# Refactoring matrices.R

Ali Taqi

## Normal Matrices

### Basic Example

```r
# Set seed
set.seed(23)
# Set parameters
M <- 5
mu <- 0
sd <- 2
normal_args <- c(mu, sd)
# Generate matrix
P <- RM_normal(M, normal_args = c(mu, sd), symm = T)
```

```r
entries_P <- vectorize_matrix(P)
normal_params(entries_P)
```

```
## [1] "Mean: 0.915"
## [1] "Standard Deviation: 1.821"
```

```
##            [,1]        [,2]       [,3]        [,4]      [,5]
## [1,]  0.3864247 -0.86936422  1.8265342  3.58677618  1.993210
## [2,] -0.8693642 -0.55617257  2.0384110  0.09087436  3.151559
## [3,]  1.8265342  0.09087436 -0.5773773  0.96310057 -2.432753
## [4,]  2.0384110  0.96310057  3.1515592 -1.19862562  2.589156
## [5,]  3.5867762  1.99321021 -2.4327529  2.58915566 -1.061640
```

```
## [1] FALSE
```

### Symmetric Normal Matrices

```r
# Set seed
set.seed(23)
# Set parameters
M <- 5
mu <- 0
sd <- 1
normal_args <- c(mu, sd)
# Generate matrix
P <- RM_normal(M, normal_args, symm = T)
```

```r
if(bloud){P}
```

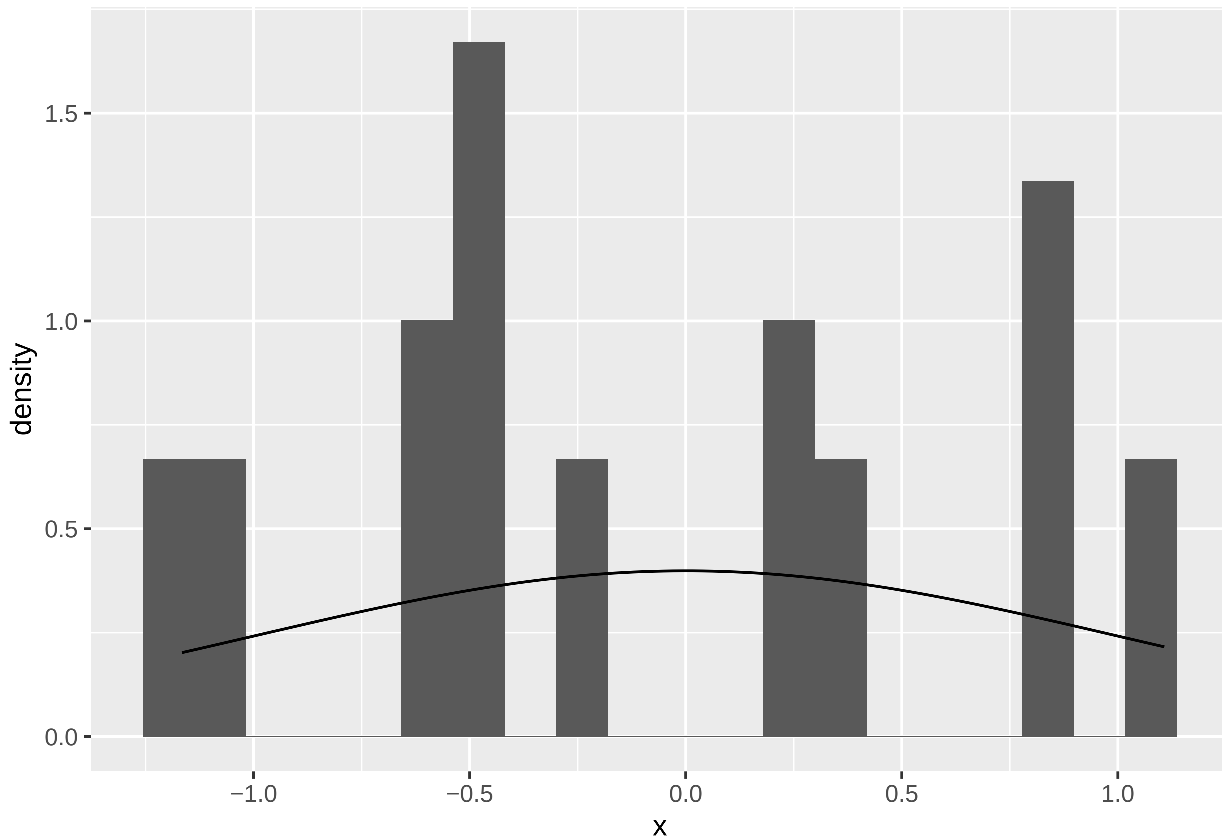
```
##           [,1]       [,2]       [,3]       [,4]       [,5]
## [1,] 0.1932123  1.1074905  0.2182885  0.3081369  0.8353912
## [2,] 1.1074905 -0.2780863 -1.0465353 -0.5201783 -0.5660151
```

```
## [3,] 0.2182885 -1.0465353 -0.2886886 -0.4423138  0.7884194
## [4,] 0.3081369 -0.5201783 -0.4423138 -0.5993128 -1.1659293
## [5,] 0.8353912 -0.5660151  0.7884194 -1.1659293 -0.5308200
```

```r
is_symmetric(P)
```

```
## [1] TRUE
```

```r
if(!bplot){visualize_normal_entries(P, normal_args)}
```



## Stochastic Matrices

### Sparse Stochastic Matrices

```
# Set seed
set.seed(23)
# Set parameters
M <- 3
# Generate matrix
P <- RM_stoch(M, symm = F, sparsity = T)
```

```
##      [,1]      [,2]      [,3]
## [1,]    0 0.4019552 0.5980448
## [2,]    0 1.0000000 0.0000000
## [3,]    1 0.0000000 0.0000000
```

```
## [1] TRUE
```

```
## [1] FALSE
```

### Non-sparse Stochastic Matrices

```
# Set seed
set.seed(23)
# Set parameters
M <- 3
# Generate matrix
P <- RM_stoch(M, symm = F, sparsity = F)
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.5095594 0.1971352 0.2933055
## [2,] 0.3637477 0.4193927 0.2168595
## [3,] 0.3463251 0.3515677 0.3021073
```

```
## [1] TRUE
```

```
## [1] FALSE
```

**Symmetric Stochastic Matrices**

**Sparse Symmetric Stochastic Matrices**

```
# Set seed
set.seed(23)
# Set parameters
M <- 5
# Generate matrix
P <- RM_stoch(M, symm = T, sparsity = T)
```

```
##            [,1] [,2]      [,3] [,4] [,5]
## [1,] -0.5621547    0 0.5621547    0    1
## [2,]  0.0000000    1 0.0000000    0    0
## [3,]  0.5621547    0 0.4378453    0    0
## [4,]  0.0000000    0 0.0000000    1    0
## [5,]  1.0000000    0 0.0000000    0    0
```

```
## [1] TRUE
```

```
## [1] TRUE
```

**Non-sparse Symmetric Stochastic Matrices**

```
# Set seed
set.seed(23)
# Set parameters
M <- 3
mu <- 1
sd <- 2
# Generate matrix
P <- RM_stoch(M, symm = T, sparsity = F)
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.2899272 0.3637477 0.3463251
## [2,] 0.3637477 0.2846846 0.3515677
## [3,] 0.3463251 0.3515677 0.3021073
```

```
## [1] TRUE
```

```
## [1] TRUE
```

## Tridiagonal Matrices

### Basic example

```r
# Set seed
set.seed(23)
# Set parameters
M <- 3
# Generate matrix
# Need not be symmetric : fix laater
P <- RM_trid(M)
```

```
##           [,1]       [,2]      [,3]
## [1,] 0.3864247  1.7933881 0.0000000
## [2,] 1.7933881 -0.8693642 0.9966051
## [3,] 0.0000000  0.9966051 1.8265342
```

### Symmetric Tridiagonal Matrices

```r
# Set seed
set.seed(23)
# Set parameters
M <- 3
# Generate matrix
P <- RM_trid(M)
# Need not be symmetric : fix later
```

## p-Sparse Matrices

### Basic example

```r
# Set seed
set.seed(23)
# Set parameters
M <- 3
p <- 0.2
# Generate matrix
P <- RM_erdos(M, p, stoch = F)
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.0000000 0.2230729 0.3318966
## [2,] 0.0000000 0.0000000 0.0000000
## [3,] 0.8459473 0.0000000 0.5181206
```

```
## [1] FALSE
```

### Stochastic p-Sparse Matrices

```r
# Set seed
set.seed(23)
# Set parameters
M <- 3
p <- 0.2
# Generate matrix
P <- RM_erdos(M, p, stoch = T)
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.0000000 0.4019552 0.5980448
## [2,]       NaN       NaN       NaN
## [3,] 0.6201651 0.0000000 0.3798349
```

```
## [1] TRUE
```

# Notation

Suppose we have a $M \times M$ square matrix $\mathbf{P}$ (for some $M \in \mathbb{N}$) on a field $F$. We notate $\mathbf{P} \in \mathcal{M}_F[M^2]$.

Take $\mathbf{P} \in \mathcal{M}_F[M^2]$.

## Structural Properties of Matrices

If $\mathbf{P}$ is symmetric, then its upper triangle is equal to the lower triangle.

If $\mathbf{P}$ is tridiagonal, then it is a band matrix of width 1.

## Entry-wise Properties of Matrices

If $\mathbf{P}$ is row-stochastic, then $\forall i : \sum_j p_{ij} = 1$.

```
RM_stoch <- function(M, symm = F, sparsity = F){...}
```

If $\mathbf{P}$ is $\mathcal{N}(\mu, \sigma^2)$, then its entries satisfy $p_{ij} \sim \mathcal{N}(\mu, \sigma^2)$.

```
RM_normal <- function(M, normal_args = c(0,1), symm = F){...}
```

If $\mathbf{P}$ is $p-$sparse, then $\forall i, j \in S_M : p_{ij}/c \sim \mathrm{Bern}(p)$ for some $c \in \mathbb{R}$.

```
RM_erdos <- function(M, p_sparse){...}
```