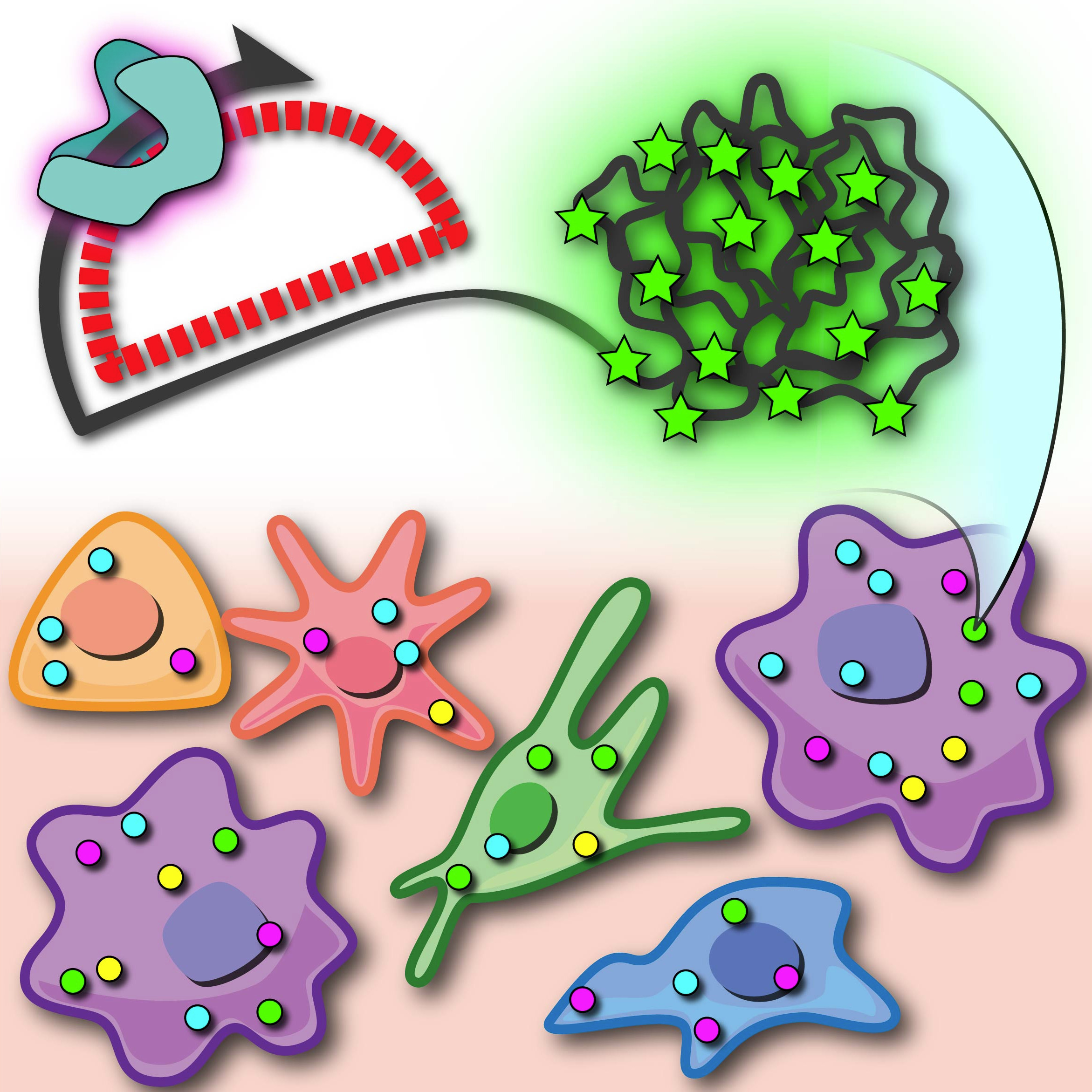
**ISS Data Analysis Manual**



**Olga Surova, Sergio Marco Salas, Erik Samuelsson, Xiaoyan Qian, Markus M. Hilscher, Christoffer Mattsson Langseth, Chika Yokota, Daniel Gyllborg**

**September, 2019**

General Index

[2 Introduction to the pipelines 3](#_Toc20309917)

[3 Standard manual pipeline 5](#_Toc20309918)

[3.1 PART 1. Maximum Intensity Projection of a Z-stack and Stitching. 5](#_Toc20309919)

[3.2 PART 2. Automatic Image Registration with MATLAB. 9](#_Toc20309920)

[3.3 PART3. Image Tiling with MATLAB 10](#_Toc20309921)

[3.4 PART4. CellProfiler Image Analysis. 12](#_Toc20309922)

[3.5 PART5. Sequence Decoding – MATLAB 15](#_Toc20309923)

[3.5.1 Marker Specifiers 16](#_Toc20309924)

[3.5.2 Specifier 16](#_Toc20309925)

[3.5.3 Color Specifiers 16](#_Toc20309926)

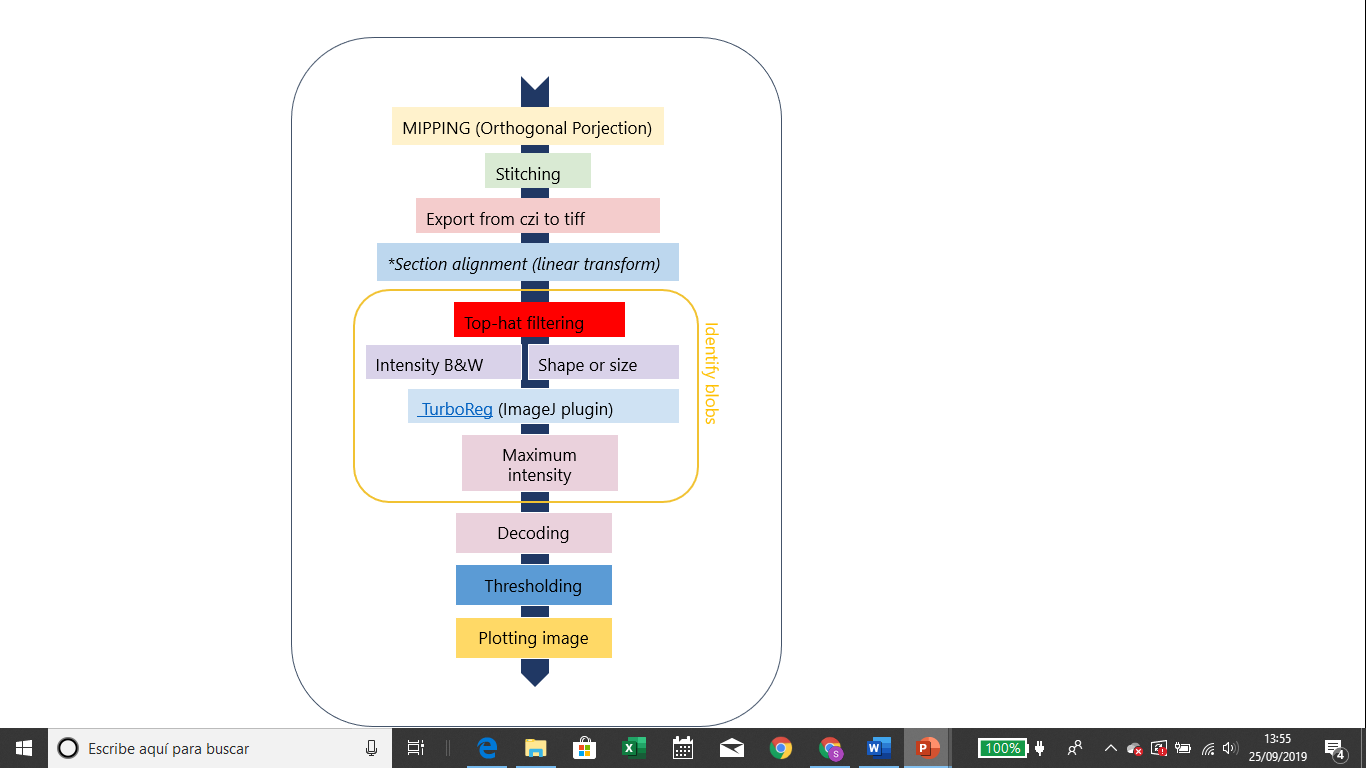
[3.5.4 Specifier 16](#_Toc20309927)

[3.6 PART6. Threshold Sequencing– MATLAB 17](#_Toc20309928)

[3.7 PART7. Plotting Global Sequencing – MATLAB 18](#_Toc20309929)

# Introduction to the pipelines

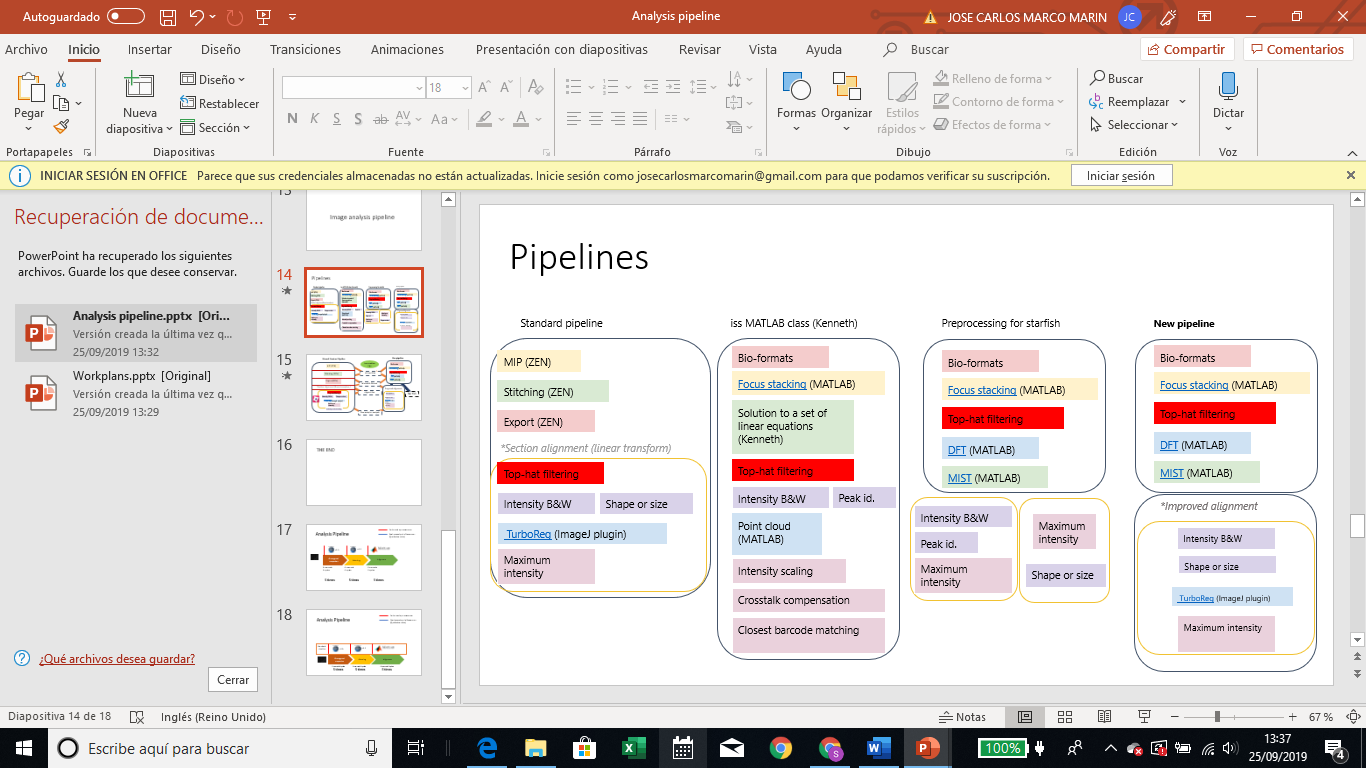
Image analysis pipelines are essential for In Situ Sequencing (ISS) to allow to go from single images of each cycle to the detection of certain genes by decoding their barcodes. At this point, all pipelines need the **same input**, which is **single projects (in CZI format, in the case of ZEISS) imaging each of the cycles**. The number of cycles done will depend on the length of the barcodes. For a typical barcode of 4 nucleotides, like ATTG, four cycles of detection are performed, and 4 different initial files are created, one for each cycle.

All the pipelines share some common steps, which can be summarized as:

In addition to all these steps**, Tilling**, which is dividing the large images in smaller pieces, and retiling, needs to be performed in order to process quicker the whole images. These tiling steps are usually required before identifying the blobs, and re-tilling process is applied before decoding and plotting the image.

There are several pipelines currently available in the group.

General scheme of the different pipelines is presented below:

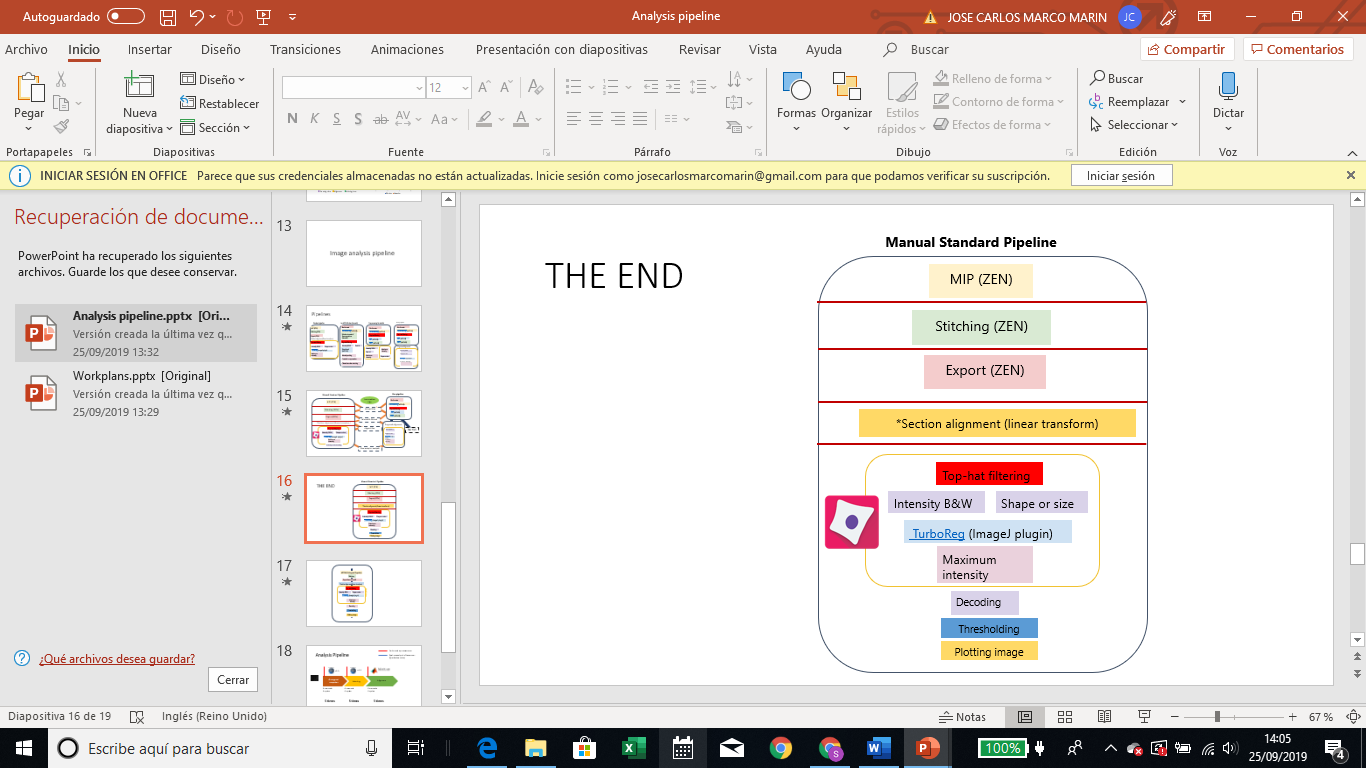


Briefly, **standard pipeline is the most used one** as it can be easily customized. However, there are many other options:

* **Standard pipeline:** perfect for accurate processing. Every step can be customized. Generally, time consuming to run, as every step has to be done “manually”. Requires ZEN, IMAGEJ AND CELLPROFILER, apart from MATLAB.
* **ISS MATLAB class:**
* **Starfish:** It includes a first step of pre-preprocessing of the sample, followed by the implementation of the Starfish pipeline. STARFISH is a common analysis tool implemented for different spatial transcriptomics methods such as MERFISH or osmFISH etc. Therefore, it can be helpful for processing samples and compare them with other methods. It’s quite automatic, so most of the processes are done together. However, it can’t be customized that much. This pipeline requires MATLAB and the specific Starfish package in order to run correctly.

# Standard manual pipeline

The Standard Manual pipeline follows the following Schema:



## PART 1. Maximum Intensity Projection of a Z-stack

**DEFINITION**

**Maximum intensity projection** is a scientific visualization technique that takes 3D data (in our case a Z-stack of microscope images) and turns it into a single 2D image. The projection takes the brightest pixel in each layer and displays that pixel intensity value in the final 2D image. Commonly the projection is created so that it is being viewed “down” into the image stack along the Z axis.

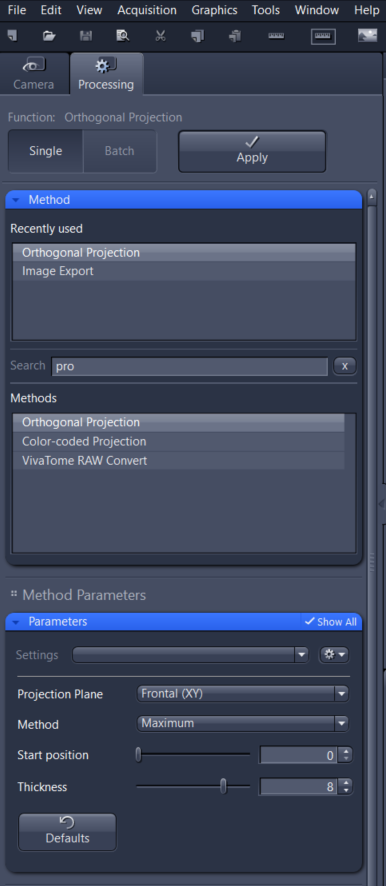
**USAGE**

**NOTE!** For convenience, all microscope image files “czi” can be moved into common folder, e.g. “Analysis” folder, and given common name that differs only by Base number, for example “Start Date of experiment\_Experiment Name\_Base 1”, “Start Date of Experiment\_Experiment Name\_Base 2” etc.

To create a maximum intensity projection of the images, follow the steps below:

1. Open “czi” image file in **ZEN** **BLUE (version 2.3)**. Start with “BASE1” FILE; continue with each base one by one.
2. In the DISPLAY tab choose BEST FIT▶Select DAPI CHANNEL▶ go through Image Z-STACKS one by one to identify layers that best show the object(s) of interest.
3. Repeat step 2 for all remaining channels one by one
4. In the PROCESSING tab (left side), within the METHOD window, choose ▶ORTHOGONAL PROJECTION
5. In PARAMETERS tab▶Set START POSITION and THICKNESS depending on number of Z-stacks to be projected▶APPLY

**OBS!** START POSITION 0 = Z-STACK1; THIKNESS indicates number of Z-stacks (including Z-stack start position).



**EXAMPLE:**

Total Z-stacks=12

Z-stacks in focus: 3 to 9

Put▶Start Position=2; Thickness=8.

## PART 2: Stitching (done in zen)

**DEFINITION**

The images generated from the tile scan are not automatically stitched together. To stitch the tiled image, the stitch function under the processing tab must be used. Once the image to be stitched is selected, a threshold for stitching must be set. This threshold parameter can be thought of as a metric of how strict the software will be when trying to match features across tiles. If the threshold is too strict/loose then sample features that span across tiles may not be reconstructed correctly.

**USAGE**

Stitching is done just after the last step of the previous section.

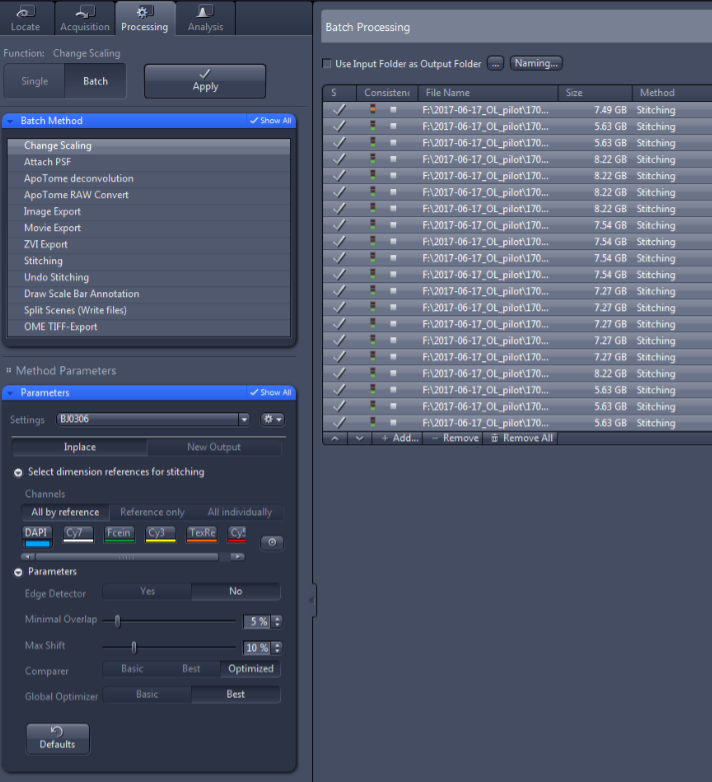
1. Optional▶SAVE MIP FILE or proceed▶STITCHNING

2. In TOOLS tab▶MODULE MANAGER▶ Activate PANORAMA

3. In PROCCESSING tab▶under METHOD window▶select STITCHING

4. In PARAMETERS tab▶Select DIMENTION REFERENCE for STITCHING▶ DAPI CHANNEL

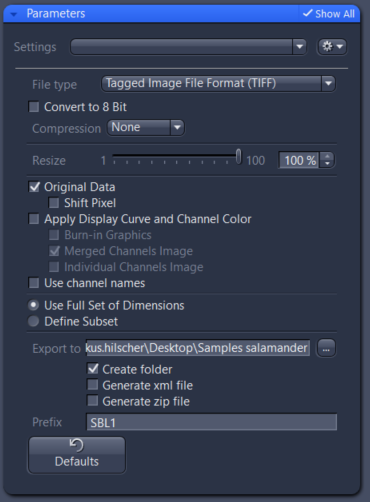
5. In PARAMETERS tab▶Set MINIMAL OVERLAP=5%▶MAXIMUM SHIFT=10%▶APPLY



After Stitching is done, files are exported into a .TIF file.

For this:

In PROCCESSING tab▶IMAGE EXPORT▶Set PARAMETERS▶TIFF, COMPRESSION NONE, ORIGINAL DATA▶Uncheck MERGED CHANNELS IMAGE, ▶Uncheck “CONVERT TO 8Bit” ▶EXPORT TO – select folder



## PART 3. Automatic Image Registration/alignment with MATLAB.

**DEFINITION**

Image registration, also known as alignment, is the process of aligning images from two or more data sets. Determining an effective image registration approach is situation-dependent. It requires careful selection of a point transformation model to provide reference points between the images, and a method for comparing information to identify the parameters required to properly align the images. Therefore, it’s a KEY step in the process. We will define two images in the comparisons:

**Reference (fixed) image:** Image in the target orientation

**Target/ float (moving) image:** Image to be transformed into alignment with the reference

There are two main ways to align images (USE ONLY ONE):

* **MANUAL ALIGNMENT**: it’s trial and error. You adjust the transformation by trying combinations and observing the result. Time consuming.
* **AUTOMATIC ALIGNMENT**: utilizes MATLAB script (by Erik Samuelsson), developed to achieve the best alignment possible via similarity metrics measurement and optimization. Total running script time – about 30 min for all the 4 bases, if images sizes do not exceed 0,5GB.

**MANUAL ALIGNMENT USAGE**

To start images alignment – move exported files into one folder

Check for the image with the best signals (most anchor stain, e.g. b2.)

**OBS!** typically the base sequenced 2nd in the row, gives best signal

Check the “sequence” of movement and align the images to the middle one (e.g. b2, to avoid extensive calculations)

Create a folder “Aligned” in the “Exported” folder, copy b2 file into there and align the images of b1, b3, b4 and b5 to the reference images of b2.

Run script (NAME) and save it for each base separately.

***IN MATLAB:***

*REF\_IMAGE=specify the Reference Image path-c1(channel1)\_DAPI image of the reference base*

*INPUT\_IMAGE\_PREFIX=specify the path to the target image (e.g. “b2-mip\_c”)*

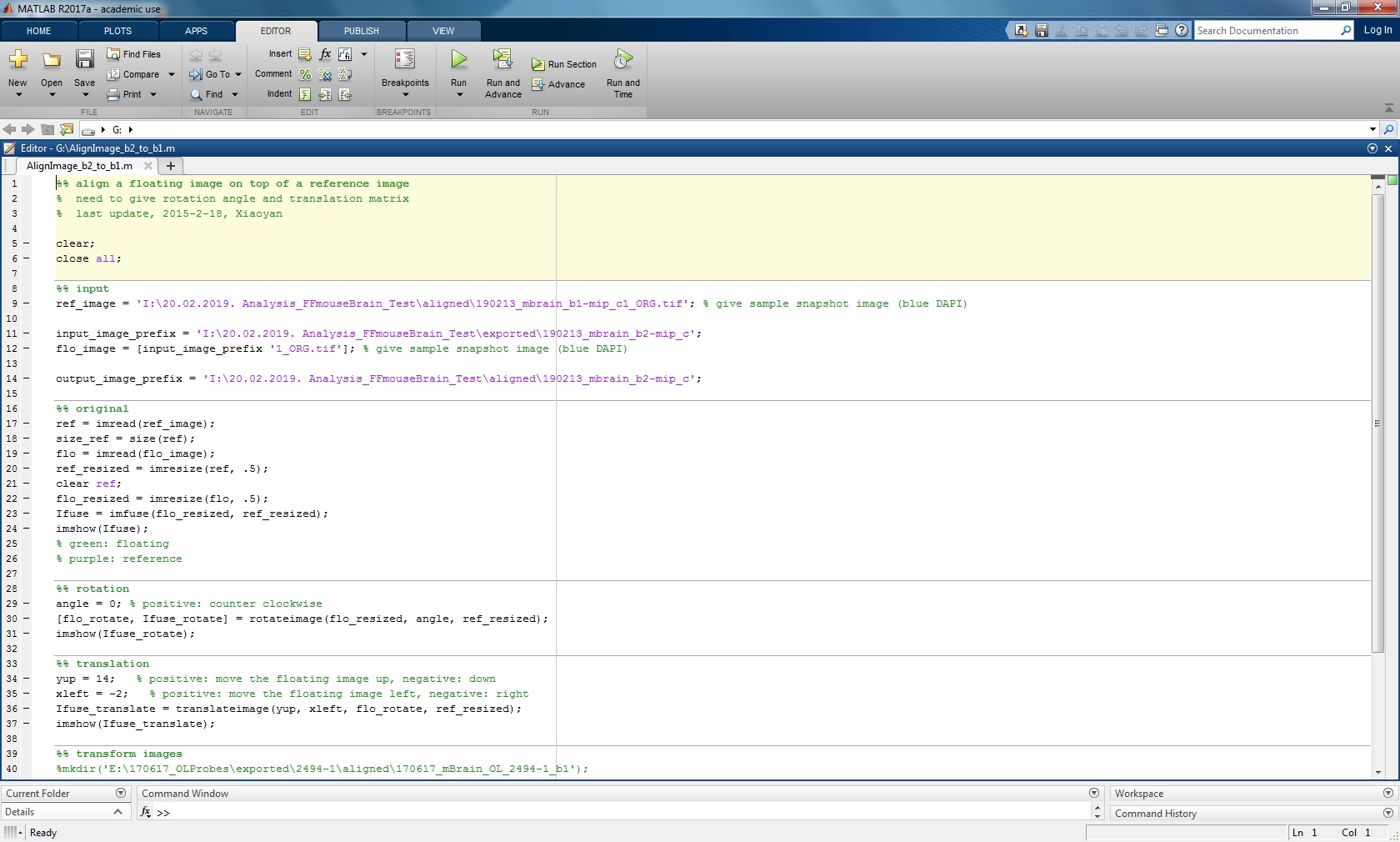
*FLO\_IMAGE=[input\_image\_prefix ‘1\_ORG.tif’]*

*OUTPUT\_IMAGE\_PREFIX▶ “Aligned” folder path where aligned images will be saved*

*In ROTATION▶ANGLE=0 (positive: counter clockwise)*

*In TRANSLATION▶Y(up)=pixels number (positive: move the floating image up, negative: down)*

*In TRANSLATION ▶Xleft=number of pixels (positive: move the floating image left, negative: right*

**

**AUTOMATIC ALIGNMENT USAGE**

**Problem Statement**

One issue for the molecular profiling section of the project lies in the generation of the gene expression maps. Due to the handling of the tissue sections during imaging on the microscope, the samples themselves can be translated, rotated or compressed, leading to slightly different images that then require a registration step to align the base images to a template image. Currently, the respective base images are aligned manually, via entering the translation and rotation variables manually in the script, assessing the alignment by eye, and repeating until a satisfactory alignment is achieved can take up to two hours of time.

Since the problem of image alignment cannot be solved during the image collection phase (i.e., collecting all base images in one run without jostling the samples), it must be solved during the pre-processing step before the images are run through the gene mapping software.

**Problem Characterization**

The fixed tissue images used in this workflow all benefit from being only slightly different from one another in terms of deformation effects. Any deformation effects are minimized in terms of there being no stain variations, missing tissues or differing data preparation processes. The only deformation effects of relevance to this specific issue are rotation, translation, and morphological deformations (due to compression of the tissue slides during imaging).

Possible deformation effects can be caused in individual data preparation processes, involving morphological deformations, stain variations, stain artifacts, rotation, translation, and missing tissues. The combining deformation effects tend to make existing automatic registration methods perform poorly.

**How it works *IN MATLAB:***

1. Input image locations and details via user prompt.

a. Note: All .tiff files must be named in the following manner, otherwise the script will not work:

Base #\_c#\_ORG.tiff

2. Select reference image based on Shannon entropy (see Automatic Selection of Reference Images)

3. Register each floating base DAPI channel to the reference image using the imregister function in Matlab

a. Optimizer and Metric: Since the images analyzed have different intensity distributions and are collected from multiple channels, a multimodal configuration is selected for the image registration.

b. Metric: In order to register the image, a random sampling of 500 pixels is taken from the reference image and compared to the floating image

4. Apply the transformation matrix from registration to all channels of the floating Base and save to a folder ‘Aligned\_Images\_Rigid’

Outputs:

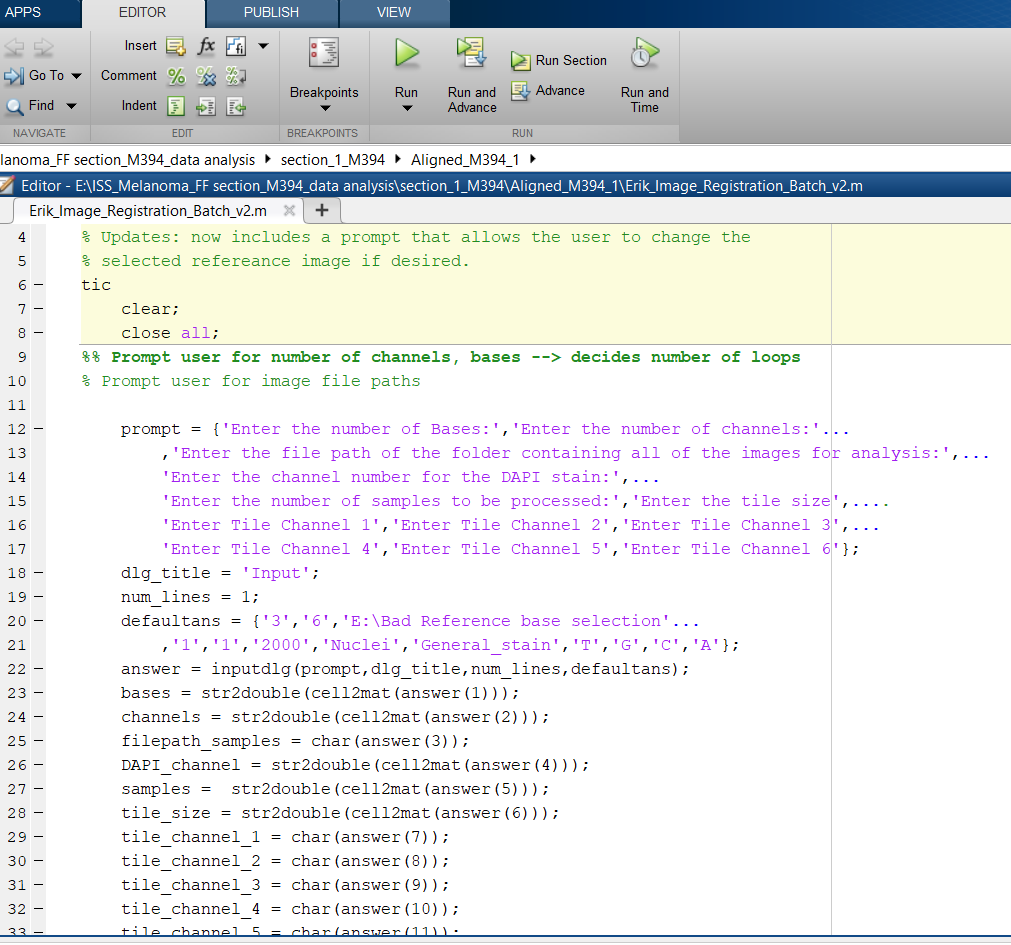
Variables:

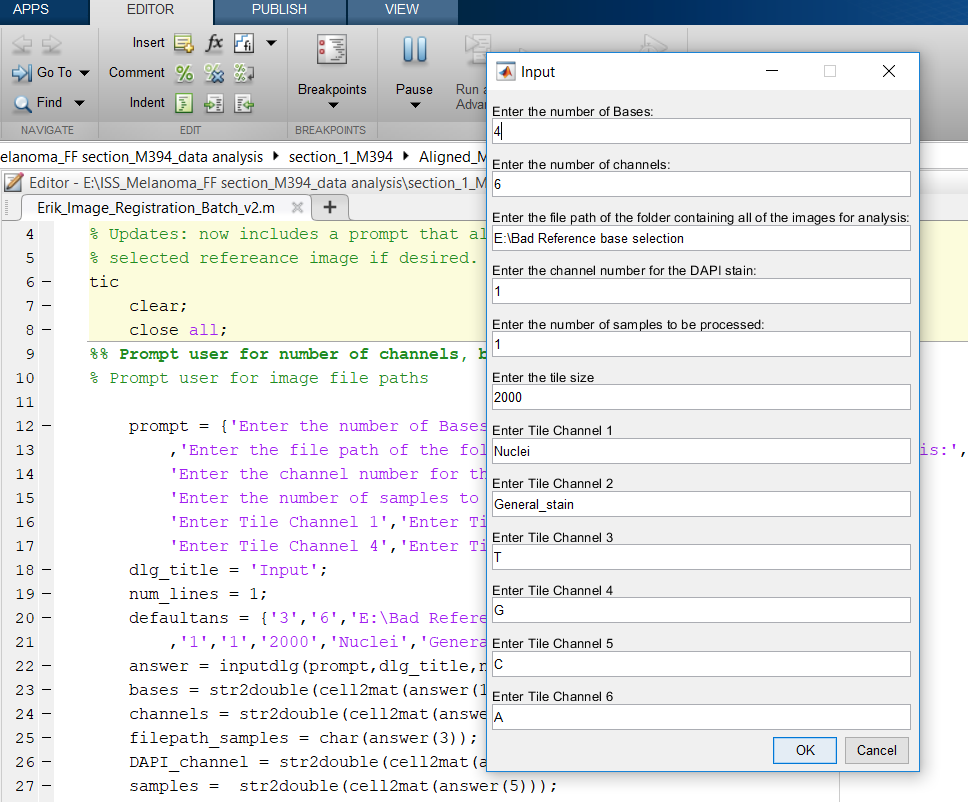
· R\_before:

· R\_after:

· tRegtform:

Images: Base #\_c#\_ORG.tiff





## PART 4. Image Tiling with MATLAB

**DEFINITION**

Tiling lets displaying of images that are too large to be read entirely into memory, by

segmenting them into smaller, more manageable image tiles.

**USAGE MANUAL ALIGNMENT**

Use MATLAB script “Sequencing”

1. Choose functions to run▶RUN\_TILING\_YN = 1
2. Choose functions to run▶ DECODE, THRESHOLD,PLOTTING=0
3. T.FOLDER\_IMAGE=full path name to Aligned images folder
4. T.FILENAME\_BASE\_PREFIX=common file name part, e.g. ‘date\_experiment\_b’
5. T.IN\_SUBFOLDER\_YN=0
6. T.FILENAME\_CHANNEL\_PREFIX=’-MIP\_C’
7. T.FILENAME\_SUFFIX=’\_ORG.TIF’
8. T.BASE\_START=1▶ T.BASE\_END=4
9. T.CHANNEL\_START=1▶ T.CHANNEL\_END=6
10. T.TILE\_SIZE=2000
11. T.CHANNEL\_ORDER= {‘Nuclei’ ‘General\_stain’ ‘T’ ‘G’ ‘C’ ‘A’}
12. T.CSV\_FILENAME\_PREFIX=’Tiled’
13. RUN▶ the images will be created in the “Aligned” folder

Ouput: Tiled\_SBL1\_c1\_ORG including tile1.tif – tile2.tif & Tiled.csv

e.g. 2000 x 2000 pixels (3 px ~ 1 um with 20x objective)

Tiled.csv is further used for Cell Profiler analysis, which generates a folder cpresults with Alignment folder, blobs.csv, Experiment.csv, hybs.csv, Image.csv, preblobs.csv, prehybs.csv

## PART 5. CellProfiler Image Analysis. (Blob identification)

**DEFINITION**

Cell profiler’s main pipeline include several processing steps, with different importance:

**-Enhance or supress features**: enhances or suppresses certain image features (such as speckles, ring shapes, and neurites), which can improve subsequent identification of objects. This module enhances or suppresses the intensity of certain pixels relative to the rest of the image, by applying image processing filters to the image. One step is done for each single channel

-**Identify primary objects:** find the blobs, depending on the specified parameters. This is one of the most important steps in the process, as depending on the detected blobs, the specificity will vary. Two steps are done: one for general stain (anchor) and another for all the bases (ATCG) together

- **ExpandOrShrinkObjects** expands or shrinks objects by a defined distance. The module expands or shrinks objects by adding or removing border pixels. You can specify a certain number of border pixels to be added or removed, expand objects until they are almost touching, or shrink objects down to a point. The module can also separate touching objects without otherwise shrinking them, and can perform some specialized morphological operations that remove pixels without completely removing an object. Two steps are done: one for general stain (anchor) and another for all the bases (ATCG) together

**-Gray to color**: adds color

- **Measure object intensity**

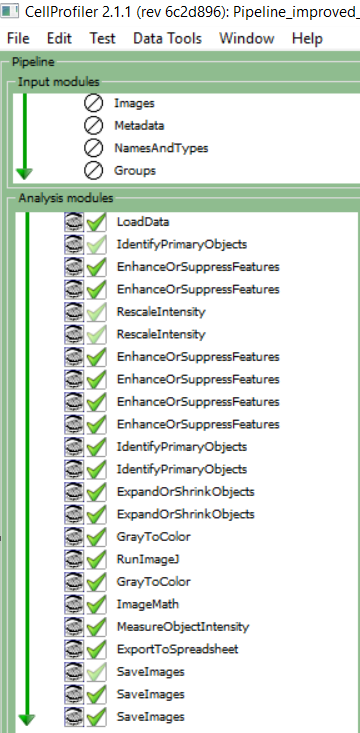
**- Export:** two exporting steps are done: export to spreadsheet and SaveImages

**USAGE**

CellProfiler requires 3 plugins:

1. MultiStackReg1.45\_.jar
2. StackReg\_.jar
3. TurboReg\_.jar

Plug-Ins can be found in the C:\Users\Public\imagej folder



hybs

Enhanced\_C

Enhanced\_GeneralBlob

Enhanced\_SpecBlob

Enhanced\_A

Enhanced\_G

Enhanced\_T

preblobs

prehybs

blobs

Tiled.csv

blobs\_prealign

blobs\_align

AlignmentMath

blobs.csv

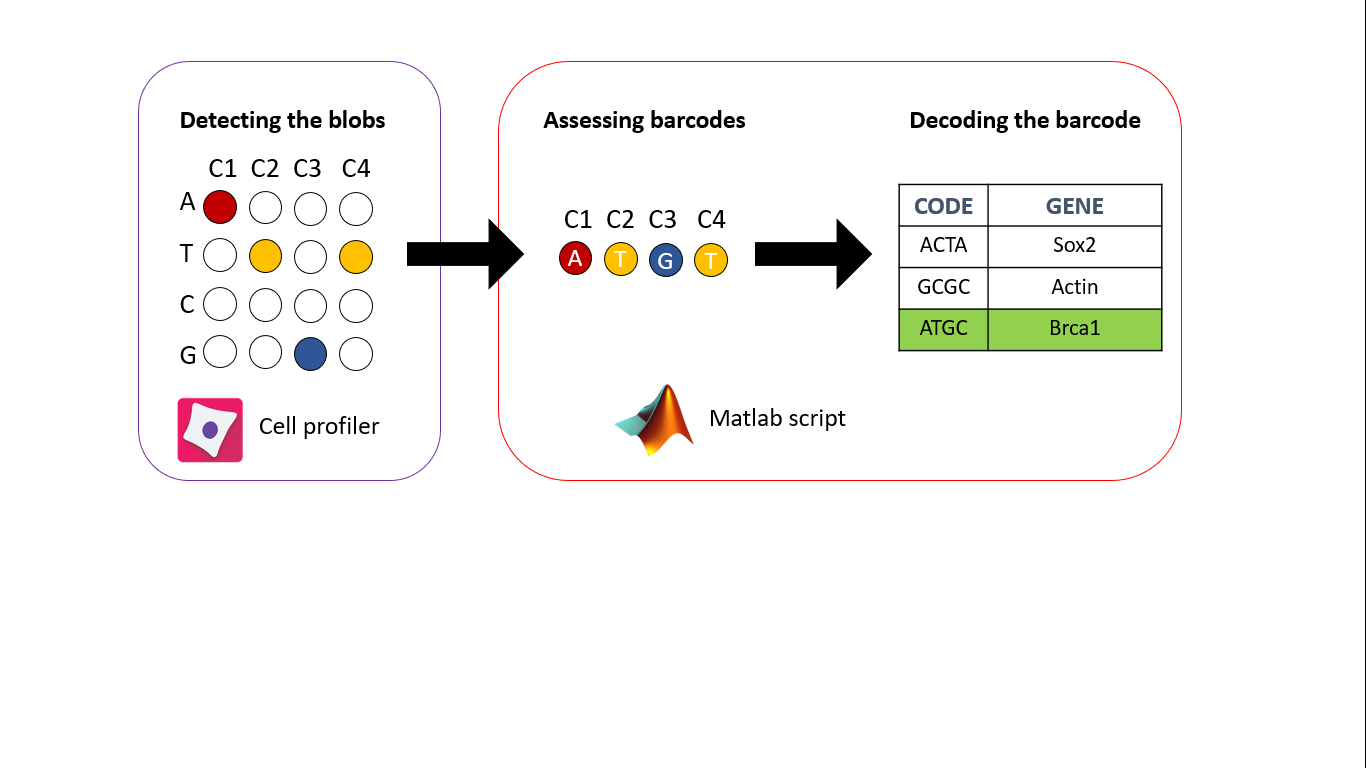
1. Set Yes to Process just a range of rows (e.g. 400 to 600)
2. Click to open “eye symbol” on EnhanceOrSupressFeatures for generalblob and specblob
3. Click to open “eye symbol” on IdentifyPrimaryObjects for generalblob and specblob
4. Click “Start test Mode”
5. Click “Step”
6. Set FeatureSize in EnhanceOrSupressFeatures
7. Set Manual threshold in IdentifyPrimaryObjects
8. When you are done close the 4 “eye symbols” on EnhanceOrSupressFeatures and IdentifyPrimaryObjects
9. Set No to Process just a range of rows
10. Click Analyze Images

CellProfiler results are summarized in output file called BLOBS.CSV (blobs combined).

## PART 6. Sequence Decoding – MATLAB

**DEFINITION**

**Sequence Decoding** decodes the different blobs found in the different channels and cycles into the presence of a specific gene. Schematic explanations is presented below:



**USAGE**

After getting CellProfiler results, the output file BLOBS.CSV is processed in MATLAB using “**Decoding Sequence**” function.

1. Choose functions to run▶RUN\_DECODE\_YN=1
2. D.INPUT\_FILE = File path to BLOBS.CSV
3. D.GENERAL\_ALIGNMENT\_ACGT\_COLUMN\_NUMBER=[0,0,6,7,8,9]
4. D.XYPOSITION\_PARENTCELL\_COLUMN\_NUMBER= [10,11,0]
5. D.NUM\_BYBS=4
6. D.TAGLIST=path to Taglist.CSV file (file with columns: code, name, symbol (optional), no header)
7. D.CSV\_FILE\_CONTAIN\_TILE\_POSITION=path to “TILED.CSV” file
8. D.OUTPUT\_DIRECTORY=path to ’Decoding’ folder
9. In OPTIONS▶d.check\_parent\_cell\_YN=0; d.check\_alignment\_YN=0; alignment\_min\_threshold=1.8; d.abnormal\_sequencing\_YN=1; d.sequencing\_order=’1234’

Output:

Decoding folder with beforeQT.mat

beforeQT\_code\_n\_count.csv

beforeQT\_details.csv

beforeQT\_gene\_n\_count.csv

beforeQT\_qualitybar.csv

**MATLAB SYMBOLS**

[**https://se.mathworks.com/help/matlab/ref/linespec.html**](https://se.mathworks.com/help/matlab/ref/linespec.html)

### Marker Specifiers

|  |  |
| --- | --- |
| Specifier | **Marker Type** |
| '+'  'o'  '\*'  '.'  'x'  'square' or 's'  'diamond' or 'd'  '^'  'v'  '>'  '<'  'pentagram' or 'p'  'hexagram' or 'h' | Plus sign  Circle  Asterisk  Point  Cross  Square  Diamond  Upward-pointing triangle  Downward-pointing triangle  Right-pointing triangle  Left-pointing triangle  Five-pointed star (pentagram)  Six-pointed star (hexagram) |

### Color Specifiers

|  |  |
| --- | --- |
| Specifier | **Color** |
| r  g  b  c  m  y  k  w | Red  Green  Blue  Cyan  Magneta  Yellow  Black  White |

## PART 7. Threshold Sequencing– MATLAB

**DEFINITION**

Threshold sequencing is a Quality control step, where data is filtered depending on the Quality threshold and general stain threshold.

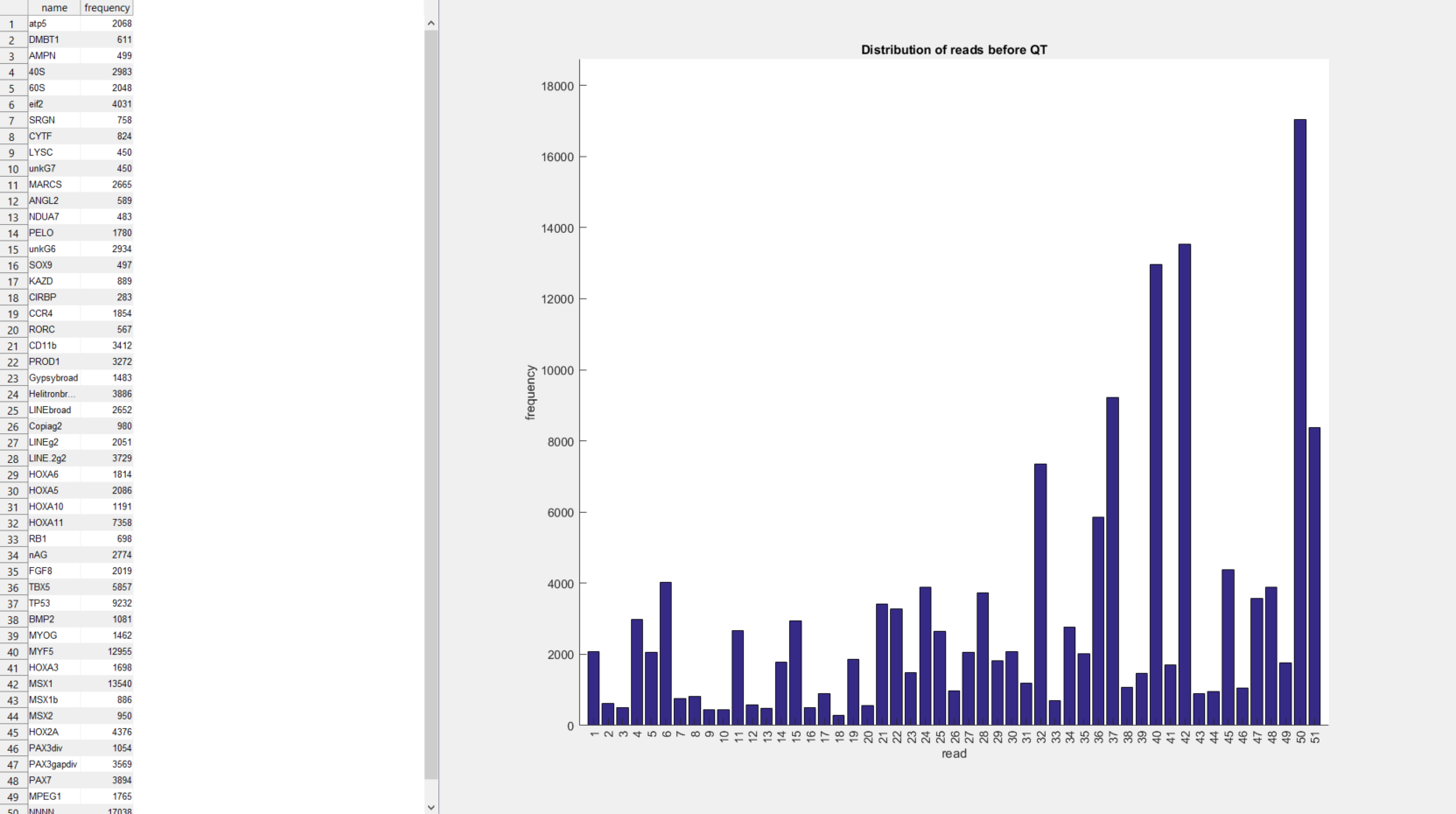
**-Quality threshold:**

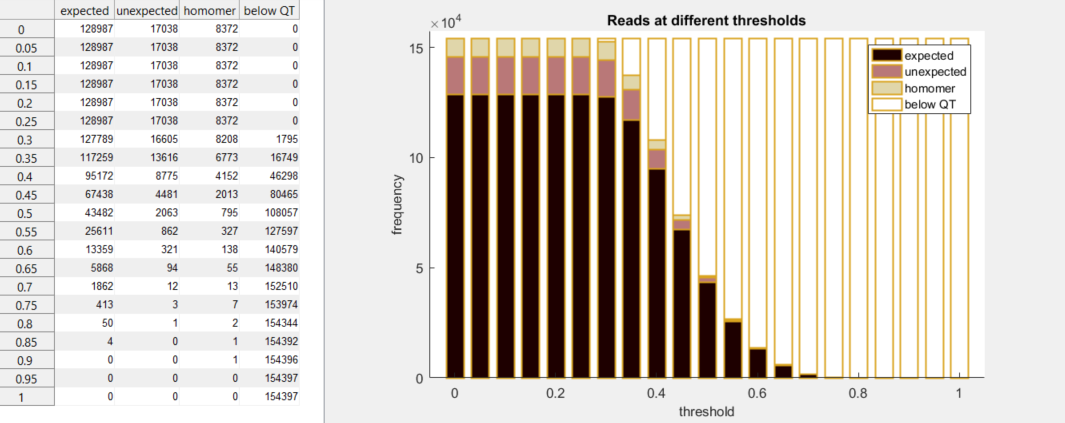
**-General\_strain\_threshold:**

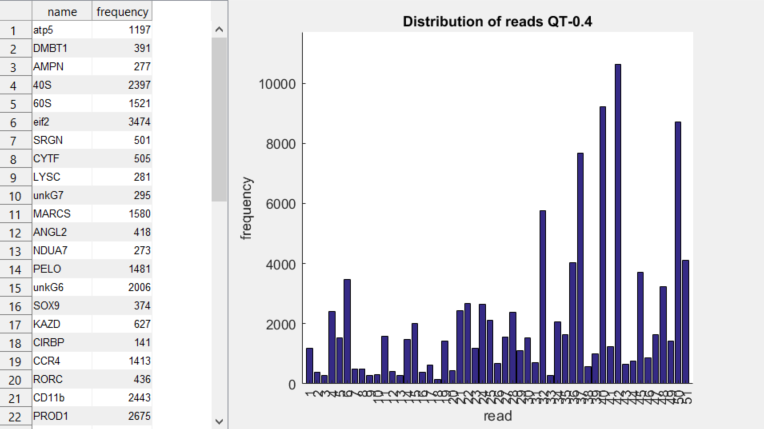
**USAGE**

1. Choose functions to run▶RUN\_THRESHOLD\_YN=1
2. Q.quality\_threshold=0.35
3. Q,general\_stain\_threshold=0.00000001

* Output: QT\_0.4\_0.001.mat,
* QT\_0.4\_0.001\_code\_n\_count.csv,
* QT\_0.4\_0.001\_details.csv,
* QT\_0.4\_0.001\_gene\_n\_count.csv







## PART 8. Plotting Global Sequencing – MATLAB

To create Plots Background Image▶**Open DAPI Reference Base Image, Stitched and Projected MIP** file in ZEN BLUE:

1. In PROCCESSING tab▶IMAGE EXPORT▶Set PARAMETERS▶JPEG, COMPRESSION=20%, ORIGINAL DATA▶Uncheck MERGED CHANNELS IMAGE, ▶Uncheck “CONVERT TO 8Bit” ▶EXPORT TO – Create Background folder
2. Write down the image compression size or incorporate the into file name – will be further needed in p.scale MATLAB parameter

**In MATLAB**

1. Choose functions to run▶RUN\_PLOTTING\_GLOBAL\_YN=1
2. P.BACKGROUND\_IMAGE=path to Background image file
3. P.SCALE=0.2 (if background image was compressed to 20% )
4. P.EXCLUDE\_NNNN\_YN=1

