Visual proof of Pythagoras’ theorem

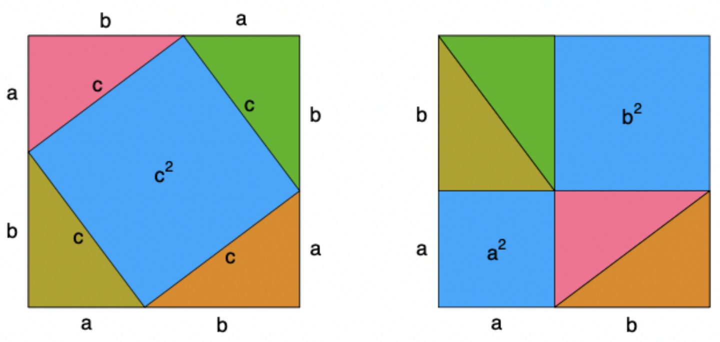
Asked yesterday Modified today Viewed 50 times Part of R Language Collective

0

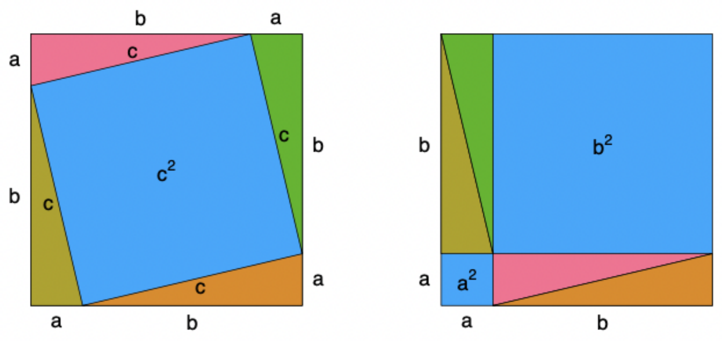
The following two graphs provide a visual proof of Pythagoras’ theorem:

$$c^2 = a^2 + b^2.$$

(Can you see why?)



(a) Write two R functions, each with two arguments for a and b, that can be used to produce the two graphs, respectively, while a and b may take any positive values. For example, the above two graphs are produced using $a = 3$ and $b = 4$, and with $a = 2.5$ and $b = 10.6$, the following two graphs are produced. All (non-black) colours needed must be produced from function `hcl()`, but they don’t have to look exactly the same as shown here. Demonstrate your functions work well.



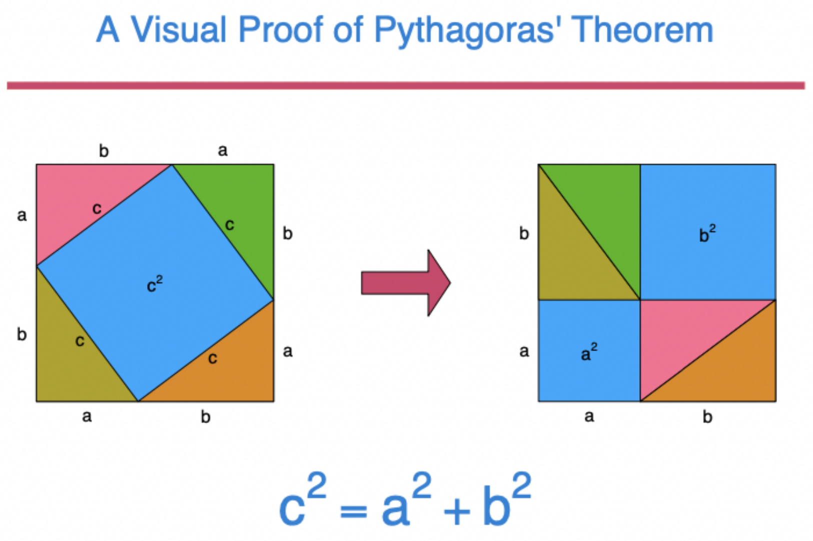
(b) In aid of function `layout()` and the two functions implemented in Part (a), write some R code to reproduce the following graph. Make sure all colours needed are produced from function `hcl()`. [If you are unable to finish Part (a), you may replace the two graphs with any graphs. You won’t lose marks for doing this for Part (b).]

I'm trying to create proof of Pythagoras’ theorem in R but I'm very confused.

I tried using the polygon function to make the centre diamond shape.

a) Write two R functions, each with two arguments for a and b, that can be used to produce the two graphs, respectively, while a and b may take any positive values. For example, the above two graphs are produced using a = 3 and b = 4, and with a = 2.5 and b = 10.6, the following two graphs are produced. All (non-black) colours needed must be produced from function hcl(), but they don't have to look exactly the same as shown here. Demonstrate your functions work well.

b) In aid of function layout() and the two functions implemented in Part (a), write some R code to reproduce the following graph. Make sure all colours needed are produced from function hcl().



my code so far:

```
plot.new()
plot.window(xlim = c(0, 10), ylim = c(0, 10))


half_side_length <- 2.5


x_coords_polygon <- c(5, 7.5, 5, 2.5)
y_coords_polygon <- c(2.5, 7.5, 10, 7.5)


x_coords_triangles <- list(
  c(5, 6.25, 5),
  c(6.25, 7.5, 6.25),
  c(5, 3.75, 5),
  c(3.75, 2.5, 3.75)
)

y_coords_triangles <- list(
  c(2.5, 3.75, 5),
  c(7.5, 8.75, 10),
  c(7.5, 6.25, 5),
  c(2.5, 1.25, 0)
)

 polygon(x_coords_polygon, y_coords_polygon, col = "blue", border = "black")
```

```
for (i in 1:4) {
  polygon(x_coords_triangles[[i]], y_coords_triangles[[i]], col = "lightgreen", border
= "black")
}
```

r Edit tags

Share Edit Follow Close Flag

edited yesterday
Phil
7,237 3 36 65

asked yesterday
user
1 1
New contributor

Hi user - what's your question, beyond the questions in the exercise? – Phil yesterday

I just have no idea where to start. For the first function I was able to make a diamond shape but it doesn't look like how its meant to and then I have no idea how to add the triangles along the side. so sorry – user yesterday

@Phil are you please able to help me with this question in anyway. I would appreciate it a lot :) – user yesterday

1 Please see the community guidelines on asking homework questions meta.stackoverflow.com/a/334823/5221626 – Phil yesterday

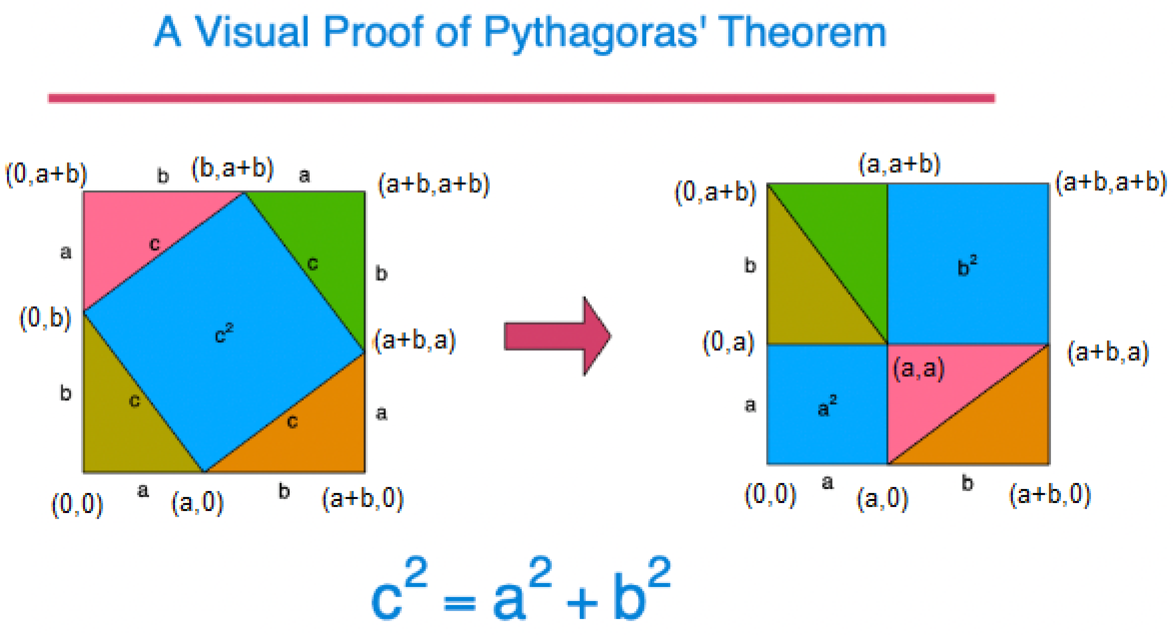
You mentioned that you made an attempt for a plot, please add as part of your question the code that you have attempted so far so that we can reproduce the issue on our end, and amend your code as needed. – Phil yesterday

2 Answers

Sorted by:
Reset to default
Date modified (newest first)

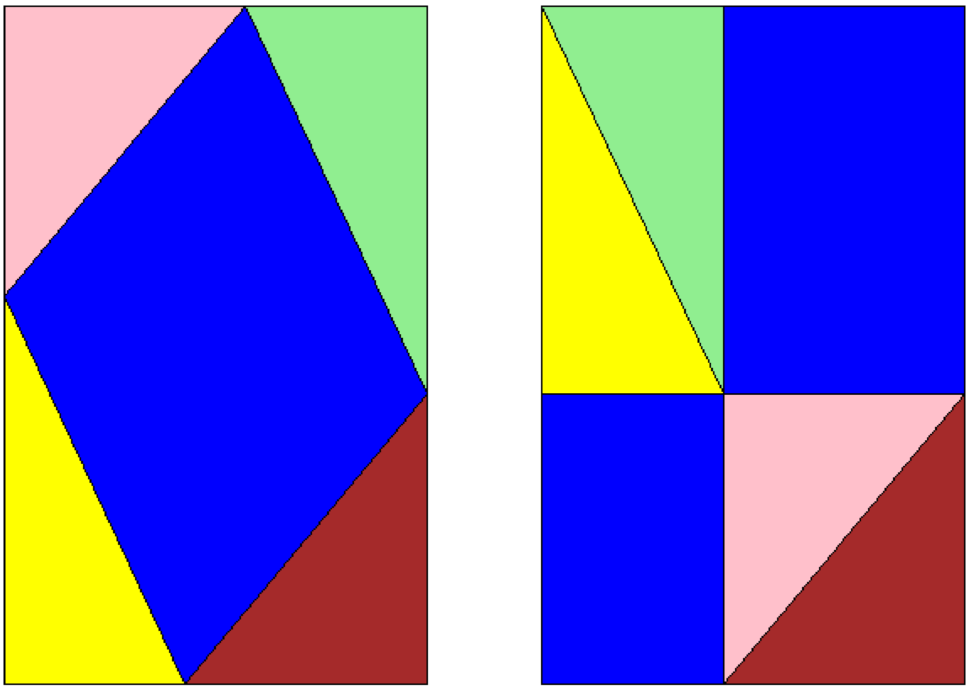
0

Let's define the coordinate system, find the coordinates of the corner points of the polygons and parameterize the functions (the parameters being `a` and `b`), as shown in the next figure:



Now, draw the LHS and RHS squares:

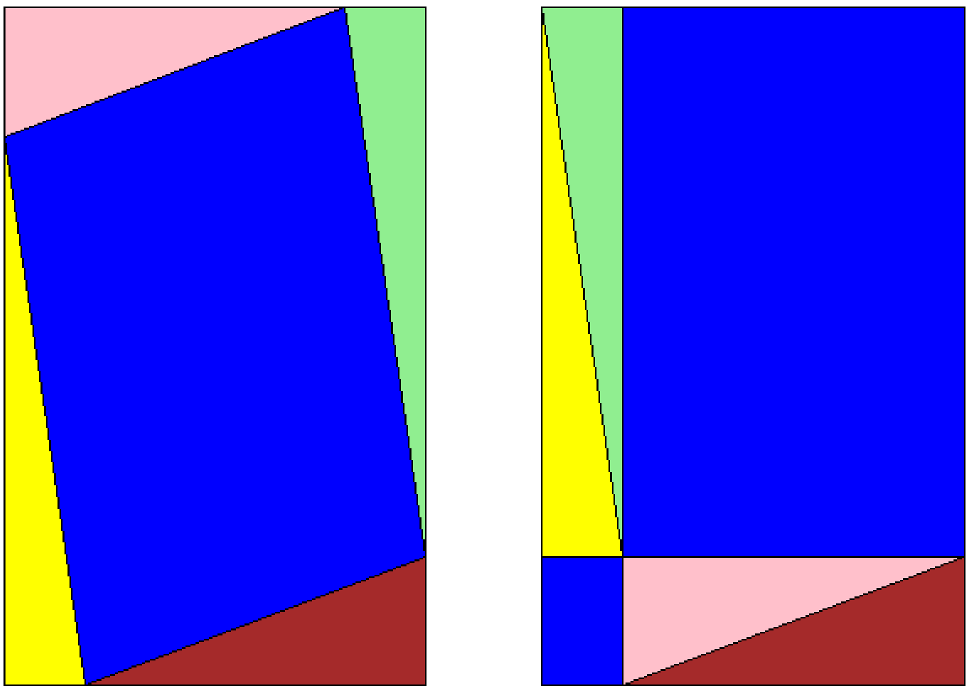
```
draw_left_square <- function(a, b, xlim=8, ylim=8) {

  plot.new()
  plot.window(xlim = c(0, xlim), ylim = c(0, ylim))

  x_coords_polygon <- c(0, a, a+b, b)
  y_coords_polygon <- c(b, 0, a, a+b)
  polygon(x_coords_polygon, y_coords_polygon, col = "blue", border = "black")

  x_coords_triangles <- list(
    c(0, 0, a),
    c(a, a+b, a+b),
    c(a+b, a+b, b),
    c(0, 0, b)
  )
  y_coords_triangles <- list(
    c(b, 0, 0),
    c(0, 0, a),
    c(a, a+b, a+b),
    c(b, a+b, a+b)
  )
  cols <- c("yellow", "brown", "lightgreen", "pink")
  for (i in 1:4) {
    polygon(x_coords_triangles[[i]], y_coords_triangles[[i]], col = cols[i], border =
"black")
  }
}

draw_right_square <- function(a, b, xlim=8, ylim=8) {

  plot.new()
  plot.window(xlim = c(0, xlim), ylim = c(0, ylim))

  x_coords_polygon <- c(0, a, a, 0)
  y_coords_polygon <- c(0, 0, a, a)
  polygon(x_coords_polygon, y_coords_polygon, col = "blue", border = "black")
  x_coords_polygon <- c(a, a+b, a+b, a)
  y_coords_polygon <- c(a, a, a+b, a+b)
  polygon(x_coords_polygon, y_coords_polygon, col = "blue", border = "black")
  x_coords_triangles <- list(
    c(0, a, 0),
    c(a, a, 0),
    c(a, a+b, a+b),
    c(a, a, a+b)
  )
  y_coords_triangles <- list(
    c(a, a, a+b),
    c(a, a+b, a+b),
    c(0, 0, a),
    c(0, a, a)
  )
  cols <- c("yellow", "lightgreen", "brown", "pink")
  for (i in 1:4) {
    polygon(x_coords_triangles[[i]], y_coords_triangles[[i]], col = cols[i], border =
"black")
  }
}

a <- 3
b <- 4

par(mar=c(0.2, 0.2, 0.2, 0.2), mfrow=c(1,2),
    oma = c(4, 4, 0.1, 0.1))
draw_left_square(a, b)
draw_right_square(a, b)
```



with different values of `a`, `b`:

```
a <- 2.5
b <- 10.6
```

```
par(mar=c(0.2, 0.2, 0.2, 0.2), mfrow=c(1,2),
    oma = c(4, 4, 0.1, 0.1))
draw_left_square(a, b, xlim=15, ylim=15)
draw_right_square(a, b, xlim=15, ylim=15)
```



Share Edit Delete Flag

answered 1 min ago

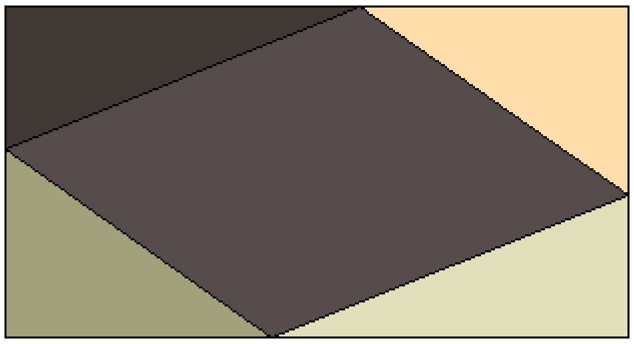
Sandipan Dey
21.4k 2 49 63

0

The problem I'm facing is that I lack the background and context for the exercises. Is this about learning on how to use base R's plotting devices, is this for mathematical learning? I'm concerned that I could send you astray because nothing in my answer relies on the theorem.

I'm providing an answer for the first function of part A below, loosely based on the code you started. The idea is to place the limits of each polygon based on the labels in the diagrams shown with the questions. So for the diamond, if the bottom point is at `x = 1 + a, y = 1`, then the right point is at `x = 1 + a + b, y = 1 + a`, the upper point is at `x = 1 + b, y = 1 + a + b`, and the left point is at `x = 1, y = 1 + b`.

```
fun1 <- function(a, b) {
  plot.new()
  plot.window(xlim = c(1, 1 + a + b),
              ylim = c(1, 1 + a + b))

  # Diamond
  polygon(x = c(1 + a, 1 + a + b, 1 + b, 1),
          y = c(1, 1 + a, 1 + a + b, 1 + b),
          col = hcl(runif(1, 0, 100), runif(1, 0, 100), runif(1, 0, 100)))

  # Triangles

  # bottom left
  polygon(x = c(1, 1 + a, 1),
          y = c(1, 1, 1 + b),
          col = hcl(runif(1, 0, 100), runif(1, 0, 100), runif(1, 0, 100)))
  # bottom right
  polygon(x = c(1 + a, 1 + a + b, 1 + a + b),
          y = c(1, 1, 1 + a),
          col = hcl(runif(1, 0, 100), runif(1, 0, 100), runif(1, 0, 100)))
  # top right
  polygon(x = c(1 + a + b, 1 + a + b, 1 + b),
          y = c(1 + a, 1 + a + b, 1 + a + b),
          col = hcl(runif(1, 0, 100), runif(1, 0, 100), runif(1, 0, 100)))
  # top left
  polygon(x = c(1 + b, 1, 1),
          y = c(1 + a + b, 1 + a + b, 1 + b),
          col = hcl(runif(1, 0, 100), runif(1, 0, 100), runif(1, 0, 100)))
}

fun1(3, 4)
```



Share Edit Follow Flag

answered yesterday
Phil
7,237 3 36 65