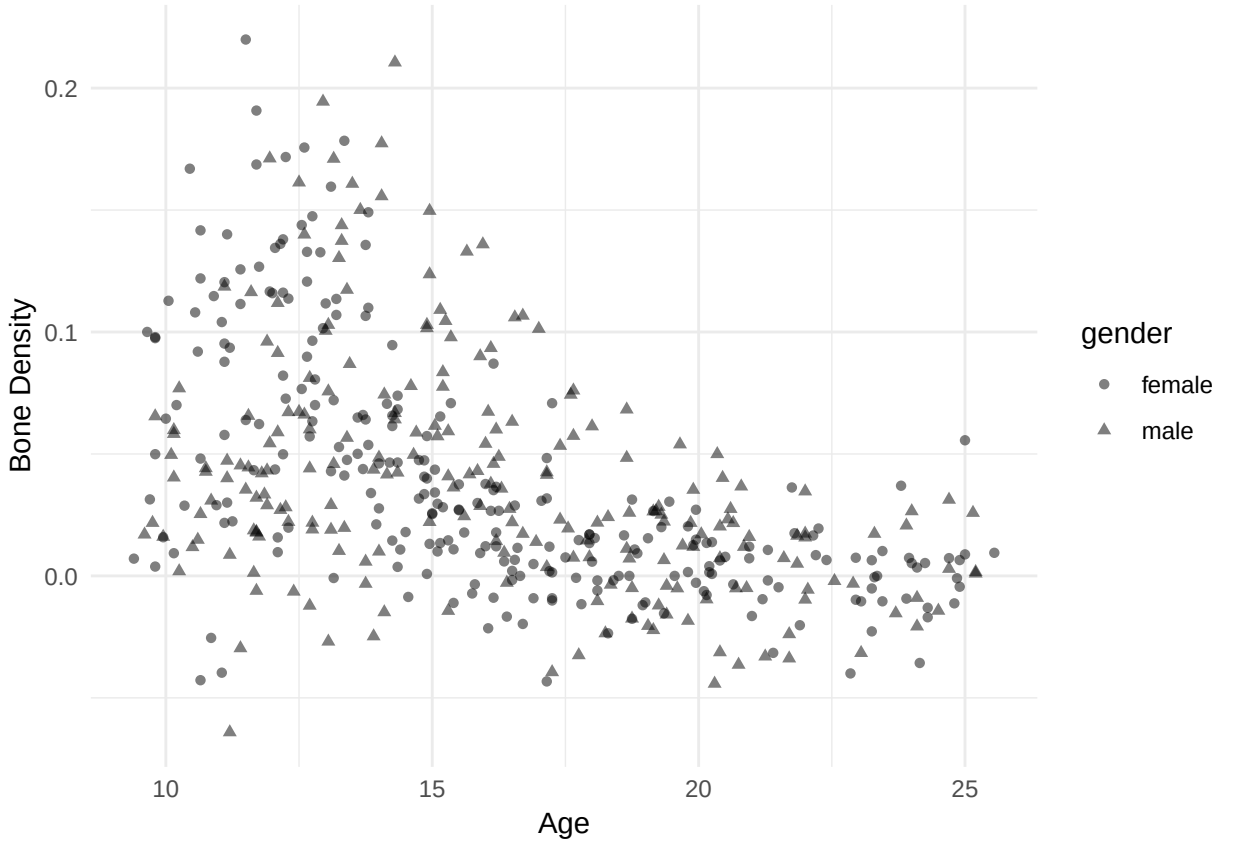
MA 751: Data Assignment 3

Benjamin Draves

4/4/2019

In this exercise we look to compare three different approaches for understanding the bone mineral density dataset. We plot the data here.

```
bones <- read.table("~/Documents/Work/github/Courses/MA 751/DA3/data/bones.txt",
    header = TRUE)

library(ggplot2)
p1 <- ggplot(bones, aes(x = age, y = spnbmd, shape = gender)) +
    geom_point(alpha = 0.5) + labs(x = "Age", y = "Bone Density") +
    theme_minimal()
p1
```



We first fit a smoothing spline to this dataset. Next, we change our focus to a Bayesian setting and place a prior distribution over the coefficients θ. From here, after incorporating information from the data, we obtain a posterior distribution of the parameters and base our inference on this full distribution. Next, we compare these estimators to a bootstrapped version of the smoothing spline by bagging these bootstrapped estimates. Finally, we attempt to compare the error estimates between the bagging estimator and the original smoothing spline.

Smoothing Splines

As the focus of this assignment is the comparison of the methods mentioned above, we use the built in *smooth-spline* function in R to obtain a smoothing spline estimate of the function mapping *age* to *bone density*. Moreover, as we place knots at each one of the age datapoints, our model will have knots at the unique values of age (all.knots = TRUE). Lastly, we choose our smoothing parameter based on the generalized cross-validation criterion. The typical leave-one-out criterion can be written as (5.26)

$$\text{CV}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}_\lambda(x_i)}{1 - S_\lambda(i,i)}\right)^2$$

where $S_\lambda = \mathbf{H}(\mathbf{H}^T\mathbf{H} + \lambda\Omega_n)^{-1}\mathbf{H}^T$. Instead we choose to use the GCV metric (7.52)

$$\text{CCV}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}_\lambda(x_i)}{1 - \text{Tr}(S_\lambda)/n}\right)^2$$

as noted in ESL page 245 that calculating the entries $S_\lambda(i,i)$ could be quite computationally difficult why calculating $\text{Tr}(\mathbf{S}_\lambda)$ may be more stable in some contexts. Moreover, GCV can lead to more smoothing as well as serves as a rough approximation to the AIC model criterion. For these reasons, we use this error criterion in our model fitting procedures.

Having chosen this optimal smoothing parameter, we also look to find an approximate 90% confidence bands for these estimates. Recall from data assignment 2 that the mean and variance of our estimated values, assuming $\mathbf{Y}|\mathbf{X} \sim N(\mathbf{H}\theta, \sigma^2\mathbf{I})$, are given by

$$\begin{aligned}
\mathbb{E}[\hat{\mathbf{Y}}|\mathbf{X}] &= S_\lambda\mathbb{E}[\mathbf{Y}] = S_\lambda\mathbf{H}\theta = \mathbf{H}(\mathbf{H}^T\mathbf{H} + \lambda\Omega_n)^{-1}\mathbf{H}^T\mathbf{H}\theta \\
\text{Var}[\hat{\mathbf{Y}}|\mathbf{X}] &= S_\lambda\text{Var}[\hat{\mathbf{Y}}]S_\lambda^T = \sigma^2 S_\lambda S_\lambda = \sigma^2\mathbf{H}(\mathbf{H}^T\mathbf{H} + \lambda\Omega_n)^{-1}\mathbf{H}^T\mathbf{H}(\mathbf{H}^T\mathbf{H} + \lambda\Omega_n)^{-1}\mathbf{H}^T
\end{aligned}$$

where $h(x) = (1 \quad x \quad N_1(x) \quad N_2(x) \dots N_{p-2}(x))^T$ and

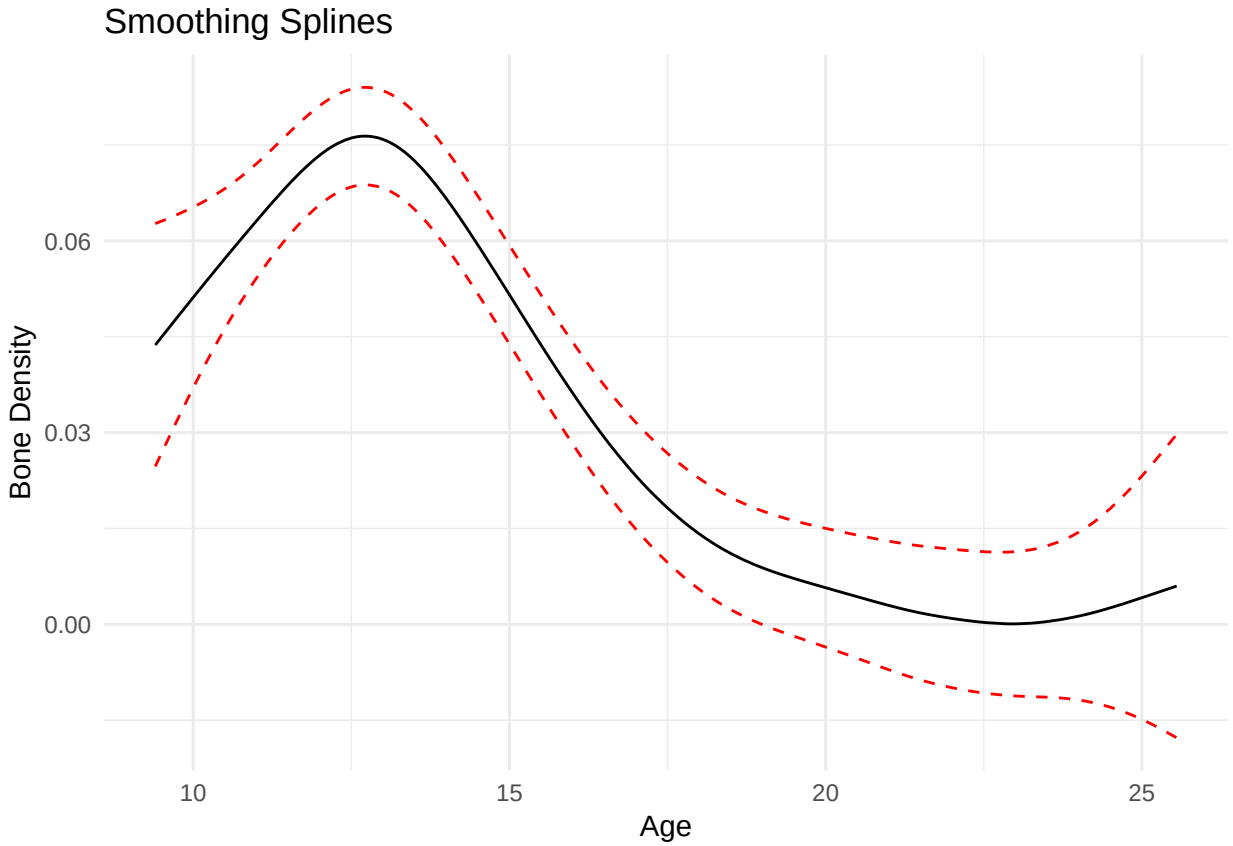
$$\mathbf{H} = \begin{bmatrix} h(x_1)^T \\ h(x_2)^T \\ \vdots \\ h(x_n)^T \end{bmatrix} \in \mathbb{R}^{n\times p}$$

Therefore, an approximate $1-\alpha$ percent confidence interval for $\hat{\mathbf{Y}}_j$ is given by

$$h(X_j)^T\hat{\theta} \pm z_{\alpha/2}\hat{\sigma}\sqrt{(S_\lambda S_\lambda)_{jj}}$$

Therefore, we need to find an explicit expression for S_λ in order to calculate these confidence intervals. Notice, however, that S_λ is a function of $\mathbf{X}$ and λ. Therefore, having fixed these values, we can change values of $\mathbf{y}$ to attain different values of the S_λ matrix. For instance, suppose that $e_j = (0, 0, \quad, 1, \quad, 0)^T$. Then when $\mathbf{Y} = e_j$ we have the identity $\hat{\mathbf{Y}} = S_\lambda e_j = (S_\lambda)_{\cdot j}$. Therefore, as *smooth.splines* will generate $\hat{\mathbf{Y}}$, we can directly construct S_λ by calling the *smooth.spline* function for every value of $e_j, j \in [n]$. Having explicitly found S_λ, we can plot the fit and corresponding 90% confidence intervals as follows.

```r
#-----------------------------------------------
# Fit Smoothing spline
#-----------------------------------------------

# fit full smoothing spline
sspline <- smooth.spline(bones$age, bones$spnbmd, all.knots = T)

# recover S-lambda
Sl <- t(apply(diag(nrow(bones)), 2, function(y) predict(smooth.spline(bones$age,
    y, all.knots = T, lambda = sspline$lambda), bones$age)$y))

# get estimated conditional variance
sigma <- sqrt(as.numeric(crossprod(bones$spnbmd - Sl %*% bones$spnbmd))/(nrow(bones) -
    length(unique(bones$age))))

# get sqrt(diag(SlSl))
widths <- sqrt(diag(crossprod(Sl)))


# set up + plot spline fit
df <- data.frame(Age = bones$age, fit = Sl %*% bones$spnbmd,
    points = bones$spnbmd, upper = Sl %*% bones$spnbmd + sigma *
        qnorm(0.95) * widths, lower = Sl %*% bones$spnbmd - sigma *
        qnorm(0.95) * widths)
df2 <- df[!duplicated(df$Age), ]

p1 <- ggplot(df2, aes(x = Age, y = points)) + geom_line(aes(y = fit)) +
    geom_line(aes(y = upper), linetype = "dashed", col = "red") +
    geom_line(aes(y = lower), linetype = "dashed", col = "red") +
    theme_minimal() + labs(x = "Age", y = "Bone Density", title = "Smoothing Splines")
p1
```



Bayesian Approaches

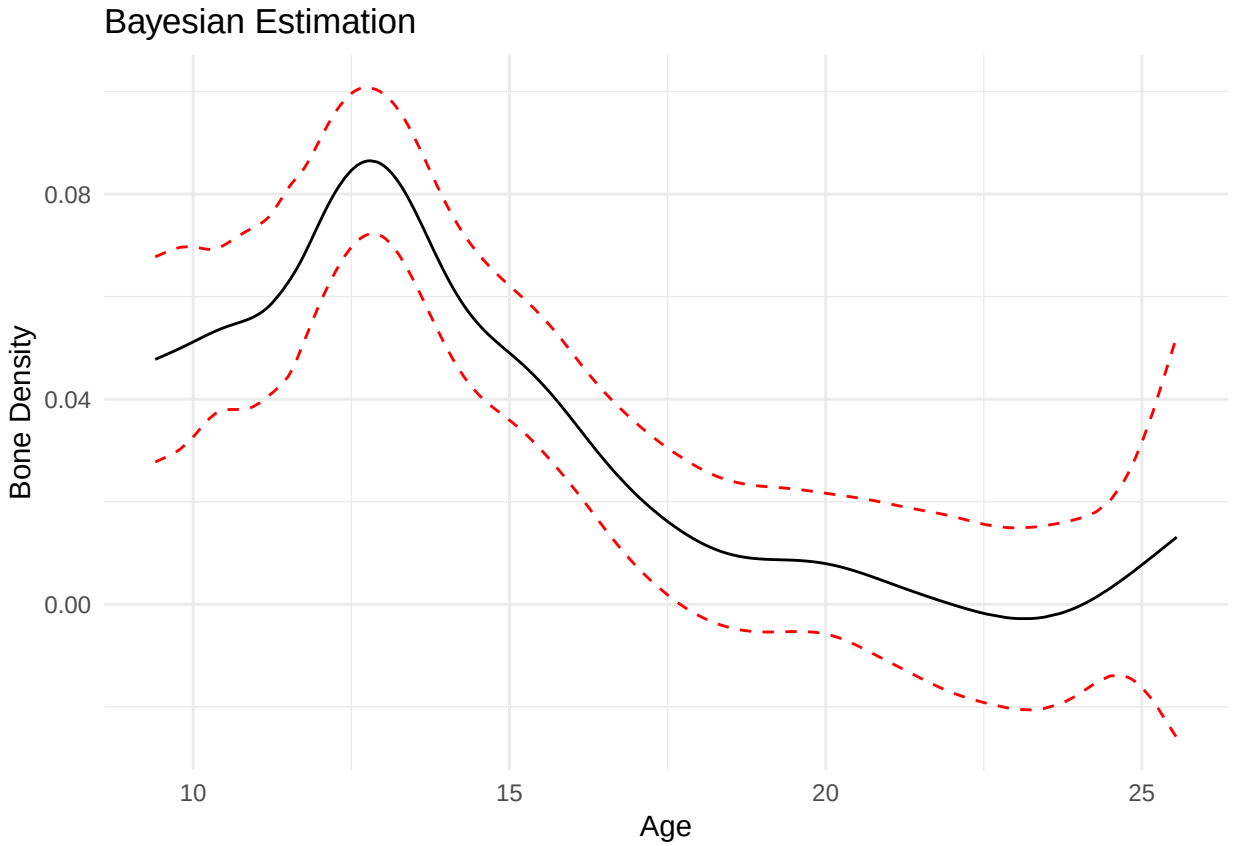
Instead of this non-parameteric approach, we now turn to a Bayesian methodology where we place a prior distribution on the parameters of the model $\theta \sim N(0, \tau\Sigma)$ where $\tau \in \mathbb{R}^+$ is a prior parameter that scales the variance inline with our prior information. Here, we assume the model is a spline model with form $\mu(X) = \mathbf{H}\theta$ where $\mathbf{H}$ is given above. Then as stated in ESL (8.27) and (8.28) the posterior distribution of θ is $\theta \sim N((\mathbf{H}^T\mathbf{H} + \frac{\sigma^2}{\tau}\Sigma^{-1})^{-1}\mathbf{H}^T\mathbf{Y}, \sigma^2(\mathbf{H}^T\mathbf{H} + \frac{\sigma^2}{\tau}\Sigma^{-1})^{-1})$ and using this, the posterior distribution for the spline estimate has the properties

$$\mathbb{E}[\mu(\mathbf{X})] = \mathbf{H}(\mathbf{H}^T\mathbf{H} + \frac{\sigma^2}{\tau}\Sigma^{-1})^{-1}\mathbf{H}^T\mathbf{Y}$$
$$\text{Var}[\mu(\mathbf{X})] = \sigma^2\mathbf{H}(\mathbf{H}^T\mathbf{H} + \frac{\sigma^2}{\tau}\Sigma^{-1})^{-1}\mathbf{H}^T$$

For simplicity we assume that $\tau = 1$ and $\Sigma = \mathbf{I}$. Now, assuming that the posterior of $\mu(\mathbf{X})$ is approximately normal, we can construct credibility intervals in the same way we did as above using $z_{\alpha/2}$ as appropriate scaling factors. We plot these estimates and their approximate credibility intervals below.

```
#---------------------------------------
# Bayesian Estimation
#---------------------------------------

# construct H matrix
N <- function(x, num, knots) {
    if (num > length(knots)) {
```

```
        print("num must be less than # of Knots")
        return(-1)
    }
    if (num == 1)
        return(x)

    k <- num - 1
    K <- length(knots)
    dk <- (ifelse(x - knots[k] > 0, (x - knots[k])^3, 0) - ifelse(x -
        knots[K] > 0, (x - knots[K])^3, 0))/(knots[K] - knots[k])
    dKm1 <- (ifelse(x - knots[K - 1] > 0, (x - knots[K - 1])^3,
        0) - ifelse(x - knots[K] > 0, (x - knots[K])^3, 0))/(knots[K] -
        knots[K - 1])

    return(dk - dKm1)
}
h <- function(x, knots) {
    vec <- numeric(length(knots) - 1)
    for (i in 1:length(vec)) vec[i] <- N(x, i, knots)
    return(vec)
}
H <- t(sapply(bones$age, FUN = function(x) h(x, unique(bones$age))))
H <- cbind(rep(1, nrow(H)), H)

# get estimates and confidence bands
tau <- 1
est <- H %*% solve(crossprod(H) + (1/tau) * diag(ncol(H))) %*%
    crossprod(H, bones$spnbmd)
sigma <- sigma <- sqrt(as.numeric(crossprod(bones$spnbmd - est))/(nrow(bones) -
    length(unique(bones$age))))
sdev <- sqrt(diag(H %*% solve(crossprod(H) + (1/tau) * diag(ncol(H))) %*%
    t(H)))

df <- data.frame(Age = bones$age, fit = est, points = bones$spnbmd,
    lower = est - qnorm(0.95) * sigma * sdev, upper = est + qnorm(0.95) *
        sigma * sdev)
df2 <- df[!duplicated(df$Age), ]

p2 <- ggplot(df2, aes(x = Age, y = points)) + geom_line(aes(y = fit)) +
    geom_line(aes(y = upper), linetype = "dashed", col = "red") +
    geom_line(aes(y = lower), linetype = "dashed", col = "red") +
    theme_minimal() + labs(x = "Age", y = "Bone Density", title = "Bayesian Estimation")

p2
```



Bootstrapping Smoothing Splines

Having introdcued both a frequentist and bayesian approach to fitting a nonparametric smoothing spline, we now consider the parametric bootstrap as a resampling technique that may improve our fitting procedure. In this setting, we use the parametric bootstrap which we discribe here for completeness. Consider the natural cubic spline model

$$\begin{aligned} \mathbf{Y} &= \mu(\mathbf{X}) + \epsilon; \epsilon \sim N(0, \sigma^2) \\ \mu(x) &= h(x)^T \theta \end{aligned}$$

Under the parametric distribution we create *new* response values

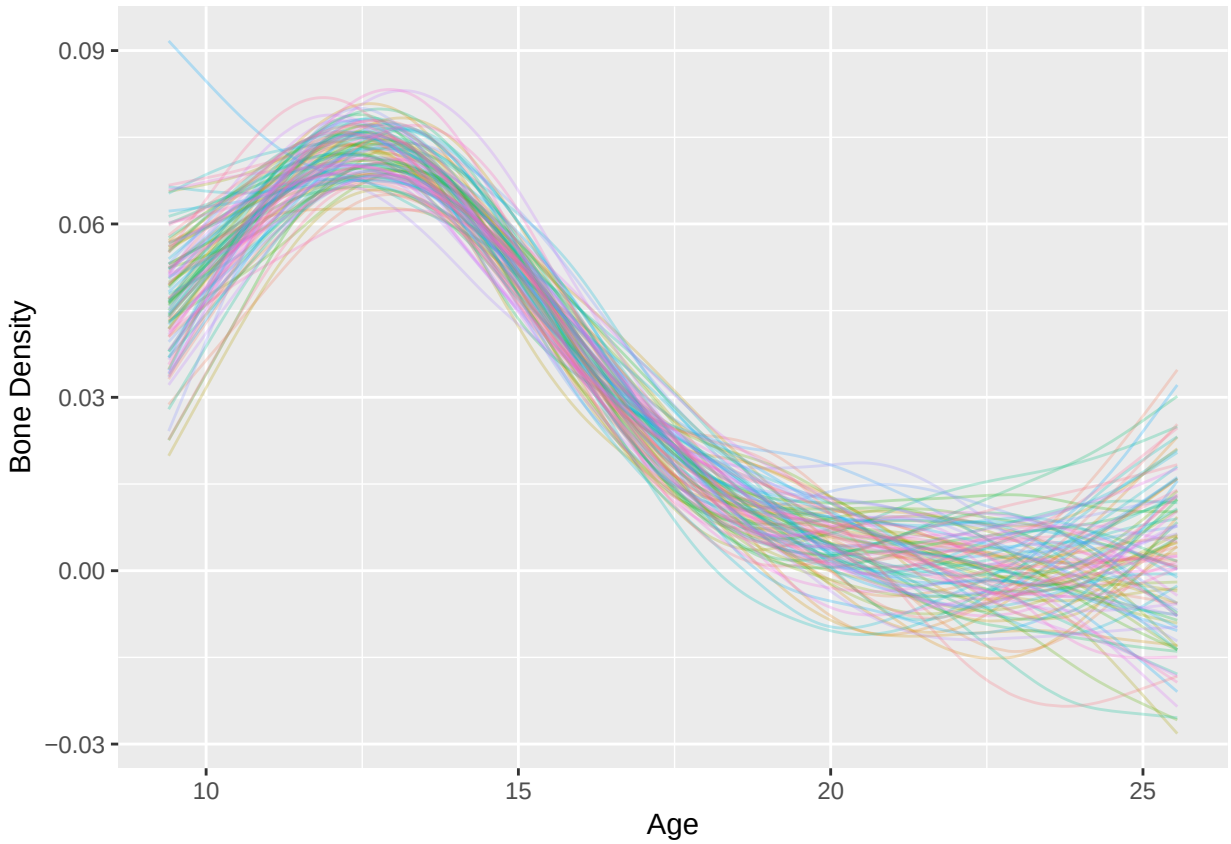
$$\mathbf{Y}^{(b)} = \hat{\mu}(\mathbf{X}) + \epsilon^{(b)}; \epsilon^{(b)} \sim N(0, \hat{\sigma}^2 \mathbf{I})$$

for $b = 1, 2, \ldots, B$. With each new response variable, we achieve a new estimate of $\mu(\cdot)$

$$\hat{\mu}^{(b)}(\mathbf{X}) = \mathbf{H}(\mathbf{H}^T\mathbf{H} + \hat{\lambda}_{opt}\Omega_n)^{-1}\mathbf{H}^T\mathbf{Y}^{(b)} = S_\lambda \mathbf{Y}^{(b)}$$

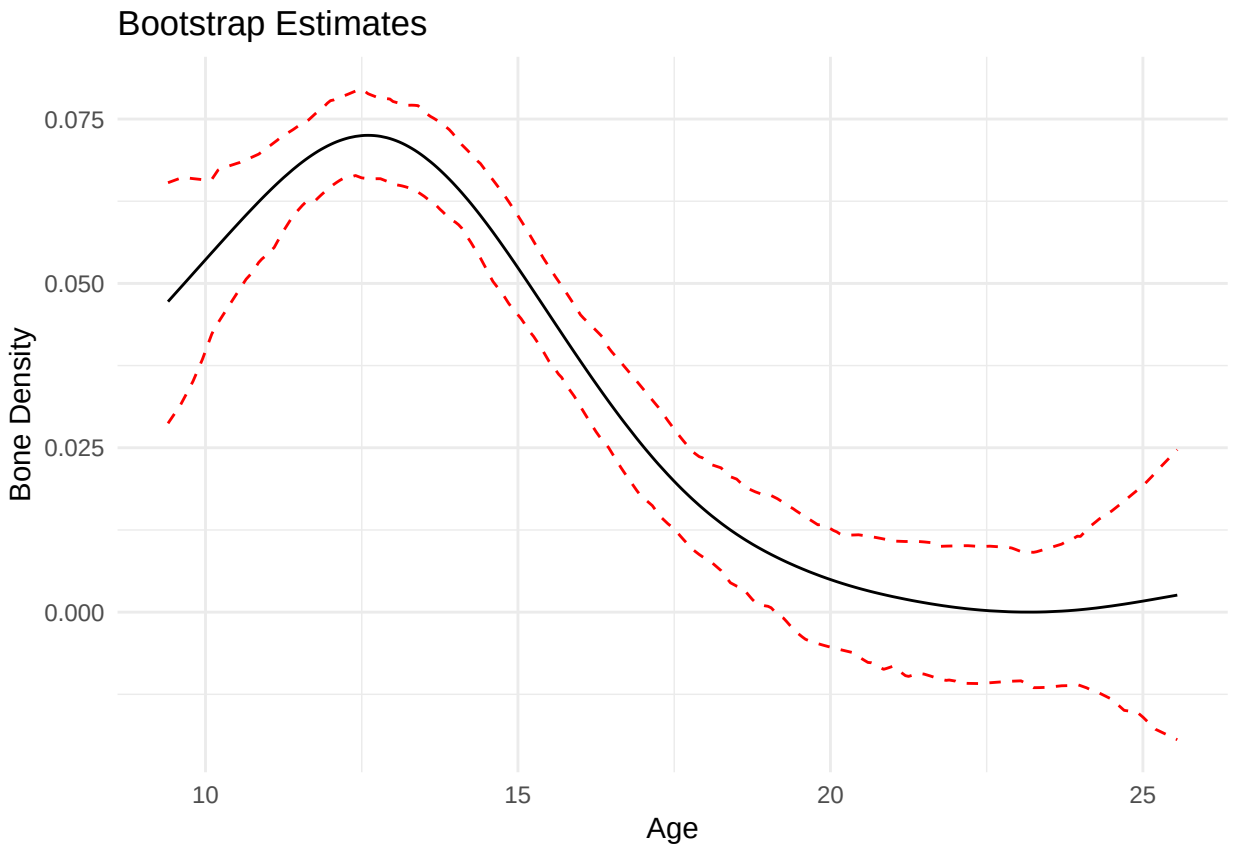
where $\hat{\lambda}_{opt}$ is the parameter that minimized GCV from part a. We implement this bootstrap below and plot all of the estimates.

```r
#-----------------------------------------
# Bootstrapping Estimates
#-----------------------------------------

# set up BS parameters
B <- 100
ystar <- matrix(NA, ncol = B, nrow = nrow(H))
set.seed(1985)
# set up fit statistics
fit <- Sl %*% bones$spnbmd
sigma <- sqrt(as.numeric(crossprod(bones$spnbmd - Sl %*% bones$spnbmd))/(nrow(bones) -
    length(unique(bones$age))))

# bootstrap
for (b in 1:B) {
    # make new y values
    ystar[, b] <- fit + rnorm(length(fit), mean = 0, sd = sigma)
}

# get new estimates
df <- data.frame(fit = Sl %*% ystar, Age = bones$age, points = bones$spnbmd)
df2 <- df[!duplicated(df$Age), ]
library(reshape)
fit_boot <- melt(df2, id = c("Age", "points"))

ggplot(fit_boot, aes(x = Age, y = value, col = variable)) + geom_line(alpha = 0.25) +
    theme(legend.position = "none") + labs(x = "Age", y = "Bone Density")
```



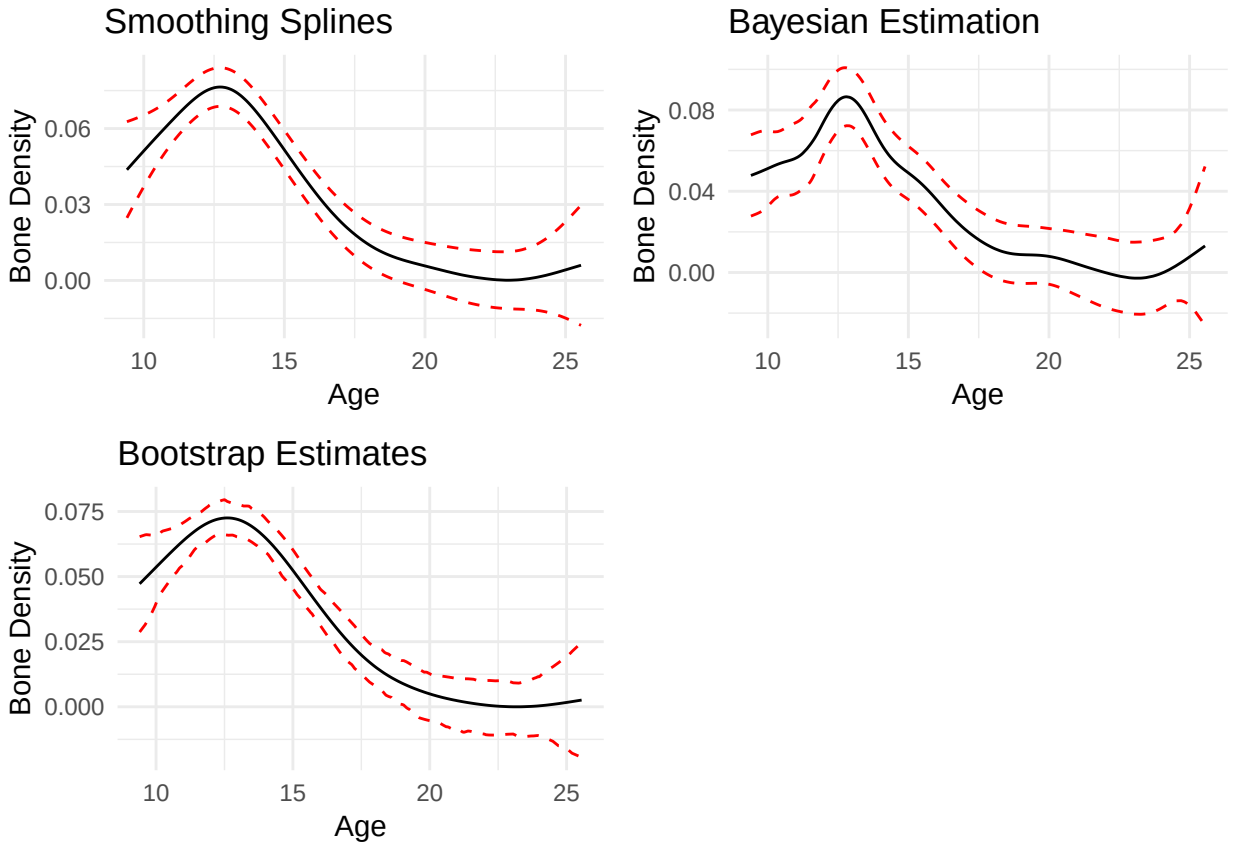
Now that we have achieved $B = 100$ different estimates $\hat{\mathbf{Y}}^{(b)}$ we can construct empirical confidence intervals for each value of x. That is, for Age $= x$, we can construct an empiracal $1 - \alpha$ confidence interval centered at the bagging estiamte given by $\left(\delta^*_{\alpha/2}, \delta^*_{1-\alpha/2}\right)$ where δ^*_α is the empiracle α quantile of the bootstrap distribution. We plot the bagging estimate and these estimated confidence intervals below.

```
#----------------------------------------
# Construct Bagging/Bootstrap Estimator
#----------------------------------------


Bag <- apply(df2[, 1:100], 1, mean)
df3 <- data.frame(Age = unique(df2$Age), fit = Bag, upper = apply(df2[,
    1:100], 1, function(x) unname(quantile(x, 0.95))), lower = apply(df2[,
    1:100], 1, function(x) unname(quantile(x, 0.05))))

p3 <- ggplot(df3, aes(x = Age, y = fit)) + geom_line() + geom_line(aes(y = upper),
    linetype = "dashed", col = "red") + geom_line(aes(y = lower),
    linetype = "dashed", col = "red") + theme_minimal() + labs(x = "Age",
    y = "Bone Density", title = "Bootstrap Estimates")
p3
```



Now to compare the estimates considered to this point, we plot each fit with their corresponding 90% confidence intervals, credibility intervals, or empirical intervals below.

```
library(gridExtra)
grid.arrange(p1, p2, p3, nrow = 2, ncol = 2)
```



In the top left of this plot is the natural cubic spline. This estimate appears to vary smoothing with changes in Age. Moreover, the confidence intervals appear to be tighter in more dense areas of Age. In the top right figure, the Bayesian estimator is plotted. It appears that this line may be overfitting the data as the credibility intervals and the line itself seem to vary frequently with small changes in Age. Lastly, the bootstrap estimator appears to match the smoothing spline closely but appears more smooth in general. This could be due to the additional averaging step taken by the bagging estimate. Moreover, the empiracle confidence bands appear to be more rough than that of the smoothing spline. Generally speaking it appears that all three of these estimators capture the general trend of the data with varying degree of smoothness.

Bagging & Error Estimation

Lastly we look to compare the error incurred by the smoothing spline and the bootstrapped version of this estimator. Recall that we defined our error by the generalized cross validiation GCV given by

$$\text{GCV}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}(x_i)}{1 - \text{Tr}(S_\lambda)/n}\right)^2$$

As we wish to compare this error between the smoothing spline to the bagging estimator, recall both of these estimators are given by

$$\hat{f}_{Spline}(x) = S_\lambda \mathbf{Y}$$
$$\hat{f}_{Bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}_b(x) = \frac{1}{B}\sum_{b=1}^{B} S_\lambda \mathbf{Y}^{(b)} = S_\lambda\left(\frac{1}{B}\sum_{b=1}^{B}\mathbf{Y}^{(b)}\right)$$

Now notice as we fixed $\lambda = \hat{\lambda}_{opt}$ both of these methods have the same smoothing matrix $\mathbf{S}_\lambda$, we can define the GCV error for bowth estimator as follows

$$\text{GCV}^{(spline)}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - (S_\lambda \mathbf{Y})_i}{1 - \text{Tr}(S_\lambda)/n}\right)^2$$

$$\text{GCV}^{(boot)}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{B}\sum_{b=1}^{B}\frac{y_i - (S_\lambda \mathbf{Y}^{(b)})_i}{1 - \text{Tr}(S_\lambda)/n}\right)^2$$

The function *smooth.spline* outputs this GCV error and we see that $GCV^{(spline)}(\lambda) = 0.00167295$. Next we calculate this value for the bagging estimate.

```
#-------------------------------------------
# Estimating GCV Error
#-------------------------------------------

# setup up initial parameters
n <- nrow(bones)
B <- 100
tmp <- cbind(unique(bones$age), Bag)

const <- 1 - sum(diag(Sl))/n

# get dataframe (Age, Y, Yhat - Bag)
df <- cbind(bones$age, bones$spnbmd, NA)
for (i in 1:n) {
    df[i, 3] <- tmp[which(df[i, 1] == tmp[, 1]), 2]
}
colnames(df) <- c("Age", "Bone Density", "Estimated Bone Density")

# estimate GCV
GCV_boot <- 1/n * crossprod((df[, 2] - df[, 3])/const)

# print GCV Boot
message(paste("Generalized CV Error Boostrap: ", GCV_boot))
```

```
## Generalized CV Error Boostrap:  0.00168754332534836
```

Therefore we see that $GCV^{(spline)} \approx GCV^{(boot)}$. This should not be surprise as $\mathbb{E}[\hat{f}_{bag}(x)] = \mathbb{E}[\hat{f}_{spline}(x)]$ and $\text{Var}[\hat{f}_{bag}(x)] = \text{Var}[\hat{f}_{spline}(x)]$. Therefore, we see that both of these methods given almost identical GCV errors while providing different smoothing fits.