Construindo Mandalas com R

in:IX Xornada de Usuarios de R en Galicia

Profa. Dra. Luciane Alcoforado

Academia da Força Aérea

20 de outubro de 2022 (updated: 2022-10-10)

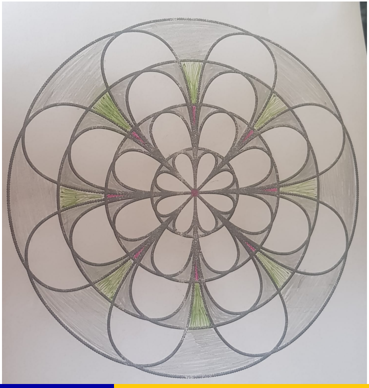
Prof. Luciane https://altabooks.com.br/produto/utilizando-a-linguagem-r/

Figure 1: Livro

Introdução

A relação entre a Mandala e a matemática encontra-se no seu significado (círculo). As mandalas são compostas por figuras geométricas que se repetem através de transformações matemáticas como isometrias e homotetias.

Mandalas: círculo em Sânscrito



Objetivos

- Apresentar as equações parametrizadas de curvas clássicas da matemática;
- Criar uma base de pontos (x,y) em um data.frame pertencentes a curva escolhida;
- Empregar transformações matemáticas para produção das mandalas;
- programar a visualização dos pontos da mandala
- Criar diversas Mandalas com Matemática e programação em R

Equações paramétricas

Definição de MathWorld (2022): é um conjunto de equações que expressam um conjunto de pontos como funções explícitas de uma série de variáveis independentes, conhecidas como “parâmetros”.

$x = f(t); y = g(t)$

Transformações Matemáticas

isometrias - preservam distância

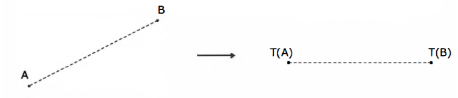
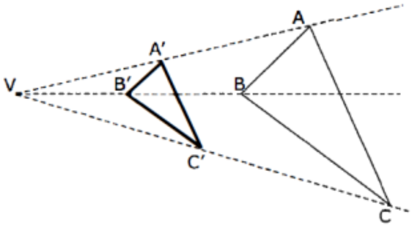
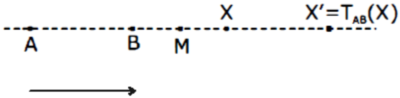


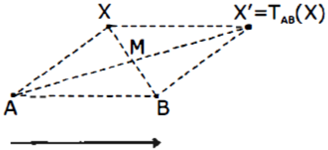
Figure 2:

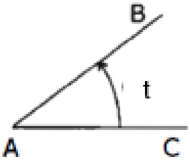
homotetias - preservam forma

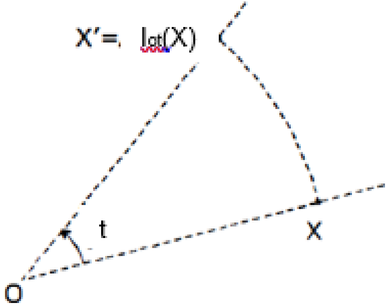


Isometrias: translação e rotação







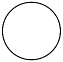


Curvas

Círculo

- Equações paramétricas

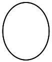
$$x = r \cdot \cos(t)$$
$$y = r \cdot \sin(t) \qquad t \in [0, 2\pi]$$



Elipse

- Equações paramétricas:

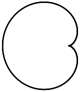
$$x = a \cdot \cos(t)$$
$$y = b \cdot \sin(t) \qquad t \in [0, 2\pi]$$



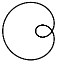
Curvas

Cardioide

- "Similar a um coração"
- Equações paramétricas:
  $x = 2 \cdot r \cdot \cos(t) - r \cdot \cos(2t)$
  $y = 2 \cdot r \cdot \sin(t) - r \cdot \sin(2t)$
  $t \in [0, 2\pi]$



Limaçon de Pascal



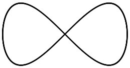
- Equações paramétricas:

$$x = (b + a \cdot \cos(t)) \cdot \cos(t) = \frac{a}{2} + b \cdot \cos(t) + \frac{a}{2} \cdot \cos(2t)$$
$$y = (b + a \cdot \cos(t)) \cdot \sin(t) = b \cdot \sin(t) + \frac{a}{2} \cdot \sin(2t)$$

Curvas

Lemniscata

- representa o sinal matemático do "infinito"



- Parametrização:
  $x = \sin(t), \qquad y = \cos(t) sen(t), \qquad t \in [0, 2\pi]$

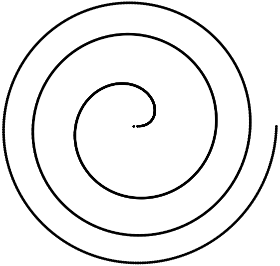
$$x = \frac{a \cdot \sin(t)}{1 + \cos^2(t)}, \qquad y = \frac{a \cdot \cos(t)\sin(t)}{1 + \cos^2(t)}, \qquad t \in [0, 2\pi]$$

Curvas

Espiral de Fermat

- Equações paramétricas são:

$$x = a \cdot \sqrt{t} \cdot \cos(t) \quad y = a \cdot \sqrt{t} \cdot \sin(t) \quad t \geqslant 0$$



Programando em R

O procedimento está decomposto em duas partes:

- na primeira parte elabora-se a criação de uma base de dados contendo todos os pontos das coordenadas (x,y) que formam o desenho da mandala;
- na segunda parte elabora-se a visualização do desenho utilizando-se o pacote ggplot2.

Gerando pontos para o círculo

```
#Passo 1: Cálculos e tabela com os pontos P(x,y)
n = 500; r = 1; t = seq(0, 2*pi, length.out = n)
x = r*cos(t); y = r*sin(t)
dt = data.frame(x,y)
str(dt)
```

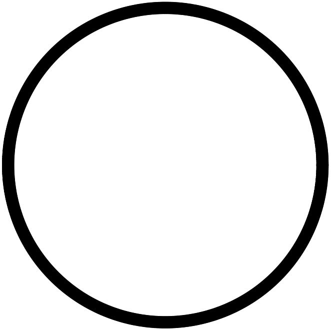
```
## 'data.frame':    500 obs. of  2 variables:
##  $ x: num  1 1 1 0.999 0.999 ...
##  $ y: num  0 0.0126 0.0252 0.0378 0.0503 ...
```

```
head(dt)
```

```
##           x          y
## 1 1.0000000 0.00000000
## 2 0.9999207 0.01259122
## 3 0.9996829 0.02518045
```

Visualizando o círculo

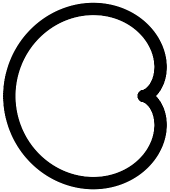
```
#Passo 2: Construindo a curva
require(ggplot2) #carregando o pacote
p = ggplot() + coord_fixed() + theme_void()
p + geom_point(data=dt, aes(x=x, y=y), color='black')
```



Visualizando o cardióide

```r
#Passo 1: Cálculos e tabela com os pontos P(x,y)
n = 500;  r = 1; t = seq(0, 2*pi, length.out = n)
x = 2*r*cos(t)-r*cos(2*t); y = 2*r*sin(t)-r*sin(2*t)
dt = data.frame(x,y)
```

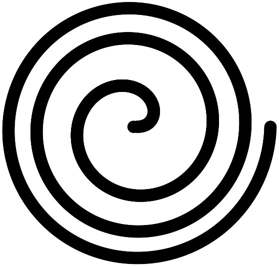
```r
#Passo 2: Construindo a curva
require(ggplot2) #carregando o pacote
p = ggplot() + coord_fixed() + theme_void()
p + geom_point(data=dt, aes(x=x, y=y), color='black')
```



Visualizando a espiral de Fermat

```
#Passo 1: Cálculos e tabela com os pontos P(x,y)
n = 1500; r = 1; t = seq(0, 6*pi, length.out = n)
x = r*sqrt(t)*cos(t); y = r*sqrt(t)*sin(t)
dt = data.frame(x,y)
```

No passo 2 repetimos o mesmo código anterior!

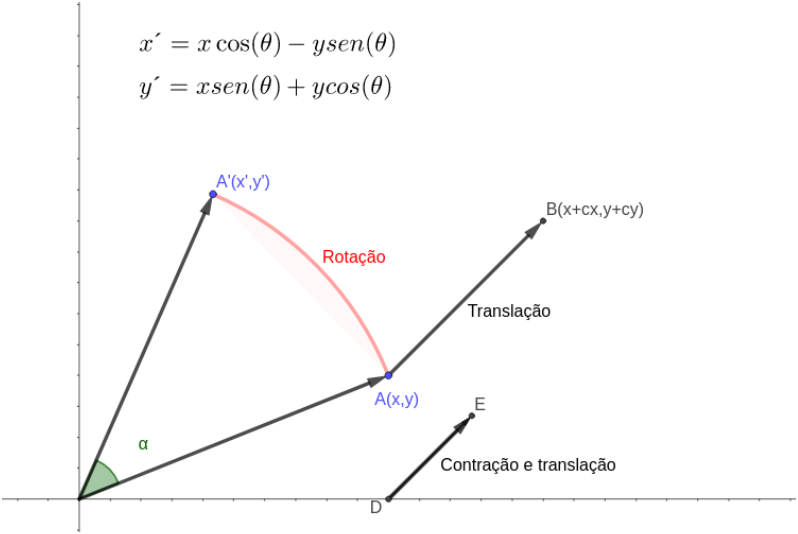


Reutilização do código de visualização

O código é utilizado em todas as visualizações subsequentes.

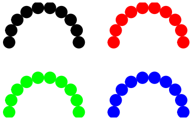
```
require(ggplot2)
p = ggplot() + coord_fixed() + theme_void()
p + geom_point(data=dt, aes(x=x, y=y), color='black')
```

Transformações Geométricas



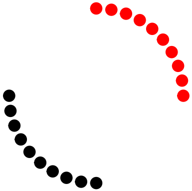
Exemplo de translação

```
#Semi círculo (red)
n=10; raio=1; t=seq(0,pi, length.out = n)
x1 = raio*cos(t); y1 = raio*sin(t); dt1 = data.frame(x=x1, y=y
#Translação no eixo x (black)
x=x1-3; y=y1; dtx = data.frame(x,y)
#Translação no eixo y (blue)
x=x1; y=y1-2; dty = data.frame(x,y)
#Translação no eixo x e y (green)
x=x1-3; y=y1-2; dtxy = data.frame(x,y)
```



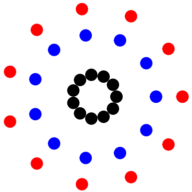
Exemplo de Rotação

```
#Semi círculo (red)
n=10; raio=1; t=seq(0,pi/2, length.out = n)
x1 = raio*cos(t); y1 = raio*sin(t); dt1 = data.frame(x=x1, y=y
#Rotação de pi (black)
xt1=x1*cos(pi)-y1*sin(pi); yt1=x1*sin(pi)+y1*cos(pi)
dt = data.frame(x=xt1,y=yt1)
```



Exemplo de Redução

```
#círculo de raio 1 (red)
n=12; raio=1; t=seq(0,2*pi, length.out = n)
x1 = raio*cos(t); y1 = raio*sin(t); dt1 = data.frame(x=x1, y=y
#Redução de um fator de 25% (black)
xt1=x1*0.25; yt1=y1*0.25; dt = data.frame(x=xt1,y=yt1)
#Redução de um fator de 70% (blue)
xt1=x1*0.7; yt1=y1*0.7; dt2 = data.frame(x=xt1,y=yt1)
```



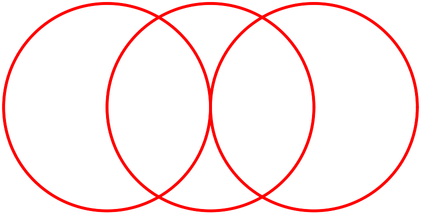
Construindo três círculos

```
#Primeiro círculo
n=500; raio=1; t=seq(0,2*pi, length.out = n)
x1=raio*cos(t); y1=raio*sin(t)
#pontos para os 3 círculos: translação dos pontos iniciais (x1
x=c(x1,x1-raio,x1-2*raio)
y=c(y1,y1,y1)

dt=data.frame(x,y,z="circulo")
str(dt)
```

```
## 'data.frame':    1500 obs. of  3 variables:
##  $ x: num  1 1 1 0.999 0.999 ...
##  $ y: num  0 0.0126 0.0252 0.0378 0.0503 ...
##  $ z: Factor w/ 1 level "circulo": 1 1 1 1 1 1 1 1 1 1 ...
```

Visualizando

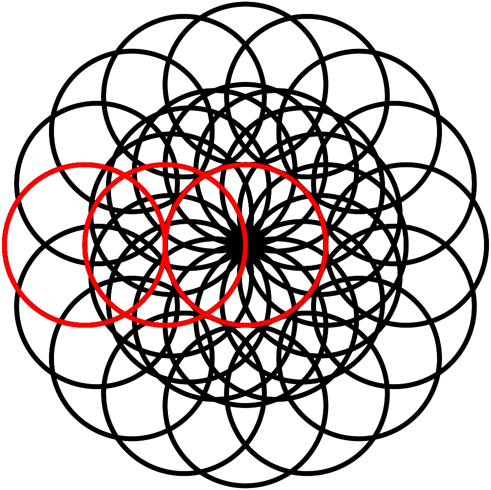


Aplicando rotações $\pi/8, 2\pi/8, ..., 2\pi$

```
rotacao = (pi/8)*(1:16); n=length(x); xt1=x; yt1=y
for(i in 1:length(rotacao))
  {
xt1=c(xt1,x[1:n]*cos(rotacao[i])-y[1:n]*sin(rotacao[i]))
yt1=c(yt1,x[1:n]*sin(rotacao[i])+y[1:n]*cos(rotacao[i]))
}
dt= data.frame(x=xt1,y=yt1,z="circulo")
str(dt)
```

```
## 'data.frame':    25500 obs. of  3 variables:
##  $ x: num  1 1 1 0.999 0.999 ...
##  $ y: num  0 0.0126 0.0252 0.0378 0.0503 ...
##  $ z: Factor w/ 1 level "circulo": 1 1 1 1 1 1 1 1 1 1 ...
```

Visualização Mandala da vida



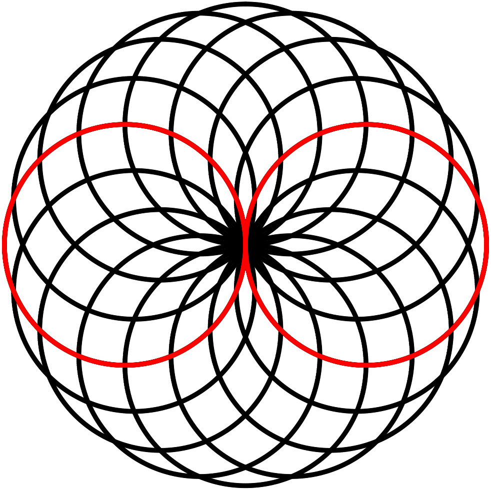
Variações

Dois círculos de mesmo raio, tangentes com rotação de meia volta.

```
n=500; raio=1; t=seq(0,2*pi, length.out = n)
x=raio*cos(t); y=raio*sin(t)
#pontos para os 2 círculos
xt=c(x+1*raio,x-1*raio); yt=c(y,y)
rotacao = (pi/8)*(1:8); n=length(xt); x=xt; y=yt
for(i in 1:length(rotacao))
  {
x=c(x,xt[1:n]*cos(rotacao[i])-yt[1:n]*sin(rotacao[i]))
y=c(y,xt[1:n]*sin(rotacao[i])+yt[1:n]*cos(rotacao[i]))
}
dt= data.frame(x,y,z="circulo")
str(dt)
```

```
## 'data.frame':    9000 obs. of  3 variables:
##  $ x: num  2 2 2 2 2 ...
```

Visualização



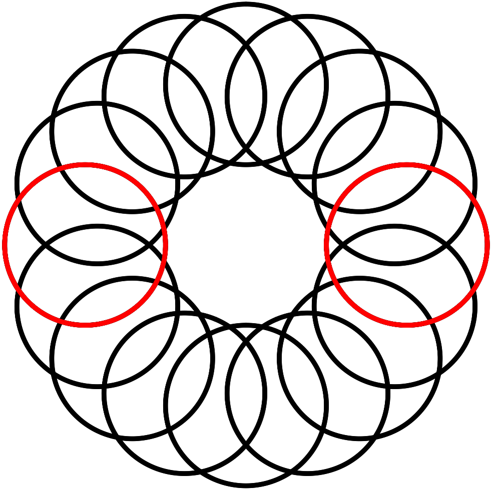
Variações

Dois círculos de mesmo raio, não tangentes com rotação de meia volta.

```
n=500; raio=1; t=seq(0,2*pi, length.out = n)
x=raio*cos(t); y=raio*sin(t)
#pontos para os 2 círculos
xt=c(x+2*raio,x-2*raio); yt=c(y,y)
rotacao = (pi/8)*(1:8); n=length(xt); x=xt; y=yt
for(i in 1:length(rotacao))
  {
x=c(x,xt[1:n]*cos(rotacao[i])-yt[1:n]*sin(rotacao[i]))
y=c(y,xt[1:n]*sin(rotacao[i])+yt[1:n]*cos(rotacao[i]))
}
dt= data.frame(x,y,z="circulo")
str(dt)
```

```
## 'data.frame':    9000 obs. of  3 variables:
##  $ x: num  3 3 3 3 3 ...
```

Visualização



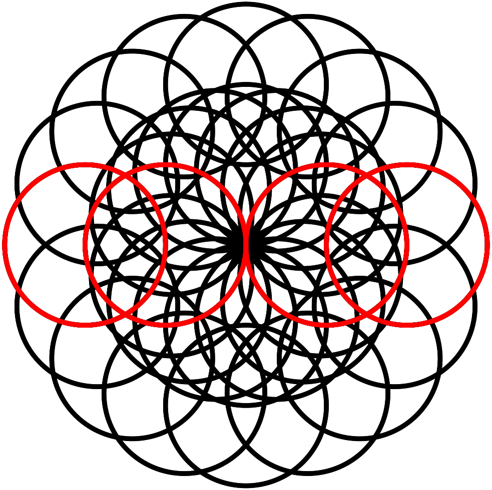
Variações

Quatro círculos de mesmo raio, entrelaçados, com rotação de meia volta.

```
n=500; raio=1; t=seq(0,2*pi, length.out = n)
x=raio*cos(t); y=raio*sin(t)
#pontos para os 4 círculos
xt=c(x+1*raio,x+2*raio, x-1*raio, x-2*raio); yt=c(y,y,y,y)
rotacao = (pi/8)*(1:8); n=length(xt); x=xt; y=yt
for(i in 1:length(rotacao))
  {
x=c(x,xt[1:n]*cos(rotacao[i])-yt[1:n]*sin(rotacao[i]))
y=c(y,xt[1:n]*sin(rotacao[i])+yt[1:n]*cos(rotacao[i]))
}
dt= data.frame(x,y,z="circulo")
str(dt)
```

```
## 'data.frame':    18000 obs. of  3 variables:
##  $ x: num  2 2 2 2 2 ...
```

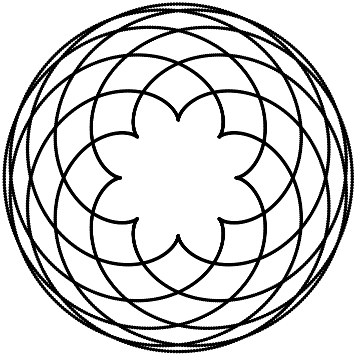
Visualização



Mandala do coração: cardioide

Rotações variando de $\pi/4, 2\pi/4, 3\pi/4, ..., 7\pi/4$

Visualização

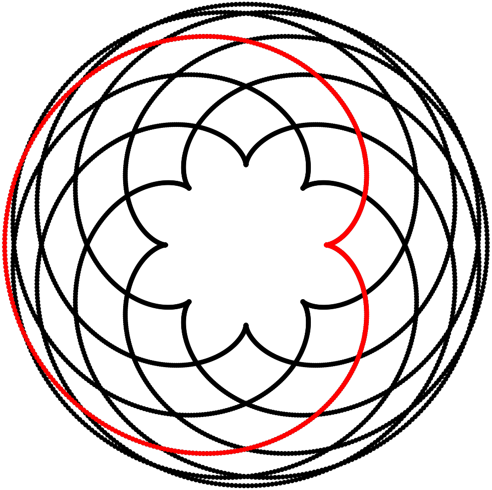


Construindo o cardióide com 7 rotações

```
n=500; t=seq(0, 2*pi, length.out = n); rotacao=pi/4*(1:7)
x=c(2*raio*cos(t)-raio*cos(2*t))
y=c(2*raio*sin(t)-raio*sin(2*t))
xt=x; yt=y #rotação dos pontos
for(i in 1:length(rotacao)){
xt=c(xt, x[1:n]*cos(rotacao[i])-y[1:n]*sin(rotacao[i]))
yt=c(yt, x[1:n]*sin(rotacao[i])+y[1:n]*cos(rotacao[i]))
}
dt= data.frame(x=xt, y=yt, z="cardióide")
str(dt)
```

```
## 'data.frame':    4000 obs. of  3 variables:
##  $ x: num  1 1 1 1 1 ...
##  $ y: num  0.00 2.00e-06 1.60e-05 5.39e-05 1.28e-04 ...
##  $ z: Factor w/ 1 level "cardióide": 1 1 1 1 1 1 1 1 1 1 ..
```

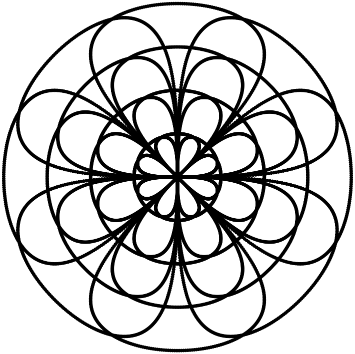
Visualização



Mandala do infinito: Lemniscata

Sobreposição: Lemniscata com rotações $\pi/4, \pi/2, 3\pi/4$ e 3 contrações por fatores 25%, 50% e 75%.

Visualização

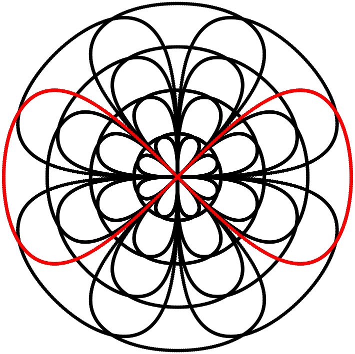


Construindo a lemniscata com 3 rotações e 3 reduções

```r
n=500; t=seq(0, 2*pi, length.out = n); rotacao=pi/4*(1:3)
x=sin(t);  y=sin(t)*cos(t)
xt=x; yt=y#rotações
for(i in 1:length(rotacao)){
xt=c(xt, x[1:n]*cos(rotacao[i])-y[1:n]*sin(rotacao[i]))
yt=c(yt, x[1:n]*sin(rotacao[i])+y[1:n]*cos(rotacao[i])) }
xtt=NULL; ytt=NULL; red=c(0.25, 0.5, 0.75) #redução
for(i in 1:length(red)){
provx=paste0("x",i); provy=paste0("y",i)
xtt=c(xtt, assign(provx, xt*red[i]))
ytt=c(ytt, assign(provy, yt*red[i])) }
dt=data.frame(x=c(xt, xtt), y=c(yt, ytt), z="lemniscata")
str(dt)
```

```
## 'data.frame':    8000 obs. of  3 variables:
##  $ x: num  0 0.0126 0.0252 0.0378 0.0503 ...
```

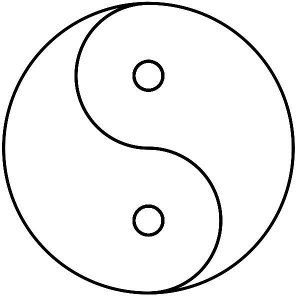
Visualização

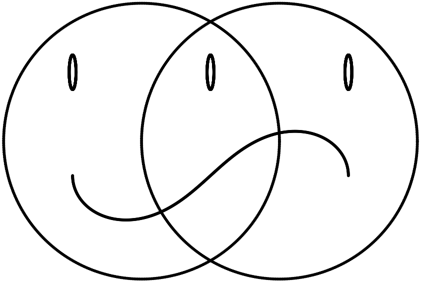


Muitas Possibilidades

- O número de combinações para produção de figuras distintas depende apenas do usuário
- As curvas famosas(clássicas) podem ser utilizadas em um número infinito de possibilidades
- Outras curvas podem ser adicionadas às curvas apresentadas: círculo, elipse, deltoide, astroide, limaçon, lemniscata, cardióide
- Variação do ângulo de rotação, combinação das contrações e posições

Algumas combinações interessantes





À esquerda: Ying Yang - mandala da filosofia chinesa feita apenas com círculos. Representa o princípio das dualidades presentes na natureza, onde o positivo não vive sem o negativo e vice e versa.

À direita: Sentimentos - mandala feita com círculos, elipses e infinito, representando a transição de sentimentos.

. . .

Aplicativo shiny

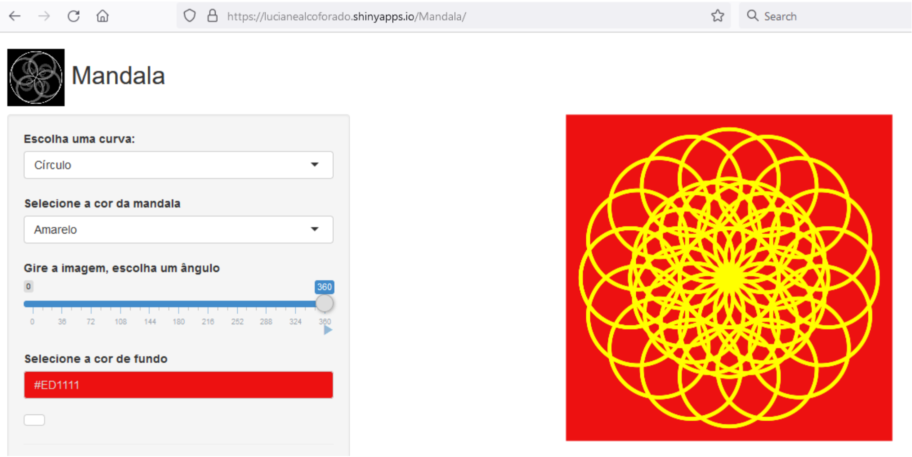


Figure 4: https://lucianealcoforado.shinyapps.io/Mandala/

Referências

[1] Alcoforado, L.F. (Mandala . Aplicativo Disponível em https://lucianealcoforado.shinyapps.io/Mandala/ Mandala/. Acesso em 11/09/2022.

[2] Alcoforado, L.F. (2021) Utilizando a Linguagem R : conceitos, manipulação, visualização, Modelagem e Elaboração de Relatório , Alta Books, Rio de Janeiro.

[3] Bezerra, J. 2022), Mandalas Toda Materia. Disponível em https:// www.todamateria.com.br/mandala/ mandala/. Acesso em 11/09/2022.

[4] Rezende, E. Q. F, Queiroz, M.L.B. (2008)). Geometria Euclidiana Plana de Construções Geométricas , 2a. ed., Unicamp, Campinas.

[5] Stover, C .., Weisstein, E. W. (2022). Parametric Equations From MathWorld A Wolfram Web R esource. https://mathworld.wolfram.com/ParametricEquations.html , Acesso em 11/09/2022