How to run SIMnoise from Matlab

Introduction

This readme explains how to process a TIFF file using Matlab.

The purpose of the software is to demonstrate new noise-controlled SIM image reconstructions. The code has been written in a functional architecture, in order to express the underlying ideas in code in a clear way, and has not been optimized in any way for processing speed (e.g. no parallel processing used anywhere) or for ease-of-use (no GUI provided, just scripts). The envisioned users are therefore more code developers that are interested in some of the underlying algorithms than end-users in biology, although with some effort the code can be applied without detailed understanding of the inner workings. We invite anyone interested and willing to improve usability or re-develop code for different, more usable, platforms.

The code is free for use under the MIT-license. If parts of the code are re-used or the underlying algorithms proposed in the paper are re-coded a citation of our paper is expected as courtesy.

The software is built on our own ideas, but also on the body of ideas on SIM reconstruction that we found in the literature and that we learnt from personal interaction with experts in the field. In particular we thank Lothar Schermelleh, Kai Wicker, and Rainer Heintzmann for inspiration. Flaws are our own of course.

Carlas Smith and Sjoerd Stallinga

System requirements

To function properly, the pipeline requires a computer that meets the following specifications:

- Any Intel or AMD x86-64 processor (min. 4 cores recommended)
- 32 GB of RAM or more (min. 20 GB)
- 4-8 GB of space for typical installation & data

The performance of the sim noise pipeline depends critically on the dataset size and the available computational power. For details on the hardware configurations that have been tested, refer to Table 1.

System configuration				**SIMnoise performance**	
CPU	**Memory**	**System type**	**OS/MatLab/DIP**	**Version**	**Runtime**
Intel i9-8950hk 6 cores@2.90GHz	DDR4 32GB@2666MHz	64bit	Windows 10 ent. R2020a / DIP 2.9	1.0	954 s
Intel i9-10980HK 8 cores@4.4 GHz	DDR4 32GB@2666MHz	64bit	Windows 10 ent. R2020a / DIP 2.9	1.0	812 s

Table 1. Specifications of tested systems.

Windows installation

The SIMnoise software is developed in Matlab2018b and Matlab2019b, running on Windows 10, and is expected to run on older and newer versions of Matlab as well. Some functions from the dipimage/diplib toolbox (www.diplib.org) are used. Installation of this toolbox is needed to run the full code.

Steps:

1. Clone repository or download software from https://github.com/qnano/simnoise
2. Install MatLab using the installation materials and procedures provided by your institution or acquired with the MatLab license. The following Matlab toolboxes are needed:
    a. **Image Processing Toolbox** (tested version 11.1)
    b. **Statistics and Machine Learning Toolbox** (tested version 11.7)
    c. **Signal Processing Toolbox** (tested version 8.4)
3. Install Bio-formats (tested version 5.5.3)
    a. Download the latest version from https://www.openmicroscopy.org/bio-formats/downloads/.
    b. Unzip bfmatlab.zip
    c. copy the unzipped bfmatlab folder into "/simnoise/helperfunctions/" - here it will be automatically added to your Matlab search path.
4. Install DIPimage for MatLab (tested version 2.9)
    a. Download the latest version from http://www.diplib.org/download. To maximize stability, make sure that the version you download has been tested with the MatLab version you use.
    b. Install DIPimage for MatLab (refer to DIPimage User Manual for assistance)
    c. Add the following lines to the MatLab startup file startup.m (usually found in Documents\MATLAB; create file if it does not exist yet):

        run('C:\Program Files\DIPimage\dip_initialize');

        Note: if DIPimage was not installed in the default location, the line above needs to be changed. In some DIPimage distributions, the DIPimage folder in Program Files is given a name that includes the version number (e.g. DIPimage 2.9).
    d. Confirm that DIPimage was installed successfully by executing the command dipimage in MatLab. Refer to the *DIPimage User Manual* for assistance with any unresolved errors.

Data processing

This section describes each component, using the example dataset. Here we will walk you through the steps to analyse the example dataset:

Steps:

1. Download the example data from https://doi.org/10.4121/14428124 and place it in the directory "/simnoise/data/".
    I. Other data please can be downloaded from https://doi.org/10.4121/12942932. place it in the directory: "/simnoise/data/OMXdatafiles" or adjust the variables dataparams.datadir & dataparams.otfdir.

2. Easy mode: Open "/simnoise/SIMnoise code/SIMnoise.m" and run in Matlab (press F5). This will automatically set the working directory to: "/simnoise/SIMnoise code/".

   SIMnoise.m is the main script of the software and is organized in three main parts, devoted to (1) pre-processing, (2) illumination pattern parameter retrieval and constructing the image and OTF Fourier orders, and (3) making the different required SIM reconstructions. Added comments with/in the different function m-files will further help a user to find his/her way in the code. **SIMnoise.m is an example for OMX data. An equivalent example for Zeiss data can be found in SIMnoise_zeiss.m.**

   Expert mode (optional): The individual steps can be executed separately, and are available as separate scripts. Relevant parameters are added to the struct "SIMparams" along the way. In developing the code we found this easier for focusing on sub-parts of the entire processing chain. The results are saved in: \SIM noise\data\[dataset filename]\. We have included a flag for displaying intermediate results in graphical form, for debugging or quality control purposes.

   a. Run "SIMdata_reformat_omx.m", this formats data and meta data.

      Flags and settings:
      Section 1:
      i. Load dataset to process
      Section 2:
      ii. Parameters of acquisition, optical system, and detector

      The script reads the data as an image stack, which is reshaped to a 7D array "allimages_in". The 7 dimensions of the array refer to numpixelsx, numpixelsy, numsteps, numfocus, numchannels, numframes, numangles. This data format allows for the processing of 2D/3D SIM, multiple color channels and multiple repeated frames e.g. in live aquisitions. Also, a metadata-file, containing information on e.g. wavelength, NA and pixel sizes, is expected to be present. For another example see "SIMdata_reformat_zeiss.m".

   b. Run "/simnoise/SIMnoise code/SIMdata.m to pre-process the data. Step a has produced metadata and imagedata mat file(s), which will be loaded.

Flags and settings:

Section 1:

i. Load dataset to process

Section 2:

ii. Crop data x,y,z,t
iii. Apodization and padding x,y,z
iv. Drift correction on/off
v. Random binomial datasplit for resolution determination
vi. Equalize intensities of angles and z layers

Section 3:

vii. Visual check of images

c. Run "/simnoise/SIMnoise code/SIMpatterns.m" to estimate the pattern parameters and unmixes the raw data. Step b has produced a SIMparamsfile.mat and preprocessedimages mat file(s), which will be loaded.

Flags and settings:

Section 1:

i. Load dataset to process

Section 2:

ii. Optical transfer function (OTF) input

Section 3:

iii. Do pattern estimation y/n
iv. Maximum #iterations in illumination peak detection
v. Tolerance criterion in illumination peak detection
vi. Zoom factor in Fourier space

Section 4:

vii. Store at single precision to save disc space

d. Run "/simnoise/SIMnoise code/SIMreconstruction.m" to create the reconstructions. Step c has produced an “image Fourier order data.mat” and processed image data results mat files, which will be loaded.

Flags and settings:

Section 1:

i. Load dataset to process

Section 2:

ii. The different type of reconstructions available
iii. Maximum #iterations in illumination peak detection
iv. Notch filter settings
v. Order strengths : estimated / user defined
vi. Apodization filter settings
vii. Show reconstruction

The final reconstructions are made per color channel per time frame per reconstruction type. Possible reconstructions are of the type "stateofart", "flatnoise" or "truewiener", and can be generated with or without notch filtering or with multiple/different parameter settings.

3. To create the figure and movie files from the manuscript the code is provided in the folder '/simnoise/makefiguresmovies'. The code can easily be modified for other purposes.