STAT 471: Homework 2

Nico Melton

Due: October 4, 2021 at 11:59pm

Contents

Instructions

Setup

Pull the latest version of this assignment from Github and set your working directory to `stat-471-fall-2021/homework/homework-2`. Consult the getting started guide if you need to brush up on `R` or `Git`.

Collaboration

The collaboration policy is as stated on the Syllabus:

> "Students are permitted to work together on homework assignments, but solutions must be written up and submitted individually. Students must disclose any sources of assistance they received; furthermore, they are prohibited from verbatim copying from any source and from consulting solutions to problems that may be available online and/or from past iterations of the course."

In accordance with this policy,

Please list anyone you discussed this homework with: Kennedy Manley

Please list what external references you consulted (e.g. articles, books, or websites): None

Writeup

Use this document as a starting point for your writeup, adding your solutions after "**Solution**". Add your R code using code chunks and add your text answers using **bold text**. Consult the preparing reports guide for guidance on compilation, creation of figures and tables, and presentation quality.

Programming

The `tidyverse` paradigm for data wrangling, manipulation, and visualization is strongly encouraged, but points will not be deducted for using base `R`.

We'll need to use the following `R` packages:

```
library(tidyverse)  # tidyverse
library(kableExtra) # for printing tables
library(cowplot)    # for side by side plots
library(FNN)        # for K-nearest-neighbors regression
```

We'll also need the `cross_validate_spline` function from Unit 2 Lecture 3:

```
source("../../functions/cross_validate_spline.R")
```

Grading

The point value for each problem sub-part is indicated. Additionally, the presentation quality of the solution for each problem (as exemplified by the guidelines in Section 3 of the preparing reports guide will be evaluated on a per-problem basis (e.g. in this homework, there are three problems). There are 100 points possible on this homework, 85 of which are for correctness and 15 of which are for presentation.

Submission

Compile your writeup to PDF and submit to Gradescope.

1 Case study: Bone mineral density (40 points for correctness; 10 points for presentation)

In this exercise, we will be looking at a data set (available online) on spinal bone mineral density, a physiological indicator that increases during puberty when a child grows.

Below is the data description:

> "Relative spinal bone mineral density measurements on 261 North American adolescents. Each value is the difference in spnbmd taken on two consecutive visits, divided by the average. The age is the average age over the two visits."
>
> Variables:
>
> `idnum`: identifies the child, and hence the repeat measurements
>
> `age`: average age of child when measurements were taken
>
> `gender`: male or female
>
> `spnbmd`: Relative Spinal bone mineral density measurement

The goal is to learn about the typical trends of bone mineral density during puberty for boys and girls.

1.1 Import (2 points)

- Using `readr`, import the data from the above URL into a tibble called `bmd`. Specify the column types using the `col_types` argument.
- Print the imported tibble (no need to use `kable`).

Solution.

Read in the data from the link using `read_tsv`. Store the tibble in `bmd` and print.

```
# import data
bmd = read_tsv("https://web.stanford.edu/~hastie/ElemStatLearn/datasets/bone.data",
               col_types = "nncn")
bmd # print
```

```
## # A tibble: 485 x 4
##    idnum   age gender   spnbmd
##    <dbl> <dbl> <chr>     <dbl>
##  1     1  11.7 male    0.0181
##  2     1  12.7 male    0.0601
##  3     1  13.8 male    0.00586
##  4     2  13.2 male    0.0103
##  5     2  14.3 male    0.211
##  6     2  15.3 male    0.0408
##  7     3  11.4 male   -0.0296
##  8     3  12.4 male   -0.00643
##  9     3  13.4 male    0.0566
## 10     4  10.6 female  0.108
## # ... with 475 more rows
```

Table 1: Number of children by gender in the dataset.

gender	count	median age
female	259	15.3
male	226	15.6

1.2 Explore (10 points)

- To keep things simple, let's ignore the fact that we have repeated measurements on children. To this end, remove the `idnum` column from `bmd`.
- What is the number of boys and girls in this dataset (ignoring the fact that there are repeated measurements)? What are the median ages of these boys and girls?
- Produce boxplots to compare the distributions of `spnbmd` and `age` between boys and girls (display these as two plots side by side, one for `spnbmd` and one for `age`). Are there apparent differences in either `spnbmd` or `age` between these two groups?
- Create a scatter plot of `spnbmd` (y axis) versus `age` (x axis), faceting by `gender`. What trends do you see in this data?

Solution.

Use `select` to remove the `idnum` column. Use `group_by` and `summarise` to find the number of boys and girls in the dataset. See Table 1.

```
# remove `idnum` from `bmd`
bmd = bmd %>% select(-idnum)
# find the number of boys and girls in `bmd`
bmd %>%
  group_by(gender) %>%
  summarise(count = n(), `median age` = median(age)) %>%
  kable(format = "latex", row.names = NA,
      booktabs = TRUE,
      caption = "Number of children by gender in the dataset.") %>%
  kable_styling(position = "center")
```

There are 249 girls and 226 boys in this dataset. The median ages for boys and girls are 15.6 and 15.3 respectively. See Table 1.

Use `geom_boxplot` to plot boxplots of `spnbmd` and `age` by gender. See Figure 1

```
# plot boxplots of the distributions of `spnbmd` and `age`
#between boys and girls side by side
p.spn = bmd %>%
  ggplot(aes(x = gender, y = spnbmd)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Relative Spinal Bone Mineral Density") +
  theme_bw()
p.age = bmd %>%
  ggplot(aes(x = gender, y = age)) +
  geom_boxplot() +
  labs(x = "Gender", y = "Average Age (years)") +
```

```
  theme_bw()
plot_grid(p.spn, p.age, ncol = 2) # plot
```

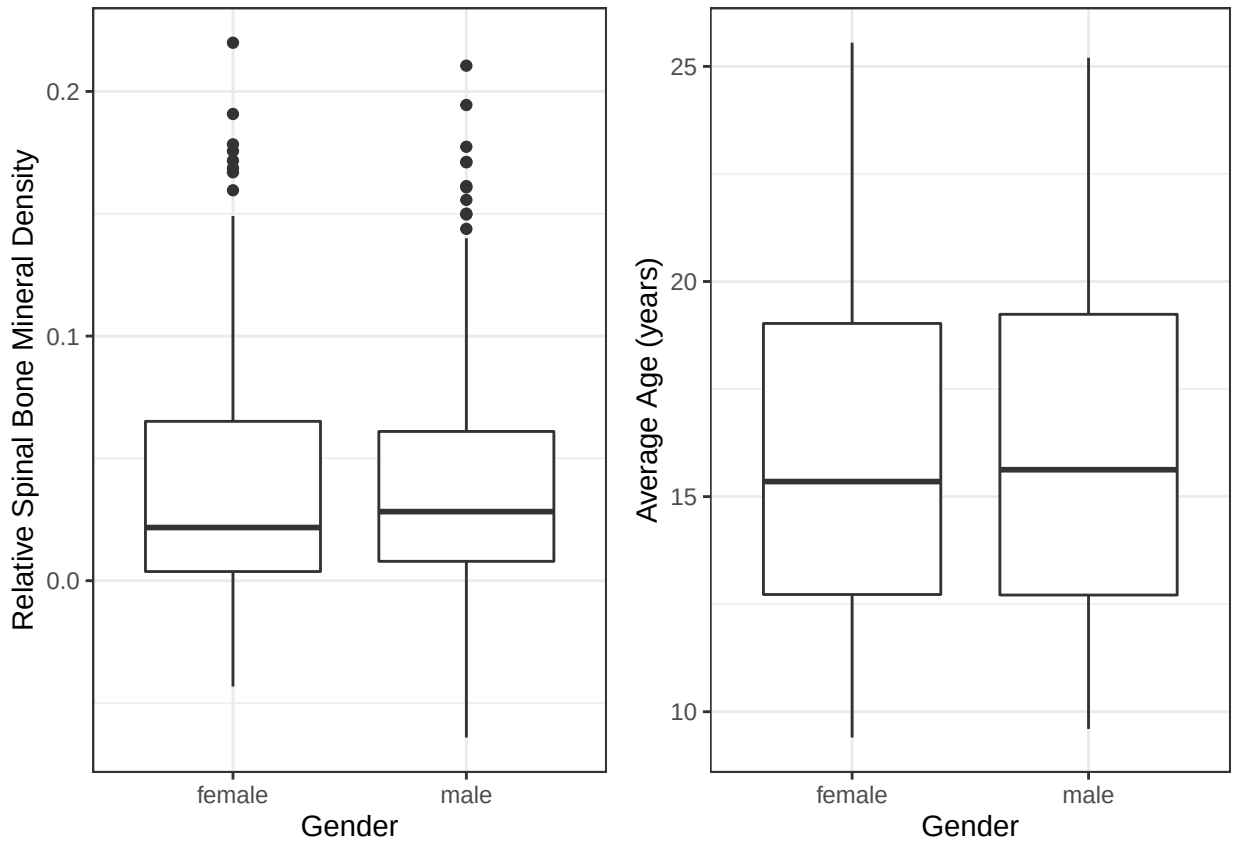

Figure 1: Boxplots of Relative Spinal Bone Mineral Density and Average Age of participants by gender.

The ages and relative spinal bone mineral densities are pretty similar between females and males. Females have a slightly larger spread of relative spinal bone mineral density than males but a lower median relative spinal bone mineral density. Males in this dataset are slightly older on average than females. See Figure 1

Use `geom_point` and `facet_wrap` to plot a scatter plot of `spnbmd` by `age` and faceting by `gender`. See Figure 2

```
# plot a scatter plot of `spnbmd` by `age` faceting by `gender`
bmd %>%
  ggplot(aes(x = age, y = spnbmd)) +
  geom_point() +
  facet_wrap(~gender) +
  labs(x = "Age (years)", y = "Relative Spinal Bone Mineral Density") +
  theme_bw()
```

The scatter plots for males and females both have weak negative correlations between age and relative spinal bone mineral density that seems to flatten out after puberty. For males, however, younger ages seem to have lower relative bone mineral density than females. See Figure 2

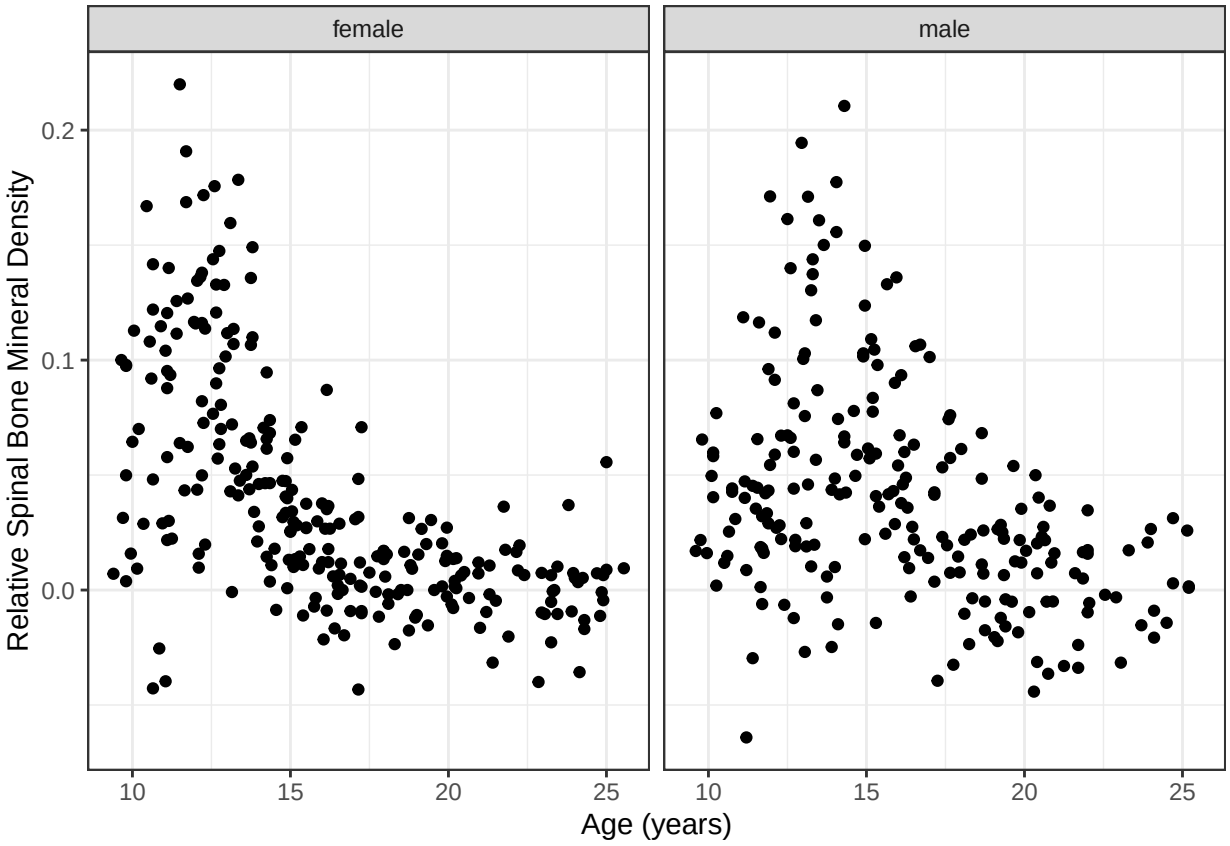

Figure 2: Scatter plots of Relative Spinal Bone Mineral Density by Age for females and males.

1.3 Model (15 points)

There are clearly some trends in this data, but they are somewhat hard to see given the substantial amount of variability. This is where splines come in handy.

1.3.1 Split

To ensure unbiased assessment of predictive models, let's split the data before we start modeling it.

- Split `bmd` into training (80%) and test (20%) sets, using the rows in `train_samples` below for training. Store these in tibbles called `bmd_train` and `bmd_test`, respectively.

Solution.

Create tibbles `bmd_train` and `bmd_test` with 80% and 20% of the data respectively from `bmd`.

```
set.seed(5) # seed set for reproducibility (DO NOT CHANGE)
n = nrow(bmd)
train_samples = sample(1:n, round(0.8*n))
# get the training data
bmd_train = bmd[train_samples,]
# get the test data
bmd_test = bmd[-train_samples,]
```

1.3.2 Tune

- Since the trends in `spnbmd` look somewhat different for boys than for girls, we might want to fit separate splines to these two groups. Separate `bmd_train` into `bmd_train_male` and `bmd_train_female`, and likewise for `bmd_test`.
- Using `cross_validate_spline` from Lecture 3, perform 10-fold cross-validation on `bmd_train_male` and `bmd_train_female`, trying degrees of freedom 1,2,...,15. Display the two resulting CV plots side by side.
- What are the degrees of freedom values minimizing the CV curve for boys and girls, and what are the values obtained from the one standard error rule?
- For the sake of simplicity, let's use the same degrees of freedom for males as well as females. Define `df.min` to be the maximum of the two `df.min` values for males and females, and define `df.1se` likewise. Add these two spline fits to the scatter plot of `spnbmd` (y axis) versus `age` (x axis), faceting by `gender`.
- Given our intuition for what growth curves look like, which of these two values of the degrees of freedom makes more sense?

Solution.

Use `cross_validate_spline` to get cross validation plots for both boys and girls using training data. See figure 3.

```
# separate `bmd_train` and `bmd_test` into tibbles for males and females
bmd_train_male = bmd_train %>% filter(gender == "male")
bmd_train_female = bmd_train %>% filter(gender == "female")
bmd_test_male = bmd_test %>% filter(gender == "male")
bmd_test_female = bmd_test %>% filter(gender == "female")
```

```r
# use `cross_validation_spline` to perform 10-fold cross-validation
dfs = c(1:15) # create a vector of dfs from 1 to 15
## on bmd_train_male
cv_bmd_male =  cross_validate_spline(x = bmd_train_male$age,
                                     y = bmd_train_male$spnbmd,
                                     nfolds = 10, df_values = dfs)
## on bmd_train_female
cv_bmd_female =  cross_validate_spline(x = bmd_train_female$age,
                                     y = bmd_train_female$spnbmd,
                                     nfolds = 10, df_values = dfs)
# plot
plot_grid(cv_bmd_male$cv_plot + labs(title = "Male"),
          cv_bmd_female$cv_plot + labs(title = "Female"),
          nrow = 2)
```

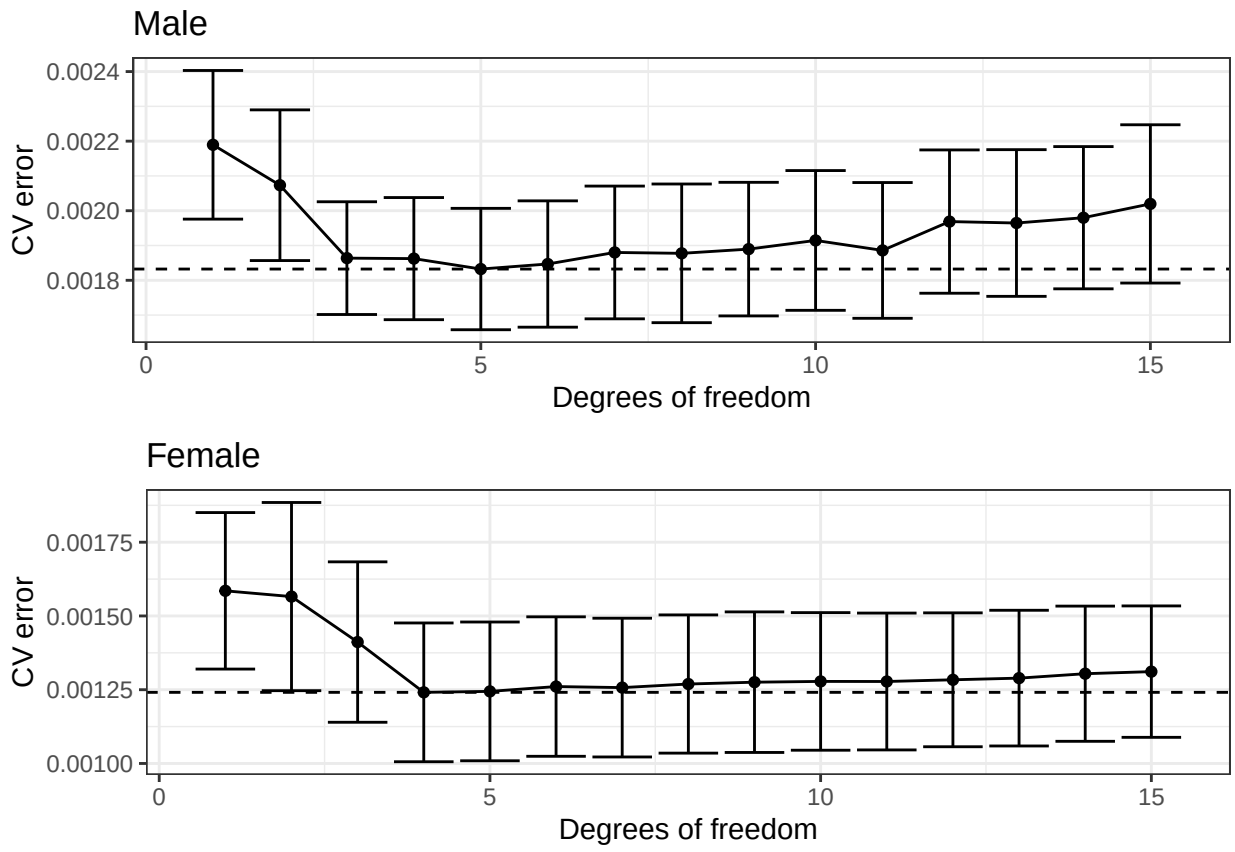

Figure 3: 10-Fold Cross Validation Plots for Males and Females

The degrees of freedom for minimizing values are 5 and 4 for boys and girls respectively. The values obtained from the one standard error rule are 3 for both boys and girls. See figure 3.

Fit splines with degrees of freedom 3 and 5 on scatter plots of relative spinal bone mineral density and age for both male and females. See figure 4.

```r
# define df.min
df.min = max(cv_bmd_male$df.min, cv_bmd_female$df.min)
```

```r
# define df.1se
df.1se = max(cv_bmd_male$df.1se, cv_bmd_female$df.1se)
# plot a scatter plot of `spnbmd` by `age` faceting by `gender` with splines
bmd %>%
  ggplot(aes(x = age, y = spnbmd)) +
  geom_point() +
  geom_smooth(method = "lm",
              formula = "y ~ splines::ns(x, df = df.min)",
              aes(color = factor(df.min)),
              se = FALSE) +
  geom_smooth(method = "lm",
              formula = "y ~ splines::ns(x, df = df.1se)",
              aes(color = factor(df.1se)),
              se = FALSE) +
  facet_wrap(~gender) +
  labs(x = "Age (years)", y = "Relative Spinal Bone Mineral Density",
       color = "Deg. of Freedom") +
  theme_bw()
```

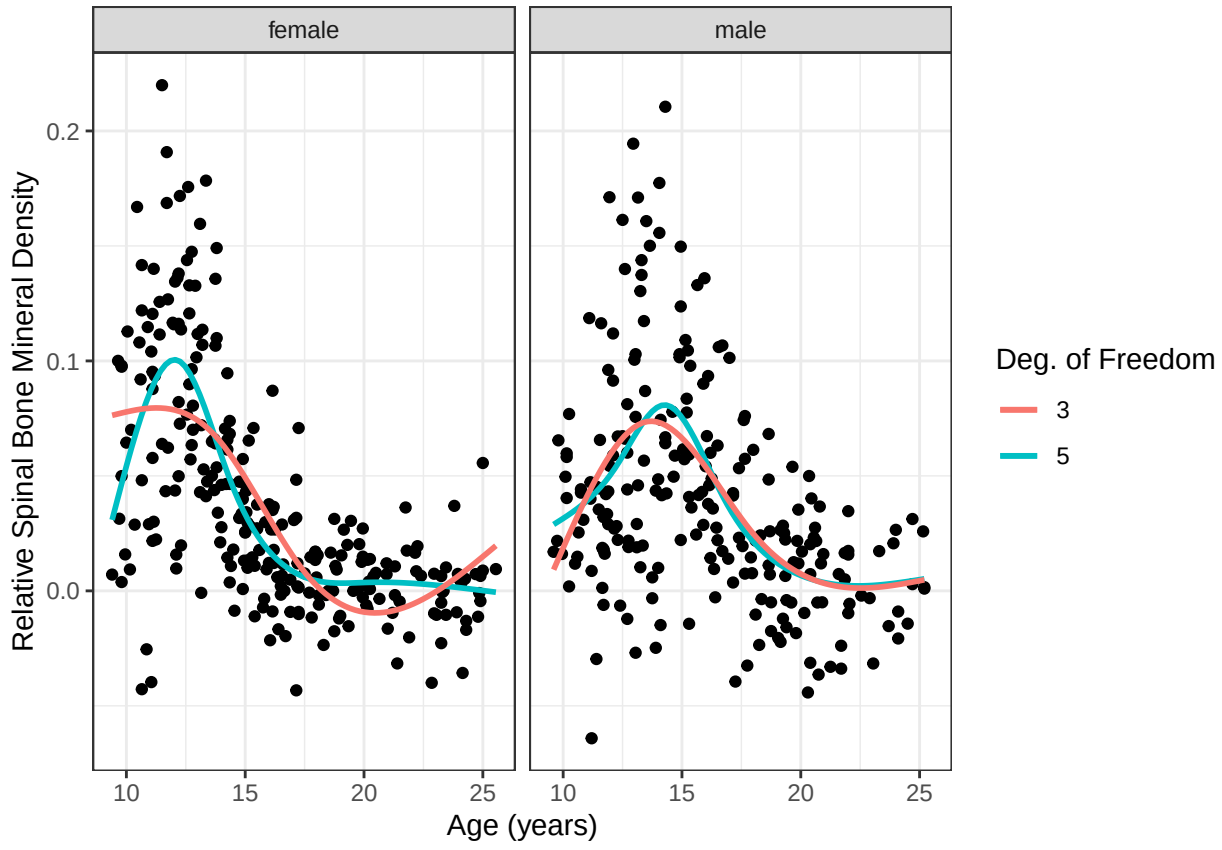

Figure 4: Scatter plots of Relative Spinal Bone Mineral Density by Age for females and males with spline fits.

I think 5 degrees of freedom (`df.min`) makes more sense because the curve is levels off ass the children reach the age of 25. The curve with 3 degrees of freedom for girls has a dip and then rises again as age increases - this is inconsistent with growth curves in reality. See figure 4.

1.3.3 Final fit

- Using the degrees of freedom chosen above, fit final spline models to `bmd_train_male` and `bmd_train_female`.

Solution.

Use `lm` and `splines::ns` to fit the final spline models to the training data. Store the models for males and females in `m.males` and `m.females`.

```
# fit final spline model (with 3 df) on `bmd_train_female`
m.females = lm(formula = spnbmd ~ splines::ns(age, df = df.min), data = bmd_train_female)
# fit final spline model (with 3 df) on `bmd_train_male`
m.males = lm(formula = spnbmd ~ splines::ns(age, df = df.min), data = bmd_train_male)
```

1.4 Evaluate (6 points)

- Using the final models above, answer the following questions for boys and girls separately: What percent of the variation in `spnbmd` is explained by the spline fit in the training data? What is the training RMSE? What is the test RMSE? Print these three metrics in a nice table.
- How do the training and test errors compare? What does this suggest about the extent of overfitting that has occurred?

Solution

Use `lm` to get the model on the test data. Use `summary` to get R^2 and RMSE. Put values in a tibble and display table with `kable`. See Table 2

```
# save summaries of final models and to get R^2 for the model for both boys and girls
results.m.females = summary(m.females)
results.m.males = summary(m.males)
# test final spline model (with 3 df) on `bmd_test_female` and save summary
fm.females = lm(formula = spnbmd ~ splines::ns(age, df = df.min), data = bmd_test_female)
results.fm.females = summary(fm.females)
# test final spline model (with 3 df) on `bmd_test_male`
fm.males = lm(formula = spnbmd ~ splines::ns(age, df = df.min), data = bmd_test_male)
results.fm.males = summary(fm.males)
```

```
# create tibble with R^2 and RMSE for females and display table
r2.rsme.females = tibble(`Percent explained` = results.m.females$adj.r.squared,
                         `Training RMSE` = results.m.females$sigma,
                         `Test RMSE` = results.fm.females$sigma)
r2.rsme.females %>%
   kable(format = "latex", row.names = NA,
      booktabs = TRUE,
      caption = "Final Model - Females") %>%
  kable_styling(position = "center")
```

For females, 49.8% of the variation in `spnbmd` is explained by the model in the training data. The training RMSE and test RMSE are 0.034 and 0.038 respectively. This test error is pretty close but the training RMSE being smaller shows that there could have been some overfitting in the training data if any. See Table 2

Table 2: Final Model - Females

Percent explained	Training RMSE	Test RMSE
0.498	0.034	0.038

Table 3: Final Model - Males

Percent explained	Training RMSE	Test RMSE
0.283	0.042	0.034

```
# create tibble with R^2 and RMSE for males and display table
r2.rsme.males = tibble(`Percent explained` = results.m.males$adj.r.squared,
                       `Training RMSE` = results.m.males$sigma,
                       `Test RMSE` = results.fm.males$sigma)
r2.rsme.males %>%
   kable(format = "latex", row.names = NA,
      booktabs = TRUE,
      caption = "Final Model - Males") %>%
  kable_styling(position = "center")
```

Use `lm` to get the model on the test data. Use `summary` to get R^2 and RMSE. Put values in a tibble and display table with `kable`. For males, 28.3% of the variation in `spnbmd` is explained by the model in the training data. The training RMSE and test RMSE are 0.042 and 0.034 respectively. This test error is pretty close but the training RMSE being larger shows that there could have been some underfitting in the training data. See Table 3

1.5 Interpret (7 points)

- Using the degrees of freedom chosen above, redo the scatter plot with the overlaid spline fits, this time without faceting in order to directly compare the spline fits for boys and girls. Instead of faceting, distinguish the genders by color.
- The splines help us see the trend in the data much more clearly. Eyeballing these fitted curves, answer the following questions. At what ages (approximately) do boys and girls reach the peaks of their growth spurts? At what ages does growth largely level off for boys and girls? Do these seem in the right ballpark?

Solution

Plot a scatter plot of `spnbmd` by `age` with spline fits and coloring by `gender`.

```
# plot a scatter plot of `spnbmd` by `age` with spline fits and coloring by `gender`
bmd_train %>%
  ggplot(aes(x = age, y = spnbmd, color = gender)) +
  geom_point() +
  geom_smooth(method = "lm",
              formula = "y ~ splines::ns(x, df = df.min)",
              aes(color = factor(gender)),
              data = bmd_train_female,
              se = FALSE) +
```

```
  geom_smooth(method = "lm",
              formula = "y ~ splines::ns(x, df = df.min)",
              aes(color = factor(gender)),
              data = bmd_train_male,
              se = FALSE) +
  labs(x = "Age (years)", y = "Relative Spinal Bone Mineral Density",
       color = "Gender") +
  theme_bw()
```

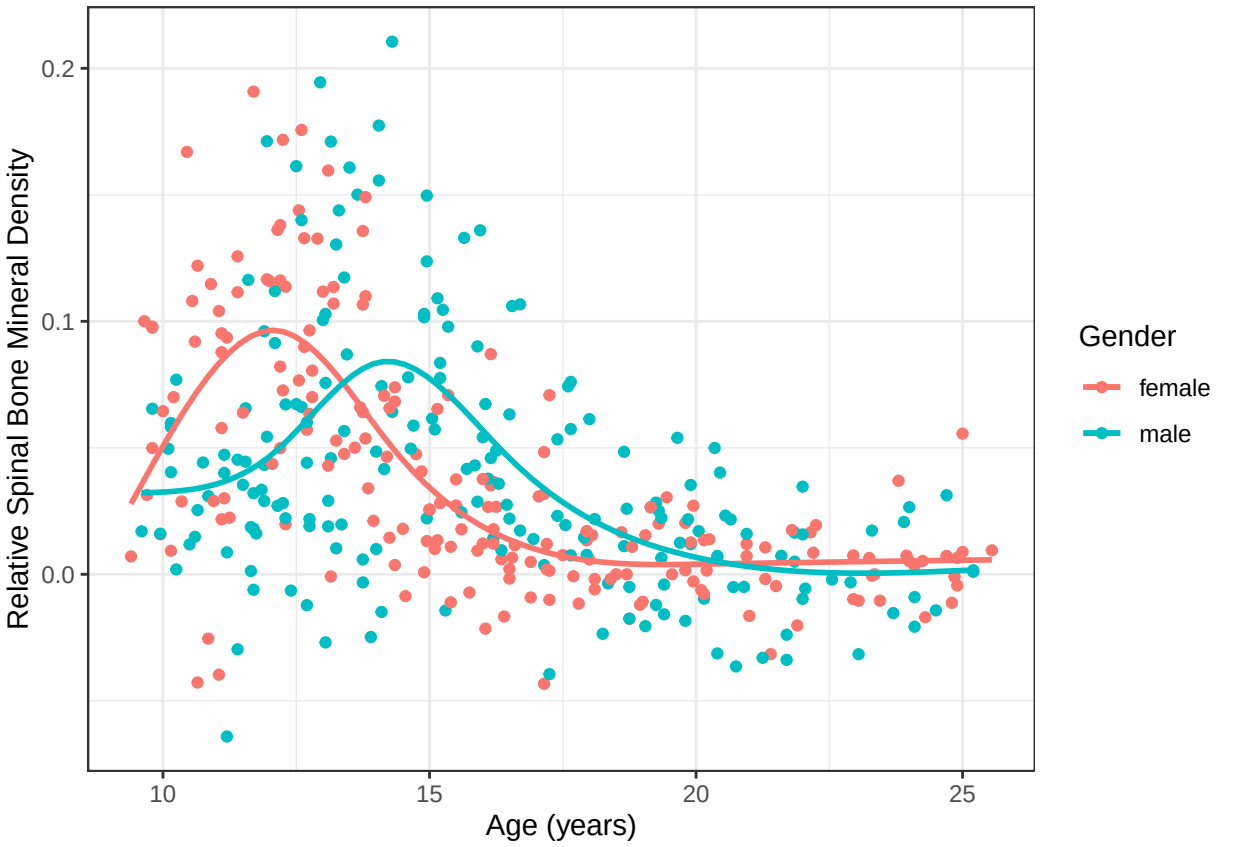

Figure 5: Scatter plots of Relative Spinal Bone Mineral Density by Age for colored by females and males with spline fits for each.

Girls reach the peaks of their growth spurts at approximately 12 years and boys at approximately 14. Girls' growth largely level off around 16 and boys' at around 20. These ages seem to be in the right ballpark since girls often hit their growth spurts before boys. See Figure 5

2 KNN and bias-variance tradeoff (45 points for correctness; 5 points for presentation)

Setup: Apple farming

You own a square apple orchard, measuring 200 meters on each side. You have planted trees in a grid ten meters apart from each other. Last apple season, you measured the yield of each tree in your orchard (in

average apples per week). You noticed that the yield of the different trees seems to be higher in some places of the orchard and lower in others, perhaps due to differences in sunlight and soil fertility across the orchard.

Unbeknownst to you, the yield Y of the tree planted X_1 meters to the right and X_2 meters up from the bottom left-hand corner of the orchard has distribution

$$Y = 50 + 0.001X_1^2 + 0.001X_2^2 + \epsilon, \quad \epsilon \sim N(0, \sigma^2), \quad \sigma = 4.$$

The data you collected are as in Figure 6.

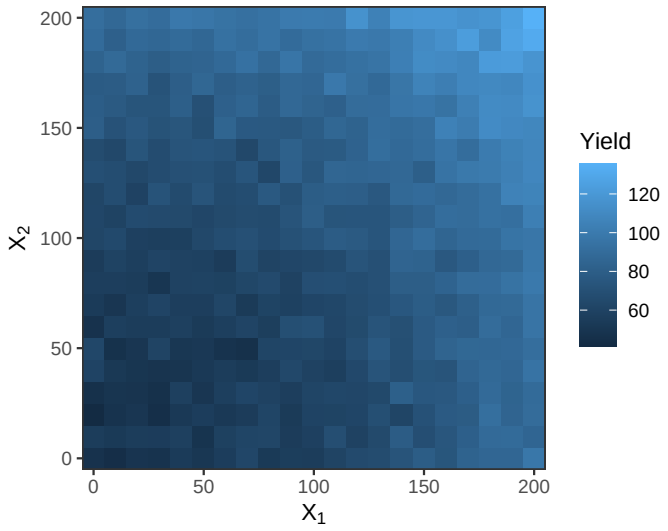

Figure 6: Apple tree yield for each 10m by 10m block of the orchard in a given year.

The underlying trend is depicted in Figure 7, with the top right-hand corner of the orchard being more fruitful.

2.1 A simple rule to predict this season's yield (15 points)

This apple season is right around the corner, and you'd like to predict the yield of each tree. You come up with perhaps the simplest possible prediction rule: predict this year's yield for any given tree based on last year's yield from that same tree. Without doing any programming, answer the following questions:

- What is the expected training error of such a rule?
- Averaged across all trees, what is the squared bias, variance, and ETE of this prediction rule?
- Why is this not the best possible prediction rule?

Solution. note: I use the terms "bias" and "squared bias" interchangeably.

1. Given that the model is based on last year's yield, which is the training data, the expected training error of this rule is 0 because the model fits to each training data point.

2. Averaged across all trees, the square bias is 0 because this prediction rule is based off of the underlying yield equation shown above and on average is just the underlying trend depicted in Figure 7. The variance $\epsilon \sim N(0, \sigma^2), \quad \sigma = 4$ because the model is 1 error term away from

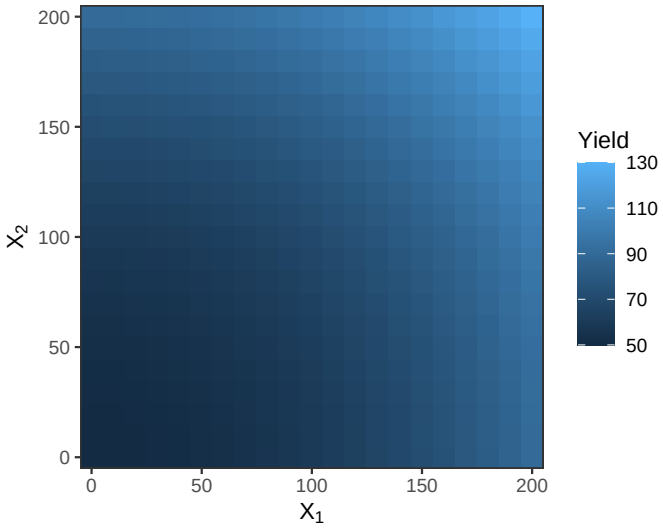

Figure 7: Underlying trend in apple yield for each 10m by 10m block of the orchard.

the truth and the test data will have another 1 layer of error away from the truth. The ETE is therefore the sum of the squared bias, variance, and the irreducible error: $ETE = 2 \times \epsilon = 2 \times \sigma^2$**.**

3. This is not the best possible prediction because it only takes into account the one specific tree and not the other surrounding trees that could better account for general trends of soil and/or sunlight differences by area. Just basing the prediction on one tree will lead to overfitting the model to the noise of the training data.

2.2 K-nearest neighbors regression (conceptual) (15 points)

As a second attempt to predict a yield for each tree, you average together last year's yields of the K trees closest to it (including itself, and breaking ties randomly if necessary). So if you choose $K = 1$, you get back the simple rule from the previous section. This more general rule is called *K-nearest neighbors (KNN) regression* (see ISLR p. 105).

KNN is not a parametric model like linear or logistic regression, so it is a little harder to pin down its degrees of freedom.

- What happens to the model complexity as K increases? Why?
- The degrees of freedom for KNN is sometimes considered n/K, where n is the training set size. Why might this be the case? [Hint: consider a situation where the data are clumped in groups of K.]
- Conceptually, why might increasing K tend to improve the prediction rule? What does this have to do with the bias-variance tradeoff?
- Conceptually, why might increasing K tend to worsen the prediction rule? What does this have to do with the bias-variance tradeoff?

Solution. note: I use the terms "bias" and "squared bias" interchangeably.

1. As K **increases, the model will become less complex. This is because the prediction for each point will become less "specialized" based on just that point and will depend more on general**

trends of neighboring points. The most extreme case is to increase K to include all points. This would be the most simple model because the prediction would just be the average of all points.

2. From the spline models we said that the number of degrees of freedom is the number of coefficients. We can think degrees of freedom for the KNN model in a similar way. First think about the extreme scenarios where $K = 1$ and $K = n$. When $K = n$ then the prediction is the same for all of the points (the average) and thus it's like there is one coefficient for the model and $df = 1$. When $K = 1$ each point has a separate prediction so there has to be a unique coefficient for each point and thus $df = n$. One can think of a case in between, where $K = 3$, for example, and each group of three points (more or less) needs a coefficient in the model.

3. Increasing K can improve the prediction model if the current model overfits the training data. In the simple model example from the previous question, the model overfit the training model with $K = 1$. This meant that there was low bias but high variance. In this extreme case increasing K, or making the model more general/simple, would decrease variance (probably) more than it would increase bias.

4. Decreasing K can improve the prediction model if the current model underfits or is too simplified. In an extreme model where $K = n$, bias is very high ad variance is low. In this extreme case decreasing K, or making the model more complex, would decrease bias (probably) more than it would increase variance.

2.3 K-nearest neighbors regression (simulation) (15 points)

Now, we try KNN for several values of K. For each, we compute the bias, variance, and ETE for each value based on 50 resamples. The code for this simulation, provided for you below (see Rmd file; code omitted from PDF for brevity), results in Figure 8.

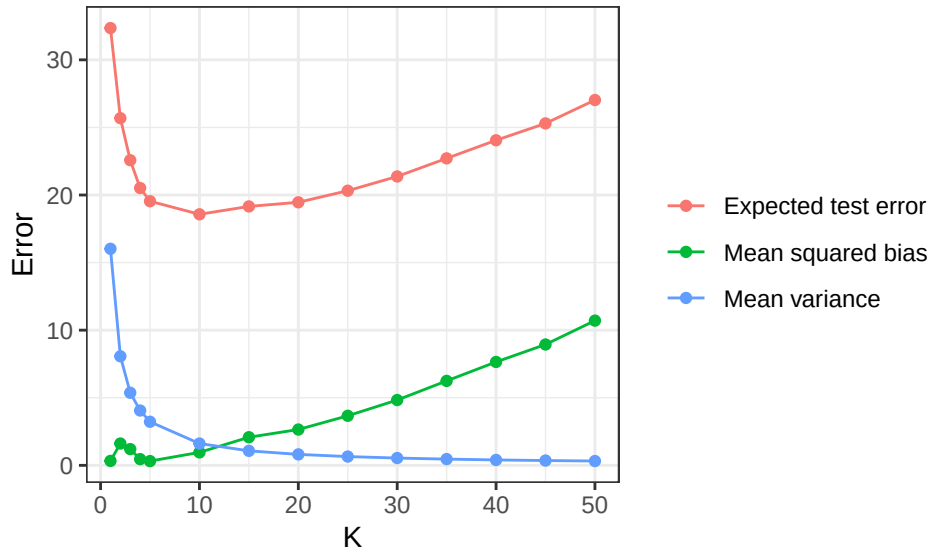

Figure 8: Bias-variance trade-off for KNN regression.

- Based on Figure 8, what is the optimal value of `K`?
- We are used to the bias decreasing and the variance increasing when going from left to right in the plot. Here, the trend seems to be reversed. Why is this the case?

Table 4: Tree with Highest Absolute Bias

K	X1	X2	Abosolute Bias
50	200	200	20.6

- The squared bias has a strange bump between `K` $= 1$ and `K` $= 5$, increasing from `K` $= 1$ to `K` $= 2$ but then decreasing from `K` $= 2$ to `K` $= 5$. Why does this bump occur? [Hint: Think about the rectangular grid configuration of the trees. So for a given tree, the closest tree is itself, and then the next closest four trees are the ones that are one tree up, down, left, and right from it.]
- The data frame `training_results_summary` contains the bias and variance for every tree in the orchard, for every value of `K`. Which tree and which value of `K` gives the overall highest absolute bias? Does the sign of the bias make sense? Why do this particular tree and this particular value of `K` give us the largest absolute bias?
- Redo the bias-variance plot above, this time putting `df = n/K` on the x-axis. What do we notice about the variance as a function of `df`? Derive a formula for the KNN variance and superimpose this formula onto the plot as a dashed curve. Do these two variance curves match? [Hint: To derive the KNN variance, focus first on the prediction of a single tree. Recall the facts that the variance of the sum of independent random variables is the sum of their variances, and that the variance of a constant multiple of a random variable is the square of that constant times its variance.]

Solution. note: I sometimes use the terms "bias" and "squared bias" interchangeably.

1. The optimal value of K is when the expected test error is the smallest. This is when $K = 10$.

2. Usually the bias decreases and the variance increases when going from left to right because usually (with splines) the x-axis is the degrees of freedom and the complexity of the model increases from left to right. But with KNN models, the x-axis is K so the complexity of the model decreases from left to right.

3. This bump in squared bias occurs because of how the trees are arranged - in a grid pattern. When $K = 2$ the bias increases because having a pair of closest neighbors for prediction, will create the prediction to be "off" from the actual point. This is because the second neighbor will be either directly above, below, to the left, or, to the right of the first neighbor. One can see how making a prediction based on these two points can skew the prediction fo the actual point off by a bit (either skew it above, below, to the right, or to the left). The case with $K = 3$ and $K = 4$ is similar because the other neighbors closest to the first neighbor will still create a skew. It's not until $K = 5$ where the first neighbor will be equally surrounded by four neighbors and thus the skew decreases.

```
# find the tree and value of `K` in `training_results_summary` with the
#highest absolute bias and display in a table
training_results_summary %>%
  mutate(abs_bias = abs(bias)) %>%
  arrange(desc(abs_bias)) %>%
  slice_head(n = 1) %>%
  select(K, X1, X2, `Abosolute Bias` = abs_bias) %>%
  kable(format = "latex", row.names = NA,
      booktabs = TRUE,
      caption = "Tree with Highest Absolute Bias") %>%
  kable_styling(position = "center")
```

4. The tree and value of K that give the overall highest absolute bias is when $K = 50$, $X1 = 200$, and $X2 = 200$. In this case the absolute bias is 20.6. The sign of the bias is negative and this

makes sense based on the model because the tree at $X1 = 200$ and $X2 = 200$ has (on average) the highest yield. Thus, when this point is predicted with K trees in consideration, the prediction will usually be less than just the yield of that one tree since all the other trees in the prediction will usually have a smaller yield. This is also why this point has $K = 50$, since this includes the most trees that are smaller than the point in the prediction. See Table 4

```
# find n (the number of trees in the orchard)
n = (orchard_width/tree_distance + 1)^2
# plot the bias, variance, and ETE
overall_results %>%
  pivot_longer(-K, names_to = "metric", values_to = "Error") %>%
  mutate(metric = fct_recode(metric,
                             "Expected test error" = "expected_test_error",
                             "Mean squared bias" = "mean_sq_bias",
                             "Mean variance" = "mean_variance")) %>%
  ggplot(aes(x = n/K, y = Error, colour = metric)) +
  geom_line() +
  geom_point() +
  geom_abline(intercept = 0, slope = (sigma^2)/n, linetype = "dashed") +
  labs(x = "n/K") +
  theme_bw() +
  theme(legend.title = element_blank())
```

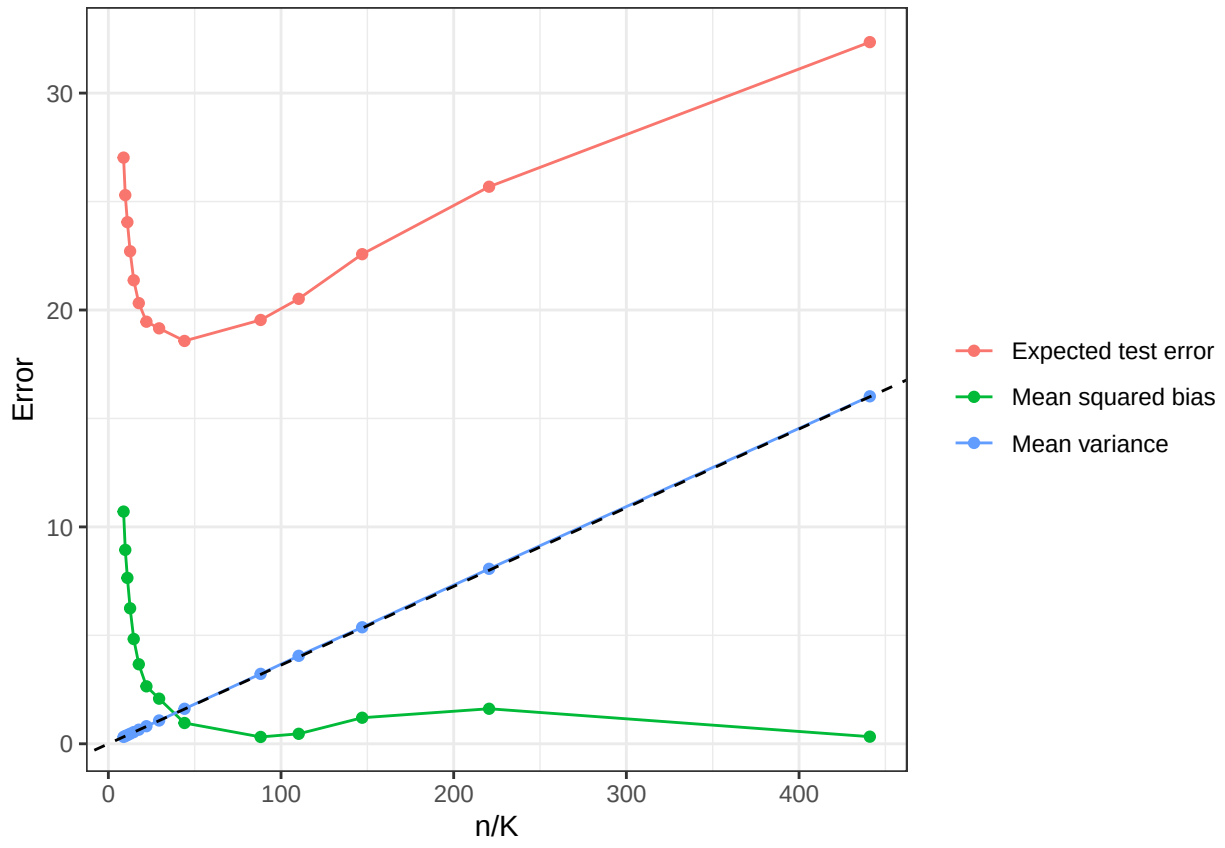

Figure 9: Bias-variance trade-off for KNN regression with n/K on x-axis.

5. We can see that the variance as a function of `df` is similar to what we've seen in the past with

spline models - a positive linear function. We can derive the formula for the KNN variance. Based on `overall_results`. The mean variance for all trees for a value K is $\frac{\sigma^2}{K}$. We know that the variance of the sum of independent random variables is the sum of their variances. Thus we can sum $\frac{\sigma^2}{K}$, K times to get $\frac{K\sigma^2}{K}$ and finally σ^2. Now we can take the average of this across all n trees to get a final variance of $\frac{\sigma^2}{n}$. This formula for variance is superimposed on the plot with a dashed line. These variance curves match. See Figure 9