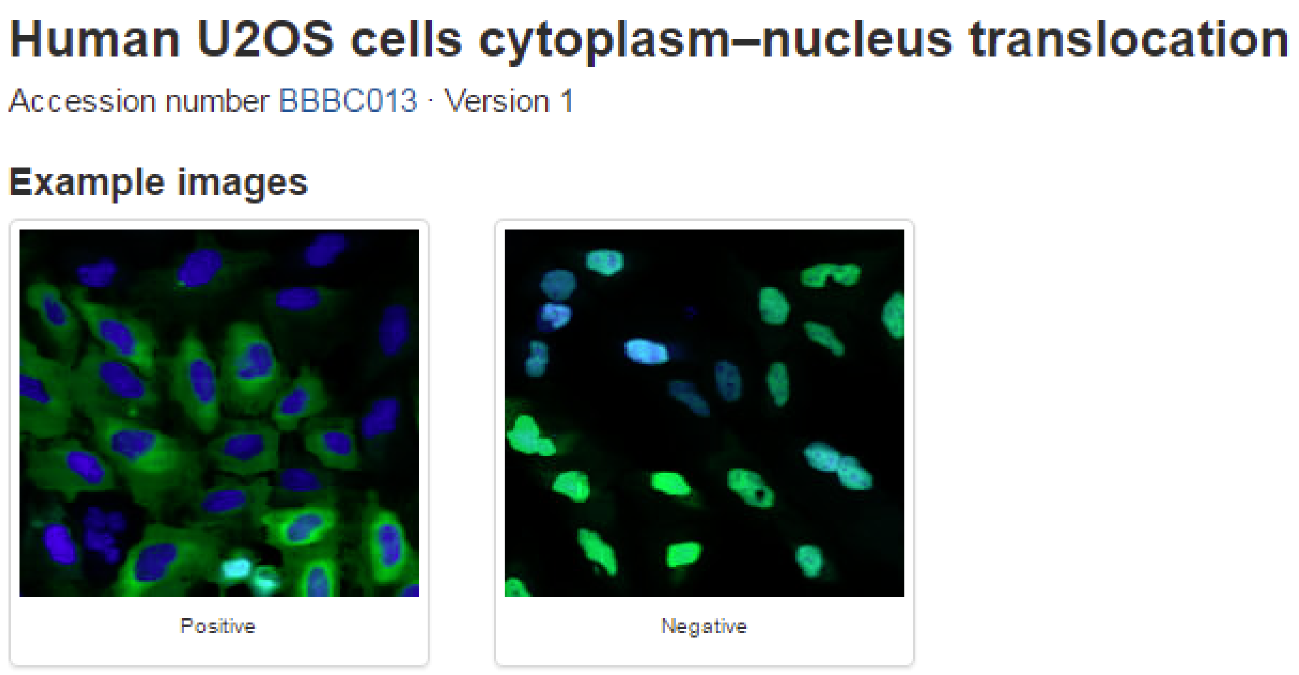
Intro DTD

Steven Wink Ph.D.

Tue 13 Sept 15.30 - 17.30: LCP 06

Note to students: during the exam no knowlegde of R will be required. R is merely used as a tool. However a subset of questions of this practical, or questions closely related to, can be asked during the exam. Make sure you try to understand all these questions, and ask the examinators for help during the practical or during exam-preparation as some of them can be difficult.



Human U2OS cells cytoplasm–nucleus translocation

Accession number BBBC013 · Version 1

Example images

Positive

Negative

Biological application

This 96-well plate has images of cytoplasm to nucleus translocation of the Forkhead (FKHR-EGFP) fusion protein in stably transfected human osteosarcoma cells, U2OS. In proliferating cells, FKHR is localized in the cytoplasm. Even without stimulation, Forkhead is constantly moving into the nucleus, but is transported out again by export proteins. Upon inhibition of nuclear export, FKHR accumulates in the nucleus. In this assay, export is inhibited by blocking PI3 kinase / PKB signaling by incubating cells for 1 h with Wortmannin or with the compound LY294002. Both drugs are considered positive controls in the assay. Nuclei are stained with DRAQ, a DNA stain.

source: https://data.broadinstitute.org/bbbc/BBBC013/ (https://data.broadinstitute.org/bbbc/BBBC013/)

In this systems microscopy workshop:

We will learn some basic methods of analyzing imaging data obtained from a high content imager. First we determine the image analysis parameters, then we will run a pre-made script to analyze all the images.

complete the analysis by following the steps below. There are do’s (numbered) and questions (Q#)

1) In RStudio, go to File -> New File -> R Script*

The new R script is displayed in the text editor pane. This is where you type the commands/ code. Execute a line of your code by pressing *CTRL ENTER*. The line where your cursor is active is executed. You can also select multiple lines or a small part of code to execute in the same manner.

2) Install the following external packages, which contain the needed functionality for image analysis and graphics. If the installation fails, skip and follow the next block.

```
#
#  install.packages(c("tiff", "jpeg", "png", "locfit" , "fftwtools"))
#
#
#  source("http://bioconductor.org/biocLite.R") # bioconductor installation scripts
#
#  biocLite("EBImage")
#
#  install.packages(c("ggplot2","stringr"))
```

if installation is not successful (there seems to be a problem with win binaries at the moment), install the packages in this manner:

```
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/stringi_1.1.1.zi
p", repos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/stringr_1.1.0.zi
p", repos =NULL)

install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/scales_0.4.0.zip",
repos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/Rcpp_0.12.7.zip",
repos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/munsell_0.4.3.zi
p", repos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/colorspace_1.2-6.z
ip", repos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/plyr_1.8.4.zip", r
epos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/ggplot2_2.1.0.zi
p", repos =NULL)

install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/locfit_1.5-9.1.zi
p", repos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/tiff_0.1-5.zip", r
epos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/jpeg_0.1-8.zip", r
epos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/png_0.1-7.zip", re
pos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/fftwtools_0.9-7.zi
p", repos =NULL)

install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/gtable_0.2.0.zip",
repos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/magrittr_1.5.zip",
repos =NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/labeling_0.3.zip",
repos = NULL)
install.packages("https://cran.r-project.org/bin/windows/contrib/3.3/digest_0.6.10.zi
p", repos = NULL)


source("http://bioconductor.org/biocLite.R") # bioconductor installation scripts

biocLite("EBImage")
```

3) Load the installed packages (this has to be done for every R-session)

```
library(ggplot2)
library(EBImage)
library(stringr)
options(EBImage.display = "raster") # set display options
```

EBImage: This package contains our image analysis functions.
for more information on this package you can visit:
https://bioconductor.org/packages/release/bioc/vignettes/EBImage/inst/doc/EBImage-

introduction.html (https://bioconductor.org/packages/release/bioc/vignettes/EBImage/inst/doc/EBImage-introduction.html)
ggplot2: A graphics package
*stringr: A regular expression package for working with strings

4) Execute the following 5 commands, then have a look at the created folder structure.

```
dir.create("~/imageanalysis")

setwd("~/imageanalysis")

dir.create("results")

dir.create("layout")

dir.create("scripts")
```

5) Download and unzip the BBBC013 image set:

```
download.file(url = "http://d1zymp9ayga15t.cloudfront.net/content/Examplezips/ExampleSB
SImages.zip",
               destfile = "images.zip")

unziplist <- unzip( zipfile = "images.zip", list = TRUE) # list all files inside the zi
p
unzipfiles <- unziplist$Name[ grepl(".tif", unziplist$Name)] # pull out all files conta
ining ".tif"
unzip(zipfile = "images.zip",  files = unzipfiles)
```

6) Unpack and load the file containing the metadata information.

```
unzipmetadata <- unziplist$Name[ grepl(".csv", unziplist$Name)] # pull out all files co
ntaining ".tif"
unzip(zipfile = "images.zip", exdir = "layout",  files = unzipmetadata )

metadata <- read.csv( file = "layout/ExampleSBSImages/1049_Metadata.csv")
```

Now that the images are in `imageanalysis/ExampleSBSImages` and the metadata in `imageanalysis/layout/ExampleSBSImages`
we can start our analysis

An image is basically a set of matrixes (containing the pixel values) with headers containing information needed for image display:

7) Look at one of the images from our image set

```
path_to_image <- "ExampleSBSImages/Channel1-23-B-11.tif"

readImage(path_to_image)

display(readImage(path_to_image))
```

Different dyes can be used to observe distinct features, including structures, proteins, and different organelles within cells. This image set contains two channels of which one constitutes the nucleus (Channel2), and the the other channel is FKHR-GFP (Channel1) which can be located in the nucleus and the cytosol.

8) Look at both channels superimposed on one another.

```
image_paths <- paste0("ExampleSBSImages/", dir("ExampleSBSImages")) # define paths to i
mages
image_paths <- image_paths[grepl(".tif$", image_paths)] # keep only paths to tif images

image_paths_dna <- image_paths[grepl("Channel2", image_paths)]
image_paths_fkhr <- image_paths[grepl("Channel1", image_paths)]

image_dna <- readImage( image_paths_dna[13] )
image_fkhr <- readImage( image_paths_fkhr[13] )

display(image_dna)
display(image_fkhr)
rgb_img <- rgbImage(green = image_fkhr, blue = image_dna)
display(rgb_img)
```

Q 1) Which treatment did we display? (the indexing of all the paths was done with `[13]`)
Q 2) Describe the difference between a negative and positve control. Do this by displaying the controls (check the metadata or, go to the url displayed on the first page and look at the plate layout).

Nuclei segmentation and object identification

In the context of image analysis, segmentation is the recognition of objects of interest in images by a computer, using for example the pixel intensity values. Consider an image to be a collection of pixels in the 2D grid/matrix. The stained DNA/ nuclei thus each form a collection/cluster of higher valued pixels than the surrounding background. Setting a threshold/ offset to determine “foreground” objects (the nuclei) vs the lower intensity value pixels of the background is a usefull method to determine which pixels belong to nuclei objects.

There are 3 parameters needed for a correct segmentation.
We will use an adaptive threshholding algorithm, so the threshhold value can change in windows of width X height (in units of pixels). The starting value for the thresholding optimization is the `nuclei.offset` .
Make sure you are happy with your chosen `adaptive.width` , `adaptive.height` and `nuclei.offset` settings because these will be used for the full analysis of all images later on. The adaptive width and height are measured in pixels.

tip: what is the mean intensity value? `mean(nuclei.image)`

Q3) Why would the mean intensity value of the stained nuclei be usefull? (Think of pixel intensity values)
Q4) What would you consider to be the maximum values of the adaptive width and height parameters?

9) Set the parameters and test on the selected nuclei image. Make sure the segmentation is satisfactory by playing with the parameter values

```
mean(image_dna)

adaptive.width <- 10 # set parameter (the size in pixels of the windows in which the op
timized threshold values will be constant )
adaptive.height <- 10  # set parameter
nuclei.offset <-  0.02 # set parameter

# remember to keep looking at the original image to compare:
display(image_dna)

#the ```thresh()``` function will perform the segmentation, this function takes your pa
rameters as arguments

nmask = thresh(image_dna, w=adaptive.width, h=adaptive.height, offset=nuclei.offset)
display(nmask)
```

Q5) Describe what the foreground objects from the segmentation by the `thresh()` look like when displayed.
Q6) Explain what thresholding is - i.e. what information is gained by the algorithm?

To improve the thresholding, often smoothing filters are applied to the raw images - before the segmentation.
Noisy (outlier) pixel intensities are averaged out, making it easier for the algorithm to recognize the cells because the cells then consist of more or less the same pixel intensities, and the background will also have more of the same pixel intensity values. In effect we lowered the intensity variance within the foreground (the cells in our case) and background.

10) filter the raw images before segmenting

```
image_dna_f <- medianFilter(image_dna, size = 2)
display(image_dna_f)
nmask = thresh(image_dna_f, w=adaptive.width, h=adaptive.height, offset=nuclei.offset)
display(nmask)
```

Q7) Desribe the difference you see by increasing the median filter size. Can you explain this? (Do no set a smaller median filter size than 2, the algorithm will take too long to compute)

11) erode & dilate the masked image

```
nmask = opening(nmask, makeBrush(7, shape="disc")) # erosion + dilation
display(nmask)
```

Eroding is removing ‘outer’ pixels from our masked objects in the segmented image, dilating is adding ‘outer’ pixels. Think of two slightly overlapping cells that are mistakingly segmented as 1 object, and how this algorithm will affect this segmented object.

Q8) Did the segmentation improve using erosion/dilation algorithm? Q9) Explain why this would work

12) label and count the identified foreground objects from the mask-image.

```
nmask = bwlabel(nmask) # label nuclei: defining objects (each distincly segmented nucle
i gets a number)
max(nmask) # number of nuclei
```

Cytosol propagation

Here the nuclei objects are used as starting points (seeds) to find the corresponding GFP signal from the FKHR-GFP.

13) Set the `cytosol.offset` for a good segmentation

```
mean(image_fkhr)
cytosol.offset <- 0.08  # set parameter, play with this parameter for optimal result

colorMode(image_fkhr) <- 0 # set to required greyscale colormode
cytosolmask = opening(image_fkhr > cytosol.offset, makeBrush(5, shape="disc")) # erode
+ dilate, threshhold and smooth
display(cytosolmask)
celmask = propagate(image_fkhr, seeds=nmask, mask=cytosolmask) # use cytosol mask to as
sign a cytosol to each nuclei
display(celmask)

res <- paintObjects(celmask, rgb_img, col="#ff00ff")  # paintObjects makes nice outline
s of segmentation results
colorMode(nmask) <- 0
res <- paintObjects(nmask, res, col= "#ffff00") # add nuclei mask outline
display(res) # display final results
```

Q10) Describe the yellow and purple lines. What do they represent?
Q11) Describe the algorithms used to obtain this result in your own words

Compute features

14) Use the computeFeatures function to calculate the intensity data for each cell.
15) Visualize the distribution of the intensity values.

```
nuclei_intensity <- computeFeatures.basic(x = nmask[ , , 1], image_fkhr[ , ,1])[ , 1] #
calculate nuclei intensities of channel 1
cytosol_intensity <- computeFeatures.basic(x = celmask[ , , 1], image_fkhr[ , ,1])[ , 1
] # calculate cell intensities of channel 1

cytosol_intensity <- cytosol_intensity - nuclei_intensity # subtract nuclei intensity

plot(density(cytosol_intensity))
plot(density(nuclei_intensity))
plot(density(cytosol_intensity / nuclei_intensity))
```

Q12) Why was the nuclei intensity subtracted from the cell intensity values?
Q13) Explain what the division `cytosol_intensity / nuclei_intensity` means regarding FKHR levels

Q14) Why is there a small peak at 0 for the density figure of cytosol_intensity?

Analyse the full image set

Now you know what is going on, we will run a function which basically does the same we did before - but for all images.
The function will require your image-analysis parameters as arguments With `source(function name)` we will load this function in R memory so that it can be called.

The `AnalyzeAll.R’ script contains the `AnalyzeAll()` function, this function will load the images from your image folder, store segmentation results as .jpg files
and output the single cell quantitative intensity data.

15) Download the AnalyzeAll.R function from git, and source the function

```
download.file( url = "https://raw.github.com/Hardervidertsie/teaching_intro/master/Anal
yzeAll.R",
destfile = "scripts/AnalyzeAll.R" )





source("scripts/AnalyzeAll.R")
```

16) set the function arguments according to your own parameter optimizations (or use the provided defaults) and then run the function.

```
data_out <- lapply( image_paths_dna, AnalyzeAll,
                      adapt.width  = 10,
                      adapt.height = 10,
                      nuc.offset   = 0.02,
                      cyto.offset  = 0.08
                    )

data_out <- do.call('rbind', data_out)
head(data_out) # preview the data
```

17) Have a look at the segmentations, decide if the parameters are satisfactory. If not - re do the analysis.

From image analysis to data analysis

The images have been analysed, our result consists of quantitative data, representing features we are interested in (the GFP levels in the nucleus and cytosol and the cell count).
First we have to organize the data in a for us meaningfull way.

18) Add human readable metadata by downloading the plate layout file and merging this to our data.

```
download.file( url = "https://raw.github.com/Hardervidertsie/teaching_intro/master/layo
ut-BBBC013.txt",
destfile = "layout/layout-BBBC013.txt" )

layout <- read.delim( file = "layout/layout-BBBC013.txt", sep ="\t")
head(layout)

data_out$image.name <- gsub(".tif", "", data_out$image.name)

data_single_cells <- merge(data_out, layout, by.x = "image.name", by.y = "locationID")
head(data_single_cells)
```

19) Plot the single cell intensity distributions.

```
plot(density(data_single_cells$cytosol_intensity))
plot(density(data_single_cells$nuclei_intensity))
```

Q15) Explain for the cytosol intensity the negative values. What does this mean biologically?
Q16) Explain for the nuclei intensity the right tailed distribution. What does this mean biologically?

20) calculate the population mean of the intensity ratios per replicate and plot the dose response curves for each replicate

```
summarized_data_intensity <- aggregate( ratio_cyto_nucl ~ treatment + dose + replID  ,
data = data_single_cells, FUN = mean)

head(summarized_data_intensity)

ggplot( data = summarized_data_intensity, aes( x = log(dose +1), y = ratio_cyto_nucl, c
olor = replID )) +
  geom_point( aes( group = dose ) ) + facet_wrap( ~treatment, ncol = 1 , scales = "free
_x" ) +
  geom_smooth( )
```

Q17) Are the drugs working? Please explain.

21) Visualize the correlation between the cell count and the intensity ratio

```
summarized_data_cellcount <- aggregate( cellN ~ treatment + dose + replID  , data = dat
a_single_cells, FUN = mean)

plot( x = summarized_data_intensity$ratio_cyto_nucl, y = summarized_data_cellcount$cell
N)
trend <- lm(  summarized_data_cellcount$cellN ~ summarized_data_intensity$ratio_cyto_nu
cl)
abline(trend)
RS <- summary(trend)
text(x= 0.2, y = 260, label = paste("RS:", round( RS$r.squared, digits = 2 )))
text(x= 0.1, y = 270, label = paste("pval slope:", round( RS$coefficients[ , "Pr(>|t
|)"][2], digits = 2 )))
```

Q18) Based on the data explanation on the broad website, what would you expect regarding cell count and the cytosol/ nucleus intensity ratio?
Q19) Explain if you see any evidence that this is the case