Markov Chain Simulation

```
# 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
}
```

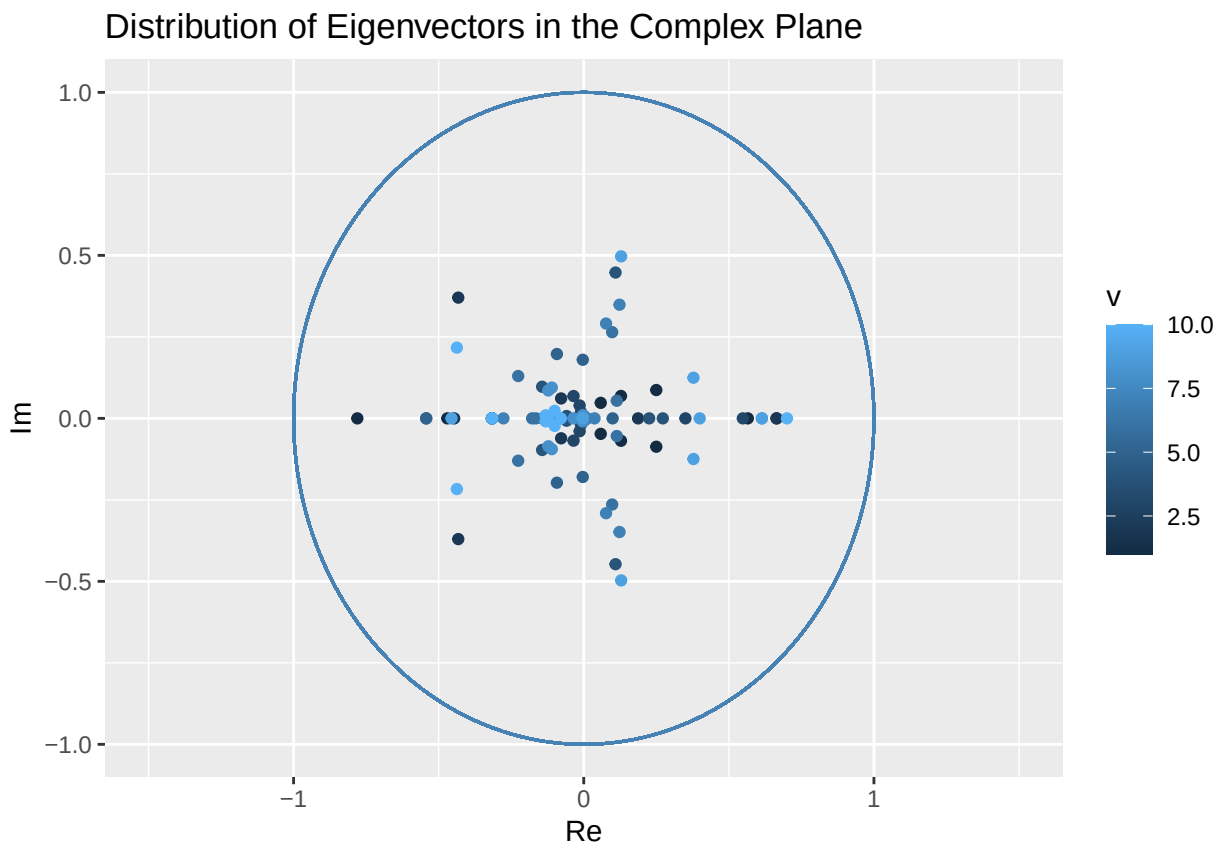
```
# initialize random P
rand_M <- function(M,row_fxn){
  P <- matrix(rep(NA, M * M), ncol = M)  # create transition matrix
  for(i in 1:M){P[i,] = row_fxn(M)}
  P
}
```

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

```
##             [,1]       [,2]      [,3]       [,4]      [,5]       [,6]
##  [1,] 0.00000000 0.00000000 1.0000000 0.00000000 0.0000000 0.00000000
##  [2,] 0.00000000 0.00000000 0.0000000 0.00000000 0.0000000 0.00000000
##  [3,] 0.48141970 0.00000000 0.0000000 0.05361701 0.0000000 0.00000000
##  [4,] 0.22527154 0.00000000 0.0000000 0.00000000 0.0000000 0.00000000
##  [5,] 0.17882660 0.17109919 0.0547007 0.08549995 0.1702905 0.16900090
##  [6,] 0.00000000 0.00000000 0.0000000 0.00000000 0.0000000 0.00000000
##  [7,] 0.00000000 0.00000000 0.0000000 0.00000000 0.0000000 0.69568322
##  [8,] 0.12450877 0.07568462 0.1414883 0.02558089 0.1868578 0.11768046
##  [9,] 0.00000000 0.51835664 0.0000000 0.03857613 0.0000000 0.07590837
## [10,] 0.06743264 0.00000000 0.1163968 0.17472922 0.0632536 0.00000000
##             [,7]       [,8]      [,9]     [,10]
##  [1,] 0.00000000 0.00000000 0.0000000 0.0000000
##  [2,] 1.00000000 0.00000000 0.0000000 0.0000000
##  [3,] 0.00000000 0.17660224 0.0000000 0.2883610
##  [4,] 0.13673930 0.36882267 0.0994232 0.1697433
##  [5,] 0.00000000 0.00000000 0.1705821 0.0000000
##  [6,] 0.00000000 1.00000000 0.0000000 0.0000000
##  [7,] 0.12371501 0.00000000 0.0000000 0.1806018
##  [8,] 0.06981941 0.09616244 0.1622174 0.0000000
##  [9,] 0.00000000 0.36715886 0.0000000 0.0000000
## [10,] 0.13290660 0.17147337 0.1630767 0.1107311
```

```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)
  }
}
```

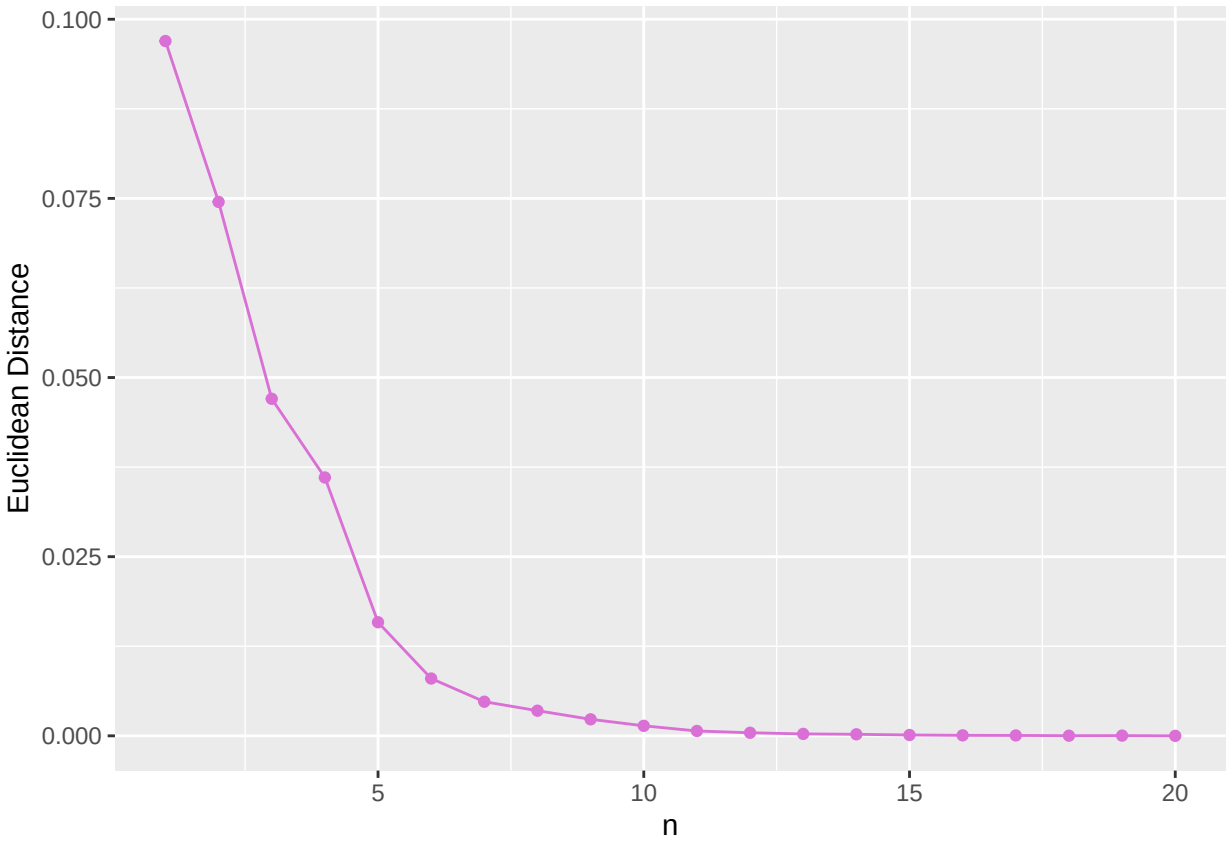
```r
r <- 1
ep <- 0.5
ggplot(complex) +
  geom_point(aes(x = Re, y = Im, color = v)) +
  labs(x = "Re", y = "Im", title = "Distribution of Eigenvectors in the Complex Plane") +
  xlim(-(r+ep),r+ep) + ylim(-r,r) +
  ggforce::geom_circle(aes(x0=0,y0=0,r=r), color = "steelblue")
```




```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
```


0.100
0.075
0.050
0.025
0.000
Euclidean Distance
5
10
15
20
n