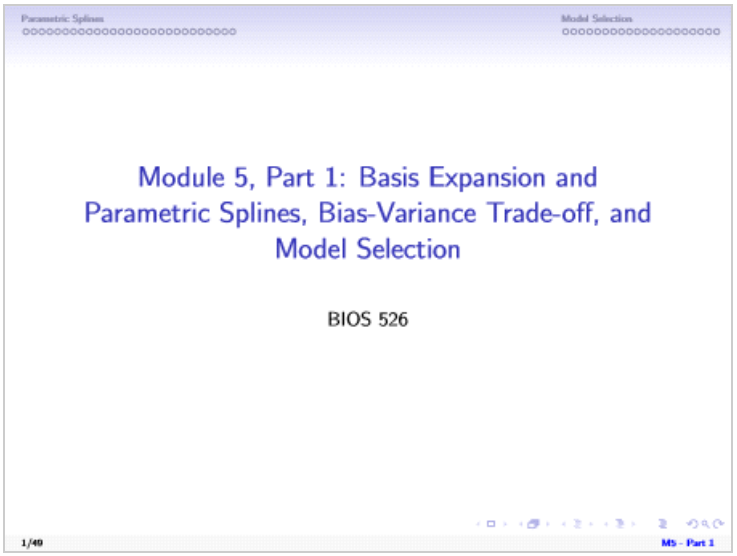
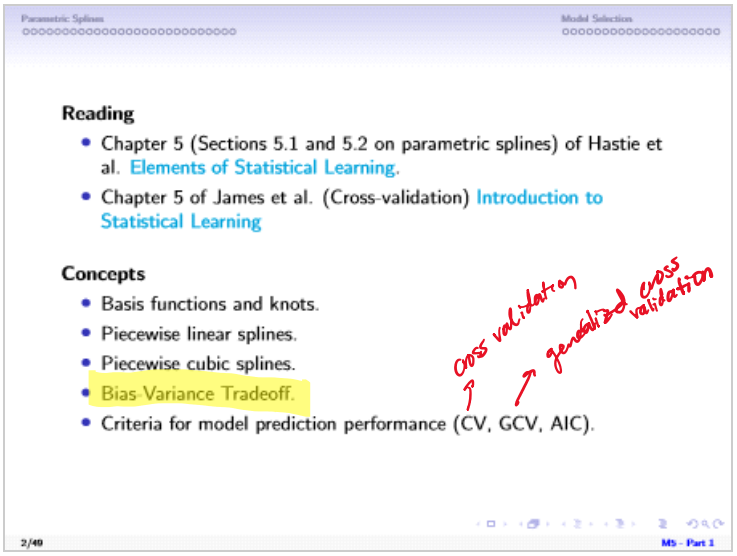
Module 5 Part 1: Splines

Wednesday, October 25, 2023 14:29

BIOS526_M
5_PartI_S...





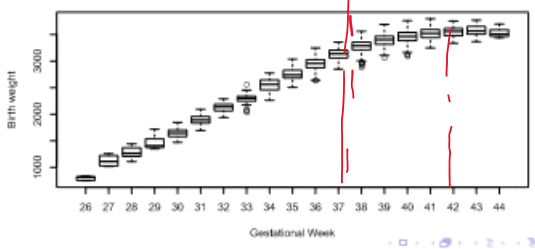
Parametric Splines

Motivating Example: Birth Weight and Mother's Age

- gw: gestational age in weeks
- age: maternal age at delivery.
- bw: birth weight at delivery in grams.

```
> load ("GAbirth.RData")
> str (dat)
'data.frame':   5000 obs. of  3 variables:
 $ gw : num  41 38 39 40 40 38 40 40 38 37 ...
 $ age: int  33 23 36 30 26 18 19 37 23 25 ...
 $ bw : num  3540 3287 3438 3419 3278 ...
```

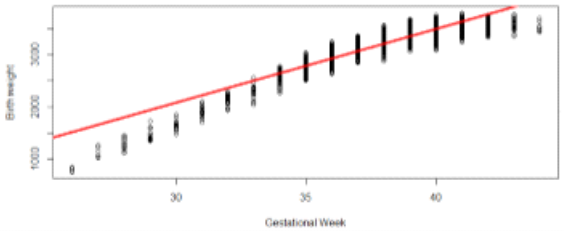
See the nonlinearity



Linear Regression

birth weight ~ gest. week

```
> fit = lm (bw~gw, data = dat)
> summary(fit)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -2177.513     39.351  -55.34   <2e-16 ***
gw            141.808      1.019  139.10   <2e-16 ***

> plot (dat$bw~dat$gw, xlab = "Gestational Week", ylab = "Birth weight")
> abline(fit, col=2, lwd = 3)
```

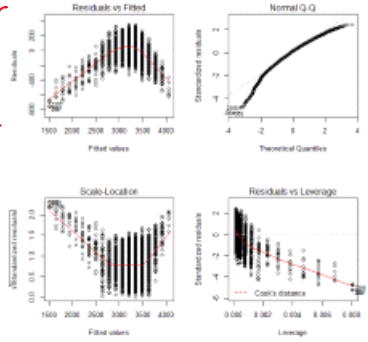


Linear Regression Diagnostics

```
> par(mfrow = c(2,2)); plot (fit)
```

strong evidence of non-linearity

terrible...



this is "functional data analysis"

Basis Expansion

Consider the regression model:

just some function of gestational week

$$y_i = g(x_i) + \epsilon_i \, .$$

One approach: assume $g(x_i)$ is a linear combination of M functions of x_i

$$y_i = \sum_{m=1}^{M} \beta_m b_m(x_i) + \epsilon_i \, . \qquad (1)$$

sum of basis functions

We will choose some simple $b_m(x_i)$.

some known, fixed function

Instead of estimating $g(x_i)$, we estimate the scalars β_m. For sufficiently large M, can do a good job of approximating any $g(x_i)$.

estimate the magnitude

Nice class of fcns to work on?

Replace x_i with $b_m(x_i)$, $m = 1, \ldots, M$, in a new regression.

Note $b_m(x_i)$ is simply a transformation of the original variable x_i.

$b_m(x_i)$ is called a basis function.

Basis Expansion, continued

- Basis functions are a core tool in the field of functional data analysis: observations are curves or functions.
- Basis functions extend basis vectors from linear algebra.
- Recall: in OLS, $\hat{\mathbf{Y}} = \mathbf{X}\hat{\beta}$, represent $\mathbf{Y} \in \mathbb{R}^n$ with the rank-p colspace($\mathbf{X}$).
  - but now corresponds to diff. functions?
- $\mathbf{X}$: each covariate forms a separate vector of the basis.
- Now, we consider a single covariate, but multiple functions of the single covariate.

Design matrix for a single variable X becomes

$$\begin{bmatrix} 1 & b_1(x_1) & \cdots & b_m(x_1) \\ 1 & b_1(x_2) & & \vdots \\ \vdots & \vdots & & \\ 1 & b_1(x_n) & & b_m(x_n) \end{bmatrix}$$

Concrete example

Basis Expansion Examples

we choose, then estimate betas from there

Polynomial

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \varepsilon_i$$

$g(x_i) = \sum \beta_n b_n(x_i)$??

$$b_0(x_i) = 1 \qquad b_1(x_i) = x_i \qquad b_2(x_i) = x_i^2 \qquad b_3(x_i) = x_i^3$$

Indicator

jump in predicted $\hat{y}$'s

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 I_{x_i > 0} + \varepsilon_i$$

$$b_0(x_i) = 1 \qquad b_1(x_i) = x_i \qquad b_2(x_i) = I_{x_i > 0}$$

1 if $x_i > 0$
0 otherwise

Periodic

$$y_i = \beta_0 + \beta_1 \sin(x_i/K) + \beta_2 \cos(x_i/K) + \varepsilon_i$$

$$b_0(x_i) = 1 \qquad b_1(x_i) = \sin(x_i/K) \qquad b_2(x_i) = \cos(x_i/K)$$

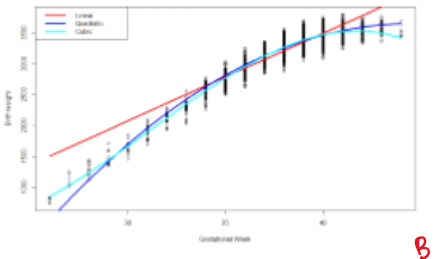
Polynomial Regression

creates x^2

```
> fit2 = lm (bw~gw+I(gw^2), data = dat)
> fit3 = lm (bw~gw+I(gw^2) + I(gw^3), data = dat)
```

Quadratic polynomial

↑ Wait polynomial



BAD TAIL BEHAVIOR

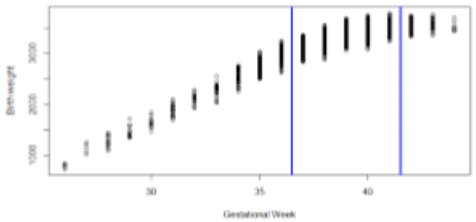
Note that polynomial functions often cannot model the ends very well.

Growth rates may differ

Piecewise Regression aka step functions

To model non-linear relationship: divide range of the covariate into some suitable regions → these will define our knots

E.g., pregnancy length: preterm (< 37 weeks), full-term (37-41 weeks), and post-term (> 42 weeks).



Model with polynomial functions separately within each region. The interior values that define the regions are called knots.

Piecewise Regression Specification

Piecewise regression is equivalent to a model where the basis functions of x_i interact with dummy variables for regions.

E.g., Piecewise quadratic regression:

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_i + \beta_2 x_i^2 \\ &+ \beta_3 D_{1i} + \beta_4 x_i \times D_{1i} + \beta_5 x_i^2 \times D_{1i} \\ &+ \beta_6 D_{2i} + \beta_7 x_i \times D_{2i} + \beta_8 x_i^2 \times D_{2i} \end{aligned}$$

linear effect
quadratic effect

where

region1

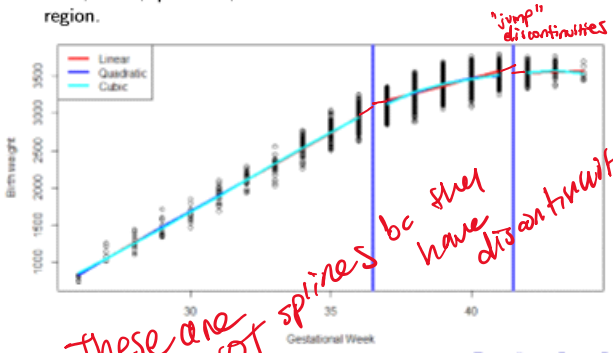
$$D_{1i} = \begin{cases} 1 & 37 \le x_i < 42 \\ 0 & \text{otherwise} \end{cases} \qquad D_{2i} = \begin{cases} 1 & x_i \ge 42 \\ 0 & \text{otherwise} \end{cases}$$

We only need **two** dummy variables for three regions.

- separate intercept indicator var...

Piecewise Regression

- The positive association between pregnancy length and birth weight decreases for higher gestational weeks.
- Here, linear, quadratic, and cubic result in similar trend in each region.



"jump" discontinuities

these are NOT splines bc they have discontinuities

Piecewise Splines

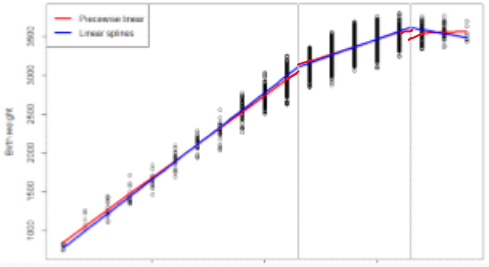
Piecewise polynomials have limitations:

- The regression functions do not match at knot locations.
- Requires many regression coefficients (degrees of freedom).

Splines are basis functions for piecewise regressions that are connected at the interior knots.

i.e. will be continuous

Splines are continuous, which makes them aesthetically appealing.



Linear Splines

"statistical learning"

A linear piecewise spline at knot locations 36.5 and 41.5 is specified as

one intercept

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 (x_i - 36.5)_+ + \beta_3 (x_i - 41.5)_+$$

"global" slope

Here we only need 4 regression coefficients, instead of 6 in a model that does not force the regression lines to connect at knots.

Count the parameters: 2 parameters per line * 3 lines - 2 constraints = 4

where $(\ldots)_+$ denotes the positive part

$$(x_i - 36.5)_+ = \begin{cases} x_i - 36.5 & \text{if } x_i \geq 36.5 \\ 0 & \text{otherwise} \end{cases}$$

$$(x_i - 41.5)_+ = \begin{cases} x_i - 41.5 & \text{if } x_i \geq 41.5 \\ 0 & \text{otherwise} \end{cases}$$

Linear Splines: Coefficient Interpretation

$$y_i = \underbrace{\beta_0 + \beta_1 x_i}_{<36.5} + \underbrace{\beta_2 (x_i - 36.5)_+}_{36.5-41.5} + \underbrace{\beta_3 (x_i - 41.5)_+}_{>41.5}$$

For $x_i < 36.5$:

$$y_i = \beta_0 + \beta_1 x_i \,.$$

For $36.5 \geq x_i < 41.5$:

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_i + \beta_2 (x_i - 36.5) \\ &= (\beta_0 - \beta_2 * 36.5) + (\beta_1 + \beta_2) x_i \,. \end{aligned}$$

changes intercept (glues together $\hat{y}_i$ to make it continuous)

changes slope

For $41.5 \geq x_i$:

$$\begin{aligned} y_i &= \beta_0 + \beta_1 x_i + \beta_2 (x_i - 36.5) + \beta_3 (x_i - 41.5) \\ &= (\beta_0 - \beta_2 * 36.5 - \beta_3 * 41.5) + (\beta_1 + \beta_2 + \beta_3) x_i \,. \end{aligned}$$

β_2 and β_3 represent changes in slope compared to the previous region

Linear Splines: Coefficient Interpretation

See MS_part1_Splines.R

```
> Sp1 = (dat$gw - 36.5)*as.numeric(dat$gw >= 36.5 )
> Sp2 = (dat$gw - 41.5)*as.numeric(dat$gw >= 41.5)

> fit7 = lm (bw~ gw+Sp1 + Sp2, data = dat)
> summary (fit7)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -5013.768     62.747  -79.90   <2e-16 ***
gw            222.620      1.757  126.71   <2e-16 ***
Sp1          -121.427      2.549  -47.64   <2e-16 ***
Sp2          -152.948     10.492  -14.58   <2e-16 ***
```

$\widehat{bw}_i = -5014 + 223\,week_i - 121\,(week_i - 36.5)_+ - 153\,(week_i - 41.5)_+$

- A 223 g ($CI_{95\%}$ 219, 226) increase in birth weight was associated with a one week increase in pregnancy length prior to week 36.5.
  (gestation time)
- The rate of increase then drops by 121.4 g/week to 101.2 g/week ($CI_{95\%}$ 99, 104) between pregnancy weeks 36.5 to 41.5.
  (see R code)
- After pregnancy week 41.5, birth weight was negatively associated with gestational week by -52 g/week ($CI_{95\%}$ -72, -32).

slope $\pm$ $t_{.975}$ (SE)

where SE → $Cov(\hat{C}'\beta) = C' Vcov(\hat{\beta}) C$

- selection region? A vector of constants, e.g. c(0,1,1,0)
- variance covariance matrix of the coefficients

Cubic Splines

Linear splines are easy to interpret → e.g. like a shift in growth rate

The resulting function $g(x_i)$ is not smooth. The first-derivative of

BUT

Linear splines are easy to interpret. → e.g. like a shift in growth rate

The resulting function $g(x_i)$ is not smooth. The first-derivative of BUT

$$g(x_i) = \beta_0 + \beta_1 x_i + \beta_2 (x_i - 36.5)_+ + \beta_3 (x_i - 41.5)_+$$

is discontinuous at the knots.

We can also work with piecewise cubic functions that are connected at the knots:

$$g(x_i) = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 (x_i - 36.5)_+^3 + \beta_5 (x_i - 41.5)_+^3$$

Cubic splines are popular because:

- continuous 2nd-derivatives, typically what we view as *smooth* visually.

"folks like these bc they look nice"

18/49 MS - Part 1

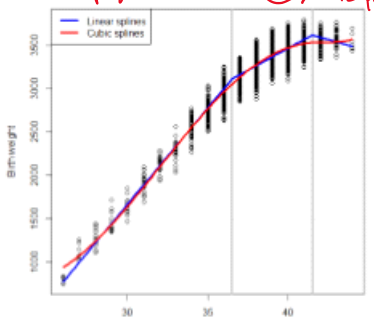
e.g.
c(0,1,1,0)
will select
β_1 & β_2

Parametric Splines Model Selection

Cubic versus Linear Splines

```
> Sp1 = (dat$gw - 36.5)^3*as.numeric(dat$gw >= 36.5 )
> Sp2 = (dat$gw - 41.5)^3*as.numeric(dat$gw >= 41.5)
> fit8 = lm (bw~ gw+I(gw^2) + I(gw^3) + Sp1 + Sp2, data = dat)
```

cubic polynomial → cubic polynomial w/ knot



19/49 MS - Part 1

Parametric Splines Model Selection

Polynomial Splines

The general formulation of a d-degree polynomial spline model with M knots, $\kappa_1, \kappa_2, \ldots, \kappa_M$ is

$$y_i = \beta_0 + \sum_{j=1}^{d} \beta_j x_i^j + \sum_{m=1}^{M} \beta_{d+m} (x_i - \kappa_m)_+^d .$$

knots

The above is known as the truncated power formulation.

Linear (d = 1):

$$y_i = \beta_0 + \beta_1 x_i + \sum_{m=1}^{M} \beta_{1+m} (x_i - \kappa_m)_+ .$$

Quadratic (d = 2):

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \sum_{m=1}^{M} \beta_{2+m} (x_i - \kappa_m)_+^2 .$$

20/49 MS - Part 1

Interpreting Polynomial Splines

- Linear splines: a good choice for interpretability – the coefficients at the knots represent the change in slope from the previous region.
- Cubic splines: appears smooth, which is often biologically reasonable. [Generally we do not interpret the individual coefficients of the truncated polynomials. for d>1]

```
lm(formula = bw ~ gw + I(gw^2) + I(gw^3) + Sp1 + Sp2, data = dat)
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.890e+04  3.717e+03   7.776 9.02e-15 ***
gw          -2.968e+03  3.335e+02  -8.901  < 2e-16 ***
I(gw^2)      9.972e+01  9.900e+00  10.073  < 2e-16 ***
I(gw^3)     -1.036e+00  9.734e-02 -10.641  < 2e-16 ***
Sp1          7.732e-01  2.482e-01   3.115  0.00185 **
Sp2          7.455e+00  3.471e+00   2.148  0.03177 *
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Residual standard error: 109.8 on 4994 degrees of freedom
Multiple R-squared:  0.881,Adjusted R-squared:  0.8809
F-statistic:  7397 on 5 and 4994 DF,  p-value: < 2.2e-16
```

don't want to interpret coefs individually (eventually we will use approx. F-stats)

B-Splines → lots of "incremental improvements"

The truncated power formulation can sometimes experience numerical problems when fitting because:

- the covariates are highly correlated;
- d^{th} power can be small or large with large d.

B-splines can be used to represent the same space as truncated polynomial splines. Just a diff. way of representing polynomial splines. You'll get the same fit

Mathematically, these will produce the same predictions as truncated polynomial splines.

Computationally, may be more accurate – avoid overflow errors.

Scaled between 0 and 1. Provide better numerical stability.

Commonly implemented in statistical software but in my experience you can just use cubic splines with modern computers.

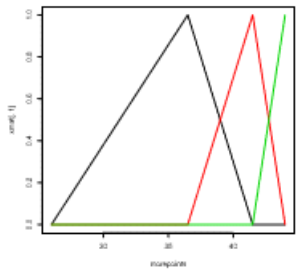
B-Splines and computation

The bs() function in R will create the appropriate design matrix.

```
### Load the splines package
library (splines)

### Linear splines                                          linear
fit1 = lm ( bw ~ bs(gw, knots = c(36.5, 41.5), degree = 1),  data = dat)

### Quadratic splines                                       quadratic
fit2 = lm ( bw ~ bs(gw, knots = c(36.5, 41.5), degree = 2),  data = dat)

### Cubic splines                                           cubic
fit3 = lm ( bw ~ bs(gw, knots = c(36.5, 41.5), degree = 3),  data = dat)
```

B-Splines

E.g. of degree 1 (linear) B-spline and degree 3 (cubic) B-spline.
In general, we do not use degree 1 B-splines – piecewise linear splines are fine.
It's the cubic polynomials that blow up. Cubic b-splines are popular.



Linear B-spline

B-Splines

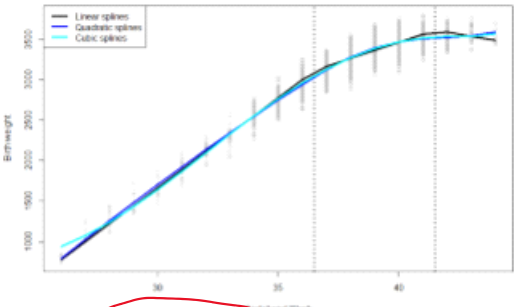
Coefficients are not interpretable.

```
lm(formula = bw ~ bs(gw, knots = c(36.5, 41.5), degree = 3),
    data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-399.13  -74.61    4.47   77.06  301.40

Coefficients:
                                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                                937.45      34.65  27.056  < 2e-16
bs(gw, knots = c(36.5, 41.5), degree = 3)1  408.19      57.92   7.048 2.07e-12
bs(gw, knots = c(36.5, 41.5), degree = 3)2 2037.52      32.46  62.779  < 2e-16
bs(gw, knots = c(36.5, 41.5), degree = 3)3 2655.45      38.15  69.608  < 2e-16
bs(gw, knots = c(36.5, 41.5), degree = 3)4 2582.17      35.94  71.851  < 2e-16
bs(gw, knots = c(36.5, 41.5), degree = 3)5 2633.37      52.76  49.912  < 2e-16
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Residual standard error: 109.8 on 4994 degrees of freedom
Multiple R-squared:  0.881,Adjusted R-squared:  0.8809
F-statistic:  7397 on 5 and 4994 DF,  p-value: < 2.2e-16
```

Cubic version

Again, individually they are not interpretable

B-Splines



Note:

```
> sum((fit7$fitted-fit9$fitted)^2)
[1] 4.9437e-18
> sum((fit8$fitted-fit11$fitted)^2)
[1] 2.056156e-13
```

(numerically, no difference close enough to 0)

linear splines

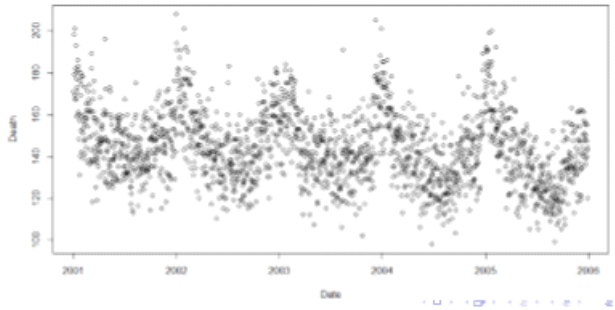
linear B-splines

Intuition:
$\beta_1 x_i$ or $(2\beta_1)\left(\frac{x_i}{2}\right)$
is the same thing

Example: Daily Death Counts in NYC

```
> load ("NYC.RData")
> plot (health$alldeaths~health$date, xlab = "Date", ylab = "Death")
```

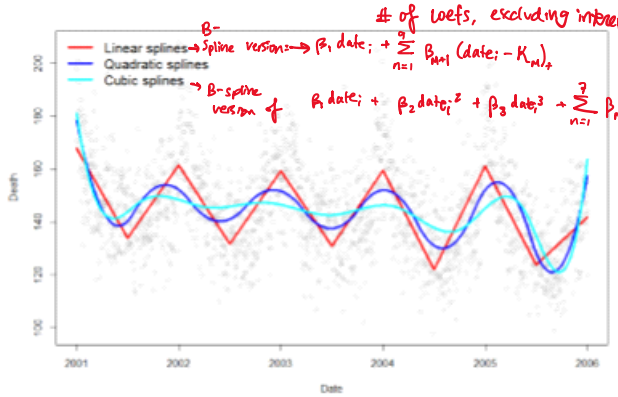
Daily non-accidental mortality counts in 5-county New York City, 2001-2006.



Knots at quantiles, df=10

of coefs, excluding intercept

B-spline version: $\beta_1 date_i + \sum_{m=1}^{9} \beta_{m+1} (date_i - K_m)_+$

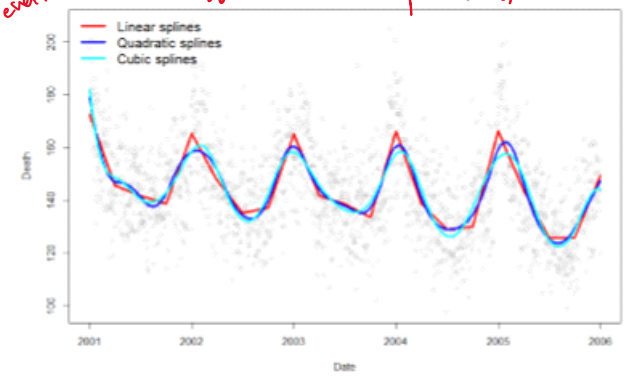


B-spline version of $\beta_1 date_i + \beta_2 date_i^2 + \beta_3 date_i^3 + \sum_{m=1}^{7} \beta_{m+3} (date_i - K_m)_+^3$

penalize the wiggliness eventually?

Knots at quantiles, df = 20

wigglier curves, 20 parameters, more knots



Model Selection

Model Selection

What model to use? Cross validation can help us pick.

We need to pick:

1. Which basis function? Linear versus cubic. → don't worry too much, we'll use mgcv
2. How many knots?
3. Where to put the knots?

Challenge: these models are usually not nested!

Main Ideas:

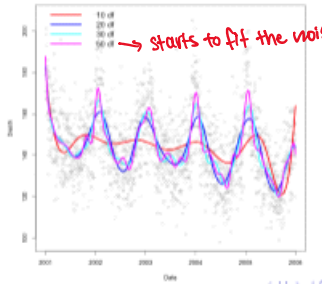
- Choice of basis function usually does not have a large impact on model fit, especially when there are enough knots and $g(x)$ is smooth.
- When there are enough knots, their locations are less important too.

For now, we'll focus on the **how many** question.

Bias-Variance Trade-Off

More knots → { Can better capture fine-scale trends; Need to estimate more coefficients → so we risk overfitting }

Cubic Splines with Different Number of Knots



Low bias, High variance (when u use testing dataset)

High bias low variance

Towards the Bias-Variance Decomposition

Assume Y comes from some true model

$$Y = g(X) + \epsilon, \qquad \epsilon \sim (0, \sigma^2) .$$

start w/ non-parametric regression

where $g(\cdot)$ is some unknown function.

For given x_i, we estimate Y with $\hat{g}(x_i)$, where here we assume $\hat{g}(x_i) \perp\!\!\!\perp \epsilon$. In other words, x_i is an "unseen" data point.

The expected squared difference between Y and the estimator $\hat{g}(X)$ is the population mean squared error (MSE). For given x_i,

population MSE

$$E[\{Y - \hat{g}(X)\}^2 | X = x_i] = \texttt{derive...}$$
$$= \sigma^2 + E\{g(x_i) - \hat{g}(x_i)\}^2$$

don't know — true mean — estimate

Derivations

$E[Y - \hat{g}(x_i)]^2 = E[(Y - g(x_i)) + g(x_i) - \hat{g}(x_i)]^2$

$= E(Y - g(x_i))^2 + 2E(Y - g(x_i))(g(x_i) - \hat{g}(x_i)) + E[g(x_i) - \hat{g}(x_i)]^2$

$E(Y - g(x_i))^2 = \sigma^2$

$2E(Y - g(x_i))(g(x_i) - \hat{g}(x_i)) = 2E[\epsilon_i (g(x_i) - \hat{g}(x_i))]$

$= \sigma^2 + 2E\epsilon_i \, E[g(x_i) - \hat{g}(x_i)] + E[g(x_i) - \hat{g}(x_i)]^2$

$2E\epsilon_i = 0$

$= \boxed{\sigma^2 + E[g(x_i) - \hat{g}(x_i)]^2}$

can't do anything about σ^2

Bias-Variance Decomposition, cont.

Note: $\hat{g}$ is estimated from y, which is random

$$E\{g(x_i) - \hat{g}(x_i)\}^2 = \ldots$$

$= E[g(x_i)^2] - 2E\, g(x_i)\hat{g}(x_i) + E[\hat{g}(x_i)^2]$

so $E[g(x_i)]$ is fixed but $E[\hat{g}(x_i)]$ is NOT

$= g(x_i)^2 - 2g(x_i)\, E\hat{g}(x_i) + E[\hat{g}(x_i)^2]$ ★

$\text{Bias}^2 = [g(x_i) - E\hat{g}(x_i)]^2 = g(x_i)^2 - 2g(x_i)E\hat{g}(x_i) + [E\hat{g}(x_i)]^2$

$\text{Variance} = E[\hat{g}(x_i)^2] - E[\hat{g}(x_i)]^2$

$\text{Bias}^2 + \text{Variance} = g(x_i)^2 - 2g(x_i)E\hat{g}(x_i) + [E\hat{g}(x_i)]^2 + E[\hat{g}(x_i)^2] - E[\hat{g}(x_i)]^2$

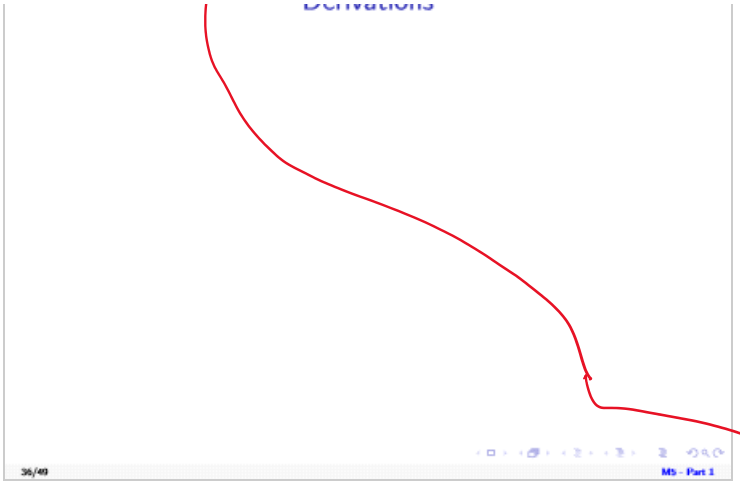
$= g(x_i)^2 - 2g(x_i)E\hat{g}(x_i) + E[\hat{g}(x_i)^2]$

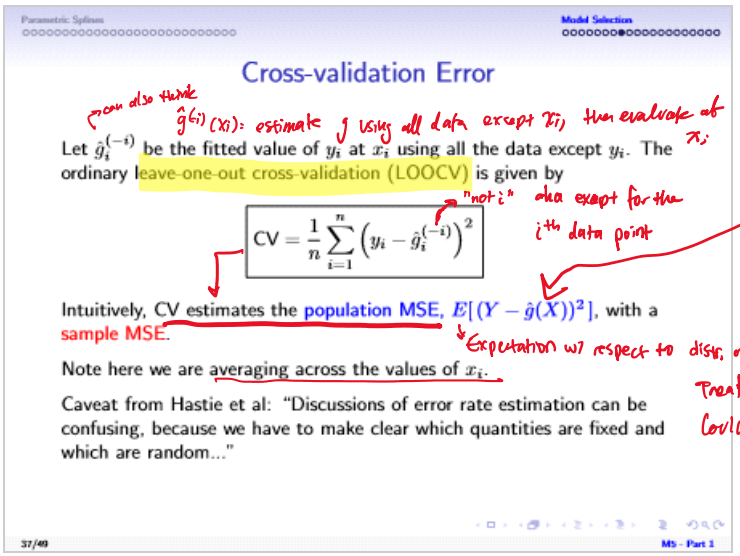
$=$ ★

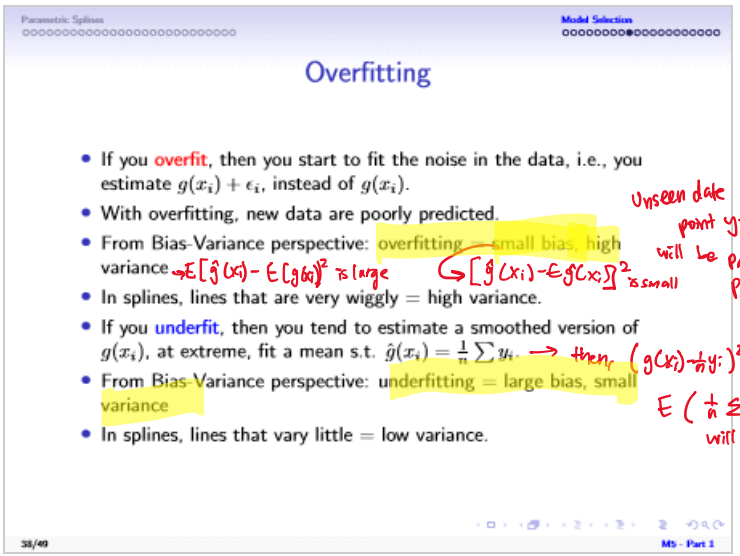
$=$ MSE

$\text{Var}(x) = E(x^2) - E(x)^2$

Derivations


Derivations
36/49
MS - Part 1


Parametric Splines
Model Selection
Cross-validation Error
Let $\hat{g}_i^{(-i)}$ be the fitted value of y_i at x_i using all the data except y_i. The ordinary leave-one-out cross-validation (LOOCV) is given by
$$\text{CV} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{g}_i^{(-i)}\right)^2$$
Intuitively, CV estimates the population MSE, $E[(Y - \hat{g}(X))^2]$, with a sample MSE.
Note here we are averaging across the values of x_i.
Caveat from Hastie et al: "Discussions of error rate estimation can be confusing, because we have to make clear which quantities are fixed and which are random..."
37/49
MS - Part 1


Parametric Splines
Model Selection
Overfitting
If you overfit, then you start to fit the noise in the data, i.e., you estimate $g(x_i) + \epsilon_i$, instead of $g(x_i)$.
With overfitting, new data are poorly predicted.
From Bias-Variance perspective: overfitting = small bias, high variance
In splines, lines that are very wiggly = high variance.
If you underfit, then you tend to estimate a smoothed version of $g(x_i)$, at extreme, fit a mean s.t. $\hat{g}(x_i) = \frac{1}{n}\sum y_i$.
From Bias-Variance perspective: underfitting = large bias, small variance
In splines, lines that vary little = low variance.
38/49
MS - Part 1

Cross-validation Error, OLS

In OLS, leave-one-out cross-validation can actually be done w/o fitting n regressions

For linear regression, LOOCV can be obtained easily from $\hat{y}_i$, as predicted using estimators from the *full* data:

here, from $\hat{\beta}$ estimated w/ all data

only in linear regression →

$$\text{CV} = \frac{1}{n}\sum_{i=1}^{n}\frac{(y_i - \hat{y}_i)^2}{(1 - H_{ii})^2},$$

$(X(X'X)^{-1}X')_{ii}$ ← ith diagonal element

where H_{ii} is the (i,i)th element of the hat matrix.

(This results from a clever application of a linear algebra trick, not discussed here.)

applies to any X, we can construct this from our basis i.e. also to $\begin{bmatrix} b_0(x_1) & \cdots & b_n(x_1) \\ \vdots & \ddots & \vdots \\ b_0(x_n) & \cdots & b_n(x_n) \end{bmatrix}$

This is the formula for any design matrix in OLS.

- weird thing that happens in OLS does not apply to penalized regression

Generalized Cross-validation Error

One variation of LOOCV is the generalized cross-validation (GCV)

generalized for penalized regressions →

$$\text{GCV} = \frac{1}{n}\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{[1 - n^{-1}tr(\mathbf{H})]^2}$$

trace means just the diagonals i.e. $\frac{1}{n}\sum_{i=1}^{n} H_{ii}$ aka the average.

Compared to CV,

- GCV replaces each H_{ii} by the average of all H_{ii}.
- GCV is a weighted version of CV.
- For complicated models (not OLS), CV is computationally costly, as it requires fitting the model many times.
- GCV often easier to calculate. can always do if $\hat{Y} = SY$

penalized regression

We want to pick a model which minimizes CV or GCV.

Note: k-fold not part of exam but GCV will be

K-Fold CV

used to estimate $\hat{g}^{-S_1}$, (not S_1)

1	2	3	4	5
S_1				

then apply $\hat{g}^{-S_1}$ to all indices contained in S_1 ? all $i \in S_1$

obtain $\hat{y}_i^{-S_1}$

- In K-Fold CV, the data are divided into K subsets

$$MSE_k = \frac{1}{n/K}\sum_{i \in S_k}(y_i - \hat{y}_i^{-S_k})^2$$

Cross-validation averages the MSE

$$CV = \widehat{MSE} = \frac{1}{K}\sum_{k=1}^{K} MSE_k$$

- Each partition has a training dataset with $(K-1) * n/K$ observations and test dataset with n/K observations.
- Then the model is fit K times.
- LOOCV is a special case where the number of folds is n.

Training, validation, and test dataset

issue: CV can still have issues w/ overfitting

Another approach is to divide the dataset into two datasets and use one dataset for estimation (training) and another for testing (test).

Popular splits: 70% (training) and 30% (test), or 80% and 20%.

This is often combined with cross-validation, where CV is applied to the training dataset to obtain the model, and then the model is applied to the unseen data to evaluate accuracy.

The "validation dataset" is the data left out when using cross-validation. In practice, we move across the different folds to tune hyperparameters. E.g., number of knots.

Popular in computer science with huge datasets.

Is there anything wrong with this approach? requires a lot of data

K-Fold CV, cont.

Bias-var tradeoff on MSE

- There is another bias-variance tradeoff ("meta" bias-variance tradeoff) regarding the optimal K
- K impacts the MSE of the sample estimate of the MSE,

$$E(MSE - \widehat{MSE})^2$$

where $\widehat{MSE}$ is from K-fold CV.

- If we let $K = 2$, we will do a poorer job of fitting the model (using less data) $\rightarrow$ upwardly biased $\widehat{MSE}$.

could underfit the model (too much bias)

- At the other extreme, LOO has least bias because we are using $n - 1$ observations to estimate the models, but each term in the summand is highly correlated, leading to higher variance.

leave one out

higher variance of $\widehat{MSE}$

- General recommendations: 5 or 10-fold CV to balance this tradeoff.
- A discussion is James et al 2013 Chapter 5.

We will use for elastic net $k = 5$ or 10 fold

for splines, we will use GCV or REML

Likelihood-Based Approach

With basis expansion, we assume g_i can be expressed as a linear combination of some linear regressors,

$$\mathbf{Y} \sim N(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I}) .$$

The Gaussian (Normal) likelihood is given by

$$f(\boldsymbol{\beta}, \sigma^2) = (2\pi\sigma^2)^{-n/2} \exp\{-\frac{1}{2\sigma^2}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})\}$$

with log-likelihood function

$$\log f(\boldsymbol{\beta}, \sigma^2) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) .$$

Given our estimated coefficients $\hat{\boldsymbol{\beta}}$, we hope to maximize

$$E\left[\log f(\hat{\boldsymbol{\beta}}, \sigma^2)\right] = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}E\left[(\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}})'(\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}})\right] .$$

Mallow's C_p and Akaike Information Criterion

It can be shown that

for Gaussian w/ known σ^2

$$E\left[\log f(\hat{\boldsymbol{\beta}}, \sigma^2)\right] \approx \log f(\hat{\boldsymbol{\beta}}, \sigma^2) + \frac{n}{2} - p.$$

It can be shown that

$$E\left[\log f(\hat{\beta}, \sigma^2)\right] \approx \log f(\hat{\beta}, \sigma^2) + \frac{n}{2} - p.$$

So one approach is to select a model that maximizes the above. If σ^2 is assumed known, this gives rise to the Mallow's C_p criterion:

$$\boxed{C_p = \frac{1}{\sigma^2}(\mathbf{Y} - \hat{\beta}\mathbf{X})'(\mathbf{Y} - \hat{\beta}\mathbf{X}) - n + 2p}.$$

In practice, σ^2 is often replaced by its estimate from the largest model considered.

Akaike information criterion (AIC) generalizes this to any likelihood:

$$\boxed{AIC = -2 \times \log f(\hat{\beta}, \hat{\sigma}^2) + 2p},$$

which we wish to minimize also.

45/49

Parametric Splines | Model Selection

Example: Mortality Trends

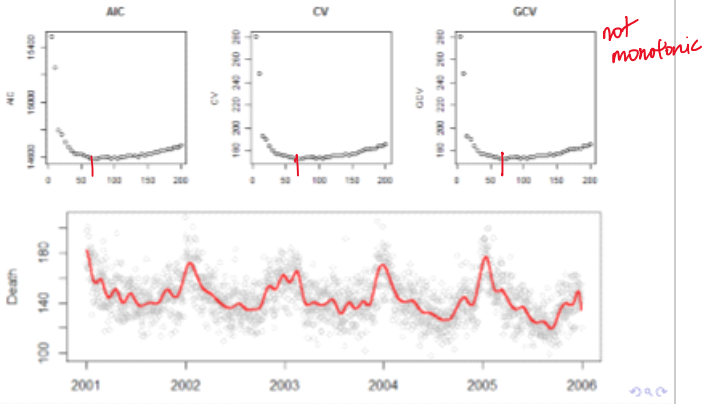
```
> ### AIC, CV, and GCV calculation examples:

> fit = lm (alldeaths~bs (date, df = 10), data = health)
>
> ##AIC
> AIC (fit)
[1] 15252.54
>

> ##GCV/CV
> H =  hatvalues (fit)

> CV = mean(  (fit$fitted - health$alldeaths)^2 / (1-H)^2 )
> CV
[1] 248.0374

> GCV = mean ((fit$fitted - health$alldeaths)^2) / (1 - mean(H))^2
> GCV
[1] 248.1518
```

46/49 | MS - Part 1

Parametric Splines | Model Selection

Example: Daily Death Counts in NYC

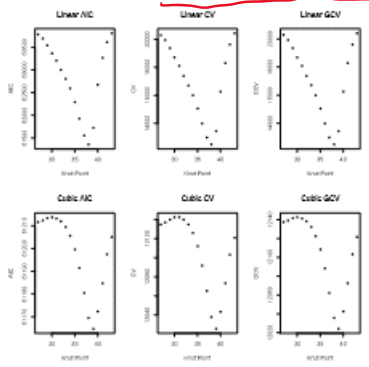
Optimal = 70 df.



47/49 | MS - Part 1

Example: Birth Weight and Pregnancy Length

Linear or cubic splines with 1 knot placed at different locations.



Example: Birth Weight and Pregnancy Length