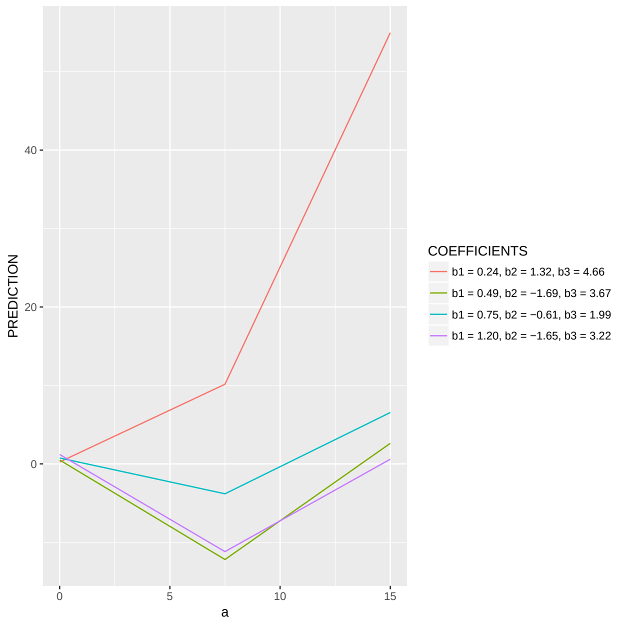
Stat 574B: HW 3 Problem 2

Nate Hattersley

Octorber 25, 2017

part a.

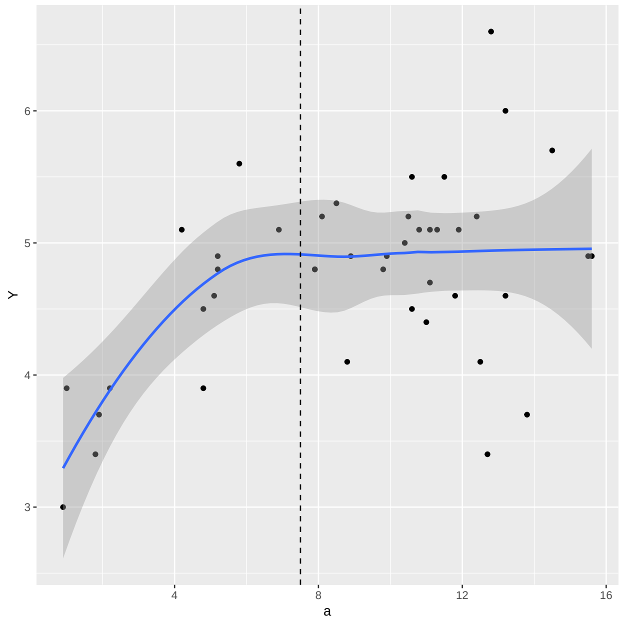
I'm going to randomly select $(\beta_1, \beta_2, \beta_3)$ from the intervals $[0, 2], [-2, 2], [1, 5]$, respectively. Here is a plot of a vs. Y for four different splines:



These "splines" do indeed look like two straight lines connected at the knot, as you described it in the problem statement.

part b.

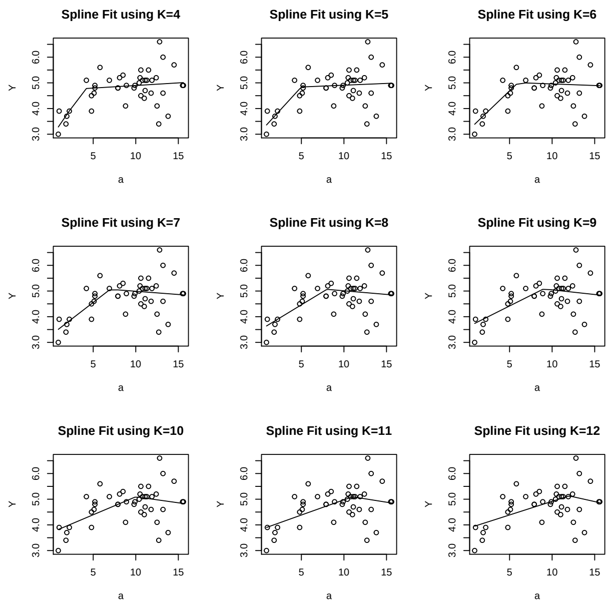
I'm going to make a scatter plot of the data and add a loess fit to the data to determine local trends. I also add a vertical line at $K = 7.5$ to assess how our current knot does at partitioning the data:

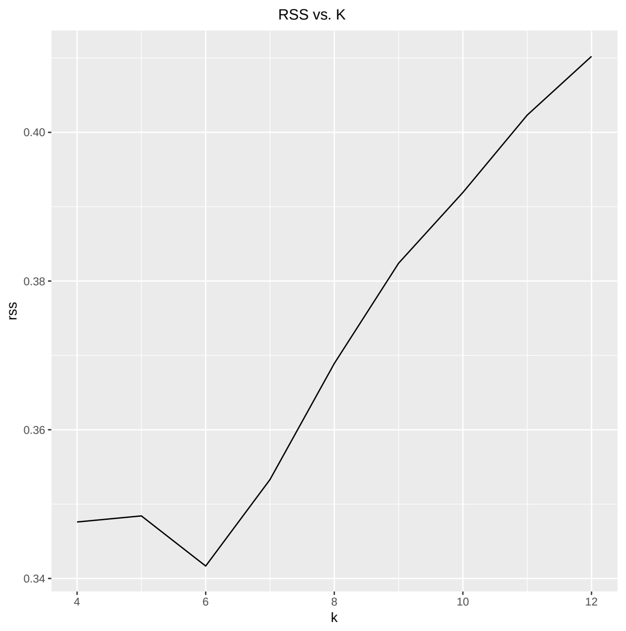


The smoothing very much would indicate that we should use a knot here; it's like two different lines connecting at one point. The loess fit also would indicate that our knot should be smaller than 7.5.

part c.

First, I will plot the spline fit against the data for each possible value of K. Then, I make a plot of RSS vs. K. It would seem that $K = 6$ is the best choice of knot.





part d.

First, assume a flat prior on K for $K \in [4, 12]$. Then, the posterior desnity for K will be proportional to the likelihood. We can write the likelihood of K as:

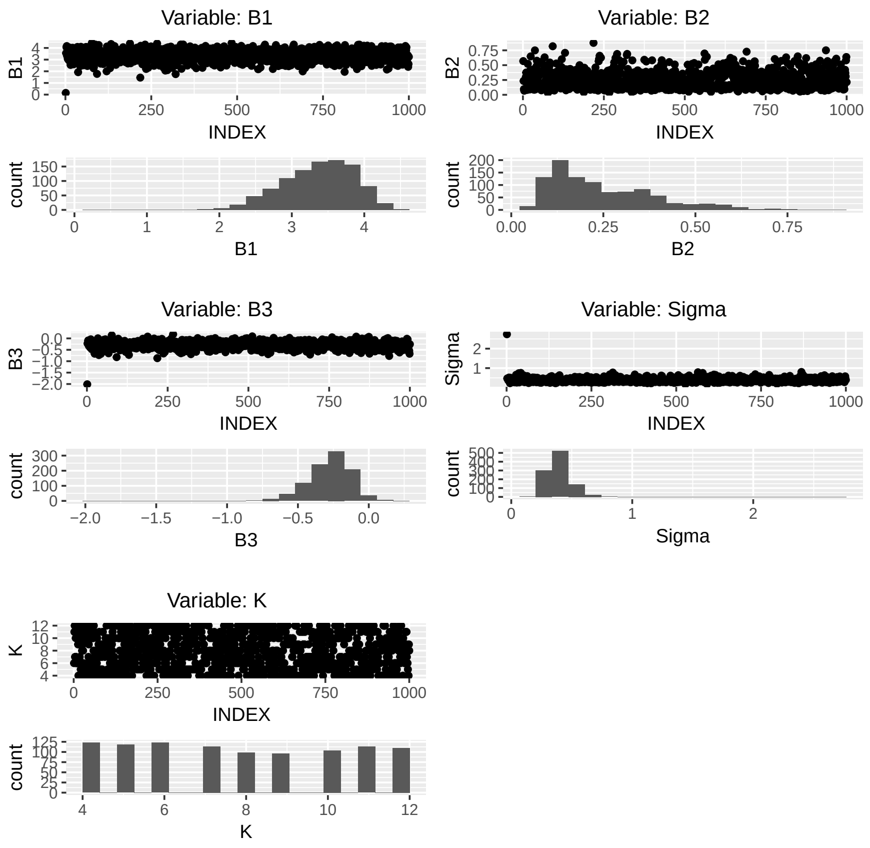
$$L(\kappa|\beta, \sigma^2, y) = \left(\sigma^2\right)^{-.5n} \exp\left\{-\frac{1}{2\sigma^2}(Y - X(\kappa)\beta)^{\mathrm{T}}(Y - X(\kappa)\beta)\right\} \tag{1}$$

I then use a normalized version of the likelihood across the possible values of K to create a probability mass. What I found most computationally stable was to calculate the negative log likelihood, $l = -\log L$, normalize such that the sum of the negative log likelihoods is one, i.e. $l_i^* = \frac{l_i}{\sum_i l_i}$. Then, to sample from the posterior of K conditional on the other parameters, I used the *sample* function in R, with weights $w_i = e^{-l_i^*}$.

My choice of prior for β, σ^2 is the semi-conjugate prior. I also am not going to be daring or adventurous, so I will assume

$$\beta_1, \beta_2, \beta_3 \sim \text{i.i.d. } N(0, 10^6)$$
$$\frac{1}{\sigma^2} \sim \text{Gamma}(10^{-3}, 10^{-3})$$

The posteriors of β, σ^2 conditional on the other parameters are provided in the regression notes, and we just derived the posterior of K, so I will begin a Gibbs sample. I took a run of 1,000 samples. For each parameter, I include an index plot and histogram to show convergence.



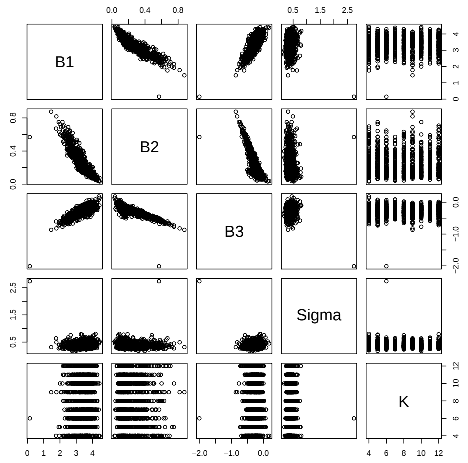
It looks like all of the parameters converged, except for K. Here is a summary of the regression parameters, along with correlations and a scatterplot matrix.

	Mean	Median	SD	X2.5th.percentile	X97.5th.percentile
B1	3.37	3.41	0.50	2.30	4.17
B2	0.25	0.21	0.15	0.07	0.60
B3	-0.28	-0.26	0.16	-0.60	-0.03
Sigma	0.39	0.37	0.12	0.25	0.61
K	7.88	8.00	2.63	4.00	12.00

Table 1: Summary of the Regression Parameters

	B1	B2	B3	Sigma	K
B1	1.00	-0.91	0.82	-0.02	0.06
B2	-0.91	1.00	-0.85	-0.08	-0.04
B3	0.82	-0.85	1.00	-0.14	0.04
Sigma	-0.02	-0.08	-0.14	1.00	-0.02
K	0.06	-0.04	0.04	-0.02	1.00

Table 2: Correlation of the Parameters

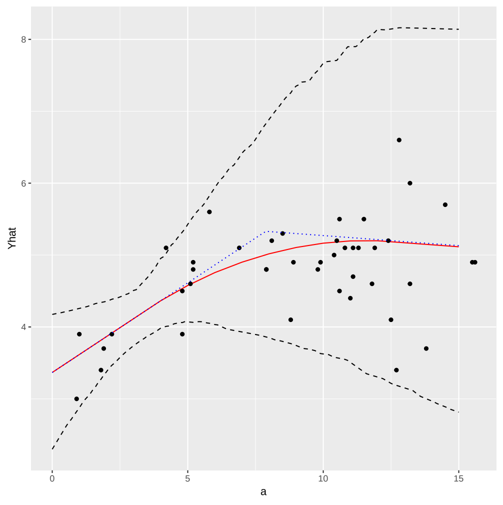


part e.

Looking at the summary of K above, it looks like the data are very uninformative! The histogram and index plots are all over the place, and the 95% confidence interval on K does not restrict our choice of K at all. Our data are really *knot* informative.

part f.

To get the 95% estimate on $\mu(a, \beta, \kappa)$, I first chose a value of a. Then I computed the estimate of $\mu(a, \beta, \kappa)$ for each of my Gibbs samples. Plotted are the means and upper- and lower-2.5% bounds of the Gibbs sample for each a. The mean is in red, and the bounds are dashed black. I let K vary, so this looks like a non-linear function of a. For reference, I plotted the posterior $\mu(a, \beta, \kappa)$ estimate with K fixed at its posterior mean in blue. This one looks more like a spline with one knot. The red line seems to match the trend in the data pretty well. Also, the points where the confidence interval widens seem to be the points where the loess smoothing also diverges. This could be representative of increased variance in the data.



Code

```r
peptide.df <- read.csv("~/Documents/workspace/math/574B/peptide.csv",
    header = F, col.names = c("a", "b", "Y"))
n <- length(peptide.df[, 1])

## given knot(s) k_i, adds column(s) to the data with the
## dummy variable (a-k_i)+
knot <- function(k, data = peptide.df) {
    if (!is.numeric(k))
        stop("Non-numeric argument to function")
    dummy.df <- data
    for (i in 1:length(k)) {
        dummy.col <- with(dummy.df, ifelse(a > k[i], a - k[i],
            0))
        dummy.df %<>% cbind(dummy.col)
        colnames(dummy.df)[ncol(data) + i] <- paste0("k_", k[i])
    }
    dummy.df
}
## Quickly make an lm object given a k
knot.lm <- function(k) {
    if (!is.numeric(k))
        stop("Non-numeric argument to function")
    form <- "Y ~ a"
    for (j in k) {
        form %<>% paste0("+k_", j)
    }
    lm(form, data = knot(k))
}

## Frauda for part a.
data.df <- knot(7.5, data = data.frame(a = seq(0, 15, 0.5)))
predict.knotlm <- function(betas, data = knot(7.5)) {
    with(data, 1 * betas[1] + a * betas[2] + k_7.5 * betas[3])
}
set.seed(186)

## Four sets of random betas
betas <- matrix(c(runif(4, 0, 2), runif(4, -2, 2), runif(4, 1,
    5)), ncol = 4, byrow = T)

## Evaluate the four models for a series of a
predictions <- c()
headers <- c()

for (i in 1:4) {
    predictions %<>% c(predict.knotlm(betas[, i], data = data.df))
    headers %<>% c(sprintf("b1 = %.2f, b2 = %.2f, b3 = %.2f",
        betas[1, i], betas[2, i], betas[3, i]))
}
```

```r
## Long form of the above for ggplot
graphing.df <- data.frame(a = rep(data.df$a, 4), COEFFICIENTS = rep(factor(headers),
    each = length(data.df[, 1])), PREDICTION = predictions)

## Graph for part a.
ggplot(data = graphing.df) + geom_line(aes(x = a, y = PREDICTION,
    col = COEFFICIENTS))
## Scatterplot for part b.
ggplot(data = peptide.df, aes(a, Y)) + geom_point() + geom_smooth(method = "loess") +
    geom_vline(xintercept = 7.5, linetype = "dashed")
## Calculate the RSS and plot our spline fits
rss <- c()
par(mfrow = c(3, 3))
for (i in 4:12) {
    lmi <- knot.lm(i)
    rss %<>% c(summary(lmi)$sigma^2)  ## obtain SSE from the spline lm
    with(peptide.df, {
        ## Data scatter
        plot(Y ~ a, main = paste0("Spline Fit using K=", i))
        ## LM lines
        lines(sort(a), lmi$fitted.values[order(a)])
    })
}
## Plot of RSS vs. K
grid.arrange(ggplot(data.frame(k = 4:12, rss), aes(k, rss)) +
    geom_line(), top = "RSS vs. K")
## Eponymous helper functions
design.matrix <- function(k) {
    namecol <- paste0("k_", k)
    knot(k)[, c("a", namecol)] %>% as.matrix %>% cbind(`1` = rep(1,
        n), .)
}
sse.b <- function(betas, k) {
    X <- design.matrix(k)
    Y <- peptide.df$Y
    t(Y - X %*% betas) %*% (Y - X %*% betas)
}
beta.hat <- function(X, Y) {
    Y <- peptide.df$Y
    solve(t(X) %*% X) %*% t(X) %*% Y
}

## Functions to sample from each conditional posterior

k_weights <- numeric(9)  # Reduce and reuse
# Faster than using sapply(k,L)
sample_k <- function(betas, sigma.sq) {
    ## This is the negative log likelihood
    L <- function(k) {
        (0.5 * n) * log(sigma.sq) + 0.5 * sigma.sq * sse.b(betas,
            k)
    }
    for (k in 4:12) {
        k_weights[k - 3] <<- L(k)
    }
    k_weights <<- k_weights/sum(k_weights)
    sample(4:12, 1, prob = exp(-1 * k_weights))
}

sample_beta <- function(k, sigma.sq) {
    X <- design.matrix(k)
    Sigma <- diag(10^-6, 3) + (t(X) %*% X)/sigma.sq
    Sigma %<>% solve
    mu <- Sigma %*% (((t(X) %*% X)/sigma.sq) %*% beta.hat(X))
    mvrnorm(1, mu = mu, Sigma = Sigma)
}

sample_sigma <- function(betas, k) {
    vn <- n + 0.002  # prior sample size of 2e where e=10^-3
    vnb <- 0.002 + sse.b(betas, k)
    1/rgamma(1, vn/2, vnb/2)
}

## STARTING GIBBS SAMPLE
gibbsno <- 1000
output <- matrix(nrow = gibbsno, ncol = 6)
colnames(output) <- c("INDEX", "B1", "B2", "B3", "Sigma", "K")

## Initial values of K,sigma
k0 <- 12
s0 <- 100

## Concise loop makes coder happy
k <- k0
s <- s0
b <- numeric(3)
for (i in 1:gibbsno) {
    b <<- sample_beta(k, s)
    s <<- sample_sigma(b, k)
    k <<- sample_k(b, s)
    output[i, ] <- c(i, b, s, k)
}

output %<>% data.frame

## Collect the index plots and histograms
plots <- list()
for (i in colnames(output)[-1]) {
    indexplot <- ggplot(output, aes_string("INDEX", i)) + geom_point()

    histogram <- ggplot(output, aes_string(i)) + geom_histogram(bins = 20)
```

```r
    plots[[i]] <- arrangeGrob(indexplot, histogram, name = paste("Plot",
        i), top = textGrob(paste("\nVariable:", i)))
}
do.call(grid.arrange, plots)  ## Prettily arrange
## Xtable to print LATEX tables of the summaries
require(xtable)
percentile <- function(i) {
    function(k) {
        quantile(k, probs = i/100)
    }
}

## xtable mean, sd, ci summary
summary.df <- data.frame(Mean = sapply(output[, -1], mean), Median = sapply(output[,
    -1], median), SD = sapply(output[, -1], sd), `2.5th percentile` = sapply(output[,
    -1], percentile(2.5)), `97.5th percentile` = sapply(output[,
    -1], percentile(97.5)))
print(xtable(summary.df, caption = "Summary of the Regression Parameters"))
## xtable correlation matrix
print(xtable(cor(output[, -1]), caption = "Correlation of the Parameters"))
## scatterplot matrix
pairs(output[, -1])
a <- seq(0, 15, 0.1)

Yhat <- numeric(length(a))
Lower <- numeric(length(a))
Upper <- numeric(length(a))
Yhat2 <- numeric(length(a))

## Yhat2, estimates2, v2 used for the posterior estimates of
## mu given a fixed K

estimates2 <- numeric(gibbsno)
estimates <- numeric(gibbsno)
kbar <- mean(output$K)
## for each value of a
for (i in 1:length(a)) {
    v2 <- ifelse(a[i] > kbar, a[i] - kbar, 0)
    ## iterate over each gibbs sample
    for (j in 1:gibbsno) {
        r <- output[j, c("B1", "B2", "B3", "K")]
        v <- ifelse(a[i] > r$K, a[i] - r$K, 0)
        estimates[j] <- (c(1, a[i], v) * c(r$B1, r$B2, r$B3)) %>%
            sum
        estimates2[j] <- (c(1, a[i], v2) * c(r$B1, r$B2, r$B3)) %>%
            sum
    }
    Yhat[i] <- mean(estimates)
    Yhat2[i] <- mean(estimates2)
    Lower[i] <- quantile(estimates, 0.025)
    Upper[i] <- quantile(estimates, 0.975)
}

graphing.df <- data.frame(a = a, Yhat = Yhat, Yhat2 = Yhat2,
    Lower = Lower, Upper = Upper)

ggplot(graphing.df, aes(x = a)) + geom_line(aes(y = Yhat), color = "red") +
    geom_line(aes(y = Lower), linetype = "dashed") + geom_line(aes(y = Upper),
    linetype = "dashed") + geom_point(data = peptide.df, aes(a,
    Y)) + geom_line(aes(y = Yhat2), linetype = "dotted", color = "blue")
```