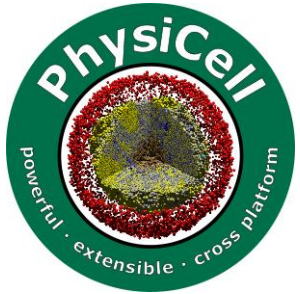
PhysiCell: an opensource framework for multicell modeling

Furkan Kurtoglu
@FKurtogluSysBio

PhysiCell Project

February 14, 2022



EMMCCT 2022 – Mini Course 2

- Today (14/2/2022) – Furkan
  - Introduction to Agent-Based Modeling (ABM)
  - PhysiCell
  - PhysiCell First Dive
- Tomorrow (15/2/2022) – Aneequa
  - Microenvironment
  - Phenotype
- Wednesday (16/2/2022) – Aneequa & Furkan
  - Functions
  - Model Building in PhysiCell (Chemical Communication)

Goals

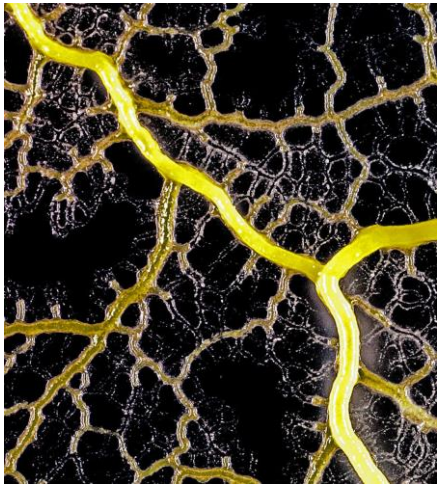
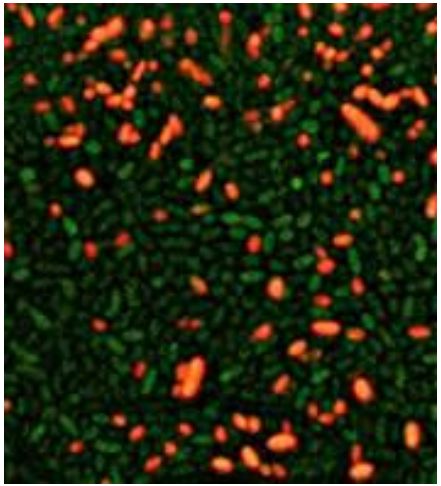
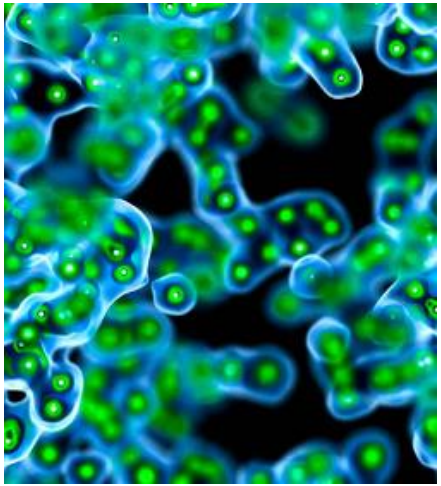
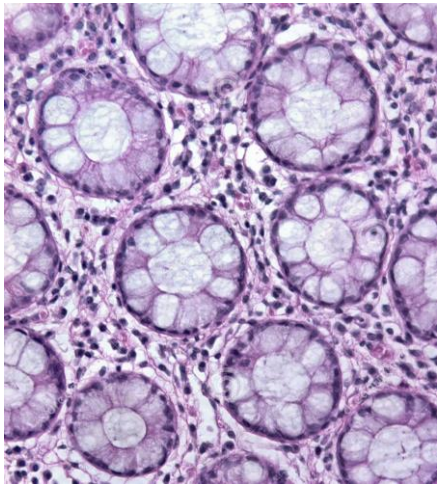
- Learn the motivation for and concepts of agent-based modeling
- Briefly survey the main types of agent-based modeling approaches
- Learn about PhysiCell's agent modeling approaches
- See some examples

Simple single-cell behaviors …

- Growth
- Division
- Death
- Adhesion
- Mechanics
- Motility
- Secretion
- Uptake
- Sampling
- Predation
- Differentiation
- …

Give rise to complex systems

- ***Multicellular systems***—composed of multiple cells of multiple types—can exhibit remarkable diversity, with complex emergent behaviors.

How do these systems self-organize and sustain themselves?

How do we understand these multiscale systems?

Interconnected systems and processes:

- Single-cell behaviors
- Cell-cell communication
- Physics-imposed constraints (e.g., diffusion)
- Systems of systems (e.g., immune system)

In diseases, these systems become dysregulated.

Treatments target *parts* of these systems.

Health is a **complex system:**
changing one part can have **surprising effects**!

Modeling can help **understand** this system.
This is **multicellular systems biology.**

If we can **control** these systems, we've arrived at **multicellular systems engineering**.

Source: Hanahan & Weinberg (2011)
DOI: 10.1016/j.cell.2011.02.013

Analogy: multicellular biology as a play

- The **microenvironment** is the **stage**.
- The **cells** are the **actors**.
- The **cell actors** follow their own **scripts**.

- ***BUT***:
  - The scripts change based on the stage. (microenvironment-dependent phenotype)
  - The actors' dialog is critical. (cell-cell communication)
  - The actors can tear up and remodel the stage. (tissue remodeling)
  - The actors can ignore their scripts and ad lib. (Mutations, evolution)

It's our job as scientists to figure out each actor's script by watching the play.

Clinicians and engineers want to rewrite the script.

Agent-based modeling is a modeling paradigm for these complex multicellular systems:

Cells are ***software agents*** that move and live a ***virtual tissue environment***.

What is a discrete model?

- "**Discrete**" applies to discrete mathematics.

- **Continuum models** describe *continuous variables* with continuous (and differentiable) operations. The variables take continuous values. (e.g., positive real numbers)
  - **Example:** a cell population density ρ modeled with the Fisher's equation with diffusion (D) and a birth rate (r) up to a carrying capacity ($\rho_{\max}$).

$$\frac{\partial \rho}{\partial t} = D\nabla^2\rho + r\rho\left(1 - \frac{\rho}{\rho_{\max}}\right)$$

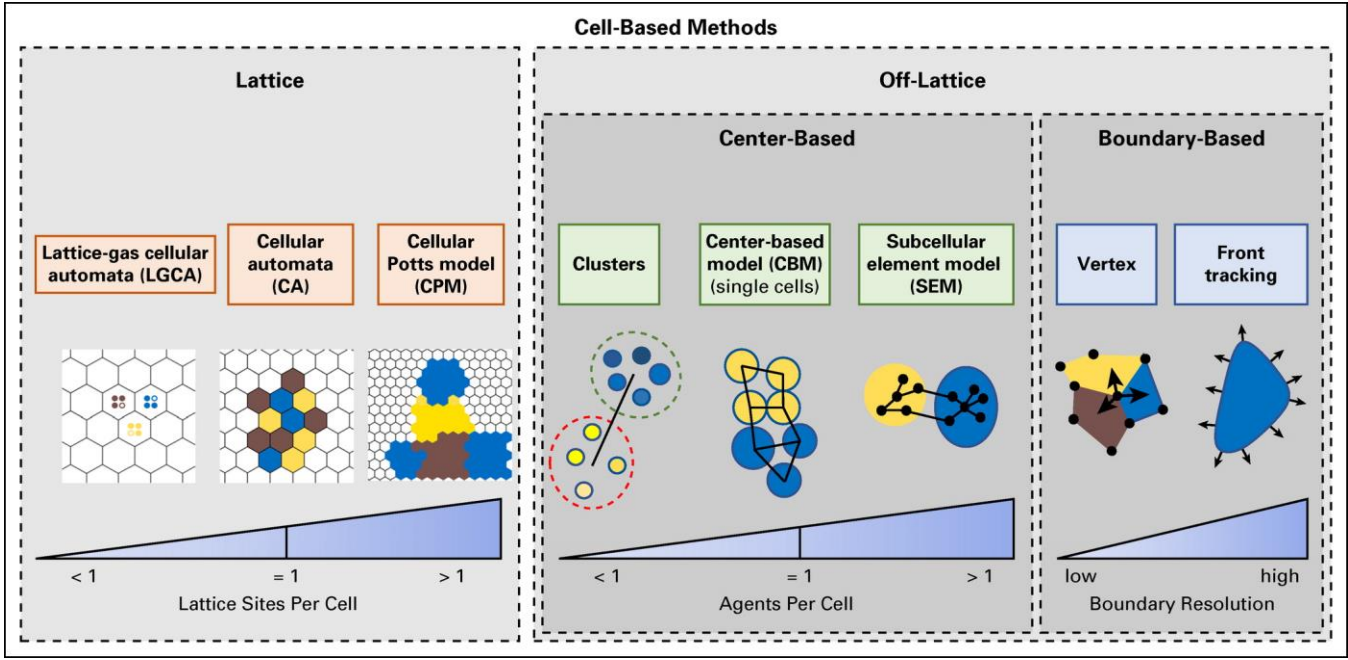
- **Discrete models** describe *distinct individuals* with discrete events. The variables tend to take discrete values. (e.g., Boolean or integer variables)
  - **Example:** A cell population $X(t)$ models birth events as a Poisson process with rate r: Between now (t) and the next time step ($t + \Delta t$), each cell has a probability $P = r\Delta t$ of a birth event that increases X by one.

Typical ABM program flow

- Read parameters
- Set up microenvironment
  - Create meshes, initialize chemical substrates, diffusion solvers, etc.
- Set up cell agents
  - Define all cell types
  - Instantiate cells
- For each time:
  - Update microenvironment
    - Solve reaction-diffusion equations (as needed)
    - Solve tissue mechanics (as needed)
  - Update each cell's state
    - Sample environment
    - Run signaling model (as needed)
    - Update behavioral parameters based on signaling model and sampled environment
    - Run cell process models (growth, cycling, death, …)
  - Calculate cell velocities
  - Update cell positions
  - Advance time

Types of cell-based models

- **lattice-bound**
  - resolution:
    - ♦ < 1 site / cell:
      - » lattice gas
    - ♦ 1 site / cell
      - » cellular automaton
    - ♦ many sites / cell
      - » cellular Potts
- **off-lattice**
  - **center-based**
  - boundary-based



J. Metzcar, Y. Wang, R. Heiland, and P. Macklin. A review of cell-based computational modeling in cancer biology. *JCO Clinical Cancer Informatics* 3:1-13, 2019 (invited review). DOI: 10.1200/CCI.18.00069.

Where does PhysiCell fit in?

- PhysiCell is an **off-lattice, center-based** modeling platform
  - **Spatial resolution**: one agent per cell
  - **Trick:** Use bigger agents to model cell collections or pieces of tissue.
  - **Trick:** Use smaller agents to model cell parts

- PhysiCell couples with PDE models of the microenvironment, making it a **hybrid discrete-continuum approach.**
  - Since most useful agent-based models are coupled to PDE models of the microenvironment, we simply refer to them as agent-based models.

- PhysiCell uses ODEs and other technical to model dynamical details in individual cells. This makes it **multiscale.**

BioFVM: Simulating 3-D biotransport

Design goal: Simulate multiple diffusing substrates in 3D with desktops or single HTC/HPC nodes

Typical use: pO_2, glucose, metabolic waste, signaling factors, and a drug, on 10 mm^3 at 20 μm resolution

Features:

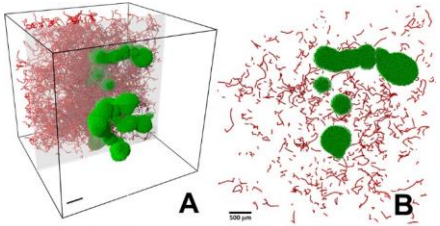
- Off-lattice cell secretion and uptake
- 2nd-order accurate (space), 1st-order accurate (time), numerically stable

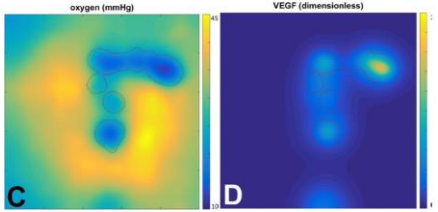
Method:

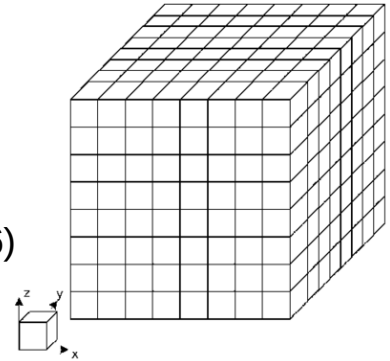
- Operator splitting, LOD, customized Thomas solvers, etc.
- Standard C++11, cross-platform
- OpenMP parallelization
- O(*n*) cost scaling in # substrates, # voxels
- Easy to simulate 5-10 substrates on 10^6 voxels

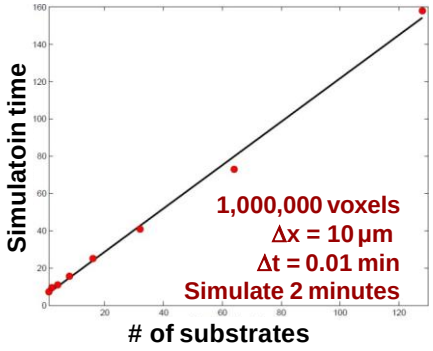
Reference: Ghaffarizadeh et al., *Bioinformatics* (2016)

DOI: 10.1093/bioinformatics/btv730









BioFVM: Simulating 3-D biotransport

$$\frac{\partial \boldsymbol{\rho}}{\partial t} = \overbrace{\mathbf{D}\nabla^2\boldsymbol{\rho}}^{\text{diffusion}} - \overbrace{\boldsymbol{\lambda}\boldsymbol{\rho}}^{\text{decay}} + \overbrace{\mathbf{S}(\boldsymbol{\rho}^* - \boldsymbol{\rho})}^{\text{bulk source}} - \overbrace{\mathbf{U}\boldsymbol{\rho}}^{\text{bulk uptake}}$$
$$+ \overbrace{\sum_{\text{cells } k} \delta(\mathbf{x} - \mathbf{x}_k) W_k [\mathbf{S}_k(\boldsymbol{\rho}_k^* - \boldsymbol{\rho}) - \mathbf{U}_k \boldsymbol{\rho}]}^{\text{sources and uptake by cells}} \quad \text{in } \Omega$$

PhysiCell: A multicellular framework

Design goal: Simulate 10^6 or more cells in 2D or 3D on desktops or single HPC nodes

Features:

- Off-lattice cell positions
- Mechanics-based cell movement
- Cell processes (cycling, motility, …)
- Signal-dependent phenotype
- Can dynamically attach custom data and functions on a cell-by-cell basis
- **Deployed from Raspberry Pi to Crays**

2019 PLoS Computational Biology Research Prize *for* Public Impact

Method:

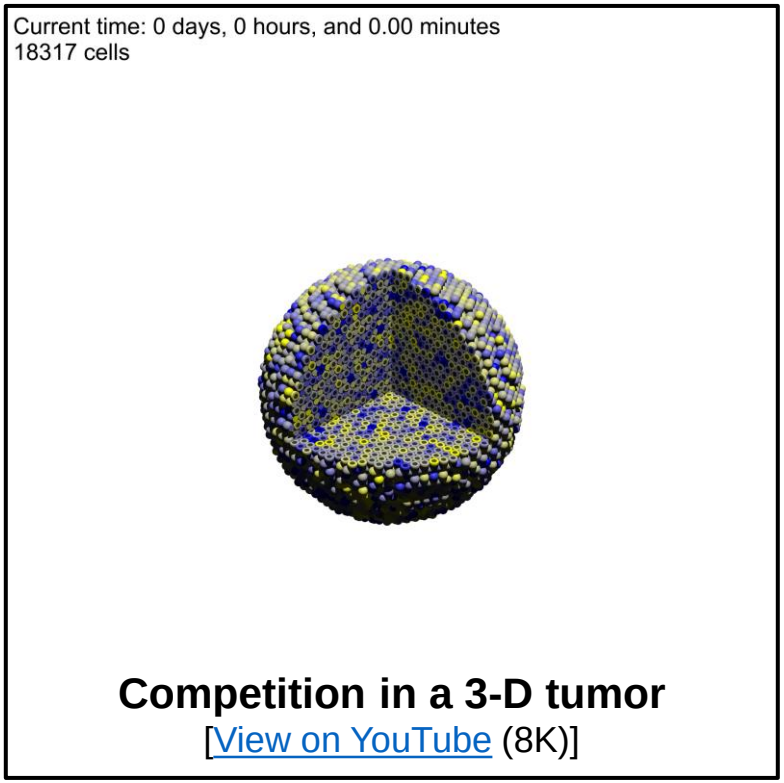
- Standard C++11, cross-platform
- OpenMP parallelization
- O(n) cost scaling in # cells

Try this model yourself!

nanohub.org/tools/pc4heterogen

Reference: Ghaffarizadeh et al., PLoS Comput. Biol. (2018)

DOI: 10.1371/journal.pcbi.1005991



Competition in a 3-D tumor

[View on YouTube (8K)]

Key components of PhysiCell

BioFVM: simulate the (chemical) microenvironment (the stage)

- chemical diffusion and decay
- boundary conditions
- cell-based secretion, uptake, and export

PhysiCell: simulate individual cell agents (the players)

- live in a microenvironment
- single-cell biological behaviors
- cell functions model biological hypotheses to trigger core behaviors
- cell-cell interactions

Key parts of a PhysiCell model (1)

- **Microenvironment (stage):**
  - diffusing substrates
    - diffusion coefficient
    - decay rate
    - boundary conditions
    - Defined in XML configuration file

- **Cell Definitions (types of players):**
  - name
  - default phenotype (more on next page)
  - defined in XML configuration file

Key parts of a PhysiCell model (2)

- **Cell agents (individual players)**:
  - Which cell type? (The cell agent is initialized based on a cell definition.)
  - State variables:
    - position
    - mechanical pressure
    - interaction list (optional)
  - Phenotype **(the script)**
    - Cell cycle
    - Volume
    - Death
    - Motility
    - Mechanics
    - Substrate uptake & release
  - Custom variables
  - Custom functions that act upon the phenotype, variables, and state **(script)**

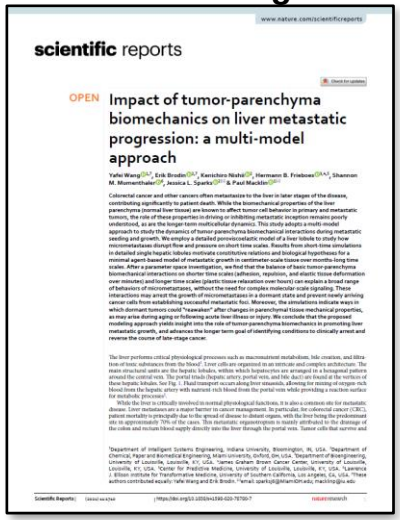
A note about time steps

- PhysiCell is designed to account for the multiple time scales inherent to these problems, and has 3 time scales:
  - $\Delta t_{\text{diffusion}}$ diffusion, secretion, and uptake (default: 0.01 min)
  - $\Delta t_{\text{mechanics}}$ cell movement (default: 0.1 min)
  - Δt_{cell} phenotype and volume changes (default: 6 min)

- This allows some efficiency improvements: not all functions need to be evaluated at each time step.

Some recent examples

Work led by:
Yafei Wang


www.nature.com/scientificreports

scientific reports

OPEN **Impact of tumor-parenchyma biomechanics on liver metastatic progression: a multi-model approach**

Yafei Wang[1,2], Erik Brodin[1,2], Kenichiro Nishii[3], Hermann B. Frieboes[1,2,4], Shannon M. Mumenthaler[3], Jessica L. Sparks[1,2,4] & Paul Macklin[1,2,4]

Colorectal cancer and other cancers often metastasize to the liver in later stages of the disease, contributing significantly to patient death. While the biomechanical properties of the liver parenchyma (normal liver tissue) are known to affect tumor cell behavior in primary and metastatic tumors, the role of these properties in driving or inhibiting metastatic inception remains poorly understood, as are the longer-term multicellular dynamics. This study adopts a multi-model approach to study the dynamics of tumor-parenchyma biomechanical interactions during metastatic seeding and growth. We employ a detailed poroviscoelastic model of a liver lobule to study how micrometastases disrupt flow and pressure on short time scales. Results from short-time simulations in detailed single hepatic lobules motivate constitutive relations and biological hypotheses for a minimal agent-based model of metastatic growth in centimeter-scale tissue over months-long time scales. After a parameter space investigation, we find that the balance of basic tumor-parenchyma biomechanical interactions on shorter time scales (adhesion, repulsion, and elastic tissue deformation over minutes) and longer time scales (plastic tissue relaxation over hours) can explain a broad range of behaviors of micrometastases, without the need for complex molecular-scale signaling. These interactions may arrest the growth of micrometastases in a dormant state and prevent newly arriving cancer cells from establishing successful metastatic foci. Moreover, the simulations indicate ways in which dormant tumors could "reawaken" after changes in parenchymal tissue mechanical properties, as may arise during aging or following acute liver illness or injury. We conclude that the proposed modeling approach yields insight into the role of tumor-parenchyma biomechanics in promoting liver metastatic growth, and advances the longer-term goal of identifying conditions to clinically arrest and reverse the course of late-stage cancer.

The liver performs critical physiological processes such as macronutrient metabolism; bile creation; and filtration of toxic substances from the blood[1]. Liver cells are organized in an intricate and complex architecture. The main structural units are the hepatic lobules, within which hepatocytes are arranged in a hexagonal pattern around the central vein. The portal triads (hepatic artery, portal vein, and bile duct) are found at the vertices of these hepatic lobules. See Fig. 1. Fluid transport occurs along liver sinusoids, allowing for mixing of oxygen-rich blood from the hepatic artery with nutrient-rich blood from the portal vein while providing a reaction surface for metabolic processes[2].

While the liver is critically involved in normal physiological functions, it is also a common site for metastatic disease[3]. Colorectal cancer is a major barrier to cancer management. In particular, for colorectal cancer (CRC), patient mortality is principally due to the spread of disease to distant organs, with the liver being the predominant site in approximately 70% of the cases. This metastatic organotropism is mainly attributed to the drainage of the colon and rectum blood supply directly into the liver through the portal vein. Tumor cells that survive and

[1]Department of Intelligent Systems Engineering, Indiana University, Bloomington, IN, USA. [2]Department of Chemical, Paper and Biomedical Engineering, Miami University, Oxford, OH, USA. [3]Department of Bioengineering, University of Louisville, Louisville, KY, USA. [4]James Graham Brown Cancer Center, University of Louisville, Louisville, KY, USA. [5]Center for Predictive Medicine, University of Louisville, Louisville, KY, USA. [6]Lawrence J. Ellison Institute for Transformative Medicine, University of Southern California, Los Angeles, CA, USA. *These authors contributed equally: Yafei Wang and Erik Brodin. ✉email: sparksjl@miamiOH.edu; macklinp@iu.edu

Scientific Reports | (2021) 11:1593 | https://doi.org/10.1038/s41598-020-78780-7 nature research

Example 1:

tumor-parenchyma interactions in micrometastases

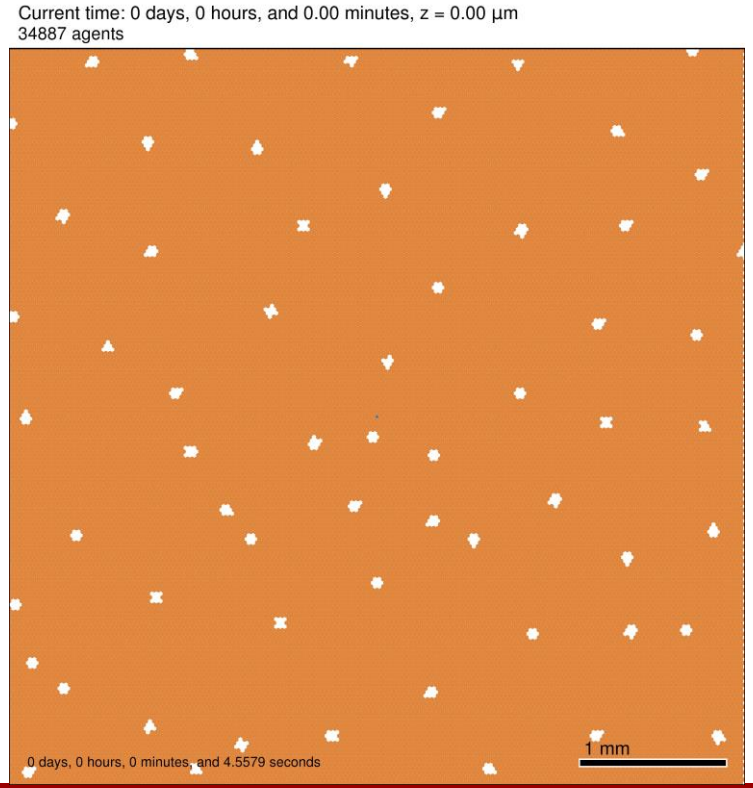
Wang et al. *Sci. Rep.* (2021)
Open Access: https://doi.org/10.1038/s41598-020-78780-7

How can liver parenchyma impact colorectal cancer (CRC) metastases?

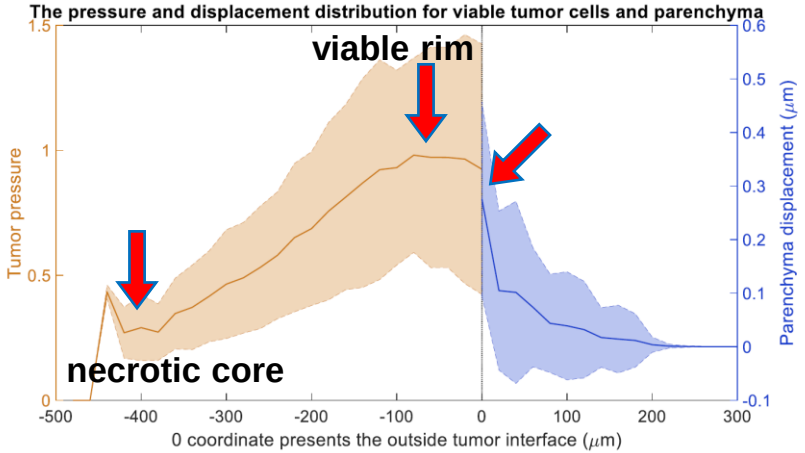
- **How can a micrometastasis clear space for expansion?**
  - Displacement and compression of liver parenchyma
  - Compressive forces on the micrometastasis

- Key model elements: **biomechanical tumor-parenchyma feedbacks**
  - Parenchyma → Tumor:
    - Pressure (compression) down-regulates tumor cell proliferation
  - Tumor → Parenchyma:
    - Parenchyma agents use plastic-elastic model
      - Elastic restorative force on short time scales
      - Plastic reorganization on long time scales
      - Apoptosis under sustained deformation

Growth with feedback



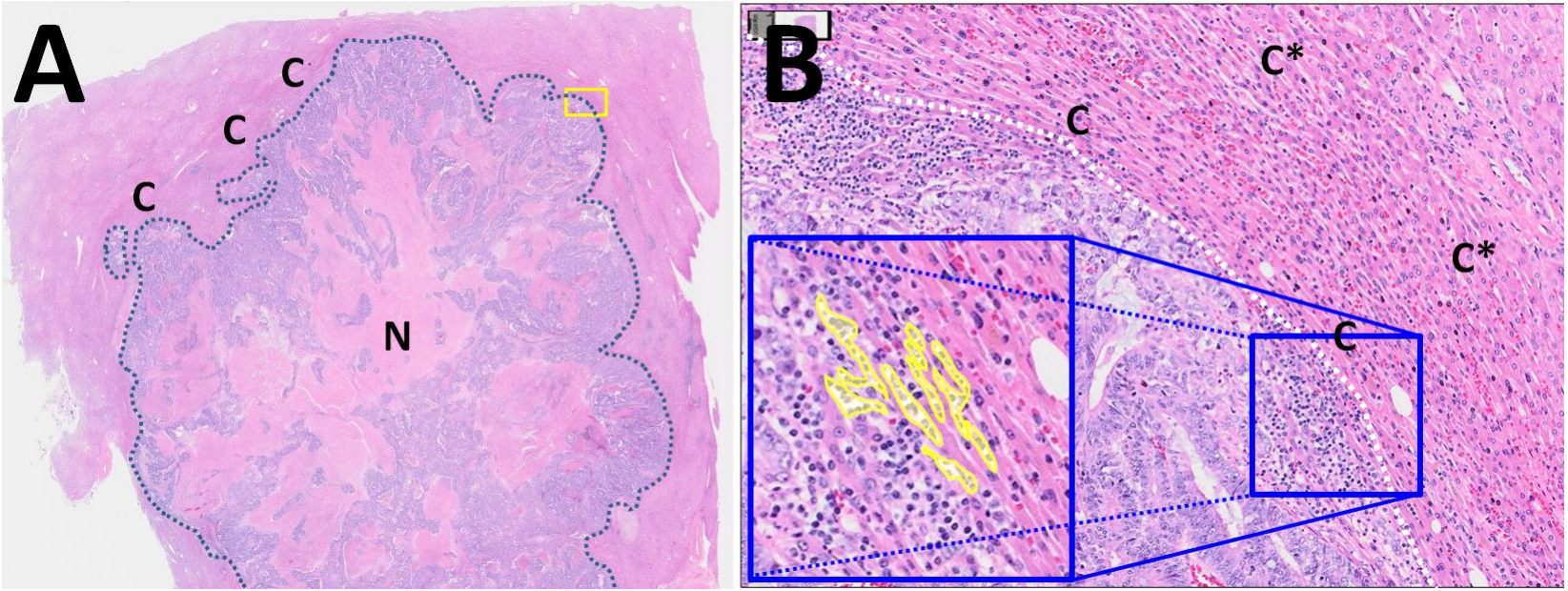
Yellow tumor cells have arrested cycling due to high pressure.



Try this model yourself!

nanohub.org/tools/pc4livermedium

Comparison with a typical clinical sample



Example 2:

Cancer-immune contact interactions

Simple model of cancer-immune interactions

Heterogeneous tumor cells (blue to yellow):

- Cycle entry rate scales with O_2
- Cells necrose in very low O_2
- Yellow cells are most proliferative;
  - blue are least proliferative
- Yellow cells are most immunogenic
  - simplified model of MHC

Immune cells (red):

- Biased random walk towards tumor
- Test for contact with cells
- Form adhesion
- Attempt to induce apoptosis
  - (e.g., FAS receptor)
  - success depends on immunogenicity
- Eventually detach from cell, continue search

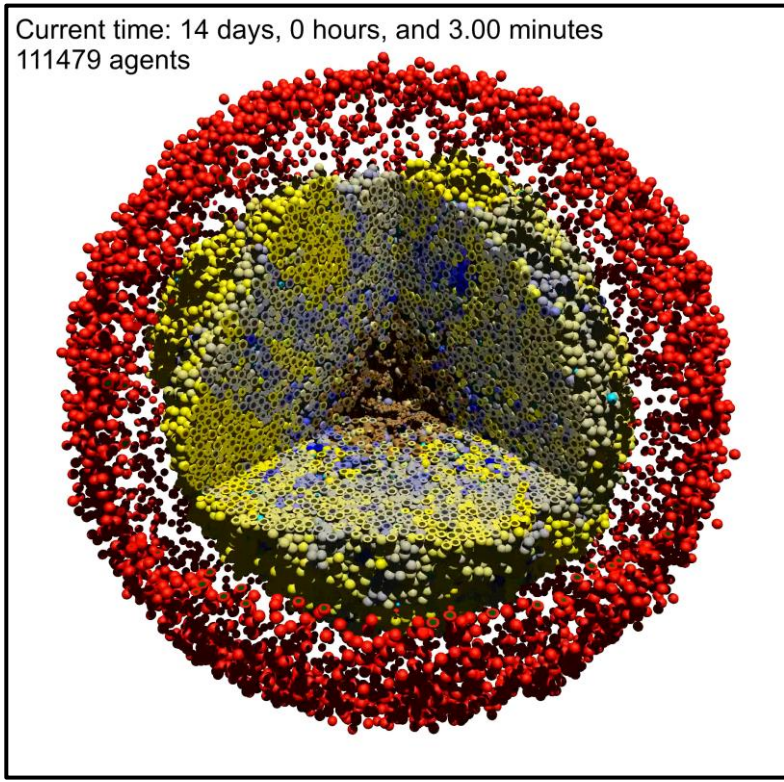
Movie: [View on YouTube (4K)]

References:

- Ghaffarizadeh et al. (2018)
- Ozik et al. (2018)
- Ozik et al. (2019)

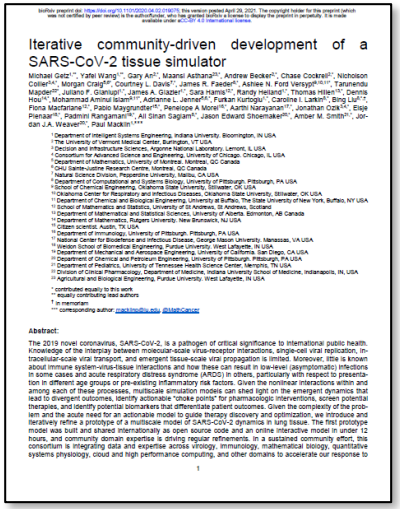
Try this model yourself! (2D)

nanohub.org/tools/pc4cancerimmune



Example 3:

Work led by:
Michael Getz



Iterative community-driven development of a SARS-CoV-2 tissue simulator

Michael Getz[1,*], Yafei Wang[1,*], Gary An[2], Maansi Asthana[3†], Andrew Becker[4], Chase Cockrell[5], Nicholson Collier[6,4], Morgan Craig[7,8], Courtney L. Davis[1], James R. Faeder[9], Ashlee N. Ford Versypt[10,11†], Tarunendu Mapder[12], Juliano F. Gianlupi[13], James A. Glazier[14], Sara Hamis[15], Randy Heiland[14], Thomas Hillen[15], Dennis Hou[16], Mohammad Aminul Islam[9,17], Adrianne L. Jenner[8,4], Furkan Kurtoglu[1], Caroline I. Larkin[8], Bing Liu[6,4], Fiona Macfarlane[18], Pablo Maygrundter[19], Penelope A. Morel[19], Aarthi Narayanan[17,*], Jonathan Ozik[4,*], Elsje Pienaar[18], Padmini Rangamani[20], Ali Sinan Saglam[21], Jason Edward Shoemaker[22], Amber M. Smith[21], Jordan J.A. Weaver[20], Paul Macklin[1,***]

[1] Department of Intelligent Systems Engineering, Indiana University, Bloomington, IN USA
[2] The University of Vermont Medical Center, Burlington, VT USA
[3] Decision and Infrastructure Sciences, Argonne National Laboratory, Lemont, IL USA
[4] Consortium for Advanced Science and Engineering, University of Chicago, Chicago, IL USA
[5] Department of Mathematics, University of Montreal, Montreal, QC Canada
[6] CHU Sainte-Justine Research Centre, Montreal, QC Canada
[7] Natural Science Division, Pepperdine University, Malibu, CA USA
[8] Department of Computational and Systems Biology, University of Pittsburgh, Pittsburgh, PA USA
[9] School of Chemical Engineering, Oklahoma State University, Stillwater, OK USA
[10] Oklahoma Center for Respiratory and Infectious Diseases, Oklahoma State University, Stillwater, OK USA
[11] Department of Chemical and Biological Engineering, University at Buffalo, The State University of New York, Buffalo, NY USA
[12] School of Mathematics and Statistics, University of St Andrews, St Andrews, Scotland
[13] Department of Mathematical and Statistical Sciences, University of Alberta, Edmonton, AB Canada
[14] Department of Mathematics, Rutgers University, New Brunswick, NJ USA
[15] Citizen scientist, Austin, TX USA
[16] Department of Immunology, University of Pittsburgh, Pittsburgh, PA USA
[17] Biocomplexity Institute & Initiative, University of Virginia, Charlottesville, VA USA; University, Manassas, VA USA
[18] Wolfson School of Biomedical Engineering, Purdue University, West Lafayette, IN USA
[19] Department of Mechanical and Aerospace Engineering, University of California, San Diego, CA USA
[20] Department of Chemical and Petroleum Engineering, University of Pittsburgh, Pittsburgh, PA USA
[21] Department of Pediatrics, University of Tennessee Health Science Center, Memphis, TN USA
[22] Division of Clinical Pharmacology, Department of Medicine, Indiana University School of Medicine, Indianapolis, IN, USA
[23] Agricultural and Biological Engineering, Purdue University, West Lafayette, IN USA

* contributed equally to this work.
† equally contributing lead authors
** In memoriam
*** corresponding author: macklinlab.edu, @MathCancer

Abstract:

The 2019 novel coronavirus, SARS-CoV-2, is a pathogen of critical significance to international public health. Knowledge of the interplay between molecular-scale virus-receptor interactions, single-cell viral replication, intracellular-scale viral transport, and emergent tissue-scale viral propagation is limited. Moreover, little is known about immune system–virus–tissue interactions and how these can result in low-level (asymptomatic) infections in some cases and acute respiratory distress syndrome (ARDS) in others, particularly with respect to presentation in different age groups or pre-existing inflammatory risk factors. Given the nonlinear interactions within and among each of these processes, multiscale simulation models can shed light on the emergent dynamics that lead to divergent outcomes, identify actionable "choke points" for pharmacologic interventions, screen potential therapies, and identify potential biomarkers that differentiate patient outcomes. Given the complexity of the problem and the acute need for an actionable model to guide therapy discovery and optimization, we introduce and iteratively refine a prototype of a multiscale model of SARS-CoV-2 dynamics in lung tissue. The first prototype model was built and shared internationally as open source code and an online interactive model in under 12 hours, and community domain expertise is driving regular refinements. In a sustained community effort, this consortium is integrating data and expertise across virology, immunology, mathematical biology, quantitative systems physiology, cloud and high performance computing, and other domains to accelerate our response to

iterative development of a SARS-CoV-2 tissue simulator

Getz et al. *bioRxiv.* (2021)
Preprint: https://doi.org/10.1038/s41598-020-78780-7

Thank you to our coalition!

Multinational:
U.S.
Canada
United Kingdom

Federal partners:
Veterans Affairs
Argonne National Lab

Across Indiana:
Luddy School (lead)
UITS
IU Health
Purdue

Industry:
Pfizer
…

Rapid community-driven development of a SARS-CoV-2 tissue simulator

Michael Getz[1,**], Yafei Wang[1,**], Gary An[2,*], Andrew Becker[2,*], Chase Cockrell[2,*], Nicholson Collier[3,4,*], Morgan Craig[5,6*], Courtney L. Davis[7,*], James Faeder[8,*], Ashlee N. Ford Versypt[9,10,*],Juliano F. Gianlupi[1,*], James A. Glazier[1,*], Sara Hamis[11,*], Randy Heiland[1,*], Thomas Hillen[12,*], Dennis Hou[13,*], Mohammad Aminul Islam[9,*], Adrianne Jenner[5,6,*], Furkan Kurtoglu[1,*], Bing Liu[8,*,†], Fiona Macfarlane[11,*], Pablo Maygrundter[14,*], Penelope A Morel[15,*], Aarthi Narayanan[16,*], Jonathan Ozik[3,4,*], Elsje Pienaar[17,*], Padmini Rangamani[18,*], Jason Edward Shoemaker[19,*], Amber M. Smith[20,*], Paul Macklin[1,***]

[1] Department of Intelligent Systems Engineering, Indiana University. Bloomington, IN USA
[2] The University of Vermont Medical Center. Burlington, VT USA
[11] School of Mathematics and Statistics, University of St Andrews, St Andrews, Scotland.
[12] Dep… Statistical Sciences, University of Albert… Edmonton, AB Canada
[13] Dep… …ers University. New Brunswick, NJ USA
[14] Citiz…
[15] Dep… …ersity of Pittsburgh. Pittsburgh, PA USA
… …d Infectious Disease, George Mason U…
… …ineering, Purdue University. West Lafay…
… …erospace Engineering, University of Cali…
… …oleum Engineering, University of Pittsbu…
… …ity of Tennessee Health Science Cente…
** equa…
† in me…
*** corresponding author: macklinp@iu.edu, @MathCancer

40+ regular contributors from 20+ institutions

Michael Getz
Indiana U.

Yafei Wang
Indiana U.

Note: This is a rapid prototyping project. For the very latest, see http://COVID-19.physicell.org

Collaborative, Iterative Progress

Approach

- **Rapid prototyping**
  - Build, test, and refine
- **Multidisciplinary team**
  - Domain experts guide modelers
  - Subteams work in parallel
  - Integration team coordinates the work
- **Rapid communication**
  - Preprints (open science)
  - Cloud-hosted models for live demos to team experts
- **Open source** software

Progress

Phase I (community building)

- **v1 prototype** (March 2020) built in 12 hours
- **v2 model** (April) added ACE2 receptor trafficking

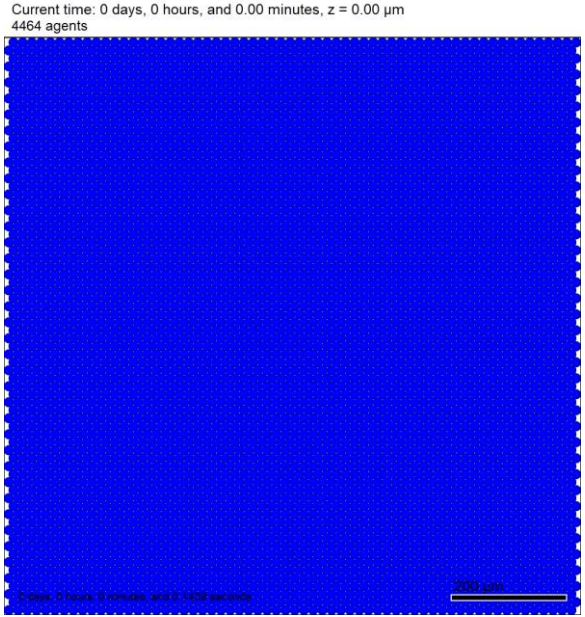
Phase II (community-driven) (current)

- **v3 model** (May-July 2020) added tissue immune responses
- **v4 model** (August-November 2020) added: interferon signaling ● pyroptosis ● systems-scale immune model ● immune cell trafficking ● improved tissue immune model ● better receptor-virus binding ● better viral replication ● tissue fibrosis.
- **v5 model** (November 2020-July 2021) added: neutralizing antibodies ● "bystander" killing by ROS ● anti-inflammatory signals ● improved tissue immune model ● better receptor-virus binding ● ….

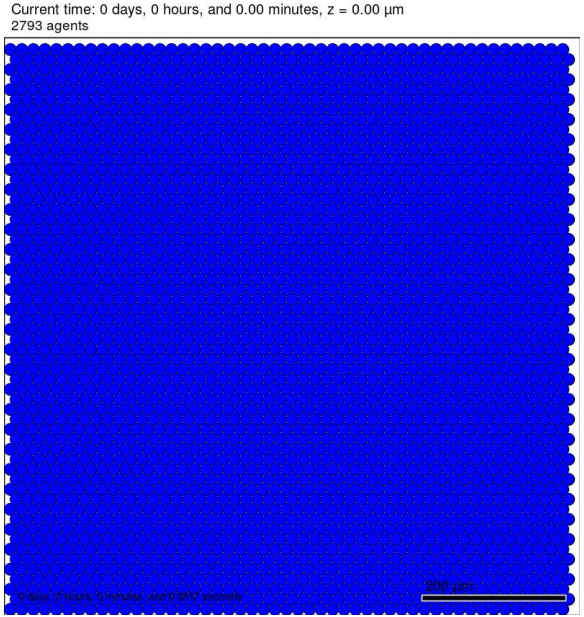
Iterative modeling progress

Versions 1 to 3

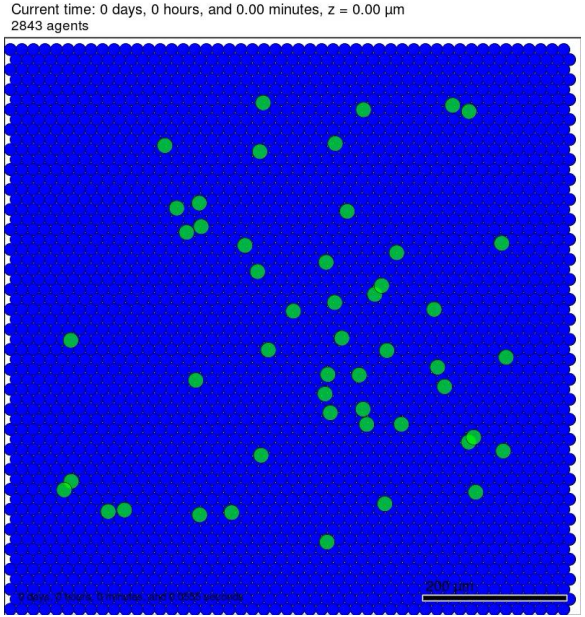
v1: prototype



v2: +ACE2 +random virion seeding



v3: +tissue immune



Version 4

multiscale immune model advances

Improved macrophages:
- Macrophages exhaustion & death
- Phenotype changes from CD8+ T cell contact
  - Stop secreting pro-inflammatory cytokine
- Enable phagocytosis of live infected cells

Dendritic cells:
- Resident DCs activated by virus or infected cells
- DCs traffick to lymph node to drive T cell expansion

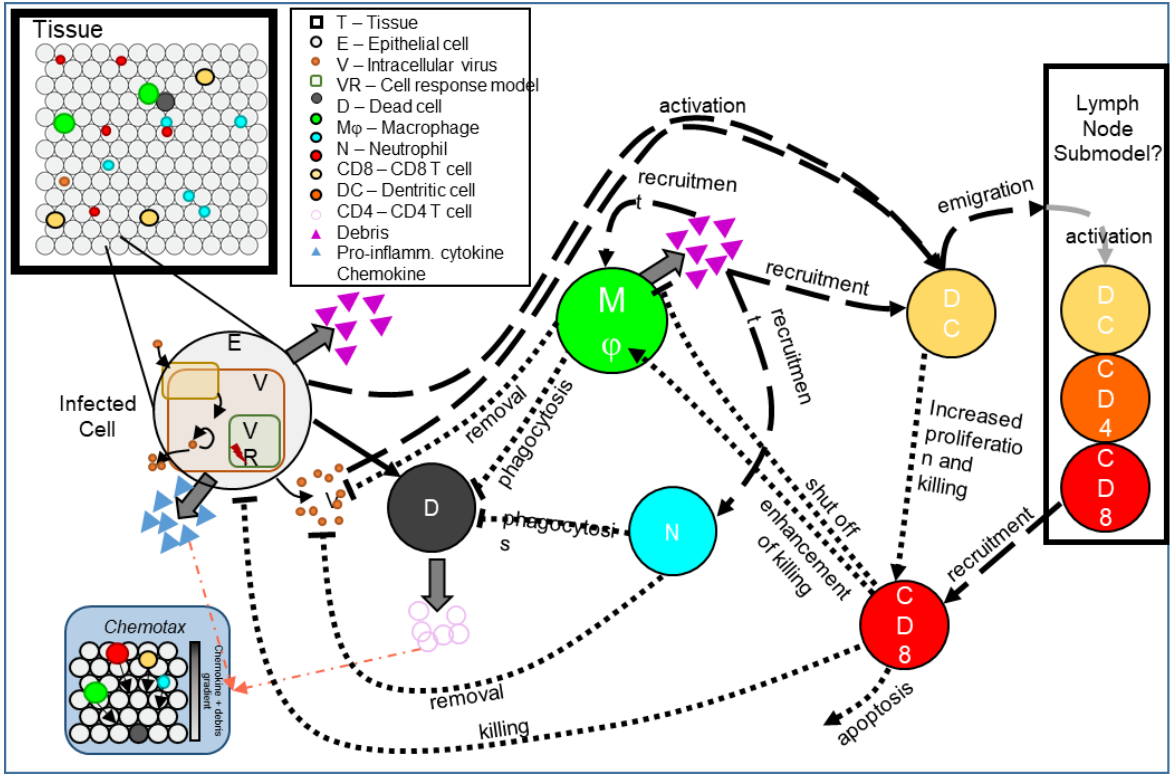
More T cell types

Epithelial cells present antigens

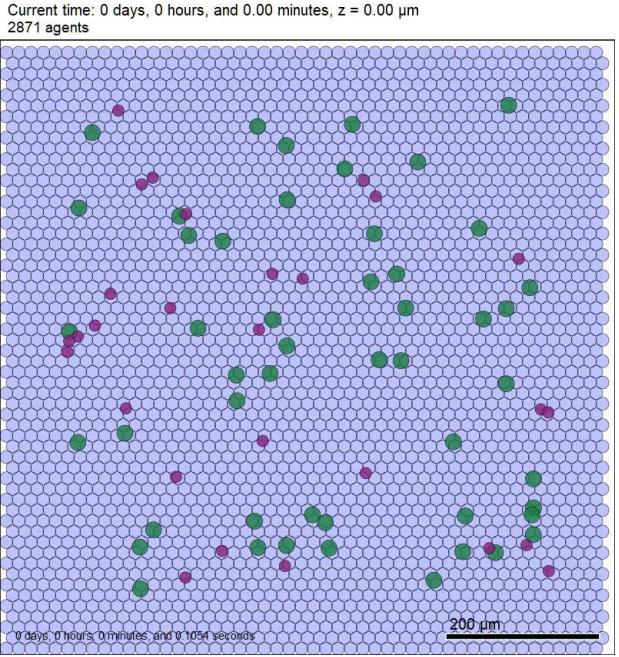
Systems-scale model of immune activation

systems scale:
Tarunendu Mapder
IUPUI

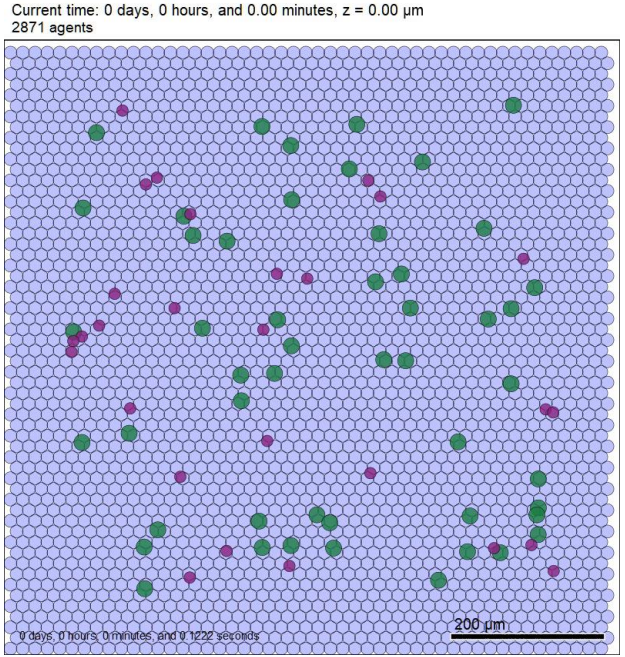
tissue scale:
Adrianne Jenner
U. Montreal



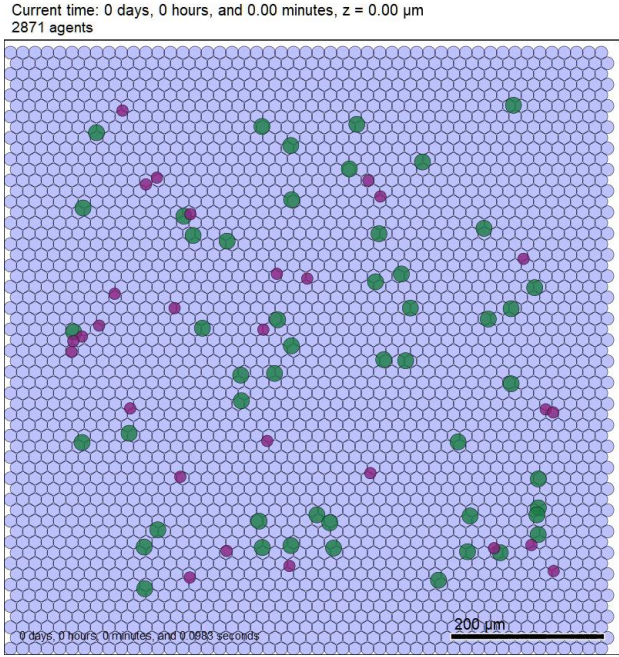
Type I interferons slow progression



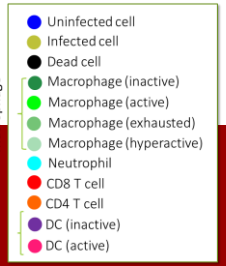
no interferon secretion



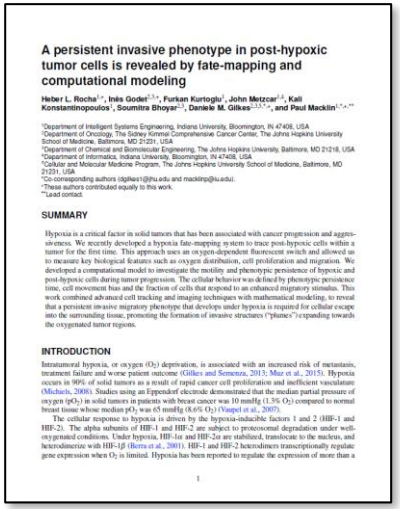
low interferon secretion



10x interferon secretion



Work led by:
Heber L. Rocha



A persistent invasive phenotype in post-hypoxic tumor cells is revealed by fate-mapping and computational modeling

Heber L. Rocha[1,4], Inês Godet[2,3,4], Furkan Kurtoglu[1], John Metzcar[1,4], Kali Konstantinopoulos[3], Soumitra Bhoyar[1,3], Daniele M. Gilkes[2,3,5,6,*], and Paul Macklin[1,4,*,**]

[1]Department of Intelligent Systems Engineering, Indiana University, Bloomington, IN 47408, USA
[2]Department of Oncology, The Sidney Kimmel Comprehensive Cancer Center, The Johns Hopkins University School of Medicine, Baltimore, MD 21231, USA
[3]Department of Chemical and Biomolecular Engineering, The Johns Hopkins University, Baltimore, MD 21218, USA
[4]Department of Informatics, Indiana University, Bloomington, IN 47408, USA
[5]Cellular and Molecular Medicine Program, The Johns Hopkins University School of Medicine, Baltimore, MD 21231, USA
[6]Co-corresponding authors (dgilkes1@jhu.edu and macklinp@iu.edu).
[*]These authors contributed equally to this work.
[**]Lead contact.

SUMMARY

Hypoxia is a critical factor in solid tumors that has been associated with cancer progression and aggressiveness. We recently developed a hypoxia fate-mapping system to trace post-hypoxic cells within a tumor for the first time. This approach uses an oxygen-dependent fluorescent switch and allowed us to measure key biological features such as oxygen distribution, cell proliferation and migration. We developed a computational model to investigate the motility and phenotypic persistence of hypoxic and post-hypoxic cells during tumor progression. The cellular behavior was defined by phenotypic persistence time, cell movement bias and the fraction of cells that respond to an enhanced migratory stimulus. This work combined advanced cell tracking and imaging techniques with mathematical modeling, to reveal that a persistent invasive migratory phenotype that develops under hypoxia is required for cellular escape into the surrounding tissue, promoting the formation of invasive structures ("plumes") expanding towards the oxygenated tumor regions.

INTRODUCTION

Intratumoral hypoxia, or oxygen (O_2) deprivation, is associated with an increased risk of metastasis, treatment failure and worse patient outcome (Gilkes and Semenza, 2013; Muz et al., 2015). Hypoxia occurs in 90% of solid tumors as a result of rapid cancer cell proliferation and inefficient vasculature (Michiels, 2008). Studies using an Eppendorf electrode demonstrated that the median partial pressure of oxygen (pO_2) in solid tumors in patients with breast cancer was 10 mmHg (1.3% O_2) compared to normal breast tissue whose median pO_2 was 65 mmHg (8.6% O_2) (Vaupel et al., 2007).

The cellular response to hypoxia is driven by the hypoxia-inducible factors 1 and 2 (HIF-1 and HIF-2). The alpha subunits of HIF-1 and HIF-2 are subject to proteosomal degradation under well-oxygenated conditions. Under hypoxia, HIF-1α and HIF-2α are stabilized, translocate to the nucleus, and heterodimerize with HIF-1β (Berra et al., 2001). HIF-1 and HIF-2 heterodimers transcriptionally regulate gene expression when O_2 is limited. Hypoxia has been reported to regulate the expression of more than a

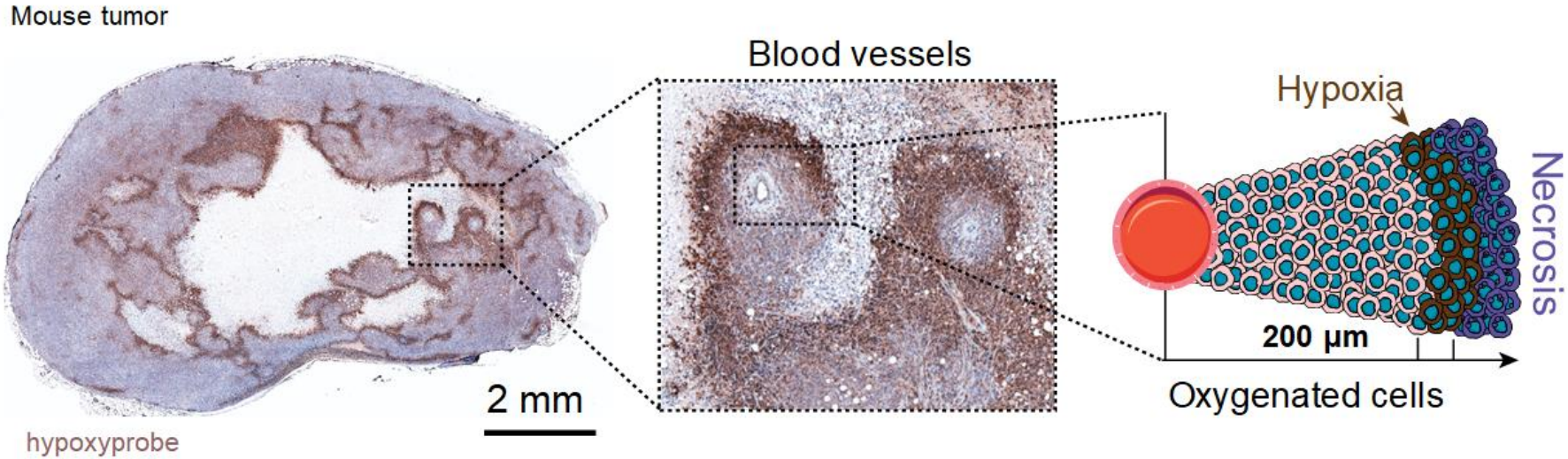
1

Example 4:

Hypoxia-driven breast cancer invasion

Rocha et al., *iScience* (2021, accepted)
Preprint: https://doi.org/10.1101/2020.12.30.424757

Intratumoral hypoxia

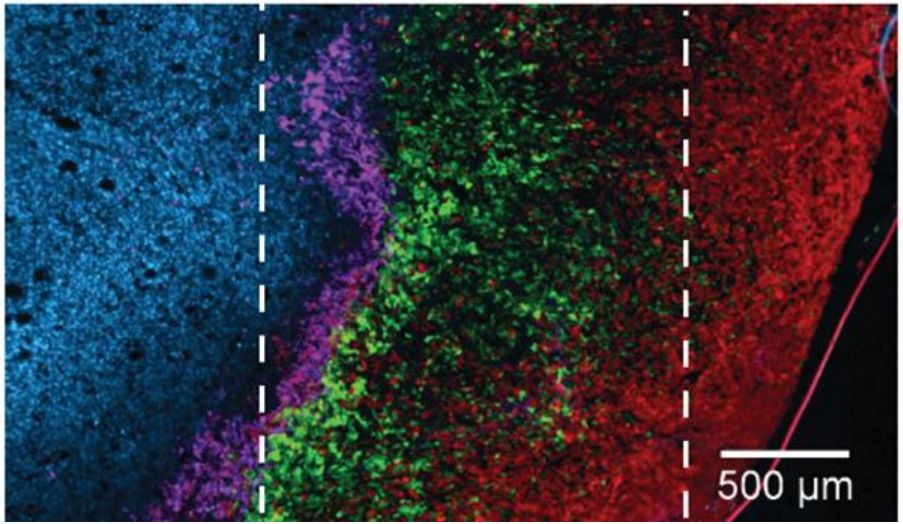


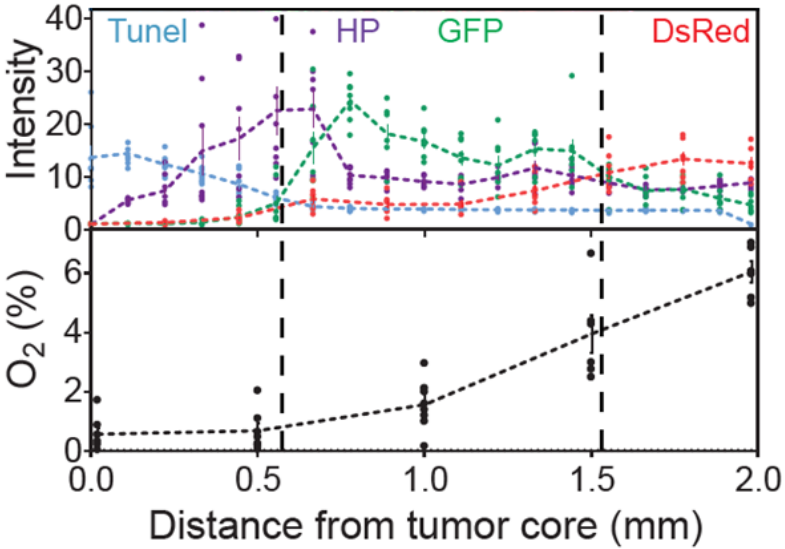
Questions

What are the **rules** of hypoxic cell motility?

How persistent is their response to hypoxic stress?

Hypoxic and post-hypoxic cells



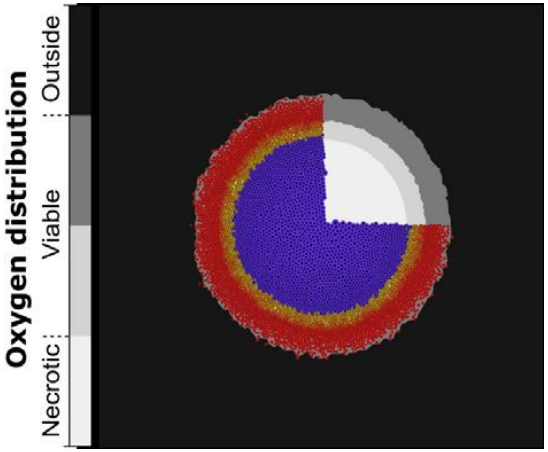


Hypoxia computational model

- We used Physicell to develop a mathematical model representative of tumor progression that incorporates the **cell phenotypes**, **location**, and exposure to **oxygen concentrations**.

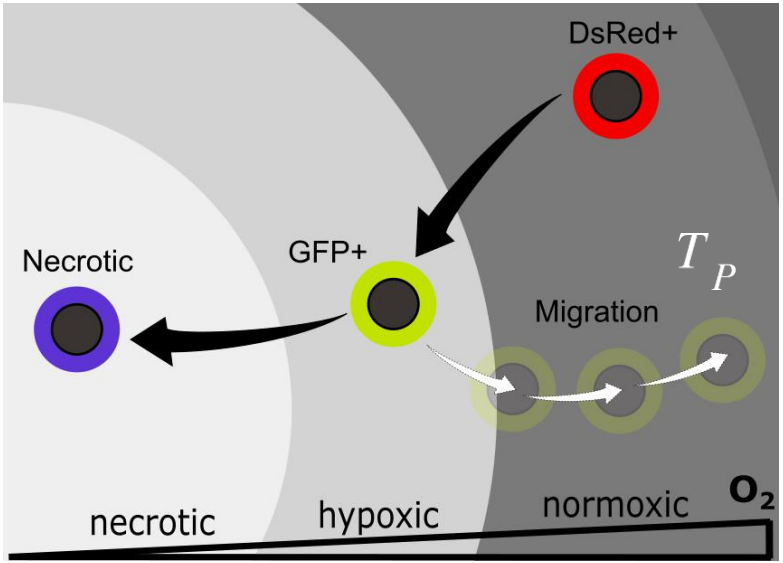
- Phenotypic states of the cancer cells based on the oxygen distribution: **normoxic**, **hypoxic**, and **necrotic**.

- The oxygen concentration (σ) is distributed in the environment using the standard transport equations from BioFVM and PhysiCell.



Phenotypic transitions

Eventually, cells may undergo phenotypic transitions due to their **motility** or **changes in the microenvironment**.



Simple ODE model for protein expression based on the "genes"

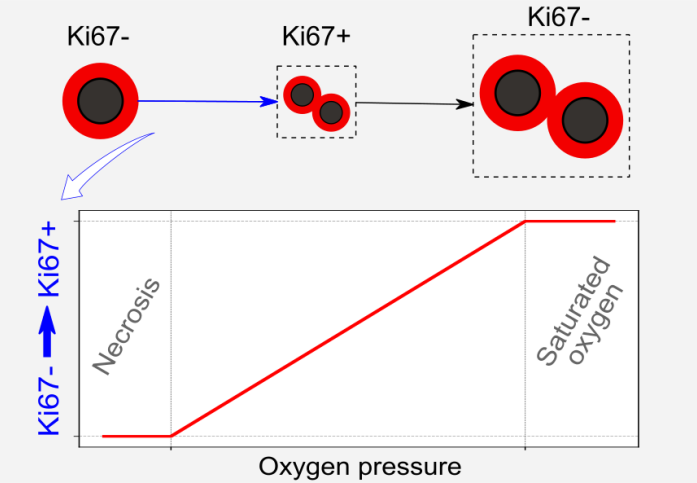
$$G = \left(G_0 , G_1 \right)$$

$$\frac{d[DsRed]}{dt} = G_0 \alpha_0 (1 - [DsRed]) + \beta_0 (G_0 - [DsRed])$$

$$\frac{d[GFP]}{dt} = G_1 \alpha_1 (1 - [GFP]) + \beta_1 (G_1 - [GFP])$$

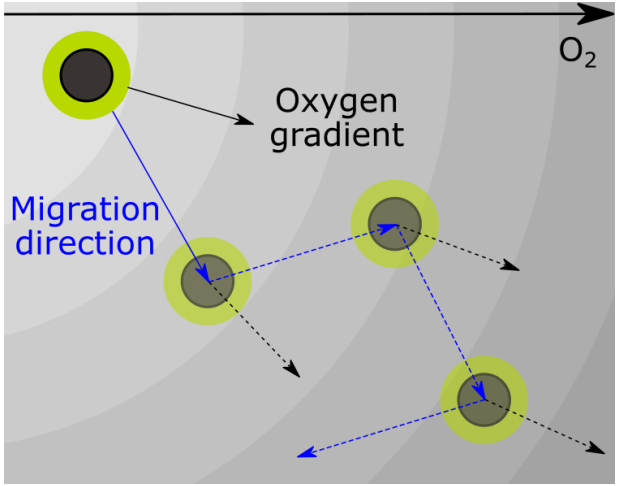
Proliferation and migration

Cell proliferation



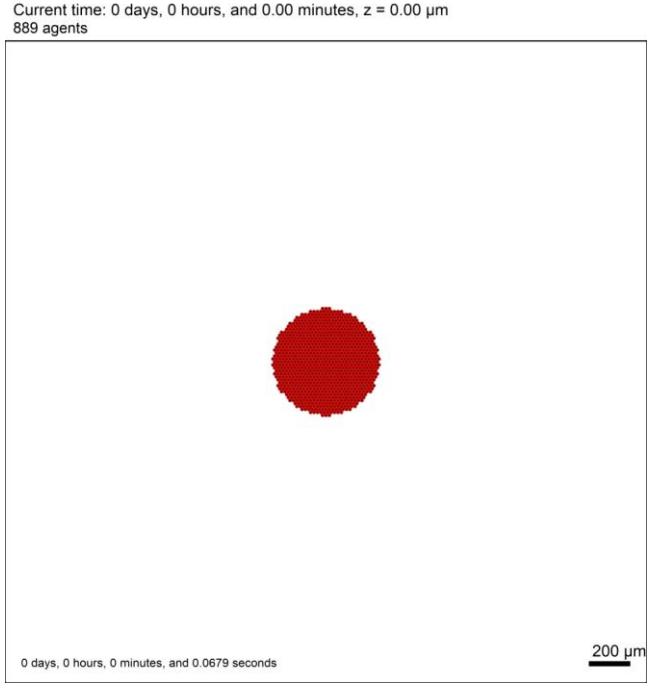
Basic Ki-67 model

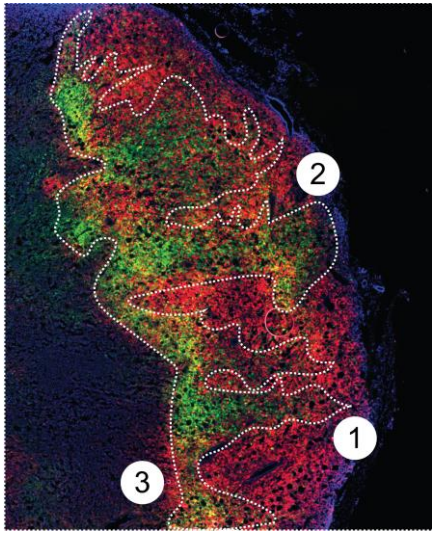
Cell migration

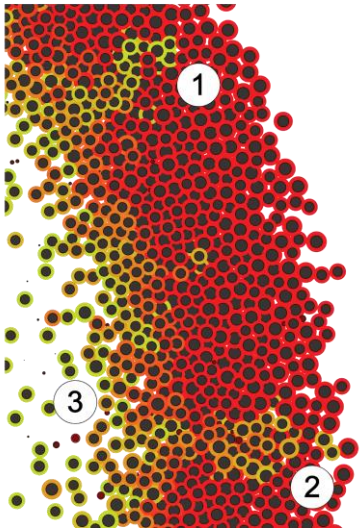


- **Speed (s)**
- **Bias (b)**
- **Persistence time (t)**

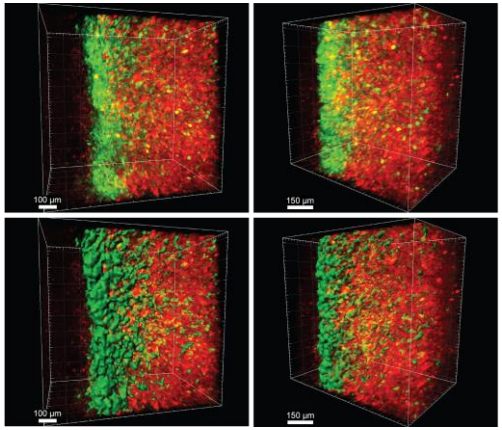
Mathematical model explains biological observations

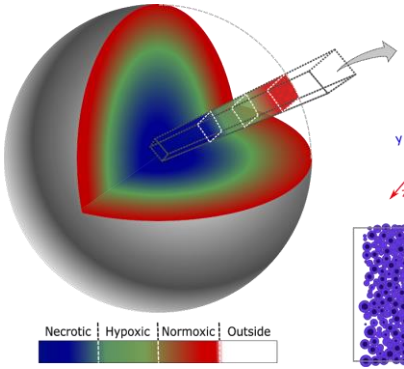


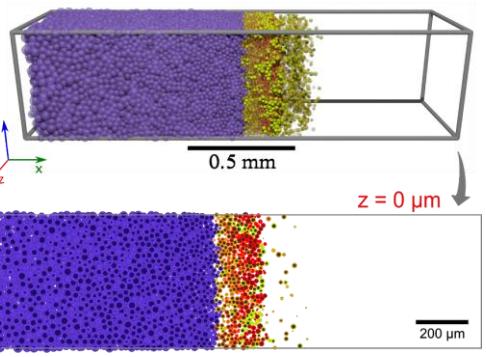


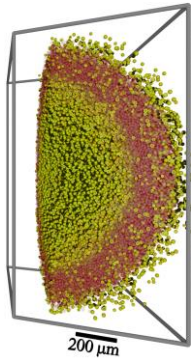


It also works in 3D

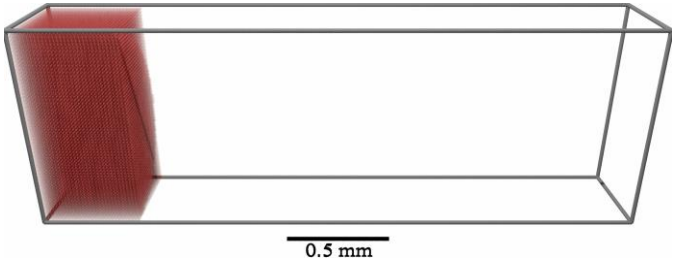








Try this model yourself!
nanohub.org/tools/pc4tumorhypoxia



Break ?

Goals

- Learn how to work with sample projects
  - Structure of the code
  - Briefly touching cell agents
  - Get a list of sample projects
  - Populate a project
  - Look at typical project structure
  - Modify settings
  - Compile and run a populated project
  - See typical model outputs
  - Clear out data and reset

Cells (1)

- Cells are the key entity in PhysiCell.

- Each cell keeps track of:
  - Type
  - ID
  - Position and velocity
  - State
  - Phenotype
    - Intracellular model and data are included here.
  - Custom data

Cells (2)

- Cells have built-in techniques for:
  - Division
  - Death
  - Changing type
  - Accessing / sampling the microenvironment
  - Finding nearby cells
  - Ingesting cells
  - And more behaviors via phenotype (Sessions 2)

Key cell information

- Each cell agent is a member of the `Cell` class.
- Some key data:
  - `std::string type_name` // human-readable name of cell type
  - `int type` // machine-readable unique integer identifier for cell type
  - `int ID` // cell agent's unique integer identifier. (different for each cell)
  - `std::vector<double> position` // the cell's current position (**never write this!**)
  - `std::vector<double> velocity` // the cell's current velocity
  - `cell_state` // things like size, pressure, and cells in contact
  - `phenotype` // behavioral properties / state (Sessions 2)
  - `custom_data` // custom scalar and vector data (Session 3)
  - `functions` // list of key cell functions (Session 3)

Future refinement:

Each cell should have a pointer to its Cell_Definition

Cell state

- Each **Cell** has an instance of **Cell_State** called **state:**
  - std::vector<Cell*> attached_cells:
    Use this for your own custom storage of interacting cells for contact-based interactions
  - std::vector<Cell*> neighbors:
    - a vector of pointers to all (mechanically interacting) neighbor cells.
    - Automatically updated to include all cells with non-zero mechanical interactions
  - std::vector<double> orientation:
    - Used during cell division: division plane is perpendicular to orientation.
    - A unit vector (length 1) directed from the cell's base to its apex
    - Cell division places daughter cells (randomly) perpendicular to this vector
    - In 2D, orientation = [0,0,1] so that daughter cells stay in plane
  - double simple_pressure:
    - a (normalized) measure of forces exerted by nearby adhered cells
    - in a 3-D, fully confluent (packed) tissue, 12 neighbors, and simple_pressure = 1
    - in a 2-D, fully confluent (packed) tissue, 6 neighbors, and simple_pressure = 0.5

Cell phenotype

- One of the most critical data elements in a PhysiCell Cell is ***phenotype***
- Hierarchically organize key behavioral elements:
  - Phenotype (Sessions 2)
    - **cycle**: advancement through a cell cycle model
    - **death**: one or more types of cell death
    - **volume:** cell's volume regulation
    - **geometry:** cell's radius and surface area
    - **mechanics:** adhesion and resistance to deformation ("repulsion")
    - **motility:** active motion (other than "passive" mechanics)
    - **secretion:** both release and uptake of chemical substrates. Interfaces with BioFVM
    - **molecular:** a place to store internalized substrates
    - **intracellular:** a place for intracellular models (ODEs, FBAs, Boolean)

Phenotype-centric programming

- The core cell behaviors are implemented:
  - Cell cycling (with user-selectable models)
  - Cell death
  - Cell adhesion / repulsion
  - Cell motility
  - Cell secretion / uptake

- Modelers can focus on writing functions that control these behaviors.
- This is **phenotype-centric programming.**

Cell Definitions

- A **Cell Definition** is a convenient way to set the parameters and functions for a whole class of cells
  - Users can instantiate cells of a specific type using `create_cell( A_cell_defn )`
  - With no argument, new cells use the `cell_defaults` definition

- Best practice: set up the `cell_defaults` definition first. Copy this to create other cell types

- Tip: Refer back to the phenotype in your agent's cell definition as a reference parameter set (i.e., to get the initial parameter values)

Sample projects

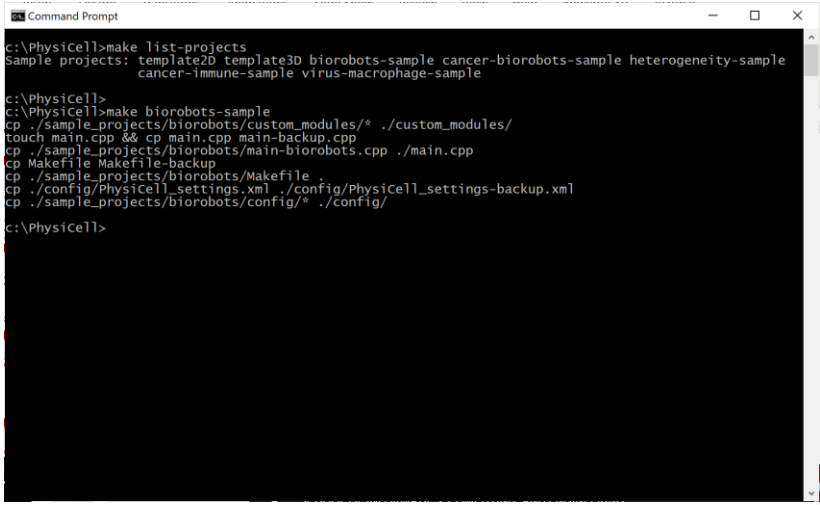
- It's inefficient (and a little insane) to code new projects *entirely* from scratch.
- So, we provide sample projects:
  - Template project
  - Cancer models
  - Synthetic multicellular systems
  - Viral dynamics in tissue
  - and more …

- **make [project-name]:** populate a sample project (puts all the source files where they belong)
  - Then use **make** to compile it
- **make data-cleanup**: clean up the output data
- **make reset**: return to a "clean slate" (depopulate the project)
- **make list-projects:** display all available sample projects

Documentation: User Guide Sections 6, 7.

PhysiCell Project Essentials (1)

- Each PhysiCell release includes sample projects. To list them:
  - **`make list-projects`**

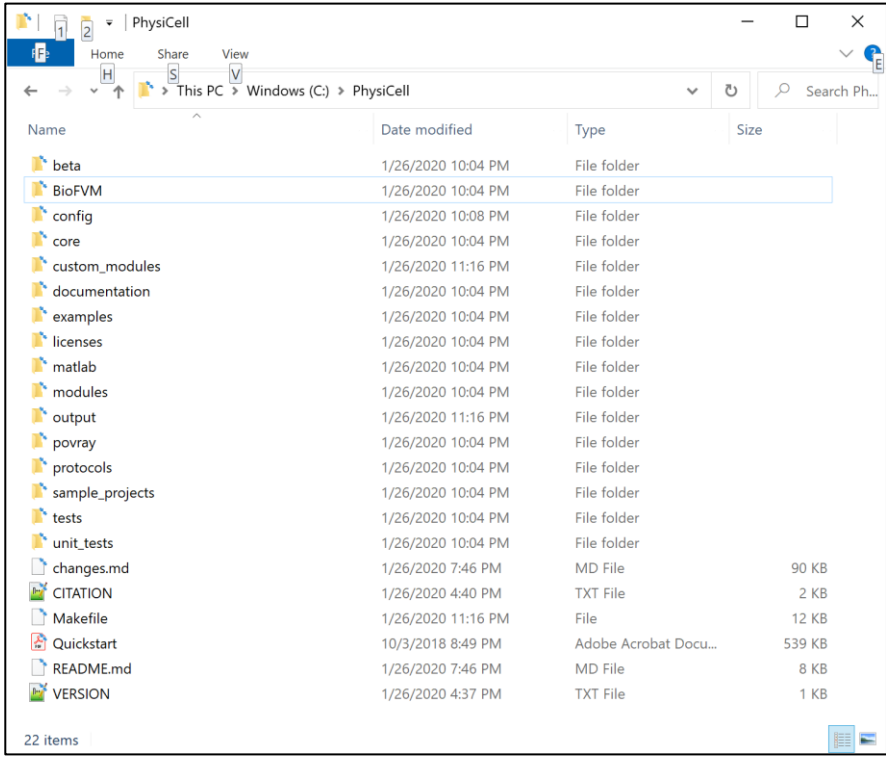


- Your first step is to **populate a project.**
  - **`make <project_name>`**
  - Let's use biorobots-sample:
    - **`make biorobots-sample`**
  - This copies source code, a tailored make file, and configuration files

Project directory structure

- (key) directories:
  - **./ (root):** main source, Makefile, and executable go here
  - **./beta**: for beta-testing (don't use)
  - **./BioFVM:** diffusion solver
  - **./config:** configuration files
  - **./core:** PhysiCell core functions
  - **./custom_modules:** put custom code for your project here.
  - **./documentation:** user guide, etc.
  - **./examples:** deprecated
  - **./licenses:** yep
  - **./matlab:** scripts and functions to load data in matlab
  - **./modules:** standard add-ons for PhysiCell
  - **./addons:** officially supported addons like PhysiBoSS and libRoadrunner
  - **./output:** where data are stored (by default, but can be changed)
  - **./povray:** deprecated
  - **./protocols:** instructions mostly for maintainers (e.g., release protocols)
  - **./sample_projects:** where we add sample projects
  - **./tests:** for automated testing (WIP)
  - **./unit_tests:** for automated testing (WIP)

Most of your work will be in the red directories



Project structure: config files

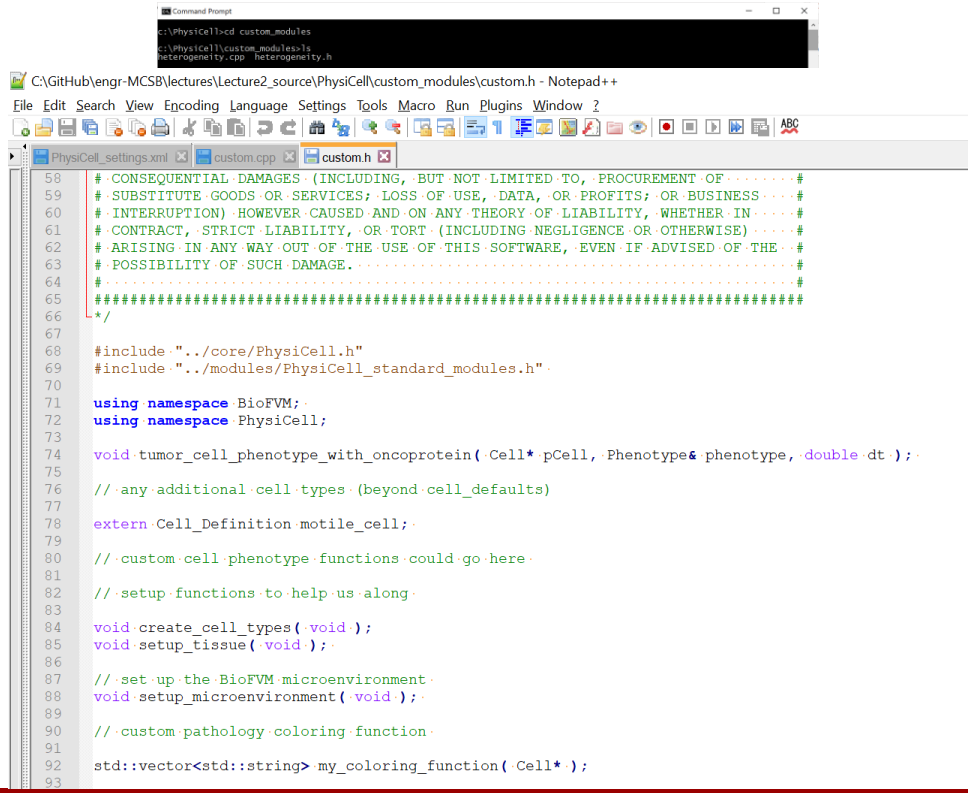
- Configuration files (XML)
  - **domain:** domain size and resolution
  - **overall:** general options
    - Final simulation time
    - Time step sizes
  - **parallel:** parallelization options
    - Number of threads
  - **save:** save options
    - Save where?
    - Save SVGs? (how often?)
    - Save full data? (how often?)
    - Save legacy data (don't)
  - **microenvironment_setup:** diffusion settings
    - more later
  - **cell_definitions:** define different cell types and starting parameters
    - more later
  - **user_parameters:** simulation-specific settings
    - more later

PhysiCell_settings.xml

```
<PhysiCell_settings version="devel-version">
    <domain>
        <x_min>-750</x_min>
        <x_max>750</x_max>
        <y_min>-750</y_min>
        <y_max>750</y_max>
        <z_min>-750</z_min>
        <z_max>750</z_max>
        <dx>20</dx>
        <dy>20</dy>
        <dz>20</dz>
        <use_2D>false</use_2D>
    </domain>

    <overall>
        <max_time units="min">30240</max_time> <!-- 21 days * 24 h * 60 min -->
        <time_units>min</time_units>
        <space_units>micron</space_units>
        <dt_diffusion units="min">0.01</dt_diffusion>
        <dt_mechanics units="min">0.1</dt_mechanics>
        <dt_phenotype units="min">6</dt_phenotype>
    </overall>

    <parallel>
        <omp_num_threads>8</omp_num_threads>
    </parallel>

    <save>
        <folder>output</folder> <!-- use . for root -->

        <full_data>
            <interval units="min">360</interval>
            <enable>true</enable>
        </full_data>

        <SVG>
            <interval units="min">60</interval>
            <enable>true</enable>
        </SVG>

        <legacy_data>
            <enable>false</enable>
        </legacy_data>
    </save>

    <microenvironment_setup>
    <user_parameters>
```

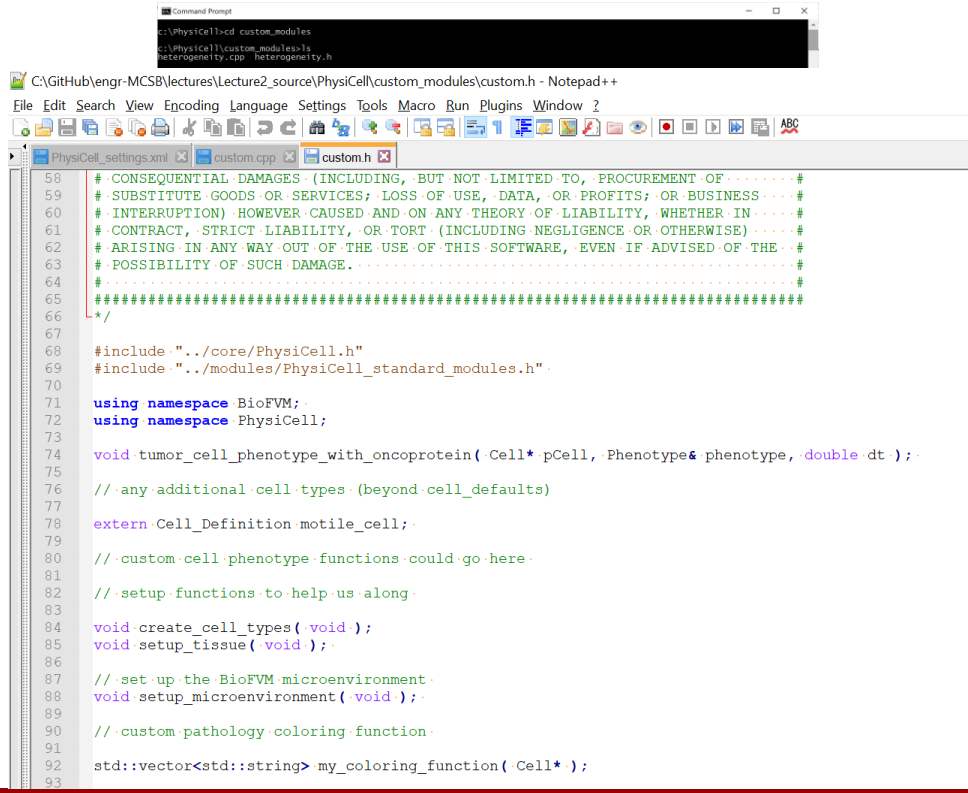
Project structure: custom modules

- Custom Modules
  - Setup functions
  - Cell definitions
  - Custom functions
  - any other modeling
  - Custom coloring functions



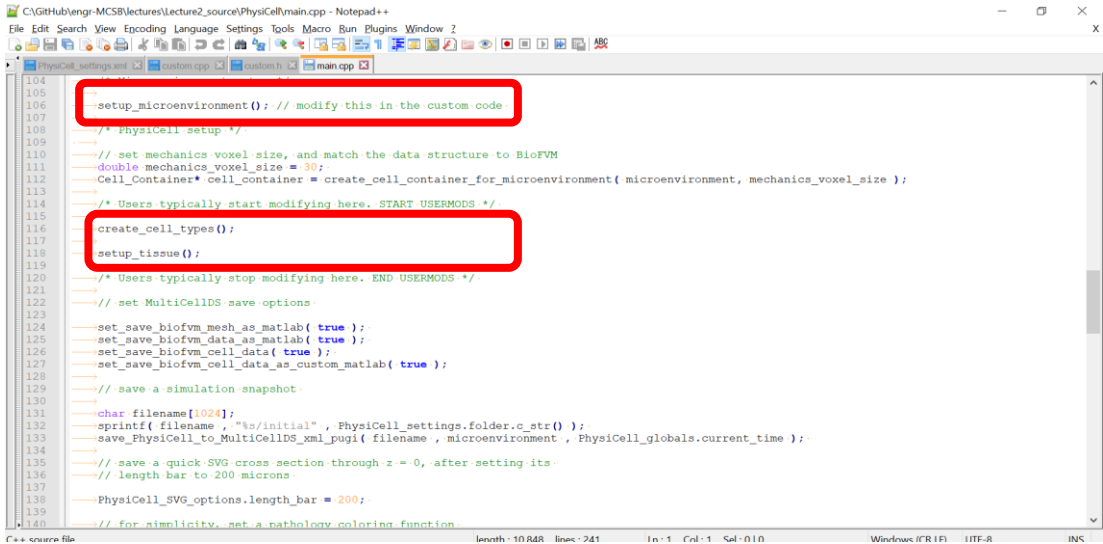
Project structure: custom modules

- Custom Modules
  - Any user-defined globals (at top)
    - ♦ Declared cell types
  - Setup functions
    - ♦ **`create_cell_types()`**
      - » Do all setup on all cell types
        - ○ Adjust phenotype
        - ○ Add / adjust custom data
        - ○ Set functions
    - ♦ **`setup_tissue()`**
      - » Place initial cells in microenvironment
      - » Modify each cell as needed
  - Custom functions
  - any other modeling
  - Custom coloring functions



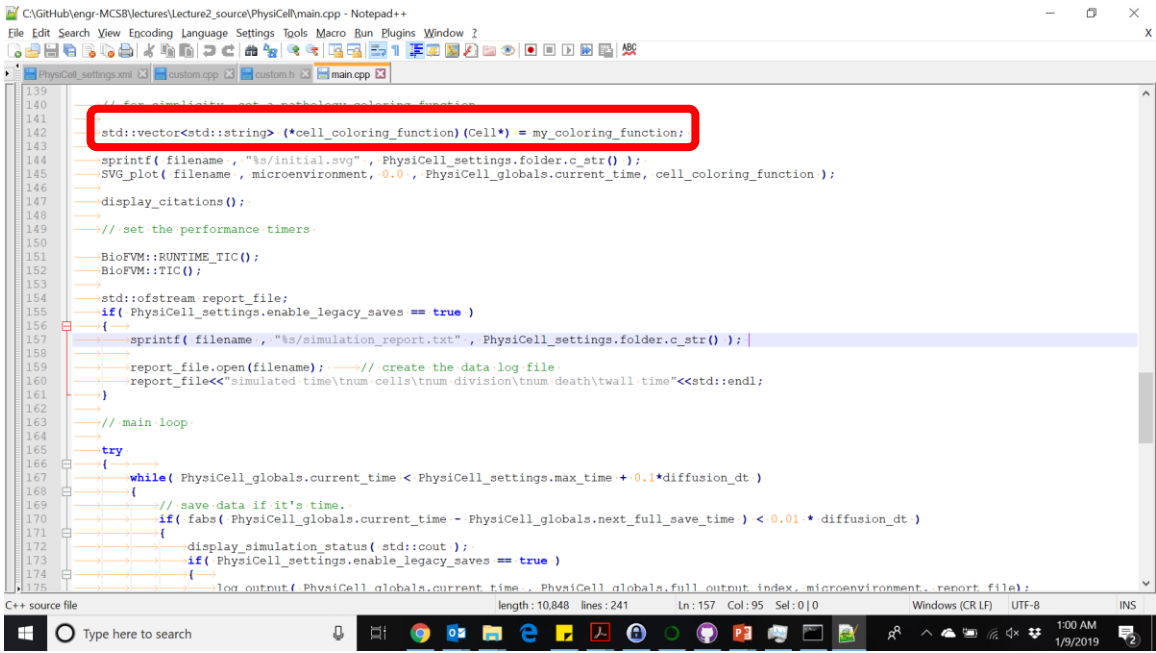
Project structure: main.cpp

- `main.cpp`
  - (in the root directory)
  - calls the setup functions



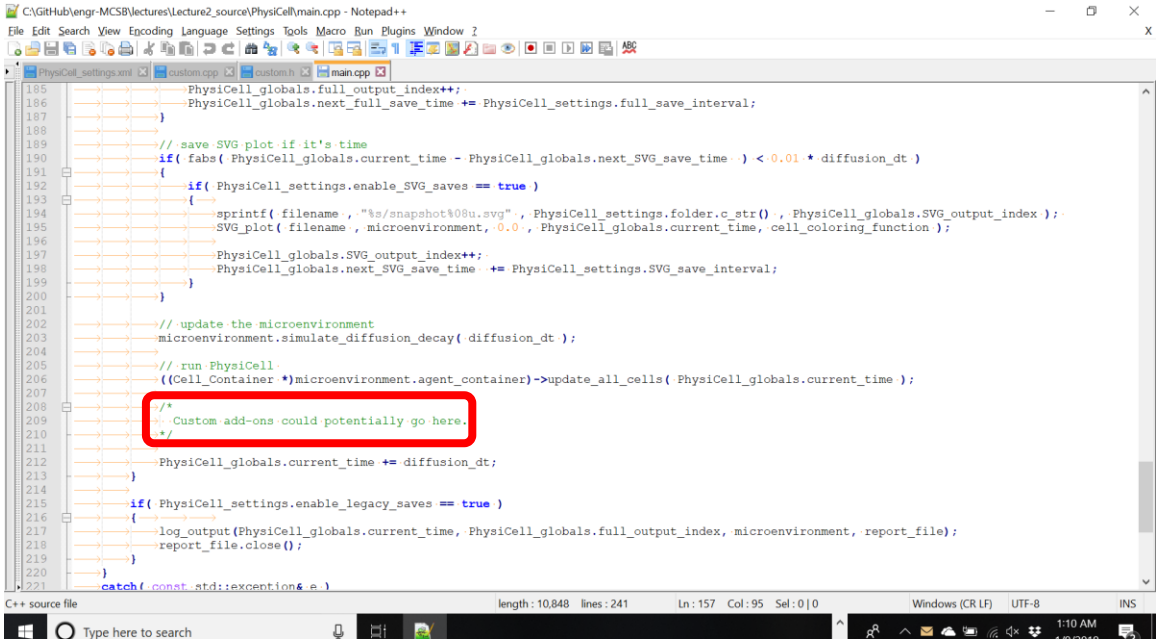
Project structure: main.cpp (continued)

- `main.cpp`
  - set coloring function



Project structure: main.cpp (continued)

- `main.cpp`
  - insert custom routines
  - **This would be a good place to put extensions.**



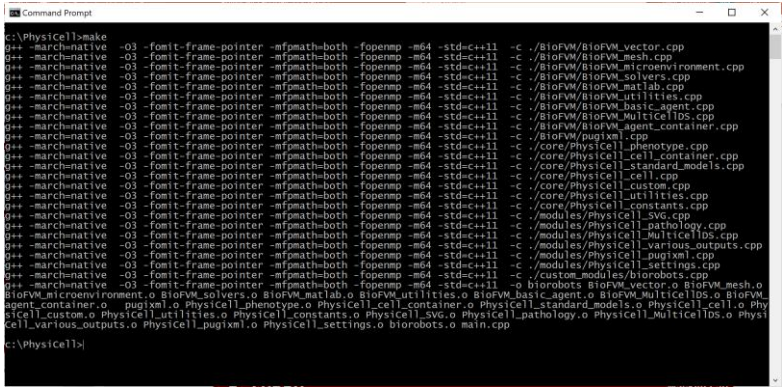
Now, let's get back to working with sample projects.

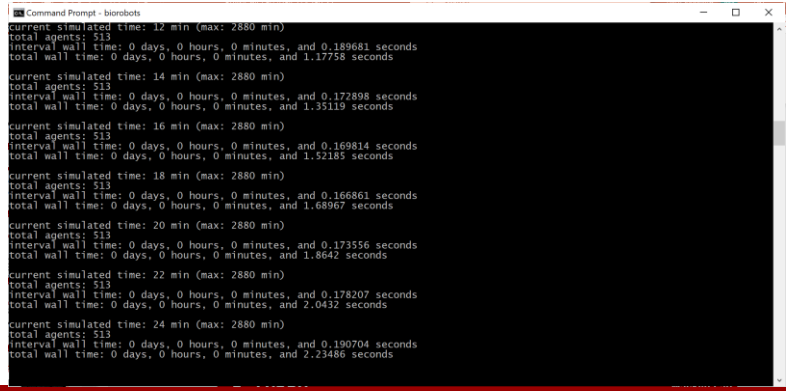
PhysiCell Project Essentials (2)

- Now, **compile the project**
  - `make`

- Then, **run the project**
  - `./biorobots` (Linux, MacOS)
  - `biorobots.exe` (Windows)

- This should take about 5 minutes



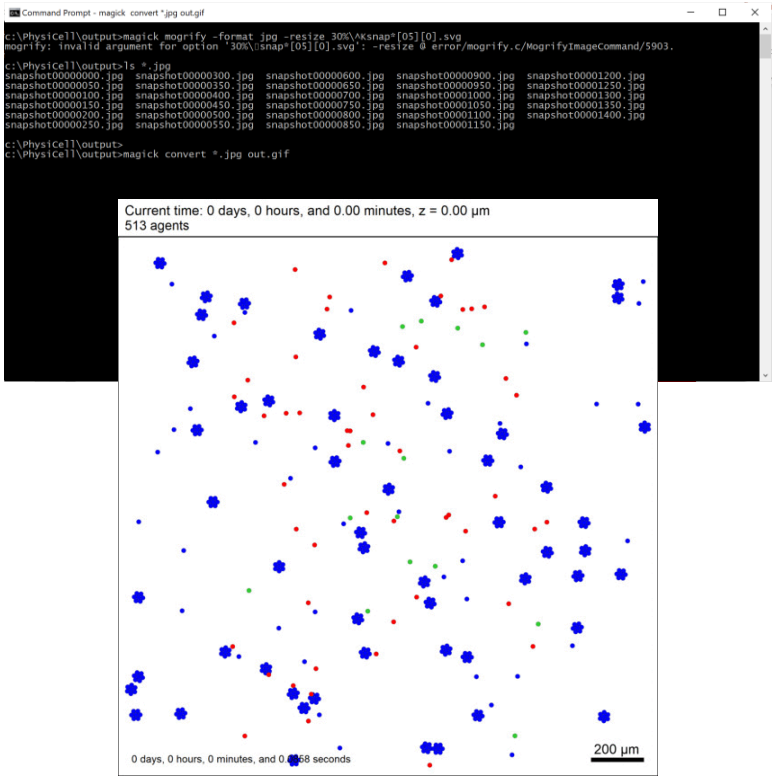


PhysiCell Project Essentials (3)

- **Look at saved data**
  - Most projects save data to ./output
    - XML files give metadata, mesh, and substrate info
    - MAT file save (compressed) substrate and cell data
    - SVG files are visual quick snapshots
    - More on loading XML / MAT files in Python later

- Let's convert SVG to rescaled JPEG
  - **`magick mogrify -format jpg -resize 30% snap*.svg`**
    - Convert snapshot00000000.svg, snapshot00000001.svg, ...
  - **`magick mogrify -format jpg -resize 30% snap*[05][0].svg`**
    - Convert snapshot00000000.svg, snapshot00000050.svg, ...

- Now, let's create an animated GIF
  - **`magick convert *.jpg out.gif`**



Working with the images

- To convert all the SVG files to PNG format

```
magick mogrify -format png snap*.svg
```

- To convert every SVG file ending in 0 or 5 to JPG format

```
magick mogrify -format jpg snap*[05].svg
```

- To convert the JPG files to an animated GIF

```
magick convert *.jpg out.gif
```

- To create an mp4 movie:

```
ffmpeg -r 24 -f image2 -i snapshot%08d.jpg -vcodec libx264 -pix_fmt yuv420p -strict -2 -tune animation -crf 15 -acodec aac out.mp4
```

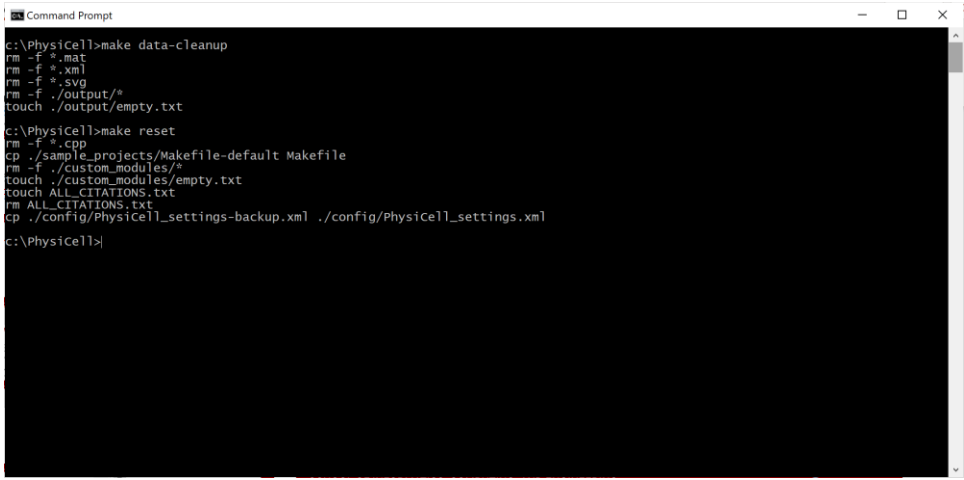
Handy tricks!

Use **make jpeg** to create a full set of JPGs

Use **make movie** easily create the mp4.

PhysiCell Project Essentials (4)

- **Data cleanup**
  - Clean up data to get ready for another run
  - `make data-cleanup`

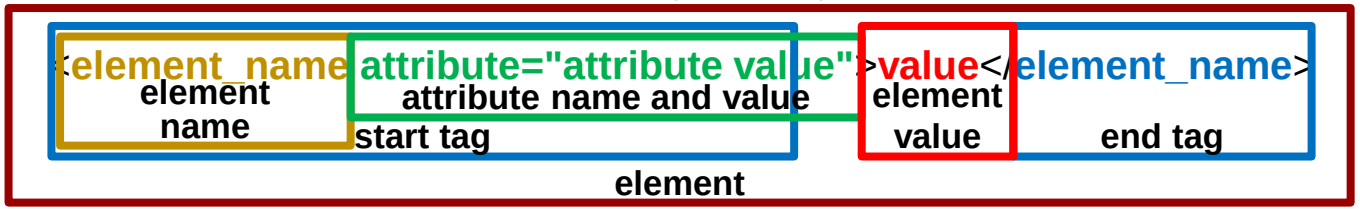


- **Reset** to a clean slate
  - De-populate the project
  - Get ready for another project
  - `make reset`

Changing settings in a project

XML Refresher (1)

- XML stands for e**X**tensible **M**arkup **L**anguage
  - (Think of it as a generalization of HTML.)
- Information in XML are stored in elements. Key elements are:
  - element name in a start tag
  - attributes and values
  - element value
- If an element has a value, it must have a matching end tag:



- If an element has attributes but no value, you can use a more compact form:

```
<element_name attribute1="attribute 1 value" attribute2="attribute 2 value" />
```

XML Refresher (2)

- Just like HTML, XML can have sub-elements:

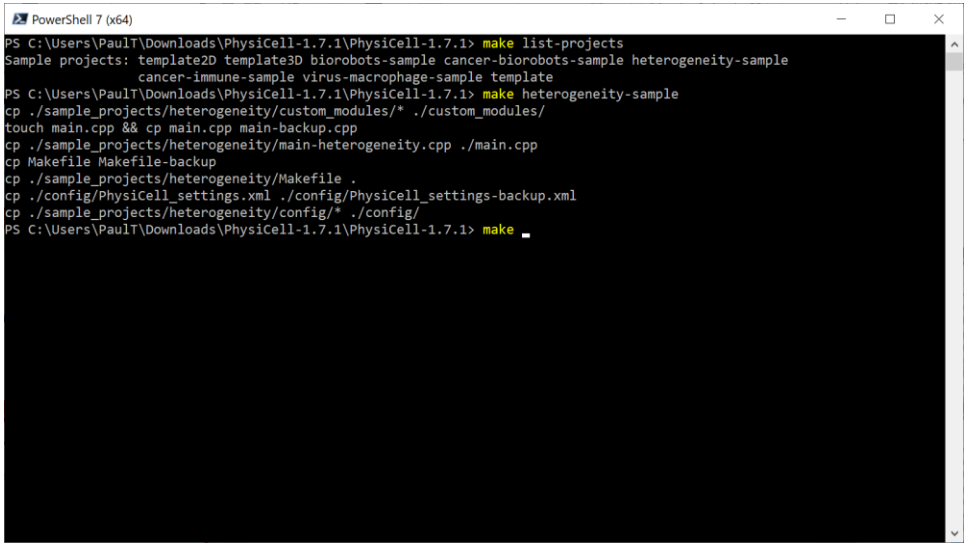
```
<element_name attribute1="attribute 1 value" attribute2="attribute 2 value">
        <subelement_name attribute="attribute value">subvalue1</subelement_name >
        <subelement_name attribute="attribute value">subvalue2</subelement_name >
</element_name >
```

- By convention:
  - the name of the element is a parameter name
  - the element's value is the parameter value
  - attributes are used to store metadata or other clarifications (e.g., units)

```
<diffusion_coefficient units="micron/min^2">1000</diffusion_coefficient>
```

First, populate the cancer heterogeneity project

- List all available sample projects
- Populate the cancer heterogeneity project
- Build the project
- Change some settings (next slide)



How to change settings in XML

- Open config/PhysiCell_settings.xml
- Major sections:
  - **domain** -- how big of a region to simulate
  - **overall** -- how long to simulate, time step sizes
  - **parallel** -- OpenMP settings
  - **save** -- how often to save SVG images and full data
  - **microenvironment** -- settings on diffusing substrates
  - **user_parameters** -- model-specific settings
  - **cell_definitions** -- set baseline cell properties

Edit settings

- Open the settings file:
  - **./config/PhysiCell_settings.xml**

- Let's change:
  - Change domain to [-400,400] × [-400,400]
  - Reduce max simulation time to 2160 minutes
  - Save full data ever 360 minutes
  - Set oncoprotein standard deviation to 3 (increase heterogeneity)
  - Set the max oncoprotein value to 10 (mean + 3 standard deviations)

Edit settings: XML

- Open ./config/PhysiCell_settings.xml
- Major sections:
  - **domain** -- how big of a region to simulate
  - **overall** -- how long to simulate, time step sizes
  - **parallel** -- OpenMP settings
  - **save** -- how often to save SVG images and full data
  - **microenvironment** -- settings on diffusing substrates
  - **user_parameters** -- model-specific settings
  - **cell_definitions** -- set baseline cell properties

Edit settings: Domain size

- Open `./config/PhysiCell-settings.xml`
- Let's set the domain size in the **domain** block
  - Switch to [-500,500] x [-500,500] x [-10,10] to speed it up

```
<PhysiCell_settings version="devel-version">
        <domain>
                <x_min>-400</x_min>
                <x_max>400</x_max>
                <y_min>-400</y_min>
                <y_max>400</y_max>
                <z_min>-10</z_min>
                <z_max>10</z_max>
                <dx>20</dx>
                <dy>20</dy>
                <dz>20</dz>
                <use_2D>true</use_2D>
        </domain>
```

Edit settings: User parameters

- Let's also look at the **user_parameters** block
  - Let's change the oncoprotein standard deviation (`oncoprotein_sd`) to 3 (more variation)
  - Let's change the max oncoprotein (`oncoprotein_max`) to mean + 3 sds = 1 + 9 = 10

```
<user_parameters>
  <tumor_radius type="double" units="micron">250.0</tumor_radius>
  <oncoprotein_mean type="double" units="dimensionless">
    1.0</oncoprotein_mean>
  <oncoprotein_sd type="double" units="dimensionless">3.0</oncoprotein_sd>
  <oncoprotein_min type="double" units="dimensionless">0.0</oncoprotein_min>
  <oncoprotein_max type="double" units="dimensionless">10</oncoprotein_max>
  <random_seed type="int" units="dimensionless">0</random_seed>
</user_parameters>
```

Edit settings: Save settings

- Let's look at the **overall** block
  - Set max time to 1.5 days = 1.5 x 24 x 60 = 2160 minutes

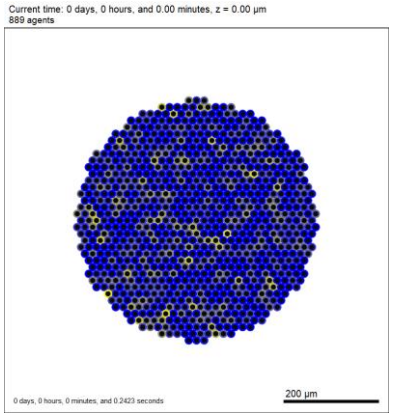
```
<overall>
        <max_time units="min">2160</max_time> <!-- 36 h * 60 min -->
        <time_units>min</time_units>
        <space_units>micron</space_units>
```

- Let's look at the **save** block
  - Set the full save interval to 6 hours = 360 minutes

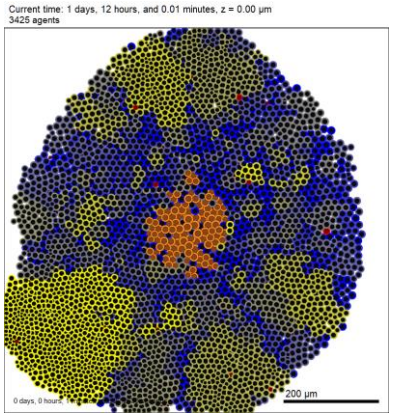
```
<save>
        <folder>output</folder> <!-- use . for root -->
        <full_data>
                <interval units="min">360</interval>
                <enable>true</enable>
        </full_data>
```

Run and View Results: Snapshots

- run:
  - **./heterogeneity** (MacOS or Linux)
  - **heterogeneity.exe** (Windows)



initial.svg



final.svg

- Look in output:
  - Look at snapshot SVG files
  - Look at **legend.svg**
    - (Not much to see on this example)

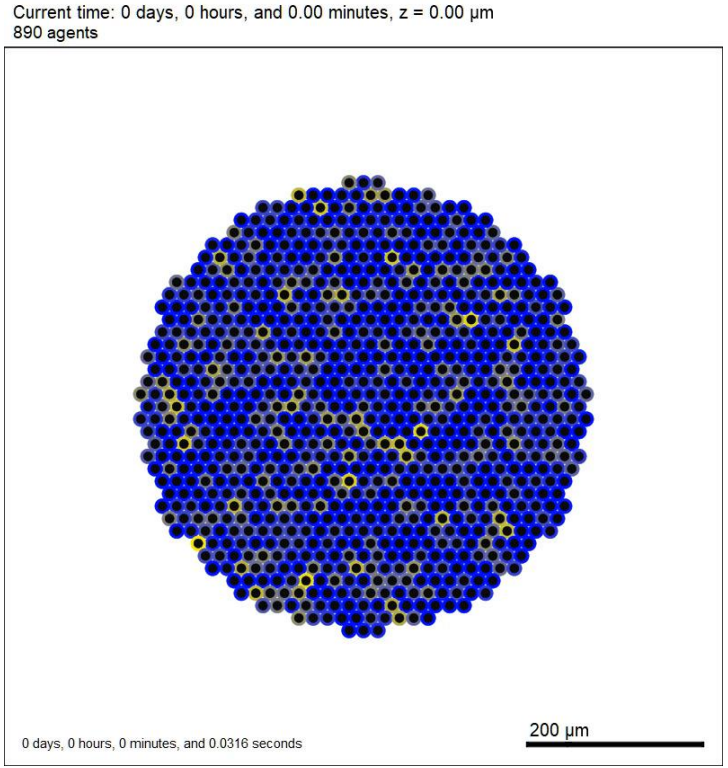
legend.svg

- Convert snapshots to JPEG:
  - **`make jpeg`** (results: output/snapshot00000000.jpg ...)

View results: GIF and movie

- Make and animated GIF:
  - **`make gif`** (result: output/out.gif)

- Make an mp4 movie
  - **`make movie`** (result: output/out.mp4)



Acknowledgment

Funding

- National Cancer Institute (U01CA232137)
- National Science Foundation (1720625)
- Jayne Koskinas Ted Giovanis Foundation for Health and Policy
- Breast Cancer Research Foundation

MathCancer Lab

Paul Macklin

Randy Heiland

Heber Rocha

Michael Getz

John Metzcar

Aneequa Sundus

Yafei Wang

Kali Konstantinopoulos