# **Instruction to Deep-SMOLM**

Contents

[**Instruction to Deep-SMOLM** 1](#_Toc109752481)

[**Preparing the environment and software** 3](#_Toc109752482)

[**Matlab** 3](#_Toc109752483)

[Install matlab here 3](#_Toc109752484)

[**Visual studio code (VS code)** 3](#_Toc109752485)

[Download visual studio code here: 3](#_Toc109752486)

[Learning debugging in visual studio code here: 3](#_Toc109752487)

[**Anaconda** 3](#_Toc109752488)

[Download the Anaconda here: 3](#_Toc109752489)

[Build environments for Deep-SMOLM: 3](#_Toc109752490)

[**Install packages** 4](#_Toc109752491)

[Launch visual studio from anaconda 4](#_Toc109752492)

[Change the environment to the new created environment through terminal in VS code 4](#_Toc109752493)

[Install packages through conda using terminal 4](#_Toc109752494)

[**Generating the training data** 5](#_Toc109752495)

[**Option 1**: generating training data from matlab scripts 5](#_Toc109752496)

[**Option 2**: use training data in OSF 5](#_Toc109752497)

[**Training Deep-SMOLM** 6](#_Toc109752498)

[**Set comet ml account to monitor the training process** 6](#_Toc109752499)

[**Modify the config\_orientations.json** 6](#_Toc109752500)

[Parameters you might need to change 6](#_Toc109752501)

[Parameters you can keep it be the same 8](#_Toc109752502)

[**Running the training code** 8](#_Toc109752503)

[**Estimation using Deep-SMOLM** 8](#_Toc109752504)

[**For simulated data with ground truth** 8](#_Toc109752505)

[Check if the data loader method match to your data 8](#_Toc109752506)

[Modify the config\_orientations.json file 8](#_Toc109752507)

[Run the estimation code 9](#_Toc109752508)

[**For Experimental data** 9](#_Toc109752509)

[Check if the data loader method match to your data 9](#_Toc109752510)

[Modify the config\_orientations.json file 9](#_Toc109752511)

[Run the estimation code 10](#_Toc109752512)

# **Preparing the environment and software**

## **Matlab**

Matlab is used to generate the simulate data

### Install matlab here

[Installation and Licensing Documentation (mathworks.com)](https://www.mathworks.com/help/install/)

## **Visual studio code (VS code)**

I recommend use visual studio code for running Deep-SMOLM due to its user-friendly debugging interface. But any other platforms will also work.

### Download visual studio code here:

[Download Visual Studio Code - Mac, Linux, Windows](https://code.visualstudio.com/Download)

### Learning debugging in visual studio code here:

<https://code.visualstudio.com/Docs/editor/debugging>

## **Anaconda**

Using anaconda for installing packages

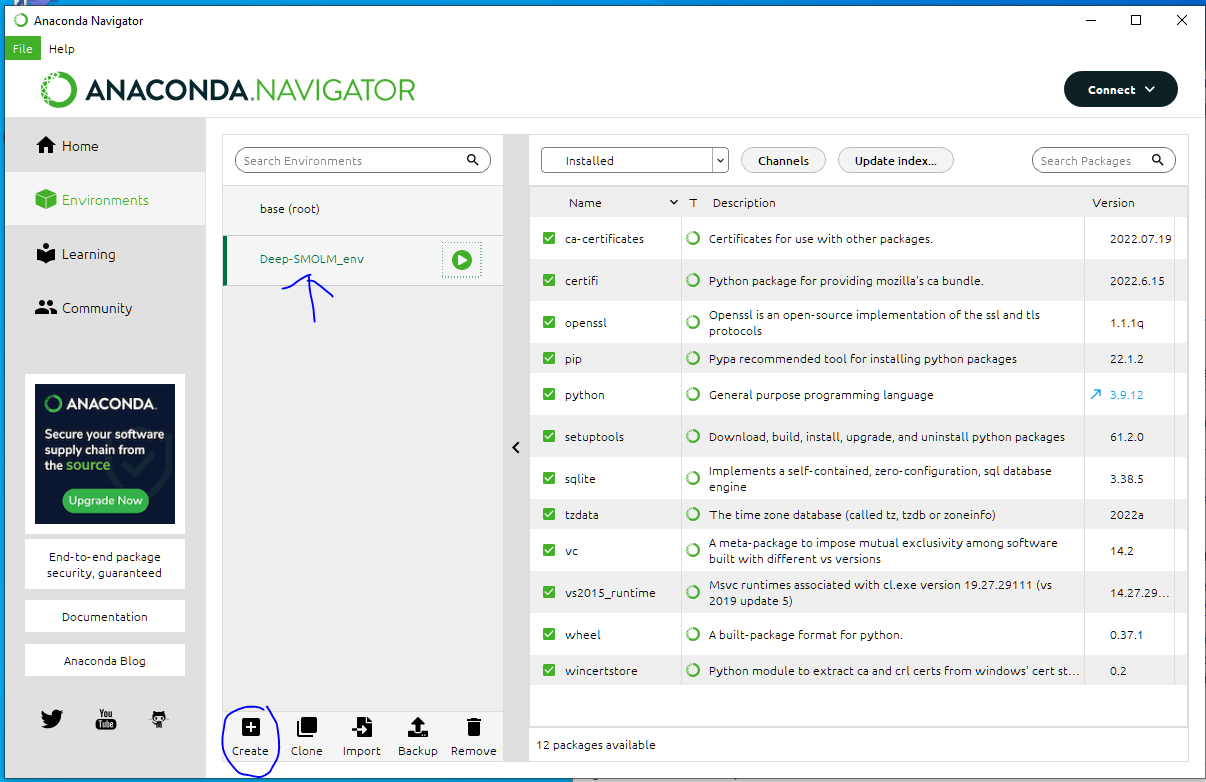
### Download the Anaconda here:

[Anaconda | The World's Most Popular Data Science Platform](https://www.anaconda.com/)

### Build environments for Deep-SMOLM:

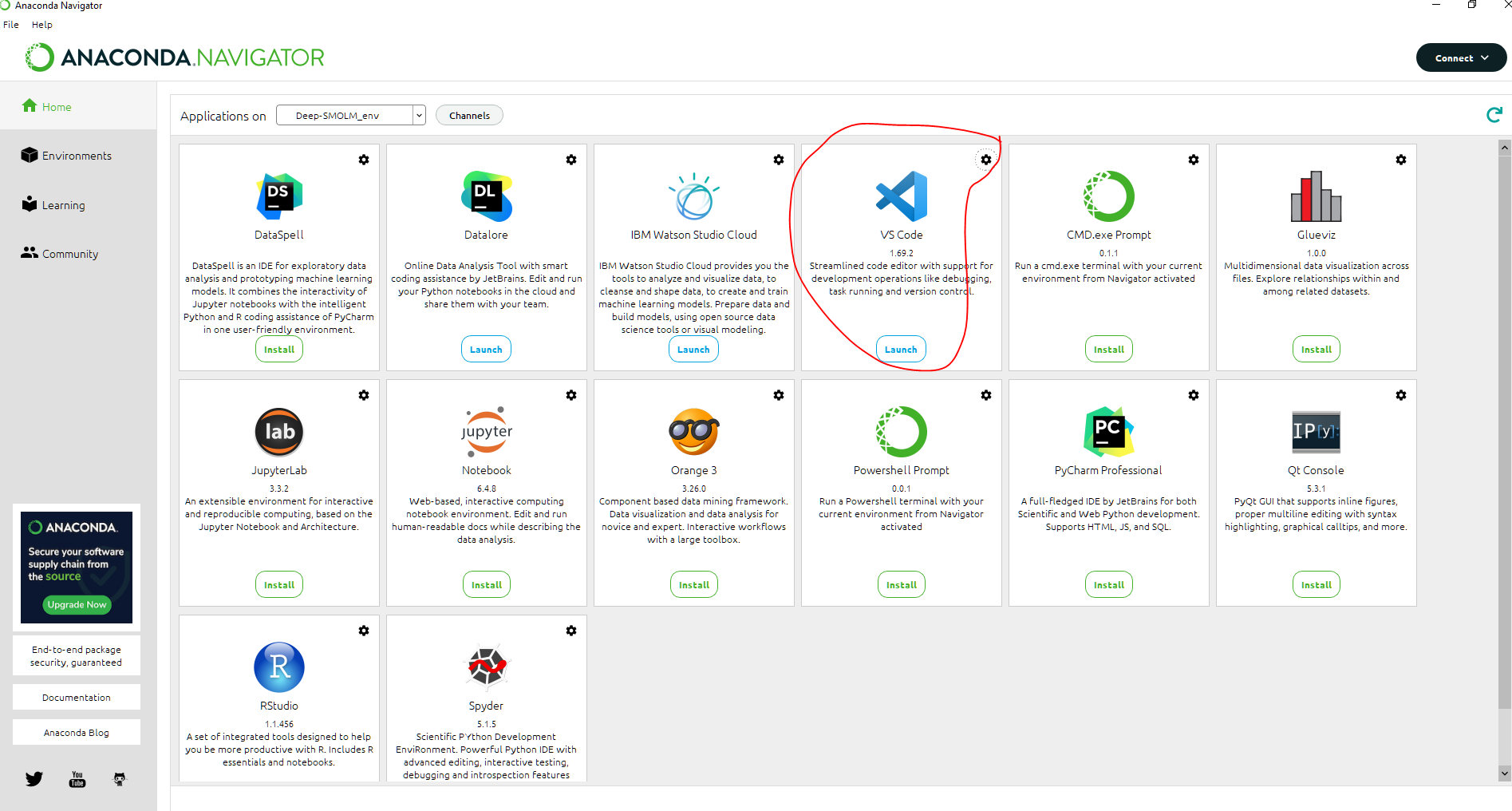
To separate the running environment of Deep-SMOLM from system’s basic environment or environments of other algorithms, I recommend building a specific environment for Deep-SMOLM and install required packages inside this environment.

In here, I created a new environment named Deep-SMOLM\_env



## **Install packages**

### Launch visual studio from anaconda



### Change the current environment to Deep-SMOLM’s environment through terminal

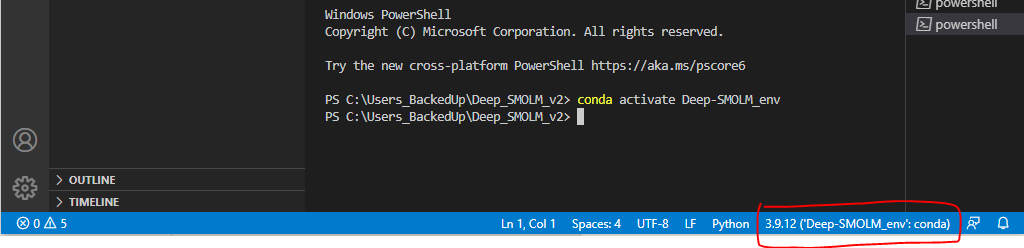
* Open terminal on visual studio



* Type in: conda activate [new environment name]



* If success, you will see that the visual studio is running under the designed environment



### Install packages using conda

* Make sure the previous changing environment is successful. Otherwise, the packages won’t install in the desired environment.
* Type in the following command in the terminal of visual studio one by one. This will take some time.

conda install -c pytorch pytorch

conda install -c anaconda numpy

conda install -c comet\_ml comet\_ml

conda install -c anaconda scipy

conda install -c conda-forge tifffile

conda install -c conda-forge matplotlib

conda install -c anaconda pillow

conda install -c conda-forge opencv

conda install -c anaconda scikit-image

conda install -c pytorch torchvision

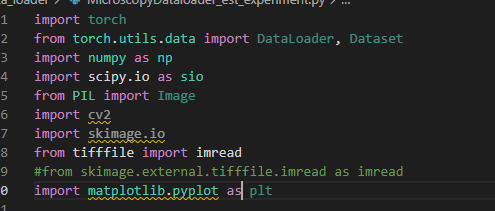
conda install -c conda-forge tqdm

conda install -c conda-forge iprogress

conda install -c anaconda jupyter

* Once you successfully installed all the packages; the wave underlines in the code will disappear

Examine all the .py files in Deep-SMOLM, make sure there is no wave underlines. If there is, install the related package



# **Generating the training data**

There are two options for getting the training data

## **Option 1**: generating training data from MATLAB scripts (recommended)

As the data generating code is relatively fast (~2h for 30K data), I recommend generating the training data from the MATLAB scripts. Also, in the Deep-SMOLM algorithm package, there are small amount of training data (100 images) for you to test the algorithm is running fine.

* All the simulated data is generated using Matlab scripts under Deep\_SMOLM\_v2\forward\_model\_pixOL; Add you this folder to your MATLAB path.
* Generate training data using generate\_training\_images\_pixOL\_perfect.m for pixOL microscope without aberration ((algorithm used for Fig.1-3 in Deep-SMOLM paper).).
* Generate training data using generate\_training\_images\_retrieved\_pixOL\_w\_focal\_drift.m. for calibrated pixOL microscope (algorithm used for Fig.4 in Deep-SMOLM paper).; This code generates training images containing emitters that is imaged with focal plane randomly sampled from [-150,150] nm.

## **Option 2**: use training data in OSF

Instead of generating your own training data, you can also directly use the training data stored in the OSF

https://osf.io/x6p8r/?view\_onlyb263a8693c5e4418a0b962df31ca0101

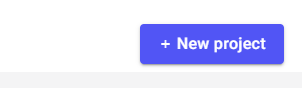
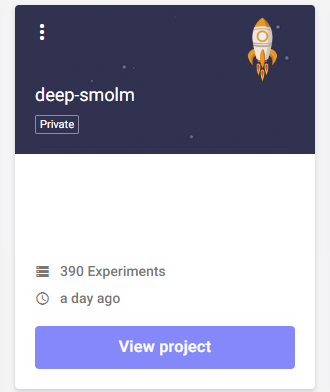
# **Training Deep-SMOLM**

All the user defined parameters are in the config\_orientations.json file.

## **Set comet ml account to monitor the training process**

I use comet.ml to monitor the training process in live.

* Create your comet.ml account here: [www.comet.com](http://www.comet.com)
* Create new project

* Find the API of the new account and enter your API to configuration file

Read your API key number under the account profile. (<https://www.comet.com/docs/rest-api/getting-started/>)

At the Deep-SMOLM file: config\_orientations\_v2.json, enter your API key and change the “savedata” option to be true if you want to turn on the monitoring

Text

Description automatically generated

## **Modify the config\_orientations.json**

config\_orientations\_v2.json is a file that contains all the information about your training, testing, validation parameters. Idealy, you just need to modiy this configuration file to run the Deep-SMOLM algorithm.

### Parameters you might need to change

* Name of the task; Tother names (name1, name2) won’t be read by the code. You can switch between different tasks by assigning the [name] to the desied task.

"name": "training\_perfect\_pixOL",

* Information of your comet.ml account; the training/testing loss can be monitored on the comet website.

    "comet": {

        "api": "your API key",

        "offline": false,

        "savedata": false

    },

* Change the “save\_dir”: directions you want save the trained results/estimation results

  "trainer": {

        "epochs": 100,

        "warmup": 0,

        "save\_dir": "Examples/trained\_Deep-SMOLM\_model",

        "save\_period": 1,

        "verbosity": 2,

        "asym": false,

        "sym": false,

        "percent": 0.9,

        "subset\_percent": 1.0,

        "monitor": "min test\_loss",

        "early\_stop": 10

    },

* change the [batch\_size] and [num\_workers] based on your GPU memory and CPU cores

    "\*\*\*\*[data\_loader]\*\*\*\*\*\*\*":" ",

    "data\_loader": {

        "type": "MicroscopyDataLoader",

        "args":{

            "batch\_size": 6,

            "shuffle": false,

            "validation\_split": 0,

            "num\_workers": 8

        }

    },

* Change your training data’s name, location, and size

“noiseless\_image\_name”: the passion shot noise is added during the training, so please use noiseless image

“GT\_image\_name”: name of the ground truth image

“file\_folder”: location where you save your training data

“number\_images”: number of the training data

    "training\_dataset": {

        "noiseless\_image\_name":"image\_noiseless",

        "background\_name":"",

        "GT\_image\_name": "image\_GT\_up",

        "GT\_list\_name": "",

        "file\_folder": "Examples/training\_data/example1\_100images",

        "number\_images":100

    },

### Parameters you can keep it be the same

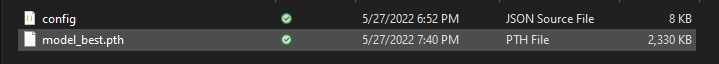
Please understand other parameters by reading through the code

## **Running the training code**

* Once you modify the config\_orientations.json, you can directly run the code: Deep\_SMOLM\_main.py either through running or debug running.

I highly recommend running through the debug mode, so that if something is run, you can easily debug the code

* You can monitoring your training curves at [www.comet.ml](http://www.comet.ml)
* Once the running is finished, you will find your results at the folder your assigned for saving the data



# **Estimation using Deep-SMOLM**

## **For simulated data with ground truth**

Use Deep\_SMOLM\_est.py

### Check if the data loader method matches to your data

* Modify the MicroscopyDataloader\_est.py data reading code to fit into your data if needed.

Current reading method: each estimation paired image is saved into one .mat file; the code read the data batch by batch

### Modify the config\_orientations.json file

Give the “noise\_image\_name”, “file\_folder”, and “save\_name”. You can also give ground truth of your name in “GT\_list\_name” if applicable, otherwise using “”

    "est\_dataset": {

        "noise\_image\_name":"image\_with\_poission",

        "GT\_image\_name": "image\_GT\_up",

        "background\_data":"img\_bkg\_",

        "GT\_list\_name": "GT\_list",

        "file\_folder": "Examples/simulated\_biological\_fiber/fiber\_omega0\_100images",

        "number\_images":100,

        "batch\_size": 16,

        "save\_name":"phantom\_orientation\_estimation\_dense\_SMs.mat"

    },

### Run the estimation code

* Use Deep\_SMOLM\_est.py
* Give the direction of Deep-SMLM model; this is the direction that you save your trained Deep-SMOLM model.

args.add\_argument('-r', '--resume', default="your direction/model\_best.pth", type=str,help='path to latest checkpoint (default: None)')

## **For Experimental data**

Use Deep\_SMOLM\_est\_experiment.py

### Check if the data loader method matches to your data

* Modify the MicroscopyDataloader\_est\_experiment.py data reading code to fit into your data if needed.

Current reading method: a stack of images is saved in the .tiff format; the code read the offset image, and read the single molecule image batch by batch

### Modify the config\_orientations.json file

    "est\_dataset\_experiment": {

        "noise\_image\_name": "\_centerY\_y410\_x\_200\_FoV420\_220\_",

        "GT\_image\_name": "",

        "file\_folder": "/Examples/Example\_training\_data/",

        "batch\_size":16,

        "number\_images\_per\_dataset":1000,

        "number\_FoV": 14,

        "number\_dataSet": 5,

        "starting\_dataSet":12,

        "upsampling":6,

        "offset\_name":"offset\_centerY\_y410\_x\_200\_FoV420\_220\_",

        "background\_name":"\_bkg\_centerY\_y410\_x\_200\_FoV420\_220\_",

        "tophoton": 0.29,

        "y2x\_channel\_ratio":1.145,

        "save\_name":"A\_beta\_data"

    }

### Run the estimation code

* Use Deep\_SMOLM\_est\_experiment.py
* Give the direction of Deep-SMLM model. This is the direction that you save your trained Deep-SMOLM model.

args.add\_argument('-r', '--resume', default="your direction/model\_best.pth", type=str,help='path to latest checkpoint (default: None)')