

Model Free Curve Fitting

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

1

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.

2

- Check the fit of a parametric model

- Make predictions (interpolation)

Must store $\begin{matrix} X_1 & \hat{Y}_1 \\ X_2 & \hat{Y}_2 \\ \vdots & \vdots \\ X_k & \hat{Y}_k \end{matrix}$

Use linear interpolation?

- Extrapolation?

3

Diabetes data: (Socket, et al. 1987)

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

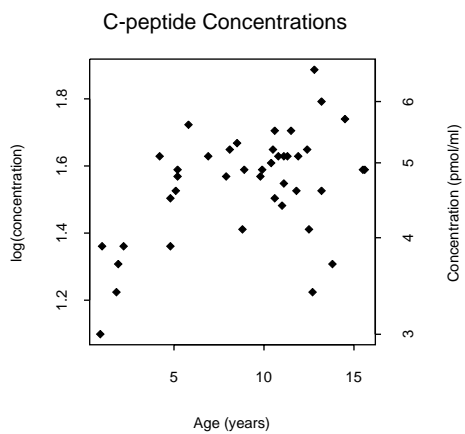
4

subject	age	basedef	Cpeptide	Y
15	0.9	-11.6	3.0	1.099
24	1.0	-8.2	3.9	1.361
6	1.8	-19.2	3.4	1.224
10	1.9	-25.0	3.7	1.308
11	2.2	-3.1	3.9	1.361
36	4.2	-17.0	5.1	1.629
12	4.8	-7.8	4.5	1.504
35	4.8	-9.5	3.9	1.361
34	5.1	-5.1	4.6	1.526
1	5.2	-8.1	4.8	1.569
14	5.2	-4.5	4.9	1.589
9	5.8	-2.8	5.6	1.723
37	6.9	-3.3	5.1	1.629
13	7.9	-13.9	4.8	1.569
17	7.9	-2.0	4.8	1.569
27	8.1	-1.6	5.2	1.649
20	8.5	-0.2	5.3	1.668
2	8.8	-16.1	4.1	1.411
42	8.9	-10.0	4.9	1.589
30	9.8	-1.2	4.8	1.569
39	9.9	-3.3	4.9	1.589

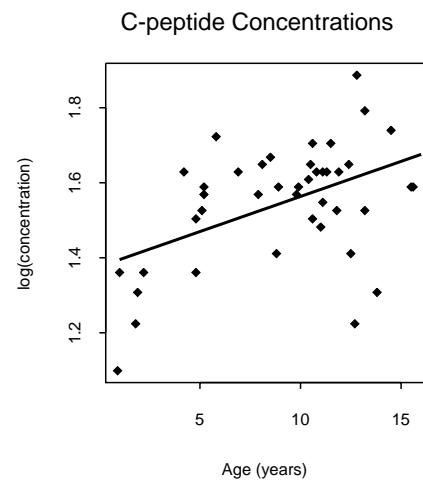
5

subject	age	basedef	cpeptide	Y
5	10.4	-29.0	5.0	1.609
3	10.5	-0.9	5.2	1.649
4	10.6	-7.8	5.5	1.705
19	10.6	-11.2	4.5	1.504
43	10.8	-13.5	5.1	1.629
31	11.0	-14.3	4.4	1.482
21	11.1	-6.1	4.7	1.548
33	11.1	-16.8	5.1	1.629
23	11.3	-3.6	5.1	1.629
18	11.5	-9.0	5.5	1.705
16	11.8	-2.1	4.6	1.526
26	11.9	-2.0	5.1	1.629
32	12.4	-0.8	5.2	1.649
40	12.5	-13.6	4.1	1.411
7	12.7	-18.9	3.4	1.224
22	12.8	-1.0	6.6	1.887
38	13.2	-0.7	6.0	1.792
41	13.2	-1.9	4.6	1.526
28	13.8	-11.9	3.7	1.308
25	14.5	-0.5	5.7	1.740
29	15.5	-0.7	4.9	1.589
8	15.6	-10.6	4.9	1.589

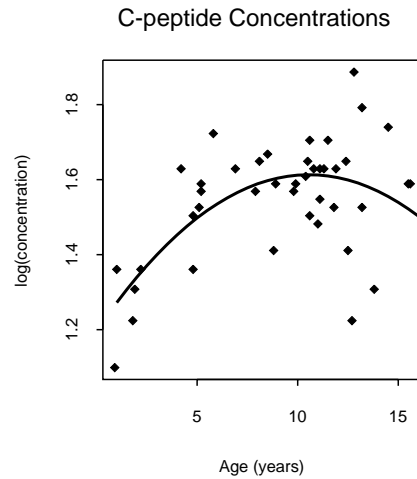
6



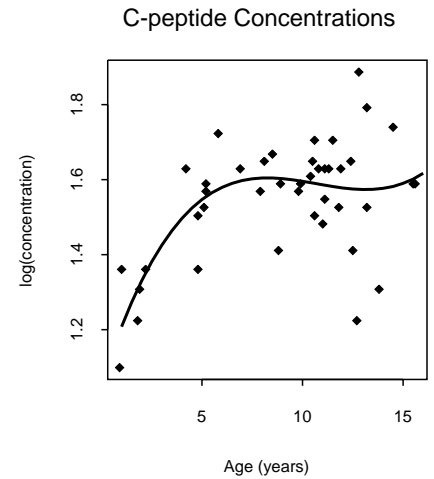
7



8



9



10

```
# 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$Y <- log(cpep$Cpeptide)
cpep$Y <- round(cpep$Y,digits=3)
```

11

```
# 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.
```

12

```

par(fin=c(7.0,7.0),pch=18,mkh=.1,mex=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),
     lab=Y.exp, srt=90)
mtext("Concentration (pmol/ml)",
     side=4, line=3)

```

13

```

# Fit a straight line model

cpep.lin <- lm(Y~age,data=cpep)

par(fin=c(7.0,7.0),pch=18,mkh=.1,mex=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)

```

14

```

# Fit a quadratic model

cpep.q <- lm(Y~age+age^2,data=cpep)

par(fin=c(7.0,7.0),pch=18,mkh=.1,mex=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)

```

15

```

# 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,mex=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)

```

16

“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

17

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

18

Running mean or median smoothers

- Simple to compute
- May not be smooth enough
- Tend to flatten out trends near the boundaries)

19

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)

- Using larger neighborhoods produces smoother curves.

20

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.

21

- 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

22

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” .

23

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

$$K\left(\frac{X_j - X}{b}\right) = \begin{cases} \frac{3}{8b} (3 - 5\left[\frac{X_j - X}{b}\right]^2) & \text{if } \left|\frac{X_j - X}{b}\right| < 1 \\ 0 & \text{otherwise} \end{cases}$$

This choice of weights minimizes the large sample variance of the estimator.

24

- The “kernel” is truncated at the end of the data.
- Simulation studies have shown that
 - the choice of the form of the kernel is not very important.
 - the “bandwidth” is important.

25

Locally Weighted Running Line Smoothers (loess)

Data:

$$\begin{pmatrix} X_1 & Y_1 \\ X_2 & Y_2 \\ \vdots & \vdots \\ X_n & Y_n \end{pmatrix}$$

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

26

(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

$$\Delta_K(X) = \max_{X_j \in N_K(X)} |X - X_j|$$

27

(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

$$W(u) = \begin{cases} (1 - u^3)^3, & 0 \leq u < 1 \\ 0, & \text{otherwise} \end{cases}$$

28

(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:

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

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

29

where

$$\bar{X} = \frac{\sum_{j=1}^n W_j X_j}{\sum_{j=1}^n W_j}$$

$$\bar{Y}_X = \frac{\sum_{j=1}^n W_j Y_j}{\sum_{j=1}^n W_j}$$

(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:

30

- 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.

31

C-peptide Concentrations
Loess Curves



32

How wide should your local neighborhoods be?

- Small
 - curve is less smooth (increase variability)
 - react to local changes (reduce bias)
- Large
 - curve is smoother (less variability)
 - may “smooth out” local patterns (more bias).

33

```
> # 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=.75,degree=1)
> cpep.lo25 <- loess(formula=Y~age,
                      data=cpep,span=.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)
```

34

Analysis of Variance Table

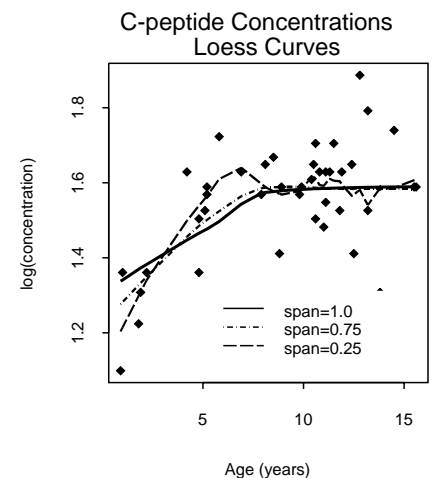
	ENP	RSS	Test	F Value	Pr(F)
1	2.3	0.72033	1 vs 2	2.88	0.098068
2	2.9	0.66027	2 vs 3	0.31	0.951810
3	8.5	0.61296			

Plot residuals

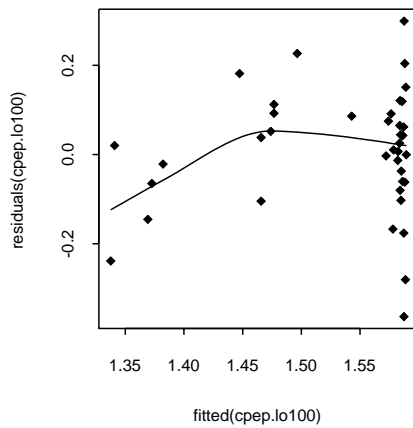
```
scatter.smooth(fitted(cpep.lo100),
              residuals(cpep.lo100), span=1, degree=1)
scatter.smooth(fitted(cpep.lo75),
              residuals(cpep.lo75), span=1, degree=1)
scatter.smooth(fitted(cpep.lo25),
              residuals(cpep.lo25), span=1, degree=1)
```

```
qqnorm(residuals(cpep.lo75))
qqline(residuals(cpep.lo75))
```

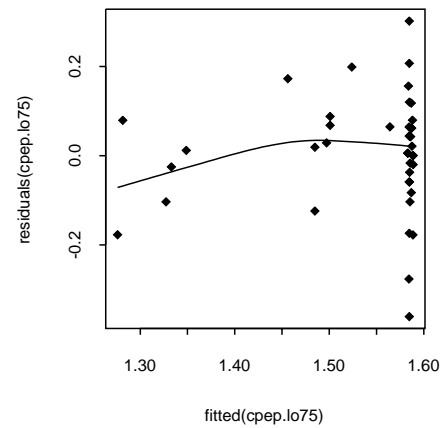
35



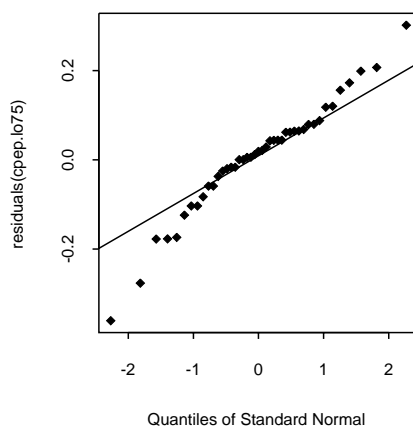
36



37



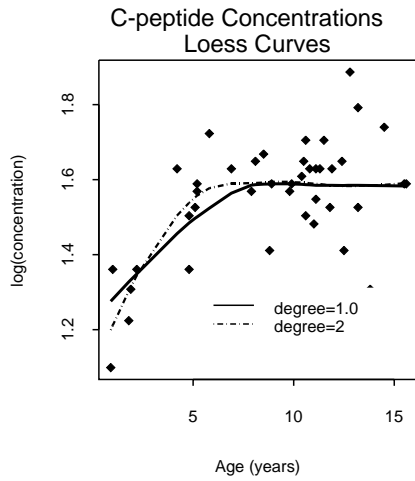
38



39

```
# Consider a second degree polynomial smoother
par(fin=c(7.0,7.0),pch=18,mkh=.1,mex=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 \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")
```

40



41

```
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

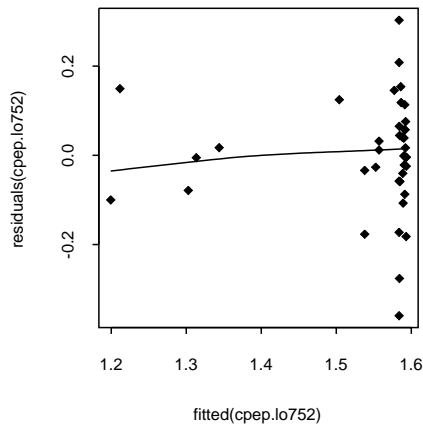
	ENP	RSS	Test	F Value	Pr(F)
1	2.9	0.66027	1 vs 2	1.54	0.22873
2	4.6	0.61928			

```
# Plot residuals

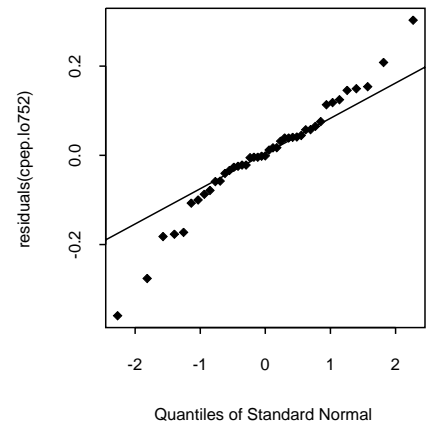
scatter.smooth(fitted(cpep.lo752),
  residuals(cpep.lo752), span=1, degree=1)

qqnorm(residuals(cpep.lo752))
qqline(residuals(cpep.lo752))
```

42



43



44

```

# 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
 [6] 1.661133 1.671951 1.682351 1.668795 1.659994
[11] 1.652856 1.647018 1.648919 1.665167 1.710045

$fit:
 [1] 1.211487 1.323710 1.417012 1.491429 1.547887
 [6] 1.581510 1.590187 1.590938 1.593441 1.593186
[11] 1.589101 1.584477 1.583726 1.584747 1.587834

$lower:
 [1] 1.072218 1.227558 1.337255 1.411942 1.465219
 [6] 1.501887 1.508422 1.499525 1.518086 1.526378
[11] 1.525347 1.521935 1.518533 1.504327 1.465624

```

45

```

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

```

```

# Plot the curve with approximate pointwise
# confidence limits

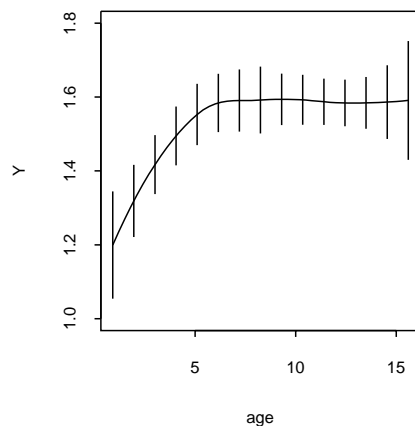
```

```

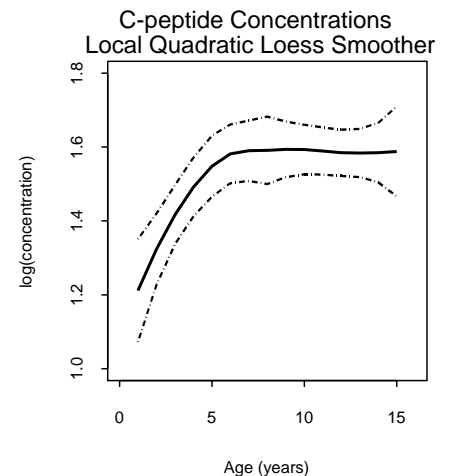
par(fin=c(7.0,7.0),pch=18,mkh=.001,mex=1.5,
    plt=c(.2,.8,.2,.8))
plot(cpep$age, cpep$Y, type="n", xlim=c(0,16),
     ylim=c(1.0,1.8), xlab="Age (years)",
     ylab="log(concentration)",
     main="C-peptide Concentrations
\n Local Quadratic Loess Smoother")
lines(smooth.spline(cpep.locl$x, cpep.locl$fit),
      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)

```

46



47



48