Statistical abstraction for multi-scale spatio-temporal systems

Statistical abstraction for multi-scale spatio-temporal systems

MICHALIS MICHAELIDES, JANE HILLSTON, and GUIDO SANGUINETTI, School of Informatics, University of Edinburgh, United Kingdom

Modelling spatio-temporal systems exhibiting multi-scale behaviour is a powerful tool in many branches of science, yet it still presents significant challenges. Here we consider a general two-layer (agent-environment) modelling framework, where spatially distributed agents behave according to external inputs and internal computation; this behaviour may include influencing their immediate environment, creating a medium over which agent-agent interaction signals can be transmitted. We propose a novel simulation strategy based on a statistical abstraction of the agent layer, which is typically the most detailed component of the model and can incur significant computational cost in simulation. The abstraction makes use of Gaussian Processes, a powerful class of non-parametric regression techniques from Bayesian Machine Learning, to estimate the agent's behaviour given the environmental input. We show on two biological case studies how this technique can be used to speed up simulations and provide further insights into model behaviour.

CCS Concepts: • **Theory of computation → Abstraction**; • **Applied computing → Systems biology**;

Additional Key Words and Phrases: multi-scale systems, spatio-temporal, agent-based, statistical abstraction, coarsening

ACM Reference Format:
Michalis Michaelides, Jane Hillston, and Guido Sanguinetti. 2019. Statistical abstraction for multi-scale spatio-temporal systems. 1, 1 (April 2019), 28 pages. https://doi.org/0000001.0000001

1 INTRODUCTION

Science is often tasked with examining natural or artificial systems characterised by spatial dependence and complex dynamics. The complexities that these characteristics induce on the emergent system behaviour mean that detailed models are often constructed in order to study them through simulation. This approach has been used extensively in applications, ranging from cyber-physical systems to collective adaptive systems of human behaviour and to cellular systems. Nevertheless there is still room for advancement through automating the ability to recover simpler models that still capture the dynamics with sufficient faithfulness, but which may have a lower computation cost. This is especially true for systems involving onerous stochastic simulations [Dada and Mendes 2011; Gilbert et al. 2015].

We consider here a general framework which encompasses a large class of spatio-temporal systems. In this framework, multiple identical agents are distributed in space over an external field. The agents perceive the field locally and perform internal stochastic computations to determine their subsequent behaviour, such that their actions are influenced by their environment. We also

Authors' address: Michalis Michaelides, mic.michaelides@ed.ac.uk; Jane Hillston, jeh@inf.ed.ac.uk; Guido Sanguinetti, gsanguin@inf.ed.ac.uk, School of Informatics, University of Edinburgh, 10 Crichton Street, Edinburgh, EH8 9AB, United Kingdom.

XXXX-XXXX/2019/4-ART $15.00
https://doi.org/0000001.0000001

allow the agents to act locally upon the external field, enabling the latter to become a medium for signals between agents. This framework subsumes a wide range of systems, from swarm robots performing a task in space, to bacteria exploring a nutrient field, or agents responding to distress signals.

In this paper, we propose a method to replace expensive stochastic parts of the model with input-output maps estimated via a machine learning procedure. We focus on a particular macro-scale behaviour as output from the model, and devise a statistical abstraction of the system in order to produce a simpler system which preserves the macro-scale behaviour. Crucially, we do not care for the detailed internal state of the model, but only an abstracted version sufficient to capture its qualitative behaviour. The abstracted state is formalised as the satisfaction output of a set of logical properties evaluated on the original state. We estimate the necessary input-output relation by learning a parameters-to-behaviours regression map using Gaussian Processes (GPs), a powerful class of non-parametric Bayesian regression models. Our work is motivated by earlier work on using GPs to learn effective characterisations of system behaviour [Bortolussi et al. 2015, 2016; Michaelides et al. 2016].

This paper is an extension of our previous conference paper [Michaelides et al. 2017]. The major extension compared to the conference version consists in adapting the statistical framework to also handle agent-environment interactions, thereby closing the information loop and allowing for environment-mediated agent-agent interactions. To illustrate this methodological extension, we also provide as an additional case study a new model and abstraction for the chemotactic aggregation of the social amoeba *Dyctostelium discoideum*. In addition to this methodological extension, we also provide a more comprehensive explanation of the mathematical steps required by our abstraction strategy, with the aim to make the method, and in particular the underpinning machine learning, more accessible to a wider community.

The rest of the paper is organised as follows: we start with some background on spatio-temporal systems (Section 2). The general framework for our statistical abstraction methodology is presented in Section 3, followed by a brief discussion of related work (Section 4. We then present two case studies describing applications of the abstraction on a model of *E. coli* chemotaxis and a model of *D. discoideum* aggregation (Sections 5 and 6 respectively), which exemplify the methodology and provide results assessing the quality and efficiency of the abstraction. We conclude with a discussion on the utility of the method and closing remarks about prospective expansion of the work (Section 7).

2 BACKGROUND

2.1 Spatio-temporal agent models

We start by defining the class of spatio-temporal agent models we will consider in this paper. Let $\mathcal{D}$ be a spatial domain (usually a compact subset of $\mathbb{R}^n$ with $n = 2, 3$), and let $[0, T]$ be the temporal interval of interest. We define the *spatio-temporal field* $f : \mathcal{D} \times [0, T] \to \mathbb{R}$ to be a real-valued function defined on the spatial and temporal domains of interest. A spatio-temporal agent model is a triple $(\mathcal{D}, f, \mathcal{A})$ where $\mathcal{A}$ is a collection of point *agents* whose location follows a stochastic process which depends on the spatio-temporal field. Note that even though we realise that this is not the most general case, as agents may be spatially extended, or directly interact with each other, a form of agent-agent interaction is feasible within this framework. As illustrated through the case study of a *D. discoideum* model in Section 6, the agents may affect the evolution of the spatio-temporal field — this allows the field to transmit signals from agent to agent, enabling interaction.

2.2 Multi-scale models

In many practical situations, one is interested in modelling not only the movement of the agents, but also the mechanism through which sensing and decision making is carried out within each agent. This naturally leads to structured models with distinct layers of organisation, with behaviour in each layer informing the simulation that takes place at the layer above or below. We will assume that the internal workings of the agent are also stochastic, and we model them here as a Markov chain with a discrete state-space.

In the first case-study presented (Section 5), the internal workings of an agent are modelled by a *population Continuous Time Markov Chain* (pCTMC). Note that the pCTMC is the internal model for a *single* agent here, not for multiple agents. Formally, a pCTMC is defined as follows.

Definition 2.1. A population CTMC is a continuous-time Markov chain [Norris 1998] with a discrete state-space $\mathcal{V}$, and an associated transition rate matrix Q. Each state $\mathbf{V} \in \mathcal{V}$ counts the number of entities of each type or "species" in a population, $\mathbf{V} \in \left\{\mathbb{N}^0\right\}^d$ for d species. Transitions in this space occur according to the rates given by Q.

The transitions can be regarded as occurrences of chemical reactions, written as

$$\sum_{i=1}^{d} r_i V_i \xrightarrow{\tau(\mathbf{V})} \sum_{i=1}^{d} s_i V_i, \tag{1}$$

where for every species V_i, r_i particles of V_i are consumed and s_i particles are created. The transition rate $\tau(\mathbf{V})$ depends upon the current state of the system, and is the rate parameter of an exponential distribution governing the waiting times for this transition. The above transition rates of allowed reactions define all elements of the rate matrix Q.

In the second case-study (Section 6), the internal workings of a cell are modelled by a Discrete Time Markov Chain (DTMC). Formally, a DTMC is defined as follows.

Definition 2.2. Consider a random variable X_n which takes values from a countable state-space $I = \{1, 2, 3, \dots\}$. Let π be a probability distribution over I, such that $\sum_{i \in I} \pi_i = 1$, and let $P = (p_{ij} : i, j \in I)$ be a *stochastic matrix*, with $\sum_{j \in I} p_{ij} = 1$. A collection of random variables $\{X_n\}_{n \in \mathbb{N}_0}$ constitutes a *discrete-time Markov chain* with *initial distribution* π and *stochastic matrix* P (DTMC(π, P)), if and only if it satisfies

- $P(X_0 = i) = \pi_i$, and
- $P(X_{n+1} = j \mid X_n = i, \dots, X_0 = k) = P(X_{n+1} = j \mid X_n = i) = p_{ij}$.

A trajectory of length M drawn from such a DTMC is a particular realisation of the random variable collection $\{X_n\}$ $\forall n \in \{1, \dots, M\}$.

2.3 Simulating multi-scale systems

Multi-scale spatio-temporal systems are in general amenable to analytical techniques only in the simplest of cases. For the vast majority of real-world models, simulation-based analysis is the only option to gain behavioural insights.

Simulation of spatio-temporal systems typically employs nested algorithms: having chosen a time-discretisation for the spatial motion (which is assumed to have the slower time-scale), a spatial step is taken. Then, the value of the external field is updated, and the internal model is run for the duration of a given time-step with the new rates (corresponding to the updated value of the external field). A sample from the resulting state distribution then determines the velocity of the agent for the next time-step.

Clearly, this iterative procedure, while asymptotically exact (in the limit of small time discretisation), is computationally very demanding. This has motivated several lines of research in recent years [Bortolussi et al. 2015; Goutsias 2005; Haseltine and Rawlings 2002; Rao and Arkin 2003].

3 METHODOLOGY FOR STATISTICAL ABSTRACTION

In a multi-scale system, output from a set of processes in one layer in the system is passed as input to another layer; these processes are often computationally expensive. We present a methodology to abstract away such a set of processes and replace them with a more efficient stochastic map from the input to the output, governed by an underlying probability function. We approximate this probability function using Gaussian processes after observing many input-output pairs from the processes to be abstracted. The output consists of truth evaluations of properties expressed in logical formulae, which capture some behaviour of the system that is to be preserved by the abstraction.

3.1 Statistical abstraction framework

Consider a Markov chain S, which given an initial state $\mathbf{s}_0$, running time Δt, and input $\mathbf{q} \in \mathbb{R}^D$ which completely determines transition rates, generates a trajectory $\mathbf{s}_{[0,\Delta t]}$. At each time step n in the simulation of a multi-scale system, the trajectory $\mathbf{s}^{(n)}_{[0,\Delta t]}$ is checked for satisfaction of a logical property resulting in output $y^{(n)} = f\left(\mathbf{s}^{(n)}_{[0,\Delta t]}\right),\ y^{(n)} \in \{\top, \bot\}$. For the next time step, the last state in the Markov chain is kept as the new initial state (i.e. $\mathbf{s}^{(n)}_0 = \mathbf{s}^{(n-1)}_{\Delta t}$), and with new input $\mathbf{q}^{(n)} \in \mathbb{R}^D$ to determine transition rates the process is repeated. This layer of the multi-scale system can therefore be described as a set of operations at each time step n:

$$S\left(\mathbf{s}^{(n)}_0 = \mathbf{s}^{(n-1)}_{\Delta t}, \Delta t, \mathbf{q}^{(n)}\right) = \mathbf{s}^{(n)}_{[0,\Delta t]}; \tag{2}$$

$$f\left(\mathbf{s}^{(n)}_{[0,\Delta t]}\right) = y^{(n)}. \tag{3}$$

Note that we consider a single property here for simplicity (so a single binary value), but one could generalise to multiple properties, and hence, multi-valued output. This output then becomes input to a higher layer in the multi-scale system.

Our goal is to construct a system $\tilde{S}$ that is cheaper to simulate, whose output will be consistent with the original system S. Since the system is stochastic, *consistent* refers to having the same probability distribution for the output random variable $y^{(n)}$ given the same input $\mathbf{q}^{(n)}$ and following previous output $y^{(n-1)}$. Fundamentally, we seek to approximate the (generally) non-Markov process of outputs $\{y^{(n)}\}$ with a Markov one. To describe the abstracted system, we write:

$$\tilde{S}\left(y^{(n-1)}, \mathbf{q}^{(n)}\right) = y^{(n)}. \tag{4}$$

Replacing the initial state $\mathbf{s}^{(n)}_0 = \mathbf{s}^{(n-1)}_{\Delta t}$ input with the previous output $y^{(n-1)}$ allows us to substitute the whole layer of fine operations (2, 3) with the cheaper abstracted system $\tilde{S}$ (4), unburdening the multi-scale system. We regard this abstracted system to be a stochastic map from the internal state of the system, now abstracted to the last output $y^{(n-1)}$, and some external input $\mathbf{q}^{(n)}$, to a new output $y^{(n)}$. The latter being a discrete random variable, the task is to estimate a probability distribution over the output domain from which to sample the output. Since we expect this distribution to depend upon the previous output $y^{(n-1)}$ and external input $\mathbf{q}^{(n)}$, we use Gaussian process regression with an appropriate observation likelihood to estimate an underlying probability function $\Psi(y, \mathbf{q})$ which governs the output of $\tilde{S}$.

It follows that the abstraction will become more accurate the faster the Markov chain S mixes, since dependence on the initial state of the chain will no longer matter — in fact, for fast enough mixing times relative to Δt, one could even drop the output feedback and produce stochastic output y only given the input $\mathbf{q}$. Thus, we expect the abstraction to work particularly well for the components of a system which equilibrate faster than the others. For the cases we present below, notice that the internal agent dynamics which determine motility are faster than the changes in the environmental input that affects them.

In our general construction, the output of the system is taken to be a combination of boolean satisfaction values for a set of properties. Owing to its discrete nature, the resulting abstraction could be interpreted as a discrete-time Markov chain (DTMC) whose state-space comprises of every output combination. Our task is then to determine transition rates for this DTMC to make its paths consistent with output of the original system. If one wishes to increase accuracy, the DTMC can be made to be of a higher order. A higher order DTMC means that a longer output history is retained and affects the next output, and can therefore be expected to better approximate the original output dynamics. In the two examples presented here, we construct a first-order DTMC for the first case and a second-order DTMC for the second case.

3.2 Approximating the underlying probability function

There are many approaches one could take to infer the probability function $\Psi(y, \mathbf{q})$, necessary for the abstraction. Here we make use of Gaussian processes (GPs), a powerful non-parametric regression method, originating in geo-statistics and significantly extended in the realm of machine learning. Consider a collection of stochastic variables $\{Y(x)\}$, where x lives in a continuous domain $\mathcal{X}$. Such a stochastic process is termed a Gaussian process iff any finite set of variables $\{Y(x_i) : x_i \in \mathcal{X}, i = 1, \ldots, N\}$ in the collection, is normally distributed. Equivalently, a GP follows a normal distribution over a separable Hilbert space of functions. This normal distribution can be conditioned on a finite number of (potentially noisy) observations of the function to be inferred, learning new mean and covariance parameters. These are computable at any point in the domain and correspond to the expected value of the function and associated variance at that point, respectively.

GPs are universal function approximators. The choice of covariance kernel determines the prior over the space of functions considered, and thus affects how many observations are required to get a good estimate of the underlying function.[1] However, given enough observations, a GP with an appropriate kernel will approximate any function within a particular family arbitrarily well. Here we make use of the squared exponential, or Gaussian, kernel

$$k(x, x') = \sigma^2 \exp\left[-\frac{1}{2}(x - x')^\top M (x - x')\right],$$

where σ is a scalar *amplitude* hyperparameter which indicates the magnitude of variation in the function, and $M = \mathrm{diag}(\ell)^{-2}$ is a diagonal matrix which scales each input dimension by a *characteristic length-scale*, indicative of how correlated output is along that dimension. Intuitively, functions that exhibit more frequent variations along a dimension i are more probable when ℓ_i is smaller, and functions with larger amplitudes of variation are more probable when σ^2 is larger. The squared exponential kernel above provides a prior over functions $f : \mathcal{X} \to \mathbb{R}$, $\mathcal{X} \subseteq \mathbb{R}^n$, which populate a dense subspace of $L^2(\mathcal{X})$; our GP should therefore be able to approximate arbitrarily well any function in $L^2(\mathcal{X})$. We refer to [Rasmussen and Williams 2006] for a more comprehensive account of GPs.

[1] If an oracle allowed observation of the function at any point in the input domain, we would be able to actively reduce our uncertainty over it as desired (i.e. we could take observations where we deem lower uncertainty necessary). In the application cases presented here the system is observed through entire simulations, and so we do not have this power.

Since training observations are binary samples of a Bernoulli distribution (satisfaction *true* or *false* of logical properties) or samples of a categorical distribution (in the case of multiple properties), but GPs regress over a continuous unbounded variable, some adjustments to the standard GP regression must be made for correct evaluation of the underlying probability function Ψ. GP regression with its many variations for different problem tasks is well described in [Rasmussen and Williams 2006]. The necessary adjustments which we adopt here are found in the Gaussian process classification (GPC) section of the book, and essentially amount to identifying that the class probability function is Ψ, where the class is the property satisfaction outcome. A detailed explanation can be found in Appendix A, available in the digital version of the paper.

As discussed, GP regression is a statistical approach to approximate an unknown function based on a finite set of input/ output instantiations. The quality of this approximation is affected by a number of factors, including model choices such as the covariance function, but naturally the prime determinant of approximation quality is the amount of data available. In this application, since the data is generated by model simulations, we have a degree of control on how much data is available. In practice, however, it is extremely difficult to estimate *a priori* the size of data set required for a certain accuracy; in the case study in Section 5 we present a practical empirical strategy to address this problem.

4 RELATED WORK

When it comes to *abstracting* stochastic systems, there is a wide literature of methods to consider. For the cases where the system is solely defined in terms of a population continuous-time Markov chain (pCTMC) found normally at large counts, the chemical Langevin equation provides an approximation for the whole process, while systematic approximation methods for the moments of the distribution of the process existed since the 60s [Gillespie 2000; Kampen 1961; Kurtz 1971]. Both approaches have seen considerable improvement over the last decades, increasing their range of applicability [Schnoerr et al. 2017]. Other approaches to the problem of efficiently solving biochemical systems attempt to gradate species' concentration levels to discrete intervals [Ciocchetta and Hillston 2009; Palaniappan et al. 2017], thereby reducing the state-space of the underlying CTMC to be solved, or employ time-scale separation if possible [Bortolussi et al. 2015; Goutsias 2005; Haseltine and Rawlings 2002; Rao and Arkin 2003].

All of the above are firmly situated in the domain of pCTMCs and are agnostic to the demands made of the process downstream — whether the pCTMC is checked for reaching a particular value, or having maintained a value for a particular duration in some time interval, does not affect the approximation these methods will yield. We take a more holistic view in this work and consider the system to be abstracted as a component of a larger multi-scale system. As such, only a particular aspect of the component is relevant to the multi-scale system, and it is this aspect which our abstraction attempts to preserve.

We take the relevant component output to be the evaluation of a logical property on a stochastic trajectory drawn from a Markov chain, which comprises the internal process of the component to be abstracted. The transition rates of the chain depend on some input the component receives, and this enables us to utilise results in [Bortolussi et al. 2016] to estimate the output given the input via the method of Gaussian process regression.

5 THE CASE OF *E. COLI* CHEMOTAXIS

5.1 Background on chemotaxis in the *Escherichia coli* bacterium

Foraging is a central problem for microbial populations. The bacterium *Escherichia coli* will normally perform a random walk within a spatial domain where nutrient concentration is constant (e.g.

a Petri dish). When presented with a spatially varying nutrient field, a phenomenon known as *chemotaxis* arises. As the bacterium performs a random walk in the nutrient field encountering changing nutrient levels, its sensory pathway effectively evaluates a temporal gradient of the nutrients (or ligands) it experiences; the walk is biased so that the bacterium experiences a positive temporal gradient more often than not [Berg et al. 1972; Sourjik and Wingreen 2012; Vladimirov et al. 2008]. Since the bacterium is moving in the field, the temporal gradient is implicitly translated into a spatial one, so the bacterium drifts toward advantageous concentrations. Implicitly translating a temporal gradient to a spatial one through motion is necessary for the bacterium cell, because its body size is too small to allow for effective calculation of the spatial gradient of a chemical field at its location. As a result, we can safely regard the bacteria as point-like agents.

Motor control in E. coli. An *E. coli* cell achieves motility by operating *multiple* flagellum/motor pairs (F/M), which can either drive it straight (subject to small Brownian perturbation), or rotate it in place. Thus, the cell can either be 'tumbling' (re-orienting itself while stationary) or 'running' (propelling itself forward while maintaining direction) at any time (Figure 1: left, centre). The motility state, RUN/TUMBLE, of the cell is determined by the number of flagella found in particular conformations. The model in [Sneddon et al. 2012] suggests three possible conformations for a flagellum: *curly* (C), *semicoiled* (S) and *normal* (N). The associated motor is modelled as a stochastic bistable system, which rotates either clockwise (CW) or counter-clockwise (CCW). Changes in motor rotation induce conformational changes on the associated flagellum. Transition rates between motor states are given by rate parameters k_+ and k_- for transitions CW $\rightarrow$ CCW and CCW $\rightarrow$ CW, respectively. The possible transitions between flagellum/ motor states are summarised in the schematic diagram in Figure 1: right. *E. coli* normally has of the order of ten flagella and associated motors; the dynamics of the pair flagellum/motor population therefore lends itself to be easily described as a pCTMC. The $k_\pm$ transition rates depend on the temporal gradient evaluated by the chemotaxis pathway, and represent the functional interface of the bacterium with its external environment.

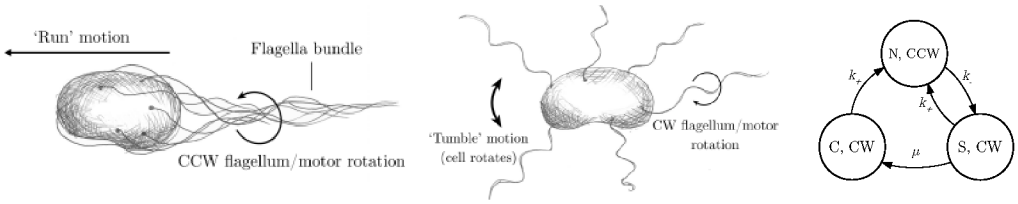

Fig. 1. The two motility modes of an *E. coli* cell. Left: the F/M are in CCW conformations, forming a helical bundle and propelling the cell. Centre: the F/M are in CW conformations, breaking the bundle apart and causing the cell to re-orient in place. Right: CTMC for a single F/M, with three conformation states and transition rates $k_\pm(m, L)$ and fixed $\mu = 5\text{s}^{-1}$.

The classical mathematical model for the sensory response of the cell to external ligand concentration changes is provided by the Monod-Wyman-Changeux (MWC) model [Hansen et al. 2008; Sneddon et al. 2012; Sourjik and Berg 2004]. The model considers sensor clusters which signal information about ligand concentration changes to the motors, by triggering a biochemical response in the cell (phosphorylation of the CheY protein which binds to the motors) affecting the switching rates of rotation direction, $k_\pm$.

The full MWC model is still highly complex; in practice, we follow [Sneddon et al. 2012] and adopt a simplified model of sensory response to describe the dependency of motor rates $k_\pm$ on ligand concentrations. This involves resolving the CheY signalling pathway to the single variable

m, which represents the methylation state of the ligand receptors and whose stochastic evolution is dependent on the ligand concentration L. Since m depends on past L concentrations the cell has been in, one may think of it as a *chemical memory* of sorts which encodes the value of L at previous times. The time comparison window is determined by how fast methylation happens — faster methylation leads to a shorter memory.

Sneddon et al. [2012] then resolve the entire dependency chain of the chemotaxis pathway to Equations 5 and 6. The motor switching rates $k_{\pm}(m, L)$ are given by the deterministic equation

$$k_{\pm} = \omega \cdot \exp\left\{ \pm \left[\frac{g_0}{4} - \frac{g_1}{2} \left(\frac{Y_p(m,L)}{Y_p(m,L) + K_D} \right) \right] \right\}, \tag{5}$$

where

$$Y_p(m,L) = \alpha \cdot \left[1 + e^{\epsilon_0 + \epsilon_1 m} \cdot \left(\frac{1 + L/K^{\text{off}}_{\text{TAR}}}{1 + L/K^{\text{on}}_{\text{TAR}}} \right)^{n_{\text{TAR}}} \cdot \left(\frac{1 + L/K^{\text{off}}_{\text{TSR}}}{1 + L/K^{\text{on}}_{\text{TAR}}} \right)^{n_{\text{TSR}}} \right]^{-1}.$$

The methylation process can be naturally modelled as a birth / death process with rates depending on ligand concentration; again following [Sneddon et al. 2012] we take a fluid approximation of this, yielding the Ornstein-Uhlenbeck (OU) process:

$$\frac{dm}{dt} = -\frac{1}{\tau}(m - m_0(L)) + \eta_m(t). \tag{6}$$

In the above stochastic differential equation (SDE), $\eta_m = \sigma_m \sqrt{2/\tau}\Gamma(t)$, $\Gamma(t)$ is the normally distributed random process with 0 mean and unit variance, σ_m is the standard deviation of fluctuations in the methylation level, and $m_0(L)$ is an empirically derived function whose output is the methylation level required for full adaptation at the current external ligand concentration L. The adaptation rate τ, determines how fast methylation occurs and so, how long the 'chemical memory' of previous L values is in the system. The constants τ, along with mb_0 and α involved in the $m_0(L)$ function (see [Sneddon et al. 2012]), fully parametrise the methylation evolution. See [Vladimirov et al. 2010] for reported values of constants used in Equation 5 and [Frankel et al. 2014; Sneddon et al. 2012] for a detailed derivation of the results. Equations 5 and 6 couple the transition rates of the pCTMC in Figure 1: Right, with the external ligand concentrations, and therefore fully describe the internal model of the *E. coli* chemotactic response.

5.2 Simulating chemotaxis in *E. coli*

Simulations of the *E. coli* model outlined proceed along the general lines discussed in Section 2.3. Given a value of the ligand field and a characteristic time-step Δt, we draw samples of the SDE (6) using the Euler-Maruyama method, a standard method for simulating SDEs.

In the F/M pCTMC system and following the reaction equation style, each species represents a different F/M conformation for a total of three species. With respect to the CTMC in Figure 1: Right, the pCTMC counts how many F/M (molecules) there are of each conformation (species). The following transitions (reactions) are said to occur:

$$\begin{aligned} (S\_CW) \xrightarrow{\mu} (C\_CW),\ (C\_CW) \xrightarrow{k_+} (N\_CCW), \\ (S\_CW) \xrightarrow{k_+} (N\_CCW),\ (N\_CCW) \xrightarrow{k_-} (S\_CW). \end{aligned} \tag{7}$$

Note that in the above rate transitions there are dependencies on both external (L) and internal (m) states: $k_{\pm}(m, L)$, where L is an external input to the system (the external chemoattractant concentration at the time) and m is the current methylation level (sampled from the OU process in Equation 6 every Δt). Instead, the rate transition for $(S\_CW) \rightarrow (C\_CW)$ is fixed, $\mu = 5\text{s}^{-1}$.

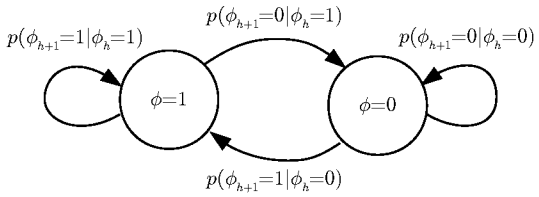

Fig. 2. DTMC with two states, $\phi_{\text{RUN}} \in \{\top, \bot\}$. The transition probabilities depend on internal methylation level m and external ligand concentration L.

Using the exact Gillespie algorithm [Gillespie 1977], we then simulate the internal pCTMC for a length of time Δt to draw a sample configuration of the flagella/motors system. Formally, trajectories of length Δt are checked against a property specifying the motility state for the cell (RUN/TUMBLE),

$$\phi_{\text{RUN}}(\mathbf{s}) = (N \geq 2) \wedge (S = 0), \tag{8}$$

where $\mathbf{s} = (S, C, N)$ is the last state of the flagellum/motor pairs in the CTMC trajectory (each element of the vector counts how many F/M pairs there are in each conformation). The logical property above evaluates to *true* (1) if a given cell state $\mathbf{s}$ has more than 2 F/M in the *normal* conformation and none in the *semicoiled* conformation; otherwise, it evaluates to *false* (0).

The spatial location of the bacterium is then updated according to a simple rule: if the sampled internal state corresponds to RUN, the agent moves rectilinearly and updates its position $\vec{r} \leftarrow \vec{r} + \vec{v} \cdot \Delta t$, where $v = 20\mu$m/s, the speed of the bacterium. Otherwise, if the internal state corresponds to TUMBLE, the agent remains still and its velocity is updated $\vec{v} \leftarrow R(\theta) \cdot \vec{v}$, where $R(\theta)$ is the standard 2D unitary rotation matrix through an angle θ, and θ is a tumbling angle sampled from a Gamma distribution as reported in Sneddon et al. [Sneddon et al. 2012].

The above simulation scheme, outlined in Algorithm 1 (Appendix B), produces a chemotactic response to a ligand gradient. It takes $\sim$ 270s to simulate a single cell trajectory of $t_{\text{end}} = 500$s with a time-step $\Delta t = 0.05$.

5.3 Abstracting the *E. coli* chemotaxis pathway

Following the abstraction framework put forth in Section 3.1, we associate the original system S with the pCTMC system of F/M conformations (Equations 7), along with the OU methylation process in Equation 6. The input starting state $\mathbf{s}_0$ is the last F/M state of the pCTMC, and the last methylation level m. The simulation time T is the variable Δt from Section 5.1, also used for the integration step-size of the OU in the Euler-Maruyama scheme. The transition rates $k_{\pm}$ are calculated using the variables m and L, the last methylation level and external ligand concentration at the position of the cell, respectively. The output of this system, $\mathbf{s}_t$, is then a sampled pCTMC trajectory and new methylation level. Finally, the run property (8) is evaluated on (the last state of) the drawn pCTMC trajectory and the output determines whether the cell 'runs' or 'tumbles'.

In observing the truth value of property ϕ_{RUN} for the state of the pCTMC at regular intervals of Δt, we cast the original pCTMC model (S) into a DTMC (Figure 2). This DTMC has only two states, $\phi_{\text{RUN}} \in \{\top, \bot\}$, and transition probabilities depending on the transition rates $k_{\pm}$, μ, of the original pCTMC.

Since this is only a two-state DTMC, the state at the next time-step conditioned on the current one can be modelled as a Bernoulli random variable:

$$\phi' \mid \phi \sim \mathit{Bernoulli}(p = p_{\phi'=1|\phi}(m, L)), \tag{9}$$

where $\phi,\ \phi'$ are the ϕ_{RUN} DTMC states at time-steps $h,\ h+1$ respectively. Also, the boolean $\{\bot, \top\}$ truth values of the properties have been mapped to the standard corresponding integers $\{0, 1\}$ for mathematical ease.

We recognise that a *single step* transition of this DTMC $(\phi' \mid \phi, m, L)$ is the output $y' \mid y, \mathbf{q}$ produced by the abstracted layer $\tilde{S}(y, \mathbf{q})$. Identifying the corresponding probability function $p_{\phi'=1|\phi}(m, L)$ as the underlying governing function $\Psi(y, \mathbf{q})$ completes the setting of *E. coli* chemotaxis model abstraction to the methodology framework given above (Section 3.1). Note that the OU process for methylation is retained in the abstracted model as a parallel running process in the same layer of the multi-scale system. The OU process output m, together with the ligand concentration L (output of a different layer in the multi-scale system), constitute the input $\mathbf{q}$. The altered simulation scheme for this abstracted model is outlined in Algorithm 2 (Appendix B). Notice how Steps 5, 6 there replace the more expensive Steps 22, 23 in Algorithm 1 (Appendix B).

Philosophical remark. One may observe that our method requires choosing which parts of the model to abstract using our framework, and this is at the modeller's discretion. In this case, for instance, asking of the method to abstract a large pCTMC modelling the methylation process might be feasible, but redundant, as we already know of a very efficient abstraction for it: the Langevin SDE for the OU process. It is therefore beneficial and desirable to aid the method where possible because we have particular insight. This agency reflects our focus on inquiring whether a particular interpretation of an accurate micro-scale model may provide a useful mechanism for observed macro-scale behaviour, especially in areas where domain knowledge is lacking. In this sense, the nature of the attempted abstraction puts different questions to the model.

Constructing Ψ *in* E. coli *chemotaxis.* A central part of our abstraction methodology is estimating an underlying probability function Ψ, which is used to produce stochastic output. In our *E. coli* example model, a single DTMC transition $(\phi' \mid \phi, m, L)$ corresponds to the output $y' \mid y, \mathbf{q}$ produced by the stochastic mapping $\tilde{S}(y, \mathbf{q})$. Therefore, $\tilde{S}(\phi, (m, L))$ consists of sampling from a Bernoulli distribution $\mathit{Bernoulli}(p = p_{\phi'=1|\phi}(m, L))$ where $p_{\phi'=1|\phi}(m, L)$ is the underlying probability function $\Psi(y = \phi,\ \mathbf{q} = (m, L))$ in the general formalism.

We approximate $\Psi(y, \mathbf{q}) = p_{\phi'=1|\phi}(m, L)$, using GPs trained on observations from *micro-trajectories*, i.e. trajectories of the fine F/M pCTMC system which are then mapped onto the property space, $\phi \in \{0, 1\}$, to serve as training data. The nature and training of the GPs is described in 3.2. Note that the Bernoulli distribution likelihood, used here for Gaussian process classification (GPC), is a special case result because of both the binary $y = \phi$ output and the single observation of transitions at a particular (m, L) parametrisation.[2] Lifting these restrictions would result in the more general multinomial distribution likelihood.

Observations are gathered from simulation of the original system, and are therefore generated as follows. At a given (m, L) the pCTMC with transition rates $k_{\pm}(m, L)$ is at a state $\mathbf{s}_0$ which maps onto $\phi(\mathbf{s}_0)$. After a time Δt, the same CTMC is found at a state $\mathbf{s}_{\Delta t}$, which maps onto $\phi(\mathbf{s}_{\Delta t})$. An observation $\phi(\mathbf{s}_{\Delta t}) \mid \phi(\mathbf{s}_0), m, L$ is in this way recorded for every parametrisation (m, L) the bacterium has visited in the micro-trajectories.

Since the output of $\tilde{S}$ is binary $(y = \phi \in \{0, 1\})$ we construct two probability functions $\Psi_\phi(m, L) = p_{\phi'=1|\phi}(m, L)$. Each is approximated with a separate GPC function, where $\Psi_0(m, L)$ is trained on observations of transitions originating from the 'TUMBLE' state $(p_{\phi'=1|\phi=0}(m, L))$ and $\Psi_1(m, L)$ using transitions from the 'RUN' state $(p_{\phi'=1|\phi=1}(m, L))$. Notice that we need not estimate separate functions for $\phi' = \{0, 1\}$, since $p_{\phi'=1|\phi}(m, L) = 1 - p_{\phi'=0|\phi}(m, L)$. Having access to these underlying

[2]It is highly unlikely to have more than a single transition since $(m,\ L)$ are continuous values that constantly change for the bacterium.

probability functions we are now able to sample the DTMC at any parametrisation (m, L) the bacterium finds itself in, by using the function estimate for $p_\phi(m, L)$ despite not having observations at that m, L.

The function $p_{\phi'=1|\phi}(m, L)$ is particularly challenging for GPs. This is due to a sharp boundary in the $m,\ L$ domain, where there is a transition from $p_{\phi'=1|\phi}(m, L) \approx 0$ to $p_{\phi'=1|\phi}(m, L) \approx 1$. The bacterium has a steady state very close to this boundary, determined by the motor bias mb_0, and that is where they are most often found. Therefore, accurate estimation of this boundary is crucial for this problem. Furthermore, the low probability of finding bacteria away from the boundary (in a relatively smooth ligand field) gives a very narrow window of where the function is observed. To get a better overall estimate, we sporadically perturb the position of bacteria in the micro-trajectory phase of collecting observations, such that the bacterium finds itself producing observations away from the boundary for a while, before the system returns close to steady state again. Despite these difficulties, we produce a good reconstruction of the underlying functions $p_{\phi'=1|\phi=0}$ and $p_{\phi'=1|\phi=1}$ over the $m,\ L$ domain (see Figure 3).

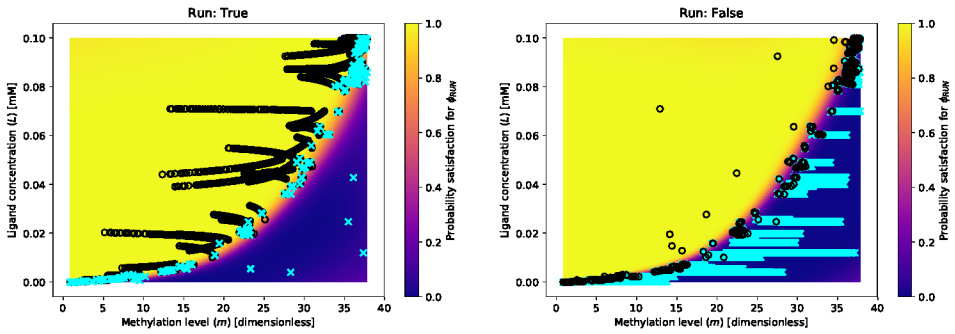

Fig. 3. The probability functions $p_{\phi'=1|\phi}(m, L)$ (left: $\phi = \top$, right: $\phi = \bot$) produced by the GP with hyperparameters $\ln(\ell) = (3.5, -2.5)$ and $\ln(\sigma) = 5$, 100 inducing points (FITC approximation), and 10,000 observations (black circles for $\phi' = \top$, cyan crosses for $\phi' = \bot$). The steep boundary is accurately captured, producing a sharp, switch-like transition from the run domain to the tumble domain.

The function estimates shown in Figure 3 were obtained by using information from 10,000 simulation runs to construct a data set for GP regression. It is an interesting question how robust our results are against changes in the choice of number of training simulations. To assess this, we repeated the full analysis (including the one to be discussed in the next section) with 5,000, 2,000 and 1,000 training points. This exercise resulted in very similar overall results for 5,000 training simulations; however, for 2,000 and 1,000 simulations a significant deterioration of the approximation quality became apparent.

While it is difficult to *a priori* decide on the number of training points, a useful empirical diagnostic can be obtained by observing plots of the probability function $p_{\phi'=1|\phi}(m, L)$. Figure 4 shows clearly that, while the left-hand plot (corresponding to 5,000 training points) still retains a good approximation of the probability function, the middle and particularly right-hand panels (corresponding to 2,000 and 1,000 points respectively) give an inaccurate reconstruction where the areas far from the boundary region reverts heavily to the prior GP mean. The latter is due to a lack of sufficient observations belonging to the minority class (these are the cyan crosses for the plotted $p_{\phi'=1|\phi=1}(m, L)$ in Figure 4); the $p_{\phi'=1|\phi=1}(m, L) \approx 1$ domain, mostly populated by positive observations (black circles), is affected to a lesser degree.

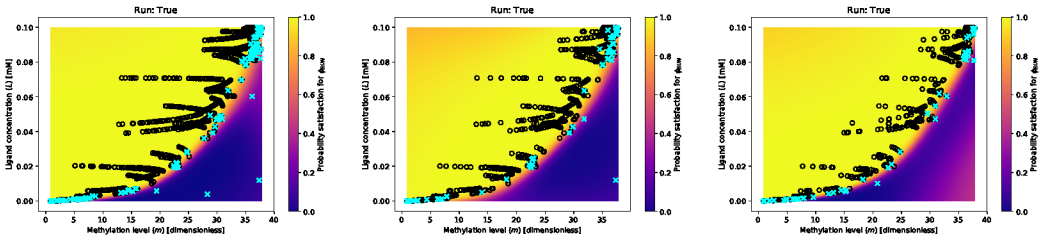

Fig. 4. Sensitivity of the estimate of the probability function $p_{\phi'=1|\phi=1}(m, L)$ when varying the number of training points (left, 5,000, middle, 2,000, right, 1,000). The middle and left figures clearly show insufficient exploration of the property space, resulting in an inaccurate estimation of the probability function. Particularly and especially away from the transition boundary, there are not enough cyan observations causing the function to revert to the mean prior (observe colour difference near the edges).

5.4 Results

When assessing performance of our method for statistical abstraction, there are two things of interest: accuracy and computational savings. Accuracy refers to how similar behaviour of the abstracted system is to the behaviour of the original system. In our case of chemotaxis in *E. coli*, this is seen by comparing population distributions in a ligand field, resulting from simulations using the original fine system and the abstracted one. We also compare run and tumble duration distributions as another metric of how closely we approximate the output and behaviour of the original model.

Learning the transition probability functions for the dual-state DTMC enabled us to simulate bacteria using our abstracted model on a host of different ligand field profiles. Beyond comparing bacteria population distributions under the original Gaussian ligand field used for learning (see L_1 below), we did the same for a linear and dynamic field (L_2, L_3 below), using the same learned functions $p_{\phi'=1|\phi}(m, L)$, $\phi \in \{0, 1\}$.

The ligand fields tested were:

$$L_1(\vec{r}) = 0.1 \cdot \exp\left[-0.5(\vec{r}^{\top}\Sigma^{-1}\vec{r})\right], \qquad \Sigma = 3 \cdot \mathbf{I}_2; \tag{10}$$

$$L_2(\vec{r}) = \max\left(10^{-5},\ 0.1 - 0.05\sqrt{(\mathbf{A}\vec{r})^{\top}\mathbf{A}\vec{r}}\right), \qquad \mathbf{A} = \begin{pmatrix} 1/5 & 0 \\ 0 & 1/2 \end{pmatrix}; \tag{11}$$

$$L_3(\vec{r}, t) = 0.1 \cdot \exp\left[-0.5(\vec{r}^{\top}\Sigma(t)^{-1}\vec{r})\right], \qquad \Sigma(t) = 3(t/50 + 1) \cdot \mathbf{I}_2. \tag{12}$$

In the fields above, the maximum value is 0.1 (units are mM) and this peak concentration is at $\vec{r} = (0, 0)$. The field L_2 is a static, non-isotropic, linear field, whereas L_3 is a dynamic field: a Gaussian spreading out over time, similar to what one might expect to be produced by a diffusing drop of nutrients. As expected, as long as the stimulus concentrations and their spatial gradients are within the region observed in training, the population distributions show consistency with those produced when simulating using the original full model (see the discussion on *accuracy evaluation* below, as well as Table 1 and Figure 5).

Computational cost savings. Computational savings are given empirically here by comparing running times of simulations for both systems. A hundred (100) cells are simulated in each of the ligand fields, for a time $t_{\text{end}} = 500$s and a time-step of $\Delta t = 0.05$. Therefore, one million (1000000) iterations of the main while loop in Algorithms 1, 2 (Appendix B) are compared in the reported speed-up factor (Table 1). We observe a speed-up factor of ~ 8, reducing running times from $\sim$ 460m to $\sim$ 60m. Table 1 reports speed-up factors for each ligand field experiment.

The reported factor values do not include the costs paid for training the GP and producing the training data. It takes ~ 4min to train GPs for both Ψ_ϕ functions, and ~ 10min for producing 20000 observations of pCTMC transitions from the original fine system (10000 training points for each Ψ_ϕ function). The relatively low times compared to simulation times, combined with the fact that one only pays this once, upfront, make these costs negligible.

Accuracy evaluation. To evaluate how closely results from the abstracted model are compared to the original one, we applied the Kolmogorov-Smirnov (KS) two-sample test [Chakravarty et al. 1967] to the population distributions of the two models at several time-points in the simulation, as well as to the distributions of running and tumbling duration. We have 100 samples from each population distribution since we simulated 100 cells. However, in the case of 'Run' and 'Tumble' duration distributions we have ~ 60000 observations from each, because we aggregate observations from the entire trajectory; we choose a random 1000 sample of these to perform the KS test.[3] In light of these difficulties, a different test which quantifies the distance between the two distributions (e.g. Jensen-Shannon divergence) might be more useful here, but that requires analytic forms of the distributions.

Inspecting Table 1 we find no KS distance higher than 0.2 indicating very similar distributions, as supported by the associated high p-values. The latter do not allow rejecting the null hypothesis with the current sample, which is that the samples originate from the same distribution. An exception is the 'Tumble' duration distributions in the L_1 ligand field, where the somewhat higher KS distance of the large sample sizes gives an exaggerated p-value (see footnote 3).

We note how even in the case of the dynamic L_3 field, the resulting population behaviour of the abstracted model is preserved without any additional training necessary. The fact that the original training occurred in a static field does not affect the ability of the abstract model to cope with a dynamic one.

Table 1. KS two-sample test statistics, where the first (top) value reports KS distance and the second (in brackets, bottom) the associated p-value. One sample came from 100 trajectories of fine *E. coli* system simulations, and the other from 100 abstracted system simulations. The first four columns show KS test results of original and abstracted bacterial population distances from peak concentration at various times t (shown in Figure 5). 'Run' and 'Tumble' columns compare the distributions of run and tumble durations respectively for 1000 samples from each system. The last column reports the observed speed-up factor based on running times and normalising for core utilisation.

Field	$t = 125$s	$t = 250$s	$t = 375$s	$t = 500$s	**Run**	**Tumble**	**Speed-up factor**
Gaussian:	0.110	0.160	0.170	0.160	0.039	0.101	
$L_1(\vec{r})$	(0.556)	(0.140)	(0.099)	(0.140)	(0.425)	$(7 \cdot 10^{-5})$	7.8
Linear:	0.010	0.150	0.170	0.130	0.022	0.014	
$L_2(\vec{r})$	(0.677)	(0.193)	(0.100)	(0.344)	(0.967)	(0.100)	9.4
Dynamic	0.140	0.070	0.140	0.080	0.047	0.039	
Gaussian: $L_3(\vec{r}, t)$	(0.261)	(0.961)	(0.261)	(0.894)	(0.214)	(0.425)	8.9

[3] We sub-sample because the KS test p-value depends heavily on sample size. Even if two distributions generating samples might be very close, in the limit of an infinite sample size one approaches the true distributions. In such a case, the KS test will reject that the two samples were produced by *the same* distribution, returning lower p-values as sample size increases (for the same KS distance). We do not expect to produce the same distributions here since we are making approximations, so comparing p-values for very large sample sizes is not of interest.

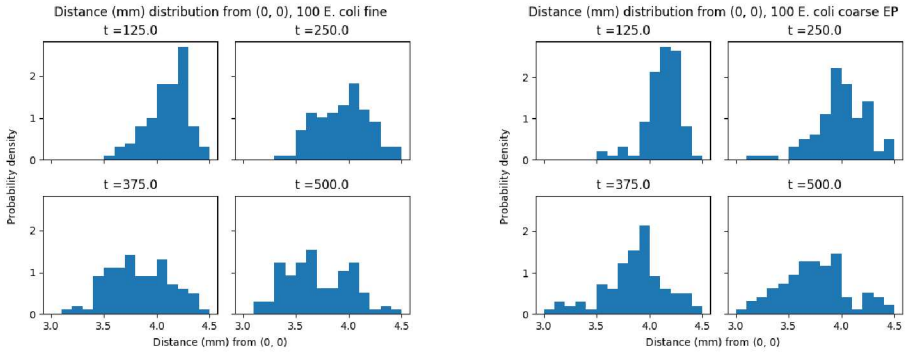

Fig. 5. Empirical distributions for the distance of 100 bacteria from peak of L_1 ligand field at different times t of the simulation. Left: original full system simulations. Right: abstracted system simulations.

6 CLOSING THE INFORMATION LOOP

6.1 Introduction

The *E. coli* model we abstracted had a one-way flow of information: the environment affected the agent state, but the agents had no effect on the state of the environment layer. Here, we demonstrate the method on a model which closes this information loop by having agents influencing their immediate environment (local interactions) which additionally allows for agent-agent interaction with the environment layer acting as a conduit for signalling. *Dictyostelium discoideum* (commonly *slime mould*) is a species of amoeba which naturally displays such behaviour, making models thereof a suitable choice for our abstraction framework. These single-cell organisms live in colonies, where they are in close proximity to many other members of the species. They exhibit agent-agent interaction through emission and response to a particular chemical. This triggers a chemotactic response which enables the amoebas to aggregate into multi-cellular groups, which is beneficial to their survival under starvation conditions, as well as a natural part of their reproductive cycle.

The chemotactic behaviour of the amoebas exhibits all the hallmarks of a multi-level spatio-temporal system where agents can also communicate with each other through environmental signals. In keeping with our general philosophy, our angle of attack is to choose a detailed model of the individual cell's motility response and abstract it, so that the emergent aggregation behaviour in the population is still observed. Extensive observations at cell level have shown that *D. discoideum* move by extending tentacle-like cell wall deformities called *pseudopodia*, and shifting themselves in that direction [Bosgraaf and Haastert 2009; Haastert 2010]. Based on this general principle, there are various approaches to model the trajectories produced by such a mechanism. Here, we opt to work with discrete models where the cells take indivisible steps towards a particular direction. Other approaches exist where the cell is taken to perform continuous stochastic motion with correlations, as in [Li et al. 2011].

It has been further well observed that in the absence of environmental stimuli, the angle at which the next pseudopodium is extended relative to the current one follows a symmetric probability distribution over $(-\pi, \pi)$, centred at 0. This causes the amoebae to explore their environment by performing isotropic random walks in the long run. However, the angle probability distribution shifts in the presence of a cyclic Adenosine Mono Phosphate (cAMP) chemical gradient to induce a drift in the random walk aligned to the gradient. When *D. discoideum* cells fall into starvation they emit this chemical which diffuses through space and reaches other nearby amoebae [Haastert and Bosgraaf 2009; Robertson and Grutsch 1981]. In response, the sensing amoebae emit more cAMP

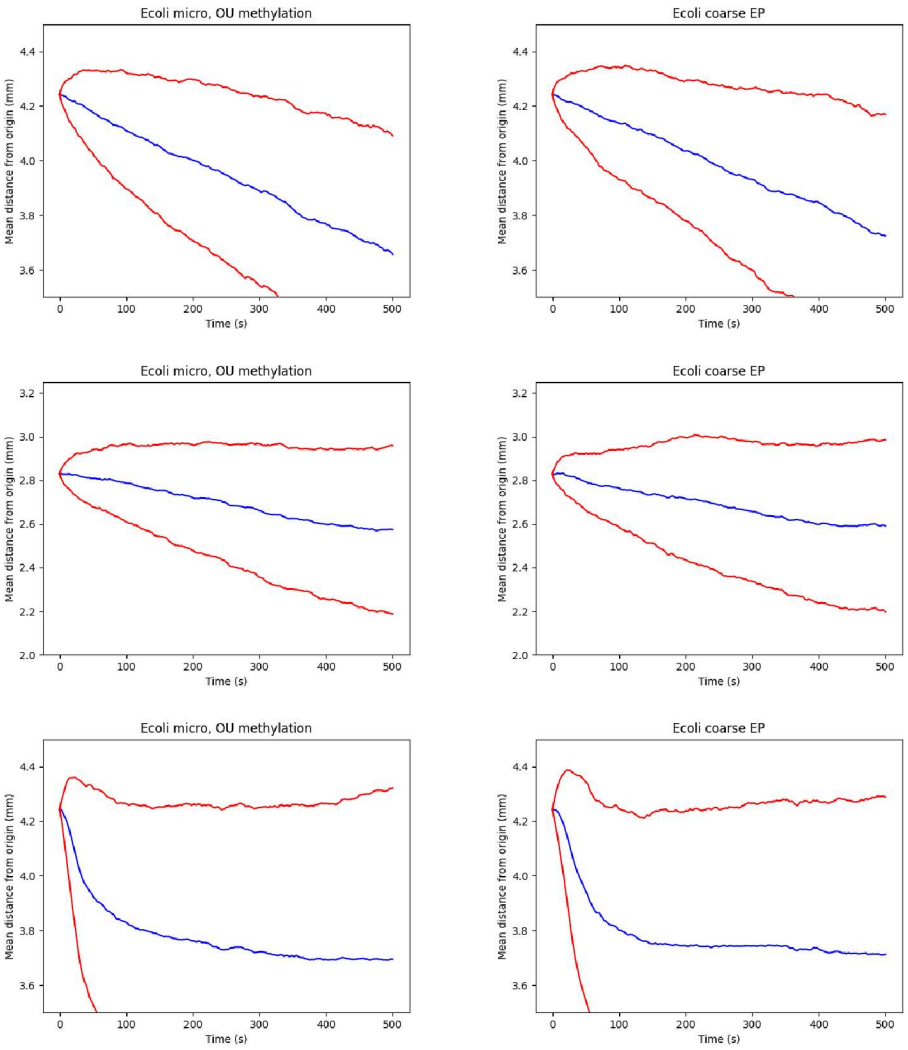

Fig. 6. Average (blue) and standard deviation (red) of distance from peak ligand concentration for a population of 100 *E. coli* over a time of 500s. Left: original full system simulations. Right: abstracted system simulations. Rows (top to bottom): L_1, L_2, L_3 ligand fields respectively.

which relays the signal further, and also begin to chemotax towards the cAMP source. The amoebae eventually congregate into a *multicellular pseudoplasmodium* — a slug-like structure which behaves as a single organism, giving the population under the starvation conditions a better chance of survival. The slug collective moves as one to find environmentally favourable conditions, where it settles and begins a reproductive process [Robertson and Grutsch 1981].

The model we abstract is an amalgamation of two other models, one focusing on the internal workings of the amoeba to induce motility by [Eidi 2017], and the other focusing on cAMP emission and response cycles by [Calovi et al. 2010]. The former provides a discrete-time Markov chain (DTMC) formalism where directions in which the cell moves are states of the chain, and the

transition probabilities depend on the sensed cAMP gradient. The latter provides an efficient method to compute the cAMP chemical field at a point in space and time $\gamma(\mathbf{x}, t)$, by utilising a Green's function (GF) method to solve the diffusion equation with degradation. Having melded the two models, we verify that the aggregation behaviour between the agents under cAMP emission is consistently observed and show that it is preserved in the abstraction.

We postulated that the motion of the amoeba cells can be well approximated by a simple Markovian process: a Bernoulli trial which determines the alignment of the next pseudopodium to be extended with respect to the cAMP gradient, based on the latest kind of pseudopodium (split/ de novo) and its alignment to the direction of the cAMP gradient. Pseudopodia can either be a result of *splitting* the current pseudopodium, or a *de novo* extension unrelated to the current one. The two kinds induce probability distributions over the possible extension angles of the next pseudopod with respect to the cAMP gradient. Since this dynamics is more complex than the simple *run/tumble* dynamics of the *E. coli* model, and the agent layer model in [Eidi 2017] is already considerably simplified, we do not expect the abstraction to yield significant computational gains. Nevertheless, it is still a useful proof of principle of our methodology, and illustrates how additional insights can be gained through identifying the necessary properties for preserving the qualitative behaviour of this model, which rests upon interaction between agents. We find that we retain the influence of agents on the environment and observe that the agent-environment-agent communication results in the macro-scale behaviour of aggregating amoebae for the abstracted model as well, albeit with some loss in accuracy.

6.2 Original model

The original model consists of two layers, the environment layer and the internal agent (*D. Discoideum* cell) layer. The two are coupled such that output from one layer is the input to the other layer and vice versa. The environment layer takes as input the cAMP emission history from each amoeba cell, and evaluates cAMP concentration and its gradient at all agent positions. This serves as input to the internal agent layer, which picks a direction for the cell to move and updates its position accordingly; it also updates the cAMP emission history for the cell with the latest cAMP emission value.

Environment layer. The model laid out in [Calovi et al. 2010] for cAMP diffusion in space and the corresponding methodology is taken as the environment layer we use. Following the paper, we describe cAMP diffusion with degradation and emitting sources through the coupled differential equations of [Martiel and Goldbeter 1987] (MG equations):

$$\partial_t \gamma(\mathbf{x}, t) = \frac{k_t}{h}\beta(\mathbf{x}, t) - k_e\gamma(\mathbf{x}, t) + D\nabla^2\gamma(\mathbf{x}, t), \tag{13}$$

$$\beta(\mathbf{x}, t) = \sum_{j=1}^{N} \beta_j(t) \exp\left[-\frac{4}{\sigma^2}(\mathbf{x} - \mathbf{x}_j)^2\right], \tag{14}$$

$$\frac{d\beta_j}{dt} = \phi(\rho_j, \gamma_j) - (k_i - k_t)\beta_j, \qquad \frac{d\rho_j}{dt} = f_2(\gamma_j)(1 - \rho_j) - f_1(\gamma_j)\rho_j, \tag{15}$$

where f_1, f_2, ϕ, and Y_j are defined as:

$$f_1(\gamma_j) = \frac{k_1 + k_2\gamma_j}{1 + \gamma_j}, \qquad f_2(\gamma_j) = \frac{k_{-1} + \lambda_1 k_{-2}\gamma_j}{1 + \lambda_1\gamma_j}, \tag{16}$$

$$\phi(\rho_j, \gamma_j) = \frac{\lambda_2 + Y_j^2}{\lambda_3 + Y_j^2} \times 1800, \qquad Y_j = \frac{\rho_j\gamma_j}{1 + \gamma_j}, \tag{17}$$

and $\mathbf{x}_j$ is the location of the jth amoeba in Cartesian coordinates. The cAMP concentration field is given by $\gamma(\mathbf{x}, t)$, which evolves according to the partial differential equation (PDE) 13. The term $\beta(\mathbf{x}, t)$ describes the emission of cAMP from every cell, and $-k_e\gamma(\mathbf{x}, t)$ models the chemical's natural degradation. Constants used in the above equations are fixed (Table 2 Appendix C, as given in [Calovi et al. 2010]).

The last couple of differential equations (Eqs. 15) describe intracellular concentration and the ratio of active cAMP receptors of the jth amoeba respectively. These are intracellular processes independent within each cell, and so we implement them in the internal agent layer of our model.

To solve Equation 13, we implement a Green's function method [4] as in the paper, and produce the solution

$$\gamma(\mathbf{x}, t) = \sum_{j=1}^{N} \int_0^t \frac{c_j(s)}{2^{2d}} \exp\left[-k_e(t-s)\right] \prod_{k=1}^{d} \left[\operatorname{erf}\left(\frac{l_{j,k} + R_\sigma}{\sqrt{4D(t-s)}} \right) - \operatorname{erf}\left(\frac{l_{j,k} - R_\sigma}{\sqrt{4D(t-s)}} \right) \right] ds, \tag{18}$$

where $c_j(s) = \frac{k_t}{2h}\beta_j(s)$ is the amount of cAMP created by the jth amoeba at time $t-s$, and $l_{j,k}$ is the distance between jth amoeba and $\mathbf{x}$ on the k Cartesian coordinate.

Despite the integration in time in Equation 18 which necessitates the use of numerical integration techniques, it is still a more efficient solution to calculate than alternative finite element techniques for integrating the PDE 13. The latter require discretisation of space to a fine resolution (comparable to amoeba cell size) and would scale accordingly, whereas Equation 18 scales linearly with the number of amoebae. Additionally, the natural degradation of cAMP allows us to only keep a finite history of the emission from every agent, and limits the integration time required for achieving a good enough value for the concentration field.

Agent layer. The intracellular set of processes takes as input the concentration and gradient of cAMP at the cell's location and determines the cell's cAMP emission rate and its direction for the next Δt time step. The MG Equations 15 model the emission rate evolution and are solved in a forward Euler manner as shown below.

Formally, the agent layer comprises of a set of equations:

$$\mathbf{s}^t = S(\mathbf{s}^{t-1}, \nabla\gamma_j^{t-1}), \tag{19}$$

$$\beta_j^t = \beta_j^{t-1} + \Delta t \left[\phi(\rho_j^{t-1}, \gamma_j^{t-1}) - (k_i - k_t)\beta_j^{t-1}\right], \tag{20}$$

$$\rho_j^t = \rho_j^{t-1} + \Delta t \left[f_2(\gamma_j^{t-1})(1-\rho_j^{t-1}) - f_1(\gamma_j^{t-1})\rho_j^{t-1}\right], \tag{21}$$

where the last two equations are forward Euler methods for Eqs. 15, the superscripts $t \in \mathbb{N}$ denote time steps, and S is a step on a Markov chain. The Markov chain has state space $I = \{(s_1, s_2)\}$ where $s_i \in \{0, 1, 2, 3, 4, 5\}$ signifies the angle $\theta = s_i(2\pi/6)$ of a step taken by the cell, with respect to the horizontal axis in a 2D space. The state space consists of tuples of step directions since both the latest step and the one taken before it are necessary to correctly describe the cell's motion. The transition probabilities satisfy the condition $s_2^t = s_1^{t-1}$ that the second element of the tuple is the previous step direction. They also shift according to $\theta_{rel}(s_i, \nabla\gamma)$, the angle between a potential step direction and the cAMP gradient $\nabla\gamma$, in order to bias the next step direction to align with the cAMP gradient. As modelled in [Eidi 2017], the bias is a linear superposition of the term $\epsilon\cos(\theta_{rel}(s_i, \nabla\gamma))$ on the four s_i directions other than the latest step direction of the cell and the

[4] A well-known method for solving a partial differential equation (PDE), the GF is the inverse of the linear differential operator in the PDE; it is therefore the solution $G(x, y, t, s)$ of the PDE at (x, t) when driven by a point source $\delta(x-y, t-s)$. In the case of the diffusion equation with degradation, the solution for a point source is analytically known. Since the operator is linear, one can then reconstruct the solution to the PDE with the actual source $f(x, t)$ by superposition of the GF solutions for every point source in the domain of space and time — i.e. computing $\int dyds G(x, y, t, s)f(y, s)$. See [Butkov 1995] for a comprehensive exposition.

one directly opposite it. Picking an appropriate $\epsilon = 0.04$, we retain the probability conditions for the transition probabilities of the Markov chain, $0 \leq p_{ik} \leq 1$ and $\sum_j p_{ik} = 1, \forall ik$. The result is transition probabilities $p_{ik} = p^0_{ik} + \epsilon \cos(\theta_{rel}(s_k, \nabla\gamma))$ where p^0_{ik} are the unbiased transition probabilities. Note that these states are tuples of step directions, and so the difference between consecutive steps is relevant in determining probabilities for the next step direction.

Simulation scheme. We perform simulations of the original model presented above with the following set-up: we initialise 250 agents at random positions drawn from a uniform distribution over a 0.0625mm^2 square with centre $(0, 0)$. We set $\Delta t = 0.3\text{m}$ and iterate between executing processes in the environment layer (evaluating the cAMP concentration and gradient at the agents' positions), and executing processes in the internal agent layer (updating the agents' positions and cAMP emissions). The simulation ends at $t_{end} = 30\text{m}$, at which point the agents have congregated into one or more clusters. Initial values for the internal agent parameters are $\beta^0_j = 0$, and ρ^0_j drawn from a uniform distribution $U[0, 1]$ for each j agent.

6.3 Abstracted model

Our aim is to find appropriate logical properties evaluated on the states such that their satisfaction probabilities can be used to correctly recreate the motility characteristics of *D. discoideum* cells. Guided by the biological theory used to craft the model by Eidi, we formalise the abstracted internal agent model with the following equations:

$$\mathbf{s}^t = \tilde{S}\left(\mathbf{s}^{t-1}, \mathbf{s}^{t-2}, \nabla\gamma_j^{t-1}\right), \tag{22}$$

$$\text{where} \qquad \tilde{S}\left(\mathbf{s}^{t-1}, \mathbf{s}^{t-2}, \nabla\gamma_j^{t-1}\right) = \mathit{Categorical}(p_\phi); \tag{23}$$

$$\phi \sim \mathit{Bernoulli}\left(\Psi_y\left(\theta_{rel}\left(s_1^{t-1}, \nabla\gamma_j^{t-1}\right)\right)\right), \tag{24}$$

with Bernoulli outcomes 1 for $|\theta_{rel}(s_1^t, \nabla\gamma_j^{t-1})| < \pi/2$ (extension will be aligned to cAMP gradient), or 0 otherwise (not aligned); and

$$y = \begin{cases} 1 & \text{for } |\theta_{rel}(s_1^{t-1}, s_2^{t-1})| < \pi/2 \text{ (split extension)}, \\ 0 & \text{otherwise (de novo extension)}. \end{cases} \tag{25}$$

The above describes the process of receiving an input $(\mathbf{s}^{t-1}, \nabla\gamma_j^{t-1})$ and going through the steps of: (1) evaluating whether the current pseudopod extension is of a split or de novo nature ($y \in \{1, 0\}$); (2) using the appropriate learned stochastic function Ψ_y to sample whether the next pseudopod will be within $\pi/2$ rad from the cAMP gradient direction (ϕ); and finally (3) determining the direction of the next pseudopod by picking from a categorical distribution of the possible directions with event probability vectors p_ϕ. The vectors p_ϕ are constructed as follows:

$$p_\phi = [r_1, \cdots, r_6], \tag{26}$$

with elements

$$r_i = \begin{cases} 1/Z & \text{if } \left(i \neq s_1^{t-1}\right) \wedge \left(\left(\phi = 1 \wedge |\theta_{rel}(i, \nabla\gamma_j^{t-1})| \leq \pi/2\right) \vee \left(\phi = 0 \wedge |\theta_{rel}(i, \nabla\gamma_j^{t-1})| \geq \pi/2\right)\right), \\ 0 & \text{otherwise}, \end{cases}$$

where Z is a normalisation constant such that $\sum_i r_i = 1$. The conditions for the elements of p_ϕ imply that two consecutive steps cannot be taken in the same direction ($i \neq s_1^{t-1}$), and that for $\phi = 1$ the step must be in an angle within $\pi/2$ rad relative to the gradient $\nabla\gamma_j^{t-1}$ and vice versa for $\phi = 0$. If these conditions are not met for a step direction i, no probability mass is given ($r_i = 0$). Note

also that the probability mass is distributed equally amongst allowed directions ($1/Z$), violating the original model's rule that a step directly opposite the last one has a constant probability of occurring ($1/21$) which is unaffected by the perceived cAMP gradient. The abstracted system is essentially a second-order DTMC with transition rates dependent on an external input, the cAMP gradient.

Finally, we retain the forward Euler method for solving the MG differential equations for cAMP emission (Equations 20, 21):

$$\beta_j^t = \beta_j^{t-1} + \Delta t \left[\phi(\rho_j^{t-1}, \gamma_j^{t-1}) - (k_i - k_t)\beta_j^{t-1}\right], \tag{27}$$

$$\rho_j^t = \rho_j^{t-1} + \Delta t \left[f_2(\gamma_j^{t-1})(1 - \rho_j^{t-1}) - f_1(\gamma_j^{t-1})\rho_j^{t-1}\right]. \tag{28}$$

As before, the probability functions $\Psi_y(\theta_{rel}(s_1^{t-1}, \nabla\gamma_j^{t-1}))$ for $y \in \{1, 0\}$ are approximated by GPs trained on original model observations (Figure 7). However, the two functions here do not correspond to the satisfaction value ϕ of the property being approximated by the GP, but a different one. Effectively we define two separate properties to achieve a good abstraction: one for whether the current pseudopod was a split or de novo extension ($y \in \{1, 0\}$), used to differentiate which stochastic function Ψ_y is used; and another of whether the next extension will be aligned to the cAMP gradient ($\phi = 1$ if s_1^t within $\pi/2$ rad of $\nabla\gamma_j^{t-1}$, 0 otherwise). Because of the two different properties, this abstraction is not simply a reduction of the original Markov chain to one with less states as the one for *E. coli* was, but rather a re-defined set of processes making use of learned stochastic functions to estimate needed probabilities for preserving the motility behaviour. As in our previous abstraction of the *E. coli* model, the same input to the layer is needed to produce the same output, so we can replace the original internal model with the abstracted one.

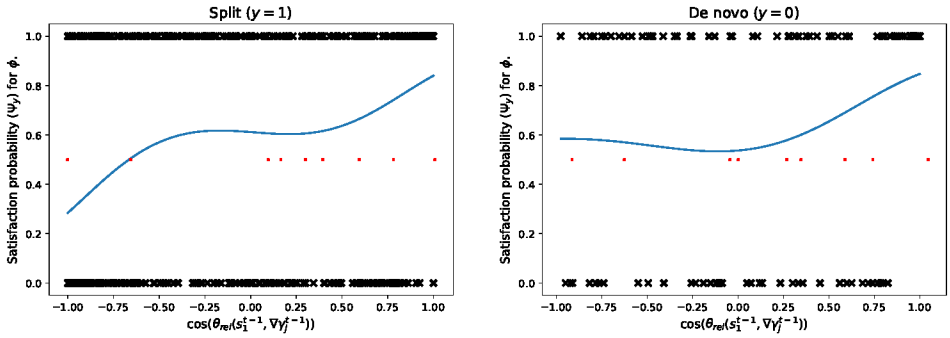

Fig. 7. $\Psi_{y=1}$ left, $\Psi_{y=0}$ right. Black crosses are observed outcomes from simulation of the original system. Red dots are the (nine) inducing points used for the FITC approximation. The curves show how the satisfaction probability of ϕ (whether the next step will be aligned to the cAMP gradient), is inferred to respond to the relative angle between the cosine of current step and the gradient: $\cos\left(\theta_{rel}(s_1^{t-1}, \nabla\gamma_j^{t-1})\right)$. As can be seen, the nature of the current pseudopod (split $y = 1$, or de novo $y = 0$) produces significantly different curves.

This *D. discoideum* model abstraction involves re-interpreting the original 30 state Markov chain producing 6 discrete possible outcomes for the internal agent layer, as chained functions. The first function takes as input the nature of the current pseudopod (split / de novo) and its angle relative to the cAMP gradient $\theta_{rel}(s_1^{t-1}, \nabla\gamma_j^{t-1})$, and stochastically determines whether the next step will be within $\pi/2$ rad of the cAMP gradient; the second function takes this decision and (stochastically) converts it to one of 6 possible directions for the next pseudopod which are

consistent with the decision. The conversion to direction in the latter function happens through categorical distributions with fixed probability vectors which do not depend on the angle θ_{rel} — i.e. the first function removes the dependence of the layer output on the relative angle and split / de novo nature of pseudopod. Given ϕ, the satisfaction of the alignment to cAMP gradient for the next step, we have all the necessary information to produce the internal layer's output.

6.4 Results

As before, we present results of both the accuracy and computational costs of the abstracted model with respect to the original one, after running twenty simulations of each model. In the original model, the internal agent layer amounted to sampling a single transition in a discrete-time Markov chain of effectively 30 states but with only 5 possible transitions from each state. Transition probabilities had to be re-evaluated before each sampling according to environmental input, but the whole layer has low computational costs. The abstracted model instead takes as input a binary satisfaction for a logical property and a continuous value $\in [-1, 1)$ and outputs one of 5 possible values (corresponding to the 5 permitted transitions from each state of the DTMC) stochastically. Due to the low computational cost of the original layer, we do not expect significant gains from the abstraction; we therefore focus on recovery of the emergent behaviour dependent on non-linear interaction effects between parts of the model which we are abstracting.

The macro-scale phenomenon we wish to retain here is the aggregation of agents, which we attempt to quantify for comparison purposes. To that end, we construct an empirical distribution of the agents' locations over space through the use of Gaussian kernel density estimators (KDE) [Bishop 2006], at various times in the evolution of both systems, as seen in Figures 8, 9. An inspection of Figures 8, 9 shows that qualitatively both the original and the abstracted model show an aggregation behaviour into a major cluster within the time-frame of the experiment. Since there is a single cluster forming and the uniform distribution was centred at $(0, 0)$, we expect that the distribution of the agents' distance from the centre is a good proxy for quantifying aggregation of the population.

To get a quantitative measure of the agreement between original and abstracted model, we pool the results of 20 simulations and compare the histogram distributions of agent distance from the origin at different time points. Figure 10, left panel, shows the first two statistics (mean and standard deviation) of the distance of the agents from the origin as a function of time for both models. This figure shows an excellent statistical agreement between the two models, even though the rate of aggregation seems slightly higher in the abstracted model than the original one. We can gain some insights into the origin of this discrepancy by looking at the histogram of agent distance from origin at the end of the simulation time, shown in Figure 10, right panel. This shows that, while the bulk and the mode of the distribution are well matched between the two models, the original model distribution has a heavier tail than the abstracted one. A potential cause of these deviations is that the choice of property matched (Eq. 24) does not contain sufficient information to precisely match the macro-scale behaviour, particularly in term of rarer events. This points to a need for automatic property construction method, which is something to be explored in future work.

Running times for a simulation were 219 ± 6**min and** 249 ± 8**min for the original and abstracted system respectively.** Experiments were run on a server machine with 48 CPU cores working in parallel. Note that most resources went towards evaluating the integral in Eq. 18 for each agent at every step. As expected, the abstraction does not yield computational gains for this model. We can attribute this to the small size of the original model and the overhead of producing predictions from a trained GP. This illustrates a fundamental trade-off within our abstraction approach: while the approach is generally applicable, the wisdom of applying such an abstraction depends on the specific model. In particular, when the internal agent layer is straightforward to

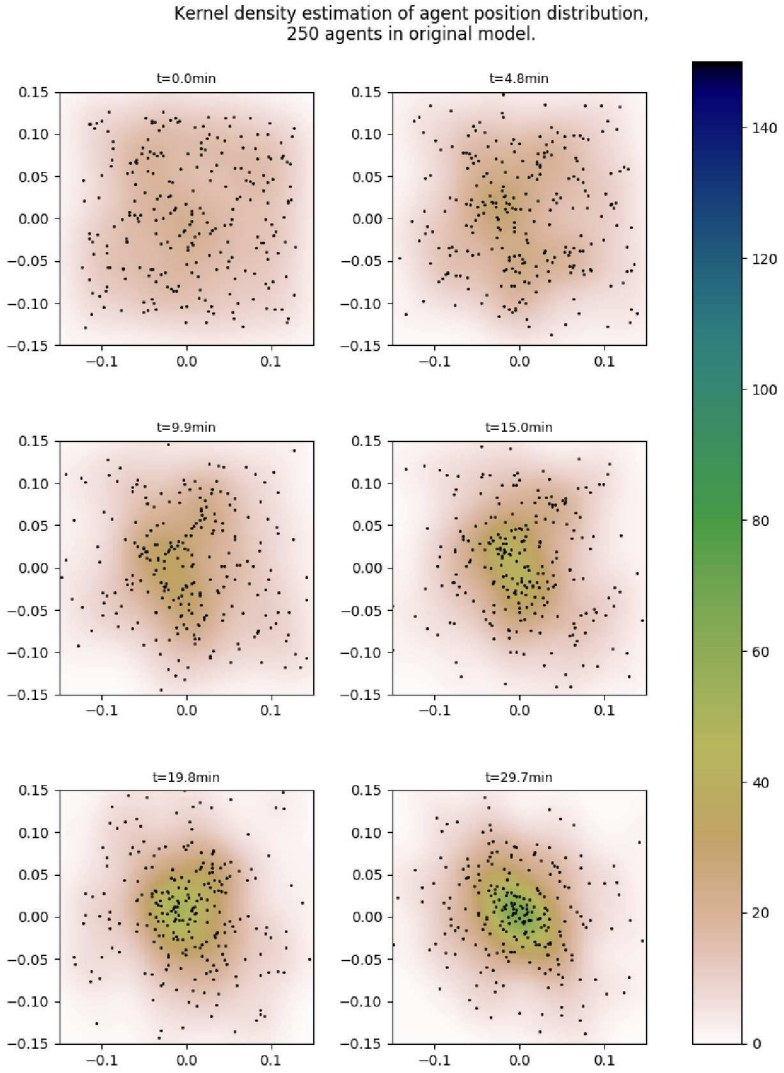

Fig. 8. Original model, 250 agents (black dots) in a 0.0625mm^2 space at different times. Background intensity field shows the KDE estimation of population distribution density. Note how the agents begin to aggregate to one major cluster by $t = 15$min.

simulate, the additional overheads of the GP abstraction can nullify the computational advantages stemming from using a simpler system.

7 DISCUSSION

In many domains, ranging from cyber-physical systems to biological and medical processes, consideration of spatio-temporal aspects of behaviour is essential. However, this comes at great

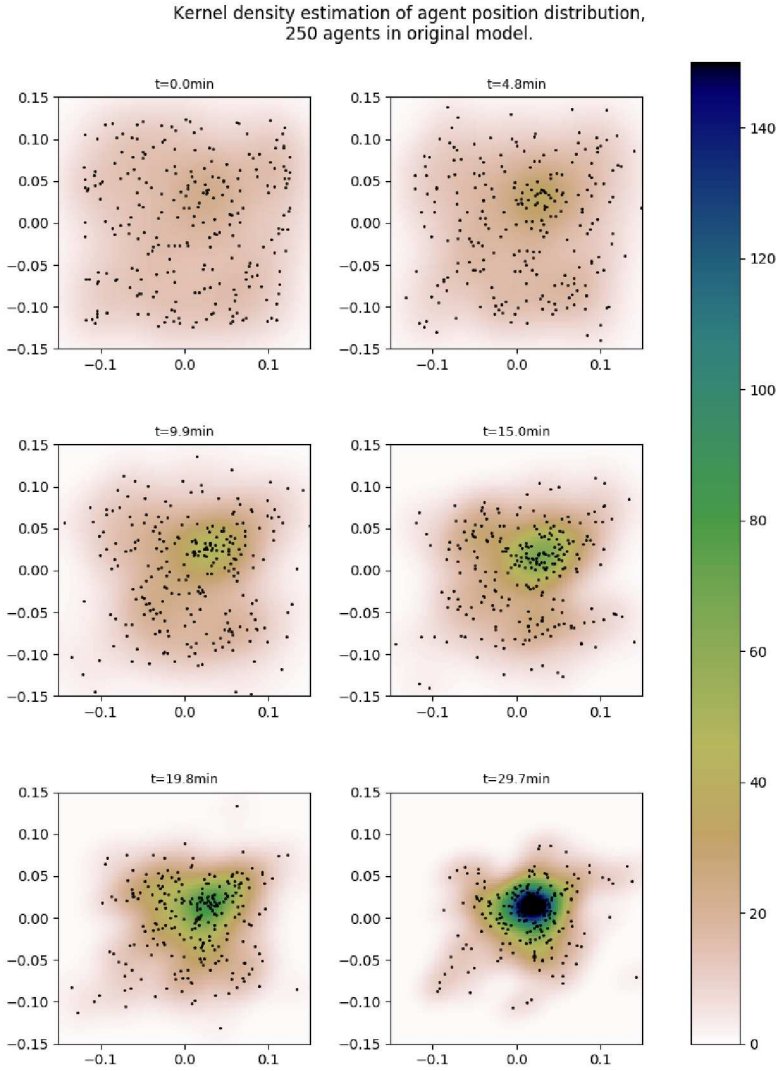

Fig. 9. Same as Figure 8 for abstracted model. Note here that the agents begin to aggregate sooner than the original model. By t = 15min, what will be the final major cluster has already formed.

computational expense. We have presented a methodology that allows layers of a computationally intensive multi-scale model to be replaced by more efficient abstract representations. This is a stochastic map, constructed based on some exploratory simulations of the full model and GP regression. Our results show that we are able to achieve significant speed-up without sacrificing accuracy. This establishes a framework for such statistical abstraction on which we plan to elaborate in future work.

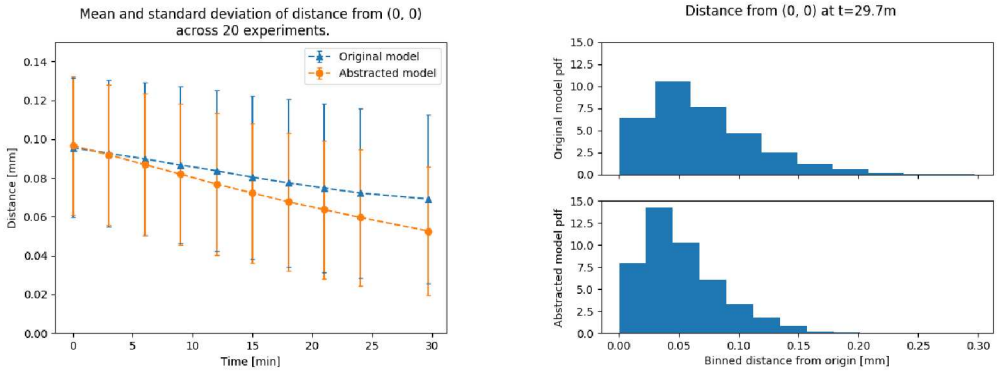

Fig. 10. Results from all 20 experiments were pooled together and treated as a single population for this analysis to increase sample size. Individual experiments are consistent with the pooled results. Left: original and abstracted model mean distance of *D. discoideum* agents from origin (blue triangles and orange circles respectively) and standard deviation (error bars) over time. It can be seen that the population moves towards the origin as the system evolves, which reflects the aggregating tendencies of both models. Deviations exist (abstracted model aggregates faster), but general behaviour is retained. Right: density histogram of the pooled populations at the last time step of the simulation. One should note that despite the closeness in median values, the heavier tail of the original model distribution is much diminished in the abstraction.

It should be noted that the specifics of the abstraction are not automatically determined by this framework, but are left to the researcher. Having to manually specify the abstraction introduces an element of flexibility, since different abstractions may be tested and so one can see which are suitable and produce accurate approximations, indicating that pertinent elements of the original model are preserved in the coarsening. Additionally, there may be various valid ways to coarsen a model, depending on what the focus of the inquiry is. On the other hand, it shifts some of the burden of abstracting the model to the researcher, who has to find a suitable set of properties which capture the output behaviour of the layer to be abstracted.

Complete automation is commonly embraced by other methodologies, where every abstraction choice may be driven by accuracy metrics. We sacrifice complete automation in this work for the ability to examine different kinds of fundamental models and assess various abstraction mechanisms. Since the era of modern science, the task has largely been to determine a minimal set of laws that concisely describe the range of phenomena observed. As such, fundamental equations at different levels of nature abound and mechanisms to bridge the various gaps, accompanied by ways to validate them, are high in demand. Our framework allows validation of an existing mechanism, but does not generate the mechanism itself — an abstraction framework which automatically produced interpretable mechanisms would solve many problems.

Our choice of application case studies illustrates well the importance that such an automated abstraction framework would have. While the *E. coli* chemotaxis model presents an obvious avenue for accurate abstraction, the *D. discoideum* model considered here does not have an obvious set of properties to act as a conduit from micro- to macro-scale behaviour. Specifically, the *D. discoideum* model consisted of a DTMC with states corresponding to directions of agent motion. Abstracting it required re-interpreting the model as motion dictated by a set of relevant properties; namely, the property of whether the cell produced a split or de novo pseudopod last, determined whether or not the next pseudopod direction would be aligned to the cAMP gradient sensed by the cell. The process of re-interpretation might not produce a particularly useful abstraction in terms of

computational efficiency, but it is valuable in understanding the processes in the layer and the relevant information flow through them.

Because the properties necessary for an accurate abstraction have to be manually defined, we can also use this abstraction framework to evaluate consistency of the original model with other higher-level models which have different assumptions. For instance, we first attempted to replace the agent layer of the *D. discoideum* model with a stochastic function which receives as input whether the current pseudopod was a split or de novo nature and produces output of whether the next one will be a result of split or de novo, akin to the *E. coli* case. This is a simpler abstracted model than the one we presented, and is supported by the literature [Haastert and Bosgraaf 2009; Li et al. 2008]. The result was a population of agents which did not display aggregation and instead retained their original uniform distribution in space, or even slightly diffused. We can therefore assert that the original model by Eidi used here cannot be cast down to a simple first order two-state discrete Markov chain (split / de novo being the two states) which has transition probabilities dependent on the output. The property of whether the next step is aligned to the cAMP gradient (beyond split / de novo) *is relevant* and cannot be discarded. This raises the question of which is the better model for *D. discoideum* motility, since other models claim to achieve similar aggregation behaviour with simple split / de novo models. Failure to reconcile models in this manner is indicative of inherent differences in the models, which may prove useful in assessing their veracity with respect to reality.

Future work avenues include, for example, allowing more properties to be expressed and using them to guide the abstraction to capture more complex behaviours. Additionally, we could infer abstracted model parameters or underlying functions from real data, instead of synthetic ones. Finally, one would ideally like to have a way to infer suitable properties for preserving a particular macro-scale behaviour. As seen in the case of the *D. discoideum* model, this is not trivial to achieve and often the properties fall short of accurately reproducing the behaviour. An automatic way to construct this properties would relieve the researcher from having to make the choice, and might reveal further insights to the models abstracted.

REFERENCES

Howard C. Berg, Douglas A. Brown, and others. 1972. Chemotaxis in Escherichia coli analysed by three-dimensional tracking. *Nature* 239, 5374 (1972), 500–504.

Christopher M. Bishop. 2006. *Pattern recognition and machine learning*. Springer, New York.

Luca Bortolussi, Dimitrios Milios, and Guido Sanguinetti. 2015. Efficient Stochastic Simulation of Systems with Multiple Time Scales via Statistical Abstraction. In *Computational Methods in Systems Biology*. Springer, Cham, 40–51. https://doi.org/10.1007/978-3-319-23401-4_5

Luca Bortolussi, Dimitrios Milios, and Guido Sanguinetti. 2016. Smoothed model checking for uncertain Continuous-Time Markov Chains. *Information and Computation* 247 (2016), 235–253. https://doi.org/10.1016/j.ic.2016.01.004

Leonard Bosgraaf and Peter J. M. Van Haastert. 2009. The Ordered Extension of Pseudopodia by Amoeboid Cells in the Absence of External Cues. *PLOS ONE* 4, 4 (April 2009), e5253. https://doi.org/10.1371/journal.pone.0005253

Eugene Butkov. 1995. *Mathematical physics* (32. print ed.). Addison-Wesley, Reading, Mass. OCLC: 258506311.

Daniel S. Calovi, Leonardo G. Brunnet, and Rita M. C. de Almeida. 2010. cAMP diffusion in '*Dictyostelium discoideum*': A Green's function method. *Phys. Rev. E* 82, 1 (2010), 011909. https://doi.org/10.1103/PhysRevE.82.011909

I M Chakravarty, R G Laha, and J D Roy. 1967. *Handbook of methods of applied statistics*. McGraw-Hill, New York, NY.

Federica Ciocchetta and Jane Hillston. 2009. Bio-PEPA: A framework for the modelling and analysis of biological systems. *Theoretical Computer Science* 410, 33-34 (Aug. 2009), 3065–3084. https://doi.org/10.1016/j.tcs.2009.02.037

Joseph O. Dada and Pedro Mendes. 2011. Multi-scale modelling and simulation in systems biology. *Integrative Biology* 3, 2 (2011), 86. https://doi.org/10.1039/c0ib00075b

Zahra Eidi. 2017. Discrete Modeling of Amoeboid Locomotion and Chemotaxis in Dictyostelium discoideum by Tracking Pseudopodium Growth Direction. *Scientific Reports* 7, 1 (Oct. 2017), 12675. https://doi.org/10.1038/s41598-017-12656-1

Nicholas W. Frankel, William Pontius, Yann S. Dufour, Junjiajia Long, Luis Hernandez-Nunez, and Thierry Emonet. 2014. Adaptability of non-genetic diversity in bacterial chemotaxis. *eLife* 3 (2014), e03526. https://doi.org/10.7554/eLife.03526

David Gilbert, Monika Heiner, Koichi Takahashi, and Adelinde M. Uhrmacher. 2015. Multiscale Spatial Computational Systems Biology (Dagstuhl Seminar 14481). (2015). https://doi.org/10.4230/DagRep.4.11.138

Daniel T. Gillespie. 1977. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry* 81, 25 (1977), 2340–2361. https://doi.org/10.1021/j100540a008

Daniel T. Gillespie. 2000. The chemical Langevin equation. *The Journal of Chemical Physics* 113, 1 (July 2000), 297–306. https://doi.org/10.1063/1.481811

John Goutsias. 2005. Quasiequilibrium approximation of fast reaction kinetics in stochastic biochemical systems. *The Journal of Chemical Physics* 122, 18 (2005), 184102. https://doi.org/10.1063/1.1889434

Peter J. M. Van Haastert. 2010. Chemotaxis: insights from the extending pseudopod. *J Cell Sci* 123, 18 (Sept. 2010), 3031–3037. https://doi.org/10.1242/jcs.071118

Peter J. M. Van Haastert and Leonard Bosgraaf. 2009. Food Searching Strategy of Amoeboid Cells by Starvation Induced Run Length Extension. *PLOS ONE* 4, 8 (Aug. 2009), e6814. https://doi.org/10.1371/journal.pone.0006814

Clinton H. Hansen, Robert Endres, and Ned Wingreen. 2008. Chemotaxis in Escherichia coli: A Molecular Model for Robust Precise Adaptation. *PLOS Comp Bio* 4, 1 (2008), e1. https://doi.org/10.1371/journal.pcbi.0040001

Eric L. Haseltine and James B. Rawlings. 2002. Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. *The Journal of Chemical Physics* 117, 15 (2002), 6959–6969. https://doi.org/10.1063/1.1505860

N. G. van Kampen. 1961. A POWER SERIES EXPANSION OF THE MASTER EQUATION. *Canadian Journal of Physics* 39, 4 (April 1961), 551–567. https://doi.org/10.1139/p61-056

T. G. Kurtz. 1971. Limit Theorems for Sequences of Jump Markov Processes Approximating Ordinary Differential Processes. *Journal of Applied Probability* 8, 2 (1971), 344–356. https://doi.org/10.2307/3211904

Liang Li, Edward C. Cox, and Henrik Flyvbjerg. 2011. 'Dicty dynamics': Dictyostelium motility as persistent random motion. *Phys. Biol.* 8, 4 (2011), 046006. https://doi.org/10.1088/1478-3975/8/4/046006

Liang Li, Simon F. Nørrelykke, and Edward C. Cox. 2008. Persistent Cell Motion in the Absence of External Signals: A Search Strategy for Eukaryotic Cells. *PLOS ONE* 3, 5 (May 2008), e2093. https://doi.org/10.1371/journal.pone.0002093

Jean-Louis Martiel and Albert Goldbeter. 1987. A Model Based on Receptor Desensitization for Cyclic AMP Signaling in Dictyostelium Cells. *Biophysical Journal* 52, 5 (Nov. 1987), 807–828. https://doi.org/10.1016/S0006-3495(87)83275-7

Michalis Michaelides, Jane Hillston, and Guido Sanguinetti. 2017. Statistical Abstraction for Multi-scale Spatio-Temporal Systems. In *Quantitative Evaluation of Systems*, Nathalie Bertrand and Luca Bortolussi (Eds.). Springer International Publishing, 243–258. https://doi.org/10.1007/978-3-319-66335-7_15

Michalis Michaelides, Dimitrios Milios, Jane Hillston, and Guido Sanguinetti. 2016. Property-Driven State-Space Coarsening for Continuous Time Markov Chains. In *Quantitative Evaluation of Systems*. Springer, Cham, 3–18. https://doi.org/10.1007/978-3-319-43425-4_1

J. R. Norris. 1998. *Markov Chains*. Cambridge University Press.

Sucheendra K Palaniappan, François Bertaux, Matthieu Pichené, Eric Fabre, Gregory Batt, and Blaise Genest. 2017. Abstracting the dynamics of biological pathways using information theory: a case study of apoptosis pathway. *Bioinformatics* 33, 13 (July 2017), 1980–1986. https://doi.org/10.1093/bioinformatics/btx095

Christopher V. Rao and Adam P. Arkin. 2003. Stochastic chemical kinetics and the quasi-steady-state assumption: Application to the Gillespie algorithm. *The Journal of Chemical Physics* 118, 11 (2003), 4999–5010. https://doi.org/10.1063/1.1545446

Carl Edward Rasmussen and Christopher K. I. Williams. 2006. *Gaussian processes for machine learning*. MIT Press, Cambridge, Mass.

Anthony D.J. Robertson and James F. Grutsch. 1981. Aggregation in Dictyostelium discoideum. *Cell* 24, 3 (June 1981), 603–611. https://doi.org/10.1016/0092-8674(81)90087-8

David Schnoerr, Guido Sanguinetti, and Ramon Grima. 2017. Approximation and inference methods for stochastic biochemical kinetics—a tutorial review. *Journal of Physics A: Mathematical and Theoretical* 50, 9 (2017), 093001. https://doi.org/10.1088/1751-8121/aa54d9

Michael W. Sneddon, William Pontius, and Thierry Emonet. 2012. Stochastic coordination of multiple actuators reduces latency and improves chemotactic response in bacteria. *PNAS* 109, 3 (2012), 805–810. https://doi.org/10.1073/pnas.1113706109

Edward Snelson and Zoubin Ghahramani. 2006. Sparse Gaussian Processes using Pseudo-inputs. In *Advances in Neural Information Processing Systems 18*, Y. Weiss, P. B. Schölkopf, and J. C. Platt (Eds.). MIT Press, 1257–1264.

Victor Sourjik and Howard C. Berg. 2004. Functional interactions between receptors in bacterial chemotaxis. *Nature* 428, 6981 (2004), 437–441. https://doi.org/10.1038/nature02406

Victor Sourjik and Ned S. Wingreen. 2012. Responding to Chemical Gradients: Bacterial Chemotaxis. *Curr Opin Cell Biol* 24, 2 (2012), 262–268. https://doi.org/10.1016/j.ceb.2011.11.008

Nikita Vladimirov, Dirk Lebiedz, and Victor Sourjik. 2010. Predicted Auxiliary Navigation Mechanism of Peritrichously Flagellated Chemotactic Bacteria. *PLOS Comp Bio* 6, 3 (2010), e1000717. https://doi.org/10.1371/journal.pcbi.1000717

Nikita Vladimirov, Linda Løvdok, Dirk Lebiedz, and Victor Sourjik. 2008. Dependence of Bacterial Chemotaxis on Gradient Shape and Adaptation Rate. *PLOS Comp Biol* 4, 12 (2008), e1000242. https://doi.org/10.1371/journal.pcbi.1000242

A GAUSSIAN PROCESS CLASSIFICATION DETAILS

In mathematical language, we observe the mapping we wish to approximate at N points of its domain, $\mathcal{D} = \{(X_i, y_i)\}_{i=1}^N$ where $X_i \in \mathcal{X}$ is the input value and y_i the output. In our case, the output is a binary value of the property evaluation $y_i \in \{\top = 1, \bot = 0\}$. We denote the collection of all $\{X_i\}_{i=1}^N = \mathbf{X}$ and $\{y_i\}_{i=1}^N = \mathbf{y}$. Given $\mathcal{D}$, we would like to infer the underlying probability function $\Psi : \mathcal{X} \to [0, 1]$ which we assume to give the probability parameter of the Bernoulli distribution generating the observations y:

$$y \mid X \sim \mathit{Bernoulli}(p = \Psi(X)). \tag{29}$$

To learn Ψ from $\mathcal{D}$ observations, the GP assumes the existence of a latent function $f : \mathcal{X} \to \mathbb{R}$ which we pass through an inverse-probit transformation to bring it within Bernoulli parameter domain range $[0, 1]$, such that $\Psi(X_i) = \Phi(f(X_i))$, where $\Phi : \mathbb{R} \to [0, 1]$ is the cumulative distribution function of the standard normal distribution $\mathcal{N}(0, 1)$. This forms the likelihood of the Bernoulli parameter being approximated ($\Psi(\cdot)$) given the latent function $f(\cdot)$. In fact, the GP assumes a whole distribution over possible latent functions; this is a multivariate normal defined by our choice of covariance structure (kernel $k(\cdot, \cdot)$) and mean function $m(\cdot)$, denoted as $f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot))$. Considering only the collection of function values $\mathbf{f} = [f(X_i)]_{i=1}^N$ where we have observations $\mathbf{X}$, our prior distribution becomes a finite-dimensional Gaussian $\mathbf{f} \mid \mathbf{X} \sim \mathcal{N}(\mathbf{m}, \mathbf{K})$, where bold letters denote the functions evaluated at each observation X_i or X_i, X_j pair. We proceed to condition on outputs $\mathbf{y}$ to obtain a posterior distribution over the latent functions through standard Gaussian distribution results.

$$p(y_i \mid X_i, f(\cdot)) = \mathit{Bernoulli}(\Psi(X_i)) = \Phi(f(X_i))^{y_i}(1 - \Phi(f(X_i)))^{1-y_i}, \tag{30}$$

$$\text{giving posterior } p(\mathbf{f} \mid \mathcal{D} = \mathbf{X}, \mathbf{y}) \propto p(\mathbf{y} \mid \mathbf{f})p(\mathbf{f} \mid \mathbf{X}). \tag{31}$$

To predict $\Psi(X_*)$ at an unobserved domain point X_* not in $\mathcal{D}$, we take a weighted average of all possible values of $\Phi(f(X_*))$ under the posterior distribution of latent function values $\Psi(X_*) = \langle \Phi(f(X_*)) \rangle_{f_* \mid \mathcal{D}, x_*}$, where for notational simplicity $f_* = f(X_*)$.

$$\Psi(X_*) = \int \Phi(f_*)p(f_* \mid \mathcal{D}, x_*)df_*, \tag{32}$$

$$\text{where } p(f_* \mid \mathcal{D}, x_*) = \int p(f_* \mid X, \mathbf{f}, x_*)p(\mathbf{f} \mid \mathcal{D})\, d\mathbf{f}. \tag{33}$$

GP regression with its many variations for different problem tasks is well described in [Rasmussen and Williams 2006]. The necessary adjustments which we adopt here are found in the Gaussian process classification (GPC) section of the book, and essentially amount to identifying that the class probability function is Ψ, where the class is the property satisfaction outcome. Standard results make the integrals in Eqs. 32-33 analytically tractable if we approximate the posterior $p(\mathbf{f} \mid \mathcal{D})$ with a Gaussian. To do this, we use Minka's Expectation-Propagation (EP) technique because it is more accurate than the alternative Laplace approximation. Further, we use fully independent training conditional (FITC) approximation [Snelson and Ghahramani 2006] to allow a large number of observations to be considered for learning the underlying function, while maintaining a low cost of predicting at any point of the domain. The approximation essentially replaces the original training data (input-output set) with a smaller set of *inducing points* (dummy input-output set) which is constructed from the former and used in prediction; these *inducing points* produce a conditional distribution over functions that is close to the one produced by conditioning on the real data. It relies on a rank-reduced approximation of the original covariance matrix.

B SIMULATION SCHEMES FOR *E. COLI* MODEL

B.1 Simulation scheme for original *E. coli* model

Algorithm 1 Simulation scheme for the *E. coli* model, based on full simulation of the pCTMC describing F/M conformation changes. Below, τ, mb_0, α are constants which parametrise the model (see [Sneddon et al. 2012]), and Δt is the fixed simulation time-step.

function RUN($\vec{r}$, $\vec{v}$, Δt)
  $\vec{r} \leftarrow \vec{r} + \vec{v} \cdot \Delta t$
  return $\vec{r}$
end function

function TUMBLE($\vec{v}$, Δt)
  $\theta \sim \Gamma(\text{shape} = 4, \text{scale} = 18.32)$ ▷ Sample tumbling angle from distribution given in [Sneddon et al. 2012].
  $\vec{v} \leftarrow R(\theta) \cdot \vec{v}$ ▷ $R(\theta)$ is a 2D rotation matrix through angle θ.
  return $\vec{v}$
end function

function OU-EULER-MARUYAMA(m, L, Δt)
  $\bar{m} \leftarrow \text{MEANMETH}(L, mb_0, \alpha)$ ▷ Mean methylation level $\bar{m}(L, mb_0, \alpha)$ as in [Frankel et al. 2014; Sneddon et al. 2012].
  $m \leftarrow m + \left[\Delta t/\tau(\bar{m} - m) + \sigma_m\sqrt{2/\tau}dW(\Delta t)\right]$
  return m
end function

procedure SIMULATEFINEECOLICELL(t_{end})
  $t \leftarrow 0$
  while $t < t_{\text{end}}$ **do**
    $L \leftarrow L(\vec{r}, t)$ ▷ The ligand field L value, at the cell's location $\vec{r}$.
    $\mathbf{s} \leftarrow \text{pCTMC}(\mathbf{s}, m, L, \Delta t)$ ▷ Drawing F/M pCTMC trajectory of length Δt, with parameters $k_{\pm}(m, L)$ and initial state the last pCTMC state of the cell.
    $\psi \leftarrow \phi_{\text{RUN}}(\mathbf{s})$ ▷ Evaluating the ϕ_{RUN} on (the final state of) the pCTMC trajectory.
    if ψ **then**
      $\vec{r} \leftarrow \text{RUN}(\vec{r}, \vec{v}, \Delta t)$
    else
      $\vec{v} \leftarrow \text{TUMBLE}(\vec{v}, \Delta t)$
    end if
    $m \leftarrow$ OU-EULER-MARUYAMA(m, L, Δt) ▷ Evolving methylation.
    $t \leftarrow t + \Delta t$
  end while
end procedure

B.2 Simulation scheme for abstracted *E. coli* model

Algorithm 2 Simulation scheme for the abstracted *E. coli* model, based on GP approximation for the RUN/TUMBLE probability. Steps 5, 6 here replace the expensive Steps 22, 23 in Algorithm 1.

1: **procedure** SIMULATEABSTRACTEDECOLICELL(t_{end})
2: $t \leftarrow 0$
3: **while** $t < t_{\text{end}}$ **do**
4: $L \leftarrow L(\vec{r}, t)$
5: $p \leftarrow \text{GP}_{\psi}(m, L)$
6: $\psi \sim Bernoulli(p)$
7: **if** ψ **then**
8: $\vec{r} \leftarrow \text{RUN}(\vec{r}, \vec{v}, \Delta t)$
9: **else**
10: $\vec{v} \leftarrow \text{TUMBLE}(\vec{v}, \Delta t)$
11: **end if**
12: $m \leftarrow \text{OU-EULER-MARUYAMA}(m, L, \Delta t)$
13: $t \leftarrow t + \Delta t$
14: **end while**
15: **end procedure**

C CONSTANTS OF *D. DISCOIDEUM* MODEL

Table 2. Table of fixed constants for the *D. Discoideum* model. These are used in Equations 13-17 and taken from [Calovi et al. 2010], where their physical interpretation is also examined.

Parameter	Value	Parameter	Value
λ_1	10	k_i	1.7 min^{-1}
λ_2	0.18	k_t	0.9 min^{-1}
λ_3	463.5	k_e	5.4 min^{-1}
k_1	0.036 min^{-1}	σ	0.01 mm
k_{-1}	0.36 min^{-1}	D	0.024 mm^2/min
k_2	0.666 min^{-1}	h	0.025
k_{-2}	0.00333 min^{-1}		