Matrix Refactor

Ali Taqi

```r
# Global parameters
M <- 3
```

Normal Matrices

```r
# Set seed
set.seed(23)
# Set parameters
mu <- 1
sd <- 2
# Generate matrixx
P <- RM_normal(M, normal_args = c(mu, sd), symm = F)
P
```

```
##           [,1]      [,2]     [,3]
## [1,] 1.3864247 0.1306358 2.826534
## [2,] 4.5867762 2.9932102 3.214981
## [3,] 0.4438274 3.0384110 1.090874
```

Symmetric Normal Matrices

```r
# Set seed
set.seed(23)
# Set parameters
mu <- 1
sd <- 2
# Generate matrixx
P <- RM_normal(M, normal_args = c(mu, sd), symm = T)
P
```

```
##           [,1]     [,2]      [,3]
## [1,]  9.928535 15.83749  4.095652
## [2,] 15.837494 40.33393 14.637480
## [3,]  4.095652 14.63748 10.618931
```

Stochastic Matrices

Sparse

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

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

Non-sparse

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

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

Symmetric Stochastic Matrices

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

```
##           [,1]      [,2] [,3]
## [1,] 0.5192256 0.4019552    0
## [2,] 0.4019552 1.0000000    0
## [3,] 0.0000000 0.0000000    1
```

Non-sparse

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

```
##           [,1]      [,2]      [,3]
## [1,] 0.3845411 0.3316342 0.3343892
## [2,] 0.3316342 0.3552307 0.3389347
## [3,] 0.3343892 0.3389347 0.3348097
```

Tridiagonal Matrices

```r
# Set seed
set.seed(23)
# Generate matrix
P <- RM_trid(M)
P
```

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

p-Sparse Matrices

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

Stochastic p-Sparse Matrices

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

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){...}
```