Start&Stop - a PhysiCell and PhysiBoSS 2.0 add-on for interactive simulation control

Riccardo Smeriglio[1], Roberta Bardini[1], Alessandro Savino[1], Stefano Di Carlo[1*]

[1]Department of Control and Computer Engineering, Politecnico di Torino, Corso Duca degli Abruzzi, 24, Turin, 10129, TO, Italy.

*Corresponding author(s). E-mail(s): stefano.dicarlo@polito.it;
Contributing authors: riccardo.smeriglio@polito.it;
roberta.bardini@polito.it; alessandro.savino@polito.it;

Abstract

In computational biology, *in silico* simulators are vital for exploring and understanding the behavior of complex biological systems. Hybrid multi-level simulators, such as PhysiCell and PhysiBoSS 2.0, integrate multiple layers of biological complexity, providing deeper insights into emergent patterns. However, one key limitation of these simulators is the inability to adjust simulation parameters once they have started, which impedes real-time exploration and adaptation of dynamic protocols—ranging from biofabrication to *in vitro* pharmacological testing. To address this challenge, we introduce the Start&Stop add-on for PhysiCell, which is automatically adaptable to PhysiBoSS 2.0. This add-on offers multi-level state preservation and multi-modal stop control—triggered by simulation time or cell conditions—enabling users to pause a simulation, adjust parameters, and then resume from the exact halted state. We validate Start&Stop using a well-established PhysiBoSS 2.0 tumor spheroid 3T3 mouse fibroblasts use case under tumor necrosis factor (TNF) stimulation, demonstrating that it preserves the simulator's original behavior while enabling interactive, real-time configuration changes that facilitate the exploration of diverse and adaptive treatment strategies. By enhancing flexibility and user interaction, Start&Stop makes PhysiCell and PhysiBoSS 2.0 more akin to real *in vitro* scenarios, thus expanding the range of potential simulations and advancing more effective protocol development in a variety of applications.

Keywords: Computational Biology, Modeling, In Silico, Simulation, Multi-Scale

1 Background

In computational biology, *in silico* simulators are vital for exploring and understanding the behavior of complex biological systems [2]. Hybrid multi-level simulators integrate multiple layers of biological complexity, providing deeper insights into emergent patterns comprising biological properties and behaviors at different scales and allowing the examination of population dynamics in various environments and under different conditions [4][11]. `PhysiCell` and `PhysiBoSS 2.0` are exemplary multi-level simulation tools [13][5]. For example, in *in silico* pharmacological research, these simulators help users better understand and test drug administration schemas more quickly and cost-effectively than traditional trial-and-error methods [14].

`PhysiCell` is a multicellular simulation framework utilizing various solvers, such as a diffusion and transport solver for simulating chemical microenvironment interactions with cells, a mechanical solver for simulating intercellular mechanical interactions, and additional components for cellular processes like growth, division, cell differentiation, and death [9]. To efficiently simulate secretion, diffusion, uptake, and decay of multiple substrates in large three-dimensional (3D) microenvironments, `PhysiCell` employs the `BioFVM` solver [8].

`PhysiBoSS 2.0` extends `PhysiCell` by integrating it with `MaBoSS`, a continuous-time Markovian simulator that models intracellular signaling networks and regulatory processes within cells [16]. It provides a broad range of implemented use cases, from drug treatment analysis of tumors to cell differentiation [13].

In the context of multi-scale simulators like `PhysiCell` and `PhysiBoSS 2.0`, real-time monitoring and control over cell conditions are fundamental for developing effective protocols to simulate cell behavior across various applications, from biofabrication [7][10][3] to *in vitro* pharmacological testing [17]. For the exploration and testing of pharmacological protocols, the ability to monitor the simulation's behavior and adjust protocol settings after the simulation has started is crucial. This capability would allow real-time adjustments to the current protocol to enhance its effectiveness and enable the exploration of new protocols, making step-by-step decisions as needed. Both `PhysiCell` and `PhysiBoSS 2.0` support easy customization of various parameters, such as simulation duration, set of administered stimuli, and initial position of cells. However, monitoring and controlling the simulation once it has started is impossible. In the context of *in silico* pharmacological research, this limitation prevents real-time monitoring of simulated drug administration. It reduces the range of administration protocols that can possibly be explored and tested.

2 Implementation

To overcome this limitation, this paper introduces the `Start&Stop` add-on for `PhysiCell` (automatically adaptable to `PhysiBoSS 2.0`), providing:

- *Multi-level state preservation*: allows users to pause a simulation and restart it from the same state it stopped.
- *Multi-modal stop control*: Both simulation time and predefined cell conditions can trigger the pause.

This mechanism allows changing the configuration parameters before restarting the simulation, enhancing user control with the simulator during the simulation, and enabling the interactive and adaptive exploration of a broader range of stimulation protocols.

The `Start&Stop` add-on allows users to pause and restart a simulation, preserving its complex state and covering, as defined below, all relevant model layers:

- **Cellular aggregate conformation**: an agent-based model at the core of `PhysiCell`, implemented in `C++`, of all cells, their states, and their positions.
- **Intracellular states**: If using `PhysiBoSS 2.0`, boolean models of the intracellular regulation network within each cell and their states. The `MaBoSS` add-on embedded in `PhysiBoSS 2.0` covers this layer.
- **Microenvironment**: reaction-diffusion Physical Differential Equations (PDEs) models, implemented in `C++`, represent the 3D environment, divided into 3D voxels, each one reporting concentration values of all the relevant diffusing substances.

Figure 1 shows a schema of the integration into `PhysiCell` and `PhysiBoSS 2.0` of the `Start&Stop` add-on. The `Start&Stop` add-on relies on three primary operations

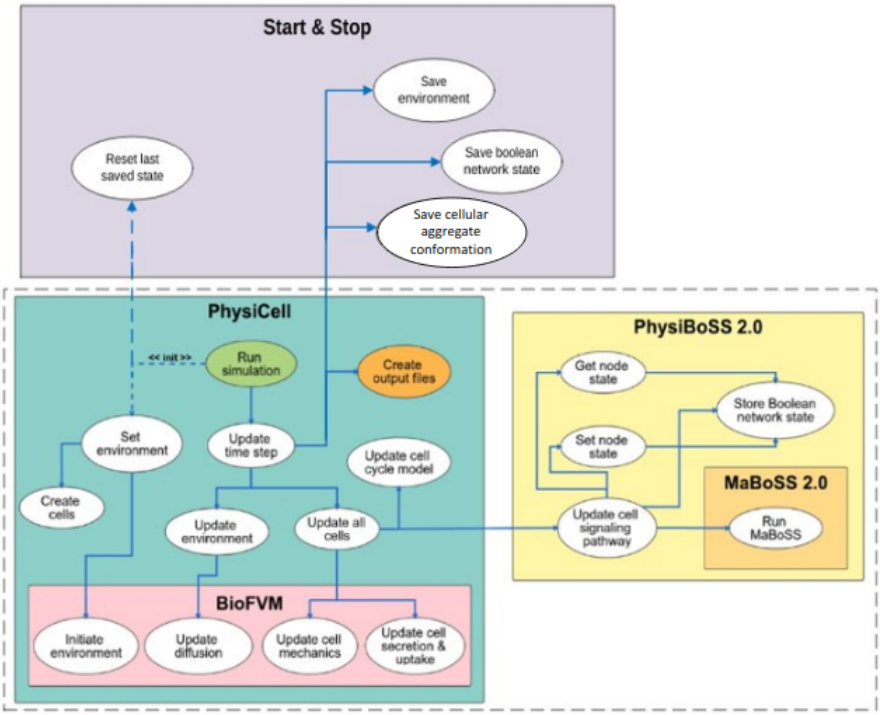

Fig. 1 *Integration of the `Start&Stop` add-on:* This schema represents the integration of the `Start&Stop` add-on into the `PhysiCell` and `PhysiBoSS 2.0` framework. The dashed rectangle represents the original `PhysiCell`, and `PhysiBoSS 2.0` framework (adapted from [13]).

detailed in the following sections: *current state saving* (Section 2.1), *auto-stop* (Section 2.2), and *state restoring* (Section 2.3).

2.1 Current state saving

In order to save the current simulation state, the `Start&Stop` add-on implements streaming operators in the simulator's core. These operators, implemented in `C++`, are included extensively in the `PhysiCell` classes such as `cells`, `cell container`, `custom`, `phenotype`, and `microenvironment`, as well as in `MaBoSS`. The `Start&Stop` add-on saves several files containing:

- Information about the *microenvironment*, with each voxel position and the concentration of all substances along with several metadata entries;
- The state of the *cells*;
- The cells' *positions* with x, y, and z coordinates for all the cells in the 3D environment;
- `PhysiCell` 's global parameters, such as the current time, the next saving time, and other parameters;
- If using `PhysiBoSS 2.0`, each cell's intracellular state is represented as the state of a Boolean model.

Importantly, all these files are *human-readable* for straightforward interpretation and automatically saved during the execution of the simulation at a frequency that the user can set.

2.2 Auto-stop

Each use case in `PhysiCell` and `PhysiBoSS 2.0` includes the `custom.cpp` file, enabling `PhysiCell` and `PhysiBoSS 2.0` users to define custom functions. The `Start&Stop` add-on allows the user to customize the function `auto_stop` in the `custom.cpp`, to implement its own *monitoring system* and *stop conditions*. An example of this function is provided in the `PhysiBoSS 2.0` sample use case `spheroid_tnf_model`, located in the file `sample_project_intracellular/boolean/spheroid_tnf_model /custom_modules/custom.cpp`. In this example, the monitoring system can track either the number of cells resistant to the treatment or the number of alive cells. The stopping condition in both cases consists of a threshold for the two types of cells mentioned earlier.

2.3 State restoration

When the `Start&Stop` activates, the program reads all the saved files during the simulation setup. It restores all objects to their exact state at the previous simulation stop through the streaming operator, also implemented in `C++`. In this way, the multiscale state of the entire simulator is restored, allowing the simulation to continue without inconsistencies. Figure 2 describes the steps followed by the `Start&Stop` add-on in the context of a generic simulation.

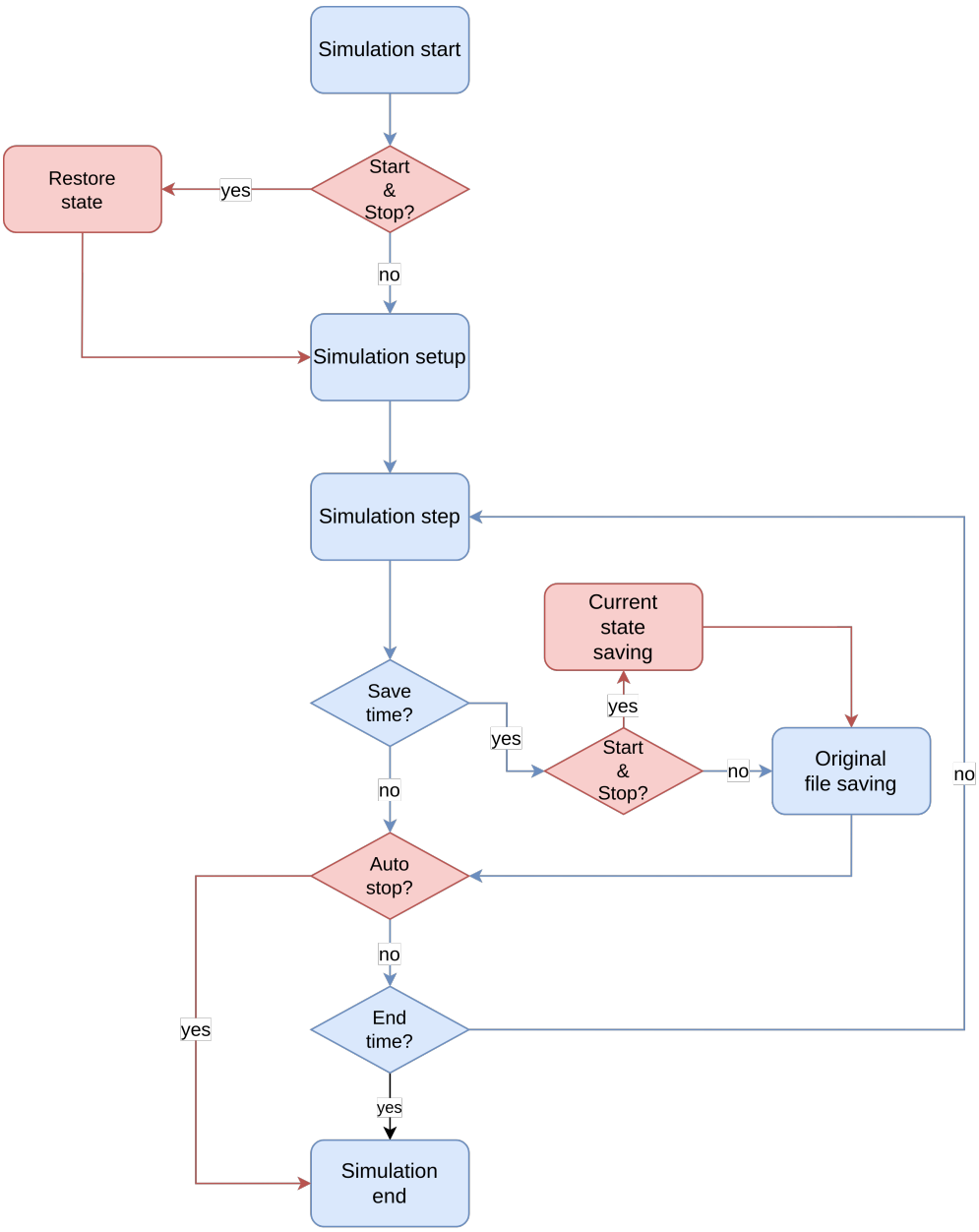

Fig. 2 *`Start&Stop` add-on functioning for a simulation.* This flow chart illustrates a generic simulation in `PhysiCell` and `PhysiBoSS 2.0`, with the inclusion of the `Start&Stop` steps. The blue steps represent the native `PhysiCell` and `PhysiBoSS 2.0` functioning, while the red ones represent the new `Start&Stop` add-on functionalities.

To set all the parameters for the simulation, `PhysiCell` and `PhysiBoSS 2.0` use an `XML` file. The `Start&Stop` add-on extends this file to contain two more flags: `start_and_stop` and `auto_stop`. To enable the `Start&Stop` add-on, the user must set these flags to `true`. At each save point of `PhysiCell` and `PhysiBoSS 2.0`, all necessary information described in Section 2.1 are saved and ready for use in subsequent simulations. The two flags are kept separate to use both functionalities independently, enabling simulations to run from a starting point without later stopping or stopping without restarting. The latter shows that the `Start&Stop` add-on is *fully integrated* into `PhysiCell` and `PhysiBoSS 2.0` and provides remarkable usage flexibility.

3 Results and Discussion

In *in silico* pharmacology, real-time monitoring and interaction with a therapy protocol are crucial for comprehensively understanding the behavior and effectiveness of drug administration. This approach provides a more complete study and a deeper insight into drug administration dynamics. To demonstrate the functionalities of the `Start&Stop` add-on, we selected a well-established and validated `PhysiBoSS 2.0` use case that examines the cellular dynamics of a tumor spheroid under various TNF regimes [12]. This multi-scale model focuses on 3T3 fibroblast spheroids to explore the complex dynamics observed during *in vitro* drug administration. By default, drug administration occurs in a pulsed manner, where TNF is delivered in pulses with a defined interval, referred to as the *period*. Pulse administration occurs at a fixed *concentration* and *duration*. Users can adjust all key parameters before launching the simulation; however, the current framework restricts simulations to scenarios in which all stimuli share fixed *period*, *duration*, and *concentration* values along the entire simulation. For the tumor spheroid use case, different protocols yield varying outcomes, defining the cell state in terms of the number of *alive*, *necrotic*, and *apoptotic* cells (see Section 2.1).

This work provides two demonstrative tests to show both the seamless integration with `PhysiCell` and `PhysiBoSS 2.0` and the relevance of the flexibility offered by the `Start&Stop` add-on.

The first test aimed to demonstrate that the `Start&Stop` add-on inclusion provides the intended functionality while preserving consistency with the original behavior of the simulator. Since both `PhysiCell` and `PhysiBoSS 2.0` are stochastic simulators, the outcome of the same simulation can vary from one trial to another, making it challenging to determine if activating the `Start&Stop` feature introduces any differences. To address this, we estimated the number of simulations needed to ensure the sample mean differs from the actual distribution mean by no more than ± 10 *alive* cells at the simulation's conclusion, using a method based on the Standard Error of the Mean [15], which yielded several 30 simulations. To be more conservative, we considered 50 simulations. As the drug administration scheme for testing, we executed a simulation with a TNF injection of 10 minutes every 150 minutes at a concentration of 0.005 TNF/μm^3 50 times and another set of 50 simulations stopping at each TNF administration period. Figure 3 reports the results. Figure 3 a, b, and c represent the box-plot

of the distribution of alive, necrotic, and *apoptotic* cells, respectively, at each time step where the simulations with the `Start&Stop` add-on were stopped and restarted.

Figure 3a and Figure 3b show that the behavior of both *alive* and *necrotic* cells is consistent when comparing the simulations with and without the `Start&Stop` activation. Regarding the behavior of *apoptotic* cells (Figure 3c), it is still consistent. However, in both cases, the standard deviation of the distributions is inherently higher, especially in the middle of simulation courses (between 450min and 1050min). This behavior is because the trend of the *apoptotic* cells in this specific use case of `PhysiBoSS 2.0` show a distribution with a higher variance than that of *necrotic* and *alive* cells. Moreover, calculations of the minimum number of simulations to perform based on the *alive* cells alone; thus, the result does not necessarily hold for the other types of cells. Results of this comparison demonstrate that the `Start&Stop` feature does not alter the function of `PhysiBoSS 2.0`.

To provide an applicative example of the relevance of the `Start&Stop` feature in the context of *in vitro* drug administration protocol design, we demonstrate its use for managing the risk of an undesirable outcome of a pharmacological test that is, drug-induced resistance in tumoral cells. In the context of 3T3 fibroblast tumor spheroids, tumor cells can survive treatment in two cases: either the stimulation is not strong enough to induce necrosis, or the cells develop resistance to the effects of the cytokines [6]. Both phenomena are modeled in this `PhysiBoSS 2.0` use case.

In Figure 4, both cases are shown. The figure illustrates a simulation with continuous TNF administration at a 0.1 TNF/μm^3 concentration. At some point, almost all the cells became resistant to the treatment, resulting in continuous growth of the living cells, as can be seen in Figure 4a. To identify resistant cells, we used the state of their Boolean model, marking as resistant all stable states that result in survival as described in [6].

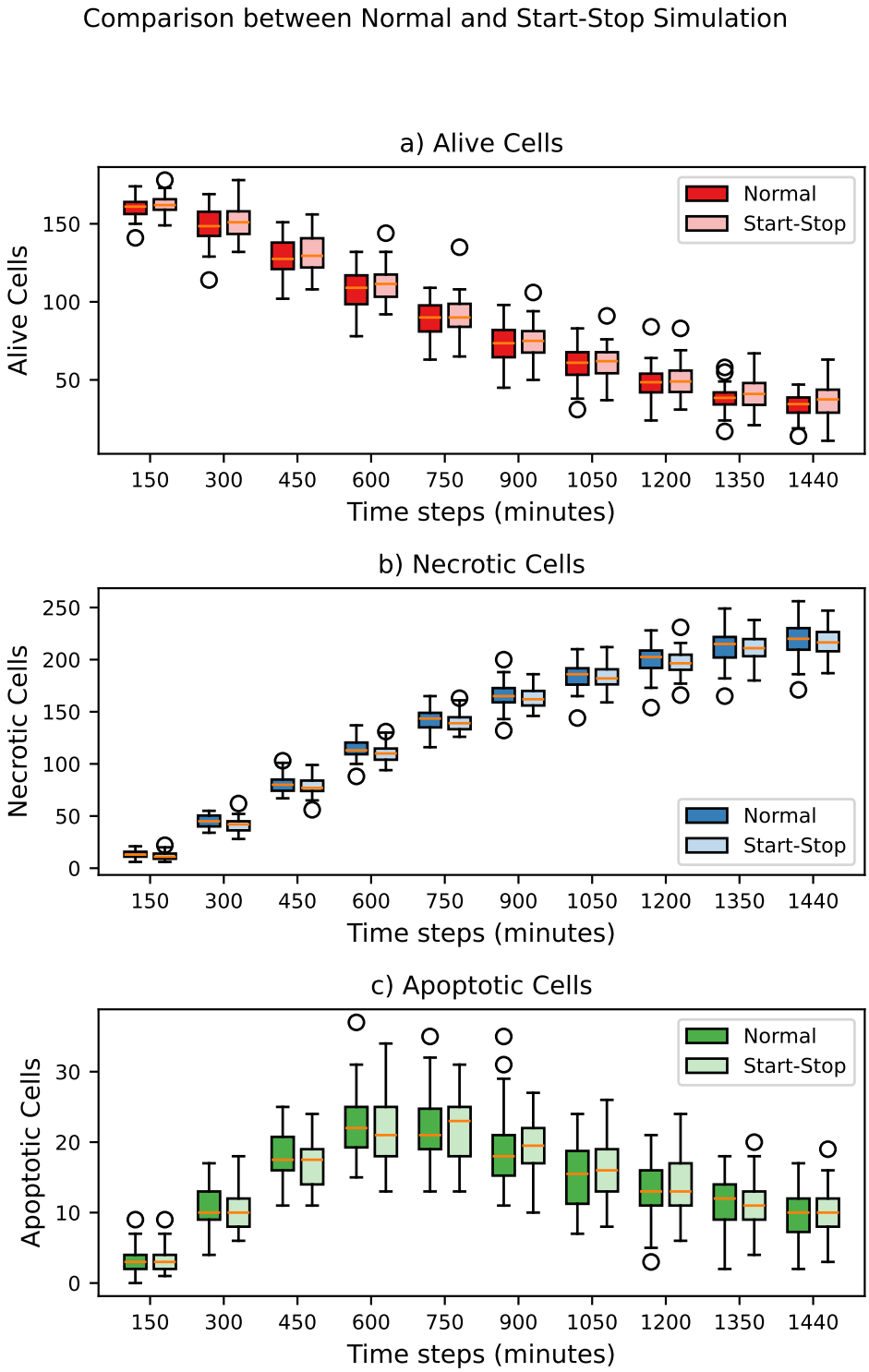

Fig. 3 *Simulation Evolution with and without the* `Start&Stop` *add-on:* a) Comparison between simulations with and without activation of the `Start&Stop` add-on, for a simulation configuration with an administration every 150 minutes of 0.005 TNF/μm^3 lasting 10 minutes, in particular for *alive* cells, b) *necrotic* cells, and c) *apoptotic* cells. Box plots are obtained by running 50 different simulations. In all simulations for the `Start&Stop` test, a stop and a restart occurred at each TNF administration time step.

To showcase the capabilities of the `Start&Stop` add-on, we used the `auto-stop` function (see Section 2.2) with the number of resistant cells as a parameter to monitor and counteract cell resistance, stopping the simulation when more than 80% of the living cells acquired a state recognized as resistant. Figure 4b shows that the simulation auto-stopped after 210 simulated minutes. This event allowed the user to adjust drug administration, providing stimulation every 150 minutes, with 0.005 TNF/μm^3 for 10 minutes. Rapidly, the cells began reducing their resistance, and the drug became effective.

However, *in vitro* drug administration can also fail due to insufficient stimulation. This scenario is illustrated in Figure 4c, where the tumor was administered 0.005 TNF/μm^3 every 600 minutes, lasting 10 minutes. It is observable that the *alive* cells are not developing resistance as they did in Figure 4a, but the treatment is still unsuccessful. To monitor this issue, another`auto-stop` function was implemented, taking as parameters the current amount of *alive* cells and determining if the living cells were continuously growing. Figure 4d shows that the simulation stopped at simulated minute 270. We then changed the drug administration scheme to a stimulation every 150 minutes with 0.005 TNF/μm^3 for 10 minutes. As a result, the *alive* cells decreased in number, demonstrating the effectiveness and flexibility of the `Start&Stop` add-on.

The inclusion of the `Start&Stop` add-on significantly enhances interaction and control over the simulation, making the simulator more flexible and akin to real *in vitro* scenarios where it is possible to monitor the state of the cells and intervene if needed. Moreover, this add-on allows users to expand the range of possible simulations. It removes the constraint of periodic, rigid, and *a priori*-defined stimulation schemes, enabling real-time customization and monitoring of the simulation.

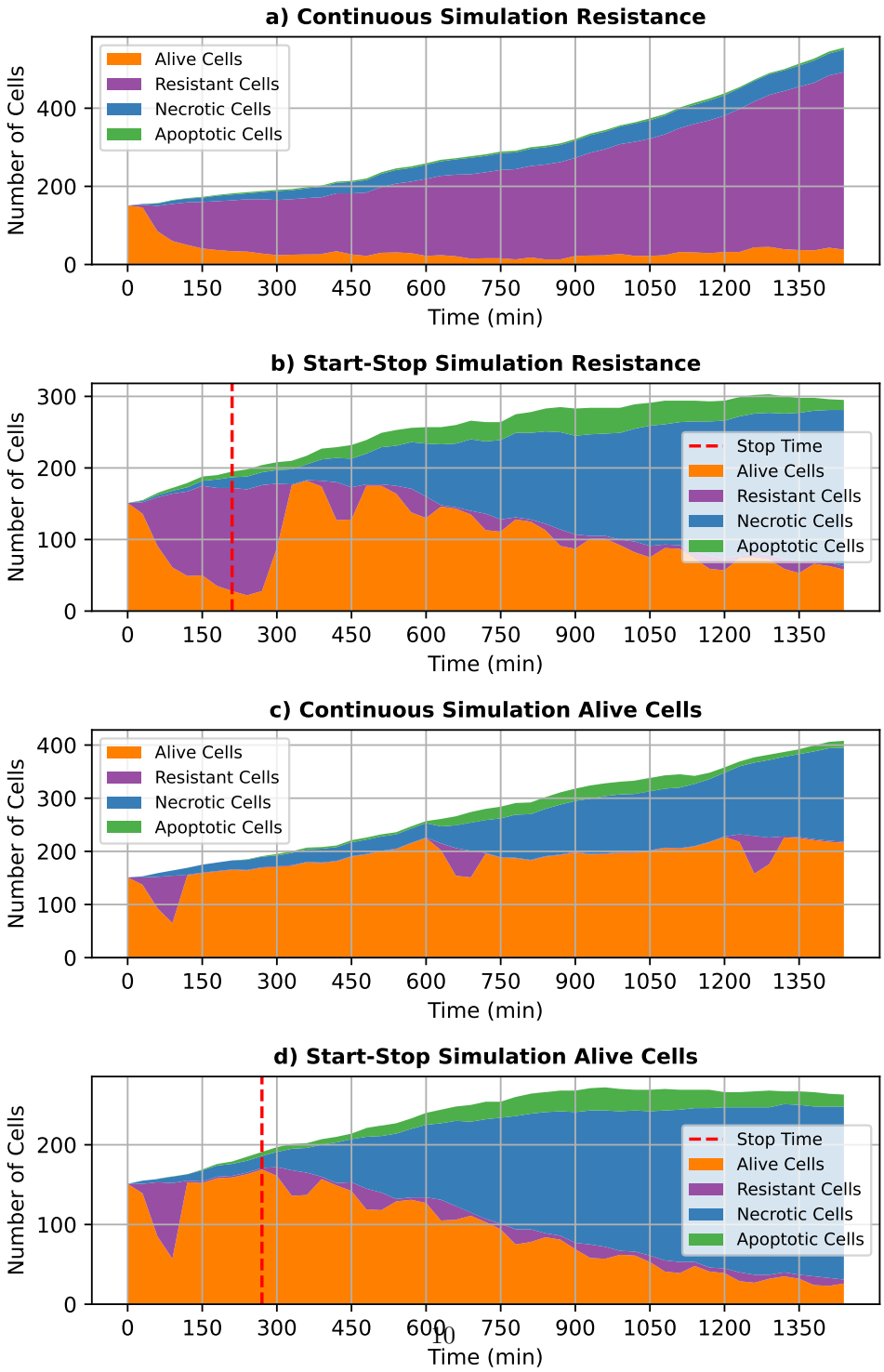

Fig. 4 *Application of the* `Start&Stop` *add-on in an in vitro drug administration Simulation*: a) Behavior of apoptotic, necrotic, alive, and resistant cells during the *in vitro* drug administration with continuous administration of 0.1 TNF/μm^3 over time. Cells develop resistance to the therapy. b) Behavior of alive, apoptotic, necrotic, and resistant cells in a simulation starting similarly to Figure 4a, but stopped at minute 210 and restarted with administration every 150 minutes of 0.005 TNF/μm^3 for 10 minutes. c) Behavior of apoptotic, necrotic, alive, and resistant cells in a simulation with administration every 600 minutes of 0.005 TNF/μm^3 for 10 minutes, where the *in vitro* drug administration efficacy is insufficient. d) Behavior of alive, apoptotic, necrotic, and resistant cells in a simulation starting similarly to Figure 4c, but stopped at minute 270 and restarted with administration every 150 minutes of 0.005 TNF/μm^3 for 10 minutes.

4 Conclusions

In this paper, we presented the *Start&Stop* add-on for `PhysiCell` and `PhysiBoSS 2.0`, enabling the user to stop and restart the simulation from the same state at the point of interruption while also providing the capability to interact with the parameters of the simulation such as *duration*, *period* and *concentration* of an *in vitro* drug administration. This capability enables a broader set of possible parameters, increasing user control and interaction with the simulation. The *Start&Stop* add-on covers all use cases provided by `PhysiCell` and `PhysiBoSS 2.0`, both existing and custom ones. The *Start&Stop* add-on allows the user to monitor the simulation and perform necessary actions in real-time.

This work demonstrates that the *Start&Stop* add-on does not change the behavior of a simulation when stopping and restarting, thereby maintaining the validated multi-scale simulation state of the original `PhysiBoSS 2.0` use case implementation. It was also tested in terms of utility, showing that it can help improve the capability of exploring possible treatments for the well-known and validated tumor spheroid use case.

This work contributes to the computational ecosystem of hybrid and multi-level models for systems biology by enhancing their representation and prediction power by integrating interactive and automated simulation control in the `PhysiCell` and `PhysiBoSS 2.0` simulation framework, thereby improving simulation configuration in terms of control and flexibility. This not only enables users to explore and learn more from simulation experiments, but it also paves the way for more flexibility in the programmatic usage of these simulators for the simulation-driven design and optimization of biological processes [7][10], complementing other ongoing efforts to make the latter more computationally feasible, for instance, through meta-modeling approaches [1].

In conclusion, the *Start&Stop* add-on can significantly expand the capabilities of the `PhysiCell` and `PhysiBoSS 2.0` simulator, making *in silico* experiments closer to *in vitro* experiments.

5 Availability and requirements

5.1 Project name

start-and-stop-test

5.2 Project home page

https://github.com/smilies-polito/start-and-stop-test

5.3 Operating systems

Platform independent.

5.4 Programming languages

`C++` for the add-on implementation.
`Python` for the tests implementation.

5.5 Other requirements

Singularity release 3.10.2
Go version 1.18.4
This entire project can be executed on a singularity container with only these two dependencies. All the other dependencies to run the code independently are listed in the `/source/optimizing_physiboss.def file`

5.6 License

MIT licence.

5.7 Any restrictions to use by non-academics

Not applicable.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not Applicable.

Availability of data and materials

The set of functions required for the `Start&Stop` functionality is implemented in `C++` and contained in the `add-ons/start_and_stop` folder, as required by `PhysiCell`. These functions need to be called in the `main.cpp` file of the selected use case. Implemented at the core of `PhysiCell`, it is possible to use the `Start&Stop` add-on for all `PhysiCell` and `PhysiBoSS 2.0` use cases. The full implementation and the script for reproducing the tests provided in this paper are publicly available in GitHub: https://github.com/smilies-polito/start-and-stop-test. The add-on code is currently under review by the authors of the `PhysiCell` simulator as part of a pull request to the `PhysiCell` original repository:https://github.com/MathCancer/PhysiCell.

Competing interests

The authors declare no competing interests.

Funding

This study was carried out within the "SAISEI - Multi-Scale Protocols Generation for Intelligent Biofabrication" [Prot. 20222RT5LC] project and the "BIGMECH - Knowledge-generation framework combining a novel Biomimetic Investigation platform and hiGh throughput screening for unravelling bone MECHanotransduction mechanisms in view of precision orthopedic medicine" [Prot. 2022F7M2A] project –

both funded by European Union – Next Generation EU within the PRIN 2022 program (D.D. 104 - 02/02/2022 Ministero dell'Università e della Ricerca). This manuscript reflects only the authors' views and opinions and the Ministry cannot be considered responsible for them. This publication is part of the project PNRR which has received funding from the MUR–DM 118/2023.

Authors' contributions

RS developed the code. RS and RB wrote the manuscript. RB, AS, and SDC supervised the work and reviewed the manuscript.

Acknowledgment

We would like to thank Vincent Noël, Miguel Ponce de Leon, Arnau Montagud, and all the `PhysiCell` and *PhysiBoSS 2.0* teams for their insightful suggestions and feedback.

References

[1] Marco P Abrate, Riccardo Smeriglio, Roberta Bardini, Alessandro Savino, and Stefano Di Carlo. Fast and accurate lstm meta-modeling of tnf-induced tumor resistance in vitro. *bioRxiv*, pages 2024–08, 2024.

[2] Roberta Bardini, Alfredo Benso, Gianfranco Politano, and Stefano Di Carlo. Nets-within-nets for modeling emergent patterns in ontogenetic processes. *Computational and Structural Biotechnology Journal*, 19:5701–5721, 2021.

[3] Roberta Bardini and Stefano Di Carlo. Computational methods for biofabrication in tissue engineering and regenerative medicine-a literature review. *Computational and Structural Biotechnology Journal*, 2024.

[4] Roberta Bardini, Gianfranco Politano, Alfredo Benso, and Stefano Di Carlo. Multi-level and hybrid modelling approaches for systems biology. *Computational and structural biotechnology journal*, 15:396–402, 2017.

[5] Ezio Bartocci and Pietro Lió. Computational modeling, formal analysis, and tools for systems biology. *PLoS computational biology*, 12(1):e1004591, 2016.

[6] Laurence Calzone, Laurent Tournier, Simon Fourquet, Denis Thieffry, Boris Zhivotovsky, Emmanuel Barillot, and Andrei Zinovyev. Mathematical modelling of cell-fate decision in response to death receptor engagement. *PLoS computational biology*, 6(3):e1000702, 2010.

[7] Alberto Castrignanò, Roberta Bardini, Alessandro Savino, and Stefano Di Carlo. A methodology combining reinforcement learning and simulation to optimize the in silico culture of epithelial sheets. *Journal of Computational Science*, 76:102226, 2024.

[8] Ahmadreza Ghaffarizadeh, Samuel H Friedman, and Paul Macklin. Biofvm: an efficient, parallelized diffusive transport solver for 3-d biological simulations. *Bioinformatics*, 32(8):1256–1258, 2016.

[9] Ahmadreza Ghaffarizadeh, Randy Heiland, Samuel H Friedman, Shannon M Mumenthaler, and Paul Macklin. Physicell: An open source physics-based

cell simulator for 3-d multicellular systems. *PLoS computational biology*, 14(2):e1005991, 2018.

[10] Leonardo Giannantoni, Roberta Bardini, and Stefano Di Carlo. A methodology for co-simulation-based optimization of biofabrication protocols. In *International Work-Conference on Bioinformatics and Biomedical Engineering*, pages 179–192. Springer, 2022.

[11] Zhiwei Ji, Ke Yan, Wenyang Li, Haigen Hu, and Xiaoliang Zhu. Mathematical and computational modeling in complex biological systems. *BioMed research international*, 2017(1):5958321, 2017.

[12] Robin EC Lee, Mohammad A Qasaimeh, Xianfang Xia, David Juncker, and Suzanne Gaudet. Nf-κb signalling and cell fate decisions in response to a short pulse of tumour necrosis factor. *Scientific Reports*, 6(1):39519, 2016.

[13] Miguel Ponce-de Leon, Arnau Montagud, Vincent Noël, Annika Meert, Gerard Pradas, Emmanuel Barillot, Laurence Calzone, and Alfonso Valencia. Physiboss 2.0: a sustainable integration of stochastic boolean and agent-based modelling frameworks. *npj Systems Biology and Applications*, 9(1):54, 2023.

[14] Fateme Pourhasanzade, SH Sabzpoushan, Ali Mohammad Alizadeh, and Ebrahim Esmati. An agent-based model of avascular tumor growth: Immune response tendency to prevent cancer development. *Simulation*, 93(8):641–657, 2017.

[15] Frank E Ritter, Michael J Schoelles, Karen S Quigley, and Laura Cousino Klein. Determining the number of simulation runs: Treating simulations as theories by not sampling their behavior. *Human-in-the-loop simulations: Methods and practice*, pages 97–116, 2011.

[16] Gautier Stoll, Barthélémy Caron, Eric Viara, Aurélien Dugourd, Andrei Zinovyev, Aurélien Naldi, Guido Kroemer, Emmanuel Barillot, and Laurence Calzone. Maboss 2.0: an environment for stochastic boolean modeling. *Bioinformatics*, 33(14):2226–2228, 2017.

[17] Errikos Streviniotis, Nikos Giatrakos, Yannis Kotidis, Thaleia Ntiniakou, and Miguel Ponce de Leon. Optimizing resource allocation for tumor simulations over hpc infrastructures. In *2023 IEEE 10th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10. IEEE, 2023.