highSCREEN: High-Throughput Screening for Plate Based Assays

Ivo D. Shterev *   Cliburn Chan   Gregory D. Sempowski

February 20, 2018

1 Introduction

This vignette describes the use of the R extension package `highSCREEN` for high throughput screening of small molecule compounds with activities measured on multi-well plates. The plate-based assay raw results can be any continuous value - for example, optical density (OD; in nanometers). Package functionalities include small molecule compound library screening data extraction and normalization, plate quality control (QC), identifying compounds that are hits according to defined criteria and visualization of compounds and controls. The framework supports `96-well` and `384-well` plate formats[1]. The package is also capable of handling any number of replicates of the data. Currently, `highSCREEN` implements three different within plate quality control (QC) procedures which determine plate pass or fail. The package implements three different normalization methods, namely the `b-score`, the `c-score` (also known as percent degranulation) and the `z-score` normalization methods [1]. The user can also plot the density and histogram of controls which can be helpful in tuning the QC procedures.

2 Data Format

2.1 Plate Layout

The following plate formats are supported:

- 96-well plate. This format represents an `8×12` matrix in which the first and last columns represent control wells and columns two to ten represent compound wells.

- 384-well plate. This format represents an `16×24` matrix. The first and last two columns represent control wells and columns three to twenty two represent compound wells.

- 384-well plate composed of four 96-well plates. This format represents an `16×24` matrix. The first and last two columns represent control wells and columns three to twenty two represent compound wells. From this 384-well plate, four 96-well plates can be constructed as shown in Fig. 1.

---

*Correspondence: i.shterev@duke.edu
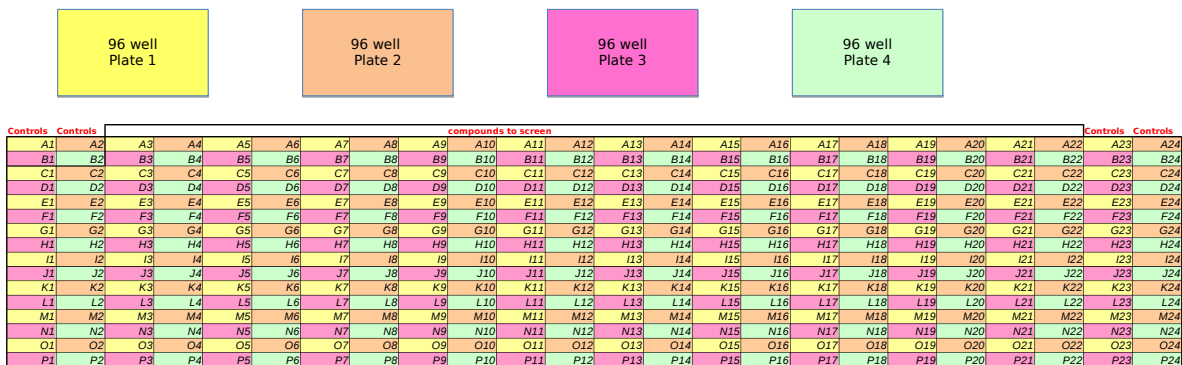[1]See section 2 for additional details on supported plate layouts.

Figure 1: 384-well plate consisting of four 96-well plates.

Controls	Controls	compounds to screen																				Controls	Controls
A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19	A20	A21	A22	A23	A24
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15	B16	B17	B18	B19	B20	B21	B22	B23	B24
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24
D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20	D21	D22	D23	D24
E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12	E13	E14	E15	E16	E17	E18	E19	E20	E21	E22	E23	E24
F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24
G1	G2	G3	G4	G5	G6	G7	G8	G9	G10	G11	G12	G13	G14	G15	G16	G17	G18	G19	G20	G21	G22	G23	G24
H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12	H13	H14	H15	H16	H17	H18	H19	H20	H21	H22	H23	H24
I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12	I13	I14	I15	I16	I17	I18	I19	I20	I21	I22	I23	I24
J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	J11	J12	J13	J14	J15	J16	J17	J18	J19	J20	J21	J22	J23	J24
K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12	K13	K14	K15	K16	K17	K18	K19	K20	K21	K22	K23	K24
L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12	L13	L14	L15	L16	L17	L18	L19	L20	L21	L22	L23	L24
M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16	M17	M18	M19	M20	M21	M22	M23	M24
N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	N12	N13	N14	N15	N16	N17	N18	N19	N20	N21	N22	N23	N24
O1	O2	O3	O4	O5	O6	O7	O8	O9	O10	O11	O12	O13	O14	O15	O16	O17	O18	O19	O20	O21	O22	O23	O24
P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24

2.2 Distribution of Controls

As mentioned in the previous subsection, it is assumed that the control wells are located either in the first and last plate columns (96-well plate), or in the first and last two plate columns (348-well plate) as shown in Fig. 1. The R package `highSCREEN` assumes that the plate contains positive and negative control wells. The package can also handle additional control types as specified by the user. The requirement is that in addition to the 384-well plate data the user also provides a control map. The map specifies the control type and its position in the control columns, and is used to identify the controls in the plate layout. There are two types of control maps, 96-well and 384-well plate control maps. The first column of the 384-well plate control map specifies the controls and their position in the first column of the 384-well plate. Similarly, the second, third and fourth control map columns correspond to the second, twenty third and twenty fourth 384-well plate columns, respectively. Similarly, the first and second columns of the 96-well plate control map specify the controls and their position in the first and second columns of the 96-well plate, respectively. In the example below, a 384-well plate contains five different types of controls, positive controls ("Control P"), negative controls ("Control N"), controls with low concentration ("Control low"), controls with medium concentration ("Control med"), and controls with high concentration ("Control high").

```
set.seed(1234)
library(highSCREEN)


## Loading required package:  gplots
##
## Attaching package:  'gplots'
## The following object is masked from 'package:stats':
##
##     lowess

nc = 24
nr = 16
# create a 384-well plate with compounds and controls
replicate = matrix(abs(rnorm(nr*nc)), nr, nc)
head(replicate)


##           [,1]      [,2]      [,3]      [,4]        [,5]      [,6]
## [1,] 1.2070657 0.5110095 0.7094400 0.5238281 0.007604756 0.1777900
## [2,] 0.2774292 0.9111954 0.5012581 0.4968500 1.777084448 0.1699941
## [3,] 1.0844412 0.8371717 1.6290935 1.8060313 1.138607737 1.3723019
## [4,] 2.3456977 2.4158352 1.1676193 0.5820759 1.367827179 0.1737872
## [5,] 0.4291247 0.1340882 2.1800396 1.1088896 1.329564791 0.8502323
## [6,] 0.5060559 0.4906859 1.3409932 1.0149620 0.336472797 0.6976087
```

```
##           [,7]       [,8]      [,9]      [,10]      [,11]      [,12]
## [1,] 1.1346080 1.10976723 0.6360998 0.5137628 0.8473501 1.12376279
## [2,] 0.8782036 0.84927420 0.2263015 0.3992718 0.2606394 3.04376589
## [3,] 0.9729168 0.02236253 1.0136903 1.6628564 0.4144197 0.23502131
## [4,] 2.1211171 0.83114062 0.2527501 0.2758934 0.1830508 0.03325861
## [5,] 0.4145235 1.24428785 1.1719483 0.5062726 0.4070561 2.73221952
## [6,] 0.4747185 0.16902641 0.6687143 0.3475520 0.6246331 0.09979059
##          [,13]      [,14]      [,15]      [,16]      [,17]      [,18]
## [1,] 0.7896469 0.03266396 0.2877097 0.05913517 1.4769696 0.6705594
## [2,] 0.4878146 1.11444896 0.6597701 0.41339889 1.2239038 0.9486326
## [3,] 2.1680325 0.41805782 2.9191401 1.09777217 0.2580684 2.0494030
## [4,] 0.5006946 0.40023524 0.6774155 0.71117526 0.4050028 0.6511136
## [5,] 0.6202102 1.49349310 0.6843203 0.71888873 0.9758033 0.8086193
## [6,] 0.9659032 1.60708094 0.1864921 0.25165107 0.3488767 0.9865806
##          [,19]      [,20]      [,21]      [,22]      [,23]      [,24]
## [1,] 1.5528590 0.02362661 0.14313216 0.27007961 0.7837751 0.04631853
## [2,] 0.1284340 0.64902822 0.02418865 1.61978988 0.2260540 2.25184180
## [3,] 0.9854434 0.50437422 0.50445152 0.21413117 1.5871030 0.60803373
## [4,] 0.1832475 1.61439150 1.58139681 0.81778246 0.5475242 1.50928817
## [5,] 1.7662292 0.44695981 0.03006642 0.05402292 1.8912270 0.23263177
## [6,] 0.6205337 0.76317676 0.71657670 0.33014161 0.8780771 0.03964870
```

```r
# create 384-well plate control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control P", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N",
floor(nr/3))), X3=c(rep("Control N", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P", floor(nr/3))), X4=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P",
floor(nr/3))))
cmap
```

```
##                X1           X2           X3           X4
## 1      Control P    Control P    Control N    Control N
## 2      Control P    Control P    Control N    Control N
## 3      Control P    Control P    Control N    Control N
## 4      Control P    Control P    Control N    Control N
## 5      Control P    Control P    Control N    Control N
## 6    Control low  Control low  Control low  Control low
## 7    Control med  Control med  Control med  Control med
## 8   Control high Control high Control high Control high
## 9    Control low  Control low  Control low  Control low
## 10   Control med  Control med  Control med  Control med
## 11  Control high Control high Control high Control high
## 12     Control N    Control N    Control P    Control P
## 13     Control N    Control N    Control P    Control P
## 14     Control N    Control N    Control P    Control P
## 15     Control N    Control N    Control P    Control P
## 16     Control N    Control N    Control P    Control P
```

```r
# create 96-well plate control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P",
floor(nr/3))))
cmap = cmap[seq(1,nr,2),]
cmap
```

```
##                X1           X2
## 1      Control P    Control N
## 3      Control P    Control N
## 5      Control P    Control N
## 7    Control med  Control med
## 9    Control low  Control low
## 11  Control high Control high
## 13     Control N    Control P
## 15     Control N    Control P
```

2.3   Distribution of Compounds

Some normalization methods implemented in `highSCREEN` assume that the compounds are distributed randomly in the plate. If there are different concentrations of the same compounds in the plate, some of the implemented normalization methods may not be biologically valid[2].

---

[2]See Section 4 for more details regarding within plate normalization methods and their applicability.

3 Assay and Activity Measurement

The package can handle single readings as well as multiple readings/replicates of 384-well plates. In the following example it is assumed that the OD data are collected at two different time instances t_0 and t_1, respectively. Each time-specific data set consists of replicates. The user can specify which plate and replicate to extract from the input data sets. When extracting a specific plate and replicate, the output data are organized in the form of a list consisting of two elements. Elements `dat0` and `dat1` contain the plates (compounds and controls) of t_0 and t_1 specific data sets, respectively.

```
set.seed(1234)
library(highSCREEN)
nc = 24
nr = 16

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicate to create t0-specific data set
replicates_t0 = list(r1, r2, r3)
names(replicates_t0) = c("R1", "R2", "R3")

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicate to create t1-specific data set
replicates_t1 = list(r1, r2, r3)
names(replicates_t1) = c("R1", "R2", "R3")

# extract plate 3, replicate 2
extractplate(replicates_t0, replicates_t1, plate=3, replicate=2)


## $dat0
##           [,1]        [,2]      [,3]        [,4]         [,5]      [,6]
## [1,] 1.2150536 0.6040689 2.0624810 1.20673873 1.038110146 1.1271329
## [2,] 0.3157640 0.1426293 1.4114046 0.41970989 0.871449472 0.1955291
## [3,] 0.2055698 0.3741440 0.4071761 3.19590120 0.494249837 0.6391582
## [4,] 3.3960635 0.1660620 0.6512008 0.84079580 1.309190860 1.1296779
## [5,] 1.1024646 0.1139629 1.2016658 1.67432651 0.143056066 0.6886715
## [6,] 0.7893944 0.1155534 1.7970624 1.42836844 0.001313548 0.5944834
## [7,] 0.5388331 0.7418648 0.8028380 1.75726622 0.848196597 2.8643468
## [8,] 0.9164891 1.9188941 0.6979752 0.04314039 0.227479443 0.6152848
##           [,7]         [,8]       [,9]      [,10]      [,11]      [,12]
## [1,] 2.1624690 1.011654377 0.67308682 0.6112307 0.54721330 0.29972049
## [2,] 1.0137177 0.132361320 0.51439651 0.1904094 0.93329983 0.38118803
## [3,] 0.4746346 0.133177628 1.14801191 0.8268475 1.16388240 0.26767327
## [4,] 0.1070204 0.506401780 0.10293639 2.1653462 1.44963736 0.92749466
## [5,] 0.6021049 0.172793618 0.04123062 1.8519568 0.14006729 0.04230266
## [6,] 1.1544213 0.655928704 0.64400786 0.7476865 1.20643412 0.05275033
## [7,] 0.3650888 0.346083490 0.49084092 0.5915064 0.02509449 0.86320262
## [8,] 0.7659322 0.006265211 0.19069132 1.9185582 1.64235904 0.10369655
##
## $dat1
##            [,1]       [,2]       [,3]      [,4]      [,5]      [,6]
## [1,] 0.01153115 0.05004926 1.22189865 2.1789535 1.6602134 0.5518484
## [2,] 0.52816051 0.47547105 1.36490125 1.4207221 0.9658158 0.9542928
## [3,] 0.26237252 0.77404562 0.06520022 0.3104332 0.7513957 1.4439064
## [4,] 1.12361059 1.77287949 1.58770807 0.5319105 1.2270525 1.0217189
## [5,] 0.05532339 0.09447017 0.92751980 0.4115962 0.5543107 0.2311760
## [6,] 0.75817883 0.08743515 0.08046895 1.1067062 0.3800668 1.1217940
## [7,] 1.31672226 0.36062217 1.23444910 1.6680156 0.1701018 1.3002615
## [8,] 0.76800213 0.94641655 2.10159395 0.8925480 1.4429385 1.5986195
##           [,7]      [,8]       [,9]      [,10]     [,11]      [,12]
## [1,] 0.3600328 0.3260510 0.26010014 0.3803121 0.3632250 0.64218268
## [2,] 1.2979704 0.5867729 2.04793696 1.6628550 0.4512819 2.22606725
## [3,] 1.3448522 1.4756239 1.35294141 1.5699014 1.1655252 0.17283428
## [4,] 0.8948498 1.7675311 0.05831046 1.2859024 1.4498347 0.54745230
## [5,] 2.3064164 0.5932569 0.49850048 0.6323894 0.4032315 2.40240516
## [6,] 0.9404297 0.2082568 0.12624092 1.3501161 0.1291349 0.02339644
## [7,] 0.9438240 0.9430414 0.18652522 0.1146617 0.6742733 0.69777449
## [8,] 1.0216503 0.7475638 0.40275905 0.4027099 1.8020764 0.16960283
```

4   Within Plate Normalization

The user can normalize individual 96-well plates and replicates via the package function `normplate()`, which utilizes one of the implemented normalization methods. Currently, `highSCREEN` implements the `b-score`, the `c-score` and `z-score` normalization methods. It is worth emphasizing that the `b-score` and `z-score` normalization methods are biologically plausible if compounds are randomly distributed within a plate. If there are different concentrations of the same compounds, the `c-score` normalization method is more appropriate.

The input data for normalization can be taken from the output of the package function `extractplate()`. However, the user must provide a 96-well plate control map. The format of the control map and the normalized data are shown in the following example.

```
set.seed(1234)
library(highSCREEN)
nc = 24
nr = 16

# create control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P", floor(nr/3))))
cmap = cmap[seq(1,nr,2),]
cmap

##              X1          X2
## 1     Control P    Control N
## 3     Control P    Control N
## 5     Control P    Control N
## 7   Control med  Control med
## 9   Control low  Control low
## 11 Control high Control high
## 13    Control N    Control P
## 15    Control N    Control P


# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicate for the t0-specific data
replicates_t0 = list(r1, r2, r3)
names(replicates_t0) = c("R1", "R2", "R3")

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicate for the t1-specific data
replicates_t1 = list(r1, r2, r3)
names(replicates_t1) = c("R1", "R2", "R3")

# extract plate 1, replicate 1
dat = extractplate(replicates_t0, replicates_t1, plate=1, replicate=1)

# normalize using c-score
head(normplate("Main Plate 1", dat[["dat0"]], dat[["dat1"]], cmap, plate=1, replicate=1, norm="cscore",
 poscont="Control P", negcont="Control N"))

##       MainPlate Time Plate Replicate   Norm well row col        S
## 1 Main Plate 1    0     1         1 cscore   A1   A   1   1.207066
## 2 Main Plate 1    0     1         1 cscore   A2   A   2 165.388950
## 3 Main Plate 1    0     1         1 cscore   A3   A   3 402.215226
## 4 Main Plate 1    0     1         1 cscore   A4   A   4  21.920877
## 5 Main Plate 1    0     1         1 cscore   A5   A   5 190.136763
## 6 Main Plate 1    0     1         1 cscore   A6   A   6 118.852789
##    welltype
## 1 Control P
## 2  Compound
## 3  Compound
```

```
## 4   Compound
## 5   Compound
## 6   Compound


# normalize using b-score (medpolish)
head(normplate("Main Plate 1", dat[["dat0"]], dat[["dat1"]], cmap, plate=1, replicate=1, norm="bscore"))


## 1: 36.28535
## Final: 36.04823
## 1: 36.18311
## Final: 35.983
##        MainPlate Time Plate Replicate   Norm well row col         S
## 1 Main Plate 1    0     1         1 bscore   A1   A   1  1.2070657
## 2 Main Plate 1    0     1         1 bscore   A2   A   2 -1.0373700
## 3 Main Plate 1    0     1         1 bscore   A3   A   3 -1.2455040
## 4 Main Plate 1    0     1         1 bscore   A4   A   4  0.8786010
## 5 Main Plate 1    0     1         1 bscore   A5   A   5 -0.3495251
## 6 Main Plate 1    0     1         1 bscore   A6   A   6  0.2527863
##    welltype
## 1 Control P
## 2  Compound
## 3  Compound
## 4  Compound
## 5  Compound
## 6  Compound


# normalize using z-score
head(normplate("Main Plate 1", dat[["dat0"]], dat[["dat1"]], cmap, plate=1, replicate=1, norm="zscore"))


##        MainPlate Time Plate Replicate   Norm well row col         S
## 1 Main Plate 1    0     1         1 zscore   A1   A   1  1.2070657
## 2 Main Plate 1    0     1         1 zscore   A2   A   2 -0.1690751
## 3 Main Plate 1    0     1         1 zscore   A3   A   3 -1.2059072
## 4 Main Plate 1    0     1         1 zscore   A4   A   4  0.4590321
## 5 Main Plate 1    0     1         1 zscore   A5   A   5 -0.2774217
## 6 Main Plate 1    0     1         1 zscore   A6   A   6  0.0346615
##    welltype
## 1 Control P
## 2  Compound
## 3  Compound
## 4  Compound
## 5  Compound
## 6  Compound
```

5   Cross-Plate Normalization

Currently not implemented.

6   Reformatting Normalized Data of Replicates

The package allows for reformatting the normalized data for easier interpretation via the function `formatRESULT()`. In the following example the normalized data of replicates are combined and reformatted for easier visualization.

```
set.seed(1234)
library(highSCREEN)
nc = 24
nr = 16

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P", floor(nr/3))))
cmap = cmap[seq(1,nr,2),]

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicates for the t0-specific data
replicates_t0 = list(r1, r2, r3)
```

```
names(replicates_t0) = c("R1", "R2", "R3")

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicate for the t1-specific data
replicates_t1 = list(r1, r2, r3)
names(replicates_t1) = c("R1", "R2", "R3")

# extract plate 1, all replicates
dat1 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=1)
dat2 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=2)
dat3 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=3)

# normalize data of all replicates
res1 = normplate("Main Plate 1", dat1[["dat0"]], dat1[["dat1"]], cmap, plate=1, replicate=1, norm="zscore")
res2 = normplate("Main Plate 1", dat2[["dat0"]], dat2[["dat1"]], cmap, plate=1, replicate=2, norm="zscore")
res3 = normplate("Main Plate 1", dat3[["dat0"]], dat3[["dat1"]], cmap, plate=1, replicate=3, norm="zscore")

# reformat data of all replicates
head(formatRESULT(rbind(res1, res2, res3), replicate="Replicate", score="S", t="Time"))

##                      ID   MainPlate Plate   Norm well row col  welltype
## 1 Main Plate 1_1_A1 Main Plate 1     1 zscore   A1   A   1 Control P
## 2 Main Plate 1_1_A2 Main Plate 1     1 zscore   A2   A   2  Compound
## 3 Main Plate 1_1_A3 Main Plate 1     1 zscore   A3   A   3  Compound
## 4 Main Plate 1_1_A4 Main Plate 1     1 zscore   A4   A   4  Compound
## 5 Main Plate 1_1_A5 Main Plate 1     1 zscore   A5   A   5  Compound
## 6 Main Plate 1_1_A6 Main Plate 1     1 zscore   A6   A   6  Compound
##         S0_R1       S0_R2       S0_R3       S1_R1       S1_R2       S1_R3
## 1   1.2070657  0.83812938  0.3734610  0.3898951  0.05987781  0.30662212
## 2  -0.1690751  0.51125068  0.2421926  2.5087697  1.16922097 -0.30944605
## 3  -1.2059072 -0.74947348 -0.3606832 -0.5092629 -0.96487503 -1.26814744
## 4   0.4590321 -0.09137493 -1.0919842 -0.3357455  0.26074171 -0.56183908
## 5  -0.2774217 -1.23745258 -1.0676386  0.3536054 -1.10089981 -0.04345365
## 6   0.0346615 -0.37110086  0.6776791  1.7928550 -0.46648819  0.22701845
```

7 QC

The package implements several QC procedures to determine if a plate is eligible for further analyses. Currently, all implemented quality checks are within plate QC procedures. Across plates QC procedures are currently not supported.

The package implements three QC procedures via the function `qcplate()`. The first QC procedure (QC1) checks if all control replicates from t_0-specific data set are above a pre-defined threshold value. If any control replicate falls below that threshold, it is determined that the plate fails QC1. The second QC procedure (QC2) computes the mean of all positive controls for a given replicate. The plate passes QC2 iff all of the three means are below a pre-defined threshold value. The third QC procedure (QC3) assumes that there are in total five different types of controls, negative controls (hypothetically denoted as "Control N"), positive controls ("Control P") and an additional control that is represented in three different concentrations ("Control low", "Control med" and "Control high"). The QC3 procedure computes the means of all t_1-specific replicates of a given control and concentration, and compares them. In order for the plate to pass QC3, the following must be satisfied[3]:

$$\text{mean(Control N)} < \text{mean(Control low)} < \text{mean(Control med)} < \text{mean(Control high)} < \text{mean(Control P)}. \quad (1)$$

A plate passes the overall QC iff it passes all individual QC procedures. This provides a

---

[3]The controls (except positive and negative) need to be specified as an input to `qcplate()` in the same order as they appear in the QC3 condition (1).

conservative QC control. The code below demonstrates the use of the package QC capability on a single 96-well plate.

```r
set.seed(1234)
library(highSCREEN)
nc = 24
nr = 16

# create 1st replicates of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P", floor(nr/3))))
cmap = cmap[seq(1,nr,2),]

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicate for the t0-specific data
replicates_t0 = list(r1, r2, r3)

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicates for the t1-specific data
replicates_t1 = list(r1, r2, r3)
names(replicates_t1) = c("R1", "R2", "R3")

# extract plate 1, replicate 1
dat11 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=1)

# extract plate 1, replicate 2
dat12 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=2)

# extract plate 1, replicate 3
dat13 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=3)

# no normalizion (norm="raw")
res11 = normplate("Main Plate 1", dat11[["dat0"]], dat11[["dat1"]], cmap, plate=1, replicate=1, norm="raw")


## [1] "raw"


res12 = normplate("Main Plate 1", dat12[["dat0"]], dat12[["dat1"]], cmap, plate=1, replicate=2, norm="raw")


## [1] "raw"


res13 = normplate("Main Pltae 1", dat13[["dat0"]], dat13[["dat1"]], cmap, plate=1, replicate=3, norm="raw")


## [1] "raw"


# combine 3 replicates
res1 = rbind(res11, res12, res13)

# reformat result
res1 = formatRESULT(res1, replicate="Replicate", score="S", t="Time")

# perform QC
qcplate(res1, poscont="Control P", negcont="Control N", qc1.val=0.225, qc2.val=2,
addcont=c("Control low", "Control med", "Control high"), welltype="welltype")


##   passQC1 passQC2 passQC3 passQC
## 1   FALSE   FALSE   FALSE  FALSE
```

8 Plate Statistical Effect Size

The package provides additional plate-based assessment, by computing `z-factor` and strictly standardized mean difference (`ssmd`) of a 96-well plate. The following example computes `z-factor` and `ssmd` of a 96-well plate replicate via the package function `zfactor.ssmd()`.

```
set.seed(1234)
library(highSCREEN)

nc = 24
nr = 16

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P", floor(nr/3))))
cmap = cmap[seq(1,nr,2),]

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicates for the t0-specific data
replicates_t0 = list(r1, r2, r3)
names(replicates_t0) = c("R1", "R2", "R3")

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicates for the t1-specific data
replicates_t1 = list(r1, r2, r3)
names(replicates_t1) = c("R1", "R2", "R3")

# extract plate 1, replicate 1
dat1 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=1)
# extract plate 1, replicate 2
dat2 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=2)
# extract plate 1, replicate 3
dat3 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=3)

# no normalizion
datraw1 = normplate("Main Plate 1", dat1[["dat0"]], dat1[["dat1"]], cmap, plate=1, replicate=1, norm="raw")


## [1] "raw"


datraw2 = normplate("Main Plate 1", dat2[["dat0"]], dat2[["dat1"]], cmap, plate=1, replicate=2, norm="raw")


## [1] "raw"


datraw3 = normplate("Main Pltae 1", dat3[["dat0"]], dat3[["dat1"]], cmap, plate=1, replicate=3, norm="raw")


## [1] "raw"


# combine 3 replicates
datraw = rbind(datraw1, datraw2, datraw3)

# reformat result
datraw = formatRESULT(datraw, replicate="Replicate", score="S", t="Time")

# compute z-factor and ssmd for each raw compound, replicate 1
zfactor.ssmd(datraw, "Control P", "Control N", "Main Plate 1", 1)

##     MainPlate replicate  ZFactor0  ZFactor1     SSMD0      SSMD1
## 1 Main Plate 1         1 -8.242231 -3.029124 0.3901324 -0.9424345
```

9 Identifying Hits

The user can identify hits via the package function `hits()`, based on specific threshold values. After identifying candidate hits, the user can rank the hits via the package function `rankhits()`, using different selection rules.

9.1 Criteria

Currently, the package implements three criteria for identifying hits. Firstly, the compounds identified as hits should pass QC1 based on the mean of t_0-specific raw replicates. Secondly, the mean of t_0-specific normalized replicates should be smaller than the mean of t_1-specific normalized replicates. Thirdly, the mean of t_1-specific normalized replicates should be larger than a pre-defined threshold value. The output of `hits()` contains columns `IND2` and `IND3`, which specify which compound passes (TRUE) or fails (FALSE) the second and third criteria respectively. Only compounds that pass QC1 and belong to plates that passed overall QC are included in the output of `hits()`.

```
set.seed(1234)
library(highSCREEN)

nc = 24
nr = 16

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)*0.01), nr, nc)

# create control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P", floor(nr/3))))
cmap = cmap[seq(1,nr,2),]

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)*0.01), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)*0.01), nr, nc)

# combine all replicates for the t0-specific data
replicates_t0 = list(r1, r2, r3)
names(replicates_t0) = c("R1", "R2", "R3")

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicate for the t1-specific data
replicates_t1 = list(r1, r2, r3)
names(replicates_t1) = c("R1", "R2", "R3")

# extract plate 1, replicate 1
dat1 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=1)

# extract plate 1, replicate 2
dat2 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=2)

# extract plate 1, replicate 3
dat3 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=3)

# no normalizion
datraw1 = normplate("Main Plate 1", dat1[["dat0"]], dat1[["dat1"]], cmap, plate=1, replicate=1, norm="raw")
```

```
## [1] "raw"
```

```
datraw2 = normplate("Main Plate 1", dat2[["dat0"]], dat2[["dat1"]], cmap, plate=1, replicate=2, norm="raw")
```

```
## [1] "raw"
```

```
datraw3 = normplate("Main Pltae 1", dat3[["dat0"]], dat3[["dat1"]], cmap, plate=1, replicate=3, norm="raw")
```

```
## [1] "raw"
```

```
# combine 3 replicates
datraw = rbind(datraw1, datraw2, datraw3)

# reformat result
datraw = formatRESULT(datraw, replicate="Replicate", score="S", t="Time")

# c-score normalization
datnorm1 = normplate("Main Plate 1", dat1[["dat0"]], dat1[["dat1"]], cmap, plate=1, replicate=1, norm="cscore",
poscont="Control P", negcont="Control N")
```

```
datnorm2 = normplate("Main Plate 1", dat2[["dat0"]], dat2[["dat1"]], cmap, plate=1, replicate=2, norm="cscore",
poscont="Control P", negcont="Control N")
datnorm3 = normplate("Main Pltae 1", dat3[["dat0"]], dat3[["dat1"]], cmap, plate=1, replicate=3, norm="cscore",
poscont="Control P", negcont="Control N")

# combine 3 replicates
datnorm = rbind(datnorm1, datnorm2, datnorm3)

# reformat result
datnorm = formatRESULT(datnorm, replicate="Replicate", score="S", t="Time")

# identify hits
head(hits(datraw, datnorm, qc.mainplates="Main Plate 1", qc1.val=0.225, hit.val=3))

##                     ID  MainPlate Plate   Norm well row col welltype
## 5   Main Plate 1_1_A5 Main Plate 1     1 cscore   A5   A   5 Compound
## 6   Main Plate 1_1_A6 Main Plate 1     1 cscore   A6   A   6 Compound
## 7   Main Plate 1_1_A7 Main Plate 1     1 cscore   A7   A   7 Compound
## 8   Main Plate 1_1_A8 Main Plate 1     1 cscore   A8   A   8 Compound
## 11 Main Plate 1_1_A11 Main Plate 1     1 cscore  A11   A  11 Compound
## 23 Main Plate 1_1_B11 Main Plate 1     1 cscore  B11   B  11 Compound
##       S0_R1       S0_R2      S0_R3     S1_R1     S1_R2      S1_R3 IND2 IND3
## 5  190.1368 -279.73258 322.36378  75.93379 129.55053  145.46611 TRUE TRUE
## 6  118.8528   23.69479  46.05115 186.03993  88.95740  339.23544 TRUE TRUE
## 7  138.3241  -43.64655 -28.22197 135.15912 -54.62282   71.21409 TRUE TRUE
## 8  307.6970  340.59177 -12.29621 191.15576  28.91436  879.95351 TRUE TRUE
## 11 356.4831  -89.51039  79.66726  18.25135 138.49332 1479.93856 TRUE TRUE
## 23 234.5600 -107.33533 278.90875  61.25786 125.86286 1491.08000 TRUE TRUE
```

9.2   Selection Rules

After identifying candidate hits, the next step is to rank them according to certain criteria/rules. The package incorporates several criteria for ranking candidate hits. One of the criteria is based on the mean of the replicates. The candidate hits are sorted according to decreasing value of the mean. Additionally, the package computes for each candidate hit, the standard deviation (SD) based on the replicates, the coefficient of variation (CV) as the ratio of the mean and standard deviation, and other parameters such as whether a compound CV is within $1.5 * IQR$, where IQR is the inter-quartile range computed from all candidate hit CVs. These additional parameters can be helpful to the user in developing their own customized hit selection rules. An example of ranking candidate hits based on the mean of replicate scores is shown below.

```
set.seed(1234)
library(highSCREEN)

nc = 24
nr = 16

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)*0.01), nr, nc)

# create control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P", floor(nr/3))))
cmap = cmap[seq(1,nr,2),]

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)*0.01), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)*0.01), nr, nc)

# combine all replicates for the t0-specific data
replicates_t0 = list(r1, r2, r3)
names(replicates_t0) = c("R1", "R2", "R3")

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicates for the t1-specific data
```

```
replicates_t1 = list(r1, r2, r3)
names(replicates_t1) = c("R1", "R2", "R3")

# extract plate 1, replicate 1
dat1 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=1)
# extract plate 1, replicate 2
dat2 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=2)
# extract plate 1, replicate 3
dat3 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=3)

# no normalizion
datraw1 = normplate("Main Plate 1", dat1[["dat0"]], dat1[["dat1"]], cmap, plate=1, replicate=1, norm="raw")


## [1] "raw"


datraw2 = normplate("Main Plate 1", dat2[["dat0"]], dat2[["dat1"]], cmap, plate=1, replicate=2, norm="raw")


## [1] "raw"


datraw3 = normplate("Main Pltae 1", dat3[["dat0"]], dat3[["dat1"]], cmap, plate=1, replicate=3, norm="raw")


## [1] "raw"


# combine 3 replicates
datraw = rbind(datraw1, datraw2, datraw3)

# reformat result
datraw = formatRESULT(datraw, replicate="Replicate", score="S", t="Time")

# c-score normalization
datnorm1 = normplate("Main Plate 1", dat1[["dat0"]], dat1[["dat1"]], cmap, plate=1, replicate=1, norm="cscore",
poscont="Control P", negcont="Control N")
datnorm2 = normplate("Main Plate 1", dat2[["dat0"]], dat2[["dat1"]], cmap, plate=1, replicate=2, norm="cscore",
poscont="Control P", negcont="Control N")
datnorm3 = normplate("Main Pltae 1", dat3[["dat0"]], dat3[["dat1"]], cmap, plate=1, replicate=3, norm="cscore",
poscont="Control P", negcont="Control N")

# combine 3 replicates
datnorm = rbind(datnorm1, datnorm2, datnorm3)

# reformat result
datnorm = formatRESULT(datnorm, replicate="Replicate", score="S", t="Time")

# identify hits
h = hits(datraw, datnorm, qc.mainplates="Main Plate 1", qc1.val=0.225, hit.val=3)

# rank hits in descending order of mean of t1-specific replicate scores "ma"
head(rankhits(h))


##                 ID    MainPlate Plate   Norm well row col welltype
## 93  Main Plate 1_1_H9 Main Plate 1     1 cscore   H9   H   9 Compound
## 23 Main Plate 1_1_B11 Main Plate 1     1 cscore  B11   B  11 Compound
## 11 Main Plate 1_1_A11 Main Plate 1     1 cscore  A11   A  11 Compound
## 66  Main Plate 1_1_F6 Main Plate 1     1 cscore   F6   F   6 Compound
## 63  Main Plate 1_1_F3 Main Plate 1     1 cscore   F3   F   3 Compound
## 68  Main Plate 1_1_F8 Main Plate 1     1 cscore   F8   F   8 Compound
##        S0_R1       S0_R2      S0_R3     S1_R1      S1_R2      S1_R3 IND2 IND3
## 93   301.9929 -174.38692 185.015900 14.567208 -21.31639 1728.583 TRUE TRUE
## 23   234.5600 -107.33533 278.908749 61.257857 125.86286 1491.080 TRUE TRUE
## 11   356.4831  -89.51039  79.667265 18.251352 138.49332 1479.939 TRUE TRUE
## 66  -170.3251 -193.59575   7.090824 55.847542  89.44584 1485.904 TRUE TRUE
## 63  -293.8078 -205.33634 250.975582 17.017140 115.50113 1419.763 TRUE TRUE
## 68   297.0844  170.26893  -2.107865 -9.687373  21.84226 1454.098 TRUE TRUE
##        diff        m0         s0         rs0        m1        s1       rs1
## 93 469.7373  104.20729 248.2578   2.3823461 573.9446 1000.1071 1.742515
## 23 424.0224  135.37780 211.3621   1.5612762 559.4002  807.5047 1.443519
## 11 430.0144  115.54664 225.1511   1.9485736 545.5611  811.4250 1.487322
## 66 662.6757 -118.94333 109.7672  -0.9228529 543.7324  816.1173 1.500954
## 63 600.1499  -82.72286 292.3573  -3.5341778 517.4270  782.9957 1.513248
## 68 333.6690  155.08182 150.1732   0.9683482 488.7508  836.1635 1.710817
##    ind_below ind_above   ind
## 93      TRUE      TRUE  TRUE
## 23      TRUE      TRUE  TRUE
## 11      TRUE      TRUE  TRUE
## 66      TRUE      TRUE  TRUE
## 63      TRUE      TRUE  TRUE
## 68      TRUE      TRUE  TRUE
```

10 Visualization

`highSCREEN` incorporates several capabilities for compounds and controls visualization. This capability is useful when determining threshold values for QC procedures. The user can plot the density of a particular control via the function `plotcont()`. In the following example the density of positive control OD values, the density of negative control OD values and the density of low, medium and high concentration control OD values are plotted in three separate plots. Additionally, the user can plot single plate heat maps using the function `plotplate()` as shown in the next example.

```r
set.seed(1234)
library(highSCREEN)
library(gplots)

nc = 24
nr = 16

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create control map
cmap = data.frame(X1=c(rep("Control P", floor(nr/3)), rep(c("Control low", "Control med", "Control high"),
(floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control N", floor(nr/3))), X2=c(rep("Control N", floor(nr/3)),
rep(c("Control low", "Control med", "Control high"), (floor(nr/3)+nr-3*floor(nr/3))/3), rep("Control P", floor(nr/3))))
cmap = cmap[seq(1,nr,2),]

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicates for the t0-specific data
replicates_t0 = list(r1, r2, r3)
names(replicates_t0) = c("R1", "R2", "R3")

# create 1st replicate of data matrix with compounds and controls
r1 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 2nd replicate of data matrix with compounds and controls
r2 = matrix(abs(rnorm(nr*nc)), nr, nc)

# create 3rd replicate of data matrix with compounds and controls
r3 = matrix(abs(rnorm(nr*nc)), nr, nc)

# combine all replicates for the t1-specific data
replicates_t1 = list(r1, r2, r3)
names(replicates_t1) = c("R1", "R2", "R3")

# extract plate 1, replicate 1
dat11 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=1)
# extract plate 1, replicate 2
dat12 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=2)
# extract plate 1, replicate 3
dat13 = extractplate(replicates_t0, replicates_t1, plate=1, replicate=3)

# no normalizion (norm="raw")
res11 = normplate("Main Plate 1", dat11[["dat0"]], dat11[["dat1"]], cmap, plate=1, replicate=1, norm="raw")

## [1] "raw"

res12 = normplate("Main Plate 1", dat12[["dat0"]], dat12[["dat1"]], cmap, plate=1, replicate=2, norm="raw")

## [1] "raw"

res13 = normplate("Main Pltae 1", dat13[["dat0"]], dat13[["dat1"]], cmap, plate=1, replicate=3, norm="raw")

## [1] "raw"

# combine 3 replicates
res1 = rbind(res11, res12, res13)
# reformat result
res1 = formatRESULT(res1, replicate="Replicate", score="S", t="Time")

layout(matrix(c(1,2,3), 3, 1, byrow = TRUE))

# plot density of all positive controls
plotcont(subset(res1, welltype=="Control P"), main="Density of Positive Controls", xaxis.marks=seq(-1,5,0.025))

# plot density of all negative controls
plotcont(subset(res1, welltype=="Control N"), main="Density of Negative Controls", xaxis.marks=seq(-1,5,0.025))

# plot density of controls with low, medium and high concentrations
plotcont(subset(res1, welltype=="Control low" | welltype=="Control med" | welltype=="Control high"), main="Density of Controls with Low,
Medium and High Concentrations", xaxis.marks=seq(-1,5,0.025))
```
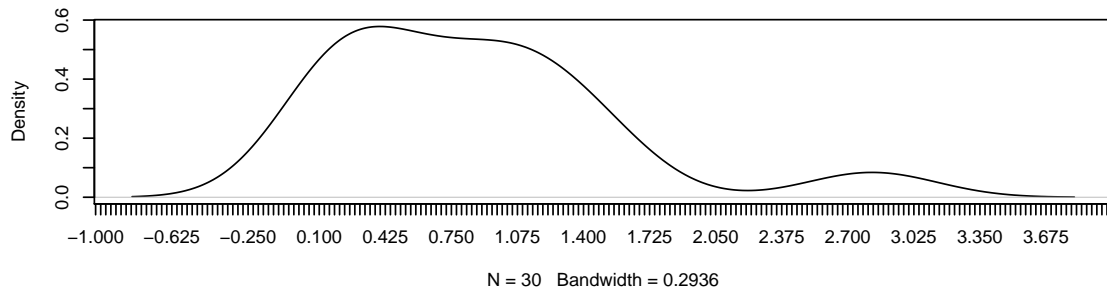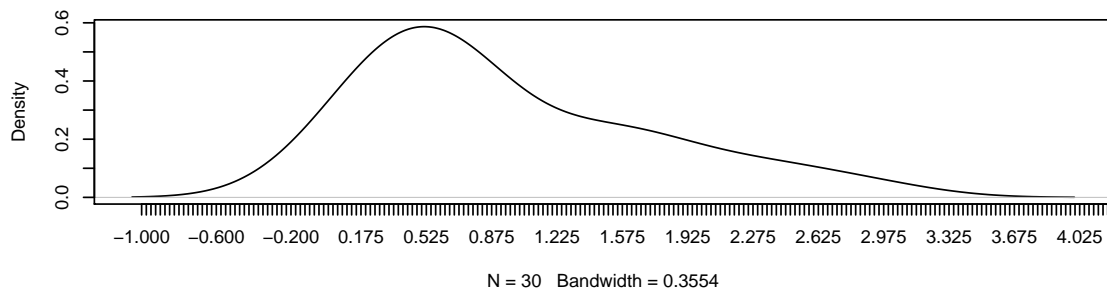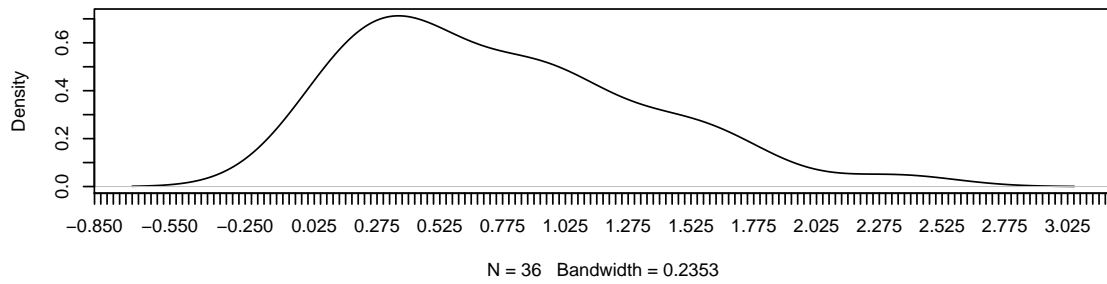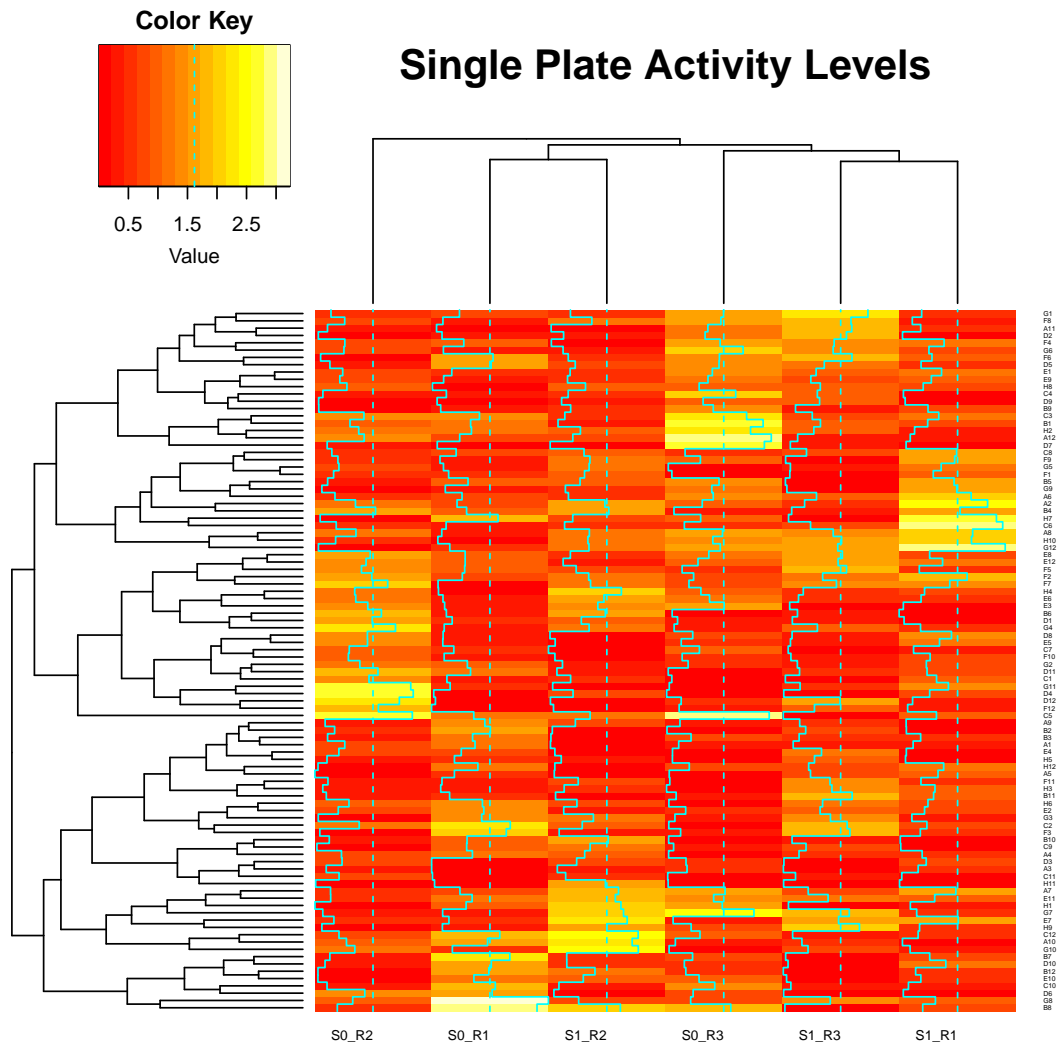
Density of Positive Controls

N = 30   Bandwidth = 0.2936

Density of Negative Controls

N = 30   Bandwidth = 0.3554

**Density of Controls with Low,
Medium and High Concentrations**

N = 36   Bandwidth = 0.2353

```
# plot single plate activity levels
plotplate(res1, main="Single Plate Activity Levels")
```

Single Plate Activity Levels

```
## [10] stringr_1.2.0      bitops_1.0-6      gtools_3.5.0
## [13] evaluate_0.10.1
```

11 Acknowledgement

References

[1] N. Malo, J. A. Hanley, S. Cerquozzi, J. Pelletier, and R. Nadon. Statistical practice in high-throughput screening data analysis. *Nature Biotechnology*, 24(2):167–175, 2006.