

Biomimicry of Bacterial Foraging for Distributed Optimization and Control

Kevin M. Passino¹

Presented by: Alexander Van de Kleut²

¹The Ohio State University
Electrical and Computer Engineering

²University of Waterloo
Centre for Theoretical Neuroscience

IEEE Control Systems Magazine, 2002

Outline

- 1 Foraging as Optimization
- 2 Building the Algorithm
- 3 Discussion

- **Foraging**

- ▶ searching for nutrients
- ▶ avoiding noxious stimuli (toxins, predators, etc)

- **Foraging**

- ▶ searching for nutrients
- ▶ avoiding noxious stimuli (toxins, predators, etc)

- **Social Foraging**

- ▶ increases likelihood of finding nutrients
- ▶ better detection and protection from noxious stimuli
- ▶ gains can offset cost of food competition

How can we view foraging as an Optimization Process?

- We have some parameters θ and a loss function $J(\theta)$ that we want to minimize

Foraging as Optimization

How can we view foraging as an Optimization Process?

- We have some parameters θ and a loss function $J(\theta)$ that we want to minimize
- θ can represent the position of an organism in its environment

How can we view foraging as an Optimization Process?

- We have some parameters θ and a loss function $J(\theta)$ that we want to minimize
- θ can represent the position of an organism in its environment
- J can represent the concentration of nutrients and noxious stimuli
 - ▶ smaller values of J = more nutrients, less noxious stimuli
 - ▶ higher values of J = more noxious stimuli, less nutrients

Foraging as Optimization

How can we view foraging as an Optimization Process?

- We have some parameters θ and a loss function $J(\theta)$ that we want to minimize
- θ can represent the position of an organism in its environment
- J can represent the concentration of nutrients and noxious stimuli
 - ▶ smaller values of J = more nutrients, less noxious stimuli
 - ▶ higher values of J = more noxious stimuli, less nutrients
- In general, J and θ can be arbitrary
 - ▶ $\theta \in \mathbb{R}^p$
 - ▶ $J : \mathbb{R}^p \rightarrow \mathbb{R}$

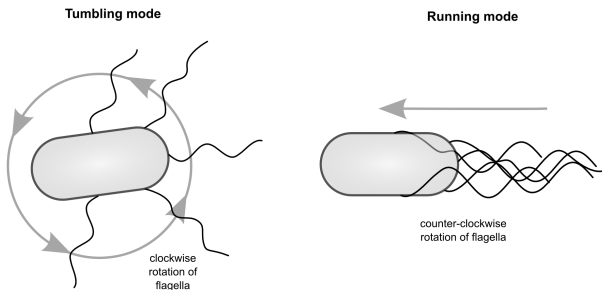
- Model organism
 - ▶ Highly studied
 - ▶ Well-characterized foraging behaviour

- Model organism
 - ▶ Highly studied
 - ▶ Well-characterized foraging behaviour
 - ▶ Probably won't feel bad about simplifying its behaviour

- Model organism
 - ▶ Highly studied
 - ▶ Well-characterized foraging behaviour
 - ▶ Probably won't feel bad about simplifying its behaviour
- Social organism
 - ▶ Secretes signals to attract others nearby
 - ▶ Encourages “swarming” or “clumping”

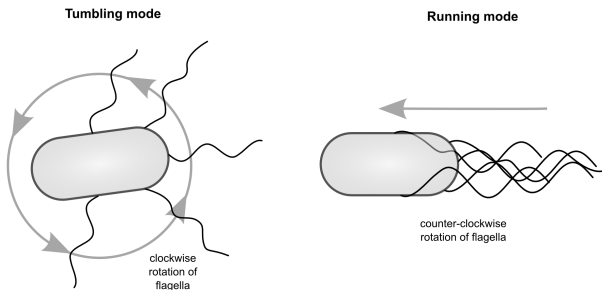
E. coli Behaviour

- Swims using left-handed helical flagella (“propellers”)



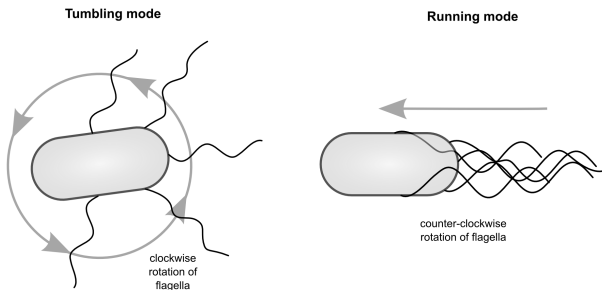
E. coli Behaviour

- Swims using left-handed helical flagella (“propellers”)
 - ▶ **Tumble:** flagella all rotate clockwise → pull on cell in all directions → random movement



E. coli Behaviour

- Swims using left-handed helical flagella (“propellers”)
 - ▶ **Tumble:** flagella all rotate clockwise → pull on cell in all directions → random movement
 - ▶ **Run:** flagella all rotate counterclockwise → flagella form a bundle → push on cell in one direction → directed movement



- If during a tumble *E. coli* swims down a nutrient concentration gradient:
 - ▶ Prolongs time spent on a run
 - ▶ Continues moving in the same direction

- If during a tumble *E. coli* swims down a nutrient concentration gradient:
 - ▶ Prolongs time spent on a run
 - ▶ Continues moving in the same direction
- Otherwise:
 - ▶ Tends to switch to a tumble (search for more)
 - ▶ Moves randomly while searching for more nutrient gradients to exploit

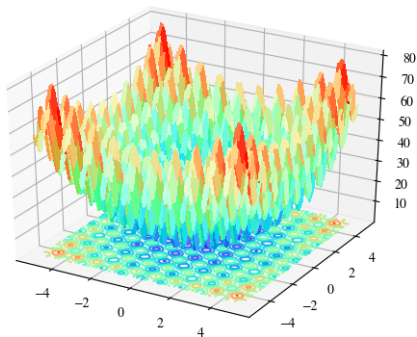
- If during a tumble *E. coli* swims down a nutrient concentration gradient:
 - ▶ Prolongs time spent on a run
 - ▶ Continues moving in the same direction
- Otherwise:
 - ▶ Tends to switch to a tumble (search for more)
 - ▶ Moves randomly while searching for more nutrient gradients to exploit
- Call a tumble followed by a run a “chemotaxis step”

Algorithm for a Single Bacterium

```
1: for  $j \leftarrow 1 \dots N_c$  do:  
2:    $J_{\text{last}} \leftarrow J(\theta)$   
3:    $\phi \sim S^p$   
4:    $\theta \leftarrow \theta + c\phi$   
5:   while  $J(\theta) < J_{\text{last}}$  do:  
6:      $J_{\text{last}} \leftarrow J(\theta)$   
7:      $\theta \leftarrow \theta + c\phi$ 
```

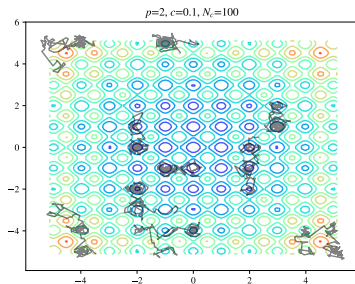
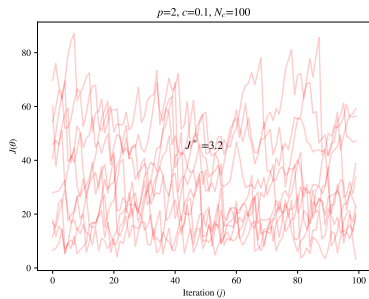
- θ : p -dimensional vector (randomly initialized)
- N_c : number of chemotaxis steps
- $\phi \sim S^p$: a random p -dimensional unit vector
- c : a step-size

Loss Function to Optimize



$$J(\theta) = An + \sum_{i=1}^n (x_i^2 - A \cos(2\pi x_i))$$

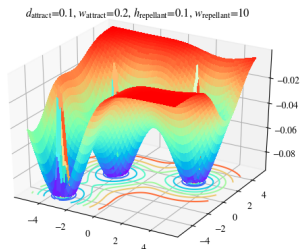
Results of Single Bacterium



J_{cc} and swarming behaviour

- *E. coli* do social foraging
- Secrete a substance to indicate to attract nearby *E. coli* and encourage swarming
- Also want to avoid crowding
- Use sum of two Gaussian functions to model this

J_{cc} and swarming behaviour



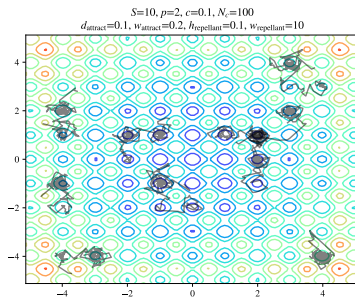
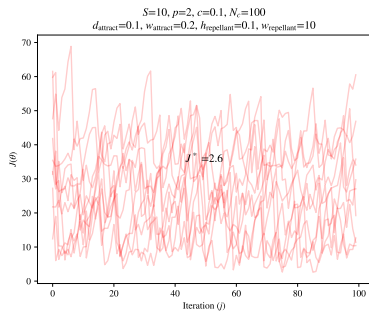
$$J_{cc}(\theta) = \sum_{i=1}^S -d_{\text{attract}} \exp(-w_{\text{attract}}(\theta - \theta_i)^T(\theta - \theta_i)) \\ + h_{\text{repellant}} \exp(-w_{\text{repellant}}(\theta - \theta_i)^T(\theta - \theta_i))$$

Algorithm for a Colony

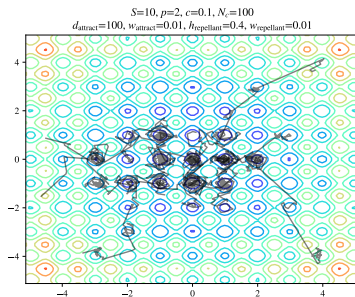
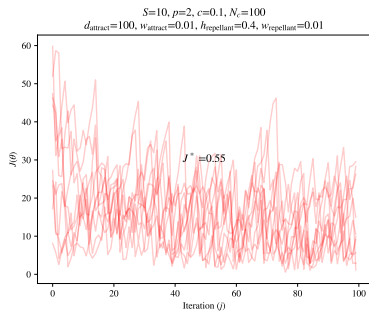
```
1: for  $j \leftarrow 1 \dots N_c$  do:
2:   for  $i \leftarrow 1 \dots S$  do:
3:      $J_{\text{last}} \leftarrow J(\theta_i) + J_{cc}(\theta_i)$ 
4:      $\phi \sim S^p$ 
5:      $\theta_i \leftarrow \theta_i + c_i \phi$ 
6:     while  $J(\theta_i) + J_{cc}(\theta_i) < J_{\text{last}}$  do:
7:        $J_{\text{last}} \leftarrow J(\theta_i)$ 
8:        $\theta_i \leftarrow \theta_i + c_i \phi$ 
```

- S : number of bacteria in the colony
- J_{cc} : cell-to-cell interactions

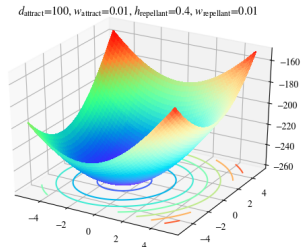
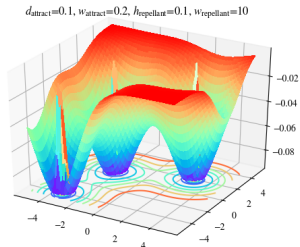
Results of Colony with Swarming



Results of Colony with Swarming



Comparing J_{cc}



E. coli reproduction

- *E. coli* “reproduce” via binary fission, which essentially produces a clone

E. coli reproduction

- *E. coli* “reproduce” via binary fission, which essentially produces a clone
- Individuals with higher values of J killed off

E. coli reproduction

- *E. coli* “reproduce” via binary fission, which essentially produces a clone
- Individuals with higher values of J killed off
- Individuals with lower values of J duplicated
 - ▶ Ideally move away due to repellence

E. coli reproduction

- *E. coli* “reproduce” via binary fission, which essentially produces a clone
- Individuals with higher values of J killed off
- Individuals with lower values of J duplicated
 - ▶ Ideally move away due to repellance
- Idea is to encourage searching in space nearby “best” individuals

E. coli reproduction

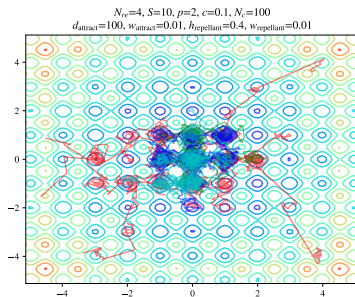
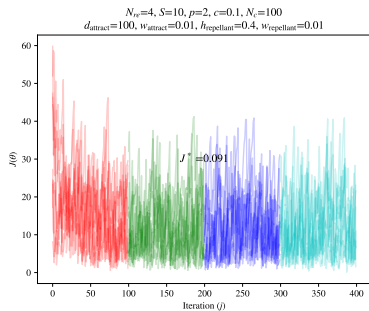
- *E. coli* “reproduce” via binary fission, which essentially produces a clone
- Individuals with higher values of J killed off
- Individuals with lower values of J duplicated
 - ▶ Ideally move away due to repellance
- Idea is to encourage searching in space nearby “best” individuals
- If repellance isn't high enough then repeated iterations of evolution can concentrate colony in local minimum

Algorithm for a Reproducing Colony

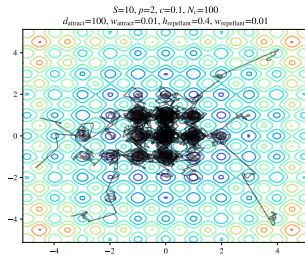
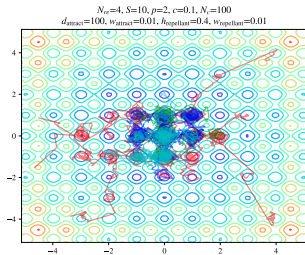
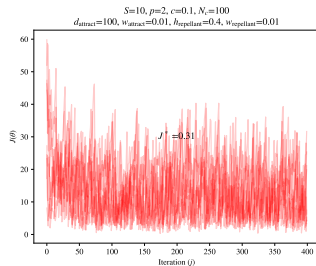
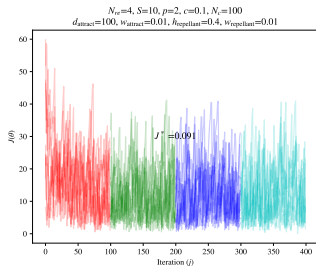
```
1: for  $k \leftarrow 1 \dots N_{re}$  do:
2:   for  $j \leftarrow 1 \dots N_c$  do:
3:     for  $i \leftarrow 1 \dots S$  do:
4:        $J_{\text{last}} \leftarrow J(\theta_i) + J_{cc}(\theta_i)$ 
5:        $\phi \sim S^p$ 
6:        $\theta_i \leftarrow \theta_i + c_i \phi$ 
7:       while  $J(\theta_i) + J_{cc}(\theta_i) < J_{\text{last}}$  do:
8:          $J_{\text{last}} \leftarrow J(\theta_i)$ 
9:          $\theta_i \leftarrow \theta_i + c_i \phi$ 
10:    delete worst  $S/2$  and reproduce best  $S/2$ 
```

- N_{re} : number of reproduction steps

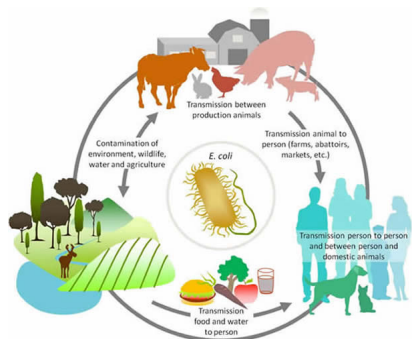
Results of Reproducing Colony



Does Reproduction Help?

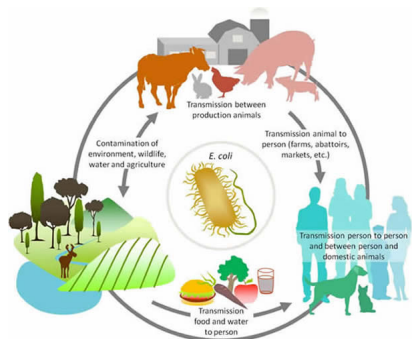


Elimination-Dispersal Events



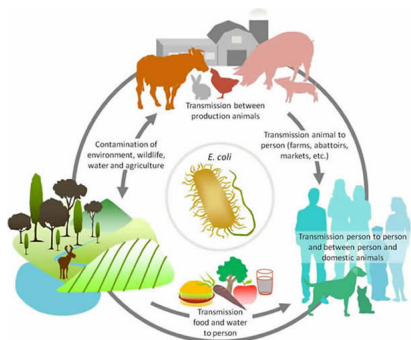
- Over time, random events disperse populations of *E. coli*
 - ▶ Water, animal activity, human intervention

Elimination-Dispersal Events



- Over time, random events disperse populations of *E. coli*
 - ▶ Water, animal activity, human intervention
- May destroy chemotactic progress
 - ▶ But may also bring *E. coli* to good food sources

Elimination-Dispersal Events



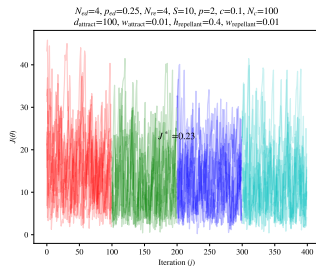
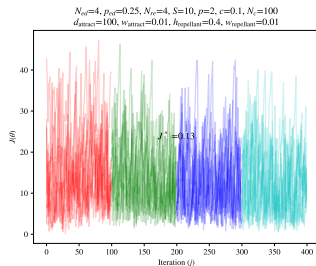
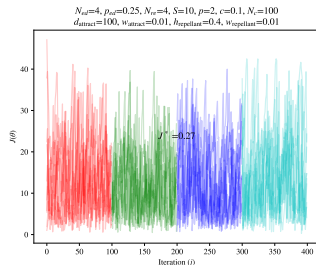
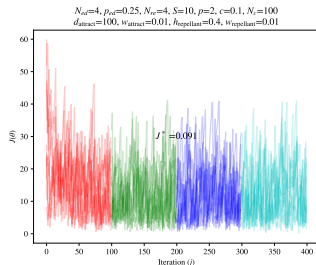
- Over time, random events disperse populations of *E. coli*
 - ▶ Water, animal activity, human intervention
- May destroy chemotactic progress
 - ▶ But may also bring *E. coli* to good food sources
- For optimization, this is a method to prevent stagnation and move out from local minima

Algorithm for a Dispersing Colony

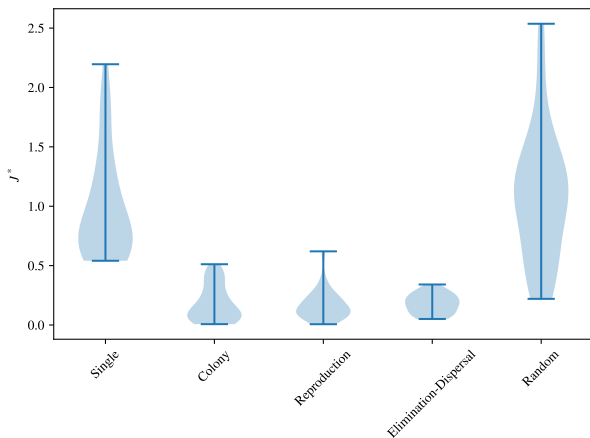
```
1: for  $l \leftarrow 1 \dots N_{ed}$  do:
2:   for  $k \leftarrow 1 \dots N_{re}$  do:
3:     for  $j \leftarrow 1 \dots N_c$  do:
4:       for  $i \leftarrow 1 \dots S$  do:
5:          $J_{\text{last}} \leftarrow J(\theta_i) + J_{cc}(\theta_i)$ 
6:          $\phi \sim S^p$ 
7:          $\theta_i \leftarrow \theta_i + c_i \phi$ 
8:         while  $J(\theta_i) + J_{cc}(\theta_i) < J_{\text{last}}$  do:
9:            $J_{\text{last}} \leftarrow J(\theta_i)$ 
10:           $\theta_i \leftarrow \theta_i + c_i \phi$ 
11:       delete worst  $S/2$  and reproduce best  $S/2$ 
12:   for  $i \leftarrow 1 \dots S$  do:
13:     if  $\epsilon \sim \mathcal{U}(0, 1) < p_{ed}$  then:
14:        $\theta_i \sim d^0(\theta)$ 
```

- N_{ed} : number of elimination-dispersal events
- p_{ed} : probability of a single elimination-dispersal event

Results of Elimination-Dispersal



Results of Elimination-Dispersal



Is this a good optimization algorithm?

Is this a good optimization algorithm?

- We don't know. The author doesn't compare to any existing methods.

Is this a good optimization algorithm?

- We don't know. The author doesn't compare to any existing methods.
- The author draws a comparison to genetic algorithms (GAs) since both are hill-climbing algorithms, but asserts that they are really different algorithms **based on their biological inspiration**

Is this a good optimization algorithm?

- We don't know. The author doesn't compare to any existing methods.
- The author draws a comparison to genetic algorithms (GAs) since both are hill-climbing algorithms, but asserts that they are really different algorithms **based on their biological inspiration**
- The method is also suspiciously similar to partical swarm optimization

Is this a good optimization algorithm?

- We don't know. The author doesn't compare to any existing methods.
- The author draws a comparison to genetic algorithms (GAs) since both are hill-climbing algorithms, but asserts that they are really different algorithms **based on their biological inspiration**
- The method is also suspiciously similar to partical swarm optimization
 - ▶ Combines local and global information with stochastic hill-climbing algorithm
 - ▶ The author never mentions this despite the popularity of PSOs (this paper publish 7 years later)

Is this a good optimization algorithm?

- We don't know. The author doesn't compare to any existing methods.
- The author draws a comparison to genetic algorithms (GAs) since both are hill-climbing algorithms, but asserts that they are really different algorithms **based on their biological inspiration**
- The method is also suspiciously similar to partical swarm optimization
 - ▶ Combines local and global information with stochastic hill-climbing algorithm
 - ▶ The author never mentions this despite the popularity of PSOs (this paper publish 7 years later)
 - ▶ (Ask me about the comparison I did afterwards!)

Is this a good optimization algorithm?

- However, this algorithm is **gradient-free**

Is this a good optimization algorithm?

- However, this algorithm is **gradient-free**
 - ▶ We can minimize functions that we may not have access to the gradient for (or it may not exist)
 - ▶ For example tuning hyperparameters
 - ▶ Or for example fitting the parameters of a neural network to perform a task

Is this a good optimization algorithm?

- However, this algorithm is **gradient-free**
 - ▶ We can minimize functions that we may not have access to the gradient for (or it may not exist)
 - ▶ For example tuning hyperparameters
 - ▶ Or for example fitting the parameters of a neural network to perform a task
- It can also explore the search space beyond the initial distribution $d^0(\theta)$ in case we do not know where the optimal value θ^* lies

Is this a good model of *E. coli*?

Is this a good model of *E. coli*?

- We don't know. The author doesn't compare to any ecological data.

Is this a good model of *E. coli*?

- We don't know. The author doesn't compare to any ecological data.
- Some important limitations from the author:

Is this a good model of *E. coli*?

- We don't know. The author doesn't compare to any ecological data.
- Some important limitations from the author:
 - ▶ Ignore characteristics of actual biological processes in favor of simplicity and capturing the essence of chemotactic hill-climbing and swarming

Is this a good model of *E. coli*?

- We don't know. The author doesn't compare to any ecological data.
- Some important limitations from the author:
 - ▶ Ignore characteristics of actual biological processes in favor of simplicity and capturing the essence of chemotactic hill-climbing and swarming
 - ▶ Ignore characteristics of the chemical medium and assume that consumption does not affect the nutrient surface

Is this a good model of *E. coli*?

- We don't know. The author doesn't compare to any ecological data.
- Some important limitations from the author:
 - ▶ Ignore characteristics of actual biological processes in favor of simplicity and capturing the essence of chemotactic hill-climbing and swarming
 - ▶ Ignore characteristics of the chemical medium and assume that consumption does not affect the nutrient surface
 - ▶ They assume a constant population size, even if there are many nutrients and generations

Is this a good model of *E. coli*?

- We don't know. The author doesn't compare to any ecological data.
- Some important limitations from the author:
 - ▶ Ignore characteristics of actual biological processes in favor of simplicity and capturing the essence of chemotactic hill-climbing and swarming
 - ▶ Ignore characteristics of the chemical medium and assume that consumption does not affect the nutrient surface
 - ▶ They assume a constant population size, even if there are many nutrients and generations
 - ▶ They assume that the cells respond to nutrients in the environment in the same way that they respond to ones released by other cells for the purpose of signaling the desire to swarm

Is this a good paper?

Is this a good paper?

Our objective is to explain how motile behaviors in both individual and groups of bacteria implement foraging and hence optimization.

to come up with a simple model that only represents certain aspects of the foraging behavior of bacteria.

Is this a good paper?

Our objective is to explain how motile behaviors in both individual and groups of bacteria implement foraging and hence optimization.

to come up with a simple model that only represents certain aspects of the foraging behavior of bacteria.

- It is not clear if the goal is create a simulation of *E. coli* foraging behaviour or to create another biologically-inspired optimization algorithm

Discussion

```
1 import numpy as np
2
3 def simulate(J, S=10, p=2, rng=(-5.12,5.12), c=0.1,
4             N_ed=2, p_ed=0.25,
5             N_re=4,
6             d_attract=0.1, w_attract=0.2, h_repellant=0.1, w_repellant=10,
7             N_c=100):
8
9     theta = rng[0] + np.random.rand(S, p)*(rng[1] - rng[0])
10
11     def J_cc(x):
12         result = 0
13         for theta_i in theta:
14             result += -d_attract*np.exp(-w_attract*((x-theta_i)**2).sum()) \
15                 + h_repellant*np.exp(-w_repellant*((x-theta_i)**2).sum())
16         return result
17
18     for l in range(N_ed):
19         for k in range(N_re):
20             for j in range(N_c):
21                 for i in range(S):
22                     phi = np.random.uniform(low=-1, high=1, size=p)
23                     phi /= np.linalg.norm(phi)
24                     J_last = J(theta[i]) + J_cc(theta[i])
25                     theta[i] = theta[i] + c*phi
26                     while J(theta[i]) + J_cc(theta[i]) < J_last:
27                         J_last = J(theta[i]) + J_cc(theta[i])
28                         theta[i] = theta[i] + c*phi
29
30                 I = np.argsort(J_histories[l, k].sum(axis=1))
31                 theta = np.concatenate((theta[I[:S//2]].copy(), theta[I[S//2:]].copy()))
32             for i in range(S):
33                 if np.random.rand() < p_ed:
34                     theta[i] = rng[0] + np.random.rand(p)*(rng[1] - rng[0])
```

Comparison to PSO

- Bacterial foraging method:
 - ▶ At least $N_{ed} \times N_{re} \times S \times N_c$ values of θ seen
- PSO: $N \times \text{iter}$ values of θ seen
 - ▶ PSO strongest with large N , so presume $\text{iter} \equiv N_c$
 - ▶ Then $N = N_{ed} \times N_{re} \times S$

Comparison to PSO

