STAT 428 Homework 8

Yiming Gao (NetID: yimingg2)

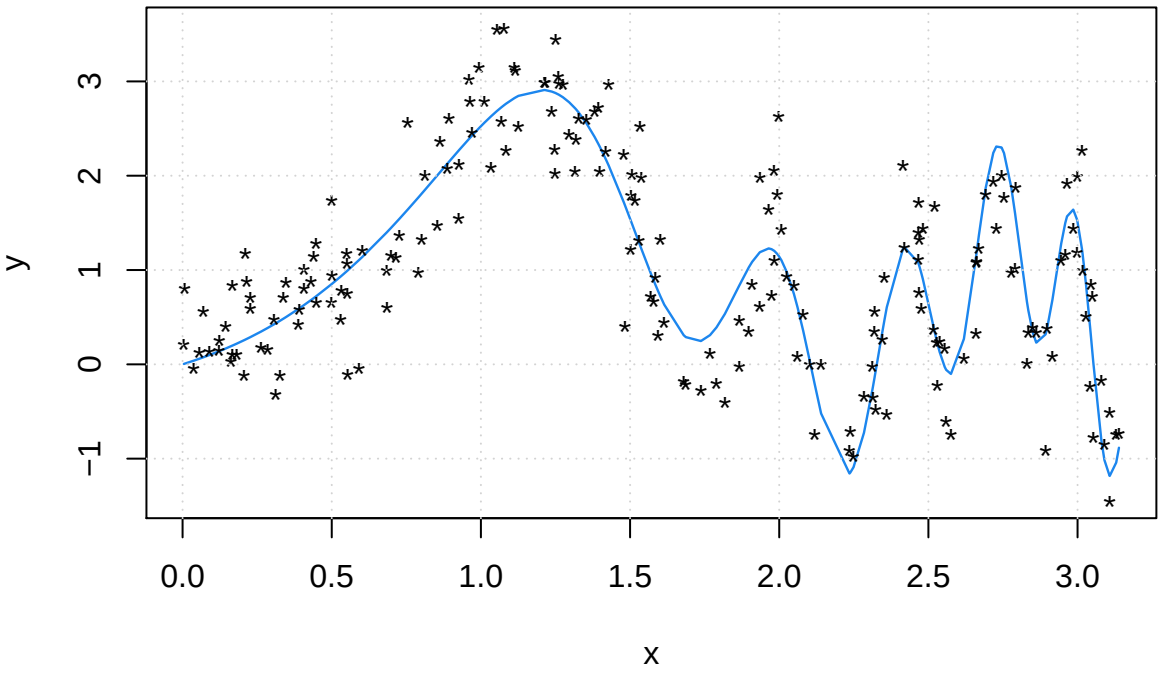
2017/4/27

Contents

Exercise 1

Generate 200 data points from the model $Y = sin(x) + sin(x^2) + sin(x^3) + e$ where x is uniform on $(0, \pi)$ and $e \sim N(0, 0.25)$. Fit the data using cubic B-splines and use LOOCV to decide on how many knots to use and roughly where the knots should be placed.

```
# Generate a sample of n = 200
set.seed(27)
x = sort(runif(200, 0, pi))
mx = sin(x) + sin(x^2) + sin(x^3)
e = rnorm(200, 0, sd = 0.5)
y = mx + e
```

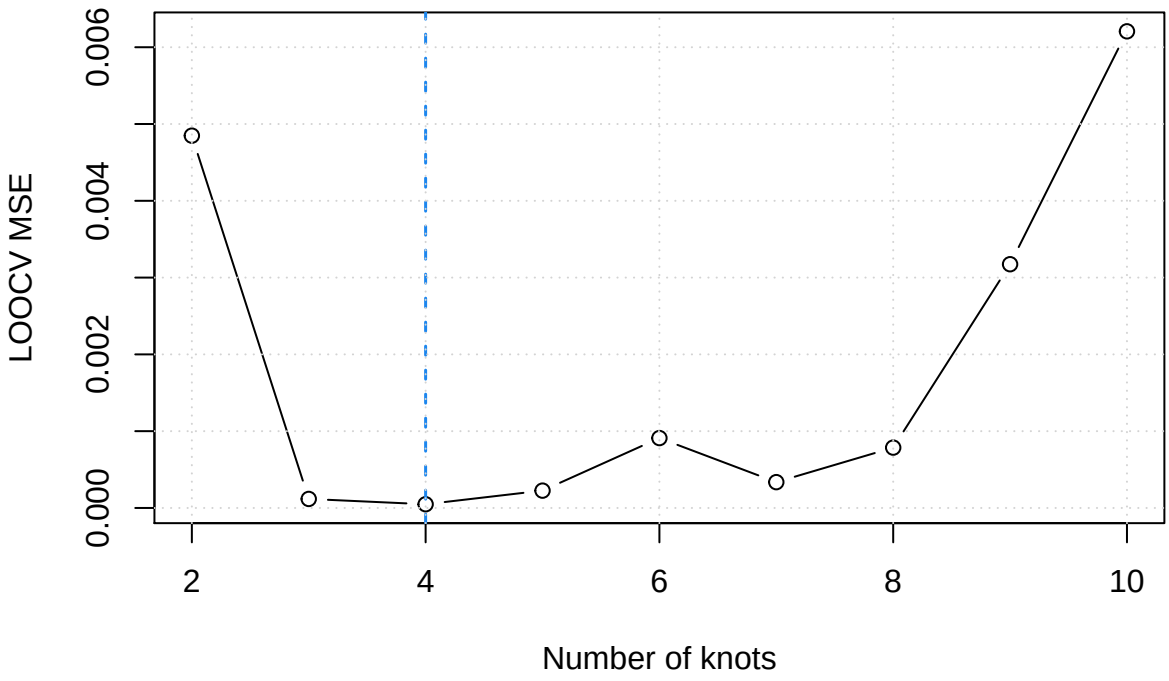


Then we try to select the number of knots based on LOOCV MSE.

```r
library(splines)
xrange = seq(2, 10, 1)
MSE = rep(0, length(xrange))

for (i in xrange) {

  for (j in 1:200) {
    yj = y[-j]
    xj = x[-j]
    MSE0 = 0

    regspline = lm(yj ~ bs(xj, knots = seq(0, pi, length.out = i + 2)[2: i + 1], degree = 3))
    pred = predict(regspline, data.frame(xj = x[j]))
    MSE0 = MSE0 + (y[j]- pred)^2 / 200
  }

  MSE[i-1] = MSE0
}

result = round(cbind(xrange, MSE), 5)
colnames(result) = c("Number of knots", "MSE")
print(result)
```

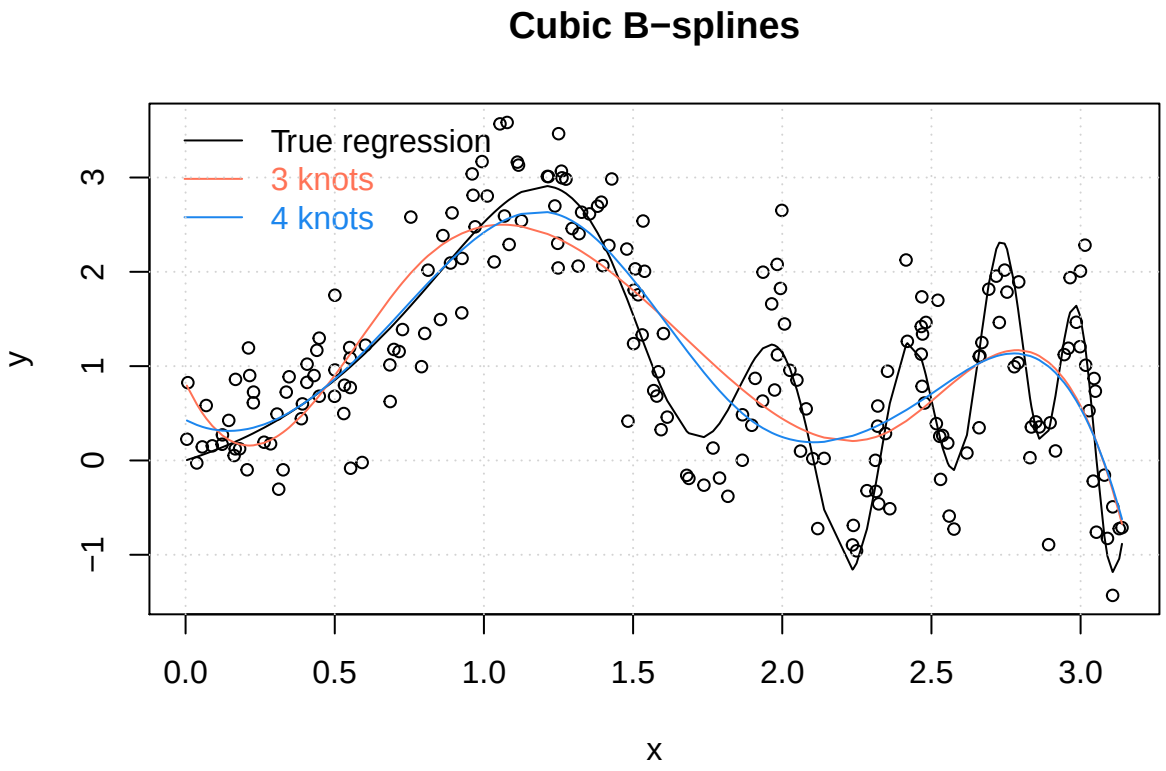
```
##       Number of knots     MSE
##  [1,]               2 0.00485
##  [2,]               3 0.00012
##  [3,]               4 0.00005
##  [4,]               5 0.00023
##  [5,]               6 0.00091
##  [6,]               7 0.00034
##  [7,]               8 0.00079
##  [8,]               9 0.00317
##  [9,]              10 0.00621
```

We plot the leave-one-out cross validated MSE vs. number of knots, and find out when we have 4 knots, we get the lowest MSE, which is 0.00005 with this seed.



We notice that in the previous case, 3 knots and 4 knots give similar low leave one out MSE. We then try the cubic splines independently and draw a plot for comparison.

```
regspline1 = lm(y ~ bs(x, knots = seq(0, pi, length.out = 5)[2: 4], degree = 3))
regspline2 = lm(y ~ bs(x, knots = seq(0, pi, length.out = 6)[2: 5], degree = 3))
```



Previously we put knots evenly at the range of x $(0, \pi)$ and find when the number of knots is 4, we get the lowest MSE. Then we try two approaches to figure out where to put these knots to achieve better results.

Grid search

First we try break the range of x into 4 even intervals, and do grid search on each interval. We create a knots grid to contain all possible combinations of knots, iterate the algorithm, and store each MSE into the vector `MSE`. Here we try $5 * 5 * 5 * 4 = 500$ combinations.

```
knot1 = seq(0, pi*(1/4), length.out = 6)[2: 6]
knot2 = seq(pi*(1/4), pi*(2/4), length.out = 6)[2: 6]
knot3 = seq(pi*(2/4), pi*(3/4), length.out = 6)[2: 6]
knot4 = seq(pi*(3/4), pi, length.out = 6)[2: 5]

knots_grid = as.matrix(expand.grid(knot1 = knot1, knot2 = knot2,
                                   knot3 = knot3, knot4 = knot4))
```

```
MSE = rep(0, nrow(knots_grid))

for (i in 1:nrow(knots_grid)) {

  for (j in 1:200) {
    yj = y[-j]
    xj = x[-j]
    MSE0 = 0

    regspline = lm(yj ~ bs(xj, knots = knots_grid[i, ], degree = 3))
    pred = predict(regspline, data.frame(xj = x[j]))
    MSE0 = MSE0 + (y[j]- pred)^2 / 200
  }

  MSE[i] = MSE0
}

index = match(sort(MSE)[1:10], MSE)
best_grid_knots = knots_grid[index[1], ]
```

```
##        Knot1  Knot2  Knot3  Knot4          MSE
##  [1,] 0.6283 1.5708 2.0420 2.5133 3.594189e-10
##  [2,] 0.7854 1.5708 2.3562 2.5133 2.371934e-08
```

```
##  [3,] 0.1571 0.9425 1.7279 2.5133 6.721818e-08
##  [4,] 0.1571 0.9425 2.0420 2.5133 1.909381e-07
##  [5,] 0.1571 0.9425 1.8850 2.5133 3.618108e-07
##  [6,] 0.7854 1.5708 1.8850 2.5133 3.814769e-07
##  [7,] 0.7854 1.5708 2.1991 2.5133 4.027642e-07
##  [8,] 0.1571 1.4137 2.3562 2.6704 9.401868e-07
##  [9,] 0.1571 1.2566 2.3562 2.6704 1.280148e-06
## [10,] 0.3142 1.2566 2.3562 2.6704 1.529347e-06
```

We print out 10 minimum MSE and corresponding knot placements. The first row gives the lowest LOOCV MSE $3.594e-10$, where the knots are placed at (0.6283, 1.5708, 2.0420, 2.5133). We fit the best cubic spline model for creating plot later.

```
regspline_grid = lm(y ~ bs(x, knots = knots[1, ], degree = 3))
```

Random Search

Then we try to place knots randomly on $(0, \pi)$. We iterate 100 times.

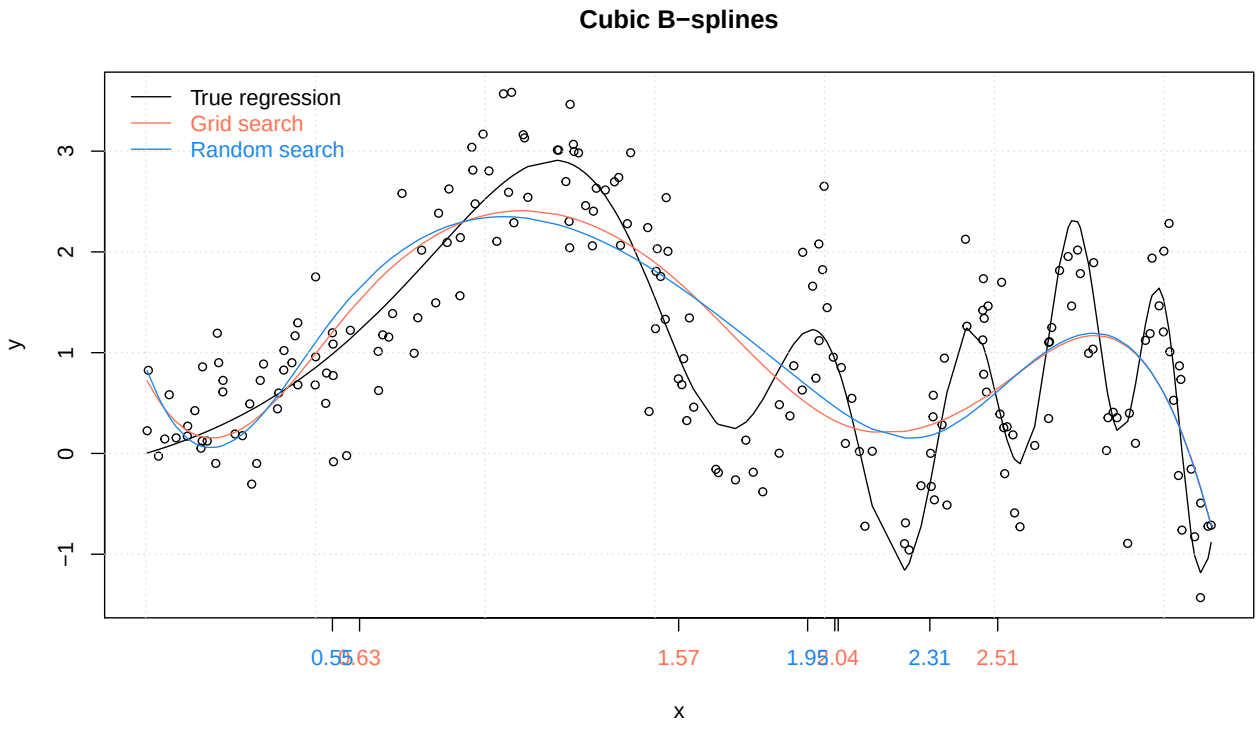
```
set.seed(27)
knots_random = matrix(runif(100 * 4, 0, pi), nrow = 100, ncol = 4)
MSE = rep(0, nrow(knots_random))

for (i in 1:nrow(knots_random)) {

  for (j in 1:200) {
    yj = y[-j]
    xj = x[-j]
    MSE0 = 0

    regspline = lm(yj ~ bs(xj, knots = knots_random[i, ], degree = 3))
    pred = predict(regspline, data.frame(xj = x[j]))
    MSE0 = MSE0 + (y[j]- pred)^2 / 200
  }

  MSE[i] = MSE0
}

index = match(sort(MSE)[1:10], MSE)
best_random_knots = knots_random[index[1], ]
```

```
##         Knot1  Knot2  Knot3  Knot4          MSE
##  [1,] 0.5500 2.3142 1.9527 2.0259 2.401248e-07
##  [2,] 2.5547 2.2354 0.9417 0.3474 2.097238e-06
##  [3,] 1.9646 2.2383 1.0614 2.3625 3.134460e-06
##  [4,] 1.0538 1.5940 2.5958 0.6605 3.944350e-06
##  [5,] 1.4789 1.2504 2.2972 1.9593 5.252522e-06
##  [6,] 1.8980 0.2063 2.1717 2.5622 5.566307e-06
##  [7,] 0.1225 1.2164 0.9288 2.3921 1.170686e-05
##  [8,] 2.5597 0.9637 1.6939 1.4099 1.488141e-05
##  [9,] 1.5170 2.3460 0.3759 0.2322 1.508685e-05
## [10,] 2.4660 0.1665 1.8045 1.8360 1.749330e-05
```

We print out first 10 minimum MSE using random search and corresponding knot placements. The first row gives the lowest LOOCV MSE $2.401e-07$, where the knots are placed at $(0.5500, 2.3142, 1.9527, 2.0259)$.

We can plot the two best spline in the following cell. I set the knots selected by grid search orange, and by random search blue on x-axis.

```
regspline_random = lm(y ~ bs(x, knots = knots[1, ], degree = 3))
```



We notice that two colored lines are almost overlapped.