C1M1_peer_reviewed

June 26, 2021

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.

[38]:
```
# 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.

[39]:
```
set.seed(123)

x_values <- round(runif(13, min = 0, max = 25))
x_values

beta_naught <- round(runif(1, min = 1, max = 5))
beta_naught

beta_one <- round(runif(1, min = 1, max = 3))
beta_one

y_values <- beta_naught + beta_one * x_values
y_values

simple_line <- data.frame(x_values, y_values)

ggplot(simple_line, aes(x = x_values, y = y_values)) +
    geom_point() +
    geom_smooth(method = "lm", se=FALSE, color="chartreuse")
```

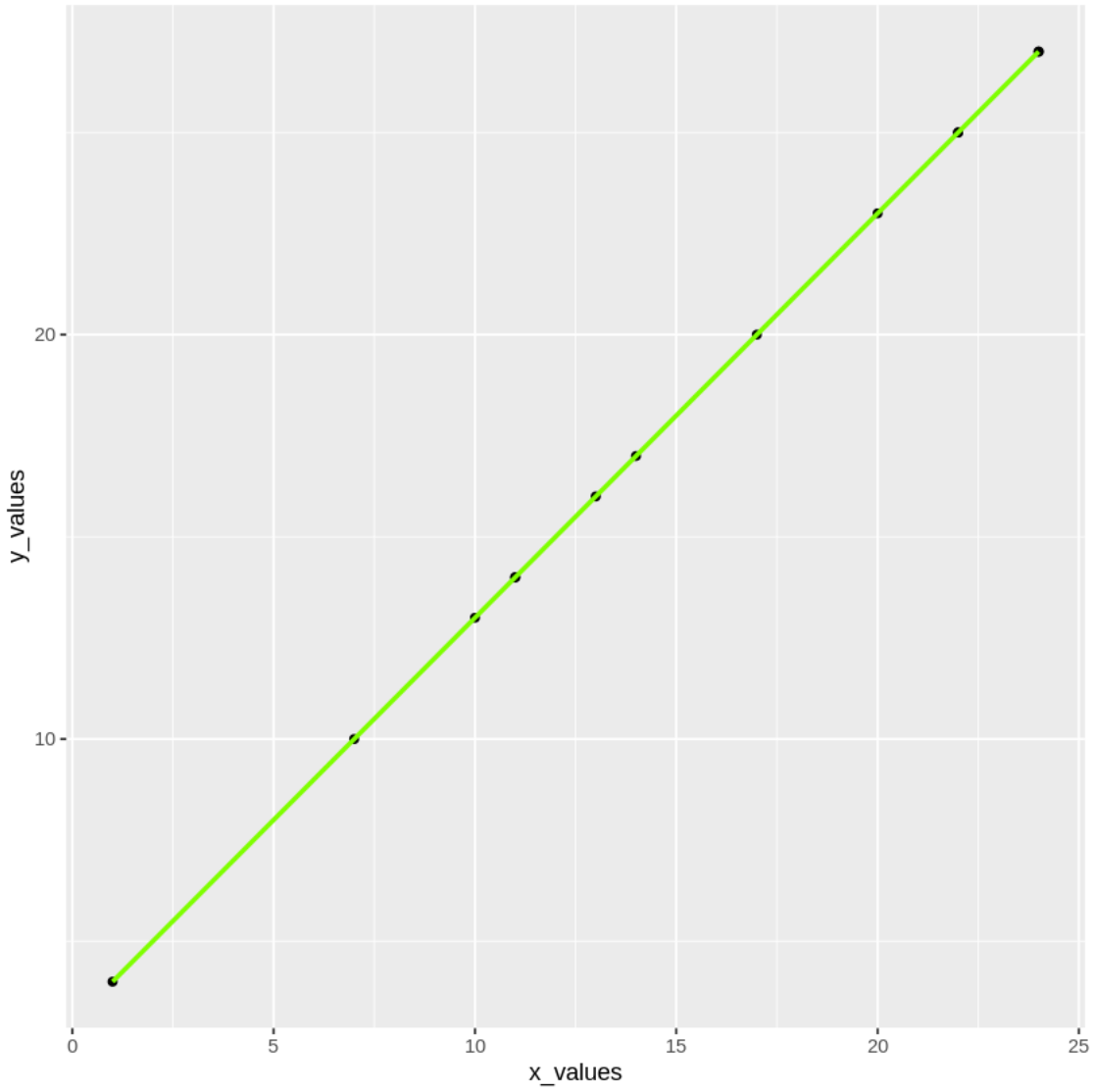
1. 7 2. 20 3. 10 4. 22 5. 24 6. 1 7. 13 8. 22 9. 14 10. 11 11. 24 12. 11 13. 17

3

1

1. 10 2. 23 3. 13 4. 25 5. 27 6. 4 7. 16 8. 25 9. 17 10. 14 11. 27 12. 14 13. 20

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



The graph contains 13 randomly generated data points. I added a line of best fit to show that the data points are perfectly linear. Everything is perfectly aligned because there is no error term ϵ_i added to the linear equation yet.

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.

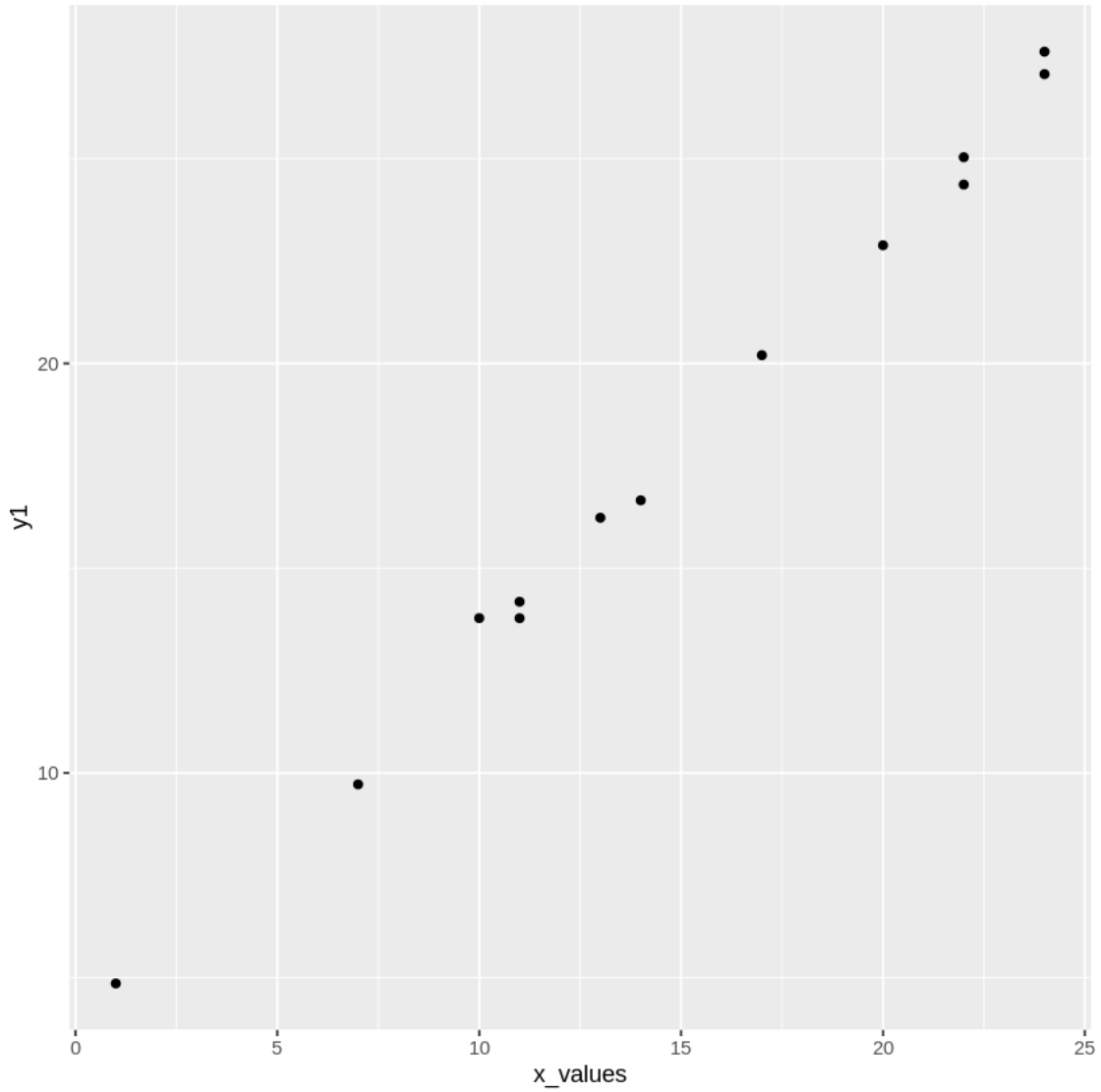
[40]:
```
set.seed(123)

error_small = rnorm(13, mean = 0, sd = 0.5)
error_small

error_medium = rnorm(13, mean = 0, sd = 2)
error_medium

error_large = rnorm(13, mean = 0, sd = 5)
error_large

y1 <- y_values + error_small
y2 <- y_values + error_medium
y3 <- y_values + error_large

small <- data.frame(x_values, y1)
medium <- data.frame(x_values, y2)
large <- data.frame(x_values, y3)

small_plot <- ggplot(small, aes(x = x_values, y = y1)) +
    geom_point()
small_plot

medium_plot <- ggplot(medium, aes(x = x_values, y = y2)) +
    geom_point()
medium_plot

large_plot <- ggplot(large, aes(x = x_values, y = y3)) +
    geom_point()
large_plot
```

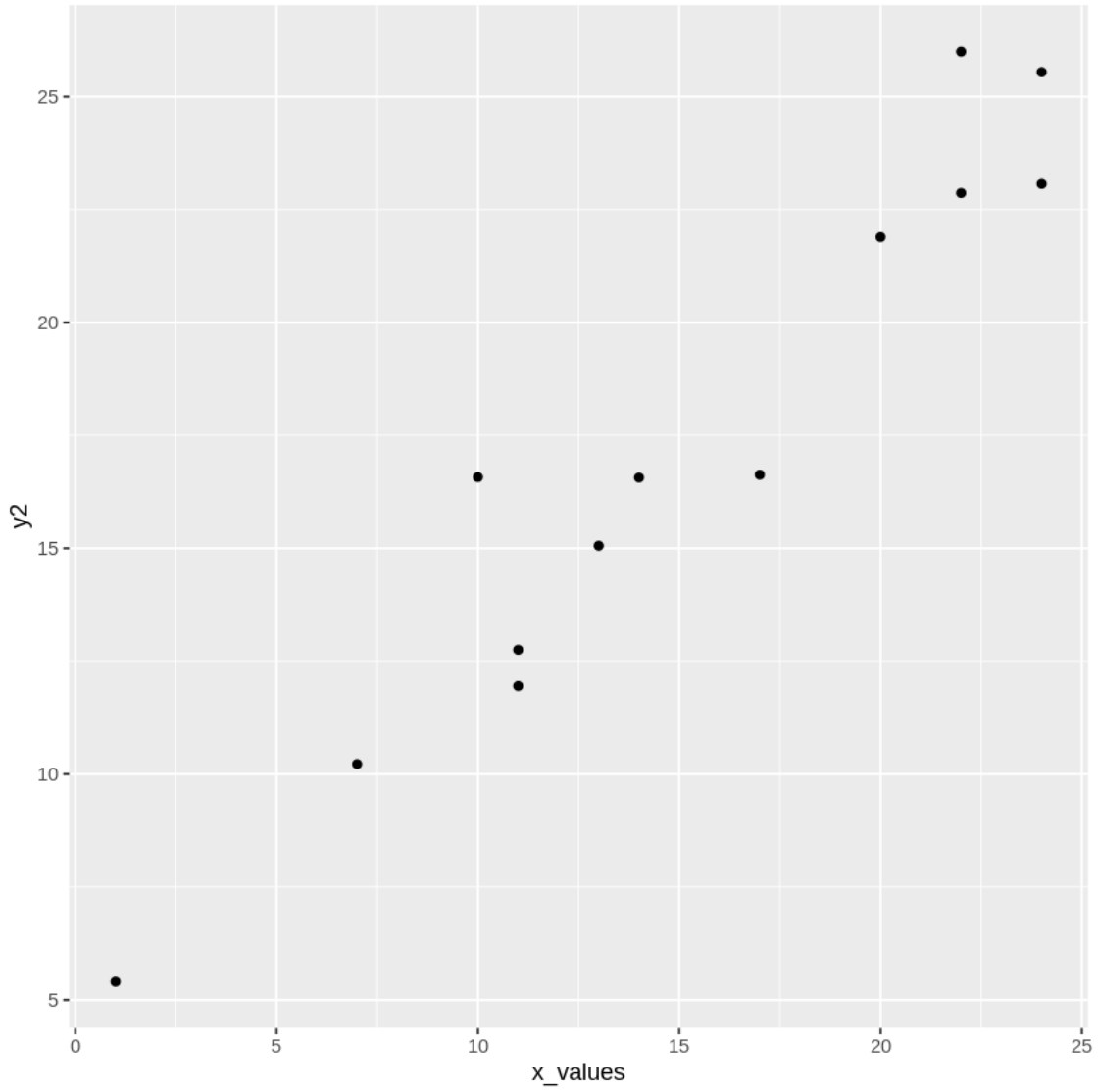
1. -0.280237823276106 2. -0.11508874474164 3. 0.779354157074562 4. 0.035254195712288
5. 0.0646438675804731 6. 0.857532493441641 7. 0.230458102994601 8. -0.632530617303267
9. -0.343426425946763 10. -0.222830985049979 11. 0.612040898719731 12. 0.179906913528682
13. 0.200385725297026

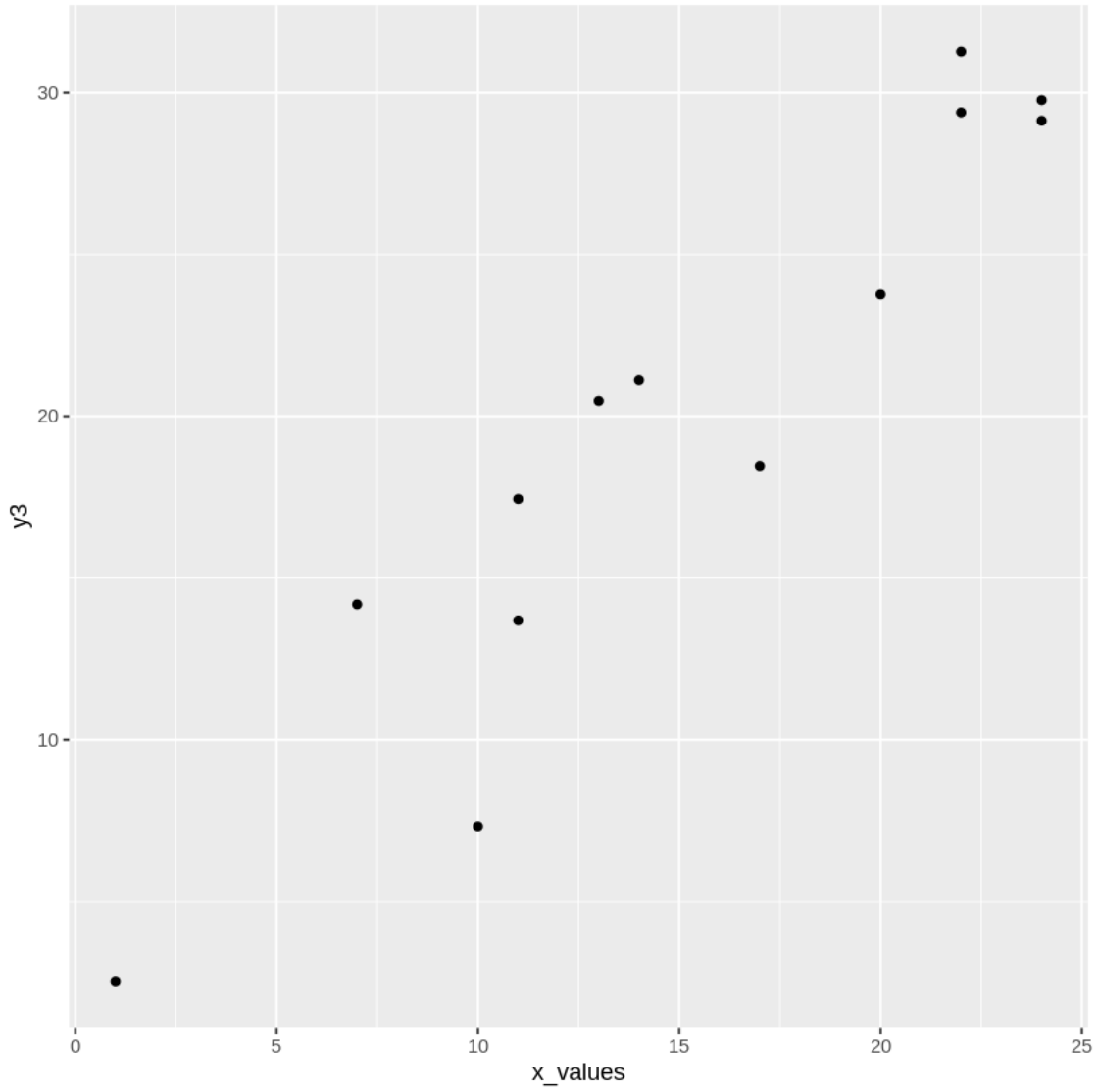
1. 0.221365431890239 2. -1.11168226950815 3. 3.57382627360616 4. 0.995700956458479
5. -3.93323431325928 6. 1.40271180312737 7. -0.945582815455868 8. -2.13564741197369
9. -0.43594982931659 10. -2.05200889661448 11. -1.45778245858228 12. -1.25007853569851
13. -3.37338662148483

1. 4.18893522247262 2. 0.766865589182576 3. -5.69068468505974 4. 6.26907460534963
5. 2.13232110738407 6. -1.47535741496136 7. 4.47562830522511 8. 4.39066743766521
9. 4.10790540818744 10. 3.44320127050046 11. 2.76958826768794 12. -0.309558552883608

13. -1.52981331869958




y2
25
20
15
10
5
0
5
10
15
20
25
x_values



The value of σ^2, known as variance, will determine how spread out the data points are from the original perfect linear equation. The larger the value of the variance, the higher each data point's residual value will be on average. Graphically, one can easily see that the dataset with a small variance and thus a small error term still has the points clustered fairly closely to the original line. The points in the dataset with the large variance and thus large error term are a lot more spread out from the original dataset with no error term.

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.

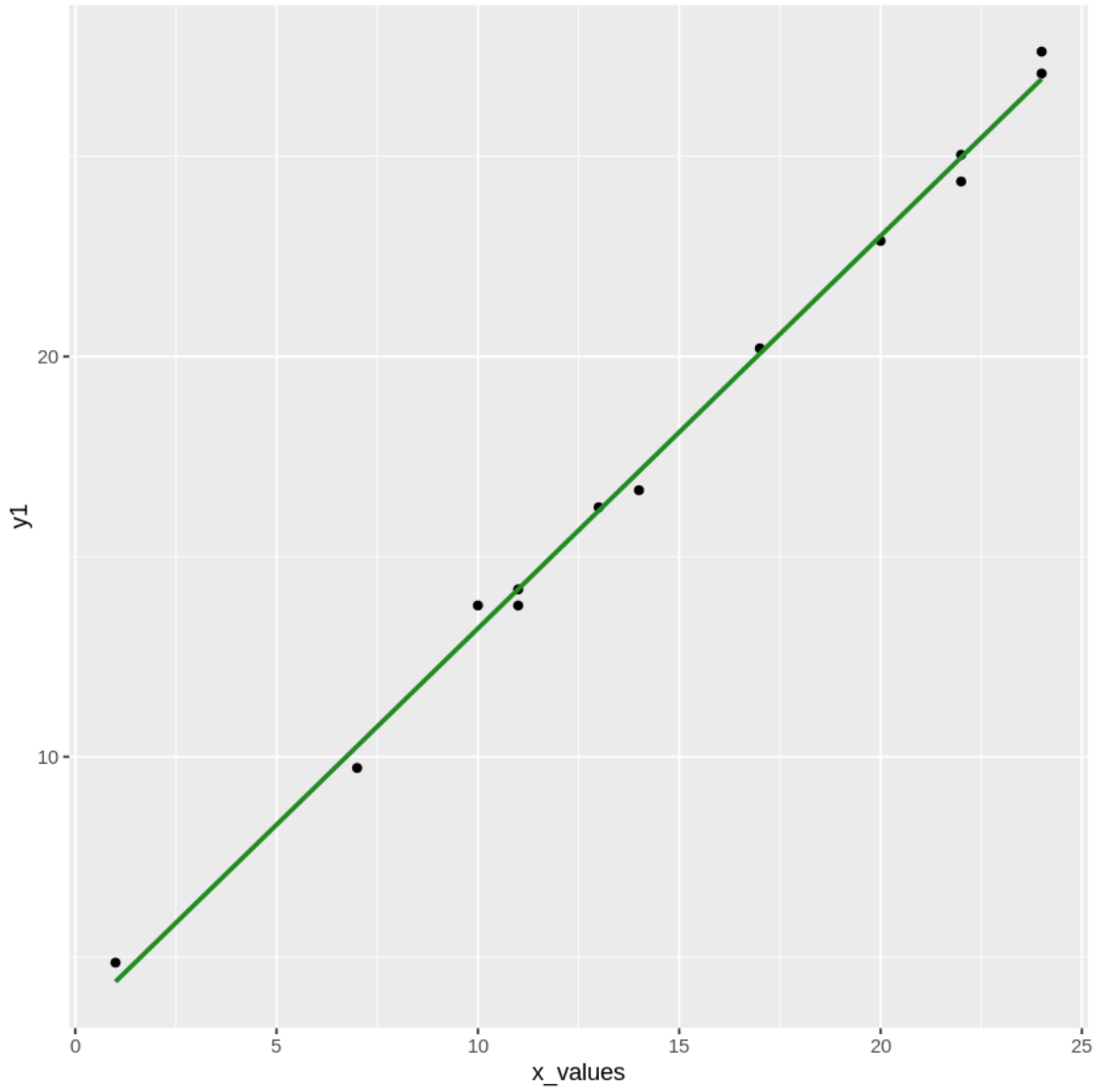
Tip: The `lm()` function requires the syntax `lm(y~x)`.

[41]:
```
small_slope <- lm(y1 ~ x_values, data=small)$coefficients[2]
medium_slope <- lm(y2 ~ x_values, data=small)$coefficients[2]
large_slope <- lm(y3 ~ x_values, data=small)$coefficients[2]

small_plot <- small_plot +
                geom_smooth(method = "lm", se=FALSE, color="forestgreen")
small_plot
small_slope

medium_plot <- medium_plot +
                geom_smooth(method = "lm", se=FALSE, color="darkorchid")
medium_plot
medium_slope

large_plot <- large_plot +
                geom_smooth(method = "lm", se=FALSE, color="firebrick1")
large_plot
large_slope
```

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

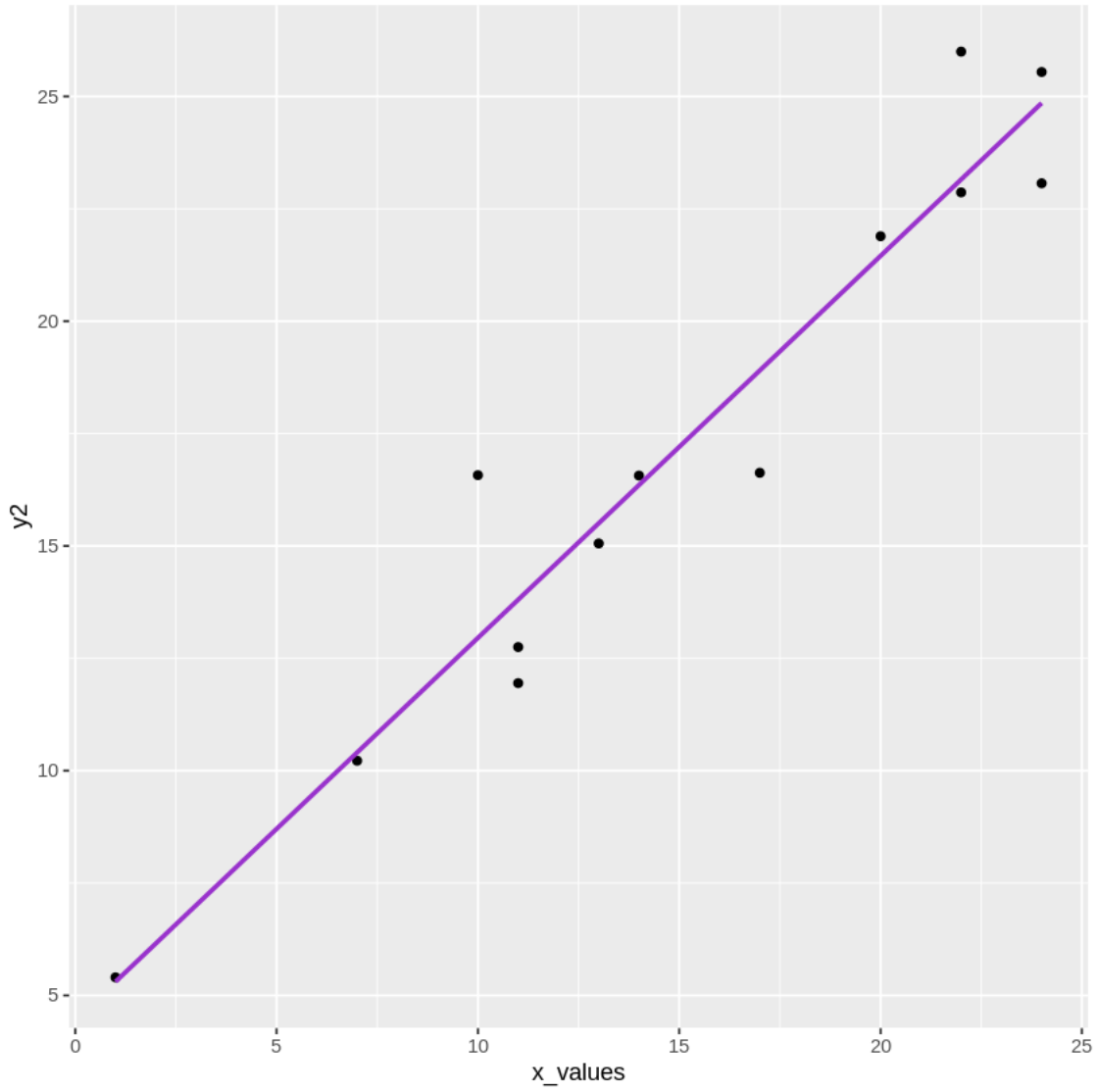
x__values: 0.980395901053497

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

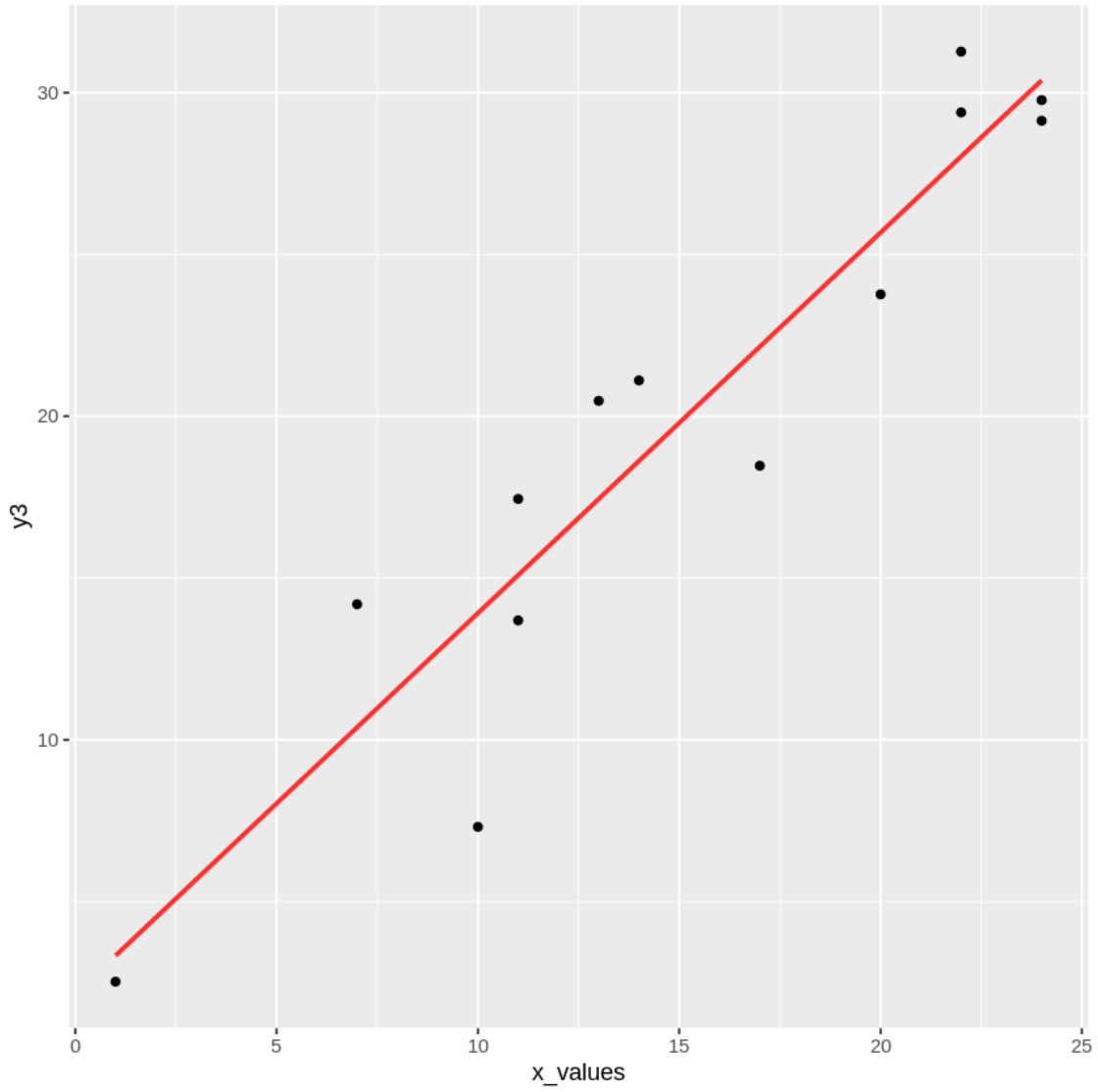


x__values: 0.849835941305201

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



x_values: 1.1762948917052



In practice, the slopes of the best fit lines do change with the variance, but the slope does not necessarily increase or decrease. The change in slope is tied to the stochastic nature of the randomly generated error terms. Because the error terms are random, the slope could be greater than the slope of the data with no error (as seen in the large error dataset), the slope could be smaller than the slope of the data with no error (as seen in the medium dataset), or the slope could remain exactly the same as the slope of the data with no error.

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?

[42]:
```
small_mod <- lm(y1 ~ x_values, data=small)
summary(small_mod)

cat("Actual equation: y_i = ", small_mod$coefficients[1], " + ",␣
 ↪small_mod$coefficients[2], "* x_i + error_i")
```

```
Call:
lm(formula = y1 ~ x_values, data = small)

Residuals:
     Min       1Q   Median       3Q      Max
-0.60185 -0.40779  0.06594  0.13454  0.68193

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.4006     0.2948   11.53 1.75e-07 ***
x_values      0.9804     0.0178   55.07 8.73e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.44 on 11 degrees of freedom
Multiple R-squared:  0.9964,Adjusted R-squared:  0.9961
F-statistic:  3033 on 1 and 11 DF,  p-value: 8.725e-15



Actual equation: y_i =  3.400605  +  0.9803959 * x_i + error_i
```

For the model of the small error dataset, a 1 unit increase in the predictor (x_i) corresponds to a ~0.9804 (slope) unit average increase in the response (y_i). This relationship makes sense because slope in general is representative of how a change in one variable relates to the change in another variable.

5 Problem 4: The Effects of Standardizing Data

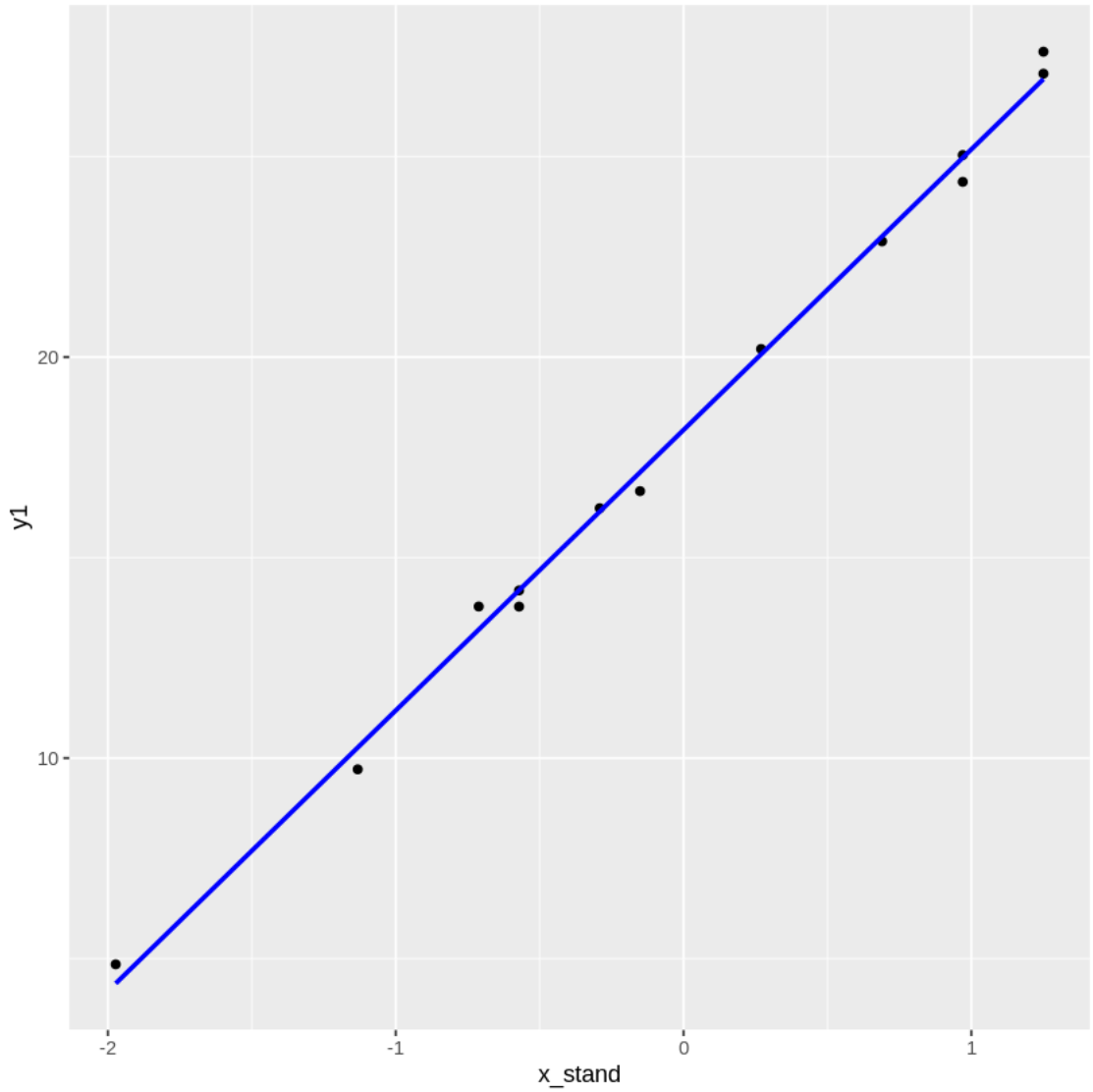
We spent some time standardizing data in the autograded assignment. Let's do that again with your simulated data.

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.

[43]:
```
small$x_stand = (small$x_values - mean(small$x_values)) / sd(small$x_values)

stand_mod <- lm(y1 ~ x_stand, data=small)
summary(stand_mod)

test <- ggplot(small, aes(x = x_stand, y = y1)) +
```

```
    geom_point() +
    geom_smooth(method = "lm", se=FALSE, color="blue")

test
```

```
Call:
lm(formula = y1 ~ x_stand, data = small)

Residuals:
     Min       1Q   Median       3Q      Max
-0.60185 -0.40779  0.06594  0.13454  0.68193

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   18.182      0.122  148.99  < 2e-16 ***
x_stand        6.995      0.127   55.07 8.73e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.44 on 11 degrees of freedom
Multiple R-squared:  0.9964,Adjusted R-squared:  0.9961
F-statistic:  3033 on 1 and 11 DF,  p-value: 8.725e-15


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



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.

[44]:
```
cat("Actual equation: y_i = ", stand_mod$coefficients[1], " + ",␣
↪stand_mod$coefficients[2], "* x_stand_i + error_i")
```

```
Actual equation: y_i =  18.18196  +  6.995264 * x_stand_i + error_i
```

A one unit increase in the standardized predictor (x_stand_i) corresponds to a ~6.99 (slope) unit average increase in the response (y_i). This value is different from the original model; the original model involved changing an unstandardized predictor by one unit but increasing a standardized predictor by one unit is effectively increasing the response by a one *standard deviation* increase in the predictor.