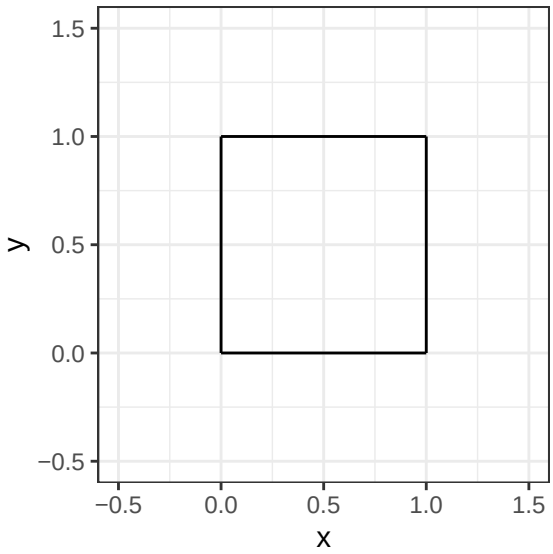
Lab Three – Matrices and Data Frames in R

- Complete the tasks below. Make sure to start your solutions in on a new line that starts with "**Solution**:".
- Make sure to use the Quarto Cheatsheet. This will make completing and writing up the lab *much* easier.

Consider the unit square depicted in Figure 1.

```
p0 <- c(0, 0)
p1 <- c(1, 0)
p2 <- c(1, 1)
p3 <- c(0, 1)
ggplot() +
  geom_segment(aes(x = p0[1], y = p0[2], xend = p1[1], yend = p1[2])) +
  geom_segment(aes(x = p1[1], y = p1[2], xend = p2[1], yend = p2[2])) +
  geom_segment(aes(x = p2[1], y = p2[2], xend = p3[1], yend = p3[2])) +
  geom_segment(aes(x = p3[1], y = p3[2], xend = p0[1], yend = p0[2])) +
  theme_bw() +
  xlim(-0.5, 1.5) +
  ylim(-0.5, 1.5) +
  labs(x = "x", y = "y")
```

Figure 1: The unit square.



The unit square can be defined by two basis vectors

$$\mathbf{v}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

From these vectors, we can compute the corner vertices.

$$\begin{aligned}
p_0 &= (0,0) = (x_0, y_0) && \text{(The origin)} \\
p_1 &= v_1 = (x_1, y_1) = (1,0) && \text{(The first basis vector)} \\
p_2 &= v_1 + v_2 = (x_2, y_2) = (1,1) && \text{(The sum of basis vectors)} \\
p_3 &= v_2 = (x_3, y_3) = (0,1) && \text{(The second basis vector)}
\end{aligned}$$

Note that the segments that form the boundary go from p_0 to p_1 to p_2 to p_3 and back to p_0.

Consider the matrix M to be the matrix we get from binding the two basis vectors at the column,

$$\mathbf{M} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

1 Question 1

Let's transform $\mathbf{M}$ to demonstrate an interesting linear algebra property.

1.1 Part a

Create the matrix `A` in `R`.

$$\mathbf{A} = \begin{bmatrix} 1 & -7 \\ 5 & 9 \end{bmatrix}$$

Solution

```
A = matrix(data = c(1, -7, 5, 9), nrow = 2, ncol = 2, byrow = TRUE)
A
```

```
     [,1] [,2]
[1,]    1   -7
[2,]    5    9
```

```
#I made a matrix that went by row instead of by column.
```

1.2 Part b

Compute the product below in `R` and store it in an object `T`.

$$\mathbf{T} = \mathbf{AM}.$$

Solution

```
M = matrix(data = c(1, 0, 0, 1), nrow = 2, ncol = 2, byrow = TRUE) # Making the matrix, M
T = A%*%M #Multiplying the matrices and naming them T
T
```

```
     [,1] [,2]
[1,]    1   -7
[2,]    5    9
```

1.3 Part c

Create `basis.vector.1` (first column of `T`) and `basis.vector.2` (second column of `T`) in `R`.

Solution

```
basis.vector.1 = c(1,5)
basis.vector.2 = c(-7,9) #I used a manual vector because there were only two data points.
basis.vector.1
```

```
[1] 1 5
```

```
basis.vector.2
```

```
[1] -7  9
```

1.4 Part d

Compute the points p_0, p_1, p_2, and p_3.

Solution

```
p0 = c(0,0)
p1 = basis.vector.1
p2 = basis.vector.1+basis.vector.2
p3 = basis.vector.2 #I used the formulas from the unit square example for the corner vertices.
p0
```

```
[1] 0 0
```

```r
p1
```

```
[1] 1 5
```

```r
p2
```

```
[1] -6 14
```

```r
p3
```

```
[1] -7  9
```

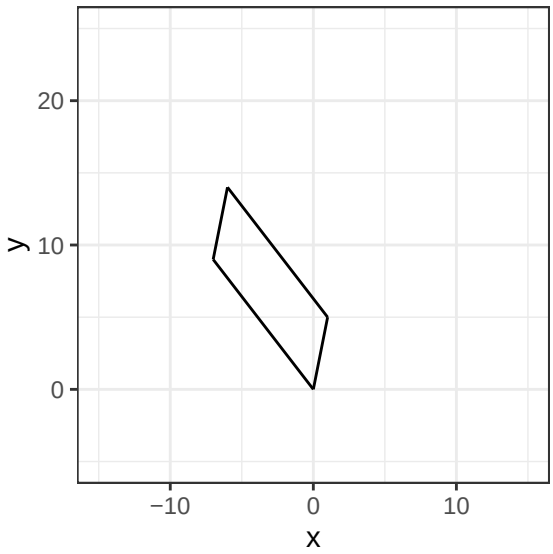
1.5 Part e

Copy and paste the code for the unit square. Edit the code to draw the segments that form the boundary based on the points in Part d.

Solution

```r
p0 = c(0,0)
p1 = basis.vector.1
p2 = basis.vector.1+basis.vector.2
p3 = basis.vector.2 # I made sure to use the same points as found in part d.
ggplot() +
  geom_segment(aes(x = p0[1], y = p0[2], xend = p1[1], yend = p1[2])) +
  geom_segment(aes(x = p1[1], y = p1[2], xend = p2[1], yend = p2[2])) +
  geom_segment(aes(x = p2[1], y = p2[2], xend = p3[1], yend = p3[2])) +
  geom_segment(aes(x = p3[1], y = p3[2], xend = p0[1], yend = p0[2])) +
  theme_bw() +
  xlim(-15, 15) +
  ylim(-5, 25) +
  labs(x = "x", y = "y")
```

Figure 2: Parallelogram 1.



1.6 Part f

Interestingly, the determinant of **A** gives us the area of the shape plotted in Part e. Find the determinant of **A**.

Solution

```r
det(A) #returns the determinant of A
```

```
[1] 44
```

2 Question 2

Recomplete Question 1 using a new matrix for **A**. This matrix is not as well behaved.

2.1 Part a

Create the matrix `A` in `R`.

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

Solution

```
A = matrix(data = c(1, 2, 2, 4), nrow = 2, ncol = 2, byrow = TRUE)
#I made a matrix that went by row instead of by column with new data.
A
```

```
     [,1] [,2]
[1,]    1    2
[2,]    2    4
```

2.2 Part b

Compute the product below in `R` and store it in an object `T`.

$$\mathbf{T} = \mathbf{AM}.$$

```
T = A %*% M # I made sure to multiply the matrices rather than their components.
T
```

```
     [,1] [,2]
[1,]    1    2
[2,]    2    4
```

2.3 Part c

Create `basis.vector.1` (first column of `T`) and `basis.vector.2` (second column of `T`) in `R`.

```
basis.vector.1 = c(1,2)
basis.vector.2 = c(2,4)
basis.vector.1
```

```
[1] 1 2
```

```
basis.vector.2
```

```
[1] 2 4
```

2.4 Part d

Compute the points p_0, p_1, p_2, and p_3.

Solution

```
p0 = c(0,0)
p1 = basis.vector.1
p2 = basis.vector.1+basis.vector.2
p3 = basis.vector.2 #same work as 1.4
p0
```

```
[1] 0 0
```

```
p1
```

```
[1] 1 2
```

```
p2
```

```
[1] 3 6
```

```
p3
```

```
[1] 2 4
```

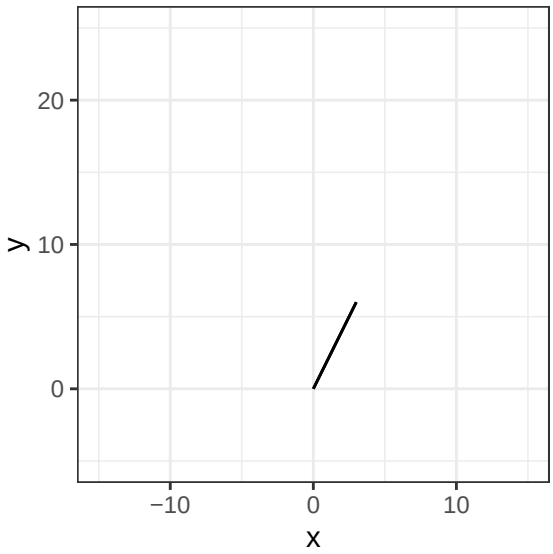
2.5 Part e

Copy and paste the code for plotting the unit square. Edit the code to draw the segments that form the boundary based on the points in Part d.

Solution

```r
p0 = c(0,0)
p1 = basis.vector.1
p2 = basis.vector.1+basis.vector.2
p3 = basis.vector.2
ggplot() +
  geom_segment(aes(x = p0[1], y = p0[2], xend = p1[1], yend = p1[2])) +
  geom_segment(aes(x = p1[1], y = p1[2], xend = p2[1], yend = p2[2])) +
  geom_segment(aes(x = p2[1], y = p2[2], xend = p3[1], yend = p3[2])) +
  geom_segment(aes(x = p3[1], y = p3[2], xend = p0[1], yend = p0[2])) +
  theme_bw() +
  xlim(-15, 15) +
  ylim(-5, 25) +
  labs(x = "x", y = "y")
```

Figure 3: Line 1.



Note: As you continue to plot this here and through question 2, you will find that the x and y limits need to change. Try `xlim(-15,15)` and `ylim(-5, 25)`.

2.6 Part f

Interestingly, the determinant of **A** gives us the area of the shape plotted in Part e. Find the determinant of **A**.

Solution

```r
det(A) #it is a line so this should return 0.
```

```
[1] 0
```

3 Question 3

We can conduct a shear transformation that keeps the area of the shape the same but horizontally or vertically slants the object.

In fact, this is one of the ways artists can make something look like it's leaning or being pushed. Other applications where this is useful include fluid dynamics and crystallography, where the area of a deformed object must remain constant, or in photography where perspective in photos can be adjusted.

A shear transformation is implemented by altering the basis vectors we start with. For vertical slanting,

$$\mathbf{M}_v = \begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix}$$

and for horizontal slanting

$$\mathbf{M}_h = \begin{bmatrix} 1 & 0 \\ k & 1 \end{bmatrix}.$$

In both cases, $k \in \mathbb{R}$ is called a shear factor and its sign determines the direction of the slant and its magnitude determines the severity of the slant.

3.1 Part a

Use the following to generate a new shape. Compare the shape and its area to your answer in Question 1.

$$\mathbf{M}_v = \begin{bmatrix} 1 & k = 0.5 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{A} = \begin{bmatrix} 1 & -7 \\ 5 & 9 \end{bmatrix}$$

Solution

```
Mv = matrix(data= c(1, k=0.5, 0, 1), nrow=2, ncol = 2, byrow = TRUE)
A = matrix(data = c(1, -7, 5, 9), nrow = 2, ncol = 2, byrow = TRUE)
T = A%*%Mv
T
```

```
     [,1] [,2]
[1,]    1 -6.5
[2,]    5 11.5
```

```
det(A) #For area of the parallelogram
```

```
[1] 44
```

```
basis.vector.1 = c(1,5)
basis.vector.2 = c(-6.5,11.5)

p0 = c(0,0)
p1 = basis.vector.1
p2 = basis.vector.1+basis.vector.2
p3 = basis.vector.2
ggplot() +
  geom_segment(aes(x = p0[1], y = p0[2], xend = p1[1], yend = p1[2])) +
  geom_segment(aes(x = p1[1], y = p1[2], xend = p2[1], yend = p2[2])) +
  geom_segment(aes(x = p2[1], y = p2[2], xend = p3[1], yend = p3[2])) +
  geom_segment(aes(x = p3[1], y = p3[2], xend = p0[1], yend = p0[2])) +
  theme_bw() +
  xlim(-15, 15) +
  ylim(-5, 25) +
  labs(x = "x", y = "y")
```

Figure 4: Parallelogram 2.

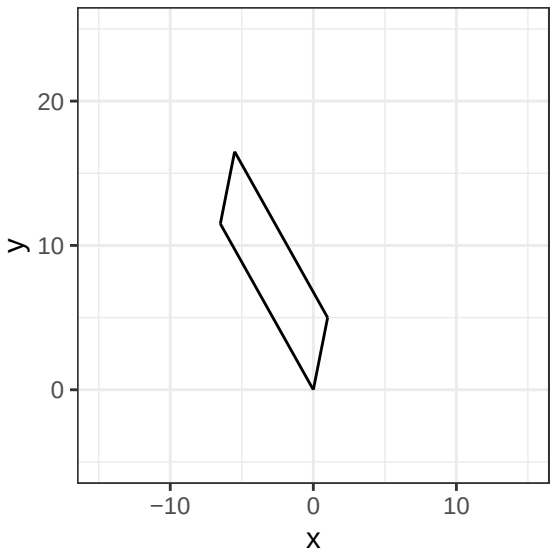


Figure 4 has different side lengths and angles than Figure 2 (giving it a stretched appearance), however their areas are the same, as shown by them having the same determinants of their respective matrix A.

3.2 Part b

Use the following to generate a new shape. Compare the shape and its area to your answer in Questions 1 and the vertical shear transform in part (a).

$$\mathbf{M}_h = \begin{bmatrix} 1 & 0 \\ k = 0.5 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{A} = \begin{bmatrix} 1 & -7 \\ 5 & 9 \end{bmatrix}$$

Solution

```
Mh = matrix(data= c(1, 0, k=0.5, 1), nrow=2, ncol = 2, byrow = TRUE)
A = matrix(data = c(1, -7, 5, 9), nrow = 2, ncol = 2, byrow = TRUE)
T = A%*%Mh
T
```

```
     [,1] [,2]
[1,] -2.5   -7
[2,]  9.5    9
```

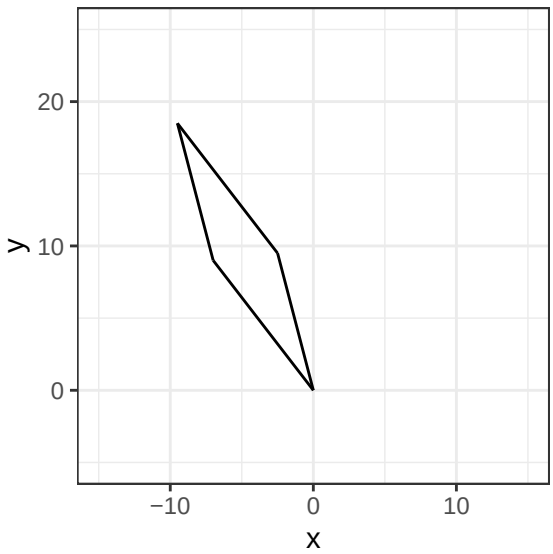
```
det(A) #For area of the parallelogram
```

```
[1] 44
```

```
basis.vector.1 = c(-2.5,9.5)
basis.vector.2 = c(-7,9)
p0 = c(0,0)
p1 = basis.vector.1
p2 = basis.vector.1+basis.vector.2
p3 = basis.vector.2

ggplot() +
  geom_segment(aes(x = p0[1], y = p0[2], xend = p1[1], yend = p1[2])) +
  geom_segment(aes(x = p1[1], y = p1[2], xend = p2[1], yend = p2[2])) +
  geom_segment(aes(x = p2[1], y = p2[2], xend = p3[1], yend = p3[2])) +
  geom_segment(aes(x = p3[1], y = p3[2], xend = p0[1], yend = p0[2])) +
  theme_bw() +
  xlim(-15, 15) +
  ylim(-5, 25) +
  labs(x = "x", y = "y")
```

Figure 5: Parallelogram 3



Compared to Figure 4, Figure 5 is significantly stretched and both its acute and obtuse angles are more extreme. But because it has the same matrix A, it has the same area as the past parallelograms.

3.3 Part c

Can we shear vertically and horizontally at the same time? Use the following to generate a new shape. Compare the shape and its area to your answers in parts (a) and (b).

$$\mathbf{M}_{vh} = \begin{bmatrix} 1 & k = 0.5 \\ k = 0.5 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{A} = \begin{bmatrix} 1 & -7 \\ 5 & 9 \end{bmatrix}$$

Solution

```
Mvh = matrix(data= c(1, k=0.5, k=0.5, 1), nrow=2, ncol = 2, byrow = TRUE)
A = matrix(data = c(1, -7, 5, 9), nrow = 2, ncol = 2, byrow = TRUE)
T = A%*%Mvh
T
```

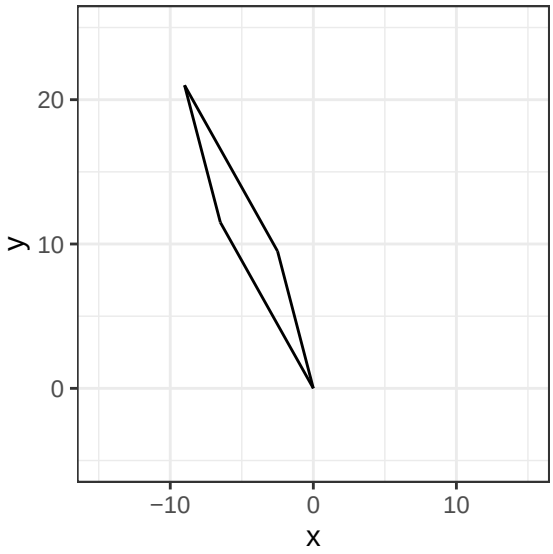
```
     [,1] [,2]
[1,] -2.5 -6.5
[2,]  9.5 11.5
```

```
det(A) #For area of the parallelogram
```

```
[1] 44
```

```
basis.vector.1 = c(-2.5,9.5)
basis.vector.2 = c(-6.5,11.5)
p0 = c(0,0)
p1 = basis.vector.1
p2 = basis.vector.1+basis.vector.2
p3 = basis.vector.2
ggplot() +
  geom_segment(aes(x = p0[1], y = p0[2], xend = p1[1], yend = p1[2])) +
  geom_segment(aes(x = p1[1], y = p1[2], xend = p2[1], yend = p2[2])) +
  geom_segment(aes(x = p2[1], y = p2[2], xend = p3[1], yend = p3[2])) +
  geom_segment(aes(x = p3[1], y = p3[2], xend = p0[1], yend = p0[2])) +
  theme_bw() +
  xlim(-15, 15) +
  ylim(-5, 25) +
  labs(x = "x", y = "y")
```

Figure 6: Parallelogram 4



Similar to the differnce between Figure 4 and Figure 5, Figure 6 is even further stretched compared to the shapes in part a and b. But because it has the same matrix A, it has the same area as the past parallelograms.

4 Question 4

4.1 Part a

Create a data frame `ladder` with the following data. Note your data frame should have three columns; I split it to save vertical space here.

x	y	group	x	y	group
0	0	1	1	8	6
0	20	1	0	10	7
1	0	2	1	10	7
1	20	2	0	12	8
0	2	3	1	12	8
1	2	3	0	14	9
0	4	4	1	14	9
1	4	4	0	16	10
0	6	5	1	16	10
1	6	5	0	18	11
0	8	6	1	18	11

Solution

```r
x = c(0,0,1,1, rep(c(0,1), times=9))
y = c(0,20,0,20, rep(seq(from = 2, to= 18, by=2), each= 2))
group = rep(seq(from=1, to= 11, by=1), each=2)
ladder = data.frame(x, y, group)
ladder
```

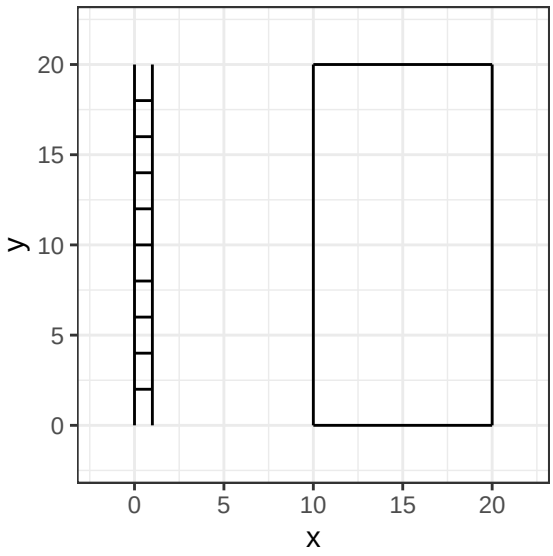
```
   x  y group
1  0  0     1
2  0 20     1
3  1  0     2
4  1 20     2
5  0  2     3
6  1  2     3
7  0  4     4
8  1  4     4
9  0  6     5
10 1  6     5
11 0  8     6
12 1  8     6
13 0 10     7
14 1 10     7
15 0 12     8
16 1 12     8
17 0 14     9
18 1 14     9
19 0 16    10
20 1 16    10
21 0 18    11
22 1 18    11
```

4.2 Part b

Remove `#|eval: false` in the code below. The result should look like a ladder and a very rectangular house if part a is correct.

```
ggplot() +
  geom_path(data = ladder,
              aes(x = x, y = y, group = group)) +
                  geom_segment(aes(x = 10, xend = 10,
                   y=0, yend = 20))+
  geom_segment(aes(x = 10, xend = 20,
                   y=20, yend = 20))+
  geom_segment(aes(x = 20, xend = 20,
                   y=0, yend = 20))+
  geom_segment(aes(x = 10, xend = 20,
                   y=0, yend = 0)) +
  theme_bw() +
  xlim(-2,22) +
  ylim(-2,22)
```

Figure 7: A ladder.



4.3 Part c

Use `as.matrix()` to create a 22×2 matrix $\mathbf{A}$ containing the `x` and `y` observations from the `ladder` data frame. Use matrix multiplication to compute

$$\mathbf{T} = \mathbf{A}\mathbf{M}_h.$$

Recall,

$$\mathbf{M}_h = \begin{bmatrix} 1 & 0 \\ k & 1 \end{bmatrix}$$

and use $k = 0.5$.

Solution

```
A = as.matrix(ladder[ ,-3]) # a matrix with "ladder" data without the group index.
k=0.5
Mh= matrix(data= c(1, 0, k, 1), nrow=2, ncol = 2, byrow = TRUE)
T= A %*% Mh
T
```

```
      [,1] [,2]
 [1,]    0    0
 [2,]   10   20
 [3,]    1    0
```

```
 [4,]   11   20
 [5,]    1    2
 [6,]    2    2
 [7,]    2    4
 [8,]    3    4
 [9,]    3    6
[10,]    4    6
[11,]    4    8
[12,]    5    8
[13,]    5   10
[14,]    6   10
[15,]    6   12
[16,]    7   12
[17,]    7   14
[18,]    8   14
[19,]    8   16
[20,]    9   16
[21,]    9   18
[22,]   10   18
```

4.4 Part d

Create a data frame `leaning.ladder` with the `x` and `y` equal to the first and second column of **T**, respectively, and the same values of `group` from `ladder`.

Solution

```r
leaning.ladder = data.frame(x=T[, 1], #this uses exclusively the first index from T
                            y=T[, 2], #this uses exclusively the second index from T
                            group = group)
leaning.ladder
```

```
    x  y group
1   0  0     1
2  10 20     1
3   1  0     2
4  11 20     2
5   1  2     3
6   2  2     3
7   2  4     4
8   3  4     4
9   3  6     5
10  4  6     5
11  4  8     6
12  5  8     6
13  5 10     7
14  6 10     7
15  6 12     8
16  7 12     8
17  7 14     9
18  8 14     9
19  8 16    10
20  9 16    10
21  9 18    11
22 10 18    11
```

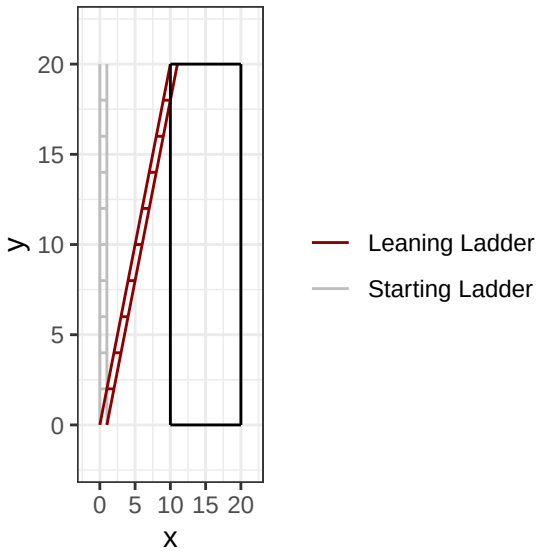
4.5 Part e

Remove `eval: false` in the code below. The result should look like a ladder and a very rectangular house if part a is correct. The new leaning ladder should look like it is leaning on the house.

Solution

```
ggplot() +
  geom_path(data = ladder,
            aes(x = x, y = y, group = group, color = "Starting Ladder")) +
  geom_path(data = leaning.ladder,
            aes(x = x, y = y, group = group, color = "Leaning Ladder")) +
  geom_segment(aes(x = 10, xend = 10,
                   y=0, yend = 20))+
  geom_segment(aes(x = 10, xend = 20,
                   y=20, yend = 20))+
  geom_segment(aes(x = 20, xend = 20,
                   y=0, yend = 20))+
  geom_segment(aes(x = 10, xend = 20,
                   y=0, yend = 0)) +
  theme_bw() +
  xlim(-2,22) +
  ylim(-2,22) +
  scale_color_manual("", values=c("darkred", "grey"))
```

Figure 8: A ladder and a leaning ladder.



5 Citations

I used the tidyverse package (Wickham et al. 2019) for developing figures and R (R Core Team 2025) as my coding program.

References

R Core Team. 2025. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.