PNAS
www.pnas.org

Supplementary Information for

Microswimmers learning chemotaxis with genetic algorithms

Benedikt Hartl, Maximilian Hübl, Gerhard Kahl, and Andreas Zöttl

Corresponding Author: Andreas Zöttl
E-mail: andreas.zoettl@tuwien.ac.at

This PDF file includes:

Supplementary text
Figs. S1 to S22
Legends for Movies S1 to S8
SI References

Other supplementary materials for this manuscript include the following:

Movies S1 to S8

Supporting Information Text

A. Microswimmer equations of motion. We model the motion of a microswimmer in one dimension by the well-known Najafi-Golestanian swimmer (see also main text) (1). It consists of three beads of radius R at positions $x_i(t)$, $i = 1, 2, 3$, separated by two arms of length $L_1(t)$ and $L_2(t)$ which vary over time t. The microswimmer agent proposes forces $F_1(t)$ and $F_3(t)$ which act on the swimmer beads 1 and 3 (see left panels in main text Fig. 1). Since the microswimmer moves force-free, the force on bead 2 is given by $F_2(t) = -F_1(t) - F_3(t)$. When the forces $F_i(t)$, $i = 1, 2, 3$, are known, the velocities of the beads $v_i(t)$ can be computed, and are linearly related to the forces via the mobility tensor $\boldsymbol{\mathcal{M}}$: $v_i(t) = \mathcal{M}_{ij}(t)F_j(t)$. The components of the mobility tensor, $\mathcal{M}_{ij}(t)$, consist of self-mobilities $\mathcal{M}_{ii}(t)$ and of cross-mobilities $\mathcal{M}_{ij}(t)$, $i \neq j$ induced by the hydrodynamic interactions (HI) between the beads. In the limit of far-field HI described by the Oseen approximation this leads to the following linear relation between the velocities and forces (2):

$$v_1(t) = \frac{F_1(t)}{6\pi\eta R} + \frac{F_2(t)}{4\pi\eta L_1(t)} + \frac{F_3(t)}{4\pi\eta(L_1(t)+L_2(t))}\,, \qquad [1]$$

$$v_2(t) = \frac{F_1(t)}{4\pi\eta L_1(t)} + \frac{F_2(t)}{6\pi\eta R} + \frac{F_3(t)}{4\pi\eta L_2(t)}\,, \qquad [2]$$

$$v_3(t) = \frac{F_1(t)}{4\pi\eta(L_1(t)+L_2(t))} + \frac{F_2(t)}{4\pi\eta L_2(t)} + \frac{F_3(t)}{6\pi\eta R}\,. \qquad [3]$$

In order to prevent the microswimmer agent from contracting or stretching the arms to an infinite amount, we add piecewise harmonic restoring forces $F_R(L_1)$ and $-F_R(L_1)$ to beads 1 and 2, respectively, and $-F_R(L_2)$ and $F_R(L_2)$ to beads 3 and 2, respectively. We define $F_R(L_i)$ for $i = 1, 2$ as

$$F_R(L_i) = \begin{cases} (L_i - L_{\min}) \times K & \text{if } L_i < L_{\min} \\ (L_i - L_{\max}) \times K & \text{if } L_i > L_{\max} \\ 0 & \text{else}\,, \end{cases} \qquad [4]$$

where K represents the spring constant of the harmonic restoring forces and $L_{\min}$ and $L_{\max}$ mark the minimum and maximum extent of the arm lengths L_i where the $F_R(L_i)$ set in. The total forces acting on the outer beads are then $F_1 \to F_1 + F_R(L_1)$ and $F_3 \to F_3 - F_R(L_2)$; by employing the force-free condition, the change in F_2 can be written as $F_2 \to F_2 - F_R(L_1) + F_R(L_2)$. We use the parameters $K = F_0/R$, $L_{\min} = 0.7L_0$ and $L_{\max} = 1.3L_0$.

Basics of Neural Networks. An artificial neural network (ANN) is a set of units (neurons) which are interconnected with each other such that an input vector, $\mathbf{x}^{(\mathrm{I})} = (x_1, \ldots, x_{N_\mathrm{I}})$ is processed, in a (non-linear) tensor operation to an output vector, $\mathbf{y}^{(\mathrm{O})} = (y_1, \ldots, y_{N_\mathrm{O}})$, which can be seen as a function of the input of the realization, $\mathcal{N}$, of the ANN, i.e. $\mathbf{y}^{(\mathrm{O})} = \mathcal{N}(\mathbf{x}^{(\mathrm{I})})$.

Usually in ANNs, neurons are organized in layers, $\nu^{(l)} = (\nu_1^{(l)}, \ldots, \nu_{N_l}^{(l)})$, and the connection from neuron i in layer l to neuron j in layer k is realized via elements, $w_{ji}^{(kl)}$, in so-called weight matrices, $\mathcal{W}^{(kl)} \in \mathbb{R}^{N_k \times N_l}$: the output vector of all N_l neurons, $\mathbf{y}^{(l)} = (y_i^{(l)}, \ldots, y_{N_l}^{(l)})$ of layer l is related to the input, $x_j^{(k)}$ of neuron j in layer k (usually $k = l + 1$) by the operation $x_j^{(k)} = \sum_{i=1}^{N_l} w_{ji}^{(kl)} y_i^{(l)} + b_j^{(k)}$, or in matrix vector notation $\mathbf{x}^{(k)} = \mathcal{W}^{(kl)}\mathbf{y}^{(l)} + \mathbf{b}^{(k)}$, where $\mathbf{b}^{(k)} = (b_1^{(k)}, \ldots, b_{N_k}^{(k)}) \in \mathbb{R}^{N_k}$ is the so-called bias vector of neuron k. The layer-wise input is then subject to a usually non-linear activation function, $\mathbf{y}^{(k)} = f(\mathbf{x}^{(k)}) = (f(x_1^{(k)}), \ldots, f(x_{N_k}^{(k)}))$, such that $\mathbf{y}^{(k)}$ renders the layer-wise output of layer k. Such an activation function can, for instance, be a perceptron activation $f_\Theta(x) = (\mathrm{sign}(x) + 1)/2$ realized through the Heaviside theta function, a simple sign function $f_\pm(x) = \mathrm{sign}(x)$, a softer, continuous activation function such as the hyperbolic tangent $f_{\mathrm{th}}(x) = \tanh(x)$, a sigmoid activation function $f_\sigma(x) = 1/(1 + \exp(-x))$, a rectified linear unit ("relu") activation function $f_{\mathrm{relu}}(x) = \max(x, 0)$, a clipping activation $f_{\mathrm{clip}}(x) = \max(\min(x, 1), -1)$, or even the identity transformation $f_\mathrm{I}(x) = x$ (the later being a linear activation function). There are notably many more activation functions possible as they can be designed on demand.

Here we described a so-called *dense* feed-forward ANN architecture, which is, arguably, conceptually the simplest way of describing these tensor like ANNs: information which is fed into the network, $\mathbf{x}^{(\mathrm{I})}$, is gradually processed layer by layer. The goal of a typical supervised learning task involving such feed forward ANNs is to optimize the weights, $\mathcal{W}^{(kl)}$, and biases, $\mathbf{b}^{(k)}$, such that the output, $\mathbf{y}^{(\mathrm{O})}$, of the ANN reproduces the correct output, $\mathbf{Y}^{(\mathrm{O})}$, of already known *training* data, as we discuss in the next section. By constraining the possible connections (see convolutional layers or residual blocks of ANNs) or by allowing feedback loops (as done so in recurrent ANNs), the topology of the network can be modified in a way that it can cope more efficiently with a given task. The success of the learning procedure strongly depends on the network topology but also on the choice of activation functions, the cost function and on the learning algorithm.

Unsupervised Learning Using NEAT. For certain tasks, such as (for instance) object classification in images, usually a training data set can be defined which maps a set of input data, $\mathbf{x}^N = (\mathbf{x}_1^{(\mathrm{I})}, \ldots, \mathbf{x}_N^{(\mathrm{I})})$, to a labeled set of desired output data, $\mathbf{Y}^N = (\mathbf{Y}_1^{(\mathrm{O})}, \ldots, \mathbf{Y}_N^{(\mathrm{O})})$; the weights and biases of the ANN can be trained in a supervised way until the ANN can faithfully reproduce (i.e., fit) the desired output data via the actual ANN output data, $\mathbf{y}^N = (\mathbf{y}_1^{(\mathrm{O})}, \ldots, \mathbf{y}_N^{(\mathrm{O})})$. This is usually

achieved by applying the back-propagation algorithm (3, 4), i.e., a gradient descent based algorithm, to minimize a cost-function of the output data $\mathcal{C}(\mathbf{y}^N, \mathbf{Y}^N) \rightarrow \text{min}$. The cost function might be a simple least mean squared function, i.e. $\mathcal{C}_{\text{LMSQ}}(\mathbf{y}^N, \mathbf{Y}^N) = \sum_{i=1}^{N} |\mathbf{y}_i^{(\text{O})} - \mathbf{Y}_i^{(\text{O})}|^2/N$, but can also be a more complicated function.

To a certain extent, this approach of minimizing a cost function can be understood as a fitting procedure such that given the input data, $\mathbf{x}^{\text{N}}$, the output of the ANN, $\mathbf{y}^{\text{N}}$, faithfully reproduces the desired output $\mathbf{Y}^N$. ANNs are excellent tools to perform (data fitting) tasks with high precision and with great numerical performance which are potentially too complicated or which are numerically too demanding for being implemented via more conventional algorithmic approaches. The predictive power of ANNs is rooted in the immense flexibility of how information can be processed by the network of interconnected artificial neurons. On the one hand, to properly predict the desired output of a training data set with sufficient accuracy and precision (i.e., to avoid underfitting) the architecture of the ANN needs to be complex enough. On the other hand, great caution has to be taken during training to ensure that a complex ANN does not simply learn the correct output of the training data "by heart"; such overfitting of the training data is usually associated with a poor quality of the predictions of an ANN when being subjected to new input data which are not present in the training data set. For the purpose of avoiding overfitting a variety of so-called regularization techniques exist. A successfully trained ANN will not only correctly map each element of the training data set to a provided target (within sufficiently small numerical uncertainties), but will also return meaningful results on input data that are similar to the training data set but which were never explicitly used in the training procedure. In literature, this behavior is often referred to as the generalization capabilities of ANNs; it is thus often assumed that ANNs are able to learn concepts of a given task after being trained on a task specific data set rather than plainly fitting the data (this of course highly depends on the topology of the used ANNs but also on the particularly employed training procedure) (4).

In the unsupervised learning task that we employ in this manuscript no meaningful mapping between input data and desired output data of the ANN can be provided *a priori*. This fact makes the learning procedure drastically more complicated since standard algorithms such as back-propagation cannot simply be applied. In our example of a microswimmer learning chemotaxis, the ANN needs to be trained by directly experiencing the problem at hand by trial and error (and positive reinforcement through a suitable reward function). Many reinforcement learning (RL) algorithms (5) are available nowadays (such as Q-learning (QL) (6), deep deterministic policy gradient (DDPG) (7), asymmetric actor critic agents (A3C) (8), or proximal policy optimization (PPO) (9)) which are, in principle, capable of performing RL tasks. In general, their performance depends on the specific problem and often requires parameter fine tuning of the employed RL algorithm.

In our case of a microswimmer learning chemotaxis we rely on the biologically inspired unsupervised learning algorithm of *NeuroEvolution of Augmenting Topologies* (NEAT) (10). In contrast to most learning algorithms NEAT does not only optimize the weights of an ANN in order to optimize a so-called target function, but, moreover, evolves the weights and the topology of the ANN simultaneously (10). This algorithm helps identifying ANN solutions of minimal complexity (i.e. a minimal number of neurons) required to successfully cope with a target task.

The NEAT algorithm we employ in this contribution (11) maintains a population (of fixed size) of different realizations of ANNs and subjects them to an evolutionary process which mainly involves the concepts of generations, reproduction, mutations and artificial selection. At each iteration step of the algorithm the current realization of the population renders a so-called generation. For each generation every ANN is subjected to perform several cycles of a given RL task (such as, for instance, to maximize the distance of uni-directional locomotion of a microswimmer in phase one or performing chemotaxis in phase two of the main text, respectively). The quality of each ANN is measured by a fitness value which numerically quantifies how well (or how poor) an ANN solution is adapted to execute the target task. In our case, the NEAT fitness value is chosen as the cumulative reward of the RL task after one episode (see phase one in the main text) or the minimum cumulative reward amongst several, qualitatively different episodes (see phase two in the main text); many other problem specific choices for the reward scheme are possible. ANN solutions with high fitness values are in contrast to those with low fitness values more likely to be selected for reproduction in order to form the next generation of the NEAT algorithm. In that way, good traits are favoured during reproduction and will prevail over time while others will eventually perish. During reproduction two different ANNs are merged together, following certain rules (11). The resulting ANNs are then subject to random mutations which allows the algorithm to explore disconnected regions in configuration space, evolving the entire population gradually – in the course of several, successive generations – towards an optimal solution to the target task.

Now we focus on the mere ANN architecture inherent to the algorithm: the above introduced layer-by-layer topology of, for instance, dense ANNs can be generalized to a graph-based architecture such that every node can in principle be connected with every other node in the network. This feature can be realized by either maintaining a feed-forward architecture (4) or, by allowing feedback circles over iterative time steps, in a recurrent way (4). The NEAT genetic algorithm relies on this generalized graph based approach for describing the architecture of ANNs, as explained in the following.

Each of the $n = 1, \ldots, N$ neurons of an ANN represents a uniquely labeled node with index ν_n, to which an associated vector of $j = 1 \ldots, N_n$ input values is assigned, $\mathbf{v}_n = (v_{n\,1}, \ldots, v_{n\,N_n})$; these values are the output signals from other nodes (or directly the input to the network). The node input is then accumulated in $A_n(\mathbf{v}_n) : \mathbb{R}^{N_n} \rightarrow \mathbb{R}$, either by summation, $A_n(\mathbf{v}_n) \equiv \sum_{j=1}^{N_n} v_{n\,j}$, by product, $A_n(\mathbf{v}_n) \equiv \prod_{j=1}^{N_n} v_{n\,j}$, or a similar operation, before the output of the neuron, $o_n = f_n(A_n(\mathbf{v}_n) + b_n)$, is evaluated through a non-linear activation function $f_n(x) \in \mathbb{R}$; here $b_n \in \mathbb{R}$ is a so-called bias to the accumulated input $A_n(\mathbf{v}_n)$. Note that additional to the bias b_n also the accumulation function, $A_n(\cdot)$, as well as the activation function, f_n, are specific to each neuron and represent optimization features of the NEAT algorithm. Next, the connectivity of neuron ν_n with other neurons, ν_m, (ν_n being input to ν_m) in the network is realized by scalar weights, $\{w_{m\,n}\}$, such that the vector product of all defined (or non-zero w_{nm}) weights with the related neurons, i.e. $\mathbf{v}_m = (v_{m\,1}, \ldots, v_{m\,N_m}) = \{w_{m\,n} \cdot o_n\}$, represents the input to neuron ν_m.

Such a network can have a far more complicated architecture as layer-wise feed-forward networks. However, ANNs evolved by the NEAT algorithm have the tremendous potential of solving specialized tasks with a minimum number of parameters (10).

Phase One: Swimmer Action Layer. In this contribution we rely on artificial ANNs to control the arm lengths of a three bead model of a microswimmer which can learn to self-propel through viscous, hydrodynamic media. Here we will discuss the sub-network discussed in phase one in the main text, i.e. the swimmer action layer (SAL) whose objective is to actively maximize the swimmer's center of mass position, $x_c(t) = (x_1(t) + x_2(t) + x_3(t))/3$, over time given the swimmer's arm lengths, $L_1(t)$, $L_2(t)$ with $L_T(t) = L_1(t) + L_2(t)$ and the arm velocities $V_1(t) = d\,L_1(t)/dt$, $V_2(t) = d\,L_2(t)/dt$ with $V_T(t) = V_1(t) + V_2(t)$. The SAL gains control over the swimming gaits by suggesting forces $F_1(t), F_3(t)$ which are applied to the outermost beads located at $x_1(t)$ and $x_3(t)$ of the three-bead swimmer at each instance of time. To improve readability, we henceforward drop the explicit time-dependency of the above mentioned quantities, unless we want to explicitly emphasize the time-dependence.

We define the input variables, L_1, L_2 and L_T as well as V_1, V_2 and V_T, in units of L_0 and $L_0 T_0^{-1}$, respectively (see main text for definition of L_0 and T_0), and collect these scalar quantities in an input vector $\mathbf{X}$ as

$$\mathbf{X} = \left(\frac{L_1}{L_0}, \frac{L_2}{L_0}, \frac{L_T}{L_0}, \frac{V_1}{L_0 T_0^{-1}}, \frac{V_2}{L_0 T_0^{-1}}, \frac{V_T}{L_0 T_0^{-1}} \right). \quad [5]$$

The output of the ANN, i.e. the forces, can also be defined in vector form as $\mathbf{F} = (F_1(\mathbf{X}), F_3(\mathbf{X}))$ and is written as an explicit function of the input variables.

Swimmer Action Layer Solutions. As described in the main text, we perform ten independent training runs each consisting of 1000 consecutive NEAT generations. For phase one training we always use the swimmer initial conditions $L_1(0) = L_0$ and $L_2(0) = L_0$.

Our NEAT training revealed solutions consisting of a dense SAL, directly connecting the input values, $\mathbf{X}$, with the output neurons, $\mathbf{F}$, without using any hidden neurons (see main text and Fig. S3). The corresponding relation can be written as

$$\mathbf{F}_{\mathrm{SAL}}(\mathbf{X}) = F_0 \; f_{\mathrm{th}} \left(\mathcal{W}_{\mathrm{SAL}} \cdot \mathbf{X} + \mathbf{b}_{\mathrm{SAL}} \right) = (F_1(\mathbf{X}), F_3(\mathbf{X})). \quad [6]$$

Here, $\mathcal{W}_{\mathrm{SAL}}$ is the 2×6 dimensional weight matrix (connecting the six dimensional input vector with the two output neurons), $\mathbf{b}_{\mathrm{SAL}}$ is a two dimensional bias vector and $f_{\mathrm{th}}(\mathbf{x})$ represents an element-wise "$\tanh(x)$" activation function, i.e. $f_{\mathrm{th}}(\mathbf{x}) = [\tanh(x_1), \ldots, \tanh(x_N)]$ for an N dimensional vector $\mathbf{x}^N = (x_1, \ldots, x_N)$. The output of the SAL, all together given by Eq. (6), represent the instantaneous actions $F_1(\mathbf{X}), F_3(\mathbf{X}) \in [-F_0, F_0]$, which are applied in each time step during swimming in the hydrodynamic environment.

The minimal complexity SAL, which we used in the main text, can be described by the following sparse weight matrix, $\mathcal{W}_{\mathrm{SAL}}$ and bias vector $\mathbf{b}_{\mathrm{SAL}}$:

$$\mathcal{W}_{\mathrm{MC-SAL}} = \begin{pmatrix} 0 & 20.202 & 0 & 0 & 0 & 0 \\ 5.720 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \text{and} \quad \mathbf{b}_{\mathrm{MC-SAL}} = \begin{pmatrix} -18.559 \\ -5.412 \end{pmatrix}, \quad [7]$$

which only requires L_1 and L_2 as input and consists of four ANN parameters in total. In the main text, we introduced the weights $w_1 = 20.2/L_0$ and $w_2 = 5.7/L_0$ which correspond to the matrix elements $(\mathcal{W}_{\mathrm{MC-SAL}})_{12}/L_0$ and $(\mathcal{W}_{\mathrm{MC-SAL}})_{21}/L_0$, respectively, while the biases b_1 and b_2 are collected in the vector $\mathbf{b}_{\mathrm{MC-SAL}} = (b_1, b_2)$. From the definition of the tanh function we can see that for lengths $L_{2,1} > L^*_{2,1} = -b_{1,2}/w_{1,2}$ the respective forces $F_{3,1}$ are positive, and otherwise negative, leading to a simple phase-shifted periodic output (Fig. 2A in main text and Movie S2) with period $T_S \approx 217 T_0$. The magnitudes of w_1 and w_2 determine the steepness of the tanh function and hence how fast the forces approach their maximum values when crossing $L^*_{2,1}$ (see Movie S2).

With the NEAT genetic algorithm we were able to identify even more efficient solutions for $\mathcal{W}_{\mathrm{SAL}}$ and $\mathbf{b}_{\mathrm{SAL}}$ in terms of maximal average swimming velocity for uni-directional locomotion strategies via a SAL given by Eqs. (5) and (6). However, such solutions are related to a more complicated internal architecture of the two-neuron based SAL. As illustrated in Fig. S3 there is a class of solutions of vastly different complexity which obtain very similar fitness values, i.e. very similar mean swimming velocity. One example of this class of optimal solutions requires only four connections. The associated weights and biases, denoted by $\mathcal{W}_{\mathrm{O-SAL-1}}$ and $\mathbf{b}_{\mathrm{O-SAL-1}}$, are given by

$$\mathcal{W}_{\mathrm{O-SAL-1}} = \begin{pmatrix} -9.613 & 50.537 & 0 & 0 & 0 & 0 \\ 30.125 & 20.763 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \text{and} \quad \mathbf{b}_{\mathrm{O-SAL-1}} = \begin{pmatrix} -32.050 \\ -50.604 \end{pmatrix}. \quad [8]$$

The O-SAL-1 solution has a stroke period of $T_S \approx 268 T_0$ and is illustrated by the black ANN in the top left inset of the left panel of main text Fig. 2A.

The ANN with the absolutely largest fitness which we have encountered in ten independent NEAT training runs (see Figs. S1 to S4) additionally features one hidden neuron that forwards the weighted, sigmoid-activated input V_2 to the F_3 output neuron (see gray ANN in the top left inset of the left panel of main text Fig. 2A). Arguably, the terminology *swimmer action layer* does not fully apply to this kind of ANN but for the sake of simplicity we refer to this NEAT ANN solution as *optimal SAL 2* (O-SAL-2). In Matrix notation, the O-SAL-2 NEAT ANN can be described by introducing an additional sigmoid activated layer for the hidden neuron as

$$x_\sigma(\mathbf{X}) = f_\sigma \left(\mathcal{W}^{(1)}_{\mathrm{O-SAL-2}} \cdot \mathbf{X} + b^{(1)}_{\mathrm{O-SAL-2}} \right), \quad [9]$$

with a (sparse) weight matrix and bias given by

$$\mathcal{W}^{(1)}_{\mathrm{O-SAL-2}} = (0,0,0,0,-132.173,0) \quad \text{and} \quad b^{(1)}_{\mathrm{O-SAL-2}} = 4.453. \tag{10}$$

The scalar output $x_\sigma(\mathbf{X})$ given by Eq. (9) can now be used to extend the regular SAL input $\mathbf{X} = (L_1/L_0, \ldots, V_\mathrm{T}/L_0T_0^{-1})$ defined in Eq. (5) as

$$\mathbf{X}_\sigma = \mathbf{X} \cup (x_\sigma(\mathbf{X})) = \left(\frac{L_1}{L_0}, \ldots, \frac{V_\mathrm{T}}{L_0T_0^{-1}}, x_\sigma(\mathbf{X})\right) \tag{11}$$

and, subsequently, the extended input $\mathbf{X}_\sigma \in \mathbb{R}^7$ can be subjected to a tanh-activated SAL given by Eq. (6), but with a 2×7 weight matrix $\mathcal{W}^{(2)}_{\mathrm{O-SAL-2}}$ and an associated bias $\mathbf{b}^{(2)}_{\mathrm{O-SAL-2}}$ defined by

$$\mathcal{W}^{(2)}_{\mathrm{O-SAL-2}} = \begin{pmatrix} 0 & 43.705 & -5.746 & 0 & 0 & -26.753 & 0 \\ 12.339 & 0 & 27.420 & 0 & 20.406 & 0 & -68.68 \end{pmatrix} \quad \text{and} \quad \mathbf{b}^{(2)}_{\mathrm{O-SAL-2}} = \begin{pmatrix} -26.998 \\ 0.947 \end{pmatrix}. \tag{12}$$

The O-SAL-2 solution has a stroke period of $T_S \approx 260T_0$ and is illustrated by the gray ANN in the top left inset of the left panel of main text Fig. 2A.

In this contribution our choice for the specific set of the weights and biases of the SAL is given by Eq. (7) due to the strikingly simple form of the weight matrix. We use these weights to further study microswimmer chemotaxis with a more advanced chemotaxis agent (see main text Fig. 3A).

Comparison of microswimmer fitness to 4-state solutions. We now compare the fitness of our optimum SAL solutions ($\bar{v} \sim 1.36 \cdot 10^{-3} R/T_0$) to 4-state solutions where the microswimmer expands/contracts one arm, while the second arm length is constant. In such a way the arms perform a square in (L_1,L_2) phase space (see Fig. S6).

The fact that we use elastic restoring forces with spring stiffness $K = F_0/R$ allows the arms to approximately take values of the lengths $L_\alpha = L_{min} - R = 0.6L_0 = 6R$ and $L_\beta = L_{max} + R = 1.4L_0 = 14R$ (see e.g. solutions in Fig. 2A in main text). Hence we consider the lengths $L_1, L_2 \in \{L_\alpha, L_\beta\}$ in the 4-state calculations.

In order to model the dynamics of the arms in the 4-state solution we consider two possible scenarios: First, we allow the arms to change their length with constant speed $|V_{1,2}| = 0.138R/T_0$, which is approximately the fasted velocity possible in our system: it occurrs when $F_1 = F_2 = \pm F_0$ and hence $F_2 = \mp 2F_0$ (see Eqs. (1) to (3)). We solve Eqs. (1) to (3) numerically for the four subsequent paths in (L_1,L_2) space under the constraint of constant arm velocities and evaluate the time dependent arm and center of mass velocities and the forces necessary to fulfill the motion. This leads to a mean swimming speed (fitness) of $\bar{v} = 2.1 \cdot 10^{-3} R/T_0$ and is hence about 50% faster than our O-SAL solutions despite the fact that the stroke period is $T_S = 4(L_\beta - L_\alpha)/|V_{1,2}| = 232T_0$ and hence about 10% larger compared to our O-SAL solutions ($T_S = 260T_0$). Note, however, that this high fitness is only possible for this 4-state solution since the forces are allowed to exceed $\pm F_0$, as can be seen in Fig. S6A.

In a second case of 4-state solutions (Fig. S6B), the forces are not allowed to exceed $\pm F_0$, but the arms are allowed to move as fast as possible under this constraint. The forces on the beads can be obtained as follows: For example, when the arm lengths (L_1, L_2) pass from state (L_α, L_α) to (L_β, L_α) the first arm stretches with maximum possible velocity when $F_1 = -F_0$, and F_3 (and hence $F_2 = -F_1 - F_3$) can be obtained from the constraint that L_2 has to be constant using again Eqs. (1) to (3). Similar the other three state transitions can be computed. In this way we obtain a relatively large stroke period $T_S = 449T_0$ and fitness $\bar{v} = 1.08 \cdot 10^{-3} R/T_0$. Hence, for this case, our O-SAL solutions are about 25% faster than this 4-state solution reiterating the fact that our RL algorithm optimizes the nontrivial policy and dynamics in (L_1, L_2) shape space which differ considerably from the simple 4-state solution.

MC-SAL solution as dynamical system. Inspired by the different quality of SAL solutions obtained in phase one (swimming vs. resting arm lengths, see also Fig. S3B,C), we consider as an important example the MC-SAL topology with solution $F_1 = F_0 \tanh(w_1 L_2 + b_1)$ and $F_3 = F_0 \tanh(w_2 L_1 + b_2)$ (see main text), which shows both swimming and resting solutions depending on the value of the weights. Together with restoring forces F_R (Eq. (4)) the total forces on the beads can be written as $F_1(L_2) + F_R(L1)$ and $F_3(L_1) - F_R(L_2)$ and are hence independent of the arm velocities which do not enter in the obtained MC-SAL policy. Therefore, the forces can be eliminated from Eqs. (1) to (3), and transformed into a dynamical system for the arm lengths $d\mathbf{L}/dt = \mathbf{V}(\mathbf{L}; \mathbf{W})$ using $\mathbf{L} = (L_1, L_2)$, $\mathbf{V} = (V_1, V_2)$ and the weight vector $\mathbf{W} = \{w_1, w_2, b_1, b_2\}$ (as control parameters):

$$12\pi \frac{dL_1}{dt} = (-4 + \frac{6}{L_1})F_R(L_1) + (2 - \frac{3}{L_1} - \frac{3}{L_2} + \frac{3}{L_1 + L_2})F_R(L_2) - 4\tanh(b_1 + L_2 w_1) + \frac{6\tanh(b_1 + L_2 w_1)}{L1} \tag{13}$$

$$- 2\tanh(b_2 + L_1 w_2) + \frac{3\tanh(b_2 + L_1 w_2)}{L_1} + \frac{3\tanh(b_2 + L_1 w_2)}{L_2} - \frac{3\tanh(b_2 + L_1 w_2)}{L1 + L2} \tag{14}$$

$$12\pi \frac{dL_2}{dt} = -\frac{3\tanh(b_1 + L_2 w_1)}{L_1} + \frac{3\tanh(b_1 + L_2 w_1)}{L_1 + L_2} - \frac{3\tanh(b_1 + L_2 w_1)}{L_2} + 2\tanh(b_1 + L_2 w_1) \tag{15}$$

$$- \frac{6\tanh(b_2 + L_1 w_2)}{L_2} + 4\tanh(b_2 + L_1 w_2) + F_R(L_1)\left(\frac{3}{L_1 + L_2} - \frac{3}{L_1} - \frac{3}{L_2} + 2\right) + \left(\frac{6}{L_2} - 4\right) F_R(L_2) \tag{16}$$

We can now numerically integrate this system of ordinary differential equations and obtain solutions of phase space curves $(L_1(t), L_2(t))$ which depend on the initial conditions $\mathbf{L}_0$ and the parameters (weights $\mathbf{W}$). For a given set of initial conditions and parameters our system approaches a stable attractor in the two-dimensional (L_1, L_2) phase space, which is either a fixed point or a limit cycle. For example, as expected, inserting the $\mathbf{W}$-values from the MC-SAL solution presented in the main text, then results in stable limit cycle solutions corresponding to swimming solutions. By tuning the parameters we are able to identify transitions from a resting state (fixed point) to a swimming state (limit cycle). This we have illustrated in Fig. S7 where we have used the same weights and biases obtained for the MC-SAL solution shown in the main text, but only varied the weight w_1. We can see that for small w_1 both arms expand until a fixed point length $\mathbf{L}^* = (L_{max} + R, L_{max} + R)$ is reached. When increasing w_1, this fixed point is shifted and becomes unstable, and a bifurcation to a stable limit cycle in (L_1, L_2) phase space emerges which corresponds to swimming solutions, including the MC-SAL solution presented in the main text $(w_1 = 2.02/R)$. By increasing w_1 further the swimming solutions disappear, and via a second bifurcation another stable fixed point at $\mathbf{L}^* = (L_{min} - R, L_{min} - R)$ appears, where the swimmer ends in a stable contraction of the arm lengths.

Conditional Permutation Control: Employing Symmetries of the Model. The equations of motion of the three bead swimmer are mirror symmetric for rightward and leftward motion: applying $-F_3(t)$ to bead one (x_1) and $-F_1(t)$ to bead three (x_3) following a force-trajectory of swimming to the right (see Fig. 2A in the main text) the microswimmer will move to the left (i.e. $v_i(t) \to -v_i(t)$).

Such symmetries of the governing equations of motion can be employed, in general, not only by predefined action-trajectories but also directly by the policy of an agent whose input and output can be related and transformed by a corresponding symmetry transformation. In our case, swimming to the left can be realized with the same policy as swimming to the right (given, for instance, by the SAL defined in Eq. (6)), but in a mirror symmetric way:

$$\mathbf{F}_{\text{perm. SAL}}(\mathbf{X}) = \mathcal{P}_F \cdot \mathbf{F}_{\text{SAL}}(\mathcal{P}_x \cdot \mathbf{X}). \tag{17}$$

Eq. (17) represents the permuted SAL policy for intentional leftward motion of the microswimmer. To evaluate the forces $\mathbf{F}_{\text{perm. SAL}}(\mathbf{X})$ defined by Eq. (17) we first permute the input, $\mathbf{X}^* = \mathcal{P}_x \cdot \mathbf{X}$, apply the SAL to the permuted input according to Eq. (6), i.e. $\mathbf{F}^*_{\text{SAL}} = \mathbf{F}_{\text{SAL}}(\mathbf{X}^*)$, and transform the output (which has been evaluated for rightward motion) accordingly, $\mathbf{F}_{\text{perm. SAL}} = \mathcal{P}_F \cdot \mathbf{F}^*_{\text{SAL}}$. The permutation matrices, $\mathcal{P}_x$ and $\mathcal{P}_F$, are defined by

$$\mathcal{P}_x = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{18}$$

exchanging $L_1 \leftrightarrow L_2$ and $V_1 \leftrightarrow V_2$, and

$$\mathcal{P}_F = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} \tag{19}$$

exchanging $F_1 \leftrightarrow -F_3$. Note that $\mathcal{P}_x\mathcal{P}_x = \mathcal{I}_x$ and $\mathcal{P}_F\mathcal{P}_F = \mathcal{I}_F$ yield the 6×6 and 2×2 identity transformations when being applied twice thus forming a group with two elements (left- and rightward motion) acting on the input $\mathbf{X}$ and on the output $\mathbf{F}$, respectively.

If an agent has access to additional information about the environment (such as the chemotaxis agent shown in main text Fig. 3A which can sense the value of a chemical field, $c(x)$), it can decide which operation to perform in order to maximize the reward (i.e. if swimming to the left or to the right is the better choice). When some basic operations are known from a preceding training-stage (e.g. phase one in the main text of this contribution), the agent can make use of this already obtained knowledge and does not need to learn the entire, rather complex task at once, which makes a subsequent, more specialized training considerably more efficient. In this contribution, the agent's task in phase two in the main text is to identify the chemical gradient $\nabla c(x)$ and to swim towards the maximum of the chemical field. If the gradient is available as a binary variable, i.e. $g = 0$ for $\nabla c(x) > 0$ and $g = 1$ for $\nabla c(x) < 0$, this information can be used to apply the linear transformation given by Eq. (17), dependent on the value of g, as

$$\mathbf{F}_{\text{PCL}}(\mathbf{X}, g) = \mathcal{P}^g_F \cdot \mathbf{F}_{\text{SAL}}(\mathcal{P}^g_x \cdot \mathbf{X}), \tag{20}$$

with $\mathcal{P}^0_x = \mathcal{I}_x$ and $\mathcal{P}^0_F = \mathcal{I}_F$; thus, either Eq. (6) or Eq. (17) is applied depending on the value of $g = 0$ or $g = 1$, respectively.

As we demonstrate via Eq. (20) both motions (left- and rightward self-propulsion) can be related with a conditional group transformation of the input (and appropriately of the output) of the SAL. We introduce two permutation control layers (PCL) in the chemotaxis agent (main text Fig. 3A) which employ the transformation given by Eq. (20) as a function of the prediction D of the sign of the chemical gradient $\nabla c(x)$ (see main text): the conditional transformation given by Eq. (20) acting on both, the input and output of the SAL can be realized as two separate layers in an ANN, i.e.

$$\mathbf{X}_D = \left[f_{\Theta_-}(D) \times \mathcal{P}_x + (1 - f_{\Theta_-}(D)) \times \mathcal{I}_x\right] \cdot \mathbf{X} \quad \text{and} \tag{21}$$

$$\mathbf{F}_D = \left[f_{\Theta_-}(D) \times \mathcal{P}_F + (1 - f_{\Theta_-}(D)) \times \mathcal{I}_F\right] \cdot \mathbf{F}_{\mathrm{SAL}}(\mathbf{X}_D), \tag{22}$$

which take respectively as input (i) $\mathbf{X}$ and $\mathbf{F}$ and, additionally, (ii) the predicted sign D of the gradient. The PCL controls via a negative perceptron activated channel, $g = f_{\Theta_-}(D) = \Theta(-D)$, if the permutation is applied ($g = 1$) or not ($g = 0$). The PCL approach is schematically visualized in Fig. S8.

The binary conditional transformation proposed in this contribution can directly be generalized to more complex tasks featuring more than two operations which are related by symmetry transformations of their input variables and their output. In that way, an agent only needs to learn how to perform one operation (or a few, qualitatively different operations) $g_i(\mathbf{X})$ from a set of N_G operations, $\{g_1(\mathbf{X}), \ldots, g_{N_\mathrm{G}}(\mathbf{X})\}$, which are related by transformations of the agent's in- and output variables, $g_j(\mathbf{X}) = \mathcal{O}_{ji}(g_i(\mathcal{I}_{ij}(\mathbf{X}))$: first, $\mathcal{I}_{ij}(\cdot)$ transforms the input $\mathbf{X}$, required to perform a particular operation g_j, into the frame of reference of the known operation g_i. Subsequently, the operation g_i is employed and the output of operation g_i is transformed by $\mathcal{O}_{ji}(\cdot)$ into the frame of reference of operation g_j. The agent then only needs to learn how to decide, which of the set of transformations, $\{g_1(\mathbf{X}), \ldots, g_{N_\mathrm{G}}(\mathbf{X})\}$, to apply based on N_D additional measurements $\mathbf{D} = \{D_1, \ldots, D_{N_\mathrm{D}}\}$ of the environment; the total input of the agent is the union $\mathbf{X} \cup \mathbf{D}$.

Phase Two: Fitness Function and Training of the Chemotaxis Agent. To train the chemotaxis agent depicted in Fig. 3A in the main text we define the reward function $r_\mathrm{c} = \sum_{i=1}^{N_\mathrm{I}} [x_\mathrm{c}(t_i) - x_\mathrm{c}(t_{i-1})] D(t_i)$, where $D(t_i) = \mathrm{sign}[\nabla c(x_\mathrm{c}(t_i))] = \pm 1$ represents the sign of the gradient at consecutive instances of time $t_i = iT_0 = t_{i-1} + T_0$, and N_I the number of steps (see main text). In that way, r_c represents the total distance that a particular microswimmer realization moves along an ascending gradient during the total integration time $T_\mathrm{I} = N_I T_0$.

We train the different chemotaxis agents (main text Fig. 3) solely on piece-wise linear chemical fields, $c(x) = \max(0, a - k|x - x_0|)$, assuming amplitudes a in the range of $a/c_0 \in [98, 102]$ and slopes k in the range $k/c_0 R^{-1} \in [0.99, 1.01]$. The fields are centered at x_0 and below we use positive and negative values of x_0 to allow positive and negative slopes k.

Based on the general reward scheme for RL learning introduced in the main text we define a (necessarily) more complicated NEAT fitness function for phase two learning (compared to phase one): during each RL step every microswimmer chemotaxis agent from the current NEAT generation has to perform chemotaxis for $N_c = 4$ different chemical environments, namely at $x_0/R \in [-10, -0.4, 0.4, 10]$, for given values of a and k specified in $N_c = 4$ independent episodes. To avoid that the agent learns the field values or field gradients by heart, we chose the values for a and k at random at the beginning of each episode (however, within the related ranges defined above). Notably, for each generation of $N = 288$ ANN based microswimmer agents in the NEAT genetic algorithm (i.e., for each RL step) we use the same chemical field amplitude of $a = (1 \pm 2 \times 10^{-2}\, \delta a) a_0$ with a slope of $k = (1 \pm 1 \times 10^{-2}\, \delta k) k_0$, where $a_0 = 100 c_0$ and $k_0 = 1 c_0/R_0$. Furthermore, we allow random initial configurations of the arm-lengths $L_1(0) = \delta L_1$ and $L_2(0) = \delta L_2$ in the range $[L_\mathrm{min}, L_\mathrm{max}]$ for each RL episode, always ensuring that $x_c(0) = 0$; the uniformly distributed random numbers $\delta a, \delta k \in [0, 1]$ and $\delta L_1, \delta L_2 \in [L_\mathrm{min}, L_\mathrm{max}]$ vary for each realization of a microswimmer agent. We evaluate the respective chemotactic reward r_c for every microswimmer instance in a NEAT generation with the specified field parameters a and k for all four field positions $x_0/R \in [-10, -0.4, 0.4, 10]$. We then define the NEAT fitness $\bar{v}_D = \min(r_\mathrm{c})/T_\mathrm{I}$ of one RL step for every individual microswimmer as proportional to the minimum value $\min(r_\mathrm{c})$ of the $N_c = 4$ cumulative rewards which the respective chemotaxis agent has collected in the independent episodes. Similar to phase one, $\bar{v}_D$ represents the mean swimming velocity but now projected onto the direction up the gradient. The objective of RL of chemotaxis via NEAT is to identify a NEAT ANN of a chemical gradient block (see main text Fig. 3B,C) by means of genetic processes with whom the fitness value related to phase two is maximized.

We present the learning progress (and details on the different training procedures) for the chemotaxis agents depicted in Fig. 3 in the main text for spatial gradient sensing in Fig. S9 and for temporal gradient sensing in Fig. S12 to S14, respectively; notably, we also used smaller slopes of $k_0 = 0.1\ c_0/R$ for the training of temporal gradient sensing agents (see Fig. S12 to S14). The corresponding NEAT ANNs for the spatial and temporal gradient sensing chemotaxis agents (see Figs. 3C,E) are specified below.

Spatial Chemical Gradient Estimation. The spatial chemical gradient estimator (SG) introduced in Fig. 3B of the main text features an ANN that is to be identified by the NEAT algorithm via phase two RL (defined in the main text and above). The NEAT ANN solution for the SG agent depicted in Fig. 3D of the main text (see also Movie S3) can be represented by a single layer ANN. The chemical field at all three bead positions is measured in units of c_0 by the agent such that the input, $\mathbf{x}_\mathrm{SG}$, and output, G, of the NEAT ANN become

$$\mathbf{x}_\mathrm{SG} = \frac{1}{c_0}\left(c(x_1(t_i)), c(x_2(t_i)), c(x_3(t_i))\right) \tag{23}$$

$$G = f_\mathrm{clip}(\mathcal{W}_\mathrm{SG} \cdot \mathbf{x}_\mathrm{SG} + b_\mathrm{SG}), \tag{24}$$

with weights and biases given by

$$\mathcal{W}_\mathrm{SG} = (-71.147, 0, 71.16) \quad \text{and} \quad b_\mathrm{SG} = 1.373, \tag{25}$$

and with $f_{\text{clip}}(x) = \max(\min(1, x), -1)$ being the clip-activation function. The NEAT ANN output G given by Eq. (24) is forwarded to the output neuron of the SG agent $D(G)$ (see main text Fig. 3B) as

$$D = f_{\pm}(G), \tag{26}$$

where $f_{\pm}(x) = \text{sign}[x]$ ensures that $D \in \{-1, 1\}$.

Recurrent Chemical Gradient Estimation. For the temporal chemical gradient sensing we evaluate the chemical gradient by temporal means, as schematically visualized in Fig. 3C of the main text. We define the input, $\mathbf{x}_{\text{TG}}(t)$, for temporal chemical gradient sensing in the following way

$$\mathbf{x}_{\text{TG}}(t_i) = \left(\frac{L_{\text{T}}(t_i)}{L_0}, \frac{V_{\text{T}}(t_i)}{L_0 T_0^{-1}}, \frac{c(x_{\text{c}}, t_i)}{c_0} \right), \tag{27}$$

capturing the total arm length $L_{\text{T}}(t_i)$ and velocity $V_{\text{T}}(t_i)$ of the swimmer in units of L_0 and $L_0 T_0^{-1}$, respectively, and the instantaneous value of the chemical field at the center of mass position, $c_c(t_i) = c(x_{\text{c}}, t_i)$, in units of c_0. The goal of the temporal CG block is to predict the sign $D(t_i)$ of the chemical gradient $\nabla_x c_c(t_i)$ at the swimmer's center of mass x_{c} at time t_i.

As mentioned in the main text, we subdivide the temporal CG block into (i) a NEAT ANN block and (ii) into a recurrent chemical memory control (CMC) cell (see main text Fig. 3C). (i) The NEAT ANN's primary goal is to predict the temporal gradient $G_y(t_i) \sim \nabla_x c_c(t_i)$ but via a second output $C_y(t_i)$ it controls the future action of the CMC cell. (ii) The CMC cell pre-processes, based on the recurrent value of $C_y(t_{i-1})$, the value of the chemical field $c_c(t_i)$ for the NEAT ANN using an internal memory $M(t_i)$ of the chemical field. Based on this recurrent chemical interface, i.e. on the input $\mathbf{x}_{\text{CMC}}(t_i)$ defined as

$$\mathbf{x}_{\text{CMC}}(t_i) = (C_y(t_{i-1}), G_y(t_{i-1}), M(t_{i-1})) \tag{28}$$

we define a particular method how the recurrent information is processed between successive time steps t_{i-1} and t_i and how the memory unit is updated based on the recurrent control signal. For that purpose we introduce the following coupled, conditional equations

$$C_x(t_i) = \alpha\, C_y(t_{i-1}) + \beta \left(\frac{c_c(t_i)}{c_0} - M(t_{i-1}) \right), \tag{29}$$

$$G_x(t_i) = G_y(t_{i-1}), \tag{30}$$

$$M(t_i) = \alpha\, M(t_{i-1}) + \beta \left(\frac{c_c(t_i)}{c_0} + \frac{\delta c}{c_0} \right), \tag{31}$$

which, in our case, are entirely controlled by the two binary variables $\alpha = (1 - f_\Theta(C_y(t_{i-1})))$ and $\beta = 1 - \alpha = f_\Theta(C_y(t_{i-1}))$ through a perceptron activation, $f_\Theta(C_y(t_{i-1})) = \Theta(C_y(t_{i-1}))$. The random variable δc in Eq. (31) accounts for noisy memory readings as discussed in the main text. An ANN representation of the CMC cell defined via Eqs. (29) to (31) is shown in Fig. S11A.

The CMC cell provides the NEAT ANN with recurrent inputs $C_x(t_i)$ and $G_x(t_i)$ which cover information about the temporal field gradient. The NEAT ANN, that is to be identified by the NEAT algorithm via phase two RL (discussed in the main text and above) maps the input $\mathbf{x}_R(t_i)$, defined as

$$\mathbf{x}_R(t_i) = \left(\frac{L_{\text{T}}(t_i)}{L_0} \frac{V_{\text{T}}(t_i)}{L_0 T_0^{-1}}, C_x(t_i), G_x(t_i) \right), \tag{32}$$

onto two scalar output values, $\mathbf{y}_R(t_i) = \mathcal{N}(\mathbf{x}_R(t_i))$, i.e.

$$\mathbf{y}_R(t_i) = (G_y(t_i), C_y(t_i))\,. \tag{33}$$

$G_y(t_i)$ is the predicted chemical gradient at the swimmer's center of mass x_{c} at time t_i and $C_y(t_i)$ is the recurrent control output forwarded to the CMC cell.

The output of the entire temporal CG block is the predicted sign $D(t_i)$ of the chemical gradient $\nabla_x c_c(t_i)$ which we define as

$$D(t_i) = f_{\pm}(G_y(t_i)), \tag{34}$$

where $f_{\pm}(x) = \text{sign}[x]$ is the sign function. Thus, the temporal CG block maps $\mathbf{x}_{\text{TG}}(t_i)$ given by Eq. (27) onto $D(t_i)$ given by Eq. (34) via the internal recurrent mechanisms given by Eqs. (28) to (33).

We want to stress that, aside from the input and output interface of the NEAT algorithm given by Eq. (32) and (33), we have only specified that $C_x(t_i)$ and $G_x(t_i)$ represent recurrent signals which are related to the outputs $C_y(t_{i-1})$ and $G_y(t_{i-1})$ of the NEAT ANN at the previous time step $t_{i-1} = t_i - T_0$ via Eqs. (29) to (31):

The NEAT algorithm is used to identify the ANN, $\mathcal{N}$, which processes the input $\mathbf{x}_R(t)$ to the output $\mathbf{y}_R(t)$ such that the swimmer maximizes the mean swimming velocity $\bar{v}_D$ into the direction up the gradient of a chemical field $c_c(t)$ (see phase two in the main text and above). The complexity of the final, NEAT-evolved ANN depicted in main text Fig. 3E is, again, remarkably simple such that, in addition to the fixed recurrent topology defined by Eqs. (27) to (34), the recurrent control output can be written as

$$C_y(t_i) = f_{\text{clip}}\left(\mathcal{W}_C \cdot \frac{L_{\mathbf{T}}(t_i)}{L_0} + b_C\right), \quad [35]$$

and the gradient estimate $G_y(t_i)$ takes the form

$$y_{\text{I}}(t_i) = \mathcal{W}_{\text{I}} \cdot G_x(t_i) + b_{\text{I}}, \quad [36]$$
$$y_{\text{P}}(t_i) = \Pi\left(\mathcal{W}_{\text{P}} \otimes (G_x(t_i), y_{\text{I}}(t_i))\right) + b_{\text{P}}, \quad [37]$$
$$y_{\text{R}}(t_i) = f_{\text{relu}}(\mathcal{W}_{\text{R}} \otimes y_{\text{P}}(t_i) + b_{\text{R}}), \quad [38]$$
$$G_y(t_i) = f_{\text{clip}}(\Pi\left(\mathcal{W}_{\text{G}} \otimes (G_x(t_i), y_{\text{P}}(t_i), y_{\text{R}}(t_i), C_x(t_i))\right) + b_{\text{G}}), \quad [39]$$

with weights and biases given by

$$\mathcal{W}_{\text{C}} = (3.3001) \quad \text{and} \quad b_{\text{C}} = -7.3894, \quad [40]$$
$$\mathcal{W}_{\text{I}} = (-7.3303) \quad \text{and} \quad b_{\text{I}} = -3.5747, \quad [41]$$
$$\mathcal{W}_{\text{P}} = (9.9623, 3.7959), \quad \text{and} \quad b_{\text{P}} = 31.6729, \quad [42]$$
$$\mathcal{W}_{\text{R}} = (-0.9578) \quad \text{and} \quad b_{\text{R}} = 0.02371 \quad \text{and} \quad [43]$$
$$\mathcal{W}_{\text{G}} = (4.6648, 1.4068, -2.9839, -4.0215) \quad \text{and} \quad b_{\text{G}} = -7.3599. \quad [44]$$

$\Pi(\mathbf{x}) = x_1 \cdot x_2 \ldots x_N$ denotes aggregation by product of the components of a vector $\mathbf{x} = (x_1, \ldots, x_N)$ and $\mathbf{a} \otimes \mathbf{b} = (a_1 b_1, \ldots, a_N b_N)$ is an element-wise product of two vectors $\mathbf{a} = (a_1, \ldots, a_N)$ and $\mathbf{b} = (b_1, \ldots, b_N)$. The activation functions used in Eqs. (28) to (39) are given by $f_{\text{clip}}(x) = \max(\min(x, 1), -1)$, $f_{\pm}(x) = \text{sign}[x]$, and $f_{\text{relu}}(x) = \max(x, 0)$ and are referred to as linear clip-, sign- and relu-activation functions, respectively. Note, that in this contribution we clipped the output values $C_y(t_i)$ and $G_y(t_i)$ of the ANNs to an interval of $[-1, 1]$.

The entire temporal chemical gradient sensing block of the recurrent chemotaxis agent is defined through Eqs. (27) to (39) and can be subdivided into two self-contained sub-blocks: (i) the recurrent chemical memory control (CMC) cell introduced in phase two in the main text which is defined by Eqs. (29) to (31) and illustrated in Fig. S11A and (ii) the evolved NEAT ANN defined by Eqs. (35) to (39) (see main text Fig. 3E).

The task of the CMC cell is to control the entire process of recurrent feedback and memory updates of the chemical field within the microswimmer-agent. Per default, i.e. whenever $\alpha = 1$, the recurrent inputs of the current time step are the recurrent outputs from the previous ANN evaluation, i.e. $C_x(t_i) = C_y(t_{i-1})$, $G_x(t_i) = G_y(t_{i-1})$ and $M(t_i) = M(t_{i-1})$. As soon as the control aspect of the recurrent process is activated, i.e. when $\beta = 1$, the memory unit is updated $M(t_i) = c(x_c, t_i)/c_0 + \delta c$ and the recurrent input $C_x(t_i)$ of the current time step t_i is defined as the (delayed) chemical gradient $C_x(t_i) = c_c(t_i)/c_0 - M(t_{i-1})$. In other words, the CMC cell can act in two different modes, namely as a *bypass* that maintains the current state of its input, or as a *temporal chemical gradient estimate* with respect to previously measured chemical field values; both modes are schematically depicted in Fig. S11B,C.

The dynamical behavior of the input and output signals of the NEAT ANN for different chemical environments is depicted in Figs. S15 to S17. These figures highlight the two mechanisms the NEAT ANN unites: (i) The control output defined via Eq. (35) is sensitive to the total arm-length of the swimmer and is activated whenever the latter exceeds $L_{\text{T}}(t_i) \gtrsim 2.23 L_0$ (see $C_y(t_i)$ in Figs. S15 to S17). Given the periodic swimming gates of the SAL block in the chemotaxis agent (see main text Fig. 3A and $L_{\text{T}}(t_i)$ in Figs. S15 to S17), the control neuron is periodically activated and thus acts as a natural pacemaker for inducing measurements of the chemical field. (ii) The chemical gradient estimate defined via Eq. (39) is a highly non-linear function which correlates the previous values of the gradient estimate $G_x(t_i)$ (to the third power), the relu-activated $G_x(t_i)$ (squared) and the recurrent value of the control neuron, $C_x(t_i)$, in a clip-activated neuron.

Chemotaxis Learning Progress. We performed ten independent training runs, (i) for the spatial gradient estimator (see main text Fig. 3B,D) and (ii) for the temporal (recurrent) chemical gradient estimator (see main text Fig. 3C,E). For both cases we used two different initial conditions for the swimmer arm lengths, namely (i) random initial conditions, and (ii) equal arm-lengths $L_1(0) = L_2(0) = L_0$ at the beginning of each RL episode.

Typical results of the training progress (i.e., of the fitness versus NEAT generations), the corresponding NEAT ANN solutions and the chemotactic behavior of the respective spatial chemical gradient agents for selected chemical environments are presented in Fig. S9. For the given episode-setup of phase two RL, the randomization of the initial arm lengths during training helps the NEAT algorithm in identifying numerically more robust spatial sensing ANNs, such as given by Eq. (26), that are able to perform chemotaxis in gradients of steepness k of the chemical field which have not been used during training (Fig. S9C). The random initial conditions for the arm lengths result in a stochastic training progress since different initial conditions belong to slightly different optimum ANN solutions. Because of the simple ANN structure training basically converges after 30 NEAT generations.

In Fig. S12 we show typical training curves for the temporal chemical gradient estimators which have been performed at steepness $k_0 = 1\ c_0/R$ (left) and $k_0 = 0.1\ c_0/R$ (right), again, for both random initial arm lengths (top) and for equal initial arm lengths (bottom). In contrast to the training of spatial sensing ANNs the temporal gradient agents had to train significantly longer, i.e. hundreds or even thousands of RL steps. Interestingly, for $k_0 = 1\ c_0/R$, compared to training with

initial equal arm lengths, we found that training with initial random arm lengths typically approaches the optimum solution faster, leads to less biased solutions, and allows to perform chemotaxis at very small gradient steepness k, see Fig. S14A,C. Training at smaller $k_0 = 0.1\ c_0/R$ is, however, less stable under random arm length initial conditions, the optimum fitness fluctuates more strongly, and occasionally even decreases towards zero fitness, see Fig. S14B.

In Fig. S13 we show the corresponding ANN solutions obtained from the training runs shown in Fig. S12: While in B-D we show the final solution at generation 15000, in A (corresponding to the solution shown in main text Fig. 3E) we show a solution which already occurred at ~ 300 generations, but which has already optimum fitness despite its simple topology. Interestingly, the networks shown in A,C,D use the (periodically varying) total swimmer arm length L_T as a decision input for the control neuron C_y, while the solution shown in B does not make use at all of the internal locomotive state of the swimmer.

In Fig. S14 we have compared corresponding typical trajectories in a linear chemical field of different steepness k. It demonstrates the diversity of k values microswimmers which have learned in different training environments are able to perform chemotaxis.

In order to better understand the operation of the ANNs obtained for temporal sensing shown in the main text and in Fig. S13A, we have shown the interplay between swimmer trajectory, input signals, chemical memory, gradient estimators and control neurons in Figs. S15 to S17. It demonstrates the decisions a microswimmer makes about keeping or changing the direction based on the temporal measurement of the chemical field. Details are given in the figure captions.

Effect of Noise and Gradient Steepness. The chemotactic response of the temporally sensing microswimmer discussed in the main text was trained in chemical gradients of slope $k/c_0R^{-1} \in [0.99, 1.01]$. As shown above, we found that after training the swimmer is able to perform chemotaxis in much weaker gradients, and, as shown in the main text, to perform biased run-and tumble motion in the presence of noise. In Fig. S20 we show trajectories of ensembles of 100 non-interacting microswimmers moving in constant gradients $c(x) = kx$ for different noise strengths ξ and for a broad range of gradient steepness k. Depending on the noise strength, we see either ballistic behavior moving up the gradient, run-and-reverse motion up the gradient, or run-and-reverse motion in the direction against the gradient due to the negative bias visible for this microswimmer at small k or sufficiently large ξ values. These trajectories form the basis for Fig. 4D shown in the main text (see discussion in main text) which show the mean chemotactic drift velocity v_c, defined as the ensemble and time average of the individual microswimmer velocities, depending on k and ξ .

In Fig. S21 we show the chemotactic behavior of microswimmers which have learned in four different environments ($k_0 = c_0/R$ and $k_0 = 0.1c_0/R$, each with random/equal initial arm lengths during training, see e.g. respective ANN solutions in Fig. S13). The left column shows the chemotactic velocity compared to the MC-SAL fitness, $v_c/\bar{v}_{MC}$, depending on the noise strength ξ for different values of the gradient steepness k. In contrast, the center columns show $v_c/\bar{v}_{MC}$ depending on k for different ξ. The right column shows the extracted run times to the right, τ_R, and to the left, τ_L (see also main text). We can see that the microswimmers which use the swimmer length L_T as decision-making input for the measurement of the chemical field through the control neuron C_y are able to perform chemotaxis for a broad range of gradient steepness and noise levels, while the network which does not use the internal state of the swimmer fails to perform chemotaxis in gradient steepness not used during training.

Movie S1. Evolution of the artificial neural networks in phase one shows the time evolution of the fittest network per generation, quantified by its fitness $\bar{v}$. The connections between neurons are color-coded in terms of the strength of the weights. Color-coding of the neurons corresponds to their bias. In addition the (L_1, L_2) and (F_1, F_3) phase space curve of the network output in the absence of a chemical field are shown. The transition from non-swimming to swimming solutions is indicated by the emergence of periodic solutions in these phase space plots.

Movie S2. Dynamics of a microswimmer in the absence of a chemical field (corresponding to blue trajectory shown in Fig. 2A of the main text). Top left: The x axis shows the positions x_i of the three beads (gray trajectories and red circles (not to scale)) and of the center of mass x_c (black trajectories and gray diamond). The thickness of the arms (blue) represents the strength of the forces applied to the outer beads (thin: extraction; thick: compression). Top right: Dynamics of the arm lengths in (L_1, L_2) phase space (blue moving point). The blue line shows the steady state periodic stroke pattern leading to motion in positive x direction. Bottom right: Dynamics of the forces in (F_1, F_3) phase space. Color code same as for arm lengths. Bottom left: Dynamics of the instantaneous policy of the artificial neural network which proposes the forces on the beads. Bottom center: Sketch of the corresponding network topology.

Movie S3. Evolution of the artificial neural networks in phase two for spatial sensing showing the time evolution of the fittest network per generation, quantified by its fitness $\bar{v}_D$, and which leads to the solution shown in Fig. 3D in the main text. The connections between neurons are color-coded in terms of the strength of the weights. Color-coding of the neurons corresponds to their bias.

Movie S4. Evolution of the artificial neural networks in phase two for temporal sensing showing the time evolution of the fittest network per generation, quantified by its fitness $\bar{v}_D$, and which leads to the solution shown in Fig. 3E in the main text. The connections between neurons are color-coded in terms of the strength of the weights. Color-coding of the neurons corresponds to their bias.

Movie S5. Dynamics of a temporally sensing microswimmer in a piecewise linear chemical field (corresponding to the blue/red trajectory shown in Fig. 2B of the main text). Left: The x axis shows the positions x_i of the three beads (gray trajectories and red circles (not to scale)) and of the center of mass x_c (black trajectories and gray diamond). The thickness of the arms (blue) represents the strength of the forces applied to the outer beads (thin: extraction; thick: compression). Top right: Dynamics of the arm lengths in (L_1, L_2) phase space (blue/red moving point). The blue line shows the steady state periodic stroke pattern leading to motion in positive x direction. The red line shows the symmetry-transformed stroke pattern (motion in negative x direction). Bottom right: Dynamics of the forces in (F_1, F_3) phase space. Color code same as for arm lengths.

Movie S6. Dynamics of a temporally sensing microswimmer in a Gaussian chemical field (corresponding to the blue/red trajectory shown in Fig. 2C of the main text). Left: The x axis shows the positions x_i of the three beads (gray trajectories and red circles (not to scale)) and of the center of mass x_c (black trajectories and gray diamond). The thickness of the arms (blue) represents the strength of the forces applied to the outer beads (thin: extraction; thick: compression). Top right: Dynamics of the arm lengths in (L_1, L_2) phase space (blue/red moving point). The blue line shows the steady state periodic stroke pattern leading to motion in positive x direction. The red line shows the symmetry-transformed stroke pattern (motion in negative x direction). Bottom right: Dynamics of the forces in (F_1, F_3) phase space. Color code same as for arm lengths.

Movie S7. Dynamics of a spatially sensing microswimmer in a Gaussian chemical field (corresponding to the black trajectory shown in Fig. 2C of the main text). Left: The x axis shows the positions x_i of the three beads (gray trajectories and red circles (not to scale)) and of the center of mass x_c (black trajectories and gray diamond). The thickness of the arms (blue) represents the strength of the forces applied to the outer beads (thin: extraction; thick: compression). Top right: Dynamics of the arm lengths in (L_1, L_2) phase space (blue/red moving point). The blue line shows the steady state periodic stroke pattern leading to motion in positive x direction. The red line shows the symmetry-transformed stroke pattern (motion in negative x direction). Bottom right: Dynamics of the forces in (F_1, F_3) phase space. Color code same as for arm lengths.

Movie S8. Dynamics of a temporally sensing microswimmer subjected to intrinsic noise of the chemical memory readings using the parameters shown in Fig. 4A (green trajectory) of the main text. Left: The x axis shows the positions x_i of the three beads (gray trajectories and red circles (not to scale)) and of the center of mass x_c (black trajectories and gray diamond). The thickness of the arms (blue) represents the strength of the forces applied to the outer beads (thin: extraction; thick: compression). Right: Time evolution of the run time statistics: Blue: Run times in positive x direction; red: run times in negative x direction.

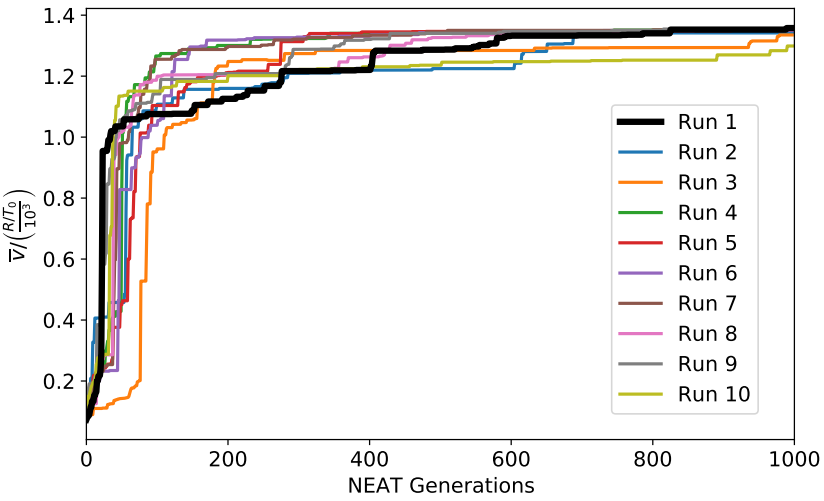

Fig. S1. Maximum fitness curves obtained for an ensemble of $N = 200$ ANNs obtained for 10 independent NEAT training runs at phase one.

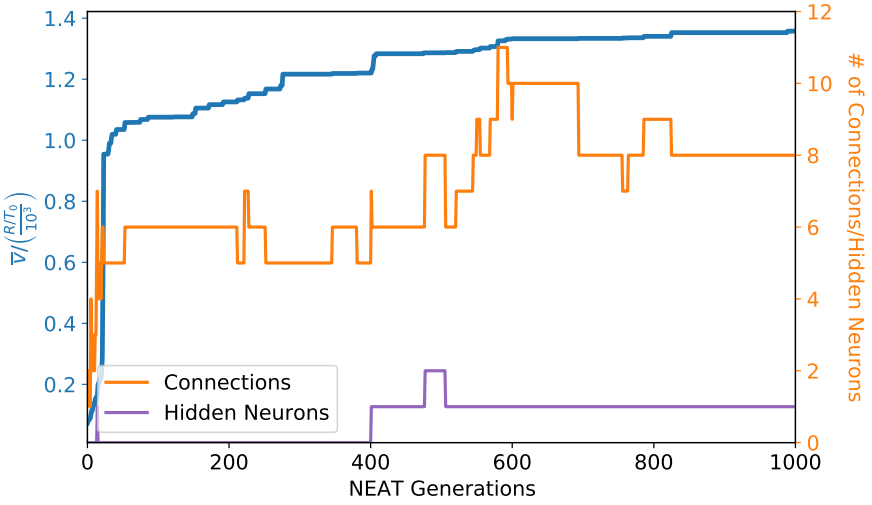

Fig. S2. Maximum fitness, number of network connections, and number of hidden neurons for Training Run 1 (see Fig. S1) which leads to the most fittest ANN solution (O-SAL-2) of phase one training consisting of 8 connections and one hidden neuron (see also Fig. 1D and Fig. 2A in main text, and Movie S1).

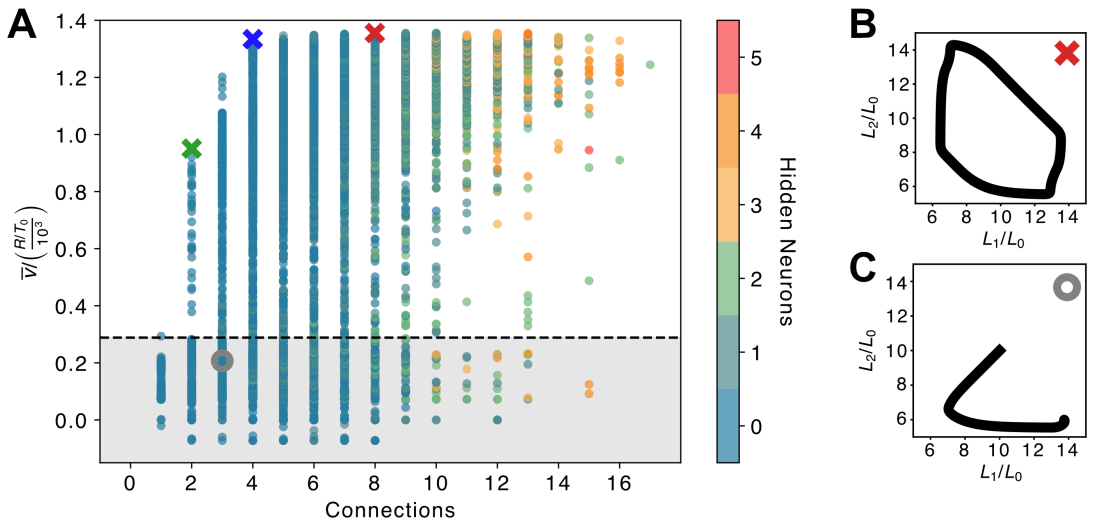

Fig. S3. (A): Scatter Plot of obtained microswimmer fitness values $\bar{v}$ depending on the number of network connections and hidden neurons. The black dashed line approximately divides good swimming solutions (periodic motion in (L_1, L_2) phase space) (above), and non-swimming solutions where transient arm motion stops at a certain position in (L_1, L_2) phase space (below). Note, bad swimming solutions also have fitness values below the black dashed line. (B) Sketch of swimming solution in (L_1, L_2) phase space. (C) Sketch of non-swimming solution in (L_1, L_2) phase space. See also figure S7 for discussion on bifurcation between swimming and non-swimming solutions.

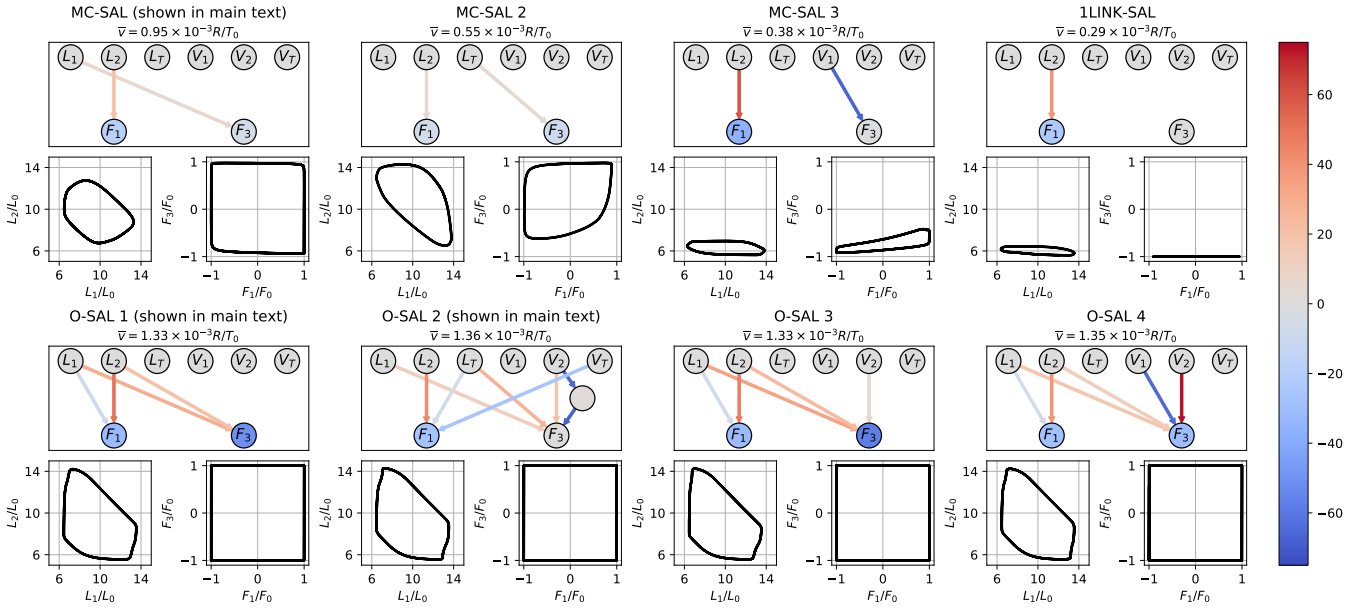

Fig. S4. Different minimal complexity (MC-SAL) and optimum (O-SAL) solutions for the swimmer action layer (SAL) network and dynamics in (L_1, L_2) and (F_1, F_3) phase space. The values for the weights and biases are color-coded in the connections, and neurons, respectively see color bar. Top row: 1st column: MC-SAL solution as presented in the main text; 2nd column: MC-SAL-2 solution connecting L_T with F_3 but of smaller fitness compared to MC-SAL presented in main text; 3rd column: MC-SAL-3 solution using the arm velocity V_1 to determine the force F_3; 4th column: one-link solution which leads to swimming (yet inefficient) solution due to the exploitation of elastic restoring forces. Bottom row: Examples of O-SAL solutions all of similar fitness and dynamics in (L_1, L_2) and (F_1, F_3) phase space demonstrating the robustness of the optimum solutions. 1st and 2nd column: O-SAL solutions presented in the main text; 3rd column: O-SAL-3 solution consisting of five connections (same connections and similar weights as O-SAL-1, but one connection added); 4th column: O-SAL-4 solution consisting of six connections (same connections and similar weights as O-SAL-1, but two connections added).

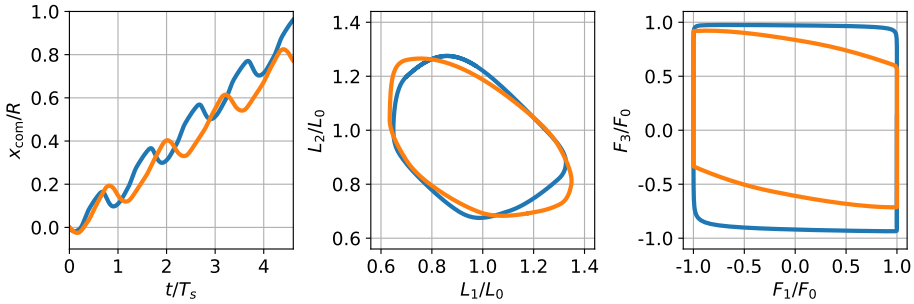

Fig. S5. Comparison of MC-SAL solutions presented in the main text using different activation functions but same topology and weights. Blue: MC-SAL (see Fig. S4) with standard $\tanh$ activation corresponding to a fitness $\bar{v} = 0.95 \times 10^{-3} R/T_0$. Orange: MC-SAL with its activation replaced by a clip function $f_{\text{clip}}(x) = \max(\min(x, 1), -1)$ leading to a fitness of $\bar{v} = 0.77 \times 10^{-3} R/T_0$.

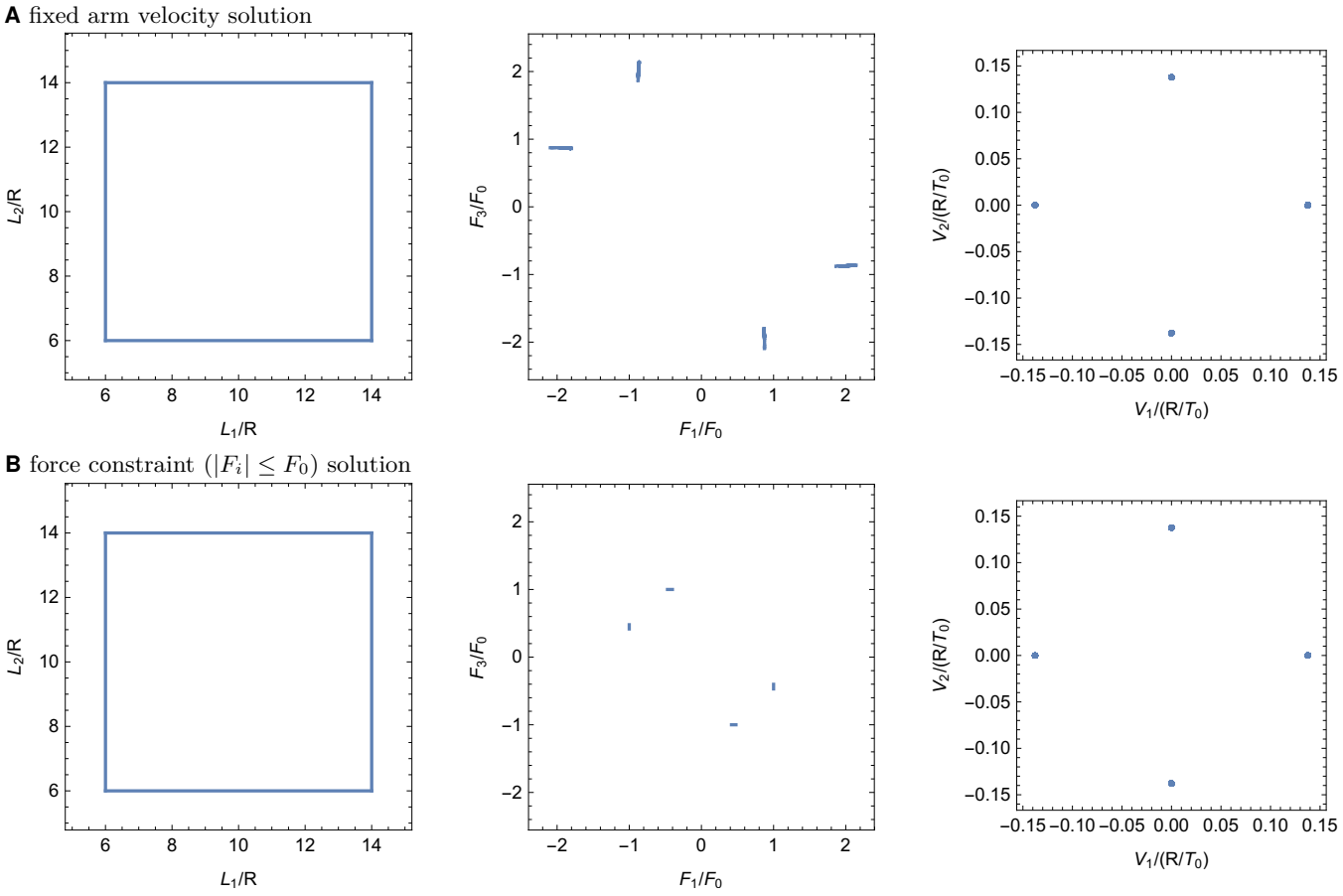

Fig. S6. 4-state solutions of the three bead swimmer. Left: 4-state dynamics in (L_1,L_2) state space. Center: Corresponding dynamics of the forces (F_1,F_3). Right: Corresponding arm velocities (V_1,V_2). (A) 4-state swimming solution where the velocity of the arms have been fixed to $|V_{1,2}| = 0.138R/T_0$. (B) 4-state swimming solution where the forces on the beads have been limited to $|F_{1,3}| \leq F_0$.

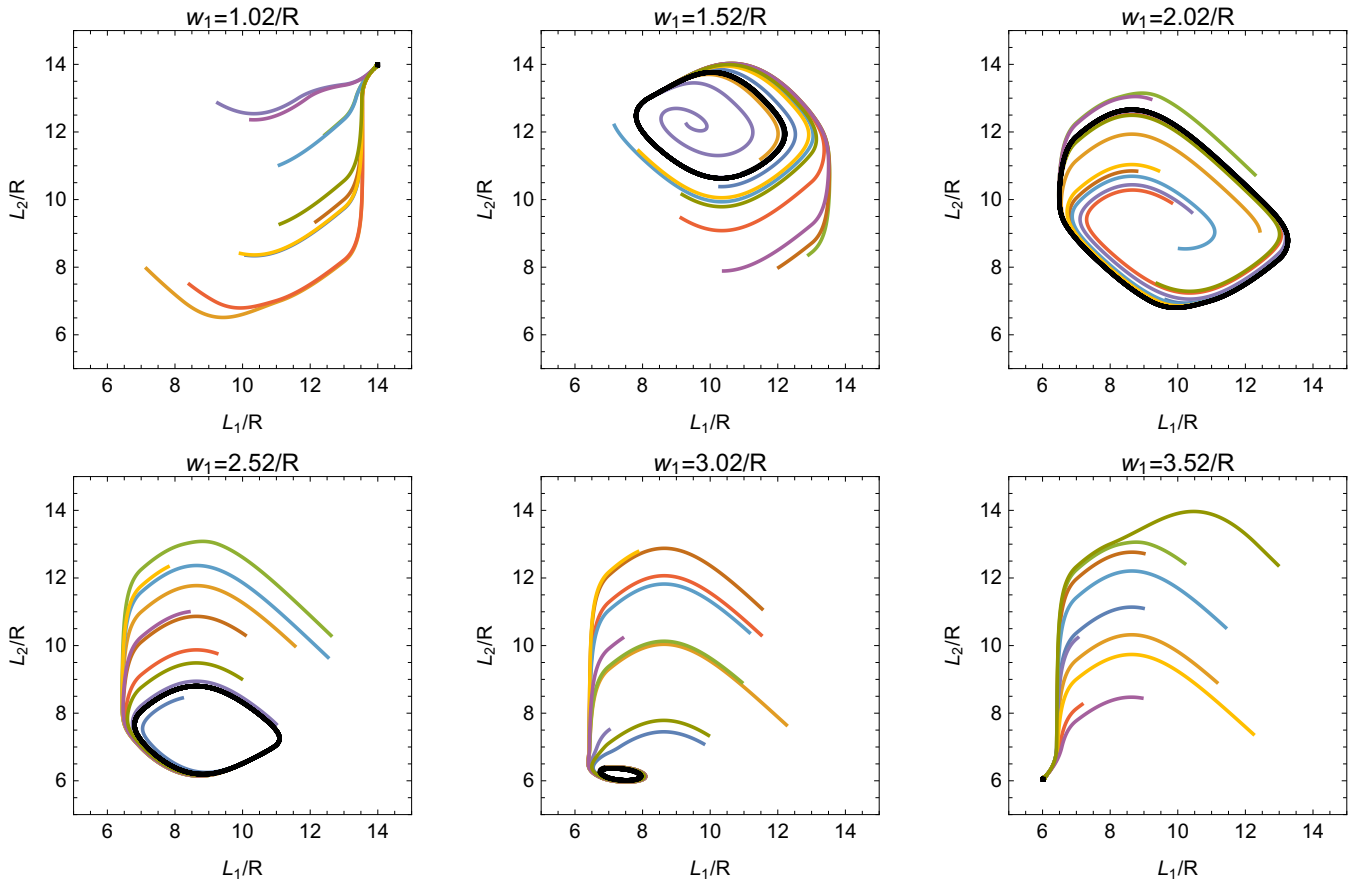

Fig. S7. Solutions for the arm length trajectories $(L_1(t), L_2(t))$ from Eqs. (13) to (16) for different initial conditions (colored curves) for the MC-SAL topology presented in the main text. The weights have been chosen as for the MC-SAL solution presented in the main text, except for w_1 which has been used as a control parameter. By tuning w_1 a bifurcation from a fixed point to a limit cycle, and a second bifurcation to another fixed point occurs. The MC-SAL solution presented in the main text is captured by $w_1 = 2.02/R$.

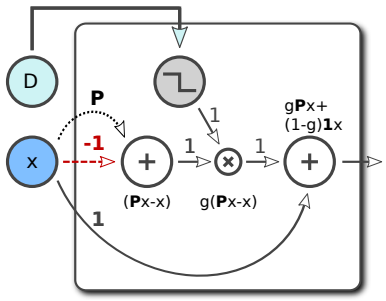

Fig. S8. Permutation control layer (PCL) which represent a building block of the proposed chemotaxis agent, see main text Fig. 3A. The PCL performs a permutation transformation, $\mathbf{X}_D = g(D)\,\mathcal{P}\mathbf{X} + (1 - g(D))\,\mathbf{X}$, on an arbitrary input $\mathbf{X}$, depending on a control input D via $g(D) = f_{\Theta\_}(D) = \Theta(-D)$. In the case of a chemotaxis agent (see main text Fig. 3A) this transformation is applied to both, the SAL input via Eq. (18) and the SAL output via Eq. (19).

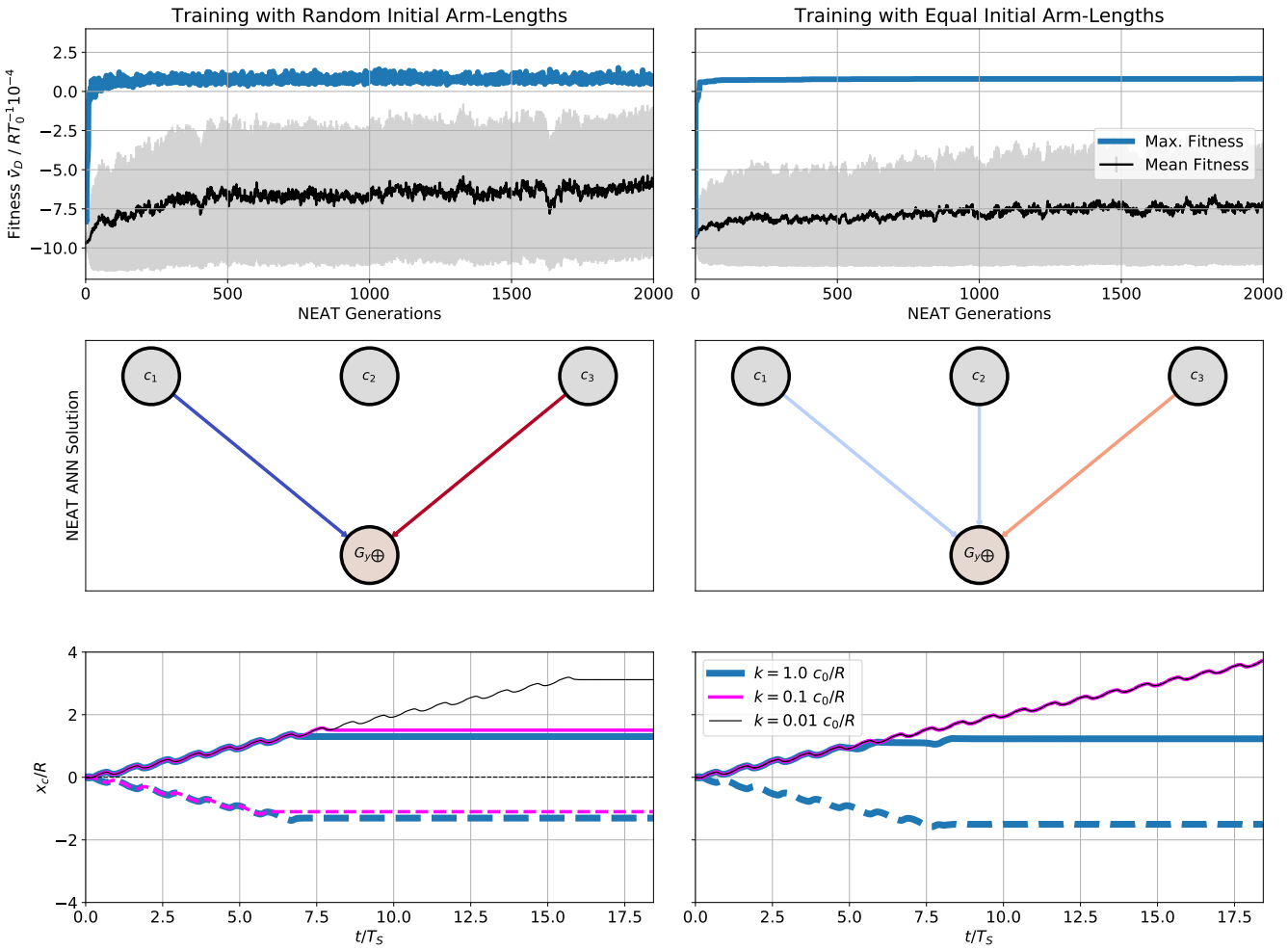

Fig. S9. Top panel: Typical NEAT learning progress for the spatial chemical gradient sensing microswimmer-agent left for training with randomized arm-lengths at the beginning of each RL episode and right with equal initial arm-lengths $L_1(0) = L_2(0) = L_0$. The final ANN (central left) is given by Eqs.(23) and (24), see main text Fig. 3D and Movie S3. The mean fitness value (black), the maximum fitness value (blue) and the standard deviation of all fitness values (gray vertical lines) are presented for a total number of 288 ANNs per NEAT generation for consecutive NEAT generations. The spatial gradient sensing agent was trained using the fitness function related to phase two in the main text (and specified in the text above). The training was employed on the Vienna Scientific Cluster (VSC-4) (12) and is practically converged after 30 NEAT generations for both cases (left and right) which corresponds to roughly 30 CPU hours (0.6 hours on a 48 core node for a total number of 288 ANNs per NEAT generation). Central panel: Final NEAT ANNs of the training procedures with (left) and without (right) randomized initial arm lengths at the beginning of each episode. The colors of the connections and the output neurons emphasize the numerical value of the respective weights and biases (blue: negative, red: positive) which evaluate to $\mathcal{W}_{\rm SG} = (-71.147, 0, 71.16)$ and $b_{\rm SG} = 1.373$ (left), and $\mathcal{W}_{\rm SG} = (-15.092, -13.331, 29.311)$ and $b_{\rm SG} = 1.092$ (right). Bottom panel: Center of mass $x_c(t)$ dynamics of the spatial CG agents, equipped with the respective NEAT ANN (as shown in the central left and central right panels), in linear fields of the form $c(x) = \max(0, a - k|x - x_0|)$ (see Fig. 2B in the main-text) with three different values of the slope $k/(c_0 R) \in \{1, 0.1, 0.01\}$ (blue, magenta, black) and with peak positions at $x_0 = +2R$ (solid lines) and $x_0 = -2R$ (dashed lines) for $a = 100\ c_0$.

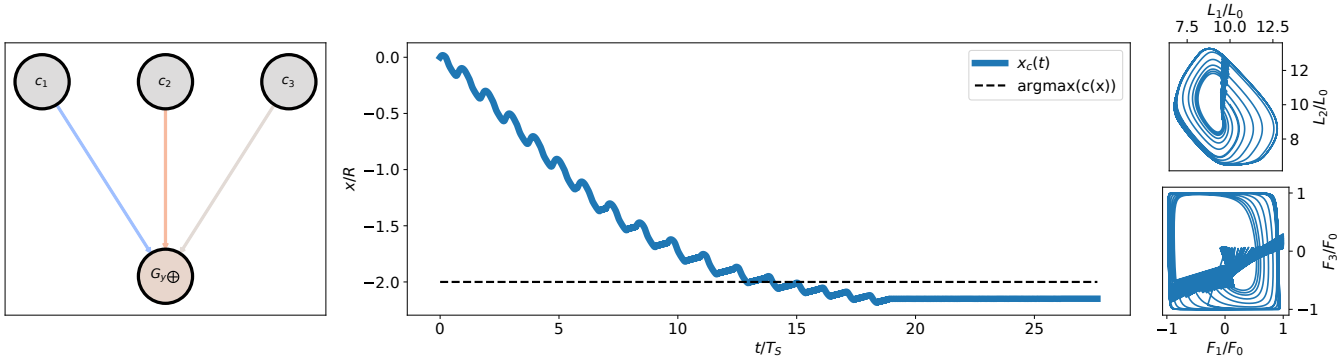

Fig. S10. Non-optimum solution obtained for spatial gradient detection which was trained with a 10-fold increased slope-noise $10 \times \delta k$ and equal initial arm-lengths $L_1(0) = L_2(0) = L_0$. The weight matrix and bias of the NEAT ANN (left panel) evaluate to $\mathcal{W}_{\mathrm{SG}} = (-23.694, 19.184, 2.456)$ and $b_{\mathrm{SG}} = 1.658$ (see Eqs.(23) and (24)). The solution shows that a microswimmer is able to learn a policy where it slows down (see trajectory x_c and phase space curves) when approaching the peak of a chemical field (here a piece-wise linear field of the form $c(x) = \max(0, a - k|x - x_0|)$ with slope $k = 1\ c_0/R$ centered at $x_0 = -2\ R$ and an amplitude of $a = 100\ c_0$), despite the fact that training was divided to phase one and phase two using MC-SAL (see main text).

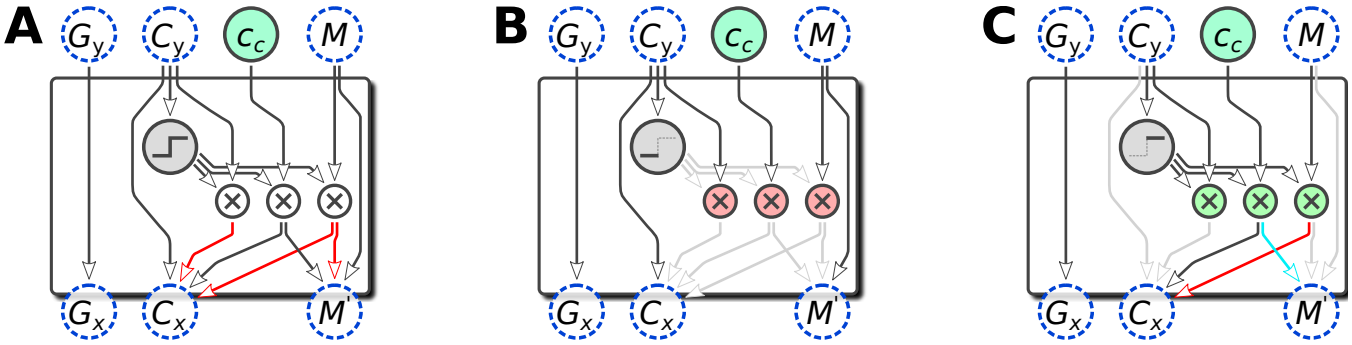

Fig. S11. (A) Sketch of the chemical memory control (CMC) cell which is used in a temporal chemical gradient block (see main text Fig. 3C) to estimate the gradient of a chemical field $c_c(t_i) = c(x_c(t_i), t_i)$ at the center of mass position $x_c(t_i)$ of a microswimmer over time. The functionality of the CMC cell can be described by Eqs. (29) to (31) (see also equations in phase two in the main text): a recurrent control neuron (labeled C_y) is used to control the value of a memory neuron (labeled M) via a perceptron activated neuron (gray neuron) following Eqs. (29) and (31); the recurrent gradient estimate $G_x(t_i) = G_y(t_{i-1})$ is directly forwarded as defined via Eq. (30), independent of the value of C_y. The flow of information between the neurons in the CMC cell is indicated via arrows. Black (red) arrows mark positive (negative) unit weights 1 (-1) and $\bigotimes$ symbols represent simple multiplication neurons (without any bias). (B) Sketch of the *bypass mode* of the CMC cell induced by $C_y(t_{i-1}) < 0$: per default, the value of neuron C_y is forwarded as $C_x(t_i) = C_y(t_{i-1})$ to the NEAT-ANN of the chemical gradient block (see main text Fig. 3C) due to the inactive gray perceptron (the three multiplier neurons block all incoming signals and are thus colored in red). Simultaneously, the internal chemical memory $M(t_i) = M(t_{i-1})$ is preserved over time. Active (inactive) connections are colored black (light-gray). (C) Sketch of the *chemical gradient mode* of the CMC cell induced by $C_y(t_{i-1}) > 0$: due to the active gray perceptron, the multiplier neurons (now colored in green) forward their respective input values. Now, the value of the chemical field c_c is stored into the chemical memory, $M(t_i) = c_c(t_i)/c_0 + \delta c/c_0$ and the value of C_x is updated according to $C_x(t_i) = c_c(t_i)/c_0 - M(t_{i-1})$ thereby quantifying the chemical gradient between two delayed measurements. The field-memory reading may be influenced by noise as indicated by the (Gaussian distributed) random variable δc which is represented by the cyan colored noisy input channel of M. The gray lines indicate signals which cancel each other out to zero.

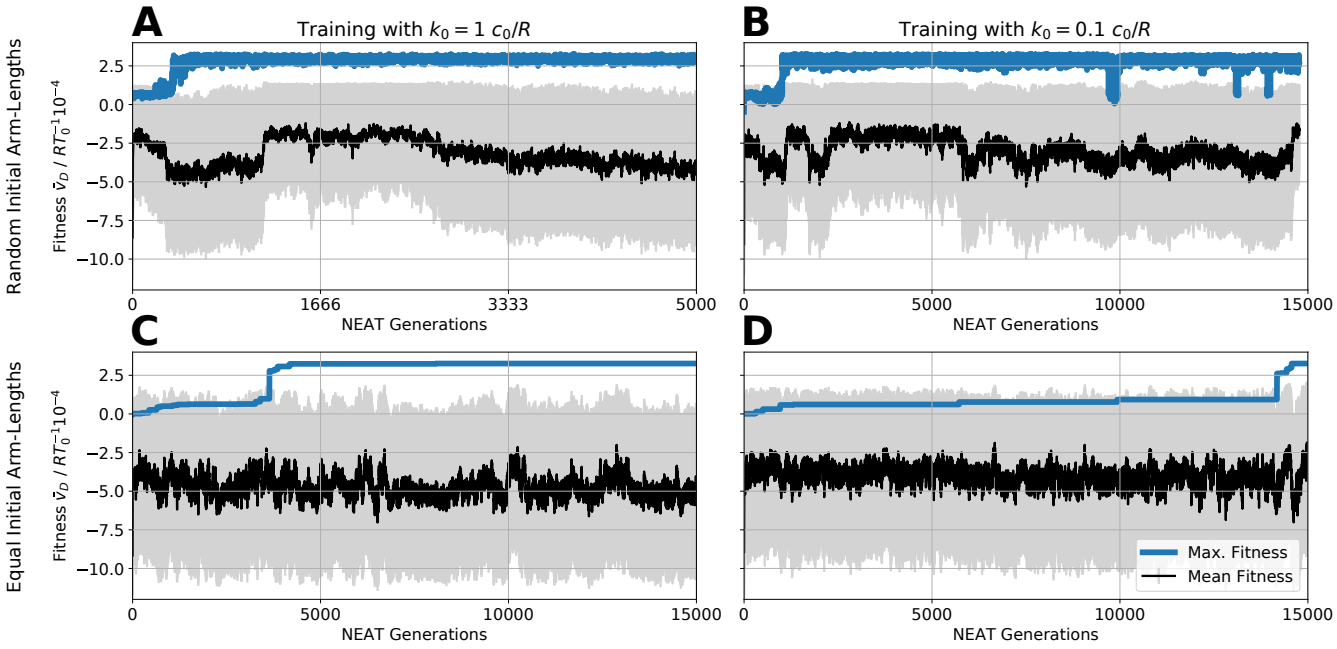

Fig. S12. A: Typical NEAT learning progress for the temporal chemical gradient sensing microswimmer-agent in a chemical field of slope $k_0 = 1\ c_0/R$ with randomized initial arm-lengths at the beginning of each RL episode, giving rise to the NEAT ANN defined by Eqs.(32) to (39) which is used for the temporal chemotaxis analysis in this contribution (see main text Fig. 3C, Fig. S13A and Movie S4). The mean fitness value (black), the maximum fitness value (blue) and the standard deviation of all fitness values (gray vertical lines) are presented for a total number of 288 ANNs per NEAT generation for consecutive NEAT generations. The temporal gradient sensing agent is trained using the fitness function $\bar{v}_D$ related to phase two in the main text (and specified in the text above). B: same as panel A but for training with $k_0 = 0.1\ c_0/R$. C (D): same as panels A (B) but for equal initial arm lengths $L_1(0) = L_2(0) = L_0$ at the beginning of each RL episode. As compared to the spatial gradient sensing agent (see Fig. S9) the training of the temporal gradient sensing agents takes orders of magnitude longer to converge: For the training depicted in panel A after 350 NEAT generations a plateau of the fitness is reached featuring the ANN depicted in the main text Fig. 3E. For the training depicted in panel B roughly 1000 NEAT generations are necessary to reach comparable results. For the training depicted in panel C reasonable ANN solutions emerge after roughly 4000 NEAT generations and for the training depicted in panel D almost 15000 generations are explored by the NEAT genetic algorithm to identify reasonable temporal sensing ANNs. The training runs are all performed with 288 ANNs per NEAT generation. The training shown in the top left panel are performed over ~ 14 hours on a 48 core node on the Vienna Scientific Cluster (VSC-4) (12) which amounts to 672 CPU hours. The other training runs are performed over ~ 40 hours each on the same hardware, i.e., for an equivalent of 1920 CPU hours. ANN results obtained from the four training runs are depicted in Fig. S13 and corresponding applied chemotaxis of the four agents in different chemical environments are shown in Fig. S14.

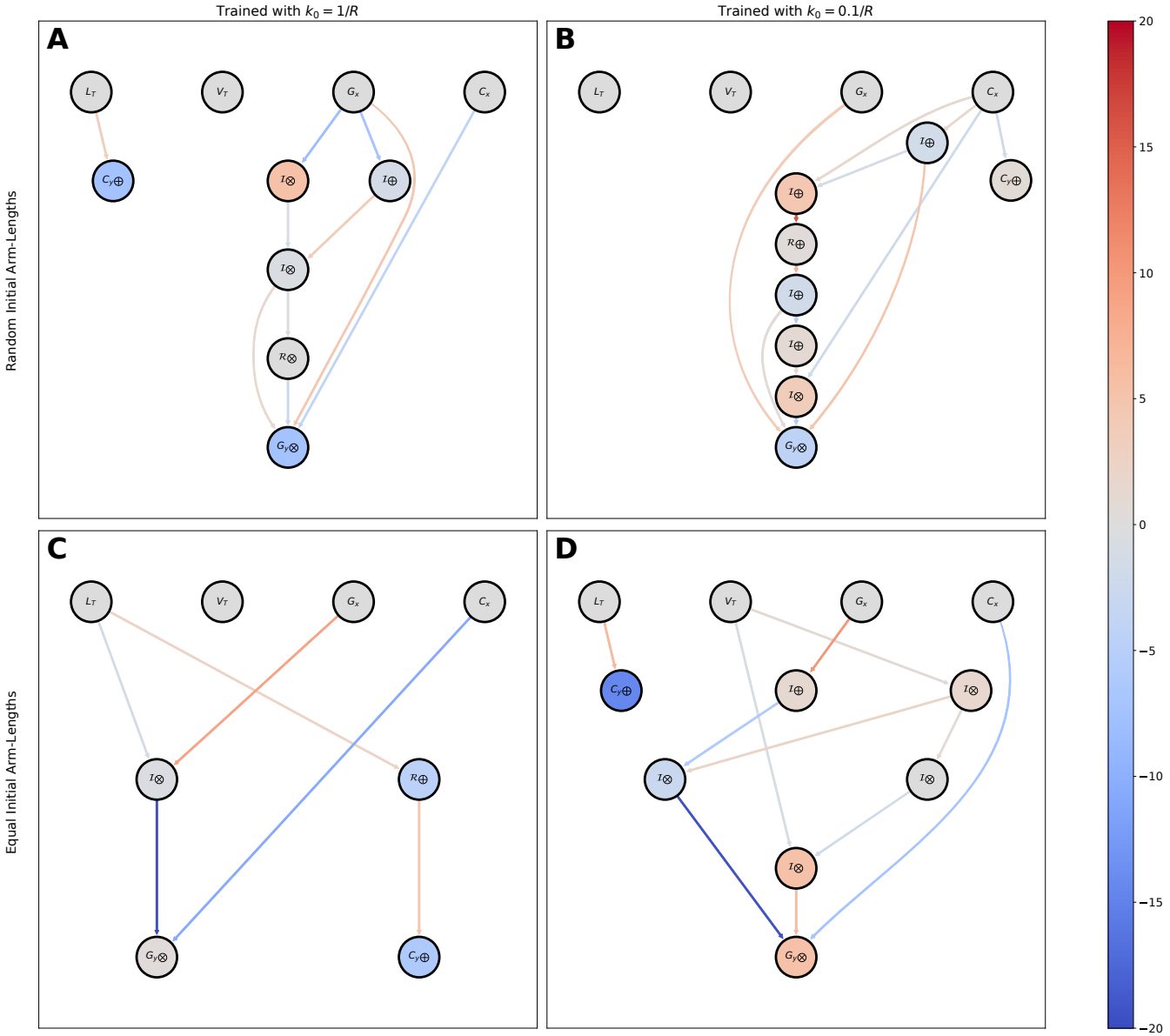

Fig. S13. Comparison of different temporal CG NEAT ANN solutions corresponding to the training runs depicted in Fig. S12 (same respective panel arrangement). Linear (identity) or relu activation functions of the hidden neurons are indicated as $\mathcal{I}$ or $\mathcal{R}$ with indices $\oplus$ or $\otimes$ labelling whether accumulation by summation or by product is used in the respective neurons. The activation of the output units C_y and G_y is fixed as to $f_{\mathrm{clip}}(x)$, normalizing the output of the ANNs to values of $[-1, 1]$. Weights and biases are emphasized via red (positive) and blue (negative) coloring according to the colorbar. A: NEAT evolved ANN, equivalent to the solution shown in main text Fig. 3E and described via Eqs. (35) to (39), which is mainly used in this contribution for temporal chemotaxis analysis. Numerical details on the weights and biases of the ANN are given by Eqs. (35) to (39) and the corresponding output signals of each neuron are explicitly labeled in Fig. 3E in the main text. Notably, the red colored $\mathcal{I}_\otimes$ identity neuron with product aggregation is removed in the ANN shown in Fig. 3E in the main text to reduce the complexity of the ANN and the corresponding weights and biases are absorbed into b_{I}, b_{P} and $\mathcal{W}_{\mathrm{P}}$ defined in Eqs. (36) and (37). B: Temporal sensing NEAT ANN which does not require L_T (or V_T) as input for the control neuron C_y but which can detect chemical gradients solely via recurrent processes (relying on the interface with the CMC cell, see Eqs. (29) to (30)). Here, $C_y(t)$ is only sensitive to its recurrent state $C_x(t) = C_y(t_{i-1})$ whenever $C_y(t_{i-1}) < 0$ or to the measured temporal chemical gradient $C_x(t_i) = c_c(t_i)/c_0 - M(t_{i-1})$ (provided by the CMC cell) for $C_y(t_{i-1}) > 0$. $G_y(t)$ is, again, a highly non-linear function of the inputs $G_x(t)$ and $C_x(t)$. The ANN can be used in a temporal chemotaxis agent to efficiently perform chemotaxis (see Fig. S14B) but has shortcomings in chemical environments with slopes that are different to $k \approx 0.1c_0/R$ (which was used during training, see Fig. S21B). C: Simple ANN solution for temporal gradient sensing which only requires two hidden neurons. The ANN can cope with a wide range of slopes k of a chemical field when used in a temporal chemotaxis agent (see Fig. S14C and Fig. S21C). Compared to the ANNs presented in panels A, B and D, the ANN depicted in panel C exhibits a relatively large bias towards the right for small values of k. D: Slightly more complex but numerically more robust ANN for temporal gradient sensing compared to the panel C ANN. A corresponding temporal CG agent can cope with a large range of k values compared to $k_0 = 0.1c_0/R$ used during training but is slightly more biased (to the right) as compared to the ANN shown in panel A, especially for small values of k (see Fig. S14D and Fig. S21D).

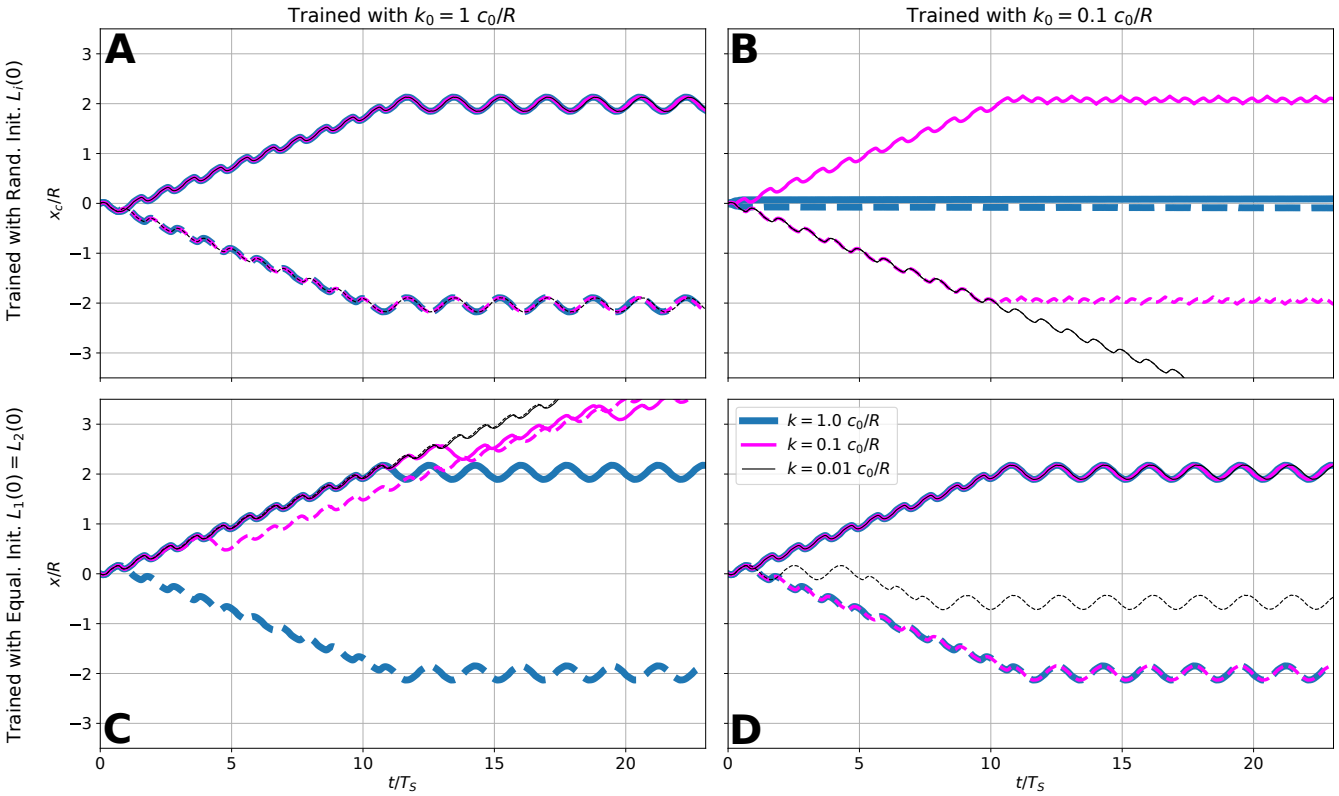

Fig. S14. Comparison of center of mass dynamics $x_c(t)$ of different temporal CG NEAT ANN solutions corresponding to the training runs depicted in Fig. S12 (same respective panel arrangement). The NEAT ANNs used in the temporal chemotaxis agents (see Fig. 3C) to evaluate the different trajectories in panels A to D are depicted Fig. S13. Similar to the bottom panels of Fig. S9, the four different temporal chemotaxis agents A to D are each applied to six different piece-wise linear chemical fields $c(x) = \max(0, a - k|x - x_0|)$, with slopes $k/(c_0 R) = \{1, 0.1, 0.01\}$ (blue, magenta, black), peak centers at either $x_0 = 2\ R$ (solid lines) or $x_0 = -2\ R$ (dashed lines) and a peak amplitude of $a = 100\ c_0$. A: The presented chemotaxis agent quickly approaches the peak maxima at $x_0 = \pm 2\ R$ for all variants of the chemical field, then oscillates around the peaks. B: The presented chemotaxis agent quickly approaches the peak maxima at $x_0 = \pm 2\ R$ for $k = 0.1\ c_0/R$ (for which it was trained) and performs narrow oscillations around the maxima. For all other investigated values of the slopes, $k = 1.0\ c_0/R$ and $k = 0.01\ c_0/R$, the chemotaxis strategy of the swimmer fails. C: The presented chemotaxis agent quickly approaches the peak maxima at $x_0 = \pm 2\ R$ for $k = 1\ c_0/R$ (for which it was trained) and performs oscillations around the maxima. For smaller slopes values of $k < 1.0\ c_0/R$, the chemotaxis strategy of the swimmer fails (see also Fig. S21). D: The presented chemotaxis agent quickly approaches the peak maxima of the chemical field at $x_0 = +2\ R$ for all shown slope scenarios and the peak at $x_0 = -2\ R$ for $k = 1\ c_0/R$ and $k = 0.1\ c_0/R$, then oscillates around the peaks. For $k = 0.01\ c_0/R$ the swimmer does not identify the peak maximum at $x_0 = -2\ R$ correctly (black dashed line) but oscillates for $t > 10 T_S$ around $x \approx 0.5\ R$, revealing a slight bias to the right for small k values.

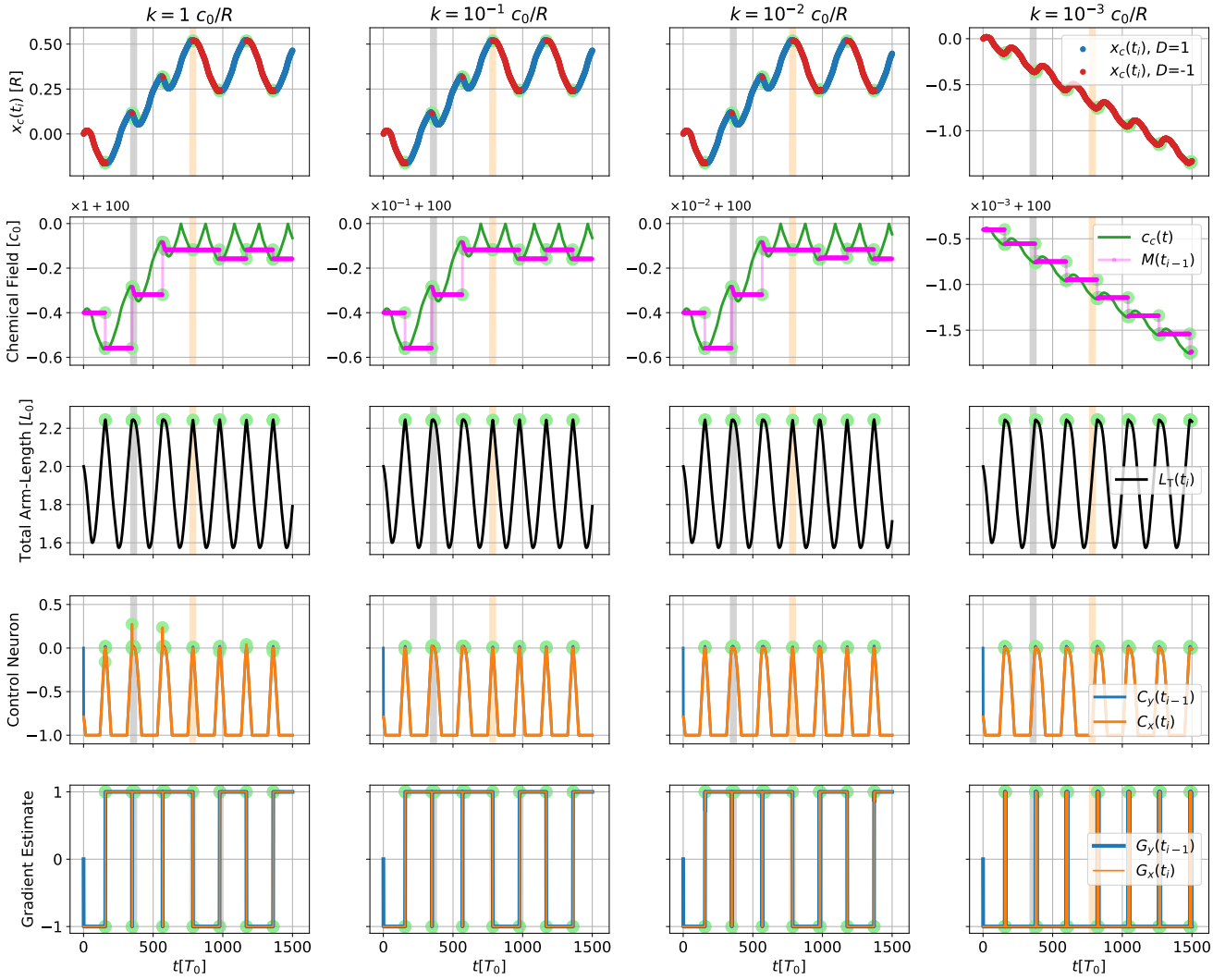

Fig. S15. Analysis of decision-making strategy of the temporal chemotaxis agent introduced in the main text (see main text Fig. 3C,E and in Fig. S13A) in a piece-wise linear field of the form $c(x) = \max(0, a - k|x - x_0|)$ (see Fig. 2B in the main-text) with four different values of the slope $k = \{1, 10^{-1}, 10^{-2}, 10^{-3}\}\ c_0/R$ (column-wise arranged from left to right) centered at $x_0 = 0.4\ R$ with an amplitude of $a = 100\ c_0$. The panels from top to bottom show for the respective (column-wise) chemical environments (i) the time evolution of the swimmer's COM position $x_c(t_i)$, (ii) the value of the chemical field $c_c(t_i)$ at the COM position of the microswimmer (note the different slope-scales in the second row) and the associated chemical memory $M(t_{i-1})$, (iii) the total arm-length $L_T(t_i) = L_1(t_i) + L_2(t_i)$, (iv) the value of the control neuron $C_y(t_{i-1})$ ($C_x(t_i)$) prior (posterior) to the CMC-cell, see Eq. (29), and (v) the gradient estimate $G_y(t_{i-1})$ ($G_x(t_i)$) prior (posterior) to the CMC-cell, see Eq. (30); the blue (red) color-coding in the top panels indicates the current intentional rightward (leftward) motion, i.e. $D = 1$ ($D = -1$), for the respective chemical environments. Short measurement intervals are induced periodically after approximately one stroke period $T_S = 217\ T_0$ of the swimmer whenever the total arm-length $L_T/L_0 \gtrsim 2.23$ as can be seen in the third row (see main-text Fig. 3E and Eq. (35)); measurement time-instances t_β when $\beta(C_y(t_{i-1})) = \Theta(C_y(t_{i-1})) = 1$ or, equivalently, $C_y(t_{i-1}) > 0$ (see Fig. S11C) are indicated by large green circles in all panels, respectively. The chemotaxis agent can predict the chemical gradient in slopes of $k = 1$ to $10^{-2}\ c_0/R$ (left three columns) but fails for slope values of $k = 10^{-3}\ c_0/R$ and smaller (right column). During the measurement intervals the agent either decides whether to continue swimming into a certain direction (e.g. at $t_\beta = 348$ to $366\ T_0$ for $k = 1$ to $10^{-2}\ c_0/R$ emphasized by the gray vertical region and illustrated in Fig. S16) or to reverse its direction (e.g. at $t_\beta = 785$ to $786\ T_0$ for for $k = 1$ to $10^{-2}\ c_0/R$ emphasized by the orange vertical region and illustrated in Fig. S17); for completeness, the corresponding data for $k = 10^{-3}\ c_0/R$ is also shown in Figs. S16 and S17 in the respective time-frames.

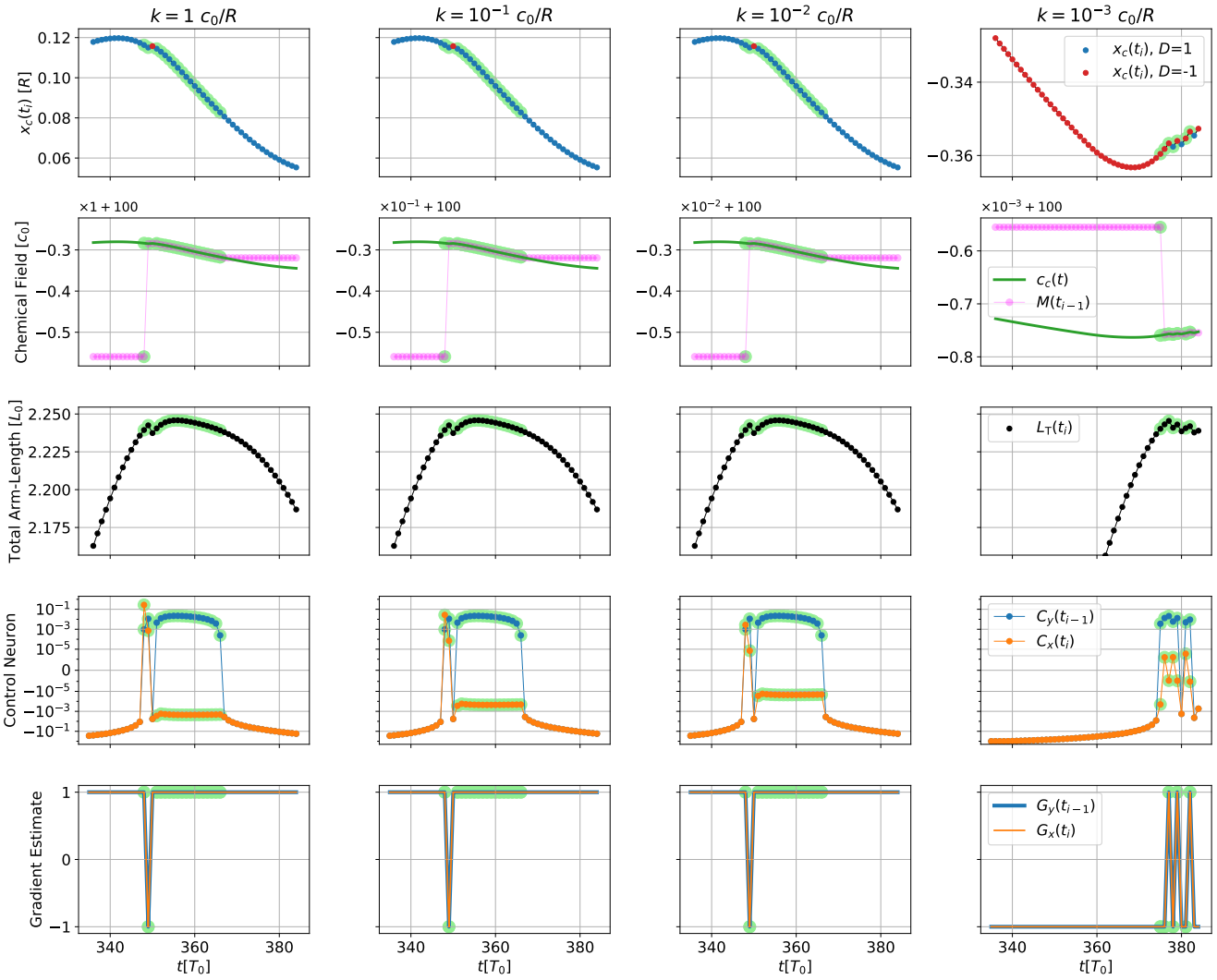

Fig. S16. Same as Fig. S15 but focused on the time interval $t_\beta \in [348, 366]\ T_0$ (highlighted by large green circles) where the temporal chemotaxis agents (see main text Fig. 3C,E and top-left panel of Fig. S13) for $k/(c_0 R) = 1, 10^{-1}$ and 10^{-2} (left three columns) decide to continue with their rightward swimming strategy; the y-axis of the control neuron (forth row) is presented on a *sym-log* scale which is linear in the interval $C_x(t_i), C_y(t_{i-1}) \in [-10^{-6}, 10^{-6}]$ and logarithmic otherwise, respectively for positive and negative values. Per default, i.e., whenever the control output of the ANN $C_y(t_{i-1}) < 0$ the respective recurrent input $C_x(t_i) = C_y(t_{i-1})$. Measurements of the chemical field are induced by the CMC cell whenever the recurrent ANN output $C_y(t_{i-1}) > 0$ as defined by Eq. (29) which updates $C_x(t_i) = c_c(t_i)/c_0 - M(t_{i-1})$. The NEAT evolved ANN learned to measure the chemical field only in short intervals whenever $L_T \gtrsim 2.23 L_0$ (see large green circles): the request by the ANN to perform measurements is controlled by $C_y(t_i)$ and solely depends on the total arm-length L_T as defined by Eq. (35). The temporal chemotaxis agent measures (here for $k = 1$ to $10^{-2}\ c0/R$) the chemical field in an interval of a leftward moving COM (see first row). In this time-interval, the temporal agent performs a series of measurements of the temporal gradient in consecutive time steps $t_i = t_{i-1} + T_0$ (emphasized by large green circles). After a short dynamical onset at the beginning of the measurement which lasts for $2T_0$ to $3T_0$, the agent interprets the corresponding time-delayed gradient input $C_x(t_i) = c_c(t_i)/c_0 - M(t_{i-1}) < 0$ correctly as positive gradient direction (see plateaus at small negative values of $C_x(t_i) \approx 10^{-3}, 10^{-4}$ and 10^{-5} in the first three columns in row four). Hence, as long as for consecutive time steps the value of $c_c(t_i) - c_c(t_{i-1}) < 0$ during the *steady state* time-frame of the measurement interval (corresponding to the plateaus in $C_x(t_i)$), the swimmer maintains its current swimming direction. For the depicted chemotaxis agent we find empirically that for slopes of $k \gtrsim 10^{-2}\ c_0/R$ the chemical gradient can be correctly predicted (left three columns) but for slopes of $k \lesssim 10^{-3}\ c_0/R$ the temporal gradient sensing strategy fails (right column). This k-dependent change in behavior can be motivated by the coupled dynamical system of changing arm-lengths which induce measurements, predictions of the chemcial gradient that cause the COM position to update and thereby influence, at which positions the chemical field is measured and, consequently, which temporal gradient $C_x(t_i) = c_c(t_i)/c_0 - M(t_i)$ the ANN perceives. If two measurements appear between two consecutive time steps, $C_x(t_i) = (c_c(t_i) - c_c(t_{i-1}))/c_0$ can be very small and even (see column four) change sign. Thus, in the latter case the chemical gradient may not be predicted correctly and temporal chemotaxis fails.

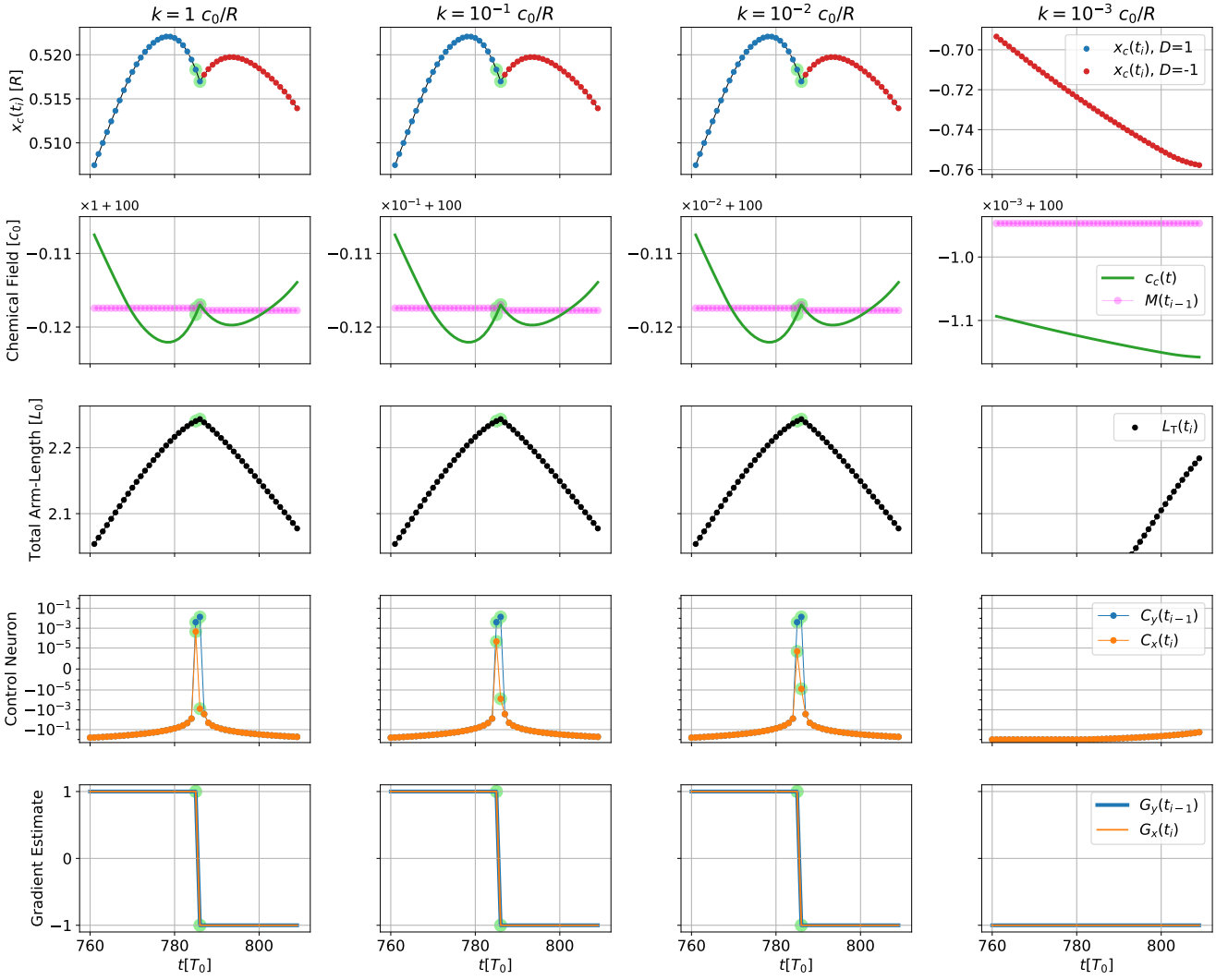

Fig. S17. Same as Figs. S15 and S16 but focused on a time interval where the swimmer initially moves rightwards but its COM position is located in a chemical field with negative slope $k = \{-1, -10^{-1}, -10^{-2}\}\ c_0/R$ (left three columns), such that it has to switch directions in order to move up the gradient. At $t_i = 785$ and $786\ T_0$, two measurement of the chemical field are induced and the swimmer correctly reverses its direction only based on these two measurements. The gradient sensing for $k = 10^{-3}\ c_0/R$ fails (right column) and the trajectory is shown for the sake of completeness in the same time-frame as the left three columns (see also Figs. S15 and S16).

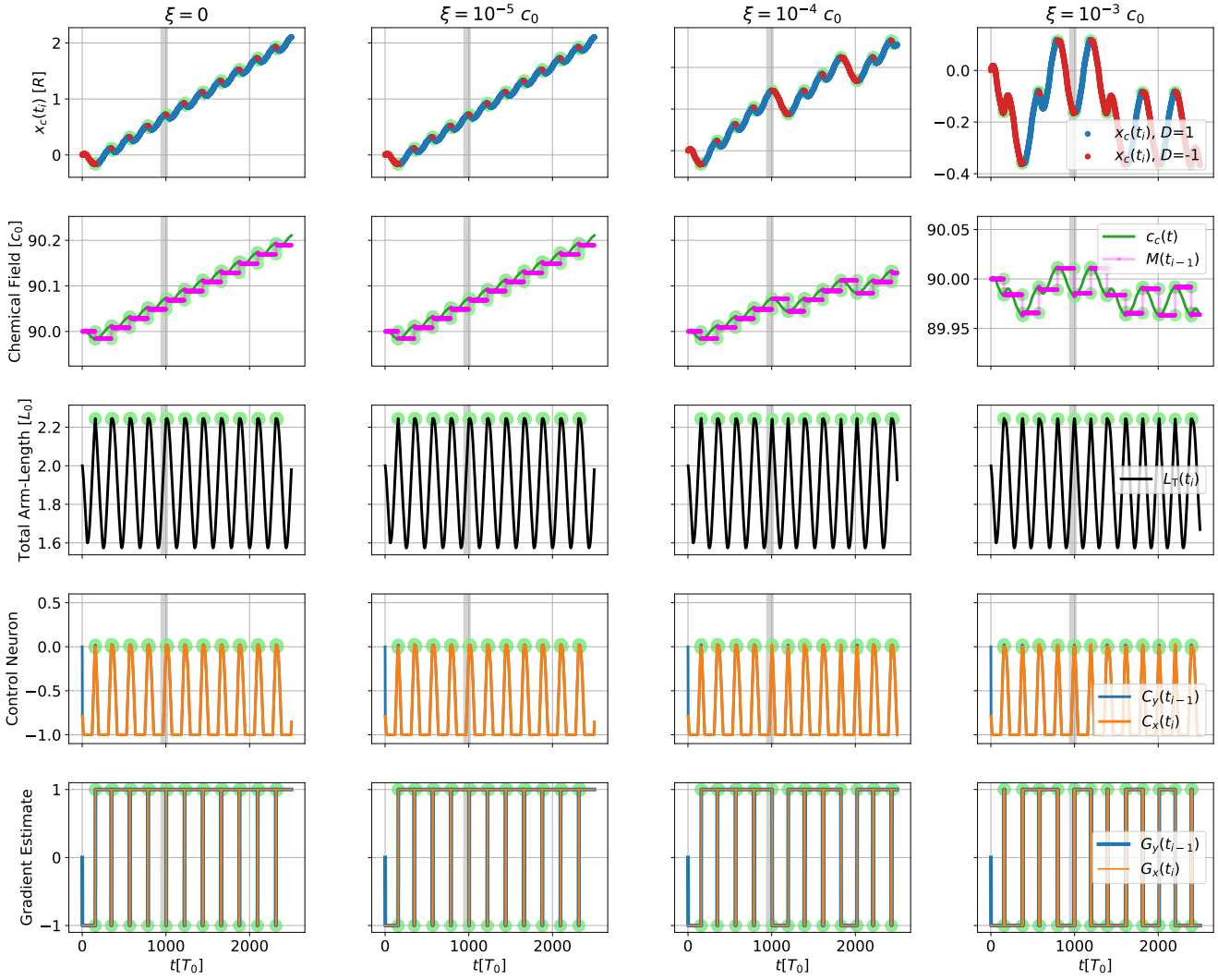

Fig. S18. Similar as Fig. S15 but for a fixed slope value of $k = 0.1\ c_0/R$ (see main text Fig. 4A) and for four different memory reading noise levels $\xi = \{0, 10^{-5}, 10^{-4}, 10^{-3}\}c_0$ (column-wise from left to right) during times of measurement $C_y(t_{i-1}) > 0$ (green circles), inducing updates of the chemical memory $M(t_i) = (c_c(t_i) + \delta c)/c_0$ following Eq. (31); δc are Gaussian distributed random numbers with standard deviation ξ. We see that for the left two cases where $\xi = 0$ and $10^{-5}\ c_0$, the temporal chemotaxis agent moves up the gradient in a *ballistic* way. For a noise level of $\xi = 10^{-4}\ c_0$ (third column) the microswimmer occasionally changes direction from rightward to the leftward motion (finite red colored $x_c(t_i)$ intervals) although being immersed in a chemical environment with positive slope $k = 0.1\ c_0/R$. The mean drift velocity is still positive and the swimmer performs a biased run-and-reverse motion towards positive x values. For an even larger noise of $\xi = 10^{-3}\ c_0$ (right column) the chemotaxis agent can not predict the chemical gradient for $k = 0.1\ c_0/R$ correctly and a stochastic behavior with center of mass oscillations around $x(t_i) = 0$ emerges. The gray vertical area in all panels correspond to a situation where the microswimmer for the low-noise regime (left two columns) can correctly resolve the gradient during a measurement interval, where for the run-and-reverse regime (third column) a *misinterpretation* of the temporal chemical field differences leads to a change in direction, and where for the stochastic regime (right column) the agent returns, by change, to the correct estimated gradient direction; details on the input and output signals of the NEAT ANN are provided in Fig. S19. We further show in main text Fig. 4D and in the top row of Fig. S21 that a uniform $\xi/(kR) \approx 10^{-3}$ ratio exists for the investigated temporal chemotaxis agent (see Fig. 3C,E), which separates the ballistic regime ($\xi/(kR) \lesssim 10^{-4}$, left two columns) and the stochastic regime ($\xi/(kR) \gtrsim 10^{-2}$, right column) and which allow the chemotaxis agent to perform run-and-reverse ($\xi/(kR) \approx 10^{-3}$, third column).

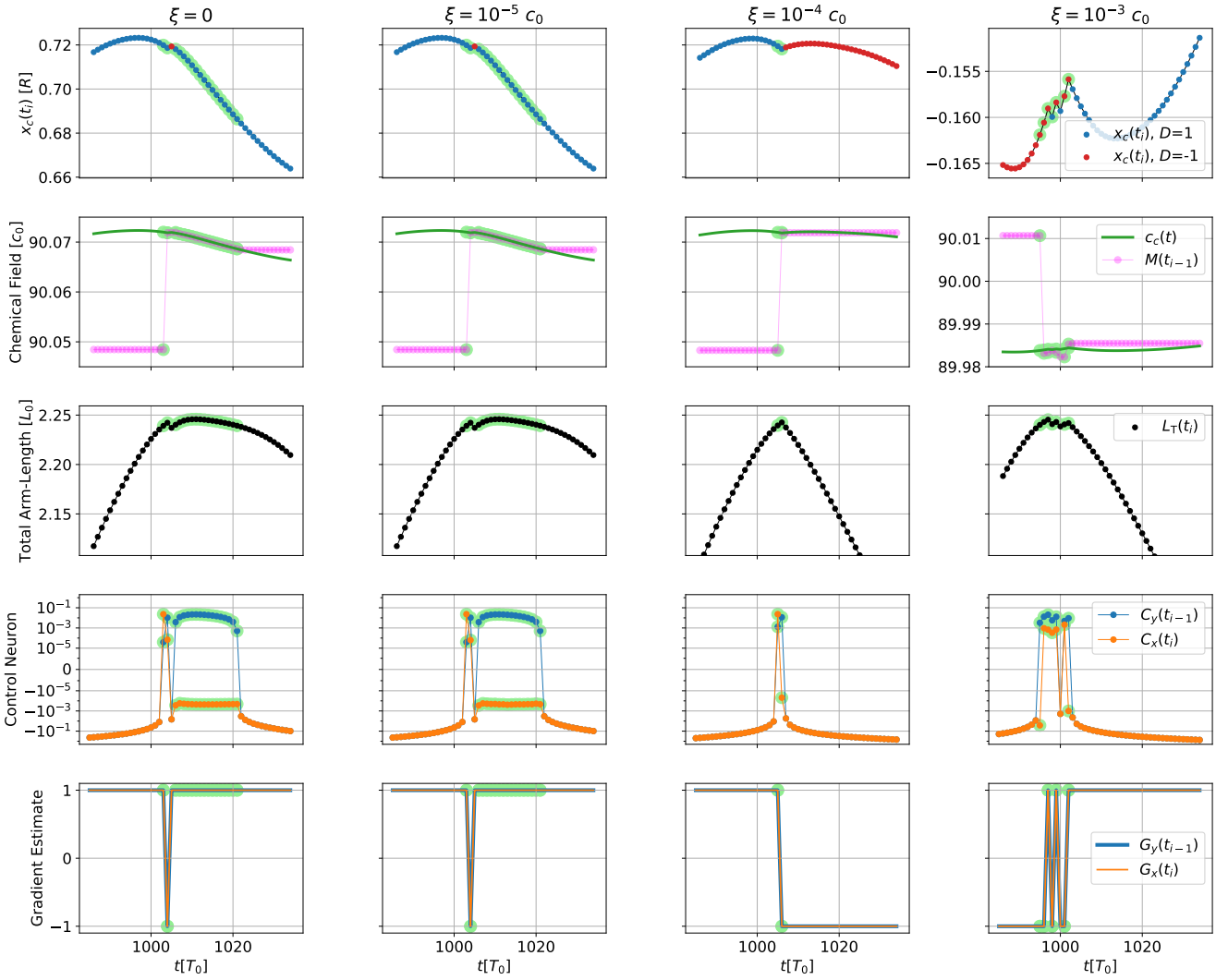

Fig. S19. Same as Fig. S18 but focused on a time interval of $t_i \approx 1000\ T_0$, highlighted in Fig. S18. The temporal chemical gradient $C_x(t_i) = c_c(t_i)/c_0 - M(t_{i-1})$ whenever a measurement occurs (i.e., $C_y(t_{i-1}) > 0$) is subjected to Gaussian distributed noise δc of standard deviation ξ via noisy memory readings $M(t_i) = (c_c(t_i) + \delta c)/c_0$ following Eq. (31). In the low-noise regime $\xi = 0$ and $10^{-5}\ c_0$ (left two columns), the chemotaxis agent can perfectly predict the gradients of the chemical field (with steepness $k = 0.1\ c_0/R$). For values of $\xi = 10^{-4}\ c_0$ (third column) the noisy memory reading can lead to occasional misinterpretations of the chemical gradient, since consecutive measurements of the chemical field $C_x(t_i) = (c_c(t_i) - c_c(t_{i-1}) + \delta_c)/c_0$ (for $C_y(t_{i-1}) > 0$) can change in sign (see also Figs. S16 and S17) which may cause the microswimmer to switch directions. For large noise values of $\xi = 10^{-3}\ c_0$ (last column) we find that, owed to the intrinsic dynamical behavior of the microswimmer, $|c_c(t_i) - c_c(t_{i-1})| \lesssim \xi$ for consecutive measurements of a chemical field with steepnes $k = 0.1\ c_0/R$, resulting in predictions of the chemotaxis gradient that are solely governed by noise and, as a consequence, in a stochastic motion of the microswimmer.

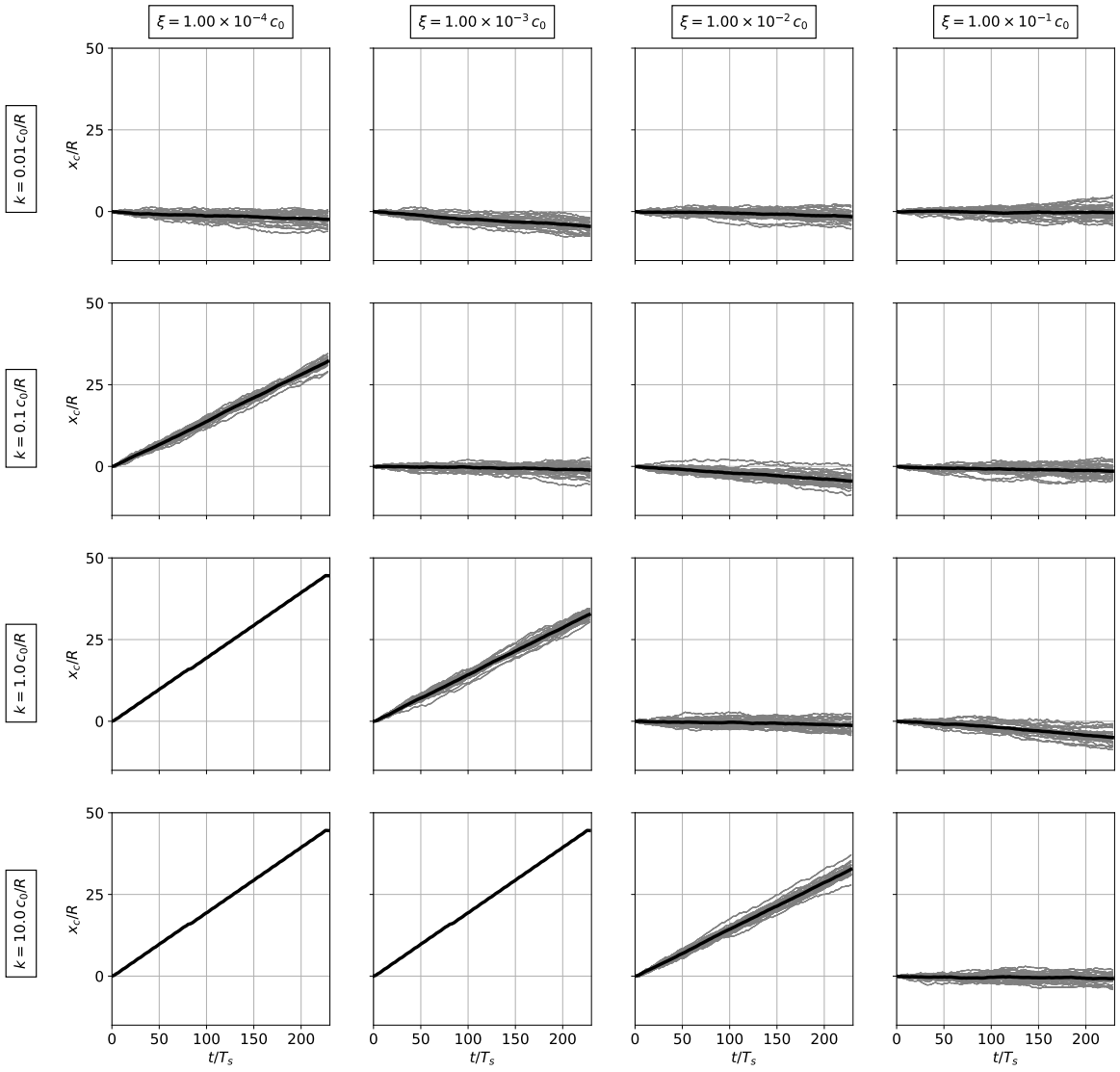

Fig. S20. Trajectories of an ensemble of 100 non-interacting temporal sensing microswimmers moving in a chemical field of slope k, their internal memory reading is influenced by a noise strength ξ. Gray lines indicate the individual center of mass trajectories of the microswimmers and black lines represent the average center of mass positions as functions time. Solutions correspond to the microswimmer presented in the main text using temporal sensing of the chemical field.

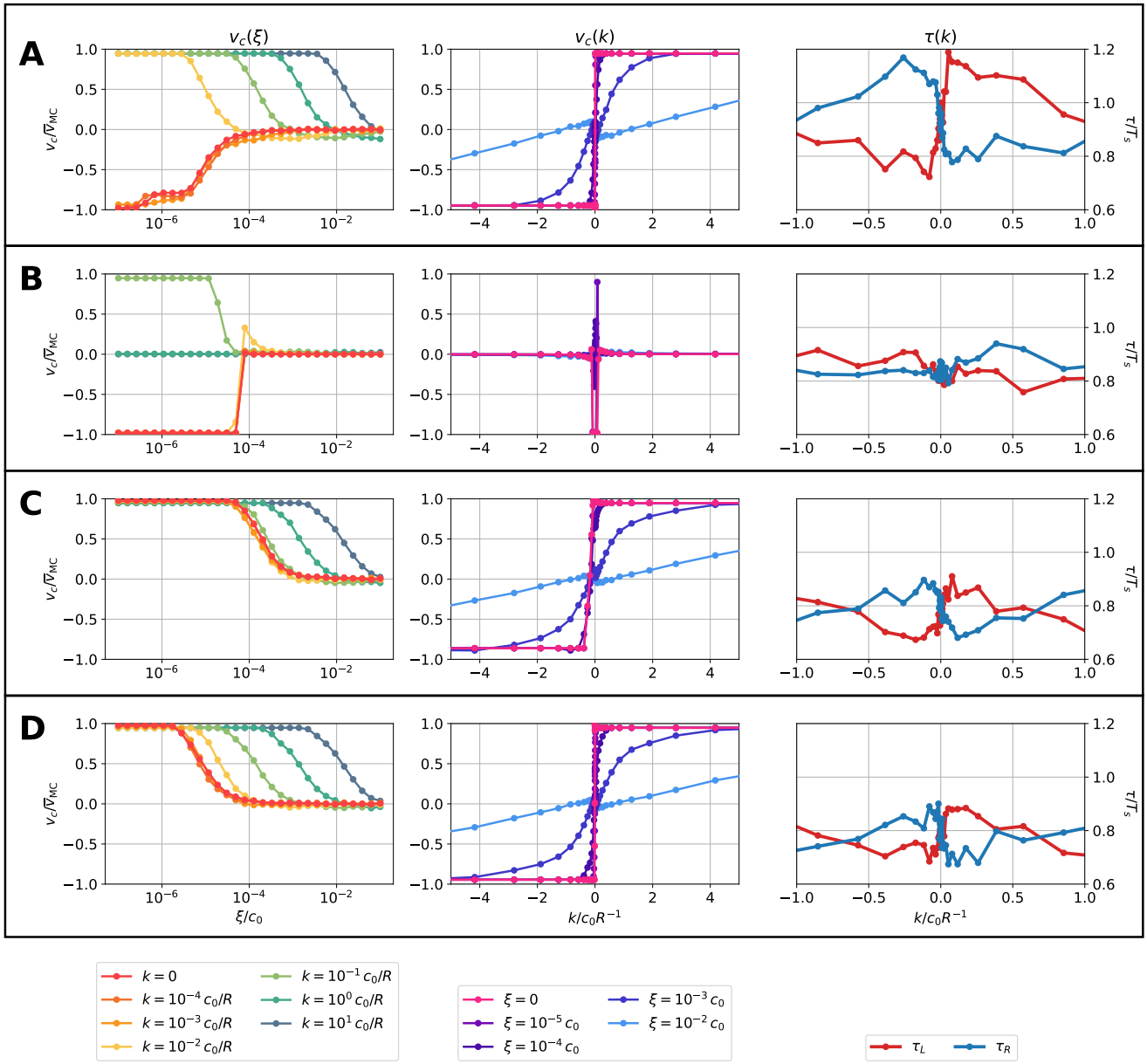

Fig. S21. (A)-(D) Run and reverse dynamics corresponding to four different chemotactic gradient estimating ANN solutions presented in the respective panels (A)-(D) in Fig. S13 where (A) shows the solutions corresponding to Fig. 4D in the main text. Left column: Mean chemotactic drift velocity v_c of a temporal sensing microswimmer (see main text Figs. 3DE) as a function of ξ for different slopes k. Center column: Same as in left column but as a function of k for different ξ. We would like to stress that in panels (A,C,D) chemotaxis is not only possible to the maximum values shown ($k = 5c_0R$). We have additionally tested the chemotactic behavior for these solutions up to ($k = 100c_0R$), and we found that they are still able to perform chemotaxis in these very steep gradients. Right column: Decay times to the right (τ_R) and to the left τ_L depending on k at noise level $\xi = 0.01c_0$.

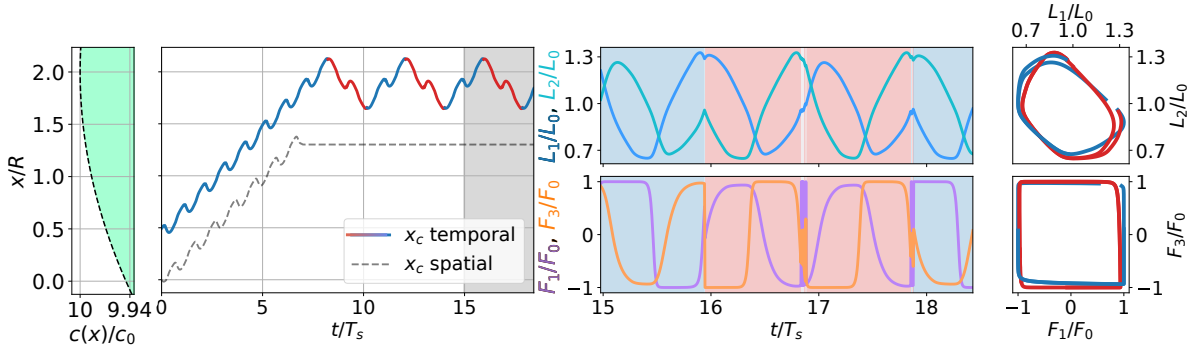

Fig. S22. Same as Fig. 2C of the main text but with a larger width ($\sigma = 19R$) of the Gaussian chemical profile which is centered at $x_0 = 2R$, leading to a broader region where the microswimmer moves around the maximum of the profile. The mean COM position of the swimmer for $t \gtrsim 7.5\ T_S$ is slightly biased towards smaller values of x than the actual peak position of the chemical field at $x = 2R$.

References

1. A Najafi, R Golestanian, Simple swimmer at low Reynolds number: Three linked spheres. *Phys. Rev. E* **69**, 062901 (2004).
2. R Golestanian, A Ajdari, Analytic results for the three-sphere swimmer at low Reynolds number. *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.* **77**, 1–6 (2008).
3. HJ Kelley, Gradient Theory of Optimal Flight Paths. *ARS J.* **30**, 947–954 (1960).
4. I Goodfellow, Y Bengio, A Courville, *Deep Learning.* (MIT Press), (2016).
5. RS Sutton, AG Barto, *Reinforcement Learning: An Introduction.* (The MIT Press), Second edition, (2018).
6. CJCH Watkins, P Dayan, Q-learning. *Mach. Learn.* **8**, 279–292 (1992).
7. TP Lillicrap, et al., Continuous control with deep reinforcement learning (2015).
8. Z Gu, Z Jia, H Choset, Adversary A3C for Robust Reinforcement Learning (2019).
9. J Schulman, F Wolski, P Dhariwal, A Radford, O Klimov, Proximal Policy Optimization Algorithms (2017).
10. KO Stanley, R Miikkulainen, Evolving Neural Networks through Augmenting Topologies. *Evol. Comput.* **10**, 99–127 (2002).
11. A McIntyre, M Kallada, CG Miguel, CF da Silva, neat-python (https://github.com/CodeReclaimers/neat-python) (2019).
12. Vienna Scientific Cluster (VSC4) (http://typo3.vsc.ac.at/systems/vsc-4) (2019).