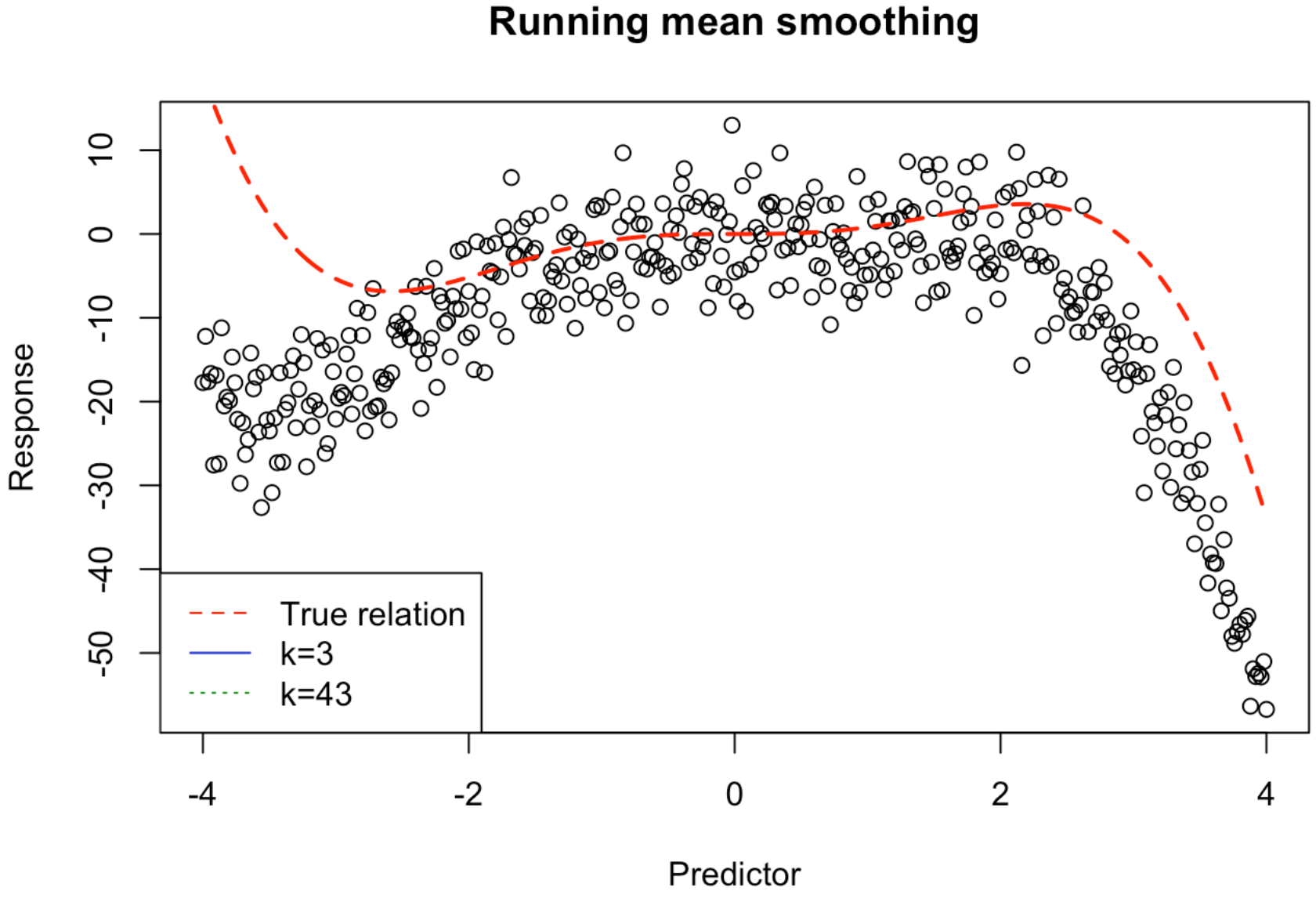
tutorial_w4

Likun Cui(470195873)

28 August, 2018

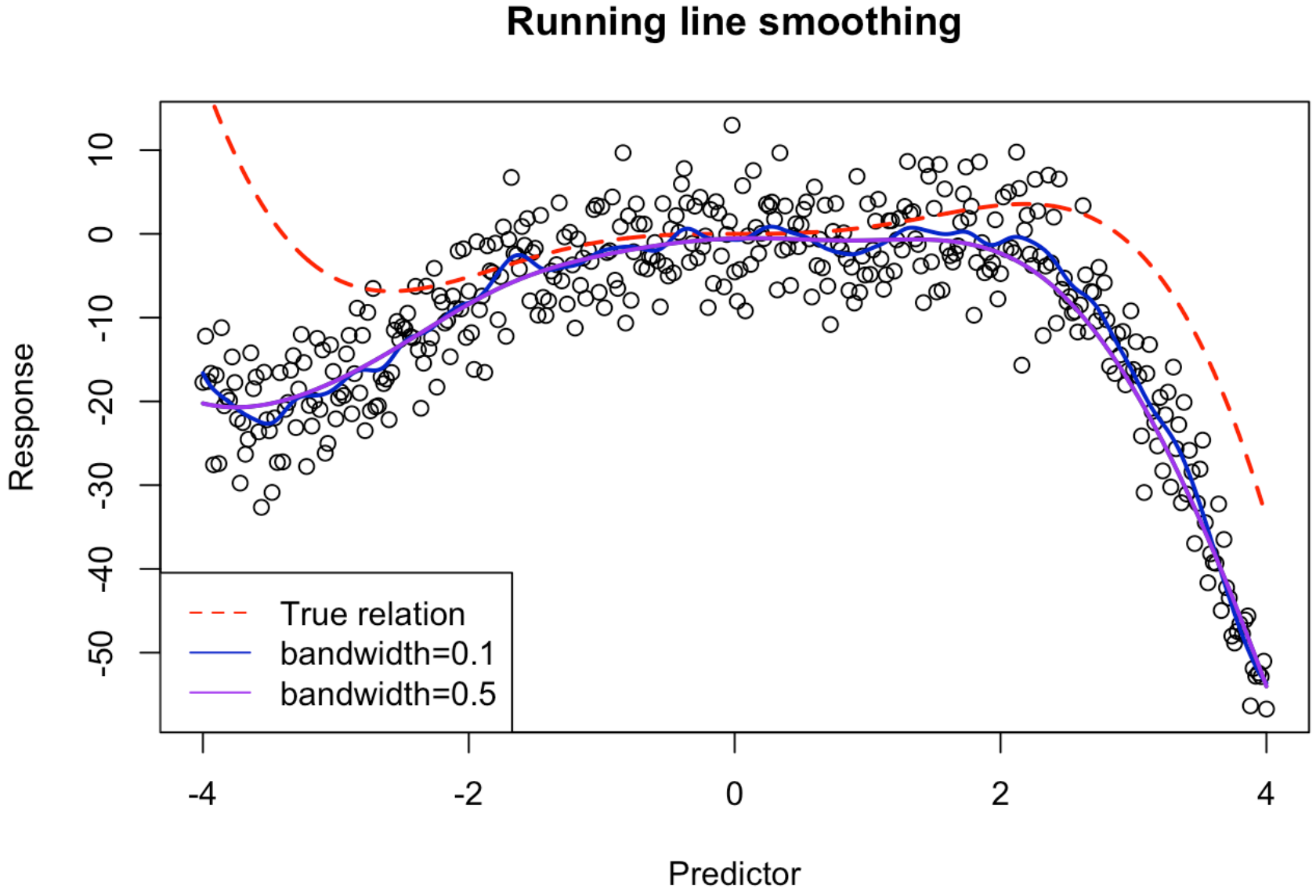
```
#Task1.1 Create a scatter plot of the data

setwd("/Users/likuncui/Downloads/5003/tutorial4/")
easy <- read.table("datasmooth.txt", header=T)
x <- easy$x
y <- easy$y

# The true relationship
s <- function(x){(x^3) * sin((x+3.4)/2)}
# plot data and create the true relationship line
x.plot <- seq(min(x),max(x),length.out=1000)
y.plot <- s(x.plot)
plot(x, y, xlab="Predictor", ylab="Response", main="Running mean smoothing")

#Task1.2 overlay the true relationship line on the plot.
lines(x.plot, y.plot, lty=2, lwd=2, col="red")

legend("bottomleft",c("True relation","k=3","k=43"),
       lty=c(2,1,3), col=c("red", "blue3", "green4"))
```



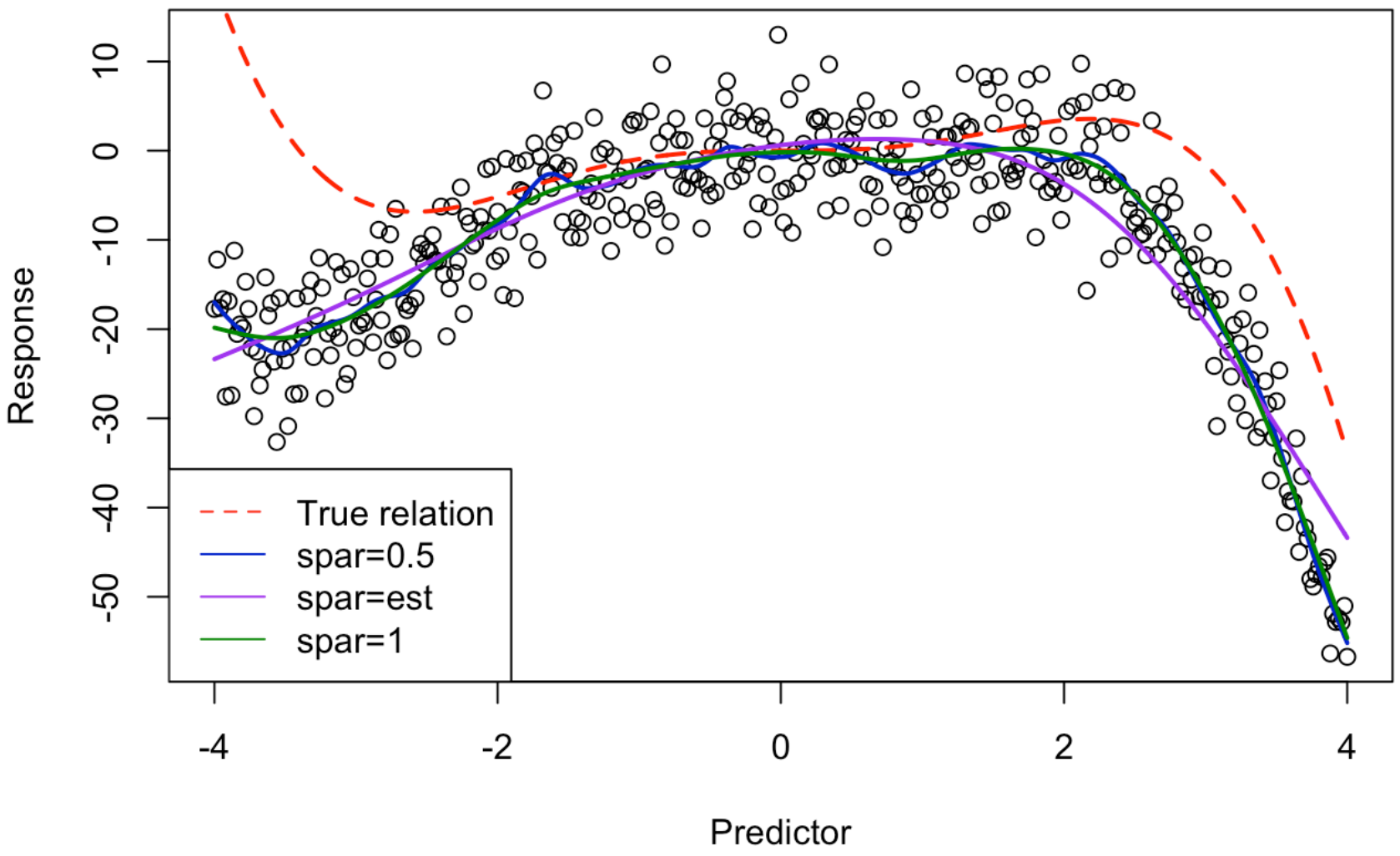

```
#Task2 Try normal kernel smoothing with different bandwidths on “datasmooth.txt”.
# setup the plot of original data and true relationship
x.plot <- seq(min(x), max(x), length.out=1000)
y.plot <- s(x.plot)
plot(x, y, xlab="Predictor", ylab="Response", main="Running line smoothing")
lines(x.plot, y.plot, lty=2, lwd=2, col="red")

# apply kernal smoothing using "KernSmooth" package
library(KernSmooth)
```

```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

```
fit1 <- locpoly(x, y, kernel="normal", bandwidth = 0.1)
lines(fit1, col="blue3", lwd=2)
fit2 <- locpoly(x, y, kernel="normal", bandwidth = 0.5)
lines(fit2, col="green4", lwd=2)

# plot smoothing line with estimated optimal bandwidth
lines(fit2, col="purple", lwd=2)
legend("bottomleft",c("True relation","bandwidth=0.1", "bandwidth=0.5"),
      lty=c(2,1,1), col=c("red", "blue3", "purple"))
```



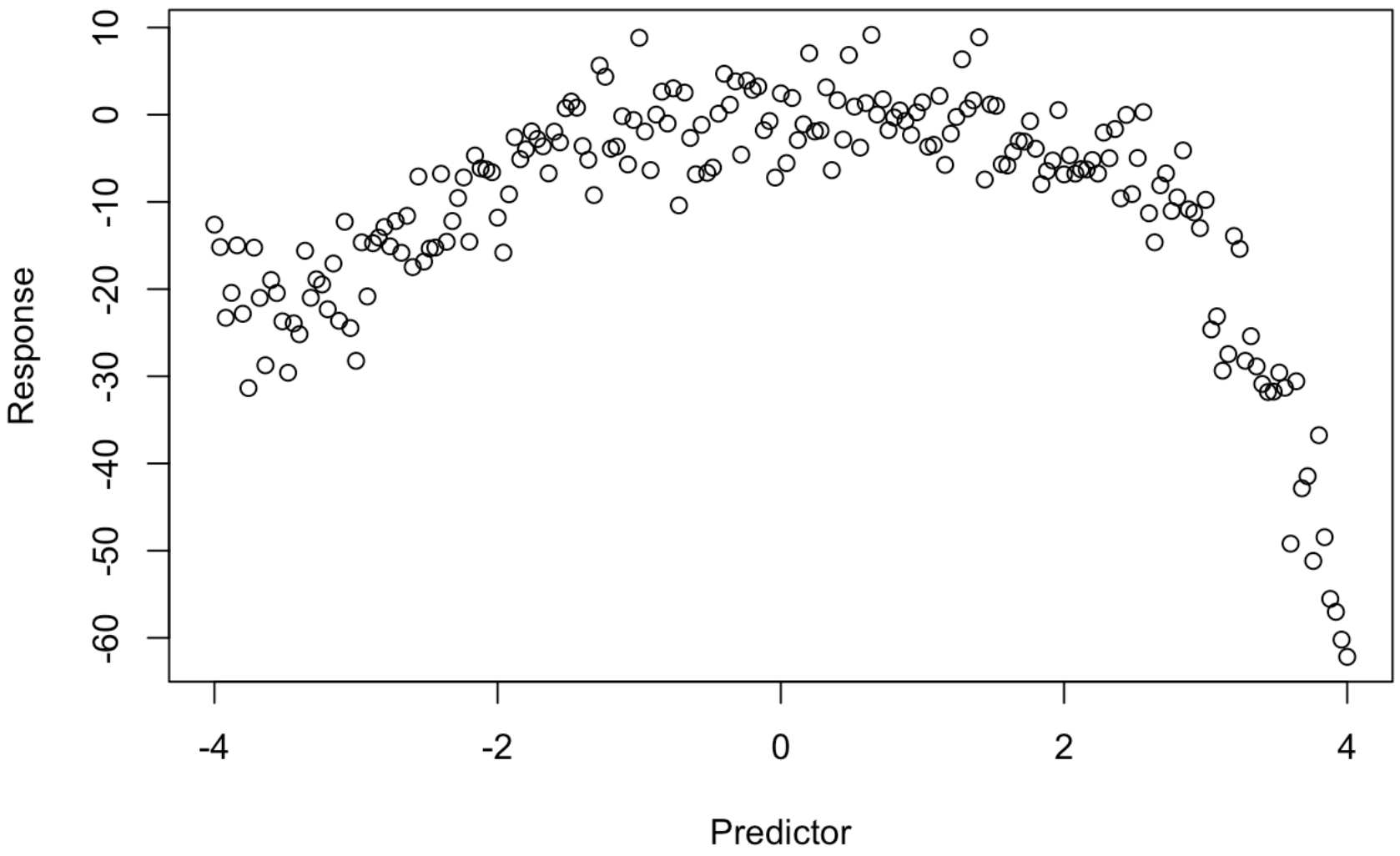

```r
#Task3 Apply cubic spline with different spars on “datasmooth.txt”.
# Use the smooth.spline function in R
cubicSpline1.fit <- smooth.spline(x=x, y=y, cv=FALSE, spar=0.5)
cubicSpline2.fit <- smooth.spline(x=x, y=y, cv=FALSE, spar=1)
cubicSpline3.fit <- smooth.spline(x=x, y=y, cv=TRUE)

## Out put smoothing line
s = function(x){(x^3) * sin((x+3.4)/2)}
x.plot = seq(min(x),max(x),length.out=1000)
y.plot = s(x.plot)
plot(x,y,xlab="Predictor",ylab="Response", main="Cubic spline smoothing")
lines(x.plot, y.plot, lty=2, lwd=2, col="red")
lines(cubicSpline1.fit, col="blue3", lwd=2)
lines(cubicSpline2.fit, col="purple", lwd=2)
lines(cubicSpline3.fit, col="green4", lwd=2)
legend("bottomleft",c("True relation","spar=0.5", "spar=est", "spar=1"),
      lty=c(2,1,1,1), col=c("red", "blue3", "purple" ,"green4"))
```




```
#Task 4.1 Utilise this new dataset to estimate mean squared error
new_easy <- read.table("newDatasmooth.txt", header=T)
new_x <- new_easy$x
new_y <- new_easy$y
# The true relationship
s <- function(new_x){(x^3) * sin((new_x+3.4)/2)}
# plot data and create the true relationship line
new_x.plot <- seq(min(new_x),max(new_x),length.out=1000)
new_y.plot <- s(new_x.plot)
```

```
## Warning in (x^3) * sin((new_x + 3.4)/2): longer object length is not a
## multiple of shorter object length
```

```
plot(new_x, new_y, xlab="Predictor", ylab="Response", main="The new Dataset")
```




```
rmse=(mean(new_y-y)^2)^0.5
```

```
## Warning in new_y - y: longer object length is not a multiple of shorter
## object length
```

```
rmse
```

```
## [1] 0.192121
```

```
#select best bandwidths
#For kernel smoother
#plot(new_x, new_y, xlab="Predictor", ylab="Response", main="The new Dataset")
rmse=(mean(new_y-fit1$y)^2)^0.5
```

```
## Warning in new_y - fit1$y: longer object length is not a multiple of
## shorter object length
```

```
#For bandwidth=0.1
rmse
```

```
## [1] 0.1871664
```

```
#For bandwidth=0.5
rmse=(mean(new_y-fit2$y)^2)^0.5
```

```
## Warning in new_y - fit2$y: longer object length is not a multiple of
## shorter object length
```

```
rmse
```

```
## [1] 0.3931824
```

```
print("Therefore, bandwidth=0.1 is better fitting kernel smoother.")
```

```
## [1] "Therefore, bandwidth=0.1 is better fitting kernel smoother."
```

```
#For cubic spline
rmse1=(mean(new_y-cubicSpline1.fit$y)^2)^0.5
```

```
## Warning in new_y - cubicSpline1.fit$y: longer object length is not a
## multiple of shorter object length
```

```
rmse2=(mean(new_y-cubicSpline2.fit$y)^2)^0.5
```

```
## Warning in new_y - cubicSpline2.fit$y: longer object length is not a
## multiple of shorter object length
```

```
rmse3=(mean(new_y-cubicSpline3.fit$y)^2)^0.5
```

```
## Warning in new_y - cubicSpline3.fit$y: longer object length is not a
## multiple of shorter object length
```

```
rmse1
```

```
## [1] 0.192121
```

```
rmse2
```

```
## [1] 0.192121
```

```
rmse3
```

```
## [1] 0.192121
```

```
print("Therefore, all are with the same mse. They are in equal preformance")
```

```
## [1] "Therefore, all are with the same mse. They are in equal preformance"
```

```
print("According to the mse calculation, kernel smoother is better.")
```

```
## [1] "According to the mse calculation, kernel smoother is better."
```