Optimizing collective behavior of communicating active particles with machine learning

MACHINE LEARNING Science and Technology

OPEN ACCESS

RECEIVED
9 October 2023

REVISED
20 December 2023

ACCEPTED FOR PUBLICATION
8 January 2024

PUBLISHED
19 January 2024

PAPER

Optimizing collective behavior of communicating active particles with machine learning

Jens Grauer[1], **Fabian Jan Schwarzendahl**[1], **Hartmut Löwen**[1,*] **and Benno Liebchen**[2,*]

[1] Institut für Theoretische Physik II: Weiche Materie, Heinrich-Heine-Universität Düsseldorf, D-40225 Düsseldorf, Germany
[2] Institut für Physik der kondensierten Materie, Technische Universität Darmstadt, 64289 Darmstadt, Germany
[*] Authors to whom any correspondence should be addressed.

E-mail: hlowen@hhu.de and benno.liebchen@pkm.tu-darmstadt.de

Keywords: active particles, active matter, colloids, collective behavior, chemotaxis, quorum sensing, reinforcement learning

Supplementary material for this article is available online

Abstract

Bacteria and other self-propelling microorganisms produce and respond to signaling molecules to communicate with each other (quorum sensing) and to direct their collective behavior. Here, we explore agents (active particles) which communicate with each other to coordinate their collective dynamics for maximizing nutrient consumption. Using reinforcement learning and neural networks, we identify three different strategies: a 'clustering strategy', where the agents accumulate in regions of high nutrient concentration; a 'spreading strategy', where particles stay away from each other to avoid competing for sparse resources; and an 'adaptive strategy', where the agents adaptively decide to either follow or stay away from others. Our work exemplifies the idea that machine learning can be used to determine parameters that are evolutionarily optimized in biological systems but often occur as unknown parameters in mathematical models describing their dynamics.

1. Introduction

Bacteria and other self-propelling microorganisms have the ability to navigate in chemical concentration gradients, which is crucial for their survival. This ability, called chemotaxis, allows them to sense and respond to concentration gradients in their environment, which they use to find food and to avoid toxins [1, 2]. Interestingly, some microorganisms can produce the chemicals to which they respond themselves, which they use to communicate with each other (quorum sensing) and to coordinate their collective behavior. Examples are *E. coli* bacteria that use autoinducer II molecules for quorum sensing [3, 4] and *Dictyostelium* cells, which produce and respond to cAMP molecules [2, 5]. It is now known that besides biological microorganisms also self-propelling synthetic active particles communicate with each other via self-produced chemical fields. These particles catalyze a certain chemical reaction on part of their surface and respond to the resulting reagent/product concentration gradient by diffusiophoresis [6] or a similar mechanism. Here, the concentration gradients decay slowly in space and cause (long-ranged) chemical cross-interactions, providing a synthetic analog to microbial communication [7–11]. While this form of communication in synthetic active systems is essentially determined by particle design, recent progress in the development of feedback-controlled synthetic active particles [12–15] allows it to realize almost arbitrary communication rules. For example, there are now realizations of synthetic active particle systems [16], that have been equipped with artificial visual perception [14] and certain quorum sensing rules [13].

However, despite this progress in the realization of communicating active particles, to date, models and experiments have treated the communication rules simply as 'novel couplings' (interactions) that lead to a range of interesting new phenomena [10, 13, 14, 17–20], but that do not have a purpose. In contrast, in

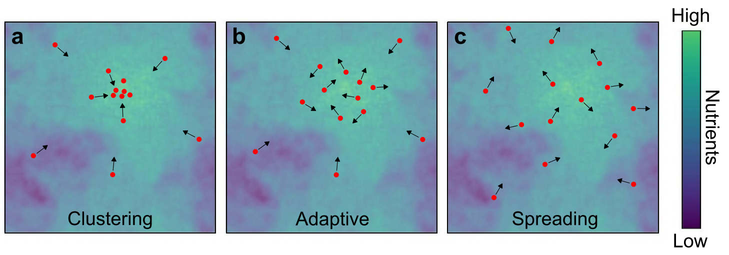

Figure 1. Illustration of the characteristic motion patterns that result from the learned communication strategies for gathering nutrients. Red dots show the agents and black arrows indicate their direction of motion. Panel (a) shows the clustering strategy, where all agents aggregate in regions of high nutrient concentration. Panel (b) shows the adaptive strategy where agents at low concentration move towards others (as for the clustering strategy) whereas agents at high concentration avoid others and tend to move away from the cluster (oppositely to the clustering strategy). Panel (c) shows the spreading strategy where the agents behave aversely and stay away from each other to avoid competing for food.

biological systems, communication rules can be evolutionarily optimized and help the agents approaching a common goal that typically benefits their survival chances or reproduction success.

In the present work, we aim to close this gap in the literature. To this end, we explore the idea to use machine learning to understand which communication rules are optimal for an ensemble of microorganisms or synthetic active particles to approach a common goal, such as maximizing their access to nutrients in the present case. To achieve this, we build on recent progress at the interface of active matter physics and artificial intelligence [21, 22] which is presently used to equip feedback controlled colloids, small robots and other active particles with the ability to learn from previous experience [23]. Recent examples comprise microswimmers that learn a complex flow field [24–29], gliders navigating in turbulent flow [30, 31], active particles swimming at low Reynolds number [32], learning chemotaxis [33, 34], flocking of a collection of agents [35] or navigating optimal paths in flow [36–39], motility fields [40], or force fields [41–43].

In the following, we formulate a minimal model describing communicating active particles and use reinforcement learning (RL) and neural networks (Deep Q-networks) to optimize their nutrient consumption. Notably, we find that the agents use their ability to communicate with each other to coordinate their collective behavior in a way that resembles characteristic food hunting strategies that can be observed for groups of animals or microorganisms in nature. In particular, we find that when the overall nutrient concentration is high (or when the consumption rate of the agents is low) the agents self-organize in a cluster and accumulate in regions of high local food concentration (figure 1(a)). Here, agents basically follow others that have already discovered regions of high food concentration. At lower nutrient concentration we observe a different strategy which we call 'spreading strategy'. Here, the agents stay away from each other to avoid competing for nutrients (figure 1(c)). In between these two strategies, the agents follow an adaptive strategy, where some agents (temporarily) aggregate while others stay away from them and hunt for food individually (figure 1(b)), resembling liquid–gas phase coexistence. These predictions can be tested in state of the art experiments with feedback controlled colloidal microswimmers [13] or with programmable robots [44]. In addition, interestingly, similar hunting strategies have been observed in various animals [45]. For instance, Pharaoh's ant, Monomorium pharaonis, lay pheromone trails to share information about the location of food sources and accumulate there [46], i.e. they essentially show the 'clustering strategy'. Similarly, when a honeybee has identified the location of a nectar or pollen, it often performs the famous 'waggle dance' to inform others about the location of the food source, which then also accumulate there [47]. Other works suggest that for certain birds and other animals at low nutrient concentration, the community is structured by competition and foraging agents tend to avoid each other [48, 49], similarly as for the spreading strategy.

Overall, the present work shows that machine learning can be used to optimize the communication rules and the resulting collective behavior of active systems, similarly as evolution optimizes the behavior of biological systems. (Note that in both cases the meaning of optimality is of course context dependent and the optimization process may lead to some 'local' optimum.) Our results can be used in the future for example to determine how synthetic active particles need to interact with each other and with their environment to efficiently collect certain targets, as relevant e.g. for decontaminating polluted water [50] or to collect passive colloidal particles, such as microplastics [51]. In addition, our results might also be useful in the formulation of mathematical models for the collective dynamics of biological systems. While such models normally involve free parameters that are optimized in the underlying biological system, the present work shows how machine learning can be used to determine such parameters to reduce the number of free parameters.

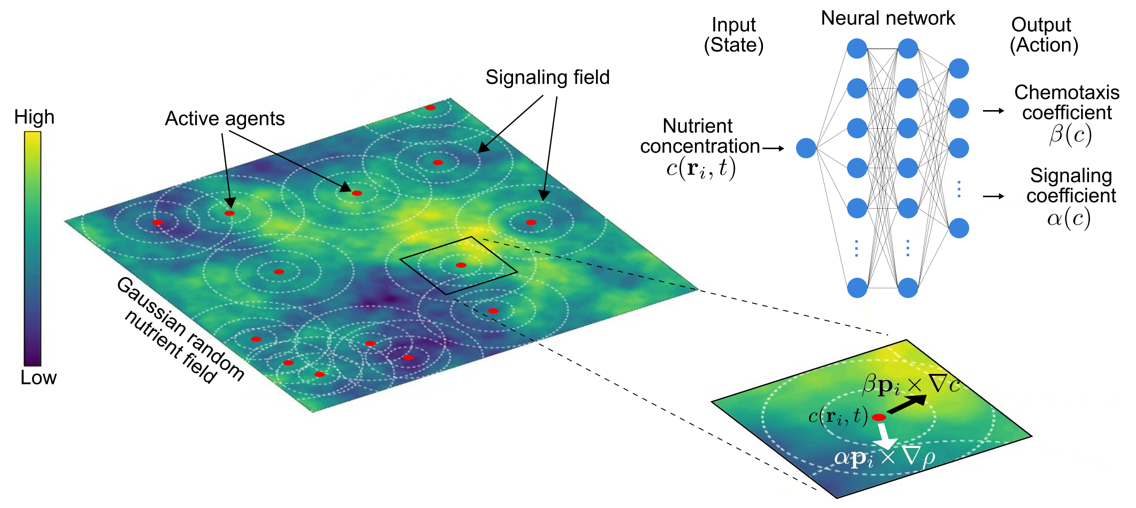
High
Low
Active agents
Signaling field
Gaussian random nutrient field
Input (State)
Neural network
Output (Action)
Nutrient concentration $c(\mathbf{r}_i,t)$
Chemotaxis coefficient $\beta(c)$
Signaling coefficient $\alpha(c)$
$\beta\mathbf{p}_i\times\nabla c$
$c(\mathbf{r}_i,t)$
$\alpha\mathbf{p}_i\times\nabla\rho$

Figure 2. Schematic of the model and the learning approach. The active particles (red dots) move in a concentration field (background color indicates nutrient concentration) and communicate with each other by producing quorum sensing molecules (white circles illustrate iso-concentration lines of the signaling molecules due to each agent). We feed the nutrient concentration at the position of each agent into a neural network and use RL to predict the functional form of the sensitivity functions $\alpha(c), \beta(c)$. These functions determine the response of the agents to the nutrient concentration (determined by $\alpha(c)$) and the signaling molecule concentration ($\beta(c)$).

2. Methods

2.1. Model

We consider an ensemble of N overdamped particles representing artificial or biological agents located in a Gaussian random nutrient field $c(\mathbf{r},t)$ with values $\in [0, c_{\max}]$. The particles have position and orientation vectors $\mathbf{r}_i$, $\mathbf{p}_i = (\cos\theta_i, \sin\theta_i)$, where $i = 1, 2, \ldots, N$. They move with a constant self-propulsion speed v_0 in a two-dimensional quadratic simulation box of size L and with periodic boundary conditions. All agents are randomly placed in the simulation box and are subject to a nutrient concentration field $c(\mathbf{r},t)$ that we initialize as a Gaussian random field, created via the power spectrum realization method [52–54] with a power spectrum of $\langle\tilde{c}(\mathbf{k})\tilde{c}(-\mathbf{k})\rangle \propto k^{-3}$ (where $k = |\mathbf{k}|$) and a grid discretization that is small compared to all other relevant length scales in the system (see background of figure 2, left panel, for an exemplary realization). Here, $\tilde{c}(\mathbf{k})$ is the Fourier transform of the reduced concentration field $c(\mathbf{r}, t=0)/c_{\max}$. We choose the numerical proportionality coefficient in the above expression for the power spectrum such that $c(\mathbf{r}, t=0) \in [-c_{\max}/2, c_{\max}/2]$ and finally shift $c(\mathbf{r}, t=0) \to c(\mathbf{r}, t=0) + c_{\max}/2 \in [0, c_{\max}]$.

For simplicity, we assume that the particles consume nutrients only at their present location (the nearest grid point) and that the nutrients are somewhat sparse so that the agents consume them with a rate $k_d c(\mathbf{r}_i, t)$ that is proportional to the local nutrient concentration. The nutrient concentration field evolves diffusively, with diffusion coefficient D_c, and an additional sink term describing the nutrient consumption due to the agents at their instantaneous positions. There are many different models for the form of the (nutrient) consumption term in the literature; here we assume that nutrients are sparse, and the consumption is proportional to the local concentration (see e.g. [11, 55–57] and references therein for similar models):

$$\dot{c}(\mathbf{r},t) = D_c \Delta c(\mathbf{r},t) - k_d \sum_{i=1}^{N} c(\mathbf{r}_i,t)\,\delta(\mathbf{r}-\mathbf{r}_i)\ . \tag{1}$$

Here we are mainly interested in the situation where the nutrient field changes only due to consumption. Still, we explicitly model the dynamics of c, with a very small D_c, to smoothen the gradients created by the consuming agents, i.e. for technical reasons.

We now equip the agents with the ability to communicate with each other and task them to use this ability to autonomously coordinate their self-propulsion directions such that all agents together deplete the nutrient field as efficiently as possible. To choose their self-propulsion direction, the agents try to find the best possible compromise between greedily swimming up the local concentration gradient (chemotaxis) and a direction which is suggested by communication with the other agents (quorum-sensing). To model the latter influence, we assume that each agent produces the chemicals which are relevant for communication with a rate λ and that all other agents respond to the resulting concentration field $\rho(\mathbf{r},t)$. Like the nutrient field, the concentration field of the signaling molecules evolves diffusively, with diffusion coefficient D_ρ [11, 55–57].

We model the production due to each agent as a point source with a constant production rate λ and assume that the agents consume the molecules (degradation) with a rate μ (see [11, 55–57] for similar models):

$$\dot{\rho}(\mathbf{r},t) = D_\rho \Delta\rho(\mathbf{r},t) + \lambda \sum_{i=1}^{N} \delta(\mathbf{r}-\mathbf{r}_i) - \mu\rho(\mathbf{r},t) \ . \tag{2}$$

We assume that the diffusion of the signaling molecules is fast; i.e. we solve equation (2) in steady state ($\dot{\rho}=0$) yielding in two-dimensions:

$$\rho = \frac{\lambda}{D_\rho} K_0(\kappa r) \,. \tag{3}$$

Here, $K_0(x)$ is the modified Bessel function of second kind and $\kappa = \sqrt{\frac{\mu}{D_\rho}}$. We deliberately use the solution in two dimensions to get the same results as when solving equation (2) explicitly in the steady state. Note that solving the simulation model including equations (1) and (2) in three dimensions would not qualitatively change our results [18].

The following equations describe the dynamics of the particles depending on the nutrient concentration field $c(\mathbf{r},t)$ as well as the field of quorum sensing molecules $\rho(\mathbf{r},t)$

$$\dot{\mathbf{r}}_i(t) = v_0 \mathbf{p}_i \tag{4}$$

$$\dot{\theta}_i(t) = \alpha \mathbf{p}_i \times \nabla\rho + \beta \mathbf{p}_i \times \nabla c, \tag{5}$$

where $\mathbf{a}\times\mathbf{b} = a_1 b_2 - a_2 b_1$ is the two-dimensional cross product and α, β represent the alignment rate of the self-propulsion direction of the agents with the local quorum sensing and nutrient fields, respectively. The particles move up (down) the local concentration gradient of signaling molecules when $\alpha > 0$ ($\alpha < 0$) and always move up the food concentration gradient ($\beta > 0$).

As further discussed below, we consider $\alpha(c)$, $\beta(c)$ as parameter functions that the agents can optimize. Accordingly, in equation (5), the values of α,β depend on the current position of the agents, i.e. $\alpha = \alpha(c(\mathbf{r}_i(t)))$, and $\beta = \beta(c(\mathbf{r}_i(t)))$.

Considering α,β as parameter functions that will be fixed by RL later, our model has 8 parameters ($c_{\max}, D_c, k_d, D_\rho, \lambda, \mu, v_0$ and the agent density N/L^2). Choosing time and lengths units as $t_0 = \frac{1}{\lambda}$ and $l_0 = \frac{1}{\kappa}$, we are formally left with 6 dimensionless parameters [58]. In the following we consider the agent density $Nl_0^2/L^2 = N/(\kappa^2 L^2)$ and the consumption rate $k_d t_0/l_0^2 = k_d\kappa^2/\lambda$ as our key control parameters which we vary in the following. The following other dimensionless parameters are kept fixed in our simulations: $\frac{D_\rho}{l_0{}^2/t_0} = \frac{\mu}{\lambda} = 10^{-3}$, $\frac{v_0}{l_0/t_0} = \frac{v_0\kappa}{\lambda} = 10^{-3}$, $c_{\max} l_0^2 = \frac{c_{\max}}{\kappa^2} = 10^4$ and finally $\frac{D_c}{l_0{}^2/t_0} = \frac{D_c\kappa^2}{\lambda} = 10^{-8}$ has been chosen smaller than all other parameters to make equation (1) quasi-stationary.

2.2. Learning approach

We use RL [59] combined with a neural network to optimize the functions $\alpha(c)$ and $\beta(c)$. To achieve this, in short, we iteratively propagate the particles and update the neural network which essentially represents a mapping from a c-value to a probability distribution for $\alpha(c), \beta(c)$ (where high probabilities represent α,β values that are likely to lead to a high nutrient consumption in the future). While moving through space, the agents dynamically feed the c-values at their momentaneous position into the neural network and select α,β values from the probability distribution, according to some policy. (In the simplest case, each agent chooses the α,β values with the highest probability.) As opposed to methods from classical optimal control theory [60] and tabular RL methods, the key advantage of the deep RL method which we use, is that it can handle an enhanced complexity, as required here to optimize the dynamics of many interacting agents.

We now describe our procedure in detail. For each active particle (agent), we initially define a state s_0 that is given by the value of the nutrient concentration field at the agent's position, i.e. for agent i we have $s_0 = c(\mathbf{r}_i(0),0) \in [0, c_{\max}]$. Then we simulate the dynamics of all agents and of their environment (equations (1)–(5)) for a time interval $\Delta t = 100t_0$, after which we update the state of all agents, i.e. for agent i we have $s_1 = c(\mathbf{r}_i(\Delta t), \Delta t)$, and then after iterating equations (1)–(5) for a time of Δt again, we update the state to $s_2 = c(\mathbf{r}_i(2\Delta t), 2\Delta t)$ and so on.

At each step n (corresponding to a time interval Δt), for each agent, we choose an individual action a_n which corresponds to choosing a combination of $\alpha\kappa^3/\lambda \in \{-5\times10^{-6}, -4\times10^{-6}, \ldots, 5\times10^{-6}\}$ and $\beta\kappa^3/\lambda \in [0, 0.5\times10^{-6}, \ldots, 5\times10^{-6}]$, leading to $11\times11 = 121$ possible action choices for each agent. The parameter intervals have been chosen such that the learned (optimal) values are just not at the edge of the intervals (see figure 5), to achieve the best possible numerical efficiency without affecting the resulting optimal values.

To motivate the action choices, at each step, we assign a reward to each agent. When agent i transitions from state s_n to state s_{n+1}, it receives an immediate reward of $r_n = \frac{1}{\Delta t c_{\max}} \int_{n\Delta t}^{(n+1)\Delta t} c(\mathbf{r}_i, t')\mathrm{d}t' - 1$. This reward has been designed to be between zero, when the agent is at a location with the highest possible concentration ($c(\mathbf{r}, t) = c_{\max}$), and negative at lower concentrations, with a minimum of -1, at locations where the local concentration vanishes. The individual action choice (α, β) for each agent is then motivated by the attempt to maximize the expected future reward, which we calculate for each agent individually and which is defined as $R_n = \sum_{n'=n}^{n_{\max}} \gamma^{n'-n} r_{n'} = r_n + \gamma R_{n+1}$. Here, the sum runs over all $n_{\max}$ steps in an episode and the discount factor $\gamma = 0.9$ weights the importance of future rewards compared to immediate ones. To estimate the value R_n and hence to allow the agents choosing an action, we determine the so-called Q-function $Q(s, a)$ that is shared by all agents and that indicates the expected sum of discounted future rewards when taking action $a_n = a$ in state $s_n = s$ (and following policy π, discussed below). Before discussing how we determine the Q-function in practice, we first note that it can be written as [59]

$$Q(s,a) = \sum_{s',r} P(s',r|s,a) \sum_{a'} \pi(a'|s') \left[r + \gamma Q(s',a')\right] . \tag{6}$$

Here, $P(s', r|s, a)$ denotes the probability that an agent which is in state s and chooses action a reaches state s' and receives a 'path-dependent' reward r. The policy $\pi(a|s)$ in turn represents the probability of taking action a in state s. In Q-learning, after convergence, the optimal policy consists in always taking the action a with the largest Q-value $\max_a Q(s, a)$. However, during training, to encourage exploration, the agents follow a ϵ-greedy policy. This means that a random action is chosen with probability ϵ, while the action with the highest Q-value is chosen with probability $1 - \epsilon$.

The entire process of iteratively propagating the agents (equations (1)–(5)) and choosing actions (α, β) at times $\Delta t, 2\Delta t\ldots$, until $t_{max} = 2000t_0 = 20\Delta t$ makes up one episode. Overall we run 5000 episodes for each of which we create a new random nutrient field and place the agents randomly in the simulation box. Within the first 4000 episodes, ϵ is exponentially decreased from 1 to 0.01, while in the last 1000 episodes it is kept constant at $\epsilon = 0.01$.

Let us now discuss how we determine the Q-function. Since a direct iteration of the Q-function, as in tabular Q-learning, leads to a non-managable numerical cost (Q would be a matrix of dimensions: number of states times number of actions), we determine (approximate) the Q-values by an artificial neural network $Q(s, a; \theta)$ (called deep Q-network (DQN)) [61]. Here θ describes the set of parameters of the neural network (weights and biases) that are adjusted during training to minimize the difference between the DQN's output and the right-hand side of equation (6). This is achieved by storing the information of all transitions $\{s_n, a_n, s_{n+1}, r_n\}$ during an episode (replay buffer) and using this information after an episode is finished to update the parameters of the neural network. We use the mean squared error as a loss function, i.e. the mean value of the difference $(r_n + \gamma \max_a Q(s_{n+1}, a; \theta) - Q(s_n, a_n; \theta))^2$ is calculated over all transitions of an episode. An Adam optimizer [62] with a learning rate of 0.01 and a learning rate decay of 0.05 is applied to update the network parameters. The DQN is a fully connected network that has a single neuron as an input layer. The network has two hidden layers, each with 256 neurons, using a ReLU activation function [63]. The output layer consists of one output neuron for every action (i.e. we have 121 output neurons) with a linear activation function. (Hyperparameters have been optimized based on the random search procedure.) Overall, at each step, each agent feeds the concentration value at its current position into the network which then provides a weight (probability) for each possible combination of α, β values. We trained the neural network with the help of TensorFlow (version 2.6.0) [64]. We have performed our simulations on 23 nodes of our local computer cluster where each node has two 4-core 2.4 GHz Intel Xeon CPUs, and 24 GB of RAM. Generating a single RL model takes about 3–10 h (depending on particle density). To improve the accuracy of our predictions each data point in figure 4 is based on averages over at least 500 learned models.

Let us now exemplarily explore how the nutrient consumption rate of the agents evolves during training. To this end, we define the relative performance as $C/C_{\text{rand}} - 1$, where C is the total amount of nutrients that have been consumed by all agents in the system in a given episode and C_{rand} is the total amount of nutrients that are consumed by the same number of randomly moving agents in the same time span. Figure 3 shows that the relative performance of the agents before training (blue line at episode 0) corresponds to that of an agent taking random actions, i.e. random values of α, β, leading to a random-walk-like dynamics (orange line). The performance then systematically increases during training and, irrespective of fluctuations, it finally converges to a constant value (yellow line). Importantly, this value is significantly larger than that of a system of greedy agents (grey line), that use a constant $\beta\kappa^3/\lambda = 10^{-5}$ to always move up the local concentration gradient but that do not communicate with each other ($\alpha = 0$). This clearly shows that communication allows the agents to enhance their performance.

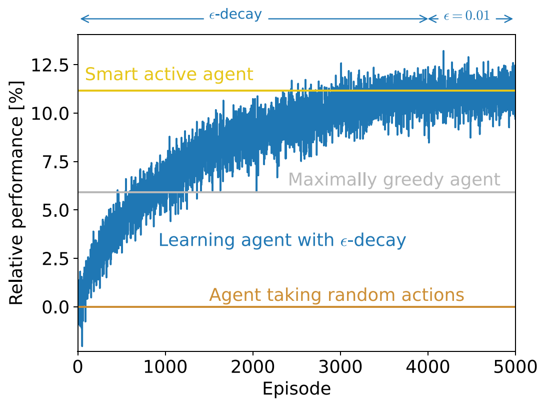

Figure 3. The relative performance of agents trained with the deep Q-networks algorithm (blue line) is shown as a function of the number of training episodes in comparison with an agent performing random actions (orange line) and a maximally greedy agent (grey line). Here, the relative performance is defined as the relative improvement compared to the agent taking random actions in per cent. The yellow and the gray line have been obtained numerically. The yellow line shows the late-time average of the blue line. The gray curve is an average over numerical simulations with a constant $\beta\kappa^3/\lambda = 10^{-5}$ such that the agent always moves up the local concentration gradient but does not communicate with others ($\alpha = 0$). Parameters: $k_d\kappa^2/\lambda = 10^{-7}$, $Nl_0^2/L^2 = 60$).

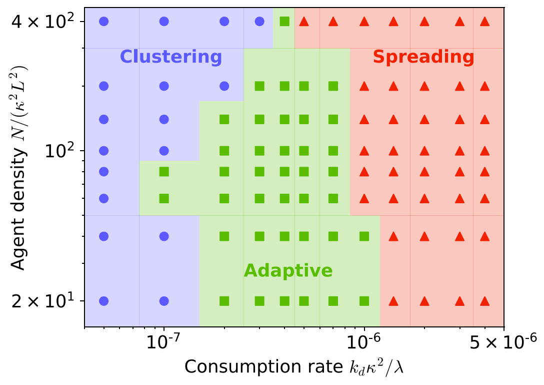

Figure 4. State diagram. The agents employ different strategies depending on the agent density $N/(\kappa^2L^2)$ and the dimensionless consumption rate $k_d\kappa^2/\lambda$. Clustering strategy: blue dots show regions where $\alpha(c) \geqslant 0$ for all concentrations such that the agents cluster, typically in regions of high nutrient concentration. Spreading strategy: red dots show regions where $\alpha < 0$ for all nutrient concentrations, such that the agents stay away from each other and try to individually reach regions of high nutrient concentration. Adaptive strategy: green dots represent parameter combinations where $\alpha(c) \geqslant 0$ for small concentrations and where $\alpha(c) < 0$ for large concentrations, such that the agents cluster at low concentration but avoid each other at higher concentrations.

2.3. Results

We now apply the RL-approach to learn functions $\alpha(c)$, $\beta(c)$ that optimize the nutrient consumption of the agents depending on the reduced consumption rate $k_d\kappa^2/\lambda$ and the agent density $N/(\kappa^2L^2)$. We represent the resulting spatiotemporal dynamics of the agents in a state diagram, figure 4, which contains three qualitatively different functional forms of $\alpha(c)$ (see caption of figure 4 for a definition of the states).

For low consumption rates, we find a 'clustering strategy' (blue dots in figure 4), which represents states where $\alpha(c) > 0$ for all concentrations (figure 5), i.e. where the agents always align their orientation towards higher particle concentration, leading to clustering of the agents. What is the advantage of clustering? It allows an agent that is located at very low nutrient concentration to identify (remote) regions of higher concentration by following other agents. This is often more efficient than only following the nutrient concentration gradient which would bear the risk to end up in a tiny local maximum. Why does the clustering strategy emerge at low food consumption rates only? This is because at high consumption rates, clustered agents would rapidly deplete the available nutrients. In addition to this, the blue curves figure 5 provide additional information: they show that at low concentration the agents react stronger to other agents than at large concentration (top panel), whereas their response to the local nutrient concentration behaves oppositely. This behavior of $\alpha(c), \beta(c)$ is plausible since an agent that is located at very low nutrient concentration will statistically benefit more from following others to identify regions of higher concentration than an agent that is already in a favorable region. Conversely, at high concentration agents primarily focus on staying at high nutrient concentrations and not to care too much about the others. It is interesting that

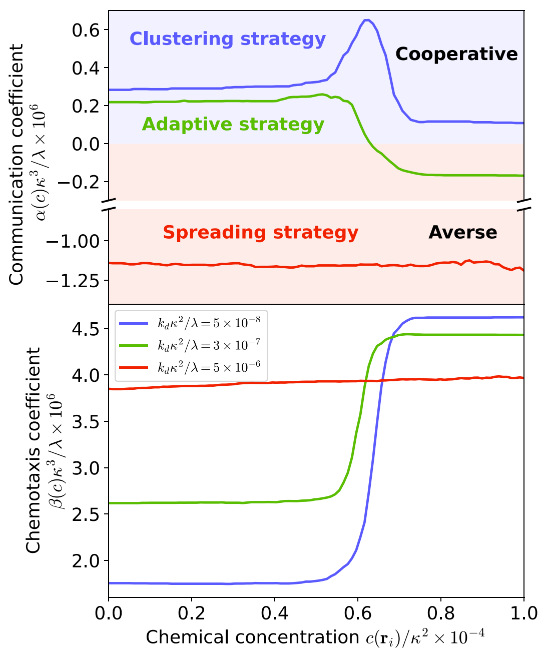

Figure 5. Learned coefficients determining the sensitivity of the agents to the quorum sensing concentrations (top) and the nutrient concentration (bottom). The shown curves are based on data at 100 different concentrations. Agent density: $N/(\kappa^2 L^2) = 60$.

$\alpha(c)$ shows an additional peak around concentrations of $c/\kappa^2 \sim 6.5 * 10^3$, i.e. in the regime where $\beta(c)$ sharply increases.

For large consumption rates (and high particle density), the agents follow the 'spreading strategy' (red dots in figure 4), which is defined by $\alpha(c) < 0$ at all concentrations (figure 5). That is, following the spreading stratey, particles stay away from each other to avoid competing for nutrients. Thus they individually hunt for food by moving up the local food concentration ($\beta(c) > 0$ is almost constant).

The third regime in the state diagram is the 'adaptive strategy', where $\alpha(c) > 0$ at low *c* but $\alpha(c) < 0$ at large *c*. That is, the agents adaptively decide, depending on the local concentration, if they move towards other agents or stay away from them. This is plausible since at low concentration the agent depend on others to identify (remote) regions of high nutrient concentration. (Simply following the nutrient concentration gradient would bear the risk of ending up in a tiny maximum.) When an agent is already at a large concentration, it does no longer depend on others and its primary concern is to avoid that the concentration does not get depleted too rapidly; thus we have $\alpha(c) < 0$ at large *c*.

3. Conclusions

In conclusion, this work exemplifies the idea that RL can be used to optimize the interactions—or communication rules—of self-propelling agents to achieve a common goal. For the present example, the learned communication rules allow the agents to enhance their nutrient consumption and lead to three distinct motion patterns that closely resemble food hunting strategies observed in nature. On a more conceptual level, this work also illustrates the idea to use machine learning to determine parameters (or parameter functions) that are fixed in biological systems (evolutionarily optimized) but often occur as non-optimized unknown parameters in mathematical models describing their dynamics. That way, machine learning could be used in the future to reduce the number of free parameters in mathematical models of biological systems. For smart active particles such as future programmable microrobots, our results could also be used to determine how these agents need to interact with each other and with their environment to efficiently collect certain targets, e.g. when decontaminating polluted water [50] or collecting microplastics [51].

We finally note that the present work just serves as a starting point that aims to illustrate the discussed ideas. It employs a minimal model and a rather simple machine learning approach, that invite the development of more sophisticated approaches in the future that could account e.g. for a visual perception of the agents and could involve planning. Other future generalizations could use more involved ways to control the communication among the agents or could extend the information that particles receive about each other; providing them e.g. with knowledge about the orientations of other particles, could allow them e.g. to flock [35] and it would be interesting to find out when this is beneficial to further enhance the nutrient consumption efficiency. Finally, it would also be interesting to explore non-identical agents with different speeds or consumption rates in the future, which could allow it to make interesting links to inter-species cooperation and competition in foraging theory.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

ORCID iDs

Hartmut Löwen https://orcid.org/0000-0001-5376-8062
Benno Liebchen https://orcid.org/0000-0002-7647-6430

References

[1] Berg H C 2004 *E. Coli in Motion* (Springer)
[2] Eisenbach M, Tamada A, Omann G, Segall J, Firtel R, Meili R, Gutnick D, Varon M, Lengeler J W and Murakami F 2004 *Chemotaxis* (World Scientific Publishing Company)
[3] Miller M B and Bassler B L 2001 *Annu. Rev. Microbiol.* **55** 165
[4] Laganenka L, Colin R and Sourjik V 2016 *Nat. Commun.* **7** 12984
[5] Gerisch G 1968 *Cell Aggregation and Differentiation in Dictyostelium* (*Current Topics in Developmental Biology* vol 3) (Elsevier) pp 157–97
[6] Anderson J L 1989 *Annu. Rev. Fluid Mech.* **21** 61–99
[7] Hong Y, Blackman N M K, Kopp N D, Sen A and Velegol D 2007 *Phys. Rev. Lett.* **99** 178103
[8] Saha S, Golestanian R and Ramaswamy S 2014 *Phys. Rev.* E **89** 062316
[9] Stark H 2018 *Acc. Chem. Res.* **51** 2681–8
[10] Liebchen B and Löwen H 2018 *Acc. Chem. Res.* **51** 2982–90
[11] Liebchen B and Mukhopadhyay A K 2021 *J. Phys.: Condens. Matter* **34** 083002
[12] Khadka U, Holubec V, Yang H and Cichos F 2018 *Nat. Commun.* **9** 3864
[13] Bäuerle T, Fischer A, Speck T and Bechinger C 2018 *Nat. Commun.* **9** 3232
[14] Lavergne F A, Wendehenne H, Bäuerle T and Bechinger C 2019 *Science* **364** 70
[15] Sprenger A R, Fernandez-Rodriguez M A, Alvarez L, Isa L, Wittkowski R and Löwen H 2020 *Langmuir* **36** 7066–73
[16] Bechinger C, Di Leonardo R, Löwen H, Reichhardt C, Volpe G and Volpe G 2016 *Rev. Mod. Phys.* **88** 045006
[17] Barberis L and Peruani F 2016 *Phys. Rev. Lett.* **117** 248001
[18] Grauer J, Löwen H, Be'er A and Liebchen B 2020 *Sci. Rep.* **10** 5594
[19] Zampetaki A V, Liebchen B, Ivlev A V and Löwen H 2021 *Proc. Natl Acad. Sci.* **118** e2111142118
[20] Ziepke A, Maryshev I, Aranson I S and Frey E 2022 *Nat. Commun.* **13** 6727
[21] Cichos F, Gustavsson K, Mehlig B and Volpe G 2020 *Nat. Mach. Intell.* **2** 94
[22] Nasiri M, Löwen H and Liebchen B 2023 *Europhys. Lett.* **142** 17001
[23] Muinos-Landin S, Fischer A, Holubec V and Cichos F 2021 *Sci. Robot.* **6** eabd9285
[24] Colabrese S, Gustavsson K, Celani A and Biferale L 2017 *Phys. Rev. Lett.* **118** 158004
[25] Colabrese S, Gustavsson K, Celani A and Biferale L 2018 *Phys. Rev. Fluids* **3** 084301
[26] Gustavsson K, Biferale L, Celani A and Colabrese S 2017 *Eur. Phys. J.* E **40** 110
[27] Alageshan J K, Verma A K, Bec J and Pandit R 2020 *Phys. Rev.* E **101** 043110
[28] Qiu J, Huang W, Xu C and Zhao L 2020 *Sci. China Phys. Mech. Astron.* **63** 284711
[29] Biferale L, Bonaccorso F, Buzzicotti M, Clark Di Leoni P and Gustavsson K 2019 *Chaos* **29** 103138
[30] Reddy G, Celani A, Sejnowski T J and Vergassola M 2016 *Proc. Natl Acad. Sci.* **113** E4877–84
[31] Reddy G, Wong-Ng J, Celani A, Sejnowski T J and Vergassola M 2018 *Nature* **562** 236
[32] Tsang A C H, Tong P W, Nallan S and Pak O S 2020 *Phys. Rev. Fluids* **5** 074101
[33] Hartl B, Hübl M, Kahl G and Zöttl A 2021 *Proc. Natl Acad. Sci.* **118** e2019683118
[34] Dou Y and Bishop K J 2019 *Phys. Rev. Res.* **1** 032030
[35] Durve M, Peruani F and Celani A 2020 *Phys. Rev.* E **102** 012601
[36] Liebchen B and Löwen H 2019 *Europhys. Lett.* **127** 34003
[37] Daddi-Moussa-Ider A, Löwen H and Liebchen B 2021 *Commun. Phys.* **4** 1
[38] Zanovello L, Caraglio M, Franosch T and Faccioli P 2021 *Phys. Rev. Lett.* **126** 018001
[39] Nasiri M and Liebchen B 2022 *New J. Phys.* **24** 073042
[40] Monderkamp P A, Schwarzendahl F J, Klatt M A and Löwen H 2022 *Mach. Learn.: Sci. Technol.* **3** 045024
[41] Schneider E and Stark H 2019 *Europhys. Lett.* **127** 64003
[42] Yang Y and Bevan M A 2018 *ACS Nano* **12** 10712
[43] La H M, Lim R and Sheng W 2014 *IEEE Trans. Control Syst. Technol.* **23** 52
[44] Mijalkov M, McDaniel A, Wehr J and Volpe G 2016 *Phys. Rev.* X **6** 011008
[45] Sumpter D J 2010 *Collective Animal Behavior* (Princeton University Press)
[46] Sumpter D J and Beekman M 2003 *Animal Behav.* **66** 273–80
[47] Seeley T D 2009 *The Wisdom of the Hive: The Social Physiology of Honey Bee Colonies* (Harvard University Press)
[48] Werner E E and Hall D J 1979 *Ecology* **60** 256
[49] Mitchell W A, Abramsky Z, Kotler B P, Pinshow B and Brown J S 1990 *Ecology* **71** 844
[50] Gao W and Wang J 2014 *ACS Nano* **8** 3170
[51] Wang L, Kaeppler A, Fischer D and Simmchen J 2019 *ACS Appl. Mater. Interfaces* **11** 32937
[52] Pen U-L 1997 *Astrophys. J.* **490** L127

[53] Bertschinger E 2001 *Astrophys. J. Suppl. Ser.* **137** 1
[54] Goon G 2021 Gaussian fields (available at: https://garrettgoon.com/gaussian-fields/)
[55] Murray J D and Murray J D 2003 *Mathematical Biology: II: Spatial Models and Biomedical Applications* vol 3 (Springer)
[56] Ben-Jacob E, Cohen I and Levine H 2000 *Adv. Phys.* **49** 395
[57] Hillen T and Painter K J 2009 *J. Math. Biol.* **58** 183–217
[58] Alternatively, when starting with equation (3) instead of with equation (2) , we have one parameter less (λ/D_ρ, μ/D_ρ instead of D_ρ, λ, μ). Then choosing e.g. $l_0 = 1/\kappa$ and $t_u = 1/(\kappa v_0)$ results in 5 dimensionless parameters
[59] Sutton R S and Barto A G 2018 *Reinforcement Learning: An Introduction* (MIT Press)
[60] Kirk D E 2004 *Optimal Control Theory: An Introduction* (Courier Corporation)
[61] Mnih V *et al* 2015 *Nature* **518** 529
[62] Kingma D P and Ba J 2014 arXiv:1412.6980
[63] Nair V and Hinton G E 2010 Rectified linear units improve restricted Boltzmann machines *Proc. 27th Int. Conf. on Int. Conf. on Machine Learning* (Omnipress)
[64] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado G S, Davis A and Dean J 2015 TensorFlow: large-scale machine learning on heterogeneous systems software available from tensorflow.org (available at: www.tensorflow.org/)