Generalized additive models I

Patrick Breheny

October 6

Introduction

- Thus far, we have discussed nonparametric regression involving a single covariate
- In practice, we often have a p-dimensional vector of covariates for each observation
- The nonparametric multiple regression problem is therefore to estimate

$$\mathrm{E}(y|\mathbf{x}) = f(\mathbf{x})$$

where $f : \mathbb{R}^p \mapsto \mathbb{R}$

Generalized additive models

- In a future lecture, we will discuss the possibility of multidimensional splines and what advantages and drawbacks they have
- For now, we will consider a restricted class of functions; namely, those that have an *additive* form:

$$\mathrm{E}(y|\mathbf{x}) = \alpha + f_1(x_1) + f_2(x_2) + \cdots + f_p(x_p)$$

- By introducing a link function into linear regression, we have generalized linear models (GLMs)
- By introducing a link function into additive models, we have generalized additive models (GAMs):

$$g\{\mathrm{E}(y|\mathbf{x})\} = \alpha + \sum_j f_j(x_j)$$

Backfitting

Fitting generalized additive models is relatively easy from a computational standpoint, as we can employ a simple algorithmic approach called *backfitting* for turning any one-dimensional smoother into a method for fitting additive models:

(1) Initialize: $\hat{\alpha} = \frac{1}{n}\sum_i y_i$, $\hat{f}_j = 0$ for all j

(2) Cycle over j until convergence:

- (a) Compute $\tilde{y}_i = y_i - \hat{\alpha} - \sum_{k \neq j} f_k(x_{ik})$ for all i
- (b) Apply the one-dimensional smoother to $\{x_{ij}, \tilde{y}_i\}$ to obtain $\hat{f}_j$
- (c) Set $\hat{f}_j$ equal to $\hat{f}_j - n^{-1}\sum_i \hat{f}_j(x_{ij})$

Note that we require $\sum_i \hat{f}_j(x_{ij}) = 0$ for all j; otherwise the model is not identifiable

Backfitting (cont'd)

- The modular nature of the backfitting algorithm makes it easy to fit very general models, such as models that mix parametric and nonparametric terms and models with interactions resulting in separate smooth functions for each class
- Computing degrees of freedom is also a simple extension of earlier results: letting $\mathbf{S}_j$ denote the smoother matrix for the jth term, the degrees of freedom of the jth term is $\mathrm{tr}(\mathbf{S}_j) - 1$

PROC GAM

- Smoothing splines, and generalized additive models based on them, are available in SAS via `PROC GAM`
- The basic syntax is as follows:

```
PROC GAM DATA=bmd;
  MODEL Spnbmd = SPLINE(Age);
RUN;
```

- By default, SAS uses 4 degrees of freedom for each spline term; to have λ selected by GCV, specify `METHOD=GCV`:

```
PROC GAM DATA=bmd;
  MODEL Spnbmd = SPLINE(Age) / METHOD=GCV;
RUN;
```

PROC GAM: Further options

- To fit separate splines for males and females, we can use a BY statement:

```
PROC GAM DATA=bmd;
  CLASS Gender;
  BY Gender;
  MODEL Spnbmd = SPLINE(Age) / METHOD=GCV;
RUN;
```

- PROC GAM allows for multiple splines, and for both spline and parametric (linear) terms; if additional continuous variables X1 and X2 were present in the data set, we could submit:

```
PROC GAM DATA=bmd;
  CLASS Gender;
  BY Gender;
  MODEL Spnbmd = PARAM(X1) SPLINE(X2) SPLINE(Age) / METHOD=GCV;
RUN;
```

PROC GAM: Modeling details

- PROC GAM interprets the line MODEL y = SPLINE(x) to mean

$$\mathrm{E}(y|x) = \beta_0 + \beta_1 \mathbf{x} + s(x);$$

  i.e., SAS adds a linear term into the model automatically, and separates $f(x)$ into two parts, the linear and nonlinear components
- This facilitates testing of the nonlinearity of x using likelihood ratio tests

PROC GAM output

PROC GAM provides the following output sections:

- "Regression Model Analysis": Estimates and inferences concerning the parametric portions of the model (NOTE: the linear effects of the spline terms are included here; the test of $\beta_j = 0$ here involves removing $f_j(x_j)$ from the model, though still adjusting for the other covariates in a nonparametric way)
- "Smoothing Model Analysis: Fit Summary for Smoothing Components": The value of λ chosen by GCV (or specified by degrees of freedom)
- "Smoothing Model Analysis: Analysis of Deviance": Likelihood ratio tests of nonlinearity for each spline term

PROC GAM: Visualization

- An important part of fitting GAMs is the ability to visualize the resulting smooth functions
- One can obtain plots of the resulting smooth terms via:

```
ODS GRAPHICS ON;
PROC GAM DATA=bmd PLOTS=COMPONENTS(CLM ADDITIVE);
  CLASS Gender;
  BY Gender;
  MODEL Spnbmd = SPLINE(Age) / METHOD=GCV;
RUN;
ODS GRAPHICS OFF;
```

- The CLM option creates shaded confidence bands, while ADDITIVE requests a plot of $f(x) = \beta x + s(x)$, the combined effect of the linear and nonlinear components of the effect of x

The mgcv package

- The mgcv package in R is similar to PROC GAM in terms of the features it provides, although there are subtle differences in the syntax and the underlying computations
- For example, the implementation in mgcv is based not on backfitting, but rather on something called the *Lanczos algorithm*, a way of efficiently calculating truncated matrix decompositions that is beyond the scope of this course

Syntax: mgcv

- The basic syntax of gam in the mgcv package is:

```
fit <- gam(spnbmd~s(age),data=bmd)
fit <- gam(spnbmd~s(age,by=gender),data=bmd)
```

- One can add arguments to the s() function, but the default behavior is to use a natural cubic spline basis and to automatically choose the smoothing parameter via optimization of the GCV (which the package calls $UBRE$)

mgcv: Modeling details

- The most important difference between PROC GAM and mgcv from a user's perspective is that, by default, mgcv does break up its smooth terms into their linear and nonlinear components
- Thus, by default, mgcv does not give you a parametric test of the linear effect of age or of the nonlinearity of age
- Instead, the summary provided by summary(fit) tests whether age has any effect (linear or otherwise) on the relative change in spinal bone mineral density (separately for males and females)
- Note also that mgcv allows for interactions between parametric and nonparametric terms, unlike PROC GAM (fitting separate models is not the same thing as modeling an interaction)

mgcv: Modeling details

To test hypotheses regarding nonlinearity using mgcv, you can manually set up a restricted model and carry out the likelihood ratio test yourself:

```
fit0 <- gam(spnbmd~gender*age,data=bmd)
anova(fit0,fit,test="F")
```

anova vs. summary

It is important to note that, while anova and summary agree for parametric models, in the nonparametric case they are taking different approaches to testing, and do not produce the same results for the same tests:

```
> fit <- gam(spnbmd~s(age),data=bmd)
> fit0 <- gam(spnbmd~1,data=bmd)
> anova(fit0,fit,test="F")
...
  Resid. Df Resid. Dev    Df Deviance      F    Pr(>F)
2    478.61    0.78961 5.387  0.40303 45.349 < 2.2e-16 ***
...
> summary(fit)
...
          edf Ref.df     F p-value
s(age) 5.387  6.527 36.52  <2e-16 ***
...
```

anova vs. summary

- This is because the hypothesis test in `summary` attempts to adjust for the fact that λ is chosen based on the data rather than fixed; to do so, it computes a modified χ^2 statistic with an inflated number of degrees of freedom (thus reducing the test statistic somewhat)
- In larger models with multiple smooth terms, the tests can also differ because when models are refit, the optimal values of λ_j do not stay the same across fits

Comments on hypothesis testing

- It should be pointed out that these tests are approximate, and should be taken as only a rough guide concerning statistical significance
- This issue is in no way unique to GAMs; any time model selection is performed, the resulting p-value are no longer valid and should be taken as only rough indicators of significance
- Which type of test (`anova` vs. `summary`) is appropriate depends on the situation and the goals of the analysis
- Finally, keep in mind that hypothesis testing is often not the primary goal in regression modeling in the first place, and that building a model that predicts the outcome well (as measured by GCV for instance) is often a more meaningful goal

mgcv: Visualization

```
plot(fit,shade=TRUE)
```