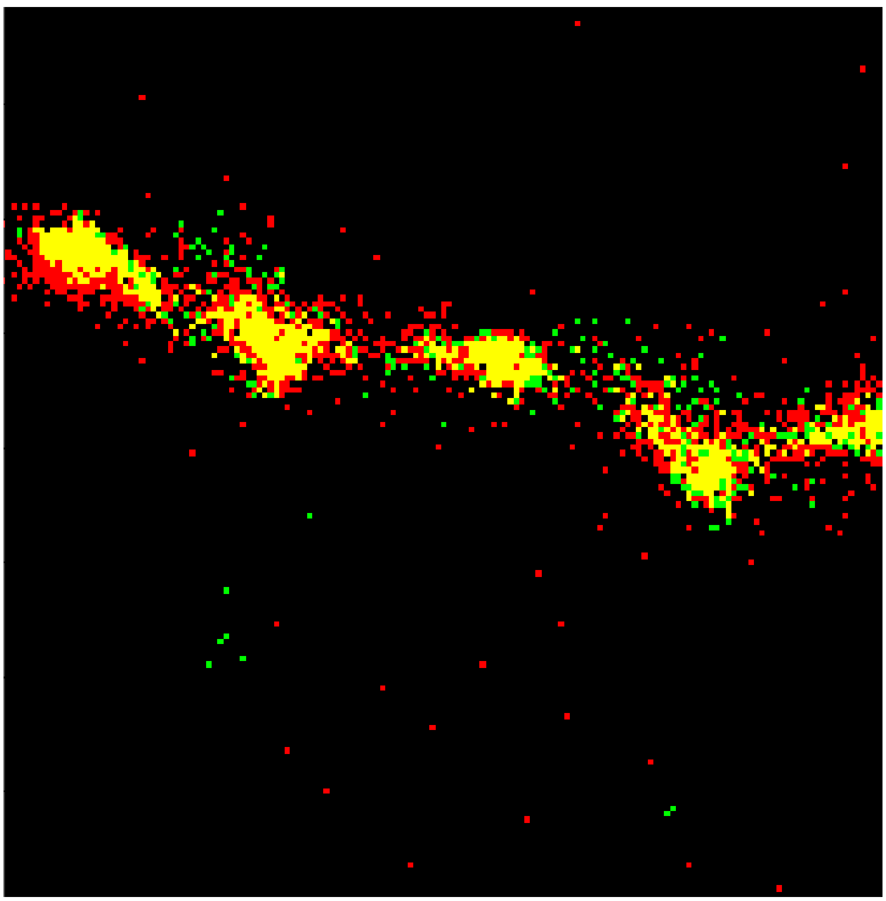
Tc Reconstruction

User Manual



(See Fig.S2, main text; yellow: SMLM&tcSMLM, red: SMLM; green: tcSMLM)

V.1

Table of Contents

Getting Started

Here, we provide a set of instructions toward employing tcReconstruction.

1. Requirements:

MATLAB. Has been tested on R2016b but scripted to work on all versions from 2014-onward.

Parallel Computing Tollbox:

https://www.mathworks.com/products/parallel-computing.html

2. Installation:

A. Download the folder 'tcReconstruction' from our GitHub account: https://github.com/ShermanLab/tcSMLM and unzip it.

B. Choose 'tcReconstruction' as the main folder in your MATLAB.

C. Open "examples.mat" in MATLAB.

D. Download the following samples:

https://drive.google.com/file/d/1p_OejV6NGrte2murSd4b5LyteKR-sTrL/view?usp=sharing

https://drive.google.com/file/d/1eYBUwUtZwMwMh1c0-VyRX1DkHlVNbjjJ/view?usp=sharing

Insert the samples into the tcReconstruction folder.

Now, in the workspace, there are two RawData sets:

1.) Alexa647 experimental data with one emitter (See methods)

2.) Simulation data with two emitters.

In addition, there is one .nd2 file in your current folder, one .tif file and the tcReconstruction functions.

3. Main scripts and examples

A. tcData is created using the matlab script named "tcData.m". In order to apply the script, a 3D matlab matrix containing the RawData should be available (either created using dSTORM/STORM method or by other methods containing temporal information). The tcData function has 3 parameters:

time_step: time_step = 1 is recommended;

MW: see below how to choose the optimal MW;

RawData: the original, unresolved data (intensity information).

Example (1):

tc_Alexa647 = tcData(1,65,Alexa647);

B. In case you use an .nd2 file, a matlab function named "Import_nd2.m" is available. The function also creates a .tif stack file in the folder containing the .nd2 file.

Example (2):

PAmC_NRas = Import_nd2 ('Desktop/tcReconstruction', 'PAmC_NRas', 2000, 0);

C. Otherwise, if you have a RawData in a .tif stack file, a simple function named "ReadTif.m" is available. The function provides a faster way to import .tif files comparable to a direct import using Matlab.

Example (3):

PAmC_NRas = ReadTif('PAmC_NRas.tif', 2000);

4. Finding the optimal MW

The default parameters are in the range: $25 \leq MW \leq 55$. Lower for slow frame rate (20-50 fps), higher for fast frame rate (100+ fps). The function to find the optimal MW is: 'FindBestMovingWindow.m'. The function has 4 parameters:

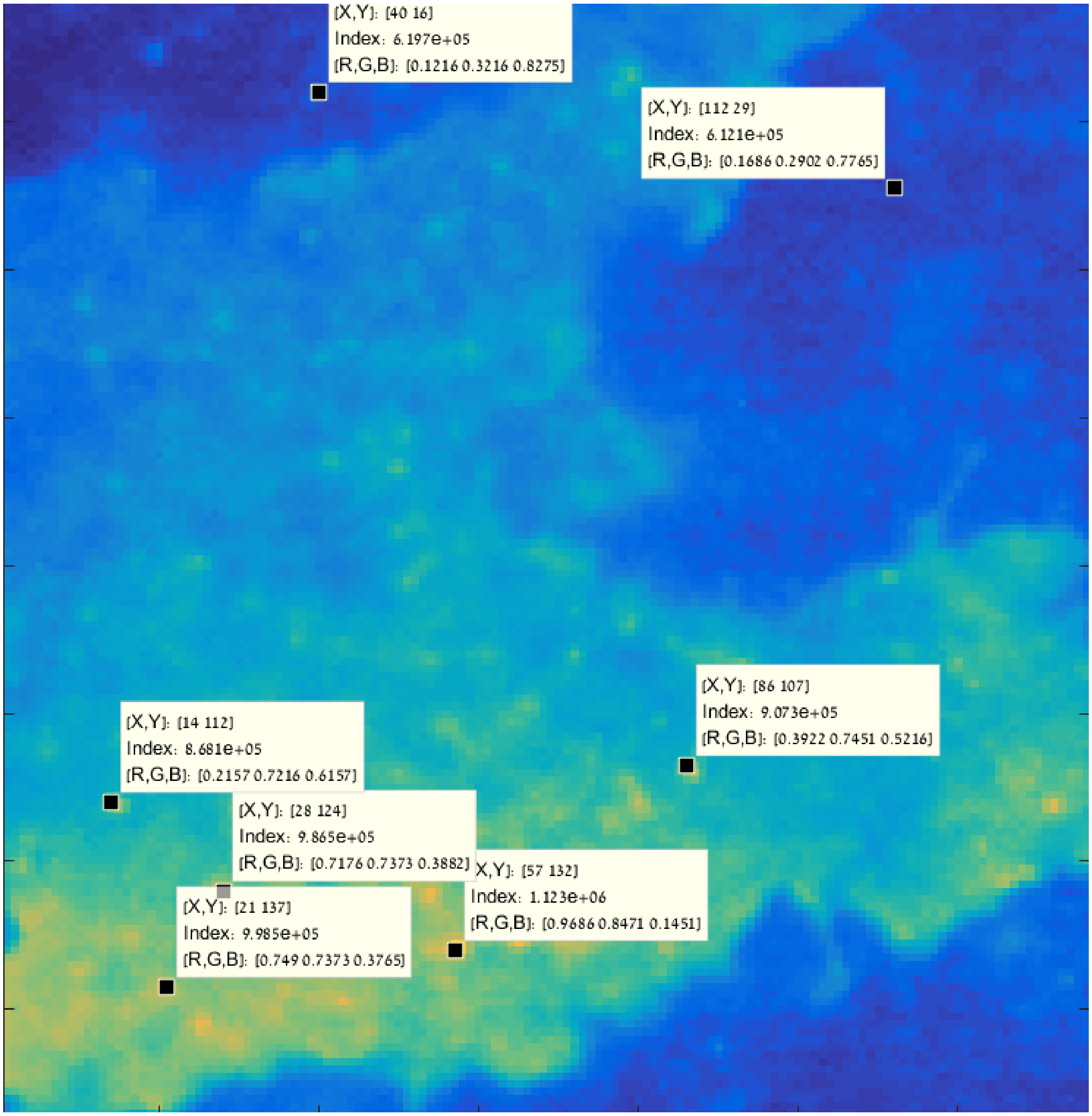
X = an array of size n.

Y = an array of size n.

Smooth = moving average. Ranges between 10-20, depends on the noise in the data.

rawdata = the original (experimental, as intensity) data.

For example, we look at the "PAmC_NRas" sample. Here is the sum intensity where we chose a number of locations:



These locations contain the X and Y arrays:

X = [40,112,14,28,21,57,86], Y=[16,29,112,124,137,132,107].

So:

FindBestMovingWindow([40,112,14,28,21,57,86], [16,29,112,124,137,132,107], 20, PAmC_NRas);

We receive 3 figs in return. The important of them is the third one, containing the MW for each curve. Here, we find: the minimal MW which is bigger that 20 (see main text, Figs5&6) is MW = 27. Therefore, the optimal tcData for PAmC_NRas is:

tc_ PAmC_NRas = tcData(1,27, PAmC_NRas);

5. Final Reconstruction

Once the tcData is created any SMLM / super-resolution algorithm could be applied. That could be done by saving the data to an external data file (such as .tif) or using a MATLAB based super resolution algorithm.