Outline

- Locally weighted averages
- Penalized least squares smoothers
- Properties of smoothers
- CV and the smoothing parameter
Estimating a curve or surface.
The additive statistical model:

Given $n$ pairs of observations $(x_i, y_i), \ i = 1, \ldots, n$

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

$\epsilon_i$'s are random errors
and $g$ is an unknown smooth function.

The goal is to estimate a function $g$ based on the observations.
A 2-d example

Predict mean summer rainfall

Mean rainfall (tenths mm) for JJA. Approximately 1700 stations reporting.
Local linear surface estimates

estimate \( g(x) \)

- A local weighted average of observations close to \( x \)
- Adjustments from covariates – possibly estimated from global information
- Account for measurement error

We will spend several lectures developing what the weights should be!
A kernel estimator

Determine the weights based on the distance to the prediction points

Weight for observation $x_i$ to predict at $x$

BUMP( distance( $x$, $x_i$) )

i.e.

$$w_i \sim (1/h)K(||x - x_i||/h)$$

(and normalize so that the weights sum to one.)

**Kernel: $K$** A bump shaped function e.g. a normal, $K(u) = \exp(-u^2)$

**Bandwidth: $h$** A scale parameter that controls the spread of $K$. As $h$ gets large the estimate is just the average.
The estimator:

\[ g(x) = \frac{\sum K(||x - x_i||/h)Y_i}{\sum K(||x - x_j||/h)} \]

The big idea:
Kernel, covariance model, roughness penalty

Bandwidth, nugget variance \((\sigma)\), smoothing parameter.
Some kernel estimates for rainfall

Data, Kernel smooths with bandwidths (.02,.06,.12)
Some Problems

- How large should the neighborhood/bandwidth be?
- What is the uncertainty of the prediction?
- Predicting in between observations is *ad hoc* and can get weird when the error is small.

But the theoretical properties of kernel estimators are well understood...

\[
E \left[ g(x) - \hat{g}(x) \right]^2 \approx h^4 K_2/4 + \frac{\sigma^2}{nh} K_0
\]

\[MSE = Bias^2 + Variance\]

g has two continuous derivatives, measurement error has variance \(\sigma^2\)
Something different
Penalized least squares

Start with your favorite $m$ basis functions $\{\phi_k\}_{k=1}^n$

The estimate has the form

$$g(x) = \sum_{l=1}^m c_k \phi_k(x)$$

$c = (c_1, \ldots, c_m)$ are the coefficients.

Let $\Phi_{i,k} = \phi_k(x_i)$ so $g = \Phi c$

*Basis functions are fixed and the coefficients will be estimated from data.*
Ridge regression

Sum of squares($c$) + penalty on $c$

Classical ridge regression

$$\min_c \sum_{i=1}^{n} (y - [\Phi c]_i)^2 + \lambda \sum_i c_i^2$$

"shrink the coefficients to zero"

Originally intended to deal with near collinear regression problems.

More general ridge regression

$$\min_c \sum_{i=1}^{n} (y - [\Phi c]_i)^2 + \lambda c^T B c$$

with $\lambda > 0$ a hyperparameter and $B$ a nonnegative definite matrix.

Typically $c^T B c$ measures how different the coefficients are from one another. ... or as we will see penalizes curves that are not very smooth.
The estimator ...

Coefficients:

\[ \hat{c} = (\Phi^T \Phi + \lambda B)^{-1} \Phi^T y \]

Estimated curve:

\[ g(x) = \sum_{l=1}^{m} \hat{c}_k \phi_k(x) \]

Predicted values at data locations:

\[ \hat{g} = \Phi \hat{c} = \Phi (\Phi^T \Phi + \lambda B)^{-1} \Phi^T y = A(\lambda)y \]

The rows of \( A(\lambda) \) are the smoothing weights
An example

Some synthetic data: \( Y_k = h(x_k) + e_k \)

\( x_k \) are 150 unequally spaced points in \([0, 1]\)
\( h(x) = 10x(1 - x) \), \( e_k \sim N(0, (.2)^2) \)
Basis functions are Gaussian bumps

Basis function centers \( \{u_1, u_2, \ldots, u_{20}\} \)

In this example we will use 20 basis functions at equally spaced locations in \([0, 1]\) scaled to overlap two nearest neighbors
The $B$ matrix is less obvious

A good penalty on the coefficients is

$$(c_2 - c_1)^2 + (c_3 - c_2)^2 + \ldots + (c_{20} - c_{19})^2$$

Write the penalty in terms of a difference matrix

$$\begin{pmatrix} c_2 - c_1 \\ c_3 - c_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} -1 & 1 & 0 & 0 & \cdots \\ 0 & -1 & 1 & 0 & \cdots \\ \vdots & & & & \ddots \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \end{pmatrix} = Ac$$

Now use this in the sum of squares:

$$(c_2 - c_1)^2 + (c_3 - c_2)^2 + \ldots + (c_{NB} - c_{NB-1})^2 = (Ac)^T Ac = c^T (A^T A) c = c^T (B) c$$
Curve estimates at a few different lambda values

lambda = 0.001  trace 19.97

lambda = 1  trace 13.13

lambda = 10  trace 7.89

lambda = 100  trace 3.88

Which is the best choice? How do you know?
A penalty that is on $g$ not $c$

Because the estimate is a linear function of the coefficients we can also set up a penalty directly on $g$.

A reasonable penalty used in numerical analysis is the integrated squared derivative.

e.g.

$$\int \left( \frac{d}{dx} g(x) \right)^2 \, dx$$

Use the basis function form for $g$:

$$\int \left( \frac{d}{dx} g(x) \right)^2 \, dx = \int \left( \frac{d}{dx} \sum_{k=1}^{M} \phi_k(x)c_j \right)^2 \, dx = \int \left( \sum_{k=1}^{M} \phi'_k(x)c_k \right)^2 \, dx$$
Identifying the $B$ matrix

Expand the derivative penalty so that the coefficients and $H$ are recognized.

$$
\int \left( \sum_{k=1}^{M} \phi_k'(x)c_k \right)^2 dx = \int \left( \sum_{k,j=1}^{M} \phi_k'(x)c_kb_j'(x)c_j \right) dx
$$

Bring the integral inside the sum

$$
\sum_{k,j=1}^{M} \left( \int \phi_k'(x)\phi_j'(x)dx \right)c_k c_j = \sum_{k,j=1}^{M} B_{k,j} c_k c_j
$$

$$
B_{k,j} = \int \phi_k'(x)\phi_j'(x)dx
$$

It is possible to work out the integrals in closed form. Not much fun to do ...
Results with derivative penalty

\[
\begin{align*}
\lambda = 0.1 & \quad \text{trace } 19.24 \\
\lambda = 1 & \quad \text{trace } 15.24 \\
\lambda = 5 & \quad \text{trace } 9.97 \\
\lambda = 50 & \quad \text{trace } 4.15
\end{align*}
\]
Comparing the two penalty matrices

squared differences of coefficients

integrated squared derivative
Linear smoothers

Let \( \hat{g} = g(x_1), ..., g(x_n) \) be the prediction vector at the observed points.

A smoother matrix satisfies \( \hat{g} = Ay \) where

- \( A \) is an \( n \times n \) matrix
- eigenvalues of \( A \) are in the range \([0,1]\).

Note: \( ||Ay|| \leq ||y|| \)

Usually values in between the data are filled in by interpolating the predictions at the observations.
The form of the smoother matrix

The monster ...

\[ \hat{c} = (\Phi^T \Phi + \lambda B)^{-1} \Phi^T y \]

\[ \hat{g} = \Phi \hat{c} = \Phi (\Phi^T \Phi + \lambda B)^{-1} \Phi^T y = A(\lambda) y \]

Why is this a smoothing matrix?

\[ A(\lambda) = \Phi (\Phi^T \Phi + \lambda B)^{-1} \Phi^T \]
Effective degrees of freedom

For linear regression trace $A(\lambda)$ gives us the number of parameters. (Because it is a projection matrix)

$\text{trace} \left( \Phi(\Phi^T\Phi)^{-1}\Phi^T \right) = \text{rank of } \Phi$

By analogy, $\text{tr}A(\lambda)$ is measure of the effective number of degrees of freedom attributed to the smooth surface
A useful decomposition

One can always find an orthogonal matrix, $U$ and a non-negative diagonal matrix $\Gamma$

$$A(\lambda) = U(I + \lambda \Gamma)^{-1}U^T$$

A simple formula for the trace

So

$$\text{tr}A(\lambda) = \text{tr}U(I + \lambda \Gamma)^{-1}U^T = \text{tr}(I + \lambda \Gamma)^{-1}U^TU$$

$$= \text{tr}(I + \lambda \Gamma)^{-1} = \sum_{i=1}^{n} \frac{1}{1 + \lambda \Gamma_{ii}}$$

The Gamma’s are all nonnegative so this must be increasing as $\lambda$ decreases.

The relationship is one-to-one with $\lambda$ and independent of the data so we can always talk about $\lambda$ in terms of the effective degrees of freedom.
$\lambda$ and $\text{tr } A(\lambda)$
Choosing $\lambda$ by Cross-validation

Sequentially leave each observation out and predict it using the rest of the data. Find the $\lambda$ that gives the best out of sample predictions.

Refitting the spline when each data point is omitted, and for a grid of $\lambda$ values is computationally demanding.

Fortunately there is a shortcut.
The magic CV formula

leave-one-out CV

\( \hat{g}_{-i} \) the prediction at \( x_i \) having omitted \( u_i \) from the data set.

For penalized least squares

residual for \( g(x_i) \) having omitted \( y_i \)

\[
(y_i - \hat{g}_{-i}) = (y_i - \hat{g}_i)/(1 - A(\lambda))_{i,i}
\]

Called the CV residual. This has a simple form because adding the data pair \((x_i, \hat{g}_{-1})\) to the data does not change the sums of squares.
CV and Generalized CV criterion

\[ CV(\lambda) \]

\[
\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{g}_i)^2 = \frac{1}{n} \sum_{i=1}^{n} \frac{(y_i - \hat{g}_i)^2}{(1 - A(\lambda)_{i,i})^2}
\]

\[ GCV(\lambda) \quad A(\lambda)_{i,i} \approx \text{tr}A(\lambda)/n \]

\[
\left(\frac{1}{n}\right) \frac{\sum_{i=1}^{n} (y_i - \hat{g}_i)^2}{(1 - \text{tr}A(\lambda)/n)^2}
\]

Minimize CV or GCV over \( \lambda \) to determine a good value
Results for GCV

Cross validation

lambda

Trace A
The GCV estimate
GCV estimate using derivative penalty

Effective degrees of freedom is 15.3
Summary

We have formulated the curve/surface fitting problem as penalized least squares.

Splines treat estimating the entire curve but also have a finite basis related to a covariance function (reproducing kernel).

One can use CV or GCV to find the smoothing parameter.

Thank you!