Additive models

Zihan Zhang

2020/11/1

Table of contents

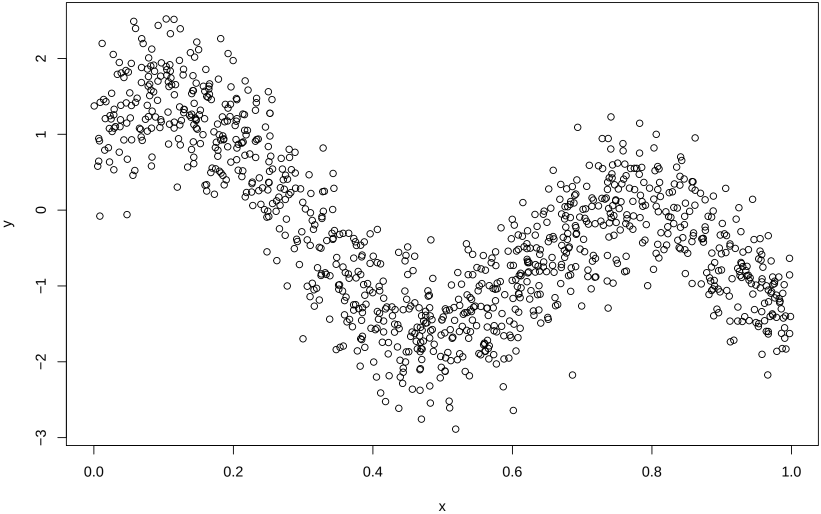
- Regression splines
- (Generalized) additive models
- Examples/ applications

Estimation and prediction

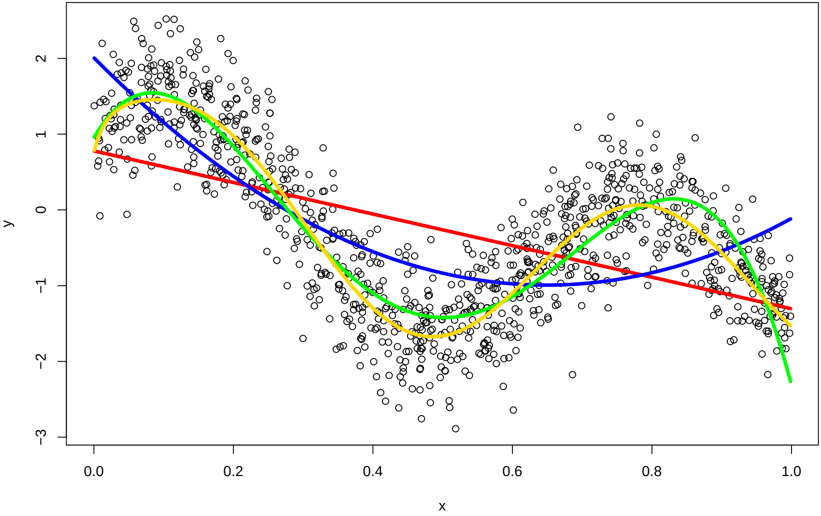
Consider modeling the relationship between y and a single x.

$$y = f(x)$$

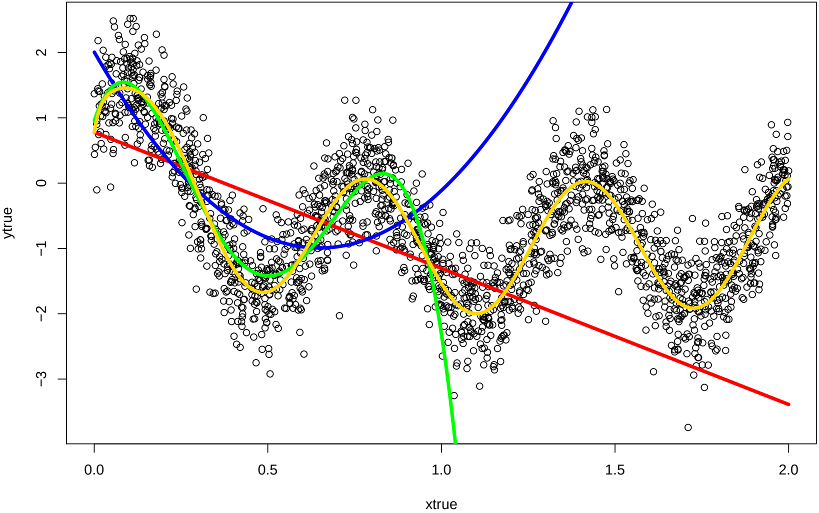
Estimation and prediction



Estimation and prediction



True data



Piecewise regression

- **Piecewise regression** breaks the input space into distinct regions and fit a different relationship in each region.

$$y = \sum_{m}^{M} \beta_m \phi_m(x) + \epsilon$$

- $\phi(x)$ is called the **basis function**.

Piecewise constant regression

Model:

$$y = \sum_m^M \beta_m \phi_m(x) + \epsilon$$

, where

$$\phi_m(x) = \begin{cases} \phi_1(x) = \mathcal{I}\left(x < C_1\right) \\ \phi_m(x) = \mathcal{I}\left(C_{m-1} \leq x < C_m\right) \\ \phi_M(x) = \mathcal{I}\left(C_{M-1} \leq x\right) \end{cases}$$

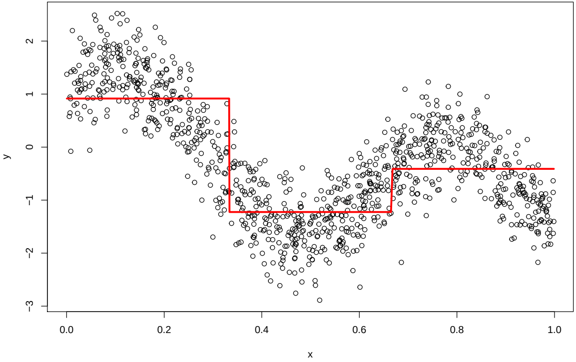
, where C_m is called the **knot** point.

Thus, in this case

$$\beta_m = \bar{y_m}$$

Piecewise constant regression

```
## cut(x, 3)(-0.000504,0.333]      cut(x, 3)(0.333,0.666]
##                   0.9162606                  -1.2245036
##          cut(x, 3)(0.666,1]
##                  -0.4105306
```



Piecewise polynomial regression

However, the basis function $\phi(x)$ is not limited to $\mathcal{I}\left(C_{m-1} \leq x < C_m\right)$. It can be polynomial of x.

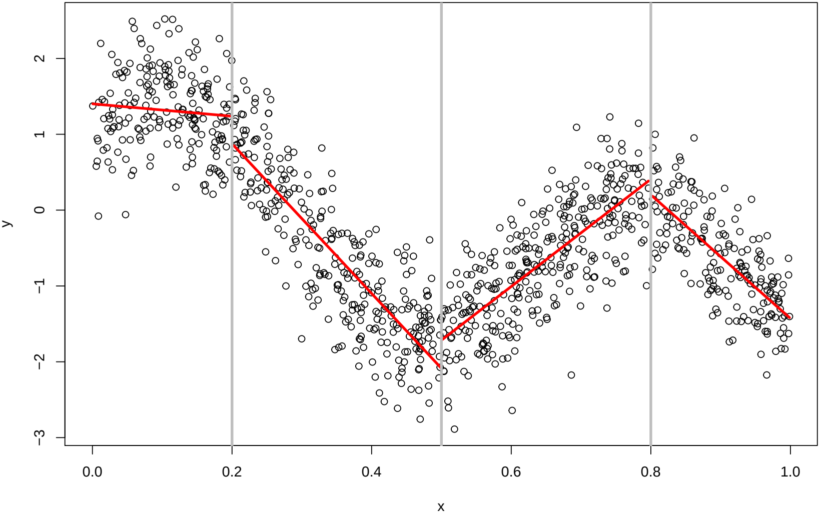
Piecewise polynomial regression

E.g. Piecewise linear regression.

$$y = \beta_0 + \sum_{m=1}^{4} \beta_m \phi_m(x) \mathcal{I}\left(C_{m-1} \leq x < C_m\right)$$

Assuming $C_0 = 0, C_4 = 1$

Piecewise linear regression



Dealing with discontinuity

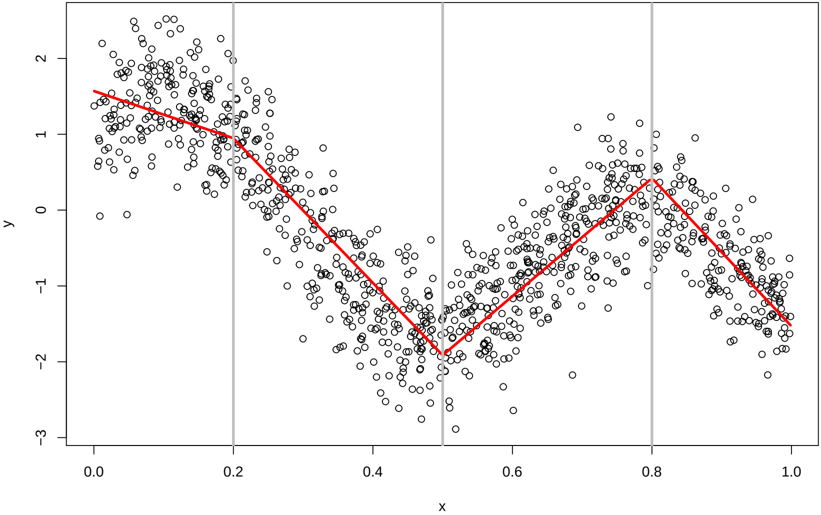
- The regression functions are discontinuous at knots.
- Modify the function form $\Rightarrow$ continuous at knots.

$$y = \begin{cases} \alpha_{10} + \alpha_{11}x + \epsilon & x < C_1 \\ \alpha_{20} + \alpha_{21}(x - C_1) + \epsilon & C_1 \leq x \leq C_2 \\ \alpha_{30} + \alpha_{31}(x - C_2) + \epsilon & C_2 \leq x \leq C_3 \\ \alpha_{40} + \alpha_{41}(x - C_3) + \epsilon & x > C_3 \end{cases}$$

under the constraints that

$$\begin{aligned} \alpha_{10} + \alpha_{11}C_1 &= \alpha_{20} \\ \alpha_{20} + \alpha_{21}C_1 &= \alpha_{30} \\ \alpha_{30} + \alpha_{31}C_1 &= \alpha_{40} \end{aligned}$$

Piecewise linear regression



Piecewise polynomial regression

- The model can be more flexible $\Rightarrow$ Basis function can be polynonial.
- For higher order polynomial, we want something more than `Continuity`, $\Rightarrow$ that is `Smoothness`.
- How it works? $\Rightarrow$ By requiring the derivatives of the piecewise polynomials to be continuous at the knots.
- This is called **Regression Spline**.

Regression spline

A degree- d spline is a piecewise degree-d polynomial, with continuity in derivatives up to degree $d-1$ at each knot.

Regression spline

For example, a degree - 3 spline (piecewise cubic) with only 1 knot at $x = C$:

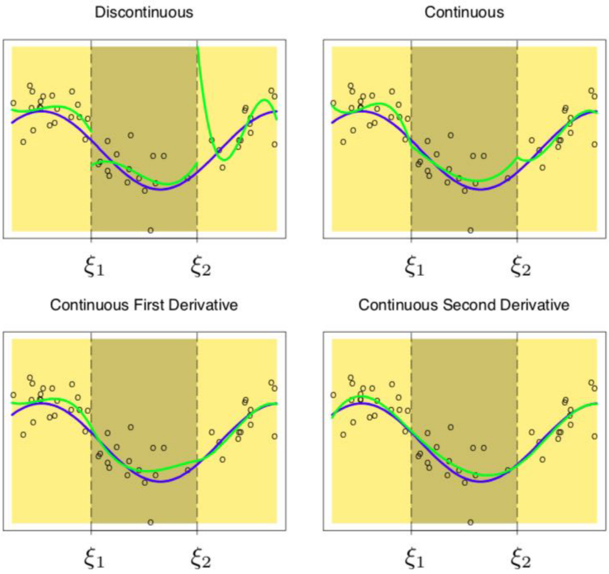
$$y = \begin{cases} \alpha_{10} + \alpha_{11}x + \alpha_{12}x^2 + \alpha_{13}x^3 + e & x < C \\ \alpha_{20} + \alpha_{21}(x - C) + \alpha_{22}(x - C)^2 + \alpha_{23}(x - C)^3 + e & x \geq C \end{cases}$$

under the constriants:

$$\begin{aligned} \alpha_{10} + \alpha_{11}C + \alpha_{12}C^2 + \alpha_{13}C^3 &= \alpha_{20} \\ \alpha_{11} + 2\alpha_{12}C + 3\alpha_{13}C^2 &= \alpha_{21} \\ \alpha_{12} + 3\alpha_{13}C &= \alpha_{22} \end{aligned}$$

Polynomial, 1st and 2nd derivatives are continuous at knot.

Regression spline



B-spline basis function

- Defination: B-spline basis function of degree p

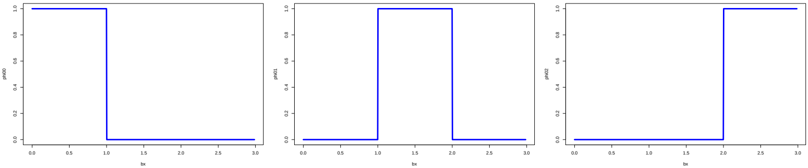
$$\phi_{m,0}(x) = \begin{cases} 1 & \text{if } C_m \leq x < C_{m+1} \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_{m,p}(x) = \frac{x - C_m}{C_{m+p} - C_m}\phi_{m,p-1}(x) + \frac{C_{m+p+1} - x}{C_{m+p+1} - C_{m+1}}\phi_{m+1,p-1}(x)$$

B-spline basis function

- Suppose $x \in [0, 3]$, and we have 2 knots, $C_1 = 1$ and $C_2 = 2$. Meanwhile, we let the boundary $C_0 = 0$ and $C_3 = 3$.
- degree 0 B- spline basis function:

$$\phi_{m,0}(x) = \begin{cases} 1 & \text{if } C_m \leq x < C_{m+1} \\ 0 & \text{otherwise} \end{cases}$$



B-spline basis function

- degree 1 B-spline basis function

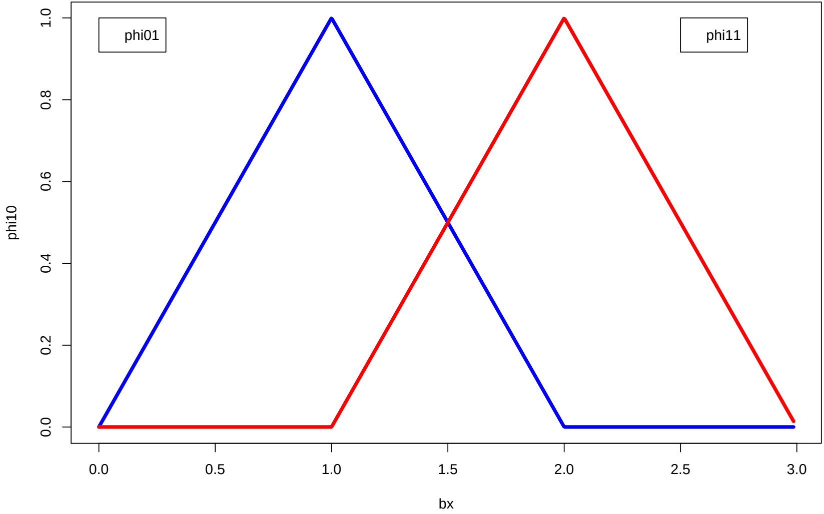
$$\phi_{m,1}(x) = \frac{x - C_m}{C_{m+1} - C_m}\phi_{m,0}(x) + \frac{C_{m+2} - x}{C_{m+2} - C_{m+1}}\phi_{m+1,0}(x)$$

- For example,

$$\begin{aligned}\phi_{0,1}(x) &= \frac{x - C_0}{C_1 - C_0}\phi_{0,0}(x) + \frac{C_2 - x}{C_2 - C_1}\phi_{1,0}(x) \\ &= x\phi_{0,0}(x) + (2 - x)\phi_{1,0}(x)\end{aligned}$$

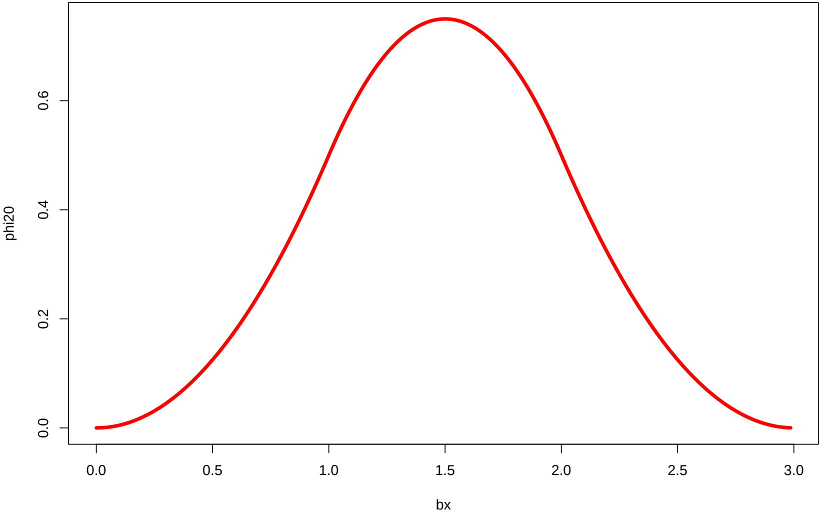
B-spline basis function

- degree 1 B-spline basis function

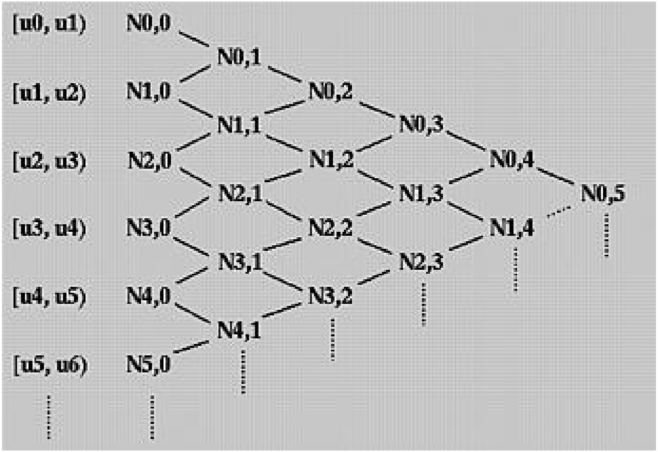


B-spline basis function

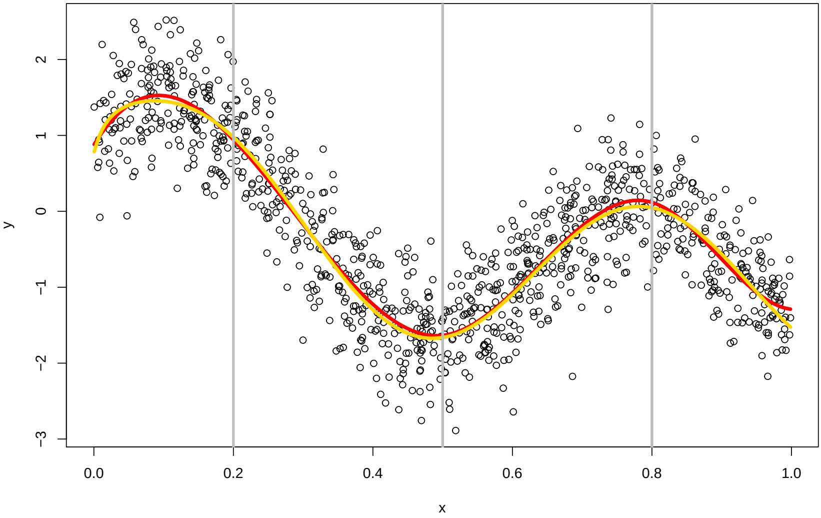
- degree 2



B-spline basis function



Regression spline



Single x to multiple $\mathbf{x}$

- Previous discussions focus on modeling the relationship between y and a single x.
- How about extending it to mutiple $\mathbf{x}$?

Generalized additive model (GAM)

- **GAM**:

$$y = f_0 + f_1(x_1) + f_2(x_2) + \cdots f_p(x_p) + \epsilon$$

, where

$$f_j(x_j) = \sum_{m=1}^{M_j} \beta_{jm} \phi_{jm}(x_j)$$

- Each x_i are allowed to take a regression spline form.

Fitting algorithm

- Since each $f_j(x_j)$ takes the regression spline form, the GAM is analogous to a GLM.
- Parameters could be estimated with the same idea as the GLM.

Fitting algorithm

1. For each j, $f_j(x_j) = \sum_{m=1}^{M_j} \beta_{jm}\phi_{jm}(x_j)$
2. Let M_j be sufficiently large for over fitting.
3. Maximize the log likelihood with a penalty to '`departure from smoothness`'

$$\max \mathcal{L}(\beta) - \sum_j \lambda_j \int f_j''(x)^2 dx$$

4. Since $\int f_j''(x)^2 dx = \beta_j^T S_j \beta_j$

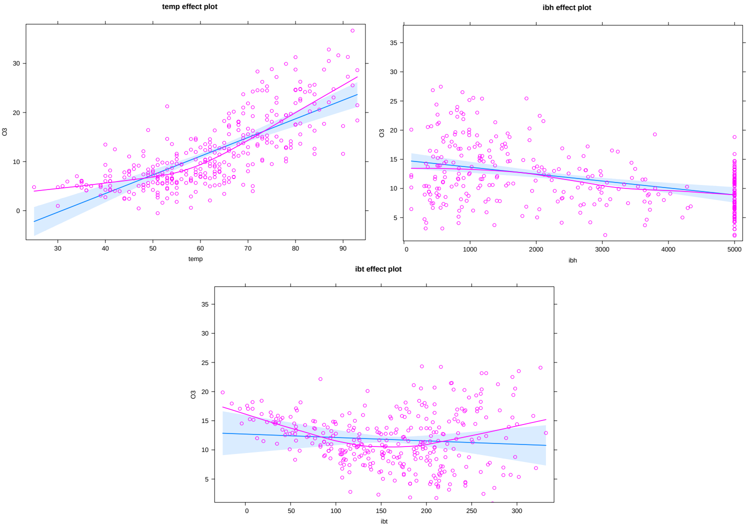
$$\hat{\beta} = \text{argmax}_\beta \left\{ \mathcal{L}(\beta) - \sum_j \lambda_j \beta^T S_j \beta \right\}$$

5. λ could be optained from Generalized cross validation.

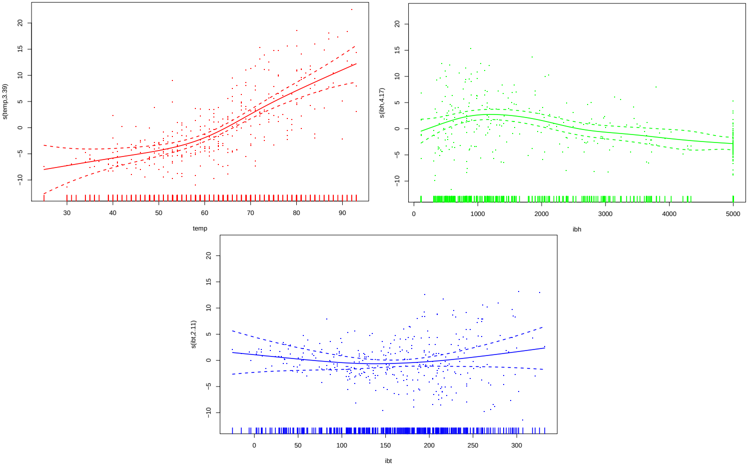
Modeling Ozone Concentration in LA

```
##
## Call:
## lm(formula = O3 ~ temp + ibh + ibt, data = ozone)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.3224  -3.1913  -0.2591   2.9635  13.2860
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7.7279822  1.6216623  -4.765 2.84e-06 ***
## temp         0.3804408  0.0401582   9.474  < 2e-16 ***
## ibh         -0.0011862  0.0002567  -4.621 5.52e-06 ***
## ibt         -0.0058215  0.0101793  -0.572    0.568
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
```

Modeling Ozone Concentration in LA



Modeling Ozone Concentration in LA



Generalized additive models (GAM)

Actually, regression splines are not the only choice for $f_j(x_j)$.

- For example, we can fit $f_1(x_1)$ with linear regression, $f_2(x_2)$ with cubic, $f_3(x_3)$ with random forest, $f_4(x_4)$ with neural net, etc.

How it works?

Recall GAM:

$$y = f_0 + f_1(x_1) + f_2(x_2) + \cdots f_p(x_p) + \epsilon$$

By **Backfitting algorithm**

1. Initialize each $f_j(x_j)$.
2. Cycle from $j = 1, ..., p, 1, ..., p, ...,$

$$f_j = S\left(x_j, y - \beta_0 - \sum_{i \neq j} f_i(X_i)\right)$$

, where $S(x, y)$ means the smooth on data (x, y)

Fitting algorithm - backward fitting

E.g. backward fitting for linear model

$$\mathsf{E}[Y \mid X] = \sum_{j=0}^{p} \beta_j x_j$$

$$\begin{aligned}
\mathrm{E}\left[Y \mid X_k = x_k\right] &= \mathrm{E}\left[\mathrm{E}\left[Y \mid X_1, X_2, \ldots X_k, \ldots X_p\right] \mid X_k = x_k\right]^1 \\
&= \mathrm{E}\left[\sum_{j=0}^{p} \beta_j X_j \mid X_k = x_k\right] \\
&= \beta_k x_k + \mathrm{E}\left[\sum_{j \neq k} \beta_j X_j \mid X_k = x_k\right]
\end{aligned}$$

[1] the law of total expectation $\mathbf{E}[Y \mid X] = \mathrm{E}[\mathrm{E}[Y \mid X, Z] \mid X]$

Fitting algorithm - backward fitting

$$\beta_k x_k = \mathrm{E}\left[Y \mid X_k = x_k\right] - \mathrm{E}\left[\sum_{j \neq k} \beta_j X_j \mid X_k = x_k\right]$$

$$\beta_k x_k = \mathrm{E}\left[Y - \left(\sum_{j \neq k} \beta_j X_j\right) \mid X_k = x_k\right]$$

By doing regression on the equation above, we can estimate β_k with a single regression instead of doing a multiple regression.

Backfitting algorithm - an example

True model

$$y = 20x_1^4 - 16x_1^2 + 2\sin(10x_2) - 3x_3 + \epsilon$$

Fit with

$$\mathrm{E}(y) = \alpha + f_1(x_1) + f_2(x_2) + f_3(x_3)$$

where f_1 is a regression spline, f_2 is a neural net, and f_3 is a linear regression.

Backfitting algorithm - an example

```
while (err > 1e-3) {
  alpha <- mean(mu)
  mu <- y - alpha ## fitting alpha
  data <- data.frame(mu,x1,x2,x3)
  fit1 <- gam(mu~s(x1),data = data)
  mu <- y - predict(fit1,data) ## fitting f1
  data <- data.frame(mu,x1,x2,x3)
  fit2 <- nnet(mu~x2,data = data,size = 10,linout = T) ## f
  mu <- y - predict(fit2,data)
  data <- data.frame(mu,x1,x2,x3)
  fit3 <- lm(mu~x3+0,data = data) ## fitting f3
  mu <- y - predict(fit3,data)
  muall <- y -predict(fit1,data)-as.numeric(predict(fit2,da
  err <- sum((muall-mu0)^2)/n  ## calculate MSE
  mu0 <- muall
}
```

Backfitting algorithm - an example

```
MSE
```

```
##       Backward        RF      nnet   linear
## [1,] 6.923679 0.4121533 0.8325506 4.221215
```