Parallel Model Exploration for Tumor Treatment Simulations

Charilaos Akasiadis[1,+], Miguel Ponce-de-Leon[2,+], Arnau Montagud[2,†], Evangelos Michelioudakis[1,†], Alexia Atsidakou[1,†], Elias Alevizos[1,†], Alexander Artikis[1,‡], Alfonso Valencia[2,‡], and Georgios Paliouras[1,‡]

[1] Institute of Informatics and Telecommunications, NCSR 'Demokritos', Agia Paraskevi, Greece {cakasiadis,vagmcs,aatsidakou,alevizos.elias,a.artikis,paliourg}@iit.demokritos.gr
[2] Life Sciences Department, Barcelona Supercomputing Center, Barcelona, Spain {miguel.ponce,arnau.montagud,alfonso.valencia}@bsc.es
[+,†,‡]Equal author contribution

Abstract. Computational systems and methods are being applied to solve biological problems for many years. Incorporating methods of this kind in the research for cancer treatment and related drug discovery in particular, is shown to be challenging due to the complexity and the dynamic nature of the related factors. Usually, there are two objectives in such settings; first to calibrate the simulators so as to reproduce real-world cases, and second, to search for specific values of the parameter space concerning effective drug treatments. We combine a multi-scale simulator for tumor cell growth and a Genetic Algorithm (GA) as a heuristic search method for finding good parameter configurations in reasonable time. The two modules are integrated into a single workflow that can be executed in a parallel manner on high performance computing infrastructures, since large-scale computational and storage capabilities are necessary in this domain. After using the GA for calibration, our goal is to explore different drug delivery schemes. Among these schemes, we aim to find those that minimize tumor cell size and the probability of emergence of drug resistant cells in the future. Results from experiments on high performance computing infrastructure illustrate the effectiveness and timeliness of the approach.

1 Introduction

Computational systems biology is a research field that combines mathematical and computational models together with molecular data in order to gain a better understanding of various biological systems [17]. Among many different problems, it has also been applied to the study of cancer in an attempt to understand the behavior of tumors and predict treatment effectiveness [13]. The development and evolution of tumor cells is the result of many different interacting processes that occur at different scales, both in terms of time and space. Take as an example the mutations and other DNA alterations that happen at the molecular level and can potentially damage the function of genes [33]. These mutated genes may have an impact on the correct functioning of cell processes, such as signal transduction and gene regulation. In turn, this can lead to the transformation of healthy cells into malignant or tumor cells [12]. As a consequence, modelling and simulating cancer is a challenging problem because of the multi-scale nature of this complex multicellular disease [7]. In parallel, models are critical to interpret experiments and to derive mechanistic explanations that can be translated into new experimentally testable hypotheses [2]. This way, faster and zero risk experiments can be performed in-silico to provide further insight to domain experts.

Focusing on drug treatment exploration, the systems biology community has developed various approaches for predicting novel targets and new drugs that can potentially stop or reduce tumor growth [8]. Each of these approaches may focus on a different cellular process associated with cancer, such as cell signalling [5], or metabolism [34]. However, a known fact of cancer biology is that tumors display a high degree of heterogeneity between the cells they consist of, and that this heterogeneity is strongly related to the capacity of tumors to develop resistance to drugs or other treatments [20,26].

This has motivated the development of multi-scale agent-based models that integrate intracellular models into individual cell agents allowing the simulation of heterogeneous populations and the modelling of how cell variability can affect a treatment outcome [22]. Multi-scale models have been used to simulate the evolution of a tumor by taking into account the molecular details of individual cells and thus are promising tools for performing in-silico experiments [11,18].

Although promising, an issue that needs to be addressed by such multi-scale models is that they have many different parameters (e.g., diffusion constants, rates of different cellular processes) whose values must be properly fine-tuned in order to reproduce biologically plausible simulations that are in-line with real-world outcomes [3]. For a subset of these parameters, it might be possible to find experimental measurements in the literature or in relevant databases. Nonetheless, for most of them, there are no such real-world measurements. Their values must thus be calibrated or inferred using indirect sources of information. A common practice is to employ

optimization methods in order to find the parameters' values that better explain the available experimental observations [32]. However, multi-scale models are complex objects and the estimation of model parameters cannot be done analytically. As a result, the evaluation of a candidate set of parameters' values must be performed, by executing one or more simulations that require significant time and computational resources [25]. The results can then be inspected visually by experts or compared against gold, "ideal" standard simulations, which are already known to be biologically plausible and interesting. Due to the size and complexity of the parameter space, an exhaustive search is prohibitively expensive. Therefore, efficient methods for exploring such complex parameter spaces are required [24].

We have implemented a multi-scale agent-based model to simulate the growth of a tumor spheroid. Our model takes into account the effects of a signalling molecule that binds to cell receptors and can trigger a wide range of different responses [19]. In particular, we are interested in the Tumor Necrosis Factor (TNF), a protein used by the immune system, which can induce death in cancer cells by activating specific downstream signalling pathways, thus restraining the growth of a tumor in a manner similar to what a drug would induce.

Moreover, we implemented a workflow in a High Performance Computing (HPC) cluster for the exploration of the parameter space in order to calibrate the values of the model's unknown parameters. We use an already published set of time series data for cell growth and death as the "gold standard" to guide the search of the unknown parameters. We investigate different distance metrics for comparing the generated simulations against these gold standards, as well as two different search strategies, a plain sweep search method, and an informed search, based on a Genetic Algorithm.

We use the calibrated model and the implemented exploration workflow to investigate drug treatment strategies that minimize tumor size while avoiding the emergence of resistant cells in the long term, i.e., cells that become unresponsive to a drug or signal after being exposed to it for extended periods. For a more in-depth study regarding this process, please refer to Calzone et al. 2010 [6]. By analyzing our results, we can investigate on the effect that drug dosage and frequency have on the overall cell survival. In brief, the contributions of this work are:

1. We incorporate a parallel GA implementation that is capable of exploiting HPC infrastructures for model exploration in tumor treatment simulations. Also, the source-code and datasets used are publicly available for further research and reproducibility.
2. Being equipped with this workflow, we calibrate the current version of a multi-scale agent-based simulator, that is used to conduct in-silico experiments and explore tumor treatments.
3. We combine the calibrated simulator with search methods to discover effective drug treatment configurations, which could be proved truly beneficial for patients.

The remainder of this paper is structured as follows: Section 2 briefly discusses the related work; Section 3 presents the model formulation that the multi-scale simulator is based on. Then, Section 4 introduces the parameter exploration problem that we address, as well as the distributed Genetic Algorithm implementation that is used to solve the exploration problem. Section 5 analyzes the experimental results, while Section 6 concludes and proposes directions for future work.

2 Related Work

Computational tools and methods are being applied extensively to solve problems of biology and chemistry, with Genetic Algorithms and other Machine Learning techniques being successful in a variety of use-cases [21,30]. For instance, Sahlol et al. [29] implemented Neural Networks that had their weights optimized by a GA to characterize genes according to their expression responding to cisplatin-based chemotherapy. In each GA iteration, the approach minimizes the mean squared error that serves as an index of whether the model is capable of predicting gene expression accurately. Other researchers have applied GAs to find beneficial drug configurations for HIV patients, by also taking into account stochasticity in their model [28]. In this case, the cost function minimises the number of virus particles left in the organism.

In a very different domain, King et al. [15] employed a GA to search very large mutant libraries for amino acid sequences that optimize certain desirable biochemical properties, such as the binding affinities for target cell receptors. The authors evaluate candidate sequences by running molecular dynamic simulations, a computational method which allows to estimate binding affinity in-silico. Using their approach, the number of the samples examined is significantly reduced and the quality of the acquired solutions is increased.

GAs and other Machine Learning approaches have been extensively used to explore multi-scale models of multi-cellular systems, such as tumor growth. For example, Jagiella et al. [14] developed a parallel approximate Bayesian computation algorithm to parametrize multi-scale models of cells growing in a dynamically changing 3D nutrient environment. An early rejection mechanism is utilized in their work, i.e. a threshold on the objective function is set to determine if a non-interesting case appears and should be skipped, thus sparing computational resources.

More recently, Ozik et al. have integrated the mechanistic 3D multicellular simulator PhysiCell with the model exploration platform EMEWS [24] and have used it to adaptively sample control parameters that maximize

cancer regression in an agent-based model of immunosurveillance against heterogeneous tumours [23]. EMEWS (Extreme-scale Model Exploration With Swift) is a framework that enables the parallel execution of multiple model exploration tasks. An Active Learning approach is used to explore the parameter space and discover optimal cancer regression regions for the parameters, within a feasible space defined by biological and clinical constraints. To validate the results of active learning, a GA was employed to seek points in the space subject to desired characteristics. The best solutions were then compared with the results of Active Learning, to check if they were indeed located inside the areas of interest [23].
In this paper, we build a multi-scale agent-based model of a tumour spheroid using PhysiCell and provide the cell agents with an intra-cellular signal transduction model. The signalling model is used to compute cell responses to perturbations such as the presence of signalling molecules and drugs, which in our case is the Tumor Necrosis Factor (TNF) protein. We integrate this cancer model into the model exploration framework EMEWS as performed by other works in the literature [24]. However, we develop a different workflow composed of two stages: First, we calibrate the bio-physical parameters of our model. The calibration is performed by employing a GA to find the set of bio-physical parameters that optimize the fitting to selected "gold standards". To measure the distances between simulation time-series and the "gold standards", we employ different distance metrics. In the second stage, we use the calibrated models to explore tumour reduction strategies based on the periodic injection of a signal molecule, the TNF, that is able to induce tumor cell death. This is not a trivial problem since cells exposed to the signal for long periods can develop resistance to the administered drug and evade death, a key behaviour which is captured by our model. Therefore, the proposed method for model exploration searches for TNF supply strategies that maximize tumour regression, while avoiding the emergence of cells resistant to the treatment. As we demonstrate in Sec. 5, the search is significantly improved in terms of consumed time and computational resources when we incorporate a GA instead of performing uninformed exhaustive search.

3 Simulation of the Biology Mechanics

We have implemented a multi-scale model of a 3D tumor spheroid using the PhysiCell framework[11] with PhysiBoSSv2.0 [18].[3] PhysiCell is an open-source physics-based cell simulator for 3D multicellular systems that allows us to study many interacting cells in dynamic tissue microenvironments [11]. Cell mechanics are simulated using classic mechanical equations with default parameters from PhysiCell. The microenvironment is modelled using BioFVM [10], a solver for partial differential equations that can efficiently simulate key cell processes, such as secretion, diffusion, uptake, and decay of multiple substrates in large 3D domains. On the other hand, PhysiBoSSv2 is an extension of PhysiCell that enhances the modelling capabilities by allowing simulations of intracellular signal transduction models within each individual cell-agent.
Figure 1 shows that our model represents a tumor spheroid composed by different cells that are set in a defined microenvironment. The microenvironment is a 3D domain that includes two diffusive molecules, one corresponding to oxygen (required for cell growth) and another corresponding to tumor necrosis factor (TNF), a molecule that can induce death to tumor cells. This allows us to simulate specific experimental conditions where the TNF is supplied in different concentrations, so to try to minimize tumor growth. Moreover, each tumor cell is modelled as an individual agent that, in principle, has several internal sub-models that are required to simulate the respective complex biology related mechanisms (shown in the right of Fig. 1).
First, the TNF Receptor (TNFR) dynamics, i.e. the way molecules attach to cells, are modelled using a set of differential equations that account for the experimental characterization of the TNFR binding dynamics described in in-vivo cells [9]. This model considers *(a)* the binding of the TNF to the cell receptor (k_1), *(b)* the internalization of the TNF-receptor complex (k_2), *(c)* the recycling of the receptor (k_3), and *(d)* the cell growth rate (k_4, not shown in the figure). Cell receptors are proteins with the function of sensing different stimulus and transduce signals by activating downstream signalling pathways. In our model, the receptor is the TNFR, and the TNF is the signalling molecule that binds to TNFR and can trigger a wide range of effects. The binding process has an associated rate or kinetic constant (k_1). After binding and triggering its effect, the complex formed by the TNF bound to the receptor gets internalized into the cell (endocytosis) at a given rate (k_2), and once inside the cell, the signalling molecule (TNF) is degraded and the receptor recycled for its further reuse (k_3).
When TNFR complex concentrations reach a defined threshold, the activation of specific signalling pathways is triggered [9]. The signal propagation through the signalling pathways is modelled using the Cell Fate Boolean model [6]. This model accounts for the most relevant signalling pathways in cancer cell and for three different cell fates or phenotypes, named Proliferation ('Alive' cells), Apoptosis (programmed cell death, or 'Apoptotic' cells), and Necrosis (non-apoptotic cell death, or 'Necrotic' cells). In general, the signal triggered by the binding of the TNF to the receptor induces death in cancer cells. Nonetheless, after prolonged periods of exposure to the stimulus, cells find a way to bypass the death-inducing signal of the TNF and become resistant to the effect

[3] The PhysiBoSSv2.0 is available at: `https://github.com/bsc-life/PhysiBoSSv2`

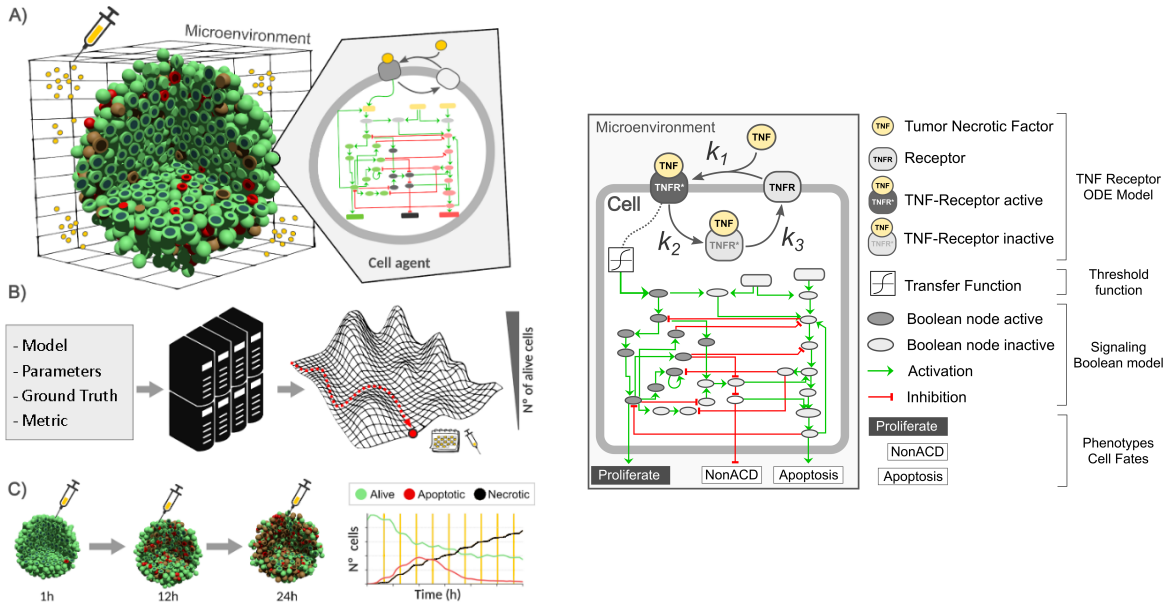

Fig. 1: Overview of our model exploration approach. Left: A) The multi-scale model of the tumor spheroid where each individual cell agent has a TNF receptor connected to the signal transduction network. B) The multi-scale model is combined with model exploration in a HPC cluster to calibrate parameters and to explore treatment strategies. C) An example of the outcome of a treatment strategy that reduces tumor size and avoids the emergence or resistance. Right: Graphical representation of the hybrid model which includes the TNF receptor model, the signal boolean model, a transfer function to connect both, the models and the phenotypes or readout from the boolean model.

of the molecule. As a consequence, treatment strategies based on short pulses of TNF have been proposed to avoid the emergence of resistant cells. For further details, we refer the reader to the work of Calzone et al. (2010) [6].

Within each cell, the TNF receptor model is simulated numerically using the backward Euler method and the numerical integration is conducted in the same time scale used to solve the diffusion. The Boolean model cell is simulated stochastically using the MaBoSS method [31]. Cell growth and death are modelled using the standard PhysiCell models and default parameters, with the exception of cell growth rate k_4 which is calibrated, as explained later. The different treatment strategies are defined as TNF pulses of a given concentration, duration and frequency, i.e. three numerical values that are subject to exploration. Having the simulator in hand, we select sets of parameter values, either regarding the k_1, k_2, k_3, k_4 rates of the agent model when calibrating, or the drug treatment characteristics (k_5, k_6, and k_7) when exploring. Subsequently, we use an optimization via simulation approach to first try to obtain values for the four k_i rates, and then to identify treatment strategies that minimize the number of alive cells and avoid the emergence of resistance, i.e. cells that have activated their survival mechanisms upon TNF reception and are deemed unresponsive to the TNF treatment. Then, simulations are executed in parallel on an HPC infrastructure using the EMEWS framework [25]. After the results are obtained, a post-analysis is performed where only the relevant parts are examined, which in our case are the time-series of alive, apoptotic, and necrotic tumor cells.

4 Parameter Exploration: Calibration and Discovery of Treatments

Our implementation combines EMEWS for parallelizing the candidate solution evaluations, and the DEAP Python library[4] for the Genetic Algorithm. Our approach follows the main principles of model exploration and, in particular, we search for parameter values that manage to minimize desired metrics defined over the simulation results. Our simulator integrates PhysiBoSSv2. Similar to most simulation frameworks, PhysiBoSSv2 includes multiple configuration parameters that can be fine-tuned, for example in order to achieve the desired levels of fitting a set of experimental observations, or to estimate the outcome of a drug treatment configuration.

[4] `https://github.com/DEAP/deap`

In an attempt to characterize the parameter space, we could perform a sweep search. However, the search space requires significant time and computational resources for exploration. A more sophisticated solution would be to incorporate a heuristic search approach, e.g. a Genetic Algorithm [27], that is able to search in a more informed manner, based on a fitness function that evaluates candidate solutions. This way, we expect to achieve better results using significantly less resources when compared to the exhaustive sweep approach.

4.1 Configuring the multi-scale model of tumor growth

For the proper calibration of the simulator, it is crucial to determine and fix the values for *(i)* the TNFR binding rate, *(ii)* the TNFR endocytosis rate, *(iii)* the TNFR recycling rate, and *(iv)* the cell growth rate. To ease notation, we refer to these parameters as k_i, $i \in \{1, 2, 3, 4\}$, respectively. After calibrating the simulator, we may explore the effectiveness of various drug treatment policies. In this second step, the k_i values correspond to the respective drug treatment configuration parameters, i.e. *(i)* the TNF Administration Frequency, which actually shows how often the drug is injected measured in minute intervals, *(ii)* the TNF Duration dictating for how long the drug will be administered in each iteration, and *(iii)* the TNF Concentration. These are noted as k_i, $i \in \{5, 6, 7\}$, respectively. The allowed value ranges for these variables are shown in Table 1. These ranges

Table 1: Ranges of the PhysiBoSS parameters.

Parameter	Min	Max
k_1: TNFR Binding rate	0.01	1
k_2: TNFR Endocytosis rate	0.01	1
k_3: TNFR Recycling rate	0.01	1
k_4: Cell growth rate	0.0015	0.0075
k_5: TNF Administration Frequency	10	1000
k_6: TNF Duration	5	120
k_7: TNF Concentration	0	1

are estimated empirically by domain experts with long experience in this field of research.

4.2 Fitness function

The suitability of k_i, $i \in \{1, 2, 3, 4\}$ value selection can be evaluated if we compare the simulations produced by using particular values for the k_i, against some "gold standard" simulations that are considered as ground-truth. These gold standards constitute the time series of the evaluation of 'Alive', 'Apoptotic', and 'Necrotic' cells over time, when applying the following TNF treatment configurations to the cells in the simulation:

- TNF injections every 150 minutes (TNF=150)
- TNF injections every 600 minutes (TNF=600)

and, in both cases, the duration is 10 minutes, and the concentration of TNF is 0.02 TNF/μm^3. A graphical representation of the gold standards is shown in Figure 8 as solid lines. In general, gold standard cases may originate either from real world studies (possibly small-scale), or by other simulators that have been validated and shown realistic results in the past. Here, we consider results from a stable previous version of the PhysiBoSSv2 simulator.

Now, our main goal is to first determine the k_i, $i \in \{1, 2, 3, 4\}$ values that produce simulations as close as possible to the ground truth set by the gold standards. Then, given the calibrated model, the other set of k_i, $i \in \{5, 6, 7\}$ values is explored, to find good drug treatment policies that can be proved beneficial in real-world trials. To address these two cases, the evaluation functions that are chosen as minimization objectives in the employed Genetic Algorithm, are: *(i)* the sum of distances between the time points of the gold standards and the time points of the simulations produced by a set of k_i values, and *(ii)* the number of 'Alive' cells at the end of each simulation. The second case is used to indicate the effectiveness of different drug treatment policies. For the first case, that of calibration, we employ the following distance metrics:

1. The *Euclidean* distance.
2. The distance computed according to the *Dynamic Time Warping* algorithm [4].
3. The L_1*-norm*, i.e. the absolute differences between time points.

Dynamic Time Warping (DTW) is a special type of distance function that aims to minimize distance that is caused by phase differences between two time-series.

More formally, for a simulation with a length of T time points, let L_t^j, N_t^j, and P_t^j, $t \in 1, ..., T$ be the normalized number of 'Alive', 'Necrotic', and 'Apoptotic' cells, for each of $j \in \{\text{TNF=150,TNF=600}\}$ gold standard cases, and l_t^j, n_t^j, and p_t^j, $t \in 1, ..., T$ be the normalized numbers of corresponding cell categories from the simulations

produced by certain k_i values. The normalization of cell numbers is made by dividing each value of the time series ('Alive', 'Apoptotic', and 'Necrotic' cell counts) with the maximum number of 'Alive' cells from the respective j case. This has shown to be necessary in order to remove bias, since, in the typical case, the absolute cell counts of TNF=600 are larger, resulting to increased absolute distance.
Then, the value of the fitness function is given by:

$$F = \sum_j ||\mathbf{L}^j - \mathbf{l}^j||_D + ||\mathbf{N}^j - \mathbf{n}^j||_D + ||\mathbf{P}^j - \mathbf{p}^j||_D \quad (1)$$

where $||.||_D$ denotes the distance type for each of the three cases above. Equation 1 is the fitness function that the GA seeks to minimize. Next, we describe the implementation of the GA that we have used for optimizing the k_i values.

4.3 Genetic algorithm

Heuristic search approaches, such as the Genetic Algorithm (GA), can be used in this setting with the expectation that they will converge to optimal areas of the k_i parameter space and reveal high quality combinations of the desired parameters relatively easy. A flowchart illustrating the execution stages of the proposed GA is shown in Fig 2.

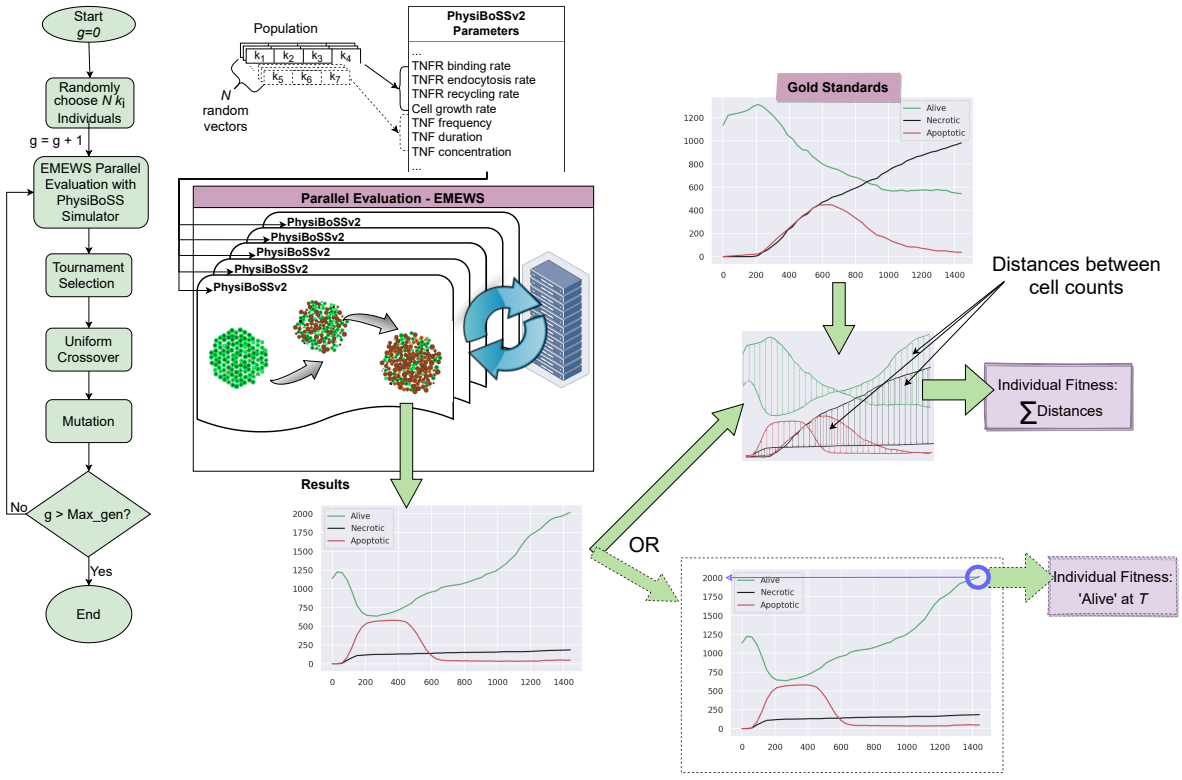

Fig. 2: Genetic algorithm flowchart. g denotes generation index; T denotes last simulation interval.

First, a random initial population of a given size is generated, i.e. random selections of k_i values, which from now on will be referred to as individuals. Individuals are represented as tuples of real values. According to which scenario is investigated (either calibration, or drug treatment exploration that is marked with dashed border lines), the values of the other tuples are fixed throughout the experiments: For the calibration case we fix TNF time, duration, and concentration as dictated by the gold standard cases, and in the exploration case we use the best binding, endocytosis, recycling, and growth rates discovered from the calibration experiments. Next, PhysisBoSSv2 performs simulations using parameter values that correspond to each individual, and the results are evaluated according to the fitness function, e.g. the one defined by eq. (1). Due to some inherent

simulator stochasticity—a feature that leads to slightly different cell count values in each run using the same k_i values—there is an explicit configuration to execute replicate evaluations of the same individual. If the number of replicate simulations is larger than 1, then the fitness score of each individual is the average of the scores produced by the replicate runs.
Next, by taking into account the fitness of each individual, the selection operator is applied. There are multiple selection operator types that can be employed, and further implementation details are given in the next section. Subsequently, the crossover operator is applied on the selected individuals, in order to come up with new ones that in principle maintain desirable chromosomes (k_i values) from the original ones, while exploring alternatives that may lead to better fitness values. Finally, mutation is randomly performed on each of the chromosomes of the individuals, where a chromosome value is changed,based on a given probability threshold.
This procedure, i.e. evaluation, selection, crossover, and mutation is performed repeatedly for a designated number of iterations *Max_gen*, or following other termination criteria, e.g. by checking population convergence, or setting thresholds on the fitness scores. In our implementation, the GA can be configured to work with four different termination criteria, apart from reaching a maximum generation number, it may also stop when a minimum fitness score value is reached, or the population fitness variance or average scores stay below a threshold for five consecutive generations. Finally, the output files include the k_i values that produced the minimum fitness score, the score value itself, and the results of the corresponding PhysiBoSSv2 simulations run during the evaluation for that particular individual. This comes in the form of time-series depicting the cell count of each cell category of interest. The main computational bottleneck in the presented GA workflow is the evaluation of each individual with PhysiBoSSv2. For this reason, in the EMEWS workflow, each simulation is divided in a number of threads, so that each generation is effectively delegated to different processors. This enables the parallel and distributed execution, thus reducing the computational burden in each single processor.

5 Experimental Results

Our GA implementation that we are incorporating for the experimental evaluation can be found in an online repository, along with instructions for reproducing the experiments, and links to the current results dataset.[5]
For the evaluation purposes, the GA was configured to run for 30 generations, with a population number of 40 individuals. As a selection operator, we employed the 'Tournament Selection' with tournament size 3, after empirically observing that it can lead to high quality solutions. Among the various crossover operator types, we choose the Uniform crossover that applies equal probability to the parent individuals' chromosomes to be inherited by the offspring. The crossover probability was set to 0.75, and the mutation probability to 0.5. The chosen hyperparameter configuration was obtained after performing GA runs on a reduced search space, and exploring the performance of different values. Figure 3 shows the course of the best discovered individual for different crossover and mutation probabilities in these small scale experiments.
To better illustrate the performance of the proposed GA approach, we compare it with an exhaustive Sweep search, an alternative method for searching that serves as a benchmark. The Sweep search iteratively evaluates a grid of individuals that are uniformly distributed in the search space. The grid is predetermined, and one simulation for each point is executed to calculate the fitness score that characterizes it. No particular prioritization is given regarding to which points or subsets of them will be searched first, and we discretise the space so that a finite set of points is obtained. A drawback of this method is that when we try to balance the trade-off between execution time and amount of points evaluated, we may end up with a sparse grid, and it is possible that well-performing points will be skipped and never examined. Instead, as we will be illustrating in this section, the GA manages to find solutions of high quality using significantly less resources than the Sweep search.
All experiments were conducted using the Mare Nostrum 4 (MN4) HPC infrastructure provided by the Barcelona Supercomputing Centre.[6] We used up to 8 nodes with 384 processors in total.

5.1 Simulator Calibration

To begin, we examine how our approaches compare with respect to the simulator calibration capabilities. For Sweep search we create a grid of uniformly distributed points in the four dimensional space of the k_i parameters, and evaluate each one of them without a particular preference in the ordering of the simulations. The scores reported for each set of parameter configurations correspond to the color of the square in the heatmap shown in Figure 4. We can observe that there are regions of the examined space that appear to include parameter values which lead to more similar results to the gold standards (light blue squares). Also, there are critical regions beyond which parameter values produce much different simulations with increased distance values from the gold standards (orange, red and darker blue squares). The k 4-tuple with the lowest L_1 distance is marked

[5] `https://github.com/xarakas/spheroid-tnf-v2-emews`
[6] `https://www.bsc.es/marenostrum/marenostrum`

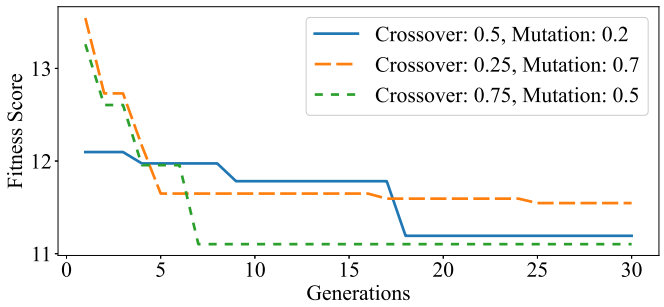

Fig. 3: Performance of the best individual for different crossover and mutation probabilities in a subset of the search space.

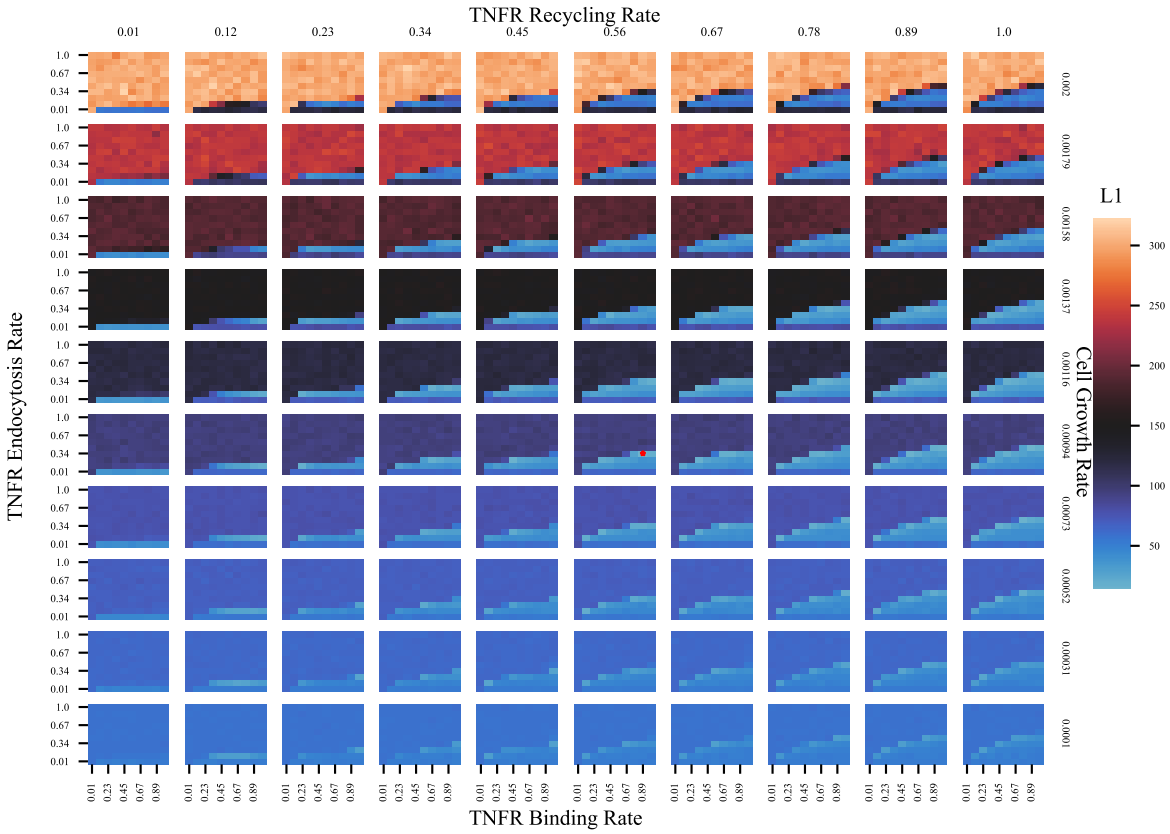

Fig. 4: Heatmap of all k 4-tuple L_1 distances, as evaluated by the Sweep search. The red star denotes the configuration with the minimum score. Each x-axis denotes the TNFR Binding Rate, each y-axis, the TNFR Endocytosis Rate; columns correspond to TNFR Recycling Rate values, and rows to Cell Growth Rate.

with a red star. Moreover, it appears that in the examined space there exist ridges between 'worse' and 'better' regions, where minimum local solutions lie within. The L_1 distance is selected as the illustration example because it is shown to converge to solutions of good quality, and has more variance than the Euclidean, making the comparison more clear and accessible.

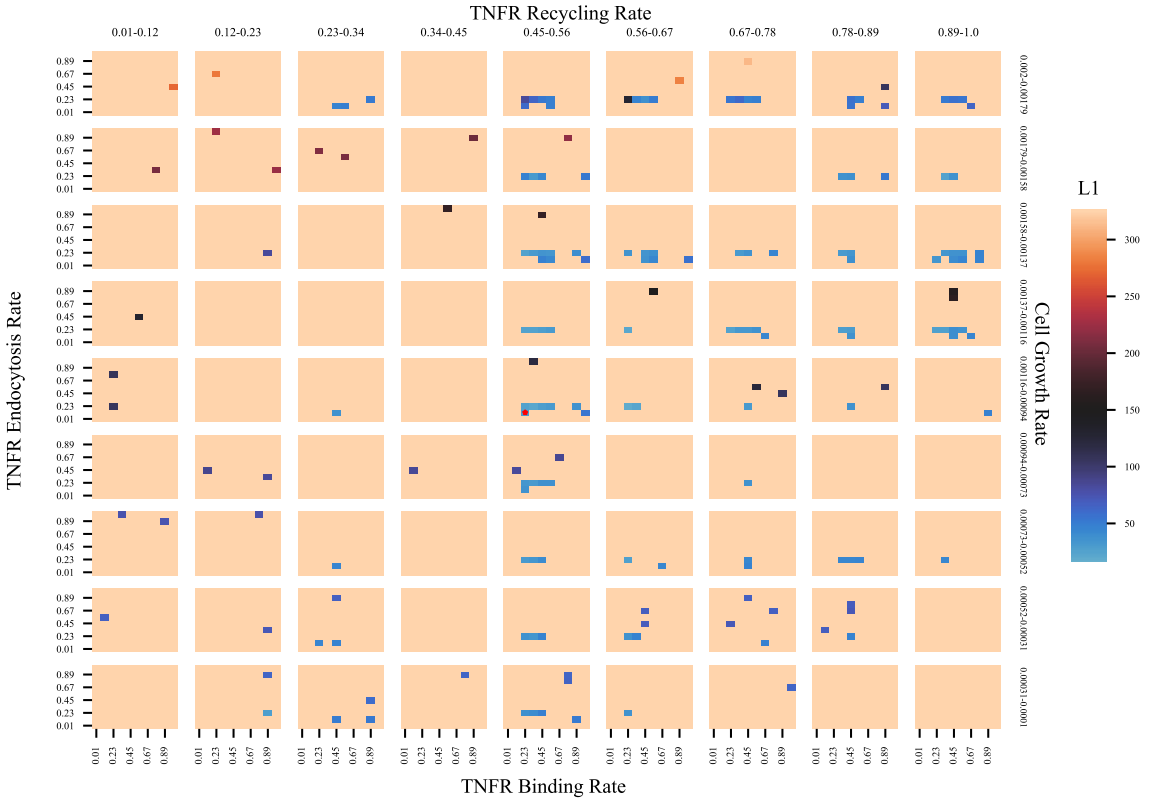

Fig. 5: Heatmap of all k 4-tuple L_1 distances, as evaluated by the GA search. The red star denotes the configuration with the minimum score. Each x-axis denotes the TNFR Binding Rate ranges, each y-axis, the TNFR Endocytosis Rate ranges; columns correspond to TNFR Recycling Rate ranges, and rows to Cell Growth Rate ranges.

Up next, we run the GA evaluation for each distance type. In Fig. 5 we present the respective heatmap depicting the space visited by the GA using the L_1 distance. We should note that, due to the fact that the grid examined by Sweep search includes only 10 predetermined values for each dimension, it is probable that the actual k_i values that the GA produces might not 'fall' directly into one of them. Thus, to be able to visualize the results for the sake of comparison, we group the GA points in the ranges set by the grid values. The best solution is marked again with a red star, and the light orange squares indicate 'N/A' values, i.e. value ranges that the GA did not assess. Also, we can see that the GA generates a far more sparse grid of points subject to evaluation, and that, in their majority, those points are placed inside the 'interesting' regions of Fig. 4.
To gain more intuition, in Fig. 6 we fix the k_4 parameter to the region indicated in the sixth and fifth row of Fig. 4 and Fig. 5, respectively. We can see that the GA manages to skip the evaluation of points of low quality, by utilizing the underling information regarding suitability (encoded by the L_1 distance in the fitness function) and the genetic operators (selection, crossover, and mutation). Furthermore, although the distinction between 'interesting' and 'not interesting' regions cannot be made using a smooth surface, the GA visits subspaces with local optima, and focuses its search during the later generations at specific ones that give promising results.
The actual values of the k 4-tuples given by each method are shown in Table 2, along with the corresponding distances. Note that, each distance type generates values of different magnitude, thus it is not plausible to compare different distance types in an absolute manner. First of all, the indicated k_4 values are close for both methods and all distance types, implying that the realistic value of cell growth rate is more or less bounded. In the case of k_3, the recycling rate, we have again similar results, except for the case of DTW distance for both search methods. This can be justified by the fact that the recycling rate is related to periodic effects,

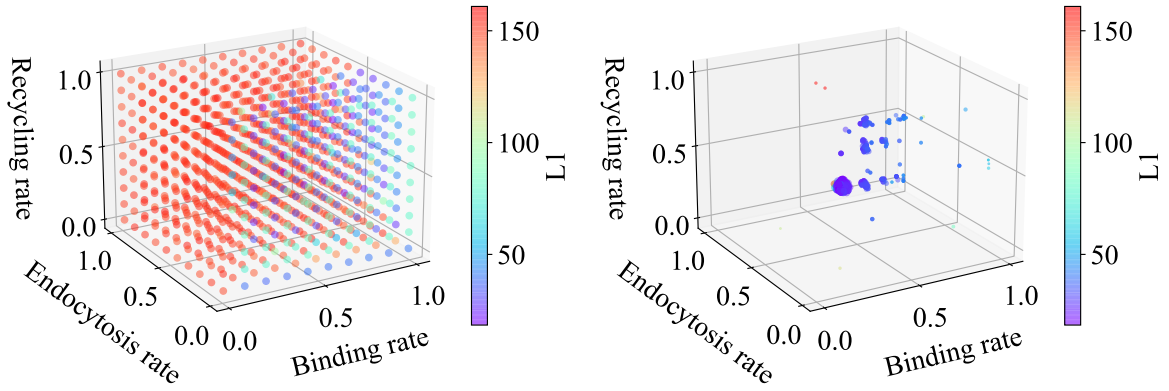

Fig. 6: Points evaluated by Sweep search (left) and GA (right), with the k_4 cell growth parameter lying in the range [0.0009, 0.0015]. Color denotes the L_1 distance from the gold standards, and point sizes in the GA case indicate temporal relationship, i.e. bigger point radius means that it is visited in later generations.

for which the DTW distance is not strict; it is designed to bring close instances, even though them having different phases. Finally, for k_1 and k_2, Sweep search and GA point to different regions. This happens because the particular best point found by Sweep search is placed within a sequence of local minima that are not very far from one another. This can be considered as a 'trap', which GA approaches are known to easily fall into. Overall, both Sweep search and GA lead to acceptable results that are not too far from each other in terms of distances from the gold standard time-series.

Table 2: Calibration results from Sweep search and the Genetic Algorithm

Method and Distance		k_1	k_2	k_3	k_4	Score
Sweep	Euclidean	0.89	0.34	0.56	0.000944	2.53
	DTW	0.89	0.45	1	0.000944	7.8
	L_1	0.89	0.34	0.56	0.000944	14.4
GA	Euclidean	0.21	0.16	0.53	0.00084	2.89
	DTW	0.33	0.17	0.34	0.001	9.85
	L_1	0.22	0.16	0.54	0.00097	16.28

Table 3 shows the number of k 4-tuples that each method examines throughout its execution, as well as the total number of simulations that were conducted. Sweep search is shown in one row, because in a single run we measure all three distances simultaneously. First of all, it is clear that significantly less points are examined by the GA for all distances, than in the Sweep search case. Since we compare with two different gold standard cases, in all methods, the number of simulations is twice the number of k 4-tuples examined. In general, as GA examines less individuals, consequently, far less computational resources are consumed. Recall that the GA includes 40 individuals for 30 generations, equaling to 1.200 individuals. However, a proportion of this number concerns duplicate individuals, which managed to survive across generations, and are not evaluated again. This computational gain for the GA however, comes with a small impact on accuracy, since although managing to find solutions of high quality, there is still some room for improvement if we compare the scores of Table 2.
Figure 7 depicts the normalized scores of the best individuals of each generation in the initial experiments. We can observe that, despite the different magnitude in the non-normalized fitness scores for each distance type, when examining normalized scores, the algorithm is shown to converge to individuals of high quality in all cases. When using the L_1 distance though, we can observe a more smooth downward movement, which illustrates that this distance type can help the GA have a stable evolution.
As the last part of the calibration process, we perform visual inspection on the actual time series that are produced by the simulator using the designated values, and the two gold standards. The results are presented in Figure 8. The gold standards are shown with solid curves. For the TNF=150, we can see fluctuating numbers of the alive cells every 150 minutes, which is the administration frequency in this case, but nevertheless having a decreasing trend. Respectively, necrotic cell numbers are increasing as time passes. The apoptotic cell type increase in the first half of the simulation to later drop in a similar slope with the alive cell count. For the

Table 3: Comparison between Sweep search and Genetic Algorithm for the calibration experiments that used 384 CPUs each.

Method		k 4-tuples	Simulations	Minutes
Sweep	All distances	10000	20000	5268
GA	Euclidean	940	1880	877
	DTW	940	1880	881
	L_1	871	1742	876

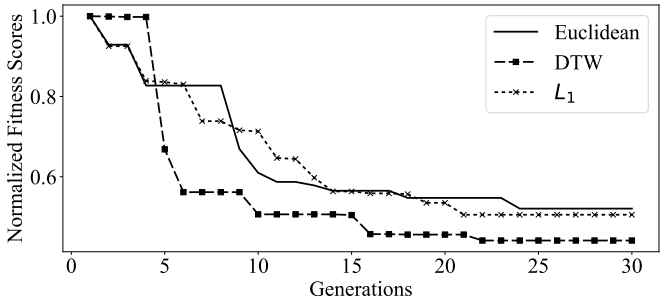

Fig. 7: Genetic algorithm convergence.

second gold standard, TNF=600, seems to describe a different scenario where the drug administration does not have the desired effect, since the number of alive tumor cells grows larger than their count at the beginning of the simulation.

In the case of Sweep search (top figures), the L_1 distance points to the same solution as the Euclidean distance, thus the corresponding time-series are the same. For the TNF=150 gold standard, the Euclidean and L_1 distances select a configuration that creates simulations slightly closer to the desired one, especially in the first half of the simulation. The same holds for the TNF=600 gold standard as well, where the configuration that matches best the desired curves is the one with the minimum Euclidean and L_1 distance. Note though, that the DTW distance too selects a configuration that is not very far from the gold standards. We can see that in the case of TNF=150 the individual with the minimum Euclidean distance (dot-marked curve) matches quite well the alive cell count values, while the individual with the minimum DTW (square-marked curve) distance does not perform so well for the necrotic type. In TNF=600, the Euclidean distance leads to alive cell counts that are less close to the original, and L_1 (x-marked curve) has a slightly better fit. Overall, judging by the results of the visual inspection, we conclude that the simulator is calibrated successfully, and we select the k 4-tuple championed by the Sweep search when using the Euclidean and L_1 distance. These values are used for configuring the simulator for the next set of experiments, which examines drug policies that could be proven beneficial for patients.

5.2 Drug policy exploration

Being equipped with the best PhysiBoSSv2 configuration for the four rate values, we continue the experiments and search on different parameters, now focusing on drug treatment characteristics. In particular, the variables we wish to tune are the frequency of TNF injections, the duration, and the concentration of TNF dosages given to patients, i.e. each individual is represented as $\{$*TNF Administration Frequency*$/k_5$, *TNF Duration*$/k_6$, *TNF concentration*$/k_7\}$. Since in this stage there is no gold standard case to match with, instead of having distance measurements as the fitness score, the fitness function counts the number of 'Alive' cells at the final time step of each simulation—which in our case is after 1440 minutes of treatment.

Similar to the previous set of experiments, we first perform a Sweep search of the space to try to identify interesting regions. Also, to further simplify presentation, we distinguish solution candidates to viable and unviable ones. The unviable ones, which we omit from the following figures, are those that end up with more alive tumor cells at the end of the simulation than in the beginning, meaning that the particular drug treatment is not successful. On the contrary, the viable ones are those that manage to reduce the tumor cell count by the end of the simulation.

The results from Sweep search are shown in the left of Figure 9, marking with different color each individual according to its fitness score. As we can see, there is an interesting region that contains parameter values, which lead to promising drug treatment configurations (dark blue and purple points). The GA on the other hand, as shown in the right of Figure 9, cannot be used to effectively characterize the whole search space, since the individuals examined are not uniformly distributed. However, it is apparent that the GA converges

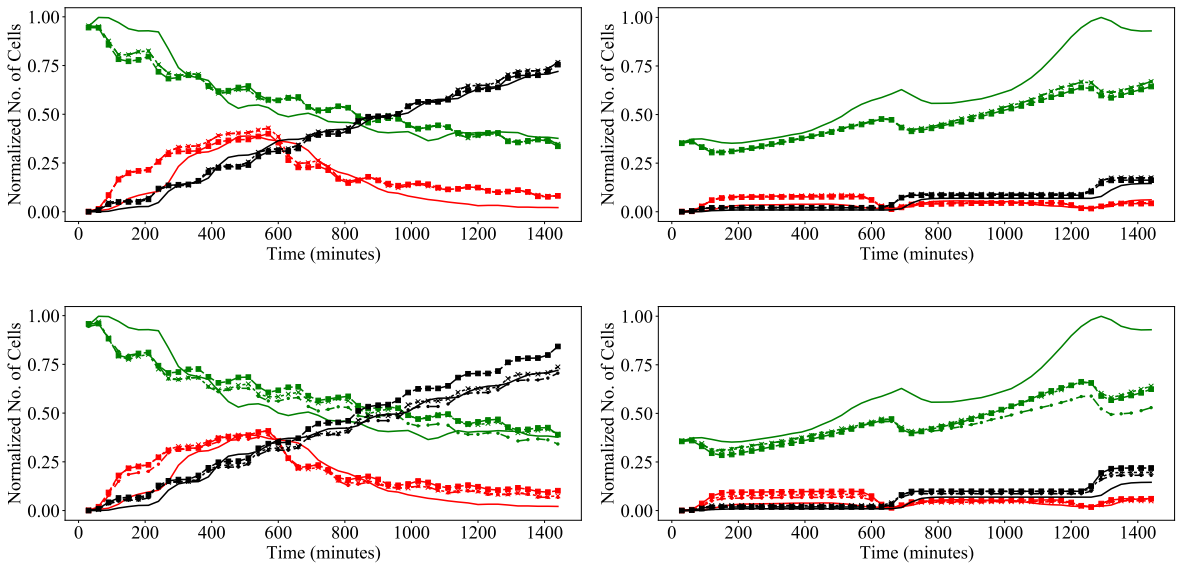

Fig. 8: Time-series produced by the fittest $\{$*TNFR Binding rate*/k_1, *TNFR Endocytosis rate*/k_2, *TNFR Recycling rate*/k_3 *Cell growth rates*/$k_4\}$ values combinations. Top: Sweep search; Bottom: Genetic Algorithm. Left: 150' TNF pulse; right: 600' TNF pulse. Green curves show the number of alive cells, red the number of apoptotic cells, and black the number of necrotic cells. Solid lines are the gold standards. Dot-marked curves correspond to the solution provided by the Euclidean distance, squared-marked curves correspond to those computed with the use of the DTW distance, and 'x'-marked curves computed with the use of the L_1 distance.

to points that lie within the interesting region marked by the Sweep search. Again, for the GA case, the size of each point increases proportionally to the generation that it is examined, meaning that larger points are visited later during the search. Thus, we can conclude that the GA is able to distinguish among better and worse quality solutions, and expand the search around potentially interesting regions of the search space. The actual parameter values produced by each of the two approaches is shown in Table 4, with the GA pointing to a solution that restrains more tumor cells, with a final score that counts 191 cells, whereas the Sweep search, 197.

Table 4: Drug policy exploration results from the Sweep search and the Genetic Algorithm

Method	k_5	k_6	k_7	Score
Sweep	260.52	20.78	0.067	197
GA	192	19	0.0457	191

Table 5 presents the number of individuals examined and the number of simulations performed in each case. The GA is shown to be the better option since it manages to find solutions leading to less 'Alive' cell counts, by examining one order of ten less candidate solutions than Sweep search examines.

Table 5: Comparison between the Sweep search and the GA for the drug policy exploration experiment. Each, used 384 CPUs.

Method	k triples	Simulations	Minutes
Sweep	8000	8000	2809
GA	632	632	1120

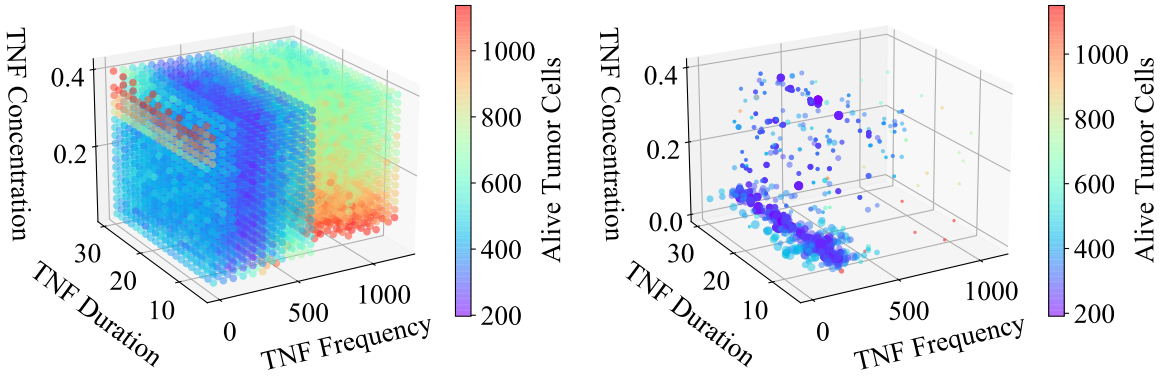

Fig. 9: k-triple evaluations for drug policy exploration, Sweep search on the left, and GA on the right. Color denotes the number of the alive tumor cells at the end of the simulation.

6 Conclusions & Further Work

Multi-scale simulations have been proven quite valuable in various application areas, and the domain of drug discovery is no exception. We incorporated PhysiBoSSv2 bundled into EMEWS workflows to allow the parallel execution of large-scale cancer cells growth experiments. Our goal was first to configure the new version of the simulator properly so to produce realistic results, and then to use it for discovering effective drug treatment policies that are able to attack cancer cells and contain their growth. In both cases the search was performed using a Genetic algorithm that can search the space in a more informed manner, by utilizing a fitness function to characterize the suitability of candidate solutions. We employed different types of fitness functions, and analyzed their capability for discovering best solutions. Our results from experimental evaluations performed in a high performance computing infrastructure validate the effectiveness of the approach, highlighting the difference between informed and uninformed search approaches in this setting.
In the future, we plan to employ additional AI methods to compare their performance with the GA, and also expand our search to other simulator parameters as well. Moreover, we are in the process of devising an approach for the early termination of simulation instances, which are not relevant to the objectives of biologists and cannot be used to extract any useful information. This could be achieved, for example, with the incorporation of early time-series classification techniques [1,16]. Furthermore, our approach is going to be incorporated in the HORIZON-2020 INFORE project, in order to serve as a sub-module in a larger system designed for extreme-scale analytics.

Acknowledgements

This work has received funding from the EU Horizon 2020 RIA program INFORE under grant agreement No 825070.

References

1. Alevizos, E., Artikis, A., Paliouras, G.: Wayeb: a tool for complex event forecasting. pp. 26–35. LPAR-22: 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning (2018)
2. An, G.: Closing the Scientific Loop: Bridging Correlation and Causality in the Petaflop Age. Science Translational Medicine **2**(41), 41ps34 (Jul 2010). https://doi.org/10.1126/scitranslmed.3000390, `http://stm.sciencemag.org/content/2/41/41ps34`
3. Babtie, A.C., Stumpf, M.P.H.: How to deal with parameters for whole-cell modelling. Journal of The Royal Society Interface **14**(133), 20170237 (Aug 2017). https://doi.org/10.1098/rsif.2017.0237, `https://royalsocietypublishing.org/doi/full/10.1098/rsif.2017.0237`, publisher: Royal Society
4. Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. vol. 10, pp. 359–370. Seattle, WA, USA:, KDD workshop (1994)
5. Calzone, L., Barillot, E., Zinovyev, A.: Logical versus kinetic modeling of biological networks: applications in cancer research. Current Opinion in Chemical Engineering **21**, 22–31 (Sep 2018). https://doi.org/10.1016/j.coche.2018.02.005, `http://www.sciencedirect.com/science/article/pii/S2211339817300941`

6. Calzone, L., Tournier, L., Fourquet, S., Thieffry, D., Zhivotovsky, B., Barillot, E., Zinovyev, A.: Mathematical modelling of cell-fate decision in response to death receptor engagement. PLoS Computational Biology **6**(3), e1000702 (Mar 2010). https://doi.org/10.1371/journal.pcbi.1000702, http://dx.plos.org/10.1371/journal.pcbi.1000702, publisher: Public Library of Science ISBN: 1553-7358 (Electronic)\n1553-734X (Linking)
7. Deisboeck, T.S., Wang, Z., Macklin, P., Cristini, V.: Multiscale cancer modeling. Annual review of biomedical engineering **13**, 127–55 (Aug 2011). https://doi.org/10.1146/annurev-bioeng-071910-124729, http://www.ncbi.nlm.nih.gov/pubmed/21529163, publisher: NIH Public Access
8. Du, W., Elemento, O.: Cancer systems biology: embracing complexity to develop better anticancer therapeutic strategies. Oncogene **34**(25), 3215–3225 (Jun 2015). https://doi.org/10.1038/onc.2014.291, https://www.nature.com/articles/onc2014291, number: 25 Publisher: Nature Publishing Group
9. Fischer, R., Maier, O., Naumer, M., Krippner-Heidenreich, A., Scheurich, P., Pfizenmaier, K.: Ligand-induced internalization of tnf receptor 2 mediated by a di-leucin motif is dispensable for activation of the nfkb pathway. Cellular Signalling **23**(1), 161–170 (Jan 2011). https://doi.org/10.1016/j.cellsig.2010.08.016
10. Ghaffarizadeh, A., Friedman, S.H., Macklin, P.: BioFVM: an efficient, parallelized diffusive transport solver for 3-D biological simulations. Bioinformatics **32**(8), 1256–1258 (Apr 2016). https://doi.org/10.1093/bioinformatics/btv730, https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btv730, publisher: Oxford University Press
11. Ghaffarizadeh, A., Heiland, R., Friedman, S.H., Mumenthaler, S.M., Macklin, P.: PhysiCell: An open source physics-based cell simulator for 3-D multicellular systems. PLOS Computational Biology **14**(2), e1005991 (Feb 2018). https://doi.org/10.1371/journal.pcbi.1005991, http://www.ncbi.nlm.nih.gov/pubmed/29474446
12. Hanahan, D., Weinberg, R.A.: Hallmarks of cancer: The next generation. Cell **144**(5), 646–674 (Mar 2011). https://doi.org/10.1016/j.cell.2011.02.013, http://www.ncbi.nlm.nih.gov/pubmed/21376230
13. Hornberg, J.J., Bruggeman, F.J., Westerhoff, H.V., Lankelma, J.: Cancer: A Systems Biology disease. BioSystems **83**(2-3 SPEC. ISS.), 81–90 (2006). https://doi.org/10.1016/j.biosystems.2005.05.014
14. Jagiella, N., Rickert, D., Theis, F.J., Hasenauer, J.: Parallelization and High-Performance Computing Enables Automated Statistical Inference of Multi-scale Models. Cell Systems **4**(2), 194–206.e9 (Feb 2017). https://doi.org/10.1016/j.cels.2016.12.002, https://www.cell.com/cell-systems/abstract/S2405-4712(16)30412-4, publisher: Elsevier
15. King, M.D., Long, T., Andersen, T., McDougal, O.M.: Genetic algorithm managed peptide mutant screening: Optimizing peptide ligands for targeted receptor binding. Journal of chemical information and modeling **56**(12), 2378–2387 (2016)
16. Kladis, E., Akasiadis, C., Michelioudakis, E., Alevizos, E., Artikis, A.: An empirical evaluation of early time-series classification algorithms. p. to appear. SIMPLIFY-2021, EDBT Workshop (2021)
17. Kohl, P., Noble, D.: Systems biology and the virtual physiological human. Molecular systems biology (2009)
18. Letort, G., Montagud, A., Stoll, G., Heiland, R., Barillot, E., Macklin, P., Zinovyev, A., Calzone, L.: PhysiBoSS: a multi-scale agent-based modelling framework integrating physical dimension and cell signalling. Bioinformatics (2019). https://doi.org/10.1093/bioinformatics/bty766, https://academic.oup.com/bioinformatics/advance-article/doi/10.1093/bioinformatics/bty766/5087713
19. Li, J., Yin, Q., Wu, H.: Structural Basis of Signal Transduction in the TNF Receptor Superfamily. Advances in immunology **119**, 135–153 (2013). https://doi.org/10.1016/B978-0-12-407707-2.00005-9, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3781945/
20. Lipinski, K.A., Barber, L.J., Davies, M.N., Ashenden, M., Sottoriva, A., Gerlinger, M.: Cancer Evolution and the Limits of Predictability in Precision Cancer Medicine. Trends in Cancer **2**(1), 49–63 (Jan 2016). https://doi.org/10.1016/j.trecan.2015.11.003, http://www.ncbi.nlm.nih.gov/pubmed/26949746
21. Lo, Y.C., Rensi, S.E., Torng, W., Altman, R.B.: Machine learning in chemoinformatics and drug discovery. Drug Discovery Today **23**(8), 1538 – 1546 (2018). https://doi.org/https://doi.org/10.1016/j.drudis.2018.05.010, http://www.sciencedirect.com/science/article/pii/S1359644617304695
22. Metzcar, J., Wang, Y., Heiland, R., Macklin, P.: A Review of Cell-Based Computational Modeling in Cancer Biology. JCO Clinical Cancer Informatics (3), 1–13 (Feb 2019). https://doi.org/10.1200/CCI.18.00069, https://ascopubs.org/doi/full/10.1200/CCI.18.00069, publisher: American Society of Clinical Oncology
23. Ozik, J., Collier, N., Heiland, R., An, G., Macklin, P.: Learning-accelerated discovery of immune-tumour interactions. Molecular Systems Design and Engineering **4**, 747–760 (08 2019). https://doi.org/10.1039/c9me00036d
24. Ozik, J., Collier, N., Wozniak, J.M., Macal, C., Cockrell, C., Friedman, S.H., Ghaffarizadeh, A., Heiland, R., An, G., Macklin, P.: High-throughput cancer hypothesis testing with an integrated PhysiCell-EMEWS workflow. BMC Bioinformatics **19**(18), 483 (Dec 2018). https://doi.org/10.1186/s12859-018-2510-x, https://doi.org/10.1186/s12859-018-2510-x

25. Ozik, J., Collier, N.T., Wozniak, J.M., Spagnuolo, C.: From desktop to Large-Scale Model Exploration with Swift/T. pp. 206–220. 2016 Winter Simulation Conference (WSC) (Dec 2016). https://doi.org/10.1109/WSC.2016.7822090, iSSN: 1558-4305
26. Robertson-Tessi, M., Gillies, R.J., Gatenby, R.A., Anderson, A.R.A.: Impact of Metabolic Heterogeneity on Tumor Growth, Invasion, and Treatment Outcomes. Cancer Research **75**(8), 1567–1579 (Apr 2015). https://doi.org/10.1158/0008-5472.CAN-14-1428, http://www.ncbi.nlm.nih.gov/pubmed/25878146
27. Russell, S.J., Norvig, P.: Artificial intelligence: a modern approach. Pearson Education Limited (2016)
28. Saeedizadeh, F., Moghaddam, R.K.: Optimal control of hiv stochastic model through genetic algorithm. pp. 401–405. 2017 7th International Conference on Computer and Knowledge Engineering (ICCKE) (2017)
29. Sahlol, A.T., Moemen, Y.S., Ewees, A.A., Hassanien, A.E.: Evaluation of cisplatin efficiency as a chemotherapeutic drug based on neural networks optimized by genetic algorithm. pp. 682–685. IEEE, 2017 12th International Conference on Computer Engineering and Systems (ICCES) (2017)
30. Sliwoski, G., Kothiwale, S., Meiler, J., Lowe, E.W.: Computational methods in drug discovery. Pharmacological reviews **66**(1), 334–395 (2014)
31. Stoll, G., Viara, E., Barillot, E., Calzone, L.: Continuous time boolean modeling for biological signaling: application of gillespie algorithm. BMC systems biology **6**, 116 (2012). https://doi.org/10.1186/1752-0509-6-116
32. Tekin, E., Sabuncuoglu, I.: Simulation optimization: A comprehensive review on theory and applications. IIE Transactions **36**(11), 1067–1081 (Nov 2004). https://doi.org/10.1080/07408170490500654, https://doi.org/10.1080/07408170490500654, publisher: Taylor & Francis eprint: https://doi.org/10.1080/07408170490500654
33. Vogelstein, B., Papadopoulos, N., Velculescu, V.E., Zhou, S., Diaz, L.A., Kinzler, K.W.: Cancer Genome Landscapes. Science **339**(6127), 1546–1558 (Mar 2013). https://doi.org/10.1126/science.1235122, http://www.ncbi.nlm.nih.gov/pubmed/23539594
34. Yizhak, K., Chaneton, B., Gottlieb, E., Ruppin, E.: Modeling cancer metabolism on a genome scale. Molecular systems biology **11**(6), 817 (Jan 2015). https://doi.org/10.15252/msb.20145307, http://msb.embopress.org/cgi/doi/10.15252/msb.20145307