INSTRUCTION FOR INSTALLING THE R-PACKAGE 'RBVNF'

FOR BAYESIAN REGRESSION FOR DIRECTIONAL RESPONSES

INSTALLING 'RBVNF' FOR GITHUB

- install.packages("devtools")
- library(devtools)

The R package (Beta Version)

- install_github("subhadippal2019/RBVNF")
- library(RBVNF)

INSTALL AND LOAD THE FOLLOWING LIBRARIES

- library(gsl)
- library(Bessel)
- library(ghyp)
- library(ggplot2)
- library(Rfast)
- library(mvtnorm)

- # The following Libraries are for associated PLOTS
- library(cowplot)
- library(grid)
- library(gridExtra)

Now the Initial Setup is complete, the r functions are available to be utilized to analyze the data.

Rest of the slides are some basic testing.

CHECKING THE FUNCTION

- library(RBVNF)
- help(MCMC_Dir_regression_sampler_V1)

```
library(Rfast)
library(Bessel)
library(gsl)
 library(ghyp)
n=500 # NUmber of the samples
p=4 # NUmber of the regression covariates
d=3 # Number of direcions in the direcional data
```

RUNNING THE SAMPLER

```
data_lst = Data_generator_vnf_reg(n=n,
                                  p=p,
                                  d=d, concentration_factor = 1,
                                   beta_factor = 5)
Y = data_lst$Y;
X=data_lst$X
```

```
lst=MCMC_Dir_regression_sampler_V1(Y=Y, X=X, MCSamplerSize =100)
```

```
i=1;j= 1
library(ggplot2)
library(cowplot)
Plot_MCMC_Diag_Triplet(lst$MC$Mc_Beta[,i,j],y_lab_text = bquote(beta[.(i)][.(j)]))
```

Running the Lasso Optimization and Crossvalidation

CHECKING THE FUNCTION

- library(RBVNF)
- help(MCMC_Dir_regression_sampler_V1)

```
library(Rfast)
library(Bessel)
library(gsl)
 library(ghyp)
n=1000 # Number of the samples
p=10 # Number of the regression covariates
d=3 # Number of directions in the directional data
```

GENERATING SIMULATED DATA

```
data_lst = Data_generator_vnf_reg_sparse (n=n,
                                          p=p,
                                          d=d,
                                          SetUp = 2,
                                          NumOfNonZeroBeta=c(3, 3, 10))

Y = data_lst$Y;
X=data_lst$X
```

```
      [,1]     [,2]     [,3]
[1,] 6.863187 5.308869 9.037498
[2,] 4.778856 8.798885 3.131229
[3,] 7.162588 3.121234 3.196535
[4,] 0.000000 0.000000 0.000000
[5,] 0.000000 0.000000 0.000000
[6,] 0.000000 0.000000 0.000000
[7,] 0.000000 0.000000 0.000000
[8,] 0.000000 0.000000 0.000000
```

NumOfNonZeroBeta=c(3, 3, 10)

Only 3 nonzero
In Each Column
Change it to get configuration

Nonzero elements are
between3 and 10
Change it get other type of
simulated data

RUNNING CROSS VALIDATION AND PLOTTING

```
Cv_object<-EM_BLASSO_Dir_regression_optimizer_V1.cv(Y=Y,
                                                    X=X,
                                                    beta_init = NULL,
                                                    Max_EM_iter=1000,
                                                    cv_k_fold = 10,        # cross validation fold
                                                    cv_lambda_n = 50,  # number of lambda to test
                                                    epsilon_lambda_range_min = .001, # Ratio between min and max lambda
                                                    lambda_Range_Type = 2
                                                     )
```

```
plot.cv.Dir_Lasso_Reg(Cv_object)
```

```
Library(ggplot2)
Library(ggfx)
plot.cv.Dir_Lasso_Reg_gg(Cv_object, color_theme = 2)
```

RUNNING THE EM LASSO OPTIMIZER

```
betaEst1<-EM_BLASSO_Dir_regression_optimizer_V1(Y = Y, X = X, lasso_lambda =
Cv_object $lambda.min)
```

```
betaEst2<-EM_BLASSO_Dir_regression_optimizer_V1(Y = Y, X = X, lasso_lambda =
max(Cv_object $lambda.1se))
```

Checking number of zero elements in **betaEst1** and **betaEst2**