Chapter 18

Cooperative Foraging and Search

Chapter Contents

18.1 Foraging Theory . . . 768
- 18.1.1 Elements of Foraging Theory . . . 768
- 18.1.2 Behavioral/Sensory Ecology of Search . . . 769
- 18.1.3 Cooperative/Social Foraging . . . 772
- 18.1.4 Nongradient Optimization Models . . . 775

18.2 Bacterial Foraging: *E. coli* . . . 776
- 18.2.1 Swimming and Tumbling via Flagella . . . 777
- 18.2.2 Bacterial Motile Behavior: Climbing Nutrient Gradients . . . 780
- 18.2.3 Underlying Sensing and Decision-Making Mechanisms . . . 782
- 18.2.4 Elimination and Dispersal Events . . . 783
- 18.2.5 Evolution of Bacteria . . . 783
- 18.2.6 Taxes in Other Swimming Bacteria . . . 784
- 18.2.7 Other Group Phenomena in Bacteria . . . 785

18.3 *E. coli* Bacterial Swarm Foraging for Optimization . . . 787
- 18.3.1 An Optimization Model for *E. coli* Bacterial Foraging . . . 788
- 18.3.2 Bacterial Foraging Optimization Algorithm . . . 792
- 18.3.3 Guidelines for Algorithm Parameter Choices . . . 794
- 18.3.4 Relations to Other Nongradient Optimization Methods . . . 795
- 18.3.5 Example: Function Optimization via *E. coli* Foraging . . . 796

18.4 Stable Social Foraging Swarms . . . 798
- 18.4.1 Example: The Biology of Honey Bee Swarms . . . 799
- 18.4.2 Swarm and Environment Models . . . 800
- 18.4.3 Stability Analysis of Swarm Cohesion Properties . . . 804
- 18.4.4 Cohesion Characteristics and Swarm Dynamics . . . 814

18.5 Design Example: Robot Swarms . . . 817
- 18.5.1 Robot Swarm Formulation . . . 817
- 18.5.2 Performance in Obstacle Avoidance and Noise Effects . . . 817
- 18.5.3 Additional Robot Swarm Design Challenges . . . 818

18.6 Exercises and Design Problems . . . 820

For many organisms, the survival-critical activity of foraging involves trying to find and consume nutrients in a manner that maximizes energy obtained from nutrient sources per unit time spent foraging, while at the same time minimizing exposure to risks from predators. If the organism has a decision-making mechanism (e.g., a brain), then we can view this mechanism as the controller and the remainder of the organism and environment as the "plant" (process to be controlled). "Decisions" involve where and when to move, what to eat, and so on. Decision heuristics, planning, attention, and learning can all play a role in foraging (if the organism is endowed with such capabilities). Moreover, evolution can be viewed as a process that optimizes the foraging strategy; it redesigns (tunes) all supporting physiology and the foraging decision-making strategy.

In this chapter, we outline the foundations of foraging and basic concepts related to how animals work together to "socially" forage. We do this by modeling the foraging process of a species of bacteria as optimization processes. You will see that they use elements of gradient-type search (e.g., approximations of gradients) without relying on explicit gradient information. Hence, foraging methods naturally build on the gradient optimization methods of Part III. However, they also incorporate evolutionary aspects, and are nongradient methods, so they provide a natural bridge between the optimization methods of Part III and Part IV, by giving ideas for how the various types of optimization algorithms can be merged (e.g., viewing learning as gradient-based and occurring over a lifetime, and evolution as nongradient, population-based, and occurring over long time epochs). You will also see relationships between the foraging algorithms and the pattern search and SPSA methods of Part IV. Later in the chapter, we show how to model swarms, characterize their cohesiveness as a stability property, and provide conditions under which they will converge to maintain a cohesive group. In simulation, we will study how stable swarms forage. Moreover, we show how a cooperative robotics problem of guiding a group of robots around some obstacles to a goal position in a factory can be formulated and solved via the ideas from foraging swarms.

Broadly speaking, you could simply view this chapter and the next as building on all the earlier chapters in the sense that the focus here is on the development of control strategies for guidance of an organism (e.g., the last section of this chapter will show a firm connection to Chapter 6, where we study path planning for obstacle avoidance by an autonomous vehicle) or a group of organisms. Here, however we add details on how specific animals forage (i.e., how they solve the control problem of guiding themselves successfully through their environment) and the relevant connections to optimization theory. We study the case of cooperative foraging, where animals work together in this chapter, and in the next, we study the case where they compete. Moreover, the final section in the next chapter serves to challenge the reader to develop an intelligent foraging team, and to do this, many of the concepts developed in this and the next chapter will be quite useful, not to mention the rest of the book.

18.1 Foraging Theory

In this section we outline the basic principles of foraging theory and highlight aspects of social foraging where animals cooperate in foraging activities.

18.1.1 Elements of Foraging Theory

A large portion of foraging theory is based on the assumption that animals search for and obtain nutrients in a way that maximizes their energy intake E per unit time T spent foraging. Hence, some animals try to maximize a function like

Foraging is an optimization process created by evolution.

$$\frac{E}{T}$$

Maximization of such a function gives them the nutrient sources to survive and additional time for other important activities (e.g., fighting, fleeing, mating, reproducing, sleeping, or shelter-building). Other possible currencies may be used in foraging such as "energetic efficiency," which is the net energy gained, divided by the energy invested to get it. Some animals seem to switch between optimizing the net rate of energy gain E/T to optimizing energetic efficiency. Sometimes variance in energetic gain is the primary variable that drives foraging behavior (e.g., when there is a need to meet a daily energetic intake requirement before nightfall), while other times, predation is a significant factor. Regardless, most biologists argue that animals seek to optimize some variable that is correlated with fitness.

Clearly, foraging characteristics can be very different for different species. Herbivores generally find food easily, but must eat a lot. Carnivores generally find it difficult to find food, but do not have to eat as much, since their food is of high energy value. Some other activities are related to foraging. For instance, seeking favorable environments and avoiding harmful ones (e.g., finding shelter from the weather), or searching for a suitable mate, are both related to foraging.

The "environment" of the animal establishes the pattern of nutrients that are available (e.g., via what other organisms and nutrients are available, constraints such as rivers and mountains, and weather patterns), and it places constraints on obtaining that food (e.g., small portions of food may be separated by large distances). It also affects the availability of resources (e.g., weather). Some foragers have a search rhythm (e.g., daily, nightly, etc.), but others forage opportunistically and, depending on their needs, independent of such a rhythm. During foraging there can be risks due to predators, the prey may be mobile so it must be chased, and the physiological characteristics of the forager constrain its capabilities and ultimate success. Often, in biology, researchers think of evolution as having optimized the foraging behavior of a species for an ecological niche and within its physiological constraints (which may change due to evolution also).

For some animals, there are multiple prey types that could be chosen, and the choice may depend on its diet, the abundances of the prey types, and how easy they are to find. For other animals, nutrients are distributed in "patches" (e.g., a lake, a meadow, a bush with berries, a group of trees with fruit). Foraging involves finding such patches, deciding whether to enter a patch and search for food, and whether to continue searching for food in the current patch or to find another patch that, hopefully, has a higher quality and quantity of nutrients than the current patch. Patches or prey are generally encountered sequentially and sometimes great effort and risk is needed to travel from one patch or prey to another. Some patches of food or prey naturally appear and disappear, and appearance can "trigger" certain foraging behavior (as can hunger). Generally, if an animal encounters a nutrient-poor patch or undesirable prey, but based on past experience it expects that there should be a better patch or prey, then it will consider risks and efforts to find another patch or prey and, if it finds them acceptable, it will seek another patch or prey. Also, if an animal has been in a patch for some time, it can begin to deplete its resources, so there should be an optimal time to leave the patch and venture out to try to find a richer one. It does not want to waste resources that are readily available, but it also does not want to waste time in the face of diminishing energy returns.

Decision-making in foraging is a control strategy for organism guidance.

Optimal foraging theory formulates the foraging problem as an optimization problem and via computational or analytical methods, can provide an optimal foraging "policy" that specifies how foraging decisions are made. There are quantifications of what foraging decisions must be made, measures of "currency" (the opposite of cost), and constraints on the parameters of the optimization. For instance, researchers have studied how to maximize long-term average rate of energy intake, where only certain decisions and constraints are allowed. Constraints due to incomplete information (e.g., due to limited sensing capabilities), predation, and risks (e.g., due to predators) have been considered.

Here, the interesting fact is that the foraging problem can be formulated as an optimization problem that results in an optimal decision policy, if the optimization problem can be solved (traditionally, dynamic programming formulations have been used [490]). Essentially, these optimization approaches seek to construct an optimal controller (policy) for making foraging decisions. Some biologists have questioned the validity of such an approach, arguing that no animal can make optimal decisions. However, the optimal foraging formulation is only meant to be a model that explains what optimal behavior would be like. Biologists have shown that foraging decision heuristics are used very effectively by animals to approximate optimal policies, given the physiological (and other) constraints that are imposed on the animal. Such an approach is quite rational, even in engineering applications. In the construction of a computer decision-making system, dynamic programming is sometimes found to be impractical due to computational complexity issues. Heuristics and approximations are then sometimes used to try to provide a suboptimal solution, but one that is as good as possible, given the available computing resources.

18.1.2 Behavioral/Sensory Ecology of Search

Foraging has been studied for many years from both experimental and theoretical perspectives. One aspect of foraging that is particularly relevant here is the search strategy (an optimization process) that is employed in finding food. Here, we outline some relevant theory pertaining to search strategies for foraging. Later, we briefly discuss how nongradient optimization methods might be useful in modeling some search strategies in foraging.

Cruise, Saltatory, and Ambush Search

In one approach to the study of foraging search strategies [390], predation is broken into components that are similar for many animals. First, predators must search for and locate prey. Next, they pursue and attack the prey. Finally, they handle and ingest the prey. The importance of various components of the foraging behavior depends on the relationship between the predator and the prey. If the prey is larger than the predator, then the pursuit, attack, and handling can be most important. The prey may be easy to find, but the prey's size gives it an advantage. If the prey are smaller than the predator, then generally the search component of foraging is most important. Small size can be an advantage for the prey. Since prey are often smaller than predators, for many animals they must be consumed often and in large numbers; this makes the search time limit other components of the predation cycle. Here, suppose we consider cases where the searching behavior is the dominant factor in foraging. This is the case for foragers, such as many birds, fish, lizards, and insects.

Animals use a variety of search strategies in foraging that are each optimization methods.

Some animals are "cruise" or "ambush" searchers. For the cruise approach to searching, the forager moves continuously through the environment constantly searching for prey at the boundary of the area being searched (tuna fish and hawks are cruise searchers). In ambush search, the forager sits and waits for prey to cross into strike range (rattlesnakes are ambushers). The search strategies of many species are actually in between the cruise and ambush extremes. In particular, in "saltatory search strategies," an animal will intermittently cruise and sit and wait, possibly changing direction at various times when it stops and possibly while it moves. To envision this strategy, consider Figure 18.1, where distance traveled while searching is plotted versus time. In cruise search, distance increases at a constant rate dictated by how fast the animal moves in search. At the other extreme, the ambusher sits and waits for a long time, then makes a move to try to obtain a prey. In between, there are many possible saltatory search strategies that are based on an alternating sequence of cruising and waiting. Many animals' foraging strategies seem to lie somewhere on the continuum between ambush and cruise, and hence, are saltatory search strategies.

While cruisers tend to search at the boundary of the search space and ambushers stay in one place, saltatory searchers generally move throughout the search space. Saltatory search can be adjusted to suit the environment by changing rates of movement during cruises, and the lengths of cruises and waits. For instance, some fish are known to pause more briefly and swim farther and faster during repositioning when searching for large prey compared to small ones. This is consistent with foraging theory in that the fish is willing to spend more effort to obtain more food (energy).

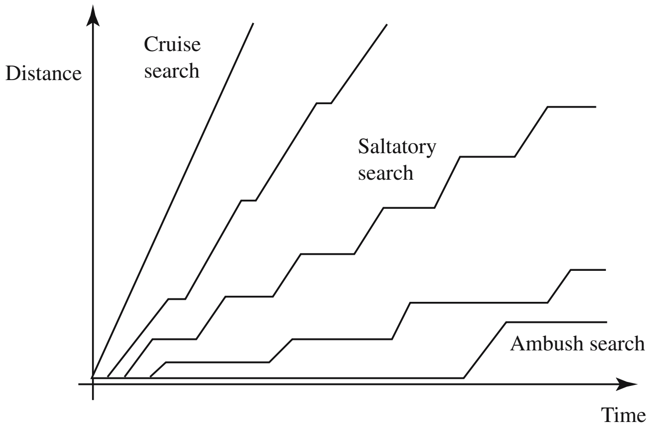

Figure 18.1: Illustration of the range of search strategies for foraging animals (figure adapted from [390], © Sigma Xi, The Scientific Research Society, and used with permission).

Scanning and Repositioning Relationships

Studies of foraging for some fish have shown that they do not search for prey while they are moving, but only during the stationary pause between repositioning moves. They stop and look around. Hence, the repositioning moves serve only to move the fish into regions where they have not looked before. Generally, if the animal searches only during pauses, then the repositioning moves (length and direction) depend on the sensing capabilities of the animal. For instance, consider Figure 18.2, where, at the top, the animal is imagined to be at the center of the circle which represents a local scan range. Suppose that for this particular animal, it can search in the pie-shaped region that is shaded. (Other animals have different shaped regions, some that almost fill the entire circle.) How large should the repositioning move be? As illustrated, if the move is too short, then a significant portion of the search space is searched again after the move and this is generally a waste of resources. If the move is too large, then there is no overlap and there can be some part of the space that is not searched, representing possible missed opportunities. For some intermediate length moves, there will be some overlap but not too much search space ignored. As illustrated on the bottom, changes in direction are dependent on the shape of the scan area also. Finally, note that as the geometry of the shaded region changes (e.g., the pie-shaped region becomes a larger piece of the pie), then identical intermediate-sized moves result in larger overlap of the search space (draw a figure and convince yourself of this). It has been shown that some fish choose repositioning moves so as to maximize net energy gain, consistent with basic ideas in foraging theory.

In many species the pauses are used for orienting the animal toward prey. That is, they stop and change their direction based on their scan information.

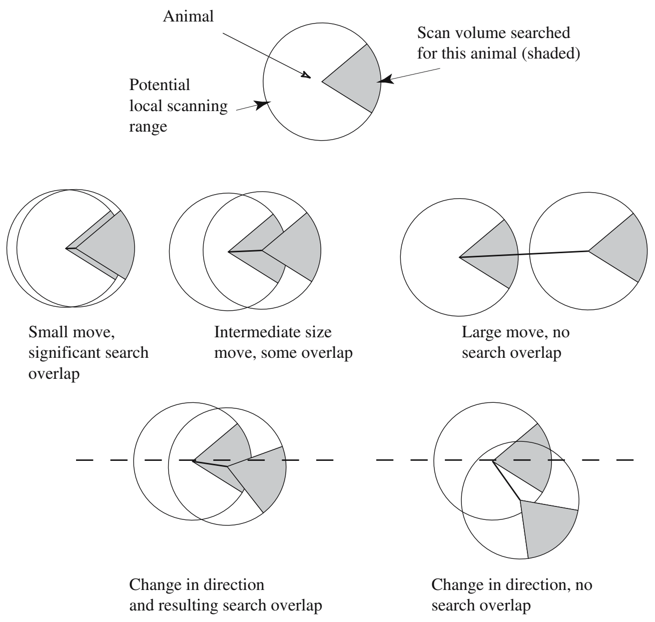
Animal
Scan volume searched for this animal (shaded)
Potential local scanning range
Small move, significant search overlap
Intermediate size move, some overlap
Large move, no search overlap
Change in direction and resulting search overlap
Change in direction, no search overlap

Figure 18.2: Illustration of trade-offs between repositioning and scanning; trying to scan everywhere and not rescan already visited areas (figure adapted from [390], © Sigma Xi, The Scientific Research Society, and used with permission).

Then, for instance, if there is not an abundance of prey, in some species of fish, fewer pursuits follow pauses. Also, in some fish, the length of the stop and wait generally decreases when they are looking for large, easily located prey. Often as the difficulty of the search increases, the pauses get longer. In environments where there are few prey, the fish persistently search.

Finally, note that there are also effects of likelihood and frequency of encounter of prey that would influence repositioning and direction-changing behavior. Effects of risk of moving (e.g., from some predator) should also be taken into consideration. All these aspects can result in the animal dynamically adjusting its saltatory search strategy.

18.1.3 Cooperative/Social Foraging

The foraging concepts discussed above were for individual animals. Foragers, however, live in environments with both a biotic and abiotic part, so a more complete formulation includes the other foragers in the environment. There can be advantages to group cooperative (or "social") foraging. Some method of sharing information is necessary for cooperative foraging. The shared information could come in many forms. In humans, this could include language. In other animals, it might be certain movements, noises, or "trail-laying" mechanisms. Such information is in the form of cues or signals. Shared information could also arise via possession of shared genetic material.

The advantages of group foraging include:

Group foraging requires group search strategies and hence, distributed cooperation within the group.

- The per capita rate of energetic gain for each animal may be higher if the animal is in a group. This can be the case even if the gains are not split evenly in the group.
- In some cases, the net rate of energetic gain of each individual may go down if it joins a group. However, it may still be a good strategy to join a group if the variance in the rate of energetic gain is lower. Sometimes, when there are more animals searching for nutrients, the likelihood of finding nutrients may increase. When one animal finds some nutrients, it can tell others in the group where the nutrients are. You may think of joining a group as gaining access to an "information center" for helping with survival.
- There may be increased capabilities to cope with larger prey. The group can "gang up" on a large prey and kill and ingest it, while a single small predator may not be able to do this.
- There may be protection from predators that can be provided by members of the group (e.g., in some species of birds, the members in the middle of the group are protected by the ones at the edges).
- In some cases, phenotypic diversity is profitably exploited by groups to produce highly efficient coordinated group behavior (e.g., for some ants and bees).

Sometimes it is useful to think of a group of animals as a *single* living creature, where via grouping, each individual essentially gains additional physiological capabilities that help it to succeed in foraging (and the gains may offset the possibility of food-competition problems in groups). Some call this the "super-organism" viewpoint.

For group foraging, you may think of how a pack of wolves hunts, or a flock of birds, colony of ants, or school of fish behave. Connections between optimization, engineering applications, and foraging behavior of colonies of ants have been studied [73]. There, it is explained how colonies of ants can solve shortest path problems, minimum spanning tree problems, and traveling salesperson problems (all combinatorial optimization problems) among other engineering applications. (The resulting computer algorithms are called, for instance, "ant colony optimization" algorithms.) These ants use "indirect" communications called "stigmergy," where one ant can modify its environment and later, another ant can change its behavior due to that modification. For instance, if an ant goes out foraging, it may search far and wide in a relatively random pattern; however, once it finds a food source, it goes back to the ant hill, laying a trail of "pheromone" (that can evaporate, but normally stays in place, possibly up to several months). Then, when other ants go out foraging, they tend to follow the pheromone trail and find food more easily. You can then think of the first ant as having "recruited" additional foragers and the trails as a type of memory for the whole ant colony (i.e., using communications and working together, they gain the important physiological capability of learning). Viewed as a superorganism, you may even detect elements of planning. Communications, memory, and learning, result in more efficient foraging for the group. Other social insects use other communication methods. For instance, after successful foraging, a bee will come back to the hive and communicate the profitability and location of the food source via a "dance." These dances then proportionally recruit foragers based on forage site profitability, and the result is a dynamic proportioning of foragers across a wide area.

Social behavior enables what can be thought of as higher-level cognitive functions of the group.

To be more concrete about the connections between foraging and optimization, consider Figure 18.3. There, initially boxes 1 and 2 hold two colonies of the Argentine ant *Iridomyrmex humilis.* These colonies interact via ants traveling between the two colonies. Since trail densities on the shorter path tend to grow faster, more and more ants tend to choose that path and thereby avoid the longer path (the figure shows ants traveling after they have, as a group, found the shortest path). A similar behavior is found when one box holds a colony of ants and the other holds a food source [25] and this illustrates that in foraging, the ants work together to find the shortest path to food sources.

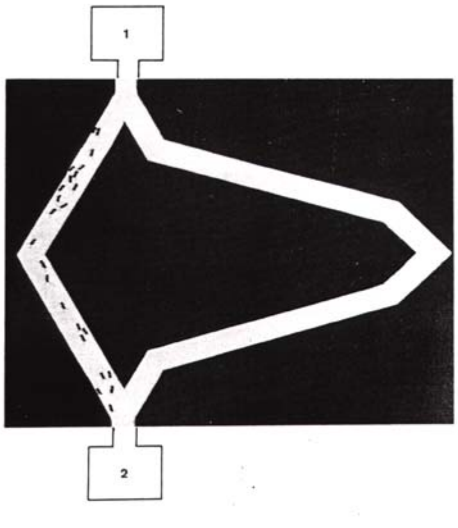

Figure 18.3: Experiment showing that ants will select a shortest path between two colonies (figure taken from [25], © Springer-Verlag GmbH and Co., and used with permission).

Finally, while we discuss "intelligent foraging" in more detail in Section 19.6, we briefly note that the individual characteristics of the animal can significantly affect its success in foraging. If an individual forager can pay *attention* to the critical parts of the environment and *learn* about the environment (e.g., by developing and storing a "cognitive map") and characteristics of its prey, then it can probably increase its foraging success. If, based on such learned information it can *plan* its foraging, then it may gain further increases in efficiency. Furthermore, if groups of foragers can learn and plan their activities together, it is possible that even greater success might be obtained. Indeed, it seems that humans often act as group foragers that can collectively learn and plan. We can think of many individuals working together to achieve something that is unachievable by any individual.

In the next section, we will consider individual and group foraging in bacteria, organisms that are much more simple than an ant or human, yet which can still work together for the benefit of the group. First, however, we explain some connections to nongradient optimization.

18.1.4 Nongradient Optimization Models

Before we turn to specific foraging models, it is important to point out that there are close connections to some of the optimization methods considered earlier in this book, both gradient and nongradient methods. First, note that it is impossible for most animals (e.g., bacteria) to know an analytical expression for the gradient of a nutrient concentration profile (i.e., a mathematical expression for how the nutrient concentration will change as the bacterium makes small changes in its position). This is both because it does not have the memory to store it, and also due to the high level of uncertainty about the environment it lives in (e.g., time-varying and stochastic effects). Moreover, even with sufficient physiology for remembering an analytical gradient, in general, it is impossible for any animal (besides perhaps a human) to know an analytical form for the gradient of the surface being searched (e.g., one that represented food locations, predators, risk, etc.) with respect to its location in the search domain. Animals sequentially decide where to explore, and in doing so they encounter new parts of a search domain, and the environment has significant random effects. In foraging, animals conduct an optimization process without use of an analytical expression for the gradient and hence, we say that they perform nongradient optimization or "search."

Nongradient optimization models can form a set of tools for modeling social foraging.

Motivated by the earlier sections, and studies on search strategies for foraging, in this section, we discuss several "conventional" (i.e., not biologically motivated) deterministic and stochastic approaches to perform optimization without the use of analytical gradient information or measures of the gradient. As you read about the methods, it will be useful to draw analogies with the basic search mechanisms of the bacteria discussed later in this chapter. For instance, in one way or another, most nongradient methods use measurements of the cost function and form approximations to the gradient to decide which direction to move. (Some of these are what might be called "regional" approximations, since they use a pattern of points over possibly a large region to provide a gross approximation to the gradient.) In the context of foraging, you can then think of the process of obtaining measurements and deciding where to move (i.e., the steps of the algorithms we cover in this section) in one of three following ways:

1. You can view the taking of a measurement of the cost function as a single forager going to the location in the search domain and taking a single measurement by using, for example, its vision to assess what food is at that location (or the likelihood that it is there). The forager may make several such local "exploratory" moves from its current position before it tries to move in what it considers to be generally a good direction to find food. (Of course, it may get lucky and get something worth stopping and eating in this sampling process.)

2. You can think of the forager as being at a single location, taking several local measurements at locations in the search domain, processing these, and then deciding which direction to move. We think of the forager as having sensors that it can focus on different nearby regions of the optimization domain, and only moving after it has scanned its environment. Some lower life forms (e.g., *E. coli*) cannot sense at a distance; they must go to a position to find out if there is food there. Higher life forms can generally sense at a distance and this saves them the energy needed to travel to every position to find out if food is there.

3. You can view the algorithm as modeling a social foraging process, where there are a finite number of foragers who each make measurements of the cost function and then via communications decide how to move the entire group of foragers. For example, in the "pattern search methods" of the last part, we may think of each point in the pattern of (local) cost function measurements as representing a location of a forager. We think of the group as having a capability to communicate how well they are doing to all the other members of the group, and come to an agreement on which is the best way to move the group to ensure foraging success.

In a sense, you may ask yourself if biology mimics any of the conventional strategies; is there an animal that forages according to a particular pattern search method? If so, why did evolution create this search strategy? What features of the environment drove the creation of the strategy?

18.2 Bacterial Foraging: *E. coli*

The *E. coli* bacterium is shown in Figure 18.4. It has a plasma membrane, cell wall, and capsule that contain, for instance, the cytoplasm and nucleoid. The pili (singular, pilus) are used for a type of gene transfer to other *E. coli* bacteria, and flagella (singular, flagellum) are used for locomotion. (Only one is shown, but in the actual cell there are as many as six.) The cell is about $1\mu m$ in diameter, and $2\mu m$ in length. The *E. coli* cell only weighs about 1 picogram, and is composed of about 70% water. *Salmonella typhimurium* is a similar type of bacterium.

Think of the E. coli *as a small underwater vehicle, a nanotechnology.*

The *E. coli* bacterium is probably the best understood microorganism. Its entire genome has been sequenced; it contains 4,639,221 of the A, C, G, and

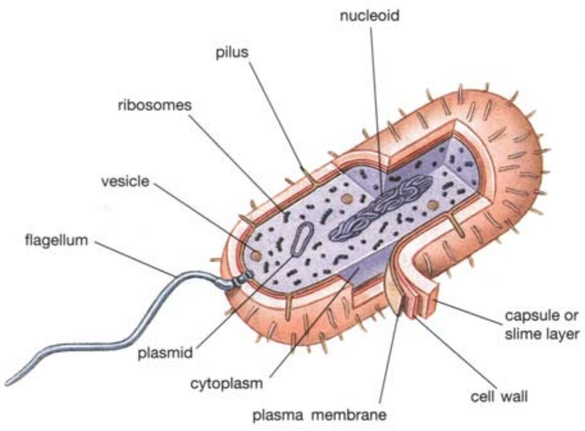

Figure 18.4: *E. coli* bacterium (figure taken from [34], © Pearson Education Inc., used with permission).

T "letters"—adenosine, cytosine, guanine, and thymine—arranged into a total of 4,288 genes. When *E. coli* grows, it gets longer, then divides in the middle into two "daughters." Given sufficient food and held at the temperature of the human gut (one place where they live) of 37 deg. C, *E. coli* can synthesize and replicate everything it needs to make a copy of itself in about 20 min.; hence, growth of a population of bacteria is exponential with a relatively short "time to double" the population size. For instance, following [61], if at noon today you start with one cell and sufficient food, by noon tomorrow there will be $2^{72} = 4.7 \times 10^{21}$ cells, which is enough to pack a cube 17 meters on one side. (It should be clear that with enough food, at this reproduction rate, they could quickly cover the entire earth with a knee-deep layer!)

The *E. coli* bacterium has a control system that enables it to search for food and try to avoid noxious substances (the resulting motions are called "taxes"). For instance, it swims away from alkaline and acidic environments, and towards more neutral ones. To explain the motile behavior of *E. coli* bacteria, we will explain its actuator (the flagella), "decision-making," sensors, and closed-loop behavior (i.e., how it moves in various environments—its "motile behavior"). You will see that *E. coli* perform a type of "saltatory search," a concept that is discussed in Section 18.1.2.

18.2.1 Swimming and Tumbling via Flagella

Locomotion is achieved via a set of relatively rigid flagella that enable it to "swim" via each of them rotating in the same direction at about $100 - 200$ revolutions per second (in control systems terms, we think of the flagella as providing for actuation). Each flagellum is a left-handed helix configured so that as the base of the flagellum (i.e., where it is connected to the cell) rotates counterclockwise, as viewed from the free end of the flagellum looking towards the cell, it produces a force against the bacterium so it pushes the cell. You may think of each flagellum as a type of propeller. If a flagellum rotates clockwise, then it will pull at the cell. From an engineering perspective, the rotating shaft at the base of the flagellum is quite an interesting contraption that seems to use what biologists call a "universal joint" (so the rigid flagellum can "point" in different directions, relative to the cell). In addition, the mechanism that creates the rotational forces to spin the flagellum in either direction is described by biologists as being a biological "motor" (a relatively rare contraption in biology even though several types of bacteria use it) as shown in Figure 18.5. The motor is quite efficient in that it rotates a complete revolution using only about 1000 protons and thereby *E. coli* spends less than 1% of its energy budget for motility.

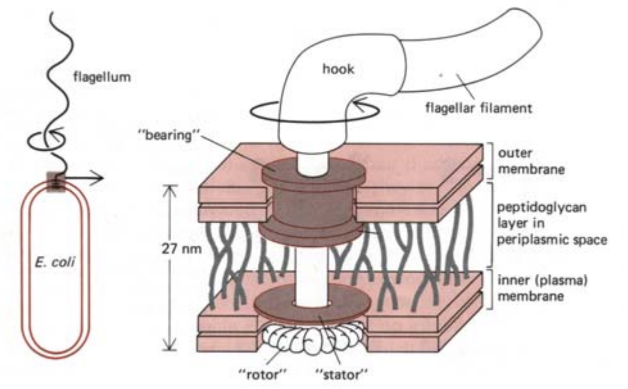

Figure 18.5: *E. coli* bacterium, flagellar connection, and biological "motor" (figure taken from [8], © Garland Science/Taylor and Francis Books, Inc., used with permission).

An *E. coli* bacterium can move in two different ways: it can "run" (swim for a period of time) or it can "tumble," and it alternates between these two modes of operation its entire lifetime (i.e., it is rare that the flagella will stop rotating). First, we explain each of these two modes of operation. Following that, we will explain how it decides how long to swim before it tumbles.

If the flagella rotate clockwise, each flagellum pulls on the cell and the net effect is that each flagellum operates relatively independent of the others and so the bacterium "tumbles" about (i.e., the bacterium does not have a set direction of movement and there is little displacement). See Figure 18.6(a). To tumble after a run, the cell slows down or stops first; since bacteria are so small they experience almost no inertia, only viscosity, so that when a bacterium stops swimming, it stops within the diameter of a proton. Call the time interval during which a tumble occurs a "tumble interval." Under certain experimental conditions (an isotropic, homogeneous medium—one with no nutrient or noxious substance gradients) for a "wild type" cell (one found in nature), the mean tumble interval is about 0.14 ± 0.19 sec. (mean $\pm$ standard deviation, and it is exponentially distributed) [61, 62]. After a tumble, the cell will generally be pointed in a random direction, but there is a slight bias toward being placed in a direction it was traveling before the tumble.

There are two modes of operation of the actuator: one leading to swimming and the other to tumbling.

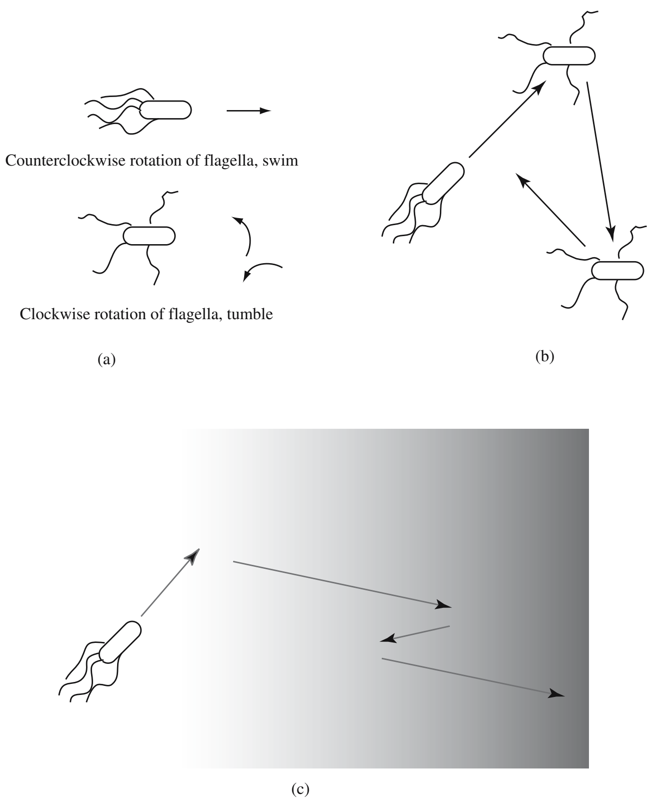
Counterclockwise rotation of flagella, swim
Clockwise rotation of flagella, tumble
(a)
(b)
(c)

Figure 18.6: Bundling phenomenon of flagella shown in (a), swimming and tumbling behavior of the *E. coli* bacterium is shown in (b) in a neutral medium and in (c) where there is a nutrient concentration gradient, with darker shades indicating higher concentrations of the nutrient. (Note: Relative sizes of the bacteria and lengths of runs are not to scale.)

If the flagella move counterclockwise, their effects accumulate by forming a "bundle" (it is thought that the bundle is formed due to the viscous drag of the medium) and hence, they essentially make a "composite propeller" and push the bacterium so that it runs (swims) in one direction (see Figure 18.6(a)). On a run, bacteria swim at a rate of about $10 - 20$ μmeters/sec., or about 10 body lengths per second (assuming the faster speed and an *E. coli* that is 2 μ meters long, a typical length), but in a rich medium they can swim even faster

[335]. This is a relatively fast rate for a living organism to travel; consider how fast you could move through water if you could swim at 10 of your body lengths per second. (You would certainly win a gold medal in the Olympics!) Call the time interval during which a run occurs the "run interval." Under certain experimental conditions (an isotropic, homogeneous medium—the same as the one mentioned above) for a wild type cell, the mean run interval is about 0.86 ± 1.18 sec. (and it is exponentially distributed) [61, 62]. Also, under these conditions, the mean speed is 14.2 ± 3.4 $\mu m/sec$. Runs are not perfectly straight since the cell is subject to Brownian movement that causes it to wander off course by about 30 deg. in 1 sec. in one type of medium, so this is how much it typically can deviate on a run. In a certain medium, after about 10 sec. it drifts off course more than 90 deg. and hence, essentially forgets the direction it was moving [61]. Finally, note that in many bacteria, the motion of the flagella can induce other motions, e.g., rotating the bacteria about an axis.

18.2.2 Bacterial Motile Behavior: Climbing Nutrient Gradients

The motion patterns (called "taxes") that the bacteria will generate in the presence of chemical attractants and repellents are called "chemotaxes." For *E. coli*, encounters with serine or aspartate result in attractant responses, while repellent responses result from the metal ions Ni and Co, changes in pH, amino acids like leucine, and organic acids like acetate. What is the resulting emergent pattern of behavior for a whole group of *E. coli* bacteria? Generally, as a group they will try to find food and avoid harmful phenomena, and when viewed under a microscope, you will get a sense that a type of intelligent behavior has emerged, since they will seem to intentionally move as a group (analogous to how a swarm of bees moves).

Simple control rules are used to decide whether to swim or tumble.

To explain how chemotaxis motions are generated, we simply must explain how the *E. coli* decides how long to run since, from the above discussion, we know what happens during a tumble or run. First, note that if an *E. coli* is in some substance that is neutral, in the sense that it does not have food or noxious substances, and if it is in this medium for a long period of time (e.g., more than one minute), then the flagella will simultaneously alternate between moving clockwise and counterclockwise so that the bacterium will alternately tumble and run. This alternation between the two modes will move the bacterium, but in random directions, and this enables it to "search" for nutrients (see Figure 18.6(b)). For instance, in the isotropic homogeneous environment described above, the bacteria alternately tumble and run with the mean tumble and run lengths given above, and at the speed that was given. If the bacteria are placed in a homogeneous concentration of serine (i.e., one with a nutrient but no gradients), then a variety of changes occur in the characteristics of their motile behavior. For instance, mean run length and mean speed increase and mean tumble time decreases. They do, however, still produce a basic type of searching behavior; even though it has some food, it persistently searches for more. As an example of tumbles and runs in the isotropic homogeneous medium described above, in one trial motility experiment lasting 29.5 sec., there were 26 runs, the maximum run length was 3.6 sec., and the mean speed was about 21 $\mu m/sec.$ [61, 62].

Next, suppose that the bacterium happens to encounter a nutrient gradient (e.g., serine) as shown in Figure 18.6(c). The *change* in the concentration of the nutrient triggers a reaction such that the bacterium will spend more time swimming and less time tumbling. As long as it travels on a positive concentration gradient (i.e., so that it moves towards increasing nutrient concentrations) it will tend to lengthen the time it spends swimming (i.e., it runs farther). The directions of movement are "biased" towards increasing nutrient gradients. The cell does not change its *direction* on a run due to changes in the gradient—the tumbles basically determine the direction of the run, aside from the Brownian influences mentioned above.

The bacterial foraging behavior results in a type of hill-climbing optimization algorithm.

On the other hand, typically if the bacterium happens to swim down a concentration gradient (or into a positive gradient of noxious substances), it will return to its baseline behavior so that essentially it tries to search for a way to climb back up the gradient (or down the noxious substance gradient). For instance, under certain conditions, for a wild-type cell swimming up serine gradients, the mean run length is 2.19 ± 3.43 sec., but if it swims down a serine gradient, mean run length is 1.40 ± 1.88 sec. [62]. Hence, when it moves up the gradient, it lengthens its runs. The mean run length for swimming down the gradient is the one that is expected, considering that the bacteria are in this particular type of medium; they act basically the same as in a homogeneous medium so that they are engaging their search/avoidance behavior to try to climb back up the gradient.

Finally, suppose that the concentration of the nutrient is constant for the region it is in, after it has been on a positive gradient for some time. In this case, after a period of time (not immediately), the bacterium will return to the same proportion of swimming and tumbling as when it was in the neutral substance so that it returns to its standard searching behavior. It is never satisfied with the amount of surrounding food; it always seeks higher concentrations. Actually, under certain experimental conditions, the cell will compare the concentration observed over the past 1 sec. with the concentration observed over the 3 sec. before that and it responds to the difference [61]. Hence, it uses the past 4 sec. of nutrient concentration data to decide how long to run [459]. Considering the deviations in direction due to Brownian movement discussed above, the bacterium basically uses as much time as it can in making decisions about climbing gradients [60]. In effect, the run length results from how much climbing it has done recently. If it has made lots of progress and hence, has just had a long run, then even if for a little while it is observing a homogeneous medium (without gradients), it will take a longer run. After a certain time period, it will recover and return to its standard behavior in a homogeneous medium.

Basically, the bacterium is trying to swim from places with low concentrations of nutrients to places with high concentrations. An opposite type of behavior is used when it encounters noxious substances. If the various concentrations move with time, then the bacteria will try to "chase" after the more favorable environments and run from harmful ones. Clearly, nutrient and noxious substance diffusion and motion will affect the motion patterns of a group of bacteria in complex ways.

18.2.3 Underlying Sensing and Decision-Making Mechanisms

Consider Figure 18.7, where a cross-section of one corner of the *E. coli* bacterium is shown. The sensors are the receptor proteins, which are signaled directly by external substances (e.g., in the case for the pictured amino acids) or via the "periplasmic substrate-binding proteins." The "sensor" is very sensitive, in some cases requiring less than 10 molecules of attractant to trigger a reaction, and attractants can trigger a swimming reaction in less than 200 ms. You can then think of the bacterium as having a "high gain" with a small attractant detection threshold (detection of only a small number of molecules can trigger a doubling or tripling of the run length). On the other hand, the corresponding threshold for encountering a homogeneous medium after being in a nutrient rich one is larger. Also, there is a type of time-averaging that is occurring in the sensing process. The receptor proteins then affect signaling molecules inside the bacterium. Also, there is in effect an "adding machine" and an ability to compare values and to arrive at an overall decision about which mode the flagella should operate in; essentially, the different sensors add and subtract their effects, and the more active or numerous have a greater influence on the final decision. Even though the sensory and decision-making system in *E. coli* is probably the best understood one in biology, we are ignoring the underlying chemistry that is needed for a full explanation (the interested reader can see the "For Further Study" chapter at the end of this part to find references that explain it in detail).

The decision-making is implemented by chemical reactions driven by sensing of chemicals in the environment.

It is interesting to note that the "decision-making system" in the *E. coli* bacterium must have some ability to sense a *derivative*, and hence, it has a type of memory! At first glance it may seem possible that the bacterium senses concentrations at both ends of the cell and finds a simple difference to recognize a concentration gradient (a spatial derivative); however, this is not the case. Experiments have shown that it performs a type of sampling, and roughly speaking, it remembers the concentration a moment ago, compares it with a current one, and makes decisions based on the difference (i.e., it computes something like an Euler approximation to a time derivative). Actually, in [553] the authors show how internal bacterial decision-making processes involve some type of integral feedback control mechanism.

In summary, we see that with memory, a type of addition mechanism, an ability to make comparisons, a few simple internal "control rules," and its chemical sensing and locomotion capabilities, the bacterium is able to achieve a complex type of searching and avoidance behavior. Evolution has designed this control system. It is robust and clearly very successful at meeting its goals of survival when viewed from a population perspective.

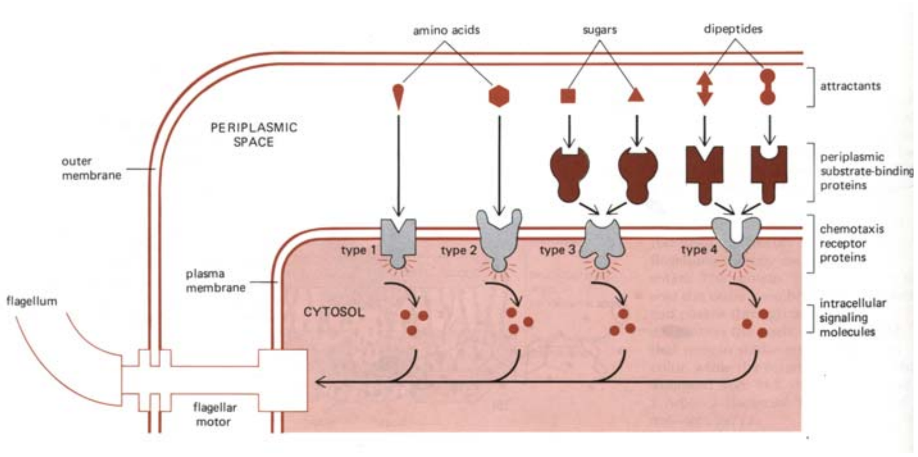

Figure 18.7: Sensing and internal mechanisms for control in the *E. coli* bacterium (figure taken from [8], © Garland Science/Taylor and Francis Books, Inc., used with permission).

18.2.4 Elimination and Dispersal Events

It is possible that the local environment where a population of bacteria lives changes either gradually (e.g., via consumption of nutrients) or suddenly due to some other influence. There can be events such that all the bacteria in a region are killed or a group is dispersed into a new part of the environment. For example, local significant increases in heat can kill a population of bacteria that are currently in a region with a high concentration of nutrients (you can think of heat as a type of noxious influence). Or, it may be that water or some animal will move populations of bacteria from one place to another in the environment. Over long periods of time, such events have spread various types of bacteria into virtually every part of our environment, from our intestines, to hot springs and underground environments, and so on.

What is the effect of elimination and dispersal events on chemotaxis? It has the effect of possibly destroying chemotactic progress, but it also has the effect of assisting in chemotaxis since dispersal may place bacteria near good food sources. From a broad perspective, elimination and dispersal is part of the population-level motile behavior.

Elimination and dispersal of bacteria should be thought of as a component of their overall motility.

18.2.5 Evolution of Bacteria

Mutations in *E. coli* occur at a rate of about 10^{-7} per gene, per generation. In addition to mutations that affect its physiological aspects (e.g., reproductive efficiency at different temperatures), *E. coli* bacteria occasionally engage in a type of "sex" called "conjugation," where small gene sequences are unidirectionally transferred from one bacterium to another. It seems that these gene sequences apparently carry good fitness characteristics in terms of reproductive capability, so conjugation is sometimes thought of as a transmittal of "fertility." To achieve conjugation, a pilus extends to make contact with another bacterium, and the gene sequence transfers through the pilus.

It is important to note that there are some very basic differences in evolution for higher organisms and bacteria. While conjugation apparently spreads "good" gene sequences, the "homogenizing effect" on gene frequency from conjugation is relatively small compared to how sex works in other organisms. This is partly since conjugation is relatively rare, and partly since the rate of reproduction is relatively high, on the order of hours depending on environmental conditions. Due to these characteristics, population genetics for *E. coli* may be dominated by selection sweeps triggered by the acquisition, via sex, of an adaptive allele.

18.2.6 Taxes in Other Swimming Bacteria

There are a wide range of foraging behaviors in bacteria, all of which can be modeled as optimization processes.

While most bacteria are motile and many types have analogous taxes capabilities to *E. coli* bacteria, the specific sensing, actuation, and decision-making mechanisms are different [384, 24]. For instance, while the proton-driven motor on *E. coli* rotates at a few hundred revolutions per second, Na^+-driven motors on some bacteria rotate at speeds up to 1000 revolutions per second, and on some species, the motor can turn in either direction or stop. Different types of bacteria can sense different phenomena and have different underlying decision-making, so they may search for and try to avoid different phenomena. Some bacteria can sense their own metabolic state and only respond to compounds currently required for growth and their pattern of responses may change based on their environment. Studies of the mechanisms for decision and control in various bacteria do, however, indicate that they have common features and hence, some have suggested that there was a single early evolutionary event that resulted in the swimming capability of bacteria. Swimming generally moves a bacterium to a more favorable environment for growth, or it maintains it in its current position, and hence, it gives the bacteria a survival advantage. Some scientists have suggested that the shapes of motile bacteria developed to allow efficient swimming. Some bacteria even change their shape to reduce the adverse effects of moving through more viscous media. Even though there can be significant differences between species, all swimming bacteria seem to have similar swimming patterns, where there is an alternation between smooth swimming and a change in direction (i.e., a type of saltatory search, a concept that is explained in Section 18.1.2). Next, several examples of other types of sensing and taxes in swimming bacteria are provided.

Some bacteria can search for oxygen, and hence their motility behavior is based on "aerotaxis," while others search for desirable temperatures resulting in "thermotaxis." Actually, the *E. coli* is capable of thermotaxis in that it seeks warmer environments with a temperature range of 20 deg. to 37 deg. C. Other bacteria, such as *Thiospirillum jenense*, search for or avoid light of certain wavelengths and this is called "phototaxis" Actually, the *E. coli* tries to avoid intense blue light, so it is also capable of phototaxis. Some bacteria swim along magnetic lines of force that enter the earth, so that when in the northern hemisphere, they swim towards the north magnetic pole, and in the southern hemisphere, they swim towards the south magnetic pole. (This is due to the presence of a small amount of magnetic material *in the cell* that essentially acts as a compass to passively reorient the cell.)

There are square-shaped bacteria that are propelled either forward or backward via flagella, and when multiple such bacteria naturally collide, their flagella can become "clumped," and this seems to be responsible for their tumbling. Hence, their motility behavior is characterized by forward movement, followed by either forward or backward movement, and an intermittent change in direction via tumbling [7]. *Vibrio alginolyticus* move differently when free-living versus living on a surface. Free-living *Vibrio alginolyticus* swims using a Na^+-driven motor on its flagella but when it is on the surface of a liquid, it senses the increased viscosity via the flagellar motor and then synthesizes many proton-driven flagella, which then allow the cell to move over surfaces [24]. The cells move as groups ("rafts"), since this is thought to help overcome viscous drag and surface tension. In other bacteria, flagella can be synthesized and discarded as they are needed.

18.2.7 Other Group Phenomena in Bacteria

A particularly interesting group behavior has been demonstrated for several motile species of bacteria, including *E. coli* and *S. typhimurium*, where intricate stable spatio-temporal patterns (swarms)[1] are formed in semi-solid nutrient media [84, 71, 83, 544, 24] (see Figure 18.8). When a group of *E. coli* cells is placed in the center of a semi-solid agar with a single nutrient chemo-effector (sensor), they move out from the center in a traveling ring of cells by moving up the nutrient gradient created by consumption of the nutrient by the group. Moreover, if high levels of the nutrient called succinate are used as the nutrient, then the cells release the attractant aspartate, so that they congregate into groups and hence, move as concentric patterns of groups with high bacterial density; see the concentric pattern of dots in Figure 18.8. (Note that many cells in those groups permanently lose motility.) The spatial order results from outward movement of the ring and the local releases of the attractant; the cells provide an attraction signal to each other so they swarm together. Pattern formation can be suppressed by a background of aspartate (since it seems that this will in essence scramble the chemical signal by eliminating its directionality). The pattern seems to form based on the dominance of two stimuli (cell-cell signaling and foraging).

Swarms arise due to communications, and can lead to more successful foraging (optimization).

The role of these patterns in natural environments is not understood; however, there is evidence that stress to the bacteria results in them releasing chemical signals that other bacteria are chemotactic towards. If enough stress is

[1]Actually, microbiologists reserve the term "swarming" for other characteristics of groups of bacteria. Here, we abuse the terminology and favor using the terminology that is used for higher forms of animals such as bees.

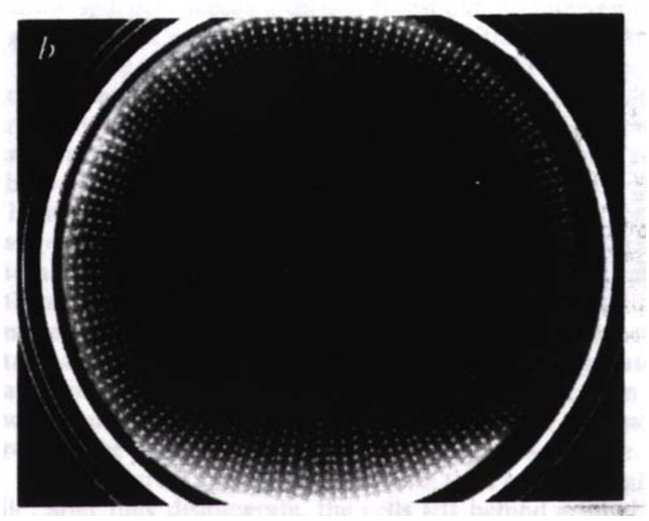

Figure 18.8: Swarm pattern of *E. coli* (figure taken from [84], © Macmillan Magazines, used with permission from Nature).

present, then a whole group can secrete the chemical signal strengthening the total signal, and hence, an aggregate of the bacteria forms. It seems that this aggregate forms to protect the group from the stress (e.g., by effectively hiding many cells in the middle of the group). It seems that the aggregates of the bacteria are not necessarily stationary; under certain conditions they can migrate, split, and fuse. This has led researchers to hypothesize that there may be other communication methods being employed that are not yet understood.

As another example, there are "biofilms" that can be composed of multiple types of bacteria (e.g., *E. coli*) that can coat various objects (e.g., roots of plants or medical implants). It seems that both motility and "quorum sensing" are involved in biofilm formation. A biofilm is a mechanism for keeping a bacterial species in a fixed location, avoiding overcrowding, and avoiding nutrient limitation and toxin production by packing them at a low density in a "polysaccharide matrix." Secreted chemicals provide a mechanism for the cells to sense population density, but motility seems to assist in the early stages of biofilm formation. It is also thought that chemotactic responses are used to drive cells to the outer edges of the biofilm, where nutrient concentrations may be higher.

In a variety of bacteria, including *E. coli*, complex patterns result primarily not from motility, but from reproduction [464]. In some bacteria, it seems that there is a type of signaling that occurs and results in the formation of regular patterns as the culture of bacteria grows. Formation of such patterns is sometimes thought of as a type of multicellular "morphogenesis." For example, the formation of the "fruiting bodies" by *Myxococcus xanthus* can be viewed as a type of morphogenesis, but one that seems to be primarily based on motility and cell deaths rather than reproduction [467].

Other types of bacteria exhibit group behaviors [334]. For instance, there are luminous bacteria that will emit no light until the population reaches a certain density. For instance, the bacteria *Vibrio fischeri* lives in the ocean at low concentrations and its secreted "autoinducer" chemical signal is quite dilute. However, the squid *Euprymna scolopes* selects these bacteria to grow in its light organ. When a sufficiently large population is cultivated in its light organ, the autoinducer chemical signals given off by each bacterium effectively add to result in a high concentration of this chemical and, when it reaches a certain threshold, each cell will switch on its luminescence property so that as a group they emit a visible light [334]. The squid, which is a nocturnal forager, benefits since the light camouflages it from predators below, since its light resembles moonlight and hence, effectively eliminates its shadow. The bacteria benefit by getting nourishment and shelter. The bacteria and squid are in a symbiont relationship (i.e., they live together to benefit each other).

Also, the soil-dwelling *streptomycete* colonies can grow a branching network of long fiber-like cells that can penetrate and degrade vegetation and then feed on the resulting decaying matter. (In terms of combinatorial optimization, you may think of finding optimal trees or graphs.) Under starvation conditions, they can cooperate to produce spores on a structure called an "aerial mycelium" that may be carried away.

As another example, in *Proteus mirabilis* the rod-shaped cells exist as "swimmers" that are driven by fewer than 10 flagella when they are in liquid media and they have chemotactic responses analogous to those of *E. coli*. If, however, these swimmers are placed on a solid surface, the swimmer cell "differentiates" (changes) into a "swarmer cell" that is an elongated rod (of roughly the same diameter) with more than $10,000$ flagella. On solid surfaces, the cells aggregate and exhibit swarm behavior in foraging via group chemotaxis. If they are then placed back in a liquid medium, there is a process of "consolidation" where swarmer cells split into swimmer cells. Moreover, when swarming they exhibit the "Dienes phenomenon," where swarms of the same type of bacteria try to avoid each other. (The mechanisms of this apparent territorial behavior are not well-understood.)

18.3 *E. coli* Bacterial Swarm Foraging for Optimization

Suppose that we want to find the minimum of $J(\theta)$, $\theta \in \Re^p$, where we do not have measurements, or an analytical description, of the gradient $\nabla J(\theta)$. Here, we use ideas from bacterial foraging to solve this "nongradient" optimization problem. First, suppose that θ is the position of a bacterium and $J(\theta)$ represents the combined effects of attractants and repellents from the environment, with, for example, $J(\theta) < 0$, $J(\theta) = 0$, and $J(\theta) > 0$ representing that the bacterium at location θ is in nutrient-rich, neutral, and noxious environments, respectively. Basically, chemotaxis is a foraging behavior that implements a

The bacterial foraging algorithm is a nongradient stochastic optimization method.

type of optimization where bacteria try to climb up the nutrient concentration (find lower and lower values of $J(\theta)$) and avoid noxious substances and search for ways out of neutral media (avoid being at positions θ where $J(\theta) \geq 0$).

18.3.1 An Optimization Model for *E. coli* Bacterial Foraging

To define our optimization model of *E. coli* bacterial foraging, we need to define a population (set) of bacteria, and then model how they execute chemotaxis, swarming, reproduction, and elimination/dispersal. After doing this, we will highlight the limitations (inaccuracies) in our model.

Population and Chemotaxis

Define a chemotactic step to be a tumble followed by a tumble or a tumble followed by a run. Let j be the index for the chemotactic step. Let k be the index for the reproduction step. Let ℓ be the index of the elimination-dispersal event. Let

$$P(j,k,\ell) = \left\{\theta^i(j,k,\ell) | i = 1, 2, \ldots, S\right\}$$

represent the positions of each member in the population of the S bacteria at the j^{th} chemotactic step, k^{th} reproduction step, and ℓ^{th} elimination-dispersal event. Here, let $J(i,j,k,\ell)$ denote the cost at the location of the i^{th} bacterium $\theta^i(j,k,\ell) \in \Re^p$ (sometimes we drop the indices and refer to the i^{th} bacterium position as θ^i). Note that we will interchangeably refer to J as being a "cost" (using terminology from optimization theory) and as being a nutrient surface (in reference to the biological connections). For actual bacterial populations, S can be very large (e.g., $S = 10^9$), but $p = 3$. In our computer simulations, we will use much smaller population sizes and will keep the population size fixed. We will allow $p > 3$, so we can apply the method to higher dimensional optimization problems.

Let N_c be the length of the lifetime of the bacteria as measured by the number of chemotactic steps they take during their life. Let $C(i) > 0$, $i = 1, 2, \ldots, S$, denote a basic chemotactic step size that we will use to define the lengths of steps during runs. To represent a tumble, a unit length random direction, say $\phi(j)$, is generated; this will be used to define the direction of movement after a tumble. In particular, we let

$$\theta^i(j+1,k,\ell) = \theta^i(j,k,\ell) + C(i)\phi(j)$$

so that $C(i)$ is the size of the step taken in the random direction specified by the tumble. If at $\theta^i(j+1,k,\ell)$ the cost $J(i,j+1,k,\ell)$ is better (lower) than at $\theta^i(j,k,\ell)$, then another step of size $C(i)$ in this same direction will be taken, and again, if that step resulted in a position with a better cost value than at the previous step, another step is taken. This swim is continued as long as it continues to reduce the cost, but only up to a maximum number of steps, N_s. This represents that the cell will tend to keep moving if it is headed in the direction of increasingly favorable environments.

Swarming Mechanisms

The above discussion was for the case where no cell-released attractants are used to signal other cells that they should swarm together. Here, we will also have cell-to-cell signaling via an attractant and will represent that with $J^i_{cc}(\theta, \theta^i(j,k,\ell))$, $i = 1, 2, \ldots, S$, for the i^{th} bacterium. Let

Swarm optimization exploits a regional approximation to a gradient and helps it to climb over noise.

$$d_{attract} = 0.1$$

be the depth of the attractant released by the cell (a quantification of how much attractant is released) and

$$w_{attract} = 0.2$$

be a measure of the width of the attractant signal (a quantification of the diffusion rate of the chemical). The cell also repels a nearby cell in the sense that it consumes nearby nutrients and it is not physically possible to have two cells at the same location. To model this, we let

$$h_{repellent} = d_{attract}$$

be the height of the repellent effect (magnitude of its effect) and

$$w_{repellent} = 10$$

be a measure of the width of the repellent. The values for these parameters are simply chosen to illustrate general bacterial behaviors, not to represent a particular bacterial chemical signaling scheme. The particular values of the parameters were chosen with the nutrient profile in mind, which we will use later in Figure 18.10. For instance, the depth and width of the attractant is small relative to the nutrient concentrations represented in Figure 18.10. Let

$$\begin{aligned} J_{cc}(\theta, P(j,k,\ell)) &= \sum_{i=1}^{S} J^i_{cc}(\theta, \theta^i(j,k,\ell)) \\ &= \sum_{i=1}^{S} \left[-d_{attract} \exp\left(-w_{attract} \sum_{m=1}^{p} (\theta_m - \theta^i_m)^2 \right) \right] \\ &+ \sum_{i=1}^{S} \left[h_{repellent} \exp\left(-w_{repellent} \sum_{m=1}^{p} (\theta_m - \theta^i_m)^2 \right) \right] \end{aligned}$$

denote the combined cell-to-cell attraction and repelling effects, where $\theta = [\theta_1, \ldots, \theta_p]^\top$ is a point on the optimization domain and θ^i_m is the m^{th} component of the i^{th} bacterium position θ^i (for convenience, we omit some of the indices). An example for the case of $S = 2$ and the above parameter values is shown in Figure 18.9. Here, note that the two sharp peaks represent the cell locations, and as you move radially away from the cell, the function decreases and then increases (to model the fact that cells far away will tend not to be attracted, whereas cells close by will tend to try to climb down the cell-to-cell nutrient gradient towards each other and hence try to swarm). Note that as each cell moves, so does its $J^i_{cc}(\theta, \theta^i(j,k,\ell))$ function, and this represents that it will release chemicals as it moves. Due to the movements of all the cells, the $J_{cc}(\theta, P(j,k,\ell))$ function is *time-varying* in that, if many cells come close together, there will be a high amount of attractant and hence, an increasing likelihood that other cells will move towards the group. This produces the swarming effect. When we want to study swarming, the i^{th} bacterium, $i = 1, 2, \ldots, S$, will hill-climb on

$$J(i,j,k,\ell) + J_{cc}(\theta, P)$$

(rather than the $J(i,j,k,\ell)$ defined above) so that the cells will try to find nutrients, avoid noxious substances, and at the same time try to move towards other cells, but not too close to them. The $J_{cc}(\theta, P)$ function dynamically deforms the search landscape as the cells move to represent the desire to swarm (i.e., we model mechanisms of swarming as a minimization process).

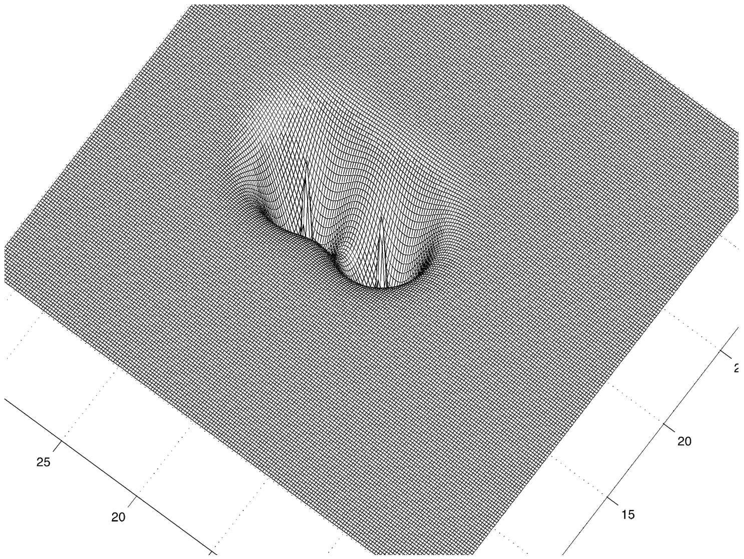

Figure 18.9: Cell-to-cell chemical attractant model, $S = 2$.

Reproduction and Elimination/Dispersal

After N_c chemotactic steps, a reproduction step is taken. Let N_{re} be the number of reproduction steps to be taken. For convenience, we assume that S is a positive even integer. Let

$$S_r = \frac{S}{2} \tag{18.1}$$

be the number of population members who have had sufficient nutrients so that they will reproduce (split in two) with no mutations. For reproduction, the population is sorted in order of ascending accumulated cost (higher accumulated cost represents that it did not get as many nutrients during its lifetime of foraging and hence, is not as "healthy" and thus unlikely to reproduce); then the S_r least healthy bacteria die and the other S_r healthiest bacteria each split into two bacteria, which are placed at the same location. Other fractions or approaches could be used in place of Equation (18.1); this method rewards bacteria that have encountered a lot of nutrients, and allows us to keep a constant population size, which is convenient in coding the algorithm.

Run length, reproduction, elimination, and dispersal all help to avoid local minima.

Let N_{ed} be the number of elimination-dispersal events, and for each such elimination-dispersal event, each bacterium in the population is subjected to elimination-dispersal with probability p_{ed}. We assume that the frequency of chemotactic steps is greater than the frequency of reproduction steps, which is in turn greater in frequency than elimination-dispersal events (e.g., a bacterium will take many chemotactic steps before reproduction, and several generations may take place before an elimination-dispersal event).

Foraging Model Limitations

Clearly, we are ignoring many characteristics of the actual biological optimization process in favor of simplicity and capturing the gross characteristics of chemotactic hill-climbing and swarming. For instance, we assume that consumption does not affect the nutrient surface (e.g., while a bacterium is in a nutrient-rich environment, we do not increase the value of J near where it has consumed nutrients) where clearly in nature, bacteria modify the nutrient concentrations via consumption. A tumble does not result in a perfectly random new direction for movement; however, here we assume that it does. Brownian effects buffet the cell, so that after moving a small distance, it is within a pie-shaped region of its start point at the tip of the piece of pie. Basically, we assume that swims are straight, whereas in nature they are not. Tumble and run lengths are exponentially distributed random variables, not constant, as we assume. Run-length decisions are actually based on the past 4 sec. of concentrations, whereas here we assume that at each tumble, older information about nutrient concentrations is lost. Although naturally asynchronous, we force synchronicity by requiring, for instance, chemotactic steps of different bacteria to occur at the same time, all bacteria to reproduce at the same time instant, and all bacteria that are subjected to elimination and dispersal to do so at the same time. We assume a constant population size, even if there are many nutrients and generations. We 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. (A more biologically accurate model of the swarming behavior of certain bacteria is given in [544].) Clearly, other choices for the criterion of which bacteria should split could be used (e.g., based only on the concentration at the end of a cell's lifetime, or on the quantity of noxious substances that were encountered). We are also ignoring conjugation and other evolutionary characteristics. For instance, we assume that $C(i)$, N_s, and N_c remain the same for each generation. In nature it seems likely that these parameters could evolve for different environments to maximize population growth rates.

18.3.2 Bacterial Foraging Optimization Algorithm

For initialization, you must choose p, S, N_c, N_s, N_{re}, N_{ed}, p_{ed}, and the $C(i)$, $i = 1, 2, \ldots, S$. If you use swarming, you will also have to pick the parameters of the cell-to-cell attractant functions; here we will use the parameters given above. Also, initial values for the θ^i, $i = 1, 2, \ldots, S$, must be chosen. Choosing these to be in areas where an optimum value is likely to exist is a good choice. Alternatively, you may want to simply randomly distribute them across the domain of the optimization problem. The algorithm that models bacterial population chemotaxis, swarming, reproduction, elimination, and dispersal is given below (initially, $j = k = \ell = 0$). For the algorithm, note that updates to the θ^i automatically result in updates to P. Clearly, we could have added a more sophisticated termination test than simply specifying a maximum number of iterations.

1. Elimination-dispersal loop: $\ell = \ell + 1$
2. Reproduction loop: $k = k + 1$
3. Chemotaxis loop: $j = j + 1$
   (a) For $i = 1, 2, \ldots, S$, take a chemotactic step for bacterium i as follows.
   (b) Compute $J(i, j, k, \ell)$. Let
   $$J(i, j, k, \ell) = J(i, j, k, \ell) + J_{cc}(\theta^i(j, k, \ell), P(j, k, \ell))$$
   (i.e., add on the cell-to-cell attractant effect to the nutrient concentration).
   (c) Let $J_{last} = J(i, j, k, \ell)$ to save this value, since we may find a better cost via a run.
   (d) Tumble: generate a random vector $\Delta(i) \in \Re^p$ with each element $\Delta_m(i)$, $m = 1, 2, \ldots, p$, a random number on $[-1, 1]$.
   (e) Move: let
   $$\theta^i(j+1, k, \ell) = \theta^i(j, k, \ell) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^\top(i)\Delta(i)}}$$
   This results in a step of size $C(i)$ in the direction of the tumble for bacterium i.
   (f) Compute $J(i, j+1, k, \ell)$, and then let $J(i, j+1, k, \ell) = J(i, j+1, k, \ell) + J_{cc}(\theta^i(j+1, k, \ell), P(j+1, k, \ell))$.

(g) Swim (note that we use an approximation, since we decide swimming behavior of each cell as if the bacteria numbered $\{1, 2, \ldots, i\}$ have moved, and $\{i+1, i+2, \ldots, S\}$ have not; this is much simpler to simulate than simultaneous decisions about swimming and tumbling by all bacteria at the same time):

   i. Let $m = 0$ (counter for swim length).

   ii. While $m < N_s$ (if have not climbed down too long)

   - Let $m = m + 1$.
   - If $J(i, j+1, k, \ell) < J_{last}$ (if doing better), let $J_{last} = J(i, j+1, k, \ell)$ and let

$$\theta^i(j+1, k, \ell) = \theta^i(j+1, k, \ell) + C(i)\frac{\Delta(i)}{\sqrt{\Delta^\top(i)\Delta(i)}}$$

     and use this $\theta^i(j+1, k, \ell)$ to compute the *new* $J(i, j+1, k, \ell)$ as we did in (f) above.
   - Else, let $m = N_s$. This is the end of the while statement.

(h) Go to next bacterium $(i+1)$ if $i \neq S$ (i.e., go to (b) above to process the next bacterium).

4. If $j < N_c$, go to step 3. In this case, continue chemotaxis, since the life of the bacteria is not over.

5. Reproduction:

   (a) For the given k and ℓ, and for each $i = 1, 2, \ldots, S$, let

$$J_{health}^i = \sum_{j=1}^{N_c+1} J(i, j, k, \ell)$$

   be the health of bacterium i (a measure of how many nutrients it got over its lifetime and how successful it was at avoiding noxious substances). Sort bacteria and chemotactic parameters $C(i)$ in order of ascending cost J_{health} (higher cost means lower health).

   (b) The S_r bacteria with the highest J_{health} values die and the other S_r bacteria with the best values split (and the copies that are made are placed at the same location as their mother).

6. If $k < N_{re}$, go to step 2. In this case, we have not reached the number of specified reproduction steps, so we start the next generation in the chemotactic loop.

7. Elimination-dispersal: for $i = 1, 2, \ldots, S$, with probability p_{ed}, eliminate and disperse each bacterium (this keeps the number of bacteria in the population constant). To do this, if you eliminate a bacterium, simply disperse one to a random location on the optimization domain.

8. If $\ell < N_{ed}$, then go to step 1; otherwise end.

18.3.3 Guidelines for Algorithm Parameter Choices

The bacterial foraging optimization algorithm requires specification of a variety of parameters. First, you can pick the size of the population, S. Clearly, increasing the size of S can significantly increase the computational complexity of the algorithm. However, for larger values of S, if you choose to randomly distribute the initial population, it is more likely that you will start at least some bacterium near an optimum point, and over time, it is then more likely that many bacterium will be in that region, due to either chemotaxis or reproduction.

What should the values of the $C(i)$, $i = 1, 2, \ldots, S$, be? You can choose a biologically motivated value; however, such values may not be the best for an engineering application. If the $C(i)$ values are too large, then if the optimum value lies in a valley with steep edges, it will tend to jump out of the valley, or it may simply miss possible local minima by swimming through them without stopping. On the other hand, if the $C(i)$ values are too small, then convergence can be slow, but if it finds a local minimum, it will typically not deviate too far from it. You should think of the $C(i)$ as a type of "step size" for the optimization algorithm.

The size of the values of the parameters that define the cell-to-cell attractant functions J_{cc}^i will define the characteristics of swarming. If the attractant width is high and very deep, the cells will have a strong tendency to swarm (they may even avoid going after nutrients and favor swarming). On the other hand, if the attractant width is small, and the depth shallow, there will be little tendency to swarm and each cell will search on its own. Social versus independent foraging is then dictated by the balance between the strengths of the cell-to-cell attractant signals and nutrient concentrations.

Next, large values for N_c result in many chemotactic steps, and, hopefully, more optimization progress, but of course, more computational complexity. If the size of N_c is chosen to be too short, the algorithm will generally rely more on luck and reproduction, and in some cases, it could more easily get trapped in a local minimum ("premature convergence"). You should think of N_s as creating a bias in the random walk (which would not occur if $N_s = 0$), with large values tending to bias the walk more in the direction of climbing down the hill.

If N_c is large enough, the value of N_{re} affects how the algorithm ignores bad regions and focuses on good ones, since bacteria in relatively nutrient-poor regions die (this models, with a fixed population size, the characteristic where bacteria will tend to reproduce at higher rates in favorable environments). If N_{re} is too small, the algorithm may converge prematurely; however, larger values of N_{re} clearly increase computational complexity.

A low value for N_{ed} dictates that the algorithm will not rely on random elimination-dispersal events to try to find favorable regions. A high value increases computational complexity but allows the bacteria to look in more regions to find good nutrient concentrations. Clearly, if p_{ed} is large, the algorithm can degrade to random exhaustive search. If, however, it is chosen appropriately, it can help the algorithm jump out of local optima and into a global optimum.

18.3.4 Relations to Other Nongradient Optimization Methods

There are *algorithmic analogies* between the genetic algorithm and the above optimization model for foraging. There are analogies between the fitness function and the nutrient concentration function (both a type of "landscape"), selection and bacterial reproduction (bacteria in the most favorable environments gain a selective advantage for reproduction), crossover and bacterial splitting (the children are at the same concentration, whereas with crossover they generally end up in a region around their parents on the fitness landscape), and mutation and elimination and dispersal. However, the algorithms are not equivalent, and neither is a special case of the other. Each has its own distinguishing features. The fitness function and nutrient concentration functions are *not* the same (one represents likelihood of survival for given phenotypic characteristics, whereas the other represents nutrient/noxious substance concentrations, or for other foragers predator/prey characteristics). Crossover represents mating and resulting differences in offspring, something we ignore in the bacterial foraging algorithm (we could, however, have made less than perfect copies of the bacteria to represent their splitting). Moreover, mutation represents gene mutation and the resulting phenotypical changes, not physical dispersal in an environment.

From one perspective, note that all the typical features of genetic algorithms could augment the bacterial foraging algorithm by representing evolutionary characteristics of a forager in their environment. From another perspective, foraging algorithms can be integrated into evolutionary algorithms and thereby model some key survival activities that occur during the lifetime of the population that is evolving (i.e., foraging success can help define fitness, mating characteristics, etc.). For the bacteria studied here, foraging happens to entail hill-climbing via a type of biased random walk, and hence, the foraging algorithm can be viewed as a method to integrate a type of approximate stochastic gradient search (where only an approximation to the gradient is used, not analytical gradient information) into evolutionary algorithms. Of course, standard gradient methods, quasi-Newton methods, etc., depend on the use of an explicit analytical representation of the gradient, something that is not needed by a foraging or genetic algorithm. Lack of dependence on analytical gradient information can be viewed as an advantage (fewer assumptions), or a disadvantage (e.g., since, if gradient information is available, then the foraging or genetic algorithm may not exploit it properly).

You probably also recognize some similarities between certain features of the foraging algorithm and SPSA. What are they? What are the relationships to the nongradient methods of the last part? There are in fact many approaches to "global optimization" when there is no explicit gradient information available; however, it is beyond the scope of this book to evaluate the relative merits of foraging algorithms to the vast array of such methods that have been studied for many years. To start such a study, it makes sense to begin by considering the theoretical convergence guarantees for certain types of evolutionary algorithms, stochastic approximation methods, and pattern search methods (e.g., see [481] for work along these lines), and then proceed to consider foraging algorithms in this context. It also seems useful to consider how well the foraging algorithms will perform for time-varying nutrient landscapes, which occurs in the underlying biological problem and many engineering problems.

18.3.5 Example: Function Optimization via *E. coli* Foraging

As a simple illustrative example, we use the algorithm to try to find the minimum of the function in Figure 18.10 (note that the point $[15,5]^\top$ is the global minimum point).

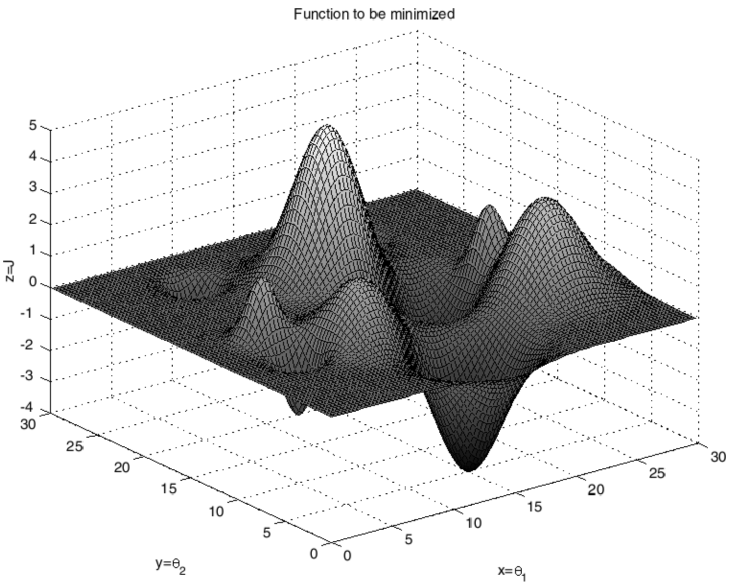

Figure 18.10: Nutrient landscape.

Nutrient Hill-Climbing: No Swarming

According to the above guidelines, choose $S = 50$, $N_c = 100$, $N_s = 4$ (a biologically motivated choice), $N_{re} = 4$, $N_{ed} = 2$, $p_{ed} = 0.25$, and the $C(i) = 0.1$, $i = 1, 2, \ldots, S$. The bacteria are initially spread randomly over the optimization domain. The results of the simulation are illustrated by motion trajectories of the bacteria on the contour plot of Figure 18.10, as shown in Figure 18.11. In the first generation, starting from their random initial positions, searching is occurring in many parts of the optimization domain, and you can see the chemotactic motions of the bacteria as the black trajectories where the peaks are avoided and the valleys are pursued. Reproduction picks the 25 healthiest bacteria and copies them, and then, as shown in Figure 18.11 in generation 2, all the chemotactic steps are in five local minima. This again happens in going to generations 3 and 4, but bacteria die in some of the local minima (due essentially to our requirement that the population size stay constant), so that in generation 3, there are four groups of bacteria in four local minima, whereas in generation 4, there are two groups in two local minima.

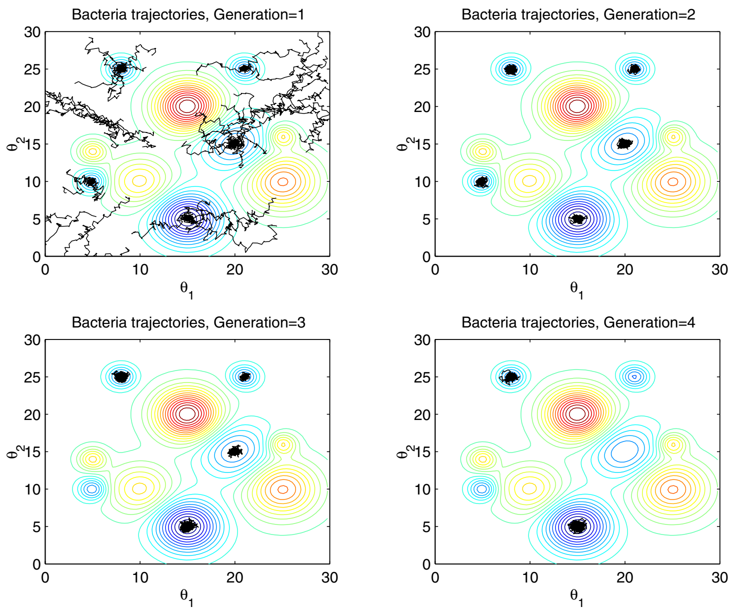

Figure 18.11: Bacterial motion trajectories, generations 1–4, on contour plots.

Next, with the above choice of parameters, there is an elimination-dispersal event, and we get the *next* four generations shown in Figure 18.12. Notice that elimination and dispersal shifts the locations of several of the bacteria and thereby the algorithm explores other regions of the optimization domain. However, qualitatively we find a similar pattern to the previous four generations where chemotaxis and reproduction work together to find the global minimum; this time, however, due to the large number of bacteria that were placed near the global minimum, after one reproduction step, all the bacteria are close to it (and remain this way). In this way, the bacterial population has found the global minimum.

Swarming Effects

Here we use the parameters defined earlier to define the cell-to-cell attraction function. Also, we choose $S = 50$, $N_c = 100$, $N_s = 4$, $N_{re} = 4$, $N_{ed} = 1$, $p_{ed} = 0.25$, and the $C(i) = 0.1$, $i = 1, 2, \ldots, S$. We will first consider swarming effects on the nutrient concentration function with contour map shown on Figure 18.13 which has a zero value at $[15, 15]^\top$ and decreases to successively more negative

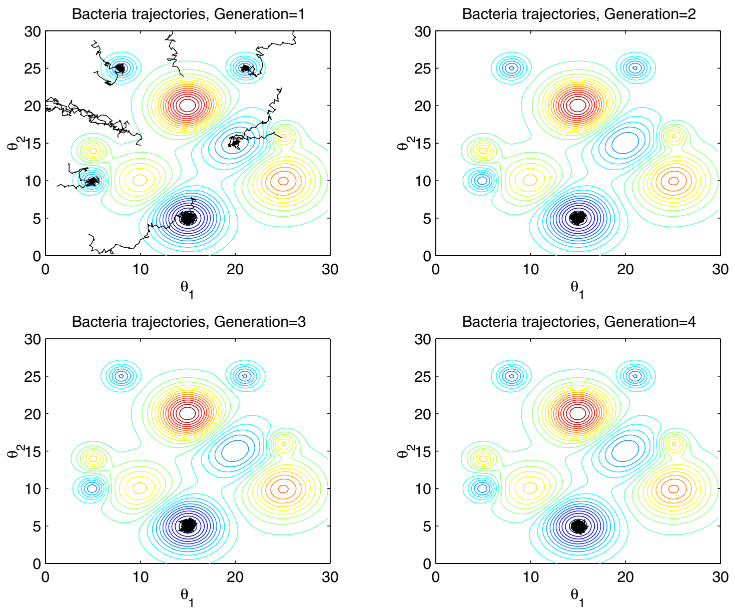

Figure 18.12: Bacterial motion trajectories, generations 1–4, on contour plots, after an elimination-dispersal event.

values as you move away from that point; hence, the cells should tend to swim away from the peak. We will initialize the bacterial positions by placing all the cells at the peak $[15, 15]^\top$. Using these conditions, we get the result in Figure 18.13. Notice that in the first generation, the cells swim radially outward, and then in the second and third generations, swarms are formed in a concentric pattern of groups. Notice that with our simple method of simulating health of the bacteria and reproduction, some of the swarms are destroyed by the fourth generation. We omit additional simulations that show the behavior of the swarm on the surface in Figure 18.10, since qualitatively the behavior is as one would expect from the above simulations. The interested reader can obtain the code mentioned above and further study the behavior of the algorithm.

18.4 Stable Social Foraging Swarms

In this section, we first overview some biology of swarms, with a focus on the honey bee in order to provide a concrete example. Then, we introduce a mathematical model for a generic swarm of agents. We conduct a mathematical analysis to prove stability (cohesiveness) of the swarm and perform simulations to provide insights into swarm dynamics.

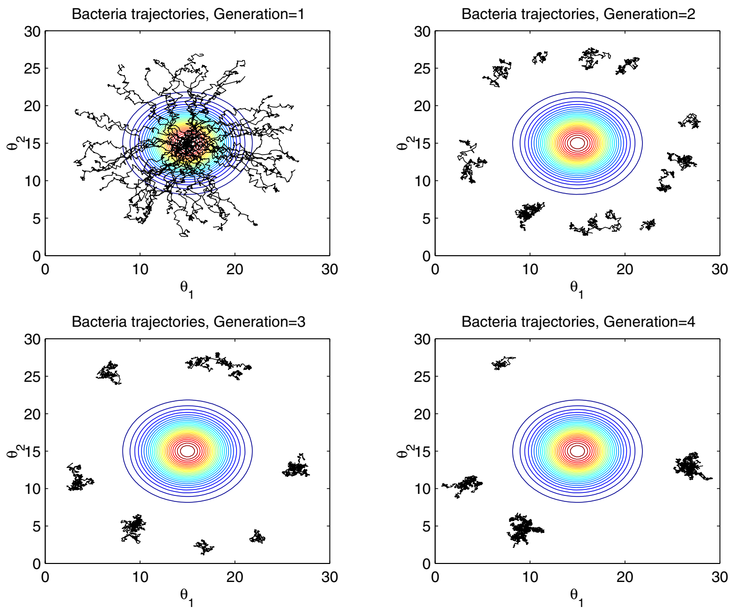

Figure 18.13: Swarm behavior of *E. coli* on a test function.

18.4.1 Example: The Biology of Honey Bee Swarms

There are many types of animals that swarm, including bacteria, insects, birds, fish, horses, and so on. (Groupings of different animals are typically given different names, but for convenience, here we will call them all swarms.) Such groups are composed of individuals with different physiological capabilities, and when operating as a group, they can achieve different types of emergent behaviors. Here, in order to be more concrete about how swarms operate, we will describe in more detail how and why one species, the honey bee, swarms.

First, when a hive splits, a group of bees will "cluster" around the queen on, for example, a nearby branch. This is a type of swarming where the group contracts together and forms a tightly packed group of relatively stationary bees. After forming such a cluster, the bees perform nest site selection. Upon liftoff, after the scouts generally reach agreement on a nest site, the honey bee swarm forms a spherical group that hovers near the place where the swarm had grouped while nest site selection occurred. Next, the group slowly starts to move in the direction of the new nest site, elongating laterally, and accelerating as a group to about 11 kilometers/hr. There is a type of "inertia" in getting the group moving due to the swarm dynamics. It seems that several factors contribute to in-transit swarm cohesiveness, two of which may involve pheromones. First, the queen releases a pheromone (but the queen is not in the lead so it does not seem likely that it can steer the bees that are ahead of her). Second, some evidence suggests that other bees in the swarm release Nasanov pheromones that help to maintain cohesiveness, but it seems that this is still not fully verified. How does the swarm know which *direction* to go? Again, there seem to be several factors. First, during the site selection process, it seems that those observing the final phase of the agreement process may know which direction to go. However, perhaps not all of the bees will know the direction, and this may explain the way that some bees in the swarm seem to wander somewhat within the swarm, apparently lacking knowledge of the direction to the nest. Alternatively, it seems possible that even if they knew the direction, they may not be able to easily navigate while in the swarm due to obstruction of navigation cues due to interference from many other bees. Another factor that certainly seems to influence the direction of movement of the swarm is the presence of the scouts that do know the way to the site, and which "pilot" the swarm by "streaking" through it in the general direction of the nest site. (You could think of this as an "aerial dance signal" to recruit bees in the proper direction.) Nevertheless, even with the possibility of several methods of group navigation, there are significant motions of bees within the swarm that are not directed towards the site. Aside from navigational obstructions, and possible bee-bee collisions, it also seems that it may be possible that there are conflicting indictions of the general direction to move, due to dynamic changes in the dominance of factors that guide the bee (e.g., pheromones, its own sense of direction, and the indications of scout piloting). Also, it should be clear that wind (perhaps even currents that are induced by the swarm) will disrupt a uniform flow of the swarm towards the nest site. At the same time, it should be clear that some type of "robust" group guidance and navigation is achieved since they successfully reach the site. Upon reaching the site the group stops (and it is not understood how that occurs), some scouts drop into the new nest entrance and release a pheromone, and this attracts the group to the new nest. A model of the clustering and in-transit motion of bees has been studied (see the "For Further Study" section at the end of the part for more details).

18.4.2 Swarm and Environment Models

Here, we describe the agent, communications, and the environment that the agents move in.

Agent Dynamics and Communications

Here, rather than focusing on the particular characteristics of one type of animal or autonomous vehicle, we consider a swarm to be composed of an interconnection of N "agents," each of which has point mass dynamics given by

$$\begin{aligned} \dot{x}^i &= v^i \\ \dot{v}^i &= \frac{1}{M_i} u^i \end{aligned} \qquad (18.2)$$

where $x^i \in \Re^n$ is the position, $v^i \in \Re^n$ is the velocity, M_i is the mass, and $u^i \in \Re^n$ is the (force) control input for the i^{th} agent. We use this simple linear model for an agent in order to illustrate the basic features of swarming and stability analysis of cohesion. Other nonlinear and stochastic models for agents in swarms have been considered in the literature (see the "For Further Study" section at the end of this part). Here, we will use $n = 3$ for swarms moving in a three-dimensional space. We will assume that each agent can sense information about the position and velocity of other agents, but only with some noise that we will define below.

The agents interact to form groups, and in some situations groups will split. Some think of having local interactions between agents, which "emerge" into a global behavior for the group. One way to represent which agents can interact with each other is via a directed graph (G, A) where $G = \{1, 2, \ldots, N\}$ is a set of nodes (the agents) and

$$A = \{(i, j) : i, j \in G, i \neq j\}$$

represents a "sensing/communication topology" (in general, then, each "link" (i, j) could be a dynamical system). For example, if $(i, j) \in A$, then it could be assumed that agent i can sense the position and velocity of agent j. In some vehicular systems it may be possible that A is fixed and independent of vehicle positions and velocities. Clearly, however, for biological systems it is often the case that A is not fixed, but changes dynamically based on the positions of the agents (e.g., so that only agents within a line-of-sight and close enough can be sensed). Here, for simplicity, we will assume that A does not change based on the positions and velocities of the agents, and that A is fully connected (e.g., that for all $i \in G$, $(i, j) \in A$). We also assume that there are no delays or noise in communicating, communications are not range constrained, and that there is infinite bandwidth. More general formulations may also include a "communications topology" that specifies which agents can send and receive messages to other agents. Such messages could represent a wide variety of communication capabilities of the agents (e.g., bee scout leadership in a swarm, sounds, and so on.).

Agent to Agent Attraction and Repulsion

Agent to agent interactions considered here are of the "attract-repel" type, where each agent seeks to be in a position that is "comfortable" relative to its neighbors (and for us, all other agents are its neighbors). Attraction indicates that each agent wants to be close to every other agent and provides the mechanism for achieving grouping and cohesion of the group of agents. Repulsion provides the mechanism where each agent does not want to be too close to every other agent (e.g., for animals to avoid collisions and excessive competition for resources). There are many ways to define attraction and repulsion, each of which can be represented by characteristics of how we define u^i for each agent, and we list a few of these below:

Agents want to be close to each other, but not too close.

- *Attraction:* There can be linear attraction and in this case we have terms in u^i of, for example, the form

$$-k_a \left(x^i - x^j \right)$$

where $k_a > 0$ is a scalar that represents the strength of attraction. If the agents are far apart, then there is a large attraction between them, and if they are close, there is a small attraction. In other cases, there can be nonlinear attraction terms that can be expressed in terms of nonlinear functions of $\left(x^i - x^j\right)$. Some attraction mechanisms are "local" (i.e., for range-constrained sensing, where the agent only tries to move to other agents that are close to it) and others which are "global" (i.e., where agents can be attracted to move near other agents no matter how far away they are). Attraction terms can be specified in terms of a variety of agent variables. For example, the above term is for positions, but we could have similar terms for velocity so that the agents will try to match the velocities of other agents. Moreover, the above term is "static" but we would have a local "dynamic" controller that tries to match agent variables.

- *Repulsion:* As with attraction, there are many types of repulsion terms, some local, global, static, or dynamic, each of which can be expressed in terms of a variety of agent variables. Sometimes repulsion is defined in terms that also quantify attraction, other times they quantify only a repulsion.

  - *Seek a "comfortable distance":* For example, a term in u^i may take the form

$$\left[-k \left(||x^i - x^j|| - d \right)\right] \left(x^i - x^j \right)$$

where $||x^i - x^j|| = \sqrt{\left(x^i - x^j\right)^\top \left(x^i - x^j\right)}$, $k > 0$ is the magnitude of the repulsion, and d can be thought of as a comfortable distance between the i^{th} and j^{th} agents. Here, the quantity in the brackets sets the size of the repulsion. When $||x^i - x^j||$ is small (relative to d), the term in the bracket is positive so that agents i and j try to move away from each other (there is repulsion). When $||x^i - x^j||$ is big (relative to d), then the term in the brackets is negative so that the agents are attracted to each other. Balance between attraction and repulsion (a basic concept in swarm dynamics that is sometimes referred to as an "equilibrium," even though it may not be one in the stability-theoretic sense) may be achieved when $||x^i - x^j|| = d$ so that the term above is zero.

  - *Repel when close:* Another type of repulsion term in u^i, which may be used with, for example, a linear attraction term, may take the form

$$k_r \exp\left(\frac{-\frac{1}{2}\|x^i - x^j\|^2}{r_s^2} \right) \left(x^i - x^j \right) \tag{18.3}$$

where $k_r > 0$ is the magnitude of the repulsion, and $r_s > 0$ quantifies the region size around the agent from which it will repel its neighbors. When $\|x^i - x^j\|$ is big relative to r_s, the whole term approaches zero.

– *Hard repulsion for collision avoidance:* The above repulsion terms are "soft," in the sense that when the two agents are at the same location, the repulsion force is finite. In some cases, it is appropriate to use a repulsion term that becomes increasingly large as two agents approach each other. One such term, that does not have an attraction component, is in the form of

$$\left[\max\left\{\left(\frac{a}{b||x^i - x^j|| - w} - \epsilon\right), 0\right\}\right]\left(x^i - x^j\right) \tag{18.4}$$

Here, $w > 0$ affects the radius of repulsion of the agents, $a > 0$ is a gain on the magnitude of the repulsion, $b > 0$ can change the shape of the repulsion gain, and $\epsilon > 0$ can be used to define the term so that it only has a local influence. For instance, the radius of the repulsion is

$$R' = \frac{1}{b}\left(\frac{a}{\epsilon} + w\right)$$

For agent positions such that $||x^i - x^j|| \geq R'$, the term in the brackets is zero. For given values of w, a, and b, you can choose ϵ to get any value of $R' > 0$. Note that a key feature of this repulsion term is that as $||x^i - x^j||$ goes from a large value to w/b, the value in the brackets goes to infinity to provide a "hard" repelling action and hence, avoid the possibility that two agents ever end up at the same position (i.e., to avoid collisions). Another way to define a hard repulsion is to consider agents to be solid "balls," where, if they collide, they do not deform.

For more ideas on how to define attraction and repulsion terms, see the "For Further Study" section at the end of this part.

Environment and Foraging

Next, we will define the environment that the agents move in. While there are many possibilities, here we will simply consider the case where they move over what we will call "resource profile" (e.g., nutrient profile) $J(x)$, where $x \in \Re^n$. We will however, think of this profile as being something where the agents want to be in certain regions of the profile and avoid other regions (e.g., where there are noxious substances). We will assume that $J(x)$ is continuous with finite slope at all points. Agents move in the direction of the negative gradient of $J(x)$

Agents follow resource profiles to meet their objectives.

$$-\nabla J(x) = -\frac{\partial J}{\partial x}$$

in order to move away from bad areas and into good areas of the environment (e.g., to avoid noxious substances and find nutrients). Hence, they will use a term in their u^i that holds the negative gradient of $J(x)$.

Clearly, there are many possible shapes for $J(x)$, including ones with many peaks and valleys. Here, we simply list two simple forms for $J(x)$ as follows:

- *Plane:* In this case, we have $J(x) = J_p(x)$ where

$$J_p(x) = R^\top x + r_o$$

  where $R \in \Re^n$ and r_o is a scalar. Here, $\nabla J_p(x) = R$.

- *Quadratic:* In this case, we have $J(x) = J_q(x)$ where

$$J_q(x) = \frac{r_m}{2}\|x - R_c\|^2 + r_o$$

  where r_m and r_o are scalars and $R_c \in \Re^n$. Here, $\nabla J_q(x) = r_m\,(x - R_c)$.

Below, we will assume that each agent can sense the gradient of the resource profile, but only with some noise.

It could be that the environment has many different types of agents in it, or the same types of agents with different objectives. In this case, there may be different resource profiles for each agent, or the agents may switch the profiles it follows, or strategies for following them. Different agents may have different capabilities to sense the profile (e.g., sensing only at a point, or sensing a range-constrained region) and move over it. If the agents are consuming food, this may change the shape of the profile, and of course, an agent may pollute its environment so it may affect the profile in that manner also. This would create a time-varying profile that is dependent on agent position, and possibly many other variables. See the "For Further Study" section at the end of this part for more discussion on nutrient profiles and, for example, their use in modeling bee swarming.

18.4.3 Stability Analysis of Swarm Cohesion Properties

Cohesion and swarm dynamics can be quantified and analyzed using stability analysis (e.g., via Lyapunov's method). You can pick agent dynamics, interactions, sensing capabilities, attraction/repulsion characteristics, and foraging environment characteristics, then quantify and analyze cohesion properties. Here, we will do this for a few simple cases in order to give a flavor of the type of analysis that is possible, and to provide insight into swarm properties and dynamics during social foraging.

Sensing, Noise, and Error Dynamics

First, let

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x^i$$

be the center of the swarm and

$$\bar{v} = \frac{1}{N}\sum_{i=1}^{N} v^i$$

be the average velocity (vector) which we view as the velocity of the group of agents. We assume that each agent can sense the distance from itself to $\bar{x}$, and the difference between its own velocity and $\bar{v}$. We assume that each agent knows its own velocity, but not its own position. Note that for some animals, its senses and sensory processing may naturally provide the distance to $\bar{x}$ and $\bar{v}$, but not all the individual positions and velocities of all agents that could be used to compute these. Also, we will consider below the case where there is noise in sensing these quantities.

The objective of each agent is to move so as to end up at or near $\bar{x}$ and have its velocity equal to $\bar{v}$; in this way, an emergent behavior of the group is produced where they aggregate dynamically and end up near each other and ultimately move in the same direction (i.e., they achieve cohesion). The problem is that since all the agents are moving at the same time, $\bar{x}$ and $\bar{v}$ are generally time-varying; hence, in order to study the stability of swarm cohesion, we study the dynamics of an error system with

$$e_p^i = x^i - \bar{x}$$

and

$$e_v^i = v^i - \bar{v}$$

Other choices for error systems are also possible and have been used in some studies of swarm stability. For instance, you could use $\tilde{e}_p^i = \sum_{j=1}^{N}\left(x^i - x^j\right)$. This corresponds to computing the errors to each other agent and then trying to get all those errors to go to zero. Note, however, that

$$\tilde{e}_p^i = N\left(x^i - \frac{1}{N}\sum_{j=1}^{N} x^j\right) = N\left(x^i - \bar{x}\right) = N e_p^i$$

The same relationship holds for velocity.

Given the above choices, the error dynamics are given by

$$\begin{aligned} \dot{e}_p^i &= e_v^i \\ \dot{e}_v^i &= \frac{1}{M_i}u^i - \frac{1}{N}\sum_{j=1}^{N}\frac{1}{M_j}u^j \end{aligned} \tag{18.5}$$

The challenge is to specify the u^i so that we get good cohesion properties and successful social foraging.

Assume that each agent can sense its position and velocity relative to $\bar{x}$ and $\bar{v}$, but with some bounded errors. In particular, let $d_p^i(t) \in \Re^n$, $d_v^i(t) \in \Re^n$ be these errors for agent i, respectively. We assume that $d_p^i(t)$ and $d_v^i(t)$ are sufficiently smooth and are independent of the state of the system. Each agent will try to follow the resource profile J_p defined earlier. (We use the plane profile for the sake of illustration, as it will show how swarm dynamics operate over a simple but representative surface.) We assume that each agent senses the gradient of J_p, but with some sufficiently smooth error $d_f^i(t) \in \Re^n$. You may think of this as either a sensing error or as variations (e.g., high frequency ripples) on the resource profile. Below, we will refer to the signals $d_p^i(t)$, $d_v^i(t)$, and $d_f^i(t)$ as "noise" signals, but clearly there is no underlying probability space and all signals in this section are deterministic. You may think of these signals as being generated by, for example, a chaotic dynamic system.

We assume that all the sensing errors are bounded such that

$$\begin{aligned}\|d_p^i\| &\leq D_p \\ \|d_v^i\| &\leq D_v \\ \|d_f^i\| &\leq D_f\end{aligned}$$

where $D_p > 0$, $D_v > 0$, and $D_f > 0$ are known constants. Similar results to what we find below can be found for the more general case, where we have $\|d_p^i\| \leq D_{p_1}\left\|E^i\right\| + D_{p_2}$, $\|d_v^i\| \leq D_{v_1}\left\|E^i\right\| + D_{v_2}$, and $\|d_f^i\| \leq D_f$ where D_{p_1}, D_{p_2}, D_{v_1} and D_{v_2} are known positive constants and E^i is defined in Equation (18.10). See the "For Further Study" section at the end of this part.

Thus, each agent can sense noise-corrupted versions of e_p^i and e_v^i, as

$$\begin{aligned}\hat{e}_p^i &= e_p^i - d_p^i \\ \hat{e}_v^i &= e_v^i - d_v^i\end{aligned}$$

Also, each agent can sense

$$\nabla J_p\left(x^i\right) - d_f^i$$

at the location x^i where the agent is located.

Noise makes it more difficult to get cohesive behavior and degrades foraging effectiveness.

Suppose that in order to steer itself, each agent uses

$$\begin{aligned}u^i &= -M_i k_a \hat{e}_p^i - M_i k_a \hat{e}_v^i - M_i k_v v^i \\ &+ M_i k_r \sum_{j=1, j\neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2}\right)\left(\hat{e}_p^i - \hat{e}_p^j\right) \\ &- M_i k_f\left(\nabla J_p\left(x^i\right) - d_f^i\right)\end{aligned} \tag{18.6}$$

Here, we assume that each agent knows its own mass M_i and velocity v^i. The parameter $k_v > 0$ is the gain for a "velocity damping term." We think of the scalar $k_a > 0$ as the "attraction gain" that indicates how aggressive the agents are in aggregating. The gain k_r is a "repulsion gain," which sets how much the agents want to be away from each other. Note that use of the repulsion term assumes that the i^{th} agent knows, within some errors, the relative distance of all other agents from the swarm center. Also, since

$$\hat{e}_p^i - \hat{e}_p^j = \left(\left(x^i - \bar{x}\right) - d_p^i\right) - \left(\left(x^j - \bar{x}\right) - d_p^j\right) = \left(x^i - x^j\right) - \left(d_p^i - d_p^j\right)$$

we are assuming that the i^{th} agent knows its position (and velocity) relative to each other agent within some bounded errors. Note, however, that when x^i and x^j are far apart, the $\exp(\cdot)$ term is close to zero. (So in effect, if each agent did not use distant agents' values in the repulsion term, we would get approximately the same results.) Also note that if $D_p = D_v = 0$, there is no sensing error on attraction and repulsion, thus, $\hat{e}_p^i = e_p^i$, $\hat{e}_v^i = e_v^i$, and $e_p^i - e_p^j = x^i - x^j$, and we get a repulsion term of the form explained in Equation (18.3). The sensing errors create the possibility that agents will try to move away from each other when they may not really need to, and they may move towards each other when they should not. Clearly, this complicates the ability of the agents to avoid collisions with their neighbors. The last term in Equation (18.6) indicates that each agent wants to move along the negative gradient of the resource profile with the gain k_f proportional to the agent's desire to follow the profile.

Social Foraging in Noise: Groups Can Increase Foraging Effectiveness

Next, we will substitute this choice for u^i into the error dynamics described in Equation (18.5) and study their stability properties. First, however, we will study how the *group* can follow the resource profile in the presence of noise. To do this, consider $\dot{e}_v^i = \dot{v}^i - \dot{\bar{v}}$. First, note that

$$\begin{aligned}
\dot{\bar{v}} &= \frac{1}{N}\sum_{i=1}^{N}\frac{1}{M_i}u^i \\
&= -\frac{k_a}{N}\sum_{i=1}^{N}\left(\hat{e}_p^i + \hat{e}_v^i\right) - \frac{k_v}{N}\sum_{i=1}^{N} v^i \\
&\quad + \frac{k_r}{N}\sum_{i=1}^{N}\sum_{j=1, j\neq i}^{N}\exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2}\right)\left(\hat{e}_p^i - \hat{e}_p^j\right) \\
&\quad - \frac{k_f}{N}\sum_{i=1}^{N}\left(R - d_f^i\right)
\end{aligned} \tag{18.7}$$

Notice that

$$\frac{1}{N}\sum_{i=1}^{N}\hat{e}_p^i = \frac{1}{N}\sum_{i=1}^{N}\left((x^i - \bar{x}) - d_p^i\right) = \bar{x} - \frac{1}{N}N\bar{x} - \frac{1}{N}\sum_{i=1}^{N} d_p^i = -\frac{1}{N}\sum_{i=1}^{N} d_p^i$$

Also, the term due to repulsion in Equation (18.7) is zero as we show next. Note that

$$\sum_{i=1}^{N}\sum_{j=1, j\neq i}^{N}\exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2}\right)\left(\hat{e}_p^i - \hat{e}_p^j\right) =$$

$$\left[\sum_{i=1}^{N}\hat{e}_p^i\sum_{j=1, j\neq i}^{N}\exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2}\right)\right]$$

$$- \left[\sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2} \right) \hat{e}_p^j \right] \tag{18.8}$$

The last term in Equation (18.8)

$$\sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2} \right) \hat{e}_p^j = \sum_{j=1}^{N} \hat{e}_p^j \sum_{i=1, i \neq j}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^j - \hat{e}_p^i\|^2}{r_s^2} \right)$$

and since

$$\exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2} \right) = \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^j - \hat{e}_p^i\|^2}{r_s^2} \right)$$

we have this the same as

$$\sum_{j=1}^{N} \hat{e}_p^j \sum_{i=1, i \neq j}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2} \right) = \sum_{i=1}^{N} \hat{e}_p^i \sum_{j=1, j \neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2} \right)$$

but this last value is the same as the first term on the right-hand side of Equation (18.8). So overall its value is zero. This gives us

$$\dot{\bar{v}} = \frac{k_a}{N} \sum_{i=1}^{N} d_p^i + \frac{k_a}{N} \sum_{i=1}^{N} d_v^i + \frac{k_f}{N} \sum_{i=1}^{N} d_f^i - k_v \bar{v} - k_f R$$

Letting $\bar{d}_p(t) = \frac{1}{N} \sum_{i=1}^{N} d_p^i(t)$ and similarly for $\bar{d}_v(t)$ and $\bar{d}_f(t)$, we get

$$\dot{\bar{v}} = -k_v \bar{v} + \underbrace{k_a \bar{d}_p + k_a \bar{d}_v + k_f \bar{d}_f - k_f R}_{z(t)} \tag{18.9}$$

This is an exponentially stable system with a time-varying but bounded input $z(t)$ so we know that $\bar{v}(t)$ is bounded. To see this, choose a Lyapunov function

$$V_{\bar{v}} = \frac{1}{2} \bar{v}^\top \bar{v}$$

defined on $D = \{\bar{v} \in \Re^n \mid \|\bar{v}\| < r_v\}$ for some $r_v > 0$, and we have

$$\dot{V}_{\bar{v}} = \bar{v}^\top \dot{\bar{v}} = -k_v \bar{v}^\top \bar{v} + z(t)^\top \bar{v}$$

with

$$\left\| \frac{\partial V_{\bar{v}}}{\partial \bar{v}} \right\| = \|\bar{v}\|$$

Note that $\|z(t)\| \leq \left\|k_a \bar{d}_p\right\| + \left\|k_a \bar{d}_v\right\| + \left\|k_f \bar{d}_f\right\| + \|k_f R\| \leq \delta$, where $\delta = k_a D_p + k_a D_v + k_f D_f + k_f \|R\|$. If $\delta < k_v \theta r_v$ for all $t \geq 0$ for some positive constant $\theta < 1$, and all $\bar{v} \in D$, then it can be proven that for all $\|\bar{v}(0)\| < r_v$, and some finite T, we have

$$\|\bar{v}(t)\| \leq \exp\left[-(1-\theta) k_v t\right] \|\bar{v}(0)\|, \ \forall\ 0 \leq t < T$$

and

$$\|\bar{v}(t)\| \leq \frac{\delta}{k_v \theta},\ \forall\ t \geq T$$

Since this holds globally, we can take $r_v \to \infty$ so these inequalities hold for all $\bar{v}(0)$. If δ and θ are fixed, with increasing k_v we get that $\|\bar{v}(t)\|$ decreases faster for $0 \leq t < T$ and smaller bound on $\|\bar{v}(t)\|$ for $t \geq T$. If δ gets larger with k_v and θ fixed, $\|\bar{v}(t)\|$ has larger bound for $t \geq T$; hence, if the magnitude of the noise increases, this increases δ and hence, there can be larger magnitude changes in the ultimate average velocity of the swarm (e.g., the average velocity could oscillate). Note that if in Equation (18.9) $z(t) \approx 0$ (e.g., due to noise that destroys the directionality of the resource profile R), then the above bound may be reduced, but the swarm could be going in the wrong direction.

Regardless of the size of the bound, it is interesting to note that while the noise can destroy the ability of an individual agent to follow a gradient accurately, the average sensing errors of the group is what changes the direction of the group's movement relative to the direction of the gradient of $\bar{J_p}(x)$. In some cases when the swarm is large (N big), it can be that $\bar{d_p} \approx \bar{d_v} \approx \bar{d_f} \approx 0$ to give a zero average sensing error and the group will perfectly follow the proper direction for foraging. (This may be a reason why, for some organisms, large group size is favorable.) In the case when $N = 1$ (i.e., single agent), there is no opportunity for a cancellation of the sensor errors; hence, an individual may not be able to climb a noisy gradient as easily as a group, and in some cases, a group may be able to follow a profile where an individual cannot. This characteristic has been found in biological swarms [230]. From an optimization perspective, you should think of an individual trying to execute a gradient optimization method, which we know can result in it getting stuck in local minima. The group is producing a type of approximation to the gradient by a larger spatial sampling and attraction/repulsion terms. Intuitively, it filters out the noise and moves in the proper direction. Of course, the group itself can get stuck in a local minimum if the basin of attraction of that minimum is large.

Social groups can climb noisy gradients better than individuals.

It is also important to note that there is an intimate relationship between sensor noise and observations of biological swarms (e.g., in bee swarms [458]) in that there is a type of "inertia" of a swarm. Note that for large swarms (N big), there can be regions where the average sensor noise is small, so that agents in that region move in the right direction. In other regions there may be alignments of the errors and hence, the agents may not be all moving in the right direction so they may get close to each other and impede each other's motion, having the effect of slowing down the whole group. With no noise, the group inertia effect is not found, since each agent is moving in the right direction. The presence of sensor noise generally can make it more difficult to get the group moving in the right direction (e.g., for foraging, migration, or movement to a nest site). Large swarms can help move the group in the right direction, but at the expense of possibly slowing their movement initially in a transient period.

Cohesive Social Foraging in Noise

Next, we return to the problem of finding the error dynamics and then stability analysis by considering the $\dot{v}^i$ term of $\dot{e}_v^i = \dot{v}^i - \dot{\bar{v}}$ in the error dynamics of Equation (18.5). Note that

$$\begin{aligned}
\dot{v}^i = \frac{1}{M_i}u^i &= -k_a\hat{e}_p^i - k_a\hat{e}_v^i - k_v v^i \\
&+ k_r \sum_{j=1, j\neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2}\right)\left(\hat{e}_p^i - \hat{e}_p^j\right) - k_f\left(\nabla J_p\left(x^i\right) - d_f^i\right) \\
&= -k_a e_p^i + k_a d_p^i - k_a e_v^i + k_a d_v^i - k_v v^i \\
&+ k_r \sum_{j=1, j\neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2}\right)\left(\hat{e}_p^i - \hat{e}_p^j\right) - k_f\left(R - d_f^i\right)
\end{aligned}$$

Hence, we have

$$\begin{aligned}
\dot{e}_v^i = \dot{v}^i - \dot{\bar{v}} &= -k_a e_p^i - k_a e_v^i - k_v e_v^i + k_a\left(d_p^i - \bar{d}_p\right) + k_a\left(d_v^i - \bar{d}_v\right) \\
&+ k_r \sum_{j=1, j\neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\left(x^i - x^j\right) - \left(d_p^i - d_p^j\right)\|^2}{r_s^2}\right)\left(\left(x^i - x^j\right)\right. \\
&- \left.\left(d_p^i - d_p^j\right)\right) + k_f\left(d_f^i - \bar{d}_f\right)
\end{aligned}$$

To study the stability of the error dynamics, and hence, swarm cohesiveness, define

$$E^i = \left[{e_p^i}^\top,\ {e_v^i}^\top\right]^\top \tag{18.10}$$

and $E = \left[{E^1}^\top, {E^2}^\top, \ldots, {E^N}^\top\right]^\top$, and choose a Lyapunov function

$$V(E) = \sum_{i=1}^{N} V_i\left(E^i\right)$$

where

$$V_i\left(E^i\right) = {E^i}^\top P E^i$$

with $P = P^\top$ and $P > 0$ (a positive definite matrix). We know that

$$\lambda_{min}(P){E^i}^\top E^i \leq {E^i}^\top P E^i \leq \lambda_{max}(P){E^i}^\top E^i$$

Notice that with I an $n \times n$ identity matrix, we have

$$\dot{E}^i = \underbrace{\begin{bmatrix} 0 & I \\ -k_a I & -\left(k_a + k_v\right)I \end{bmatrix}}_{A} E^i + \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{B} g^i(E)$$

where

$$\begin{aligned} g^i(E) &= k_a\left(d_p^i - \bar{d}_p\right) + k_a\left(d_v^i - \bar{d}_v\right) \\ &+ k_r \sum_{j=1, j\neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\hat{e}_p^i - \hat{e}_p^j\|^2}{r_s^2}\right)\left(\hat{e}_p^i - \hat{e}_p^j\right) \\ &+ k_f\left(d_f^i - \bar{d}_f\right) \end{aligned} \quad (18.11)$$

Note that any matrix

$$\begin{bmatrix} 0 & I \\ -k_1 I & -k_2 I \end{bmatrix}$$

with $k_1 > 0$ and $k_2 > 0$ has eigenvalues given by the roots of $\left(s^2 + k_2 s + k_1\right)^n$, which are in the strict left half plane. Since $k_a > 0$ and $k_v > 0$, the matrix A above is Hurwitz (i.e., has eigenvalues all in the strict left half plane).

We have

$$\dot{V}_i = {E^i}^\top P \dot{E}^i + {\dot{E}^i}{}^\top P E^i = {E^i}^\top \left(PA + A^\top P\right) E^i + 2{E^i}^\top P B g^i(E) \quad (18.12)$$

Note that if $-Q = \left(PA + A^\top P\right)$, then Q is such that $Q = Q^\top$ and $Q > 0$, and the unique solution P of $PA + A^\top P = -Q$ has $P = P^\top$ and $P > 0$ as needed. Also, since $\|B\| = 1$, ${E^i}^\top Q E^i \geq \lambda_{min}(Q){E^i}^\top E^i$, and $\|P\| = \lambda_{max}(P)$ with $P = P^\top > 0$, we have

$$\begin{aligned} \dot{V}_i &\leq -\lambda_{min}(Q)\left\|E^i\right\|^2 + 2\left\|E^i\right\|\lambda_{max}(P)\|g^i(E)\| \\ &= -\lambda_{min}(Q)\left(\left\|E^i\right\| - \frac{2\lambda_{max}(P)}{\lambda_{min}(Q)}\|g^i(E)\|\right)\left\|E^i\right\| \end{aligned} \quad (18.13)$$

Suppose for a moment that for each $i = 1, 2, \ldots, N$, $\|g^i(E)\| < \beta$ for some known β. Then, if

$$\left\|E^i\right\| > \frac{2\lambda_{max}(P)}{\lambda_{min}(Q)}\|g^i(E)\| \quad (18.14)$$

we have that $\dot{V}_i < 0$. Hence, the set

$$\Omega_b = \left\{E : \ \left\|E^i\right\| \leq 2\frac{\lambda_{max}(P)}{\lambda_{min}(Q)}\|g^i(E)\|, \ i = 1, 2, \ldots, N\right\} \quad (18.15)$$

is attractive and compact. Also we know that within a finite amount of time, $E^i \to \Omega_b$. This means that we can guarantee that if the swarm is not cohesive, it will seek to be cohesive, but this can only be guaranteed if it is a certain distance from cohesiveness, as indicated by Equation (18.14).

The positions and velocities of all agents can oscillate, yet overall swarm cohesiveness can be maintained.

It remains to show that for each i, $\|g^i(E)\| < \beta$ for some β. Note that

$$\|g^i(E)\| \leq k_a\|d_p^i - \bar{d}_p\| + k_a\|d_v^i - \bar{d}_v\| + k_f\|d_f^i - \bar{d}_f\| + k_r \sum_{j=1, j\neq i}^{N} \exp\left(\frac{-\frac{1}{2}\|\psi\|^2}{r_s^2}\right)\|\psi\| \quad (18.16)$$

where $\psi = \hat{e}_p^i - \hat{e}_p^j = \left(x^i - x^j\right) - \left(d_p^i - d_p^j\right)$. Notice that $\frac{1}{N}\sum_{j=1}^N \|d_p^j\| \leq D_p$ since $\|d_p^j\| \leq D_p$. Also

$$d_p^i - \frac{1}{N}\sum_{j=1}^N d_p^j \leq \|d_p^i\| + \frac{1}{N}\|\sum_{j=1}^N d_p^j\| \leq \|d_p^i\| + \frac{1}{N}\sum_{j=1}^N \|d_p^j\|$$

$\|d_p^i - \bar{d}_p\| \leq 2D_p$, $\|d_v^i - \bar{d}_v\| \leq 2D_v$, and $\|d_f^i - \bar{d}_f\| \leq 2D_f$.

For the last term in Equation (18.16), note that as $\|x^i - x^j\|$ becomes large for all i and j, the agents are all far from each other and the repulsion term goes to zero. Also, the term due to the repulsion is bounded with a unique maximum point. To find this point, note that

$$\frac{\partial}{\partial\|\psi\|}\left(\|\psi\|\exp\left(\frac{-\frac{1}{2}\|\psi\|^2}{r_s^2}\right)\right) = \exp\left(\frac{-\frac{1}{2}\|\psi\|^2}{r_s^2}\right) - \frac{\|\psi\|^2}{r_s^2}\exp\left(\frac{-\frac{1}{2}\|\psi\|^2}{r_s^2}\right)$$

The maximum point occurs at a point such that

$$1 - \frac{\|\psi\|^2}{r_s^2} = 0$$

or when $\|\psi\| = r_s$. Hence, we have

$$\begin{aligned}\|g^i(E)\| &\leq 2k_a\left(D_p + D_v\right) + 2k_f D_f + k_r \sum_{j=1, j\neq i}^N \exp\left(-\frac{1}{2}\right) r_s \\ &= 2k_a\left(D_p + D_v\right) + 2k_f D_f + k_r r_s (N-1)\exp\left(-\frac{1}{2}\right) = \beta\end{aligned}$$

If you substitute this value for β into Equation (18.15), you get the set Ω_b that ultimately all the trajectories will end up in.

Cohesive Social Foraging with No Noise: Optimization Perspective

When there is no noise, tighter bounds and stronger results can be obtained. First, we can eliminate the effect of P via $\lambda_{max}(P)$ on the bound for the no-noise case. Assume there is no sensor noise so $D_p = D_v = D_f = 0$. Choose

$$\begin{aligned}u^i &= -M_i k_a e_p^i - M_i k_a e_v^i - M_i k_v v^i \\ &\quad + M_i k_r \left(B^\top P^{-1} B\right) \sum_{j=1, j\neq i}^N \exp\left(\frac{-\frac{1}{2}\|e_p^i - e_p^j\|^2}{r_s^2}\right)\left(e_p^i - e_p^j\right) \\ &\quad - M_i k_f R \end{aligned} \tag{18.17}$$

where $P = P^\top$, $P > 0$ was defined earlier, so P^{-1} exists. Also

$$\dot{V}_i \leq -\lambda_{min}(Q)\left\|E^i\right\|^2$$

$$\begin{aligned}
&+ \quad 2E^{i\top}PB\left(k_rB^\top P^{-1}B\sum_{j=1,j\neq i}^{N}\exp\left(\frac{-\frac{1}{2}\|e_p^i-e_p^j\|^2}{r_s^2}\right)\left(e_p^i-e_p^j\right)\right)\\
&= \quad -\lambda_{min}(Q)\left\|E^i\right\|^2+2k_rE^{i\top}B\sum_{j=1,j\neq i}^{N}exp\left(\frac{-\frac{1}{2}\|e_p^i-e_p^j\|^2}{r_s^2}\right)\left(e_p^i-e_p^j\right)\\
&\leq \quad -\lambda_{min}(Q)\left\|E^i\right\|^2+2k_r\left\|E^i\right\|(N-1)\exp\left(-\frac{1}{2}\right)r_s
\end{aligned}$$

So $\dot{V}_i<0$ if $\left\|E^i\right\|>\frac{2k_r(N-1)r_s}{\lambda_{min}(Q)}\exp\left(-\frac{1}{2}\right)$. Let

$$\Omega_b'=\left\{E:\ \left\|E^i\right\|\leq\frac{2k_rr_s(N-1)}{\lambda_{min}(Q)}\exp\left(-\frac{1}{2}\right),\ i=1,2,\ldots,N\right\}$$

Next, note that in the set Ω_b, we have bounded e_p^i and e_v^i but we are not guaranteed that $e_v^i\to0$ for any i. Achieving $e_v^i\to0$ for all i would be a desirable property, since this represents that $v^i=\bar{v}$ for all i so that the group will all move cohesively in the same direction. To study this, consider Ω_b', and consider a Lyapunov function $V^o(E)=\sum_{i=1}^{N}V_i^o\left(E^i\right)$ with

$$V_i^o\left(E^i\right)=\frac{1}{2}k_ae_p^{i\top}e_p^i+\frac{1}{2}e_v^{i\top}e_v^i+k_rr_s^2\sum_{j=1,j\neq i}^{N}\exp\left(\frac{-\frac{1}{2}\|e_p^i-e_p^j\|^2}{r_s^2}\right)$$

Note that this Lyapunov function satisfies $V_i^o\left(E^i\right)\geq0$. You should view the objective of the agents as being that of *minimizing* this Lyapunov function; they try to minimize the distance to the center of the swarm, match the average velocity of the group, and minimize the repulsion effect (to do that, the agents move away from each other). We have

$$\begin{aligned}
\nabla_{e_p^i}V_i^o &= k_ae_p^i-k_r\sum_{j=1,j\neq i}^{N}\exp\left(\frac{-\frac{1}{2}\|e_p^i-e_p^j\|^2}{r_s^2}\right)\left(e_p^i-e_p^j\right)\\
\nabla_{e_v^i}V_i^o &= e_v^i
\end{aligned}$$

so

$$\begin{aligned}
&\dot{V}_i^o=\left(\nabla V_i^o\left(E^i\right)\right)^\top\dot{E}_i\\
&= \quad k_ae_p^{i\top}e_v^i-k_r\sum_{j=1,j\neq i}^{N}\exp\left(\frac{-\frac{1}{2}\|e_p^i-e_p^j\|^2}{r_s^2}\right)\left(e_p^i-e_p^j\right)^\top e_v^i\\
&+ \quad e_v^{i\top}\left(-k_ae_p^i-k_ae_v^i-k_ve_v^i+k_r\sum_{j=1,j\neq i}^{N}\exp\left(\frac{-\frac{1}{2}\|e_p^i-e_p^j\|^2}{r_s^2}\right)\left(e_p^i-e_p^j\right)\right)\\
&= \quad -\left(1+k_v\right)e_v^{i\top}e_v^i
\end{aligned}$$

Hence,

$$\dot{V}^o = -(1+k_v)\sum_{i=1}^{N} \|e_v^i\|^2 \leq 0$$

on $E \in \Omega$ for a compact set Ω. Choose Ω so it is positively invariant, which is clearly possible, and so $\Omega_e \in \Omega$ where

When noise is not present, ultimately the velocity of all agents becomes the same and the swarm moves directly down the resource profile.

$$\Omega_e = \{E: \ \dot{V}^o(E) = 0\} = \{E: \ e_v^i = 0, \ i = 1, 2, \ldots, N\}$$

From LaSalle's Invariance Principle, we know that if $E(0) \in \Omega$ then $E(t)$ will converge to the largest invariant subset of Ω_e. Hence,

$$e_v^i(t) \to 0$$

as $t \to \infty$. When $R = 0$ (no resource profile effect), $\bar{v}(t) \to 0$ and hence $v^i(t) \to 0$ as $t \to \infty$ for all i (i.e., ultimately no oscillations in the average velocity). If $R \neq 0$, then $\dot{\bar{v}} = -k_v\bar{v} - k_f R$ and $\bar{v}(t) \to -\frac{k_f}{k_v}R$ as $t \to \infty$, and thus, $v^i(t) \to -\frac{k_f}{k_v}R$ for all i as $t \to \infty$, i.e, the group follows the profile. These results help to highlight the effects of the noise. The noise makes it so that the swarm may not follow the profile as well (but makes following it possible when it may not be possible for a single individual), and it destroys tight cohesion characterized by getting $e_v^i(t) \to 0$. Next, we will study additional characteristics of swarms by analyzing the results of this and the previous sections in more detail.

18.4.4 Cohesion Characteristics and Swarm Dynamics

Here, we will study the effects of various parameters on cohesion characteristics and then provide a simulation to provide insight into swarm dynamics, especially transient behavior. Suppose that u^i is given by Equation (18.6).

Effects of Parameters on Swarm Size

The size of Ω_b in Equation (18.15), which we denote by $|\Omega_b|$, is directly a function of several known parameters. Consider the following cases:

Attraction gains should be set high to get tight swarm cohesion, but not too high or the attraction will amplify the noise and swarm compactness can degrade.

- *No sensing errors:* If there are no sensing errors, i.e., $D_p = D_v = D_f = 0$, and if $Q = k_a I$, we obtain

  $$\Omega_b = \left\{E: \ \|E^i\| \leq \frac{2k_r r_s(N-1)}{k_a}\lambda_{max}(P)\exp\left(-\frac{1}{2}\right), \ i = 1, 2, \ldots, N\right\}$$

  If N, k_r, and r_s are fixed, then if k_a increases from zero, we get $\frac{\lambda_{max}(P)}{k_a} \to 1$ from above and we get a decrease in $|\Omega_b|$, but only up to a certain point.

- *Sensing errors:* There are several characteristics of interest:

  - *Noise cancellations:* In the special situation when $d_p^i = d_p^j$, $d_v^i = d_v^j$, and $d_f^i = d_f^j$ for all i and j, then $d_p^i - \bar{d}_p = d_v^i - \bar{d}_v = d_f^i - \bar{d}_f$ for all i and it is as if there is no error and $|\Omega_b|$ is smaller.

– *Repel effects:* For fixed values of N, k_a, and k_r if we increase r_s, each agent has a larger region from which it will repel its neighbors so $|\Omega_b|$ is larger. For fixed k_r, k_a, and r_s, if we let $N \to \infty$, then $|\Omega_b| \to \infty$ as we expect due to the repulsion. (The bound is conservative since it depends on the special case of all agents being aligned on a line so there are $N-1$ inter-agent distances that sum to make the bound large.)

– *Attraction can amplify noise:* Let $D_s = D_p + D_v$ and J quantify the size of the set Ω_b. Next, we study the special case of choosing $Q = k_a I$. Fix all values of the parameters except k_a and D_s. A plot of J versus k_a and D_s is shown in Figure 18.14, where the locus of points are those values of k_a that minimize J for each given value of D_s. This plot shows that if there is a set magnitude of the noise,

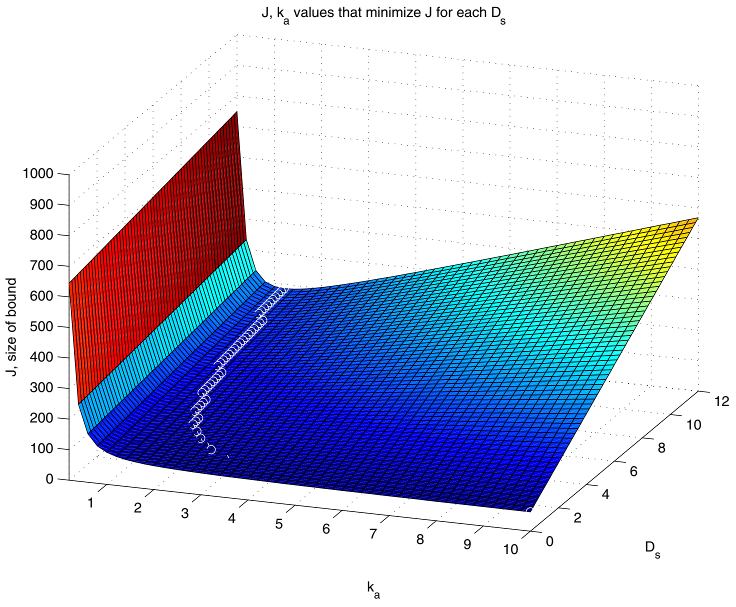

Figure 18.14: Values of k_a that minimize J for given values of noise magnitude D_s.

then to get the best cohesiveness (smallest Ω_b), k_a should not be too small (or it would not hold the group together), but also not too large since then the noise is also amplified by the attraction gain and poor cohesion results. Could you interpret the plot as a type of fitness function, and then come to conclusions about the evolution of the agent parameters?

- *Swarm size N:* In some situations, when N is very large, $\bar{d_p} \approx \bar{d_v} \approx \bar{d_f} \approx 0$ and there is no biasing of sensing errors so that on average they are zero and this reduces the above bound on $\|g^i(E)\|$.

For additional analysis of swarm properties, see the "For Further Study" section at the end of this part.

Swarm Dynamics: Individual and Group

Here, we will simply simulate a swarm for the no noise case in order to provide some insights into the dynamics, especially the transient behavior. We will in particular seek to study the individual motions and how they collectively move as a group to achieve cohesion, and the dynamics of the motion of the group. We use linear attraction, velocity damping, the Gaussian form for the repulsion term, and the resource profile with the shape of a plane. The parameters for the simulation are $N = 50$, $k_a = 1$, $k_r = 10$, $r_s^2 = 0.1$, $k_v = k_f = 0.1$, $R = [1, 2, 3]^\top$, and $r_o = 0$. Simulating for 10 sec. we get the agent trajectories in Figure 18.15.

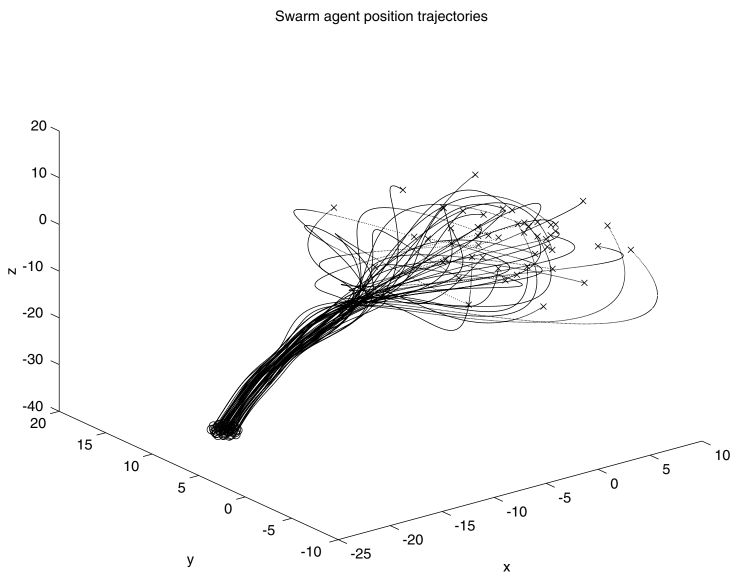

Figure 18.15: Agent trajectories in a swarm.

Notice that initially the agents move to achieve cohesion. By the end of the simulation, the agents are moving at the same velocity and as a group, with some constant inter-agent spacing between each pair of agents. Moreover, due to the choice of initial conditions (actually random), the group achieves a certain level of aggregation relatively quickly, then the *group* moves to follow the foraging profile. Of course, some orientation towards following the profile is achieved during the initial aggregation period, but in this simulation, significant reorientation towards following the resource profile occurs *after* there is tight aggregation.

18.5 Design Example: Robot Swarms

This problem is an extension of the one in Section 6.2, where we sought to develop a guidance algorithm for moving a robot from an initial position to a goal position in a factory while avoiding collisions with obstacles. Here, we consider one approach to guiding multiple robots through the same maze as was studied there, using the swarm approach discussed in the last section.

18.5.1 Robot Swarm Formulation

A robot swarm is simply a group of robots that move in some cohesive fashion in order to perform some task. Here, the task is simply to get the group to a certain location in a factory, so that they can perform some activity together there. In moving them from one location to another, the key challenge is to have them all avoid certain obstacles that appear in their paths. Here, we use the same maze as described in Section 6.2 with the same final goal position. We use the same "obstacle function" defined there to represent the positions of the obstacles and the same "goal function" to represent where we want the group to go. To solve this problem, we view the combined obstacle/goal functions as a cost function just as we did in the chapter on planning. Here, however, we take the view that the group of robots is engaged in social foraging over the cost function, which we think of as a nutrient profile. The obstacles are thought of as regions with a noxious substance, and increasing amounts of food are obtained by moving towards the goal position. We use a different model for the robot from the one considered in Section 6.2. Here, we use the model of a swarm agent from the last section with all the masses of the robots the same, at $M_i = 1$.

Cooperative guidance strategies based on swarms are useful for groups of robots.

We assume that each vehicle knows its own velocity. Here, we will first consider the case where each robot perfectly knows the swarm center and swarm average velocity. For some types of robotic systems, this would require each robot to know the positions and velocities of all the other robots so it can compute the swarm center and average velocity, but for others there may be a sensor which could directly sense these. We will also consider the case where there is noise in sensing, of the type described in Design Problem 18.7, so that each robot does not perfectly know the swarm center and average velocity.

18.5.2 Performance in Obstacle Avoidance and Noise Effects

Suppose that there are 30 robots in the swarm. Let $k_p = 1$, $k_v = 0.1$, $k_f = 0.1$, $k_r = 10$, and $r_s^2 = 1$. We pick some random initial locations and velocities, but within some fixed ranges. We pick $w_1 = 120$ and $w_2 = 0.1$ after some tuning. Also, to keep the simulation simple, we use an Euler approximation and simulate the swarm as a discrete-time system. For this, we use a sampling period of 0.01 and simulate for 80 sec. Clearly, due to the use of a different form of a "resource profile" and the discrete-time approach, the stability results of the previous section do not apply directly; however, the simulations will illustrate that the basic ideas still do apply.

The noise-free case is shown in Figure 18.16. This shows the position trajectories of the 30 robots in moving from an initial position to the point where they are around the goal position. As time goes on, the robots slow down and stop when they are near the goal position. If you study the trajectories you will see that they successfully avoid the obstacles and still maintain a cohesive group. The ultimate size of the group is set by the attraction and repulsion parameters, and the shape of the goal (and obstacle) functions (e.g., if they resulted in a very steep slope near the goal position, then the group of robots would be packed tighter in their final position).

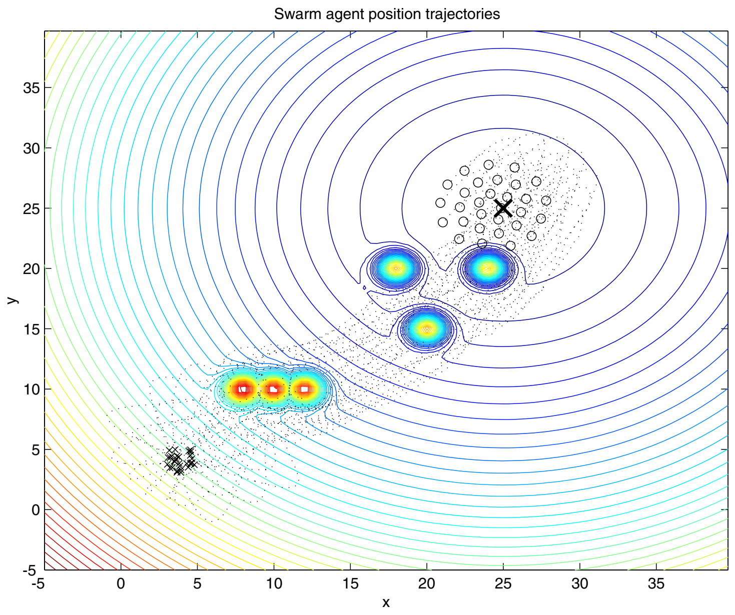

Figure 18.16: Robot swarm, robot position trajectories.

Figure 18.17 shows what happens if the robots sense with noise of the type discussed in Design Problem 18.7. Here, due to the sensing errors, there are actually collisions with some of the obstacles by some of the robots (actually due to noise in use of the obstacle and goal functions in the simulation). Moreover, the positions and velocities of the robots oscillate near the goal position. Clearly, noise can adversely affect cohesion and orderly operation of a group of robots.

18.5.3 Additional Robot Swarm Design Challenges

As outlined in Section 6.2.4, there are many additional challenges that can arise in autonomous vehicle guidance, and these are compounded in the case where we want to guide multiple autonomous robots. The challenges there included

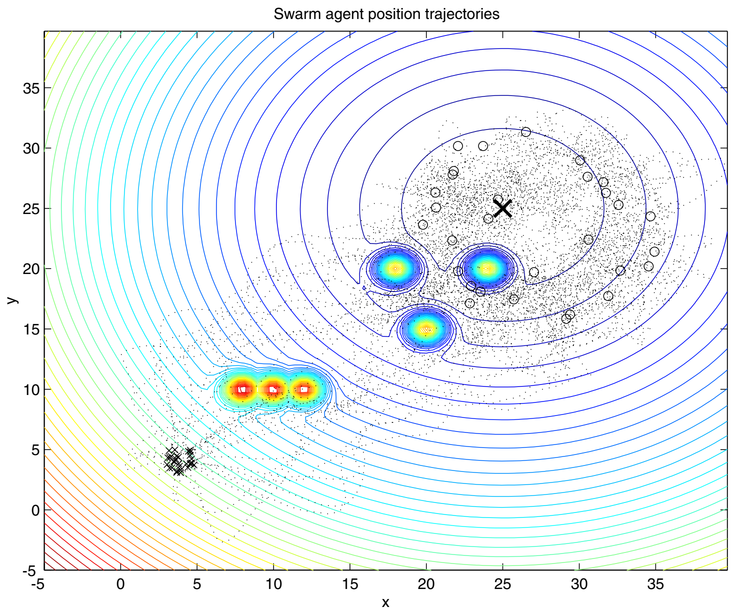

Figure 18.17: Robot swarm, robot position trajectories, noise case.

complex mazes that led to dead ends and circular loops, mobile obstacles, and uncertainty. Dead ends and circular loops that one, or some subset, of the robots get stuck in may be solved in the swarm case by having some robots that are moving in the right direction "pull" them out of the problem (via appropriate desire to stay grouped). However, it can also be that the ones that get stuck in a dead end or circular loop, hold back the others that are headed in the right direction. The problems with mobile obstacles are multiplied for the swarm case, since now the whole group must avoid such obstacles. It may be easier to avoid mobile obstacles if the robots cooperate by telling each other when they sense a mobile obstacle, but the very existence of a swarm creates more problems with mobile obstacles, since each robot in the group can be thought of as a mobile obstacle to avoid (in the last section, we considered point-sized vehicles and hence, avoided the whole issue of inter-robot collisions).

Uncertainty makes each of these issues more challenging, and the swarm can create more problems with uncertainty. For instance, suppose that the group of robots is connected via a communication network, with its "topology" specifying which robots are connected. If this communication network is less than perfect (e.g., via bandwidth constraints, delays, or topology changes), then clearly the maintenance of cohesive robot swarm behavior will be even more challenging. Some of these issues will be discussed in more detail in the "Challenge Problem" in the next chapter.

18.6 Exercises and Design Problems

Exercise 18.1 (*E. coli* Swarm Foraging):

(a) Illustrate for the simulations in the chapter the effects of changing N_c and the parameters of the cell-to-cell attractant function. Explain how to make poor choices for these parameters (e.g,. ones that result in swarming, but little optimization progress on the nutrient concentration profile) and good choices (e.g., ones where cells pull each other toward minima), and illustrate in each case the resulting behavior of the algorithm.

(b) Suppose that you use Rosenbrock's function in Equation (15.9) as a cost function. Make appropriate algorithm parameter choices and illustrate the performance of the algorithm for this function.

Exercise 18.2 (Foraging of Square-Shaped Bacteria):

(a) Based on the brief description the foraging behavior of square bacteria, develop an optimization algorithm that models its foraging behavior. To do this, specify how each bacterium moves forward and backward and changes directions. Ignore reproduction, elimination, and dispersal. Write a simulation of the algorithm.

(b) Use the algorithm to perform optimization over the multiple-extremum function in Figure 18.10. Show plots to illustrate the characteristics of the algorithm.

Exercise 18.3 (ODE Model of Dynamic Labor Force Allocation for Bees): In [89] the authors develop a nonlinear differential equation model of the key functional aspects of dynamic labor force allocation of honey bees and validate the model against T. Seeley and his colleagues' earlier experimental work (see [456]).

(a) Simulate this model, compare to Seeley's results, and explain the key aspects of the model and its properties.

(b) In what ways is the model inaccurate? What does it not represent?

(c) Can you modify the ODEs with some nonlinearities so that the simulations will fit Seeley's data better?

Exercise 18.4 (Effects of Parameters on Swarm Dynamics): In this problem, you will simulate the swarm in Section 18.4.4 for appropriate parameter choices to illustrate different behaviors (this is a noise-free case). Start with the parameter values given there and tune only the parameter indicated in each part.

(a) Pick a value for k_a that will result in a tighter packed ultimate swarm (i.e., a smaller ultimate swarm size).

(b) Pick a value for k_r that will result in a more loosely packed ultimate swarm (i.e., a larger ultimate swarm size).

(c) Repeat (b), but for the r_s parameter.

(d) Repeat (a)–(c) but for the case where there is noise in sensing. Hint: Simulate the noise by generating it as the output of a chaotic system (e.g., via Duffing's equation) so that it satisfies the smoothness requirement on the sensing errors.

Exercise 18.5 (Hard Repulsion, Collision-Avoidance, and Swarm Dynamics): Replace the Gaussian-type repulsion term in Section 18.4 with a repulsion term that will provide a "hard" constraint to avoid collisions between agents. Add velocity damping, a foraging plane, and an appropriate attraction term. Simulate the swarm and demonstrate via plots that there are no collisions both during the transient and in the steady-state. Investigate the effects of all parameters of the controller.

Design Problem 18.1 (Extensions to *E. coli* Swarm Foraging): Study how to find the minimum of the function in Figure 18.10, but model, code, and illustrate the performance and algorithmic characteristics of the algorithm for the following cases.

(a) Change the nutrient concentration to represent consumption of nutrients.

(b) Model Brownian effects so that the bacteria cannot swim straight.

(c) Make the tumble and run lengths exponentially distributed random variables consistent with what has been found in nature.

(d) Make run-length decisions based on the past 4 sec. of concentrations.

(e) Develop a fully asynchronous model.

(f) Allow a time-varying population size.

(g) A more biologically accurate model of the swarming behavior of certain bacteria is given in [544]. Simulate the partial differential equation model given there.

(h) Develop other criteria by which bacteria split. Add effects of conjugation and other evolutionary characteristics (e.g., evolve $C(i)$, N_s, and N_c).

Design Problem 18.2 (*M. xanthus* Swarm Foraging)*: Review the current literature and develop a cellular automaton model of the foraging of *M. xanthus*. Include the modeling of fruiting bodies and the full lifecycle of the bacteria.

Design Problem 18.3 (Robot Swarms): In this problem, you will investigate aspects of the robot swarm problem studied in Section 18.5.

(a) Explain via simulations, the effects of k_p, k_v, k_r, and r_s on the transient and ultimate (as $t \to \infty$) swarm behavior (i.e., show in simulation the effects of changing each of these parameters and, for example, explain whether the swarm size (diameter) increases or decreases).

(b) Design a new control that is based on a "hard repel" and demonstrate via simulations that you can design it so that there will be no inter-robot collisions and no collisions with obstacles. This may require retuning the parameters of the simulation.

(c) Repeat (a) and (b) except simulate the system as a continuous-time system. Hint: This will require using the gradient of the cost function that holds the goal and obstacle functions.

(d) Repeat (a), (b), and (c) but for the case where noise of the type described in Design Problem 18.7 is used.

(e) Define a communication topology and an "intelligent" strategy (e.g., one that uses planning, learning, and/or attention) for each of the robots to coordinate their actions to solve the problem. Simulate to evaluate performance.

Design Problem 18.4 (Social Foraging Strategies for Indirect/Direct Adaptive Control): In this problem, you will study the development of indirect and direct adaptive controllers for the process control problem studied in Sections 12.4 and 12.6, but where foraging algorithms are used for the optimization method. First, read the part in the chapter on the genetic adaptive control strategies in Section 16.5. Basically, you will simply replace the online genetic algorithm with a foraging algorithm. This changes how we view the operation of the algorithm. From a foraging perspective, we view θ^i as the location of the i^{th} forager in its environment. In a foraging method, we will move the position of the forager θ^i so as to minimize $J(\theta^i)$. The particular manner used to adjust the $\theta^i(k-1)$ to find $\theta^i(k)$ will depend on the choice of the foraging algorithm steps (e.g., will it involve swarming, or other communications between individuals that tell each other when they are doing well?). In an indirect adaptive control strategy, we view foraging as searching for good model information. If a foraging strategy is used, we view $\theta(k)$ in Equation (16.2) as the forager who has found the best model information. With a foraging strategy, we could view the fixed-position members that were discussed in the chapter as "information centers," if the foragers were endowed with communication capabilities. If such centers have good model information, they will tend to attract foragers. Clearly, you can specify a direct adaptive control strategy based on social foraging in a similar way to how we did for direct genetic adaptive control.

Next, we show how to develop an indirect adaptive controller based on foraging for the tank problem. The problem given after that will focus on various extensions of the method.

Our forager's position in one dimension is given by θ_α and in the other dimension by θ_β so our i^{th} forager's position is $\theta^i = [\theta^i_\alpha, \theta^i_\beta]^\top$, $i = 1, 2, \ldots, S$. We choose $S = 10$ as the population size. The foraging strategy we employ is the one from Section 18.3, which is based on *E. coli* chemotaxis, but without swarming, elimination-dispersal, and reproduction. Hence, we only use the chemotactic hill-climbing strategy to adjust the parameters. At each time step, we take one foraging step, which for our foraging strategy means that we use either one tumble-tumble step or part of a "run." In this way, the foraging occurs while the control system operates with foraging (searching) for parameters occurring at each time step. For instance, if over one time step the cost did not decrease for an individual, then there is a tumble, and to do this, we generate a random direction and update the parameters (location of the forager) in that direction. If, however, the cost improved from the last step, then another step in the same direction taken last time is made (provided not too many steps in that direction have already been made). In this case, the forager is on a "run" in a good direction, down the cost function.

The step size $C(i,k)$ is set to be 0.05 for all bacteria for all times. The maximum number of steps along a good direction is $N_s = 4$, and $\theta^i_\alpha(0) = 2$, $\theta^i_\beta(0) = 0.5$, $i = 1, 2, \ldots, S$. We use the cost evaluation procedure with $J_s(\theta^i(k-1), N)$ in Equation (16.3) with $N = 100$. To keep things simple, we will simply use $\theta_\alpha(k) = \theta_\alpha(0)$ and $\theta_\beta(k) = \theta_\beta(0)$ for $k = 1, 2, \ldots, N+1$ (i.e., we wait till we have computed all the values needed for the cost evaluation before we start the adaptation procedure). Clearly, other initialization choices are possible. For instance, in the beginning you could use an expanding length window of data to evaluate the quality of the estimators.

The performance of the closed-loop system is illustrated in Figure 18.18, where we see that, after an initial transient period that results in part due to the poor initialization of estimators and the controller start-up method, we get reasonably good tracking of the reference input. Next, Figure 18.19 shows that the estimate of the tank liquid level is quite good, even though at times the individual estimates of the nonlinearities are not.

To further illustrate some properties of the adaptive controller, see Figure 18.20 where we plot the cost of the best individual in the population (the one that leads to the specification of the controller) and the index i of the best individual in the population for every time step. First, note that early in the simulation, cost is zero due to how we start up the controller. Then, when we start the controller at $t = 10$ sec., the cost jumps to a relatively high value; this represents that we have a poor initialization for the population. After some time, however, the foraging strategy is somewhat successful at adjusting the population members so that the estimation error decreases and hence, the best cost decreases. Note, however, that the cost does not *always* decrease over time. It can also increase and one cause of this can be the change in the reference input. Next, note that in the

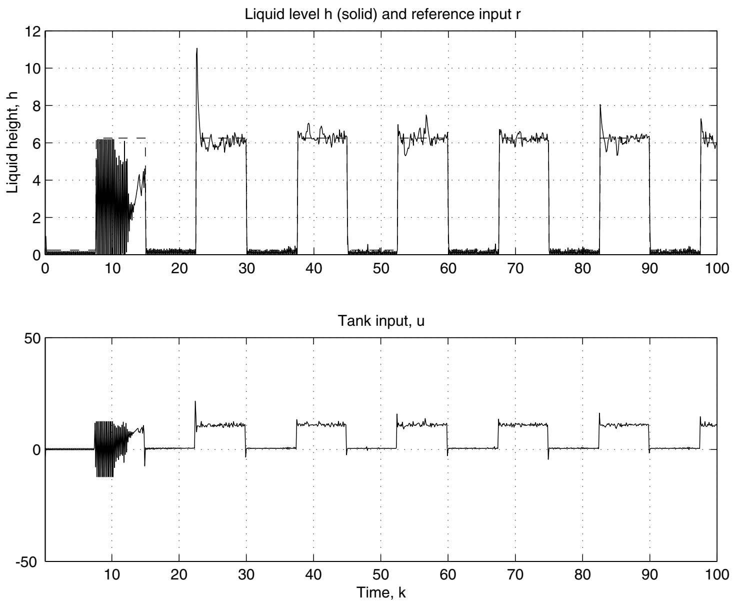

Figure 18.18: Indirect adaptive controller based on foraging.

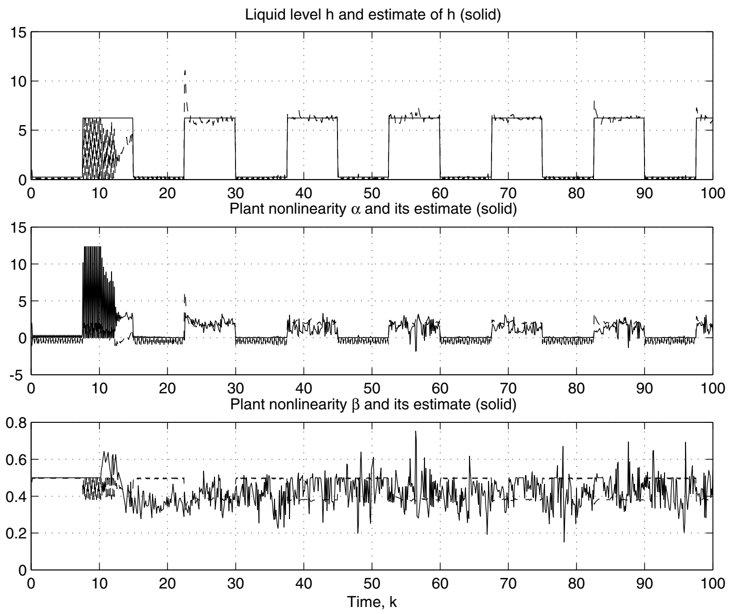

Figure 18.19: Indirect adaptive controller based on foraging, estimates of liquid level, and nonlinearities.

bottom plot we show the index of the best individual in the population at each step. Notice that there are some short stretches of time where the best individual does not change; however, there is a significant amount of switching between different members of the population that provide better estimates at different times.

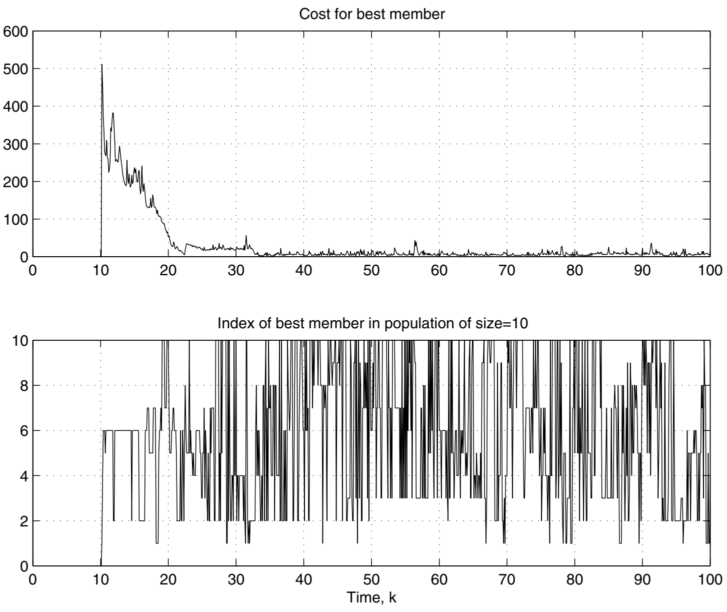

Figure 18.20: Indirect adaptive controller based on foraging, best cost, and index of best individual in the population.

(a) Develop an indirect adaptive controller for the process control problem using a foraging strategy for parameter adjustments. You may build your approach above where an *E. coli* chemotactic foraging strategy was simulated, but you must add some additional feature to the foraging algorithm that represents a situation in nature, where foragers can communicate to each other how well they are doing and subsequently, use that information to improve their foraging success. Regardless of which approach you use, verify the operation of your controller in the same manner as was done above. Study the effect of the choice of the reference input on the ability of the approximator mappings to match the underlying unknown nonlinearities. Provide plots to illustrate the quality of the matching as was done in the chapter.

(b) Develop a direct adaptive controller for the process control problem that is based on a foraging strategy that includes some type of communication of information between foragers, and subsequent use of that information to improve foraging success. Regardless of which foraging approach you use, verify the operation of your controller in the same manner as was done above. Study the effect of the choice of the reference input on the ability of the approximator mapping to match the "ideal" controller nonlinearity discussed in the chapter. Provide plots to illustrate the quality of the matching as was done in the chapter.

(c) Compare the performance of the indirect and direct methods and discuss. Evaluate the value of using fixed portions of the population of models or controllers. Study the effects of good population initialization.

Design Problem 18.5 (Foraging in Colombia)*: This problem continues with Design Problem 14.2. Suppose that you have a group of foragers that find more food at higher elevations. Design a foraging algorithm that successfully finds the highest point on the topographical map. Compare its performance to that of the genetic algorithm. Next, suppose that you have a forager who is searching for coffee beans in Colombia. Formulate a foraging landscape and develop a simulation that illustrates the success of foragers across that landscape.

Design Problem 18.6 (Foraging for Approximator Tuning)*: We can think of learning as a process of foraging for information. Can a foraging algorithm be used to tune an approximator structure? Is this a good idea? Why or why not? It is generally not possible to compute a gradient with respect to a structure change in an approximator (e.g., the change in the output with respect to the change in the number of neurons in a hidden layer). Can a foraging algorithm be used to tune the structure of an approximator? How? Develop an example and method, and test its performance relative to standard ones.

Design Problem 18.7 (Stable Social Foraging Swarms)*: Use the same model as in the chapter for a swarm of agents. Suppose that, however, $\|d_p^i\| \leq D_{p_1}\left\|E^i\right\| + D_{p_2}$, $\|d_v^i\| \leq D_{v_1}\left\|E^i\right\| + D_{v_2}$, and $\|d_f^i\| \leq D_f$ where D_{p_1}, D_{p_2}, D_{v_1} and D_{v_2} are known positive constants.

(a) Find conditions under which you are guaranteed to have the trajectories of the swarm uniformly ultimately bounded.

(b) Find bounds on the size of the trajectories of the error dynamics in terms of the parameters of the problem.

(c) Simulate the noise as the output of a chaotic system (e.g., via Duffing's equation), but one which still satisfies the above bounds. Study in simulation the effects of the parameters of the problem (e.g., k_a, k_r, and the noise bounds). Show simulations of the agent trajectories to illustrate the transient behavior of the swarm. To see how to do this for a *discrete-time* approximation to the continuous-time system, see Section 18.5.2.

Design Problem 18.8 (Modeling and Simulation of Honey Bee Clusters and In-Transit Swarms)*: Research the literature on how honey bees form clusters when a colony splits and then swarm together to a new nest site. Using the modeling ideas in this chapter, produce an ordinary differential equation model of both the clustering (aggregation process) and the in-transit swarm. Try to validate your model against the experimental results found in the literature.

Design Problem 18.9 (Predation and Aggregation)*: Read the paper entitled "Geometry for the Selfish Herd" by W.D. Hamilton in [240] that studies why an organism that is trying to protect itself will at times aggregate (group closely) with conspecifics (it also discusses other reasons for aggregation).

(a) Develop and simulate "jumping rules" (e.g., methods to generate Figure 1 in [240]). Characterize and analyze emergent aggregation behaviors by showing differences in aggregation behavior for different choices of rules. You could consider the case where all the "frogs" have the same rules, and the case where there are nonhomogeneous rules.

(b) Consider Hamilton's statement on p. 296 of [240]: "I know of no rule of jumping that can prevent them from aggregating." Why does Hamilton say this? Can you find a "rule" like the one he is talking about that can result in them not aggregating?

(c) Model and simulate a two-dimensional case, using some of the ideas in Section 3 of [240].

Design Problem 18.10 (Stable Dynamic Sphere Packing)*: There is a problem found in physics and biology of trying to pack objects on the surface of a sphere.

(a) Suppose that you seek to pack spherical swarm agents on a sphere. Suppose the agents are modeled as in the chapter via a double integrator but that a type of "hard-repel" term is used that makes them into spherical agents. Define the environment to be a sphere with the objective of each agent to be on the surface of the sphere by touching it at one point. Define appropriate attract-repel terms between agents so that only local interactions are allowed (i.e., so that every agent cannot be influenced by every other one, but only by its "neighbors"). Simulate the dynamic formation of packing on the sphere.

(b) Next, perhaps with the help of the literature in mathematics and physics, characterize the equilibria that represent the sphere being packed and show in simulation that these equilibria can be achieved. This is not a trivial problem. Are there equilibria that correspond to the agents moving about on the sphere?

(c) Provide conditions for Lyapunov stability or asymptotic stability of the equilibria specified in (b).

(d) Repeat (a)–(c) but for packing different shapes (e.g., an ellipsoid or a cube).

Design Problem 18.11 (Distributed Synchronization—From Fireflies to Simulations and Circuits)*: In this problem, you will investigate a biological phenomenon, simulate it, and then implement a circuit realization of it.

(a) Some types of fireflies exhibit a property, where they can synchronize their flashing. Investigate the literature on this and write a description of this phenomenon. Start with [496].

(b) Build a network of electronic fireflies according to the design in [194]. For this, build a line topology of at least four coupled oscillators.

(c) There are mathematical models of coupled nonlinear oscillators that can be used to represent the distributed sychronization. Study the one in [495], which has

$$\dot{\theta}_i = \omega_i + \frac{K}{N}\sum_{j=1}^{N}\sin(\theta_j - \theta_i)$$

for $j = 1, 2, \ldots, N$, where θ_i is the phase of oscillator i, ω_i is its natural frequency, and $K \geq 0$ is a coupling strength. Let $N = 10$, the $\omega_i = 1$, and $K = 1$. Develop a simulation of the N coupled oscillators and simulate for 20 sec. Plot the phase angles.

(d) Use the ideas in [495] to pick parameters and study "phase transitions" by illustrating these via a series of simulations.

(e) Explain the relationship between distributed synchronization and swarming (e.g., what is the "attraction/repulsion" function in the above distributed synchronization model?). Can you find conditions under which the distributed synchronization system is stable? What type of stability property does it possess?

(f) Model the system that you implemented in (b). Characterize and analyze the stability properties of the model. Compare to what is found in actual experiments.