# MA678 Homework 2

**9/10/2020**

# 11.5

Residuals and predictions: The folder Pyth contains outcome y and predictors x1, x2 for 40 data points, with a further 20 points with the predictors but no observed outcome. Save the file to your working directory, then read it into R using read.table().

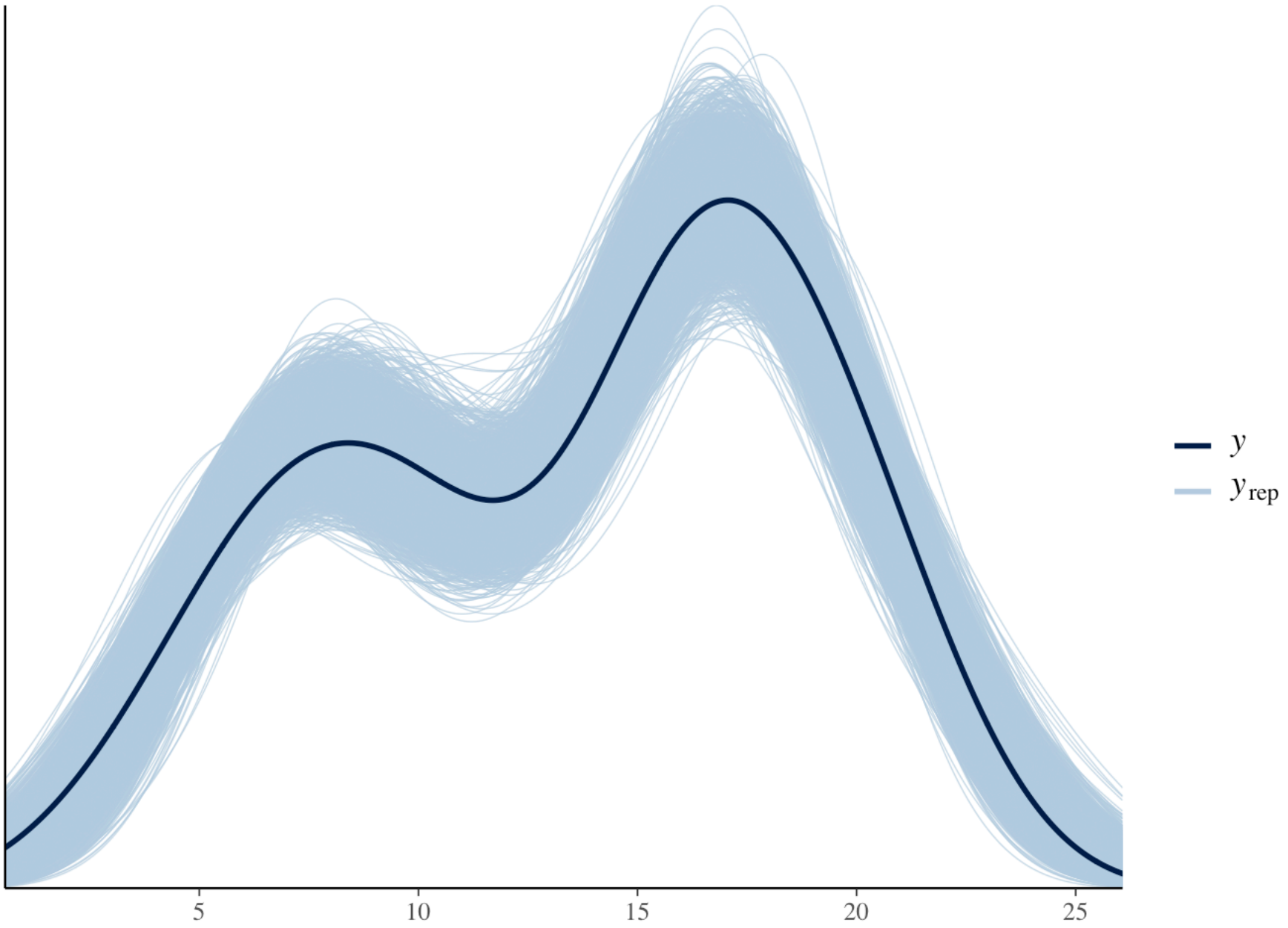
## (a)

Use R to fit a linear regression model predicting y from x1, x2, using the first 40 data points in the file. Summarize the inferences and check the fit of your model.

```
pyth <- read.table("https://raw.githubusercontent.com/avehtari/ROS-Examples/master/Py
th/pyth.txt", header = T)

df1 <- pyth %>% slice(1:40)
df2 <- pyth %>% slice(41:60)
y <- df1$y
x1 <- df1$x1
x2 <- df1$x2
fit <- stan_glm(y ~ x1 + x2, data = df1, refresh=0)
fit
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x1 + x2
##  observations: 40
##  predictors:   3
## ------
##             Median MAD_SD
## (Intercept) 1.3    0.4
## x1          0.5    0.0
## x2          0.8    0.0
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.9    0.1
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
## From this regression output we can see that for the first 40 observations, if x1 a
nd x2 are equal to 0, the average y value is 1.3. Each 1 unit increase in x1 correspo
nds to, on average, a 0.5 unit increase in y. Each 1 unit increase in x2 corresponds
to, on average, a 0.8 unit increase in y.

post_fit <- posterior_predict(fit)
ppc_dens_overlay(y, post_fit)
```




```
#Based on the ppc density overlay plot, it appears as though the model is fitting the
data very well.
```

# (b)

Display the estimated model graphically as in Figure 10.2

```
mean(x2)
```

```
## [1] 11.7815
```

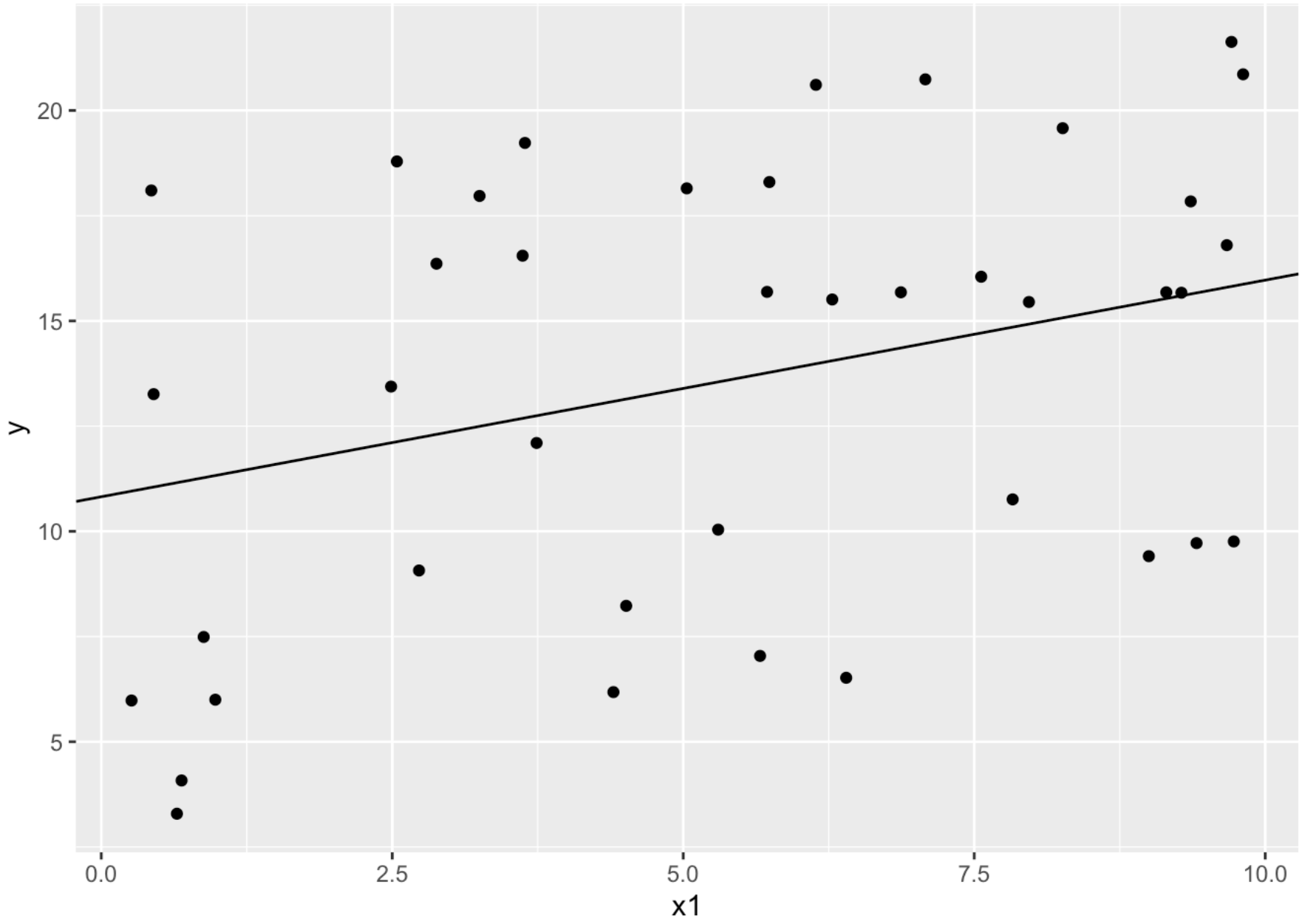
```
mean(x1)
```

```
## [1] 5.3775
```

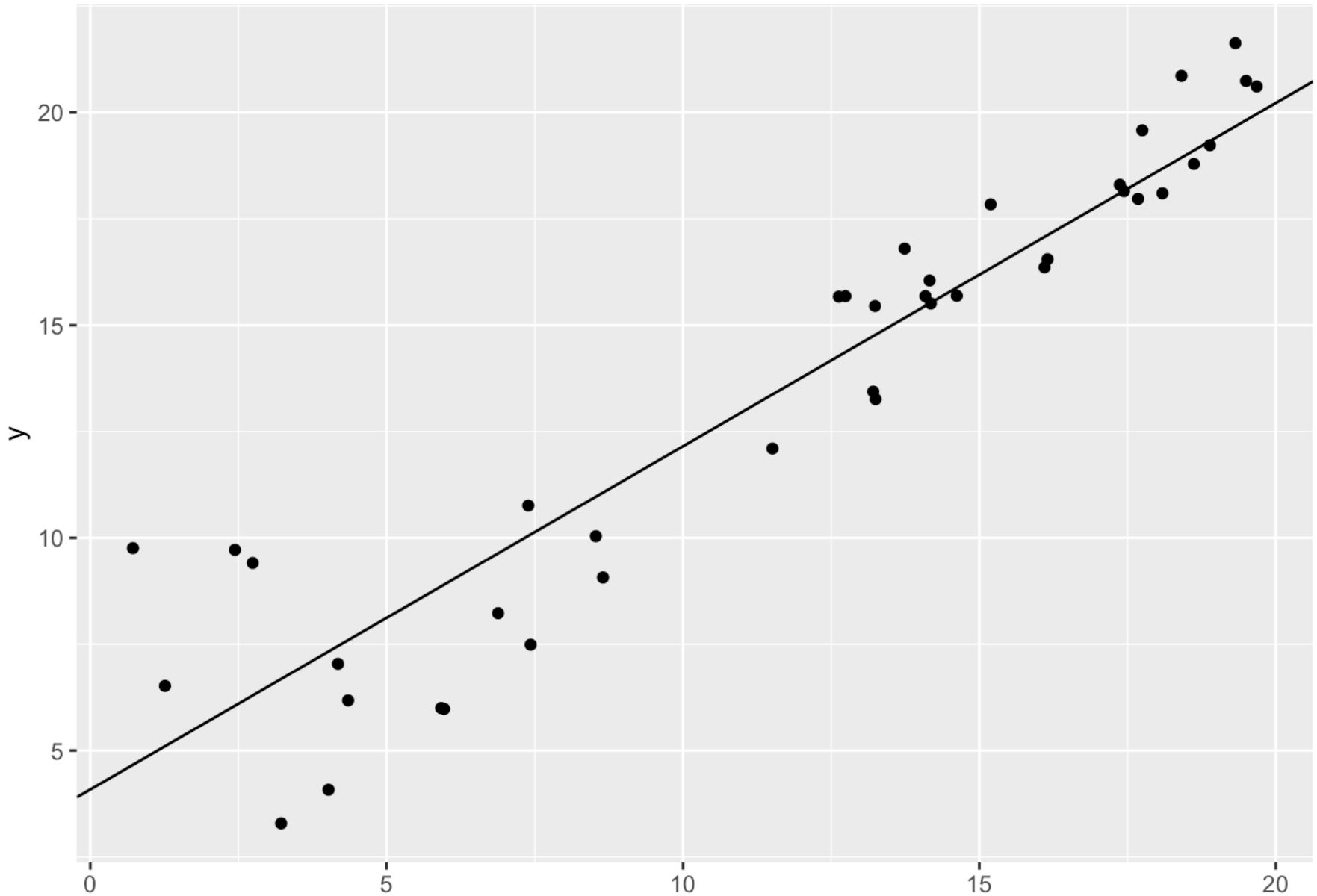
```
fit <- stan_glm(y ~ x1 + x2, data = df1, refresh=0)
fit
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x1 + x2
##  observations: 40
##  predictors:   3
## ------
##             Median MAD_SD
## (Intercept) 1.3    0.4
## x1          0.5    0.0
## x2          0.8    0.0
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.9    0.1
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
ggplot(data = df1, aes(x1, y)) + geom_point() + geom_abline(slope = coef(fit)[2], int
ercept = coef(fit)[1]+coef(fit)[3]*mean(x2))
```




```
##This scatterplot shows the relationship between x1 and y for the first 40 observati
ons, with x2 held constant at its mean.

ggplot(data = df1, aes(x2, y)) + geom_point() + geom_abline(slope = coef(fit)[3], int
ercept = coef(fit)[1]+coef(fit)[2]*mean(x1))
```




```
##This scatterplot shows the relationship between x2 and y for the first 40 observati
ons, with x1 held constant at its mean.


plot_ly(x=x1, y=y, z=x2, type="scatter3d", mode="markers")
```

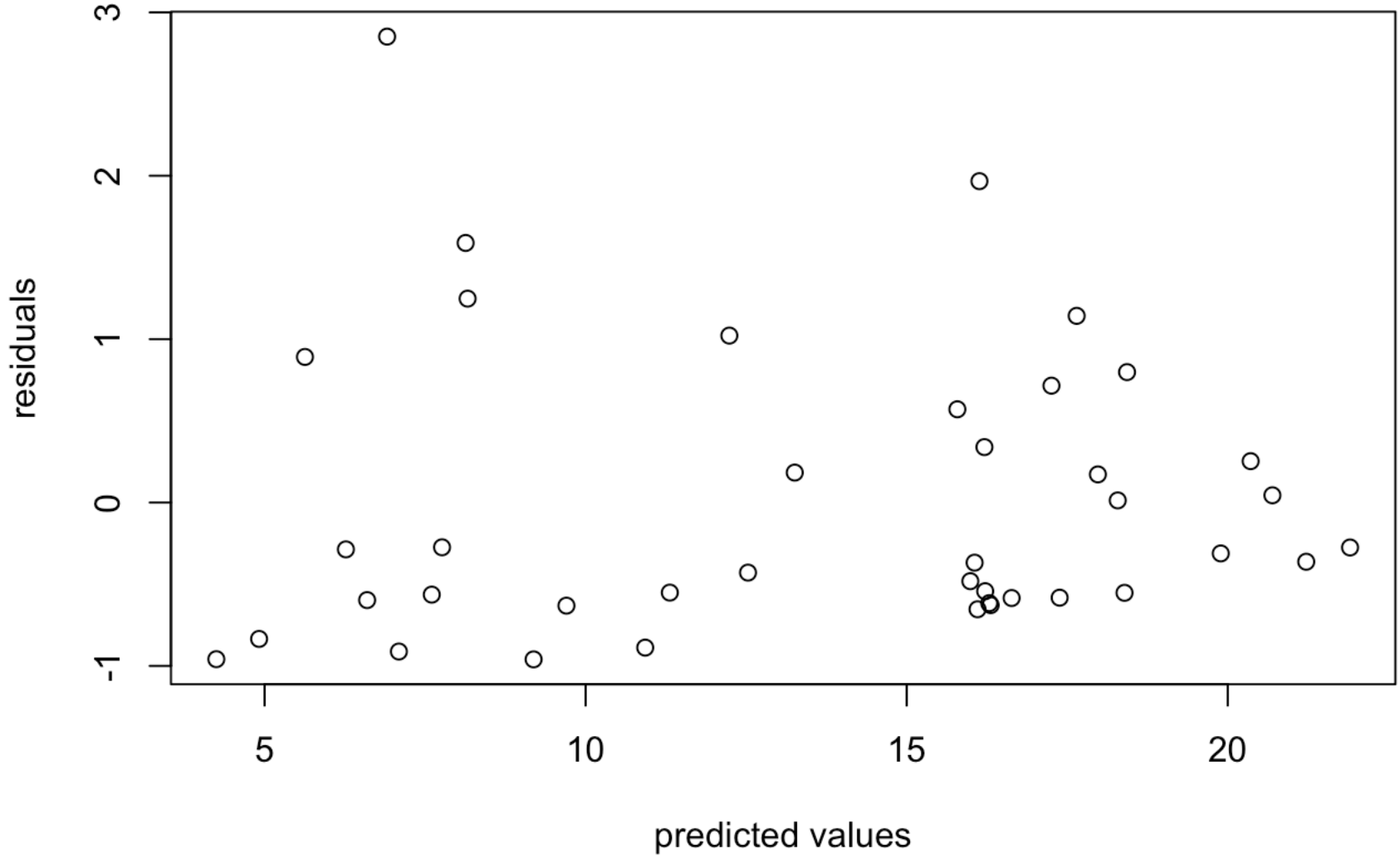
```
## Warning: `arrange_()` is deprecated as of dplyr 0.7.0.
## Please use `arrange()` instead.
## See vignette('programming') for more help
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
##This plot shows a 3D display of the model with x=x1, y=y, and z=x2, each for the fi
rst 40 observations in the dataset.
```

# (c)

Make a residual plot for this model. Do the assumptions appear to be met?

```
predicted <- predict(fit)
resid <- y - predicted

plot(predicted, resid, xlab = "predicted values", ylab = "residuals")
```




```
## The residuals look to be somewhat spread evenly around 0. There isn't a clear patt
ern in the residuals, although it doesn't look perfectly even. I don't really have a
firm grasp on just how evenly distributed the residual plots should be so it's hard t
o say for sure, but I think this looks good enough.
```

# (d)

Make predictions for the remaining 20 data points in the file. How confident do you feel about these predictions?

```
df2 <- pyth %>% slice(41:60)

new_predicted <- predict(fit, newdata = df2)

#I feel fairly confident about these predictions because based on the output above, t
he model appears to be fitting the data well. This is especially apparent when we loo
k at the ppc_dens_overlay plot. Because we don't have any observed data for y for the
last 20 observations, we can't calculate the residuals and actually test our predicti
on, but based on the regression model that was developed above I think the prediction
will be pretty accurate.
```

# 12.5

Logarithmic transformation and regression: Consider the following regression: log(weight)=-3.8+2.1log(height)+error, with errors that have standard deviation 0.25. Weights are in pounds and heights are in inches.

## (a)

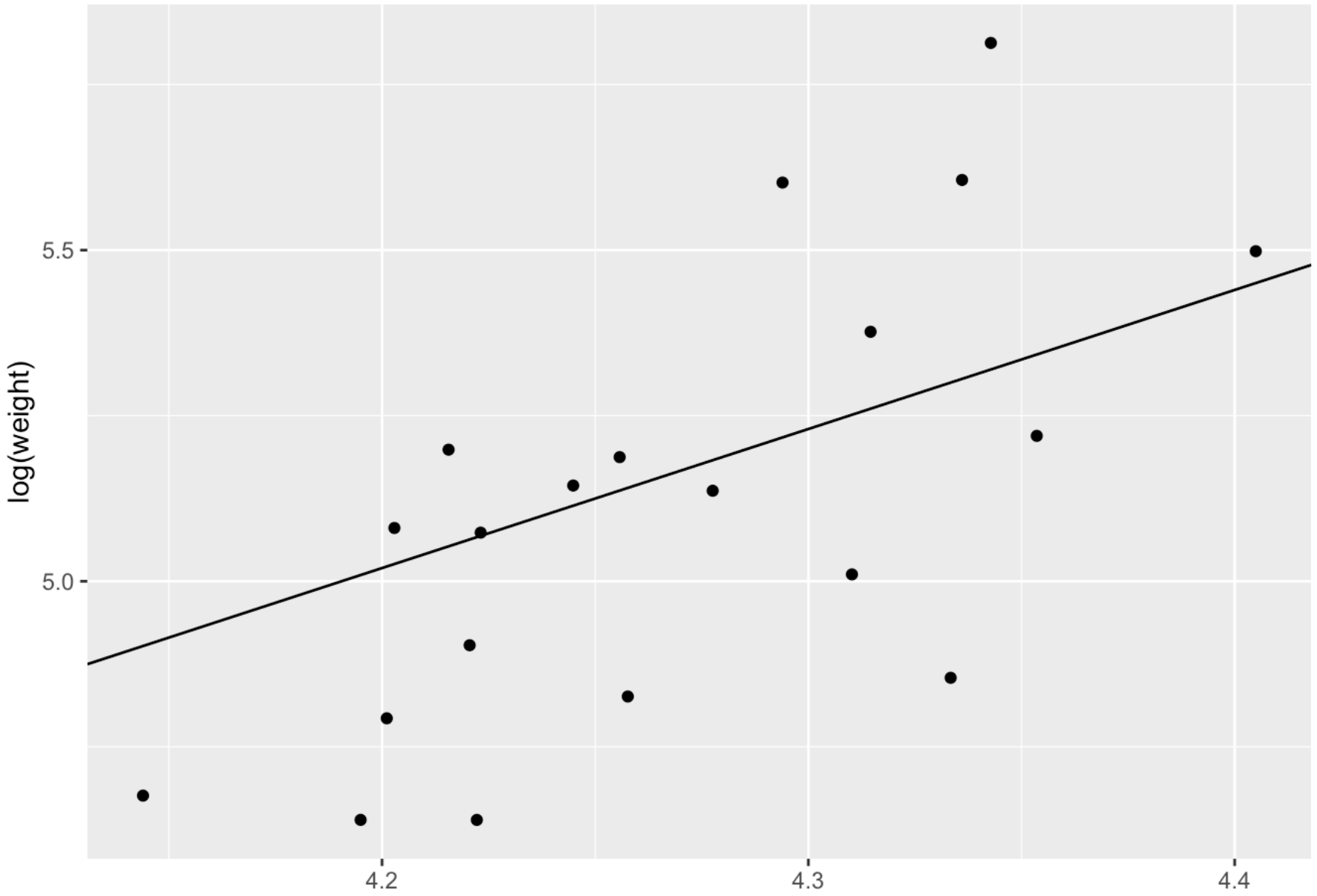
Fill in the blanks: Approximately 68% of the people will have weights within a factor of _____ and _____ of their predicted values from the regression.

```
# 0.779, 1.284

#68% of log(weights) will be within + or - 0.25 of their predicted values. On the ori
ginal scale, this means that 68% of people will have weights between a factor of exp(
-0.25)=0.779 and exp(0.25)=1.284 of their predicted values.
```

## (b)

Using pen and paper, sketch the regression line and scatterplot of log(weight) versus log(height) that make sense and are consistent with the fitted model. Be sure to label the axes of your graph.

```
n <- 20
sigma <- 5
x <- rnorm(n, 70, sigma) #x=height
log_x <- log(x)
log_y <- -3.8 + 2.1 * log_x + rnorm(n, 0, 0.25)
df <- data.frame(log_x, log_y)

ggplot(data = df, aes(log_x, log_y)) + geom_point() + geom_abline(intercept = -3.8, s
lope = 2.1) + labs(x="log(height)", y="log(weight)")
```



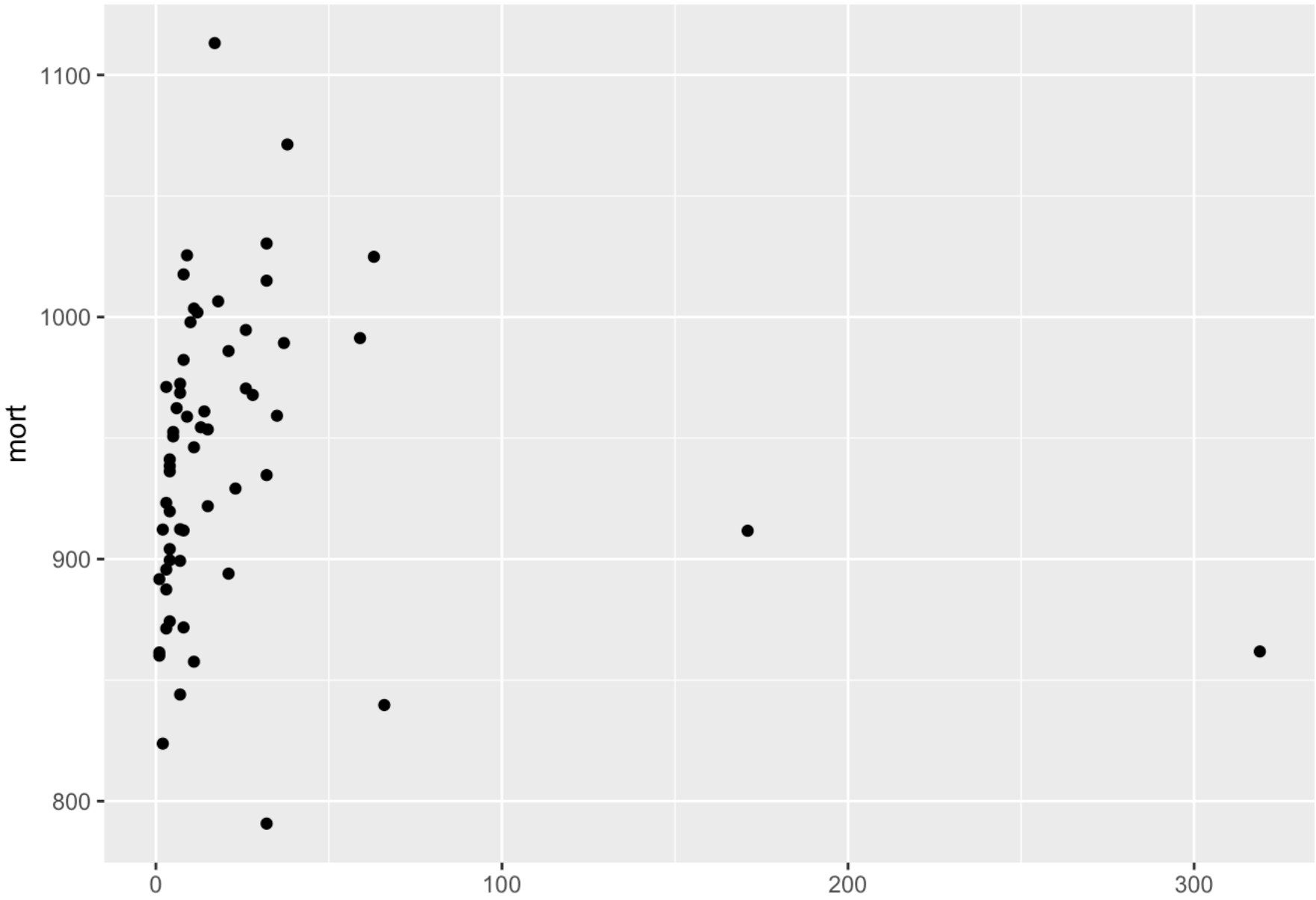

# 12.6

Logarithmic transformations: The folder Pollution contains mortality rates and various environmental factors from 60 US metropolitan areas. For this exercise we shall model mortality rate given nitric oxides, sulfur dioxide, and hydrocarbons as inputs. this model is an extreme oversimplication, as it combines all sources of mortality and does not adjust for crucial factors such as age and smoking. We use it to illustrate log transformation in regression.

## (a)

create a scatterplot of mortality rate versus level of nitric oxides. Do you think linear regression will fit these data well? Fit the regression and evaluate a residual plot from the regression.

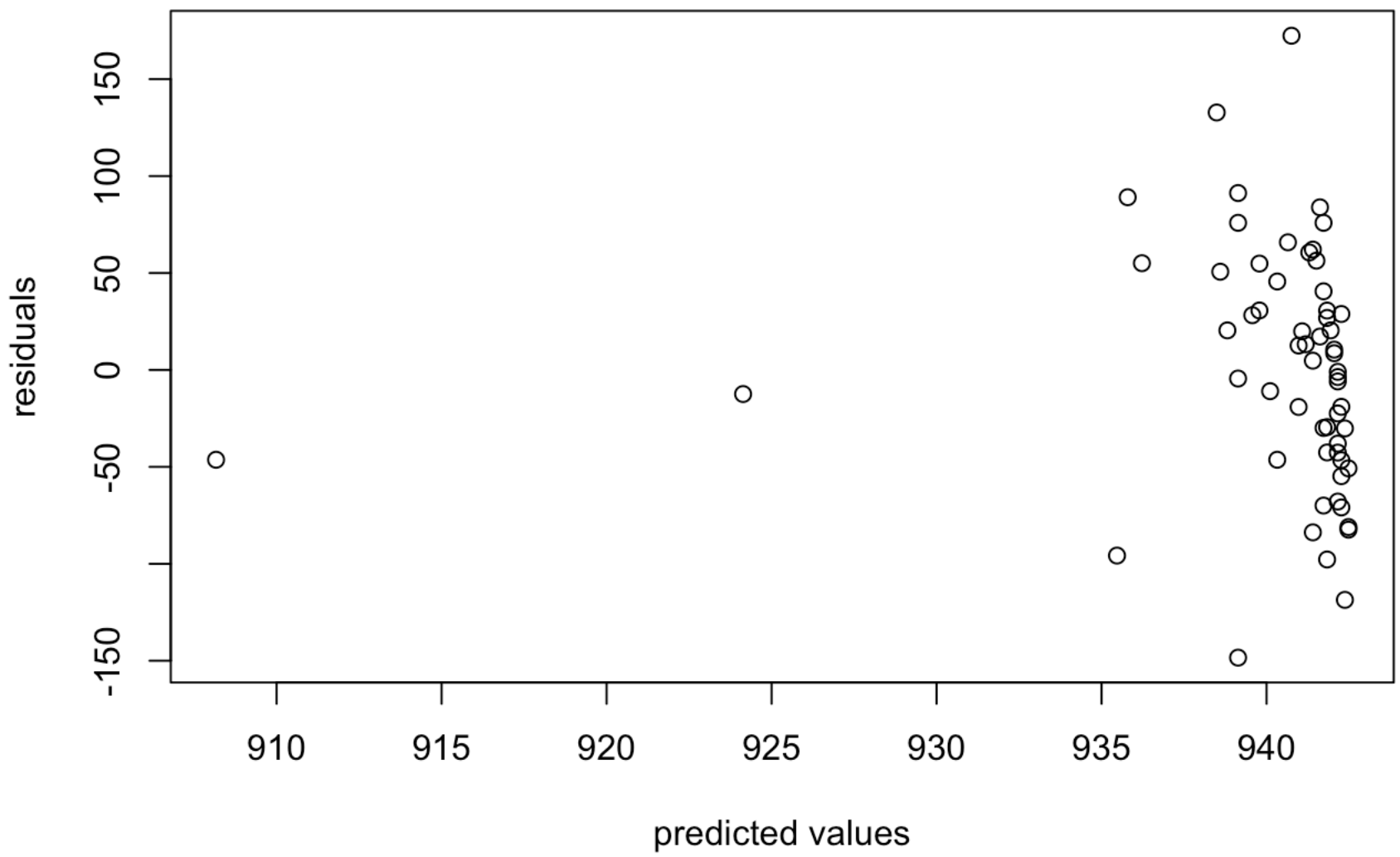
```
pollution <- read.csv("https://raw.githubusercontent.com/avehtari/ROS-Examples/master
/Pollution/data/pollution.csv")

ggplot(data = pollution, aes(nox, mort)) + geom_point()
```




```
#a linear regression will probably not fit this data very well. It looks like there a
re a few points that might be outliers or something, and could pull a regression line
away from most of the other points along the left hand side of the plot

lin_fit <- stan_glm(mort ~ nox, data = pollution, refresh=0)
lin_fit
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      mort ~ nox
##  observations: 60
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) 942.6    8.9
## nox          -0.1    0.2
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 62.9    6.1
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
predicted <- predict(lin_fit)
resid <- pollution$mort - predicted

plot(predicted, resid, xlab = "predicted values", ylab = "residuals")
```

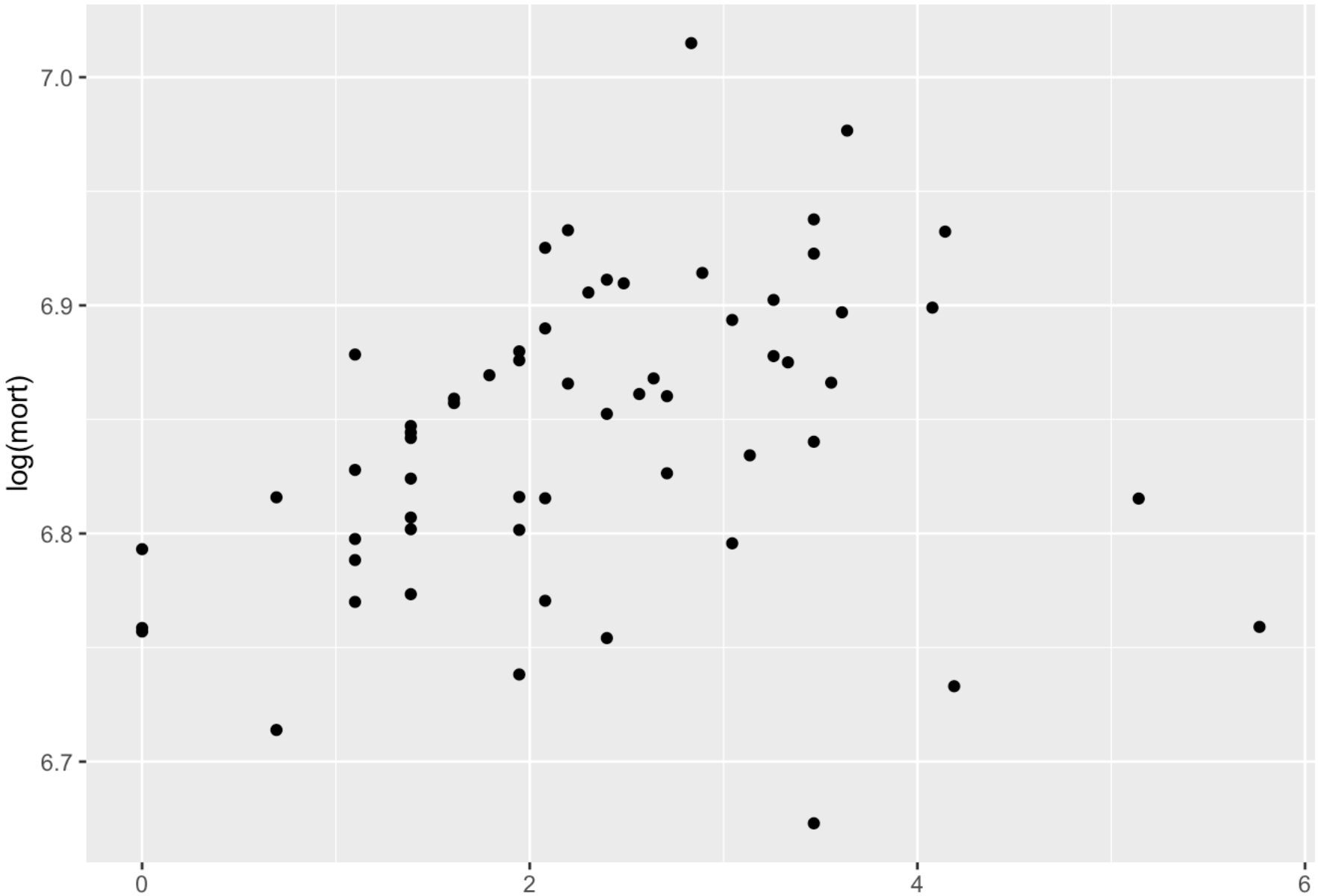



```
#These residuals are kind of a mess. They do not have an equal spread around 0, they are almost all stacked in one corner of the plot aside from a few random points. This makes sense when you look at the plot of the data, because you can see a general tren d on one side of the graph and a few outliers.
```

# (b)

Find an appropriate reansformation that will result in data more appropriate for linear regression. Fit a regression to the transformed data and evaluate the new residual plot.

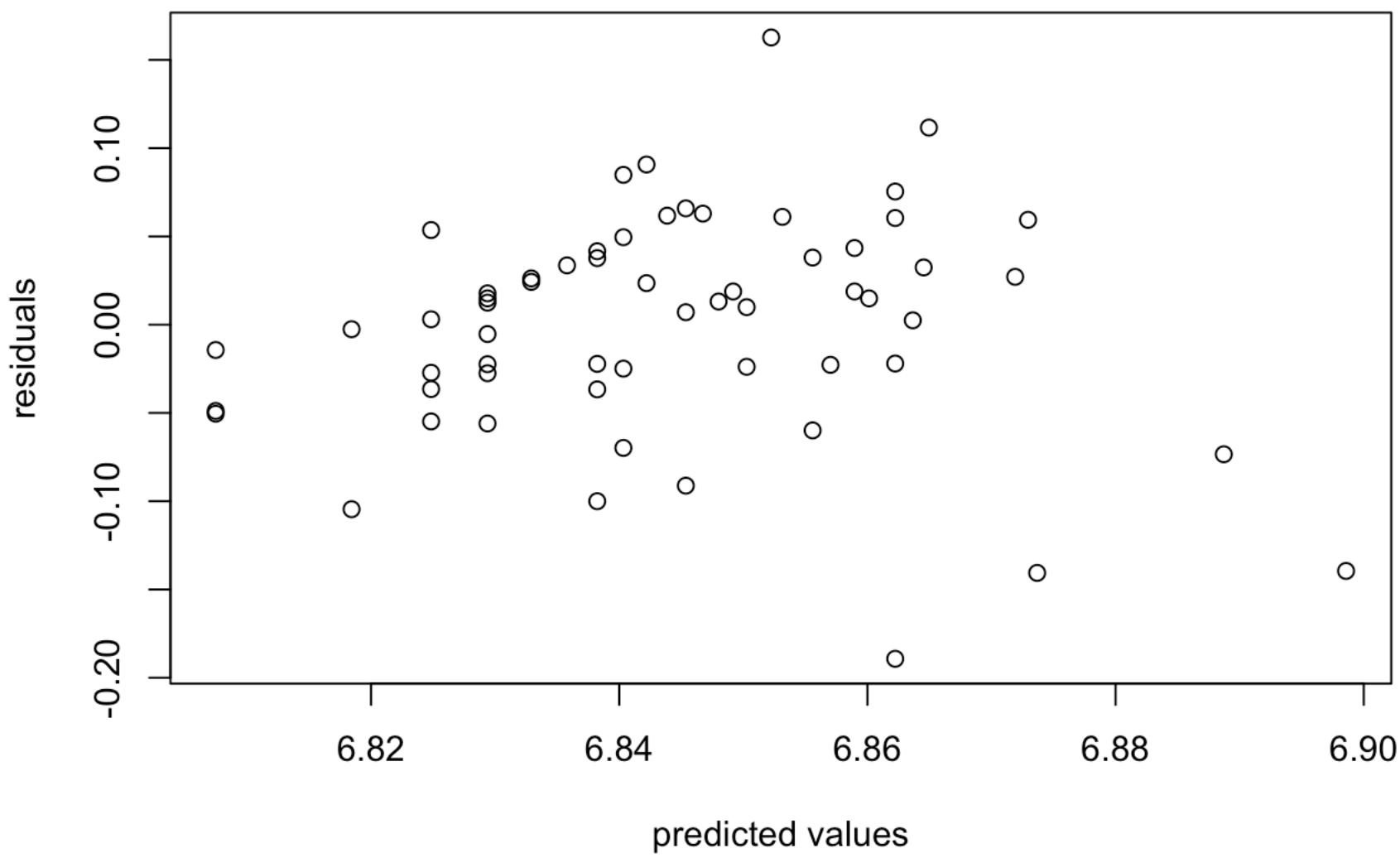
```
ggplot(data = pollution, aes(log(nox), log(mort))) + geom_point() + geom_abline()
```




```
fit2 <- stan_glm(log(mort) ~ log(nox), data = pollution, refresh = 0)
print(fit2, digits = 4)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      log(mort) ~ log(nox)
##  observations: 60
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) 6.8075 0.0189
## log(nox)    0.0158 0.0073
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.0644 0.0058
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
predicted <- predict(fit2)
resid <- log(pollution$mort) - predicted

plot(predicted, resid, xlab = "predicted values", ylab = "residuals")
```




```
#These residuals still don't look perfectly evenly distributed, but they're certainly
a lot better than what we had before.
```

# (c)

Interpret the slope coefficient from the model you chose in (b)

```
# A coefficient of 0.016 on the log scale is equal to 1.016 on the original scale. Be
cause we have taken the log of both the predictor and outcome variables, this coeffic
ent can be interpreted as the percent change in mortality rate that is associated wit
h a 1% increase in nox levels. So if nox levels increase by 1%, we can expect to see,
on average, a 1.6% increase in mortality (although we must be careful not to imply ca
usation from this data).
```

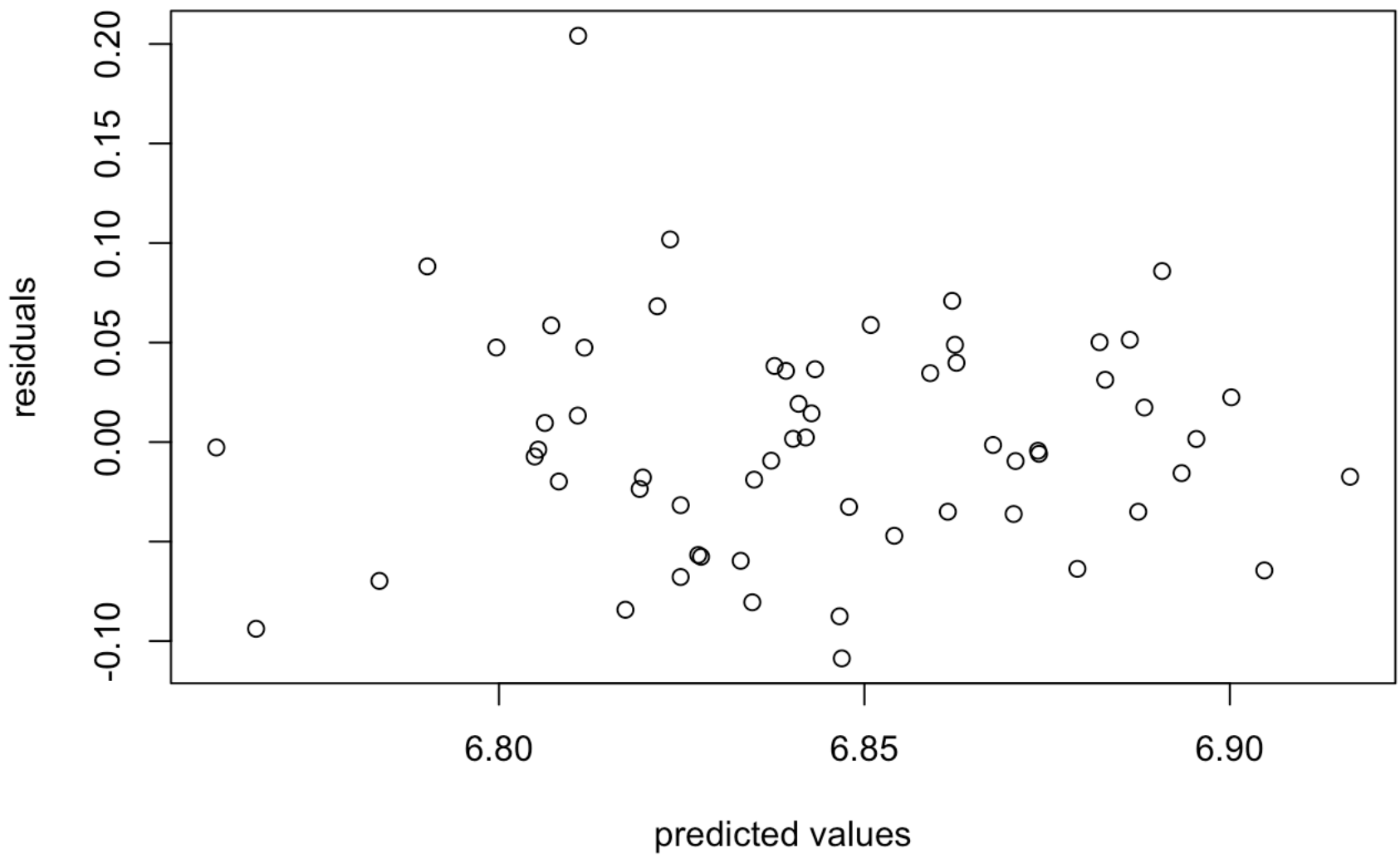
# (d)

Now fit a model predicting mortality rate using levels of nitric oxides, sulfur dioxide, and hydrocarbons as inputs. Use appropriate transformation when helpful. Plot the fitted regression model and interpret the coefficients.

```
fit3 <- stan_glm(log(mort) ~ log(nox) + log(hc) + log(so2), data = pollution, refresh
= 0)
print(fit3, digits = 4)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      log(mort) ~ log(nox) + log(hc) + log(so2)
##  observations: 60
##  predictors:   4
## ------
##             Median  MAD_SD
## (Intercept)  6.8249  0.0231
## log(nox)     0.0562  0.0223
## log(hc)     -0.0578  0.0195
## log(so2)     0.0148  0.0074
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.0579 0.0056
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
predicted <- predict(fit3)
resid <- log(pollution$mort) - predicted

plot(predicted, resid, xlab = "predicted values", ylab = "residuals")
```

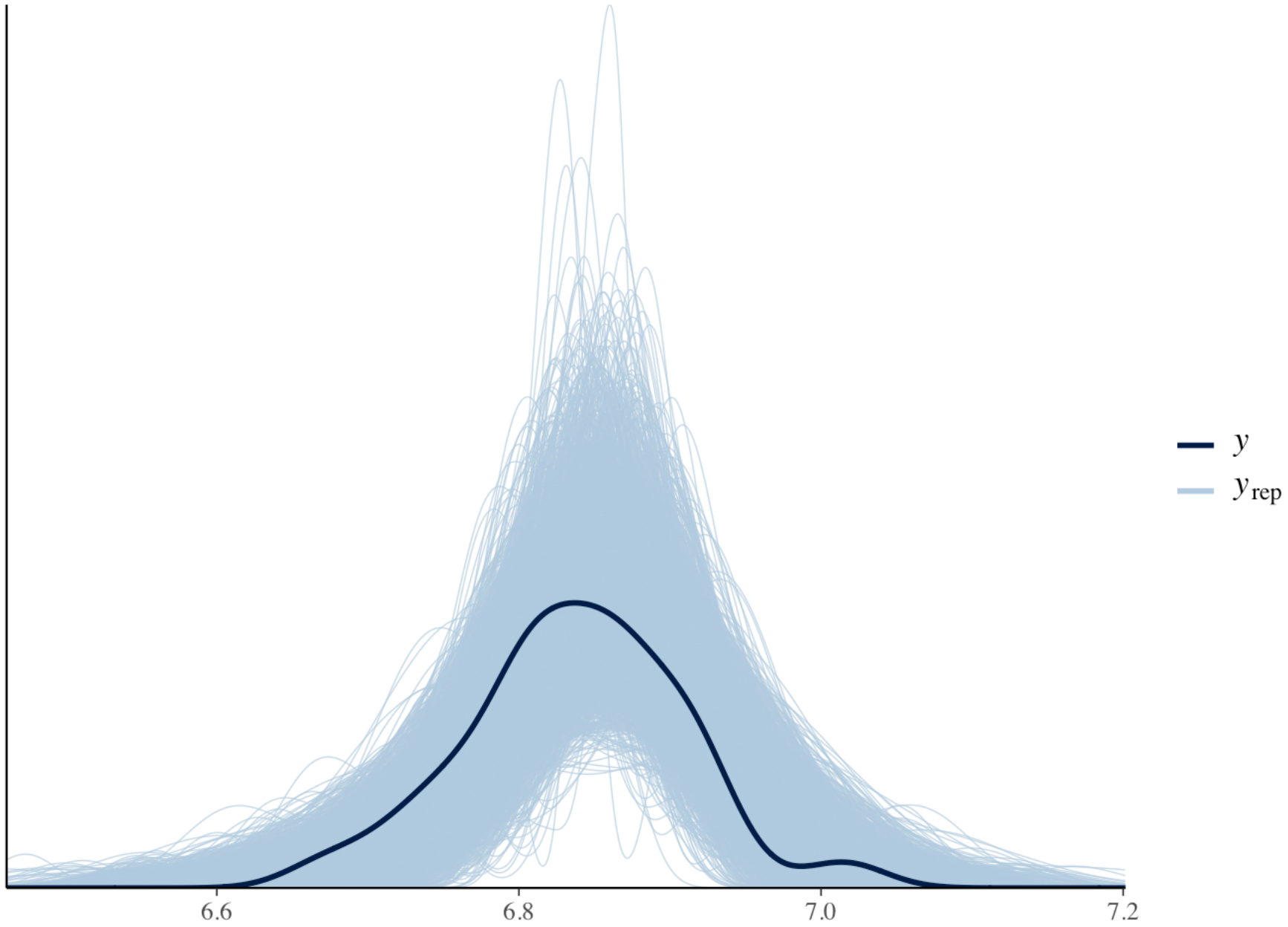



```
# The intercept is 6.8254 on the log scale, which is 925.2 on the original scale. Thi
s can be interpreted as the baseline for mortality. The log(nox) coefficent = 0.0559,
which is 1.057 on the original scale. This means that a 1% increase in nox is associa
ted with a 5.7% increase in mortality. The hc coefficient is -0.0572 on the log scale
, and 0.944 on the original scale. This means that a 1% increase in hc is associated
with about a 5.6% decrease in mortality (mortality gets multiplied by 94.4%). Finally
, the so2 coefficient is 0.0148 on the log scale and 1.0149 on the original scale. Th
is means that a 1% increase in so2 is associated with a 1.5% increase in mortality.
```

## (e)

Cross validate: fit the model you chose above to the first half of the data and then predict for the second half. You used all the data to construct the model in (d), so this is not really cross validation, but it gives a sense of how the steps of cross validation can be implemented.

```
training <- pollution %>% slice(1:30)
test <- pollution %>% slice(31:60)
fit4 <- stan_glm(log(mort) ~ log(nox) + log(hc) + log(so2), data = training, refresh
= 0)
post_fit <- posterior_predict(fit4, newdata = test)
ppc_dens_overlay(log(test$mort), post_fit)
```




# 12.7

Cross validation comparison of models with different transformations of outcomes: when we compare models with transformed continuous outcomes, we must take into account how the nonlinear transformation warps the continuous outcomes. Follow the procedure used to compare models for the mesquite bushes example on page 202.

## (a)

Compare models for earnings and for log(earnings) given height and sex as shown in page 84 and 192. Use earnk and log(earnk) as outcomes.

```
earnings <- read.csv("https://raw.githubusercontent.com/avehtari/ROS-Examples/master/
Earnings/data/earnings.csv")
fit_earnk <- stan_glm(earnk ~ height + male, data=earnings, refresh=0)

loo_1 <- loo(fit_earnk)
```

```
## Warning: Found 1 observation(s) with a pareto_k > 0.7. We recommend calling 'loo'
again with argument 'k_threshold = 0.7' in order to calculate the ELPD without the as
sumption that these observations are negligible. This will refit the model 1 times to
compute the ELPDs for the problematic observations directly.
```

```
print(loo_1)
```

```
##
## Computed from 4000 by 1816 log-likelihood matrix
##
##          Estimate    SE
## elpd_loo  -8153.3 172.0
## p_loo        28.2  20.7
## looic     16306.7 344.0
## ------
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     1815  99.9%   1047
##  (0.5, 0.7]   (ok)          0   0.0%   <NA>
##    (0.7, 1]   (bad)         0   0.0%   <NA>
##    (1, Inf)   (very bad)    1   0.1%   7
## See help('pareto-k-diagnostic') for details.
```

```
kfold_1 <- kfold(fit_earnk, K=10)
```

```
## Fitting model 1 out of 10
```

```
## Fitting model 2 out of 10
```

```
## Fitting model 3 out of 10
```

```
## Fitting model 4 out of 10
```

```
## Fitting model 5 out of 10
```

```
## Fitting model 6 out of 10
```

```
## Fitting model 7 out of 10
```

```
## Fitting model 8 out of 10
```

```
## Fitting model 9 out of 10
```

```
## Fitting model 10 out of 10
```

```
fit_log_earnk <- stan_glm(log(earnk) ~ height + male, data = earnings, refresh=0, sub
set = earnk>0)

loo_2 <- loo(fit_log_earnk)
print(loo_2)
```

```
##
## Computed from 4000 by 1629 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo  -2083.6 38.7
## p_loo         4.9  0.4
## looic      4167.3 77.5
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
#loo_2_with_jacobian <- loo_2
#loo_2_with_jacobian$pointwise <- loo_2_with_jacobian$pointwise[,1] - log(earnings$ea
rnk)

#sum(loo_2_with_jacobian$pointwise)
#loo_compare(kfold_1, loo_2_with_jacobian)


#Based on the loo outputs, the log model is a better fit. This can be seen when you l
ook at the values for elpd_loo. For the log model, that value is closer to zero. The
more negative the value, the worse the model fit.

#I tried to make a jacobian adjustment so that I could compare these models more dire
ctly, but was not able to get past the error messages.
```

# (b)

Compare models from other exercises in this chapter.

# 12.8

Log-log transformations: Suppose that, for a certain population of animals, we can predict log weight from log height as follows:

- An animal that is 50 centimeters tall is predicted to weigh 10 kg.
- Every increase of 1% in height corresponds to a predicted increase of 2% in weight.
- The weights of approximately 95% of the animals fall within a factor of 1.1 of predicted values.

## (a)

Give the equation of the regression line and the residual standard deviation of the regression.

```
#exp(height) = 1.02
#height = 0.0198
#log(weight) = 2.3026 + 0.0198*log(height) + error

#exp(2*sigma) = 1.1
# sigma = log(1.1)/2 = 0.04766
```

## (b)

Suppose the standard deviation of log weights is 20% in this population. What, then, is the $R^2$ of the regression model described here?

```
#R^2 = 1 - ((0.04766)^2)/((0.2)^2) = 0.9432
```

# 12.9

Linear and logarithmic transformations: For a study of congressional elections, you would like a measure of the relative amount of money raised by each of the two major-party candidates in each district. Suppose that you know the amount of money raised by each candidate; label these dollar values Di and Ri. You would like to combine these into a single variable that can be included as an input variable into a model predicting vote share for the Democrats. Discuss the advantages and disadvantages of the following measures:

## (a)

The simple difference, $D_i - R_i$

```
#The simple differnce is easy to calculate and easy to understand/interpret. However,
it might not be very informative, since it only shows the difference between the two
without giving us a sense of how much money is being raised by each party. For exampl
e, a difference of 1000 is a very large difference proportionally if Di= 2000 and Ri=
1000. However, this difference doesn't seem nearly as signigicant if each candidate i
s raising millions of dollars.
```

## (b)

The ratio, $D_i/R_i$

```
#The advantages and disadvantages here are similar to that above. It is easy to calcu
late and understand, but isn't necessarily very informative depending on how large of
numbers Di and Ri are.
```

## (c)

The difference on the logarithmic scale, $log\ D_i - log\ R_i$

```
#Using a log scale is nice because it sort of pulls in large numbers. However, this m
ight not be good if you are calculating the difference, because your difference is li
kely to be smaller and more difficult to interpret.
```

## (d)

The relative proportion, $D_i/(D_i + R_i)$.

```
#To me, this seems more informative than the above options. It's nice to use relative
proportions and ratios instead of simple differences because you can get a better sen
se of both parties' fundraising instead of just a value without any real context. I'm
not really sure what the disadvantages would be here, other than that it's just not a
n all-encompassing predictor. You would still be curious to know other information ab
out the candidates to be able to predict vote share.
```

# 12.11

Elasticity: An economist runs a regression examining the relations between the average price of cigarettes, P, and the quantity purchased, Q, across a large sample of counties in the United States, assuming the functional form, $logQ = \alpha + \beta logP$. Suppose the estimate for $\beta$ is 0.3. Interpret this coefficient.

```
#This means that for a 1% change in the price of cigarettes, there is an associated 0
.3% change in the quantity purchased.
```

# 12.13

Building regression models: Return to the teaching evaluations data from Exercise 10.6. Fit regression models predicting evaluations given many of the inputs in the dataset. Consider interactions, combinations of predictors, and transformations, as appropriate. Consider several models, discuss in detail the final model that you choose, and also explain why you chose it rather than the others you had considered.

```
Beauty <- read.csv("https://raw.githubusercontent.com/avehtari/ROS-Examples/master/Beauty/data/beauty.csv")

fit <- stan_glm(eval ~ age + minority + age:minority, data = Beauty, refresh=0)
print(fit)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      eval ~ age + minority + age:minority
##  observations: 463
##  predictors:   4
## ------
##              Median MAD_SD
## (Intercept)   4.3    0.1
## age           0.0    0.0
## minority     -2.3    0.5
## age:minority  0.0    0.0
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.5    0.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
ggplot(data = Beauty, aes(age, eval)) + geom_point(aes(color = minority)) + geom_abline(intercept = coef(fit)[1], slope = coef(fit)[2]) + geom_abline(intercept = coef(fit)[1]+coef(fit)[3], slope = coef(fit)[2]+coef(fit)[4], color = "red")
```


```
# This model shows a regression of evaluation on age, minority, and the interaction b
etween the two. I chose this model because it shows a very clear age:minority interac
tion with teachers who are older and minorities receiving higher evaluation scores, a
nd teachers who are younger and not minorities receiving lower scores. There did not
appear to be a strong slope for age. I tried standardizing the age by subtracting the
mean age, but because there was not a big slope for age to begin wiht, standardizing
did not do anything.
```

# 12.14

Prediction from a fitted regression: Consider one of the fitted models for mesquite leaves, for example fit_4, in Section 12.6. Suppose you wish to use this model to make inferences about the average mesquite yield in a new set of trees whose predictors are in data frame called new_trees. Give R code to obtain an estimate and standard error for this population average. You do not need to make the prediction; just give the code.

```
#predicted <- posterior_predict(fit_4, newdata= new_trees, draws=1000)
#mean(predicted)
#sd(predicted)
```