Model comparison, interactions

Frank Edwards

3/6/2020

How to assess a model

If our goal is causal inference

1. Construct causal theories (DAGs)
2. Determine testable implications of model (conditional indepencies)
3. Evaluate, critique, refine

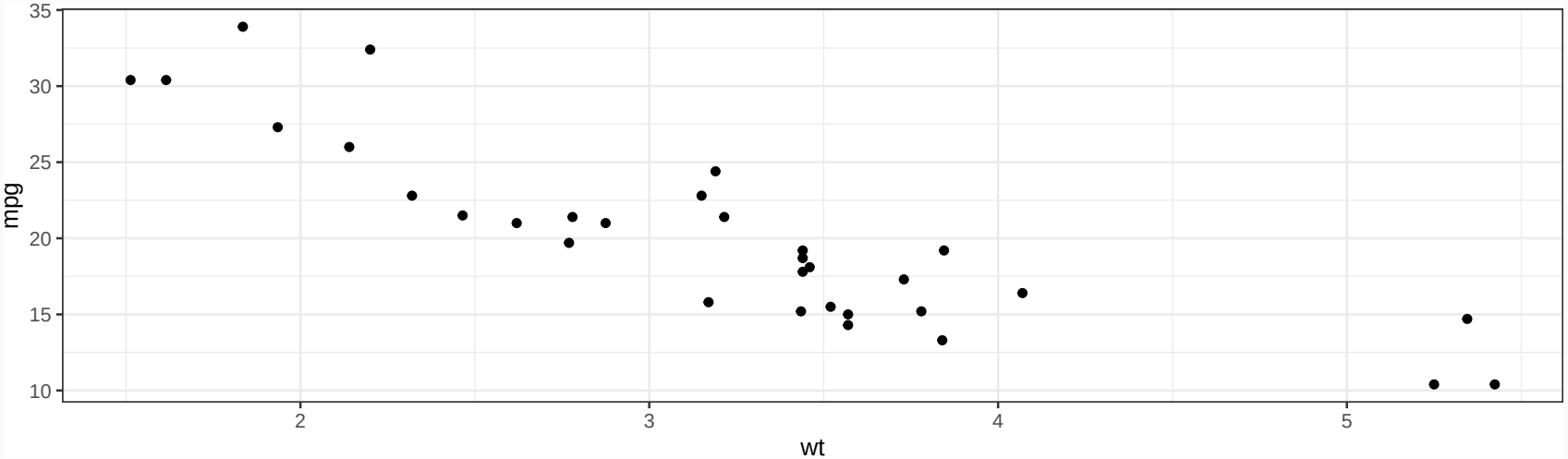
If our goal is prediction

1. Establish what we'd like to predict
2. Estimate competing models
3. Compare with metric of accuracy / fit

Overfitting

Data for today

```
data(mtcars)
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point()
```



The most common measure of model fit: R^2

$$R^2 = \frac{var(\text{outcome}) - var(\text{residuals})}{var(\text{outcome})}$$

$$R^2 = 1 - \frac{var(\text{residuals})}{var(\text{outcome})}$$

R^2 in action

```
m1<-quap(alist(
  mpg ~ dnorm(mu, sigma),
  mu<-a + bW * wt,
  a ~ dnorm(0, 10),
  bW ~ dnorm(0, 5),
  sigma ~ dexp(1)
  ), data = mtcars)
```

```
## [1] 0.7494697
```

R^2 tells us how much of the variance in the outcome variable is predictable based on the included predictor variable(s). Here: we can predict 75 percent of the observed variance in *mpg* after knowing *wt*

Let’s work with a sample of the data

```
car_sample<-mtcars %>%
  sample_n(10)

d<-car_sample
```

For the linear model with weight as a predictor, $R^2 = $

```
## [1] 0.794339
```

Can we improve the fit?

Yes! Let's use this linear function:

$$\mu = \alpha + \beta_1 W + \beta_2 W^2$$

$R^2 = $

```
## [1] 0.8438117
```

Can we improve the fit?

Yes!

$$\mu = \alpha + \beta_1 W + \beta_2 W^2 + \beta_3 W^3$$

$R^2 = $

```
## [1] 0.8584337
```

Can we improve the fit?

Yes!

$$\mu = \alpha + \beta_1 W + \beta_2 W^2 + \beta_3 W^3 + \beta_4 W^4$$

$R^2 = $

```
## [1] 0.9018586
```

Can we improve the fit?

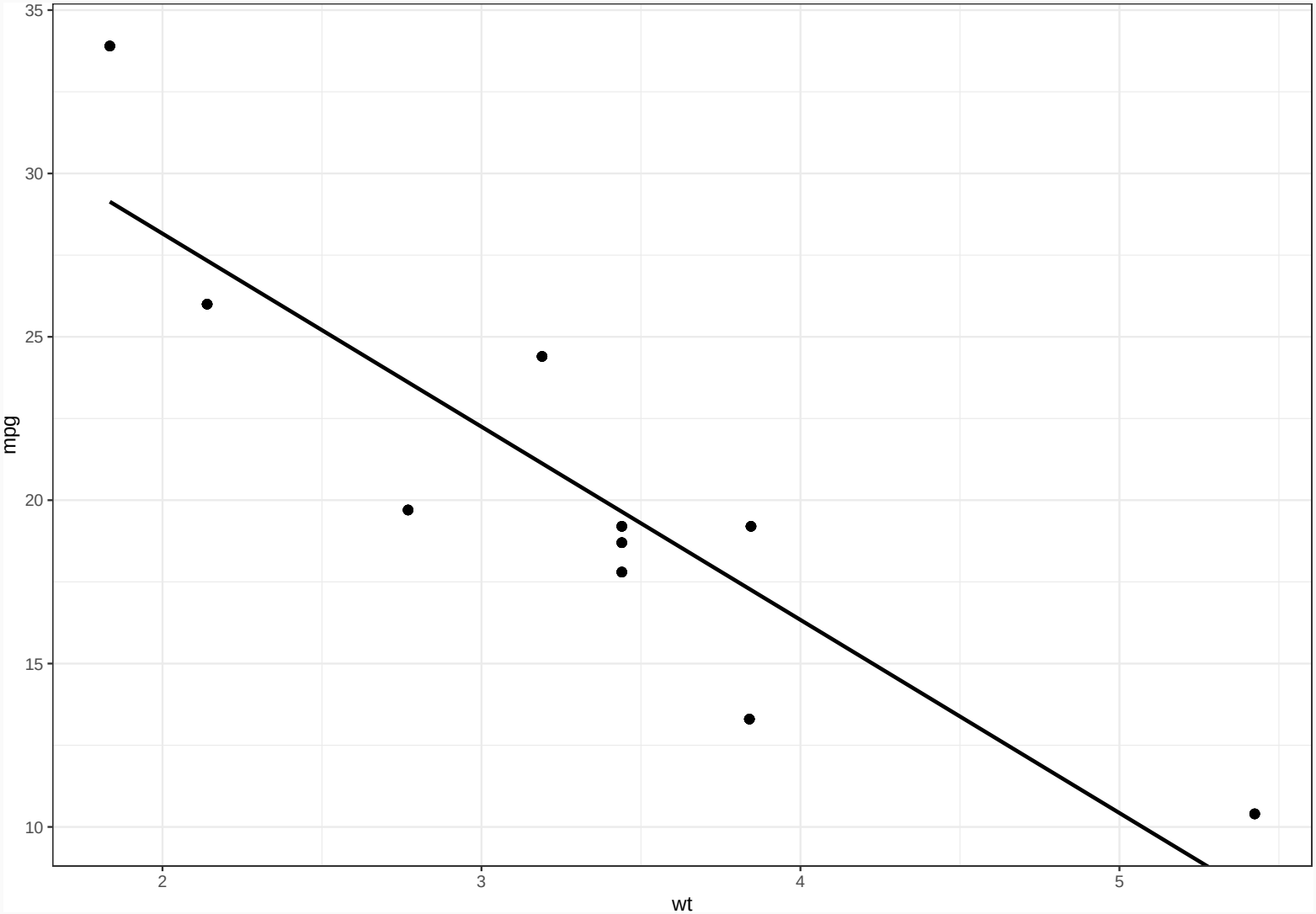
Yes!

$$\mu = \alpha + \beta_1 W + \beta_2 W^2 + \beta_3 W^3 + \beta_4 W^4 + \beta_5 W^5 + \beta_6 W^6$$

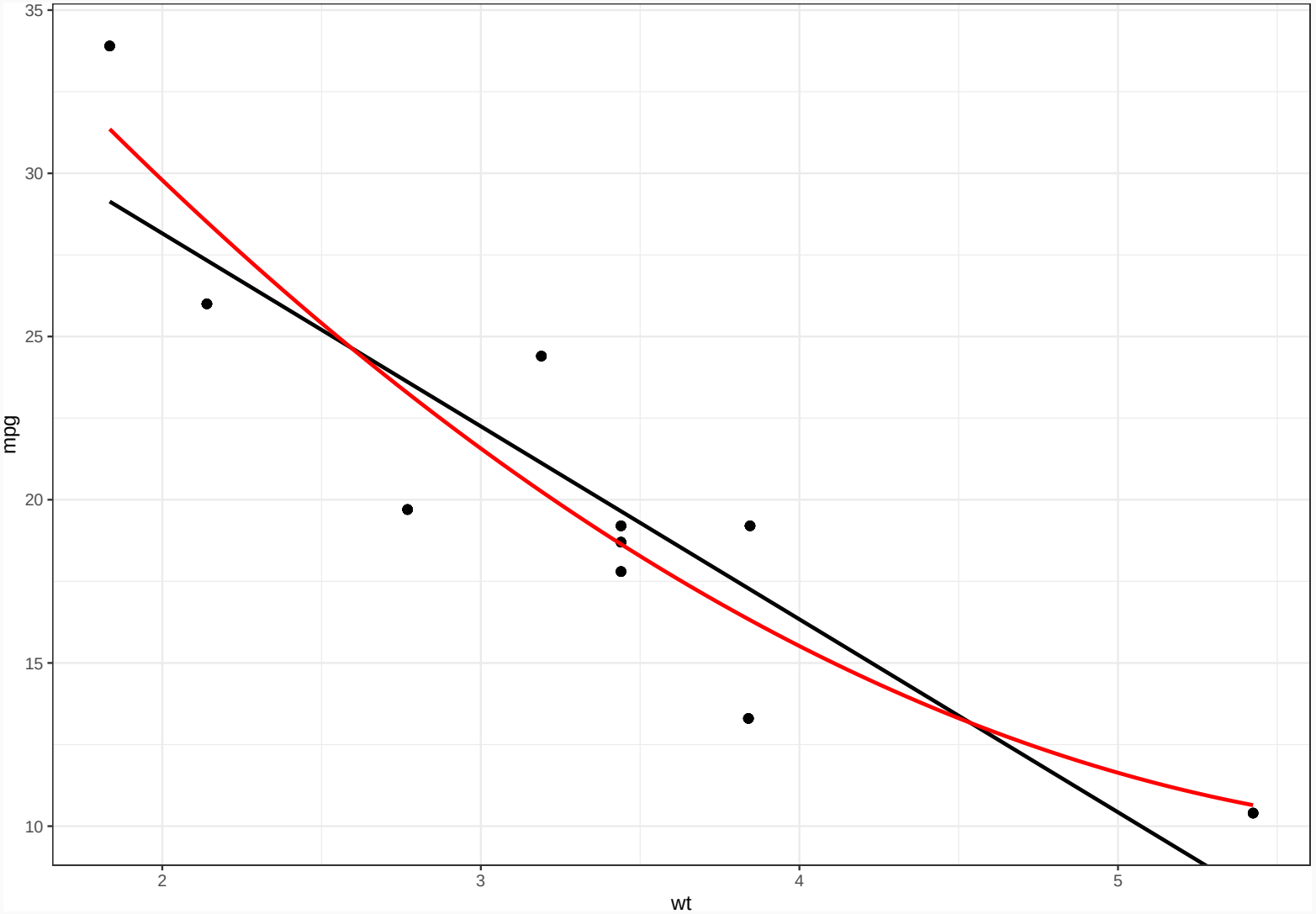
$R^2 = $

```
## [1] 0.9567988
```

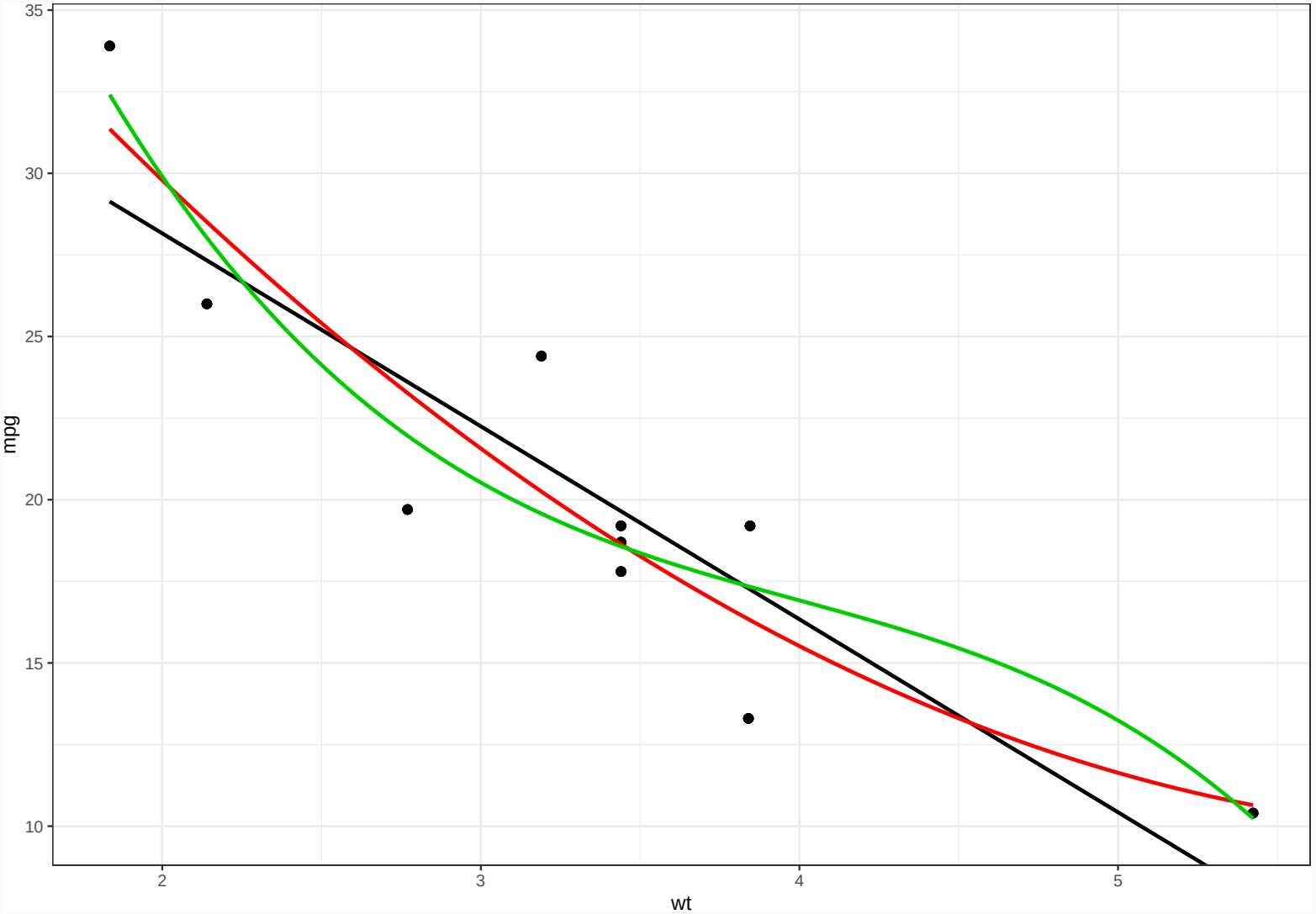
Compare these fits



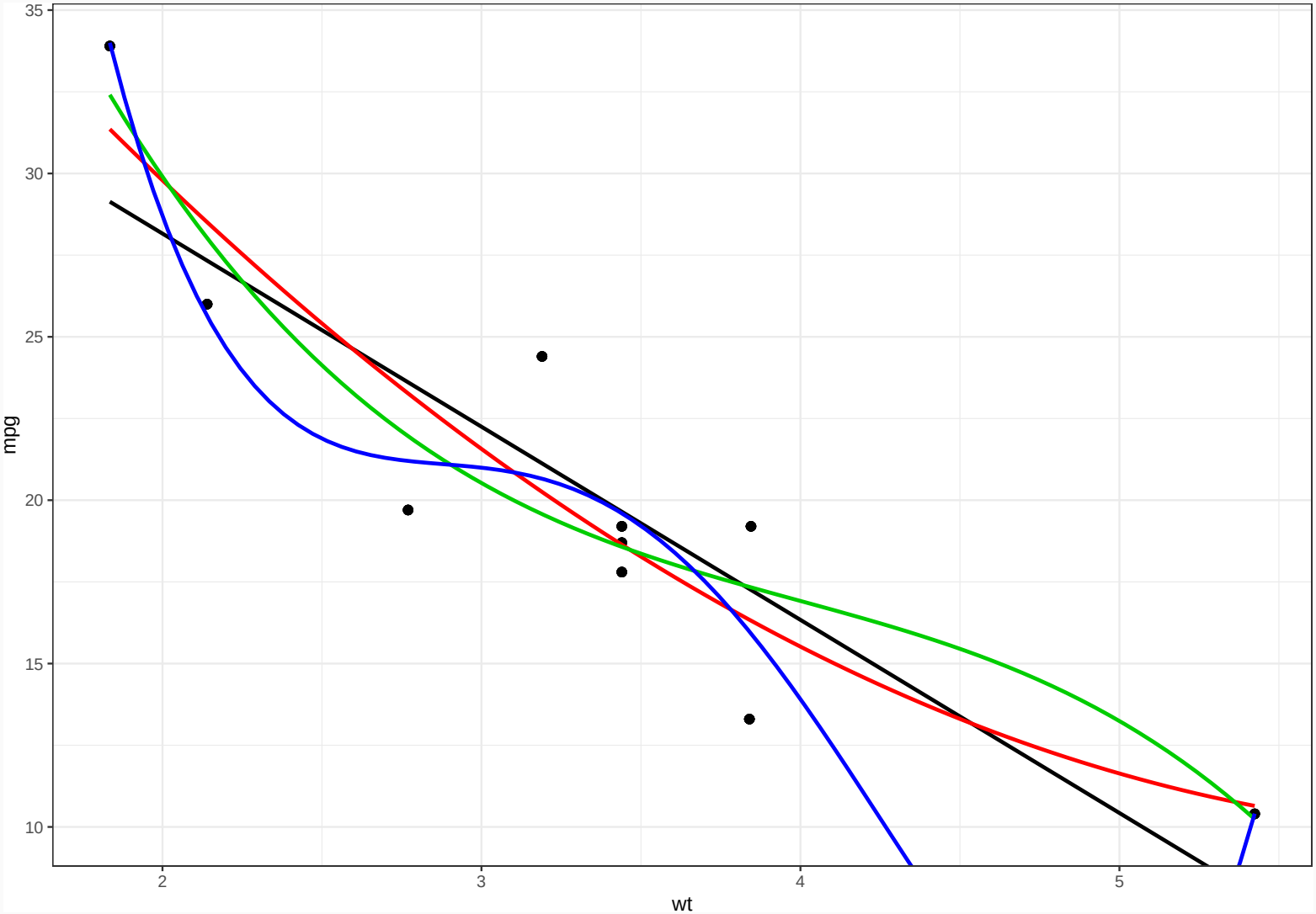
Compare these fits



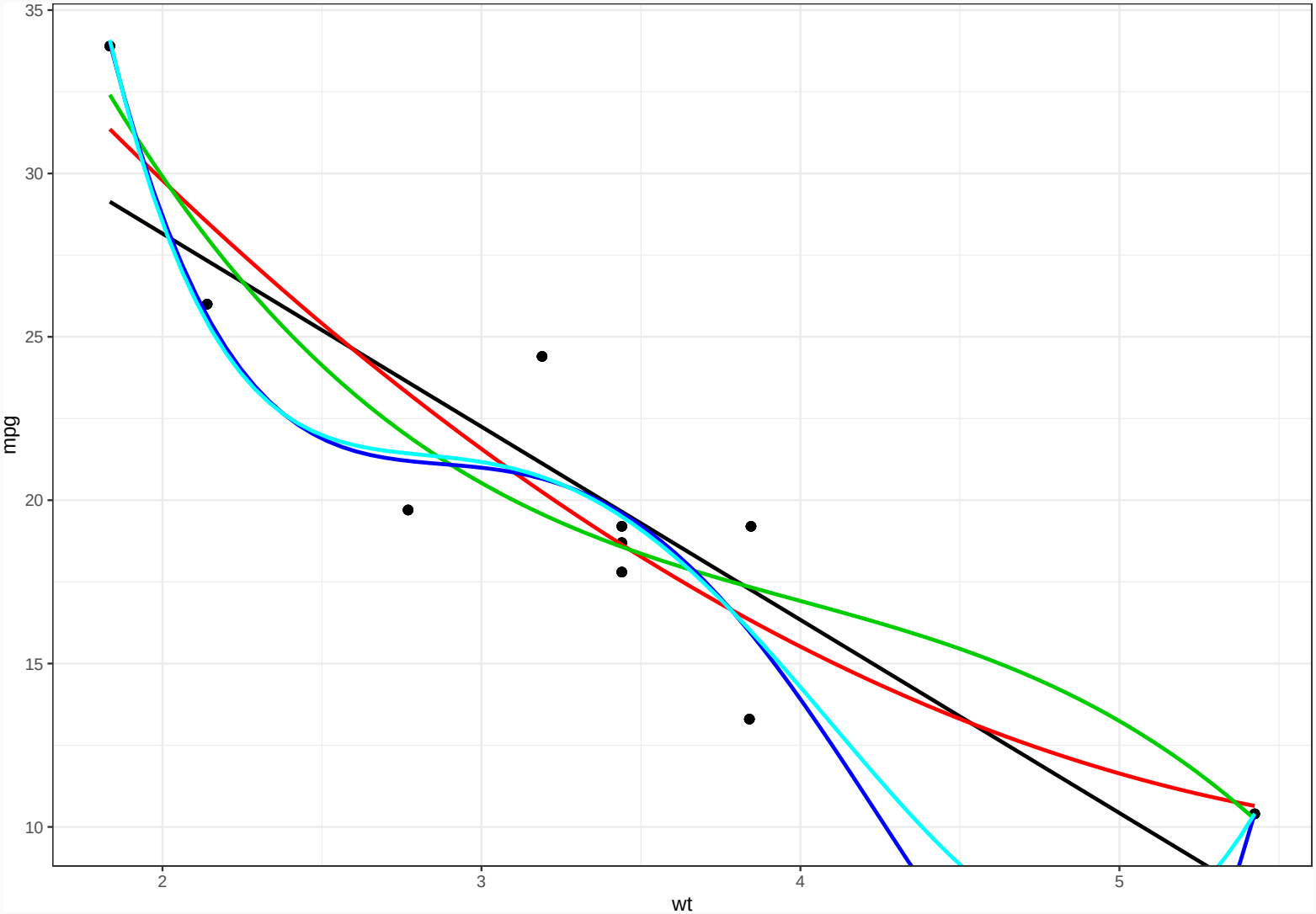
Compare these fits



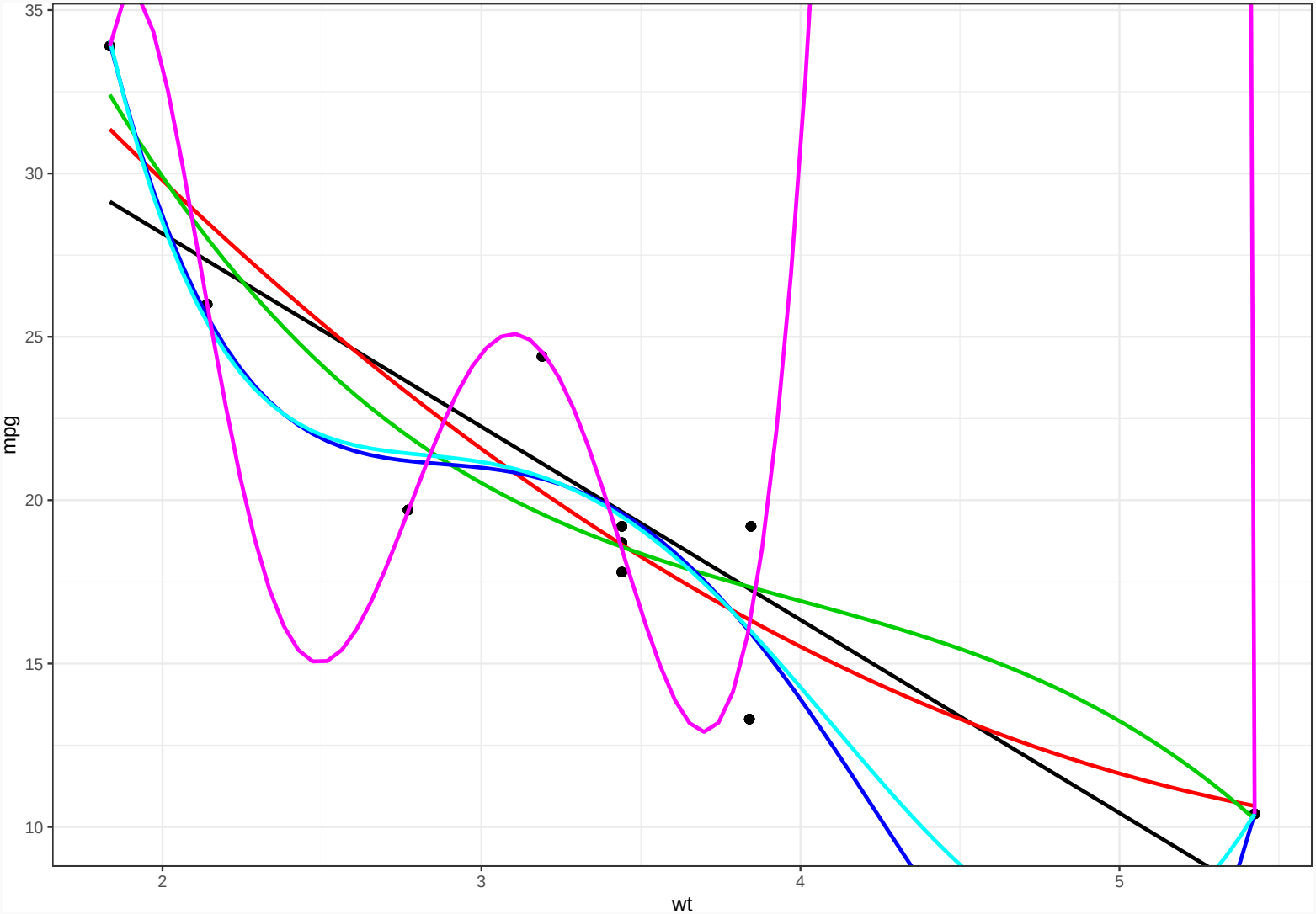
Compare these fits



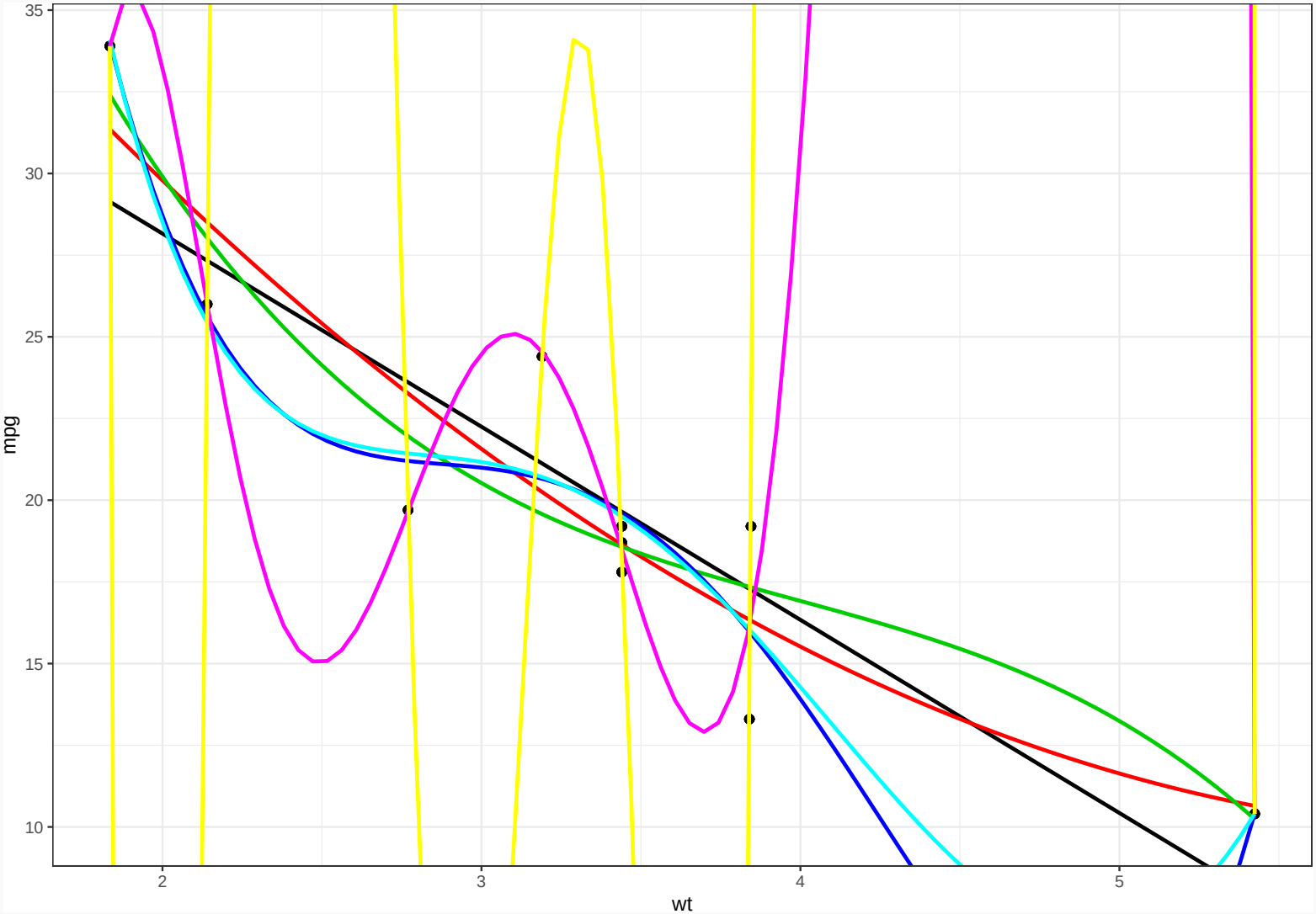
Compare these fits



Compare these fits



Compare these fits

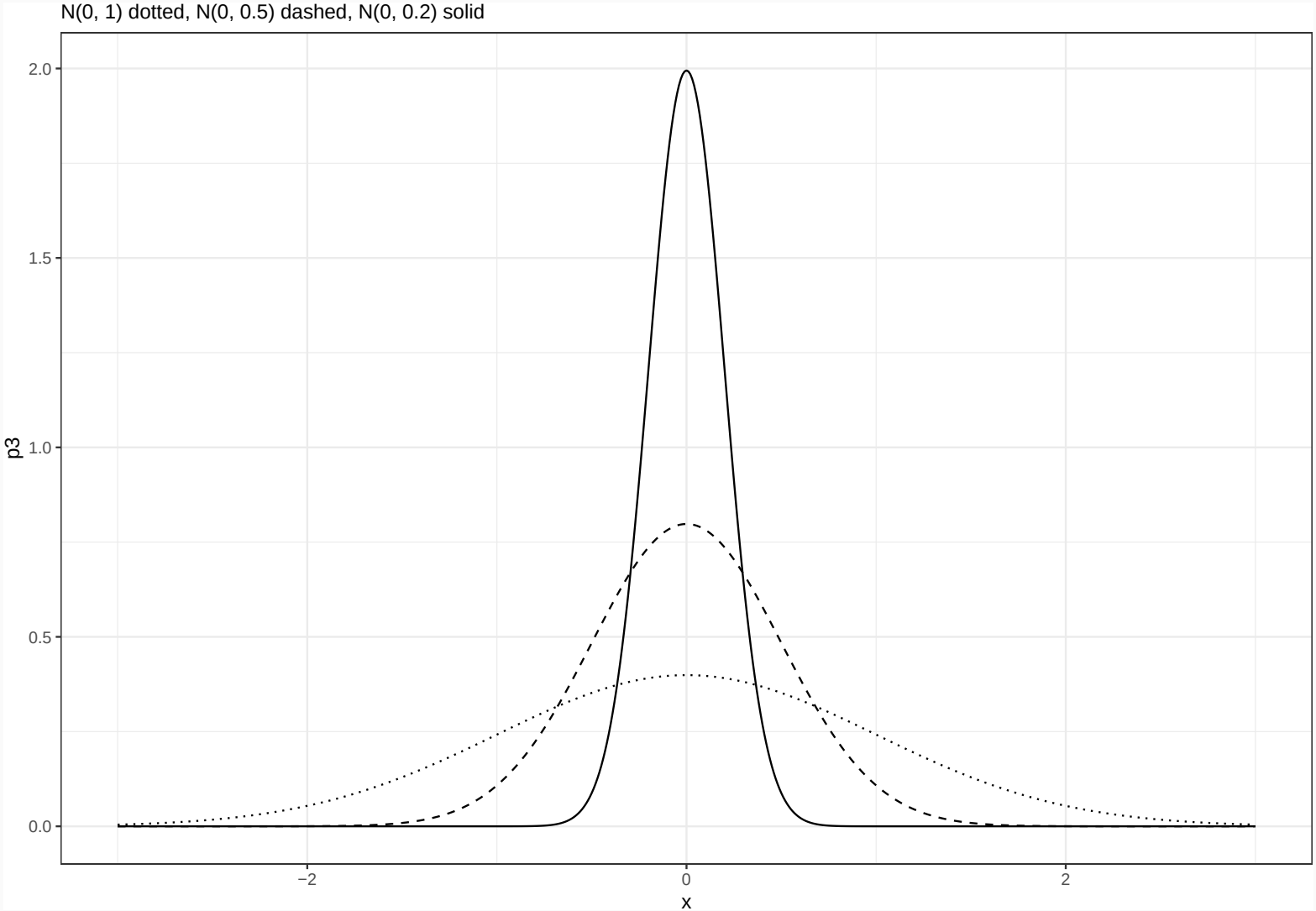


Overfitting and underfitting

- Adding complexity will improve goodness-of-fit measures like R^2
- But goodness-of-fit doesn't mean that we've modeled the process well, just that we've fit our *sample* well
- Adding too few parameters though can mean we aren't learning enough from the data
- *Goal:* balance model complexity with predictive accuracy

How to address overfitting

Regularizing priors



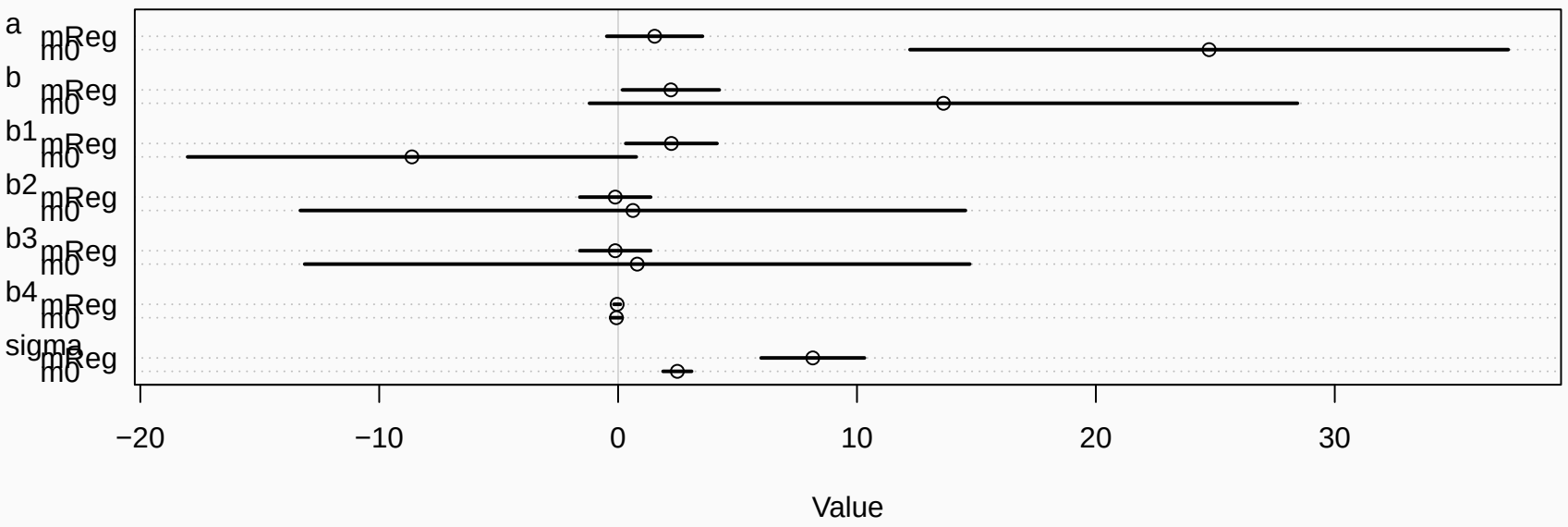
Use these priors to fit our polynomial model

```
m0<-quap(alist(
  mpg ~ dnorm(mu, sigma),
  mu<-a + b * wt + b1 * wt^2 + b2 * wt^3 + b3 * wt^3 + b4 * wt^4,
  a ~ dnorm(0, 10),
  b ~ dnorm(0, 10),
  b1 ~ dnorm(0, 10),
  b2 ~ dnorm(0, 10),
  b3 ~ dnorm(0, 10),
  b4 ~ dnorm(0, 10),
  sigma ~ dexp(1)
), data = mtcars)

mReg<-quap(alist(
  mpg ~ dnorm(mu, sigma),
  mu<-a + b * wt + b1 * wt^2 + b2 * wt^3 + b3 * wt^3 + b4 * wt^4,
  a ~ dnorm(0, 1),
  b ~ dnorm(0, 1),
  b1 ~ dnorm(0, 1),
  b2 ~ dnorm(0, 1),
  b3 ~ dnorm(0, 1),
  b4 ~ dnorm(0, 1),
  sigma ~ dexp(1)
), data = mtcars)
```

Narrower priors constrain overfitting

These priors force the model to learn less from the data. Flat priors allow the posterior to overfit the data.

```
plot(coeftab(m0, mReg))
```



Methods for comparing models

How well can we predict cases that we didn't use to fit the model?

Out of sample prediction is a general set of methods that evaluate how well a model performs at predicting new cases.

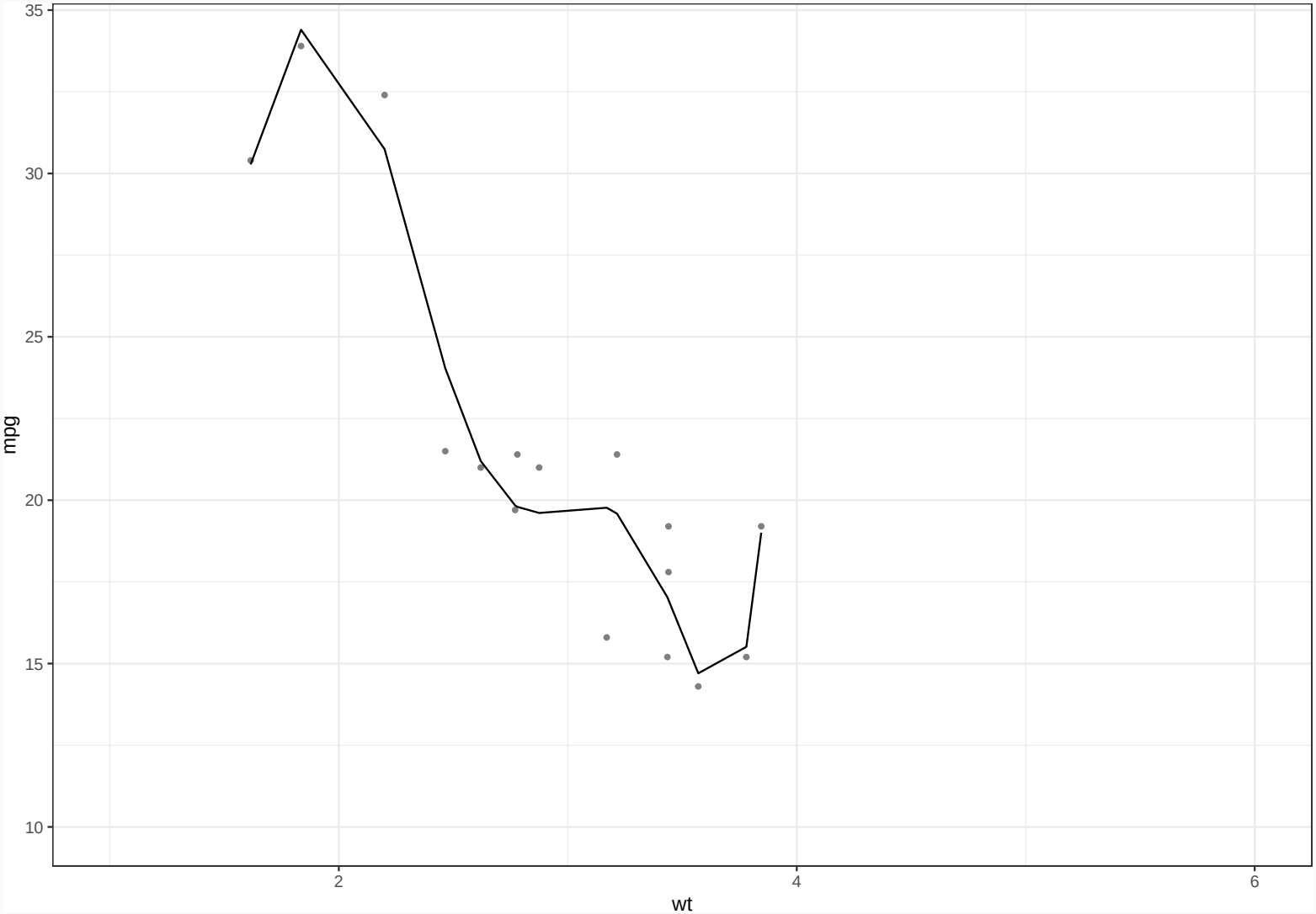
In general, we'd like to know how well our model predicts *new* observations, because it is easy to overfit on observed data.

Training and test methods

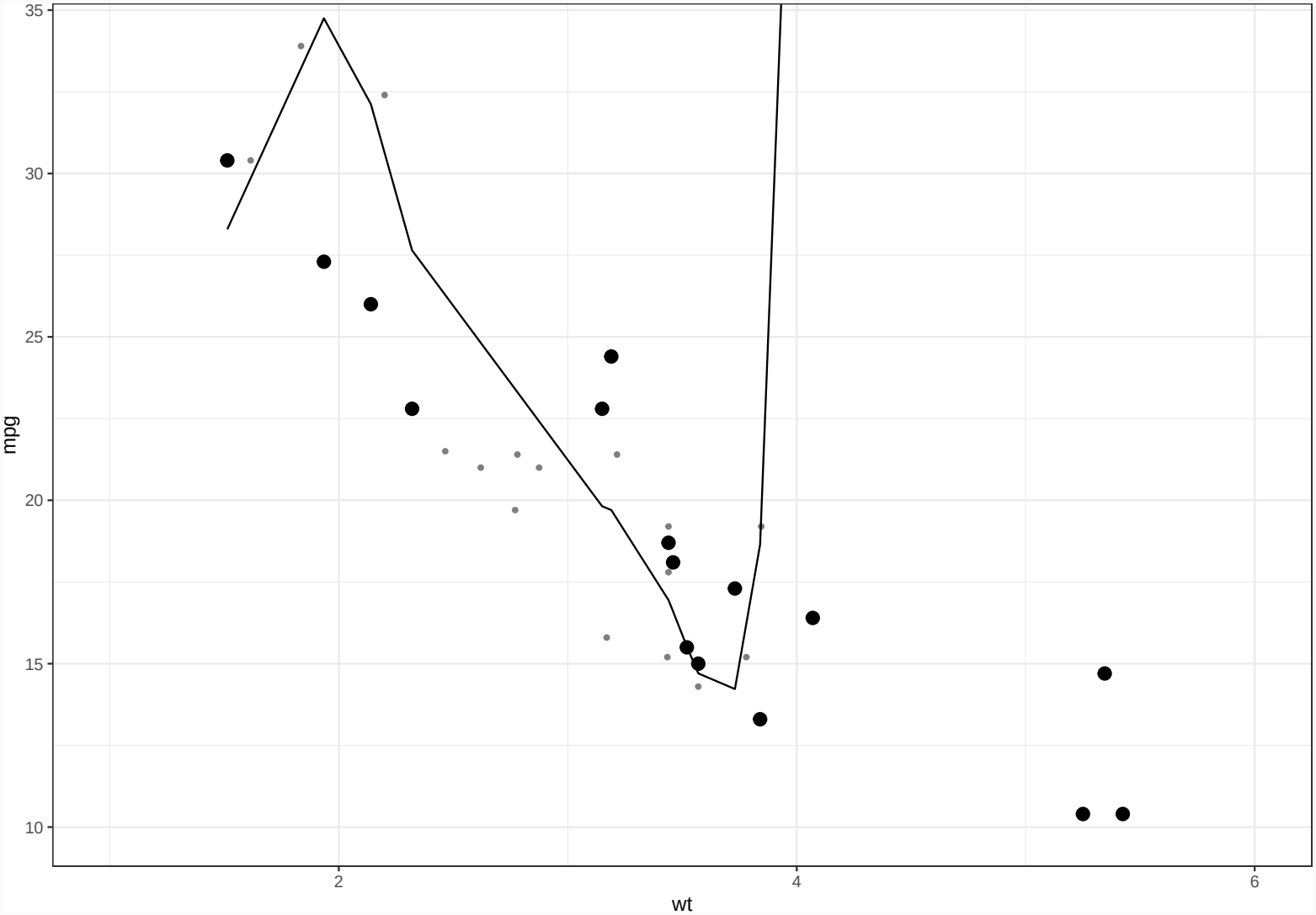
Let's partition the data into two equal sets: a *training* sample that we'll use to fit the model, and a *test* sample that we'll use to evaluate predictive accuracy

We'll extend this later to a general approach: *leave-one-out cross-validation*

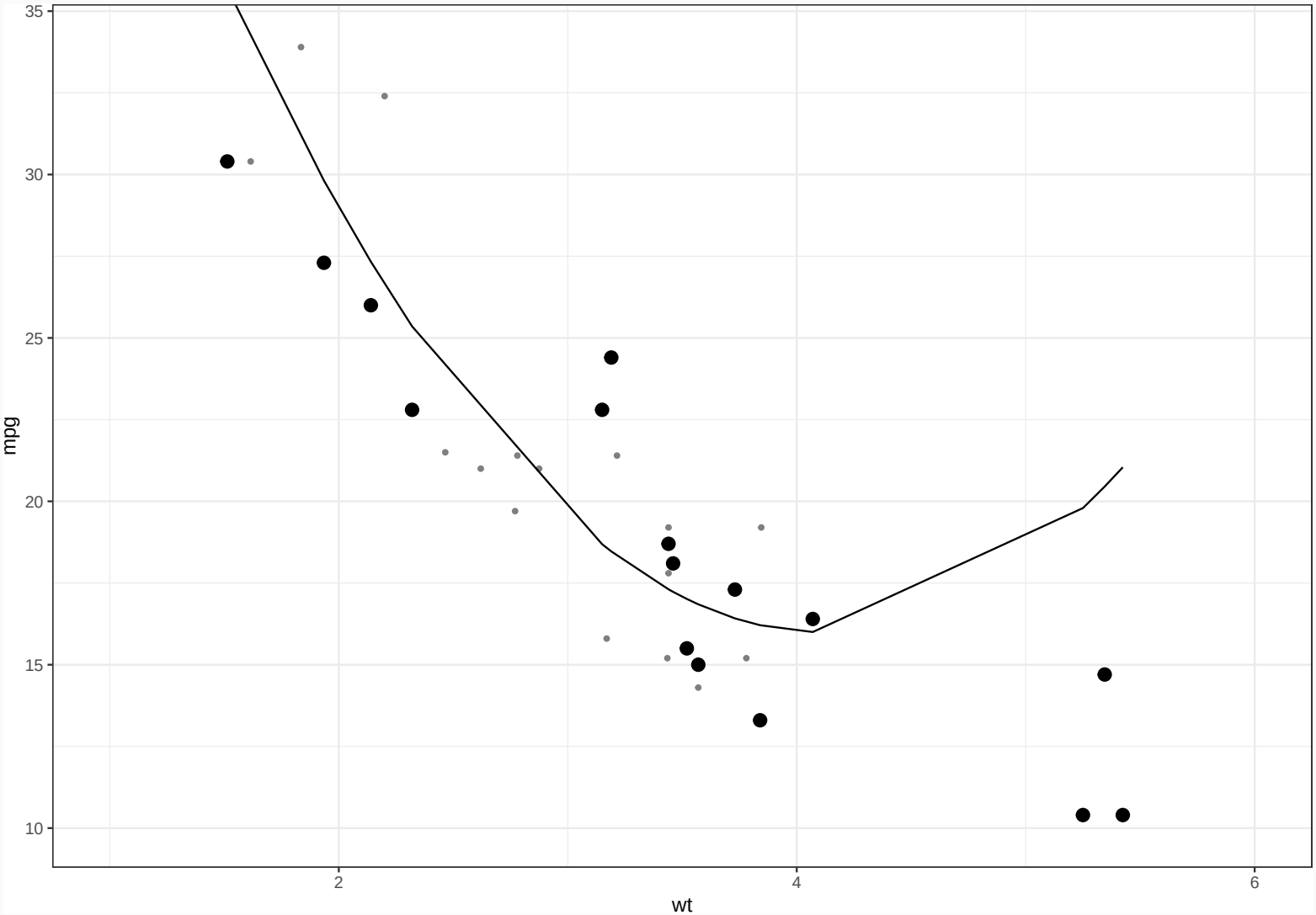
Checking the fit for our 6 degree polynomial model



Checking the fit for each of the models: compare to test data



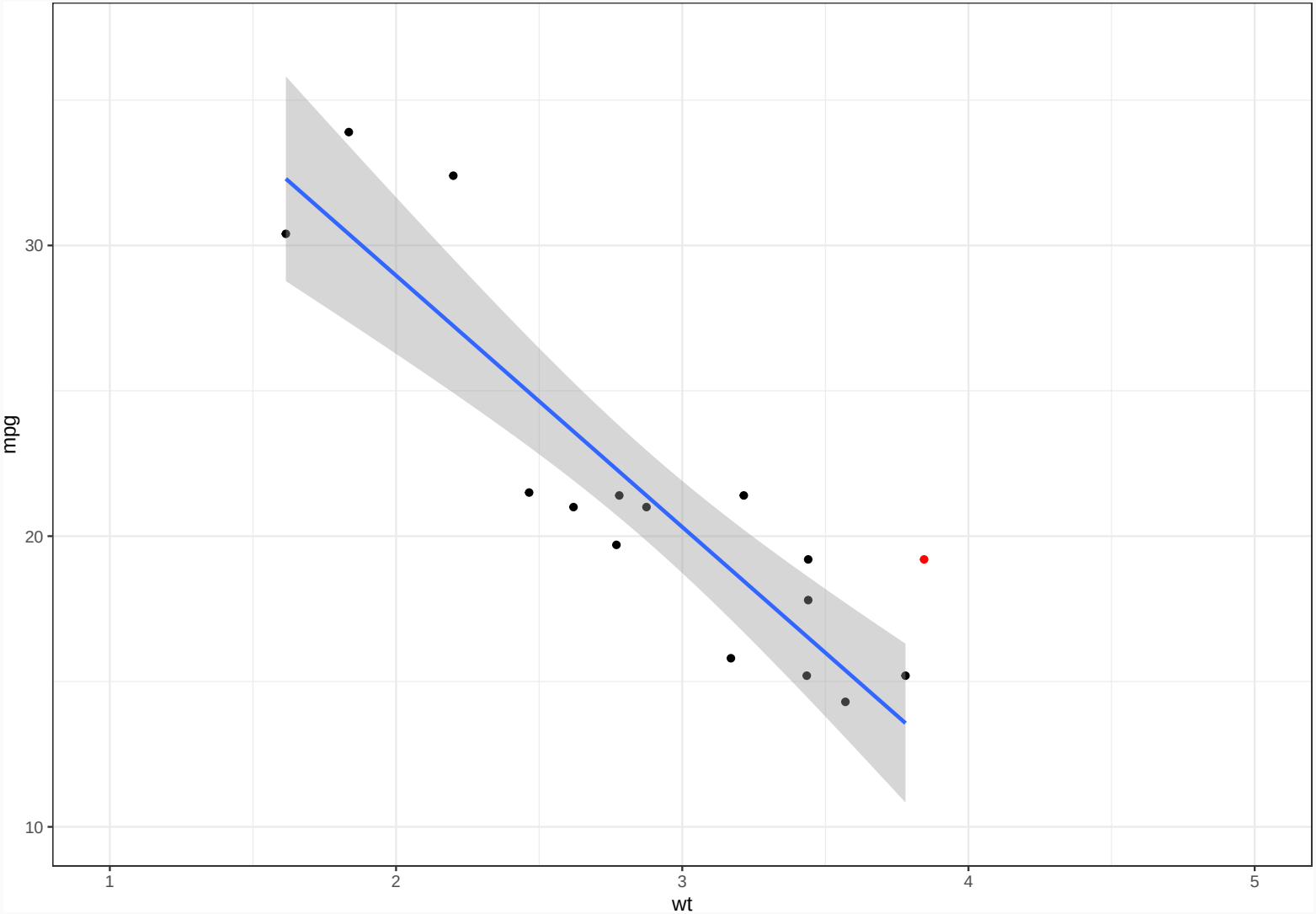
Checking the fit for each of the models: compare to test data, less complex model



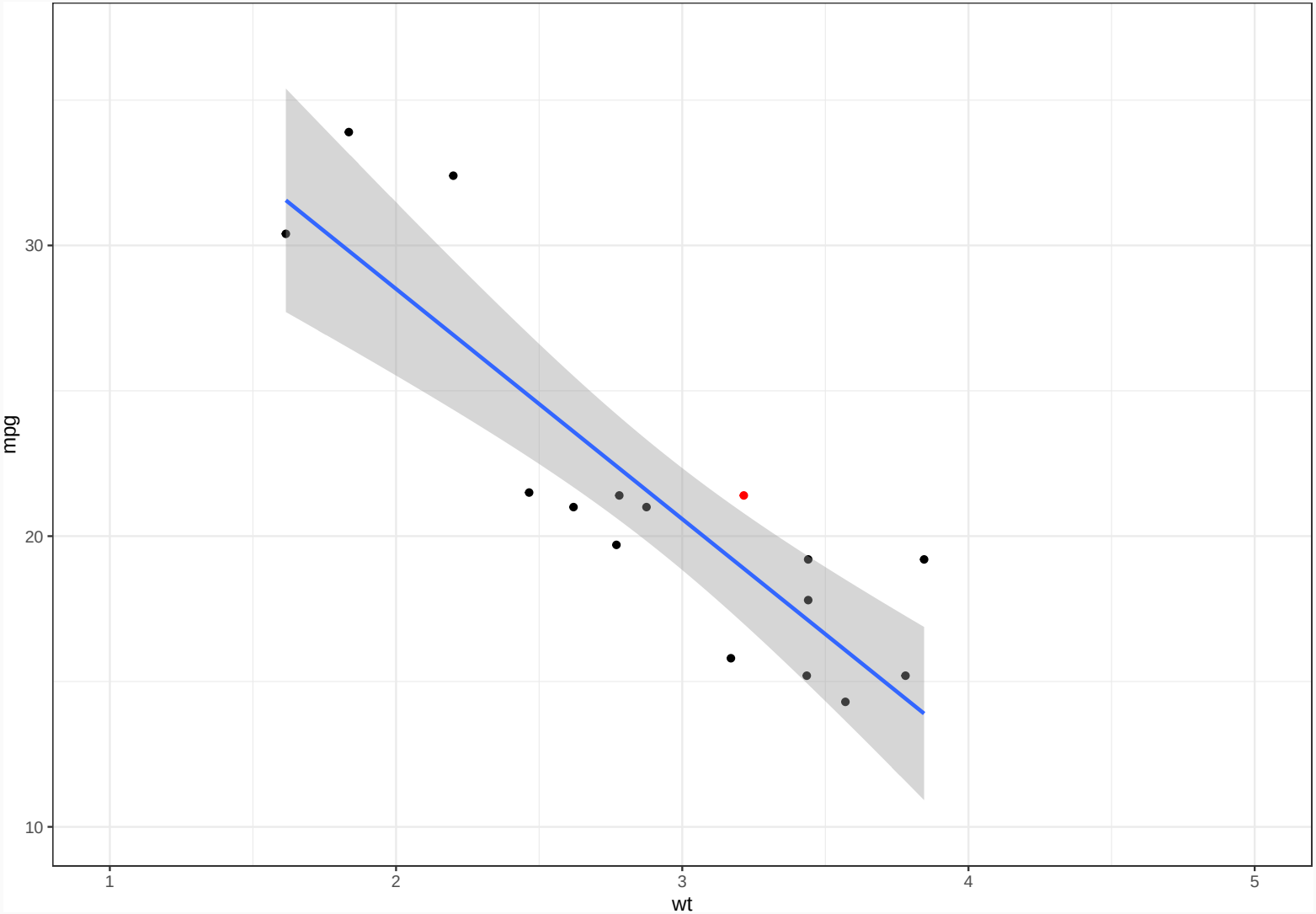
Leave-one-out cross validation

1. Define a model
2. Estimate the model, holding the first observation out
3. Predict the value for the held out observation, estimate prediction error
4. Repeat with 2:n observations
5. Average the error across each iteration

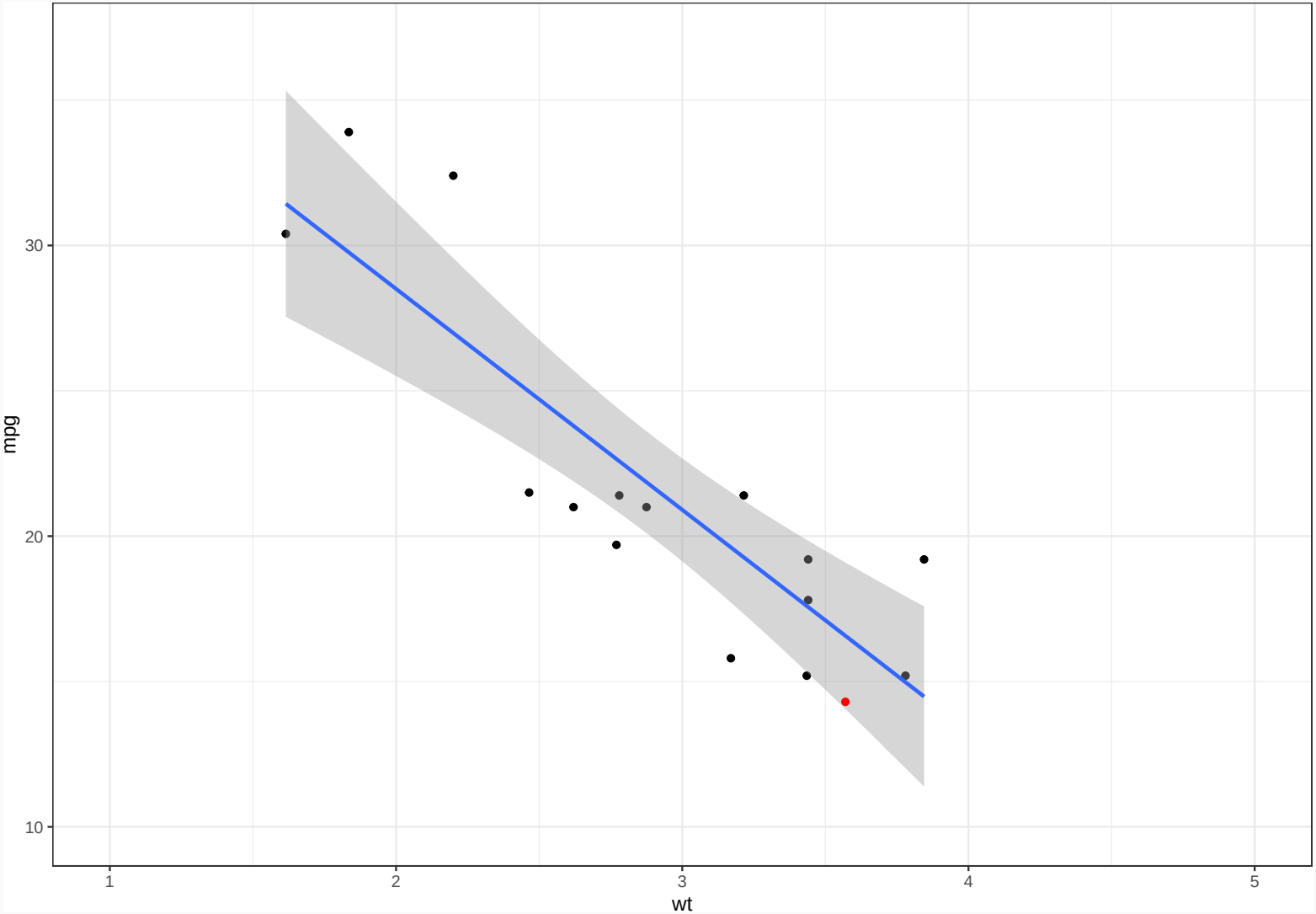
Leave-one-out cross validation



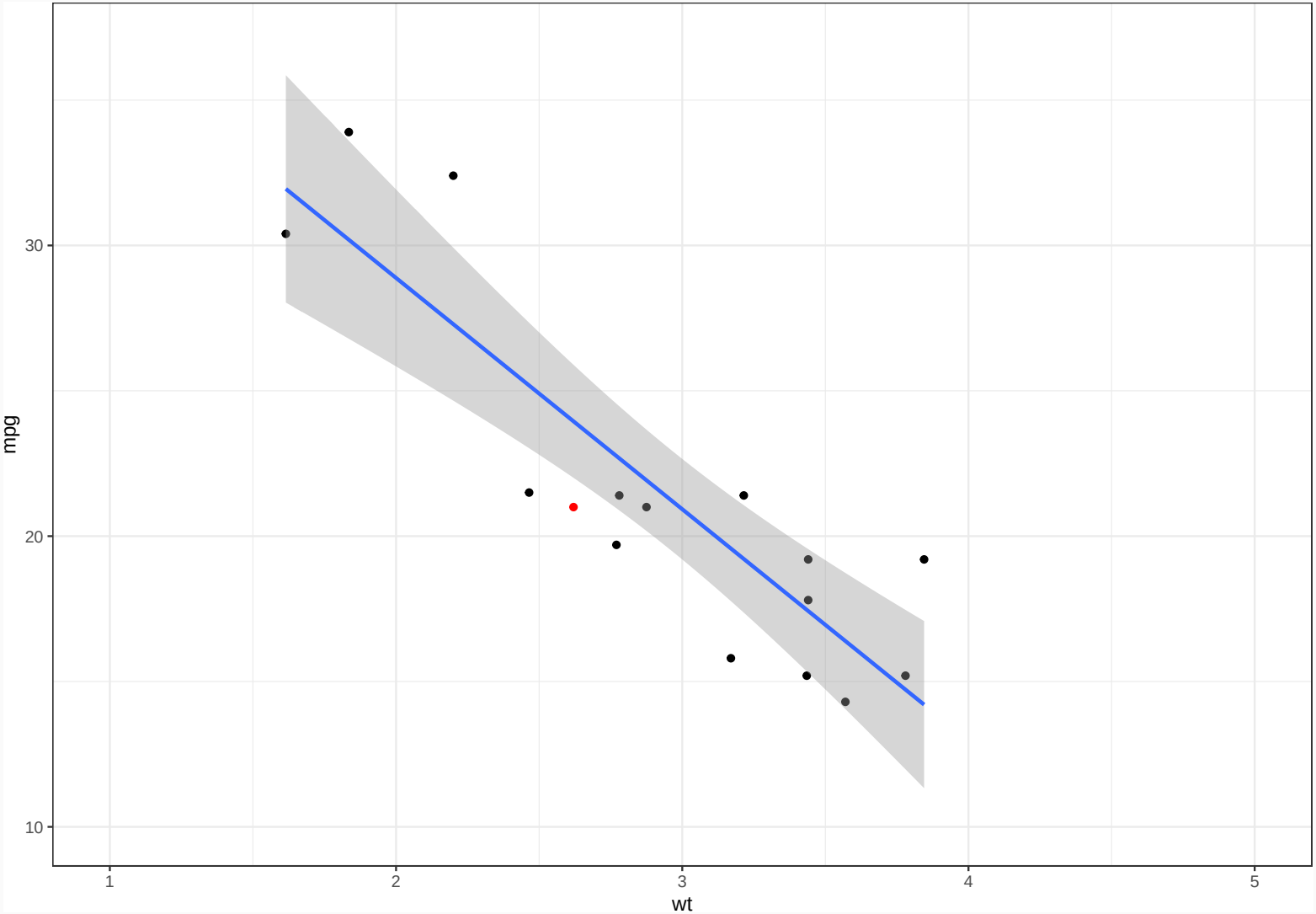
Leave-one-out cross validation



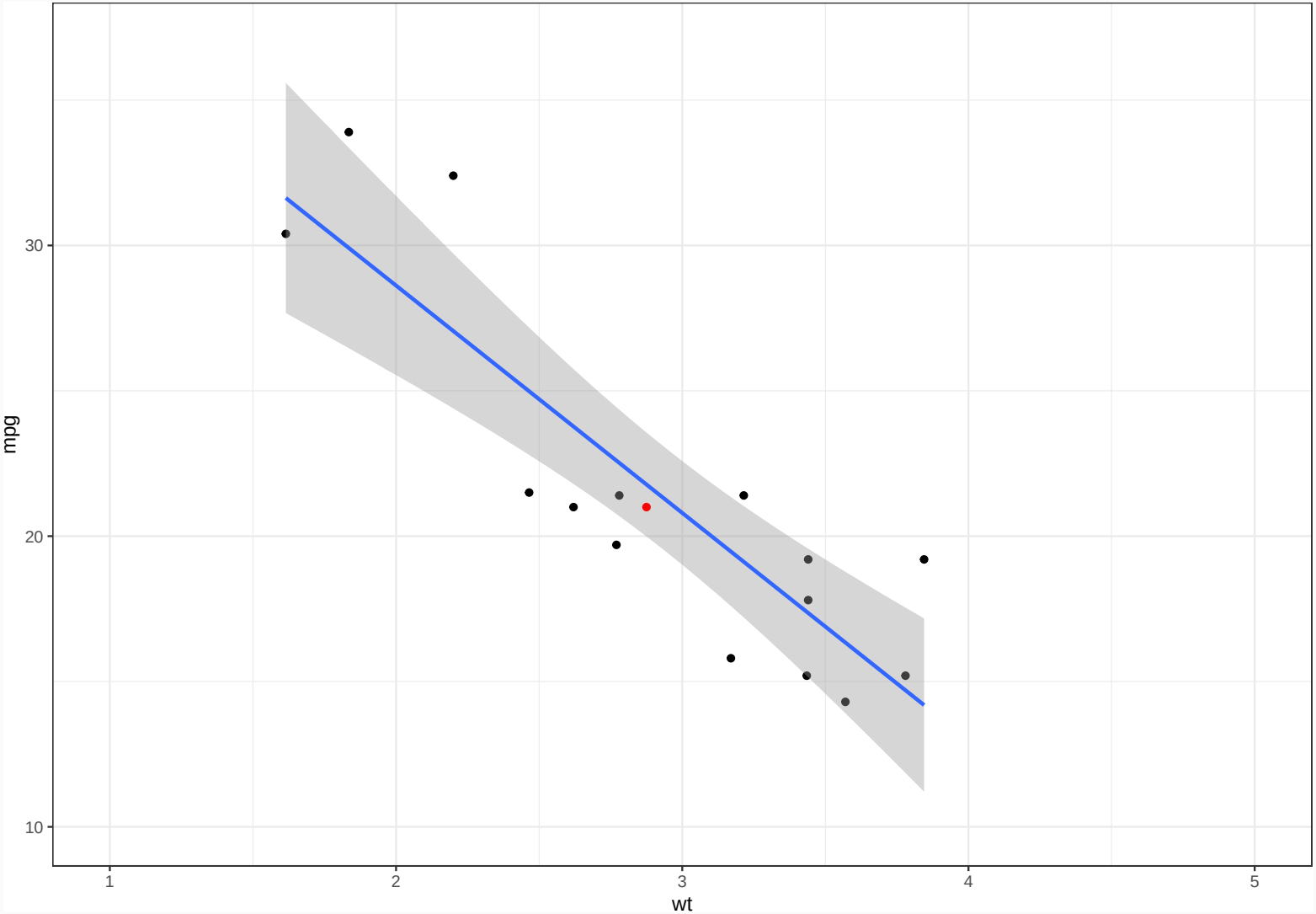
Leave-one-out cross validation



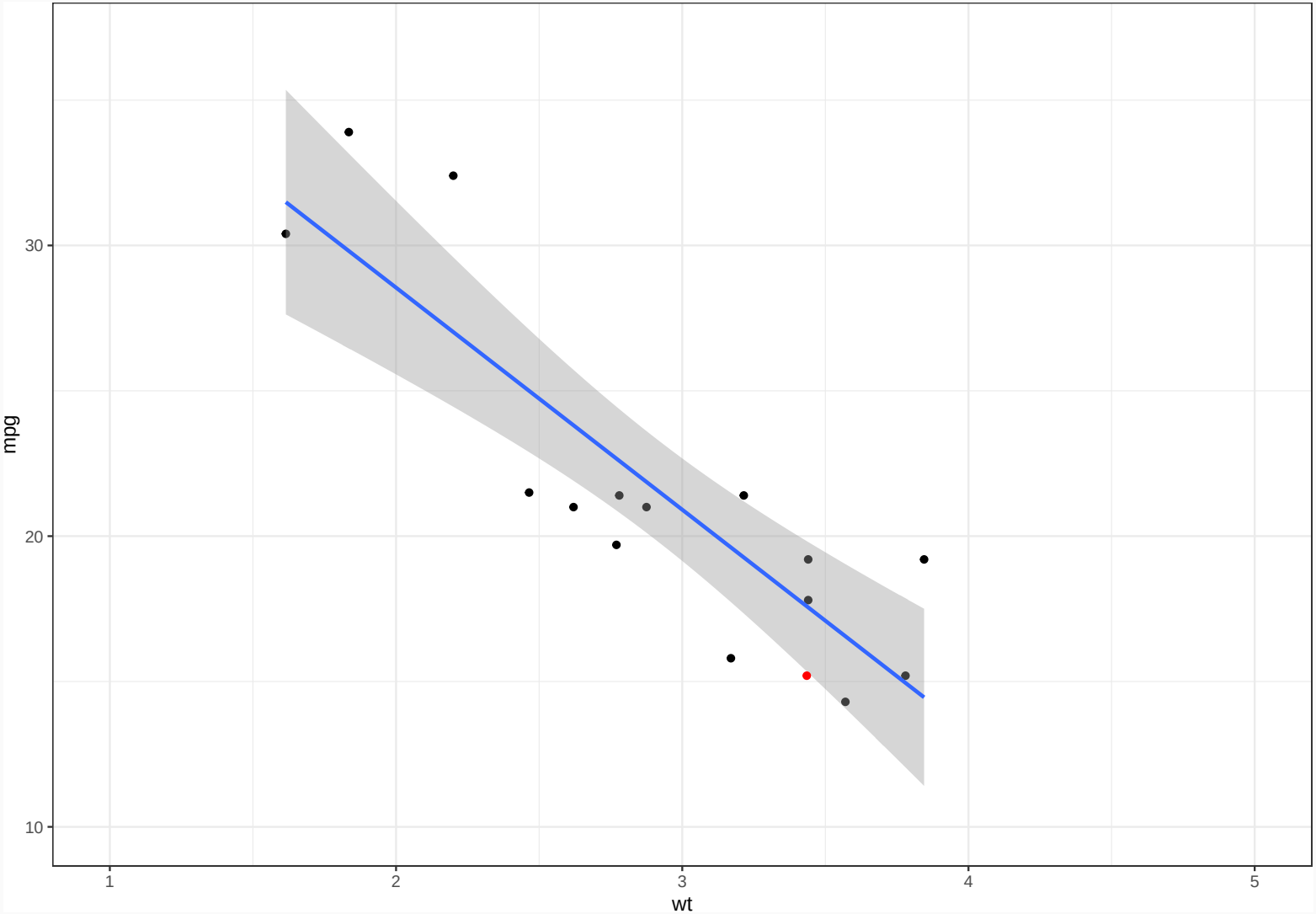
Leave-one-out cross validation



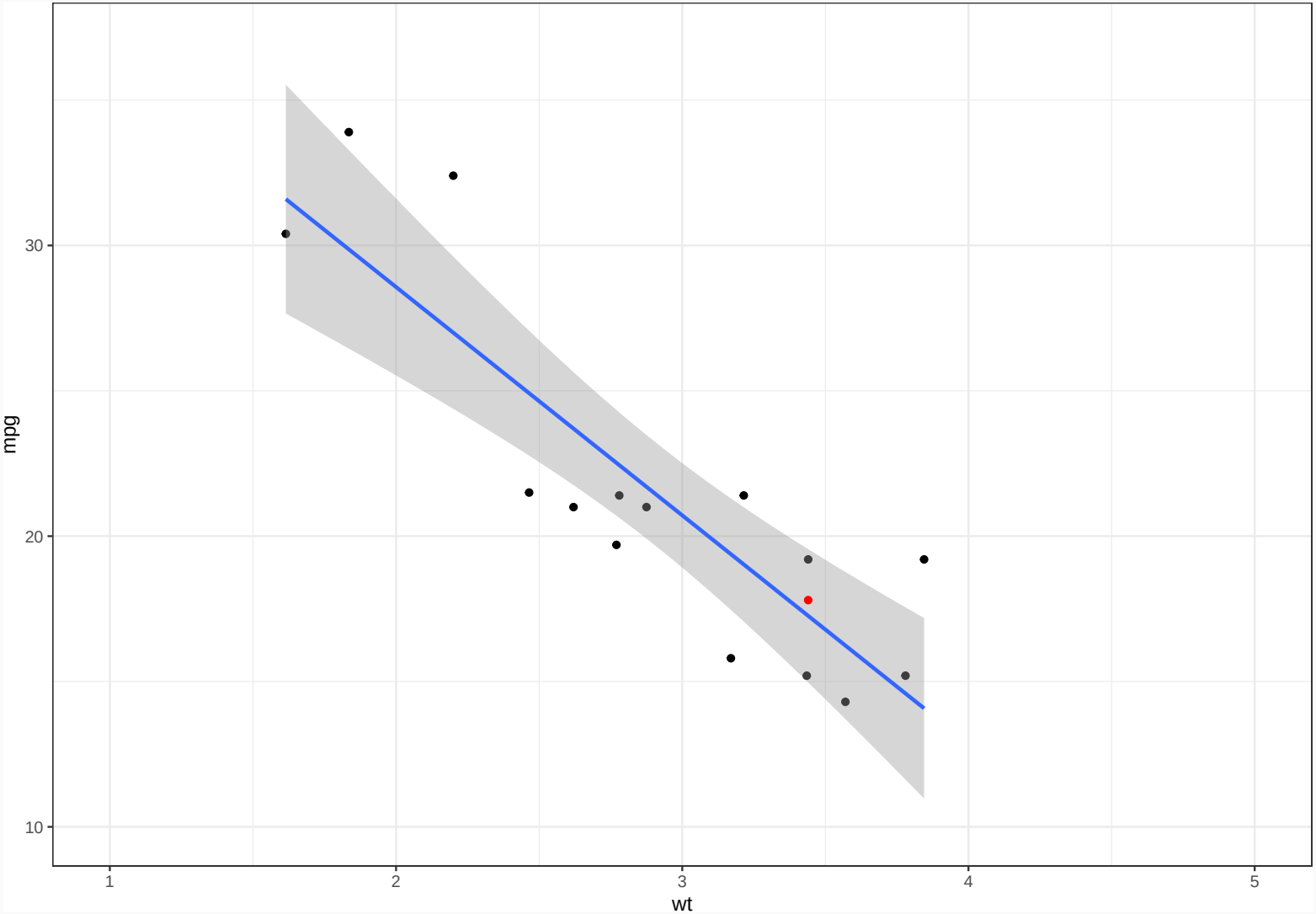
Leave-one-out cross validation



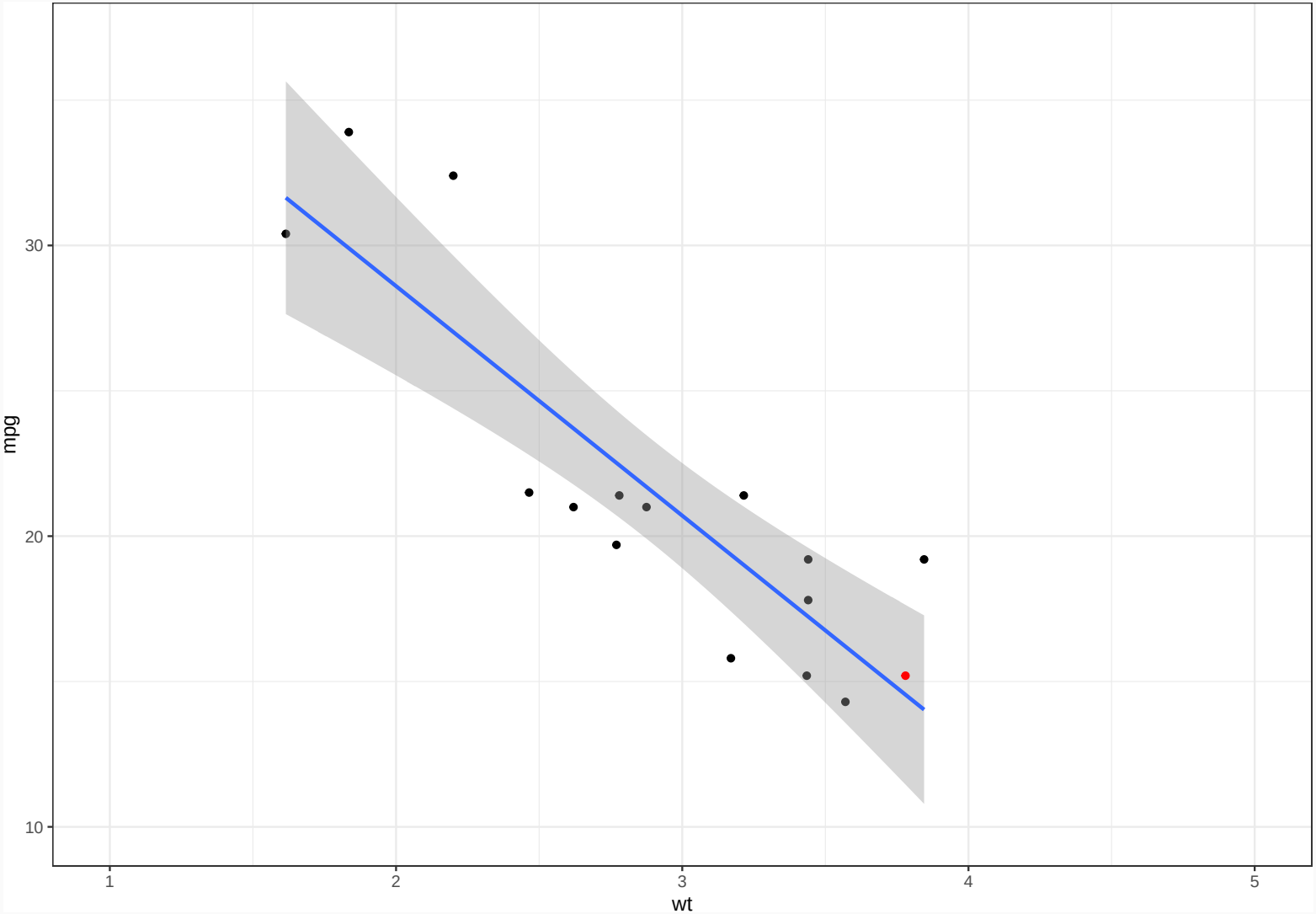
Leave-one-out cross validation



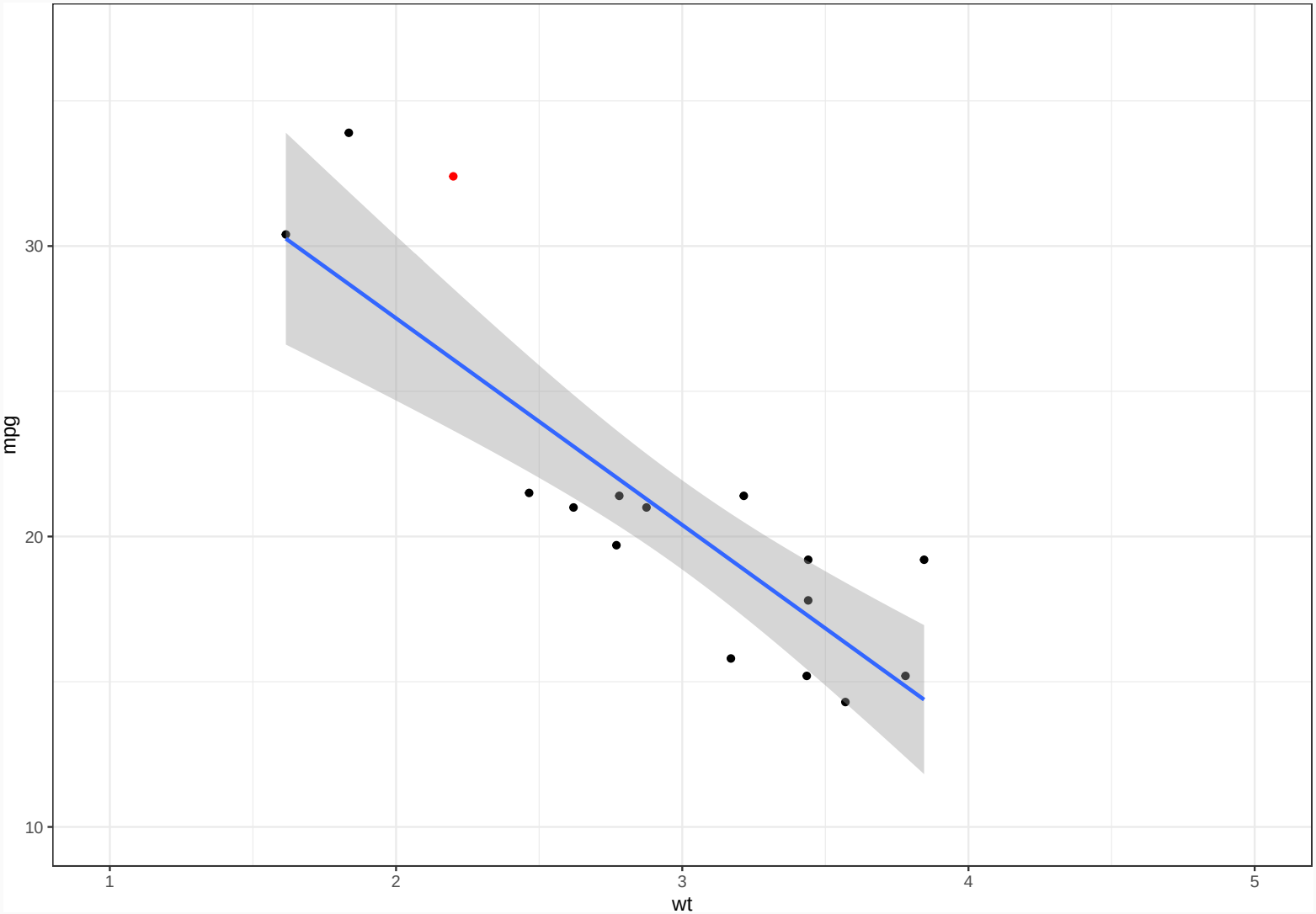
Leave-one-out cross validation



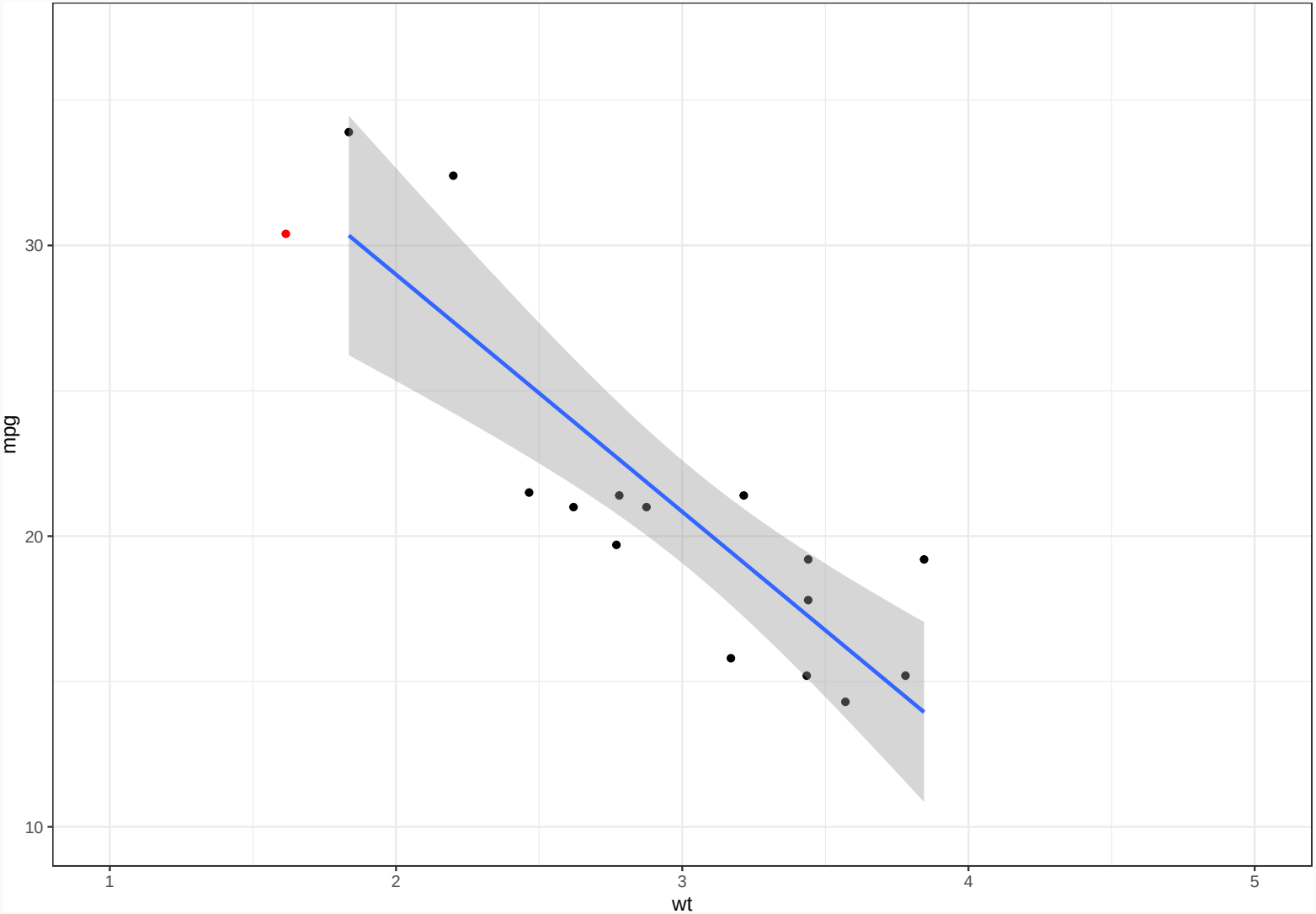
Leave-one-out cross validation



Leave-one-out cross validation



Leave-one-out cross validation



Implenting LOO-CV in R

```
cv_quap(mReg)
```

```
## [1] -119.683
```

PSIS provides an approximation of LOO-CV that is more computationally efficient

```
PSIS(mReg)
```

```
##      PSIS      lppd  penalty  std_err
## 1 240.551 -120.2755 3.640157 14.12346
```

Model comparison using cross validation and information criteria

General approach

This approach is appropriate for comparing the estimated *predictive* accuracy of models. It is not appropriate for causal inference.

1. Estimate competing models
2. Compute the indicator of interest
3. Evaluate relative model predictive performance

Return to the plant fungus experiment

From the prior lecture and chapter 6: we conduct an experiment on the effect of an antifungal agent on plant growth. Conditioning on the presence of fungus confounds the relationship between the treatment and plant growth.

- fungus_0 is an intercept only model
- fungus_1 includes the treatment as a predictor
- fungus_2 includes both the treatment and fungus as a predictor

Compare these models using an information criterion

Information criteria compute an expected out-of-sample predictive score. AIC and BIC are routinely used, but better approaches now exist. We'll use WAIC (widely applicable information criteria).

In general, information criteria have two components:

$$IC = \log \text{ probability of model} - \text{penalty term}$$

The penalty term (crudely) penalizes models for additional complexity. For two models with the same fit (log probability), the IC will prefer the less complex model

Comparing model fits with WAIC

```
compare(fungus_0, fungus_1, fungus_2)
```

```
##              WAIC       SE    dWAIC      dSE    pWAIC       weig
## fungus_2 370.4278 15.72564  0.00000       NA 3.908673 1.000000e-
## fungus_0 420.6603 12.79034 50.23254 11.65051 2.765443 1.236350e-
## fungus_1 421.0453 12.74626 50.61750 11.60759 2.933149 1.019883e-
```

- fungus_2 has the lowest WAIC score. This means that it is expected to predict new cases with the highest accuracy.
- The difference in WAIC between fungus_2 and fungus_1 is large and meaningful
- Note the dSE (difference standard error) value relative to dWAIC (the difference in WAIC between the lowest scoring model and all others)

Comparing model fits with WAIC

```
compare(fungus_0, fungus_1, fungus_2)
```

```
##              WAIC       SE    dWAIC      dSE    pWAIC       weig
## fungus_2 370.4278 15.72564  0.00000       NA 3.908673 1.000000e-
## fungus_0 420.6603 12.79034 50.23254 11.65051 2.765443 1.236350e-
## fungus_1 421.0453 12.74626 50.61750 11.60759 2.933149 1.019883e-
```

- fungus_2 has the lowest WAIC score. This means that it is expected to predict new cases with the highest accuracy.
- The difference in WAIC between fungus_2 and fungus_1 is large and meaningful
- Note the dSE (difference standard error) value relative to dWAIC (the difference in WAIC between the lowest scoring model and all others)
- Information criteria don't care about causation!

Comparing model fits with PSIS

PSIS is an approximation of the LOO-CV method. It will warn you when there are influential observations that may bias the PSIS estimate. See 7.5.2 for a detailed example of how to address this using *robust regression*.

```
compare(fungus_0, fungus_1, fungus_2, func = PSIS)
```

```
##              PSIS       SE    dPSIS      dSE    pPSIS       wei
## fungus_2 370.2318 15.82934  0.00000       NA 3.787902 1.000000e-
## fungus_0 420.8314 12.89778 50.59965 11.61946 2.840778 1.029024e-
## fungus_1 421.0323 12.89876 50.80055 11.60069 2.933350 9.306789e-
```

Interactions

Motivation for interactions

- A linear model assumes that predictors are independent
- It tells us about the expected relationship between X and Y, once we know Z

Motivation for interactions

- A linear model assumes that predictors are independent
- It tells us about the expected relationship between X and Y, once we know Z
- What if we think the relationship between X and Y varies based on Z?

Relationships between bad geography and a country's economic performance

- Countries with smoother terrain tend to have stronger economies.
- However, economies in Africa appear to benefit from rough terrain.
- This may be due to the protections that rugged terrain offered against the slave trade.

The data

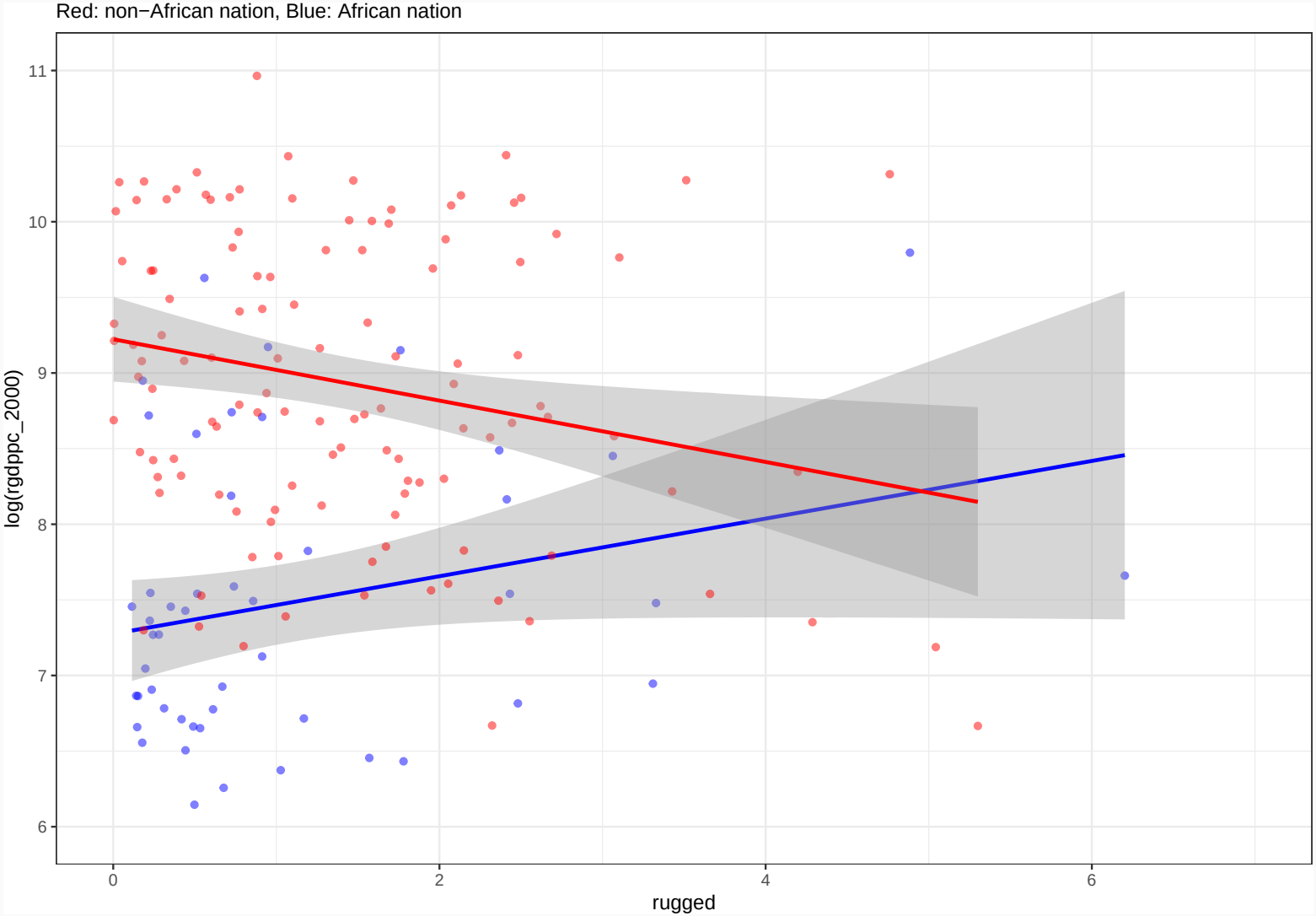
This data measures (among other things)

- Terrain ruggedness (variance in topography)
- A country's gross domestic product per capita (the ratio of economic productivity to population) in 2000
- Continent

```
data(rugged)
glimpse(rugged %>% select(country, rugged, cont_africa, rgdppc_2000))
```

```
## Observations: 234
## Variables: 4
## $ country     <fct> Aruba, Afghanistan, Angola, Anguilla, Albania, Andorra,...
## $ rugged      <dbl> 0.462, 2.518, 0.858, 0.013, 3.427, 5.717, 0.255, 0.769,...
## $ cont_africa <int> 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1...
## $ rgdppc_2000 <dbl> NA, NA, 1794.729, NA, 3703.113, NA, NA, 20604.460, 1217...
```

The relationships inside and outside Africa: separate regressions



Set up the data for analysis

```
d<-rugged %>%
  mutate(log_gdp = log(rgdppc_2000),
         log_gdp.s = scale(log_gdp),
         rugged.s =scale(rugged)) %>%
  filter(complete.cases(log_gdp))

m8.1<-quap(alist(
  log_gdp.s~dnorm(mu,sigma),
  mu<-a+b*rugged.s,
  a~dnorm(0, 0.1),
  b~dnorm(0,0.3),
  sigma~dexp(1)),
  data=d)

precis(m8.1)
```

```
##                mean         sd        5.5%      94.5%
## a      5.932893e-05 0.06065618 -0.09688096 0.09699962
## b      3.048652e-03 0.08589175 -0.13422296 0.14032026
## sigma 9.941477e-01 0.05368005  0.90835664 1.07993882
```

What if we add intercepts for African / non-African nations?

```
d<-d %>%
  mutate(cont_africa = cont_africa + 1)

m8.2<-quap(alist(
  log_gdp.s~dnorm(mu,sigma),
  mu<-a[cont_africa]+b*rugged.s,
  a[cont_africa]~dnorm(0, 0.1),
  b~dnorm(0,0.3),
  sigma~dexp(1)),
  data=d)

precis(m8.2, depth = 2)
```

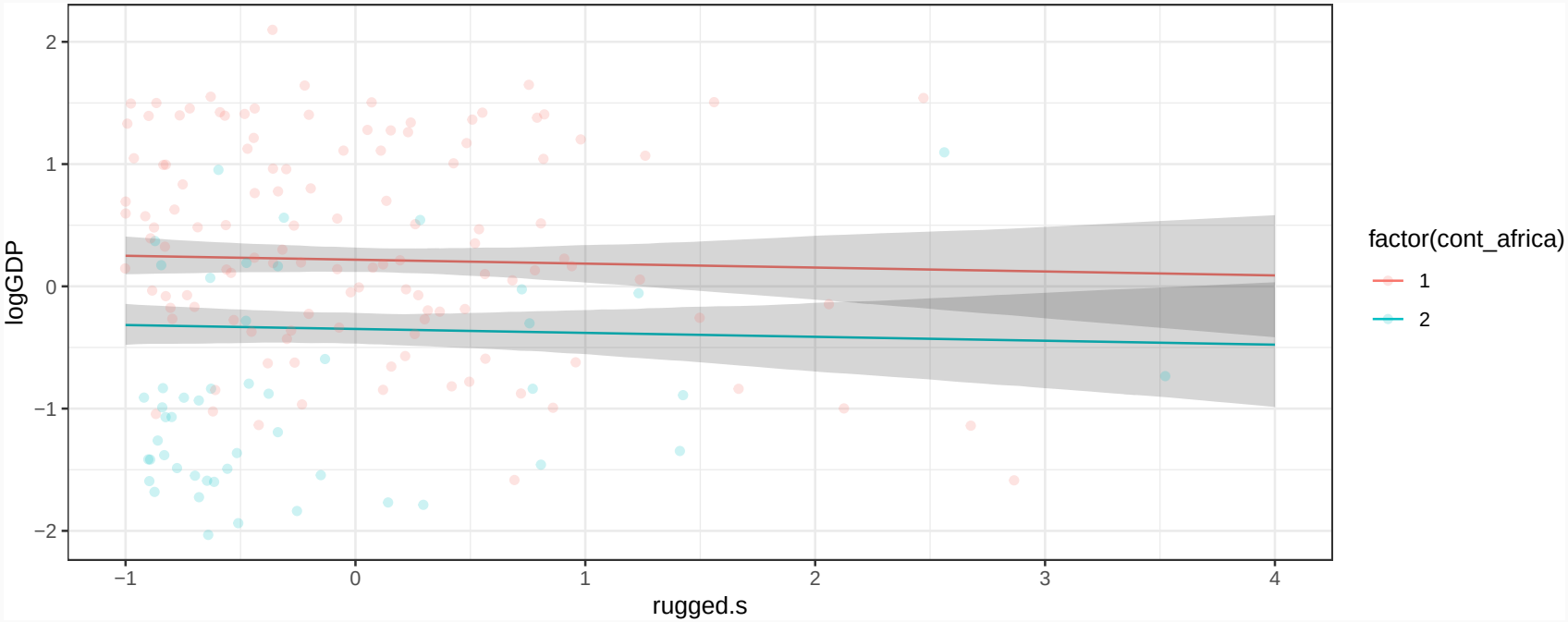
```
##              mean         sd       5.5%       94.5%
## a[1]   0.21919572 0.06315934  0.1182549  0.32013655
## a[2]  -0.34560910 0.08208710 -0.4768001 -0.21441807
## b     -0.02893965 0.07677022 -0.1516333  0.09375398
```

What do model expectations look like?

```
## generate counterfactual data for sim
plot_dat<-data.frame(rugged.s =
                       rep(seq(-1, 4, length.out = 100), 2),
                     cont_africa = rep(c(1, 2), each = 100))
sims<-link(m8.2, plot_dat) # draw posterior samples
sims_mu<-apply(sims, 2, mean) # calculate mean mu
sims_pi<-apply(sims, 2, PI) # calculate 89% PI
plot_dat<-plot_dat %>% # attach for plotting
  mutate(logGDP = sims_mu,
         logGDP_lwr = sims_pi[1,],
         logGDP_upr = sims_pi[2,])
```

Plot model expectation μ

```
ggplot(plot_dat, aes(x = rugged.s, y = logGDP,
                     group = cont_africa)) +
  geom_line(aes(color = factor(cont_africa))) +
  geom_ribbon(alpha = 0.2, aes(ymin = logGDP_lwr, ymax = logGDP_upr)) +
  geom_point(data = d, aes(x = rugged.s, y = log_gdp.s, color = factor(cont_africa)), alpha = 0.2)
```



Model specification

The linear component of our Africa / not - Africa intercept model is:

$$\mu = \alpha_{\text{Africa}[i]} + \beta r_i$$

Model specification

The linear component of our Africa / not - Africa intercept model is:

$$\mu = \alpha_{\text{Africa}[i]} + \beta r_i$$

This specification estimates one intercept for African countries, and one for non-African countries. Slopes are identical for all countries.

Model specification: adding an interaction

Adding a slope for each group is easy:

$$\mu = \alpha_{\text{Africa}[i]} + \beta_{\text{Africa}[i]} r_i$$

Model specification: adding an interaction

Adding a slope for each group is easy:

$$\mu = \alpha_{\text{Africa}[i]} + \beta_{\text{Africa}[i]} r_i$$

This is the interaction: the slope of ruggedness now depends on continent.

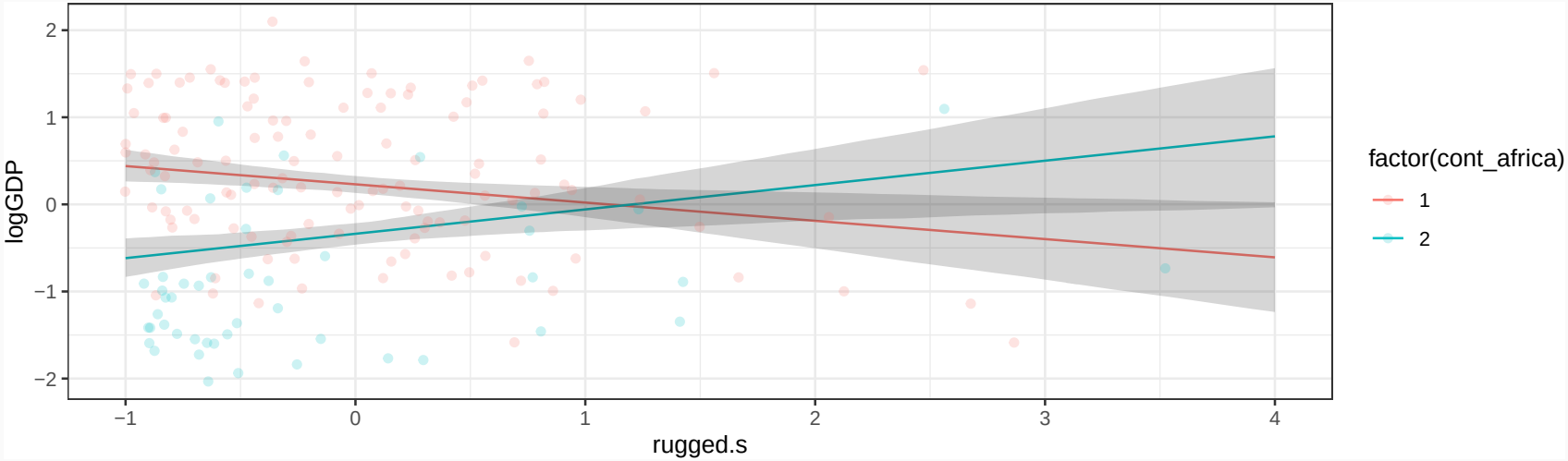
Estimating the interaction model

```
m8.3<-quap(alist(
  log_gdp.s~dnorm(mu,sigma),
  mu<-a[cont_africa]+b[cont_africa]*rugged.s,
  a[cont_africa]~dnorm(0, 0.1),
  b[cont_africa]~dnorm(0,0.3),
  sigma~dexp(1)),
  data=d)

compare(m8.1, m8.2, m8.3)
```

```
##           WAIC       SE     dWAIC      dSE    pWAIC        weight
## m8.3 435.7359 13.88377  0.000000       NA 4.294965 9.906037e-01
## m8.2 445.0520 12.98362  9.316006 7.314588 2.911406 9.396258e-03
## m8.1 486.4011 13.37420 50.665135 8.982639 2.305062 9.865193e-12
```

Visualizing the interaction

```
sims<-link(m8.3, plot_dat) # draw posterior samples
sims_mu<-apply(sims, 2, mean) # calculate mean mu
sims_pi<-apply(sims, 2, PI) # calculate 89% PI
plot_dat<-plot_dat %>% # attach for plotting
  mutate(logGDP = sims_mu,
         logGDP_lwr = sims_pi[1,],
         logGDP_upr = sims_pi[2,])

ggplot(plot_dat, aes(x = rugged.s, y = logGDP,
                     group = cont_africa)) +
  geom_line(aes(color = factor(cont_africa))) +
  geom_ribbon(alpha = 0.2, aes(ymin = logGDP_lwr, ymax = logGDP_upr)) +
  geom_point(data = d, aes(x = rugged.s, y = log_gdp.s, color = factor(cont_africa)), alpha = 0.2)
```



Continuous interactions

- We just estimated an interaction between a *categorical* variable and a *continuous variable*
- This results in *k* slopes, where *k* is the number of categories in the categorical variable
- If we interact a continuous varible with another continuous variable we estimate an infinite number of slopes (!).
- Directly interpreting posterior parameter estimates is very difficult

The example data

- Plant growth depends on light and water
- If we've got light and no water, no growth
- If we've got water and no light, no growth
- How do tulip blooms vary as a function of shade and water?

```
data(tulips)
head(tulips)
```

```
##   bed water shade blooms
## 1   a     1     1   0.00
## 2   a     1     2   0.00
## 3   a     1     3 111.04
## 4   a     2     1 183.47
## 5   a     2     2  59.16
## 6   a     2     3  76.75
```

Fitting a model without interactions

Our linear function for this model will be:

$$\mu = \alpha + \beta_W W_i + \beta_S S_i$$

```
d<-tulips %>%
  mutate(blooms.s = blooms / max(blooms),
         water.s = water - 2,
         shade.s = shade - 2)

tulip0<-quap(alist(
  blooms.s ~ dnorm(mu, sigma),
  mu <- a + bw * water.s + bs * shade.s,
  a ~ dnorm(0.5, 0.25),
  bw ~ dnorm(0, 0.25),
  bs ~ dnorm(0, 0.25),
  sigma ~ dexp(1)
), data = d)
```

Add the interaction

Interactions allow for the relationship between B and W to depend on S (equivalently B and S depends on W).

$$\mu = \alpha + \beta_w W_i + \beta_s S_i + \beta_{sw} W_i S_i$$

```
tulip1<-quap(alist(
  blooms.s ~ dnorm(mu, sigma),
  mu <- a + bw * water.s + bs * shade.s + bws * water.s * shade.s,
  a ~ dnorm(0.5, 0.25),
  bw ~ dnorm(0, 0.25),
  bs ~ dnorm(0, 0.25),
  bws ~ dnorm(0, 0.25),
  sigma ~ dexp(1)
), data = d)
```

Look at the posterior

```
compare(tulip0, tulip1)
```

```
##             WAIC        SE    dWAIC      dSE    pWAIC      weight
## tulip1 -22.43088 10.398908  0.00000       NA 6.453630 0.994747659
## tulip0 -11.94325  9.065879 10.48763 8.134821 5.454518 0.005252341
```

```
precis(tulip1)
```

```
##             mean         sd       5.5%       94.5%
## a      0.3579828 0.02391897  0.3197556  0.39620990
## bw     0.2067291 0.02923472  0.1600064  0.25345186
## bs    -0.1134613 0.02922769 -0.1601728 -0.06674980
## bws   -0.1431612 0.03567987 -0.2001846 -0.08613791
## sigma  0.1248457 0.01694070  0.0977712  0.15192022
```

- α is the intercept, same as usual
- β_W is now the slope for water when shade is zero
- β_S is the slope for shade when water is zero
- β_{WS} moves the slope by $W \times S$ when neither is zero

This is very difficult to interpret... Don't try! Let's visualize instead.

The plan for a visual

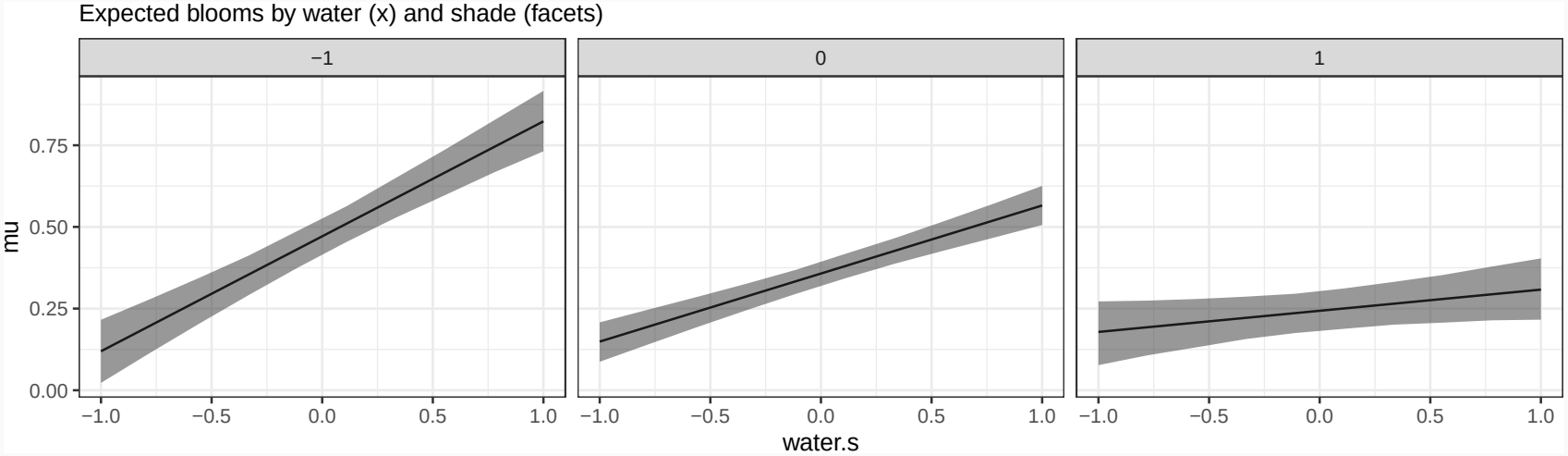
- Let's visualize the counterfactual relationship between water and blooms for a range of values of shade.
- This means multiple plots, partitioning the continuous into ordered categories
- We want a full range of water along a range of values of shade
- If we want to see k values of water and j values of shade, we'll have $k \times j$ scenarios in our counterfactual data

```
plot_dat<-data.frame(water.s = rep(seq(-1, 1, length.out = 10),3),
                     shade.s = rep(c(-1, 0, 1), each = 10))

head(plot_dat)
```

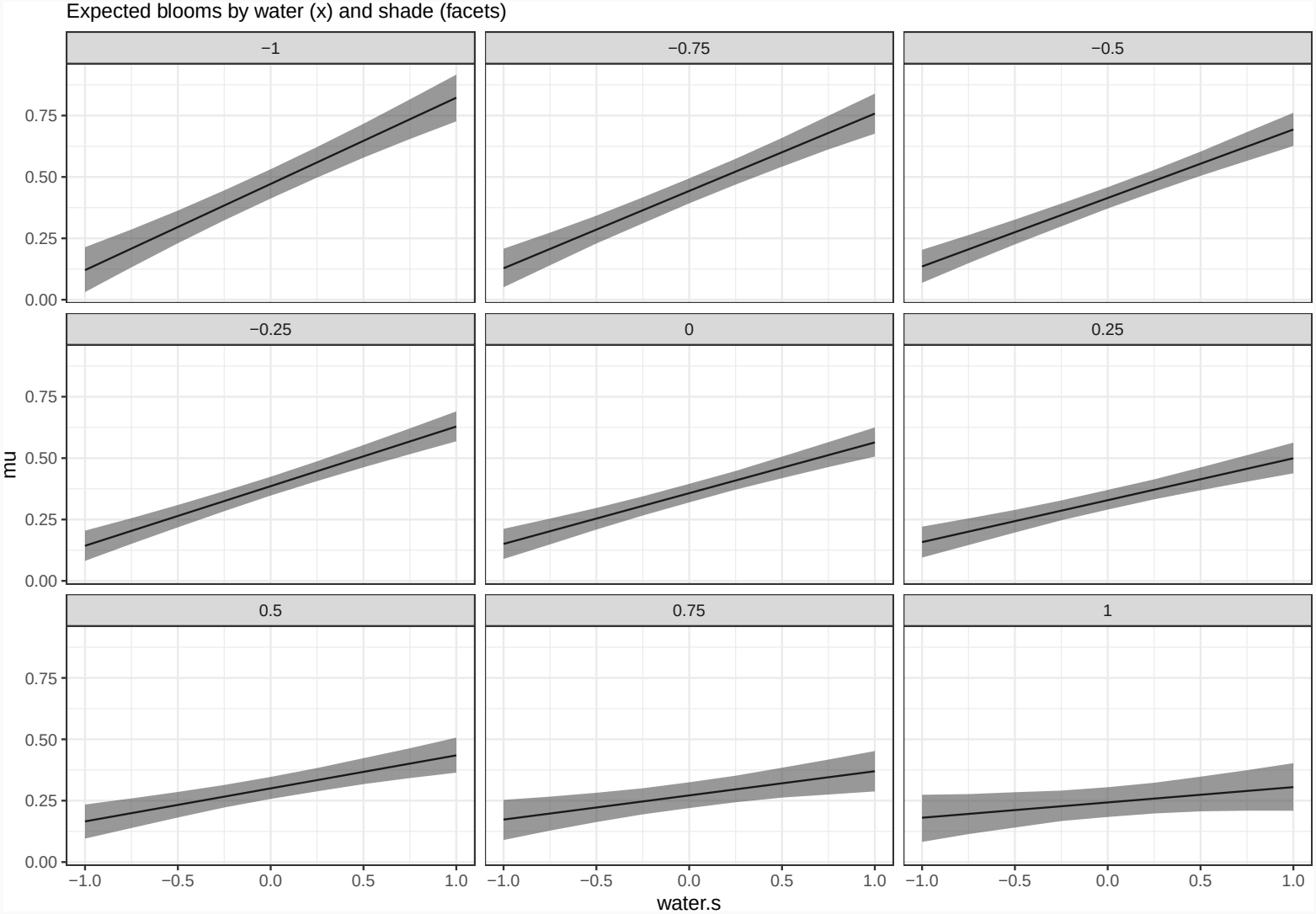
```
##      water.s shade.s
## 1 -1.0000000      -1
## 2 -0.7777778      -1
## 3 -0.5555556      -1
## 4 -0.3333333      -1
## 5 -0.1111111      -1
## 6  0.1111111      -1
```

Plotting it

```
mus<-link(tulip1, plot_dat)
mu_mn<-apply(mus, 2, mean)
mu_pi<-apply(mus, 2, PI)
plot_dat<-plot_dat %>%
  mutate(mu = mu_mn,
         mu_lwr = mu_pi[1,],
         mu_upr = mu_pi[2,])

ggplot(plot_dat, aes(x = water.s, y = mu,
                     ymin = mu_lwr, ymax = mu_upr)) +
  geom_line() +
  geom_ribbon(alpha = 0.5) +
  facet_wrap(~shade.s)+
  labs(subtitle = "Expected blooms by water (x) and shade (facets)")
```



Pushing this further



Wrapping up

- Homework: Complete problem 8H4 from the book (Chapter 8 Practice)
- We'll review this material and discuss in lab on Tuesday