C1M1_peer_reviewed

May 26, 2023

1 Module 1: Peer Reviewed Assignment

1.0.1 Outline:

The objectives for this assignment:

1. Learn when and how simulated data is appropriate for statistical analysis.
2. Experiment with the processes involved in simulating linear data.
3. Observe how the variance of data effects the best-fit line, even for the same underlying population.
4. Recognize the effects of standardizing predictors.
5. Interpreting the coefficients of linear models on both original and standardized data scales.

General tips:

1. Read the questions carefully to understand what is being asked.
2. This work will be reviewed by another human, so make sure that you are clear and concise in what your explanations and answers.

A Quick Note On Peer-Reviewed Assignments

Welcome to your first peer reviewed assignment! These assignments will be a more open form than the auto-graded assignments, and will focus on interpretation and visualization rather than "do you get the right numbers?" These assignments will be graded by your fellow students (except in the specific cases where the work needs to be graded by a proctor) so please make your answers as clear and concise as possible.

[2]:
```
# This cell loads the necesary libraries for this assignment
library(tidyverse)
```

2 Problem 1: Simulating Data

We're going to let you in on a secret. The turtle data from the autograded assignment was simulated...fake data! Gasp! Importantly, simulating data, and applying statistical models to simulated data, are very important tools in data science.

Why do we use simulated data? Real data can be messy, noisy, and we almost never *really* know the underlying process that generated real data. Working with real data is always our ultimate end goal, so we will try to use as many real datasets in this course as possible. However, applying

models to simulated data can be very instructive: such applications help us understand how models work in ideal settings, how robust they are to changes in modeling assumptions, and a whole host of other contexts.

And in this problem, you are going to learn how to simulate your own data.

1. (a) A Simple Line Starting out, generate 10 to 20 data points for values along the x-axis. Then generate data points along the y-axis using the equation $y_i = \beta_0 + \beta_1 x_i$. Make it a straight line, nothing fancy.

Plot your data (using ggplot!) with your **x** data along the x-axis and your **y** data along the y-axis.

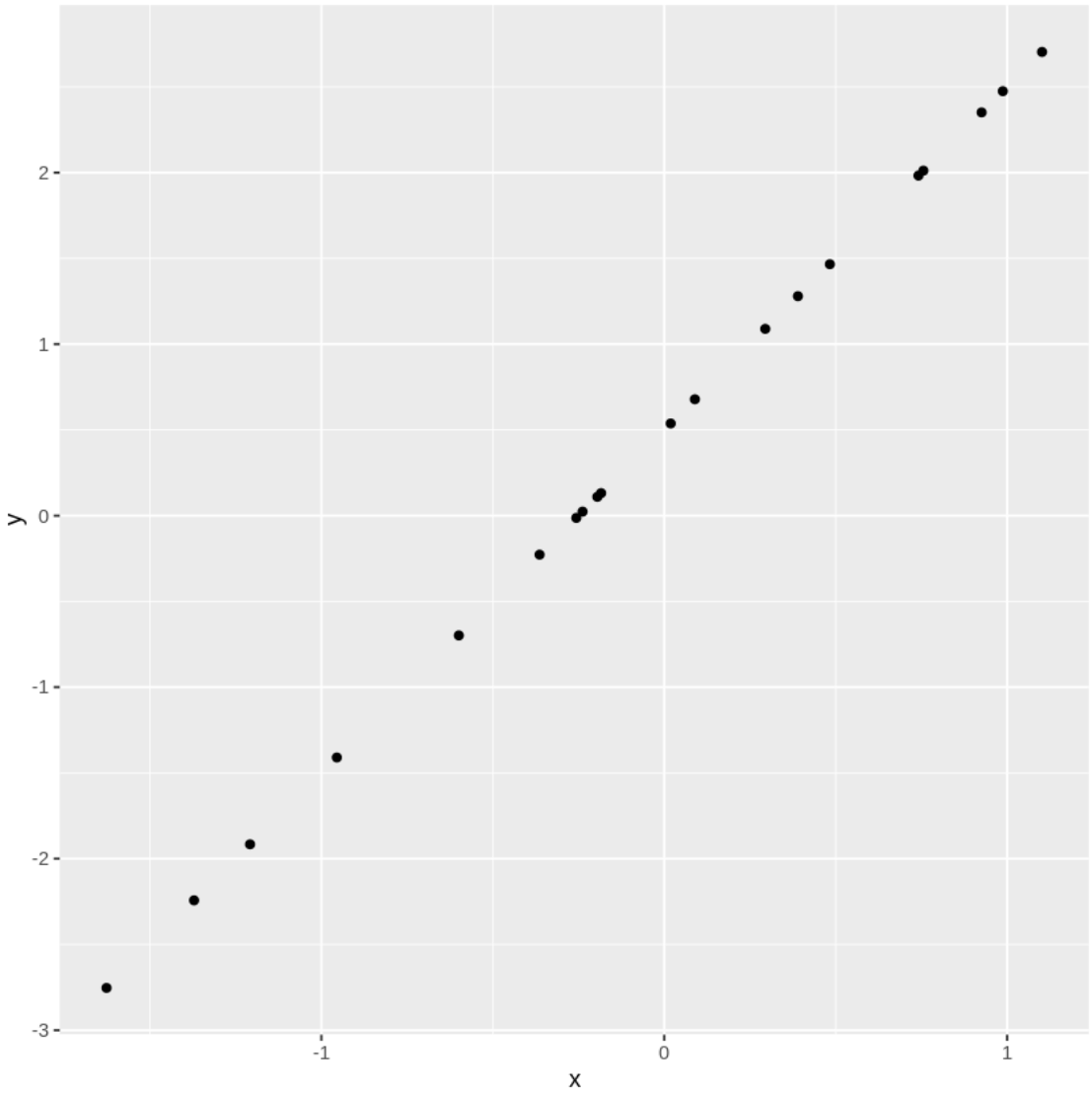
In the *Markdown* cell below the `R` cell, describe what you see in the plot.

Tip: You can generate your x-data *deterministically*, e.g., using either `a:b` syntax or the `seq()` function, or *randomly* using something like `runif()` or `rnorm()`. In practice, it won't matter all that much which one you choose.

[3]:
```
# Your Code Here

# set seed to get the same numbers each time
set.seed(10)
# create a set of 20 random numbers called x
x <- c(rnorm(20))
x
# set bo and b1 for the incercept and slope values
bo = 0.5
b1 = 2
# apply b0 and b1 values to generate y values from x
y <- bo + (b1*x)
y
# create a dataframe from x and y values
df_xy<-data.frame(x,y)
# plot the dataframe using ggplot
ggplot(df_xy, aes(x=x,y=y))+geom_point()
```

1. 0.0187461709418264 2. -0.184252542069064 3. -1.37133054992251 4. -0.599167715783718 5. 0.294545126567508 6. 0.389794300700167 7. -1.20807617542949 8. -0.363676017470862 9. -1.62667268170309 10. -0.256478394123992 11. 1.10177950308713 12. 0.755781508027337 13. -0.238233556018718 14. 0.98744470341339 15. 0.741390128383824 16. 0.0893472664958216 17. -0.954943856152377 18. -0.195150384667239 19. 0.92552126209408 20. 0.482978524836611

1. 0.537492341883653 2. 0.131494915861873 3. -2.24266109984502 4. -0.698335431567436 5. 1.08909025313502 6. 1.27958860140033 7. -1.91615235085897 8. -0.227352034941725 9. -2.75334536340618 10. -0.0129567882479833 11. 2.70355900617426 12. 2.01156301605467 13. 0.023532887962563 14. 2.47488940682678 15. 1.98278025676765 16. 0.678694532991643 17. -1.40988771230475 18. 0.109699230665522 19. 2.35104252418816 20. 1.46595704967322



2.1 YOUR ANSWER:

In the plot, I see that the dots represent each x value that I've created using rnorm() and the y values associated with each x value using the applied linear equation.

They form a straight line with a slope of b1=2, and the y intercept at x=0 is b0=0.5.

1. (b) The Error Component That is a perfect set of data points, but that is a problem in itself. In almost any real life situation, when we measure data, there will be some error in those measurements. Recall that our simple linear model is of the form:

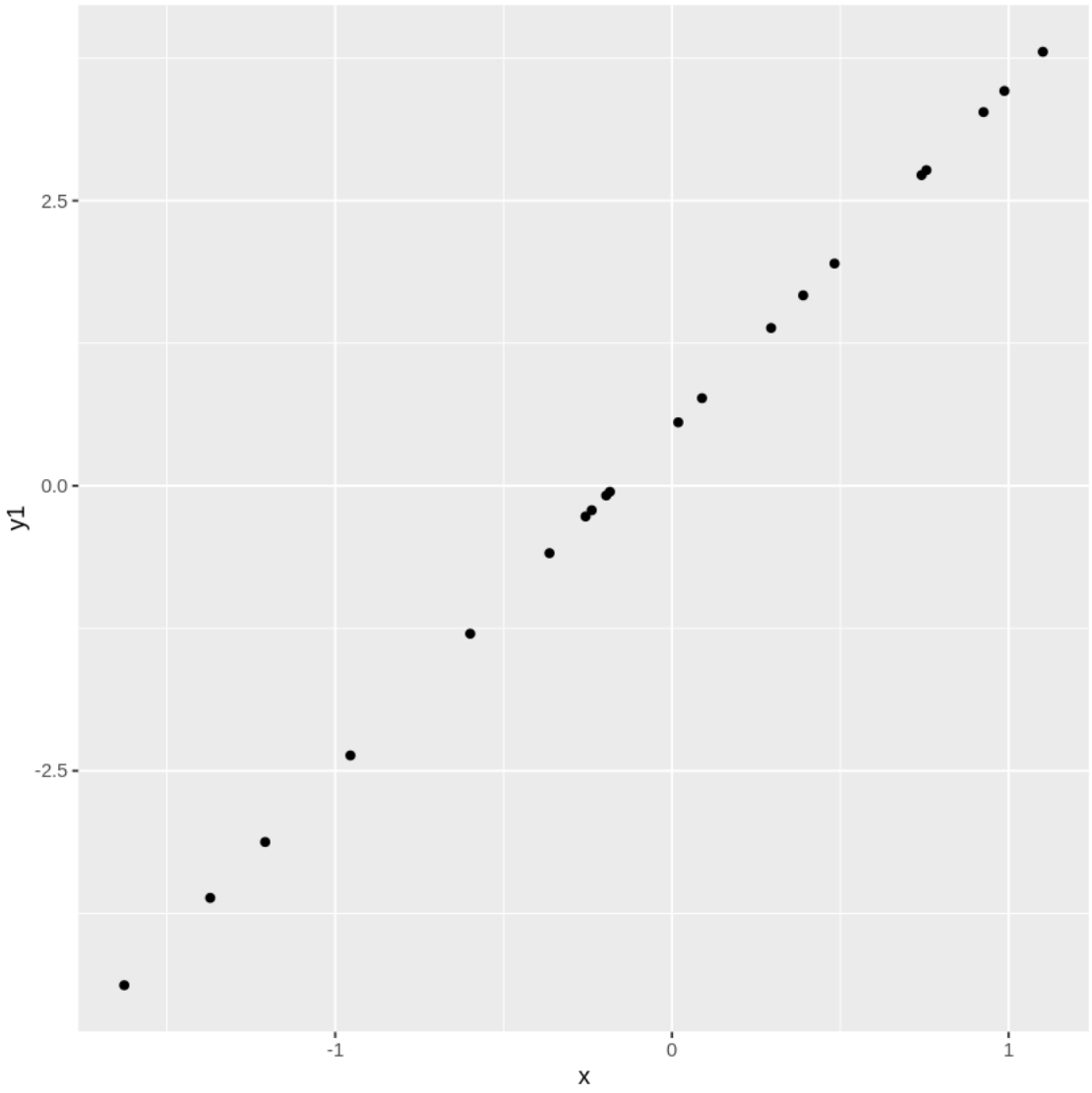
$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i, \qquad \epsilon_i \sim N(0, \sigma^2)$$

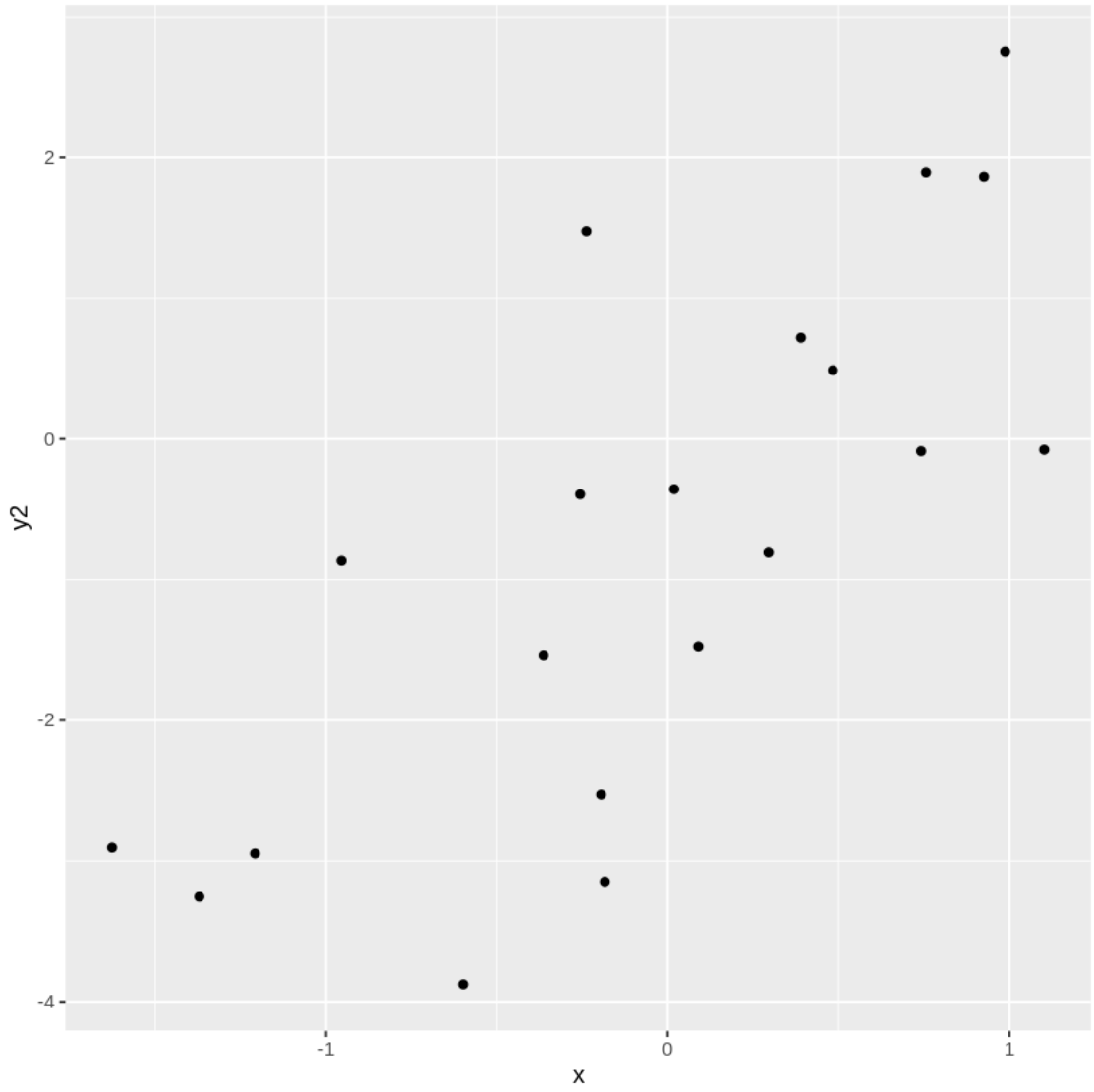
Add an error term to your y-data following the formula above. Plot at least three different plots (using ggplot!) with the different values of σ^2.

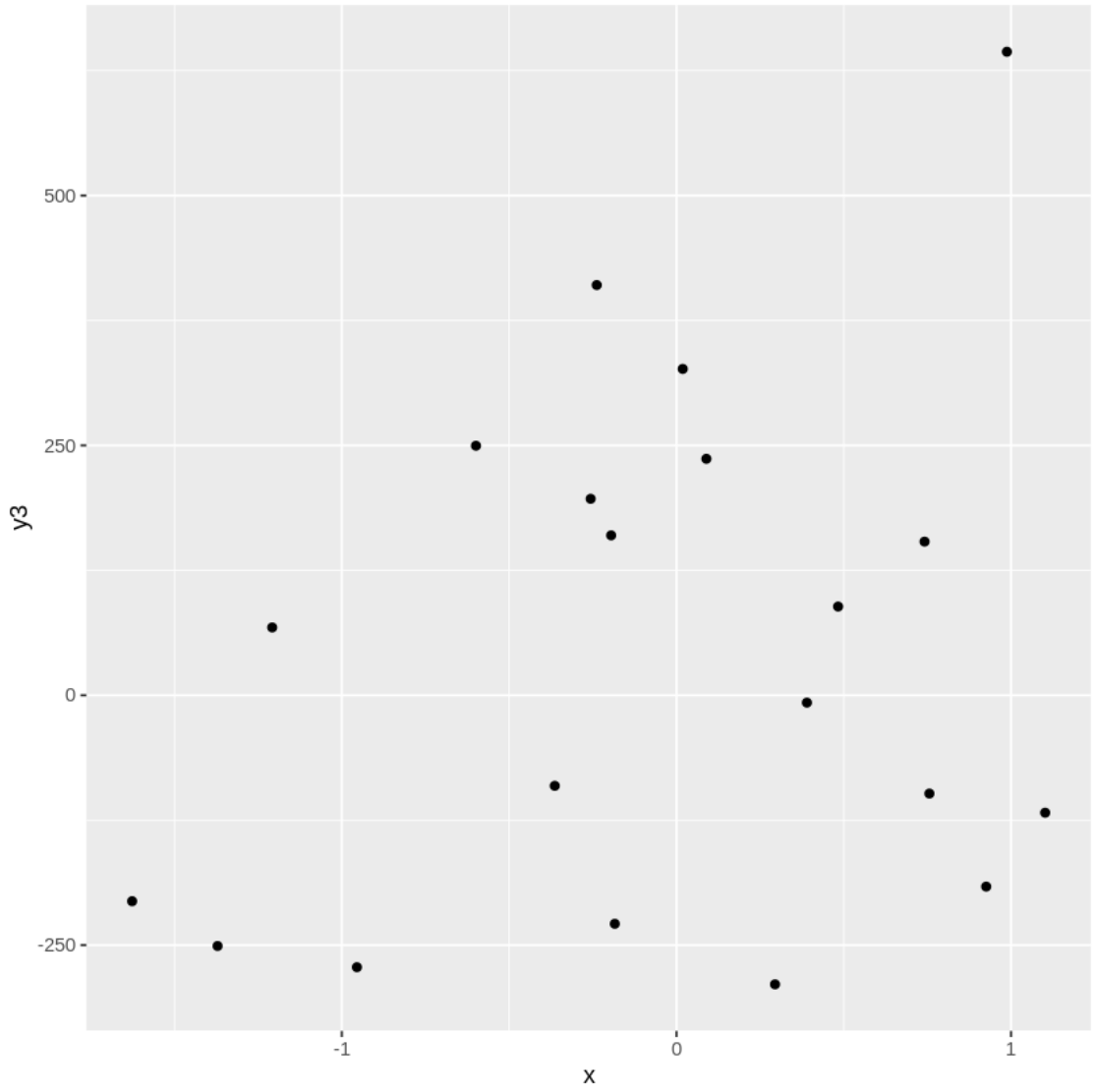
How does the value of σ^2 affect the final data points? Type your answer in the *Markdown* cell below the `R` cell.

Tip: To randomly sample from a normal distribution, check out the `rnorm()` function.

[7]:
```r
# Your Code Here

# setting seed to get the same values each time
set.seed(10)

# creating a set of 20 error values from a normal distribution
# having variance values of 1^2, 1.5^2, and 300^2
error1 = rnorm(20, sd=1)
error2 = rnorm(20, sd=1.5)
error3 = rnorm(20, sd=300)


# apply b0 and b1 values to generate y values from x
y1 <- bo + (b1*x) + error1
y2 <- bo + (b1*x) + error2
y3 <- bo + (b1*x) + error3

# create a dataframe from x and y values
df_xy1<-data.frame(x,y1)
df_xy2<-data.frame(x,y2)
df_xy3<-data.frame(x,y3)

# plot the dataframe using ggplot
ggplot(df_xy1, aes(x=x,y=y1))+geom_point()
ggplot(df_xy2, aes(x=x,y=y2))+geom_point()
ggplot(df_xy3, aes(x=x,y=y3))+geom_point()
```


2.5
0.0
-2.5
y1
-1
0
1
x


2
0
-2
-4
y2
-1
0
1
x



2.2 YOUR ANSWER:

Having a small variance in the error values that are sampled and added to the linear equation results in individual data points that resemble a straight line, with little variance how much these points deviate from the line.

However, as the variance term gets larger, the error becomes larger, and the each data point deviates less consistently and farther from the line.

3 Problem 2: The Effects of Variance on Linear Models

Once you've completed **Problem 1**, you should have three different "datasets" from the same underlying data function but with different variances. Let's see how those variance affect a best fit line.

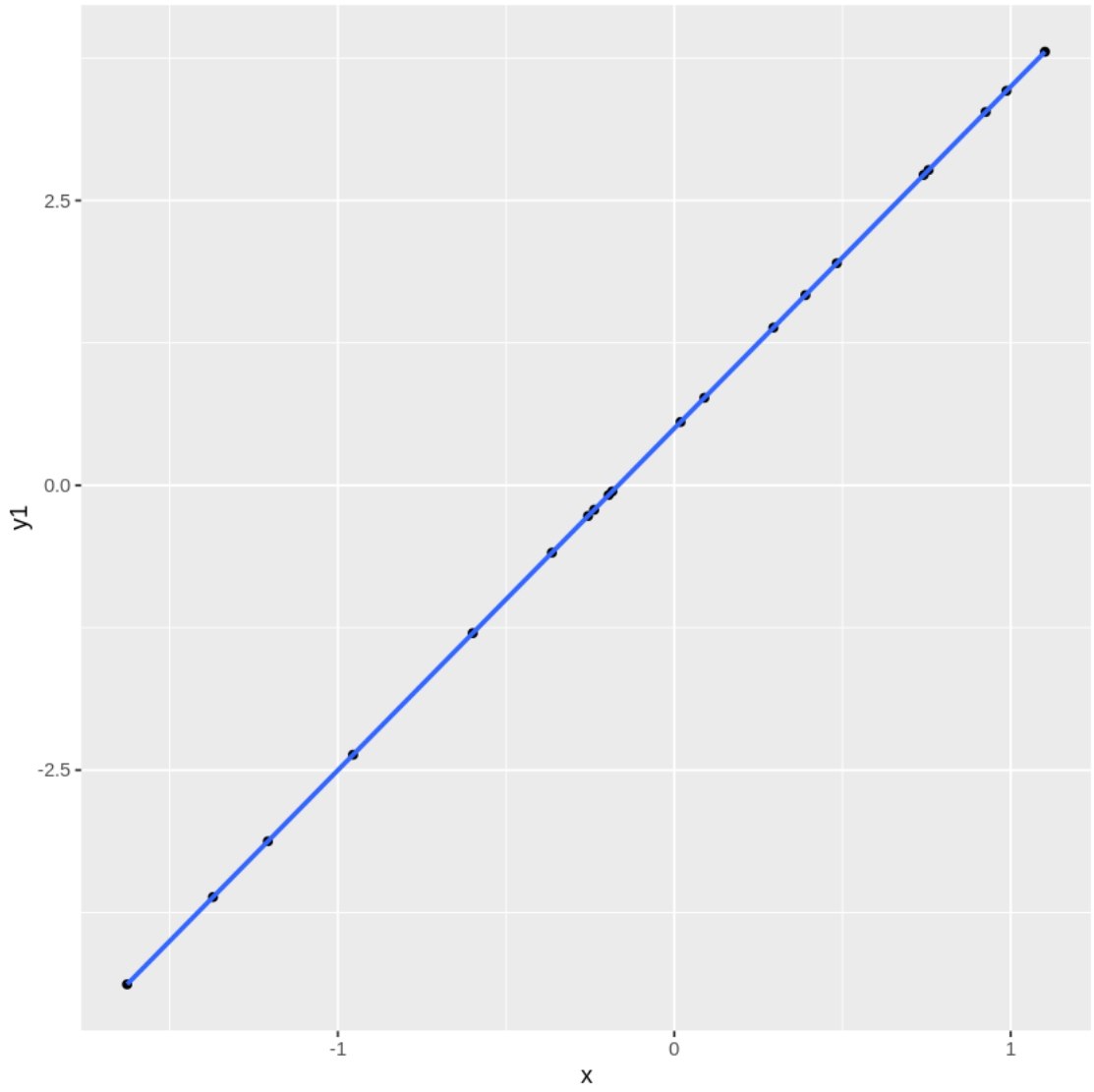
Use the `lm()` function to fit a best-fit line to each of those three datasets. Add that best fit line to each of the plots and report the slopes of each of these lines.

Do the slopes of the best-fit lines change as σ^2 changes? Type your answer in the *Markdown* cell below the `R` cell.

Tip: The `lm()` function requires the syntax `lm(y~x)`.

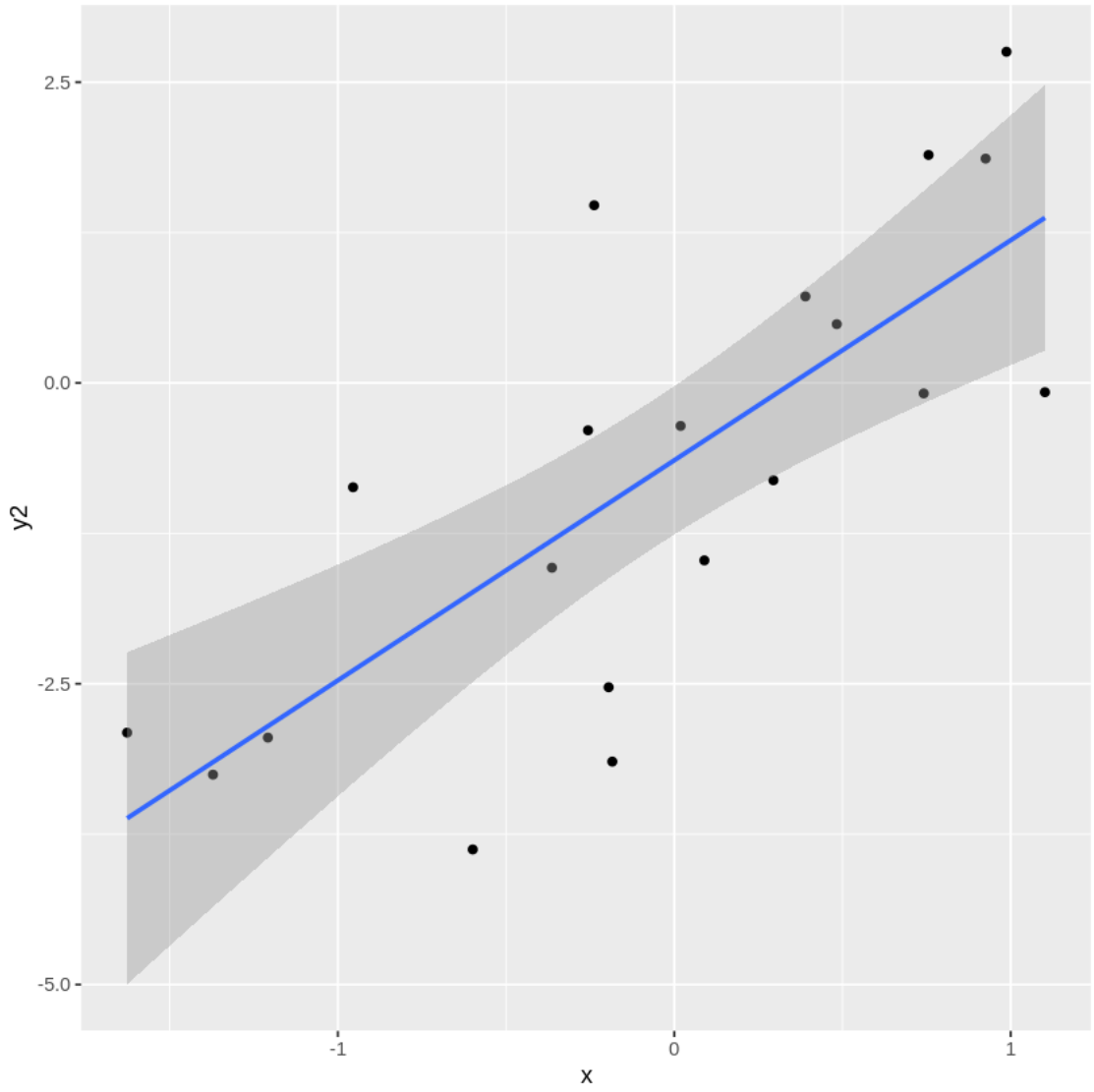
[36]:
```
# Your Code Here

# using the lm() method to create a smooth line that best-fits three datasets
ggplot(df_xy1, aes(x=x,y=y1)) + geom_point() + geom_smooth(method = 'lm')
lm(y1~x)

ggplot(df_xy2, aes(x=x,y=y2)) + geom_point() + geom_smooth(method = 'lm')
lm(y2~x)

ggplot(df_xy3, aes(x=x,y=y3)) + geom_point() + geom_smooth(method = 'lm')
lm(y3~x)
```

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


Call:
lm(formula = y1 ~ x)

Coefficients:
(Intercept)            x
        0.5          3.0



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

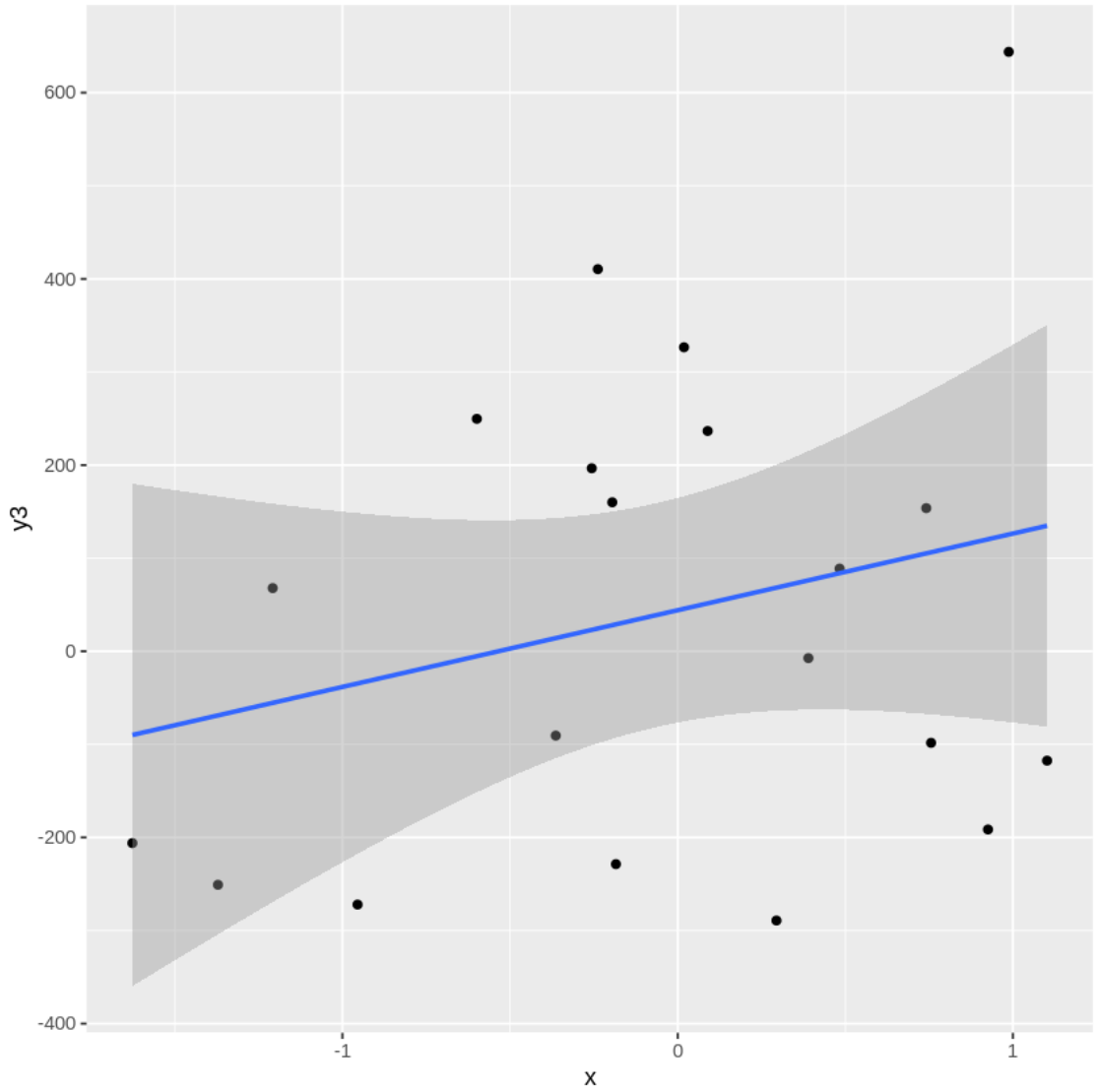



```
Call:
lm(formula = y2 ~ x)

Coefficients:
(Intercept)            x
    -0.6426       1.8291
```

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




```
Call:
lm(formula = y3 ~ x)

Coefficients:
(Intercept)            x
      44.08         82.36
```



3.1 YOUR ANSWER:

Yes, the slope changes as the variance changes. With a small variance, the slope is close to the actual slope of the equation, which is 3. With a large variance, the slope is very far from the actual slope of the equation, which means that as the variance of the dataset increases, it becomes a less effective way to estimate the slope.

4 Problem 3: Interpreting the Linear Model

Choose one of the above three models and write out the actual equation of that model. Then in words, in the *Markdown* cell below the `R` cell, describe how a 1 unit increase in your predictor

affects your response. Does this relationship make sense?

[19]:
```
# Your Code Here
model_y2 = -0.6426 + (1.8291*x) + error2
```

4.1 YOUR ANSWER:

A 1 unit increase in my predictor affects my response by the slope value, which is 1.8291 + its error and intercept value. This relationship makes sense because it follows by the linearity assumption in linear regression, as well as homoskedasticity, since the variance has been sampled from a normal distribution.

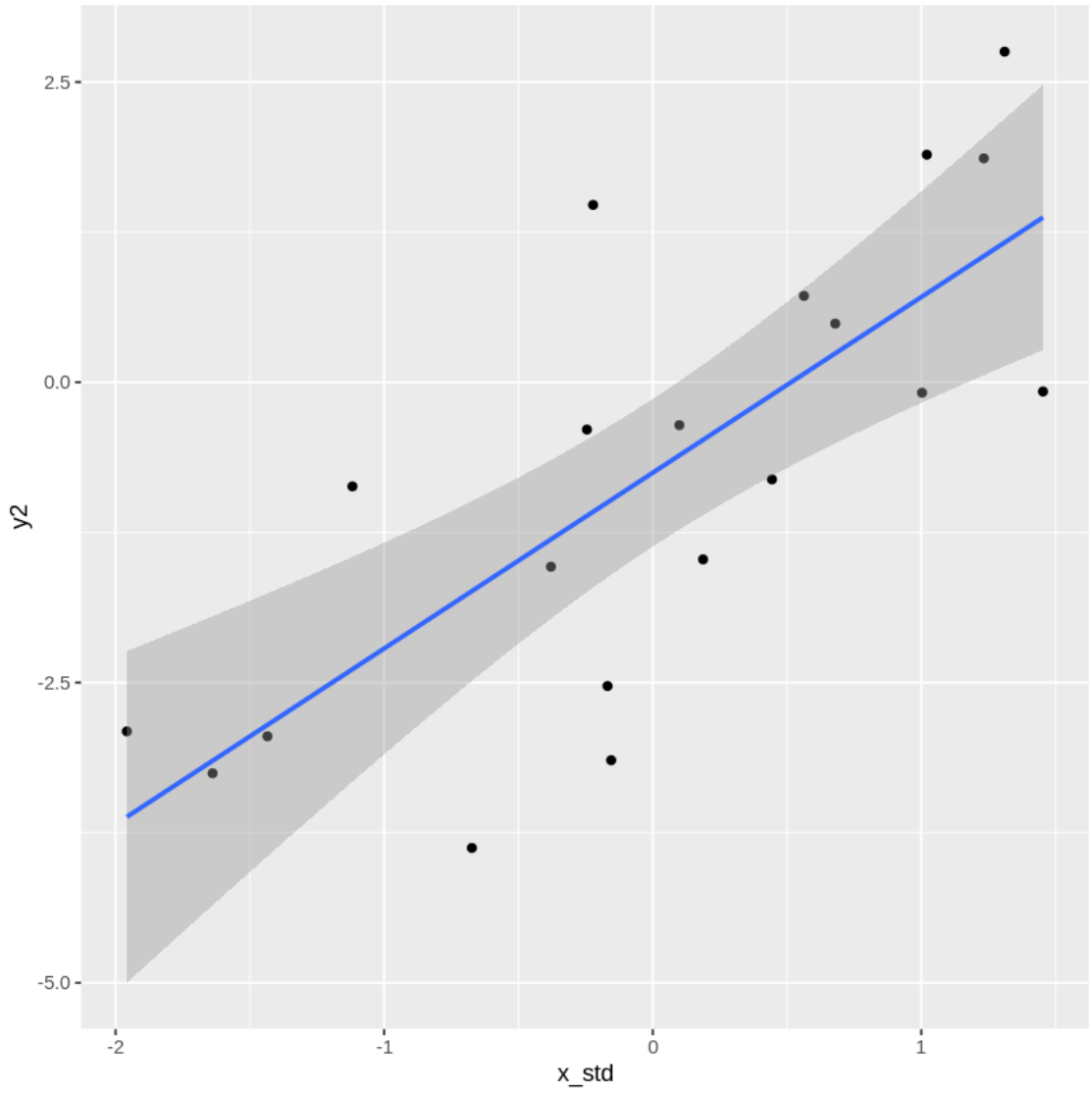
5 Problem 4: The Effects of Standardizing Data

We spent some time standardizing data in the autograded assignment. Let's do that again with your simulated data.

Using the same model from **Problem 3**, standardize your simulated predictor. Then, using the `lm()` function, fit a best fit line to the standardized data. Using ggplot, create a scatter plot of the standardized data and add the best fit line to that figure.

[53]:
```
# Your Code Here

# standardizing the predictor, usin the y2 model
model_y2_standardized <-
  df_xy2 %>%
  mutate(x_std = scale(x))

# plot the model using ggplot
ggplot(model_y2_standardized, aes(x=x_std,y=y2)) + geom_point() +␣
 ↪geom_smooth(method = 'lm')

lm(y2 ~ scale(x))
```

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


Call:
lm(formula = y2 ~ scale(x))

Coefficients:
(Intercept)     scale(x)
    -0.7533       1.4631
```



6 Problem 5: Interpreting the Standardized Model

Write out the expression for your standardized model. In words, in the *Markdown* cell below the `R` cell, describe how a 1 unit increase in your standardized predictor affects the response. Is this value different from the original model? If yes, then what can you conclude about interpretation of standardized predictors vs. unstandardized predictors.

[54]:
```
# Your Code Here

model_y2_standardized = -0.7533 + (1.4631*x) + error2
```

6.1 YOUR ANSWER:

The standardization of the model changes the predictor and response so that a 1 unit increase in the predictor increases the response by 1 standard deviation, rather than the actual slope and error of the raw data. Unstandardized predictors give us the data as they are collected, so it gives us the effect of individual effect of the predictor to the response.

When predictors are standardized, the response become unit-less. The change in one standard deviation of the predictor is associated with the response. This is helpful in observing the overall effect of the predictors on the response.

[ ]: