

# Package: nanostringr (via r-universe)

August 20, 2024

**Type** Package

**Title** Performs Quality Control, Data Normalization, and Batch Effect Correction for 'NanoString nCounter' Data

**Version** 0.4.2

**Description** Provides quality control (QC), normalization, and batch effect correction operations for 'NanoString nCounter' data, Talhouk et al. (2016) <[doi:10.1371/journal.pone.0153844](https://doi.org/10.1371/journal.pone.0153844)>. Various metrics are used to determine which samples passed or failed QC. Gene expression should first be normalized to housekeeping genes, before a reference-based approach is used to adjust for batch effects. Raw NanoString data can be imported in the form of Reporter Code Count (RCC) files.

**License** MIT + file LICENSE

**URL** <https://github.com/TalhoukLab/nanostringr/>,  
<https://talhouklab.github.io/nanostringr/>

**BugReports** <https://github.com/TalhoukLab/nanostringr/issues>

**Depends** R (>= 3.5.0)

**Imports** assertthat, ccaPP, dplyr, purrr, rlang, tidyr

**Suggests** covr, knitr, magrittr, rmarkdown, testthat

**VignetteBuilder** knitr

**Encoding** UTF-8

**LazyData** TRUE

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.1

**Repository** <https://talhouklab.r-universe.dev>

**RemoteUrl** <https://github.com/talhouklab/nanostringr>

**RemoteRef** HEAD

**RemoteSha** 567954290882000ac056c5579a51bf6900daacb4

## Contents

CCplot . . . . .	2
cohort . . . . .	4
expQC . . . . .	5
HKnorm . . . . .	5
NanoStringQC . . . . .	6
normalize_pools . . . . .	7
normalize_random . . . . .	8
rcc . . . . .	9
refMethod . . . . .	10
<b>Index</b>	<b>11</b>

---

CCplot	<i>Concordance Correlation Plot</i>
--------	-------------------------------------

---

### Description

Plotting function for reliability measure.

### Usage

```
CCplot(
  method1,
  method2,
  Ptype = "None",
  metrics = FALSE,
  xlabel = "",
  ylabel = "",
  title = "",
  subtitle = NULL,
  xrange = NULL,
  yrange = NULL,
  MArange = c(-3.5, 5.5)
)
```

### Arguments

method1	measurements obtained in batch 1 or using method 1
method2	measurements obtained in batch 2 or using method 2
Ptype	type of plot to be outputted c("scatter", "MAplot")
metrics	if TRUE, prints Rc, Ca, and R2 to console
xlabel	x-axis label for scatterplot
ylabel	y-axis label for scatterplot
title	title for the main plot

subtitle	subtitle of plot
xrange	range of x axis
yrange	range of y axis
MArange	MA range

**Value**

Either a scatterplot or MA plot showing concordance correlation.

**Author(s)**

Aline Talhouk

**Examples**

```
# Simulate normally distributed data
set.seed(12)
a1 <- rnorm(20) + 2
a2 <- a1 + rnorm(20, 0, 0.15)
a3 <- a1 + rnorm(20, 0, 0.15) + 1.4
a4 <- 1.5 * a1 + rnorm(20, 0, 0.15)
a5 <- 1.3 * a1 + rnorm(20, 0, 0.15) + 1
a6 <- a1 + rnorm(20, 0, 0.8)

# One scatterplot
CCplot(a1, a2, Ptype = "scatter")

m2 <- list(a1, a2, a3, a4, a5, a6)
mains <- c("Perfect Agreement", "Very Good Agreement", "Location Shift",
          "Scale Shift", "Location and Scale Shift", "Measurement Error")
subs <- letters[1:6]
par(mfrow = c(3, 2), mar = c(5.1, 4.1, 1.5, 1.5))

# Scatterplots
mapply(function(y, t, s)
  CCplot(method1 = a1, method2 = y, Ptype = "scatter",
         xlabel = "X", ylabel = "Y", title = t, subtitle = s),
  y = m2, t = mains, s = subs)

# MAplots and show metrics
mapply(function(y, t, s)
  CCplot(method1 = a1, method2 = y, Ptype = "MAplot",
         title = t, subtitle = s, metrics = TRUE),
  y = m2, t = mains, s = subs)
```

---

cohort

*NanoString Experiment Cohorts*

---

### Description

There were five different cohorts used in NanoString experiments.

### Usage

`hld.r`

`ovd.r`

`ovc.r`

`hlo.r`

`ovo.r`

### Format

- `hld.r` Hodgkin Lymphoma Clinical Samples: a data frame with 232 rows and 77 columns
- `ovd.r` Ovarian Cancer Clinical Samples: a data frame with 133 rows and 261 columns
- `ovc.r` Ovarian Cancer Cell Lines: a data frame with 133 rows and 29 columns
- `hlo.r` DNA Oligonucleotides for the HL CodeSet: a data frame with 40 rows and 71 columns
- `ovo.r` DNA Oligonucleotides for the OC CodeSet: a data frame with 133 rows and 138 columns

An object of class `data.frame` with 232 rows and 77 columns.

An object of class `data.frame` with 133 rows and 261 columns.

An object of class `data.frame` with 133 rows and 29 columns.

An object of class `data.frame` with 40 rows and 71 columns.

An object of class `data.frame` with 133 rows and 138 columns.

### Details

Each data object contains raw expression counts, so no normalization has been applied. The format is a data frame with genes as rows, samples as columns. Note that the first three columns contain gene metadata and are always labelled "Code.Class", "Name", and "Accession", and the rest are sample names. Hence, for the `hld.r` data, the raw counts are contained in 232 genes for  $77 - 3 = 74$  samples. The total number of samples is  $74 + 258 + 26 + 68 + 135 = 561$ , which matches the number of rows in `expQC`, the expression QC data.

### Source

See Table 1 of <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0153844> for details.

**See Also**[expQC](#)

---

`expQC`*Expression QC data*

---

**Description**

Quality control metrics for the five cohorts analyzed in NanoString experiments.

**Format**

A data frame with 561 rows and 23 columns.

**Details**

The total number of samples from the five cohorts is 561.

**See Also**[cohort](#)

---

`HKnorm`*Normalization to Housekeeping Genes*

---

**Description**

Normalizes the gene expression of NanoString nCounter data to housekeeping genes. This is done by subtracting the average log housekeeping gene expression from the expression level of every gene in each sample.

**Usage**

```
HKnorm(raw, is.logged = FALSE, corr = 1e-04)
```

**Arguments**

<code>raw</code>	data frame of raw counts obtained from nCounter (rows represent genes, columns represent samples). The first three columns must be labeled: <code>c("Code.Class", "Name", "Accession")</code> and contain that information.
<code>is.logged</code>	logical; If TRUE, normalization has already been done on log base 2 scale, no need log the data
<code>corr</code>	small correction to avoid error

**Value**

data frame of log normalized data in the same format but without reference genes

**Author(s)**

Aline Talhouk, Derek Chiu

**Examples**

```
HKnorm(ovd.r)
HKnorm(ovd.r, is.logged = TRUE)
```

---

NanoStringQC

*QC metrics for NanoString Data*

---

**Description**

Computes and returns NanoString quality control metrics and flags.

**Usage**

```
NanoStringQC(raw, exp, detect = 80, sn = 150)
```

**Arguments**

raw	data frame of raw counts obtained from nCounter (rows represent genes, columns represent samples). The first three columns must be labeled: c("Code.Class", "Name", "Accession") and contain that information.
exp	data frame of annotations with rows in the same order as the columns of raw. Requires a column labeled "File.Name" with entries corresponding to sample names in raw, also needs columns c("fov.counted", "fov.count", "binding.density"). These fields can be extracted from the nanostring RCC files.
detect	threshold of percentage of genes expressed over limit of detection (LOD) that we would like to detect (not decimal), defaults to 80 percent.
sn	signal to noise ratio of the housekeeping genes we are willing to tolerate, defaults to 150.

**Value**

data frame of annotations updated with normalization parameters

**Author(s)**

Aline Talhouk, Derek Chiu

**Examples**

```
exp.OVD <- subset(expQC, OVD == "Yes")
expOVD <- NanoStringQC(ovd.r, exp.OVD)
```

---

normalize_pools	<i>Normalize data using common pools</i>
-----------------	--

---

## Description

Normalize nanostring gene expression using common pools between two CodeSets.

## Usage

```
normalize_pools(x, ref, x_pools, ref_pools, p = 3, weigh = TRUE)
```

## Arguments

x	target data
ref	reference data
x_pools	target pool samples
ref_pools	reference pool samples
p	number of pool sample sets. Defaults to 3.
weigh	logical; if TRUE, the average expression in x_pools is reweighed by the distribution of the p pool sample sets in ref_pools.

## Details

The target and reference expression samples, as well the target and reference pool samples all need to be specified. We recommend reweighing the target pool samples when calculating the average expression by the distribution of reference pools.

## Value

normalized gene expression

## Author(s)

Derek Chiu

---

normalize_random	<i>Normalize data using random reference samples</i>
------------------	--

---

**Description**

Normalize nanostring gene expression using randomly chosen samples for the reference-based approach for batch adjustment.

**Usage**

```
normalize_random(x, ref, n = 1, strata = NULL, seed = NULL)
```

**Arguments**

x	target data
ref	reference data
n	number of random reference samples to select for normalization
strata	a grouping variable for stratified random sampling. If strata has k levels, then n * k random samples are selected.
seed	random seed for reproducibility

**Details**

The number of randomly chosen numbers can be selected, and optionally a strata can be specified such that n reference samples are selected from each level (like a stratified bootstrap). In relation to the reference method, the random samples removed from ref form R1, the random samples removed from x form R2, and the remaining samples from x form Y. See refMethod() for details.

In subsequent analyses, we refer to a method using normalize\_random(n) as the "Random n" method.

**Value**

normalized gene expression

**Author(s)**

Derek Chiu



---

**rcc** *Read NanoString RCC files*

---

**Description**

Read RCC files and extract count and attribute data. Use `read_rcc()` for multiple files, and use the `parse_*()` functions for single files.

**Usage**

```
read_rcc(path = ".")  
  
parse_counts(file)  
  
parse_attributes(file)
```

**Arguments**

<code>path</code>	directory path for multiple RCC files
<code>file</code>	RCC file name

**Details**

RCC files for a sample are direct outputs from NanoString runs. We can extract counts for each gene in a sample. Sample attributes include sample ID, GeneRLF, date, cartridge ID, lane number, Fov count, Fov counted, and binding density. `read_rcc()` merges both count and attribute data across samples.

If `path` points to a zipped RCC file with multiple samples, the zip file is uncompressed and a directory of RCC sample files is created with the same name. Only file extensions ".RCC" or ".rcc" are allowed.

**Value**

`read_rcc()` reads in a directory of RCC files and outputs a list with two elements:

- `raw`: A tibble of parsed counts for multiple RCC files created by calling `parse_counts()` on each sample. Columns include "Code.Class", "Name", "Accession", and a column for each sample ID. There is one row per gene.
- `exp`: A tibble of parsed attributes for multiple RCC files created by calling `parse_attributes()` on each sample. Columns include "File.Name" (sample ID), "geneRLF", "nanosting.date", "cartridgeID", "lane.number", "fov.count", "fov.counted", "binding.density". There is one row per sample.

`parse_counts()` reads a single RCC file and returns a tibble of parsed counts.

`parse_attributes()` reads a single RCC file and returns a list of parsed attributes.

**Author(s)**

Derek Chiu

**Examples**

```
rcc_file <- system.file("extdata", "example.RCC", package = "nanostringr")
parse_counts(rcc_file)
parse_attributes(rcc_file)
```

---

refMethod

*Reference-based approach for batch adjustment*

---

**Description**

Batch adjustment by considering a measure relative to a reference sample

**Usage**

```
refMethod(Y, R1, R2)
```

**Arguments**

Y	data run in first or second batch, samples are rows and genes are columns. If correcting one batch only R1 is needed and would correspond to reference run in the same batch as Y, if calibrating one batch to the other Y represents the data from batch 2 and R1 would be reference run in batch 1 and R2 would be reference from batch 2
R1	reference data run in the first batch
R2	reference data run in the second batch

**Value**

The Y data adjusted calibrated to batch 1 (if two batches are presented) or the data with reference sample expression removed if only one data is provided

**Author(s)**

Aline Talhouk

**Examples**

```
set.seed(12)
A <- matrix(rnorm(120), ncol = 10)
B <- matrix(rnorm(80), ncol = 10)
C <- matrix(rnorm(50), ncol = 10)
refMethod(A, B, C)
```

# Index

## \* datasets

cohort, 4

CCplot, 2

cohort, 4, 5

expQC, 4, 5, 5

HKnorm, 5

hld.r (cohort), 4

hlo.r (cohort), 4

NanoStringQC, 6

normalize\_pools, 7

normalize\_random, 8

ovc.r (cohort), 4

ovd.r (cohort), 4

ovo.r (cohort), 4

parse\_attributes (rcc), 9

parse\_counts (rcc), 9

rcc, 9

read\_rcc (rcc), 9

refMethod, 10