Zuur Ch 03 (Additive modelling)

HARUG! QRantine edition

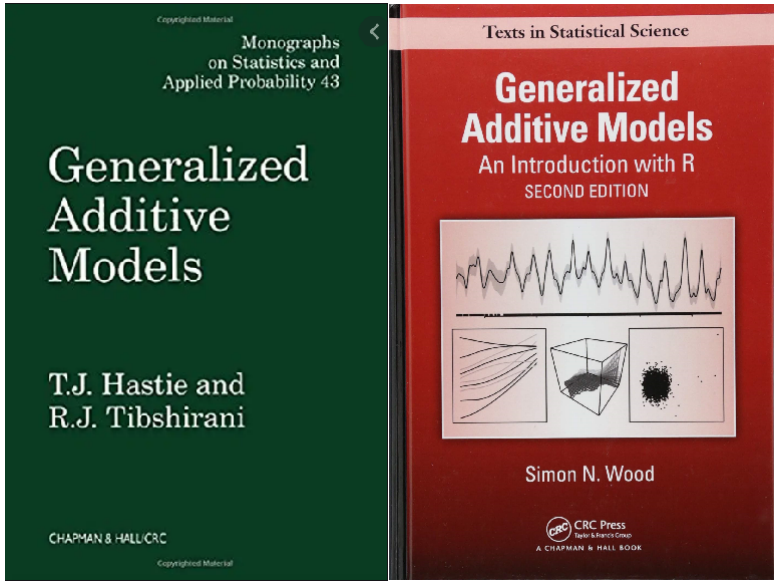
Ed Harris

2020.04.22

What the heck is a GAM?

Basically, non-linear regression

The classics:



Ch 03 outline

-3.1 Intro to GAM

-3.2 Additive modeling description

-3.3 Some GAM tech details

-3.4 and 3.5 some GAM examples

-3.6 Inference…

-3.7 Summary

3.1 Intro to GAM

A general model for a LINEAR model

$$Y_i = \alpha + \beta_1 \times X_{1i} \times Z_i + \epsilon_i$$

Where, $\epsilon_i \sim N(0, \sigma^2)$

Z_i is any term that maintains linear framework (like a transformation term, etc.)

Essentially, if you CANNOT coerce to linear, use GAM

NB references and dates (~1990 - 2008…)

3.2 Additive modelling description

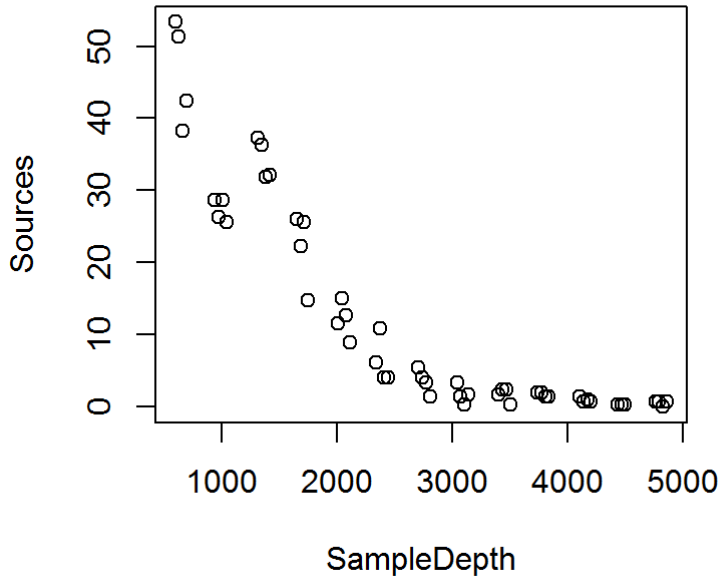
Pelagic bioluminescence data along a depth gradient: ISIT.txt

```
data <- read.table("ISIT.txt", header = T)
head(data[, 1:6])
```

```
##   SampleDepth Sources Station Time Latitude Longitude
## 1         517   28.73       1    3  50.1508  -14.4792
## 2         582   27.90       1    3  50.1508  -14.4792
## 3         547   23.44       1    3  50.1508  -14.4792
## 4         614   18.33       1    3  50.1508  -14.4792
## 5        1068   12.38       1    3  50.1508  -14.4792
## 6        1005   11.23       1    3  50.1508  -14.4792
```

3.2 Additive modelling description

Non-linear, wandering variance, this is just 1 station of many



Try GAM…

3.2 Additive modelling description

In R there are many ways to analyse GAMs:

Package {gam} - Hastie and Tibshirani (1990)
-simple, widely known

Package {mgcv} - Wood 2ed (2017)
-more modern, can do mixed effects

3.2.2 GAM in gam with LOESS

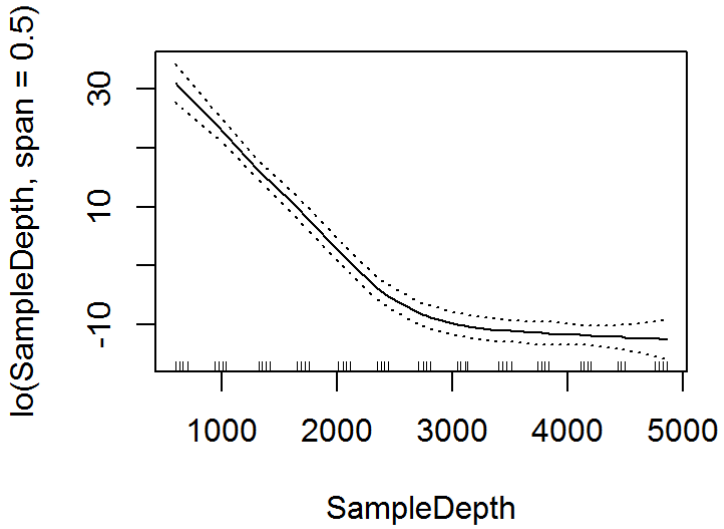
$$Y_i = \alpha + f(X_i) + \epsilon_i$$

Where, $\epsilon_i \sim N(0, \sigma^2)$

Here, the $f()$ refers to a function to describe the curve

3.2.2 GAM in gam with LOESS

```
library(gam)
M1 <-gam(Sources ~ lo(SampleDepth, span = 0.5),
         data= data[data$Station == 16,])
plot(M1, se = TRUE) #Fig.3.1B
```



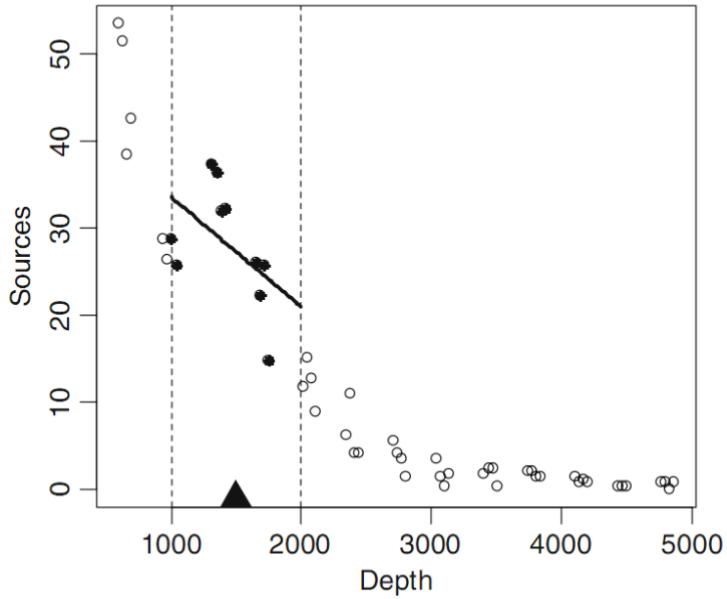
3.2.2.1 LOESS Smoothing

LOcally Estimated Scatterplot Smoothing

-weights estimates of y,by some x variable at intervals of each x

3.2.2.1 LOESS Smoothing

-LOESS can be achieved in a few ways
-using lo() we apply linear regression in a sliding window



3.2.2.1 LOESS Smoothing

-the span argument in lo(SampleDepth, span = 0.5) determines the size of the window

-you can overfit and underfit…

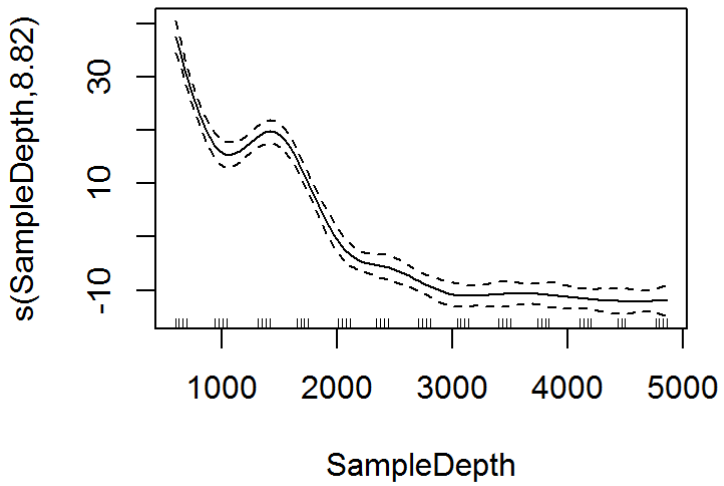
3.2.2.1 LOESS Smoothing

3.2.2.1 LOESS Smoothing

In practice we find LOESS window size using a few methods
-trial and error, graph inspection
-residual fit graphs
-AIC methods…

GAM in mgcv, Cubic Regressions Splines

```
library(mgcv)
M2 <-gam(Sources ~ s(SampleDepth, fx = F, k=-1, bs="cr"),
         data= data[data$Station == 16,])
plot(M2, se = TRUE) #Fig.3.5B
```



GAM in mgcv, Cubic Regressions Splines

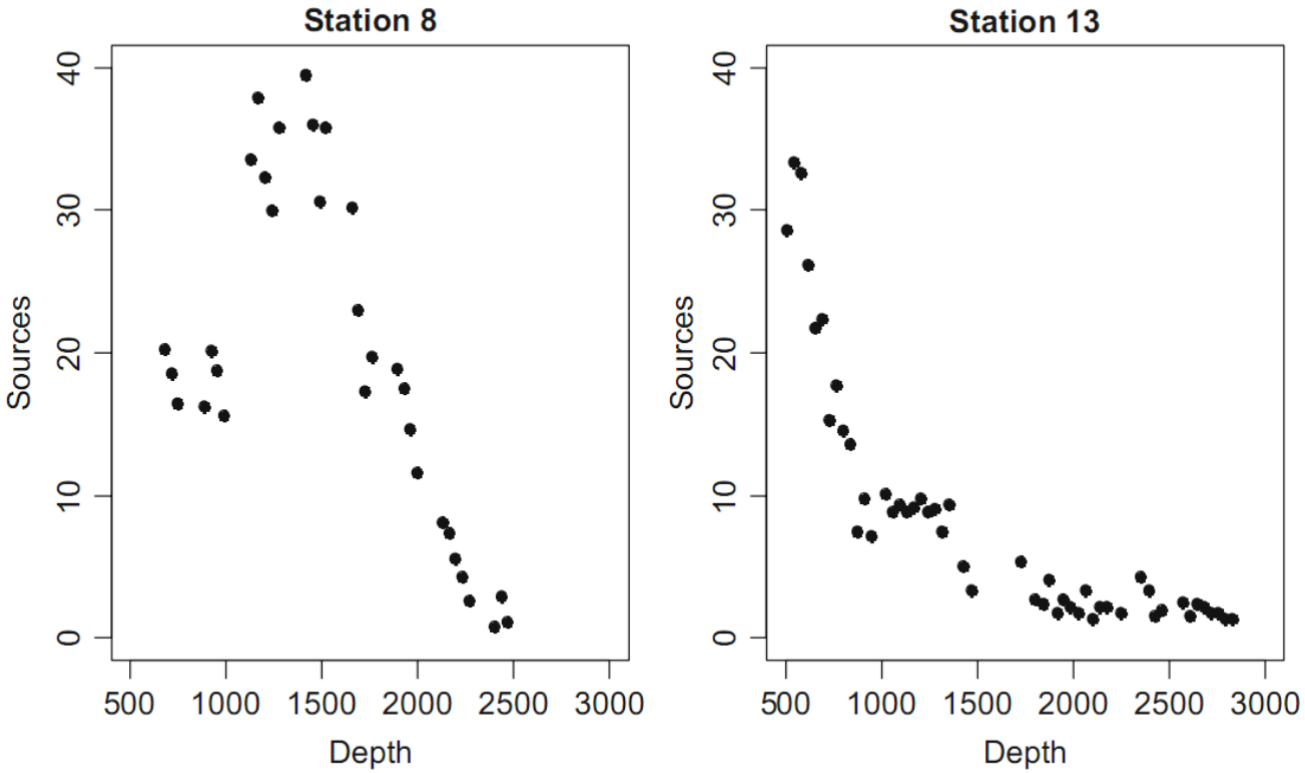
M2 <-gam(Sources ~ s(SampleDepth, fx = F, k = -1, bs = “cr”), data = data[data$Station == 16,])

fx = F, no fixed function for amount of smoothing
k = -1, dimension of smoothing term (-1 mean no limits)
bs = “cr”, basis of smoothing (cr mean cubic regression)

3.3 Some GAM tech details

I skipped this part in the interest of time. If you want to know more (and are actually serious about it), I highly recommend investing some time into Simon Wood's book.

3.4 Biolum. Data for 2 Stations

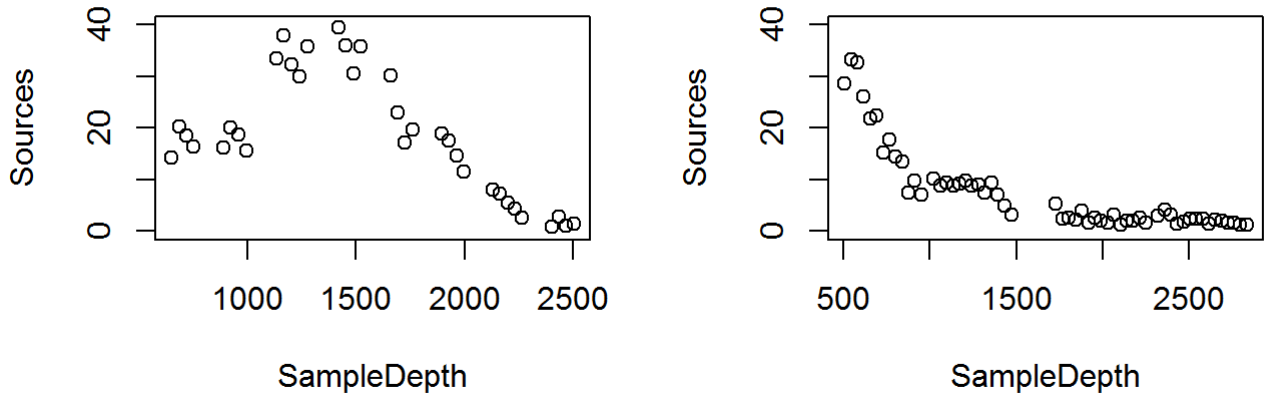


3.4 Biolum. Data for 2 Stations

```
> library(AED); data(ISIT)
> S8  <- ISIT$Sources[ISIT$Station == 8]
> D8  <- ISIT$SampleDepth[ISIT$Station == 8]
> S13 <- ISIT$Sources[ISIT$Station == 13]
> D13 <- ISIT$SampleDepth[ISIT$Station == 13]
> So <- c(S8, S13); De <- c(D8, D13)
> ID <- rep(c(8, 13), c(length(S8), length(S13)))
> mi <- max(min(D8), min(D13))
> ma <- min(max(D8), max(D13))
> I1 <- De > mi & De < ma
> op <- par(mfrow = c(1, 2))
> plot(D8[I1], S8[I1], pch = 16, xlab = "Depth",
     ylab = "Sources", col = 1, main = "Station 8",
     xlim = c(500, 3000), ylim = c(0, 40))
> plot(D13[I1], S13[I1], pch = 16, xlab = "Depth",
     ylab = "Sources", col = 1, main = "Station 13",
     xlim = c(500, 3000), ylim = c(0, 40))
> par(op)
```

3.4 Biolum. Data for 2 Stations

```
x8 <- which(data$Station ==  8)
x13 <- which(data$Station ==  13)
par(mfrow = c(1,2))
plot(Sources ~ SampleDepth, data = data[x8,],
     ylim=c(0,40))
plot(Sources ~ SampleDepth, data = data[x13,],
     ylim=c(0,40))
```



3.4 Biolum. Data for 2 Stations

```
library(mgcv)
x <- c(x8,x13)
mi <-max(min(data$SampleDepth[x8]),
         min(data$SampleDepth[x13]))
ma <-min(max(data$SampleDepth[x8]),
         max(data$SampleDepth[x13]))
I1 <- data$SampleDepth[x] > mi & data$SampleDepth[x] < ma

M4 <-gam(Sources ~ s(SampleDepth) + factor(Station),
         data = data[x,], subset=I1)
```

3.4 Biolum. Data for 2 Stations

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Sources ~ s(SampleDepth) + factor(Station)
##
## Parametric coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         19.198      1.054  18.207  < 2e-16 ***
## factor(Station)13  -12.296      1.397  -8.801 7.59e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                   edf Ref.df     F  p-value
## s(SampleDepth) 4.849  5.904 14.77 7.08e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3.4 Biolum. Data for 2 Stations

```
anova(M4)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Sources ~ s(SampleDepth) + factor(Station)
##
## Parametric Terms:
##                  df     F  p-value
## factor(Station)  1 77.46 7.59e-13
##
## Approximate significance of smooth terms:
##                   edf Ref.df     F  p-value
## s(SampleDepth) 4.849  5.904 14.77 7.08e-12
```

3.6 Inference…

3.7 Summary