Poisson-PDNN manual

**1. Description**

This software implements all the methods in the paper “Modelling non-uniform read distribution based on the Poisson-PDNN model in RNA-Seq data. This software cites some functions in R package *mseq*.

**2. Functions in the Poisson-PDNN model**

*2.1NewtonCV*

This function applies Newton-Decline algorithm for cross-validation. T The inputs of this function are the data after using the function of *expData* which is provided in *mseq* and the initial parameters. The output of this function is R2 and the positional weight or stacking energy.

*2.2 Newton*

Optimize the least square method to find optimal solution. The inputs of this function are weight parameters and the matrix of the data after using the function of *expData* which is provided in *mseq*.

*2.3 getNullCount\_pdnn*

Get the counts of reads for each position under the null hypothesis (each positions have the same weights)

*2.4 getPredCount\_pdnn*

Get the predicted counts using the results of the positional weight or stacking energy after using the function of *Newton*.

*2.5 getDev\_pdnn*

Calculate the deviance under Poisson-PDNN model

*2.6 energy\_weight*

Use the output of *expData* to get the energy weight in order to use the function *Energy* to calculate the optimized stacking energy.

*2.7 Energy*

This function is used to calculate the stacking energy between two nucleotides and gets a matrix of stacking energy.

*2.8 compute\_exe*

This function is used to compute R2 and train the weight and energy parameters. It first read the data set. Then it uses *expData* to get a csv file named data1 as an input of the *energy* function. After that, it gets an energy matrix called data11 which is the input of *NewtonCV* function. It gets R2, positional weight and stacking energy after executing the *NewtonCV* function.

*2.9 geneExpression*

This function is used to calculate the relative transcript abundance. The inputs are the test data, the positional weight and stacking energy which is the output of *compute\_exe*.

**3. Test data**

Download the test dataset “g1.tar” here. This file contains two datasets: “g1\_train.csv” and “g1\_test.csv”. These two datasets both come from the dataset “g1.csv” provided by *mseq*, which consists of the top 100 single-isoform genes with the highest expression levels. We divided this file into two parts. The training dataset consists of the first 40 highest expression genes and the testing dataset consists of the other 60 genes.

**4. Output description**

The result of the function *compute\_exe* is a list which contains the R2 and the positional weight and stacking energy. The result of function *geneExpression* is the relative transcript abundance.

**5. Examples**

data\_test=read.csv("g1\_test.csv")

data\_train=read.csv(“g1.train.csv”)

result=compute\_exe(data\_train,eps=0.01,iter=3000)

b=result$weight

e=result$energy

gene\_expression=geneExpression(data\_train,b,e)