Regression

Mark Hagemann

March 5, 2015

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Outline

- What is a regression model and what's it good for?

Outline

- What is a regression model and what's it good for?
  - Explanation

Outline

- What is a regression model and what's it good for?
  - Explanation
  - Quantifying relationships

Outline

- What is a regression model and what's it good for?
  - Explanation
  - Quantifying relationships
  - Prediction

Outline

- What is a regression model and what's it good for?
  - Explanation
  - Quantifying relationships
  - Prediction
- Limitations

Outline

- What is a regression model and what's it good for?
  - Explanation
  - Quantifying relationships
  - Prediction
- Limitations
  - Assumptions (linearity, i.i.d., etc.)

Outline

- What is a regression model and what's it good for?
  - Explanation
  - Quantifying relationships
  - Prediction
- Limitations
  - Assumptions (linearity, i.i.d., etc.)
  - Predicting is hard!

Approach

- Look at your data

Approach

- Look at your data
- build model(s)

Approach

- Look at your data
- build model(s)
- Check assumptions

Approach

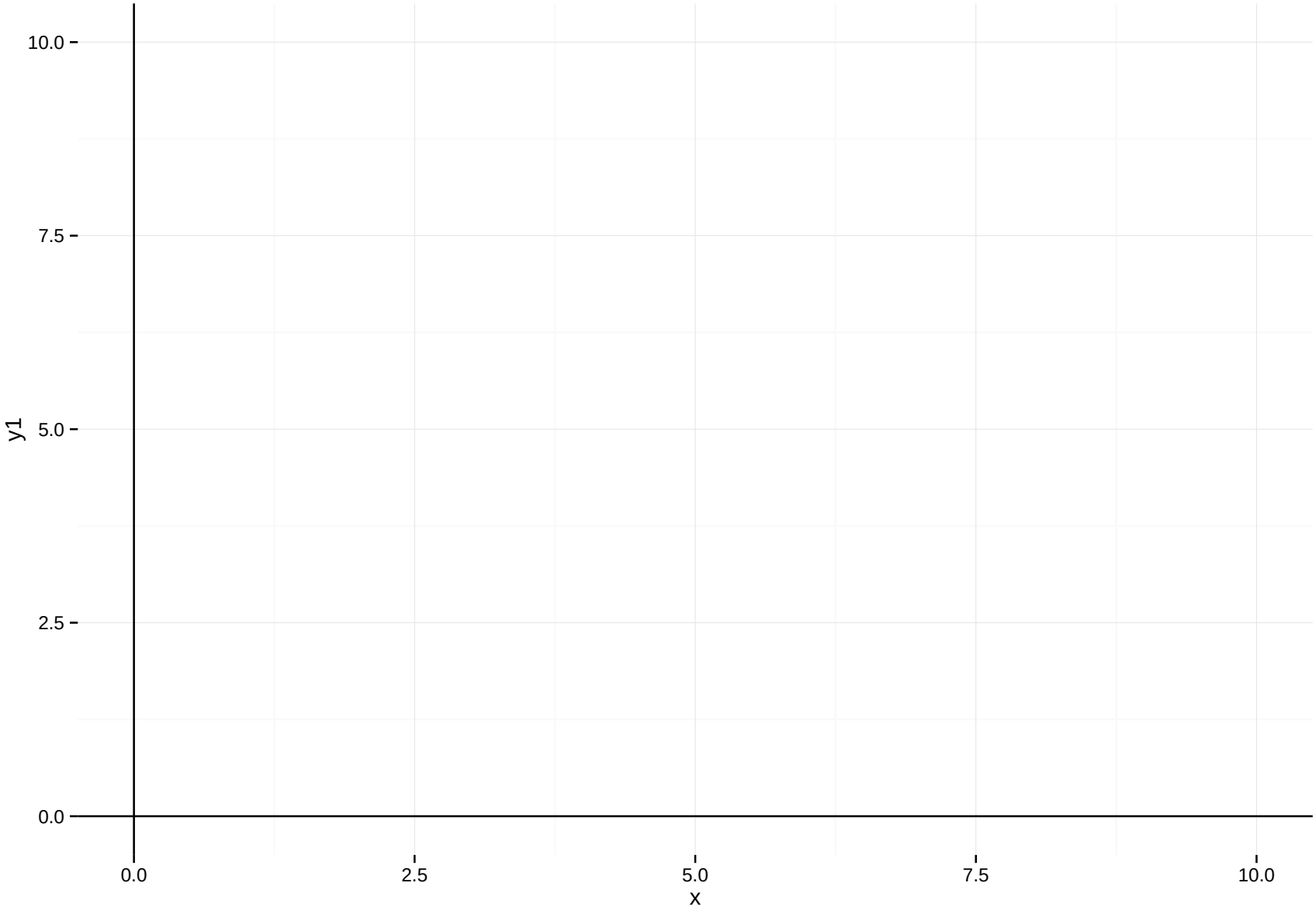
- Look at your data
- build model(s)
- Check assumptions
- select model

Approach

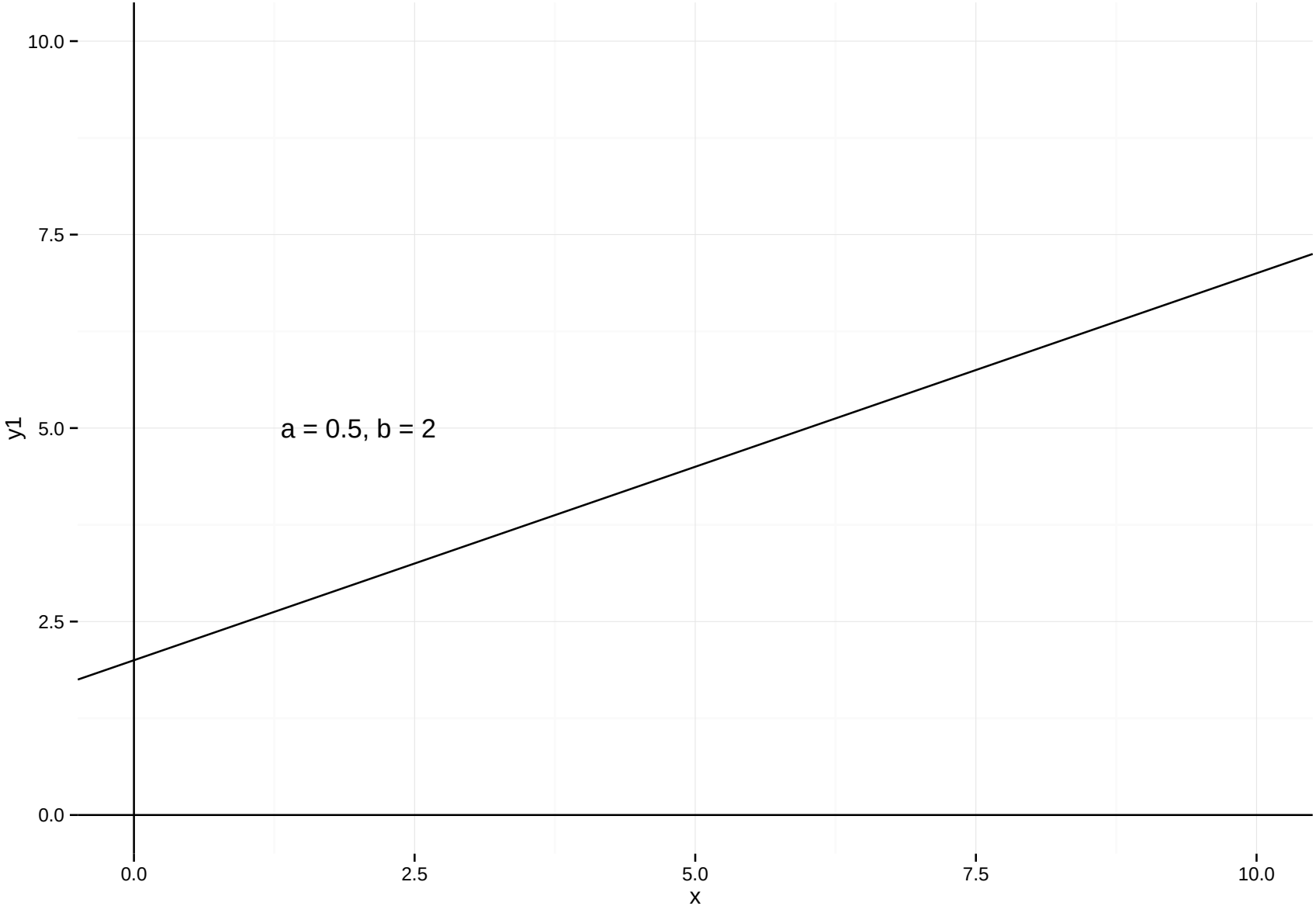
- Look at your data
- build model(s)
- Check assumptions
- select model
- Use model

Math background

Linear functions

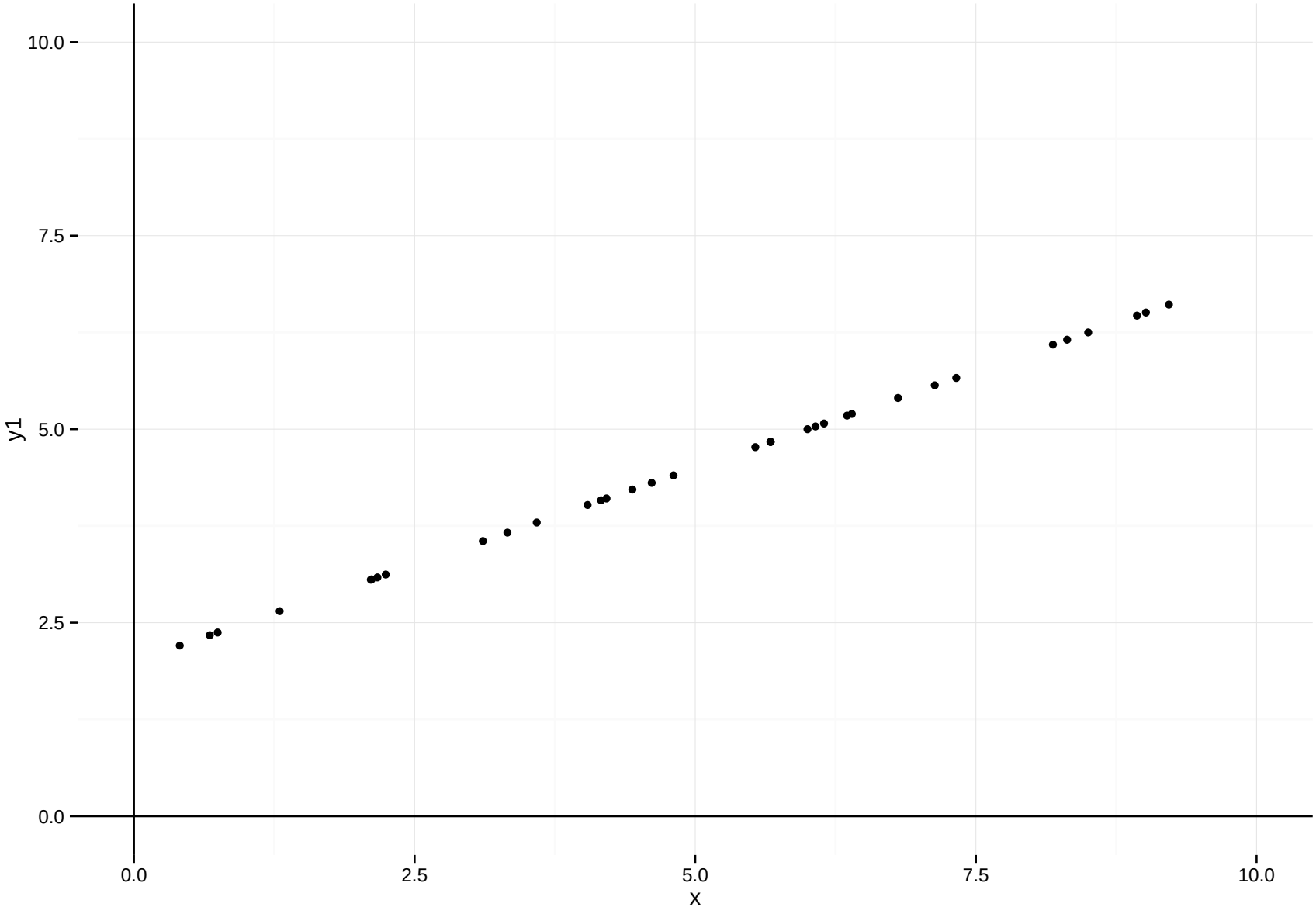


Linear functions

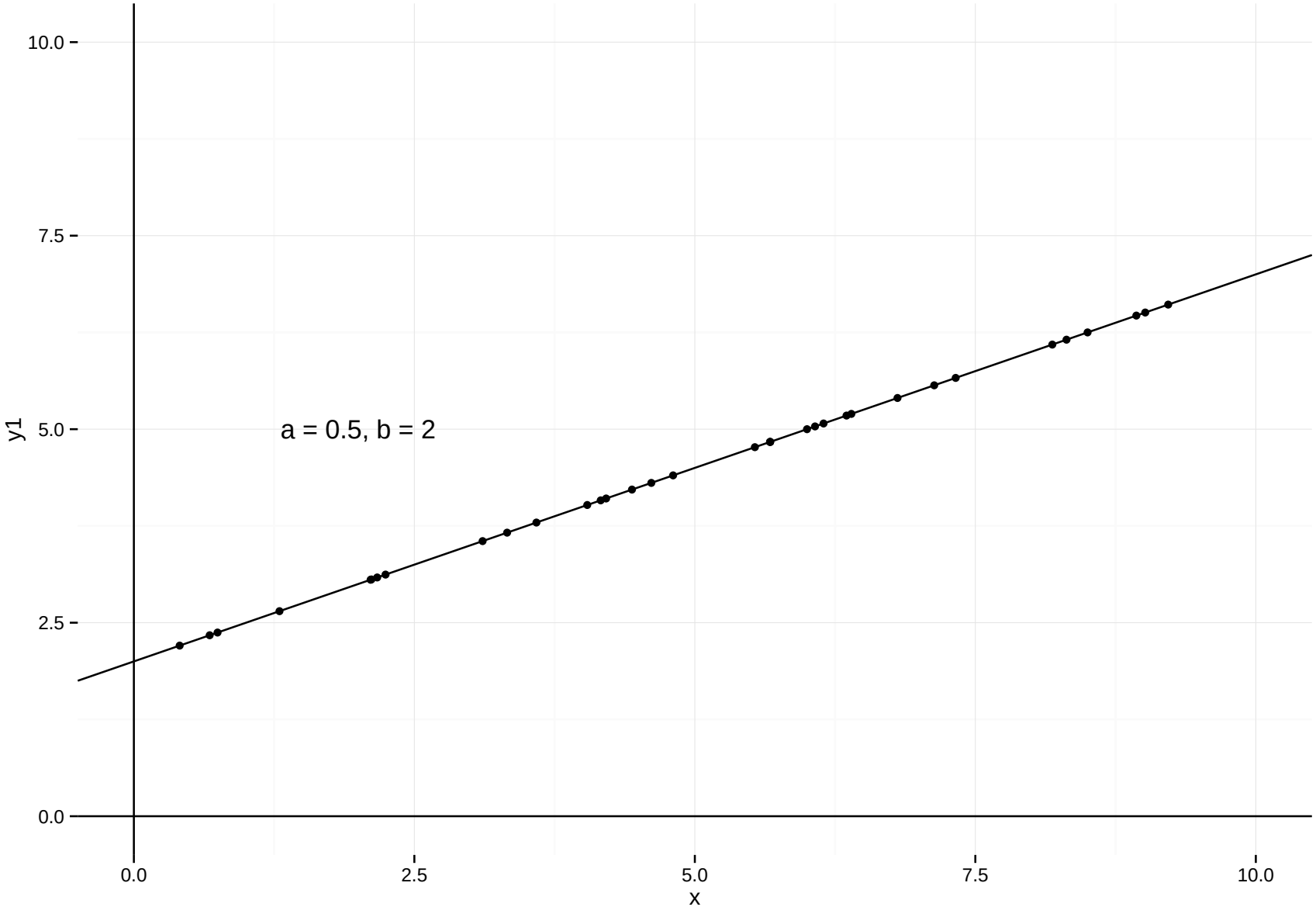


- $y = a \times x + b$

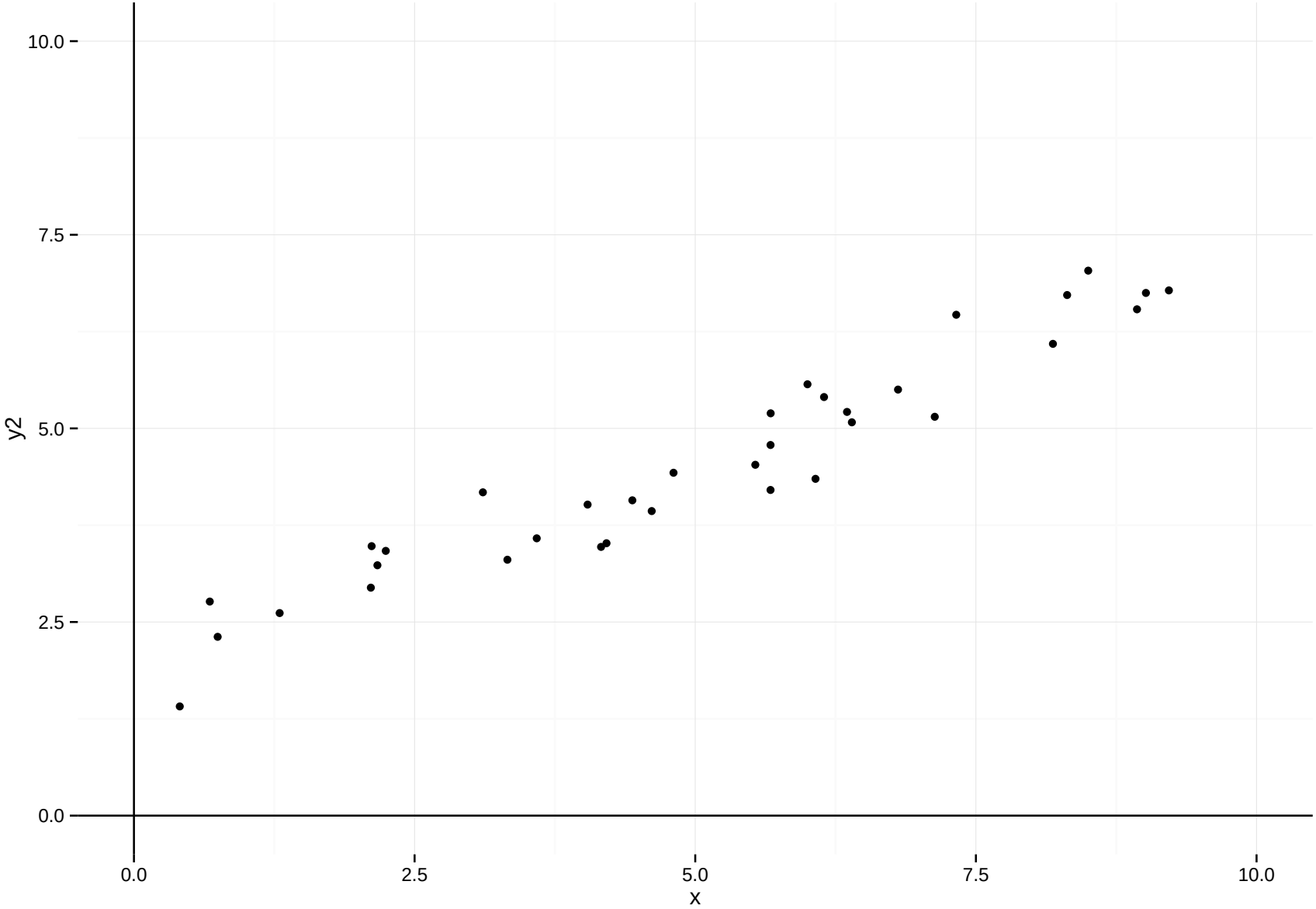
Some (perfectly linear) data



A (perfect) linear regression line

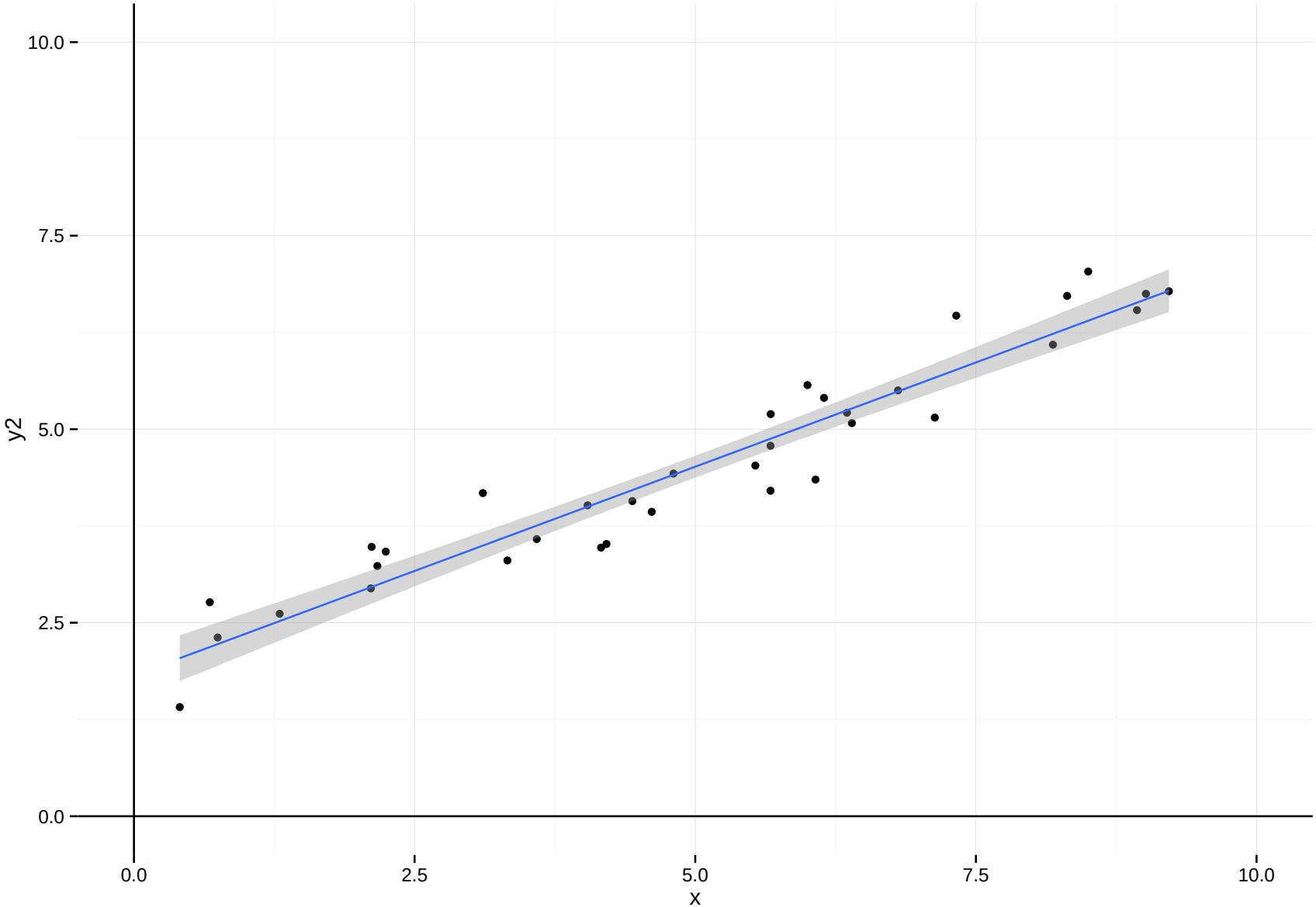


More realistic case

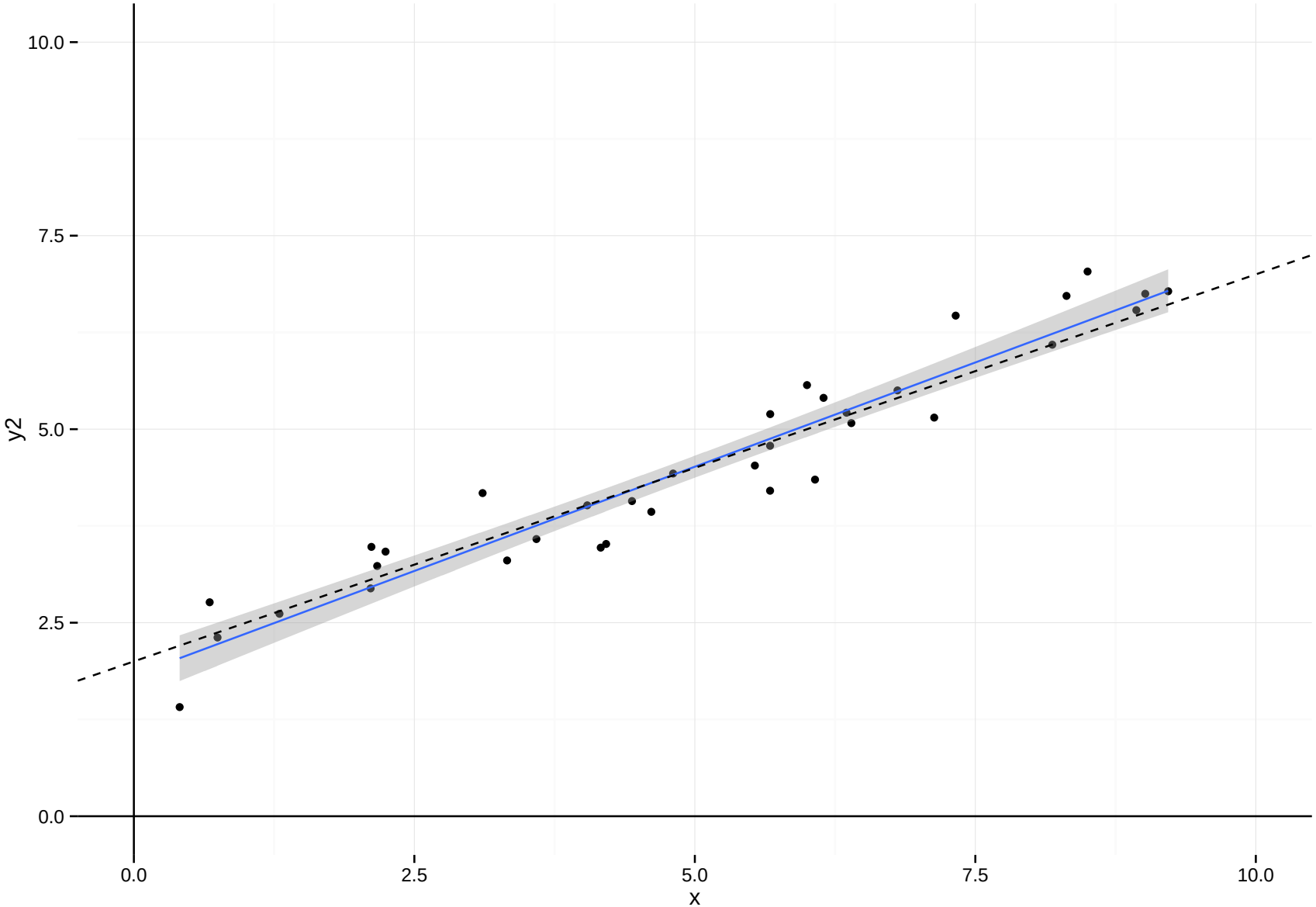


In reality, we start with the (noisy) data, and we seek the underlying function

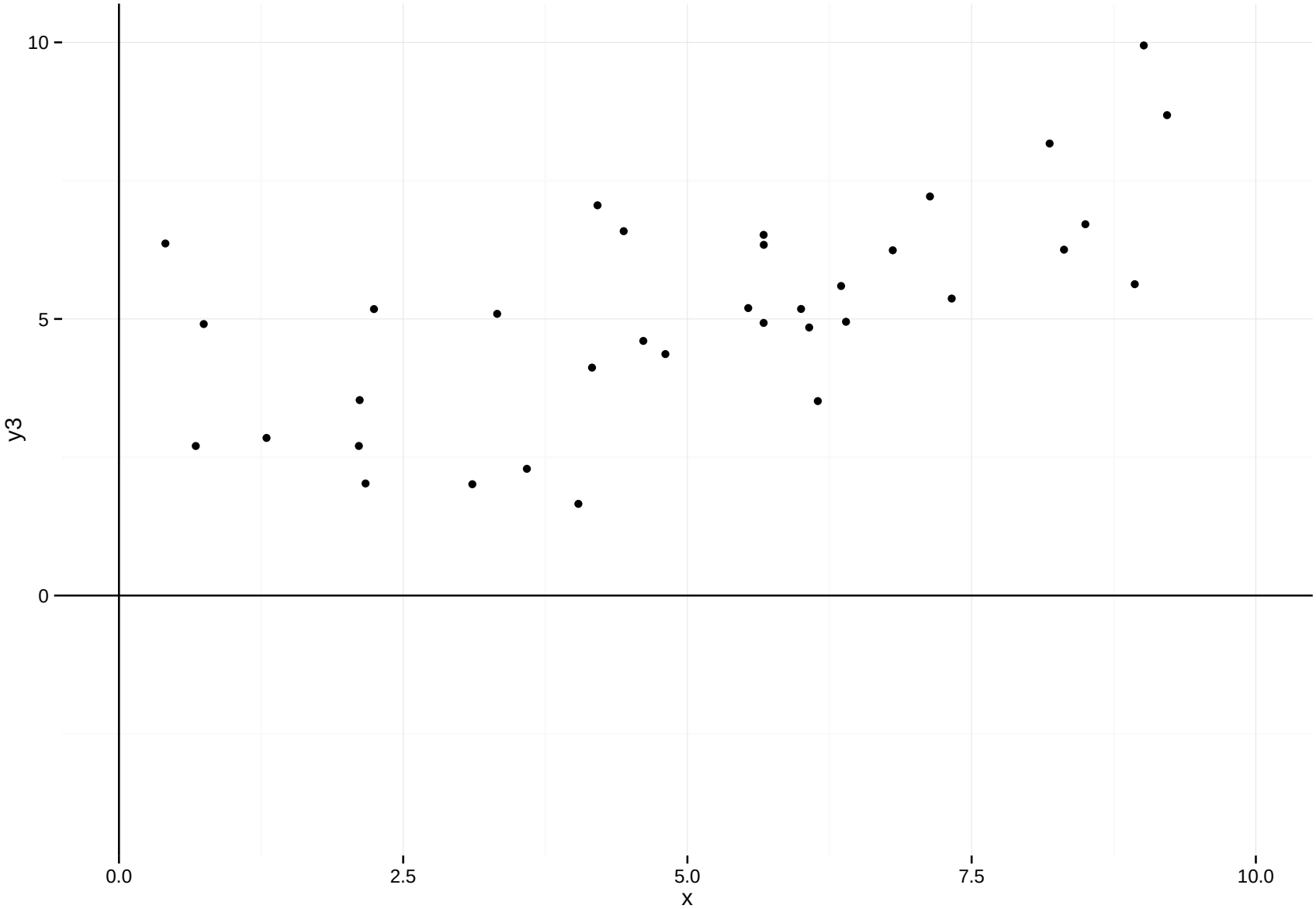
Fit a line



Compare to “actual” function

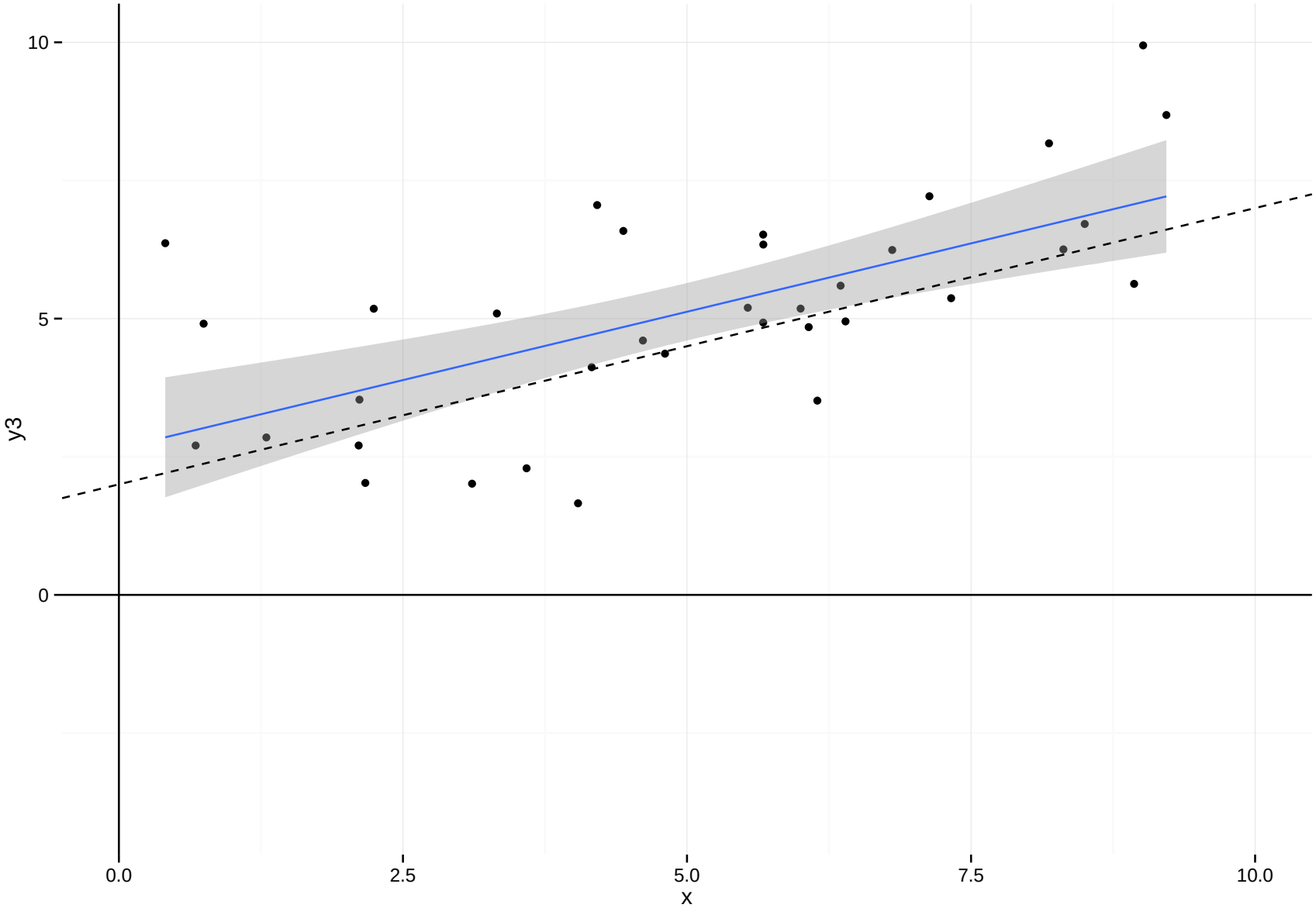


Impact of noise



These data are 4 times noisier than the previous

Now fit a line



Dashed line is generating function.

Example: state demographics

```
data(state)
kable(head(state.x77))
```

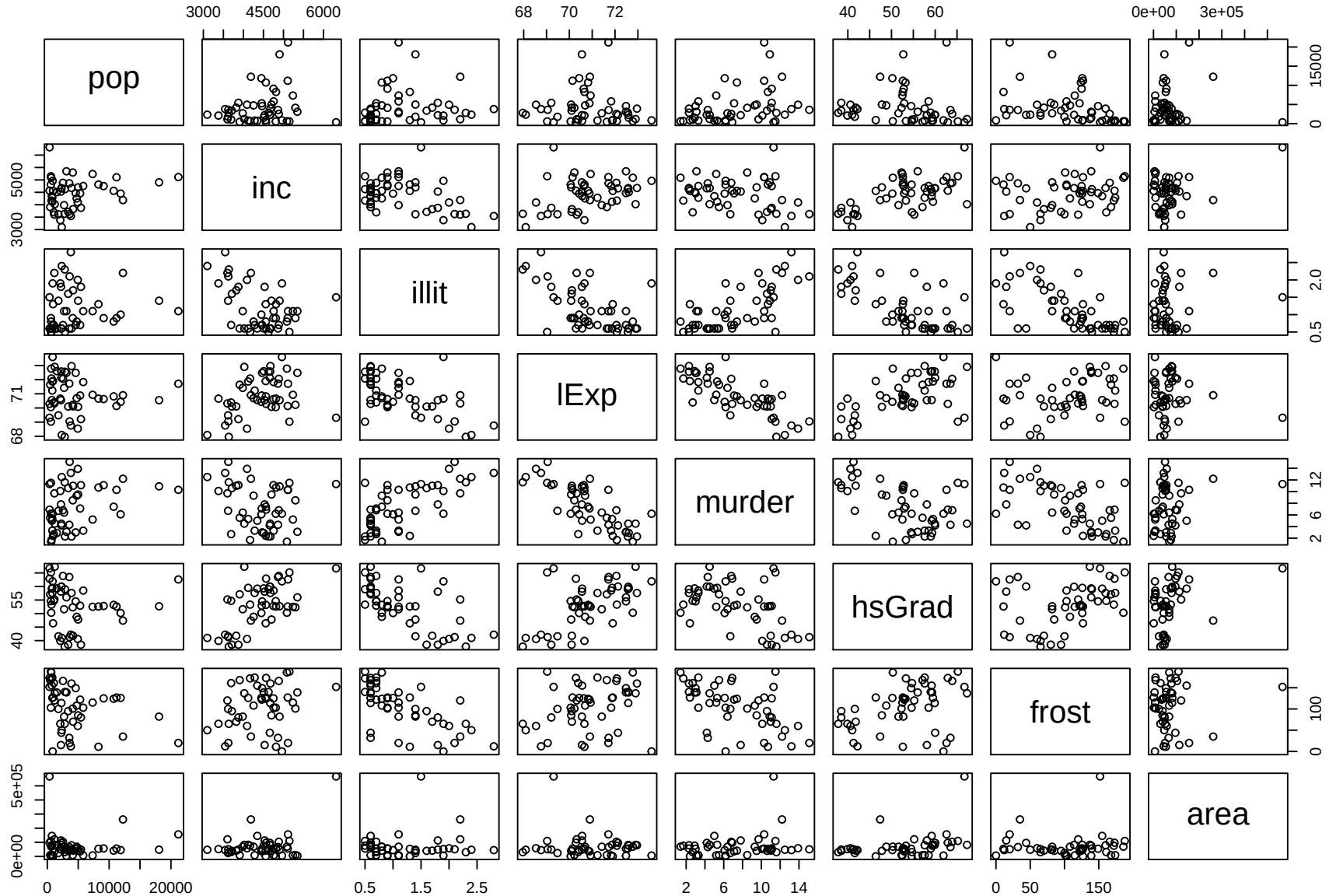
	Population	Income	Illiteracy	Life Exp	Murder	HS Gr
Alabama	3615	3624	2.1	69.05	15.1	41
Alaska	365	6315	1.5	69.31	11.3	66
Arizona	2212	4530	1.8	70.55	7.8	58
Arkansas	2110	3378	1.9	70.66	10.1	39
California	21198	5114	1.1	71.71	10.3	62
Colorado	2541	4884	0.7	72.06	6.8	63

Rename the columns

```
state = state.x77 %>%
  as.data.frame() %>%
  setNames(c("pop", "inc", "illit", "lExp", "murder", "hsGrad",
kable(head(state))
```

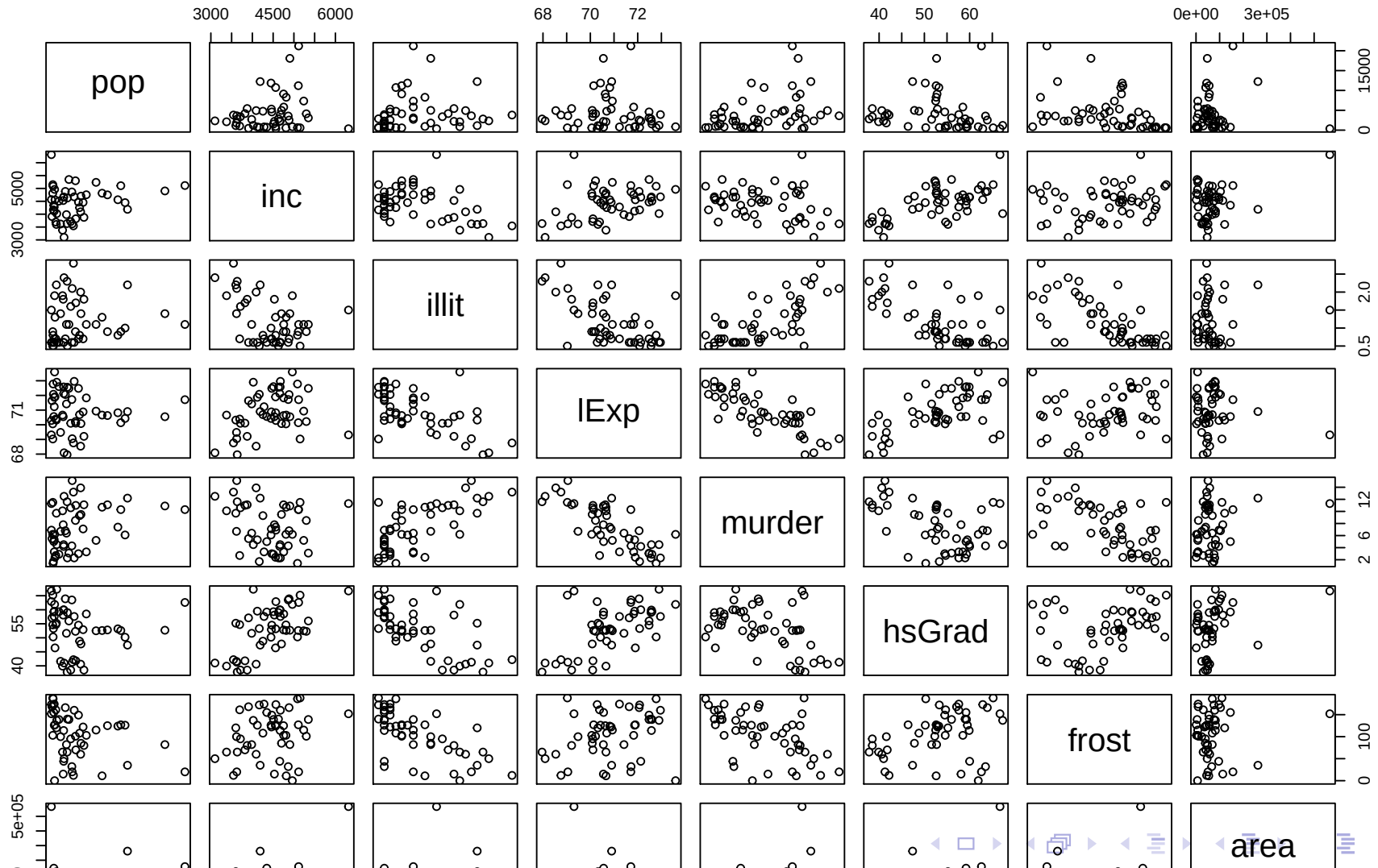
	pop	inc	illit	lExp	murder	hsGrad	frost	ar
Alabama	3615	3624	2.1	69.05	15.1	41.3	20	507
Alaska	365	6315	1.5	69.31	11.3	66.7	152	5664
Arizona	2212	4530	1.8	70.55	7.8	58.1	15	1134
Arkansas	2110	3378	1.9	70.66	10.1	39.9	65	519
California	21198	5114	1.1	71.71	10.3	62.6	20	1563
Colorado	2541	4884	0.7	72.06	6.8	63.9	166	1037

Inspect all pairwise relationships using `pairs()` function



Make a model! For life expectancy...

```
mod1 = lm(lExp ~ hsGrad, state)
mod2 = lm(lExp ~ murder + hsGrad, state)
mod3 = lm(lExp ~ murder + hsGrad + illit, state)
```



See the good parts using `summary()`

Model 1:

```
##
## Call:
## lm(formula = lExp ~ hsGrad, data = state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.01867 -0.67517 -0.07538  0.64483  2.17311
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 65.73965    1.04748  62.760  < 2e-16 ***
## hsGrad       0.09676    0.01950   4.961  9.2e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
##
## Residual standard error: 1.103 on 48 degrees of freedom
## Multiple R-squared:  0.339,  Adjusted R-squared:  0.3252
## F-statistic: 24.61 on 1 and 48 DF,  p-value: 9.196e-06
```

Model 2:

```
##
## Call:
## lm(formula = lExp ~ murder + hsGrad, data = state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66758 -0.41801  0.05602  0.55913  2.05625
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 70.29708    1.01567  69.213  < 2e-16 ***
## murder      -0.23709    0.03529  -6.719 2.18e-08 ***
## hsGrad       0.04389    0.01613   2.721  0.00909 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
##
## Residual standard error: 0.7959 on 47 degrees of freedom
## Multiple R-squared:  0.6628, Adjusted R-squared:  0.6485
## F-statistic:  46.2 on 2 and 47 DF,  p-value: 8.016e-12
```

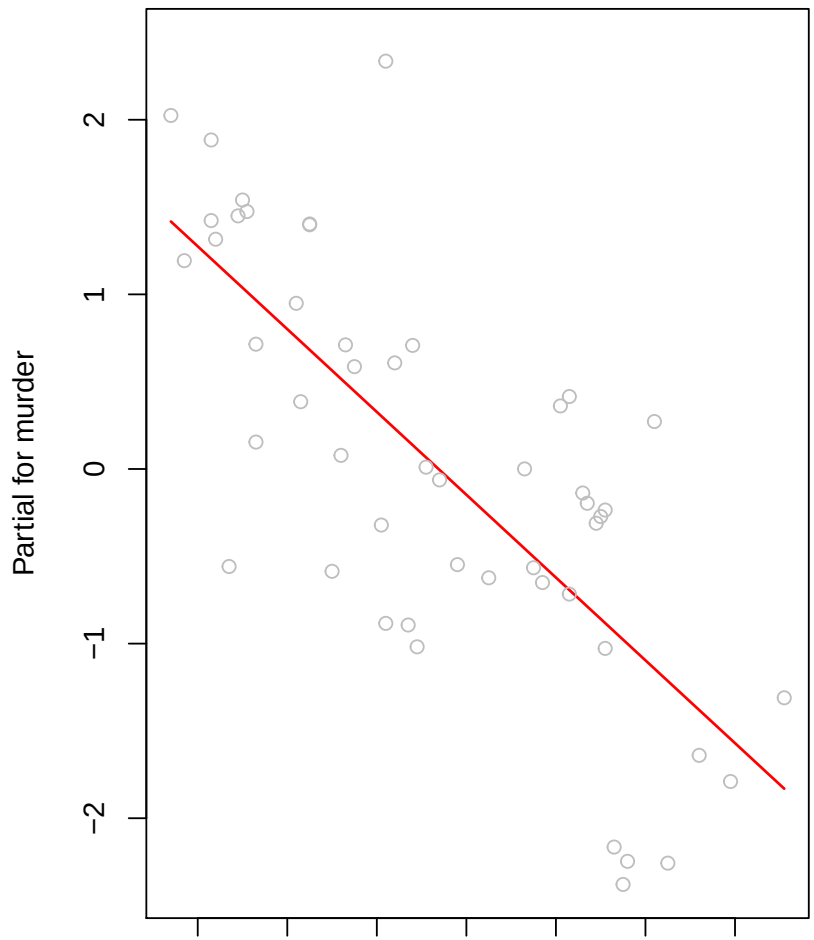
Model 3:

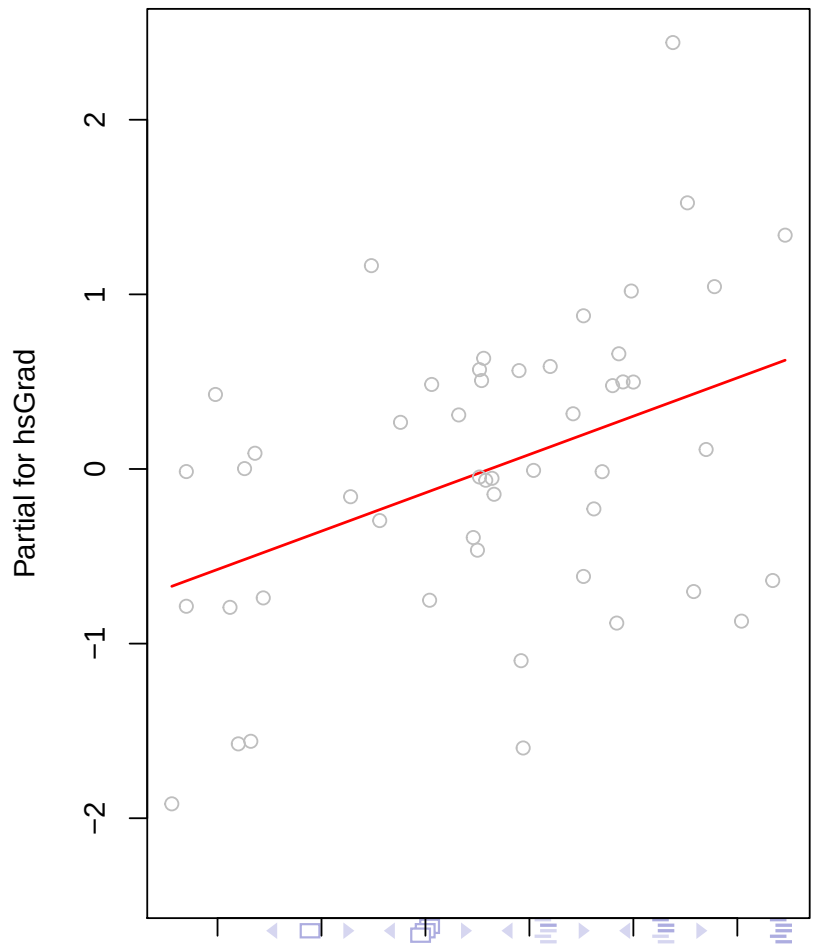
```
##
## Call:
## lm(formula = lExp ~ murder + hsGrad + illit, data = state)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65922 -0.46400  0.08517  0.59643  1.77657
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 69.73545    1.22208  57.063  < 2e-16 ***
## murder      -0.25813    0.04350  -5.934 3.63e-07 ***
## hsGrad       0.05179    0.01876   2.761  0.00825 **
## illit        0.25398    0.30508   0.833  0.40942
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
##
## Residual standard error: 0.7985 on 46 degrees of freedom
## Multiple R-squared:  0.6679, Adjusted R-squared:  0.6462
## F-statistic: 30.83 on 3 and 46 DF,  p-value: 4.444e-11
```

Inspect the model!

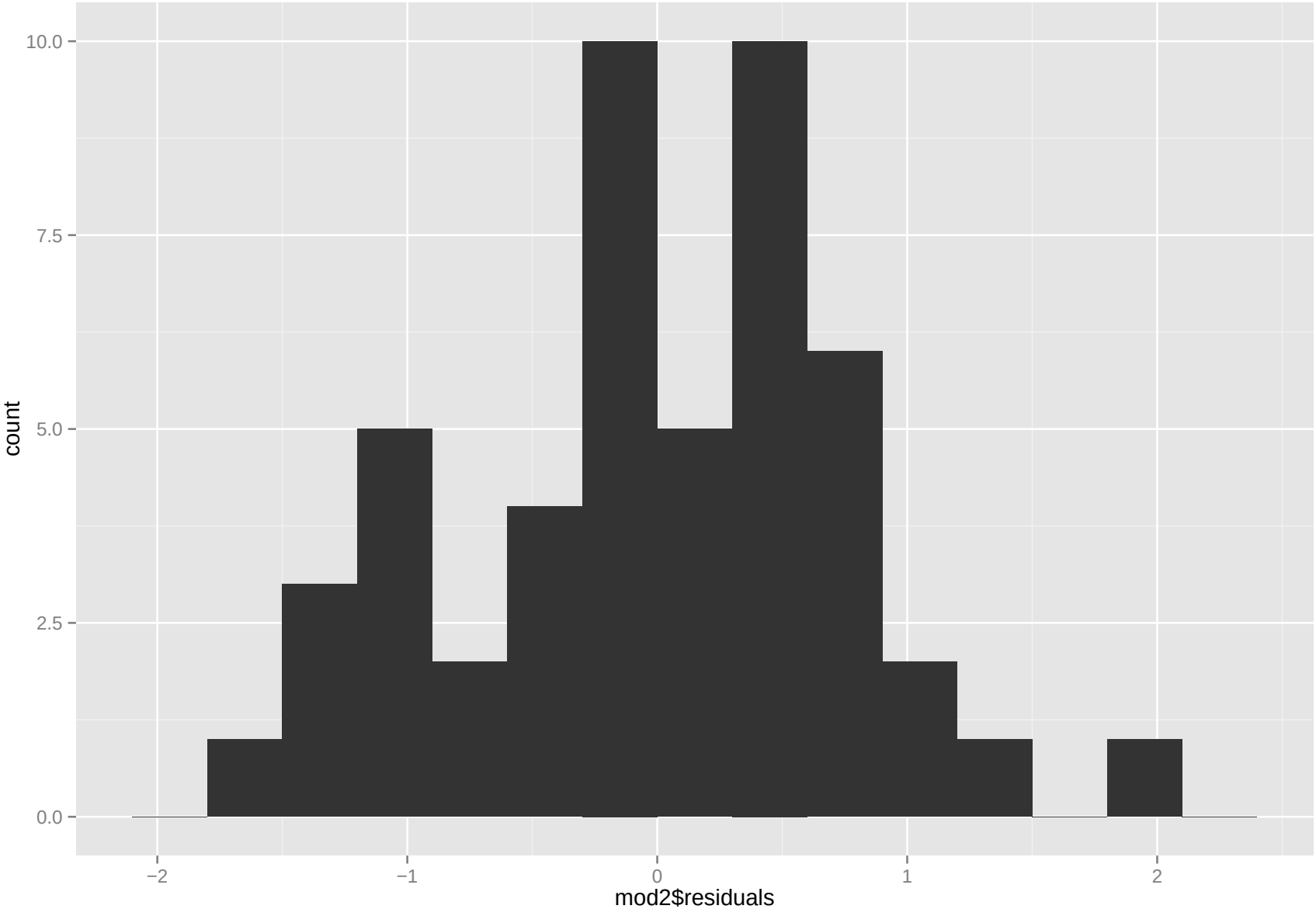
`termplot()` function

```
par(mfrow = c(1, 2))
termplot(mod2, partial.resid = TRUE)
```





`qplot()` to look at residual distribution



- Could also use `hist()`, `qqnorm()`, etc.

Make a prediction!

`predict()` function

```
newdat = data.frame(murder = 9, hsGrad = 77)
kable(newdat)
```

murder	hsGrad
9	77

Make a prediction!

`predict()` function

```
predict(mod2, newdat, se.fit = TRUE) %>%
  as.data.frame() %>%
  kable()
```

fit	se.fit	df	residual.scale
71.5426	0.4311892	47	0.7958717

Other topics:

- `glm()`

That's all!

Other topics:

- `glm()`
- `gam()` - **mgcv** package

That's all!

Other topics:

- `glm()`
- `gam()` - **mgcv** package
- `arima()` - time series models

That's all!

Other topics:

- `glm()`
- `gam()` - **mgcv** package
- `arima()` - time series models
- automated model selection

That's all!

Other topics:

- `glm()`
- `gam()` - **mgcv** package
- `arima()` - time series models
- automated model selection
  - stepwise `step()` function,

That's all!

Other topics:

- `glm()`
- `gam()` - **mgcv** package
- `arima()` - time series models
- automated model selection
  - stepwise `step()` function,
  - best subset **leaps** package

That's all!

Other topics:

- `glm()`
- `gam()` - **mgcv** package
- `arima()` - time series models
- automated model selection
  - stepwise `step()` function,
  - best subset **leaps** package
- machine learning methods - various packages

That's all!