Generalized Additive Models (GAMs)

CMDA 4654 Project 2

Group 19: Brady Bolton, Eryk Jesse, Charles Lee, Dan Schlicht

Generalized Additive Models and why we use them

- Type of generalized linear model
- Response variable depends on smooth functions $f_i(x_i)$
- General structure of a GAM:

$$g(\mathrm{E}(Y)) = \beta_0 + f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n)$$

- Smooth functions can be many different things (polynomials, splines, weighted means, etc)

Summations

A Linear Model sums the linear terms

$$y_i = \beta_0 + \sum_j \beta_j x_{ji} + \epsilon_i$$

GAMs sums the *smooth functions*

$$y_i = \beta_0 + \sum_j s_j(x_{ji}) + \epsilon_i$$

Where

$$\epsilon_i \sim N(0, \sigma^2), y_i \sim \text{Normal}$$

Splines

A *Spline* is a function made of up basis functions (the smoothing functions)

These simpler functions form a set of functions called the *basis*

When using a spline for GAMs, each basis function has a coefficient

The spline is formed by weighing the basis function coefficients and summing them at each value of x

Wiggliness and Penalized fit

As in the same case with a polynomial regression of excess “wiggles” not constraining the “Wiggliness” in the way that we penalize the fit in order to prevent overfitting

W or wiggliness is defined by:

$$\int_{\mathbb{R}} [f'']^2 dx = \boldsymbol{\beta}^{\mathsf{T}} \mathbf{S} \boldsymbol{\beta} = W$$

Constraining Wiggliness

We have to make wiggliness important by looking into the log-likelihood, or the measure of closeness to the data

The term **smoothing operator** λ defines the trade-off to find *spline* coefficients to maximize the penalized log-likelihood fit

$$\mathcal{L}_p(\beta) = \mathcal{L}(\beta) - \frac{1}{2}\lambda\beta^\mathsf{T}\mathbf{S}\beta$$

or

$$\mathcal{L}_p = \log(\text{Likelihood}) - \lambda W$$

Selecting smooth

There are multiple methods to choose from the right amount of wiggle, some are: AIC, Mallow $\mathcal{C}_p$, Maximum Likelihood(ML), and Restricted Maximum Likelihood(REML). The most commonly method is REML for it's numerical stability

There are two ways to optimize the given λ:

- Predictive: Reducing generalization error
- Bayesian: Using priors for basis coefficients

Maximizing Wiggliness

In a regular regression, the degree of freedom typically equal the predictors in the model. In the case for GAMs, we look at the smoothing *basis* of size k and consider that with **penalized** fitting, their parameters are limited. Thus, the models ***effective*** degrees of freedom (EDF) will not equal the size k

The models effective degrees of freedom are given by trace(F) where F is the EDF matrix

$$F = (X^T WX + \sum_j \lambda_j S_j)^{-1} X^T WX$$

Generalized Additive Models in R

- Two options for packages, mgcv and gam
- mgcv is more commonly used and better supported

Smooth interactions in R

1. Bivariate smoothing

   - s(x, z, bs = ‘tp’)

2. Tensor products smoothing

   - te(x, z)

gam.check()

- A good way to check how well GAM model fits
- Usually Gaussian by default
- Creates 4 plots
  - Q-Q Plot of Residuals
  - Histogram of Residuals
  - Residual vs Linear Predictor Plot
  - Observed vs. fitted values

Simulated Data (Baby Example 1)

```r
library(mgcv)
set.seed(0)
sim_data <- gamSim(1, n = 400, dist="normal", scale=2)
```

```
Gu & Wahba 4 term additive model
```

```r
head(sim_data)
```

```
          y        x0        x1         x2         x3         f        f0
1  5.114211 0.8966972 0.1478457 0.34826473 0.04572472  7.962274 0.6377368
2  2.175828 0.2655087 0.6588776 0.85868745 0.36652658  5.514517 1.4814113
3  6.334878 0.3721239 0.1850700 0.03443876 0.74139303  3.576406 1.8407682
4  6.853276 0.5728534 0.9543781 0.97099715 0.93350625  8.692625 1.9478442
5  7.743879 0.9082078 0.8978485 0.74511014 0.67320995  8.752859 0.5687870
6 13.920886 0.2016819 0.9436971 0.27325524 0.70135711 16.190349 1.1841037
        f1           f2 f3
1 1.344055 5.980482e+00  0
2 3.735028 2.980780e-01  0
3 1.447937 2.877006e-01  0
4 6.744695 8.611364e-05  0
5 6.023672 2.160400e+00  0
6 6.602142 8.404104e+00  0
```

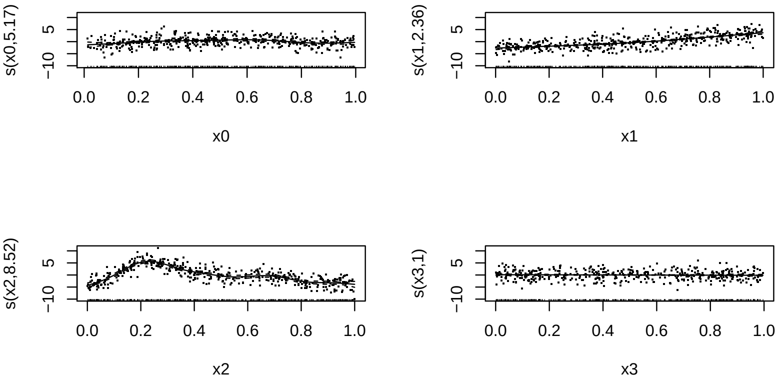
Simulated Data using gam()

```r
fit <- gam(y ~ s(x0) + s(x1) + s(x2) + s(x3), data = sim_data)
summary(fit)
```

```
Family: gaussian
Link function: identity

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

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.9150     0.1049   75.44   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
        edf Ref.df       F  p-value
s(x0) 5.173  6.287   4.564 0.000134 ***
s(x1) 2.357  2.927 103.053  < 2e-16 ***
s(x2) 8.517  8.931  84.308  < 2e-16 ***
s(x3) 1.000  1.000   0.441 0.506929
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.726   Deviance explained = 73.7%
GCV =  4.611  Scale est. = 4.4029    n = 400
```

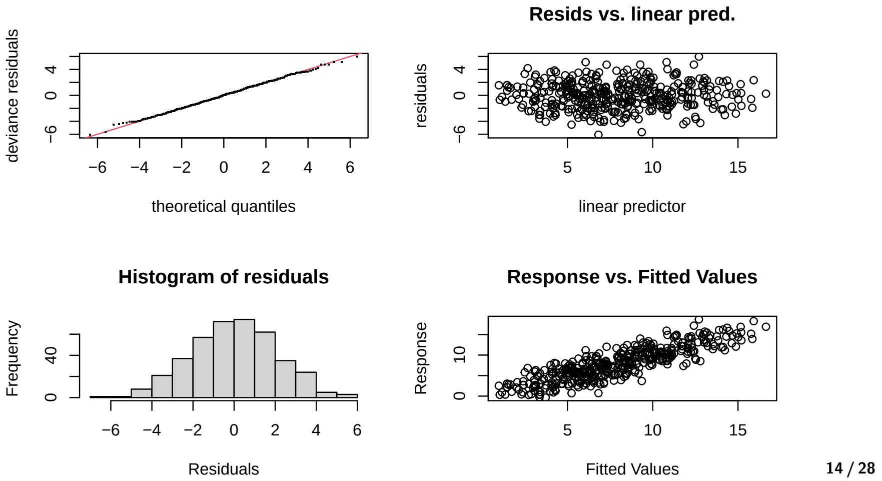
Simulated Data using plot() of GAM fit

```r
plot(fit, pages=1, residuals=TRUE)
```



Simulated Data using gam.check()

```r
gam.check(fit)
```



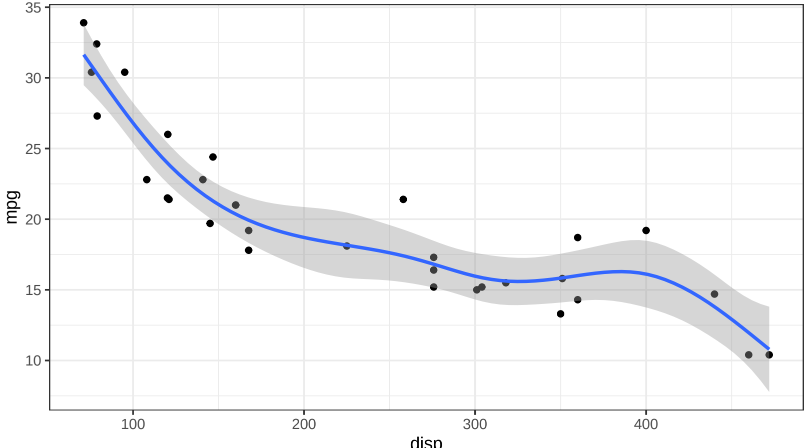
Mtcars Data (Baby Example 2)

```r
data("mtcars")
mtcars_gam <-
  gam(mpg ~ s(disp), data = mtcars, method = "REML")
summary(mtcars_gam)
```

```
Family: gaussian
Link function: identity

Formula:
mpg ~ s(disp)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  20.0906     0.3788   53.04   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
          edf Ref.df    F p-value
s(disp) 4.884  5.904 36.3  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.874   Deviance explained = 89.4%
-REML = 74.101  Scale est. = 4.5918    n = 32
```

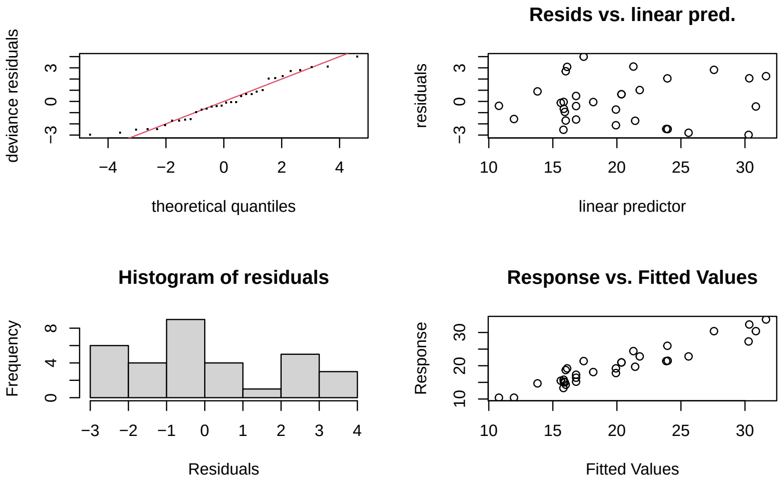
Mtcars Data using GAM on ggplot

```r
library(ggplot2)
ggplot(data = mtcars, aes(x = disp, y = mpg)) +
  theme_bw() + geom_point() +
  geom_smooth(method = "gam", formula = y ~ s(x))
```



Mtcars Data using gam.check()

```
gam.check(mtcars_gam)
```



Health Data - Explaining Data Set

We will be demonstrating GAMs on heart-rate data from an Ironman Triathlete sleeping. The original study also examined the heart-rates of subjects practicing Yogi and Chi meditation techniques (taken from a paper titled "Heart Rate Oscillations during Meditation"). More info including the source material can be found here:

https://physionet.org/content/meditation/1.0.0/

We will focus on a small snippet of the non-linear material.

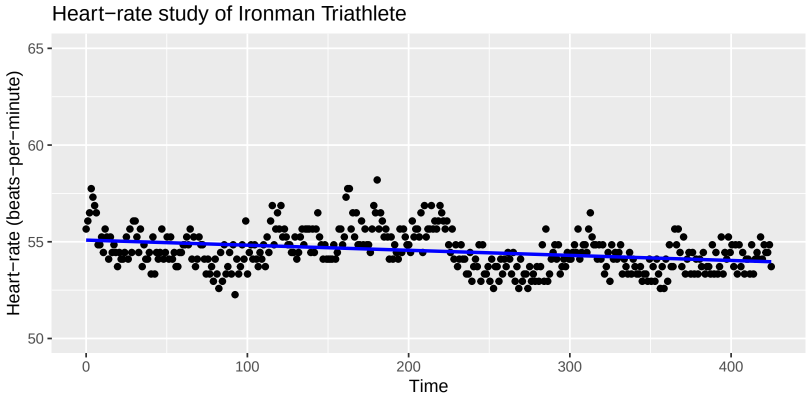
Health Data - Data Cleaning

```r
# Read dataset
i_df <- read.csv("./i9.csv", header = FALSE)
colnames(i_df) <- c("time", "hr")
# Re-index the time
i_df$time <- i_df$time - 42301.30
i_df <- i_df[ !(i_df$time > 425),]
# Remove outliers
i_df <- i_df[ !(i_df$hr > 80),]

# Create scatter-plot
p <- ggplot(i_df, aes(x=time, y=hr)) +
  geom_point() +
  labs(title = 'Heart-rate study of Ironman Triathlete',
       x = "Time",
       y = "Heart-rate (beats-per-minute)") +
  ylim(50, 65)

head(i_df)
```

```
    time      hr
1 -0.003 55.6586
2  1.067 56.0748
3  2.130 56.4972
4  3.169 57.7478
5  4.216 57.3066
6  5.270 56.8720
```

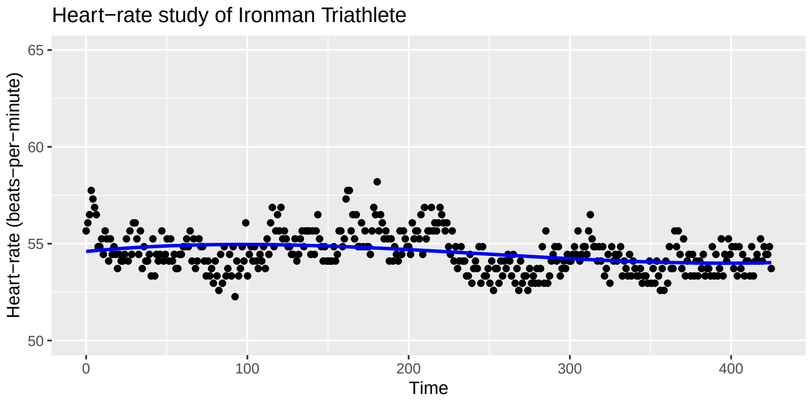
Health Data - LM

```r
# Linear Model Fit (uses formula = y~x)
p + geom_smooth(method = 'lm', se=FALSE, color = 'blue')
```



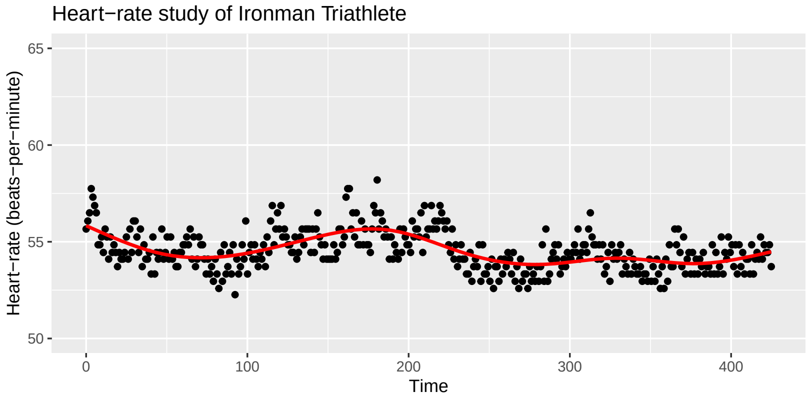
Health Data - Splines

```r
# Model Fit adjusting splines
p + geom_smooth(method = 'lm', formula = y~splines::bs(x,3), se=FALSE, color = 'blue')
```



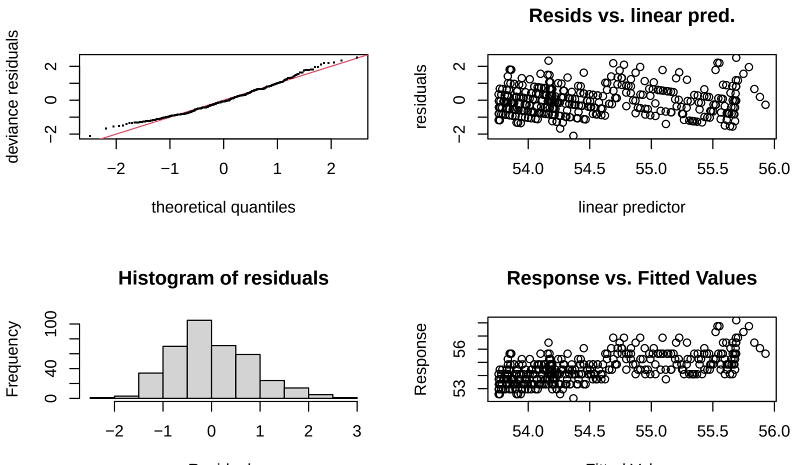
Health Data - GAM

```r
# GAM Model Fit (formula = y ~ s(x, bs = "cs"))
p + geom_smooth(method = 'gam', se=FALSE, color = 'red')
```



Health Data - GAM diagnostics

```r
gam_fit <- gam(hr ~ s(time), data = i_df, method = "REML")
gam.check(gam_fit)
```



LOESS vs. GAM

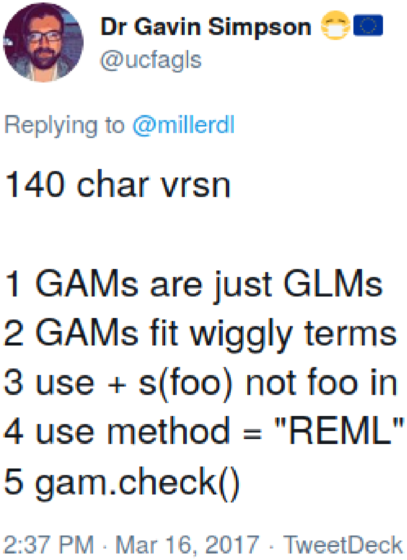
While watching a presentation on GAMs and how they with non-linear data, you might have asked yourself: why use GAMs? Why not just use LOESS (or LOWESS)? Some differences to consider:

- Memory efficiency: LOESS uses O(nˆ2) memory
- Runtime efficiency: GAM can be slower
- Both are used by default in geom_smooth
    - Uses LOESS if $n < 1000$
    - Uses GAM otherwise

Conclusion

- The benefits of implementing GAMs provides a flexible framework to accurately model nonlinear relationships.
- It's formed from basis functions, which weigh regression functions to form larger functions known as smooths
- To control overfitting, we penalize the fit of the model by adjusting the goodness of fit

Fun Reminder



Dr Gavin Simpson 😷 🇪🇺
@ucfagls

Replying to @millerdl

140 char vrsn

1 GAMs are just GLMs
2 GAMs fit wiggly terms
3 use + s(foo) not foo in frmla
4 use method = "REML"
5 gam.check()

2:37 PM · Mar 16, 2017 · TweetDeck

References

- https://m-clark.github.io/generalized-additive-models/introduction.html - Clark GAMs Tutorial
- https://fromthebottomoftheheap.net/slides/gam-intro-webinar-2020/gam-intro.html#1 - Simpson Intro to GAMs
- https://www.mrc-bsu.cam.ac.uk/wp-content/uploads/GAM_slides1.pdf - MRC BioStatistics AM & GAMs
- anson.ucdavis.edu/~jihao/handout5_w15.html - Difference between mgcv and gam packages
- https://physionet.org/content/meditation/1.0.0/ - Data used for our Health Example
- https://multithreaded.stitchfix.com/blog/2015/07/30/gam/ - GAM: The Predictive Modeling Silver Bullet by Kim Larson
- http://environmentalcomputing.net/intro-to-gams/ - Generalised additive models (GAMs): an introduction by Environmental Computing

Further Reading

GAM itself can be a very wide and deep topic. The following resources might also be of interest to the reader:

- *Generalized Additive Models: An Introduction with R* (Simon N. Wood)
- *Generalized Additive Models* (T.J. Hastie, R.J. Tibshirani)
- *An Introduction to Statistical Learning: With Applications in R* (Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani)