NRselect: A R package for selecting the best way for RNA-Seq data pretreatment

1.INTRODUCTION

Before submitting scRNA-Seq data to unsupervised clustering methods, normalization and dimension tools are often used to pretreat scRNA-Seq data. Different combinations of normalization and dimensional reduction tools will strongly influence the results of clustering and cell type enrichment analysis, so there is a need to develop a tool to find the best combination of normalization and dimension reduction.

NRselect is a R package developed to select the best combination of normalization and dimensional reduction tools. To achieve this, the NRselect R package contains two functions. First, we develop an reliable index NRindex to evaluate the quality of a producing data (data that produced by a combination of normalization and dimension reduction tools), then, function NRselect will test all possible combinations and for each combinations, calculate NRindex of the producing data. The combination that has the highest NRindex will be chosen as the best combination.

2.NRselect

2.1 Requirements

Two things need to be prepared before using NRselect

- Gene expression data: NRselect only receive a data matrix that each raw represents a gene, each column represents a cell.
- Normalization and dimension reduction tools: before using NRselect, make sure all the normalization and dimension reduction tools are installed. And make sure you know how to call these tools and how to transform the results of these tools into right format.

2.2 Examples

The example shows how to use NRselect to choose the best normalization and dimension reduction tools. First, we load the required packages, and the dataset. We load Biase which contains 20215 genes and 90 cells.

```
>library(Rcpp)
>library(dplyr)
>library(NRselect)
>data(Biase)
```

Second, create two arrays `Norm` and `Red`, and assign them with correct commands of normalization and dimension reduction tools you want to test. The arguments that represents the input data of normalization and dimension reduction tools should be replaced by `Data`. For example, if you want to use R function `scale()`, string "`scale(Data)`"should be included in array `Norm`. Notice that one normalization tool is only allowed using one line to call it. If your normalization tool needs more than one line to call, please write a function and use one line to call this function. For example, if you want to test a normalization tool like this.

```
>Data=scale(Data)
>log(as.matrix(Data)+1)
```

You need to write a function to wrap it.

```
My_normalization<-function(Data)
{
    Data=scale(Data)
    return ((log(as.matrix(Data)+1))
}
```

And add string "`My_normalization(Data)`" into `Norm`.

The rule to create `Red` is the same as `Norm`. We test three normalization tools(EDASeq, scale, Linnorm) and two dimension reduction tools(tSNE, sammon) in this example.

```
>Norm<-c("betweenLaneNormalization(as.matrix(Data),which=\"full\")",
"scale(Data)",
"Linnorm(as.matrix(Data))")
>Red<-c("Rtsne(t(Data),dim=2,perplexity=15)",
"sammon(d=dist(t(Data),method=\"euclidean\"),k=2)")
```

After normalization, the result will be saved in `Data`. But some normalization tools may return a complex object and cannot be analysis by following dimension reduction tools, so we should create an array NormReturn. The ith element of NormReturn should contain the command that can get the right result from ith normalization tools and change it to right form that following dimension reduction tools could analysis. In this example, all normalization tools return a matrix all dimension reduction tools can deal with matrix, so we can let all element of NormReturn be “Data”.

```
> NormReturn<-c("Data","Data","Data")
```

After dimension reduction, the result will also be saved in `Data`, and because of the bfore-mentioned reasons, we need to create an array RedReturn to make sure the output data can be analysised by NRindex. NRindex can only deal with matrix, so commands in RedReturn should change the dimension reduction result to matrix. For example, function Rtsne will return a list and use `Data$Y` to extract the element we want, so the 1st element of RedReturn in this example should be “Data$Y”. For the same reason, the second element of RedReturn should be `"Data$points"`

```
> RedReturn<-c("Data$Y","Data$points")
```

Then, we create two arrays NormName and RedName that contains name of each normalization and dimension reduction tools, and run NRselect, NRselect will return a list that contains the normalization and dimension reduction with the highest index.

```
> NormName<-c("EDASeq","scale","Linnorm")
> RedName<-c("tSNE","sammon")
```

Last, we create two strings ClusterName and ClusterNumber. ClusterName is used to choose the clustering method you want to use. NRindex provides two cluster methods: hclust and kmeans. ClusterNumber represents the number of clusters you want.

```
>ClusterName<-‘kmenas’
>CLusterNumber<-4
> res<-
NRselect(Data,Norm,Red,NormReturn,RedReturn,NormName,RedName,ClusterNam
e,ClusterNumber)
> res$list
```

The steps showed below can be time-consuming. To simplify this step, NDRindex included five normalization tools TMM, Linnorm, scale, scarn, Seurat and three dimensionality reduction tools tSNE, PCA, Sammon. If you only want to test these tools, you can just call NRindex like following.

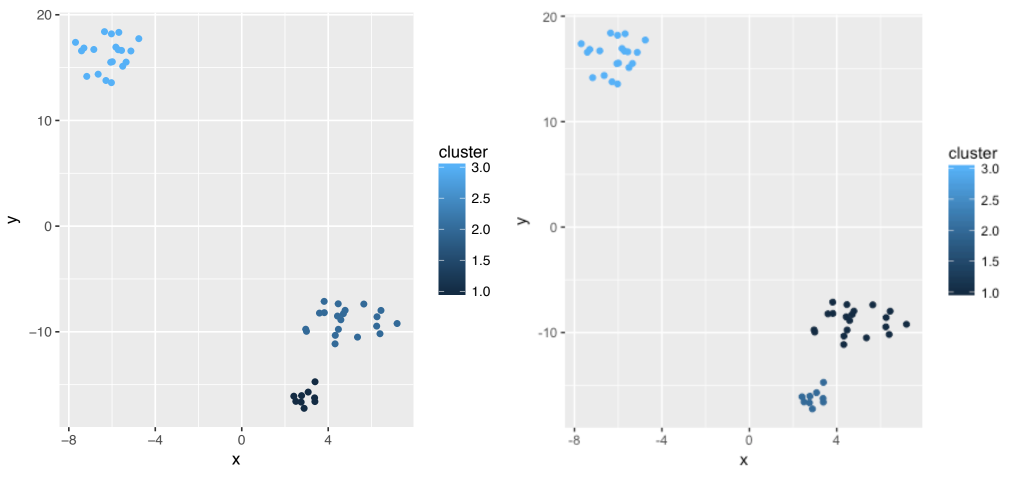
```
> res<-NRselect(Data)
> res$list
```

The best combination produced is saved in `res$resData` and the clustering result is saved in `res$cluster`

```
>res$resData
>res$cluster
```

You can draw the figure of the clustering result.

```
>ires<-res$cluster
>sres<-c(rep(1,9),rep(2,20),rep(3,20))
>
ggplot(data.frame(x=res$resData[,1],y=res$resData[,2],cluster=ires),aes
(x=x,y=y,colour=cluster))+geom_point()
>
ggplot(data.frame(x=res$resData[,1],y=res$resData[,2],cluster=sres),aes
(x=x,y=y,colour=cluster))+geom_point()
```



(a) (b)

Figure 1| Figure 1(a) is the result we get from k-means, figure 1(b) is the standard results. Two results are very similar, means that after Linnorm and tSNE, data becomes very suitable for clustering.

2.NRindex

2.1 Examples

Here is an example of using NRindex.

First, we load the required packages, and the dataset.

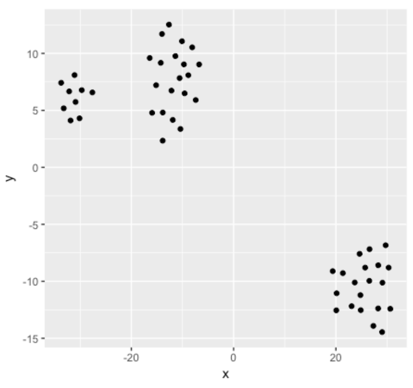
```
>library(Rcpp)
>library(dplyr)
```

```
>library(NRselect)
>data(Biase)
```

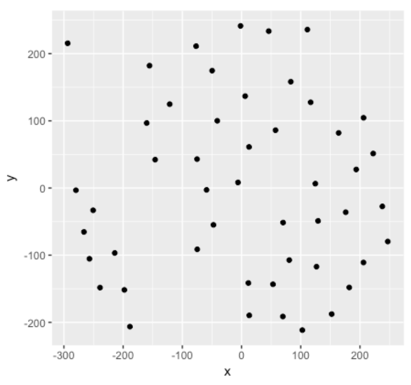
Then we use two different combination of normalization and dimension reduction tools(Linnorm and tSne, TMM and sammon) to pretreat the dataset, and use NRindex to evaluate the quality of the producing data.

```
#library necessary packages for normalization and dimension reduction
tools
>library(Linnorm)
>library(EDASeq)
>library(edgeR)
>library(MASS)
>library(Rtsne)
#pretreat data with Linnorm and tSne
>Data<-Linnorm(as.matrix(Data))
>Data<-Rtsne(t(Data),dim=2,perplexity = 15)
>Data<-Data$Y
#plot Data
>ggplot(data.frame(x=Data[,1],y=Data[,2]),aes(x=x,y=y))+geom_point()
#NRindex
>NRindex(Data,nrow(Data),ncol(Data))
#result is 0.8168423
#pretreat data with TMM and sammon
>sec<-calcNormFactors(Data,method="TMM")
>Data<-log2(t(t(Data)/sec)+1)
>Data<-sammon(d=dist(t(Data),method="euclidean"),k=2)
>Data<-Data$points
#plot Data
>ggplot(data.frame(x=Data[,1],y=Data[,2]),aes(x=x,y=y))+geom_point()
#NRindex
>NRindex(Data,nrow(Data),ncol(Data))
#result is 0.5482054
```

We can plot these two producing data and see the difference.



(a)



(b)

Figure 2| Figure 2(a) is the producing data of Linnorm and tSNE, figure 2(b) is the producing data of Linnorm and tSNE. We can see that data which get high NRindex is 'centrally' than data which get low NRindex.