cdev examination

Initialaze, git clone https://github.com/ttdtrang/cdev-paper.git and set cdev-paper as working directory.

```
devtools::load_all('./')
```

```
## i Loading cdev.paper
```

Create expression matrix, random "normalization coefficients" and "normalized" matrix

```
NSAMPLES = 400 # I used 400 samples to highlight peroformance differences
nGene = 10000
m0 = matrix(rpois(NSAMPLES*nGene,100),ncol=NSAMPLES) # generate random read count matrix
v1 = runif(NSAMPLES,0.1,10) # generate "normalization coefficients" ()
m1 = sweep(m0,2,v1,'/')
```

Create simpler versions of cdef functions. First make a function to calculate cdev from vector of norm coefficients

```
cdev.from.norm.coefs = function(c){max(c)/min(c)}
```

Make a function that recreates normalization coefficients from two matrices: normalized and non-normalized. Note, that this function is only needed to construct cdev1 with same syntax as original cdev, in any real application normalization factors can be used explicitly (see below).

```
norm.coefs.from.matrces = function(d1,d2){apply(d1/d2,2,median)} #actually just d1[1,]/d2[2,] can be us
```

Finaly, make a faster version of cdev, i have to note that it is rather trivial function that simply divide maximal normalization coefficient by minimal one

```
cdev1 = function(d1,d2)cdev.from.norm.coefs(norm.coefs.from.matrces(d1,d2))
```

Lets compare performance of cdev function from the paper and my simple version

```
system.time({cdev.res=cdev(m0,m1)})
```

```
##    user  system elapsed
##  13.603   0.109  13.874
```

My version is much faster

```
system.time({cdev1.res=cdev1(m0,m1)})
```

```
##    user  system elapsed
##   0.194   0.056   0.253
```

Are results the same?

```
cat('cdev.res=',cdev.res,'\ncdev1.res=',cdev1.res)
```

```
## cdev.res= 89.9867
## cdev1.res= 89.9867
```

How to compare two normalization? Lets try RLE normalization for instance

```
library(edgeR)
```

```
## Loading required package: limma
```

```
v2 = calcNormFactors(DGEList(m0),method = 'RLE')$samples$norm.factors
m2 = sweep(m0,2,v2,'/')
```

Compare RLE and "random normalization"

```
system.time({cdev.res1=cdev(m1,m2)})
```

```
##    user  system elapsed
##  13.722   0.052  13.899
```

```
system.time({cdev1.res1=cdev1(m1,m2)})
```

```
##    user  system elapsed
##   0.235   0.020   0.259
```

Results are the same again

```
cat('cdev.res=',cdev.res1,'\ncdev1.res=',cdev1.res1)
```

```
## cdev.res= 90.01982
## cdev1.res= 90.01982
```

And even easier and faster way to compare two scaling normalization

```
system.time({cdev2.res1=cdev.from.norm.coefs(m1/m2)})
```

```
##    user  system elapsed
##   0.032   0.000   0.033
```

```
cdev2.res1
```

```
## [1] 90.01982
```