Markov Chain Simulation

```r
# generates rows of size P which are valid probability distributions
r0 <- function(M){
  prob <- runif(M,0,1)
  prob/sum(prob) # return normalized random row vector
}

r1 <- function(M){
  prob <- runif(M,0,1)
  num_zeros <- sample(1:M,1)
  choices <- sample(1:M, num_zeros)
  prob[choices] <- 0
  prob/sum(prob) # return normalized random row vector
}
```

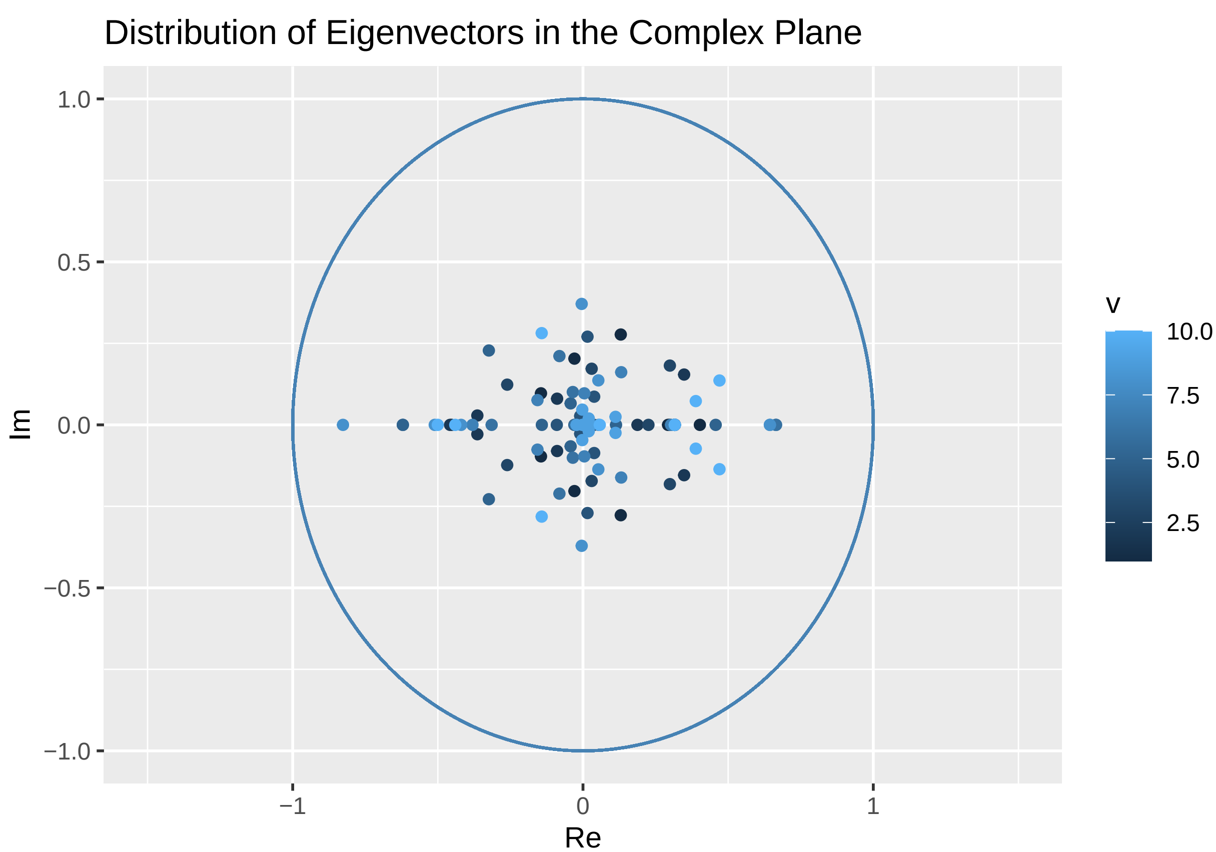
```r
# initialize random P
rand_M <- function(M,row_fxn){
  P <- matrix(rep(NA, M * M), ncol = M)  # create transition matrix
  for(i in 1:M){P[i,] = row_fxn(M)}
  P
}
```

```r
M <- 10
P <- rand_M(M,r1)
P
```

```
##            [,1]       [,2]       [,3]        [,4]       [,5]      [,6]
##  [1,] 0.0000000 0.00000000 0.15117582 0.000000000 0.00000000 0.0000000
##  [2,] 0.1970193 0.03594690 0.04547135 0.172753928 0.07741439 0.1422434
##  [3,] 0.0000000 0.00000000 0.34270005 0.006826629 0.29199345 0.0000000
##  [4,] 0.0000000 0.00000000 0.06670468 0.073043993 0.00000000 0.0000000
##  [5,] 0.0000000 0.00000000 0.00000000 1.000000000 0.00000000 0.0000000
##  [6,] 0.0000000 0.00000000 0.46057466 0.000000000 0.00000000 0.5394253
##  [7,] 0.1941519 0.12001876 0.19176819 0.095219105 0.18478533 0.0000000
##  [8,] 0.0000000 0.17619292 0.00000000 0.164998105 0.28309423 0.0000000
##  [9,] 0.1048788 0.11419434 0.16956800 0.061386920 0.11653360 0.0000000
## [10,] 0.1216580 0.05233542 0.04164266 0.017754980 0.21599197 0.3087093
##             [,7]       [,8]       [,9]      [,10]
##  [1,] 0.21225929 0.00000000 0.63656489 0.00000000
##  [2,] 0.00000000 0.09587366 0.08530077 0.14797635
##  [3,] 0.22581815 0.06989863 0.00000000 0.06276309
##  [4,] 0.00000000 0.00000000 0.86025133 0.00000000
##  [5,] 0.00000000 0.00000000 0.00000000 0.00000000
##  [6,] 0.00000000 0.00000000 0.00000000 0.00000000
##  [7,] 0.15971701 0.04327551 0.00000000 0.01106416
##  [8,] 0.13881649 0.00000000 0.23689825 0.00000000
##  [9,] 0.14164875 0.01311641 0.16771792 0.11095529
## [10,] 0.06422282 0.03541740 0.14226745 0.00000000
```

```r
eig_P <- eigen(P)
eig_vectors <- eig_P[2]
evec <- data.frame(eig_vectors)

cols <- 3 # set 3 to hold (re,im) pair and whose row it belongs to
complex <- matrix(rep(NA,cols*M*M), ncol = cols)
colnames(complex) <- c("Re","Im","v")
for(i in 1:M){
  for(j in 1:M){
    curr <- evec[i,j]
    complex[M*(i-1) + j, ] <- c(Re(curr),Im(curr),i)
  }
}
```

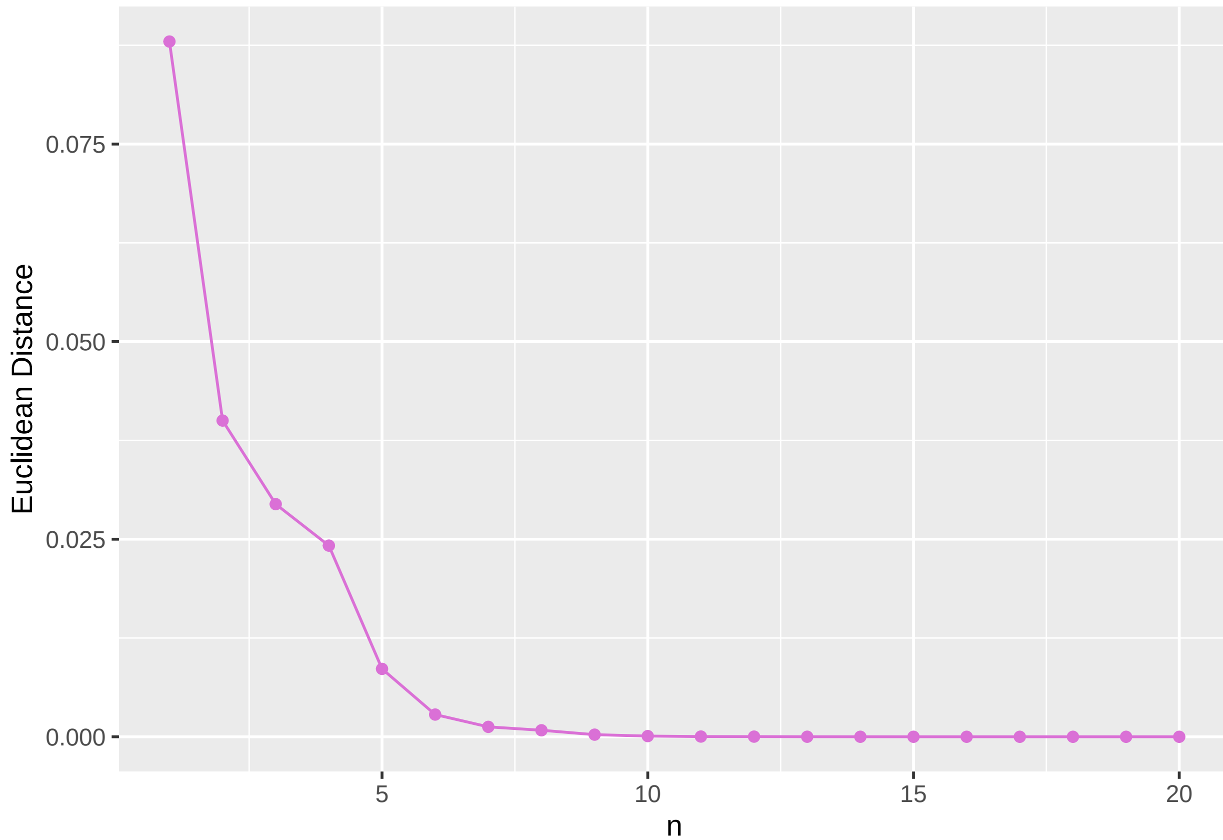
```r
r <- 1
ep <- 0.5
ggplot(complex) +
  geom_point(aes(x = Re, y = Im, color = v)) +
  labs(x = "Re", y = "Im", title = "Distribution of Eigenvectors in the Complex Plane") +
  xlim(-(r+ep),r+ep) + ylim(-r,r) +
  ggforce::geom_circle(aes(x0=0,y0=0,r=r), color = "steelblue")
```




```r
set.seed(23)
it <- 20 # set number of iterations of transition matrix
pi <- r1(M) # create some initial distribution

# simulate and record evolution of pi
```

```r
vals <- matrix(rep(NA, (M+1) * it), ncol = (M+1))
for(i in 1:it){
  vals[i, ] = c(i, pi %*% matrix.power(P,i))
}

# rename the columns
str_vec <- rep(NA, M)
for(i in 1:M){str_vec[i] = paste("x",i,sep="")}
colnames(vals) <- c("n",str_vec)

#store the values in a dataframe
vals_ <- data.frame(vals)
vals <- subset(vals_, select = -c(n))
```

```r
#plot difference from a reference/stationary distribution
ref_dist <- vals[it,]
diff <- rbind(vals,ref_dist)

dist_vec <- rep(0, it)
for(i in 1:it){
  curr_dist <- stats::dist(diff[c(i,it+1),], method = "euclidean")
  dist_vec[i] <- curr_dist
}

dist_vec <- data.frame(dist_vec)
dist_plot <- ggplot(dist_vec, mapping = aes(x = 1:it, y = dist_vec)) +
  geom_point(color = col_str) + geom_line(color = col_str) +
  labs(x = "n", y = "Euclidean Distance")
dist_plot
```


Euclidean Distance
0.075
0.050
0.025
0.000
5
10
15
20
n