nacho: a nanoString quality control dashboard

G.A. Bouland, Dr. R.C. Slieker

Department of Cell and Chemical Biology, Leiden University Medical Center, Leiden, The Netherlands

19 juni 2018

Introduction

Nacho(NanoString Quality Control Dashboard) is specially developed for NanoString nCounter data. NanoString nCounter data are gene expression assays where there is no need for the use of enzymes or amplification protocols and work with fluorescent barcodes. Each barcode is assigned a mRNA/miRNA which after bonding with its target can be counted. As a result each count of a specific barcode represents the presence of its target mRNA/mRNA. **nacho** is able to analyse the exported data a NanoString nCounter assay produces and facilitates the user in performing a quality control. **nacho** does this by visualizing QC metrics, expression of control genes, principal components and initial sample specific size factors in an interactive web application. With the use of two convenient function calls RCC files are summarized and visualized, namely; ***summarize()*** and ***visualize()***. **nacho** also includes a function; ***normalize()***, which calculates sample specific size factors. The user has the choice if these calculated size factors ought to be applied to the counts and thereby transforming the data, either when it is desired the raw counts with the size factors separately can also be inquired.

To display the usage and utility of **nacho**, we show four examples in which the above mentioned functions are used and the result is briefly examined. **nacho** comes with presummarized data and in the first example we use this data to call upon the interactive web application with the use of ***visualize()***. In this example we also show how **nacho** can be used to identify batch effects. In the second example we show the process of going from raw RCC files to visualisations with a data set queried from GEO using GEOquery. In the third example we use the summarized data from second example to calculate the sample specific size factors using ***normalize()*** and its added functionality to predict housekeeping genes.

Besides creating interactive visualisations, **nacho** also identifies poorly performing samples which can be seen under the Outlier Table tab in the interactive web application. While calling ***normalize()*** the user has the possibility to remove these outliers before size factor calculation.

Example using presummarized nanoString nCounter data

This example shows how to use summarized data to call upon the interactive web application. The raw data used in the summarization is from a study of Bruce JP et al[1]. and was acquired from the NCBI GEO public database[2].

```
library(nacho)
data(exampleData)
visualize(exampleData)
```

When ***visualize()*** is called, the web application that opens should look something like Figure 1. In this figure the tabs QC Visualuals and Average Count vs Binding Denisty are chosen because this shows a clear discrepancy between the CartridgeIDs.

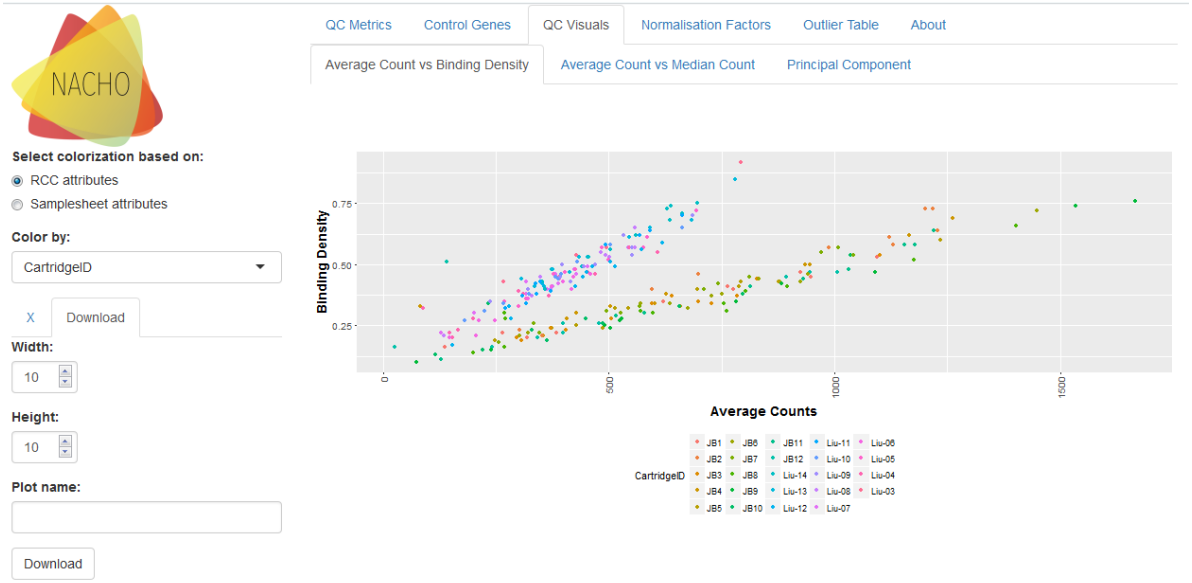

Figure1: Screenshot of NACHO: On the left there are widgets that control the interactive plotting area. These widgets differ dependent on which main tab is chosen. The second layer of tabs is also interactive and also changes based on which main tab is chosen. Each sample in the plots can be coloured based on either technical specifications which are included in the RCC files or on specifications of your own choosing, though these specifications need to be included in the samplesheet.

Example using nanoString nCounter data from GEO

Numerous NanoString nCounter data sets are available from GEO. In this example we use a mRNA dataset from the study of Liu M.C et al.[1] with the GEO accession number: **GSE74821**. The data is extracted and prepared using the following code.

```
library(GEOquery)
gse <- getGEO("GSE74821")
targets <- pData(phenoData(gse[[1]]))
getGEOSuppFiles("GSE74821")
untar("GSE74821/GSE74821_RAW.tar", exdir = "example/Data")
IDs <- list.files("example/Data")
targets$IDFILE <- IDs
write.csv(targets,file="example/Samplesheet.csv")
```

After we extracted the dataset to the ***/Data*** directory, the ***Samplesheet.csv*** is written containing a column with the exact names of the files for each sample. Subsequently ***summarize()*** is called. The first argument requires the path to the directory containing the RCC files, the second argument is the location of samplesheet followed by third argument with the column name containing the exact names of the files. The following two arguments consecutively indicate which housekeeping genes and normalization method ought to be used. When the summarization is done, the summarized data can be visualized using the ***visualize()*** function as can be seen in the following chunk of code.

```
library(nacho)
summary <- summarize(data_dir ="example/Data", ssheet = "example/Samplesheet.csv",
                     id_colname = "IDFILE", housekeep = "predict",norm = "GLM")
visualize(summary)
```

Normalize example using housekeeping discovery

Nacho is equipped with an algorithm which can discover housekeeping genes within your own dataset. Nacho finds the five best suitable housekeeping genes, however, it is possible that one of these five genes might not be suitable, which is why a subset of these discovered housekeeping genes might work better in some cases. For this example we use the **GSE74821** dataset from the previous example.

```
library(nacho)
#Note the argument that is passed on to the housekeep variable.
my_summary <- summarize(data_dir ="example/Data", ssheet = "example/Samplesheet.csv",
                     id_colname = "IDFILE", housekeep = "predict",norm = "GLM")
```

The discovered housekeeping genes are saved in a global variable named **predicted_housekeeping**. This gives the user absolute control over which housekeeping genes are used. For instance, the suitablity of the discovered housekeeping genes can manually be checked.

```
print(predicted_housekeeping)
```

```
## [1] "UBE2T"  "ACTR3B" "BAG1"   "MDM2"   "BLVRA"
```

Let's say *BAG1* and *MDM2* are not suitable,therefore, you want to exclude these genes from the normalization process.

```
my_housekeeping <- predicted_housekeeping[-c(3,4)]
print(my_housekeeping)
```

```
## [1] "UBE2T"  "ACTR3B" "BLVRA"
```

The next step is the actual normalization. The first argument requires the summary which is created with the ***summarize()*** function. The second arugument requires a vector of gene names. In this case it is a subset of the discovered housekeeping genes we just made. With the third argument the user has the choice to remove the outliers. Lastly the normalization method can be choosed. Here the user has a choice between **GLM** or **GEO**. The differences between normalization methods are nuanced, however, a preference for either method are use case specific. In this example **GLM** is used.

```
my_data <- normalize(summary = my_summary, housekeep = my_housekeeping, remove.outliers = T,
                     norm = "GLM")
```

The ***normalize()*** function has three dataframes as output; 1) **scaling**, 2) **normalized** and 3) **raw**. These dataframes can be accessed and used using.

```
my_data$scaling #Table 1
my_data$counts #Table 2
my_data$normalized #Table 3
```

In *Table 1* the calculated normalization values can be seen. Here the first five sample are rendered. The positive scaling factors are caluculated using the **GLM** method as indicated when the ***normalize()*** function was called. The same is true for the background signal. The housekeeping factor was calulated using a subset of the discovered housekeeping genes.

Table 1: Normalization values

	Positive Factor	Background Signal	Housekeeping Factor
GSM1934697	0.732	16.286	1.009
GSM1934698	0.770	13.305	1.498
GSM1934699	0.814	11.328	1.302
GSM1934700	0.730	13.237	1.750
GSM1934701	0.788	18.344	0.765

In *Table 2* the untransformed counts of the first five samples and first five genes are rendered. Some gene analysis packages require raw counts together with scaling factors.

Table 2: Raw counts

	GSM1934697	GSM1934698	GSM1934699	GSM1934700	GSM1934701
FOXA1	2,845	101	2,455	2,859	5,128
EXO1	388	1,020	422	271	658
CDH3	115	2,088	354	163	946
BIRC5	682	2,516	1,386	345	352
MKI67	775	2,073	1,319	321	712

While other packages or analysis methods require already transformed data, which can be seen in *Table 3*. In this Table, the sample specific scaling factors are applied on the counts and thereby transformed.

Table 3: Transformed counts

	GSM1934697	GSM1934698	GSM1934699	GSM1934700	GSM1934701
FOXA1	2,090	101	2,590	3,636	3,080
EXO1	275	1,161	435	329	386
CDH3	73	2,392	363	191	559
BIRC5	492	2,885	1,457	424	201
MKI67	561	2,375	1,386	393	418

Session info

```
sessionInfo()
```

```
## R version 3.4.4 (2018-03-15)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 7 x64 (build 7601) Service Pack 1
##
## Matrix products: default
##
## locale:
```

```
## [1] LC_COLLATE=Dutch_Netherlands.1252  LC_CTYPE=Dutch_Netherlands.1252
## [3] LC_MONETARY=Dutch_Netherlands.1252 LC_NUMERIC=C
## [5] LC_TIME=Dutch_Netherlands.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] reshape2_1.4.3 nacho_0.1.0    png_0.1-7      knitr_1.20
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.16     magrittr_1.5     munsell_0.4.3    xtable_1.8-2
##  [5] colorspace_1.3-2 R6_2.2.2         rlang_0.2.0      vipor_0.4.5
##  [9] stringr_1.3.0    plyr_1.8.4       tools_3.4.4      DT_0.4
## [13] grid_3.4.4       beeswarm_0.2.3   ggbeeswarm_0.6.0 gtable_0.2.0
## [17] stargazer_5.2.2  gtools_3.5.0     htmltools_0.3.6  yaml_2.1.18
## [21] lazyeval_0.2.1   rprojroot_1.3-2  digest_0.6.15    tibble_1.4.2
## [25] shiny_1.0.5      ggplot2_2.2.1    htmlwidgets_1.0  mime_0.5
## [29] evaluate_0.10.1  rmarkdown_1.9    stringi_1.1.7    pillar_1.2.1
## [33] compiler_3.4.4   scales_0.5.0     backports_1.1.2  httpuv_1.3.6.2
```

References

[1] Bruce JP, Hui AB, Shi W, Perez-Ordonez B et al. Identification of a microRNA signature associated with risk of distant metastasis in nasopharyngeal carcinoma. Oncotarget 2015 Feb 28;6(6):4537-50. PMID: 25738365

[2]Barrett T, Wilhite SE, Ledoux P, Evangelista C, Kim IF, Tomashevsky M, Marshall KA, Phillippy KH, Sherman PM, Holko M, Yefanov A, Lee H, Zhang N, Robertson CL, Serova N, Davis S, Soboleva A. NCBI GEO: archive for functional genomics data sets–update. Nucleic Acids Res. 2013 Jan;41(Database issue):D991-5.

[3] Liu MC, Pitcher BN, Mardis ER, Davies SR, Friedman PN, Snider JE, Vickery TL, Reed JP, DeSchryver K, Singh B, Gradishar WJ, Perez EA, Martino S, Citron ML, Norton L, Winer EP, Hudis CA, Carey LA, Bernard PS, Nielsen TO, Perou CM, Ellis MJ, Barry WT. PAM50 gene signatures and breast cancer prognosis with adjuvant anthracycline- and taxane-based chemotherapy: correlative analysis of C9741 (Alliance). Npj Breast Cancer 2:15023 (2016) http://dx.doi.org/10.1038/npjbcancer.2015.23