

Development of a Parameter Estimation Pipeline for Multi-Cellular Biological Processes

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von
Emad Mohammed Alamoodi
aus
Tabuk, Saudi Arabia

Bonn 2026

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen
Friedrich-Wilhelms-Universität Bonn

Gutachter/Betreuer:	Prof. Dr. Jan Hasenauer
Gutachter:	Prof. Dr. Kevin Thurley

Tag der Promotion: 08. Juni 2026
Erscheinungsjahr: 2026

Acknowledgements

The PhD journey is rarely easy, especially one undertaken during the COVID-19 pandemic, yet it is very rewarding once it reaches its end. For me, reaching the end was not possible without the help of many people who encouraged, supported, and accompanied me throughout the path.

First and foremost, I thank God for his countless blessings and guidance throughout my life.

I'm grateful for my parents, my wife, my siblings, and my children for their unconditional patience, love, and support. A heartfelt thanks also goes to Claudia Reske and her family for becoming our extended family abroad and for making our time away from home feel far less distant.

I owe special thanks to Prof. Jan Hasenauer for his supervision, mentorship, and continuous support throughout my time as a PhD student. He gave me a chance to be a member of his group and had the knowledge, the patience, and the kindness to allow me to grow and mature scientifically.

I am also grateful to Prof. Kevin Thurley for being the second reviewer for my dissertation and to Prof. Stephan Jonas and Prof. Thomas Schultz for joining my doctoral committee.

I would further like to thank my cooperation partners, whose contributions and collaboration made many of my projects possible.

Finally, I would like to thank all Hasenauer group members for making my time at the group memorable. In particular, I would like to thank Yannik Schälte, Jakob Vanhoefer, Elba Raimúndez, Simon Merkt, Erika Dudkin, Jonas Arruda, Dilan Pathirana, Clemens Peiter, Polina Lakrisenko, Marc Vaisband, Daniel Weindl, Lea Seep, Domagoj Dorešić, Vincent Wieland, Stephan Grein, Roy Gusinow, Manuel Huth, Paul Jost, Maren Philipps, and Nina Schmid.

Emad Alamoudi, Jeddah 2025

Abstract

Tissue dynamics are complex and involve interactions between different cell types and extracellular components on different spatial and temporal scales. Specific tissue properties are relevant for a broad range of processes, including tissue homeostasis, viral infection, and tumor development and treatment. The multi-scale and multi-cellular model has been proven to be a valuable tool to study these dynamics. However, these models remain challenging to use in practice. The lack of computational tools and widely adopted data standards that facilitate its simulation and calibration hinders its wider adaptation.

The goal of this cumulative thesis is to advance the computational foundation needed for multi-scale and multi-cellular models to become more accessible, scalable, and reproducible, with a special focus on the parameter estimation aspect.

First, we develop the FitMultiCell pipeline for simulating and parameterizing multi-scale and multi-cellular biological systems. The pipeline has been tested in different real biological applications to validate its robustness and scalability. It provides a user interface for easier adaptation with an extensive set of documentation.

Second, we improve the wall-time of the approximate Bayesian computation—the sequential Monte Carlo (ABC-SMC) inference method in the pipeline by introducing a new scheduling strategy where we are able to achieve better wall-time by introducing a higher resource utilization. We evaluate the new strategy across multiple realistic application scenarios and show consistent performance improvements. By shortening runtimes and improving scalability, the approach enables the calibration of more complex models within practical time limits.

Third, we introduce a standardized format, PETab-MS, to encapsulate the parameter estimation problem of a multi-scale and multi-cellular nature. PETab-MS provides a structured, reusable, and machine-readable way to define the model, experimental data, conditions, parameters, and objective function in a reusable and machine-readable form. It enables easier sharing and comparison of results across different studies. This will improve collaboration and advance the research of these models.

Fourth, we apply these advances to investigate the cell migration process in a pillar-forest microenvironment by using a cellular Potts model (CPM) calibrated on experimental data. The results not only show the impact of spatial constraints on impacting the migration dynamics but also highlight the potential of data-driven modeling pipelines to derive mechanistic insights from complex systems.

In conclusion, the advances presented in this thesis enable wider adaptation through the introduction of a general-purpose pipeline, a standardized calibration format, and a faster inference strategy. The practical value of these contributions is demonstrated across multiple biological case studies, both individually and as an integrated end-to-end workflow.

Contents

1	Introduction	1
1.1	Research Question	4
1.2	Contributions of this thesis	5
1.3	Outline of this thesis	6
2	Background	7
2.1	Modeling of biological processes	8
2.1.1	Modeling formalisms	8
2.1.2	Modeling cell migration	13
2.1.3	Morpheus	14
2.2	Bayesian parameter estimation	15
2.2.1	Markov chain Monte Carlo	15
2.2.2	Sequential Monte Carlo	16
2.2.3	Challenges in multi-cellular modeling	16
2.3	Likelihood-free inference	17
2.3.1	Approximate Bayesian computation	17
2.3.2	ABC rejection sampler	18
2.3.3	Markov chain Monte Carlo ABC (ABC-MCMC)	19
2.3.4	Approximate Bayesian computation - sequential Monte Carlo	19
2.3.5	Parallelization strategies for ABC-SMC	21
2.3.6	Accelerating ABC-SMC through HPC resources	21
2.3.7	Surrogates and emulators to accelerate simulation-based inference	22
2.4	Amortized inference	23
2.5	Standardization and reproducibility	23
2.6	Cell migration	24
2.6.1	Overview of the cytoskeleton in cell migration	25
2.6.2	Experimental methods for studying cell migration	25
3	Integrated Pipeline for Simulating and Inferring Unknown Parameters of Multi-scale Processes	27
3.1	Motivation	27
3.2	Methodology	28
3.3	Significance within the thesis	28
3.4	Application to Biological Case Studies	29
3.5	Author's contribution	30

4	Parallelization Approach to Minimize Wall-Time for Approximate Bayesian Computation	31
4.1	Motivation	31
4.2	Methodology	32
4.3	Application to biological case study	33
4.4	Significance within the Thesis	34
4.5	Author's Contribution	34
5	Inference of Cell Migration Dynamics in Confined Microenvironments	35
5.1	Motivation	35
5.2	Methodology	36
5.2.1	Estimation framework	37
5.3	Application to biological case study	37
5.4	Significance within the thesis	37
5.5	Author's Contribution	39
6	Discussion	41
6.1	Discussion of Results	41
6.2	Outlook and future directions	42
7	Appendices	45
7.1	FitMultiCell: simulating and parameterizing computational models of multi-scale and multi-cellular processes	45
7.2	A wall-time minimizing parallelization strategy for approximate Bayesian computation	66
7.3	Simulation-Based Inference of Cell Migration Dynamics in Complex Spatial Environments	109
	Bibliography	147
	List of Figures	159
	List of Algorithms	161

Introduction

But biology and computer science – life and computation – are related. I am confident that at their interface, great discoveries await those who seek them.

— Leonard Adleman

Biological tissues are dynamic and complex, composed of cells and extracellular components. Modeling this complexity requires a detailed understanding of the interactions between different cells and also within the cell itself. Computational modeling for multi-cellular systems has become a valuable tool to understand the processes determining tissue dynamics. A multi-cellular system is a system that consists of more than one cell, as opposed to a unicellular system. These systems start from a single cell, which grows, divides, differentiates, and forms tissues, organs, systems, and organisms (see Figure 1.1). A key issue of these systems is that of scale: their components interact at multiple spatial scales, from the molecular level to the tissue level to the whole organism. Moreover, these interactions occur over different temporal scales, from seconds in chemical reactions (i.e., protein binding to DNA) to hours in circadian rhythms and even months with seasonal changes (i.e., light-dark cycle changes) [1]. A uni-scale model cannot capture all these different spatial and temporal scales.

Thus, to handle the interactions that span over multiple scales, from individual molecules to whole organisms, multi-scale models are introduced (see Figure 1.2). Multi-scale models can link different spatial or temporal scales using two main strategies. Top-down approaches begin with observed features on a high level of a system and then attempt to infer what kinds of mechanisms exist on lower levels. On the other hand, bottom-up models aim to derive a system's behavior on higher spatial or temporal scales from the dynamics and interactions of model components that exist on lower ones. In addition to that, the modeler must identify the most useful definition of the different scales, their functional components, and how information would be exchanged between different scales of a model.

Nowadays, there are a number of software tools available that allow users to create and simulate quantitative models of cellular signaling processes without having to worry about the more technical aspects. These tools include Biocellion [2], CellSys [3], Chaste [4], CompuCell3D [5], Morpheus [6], PhysiCell [7], and VirtualLeaf [8]. Different scales can have different characteristics, each with its own temporal and spatial resolution, and can be modeled using the simulation approaches associated with them.

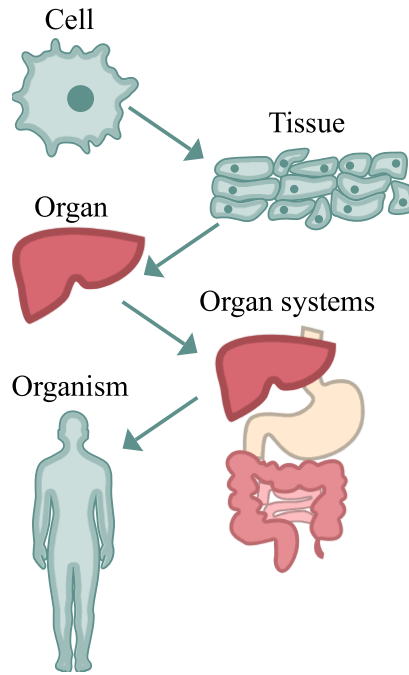


Figure 1.1: Cells work together to make up tissues, which work together to make up organs, which make up organ systems, which work together to make up an organism.

Moving from a schematic view of scales to concrete biological behavior, we highlight the case of chemokine-guided cell migration in structured tissues. To reach lymphoid organs, dendritic cells must migrate through complex tissue environments directed by chemokine gradients. To understand the mechanisms of the immune system, one needs to understand that the cell can integrate chemical signals with physical constraints [11]. These signals can affect the polarity, persistence, and speed of the cell.

Spatial models such as the cellular Potts model capture morphology, adhesion, and chemotaxis and are implemented in many of the established tools [5–7, 12, 13]. These models have been used to study the effect of chemokines on cell migration and the trade-off between speed and persistence. However, the parameters of complex models are rarely observable and therefore need to be calibrated from data [14]. Bayesian methods, notably ABC-SMC, enable such inference for stochastic simulators and have been demonstrated at scale for multi-cellular systems [15, 16].

Despite their usefulness, multi-scale models of multi-cellular systems remain technically challenging. Fletcher and Osborne [17] have identified seven issues with these models that need to be addressed: 1) constructing the model, 2) calibrating it, 3) finding a numerical solution, 4) software and hardware implementation, 5) validating the model, 6) dealing with data/code standards, and 7) comparing modeling assumptions and approaches.

Hazelbag et al. [18], who reviewed the individual-based models (IBMs) used in epidemiology, found that less than half of the studied cases use algorithmic parameter search with a quantitative goodness-of-fit measure, while the remainder use unclear or non-reproducible methods. A small fraction of the studies' cases justified their calibration choice or reported model validation, and only about one-third reported parameter uncertainty [18]. The authors conclude that better-documented,

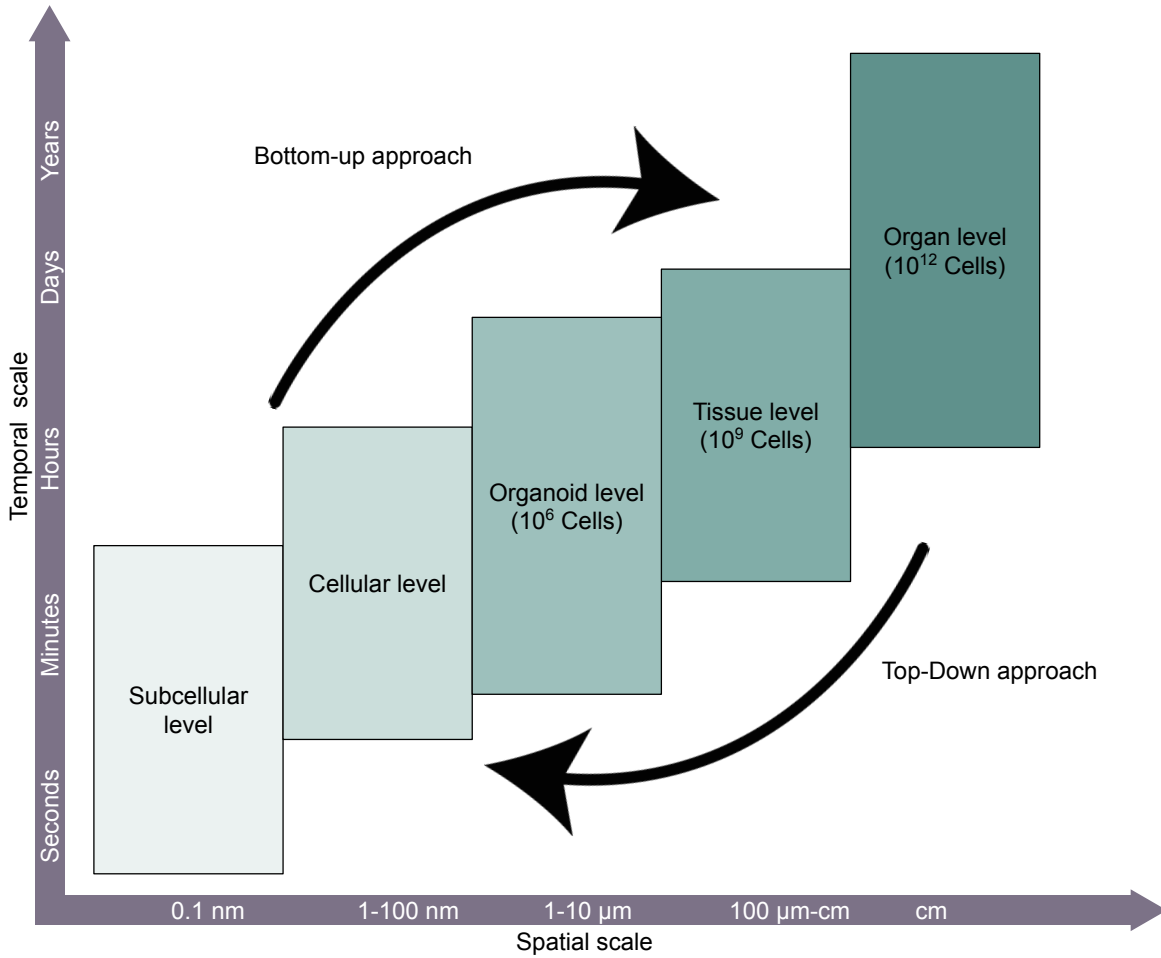


Figure 1.2: The figure illustrates the multi-scale concept of various biological processes that can take place across different spatial and/or temporal scales, such as the molecular level, cell level, organoid level (10^6 cell), tissue level (10^9 cell), and organ level (10^{12} cell) [9] with a spatial resolution that ranges from nanometers to centimeters and a temporal resolution that varies from seconds to years [10].

algorithmic calibration methods are required to improve reproducibility and the quality of inference. Although this review focused on epidemiological IBMs, the same issues arise in stochastic, spatially explicit multi-cell models [19, 20].

To achieve reproducibility in the context of multi-scale models, Pesonen et al. [21] argue that there is a need for inference frameworks that reduce the manual overhead of the ad hoc simulation setup, motivating the development of reusable, standardized pipelines. Another way to improve reproducibility is through standardized formats. Höpfl et al. [22] show that models with fully reproducible standardized calibration workflows (using formats such as SBML for models and PTab for data) have a significantly higher reuse and impact in the community.

Moreover, Pesonen et al. [21] also discusses that well-established methods for likelihood-free inference, such as ABC-SMC, are time-consuming. Thus, further optimization is needed to enable tackling complex models efficiently and effectively.

Given all these challenges, the goal of this cumulative thesis is to address these challenges mentioned by Fletcher and Osborne. Each of the thesis papers contributes to one or more of these aspects. Accomplishing this goal requires the contribution of the software pipeline, algorithm innovation, and application case studies, which are reflected in the thesis's structure.

1.1 Research Question

The primary objective of this thesis is to investigate how to make simulation-based inference reproducible and scalable for multi-scale and multi-cellular systems to better understand biological processes like the mechanistic aspects of chemokine-guided cell migration in structured environments. In this thesis, I focus on the following open problems and challenges:

- I. **How can we develop general-purpose, user-friendly workflows that simulate and parameterize multi-scale and multi-cellular models while also supporting other features like model selection, uncertainty quantification, and result validation?** There are some dedicated pipelines that were established in the past, but they were built for a specific problem, thus limiting their reusability. Therefore, there is a need for more general-purpose and user-friendly tools that can be applied to a wide range of multi-scale modeling problems. Such a tool should not only facilitate the simulation and parameterization but also allow for evaluating parameters, validating results, and assessing model predictions.
- II. **How can we standardize the full specification of multi-scale and multi-cell inference problems (model, data, observables, priors, distance, etc.) so that studies become reproducible and shareable between tools and groups?** The lack of standards poses challenges for the reproducibility and comparability of results across different studies. It hinders the ability to validate and refine multi-scale and multi-cellular models, limiting their potential impact in understanding complex biological systems. Efforts must be made to establish standardized guidelines and protocols for parameter estimation in multi-cellular models, promoting transparency and facilitating collaboration among researchers in this field.
- III. **How can we reduce the overall wall-clock time and improve the resource utilization of likelihood-free calibration, more specifically, the ABC-SMC, without compromising the inference accuracy?** ABC-SMC is popular for its simplicity and wide applicability. However, a recognized limitation of this method is its dependence on extensive simulations, often running into hundreds of thousands to millions of iterations. Thus, techniques to improve scalability and efficiency are needed.
- IV. **How can we improve our quantitative understanding of cell migration using a model of chemokine-directed cell migration in a maze-like microenvironment against real single cell trajectories to quantify the roles of persistence, chemokine sensitivity, and geometric confinement?** Chemokines are essential in many biological processes and are involved in various pathological conditions. Therefore, they are central to research in developmental biology and clinical studies targeting cancer, inflammation, and immune-related diseases. Thus, a quantitative analysis of these processes will extend our knowledge of the underlying mechanisms.

1.2 Contributions of this thesis

The abovementioned questions are addressed in three peer-reviewed publications and summarized in chapters 3, 4, and 5:

- Question (I) was addressed in the first paper [23] by developing a platform for simulating and parameter estimation of multi-scale and multi-cellular models. We developed a pipeline called FitMultiCell, which can handle model construction, simulation, and calibration on experimental data. This platform can assess the quality at each step by generating diagnostic plots and figures.
- The second question (II) was also addressed in the same paper [23]. We introduced a standard format for the inference problem that can handle the complexity of the multi-scale and multi-cellular system called PÉtab-MS. PÉtab-MS is an extension of the PÉtab format and is available as a Python package on PyPi. This format allows users with no technical background to define the inference problems as a set of tables that specify things like measurement, observable, parameter information, etc.
- Question (III) is tackled in paper [24], where we present a look-ahead scheduling strategy that minimizes wall-time for ABC-SMC on multi-core and distributed systems. This new strategy tries to improve the speed-up by optimizing the utilization of computational resources. Using the new strategy, the computational resources are effectively allocated and utilized throughout the process, resulting in a significant reduction in wall-time for ABC.
- Question (IV) is investigated in paper [25] where we calibrate a cellular Potts model of dendritic-cell migration in a pillar-forest (maze-like) microenvironment using time-lapse microscopy trajectories and simulation-based inference.

Some of the work in this thesis has already been published in peer-reviewed journals or is currently being reviewed. This thesis includes contributions that are similar to or identical to the following publications:

- **Alamoudi, E.**, Schälte, Y., Müller, R., Starruss, J., Bundgaard, N., Graw, F., Brusch, L., & Hasenauer, J. FitMultiCell: Simulating and parameterizing computational models of multi-scale and multi-cellular processes. *Bioinformatics* 39.11 (2023). doi.org/10.1093/bioinformatics/btad674
- **Alamoudi, E.**, Reck, F., Bundgaard, N., Graw, F., Brusch, L., Hasenauer, J., & Schälte, Y. A Wall-time Minimizing Parallelization Strategy for Approximate Bayesian Computation. *PLOS ONE* 19.2 (2024) 1. doi.org/10.1371/journal.pone.0294015.
- Arruda, J., **Alamoudi, E.**, Mueller, R., Vaisband, M., Molkenbur, R., Kiermaier, E., Hasenauer, J., Simulation-Based Inference of Cell Migration Dynamics in Complex Spatial Environments, *bioRxiv*, 2025-9.

Additionally, results of the following publications in which I was involved are used, albeit not discussed in detail:

- **Alamoudi, E.**, Starruß, J., Bundgaard, N., Muller, R., Reck, F., Graw, F., ... & Schalte, Y. (2022). Massively Parallel Likelihood-Free Parameter Inference for Biological Multi-Scale Systems. *NIC Symposium* 2022.

-
- Schälte, Y., Klinger, E., **Alamoudi, E.**, & Hasenauer, J. (2022). pyABC: Efficient and robust easy-to-use approximate Bayesian computation. *Journal of Open Source Software*, 7(74), 4304.
 - Schälte, Y., **Alamoudi, E.**, & Hasenauer, J. (2021). Robust adaptive distance functions for approximate Bayesian inference on outlier-corrupted data. *bioRxiv*, 2021-07.

My doctoral research has yielded other publications that are not included in this thesis:

- Raimúndez, E., Dudkin, E., Vanhoefer, J., **Alamoudi, E.**, Merkt, S., Fuhrmann, L., Bai, F., & Hasenauer, J. (2021). COVID-19 outbreak in Wuhan demonstrates the limitations of publicly available case numbers for epidemiological modeling. *Epidemics*, 34, 100439.

As a part of my involvement in these research papers, I developed several toolboxes that make it possible to reuse the methods presented here:

- FitMultiCell (<https://gitlab.com/fitmulticell/fit>), see [26]: A Python-based pipeline that combines the parameter estimation tool pyABC with Morpheus (<https://gitlab.com/morpheus.lab/morpheus>), a tool to build and simulate multi-scale and multi-cellular systems
- pyABC (<https://github.com/ICB-DCM/pyABC>), see [27, 28]: A Python-based package for sequential approximate Bayesian computation (ABC) on local and high-performance infrastructure.
- PEstab-MS (<https://gitlab.com/fitmulticell/libpetab-python-MS>), see [26]: A Python-based package for a systems biology parameter estimation problem for a model of multi-scale nature. It contains all the information needed for the estimation process.

1.3 Outline of this thesis

This thesis is organized as follows: In Chapter 2, I introduce the topic of modeling biological processes. There, I explain different model formalizations used in the field, and then I focus on multi-scale models. Next, I introduce the ABC method. There, I explain the different likelihood-free approaches, more specifically, ABC-SMC and amortized inference. Later, I discuss the topic of standards and reproducibility and show their value in the field of computational biology. Next, I provide an overview of the process of cell migration with a brief historical background. That is followed by the topics of the cytoskeleton and its role in cell movement, the experimental method, and concluding by talking about the used model formalism. Chapters 3, 4, and 5 contain my main contribution to this thesis. In Chapter 3, I introduce and describe the FitMultiCell as a pipeline that facilitates the simulation and estimation of multi-scale and multi-cellular models. Then, I use it to fit different real application problems, each representing a different biological context. In Chapter 4, I propose a new parallelization strategy that minimizes wall time by efficiently utilizing computational resources. I show the benefit of this new method on different toy and real application problems. In Chapter 5, I constructed a multi-cellular model that describes the chemokine-guided cell migration process. Then, I fit the model to real time-lapse cell trajectories. I wrap up this thesis in Chapter 6, in which a summary of the contribution of the thesis is given and an outlook for the future direction.

Background

In this chapter, I introduce the needed background information on the terminology and methodology that underlie and are relevant throughout this thesis. First, I describe the modeling of biological processes and the different model formalisms that are commonly used in the field. Then, I focus on the multiscale and multi-cellular process and its importance in understanding complex biological systems. Additionally, I discuss the challenges associated with integrating different scales and cell types into a unified model. Second, I introduce the approximate Bayesian computation (ABC) method and its application in parameter estimation for biological models. I explain how this method enables the inference of the model's parameters based on observed data, even when the likelihood function is intractable. Further, I discuss different approximate Bayesian computation methods and focus on approximate Bayesian computation sequential Monte Carlo (ABC-SMC) in addition to other likelihood-free methods such as amortized inference. Then, I discuss the standardization and reproducibility effort in the systems biology domain. Finally, I move on to a biological example to demonstrate the capability of the introduced algorithm. I give an overview of the cell migration process with a quick look at the dynamics behind the movements. This chapter offers a broad and relatively high-level overview, but the main contribution chapters that follow may include more detailed information associated with those topics.

Parts of this chapter are based on and partly identical to the following publications of the thesis author:

- **Alamoudi, E.**, Starruß, J., Bundgaard, N., Muller, R., Reck, F., Graw, F., ... & Schälte, Y. (2022). Massively Parallel Likelihood-Free Parameter Inference for Biological Multi-Scale Systems. NIC Symposium 2022.
- Schälte, Y., Klinger, E., **Alamoudi, E.**, & Hasenauer, J. (2022). pyABC: Efficient and robust easy-to-use approximate Bayesian computation. arXiv preprint arXiv:2203.13043.
- Schälte, Y., **Alamoudi, E.**, & Hasenauer, J. (2021). Robust adaptive distance functions for approximate Bayesian inference on outlier-corrupted data. bioRxiv, 2021-07.

2.1 Modeling of biological processes

Biological systems such as cells, tissues, or the human body are dynamic and complex, involving complex interactions between various components. Mathematical models are useful tools that allow researchers to gain insight into the fundamental mechanics of these processes by using mathematical formulations and numerical simulations. They allow scientists to make predictions and test hypotheses about biological systems. They typically contain a large number of interacting components, and the overall system behavior emerges from their combined dynamics. In the following, I give an overview of the available model formalisms and their applications in various fields of systems biology. For a general review of the field and its formalisms, refer to [1, 29].

2.1.1 Modeling formalisms

In systems biology, the complexity of living systems and the wide range of biological scales and mechanisms have led to the development of several modeling formalisms. These models are not a perfect representation of the biological process under investigation, but an approximation of the underlying dynamics. Each method must compromise over the trade-off between the accuracy of the system representation and computational efficiency (Figure 2.1). As the size of the investigated system increases, the complexity of the models also increases, making it challenging to accurately capture all interactions and behaviors within the system [30]. Thus, each of these methods has its advantages and limitations, depending on the available data and the research question. To select the right modeling approach, one needs to think about the specific goals of the study and the trade-offs that need to be made to ensure that the results are both informative and feasible for analysis.

Modeling formalisms in systems biology are based on differential equations, stochastic systems, agent-based modeling, etc. (for a review, [31]). In the following sections, I discuss some of the common modeling techniques in more detail, ranging from qualitative approaches, which describe general system behavior without precise numerical values, to quantitative approaches, which use numerical data and mathematical equations to provide detailed, precise information about the system's dynamics.

Boolean networks and logical models

The Boolean network model (also called the logical model) does not require knowledge of kinetic parameters, such as the synthesis and degradation rates, and yet it is able to provide a qualitative dynamic description of the biological system [32]. The system's overall value at a given time is determined by the value of all nodes on the system. These models define the interaction in terms of a logical function instead of a reaction rate equation. It consists of a set of nodes with Boolean values. Each node has a binary state, ON (1) or OFF (0), which represents a gene in an active or inhibited state [33]. The Boolean function denoted by f is a logical rule that results in a binary value as an output based on a binary input, as defined in Equation 2.1.

$$f : \{0, 1\} \rightarrow \{0, 1\} \quad (2.1)$$

They have been particularly useful for systems with hundreds of components. The Boolean network model has been introduced as an abstract version of genetic regulatory networks [34, 35]. The power

of these types of models is that they can lead to a predictive and testable hypothesis, which can be of high value for complicated large-scale systems [32].

As the Boolean network is good at providing a simpler and more intuitive version to study a poorly understood system than other formalisms (e.g., ODE- and PDE-based models), it often cannot capture the fine-grained dynamics. Yet, they can still reproduce the important dynamical property of some complex biological networks [36, 37].

The Boolean network model has been successfully applied in gene signaling and regulatory networks in different species [38–40]. In addition to that, it has also been used in additional biological processes, such as cell differentiation during development [41], hematopoiesis [42], and cell cycle control [43]. For a general review on the logical modeling of the Boolean network, check the existing review [44].

Cellular automata

Cellular automata (CA) are a grid-based model that consists of an n-dimensional lattice of regular sites (cells). Each site has a finite state variable and a set of adjacent neighboring cells that encapsulate the locality of cell interactions.

Unlike 'top-down' approaches such as ODEs and PDEs, CA is a bottom-up approach in which a complex system is defined by its individual components. A cell can be either empty or occupied by an object from the system. The object can interact with other objects at discrete time steps and according to a set of rules. Once all cells in the grid have computed and updated their state, it will be considered as one iteration. Once the interaction happens, according to its rule, each cell can evolve into a finite set of available states based on a function of the current state of the cell and its neighboring cell in the previous time step. A CA has four basic elements 1) the grid of cells; 2) the neighboring cells; 3) the initial state of all cells on the grid; and 4) the set of rules that change the state of the cell [45].

The CA rules can be deterministic or probabilistic, and they operate in discrete time and space. The deterministic CA follows a set of rules that apply in the same way to all cells. On the other hand, the probabilistic CA gives each cell a stochastic choice to update its state to account for uncertainty. The overall system dynamics can be simulated using different schemes. The system update can be synchronous, meaning that the update occurs at once for all cells. The update can also be asynchronous, where each cell update happens sequentially.

A cell within the CA's lattice can be spatially configured into one-, two-, or three-dimensional models, where the two-dimensional one is the most common [46]. Within the two-dimensional grid, the von Neumann neighbors is the most common, where a cell has four orthogonal neighbors, adding the four corner cells will give the extended neighborhood called the Moore neighborhood [46].

While ODE and PDE treat time and space as a continuum, CA treats them in a discrete manner. This brings about a trade-off between mathematical tractability and accuracy. CA is intuitive and flexible, but computationally complex.

CA has been applied to several biological systems. It has been used to model tumor growth in which grid sites are occupied by cells that can divide, proliferate, die, or move to another grid site [47]. CA has also been used to model infection spreading among a population of cells, e.g., influenza spreading [48]. Pattern formation was another application where CA was applied [49].

Cellular Potts model

The cellular Potts model, also known as the Glazier–Graner–Hogeweg model, is a cell-based modeling approach that is used to describe complex cell behavior [12, 50]. It is a grid-based approach in which each grid point represents a cell, subcellular, or extracellular matrix component, and the interaction and behavior are described in terms of energy and constraints, which enable cells to form arbitrary shapes and formations. Due to its straightforward expression of cell shape and movement, CPM is regarded as a convenient option for describing cell behavior and tissue structure across different scales [51].

The cell behavior in CPM is governed by the balance of forces encoded in a generalized energy function H . In its basic form, the energy function H include a term for volume constraint responsible for maintaining an approximately constant cell volume and a term for surface energy responsible for cell-cell adhesion. By adjusting these terms, CPM can capture basic cell behaviors. The dynamics of the system are determined by minimizing H using a dynamic Monte Carlo simulation algorithm. The outcome of the process enables the formation of complex multicellular structures that mimic biological tissues.

The CPM model formalism was first used by [12, 50] on cell sorting, yet it has been used on many other application including blood vessel network formation [52, 53], cell migration in 2D and 3D [25, 54], and tissue growth and deformation [55, 56]. Please refer to other reviews for additional applications and additional mathematical details [51, 57, 58].

Agent-based models

In recent years, agent-based models (ABMs) have become very powerful and popular tools for simulating complex systems and studying emergent phenomena. They represent discrete and often stochastic systems as spatially defined models or as compartmental models with many interacting agents. ABMs offer a flexible yet strong framework for the development of multi-scale models. They can be described in one-, two-, or three-dimensional space, but as the dimensions increase, the computational cost will also increase.

Each agent interacts with other agents in the system and with the environment according to specific behavioral rules and evolves over a defined time frame, leading to emergent properties of the system as a whole. Each agent represents an individual entity, e.g., a cell, with its own attribute that is updated and changes according to the rules and information available to the agent [59, 60].

Another bottom-up approach, besides CA, is ABM. Both CA and ABM use individual-level representations, yet CA emphasizes simple local interactions on a grid, while ABM allows agents to make more complex and knowledge-based decisions [59].

Building an agent-based model from scratch might be an intensive and time-consuming process. Thus, some tools and frameworks have been developed to facilitate model building, simulation, and diagnoses. Some of these tools are BioDynaMo [61], Agent.jl [62], GAMA [63], Mason [64], and Mesa [65]. A more detailed overview of the available tools and the differences between them is available in [66].

A typical ABM would consist of three primary components. 1) Agents that utilize both space and stochasticity, 2) a set of rules that define how agents can interact with each other, and 3) an environment that gives agents spatial information about different agents [67]. Any additional complexity in model design requires extra rules. ABMs have been used to simulate multi-cellular processes such as

cell-to-cell interaction [68], tumor growth [69], and the spread of infectious diseases [70], and many others.

A common issue with ABM is that they are usually computationally intensive since each agent is simulated explicitly [59]. Their computational cost increases with the number of simulated agents. Also, with too many details, the ABM might be treated as a black box due to the growing number of rules, parameters, and interactions, which affect the model's interpretability.

Ordinary differential equations

Ordinary differential equations (ODEs) are a very common modeling approach in many fields, including but not limited to systems biology, physics, chemistry, and economics. They model the time evolution of the underlying species in living cells. They are popular due to their ability to produce a useful approximation of the underlying process while still being computationally efficient.

ODE models do this by assuming that the modeled biochemical species are sufficiently abundant so that deterministic and continuous dynamics govern the process. These deterministic dynamics assume that random fluctuations are negligible, which simplifies the analysis and simulation of the system.

Also, they assume that all species are homogeneously distributed. The ODE model usually describes the temporal evolution of a state vector $z = z(t, \theta) \in \mathbb{R}^{n_z}$ of continuous concentrations of the modeled species by the parameterized initial value problem

$$\dot{z}(t, \theta) = f(z(t, \theta), \theta), \quad z(t_0, \theta) = z_0(\theta) \quad (2.2)$$

in which the vector field $f : \mathbb{R}^{n_z} \times \mathbb{R}^{n_\theta} \rightarrow \mathbb{R}^{n_z}$ denotes the temporal evolution of the state variables $z = z(t, \theta) \in \mathbb{R}^{n_z}$ at time $t \in [t_0, T] \subset \mathbb{R}$. The state vector $z(t, \theta) \in \mathbb{R}^{n_z}$ describes the quantity of interest for the system being modeled. The vector of unknown parameters is denoted by $\theta \in \Theta \subset \mathbb{R}^{n_\theta}$, where Θ is the set of possible values of the parameters. This vector often includes the initial count of species and the rate of their reactions and is estimated using measurement data.

These ODEs are usually based on chemical reaction rate equations, e.g., mass action kinetics, which assume that the rate of reaction is proportional to the product of the concentrations of reactants, or Michaelis-Menten [71], which describes enzyme-substrate interactions.

Another feature of ODEs is their flexibility, which means that it allows their integration with other mathematical models in order to develop increasingly complex systems that better capture the details of the biological phenomena. For example, they can be coupled with stochastic processes or partial differential equations to account for noise, spatial effects, or other phenomena that are not captured by traditional ODEs [72, 73]. This makes ODEs very useful in the study of dynamic systems, capable of providing insights into both the qualitative and quantitative behavior.

Stochastic models for cellular processes

The basic assumption of deterministic methods, such as ODEs and partial differential equations (PDEs), is that 1) the dynamics of the systems are deterministic and that 2) the quantity of molecular species is high. However, these assumptions might not always hold. Stochastic models explicitly represent the inherent randomness of biochemical reactions in cells. The chemical reaction occurrence times and the diffusion trajectory are both inherently random. In addition, not all biological systems are available in a large number of molecules. Thus, stochastic methods that do not only rely on the population average are needed.

Stochastic methods help to understand various biological phenomena, which include gene expression, cellular signaling pathways, and epidemiology. They account for intrinsic variability in the biological system, making them suitable for modeling cases with low particle numbers or high fluctuations compared to deterministic methods. It's especially true at the cellular level, where the interaction at the molecular scale may give different results due to their probabilistic nature. An example of stochastic methods is stochastic differential equations (SDEs), which are similar to ODEs but with additional terms that represent the random fluctuations [74]. A basic form of SDEs would be

$$dX_t = f(X_t)dt + g(X_t)dW_t \quad (2.3)$$

where X_t is the state vector, $f(X)$ represents the reaction rate, $g(X)$ is the diffusion term, and W_t is the noise term.

Another stochastic method is the chemical master equation (CME) [75]. At low copy number, the state of a well-mixed biochemical system can be described as a continuous-time Markov chain (CTMC), whose discrete states correspond to the molecule counts of each species and whose transitions correspond to the individual reaction events that occur at random points in time [1] [76]. It is considered to be one of the main modeling methods for describing chemical reaction networks. The CME can be seen as a set of coupled ODEs that cover all possibilities of the system state. Typically, it tracks different molecules of different species. However, as the population of species increases, the number of reachable states grows and the CME becomes harder to solve [77].

Instead of solving the CME directly, one can simulate the system dynamics by using Monte Carlo methods. Rather than evolving the full probability distribution over the CTMC states, these methods generate individual realizations of the same CTMC. The stochastic simulation algorithm (SSA), also known as the Gillespie algorithm, is a widely used approach to simulate the time evolution of chemical reactions set within a well-mixed system [78]. The basic Gillespie approach will be as follows: 1) in each time step, the propensity of all possible reactions a_j is calculated; 2) the time for the next reaction τ is sampled from a specific distribution; and then 3) the reaction is randomly selected with a probability weighted by the reaction propensities. Once a reaction is selected, the system will be updated based on the chosen reaction, and the process will be repeated.

Partial differential equations

Although ODE provides an easy and fast approach to modeling biological reactions, its intrinsic assumptions of spatial homogeneity might not always hold in more complex systems. Biological systems often exhibit spatial features, such as diffusion or spatial heterogeneity, that might not be fully captured by ODEs. These spatial aspects can play an important role in various biological processes, from intracellular signaling to population dynamics. To account for this, more advanced mathematical models are often necessary to accurately represent and build these complex systems.

PDEs are frequently used to model complex systems that incorporate spatial components. They allow for the representation of both temporal and spatial components, providing a more comprehensive description of biological phenomena. However, PDEs are mathematically and computationally more demanding compared to ODEs. This increased complexity often necessitates more sophisticated numerical methods and a more powerful computing infrastructure to solve and analyze PDE-based models. A PDE represents a relation of multiple independent variables and their partial derivatives $x, y, \dots$ and a dependent variable u , along with the partial derivative of the dependent variable u with

respect to $x, y, \dots$, such as

$$f(x, y, \dots, u, u_x, u_y, \dots, u_{xx}, u_{xy}, \dots) = 0 \quad (2.4)$$

where u is an unknown function to be solved. The variables are denoted by u_x, u_y, u_{xx}, u_{xy} are partial derivatives of u with respect to the variable x, y , such as

$$u_x = \frac{\partial u}{\partial x}, u_y = \frac{\partial u}{\partial y}, u_{xx} = \frac{\partial^2 u}{\partial x^2}, u_{xy} = \frac{\partial^2 u}{\partial x \partial y} \quad (2.5)$$

A basic form of a PDE would look like

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0. \quad (2.6)$$

where t is the time variable and x is the spatial variable. Solving PDE requires defining three components. First, the initial condition, i.e., the initial distribution of species over space, as opposed to the global initial concentration required for ODEs. Second, the boundary condition, which describes the concentration of the species on the system's boundaries. Finally, the system parameters, such as the diffusion coefficient, which represents how quickly a species diffuses through the modeled space, and the reaction terms. Once the spatial distribution of the system is explicitly considered, reaction-diffusion equations become a suitable choice. They define how the concentration of substances changes over time.

It has been used to model many biological processes, such as tumor development [79], tissue formation [80], cell migration [81], and disease spreading modeling [82]. For a general review about the uses of PDE in biology, refer to [83].

2.1.2 Modeling cell migration

Modeling cell migration has advanced our understanding of how cells coordinate between internal and external forces to achieve motility. Mathematical models have offered a framework to integrate the experimental knowledge and testing available hypotheses.

Cell migration is a multi-scale process that requires processes at the cellular and tissue levels to fully understand its complexity. A comprehensive model of directed cell migration should include four events that are essential during the process [84]. The first event is generating a signal that will trigger the migration event. This signal might occur as a secreted chemical element in the environment. An example of this is the chemoattractant, which is a chemical substance that is essential to chemotaxis migration. When the diffusible chemical is uniformly present around the cell, this will lead to a random movement. However, if the diffusible substance is present in a concentration gradient, the cell will undergo a directed motion toward the area of higher concentration. Second, the cell must be able to sense the generated signal and transform this signal into a physical response, such as changes in cell shape or movement. For chemical cues, this is often sensed by cell receptors like GPCRs. Third event is transmitting the signal to the cytoskeleton, which is responsible for carrying out cell movement. This often involves amplification of the signal and polarization of intracellular signaling molecules. Finally, the sensed signal is converted into a force that causes the cell to move in a specific direction.

Cell migration models are available as 2D or 3D, each with its advantages and disadvantages. For instance, a 2D model is simpler and thus faster to run, but it provides fewer details. On the other hand,

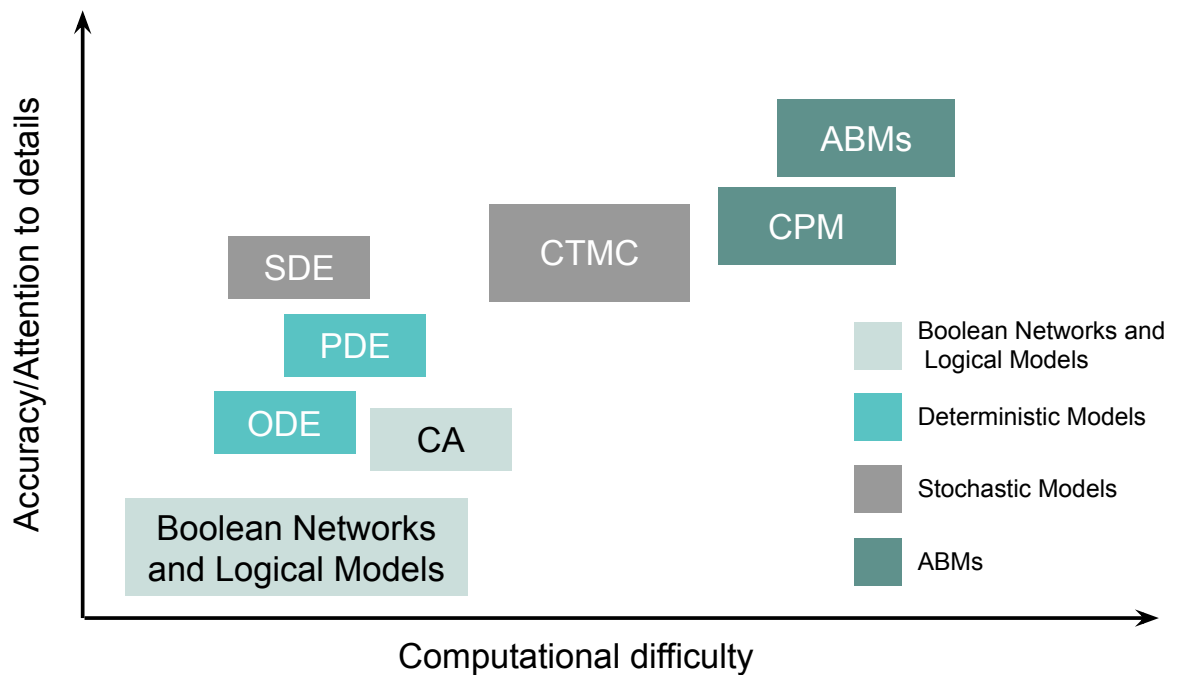


Figure 2.1: A display of the trade-off between the computational difficulty and the attention to detail for different simulation methods

3D models are more complex and computationally intensive, but they can provide a more realistic representation of the cell migration process. Ultimately, the choice between 2D and 3D models depends on the specific research question and the goals of the study.

Cell migration has been modeled using different approaches including CPM [85]. Agent-based models are also used to simulate cell migration, where each cell is an agent that has a set of rules and behaviors [86].

2.1.3 Morpheus

Morpheus is a model development and simulation tool for multi-cellular models [6]. It supports several modeling approaches, including CPM, ODE, PDE, etc., allowing for the representation of different biological processes from cellular to tissue-level dynamics. By coupling these approaches, Morpheus allows biological processes to be integrated on different scales.

One of Morpheus's key strengths is its user interface that facilitates the rapid model construction, editing, and execution, thus allowing non-computational researchers in fields like life science to build models without an advanced technical background.

Since standardized model formats, such as the Systems Biology Markup Language (SBML), do not fully cover the multi-cellular complexity, Morpheus built its own model description format called MorpheusML. MorpheusML is a high-level, fully declarative, domain-specific, and XML-based language that uses bio-mathematical syntax where symbolic identifiers in mathematical expressions are used to describe model components.

Moreover, Morpheus is highly interoperable and can integrate with various computational biology

tools and standardized workflows. It supports importing existing biochemical network models encoded in SBML. Recently, Morpheus has been integrated into broader computational pipelines such as FitMultiCell, enabling parameter inference for multi-cellular models via ABC.

The architecture of Morpheus can be easily extended due to its plugin-based system, allowing users to build customized functionality to suit specific biological questions. Also, it provides built-in features, including automated batch processing, parameter sweeps, and visualization of the simulation result to enable in-depth exploration of complex multi-cellular processes.

Morpheus is broadly applicable, widely used for a variety of biological problems, and computationally efficient (see, e.g., [87], [88], [89] and currently more than 80 models shared in the Morpheus model repository).

2.2 Bayesian parameter estimation

There are an abundance of models in systems biology; however, many of them typically contain unknown parameters that are hard to measure experimentally and must therefore be estimated from experimental data. To make these models accurate and expressive of actual biological processes, Bayesian methods are often used to incorporate prior knowledge and uncertainty into the parameter estimation process. Bayesian inference dates back to 1763, when it was first introduced by Thomas Bayes as the inverse probability problem [90]. It is used regularly to combine our knowledge of the experimental data with the mathematical model to validate the uncertainty of our model.

In Bayesian inference, the available knowledge of the process of interest is encoded as an a prior probability distribution. Bayesian inference transforms the prior probability distributions for unknown models, parameters, and latent variables into posterior distributions [91].

By Bayes' theorem, the posterior distribution is

$$P(\theta | D) = \frac{P(D | \theta)\pi(\theta)}{P(D)} \quad (2.7)$$

where $\pi(\theta)$ denotes the prior distribution and $P(D | \theta)$ denotes the likelihood function, and the $P(D)$ is the marginal likelihood.

If the model has a conjugate prior available, then the numerical calculation will be straightforward because the prior and likelihood belong to the same family, making it analytically tractable. If not, then there are a variety of computational methods for sampling from the posterior, which include Markov chain Monte Carlo (MCMC) and particle filter (PF) methods.

2.2.1 Markov chain Monte Carlo

Markov chain Monte Carlo algorithms provide a way to sample from the Bayesian posterior

$$P(\theta | D) \propto P(D | \theta)\pi(\theta) \quad (2.8)$$

even when the normalization constant $P(D)$ cannot be computed analytically. A classic MCMC method is the Metropolis–Hastings algorithm, which proposes a candidate θ^* that is drawn from a proposal distribution $q(\cdot | \theta)$ and accepts with probability

$$A = \min \left(1, \frac{P(\theta^* | D) q(\theta | \theta^*)}{P(\theta | D) q(\theta^* | \theta)} \right) \quad (2.9)$$

The resulting Markov chain is constructed so that its stationary distribution is the desired posterior.

Besides the Metropolis–Hastings, there exist other MCMC algorithms, such as Gibbs sampling, Hamiltonian Monte Carlo, and particle filtering, but they all require repeatedly evaluating the likelihood $P(D | \theta)$ and the prior to decide how to move through parameter space.

Properly designed MCMC will be able to generate samples from the desired distribution after an initial transition period (burn-in). However, it is hard to pinpoint exactly when the burn-in phase ends. Therefore, during the running time, the Markov chain could fail to converge to its stationary distribution, which results in the fact that the generated samples might not well represent the target distribution. This will affect the result of any inference performed using them. For this reason, it is important to study the output of the algorithm to determine if the MCMC simulation has converged properly.

2.2.2 Sequential Monte Carlo

Sequential Monte Carlo (SMC) is another family of Monte Carlo algorithms. Compared to grid-based methods, SMC is flexible, simple to implement, embarrassingly parallelizable and applicable in very general settings [92].

It is a powerful method that approximates a target distribution by a set of weighted samples (or particles) using a sequence of intermediate distributions [93]. The SMC method provides the approximation assuming that the prior is proper (i.e., it integrates to one) and both the prior and data model can be easily sampled. Each weight determines how well the particle describes the target density. When moving through the intermediate sequence, the particle weights are updated to reflect the transition between distributions, and low-weight particles are replaced through resampling to maintain only the highly informative ones.

Like MCMC, SMC algorithms also depend on simulation and likelihood evaluation for each particle, and each update or resampling step requires running the model to update weights. This process can be computationally expensive, especially in the case of a large number of particles. However, one advantage of the SMC is that each particle can be processed in parallel. Therefore, the method is highly parallel and can be distributed across several computational resources to reduce the overall computational overhead.

2.2.3 Challenges in multi-cellular modeling

Calibrating multi-scale and multi-cellular models comes with an additional layer of challenges. These models include many interacting components and usually express stochasticity to reflect biological variability. This increases the parameter dimensionality and thus slows convergence. The number of state variables and parameters grows as the complexity of the system increases, leading to highly complex systems. This complexity makes the likelihood function $P(D | \theta)$ either prohibitively expensive to compute or intractable to calculate. For example, in an agent-based model, computing the likelihood $P(D | \theta)$ of the observed data D given parameters θ would require integrating over all possible unobserved latent trajectories the system could have taken. This integral is extremely high-dimensional and cannot be computed analytically. As a result, sampling methods like MCMC

and particle filters, which require evaluating $P(D | \theta)$, cannot be applied directly to most multi-scale and multi-cellular models. However, in cases where the calculation of the likelihood function is intractable due to complex models or large datasets, approximate methods, such as the likelihood-free inference, are used that do not directly calculate the likelihood function.

2.3 Likelihood-free inference

Likelihood-free inference methods have emerged for unnormalized models (e.g., [94, 95]). These methods are based on the assumption that the statistical model generating the data is unknown. They have evolved in multiple ways in different disciplines, all sharing the idea of choosing the model's parameters that best fit the data. It includes Approximate Bayesian Computation (ABC) in genetics [96–98], indirect inference in economics [99], and synthetic likelihood in ecology [100].

The core idea behind the likelihood-free inference is that one can replace the analytical calculation of the likelihood with the assumption that one can still sample from the model. Methods like the approximate Bayesian computation accomplish that by generating summary statistics of simulated data that are close enough to the summary statistics of the real data [101]. In the indirect inference, one constructs a simpler statistical model with its parameter vector ϕ , distinct from θ , the parameters of the model of interest, that can be fitted to the real data using the likelihood-based methods, then using ϕ as a set of informative summary statistics for the original complex model [102]. Other methods, like the synthetic likelihood method, achieve that by trying to estimate the likelihood $P(D|\theta)$ and then using the estimation result in the inference process (like MCMC) to generate the posterior $P(\theta|D)$.

2.3.1 Approximate Bayesian computation

A popular likelihood-free inference method is approximate Bayesian computation, which is a widely applicable approach to estimating unknown parameters of mechanistic models [96, 103]. It is useful in cases where the likelihood function is computationally expensive or impossible to calculate. It generates samples from an approximation to the true Bayesian posterior distribution. The samples can then be used to make inferences about the parameters. A basic ABC algorithm is presented in Figure 2.2.

A main trade-off that one needs to understand about the summary statistics is that it sacrifices some information of the full dimensional data, meaning the resulting posterior is conditioned on the summary statistics rather than the full data. However, when relying on the summary statistics, one can escape the curse of dimensionality by reducing the space of the inference problem, making comparison between simulated and observed data possible even when the raw data are extremely high-dimensional.

While asymptotically exact, ABC suffers from two main limitations [104, 105]. First, they often have a high computational cost. Second, the assessment of differences between the simulated and observed data involves the choice of the summary statistics and the distance function, which both can play a key role in the quality of the inference. Thus, different approaches and enhancements to ABC have been developed to address this limitation.

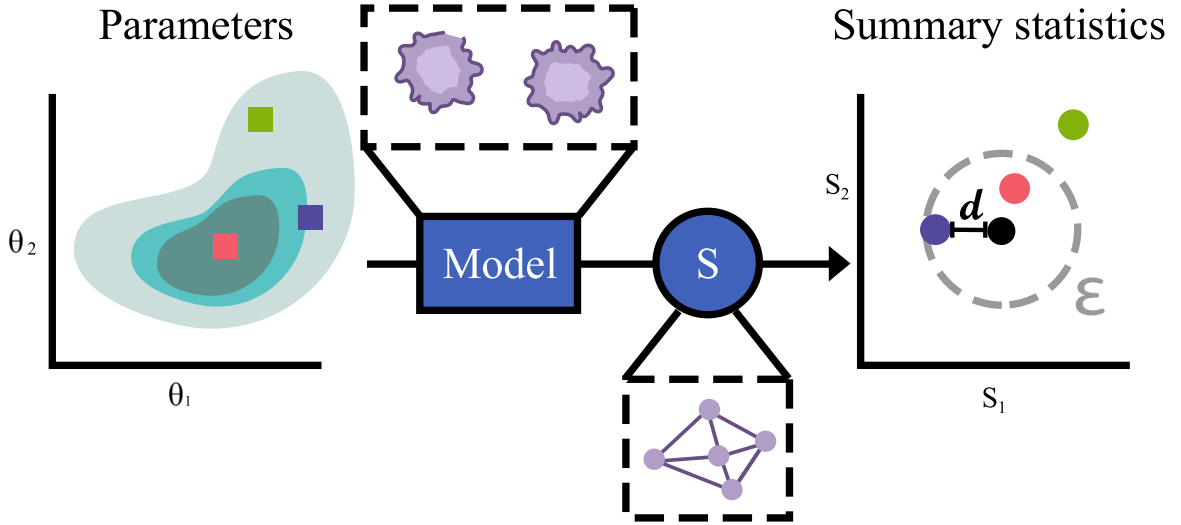


Figure 2.2: Basic ABC algorithm. Each parameter $\theta \sim \pi(\theta)$ is sampled either from the prior or a proposal distribution. Then, it is used to simulate the process from a black-box simulator. One can add an optional step to use summary statistics, which reduce the size of the simulated data, giving a low-dimensional representation $S(y)$. The distance d between the summary statistics of simulated and observed data is calculated, and the sampled parameters are either accepted or not based on an acceptance threshold ε .

2.3.2 ABC rejection sampler

The simplest ABC algorithm is the ABC rejection sampler [106]. The basic concept is that it proposes parameter vectors θ^* from a suggested prior distribution $\pi(\theta)$, then simulates the data D^* using the parameter vector θ^* , and only accepts those vectors whose simulated data closely match the observed data based on some distance function $d(D_0, D^*) \geq \varepsilon$, where ε is the tolerance threshold, see Algorithm 1. However, in cases where the data are of high dimensions, a lower-dimension summary statistic s^* can be used instead to reduce dimensionality while still capturing the essential features of the data. The rejection sampler requires some tuning parameters: the number of simulations N , a threshold $\varepsilon \geq 0$, a function $S(y)$ mapping data into lower-dimensional summary statistics, and a distance function $d(s^*, s_{obs})$.

Algorithm 1 ABC rejection Sampler

1. sample θ_i^* from $\pi(\theta)$ for $1 \leq i \leq N$
 2. simulate D_i^* from $P(D | \theta)$ for $1 \leq i \leq N$
 3. calculate $s^* = S(D^*)$ for $1 \leq i \leq N$
 4. calculate $d^* = d(s^*, s_{obs})$ where $s_{obs} = S(D_{obs})$
 5. if $d < \varepsilon$ store θ , go to step 1 and repeat
-

The limitation of ABC rejection is that it might have a low acceptance rate if the high probability region is different between the prior and the posterior one. To circumvent this, another flavor of ABC was introduced, the ABC Markov chain Monte Carlo [97].

2.3.3 Markov chain Monte Carlo ABC (ABC-MCMC)

ABC-MCMC is an upgrade to the ABC rejection, as it does not sample from the prior but rather from a distribution that is closer to the posterior than from the prior. An ABC-MCMC algorithm based on [97], will work as in Algorithm 2.

Algorithm 2 ABC MCMC

1. initialize $\theta_i, i = 0$
 2. sample θ^* from a proposal distribution $q(\theta|\theta_i)$
 3. simulate a dataset D_i^* from $P(D | \theta)$
 4. calculate $s^* = S(D^*)$ for $1 \geq i \geq N$
 5. calculate $d^* = d(s^*, s_{obs})$ where $s_{obs} = S(D_{obs})$
 6. if $d^* < \varepsilon$ continue, otherwise, go to the step 8
 7. Set $\theta_{i+1} = \theta^*$ with the probability

$$\alpha = \min \left(1, \frac{\pi(\theta^*)q(\theta|\theta^*)}{\pi(\theta_i)q(\theta^*|\theta_i)} \right)$$
 and $\theta_{i+1} = \theta_i$ with a probability $1 - \alpha$
 8. Update $i = i + 1$ and go to step 2
-

In the ABC-MCMC algorithm, one will start by sampling parameters from the prior distribution $\pi(\theta)$, however, in the next iteration, one will sample from a proposal distribution $q(\cdot)$ that depends on current θ . Then, one will simulate a dataset using the sampled parameter vector (θ) and model $f(\cdot)$. After that, one would calculate summary statistics from the simulated data set and for the observed data and will accept θ only if the distance function is less than the acceptance threshold $d > \varepsilon$. By doing so, it improves the acceptance rate because the proposal distribution adaptively targets parameter regions closer to the posterior than the broad prior.

However, ABC-MCMC suffers from a number of challenges, one of which is the choice of the acceptance threshold ε . A small value will lead to a lower acceptance rate, which will lead to many proposals θ being rejected, while a large one will give θ that poorly resembles the posterior distribution. This motivated the develop the ABC-SMC approach.

2.3.4 Approximate Bayesian computation - sequential Monte Carlo

To efficiently explore the search space, ABC is often combined with a sequential Monte Carlo scheme. The ABC-SMC scheme [15, 98] can do the exploration efficiently by successively refining the posterior approximation via importance sampling while maintaining high acceptance rates. Furthermore, in ABC-SMC, the sampling for each generation can be parallelized because the candidate samples in each generation can happen independently in parallel, enabling the use of high-performance computing (HPC) infrastructure. This has, in recent years, enabled tackling increasingly complex problems via ABC-SMC. An algorithm of ABC-SMC is working as in Algorithm 3.

Similar to ABC-MCMC, one would sample from the prior, then in subsequent iterations from the proposal based on θ . The set of samples can be treated as sampled from an approximation of the posterior. One then will fit a density kernel $\sum_{j=1}^N w_{t-1}^{(j)} K_t(\theta_{t-1}^{(j)}, \theta_t^{(i)})$ around samples and then resample values from this. A typical choice of the perturbation kernel K is Gaussian, but others are used [15, 107]. Each obtained parameter θ has a weight w assigned to it to account for the fact that these parameters were not sampled directly from the prior distribution. This weighted (importance weight)

Algorithm 3 ABC SMC

1. initialize $\varepsilon_0, \dots, \varepsilon_T$
2. Set indicator $i = 1$
3. If $t = 0$, sample θ^{**} independently from $\pi(\theta)$
Else, sample θ^* from the previous population $\theta_{t-1}^{(i)}$ with weight w_{t-1} and perturb the particle to obtain $\theta^{**} \sim K_t(\theta|\theta^*)$, where K_t is the perturbation kernel.
4. If $\pi(\theta^{**}) = 0$, return to step 3
5. simulate a dataset D_i^* from $P(D | \theta^{**})$
6. calculate $s^* = S(D^*)$ for $1 \leq i \leq N$
7. calculate $d^* = d(s^*, s_{obs})$ where $s_{obs} = S(D_{obs})$
8. if $d^* < \varepsilon_t$ continue, otherwise, go to step 3
9. Set $\theta_t^{(i)} = \theta^{**}$ and calculate the weight for particle $\theta_t^{(i)}$,

$$w_t^{(i)} = \begin{cases} 1, & \text{if } t = 0, \\ \frac{\pi(\theta_t^{(i)})}{\sum_{j=1}^N w_{t-1}^{(j)} K_t(\theta_t^{(j)}, \theta_t^{(i)})} & \text{if } t > 1 \end{cases} \quad (2.10)$$

10. if $i < N$, update $i = i + 1$, and go to step 3.
 11. Normalize the weights.
 12. If $t < T$, set $t = t + 1$, go to step 2.
-

set of parameters provides an improved approximation of the posterior distribution. In addition, at each iteration, the acceptance threshold ε is reduced. This process is repeated for additional iterations, further refining the approximation.

pyABC: Python package for likelihood-free inference

pyABC is an open-source Python package that provides a framework for Approximate Bayesian Computation [28, 108]. At its core, it implements the ABC-SMC approach based on [109] with many algorithms that facilitate robust and efficient inference for a broad spectrum of applications via robust methods and automatically find the hyper-parameters. pyABC can scale with expensive problems through the use of a single machine with a multi-core implementation through Python's multiprocessing capabilities to multiple-machine distributed execution using integration with popular frameworks such as Redis. Besides that, it provides several parallelization strategies for the sampling of particles in a population of simulations: static [110–112], dynamic [27], and look-ahead scheduling [24] to provide more optimized execution that preserves computational resources.

The package is built for non-expert users with many predefined hyper-parameters and an extensive set of documentation (<http://pyabc.readthedocs.io>) that covers most of the system's needed features. Yet, it enables the expert to have full control by building a customizable setup of many options of the ABC-SMC scheme. It was built to be modular for better usability and extendability.

pyABC provides many features, including adaptive transition kernels and population size, accounting for noise, dynamically scale-normalise distance metrics, handling outliers, investigating data informativeness, etc. In addition, it provides an interface to multiple languages and tools, including Python, R, Julia, COPASI simulator, Morpheus simulator (through FitMultiCell), and PTab standard. All of this has enabled pyABC to cover a wide range of applications in e.g., cardiac electrophysiology

[113], human atrial cells [114], cancer [115], gene expression [116], universe expansion [117], and bee colonies [118].

In addition, pyABC includes a built-in interactive web server that helps users to visualize the result of the inference run easily. In a few clicks, the user can specify the inference output and get instant insights about the run's posterior distributions, acceptance rates, convergence diagnostics, and model comparisons.

It is written in Python 3 and is released under a 3-clause BSD license. The open-source codebase is actively maintained and hosted on GitHub (<https://github.com/icb-dcm/pyabc>) and can be easily downloaded from the package index (PYPI), simplifying deployment across various computing environments. The author of this thesis contributed to the building and the maintaining of this package

2.3.5 Parallelization strategies for ABC-SMC

ABC is widely applicable, yet it requires many simulations to approximate the posterior distribution accurately. The computational complexity is one of the main limitations of this method. Thus, a parallel implementation is needed for more efficient computation.

Fortunately, ABC-SMC is an embarrassingly parallelizable problem, which means it is easily divided into subtasks that can be divided across several computational resources, nodes, or cores, where there is no information exchanged between these tasks. A popular approach is to parallelize the simulation and the assessment of the objective function of the same generation; see [16]. A generation refers to an iteration during the sequential search for the parameter values. Once a generation is finished, the parameter proposal distribution typically changes, and a new set task is generated with a new proposal.

There are two strategies on how many tasks are assigned to each resource in case the number of resources is less than the number of tasks [27]. The first method is very straightforward, a centralized scheduler assigns an equal number of tasks to each computational resource. However, in case of having a variant simulation time, a resource might end up being a bottleneck and slow down the entire process. The risk will increase as the variation of the simulation time increases. The second method tries to solve this by having a pool of tasks that are distributed to resources only if they are available and have completed their previous task, allowing for a more efficient allocation of resources. The ultimate goal of this method is to try to keep all computational resources busy and minimize the overall idle time.

2.3.6 Accelerating ABC-SMC through HPC resources

While the parallelization strategies, such as dynamic scheduling and adaptive task allocation, have improved the computational efficiency of ABC-SMC, scaling to large problems can only be achieved through the use of large HPC systems. HPC systems enable running large tasks across a large network of connected processing units, fast memory, and efficient communication infrastructure, enabling the massive parallelism in simulation-based inference.

There are several technologies for exchanging information between the resources within the node and across several nodes in an HPC cluster. Shared memory parallelism can be used to exploit multiple CPU cores within a single HPC node. One of the most popular technologies in this area is OpenMP [119], which distributes work among processor cores to allow for multithreaded simulation task execution. For simulation codes written in C, C++, or Fortran, where loops or independent

computations can be parallelized with straightforward compiler directives, OpenMP is especially helpful. Parallel execution inside a node is usually accomplished in Python-based environments with the multiprocessing module, which creates independent processes free from Python's Global Interpreter Lock (GIL). This allows full CPU utilization on modern multi-core processors.

However, to send messages across different clusters' nodes during the ABC-SMC inference, one needs inter-node communication protocols. There exist many, including Message Passing Interface (MPI) [120]. MPI is a standard communication protocol between distributed memory systems. An alternative distributed communication strategy is to use an in-memory data storage such as Redis (<https://redis.io/>), where different nodes can have access to the shared database to pull tasks and push results. This simplifies the management of sending and receiving information.

Besides the parallel strategies, running a large-scale ABC-SMC pipeline in a large cluster requires the orchestration of multiple jobs and resources. Many HPC systems use a workload manager, such as the SLURM (Simple Linux Utility for Resource Management) to organize this process. These systems address node-level execution, resource allocation, and job queueing. Thus, within the inference pipeline, one can initiate several jobs to cover simulations, parameter inference, summary statistic calculation, etc., to submit or schedule for execution.

2.3.7 Surrogates and emulators to accelerate simulation-based inference

Surrogate models, also known as emulators or meta-models, are based on machine learning techniques to approximate the mechanistic models in systems biology that are based on ODE, PDE, SDE, etc. They take the model parameter and its initial condition as input to partially or fully predict the output.

To build a surrogate model, one first needs to train and validate the model, and then the proxy version can replace the main mechanistic model without the need to simulate the mechanistic model anymore. This usually improves the computational complexity by several orders of magnitude [121]. An important question to ask when building a surrogate model is what output is needed to be predicted and which input parameters of the original model should be varied during training. In many cases, we do not need to predict every output of the original model, so we focus on the area of interest to speed up the training phase [121]. After deciding what to predict and which model's inputs to vary, one needs to run multiple simulations from the original model. The generated simulations will create the input-output pairs to train and test the surrogate model. Typically, 80–90% of these data are used to train the model, and the remaining 10–20% is reserved for testing its precision [121].

Several approaches to constructing a surrogate model have recently been developed. One popular approach is the Gaussian Process (GP), a Bayesian approach that deals with the unknown relation as a stochastic process. Other approaches are the Support Vector Regression (SVR), the Radial Basis Function (RBF) interpolation, and Neural Network-based models. Each of the previously mentioned methods has its advantages and disadvantages depending on the complexity of the biological system, the dimensionality of the parameter space, the availability of training data, and the necessity for interpretability or uncertainty quantification.

Surrogate models have been used in systems biology in different areas. One usage is during parameter estimation [122] and sensitivity analysis [123], where the surrogate model can replace a costly model during Bayesian inference for fast simulation. Yet, they can also be used for real-time clinical decisions, where there is a need for quick predictions [124].

2.4 Amortized inference

Amortized inference, a machine learning technique, uses neural networks to approximate the intractable posterior distributions. It accelerates subsequent inference by using calculations done upfront. Instead of performing inference on every new data point or query, it trains a proxy model, i.e., a density estimator, that can be trained once and then used to approximate the posterior distribution. The estimator's two main purposes are to represent the complex posterior distribution over the whole range of parameter space and to estimate the posterior probability of parameters given observed behavior.

Once trained, the amortized inference network will be able to directly evaluate the posterior $P(\theta \mid D)$ of any given observation D without the need to run the expensive simulation. One can justify the upfront simulation for training the network in cases where the inference process needs to be performed on multiple datasets or in real-time settings.

Amortized inference is commonly performed with neural density estimators trained on simulated data. Three families are widely used. The first is neural posterior estimation (NPE) [125]. It learns a direct approximation to the posterior $P(\theta \mid D)$ of parameters given the observed data with a flexible density model (e.g., normalizing flows). The second method is the neural likelihood estimation (NLE), which trains a conditional density estimator to approximate the likelihood function $P(D \mid \theta)$ from simulations, then an additional MCMC step can be used to obtain the posterior [126]. The third method is the neural ratio estimation (NRE), which approximates the posterior density $P(\theta \mid D)$ indirectly by learning a density ratio that is proportional to the likelihood via classification on simulated samples; the learned ratio is sufficient for Bayesian inference [127]. Similarly to NLE, an additional MCMC sampling step can be used to generate samples from the surrogate posterior [128].

However, this efficiency comes at the cost of two limitations. 1) The quality of the inference depends on the quality of the training phase; and 2) the computational cost of the training phase [129]. It is important to carefully consider these trade-offs when implementing this approach in practice. One needs to generate a large number of simulations to train the network, which scales significantly if the simulation is costly or the parameter space to cover is big.

The ideal case to use amortized inference is when inference must be repeated across many datasets (such as high-throughput experiments) in a scenario like a real-time setting. Training typically draws $\theta \sim P(\theta)$ and simulates $D \sim P(D \mid \theta)$ to form training pairs; sequential schemes refine the proposal to cover parameter regions supported by the data. After training, posterior queries are fast (single network pass), enabling rapid uncertainty quantification at scale.

2.5 Standardization and reproducibility

While there is a continuous improvement in the analytical tools, it is also important to emphasize standardization and reproducibility in order to ensure the findability, accessibility, and reusability of data and models across different studies. Various communities may adhere to different standards and practices. Computational biology often uses different data formats such as images, omics data, and experimental assays. This mixture of data formats overcomplicates the effort of developing a standardized workflow. The community of systems biology, for example, has made efforts to establish guidelines for reproducibility in order to promote transparency and trust in research findings [130].

The community effort has led to the creation of standardized model description formats such as the Systems Biology Markup Language (SBML, <http://www.sbml.org/>), a common XML-based format

that used to represent and exchange quantitative models in systems biology; the Cell Markup Language (CellML, <http://www.cellml.org/>), another XML-based format that focuses on representing models that describe cellular processes; the Biological Pathway Exchange (BioPAX, <https://www.biopax.org/>), a standard format that facilitates the representation of biological pathways, such as the metabolic pathways; the BioNetGen Language (BNGL, <https://bionetgen.org/>) a rule-based framework that helps in building biomedical reaction networks; and PEX (https://github.com/PEX-dev/PEX), a data format that is suited for parameter estimation problems in systems biology, including model, parameter, and experimental data.

Another important factor that improves the reusability is the rich metadata. An effective model annotation is an essential step in improving the reproducibility of the simulation studies. This means that one would need to map model components into the standard components, such as genes, proteins, etc., and to capture all contextual details that will be needed to understand the model.

To coordinate the effort across the field, The COMBINE (COmputational Modeling in BIOlogy NEtwork) initiative was started in 2010 to ensure that new standards are compatible and meet the standard requirements [131]. This initiative has been able to successfully launch many of the above-mentioned standards.

Aside from the community effort to build a single standard format, the community has proposed a composite format to improve data interoperability. An example is the COMBINE Archive (<https://combinearchive.org/index/>), which is a format that packages all files needed to repeat the simulation study, such as the model (SBML or CellML), the associated datasets, and the simulation experiment descriptions.

To facilitate model sharing, a public database was needed that stores the models and their corresponding data. One of the initiatives to fill this gap was the BioModels, which is a free and open-source platform for storing and exchanging biological models (<https://www.ebi.ac.uk/biomodels/>). To store material beyond the biological models, the FAIRDOM (<https://fairdomhub.org/>) initiative has been introduced to enable the organizing and sharing of models, datasets, protocols, software, and pipelines. They promote assets that follow the FAIR data principle, which stresses that scientific outputs should be Findable, Accessible, Interoperable, and Reusable.

Adopting the field standard is not only a technical necessity but also a fundamental step in ensuring that research findings can be easily shared, replicated, and built upon by other researchers in the field. This, in turn, can help build the field's growth and credibility.

2.6 Cell migration

Cell movement is a highly dynamic phenomenon that is essential for many biological processes. It was first noted by Van Leeuwenhoek in 1675 when he noticed the cell crawling across different microscope slides. However, not until a few decades back, these movements were not the focus of the scientific community [132]. The breakthroughs in fluorescence microscopy, biochemistry, and molecular biology have further pushed our understanding by explaining the main process behind the motility process and identifying key proteins in this process. On the other hand, computational modeling and theoretical studies have helped to quantify the suggested mechanisms, and the forces generated at the smaller level are combined to generate the observed movement.

2.6.1 Overview of the cytoskeleton in cell migration

Once a cell is triggered by an external signal from its surroundings, it will start the movement process. The signal is then detected by receptors on the cell membrane, which activate intracellular signaling cascades that ultimately lead to cytoskeletal rearrangements and cell movement.

Once movement is started, most cells will begin to alternate between three main stages, which will result in the reconstruction of the cytoskeleton [133]. First, the cell extends its membrane by growing its protein network in the direction of movement. Second, it will stabilize the edge toward the direction of movement and detach from the opposite direction. Finally, the contractile forces pull the cell forward.

As cells travel, they are faced with many external forces, e.g., viscous forces, and internal forces, e.g., forces generated by the cytoskeleton. The cytoskeleton is a complex and dynamic network of protein filaments that exists in the cell's cytoplasm [132]. It provides structural support and facilitates cell movement by coordinating the forces applied on the cell. It consists of three distinct filament systems: actin, microtubules, and intermediate filaments [132].

Actin filaments are an important elements in whole-cell locomotion and control changes in the shape of cells. The cell uses actin-binding proteins to control how the actin filaments assemble and disassemble and connect to other parts (such as the plasma membrane) [134]. These proteins are important components of the actin cytoskeleton. Actin filaments form a network at the front of the cell that provides multiple benefits, including mechanical support, adjusting the cell shape, and allowing the cell to move.

Microtubules also play an important role in cell movement by determining the positions of intracellular organelles and directing intracellular transport. Similar to the actin filaments, microtubules are dynamic and go through different phases of assembly and disassembly. They are responsible for a variety of cell movements, including intracellular transport, positioning of the membrane vesicles and organelles, and the separation of chromosomes during mitosis phase [134]. Microtubules are a polar structure that has one end that expands rapidly, while the other grows slowly. This polarity behavior is what determines the directionality of movements.

Finally, unlike the actin filaments and the microtubules, the intermediate filaments do not directly contribute to cell movements. Intermediate filaments contribute to the cell's mechanical stability and resistance to shear stress [134]. They are made up of different types of proteins that are expressed in different types of cells. Together, these three filaments work and coordinate to provide different movement and migration behaviors in cells.

2.6.2 Experimental methods for studying cell migration

The behavior of cell movements under the influence of chemotaxis can be studied using many techniques, one of which is the Boyden chamber. The Boyden chamber contains two compartments; the cell is placed in the upper compartment and then allowed to migrate to the second, where the chemotactic is placed on the other end one [135].

Another technique is the under-agarose chemotaxis assay. Under-agarose [136] is famous for being an easy, cheap, and robust technique that can measure cell movement parameters such as directionality and speed. Coupled with high-resolution imaging techniques, such as confocal microscopy, detailed measurements of cell components, such as the membrane proteins, actin, etc., could be quantified.

Quantifying cell trajectory during the movements can be done by tracking time-lapse microscopy.

Manual tracking is done by tracing the cell in each image using the center of mass to determine the x, y, or even z position in 2D or 3D space. This process can be done using tools and plugins like ImageJ [137] and Fiji [138]. Although the manual tracking yields good results, it is labor-intensive, which may result in poor reproducibility. To improve that, several tools and applications have been introduced to automate the process of tracing the cell position over time, such as [139–141]. A detailed review of tracking tools could be found here [142].

Integrated Pipeline for Simulating and Inferring Unknown Parameters of Multi-scale Processes

3.1 Motivation

The dynamics within tissue arises from the interconnection between different processes on different scales. Multi-scale and multi-cellular models can capture that, but they need to be calibrated on data to be useful so that they can predict responses to perturbations, compare hypotheses, and quantify uncertainty. Advanced methods that do not rely on the calculation of the likelihood function (such as ABC), which suits the multiscale and multi-cellular models, are usually computationally heavy. Also, the need for a standardized software pipeline that can integrate simulations, calibration, and result diagnostics of these models is essential not only for reproducibility and collaboration, but also for facilitating the adoption of multi-scale modeling in the computational biology field.

This chapter introduces *FitMultiCell*, the central software contribution of the thesis, an end-to-end pipeline for simulating and calibrating stochastic multi-cell and multi-scale models. This novel pipeline facilitates the simulation and parameter estimation of biological multi-scale models. We tested and evaluated the pipeline on several models of different biological processes, such as tumor progression, cell migration, and infection spread. The pipeline was run on a variety of high-performance infrastructures with different setups to demonstrate its capabilities.

I developed the pipeline and tested it on a large computational infrastructure, applied it to real applications, and validated its results. The paper's novelty lies in building a full pipeline that starts with defining the problem in a standardized format, to the construction of multi-scale models, calibrating the model against observed data, and quantifying its result.

This summary focuses on the pipeline methodology and the application example. It is based on the following first-author publication of the thesis author, which is also reprinted in Appendix 7.1:

- **Alamoudi, E.**, Schälte, Y., Müller, R., Starruss, J., Bundgaard, N., Graw, F., Brusch, L., & Hasenauer, J. *FitMultiCell: Simulating and parameterizing computational models of multi-scale and multi-cellular processes*. *Bioinformatics* 39.11 (2023). doi.org/10.1093/bioinformatics/btad674

3.2 Methodology

In this work, we are addressing the quantitative modeling of stochastic multi-cellular processes by writing any model as

$$y = \mathcal{M}(\theta, \xi) \quad (3.1)$$

where y is the vector of the observed properties and θ is the vector of unknown parameters. Noise, from low molecular number or measurement, is denoted by the vector of random variables ξ . By marginalizing over the simulator's randomness ξ , we obtain the likelihood $\pi(y|\theta)$, which is the probability of observing y given θ .

Morpheus is a state-of-the-art tool for simulating multi-scale and multi-cellular systems using a variety of mathematical frameworks. Morpheus covers the simulation aspect of multi-scale and multi-cellular biological systems. On the other hand, pyABC covers the estimation and model calibration aspects using likelihood-free Bayesian parameter inference methods. By tightly coupling Morpheus and pyABC, FitMultiCell enables automated generation of model simulations and statistical comparison to data.

3.3 Significance within the thesis

The FitMultiCell pipeline provides an end-to-end workflow. Starting with a standardized format for both the multi-scale models (MorpheusML) and the estimation problem (PEtab-MS). These standard data formats make the model calibration process easier and encourage reproducibility and collaboration, aligning with the FAIR data principles, leading to greater adoption of the culture of open and transparent modeling in the field. Then, it enables the construction and simulation of biological processes at different scales to include cellular (CPM/CA), extracellular (PDE), and intracellular (ODE/SDE) dynamics. Then, it provides parameter estimation for the model's unknown parameters and quantifies the results.

The introduced pipeline was developed to handle all quantitative modeling, hypothesis testing, and predictions of dynamic responses to perturbations for multi-scale models. Its modular nature covers the entire cycle from model construction to simulation to parameter estimation.

The scalability of the pipeline is achieved through different layers: i) it parallelizes the simulation task across different resources in the shared memory systems, ii) it parallelizes individual evaluation of summary statistics, and iii) finally, it parallelizes the ABC-SMC population through a shared or distributed memory system.

A user-friendly interface for visualizing and diagnosing the fitting result is available along with a computing cluster script that supports running the pipeline on large clusters. The pipeline is implemented in Python and is available as open source (BSD-3) and can be directly installed via PyPi. Its detailed online documentation, along with its modular interface, lowers the entry barrier for non-technical researchers to explore and study complex models.

One of the key features of FitMultiCell is its focus on scalability and large computer cluster support. It is built to support multiple levels of parallelization across different computing configurations. This allows FitMultiCell to tackle computationally intensive problems by extending computational capability across shared and distributed memory systems. This ensures completion of the parameter estimation task, which would otherwise take longer to finish, and will accelerate the research cycle.

Another main feature is its usability. We offer an easy user interface and modular structure that allows non-expert users to use the pipeline without extensive technical knowledge. The FitMultiCell pipeline comes with a GUI that allows users to explore the estimation results and plot different diagnostic figures to assess the quality of the estimation.

In addition, FitMultiCell extends its reusability by using the standardization format for model construction, MorpheusML, and the parameter estimation problem definition, PETab-MS. These formats align with FAIR principles and promote benchmarking and sharing.

The pipeline offers several methodological contributions in the area of multi-scale modeling in systems biology:

- **End-to-end workflow orchestration:** The pipeline successfully integrates several steps into one unified workflow that covers construction, simulation, parameter estimation, and fitting diagnostics.
- **Integrating modeling and estimation tools:** instead of reinventing the wheel, FitMultiCell built on robust and state-of-the-art tools and connected them in a novel way.
- **Standard problem specification:** We introduce PETab-MS as a new standard for capturing all needed information for the parameter estimation problem for multi-scale models. FitMultiCell did that by extending the original PETab standard, originally developed for ODE-based models, to support multi-scale models.
- **Scalability through parallelization and optimization:** To enable the ABC approach in large-scale multi-scale models, FitMultiCell orchestrates the distribution of all tasks in large computer clusters. We also enhance parallel execution by further optimizing the ABC-SMC approach. We introduce a new ABC-SMC algorithm that leads to better utilization of the computing resource, yielding better performance.
- **Enhanced Usability and Accessibility:** A major goal was to make this advanced method available to a wide range of researchers without the need for a deep computational background.

The FitMultiCell pipeline is the central point of this thesis and will be used in the second paper of this cumulative thesis (see Section 7.2) and the third paper (see Section 7.3) where we apply it to models of infection spreading, cancer growth, organ regeneration, and cell migration under the influence of chemokine. It has also been used in different publications in different biological contexts, ranging from the study of the migration characteristics of cancer cells driven by pseudopod interaction [143] to the investigation of beta-cell apoptosis at the molecular level [144], and to analyze collective cell migration under the influence of guidance-by-followers [145, 146].

3.4 Application to Biological Case Studies

To test the FitMultiCell pipeline, we apply it to multiple biological models, each highlighting different capabilities. In the original manuscript, three models (M1-M3) were fitted using the pipeline. The first model was for the spread of viral infection (M1). It describes the process of virus transmission among a healthy cell population. This model illustrates a typical virology experiment in which the transmission spreads through a direct neighboring cell by cell-to-cell infection or through infecting a

far-away cell through the cell-free mechanism. The second case study was for tumor spheroid growth (M2). It highlighted a tumor spheroid model. The model includes processes such as cell proliferation, division, and death. The model was converted from its original form into the MorpheusML language, where the FitMultiCell pipeline was used to fit some key parameters. The result of the case study shows that FitMultiCell can replicate the results of the previous study and can even enhance them. The final case study was for liver regeneration and mechanosensing (M3), which models the liver tissue regeneration process. This model demonstrates a complex multi-scale model where intercellular signaling affects cell development and vice versa.

All previously mentioned case studies demonstrate the FitMultiCell pipeline's capability to simulate and infer complex multi-scale models. They were also used to test the pipeline scalability on different computing configurations ranging from one node to multi-node clusters.

3.5 Author's contribution

I designed and built the pipeline integrating the simulation and inference components, including data formation, simulation wrappers, result checking, early rejection mechanisms, etc. I extend the support of the PTab format to cover multi-cellular settings specifying models, parameters, priors, observables/measurements, and distance functions. I built the distributed orchestration layer (worker management, job queueing, and fault tolerance), prepared cluster configurations, and ran scaling studies to assess utilization and speed-up. I implemented diagnostics and visualization tools to assess the quality of the fit. I implemented some of the case studies using MorpheusML and converted all estimation problems to PTab-MS to validate the robustness of the pipeline. I developed online documentation for all aspects related to the pipeline to ensure easy onboarding. Finally, I wrote the first draft of the manuscript, contributed to its review and editing, and prepared the figures and illustrations that were used throughout the manuscript. Collaborators contributed to co-developing the pipeline, building case studies, providing supervision, theoretical guidance, and additional manuscript revisions.

Parallelization Approach to Minimize Wall-Time for Approximate Bayesian Computation

4.1 Motivation

Approximate Bayesian Computation with Sequential Monte Carlo has become widely popular for complex models where the likelihood function is intractable. Yet, it suffers greatly from the computational cost generated by the large number of simulations carried out. Distributing the workload of simulation across large computational resources helps shorten the overall computational time for calibration. Yet, using existing parallelization schemes, the utilization of the computational resource may not be optimized efficiently, leaving room for potential improvements in speed and efficiency. For example, at the end of each generation (t), when only a small number of simulations need to be carried out, a large number of resources become idle, resulting in unused resources that could have been allocated more effectively. The motivation of paper 7.2 is to reduce the idle time by pre-calculating and assigning the work of the next generation ($t + 1$) of particles before the end of this generation t , and keeping the otherwise idle/waiting resources busy executing simulation for the next generation.

We propose a look-ahead scheduling algorithm to achieve overlap work across different ABC-SMC generations and keep all workers fully utilized. The new strategy advances working on the next generation of simulation of candidate parameters that are likely to be needed on the next iteration. As soon as the generation reaches its end, where some simulation is still being executed yet no new tasks are being produced, a preliminary task will be generated from the next generation proposal. When the particle of the current generation is completed and the proposal for the next generation is calculated, the precomputed particles are used only if they are compatible with the finalized criteria. This transforms the generation-by-generation approach into a continuous flow of simulation tasks, reducing idle time across computational resources.

In this work, we present our proposed parallelization strategies that increase the computational efficiency of ABC-SMC. The proposed strategy was tested on toys and real application examples to compare its performance against existing well-established strategies.

This summary covers the methodology of the work, the application example, and the significance of the work. This summary is based on the following shared first-author publication of the thesis author,

which is also reprinted in Appendix 7.2:

- **Alamoudi, E.**, Reck, F., Bundgaard, N., Graw, F., Brusch, L., Hasenauer, J., & Schälte, Y. A Wall-time Minimizing Parallelization Strategy for Approximate Bayesian Computation. PLOS ONE 19.2 (2024) 1. doi.org/10.1371/journal.pone.0294015.

4.2 Methodology

Here, we tackle the problem in a Bayesian context when the likelihood is intractable, yet the model simulation $\pi(y|\theta)$ is possible. The ABC-SMC method (see Section 2.3.1) approaches this problem by simulating the model, using parameter vectors θ , and only accepting those that are similar to the observed data by a tolerance threshold ε . In the initial generation ($t = 1$), particles are drawn directly from the prior distribution, $g_1(\theta) = \pi(\theta)$. In subsequent generations ($t > 1$), particles are drawn from proposal distributions $g_t(\theta) \gg \pi(\theta)$, which are derived from the accepted weighted population P_{t-1} of the previous generation, for instance, through a kernel density estimate. The importance weights w_t^i are given by the Radon-Nikodym derivatives

$$w_t(\theta) = \pi(\theta)/g_t(\theta). \quad (4.1)$$

This gives an approximation $\pi_{\text{ABC},\varepsilon}(\theta|y_{\text{obs}})$ that converges to the actual posterior $\pi(\theta|y_{\text{obs}})$ as $\varepsilon \rightarrow 0$. ABC-SMC uses a lower-dimensional summary statistic version of the observed data and repeats the process for multiple generations, each containing N accepted simulations.

The classical scheduling strategies for ABC-SMC suffer from low utilization of computing resources. This has been addressed here by proposing a new scheduling approach, look-ahead scheduling (LA), which improves the resource utilization. The new strategy eliminates idle time by generating tasks for the next generation $t + 1$ using a preliminary or previous proposal once the current generation does not generate new tasks. The new particles will be sampled from a preliminary proposal $\tilde{g}_t$ that is based either on the set of finished N particles or on the previous proposal $\tilde{g}_t$ for generation t , to avoid practical bias towards fast simulation particles (see Figure 4.1).

A fundamental characteristic of the ABC-SMC method is that it generates an asymptotically unbiased Monte Carlo sample from $\pi_{\text{ABC},\varepsilon_{n_t}}(\theta | y_{\text{obs}})$, with $\pi_{\text{ABC},\varepsilon_{n_t}}(\theta | y_{\text{obs}}) \rightarrow \pi(\theta | y_{\text{obs}})$ as $\varepsilon \rightarrow 0$. The sample drawn for the new LA method preserves this property:

Each subpopulation independently provides an asymptotically unbiased estimator, since the weights

$$\tilde{w}_t(\tilde{\theta}) = \pi(\tilde{\theta})/\tilde{g}_t(\tilde{\theta}) \quad \text{and} \quad w_t(\theta) = \pi(\theta)/g_t(\theta) \quad (4.2)$$

precisely correspond to the Radon-Nikodym derivatives with respect to the respective proposal distributions.

Note that this theoretical unbiasedness is maintained regardless of whether $\tilde{g}_t$ is derived from $\hat{P}_{t-1}$ or P_{t-2} , provided that $\tilde{g}_t(\theta) \gg \pi(\theta)$. However, a practical bias may arise due to a finite sample size.

We implement the new look-ahead scheduler into the existing ecosystem used throughout this thesis. On the inference side, it is integrated with the pyABC algorithm along with the already existing scheduler schemes. On the modeling side, it interacts with the multi-cellular models through the FitMultiCell pipeline.

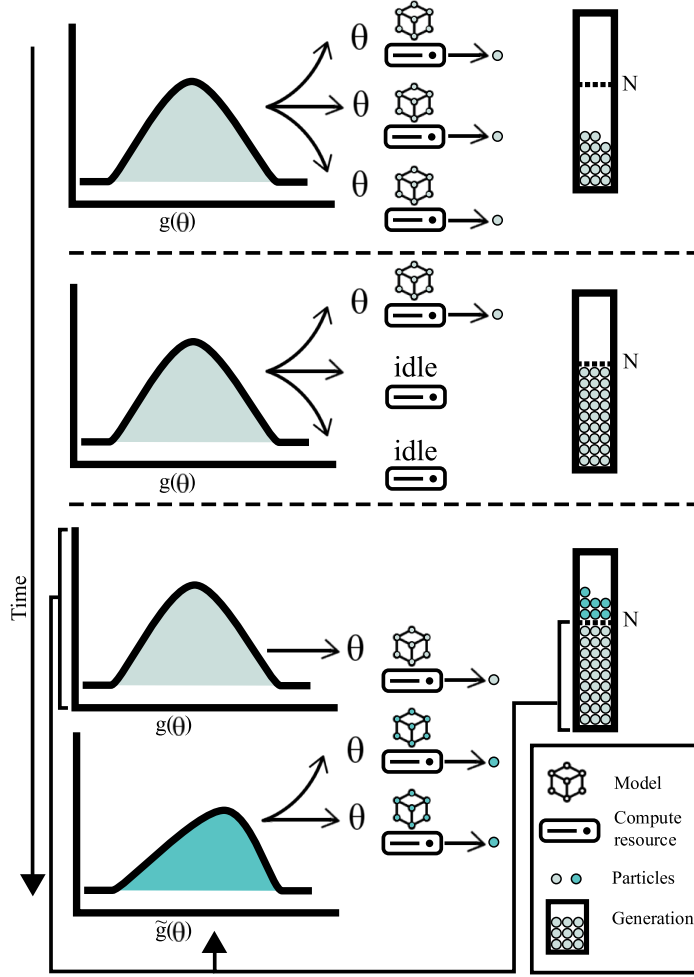


Figure 4.1: Visual representation of the look-ahead scheduling (LA). For each generation, one will sample from a prior $g_1(\theta)$ or proposal distribution $g_t(\theta)$ (upper section). As soon as there are N accepted particles, there will be some idle resources (middle section). Once that happens, new particles will be sampled from a preliminary distribution $\tilde{g}_t(\tilde{\theta})$ that is either based on the N accepted particles or the current generation proposal (last section).

4.3 Application to biological case study

The proposed look-ahead scheduling scheme was assessed on various problems: toy stochastic models (T1, T2) and real application biological test cases (M1, M2). The first model (T1) is a bimodal model $y \approx \theta^2$ that has two modes, one of which shows an artificially longer run time to replicate heterogeneity. The second toy model (T2) is an ODE model with 2 parameters describing a conversion reaction. The third model (M1) describes the growth of a tumor spheroid, modeling single cells as stochastically interacting agents and extracellular substances via PDEs. Finally, the (M2) model represents a liver regeneration process that describes the reaction network dynamics by a set of ODEs.

4.4 Significance within the Thesis

In the first paper of the cumulative thesis (see Section 7.1), we report the need to accelerate the ABC-SMC approach to build the pipeline by one that improves resource utilization and minimizes idle time. In this work (see Section 7.2), we address this challenge by introducing the look-ahead scheduling strategy that improves the speed-up while not affecting the overall inference accuracy. Integrating this look-ahead scheduler into the pipeline of the first paper improves efficiency and makes the parameter estimation process for multi-scale models using ABC-SMC more practical.

Across all application scenarios, the utilization of the computational resource remains high in cases where the model expresses heterogeneity, depending on the ratio between the population size and the number of cores. We observed a speed-up of up to 45% compared to the previously most efficient strategies. In typical application examples, we observed a speed-up of roughly 20-30%.

All benchmarks are fully documented with models, diagnostic diagrams, logs, and parameter estimations. The scheduler is released as an extension compatible with pyABC and can be directly called within the FitMultiCell pipeline. Producing preliminary particles requires attention to storage and memory, and the scheduler must account for any model-specific constraints (e.g., long initialization time).

4.5 Author's Contribution

I co-developed the scheduler within the pyABC tool and ensured seamless operation within FitMultiCell for multi-cellular models. Once developed, I proceeded by co-designing the benchmarking scenario and real-case application. I executed the scheduler on a large HPC cluster to experiment with and analyze resource utilization, acceptance behavior, and wall-clock reductions. I co-drafted the initial manuscript and contributed to its review and editing and prepared the figures and illustrations. Collaborators co-developed the scheduler and benchmark scenario, supervised the project, provided theoretical feedback, and edited the manuscript.

Inference of Cell Migration Dynamics in Confined Microenvironments

5.1 Motivation

Cell migration is important in many biological processes, including embryonic development, immune surveillance, tissue repair, and disease progression. It is controlled by the Complicated interaction of mechanical forces and biochemical signals in complex tissue environments. Although the molecular mechanisms underlying cell motility have been largely uncovered, it is still very difficult to understand how cells move through spatially structured microenvironments.

In this study, we used our previously built integrated modeling and inference framework to study the migrational behavior of immune cells, more specifically dendritic cells, in environments with geometric constraints such as dense extracellular matrix fibers. The aim is to analyze how both the chemotactic gradients and environmental constraints work together to influence the movements and directionality of cells. This was done by integrating a cellular Potts model with high-resolution experimental data.

To estimate the model's parameters from observed data, we used modern simulation-based inference methods, more precisely, the approximate Bayesian computation and the neural posterior estimate. They allow for model calibration without the need to calculate the likelihood, which is often infeasible for complex, stochastic systems such as models of multi-cellular cell migration.

Building on the pipeline of the first paper (see Section 7.1), and the improvement achieved using the look-ahead scheduler in the second paper (see Section 7.2), this study applied a data-driven inference of a cell migration model against time-lapse imaging data for dendritic cell movement across a maze-like lattice structure.

This work aims to provide new insights into how immune cells respond to chemical signal within physical obstacles by integrating biological data into our advanced computational framework. The results show not only the influence of spatial constraints in changing migration dynamics but also the potential of data-driven modeling pipelines to reveal mechanistic understanding from complex biological systems.

This summary is based on my work in the following publication, also reprinted in Appendix 7.3., and hence, parts of it may be similar to it:

-
- Arruda, J., **Alamoudi, E.**, Kiermaier, E., & Hasenauer, J. (2025). Simulation-Based Inference of Cell Migration Dynamics in Complex Spatial Environments. Currently under review.

5.2 Methodology

We study the migration behavior of bone marrow-derived dendritic cells within a microfabricated pillar forest after they are exposed to the chemokine signal (CCL19). We track the nuclear position using time-lapse video microscopy with a 30 s interval and a total time of 2 h. The microscopy readout is limited to a small view screen, which narrows the tracking window. From the raw tracks, we are able to obtain movement descriptors in 2D, such as displacement, velocity, turning angle, and angle degree (more details of these summary statistics will follow). These summary statistics are used to assess the quality of the mechanistic model of producing simulations similar to the experimental data. This reveal how dendritic cells balance the chemokine guidance with geometric constraints.

Hand-crafted summary statistics

When using the ABC-SMC approach, we compare the simulation to observable data by using hand-crafted summary statistics tailored to this model. Each simulation gives multiple cell trajectories of varying lengths, thus making a direct comparison impossible due to stochasticity. Hence, we computed the summaries for each cell and compared the simulation and experiment.

We used four widely applied descriptors of motility:

- **Displacement (D):** Time-normalized net displacement over the observed track.

$$D^c = \frac{1}{\sqrt{T^c}} \sqrt{(x_{T^c}^c - x_1^c)^2 + (y_{T^c}^c - y_1^c)^2}, \quad (5.1)$$

where T^c is the tracked step for each cell, x_i^c and y_i^c are the position of a cell c for $c \in \{1, \dots, N\}$ and at time step i for $i \in \{1, \dots, T^c\}$

- **Velocity (V):** Mean step speed, averaged over frames with $\Delta t = 30$ s.

$$V^c = \frac{1}{T^c - 2} \sum_{i=2}^{T^c} \frac{1}{\Delta t} \sqrt{(x_i^c - x_{i-1}^c)^2 + (y_i^c - y_{i-1}^c)^2}, \quad (5.2)$$

where $\Delta t = 30$ s is the interval of cell visibility.

- **Turning angle (TA):** Mean change in heading between consecutive steps.

$$TA^c = \frac{1}{T^c - 3} \sum_{i=3}^{T^c} (\theta_i^c - \theta_{i-1}^c), \quad (5.3)$$

where $\theta_i^c = (x_i^c - x_{i-1}^c, y_i^c - y_{i-1}^c)$.

-
- **Angle degree (AD):** Mean absolute heading of steps (orientation).

$$AD^c = \frac{1}{T^c - 2} \sum_{i=2}^{T^c} (x_i^c - x_{i-1}^c, y_i^c - y_{i-1}^c). \quad (5.4)$$

For the ABC distance, we computed the Wasserstein distance for each one of these summary statistics, then combined them by a weighted sum. The weights were set as the inverse of the difference between the largest and smallest statistics. Finally, the weighted distances are aggregated by summing them up.

5.2.1 Estimation framework

To analyze the data, we used the cellular Potts model to describe the experimental setup. The environmental obstacles were defined explicitly using an external image that captured the microstructure chip. A chemokine concentration, initiated as a static Gaussian, was placed at the far end of the domain, which allows cells to navigate the maze toward the source region. The model has four different parameters. These are 1) the chemokine attraction strength, which quantifies the sensitivity of cells to the chemokine gradient modulating directed movement; 2) the intrinsic random motion strength, which governs the bias toward previously randomly chosen movement directions; 3) the rate for the waiting time distribution between directional changes in intrinsic random motion; and 4) the target area a , which regulates cell size via area constraints. For more details on these parameters, see Section 7.3.

Finally, we fitted the model using four different inference strategies: (i) ABC with handcrafted summary statistics (displacement, velocity, turning angle, and angle degree); (ii) ABC employing a neural network as a feature extractor trained either to predict the posterior mean (ABC-PM) or (iii) with summaries tailored for inference (ABC-NPE); and (iv) neural posterior estimation (NPE) with a jointly trained summary network.

5.3 Application to biological case study

The dendritic cell migration was modeled using a 2D cellular Potts model with a chemokine gradient and two types of motions: random and directed movement. Within the CPM model, the cells are represented as extended domains of lattice sites. This allows for better modeling of cell deformation and shape. Cell behavior is the result of an energy minimization principle that integrates responses to external forces, such as chemokine gradients, with terms for volume constraints. This principle enables the generation of trajectories under conditions that resemble the experimental setup.

We randomly placed 50 cells in a starting area and allowed them to move toward the chemical cue, and the model output was the trajectories of these cells.

5.4 Significance within the thesis

In the first paper of this cumulative thesis (see Section 7.1), we built the FitMultiCell pipeline that enables the construction and calibration of multi-scale and multi-cellular biological processes. Then, we complement this in the second paper (see Section 7.2) by further enhancing the inference method

to make it more efficient and achieve an average speed-up between 20 and 30%. The work here puts these pieces together in a real application example. Using the FitMultiCell pipeline and the faster inference, we calibrated a cellular Potts model that captures the microstructured environment and chemokine field. The fitted model shows how spatial barriers (such as obstacles or other cells) redirect cell paths, while chemical cues provide directional persistence.

The model was calibrated on both synthetic and real data. For the synthetic data, we drew a parameter vector from the prior (for more information about the prior, refer to Section 7.3), then simulate a dataset using the vector, and employ the four strategies to recover the parameters. The accuracy of these methods was measured via the normalized root mean squared error (NRMSE), which shows that NPE gives the smallest error, closely followed by ABC and ABC-NPE. Then, the model was fitted on the 143 individual cell trajectories of varying length (18 to 120 frames per cell, with a mean duration of 33.2 min). Both ABC variants converged in 13 generations, as indicated by the small final acceptance thresholds.

Using the NPE strategy as the most flexible, the model was calibrated on the experimental data. The model was able to reproduce the main movement patterns seen in the microstructured pillar array. This enabled realistic simulations under varying environmental conditions, showing how chemical signals and structural barriers work together to control immune cell movement.

In addition, NPE was used with an ensemble of three independently trained networks (NPE-E) to provide a more robust estimate of individual network training variations. To test the fit, we compared the hand-crafted summary statistics on posterior simulations, the L1 distance in the NPE summary space, and cosine similarity in an independent UMAP projection. NPE and NPE-E were closer to the data than ABC, while ABC with hand-crafted summaries showed the largest deviation. As an extra robustness check, we repeated NPE on a segment of the data, and noticed that the posterior shapes and medians were stable. These results show that our cellular Potts model can capture essential cell behaviors. Using posterior inference with neural summaries, we successfully identified the key parameters driving migration.

However, this model has some limitations. Cell movement was only captured inside the pillar forest. The field of view limits what we can see and how cells move within the obstacles but not how many cells ultimately reach the chemokine source. A wider screening window would allow for better understanding of the overall migration dynamics and potential interactions with other cells or environmental factors. Another limitation is the temporal resolution of the observations. The 2 h track limit and interval size of 30 s might limit the amount of information available for calibration, which could affect the accuracy of the parameter estimation, especially the waiting time rate. Finally, there were multiple experimental and modeling assumptions throughout the paper. We treat chemokine as time independent, which ignores possible changes due to diffusion, degradation, or cell uptake, which could influence how cells adapt their migration over time. Moreover, we used a 2D lattice for the CPM model, which might not fully capture the complexities of 3D migration dynamics in vivo. This simplification makes the study more feasible, but may miss small details. Even though the main conclusions are not affected by these simplifications, further refinement might be interesting for future work.

5.5 Author's Contribution

I reviewed the experimental setup and translated it into a computational task and built the spatial environment. Then, I built the ABC problem, which included designing its hand-crafted summary statistics and specifying the parameters' priors and distance metrics for inference. After that, I set up the HPC configuration for large-scale runs. I developed analysis and visualization tools to assess model performance and inference quality. Finally, I helped write the original draft and was involved in the review and editing of the manuscript. Collaborators contributed to the conceptualization, formal analysis, methodological refinement, software implementation, experimental investigations, and provision of resources. They also provided supervision and theoretical guidance and worked on the writing and review of the manuscript.

Discussion

6.1 Discussion of Results

The overall goal of this thesis is to make the simulation and inference of the multi-cellular modeling more practical, faster, reusable, and easier to apply to real complex biological problems. The thesis achieved this through three advancements: i) a pipeline that handles the entire cycle of constructing, simulating, and inferring complex biological processes; ii) an algorithmic speed-up for ABC-SMC through the introduction of look-ahead scheduling that improves the computational resource utilization; and iii) a case study that fits a cell migration model to experimental cell trajectories in a maze-like environment and investigates the drivers behind cell movements. This chapter summarizes the collective results of the three papers, their limitations, and the future direction for further research on the topic.

The first paper tries to answer the research question of whether we can turn the simulation and parameter estimation of the multi-scale, multi-cellular from ad hoc environments, where every project requires highly specific custom code, non-reusable scripts, limited reproducibility, and no standard way of defining the inference problem, into a reproducible, scalable, and user-friendly framework that others can actually use. The answer to this question was the introduction of the FitMultiCell pipeline. The FitMultiCell pipeline accomplishes this by providing a full and customizable framework for simulating and parameter estimation of multi-scale, multi-cellular biological systems. The framework also makes an effort to standardize the problem definition of parameter estimation by extending an existing standardized format to accommodate multi-scale and multi-cellular models. We demonstrate that a complex biological model can be built without the need to write custom problem-specific code each time. We show this using multiple case studies that represent different biological systems. Also, its scalability has been demonstrated using several real application problems to prove its efficiency in handling large-scale simulations.

Then, we advance to answer how can we cut the estimation time of the ABC-SMC to make the estimation process more efficient? This was answered in the second paper by introducing a new scheduler that we call look-ahead. The new scheduler improves overall time by decreasing idle time for the computational resources through providing preliminary tasks that will most likely be needed in the next iteration of the ABC-SMC algorithm. This improvement was tested and proved to provide improved performance in most cases.

After answering the previous two questions, the path was paved for using the built tools for scientific

exploration. We ask how can we use the immune cell, more specifically the dendritic cell, trajectory in a microstructured environment to explain the mechanisms that underlie the cell migration behavior? Our investigation shows that the environment guides cell movement, yet persistence makes them travel farther, and the chemokines steer their direction.

All papers together in this cumulative thesis show a full addressing of the research questions proposed in section 1.1. The first paper introduces a reusable and standardized workflow, the second paper improves the efficiency of the workflow, and finally, the third paper applies both to a real use case to extract a biological insight about cell migration under environmental obstacles. We move from a one-time custom algorithm that is problem-specific to a well-established and tested pipeline that can enhance reusability and standardization at scale with minimum technical expertise.

6.2 Outlook and future directions

In recent years, there have been significant advances in the area of biological multi-scale modeling, yet there is a need for further investigation in multiple areas. One of these challenges is reproducibility and standardization [17]. Compared to some mechanistic models, there is an ongoing need for a sharable model definition format, which affects the reusability of these models. This thesis pushes forward MorpheusML as a model construction format and PEtab-MS as a problem definition of the estimation problem. Yet, the adaptation of additional common standards remains insufficient. A common effort on the domain is needed to push these forward for better reproducibility, transparency, and collaboration, and eventually lead to better compliance with FAIR data principles.

Another challenge is the validation of the parameter estimation results. Ensuring parameter identifiability is a major concern when talking about multi-scale modeling due to the complexity and nonlinearity of the systems involved. This requires robust statistical methods and sensitivity analysis to ensure the accuracy and reliability of the model predictions.

One of the major future directions in the field is the integration of amortized inference and machine learning methods that can accelerate the likelihood-free inference. New methods, such as neural posterior estimation [147, 148], neural likelihood estimation [126], and neural ratio estimation [127, 149], have shown that neural networks can identify the inverse mapping from observation to parameter space once trained on model output. Thus, one can overcome the main critique associated with the traditional ABC method.

These new methods, such as NPE, NLE, and NRE, might prove to be useful, yet ABC-SMC still has its advantages, such as its simplicity, straightforwardness, and minimal assumptions. Thus, additional development for it will still be needed. One direction is to improve its scalability and computational efficiency through algorithmic innovations (e.g., scheduling, surrogate models) and implementation strategies (e.g., GPU/parallel computing). The likelihood-based estimation framework is usually intractable for complex multi-scale models due to their high-dimensional parameter spaces and stochastic dynamics. Thus, the adaptation of the likelihood-free inference method became popular, but it came with its computational expenses. One direction for improvement is to develop a more efficient method to explore the parameter space. Another direction would be to replace the expensive and detailed model with a surrogate one that is much cheaper to simulate but still accurate enough to capture the essential system dynamics. Machine learning approaches such as Gaussian processes can approximate the system in part or in full. A third direction will be to explore new and more efficient parallel strategies for the scheduler.

One of the crucial hyper-parameters when using the ABC-SMC method is the choice of summary statistics that can reduce the data dimensionality while keeping the important information intact. The current main practice is to handcraft this summary information, which, if done incorrectly, can lead to biased or inaccurate results. A future direction, as demonstrated in paper III (see Section 7.3), is to invest in automatic summary statistics construction methods. Then, one uses this to extract features from complex data formats, such as microscopy images or cell trajectories. By improving the construction of the summary statistics, one could potentially capture more data and, thus, improve the inference accuracy.

Looking at the established FitMultiCell pipeline, many of these challenges can be directly addressed as an extension of the pipeline due to its modular design. The inference component of the pipeline could be replaced with a small modification to enable more recent simulation-based inference methods. Other extensions, such as the ability to build a low-fidelity surrogate model that can guide the parameter estimation effort, could be constructed and integrated directly without the need for a complete redesign of the existing framework.

Finally, from the application side, modeling cell migration toward a chemokine gradient could be extended in the future in many ways. Future models could introduce multiple interacting cell types to show their coordination and competition, e.g., different immune cell types. Another direction is to explore the dynamic environment in which the surroundings can evolve and change over time, such as altering the location of the chemokine concentration. This will show how the cell migration strategy can be adjusted to adapt to environmental changes in real time. Such a model will be able to answer questions like, can a cell navigate using memory of past chemical concentrations? Those studies can then be compared to our findings to better understand the mechanisms during a stable vs. time-varying gradient. Finally, adding a mechanochemical feedback so that cells not only sense but also reshape their environment (e.g., making cells consume the extracellular component and be able to secrete their own chemical signal).

To sum up, the pipeline for simulating and inferring the multi-scale and multi-cellular, the ABC-SMC algorithmic speed-up of the look-ahead scheduler, and the cell migration application with the influence of the chemokine gradient come together to advance our understanding of multi-scale models of a complex structures. In short, this work turns multi-scale and multi-cellular modeling into a shareable, scalable, and testable workflow that provides quantitative biological insight and a clear path for continued improvement.

Appendices

7.1 FitMultiCell: simulating and parameterizing computational models of multi-scale and multi-cellular processes

This publication is reprinted as part of this thesis according to Elsevier guidelines on sharing published journal articles. Material from:

E. Alamoudi et al., *FitMultiCell: simulating and parameterizing computational models of multi-scale and multi-cellular processes*, *Bioinformatics* **39.11** (2023) btad674, ISSN: 1367-4811, eprint: <https://academic.oup.com/bioinformatics/article-pdf/39/11/btad674/53689329/btad674.pdf>, URL: <https://doi.org/10.1093/bioinformatics/btad674>

Systems biology

FitMultiCell: simulating and parameterizing computational models of multi-scale and multi-cellular processes

Emad Alamoudi ¹, Yannik Schälte^{1,2,3}, Robert Müller⁴, Jörn Starruß⁴, Nils Bundgaard⁵, Frederik Graw ^{5,6,7}, Lutz Brusch⁴, Jan Hasenauer ^{1,2,3,*}

¹Life and Medical Sciences Institute, University of Bonn, Bonn 53113, Germany

²Institute of Computational Biology, Helmholtz Zentrum München—German Research Center for Environmental Health, Neuherberg 85764, Germany

³Center for Mathematics, Chair of Mathematical Modeling of Biological Systems, Technische Universität München, Garching 85748, Germany

⁴Center of Information Services and High Performance Computing (ZIH), Technische Universität Dresden, Dresden 01062, Germany

⁵BioQuant—Center for Quantitative Biology, Heidelberg University, Heidelberg 69120, Germany

⁶Interdisciplinary Center for Scientific Computing, Heidelberg University, Heidelberg 69120, Germany

⁷Department of Medicine 5, Friedrich-Alexander-University Erlangen-Nürnberg, Erlangen 91054, Germany

*Corresponding author. Faculty of Mathematics and Natural Sciences, University of Bonn, Endenicher Allee 62, 53115 Bonn, Germany. E-mail: jan.hasenauer@uni-bonn.de (J.H.)

Associate Editor: Pier Luigi Martelli

Abstract

Motivation: Biological tissues are dynamic and highly organized. Multi-scale models are helpful tools to analyse and understand the processes determining tissue dynamics. These models usually depend on parameters that need to be inferred from experimental data to achieve a quantitative understanding, to predict the response to perturbations, and to evaluate competing hypotheses. However, even advanced inference approaches such as approximate Bayesian computation (ABC) are difficult to apply due to the computational complexity of the simulation of multi-scale models. Thus, there is a need for a scalable pipeline for modeling, simulating, and parameterizing multi-scale models of multi-cellular processes.

Results: Here, we present FitMultiCell, a computationally efficient and user-friendly open-source pipeline that can handle the full workflow of modeling, simulating, and parameterizing for multi-scale models of multi-cellular processes. The pipeline is modular and integrates the modeling and simulation tool Morpheus and the statistical inference tool pyABC. The easy integration of high-performance infrastructure allows to scale to computationally expensive problems. The introduction of a novel standard for the formulation of parameter inference problems for multi-scale models additionally ensures reproducibility and reusability. By applying the pipeline to multiple biological problems, we demonstrate its broad applicability, which will benefit in particular image-based systems biology.

Availability and implementation: FitMultiCell is available open-source at <https://gitlab.com/fitmulticell/fit>.

1 Introduction

Biological tissues are complex entities composed of cells and extracellular components. Tissues occur in different developmental stages and compositions, are highly dynamic and often heavily structured. Specific tissue properties are relevant for a broad range of processes, including tissue homeostasis, viral infection, and tumor development and treatment. For the experimental analysis of biological tissues, imaging techniques are widely used. Common approaches include light and fluorescence microscopy, but more recently also imaging mass cytometry, spatial transcriptomics and related methods are employed [see Lewis *et al.* (2021) for a review]. These experimental techniques provide a variety of quantitative information about biological tissues. Yet, mechanisms underlying specific pattern formation or tissue dynamics often remain elusive. To address these aspects, computational modeling has established itself as a key element to obtain a comprehensive understanding of

causal relationships in multi-cellular spatio-temporal systems. Computational models of multi-cellular processes usually capture multiple spatial and temporal scales and describe the emergence of the system's behavior based on individual building blocks, e.g. individual cells and their interactions. There are several modeling approaches, including discrete, continuous, and hybrid model formalisms (Anderson and Quaranta 2008, Waclaw *et al.* 2015, Meyer *et al.* 2020). Cells and their interactions can, for instance, be described using (energy-based) Cellular Potts Models (CPM) or (force-based) vertex models. Several software packages have been developed, including MCell (Kerr *et al.* 2008), FLAME (Richmond *et al.* 2010), CompuCell3D (Swat *et al.* 2012), Chaste (Mirams *et al.* 2013), Morpheus (Starruß *et al.* 2014), and PhysiCell (Ghaffarizadeh *et al.* 2018) to simplify and standardize the demanding task of model formulation and implementation. Yet, while computational modeling has substantially improved the understanding of multi-cellular systems, it remains a challenging task (Fletcher and Osborne 2022).

Received: 21 February 2023; Revised: 25 October 2023; Editorial Decision: 27 October 2023; Accepted: 7 November 2023

© The Author(s) 2023. Published by Oxford University Press.

This is an Open Access article distributed under the terms of the Creative Commons Attribution License (<https://creativecommons.org/licenses/by/4.0/>), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.

For multi-scale models of multi-cellular processes, a key challenge is parameter estimation, the process in which values for the unknown model parameters are determined by fitting the model simulations to experimentally observed data. Parameter estimation is necessary to obtain quantitative models of processes, to analyse processes, to compare competing hypotheses about processes, and to predict the dynamics of processes (e.g. in response to perturbations) (Toni et al. 2011, MacLean et al. 2014, Jagiella et al. 2017, Imle et al. 2019, Durso-Cain et al. 2021, Joslyn et al. 2021). Systematic, rigorous and uncertainty-aware parameter estimation is only just becoming accessible for multi-scale models with advanced methods and growing computational resources. Reasons for this are that (i) the simulation of multi-scale models accounts for different biophysical processes (e.g. intra- and extracellular signaling as well as physical interaction), which requires computationally efficient simulation algorithms (e.g. hybrid discrete-continuum approaches), and that (ii) the stochasticity of most models (e.g. due to randomness arising from small cell numbers) necessitates repeated simulations.

To cope with the challenges of parameter estimation for advanced computational models, approximate Bayesian computation (ABC) methods have been developed and applied (Pritchard et al. 1999, Beaumont et al. 2002, Sisson et al. 2018). These methods generate samples from an approximation of the Bayesian parameter posterior distribution, without evaluating a likelihood function which can quickly become inaccessible for complex stochastic models. To enable the application of ABC methods to multi-scale models, they have been parallelized on high-performance computing (HPC) infrastructure (Johnston et al. 2014, Sottoriva et al. 2015, Babbie and Stumpf 2017, Jagiella et al. 2017). Generic implementations are provided by, in particular, pyABC (Klinger et al. 2018), ABCpy (Dutta et al. 2017), and ELFI (Kangasrääsiö et al. 2016). Yet, while simulation and inference tools are available, the parameter estimation for multi-scale models of multi-cellular processes still requires a high level of technical expertise. The available tools are not interfaced and code from published application examples is often difficult to reuse and extend. Thus, there is a need for a platform that facilitates and streamlines the entire workflow, from the construction of multi-scale models of various types based on biological principles, to systematic uncertainty-aware data-driven parameter estimation (Hasenauer et al. 2015). A first software tool tackling this problem is the spatial model editor (SME) (<https://github.com/spatial-model-editor>) (Keegan et al. 2023). However, this only supports partial differential equation models of reaction-diffusion systems, while stochastic and multi-scale models cannot be considered and parameter uncertainty analysis is not supported.

In this work, we introduce the FitMultiCell pipeline, an open-source, user-friendly, and scalable end-to-end platform that integrates modeling, simulation, and parameter estimation, to simplify the analysis of multi-scale and multi-cellular systems. FitMultiCell integrates the state-of-the-art tools Morpheus (Starruß et al. 2014) for model building and simulation, and pyABC (Schälte et al. 2022) for parameter estimation. It builds on an extension of the PEST standard (Schmiester et al. 2021) for the specification of parameter estimation problems that we additionally introduce in this manuscript. We demonstrate and evaluate the FitMultiCell pipeline using models of viral infection, tumor growth, and liver regeneration.

2 Methods

2.1 Problem description

We consider the problem of developing quantitative computational models of multi-cellular processes. These models might account for a broad range of biochemical and biophysical processes, including

- cellular signal transduction, metabolism, and gene regulation,
- cell movement, proliferation and death, and
- cell–cell communication (e.g. via direct cell–cell-contact areas or the secretion/uptake of biochemical substances).

The spectrum of modeling frameworks for such integrated processes is broad and the models are often stochastic, like the behaviors and decisions by individual biological cells are inherently stochastic. Mathematically, we can write any such model as

$$y = \mathcal{M}(\theta, \xi),$$

where y denotes the vector of observed properties and θ denotes the vector of unknown properties of biochemical, biophysical, or observation processes (e.g. reaction rates). Process noise—arising, for instance, from low molecular numbers—as well as measurement noise is described via the vector of random variables ξ . Marginalizing the model simulation over the random variables ξ yields the likelihood $\pi(y|\theta)$, which is the conditional probability of observing y given θ .

Quantitative mathematical modeling requires the inference of the unknown parameters θ from data. In this study, we consider Bayesian inference and aim to approximate the Bayesian posterior distribution

$$\pi(\theta|y_{\text{obs}}) \propto \pi(y_{\text{obs}}|\theta) \cdot \pi(\theta)$$

of the unknown model parameters for experimentally observed data y_{obs} and prior knowledge $\pi(\theta)$. The posterior distribution encodes all available information about the model parameter and, hence, allows for the assessment of parameter and prediction uncertainties as well as for the design of validation experiments. Experimental data used for the parameterization of the considered models are often obtained using imaging techniques. These can provide information about, e.g. spatial profile, temporal changes and changes between conditions. Overall, the spectrum of experimental setups is broad.

2.2 FitMultiCell pipeline

To facilitate quantitative computational modeling of multi-cellular processes, we developed the FitMultiCell pipeline (Fig. 1). This pipeline supports its users in

- 1) formulating the modeling problem, i.e.
 - a) defining computational models and
 - b) defining data obtained from experiments,
- 2) estimating unknown model parameters, and
- 3) evaluating parameter and model predictions (including planning validation experiments).

This pipeline streamlines the parameter estimation by providing a tight integration of tools for model specification and

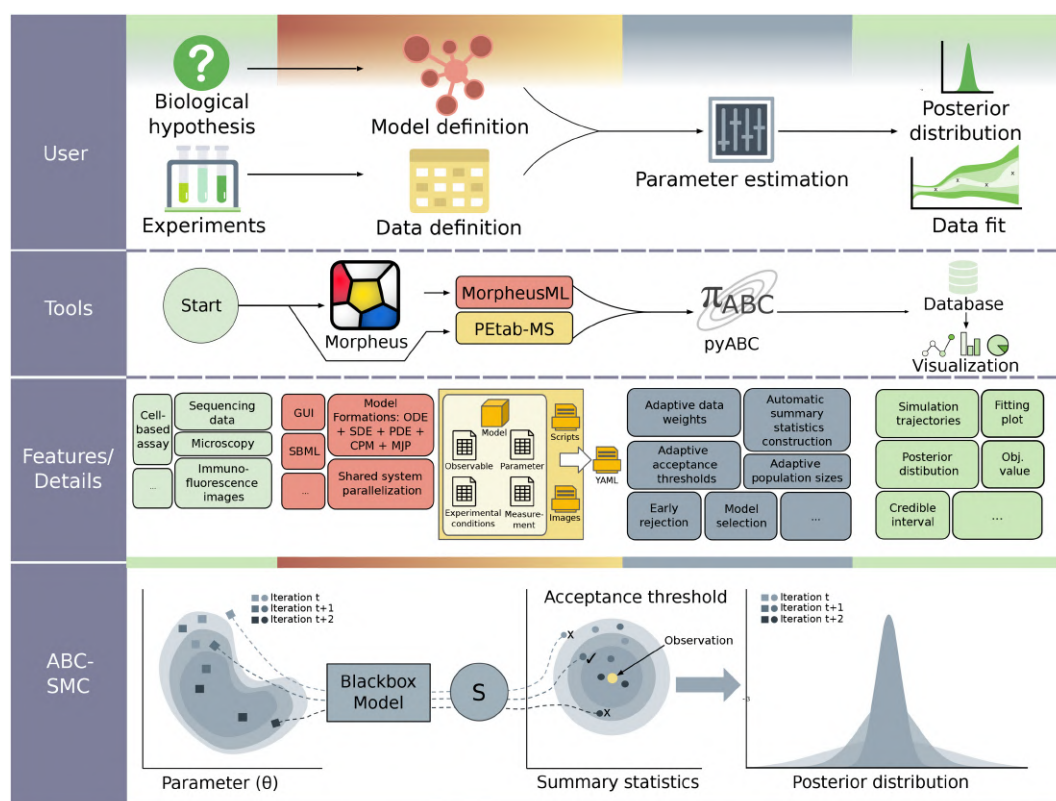


Figure 1. Overview of the FitMultiCell pipeline. First row: User perspective on the biological system, experiment, model formulation, and parameter estimation problem. Second row: Tool and format overview. Third row: Features and details for all components of the workflow. Fourth row: Visualization of the ABC-SMC parameter estimation algorithm.

simulation, and parameter estimation and reporting. This is achieved using a Python interface with a broad spectrum of functionalities. The current version supports (i) multi-scale modeling and simulation using Morpheus (Starruß *et al.* 2014) and (ii) parameter estimation and uncertainty quantification using pyABC (Klinger *et al.* 2018). These state-of-the-art tools cover modeling applications; yet, other modeling, simulation, and parameter estimation tools can be easily interfaced. Besides this Python interface, FitMultiCell offers a GUI to control pipeline execution and to analyse results. It also provides a collection of scripts and encapsulation of Morpheus that streamline pipeline execution on large computer clusters and cloud resources, making them accessible to non-expert users. Morpheus has been extended by GUI features for easily selecting parameters for estimation (auto-generated and interactive model graph to guide the user to relevant model parameters and automatic XPath creation to reference selected parameters) and by major performance improvements including adaptive parallel solvers for intracellular and partial differential equation (PDE) sub-models and the statistically exact multicore parallelization of the CPM formalism (see Supplementary data). Regarding pyABC, the construction of summary statistics has been simplified and robust adaptive distance functions were added. Lastly, PEPtab-MS was developed to ensure reproducibility and ease of problem definition, often eliminating the need for coding.

The FitMultiCell pipeline enables parameter estimation for application problems with multiple experimental conditions, datasets, and data types. Furthermore, it offers a broad range

of built-in visualization and analysis tools. To facilitate reproducibility and reusability, the FitMultiCell pipeline supports the MorpheusML and the PEPtab-MS standards.

In the following, we describe the key components of the FitMultiCell pipeline and their features.

2.2.1 Standardization

The FitMultiCell pipeline uses standardized data formats to ensure interoperability, reproducibility, and reusability—important aspects of the FAIR principles (Wilkinson *et al.* 2016). The multi-cellular models are encoded using MorpheusML, an established XML-based standard at <https://doi.org/10.25504/FAIRsharing.78b6a6>. The parameter estimation problems are encoded using the PEPtab-MS format, a newly developed tsv-file-based standard.

We developed PEPtab-MS as an extension of the Parameter Estimation tabular format (PEtab) (Schmiester *et al.* 2021), which was developed in the field of ordinary differential equation (ODE) based modeling and is already supported by various simulators and parameter estimation toolboxes. PEPtab-MS conserves most core components of PEPtab, including the tables defining model parameters and experimental conditions. Yet, PEPtab-MS allows for model expression using MorpheusML, flexible data tables (including references to image files), as well as functions for computing summary statistics. These aspects are essential for multi-cellular processes and were not captured by PEPtab. The specification of PEPtab-MS and a tool for the validation of files can be found at <https://gitlab.com/fitmulticell/libpetab-python-MS>.

As imaging data are often processed to obtain informative summary statistics, the FitMultiCell pipeline includes a class definition for the construction of summary statistics. Several common summary statistics for imaging data are already provided (see Section 3) and customized statistics can be implemented. Furthermore, it is possible to automatically construct informative summary statistics (Fearnhead and Prangle 2012, Schälte and Hasenauer 2022).

2.2.2 Simulation

The FitMultiCell pipeline is designed for the simulation-based analysis of multi-cellular processes. Accordingly, we allow for simulation of advanced models $\mathcal{M}$ for cells and tissues (using cellular automata and cellular Potts models), extracellular concentration fields (using partial differential equations) and intracellular dynamics (using ordinary or stochastic differential equations or continuous-time Markov jump processes).

The FitMultiCell pipeline implements a comprehensive interface to Morpheus, including parameter mapping and simulation result extraction. This renders the functionalities of Morpheus (e.g. model construction, simulation, and visualization) as well as features such as a comprehensive GUI for rapid prototyping available within the FitMultiCell pipeline. Morpheus is broadly applicable, widely used for a variety of biological problems, and computationally efficient [see, e.g. Köhn-Luque et al. (2011), Imle et al. (2019), Vu et al. (2019) and currently more than 80 models shared in the Morpheus model repository].

Additionally, the modular architecture of the FitMultiCell pipeline allows for the interfacing of additional modeling and simulation toolboxes, as well as user-provided simulation codes.

2.2.3 Inference

The FitMultiCell pipeline is designed for simulation-based inference, a class of approaches circumventing the evaluation of the likelihood function. This is important for the study of stochastic multi-cellular processes, but also allows for the application to deterministic models.

The FitMultiCell pipeline implements a comprehensive interface to pyABC (Schälte et al. 2022), including parameter and condition mapping for multi-experiment and multi-data type inference. This renders the state-of-the-art approximate Bayesian computation sequential Monte Carlo (ABC-SMC) method easily accessible to users of the FitMultiCell pipeline. ABC-SMC methods approximate the posterior distribution by constructing a sequence of particles which resembles the data (in terms of summary statistics) more and more closely (Toni and Stumpf 2010). Well-tested adaptation methods, e.g. for acceptance thresholds, proposal distributions, and population sizes, render pyABC accessible to non-expert users. Furthermore, pyABC allows for exact inference (Schälte and Hasenauer 2020) and we recently implemented automatic data normalization, summary statistics construction, and measurement noise handling to enable the robust estimation of parameters of multi-scale models (Schälte et al. 2021, Schälte and Hasenauer 2022). For the combined use of Morpheus and pyABC, we implemented an early rejection mechanism that discards simulations as early as possible, e.g. if they exceed an upper time limit. This makes the analysis robust to unexpectedly long-running simulations, e.g. due to an excessive number of cells.

We provided the interface to pyABC as this tool has been successfully used in a broad spectrum of applications, e.g. cardiac electrophysiology (Cantwell et al. 2019), human atrial cells (Houston et al. 2020), cancer (Colom et al. 2021), gene expression (Coulter et al. 2021), universe expansion (Bernardo and Said 2021), and bee colonies (Minucci et al. 2021). This underlines its applicability to a broad spectrum of inference tasks for multi-cellular processes. Additionally, the modular architecture of the FitMultiCell pipeline allows for the interfacing of additional inference tools, as well as user-provided inference algorithms.

2.2.4 Distributed execution

Parameter estimation often requires thousands to millions of stochastic model simulations, which is computationally demanding for complex multi-cellular processes. Distributed execution of the computational tasks is thus paramount (Fig. 2). Within the FitMultiCell pipeline, parallelization can happen on three levels to efficiently exploit HPC infrastructure:

- “Individual simulations” can be parallelized within the employed simulation toolbox. Morpheus supports the use of multiple threads using OpenMP. In addition to ODEs and PDEs, Morpheus also provides a parallel and exact solver for CPMs.

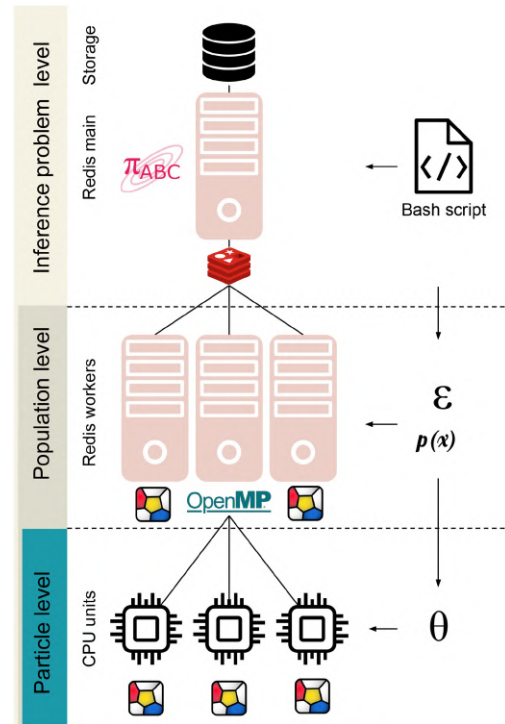


Figure 2. Illustration of the parallel framework of FitMultiCell. **Inference level:** A main process manages the pyABC analysis workflow and SQL data storage. **Population level:** In each ABC-SMC generation, the main process delegates to a number of distributed worker processes the task of sampling parameters (from proposal distributions $p(x)$) and synthetic datasets, until sufficiently many fulfil the generation-specific acceptance criterion (such as the acceptance thresholds ϵ). Work distribution, communication, and storage of intermediate results across various computational nodes and processing units are managed via a central Redis server. **Particle level:** Each single simulation of synthetic data (or groups of simulations for related perturbation scenarios in a multi-experiment setting) is performed using Morpheus, optionally with shared-memory parallelization via OpenMP.

- “Individual summary statistics evaluations” can be parallelized within the FitMultiCell pipeline if multiple individual simulations are required. This is for instance the case if experimental replicates are available for stochastic processes and multiple experimental conditions are considered. As the respective simulations are independent, they can be trivially parallelized across multiple threads.
- “Parameter estimation” can be parallelized by parameter estimation toolbox. PyABC supports single-machine multicore execution and multi-machine distributed execution. A main process manages the communication across SMC generations and post-processing of accepted particle populations, while the computation-heavy simulations and summary statistic evaluations are delegated to a set of parallel worker processes (see Fig. 2). pyABC provides two parallelization strategies, static and dynamic scheduling, which distribute work across computational resources, to minimize the overall CPU time, or the wall-time, respectively (Klinger *et al.* 2018).

To facilitate the setup of the required server-worker architecture, the FitMultiCell pipeline comes with bash scripts that handle the resource allocation and communication between nodes, and which can be used to run the pipeline on HPC clusters with minimum code modification. While parallelization is available at multiple levels, the most efficient resource use will often be achieved by dedicating many workers to the particle population and setting Morpheus to single-threaded simulation mode.

2.2.5 Analysis and visualization

The FitMultiCell pipeline provides a GUI for the analysis of the result of the parameter estimation process (see Fig. 3). This GUI can load the shareable SQL database generated by pyABC, but is flexible enough to visualize the results for other sampling tools after the result files were reformatted. Among other things, the GUI allows for the visualization of (i) samples using uni- and bi-variate plots, (ii) sampling diagnostics like the decrease of the acceptance threshold over generations, and (iii) the ability to generate the code for any plot that is generated. A detailed documentation for the GUI is available on pyABC (<https://pyabc.readthedocs.io/en/latest/visualization.html>) and FitMultiCell (<https://fitmulticell.readthedocs.io/en/latest/youtube.html>) documentation pages.

2.3 Implementation and development

The FitMultiCell pipeline is written in Python and available on GitLab (<https://gitlab.com/fitmulticell/fit>) and Zenodo (<https://doi.org/10.5281/zenodo.10040172>) under a BSD-3-Clause license. It is being developed by contributors from three institutions and others are invited to contribute. Code quality is monitored via unit tests and continuous integration. It can be installed from the Python Package Index (PyPI). Detailed documentation of the platform is available at <https://fitmulticell.readthedocs.io>, which covers installation, setting up the modeling and estimation problem, and running the parameter estimation process, including on HPC and cloud infrastructures.

To ensure ease of use, FitMultiCell pipeline integrates currently only tools which are easy to install and available under permissive licenses:

- Morpheus is available as a git repository (<https://gitlab.com/morpheus.lab/morpheus>) under the BSD-3-Clause license. It is written in C++ and available for all major operating systems as pre-compiled packages together with training materials and a model repository at <https://morpheus.gitlab.io/>.
- pyABC is hosted on GitHub (<https://github.com/icb-dcm/pyabc>) under the BSD-3-Clause license. It is written in Python and can be installed directly via (PyPI). Its many features are documented at <https://pyabc.readthedocs.io>.

3 Results

To assess the performance of the FitMultiCell pipeline, we evaluated it using a broad spectrum of test and application examples. In the following, we present its core properties for a few selected models of multi-cellular processes. Several other applications to various multi-cellular problems and different data types have been published separately and are not included here [see Boutillon *et al.* (2022), Bundgaard *et al.* (2022)].

All the following results were obtained on the JUWELS standard compute nodes of the Supercomputing Center in Jülich, Germany. Details on the technical specifications are provided in the Supplementary data.

3.1 Standards supported by FitMultiCell pipeline allow for re-implementation of published application problems

The FitMultiCell pipeline allows for the standardized description of models, datasets and parameter estimation problems. To ensure that the supported standards are practically useful, we assessed to which degree we can implement already published application problems.

We considered three applications capturing different biological and technical problems: (M1) a model of virus transmission via cell-free virions and cell-to-cell contact (Kumberger *et al.* 2018); (M2) a model of tumor spheroid growth (Jagiella *et al.* 2017); and (M3) a model describing the mechano-sensing of the metabolic status during liver regeneration (Meyer *et al.* 2020). The models M1 and M2 were originally not available in MorpheusML and were re-implemented for the purpose of this study, while M3 was already published in MorpheusML. For model M1, Morpheus allowed for a one-to-one re-implementation of the original setup up to the numerical simulation schemes. For model M2, some reformulations were necessary. Here, we aimed to (i) resemble the biological processes (and subprocesses) contained in the original models, (ii) preserve the meaning of the model parameters, and (iii) achieve similar dynamics for key process characteristics. The most substantial change was the replacement of the representation of cells and modeling based on an unstructured grid by a cellular Potts model. To confirm the validity of the re-implementation, we assessed a broad spectrum of model characteristics, including statistics over space and time (such as radial profiles and growth curves for model M2). We observed a good resemblance for model M1—as expected—and minor difference for model M2 (results not shown). For all applications, we created the PÉtab-MS description of the parameter estimation problems. PÉtab-MS was flexible enough to describe all parameter estimation problems. This suggests that the standards employed in the FitMultiCell pipeline are

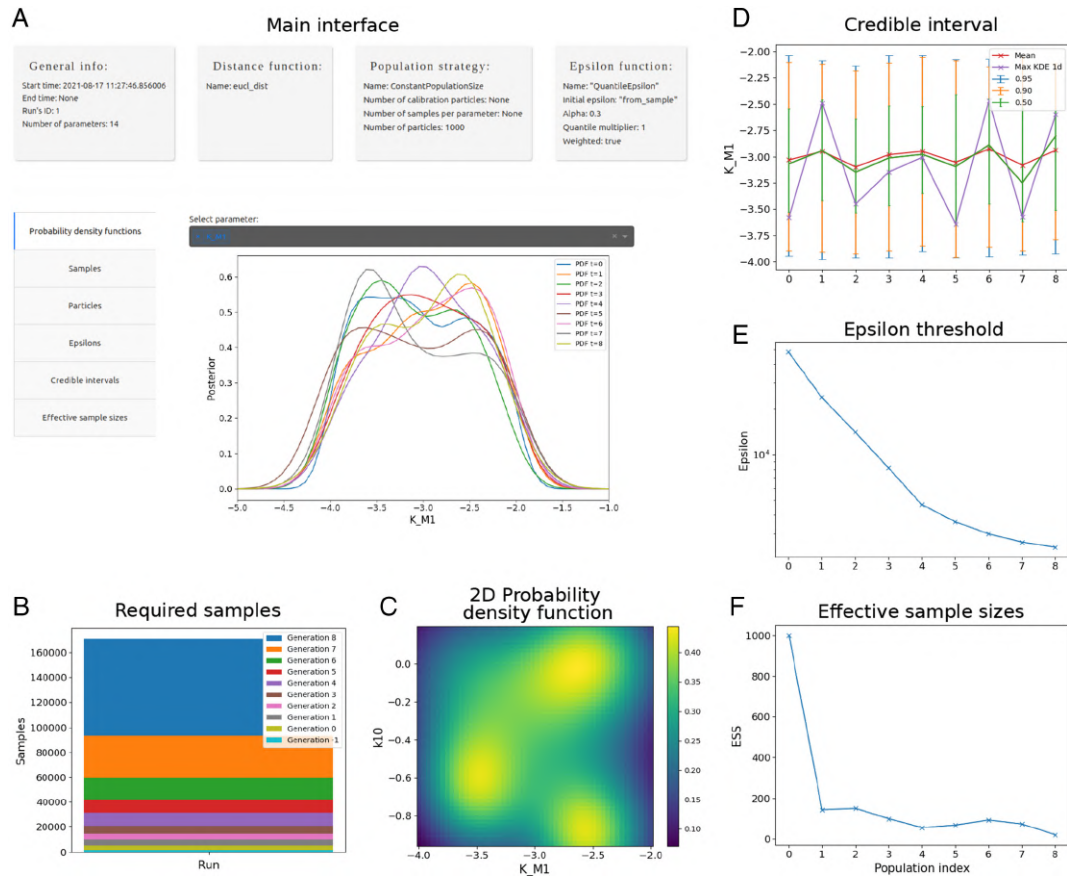


Figure 3. Several screenshots of the visualization and diagnostics GUI. (A) The main screen, (B) stacked bar plot of the number of required samples per generation, (C) 2D kernel density estimation for samples from approximate posterior distribution, (D) parallel coordinate plot for the development of the credibility intervals over generations, (E) line plot of the epsilon threshold per generation, and (F) line plot of the effective sample size per generation.

sufficient for a broad spectrum of applications. Details on the application problems are provided below and in the [Supplementary Section 3](#). Furthermore, the Morpheus and PTab-MS files provide a comprehensive standardized description.

3.2 Parallelization in FitMultiCell pipeline provides substantial wall-time reduction

As the efficient use of computing resources is of crucial importance for parameter estimation, we evaluated the parallel efficiency achieved using the FitMultiCell pipeline. We considered a model of the spread of a viral infection within a tissue distinguishing between virus transmission via cell-free virions and direct cell-to-cell contact (Kumberger *et al.* 2018) (Fig. 4A and [Supplementary Section 3.1](#)). In the FitMultiCell pipeline, parallelization is available within and across individual simulation particles. For the considered problems, we parallelize exclusively on the population level to provide the most efficient resource usage; however, this may depend on the specific problem.

To assess the parallel efficiency, we performed parameter estimation using the ABC-SMC algorithm implemented in pyABC with the dynamic scheduling option. The algorithms were run for three generations with two different population sizes: (Scenario A) 1000 particles; and (Scenario B) 10 000 particles. As higher population sizes improve the posterior approximation but are computationally more demanding, both

scenarios are relevant. We performed this task using 1, 2, 4, 8, and 16 compute nodes, each consisting of 48 CPU cores.

For both Scenarios A and B, we observed a substantial reduction in wall-time when using higher numbers of nodes, to a fraction of the single-node execution wall-time which already uses 48 CPU cores (Fig. 4B). Yet, the parallel efficiency (PE), defined as

$$PE = \frac{\{\text{wall-time for run on single compute node}\}}{\{\text{wall-time for run on } i \text{ compute nodes}\} \cdot i},$$

dropped (Fig. 4C). For the case of 16 nodes (768 cores), we observed a parallel efficiency of 49% and 75% in Scenario A and Scenario B, respectively. The higher parallel efficiency in Scenario B can be explained by the longer time required for individual generations, which reduces the effect of the idle time that usually arises at the end of each generation due to differences in computation times for individual particles [see also Klingner *et al.* (2018)].

In summary, our evaluation confirmed an overall good scaling of the FitMultiCell pipeline, yielding a wall-time reduction of several 10-fold compared to a single-node execution and several hundred-fold compared to single-core execution. As the number of proposed particles required to generate a specific number of accepted particles increases over generations, the parallel efficiency for production runs with the usual 15 to 40 generations should be higher than what has been observed here.

3.3 FitMultiCell facilitates the study of heterogeneous datasets via automatically tuned algorithms

To assess the benefit of the tight integration of modeling and parameter estimation tools in the FitMultiCell pipeline, and the availability of advanced inference algorithms, we considered the model of tumor spheroid growth (Jagiella *et al.* 2017) (Fig. 5 (left)). The original model was used to study growth control mechanisms and the effect of nutrient supply (Jagiella *et al.* 2016, 2017). Yet, the source code of this multi-cellular model entangled with its simulator is highly specific and no longer maintained, complicating the use of this model in further studies.

We implemented the model in MorpheusML and made it available through a public model repository (<https://identi>

[fiers.org/morpheus/M0007](https://morpheus/M0007)). The cell population was described using a stochastic CPM which accounts for cell division and death as well as cell–cell interactions, while the concentrations of extracellular substances were described using PDEs. The estimation problem was encoded using PetaMS, following the original publication by Jagiella *et al.* (2017). We selected the same seven unknown parameters that were also considered in Jagiella *et al.* (2017) based on a sensitivity analysis and biological relevance. This ensures comparability of our results against the reference publication. We considered all available datasets: a time-course for the tumor spheroid radius determined by bright field microscopy; and snapshots for radial profiles of markers for proliferating cells as well as extracellular matrix (ECM) abundance determined from fluorescence microscopy on day 17.

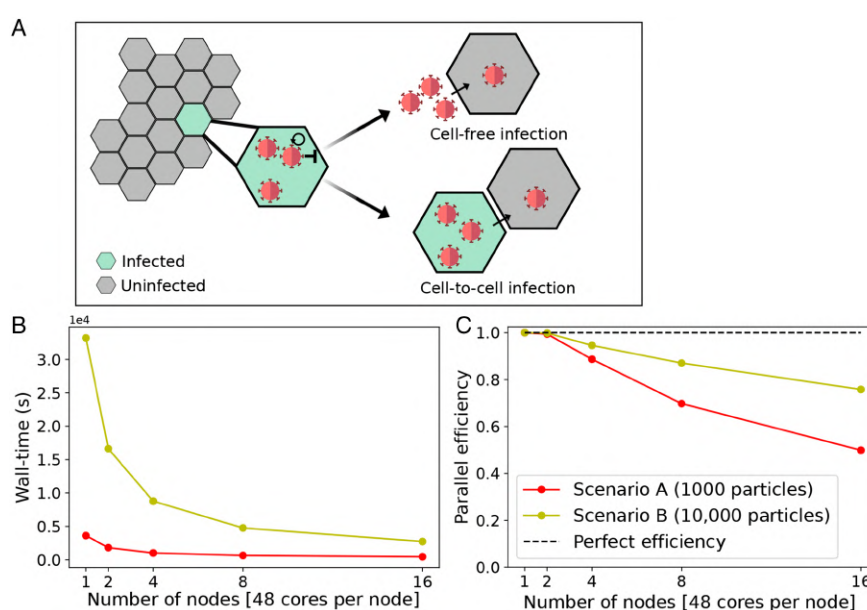


Figure 4. Performance of the parallelized FitMultiCell pipeline. (A) Illustration of the used model M1 for viral transmission within the host tissue. (B) Wall-time of two different fit scenarios across different numbers of used compute nodes, each consisting of 48 CPU cores. (C) Parallel efficiency of the FitMultiCell pipeline. The dashed line indicates perfect efficiency, i.e. a wall-time directly inversely proportional to the number of nodes, compared to single-node execution. For each choice of nodes, three consecutive ABC-SMC generations were run, with population sizes of $N = 1000$ (Scenario A) and $N = 10\,000$ (Scenario B).

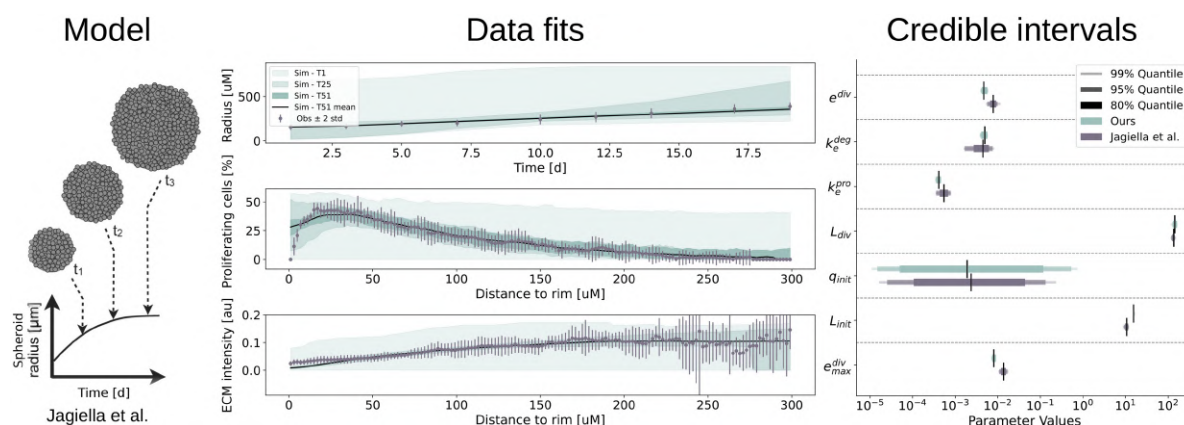


Figure 5. Result of FitMultiCell pipeline for the model of tumor spheroid growth. Left: Illustration of the model. Middle: Experimental data (gray) and model fit (green). The fit for ABC-SMC generation 1 (T1), generation 25 (T25) and final generation 51 (T51) is indicated using the 95% credibility interval of the accepted model simulations. Right: Credible intervals for the seven estimated parameters obtained using the original implementation (Jagiella *et al.* 2017) (gray) and the FitMultiCell pipeline (green).

In our previous publication considering the parameter estimation problem (Jagiella *et al.* 2017), we manually defined weights to quantify the relative importance of different summary statistics for the heterogeneous dataset. As this was a time-consuming process, here we employed a fully automatic approach for summary statistics weighting based on inverse regression (see [Supplementary Section 3.2.1](#) for details), which is available via the FitMultiCell pipeline. The specification of the parameter estimation task does only require a few lines of Python code given MorpheusML and PTab-MS files.

We ran the ABC-SMC algorithm with a population size of 500 particles and set as stopping condition a maximum wall-time of 48 h, within which the pipeline was able to finish 46 generations. The comparison of simulated and observed summary statistics showed that the model is able to fit the data accurately, with substantially improving fit quality in later ABC-SMC generations (Fig. 5 (middle)). Similar to the original study, the exceptions lie in the model's overestimation of the fraction of proliferating cells and underestimation of the ECM density at short distances from the rim. This corresponds to the findings of the original analysis (Jagiella *et al.* 2017). The analysis of credible intervals reveals that with the exception of the initial fraction of quiescent cells q_{init} , all parameters are identifiable with small uncertainties. The parameter estimates and uncertainty intervals obtained using our model are in excellent agreement with those in the original analysis (Fig. 5 (right)), even matching the distribution of the least certain parameter q_{init} . Hence, the FitMultiCell pipeline was able to reproduce the original results, while simplifying implementation and eliminating the need for manual tuning.

3.4 FitMultiCell pipeline facilitates parameter estimation for new applications

The parameters of multi-scale processes are often still adapted manually, e.g. due to the difficulty of setting up proper parameter estimation and its computational cost. As this involves manual work, it can lead to non-reproducible results and does not provide any information about parameter uncertainties. We assessed whether the FitMultiCell pipeline can easily replace manual parameter tuning. We studied a model describing the mechano-sensing of the metabolic status during liver regeneration after partial hepatectomy in mice, simulating the reaction network dynamics in hepatocytes along the central-portal axis of a liver lobule by a spatial array of coupled ODEs with spatially heterogeneous inputs (Meyer *et al.* 2020). For this model, the parameters were previously determined manually using data for two observables: the concentration of the YAP protein in the nucleus of hepatocytes (NYAP), and the total concentration of YAP protein in hepatocytes (TYAP), see Fig. 6 (left).

The model was already implemented in MorpheusML and is available from a public model repository (<https://identifiers.org/morpheus/M7990>). Yet, to avoid the simplifying assumptions that were used in the original study due to lack of appropriate parameter estimation tools and computational resources, we (i) dropped the quasi-steady-state assumption of the intracellular dynamics and performed a full spatio-temporal simulation, (ii) extended the biochemical model by an observer model of the microscopy setup that introduces additional scaling parameters between protein concentration and observed fluorescence intensity, and (iii) included a statistical model for the measurement noise. Complementary, we created the PTab-MS files encoding the estimation problem for the 14 unknown parameters. Previously, due to the

employed quasi-steady state assumption, only the ratios of forward and backward rate constants were identifiable.

We ran the ABC-SMC inference with a population size of 1000 particles and a wall-time limit of 24 h. To save computational resources, we employed an early rejection strategy to reject particles based on a maximum runtime of 15 min for individual simulations not matching the data. The simulated trajectories in later generations of the inference process fitted the data for TYAP and NYAP mostly well (Fig. 6 (middle)). An exception was the pericentral locations (CV-PV zones 1–5) at the last time point $t = 5.0$ days, where the simulations couldn't match the low concentrations present in the data, hinting at a transition to homeostasis and a time-dependence of some parameters over longer time scales of many days. The credible intervals indicate that all 14 parameters are identifiable (Fig. 6 (right)). Interestingly, the parameter describing the fluorescence intensity normalization (k_{14}) was estimated close to one, which justifies the previous assumption that both microscopy-based observables can be treated with the same normalization (Meyer *et al.* 2020). Moreover, the ratios of rate constants for reversible reactions were found to match very well with the previous point estimates using the quasi-steady state assumption in (Meyer *et al.* 2020). Extending previous results, we here obtained estimates for the four rate constants k_1 , k_5 , k_7 , k_8 and quantified the uncertainty of each estimate. Hence, our study underpinned previous conclusions (Meyer *et al.* 2020), but also demonstrated that the FitMultiCell pipeline allows directly for an uncertainty-aware study of multi-cellular processes.

4 Discussion

Quantitative data-based modeling of multi-cellular processes is challenging, because the models are often stochastic and computationally demanding. Furthermore, due to a lack of standards, the reusability of codes and pipelines is limited. Motivated by these issues, we developed the FitMultiCell pipeline, which covers the entire workflow of model development, simulation, and systematic parameter inference (on HPC or cloud infrastructures) based on standardized input formats. Thereby, it contributes to the accelerated testing of biological hypotheses.

We used the FitMultiCell pipeline to study various application problems. This demonstrated that the proposed pipeline is widely applicable and can recover and confirm previous results. Further, we showed that it can replace manual parameter tuning, thus accelerating and solidifying the quantitative modeling of multi-cellular processes. Its modular implementation scales to HPC infrastructures and thereby facilitates inference for computationally expensive problems.

While the FitMultiCell pipeline already provides the necessary features, there are multiple directions in which it can be developed further. (i) Statistical inference for computationally expensive models can become challenging even when using massive parallelization. To address this, the use of cheaper surrogate models could be investigated for guiding parameter search and reducing the number of model simulations (Prangle 2016, Prescott and Baker 2021). Furthermore, one could make use of the idle time at the end of generations, e.g. by pre-sampling the next generation and subsequently correcting for bias. (ii) Methods for the automatic construction of summary statistics need to be explored. While it is possible to construct summary statistics semi-automatically (Blum *et al.* 2013) for

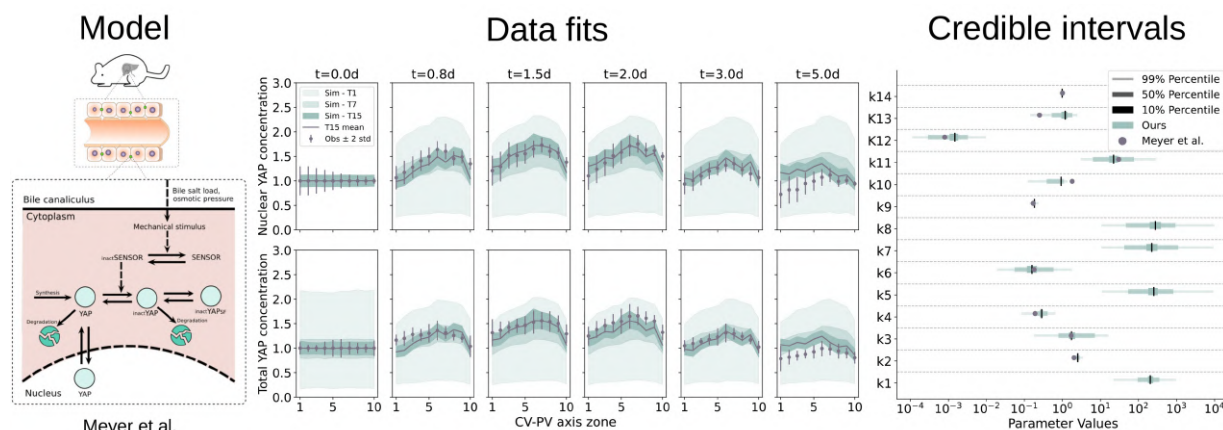


Figure 6. Result of FitMultiCell pipeline for the model of mechano-sensing of the metabolic status during liver regeneration. Left: Illustration of the model. Middle: Experimental data (gray) and simulation results (green) for different combinations of tissue location (from 1 to 10) and time point ($t = 0$ to $t = 5.0$ days). The fit for ABC-SMC generation 1 (T1), generation 7 (T7) and final generation 15 (T15) is indicated using the 95% credibility interval of the accepted model simulations. Right: Credible intervals for the 14 estimated parameters compared to the result of Meyer et al. (2020) (circle). Note that the four parameters k_1 , k_5 , k_7 , k_8 were not estimated in Meyer et al. (2020), therefore no comparison is available for those.

the type of image data frequently accompanying models of multi-cellular processes, automatic construction based on machine learning approaches, more specifically convolutional networks with prior information gained from transfer learning, could substantially improve statistics quality at decreased training cost. (iii) Extension of the FitMultiCell pipeline to additional modeling, simulation and parameter estimation frameworks will increase its applicability. While Morpheus and pyABC cover already a wide range of applications, an integration of further simulation tools [e.g. Swat et al. (2012), Mirams et al. (2013), Ghaffarizadeh et al. (2018), Merks et al. (2011)] and inference tools [e.g. Dutta et al. (2017), Kangasräsio et al. (2016)] could facilitate the use of complementary methods and thus grow the application spectrum further.

In conclusion, we illustrated that standardized workflows for the quantitative modeling of multi-cellular processes are feasible. The FitMultiCell pipeline and the standard PETA-MS provide starting points for further development. Already in its current form, a broad spectrum of projects can profit from them to achieve systematic and scalable inference of unknown parameters.

Supplementary data

Supplementary data are available at *Bioinformatics* online.

Conflict of interest

None declared.

Acknowledgements

We acknowledge fruitful discussions with colleagues at Morpheus.lab and the Gauss Centre for Supercomputing e. V. (www.gauss-centre.eu) for providing computing time on the GCS Supercomputer JUWELS [Jülich Supercomputing Centre 2019] at Jülich Supercomputing Centre (JSC).

Funding

This work was supported by the German Federal Ministry of Education and Research (BMBF) [FitMultiCell grant

nos. 031L0159B and 031L0159C, EMUNE grant nos. 031L0293C and 031L0293X]. J.H. acknowledges support by the German Research Foundation (DFG) under Germany's Excellence Strategy (Project-ID 390873048—EXC 2151 and Project-ID 390685813—EXC 2047) and via the Schlegel Professorship for J.H. F.G. was additionally supported by the Chica and Heinz Schaller Foundation. Y.S. acknowledges financial support by the German Research Foundation (DFG) (Project-ID 432325352 - SFB 1454 Metaflammation) and the Joachim Herz Stiftung. L.B. was supported by the BMBF [LiSyM-Cancer/031L0258A].

References

- Anderson ARA, Quaranta V. Integrative mathematical oncology. *Nat Rev Cancer* 2008;8:227–34.
- Babtie AC, Stumpf, MPH. How to deal with parameters for whole-cell modelling. *J R Soc Interface* 2017;14: 20170237. <https://doi.org/10.1098/rsif.2017.0237>.
- Beaumont MA, Zhang W, Balding DJ et al. Approximate Bayesian computation in population genetics. *Genetics* 2002;162:2025–35.
- Bernardo RC, Said JL. Towards a model-independent reconstruction approach for late-time Hubble data. *J Cosmol Astropart Phys* 2021; 2021:027.
- Blum MG, Nunes MA, Prangle D, Sisson SA et al. A comparative review of dimension reduction methods in approximate Bayesian computation. *Stat Sci* 2013;28:189–208.
- Boutillon A, Escot S, Elouin A et al. Guidance by followers ensures long-range coordination of cell migration through α -catenin mechanoperception. *Dev Cell* 2022;57:1529–44.e5. <https://doi.org/10.1016/j.devcel.2022.05.001>.
- Bundgaard N, Harmel C, Imle A et al. Inferring cell motility in complex environments with incomplete tracking data. *bioRxiv*, <https://doi.org/10.1101/2022.02.10.479738>, 2022, preprint: not peer reviewed.
- Cantwell CD, Mohamied Y, Tzortzis KN et al. Rethinking multiscale cardiac electrophysiology with machine learning and predictive modelling. *Comput Biol Med* 2019;104:339–51. <https://doi.org/10.1016/j.combiomed.2018.10.015>.
- Colom B, Herms A, Hall MWJ et al. Mutant clones in normal epithelium outcompete and eliminate emerging tumours. *Nature* 2021; 598:510–4.
- Coulier A, Hellander S, Hellander A. A multiscale compartment-based model of stochastic gene regulatory networks using hitting-time analysis. *J Chem Phys* 2021;154:184105.

- Durso-Cain K, Kumberger P, Schälte Y et al. HCV spread kinetics reveal varying contributions of transmission modes to infection dynamics. *Viruses* 2021;13:1308. <https://doi.org/10.3390/v13071308>.
- Dutta R, Schoengens M, Onnela J et al. Abcpy: A user-friendly, extensible, and parallel library for approximate Bayesian computation. In: *Proceedings of the Platform for Advanced Scientific Computing Conference, PASC '17*, New York, NY: ACM, 2017, 8:1–8:9. ISBN 978-1-4503-5062-4. <https://doi.org/10.1145/3093172.3093233>.
- Fearnhead P, Prangle D. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. *J R Stat Soc B* 2012;74:419–74.
- Fletcher AG, Osborne JM. Seven challenges in the multiscale modeling of multicellular tissues. *WIREs Mech Dis* 2022;14:e1527. <https://doi.org/10.1002/wsbm.1527>.
- Ghaffarizadeh A, Heiland R, Friedman SH et al. Physicell: an open source physics-based cell simulator for 3-d multicellular systems. *PLoS Comput Biol* 2018;14:e1005991. <https://doi.org/10.1371/journal.pcbi.1005991>.
- Hasenauer J, Jagiella N, Hross S et al. Data-driven modelling of biological multi-scale processes. *J Coupled Syst Multiscale Dyn* 2015;3:101–21.
- Houston C, Marchand B, Engelbert L et al. Reducing complexity and unidentifiability when modelling human atrial cells. *Philos Trans A Math Phys Eng Sci* 2020;378:20190339. <https://doi.org/10.1098/rsta.2019.0339>.
- Imle A, Kumberger P, Schnellbacher ND et al. Experimental and computational analyses reveal that environmental restrictions shape HIV-1 spread in 3D cultures. *Nat Commun* 2019;10:2144.
- Jagiella N, Müller B, Müller M et al. Inferring growth control mechanisms in growing multi-cellular spheroids of NSCLC cells from spatial-temporal image data. *PLoS Comput Biol* 2016;12:e1004412. <https://doi.org/10.1371/journal.pcbi.1004412>.
- Jagiella N, Rickert D, Theis FJ et al. Parallelization and high-performance computing enables automated statistical inference of multi-scale models. *Cell Syst* 2017;4:194–206.e9.
- Johnston ST, Simpson MJ, McElwain DLS et al. Interpreting scratch assays using pair density dynamics and approximate Bayesian computation. *Open Biol* 2014;4:140097. <https://doi.org/10.1098/rsob.140097>.
- Joslyn LR, Kirschner DE, Linderman JJ et al. Calipro: a calibration protocol that utilizes parameter density estimation to explore parameter space and calibrate complex biological models. *Cell Mol Bioeng* 2021;14:31–47.
- Kangasrääsiö A, Lintusaari J, Skytén et al. ELFI: Engine for Likelihood-Free Inference. In: *NIPS 2016 Workshop on Advances in Approximate Bayesian Inference*, Barcelona, Spain, 2016.
- Keegan L, Andriushchenko P, Schreiner H. Spatial-model-editor/spatial-model-editor: 1.4.0, 2023. <https://doi.org/10.5281/zenodo.8316368>.
- Kerr RA, Bartol TM, Kaminsky B et al. Fast Monte Carlo simulation methods for biological reaction-diffusion systems in solution and on surfaces. *SIAM J Sci Comput* 2008;30:3126–49.
- Klinger E, Rickert D, Hasenauer J et al. pyABC: distributed, likelihood-free inference. *Bioinformatics* 2018;34:3591–3.
- Köhn-Luque A, de Back W, Starruss J et al. Early embryonic vascular patterning by matrix-mediated paracrine signalling: a mathematical model study. *PLoS One* 2011;6:e24175. <https://doi.org/10.1371/journal.pone.0024175>.
- Kumberger P, Durso-Cain K, Uprichard SL et al. Accounting for space—quantification of cell-to-cell transmission kinetics using virus dynamics models. *Viruses* 2018;10:200. <https://doi.org/10.3390/v10040200>.
- Lewis SM, Asselin-Labat M-L, Nguyen Q et al. Spatial omics and multiplexed imaging to explore cancer biology. *Nat Methods* 2021;18:997–1012.
- MacLean AL, Filippi S, Stumpf MPH et al. The ecology in the hematopoietic stem cell niche determines the clinical outcome in chronic myeloid leukemia. *Proc Natl Acad Sci USA* 2014;111:3883–8. <https://doi.org/pnas.1317072111>.
- Merks RMH, Guravage M, Inzé D et al. Virtualleaf: an open-source framework for cell-based modeling of plant tissue growth and development. *Plant Physiol* 2011;155:656–66.
- Meyer K, Morales-Navarrete H, Seifert S et al. Bile canaliculi remodeling activates Yap via the actin cytoskeleton during liver regeneration. *Mol Syst Biol* 2020;16:e8985.
- Minucci JM, Curry R, DeGrandi-Hoffman G et al. Inferring pesticide toxicity to honey bees from a field-based feeding study using a colony model and Bayesian inference. *Ecol Appl* 2021;31:e02442.
- Mirams GR, Arthurs CJ, Bernabeu MO et al. Chaste: an open source C++ library for computational physiology and biology. *PLoS Comput Biol* 2013;9:e1002970. <https://doi.org/10.1371/journal.pcbi.1002970>.
- Prangle D. Lazy ABC. *Stat Comput* 2016;26:171–85.
- Prescott TP, Baker RE. Multifidelity approximate Bayesian computation with sequential Monte Carlo parameter sampling. *SIAM/ASA J Uncertain Quantif* 2021;9:788–817.
- Pritchard JK, Seielstad MT, Perez-Lezaun A et al. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Mol Biol Evol* 1999;16:1791–8.
- Richmond P, Walker D, Coakley S et al. High performance cellular level agent-based simulation with FLAME for the GPU. *Brief Bioinform* 2010;11:334–47. <https://doi.org/10.1093/bib/bbp073>.
- Schälte Y, Alamoudi E, Hasenauer J et al. Robust adaptive distance functions for approximate Bayesian inference on outlier-corrupted data. *bioRxiv*, <https://doi.org/10.1101/2021.07.29.454327>, 2021, preprint: not peer reviewed.
- Schälte Y, Hasenauer J. Efficient exact inference for dynamical systems with noisy measurements using sequential approximate Bayesian computation. *Bioinformatics* 2020;36:i551–9. <https://doi.org/10.1093/bioinformatics/btaa397>.
- Schälte Y, Hasenauer J. Informative and adaptive distances and summary statistics in sequential approximate Bayesian computation. *bioRxiv*, <https://doi.org/10.1101/2022.03.18.484896>, 2022, preprint: not peer reviewed.
- Schälte Y, Klinger E, Alamoudi E et al. pyabc: efficient and robust easy-to-use approximate Bayesian computation. *JOSS* 2022;7:4304. <https://doi.org/10.21105/joss.04304>.
- Schmiester L, Schälte Y, Bergmann FT et al. PETA—interoperable specification of parameter estimation problems in systems biology. *PLoS Comput Biol* 2021;17:e1008646. <https://doi.org/10.1371/journal.pcbi.1008646>.
- Sisson SA, Fan Y, Beaumont M et al. *Handbook of Approximate Bayesian Computation*. Chapman and Hall/CRC, Boca Raton FL, 2018.
- Sottoriva A, Kang H, Ma Z et al. A Big Bang model of human colorectal tumor growth. *Nat Genet* 2015;47:209–16. <https://doi.org/10.1038/ng.3214>.
- Starruß J, de Back W, Brusch L et al. Morpheus: a user-friendly modeling environment for multiscale and multicellular systems biology. *Bioinformatics* 2014;30:1331–2. <https://doi.org/10.1093/bioinformatics/btt772>.
- Swat MH, Thomas GL, Belmonte JM et al. Multi-scale modeling of tissues using CompuCell3D. *Methods Cell Biol* 2012;110:325–66. <https://doi.org/10.1016/B978-0-12-388403-9.00013-8>.
- Toni T, Jovanovic G, Huvet M et al. From qualitative data to quantitative models: analysis of the phage shock protein stress response in *Escherichia coli*. *BMC Syst Biol* 2011;5:69.
- Toni T, Stumpf MPH. Simulation-based model selection for dynamical systems in systems and population biology. *Bioinformatics* 2010;26:104–10.
- Vu HT-K, Mansour S, Kücken M et al. Dynamic polarization of the multiciliated planarian epidermis between body plan landmarks. *Dev Cell* 2019;51:526–42.e6. <https://doi.org/10.1016/j.devcel.2019.10.022>.
- Waclaw B, Bozic I, Pittman ME et al. A spatial model predicts that dispersal and cell turnover limit intratumour heterogeneity. *Nature* 2015;525:261–4. <https://doi.org/10.1038/nature14971>.
- Wilkinson MD, Dumontier M, Aalbersberg IJJ et al. The fair guiding principles for scientific data management and stewardship. *Sci Data* 2016;3:160018. <https://doi.org/10.1038/sdata.2016.18>.

Supplementary information:

FitMultiCell: Simulating and parameterizing computational models of multi-scale and multi-cellular processes

Emad Alamoudi¹, Yannik Schälte^{1,2,3}, Robert Muller⁴, Jörn Starrau⁴, Nils Bundgaard⁵,
Frederik Graw^{5,6,7}, Lutz Brusch⁴, and Jan Hasenauer^{1,2,3,*}

¹ Life and Medical Sciences Institute, University of Bonn, 53113 Bonn, Germany

² Helmholtz Zentrum München - German Research Center for Environmental Health, Institute of Computational Biology, 85764 Neuherberg, Germany.

³ Technische Universität München, Center for Mathematics, Chair of Mathematical Modeling of Biological Systems, 85748 Garching, Germany.

⁴ Center of Information Services and High Performance Computing (ZIH), Technische Universität Dresden, 01062 Dresden, Germany.

⁵ BioQuant - Center for Quantitative Biology, Heidelberg University, 69120 Heidelberg, Germany.

⁶ Interdisciplinary Center for Scientific Computing, Heidelberg University, 69120 Heidelberg, Germany.

⁷ Friedrich-Alexander-University Erlangen-Nürnberg, Department of Medicine 5, 91054 Erlangen, Germany.

*To whom correspondence should be addressed.

Contents

1	Algorithm implementation of ABC-SMC	2
2	Implementation and Tools	2
2.1	Technical specifications	2
2.2	FitMultiCell Software	3
2.3	pyABC Software	3
2.4	Morpheus Software	4
3	Specification of model and estimation problem	5
3.1	(M1) HCV model	5

3.2	(M2) Tumor Growth Model	6
3.2.1	Fully automatic approach for summary statistics weighting	7
3.3	(M3) Liver regeneration model	8
4	Scaling study	9

1 Algorithm implementation of ABC-SMC

ABC is a likelihood-free inference method particularly applicable to complex stochastic models, for which likelihood evaluation is computationally prohibitive. This is often the case for multi-scale models [1]. ABC only requires the ability to generate simulated data from the model given input parameters. In a nutshell, ABC calculates a distance between simulated and observed data and accepts corresponding parameters if the distance is below an acceptance threshold. ABC samples from an approximation to the Bayesian posterior distribution, thus providing uncertainty-aware parameter estimates [2]. ABC is frequently combined with an SMC approach, which allows for efficient gradual reduction of the acceptance threshold over a series of particle populations, and which is straightforward to parallelize [3, 4].

There exist various extensions to the core ABC routine, e.g. allowing to auto-tune hyperparameters [5], adapt distances [2] or population sizes [6] to the problem structure, select acceptance thresholds [7, 8], or learn low-dimensional summary statistics [9]. Many such approaches are implemented in pyABC (<https://github.com/icb-dcm/pyabc>), see [4, 10] for details. In particular, such semi-automatic self-tuned and robust approaches make the tool accessible also to users without expert knowledge.

The sampler implementation uses the *Redis* package (<https://redis.io>) as a broker between the main process and workers on a distributed high-performance computing (HPC) architecture.

2 Implementation and Tools

2.1 Technical specifications

For our test we used Anaconda 3, Python 3.8, with package versions pyABC 0.10.15, Morpheus 2.2.5, FitMultiCell 0.0.9. Most analyses were performed on the Juelich Supercomputing Center (JSC), Juwels cluster, standard compute nodes, specification of which are $2 \times$ Intel Xeon Platinum 8168 CPU, 2×24 cores, 2.7 GHz 96 (12×8) GB DDR4, 2666 MHz. Each node has 48 cores, equaling the number of workers per node used, as all models were single-threaded.

Table S1: Libraries and algorithms used in FitMultiCell

Name	Function	Web page
pyABC	Framework for distributed, likelihood-free inference	https://github.com/ICB-DCM/pyABC
Morpheus	Modelling and simulation environment for the study of multi-scale and multicellular systems	https://morpheus.gitlab.io
Redis	Message broker and in-memory database	https://github.com/antirez/redis

2.2 FitMultiCell Software

The FitMultiCell pipeline simplifies the process of estimating parameters for non-linear stochastic dynamics, especially when it’s unfeasible to calculate the likelihood function of experimental data based on a certain set of model parameters. This tool has proven effective in several multicellular stochastic processes, including scenarios such as tissue regeneration and viral propagation. FitMultiCell focuses on the development of an integrated computational platform for data-driven modelling and parameter inference of multicellular systems, with dedicated work packages for improving the scalability and acceleration of algorithms for simulation and parameter estimation. The codes for parameter inference used in FitMultiCell has been developed in-house (pyABC) and for model development and simulation, we use Morpheus (see table S1. The use of these libraries for our purposes is not limited by any licensing restrictions.

2.3 pyABC Software

pyABC is a statistical inference tool that employs an Approximate Bayesian Computing-Sequential Monte Carlo (ABC-SMC) framework [4]. pyABC supports single-machine multi-core execution and multi-machine distributed execution based on a variety of distributed execution engines, such as ad hoc clusters (e.g., the Dask distributed cluster and the IPython parallel cluster), bare grid-submission systems (e.g., SGE and UGE), and Redis based, low latency setups. One of pyABC’s distinguishing features is the offering of three parallelization strategies, Static and Dynamic, and look-ahead Scheduling, for the sampling of particles across individual populations. The Dynamic Scheduling approach stands out as an efficient parallelization strategy that can be scaled up to thousands of cores to minimize runtime. The look-ahead tries to optimize resources further by utilising the idle resources at the end of each population. The software is user-friendly, providing non-experts with well-tested default settings and automatic adaptations for elements such as acceptance threshold schedules, transition kernels, and population sizes. For more advanced users, pyABC allows customization and experimentation with ABC-SMC schemes without the need to modify the source code. Additionally, pyABC offers a web interface for visualizing ongoing and finished runs, and an API for querying and post-processing. Written in Python 3, the tool is available under a 3-clause BSD license, and its source code is accessible on GitHub (<https://github.com/icb-dcm/pyabc>) and can be installed from the Python Package Index (PyPI). A detailed documentation is avail-

able (<https://pyabc.readthedocs.io>). pyABC has proven its applicability across a broad spectrum of model types, such as ordinary differential equations, Markov jump processes, and agent-based multi-scale models [4]. It has been used successfully in several studies [11, 6?], including the analysis of a multi-scale model describing 10^6 single cells [11], and is being used by both national and international research groups in various projects. pyABC has repeatedly outperformed other tools (including manual tuning), successfully determining parameter values to provide accurate representations of experimental data where other methods have failed.

2.4 Morpheus Software

Morpheus is an open-source software framework that supports user-friendly declarative modeling and offers a graphical user interface (GUI) based on Qt (Fig. S1). Additionally, it provides the MorpheusML domain-specific modeling language, which allows for easy composition and extension of multicellular models. MorpheusML offers a bio-mathematical language where symbolic identifiers in mathematical expressions describe the dynamics and coupling of different model components. It employs the Cellular Potts Model (CPM) formalism to represent the spatial and mechanical aspects of cell interactions and adheres to the software design principle of separating the model from its implementation. This design allows for model sharing, versioning, and archiving. A numerical simulation is constructed by parsing the MorpheusML model definition and automatically scheduling predefined components in the simulator. Numerous models have been published in a model repository (<https://morpheus.gitlab.io/model>), enabling users to explore and adopt submodels. In addition to its integration with Morpheus, MorpheusML has become a standard for defining models, facilitating interoperability among multiple simulators, parameter estimation pipelines, and model repositories [12].

The GUI features an automatically generated and interactive model graph of the MorpheusML model definition (Fig. S1, lower right). Boxes represent model components, variables and processes. Each box can be clicked and then the corresponding model element, notably parameter, is highlighted in the model editor panel. That model editor has the ability to create the unique XML path string upon right-click directly from the GUI (Fig. S1-I). This XML path then serves as a unique identifier for that specific parameter within the FitMultiCell pipeline and PTab-MS file.

To speed up the FitMultiCell pipeline, several performance improvements have been implemented, including multicore parallelization of diffusion (for details see <https://morpheus.gitlab.io/post/2023/01/09/morpheus-2.3-release-notes/#reaction-diffusion-system-performance>) and the CPM model formalism (for details see <https://morpheus.gitlab.io/post/2023/01/09/morpheus-2.3-release-notes/#parallel-cpm>). CPM interactions are scaled based on lattice properties [13]. This simplifies the choice of lattice type and resolution to achieve optimal performance for a given model. Moreover, expressions (of position or of any dynamical model variable) are supported everywhere in the model to also account for heterogeneities in parameters (Fig. S1-II).

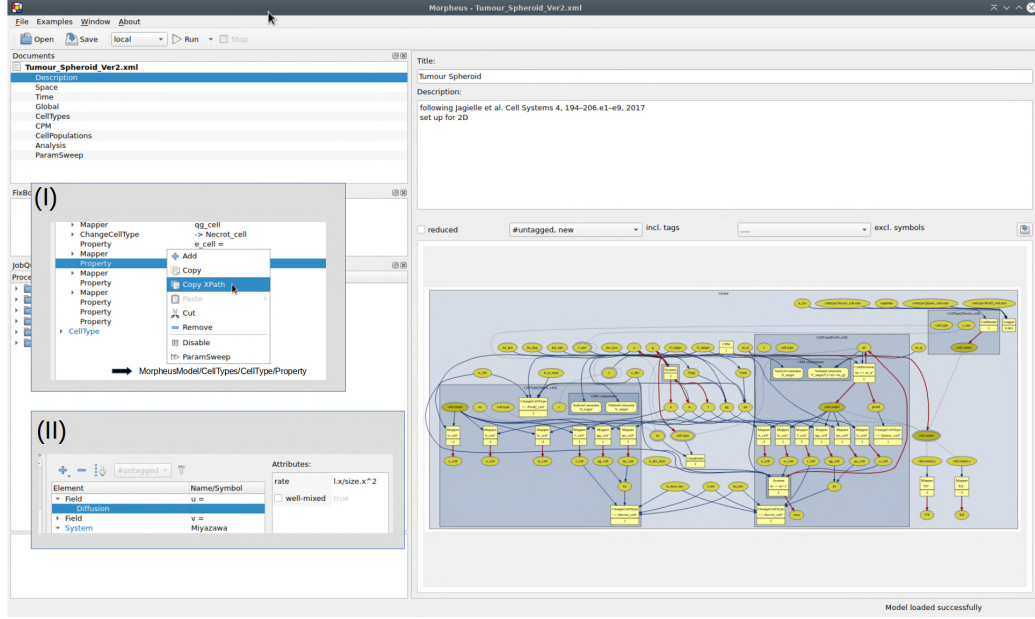
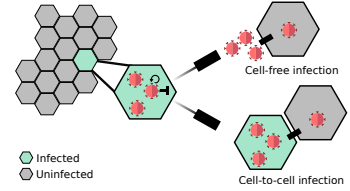


Figure S1: Graphical user interface (main panel) of the Morpheus software. (I): obtaining the XPath for a parameter, (II): spatial expression for diffusion constant.

3 Specification of model and estimation problem

3.1 (M1) HCV model

A model of the spread of the hepatitis C virus among cells was used to test the parallel efficiency of the FitMultiCell pipeline. The model was originally developed by [14] and aims to describe viral spread in a spatially-defined environment by accounting for cell-to-cell (CC) and cell-free infection (CF).



We have re-implemented the model in the FitMultiCell pipeline context. Morpheus was used to generate the MorpheusML model. The new model, similar to the original one, was built using a set of PDEs and ODEs that describes the infection process. A smaller version of the model was created, consisting of 721 cells arranged in a hexagonal grid. This smaller model served primarily for evaluating the performance and scalability of the pipeline. Only two parameters were fitted, as shown in table S2.

The model was fitted to synthetic data and the summary statistics for this model were constructed manually as 1) the number of clusters of infected cells, where a cluster is defined as a connected group of infected cells that initially start small and then increases toward the middle, before reducing again as the entire population becomes infected and forms fewer, larger clusters., 2) the number of cells infected by different modes of transmission (either CC or CF), and finally 3) the number of cells that can still contribute to CC infection. A cell remains CC infectious as long as it stays

Table S2: List of fitted parameters of model M1

Parameter	Description	Prior
$\log_{10}(\text{sf})$	Scaling factor of CF infection	$U(-3,-1)$
$\log_{10}(\text{sc})$	Scaling factor of CC infection	$U(-5,0)$

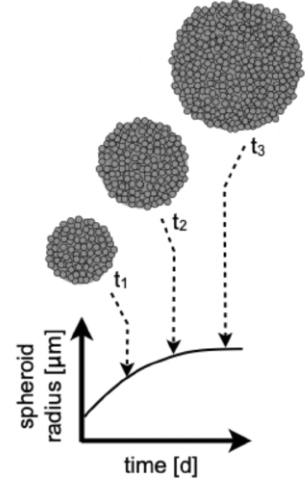
connected to a healthy cell. Given the generic nature and wide application of these three summary statistics, we have made them accessible on FitMultiCell. They can be directly applied during model construction.

A Euclidean distance was used as an objective function. The model was set to run until it reach to acceptance rate that is below 0.01. Each population was set to have a size of 200 accepted parcels.

To ensure the model’s reproducibility, a copy of the parameter estimation problem is provided in the Supplementary Material. This includes all its components, such as the distance function, synthetic data, summary statistics, and the observable function, all encapsulated within a PETab-MS.

3.2 (M2) Tumor Growth Model

The tumor growth model (M2) is our first application to experimental data. The model was developed by [15] and aims to describe the in vitro growth of a tumor spheroid while taking into account its spacial structure.



We have re-implemented the model in the FitMultiCell pipeline context. The model was created in Morpheus and encoded as MorpheusML model. In this implementation, we combine a PDE for the description of extracellular matrix density with a CPM for cell migration. In addition, cell division and cell death are implemented using the same mechanisms as in the model by [11]. For a description of the biological processes, we refer to [11]. For a detailed model specification we recommend to open the MorpheusML file in the Morpheus GUI. This will not only provide the model implementation but also information about the implementation of the individual components.

We note that the model is stochastic in nature. In particular while the cell numbers are small – which is the case for early time points – simulated trajectories show a high degree of variability. For large cell numbers a self-averaging effect is observed and the variability is rather low.

Table S3: List of fitted parameters of model M2. All parameters have been transformed to a logarithmic scale with base 10. The posterior mean and 95% credible intervals are based on the results of the last generation of the analysis.

Parameter	Description	Prior	Posterior mean	95% Credible intervals
$\log_{10}(k_{\max}^{\text{div}})$	Division rate	U(-3,-1)	-2.096	(-2.14, -2.04)
$\log_{10}(L_{\text{div}})$	Division depth	U(-5,0)	2.15	(2.10, 2.21)
$\log_{10}(L_{\text{init}})$	Initial spheroid radius	U(1,3)	1.18	(1.17, 1.21)
$\log_{10}(q_{\text{init}})$	Initial quiescent cell fraction	U(0,1.2)	-2.72	(-4.83, -0.27)
$\log_{10}(k_{\text{pro}}^e)$	ECM production rate	U(-5,0)	-3.38	(-3.45, -3.33)
$\log_{10}(k_{\text{deg}}^e)$	ECM degradation rate	U(-5,0)	-2.30	(-2.41, -2.23)
$\log_{10}(e_{\text{div}})$	ECM division threshold	U(-5,0)	-2.32	(-2.40, -2.22)

The model we implemented is a two-dimensional cross-section of the tumor spheroid. It possesses a large number of parameters, yet, following the original publication only seven parameters were estimated from the available datasets. These parameters are listed in Table S3. We note that the parameter implementation was conserved in the re-implementation process. The comparison of the parameter estimates which is presented in the main manuscript confirms the consistency.

Following the original manuscript [11], the parameters of the model are estimated from imaging data. Growth curves provide information about the time-dependent size of the spheroid. Histological imaging data were processed to obtain radial profiles, e.g. for the fraction of cells at a certain distance from the spheroid rime which is proliferating. Here, we used the post-processed data provided in [11]. For these data a distance function was formulated based on the weighted ℓ_2 distance, and the weighting was determined using an iterative procedure.

The precise specification of the parameter estimation problem, including the choice of distance function and weighting is provided using PETab-MS to ensure reproducibility.

3.2.1 Fully automatic approach for summary statistics weighting

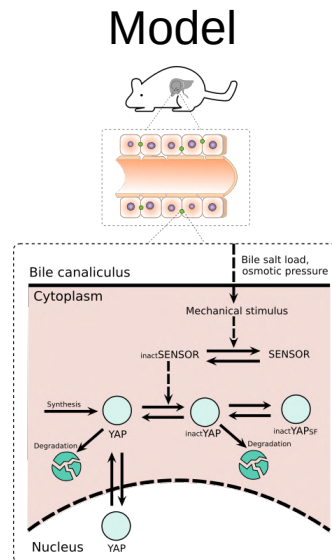
In Model (M2), we employ innovative sensitivity weights derived from regression models to measure the informativeness of data on parameters, as detailed in [16]. This approach is already incorporated in the parameter estimation tool pyABC [10]. The regression model can be pre-trained using completed particles, including both accepted and rejected ones, leading to a more efficient implementation by integrating the training phase into the actual ABC-SMC run. The regression model output then helps to assign additional weights to the data based on the informativeness of the underlying parameters. This informativeness is influenced by the sensitivity of the posterior expectation of parameters, or their transformations, given observed data, and how it would change under data perturbations.

Table S4: List of fitted parameters of model M3. All parameters have been transformed to a logarithmic scale with base 10. The posterior mean and 95% credible intervals are based on the results of the last generation of the analysis.

Parameter	Description	Prior	Posterior mean	95% Credible intervals
$\log_{10}(k_1)$	Max. flux of SENSOR activation	U(1,3)	2.30	(1.97, 2.54)
$\log_{10}(k_2)$	Ratio max. fluxes of SENSOR inact./act.	U(0,2)	0.40	(0.33, 0.45)
$\log_{10}(k_3)$	Ratio of YAP synthesis rate/YAP act.	U(-0.76,1.23)	0.23	(-0.10, 0.84)
$\log_{10}(k_4)$	Ratio of YAP inactivation rate/YAP act.	U(-1.95,0.04)	-0.54	(-0.71, -0.39)
$\log_{10}(k_5)$	YAP activation rate	U(1,4)	2.39	(1.72, 2.91)
$\log_{10}(k_6)$	Ratio of inact.YAP to SF binding/unbind.	U(-1.74,0.25)	-0.80	(-1.25, -0.23)
$\log_{10}(k_7)$	Inact. YAP unbind. rate from SF	U(1,4)	2.34	(1.62, 3.04)
$\log_{10}(k_8)$	YAP export rate from nucleus	U(1,4)	2.44	(1.66, 2.97)
$\log_{10}(k_9)$	Ratio of YAP nuclear import/export rate	U(-1.76,0.23)	-0.73	(-0.77, -0.71)
$\log_{10}(k_{10})$	Ratio of YAP degradation rate/YAP act.	U(-0.95,1.04)	-0.03	(-0.41, 0.041)
$\log_{10}(k_{11})$	Ratio of inact.YAP degrad. rate/YAP act.	U(0.47,2.47)	1.34	(0.79, 1.88)
$\log_{10}(k_{12})$	M-M const. of SENSOR activation	U(-4,-2)	-2.82	(-3.53, -2.48)
$\log_{10}(k_{13})$	M-M const. of SENSOR inactivation	U(-1.6,0.40)	0.07	(-0.27, 0.26)
$\log_{10}(k_{14})$	Intensity normalization total YAP	U(-1,1)	-0.005	(-0.027, 0.015)

3.3 (M3) Liver regeneration model

We consider a model of YAP regulation by mechanical stimulation through expansion of the bile canaliculi (BC) [17]. Single realisations of the model vary greatly based on parameter values and need runtimes ranging from 3 to 3,000 seconds. This model has 14 unknown parameters and two observables, namely nuclear YAP and total YAP intensities which were quantified from image tiles covering an entire liver lobule with portal and central veins. The details of the 14 parameters are listed in table S4. The yes-associated protein (YAP) is the downstream effector of the Hippo pathway that plays an important role in liver regeneration and developmental size regulation of many organs.



Two sub-models were used to describe the changes in osmotic pressure and the concomitant activation of YAP after partial hepatectomy. The first sub-model is a biophysics-based model to predict the local mechanical stress and apical membrane strain that result from the alteration of osmolyte (bile acid) load in the BC network after partial hepatectomy. It considers the spatial geometry of

the BC within the portal and central vein axis of the lobule. Sub-model 2 is a biochemistry-based model that predicts the cellular response of YAP to the local mechanical stress.

To account for measurement noise [18], additive normally distributed noise was added to the simulations, with a standard deviation of $\sigma = 0.0618$ and $\sigma = 0.0501$ for the two observables NYAP and TYAP, respectively, obtained via average standard errors of the mean from the observed data.

during the fitting process, an Euclidean metric was used as an objective function with a population size set to be 1000 particles. Moreover, the fitting was set to finish after 40 populations. However, most of the runs would be interrupted on around the 16th population due to a 24h maximum runtime constraint, which already yield robust results. The alpha-quantile of the epsilon threshold was set to 0.30.

Similar to models M1 and M2, all PETA-MS files for the parameter estimation problem of this model are available on the Supplementary Material.

4 Scaling study

The details of different scenarios used for the scaling study is presented in table S5. The table contains information regarding the number of populations, population sizes, number of nodes/threads, parallel efficiency, wall time in seconds, and speed-ups, total computation time, the total number of simulations performed, and finally, the acceptance rate.

Efficiency and speed-up are indicated relative to runtimes when using 48 cores as a baseline.

Table S5: Overview of scaling behaviour for different scenarios.

Scenario	Number of populations	Size of population	Nodes / cores	Parallel efficiency	Wall time (s)	Speed-up	Total computation time (s)	Total simulations	Ave. acceptance rate
A	3	1000	1 / 48	1	3,623.42	1	173924.16	6878	0.436
A	3	1000	2 / 96	0.99	1,822.43	1.98	174953.28	6735	0.445
A	3	1000	4 / 192	0.88	1,021.42	3.54	196112.64	7290	0.411
A	3	1000	8 / 384	0.69	649.23	5.58	249304.32	8016	0.374
A	3	1000	16 / 768	0.49	453.91	7.98	348602.88	8884	0.337
B	3	10,000	1 / 48	1	33,216	1	1594368	63704	0.470
B	3	10,000	2 / 96	0.99	16,637.43	1.99	1597193.28	64203	0.467
B	3	10,000	4 / 192	0.94	8,778.62	3.78	1685495.04	64199	0.467
B	3	10,000	8 / 384	0.87	4,769.68	6.96	1831557.12	64990	0.461
B	3	10,000	16 / 768	0.75	2,738.99	12.12	2103544.32	65826	0.455

References

- [1] J. Hasenauer, N. Jagiella, S. Hross, and F. J. Theis. Data-driven modelling of biological multi-scale processes. *J. Coupled Syst. Multiscale Dyn.*, 3(2):101–121, 9 2015.
- [2] Dennis Prangle. Adapting the ABC distance function. *Bayesian Analysis*, 12(1):289–309, 2017.
- [3] T. Toni and M. P. H. Stumpf. Simulation-based model selection for dynamical systems in systems and population biology. *Bioinf.*, 26(1):104–110, 10 2010.

- [4] Emmanuel Klinger, Dennis Rickert, and Jan Hasenauer. pyABC: distributed, likelihood-free inference. *Bioinf.*, 34(20):3591–3593, 10 2018.
- [5] S. Filippi, C. P. Barnes, J. Cornebise, and M. P. Stumpf. On optimality of kernels for approximate Bayesian computation using sequential Monte Carlo. *Stat. Appl. Genet. Mol.*, 12(1):87–107, 2013.
- [6] E. Klinger and J. Hasenauer. A scheme for adaptive selection of population sizes in Approximate Bayesian Computation - Sequential Monte Carlo. In J. Feret and H. Koepl, editors, *Computational Methods in Systems Biology. CMSB 2017*, volume 10545 of *Lecture Notes in Computer Science*. Springer, Cham, 2017.
- [7] Christopher C Drovandi and Anthony N Pettitt. Estimation of parameters for macroparasite population evolution using approximate Bayesian computation. *Biometrics*, 67(1):225–233, 2011.
- [8] D. Silk, S. Filippi, and M. P. H. Stumpf. Optimizing threshold-schedules for sequential approximate Bayesian computation: Applications to molecular systems. *Stat. Appl. Genet. Mol. Biol.*, 12(5):603–618, Oct. 2013.
- [9] Paul Fearnhead and Dennis Prangle. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. *J. R. Stat. Soc. B*, 74(3):419–474, 2012.
- [10] Yannik Schälte, Emmanuel Klinger, Emad Alamoudi, and Jan Hasenauer. pyabc: Efficient and robust easy-to-use approximate bayesian computation. *J. Open Source Softw.*, 7(74):4304, 2022.
- [11] N. Jagiella, D. Rickert, F. J. Theis, and J. Hasenauer. Parallelization and high-performance computing enables automated statistical inference of multi-scale models. *Cell Syst.*, 4(2):194–206, 02 2017.
- [12] Starruß J, et al. *MorpheusML*. FAIRsharing.org, 2023.
- [13] Ramiro Magno, Verônica A. Grieneisen, and Athanasius F. M. Marée. The biophysical nature of cells: potential cell behaviours revealed by analytical and computational studies of cell surface mechanics. *BMC Biophysics*, 8(1):8, May 2015.
- [14] Peter Kumberger, Karina Durso-Cain, Susan Uprichard, Harel Dahari, and Frederik Graw. Accounting for space—quantification of cell-to-cell transmission kinetics using virus dynamics models. *Viruses*, 10(4):200, Apr 2018.
- [15] N. Jagiella. *Parameterization of lattice-based tumor models from data*. Ph.d. thesis, Université Pierre et Marie Curie, Paris, France, 2012.
- [16] Yannik Schälte and Jan Hasenauer. Informative and adaptive distances and summary statistics in sequential approximate Bayesian computation. *bioRxiv*, 2022.
- [17] Kirstin Meyer, Hernan Morales-Navarrete, Sarah Seifert, Michaela Wilsch-Braeuninger, Uta Dahmen, Elly M Tanaka, Lutz Brusch, Yannis Kalaidzidis, and Marino Zerial. Bile canaliculi remodeling activates yap via the actin cytoskeleton during liver regeneration. *Mol. Syst. Biol.*, 16(2):e8985, 2020.
- [18] Yannik Schälte and Jan Hasenauer. Efficient exact inference for dynamical systems with noisy measurements using sequential approximate Bayesian computation. *Bioinf.*, 36(Supplement 1):i551–i559, 7 2020.

7.2 A wall-time minimizing parallelization strategy for approximate Bayesian computation

This publication is reprinted as part of this thesis according to Elsevier guidelines on sharing published journal articles. Material from:

E. Alamoudi et al., *A wall-time minimizing parallelization strategy for approximate Bayesian computation*, PLOS ONE **19.2** (2024) 1, URL: <https://doi.org/10.1371/journal.pone.0294015>

RESEARCH ARTICLE

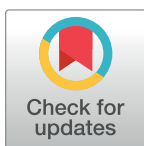
A wall-time minimizing parallelization strategy for approximate Bayesian computation

Emad Alamoudi¹, Felipe Reck¹, Nils Bundgaard², Frederik Graw^{2,3,4}, Lutz Brusch⁵, Jan Hasenauer^{1,6,7}*, Yannik Schälte^{1,6,7}

1 Life and Medical Sciences Institute, University of Bonn, Bonn, Germany, **2** BioQuant—Center for Quantitative Biology, Heidelberg University, Heidelberg, Germany, **3** Interdisciplinary Center for Scientific Computing, Heidelberg University, Heidelberg, Germany, **4** Department of Medicine 5, Friedrich-Alexander-University Erlangen-Nürnberg, Erlangen, Germany, **5** Center of Information Services and High Performance Computing (ZIH), Technische Universität Dresden, Dresden, Germany, **6** Helmholtz Zentrum München, Institute of Computational Biology, Neuherberg, Germany, **7** Center for Mathematics, Technische Universität München, Garching, Germany

* These authors contributed equally to this work.

* jan.hasenauer@uni-bonn.de



OPEN ACCESS

Citation: Alamoudi E, Reck F, Bundgaard N, Graw F, Brusch L, Hasenauer J, et al. (2024) A wall-time minimizing parallelization strategy for approximate Bayesian computation. PLoS ONE 19(2): e0294015. <https://doi.org/10.1371/journal.pone.0294015>

Editor: Abel C.H. Chen, Chunghwa Telecom Co. Ltd., TAIWAN

Received: July 3, 2023

Accepted: October 24, 2023

Published: February 22, 2024

Peer Review History: PLOS recognizes the benefits of transparency in the peer review process; therefore, we enable the publication of all of the content of peer review and author responses alongside final, published articles. The editorial history of this article is available here: <https://doi.org/10.1371/journal.pone.0294015>

Copyright: © 2024 Alamoudi et al. This is an open access article distributed under the terms of the [Creative Commons Attribution License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Data Availability Statement: The code underlying this study can be found at https://github.com/EmadAlamoudi/Lookahead_study. A snapshot of

Abstract

Approximate Bayesian Computation (ABC) is a widely applicable and popular approach to estimating unknown parameters of mechanistic models. As ABC analyses are computationally expensive, parallelization on high-performance infrastructure is often necessary. However, the existing parallelization strategies leave computing resources unused at times and thus do not optimally leverage them yet. We present look-ahead scheduling, a wall-time minimizing parallelization strategy for ABC Sequential Monte Carlo algorithms, which avoids idle times of computing units by preemptive sampling of subsequent generations. This allows to utilize all available resources. The strategy can be integrated with e.g. adaptive distance function and summary statistic selection schemes, which is essential in practice. Our key contribution is the theoretical assessment of the strategy of preemptive sampling and the proof of unbiasedness. Complementary, we provide an implementation and evaluate the strategy on different problems and numbers of parallel cores, showing speed-ups of typically 10-20% and up to 50% compared to the best established approach, with some variability. Thus, the proposed strategy allows to improve the cost and run-time efficiency of ABC methods on high-performance infrastructure.

Introduction

Mechanistic models are important tools in systems biology and many other research fields to describe and understand mechanisms underlying systemic behavior [1, 2]. Usually, such models have unknown parameters that need to be estimated by comparing model outputs to observed data [3]. For complex stochastic models, in particular multi-scale models used to describe the complex dynamics of multi-cellular systems, evaluating the likelihood of data given parameters however becomes quickly computationally infeasible [4, 5]. For this reason, simulation-based methods that circumvent likelihood evaluation have been developed, such as

code and data can be found at <https://doi.org/10.5281/zenodo.7875905>.

Funding: The authors acknowledge the Gauss Centre for Supercomputing e.V. (www.gauss-centre.eu) for funding this project by providing computing time on the GCS Supercomputer JUWELS at Jülich Supercomputing Centre (JSC). This work was supported by the German Federal Ministry of Education and Research (BMBF) (FitMultiCell/031L0159C and EMUNE/031L0293C) and the German Research Foundation (DFG) under Germany's Excellence Strategy (EXC 2047 390873048 and EXC 2151 390685813 and the Schlegel Professorship for JH). YS acknowledges support by the Joachim Herz Foundation. FG was supported by the Chica and Heinz Schaller Foundation. There was no additional external funding received for this study.

Competing interests: The authors have declared that no competing interests exist.

approximate Bayesian computation (ABC), popular for its simplicity and wide applicability [6, 7].

ABC generates samples from an approximation to the true Bayesian posterior distribution. While asymptotically exact, a known disadvantage of ABC is its computational complexity. The reason for this is that it requires often simulations of hundred thousands to millions of artificial datasets. Therefore, methods to efficiently explore the search space have been developed [8]. In particular, ABC is frequently combined with a Sequential Monte Carlo scheme (ABC-SMC), which over several generations successively refines the posterior approximation via importance sampling while maintaining high acceptance rates [9, 10]. Furthermore, in ABC-SMC the sampling for each generation can be parallelized, enabling the use of high-performance computing (HPC) infrastructure. This has in recent years enabled tackling increasingly complex problems via ABC [11–14].

It would be desirable if available computational resources were perfectly exploited at all times, to minimize both the wall-time until results become available to the researcher, and the cost associated with allocated resources. However, the problem is that established parallelization strategies to distribute ABC-SMC work over a set of workers leave resources idle at times and thus fall short of this aim. The parallelization strategy used in most established HPC-ready ABC implementations is *static scheduling* (STAT), which defines exactly as many parallel tasks as accepted particles are required [15, 16]. While it minimizes the active compute time and consumed energy, typically a substantial amount of workers become idle towards the end of each generation. *Dynamic scheduling* (DYN) mitigates this problem and reduces the overall wall-time by continuing sampling on all workers until sufficiently many particles have been accepted [17]. It was shown to reduce the wall-time substantially. However, also in this strategy at the end of each generation workers become idle, waiting until all simulations have finished.

A natural strategy to circumvent idle time is to start already sampling the next generation, given partial information about the current generation. Yet, it is not obvious how particles need to be accepted or weighted, and whether this would indeed improve efficiency. In this manuscript, we describe an ABC-SMC parallelization strategy for multi-core and distributed systems, called *look-ahead scheduling* (LA) which avoids idle time. We show that by appropriate sample reweighting we obtain an unbiased Monte Carlo sample. We provide an HPC-ready implementation and test the method on various problems. Moreover, we show that the strategy can be integrated with adaptive algorithms for e.g. summary statistics, distance functions, or acceptance thresholds.

Methods

ABC

We consider a mechanistic model described via a generative process of simulating data $y \sim \pi(y|\theta) \in \mathbb{R}^{n_y}$ for parameters $\theta \in \mathbb{R}^{n_\theta}$. Given observed data y_{obs} , in Bayesian inference the likelihood $\pi(y_{\text{obs}}|\theta)$ is combined with prior information $\pi(\theta)$ to the posterior distribution $\pi(\theta|y_{\text{obs}}) \propto \pi(y_{\text{obs}}|\theta) \cdot \pi(\theta)$. We assume that evaluating the likelihood is computationally infeasible, but that it is possible to simulate data $y \sim \pi(y|\theta)$ from the model. Then, classical ABC consists in the 3 steps of first sampling parameters $\theta \sim \pi(\theta)$, second simulating data $y \sim \pi(y|\theta)$, and third accepting θ if $d(y, y_{\text{obs}}) \leq \varepsilon$, for a distance metric $d: \mathbb{R}^{n_y} \times \mathbb{R}^{n_y} \rightarrow \mathbb{R}_{\geq 0}$ and acceptance threshold $\varepsilon > 0$. This is repeated until sufficiently many particles, say N , are accepted. The population

of accepted particles constitutes a sample from an approximation of the posterior distribution,

$$\pi_{\text{ABC},\varepsilon}(\theta|y_{\text{obs}}) \propto \int I[d(y, y_{\text{obs}}) \leq \varepsilon] \pi(y|\theta) dy \cdot \pi(\theta). \quad (1)$$

Under mild assumptions, $\pi_{\text{ABC},\varepsilon}(\theta|y_{\text{obs}})$ converges to the actual posterior $\pi(\theta|y_{\text{obs}})$ as $\varepsilon \rightarrow 0$ [18, 19]. Commonly, ABC operates not directly on the measured data, but summary statistics thereof, capturing relevant information in a low-dimensional representation [20]. Here, for notational simplicity we assume that y already incorporates summary statistics, if applicable.

ABC-SMC

The vanilla ABC formulation exhibits a trade-off between the reduction of the approximation error induced by ε , and high acceptance rates. Thus, ABC is frequently combined with a Sequential Monte Carlo scheme (ABC-SMC) [21, 22]. In ABC-SMC, a series of particle populations $P_t = \{(\theta_t^i, w_t^i)\}_{i \leq N}$ is generated, constituting samples of successively better approximations $\pi_{\text{ABC},\varepsilon_t}(\theta|y_{\text{obs}})$ of the posterior, for generations $t = 1, \dots, n_p$, with acceptance thresholds $\varepsilon_t > \varepsilon_{t+1}$. In the first generation ($t = 1$), particles are sampled directly from the prior, $g_1(\theta) = \pi(\theta)$. In later generations ($t > 1$), particles are sampled from proposal distributions $g_t(\theta) \gg \pi(\theta)$ based on the last generation's accepted weighted population P_{t-1} , e.g. via a kernel density estimate. The importance weights w_t^i are the Radon-Nikodym derivatives $w_t(\theta) = \pi(\theta)/g_t(\theta)$. This is precisely such that the weighted parameters are samples from the distribution

$$\int w_t(\theta) I[d(y, y_{\text{obs}}) \leq \varepsilon_t] \pi(y|\theta) dy \cdot g_t(\theta) = \int I[d(y, y_{\text{obs}}) \leq \varepsilon_t] \pi(y|\theta) dy \cdot \pi(\theta), \quad (2)$$

i.e. the target distribution (1) for $\varepsilon = \varepsilon_t$.

Common proposal distributions first select an accepted parameter from the last generation and then perturb it, in which case g_t takes the form $g_t(\theta) = \sum_{i=1}^N w_{t-1}^i K(\theta|\theta_{t-1}^i) / \sum_{i=1}^N w_{t-1}^i$, with e.g. $K(\theta|\theta_{t-1}^i) = \mathcal{N}(\theta|\theta_{t-1}^i, \Sigma_{t-1})$ a normal distribution with mean θ_{t-1}^i and covariance matrix Σ_{t-1} . The performance of ABC-SMC algorithms relies heavily on the quality of the proposal distribution, on its ability to efficiently explore the parameter space. Methods that adapt to the problem structure, e.g. basing Σ_{t-1} on the previous generation's weighted sample covariance matrix and potentially localizing around θ_b , have shown superior performance [23–25].

The output of ABC-SMC is a population of weighted parameters

$$P_{n_t} = \{(\theta_{n_t}^i, w_{n_t}^i)\}_{i \leq N} \sim \pi_{\text{ABC},\varepsilon_{n_t}}(\theta|y_{\text{obs}}).$$

For a statistic $f: \mathbb{R}^{n_\theta} \rightarrow \mathbb{R}$, the expected value under the posterior is then approximated via the self-normalized importance estimator

$$\mathbb{E}_{\pi_{\text{ABC},\varepsilon_{n_t}}(\theta|y_{\text{obs}})}[f] \approx \hat{f} = \sum_{i=1}^N W_{n_t}^i f(\theta_{n_t}^i),$$

which is asymptotically unbiased. Here, $W_t^i := w_t^i / \sum_{j=1}^N w_t^j$ are self-normalized weights. This is necessary because the weights $w_t(\theta) = \pi(\theta)/g_t(\theta)$ are not normalized in the joint sample space (θ, y) , therefore effectively another Monte Carlo estimator is employed for the normalization constant (for details see the Section 1.1 in [S1 File](#)).

In importance sampling, samples are assigned different weights, such that some impact estimates more than others. This can be quantified e.g. via the *effective sample size (ESS)* [8, 26]:

$$\text{ESS}(\{w_i\}_{i \leq N}) = \frac{(\sum_{i \leq N} w_i)^2}{\sum_{i \leq N} w_i^2} \quad (3)$$

Established parallelization strategies

In ABC, often hundred thousands to millions of model simulations need to be performed, which is typically the computationally critical part. To speed up inference, parallelization strategies have been developed that exploit the independence of the N particles constituting the t -th population. Suppose we have W parallel workers, each worker being a computational processor unit e.g. in an HPC environment. There are two established techniques to parallelize execution over the workers:

In *static scheduling (STAT)*, given a population size N , N tasks are defined and distributed over the workers. Each task consists in sampling until one particle gets accepted (Fig 1A). The tasks are queued if $N > W$. STAT minimizes the active computation time and number of simulations and is easy to implement, only requiring basic pooling routines available in most distributed computation frameworks. However, even for $W > N$ only N workers are employed, although the number of required simulations is usually substantially larger than N . In addition, at the end of every generation the number of active workers decreases successively, most workers idly waiting for a few to finish their tasks. STAT is available in most established ABC-SMC implementations [15, 16].

In *dynamic scheduling (DYN)*, sampling is performed continuously on all available workers until N particles have been accepted (Fig 1B). However, simply taking those first N particles as the final population would bias the population towards parameters causing short-running simulations. Therefore, DYN waits for all workers to finish, and out of then $\tilde{N} \geq N$ accepted particles, only the N that started earliest are finally accepted, not the ones that finished earliest. This has the effect that simulation time plays no longer a role in the acceptance decision [17]. This ensures that the acceptance probability of a particle is in accordance with the target distribution, independent of later events and thus its run-time.

Parallelization using look-ahead dynamic scheduling

DYN allows to exploit the available parallel infrastructure to a higher degree than STAT and therefore already substantially decreases the wall-time (by a factor of between 1.4 and 5.3 in test scenarios, see [17]). Nonetheless, some workers remain idle at the end of each generation while waiting for others to complete. This fraction of idle workers increases as the number of workers increases relatively to the population size. Additionally, the idle time increases if simulation times are heterogeneous, which is often the case, e.g. with estimated reaction rates determining the number of simulated events (Section 3.4.1 in S1 File). In case of fast model simulations, also the time between generations, e.g. to post-process and store results, may be relatively long.

Proposed algorithm. We propose to extend dynamic scheduling by using the free workers at the end of each generation to proactively sample for the next generation (Fig 2): As soon as N acceptances have been reached in generation $t - 1$ and workers thus start to get idle, we construct a preliminary proposal $\tilde{g}_t$, based on which particles for generation t are generated to start simulations on the free workers. $\tilde{g}_t$ can be based on a preliminary population of accepted

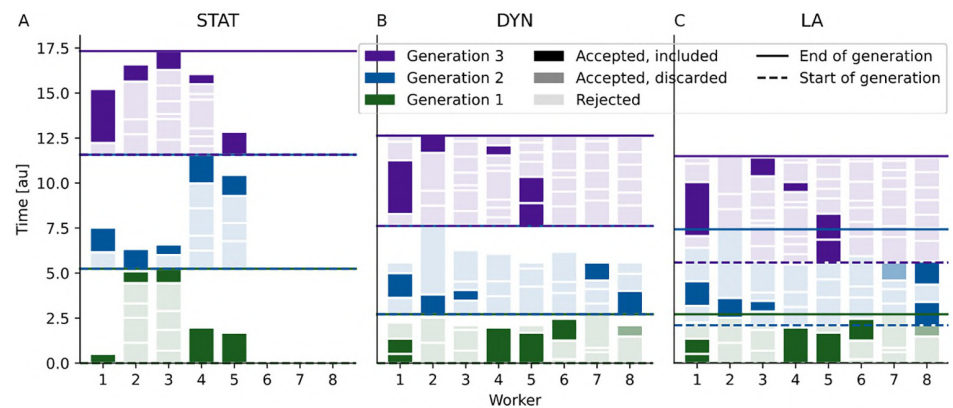


Fig 1. Illustration of core usage over run-time for static (STA), dynamic (DYN) and look-ahead (LA) scheduling for a population size $N = 5$ on $W = 8$ workers, over 3 generations (colors). The shading associated with each color indicates whether a sample satisfies the acceptance criterion and is included in the final population (dark shading), satisfies the acceptance criterion but is discarded because enough earlier-started accepted samples exist (intermediate shading, for DYN+LA), or does not satisfy the acceptance criterion and is rejected (light shading). Solid lines indicate the end of a generation, dashed lines indicate the (preliminary) beginning of a generation (different from solid lines only for LA). Delimiting white spaces between boxes indicate negligible post-processing times between simulations.

<https://doi.org/10.1371/journal.pone.0294015.g001>

particles $\hat{P}_{t-1} = \{(\hat{\theta}_{t-1}^i, \hat{w}_{t-1}^i)\}_{i \leq N}$ relying on these first N acceptances. However, $\hat{P}_{t-1}$ may introduce a practical bias (in a finite sample sense) towards particles with faster simulations times. This can in particular occur when computation time is highly parameter-dependent. Say, for example, that parameters from multiple regions in parameter space can explain the data similarly well, but that one region leads to substantially higher simulation times. Then, a sampling routine that does not wait for all started simulations to finish may under-represent or even miss out on regions in parameter space with high simulation times. The ABC-SMC routine may consequently have a low probability of generating importance samples from that region in subsequent generations, leading to a biased final posterior sample. To address this issue, the preliminary proposal can alternatively be based on P_{t-2} (such that $\tilde{g}_t = g_{t-1}$), giving

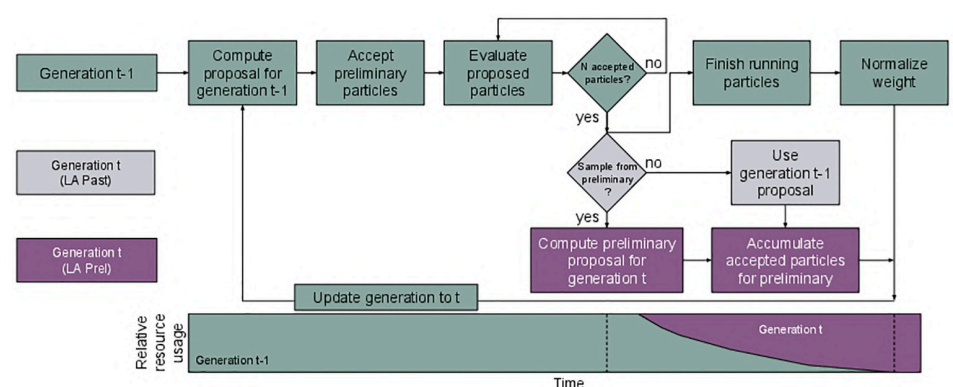


Fig 2. Concept visualization of look-ahead scheduling (LA). As soon as no more simulations are required for generation $t - 1$ (green), a preliminary simulation task for generation t is formulated either based on population P_{t-2} (grey, LA Past) or the preliminary population P_{t-1} (purple, LA Prel). Resulting simulations are considered when evaluating the next generation, and suitable weight normalization is applied to all samples (top right). Over time, the number of workers dedicated to generation $t - 1$ decreases, while that for generation t increases (bottom).

<https://doi.org/10.1371/journal.pone.0294015.g002>

inductively practically unbiased proposals. If a particle $\tilde{\theta}_t \sim \tilde{g}_t$ gets accepted according to the acceptance criteria of generation t , its non-normalized weight is calculated as $\tilde{w}_t(\tilde{\theta}_t) = \frac{\pi(\tilde{\theta}_t)}{\tilde{g}_t(\tilde{\theta}_t)}$. As soon as all simulations for generation $t - 1$ have finished and thus the actual P_{t-1} is available, all workers are updated to continue working with the actual sampling task based on proposal g_t . As the time-critical part of typical ABC applications is the model simulation, the cost of generating the preliminary sampling task is usually negligible.

The assessment of acceptance of preliminary samples depends on whether everything is pre-defined: If the acceptance components, including distance function d and acceptance threshold ε_t for generation t are defined a-priori, then acceptance can be checked directly on the workers without knowledge of the complete previous population P_{t-1} . If however any component of the algorithm is adaptive and hence based on P_{t-1} (e.g. the acceptance threshold is commonly chosen as a quantile of $\{d(y_{t-1}^i, y_{\text{obs}})\}_{i \leq N}$), the acceptance check must be delayed until the actual P_{t-1} is available. This allows to use one common acceptance criterion across all particles within a generation, so that all particles target the same distribution.

The population of generation t is then, corrected for run-time bias as in DYN by only considering the N accepted particles that were started first, given as

$$P_t = \{ \{(\tilde{\theta}_t^i, \tilde{w}_t^i)\}_{i \leq \tilde{N}}, \{(\theta_t^i, w_t^i)\}_{\tilde{N} < i \leq N} \}, \quad (4)$$

with $0 \leq \tilde{N} \leq N$ particles based on the preliminary proposal $\tilde{g}_t$, and $N - \tilde{N}$ on the final g_t . The weights need to be normalized appropriately, as explained in the following section. We call this parallelization strategy, which during generation $t - 1$ already looks ahead to generation t , *Look-ahead (dynamic) scheduling (LA)*.

Weights and unbiasedness. A key property of ABC methods is that they provide an asymptotically unbiased Monte Carlo sample from $\pi_{\text{ABC}, \varepsilon_{n_t}}(\theta | y_{\text{obs}})$, with $\pi_{\text{ABC}, \varepsilon_{n_t}}(\theta | y_{\text{obs}}) \rightarrow \pi(\theta | y_{\text{obs}})$ for $\varepsilon \rightarrow 0$. The sample (4) obtained via LA conserves this property: The point is that each subpopulation on its own gives an asymptotically unbiased estimator, since the weights $\tilde{w}_t(\tilde{\theta}) = \pi(\tilde{\theta})/\tilde{g}_t(\tilde{\theta})$, $w_t(\theta) = \pi(\theta)/g_t(\theta)$ are exactly the Radon-Nikodym derivatives w.r.t. the respective proposal distributions. Note that this theoretical unbiasedness holds regardless of whether $\tilde{g}_t$ is based on $\hat{P}_{t-1}$ or P_{t-2} , as long as $\tilde{g}_t(\theta) \gg \pi(\theta)$. As noted in the previous section, however a practical bias may occur due to finite sample size.

The subpopulation estimates are then combined, which decreases the Monte Carlo error due to the larger sample size. Instead of simply tossing all samples together, it is preferable to first normalize the weights relative to their subpopulation, $\tilde{W}_t^i := \tilde{w}_t^i / \sum_{i=1}^{\tilde{N}} \tilde{w}_t^i$, $W_t^i := w_t^i / \sum_{i=\tilde{N}+1}^N w_t^i$ (Section 1.3 in [S1 File](#)). This is because both weight functions are non-normalized, with generally different normalization constants, which renders them not directly comparable. A joint estimate based on the full population can then be given as

$$\mathbb{E}_{\pi_{\text{ABC}, \varepsilon_t}(\theta | y_{\text{obs}})}[f] \approx \beta \sum_{i=1}^{\tilde{N}} \tilde{W}_t^i f(\tilde{\theta}_t^i) + (1 - \beta) \sum_{i=\tilde{N}+1}^N W_t^i f(\theta_t^i) \quad (5)$$

with $\beta \in [0, 1]$ a free parameter. A straightforward choice is $\beta = \tilde{N}/N$, rendering the contribution of each subpopulation proportional to the respective number of samples. Instead, we propose to choose β to maximize the overall effective sample size (3), rendering the Monte Carlo estimate more robust. This is a simple constrained optimization problem with solution

$$\beta = \frac{\text{ESS}(\{\tilde{W}_t^i\}_{i \leq \tilde{N}})}{\text{ESS}(\{\tilde{W}_t^i\}_{i \leq \tilde{N}}) + \text{ESS}(\{W_t^i\}_{\tilde{N} < i \leq N})}$$

i.e. the contribution of each subpopulation is proportional to its effective sample size (Section 1.4 [S1 File](#)). Supposing that for $N \rightarrow \infty$, $\tilde{N}/N \rightarrow \alpha \in [0, 1]$, (5) converges to the left-hand side, as required. A more detailed derivation and extension to more than two proposal distributions is given in the Section 1 in [S1 File](#).

Implementation and availability

We implemented LA in the open-source Python tool pyABC [27], which already provided STAT and DYN. We employ a Redis low-latency server to handle the task distribution. If all components are pre-defined, we perform evaluation of the “look-ahead” samples $(\tilde{\theta}, \tilde{y})$ directly on the workers. If there are adaptive components, the delayed evaluation is performed on the main process. To avoid generating unnecessarily many preliminary samples in the presence of some very long-running simulations, we limited the number of preliminary samples to a default value of 10 times the number of samples in the current iteration. To not start preliminary sampling unnecessarily, we employed schemes predicting whether any termination criterion will be hit after the current generation. The code underlying this study can be found at https://github.com/EmadAlamoudi/Lookahead_study. A snapshot of code and data can be found at <https://doi.org/10.5281/zenodo.7875905>.

Results

Wall-time superiority of DYN over STAT has already been established in prior work [17]. To study the performance of LA and compare it to DYN, we applied both to several parameter estimation problems and in various scenarios of population size N and workers W . We distinguish between “LA Prel” using the preliminary $\hat{P}_{t-1}$ to generate $\tilde{g}_t$, and “LA Past” using P_{t-2} instead.

Test problems

We considered four problems ([Table 1](#)): Problems T1-T2 are simple test problems, while M1-M2 are realistic application examples.

Problem T1 is a bimodal model $y \approx \theta^2$, in which simulations from one mode have an artificially longer run-time. Specifically, if $\theta > 0$, a log-normally distributed simulation time of $\tau \sim \log \mathcal{N}(1, 2, 4)$ seconds was simulated. The goal of setting an artificially longer run-time was to specifically test the preliminary bias caused by only accepting simulations from one mode when constructing the preliminary proposal.

Problem T2 is an ordinary differential equation (ODE) model with 2 parameters describing a conversion reaction $x_1 \leftrightarrow x_2$, with observables obscured by random multiplicative noise. To analyze sampler behavior under simulation run-time heterogeneity, we added random log-normally distributed delay times t_{sleep} of various variances on top of the ODE simulations. For this model, run-times are fast, permitting repeated analyses to check correctness of the method, quantify stochastic effects and assess average behavior.

Table 1. Overview of application examples.

ID	Description	Implementation	n_θ
T1	Bimodal run-time-skewed model	Python	1
T2	Conversion reaction ODE model	Python	2
M1	Tumor spheroid growth [11]	C++	7
M2	Liver tissue regeneration [28]	Morpheus	14

<https://doi.org/10.1371/journal.pone.0294015.t001>

Problem M1 describes the growth of a tumor spheroid using a hybrid discrete-continuous approach, modeling single cells as stochastically interacting agents and extracellular substances [11]. The model combines a system of PDEs to describe the extracellular matrix, with a cellular Potts model (CPM) to describe cell configurations and mechanisms like cell division and cell death. The model has seven estimated parameters and outputs three observables, the growth curve, extra-cellular matrix and proliferation profiles.

Problem M2 describes the metabolic status of mechano-sensing during liver generation, describing the reaction network dynamics by a set of ODEs [28]. This model has 14 parameters and two observables, the nuclear YAP and total YAP intensities. These observables were quantified from image tiles covering an entire liver lobule with portal and central veins.

Further details about the test problems can be found in the Section 3 in [S1 File](#).

Biased proposal can induce practical bias in accepted population

The analysis of test model T1 revealed that for small population sizes N relative to the number of workers W , in combination with high acceptance rates, LA Prel can indeed lead to a bias towards short-running simulations ([Fig 3](#) right). This can happen when $\hat{P}_{t-1}$ is only based on short-running simulations, and solely proposes particles from that regime, enough of which are then accepted to form P_t . For larger N relative to W , this effect occurred less, likely because given large population sizes, sampling from other modes with associated high importance weights eventually happened.

Sampling from unbiased proposal solves bias

When replacing LA Prel by LA Past, i.e. sampling from P_{t-2} instead of $\hat{P}_{t-1}$, the bias no longer occurred ([Fig 3](#) right). This is expected, because $\tilde{g}_{t-2}$ has no run-time bias. In practice, we did not encounter any problems of practical bias on the considered application examples, where results from DYN, LA Prel and LA Past were highly consistent. Yet, LA Prel may fail in some situations, which also demonstrates that ABC-SMC algorithms are sensitive to potential bias in the proposal distribution. Thus, in the following, we focus on the stable LA Past algorithm, showing pendants for LA Prel in the [S1 File](#).

Look-ahead sampling gives accurate results

We used problem T2 to analyze different scenarios, with population sizes N from 20 to 1280 particles, worker numbers W from 32 to 256, and log-normally distributed simulation times of variances σ^2 from 1.0 to 4.0. We ran each scenario 13 times to obtain stable average statistics. We considered means and standard deviations as point and uncertainty measures.

Point estimates for DYN and LA converged to the same values across population sizes ([Fig 4A and 4B](#)). The proportion of accepted LA samples in the final population originating from the preliminary distribution ranged from nearly 100% to 50% (LA Prel) and 20% (LA Past), as expected decreasing for larger population sizes ([Fig 4E and 4F](#)). The more pronounced decrease for LA Past than LA Prel is reasonable because void of bias, $\hat{P}_{t-1}$ provides a better sampling distribution than P_{t-2} . Effective sample sizes were stable across DYN and LA ([Fig 4D](#)). A higher run-time variance lead to an increase in accepted samples originating from the preliminary proposal distribution ([S1 Fig in S1 File](#)). This is expected, because greater heterogeneity in run-times increases the chance of encountering exceptionally long-running simulations, which DYN has to wait for, while LA already proceeds.

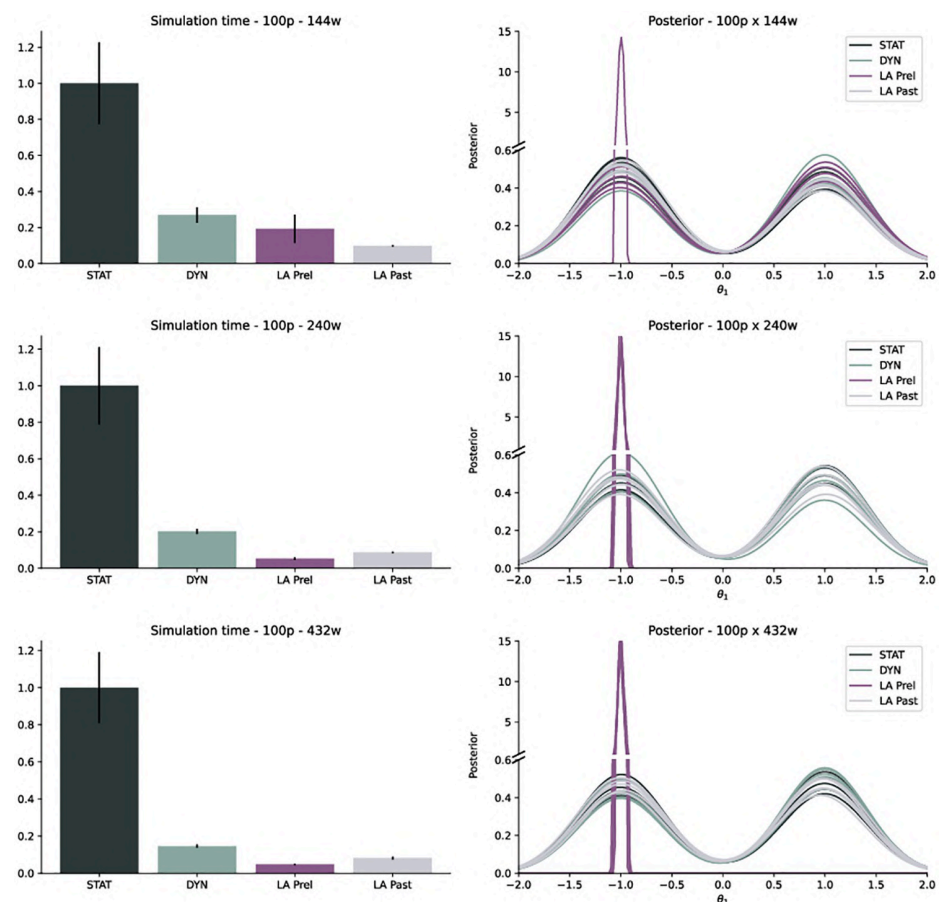


Fig 3. Run-time and posterior approximation for 5 different runs of model T1 with STAT, DYN, LA Prel and LA Past, with population size $N = 100$ on $W = 144, 240, 432$ workers (top to bottom).

<https://doi.org/10.1371/journal.pone.0294015.g003>

Considerable speed-up towards high worker numbers

To analyze the effect of scheduling strategy on the overall wall-time, we ran model T2 systematically for different population sizes and numbers of workers. We considered population sizes $32 \leq N \leq 1280$ and numbers of parallel workers $32 \leq W \leq 256$, which covers typical ranges used in practice. Each scenario was repeated between 13 times to assess average behavior, here we report mean values.

As a general tendency, the wall-time speed-up of LA over DYN became larger with increasing ratio of the number of workers by the population size. For a model sleep time variance of $\sigma^2 = 1$ (Fig 5), e.g. for $N = 20$ and $W = 256$, the average wall-time got reduced by a factor of almost 1.8. In most scenarios, a wall-time reduction by a factor of between 1.11 and 1.8 was observed. Only when the population size was large compared to the number of workers, the speed-up was comparably small. Generally, the increase in speed-up with increasing ratio W/N is as expected, as for large W the idle time between generations occurring for DYN and LA constitutes a more pronounced factor in the overall run-time.

For a sleep time variance of $\sigma^2 = 2$ (S2 Fig in S1 File), we observed similar behavior. There, the acceleration was generally more pronounced with up to a factor of roughly 1.9 and many

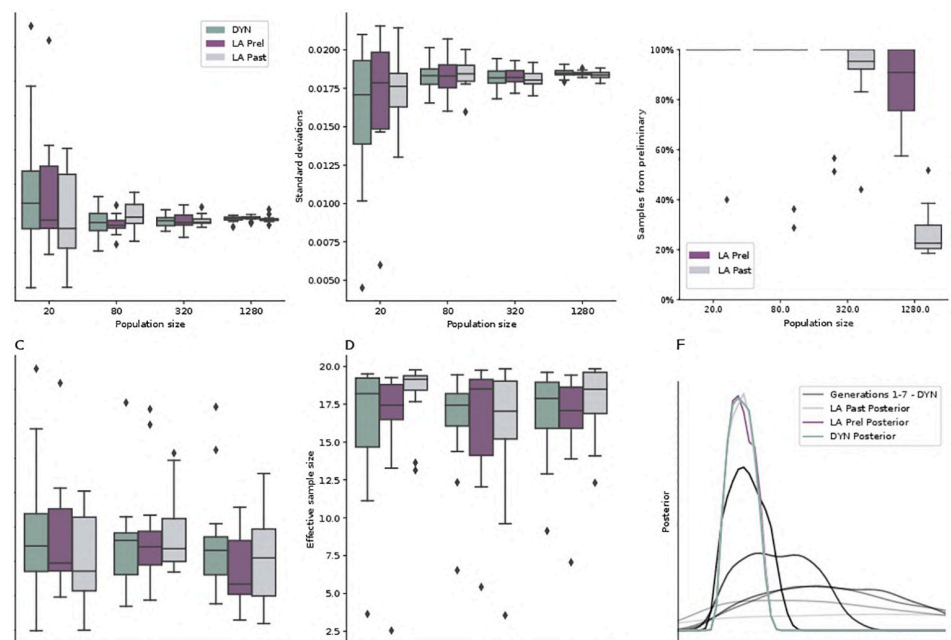


Fig 4. Results for problem T2 for different population sizes N , worker numbers W , and sleep time variances σ^2 . Unless otherwise specified, we used $N = 1280$, $W = 256$, and a log-normally distributed sleep time t_{sleep} of variance $\sigma^2 = 1$. To increase comparability, the ε_t values over $n_t = 8$ generations were pre-defined. (A) and (B): Mean and standard deviation of the posterior approximation $\pi_{\text{ABC}, \mathcal{E}_{n_t}}(\theta | y_{\text{obs}})$. Box-plot over 13 repetitions. (C): Posterior mean for different sleep time variances, for $W = 20$. (D): Effective sample size across different sleep time variances, for $N = 256$ and $W = 20$, in which case it is likely that several generations are sampled completely from the preliminary proposal. (E): Fraction $\tilde{N}/N$ of accepted samples in the final population $t = n_t$ that originate from the preliminary proposal $\tilde{g}_{n_t}(\theta)$ for LA Prel and LA Past. (F): Exemplary visualization of 1d posterior approximation marginals for single runs.

<https://doi.org/10.1371/journal.pone.0294015.g004>

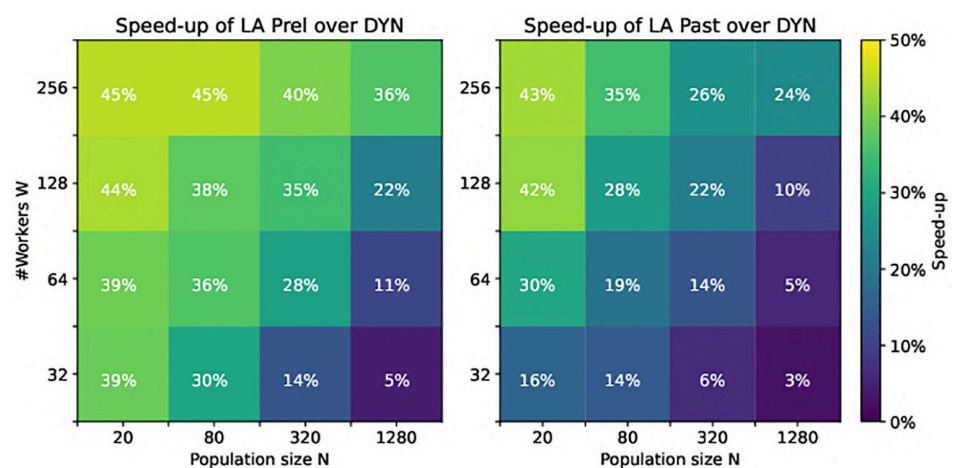


Fig 5. Speed-up ($1 - \{\text{Wall-time LA}\} / \{\text{Wall-time DYN}\}$) of LA Prel (left) and LA Past (right) over DYN for various population sizes and numbers of workers, for a model sleep time variance of $\sigma^2 = 1$.

<https://doi.org/10.1371/journal.pone.0294015.g005>

factors in the range 1.2 to 1.9. This indicates that indeed the advantage of LA over DYN is more pronounced in the presence of highly heterogeneous model simulation times.

Also on T1, the comparison of run-times (Fig 3 left) revealed a speed-up of LA over DYN. Further, we could confirm on both T1 and T2 (Fig 3 and S3 Fig in S1 File) the substantial speed-up DYN already provides over STAT, as reported in [17], on which we here improved further.

Scales to realistic application problems

Given the high simulation cost of the application problems M1–2, we only performed selected analyses to compare LA and DYN. A reliable comparison of run-times in real-life application examples is challenging, because the total wall-time varies strongly due to stochastic effects, and computations are too expensive to perform inference many times.

For the two models, the parameter estimates obtained using LA (both LA Per and LA Past) and DYN are consistent, except for expectable stochastic effects (S4 and S5 and S9–S11 Figs in S1 File). Together with the previous analyses, this indicates that for practical applications, the multi-proposal approach of LA allows for stable and accurate inference, similar to the single proposal used by DYN. In early generations, a considerable part of the accepted particles was based on the preliminary proposal distribution (near 100%), which then decreased in later generations (S6 and S12 Figs in S1 File). This is consistent with the decrease in acceptance rate and thus the relative time during which the preliminary and not the final proposal distribution is used.

For the tumor model M1, we used an adaptive quantile-based epsilon threshold schedule [29], with DYN, LA Prel and LA Past, population sizes $N \in \{250, 500, 1000\}$, and $W \in \{128, 256\}$ workers. For each considered configuration we performed 2 replicates (in total 8) to assess average behavior. Reported run-times are until a common final threshold was hit by all runs. The speed-up of LA over DYN varied depending on the ratio of population size and number of workers, similar to what we observed for T1+2. For high ratios, LA was consistently faster up to 35%. However, for low ratios, less improvement was observed. In some runs, LA was slightly slower than DYN (Fig 6). Over the 8 runs, we observed a mean speed-up of 21% (13%) and a median of 23% (16%) for LA Past (LA Prel). This indicates expected speed-ups of 13–23%. However, it should be remarked that large run-time differences and volatility could be traced back to single generations taking vast amounts of time (S7 Fig in S1 File). These long generations occurred in all scheduling variants and exist most likely because the epsilon for that generation was chosen too optimistically, indicating a weakness of the used epsilon scheme rather than the parallelization strategy.

For the liver regeneration model M2, we performed similar analyses, with adaptive quantile-based epsilon threshold schedules, population sizes $N \in \{250, 500, 1000\}$ and $W \in \{128, 256\}$ workers, with 2 replicates per configuration. Similar to model M1, we observed a faster performance of up to 35%. However, with a smaller ratio between population size and the number of workers, a slightly lower performance improvement was achieved (Fig 7). Similarly to M1, the acceleration varied quite strongly. For LA Prel we observed a mean speed-up over all 8 runs of 36% (median 31%) over DYN. However, for LA Past we observed contrarily a mean slow-down of 39% (median 43%) over DYN. It is not clear what caused this stark difference, which is again subject to high fluctuations. Further tests would be needed to assess the reasons for this specific model.

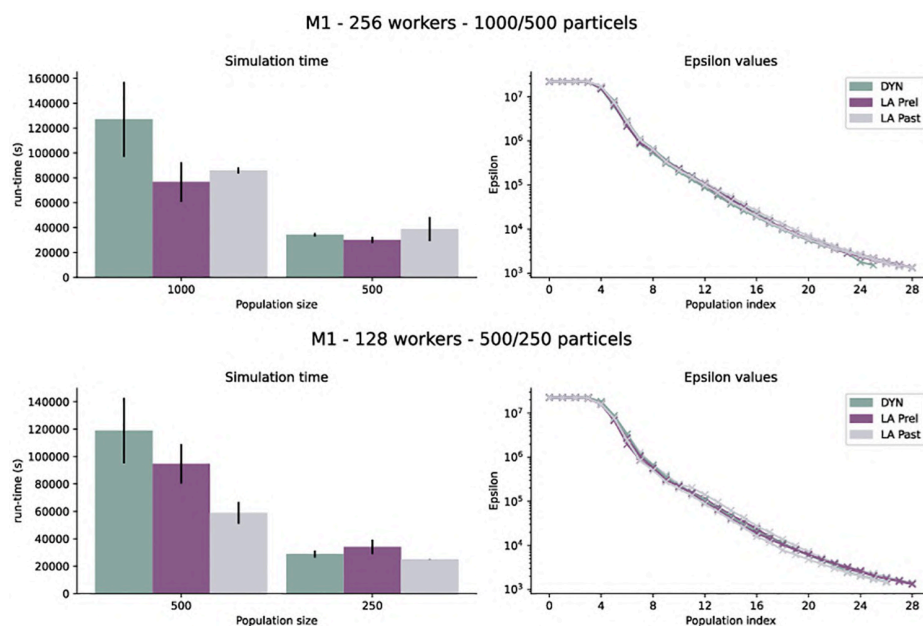


Fig 6. Run-time and posterior distributions for 2 different runs of model M1 with population size 1000, 500, 250 on 128 and 256 workers.

<https://doi.org/10.1371/journal.pone.0294015.g006>

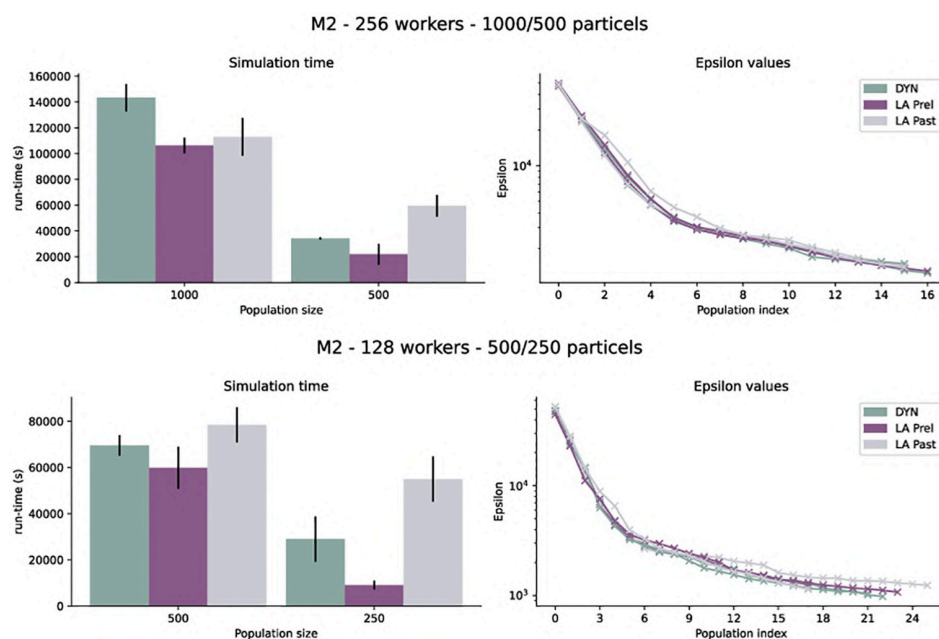


Fig 7. Run-time and posterior distribution for 2 different runs of model M2 with population size 1000, 500, 250 on 128 and 256 workers.

<https://doi.org/10.1371/journal.pone.0294015.g007>

Discussion

Simulation-based ABC methods have made parameter inference increasingly accessible even for complex stochastic models, which are however limited by computational costs. Here, we presented “look-ahead” sampling, a parallelization strategy to minimize wall-time and improve run-time efficiency by using all available high-performance computing resources at near-all times. On various test and application examples, we verified the accuracy and robustness of the novel approach in typical settings. Depending on model simulation run-time heterogeneity, and the relation of population size and the number of available cores, we observed a speed-up of up to 45% compared to dynamical scheduling as the previously most efficient strategy. Compared to widely used static scheduling, dynamic scheduling is already highly efficient, with limited room for improvement. Nevertheless, using the here proposed look-ahead sampling, on typical application examples, we observed a speed-up of often roughly 20–30%, however with some variability and sometimes efficiency on par with or even below dynamical scheduling. Assessing these variations in efficiency in more detail on expensive application examples would require further tests with considerable computational resources. Importantly, our analysis also demonstrates how ABC-SMC is sensitive to the choice of proposal distribution. Finite samples can induce a practical bias, as we observed here for parameter-dependent run-times of models—a problem that occurred in extreme cases but could only be solved by using look-ahead sampling with the previous, and not the preliminary, proposal distribution.

Conceptually and aside implementation details, the presented strategy provides the minimal wall-time among all parallelization strategies, as all cores are used at practically all times. We observed that look-ahead sampling using preliminary results (LA Prel) provided a performance speed-up over re-using the previous generation (LA Past), however at the cost of practical bias. Thus, LA Past constitutes the safe choice. Only if the possibility of critical parameter-dependent simulation times can be excluded, would we presently recommend LA Prel.

Were it possible to construct an unbiased proposal using those preliminary results, e.g. via reweighting or imbalance detection, we could thus increase the speed-up with robust performance. Alternatively, LA Past and LA Prel could also be combined, e.g. switching to LA Prel after a “burn-in”, when the probability of a bias toward short-running simulations is lessened.

When using delayed evaluation, it would be possible to parallelize the evaluation as well, which we have not done here. If evaluation times are long relative to simulation times, e.g. if (adaptive) summary statistics involve complex operations, this would be beneficial. In order to reduce a potential bias in the preliminary proposal distribution towards fast-running simulations, it may be beneficial to update it as soon as more particles finish. This would imply the use of more than two importance distributions, the theory of which we have already provided in the [S1 File](#).

In conclusion, we showed how we can minimize wall-time and associated computing cost of ABC samplers with substantial performance gains over established methods. Given that the concept is generally applicable for sequential importance sampling methods, it is of potential widespread use for different applications.

Supporting information

S1 File. Further details on the methods and results.
(PDF)

Author Contributions

Conceptualization: Jan Hasenauer, Yannik Schälte.

Formal analysis: Yannik Schälte.

Funding acquisition: Jan Hasenauer.

Investigation: Emad Alamoudi, Felipe Reck, Yannik Schälte.

Methodology: Emad Alamoudi, Felipe Reck, Yannik Schälte.

Software: Emad Alamoudi, Felipe Reck, Yannik Schälte.

Supervision: Jan Hasenauer, Yannik Schälte.

Validation: Emad Alamoudi, Felipe Reck, Nils Bundgaard, Frederik Graw, Lutz Brusch.

Visualization: Emad Alamoudi, Felipe Reck.

Writing – original draft: Emad Alamoudi, Felipe Reck, Jan Hasenauer, Yannik Schälte.

Writing – review & editing: Emad Alamoudi, Jan Hasenauer, Yannik Schälte.

References

1. Gershenfeld NA, Gershenfeld N. The nature of mathematical modeling. Cambridge university press; 1999.
2. Kitano H. Systems Biology: A Brief Overview. *Science*. 2002; 295(5560):1662–1664. <https://doi.org/10.1126/science.1069492> PMID: 11872829
3. Tarantola A. Inverse Problem Theory and Methods for Model Parameter Estimation. SIAM; 2005.
4. Tavaré S, Balding DJ, Griffiths RC, Donnelly P. Inferring coalescence times from DNA sequence data. *Genetics*. 1997; 145(2):505–518. <https://doi.org/10.1093/genetics/145.2.505> PMID: 9071603
5. Hasenauer J, Jagiella N, Hross S, Theis FJ. Data-Driven Modelling of Biological Multi-Scale Processes. *J Coupled Syst Multiscale Dyn*. 2015; 3(2):101–121. <https://doi.org/10.1166/jcsmd.2015.1069>
6. Pritchard JK, Seielstad MT, Perez-Lezaun A, Feldman MW. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Mol Biol Evol*. 1999; 16(12):1791–1798. <https://doi.org/10.1093/oxfordjournals.molbev.a026091> PMID: 10605120
7. Beaumont MA, Zhang W, Balding DJ. Approximate Bayesian Computation in Population Genetics. *Genetics*. 2002; 162(4):2025–2035. <https://doi.org/10.1093/genetics/162.4.2025> PMID: 12524368
8. Sisson SA, Fan Y, Beaumont M. Handbook of approximate Bayesian computation. Chapman and Hall/CRC; 2018.
9. Del Moral P, Doucet A, Jasra A. Sequential Monte Carlo samplers. *J R Stat Soc B*. 2006; 68(3):411–436. <https://doi.org/10.1111/j.1467-9868.2006.00553.x>
10. Sisson SA, Fan Y, Tanaka MM. Sequential Monte Carlo without likelihoods. *Proc Natl Acad Sci*. 2007; 104(6):1760–1765. <https://doi.org/10.1073/pnas.0607208104> PMID: 17264216
11. Jagiella N, Rickert D, Theis FJ, Hasenauer J. Parallelization and High-Performance Computing Enables Automated Statistical Inference of Multi-scale Models. *Cell Syst*. 2017; 4(2):194–206. <https://doi.org/10.1016/j.cels.2016.12.002> PMID: 28089542
12. Imle A, Kumberger P, Schnellbacher ND, Fehr J, Carrillo-Bustamante P, Ales J, et al. Experimental and computational analyses reveal that environmental restrictions shape HIV-1 spread in 3D cultures. *Nature Communications*. 2019; 10(1):2144. <https://doi.org/10.1038/s41467-019-09879-3> PMID: 31086185
13. Durso-Cain K, Kumberger P, Schälte Y, Fink T, Dahari H, Hasenauer J, et al. HCV spread kinetics reveal varying contributions of transmission modes to infection dynamics. *Viruses*. 2021; 13(7). <https://doi.org/10.3390/v13071308> PMID: 34372514
14. Alamoudi E, Schälte Y, Müller R, Starruß J, Bundgaard N, Graw F, et al. FitMultiCell: Simulating and parameterizing computational models of multi-scale and multi-cellular processes. *bioRxiv*. 2023. <https://doi.org/10.1093/bioinformatics/btad674> PMID: 37947308
15. Dutta R, Schoengens M, Onnela JP, Mira A. ABCpy: A User-Friendly, Extensible, and Parallel Library for Approximate Bayesian Computation. In: Proceedings of the Platform for Advanced Scientific Computing Conference. PASC'17. New York, NY, USA: ACM; 2017. p. 8:1–8:9.
16. Kangasrääsio A, Lintusaari J, Skytén K, Järvenpää M, Vuollekoski H, Gutmann M, et al. ELFI: Engine for Likelihood-Free Inference. In: NIPS 2016 Workshop on Advances in Approximate Bayesian Inference; 2016.

17. Klinger E, Rickert D, Hasenauer J. pyABC: distributed, likelihood-free inference. *Bioinf.* 2018; 34(20):3591–3593. PMID: [29762723](#)
18. Wilkinson RD. Approximate Bayesian computation (ABC) gives exact results under the assumption of model error. *Stat Appl Gen Mol Bio.* 2013; 12(2):129–141. <https://doi.org/10.1515/sagmb-2013-0010> PMID: [23652634](#)
19. Schälte Y, Alamoudi E, Hasenauer J. Robust adaptive distance functions for approximate Bayesian inference on outlier-corrupted data. *bioRxiv.* 2021;.
20. Fearnhead P, Prangle D. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. *J R Stat Soc B.* 2012; 74(3):419–474. <https://doi.org/10.1111/j.1467-9868.2011.01010.x>
21. Toni T, Welch D, Strelkowa N, Ipsen A, Stumpf MPH. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *J R Soc Interface.* 2009; 6:187–202. <https://doi.org/10.1098/rsif.2008.0172> PMID: [19205079](#)
22. Beaumont MA. Approximate Bayesian computation in evolution and ecology. *Annual review of ecology, evolution, and systematics.* 2010; 41:379–406. <https://doi.org/10.1146/annurev-ecolsys-102209-144621>
23. Beaumont MA, Cornuet JM, Marin JM, Robert CP. Adaptive approximate Bayesian computation. *Biometrika.* 2009; 96(4):983–990. <https://doi.org/10.1093/biomet/asp052>
24. Filippi S, Barnes CP, Cornebise J, Stumpf MP. On optimality of kernels for approximate Bayesian computation using sequential Monte Carlo. *Stat Appl Genet Mol.* 2013; 12(1):87–107. PMID: [23502346](#)
25. Klinger E, Hasenauer J. A scheme for adaptive selection of population sizes in Approximate Bayesian Computation—Sequential Monte Carlo. In: Feret J, Koepl H, editors. *Computational Methods in Systems Biology.* CMSB 2017. vol. 10545 of Lecture Notes in Computer Science. Springer, Cham; 2017.
26. Liu JS, Chen R, Wong WH. Rejection control and sequential importance sampling. *J Am Stat Assoc.* 1998; 93(443):1022–1031. <https://doi.org/10.1080/01621459.1998.10473764>
27. Schälte Y, Klinger E, Alamoudi E, Hasenauer J. pyABC: Efficient and robust easy-to-use approximate Bayesian computation. *J Open Source Softw.* 2022; 7(74):4304. <https://doi.org/10.21105/joss.04304>
28. Meyer K, Morales-Navarrete H, Seifert S, Wilsch-Braeuninger M, Dahmen U, Tanaka EM, et al. Bile canaliculi remodeling activates YAP via the actin cytoskeleton during liver regeneration. *Mol Syst Biol.* 2020; 16(2):e8985. <https://doi.org/10.15252/msb.20198985> PMID: [32090478](#)
29. Drovandi CC, Pettitt AN. Estimation of parameters for macroparasite population evolution using approximate Bayesian computation. *Biometrics.* 2011; 67(1):225–233. <https://doi.org/10.1111/j.1541-0420.2010.01410.x> PMID: [20345496](#)

Supplementary information for: A Wall-time Minimizing Parallelization Strategy for Approximate Bayesian Computation

Emad Alamoudi^{1,*}, Felipe Reck^{1,*}, Nils Bundgaard², Frederik Graw^{2,3,4}, Lutz Brusch⁵, Jan Hasenauer^{1,6,7}, and Yannik Schälte^{1,6,7}

¹ University of Bonn, Life and Medical Sciences Institute, 53113 Bonn, Germany

² Heidelberg University, BioQuant - Center for Quantitative Biology, 69120 Heidelberg, Germany

³ Heidelberg University, Interdisciplinary Center for Scientific Computing, 69120 Heidelberg, Germany

⁴ Friedrich-Alexander-University Erlangen-Nürnberg, Department of Medicine 5, 91054 Erlangen, Germany

⁵ Technische Universität Dresden, Center of Information Services and High Performance Computing (ZIH), 01062 Dresden, Germany

⁶ Helmholtz Zentrum München, Institute of Computational Biology, 85764 Neuherberg, Germany

⁷ Technische Universität München, Center for Mathematics, 85748 Garching, Germany

* These authors contributed equally to this work.

Contents

1	Multiple proposal importance sampling	2
1.1	Standard importance sampling	2
1.2	Effective sample size	3
1.3	Importance sampling with multiple proposal distributions	4
1.4	Subpopulation weighting	5
2	Implementation	6
2.1	Algorithm implementation in pyABC	6
2.2	Technical specifications	6

3	Test models	6
3.1	(T1) Unbalanced Modes	7
3.2	(T2) ODE Model	7
3.3	(M1) Tumor Growth Model	12
3.4	(M2) Liver regeneration model	18
3.4.1	Simulation time variability	26

1 Multiple proposal importance sampling

Let $\theta \in \mathbb{R}^{n_\theta}$ denote the latent parameters, $\pi(\theta)$ the prior, and $\pi(y|\theta)$ the likelihood which we assume to be able to sample data $y \in \mathbb{R}^{n_y}$ from, while direct evaluation is not possible. Let $y_{\text{obs}} \in \mathbb{R}^{n_y}$ be the observed data. By Bayes' Theorem, we get the posterior distribution

$$\pi(\theta|y_{\text{obs}}) = \frac{\pi(y_{\text{obs}}|\theta)\pi(\theta)}{\pi(y_{\text{obs}})} \propto \pi(y_{\text{obs}}|\theta)\pi(\theta)$$

of parameters given observed data.

Let $\varepsilon > 0$ and $d : \mathbb{R}^{n_y} \times \mathbb{R}^{n_y} \rightarrow \mathbb{R}_{\geq 0}$ be a distance metric. ABC methods can be interpreted to generate samples from the joint distribution

$$(\theta, y) \sim \pi_{\text{ABC}, \varepsilon}(\theta, y|y_{\text{obs}}) \propto I[d(y, y_{\text{obs}}) \leq \varepsilon] \pi(y|\theta) \pi(\theta), \quad (1)$$

such that the target marginal of interest,

$$\pi_{\text{ABC}, \varepsilon}(\theta|y_{\text{obs}}) \propto \int I[d(y, y_{\text{obs}}) \leq \varepsilon] \pi(y|\theta) dy \cdot \pi(\theta)$$

can be obtained by simple projection to the first component. $\pi_{\text{ABC}, \varepsilon}(\theta|y_{\text{obs}})$ is an approximation of the actual parameter posterior, with $\pi(\theta|y_{\text{obs}}) = \lim_{\varepsilon \searrow 0} \pi_{\text{ABC}, \varepsilon}(\theta|y_{\text{obs}})$ under mild assumptions.

1.1 Standard importance sampling

We employ importance sampling (ABC-IS) embedded in a sequential Monte-Carlo scheme (ABC-SMC). For a proposal distribution $g(\theta) \gg \pi(\theta)$ and a target population size N , we use an ABC-IS scheme as shown in Algorithm 1. In ABC-SMC, Algorithm 1 is then iterated multiple times for successively refined thresholds ε and proposals g . Here we focus on a single such iteration.

This form of ABC-IS generates samples from the distribution

$$(\theta, y) \sim G(\theta, y) \propto I[d(y, y_{\text{obs}}) \leq \varepsilon] \pi(y|\theta) g(\theta). \quad (2)$$

Here, the $g(\theta)$ is because we sample $\theta \sim g(\theta)$, further we simulate data y from the likelihood and discard particles not satisfying the distance condition, such that

$$y|\theta \sim I[d(y, y_{\text{obs}}) \leq \varepsilon] \pi(y|\theta).$$

while *less than N acceptances* **do**
 sample parameter $\theta \sim g(\theta)$
 simulate data $y \sim \pi(y|\theta)$
 accept θ if $d(y, y_{\text{obs}}) \leq \varepsilon$
compute weights $w_i = \frac{\pi(\theta^i)}{g(\theta^i)}$, for accepted parameters $\{\theta^i\}_{i \leq N}$
output: weighted samples $\{(\theta_i, w_i)\}_{i \leq N}$
Algorithm 1: Importance ABC algorithm.

Therefore, the importance weights, Radon-Nikodym derivatives, of the proposal (2) against the target (1) are given by

$$v(\theta, y) = \frac{\pi_{\text{ABC}, \varepsilon}(\theta, y|y_{\text{obs}})}{G(\theta, y)} = \frac{C \cdot I[d(y, y_{\text{obs}}) \leq \varepsilon] \pi(y|\theta) \pi(\theta)}{C_G \cdot I[d(y, y_{\text{obs}}) \leq \varepsilon] \pi(y|\theta) g(\theta)} = \frac{C}{C_G} \frac{\pi(\theta)}{g(\theta)} = \frac{C}{C_G} w(\theta)$$

with normalization constants C, C_G . If all normalizations were known exactly, we could calculate the unbiased importance estimator of a test function f over the ABC posterior as

$$\mathbb{E}_{\pi_{\text{ABC}, \varepsilon}(\theta, y|y_{\text{obs}})}[f] = \mathbb{E}_{G(\theta, y)}[vf] \approx \frac{1}{N} \sum_i v_i f(\theta_i, y_i).$$

In general, we however do not know C and C_G and thus v only up to normalization, and therefore need to employ the self-normalizing estimate

$$\mathbb{E}_{\pi_{\text{ABC}, \varepsilon}(\theta, y|y_{\text{obs}})}[f] = \frac{\mathbb{E}_{G(\theta, y)}[wf]}{\mathbb{E}_{G(\theta, y)}[w]} \approx \frac{\frac{1}{N} \sum_i w_i f(\theta_i, y_i)}{\frac{1}{N} \sum_i w_i} = \sum_i W_i f(\theta_i, y_i)$$

with self-normalized weights

$$W_i := \frac{w_i}{\sum_j w_j},$$

which uses another Monte-Carlo approximation for the normalization constant. It is only asymptotically unbiased, converging almost surely as $N \rightarrow \infty$.

Remark: Were we instead to accept all particles irrespective of the distance, as some implementations do, we would need to regard $I[(y, y_{\text{obs}}) \leq \varepsilon]$ as part of the weighting, as in that case $G(\theta, y) \propto \pi(y|\theta)g(\theta)$ s.t. $v(\theta, y) = C \cdot I[d(y, y_{\text{obs}}) \leq \varepsilon] \pi(\theta)/g(\theta)$. In general, this interchangeability of importance and rejection sampling occurs in various places in ABC algorithms. Further note that the here presented scenario based on a uniform acceptance kernel $I[d(\cdot, y_{\text{obs}}) \leq \varepsilon]$, which is the most widely used one, naturally generalizes to arbitrary acceptance kernels $K_\varepsilon(\cdot, y_{\text{obs}})$.

1.2 Effective sample size

Importance sampling allows to e.g. better explore high-density regions by tailored proposals. However, as the particles need to be weighted, some contribute more to estimates than others, which can be interpreted as having a lower *effective sample size (ESS)* than a sample of the same size

directly from the target distribution. A common definition of the ESS can be motivated as follows: Consider a linear combination

$$S_N = \frac{\sum_{i \leq N} w_i X_i}{\sum_{i \leq N} w_i}$$

of i.i.d. random variables X_i of variance $\sigma^2 > 0$, with weights $w_i \geq 0$. The unweighted mean $\frac{1}{N_e} \sum_{i \leq N_e} X_i$ of N_e variables has variance σ^2/N_e . Equating, we obtain

$$\frac{\sigma^2}{N_e} \stackrel{!}{=} \text{Var}(S_N) = \frac{\sum_{i \leq N} w_i^2}{(\sum_{i \leq N} w_i)^2} \sigma^2 \Rightarrow \text{ESS} := N_e = \frac{(\sum_{i \leq N} w_i)^2}{\sum_{i \leq N} w_i^2} \quad (3)$$

as a scale-invariant quantification of the ESS of the weighted sum.

1.3 Importance sampling with multiple proposal distributions

Now assume we have multiple proposal distributions $\{g_l(\theta)\}_{l \leq L}$ instead of just one, and have generated $N_l > 0$ accepted particles $\{(\theta_i^l, w_i^l)\}_{i \leq N_l}$ from each proposal, with $N = \sum_l N_l$, giving a total population $P = \{\{(\theta_i^l, w_i^l)\}_{i \leq N_l}\}_{l \leq L}$ of size N . This generalizes the scenario presented in the main manuscript, where we have two proposal distributions, a preliminary one and a final one.

Assume $N_l/N \rightarrow \alpha_l \in [0, 1]$ for $N \rightarrow \infty$. Denote the normalized importance densities $v^l(\theta, y) = \pi_{\text{ABC}, \varepsilon}(\theta, y | y_{\text{obs}}) / G_l(\theta, y) = C_l w^l(\theta, y)$ with normalization constants $C_l > 0$.

The goal is to for a test function f define a robust Monte-Carlo estimate making use of all samples. One possible estimator could be obtained by just ignoring the fact that multiple proposals were used and just throwing all samples and weights together:

$$\begin{aligned} \mathbb{E}_{\pi_{\text{ABC}, \varepsilon}(\theta, y | y_{\text{obs}})}[f] &\xleftarrow{N \rightarrow \infty} \frac{\frac{1}{N} \sum_l \sum_{i \leq N_l} w_i^l f(\theta_i^l, y_i^l)}{\frac{1}{N} \sum_l \sum_{i \leq N_l} w_i^l} = \frac{\sum_l \frac{N_l}{N} \frac{1}{N_l} \sum_{i \leq N_l} w_i^l f(\theta_i^l, y_i^l)}{\sum_l \frac{N_l}{N} \frac{1}{N_l} \sum_{i \leq N_l} w_i^l} \\ &\xrightarrow{N \rightarrow \infty} \frac{\sum_l \alpha_l \mathbb{E}_{G_l(\theta, y)}[w^l f]}{\sum_l \alpha_l \mathbb{E}_{G_l(\theta, y)}[w^l]} = \frac{\sum_l \alpha_l C_l^{-1} \mathbb{E}_{\pi_{\text{ABC}, \varepsilon}(\theta, y | y_{\text{obs}})}[f]}{\sum_l \alpha_l C_l^{-1}} \\ &= \mathbb{E}_{\pi_{\text{ABC}, \varepsilon}(\theta, y | y_{\text{obs}})}[f]. \end{aligned} \quad (4)$$

Equation (4) shows that we obtain an asymptotically unbiased estimate, however the subpopulations are weighted by the normalization constants C_l , which is in general not meaningful, unless these carry an appropriate interpretation. Further, this joint estimator does not account for the difference in importance estimator quality of the various G_l .

Instead of (4), we suggest to first consider each subpopulation separately, and obtain a joint estimate as a weighted sum of estimators

$$\begin{aligned} \mathbb{E}_{\pi_{\text{ABC}, \varepsilon}(\theta, y | y_{\text{obs}})}[f] &= \sum_l \beta_l \mathbb{E}_{G_l(\theta, y)}[v^l f] = \sum_l \beta_l \frac{\mathbb{E}_{G_l(\theta, y)}[w^l f]}{\mathbb{E}_{G_l(\theta, y)}[w^l]} \\ &\xleftarrow{N \rightarrow \infty} \sum_l \beta_l \sum_{i \leq N_l} w_i^l f(\theta_i^l, y_i^l) \end{aligned} \quad (5)$$

with coefficients β_l s.t.

$$\sum_l \beta_l = 1, \quad (6)$$

and

$$W_i^l := \frac{w_i^l}{\sum_{j \leq N_l} w_j^l}$$

subpopulation-wise self-normalized weights.

1.4 Subpopulation weighting

The β_l are free parameters and should be chosen to yield a robust estimator. A straightforward choice is

$$\beta_l = \alpha_l \approx \frac{N_l}{N}, \quad (7)$$

i.e. to normalize the contribution of each proposal by the number of samples generated from that proposal.

As the proposal quality can vary, this weighting may not be ideal. Instead of (7), we suggest to choose the β_l to e.g. maximize the overall ESS, in order to obtain a robust estimator. Here, (3) takes the shape

$$\text{ESS} = \frac{(\sum_l \sum_{i:l} \beta_l W_i^l)^2}{\sum_l \sum_{i:l} (\beta_l W_i^l)^2} = \frac{1}{\sum_l \beta_l^2 \sum_{i \leq N_l} (W_i^l)^2}. \quad (8)$$

Denote for short $Q_l := \sum_{i \leq N_l} (W_i^l)^2$, the diagonal matrix $Q := \text{diag}(Q_1, \dots, Q_L) \in \mathbb{R}^{L \times L}$, and the column vectors $\beta := (\beta_1, \dots, \beta_L)^T \in \mathbb{R}^L$ and $\mathbb{1} := (1, \dots, 1)^T \in \mathbb{R}^L$. Without loss of generality, assume that $Q_l > 0$ for all l , otherwise just set $\beta_l = 0$. Then, maximizing (8) subject to (6) is equivalent to the quadratic form with linear equality constraint

$$\begin{aligned} & \text{minimize} && \beta^T Q \beta \\ & \text{subject to} && \mathbb{1}^T \beta = 1. \end{aligned} \quad (9)$$

Evaluating $\nabla_{\beta, \lambda} [\beta^T Q \beta + \lambda(\mathbb{1}^T \beta - 1)]$, using a Lagrange multiplier $\lambda \in \mathbb{R}$, readily shows that the unique solution of (9) is given by the positive-definite linear system

$$\begin{pmatrix} 2Q & \mathbb{1} \\ \mathbb{1}^T & 0 \end{pmatrix} \cdot \begin{pmatrix} \beta \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \Leftrightarrow \begin{pmatrix} \beta \\ \lambda \end{pmatrix} = \begin{pmatrix} \frac{Q^{-1} \mathbb{1}}{\mathbb{1}^T Q^{-1} \mathbb{1}} \\ \frac{-2}{\mathbb{1}^T Q^{-1} \mathbb{1}} \end{pmatrix}$$

Written out, we obtain

$$\beta_l = \frac{\frac{1}{\sum_{i \leq N_l} (W_i^l)^2}}{\sum_k \frac{1}{\sum_{i \leq N_k} (W_i^k)^2}} = \frac{\text{ESS}(\{W_i^l\}_{i \leq N_l})}{\sum_k \text{ESS}(\{W_i^k\}_{i \leq N_k})}, \quad l = 1, \dots, L,$$

i.e. the overall ESS is maximized by setting the contribution of sub-population l proportional to its ESS, which intuitively makes sense.

2 Implementation

2.1 Algorithm implementation in pyABC

The ABC-SMC algorithm and all samplers used and developed in this work have been implemented and made available as part of the python-based open-source package pyABC (<https://github.com/icb-dcm/pyabc>). The sampler implementation uses the *Redis* package (<https://redis.io>) as broker between main process and workers on a distributed high-performance computing (HPC) architecture.

Before the analysis, a Redis server is set up. To this server the desired number of workers is then connected. Workers can also be added interactively during the analysis. In addition, there is a process for the main analysis, the ABC-SMC algorithm, which submits in each generation simulation tasks to the server, which broadcasts them to the workers and collects results, to be read in again by the main process. The whole process of environment setup and analysis has been fully automated for HPC infrastructure.

Simulation tasks are broadcast by the server via dedicated port channels, which workers listen on if they are idle. Workers return simulation results via a queue, which is continually collected from by the main process. Start times are tracked via shared variables to ensure the main process is aware which tasks need to be waited for.

2.2 Technical specifications

For our tests we used Anaconda 3, Python 3.8, with package versions pyABC 0.12.07, tumor2d 1.0.0, Morpheus 2.2.0 [1]. Analyses of model T1 was performed on the Juelich Supercomputing Center (JSC), Juwels cluster, standard compute nodes, specification of which are $2 \times$ Intel Xeon Platinum 8168 CPU, 2×24 cores, 2.7 GHz 96 (12×8) GB DDR4, 2666 MHz.

Analyses of model T2, M1, and M2 were performed on the Bonna cluster at the University of Bonn. A standard node has the specification of 2×16 cores, 6 GB RAM per core, and 2x480GB SSD. For the different models, we used 1-9 nodes for T1 and 1-8 nodes for T2, M1 and M3.

3 Test models

To test the correctness and the performance of the new concept, we used 4 models as described in the Main Manuscript, Table 1, which we describe in further detail in this section. First, we used LA on some basic toy models (T1 and T2) to examine the general behavior and some specific properties. Later, we also performed tests on more complex application examples (M1 and M2) to test the performance in realistic settings.

Table 1: List of fitted parameters of model T1. All parameters have been transformed to a logarithmic scale with base 10.

Parameter	Description	Prior
$\log_{10}(\theta)$	A squared parameter	U(-2,-4)

3.1 (T1) Unbalanced Modes

Usually, the run-time of a single simulation will not be independent of the parameter candidate but instead, be somehow correlated to the region of the parameter space. High variance of the run-time paired with correlation between the simulation time and the candidate could possibly lead to a strong imbalance in the preliminary proposal. This raises the question of what happens when it is likely that the preliminary population is strongly biased.

To analyze this phenomenon, we constructed a two-mode scenario by considering a model that simply squares its only parameter, i.e. $y = \theta^2$ and then adds additive noise to that. Assuming the observed data consist of one observation with value 1 and that the prior is uniformly distributed on $[-2, 4]$, we get two parameters that could equally be true, -1 and $+1$, see Table 1. We added a shorter idle time sampled from a log-normal distribution with mean 0.05 and $\sigma^2=0.025$ to the negative mode and a larger one (mean=1 and $\sigma^2=0.5$) to the positive mode. Given a large number of parallel workers and a comparably small population sizes, it is thus possible that the preliminary population $\hat{P}_{t-1}$ consists entirely of samples from the negative side. Depending on the shape of the thus created preliminary proposal $\tilde{g}_t$, it is then possible that it only proposes samples from the negative side. It can happen that enough of these get accepted, such that the subsequent accepted population P_t consists only of samples from the negative side.

3.2 (T2) ODE Model

While ODE based models are not the main application area of ABC, it is fairly simple to create an example for which we can easily verify the correct result. Parameter inference for such a model is also not as time intensive as real life application examples. Therefore, we can perform it several hundred times with the new concept and compare the results to the ones of established approaches, making this a good test model to perform sanity checks on the new algorithm.

Our basic ODE-model with two parameters describes the inter-conversion between two species x_1, x_2 , with rates θ_1, θ_2 . It is given by

$$\frac{d}{dt}x_1 = -\theta_1x_1 + \theta_2x_2 \quad (10)$$

$$\frac{d}{dt}x_2 = \theta_1x_1 - \theta_2x_2. \quad (11)$$

Using this ODE, we can immediately verify that the results converge towards the same value as the original version for which the asymptotical correctness is known. In order to not have a fully

deterministic system and to consider that in reality measurements are not perfectly accurate, we added multiplicative normal noise $\sim \mathcal{N}(1, 0.03)$ to each model evaluation.

To analyse the run-times and the effect of heterogeneous simulation times, we used the same ODE and added some idle time to each model call. This idle time was chosen in a way that imitates some single simulations taking vastly longer than the average value. As distribution for t_{idle} we employed a log-normal one $t_{idle} \sim \text{lognormal}(\mu_n, \sigma_n^2)$, where μ_n, σ_n^2 are the mean and variance of the underlying normal distribution. μ_n, σ_n^2 were chosen such that the real variance σ^2 takes values between 0.25 and 4 and the real mean μ is constantly equal to 1 for the different values of σ^2 , i.e.

$$\sigma_n^2 = 2 \log \left(\frac{\sqrt{1^2 + \sigma^2}}{1} \right), \quad \mu_n = \log(1) - \frac{\sigma_n^2}{2}. \quad (12)$$

We used as prior a uniform distribution on the unit square $\pi \sim U([0, 1]^2)$, initial values $(x_1, x_2)(0) = (1, 0)$, and as true parameters $\theta_1 = e^{-2.5} \approx 0.0821$, $\theta_2 = e^{-2} \approx 0.135$, so here a point close to one corner of the prior distribution see Table 2. As observed data y_{obs} we considered the trajectory of species x_2 at time points $0, 1, \dots, 10$. As distance function we used an l_1 norm. Candidates are accepted if the distance between their simulated data and the observation falls below a fixed ε -threshold for each generation, with pre-defined thresholds of $\varepsilon_1, \dots, \varepsilon_8 = 8, 4, 2, 1, 0.7, 0.5, 0.33, 0.25$.

Table 2: List of fitted parameters of model T2. All parameters have been transformed to a logarithmic scale with base 10.

Parameter	Description	Prior
$\log_{10}(\theta_1)$	Conversion rate of x_1	$U(0, 1)^2$
$\log_{10}(\theta_2)$	Conversion rate of x_2	$U(0, 1)^2$

Results – Correctness

Using (T2) we generated 13 replicas of different configurations of both using DYN and LA scheduling for each population size on 32 to 256 workers.

Both scheduling approaches seem to return equal results (see Figure 4) when taking into account that any two runs will always slightly differ due to the innate random nature of the sampling, the noise and the run-time.

(A) demonstrates how the mean for both DYN and LA converges to the same value as we increase the population size. At the same time, (B) shows that also the standard deviation within individual distributions decreases, meaning that, as N increases, both posteriors peak more sharply at the same value. (C) makes it clear that a stronger variances for t_{idle} does not affect the mean, even though there is a larger fraction of preliminary particles expected. In (D), one can see that LA does not seem to substantially affect the effective sample size either even though we here used $N = 20$, which, on 256 workers, makes it likely that several generations are sampled completely from the preliminary. In E and F, we can see the fraction $\tilde{N}/N$ of accepted samples in the final population at

$t = n_t$ that were generated using the preliminary proposal $\tilde{g}_{n_t}(\theta)$. In G, we can see a comparison of the posterior distribution of the different scheduling. One can appreciate the nice agreement in the shape of the posterior from different scheduling strategies. By showing that the preliminary and the final proposal both considerably contribute to the posterior distribution, we get a strong indication that everything works as intended. And indeed, in many of the relevant scenarios the amount of particles that were based on the preliminary in the last generation ends up making up roughly half of the population, which is returned as output by the ABC-SMC algorithm (see Figure S1).

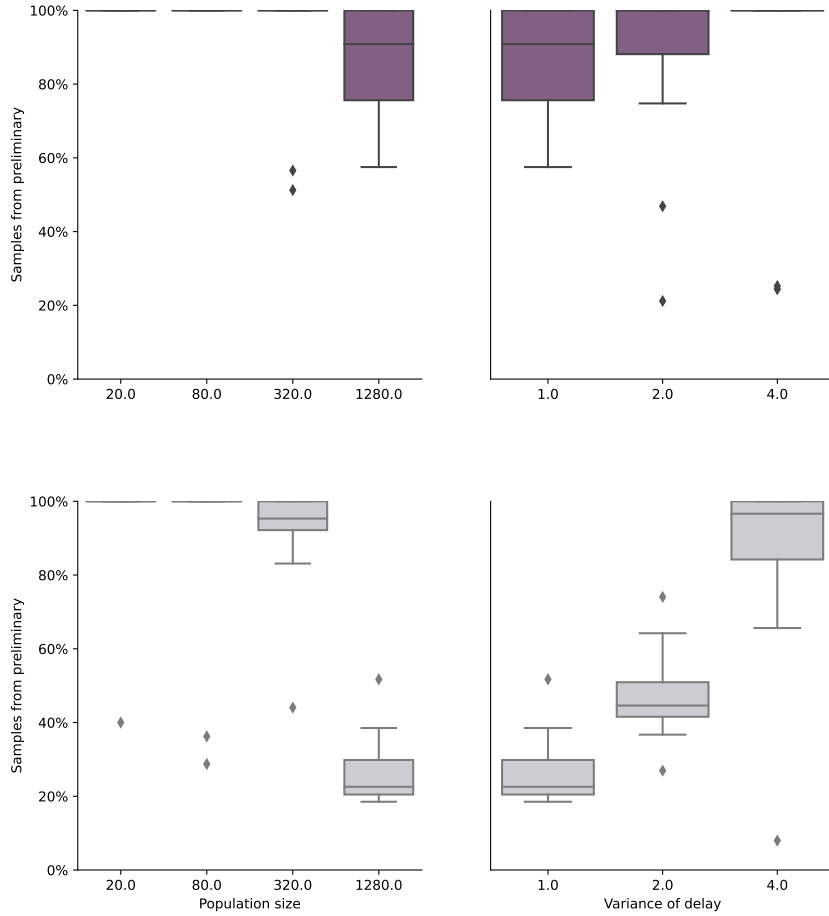


Figure S1: Fraction of particles sampled from the preliminary proposal in the last generation (for $\sigma^2 = 1$ (left) and $N = 1280$ (right)) for **LA Prel** (upper row) and **LA Past** (second row).

In Figure S1, one can also observe how the fraction of particles sampled from the preliminary decreases as the population size increases. This makes sense, since keeping the number of workers constant results in a more or less constant amount of resources used for the preliminary sampling between the N -th acceptance in a generation and the last worker to finish working on that generation. So, there should be a roughly constant number of preliminary acceptances opposite to an

increasing size of the total population.

Similarly, a higher run-time variance increases the fraction of preliminary particles, as this results in more time between the N -th acceptance and the last simulation of a generation.

Results – Run-Time

To analyze how much effect the new scheduling can have on the wall time, we ran the same model on different population size to worker ratios. First, we used an idle time with variance $\sigma^2 = 1$, the same population sizes as above, i.e. between 20 and 1280 and ran each of those on 1, 2, 4, and 8 nodes with 32 workers each, so on 32 to 256 workers. For a run-time variance of $\sigma^2 = 1$, we started with 13 repetitions of each scheduling approach (Except for STAT, where we used only 3 repetitions for time-preserving purposes) for each population size on 256 workers (see Figure 5). For $\sigma^2 = 2$, run-times were generally higher (see Figure S2).

We can observe a substantial speed-up when using LA instead of DYN scheduling (see Figure 5 and S2), whenever the population size is about as large or slightly larger than the amount of workers. In the best case, the LA approach decreased the wall time by nearly a factor of two when compared to the dynamic scheduling runs. On the other hand, the difference in efficiency is much less apparent, or even almost trivial, in case the two factors, population and worker size, are vastly different.

In the case where the amount of workers is substantially larger than our population size, only few simulations remain for each worker. This can go as far as even having enough preliminary acceptances to complete the next generation even before the previous one is finished leaving all further simulations retrospectively useless, which results in LA scheduling also having a poor parallel efficiency. Nonetheless, in that case there is still a substantial acceleration when compared to established approaches as LA can almost complete two generations in the time it takes to complete one using DYN.

When the population size is multiple times larger than the amount of available workers, dynamic scheduling already performed very well, leaving little room for improvement. However, even in that scenario we do not have any changes to the negative, so any additional computation necessary to enable the sampling from the preliminary population appears always worth the effort.

Further, using the same setup, we examined the effect of the run-time variance of the single evaluations on the speed-up. For that we ran the same tests as before with an idle time variance of $\sigma^2 = 2$; and indeed results (see Figure S2) seem to indicate that the achieved acceleration is even slightly higher than for the less varying run-time with $\sigma^2 = 1$.

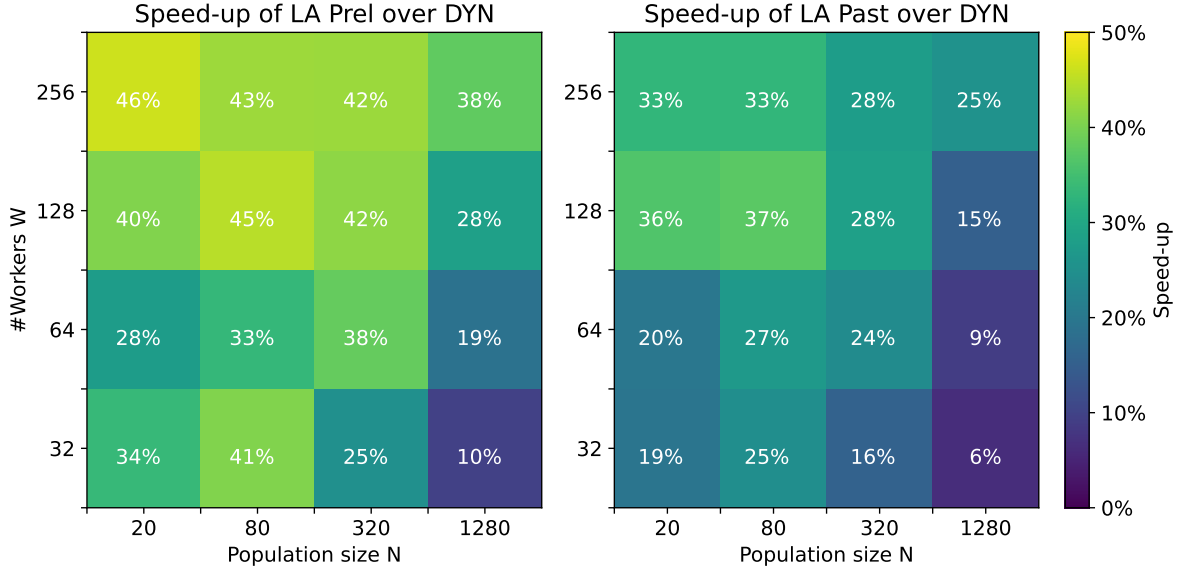


Figure S2: The speed-up for **LA Prel** (left) and **LA Past** (right) with $\sigma^2 = 2$

Finally, both DYN and LA scheduling scale better than STAT when using more computational resources (See Figure S3)

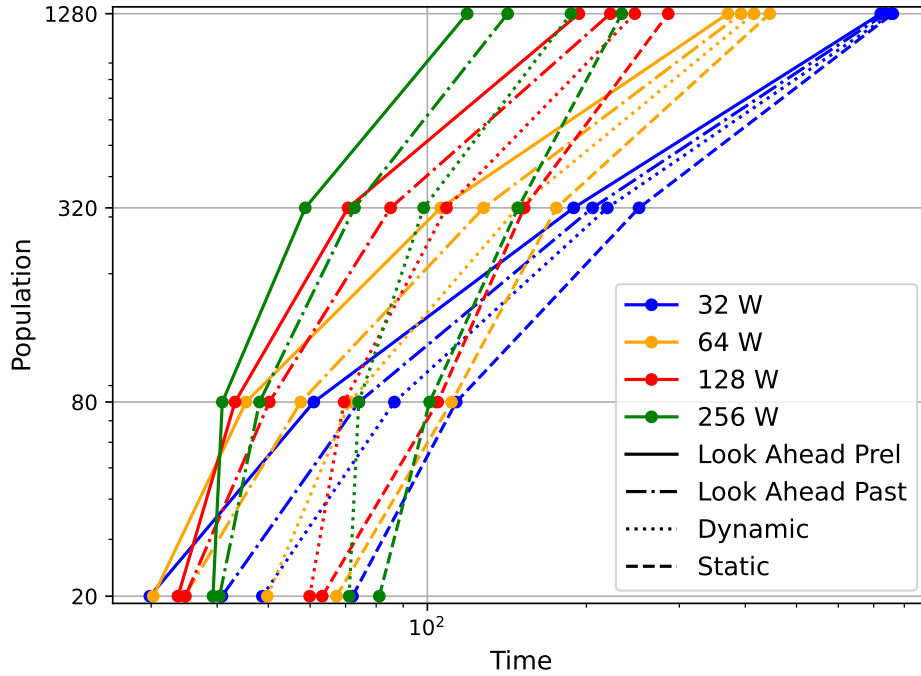
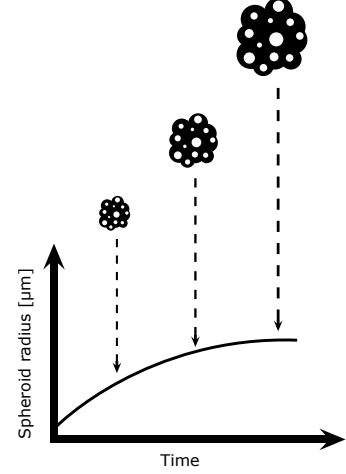


Figure S3: The time taken by different scheduling methods on different population sizes and numbers of workers.

3.3 (M1) Tumor Growth Model

The tumor growth model (M1) is our first test instance based on a real life example. The model was developed mainly in [2] and aims to describe the growth of a tumor spheroid while taking into account its spacial structure. For example, the proliferating cells are almost exclusively the ones in the outer rim of the spheroid whereas the ones contained in the core are mostly necrotic.

As hybrid discrete-continuous model it uses different mathematical tools trying to accurately depict a real life biological system.



An agent-based approach with stochastic interactions between separate particles in a system, is used to model the relations between the individual cells, while a system of PDEs describes the extracellular matrix. At the same time, mechanisms like cell division and cell death are modeled using a continuous time Markov process. For further details about the model see [2, 3] and for some biological background see e.g. [4, 5].

The version we used as test instance is a two dimensional implementation of the tumor growth model with seven parameters see Table 3. It is available in the python package *tumor2d* (<https://github.com/ICB-DCM/tumor2d>). Even using several hundred workers and a population size of e.g. 1000, the parameter inference for the tumor growth model takes hours to days, making it close to impossible to run the simulation on anything but large-scale parallelized infrastructure and strictly necessary to employ an efficient workload distribution scheme. Yet, the run-time is not yet as exorbitantly high as for some other models, such that it was possible to perform several runs to obtain at least some reliability in the results. This was particularly important as the trajectories of the tumor growth model underlie stochastic fluctuations, especially in the early phases, as there the number of cells is still very limited. Towards the end, the cell numbers are far higher and the fluctuations are much less severe due to the averaging effect of the Law of Large Numbers.

Results – Correctness

We ran model M1 with population sizes ranging from 250 to 1000, employing two different amounts of workers 128 and 256. The acceptance criteria for candidates of a generation are adaptively chosen after the previous one finishes as the 50% quantile of the previous accepted distances. The run finishes after a generation t in which the threshold ε_t falls below a certain value, in our runs $\varepsilon_{n_t} \leq 700$. Every time one scenario was run, one instance with dynamic scheduling, one with [LA Prel](#), and one with [LA Past](#) scheduling was performed with the same setup to directly compare the results.

Table 3: List of fitted parameters of model M1. All parameters have been transformed to a logarithmic scale with base 10.

Parameter	Description	Prior
$\log_{10}(k_{\max}^{\text{div}})$	Division rate	U(-3,-1)
$\log_{10}(L_{\text{div}})$	Division depth	U(-5,0)
$\log_{10}(L_{\text{init}})$	Initial spheroid radius	U(1,3)
$\log_{10}(q_{\text{init}})$	Initial quiescent cell fraction	U(0,1.2)
$\log_{10}(k_{\text{pro}}^e)$	ECM production rate	U(-5,0))
$\log_{10}(k_{\text{deg}}^e)$	ECM degradation rate	U(-5,0))
$\log_{10}(e_{\text{div}})$	ECM division threshold	U(-5,0))

During the runs, no substantial deviations in the quality of results were observed (see Figure S4 and Figure S5; detailed results of the other runs can be found in the supplementary code).

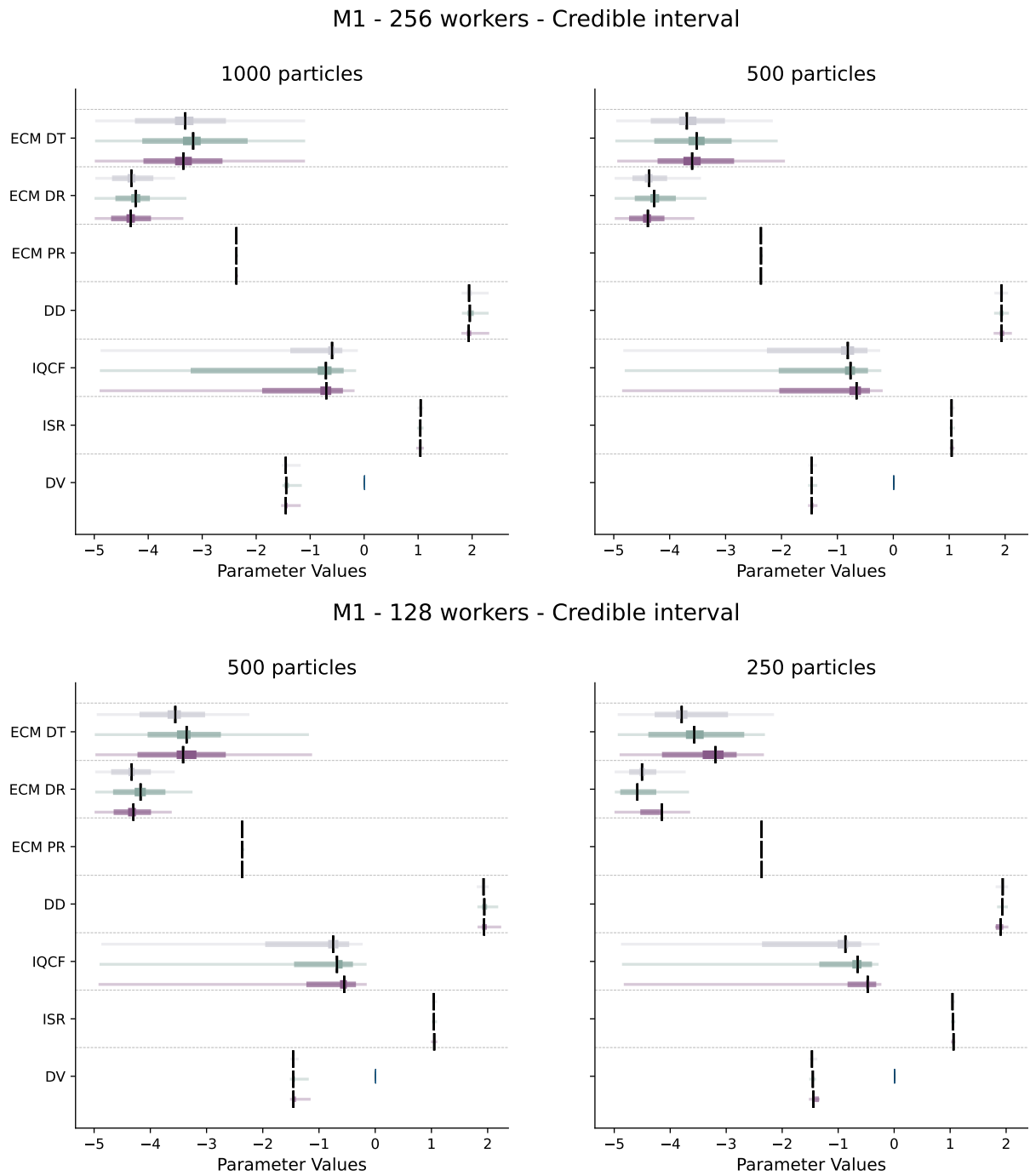


Figure S4: The credible interval of the model M1 with population size 1000, 500, 250 on 128 and 256 workers.

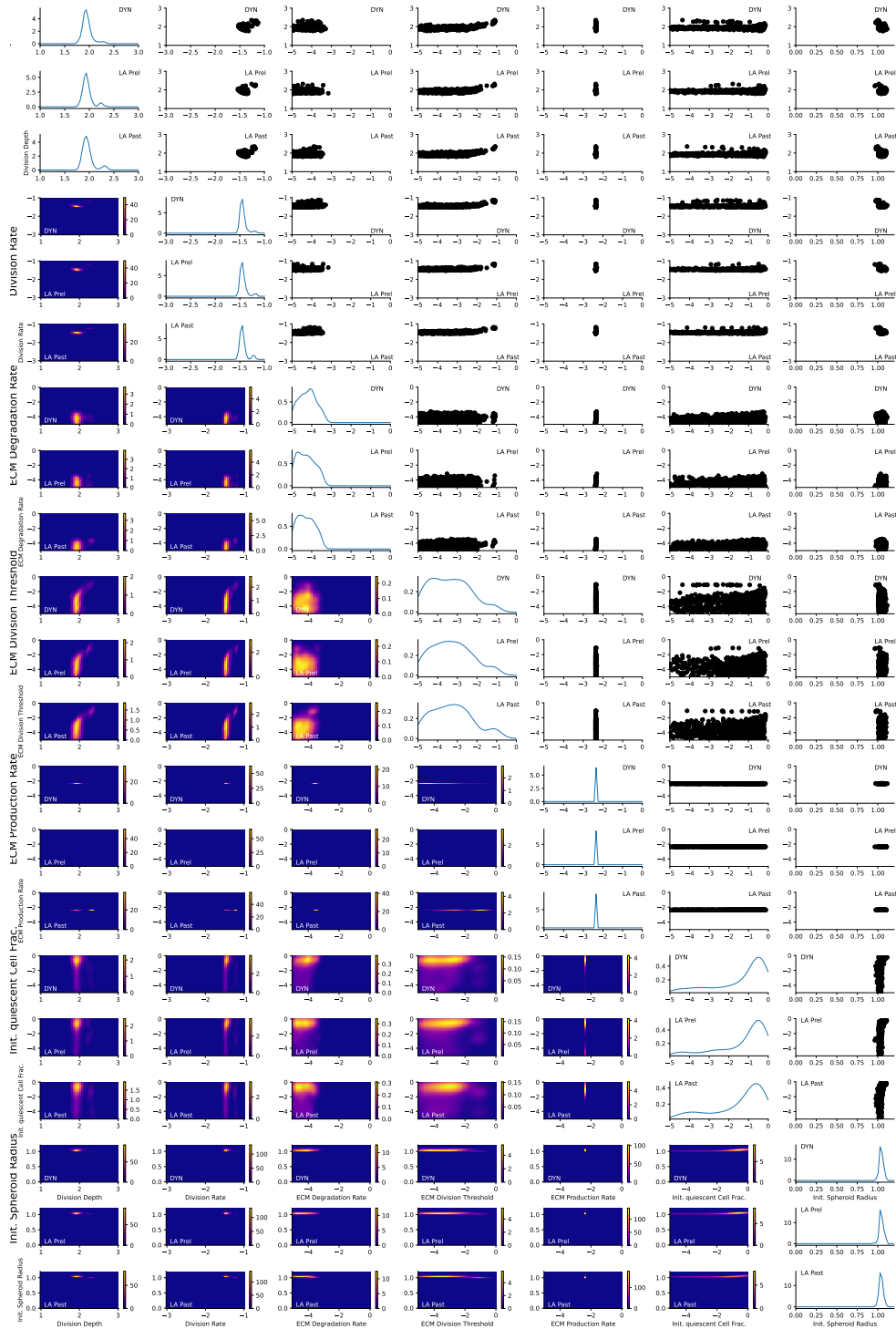


Figure S5: Direct comparison of the posterior distributions for all seven parameters of (M1) for a run with $N = 1000$ on $W = 256$ workers using DYN, LA Prel, and LA Past scheduling. The parameters with sharp peaks are visibly at the same location and if the inference returned a broader distribution, it did so in all cases.

While the fraction of preliminary particles tends to decrease over the course of the inference, it varies strongly from generation to generation (see Figure S6). As the time between the N -th acceptance in generation t and the last worker to finish is expected to be more or less constant, we have a similar amount of evaluations from the preliminary proposal in each generation. Generally, the acceptance rate decreases with the epsilon threshold however, and this also holds for the preliminary proposal based candidates. So, as more total evaluations are necessary to reach the desired number of accepted particles N , a smaller fraction of those should end up being sampled from the preliminary as the generations progress and the acceptance rate decreases.

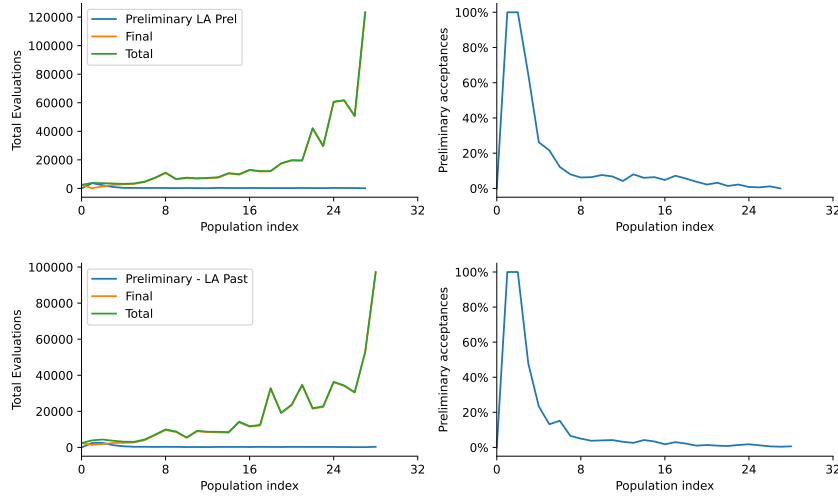


Figure S6: Total evaluations and fraction of accepted particles of M1 based on the preliminary population in each generation in the [LA Prel](#) (top) and [LA Past](#) (bottom) runs of the same run as in Figure S5

Results – Run-Time

For more complex models, it is usually far more effective to use the adaptive epsilon schedule mentioned in the previous section. That however, leads to a different final acceptance threshold in every run, which strongly affects the run-time. Together with the very stochastic start of the tumor growth model, this yields a high variance of the wall times, making a direct comparison more difficult. Nonetheless, we can for example observe the time point after which each epsilon value is achieved.

In the more extreme cases, it also occurred that the LA scheduling took slightly longer than the corresponding DYN run, but similarly also that the LA run decreased the wall time by a factor of more than 2 when compared to the DYN scheduling based one (see Figure S7).

These stronger differences in wall time usually seem to be traceable to one single generation taking vast amounts of time. Those long generations occur in all scheduling variants and exist most likely because the epsilon for that generation was chosen too optimistically.

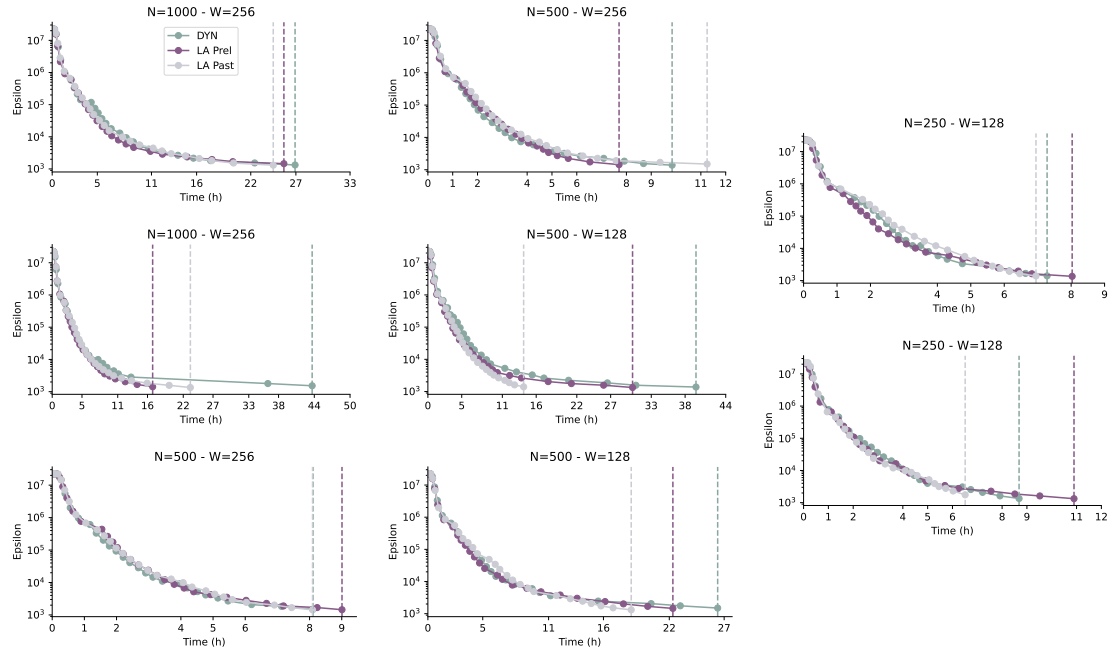


Figure S7: Development of the acceptance threshold over time for the different runs for model M1. These are all the 8 runs we performed.

On average, it seems that acceleration varies based on the population size and the number of workers to be used. Over the 8 times we have executed the inference of the tumor model with an adaptive epsilon schedule, we observed a mean acceleration of 21%, with the median value being 23% for [LA Past](#).

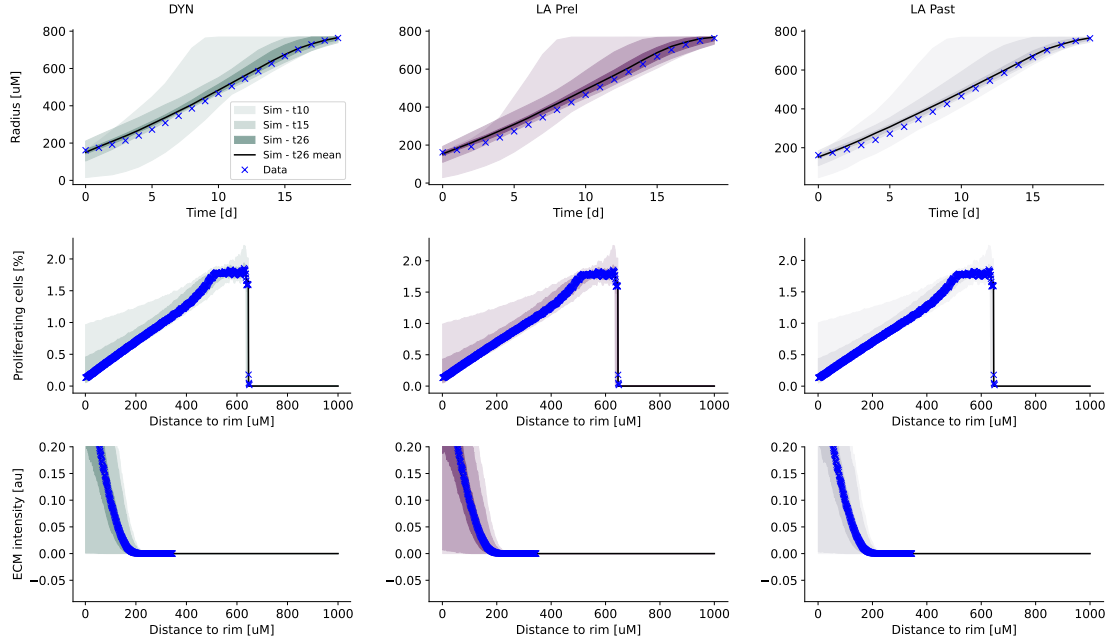


Figure S8: Comparison of best parameter fit for M1 using DYN (left), LA Prel (middle), and LA Past (right) for a population size of $N= 1000$

3.4 (M2) Liver regeneration model

M2 is a model of YAP regulation by mechanical stimulation through expansion of the bile canaliculi (BC) [6]. A single realisation of a model varies greatly based on parameter values ranging from 3 to 3,000 seconds. This model has 14 unknown parameters (see Table 4) and two observables, namely nuclear YAP and total YAP intensities which were quantified from image tiles covering an entire portal and central vein. The yes-associated protein (YAP) is an activator that activates the Hippo pathway that plays an important role in liver regeneration.

Two sub-models were used to describe the changes of osmotic pressure and the concomitant activation of YAP after PH. The first sub-model is a biophysics-based model to predict the local mechanical stress and apical membrane strain that result from the alteration of osmolyte (bile acid) load in the BC network after partial hepatectomy. It considers the spatial geometry of the BC within the portal and central vein axis of the lobule. Sub-model 2 is a biochemistry-based model that predicts the cellular response of YAP to the local mechanical stress.

Results – Correctness

Several fitting configurations were done to model (M2) with population sizes ranging from 250 to 1000, with two different amounts of workers, 128 and 256. The acceptance criteria for particles were adaptively set to be the 30% quantile of the previously accepted distances. The run set to be finished after the discrepancy threshold ε_t falls under some values. The value was set differently

Table 4: List of fitted parameters of model M2. All parameters have been transformed to a logarithmic scale with base 10.

Parameter	Description	Prior
$\log_{10}(k1)$	Max. flux of SENSOR activation	$U(1,3)$
$\log_{10}(k2)$	Max. flux of SENSOR inactivation	$U(0,2)$
$\log_{10}(k3)$	Factor of YAP synthesis rate	$U(-0.76,1.23)$
$\log_{10}(k4)$	YAP inactivation rate	$U(-1.95,0.04)$
$\log_{10}(k5)$	YAP activation rate	$U(1,4)$
$\log_{10}(k6)$	Inact. YAP binding rate to SF	$U(-1.74,0.25)$
$\log_{10}(k7)$	Inact. YAP unbin. rate from SF	$U(1,4)$
$\log_{10}(k8)$	YAP export rate from nucleus	$U(1,4)$
$\log_{10}(k9)$	YAP import rate into nucleus	$U(-1.76,0.23)$
$\log_{10}(k10)$	YAP degradation rate	$U(-0.95,1.04)$
$\log_{10}(k11)$	Inact. YAP degradation rate	$U(0.47,2.47)$
$\log_{10}(k12)$	M-M const. of SENSOR activation	$U(-4,-2)$
$\log_{10}(k13)$	M-M const. of SENSOR inactivation	$U(-1.6,0.40)$
$\log_{10}(k14)$	Intensity normalization total	$U(-1,1)$

based on different configurations ranging from $1.4e+03$ to $2.0e+03$. Similar to model (M1), the three different samplers, DYN, [LA Prel](#), and [LA Past](#), were executed on similar configurations to ensure a fair comparison.

Assessing the quality of the result, there were no substantial differences of the results during the run (see Figure S9 and S10 and Figure S11; detailed results of the other runs can be found in the supplementary code).

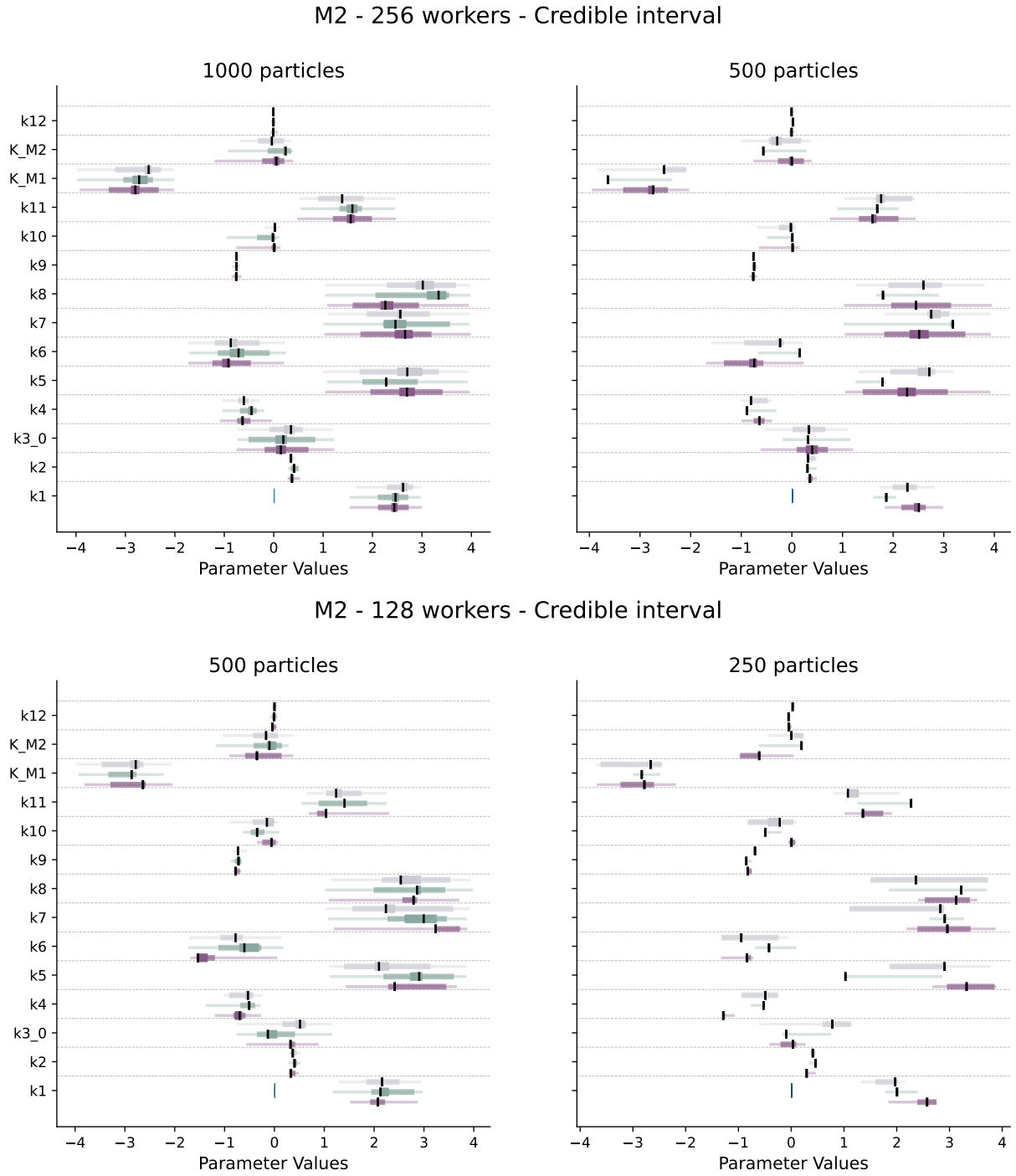


Figure S9: The credible interval of the model (M2) with population size 1000, 500, 250 on 128 and 256 workers. A similar maximum simulation time was used here as in Figure S14 (900 seconds as maximum simulation time)

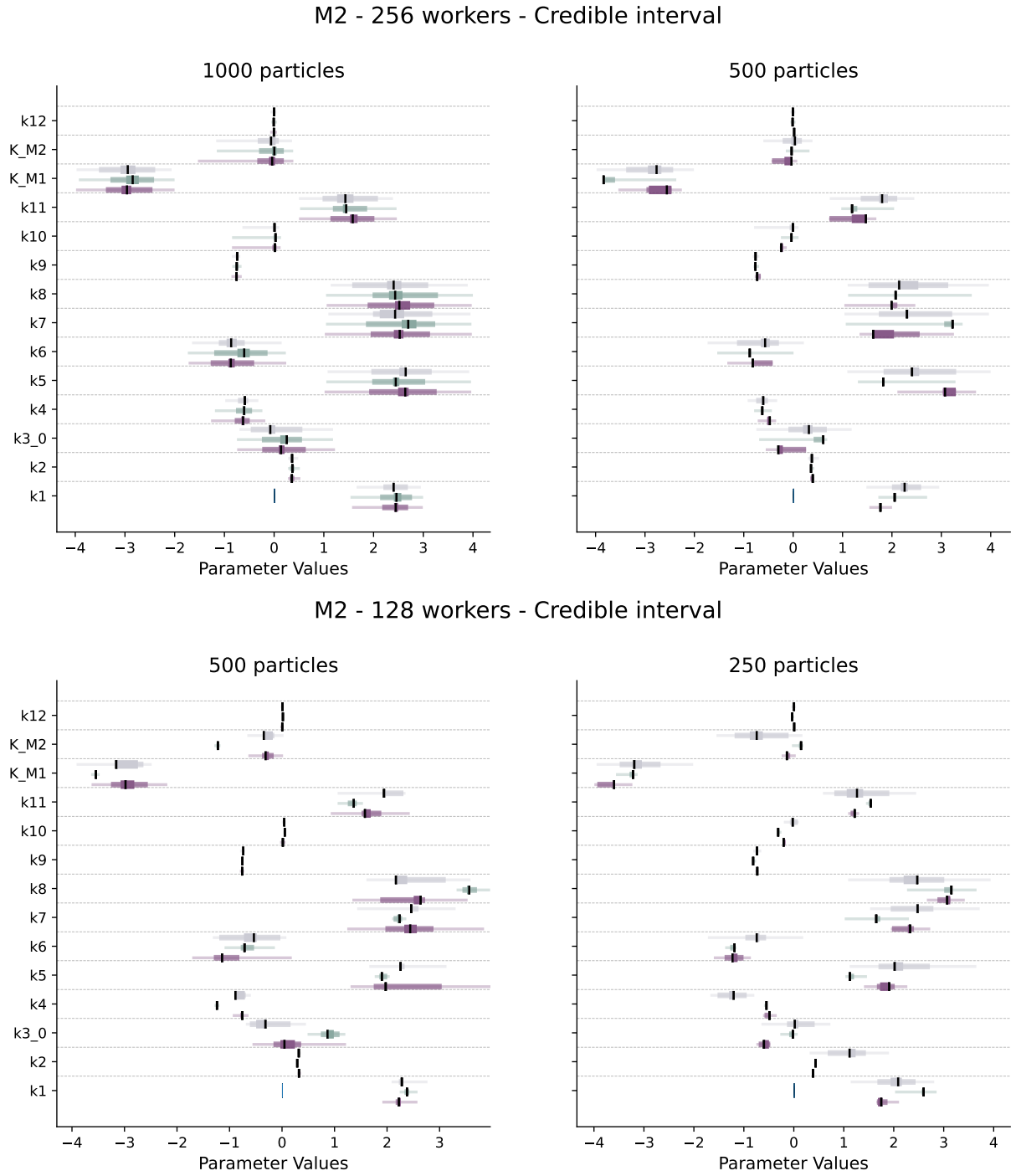


Figure S10: The credible interval of the model (M2) with population size 1000, 500, 250 on 128 and 256 workers. A maximum simulation time was used here as in Figure 6 (1800 seconds as maximum simulation time).

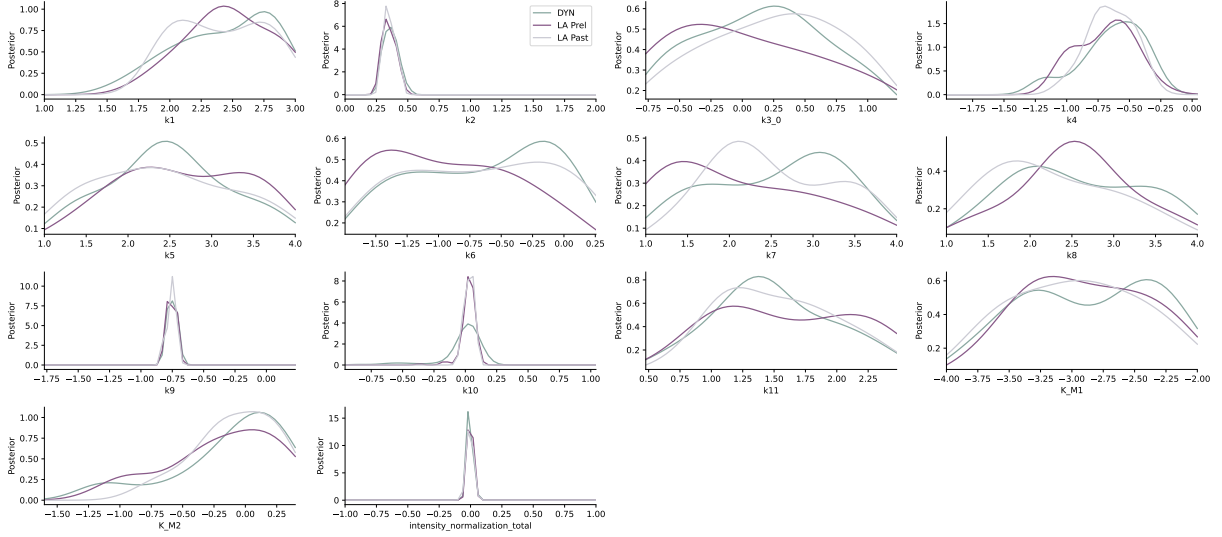


Figure S11: Direct comparison of the posterior distributions for all 14th parameters of (M2) for a run with $N = 1000$ on $W = 256$ workers using DYN, LA Prel, and LA Past scheduling. The sharp peaks of the parameters are visibly at the same location, and if the inference yielded a more broader distribution, it does. It did so in all cases..

Similar to model (M1), the fraction of M2 model preliminary particles decrease over the course of the inference, While the fraction of preliminary particles tends to decrease over the course of the inference (see Figure S12).

The fluctuations of this fraction exist because the acceptance threshold, and thus the acceptance rate, decreases less consistently than when using a static epsilon schedule, as can at least partially be observed in Figure S15.

As a large fraction of the population is based on the preliminary proposal for a substantial part of the generations, it shows that these preliminaries have a large effect on the LA version of the ABC-SMC algorithm. Nonetheless, the posteriors are consistently similar, indicating that everything works as intended and no bias was introduced.

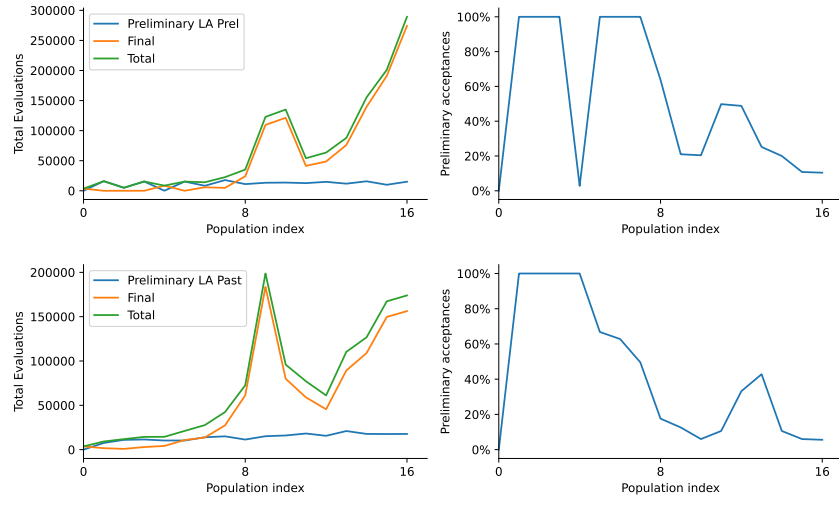


Figure S12: Total evaluations and fraction of accepted particles for (M2) based on the preliminary population in each generation in the **LA Prel** (top) and **LA Past** (bottom) runs of the same run as in Figure S11

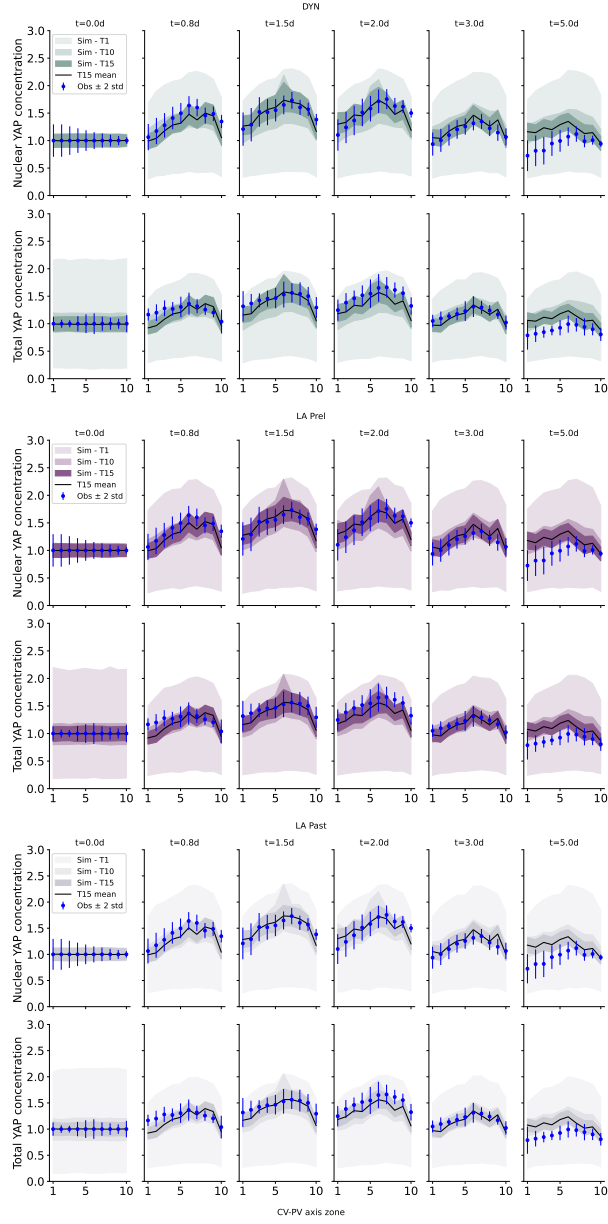


Figure S13: Comparison of best parameter fit for (M2) using DYN (top), **LA Prel** (middle), and **LA Past** (bottom) for a population size of $N= 1000$

Results – Run-Time

To save computational resources, we employed an early rejection strategy to reject particles based on a maximum run-time for individual simulations not 386 matching the data. However, this will decrease run-time heterogeneity. To assess the affect of that on the speed up of the look-ahead sampler, we fit the model using two different maximum run-time, 15, and 30 minutes. The result of the 30min maximum run-time can be seen in Figure 6. In addition, the 15min maximum run-time

is presented in figure S14. Result shows clearly that if heterogeneity increases, one would expect higher speed-up from both [LA Prel](#) and [LA Past](#) compared to DYN sampler.

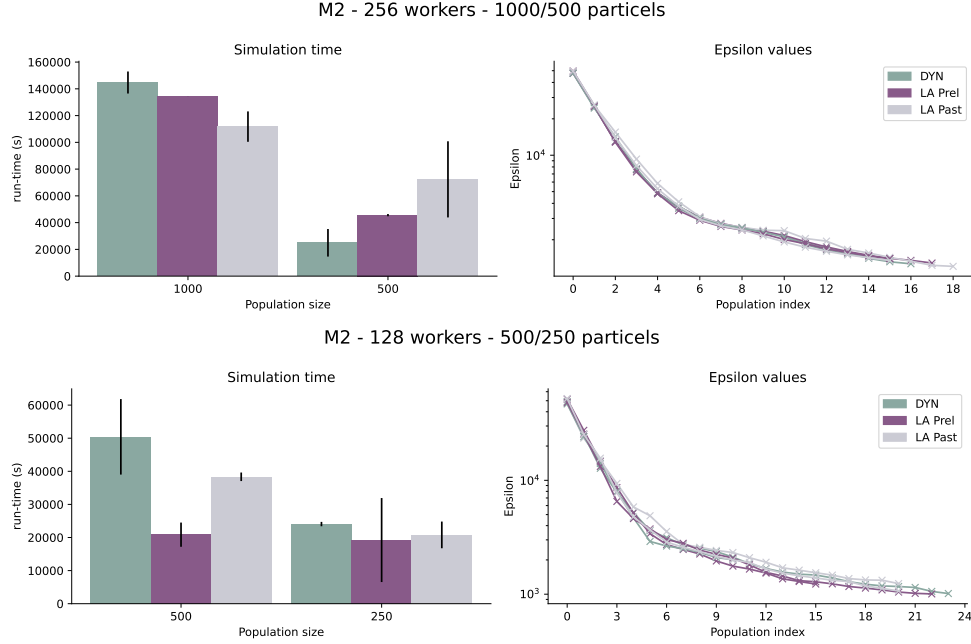


Figure S14: The run-time and posterior distributions for 2 different runs of the model M2 with population size 1000, 500, 250 on 128 and 256 workers. In this run, we use a maximum simulation time of 900 seconds, as opposed to 1900 seconds in Figure 5. We can see that decrease in heterogeneity level on simulation time drastically affected the effectiveness of the LA samplers.

Figure S15 shows the development of the epsilon threshold for the 8 different runs. In the more extreme cases, it also occurred that the LA scheduling took slightly longer than the corresponding DYN run, which was observed more in the less heterogeneous runs (e.g. when using 30min as maximum run-time).

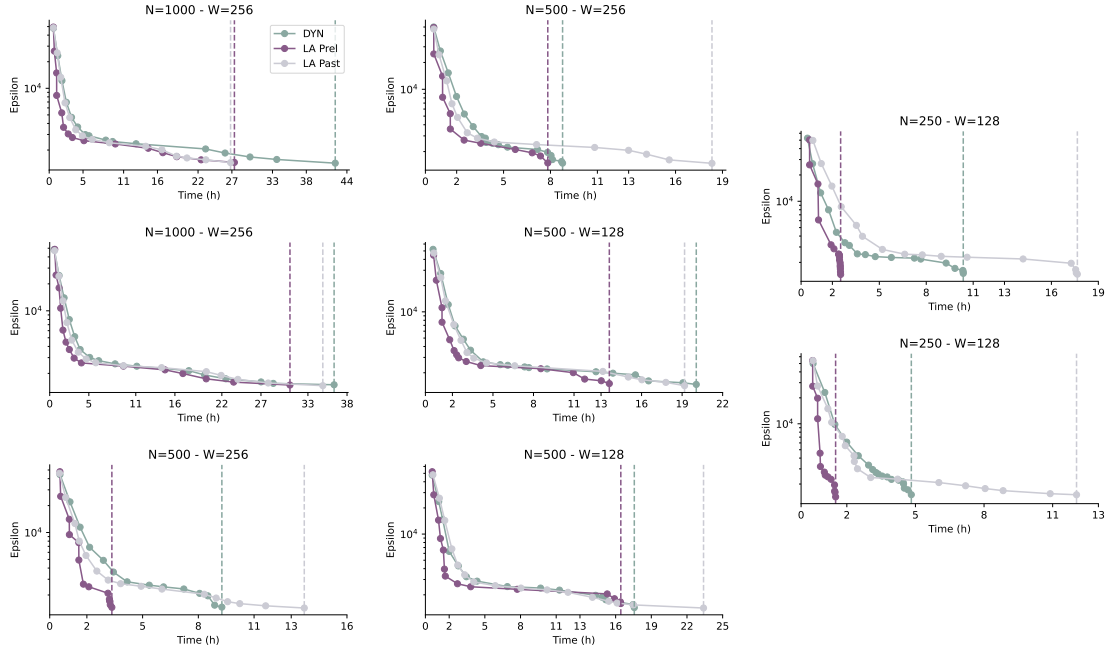


Figure S15: Development of the acceptance threshold over time for the different runs for model M2. These are all the 8 runs we performed.

On average, it seems that acceleration varies based on the population size and the number of workers to be used. Over the 8 times we have executed the inference of the liver regeneration model with an adaptive epsilon schedule, we observed a mean acceleration of 36%, with the median value being 31% for the [LA Prel](#).

3.4.1 Simulation time variability

The simulation time for a model often relies on the parameter space, as exemplified by model M2. The forward simulation of this model is heavily influenced by the chosen set of parameters. For instance, parameters such as the inactivation rate of YAP (k_4) or the activation rate of YAP (k_5) both greatly affect the computation time, as demonstrated in Figure S16.

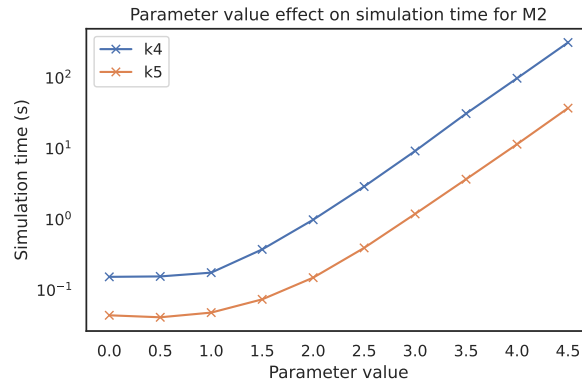


Figure S16: Simulating the liver regeneration model M2 using different values for the parameters k_4 and k_5 that range from 0 to 4.5 in $\log_{10}$ space

References

- [1] J. Starruß, W. de Back, L. Brusch, and A. Deutsch. Morpheus: A user-friendly modeling environment for multiscale and multicellular systems biology. *Bioinf.*, 30(9):1331–1332, Jan. 2014.
- [2] N. Jagiella. *Parameterization of lattice-based tumor models from data*. Ph.d. thesis, Université Pierre et Marie Curie, Paris, France, 2012.
- [3] N. Jagiella, D. Rickert, F. J. Theis, and J. Hasenauer. Parallelization and high-performance computing enables automated statistical inference of multi-scale models. *Cell Syst.*, 4(2):194–206, 02 2017.
- [4] K. Carver, X. Ming, and R. L. Juliano. Multicellular tumor spheroids as a model for assessing delivery of oligonucleotides in three dimensions. *Molecular Therapy – Nucleic Acids*, 3(e153), Mar. 2014.
- [5] K. Kwapiszewska, A. Michalczuk, M. Rybka, R. Kwapiszewski, and Z. Brzózka Z. A microfluidic-based platform for tumour spheroid culture, monitoring and drug screening. *Lab Chip*, 14(12):2096–2104, June 2014.
- [6] Kirstin Meyer, Hernan Morales-Navarrete, Sarah Seifert, Michaela Wilsch-Braeuninger, Uta Dahmen, Elly M Tanaka, Lutz Brusch, Yannis Kalaidzidis, and Marino Zerial. Bile canaliculi remodeling activates yap via the actin cytoskeleton during liver regeneration. *Mol. Syst. Biol.*, 16(2):e8985, 2020.

7.3 Simulation-Based Inference of Cell Migration Dynamics in Complex Spatial Environments

This preprint of this paper is reprinted as part of this thesis. The reference for the original preprint is as follows:

J. Arruda et al., *Simulation-Based Inference of Cell Migration Dynamics in Complex Spatial Environments*, bioRxiv (2025), eprint: <https://www.biorxiv.org/content/early/2025/09/25/2025.09.25.678460.full.pdf>, URL: <https://doi.org/10.1101/2025.09.25.678460>

Simulation-Based Inference of Cell Migration Dynamics in Complex Spatial Environments

Jonas Arruda^{1,2}, Emad Alamoudi^{1,2}, Robert Mueller³, Marc Vaisband^{1,2}, Ronja Molkenbur⁴, Jack Merrin⁵, Eva Kiermaier^{4,6,7}, and Jan Hasenauer^{*1,2}

¹Life & Medical Sciences Institute, University of Bonn, Bonn, Germany

²Bonn Center for Mathematical Life Sciences, University of Bonn, Bonn, Germany

³Center for Interdisciplinary Digital Sciences, Department Information Services and High Performance Computing, TUD Dresden University of Technology, Dresden, Germany

⁴Life & Medical Sciences Institute, Immune and Tumor Biology, University of Bonn, Bonn, Germany

⁵Institute of Science and Technology Austria, Klosterneuburg, Austria

⁶Department of Medicine 1/CITABLE, Friedrich-Alexander University Erlangen-Nürnberg, Erlangen, Germany

⁷Deutsches Zentrum Immuntherapie (DZI), Universitätsklinikum Erlangen, Erlangen, Germany

September 25, 2025

Abstract

To assess cell migration in complex spatial environments, microfabricated chips, such as mazes and pillar forests, are routinely used to impose spatial and mechanical constraints, and cell trajectories are followed within these structures by advanced imaging techniques. In systems mechanobiology, computational models serve as essential tools to uncover how physical geometry influences intracellular dynamics; however, decoding such complex behaviors requires advanced inference techniques. Here, we integrated experimental observations of dendritic cell migration in a geometrically constrained microenvironment into a Cellular Potts model. We demonstrated that these spatial constraints modulate the motility dynamics, including speed and directional changes. We show that classical summary statistics, such as mean squared displacement and turning angle distributions, can resolve key mechanistic features but fail to extract richer spatiotemporal patterns, limiting accurate parameter inference. To solve this, we applied neural posterior estimation with in-the-loop learning of summary features. This learned summary representation of the data enables robust and flexible parameter inference, providing a data-driven framework for model calibration and advancing quantitative analysis of cell migration in structured microenvironments.

*Corresponding author: jan.hasenauer@uni-bonn.de

37 Introduction

38 Tissues are complex, highly organized systems comprising cells and extracellular
 39 components that exhibit dynamic behavior and structural heterogeneity across
 40 developmental stages. This spatiotemporal complexity, encompassing both biochem-
 41 ical gradients and mechanical cues, critically influences cell behavior, particularly
 42 migration dynamics within structured or constrained environments [1, 2]. Den-
 43 dritic cells (DCs) belong to the group of innate immune cells, which activate T
 44 cells via antigen presentation [3]. To efficiently reach their final destination within
 45 the secondary lymphoid organs, DCs follow chemical guidance cues presented as
 46 chemokine gradients across the lymphatic vasculature [4]. Directional cell migration
 47 in response to chemotactic cues is a cornerstone of many (patho)-physiological
 48 processes, including immune surveillance and response [5], inflammation [6], and
 49 cancer metastasis [7, 8, 9]. This guided movement, termed chemotaxis in response to
 50 soluble attractants or haptotaxis when driven by immobilized chemokine gradients,
 51 involves the binding of chemokines to receptors on the cell surface, which activates
 52 intracellular signaling pathways that coordinate directed migration by integrating
 53 chemotactic sensitivity, migration persistence, cell morphological dynamics, and the
 54 traction forces modulated by the mechanical and chemical microenvironment [10].

55 Advances in live cell imaging have facilitated the acquisition of extensive quantita-
 56 tive data to capture the dynamic behavior of migrating cells [11]. The interpretation
 57 of these data to elucidate pattern formation and tissue dynamics necessitates ad-
 58 vanced computational modeling due to the complexity of the data. Accordingly,
 59 computational models have become valuable tools for capturing the multiscale
 60 interactions and emergent behaviors that characterize biological systems. Model-
 61 ing approaches, including discrete and continuous models in both temporal and
 62 spatial domains, as well as hybrid frameworks, have been employed to capture cell
 63 behavior and cell-cell interactions within complex environments [12, 13, 14, 15, 16].
 64 Specifically, Cellular Potts models [17] and vertex models, which are continuous
 65 in time and discrete in space, are extensively utilized to simulate cell interactions,
 66 with numerous software platforms available for implementing and testing these
 67 models [18, 19, 20, 21, 22, 23]. Various studies have leveraged simulations to unravel
 68 the dynamics of chemotaxis and cell migration, emphasizing the pivotal role of
 69 computational models in understanding cellular behavior [24, 25, 26].

70 A central challenge in applying these models is their calibration using experi-
 71 mental data. Typically, model parameters are not directly measurable, yet they
 72 are crucial for elucidating biological mechanisms. Parameter estimation enables
 73 quantitative modeling [27, 28], hypothesis testing [29], and prediction of dynamic
 74 responses to perturbations [30, 31]. The application of parameter estimation meth-
 75 ods to cell migration processes has been previously explored to infer parameters
 76 governing cell motility and intercellular forces [32, 33]. In particular, parameter
 77 inference methods, such as approximate Bayesian computation (ABC), facilitate
 78 model calibration based on simulations without the need to derive a likelihood,
 79 which can be prohibitively complex for Cellular Potts models [34, 35, 36].

80 State-of-the-art ABC pipelines typically rely on summary statistics to reduce the
 81 dimensionality of the data. These statistics, such as the mean squared displacement
 82 or turning angle distributions of cell movement, are often hand-crafted based on

domain knowledge. However, they may fail to provide relevant information for precise parameter estimation. To address these challenges in the context of ABC, recent approaches have included adaptive algorithms that weight hand-crafted summaries to maximize the posterior information gained from a dataset or to learn summaries on pairs of parameters and simulations [37, 38, 39]. However, hand-crafted summaries or summaries based on the posterior mean may not be sufficient to reconstruct the posterior shape [40, 39].

Alternatives to ABC, such as neural posterior estimation (NPE) methods, employ neural networks to learn mappings from data to model parameters, effectively handling high-dimensional data and sophisticated models and circumventing the limitations of conventional ABC methods [41]. By training neural networks, relevant features can be extracted automatically from the data, neural posterior estimation can improve parameter estimation accuracy [42].

In this study, we aimed to improve the inference of spatial cell movement models in complex environments by applying both classical and machine learning simulation-based inference methods. A comprehensive inference pipeline was implemented to evaluate the different inference approaches. As an important example of the interplay between spatial structure and cell migration, we focused on dendritic cell movement in complex environments. Therefore, we designed microstructured polydimethylsiloxane chips that impose obstacles on migrating dendritic cells and recorded their trajectory data. To analyze the data, a Cellular Potts model of migration in a spatially complex environment was constructed, and approximate Bayesian computation with different summary statistics was compared with neural posterior estimation using simulated and experimental data. We demonstrated the ability of the model to capture the modulation of speed and direction through obstacles and chemokine gradients; hence, we present a data-driven framework to analyze how environmental mechanics and structure influence cell behavior, thereby advancing computational approaches in systems mechanobiology.

Results

To quantitatively dissect how chemokine gradients and physical constraints influence dendritic cell migration, we designed a structured *in vitro* assay and built a mechanistic model *in silico* based on the Cellular Potts framework. Using synthetic data with known ground truth, we systematically evaluated four simulation-based inference strategies: (i) approximate Bayesian computation (ABC) with hand-crafted summary statistics (displacement, velocity, turning angle, and angle degree); (ii) ABC employing a neural network as a feature extractor trained either to predict the posterior mean (ABC-PM) or (iii) with summaries tailored for inference (ABC-NPE); and (iv) neural posterior estimation (NPE) with jointly trained summary networks (as described in the Methods). After identifying NPE as the most flexible approach, we applied this strategy to calibrate our model on experimentally tracked dendritic cell trajectories under chemotactic stimulation within a microstructured polydimethylsiloxane (PDMS) pillar array. The resulting model faithfully reproduced the observed migration patterns and enabled realistic simulations under varying environmental conditions, offering mechanistic insights into how chemical signals and spatial barriers jointly govern immune cell navigation.

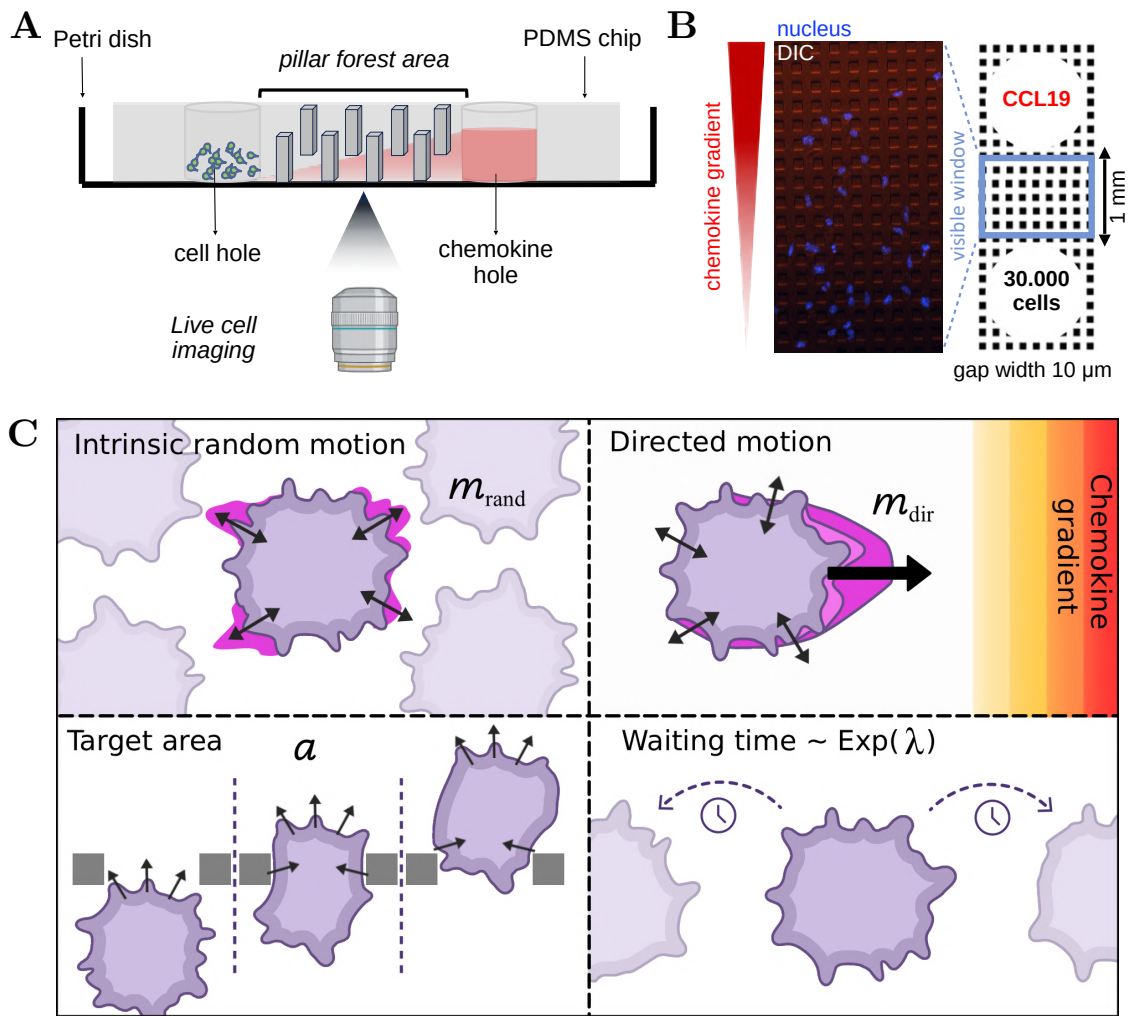


Figure 1: *Schematic depiction of the experimental and modeling setup.* (A) Dendritic cells are placed in an entry hole on the left and follow a chemokine gradient to the right through the pillar forest on a microstructured polydimethylsiloxane (PDMS) chip. (B) Top-down differential interference contrast (DIC) microscopy view within the visible window of the pillar forest, where cell movement can be tracked. Dextran was added to the CCL19 solution to visualize the chemokine gradient. (C) Visualization of a cell in the Cellular Potts model and the four model parameters to be estimated: the intrinsic random motion strength m_{rand} (upper left tile), the chemokine attraction strength m_{dir} (upper right tile), the target area of a cell a (lower left tile), and the rate λ until the cell changes its intrinsic direction (lower right tile).

128 Dendritic cell migration in pillar forests

129 Dendritic cells must efficiently migrate through complex tissue environments to
 130 reach lymphoid organs, guided by chemokine gradients. Understanding how they
 131 integrate chemical signals with physical constraints is critical for deciphering the
 132 mechanisms of the immune system.

133 To study this interplay, we exposed bone marrow-derived dendritic cells to a
 134 stable CCL19 gradient and confined them within a microfluidic chip containing a

pillar forest with 10 μm gaps (Figure 1A and [43]). Time-lapse videomicroscopy was used to track the nuclear positions at 30-second intervals in the visible window (Figure 1B), yielding trajectories of variable length depending on how long a cell remained within the field of view (further experimental details in Methods).

From these trajectories, we extracted descriptive measures of migration that captured complementary aspects of motility (see Methods). Displacement quantifies overall progress across the track, velocity reflects the average speed of migration, and turning angle and angle degree characterize directional biases toward the chemokine gradient or deflections caused by obstacles.

These summary statistics form the basis for quantitative comparison with a mechanistic model, linking experimental single-cell measurements to simulations and providing insight into how dendritic cells balance chemokine guidance with geometric restrictions imposed by tissue-like barriers.

Modeling cell migration dynamics in complex spatial environments

To analyze dendritic cell migration, we developed a 2D Cellular Potts model with chemotactic signaling and persistent random motion embedded in a simulation-based inference pipeline. In this lattice-based stochastic framework, cells are represented as extended domains of lattice sites, allowing the simulation of realistic cell shapes and their deformations. Cell behavior emerges from an energy minimization principle that combines terms for volume constraints with responses to external cues, such as chemokine gradients, allowing us to generate trajectories under conditions that mimic the experimental setup (Figure 1C).

The model possesses four parameters with a substantial influence on the migration patterns: the chemokine attraction strength m_{dir} , which quantifies the sensitivity of cells to the chemokine gradient modulating directed movement; the intrinsic random motion strength m_{rand} , which governs the bias toward previously randomly chosen movement directions; the rate λ for the waiting time distribution between directional changes of the intrinsic random motion; and the target area a , which regulates cell size via area constraints (for details, see Methods).

To estimate these parameters from data, we established a simulation-based inference pipeline. This pipeline supports (i) approximate Bayesian computation (ABC) with hand-crafted summary statistics, (ii) ABC with neural feature extractors trained to predict posterior means (ABC-PM), (iii) ABC with inference-tailored summaries (ABC-NPE), and (iv) neural posterior estimation (NPE) using jointly trained summary networks (see Methods). For NPE, we trained a summary network jointly with an inference network on a training dataset of 32,000 simulation-parameter pairs using an additional 100 simulations for validation. The inference network was implemented as a conditional normalizing flow optimized by minimizing the Kullback–Leibler divergence. The learned summary network is also used in ABC-NPE and can alternatively be adapted to directly predict the posterior mean by optimizing a mean-squared error objective without the inference network.

This pipeline enables inference of Cellular Potts model parameters from dendritic cell trajectories, linking experimental migration data to the underlying stochastic processes shaped by chemokine gradients and physical barriers.

Table 1: *Comparison of the different inference methods on simulation budget and performance metrics.* ABC-based methods were evaluated on three representative simulated datasets, whereas NPE was validated on 100 datasets. We computed the normalized root mean squared error (NRMSE). The training time accounts for both training set generation and neural network training and was measured on a high-performance computing cluster (for details, see Methods).

Method	# Simulations	Training	Inference	# Generations	NRMSE
ABC	66,000–413,000	–	2.1–19.2 h	12–16	0.23
ABC-PM	59,000–432,000	1.5 h	1.8–12.2 h	11–16	0.48
ABC-NPE	120,000–428,000	4.0 h	3.7–13.4 h	13–16	0.23
NPE	32,100	4.0 h	1 s	–	0.20

Inference-tailored summaries enable accurate inference on simulated cell movement in complex environments

To infer the parameters of the cell migration model, we established a parameter estimation pipeline. To ensure reproducibility, we conducted experiments using three synthetic datasets generated by the simulation model. For each dataset, we attempted to recover the underlying ground truth parameters, which were sampled from a prior distribution.

The ABC methods were run until either a minimum acceptance rate of 0.01 was reached or a maximum of 15 sequential generations was completed, including a pre-calibration round. In total, this required up to 432,000 simulations and runtimes between 2 and 20 h (Table 1). The assessment of the inference results revealed that all ABC approaches converged, as indicated by the low acceptance thresholds in the last generation (Supplementary Figure 1–3). For NPE, the inference procedure is amortized, meaning that once the networks are trained, the parameters can be inferred for any new dataset without retraining. To assess the convergence and calibration of the posterior in this setting, we applied simulation-based calibration on a larger validation dataset of 100 simulations, which confirmed good calibration and showed no visible bias (Supplementary Figure 4). Such rigorous validation was not feasible for the ABC methods due to their prohibitive inference time.

In terms of accuracy, measured via the normalized root mean squared error (NRMSE), NPE yielded the smallest error, closely followed by ABC and ABC-NPE (Table 1). Interestingly, the neural network trained on the posterior mean achieved a smaller NRMSE of 0.17 when directly predicting the parameters on the validation data. However, when used as a feature extractor in the ABC-PM, the error was much higher, with an NRMSE of 0.48 on the three synthetic datasets. In contrast, NPE achieved better performance using far less simulations with a training duration of approximately 4 h (Table 1). Moreover, inference with NPE is essentially instantaneous because the networks are trained once and can then be readily applied to new data, enabling the efficient processing of large numbers of datasets and allowing us to analyze parameter identifiability. For instance, the identifiability assessment for the rate parameter λ of the waiting time revealed an overall lower posterior contraction (Supplementary Figure 5) and a diminished

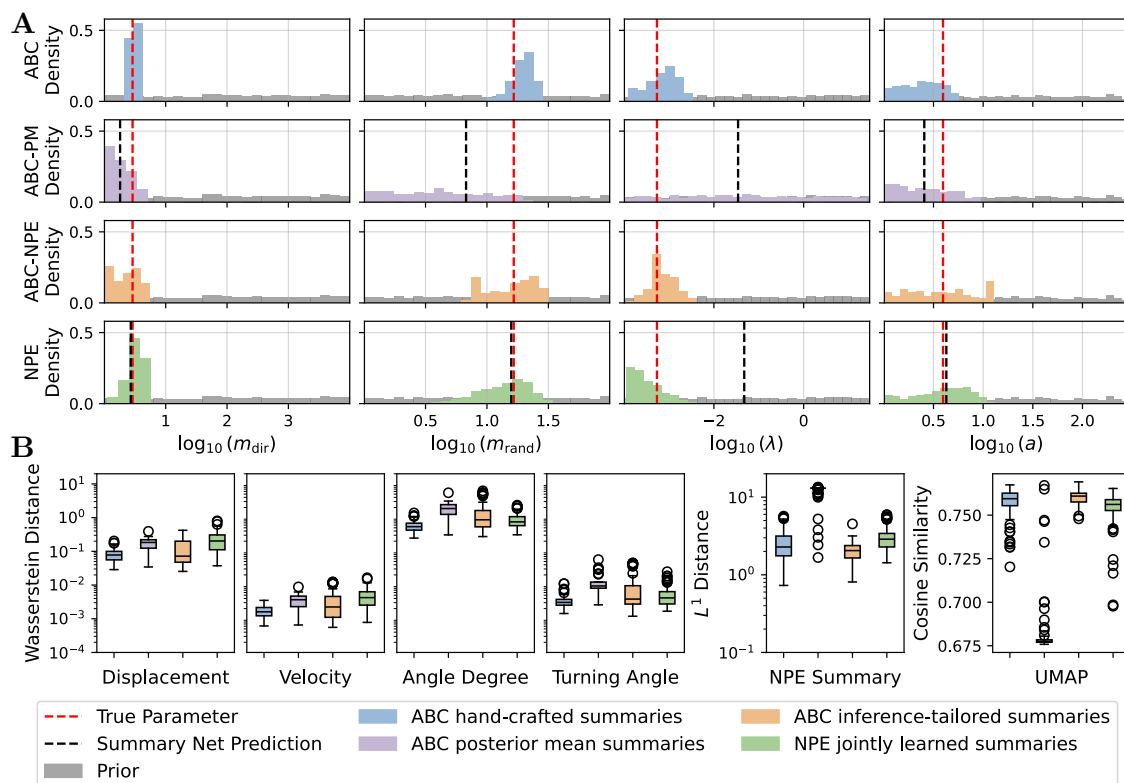


Figure 2: *Posterior approximations and summary statistics on synthetic data.* (A) Posterior approximations using the different ABC approaches and NPE with jointly trained summaries and predictions from summary latent spaces. (B) The distance between hand-crafted summaries and the statistics of the synthetic test set was quantified using the Wasserstein distance, the L^1 distance computed in NPE summary space, and cosine similarity in UMAP representation. All metrics were based on 100 simulation samples drawn from each posterior. Distributions are shown as box plots, indicating the medians, quartiles, and outliers.

parameter recovery accuracy with NPE on the validation data (Supplementary Figure 4). This lower identifiability may be attributed to the temporal resolution of the observations (30s between each image), which limits the amount of informative signal available for estimating this parameter.

Our results suggest that the ABC method, which relies on the posterior mean, exhibits a systematic bias in one of the parameters: the true value of the intrinsic random motion strength m_{rand} frequently lies outside the high-density region of the inferred posterior distribution (Supplementary Figure 5), despite the summary network itself showing no apparent bias during validation (Supplementary Figure 4). In contrast, the use of hand-crafted or inference-tailored summaries led to improved parameter recovery in ABC (Table 1).

An exploratory analysis of latent summary representations reveals that one advantage of posterior mean summaries lies in their interpretability: each component corresponds directly to a model parameter, rendering the representation transparent and easily analyzable, whereas the latent summary representations learned jointly with the normalizing flow are not inherently interpretable. However, the learned summaries in NPE were correlated with specific model parameters, indicating a

degree of functional disentanglement in the learned representation (Supplementary Figure 6). Moreover, a simple regression of the latent dimension onto the parameters already allows the prediction of the true parameters (Figure 2A).

To better understand these findings, we examined the informativeness of the summary statistics across methods. Specifically, we computed the hand-crafted summaries for simulations drawn from all four inferred posteriors. If a cell was not observed in the simulation because it moved outside the visible window, it was not included in the corresponding statistics. When using the metric employed by ABC, the Wasserstein distance, simulations from the posteriors and synthetic test data were similar across all approaches, despite their distinct posterior distributions (Figure 2B). However, alternative distance metrics, such as the L^1 distance in the NPE summary space and cosine similarity on a UMAP projection, reveal notable deviations, particularly between ABC with posterior mean summaries and the other methods. This suggests that the hand-crafted summaries may be insufficiently expressive to capture the full range of informative variations of the stochastic model.

In summary, although hand-crafted and posterior mean summaries remain interpretable, they can fail to capture the variability required for reliable parameter recovery. In contrast, inference-tailored summaries adapt their representation to the data and inference task, leading to improved parameter recovery. By optimizing the summary space directly for parameter estimation, this provides a flexible and data-driven approach that enables more accurate and robust inference.

Robust model calibration and validation of cellular dynamics on empirical data

Given the validated inference pipeline, we investigated the interplay between chemokine signaling and physical constraints in dendritic cell migration using experimental data (Figure 3A). The dataset contained 143 individual cell trajectories of varying lengths (18 to 120 frames per cell, with a mean duration of 33.2 min), reflecting heterogeneity in tracking durations across the experiment.

ABC was performed using both hand-crafted and inference-tailored summary statistics. Both approaches converged after 13 generations, as indicated by the small final acceptance thresholds, requiring approximately 200,000–300,000 simulations and 27–50 h of runtime (Supplementary Figure 7). In addition, NPE was employed, including an ensemble of three independently trained networks (NPE-E), to obtain a more robust estimate that is less sensitive to variations in individual network training.

To assess the differences between the approaches, we compared the hand-crafted summary statistics on posterior simulations, the L^1 distance in the NPE summary space, and the cosine similarity in the UMAP projection. Among the hand-crafted summaries, only the displacement feature shows a noticeably smaller distance for ABC with hand-crafted summaries (Figure 3B); the remaining summaries yield highly similar distances across all methods (Supplementary Figure 8). In contrast, the L^1 distance and cosine similarity metrics indicate that the predictions from the NPE-based methods are generally closer to the observed data. The ensemble NPE achieved the highest cosine similarity, whereas both the single and ensemble NPEs exhibited nearly identical L^1 distances. Because the UMAP projection was not

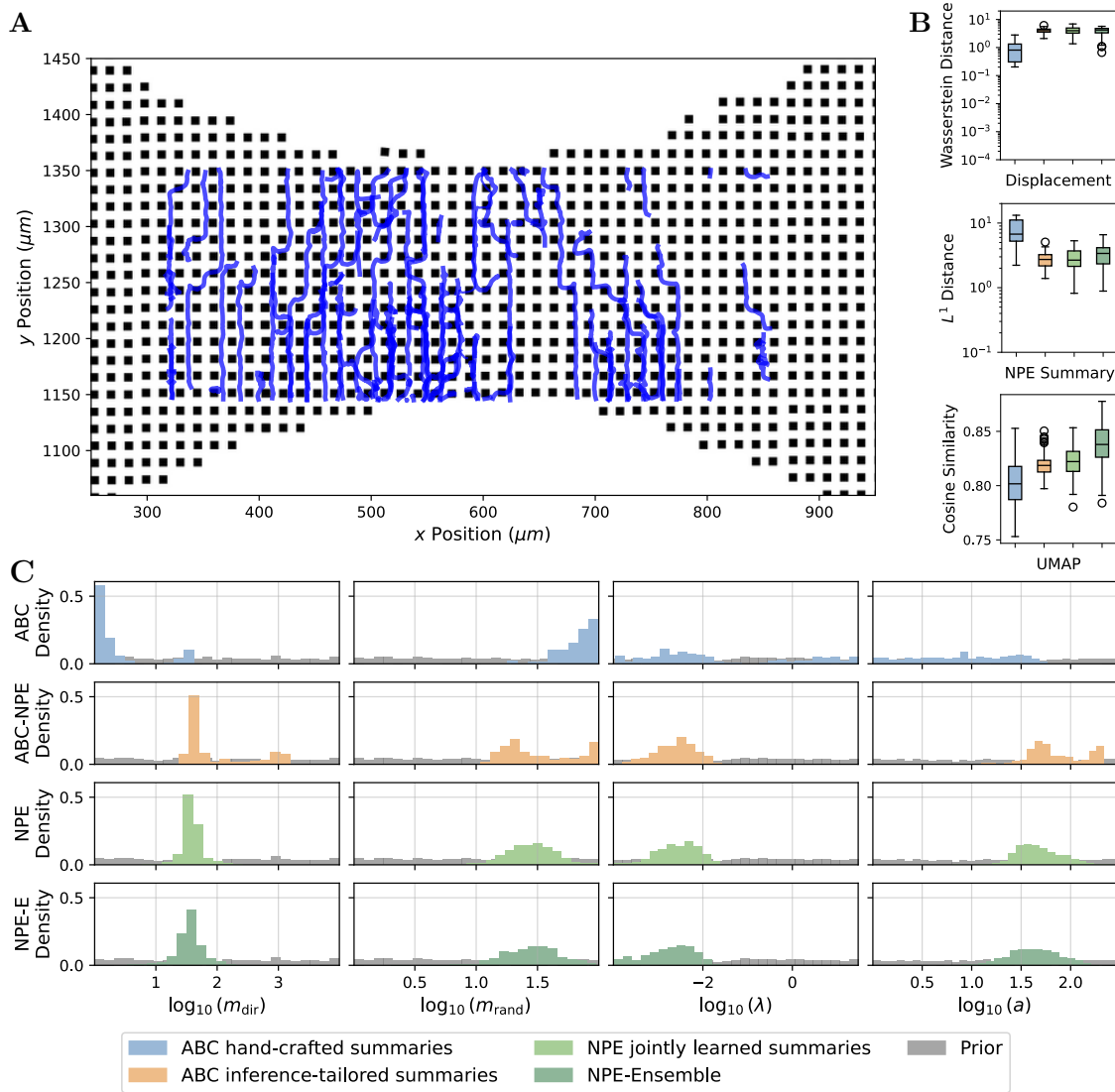


Figure 3: *Posterior approximations and summary statistics on experimental data.* (A) Experimental data (cells in blue) within the pillar forest are shown. (B) Distances between the hand-crafted summaries in the NPE summary space and UMAP representation were computed. All metrics are based on 100 simulation samples drawn from each posterior. Distributions are shown as box plots, indicating medians, quartiles, and outliers. (C) Posterior approximations using different ABC approaches and NPE with jointly trained summaries.

used during training and inference in any method, this metric offers an independent measure of posterior predictive quality. As an additional robustness check, we repeated the inference with NPE on parts of the dataset and controlled for model misspecification (see Supplementary Material).

The posterior distributions obtained from the ensemble and single NPE models were highly similar, whereas the ensemble was more conservative, indicating consistency across the training runs (Figure 3C). In contrast, the inference-tailored ABC posterior samples show slightly greater uncertainty and hint at a potentially multi-modal posterior, although its median aligns with that of the NPE-based estimates

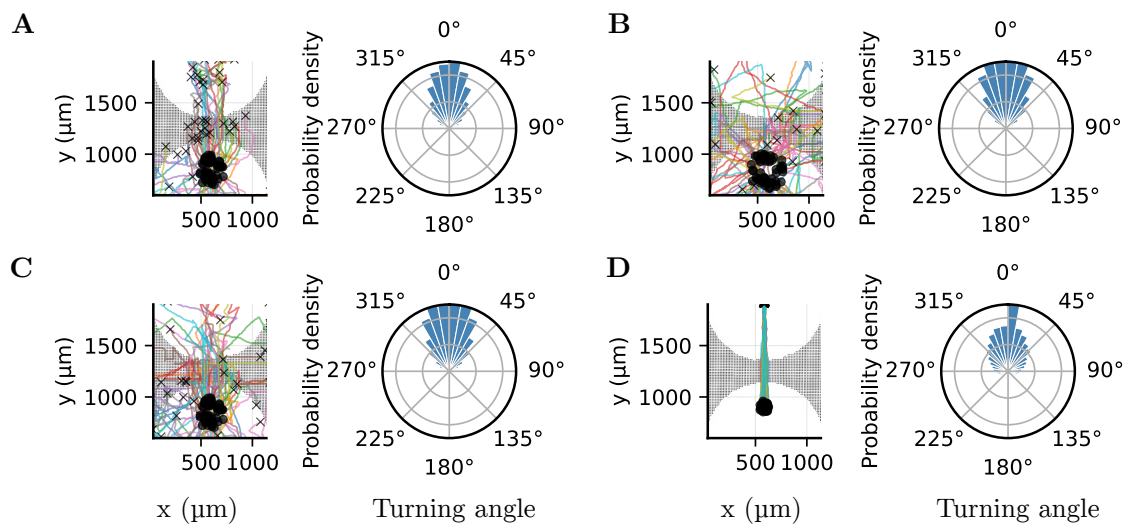


Figure 4: *Simulation study shows modulating role of obstacles.* The trajectory of 50 random cells (left) and the distribution of the turning angles (of all cells) (right) are shown. The endpoints are marked with 'x' if visible in the selected area. (A) Simulation setup mirroring the experimental setup with structural obstacles and a constant chemokine gradient. (B) Simulation setup without a chemokine. (C) Simulation setup in which the chemokine is active only for 30 min. (D) Simulation setup without persistence in the random walk but with a constant chemokine.

(Figure 3C). Notably, ABC using hand-crafted summaries yielded a markedly different posterior, with a distinct shift in the median and signs of bimodality (Figure 3C). For instance, the smaller mode in the inferred chemokine attraction strength m_{dir} aligns with the dominant mode found in the NPE posteriors (Figure 3C).

The median estimated cell area from NPE-E is $a = 41.73 \mu\text{m}^2$ (Figure 3C), aligning with the expected diameter of 10–15 μm [44, BNID 113239]. Although the model does not make any specific assumptions regarding the cell shape, this area suggests that the cells are larger than the gap between the two pillars (10 μm). Additionally, the median estimated rate parameter is $\lambda = 2.1 \text{ ms}^{-1}$, corresponding to a mean waiting time of 7.9 min, which is more than one order of magnitude larger than the time of 30 s between two images.

In conclusion, our results demonstrate the robustness and flexibility of our Cellular Potts model in capturing key cellular dynamics. By integrating posterior inference with neural summaries, we successfully identified critical parameters that govern cell migration.

Simulation study shows modulating role of physical obstacles and persistence in cell migration

To assess how physical structures modulate migratory behavior, we used our computational model and inference pipeline to investigate the impact of obstacles on cell movement. Specifically, we aimed to disentangle the independent contributions of chemokine guidance and persistence in random motility.

Based on the posterior median of the model parameters from the NPE ensemble,

we simulated cell migration under four distinct conditions: with (Figure 4A) and without a chemokine gradient that was constant over time (Figure 4B), with an initial chemokine signal lasting only 30 min (Figure 4C), and with suppressed persistence in random movement (intrinsic random movement was set to $m_{\text{rand}} = 0$) but a constant chemokine gradient (Figure 4D). Simulations were sampled at a higher resolution of 10-second intervals and also recorded outside the visible window of the experimental setting, enabling controlled comparisons of cell responses to environmental cues influencing migration dynamics.

The results demonstrate that persistence is a key driver of cell dispersion (Figure 4). Even in the absence of directional guidance through the chemokine gradient, cells with persistent movement spread widely throughout the environment (Figure 4B). In contrast, when persistence is removed, migration is spatially constrained and directionally limited (Figure 4D). The pillar forest further modulates this behavior by imposing spatial constraints that can trap cells. Moreover, simulations with a short-lived chemokine gradient revealed that the removal of guidance cues allowed trapped cells to reorient and resume dispersal (Figure 4C). This suggests a dynamic interplay: while chemokines bias movement, their prolonged presence may restrict escape from local traps in obstacle-rich environments.

This phenomenon became even more apparent when analyzing the number of cells that reached the upper clearing of our experimental setup. Simulations where the chemokine signal is active only at the first 50–75 % of the simulation time show a higher proportion of successful arrivals compared to scenarios with constant guidance (Supplementary Figure 9). Our experimental setup does not allow us to directly observe and thus verify this outcome, as our imaging is limited to the pillar forest region and does not extend to the final source location. Nevertheless, a similar effect may occur *in vivo*, potentially because of signal decay or attenuation near the source. This counterintuitive result highlights that temporally modulated guidance cues can facilitate more efficient exploration in the presence of obstacles.

Further analysis revealed that persistence is particularly important for enabling movement orthogonal to the chemokine gradient (Figure 4D). These findings underscore that intrinsic motility parameters and environmental features jointly determine the range and structure of cellular trajectories.

In summary, the simulations show that physical obstacles actively reshape migration, not only by slowing cells but also by altering the directionality and dispersion of their paths. While chemokines impose a directional bias, obstacles introduce nonlinear constraints that modify or obscure this signal in the observable data.

Discussion

A quantitative understanding of cell migration is essential across diverse domains, including immunology, oncology, and tissue engineering, where directed cell movement plays a central role in physiological and pathological processes. Despite its importance, methods for inferring the parameters of mechanistic models of migration from experimental data remain limited, especially in spatially structured or noisy microenvironments, where hand-crafted summaries may fail to capture the relevant dynamics.

350 The findings of this study provide a foundation for researchers to enhance
 351 modeling approaches for biological systems. Specifically, our study underscores the
 352 substantial impact of spatial obstacles on cell migration, as evidenced by the manner
 353 in which the pillar forest environment regulates both the speed and directional
 354 alterations. Additionally, we demonstrate that the learned summaries exhibit
 355 superior performance compared to traditional summary statistics in the context of
 356 simulation-based inference. Our simulation study further revealed that persistent
 357 cell motion plays a critical role in overcoming spatial constraints and that temporally
 358 modulated chemokine cues can facilitate dispersal in obstacle-rich environments.

359 The pillar forest used in our simulations provides a well-controlled and inter-
 360 pretable setting for uncovering the fundamental principles of cell migration. Building
 361 on our framework, future studies could examine how cells behave in spatially and
 362 temporally complex environments, where guidance cues may be transient, competing,
 363 or context-dependent. It remains an open question how cells resolve directional
 364 conflicts in the presence of multiple chemotactic sources or how robust their migra-
 365 tory behavior is in the face of spatial noise or fragmented gradients. Addressing
 366 these challenges would advance our understanding of how immune cells navigate
 367 physiologically realistic environments and flexibly respond to complex environmental
 368 stimuli.

369 The utilization of learned summaries in ABC shares similarities with other
 370 approaches that employ an estimator trained on pairs of simulations and parameters
 371 to predict the posterior mean or learn a summary based on an autoencoder [40, 45].
 372 Yet, the latent space of these autoencoders is not specifically tailored for inference
 373 but for reconstruction of the data. The posterior mean estimators are then used in
 374 the context of ABC to generate summary statistics [37, 46, 47, 48, 49]. However,
 375 we demonstrated that the posterior mean was not sufficiently informative. Our
 376 results suggest that ABC-PM exhibits a systematic bias, despite the summary
 377 network itself showing no apparent bias during validation. This discrepancy likely
 378 arises because predicting only the posterior mean does not adequately capture
 379 the uncertainty inherent in the data, resulting in biased estimates. Therefore, it
 380 was recently proposed to append noise estimates or higher-order moments of the
 381 parameters to the posterior mean [40, 39]. However, this approach remains a type of
 382 manual feature engineering. Moreover, although the neural network architecture is
 383 identical for both the posterior mean and inference-tailored summaries, joint training
 384 with the normalizing flow removes the constraint that the output dimensionality
 385 of the summary network matches the parameter space, thereby enabling greater
 386 representational flexibility.

387 Adaptive distances based on hand-crafted summaries have been extensively
 388 studied [38, 39]. Yet, identifying the optimal distance measure depends heavily
 389 on the informativeness of the chosen statistics, which may fail to capture the
 390 necessary information. In contrast, the joint learning of summaries with a neural
 391 posterior estimator ensures that the summaries are maximally informative and
 392 eliminates the need for additional upfront simulations. Moreover, our findings
 393 indicate that ABC requires 10 times more simulations to attain comparable accuracy,
 394 thereby highlighting the efficacy of neural posterior estimation, which is consistent
 395 with previous findings [50, 51]. As cells are heterogeneous in size and even have
 396 specific shapes depending on the chemical gradient [52], a further direction could

397 be to quantify individual differences between cells using an amortized hierarchical
398 framework [53, 54].

399 Instead of applying the Wasserstein distance to the hand-crafted summaries,
400 the distance has been utilized directly on the observed data [55]. However, this
401 approach has not been generalized to a spatial context with a varying number of
402 observations and therefore cannot be applied in our context. In general, if the
403 quantiles of the statistic (or the data) do not vary significantly as the parameter
404 changes, the Wasserstein distance will not change in a meaningful manner and thus
405 cannot serve as an accurate posterior approximation [56].

406 An opportunity to extend our model lies in its current assumption of a homoge-
407 neous and static environment. While this simplification has provided clear insights
408 into core migration behaviors, it does not capture the dynamic, feedback-rich nature
409 of many biological systems, where cells actively remodel their surroundings. For
410 example, the chemokine concentration diffuses over time, migrating cells may re-
411 shape local chemotactic gradients [57, 58], restructure the extracellular matrix [59],
412 or exert mechanical forces, establishing feedback loops that influence both their
413 own motility and collective behavior [60]. Moreover, dendritic cell migration is
414 tightly integrated with their immunological functions, such as antigen capture,
415 cytokine production, and T-cell priming, and is influenced by a complex interplay
416 of extracellular cues and intracellular signaling pathways [61]. Incorporating such
417 bidirectional cell–environment interactions into future models offers the potential
418 for more faithful representations of the adaptive and co-evolving dynamics that
419 characterize living tissues. Experimental evidence further shows that dendritic cells
420 require stable spatial chemokine gradients to maintain directional persistence, as
421 transient or oscillatory cues rapidly disrupt cytoskeletal polarization and migra-
422 tion [62]. This highlights that dynamic environmental signals may not only fail to
423 support chemotaxis but can actively destabilize it, reinforcing the need to jointly
424 model spatial and temporal features.

425 The findings reveal that learned summaries, tailored specifically for inference,
426 exhibit superior performance compared to hand-crafted summary statistics and
427 provide more information than summaries based on the posterior mean. This
428 enhancement in performance can be attributed to the adaptability of neural networks,
429 which automatically extract relevant features from data, thereby eliminating the need
430 for manually designed summaries. Instead, these networks learn features that are
431 specifically relevant to the inference task. By leveraging this approach, researchers
432 can develop more complex models and calibrate them effectively with experimental
433 data, paving the way for deeper insights into the fundamental mechanisms of cell
434 migration.

435 Methods

436 In this study, we analyzed the movement of bone marrow-derived dendritic cells
437 (DCs) in complex microenvironments, where both chemical signaling and physical
438 barriers influence their movement.

439 **Experimental procedures**

440 DCs are confined in an environment with 7 ng of the chemokine (C-C motif) ligand
441 19 (CCL19), which binds to the C-C chemokine receptor type 7 (CCR7) expressed
442 on the surface of mature DCs [63]. To create physical obstacles, the area between
443 the two confining surfaces was intersected by pillars with defined gaps: a “pillar
444 forest” [43] (see Figure 1). The presence of these obstacles introduces physical
445 complexity, compelling cells to traverse the gaps of 10 μm . This adds a layer of
446 mechanical challenge to their movement, as the cells constantly encounter pillars on
447 their migratory path.

448 **Dendritic cell culture from mouse bone marrow**

449 Femurs and tibias from the legs of 6–12 week-old C57BL/6J mice were removed
450 and placed in 70 % ethanol for 2 min. Bone marrow was flushed with PBS using a
451 26-gauge needle. 2×10^6 cells were seeded per 100 mm Petri dish containing 9 mL of
452 complete medium (Roswell Park Memorial Institute (RPMI) 1640 supplemented with
453 10 % Fetal Calf Serum (FCS), 2 mM L-Glutamine, 100 U/mL Penicillin, 100 $\mu\text{g mL}^{-1}$
454 Streptomycin, and 50 μM β -mercaptoethanol; ThermoFischer Scientific) and 1 mL
455 Granulocyte–Monocyte Colony Stimulating Factor (GM-CSF, supernatant from
456 hybridoma culture). On days 3 and 6, complete medium supplemented with 20 %
457 GM-CSF was added to each dish. To induce maturation, the cells were stimulated
458 overnight with 200 ng mL^{-1} lipopolysaccharide (LPS) and used for experiments on
459 day 8–9. Alternatively, DCs were frozen in FCS containing 10 % DMSO on day 7.
460 For experimental use, the cells were thawed the day before the experiment and
461 stimulated with 200 ng mL^{-1} LPS overnight.

462 **Photomask patterning**

463 Photomasks were drawn in CorelDraw and converted to Gerber in LinkCad. Pho-
464 tomasks were fabricated at JD Photodata (UK). These photomasks were used to
465 define microscale patterns on a photoresist-coated silicon wafer during UV lithog-
466 raphy. The transparent and opaque regions of the mask control the areas where
467 UV light reaches the photoresist, enabling precise pattern transfer. To produce the
468 master wafer, a 4 inch wafer was first baked at 110 °C for 5 min. 6005 TF SU8 was
469 spun up to 3000 RPM with an acceleration of 300 RPM/s for 30 s. The wafer was
470 then baked at 110 °C for 5 min. The wafer was exposed to 100 mJ/cm^2 UV light
471 through the photomask, then baked for 5 min at 110 °C, and then developed in SU8
472 developer. As a final step, the wafer was baked at 135 °C for 5 min to permanently
473 harden the SU8. The height of the wafer was approximately 5 μm .

474 **Pillar forest migration assay**

475 Microfabricated chips were produced by pouring a polydimethylsiloxane (PDMS)
476 mixture (silicon rubber and curing agent at a ratio of 10:1 by weight) onto a silicon
477 wafer with a previously designed photomask pattern. This was incubated at 80 °C
478 overnight after removing air bubbles by placing the PDMS in a vacuum desiccator.
479 The PDMS was separated from the wafer, and single chips were cut out using a
480 surgical blade. Next, two holes (diameter of 1.5 mm) on each side of the pillar forest

area were punched at a distance of 1 mm, and the PDMS was bound upside-down to a glass-bottom dish. To activate the surface, both the PDMS chip and dish were cleaned with plasma before assembly. To equilibrate the chip, it was flushed with complete medium and incubated at 37 °C, 5 % CO₂ for at least 1 h. Afterward the medium was removed from the two holes, and they were filled with 7 μ L CCL19 solution at a concentration of 1 g mL⁻¹ on one side and 3×10^4 cells in complete medium on the opposite side. Additionally, dextran was added to the CCL19 solution (1:1000) to visualize the chemokine gradient. BMDCs were stained with NucBlue (Invitrogen) and concentrated to 6.25×10^6 cells/mL in advance.

Microscopy and image analysis

After a 2–2.5 h incubation at 37 °C, 5 % CO₂, samples were imaged with an inverted wide-field Nikon Eclipse Ti-2E microscope in a humidified and heated chamber at 37 °C and 5 % CO₂ (Ibidi Gas Mixer), equipped with a Plan-Apochromat 20 $\times$ /0.8 air objective, a DS-Qi2 camera, and a Lumencor Spectra X light source (390 nm, 475 nm, 542/575 nm; Lumencor). Movies were taken for 2 h, recording one image every 30 s. Videos were analyzed using Fiji [64]. Cell nuclei tracking was performed manually or automatically using the Fiji plugin TrackMate [65]. However, owing to the microscope setup, tracking was only possible within the pillar forest between the two clearings.

Model description

Cell migration behavior was described using a modified Cellular Potts model (CPM), a lattice-based framework that effectively represents both individual cell dynamics and interactions within complex environments [17]. Each cell is represented as a set of connected lattice sites on a discrete 3-dimensional grid. The system evolves through stochastic updates that minimize the overall energy function H using a Metropolis algorithm. In each iteration, a lattice site i and one of its neighboring sites j are randomly selected. Cell j is proposed to replace cell i , and the resulting change in energy ΔH is calculated. The proposed change is then probabilistically accepted based on the Metropolis criterion.

The CPM energy function is given by

$$H = \sum_{\langle i,j \rangle} J_{\sigma(i),\sigma(j)} + \sum_{c=1}^N (v_c - a)^2, \quad (1)$$

where $\sigma(i)$ denotes the cell at site i , and the first sum runs over neighboring lattice sites $\langle i,j \rangle$ with $\sigma(i) \neq \sigma(j)$. The interaction term $J_{\sigma(i),\sigma(j)}$ models the interface energies between different cells and the medium. The second term penalizes deviations of a cell's volume v_c from a target volume a and contributes to realistic cell shapes. This volume constraint indirectly affects cell motility by limiting the number of lattice sites a cell may occupy. The combined effects of surface tension and volume conservation allow the model to capture mechanical interactions with neighboring cells and the extracellular matrix. In dense environments, deviations from the preferred volume can generate mechanical pressure, resulting in tissue-level reorganization.

521 To model chemotactically guided cell migration, we extended the basic CPM
522 with two additional components: directional movement in response to chemokine
523 gradients and intrinsic persistent random motion. To bias cell movement toward
524 higher concentrations, the chemotactic signal is integrated into the energy function
525 via an additive term

$$\Delta E_{\text{chemotaxis}} = m_{\text{dir}} \cdot (f_{x_i} - f_{x_j}), \quad (2)$$

526 where f_{x_i} and f_{x_j} are the local chemoattractant concentrations at the occupied
527 and empty lattice sites, respectively, and m_{dir} controls the strength of chemotactic
528 attraction [66]. The update is energetically favored if the move leads to a position
529 with a higher concentration.

530 To account for undirected and exploratory cell movements, we implemented a
531 persistent random walk. This implementation resembles a run-and-tumble process
532 and reflects experimentally observed persistent random motion [22]. Each cell was
533 assigned a preferred migration direction that was periodically updated. The energy
534 contribution for a proposed move in the direction $\mathbf{s}_c$ is given by another additive
535 term to the energy function H :

$$\Delta E_{\text{random}} = -m_{\text{rand}} \cdot v_{c,t} \cdot \left\langle (\cos \alpha, \sin \alpha)^T, \mathbf{s}_c \right\rangle, \quad (3)$$

536 where α is the current migration angle, $v_{c,t}$ is the current area of the cell c , and m_{rand}
537 is the strength of intrinsic motility. The scalar product penalizes deviations from
538 the preferred direction. For each cell, the angle α is updated after a waiting time
539 $t_\alpha \sim \text{Exp}(\lambda)$ with rate λ , and a new direction is sampled uniformly from $\mathcal{U}[0, 2\pi]$.

540 Because the cells are comparatively flat in the considered experimental setup, a
541 2D model was used to capture the exploration details without adding unnecessary
542 complexity. We used an image to position a dense array of pillars between the
543 two ends of the migration area, thereby reproducing the spatial constraints of
544 our experimental setup. The model was built using **MorpheusML**, an XML format
545 for encoding multiscale models and multicellular biological systems [22]. Spatial
546 resolution is set to $1.31 \mu\text{m}$ per node length. To emulate the experimental setting,
547 we simulated cell movement in the entire area and extracted the coordinates of cells
548 in the pillar forest at 30-second intervals. The chemokine gradient was modeled as a
549 time-independent Gaussian distribution with a peak concentration of $7 \times 10^6 \mu\text{m}^{-2}$
550 and a standard deviation of $550 \mu\text{m}$. This choice reflects typical diffusion distances
551 and ensures a smooth and biologically meaningful gradient.

552 In summary, the parameters to be inferred from the data in this study are

$$\phi = [\log_{10}(m_{\text{dir}}), \log_{10}(m_{\text{rand}}), \log_{10}(\lambda), \log_{10}(a)]^T \quad (4)$$

553 with their corresponding priors provided in Table 2. Uniform priors were specified
554 on the logarithmic scale, and bounds were selected based on biological constraints
555 or the computational feasibility of the simulations. The feasibility of the simulations
556 was assessed by evaluating the simulations at the upper and lower bounds of the
557 parameter space. Additional technical constants used in the CPM are documented
558 in the model file, provided here: <https://identifiers.org/morpheus/M6342>.

Table 2: *Parameters to estimate in the cell migration model.* Parameters were transformed to a logarithmic scale and had uniform priors.

Parameter	Description	Prior
$\log_{10}(m_{\text{dir}})$	Influence of chemokine attraction	$\mathcal{U}[0, 4]$
$\log_{10}(m_{\text{rand}})$	Strength of intrinsic random movement	$\mathcal{U}[0, 2]$
$\log_{10}(\lambda)$	Rate of the waiting time distribution (s^{-1})	$\mathcal{U}[-4, 1.5]$
$\log_{10}(a)$	Area of cells (μm^2)	$\mathcal{U}[0, 2.5]$

Hand-crafted summary statistics

The dynamics of the model depend on several unknown parameters that must be estimated from the available data. We employ approximate Bayesian computation using hand-crafted summary statistics that were built specifically for this model, referred to as *ABC with hand-crafted summary statistics*.

Each simulation contains multiple cell tracks of varying lengths, as cells could move out of the observation area or leave the 2D plane. Because the trajectory of a cell is governed by stochastic movement, direct matching of a simulated cell to a real cell is not possible. Instead, summary statistics that characterize individual cell movement were constructed. These statistics were computed for each cell, and the distributions of these statistics were compared between the simulation and observed data. This approach eliminates the need for direct track matching and allows for flexibility in the number of simulated cells. However, for reliable parameter inference, the number of cells should not deviate too much, as crowded and sparse environments differ significantly; for example, cells may block each other more frequently in a crowded environment.

We constructed four different widely used summary statistics to describe different aspects of cell movement [67, 68]. Denoting the position of a cell $c \in \{1, \dots, N\}$ as $\mathbf{X}_i^c = (x_i^c, y_i^c)$ at time step i for $i \in \{1, \dots, T^c\}$, we define the following statistical quantities. Note that the time steps of each cell are relative to the first time they were observed.

- **Displacement (D)** measures the squared distance traveled by the cell c over the observed time:

$$D^c = \frac{1}{\sqrt{T^c}} \sqrt{(x_{T^c}^c - x_1^c)^2 + (y_{T^c}^c - y_1^c)^2}, \quad (5)$$

where T^c is the number of observed time steps for each cell. We scaled the displacement by time so that different observed sequence lengths could be handled. For a standard random walk, the expected translation distance after T steps is of the order of $\sqrt{T}$.

- **Velocity (V)** is the average speed of a cell c and is expressed as the mean over all time steps

$$V^c = \frac{1}{T^c - 2} \sum_{i=2}^{T^c} \frac{1}{\Delta t} \sqrt{(x_i^c - x_{i-1}^c)^2 + (y_i^c - y_{i-1}^c)^2}, \quad (6)$$

where $\Delta t = 30 \text{ s}$ is the interval at which the cells were observed.

- 589 • **Turning angle (TA)** quantifies the mean change in directionality of a cell c
 590 between consecutive movements:

$$\text{TA}^c = \frac{1}{T^c - 3} \sum_{i=3}^{T^c} (\theta_i^c - \theta_{i-1}^c), \quad (7)$$

591 where $\theta_i^c = \arctan2(x_i^c - x_{i-1}^c, y_i^c - y_{i-1}^c)$.

- 592 • **Angle degree (AD)** captures the average absolute direction between two
 593 consecutive positions:

$$\text{AD}^c = \frac{1}{T^c - 2} \sum_{i=2}^{T^c} \arctan2(x_i^c - x_{i-1}^c, y_i^c - y_{i-1}^c). \quad (8)$$

594 To compute the acceptance criterion in ABC, we used the Wasserstein distance
 595 to calculate the distance between the samples of the distributions of the summaries
 596 of the simulations and the summaries of the experimental data. The Wasserstein
 597 distance quantifies the cost of transforming one distribution into another, thereby
 598 providing a comprehensive measure of discrepancy (see [69] for a detailed introduc-
 599 tion). By comparing the distributions of the summary statistics for the simulated
 600 and observed cells, we captured the differences across the entire dataset, avoiding
 601 the matching of trajectories. To integrate multiple statistics while minimizing
 602 the overreliance on any single summary, the Wasserstein distance was computed
 603 separately for each statistic. To normalize the contribution of each summary, the
 604 distance is weighted by the inverse of the difference between the largest and smallest
 605 statistics, as suggested in [70]. These weights were calculated from simulations of
 606 a pre-calibration population with parameters sampled from the prior. Finally, the
 607 weighted distances are aggregated by summing them up.

608 Neural posterior estimation with learned summary statistics

609 Neural posterior estimation (NPE) methods leverage neural networks to learn
 610 mappings from data to model parameters and can integrate learned summary
 611 networks that automatically extract relevant features from the data [42]. Instead of
 612 relying on hand-crafted summary statistics, we used a summary network s_ψ with
 613 trainable parameters ψ , designed to handle variable-length inputs, both in terms of
 614 the number of observed time points per cell and the number of cells per experiment.
 615 To extract informative features from each individual cell trajectory, we applied a
 616 combination of one-dimensional convolutional layers and a recurrent neural network,
 617 which is a standard architecture for time-series processing. To aggregate across
 618 multiple cells, we employed an attention-based pooling mechanism that produced
 619 a fixed-length summary vector while preserving flexibility with respect to the cell
 620 count (see Implementation for details). The summary network is then utilized
 621 to learn a d -dimensional representation $s_\psi(\mathbf{X}) \in \mathbb{R}^d$ of the observed data points
 622 $\mathbf{X} = \{\mathbf{X}^c(t) \in \mathbb{R}^{T^c \times 2} \mid c \in \{1, \dots, N\}\}$.

623 The summary statistics serve as inputs to a conditional normalizing flow, which
 624 transforms the posterior into a simpler density from which we know how to sample.
 625 This method allows efficient and accurate sampling and density evaluation [71,

72]. Let $\mathbf{z}$ be a latent variable described by a multivariate normal distribution $p(\mathbf{z})$. The model parameters ϕ are mapped to these latent variables conditional on the data $s_\psi(\mathbf{X})$ by a normalizing flow $f_{\psi'}(\phi, s_\psi(\mathbf{X})) = \mathbf{z}$. This invertible transformation is parameterized by ψ' and, by construction, has a tractable Jacobian. The approximation $q_{\psi'}$ to the target density $p(\phi | s_\psi(\mathbf{X}))$ is given by the change-of-variables formula

$$q_{\psi'}(\phi | s_\psi(\mathbf{X})) = p(\mathbf{z} = f_{\psi'}(\phi, s_\psi(\mathbf{X}))) \left| \det J_{f_{\psi'}}(\phi, s_\psi(\mathbf{X})) \right|. \quad (9)$$

If we know $f_{\psi'}$, we can sample from the posterior by sampling $\mathbf{z} \sim p(\mathbf{z})$ and applying $f_{\psi'}^{-1}(\mathbf{z}, s_\psi(\mathbf{X})) = \phi$.

To train the summary network s_ψ jointly with the conditional normalizing flow $f_{\psi'}$, we follow the work of [73] and minimize the Kullback-Leibler divergence between the true and approximate posterior distributions:

$$\begin{aligned} (\psi, \psi')^* &= \arg \min_{(\psi, \psi')} \mathbb{E}_{p(\mathbf{X})} [\text{KL}(p(\phi | s_\psi(\mathbf{X})) || q_{\psi'}(\phi | s_\psi(\mathbf{X})))] \\ &= \arg \min_{(\psi, \psi')} \iint -p(\mathbf{X}, \phi) \log q_{\psi'}(\phi | s_\psi(\mathbf{X})) d\mathbf{X} d\phi \\ &\approx \arg \min_{(\psi, \psi')} \frac{1}{S} \sum_{s=1}^S -\log q_{\psi'}(\phi^{(s)} | s_\psi(\mathbf{X}^{(s)})). \end{aligned} \quad (10)$$

The integral is approximated using S samples from the joint distribution $p(\mathbf{X}, \phi)$. These samples are obtained by sampling from the prior distribution $\phi \sim p(\phi)$ and simulating the model $\mathcal{M}(\phi)$. Using the transformation (9), the approximation in (10) can be efficiently evaluated. After training, we performed simulation-based calibration checks to validate the convergence of the neural posterior estimator (Supplementary Figure 1).

Approximate Bayesian computing with inference-tailored summary statistics

Studies have shown that NPE is sensitive to model misspecification and can potentially provide biased estimates [74, 75]. In contrast, ABC is sensitive to the choice of the summary statistics but converges to the optimal parameter under this summary [76]. To address these limitations, we consider a combined approach in this study: hand-crafted summaries of ABC are replaced by trained summary networks, as in NPE. The learned summaries are computed for each simulation by passing the data through the trained summary network. The L^1 distance is utilized in place of the Wasserstein distance on the learned summaries, as the objective is no longer to compare distributions but rather to enhance robustness against outliers [77]. This approach is referred to as *ABC with inference-tailored summaries* or ABC-NPE.

Approximate Bayesian computing with posterior mean summary statistics

Previous work suggested training a neural network $\tilde{s}_\psi(\mathbf{X}) \in \mathbb{R}^{\dim(\phi)}$ on data $\mathbf{X}$ to approximate the posterior mean $\mathbb{E}[\phi | \mathbf{X}]$ of the model parameters ϕ and then

using the trained network to compute summary statistics of the data [37, 46]. In this approach, the summary network is trained by optimizing the mean squared error objective

$$\psi^* = \arg \min_{\psi} \frac{1}{S} \sum_{s=1}^S \left\| \tilde{s}_{\psi}(\mathbf{X}^{(s)}) - \phi^{(s)} \right\|_2^2. \quad (11)$$

Hence, the output dimension of the network is fixed to the number of parameters. When the acceptance threshold ϵ is sufficiently small, this approach, referred to as *ABC with posterior mean summaries* (ABC-PM), converges to a posterior with the same mean as the exact posterior [37, 39].

Error metrics for posterior and simulated data

We computed the normalized root mean squared error (NRMSE) for each parameter across N posterior samples as follows:

$$\text{NRMSE} = \frac{1}{\max(\phi_{\text{true}}) - \min(\phi_{\text{true}})} \text{median} \left(\sqrt{\frac{1}{N} \sum_{i=1}^N (\phi_{\text{true}} - \phi_s^{(i)})^2} \right).$$

The reported NRMSE is the mean of the parameters.

To quantify the discrepancy between the simulated and observed data, we performed a UMAP projection [78] into a 10-dimensional space and computed the cosine similarity between each simulated and observed cell:

$$d_{\cos}(x_i, x_j) = \frac{x_i \cdot x_j}{\|x_i\| \|x_j\|}.$$

All UMAP hyperparameters were fixed to standard settings.

Implementation

We used the `FitMultiCell` [79] pipeline to load the model from `Morpheus` [22] and infer the parameters. One simulation takes between 10 s and 30 s. For the ABC algorithm, we chose a sequential Monte Carlo implementation in `pyABC` [80], setting a population size of 1000 particles and using stopping conditions of a minimal acceptance rate of 0.01 or a maximum of 15 generations. The acceptance threshold ϵ in each generation was set adaptively by calculating it as the median of the distances from the last population (as it is the default in `pyABC`). The acceptance threshold for the first generation is calculated in a pre-calibration round. We parallelized the simulations in each generation to maximize resource efficiency. If no cells were observed in the simulation, we set the Wasserstein distance to ∞ .

We built the neural posterior estimation using `BayesFlow` [81]. Using neural spline flows as an inference network [82], we trained networks with 6 to 8 layers. The smallest network served for inference, while all three together were employed as an ensemble network, i.e., samples of all networks were combined to obtain more conservative posterior estimates, as proposed by [83], and to check for model

misspecification [84, 85]. Additionally, we regularized the summary latent space to be normally distributed such that we can check for model misspecification if the real data is far from the simulated data [74].

To extract informative representations from time-resolved cellular data, we designed a summary network composed of convolutional, recurrent, and attention-based components. Each trajectory was first passed independently through a one-dimensional convolutional layer that captured local temporal features. This is followed by a gated recurrent unit (GRU) layer with 32 units, which models sequential dependencies across time points [86]. Both layers operate independently on each cell, ensuring parallel processing across cells. We implemented a temporal attention mechanism to aggregate the latent representations over time. A global query vector was computed by averaging the GRU outputs across all cells. This query is then used to attend to the time steps of each cell, enabling the model to focus on the most informative periods of each trajectory. The attended sequence is subsequently compressed via a pooling operation, resulting in a fixed-dimensional vector (dimension 8) that serves as a population-level summary statistic. Time points were appended to the coordinates, and missing observations in the trajectories (and observations outside the visible window in the experiment) were set to 0 and encoded by an additional binary indicator, as suggested by [87]. This resulted in a 4-dimensional tensor as input to the summary network. The standalone training of the summary network was performed by reducing the summary dimension to the number of parameters and minimizing the mean squared distance of the predictions and parameter values to approximate the posterior mean [37].

We used the same training set for the joint training of the summary and inference networks as for the standalone summary network. We performed 32,000 simulations, taking 1.39 h, employed a batch size of 32, and set 50 as the maximum number of epochs. From a further 100 simulations, we built a validation set and employed an early stopping. All simulations and parameters were standardized by computing the mean and standard deviation of each coordinate and parameter in the validation set. We trained with the Adam optimizer, an initial learning rate of 5×10^{-4} , and employed a cosine-decay schedule, as it is the default in **BayesFlow**. The training time for the joint training was 2.64 h and for the standalone summary network 4.04 h.

We ran all analyses on a computing cluster using 6 CPU nodes for parallelization, each with 32 cores, and one GPU for training the neural networks. The computing cluster used an Intel Xeon Sapphire Rapids CPU with a core clock speed of up to 2.1 GHz and 60 GB of RAM. The neural network training was performed on a cluster node with an Nvidia A40 graphics card with 48 GB of VRAM.

Acknowledgments

This work was supported by the German Federal Ministry of Education and Research (BMBF) (EMUNE/031L0293C), the European Union via the ERC grant INTEGRATE, grant agreement number 101126146, and under Germany’s Excellence Strategy by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) (EXC 2047—390685813, EXC 2151—390873048, and 524747443), the University of Bonn via the Schlegel Professorship of J.H., and the returning experts

730 fellowship of the Ministry of Innovation, Science, and Research of North-Rhine-
731 Westphalia (AZ: 421-8.03.03.02-137069). J.M. is a member of the Nanofabrication
732 Facility and is supported by the Institute of Science and Technology Austria. E.K.
733 acknowledges the TRA Life and Health (University of Bonn) as part of the Excel-
734 lence Strategy of the federal and state governments. We thank Laeschkir Würthner
735 for his insightful comments on the implementation of our model. The views and
736 opinions expressed are those of the authors only and do not necessarily reflect those
737 of the funding agencies.

738 Author CRediT

739 *Conceptualization:* J.A., E.K., J.H.; *Formal analysis:* J.A.; *Investigation:* R.Mo.,
740 J.M., E.K.; *Methodology:* J.A., E.A., M.V.; *Resources:* E.K., J.H.; *Software:* J.A.,
741 E.A., R.Mu.; *Visualization:* J.A., E.A.; *Writing – original draft:* J.A., E.A.; *Writing*
742 *– review & editing:* J.A., E.A., R.Mu., M.V., R.Mo., J.M., E.K., J.H..

743 Data and code availability

744 The underlying code and experimental data for this study are available and can
745 be accessed in this repository: [github.com/emune-dev/Cell-Migration-Complex-](https://github.com/emune-dev/Cell-Migration-Complex-Environments)
746 [Environments](https://github.com/emune-dev/Cell-Migration-Complex-Environments). Additionally, a snapshot of code and data is saved at Zenodo with
747 DOI [10.5281/zenodo.16893454](https://doi.org/10.5281/zenodo.16893454), and the MorpheusML model is also provided via this
748 identifier: identifiers.org/morpheus/M6342.

749 Competing interests

750 All authors declare no financial or non-financial competing interests.

751 References

- 752 [1] Yamada, K. M. & Sixt, M. Mechanisms of 3d cell migration. *Nature Reviews*
753 *molecular cell biology* **20**, 738–752 (2019).
- 754 [2] Saraswathibhatla, A., Indana, D. & Chaudhuri, O. Cell–extracellular matrix
755 mechanotransduction in 3d. *Nature Reviews Molecular Cell Biology* **24**, 495–516
756 (2023).
- 757 [3] Steinman, R. M., Kaplan, G., Witmer, M. D. & Cohn, Z. A. Identification of a
758 novel cell type in peripheral lymphoid organs of mice. v. purification of spleen
759 dendritic cells, new surface markers, and maintenance in vitro. *The Journal of*
760 *experimental medicine* **149**, 1–16 (1979).
- 761 [4] Weber, M. *et al.* Interstitial dendritic cell guidance by haptotactic chemokine
762 gradients. *Science* **339**, 328–332 (2013).

- 763 [5] Friedl, P. & Weigelin, B. Interstitial leukocyte migration and immune function.
764 *Nature immunology* **9**, 960–969 (2008).
- 765 [6] Teixeira, A. *et al.* T cell migration from inflamed skin to draining lymph nodes
766 requires intralymphatic crawling supported by icam-1/lfa-1 interactions. *Cell*
767 *reports* **18**, 857–865 (2017).
- 768 [7] Madsen, C. D. & Sahai, E. Cancer dissemination—lessons from leukocytes.
769 *Developmental cell* **19**, 13–26 (2010).
- 770 [8] Bule, P., Aguiar, S. I., Aires-da Silva, F. & Dias, J. Chemokine-directed tumor
771 microenvironment modulation in cancer immunotherapy. *International Journal*
772 *of Molecular Sciences* **22**, 9804 (2021).
- 773 [9] Ozga, A. J., Chow, M. T. & Luster, A. D. Chemokines and the immune
774 response to cancer. *Immunity* **54**, 859–874 (2021).
- 775 [10] Sarris, M. & Sixt, M. Navigating in tissue mazes: chemoattractant interpre-
776 tation in complex environments. *Current opinion in cell biology* **36**, 93–102
777 (2015).
- 778 [11] Pylvänäinen, J. W., Gómez-de Mariscal, E., Henriques, R. & Jacquemet, G.
779 Live-cell imaging in the deep learning era. *Current Opinion in Cell Biology* **85**,
780 102271 (2023).
- 781 [12] Anderson, A. R. A. & Quaranta, V. Integrative mathematical oncology. *Nat.*
782 *Rev. Cancer* **8**, 227–234 (2008).
- 783 [13] Waclaw, B. *et al.* A spatial model predicts that dispersal and cell turnover
784 limit intratumour heterogeneity. *Nature* **525**, 261–264 (2015).
- 785 [14] Tepole, A. B. Computational systems mechanobiology of wound healing.
786 *Computer methods in applied mechanics and engineering* **314**, 46–70 (2017).
- 787 [15] Meyer, K. *et al.* Bile canaliculi remodeling activates yap via the actin cytoskele-
788 ton during liver regeneration. *Mol. Syst. Biol.* **16**, e8985 (2020).
- 789 [16] Merkt, S. *et al.* A dynamic model for waddington’s landscape accounting for
790 cell-to-cell communication. *Available at SSRN 5051345* (2024).
- 791 [17] Graner, F. & Glazier, J. Simulation of biological cell sorting using a two-
792 dimensional extended Potts model. *Phys. Rev. Lett.* **69**, 2013–2016 (1992).
- 793 [18] Kerr, R. A. *et al.* Fast monte carlo simulation methods for biological reaction-
794 diffusion systems in solution and on surfaces. *SIAM J. Sci. Comput.* **30**,
795 3126–3149 (2008).
- 796 [19] Richmond, P., Walker, D., Coakley, S. & Romano, D. High performance cellular
797 level agent-based simulation with FLAME for the GPU. *Brief Bioinform.* **11**,
798 334–347 (2010).
- 799 [20] Swat, M. H. *et al.* Multi-scale modeling of tissues using CompuCell3D. *Methods*
800 *Cell Biol.* **110**, 325–366 (2012).

- 801 [21] Mirams, G. R. *et al.* Chaste: an open source C++ library for computational
802 physiology and biology. *PLoS Comput. Biol.* **9**, e1002970 (2013).
- 803 [22] Starruß, J., de Back, W., Brusch, L. & Deutsch, A. Morpheus: A user-friendly
804 modeling environment for multiscale and multicellular systems biology. *Bioinf.*
805 **30**, 1331–1332 (2014).
- 806 [23] Ghaffarizadeh, A., Heiland, R., Friedman, S. H., Mumenthaler, S. M. & Macklin,
807 P. Physicell: An open source physics-based cell simulator for 3-d multicellular
808 systems. *PLoS Comput. Biol.* **14**, e1005991 (2018).
- 809 [24] Matthäus, F., Jagodič, M. & Dobnikar, J. E. coli superdiffusion and chemo-
810 taxis—search strategy, precision, and motility. *Biophysical journal* **97**, 946–957
811 (2009).
- 812 [25] Stichel, D. *et al.* An individual-based model for collective cancer cell migration
813 explains speed dynamics and phenotype variability in response to growth factors.
814 *NPJ Syst Biol Appl* **3**, 5 (2017).
- 815 [26] Ben-Ami, Y., Pitt-Francis, J. M., Maini, P. K. & Byrne, H. M. Using a proba-
816 bilistic approach to derive a two-phase model of flow-induced cell migration.
817 *Biophysical Journal* **123**, 799–813 (2024).
- 818 [27] Toni, T., Jovanovic, G., Huet, M., Buck, M. & Stumpf, M. P. H. From
819 qualitative data to quantitative models: analysis of the phage shock protein
820 stress response in Escherichia coli. *BMC Syst. Biol.* **5** (2011).
- 821 [28] Jagiella, N., Rickert, D., Theis, F. J. & Hasenauer, J. Parallelization and high-
822 performance computing enables automated statistical inference of multi-scale
823 models. *Cell Syst.* **4**, 194–206 (2017).
- 824 [29] MacLean, A. L., Filippi, S. & Stumpf, M. P. H. The ecology in the hematopoietic
825 stem cell niche determines the clinical outcome in chronic myeloid leukemia.
826 *Proc. Natl. Acad. Sci. USA* (2014).
- 827 [30] Imle, A. *et al.* Experimental and computational analyses reveal that environ-
828 mental restrictions shape HIV-1 spread in 3D cultures. *Nature Communications*
829 **10**, 2144 (2019).
- 830 [31] Wieland, V. & Hasenauer, J. A stochastic modelling framework for cancer
831 patient trajectories: combining tumour growth, metastasis, and survival: V.
832 wieland, j. hasenauer. *Journal of Mathematical Biology* **90**, 65 (2025).
- 833 [32] Ross, R. J. *et al.* Using approximate Bayesian computation to quantify cell–cell
834 adhesion parameters in a cell migratory process. *NPJ systems biology and*
835 *applications* **3**, 9 (2017).
- 836 [33] Matsiaka, O. M., Baker, R. E., Shah, E. T. & Simpson, M. J. Mechanistic
837 and experimental models of cell migration reveal the importance of cell-to-cell
838 pushing in cell invasion. *Biomedical Physics & Engineering Express* **5**, 045009
839 (2019). URL <https://dx.doi.org/10.1088/2057-1976/ab1b01>.

- 840 [34] Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A. & Feldman, M. W. Popula-
841 tion growth of human y chromosomes: a study of y chromosome microsatellites.
842 *Mol. Biol. Evol.* **16**, 1791–1798 (1999).
- 843 [35] Beaumont, M. A., Zhang, W. & Balding, D. J. Approximate Bayesian Compu-
844 tation in Population Genetics. *Genetics* **162**, 2025–2035 (2002).
- 845 [36] Sisson, S. A., Fan, Y. & Beaumont, M. *Handbook of approximate Bayesian*
846 *computation* (Chapman and Hall/CRC, 2018).
- 847 [37] Fearnhead, P. & Prangle, D. Constructing summary statistics for approximate
848 Bayesian computation: semi-automatic approximate Bayesian computation. *J.*
849 *R. Stat. Soc. B* **74**, 419–474 (2012).
- 850 [38] Harrison, J. U. & Baker, R. E. An automatic adaptive method to combine sum-
851 mary statistics in approximate Bayesian computation. *PloS one* **15**, e0236954
852 (2020).
- 853 [39] Schälte, Y. & Hasenauer, J. Informative and adaptive distances and sum-
854 mary statistics in sequential approximate Bayesian computation. *Plos one* **18**,
855 e0285836 (2023).
- 856 [40] Albert, C., Ulzega, S., Ozdemir, F., Perez-Cruz, F. & Mira, A. Learning
857 summary statistics for Bayesian inference with autoencoders. *SciPost Physics*
858 *Core* **5**, 043 (2022).
- 859 [41] Cranmer, K., Brehmer, J. & Louppe, G. The frontier of simulation-based
860 inference. *Proceedings of the National Academy of Sciences* **117**, 30055–30062
861 (2020).
- 862 [42] Radev, S. T., Mertens, U. K., Voss, A., Ardizzone, L. & Köthe, U. BayesFlow:
863 Learning complex stochastic models with invertible neural networks. *IEEE*
864 *transactions on neural networks and learning systems* (2020).
- 865 [43] Renkawitz, J., Reversat, A., Leithner, A., Merrin, J. & Sixt, M. Micro-
866 engineered “pillar forests” to study cell migration in complex but controlled 3d
867 environments. *Methods in Cell Biology* **147**, 79–91 (2018).
- 868 [44] Milo, R., Jorgensen, P., Moran, U., Weber, G. & Springer, M. Bionumbers—the
869 database of key numbers in molecular and cell biology. *Nucleic acids research*
870 **38**, D750–D753 (2010).
- 871 [45] Park, J. W., Zhao, K. & Rane, S. Spatiodynamic inference using vision-based
872 generative modelling. *arXiv preprint arXiv:2507.22256* (2025).
- 873 [46] Jiang, B., Wu, T.-y., Zheng, C. & Wong, W. H. Learning summary statistic for
874 approximate Bayesian computation via deep neural network. *Statistica Sinica*
875 1595–1618 (2017).

- 876 [47] Wiqvist, S., Mattei, P.-A., Picchini, U. & Frellsen, J. Chaudhuri, K. &
877 Salakhutdinov, R. (eds) *Partially exchangeable networks and architectures*
878 *for learning summary statistics in approximate Bayesian computation.* (eds
879 Chaudhuri, K. & Salakhutdinov, R.) *Proceedings of the 36th International*
880 *Conference on Machine Learning*, Vol. 97 of *Proceedings of Machine Learning*
881 *Research*, 6798–6807 (PMLR, 2019). URL <https://proceedings.mlr.press/v97/wiqvist19a.html>.
882
- 883 [48] Borowska, A., Giurghita, D. & Husmeier, D. Gaussian process enhanced
884 semi-automatic approximate Bayesian computation: parameter inference in a
885 stochastic differential equation system for chemotaxis. *Journal of Computational*
886 *Physics* **429**, 109999 (2021).
- 887 [49] Åkesson, M., Singh, P., Wrede, F. & Hellander, A. Convolutional neu-
888 ral networks as summary statistics for approximate Bayesian computation.
889 *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **19**,
890 3353–3365 (2021).
- 891 [50] Lueckmann, J.-M., Boelts, J., Greenberg, D., Goncalves, P. & Macke, J.
892 Benchmarking simulation-based inference. *Proceedings of the 24th International*
893 *Conference on Artificial Intelligence and Statistics (AISTATS)* 343–351 (2021).
- 894 [51] Jørgensen, A. C. S., Ghosh, A., Sturrock, M. & Shahrezaei, V. Efficient Bayesian
895 inference for stochastic agent-based models. *PLOS Computational Biology* **18**,
896 1–28 (2022). URL <https://doi.org/10.1371/journal.pcbi.1009508>.
- 897 [52] Tweedy, L., Meier, B., Stephan, J., Heinrich, D. & Endres, R. G. Distinct cell
898 shapes determine accurate chemotaxis. *Scientific reports* **3**, 2606 (2013).
- 899 [53] Habermann, D. *et al.* Amortized Bayesian multilevel models. *arXiv preprint*
900 *arXiv:2408.13230* (2024).
- 901 [54] Arruda, J. *et al.* Compositional amortized inference for large-scale hierarchical
902 Bayesian models. *arXiv preprint arXiv:2505.14429* (2025).
- 903 [55] Bernton, E., Jacob, P. E., Gerber, M. & Robert, C. P. Approximate Bayesian
904 computation with the wasserstein distance. *J. Roy. Stat. Soc. B (Statistical*
905 *Methodology)* **81**, 235–269 (2019).
- 906 [56] Drovandi, C. & Frazier, D. T. A comparison of likelihood-free methods with
907 and without summary statistics. *Statistics and Computing* **32**, 42 (2022).
- 908 [57] Tweedy, L., Susanto, O. & Insall, R. H. Self-generated chemotactic gradi-
909 ents—cells steering themselves. *Current opinion in cell biology* **42**, 46–51
910 (2016).
- 911 [58] Tweedy, L. *et al.* Seeing around corners: Cells solve mazes and respond at a
912 distance using attractant breakdown. *Science* **369**, eaay9792 (2020).
- 913 [59] Friedl, P. & Wolf, K. Tumour-cell invasion and migration: diversity and escape
914 mechanisms. *Nature reviews cancer* **3**, 362–374 (2003).

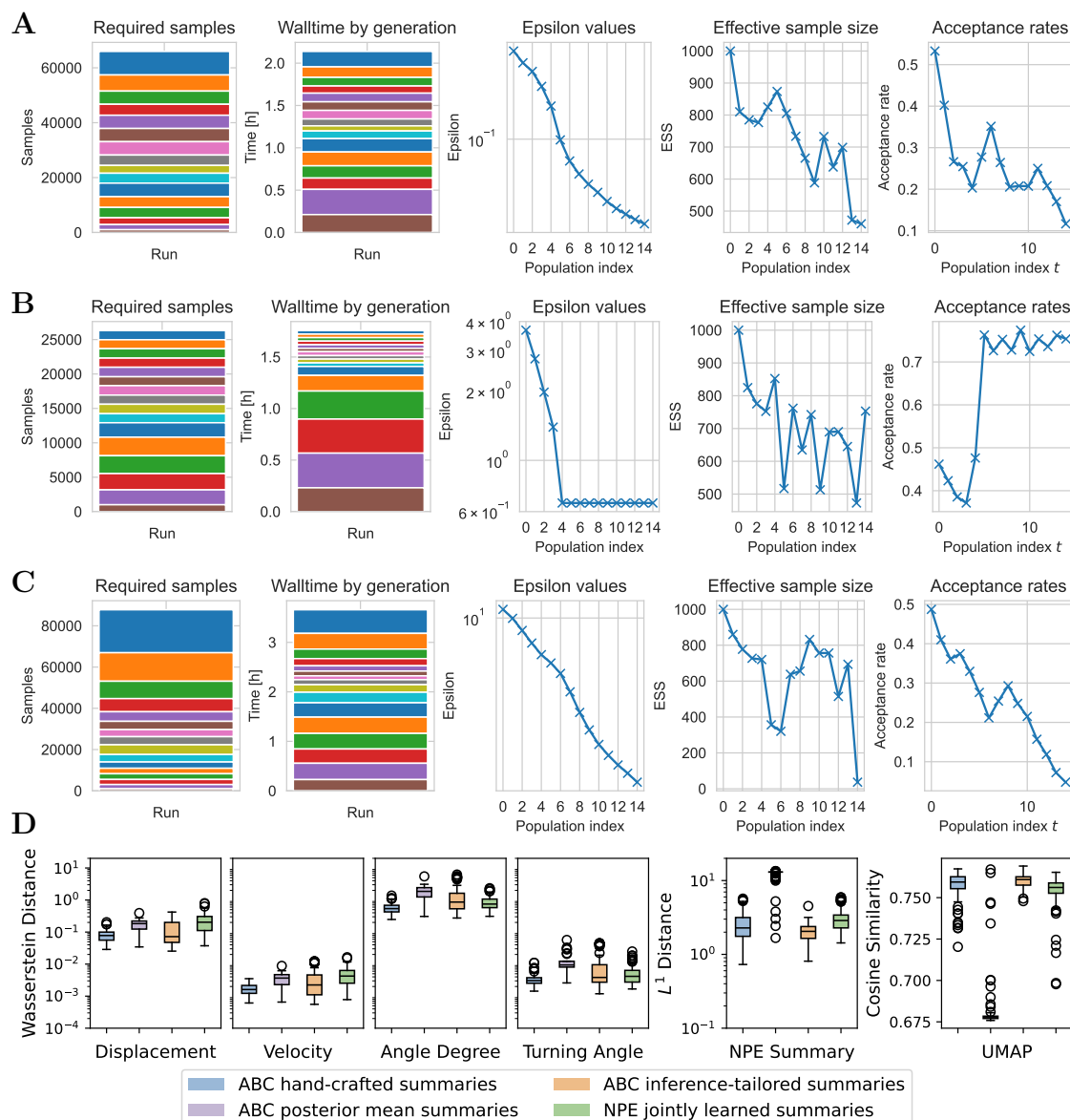
- 915 [60] Trepat, X. & Sahai, E. Mesoscale physical principles of collective cell organiza-
916 tion. *Nature Physics* **14**, 671–682 (2018).
- 917 [61] Liu, J., Zhang, X., Cheng, Y. & Cao, X. Dendritic cell migration in inflammation
918 and immunity. *Cellular & molecular immunology* **18**, 2461–2471 (2021).
- 919 [62] Quast, T. *et al.* A stable chemokine gradient controls directional persistence
920 of migrating dendritic cells. *Frontiers in cell and developmental biology* **10**,
921 943041 (2022).
- 922 [63] Förster, R. *et al.* Ccr7 coordinates the primary immune response by establishing
923 functional microenvironments in secondary lymphoid organs. *Cell* **99**, 23–33
924 (1999).
- 925 [64] Schindelin, J. *et al.* Fiji: an open-source platform for biological-image analysis.
926 *Nature methods* **9**, 676–682 (2012).
- 927 [65] Tinevez, J.-Y. *et al.* Trackmate: An open and extensible platform for single-
928 particle tracking. *Methods* **115**, 80–90 (2017).
- 929 [66] Savill, N. J. & Hogeweg, P. Modelling morphogenesis: from single cells to
930 crawling slugs. *Journal of theoretical biology* **184**, 229–235 (1997).
- 931 [67] Beltman, J., Marée, A. F. M. & De Boer, R. J. Analysing immune cell migration.
932 *Nat. Rev. Immunology* **9**, 789–798 (2009).
- 933 [68] Bundgaard, N. *et al.* Inferring cell motility in complex environments with
934 incomplete tracking data. *bioRxiv* (2022). URL [https://www.biorxiv.org/
935 content/early/2022/02/10/2022.02.10.479738](https://www.biorxiv.org/content/early/2022/02/10/2022.02.10.479738).
- 936 [69] Villani, C. *et al.* *Optimal transport: old and new* Vol. 338 (Springer, 2009).
- 937 [70] Prangle, D. Adapting the ABC distance function. *Bayesian Analysis* **12**,
938 289–309 (2017).
- 939 [71] Rezende, D. & Mohamed, S. Variational inference with normalizing flows.
940 *Proceedings of the 32nd International Conference on Machine Learning (ICML)*
941 1530–1538 (2015).
- 942 [72] Papamakarios, G., Nalisnick, E. T., Rezende, D. J., Mohamed, S. & Laksh-
943 minarayanan, B. Normalizing flows for probabilistic modeling and inference.
944 *Journal of Machine Learning Research* **22**, 1–64 (2021).
- 945 [73] Papamakarios, G., Pavlakou, T. & Murray, I. Masked autoregressive flow for
946 density estimation. *Advances in neural information processing systems* **30**
947 (2017).
- 948 [74] Schmitt, M., Bürkner, P.-C., Köthe, U. & Radev, S. T. Springer (ed.) *Detecting*
949 *model misspecification in amortized Bayesian inference with neural networks.*
950 (ed.Springer) *DAGM German Conference on Pattern Recognition*, 541–557
951 (2023).

- [75] Kelly, R. P. *et al.* Simulation-based Bayesian inference under model misspecification. *arXiv preprint arXiv:2503.12315* (2025).
- [76] Frazier, D. T. Robust and efficient approximate Bayesian computation: A minimum distance approach. *arXiv preprint arXiv:2006.14126* (2020).
- [77] Schälte, Y., Alamoudi, E. & Hasenauer, J. Robust adaptive distance functions for approximate Bayesian inference on outlier-corrupted data. *bioRxiv* (2021).
- [78] McInnes, L., Healy, J. & Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction (2020). URL <https://arxiv.org/abs/1802.03426>. arXiv:1802.03426.
- [79] Alamoudi, E. *et al.* Fitmulticell: Simulating and parameterizing computational models of multi-scale and multi-cellular processes. *Bioinformatics* **39**, btad674 (2023).
- [80] Schälte, Y., Klinger, E., Alamoudi, E. & Hasenauer, J. pyABC: Efficient and robust easy-to-use approximate Bayesian computation. *J. Open Source Softw.* **7**, 4304 (2022). URL <https://doi.org/10.21105/joss.04304>.
- [81] Radev, S. T. *et al.* BayesFlow: Amortized Bayesian workflows with neural networks. *Journal of Open Source Software* **8**, 5702 (2023).
- [82] Durkan, C., Bekasov, A., Murray, I. & Papamakarios, G. Wallach, H. *et al.* (eds) *Neural spline flows*. (eds Wallach, H. *et al.*) *Advances in Neural Information Processing Systems*, Vol. 32 (Curran Associates, Inc., 2019).
- [83] Hermans, J. *et al.* A trust crisis in simulation-based inference? your posterior approximations can be unfaithful. *stat.* **1050** (2022).
- [84] Elsemüller, L. *et al.* Sensitivity-aware amortized Bayesian inference. *arXiv preprint arXiv:2310.11122* (2023).
- [85] Alvey, J., Contaldi, C. R. & Pieroni, M. Simulation-based inference with deep ensembles: Evaluating calibration uncertainty and detecting model misspecification. *arXiv preprint arXiv:2507.13495* (2025). URL <https://arxiv.org/abs/2507.13495>.
- [86] Cho, K. *et al.* Learning phrase representations using rnn encoder-decoder for statistical machine translation (2014). URL <https://arxiv.org/abs/1406.1078>. arXiv:1406.1078.
- [87] Wang, Z., Hasenauer, J. & Schälte, Y. Missing data in amortized simulation-based neural posterior estimation. *PLOS Computational Biology* **20**, e1012184 (2024).
- [88] Talts, S., Betancourt, M., Simpson, D., Vehtari, A. & Gelman, A. Validating Bayesian inference algorithms with simulation-based calibration. *arXiv preprint arXiv:1804.06788* (2020).

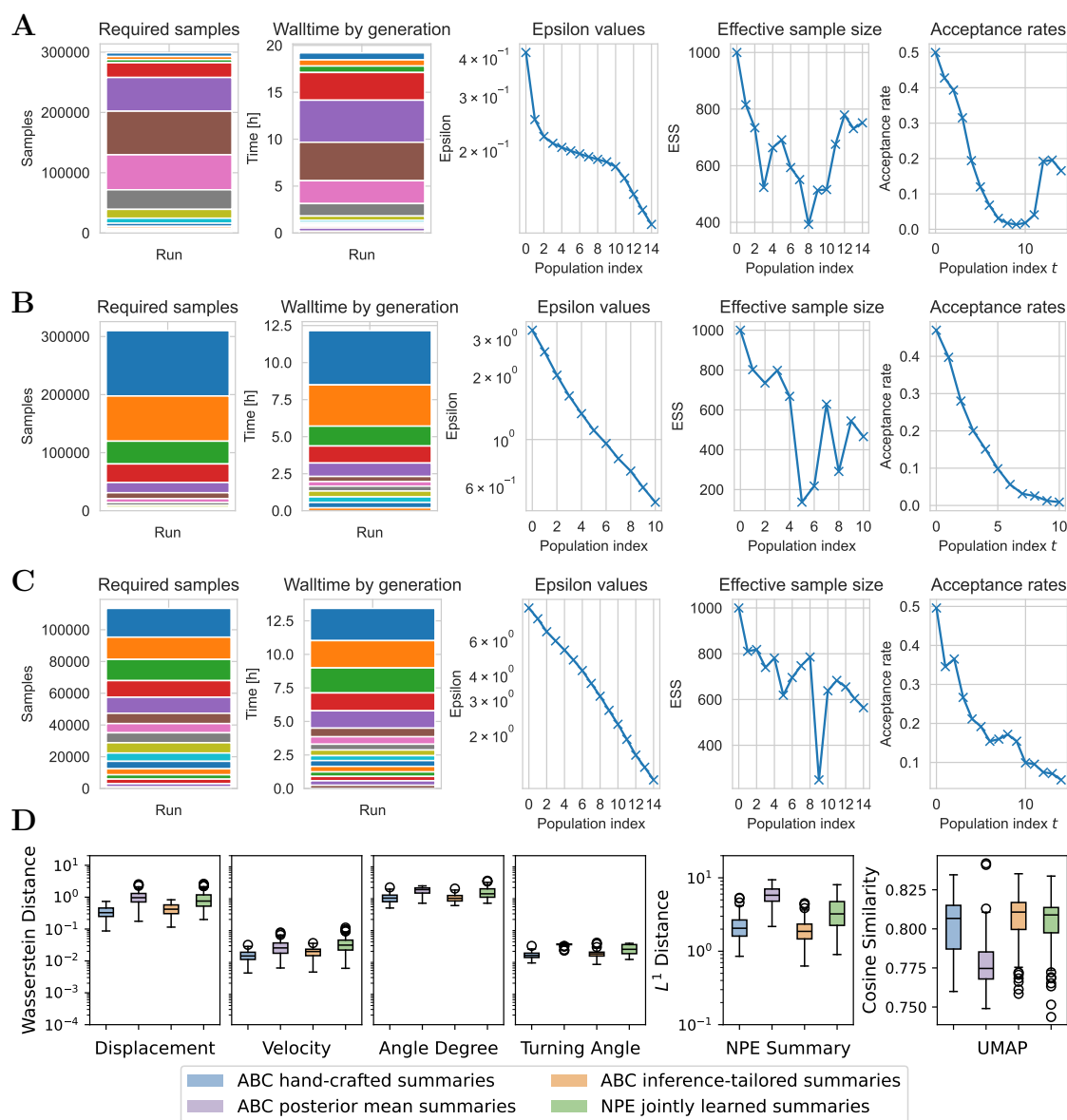
- 989 [89] Säilynoja, T., Bürkner, P.-C. & Vehtari, A. Graphical test for discrete uni-
 990 formity and its applications in goodness-of-fit evaluation and multiple sample
 991 comparison. *Statistics and Computing* **32**, 32 (2022).

Supporting information

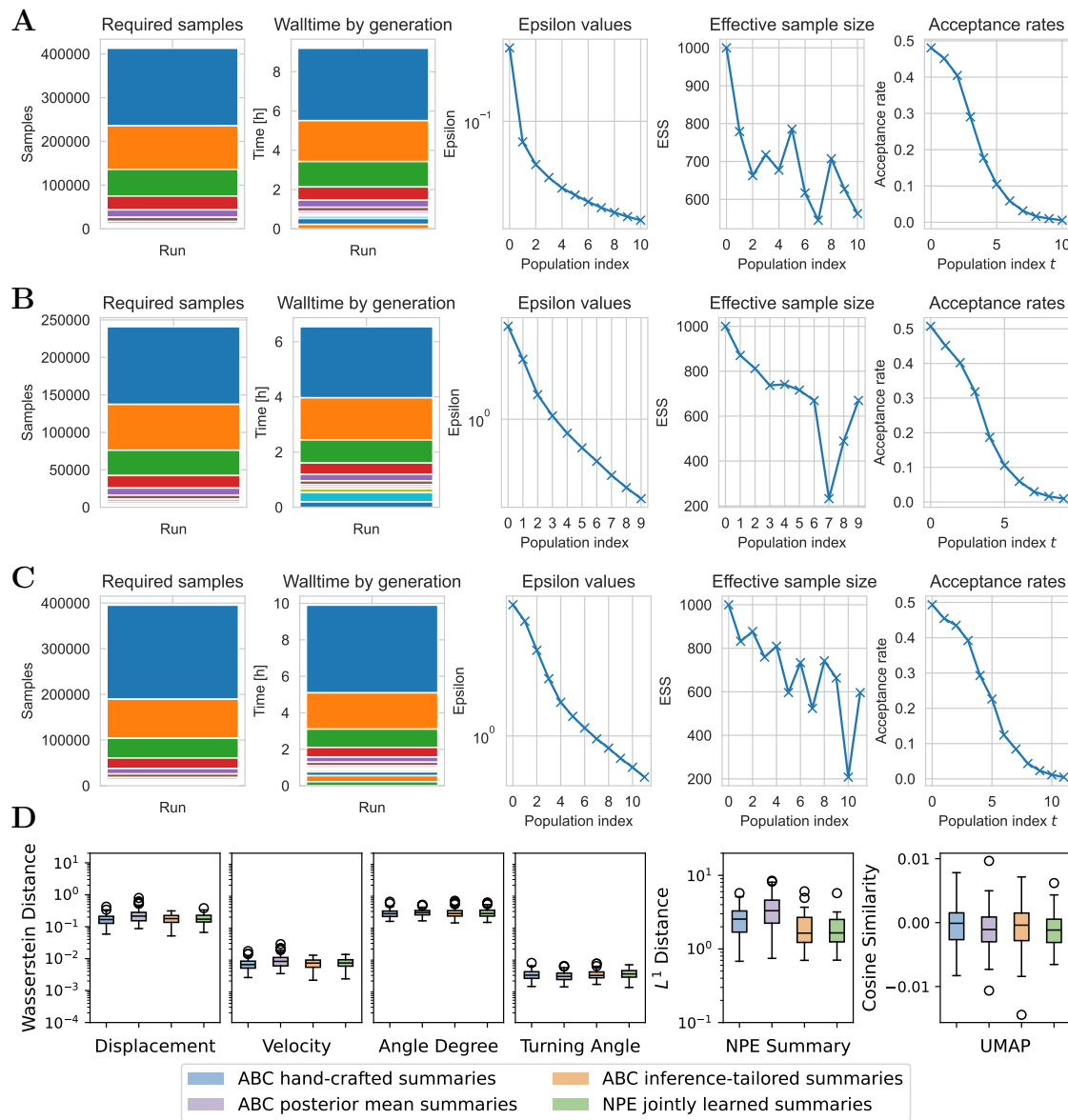
Additional results for the simulated dataset



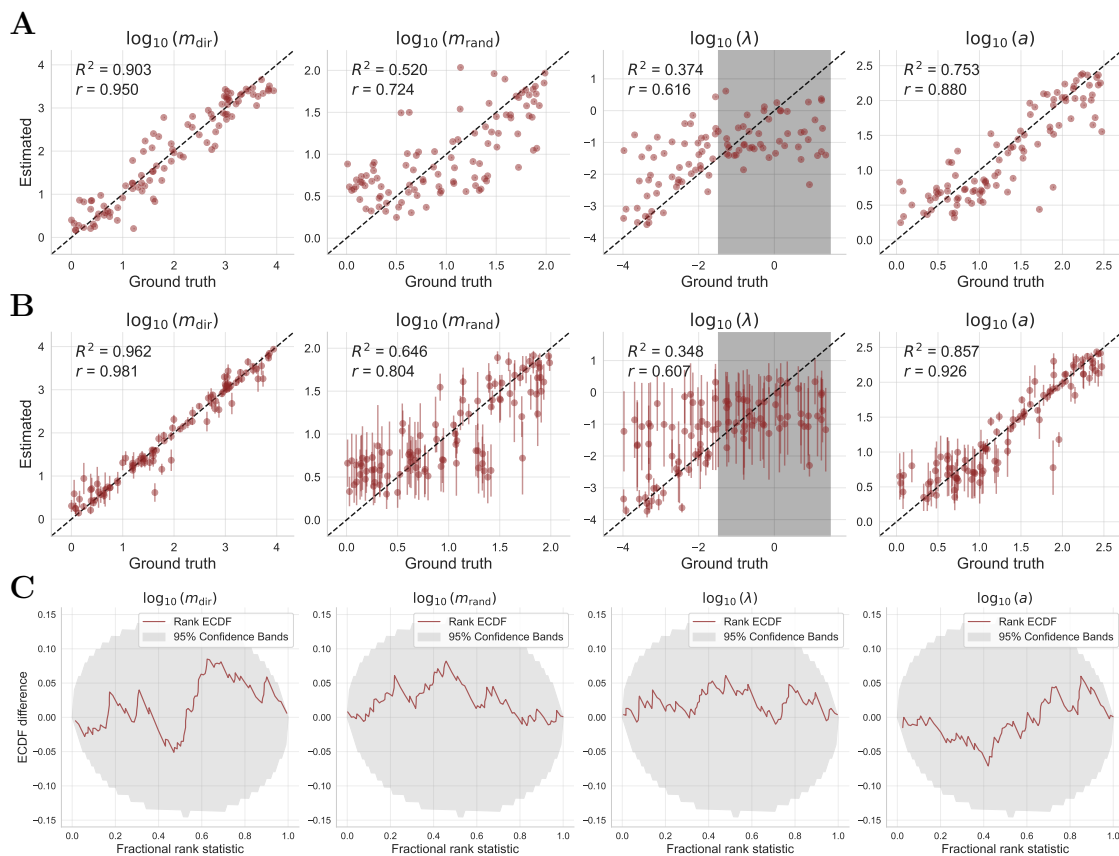
Supplementary Figure 1: *Diagnostics of ABC for each generation of the sampling run on synthetic data set 1.* (A) Diagnostics of ABC with hand-crafted summaries. (B) Diagnostics of ABC-PM with posterior mean summaries. (C) Diagnostics of ABC-NPE with inference-tailored summaries. (D) Summary statistics for ABC and NPE.



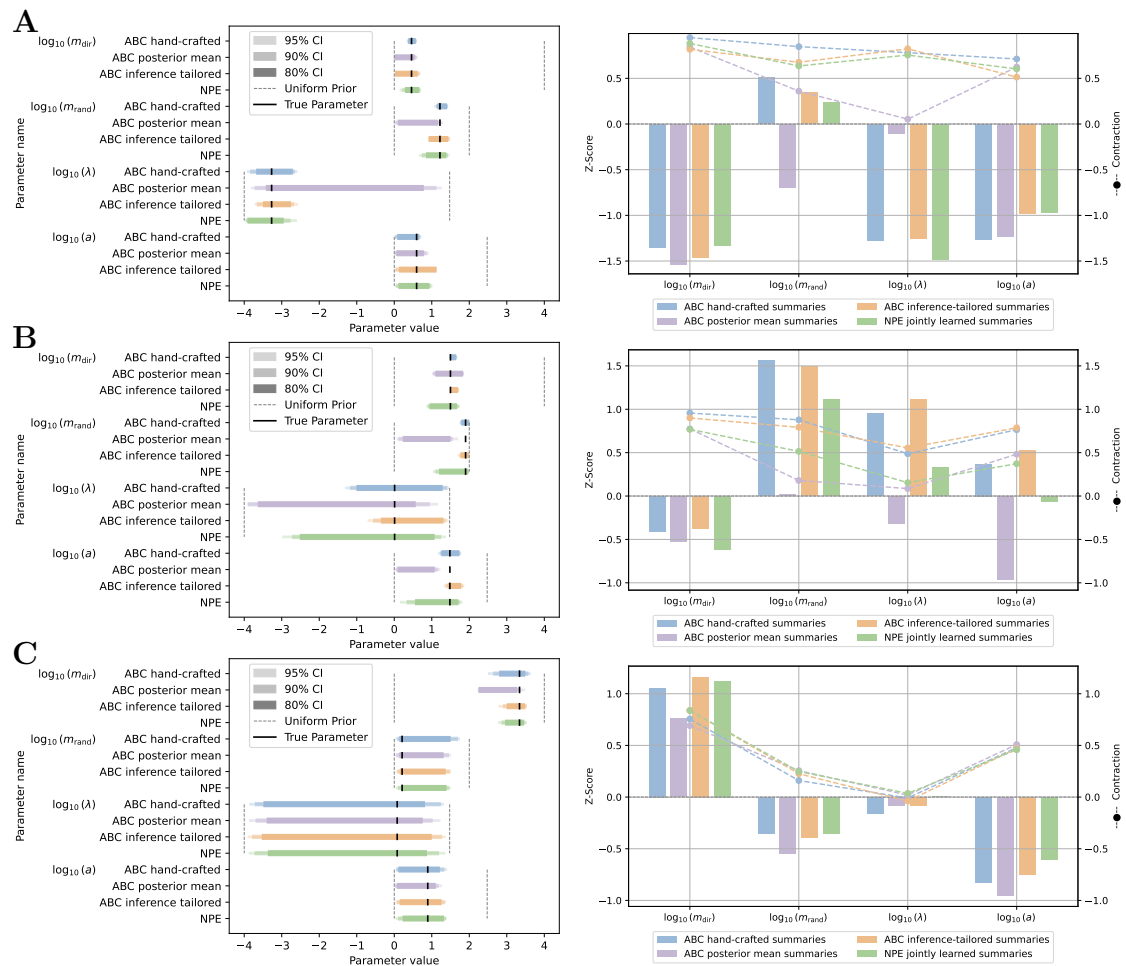
Supplementary Figure 2: *Diagnostics of ABC for each generation of the sampling run on synthetic data set 2.* (A) Diagnostics of ABC with hand-crafted summaries. (B) Diagnostics of ABC-PM with posterior mean summaries. (C) Diagnostics of ABC-NPE with inference-tailored summaries. (D) Summary statistics for ABC and NPE.



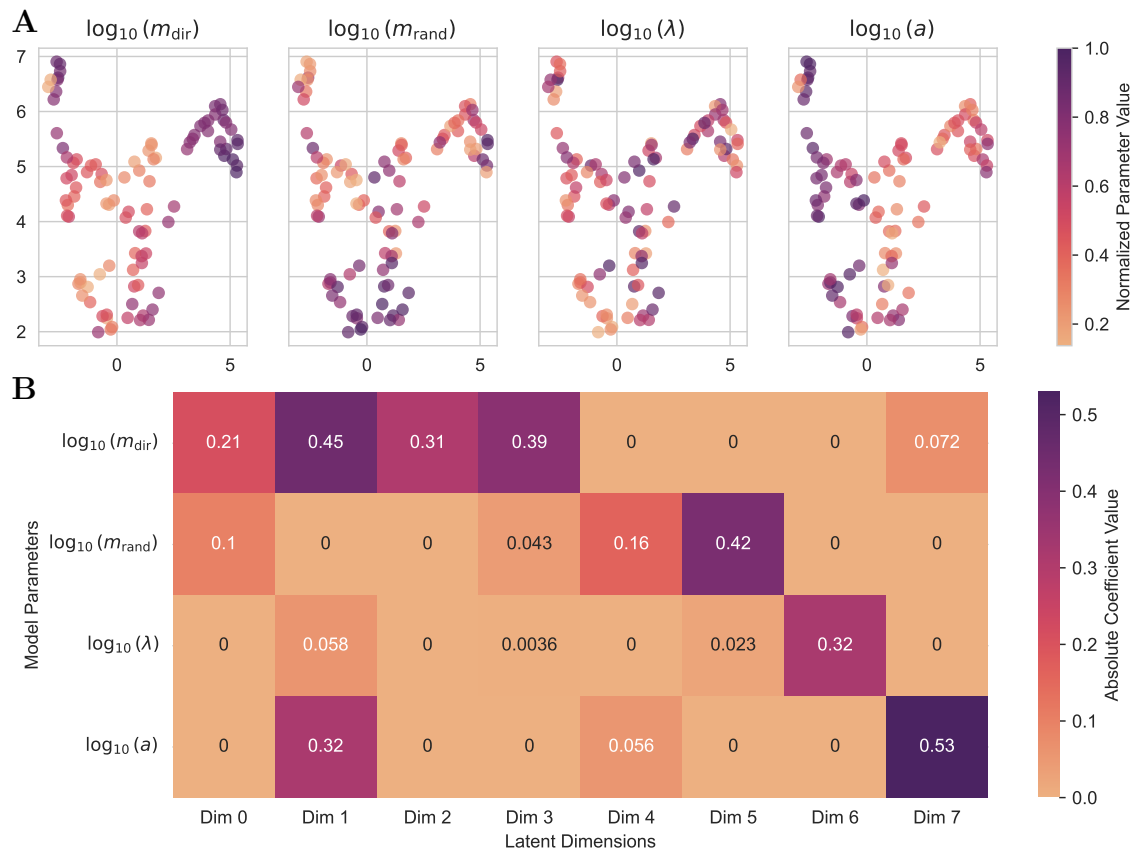
Supplementary Figure 3: *Diagnostics of ABC for each generation of the sampling run on synthetic data set 3. (A) Diagnostics of ABC with hand-crafted summaries. (B) Diagnostics of ABC-PM with posterior mean summaries. (C) Diagnostics of ABC-NPE with inference-tailored summaries. (D) Summary statistics for ABC and NPE.*



Supplementary Figure 4: *Recovery and empirical cumulative distribution plots.* (A) Recovery of the parameters by the neural network trained on the posterior mean. The recovery plot displays the posterior median along with the median absolute deviation (if possible). The grey area in the plot of the rate parameter λ indicates the area where the mean waiting time $1/\lambda$ is smaller than the observation time frame of 30 s. (B) Recovery of the parameters by the neural posterior estimator. (C) The difference between the empirical cumulative distribution (ECDF) of the rank of the true parameter and a uniform ECDF is shown. To assess the validity of the inferred posterior distributions, we employed simulation-based calibration (SBC) [88, 89]. SBC leverages the principle that, under a well-calibrated posterior, the rank of the true parameter within the posterior samples should follow a uniform distribution.

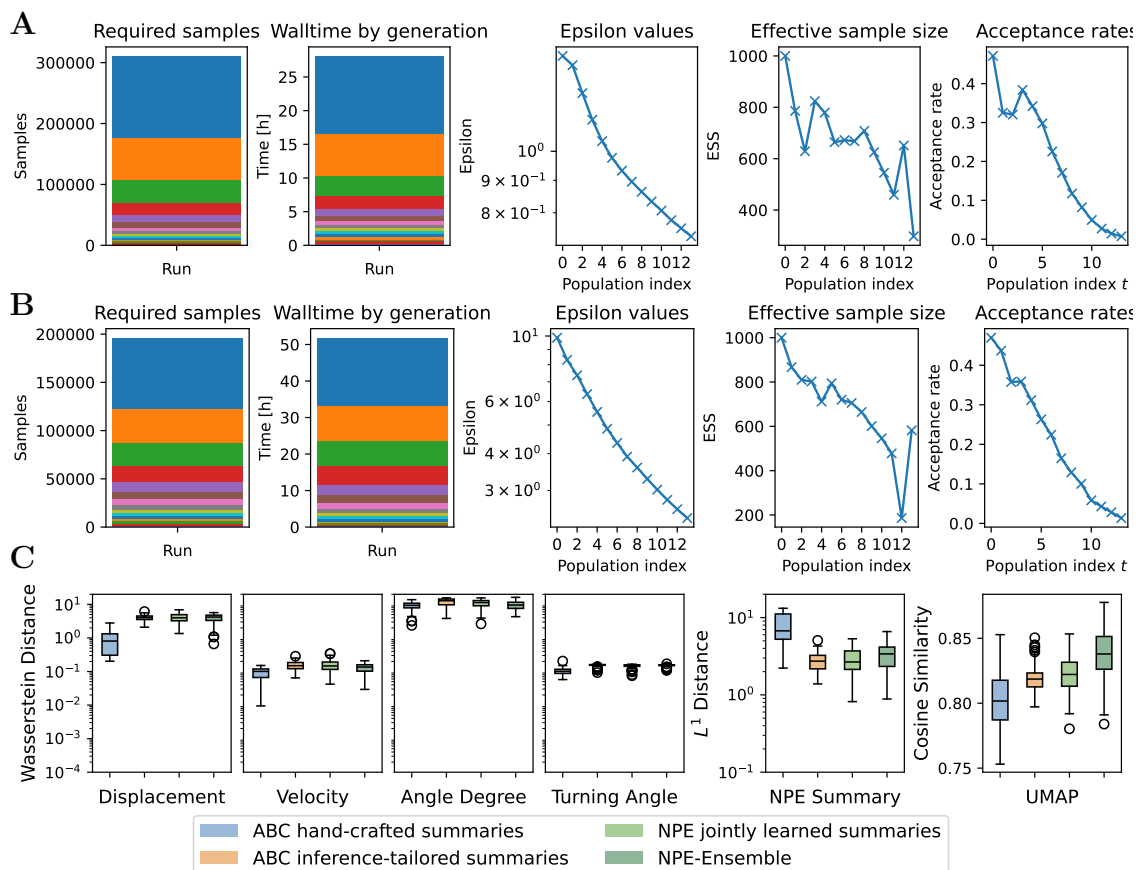


Supplementary Figure 5: *Credible intervals (left) and contraction (right) of the posteriors from the different approaches for the synthetic test data. (A) Dataset 1. (B) Dataset 2. (C) Dataset 3.*



Supplementary Figure 6: *Interpretability of the learned summary space*. **(A)** UMAP representation of the learned summary space trained jointly with NPE. **(B)** LASSO-regression ($\alpha = 0.1$) of true parameter values onto the simulation summary space representation reveals that half of the latent dimensions primarily focus on the influence strength of chemokine attraction m_{dir} , with varying degrees of influence on others.

Additional results for the experimental dataset



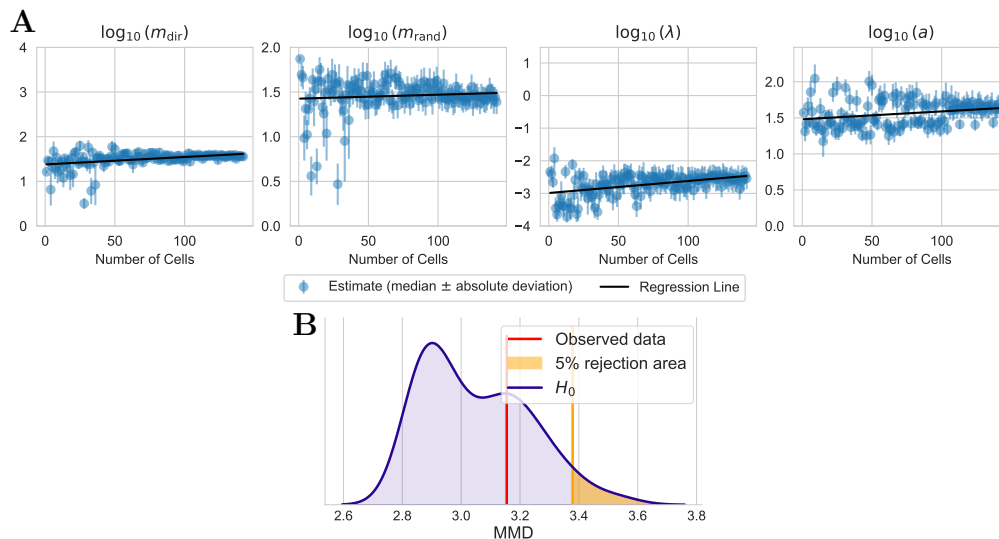
Supplementary Figure 7: *Diagnostics of ABC for each generation of the sampling run on the empirical data.* (A) Diagnostics for ABC with hand-crafted summaries on experimental data. (B) Diagnostics for ABC-NPE with learned summaries on experimental data. (C) Summary statistics for ABC and NPE.

Instead of analyzing the full dataset, we can exploit the amortization feature of the neural posterior estimator to estimate the posteriors for truncated datasets, as our summary network accommodates a variable number of cells. We increased the total number of cells in the experiment by randomly subsampling the dataset. Our results were robust to the number of cells used in the experiment (Supplementary Figure 8A).

We can use the learned summary network to compare the summaries of the experimental data with those from the simulations observed during training. During training, we enforced the inlier summary distribution to be Gaussian. Then, misspecification can be detected by a distribution mismatch in the summary space with a high maximum mean discrepancy (MMD), as proposed by [74]. The reference distribution was constructed using the validation data. However, in this case, we cannot reject the hypothesis that the experimental data are consistent with the normal distribution, meaning that no misspecification was detected (see Supplementary Figure 8B).

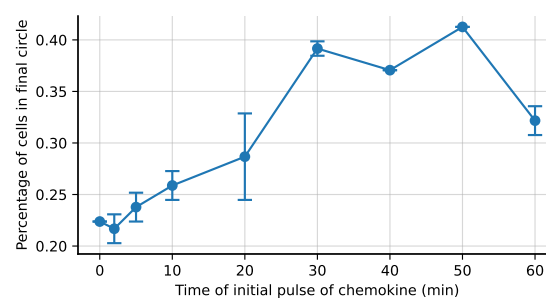
Additionally, we compute a matrix $K \in \mathbb{R}^{3 \times 3}$ of pairwise Kullback–Leibler divergences between the posteriors inferred by each of the 3 models in the ensemble,

as suggested by [85]. Each entry K_{ij} is estimated by averaging the log-density ratio between models i and j 's posteriors over samples from model i . The magnitude of the entries in K serves as a diagnostic tool for ensemble consistency, with small values indicating similar predictive behavior. A heuristic condition for consistency is the normalized maximum divergence $\max_{i,j}(K_{ij}/d) \ll 1$, where d is the dimensionality of the parameter space. In our case, the maximum normalized value is 1.01 for 100 posterior samples, indicating rather high variability between the ensemble members, but no clear evidence of inconsistency within the ensemble.



Supplementary Figure 8: *Additional robustness checks.* (A) Impact of the number of observed cells on the estimated parameters of NPE. (B) Model misspecification test using NPE.

Additional results for the simulation study



Supplementary Figure 9: *Initial pulse of the chemokine influences trajectory.* Median and absolute deviation of the percentage over three simulations from the posterior median.

Bibliography

- [1] B. Ingalls, *Mathematical modelling in systems biology: An introduction*, MIT Press, 2013 (cit. on pp. 1, 8, 12).
- [2] S.-H. Kang, S. Kahan, J. McDermott, N. Flann and I. Shmulevich, *Biocellion: Accelerating computer simulation of multicellular biological system models*, *Bioinformatics* **30** (2014) 3101 (cit. on p. 1).
- [3] S. Hoehme and D. Drasdo, *A cell-based simulation software for multi-cellular systems*, *Bioinformatics* **26** (2010) 2641, URL: <https://doi.org/10.1093/bioinformatics/btq437> (cit. on p. 1).
- [4] F. Cooper et al., *Chaste: Cancer, heart and soft tissue environment*, *Journal of Open Source Software* **5** (2020) 1848 (cit. on p. 1).
- [5] M. H. Swat et al., *Multi-scale modeling of tissues using CompuCell3D*, *Methods in cell biology* **110** (2012) 325 (cit. on pp. 1, 2).
- [6] J. Starrauß, W. de Back, L. Brusch and A. Deutsch, *Morpheus: A user-friendly modeling environment for multiscale and multicellular systems biology*, *Bioinf.* **30** (2014) 1331 (cit. on pp. 1, 2, 14).
- [7] A. Ghaffarizadeh, R. Heiland, S. Friedman, S. Mumenthaler and P. Macklin, *PhysiCell: An open source physics-based cell simulator for 3-D multicellular systems*, *PLoS Computational Biology* **14** (2018) e1005991 (cit. on pp. 1, 2).
- [8] H. B. Wolff, L. A. Davidson and R. M. H. Merks, *Adapting a plant tissue model to animal development: Introducing cell sliding into VirtualLeaf*, *Bulletin of Mathematical Biology* **81** (2019) 3322 (cit. on p. 1).
- [9] A. Montagud, M. Ponce-de-Leon and A. Valencia, *Systems biology at the giga-scale: Large multiscale models of complex, heterogeneous multicellular systems*, *Current Opinion in Systems Biology* **28** (2021) 100385 (cit. on p. 3).
- [10] R. Eftimie, *Grand challenges in mathematical biology: Integrating multi-scale modeling and data*, 2022 (cit. on p. 3).
- [11] K. M. Yamada and M. Sixt, *Mechanisms of 3D cell migration*, *Nature Reviews molecular cell biology* **20** (2019) 738 (cit. on p. 2).

-
- [12] F. Graner and J. Glazier, *Simulation of biological cell sorting using a two-dimensional extended Potts model*, *Phys. Rev. Lett.* **69** (1992) 2033 (cit. on pp. 2, 10).
- [13] G. R. Mirams et al., *Chaste: An open source C++ library for computational physiology and biology*, *PLoS Comput. Biol.* **9** (2013) e1002970 (cit. on p. 2).
- [14] P. Nanda and D. E. Kirschner, *Calibration methods to fit parameters within complex biological models*, *Frontiers in applied mathematics and statistics* **9** (2023) 1256443 (cit. on p. 2).
- [15] T. Toni, D. Welch, N. Strelkowa, A. Ipsen and M. P. H. Stumpf, *Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems*, *J. R. Soc. Interface* **6** (2009) 187 (cit. on pp. 2, 19).
- [16] N. Jagiella, D. Rickert, F. J. Theis and J. Hasenauer, *Parallelization and High-Performance Computing Enables Automated Statistical Inference of Multi-scale Models*, *Cell Syst.* **4** (2017) 194 (cit. on pp. 2, 21).
- [17] A. G. Fletcher and J. M. Osborne, *Seven challenges in the multiscale modeling of multicellular tissues*, *WIREs mechanisms of disease* **14** (2022) e1527 (cit. on pp. 2, 42).
- [18] C. M. Hazelbag, J. Dushoff, E. M. Dominic, Z. E. Mthombathi and W. Delva, *Calibration of individual-based models to epidemiological data: A systematic review*, *PLOS Computational Biology* **16** (2020) 1, URL: <https://doi.org/10.1371/journal.pcbi.1007893> (cit. on p. 2).
- [19] P. Macklin, *Key challenges facing data-driven multicellular systems biology*, *Gigascience* **8** (2019) giz127 (cit. on p. 3).
- [20] J. Karr et al., *Model integration in computational biology: the role of reproducibility, credibility and utility*, *Frontiers in systems biology* **2** (2022) 822606 (cit. on p. 3).
- [21] H. Pesonen et al., *ABC of the future*, *International Statistical Review* **91** (2023) 243, eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/insr.12522>, URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/insr.12522> (cit. on p. 3).
- [22] S. Höpfl, J. Pleiss and N. E. Radde, *Bayesian estimation reveals that reproducible models in Systems Biology get more citations*, *Scientific Reports* **13** (2023) 2695 (cit. on p. 3).
- [23] E. Alamoudi et al., *FitMultiCell: Simulating and parameterizing computational models of multi-scale and multi-cellular processes*, *bioRxiv* (2023) 2023 (cit. on p. 5).
- [24] E. Alamoudi et al., *A Wall-time Minimizing Parallelization Strategy for Approximate Bayesian Computation*, *arXiv preprint arXiv:2305.00506* (2023) (cit. on pp. 5, 20).

-
- [25] J. Arruda et al., *Simulation-Based Inference of Cell Migration Dynamics in Complex Spatial Environments*, bioRxiv (2025), eprint: <https://www.biorxiv.org/content/early/2025/09/25/2025.09.25.678460.full.pdf>, URL: <https://doi.org/10.1101/2025.09.25.678460> (cit. on pp. 5, 10, 109).
- [26] E. Alamoudi et al., *FitMultiCell: simulating and parameterizing computational models of multi-scale and multi-cellular processes*, *Bioinformatics* **39** (2023) btad674, ISSN: 1367-4811, eprint: <https://academic.oup.com/bioinformatics/article-pdf/39/11/btad674/53689329/btad674.pdf>, URL: <https://doi.org/10.1093/bioinformatics/btad674> (cit. on pp. 6, 45).
- [27] E. Klinger, D. Rickert and J. Hasenauer, *pyABC: distributed, likelihood-free inference*, *Bioinf.* **34** (2018) 3591 (cit. on pp. 6, 20, 21).
- [28] Y. Schälte, E. Klinger, E. Alamoudi and J. Hasenauer, *pyABC: Efficient and robust easy-to-use approximate Bayesian computation*, *J. Open Source Softw.* **7** (2022) 4304, URL: <https://doi.org/10.21105/joss.04304> (cit. on pp. 6, 20).
- [29] D. Machado et al., *Modeling formalisms in systems biology*, *AMB express* **1** (2011) 45 (cit. on p. 8).
- [30] B. L. Puniya, M. Verma, C. Damiani, S. Bakr and A. Dräger, *Perspectives on computational modeling of biological systems and the significance of the SysMod community*, *Bioinformatics Advances* **4** (2024) vbae090, ISSN: 2635-0041, eprint: <https://academic.oup.com/bioinformaticsadvances/article-pdf/4/1/vbae090/58362424/vbae090.pdf>, URL: <https://doi.org/10.1093/bioadv/vbae090> (cit. on p. 8).
- [31] N. ElKalaawy and A. Wassal, *Methodologies for the modeling and simulation of biochemical networks, illustrated for signal transduction pathways: A primer*, *Biosystems* **129** (2015) 1, ISSN: 0303-2647, URL: <https://www.sciencedirect.com/science/article/pii/S030326471500009X> (cit. on p. 8).
- [32] R.-S. Wang, A. Saadatpour and R. Albert, *Boolean modeling in systems biology: an overview of methodology and applications*, *Physical Biology* **9** (2012) 055001, URL: <https://dx.doi.org/10.1088/1478-3975/9/5/055001> (cit. on pp. 8, 9).
- [33] L. Chen, D. Kulasiri and S. Samarasinghe, *A novel data-driven boolean model for genetic regulatory networks*, *Frontiers in physiology* **9** (2018) 1328 (cit. on p. 8).
- [34] S. Kauffman, *Metabolic stability and epigenesis in randomly constructed genetic nets*, *Journal of Theoretical Biology* **22** (1969) 437, ISSN: 0022-5193, URL: <https://www.sciencedirect.com/science/article/pii/0022519369900150> (cit. on p. 8).
- [35] R. Thomas, *Boolean formalization of genetic control circuits*, *Journal of theoretical biology* **42** (1973) 563 (cit. on p. 8).

-
- [36] R. Samaga and S. Klamt, *Modeling approaches for qualitative and semi-quantitative analysis of cellular signaling networks*, Cell communication and signaling **11** (2013) 1 (cit. on p. 9).
 - [37] N. Le Novère, *Quantitative and logic modelling of molecular and gene networks*, Nature Reviews Genetics **16** (2015) 146 (cit. on p. 9).
 - [38] S. Li, S. M. Assmann and R. Albert, *Predicting essential components of signal transduction networks: a dynamic model of guard cell abscisic acid signaling*, PLoS biology **4** (2006) e312 (cit. on p. 9).
 - [39] E. R. Walsh et al., *Computational and experimental analysis reveals a requirement for eosinophil-derived IL-13 for the development of allergic airway responses in C57BL/6 mice*, The Journal of Immunology **186** (2011) 2936 (cit. on p. 9).
 - [40] A. Saadatpour et al., *Dynamical and structural analysis of a T cell survival network identifies novel candidate therapeutic targets for large granular lymphocyte leukemia*, PLoS computational biology **7** (2011) e1002267 (cit. on p. 9).
 - [41] A. Fauré, B. M. Vreede, É. Sucena and C. Chaouiya, *A discrete model of drosophila eggshell patterning reveals cell-autonomous and juxtacrine effects*, PLoS computational biology **10** (2014) e1003527 (cit. on p. 9).
 - [42] N. Bonzanni et al., *Hard-wired heterogeneity in blood stem cells revealed using a dynamic regulatory network model*, Bioinformatics **29** (2013) i80 (cit. on p. 9).
 - [43] D. P. Cohen et al., *Mathematical modelling of molecular pathways enabling tumour cell invasion and migration*, PLoS computational biology **11** (2015) e1004571 (cit. on p. 9).
 - [44] J. D. Schwab, S. D. Kühlwein, N. Ikonomi, M. Kühl and H. A. Kestler, *Concepts in Boolean network modeling: What do they all mean?*, Computational and structural biotechnology journal **18** (2020) 571 (cit. on p. 9).
 - [45] K. C. Clarke, “Cellular automata and agent-based models”, *Handbook of regional science*, Springer, 2021 1751 (cit. on p. 9).
 - [46] D. Bonchev, S. Thomas, A. Apte and L. B. Kier, *Cellular automata modelling of biomolecular networks dynamics*, SAR and QSAR in Environmental Research **21** (2010) 77 (cit. on p. 9).
 - [47] J. Poleszczuk and H. Enderling, *A high-performance cellular automaton model of tumor growth with dynamically growing domains*, Applied mathematics **5** (2014) 144 (cit. on p. 9).
 - [48] C. Beauchemin, J. Samuel and J. Tuszynski, *A simple cellular automaton model for influenza A viral infections*, Journal of theoretical biology **232** (2005) 223 (cit. on p. 9).
 - [49] L. Manukyan, S. A. Montandon, A. Fofonjka, S. Smirnov and M. C. Milinkovitch, *A living mesoscopic cellular automaton made of skin scales*, Nature **544** (2017) 173 (cit. on p. 9).
 - [50] J. A. Glazier and F. Graner, *Simulation of the differential adhesion driven rearrangement of biological cells*, Physical Review E **47** (1993) 2128 (cit. on p. 10).

-
- [51] T. Hirashima, E. G. Rens and R. M. Merks, *Cellular Potts modeling of complex multicellular behaviors in tissue morphogenesis*, Development, growth & differentiation **59** (2017) 329 (cit. on p. 10).
- [52] R. M. Merks, S. V. Brodsky, M. S. Goligorsky, S. A. Newman and J. A. Glazier, *Cell elongation is key to in silico replication of in vitro vasculogenesis and subsequent remodeling*, Developmental biology **289** (2006) 44 (cit. on p. 10).
- [53] D. Svoboda et al., “Vascular network formation in silico using the extended cellular potts model”, *2016 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2016 3180 (cit. on p. 10).
- [54] M. Scianna, L. Preziosi, K. Wolf et al., *A Cellular Potts Model simulating cell migration on and in matrix environments*, Math Biosci Eng **10** (2013) 235 (cit. on p. 10).
- [55] N. J. Savill and P. Hogeweg, *Modelling morphogenesis: from single cells to crawling slugs*, Journal of theoretical biology **184** (1997) 229 (cit. on p. 10).
- [56] R. Chaturvedi et al., *On multiscale approaches to three-dimensional modelling of morphogenesis*, Journal of the Royal Society interface **2** (2005) 237 (cit. on p. 10).
- [57] A. Szabó and R. M. Merks, *Cellular potts modeling of tumor growth, tumor invasion, and tumor evolution*, Frontiers in oncology **3** (2013) 87 (cit. on p. 10).
- [58] S. E. Boas, Y. Jiang, R. M. Merks, S. A. Prokopiou and E. G. Rens, “Cellular potts model: applications to vasculogenesis and angiogenesis”, *Probabilistic Cellular Automata: Theory, Applications and Future Perspectives*, Springer, 2018 279 (cit. on p. 10).
- [59] A. T. Crooks and A. J. Heppenstall, “Introduction to agent-based modelling”, *Agent-based models of geographical systems*, Springer, 2011 85 (cit. on pp. 10, 11).
- [60] N. Gilbert, *Agent-based models*, Sage Publications, 2019 (cit. on p. 10).
- [61] L. Breitwieser et al., *BioDynaMo: a modular platform for high-performance agent-based simulation*, Bioinformatics **38** (2022) 453 (cit. on p. 10).
- [62] G. Datseris, A. R. Vahdati and T. C. DuBois, *Agents.jl: a performant and feature-full agent-based modeling software of minimal code complexity*, SIMULATION **0** (2022) 003754972110688, URL: <https://doi.org/10.1177/00375497211068820> (cit. on p. 10).
- [63] P. Taillandier, D.-A. Vo, E. Amouroux and A. Drogoul, “GAMA: a simulation platform that integrates geographical information data, agent-based modeling and multi-scale control”, *Principles and Practice of Multi-Agent Systems: 13th International Conference, PRIMA 2010, Kolkata, India, November 12-15, 2010, Revised Selected Papers 13*, Springer, 2012 242 (cit. on p. 10).

-
- [64] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan and G. Balan, *Mason: A multiagent simulation environment*, *Simulation* **81** (2005) 517 (cit. on p. 10).
- [65] D. Masad, J. L. Kazil et al., “Mesa: An Agent-Based Modeling Framework.”, *SciPy*, Citeseer, 2015 51 (cit. on p. 10).
- [66] A. Antelmi, G. Cordasco, G. D’Ambrosio, D. De Vinco and C. Spagnuolo, *Experimenting with agent-based model simulation tools*, *Applied Sciences* **13** (2022) 13 (cit. on p. 10).
- [67] S. B. Shinde and M. P. Kurhekar, *Review of the systems biology of the immune system using agent-based models*, *IET Systems Biology* **12** (2018) 83, eprint: <https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-syb.2017.0073>, URL: <https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-syb.2017.0073> (cit. on p. 10).
- [68] P. Kumberger, K. Durso-Cain, S. Uprichard, H. Dahari and F. Graw, *Accounting for Space—Quantification of Cell-To-Cell Transmission Kinetics Using Virus Dynamics Models*, *Viruses* **10** (2018) 200, ISSN: 1999-4915, URL: <http://dx.doi.org/10.3390/v10040200> (cit. on p. 11).
- [69] Z. Wang, J. D. Butner, R. Kerketta, V. Cristini and T. S. Deisboeck, *Simulating cancer growth with multiscale agent-based modeling*, *Seminars in Cancer Biology* **30** (2015) 70, Cancer modeling and network biology, ISSN: 1044-579X, URL: <https://www.sciencedirect.com/science/article/pii/S1044579X14000492> (cit. on p. 11).
- [70] C. C. Kerr et al., *Covasim: An agent-based model of COVID-19 dynamics and interventions*, *PLOS Computational Biology* **17** (2021) 1, URL: <https://doi.org/10.1371/journal.pcbi.1009149> (cit. on p. 11).
- [71] L. Michaelis and M. Menten, *Die Kinetik der Invertinwirkung*, *Biochem. Z.* **49** (1913) 333 (cit. on p. 11).
- [72] E. L. Haseltine and J. B. Rawlings, *Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics*, *J. Chem. Phys.* **117** (2002) 6959 (cit. on p. 11).
- [73] S. Rüdiger et al., *Hybrid stochastic and deterministic simulations of calcium blips*, *Biophysical journal* **93** (2007) 1847 (cit. on p. 11).
- [74] D. J. Wilkinson, *Stochastic modelling for systems biology*, CRC press, 2011 (cit. on p. 12).
- [75] D. T. Gillespie, *A rigorous derivation of the chemical master equation*, *Physica A: Statistical Mechanics and its Applications* **188** (1992) 404 (cit. on p. 12).
- [76] E. Klipp, R. Herwig, A. Kowald, C. Wierling and H. Lehrach, *Systems biology in practice*, Wiley-VCH, Weinheim, 2005, ISBN: 978-3-527-31078-4 (cit. on p. 12).

-
- [77] D. Lunz, G. Batt, J. Ruess and J. F. Bonnans, *Beyond the chemical master equation: Stochastic chemical kinetics coupled with auxiliary processes*, PLoS Computational Biology **17** (2021) e1009214 (cit. on p. 12).
- [78] T. E. Turner, S. Schnell and K. Burrage, *Stochastic approaches for modelling in vivo reactions*, Comput. Biol. Chem. **28** (2004) 165 (cit. on p. 12).
- [79] H. Enderling, M. A. Chaplain, A. R. Anderson and J. S. Vaidya, *A mathematical model of breast cancer development, local treatment and recurrence*, Journal of theoretical biology **246** (2007) 245 (cit. on p. 13).
- [80] K. J. Painter, *Continuous models for cell migration in tissues and applications to cell sorting via differential chemotaxis*, Bulletin of Mathematical Biology **71** (2009) 1117 (cit. on p. 13).
- [81] R. Giniūnaitė, R. E. Baker, P. M. Kulesa and P. K. Maini, *Modelling collective cell migration: neural crest as a model paradigm*, Journal of Mathematical Biology **80** (2020) 481 (cit. on p. 13).
- [82] S. Ganesan and D. Subramani, *Spatio-temporal predictive modeling framework for infectious disease spread*, Scientific Reports **11** (2021) 6741 (cit. on p. 13).
- [83] J. P. Keener, *Biology in time and space: a partial differential equation modeling approach*, vol. 50, American Mathematical Soc., 2021 (cit. on p. 13).
- [84] S. SenGupta, C. A. Parent and J. E. Bear, *The principles of directed cell migration*, Nature Reviews Molecular Cell Biology **22** (2021) 529 (cit. on p. 13).
- [85] M. Scianna and L. Preziosi, *A Cellular Potts Model for Analyzing Cell Migration across Constraining Pillar Arrays*, Axioms **10** (2021), ISSN: 2075-1680, URL: <https://www.mdpi.com/2075-1680/10/1/32> (cit. on p. 14).
- [86] D. Walker, G. Hill, S. Wood, R. Smallwood and J. Southgate, *Agent-based computational modeling of wounded epithelial cell monolayers*, IEEE Transactions on NanoBioscience **3** (2004) 153 (cit. on p. 14).
- [87] A. Köhn-Luque et al., *Early Embryonic Vascular Patterning by Matrix-Mediated Paracrine Signalling: A Mathematical Model Study*, PLOS ONE **6** (2011) 1, URL: <https://doi.org/10.1371/journal.pone.0024175> (cit. on p. 15).
- [88] A. Imle et al., *Experimental and computational analyses reveal that environmental restrictions shape HIV-1 spread in 3D cultures*, Nat. Commun. **10** (2019) 2144 (cit. on p. 15).
- [89] H. T.-K. Vu et al., *Dynamic Polarization of the Multiciliated Planarian Epidermis between Body Plan Landmarks*, Dev. Cell **51** (2019) 526, ISSN: 1534-5807, URL: <http://www.sciencedirect.com/science/article/pii/S153458071930869X> (cit. on p. 15).
- [90] G. M. Martin, D. T. Frazier and C. P. Robert, *Computing Bayes: Bayesian computation from 1763 to the 21st century*, arXiv preprint arXiv:2004.06425 (2020) (cit. on p. 15).

-
- [91] S. Martina Perez, H. Sailem and R. E. Baker, *Efficient Bayesian inference for mechanistic modelling with high-throughput data*, *PLOS Computational Biology* **18** (2022) 1, URL: <https://doi.org/10.1371/journal.pcbi.1010191> (cit. on p. 15).
- [92] A. Doucet, N. de Freitas and N. Gordon, “An Introduction to Sequential Monte Carlo Methods”, *Sequential Monte Carlo Methods in Practice*, ed. by A. Doucet, N. de Freitas and N. Gordon, New York, NY: Springer New York, 2001 3, ISBN: 978-1-4757-3437-9, URL: https://doi.org/10.1007/978-1-4757-3437-9_1 (cit. on p. 16).
- [93] P. Del Moral, A. Doucet and A. Jasra, *Sequential Monte Carlo Samplers*, *Journal of the Royal Statistical Society Series B: Statistical Methodology* **68** (2006) 411, ISSN: 1369-7412, eprint: https://academic.oup.com/jrsssb/article-pdf/68/3/411/49795343/jrsssb_68_3_411.pdf, URL: <https://doi.org/10.1111/j.1467-9868.2006.00553.x> (cit. on p. 16).
- [94] G. E. Hinton, *Training Products of Experts by Minimizing Contrastive Divergence*, *Neural Computation* **14** (2002) 1771, ISSN: 0899-7667, eprint: <https://direct.mit.edu/neco/article-pdf/14/8/1771/815447/089976602760128018.pdf>, URL: <https://doi.org/10.1162/089976602760128018> (cit. on p. 17).
- [95] M. U. Gutmann and A. Hyvärinen, *Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics*, *Journal of Machine Learning Research* **13** (2012) 307, URL: <http://jmlr.org/papers/v13/gutmann12a.html> (cit. on p. 17).
- [96] M. A. Beaumont, W. Zhang and D. J. Balding, *Approximate Bayesian Computation in Population Genetics*, *Genetics* **162** (2002) 2025 (cit. on p. 17).
- [97] P. Marjoram, J. Molitor, V. Plagnol and S. Tavaré, *Markov chain Monte Carlo without likelihoods*, *Proc. Natl. Acad. Sci. USA* **100** (2003) 15324 (cit. on pp. 17–19).
- [98] S. A. Sisson, Y. Fan and M. M. Tanaka, *Sequential Monte Carlo without likelihoods*, *Proc. Natl. Acad. Sci.* **104** (2007) 1760 (cit. on pp. 17, 19).
- [99] C. Gourieroux, A. Monfort and E. Renault, *Indirect inference*, *J. Appl. Econ.* **8** (1993) S85 (cit. on p. 17).
- [100] S. N. Wood, *Statistical inference for noisy nonlinear ecological dynamic systems*, *Nature* **466** (2010) 1102 (cit. on p. 17).
- [101] O. Thomas, R. Dutta, J. Corander, S. Kaski and M. U. Gutmann, *Likelihood-free inference by ratio estimation*, *Bayesian Analysis* **17** (2022) 1 (cit. on p. 17).
- [102] C. Drovandi and D. T. Frazier, *A comparison of likelihood-free methods with and without summary statistics*, *Statistics and Computing* **32** (2022) 42 (cit. on p. 17).

-
- [103] S. A. Sisson, Y. Fan and M. Beaumont, *Handbook of approximate Bayesian computation*, Chapman and Hall/CRC, 2018 (cit. on p. 17).
- [104] P. Joyce and P. Marjoram, *Approximately sufficient statistics and Bayesian computation*, Stat. appl. in genetics and mol. bio. **7** (2008) (cit. on p. 17).
- [105] P. Fearnhead and D. Prangle, *Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation*, J. R. Stat. Soc. B **74** (2012) 419 (cit. on p. 17).
- [106] J. K. Pritchard, M. T. Seielstad, A. Perez-Lezaun and M. W. Feldman, *Population growth of human Y chromosomes: a study of Y chromosome microsatellites.*, Mol. Biol. Evol. **16** (1999) 1791 (cit. on p. 18).
- [107] M. A. Beaumont, *Approximate Bayesian computation in evolution and ecology*, Annual review of ecology, evolution, and systematics **41** (2010) 379 (cit. on p. 19).
- [108] Y. Schälte, E. Klinger, E. Alamoudi and J. Hasenauer, *pyABC: Efficient and robust easy-to-use approximate Bayesian computation*, arXiv preprint arXiv:2203.13043 (2022) (cit. on p. 20).
- [109] T. Toni and M. P. H. Stumpf, *Simulation-based model selection for dynamical systems in systems and population biology*, Bioinf. **26** (2010) 104 (cit. on p. 20).
- [110] E. E. O. Ishida et al., *COSMOABC: Likelihood-free inference via Population Monte Carlo Approximate Bayesian Computation*, Astronomy and Computing **13** (2015) 1 (cit. on p. 20).
- [111] E. Jennings and M. Madigan, *astroABC: an approximate Bayesian computation sequential Monte Carlo sampler for cosmological parameter estimation*, Astron. Comput. **19** (2017) 16 (cit. on p. 20).
- [112] A. H. Stram, P. Marjoram and G. K. Chen, *al3c: high-performance software for parameter inference using Approximate Bayesian Computation*, Bioinformatics **31** (2015) 3549 (cit. on p. 20).
- [113] C. D. Cantwell et al., *Rethinking multiscale cardiac electrophysiology with machine learning and predictive modelling*, Computers in Biology and Medicine **104** (2019) 339, ISSN: 0010-4825, URL: <http://www.sciencedirect.com/science/article/pii/S0010482518303147> (cit. on p. 21).
- [114] C. Houston, B. Marchand, L. Engelbert and C. D. Cantwell, *Reducing complexity and unidentifiability when modelling human atrial cells*, Philos. Trans. A Math. Phys. Eng. Sci. **378** (2020) 20190339, eprint: <https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.2019.0339>, URL: <https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2019.0339> (cit. on p. 21).
- [115] B. Colom et al., *Mutant clones in normal epithelium outcompete and eliminate emerging tumours*, Nature **598** (2021) 510 (cit. on p. 21).

-
- [116] A. Coulier, S. Hellander and A. Hellander, *A multiscale compartment-based model of stochastic gene regulatory networks using hitting-time analysis*, J. Chem. Phys. **154** (2021) 184105 (cit. on p. 21).
- [117] R. C. Bernardo and J. L. Said, *Towards a model-independent reconstruction approach for late-time Hubble data*, J. Cosmol. Astropart. Phys. **2021** (2021) 027 (cit. on p. 21).
- [118] J. M. Minucci et al., *Inferring pesticide toxicity to honey bees from a field-based feeding study using a colony model and Bayesian inference*, Ecol. Appl. **31** (2021) e02442 (cit. on p. 21).
- [119] L. Dagum and R. Menon, *OpenMP: an industry standard API for shared-memory programming*, IEEE computational science and engineering **5** (1998) 46 (cit. on p. 21).
- [120] E. Gabriel et al., “Open MPI: Goals, concept, and design of a next generation MPI implementation”, *Recent Advances in Parallel Virtual Machine and Message Passing Interface: 11th European PVM/MPI Users’ Group Meeting Budapest, Hungary, September 19-22, 2004. Proceedings 11*, Springer, 2004 97 (cit. on p. 22).
- [121] I. M. Gherman et al., *Bridging the gap between mechanistic biological models and machine learning surrogates*, PLoS Computational Biology **19** (2023) e1010988 (cit. on p. 22).
- [122] L. Cai et al., *Surrogate models based on machine learning methods for parameter estimation of left ventricular myocardium*, Royal Society open science **8** (2021) 201121 (cit. on p. 22).
- [123] M. Renardy, T.-M. Yi, D. Xiu and C.-S. Chou, *Parameter uncertainty quantification using surrogate models applied to a spatial model of yeast mating polarization*, PLoS computational biology **14** (2018) e1006181 (cit. on p. 22).
- [124] H. Gao et al., *Estimating prognosis in patients with acute myocardial infarction using personalized computational heart models*, Scientific reports **7** (2017) 13527 (cit. on p. 22).
- [125] G. Papamakarios and I. Murray, *Fast ϵ -free inference of simulation models with bayesian conditional density estimation*, Advances in neural information processing systems **29** (2016) (cit. on p. 23).
- [126] G. Papamakarios, D. Sterratt and I. Murray, “Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows”, *The 22nd international conference on artificial intelligence and statistics*, PMLR, 2019 837 (cit. on pp. 23, 42).
- [127] J. Hermans, V. Begy and G. Louppe, “Likelihood-free mcmc with amortized approximate ratio estimators”, *International conference on machine learning*, PMLR, 2020 4239 (cit. on pp. 23, 42).
- [128] A. Spurio Mancini, M. M. Docherty, M. A. Price and J. D. McEwen, *Bayesian model comparison for simulation-based inference*, RAS Techniques and Instruments **2** (2023) 710, ISSN: 2752-8200, eprint: <https://academic.oup.com/rasti/article-pdf/2/1/710/53667534/rzad051.pdf>, URL: <https://doi.org/10.1093/rasti/rzad051> (cit. on p. 23).

-
- [129] H. S. Moon, A. Oulasvirta and B. Lee, *Amortized Inference with User Simulations*, *Conference on Human Factors in Computing Systems - Proceedings (2023)* (cit. on p. 23).
- [130] N. J. Stanford et al., “Data management in computational systems biology: exploring standards, tools, databases, and packaging best practices”, *Yeast Systems Biology*, Springer, 2019 285 (cit. on p. 23).
- [131] N. Le Novère et al., *Meeting report from the first meetings of the Computational Modeling in Biology Network (COMBINE)*, *Standards in genomic sciences* **5** (2011) 230 (cit. on p. 24).
- [132] R. Ananthakrishnan and A. Ehrlicher, *The forces behind cell movement*, *International journal of biological sciences* **3** (2007) 303 (cit. on pp. 24, 25).
- [133] M. Abercrombie, *The Croonian Lecture, 1978-The crawling movement of metazoan cells*, *Proceedings of the Royal Society of London. Series B. Biological Sciences* **207** (1980) 129 (cit. on p. 25).
- [134] G. M. Cooper and K. Adams, *The cell: a molecular approach*, Oxford University Press, 2022 (cit. on p. 25).
- [135] H.-C. Chen, *Boyden chamber assay*, *Cell migration: developmental methods and protocols* (2005) 15 (cit. on p. 25).
- [136] D. G. Wilkinson, J. Wilson and B. D. Hames, *Synthesis of spore proteins during development of Dictyostelium discoideum*, *Biochemical Journal* **216** (1983) 567 (cit. on p. 25).
- [137] C. A. Schneider, W. S. Rasband and K. W. Eliceiri, *NIH Image to ImageJ: 25 years of image analysis*, *Nature methods* **9** (2012) 671 (cit. on p. 26).
- [138] J. Schindelin et al., *Fiji: an open-source platform for biological-image analysis*, *Nature methods* **9** (2012) 676 (cit. on p. 26).
- [139] E. Fazeli et al., *Automated cell tracking using StarDist and TrackMate*, *F1000Research* **9** (2020) 1279 (cit. on p. 26).
- [140] J.-B. Lugagne, H. Lin and M. J. Dunlop, *DeLTA: Automated cell segmentation, tracking, and lineage reconstruction using deep learning*, *PLoS computational biology* **16** (2020) e1007673 (cit. on p. 26).
- [141] D. Ershov et al., *TrackMate 7: integrating state-of-the-art segmentation algorithms into tracking pipelines*, *Nature methods* **19** (2022) 829 (cit. on p. 26).
- [142] N. Emami, Z. Sedaei and R. Ferdousi, *Computerized cell tracking: Current methods, tools and challenges*, *Visual Informatics* **5** (2021) 1 (cit. on p. 26).
- [143] G. A. Burger, B. van de Water, S. E. Le Dévédec and J. B. Beltman, *Density-Dependent Migration Characteristics of Cancer Cells Driven by Pseudopod Interaction*, *Frontiers in Cell and Developmental Biology Volume 10 - 2022* (2022), ISSN: 2296-634X, URL: <https://www.frontiersin.org/journals/cell-and-developmental-biology/articles/10.3389/fcell.2022.854721> (cit. on p. 29).

-
- [144] M. N. Akhtar et al., *Developmental beta-cell death orchestrates the islet's inflammatory milieu by regulating immune system crosstalk*, *The EMBO Journal* **44** (2025) 1131, eprint: <https://www.embopress.org/doi/pdf/10.1038/s44318-024-00332-w>, URL: <https://www.embopress.org/doi/abs/10.1038/s44318-024-00332-w> (cit. on p. 29).
- [145] A. Boutillon et al., *Guidance by followers ensures long-range coordination of cell migration through α -catenin mechanoperception*, *Developmental Cell* **57** (2022) 1529 (cit. on p. 29).
- [146] R. Müller et al., *Collective cell migration due to guidance-by-followers is robust to multiple stimuli*, *Frontiers in Applied Mathematics and Statistics* **Volume 9 - 2023** (2023), ISSN: 2297-4687, URL: <https://www.frontiersin.org/journals/applied-mathematics-and-statistics/articles/10.3389/fams.2023.1163583> (cit. on p. 29).
- [147] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed and B. Lakshminarayanan, *Normalizing flows for probabilistic modeling and inference*, *Journal of Machine Learning Research* **22** (2021) 1 (cit. on p. 42).
- [148] D. Greenberg, M. Nonnenmacher and J. Macke, “Automatic posterior transformation for likelihood-free inference”, *International conference on machine learning*, PMLR, 2019 2404 (cit. on p. 42).
- [149] B. K. Miller, A. Cole, G. Louppe and C. Weniger, *Simulation-efficient marginal posterior estimation with swyft: stop wasting your precious time*, arXiv preprint arXiv:2011.13951 (2020) (cit. on p. 42).
- [150] E. Alamoudi et al., *A wall-time minimizing parallelization strategy for approximate Bayesian computation*, *PLOS ONE* **19** (2024) 1, URL: <https://doi.org/10.1371/journal.pone.0294015> (cit. on p. 66).

List of Figures

1.1	Cells work together to make up tissues, which work together to make up organs, which make up organ systems, which work together to make up an organism.	2
1.2	The figure illustrates the multi-scale concept of various biological processes that can take place across different spatial and/or temporal scales, such as the molecular level, cell level, organoid level (10^6 cell), tissue level (10^9 cell), and organ level (10^{12} cell) [9] with a spatial resolution that ranges from nanometers to centimeters and a temporal resolution that varies from seconds to years [10].	3
2.1	A display of the trade-off between the computational difficulty and the attention to detail for different simulation methods	14
2.2	Basic ABC algorithm. Each parameter $\theta \sim \pi(\theta)$ is sampled either from the prior or a proposal distribution. Then, it is used to simulate the process from a black-box simulator. One can add an optional step to use summary statistics, which reduce the size of the simulated data, giving a low-dimensional representation $S(y)$. The distance d between the summary statistics of simulated and observed data is calculated, and the sampled parameters are either accepted or not based on an acceptance threshold ε	18
4.1	Visual representation of the look-ahead scheduling (LA). For each generation, one will sample from a prior $g_1(\theta)$ or proposal distribution $g_t(\theta)$ (upper section). As soon as there are N accepted particles, there will be some idle resources (middle section). Once that happens, new particles will be sampled from a preliminary distribution $\tilde{g}_t(\tilde{\theta})$ that is either based on the N accepted particles or the current generation proposal (last section).	33

List of Algorithms

1	ABC rejection Sampler	18
2	ABC MCMC	19
3	ABC SMC	20