Fractals2

Andrew Estes

9/24/2021

```
# https://rpubs.com/mstefan-rpubs/fractals

# function to create empty canvas
emptyCanvas <- function(xlim, ylim, bg="gray20") {
  par(mar=rep(1,4), bg=bg)
  plot(1,
       type="n",
       bty="n",
       xlab="", ylab="",
       xaxt="n", yaxt="n",
       xlim=xlim, ylim=ylim)
}

# example
emptyCanvas(xlim=c(0,1), ylim=c(0,1))
```

```r
# function to draw a single line
drawLine <- function(line, col="white", lwd=1) {
  segments(x0=line[1],
           y0=line[2],
           x1=line[3],
           y1=line[4],
           col=col,
           lwd=lwd)
}

# wrapper around "drawLine" to draw entire objects
drawObject <- function(object, col="white", lwd=1) {
  invisible(apply(object, 1, drawLine, col=col, lwd=lwd))
}

# example
line1 = c(0,0,1,1)
line2 = c(-3,4,-2,-4)
line3 = c(1,-3,4,3)
mat = matrix(c(line1,line2,line3), byrow=T, nrow=3)
mat
```

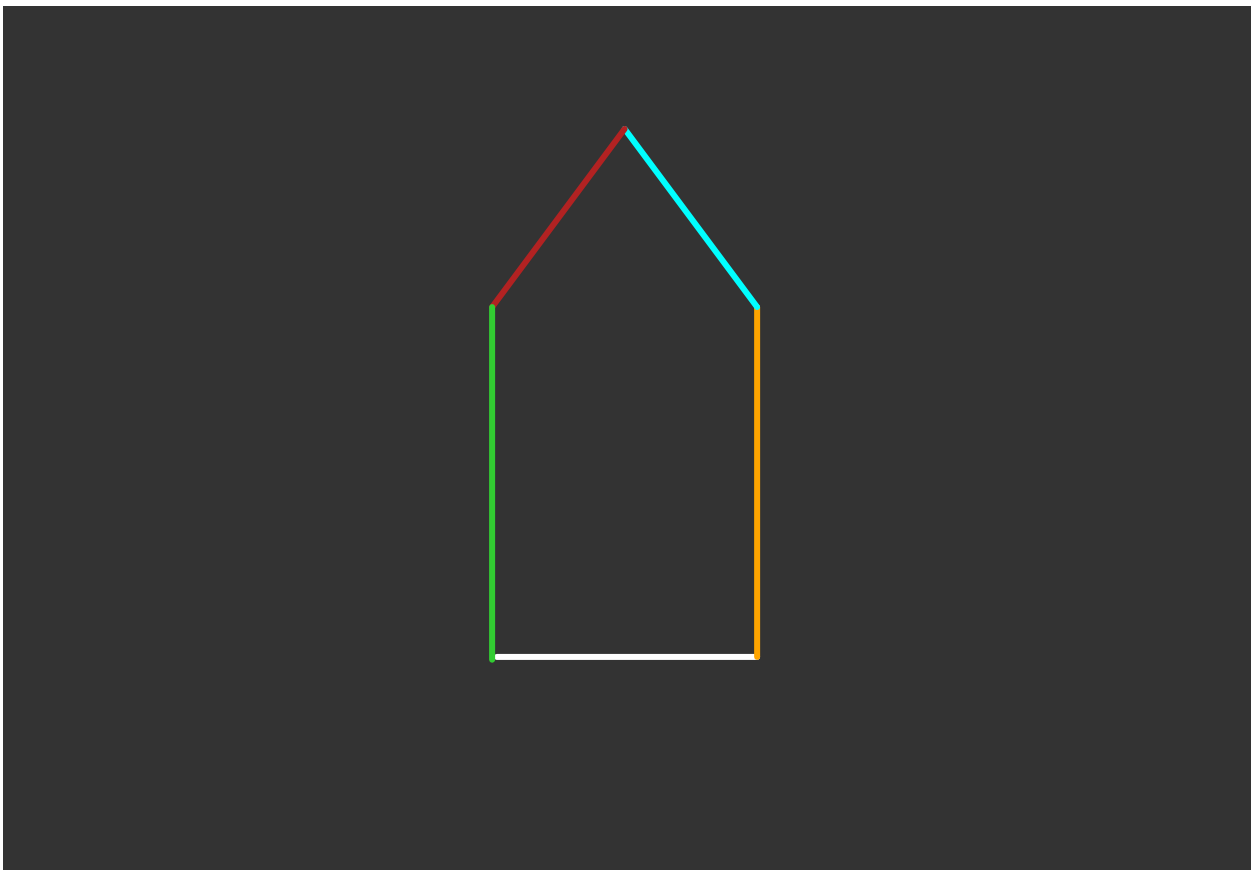
```
##      [,1] [,2] [,3] [,4]
## [1,]    0    0    1    1
## [2,]   -3    4   -2   -4
## [3,]    1   -3    4    3
```

```
# draw separately
emptyCanvas(xlim=c(-5,5), ylim=c(-5,5))
drawLine(line1)
drawLine(line2, col="orange", lwd=4)
drawLine(line3, col="cyan", lwd=6)
```

```
# draw together
emptyCanvas(xlim=c(-5,5), ylim=c(-5,5))
drawObject(mat, col="yellowgreen", lwd=3)
```

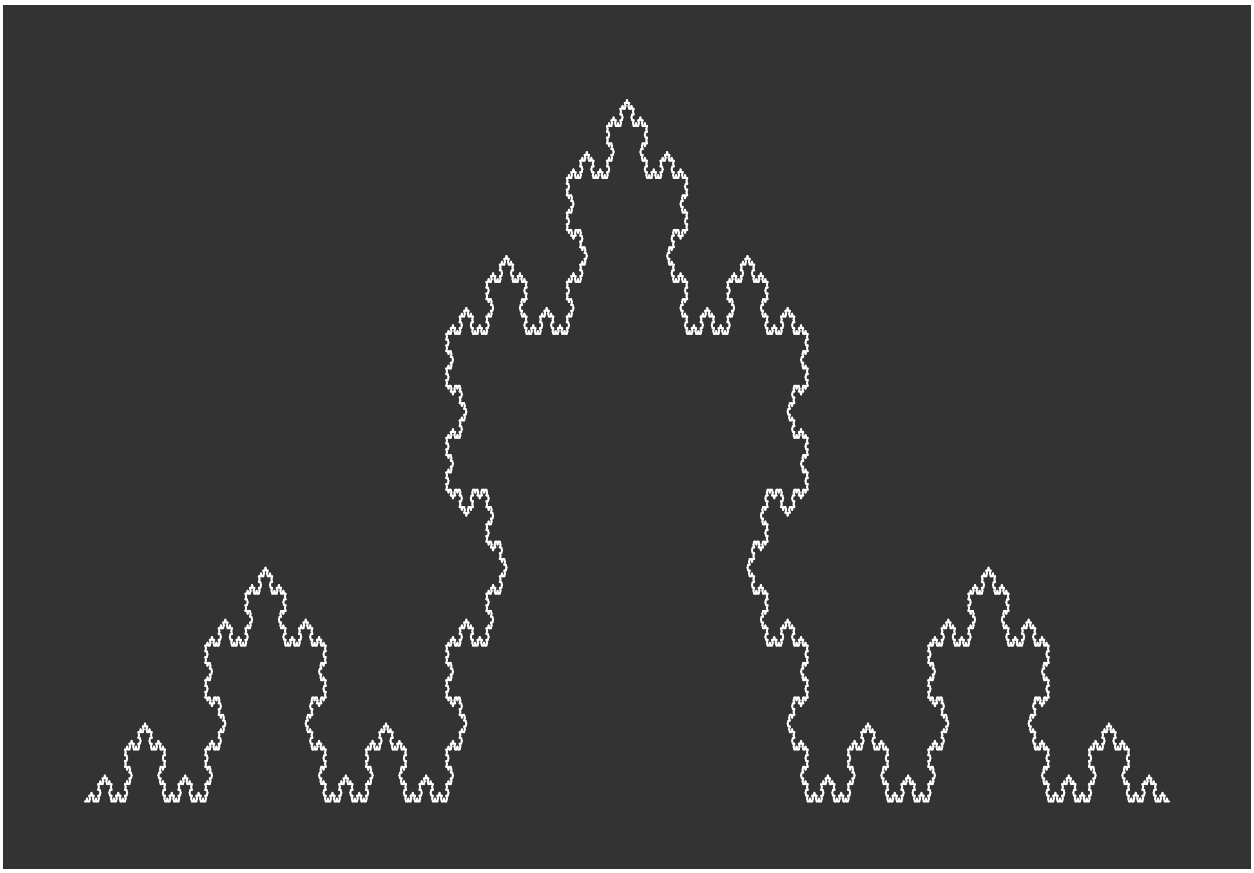
```r
# function to add a new line to an existing one
newLine <- function(line, angle, reduce=1) {

  x0 <- line[1]
  y0 <- line[2]
  x1 <- line[3]
  y1 <- line[4]

  dx <- unname(x1-x0)                    # change in x direction
  dy <- unname(y1-y0)                    # change in y direction
  l <- sqrt(dx^2 + dy^2)                 # length of the line

  theta <- atan(dy/dx) * 180 / pi        # angle between line and origin
  rad <- (angle+theta) * pi / 180        # (theta + new angle) in radians

  coeff <- sign(theta)*sign(dy)          # coefficient of direction
  if(coeff == 0) coeff <- -1

  x2 <- x0 + coeff*l*cos(rad)*reduce + dx  # new x location
  y2 <- y0 + coeff*l*sin(rad)*reduce + dy  # new y location
  return(c(x1,y1,x2,y2))

}


# example
```

```
a = c(-1.2,1,1.2,1)
b = newLine(a, angle=-90, reduce=1)
c = newLine(b, angle=45, reduce=.72)
d = newLine(c, angle=90, reduce=1)
e = newLine(d, angle=45, reduce=1.4)

# draw lines
emptyCanvas(xlim=c(-5,5), ylim=c(0,5))
drawLine(a, lwd=3, col="white")
drawLine(b, lwd=3, col="orange")
drawLine(c, lwd=3, col="cyan")
drawLine(d, lwd=3, col="firebrick")
drawLine(e, lwd=3, col="limegreen")
```

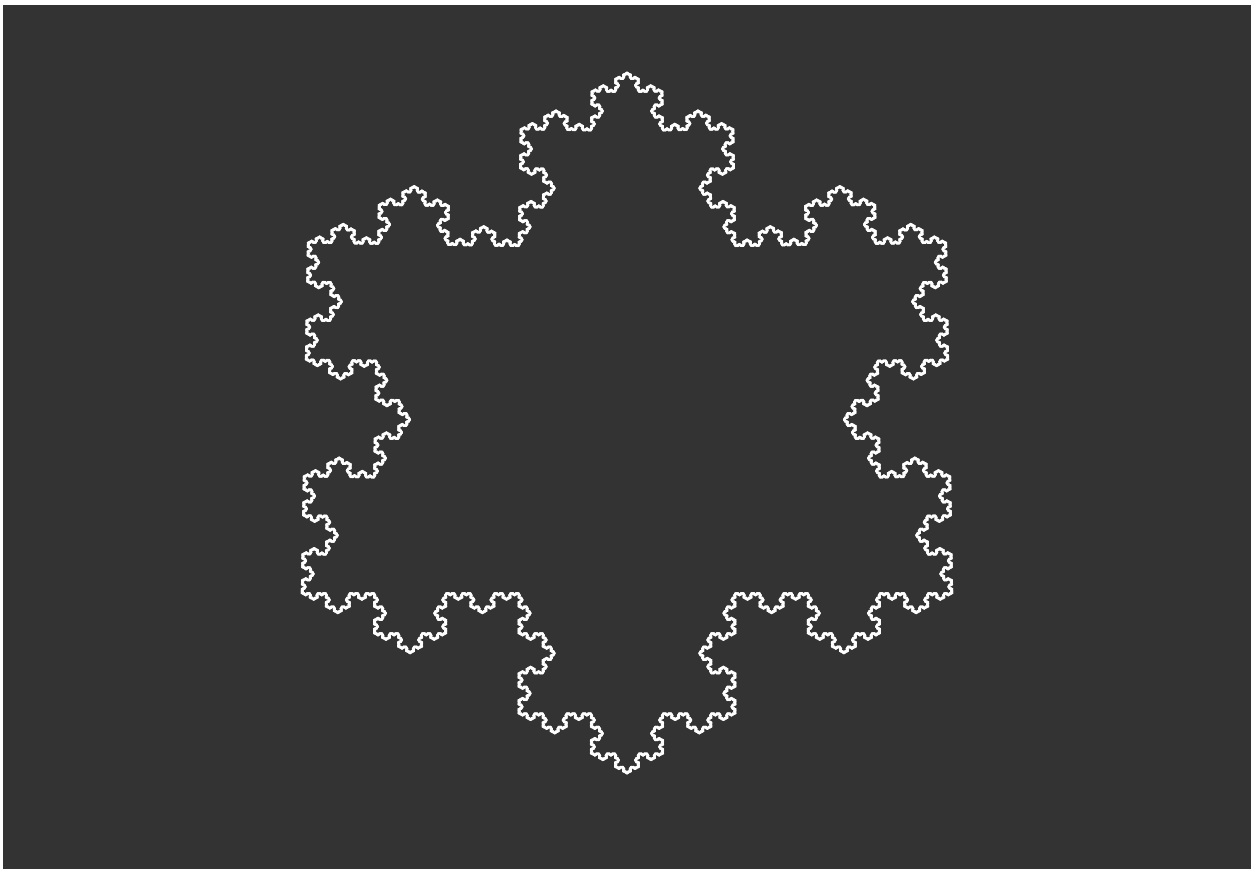


```
# function to run next iteration based on "ifun()"
iterate <- function(object, ifun, ...) {
  linesList <- vector("list",0)
  for(i in 1:nrow(object)) {
    old_line <- matrix(object[i,], nrow=1)
    new_line <- ifun(old_line, ...)
    linesList[[length(linesList)+1]] <- new_line
  }
  new_object <- do.call(rbind, linesList)
  return(new_object)
}
```

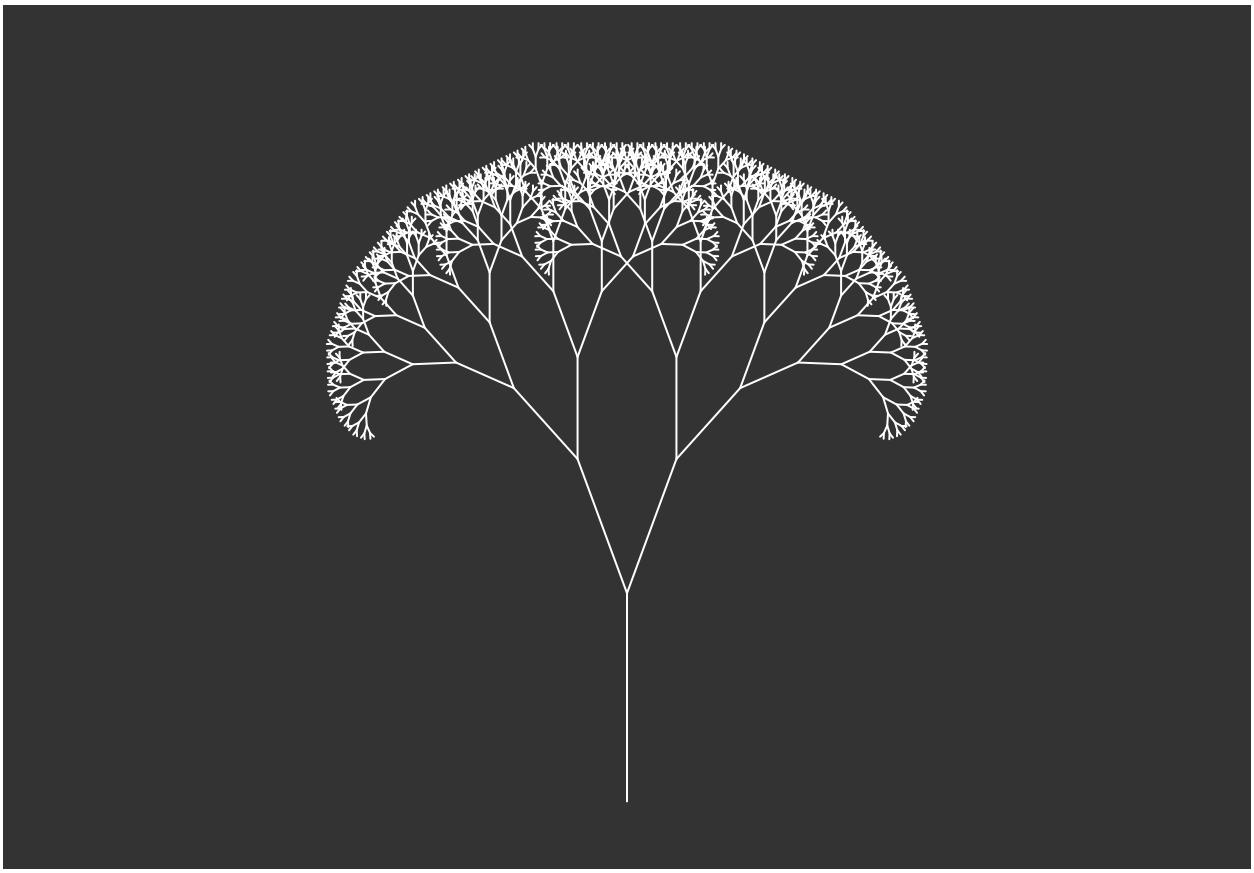
```r
# iterator function: koch curve
koch <- function(line0) {

  # new triangle (starting at right)
  line1 <- newLine(line0, angle=180, reduce=1/3)
  line2 <- newLine(line1, angle=-60, reduce=1)
  line3 <- newLine(line2, angle=120, reduce=1)
  line4 <- newLine(line3, angle=-60, reduce=1)

  # reorder lines (to start at left)
  line1 <- line1[c(3,4,1,2)]
  line2 <- line2[c(3,4,1,2)]
  line3 <- line3[c(3,4,1,2)]
  line4 <- line4[c(3,4,1,2)]

  # store in matrix and return
  mat <- matrix(c(line4,line3,line2,line1), byrow=T, ncol=4)
  return(mat)

}

# example: Koch curve (after six iterations)
fractal <- matrix(c(10,0,20,1e-9), nrow=1)
for(i in 1:6) fractal <- iterate(fractal, ifun=koch)
emptyCanvas(xlim=c(10,20), ylim=c(0,3))
drawObject(fractal)
```



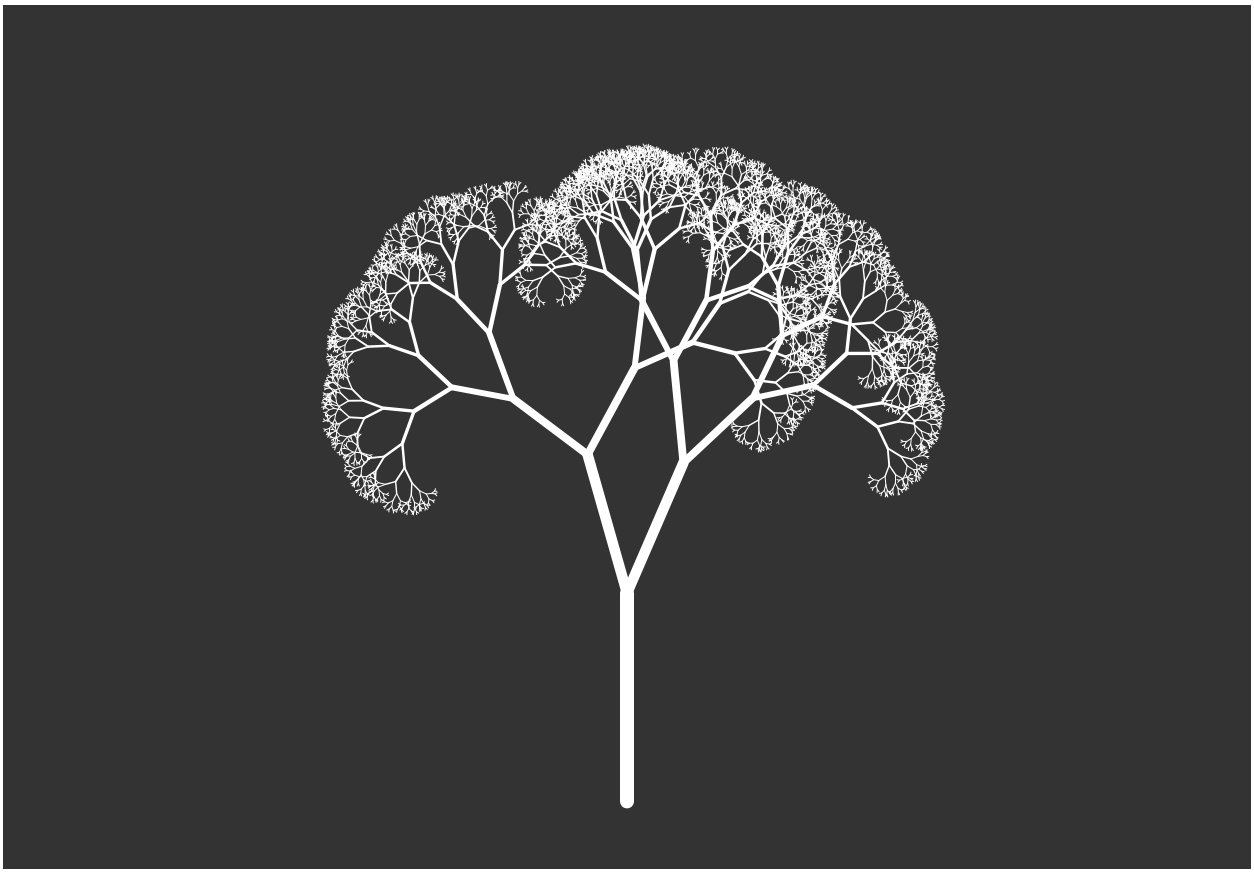
```
# Koch snowflake (after six iterations)
A <- c(0,1e-9)
B <- c(3,5)
C <- c(6,0)
fractal <- matrix(c(A,B,B,C,C,A), nrow=3, byrow=T)
for(i in 1:6) fractal <- iterate(fractal, ifun=koch)
emptyCanvas(xlim=c(-2,8), ylim=c(-2,5))
drawObject(fractal)
```



```
# iterator function: recursive tree
tree <- function(line0, angle=30, reduce=.7, randomness=0) {

  # angles and randomness
  angle1 <- angle+rnorm(1,0,randomness)  # left branch
  angle2 <- -angle+rnorm(1,0,randomness) # right branch

  # new branches
  line1 <- newLine(line0, angle=angle1, reduce=reduce)
  line2 <- newLine(line0, angle=angle2, reduce=reduce)

  # store in matrix and return
  mat <- matrix(c(line1,line2), byrow=T, ncol=4)
  return(mat)

}

# example: recursive tree (after ten iterations)
fractal <- matrix(c(0,0,0,10), nrow=1)
emptyCanvas(xlim=c(-30,30), ylim=c(0,35))
drawObject(fractal)
for(i in 1:10) {
  fractal <- iterate(fractal, ifun=tree, angle=23)
  drawObject(fractal)
}
```



```
# recursive tree (organic)
set.seed(1234)
fractal <- matrix(c(0,0,0,10), nrow=1)
emptyCanvas(xlim=c(-30,30), ylim=c(0,35))
lwd <- 7
drawObject(fractal, lwd=lwd)
for(i in 1:12) {
  lwd <- lwd*0.75
  fractal <- iterate(fractal, ifun=tree, angle=29, randomness=9)
  drawObject(fractal, lwd=lwd)
}
```



```
# example: recursive tree
Z <- c(0,0)
A <- c(1e-9,5)
B <- c(5,-1e-9)
fractal <- matrix(c(Z,A,Z,B,Z,-A,Z,-B), nrow=4, byrow=T)
emptyCanvas(xlim=c(-20,20), ylim=c(-20,20))
drawObject(fractal)
for(i in 1:11) {
  fractal <- iterate(fractal, ifun=tree, angle=29, reduce=.75)
  drawObject(fractal, col=i+1)
}
```