Linear Regression Models
P8111

Lecture 17

Jeff Goldsmith
March 26, 2015

THE DEPARTMENT OF BIOSTATISTICS
Columbia University
MAILMAN SCHOOL
OF PUBLIC HEALTH

Today's Lecture

- Spline models
- Penalized spline regression

Spline models

- Last class – talked about penalized splines as an example of penalization, especially for bias/variance tradeoff
- Today we'll talk about spline regression more particularly

Piecewise linear models

A piecewise linear model (also called a change point model or broken stick model) contains a few linear components

- Outcome is linear over full domain, but with a different slope at different points
- Points where relationship changes are referred to as "change points" or "knots"
- Often there's one (or a few) potential change points

Piecewise linear models

Suppose we want to estimate $\mathrm{E}(y|x) = f(x)$ using a piecewise linear model.

- For one knot we can write this as

$$\mathrm{E}(y|x) = \beta_0 + \beta_1 x + \beta_2 (x - \kappa)_+$$

where κ is the location of the change point

Interpretation of regression coefficients

Estimation

- Piecewise linear models are low-dimensional (no need for penalization)
- Parameters are estimated via OLS
- The design matrix is ...

Multiple knots

Suppose we want to estimate $\mathrm{E}(y|x) = f(x)$ using a piecewise linear model.

- For multiple knots we can write this as

$$\mathrm{E}(y|x) = \beta_0 + \beta_1 x + \sum_{k=1}^{K} \beta_{k+1}(x - \kappa_k)_+$$

  where $\{\kappa_k\}_{k=1}^{K}$ are the locations of the change points
- Note that knot locations are defined before estimating regression coefficients
- Also, regression coefficients are interpreted conditional on the knots.

Piecewise quadratic and cubic models

Suppose we want to estimate $\mathrm{E}(y|x) = f(x)$ using a piecewise quadratic model.

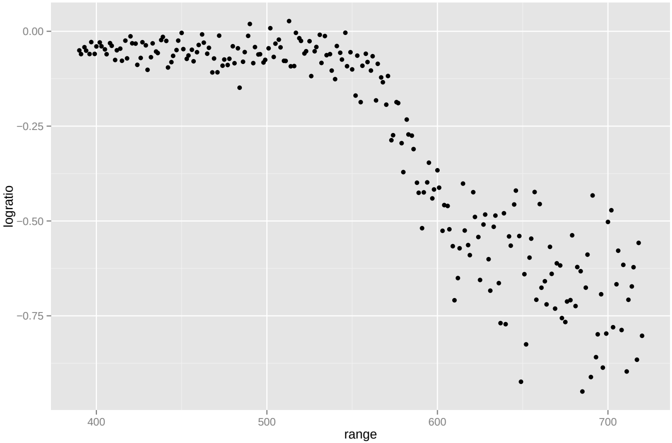
- For multiple knots we can write this as

$$\mathrm{E}(y|x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \sum_{k=1}^{K} \beta_{k+2}(x - \kappa_k)_+^2$$

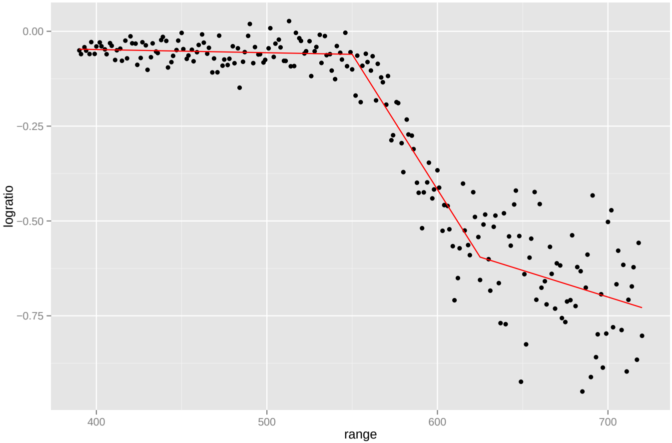
  where $\{\kappa_k\}_{k=1}^{K}$ are the locations of the change points
- Similar extension for cubics
- Piecewise quadratic models are smooth and have continuous first derivatives

Several ways to choose the location of knots ...

Example



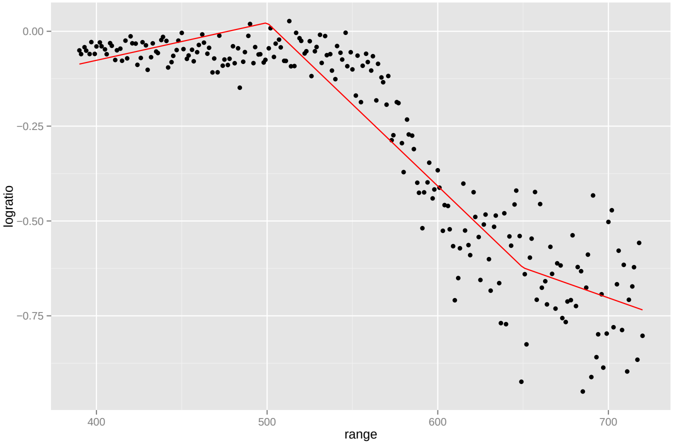
Example: piecewise linear



Example: piecewise linear

```
> knots <- c(550, 625)
> X.des = cbind(1, range, sapply(knots, function(k)
+                 ((range - k > 0) * (range - k))))
>
> lm.lin = lm(y ~ X.des - 1)
>
> summary(lm.lin)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
X.des      -1.444e-02  6.874e-02  -0.210    0.834
X.desrange -8.407e-05  1.427e-04  -0.589    0.556
X.des      -7.043e-03  3.834e-04 -18.368   <2e-16 ***
X.des       5.723e-03  5.154e-04  11.105   <2e-16 ***
```

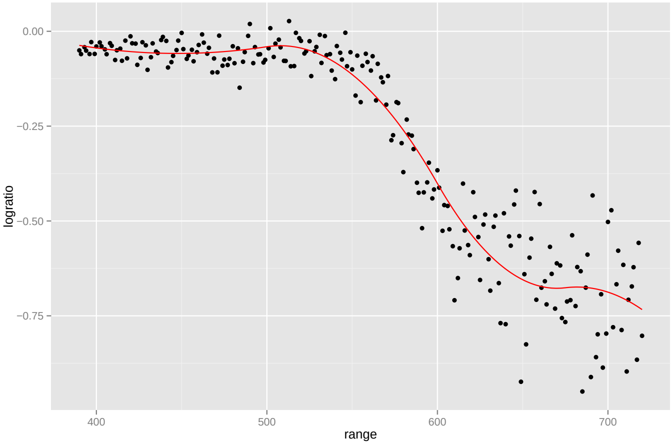
Example: piecewise linear



Example: piecewise linear

```
> knots <- c(500, 650)
> X.des = cbind(1, range, sapply(knots, function(k)
+                 ((range - k > 0) * (range - k))))
>
> lm.lin = lm(y ~ X.des - 1)
>
> summary(lm.lin)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
X.des      -0.4732124  0.1167565  -4.053 7.05e-05 ***
X.desrange  0.0009918  0.0002533   3.915 0.000121 ***
X.des      -0.0052984  0.0003695 -14.340  < 2e-16 ***
X.des       0.0027191  0.0005668   4.798 2.98e-06 ***
```

Example: piecewise quadratic



Example: piecewise quadratic

```
> knots <- c(500, 600, 675)
> X.des = cbind(1, range, range^2, sapply(knots, function(k)
+                  ((range - k > 0) * (range - k)^2)))
>
> lm.lin = lm(y ~ X.des - 1)
>
> summary(lm.lin)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
X.des       1.228e+00  1.058e+00   1.160   0.2472
X.desrange -5.749e-03  4.574e-03  -1.257   0.2101
X.des       6.424e-06  4.905e-06   1.310   0.1917
X.des      -4.923e-05  8.055e-06  -6.111 4.59e-09 ***
X.des       9.910e-05  1.012e-05   9.798  < 2e-16 ***
X.des      -9.708e-05  3.841e-05  -2.527   0.0122 *
```

Advantages of piecewise models

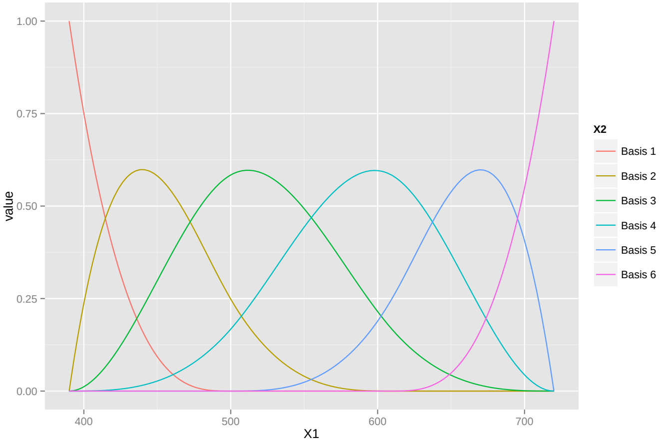
Piecewise (linear, quadratic, etc) models have several advantages

- Easy construction of basis functions
- Flexible, and don't rely on determining an appropriate form for $f(x)$ using standard functions
- Allow for significance testing on change point slopes
- Fairly direct interpretations

Other common spline models

- B-splines and natural splines similarly define a basis over the domain of x
- They are made up of piecewise polynomials of a given degree, and have defined derivatives similarly to the piecewise defined functions

B-splines



1.00
0.75
0.50
0.25
0.00
value
400
500
600
700
X1
X2
Basis 1
Basis 2
Basis 3
Basis 4
Basis 5
Basis 6

Penalized splines

- Often deciding the number and placement of knots is not straightforward
- If particular knots or associated tests are not of interest, penalization can help a lot
- It's been shown that, if one penalizes, the number and location of knots is not particularly important provided there are enough

Penalized splines

- Penalized splines are fairly common
- Recall we minimize RSS subject to a penalty; for smoothing, this penalty is often related to the second derivative of $f(x)$
- In the end, whatever penalty we use has an associated penalty matrix, and our estimates are of the form

$$\hat{\boldsymbol{\beta}}_\lambda = (\boldsymbol{X}^T\boldsymbol{X} + \lambda P)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

- Penalty matrix is known; λ is not

Penalized spline fits

Given $\hat{\boldsymbol{\beta}}_\lambda$:

- Fitted values are given by

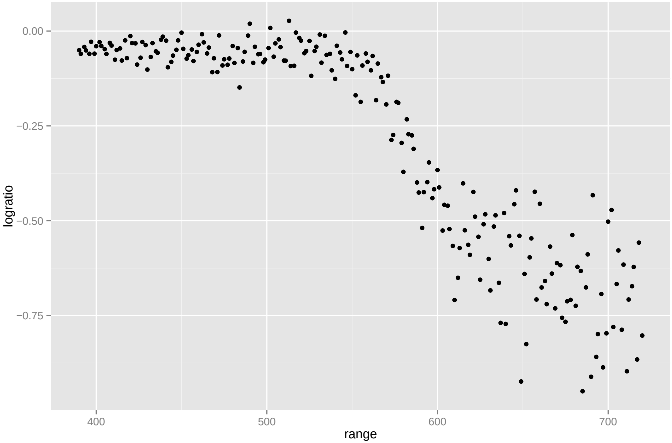
$$
\begin{aligned}
\hat{\boldsymbol{y}} &= \boldsymbol{X}\hat{\boldsymbol{\beta}}_\lambda \\
&= \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda P)^{-1}\boldsymbol{X}^T\boldsymbol{y}
\end{aligned}
$$

- The matrix $S = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X} + \lambda P)^{-1}\boldsymbol{X}^T$ is called the smoother matrix

Smoother matrix

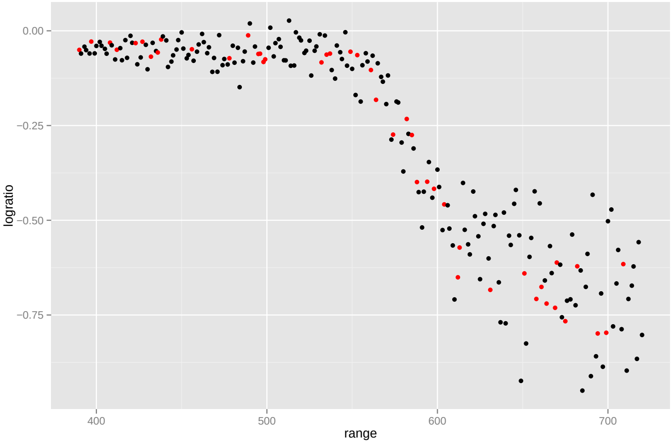
- The smoother matrix is like a hat matrix, but not quite:
  - For instance $SS \neq S$, unlike hat matrices
  - However, it gives fitted values and smooths the y's
- Recall that $tr(\boldsymbol{H}) = df$ for ordinary least squares
- For smoothing, $tr(S)$ is often called the effective degrees of freedom

Example: LIDAR data



Partition data

We'll random set aside about 20% of the data as "test" data and fit our model on the remaining points



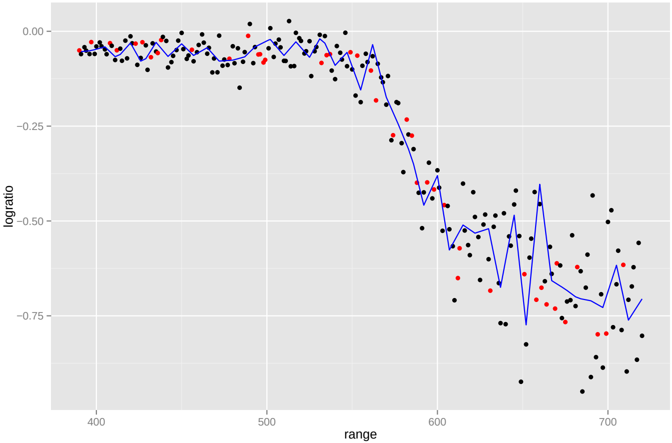
Proposed model

- Use a piecewise linear model
- Allow a lot of “break points” or knots
- (For precision, we’re using a spline model and will talk more about this when we get to scatterplot smoothing)
- Tons of flexibility in the model ... maybe even too much

Proposed model

```
> beta.lm = solve(t(X.train) %*% X.train)%*% t(X.train) %*% y.train
> as.vector(beta.lm)
 [1] -0.7072990330  0.0016678905 -0.0010621060 -0.0038462296  0.0063795230
 [6] -0.0070019621  0.0064736762 -0.0066731019  0.0084198391 -0.0068898605
[11]  0.0025832865 -0.0019203044 -0.0001653192  0.0070482441 -0.0059000564
[16]  0.0040792516 -0.0062804686  0.0166425081 -0.0272267085  0.0175451642
[21] -0.0058689652  0.0015588681 -0.0103188499  0.0250757111 -0.0299488973
[26]  0.0043359248  0.0057320047 -0.0194584632  0.0370714034 -0.0438342911
[31]  0.0321158653  0.0129884439 -0.0215919386 -0.0010528402  0.0381845809
[36] -0.0740150394  0.0675670684 -0.0380309741  0.0052680924  0.0277772008
[41] -0.0304799103  0.0155170138  0.0042768975 -0.0340085357  0.0436823108
```

Full model

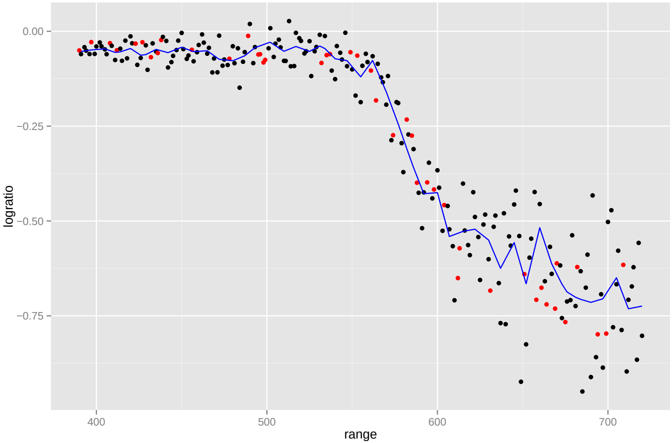


Penalized model

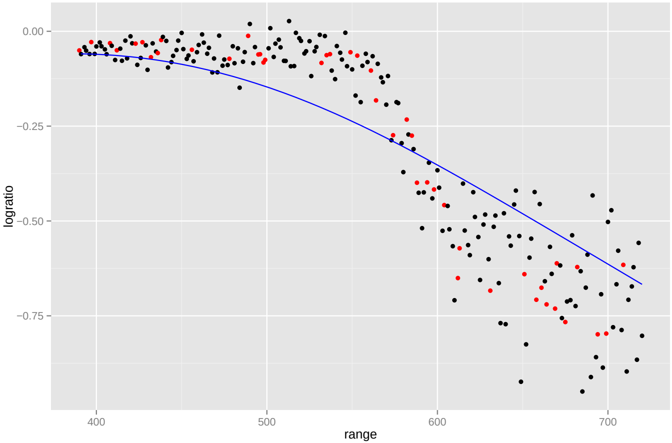
Penalization in this case can help

- Without penalization we're over-fitting, and estimates would change dramatically if we were given a different data set
- Penalization can induce "shrinkage" toward zero for the piecewise linear components
- Too much penalization will result in under-fitting ...

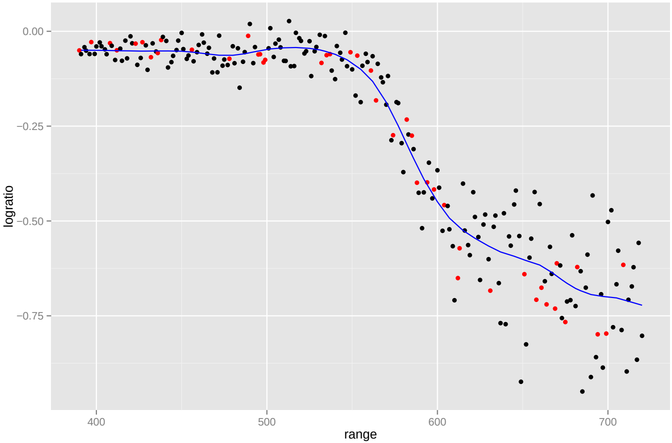
Penalized model



Penalized model



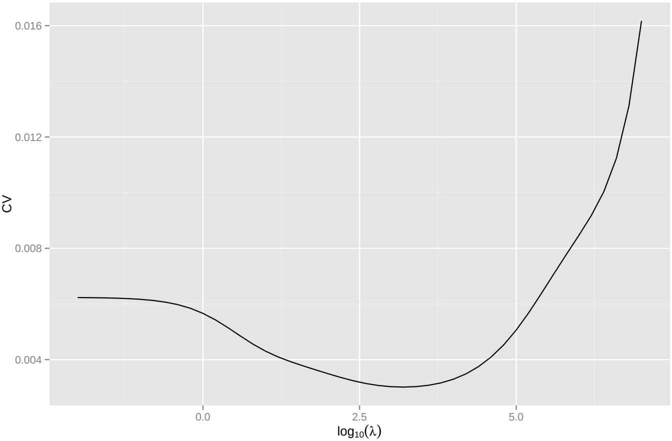
Penalized model



Comparing fits

- Visual inspection of data is useful, but maybe we should look at the validation data as a way to compare model fits
- In effect, we're asking for the impact of λ on our predictions, and trying to identify the "best" tuning parameter
- Our criterion for λ is to minimize $E(\hat{y}_{i,\lambda} - y_i)^2$

CV plot



Connection with other models

- The effects of over- and under-fitting are easiest to understand when you can see them, as in scatterplot smoothing
- These problems exist for other models, though, especially when the number of parameters is high
- In general penalization is a reasonable tool to balance variance and bias to make predictions
- Cross validation is a standard method for selecting tuning parameters

Tradeoffs for spline approaches

Penalization has some advantages and disadvantages

- Less reliant on user input for knot placement
- Overall fit is data-driven
- More complex overall model and estimation
- No interpretable change point

Non-parametric smoothing

- Spline models are parametric smoothing techniques, since

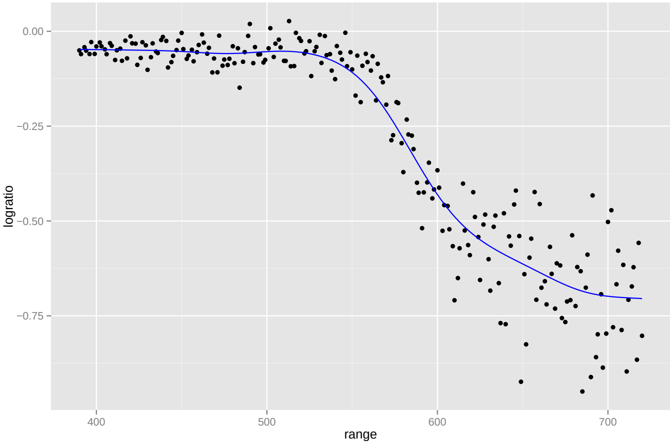
$$E(y|x) = f(x) = f(x; \beta)$$

- (Penalized regression is often called "semi-parametric" since it is an unspecified smooth function)
- There are also non-parametric smoothing approaches

Non-parametric smoothing

- A moving average is the most direct and intuitive non-parametric smoother
- To estimate $f(x_0)$, one looks at all points in a pre-defined range of x_0 and takes the average
- This is repeated for many points in the domain of x
- Can fix this up using weighted averages, where weights often depend on a "kernel" (for example, a Gaussian kernel smoother)

Example: non-parametric smoothing



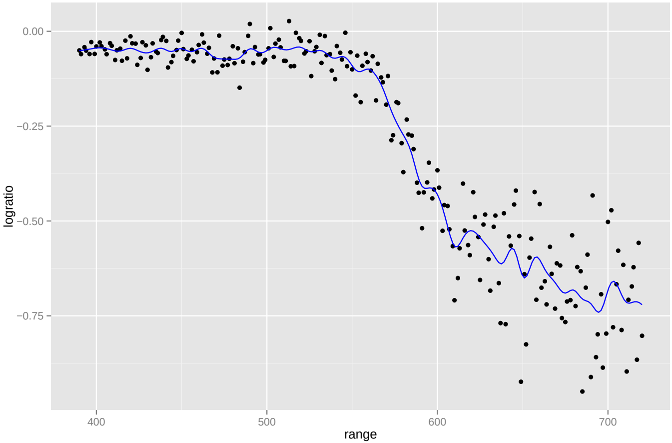
Example: Gaussian kernel smoother

```
> kern.smooth1 = ksmooth(range, y, kernel = "normal", bandwidth = 50)
```

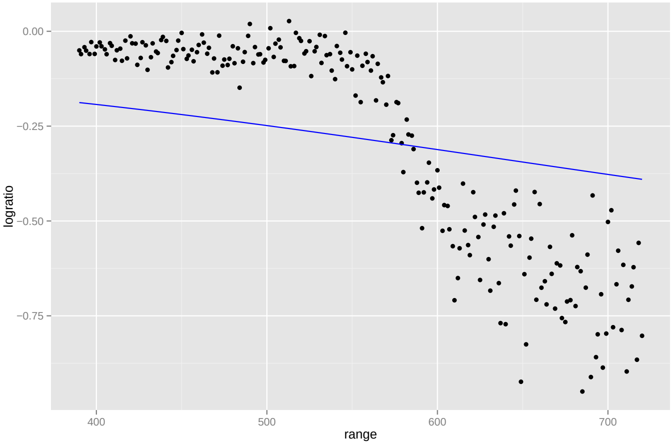
Points on non-parametric smoothing

- Non-parametric approaches are flexible and don't need model assumptions
- They also lack any interpretable distributions or parameters
- Don't avoid tuning parameters – bandwidth still has to be chosen (often via CV)

Spline models



Spline models



Today's big ideas

- Spline models
- Penalized spline regression
- Kernel smoothers