

Stat 8053, Fall 2013: Additive Models

We will only use the package `mgcv` for fitting additive and later generalized additive models. The best reference is S. N. Wood (2006), *Generalized Additive Models, An Introduction with R*, Chapman & Hall. You should be able to get enough from the examples in our textbook, the help page for the `gam` function, and handouts to use the package. If you want to apply these methods to real problems, you will find Wood's book very helpful.

The first example is of fitting an *additive* model, rather than a generalized additive model. Foresters are often interested in estimating the **Volume** of wood available in a forest from simple measurements on a tree, typically the **Girth**, the circumference of the tree at a fixed height above the ground, and the height of the tree, which can be measured, with somewhat less accuracy, using trigonometry. For this example, we have $n = 31$ black cherry trees on which volume has been measured, as shown in Figure 1.

```
library(car)
library(mgcv)
data(trees)
pairs(trees, cex=1.5)
```

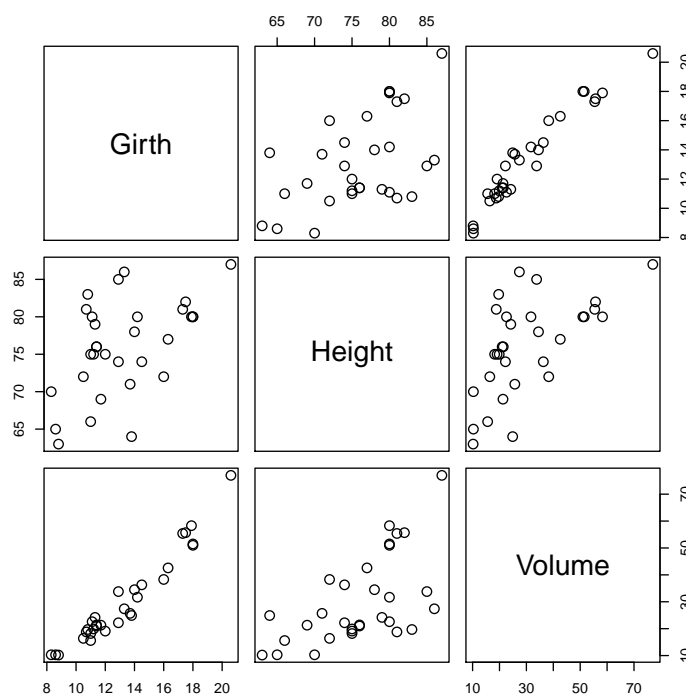


Figure 1: Cherry tree data

A standard first-order linear model fit to these data matches the data surprisingly well, but even so it misses systematic features, as can be seen from the marginal model plot¹ in Figure 2.

¹Marginal Model Plots are discussed in S. Weisberg (2005), *Applied Linear Regression*, 3ed Ed., p.185–

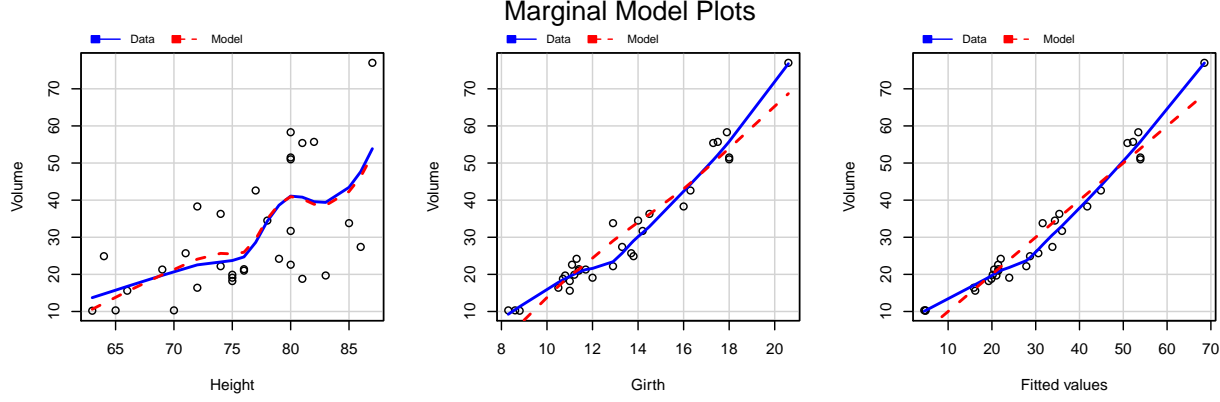


Figure 2: Cherry tree data

```
m0 <- lm(Volume ~ Height + Girth, trees)
mmps(m0, layout=c(1, 3))
```

There is relatively subtle lack-of-fit apparent in both the plot for `Girth` and the plot for fitted values, as the lines don't agree perfectly. It should be no surprise that this model can't match the data perfectly, as simple geometric considerations would suggest that $\text{Volume} \propto \text{Girth}^2 \times \text{Height}$, so fitting with all variables in log-scale would do better². Here, we will instead fit an additive model,

$$E(\text{Volume}|\text{Girth}, \text{Height}) = \beta_0 + s_1(\text{Height}) + s_2(\text{Girth}) \quad (1)$$

The package `mgcv` uses penalized regression splines for the s_j , so we can write

$$s_j(x) = \sum \beta_{jk} \phi_{jk}(x) = \beta_j' \phi_j$$

for the basis functions ϕ_{jk} that determine the splines. Given the basis function representation of the s_j , (1) is now a parametric mean function with parameters $\beta = (\beta_0, \beta_1', \beta_2')'$ and predictors X that define the intercept and the splines that define the s_j .

The penalized least squares objective function for estimating β is

$$\|Y - X\beta\|^2 + \sum_j \lambda_j \beta_j' B_j \beta_j$$

The values of the λ_j are selected in an iterative algorithm to minimize a generalized cross validation criterion.

This fit is done using the `gam` function in the `mgcv` package,

190. It is a plot of the response Y versus a predictor X , or any other quantity. Shown on the graph are two smooths: The *Data* smooth is a smooth of the data shown. The *Model* smooth is obtained by smoothing \hat{Y} versus X , a plot that is not shown. If the model is appropriate for the data, then the two smooths should agree for *any* choice of X , and if there is any X for which they disagree we have evidence of lack-of-fit.

²Another alternative is to replace the predictors by their logarithms, and then fit a GLM with normal errors and log-link.

```
summary(m1 <- gam(Volume ~ s(Height) + s(Girth), data=trees))
```

Family: gaussian

Link function: identity

Formula:

Volume ~ s(Height) + s(Girth)

Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	30.171	0.482	62.6	<2e-16

Approximate significance of smooth terms:

	edf	Ref.df	F	p-value
s(Height)	1.00	1.00	16	0.00042
s(Girth)	2.69	3.37	204	< 2e-16

R-sq.(adj) = 0.973 Deviance explained = 97.7%

GCV score = 8.4734 Scale est. = 7.1905 n = 31

The only visible parameter in this model is the intercept, estimated to be 30.171. The β_j are hidden inside the smoothers and are largely uninterpretable. For each of the predictors, a smoother was fit by the `s` functions. The default in the function `s` that does the smoothing uses *thin plate regression splines*, which are slightly different from the B-splines we discussed in class, but are apparently preferred because they don't depend as much on the number of knots selected and also they generalize to smooths of more than one variable at a time. You can also get the fit using B-splines:

```
summary(m1a <-  
  gam(Volume ~ s(Height, bs="cr") + s(Girth, bs="cr"), data=trees))
```

Family: gaussian

Link function: identity

Formula:

Volume ~ s(Height, bs = "cr") + s(Girth, bs = "cr")

Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	30.171	0.482	62.6	<2e-16

Approximate significance of smooth terms:

	edf	Ref.df	F	p-value
s(Height)	1.00	1.00	16	0.00043
s(Girth)	2.67	3.33	206	< 2e-16

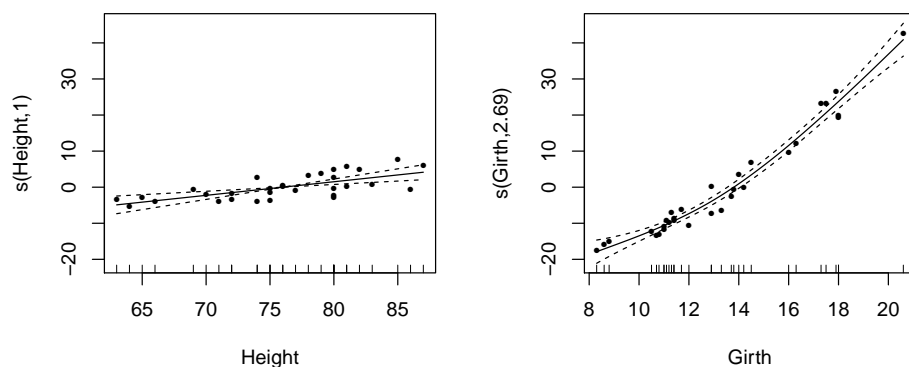
```
R-sq.(adj) = 0.973    Deviance explained = 97.7%
GCV score = 8.4847    Scale est. = 7.2065      n = 31
```

which is nearly identical to the fit with thin plate splines. You can look at the help pages for `s` and `smooth.terms` for additional options. Additional hint: The number of basis functions is set by the argument `k`, and `s` has an algorithm for selecting it. In some instances you will need to make `k` either smaller or larger to get convergence.

In the output, the “edf” is the equivalent degrees of freedom for each of the smooths. If we were using q basis functions for a smooth and ml estimation, then we would have q df for the smooth. Penalization will generally reduce the edf to a number smaller than q . For `Height` the smoother has only 1 edf, essentially fitting a straight line. The approximate significance tests for the smooths are apparently Wald tests of $\beta_j = 0$ with approximate df for the F distributions. The adjusted R^2 is more or less the square of the correlation between the observed and fitted values, with an adjustment for degrees of freedom, while the deviance explained appears to be the usual R^2 written as a percentage. The scale estimate is $\hat{\sigma}$, assumed constant throughout the data.

The usual summary of the fit of a gam is a set of pictures:

```
plot(m1, residuals=TRUE, pages=1, pch=16, cex=0.6)
```



The `plot` method with no arguments beyond the name of the fitted object returns the plots of x_j versus the estimate $s_j(x_j)$ for all j . The argument `residuals=TRUE` adds the *partial residuals*, equal to the $\hat{e}_i + s_j(x_{ij})$. By default, one graph is drawn at a time, with required user input to see the next plot. You can override this with the argument `page=1`, which will put all the plots on one page. There are many more arguments; type `?plot.gam` to see them all.

You can perform likelihood ratio type tests to compare models. For example:

```
m2 <-update(m1, ~ . - s(Girth))
anova(m2, m1, test="F")
```

Analysis of Deviance Table

Model 1: Volume ~ s(Height)

```
Model 2: Volume ~ s(Height) + s(Girth)
      Resid. Df Resid. Dev   Df Deviance    F Pr(>F)
1         29.0         5205
2         26.3          189 2.69      5016 259 <2e-16
```

Or, you can replace a smooth by a parametric fit:

```
summary(m3 <- update(m1, ~ . - s(Height) + Height))
```

```
Family: gaussian
Link function: identity
```

```
Formula:
Volume ~ s(Girth) + Height
```

```
Parametric coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.5256     7.1765    0.21  0.83329
Height        0.3769     0.0942    4.00  0.00046
```

```
Approximate significance of smooth terms:
```

```
      edf Ref.df    F p-value
s(Girth) 2.69   3.37 207 <2e-16
```

```
R-sq.(adj) = 0.973   Deviance explained = 97.7%
GCV score = 8.4734   Scale est. = 7.1905     n = 31
```

```
AIC(m2, m3, m1)
```

```
      df    AIC
m2 3.000 252.8
m3 5.693 155.4
m1 5.693 155.4
```

Methods available for **gam** objects include **predict**, **residuals** and **plot**. For example, **?predict.gam** will give you help on how to use this function. Just about any function that works with a **glm** object is likely to work with a **gam** object as well. An exception is **effects** plots, but the effects plots for the smoothed terms will have the same shape, but different values on the vertical axis, as the plots produced by the **mgcv plot** method.

Although probably not needed here, we could also fit with a bivariate smoother. Here I had to specify the number of basis functions to be used by the thin plate splines because the default was apparently too small.

```
par(mfrow=c(1, 2))
summary(m4 <- update(m1, ~ s(Height, Girth, k=8)))
```

Family: gaussian
Link function: identity

Formula:
Volume ~ s(Height, Girth, k = 8)

Parametric coefficients:
 Estimate Std. Error t value Pr(>|t|)
(Intercept) 30.171 0.516 58.4 <2e-16

Approximate significance of smooth terms:
 edf Ref.df F p-value
s(Height,Girth) 5.71 6.57 145 <2e-16

R-sq.(adj) = 0.969 Deviance explained = 97.5%
GCV score = 10.548 Scale est. = 8.2665 n = 31

AIC(m1, m4)

	df	AIC
m1	5.693	155.4
m4	7.706	161.3

```
plot(m4)
plot(predict(m1), predict(m4), xlab="Predictions, additive",
      ylab="Predictions, nonadditive")
abline(0, 1)
```

