Problem Statement

In our project at the beginning of the year we had a decision to make: whether we would make our own fitting software or build off of one that has the basics made.

Investigation/Research

We chose the latter and have started working with the fitting software, BioNetFit(BNF). BNF used to be a single Perl file which was obviously hard to debug and use in general for development. BNF 2 is written in C++. Up to now, we have been debugging BNF 2 and annotating the code. We have also started implementing a function which is our project this year.

The following is an overview of what BNF does:

- user chooses an algorithm: genetic, differential evolution, parallel swarm
- depending on said algorithm, user chooses amount of generations and runs.
- BNF then parses the model.bngl (the model file).
- it looks for variables with the definition of `variablename__FREE__`
- this defines a free parameter and what we're fitting.
- Next, the algorithm that was chosen runs for the amount of generations and runs
- it picks the best fit parameters and provides you with a model.bngl with the parameters

Most simulations must be run at a fraction of the cell volume for speed. However this can lead to incorrect fitting because of initial small particle amounts and because of how particles are defined.

The function we are currently implementing allows BNF to scale up the volume of the simulation as you are doing parameter fitting. It takes the parameters and if they are categorized as a good fit, it scales up the volume and restarts the parameter fitting at those parameters. This way you know whether your parameters are valid when you are simulating a whole cell rather than fractions.

Construction/Implementation

We are implementing this for the Particle Swarm Optimization first. Currently, we have made BioNetFit store the fraction of the volume and we have found where a "good" fit is defined. We plan on checking the parameters for a good fit, then changing the fraction, and rerunning the parameter fitting.