PhysiCell: an Open Source Physics-Based Cell Simulator for 3-D Multicellular Systems

Ahmadreza Ghaffarizadeh[1], Samuel H. Friedman[1], Shannon M Mumenthaler[1], Paul Macklin[1,*]

[1]Lawrence J. Ellison Institute for Transformative Medicine, University of Southern California, Los Angeles, CA 90033, USA

[*]Paul.Macklin@MathCancer.org

Abstract

Many multicellular systems problems can only be understood by studying how cells move, grow, divide, interact, and die. Tissue-scale dynamics emerge from systems of many interacting cells as they respond to and influence their microenvironment. The ideal "virtual laboratory" for such multicellular systems simulates both the biochemical microenvironment (the "stage") and many mechanically and biochemically interacting cells (the "players" upon the stage).

PhysiCell—physics-based multicellular simulator—is an open source agent-based simulator that provides both the stage and the players for studying many interacting cells in dynamic tissue microenvironments. It builds upon a specialized multi-substrate biotransport solver so that modelers can link cell phenotype to multiple diffusing substrates and signaling factors. It includes biologically-driven sub-models for cell cycling, apoptosis, necrosis, solid and fluid volume changes, mechanics, and motility "out of the box," allowing modelers to concentrate on microenvironment-driven hypotheses. The C++ code has minimal dependencies, making it simple to maintain across platforms. PhysiCell has been parallelized with OpenMP, and its performance scales linearly with the number of cells. Simulations up to 10^5-10^6 cells are feasible on quad-core desktop workstations; larger simulations are attainable on single HPC compute nodes.

We demonstrate PhysiCell by simulating impact of necrotic core biomechanics, 3-D geometry, and stochasticity on the dynamics of hanging drop tumor spheroids and ductal carcinoma in situ (DCIS) of the breast. PhysiCell is a powerful multicellular systems simulator that will be continually improved with new sub-models, capabilities, and performance improvements. It also represents a significant independent code base for replicating results from other simulation platforms. The PhysiCell source code, examples, documentation, and support are available under the BSD license at `http://PhysiCell.MathCancer.org` and `http://PhysiCell.sf.net`.

Author Summary

This paper introduces PhysiCell: an open source, agent-based model for 3-D multicellular simulations. It includes a standard library of sub-models for cell fluid and solid volume changes, cell cycle progression, apoptosis, necrosis, and mechanics. The code is directly coupled to a specialized biotransport solver to simulate many diffusing substrates and cell signals. Each cell can release diffusing signals, and dynamically update its phenotype according to its microenvironmental conditions. Users can customize or replace the included sub-models.

PhysiCell was designed to work on a variety of platforms (Linux, OSX, and Windows) with a minimum number of software dependencies. Its computational cost scales linearly in the number of cells. It is feasible to simulate 500,000 cells on current quad-core desktop workstations, and millions of cells on single HPC compute nodes. In this paper, we demonstrate PhysiCell to test the impact of necrotic core biomechanics, 3-D geometry, and stochasticity on hanging drop tumor spheroids (HDS) and ductal carcinoma in situ (DCIS) of the breast.

We developed PhysiCell to help the scientific community tackle multicellular systems biology problems involving many interacting cells in multi-substrate microenvironments. PhysiCell also represents an important independent, cross-platform code base for replicating simulation results from other simulation platforms.

Introduction

Many significant multicellular systems processes—such as tissue engineering, evolution in bacterial colonies, and tumor metastasis—can only be understood by studying how individual cells grow, divide, die, and interact [1–5]. Tissue-scale dynamics emerge as cells are influenced by biochemical and biophysical signals in the microenvironment, even as the cells continually remodel the microenvironment. Thus, the ideal "virtual laboratory" for multicellular systems biology must simultaneously simulate (1) the dynamics of many mechanically and biochemically interacting cells, and (2) tissue microenvironments with multiple diffusing chemical signals (e.g., oxygen, drugs, and signaling factors) [5]. We recently published and open sourced the first part of such a platform: BioFVM, a biotransport solver that can efficiently simulate secretion, diffusion, uptake, and decay of multiple substrates in large 3-D microenvironments, even on desktop workstations [6]. We now introduce and release as open source PhysiCell: a mechanistic off-lattice agent-based model built on top of BioFVM to simulate the tissue-scale behaviors that emerge from basic biological and biophysical cell processes.

Prior work and goals for PhysiCell

Several major computational frameworks are available for studying 3-D multicellular systems. CompuCell3D [7] and Morpheus [8] use cellular Potts methods to simulate cells and their morphologies. They are very user-friendly packages with graphical model editors, integrated ODE and PDE solvers, and support for molecular-scale sub-models, but they currently cannot scale to large numbers (10^5 or more) of cells. TiSim (part of the CellSys package [9]) can simulate many more cells by using a cell-centered, off-lattice approach. However, it is currently closed source, and its executables are restricted to a limited set of simulation types. Chaste [10] is a powerful, well-developed framework for multicellular modeling with integrated PDE and ODE solvers, and both cell- and vertex-based simulations of 10^5 or more cells. However, its complex codebase has many dependencies that can impede participation by new developers; it is only cross-platform compatible by virtual machines. Biocellion [11] can simulate billions of cells on cluster computers, but it is closed source, and its restrictive user license has hindered adoption. Most of these platforms offer a general-purpose pre-compiled "client" that can load models and settings from an XML file; this helps overcome difficulties stemming from complex dependencies. See the supplementary materials for a detailed software comparison.

These platforms typically require users to write their own code for all cell activities, by scripting basic built-in functions. (e.g., build a cell cycle model from API functions to overwrite cell volume and duplicate agents when appropriate.) As configured "out of the box," none have built-in models for cell cycling, apoptosis, and necrosis, even though these fundamental behaviors are needed in many multicellular simulations. Only CompuCell3D and Morpheus have built-in volume regulation features. None of these packages include transport solvers that are tailored to biology and will efficiently scale to 3-D simulations with many diffusable factors–a key requirement in reconciling secretomics with single-cell and multicellular systems biology, particularly as we work to understand cell-cell communication involving many cell-secreted factors.

PhysiCell aims to balance computational speed, built-in standard functionality, flexibility, and codebase simplicity. It includes a built-in library of standardized cell cycle and cell death models co-developed with biologists and modelers here and in the MultiCellDS standardization process [12, 13], force-based cell-cell interaction mechanics, and volume regulation. Users can replace any of these built-in models with their own, and they can dynamically assign custom functions to any agent at any time. Through BioFVM, PhysiCell can couple cell phenotype to many diffusable substrates. It is the only simulation package to explicitly model the cell's fluid content. It can simulate systems of $10^5 - 10^6$ cells on desktop workstations, and 10^6 or more cells on single HPC compute nodes. All this functionality and performance is achieved with only two external dependencies, and a fully cross-platform C++ codebase that we have compiled and tested on Linux, OSX, and Windows.

PhysiCell will help its users to test the behaviors that emerge from basic biological and physical processes, and to evaluate model predictions against multicellular data [14]. It also serves as a powerful, independent codebase to cross-validate model predictions in Chaste, Biocellion, TiSim, and other platforms.

Design and Implementation

PhysiCell is designed to study the dynamics and interactions of thousands or millions of cells in 3-D microenvironments, with microenvironment-dependent phenotypes. It uses a lattice-free, physics-based approach to reduce grid-based artifacts. It provides optimized, biologically realistic functions for key cell behaviors, including: cell cycling (multiple models for *in vitro* and *in vivo*-focused simulations) cell death (apoptosis and necrosis), volume regulation (fluid and solid biomass; nuclear and cytoplasmic subvolumes), and cell-cell mechanical interactions. This allows users to focus on modeling microenvironment-dependent triggers of standard cell processes, rather than coding these basic processes. However, to maintain flexibility, PhysiCell is written in a modular manner so that users can extend, rewrite, or replace its functions. Users can also create custom rules, and assign them to individual agents. It is fully coupled to a fast multi-substrate diffusion code (BioFVM) that solves for vectors of diffusing substrates, so that users can tie cell phenotype to many diffusing signals.

PhysiCell was built by extending the `Basic_Agent` class in BioFVM [6] (a static, non-moving object that can secrete and uptake substrates) into a fully dynamic `Cell` class with changing cell volume, cycle progression, death processes, and mechanics. This allows the cells to directly and efficiently interface with the multi-substrate microenvironment. PhysiCell is written in cross-platform compatible C++ and is self-contained (with minimal dependencies). It can be compiled in any C++11 compiler with OpenMP support. This simplifies installation and improves the reproducibility of the experiments. We have tested PhysiCell on Windows through MinGW-w64, and on OSX and Linux via g++. PhysiCell's only external dependencies are pugixml [15] (for XML parsing) and BioFVM [6] for 3-D multi-substrate diffusion. For the user's convencience, compatible versions of pugixml and BioFVM are included in every download.

The code has been parallelized in OpenMP to make use of multi-core desktop workstations and HPC compute nodes. In testing, its performance scales linearly in the number of cells. Simulations of up to 10^6 cells are feasible on desktop workstations, and simulations beyond 10^6 cells are possible on typical HPC compute nodes.

Agent-based cell model

PhysiCell implements key cell-scale processes—cell cycling and death, volume changes, mechanics, and motility—and lets users link the parameters of these processes to microenvironmental conditions. PhysiCell then scales these basic cell-scale hypotheses to simulate thousands or millions of cells, from which tissue-scale behavior emerges. Here, we summarize the key functions. For each sub-model, see the supplementary materials for the full equations, further biological motivation, and reference parameter values.

Cell characteristics and state Cell agents have a variety of phenotypic properties, including position ($\mathbf{x}_i$), volume (and sub-volumes), cell cycle or death status, and mechanics (adhesive, deformation, and motility) parameters. See the supplementary materials for a list of all the cell agents' attributes and functions to access/update them. Below, we describe standardized, built-in models to update these properties. The models can be replaced by user-defined functions; the supplied models serve as biophysically reasonable default functions that capture the key aspects of these processes.

Cell volume Each cell tracks V (total volume), V_F (total fluid volume), V_S (total solid volume), V_NS (nuclear solid volume), V_CS (cytoplasmic solid volume), V_N (total nuclear volume), and V_C (total cytoplasmic volume). Key parameters include nuclear solid, cytoplasmic solid, and fluid rate change parameters (r_N, r_C, and r_F), the cell's "target" fluid fraction f_F, target solid volume V^*_NS, and target cytoplasmic to nuclear volume ratio f_CN. These parameters are updated as the cell progresses throuh its current cycle or death process. (See Cell cycling and Cell death.)

Cell mechanics and motion We model cell mechanics and motion as in our prior work [16]: we update each cell's position $\mathbf{x}_i$ by calculating its current velocity $\mathbf{v}_i$ based upon the balance of forces acting upon it. The main forces include cell motility, drag-like forces, and cell-cell and cell-matrix interaction forces: adhesion and "repulsion" (resistance to deformation and/or volume exclusion [17]). As in prior cell-centered models [16, 18, 19], we apply an inertialess assumption to explicity solve for each cell's velocity. We model adhesion and repulsion with interaction potentials that depend upon each cell's size, maximum adhesion distance, adhesion and repulsion parameters, and distance to other cells [16].

Cell cycling PhysiCell includes a cell cycle modeling framework, where each cell cycle model is a collection of phases $\{X_k\}$, transition rates $\{r_{ij}\}$ between the phases, and a cell division phase transition. As in [16], we use the phase transition rates to calculate the phase change probabilities in any time interval $[t, t + \Delta t]$. Some cell cycle models can also replace the stochastic phase transitions with deterministic transitions.

Each cell agent tracks its current cell cycle phase $\mathcal{S}_k$ and its total time spent in that phase (t_k). Users can change the transition rates at any time, in part based upon microenvironmental conditions (e.g., based upon oxygenation or cell contact).

As a concrete example, consider the "Ki67 Advanced" model from our prior work calibrating oxygen-dependent growth to Ki67 data in ductal carcinoma in situ (DCIS) [16, 20, 21]. The phases are K_1 (Ki67+ cycling cells, prior to cell division), K_2 (Ki67+ cycling cells, after cell division), and Q (Ki67- quiescent cells). K_1 and K_2 have stochastic durations (with means T_1 and T_2). We model the transition rate from Q to K_1 as

$$r_{Q1} = \frac{1}{\overline{T}_\mathrm{Q}} \max\left\{ \left(\frac{\mathrm{pO}_2 - \mathrm{pO}_{2,\mathrm{hypoxia}}}{\overline{\mathrm{pO}_2} - \mathrm{pO}_{2,\mathrm{hypoxia}}} \right), 0 \right\}, \tag{1}$$

where cells spend a mean time of $\overline{T}_\mathrm{Q}$ in the Q phase when $\mathrm{pO}_2 = \overline{\mathrm{pO}_2}$. Cells double V^*_NS when transitioning from Q to K_1 (to double their nuclear content), and they halve V^*_NS (and all the sub-volumes) when dividing into two daughter cells at the $K_1 \longrightarrow K_2$ transition. The full set of supported cell cycle models—along with reference parameter values—is given in the supplementary materials.

Cell death PhysiCell currently includes models for two types of cell death: apoptosis (programmed cell death) and necrosis (unprogrammed cell death) [22]. At any time, each agent (with index i) has two death rates ($r_{\mathrm{A},i}$ for apoptosis, and $r_{\mathrm{N},i}$ for necrosis), which can be continually updated. For any death rate r_i and any time interval $[t, t + \Delta t]$, the cell has a probability of entering the corresponding death state D:

$$\mathrm{Prob}\Big(\mathcal{S}_i(t + \Delta t) = D \Big) = 1 - \exp\big(-r_i \Delta t \big) \approx r_i \Delta t. \tag{2}$$

Apoptosis: Upon entering the apoptotic state, we set $f_\mathrm{CN} = 0$ (to simulate shrinking and blebbing of the cytoplasm), $V^*_\mathrm{NS} = 0$ (to simulate degradation of the nucleus), and $f_\mathrm{F} = 0$ (to simulate the active elimination of water from the cell). The rates r_N, r_F, and r_C are set to match time scales of cell volume loss in apoptotic cells. The cell is removed once its volume drops below a user-set threshold, or after mean duration of T_A.

Necrosis: When a cell becomes necrotic, we set $f_\mathrm{CN} = V^*_\mathrm{NS} = 0$ to model cytoplasmic and nuclear degradation. Early necrotic cells undergo oncosis (cell death-related swelling); we model this by setting $f_\mathrm{F} = 1$. (Note that some regard oncosis as the actual death process, and necrosis as post-mortem cell degradation [23, 24].) Once the cell volume passes a critical threshold, it lyses, and we set $f_\mathrm{F} = 0$. The rate parameters r_F, r_N, and r_C are set to match expected time scales throughout necrosis [22]. PhysiCell includes codes to trigger necrosis deterministically or stochastically:

Deterministic Necrosis: This implements a common model of necrosis (see the review [2]), where cells instantly become necrotic whenever oxygenation pO_2 drops below a threshold value $\mathrm{pO}_{2,\mathrm{threshold}}$, as in our earlier work [16]. This is equivalent to the letting $r_\mathrm{N} \to \infty$.

Stochastic Necrosis: This model updates our prior work [16], based upon *in vitro* observations that cells can survive low oxygen conditions for hours or days. Here,

$$r_\mathrm{N}\left(\mathrm{pO}_2\right) = \begin{cases} 0 & \text{if } \mathrm{pO}_{2,\mathrm{threshold}} < \mathrm{pO}_2 \\ r_{\mathrm{N,max}} \left(\frac{\mathrm{pO}_{2,\mathrm{threshold}} - \mathrm{pO}_2}{\mathrm{pO}_{2,\mathrm{threshold}} - \mathrm{pO}_{2,\mathrm{crit}}} \right) & \text{if } \mathrm{pO}_{2,\mathrm{crit}} < \mathrm{pO}_2 \le \mathrm{pO}_{2,\mathrm{threshold}} \\ r_{\mathrm{N,max}} & \text{if } \mathrm{pO}_2 \le \mathrm{pO}_{2,\mathrm{crit}}. \end{cases} \tag{3}$$

That is, necrotic death begins when $\mathrm{pO}_2 < \mathrm{pO}_{2,\mathrm{threshold}}$, and the death rate ramps linearly until saturating at a maximum rate $r_{\mathrm{N,max}}$ for $\mathrm{pO}_2 < \mathrm{pO}_{2,\mathrm{crit}}$. Equivalently, cells survive on average $1/r_{\mathrm{N,max}}$ time in very low oxygen conditions [16].

Numerical implementation

Overall program flow Fig. 1 outlines PhysiCell's overall program flow. After initializing the microenvironment (through BioFVM) and cells, PhysiCell repeats the main program loop, which:

1. saves the simulation state (as needed),
2. runs BioFVM to update the microenvironment,
3. updates the cell phenotype parameters (by sampling the microenvironment),
4. advances the cell cycle/death state and runs the cell volume model,
5. calculates the force-based cell velocities, and runs any custom functions.
6. uses the velocities to update the cell positions, and
7. updates the current simulation time.

Several steps (marked with a red $\|$ symbol) are parallelized with OpenMP. See the supplementary material for further numerical details.

Estimated computational cost scaling We now assess the computational effort needed for each iteration in the main program loop. (See Fig. 1.) Step 2 (save simulation data), Step 4 (update phenotypes), and Step 6 (update positions) clearly entail a constant amount of work for each cell. Thus, summing these steps over all cells $n(t)$ requires $\mathcal{O}(n)$ work. By prior analysis, BioFVM (Step 3) also scales linearly in $n(t)$ [6].

Step 5 (update velocities) is the most computationally expensive step. In straightforward implementations, each cell tests for mechanical interaction with $n-1$ other cells, giving an $\mathcal{O}(n^2)$ total computational cost at each time step. However, the IDS (see Key code optimizations) restricts interaction testing to a smaller set $\mathcal{N}(i)$. In the supplementary material, we show that each $\mathcal{N}(i)$ has at most $N_{\max}$ cells. Thus, Step 5 has a fixed maximum cost for each cell, and the cost of the loop scales linearly in n.

Key code optimizations To prevent computational costs from scaling quadratically in the number of cells, we designed a cell-cell interaction data structure (IDS) that efficiently estimates a set $\mathcal{N}$ of possible neighbor cells for each cell agent. See the supplementary material for further detail.

PhysiCell uses OpenMP to parallelize most loops over the list of cells. This includes sampling the microenvironment, updating cell phenotype parameters, advancing the cell cycle or death model, advancing the volume model, running any custom function, and calculating the cell velocity. We do not parallelize loops that change the IDS: cell division, cell removal, and updating the cell position.

We defined three separate computational step sizes to take advantage of the multiple time scales of the multicellular system: Δt_{diff} for biotransport processes, Δt_{mech} for cell mechanics and motion, and Δt_{cells} for cell cycle, death, and volume processes. We update each process according to its own time step, rather than at each simulation step. Fig. 2 illustrates how the multiple times steps reduce the computational cost. See the supplementary materials for further detail and the default step sizes for cancer biology.

Convergence and validation testing

We performed convergence testing on all the major components of PhysiCell. BioFVM was previously tested as first-order accurate in Δt, second-order accurate in Δx, and sufficiently accurate at $\Delta x = 20\,\mu$m and $\Delta t_{\text{diff}} = 0.01$ to 0.05 min for tumor growth problems [6]. We performed two tests for cell-cell mechanics and motion: First, we placed two cells in partial overlap, simulated their relaxation to equilibrium, and measured the cell spacing at several times. Second, we created a compressed cluster of 50,000 cells, simulated its mechanical relation to equilibrium, and measured its diameter at several times. Both tests converged to first-order accuracy in Δt at all measured times, showing that PhysiCell converged in both short-time mechanical dynamics and in long-time behavior. $\Delta t_{\mathrm{mech}} \sim 0.1$ min gives sufficent accuracy for typical cancer problems.

We simulated the volume model for a single proliferating, apoptotic, and necrotic cell, and measured the sub-volumes at multiple times. It converged with first-order accuracy in Δt at all tested times, and $\Delta t_{\mathrm{cell}} = 6$ min gave sufficient accuracy. We tested the stochastic transition codes by simulating the Ki67-advanced cell cycle model and the apoptosis death model (with stochastic duration), and measuring the sub-population counts and population fractions over time for several values of Δt_{cell}. For each Δt, we performed 100 simulations and compared the mean solution behavior against known coarse-grained ODE model behavior. $\Delta t_{\mathrm{cell}} = 6$ min and 60 min both gave an excellent match between the PhysiCell behavior and theory for all the compared curves. See the supplementary materials for full testing results.

Performance testing (summary)

By our testing, recent quad-core desktop workstations (with hyperthreading, for 8 total execution threads) can simulate 10-30 days in systems of up to 10^5 to 10^6 cells in 3 days or less (wall time). Single HPC compute nodes (typically two 6-8 core processors, with hyperthreading and 24-32 execution threads) can simulate larger systems up to $\sim$2 million cells in about 2 days. Future releases of PhysiCell will address current performance bottlenecks; see Availability and Future Directions. The Results will give a demonstration of $\mathcal{O}(n)$ computational cost scaling.

Results

We demonstrated PhysiCell's potential to simulate large multicellular systems—and its ability to test the emergent tissue-scale effects of cell-scale hypotheses—on two examples arising from cancer biology. For each example, we compared the impact of the deterministic and stochastic necrosis models. (See Cell death above.) We used the Ki67-advanced cell cycle model with deterministic K_1, K_2, and A phase durations. (See Cell cycling.) We provide the parameter values in the supplementary materials, and the full source code and postprocessing routines for both examples in every PhysiCell download. Reference simulation outputs are available at `http://PhysiCell.MathCancer.org`.

Test platforms

We tested on (1) a desktop workstation (quad-core Intel i7-4790, 3.60 GHz, 8 execution threads, 16 GB memory) with mingw-w64 (g++ ver. 4.9.1) on 64-bit Windows 7, and (2) a single HPC compute node (dual 6-core Intel Xeon X5690, 3.47 GHz, 24 execution threads, 48 GB memory) with g++ (ver. 4.8.4) on Ubuntu 14.04. The CPU architecture was newer on the desktop (2014 Haswell) than on the HPC node (2011 Westmere).

Hanging drop tumor spheroids

Hanging drop spheroids (HDS)—a 3-D cell culture model where a small cluster or aggregate of tumor cells is suspended in a drop of growth medium by surface tension—are increasingly used to approximate 3-D *in vivo* growth conditions [25]. Unlike traditional 2-D monolayer experiments, HDSs allow scientists to investigate the impact of substrate gradients on tumor growth, particularly oxygen gradients. Their relatively simple geometry makes them ideal for testing computational models.

We simulated HDS growth by placing an initial cluster of ~ 2300 cells in an 8 mm^3 fluid domain, with Dirichlet conditions $\mathrm{pO}_2 = 38$ mmHg (5% oxygen: physioxic conditions [26]) on the computational boundary. The simulation results are shown in Fig. 3 for deterministic necrosis (left column) and stochastic necrosis (right column), at 4, 8, and 16 days. In Fig. 4, we show the tumor diameter (left panel) and number of agents (right panel) versus time. Both simulations reached $\sim 10^6$ cells by 18 days. See the simulation videos Video S1 and Video S2.

Deterministic versus stochastic necrosis Both models yielded similar dynamics. Hypoxic gradients emerged quickly, limiting (pO_2-dependent) cell division to the outermost portions of the tumors. This, in turn, lead the tumor diameters to grow linearly (at similar rates); see Fig. 4. This matches our theoretical expectations for a spheroid of radius $R(t)$ whose growth is restricted to an outer layer of fixed thickness T:

$$\frac{d}{dt}V(t) = c \cdot \overbrace{4\pi R^2(t) \cdot T}^{\text{growing region}} \implies \frac{d}{dt}\left(\frac{4}{3}\pi R^3(t)\right) = c \cdot 4\pi R^2(t) \cdot T \tag{4}$$

$$\implies \frac{d}{dt}R(t) = cT = \text{ constant.} \tag{5}$$

In both models, the innermost portion of the necrotic core developed a network of fluid voids or cracks. This phenomenon emerges from competing biophysical effects of the multicellular system: necrotic cells lose volume, even as they continue to adhere, leading to the formation of cracks. Similar cracked necrotic core structures have been observed with *in vitro* hanging drop spheroids (e.g., [5, 27, 28]).

There were notable differences between the models. The deterministic model had a sharp perinecrotic boundary between the viable and necrotic tissues, whereas the stochastic model demonstrated a perinecrotic transition zone with substantial mixing of viable and necrotic cells. Because cells do not immediately necrose in the stochastic model, it retained a center of quiescent viable cells longer than the deterministic model. The growth curves for the deterministic and stochastic models appear to diverge after approximately 8 days, when the deterministic necrotic core is better defined with more cracks than the stochastic core. This may be due to differences in hypoxic gradients (the tumor with more void spaces will have shallower oxygen gradients, and hence more cell cycle entry), but further simulations would be required to rule out stochastic effects. Interestingly, the stochastic model's growth curve appears to run parallel to the deterministic curve for later times, once its necrotic core becomes better defined.

Performance scaling Throughout the simulations, the computational cost (the wall time required to simulate one hour) scaled approximately linearly with the number of agents present in the simuation, on both the desktop workstation and the HPC node; see Fig. 5. (See also Estimated computational cost scaling.) Increasing the number of execution threads improved performance, even when running on slower processor cores. See the right panel in Fig. 5, where moving from the newer 8-threaded machine to the older 24-threaded machine improved performance by a factor of 2 to 2.5.

The simulations reached $\sim 10^6$ cells on our HPC tests after 67 hours (deterministic, 17 simulated days) to 76 hours (stochastic, 18.2 simulated days) of wall time, including saving full simulation output data once per simulated hour. See Fig. 4. The desktop workstation simulated past 573,000 cells (about 14.6 days of simulated time) in approxiately 80 hours of wall time. The desktop tests did not run out of memory, and the simulations can be completed to the full 18 days and 10^6 cells if needed.

Ductal carcinoma in situ (DCIS)

DCIS is a pre-malignant breast condition where epithelial cells ("tumor cells") divide abnormally to fill the breast duct lumen. Oxygen can only reach the tumor cells by diffusion from outside the duct, leading to the emergence of hypoxia and an inner necrotic core. See [16, 20, 21] for further biological and clinical discussion. As in [16], we approximate a partly-filled breast duct as a 3-D "test tube" with a level set function representation. Cell adhere to cells and the duct wall; cells and the duct wall push against cells to resist deformation. Oxygen diffuses from the duct wall and is consumed by tumor cells. The rate of cycle entry increases linearly with pO_2 (see Cell cycling).

In Fig. 6, we show DCIS simulations in a 1 mm segment of breast duct (317.5 μm diameter), using deterministic necrosis (left side) and stochastic necrosis (right side), plotted at 10 and 30 days. See also Video S3 and Video S4.

Comparison of necrosis models; comparison with the spheroid example As in the HDS example, the deterministic model had a sharp, smooth perinecrotic boundary, whereas the stochastic model demonstrated a perinecrotic boundary region with mixed viable and necrotic cells. In the stochastic model, proliferation halted in the duct center, but necrosis appeared later. The perinecrotic mixing effect was most pronouced at the leading edge of the tumor, where tissue was transitioning from non-hypoxic/non-necrotic to necrotic. Areas with longer-term hypoxia had smoother necrotic boundaries. This effect did not emerge in the HDS example due to its symmetry.

Interestingly, the mechanical "cracks" seen in the tumor spheroids do not appear here, because the breast duct compresses the necrotic core to collapse any fluid-filled voids. This shows the importance of the 3-D geometry and the biophysical impact of the basement membrane, as well as the need to account for such effects when approximating *in vivo* conditions with bioengineered model systems.

Both models gave approximately the same growth rate of ~ 1 cm/year (Fig. 7, left). We cannot select one model over the other based solely upon continuum-scale, coarse-grained outputs. However, we could further assess the models by comparing their distinct differences in multicellular-scale patterning to DCIS pathology. This further highlights the need and potential for multicellular modeling in evaluating cell-scale hypotheses.

Comparison with prior 2-D modeling results In 3D, neither necrosis model reproduced the mechanical "tears" between the proliferative rim and the necrotic core predicted by earlier 2-D simulations [16]; this is because more viable tissue is fluxing into smaller necrotic areas in the 3-D geometry compared to the 2-D geometry.

Availability and Future Directions

PhysiCell is available from `PhysiCell.MathCancer.org` and `physicell.sf.net` under the (3-clause) BSD license. A tutorial on using the code is included with every PhysiCell download, along with several examples.

Numerical improvements The biggest code bottleneck is cell-cell interaction testing: cell volume can vary by a factor of 100, and hence the cell diameter (and interaction distance) can vary by a factor of 50. The number of cells in the list of interacting neighbors $\mathcal{N}(i)$ scales inversely with the minimum cell volume; see the supplementary material. Future versions of PhysiCell will introduce a nested mesh interaction testing structure to more accurately estimate $\mathcal{N}(i)$ in regions with small cells.

Scientific improvements We will implement additional cell cycle models as the MultiCellDS standard emerges, including several based on flow cytometry data. We will introduce new built-in models for cell motility, and potentially new cell death models (e.g., autophagy). We plan to add cell more advanced cell mechanics models (e.g., as in [9, 29]), and to extend PhysiCell to include extracellular matrix mechanics.

User-focused improvements In the coming months, we will publish a series of blog posts and code samples at `http://MathCancer.org/blog/`, similarly to our efforts for BioFVM [30]. We will create an improved user-friendly API based upon user feedback, and pre-compiled clients that can initiate simulations based upon a digital snapshot (intitial arrangement of cells) and digital cell lines (self-contained, model-independent sets of cell phenotype data), using the emerging MultiCellDS standard [12, 13].

Supporting Information

Video S1

3-D simulation of 18 days of hanging drop tumor spheroid growth from 2300 cells to 1.2 million cells, using the deterministic necrosis model. Available at:
`https://www.youtube.com/watch?v=WMhYW9D4SqM`

Video S2

3-D simulation of 18 days of hanging drop tumor spheroid growth from 2300 cells to 1 million cells, using the stochastic necrosis model. Available at:
`https://www.youtube.com/watch?v=xr0qqJ_Exd4`

Video S3

3-D simulation video of 30 days of DCIS growth in a 1 mm length of breast duct, using the deterministic necrosis model. Available at:
`https://www.youtube.com/watch?v=ntVKOr9poro`

Video S4

3-D simulation video of 30 days of DCIS growth in a 1 mm length of breast duct, using the stochastic necrosis model. Available at:
`https://www.youtube.com/watch?v=-lRot-dfwJk`

Supplementary materials

Extensive supplementary materials include: full mathematical model details, supporting literature, and reference parameter values for MCF-7 cancer cells; expanded numerical implementation details; convergence and validation testing results; full parameter values for the main tests; and an expanded feature comparison of PhysiCell and other 3-D multicellular simulation projects.

Acknowledgments

We thank David B. Agus and the USC Center for Applied Molecular Medicine for generous institutional support. Partial support was provided by the USC James H. Zumberge Research and Innovation Fund (2012 Large Interdisciplinary Award), the Breast Cancer Research Foundation, and the National Institutes of Health (5U54CA143907,1R01CA180149). We thank Margy Villa for her assistance in early testing, and Edwin Juarez and Rishi Rawat for useful discussions. We greatly appreciate the deep discussions with the MultiCellDS review panel in developing standardized cell cycle and death models.

References

1. Macklin P. Biological background. In: V. Cristini and J.S. Lowengrub, Multiscale Modeling of Cancer: An Integrated Experimental and Mathematical Modeling Approach. Cambridge, UK: Cambridge University Press; 2010. p. 8–23. (invited author: P. Macklin). Available from: `http://dx.doi.org/10.1017/CBO9780511781452.003`.

2. Lowengrub J, Frieboes HB, Jin F, Chuang YL, Li X, Macklin P, et al. Nonlinear modeling of cancer: Bridging the gap between cells and tumors. Nonlinearity. 2010;23(1):R1–R91. (invited author: J. Lowengrub). Available from: `http://dx.doi.org/10.1088/0951-7715/23/1/R01`.

3. Deisboeck TS, Wang Z, Macklin P, Cristini V. Multiscale Cancer Modeling. Annu Rev Biomed Eng. 2011;13:127–155. (invited author: T.S. Deisboeck). Available from: `http://dx.doi.org/10.1146/ANNUREV-BIOENG-071910-124729`.

4. Hatzikirou H, Chauviere A, Bauer AL, Leier A, Lewis MT, Macklin P, et al. Integrative physical oncology. WIREs Syst Biol Med. 2012;4(1):1–14. (invited author: V. Cristini). Available from: `http://dx.doi.org/10.1002/wsbm.158`.

5. Macklin P, Frieboes HB, Sparks JL, Ghaffarizadeh A, Friedman SH, Juarez EF, et al. Progress Towards Computational 3-D Multicellular Systems Biology. In: Rejniak KA, editor. Systems Biology of Tumor Microenvironment, Advances in Experimental Medicine and Biology. vol. 936. Bern, Switzerland: Springer; 2016 (in press). (invited author: P. Macklin). Available from: `http://dx.doi.org/10.1007/978-3-319-42023-3_12`.

6. Ghaffarizadeh A, Friedman SH, Macklin P. BioFVM: an efficient, parallelized diffusive transport solver for 3-D biological simulations. Bioinformatics. 2016;32(8):1256–1258. Available from: `http://dx.doi.org/10.1093/bioinformatics/btv730`.

7. Swat MH, Thomas GL, Belmonte JM, Shirinifard A, Hmeljak D, Glazier JA. Chapter 13 - Multi-Scale Modeling of Tissues Using CompuCell3D. In: Asthagiri AR, Arkin AP, editors. Computational Methods in Cell Biology. vol. 110 of Methods in Cell Biology. Academic Press; 2012. p. 325 – 366. Available from: `http://dx.doi.org/10.1016/B978-0-12-388403-9.00013-8`.

8. Starruß J, de Back W, Brusch L, Deutsch A. Morpheus: a user-friendly modeling environment for multiscale and multicellular systems biology. Bioinformatics. 2014;30(9):1331–1332. Available from: `http://dx.doi.org/10.1093/bioinformatics/btt772`.

9. Hoehme S, Drasdo D. A cell-based simulation software for multi-cellular systems. Bioinformatics. 2010;26(20):2641–2642. Available from: `http://dx.doi.org/10.1093/bioinformatics/btq437`.

10. Mirams GR, Arthurs CJ, Bernabeu MO, Bordas R, Cooper J, Corrias A, et al. Chaste: An Open Source C++ Library for Computational Physiology and Biology. PLoS Comput Biol. 2013 03;9(3):1–8. Available from: `http://dx.doi.org/10.1371/journal.pcbi.1002970`.

11. Kang S, Kahan S, McDermott J, Flann N, Shmulevich I. Biocellion: accelerating computer simulation of multicellular biological system models. Bioinformatics. 2014;30(21):3101–3108. Available from: `http://dx.doi.org/10.1093/bioinformatics/btu498`.

12. Friedman SH, et al. MultiCellDS : a community-developed standard for curating microenvironment-dependent multicellular data. Nat Sci Rep. 2016 (in revision);.

13. Macklin P, et al.. MultiCellDS Project Website; 2014-2016. Available from: `https://MultiCellDS.org`.

14. Ghaffarizadeh A, Friedman SH, Macklin P. Agent-based simulation of large tumors in 3-D microenvironments. bioRxiv. 2015;Available from: `http://dx.doi.org/10.1101/035733`.

15. Kapoulkine A. pugixml: Light-weight, simple and fast XML parser for C++ with XPath support; 2015. Available from: `https://github.com/zeux/pugixml`.

16. Macklin P, Edgerton ME, Thompson AM, Cristini V. Patient-calibrated agent-based modelling of ductal carcinoma in situ (DCIS): From microscopic measurements to macroscopic predictions of clinical progression. J Theor Biol. 2012;301:122–40. Available from: `http://dx.doi.org/10.1016/j.jtbi.2012.02.002`.

17. Simpson MJ, Towne C, McElwain DLS, Upton Z. Migration of breast cancer cells: Understanding the roles of volume exclusion and cell-to-cell adhesion. Phys Rev E. 2010 Oct;82:041901. Available from: `http://link.aps.org/doi/10.1103/PhysRevE.82.041901`.

18. Drasdo D, Kree R, McCaskill JS. Monte Carlo approach to tissue-cell populations. Phys Rev E. 1995 Dec;52:6635–6657. Available from: `http://dx.doi.org/10.1103/PhysRevE.52.6635`.

19. Galle J, Loeffler M, Drasdo D. Modeling the Effect of Deregulated Proliferation and Apoptosis on the Growth Dynamics of Epithelial Cell Populations In Vitro. Biophys J. 2005;88(1):62–75. Available from: `http://dx.doi.org/10.1529/biophysj.104.041459`.

20. Edgerton ME, Chuang YL, Macklin P, Yang W, Bearer EL, Cristini V. A novel, patient-specific mathematical pathology approach for assessment of surgical volume: Application to ductal carcinoma in situ of the breast. Anal Cell Pathol. 2011;34(5):247–63. Available from: `http://dx.doi.org/10.3233/ACP-2011-0019`.

21. Hyun AZ, Macklin P. Improved patient-specific calibration for agent-based cancer modeling. J Theor Biol. 2013;317:422–4. Available from: `http://dx.doi.org/10.1016/j.jtbi.2012.10.017`.

22. Macklin P, Mumenthaler S, Lowengrub J. Modeling multiscale necrotic and calcified tissue biomechanics in cancer patients: application to ductal carcinoma in situ (DCIS). In: Gefen A, editor. Multiscale Computer Modeling in Biomechanics and Biomedical Engineering. Berlin, Germany: Springer; 2013. p. 349–80. (invited author: P. Macklin). Available from: `http://dx.doi.org/10.1007/8415_2012_150`.

23. Majno G, Joris I. Apoptosis, oncosis, and necrosis. An overview of cell death. The American journal of pathology. 1995;146(1):3.

24. Trump BE, Berezesky IK, Chang SH, Phelps PC. The Pathways of Cell Death: Oncosis, Apoptosis, and Necrosis. Toxicologic Pathology. 1997;25(1):82–88. Available from: http://tpx.sagepub.com/content/25/1/82.abstract.

25. Garvey CM, Spiller E, Lindsay D, Chiang CT, Choi NC, Agus DB, et al. A high-content image-based method for quantitatively studying context-dependent cell population dynamics. Sci Rep. 2016;6:29752. Available from: http://dx.doi.org/10.1038/srep29752.

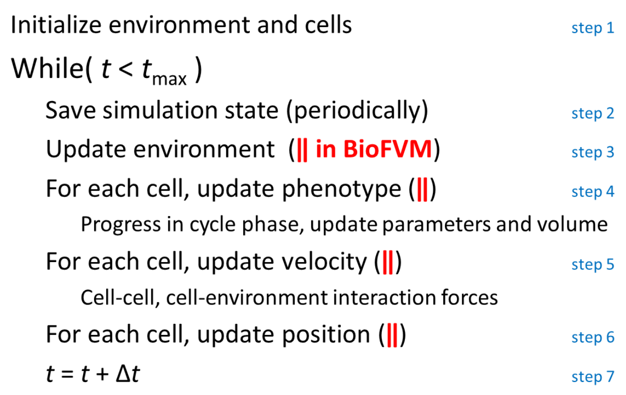
26. McKeown SR. Defining normoxia, physoxia and hypoxia in tumours—implications for treatment response. The British Journal of Radiology. 2014;87(1035):20130676. PMID: 24588669. Available from: http://dx.doi.org/10.1259/bjr.20130676.

27. Ghosh S, Joshi MB, Ivanov D, Feder-Mengus C, Spagnoli GC, Martin I, et al. Use of multicellular tumor spheroids to dissect endothelial cell–tumor cell interactions: A role for T-cadherin in tumor angiogenesis. FEBS Letters. 2007;581(23):4523–4528. Available from: http://dx.doi.org/10.1016/j.febslet.2007.08.038.

28. Ma Hl, Jiang Q, Han S, Wu Y, Tomshine JC, Wang D, et al. Multicellular Tumor Spheroids as an in Vivo–Like Tumor Model for Three-Dimensional Imaging of Chemotherapeutic and Nano Material Cellular Penetration. Molecular Imaging. 2012;11(6). Available from: http://mix.sagepub.com/content/11/6/7290.2012.00012.abstract.

29. Buske P, Galle J, Barker N, Aust G, Clevers H, Loeffler M. A Comprehensive Model of the Spatio-Temporal Stem Cell and Tissue Organisation in the Intestinal Crypt. PLoS Comput Biol. 2011 01;7(1):1–13. Available from: http://dx.doi.org/10.1371/journal.pcbi.1001045.

30. Macklin P. MathCancer Blog: BioFVM Tutorials; 2016. Available from: http://MathCancer.org/blog/biofvm-tutorials/.



```
Initialize environment and cells                                   step 1
While( $t < t_{max}$ )
    Save simulation state (periodically)                           step 2
    Update environment  (|| in BioFVM)                             step 3
    For each cell, update phenotype (||)                           step 4
        Progress in cycle phase, update parameters and volume
    For each cell, update velocity (||)                            step 5
        Cell-cell, cell-environment interaction forces
    For each cell, update position (||)                            step 6
    $t = t + \Delta t$                                             step 7
```

Figure 1. Overall PhysiCell program flow. $\|$ symbol: parallelized with OpenMP.

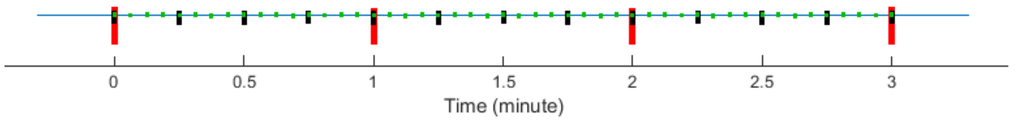


Figure 2. PhysiCell and multiple time scales: PhysiCell uses BioFVM to update the microenvironment at the short green tick marks, corresponding to Δt_{diff}. It updates cell mechanics (including cell position) less frequently at the medium black tick marks (Δt_{mech}), and it runs the cell volume and cycle/death models least frequently at the long red tick marks (Δt_{cell}). Note that the time steps shown are for illustrative purpose; the default step sizes are given in the supplementary materials.

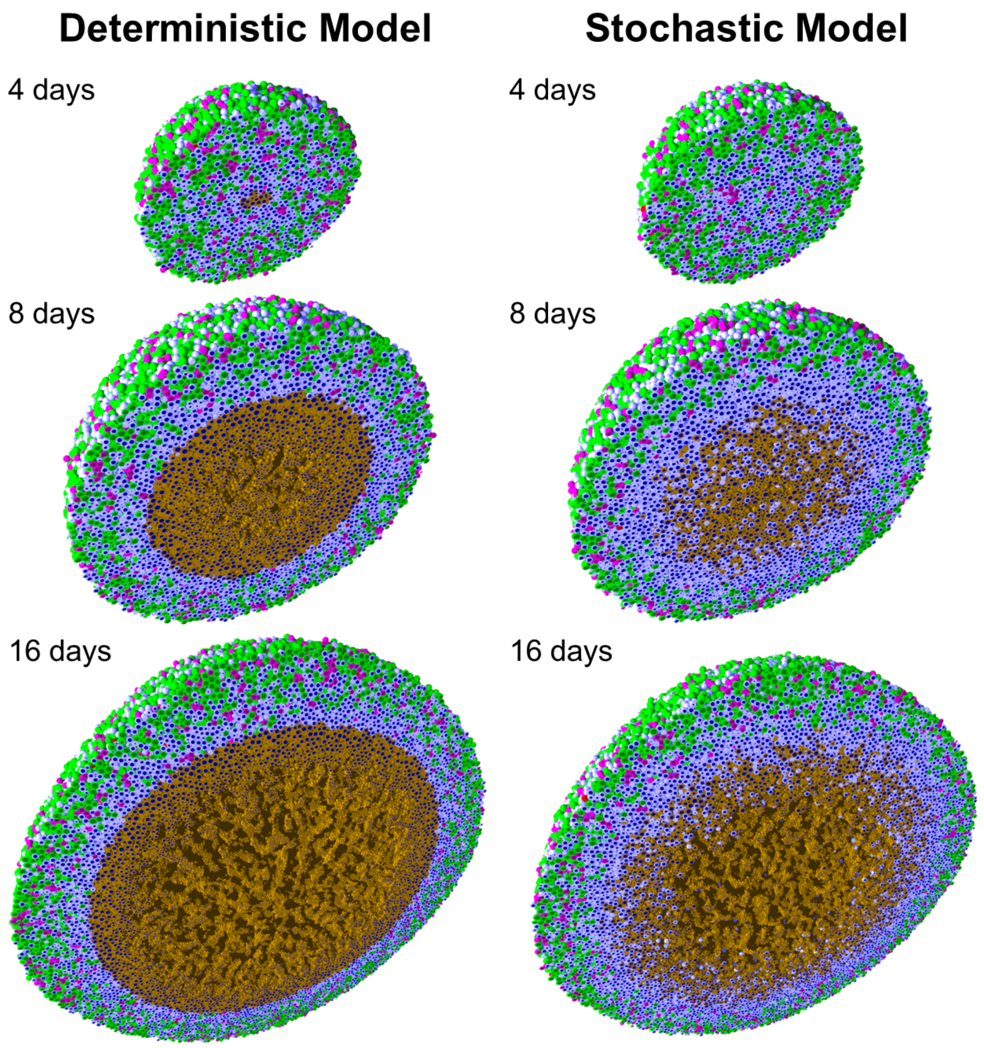


Figure 3. Hanging drop spheroid (HDS) simulations with deterministic necrosis (left) and stochastic necrosis (right), plotted at 4, 8, and 16 days. Videos are available at Video S1 and Video S2. ***Legend:*** Ki67+ cells are green before mitosis (K_1) and magenta afterwards (K_2). Pale blue cells are Ki67- (Q), dead cells are red (apoptotic) and brown (necrotic), and nuclei are dark blue.

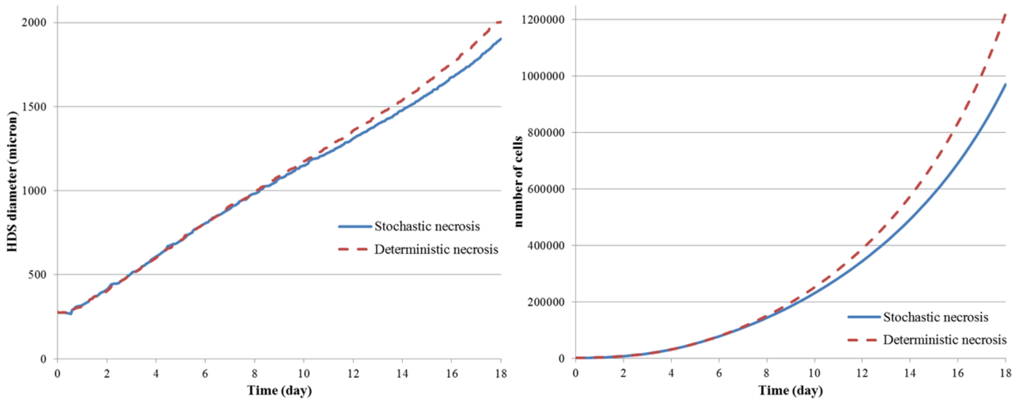


Figure 4. HDS growth: ***Left:*** The deterministic and stochastic necrosis models both give approximately linear growth (left), but the HDS with deterministic necrosis model grows faster (∼ 5% difference in diameter at day 18). ***Right:*** The HDS with stochastic necrosis has fewer cells than the deterministic model (∼ 26% difference in cell count at day 18), due to its delay in necrosis. The difference in cell count is larger than the difference in tumor diameter because most of the difference lies in the number of necrotic cells, and necrotic cells are smaller than viable cells.

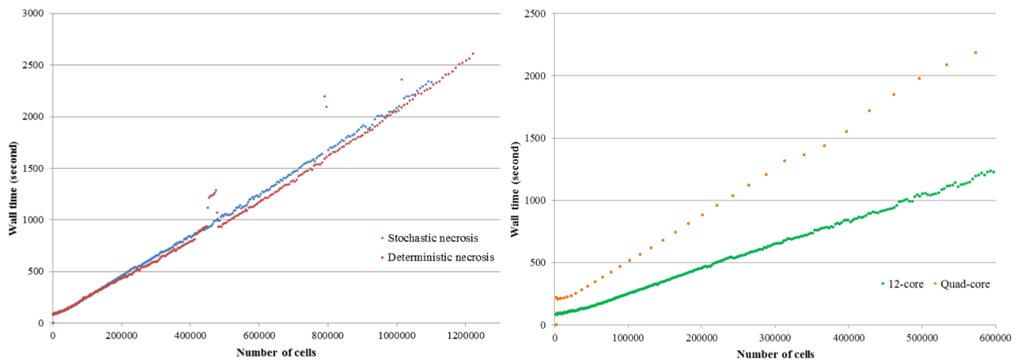


Figure 5. HDS computational cost scaling: ***Left:*** Wall-time vs. cell count for the stochastic (red) and deterministic (blue) necrosis necrosis models on a single HPC compute node. Both models show approximately linear cost scaling with the number of cell agents. ***right:*** Wall time vs. cell count for stochastic necrosis model on the desktop workstation (orange) and the single HPC node (green).

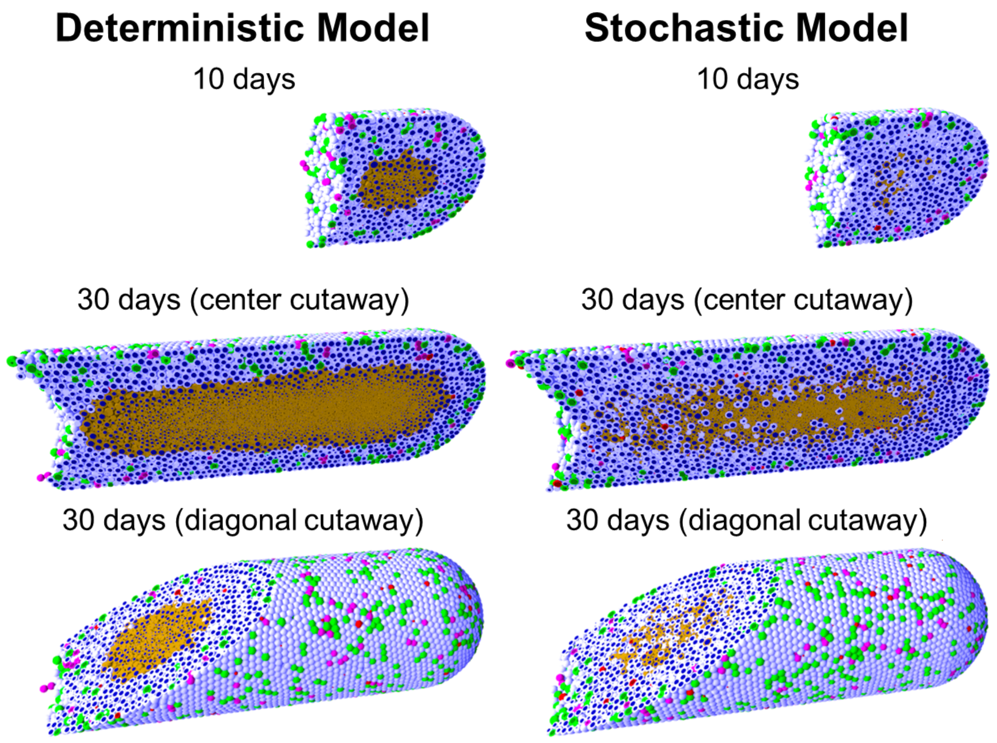


Figure 6. Ductal carcinoma in situ (DCIS) simulations with deterministic necrosis (left) and stochastic necrosis (right), plotted at 10 and 30 days (multiple views). Videos are available at Video S3 and Video S4. The figure legend is the same as Fig.3.

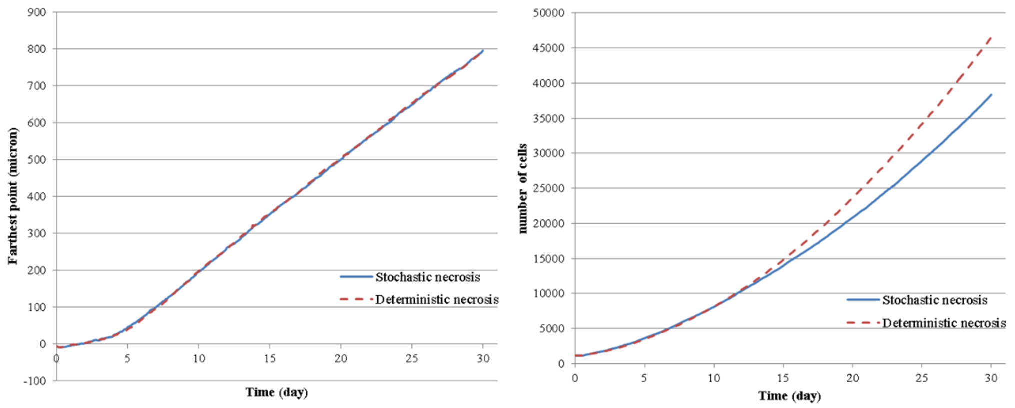


Figure 7. DCIS growth: The deterministic and stochastic necrosis models both result in linear DCIS growth at approximately 1 cm/year (left), even while their cell counts differ by 21% by the end of the simulations (right).