Instructions

1. python make_traj_path_file.py
2. python bootstrap_ml_fit.py 1
3. python gather_best_params.py
4. python plot_histograms.py

High level explanation

The goal is to fit a kinetic model of fluorophore blinking to a set of time traces (trajectories). The inputs are time traces and initial guesses for the kinetic model parameters, and the outputs are likelihoods (probability densities) and optimized values for the kinetic model parameters.

In order to determine error bars around our estimates of the model parameters, we carry out multiple optimization runs. Each run randomly selects a subset of the available trajectories (we refer to this as bootstrapping). After many runs, we can plot a histogram for each parameter in the model. The center of the histogram is our best guess and we can compute a 95% confidence interval from the percentiles of the histogram.

Lower level explanation

-make_traj_path_file.py looks for .csv files in a specified directory. It outputs a text file ("traj_paths.txt"). Each line of the text file has a path for a csv file. This text file is used for the bootstrapping procedure described above.

-bootstrap_ml_fit.py does several things:

1. It selects a random subset of the csv files listed in "traj_paths.txt". Then, it saves a new .txt file in bootstrap_files. The format of the new .txt file is the same as "traj_paths.txt", but it only contains the paths for the random subset of trajectories.
2. It calls run_optimization from the opt_fcn.py module. This is the major workhorse function. run_optimization reads the .txt path file in "bootstrap_files", runs the maximum likelihood fitting procedure, and then returns the results to bootstrap_ml_fit.py.
3. Finally, it saves the parameters determined by run_optimization and repeats step 2 for other values of N (the number of fluorophores). The output is saved in the params directory as both a pickel archive and as a human-readable html file (for convenience).

-run_job.py is a simple helper script. It calls bootstrap_ml_fit.py many times.

-gather_best_params.py looks at each parameter file in the "params" directory and creates a new .pkl (and .html) with only the best scoring (highest likelihood) parameters from each run. The idea here is that bootstrap_ml_fit.py usually calls run_optimization for a range of N values, but we only want to retain the best N from each run for our histogram plotting.

-plot_histograms.py reads the best parameters .pkl file and plots histograms of the parameters in the directory "histograms". These plots represent the final output that we want. As stated above, the center of each histogram is our best guess for that parameter and the 95% confidence interval for that parameter comes from the percentiles of the histogram.