JMnorm test run

Guanjue Xiang

2023-03-31

Description: This R Markdown file demonstrates the use of JMnorm to normalize the target signal matrix "TCD8.raw_sigmat.txt" against the reference signal matrix "ref.raw_sigmat.txt". The input matrices, "TCD8.raw_sigmat.txt" and "ref.raw_sigmat.txt", should be formatted as N-by-(M+1) matrices, where N represents the number of cCREs, and M represents the number of chromatin features. The first column of each matrix contains the cCRE IDs. The signal values for each chromatin feature in the cCREs are expected to be non-negative and in linear scale.

Load Libraries

```
start.time <- Sys.time()
library(readr)
library(pheatmap)
```

Load JMnorm source code & setup working directory.

!Change the file path of "JMnorm.script.R" and working directory path.

```
# Source the JMnorm script
source('/Users/guanjuexiang/Documents/projects/git/JMnorm.beta/bin/JMnorm_core/JMnorm.script.R')

# Set the working directory
setwd('/Users/guanjuexiang/Downloads/test_JMnorm')
```

Set parameters

```
# get feature list
feature_list = c("ATAC", "H3K27ac", "H3K27me3", "H3K36me3", "H3K4me1", "H3K4me3", "H3K9me3")

# added 1 to avoid 0 in log transformation
add_sn = 1
```

Read reference signal matrix

```
# read refe_ave_sigmat
file_ref = as.data.frame(read_table('ref.raw_sigmat.txt', col_names = F))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   X1 = col_double(),
##   X2 = col_double(),
```

```
##   X3 = col_double(),
##   X4 = col_double(),
##   X5 = col_double(),
##   X6 = col_double(),
##   X7 = col_double(),
##   X8 = col_double()
## )
```

```
file_ref_sig0 = file_ref[,-1]
```

Normalize signal to noise ratio (SNR) of reference signal matrix

```
# log2 transformation
file_ref_sig0_log2 = log2(file_ref_sig0+add_sn)
```

Determine number of clusters in reference signal matrix

```
# Determine the K value for JMnorm based on reference signal matrix
K = determineK(file_ref_sig0, sample_size=20000, add_sn=add_sn)
```

```
##  ..cutHeight not given, setting it to 8.87  ===>  99% of the (truncated) height range in dendro.
##  ..done.
## [1] "cutreeDynamic: "
## ref_km_cluster
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 1980  928  902  785  670  653  642  632  570  559  558  554  544  527  507  503
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32
##  500  500  488  479  478  448  446  439  404  396  393  376  360  333  319  314
##   33   34   35   36   37   38   39
##  312  299  281  257  244  217  203
```

Read target signal matrix

```
# read target signal matrix
target_ct_rep1_i = as.data.frame(read_table('TCD8.raw_sigmat.txt', col_names = F))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   X1 = col_double(),
##   X2 = col_double(),
##   X3 = col_double(),
##   X4 = col_double(),
##   X5 = col_double(),
##   X6 = col_double(),
##   X7 = col_double(),
##   X8 = col_double()
## )
```

```
# add noise to avoid many 0s
target_ct_rep1_i_sig0 = target_ct_rep1_i[,-1]+matrix(runif(prod(dim(target_ct_rep1_i)), min = -0.00001,

# avoid negative values
target_ct_rep1_i_sig0 = target_ct_rep1_i_sig0-min(target_ct_rep1_i_sig0)
```

Normalize signal to noise ratio (SNR) of target signal matrix

```
# normalize signal to noise ratio (SNR) across features
target_ct_rep1_i_sig = normalize_snr(target_ct_rep1_i_sig0)

# log2 transformation
target_ct_rep1_i_sig_log2 = log2(target_ct_rep1_i_sig+add_sn)
```

JMnorm normalize target against reference

```
# JMnorm
target_ct_rep1_i_sig_JMnorm = JMnorm(target_ct_rep1_i_sig_log2, file_ref_sig0_log2, K)
```

```
## Warning: Quick-TRANSfer stage steps exceeded maximum (= 53193900)
```

```
## [1]   34 2488 5150
## [1]    24 11069 27497
## [1]    13 53867 26201
## [1]    14 44484 57119
## [1]    15 31408 48893
## [1]    17 10458  7147
## [1]   20 8863  451
## [1]    32 20124 20036
## [1]    39 38506 22375
## [1]     3 50858 63815
## [1]    10 53601 73102
## [1]    36 17458 15795
## [1]    11 32257 25870
## [1]    23 32770 31463
## [1]    12 17355 30622
## [1]     1 17635 11560
## [1]    22 23961 18654
## [1]     9 24590 17449
## [1]    29 13634 20499
## [1]     7 82076 28258
## [1]    38 29409 69581
## [1]    16 45646 45550
## [1]    30 13165  7065
## [1]    21 14222 16815
## [1]     2 12109 14070
## [1]     6 67713 58079
## [1]    35 12537  9057
## [1]    18 37049 38719
## [1]    19 11700 13618
## [1]    33 34297 75193
## [1]    25 15725 23706
## [1]    31 17189 17741
## [1]     5 29018 21150
## [1]    28 48166 48438
## [1]   37 7500 9205
## [1]     4 37364 19905
## [1]    27 27124  5807
## [1]    26  3132 15347
## [1]     8 13351  2876
```

Write JMnorm normalized signal matrix

```
# add colnames
colnames(file_ref_sig0_log2) = feature_list

# correlation between features in reference cell-type
file_ref_sig0_log2_cor = cor(file_ref_sig0_log2)

# add colnames
colnames(target_ct_rep1_i_sig_JMnorm) = feature_list

# convert JMnorm normalized target signal matrix back to linear scale
target_ct_rep1_i_sig_JMnorm_linear = 2^(target_ct_rep1_i_sig_JMnorm)-add_sn

# write output JMnorm normalized target signal matrix
write.table(cbind(target_ct_rep1_i[,1], round(target_ct_rep1_i_sig_JMnorm_linear, 3)), 'TCD8.JMnorm_sig
```

Get cross-feature correlation matrix

```
# correlation between features in target cell-type before JMnorm
target_ct_rep1_i_sig_cor = cor(target_ct_rep1_i_sig_log2)

# correlation between features in target cell-type after JMnorm
target_ct_rep1_i_sig_JMnorm_cor = cor(target_ct_rep1_i_sig_JMnorm)

# calculate the R2 between the correlation matrix of target cell-type and reference cell-type
r2_after = r2(as.numeric(target_ct_rep1_i_sig_JMnorm_cor), as.numeric(file_ref_sig0_log2_cor))

r2_before = r2(as.numeric(target_ct_rep1_i_sig_cor), as.numeric(file_ref_sig0_log2_cor))
```

```
print(paste0('R2 of correlation matrix between target reference before JMnorm: ', r2_before))
```

```
## [1] "R2 of correlation matrix between target reference before JMnorm: 0.883057875832249"
```

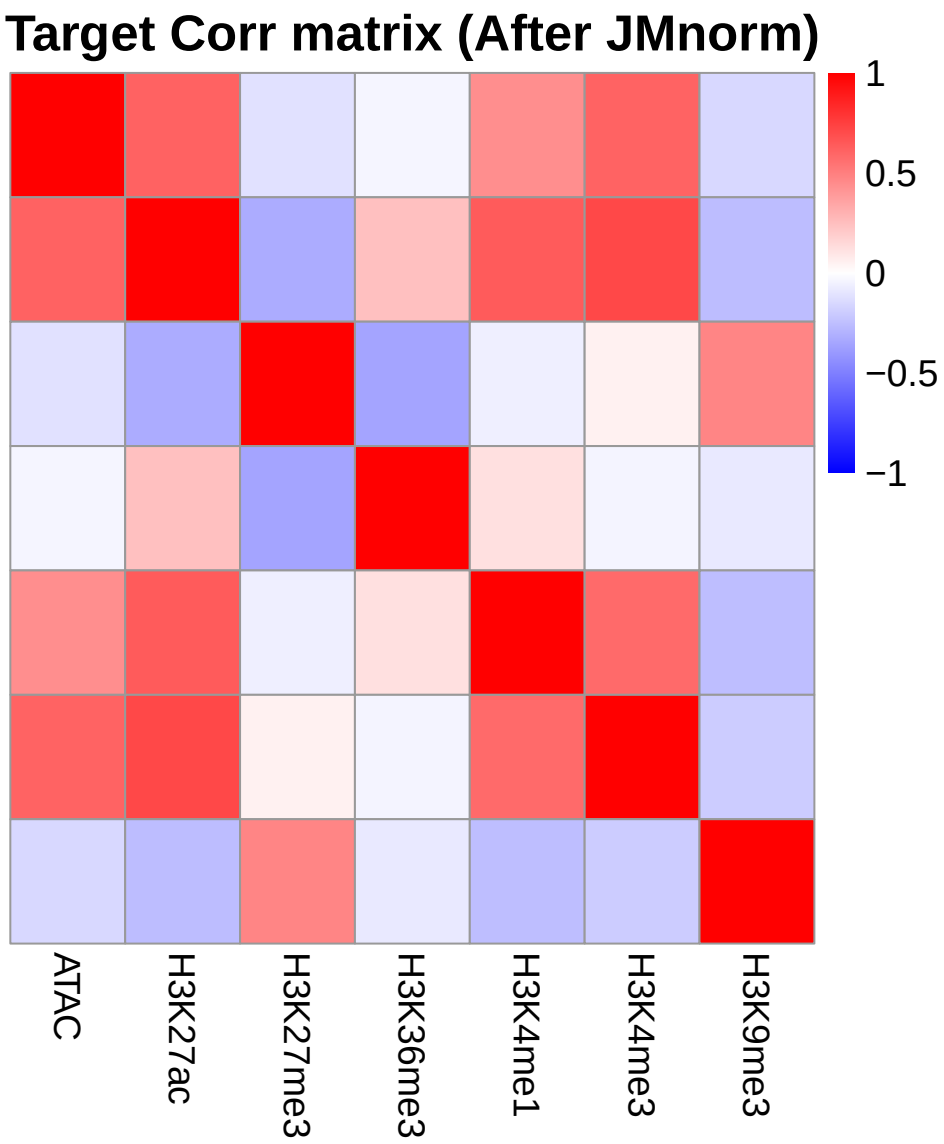
```
print(paste0('R2 of correlation matrix between target reference after JMnorm: ', r2_after))
```

```
## [1] "R2 of correlation matrix between target reference after JMnorm: 0.980565821392993"
```

Plot cross-feature correlation matrix heatmap

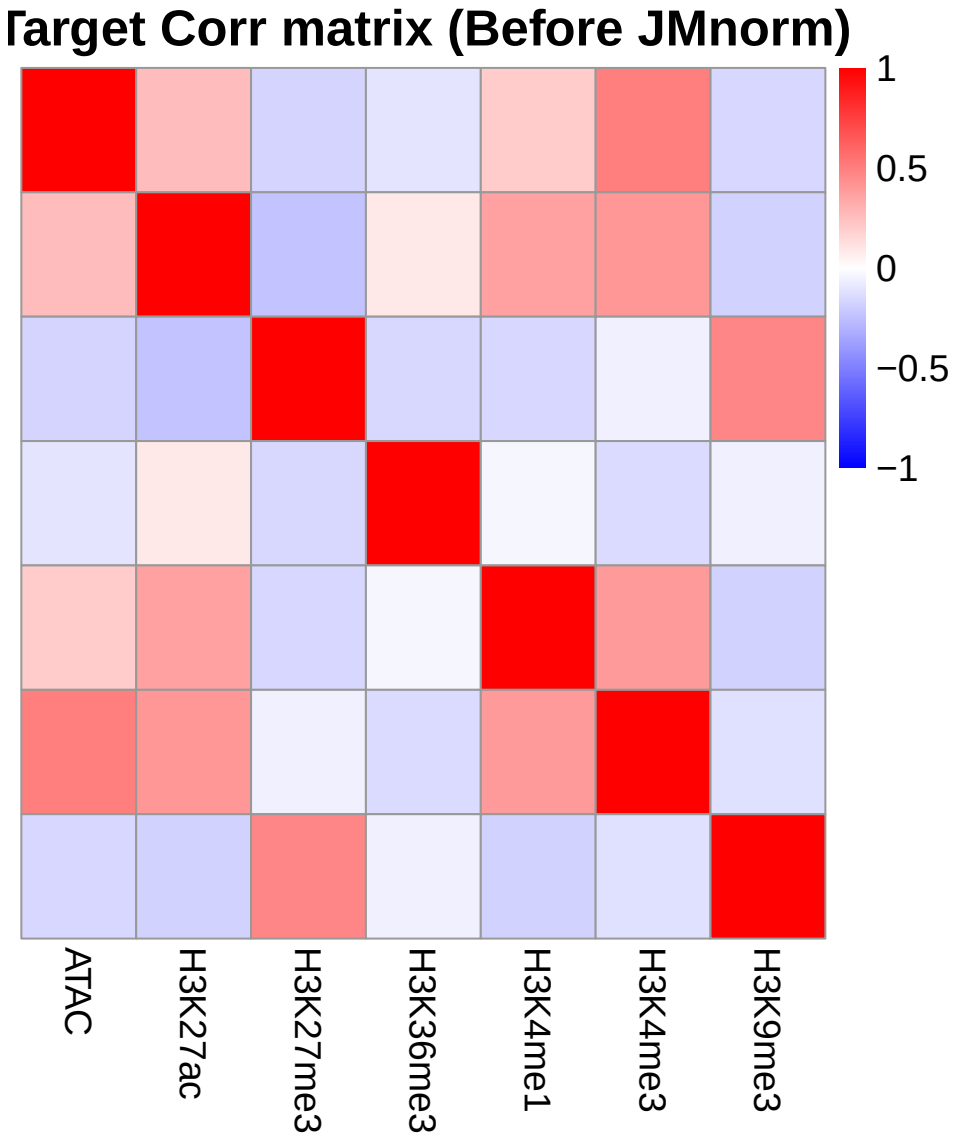
(1) plot the correlation matrix of target cell-type AFTER JMnorm

```
par(mfrow=c(1,2))
# plot the correlation matrix of target cell-type after JMnorm
breaksList = seq(-1,1, by = 0.001)
my_colorbar=colorRampPalette(c('blue', 'white', 'red'))(n = length(breaksList))
pheatmap(target_ct_rep1_i_sig_JMnorm_cor, color=my_colorbar, breaks = breaksList, cluster_rows = F, clus
```



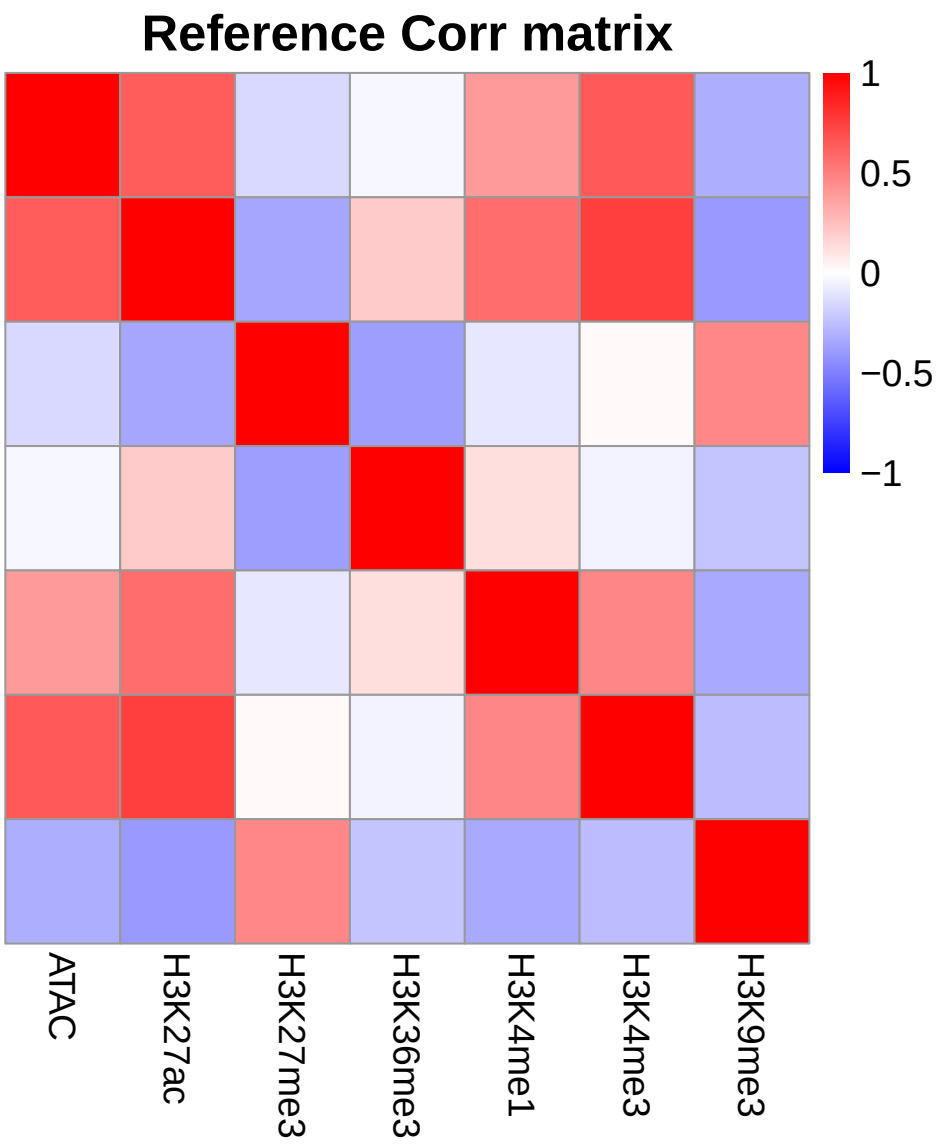
(2) plot the correlation matrix of target cell-type BEFORE JMnorm

```
# plot the correlation matrix of target cell-type before JMnorm
colnames(target_ct_rep1_i_sig_cor) = feature_list
pheatmap(target_ct_rep1_i_sig_cor, color=my_colorbar, breaks = breaksList, cluster_rows = F, cluster_co
```



(3) plot the correlation matrix of reference

```
breaksList = seq(-1,1, by = 0.001)
my_colorbar=colorRampPalette(c('blue', 'white', 'red'))(n = length(breaksList))
# plot the correlation matrix of reference cell-type
pheatmap(file_ref_sig0_log2_cor, color=my_colorbar, breaks = breaksList, cluster_rows = F, cluster_cols
```

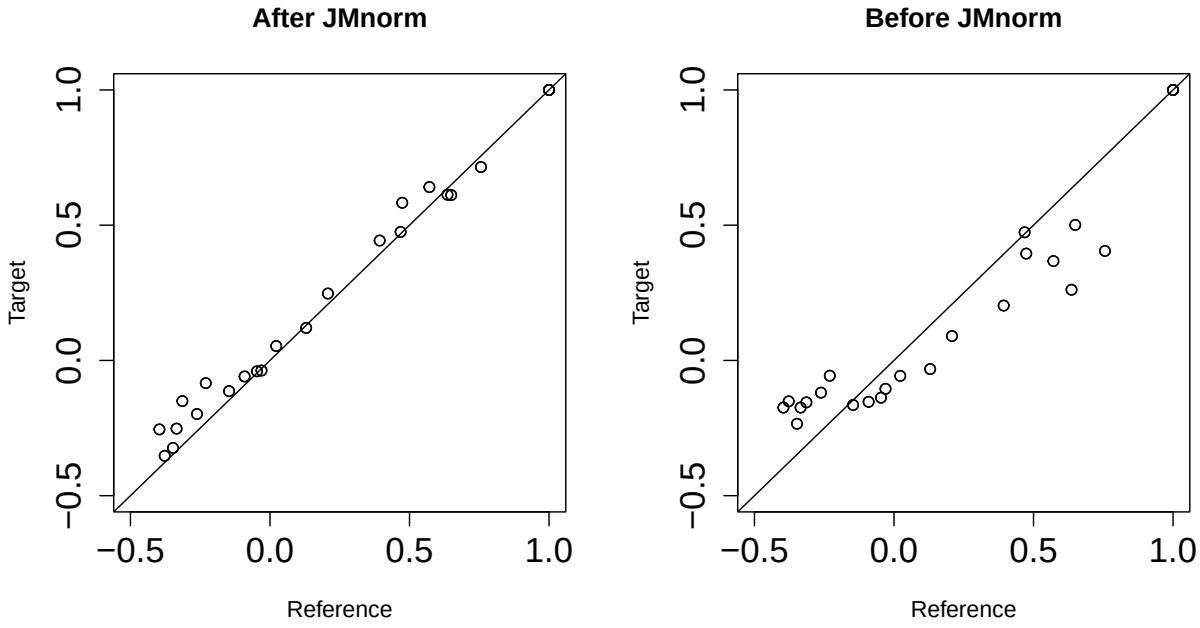


(4) plot scatterplots of correlations between reference and target (Before & After JMnorm)

```
par(mfrow=c(1,2))

# plot the correlation matrix of reference cell-type
plot(as.numeric(file_ref_sig0_log2_cor), as.numeric(target_ct_rep1_i_sig_JMnorm_cor), xlab='Reference',
abline(0,1)

plot(as.numeric(file_ref_sig0_log2_cor), as.numeric(target_ct_rep1_i_sig_cor), xlab='Reference', ylab='
abline(0,1)
```




```
end.time <- Sys.time()
```

```
print(paste0('Running time: ', round((end.time - start.time), 3)))
```

```
## [1] "Running time: 56.224"
```