DSC1107_FA4

John Benedict A. Monfero

March 5, 2025

Case study: Bone mineral density

In this exercise, we will be looking at a data set (given in `bmd-data.xlsx`) on spinal bone mineral density, a physiological indicator that increases during puberty when a child grows. In this dataset, `idnum` is an identifier for each child and `spnbmd` represents the relative change in spinal bone mineral density between consecutive doctor's visits.

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

Import

```
 # 1.   Using the readxl package, import the data into a tibble called bmd_raw
bmd_raw <- read_excel('bmd-data.xlsx')

 # 2.   Print the imported tibble.
print(bmd_raw)
```

```
## # A tibble: 169 × 9
##    idnum   age sex   fracture weight_kg height_cm medication waiting_time spnbmd
##    <dbl> <dbl> <chr> <chr>        <dbl>     <dbl> <chr>             <dbl>  <dbl>
##  1   469  57.1 F     no frac…        64      156. Anticonvu…           18  0.879
##  2  8724  75.7 F     no frac…        78      162  No medica…           56  0.795
##  3  6736  70.8 M     no frac…        73      170. No medica…           10  0.907
##  4 24180  78.2 F     no frac…        60      148  No medica…           14  0.711
##  5 17072  54.2 M     no frac…        55      161  No medica…           20  0.791
##  6  3806  77.2 M     no frac…        65      168  No medica…            7  0.730
##  7 17106  56.2 M     no frac…        77      159  No medica…           26  1.01
##  8 23834  49.9 F     no frac…        59      150  No medica…            9  0.731
##  9  2454  68.4 M     no frac…        64      167  Glucocort…            6  0.689
## 10  2088  66.3 M     no frac…        72      160. No medica…           10  0.947
## # ℹ 159 more rows
```

Tidy

Comment on the layout of the data in the tibble

- The dataset `bmd_raw` has **169 observations**, each identified by a unique `idnum`. Each observation includes eight characteristics: `age`, `sex`, `fracture`, `weight_kg`, `height_cm`, `medication`, `waiting_time`, and `spnbmd`.

What should be the variables in the data?

The variables in the data are:

```
glimpse(bmd_raw)
```

```
## Rows: 169
## Columns: 9
## $ idnum        <dbl> 469, 8724, 6736, 24180, 17072, 3806, 17106, 23834, 2454, …
## $ age          <dbl> 57.05277, 75.74122, 70.77890, 78.24718, 54.19188, 77.1777…
## $ sex          <chr> "F", "F", "M", "F", "M", "M", "M", "F", "M", "M", "M", "F…
## $ fracture     <chr> "no fracture", "no fracture", "no fracture", "no fracture…
## $ weight_kg    <dbl> 64.0, 78.0, 73.0, 60.0, 55.0, 65.0, 77.0, 59.0, 64.0, 72.…
## $ height_cm    <dbl> 155.5, 162.0, 170.5, 148.0, 161.0, 168.0, 159.0, 150.0, 1…
## $ medication   <chr> "Anticonvulsant", "No medication", "No medication", "No m…
## $ waiting_time <dbl> 18, 56, 10, 14, 20, 7, 26, 9, 6, 10, 12, 5, 11, 28, 73, 1…
## $ spnbmd       <dbl> 0.8793, 0.7946, 0.9067, 0.7112, 0.7909, 0.7301, 1.0096, 0…
```

- `idnum` : Unique identifier for each observation
- `age` : Age of the individual
- `sex` : Gender of the individual
- `fracture` : Fracture status
- `weight_kg` : Weight in kilograms
- `height_cm` : Height in centimeters
- `medication` : Type of medication
- `waiting_time` : Waiting time in days
- `spnbmd` : Spinal bone mineral density

What operation is necessary to get it into tidy format?

To tidy the data, we need to:

- Arrange idnum in ascending order.
- Rename age to `age_month` to reflect `age` in months
- Rename `waiting_time` to `waiting_time_minutes` to reduce ambiguity
- Convert `sex` , `fracture` , and `medication` to factors

Apply this operation to the data, storing the result in a tibble called bmd.

```
bmd <- bmd_raw %>%
  arrange(idnum) %>%
  mutate(
    age_month = floor(age),
    waiting_time_minutes = waiting_time,
    sex = as.factor(sex),
    fracture = as.factor(fracture),
    medication = as.factor(medication)
  ) %>%
  select(idnum, age_month, sex, fracture, weight_kg, height_cm, medication, waiting_time_minutes, spnbm
d)

glimpse(bmd)
```

```
## Rows: 169
## Columns: 9
## $ idnum                <dbl> 35, 55, 67, 69, 71, 77, 87, 103, 109, 134, 145, 1…
## $ age_month            <dbl> 56, 73, 76, 67, 69, 77, 83, 77, 69, 49, 71, 65, 8…
## $ sex                  <fct> F, F, F, F, F, F, F, F, F, F, F, F, F, F, F, M, M…
## $ fracture             <fct> no fracture, no fracture, fracture, no fracture, …
## $ weight_kg            <dbl> 68.0, 52.0, 52.0, 62.0, 68.5, 50.0, 51.0, 61.0, 7…
## $ height_cm            <dbl> 150.5, 153.0, 146.0, 159.0, 165.0, 152.0, 150.0, …
## $ medication           <fct> No medication, No medication, Glucocorticoids, No…
## $ waiting_time_minutes <dbl> 18, 89, 20, 30, 25, 24, 9, 7, 11, 60, 59, 14, 20,…
## $ spnbmd               <dbl> 0.7171, 0.7128, 0.5603, 1.0176, 0.8664, 0.6122, 0…
```

Explore

> What is the total number of children in this dataset? What are the number of boys and girls? What are the median ages of these boys and girls?

```
total_number_children <- bmd %>% summarise(total = n())

total_gender_number <- bmd %>% count(sex)

median_gender_groups <- bmd %>% group_by(sex) %>% summarise(median_age = median(age_month))

# Print results
total_number_children
```

```
## # A tibble: 1 × 1
##   total
##   <int>
## 1   169
```

```
total_gender_number
```

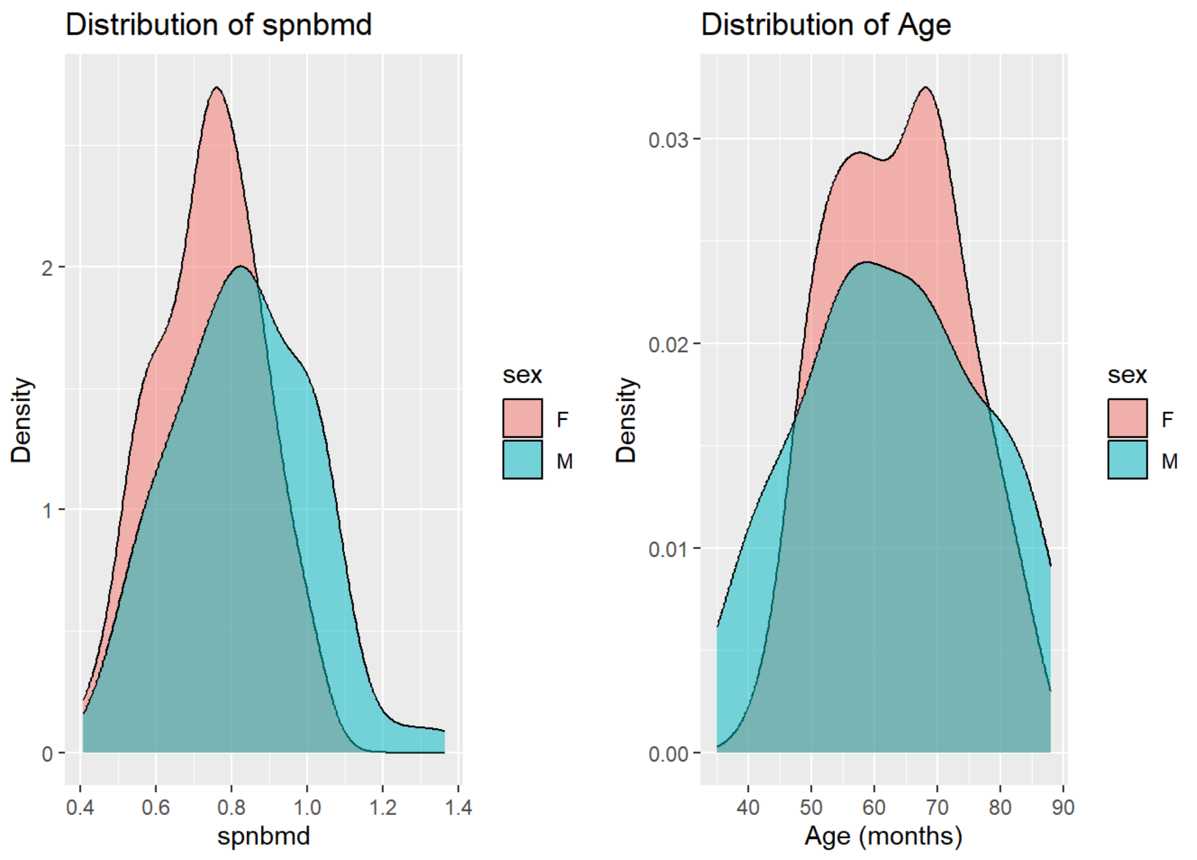
```
## # A tibble: 2 × 2
##   sex       n
##   <fct> <int>
## 1 F        83
## 2 M        86
```

```
median_gender_groups
```

```
## # A tibble: 2 × 2
##   sex   median_age
##   <fct>      <dbl>
## 1 F             63
## 2 M             63
```

Produce plots 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?

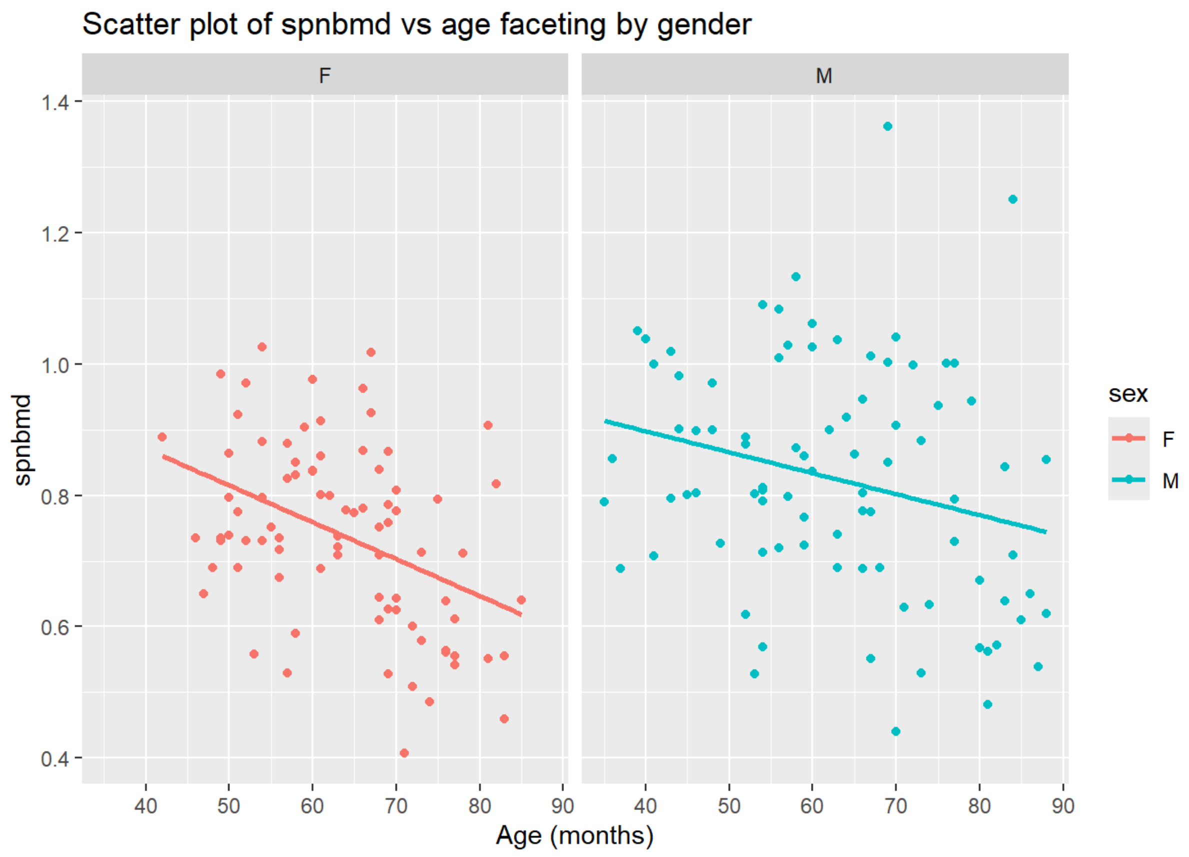
```
# Plot distributions of spnbmd and age between boys and girls
p1 <- ggplot(bmd, aes(x = spnbmd, fill = sex)) +
  geom_density(alpha = 0.5) +
  labs(title = "Distribution of spnbmd", x = "spnbmd", y = "Density")

p2 <- ggplot(bmd, aes(x = age_month, fill = sex)) +
  geom_density(alpha = 0.5) +
  labs(title = "Distribution of Age", x = "Age (months)", y = "Density")

# Display plots side by side
plot_grid(p1, p2, ncol = 2)
```



Create a scatter plot of spnbmd (y axis) versus age (x axis), faceting by gender. What trends do you see in this data?

```
# Scatter plot of spnbmd vs age faceting by gender
ggplot(bmd, aes(x = age_month, y = spnbmd, color = sex)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ sex) +
  labs(title = "Scatter plot of spnbmd vs age faceting by gender", x = "Age (months)", y = "spnbmd")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Model

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.

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.

```
set.seed(5) # seed set for reproducibility (DO NOT CHANGE)
n <- nrow(bmd)
train_samples <- sample(1:n, round(0.8*n))

bmd_train <- bmd %>% slice(train_samples)
bmd_test <- bmd %>% slice(-train_samples)
```

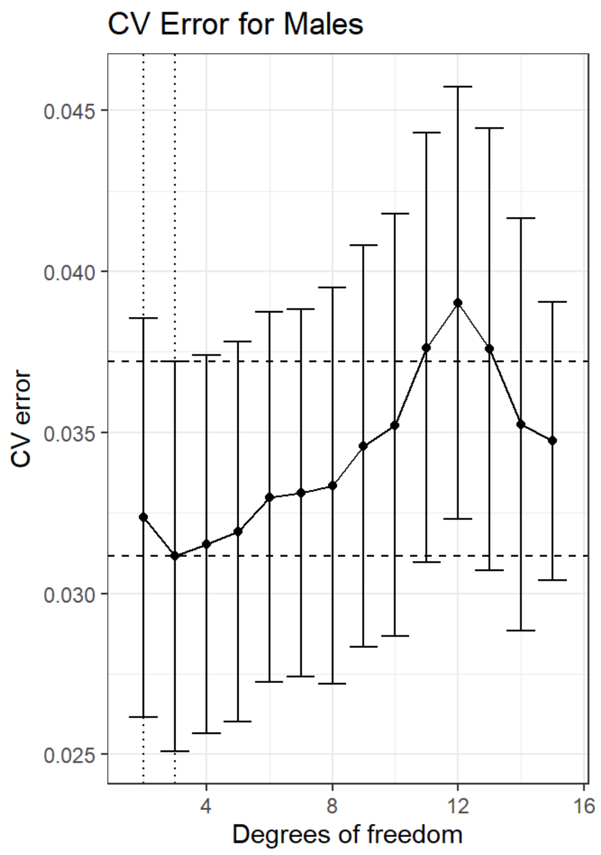
Tune

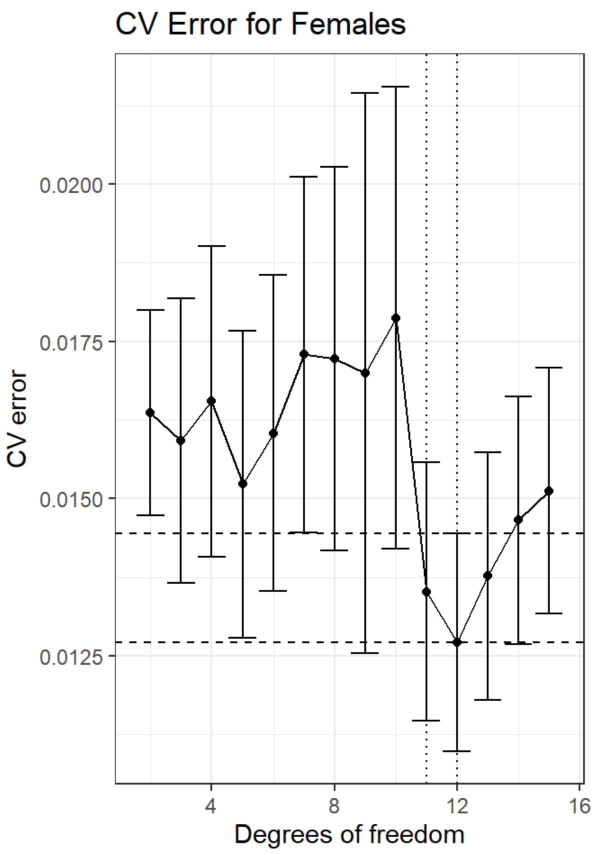
Since the trends in `spnbmd` look somewhat different for boys than for girls, we might want to fit separate splines to these two groups. Separate `bmd_train` into `bmd_train_male` and `bmd_train_female`, and likewise for `bmd_test`.

```
bmd_train_male <- bmd_train %>% filter(sex == "M")
bmd_train_female <- bmd_train %>% filter(sex == "F")

bmd_test_male <- bmd_test %>% filter(sex == "M")
bmd_test_female <- bmd_test %>% filter(sex == "F")
```

Using `cross_validate_spline` from the `stat471` R package, 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.

```
# Cannot apply degrees of freedom == 1, the argument is too small for both train data sets
cv_male <- cross_validate_spline(bmd_train_male$age_month, bmd_train_male$spnbmd, df = 2:15, nfolds = 1
0)
cv_female <- cross_validate_spline(bmd_train_female$age_month, bmd_train_female$spnbmd, df = 2:15, nfold
s = 10)

# Note: When the function `cross_validate_spline()` by `library(stat471)` executes without any pending e
rrors, it outputs successfully an object with the following attributes:
    # Given variable name: `cv_male`, the output of `cross_validate_spline()` would be:
      # cv_male$cv_table
      # cv_male$cv_plot
      # cv_male$df.1se
      # cv_male$$df.min

# Provided the instructions above, we will extract their CV plots
p1 <- cv_male$cv_plot + labs(title = "CV Error for Males")
p2 <- cv_female$cv_plot + labs(title = "CV Error for Females")

plot_grid(p1, p2, ncol = 2)
```





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?

```
# Determining the df (degrees of freedom) and corresponding se (standard errors)
df_min_male <- cv_male$df.min
df_min_female <- cv_female$df.min

df_1se_male <- cv_male$df.1se
df_1se_female <- cv_female$df.1se
```

```
cat("Degrees of Freedom Minimizing CV Error for Males:", df_min_male, "\n")
```

```
## Degrees of Freedom Minimizing CV Error for Males: 3
```

```
cat("Degrees of Freedom Minimizing CV Error for Females:", df_min_female, "\n")
```

```
## Degrees of Freedom Minimizing CV Error for Females: 12
```

```
cat("Degrees of Freedom from One Standard Error Rule for Males:", df_1se_male, "\n")
```

```
## Degrees of Freedom from One Standard Error Rule for Males: 2
```

```
cat("Degrees of Freedom from One Standard Error Rule for Females:", df_1se_female, "\n")
```

```
## Degrees of Freedom from One Standard Error Rule for Females: 11
```

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`.

```
df_min <- max(df_min_male, df_min_female)
df_1se <- max(df_1se_male, df_1se_female)

cat("df.min:", df_min, "\n")
```

```
## df.min: 12
```

```
cat("df.1se:", df_1se, "\n")
```

```
## df.1se: 11
```

Given your intuition for what growth curves look like, which of these two values of the degrees of freedom makes more sense?

```
cat("The value we obtained in variable `df.min`:", df_min, "\n\n\n","It ensures and minimizes the cross-
validation error, suggesting it provides the best fit to the training data. It might capture more nuance
s in the data, but it could also risk overfitting, especially if the data has substantial variability.",
"\n\n\n")
```

```
## The value we obtained in variable `df.min`: 12
##
##
##  It ensures and minimizes the cross-validation error, suggesting it provides the best fit to the trai
ning data. It might capture more nuances in the data, but it could also risk overfitting, especially if
the data has substantial variability.
```

```
cat("Nevertheless, the value we obtained in variable `df.1se`:", df_1se, "\n\n\n","Generally derives fro
m the one standard error rule, which typically provides a more conservative and bias-variance trade-off
fit. It balances the model complexity and generalizability, reducing the risk of overfitting while still
capturing the main trends.")
```

```
## Nevertheless, the value we obtained in variable `df.1se`: 11
##
##
##  Generally derives from the one standard error rule, which typically provides a more conservative and
bias-variance trade-off fit. It balances the model complexity and generalizability, reducing the risk of
overfitting while still capturing the main trends.
```

Intuition for Growth Curves

- Growth curves generally follow smooth, gradual changes rather than highly complex patterns. Therefore, a slightly lower degree of freedom (df.1se = 11) might make more sense as it avoids overfitting and ensures the model remains generalizable to new data.

Using df.1se (11) is likely the better choice for modeling growth curves, as it provides a balance between fitting the data well and maintaining simplicity and generalizability.

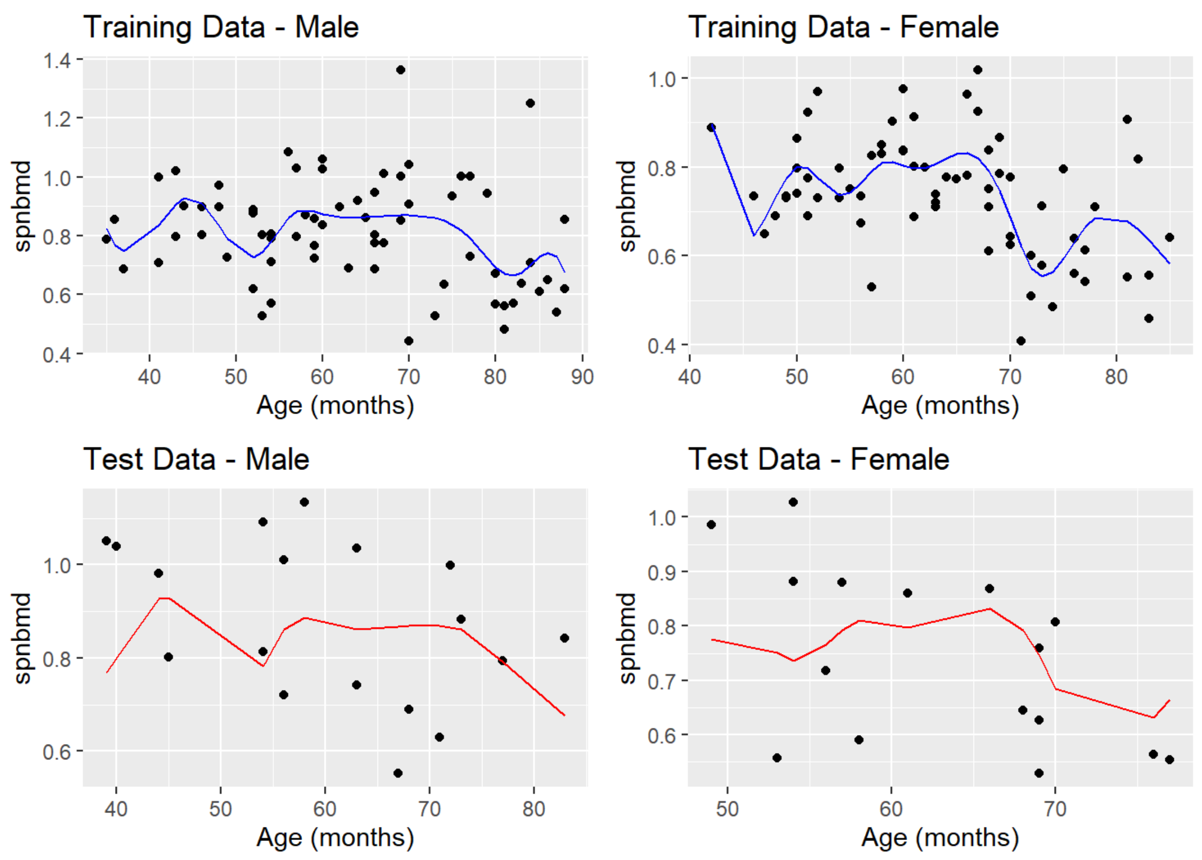
Final Fit

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

```r
library(splines)

# Fit final spline models using df_1se (11)
df_1se <- 11

# Fit spline model for males
model_male <- lm(spnbmd ~ splines::bs(age_month, df = df_1se), data = bmd_train_male)
model_female <- lm(spnbmd ~ splines::bs(age_month, df = df_1se), data = bmd_train_female)

# Predict on training data
train_pred_male <- predict(model_male, bmd_train_male)
train_pred_female <- predict(model_female, bmd_train_female)

# Predict on test data
test_pred_male <- predict(model_male, bmd_test_male)
test_pred_female <- predict(model_female, bmd_test_female)
```

Graphing the Final Fit Results:

```
# Create data frames for plotting
train_male_plot <- bmd_train_male %>%
  mutate(predicted_spnbmd = train_pred_male) %>%
  select(age_month, spnbmd, predicted_spnbmd) %>%
  mutate(type = "Training", gender = "Male")

train_female_plot <- bmd_train_female %>%
  mutate(predicted_spnbmd = train_pred_female) %>%
  select(age_month, spnbmd, predicted_spnbmd) %>%
  mutate(type = "Training", gender = "Female")

test_male_plot <- bmd_test_male %>%
  mutate(predicted_spnbmd = test_pred_male) %>%
  select(age_month, spnbmd, predicted_spnbmd) %>%
  mutate(type = "Test", gender = "Male")

test_female_plot <- bmd_test_female %>%
  mutate(predicted_spnbmd = test_pred_female) %>%
  select(age_month, spnbmd, predicted_spnbmd) %>%
  mutate(type = "Test", gender = "Female")

# Combine data frames
plot_data <- bind_rows(train_male_plot, train_female_plot, test_male_plot, test_female_plot)

# Create individual plots
p1 <- ggplot(plot_data %>% filter(type == "Training" & gender == "Male"), aes(x = age_month, y = spnbm
d)) +
  geom_point() +
  geom_line(aes(y = predicted_spnbmd), color = "blue") +
  labs(title = "Training Data - Male", x = "Age (months)", y = "spnbmd")

p2 <- ggplot(plot_data %>% filter(type == "Training" & gender == "Female"), aes(x = age_month, y = spnbm
d)) +
  geom_point() +
  geom_line(aes(y = predicted_spnbmd), color = "blue") +
  labs(title = "Training Data - Female", x = "Age (months)", y = "spnbmd")

p3 <- ggplot(plot_data %>% filter(type == "Test" & gender == "Male"), aes(x = age_month, y = spnbmd)) +
  geom_point() +
  geom_line(aes(y = predicted_spnbmd), color = "red") +
  labs(title = "Test Data - Male", x = "Age (months)", y = "spnbmd")

p4 <- ggplot(plot_data %>% filter(type == "Test" & gender == "Female"), aes(x = age_month, y = spnbmd))
+
  geom_point() +
  geom_line(aes(y = predicted_spnbmd), color = "red") +
  labs(title = "Test Data - Female", x = "Age (months)", y = "spnbmd")

# Arrange plots in a 2x2 grid
plot_grid(p1, p2, p3, p4, ncol = 2, nrow = 2)
```



Evaluate

Using the final models above, answer the following questions for boys and girls separately: What is the training `RMSE` ? What is the test `RMSE` ? Print these metrics in a nice table (Use kable).

```r
# Calculate RMSE
train_rmse_male <- sqrt(mean((bmd_train_male$spnbmd - train_pred_male)^2))
train_rmse_female <- sqrt(mean((bmd_train_female$spnbmd - train_pred_female)^2))

test_rmse_male <- sqrt(mean((bmd_test_male$spnbmd - test_pred_male)^2))
test_rmse_female <- sqrt(mean((bmd_test_female$spnbmd - test_pred_female)^2))

# Print RMSE in a table
rmse_table <- tibble(
  Gender = c("Male", "Female"),
  Training_RMSE = c(train_rmse_male, train_rmse_female),
  Test_RMSE = c(test_rmse_male, test_rmse_female)
)

rmse_table %>%
  kable() %>%
  kable_styling()
```

Gender	Training_RMSE	Test_RMSE
Male	0.1635210	0.1878212
Female	0.1015948	0.1512522

How do the `training` and `test` errors compare? What does this suggest about the extent of *overfitting* that has occurred?

Analysis

- For males, the training RMSE is `0.1635` , while the test RMSE is `0.1878` .
- For females, the training RMSE is `0.1016` , while the test RMSE is `0.1513` .

Observations

- The test RMSE is slightly higher than the training RMSE for both males and females. This increase in error from training to test data suggests that the model fits the training data well but has a slightly higher error when predicting new, unseen data.

Conclusion

The difference between training and test RMSE indicates a **moderate level of overfitting**. The model captures the trends in the training data but does not generalize perfectly to the test data. However, the increase in error is not drastic, suggesting that the model still maintains a reasonable balance between fitting the training data and generalizing to new data.

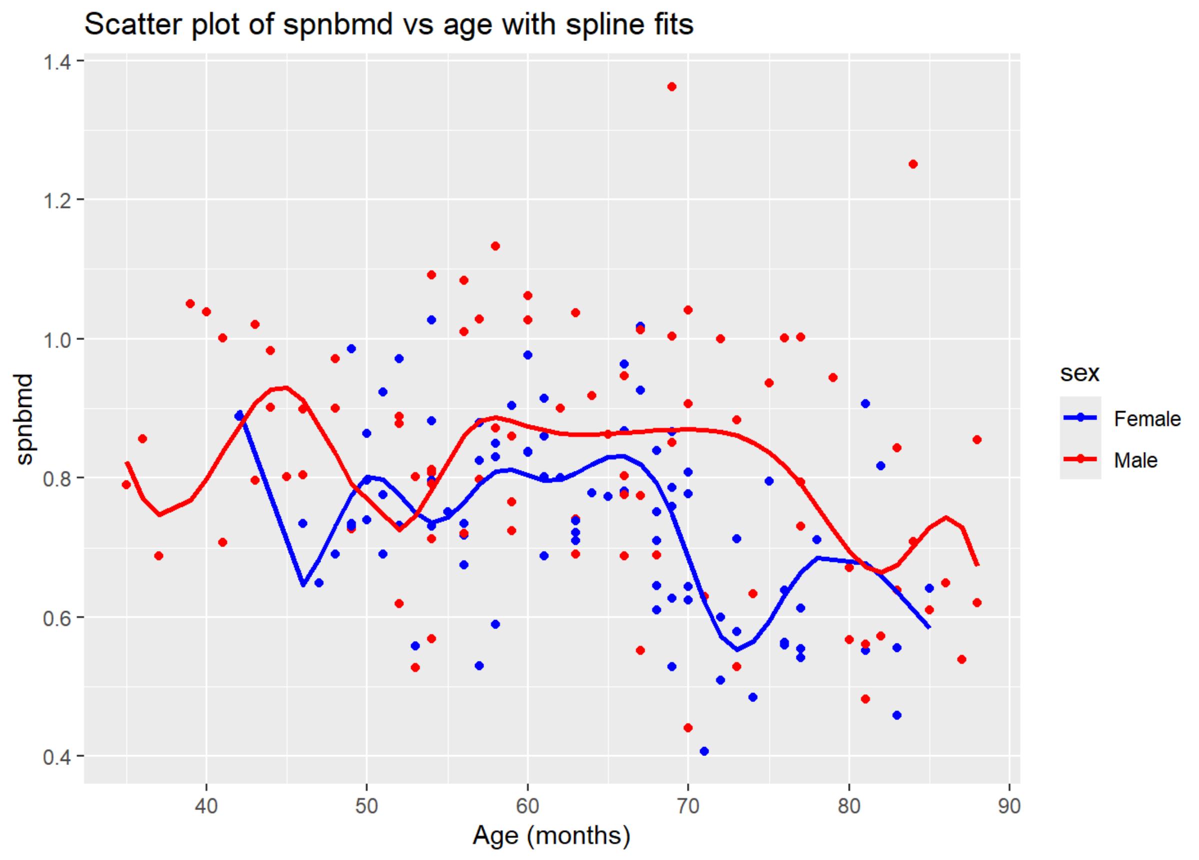
Interpret

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.

```r
# Fit final spline models using df_1se (11)
df_1se <- 11

# Fit spline model for males
model_male <- lm(spnbmd ~ splines::bs(age_month, df = df_1se, Boundary.knots = range(bmd_train_male$age_
month)), data = bmd_train_male)
model_female <- lm(spnbmd ~ splines::bs(age_month, df = df_1se, Boundary.knots = range(bmd_train_female
$age_month)), data = bmd_train_female)

# Predict on the full dataset
bmd$predicted_spnbmd <- ifelse(bmd$sex == "M",
                               predict(model_male, newdata = bmd),
                               predict(model_female, newdata = bmd))
```

```
## Warning in splines::bs(age_month, degree = 3L, knots = c(50, 54, 58,
## 61.3333333333333, : some 'x' values beyond boundary knots may cause
## ill-conditioned bases
```

```r
# Scatter plot with overlaid spline fits
ggplot(bmd, aes(x = age_month, y = spnbmd, color = sex)) +
  geom_point() +
  geom_line(aes(y = predicted_spnbmd), linewidth = 1) +
  labs(title = "Scatter plot of spnbmd vs age with spline fits", x = "Age (months)", y = "spnbmd") +
  scale_color_manual(values = c("blue", "red"), labels = c("Female", "Male"))
```



- 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?

Peaks of Growth Spurts and Levelling Off of Growth

Boys: The peak of the growth spurt for boys appears to be around 55-75 months. Growth largely levels off for boys starting around 75 months.

Girls: The peak of the growth spurt for girls appears to be around 45-70 months. Growth largely levels off for girls starting around 70 months.

Boys generally experience their growth spurts later than girls and continue growing for a longer period. Girls typically reach their peak growth spurt earlier and level off sooner compared to boys.