






cuProCell: GPU-Accelerated Analysis of Cell Proliferation With Flow Cytometry Data

Marco S. Nobile , Member, IEEE, Eric Nisoli, Thalia Vlachou , Simone Spolaor , Member, IEEE, Paolo Cazzaniga , Member, IEEE, Giancarlo Mauri , Member, IEEE, Pier Giuseppe Pelicci , and Daniela Besozzi , Member, IEEE

Abstract—The investigation of cell proliferation can provide useful insights for the comprehension of cancer progression, resistance to chemotherapy and relapse. To this aim, computational methods and experimental measurements based on *in vivo* label-retaining assays can be coupled to explore the dynamic behavior of tumoral cells. ProCell is a software that exploits flow cytometry data to model and simulate the kinetics of fluorescence loss that is due to stochastic events of cell division. Since the rate of cell division is not known, ProCell embeds a calibration process that might require thousands of stochastic simulations to properly infer the parameterization of cell proliferation models. To mitigate the high computational costs, in this paper we introduce a parallel implementation of ProCell’s simulation algorithm, named cuProCell, which leverages Graphics Processing Units (GPUs). Dynamic Parallelism was used to

efficiently manage the cell duplication events, in a radically different way with respect to common computing architectures. We present the advantages of cuProCell for the analysis of different models of cell proliferation in Acute Myeloid Leukemia (AML), using data collected from the spleen of human xenografts in mice. We show that, by exploiting GPUs, our method is able to not only automatically infer the models’ parameterization, but it is also $237\times$ faster than the sequential implementation. This study highlights the presence of a relevant percentage of quiescent and potentially chemoresistant cells in AML *in vivo*, and suggests that maintaining a dynamic equilibrium among the different proliferating cell populations might play an important role in disease progression.

Index Terms—GPGPU computing, high-performance computing, ProCell, cell proliferation, stochastic simulation, parameter estimation, FST-PSO, acute myeloid leukemia.

Manuscript received November 15, 2019; revised March 18, 2020 and June 7, 2020; accepted June 17, 2020. Date of publication June 29, 2020; date of current version November 5, 2020. This work was supported in part by European Research Council Advanced Grant - 2013 Ideas Program [EC-FP7-ERC-InMec 341131], in part by Associazione Italiana per la Ricerca sul Cancro [AIRC 2013: IG-14216], and in part by the SYSBIO/ISBE.IT Centre of Systems Biology. (Marco S. Nobile and Eric Nisoli contributed equally to this work.) (Corresponding authors: Pier Giuseppe Pelicci; Daniela Besozzi.)

Marco S. Nobile is with the Department of Industrial Engineering and Innovation Sciences of the TU/e, Eindhoven University of Technology, Eindhoven 5612 AZ, The Netherlands, and with the Department of Informatics, Systems and Communication, University of Milano-Bicocca, 20126 Milano, Italy, and also with the SYSBIO/ISBE.IT Centre of Systems Biology, 20126 Milano, Italy (e-mail: m.s.nobile@tue.nl).

Eric Nisoli and Simone Spolaor are with the Department of Informatics, Systems and Communication, University of Milano-Bicocca, 20126 Milano, Italy (e-mail: e.nisoli1@campus.unimib.it; simone.spolaor@unimib.it).

Thalia Vlachou is with the Department of Experimental Oncology, IEO, European Institute of Oncology IRCCS, 20141 Milano, Italy (e-mail: thalia.vlachou@ieo.it).

Paolo Cazzaniga is with the Department of Human and Social Sciences, University of Bergamo, 24129 Bergamo, Italy, and also with the SYSBIO/ISBE.IT Centre of Systems Biology, 20126 Milano, Italy (e-mail: paolo.cazzaniga@unibg.it).

Giancarlo Mauri and Daniela Besozzi are with the Department of Informatics, Systems and Communication, University of Milano-Bicocca, 20126 Milano, Italy, and with the SYSBIO/ISBE.IT Centre of Systems Biology, 20126 Milano, Italy, and also with the Bicocca Bioinformatics Biostatistics and Bioimaging Centre (B4), 20854 Monza, Italy (e-mail: giancarlo.mauri@unimib.it).

Pier Giuseppe Pelicci is with the Department of Experimental Oncology, IEO, European Institute of Oncology IRCCS, 20141 Milano, Italy, and also with the Department of Oncology and Hemato-Oncology, University of Milan, 20122 Milano, Italy (e-mail: piergiuseppe.pelicci@ieo.it).

Digital Object Identifier 10.1109/JBHI.2020.3005423

I. INTRODUCTION

THE comprehension of cell proliferation processes is of pivotal importance to properly address the recurrence or resistance to therapies of many complex diseases, such as cancer. Though, well-established experimental methods may not be enough to quantify and assess the kinetics of cell division events, owing to the complexity of the analysis and the interpretation of data. Moreover, such methods are not easily applicable to *in vivo* contexts.

To overcome these limitations, mathematical modeling can be exploited to investigate cell population dynamics [1]. Several approaches have been proposed in the literature to describe different aspects of cell proliferation. For example, Ordinary Differential Equations [2] are commonly employed to study (sub)populations dynamics, possibly paired with Partial Differential Equations [3], [4] to take into account different physiological features, e.g., cell age and contents; cellular automata or agents are used to describe interactions among cells, and the emergent self-organizing dynamics [5]; combination and/or variations or the above can be also found to model subclonal evolution [6]. These modeling approaches often require the use of sophisticated steps for the pre-processing of experimental data, like noise filtering and peak detection. Even though their computational costs can sometimes be mitigated by deriving the analytical solution, the models based on differential equations can also be difficult to interpret for biologists and clinicians, thus

hampering their usefulness [7], and they might as well require the estimation of a large number of parameters.

In this work, we build upon a previously developed and experimentally validated framework to investigate cell proliferation, called ProCell [8], which belongs to the class of stochastic Markov-chain based methods [1]. ProCell represents a valid alternative to classic modeling approaches—which were successfully used for the investigation of cell division and cell death processes in specific contexts (e.g., CFSE-labeled T lymphocytes [3])—since it allows to overcome the need for any assumption, as generally done in other models. Precisely, ProCell is characterized by the following features: (i) it explicitly takes into account the stochasticity of cell division times *in vivo*, which are considerably different with respect to *in vitro* studies; (ii) it assumes a specifically designed laboratory protocol—a genetic label that is stoichiometrically distributed in cells—that allows to measure the symmetric division of cell fluorescence; (iii) it is able to deal with biological systems characterized by a few cells; (iv) it is designed to maximize the interpretability of the mathematical model, while minimizing the number of parameters that need to be inferred; (v) it prevents the need for any pre-processing step on the experimental data; (vi) it does not require the choice of any hyper-parameter for the simulation (e.g., error tolerances or step size, typical of numerical integration methods).

ProCell can be used to investigate heterogeneous sub-populations of asynchronously dividing cells, monitored *in vivo* by means of flow cytometry data, even in the case of sub-populations characterized by the presence of only a few cells that are above the threshold of auto-fluorescence, e.g., in the case of experiments that span over days or a few weeks. With ProCell, users can gain insights on subclonal population proportions inside the analyzed samples, together with their division times. Models in ProCell are easily interpretable, characterized by a few parameters, and expressed in simple terms of cell proportions and division intervals. Moreover, thanks to its user-friendly Graphical User Interface that has been redesigned and improved with respect to the previous version [9], it is possible to automatically calibrate the parameters of cell population models by means of FST-PSO [10], a settings-free version of Particle Swarm Optimization [11].

The computational investigations carried out with ProCell can be computationally expensive, especially for the parameter estimation task, which generally requires tens of thousands independent simulations to accurately identify an appropriate model parameterization. In this paper we tackle this computational problem by presenting cuProCell, a parallel implementation of the stochastic simulation method at the basis of ProCell that exploits NVIDIA's *Compute Unified Parallel Architecture* (CUDA) to offload the calculations of cell division events onto the Graphics Processing Unit (GPU). cuProCell leverages Dynamic Parallelism (DP), that is, the possibility of recursively launching new computing kernels from threads running on the GPU. The rationale of this specific solution is that each cell belonging to the initial population is assigned to a specific thread in the GPU, which in turn can spawn a pair of new threads as soon as a cell division event occurs.

To assess the simulation capabilities and the performance of cuProCell, in this work we investigate cell proliferation processes in Acute Myeloid Leukemia (AML), one of the most widespread hematological malignancies, characterized by high mortality rates and relapse: whilst around 60–85% of patients below the age of 60 respond to therapy, and initially achieve complete remission, the majority of them eventually relapses and will not survive [12]. The recurrence of the disease—and its resistance to chemotherapy—could be attributed to AML's inherent intra-tumor heterogeneity. It is known that a subset of Leukemic Stem Cells can enter quiescence and serve as a functional reservoir of the tumor [13], being capable of evading current chemotherapeutic treatments that mainly target highly proliferative cells [14].

Here, considering the mathematical models of cell proliferation defined in [8] for AML cell populations taken from the bone marrow, we performed the parameter calibration and validation using as target data the fluorescence measurements of cells taken from the spleen, recorded during “pulse and chase” experiments adapted for *in vivo* cell division tracking of leukemic cell sub-populations. Specifically, an inducible lentiviral vector was used to genetically label human AML cells with histone 2B-green fluorescent protein (H2B-GFP; pulse) and, at the same time, enable the conditional suppression of transgene expression upon doxycycline (dox) administration (chasing). According to our results, the method efficiently estimated a parameterization that accurately fits with the target data. Thanks to GPU acceleration, cuProCell allows to speed-up the simulation process up to $237\times$ (using a video-card NVIDIA Tesla V100) with respect to the sequential Python implementation. This result is obtained thanks to the use of DP, which allows to achieve an optimal occupancy of the GPU's computing resources.

The paper is structured as follows. In Section II we briefly describe the experimental protocol, the ProCell framework and the FST-PSO algorithm. The implementation of cuProCell is described in detail in Section III. In Section IV we summarize and discuss the results of the parameter estimation for the models related to spleen data, and describe the computational advantages of cuProCell. Finally, we conclude the paper in Section V by describing some future developments and applications of our method.

II. MATERIALS AND METHODS

A. The Experimental Protocol

The experimental protocol is based on an *in vivo* label-retaining assay. In particular, we labelled the human leukemic blasts of a PDX model with the fusion protein H2B-GFP, by means of lentiviral infection. H2B-GFP can be successfully incorporated into the nucleosomes of the infected cells, producing a strong fluorescence signal that is localized in the nucleus. Notably, the expression of H2B-GFP is regulated by a Tet-Off inducible promoter: (i) H2B-GFP is constantly produced in the absence of any tetracycline derivatives; (ii) H2B-GFP production stops when the cells are exposed to tetracycline derivatives, such as dox. In the presence of dox (chasing period), H2B-GFP is equally distributed to the daughter cells upon each cell division

and non-labelled H2B is incorporated into the nucleosomes. As a result, the H2B-GFP fluorescence intensity is halved by each cell division, while non-dividing cells are expected to maintain their initial fluorescence levels throughout the chasing period [8], [15].

The infected blasts were isolated by fluorescence activated cell sorting (FACS) and transplanted into immunocompromised mice, which were monitored for disease engraftment and H2B-GFP expression in the peripheral blood (PB). When leukemia engraftment reached 20–30% in PB, we initiated the chasing by placing a cohort of mice on dox diet (625 mg/kg of pellets). Small groups of mice were sacrificed at selected time points after dox administration (days 0, 10 and 21) and the primary sites of leukemic infiltration were analyzed. In pathological conditions such as AML, the abnormal blasts accumulate primarily in the bone marrow (BM), subsequently invade in the PB and occasionally infiltrate other extramedullary tissues such as the spleen (SPL). Transplantation of AML specimens that highly engraft in murine hosts, as the one used in the present study, commonly results in splenic enlargement at advanced stages of the disease, a phenomenon linked to the overabundance of leukemic blasts and putative necrosis of the recipient BM [16]. For this reason, we decided to model and analyze the kinetics of leukemia proliferation separately in BM and SPL, as they may provide complementary views on the intrinsic organization of leukemic sub-populations.

The leukemic blasts derived from dox-treated (days 10 and 21) and untreated (day 0) animals were analyzed by flow cytometry. The cells were stained with anti-human CD45 antibodies, to distinguish the human blasts from the murine niche cells, and a negative control (non-infected GFP-negative AML cells; GFPneg) was used to set the autofluorescence threshold (above which cells are considered GFP-positive; GFPpos). The specific H2B-GFP fluorescence signal distribution of GFPpos cells was finally depicted as a histogram using FlowJo (v10.1 or higher) and exported in .txt format.

B. The ProCell Framework

ProCell is a framework for the modeling and stochastic simulation of proliferating cell systems [8], [9], consisting of various sub-populations characterized by different proportions and specific distributions of cell division rate. ProCell was designed to simulate the dynamic variations in cell fluorescence that can be acquired, e.g., by flow cytometry analyses. Thus, ProCell's input, which is mandatory to perform a stochastic simulation, is a fluorescence histogram $H(t)$, defined as a finite set of ordered pairs $H(t) = \{(\varphi_1, \psi_1), \dots, (\varphi_M, \psi_M)\}$, where M is the number of different fluorescence levels in the experimental measurement. For any $i = 1, \dots, M$, the values φ_i and ψ_i denote the i -th fluorescence level and the frequency of the i -th fluorescence level, respectively, measured at time t . Namely, ψ_i represents the number of cells whose measured fluorescence is equal to φ_i . $\varphi^H = (\varphi_1, \dots, \varphi_M)$ represents the vector of all fluorescence levels appearing in H .

To simulate the system evolution starting from the initial histogram $H(0)$, the user must define a model of cell proliferation,

characterized by different sub-populations (e.g., proliferating, quiescent) with specific characteristics: (i) the number of sub-populations, (ii) the proportions of the different sub-populations, (iii) the distribution of the division time for each sub-population, specified as a normal distribution $\mathcal{N}(\mu, \sigma)$, for some mean μ and standard deviation σ (see [8] for additional details and a discussion on the use of different distributions). These parameters are mandatory to perform the simulations, executed up to a specified maximum time t_{\max} in which cells are considered in the final population, unless their fluorescence levels fall below a given fluorescence threshold φ_{\min} (as explained below). Since the parameters drive the behavior of the cell system, they must be carefully selected to yield a final population fitting with the experimental data.

In ProCell, stochastic simulations are executed as follows. The algorithm begins by creating a population of cells characterized by a type, an initial fluorescence level (reflecting the initial histogram $H(0)$), and a timer that represents the waiting time before the cell's next division event (sampled using the normal distribution associated with its own cell type). All these newly created cells are represented by objects pushed onto a stack. According to the experimental protocol, every time a cell divides it generates two daughter cells with halved fluorescence levels. The simulation stack is updated accordingly: the parent cell is removed, and the newly created cells are pushed onto the stack. The sequence of cell divisions is iterated until one termination criterion is met: (i) the simulation time of the cell reaches t_{\max} , or (ii) the fluorescence levels of the daughter cells would fall below the imposed fluorescence threshold φ_{\min} . At the end of the process, a histogram of fluorescence levels of the final population of GFPpos cells is produced. A detailed description of the simulation algorithm used by ProCell can be found in [8].

In order to further simplify the creation, editing, simulation, calibration, and maintenance of models, we completely restructured the Graphical User Interface (GUI) around ProCell's core algorithms [9], achieving a user-friendly and intuitive tool (Fig. 1).

As previously mentioned, an accurate parameterization of the model is mandatory to perform a simulation of cell proliferation. The parameters estimation can be executed using the GUI, which embeds the use of Fuzzy Self-Tuning Particle Swarm Optimization (FST-PSO) [10] (Section II-C) to fit the user-specified experimental fluorescence histogram measured at any given time point.

C. Fuzzy Self-Tuning PSO

ProCell models require a specific parameterization in terms of division time distributions and cell sub-populations proportions. Since this information is generally unknown and difficult to measure by means of laboratory experiments, an indirect estimation methodology must be employed. Many computational intelligence methods were proposed to efficiently solve the parameter estimation (PE) problem [17]. However, such methods are generally characterized by multiple hyper-parameters that must be carefully set, as they ultimately determine the performance of the algorithm. To prevent this circumstance, ProCell was

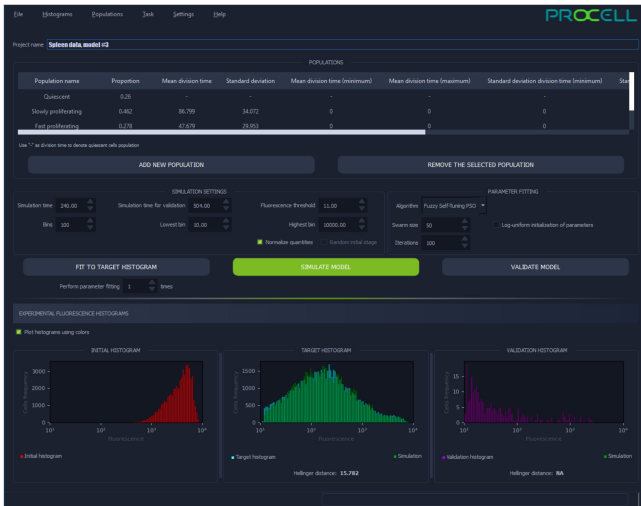


Fig. 1. ProCell's new Graphical User Interface.

coupled with FST-PSO [10], a settings-free variant of Particle Swarm Optimization that was shown to be effective in solving this specific optimization problem [17], [18].

When a PE is launched, FST-PSO creates a swarm of random particles in a D -dimensional search space, where D is the number of parameters to be estimated. In this process, each particle encodes a putative parameterization of the model. The quality of such parameterization can be assessed by means of a fitness function that associates a particle with a real value encoding the “goodness” of the fit. In this work, the core of the fitness function is the Hellinger distance [19] between simulated and the experimental fluorescence distributions (see, [8], [9] for additional information). The Hellinger distance is used in statistics to assess the difference between two distributions; thanks to the standardization and normalization steps that we perform [8], the smaller the Hellinger distance the higher the similarity between the simulation and the expected distribution, making it appealing to be used as fitness function in Computational Intelligence methods, and as a statistical measure to directly compare competing models.

In our tests, we used 50 particles and we run FST-PSO for 100 iterations: this adds up to 5000 simulations of putative parameterizations and leads to a huge running time on the CPU. In order to reduce the computational effort of the PE, we developed a parallel version of ProCell that exploit GPUs to simultaneously execute the simulations.

III. CUProCELL: GPU-POWERED ANALYSIS OF CELL POPULATION PROLIFERATION

cuProCell is a highly optimized parallel version of the original ProCell simulation algorithm. It was implemented using NVIDIA’s CUDA, an architecture that gives the programmer the possibility of offloading parts of the source code onto the GPU [20]. Specifically, in CUDA the developer must implement the kernels, that is, C/C++ functions that are launched from the host (i.e., the CPU) and executed on the device (i.e., the GPU).

When a kernel is launched, it is replicated in multiple copies named threads. Threads are organized in three-dimensional logical structures named blocks that are, in turn, organized in further structures named grids. In the GPU, a scheduler assigns the blocks to the available Streaming Multiprocessors; blocks are asynchronously executed, so that only intra-block communication is possible by means of the high-performance shared memory of the GPU. Starting from CUDA 5.0 and the introduction of DP, kernels can be called by GPU threads,¹ thus allowing recursion.

cuProCell implementation is based on the assumption that each cell division event generates two new cells that will then lead to other proliferation events. This behaviour can be represented by an unbalanced binary tree, whose root node is one cell belonging to the initial population. Each cell division adds a new level to the tree, representing the new cells created by that event. Therefore, the initial population of cells, denoted by $X(0)$, maps to a set of $C = \sum_{i=1}^M \psi_i$ unbalanced binary trees (where $\psi_i, i = 1, \dots, M$, is the frequency of the i -th fluorescence level in the initial histogram $H(0)$). We denote $X(d)$ the set of cells obtained after d division events.

The goal of cuProCell is to leverage the GPU to compute the C trees, leading to the final population $X(\text{final})$, i.e., the population formed by the cells that reached the maximum simulation time t_{max} starting from the initial population. For each cell, cuProCell also determines the vector of fluorescence levels φ^H corresponding to the final histogram. In order to parallelize this task, a kernel of exactly C threads is launched (cuProCell automatically determines the optimal balance of threads-per-block and block-per-grids); one thread is assigned to each cell in $X(0)$, and the new cells in $X(1)$ are generated following the same rules described for the sequential implementation [8], according to the user-specified model and parameterization.

cuProCell exploits DP to simulate cell proliferation on subsets of $X(d)$ corresponding to thread blocks, thanks to the absence of any dependencies between cells belonging to the same population (i.e., all division events are independent). The threads that belong to the same block are synchronized by explicit barriers, and a kernel composed by two new thread blocks is executed directly from the parent block, handling the cells in $X(d+1)$ created by division events occurred in the previous simulation step. At the end of the simulation process a histogram $H(t_{\text{max}})$ is returned in the form of a set of ordered pairs $\{(\varphi_1, \psi_1), \dots, (\varphi_N, \psi_N)\}$, where N is the number of different fluorescence values computed during the simulation.

The user can choose between CPU-based simulation or to exploit cuProCell to reduce the overall running time; in the latter case, ProCell automatically, and transparently, offloads to the GPU both the simulations and PEs.

Although CUDA is limited to 24 recursive calls, this constraint is hardly hit in the experiments with the AML models, since the maximum number of division events per cell in the considered time window is way lower than the maximal reachable depth (e.g., the fluorescence level reaches the threshold of auto-fluorescent cells after approximately 8 divisions). It is

¹Technically, recursion depth is limited to 24 due to hardware limitations.

TABLE I
UPPER AND LOWER BOUNDS OF THE SEARCH SPACE USED FOR PE (TIME EXPRESSED IN HOURS)

Proliferating		Slowly proliferating		Fast proliferating		Proportion	
μ_p	σ_p	μ_s	σ_s	μ_f	σ_f	π_q	π_f
[30, 250]	[0.01, 40]	[63, 250]	[0.01, 40]	[30, 63]	[0.01, 30]	[0, 1]	[0, 1]

worth noting that, besides the number of division events, the actual tree depth is also limited by the value of $X(0)$, since cuProCell needs to allocate enough GPU memory in advance to store all future populations $X(d)$, because of the asynchronous nested kernel calls. If a simulation reaches the practical limits of the GPU, the proliferation of cells is scheduled as a new top-level kernel execution, which involves the generation of a new set of binary trees having such cells as roots. To further accelerate the computation, cuProCell exploiting DP makes use of the binary search algorithm to update the fluorescence histogram every time a cell, assigned to a running thread, reaches the t_{\max} value. Another interesting perspective is that an increment of t_{\max} does not necessarily affect the overall running time, since most of the divided cells will go beyond the auto-fluorescence threshold and will be discarded from the simulation stack (see [8] for details).

IV. RESULTS

In this section we first describe the results of cell proliferation analysis by using fluorescence histograms of AML cells collected from the SPL of non-obese diabetic/severe xenotransplanted immunocompromised mice, and then we discuss the computational advantages of cuProCell. All results that follow were obtained using ProCell v1.5.0, cuProCell v1.6.1, FST-PSO v1.6.0.

A. Results on Spleen Data

We applied ProCell, exploiting the acceleration provided by cuProCell, to perform the PE of models of AML proliferation in PDXs. All results presented in this paper are based on new unpublished data collected from SPL. Similarly to our previous investigation of AML proliferation in BM [8], we tested three hypothetical scenarios:

- a model characterized by two sub-populations (PQ): proliferating cells and quiescent cells. This model has three parameters to be estimated;
- a model characterized by two sub-populations (SF): slowly proliferating cells and fast proliferating cells. This model has five parameters to be estimated;
- a model characterized by three sub-populations (SFQ): slowly proliferating cells, fast proliferating cells, and quiescent cells. This model has six parameters to be estimated.

The initial fluorescence histogram used for all simulations is shown in Fig. 2.

Given the nature of the models' parameters (e.g., the ratio between the populations, the standard deviations), their estimation was done by considering a linear scale for the putative values, using the search space shown in Table I. Differently from the previous works, we reduced the upper bound of slowly proliferating cells to 250 hours, with respect to the previous value

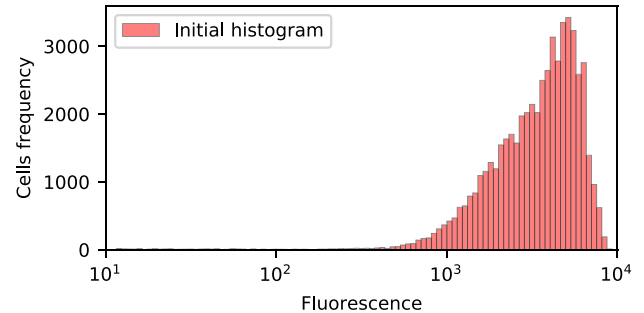


Fig. 2. Fluorescence histogram of GFPpos cells at $t = 0$.

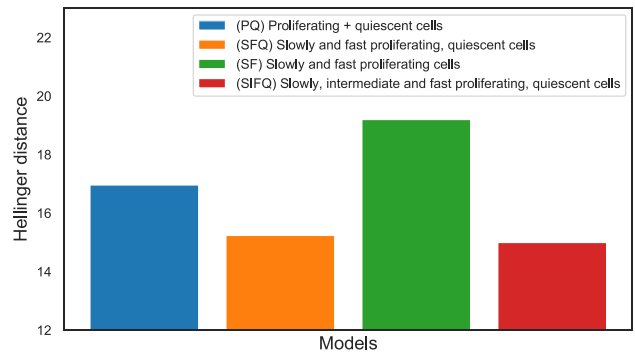


Fig. 3. Hellinger distances of the best solutions found by ProCell for each putative model.

of 504 hours for cell division. According to our preliminary tests, all sub-populations whose division rate is greater than 250 hours lead to highly skewed fluorescence distributions, characterized by a poor fitting to the target data. For this reason, we simplified the PE task by narrowing down the boundaries of slowly proliferating cells.

According to our results, the best fitting model is SFQ, i.e., the scenario characterized by three sub-populations: slowly proliferating cells, fast proliferating cells, and quiescent cells. In particular, by comparing the fitness values of the best solutions found for each model (Fig. 3), it can be seen that SFQ provided the lowest Hellinger distance.

SFQ has more parameters than PQ and SF, which could explain the better fitting with respect to the target data. In order to investigate this possibility, we tested a fourth model (named SIFQ) which extends the SFQ model by adding a third proliferating sub-population. Fig. 4 shows the convergence plot of FST-PSO, i.e., the average Hellinger distance of the best individuals of the swarm throughout the iterations. As a matter of fact, it is easier for FST-PSO to identify solutions characterized by a low Hellinger distance in the case of the SIFQ model, probably due to the additional free parameters.

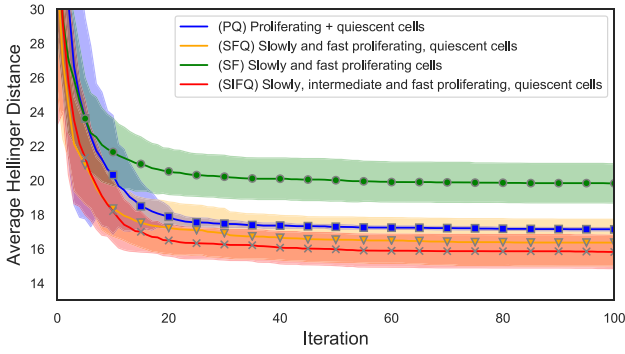


Fig. 4. Convergence of FST-PSO: the solid lines represent the ABF of best particles in the swarm evaluated over 20 runs of FST-PSO. The filled areas represent the standard deviation. Models SFQ and SIFQ converge, on average, to optimal solutions characterized by similar fitness value.

TABLE II

BEST PARAMETERIZATIONS FOUND BY PROCELL FOR ALL MODELS OF AML PROLIFERATION IN SPL. TIME EXPRESSED IN HOURS

	$\mu_p(\sigma_p)$	$\mu_s(\sigma_s)$	$\mu_f(\sigma_f)$	$\pi_q : \pi_p : \pi_s : \pi_f$
PQ Model	54.0 (39.8)	-	-	0.41 : 0.59 : - : -
SFQ Model	-	86.8 (34.0)	47.7 (29.9)	0.26 : - : 0.57 : 0.16
SF Model	-	173.13 (37.5)	54.1 (30.0)	- : - : 0.49 : 0.51

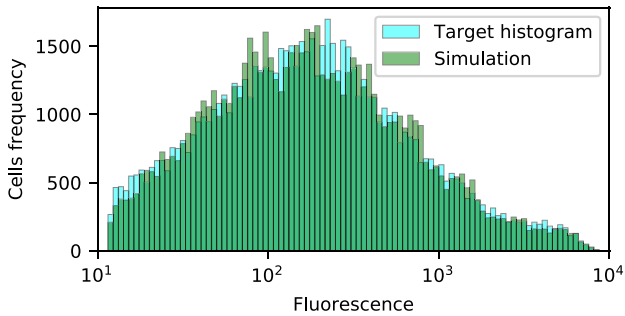


Fig. 5. Comparison between the target histogram (cyan bars) and the simulated histogram (green bars) obtained at $t_{\max} = 240$ h for the best fitting model (SFQ) with the best fitting parameterization found by FST-PSO.

However, the best fitting solution found for the SIFQ model has a Hellinger distance comparable to the best solution identified by SFQ (Fig. 3). Hence, we can conclude that the two models are equivalent as fitting, but model SFQ represents a simpler explanation for the observed data.

An interesting difference with respect to our previous investigations on BM [8], [9] is that, in the case of the SPL, the PQ model fits better than SF, highlighting the fundamental role of quiescent cells for the correct reproduction of the target fluorescence histogram (Fig. 3). These results confirm that SFQ is the simplest explanation for the observed fluorescence patterns, and that the SPL accumulates a larger number of quiescent cells with respect to the BM (approximately 26%, see Table II).

Fig. 5 shows a simulation of the best solution found for the SFQ model, which provided an excellent fit with respect to the experimental fluorescence data. Notably, thanks to FST-PSO,

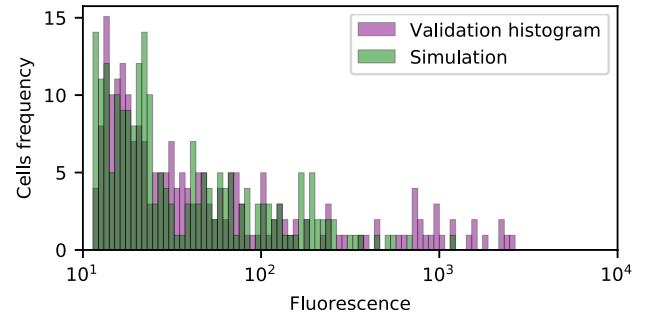


Fig. 6. Comparison between the validation histogram (purple bars) and the simulated histogram (green bars) obtained at $t_{\max} = 504$ h for the best fitting model (SFQ). Simulations run with the best fitting parameterizations found by FST-PSO.

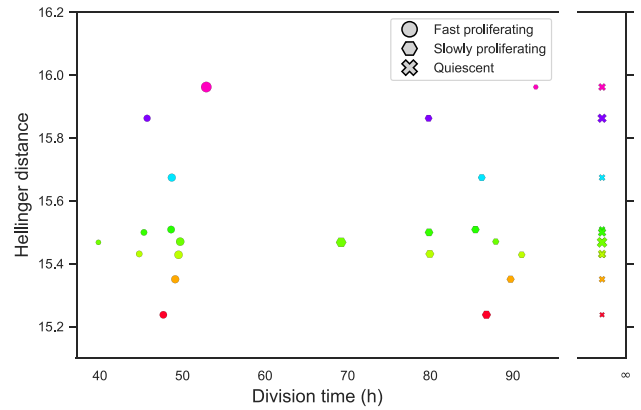


Fig. 7. Comparison of different parameterizations of the SFQ Model found by ProCell. For each parameterization, encoded with a different color corresponding to a different Hellinger distance value (y-axis), the mean time division of fast proliferating (circles) and slowly proliferating cells (hexagons), is reported. The size of the symbols used in each solution corresponds to the proportion of cells sub-population. Quiescent cells sub-population proportion is denoted with the cross symbol.

the optimization was performed without the need for any user-defined hyper-parameters.

In order to validate the SFQ model, we compared a simulation against the data collected from the SPL of the mouse after three weeks of chasing. Fig. 6 shows that the two distributions are comparable from a qualitative point of view; their matching overlap is also (quantitatively) confirmed by the Hellinger distance value, which is equal to 5.3. Clearly, the two distributions are not perfectly equal one another because of the stochastic fluctuations of the simulation that are present in this specific context (where the highest frequency value is 15), and the heterogeneous data used to calibrate and to validate the model (the AML cell populations are taken from 2 different mice after 10 and 21 days of chasing). This result further confirms the robustness and validity of the SFQ model.

As an additional analysis to the results obtained on the SFQ model, we compared the best parameterizations found by ProCell, concerning both the mean division time and cell sub-population proportions. As shown in Fig. 7, we encoded with a different color each parameterization of the SFQ model,

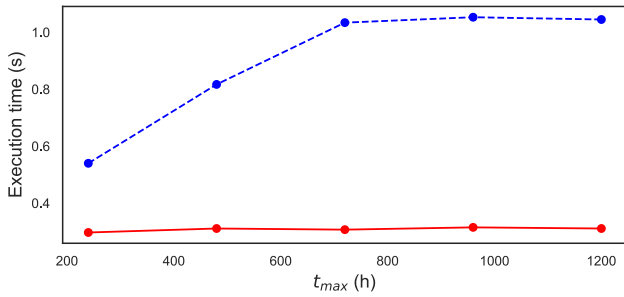


Fig. 8. Comparison between the execution time achieved by cuProCell with (red line) and without Dynamic Parallelism (blue dashed line), for different values of the maximum simulation time.

and we plot the mean division time of fast proliferating (circles) and slowly proliferating cells (hexagons). The size of the symbols used for each sub-population type corresponds to the proportion of cells (note that the proportion of quiescent cells is denoted with the cross symbol). Comparing different solutions, we observe that the mean division time of fast proliferating cells increases with the sub-population proportion; on the contrary, the mean division time of slowly proliferating cells increases as the sub-population proportion decreases. Moreover, the proportion of quiescent cells sub-population decreases as the mean division time of slowly proliferating cells increases.

B. Performance of cuProCell

To analyse the computational performances of cuProCell, as a first step we determined the impact of DP on the running time by performing GPU-powered multiple simulations—either with DP engaged or disengaged—and considering different values for t_{max} . In the cuProCell version not relying on DP, all blocks must complete their execution before a new kernel can be launched. Fig. 8 shows the result of this analysis, highlighting that DP provides a relevant reduction of the overall execution time. As a matter of fact, an implementation not relying on DP leads to a lower occupancy of resources as the number of completed blocks grows. In addition, a synchronization step between GPU and CPU is mandatory after every kernel execution, in order to avoid race conditions involved in $X(d+1)$ evaluations. Instead, thanks to DP, a block can execute a new kernel as soon as it has evaluated its assigned subset of $X(d)$, thus scheduling new thread blocks and avoiding the synchronization step due to the logical independence between cells population subsets.

In a second group of tests, we assessed the overall performance of cuProCell with respect to the original sequential implementation. To this aim, we performed multiple simulations, with increasing values of t_{max} , using a CPU Intel i5-8600 K, boost clock 4.3 GHz, and two different NVIDIA GPUs: a GeForce GTX Titan X and a Tesla V100. As reported in Fig. 9, the speed-up provided by cuProCell with respect to the Python implementation is up to $159\times$ and $237\times$ on the GeForce GTX Titan X and the Tesla V100, respectively. A part of the observed speed-up is due to the difference between compiled (C++/CUDA) and interpreted code (Python); moreover, the difference in running time between cuProCell and a CPU-bound

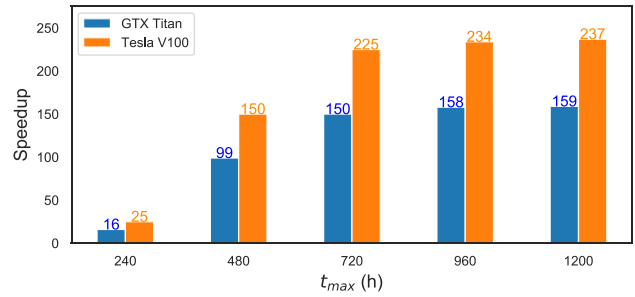


Fig. 9. Speed-up values achieved executing cuProCell on different NVIDIA video-cards with respect to the Python implementation, increasing the maximum simulation time.

version could also be mitigated by the use of multi-threading and the advanced vector extensions (AVX) provided by modern processors. Nevertheless, even by combining these approaches, the speed-up would not scale linearly because of the following issues: (i) the number of threads would largely outnumber the available cores; (ii) ProCell is a memory intensive algorithm, so that the running threads would immediately saturate the bandwidth and compete for shared resources (e.g., the cache), affecting the overall performance. It is worth noting that each cell can be simulated independently, so that a similar acceleration (excluding the overheads due to network communication) could be obtained by distributing the calculations on a multi-CPU cluster, exploiting the MPI infrastructure to collect data and build the simulated histograms. However, the downsides of this approach are two-fold: (i) it would be more expensive than a single consumer GPU; (ii) it would be energetically inefficient, since the peak power consumption of a GPU is approximately 300 W, while the power required by hundreds of CPUs would be higher by (at least) one order of magnitude [21].

Interestingly, cuProCell's speed-up is affected by the maximum simulation time t_{max} , which determines the number of recursive calls performed during the simulation and the depth of the division trees. A larger number of divisions implies a larger number of running threads, leading to higher occupancy of the GPU, which ultimately results in improved performances with respect to the CPU counterpart. The better performance of the V100 with respect to the Titan X can be explained by three factors: the higher number of streaming multiprocessors (80 vs 30), allowing a higher level of parallelism; the wider memory bus (4096 bits vs 384 bits) and the larger bandwidth (900 GB/s vs 480 GB/s), allowing a faster update of trees.

Thus, relevant speed-up allows ProCell to perform time consuming tasks more efficiently: the simulation, even in the case of large t_{max} , is almost immediate (approximately 0.3 seconds using Dynamic Parallelism, see Fig. 8) so that the running time for the whole PE is reduced from approximately 12 hours to around 90 minutes.

V. CONCLUSION

ProCell is an intuitive tool that allows to replicate *in silico* the cell population dynamics by taking into account the inherent stochasticity of cell division events, and the discrete nature of

sub-populations characterized by low cell numbers. In this work we presented cuProCell, a parallel implementation of ProCell that exploits the computational efficiency of GPUs. In particular, cuProCell makes use of Dynamic Parallelism to efficiently manage cell proliferation events. The results of our tests show that the GPU implementation of ProCell allows to drastically speed-up the simulations up to $237\times$ with respect to the Python version when running on a NVIDIA Tesla V100.

cuProCell allowed to efficiently obtain new insights on the intricate cellular organization of AML, highlighting the existence of slowly proliferating and long-term quiescent leukemic cells *in vivo*. In particular, the results presented in this work demonstrated that the analyzed SPL samples of AML xenografts are composed of populations with highly heterogeneous proliferative potential. A similar observation was reported in the case of BM samples of AML [8].

The comparison of the three models of cell proliferation, together with the best parameterization found by FST-PSO, highlighted that in the SFQ model—which allowed to obtain the best fitting with the experimental data—26% of leukemic blasts occurring in the initial population of cells does not divide during the chasing period of 10 days. Moreover, considering the proliferating cells identified in the sub-populations, 57% contribute in fueling tumor growth with lower cell turnover. It is clear that the complexity underlined by the SFQ model could have important biological and clinical consequences, as most therapies target almost exclusively fast proliferating cells.

We highlight that, in principle, ProCell could be applied to the analysis of cell proliferation in “pulse and chase” experiments exploiting fluorochromes or fusion proteins different from the ones described in this work, as long as the resulting fluorescence signal is distributed (almost) equally between the two daughter cells at each division. How the choice of different fusion proteins or fluorochrome and their respective half-lives affect the outcomes of the PE in ProCell needs to be assessed by means of *ad hoc* experiments. One example is PKH, a fluorescent labelling dye that binds to cell membranes and segregates in daughter cells after each cell division. The intensity of PKH staining correlates inversely with the number of previous cell divisions [22], so that ProCell can be exploited also with data produced with this technique. It is worth noting that this kind of labeling is not genetically introduced, so that it could be directly applied to patient samples *ex vivo*. In general, ProCell can be leveraged regardless the cell type and the fluorescence signal acquisition method (e.g., it could work, in principle, with confocal microscopy) provided that the latter generates a quantitative histogram.

As a future extension of this work, we plan to improve cuProCell to support multi-GPU systems, so that simulations of very large populations can be efficiently executed. ProCell will automatically identify the optimal partition of the initial population into multiple sets, handled by means of the Message Passing Interface [23], in order to offload the calculations onto multiple GPU-equipped computing nodes. Finally, we plan to investigate the effect of dilation functions [24] and Fourier surrogate modeling [25]: the former, applied to some components of the search space, could be effective for the simplification

of the PE problem and the general improvement of the final parameterizations; the latter would mitigate the fluctuations in the fitness evaluations due to ProCell’s stochastic simulation, thus preventing premature convergence to sub-optimal solutions.

VI. AVAILABILITY

ProCell can be easily installed using the Python Package Installer: `pip install procell`. ProCell’s source code can be downloaded from GITHUB at <https://github.com/aresio/procell>; the command `git clone --recurse-submodules` allows to automatically download the cuProCell submodule. Please refer to this page for any information about the execution of ProCell’s GUI. cuProCell source code and pre-compiled binaries are available under the GPL 2.0 license on GITHUB at <https://github.com/ericniso/cuda-pro-cell>.

VI. ETHICS STATEMENT

In vivo studies were performed after approval from the fully authorized animal facility of the Department of Experimental Oncology, notification of the experiments to the Ministry of Health (as required by the Italian Law) (IACUCs No 18/2013) and in accordance to EU directive 2010/63. Human samples were collected from patients whose informed consent was obtained in writing, according to the policies of the Ethics Committee of the European Institute of Oncology and regulations of Italian Ministry of Health. The studies were conducted in full compliance with the Declaration of Helsinki.

ACKNOWLEDGMENT

The authors would like to thank Luca Zanini for the development of ProCell’s user experience.

REFERENCES

- [1] D. A. Charlebois and G. Balázsi, “Modeling cell population dynamics,” *Silico Biol.*, vol. 13, no. 1-2, pp. 21–39, 2019.
- [2] E. A. Sarapata and L. de Pillis, “A comparison and catalog of intrinsic tumor growth models,” *Bull. Math. Biol.*, vol. 76, no. 8, pp. 2010–2024, 2014.
- [3] T. Luzyanina, J. Cupovic, B. Ludewig, and G. Bocharov, “Mathematical models for CFSE labelled lymphocyte dynamics: Asymmetry and time-lag in division,” *J. Math. Biol.*, vol. 69, no. 6–7, pp. 1547–1583, 2014.
- [4] S. Hross and J. Hasenauer, “Analysis of CFSE time-series data using division-, age- and label-structured population models,” *Bioinf.*, vol. 32, no. 15, pp. 2321–2329, 2016.
- [5] M. d’Inverno and R. Saunders, “Agent-based modelling of stem cell self-organisation in a niche,” in *Eng. Self-Organising Syst. ESOA 2004*, ser. LNCS, S. Brueckner, G. Di Marzo Serugendo, A. Karageorgos, and R. Nagpal, Eds., vol. 3464. Berlin, Germany: Springer, 2005, pp. 52–68.
- [6] A. Ibrahim-Hashim *et al.*, “Defining cancer subpopulations by adaptive strategies rather than molecular properties provides novel insights into intratumoral evolution,” *Cancer Res.*, vol. 77, no. 9, pp. 2242–2254, 2017.
- [7] T. W. Fawcett and A. D. Higginson, “Heavy use of equations impedes communication among biologists,” *PNAS*, vol. 109, no. 29, pp. 11735–11739, 2012.
- [8] M. S. Nobile *et al.*, “Modeling cell proliferation in human acute myeloid leukemia xenografts,” *Bioinf.*, vol. 35, no. 18, pp. 3378–3386, 2019.
- [9] M. S. Nobile *et al.*, “ProCell: Investigating cell proliferation with swarm intelligence,” in *Proc. 16th IEEE Int. Conf. Comput. Intell. Bioinf. Comput. Biol.*, 2019, pp. 1–8.
- [10] M. S. Nobile, P. Cazzaniga, D. Besozzi, R. Colombo, G. Mauri, and G. Pasi, “Fuzzy Self-Tuning PSO: A settings-free algorithm for global optimization,” *Swarm Evol. Comput.*, vol. 39, pp. 70–85, 2018.

- [11] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intell.*, vol. 1, no. 1, pp. 33–57, 2007.
- [12] H. Döhner, D. J. Weisdorf, and C. D. Bloomfield, "Acute myeloid leukemia," *N. England J. Med.*, vol. 373, no. 12, pp. 1136–1152, 2015.
- [13] A. Viale *et al.*, "Cell-cycle restriction limits DNA damage and maintains self-renewal of leukaemia stem cells," *Nature*, vol. 457, no. 7225, pp. 51–56, 2009.
- [14] F. Ishikawa *et al.*, "Chemotherapy-resistant human AML stem cells home to and engraft within the bone-marrow endosteal region," *Nat. Biotechnol.*, vol. 25, no. 11, pp. 1315–1321, 2007.
- [15] B. Falkowska-Hansen *et al.*, "An inducible Tet-Off-H2B-GFP lentiviral reporter vector for detection and *in vivo* isolation of label-retaining cells," *Exp. Cell Res.*, vol. 316, no. 11, pp. 1885–1895, 2010.
- [16] P. Sanchez *et al.*, "A robust xenotransplantation model for acute myeloid leukemia," *Leukemia*, vol. 23, no. 11, pp. 2109–2117, 2009.
- [17] A. Tangherloni *et al.*, "Biochemical parameter estimation vs. benchmark functions: A comparative study of optimization performance and representation design," *Appl. Soft Comput.*, vol. 81, 2019, Art. no. 105494.
- [18] M. S. Nobile *et al.*, "Computational intelligence for parameter estimation of biochemical systems," in *Proc. IEEE World Congr. Comput. Intell.*, 2018, pp. 1–8.
- [19] M. S. Nikulin, "Hellinger distance," *Encyclopedia Math.*, vol. 151, 2001.
- [20] N. Wilt, *The CUDA Handbook: A Comprehensive Guide to GPU Programming*. Upper Saddle River, NJ, USA: Pearson Education, 2013.
- [21] M. S. Nobile, P. Cazzaniga, A. Tangherloni, and D. Besozzi, "Graphics processing units in bioinformatics, computational biology and systems biology," *Briefings Bioinf.*, vol. 18, no. 5, pp. 870–885, 2017.
- [22] A. Cicalese *et al.*, "The tumor suppressor P53 regulates polarity of self-renewing divisions in mammary stem cells," *Cell*, vol. 138, no. 6, pp. 1083–1095, 2009.
- [23] D. W. Walker and J. J. Dongarra, "MPI: A standard message passing interface," *Supercomputer*, vol. 12, pp. 56–68, 1996.
- [24] M. S. Nobile, P. Cazzaniga, and D. A. Ashlock, "Dilation functions in global optimization," in *Proc. IEEE Congr. Eol. Computat.*, 2019, pp. 2300–2307.
- [25] L. Manzoni *et al.*, "Surfing on fitness landscapes: A boost on optimization by Fourier surrogate modeling," *Entropy*, vol. 22, no. 3, p. 285, 2020.