Single Particle Tracking in E. Coli Cells

Authors: Stella Stylianidou, Phil Snyder

Names and Roles

Stella –

Phil –

What the program is supposed to do

Biology has been seeing great advances in the last few years, with being able to tag proteins in live cells and observe them during the lifecycle of a cell. We can take pictures of those proteins and observe where they are found in a every frame. A main problem is to able to link those different particles to figure out the tracks they took during the duration of the experiment. There are several difficulties with this: if there are a lot of proteins it is hard to find from which one they came from in the previous frame, the protein may split into two, or a protein may merge with another protein, the proteins may disappear and reappear in a frame.

Our program receives real data of the positions of proteins in E.coli cells in each time frame. It starts with random tracks and then modifies the tracks of the proteins in each frame with proteins observed in the next frame attempting to create the track. It then attempts to add splits and merges to the tracks to all the possible places were that can happen (ends and starts of tracks), and it decides whether that is a good choice or not according to the cost.

Technique: Simulated Anneal

Our program uses simulated anneal to identify whether a neighbor state should be accepted or not. The way this works is the following

1) It starts with some random connections for the each protein in each time frame with another protein in the next time frame. This random connections are just the way the data were inserted in the program.

2) It then calculates neighbor states.

a) For the first half of the iterations the program atempts to optimize the paths. It applies the following operator to the current state to find a neighbor state: the positions of jumps are found (with distance more than the MAX\_JUMP variable). and one of those big jumps and the tracks at which it occurs is selected randomly. Another track that has a big jump at the same time is selected. The new state is created by switching the connections of the two tracks at the time where a jump is found.

b) For the second part it finds tracks that start after 0 or end before the lifetime of the cell. It randomly selects one of those tracks and it then either attempts to connect the start to a random other tracks (create a split) or merge the ends to a random other track that is alive at that time.

4) If the new state is better than the old state accept the new state

5) If the new state is worse accept it with probability exp (cost(old\_state) – cost(new\_state)/T). A higher temperature makes it more likely to accept an inferior state.

6) Returns to step 2 and lowers the temperature at every iteration

Interesting sample session

Demo instructions

1. use run (filename) where filename is one of the sample matlab files provided

2. the program should just start running and showing the process of the simulated anneal as it runs

3. to optimize you can modify the variables, and also modify the temperature and annealing schedule:

MIN\_SCORE : the minimum intensity of proteins that will be included in the calculations  
ALLOW\_SPLITS : to allow splitsBIRTH\_PENALTY : cost penalty for having birth of tracks after time 0  
DEATH\_PENALTY : cost penalty for having death of tracks before end of lifetime  
ALLOW\_MERGES : to allow mergesMAX\_TIME\_WINDOW ?  
MAX\_JUMP : the distance at which it will be considered a jump and thus considered to be changed in the neighbor algorithm

Code Excerpt

Brief Description of what we learnt

What would you add to your program

Citations for references