# Introduction of how to use R package iScreen.

Rui Zhong

2014-01-27

This is an introduction of how to use our R package iScreen for image-base high-throughput RANi screeninig data analysis. In contrast with traditional HTS, data from image-base HTS are high-content and multidimensional.

First we need to intall the R packgae iScreen and load it into R working session.

```
> library(iScreen)
```

We have two built-in datasets in this packages, autophagy and colocolization, which are both from autophagy study.

```
> head(autophagy)
```

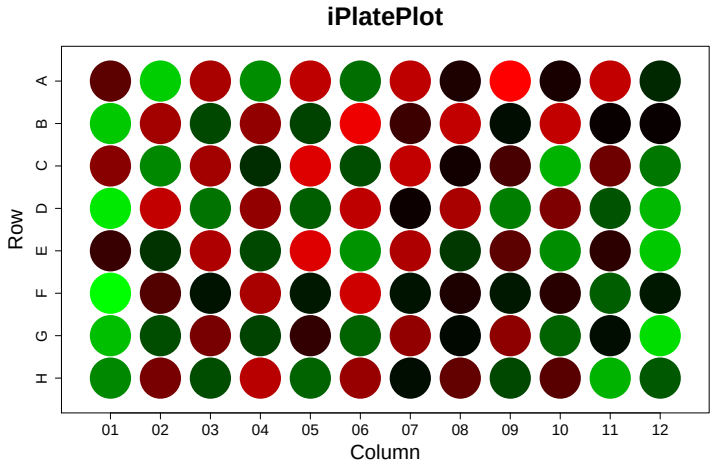
```
  WellID dot.number cell.area cell.number control treatment
1    A01          3      4299         283       1    NCpool
2    A01          8      3207         283       1    NCpool
3    A01         10      6989         283       1    NCpool
4    A01          9      4505         283       1    NCpool
5    A01          2      6307         283       1    NCpool
6    A01         13      5196         283       1    NCpool
```

```
> head(colocolization)
```

```
  WellID        x       y area  mark   col
1    A01 905.0000 854.000    1 green green
2    A01 896.0000 857.500    2 green green
3    A01 890.5000 864.500    4 green green
4    A01 842.7143 875.000    7   red   red
5    A01 903.0000 877.000    1 green green
6    A01 886.7500 879.125    8   red   red
```

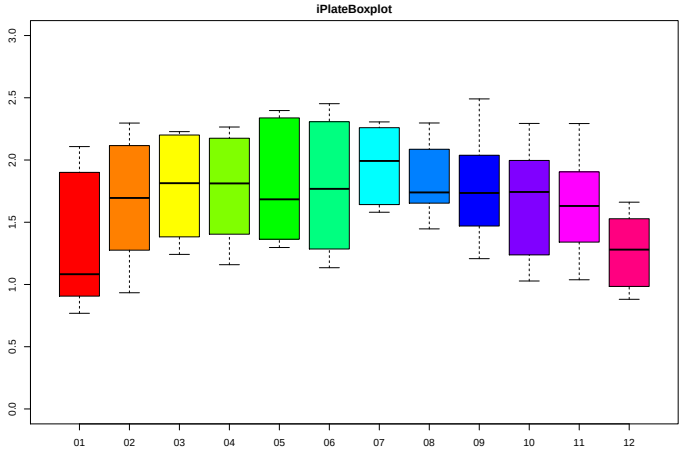
By design, image-base HTS is usually performed on 96- or 384-well plates and therefore visulization of plate is quite useful for primary data analysis and quality contorl. Like in our dataset autophagy, for each well of the plate, we have a Poisson distribuion of dot number which is indicating the autophagy activity. We want to plot mean of dot number in each well.

```
> p1 <- iPlate(autophagy, "dot.number", log=T) # dot.number is log transformed.
[1] "0 value existing and pseudocount added by 1"
> par(mar=c(5, 5, 5, 2))
> iPlatePlot(p1, cex=9, cex.axis=1.5, cex.lab=2, main="iPlatePlot",
+            cex.main=2.5)
```



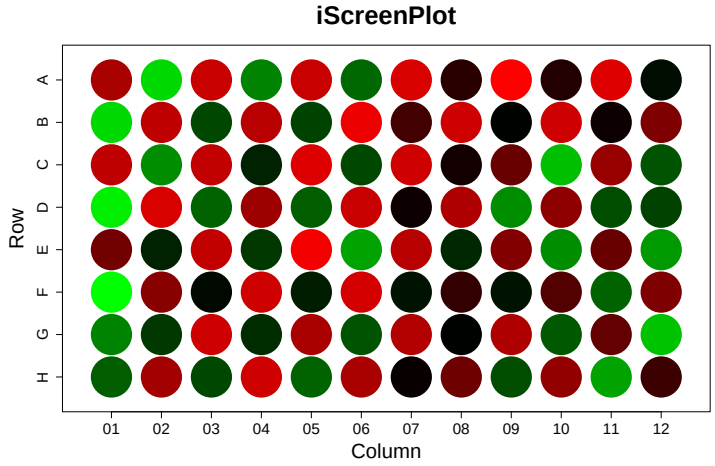

We can also generate a legend for plot by running code iPlateLegend(p1), plot not shown here.
During high-throughput RNAi screening, one concern is position effect. Therefore we have iPlateBoxplot for visualizing data by either row or column.

```
> par(mar=c(5, 5, 2, 2))
> iPlateBoxplot(p1, by="column", ylim=c(0, 3), col=rainbow(12),
+               main="iPlateBoxplot")
```



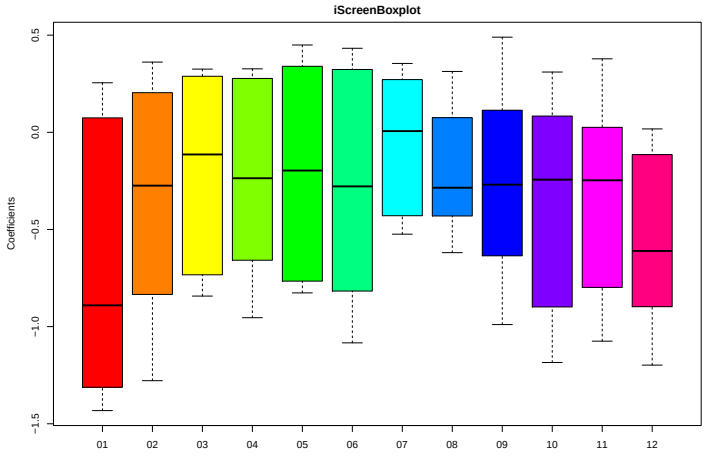
In dataset autophagy, each well has different treatment, and we are interested in knowing if any treatment can reduce the autophagy activity in terms of reducing the dot number. Since dot number assume Poisson distribtuion and therefore we want to fit a Poisson regression for dataset.

```
> fit.auto <- iScreen(autophagy, dot.number~WellID, family=poisson,
+                     control=(autophagy$control == 1))
> head(fit.auto$coefficients)

  WellID   Estimate Std..Error    z.value       p.value
1    A01  0.1663575 0.02177867   7.638551  2.196802e-14
2    A02 -1.2781800 0.04062567 -31.462373 2.843117e-217
3    A03  0.2937617 0.01902658  15.439541  8.873001e-54
4    A04 -0.9541095 0.03443892 -27.704396 6.180612e-169
5    A05  0.3026934 0.01964322  15.409561  1.411677e-53
6    A06 -0.8614450 0.03075278 -28.011939 1.162518e-172

> par(mar=c(5, 5, 5, 2))
> iScreenPlot(fit.auto, cex=9,cex.axis=1.5, cex.lab=2,
+             main="iScreenPlot", cex.main=2.5)
```




We can also check the row or column effect of iScreen object.

```
> par(mar=c(5, 5, 2, 2))
> iScreenBoxplot(fit.auto, by="column", col=rainbow(12),
+                main="iScreenBoxplot", ylab="Coefficients")
```

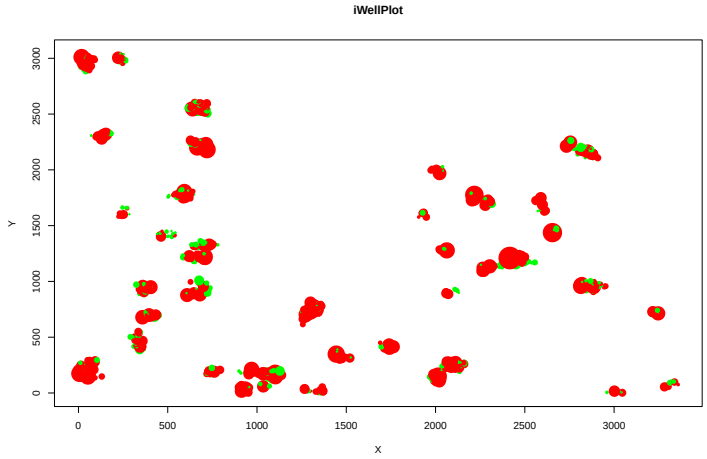



Sometimes we want to perform some custom analysis, and incoperate user-defined function into our analysis funcitons. iScreen provides such funtionality. We demonstrate with dataset colocolization.It this dataset, we have tow kinds of dots (red and green), and are interested in if two kinds of dots are correlated in each well. We write a custom function to calculate mark correlation.
First we want to fit an iWell object and plot it for data visualization.

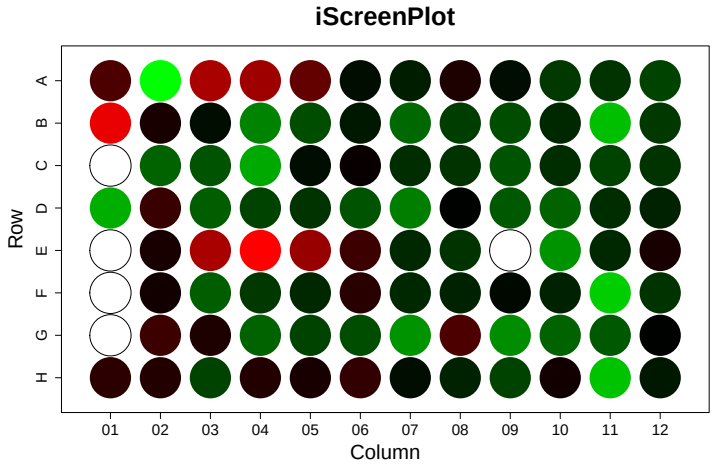
```
> data.well <- colocolization[colocolization$WellID == "A06", ]
> head(data.well)
```

```
    WellID         x        y area  mark   col
275    A06  907.0000 0.250000    4   red   red
276    A06 3046.0000 2.000000    9   red   red
277    A06  916.8333 2.333333    6   red   red
278    A06 1344.7500 3.000000    4   red   red
279    A06  920.8000 3.600000    5   red   red
280    A06  945.5000 4.500000    4 green green
```

```
> colo.well <- iWell(x=data.well$x, y=data.well$y,
+                    d=2*sqrt(data.well$area/3.14),
+                    c=data.well$col, n=10, type=1)
> par(mar=c(5, 5, 5, 2))
> iWellPlot(colo.well, main="iWellPlot")

NULL
```




```
> spatial.cor <- function(data){
+   require(spatstat)
+   x <- ppp(data$x, data$y, marks=data$mark,
+            window=owin(xrange=c(floor(min(data$x)), ceiling(max(data$x))),
+                        yrange=c(floor(min(data$y)), ceiling(max(data$y)))))
+   mk.x <- markcorr(x, r=0:15, f=function(m1, m2){m1 == m2})
+   mk.x <- c(mean(mk.x$iso), 0)
+   names(mk.x) <- c("mark.correlation", "p.value")
+   return(mk.x)
+ }
> fit.colo <- iScreen(colocolization, FUN=spatial.cor)
> head(fit.colo$coefficients)

  WellID mark.correlation p.value
1    A01        1.1836167       0
2    A02        0.6783729       0
3    A03        1.3166384       0
4    A04        1.3071622       0
5    A05        1.2167947       0
6    A06        1.0415508       0

> par(mar=c(5, 5, 5, 2))
> iScreenPlot(fit.colo, cex=9,cex.axis=1.5, cex.lab=2,
+             main="iScreenPlot", cex.main=2.5)
```




White circle in above plot indicates relevant information is missing.