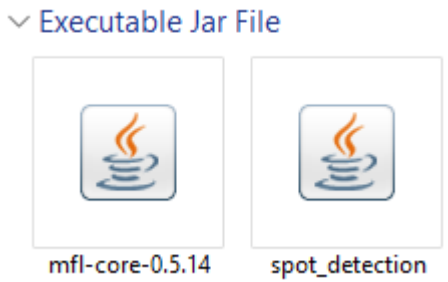
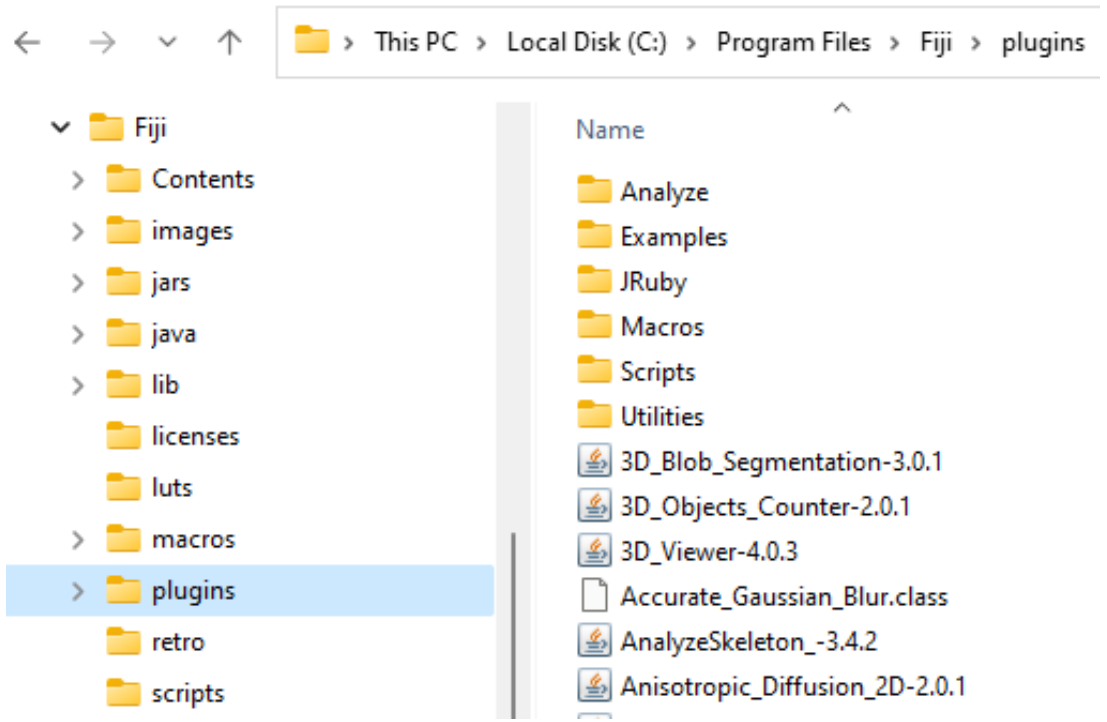
# Spot detection plugin installation

- Install Fiji or ImageJ (version 1.53 or higher) + Bio-Formats plugin
- Add these 2 files in the “plugins” folder:

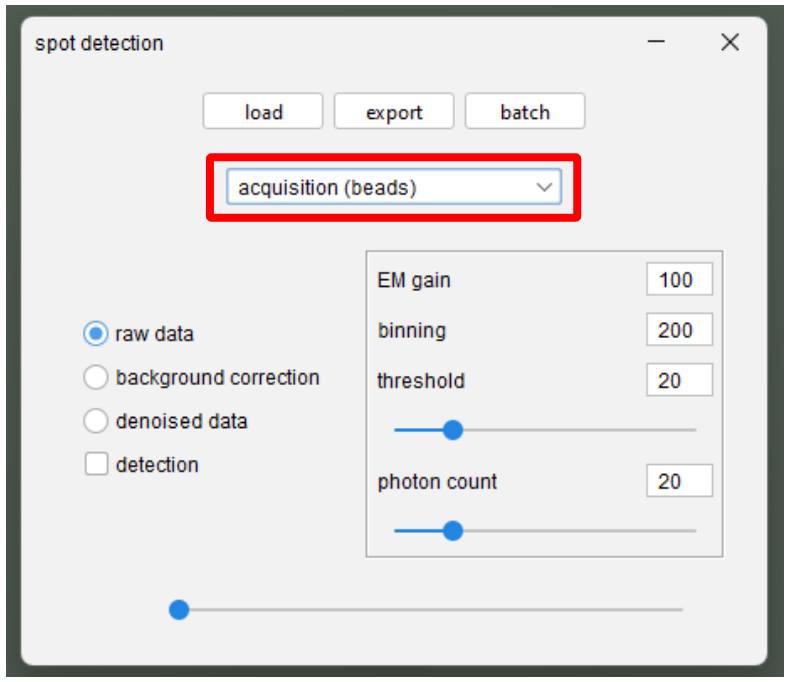



to export MATLAB cells

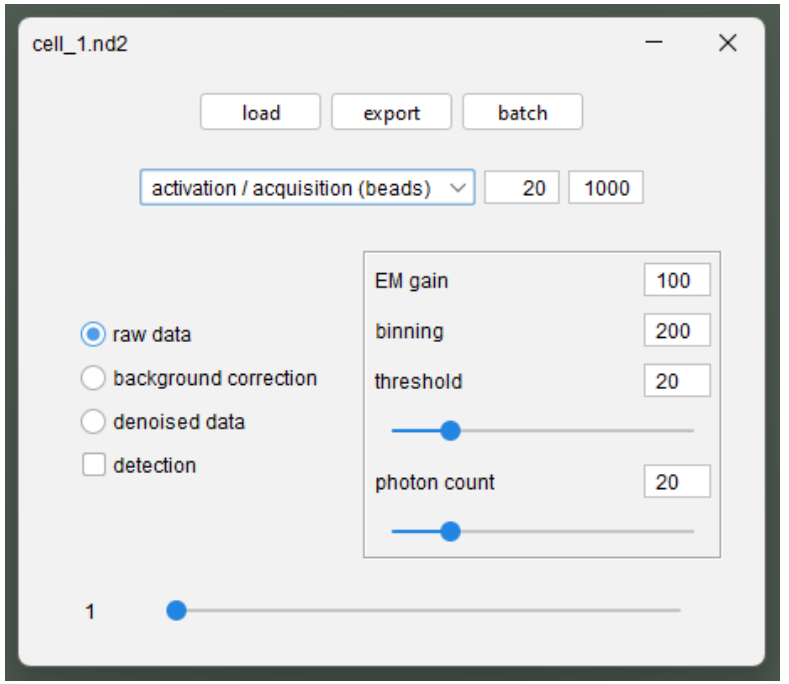
home-made plugin

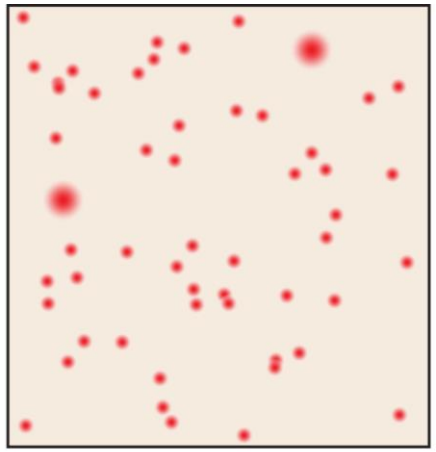


## Simple stack acquisition

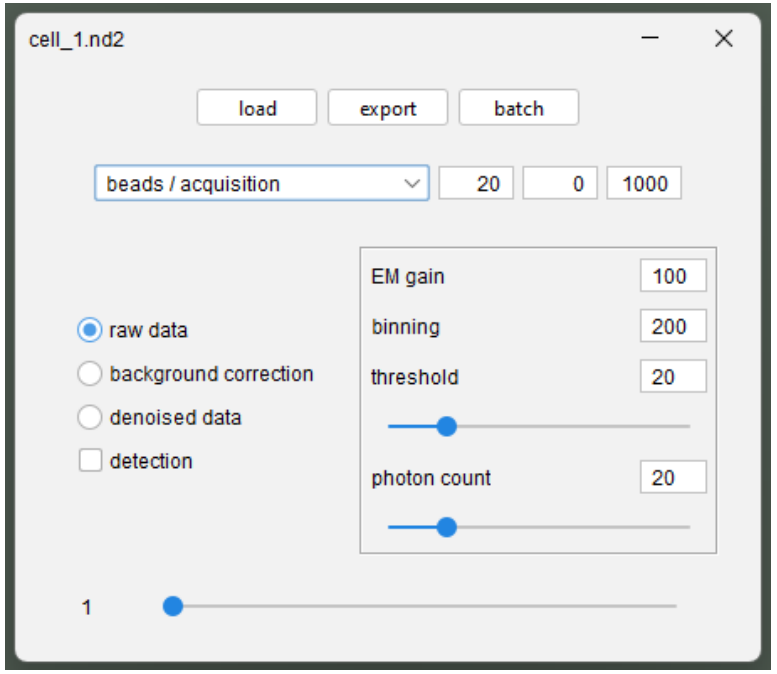


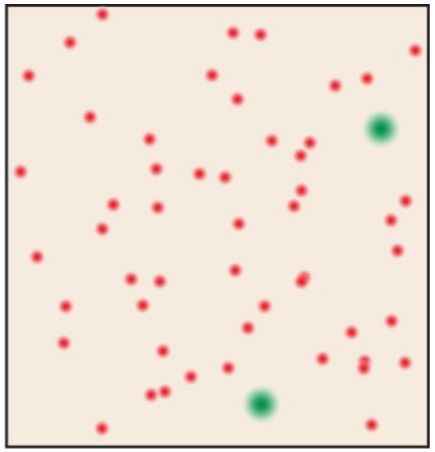

## Activation + Stack acquisition



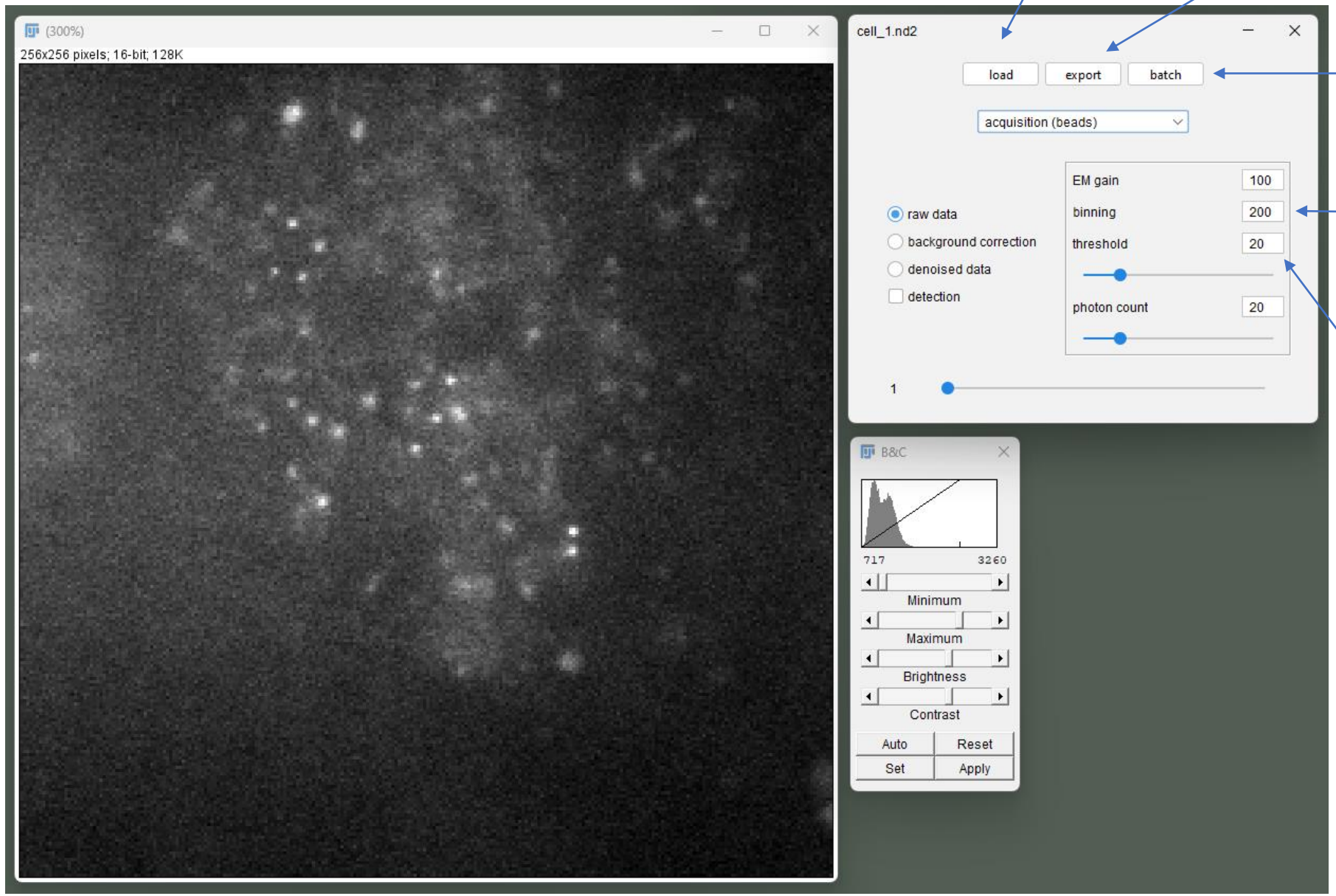



## Beads + Activation + Stack acquisition



Example:
D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\raw_data_example\cell.nd2




Load nd2 stack

Export coordinates

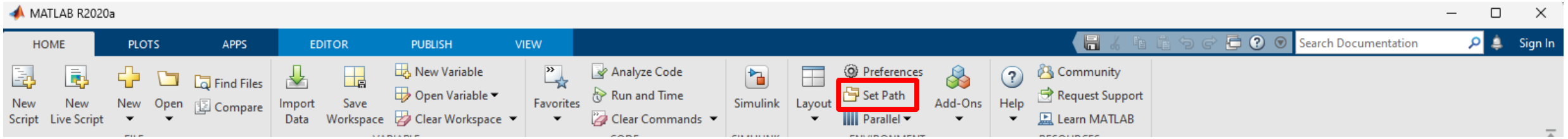
Load a list of stacks and export coordinates for each stack

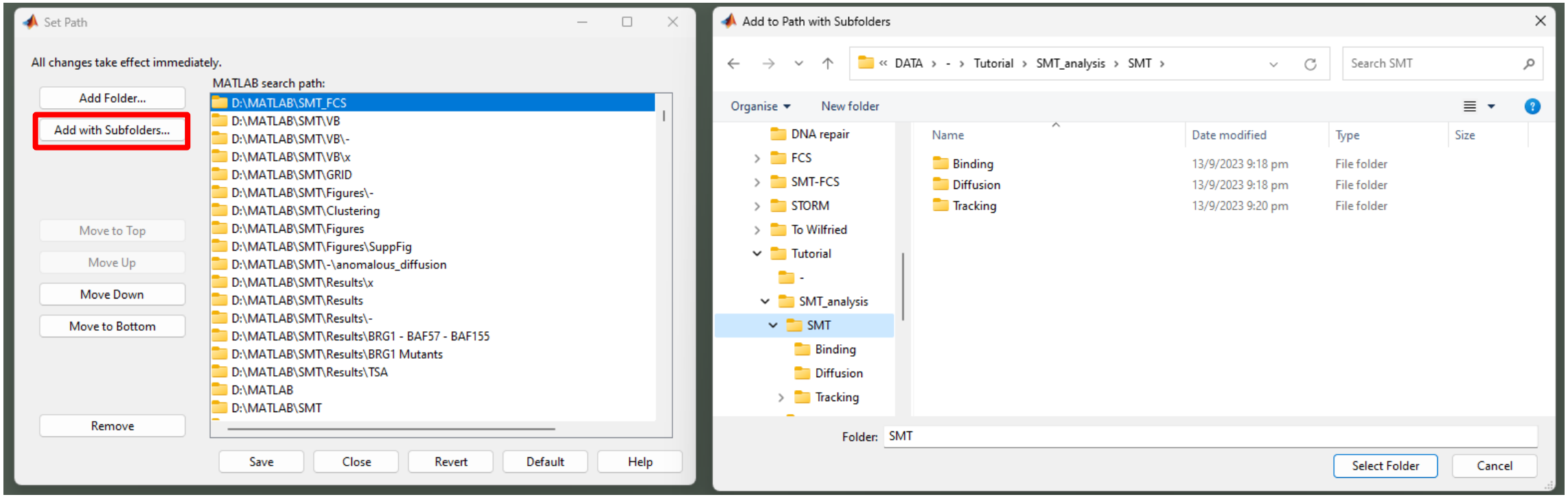
Background removal

For 1000 frames, the program is calculating the background 5 times with 200 frames each time.

Threshold to isolate individual spots

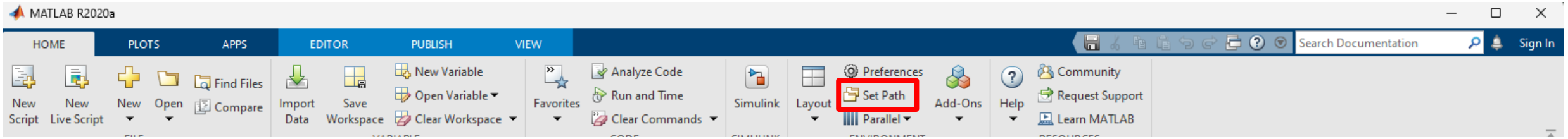
# Matlab scripts setup




Add all folders from: D:\Nat Comm codes\SMT_analysis\

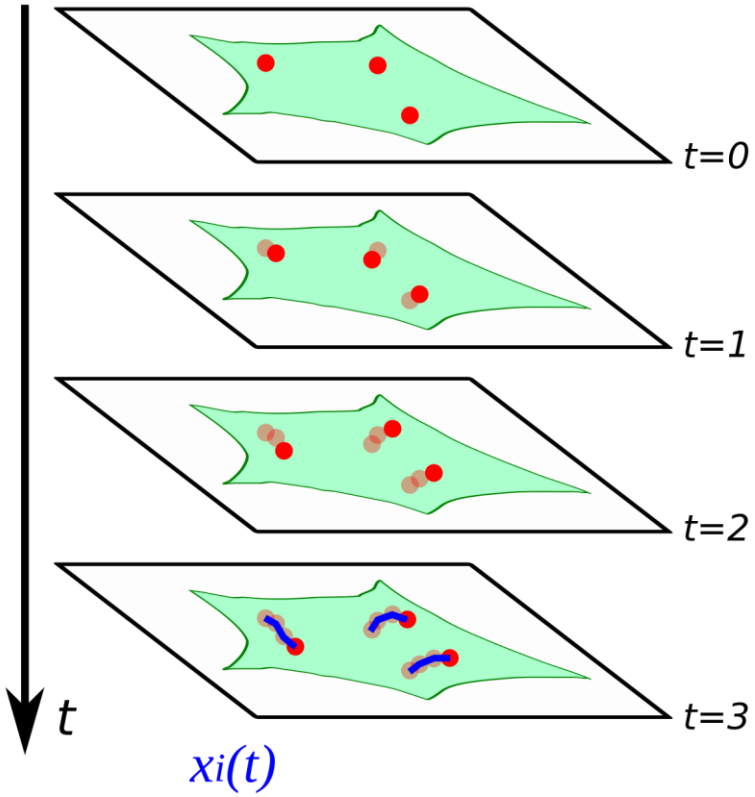
# Matlab scripts setup

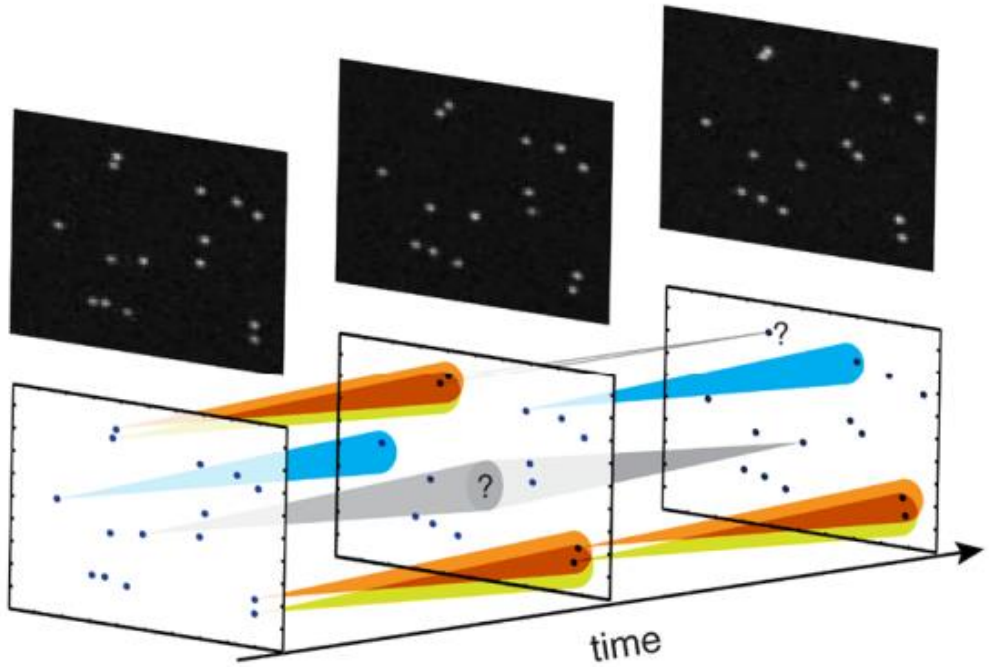



Toolboxes required:

- Image Processing Toolbox
- Curve Fitting Toolbox

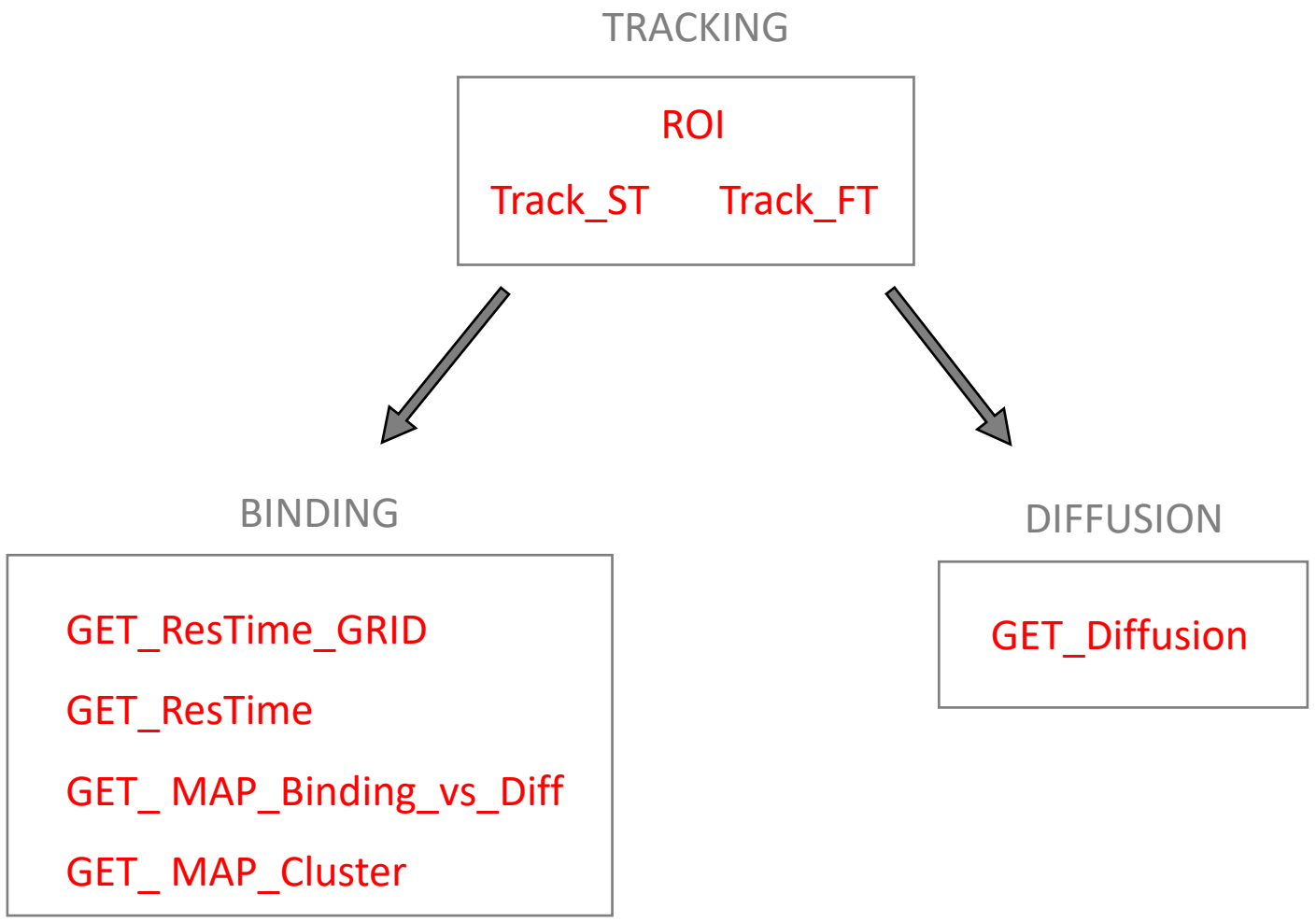
# Single-molecule tracking




mem = 2 max. number of lost frames
dmax = 1.75 max. distance to connect dots (in px)
Nmin = 2 min. number of jumps

# Matlab scripts




- Images of the nucleus are saved as:
  *cell_1.tif, cell_2.tif,…*
- ROIs of the nucleus are saved as:
  *cell_1_roi.tif, cell_2_roi.tif,…*
- Coordinates are saved as:
  *cell_1.mat, cell_2.mat,…*
- Trajectories are saved as:
  *cell_1_traj.mat, cell_2_traj.mat,…*
  (fast and slow tracking)
- Binding trajectories are saved as:
  *cell_1_bind.mat, cell_2_bind.mat,…*
  (slow tracking)

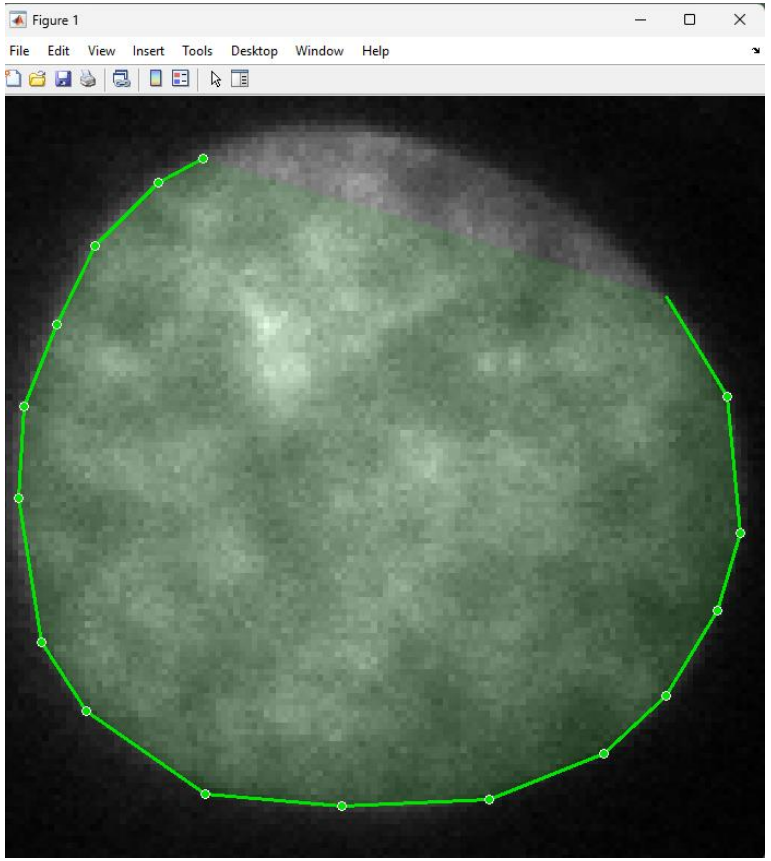
- Full trajectories and binding trajectories are located in the main folder
- Raw data and images of the ROIs are located in a subfolder (“…\main folder\-\”)

- All scripts have been tested on Matlab R2020a / R2021a / R2022a
- Calculation times are obtained from an Intel Core i5, 2.60 GHz and 32GB of RAM

# ROI

## Define an ROI based on the nucleus boundary

```
clc;
clearvars;
close all;

path0 = 'D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\fast_tracking\';

u = 7;      %%%   cell #
w = 128;    %%%   image size (px)


file = ['cell_' num2str(u)];
im = imread([path0 '-\' file '.tif']);

f = figure(1);
set(f, 'position', [500 100 700 700])
set(gca, 'position', [0 0 1 1])
imagesc(im)
axis equal off
hold on
colormap gray

roi = images.roi.Polygon(gca, 'color', [0 .9 0], 'visible', 'on', 'facealpha', .1);
draw(roi);
in = createMask(roi, w, w);
in = 1.*(in>0);

imwrite(in, [path0 '-\' file '_roi.tif'])
```



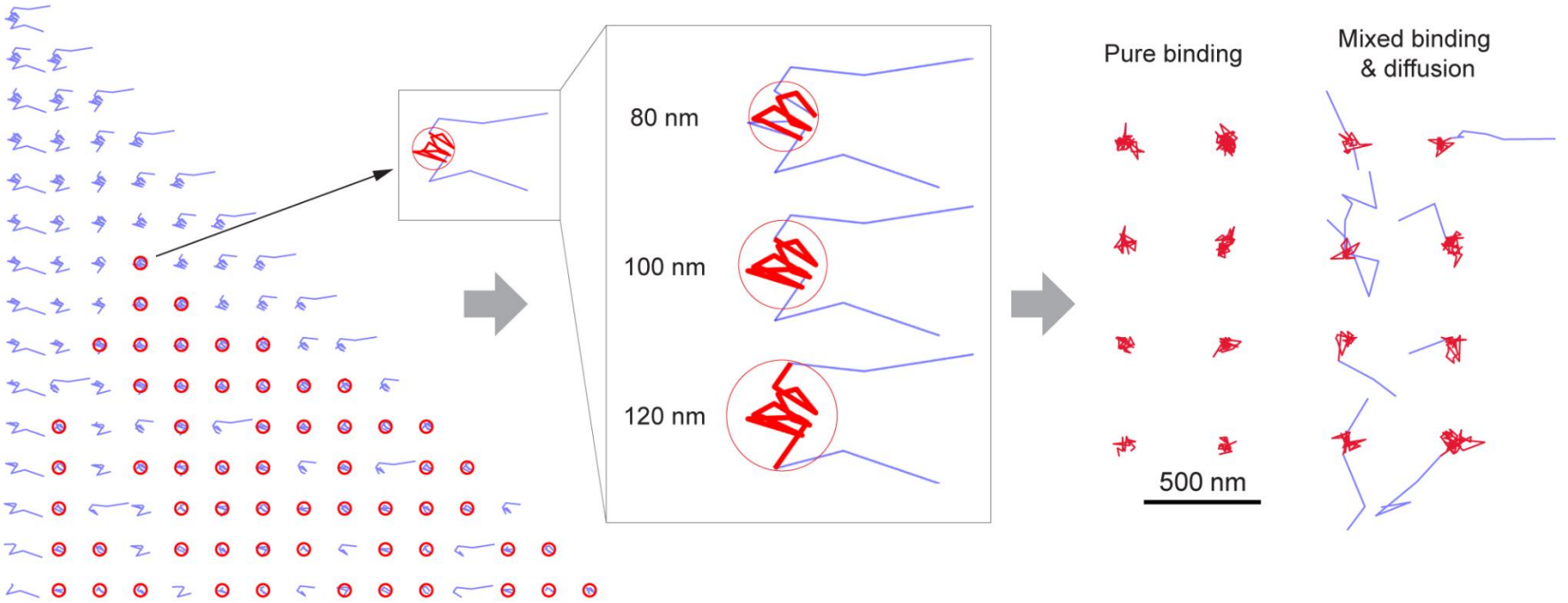
# Track_FT

Run fast-tracking

```
clc;
clearvars;
close all;

path0 = 'D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\fast_tracking\';

p.mem = 2;      %%%   max. number of lost frames
p.dmax = 4;     %%%   max. distance to connect dots between 2 consecutive frames (px)
p.Nmin = 3;     %%%   min. number of jumps


for u = 1:10

    disp(['cell_' num2str(u)]);

    load([path0 '-\cell_' num2str(u) '.mat'])
    r = imread([path0 '-\cell_' num2str(u) '_roi.tif']);
    T = fct_tracking(X, p, r);

    save([path0 'cell_'  num2str(u) '_traj.mat'], 'T')

end
```

*10 seconds in average per cell*

# Discriminating binding from diffusion in slow-tracking




All possible track segments of a given trajectory

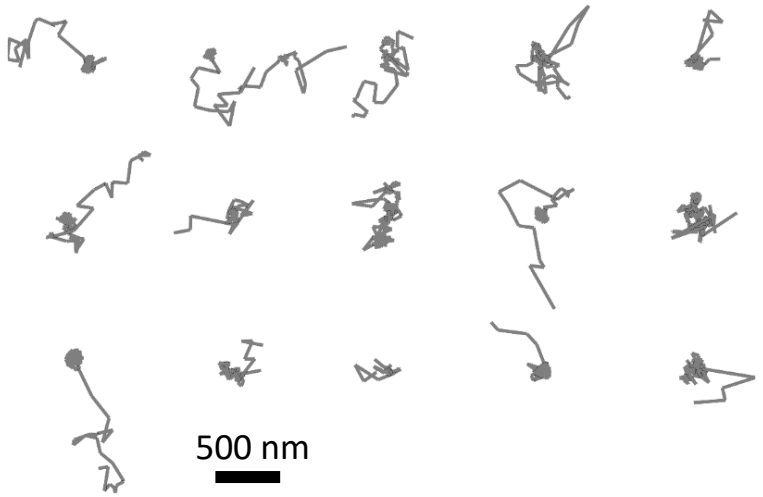
# Track_ST

Run slow-tracking and isolate binding events from full trajectories

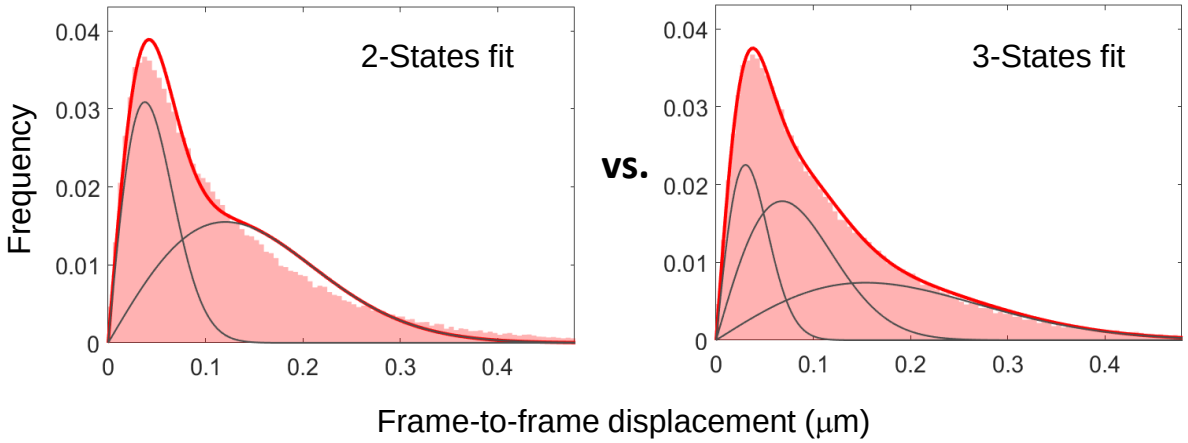
```
clc;
clearvars;
close all;

path0 = 'D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\slow_tracking\';

p.mem = 2;          %%%   max. number of lost frames
p.dmax = 1.75;      %%%   max. distance to connect dots between 2 consecutive frames (px)
p.Nmin = 2;         %%%   min. number of jumps

b.Jmin = 4;         %%%   min. number of jumps in the binding trajectory
b.px = 160;         %%%   pixel size (nm)
b.diameter = 100;   %%%   max. diameter of the circle that circumscribes the binding trajectory (nm)


for u = 1:8

    disp(['cell_' num2str(u)]);

    load([path0 '-\cell_' num2str(u) '.mat'])
    r = imread([path0 '-\cell_' num2str(u) '_roi.tif']);
    T = fct_tracking(X, p, r);

    save([path0 'cell_'  num2str(u) '_traj.mat'], 'T');

    [A, B] = Trim_Trajectories(T, b);
    T = A;
    save([path0 'cell_'  num2str(u) '_bind.mat'], 'T');
    T = B;
    save([path0 'cell_'  num2str(u) '_diff.mat'], 'T');

end
```

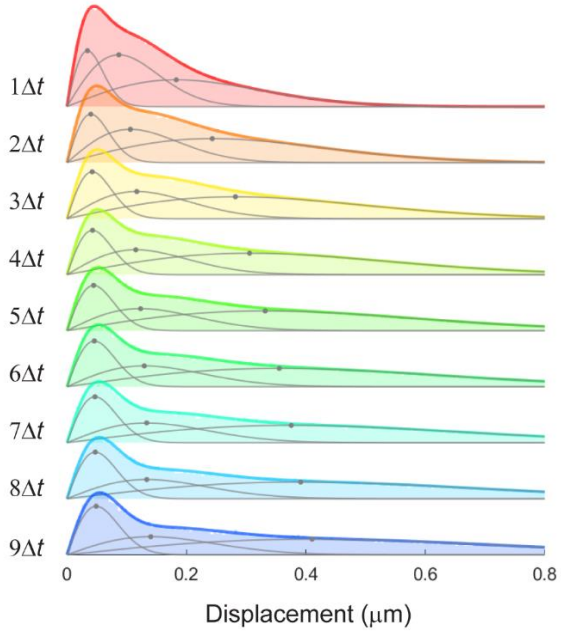
*16 seconds in average per cell*

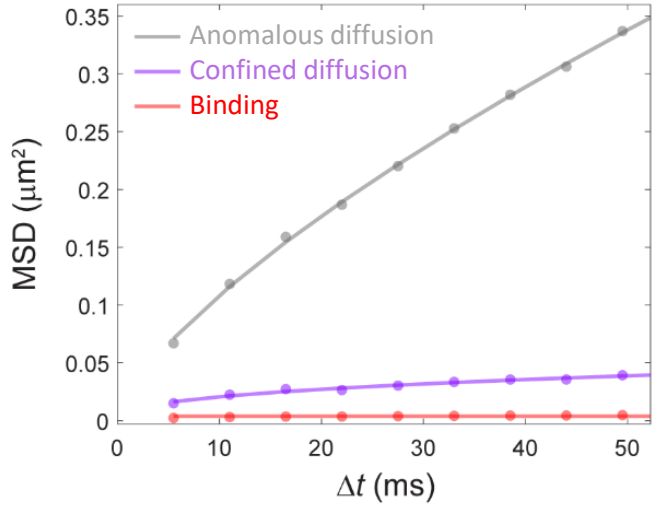
# Quantifying intranuclear diffusion in fast-tracking




**Kinetic modeling** (Hansen et al., *eLife* (2018))



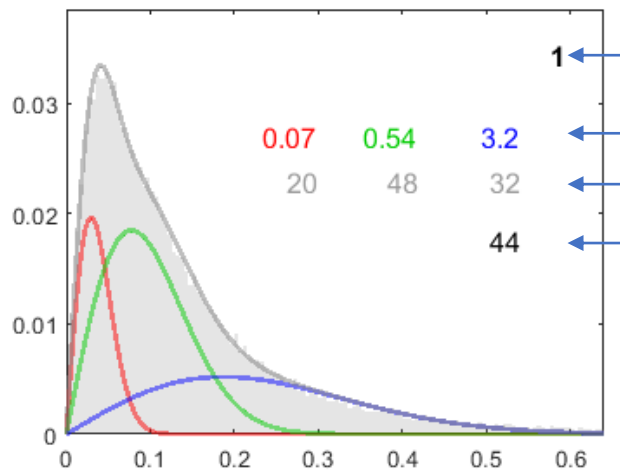



# GET_Diffusion

Get the 3 modes of diffusion based on the “Spot-on” model

```
path0 = 'D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\fast_tracking\';

global dr ds dt

px = .16;          %%%   pixel size (micron)
dt = .0055;        %%%   exposure time (s)
dr = .005;         %%%   distance for binning (micron)
ds = .010;         %%%   localizaion uncertainty (micron)
dmax = 4;          %%%   max. distance for the histogram (px)


p.start = [.05 .3 1 .4 .4];
p.LB = [.001 .01 .1 0 0];
p.UB = [.5 4 20 1 1];

X = [];
n = list_file(path0, '_traj.mat');
[Nx, Ny] = NxNy(n);

f = figure(1);
set(f, 'position', [300 530-190*Ny/2 300*Nx Ny*190+20], 'color', 'w');

for i = 1:n

    load([path0 'cell_' num2str(i) '_traj.mat']);
    jump = jump_distribution(T);
    x0 = mod(i-1,Nx);
    y0 = floor((i-1)/Nx);
    ax = axes('units', 'pixels', 'position', [55+x0*290 30+(Ny-1-y0)*185 250 175]);
    P = model_3_states(jump, dmax, p, px, i);
    X = [X; [i P]];

end

exportgraphics(f, [path0 'Diffusion.png'])
save([path0 'Diffusion.mat'], 'X')
```




cell #

diffusion coefficients ($\mu m^2/s$)

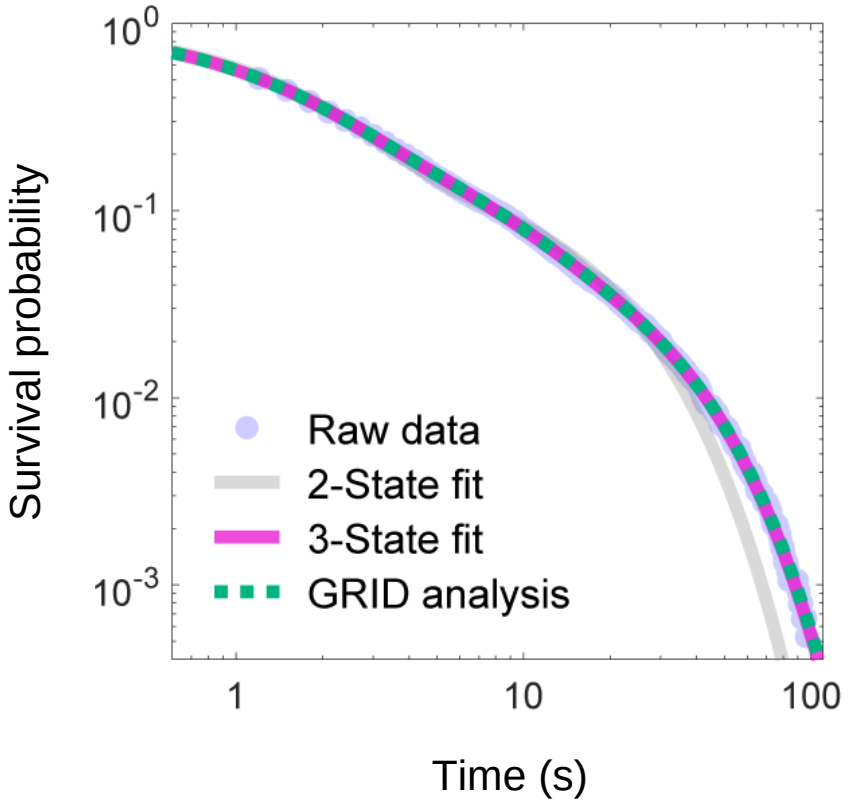
fractions

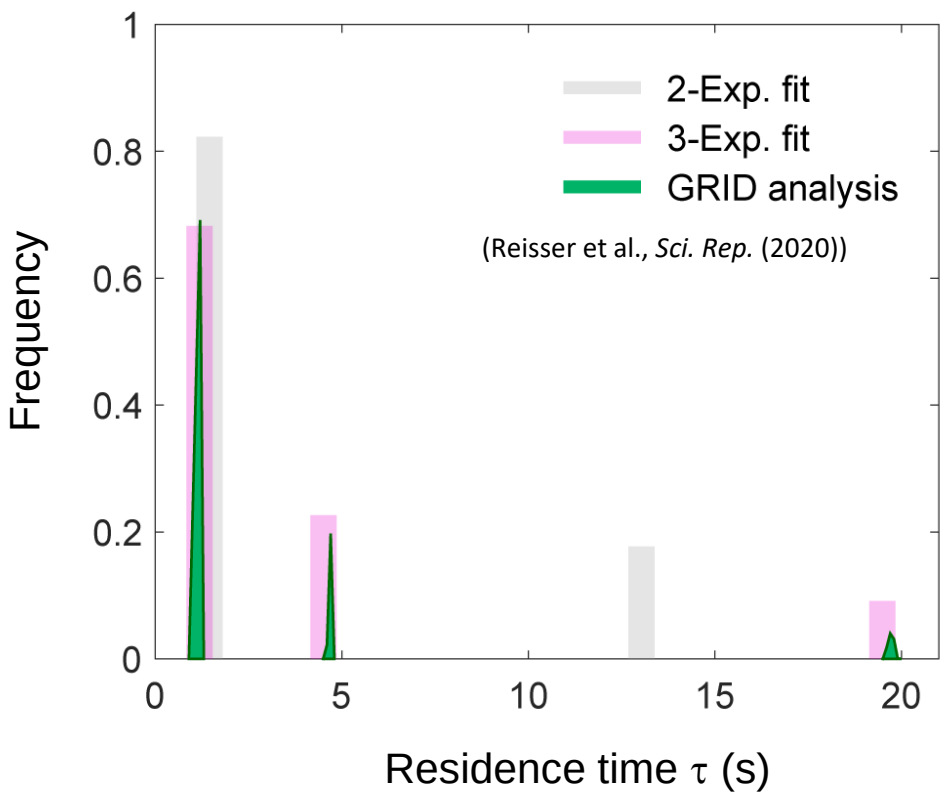
number of jumps /1000

| cell # | 1st diffusion coefficient | 2nd diffusion coefficient | 3rd diffusion coefficient | 1st fraction | 2nd fraction | 3rd fraction |
|---|---|---|---|---|---|---|
| 1.0000 | 0.0741 | 0.5437 | 3.1786 | 20.1813 | 47.6736 | 32.1451 |
| 2.0000 | 0.0912 | 0.5587 | 3.2530 | 21.3157 | 48.8034 | 29.8809 |
| 3.0000 | 0.1207 | 0.8276 | 3.6546 | 18.7813 | 45.2810 | 35.9377 |
| 4.0000 | 0.0877 | 0.7554 | 3.3134 | 17.6494 | 37.6625 | 44.6882 |
| 5.0000 | 0.0627 | 0.5218 | 2.7859 | 16.0196 | 49.3649 | 34.6155 |
| 6.0000 | 0.0786 | 0.5207 | 3.0337 | 19.8259 | 42.4001 | 37.7740 |
| 7.0000 | 0.0943 | 0.4713 | 2.7965 | 30.5156 | 44.6289 | 24.8555 |
| 8.0000 | 0.0864 | 0.5650 | 2.5058 | 24.3869 | 38.6676 | 36.9456 |
| 9.0000 | 0.0515 | 0.2997 | 1.9124 | 28.2324 | 36.7329 | 35.0347 |
| 10.0000 | 0.0788 | 0.4260 | 2.2543 | 18.6103 | 51.7324 | 29.6572 |

Data exported as .mat file

# Quantifying DNA-binding using the survival probability

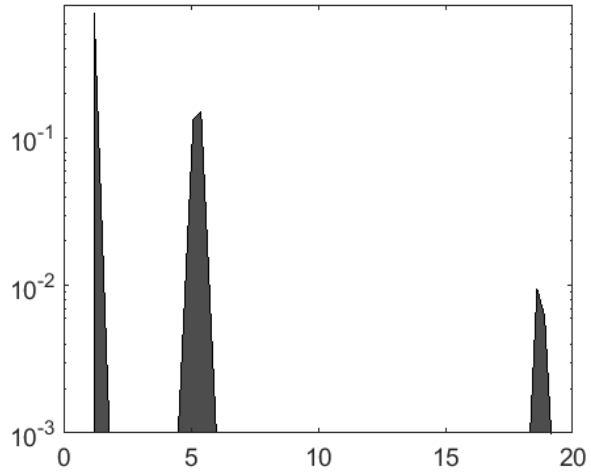


# GET_ResTime_GRID

GRID analysis to determine the number of binding modes

```
path0 = 'D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\slow_tracking\';

Jmin = 4;     %%%   min. number of jumps in the binding trajectory
dt = .3;      %%%   exposure time (s)


E = 0.005;
tmax = 20;
th = 1E-3;

ti = (Jmin*dt:dt:50);
ki = 1./ti;

ResTime = [];
n = list_file(path0, 'bind.mat');

for i = 1:n
    load([path0 'cell_' num2str(i) '_bind']);
    ResTime = [ResTime resTime_distribution(T, Jmin)];
end

[frame, prob] = survivalProb_curve(ResTime, Jmin);
t = dt*frame;

Si = GRID_spectrum(t', prob', ki, E, dt);
% [F, T] = GRID_bar(ti, Si, th);

f = figure(1);
set(f, 'position', [200 200 500 400], 'color', 'w');
a = area(ti, Si, 'FaceColor', [.3 .3 .3]);
ylim([th .8])
xlim([0 tmax])
set(gca, 'yscale', 'log')
```

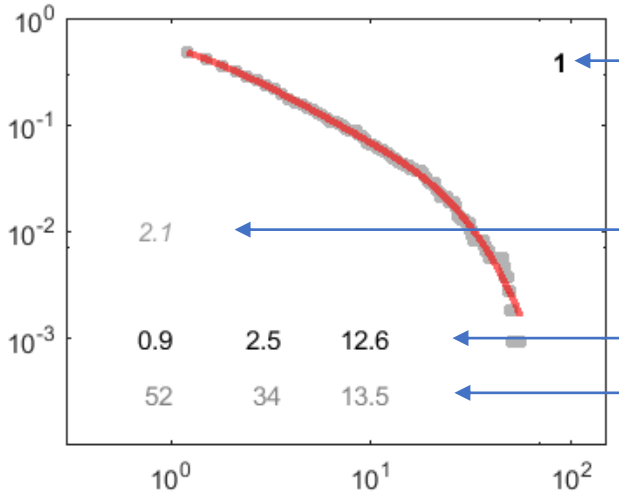



Residence time (s)

# GET_ResTime

## Get the residence time for each mode

```
path0 = 'D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\slow_tracking\';

Nexp = 3;     %%%   number of binding modes

dt = .3;      %%%   exposure time (s)
Jmin = 4;     %%%   min. number of jumps in the binding trajectory
px = .16;     %%%   pixel size (micron)

tmax = 150;   %%%   max. time to display the survival probability
ym = 1E-4;    %%%   min. value of the survival probability to display


[fun, p0] = binding_ExpFit(Nexp);

X = [];
n = list_file(path0, 'bind.mat');
[Nx, Ny] = NxNy(n);

f = figure(1);
set(f, 'position', [200 500-180*Ny/2 300*Nx Ny*190+20], 'color', 'w');

for i = 1:n
    if exist([path0 'cell_' num2str(i) '_bind.mat'], 'file')

        load([path0 'cell_' num2str(i) '_bind.mat']);
        ResTime = resTime_distribution(T, Jmin);
        Density = binding_density([path0 '-\cell_' num2str(i) '_roi.tif'], ResTime, px);

        [frame, prob] = survivalProb_curve(ResTime, Jmin);
        t = dt*frame;
        [p, P] = survivalProb_fit(t, prob, fun, p0);
        N = P(end);
        P = [i P(1:end-1) Density];
        X = [X; P];

        x0 = mod(i-1,Nx);
        y0 = floor((i-1)/Nx);
        ax = axes('units', 'pixels', 'position', [55+x0*290 30+(Ny-1-y0)*185 245 180]);
        box on
        survivalProb_plot(t, prob, fun, p, P, tmax, N);

    end
end
```

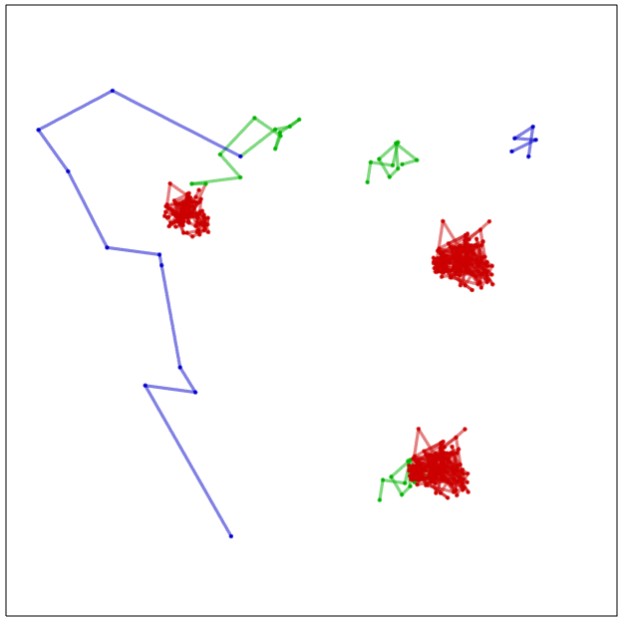



| cell # | 1st residence time | 2nd residence time | 3rd residence time | 1st fraction | 2nd fraction | 3rd fraction | binding density |
|---|---|---|---|---|---|---|---|
| 1.0000 | 0.9355 | 2.5130 | 12.6038 | 52.0040 | 34.4786 | 13.5174 | 2.0542 |
| 2.0000 | 0.9766 | 5.0952 | 24.6191 | 71.8951 | 27.1150 | 0.9899 | 2.2116 |
| 3.0000 | 0.8220 | 4.0671 | 15.3903 | 70.0392 | 25.7390 | 4.2218 | 2.7750 |
| 4.0000 | 0.8065 | 3.5284 | 10.4710 | 60.8319 | 35.3396 | 3.8286 | 4.0440 |
| 5.0000 | 1.1411 | 4.5706 | 15.4572 | 65.7115 | 32.1821 | 2.1064 | 3.7635 |
| 6.0000 | 0.8667 | 4.8577 | 17.7233 | 69.0323 | 30.0542 | 0.9135 | 3.8191 |
| 7.0000 | 0.7031 | 3.2842 | 10.3269 | 65.5688 | 24.7968 | 9.6345 | 2.6362 |
| 8.0000 | 1.2514 | 4.0627 | 15.9378 | 65.4349 | 32.9344 | 1.6307 | 4.3291 |

Data exported as .mat file

# Turning SMT trajectories into a superresolution image map

Typical trajectories of a particle diffusing and binding



The pixel size is constrained by the localization precision and the number of counts

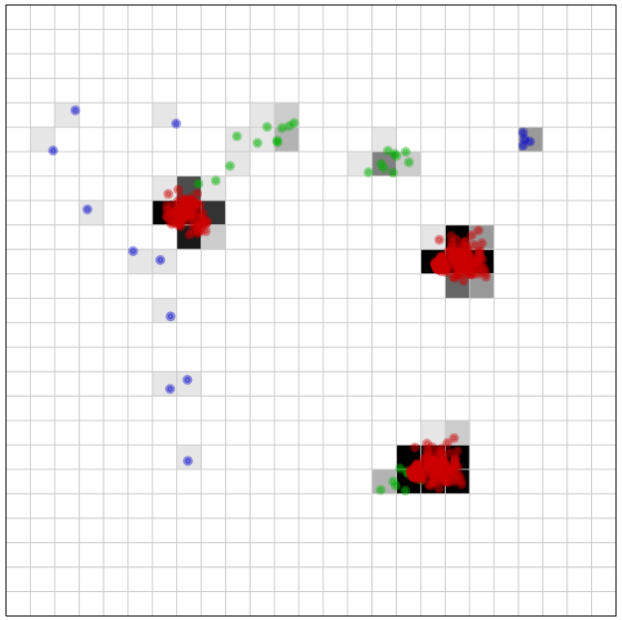
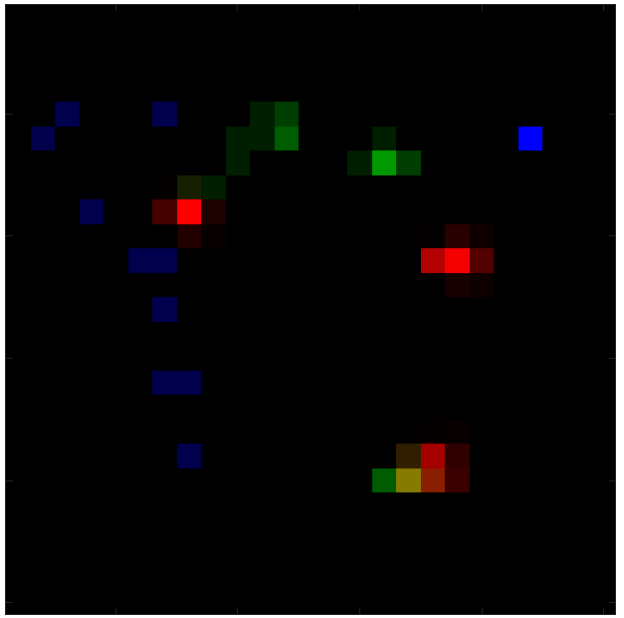


Image reconstruction based on the number of counts per pixel

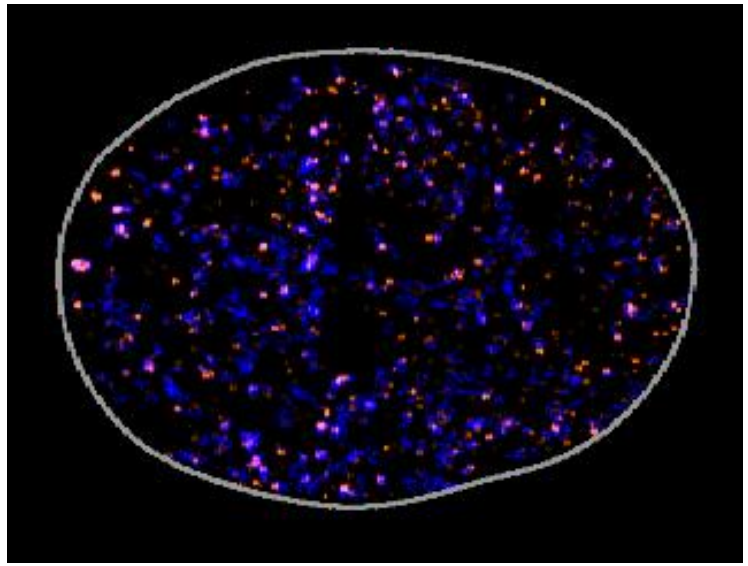


Pixel size: 40 nm

# GET_MAP_Binding_vs_Diff

Map of binding trajectories vs diffusion trajectories

```
path0 = 'D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\slow_tracking\';

p.Nmin = 4;      %%%   min. number of jumps

p.res = .25;     %%%   resolution (px)
p.sigma = .25;   %%%   gaussian blur (px)

p.dir = 1;       %%%   nucleus orientation: horizontal (1), vertical (0), no reorientation (-1)
p.edge = 1;      %%%   showing nucleus boundary: yes (1), no (0)
p.e = 1;         %%%   thickness of the nucleus boundary
dx = 800;        %%%   width (px)
dy = 500;        %%%   height (px)

A = [];
n = list_file(path0, 'bind.mat');
[Nx, Ny] = NxNy(n);

f = figure(1);
set(f, 'position', [200 400 1000 320])
set(gca, 'position', [0 0 1 1])

for j = 1:Ny
    Ax = [];
    for i = 1:Nx

        u = (j-1)*Nx+ i;

        load([path0 'cell_' num2str(u) '_diff']);
        T0 = T;
        load([path0 'cell_' num2str(u) '_bind'])
        p.r = imread([path0 '-\cell_' num2str(u) '_roi.tif']);
        im = MAP_Binding_vs_Diff(T0, T, p);
        im = trim_image(im, dx, dy);

        Ax = [Ax im];

    end
    A = [A; Ax];
end
```

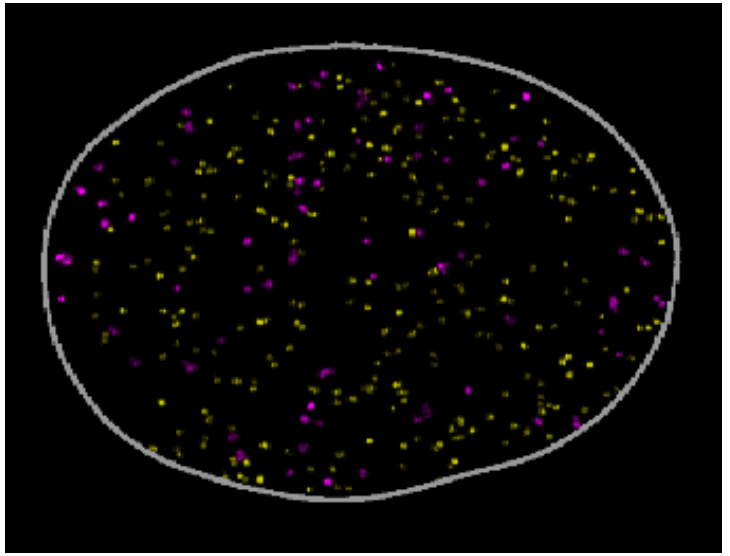


Binding Diffusion

# GET_MAP_Cluster

## Map of clusters of binding trajectories

```
path0 = 'D:\Nat Comm codes\SMT_analysis\Data_BRG1_WT\slow_tracking\';

p.Nmin = 10;     %%%   min. number of jump
p.dmax = 1.25;   %%%   max. distance between binding events in a cluster (px)
p.Nhub = 3;      %%%   min. number of binding events per cluster

p.res = .25;     %%%   resolution (px)
p.sigma = .25;   %%%   gaussian blur (px)

p.dir = 1;       %%%   nucleus orientation: horizontal (1), vertical (0), no reorientation (-1)
p.edge = 1;      %%%   showing nucleus boundary: yes (1), no (0)
p.e = 1;         %%%   thickness of the nucleus boundary
dx = 800;        %%%   width (px)
dy = 600;        %%%   height (px)

A = [];
n = list_file(path0, 'bind.mat');
[Nx, Ny] = NxNy(n);

f = figure(1);
set(f, 'position', [200 400 1000 320])
set(gca, 'position', [0 0 1 1])

for j = 1:Ny
    Ax = [];
    for i = 1:Nx

        k = (j-1)*Nx+ i;
        load([path0 'cell_' num2str(k) '_bind'])
        p.r = imread([path0 '-\cell_' num2str(k) '_roi.tif']);
        im = MAP_Cluster(T, p);
        im = trim_image(im, dx, dy);
        Ax = [Ax im];

    end
    A = [A; Ax];
end
```



Binding Cluster