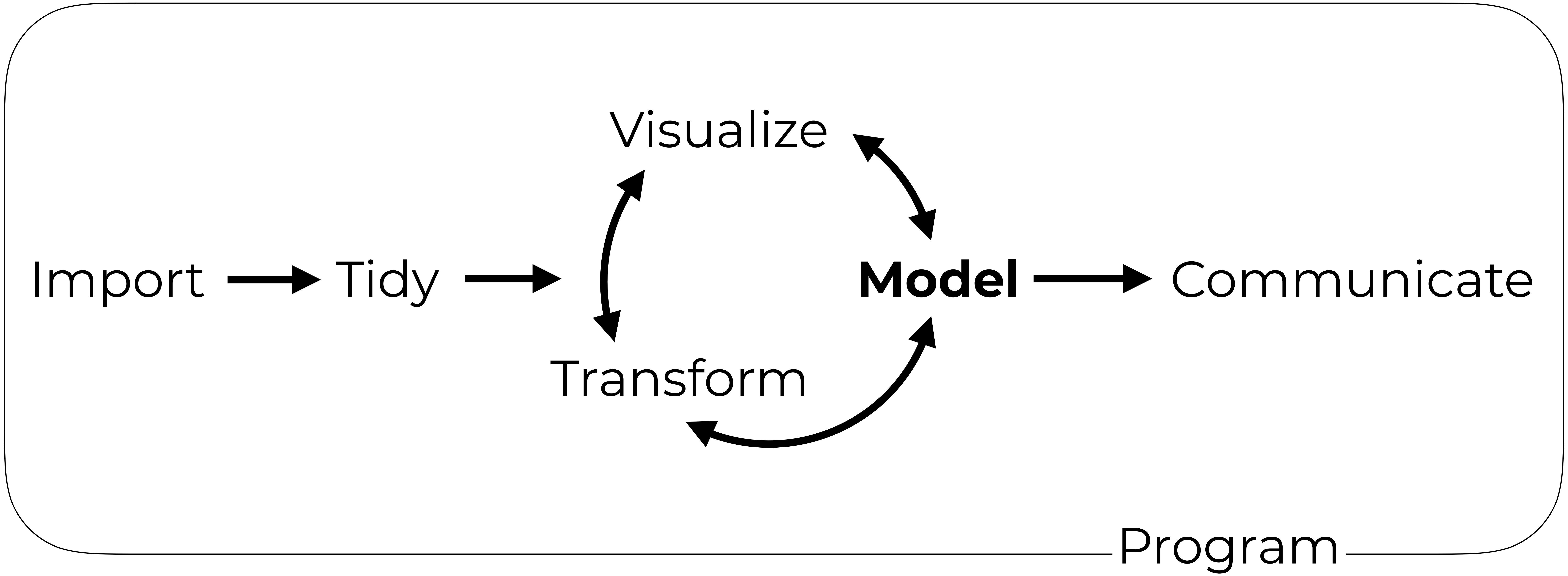
Model Data

Jake Thompson
wjakethompson.com
/ @wjakethompson

ATLAS
Accessible Teaching, Learning, and Assessment Systems

Your Turn 0

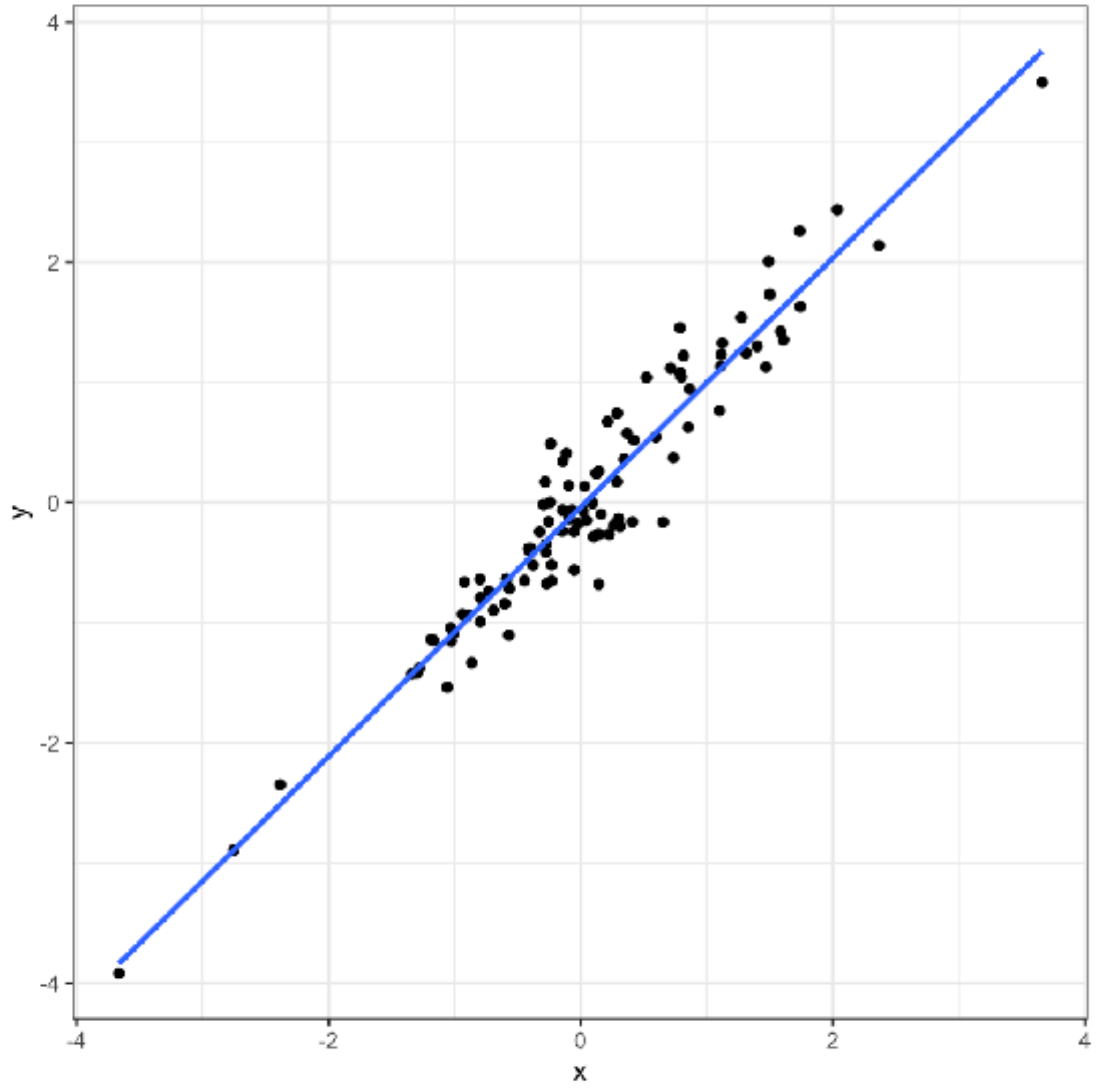
- Open **07-Model.Rmd**
- Run the setup chunk

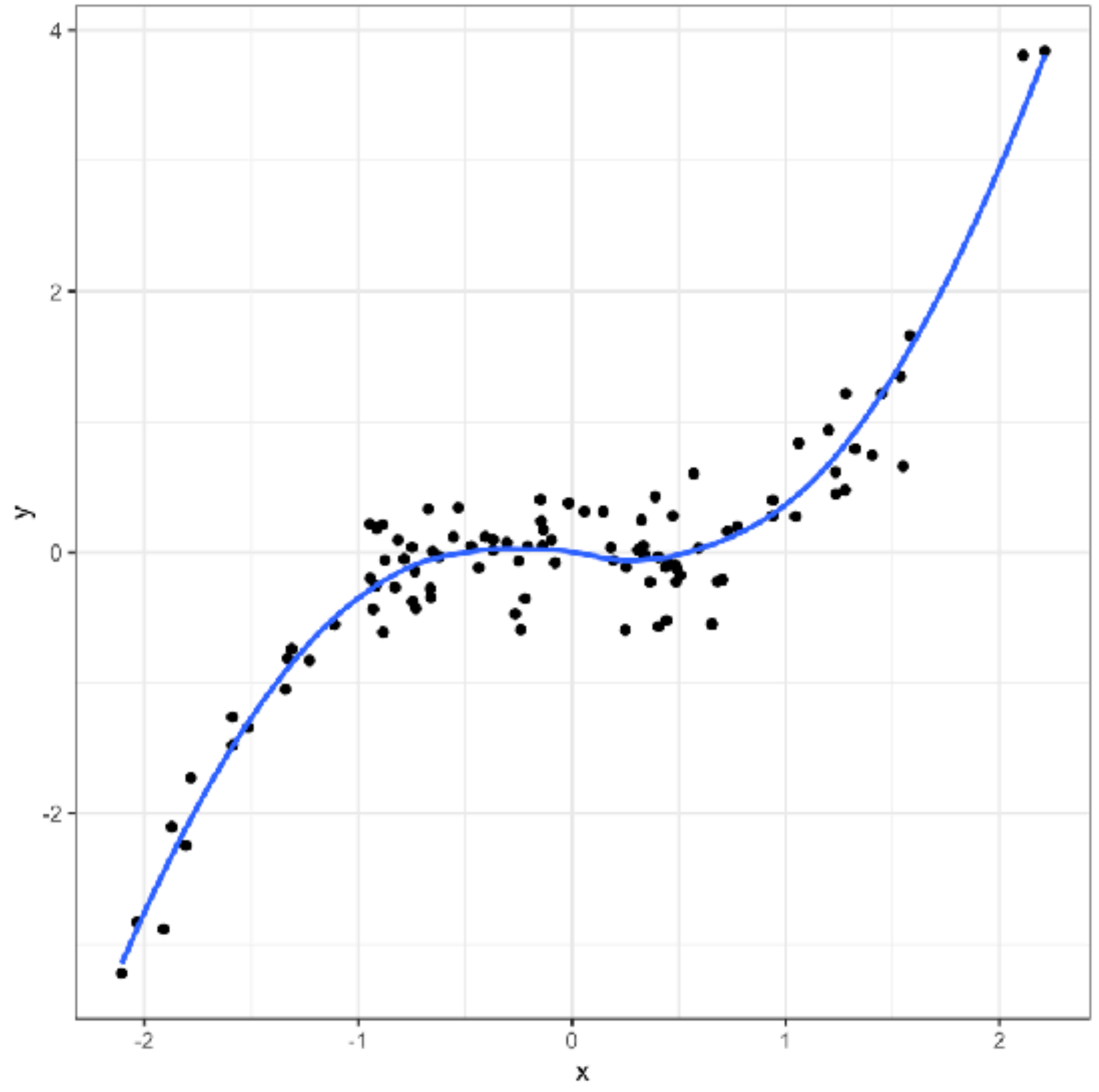

Visualize
Import
Tidy
Model
Communicate
Transform
Program

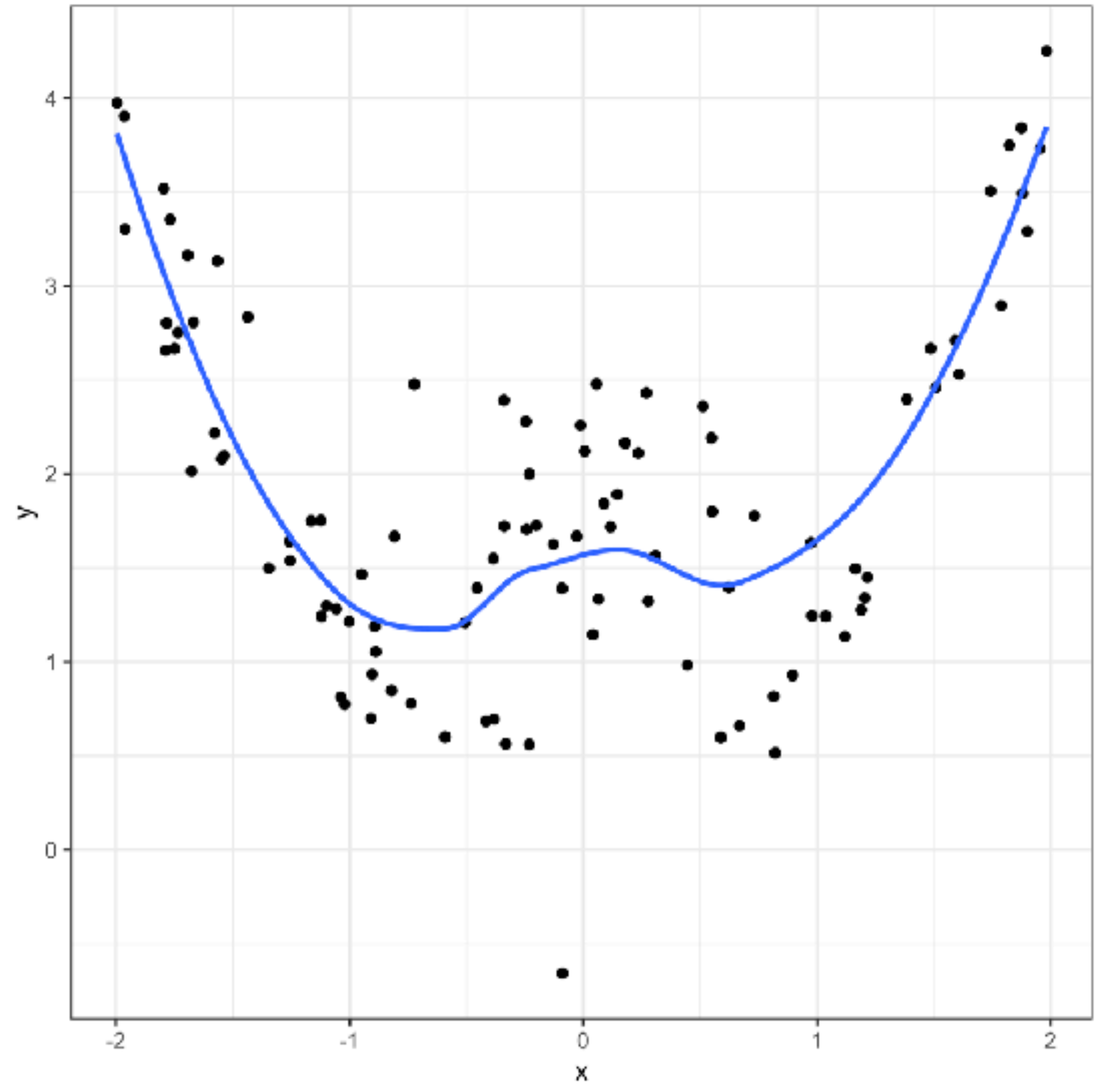
The basics

Models

A low dimensional description of a higher dimensional data set.

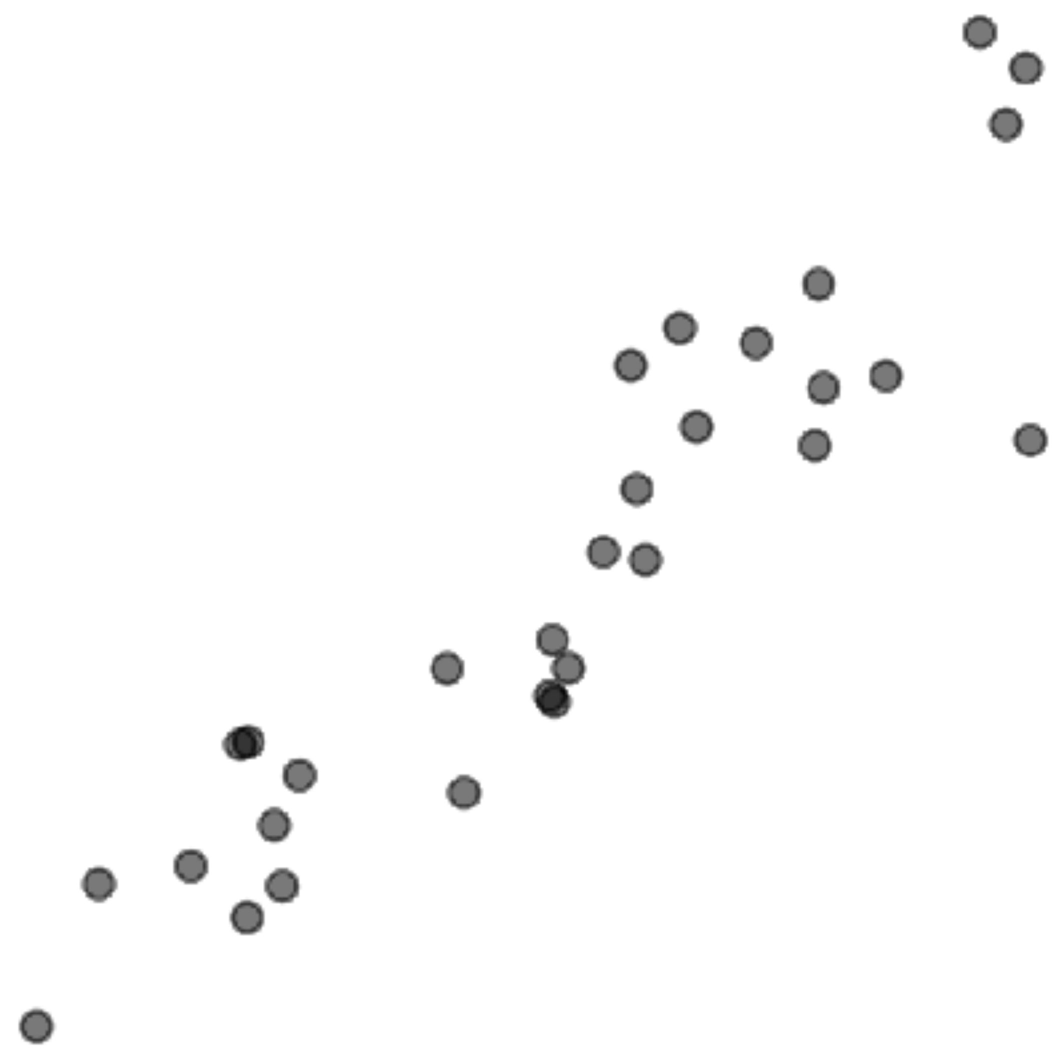






ATLAS

Models



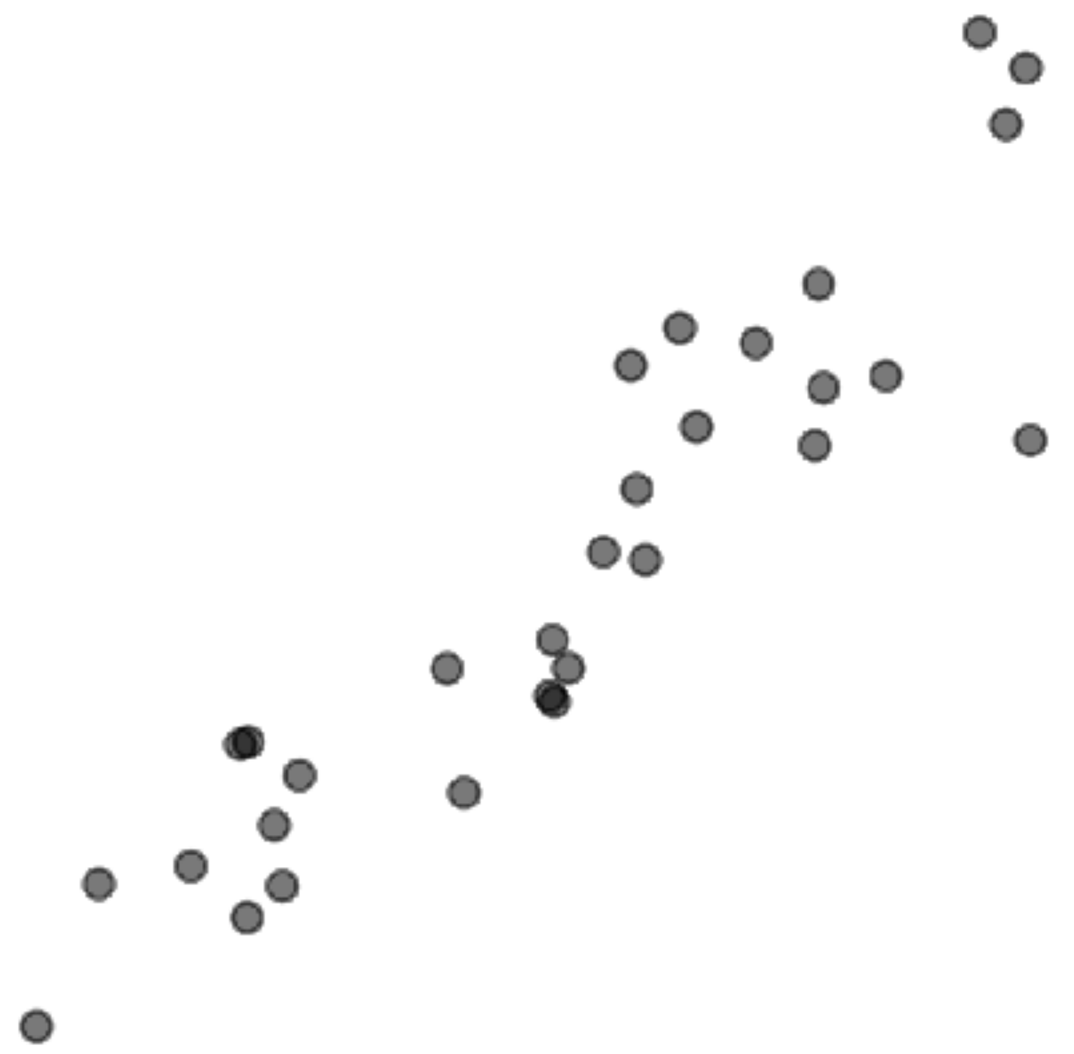
Algorithm

Data

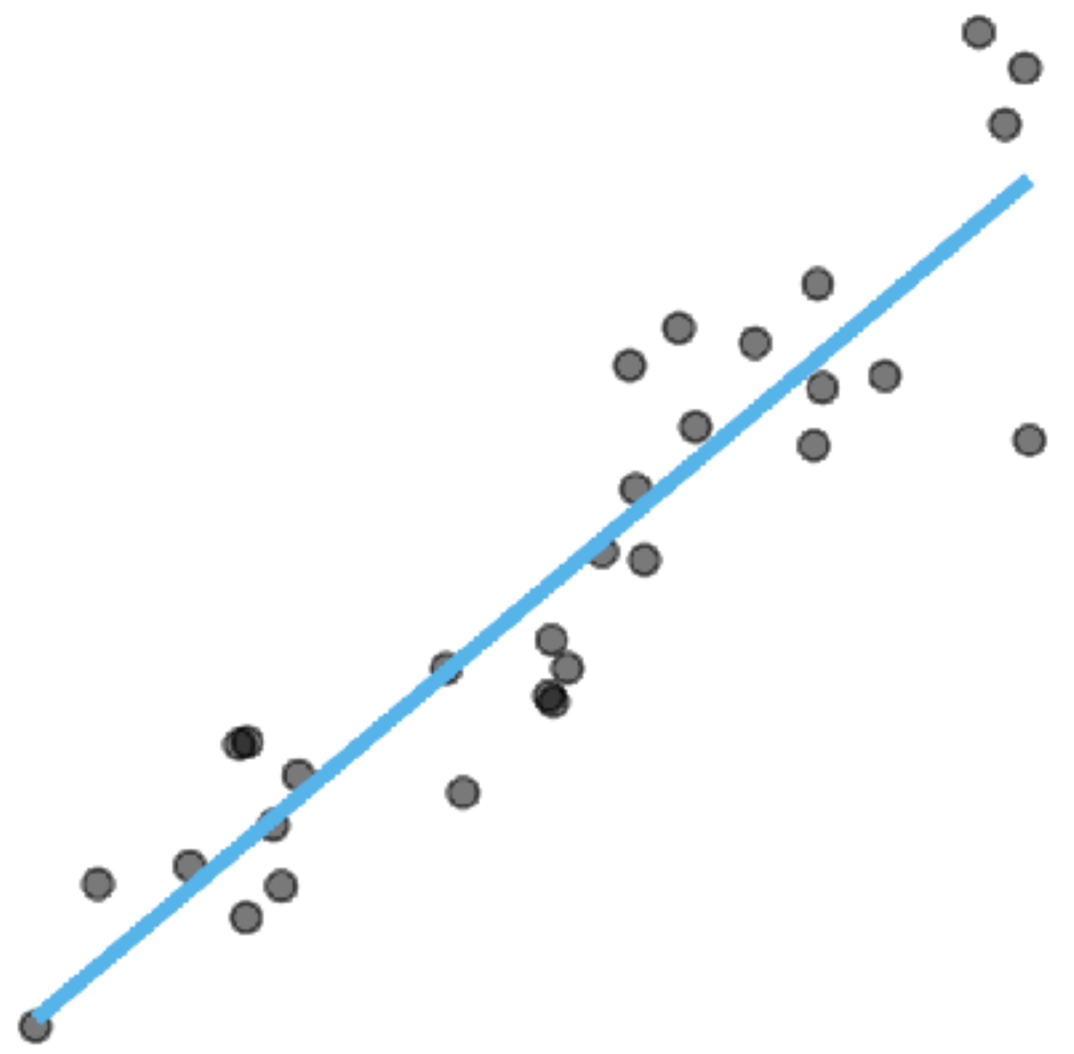
Model Function

Models

What is the **model function**?



Algorithm



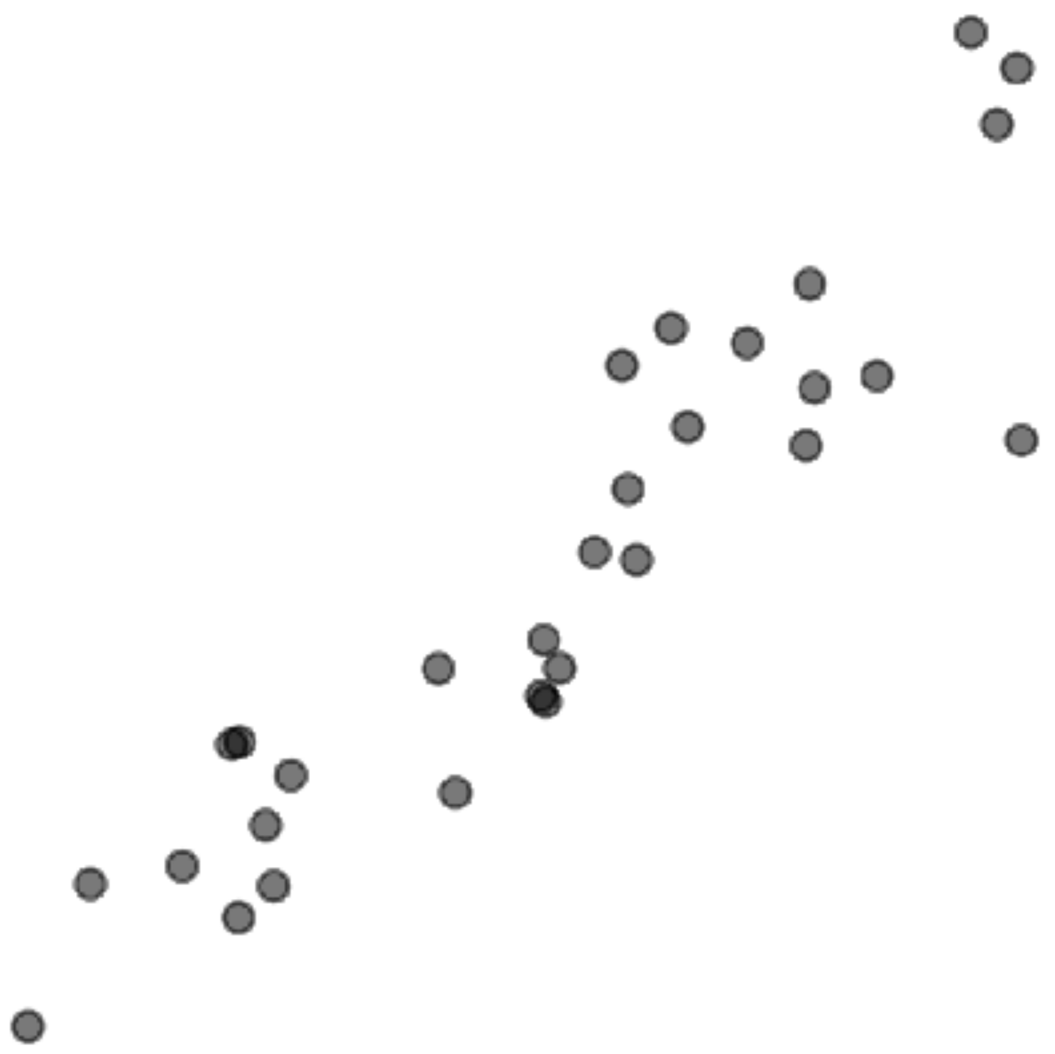
Data

Model Function

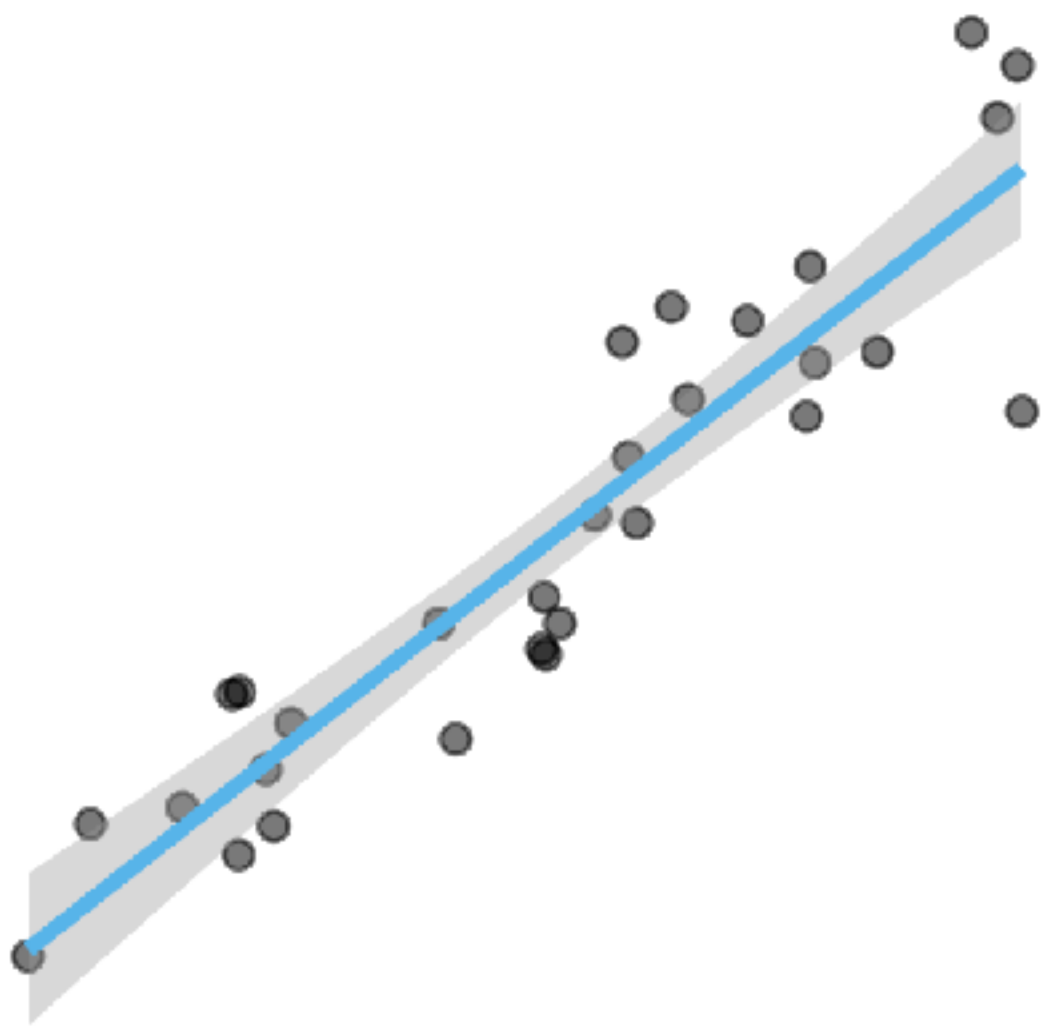
ATLAS

Models

What **uncertainty** is associated with it?



Algorithm

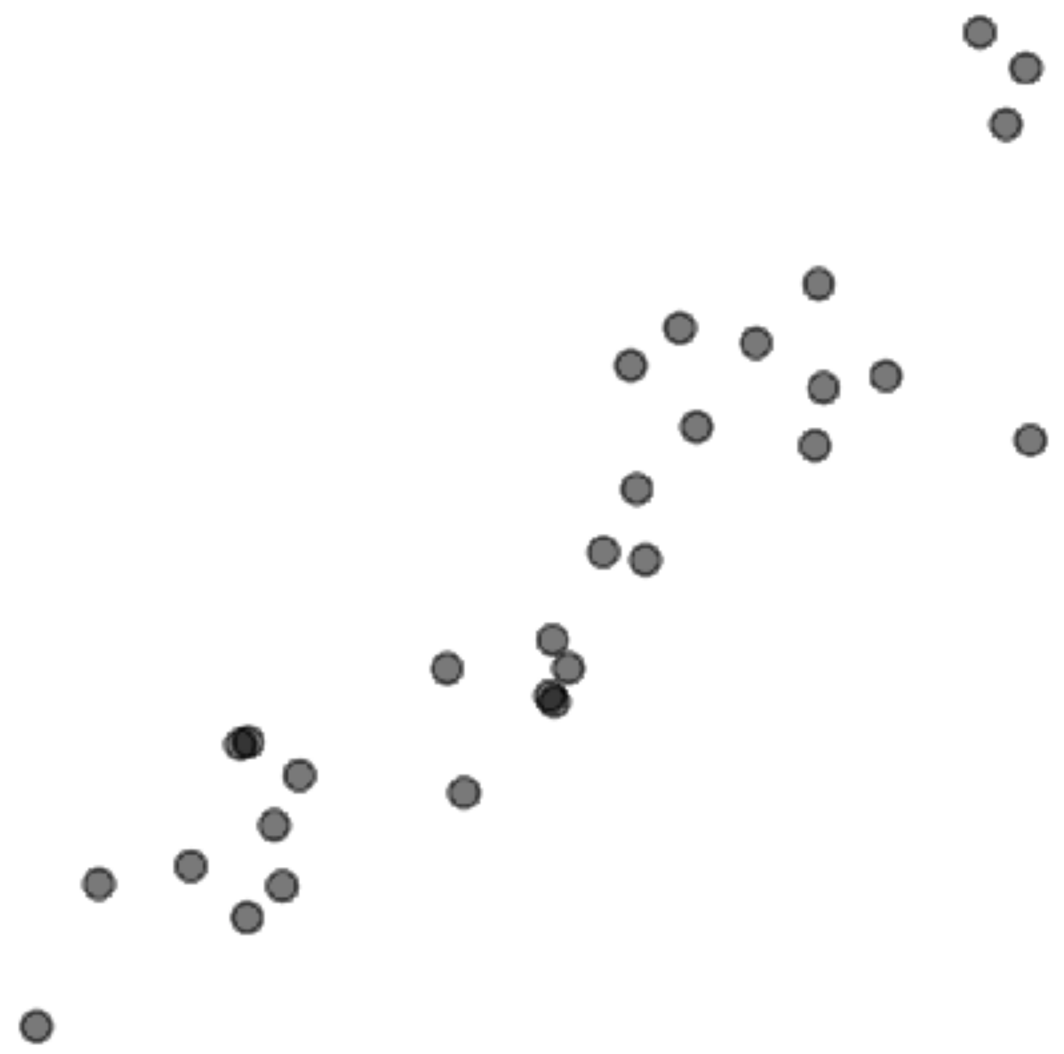


Data

Model Function

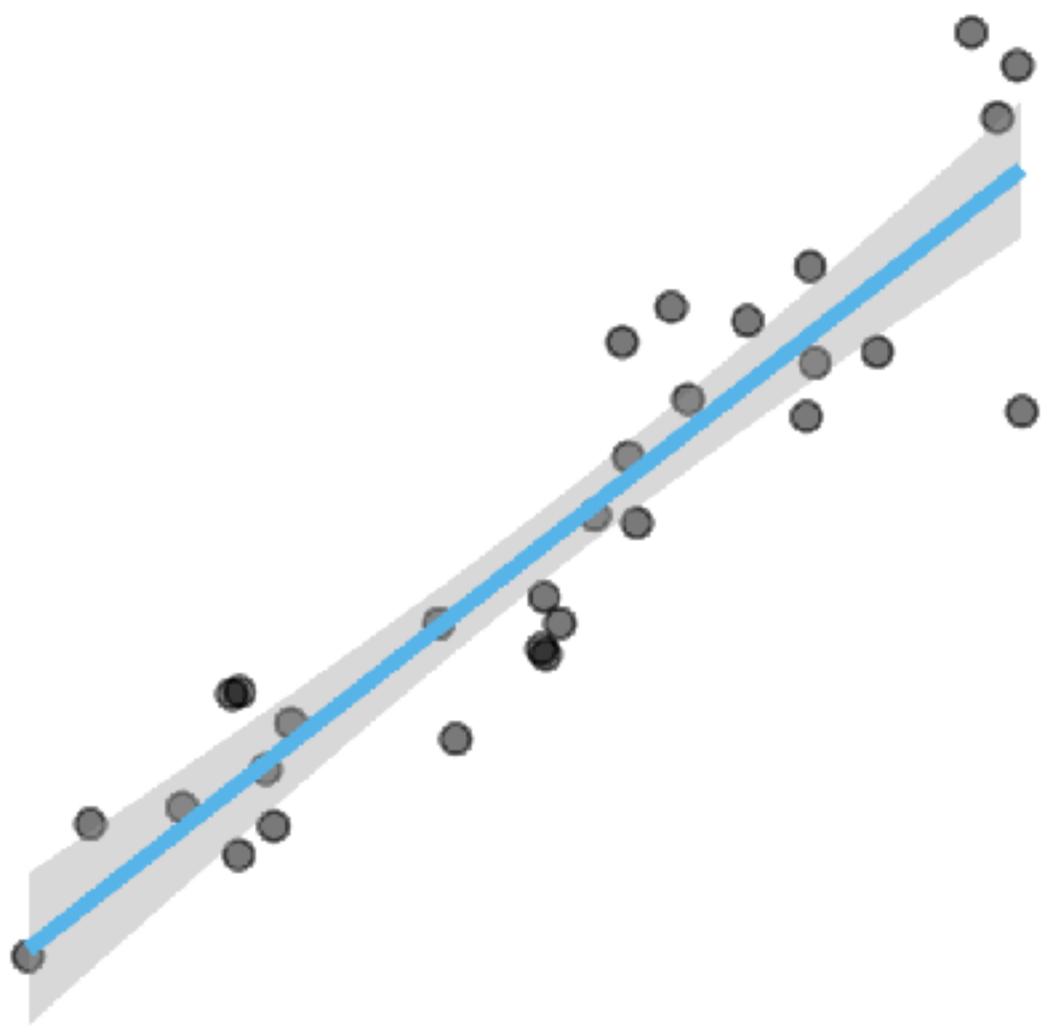
Models

How "good" is the model?



Data

Algorithm

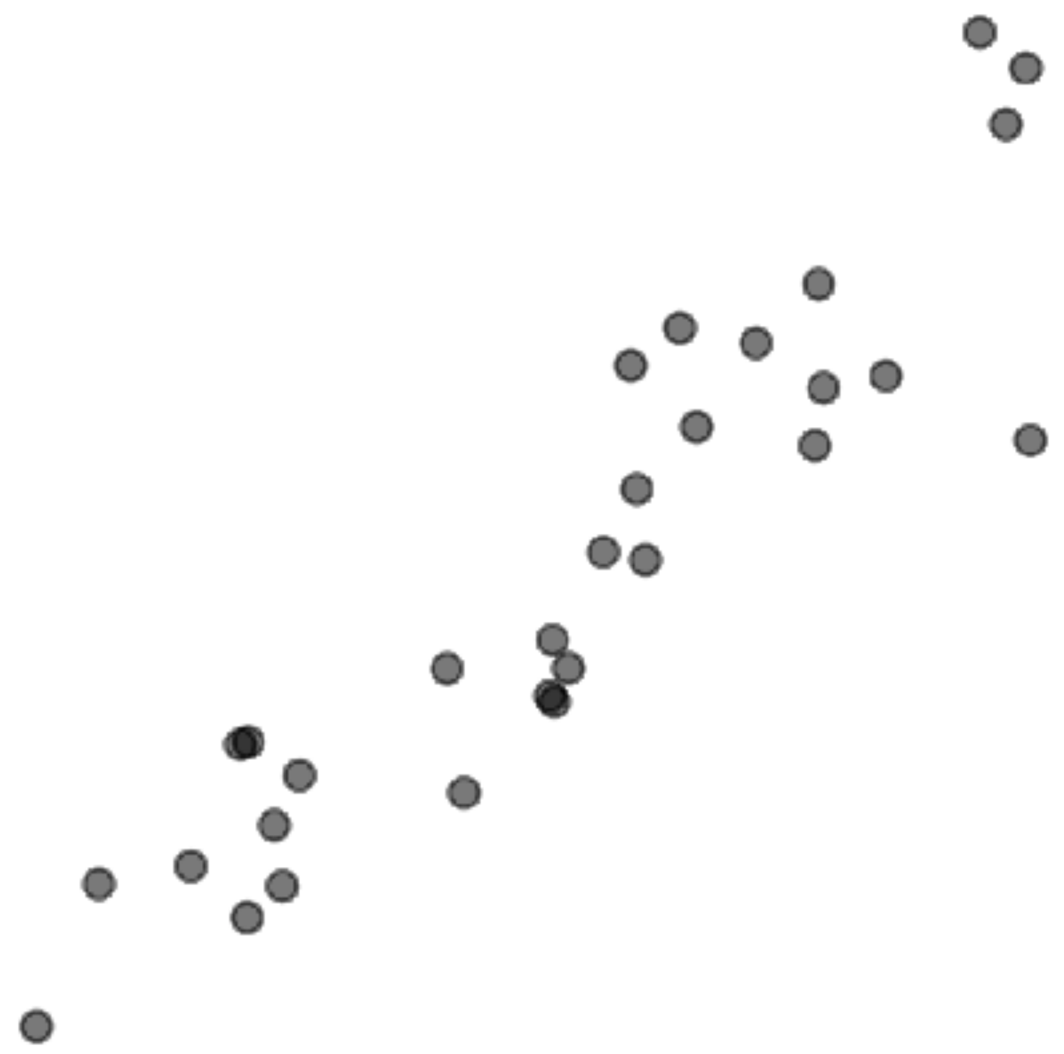


Model Function

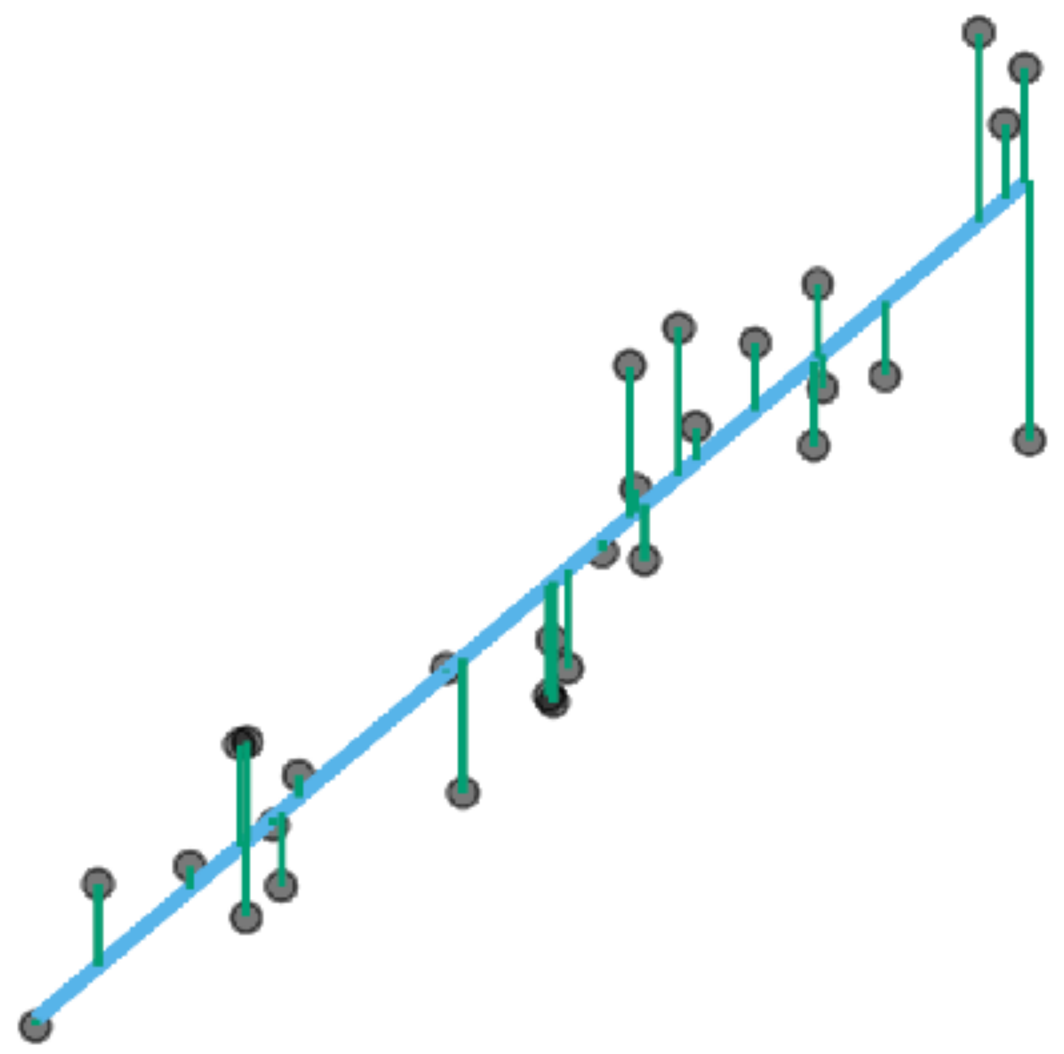
ATLAS

Models

What are the **residuals**?



Algorithm



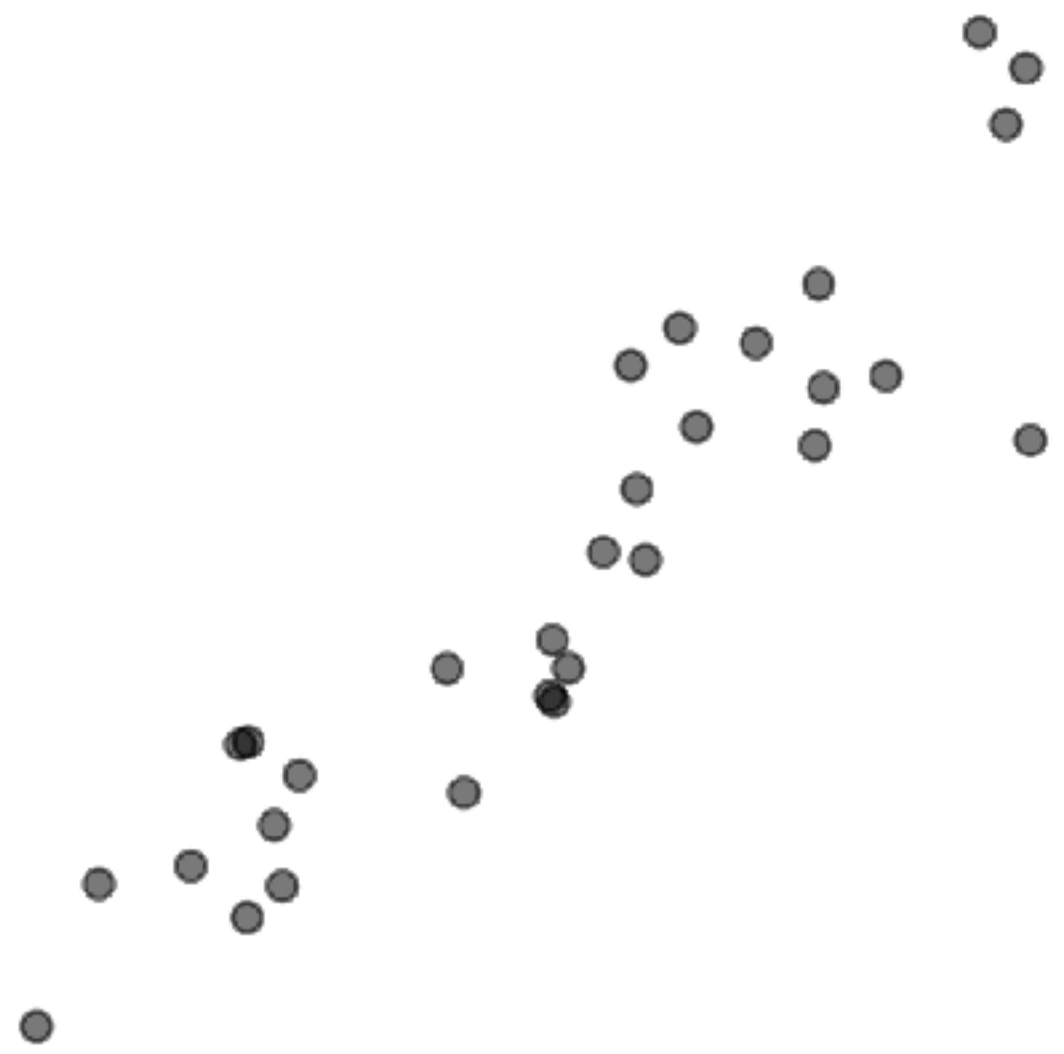
Data

Model Function

ATLAS

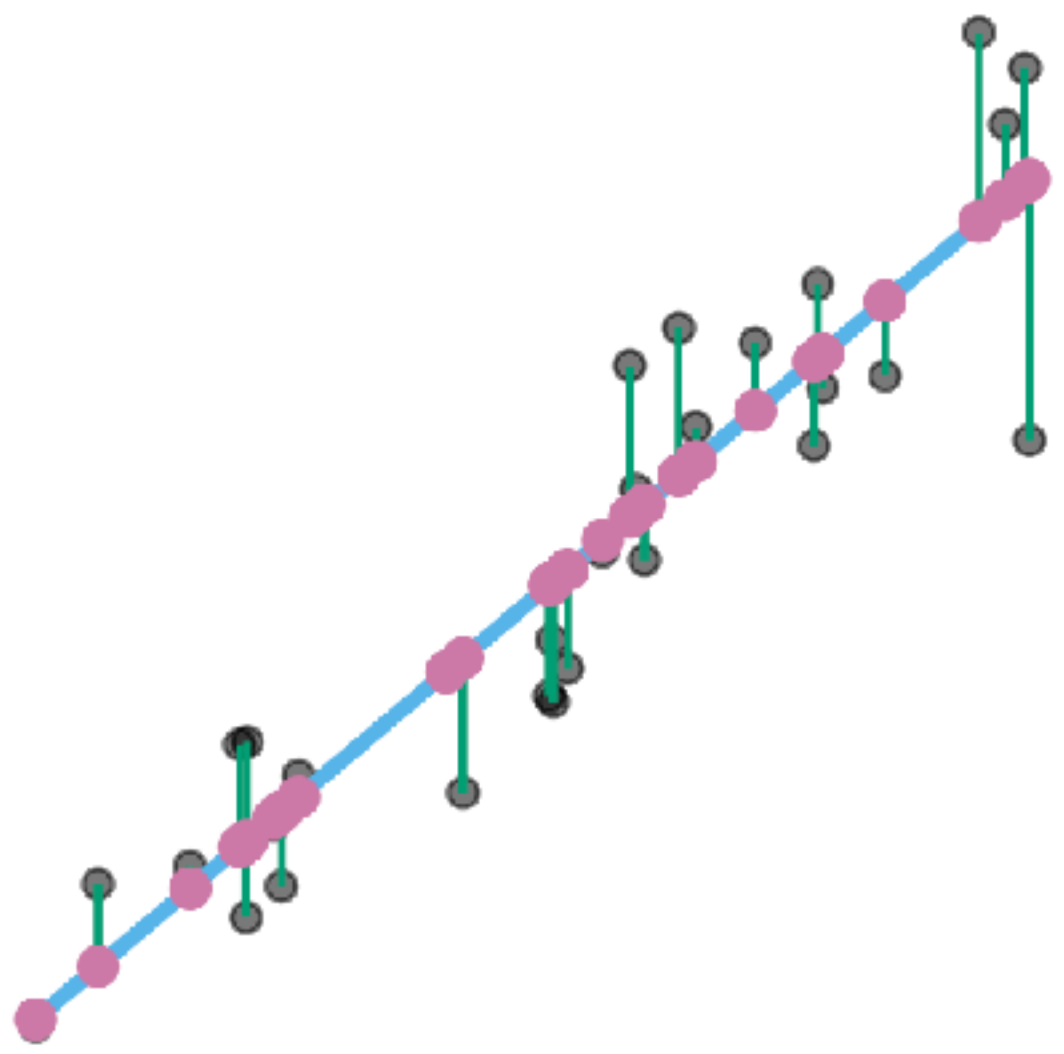
Models

What are the **predictions**?



Data

Algorithm



Model Function

Algorithm

(Popular) **model functions in R**

function	package	fits
`lm()`	stats	linear models
`glm()`	stats	generalized linear models
`lmer()`	lme4	multi-level models
`stan_glm()`	rstanarm	Bayesian regression models
`gam()`	mgcv	generalized additive models
`glmnet()`	glmnet	penalized linear models
`rlm()`	MASS	robust linear models
`rpart()`	rpart	trees
`randomForest()`	randomForest	random forrests
`xgboost()`	xgboost	gradient boosting machines

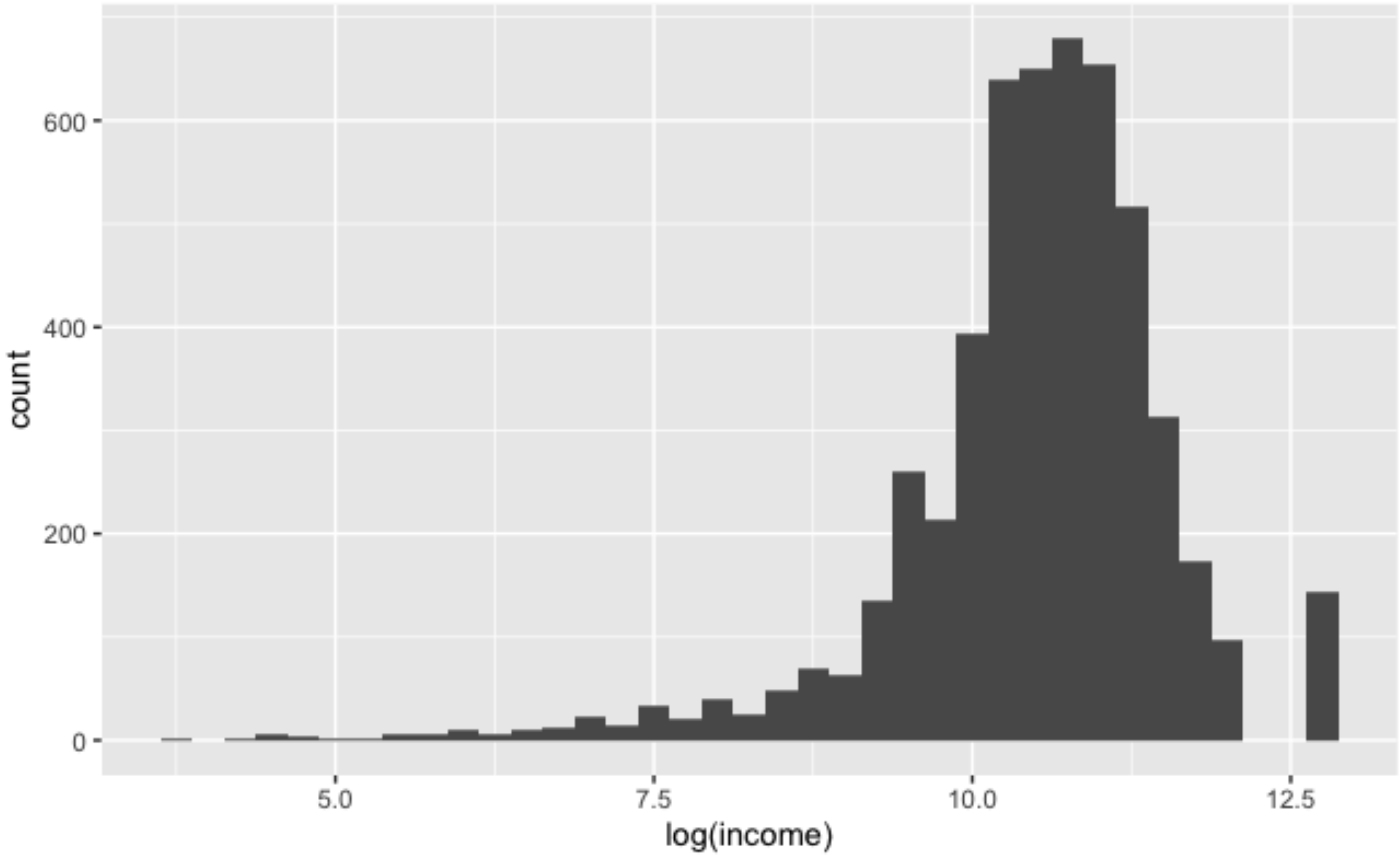
ATLAS

(Popular) **model functions in R**

function	package	fits
`lm()`	**stats**	**linear models**
`glm()`	stats	generalized linear models
`lmer()`	lme4	multi-level models
`stan_glm()`	rstanarm	Bayesian regression models
`gam()`	mgcv	generalized additive models
`glmnet()`	glmnet	penalized linear models
`rlm()`	MASS	robust linear models
`rpart()`	rpart	trees
`randomForest()`	randomForest	random forrests
`xgboost()`	xgboost	gradient boosting machines

```
wages
# A tibble: 5,266 x 8
   income height weight   age marital  sex    education  afqt
    <int>  <dbl>  <int> <int> <fct>    <fct>      <int> <dbl>
 1  19000     60    155    53 married  female        13  6.84
 2  35000     70    156    51 married  female        10 49.4
 3 105000     65    195    52 married  male          16 99.4
 4  40000     63    197    54 married  female        14 44.0
 5  75000     66    190    49 married  male          14 59.7
 6 102000     68    200    49 divorced female        18 98.8
 7  70000     64    160    54 divorced female        12 50.3
 8  60000     69    162    55 divorced male          12 89.7
 9 150000     69    194    54 divorced male          13 96.0
10 115000     64    145    53 married  female        16 67.0
# … with 5,256 more rows
```

```
wages %>%
  ggplot(mapping = aes(x = log(income))) +
    geom_histogram(binwidth = 0.25)
```



lm()

lm()

Fit a linear modle to data

```
lm(log(income) ~ education, data = wages)
```

A formula that describes the model equation

The data set

formulas

Formula only needs to include the response and predictors

$$y = \beta_0 + \beta_1 x + \epsilon$$

```
y ~ x
```

Your Turn 1

Fit the model below and then examine the output. What does it look like?

```
mod_e <- lm(log(income) ~ education, data = wages)
```

02:00

ATLAS

```r
mod_e <- lm(log(income) ~ education, data = wages)
mod_e
# Call:
# lm(formula = log(income) ~ education, data = wages)
#
# Coefficients:
# (Intercept)    education
#      8.5577       0.1418

class(mod_e)
# [1] "lm"
```

1. Not pipe friendly to have data as second argument 😢

2. Output is not tidy, or even a data frame.

ATLAS

```
summary(mod_e)
# Call:
# lm(formula = log(income) ~ education, data = wages)
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -6.7893 -0.3563  0.1328  0.5798  2.9136
#
# Coefficients:
#             Estimate Std. Error t value Pr(>|t|)
# (Intercept) 8.557691   0.073260  116.81   <2e-16 ***
# education   0.141840   0.005305   26.74   <2e-16 ***
# ---
# Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
#
# Residual standard error: 0.9923 on 5262 degrees of freedom
#   (2 observations deleted due to missingness)
# Multiple R-squared:  0.1196, Adjusted R-squared:  0.1195
# F-statistic:   715 on 1 and 5262 DF,  p-value: < 2.2e-16
```

Still not a data frame, but more information

Use "." to pipe input to somewhere other than the first argument

```
mod_e <- wages %>%
  lm(log(income) ~ education, data = .)
```

wages will be passed to here

broom

www.tidyverse.org

In R4DS

Many Models

broom

broom includes three functions which work for most types of models (and can be extended to more):

1. **tidy()** - returns model coefficients, stats
2. **glance()** - returns model diagnostics
3. **augment()** - returns predictions, residuals, and other raw values

tidy()

Returns useful **model output** as a data frame

```
mod_e %>% tidy()
# A tibble: 2 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)    8.56    0.0733      117.  0.
2 education      0.142   0.00530      26.7 8.41e-148
```

ATLAS

broom

glance()

Returns common **model diagnostics** as a data frame

```
mod_e %>% glance()
# A tibble: 1 x 11
  r.squared adj.r.squared sigma statistic   p.value    df logLik    AIC    BIC deviance df.residual
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <int>  <dbl>  <dbl>  <dbl>    <dbl>       <int>
1     0.120         0.119 0.992      715. 8.41e-148     2 -7428. 14862. 14881.    5182.        5262
```

augment()

Returns data frame of **model output related to original data points**

```
mod_e %>% augment()
# A tibble: 5,264 x 10
   .rownames log.income. education .fitted .se.fit  .resid      .hat .sigma      .cooksd .std.resid
   <chr>           <dbl>     <int>   <dbl>   <dbl>   <dbl>     <dbl>  <dbl>        <dbl>      <dbl>
 1 1                9.85        13   10.4   0.0140 -0.549  0.000199  0.992 0.0000305       -0.554
 2 2               10.5         10    9.98  0.0234  0.487  0.000554  0.992 0.0000668        0.491
 3 3               11.6         16   10.8   0.0188  0.735  0.000359  0.992 0.0000984        0.740
 4 4               10.6         14   10.5   0.0139  0.0532 0.000195  0.992 0.000000281      0.0536
 5 5               11.2         14   10.5   0.0139  0.682  0.000195  0.992 0.0000461        0.687
 6 6               11.5         18   11.1   0.0272  0.422  0.000751  0.992 0.0000680        0.425
 7 7               11.2         12   10.3   0.0160  0.896  0.000260  0.992 0.000106         0.904
 8 8               11.0         12   10.3   0.0160  0.742  0.000260  0.992 0.0000728        0.748
 9 9               11.9         13   10.4   0.0140  1.52   0.000199  0.992 0.000233         1.53
10 10              11.7         16   10.8   0.0188  0.826  0.000359  0.992 0.000124         0.832
# … with 5,254 more rows
```

ATLAS

broom

augment()

Returns data frame of **model output related to original data points**

```
mod_e %>% augment(data = wages)
```

Adds the original wages data set to the output

ATLAS

broom

Your Turn 2

Use a pipe to model **log(income)** against **height**. Then use broom and dplyr functions to extract:

1. The coefficient estimates and their related statistics
2. The **adj.r.squared** and **p.value** for the overall model

05:00

```r
mod_h <- wages %>% lm(log(income) ~ height, data = .)
mod_h %>%
  tidy()
# A tibble: 2 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)   6.98     0.237        29.4 4.13e-176
2 height        0.0520   0.00352      14.8 2.44e- 48

mod_h %>%
  glance() %>%
  select(adj.r.squared, p.value)
# A tibble: 1 x 2
  adj.r.squared  p.value
          <dbl>    <dbl>
1        0.0396 2.44e-48
```

```
mod_h %>%
  tidy() %>% filter(p.value < 0.05)
# A tibble: 2 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)   6.98     0.237        29.4 4.13e-176
2 height        0.0520   0.00352      14.8 2.44e- 48

mod_e %>%
  tidy() %>% filter(p.value < 0.05)
# A tibble: 2 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)    8.56    0.0733      117.  0.
2 education      0.142   0.00530      26.7 8.41e-148
```

So which determines income?

Multiple Regression

To fit multiple predictors, add multiple variables to the formula

```
lm(log(income) ~ education + height, data = wages)
```

Your Turn 3

Model **log(income)** against **education** *and* **height**. Do the coefficients change?

03 : 00

ATLAS

broom

```
mod_eh <- wages %>%
  lm(log(income) ~ education + height, data = .)

mod_eh %>%
  tidy()
# A tibble: 3 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)   5.35     0.231        23.1 1.00e-112
2 education     0.139    0.00521      26.6 7.12e-147
3 height        0.0483   0.00331      14.6 2.50e- 47
```

Your Turn 4

Model **log(income)** against **education** and **height** and **sex**. Can you interpret the coefficients?

03 : 00

```
mod_ehs <- wages %>%
  lm(log(income) ~ education + height + sex, data = .)

mod_ehs %>%
  tidy()
# A tibble: 4 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)  8.25      0.335        24.6 4.68e-127
2 education    0.148     0.00520      28.5 5.16e-166
3 height       0.00673   0.00479       1.40 1.61e-  1
4 sexfemale   -0.462     0.0389      -11.9 5.02e- 32
```

What does this mean?

Where is **sexmale**?

ATLAS

broom

```
# A tibble: 4 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)  8.25      0.335        24.6  4.68e-127
2 education    0.148     0.00520      28.5  5.16e-166
3 height       0.00673   0.00479       1.40 1.61e-  1
4 sexfemale   -0.462     0.0389      -11.9  5.02e- 32
```

For factors, R treats the first level as the baseline level, e.g. the mean log(income) for a male is:

log(income) = 8.25 + 0.15 * education + 0 * height

Each additional level gets a coefficient that acts as an *adjustment* between the baseline level and the additional level, e.g. the mean income for a female is:

log(income) = 8.25 + 0.15 * education + 0 * height - 0.46

```
# A tibble: 4 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)  8.25      0.335        24.6 4.68e-127
2 education    0.148     0.00520      28.5 5.16e-166
3 height       0.00673   0.00479       1.40 1.61e-  1
4 sexfemale   -0.462     0.0389      -11.9  5.02e- 32
```

For factors, R treats the first level as the baseline level, e.g. the mean log(income) for a male is:

log(income) = 8.25 + 0.15 * education + 0 * height

Each additional level gets a coefficient that acts as an *adjustment* between the baseline level and the additional level, e.g. the mean income for a female is:

log(income) = 8.25 + 0.15 * education + 0 * height - 0.46

ATLAS

broom

```
# A tibble: 4 x 5
  term        estimate std.error statistic   p.value
  <chr>          <dbl>     <dbl>     <dbl>     <dbl>
1 (Intercept)  8.25      0.335        24.6 4.68e-127
2 education    0.148     0.00520      28.5 5.16e-166
3 height       0.00673   0.00479       1.40 1.61e-  1
4 sexfemale   -0.462     0.0389      -11.9 5.02e- 32
```

But what does all of this look like?

Model Visualization

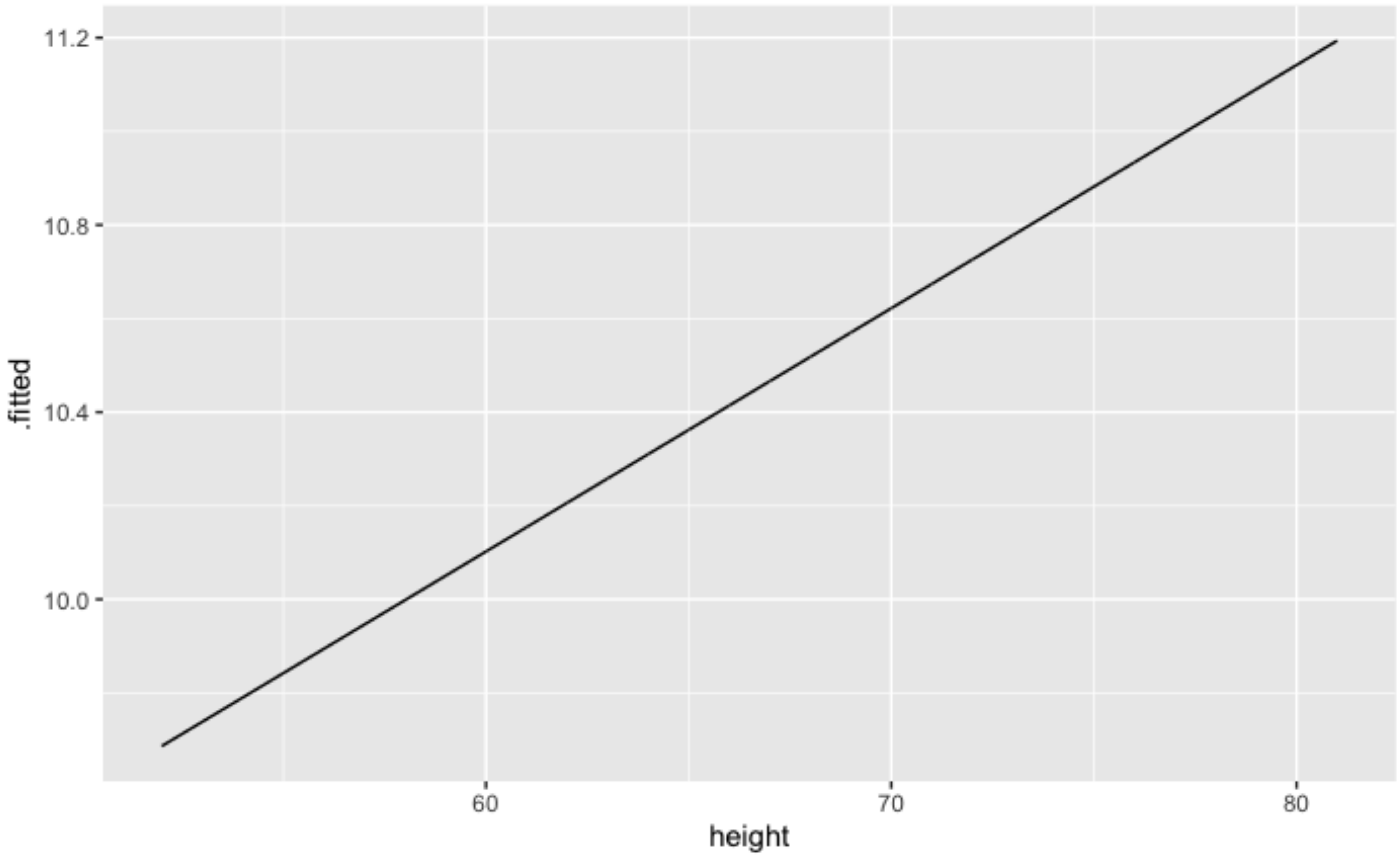
Your Turn 5

Use a broom function and ggplot2 to make a line graph of **height** vs. **.fitted** for our heights models, **mod_h**.

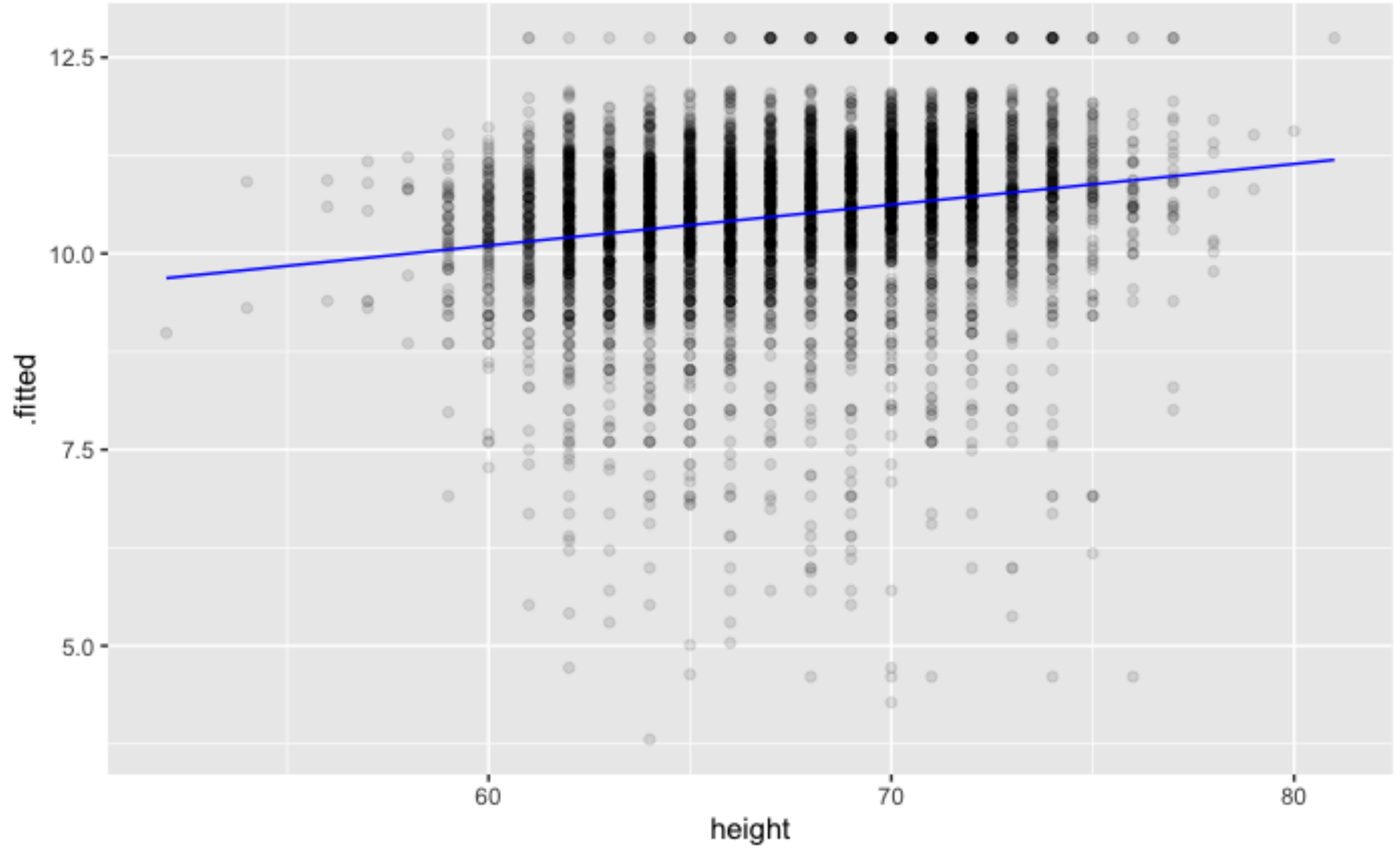
Bonus: Overlay the plat on the original data points.

03:00

```r
mod_h %>%
  augment(data = wages) %>%
  ggplot(mapping = aes(x = height, y = .fitted)) +
    geom_line()
```



```r
mod_h %>%
  augment(data = wages) %>%
  ggplot(mapping = aes(x = height, y = .fitted)) +
    geom_point(mapping = aes(y = log(income)), alpha = 0.1) +
    geom_line(color = "blue")
```



Your Turn 6

Repeat the process to make a line graph of **height** vs. **.fitted** colored by **sex** for model **mod_ehs**. Are the results interpretable?

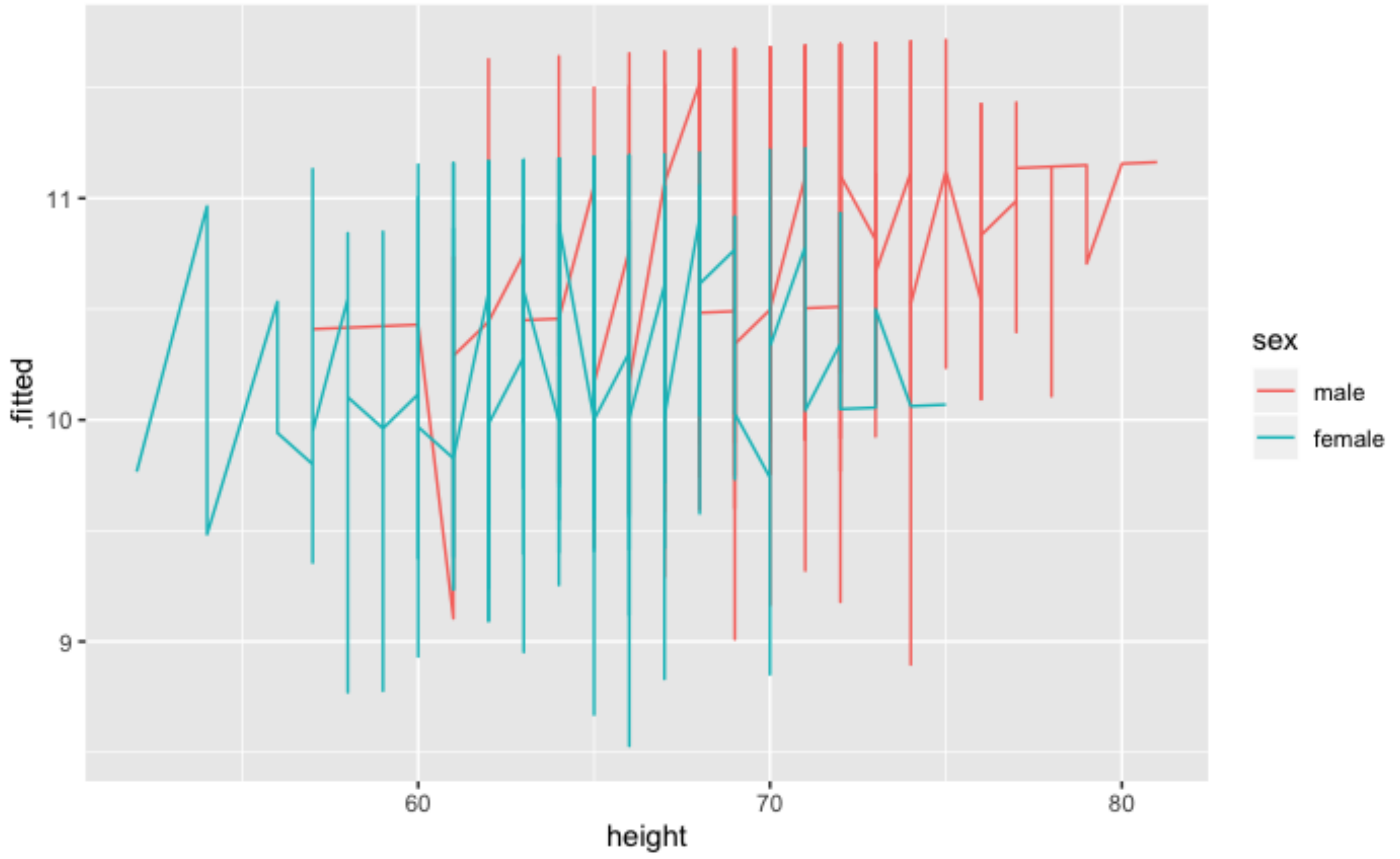
Add **+ facet_wrap(~education)** to the end of the your code. What happens?

02 : 00

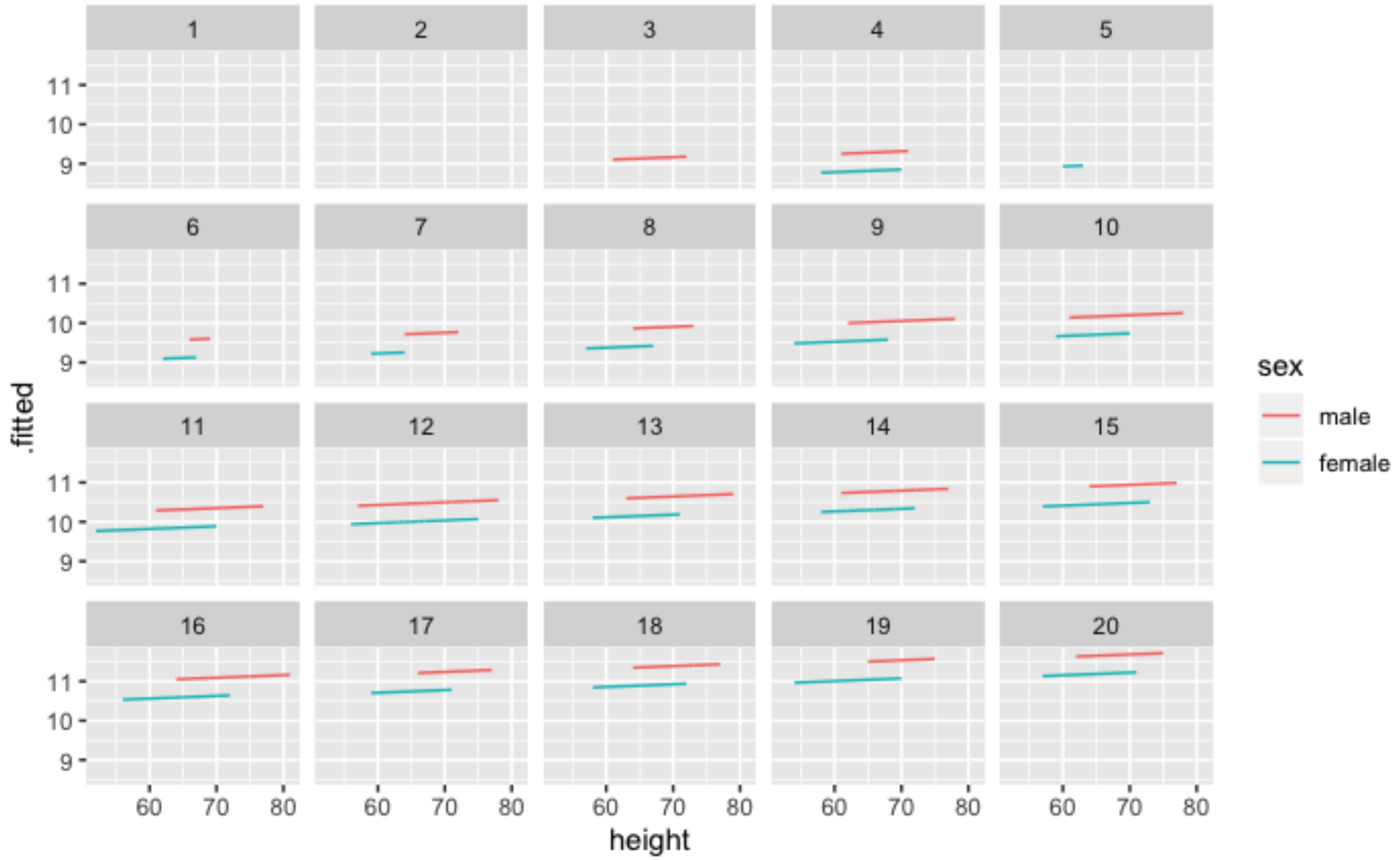
ATLAS

broom

```
mod_ehs %>%
  augment(data = wages) %>%
  ggplot(mapping = aes(x = height, y = .fitted, color = sex)) +
    geom_line()
```

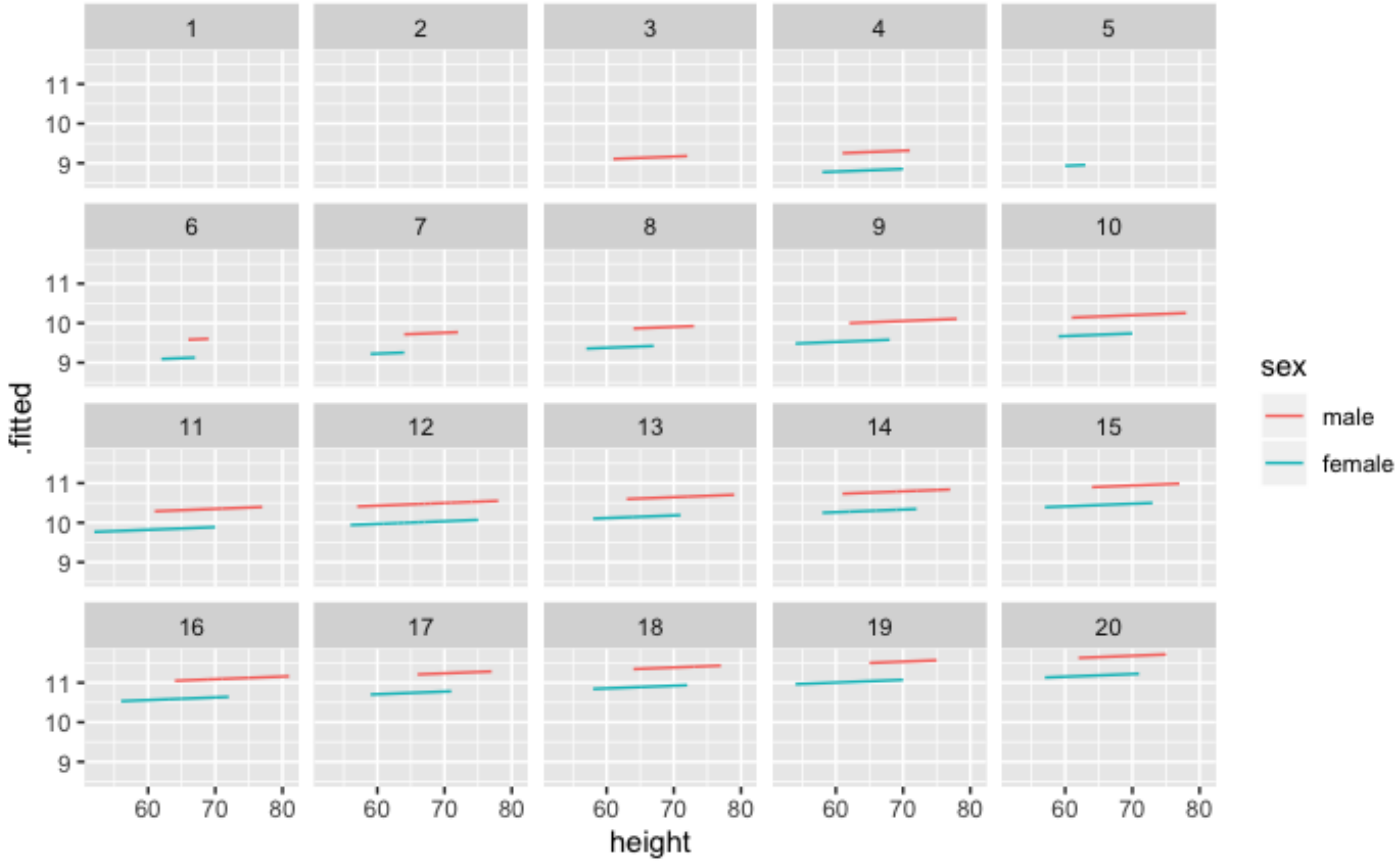


```
mod_ehs %>%
  augment(data = wages) %>%
  ggplot(mapping = aes(x = height, y = .fitted, color = sex)) +
    geom_line() +
    facet_wrap(vars(education))
```



ATLAS

broom



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
.fitted
height
sex
male
female

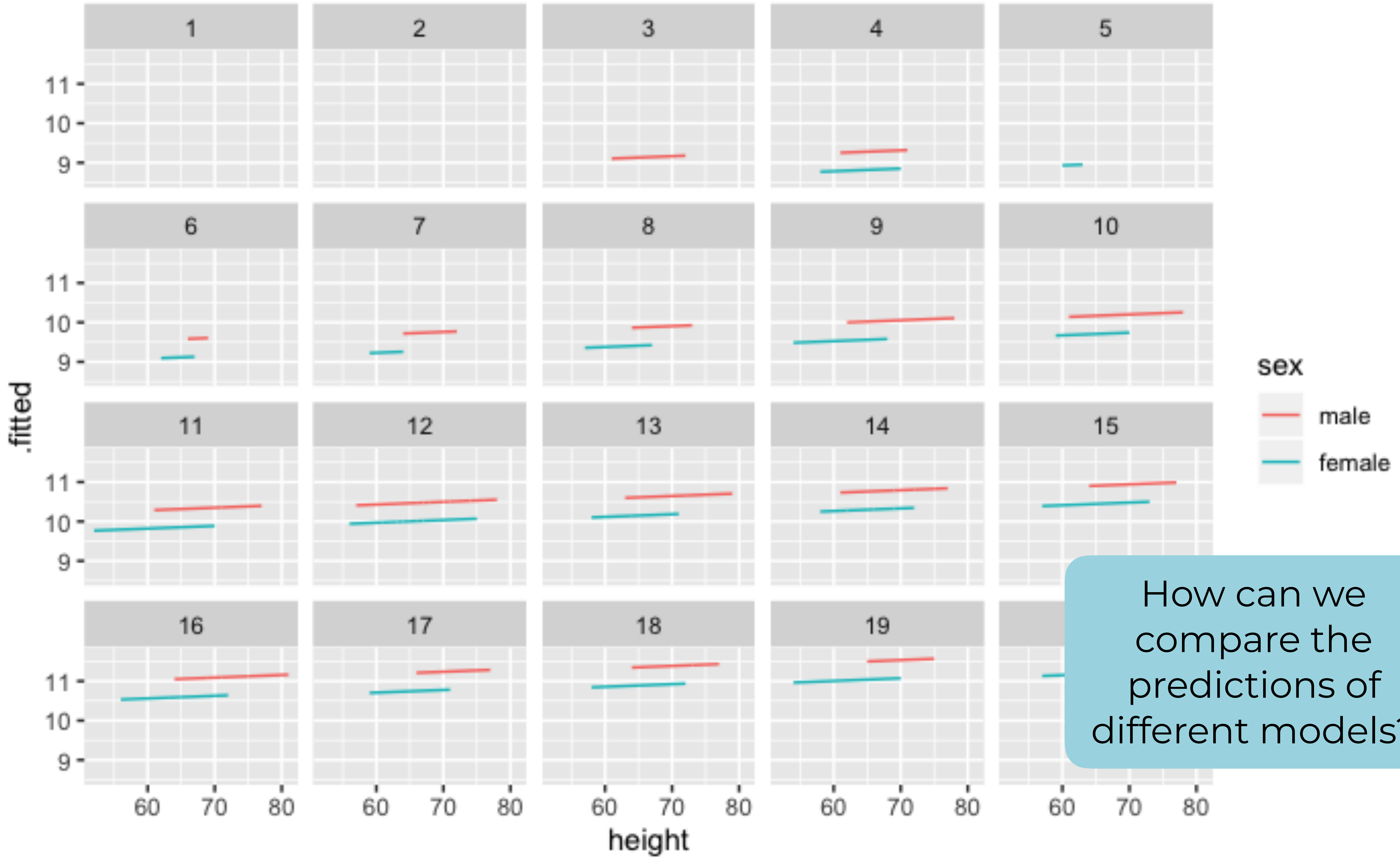
facet_wrap()

Divides plot into subplots based on a grouping variable.
"Wraps" subplots into rectangular collection.

```
+ facet_wrap(vars(var))
```

vars() - find a variable in the data frame

The grouping variable



How can we compare the predictions of different models?

Visualizing Multiple Models

modelr

www.rstudio.com

In R4DS

Model Basics

ATLAS

modelr

modelr includes three functions for working with model predictions:

1. **add_predictions()** - adds the model prediction for each case in the data set. Overlaps with broom::**augment()**

2. **spread_predictions()** - Adds predictions in separate columns

3. **gather_predictions()** - Adds predictions in a pair of key:value columns (model:pred)

ATLAS

modelr

add_predictions()

Uses the values in a data frame to generate a prediction for each case.

```
add_predictions(data, model)
```

Uses this model

To add predictions to these cases

ATLAS

```
wages %>% add_predictions(mod_h)
# A tibble: 5,266 x 9
   income height weight   age marital  sex     education  afqt  pred
    <int>  <dbl>  <int> <int> <fct>    <fct>       <int> <dbl> <dbl>
 1  19000     60    155    53 married  female         13  6.84  10.1
 2  35000     70    156    51 married  female         10 49.4   10.6
 3 105000     65    195    52 married  male           16 99.4   10.4
 4  40000     63    197    54 married  female         14 44.0   10.3
 5  75000     66    190    49 married  male           14 59.7   10.4
 6 102000     68    200    49 divorced female         18 98.8   10.5
 7  70000     64    160    54 divorced female         12 50.3   10.3
 8  60000     69    162    55 divorced male           12 89.7   10.6
 9 150000     69    194    54 divorced male           13 96.0   10.6
10 115000     64    145    53 married  female         16 67.0   10.3
# … with 5,256 more rows
```

spread_predictions()

Adds predictions for multiple models, each in their own column

```
spread_predictions(data, ...)
```

Add predictions from each of these models

To the cases in this data frame

```
wages %>% spread_predictions(mod_h, mod_eh, mod_ehs)
# A tibble: 5,266 x 11
   income height weight   age marital  sex     education  afqt mod_h mod_eh mod_ehs
    <int>  <dbl>  <int> <int> <fct>    <fct>       <int> <dbl> <dbl>  <dbl>   <dbl>
 1  19000     60    155    53 married  female         13  6.84  10.1   10.1   10.1
 2  35000     70    156    51 married  female         10 49.4   10.6   10.1    9.74
 3 105000     65    195    52 married  male           16 99.4   10.4   10.7   11.1
 4  40000     63    197    54 married  female         14 44.0   10.3   10.3   10.3
 5  75000     66    190    49 married  male           14 59.7   10.4   10.5   10.8
 6 102000     68    200    49 divorced female         18 98.8   10.5   11.1   10.9
 7  70000     64    160    54 divorced female         12 50.3   10.3   10.1    9.99
 8  60000     69    162    55 divorced male           12 89.7   10.6   10.3   10.5
 9 150000     69    194    54 divorced male           13 96.0   10.6   10.5   10.6
10 115000     64    145    53 married  female         16 67.0   10.3   10.7   10.6
# … with 5,256 more rows
```

gather_predictions()

Adds predictions for multiple models as a pair of model:pred columns

```
gather_predictions(data, ...)
```

Add predictions from each of these models

To the cases in this data frame (duplicating rows as necessary)

modelr

```
wages %>% gather_predictions(mod_h, mod_eh, mod_ehs)
# A tibble: 15,798 x 10
   model    income height weight   age marital sex     education  afqt  pred
   <chr>     <int>  <dbl>  <int> <int> <fct>   <fct>       <int> <dbl> <dbl>
 1 mod_h     19000     60    155    53 married female         13  6.84 10.1
 2 mod_eh    19000     60    155    53 married female         13  6.84 10.1
 3 mod_ehs   19000     60    155    53 married female         13  6.84 10.1
 4 mod_h     35000     70    156    51 married female         10 49.4  10.6
 5 mod_eh    35000     70    156    51 married female         10 49.4  10.1
 6 mod_ehs   35000     70    156    51 married female         10 49.4   9.74
 7 mod_h    105000     65    195    52 married male           16 99.4  10.4
 8 mod_eh   105000     65    195    52 married male           16 99.4  10.7
 9 mod_ehs 105000     65    195    52 married male           16 99.4  11.1
10 mod_h     40000     63    197    54 married female         14 44.0  10.3
# … with 15,788 more rows
```

Your Turn 7

Use one of **spread_predictions()** or **gather_predictions()** to make a line graph of **height** vs. **pred** colored by **model** for each of **mod_h**, **mod_eh**, and **mod_ehs**. Are the results interpretable?

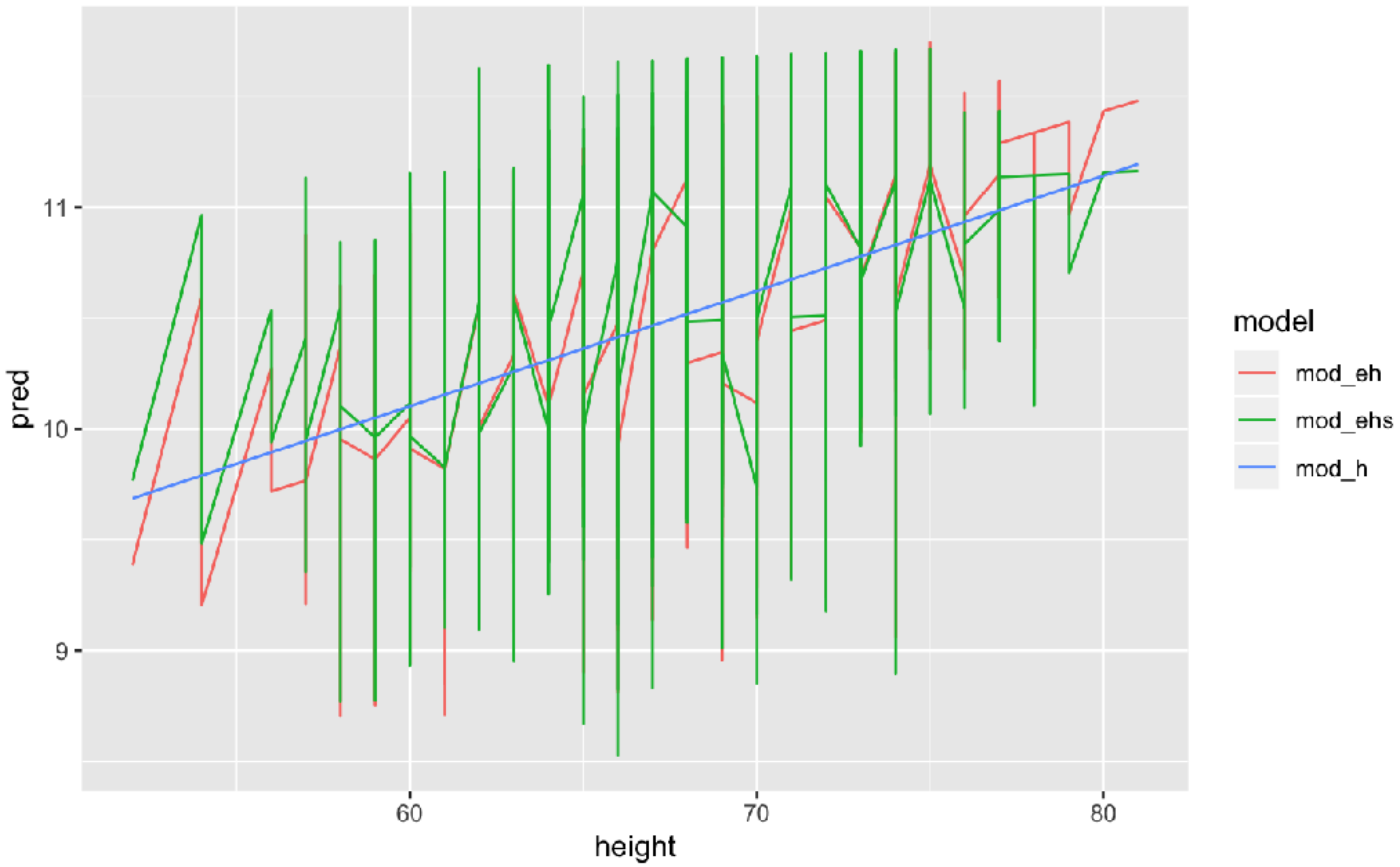
Add **+ facet_grid(sex ~ education)** to the end of your code. What happens?

04 : 00

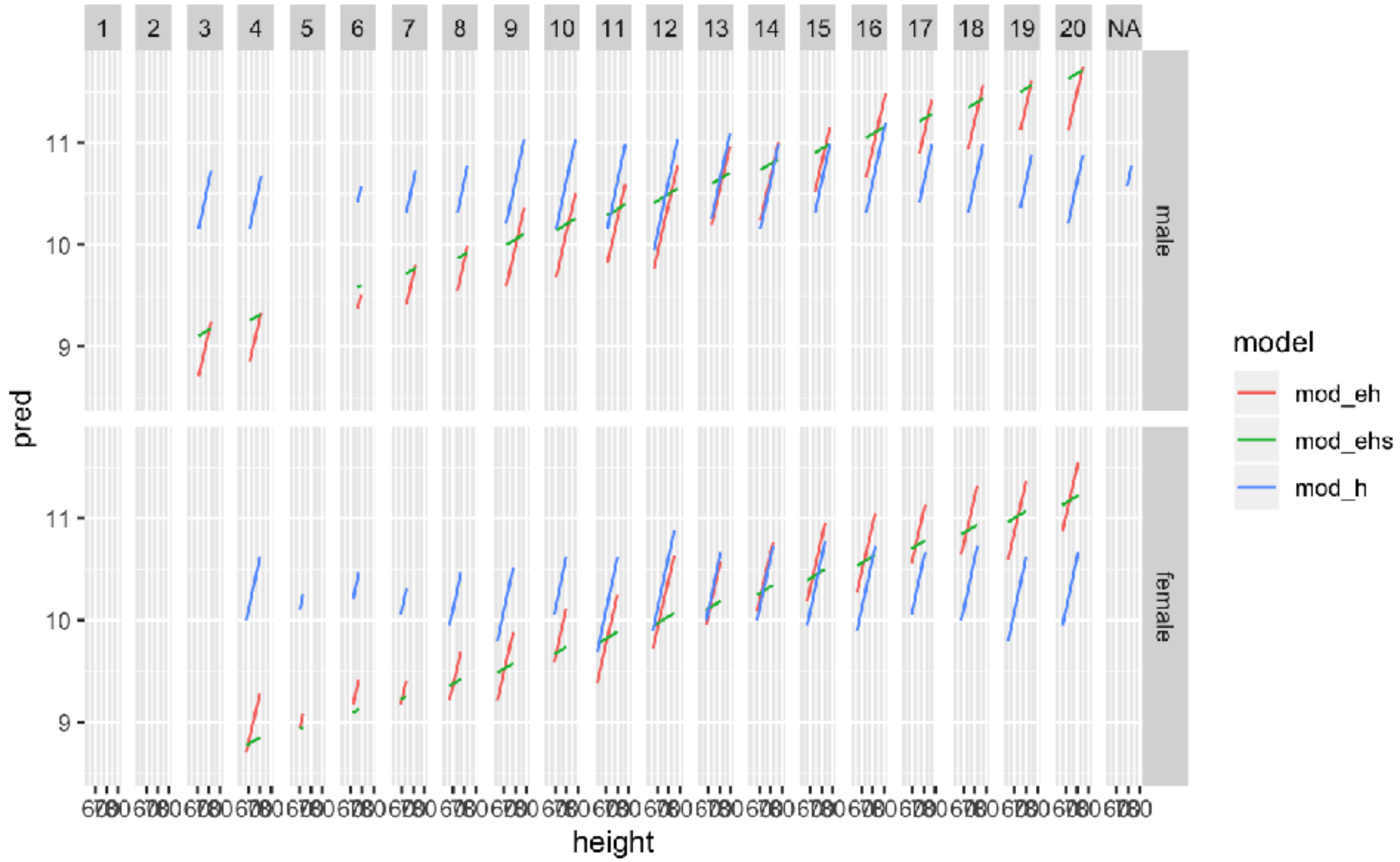
ATLAS

modelr

```
wages %>%
  gather_pedictions(mod_h, mod_eh, mod_ehs) %>%
  ggplot(mapping = aes(x = height, y = pred, color = model)) +
    geom_line()
```



```r
wages %>%
  gather_pedictions(mod_h, mod_eh, mod_ehs) %>%
  ggplot(mapping = aes(x = height, y = pred, color = model)) +
    geom_line() +
    facet_grid(rows = vars(sex), cols = vars(education))
```



ATLAS

facet_grid()

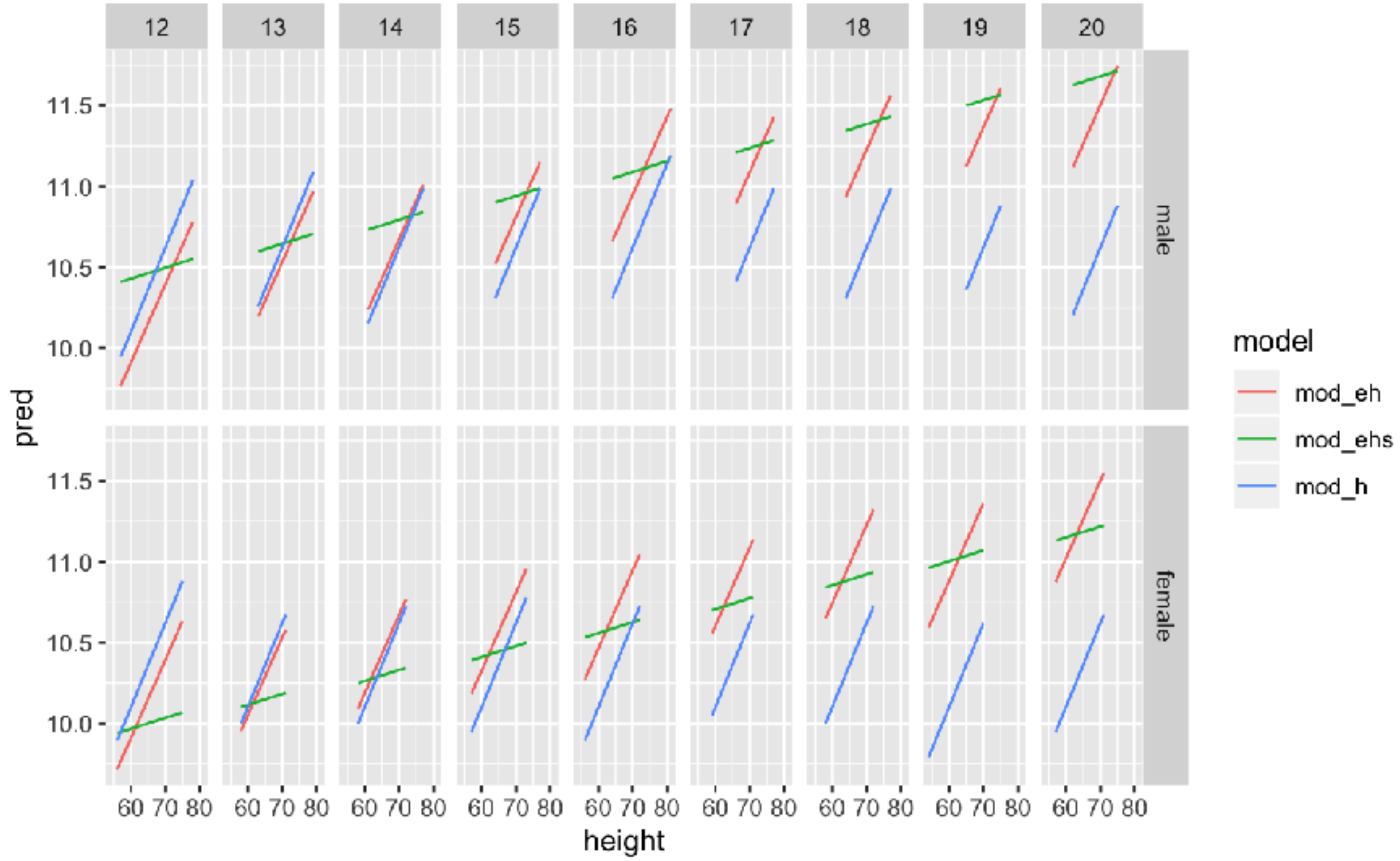
Divides plot into subplots based on a grouping variable.
Forms a matrix of panels based on row and column faceting variables.

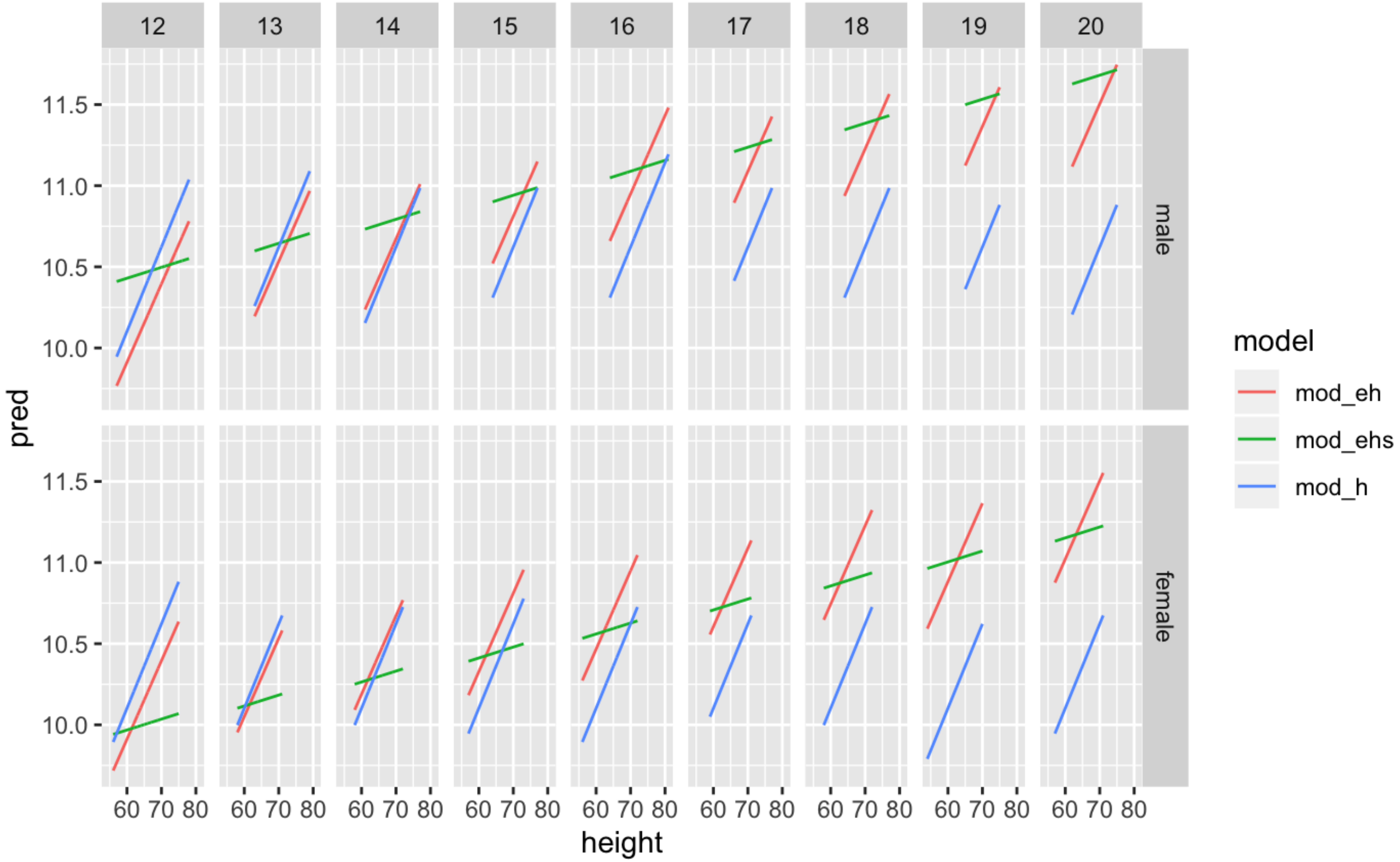
```
+ facet_grid(rows = vars(sex), cols = vars(education))
```

The grouping variable for the rows

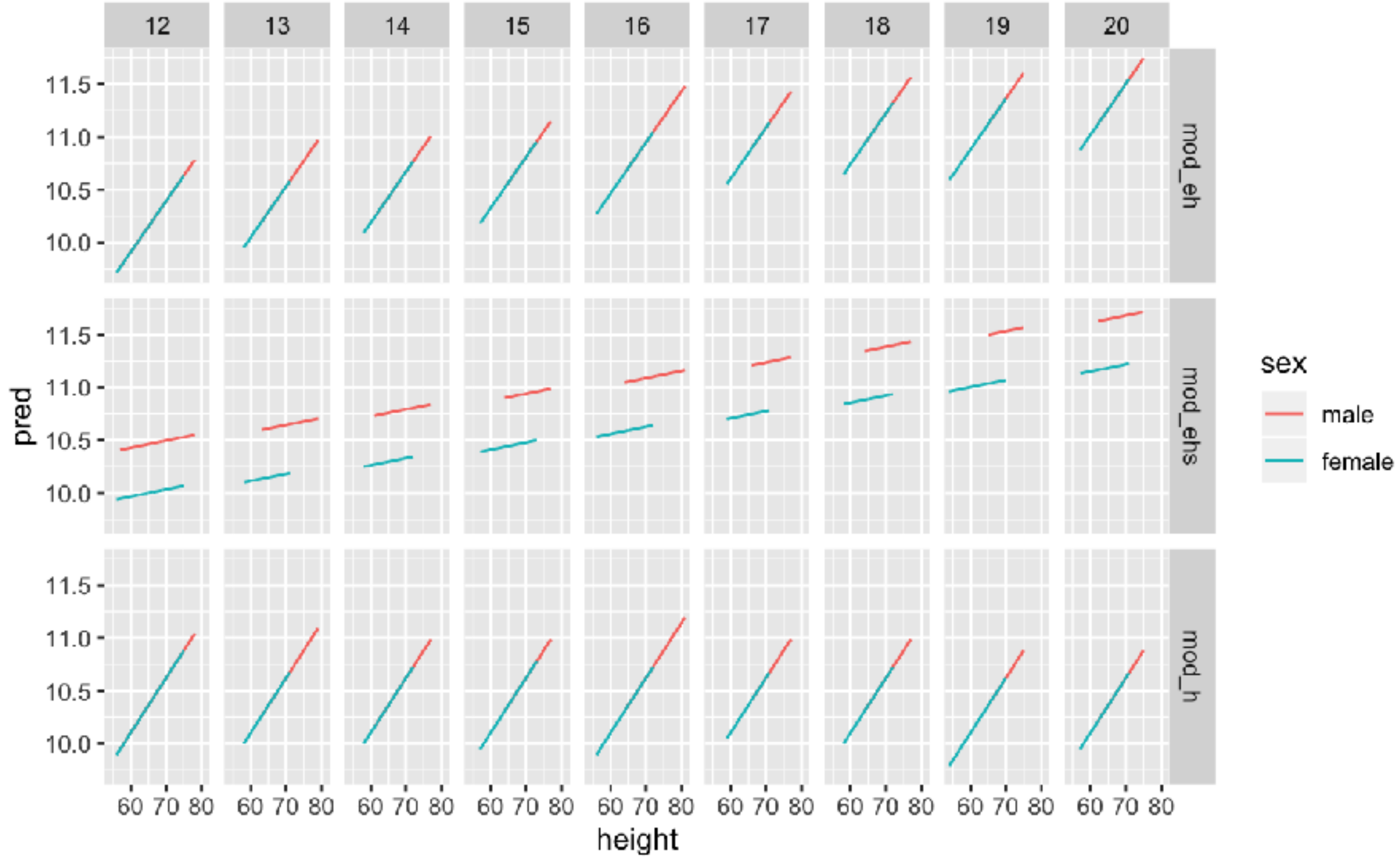
The grouping variable for the columns

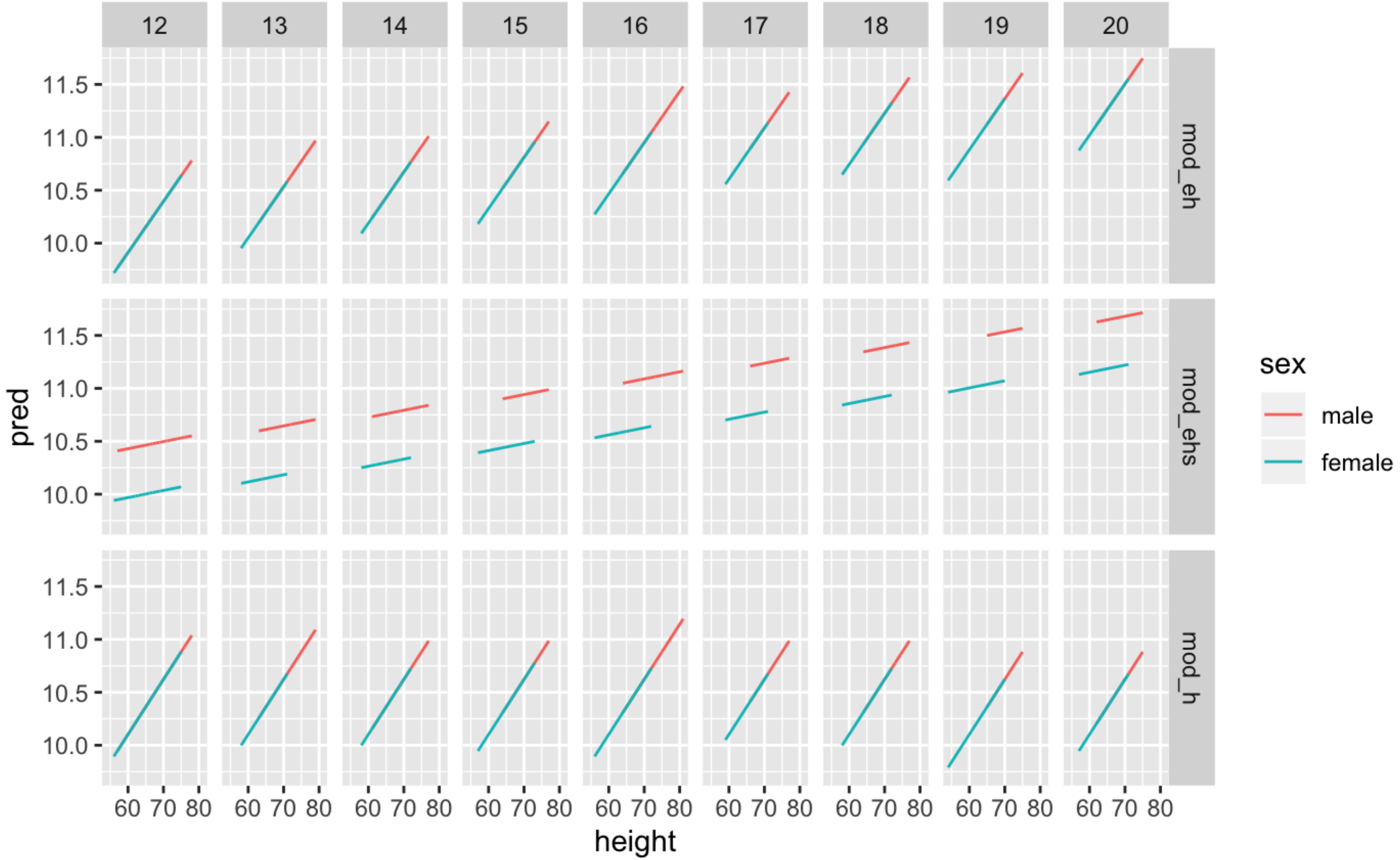
```
wages %>%
  gather_pedictions(mod_h, mod_eh, mod_ehs) %>%
  filter(education > 11) %>%
  ggplot(mapping = aes(x = height, y = pred, color = model)) +
    geom_line() +
    facet_grid(rows = vars(sex), cols = vars(education))
```




12
13
14
15
16
17
18
19
20
pred
11.5
11.0
10.5
10.0
male
female
60 70 80
height
model
mod_eh
mod_ehs
mod_h

```r
wages %>%
  gather_pedictions(mod_h, mod_eh, mod_ehs) %>%
  filter(education > 11) %>%
  ggplot(mapping = aes(x = height, y = pred, color = sex)) +
    geom_line() +
    facet_grid(rows = vars(model), cols = vars(education))
```





ATLAS

modelr

Residuals

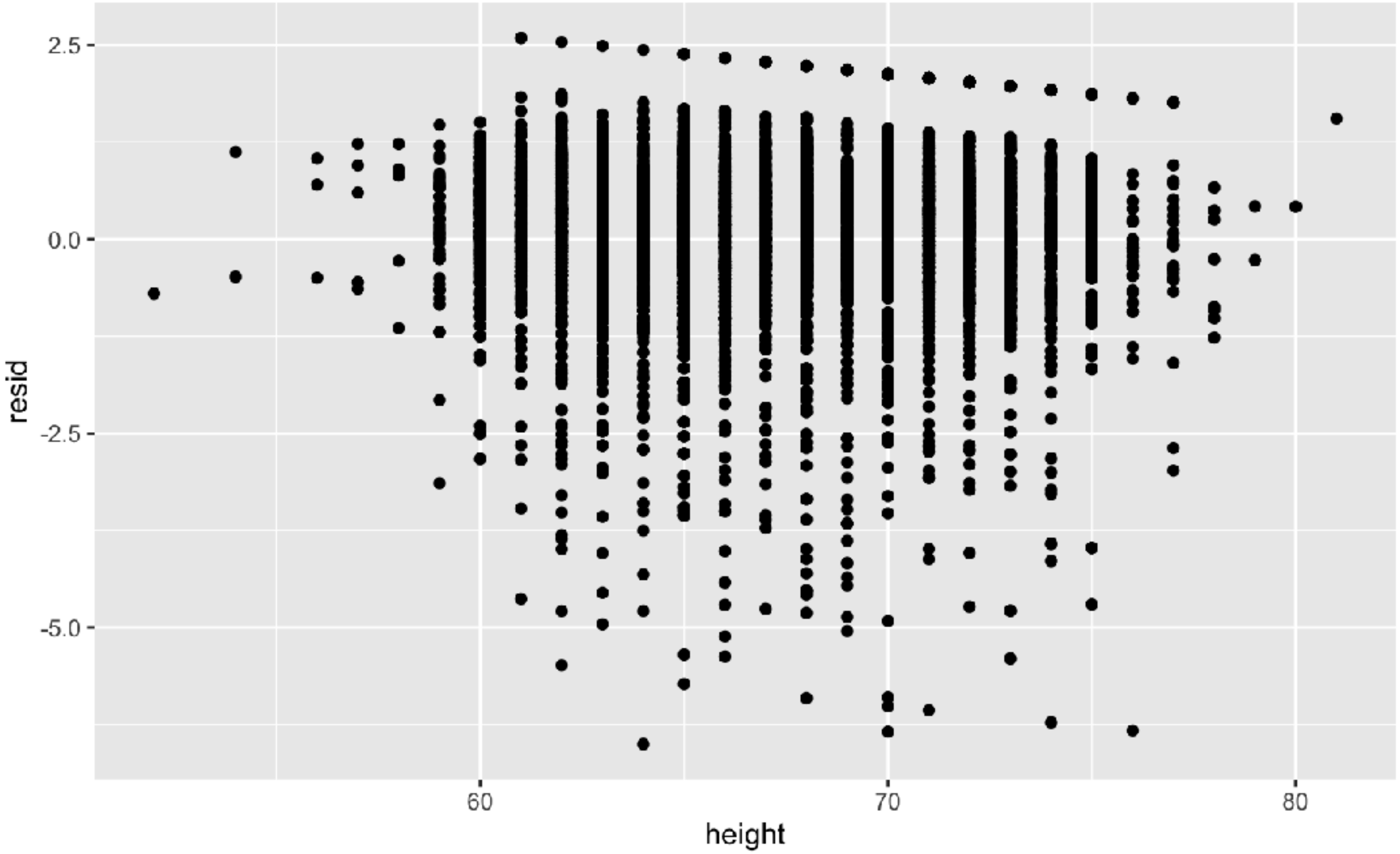
modelr provides the equivalent functions for residuals

`add_predictions()` → `add_residuals()`

`spread_predictions()` → `spread_residuals()`

`gather_predictions()` → `gather_residuals()`

```
wages %>%
  add_residuals(mod_h) %>%
  ggplot(mapping = aes(x = height, y = resid)) +
    geom_point()
```



Your Turn 8

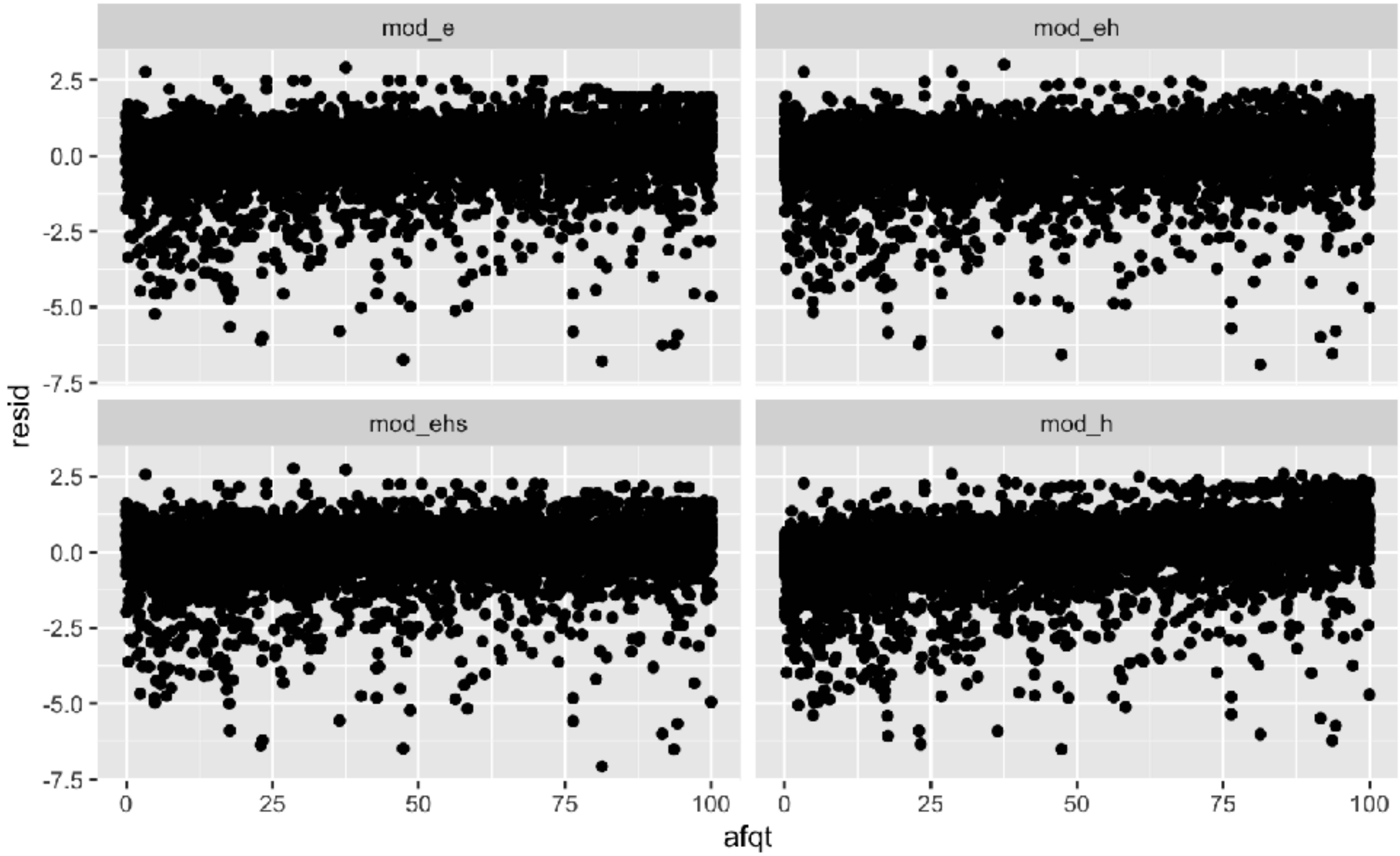
Use one of **spread_residuals()** or **gather_residuals()** to make a scatter plot of **afqt** vs. **resid** for each of **mod_e**, **mod_h**, **mod_eh**, and **mod_ehs**.

Use a faceting function to create a subplot for each model.

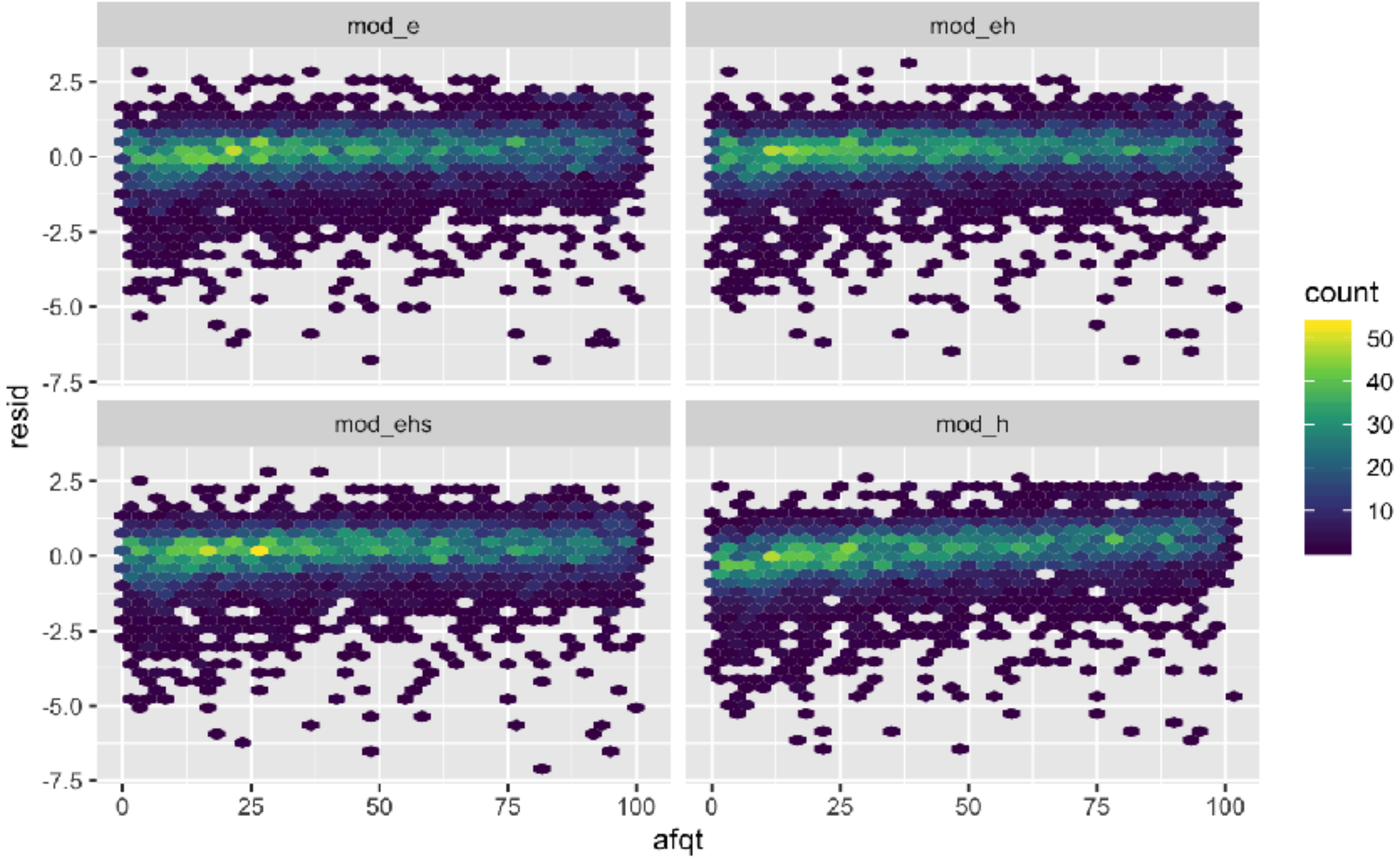
04 : 00

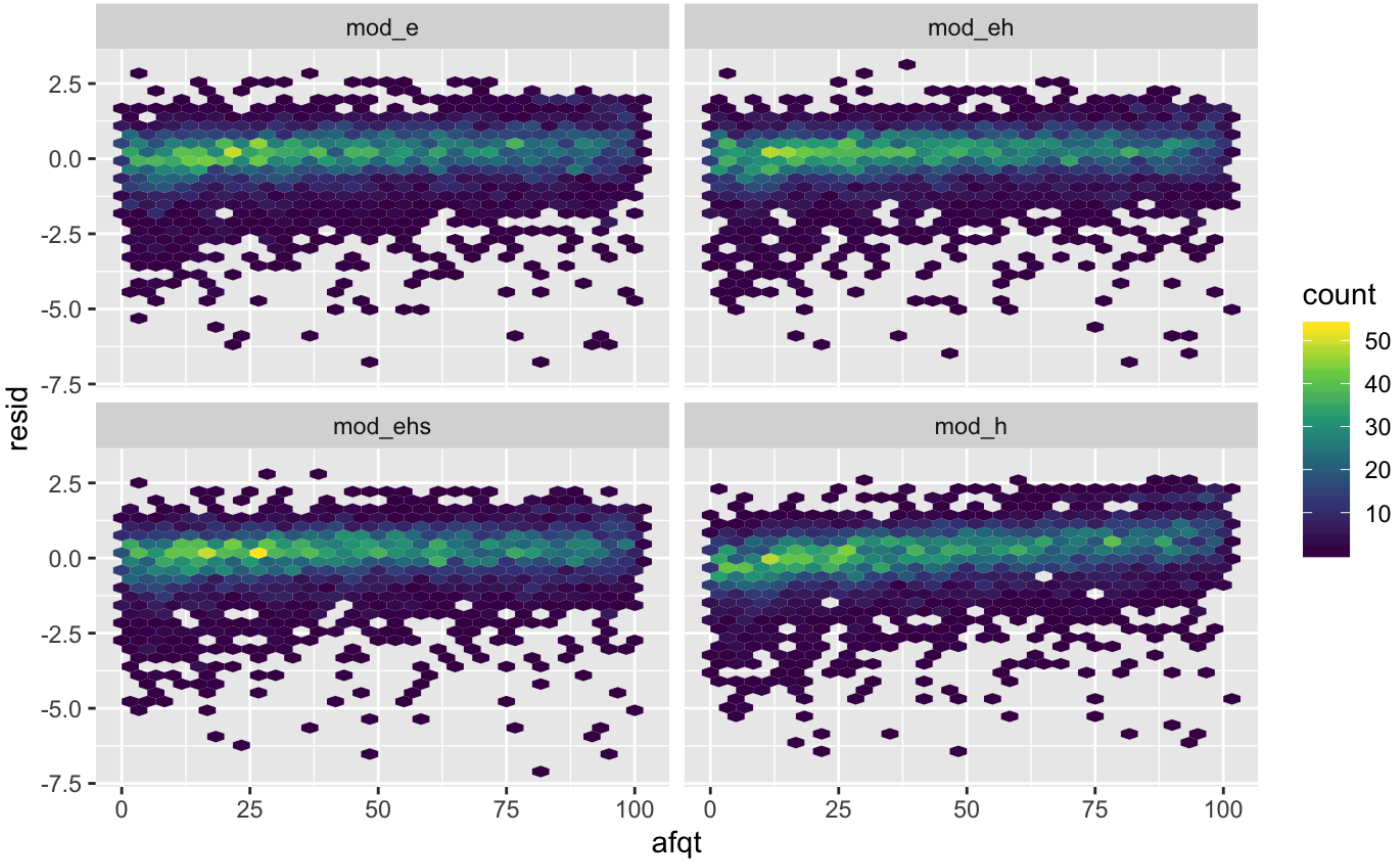
ATLAS

```r
wages %>%
  gather_residuals(mod_e, mod_h, mod_eh, mod_ehs) %>%
  ggplot(mapping = aes(x = afqt, y = resid)) +
    geom_point() +
    facet_wrap(vars(model))
```



```r
wages %>%
  gather_residuals(mod_e, mod_eh, mod_ehs, mod_h) %>%
  ggplot(mapping = aes(x = afqt, y = resid)) +
    geom_hex() +
    scale_fill_viridis_c() +
    facet_wrap(vars(model))
```




mod_e
mod_eh
mod_ehs
mod_h
2.5
0.0
-2.5
-5.0
-7.5
resid
0
25
50
75
100
afqt
count
50
40
30
20
10

Recap

Use **glance()**, **tidy()**, and **augment()** to return model values in a data frame.

Use **add_predictions()**, **spread_predictions()**, or **gather_predictions()** to visualize predictions.

Use **add_residuals()**, **spread_residuals()**, or **gather_residuals()** to visualize residuals.

Model Data

broom

modelr

ATLAS

Accessible Teaching, Learning, and Assessment Systems