Model Free Curve Fitting

- Response variable: \( Y \)
- Explanatory variables
  \[ X_1, X_2, \ldots, X_p \]
- Summarize or describe trends in the conditional mean of \( Y \).
  - No model is specified
  - Fit a “smooth” curve

Applications

- Fitting a smooth curve to a plot can be a first step in building a parametric model
  - Roughly determine the “shape” of the curve
  - Little subject matter motivation
  - No need to specify a parametric formula
    \[ Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \epsilon_i \]
  - Let the data speak for themselves.

- Check the fit of a parametric model

- Make predictions (interpolation)
  \[
  \begin{array}{ll}
  X_1 & Y_1 \\
  X_2 & Y_2 \\
  \vdots & \vdots \\
  X_k & Y_k \\
  \end{array}
  \]
  Use linear interpolation?

- Extrapolation?

Diabetes data: (Sockett, et al. 1987)

- Factors affecting patterns of insulin-dependent diabetes mellitus in children.
  - Level of serum C-peptide at diagnosis
    \[ Y = \log \text{(serum C-peptide conc.)} \]
  - \( X \) = age (in years) at diagnosis.
C-peptide Concentrations

<table>
<thead>
<tr>
<th>subject</th>
<th>age (years)</th>
<th>base (def)</th>
<th>C-peptide</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10.4</td>
<td>-29.0</td>
<td>5.0</td>
<td>1.609</td>
</tr>
<tr>
<td>3</td>
<td>10.5</td>
<td>-0.9</td>
<td>5.2</td>
<td>1.649</td>
</tr>
<tr>
<td>4</td>
<td>10.6</td>
<td>-7.8</td>
<td>5.5</td>
<td>1.705</td>
</tr>
<tr>
<td>19</td>
<td>10.6</td>
<td>-11.2</td>
<td>4.5</td>
<td>1.504</td>
</tr>
<tr>
<td>43</td>
<td>10.8</td>
<td>-13.5</td>
<td>5.1</td>
<td>1.629</td>
</tr>
<tr>
<td>31</td>
<td>11.0</td>
<td>-14.3</td>
<td>4.4</td>
<td>1.482</td>
</tr>
<tr>
<td>21</td>
<td>11.1</td>
<td>-6.1</td>
<td>4.7</td>
<td>1.543</td>
</tr>
<tr>
<td>33</td>
<td>11.1</td>
<td>-16.8</td>
<td>5.1</td>
<td>1.629</td>
</tr>
<tr>
<td>23</td>
<td>11.3</td>
<td>-3.6</td>
<td>5.1</td>
<td>1.629</td>
</tr>
<tr>
<td>18</td>
<td>11.5</td>
<td>-9.0</td>
<td>5.5</td>
<td>1.705</td>
</tr>
<tr>
<td>16</td>
<td>11.8</td>
<td>-2.1</td>
<td>4.6</td>
<td>1.526</td>
</tr>
<tr>
<td>26</td>
<td>11.9</td>
<td>-2.0</td>
<td>5.1</td>
<td>1.629</td>
</tr>
<tr>
<td>32</td>
<td>12.4</td>
<td>-0.8</td>
<td>5.2</td>
<td>1.649</td>
</tr>
<tr>
<td>40</td>
<td>12.5</td>
<td>-13.6</td>
<td>4.1</td>
<td>1.411</td>
</tr>
<tr>
<td>7</td>
<td>12.7</td>
<td>-18.9</td>
<td>3.4</td>
<td>1.224</td>
</tr>
<tr>
<td>22</td>
<td>12.8</td>
<td>-1.0</td>
<td>6.6</td>
<td>1.887</td>
</tr>
<tr>
<td>38</td>
<td>13.2</td>
<td>-0.7</td>
<td>6.0</td>
<td>1.792</td>
</tr>
<tr>
<td>41</td>
<td>13.2</td>
<td>-1.9</td>
<td>4.6</td>
<td>1.526</td>
</tr>
<tr>
<td>28</td>
<td>13.8</td>
<td>-11.9</td>
<td>3.7</td>
<td>1.308</td>
</tr>
<tr>
<td>25</td>
<td>14.5</td>
<td>-0.5</td>
<td>5.7</td>
<td>1.740</td>
</tr>
<tr>
<td>29</td>
<td>15.5</td>
<td>-0.7</td>
<td>4.9</td>
<td>1.589</td>
</tr>
<tr>
<td>8</td>
<td>15.6</td>
<td>-10.6</td>
<td>4.9</td>
<td>1.589</td>
</tr>
</tbody>
</table>
# This is SPLUS code for plotting
# log(C-peptide concentration)
# against age. This file stored as
# cpeptide1.spl

# The data are stored in the file
# cpeptide.tex

# There are four numbers on each line
# in the following order:
# Subject identification code
# Age at diagnosis (years)
# Base deficit (measure of acidity)
# C-peptide concentration (pmol/ml)
# Enter the data into a data frame
# Compute the natural log of the
# C-peptide concentration.

cpep <- read.table("cpeptide.tex", header=T)
cpep$Y <- log(cpep$Peptide)
cpep$Y <- round(cpep$Y, digits=3)

# Sort the data file by age
i <- sort.list(cpep$age)
cpep <- cpep[i,]
cpep

# Code for plotting weight against time
# Specify plotting symbol and size of
# graph in inches.
# fin=c(w,h) specifies a plot that is w
# inches wide and h inches high.
# pch=18 requests a filled diamond as a
# plotting symbol.
# mkh=b requests plotting symbols that
# are b inches high.
# mex=a sets the spacing between lines
# printed in the margins.
# plt plt=c(.2,.8,.2,.8) defines the
# fraction of figure region to use
# for plotting. This can provide
# more space for to label margins.
par(fin=c(7.0,7.0),pch=18,mkh=.1,me=1.5, plt=c(.2,.8,.2,.8))
plot(cpep$age, cpep$Y, type="p",
 xlab="Age (years)",
ylab="log(concentration)",
main="C-peptide Concentrations")

# The following three lines are for adding
# an axis for C-peptide concentration on
# the original scale (pmol/ml).
# pretty(): Returns a vector of ordered
# and equally spaced values that span
# the range of the input.

Y.exp <- pretty(range(exp(cpep$Y)))
axis(side=4, at=log(Y.exp),
 xlab=Y.exp, srt=90)
mtext("Concentration (pmol/ml)",
 side=4, line=3)

# Fit a straight line model
cpep.lin <- lm(Y~age,data=cpep)

par(fin=c(7.0,7.0),pch=18,mkh=.1,me=1.5, plt=c(.2,.8,.2,.8))
plot(cpep$age, cpep$Y, type="p",
 xlab="Age (years)",
ylab="log(concentration)",
main="C-peptide Concentrations")
a <- seq(1,16,.5)
lines(a, predict(cpep.lin, data.frame(age=a),
 type="response"),lty=1,lwd=3)

# Fit a quadratic model
cpep.q <- lm(Y~age+age^2,data=cpep)

par(fin=c(7.0,7.0),pch=18,mkh=.1,me=1.5, plt=c(.2,.8,.2,.8))
plot(cpep$age, cpep$Y, type="p",
 xlab="Age (years)",
ylab="log(concentration)",
main="C-peptide Concentrations")
a <- seq(1,16,.5)
lines(a, predict(cpep.q, data.frame(age=a),
 type="response"),lty=1,lwd=3)

# Fit a cubic model
cpep.3 <- lm(Y~age+age^2+age^3,data=cpep)

par(fin=c(7.0,7.0),pch=18,mkh=.1,me=1.5, plt=c(.2,.8,.2,.8))
plot(cpep$age, cpep$Y, type="p",
 xlab="Age (years)",
ylab="log(concentration)",
main="C-peptide Concentrations")
a <- seq(1,16,.5)
lines(a, predict(cpep.3, data.frame(age=a),
 type="response"),lty=1,lwd=3)
“Bin” Smoothers:

- Partition the range of the explanatory variable ($X$) into $p$ disjoint and exhaustive regions
- About the same number of observations in each “bin”
- Compute the average of the responses ($Y$ values) in each bin

Running mean or median smoothers

- Use a different “bin” for each value of the explanatory variable $X$
- Symmetric nearest neighbor version: Find the nearest $k$ cases to the left of $X$ and the nearest $k$ cases to the right of $X$
  - Compute the mean (or median)
  - Include $X$?
  - Boundary considerations
- Nearest neighbor version: Use the $r$ nearest cases to $X$

Running Line Smoothers

- Fit a least squares regression line to the points “near” $X$
  - Symmetric nearest neighbors
  - Nearest neighbors
- Predict the mean response at $X$
  \[
  \hat{Y}_X = b_{0,X} + b_{1,X}X
  \]
  - The estimated coefficients will not be the same for every $X$
  (local regression lines)
- parbox[t]6.0in Using larger neighborhoods produces smoother curves.
Running Line Smoothers

- In the center of the data
  - the intercept is dominant
  - the slope plays a smaller role
- Near the edges (boundaries)
  - Slope is important for picking up trends in asymmetric neighborhoods of $X$.
  - This reduces some of the "bias" associated with running means.

Points inside a neighborhood have equal weight.
- points "outside" have zero weight
- source of jaggedness
- "weight" the points in a neighborhood.
  - higher weights for points closer to $X$.
  - weights go to zero near the ends of the neighborhood.
- Cleveland's "loess" smoother

Kernel Smoothers

- Local weighted average with local weights defined by a "kernel".
  \[
  \hat{Y}_i = \frac{\sum_{j=1}^{n} Y_j K \left( \frac{X_j - X}{b} \right)}{\sum_{j=1}^{n} K \left( \frac{X_j - X}{b} \right)}
  \]
  - the value of $K \left( \frac{X_j - X}{b} \right)$ decreases in a "smooth" way as $X_j$ moves farther away from $X$.
  - $b$ is the "bandwidth".

Examples:

"Gaussian" kernel smoother
\[
K \left( \frac{X_j - X}{b} \right) = \frac{1}{\sqrt{2\pi}b} e^{-\frac{1}{2} \left( \frac{X_j - X}{b} \right)^2}
\]

"Minimum variance" kernel
\[
K \left( \frac{X_j - X}{b} \right) = \begin{cases} 
\frac{3}{35} (3 - 5[\frac{X_j - X}{b}]^2) & \text{if } |\frac{X_j - X}{b}| < 1 \\
0 & \text{otherwise}
\end{cases}
\]

This choice of weights minimizes the large sample variance of the estimator.
Locally Weighted Running Line Smoothers (loess)

Data:
\[(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\]

Objective:
Estimate the conditional means of \( Y \) at a set of \( X \) values.
- Use cases in a neighborhood of \( X \)
- Fit a regression model
- Use weighted least squares estimation

(Step 1) Identify the \( k \) observations with \( X_j \) values closest to \( X \)

Identify this set of \( k \) nearest neighbors as \( N_k(X) \).

(Step 2) Compute the distance of the farthest near neighbor
\[\Delta_k(X) = \max_{X_j \in N_k(X)} |X - X_j|\]

(Step 3) Assign weights to each of the “near” neighbors using the tricube weight function
\[W_j = W\left(\frac{|X - X_j|}{\Delta_k(X)}\right),\]
where
\[W(u) = \begin{cases} (1 - u^3)^3, & 0 \leq u < 1 \\ 0, & \text{otherwise} \end{cases}\]
(Step 4) Fit a regression line using weighted least squares.

Find $a_X$ and $b_X$ to minimize

$$\sum_{j=1}^{n} W_j(Y_j - a_X - b_X X_j)^2$$

Solution:

$$b_X = \frac{\sum_{j=1}^{n} W_j(X_j - \bar{X})(Y_j - \bar{Y})}{\sum_{j=1}^{n} W_j(X_j - \bar{X})^2}$$

$$a_X = \bar{Y} - b_X \bar{X}$$

(Step 5) Predict at $X$:

$$\hat{Y}_X = a_X + b_X(X)$$

and record $(X, \hat{Y}_X)$

Repeat Steps 1 to 5 for a series of $X$ values:

- You could fit local polynomial regression curves.
  $$\hat{Y}_X = a_X + b_X X + c_X X^2$$

- You could replace the tri-cube weight function.

- The size of $N_k(X)$ is important.
How wide should your local neighborhoods be?
- **Small**
  - curve is less smooth (increased variability)
  - react to local changes (reduce bias)
- **Large**
  - curve is smoother (less variability)
  - may "smooth out" local patterns (more bias).

```r
> # Compare the loess curves with different spans
> cpep.lo100 <- loess(formula=Y ~ age, data=cpep, span=1.00, degree=1)
> cpep.lo75  <- loess(formula=Y ~ age, data=cpep, span=0.75, degree=1)
> cpep.lo25  <- loess(formula=Y ~ age, data=cpep, span=0.25, degree=1)
> anova(cpep.lo100, cpep.lo75, cpep.lo25)

Model 1:
loess(formula = Y ~ age, data = cpep,
      span = 1, degree = 1)
Model 2:
loess(formula = Y ~ age, data = cpep,
      span = 0.75, degree = 1)
Model 3:
loess(formula = Y ~ age, data = cpep,
      span = 0.25, degree = 1)
```

### Analysis of Variance Table

<table>
<thead>
<tr>
<th>ENP</th>
<th>RSS</th>
<th>Test</th>
<th>F Value</th>
<th>Pr(F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.3</td>
<td>0.72033</td>
<td>1 vs 2</td>
<td>2.88 0.098068</td>
</tr>
<tr>
<td>2</td>
<td>2.9</td>
<td>0.66027</td>
<td>2 vs 3</td>
<td>0.31 0.951810</td>
</tr>
<tr>
<td>3</td>
<td>8.5</td>
<td>0.61296</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```r
# Plot residuals
scatter.smooth(fitted(cpep.lo100),
residuals(cpep.lo100), span=1, degree=1)
scatter.smooth(fitted(cpep.lo75),
residuals(cpep.lo75), span=1, degree=1)
scatter.smooth(fitted(cpep.lo25),
residuals(cpep.lo25), span=1, degree=1)
qqnorm(residuals(cpep.lo75))
qqline(residuals(cpep.lo75))
```
# Consider a second degree polynomial smoother

```r
par(fin=c(7.0,7.0),pch=18,mgp=.1,mex=.5,
     plt=c(.2,.8,.2,.8))
plot(cpep$age, cpep$Y, type="p",
     xlab="Age (years)",
     ylab="log(concentration)",
     main="C-peptide Concentrations 
     n Loess Curves")
lines(cpep$age, loess(formula=Y~age, data=cpep,
                        span=.75,degree=1)$fitted.values, lty=1,lwd=3)
lines(cpep$age, loess(formula=Y~age, data=cpep,
                        span=.75,degree=2)$fitted.values, lty=3,lwd=3)
legend(5,1.31,c("degree=1.0", "degree=2"),
        lty=c(1,3),bty="n")
```
cpep.lo751 <- loess(formula=Y ~ age, data=cpep, span=.75, degree=1)
cpep.lo752 <- loess(formula=Y ~ age, data=cpep, span=.75, degree=2)

anova(cpep.lo751, cpep.lo752)

Model 1:
loess(formula = Y ~ age, data = cpep, span = 0.75, degree = 1)

Model 2:
loess(formula = Y ~ age, data = cpep, span = 0.75, degree = 2)

Analysis of Variance Table
     ENP RSS Test F Value Pr( F)
1 2.9 0.66027 1 vs 2 1.54 0.22873
2 4.6 0.61928

# Plot residuals
scatter.smooth(fitted(cpep.lo752), residuals(cpep.lo752), span=1, degree=1)
qqnorm(residuals(cpep.lo752))
qqline(residuals(cpep.lo752))
# Make predictions using the predict( ) and
# compute pointwise 95% confidence intervals

cpep.se <- predict(cpep.lo752, seq(1,15,1),
                   se.fit=T)
cpep.locl <- pointwise(cpep.se, coverage=.95)
cpep.locl

$upper:
  [1] 1.350755 1.419861 1.496769 1.570916 1.630554

$fit:
  [1] 1.211487 1.323710 1.417012 1.491429 1.547887

$lower:
  [1] 1.072218 1.227558 1.337255 1.411942 1.465219

plot(cpep.lo752, confidence=15,coverage=0.95,
     ylim=c(1.0,1.8))

# Plot the curve with approximate pointwise
# confidence limits

par(fin=c(7.0,7.0),pch=18,mkh=.001,mar=1.5,
     plt=c(.2,.8,.2,.8))

plot(cpep$age, cpep$Y, type="n", xlab="Age (years)",
ylab="log(concentration)"
     main="C-peptide Concentrations
\nLocal Quadratic Loess Smoother")

lines(smooth.spline(cpep.locl$x, cpep.locl$fit),
      lty=1,lwd=3)
lines(smooth.spline(cpep.locl$x, cpep.locl$upper ),
      lty=3,lwd=3)
lines(smooth.spline(cpep.locl$x, cpep.locl$lower ),
      lty=3,lwd=3)

C-peptide Concentrations
Local Quadratic Loess Smoother

log(concentration)

Age (years)