Now that Twitter allows 280 characters, the code of some drawings I have made can fit in a tweet. In this post I have compiled a few of them.

The first one is a cardioid inspired in string art:

library(ggplot2)

n=300

t1=1:n

t0=seq(3,2\*n+1,2)%%n

t2=t0+(t0==0)\*n

df=data.frame(x=cos((t1-1)\*2\*pi/n),

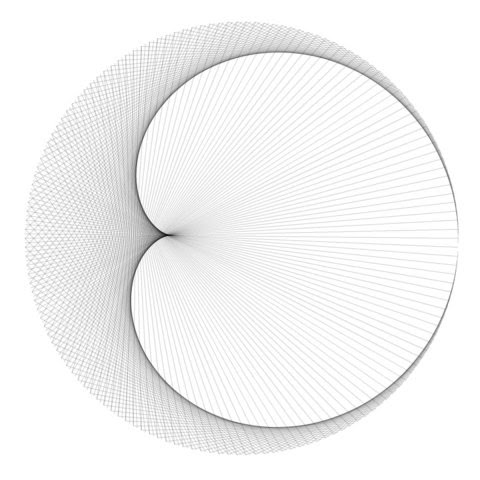
y=sin((t1-1)\*2\*pi/n),

x2=cos((t2-1)\*2\*pi/n),

y2=sin((t2-1)\*2\*pi/n))

ggplot(df,aes(x,y,xend=x2,yend=y2)) +

geom\_segment(alpha=.1)+theme\_void()

  
This other is based on Fermat’s spiral:

library(ggplot2)

library(magrittr)

setwd("YOUR-WORKING-DIRECTORY-HERE")

opt=theme(legend.position="none",

panel.background = element\_rect(fill="white"),

panel.grid=element\_blank(),

axis.ticks=element\_blank(),

axis.title=element\_blank(),

axis.text=element\_blank())

for (n **in** 1:25){

t=seq(from=0, to=n\*pi, length.out=500\*n)

data.frame(x= t^(1/2)\*cos(t), y= t^(1/2)\*sin(t)) %>% rbind(-.) -> df

p=ggplot(df, aes(x, y))+geom\_polygon()+

scale\_x\_continuous(expand=c(0,0), limits=c(-9, 9))+

scale\_y\_continuous(expand=c(0,0), limits=c(-9, 9))+opt

ggsave(filename=paste0("Fermat",sprintf("%03d", n),".jpg"), plot=p, width=3, height=3)}

library(ggplot2)

library(dplyr)

t=seq(from=0, to=100\*pi, length.out=500\*100)

data.frame(x= t^(1/2)\*cos(t), y= t^(1/2)\*sin(t))%>%

rbind(-.)%>%ggplot(aes(x, y))+geom\_polygon()+theme\_void()

  
A recurrence plot of *Gauss error function*:

library("magrittr")

library("ggplot2")

library("pracma")

RecurrencePlot = **function**(from, to, col1, col2) {

  opt = theme(legend.position  = "none",

              panel.background = element\_blank(),

              axis.ticks       = element\_blank(),

              panel.grid       = element\_blank(),

              axis.title       = element\_blank(),

              axis.text        = element\_blank())

  seq(from, to, by = .1) %>% expand.grid(x=., y=.) %>%

    ggplot( ., aes(x=x, y=y, fill=erf(sec(x)-sec(y)))) + geom\_tile() +

    scale\_fill\_gradientn(colours=colorRampPalette(c(col1, col2))(2)) + opt}

RecurrencePlot(from = -5\*pi, to = 5\*pi, col1 = "black", col2= "white")

library(dplyr)

library(ggplot2)

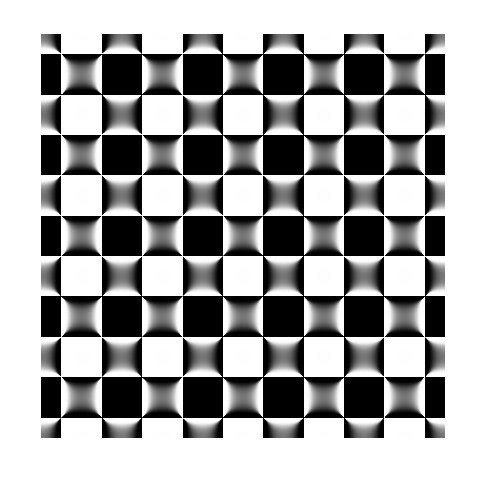
library(pracma)

seq(-5\*pi,5\*pi,by=.1)%>%expand.grid(x=., y=.)%>%

ggplot(aes(x=x, y=y, fill=erf(sec(x)-sec(y))))+geom\_tile()+

scale\_fill\_gradientn(colours=c("#000000","#FFFFFF"))+

theme\_void()+theme(legend.position="none")

  
A x-y scatter plot of a trigonometric function on R2:

library(magrittr)

library(ggplot2)

opt = theme(legend.position  = "none",

panel.background = element\_rect(fill="violetred4"),

axis.ticks       = element\_blank(),

panel.grid       = element\_blank(),

axis.title       = element\_blank(),

axis.text        = element\_blank())

seq(from=-10, to=10, by = 0.05) %>%

expand.grid(x=., y=.) %>%

#HERE COMES THE KIDNAPPED LINE

geom\_point(alpha=.1, shape=20, size=1, color="white") + opt

library(dplyr)

library(ggplot2)

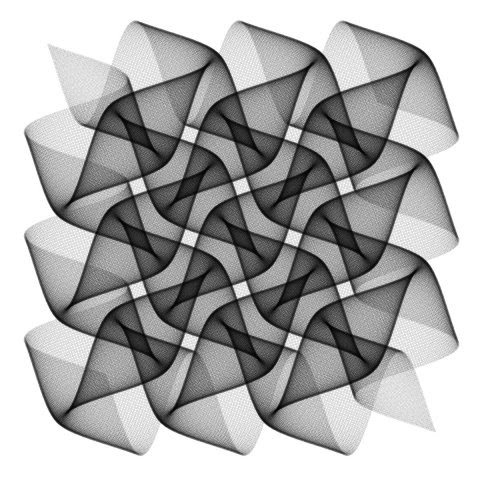
seq(from=-10, to=10, by = 0.05) %>%

expand.grid(x=., y=.) %>%

ggplot(aes(x=(x+pi\*sin(y)), y=(y+pi\*sin(x)))) +

geom\_point(alpha=.1, shape=20, size=1, color="black")+

theme\_void()

  
A turtle graphic:

library(TurtleGraphics)

turtle\_init()

turtle\_col("gray25")

for (i **in** 1:150) {

  turtle\_forward(dist=1+0.5\*i)

  turtle\_right(angle=89.5)}

turtle\_hide()

library(TurtleGraphics)

turtle\_init()

turtle\_col("gray25")

turtle\_right(angle=234)

for (i **in** 1:100) {

  turtle\_forward(dist=0.9\*i)

  turtle\_right(angle=144.3)}

turtle\_hide()

library(TurtleGraphics)

turtle\_init()

turtle\_col("gray25")

turtle\_setpos(48,36)

d=50

for (i **in** 1:300) {

  turtle\_forward(dist=d)

  if (i%%4==0) {

    turtle\_right(angle=75)

    d=d\*.95}

**else** turtle\_right(angle=90)}

turtle\_hide()

library(TurtleGraphics)

turtle\_init()

turtle\_col("gray25")

turtle\_setpos(50,35)

turtle\_right(angle=30)

d=25

turtle\_setpos(50-d/2,50-d/2\*tan(pi/6))

for (i **in** 1:100) {

  turtle\_forward(dist=d)

  d=d+.5

  turtle\_right(angle=120+1)}

turtle\_hide()

library(TurtleGraphics)

turtle\_init()

turtle\_col("gray25")

turtle\_do({

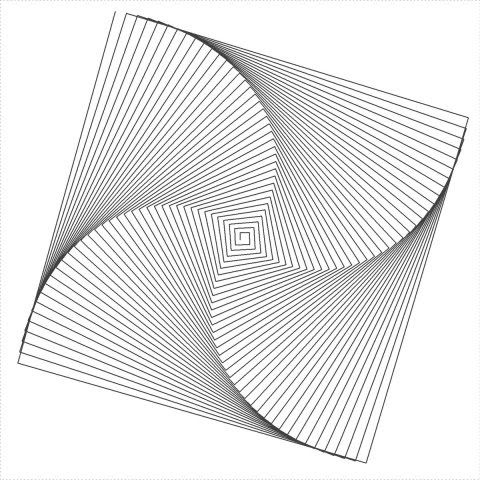
for (i in 1:150) {

turtle\_forward(dist=1+0.5\*i)

turtle\_right(angle=89.5)}

})

turtle\_hide()

  
A curve generated by a *simulated* harmonograph:

t=seq(1, 100, by=.001)

plot(exp(-0.006\*t)\*sin(t\*3.019+2.677)+

exp(-0.001\*t)\*sin(t\*2.959+2.719),

exp(-0.009\*t)\*sin(t\*2.964+0.229)+

exp(-0.008\*t)\*sin(t\*2.984+1.284),

type="l", axes=FALSE, xlab="", ylab="")

f1=jitter(sample(c(2,3),1));f2=jitter(sample(c(2,3),1));f3=jitter(sample(c(2,3),1));f4=jitter(sample(c(2,3),1))

d1=runif(1,0,1e-02);d2=runif(1,0,1e-02);d3=runif(1,0,1e-02);d4=runif(1,0,1e-02)

p1=runif(1,0,pi);p2=runif(1,0,pi);p3=runif(1,0,pi);p4=runif(1,0,pi)

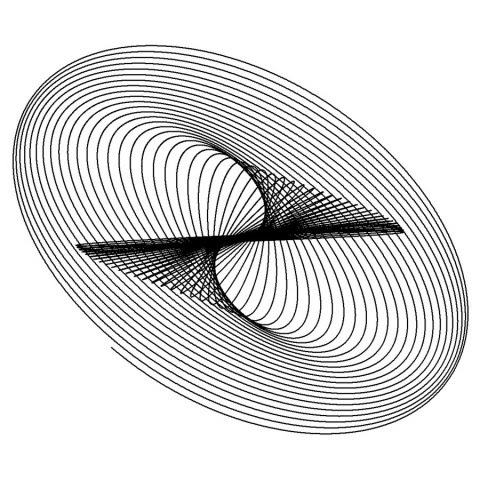
xt = **function**(t) exp(-d1\*t)\*sin(t\*f1+p1)+exp(-d2\*t)\*sin(t\*f2+p2)

yt = **function**(t) exp(-d3\*t)\*sin(t\*f3+p3)+exp(-d4\*t)\*sin(t\*f4+p4)

t=seq(1, 100, by=.001)

dat=data.frame(t=t, x=xt(t), y=yt(t))

with(dat, plot(x,y, type="l", xlim =c(-2,2), ylim =c(-2,2), xlab = "", ylab = "", xaxt='n', yaxt='n'))

  
A chord diagram of a 20×20 1-matrix:

library(circlize)

chordDiagram(matrix(1, 20, 20), symmetric = TRUE,

col="black", transparency = 0.85, annotationTrack = NULL)

par(mar = c(1, 1, 1, 1), bg="violetred4")

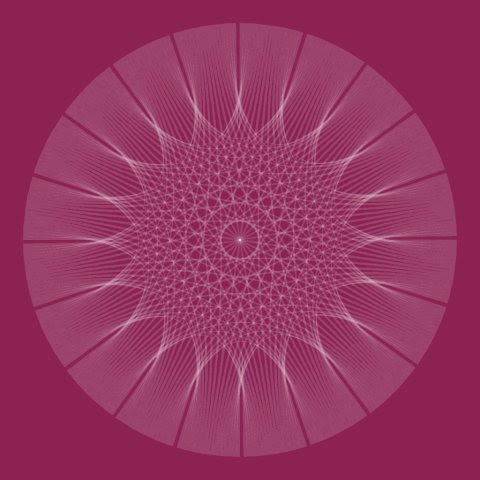
circlize::chordDiagram(matrix(1, 20, 20),

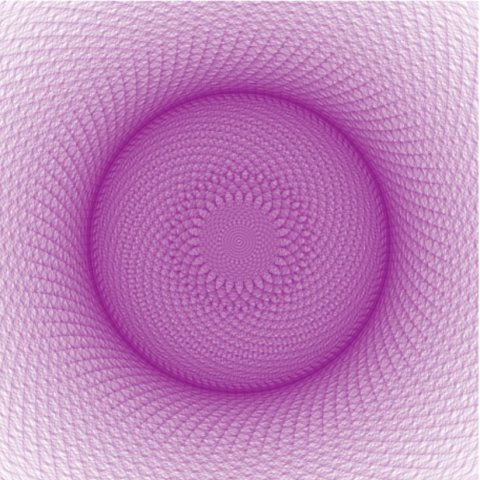
                       col="white",

                       symmetric = **TRUE**,

                       transparency = 0.85,

                       annotationTrack = **NULL**)

  
Most of them are made with ggplot2 package. I love R and the sense of wonder of how just one or two lines of code can create beautiful and unexpected patterns.

  
Furthermore, you can learn also ggplot2 while you do art.