_j vectors.
- For notational convenience we can re-write the wiggliness penalties as $\gamma_j^{\mathrm{T}}\boldsymbol{\mathcal{S}}_j\gamma_j = \boldsymbol{\beta}^{\mathrm{T}}\mathbf{S}_j\boldsymbol{\beta}$, where $\mathbf{S}_j$ is just $\boldsymbol{\mathcal{S}}_j$ padded with extra zeroes.
- So the GAM has become a parametric GLM, with some associated penalties.
- The $L_{ij}f_j$ are often confounded via the intercept. So the f_j are usually constrained to sum to zero, over the observed values. A reparameterization achieves this.

Estimating the model

- If the bases are large enough to be sure of avoiding bias, then MLE of the model will overfit/undersmooth.
- …so we use maximum *penalized* likelihood estimation.

$$\text{minimize} \quad -2l(\boldsymbol{\beta}) + \sum_j \lambda_j \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta} \ \text{ w.r.t. } \ \boldsymbol{\beta} \tag{1}$$

  where l is the log likelihood.
- $\lambda_j \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta}$ forces f_j to be smooth.
- *How* smooth is controlled by λ_j, which must be chosen.
- For now suppose that an angel has revealed values for $\boldsymbol{\lambda}$ in a dream. We'll eliminate the angel later.
- Given $\boldsymbol{\lambda}$, (1) is optimized by a penalized version of the IRLS algorithm used for GLMs

Bayesian motivation

- We can be more principled about motivating (1).
- Suppose that our *prior belief* is that smooth models are more probable than wiggly ones.
- We can formalize this belief with an exponential prior on wiggliness
$$\propto exp(-\frac{1}{2}\sum_j \lambda_j \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta})$$
- $\Rightarrow$ an improper Gaussian prior $\boldsymbol{\beta} \sim N(\mathbf{0}, (\sum_j \lambda_j \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S}_j \boldsymbol{\beta})^{-})$.
- In this case the posterior modes, $\hat{\boldsymbol{\beta}}$, of $\boldsymbol{\beta}|\mathbf{y}$ are the minimizers of (1).

The distribution of $\boldsymbol{\beta}|\mathbf{y}$

- The Bayesian approach gives us more...
- For any exponential family distribution $\text{var}(y_i) = \phi V(\mu_i)$ where $\mu_i = \mathbb{E}(y_i)$. Let $W_{ii}^{-1} = V(\mu_i) g'(\mu_i)^2$.
- The Bayesian approach gives the large sample result

$$\boldsymbol{\beta}|\mathbf{y} \sim N(\hat{\boldsymbol{\beta}}, (\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X} + \sum_j \lambda_j \mathbf{S}_j)^{-1}\phi)$$

- This can be used to get CIs with good frequentist properties (by an argument of Nychka, 1988, JASA).
- Simulation from $\boldsymbol{\beta}|\mathbf{y}$ is a very cheap way of making any further inferences required.
- `mgcv` uses this result for inference (e.g. `?vcov.gam`).

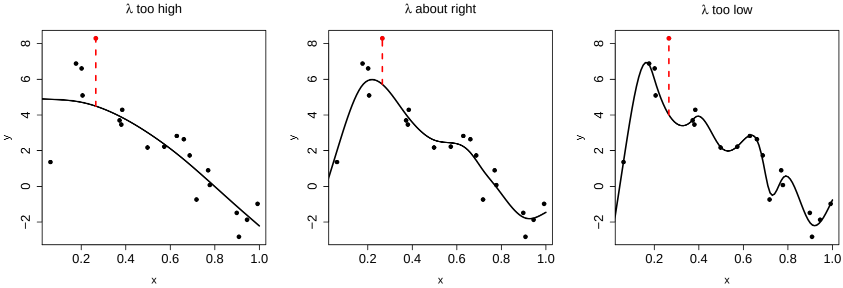
Degrees of Freedom, DoF

- Penalization reduces the freedom of the parameters to vary.
- So $\dim(\boldsymbol{\beta})$ is not a good measure of the DoF of a GAM.
- A better measure considers the average degree of shrinkage of the coefficients, $\hat{\boldsymbol{\beta}}$.
- Informally, suppose $\tilde{\boldsymbol{\beta}}$ is the unpenalized parameter vector estimate, then approximately $\hat{\boldsymbol{\beta}} = \mathbf{F}\tilde{\boldsymbol{\beta}}$ where $\mathbf{F} = (\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X} + \sum_j \lambda_j \mathbf{S}_j)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X}$.
- The F_{ii} are shrinkage factors and $\mathrm{tr}(\mathbf{F}) = \sum_i F_{ii}$ is a measure of the *effective degrees of freedom* (EDF) of the model.

Estimating the smoothing parameters, $\boldsymbol{\lambda}$

- We need to estimate the appropriate degree of smoothing, i.e. $\boldsymbol{\lambda}$.
- This involves
  1. Deciding on a statistical approach to take.
  2. Producing a computational method to implement it.
- There are 3 main statistical approaches
  1. Choose $\boldsymbol{\lambda}$ to minimize error in predicting new data.
  2. Treat smooths as random effects, following the Bayesian smoothing model, and estimate the λ_j as variance parameters using a marginal likelihood approach.
  3. Go fully Bayesian by completing the Bayesian model with a prior on $\boldsymbol{\lambda}$ (requires simulation and not pursued here).

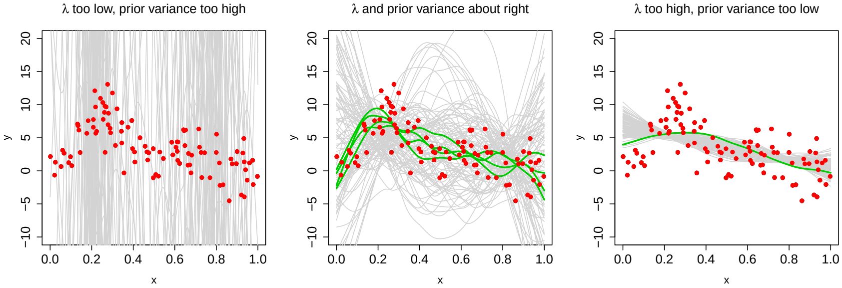
A prediction error criterion: cross validation



1. Choose λ to try to minimize the error predicting new data.
2. Minimize the average error in predicting single datapoints *omitted* from the fit. Each datum left out once in average.
3. Invariant version is Generalized Cross Validation, GCV:

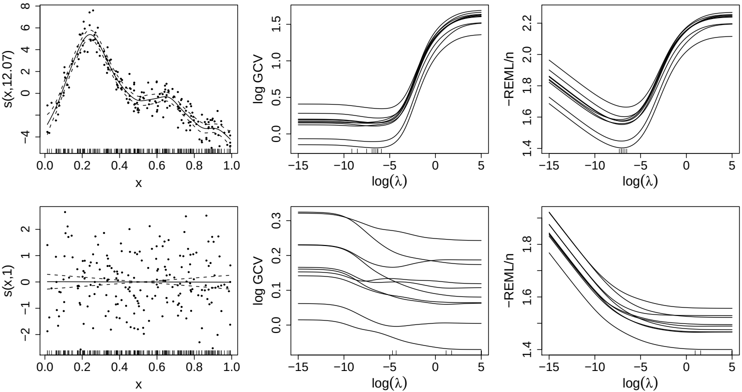
$$\mathcal{V}_g(\lambda) = \frac{l_{max} - l(\hat{\boldsymbol{\beta}}_\lambda)}{(n - EDF_\lambda)^2}$$

Marginal likelihood based smoothness selection



1. Choose λ to maximize the average likelihood of random draws from the prior implied by λ.
2. If λ too low, then almost all draws are too variable to have high likelihood. If λ too high, then draws all underfit and have low likelihood. The right λ maximizes the proportion of draws close enough to data to give high likelihood.
3. Formally, maximize e.g. $\mathcal{V}_r(\lambda) = \log \int f(\mathbf{y}|\beta) f_\lambda(\beta) d\beta$.

Prediction error vs. likelihood λ estimation



1. Pictures show GCV and REML scores for different replicates from same truth.
2. Compared to REML, GCV penalizes overfit only weakly, and so tends to undersmooth.

Computational strategies for smoothness selection

1. Single iteration. Use an *approximate* $\mathcal{V}$ to re-estimate $\boldsymbol{\lambda}$ at each step of the PIRLS iteration used to find $\hat{\boldsymbol{\beta}}$.
2. Nested iteration. Optimize $\mathcal{V}$ w.r.t. λ by a Newton method, with each trial $\boldsymbol{\lambda}$ requiring a full PIRLS iteration to find $\hat{\boldsymbol{\beta}}_\lambda$.

- 1 need not converge and is often no cheaper than 2.
- 2 is *much* harder to implement efficiently.
- In `mgcv`, 1 is known as *performance iteration*, and 2 is *outer iteration*.
  - `gam` defaults to 2 (see `method` and `optimizer` arguments).
  - `gamm` (mixed GAMs) and `bam` (large datasets) use 1.

Summary

- A semi-parametric GLM has a linear predictor depending linearly on unknown smooth functions of covariates.
- Representing these functions with intermediate rank linear basis expansions recovers an over-parameterized GLM.
- Maximum *penalized* likelihood estimation of this GLM avoids overfit by penalizing function wiggliness.
- It uses penalized iteratively reweighted least squares.
- The degree of penalization is chosen by REML, GCV etc.
- Viewing the fitting penalties as being induced by priors on function wiggliness, provides a justification for PMLE and implies a posterior distribution for the model coefficients which gives good frequentist inference.

Bibliography

- Most of the what has been discussed here is somehow based on the work of Grace Wahba, see, e.g. Wahba (1990) *Spline models of observational data*. SIAM.
- The notion of a GAM and the software interface that is `gam` and associated functions is due to Hastie and Tibshirani, see Hastie and Tibshirani (1990) *Generalized Additive Models* Chapman and Hall.
- Chong Gu developed the first computational method for multiple smoothing parameter selection, in 1992. See e.g. Gu (2002) *Smoothing Spline ANOVA*. Springer.
- Wood (2006) *Generalized Additive Models: An introduction with R*. CRC. provides more detail on the framework given here.