Proliferating Cell Collectives: A Comparison of Hard and Soft Collision Models

Manuel Lerchner
Technical University of Munich
Munich, Germany

***Abstract*—This work extends the hard (constraint-based) collision model introduced by Weady et al. [1] by systematically comparing it with a soft (potential-based) approach for simulating proliferating cell collectives within a unified computational framework. Both models successfully reproduce experimentally observed concentric ring patterns and microdomain formation in small bacterial colonies, indicating that these macroscopic patterns are robust to the specific details of collision resolution.**

At the microscopic scale, the models differ significantly. The hard model enforces strict non-overlap, maintaining realistic packing fractions around 0.9 and producing accurate stress distributions and microdomain structures. The soft model allows substantial, unphysical overlap, with packing fractions exceeding 5 in colony centers, resulting in distorted microdomains and elongated, unrealistic cell bundles. These artifacts limit its applicability for detailed analyses of cell-scale organization or mechanical stresses.

Performance benchmarks show the hard model consistently outperforms the soft model, achieving up to 9.36 times faster runtimes on 112 cores due to timesteps that can be chosen roughly 30 times larger while maintaining numerical stability, along with better parallel scaling. Adaptive timestepping based on the Courant-Friedrichs-Lewy condition proves essential, for both models, to efficiently handle the rapidly changing configurations during colony growth.

Overall, the hard model is strongly preferred for nearly all applications, while the soft model is limited to exploratory studies of very small colonies focused on macroscopic pattern formation rather than cell-scale details.

***Index Terms*—collision models, constraint-based simulation, potential-based repulsion, bacterial colony growth, proliferating cell collectives, adaptive timestepping, pattern formation, computational performance, stress-sensitive growth, microdomain formation, non-overlapping particle simulations, large-scale colony simulation, parallel computing, computational biology**

I. Introduction

A. Biological Motivation

The collective behaviors of biological entities, from microbial colonies to multicellular tissues, exemplify how local interactions among individual organisms can generate complex, large-scale structures and dynamics. These systems often exhibit emergent spatio-temporal patterns that arise from the interplay of cell growth, division, mechanical interactions, and local environmental feedback.

A particularly compelling example is found in bacterial colonies, where cell arrangement reflects growth dynamics and internal stresses. Continuous cell proliferation within colonies generates significant mechanical forces that accumulate in the densely packed interior [2], leading to compressive stresses that suppress cell growth at the colony center. As a result, peripheral cells expand outward over time while internal cells remain densely packed and growth-limited. This outward expansion over time produces the characteristic concentric ring patterns observed in bacterial species such as *E. coli*, *Bacillus subtilis*, and *Proteus mirabilis*, as well as fungi like *Setosphaeria rostrata* and *Exserohilum turcicum*, as illustrated in Figure 1 [3].

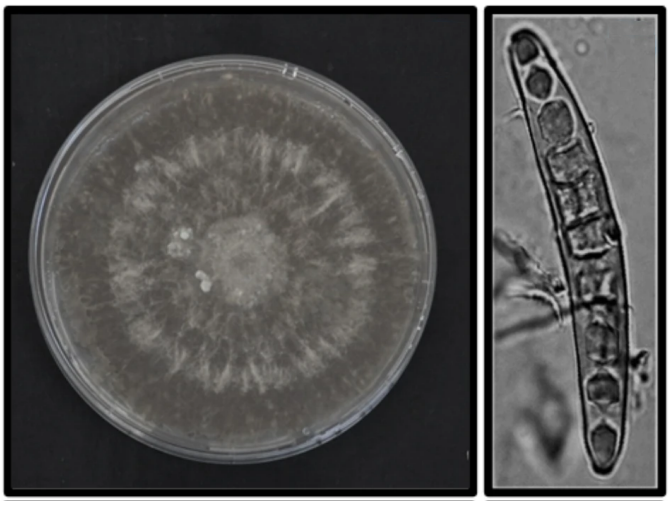


Figure 1: Morphology of the fungus *Exserohilum turcicum*. The left image shows a typical colony with concentric ring patterns, while the right image displays the elongated, rod-shaped morphology of individual fungal cells. Source: [4].

Computational modeling offers a powerful tool for dissecting the complex dynamics and emergent spatial patterns observed in those colonies. To reproduce such emergent structures and uncover their underlying mechanisms, models must accurately capture the dynamic interplay between cell growth and division, mechanical forces, and intercellular interactions.

Simulating dense, proliferating cell collectives, however, poses significant computational challenges: the large number of cells in realistic colonies, combined with the need to resolve collisions accurately across continuously changing configurations, creates a computational bottleneck. Moreover, different collision-handling approaches, such as rigid constraints versus soft repulsions, offer distinct trade-offs in physical fidelity, numerical stability, and computational cost. This work is directly motivated by these considerations and builds upon

the pioneering computational framework developed by Weady et al. [1], who demonstrated how to model bacterial colony growth while capturing stress-dependent feedback and pattern formation. We extend their work by systematically comparing two fundamentally different approaches to collision handling, examining their respective strengths and limitations in simulating large systems of proliferating cells.

II. Related Work

A. Collision Modeling Paradigms

In computational biology, agent-based models (ABMs) represent individual cells as discrete entities that grow, divide, and interact mechanically. Unlike continuum models, which treat populations as continuous fields, ABMs simulate single-cell behavior and local heterogeneity. A central computational challenge in such models is resolving cell-cell collisions efficiently. Two primary paradigms exist for handling collisions: soft (potential-based) collision models and hard (constraint-based) collision models.

Soft (Potential-Based) Models

Soft models manage cell interactions through repulsive forces defined by potentials such as Hertzian or Lennard-Jones functions. When cells overlap, these forces push them apart continuously. The primary advantage is computational efficiency: the calculations are local and embarrassingly parallelizable, enabling large-scale simulations. For instance, Warren et al. [5] demonstrated this by simulating millions of cells in three dimensions using a soft repulsion approach.

However, soft models suffer from two key limitations. First, the stiffness of repulsive potentials creates numerical instability, requiring very small timesteps to maintain accuracy and prevent divergence. Second, the finite range of repulsive forces means cells behave as elastically deformable objects rather than rigid bodies, effectively reducing their geometric diameter below the intended size [6]. This deformation accumulates and can distort simulation results, particularly in dense regions.

Hard (Constraint-Based) Models

Hard models enforce strict non-overlapping constraints, treating cells as geometrically rigid and impenetrable. Constraint-based methods are widely used in physics simulations [7]–[10], and have also been adapted for biological systems [1], [6], [11].

The main drawback is computational cost and added complexity. Enforcing global non-overlap constraints requires iterative solvers operating on large coupled systems of equations, which creates a significant bottleneck. Modern implementations, however, can efficiently reduce the numerical stiffness associated with soft models, enabling much larger timesteps while maintaining geometric fidelity [6]. Nonetheless, the fundamental trade-off remains: higher cost per timestep due to constraint solving, but fewer timesteps required overall.

B. The Benchmarking Gap

While soft and hard collision models entail distinct computational trade-offs, the question of which approach offers better performance remains unresolved. This trade-off becomes particularly critical when simulating large populations of independent agents, where the choice of collision model directly affects runtime, scalability, and the accuracy of the simulation.

To our knowledge, a direct, systematic comparison of these models is absent from the literature. Moreover, existing biological studies typically validate their chosen approach against experimental data, such as microscopy images, growth curves, and morphological features [1], [5], [11]–[19], prioritizing biological accuracy over computational performance. Few studies provide rigorous benchmarking of runtime, scalability, or numerical stability across different colony sizes. Consequently, modelers lack clear guidance on which paradigm is more efficient for a given problem scale.

This work aims to address this gap by implementing both soft and hard collision schemes within a shared computational framework, enabling direct performance comparisons. We systematically benchmark the two approaches across a range of colony sizes, evaluating their respective strengths and limitations in both biological fidelity and computational efficiency.

III. Cell Mechanics

Accurately modeling mechanical interactions between cells is essential for simulating colony growth and emergent spatial patterns. From a computational perspective, approximating bacteria and fungi as rigid spherocylinders (rods with hemispherical end caps) offers a balance between biological realism and computational efficiency. This representation captures the elongated morphology of these organisms while preserving the fundamental mechanics of cell-cell interactions and colony expansion. Although real cells exhibit some deformability, the rigid spherocylinder approximation successfully captures essential collective dynamics at significantly reduced computational cost and is well validated experimentally [11], [13], [15], [18], [19]. The spherocylinder geometry closely matches common organisms such as *E. coli* and *Setosphaeria rostrata* (see Figure 1), and has become standard in computational models of microbial collectives [1], [5], [12]–[14].

Each spherocylinder is characterized by a time-varying length ℓ that increases during growth prior to division, and a fixed diameter d that remains constant across the population. This geometry enables efficient contact detection algorithms [20] and naturally accommodates rotational dynamics essential for capturing realistic colony mechanics.

At bacterial length scales, the Reynolds number is extremely small ($\mathrm{Re} \ll 1$), so viscous forces dominate over inertia. In this highly viscous, *sticky* environment, cells stop almost immediately once forces cease [11], [21]. As a result, cell motion is governed by the overdamped Langevin equation:

$$\mathbf{u}_i = \frac{1}{\zeta \ell_i} \sum_{j \neq i} \mathbf{F}_{ij}, \quad \boldsymbol{\omega}_i = \frac{12}{\zeta \ell_i^3} \sum_{j \neq i} \mathbf{r}_{ij} \times \mathbf{F}_{ij} \tag{1}$$

Here, $\mathbf{u}_i$ and $\boldsymbol{\omega}_i$ are the translational and angular velocities of cell i, $\mathbf{F}_{ij}$ is the force exerted on cell i by cell j, ζ is the drag coefficient, and $\mathbf{r}_{ij}$ is the lever arm from the cell center to the contact point. Figure 2 illustrates the model and the forces and torques during collision.

A. Stress-Dependent Growth

Bacterial cells grow along their longitudinal axis according to the following model, adapted from [22]:

$$\dot{\ell}_i = \frac{\ell_i}{\tau} e^{-\lambda' \sigma_i} \tag{2}$$

where τ is a characteristic timescale, σ_i is the compressive stress along cell i's main axis, and λ' is a stress sensitivity parameter. As the compressive stress increases, the effective growth rate $\dot{\ell}_i$ decreases toward zero, leading to slower elongation. When $\sigma_i = 0$, the cell grows exponentially with a rate determined solely by τ.

The stress on cell i is computed by projecting contact forces onto the cell's longitudinal axis $\hat{\mathbf{t}}_i$:

$$\sigma_i = \sum_{j \neq i} \frac{1}{2} \left| \hat{\mathbf{t}}_i \cdot \mathbf{F}_{ij} \right| \tag{3}$$

Similar to [22], we nondimensionalize lengths by ℓ_0, time by τ, and stress by $\tau/(\zeta \ell_0^2)$, leaving the dimensionless growth sensitivity $\lambda = (\tau/\zeta \ell_0^2)\lambda'$ as the only free parameter. Therefore we set $\ell_0 = 1$, $d = 0.5$, $\tau = 1$, and $\zeta = 1$.

Cells divide at a critical length $\ell_{\text{crit}} = 2\ell_0$, producing two daughters with lengths sampled from $[0.98\ell_0, 1.02\ell_0]$ to avoid synchronized divisions [14].

B. Rotational Diffusion

To account for Brownian motion and other sources of random fluctuations, we apply a small rotational perturbation to each cell at every timestep. This introduces gradual rotational diffusion, preventing cells from forming artificially aligned configurations and promoting realistic spatial arrangements.

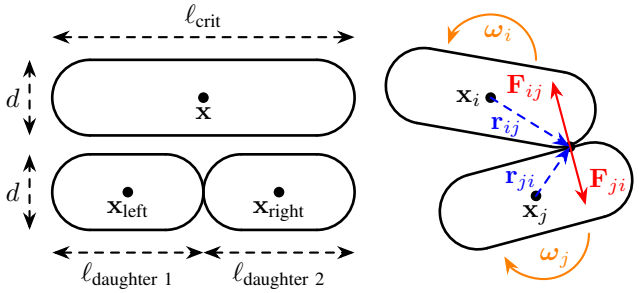


Figure 2: Spherocylinder cell model. Contact forces $\mathbf{F}_{ij}$ (red) and lever arms $\mathbf{r}_{ij}$ (dashed blue) generate torques causing angular motion $\boldsymbol{\omega}$ (orange), and also produce translational displacements of the cells. Cell division occurs when a cell reaches critical length ℓ_{crit}.

IV. Unified Computational Framework

A. Colony Representation

Following Weady et al. [22], the bacterial colony is represented by a global state vector $\boldsymbol{\mathcal{C}} = [\ldots, \mathbf{x}_n^\top, \mathbf{q}_n^\top, \ldots]^\top \in \mathbb{R}^{7N}$, where N is the number of cells. Each cell is represented by a 7-dimensional sub-vector consisting of a position vector $\mathbf{x}_n \in \mathbb{R}^3$ and a unit quaternion $\mathbf{q}_n \in \mathbb{R}^4$ representing its orientation. The lengths of all cells are stored separately in a vector $\boldsymbol{\ell} = [\ldots, \ell_n, \ldots]^\top \in \mathbb{R}^N$. Quaternions are used instead of Euler angles or rotation matrices because they avoid singularities and provide stable numerical integration over long simulations.

B. Colony Dynamics

The translational and angular velocities of all cells in the colony from Equation 1 are determined simultaneously by a force-velocity relationship $\boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{M}}^k \boldsymbol{\mathcal{F}}$. Here, $\boldsymbol{\mathcal{F}} \in \mathbb{R}^{6N}$ is the generalized force vector for the entire colony, containing both force and torque components for each cell. Similarly, $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{6N}$ is the generalized velocity vector containing the resulting translational and angular velocities. The vectors are defined as:

$$\begin{aligned} \boldsymbol{\mathcal{F}} &= [\, \mathbf{f}_1, \boldsymbol{\tau}_1, \ldots, \mathbf{f}_N, \ \boldsymbol{\tau}_N]^\top \in \mathbb{R}^{6N}, \\ \boldsymbol{\mathcal{U}} &= [\mathbf{u}_1, \boldsymbol{\omega}_1, \ldots, \mathbf{u}_N, \boldsymbol{\omega}_N]^\top \in \mathbb{R}^{6N} \end{aligned} \tag{4}$$

where, $\mathbf{f}_i = \sum_j \mathbf{F}_{ij}$ is the total force on cell i due to all other cells j in contact with it, and $\boldsymbol{\tau}_i = \sum_j \mathbf{r}_{ij} \times \mathbf{F}_{ij}$ is the total torque. The diagonal mobility matrix $\boldsymbol{\mathcal{M}}^k$ at timestep k encodes the drag coefficients for all cells based on their current lengths ℓ_i^k:

$$\boldsymbol{\mathcal{M}}^k = \text{diag}\left(\frac{1}{\zeta \ell_1^k} \mathbf{I}_3, \frac{12}{\zeta {\ell_1^k}^3} \mathbf{I}_3, \ldots \right) \in \mathbb{R}^{6N \times 6N} \tag{5}$$

where $\mathbf{I}_3$ is the 3×3 identity matrix. The resulting matrix-vector product thus implements the overdamped Langevin equation from Equation 1 for all cells simultaneously.

C. State Integration

The orientation quaternion $\mathbf{q}_n^k$ for each cell evolves according to the kinematic equation $\dot{\mathbf{q}}_n^k = \frac{1}{2} \boldsymbol{\omega}_n \otimes \mathbf{q}_n^k$, where $\otimes$ denotes quaternion multiplication. To integrate translational and rotational dynamics consistently, we introduce a mapping $\boldsymbol{\mathcal{G}}^k \in \mathbb{R}^{7N \times 6N}$, converting Cartesian translational and angular velocities from $\mathbb{R}^{6N}$ into corresponding changes in positions and quaternions in $\mathbb{R}^{7N}$ according to the quaternion kinematics (see [7], [22], [23]). Using $\boldsymbol{\mathcal{G}}^k$, the colony's state and the length vector are updated via explicit Euler integration.

$$\begin{aligned} \boldsymbol{\mathcal{C}}^{k+1} &= \boldsymbol{\mathcal{C}}^k + \Delta t \, \boldsymbol{\mathcal{G}}^k \boldsymbol{\mathcal{U}} = \boldsymbol{\mathcal{C}}^k + \Delta t \, \boldsymbol{\mathcal{G}}^k \boldsymbol{\mathcal{M}}^k \boldsymbol{\mathcal{F}} \\ \boldsymbol{\ell}^{k+1} &= \boldsymbol{\ell}^k + \Delta t \, \dot{\boldsymbol{\ell}} \end{aligned} \tag{6}$$

The distinction between hard and soft collision models lies solely in how the generalized force vector $\boldsymbol{\mathcal{F}}$ and in particular the inter-cell forces $\mathbf{F}_{ij}$ are computed. The rest of the framework, including state representation, dynamics, and integration, remains identical.

D. Soft Collision Model

The soft collision model treats cell-cell interactions through continuous, potential-based repulsive forces. When cells overlap, these forces smoothly increase, preventing significant interpenetration while allowing elastic deformation. This approach is well-suited for simulating the soft, deformable nature of real biological cells and has been widely used in prior work [5], [12]–[19].

We implement a Hertzian contact model, which describes elastic deformation between curved surfaces in contact. The repulsive elastic force exerted on cell i by cell j is:

$$\mathbf{F}_{ij}^{\text{elastic}} = k_{cc}\sqrt{d}\,\delta^{3/2}\,\hat{\mathbf{n}} \tag{7}$$

where δ is the overlap distance, d is the cell diameter, k_{cc} is an elastic constant, and $\hat{\mathbf{n}}$ is the unit normal vector at the contact point. The $\delta^{3/2}$ dependence is characteristic of Hertzian theory for elastic contacts and produces a nonlinear, smooth increase in force as overlap grows (see Figure 3).

1) Force Assembly: The total force and torque on cell n are computed by summing contributions from all pairwise elastic interactions and can be used to assemble the global force vector $\boldsymbol{\mathcal{F}}$ and compute the cell growth rates $\dot{\boldsymbol{\ell}}$ via Equation 2:

$$\mathbf{f}_i = \sum_{j\neq i} \mathbf{F}_{ij}^{\text{elastic}}, \qquad \boldsymbol{\tau}_i = \sum_{j\neq i} \mathbf{r}_{ij} \times \mathbf{F}_{ij}^{\text{elastic}} \tag{8}$$

2) Model Parameters and Numerical Stability: The elastic constant is defined as $k_{cc} = \frac{Y}{\zeta}$, where Y is the cell's Young's modulus and ζ is the fluid drag coefficient. Using typical bacterial parameters ($Y \approx 4$ MPa and $\zeta \approx 200$ Pa·h) [12], [14], we set $k_{cc} = 20000$ h^{-1}. This value is sufficiently large to prevent significant overlap between cells. However, the steep, nonlinear nature of the Hertzian force law creates numerical stiffness: cells can undergo rapid acceleration when overlap occurs, potentially causing large position changes within a single timestep. To maintain stability, an extremely small timestep ($\Delta t \sim 10^{-5}$) is required [12], [14], [15]. This timestep restriction is a well-known limitation of soft collision models and creates a significant computational bottleneck for large colonies.

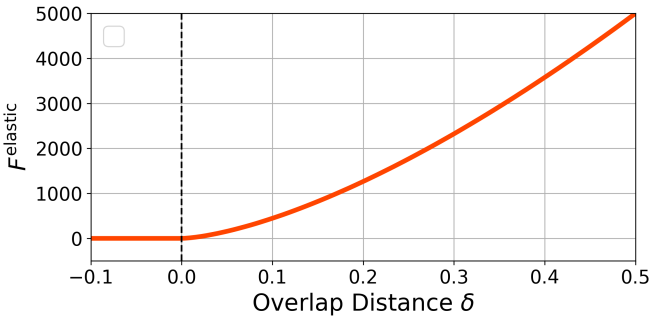


Figure 3: Hertzian contact model. When two spherocylindrical cells overlap by a distance δ, they experience a repulsive elastic force $\mathbf{F}^{\text{elastic}}$. The force acts along the normal vector $\hat{\mathbf{n}}$ at the contact point.

E. Hard Collision Model

The hard collision model adapted from Weady et al. [22] enforces strict non-overlapping constraints between cells using a constraint-based method. For each nearby cell pair, a constraint α is defined based on the closest points on their surfaces, $\mathbf{y}_n$ and $\mathbf{y}_m$. Unlike the soft model, which uses repulsive forces, the hard model determines contact forces via unknown scalar Lagrange multipliers γ_α, which represent the magnitude of the impulse needed to prevent penetration.

The force on cell n from constraint α is:

$$\mathbf{F}_{n\alpha}^{\text{hard}} = \hat{\mathbf{n}}_\alpha \gamma_\alpha \tag{9}$$

where $\hat{\mathbf{n}}_\alpha$ is the normal vector at the contact point. The total generalized force vector $\boldsymbol{\mathcal{F}}$ for the colony depends linearly on the multipliers $\boldsymbol{\gamma} = [\dots, \gamma_\alpha, \dots]^\top \in \mathbb{R}^C$ and can be conveniently expressed in matrix form:

$$\boldsymbol{\mathcal{F}}(\boldsymbol{\gamma}) = \boldsymbol{\mathcal{D}}\boldsymbol{\gamma} \tag{10}$$

where $\boldsymbol{\mathcal{D}} \in \mathbb{R}^{6N\times C}$ is a sparse matrix consisting of contact normals and lever arms for all constraints. This matrix-vector product mimics the force assembly in Equation 8, but uses the unknown multipliers $\boldsymbol{\gamma}$ instead of explicit force calculations. Similarly, the stress vector $\boldsymbol{\sigma} = [\dots, \sigma_n, \dots]^\top \in \mathbb{R}^N$ can also be expressed linearly in terms of $\boldsymbol{\gamma}$:

$$\boldsymbol{\sigma}(\boldsymbol{\gamma}) = \boldsymbol{\mathcal{L}}\boldsymbol{\gamma} \tag{11}$$

where $\boldsymbol{\mathcal{L}} \in \mathbb{R}^{N\times C}$ encodes stress contributions from each constraint, such that the matrix-vector product mimics the stress calculation in Equation 3. This also means that the growth rates $\dot{\boldsymbol{\ell}}$ depend (nonlinearly) on $\boldsymbol{\gamma}$ via Equation 2.

1) Constraint Conditions: To ensure that solving for the multipliers $\boldsymbol{\gamma}$ leads to physically meaningful results, three key conditions must be satisfied:

1) **Repulsive Forces:** $\boldsymbol{\gamma} \geq \mathbf{0}$. This ensures that the forces are repulsive, preventing cells from attracting each other.
2) **No-Overlap:** $\boldsymbol{\Phi}^{k+1} \geq \mathbf{0}$ with $\boldsymbol{\Phi}^{k+1} = [\dots, \Phi_\alpha^{k+1}, \dots]^\top$. Here, $\boldsymbol{\Phi}^{k+1} \in \mathbb{R}^C$ is the vector of signed separation distances between the pairs of cells associated with each constraint α at timestep $k+1$. Signed separation distances between cells are computed using the robust geometric algorithm described by [6], [20]. A positive value Φ_α^{k+1} indicates that the cells in the next iteration are separated, while a negative value indicates overlap. This condition enforces that no overlaps occur after the update.
3) **Complementarity Condition:** $\boldsymbol{\gamma}^\top\boldsymbol{\Phi}^{k+1} = \mathbf{0}$ (often written as $\boldsymbol{\gamma} \perp \boldsymbol{\Phi}^{k+1}$). This condition enforces that if two cells are in contact in the next iteration (i.e., $\Phi_\alpha^{k+1} = 0$), then a force may act in the current iteration to prevent overlap. Conversely, if two cells are separated in the next iteration (i.e., $\Phi_\alpha^{k+1} > 0$), no force should act in the current iteration (i.e., $\gamma_\alpha = 0$) in order to avoid force application at a distance.

These conditions are often abbreviated as:

$$\mathbf{0} \leq \boldsymbol{\gamma} \perp \boldsymbol{\Phi}^{k+1} \geq \mathbf{0}. \tag{12}$$

The colony update rule and additional conditions combine to form a system where the future state $\mathcal{C}^{k+1}$ and $\boldsymbol{\ell}^{k+1}$, and thus $\boldsymbol{\Phi}^{k+1}$ depend on $\boldsymbol{\gamma}$:

$$\begin{aligned} \mathcal{C}^{k+1} &= \mathcal{C}^k + \Delta t \mathcal{G}^k \mathcal{M}^k \mathcal{F}(\boldsymbol{\gamma}) \\ \boldsymbol{\ell}^{k+1} &= \boldsymbol{\ell}^k + \Delta t\, \dot{\boldsymbol{\ell}}(\boldsymbol{\gamma}) \\ \text{s.t.} \quad & \mathbf{0} \leq \boldsymbol{\gamma} \perp \boldsymbol{\Phi}^{k+1}(\boldsymbol{\gamma}) \geq \mathbf{0}. \end{aligned} \tag{13}$$

2) Linearization of the Constraint Condition: The central challenge is that $\boldsymbol{\Phi}^{k+1}(\boldsymbol{\gamma})$ is a nonlinear function of the updated state $\mathcal{C}^{k+1}$ and the updated lengths $\boldsymbol{\ell}^{k+1}$, and is non-trivial to compute directly. Weady et al. [22] propose a linearization approach to approximate $\boldsymbol{\Phi}^{k+1}(\boldsymbol{\gamma})$ using a Taylor expansion around the current state $\mathcal{C}^k$ and lengths $\boldsymbol{\ell}^k$.
The change in $\boldsymbol{\Phi}$ in our model arises from two main effects:

1) **Cell Motion:** All cell distances change due to the motion driven by collision forces. This is captured by the term $\dot{\boldsymbol{\Phi}}^k{}_{\text{motion}} = \frac{d\boldsymbol{\Phi}^k}{d\mathcal{C}} \frac{d\mathcal{C}}{dt} = (\nabla_{\mathcal{C}} \boldsymbol{\Phi}^k)\dot{\mathcal{C}}$.
2) **Cell Growth:** Additionally, cell growth affects separation according to $\dot{\boldsymbol{\Phi}}^k{}_{\text{growth}} = \frac{d\boldsymbol{\Phi}^k}{d\boldsymbol{\ell}} \frac{d\boldsymbol{\ell}}{dt} = -(\nabla_{\boldsymbol{\ell}} \boldsymbol{\Phi}^k)\dot{\boldsymbol{\ell}}$.
The negative sign arises because as cells elongate, they reduce the separation distance between them.

Combining these two effects leads to the following approximation for $\boldsymbol{\Phi}^{k+1}$:

$$\begin{aligned} \boldsymbol{\Phi}^{k+1}(\boldsymbol{\gamma}) &\approx \boldsymbol{\Phi}^k + \Delta t \left(\dot{\boldsymbol{\Phi}}_{\text{motion}} + \dot{\boldsymbol{\Phi}}_{\text{growth}} \right) \\ &= \boldsymbol{\Phi}^k + \Delta t \left((\nabla_{\mathcal{C}} \boldsymbol{\Phi}^k)\dot{\mathcal{C}}^k - (\nabla_{\boldsymbol{\ell}} \boldsymbol{\Phi}^k)\dot{\boldsymbol{\ell}} \right) \\ &= \boldsymbol{\Phi}^k + \Delta t \left((\nabla_{\mathcal{C}} \boldsymbol{\Phi}^k)\mathcal{G}^k \mathcal{M}^k \mathcal{F}(\boldsymbol{\gamma}) - (\nabla_{\boldsymbol{\ell}} \boldsymbol{\Phi}^k)\dot{\boldsymbol{\ell}} \right) \\ &\overset{\dagger}{=} \boldsymbol{\Phi}^k + \Delta t \left(\mathcal{D}^\top \mathcal{M}^k \mathcal{F}(\boldsymbol{\gamma}) - (\nabla_{\boldsymbol{\ell}} \boldsymbol{\Phi}^k)\dot{\boldsymbol{\ell}} \right) \\ &\overset{\ddagger}{=} \boldsymbol{\Phi}^k + \Delta t \left(\mathcal{D}^\top \mathcal{M}^k \mathcal{D}\boldsymbol{\gamma} - \mathcal{L}^\top \left(\frac{\boldsymbol{\ell}}{\tau} \odot e^{-\lambda \mathcal{L}\boldsymbol{\gamma}} \right) \right) \end{aligned} \tag{14}$$

where $\odot$ denotes the element-wise product.

Here $(\dagger)$, we use the previously introduced $\mathcal{D}$, noting that $\mathcal{D}^\top = (\nabla_{\mathcal{C}} \boldsymbol{\Phi}^k)\mathcal{G}^k$. The transpose $\mathcal{D}^\top$ can thus be interpreted as a Jacobian mapping velocities in $\mathbb{R}^{6N}$ first to changes in the colony state (via $\mathcal{G}^k$) and then to changes in the signed distance functions $\boldsymbol{\Phi}$ (see [7], [22] for details).

Similarly, in $(\ddagger)$, the previously defined $\mathcal{L}$, which encodes stress contributions from each constraint, has a transpose $\mathcal{L}^\top$ that maps stress-induced changes in cell lengths $\boldsymbol{\ell}$ to variations in $\boldsymbol{\Phi}$ (see [22] for details).

3) Nonlinear Complementarity Problem: Substituting the approximation of $\boldsymbol{\Phi}^{k+1}(\boldsymbol{\gamma})$ from Equation 14 into the constraint conditions of Equation 13 yields a nonlinear complementarity problem (NCP) for $\boldsymbol{\gamma}$ which no longer depends on the unknown future states $\mathcal{C}^{k+1}(\boldsymbol{\gamma})$ and $\boldsymbol{\ell}^{k+1}(\boldsymbol{\gamma})$:

Find $\boldsymbol{\gamma}$ such that

$$\mathbf{0} \leq \boldsymbol{\gamma} \perp \boldsymbol{\Phi}^k + \Delta t \left(\mathcal{D}^\top \mathcal{M}^k \mathcal{D}\boldsymbol{\gamma} - \mathcal{L}^\top \left(\frac{\boldsymbol{\ell}}{\tau} \odot e^{-\lambda \mathcal{L}\boldsymbol{\gamma}} \right) \right) \geq \mathbf{0}. \tag{15}$$

4) Energy Minimization Solution: To solve this nonlinear complementarity problem efficiently, we reframe it as an optimization problem using a special energy function $E(\boldsymbol{\gamma})$ whose minimization under non-negativity constraints resolves the contact forces [24]. Physically, $E(\boldsymbol{\gamma})$ represents the work done by the contact forces and the system's internal energy:

$$\begin{aligned} \min_{\boldsymbol{\gamma} \geq \mathbf{0}} E(\boldsymbol{\gamma}) = \min_{\boldsymbol{\gamma} \geq \mathbf{0}} & \boldsymbol{\gamma}^\top \boldsymbol{\Phi}^k + \frac{\Delta t}{2} \boldsymbol{\gamma}^\top \mathcal{D}^\top \mathcal{M}^k \mathcal{D}\boldsymbol{\gamma} \\ & + \mathbf{1}^\top \frac{\Delta t}{\lambda} \left(\frac{\boldsymbol{\ell}}{\tau} \odot e^{-\lambda \mathcal{L}\boldsymbol{\gamma}} \right) \end{aligned} \tag{16}$$

By construction, the gradient of this energy equals the linearized separation distances $\boldsymbol{\Phi}^{k+1}(\boldsymbol{\gamma})$ from Equation 14:

$$\begin{aligned} \nabla_{\boldsymbol{\gamma}} E &= \boldsymbol{\Phi}^k + \Delta t \left(\mathcal{D}^\top \mathcal{M}^k \mathcal{D}\boldsymbol{\gamma} - \mathcal{L}^\top \left(\frac{\boldsymbol{\ell}}{\tau} \odot e^{-\lambda \mathcal{L}\boldsymbol{\gamma}} \right) \right) \\ &= \boldsymbol{\Phi}^{k+1}(\boldsymbol{\gamma}) \end{aligned} \tag{17}$$

Because $E(\boldsymbol{\gamma})$ is convex, the constrained optimization problem has a unique global minimizer over the feasible region $\boldsymbol{\gamma} \geq \mathbf{0}$. We compute this minimum using a projected gradient method, which iteratively takes steps in the direction of the negative gradient and projects the result onto the feasible region to ensure repulsive forces.

At the optimal solution $\boldsymbol{\gamma}^*$, the Karush-Kuhn-Tucker (KKT) conditions for this constrained minimization problem are exactly equivalent to our physical constraints:

1) **Primal feasibility** ($\boldsymbol{\gamma}^* \geq 0$): We enforce this explicitly through a projection (`max` operator) during minimization, which prevents forces from becoming attractive. Physically, this ensures that contact forces only push cells apart according to the **Repulsive Forces** constraint.
2) **Dual feasibility** ($\nabla E(\boldsymbol{\gamma}^*) \geq 0$): At the solution $\boldsymbol{\gamma}^*$, the directional derivative of the energy must be non-negative in every feasible direction. A negative partial derivative would indicate that the energy could be reduced by increasing γ_α, contradicting the optimality of $\boldsymbol{\gamma}^*$. Given the identity $\nabla E(\boldsymbol{\gamma}^*) = \boldsymbol{\Phi}^{k+1}(\boldsymbol{\gamma}^*)$, this condition ensures $\boldsymbol{\Phi}^{k+1} \geq 0$, enforcing the **No-Overlap** constraint.
3) **Complementary slackness** ($\boldsymbol{\gamma}^{*\top} \boldsymbol{\Phi}^{k+1} = 0$): This condition holds for a similar reason. Since both $\boldsymbol{\gamma}^* \geq 0$ and $\boldsymbol{\Phi}^{k+1} \geq 0$ at the solution, their product can only be zero if at least one term is zero for each contact. If a contact force is active ($\gamma_\alpha > 0$), it means the system is free to adjust this force to find the minimum. At the true minimum, there can be no direction left to lower the energy. Therefore, for a non-zero force to be part of the final, optimal solution, the gradient (which equals Φ_α^{k+1}) must be zero at that point. Conversely, if $\gamma_\alpha = 0$, the gradient can be positive ($\Phi_\alpha^{k+1} \geq 0$) because the energy cannot be lowered by making γ_α more negative (as it is forbidden by the Repulsive Forces constraint). Thus, the product $\gamma_\alpha \Phi_\alpha^{k+1} = 0$ for each contact α, leading to the overall complementary slackness condition. Physically, this means forces act only where cells are touching. Separated cells never experience any repulsive force, satisfying the **Complementarity Condition**.

Thus, the unique global minimizer $\boldsymbol{\gamma}^*$ resolves all contacts according to the physical laws of rigid-body interactions. This equivalence between the KKT conditions of a convex optimization problem and a complementarity problem is a well-known result in optimization theory [25] and is widely used in constraint-based physics simulations [6]–[11], [22]–[24].

5) Numerical Solution: We solve the constrained optimization problem using the Barzilai-Borwein projected gradient descent method (BBPGD) [26], following the implementation described in [6], [22]. This method is well-suited to large-scale problems due to its computational efficiency and scalability. The solver iterates until convergence, yielding the optimal contact forces $\boldsymbol{\gamma}^*$, which are then used to compute $\boldsymbol{\mathcal{F}}(\boldsymbol{\gamma}^*)$ and $\dot{\boldsymbol{\ell}}(\boldsymbol{\gamma}^*)$ for updating the colony state via Equation 6.

Following Weady et al. [22], we adopt a convergence criterion based on a residual that directly measures physical overlap. The solver terminates when this residual falls below a user-specified tolerance of $\epsilon = d/500 = 0.5/500 = 10^{-3}$, ensuring that the maximum overlap between any pair of cells remains below 0.1% of the cell diameter $d = 0.5$.

6) Limitations of the Linearization: A fundamental limitation of the linearization is that orthogonal motions, specifically rotations about contact points and translations parallel to the contact plane, are unconstrained within the linear approximation [22]. Consequently, even after the BBPGD algorithm converges to the specified tolerance ϵ, residual overlaps may still be present that violate the non-overlap constraint.

To address this limitation, Weady et al. [22] developed the recursively generated linear complementarity problem (ReLCP) method. After each BBPGD convergence, ReLCP identifies any remaining overlaps by examining all pairwise cell distances, generates new constraints for these violations, and re-solves the complementarity problem with the augmented constraint set. This process iterates until all cell overlaps fall below the tolerance ϵ. This hierarchical approach ensures that the non-overlap condition is strictly enforced, even in the presence of complex contact geometries and coupled motions (see Figure 17b). The iterative refinement procedure typically converges within a small number of iterations, making it computationally tractable for most simulations [22].

Nevertheless, this approach still requires a sufficiently small timestep, $\Delta t \sim 2 \cdot 10^{-3}$ (see Figure 16), to prevent excessive cell motion [23]. If cells move too far in a single timestep, new overlaps can form that the linearization does not capture, preventing constraint resolution. In practice, this can cause the BBPGD method to fail to converge or the ReLCP solver to be unable to resolve all overlaps within a reasonable number of iterations, leading to simulation failure.

Despite this computational restriction, the hard model's timestep requirement is less severe than the soft model's, though it still represents a significant computational bottleneck for large colonies with millions of cells.

V. Implementation

All simulations in this paper were performed with a custom C++ framework designed for large-scale simulations of proliferating cell collectives. The framework leverages distributed-memory parallel computing and standard HPC libraries to achieve scalability, enabling simulations of hundreds of thousands of cells across hundreds of CPU cores. The implementation combines concepts from Weady et al. [22] and similar frameworks [6], [7].

A. Distributed Computing Architecture

The core architecture is built on two foundational libraries:

- **PETSc** [27]: Used as a backend for distributed vectors and sparse matrices. PETSc partitions data across MPI processes and transparently handles inter-process communication for global operations (e.g., matrix-vector products and reductions).
- **MPI**: Underlies all inter-process communication, including ghost-cell exchanges at domain boundaries to maintain consistent collision handling across partitions.

Built on these abstractions, the framework implements the full physics pipeline, including both the ReLCP solver for the hard collision model and the soft collision model.

B. Collision Handling Pipeline

Collisions are processed through a unified pipeline designed to decouple geometry from physics. The process consists of two phases: broad-phase detection using a spatial grid (reducing neighbor searches to $O(N)$) and narrow-phase detection that computes exact contact points and geometric information. For each collision detected, the framework assembles a standardized data structure containing contact points, normals, and lever arms. This structure is passed to either the hard or soft collision model, which computes the resulting forces and torques. This abstraction allows seamless switching between collision models without altering the overall simulation flow.

Each collision detector call generates a single constraint per neighboring cell-pair within a threshold distance $d = 0.5$. As this threshold is larger than zero, it captures both overlapping and closely adjacent cells. This overapproximation is crucial for the stability of the hard model. For the soft model, those non-overlapping cell-pairs contribute no forces (since $\delta < 0$), while for the hard model they provide additional global information that improves constraint resolution. Ideally, constraints would be constructed between all cell pairs regardless of distance, but for computational efficiency only nearby cells are considered [6], [23].

C. Adaptive Timestepping

To enhance stability and convergence, particularly for fast-moving cells that might overlap significantly within a timestep, we employ an adaptive timestepping scheme, allowing the timestep Δt to vary dynamically based on the current state of the colony. To our knowledge, this approach has not been implemented in comparable engines.

The adaptive timestep strategy (outlined in Algorithm 1) is based on the Courant-Friedrichs-Lewy (CFL) condition [28], which provides a stability criterion linking temporal and spatial resolution:

$$\text{CFL} = \frac{u\,\Delta t}{\Delta x} \leq C_{\max}. \tag{18}$$

For our cell-based system, we define a characteristic velocity scale u_m as the median of all cell velocities (including growth rates) and a characteristic spatial scale Δx as the solver tolerance $\varepsilon = 10^{-3}$, which controls the maximum allowable displacement per step. Enforcing that cells move no more than a fraction of ε per timestep leads to the following expression for the adaptive timestep:

$$\Delta t = \frac{c \cdot \varepsilon}{u_m} \tag{19}$$

where c is a user-defined CFL number (we use $c = 0.5$, ensuring cells move at most half the tolerance per step). Choosing Δt like this, ensures that most cells move less than half the solver tolerance in each timestep, preventing excessive overlaps that could destabilize the simulation.

Algorithm 1 Adaptive Timestep Control

Require: Particle set $\{p_i\}$ with velocities v_i and growth rates $\dot{\ell}_i$, current timestep Δt, solver tolerance ε, smoothing factor α.

Ensure: Updated timestep Δt.

1: Compute characteristic velocity $u_i = \|v_i\| + \dot{\ell}_i$ for each particle.
2: Calculate u_m as the median of $\{u_i\}$.
3: Determine Δt^* using Equation 19.
4: Smooth the timestep change to avoid oscillations: $\Delta t_{\text{new}} = (1-\alpha)\Delta t + \alpha \Delta t^*$ with $\alpha = 0.01$.
5: Clamp Δt_{new} within 20% of current Δt to prevent abrupt changes.
6: Update simulation timestep: $\Delta t \leftarrow \Delta t_{\text{new}}$.

D. Simulation Output and Availability

Simulation results are saved in VTK format for visualization with ParaView [29]. Cell attributes, including positions, orientations, lengths, stresses, growth rates and contact forces, are recorded at user-defined intervals. The framework also logs performance metrics, such as timestep sizes, solver iterations, and computation times, to facilitate analysis and debugging.

The full simulation framework, including all features and implementations described in this work, is openly available on GitHub at https://github.com/manuellerchner/MicrobeGrowthSim-IDP.

VI. Pattern Formation Analysis

A. Concentric Ring Patterns

We validate both implementations by reproducing concentric ring patterns that arise under stress-sensitive growth, as reported in [1]. As shown in Figure 4, both models successfully produce qualitatively similar ring structures, demonstrating that the core mechanics of growth and mechanical feedback are effectively captured by both collision handling approaches.

The effect of λ on pattern formation is consistent across both models. At low stress-sensitivity (e.g. $\lambda = 10^{-4}$, corresponding to weak growth inhibition), stress gradients are too weak to generate significant spatial patterns, and thus no rings form. At intermediate sensitivity (e.g. $\lambda = 10^{-3}$), distinct concentric rings emerge early in the simulation and persist as the colony expands, with ring spacing determined by the balance between growth and stress. At high sensitivity (e.g. $\lambda = 10^{-2}$, corresponding to strong growth inhibition), an initially formed ring quickly dissipates as cells reorient to random orientations and growth nearly ceases due to strong mechanical inhibition throughout most of the colony.

Despite these qualitatively similar outcomes, visual differences emerge between the models. The soft model produces visibly denser patterns with a fuzzy appearance due to allowable cell overlap, while the hard model yields sharper, more sparsely looking rings. Nevertheless, macroscopic features, such as the number of rings, spacing, and colony expansion rate, remain comparable between the two models.

B. Cell Density and Local Packing Fraction

The most striking difference between the models is the cell packing density (see Figure 5). The soft model exhibits significantly denser packing in the colony center, decreasing radially outward (see Figure 6a). For very low stress-sensitivity ($\lambda = 10^{-4}$), local packing fractions (defined as the ratio of cell area to available area) reach values of 5, indicating severe cell overlap. Values exceeding 1.0 are impossible for truly rigid spherocylinders and reflect the soft model's inability to prevent interpenetration. In contrast, the hard model maintains approximately constant, and physically realistic, packing density of 0.9 throughout the colony.

The unphysical packing fraction arises because the soft model relies on local pairwise forces that cannot fully propagate stress to the colony edge. Combined with a numerical timestep too large to resolve all collisions, this leads to excessive cell overlap and overcrowding in the interior. Figure 6b quantifies this effect: the soft model's maximum cell overlap steadily increases throughout the simulation, reaching nearly complete interpenetration by the end. In contrast, the hard model maintains maximum overlap near the solver tolerance $\epsilon = 10^{-3}$ by enforcing the strict non-overlap constraints.

A contributing factor to this excessive overlap is the adaptive timestep method (see Algorithm 1), which bases Δt solely on cell velocities but not on current overlap. A more sophisticated scheme could consider maximum overlap when adjusting Δt. Such an improvement is, however, beyond the scope of this work.

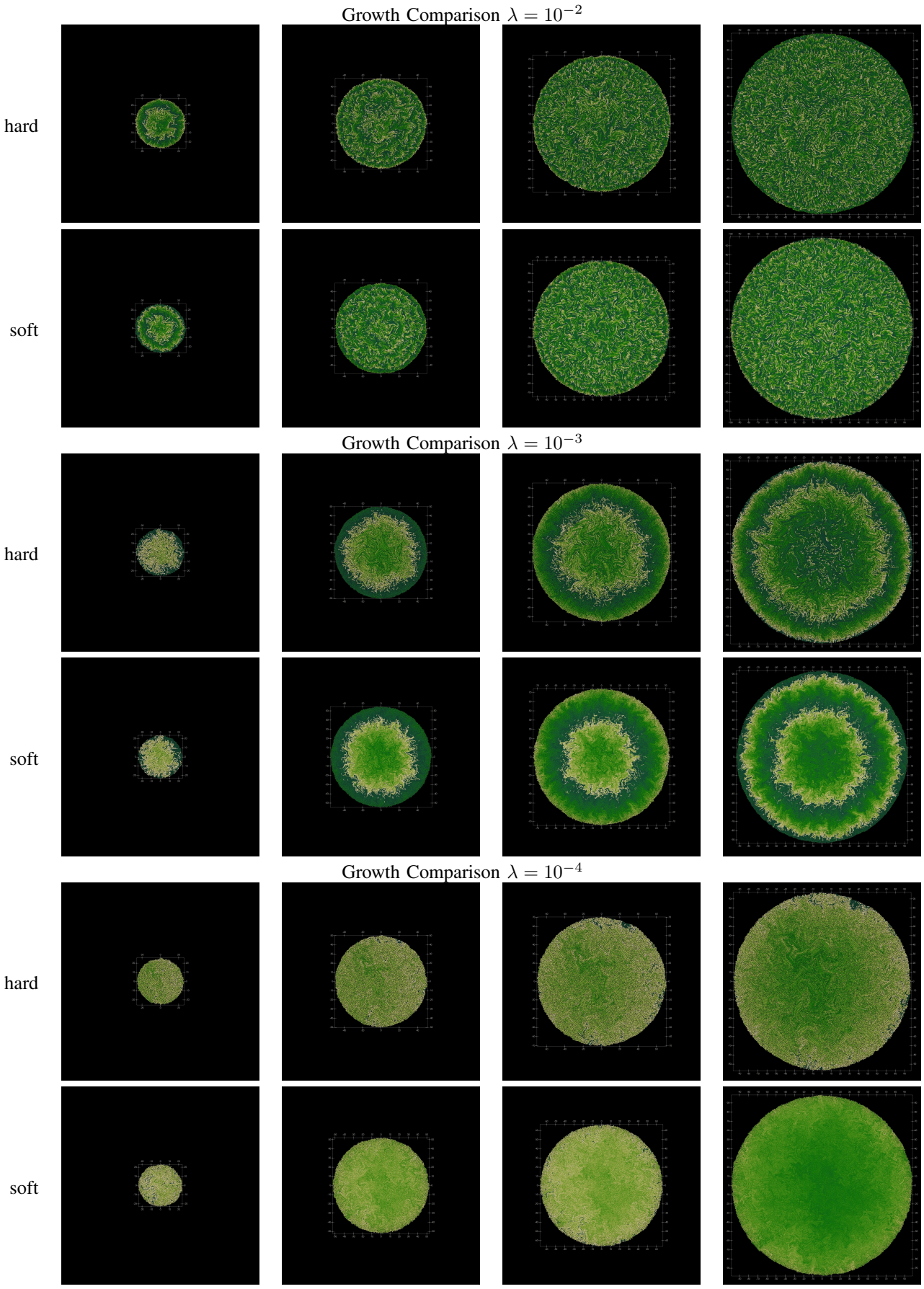
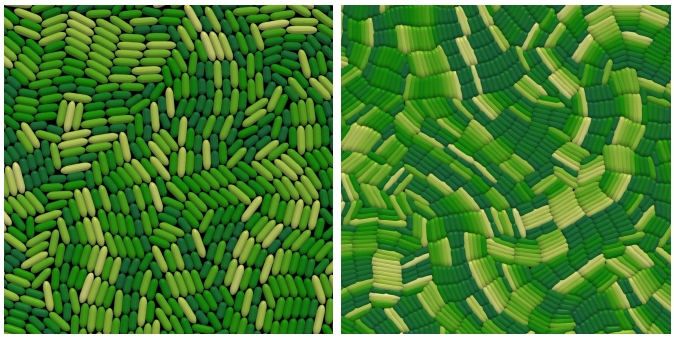
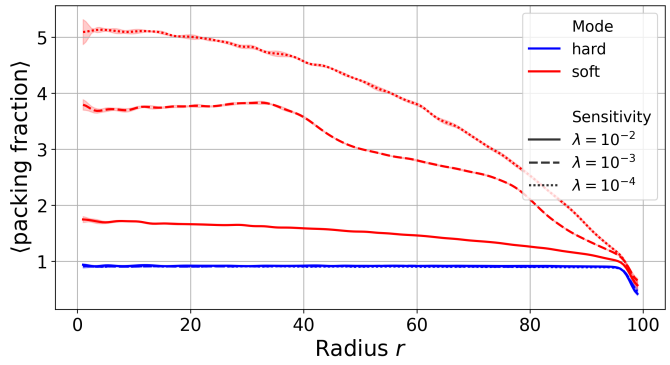


Figure 4: Comparison of pattern formation between hard and soft collision models at similar colony sizes. Each row shows the colony evolutions up to a maximum colony radius of 100. The color indicates the length of the cells (short cells are darker, longer cells are lighter). Clear concentric ring patterns are visible for both models at $\lambda = 10^{-3}$ confirming the results from [1].

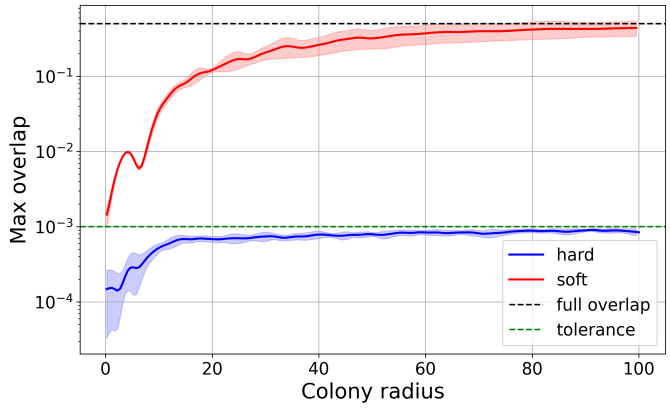


(a) Hard model (b) Soft model

Figure 5: Close-up comparison of cell packing in the colony center for both collision models. The hard model (a) maintains near-optimal packing, while the soft model (b) exhibits significant overlap and overcrowding.



(a) Radial packing fraction profiles ($\phi = \frac{A_{\text{cells}}}{A_{\text{total}}}$) for both models. Here A_{cells} is the total area occupied by cells within a radial bin of width 2 at distance r from the colony center, and A_{total} is the annular area of that bin. The hard model maintains nearly optimal packing, while the soft model exhibits higher fractions in the colony center, especially at low stress-sensitivity ($\lambda = 10^{-4}$).



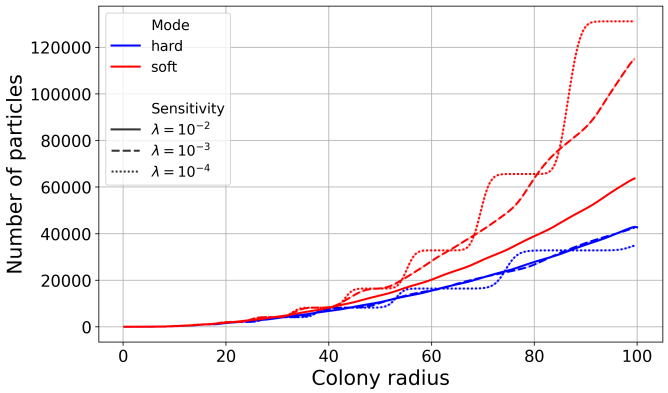
(b) Maximum cell overlap over simulation time. The hard model maintains overlap near the solver tolerance $\epsilon = 10^{-3}$, while the soft model exhibits a steady increase, reaching near-complete overlap by the simulation end.

Figure 6: Cell packing analysis comparing hard and soft collision models. (a) Radial packing fraction profiles. (b) Maximum cell overlap vs. colony growth.

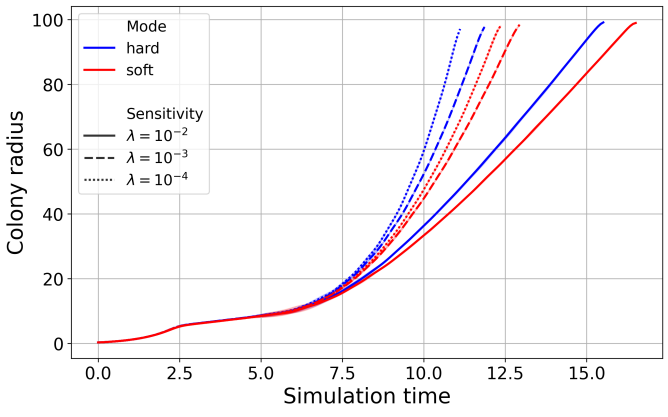
C. Colony Growth Dynamics and Number of Cells

A direct consequence of the overcrowded interior in the soft model is its effect on the total number of cells. The soft model requires substantially more cells to reach the same colony radius (see Figure 7a). In particular, when stress-sensitivity is low ($\lambda = 10^{-4}$), it needs almost three times as many cells as the hard model to achieve the same radius. This disparity shows that the densely packed interior contributes little to outward colony pressure: interior cells remain largely immobile yet continue to grow and divide, creating a large excess of cells that do not contribute to expansion.

Figure 7b compares simulation time and colony radius for both models. Despite the difference in cell numbers, their overall growth dynamics are similar, with the soft model growing slightly more slowly due to cell overlap. At high stress-sensitivity ($\lambda = 10^{-2}$), both models transition from early exponential growth to slower, linear growth as stress limits expansion in the colony interior.



(a) Number of cells required to reach a given colony radius. The soft model requires substantially more cells, particularly at low stress-sensitivity ($\lambda = 10^{-4}$).



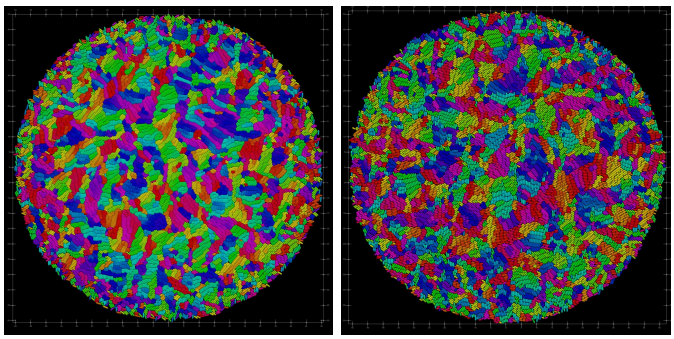
(b) Simulation time vs. colony radius. Both models exhibit similar growth dynamics, though the soft model is slightly slower due to allowable cell overlap.

Figure 7: Colony growth dynamics and cell scaling comparison between hard and soft models. (a) Number of cells vs. colony radius. (b) Simulation time vs. colony radius.

D. Microdomain Formation

Similar to You et al. [14], [16], we observe the formation of microdomains, defined as localized regions where cells share similar orientations. These domains emerge from mechanical interactions and growth dynamics and result in mosaic-like patterns of cell alignment within the colony.

As shown in Figure 8, both hard and soft collision models reproduce this phenomenon for small colonies ($R \approx 50$), producing patterns that closely resemble experimental observations of real *E. coli* colonies [14]. This agreement suggests that at small scales, where the soft model's packing artifacts remain limited, the essential features of microdomain formation can be captured through both collision handling approaches.



(a) Soft model with $\lambda = 10^{-2}$ at colony radius $R \approx 50$.

(b) Hard model with $\lambda = 10^{-2}$ at colony radius $R \approx 50$.

Figure 8: Cell orientation patterns for both models. The color indicates cell orientation angle, with similar colors representing similar angles. Both models produce comparable microdomain structures in small colonies.

Although the hard and soft collision models presented here are not directly comparable to You et al. [14], because their study assumes constant growth rates while our models use stress-dependent growth, their findings still provide useful context. You et al. show that reduced growth rates lead to larger microdomains. In our models, stress sensitivity λ produces a similar effect by inhibiting growth in the densely packed colony center, resulting in larger microdomains. This trend is evident in simulations of the soft model (see Figure 10), where higher λ values correspond to slower growth and larger microdomains, consistent with You et al.'s observations. The hard model, in contrast, shows no such trend. The consistent microdomain size across λ values in the hard model may be related to the global constraint resolution mechanism, but a detailed investigation of the underlying mechanisms is beyond the scope of this comparison.

We saw that both models produce similar orientation patterns in small colonies at a colony radius around $R \approx 50$ (see Figure 8). However, pronounced differences emerge as colonies grow larger or λ decreases. As shown in Figure 10, the hard model maintains distinct patches of similarly-sized aligned cells across all λ values. In contrast, the soft model produces elongated, unrealistic bundles of densely packed aligned cells that do not resemble experimentally observed microdomains.

This artifact stems directly from the soft model's higher interior growth rates, discussed in the previous chapter. Without sufficient outward stress propagation, the interior becomes unphysically dense, forcing cells into long, aligned bundles.

The effect of λ on microdomain structure is particularly apparent in larger colonies. As shown in Figure 9, the hard model produces microdomains of fairly uniform size across all stress-sensitivity values, while the soft model produces larger microdomains with higher variability, as λ decreases.

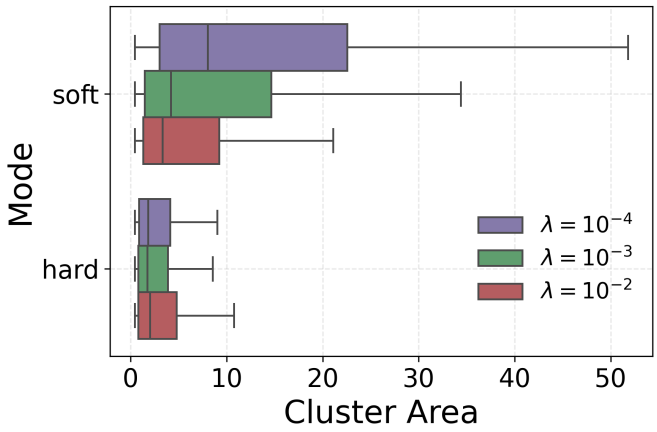


Figure 9: Microdomain area distributions for large colonies ($R \approx 100$) across λ values. The hard model produces consistent domain sizes, while the soft model shows high variability with many large clusters, especially at $\lambda = 10^{-4}$. Clusters are identified using a graph-based connected-component algorithm [14], grouping neighboring cells with similar orientations (within 3% of each other).

As a consequence of bundle formation and overcrowding, the soft model is not suitable for studying microdomain formation in large colonies or in regimes of low stress sensitivity, where these effects distort the local dynamics and lead to unrealistic structural organization. In such cases, excessive overlap and artificial clustering interfere with the natural development of orientational order, preventing a faithful representation of the physical mechanisms driving microdomain formation.

For smaller colonies ($R \lesssim 50$) and at high stress-sensitivity values, the soft model's packing artifacts remain somewhat contained and the overall orientation patterns show qualitative agreement with the hard model. Within these limited conditions, it may serve as a computationally efficient tool for preliminary exploration. For applications that require accurate characterization of microdomain properties or the exploration of weak stress-sensitivity regimes, the hard model remains the preferred choice.

Future work could further examine the influence of the stress-sensitivity parameter λ on the emergence, size, and stability of microdomains in the hard model, following an approach similar to the analyses presented by You et al. [14].

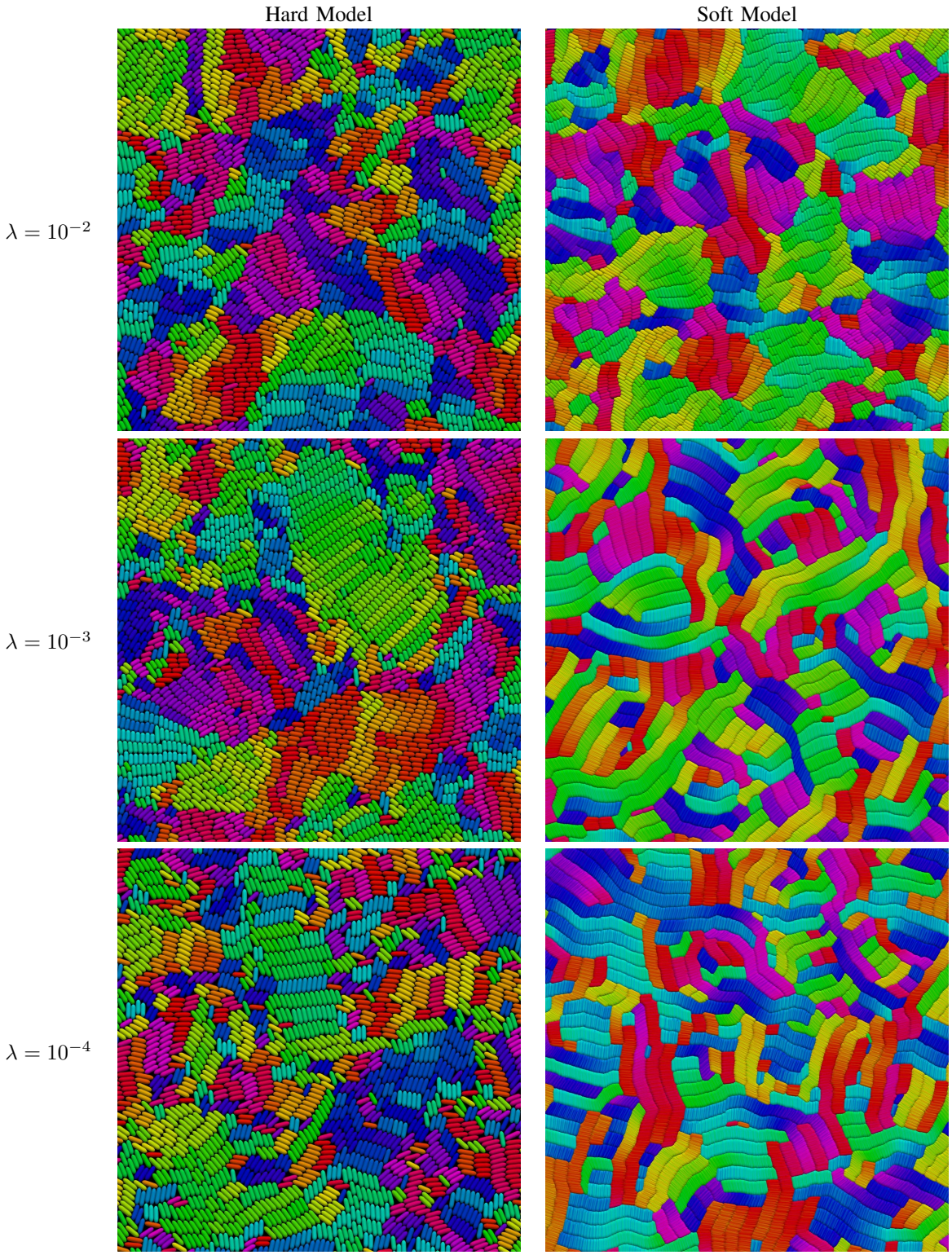


Figure 10: Comparison of orientation patterns at different stress sensitivities in the center of the colony. The color indicates the cell's orientation, with similar colors representing similar angles. The hard model is able to maintain visible patches of aligned cells for all stress sensitivities, whereas the soft model tends to produce long bundles of densely packed cells for low stress sensitivities. This effect is caused by a lack of separation forces, causing cells in the center of the colony to pile up.

E. Radial Stress Distribution and Growth Rate

Another key difference between the two modeling approaches lies in the radial stress distribution that develops within the colony (see Figure 11). Both the hard and soft models produce qualitatively similar radially averaged stress profiles, consistent with the behavior originally reported in [1]. In both cases, the stresses are highest at the colony center and gradually decrease toward the outer edge, reflecting the buildup of mechanical compression in the densely packed interior. These profiles approximately follow the derived analytical expression $\tilde{\sigma}(r) \approx \frac{2}{\lambda} \ln\left(\frac{1}{8c} - c\lambda r^2\right)$ [22], which predicts a logarithmic dependence on the colony radius.

Although the overall shape of the profiles is comparable, the magnitude of the predicted stresses differs substantially between the two models. Quantitatively, the hard model produces stress values that are up to two times higher than the analytical prediction, whereas the soft model yields stresses that are only about half the analytical value. This variation in stress magnitude is significant and likely plays a central role in explaining the discrepancies in growth dynamics and microdomain organization discussed in subsection VI-C and subsection VI-D.

The consistently elevated stresses observed in the hard model are also noted by Weady et al. [1] and are likely a consequence of the discrete cell-based representation not fully satisfying the continuum assumptions used in the analytical derivation.

In contrast, the soft model produces lower stresses because it lacks a global force propagation mechanism: stress is transmitted only through local pairwise repulsive interactions, which are insufficient to communicate mechanical load effectively across the entire colony. Consequently, stresses in the colony interior remain low, and central cells experience weaker mechanical inhibition. These cells therefore continue to grow and divide, eventually leading to the overcrowding and distorted microdomain patterns.

As shown in Figure 12, the differences in stress transmission directly manifest in the radial growth rate profiles. Cells in the soft model exhibit substantially higher growth rates than those in the hard model across most of the colony interior, particularly at low stress-sensitivity values. At higher stress-sensitivity ($\lambda = 10^{-2}$), growth becomes strongly suppressed in the colony interior for both models, and only cells located near the periphery remain actively dividing. The resulting shift from interior-driven to edge-driven proliferation is also clearly visible in Figure 7b and mirrors a well-documented behavior in real microbial systems, such as in bacterial and yeast colonies, including *E. coli* and *Saccharomyces cerevisiae*, where mechanical feedback and nutrient depletion restrict active expansion to the growing front [5], [30], [31].

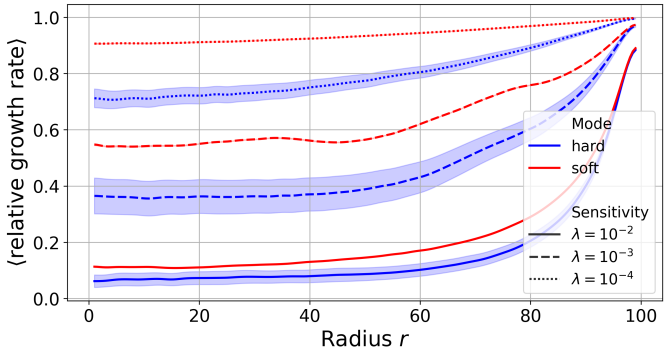


Figure 12: Radial relative growth rate profiles ($e^{-\lambda\sigma}$) for both models. Cells in the soft model grow significantly faster than those in the hard model across most of the colony, particularly at low stress-sensitivity ($\lambda = 10^{-3}$ and $\lambda = 10^{-4}$). At high stress-sensitivity ($\lambda = 10^{-2}$), growth is strongly suppressed in the interior of both models, with only cells at the colony edge actively dividing.

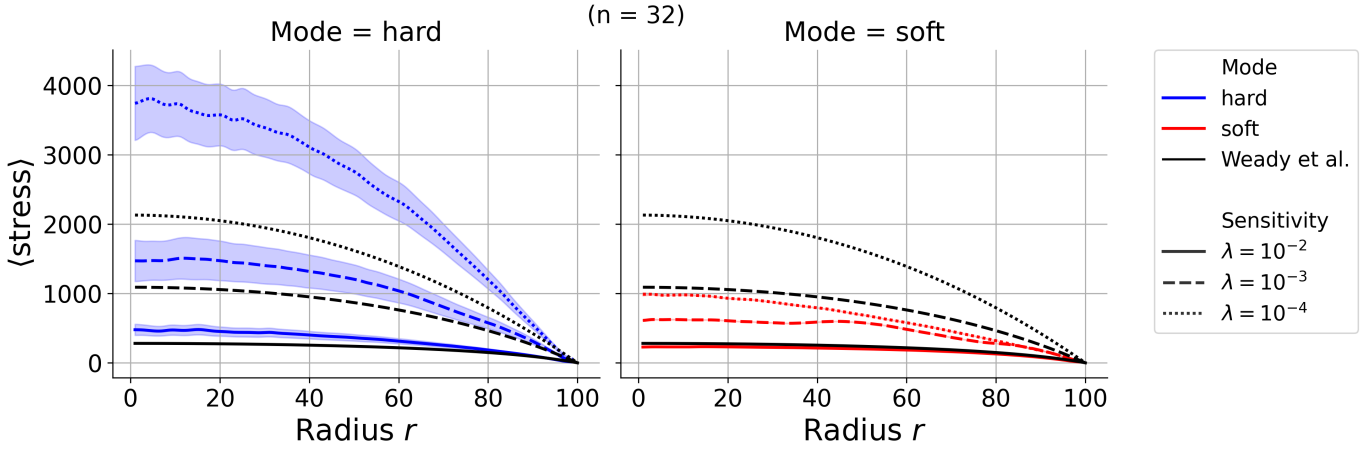


Figure 11: Radial stress profiles for both models across different λ values. Both models exhibit similar qualitative trends, with stresses peaking at the colony center and decreasing toward the edge. However, the hard model consistently produces higher stress magnitudes compared to the soft model. The dashed lines represent the analytical prediction from [22].

VII. Computational Performance

In the previous section, we saw that both the hard and soft collision models produce similar macroscopic outcomes under certain conditions. We now focus on their computational performance. To leverage multiple CPUs, the simulation is parallelized using the Message Passing Interface (MPI), which allows the workload to be distributed efficiently across many cores. All benchmarks are carried out on the `CoolMUC-4`[1] HPC cluster.

A. Domain Decomposition

For parallel execution, the simulation domain is divided into evenly spaced angular sectors of the circular colony, as illustrated in Figure 13. The center of the colony is exclusively assigned to rank 1 to avoid a singularity where all ranks would meet at the center. This angular sector decomposition ensures better load balancing for circular colonies than regular rectangular slices, which would produce significant load imbalance due to the colony's radial symmetry.

Communication between ranks occurs through ghost particle exchange at sector boundaries. At each timestep, particles near the boundaries are sent to neighboring ranks as ghost particles, allowing for correct collision detection across domain borders.

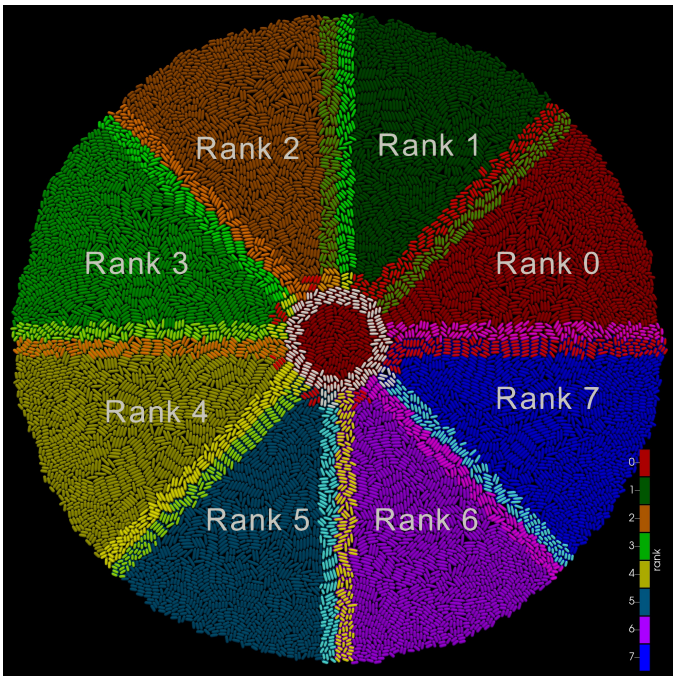


Figure 13: Domain decomposition for 8 MPI ranks showing evenly spaced angular sectors of a colony with radius 50. Each rank owns one angular sector. Ghost particles at rank boundaries are shown in brighter colors.

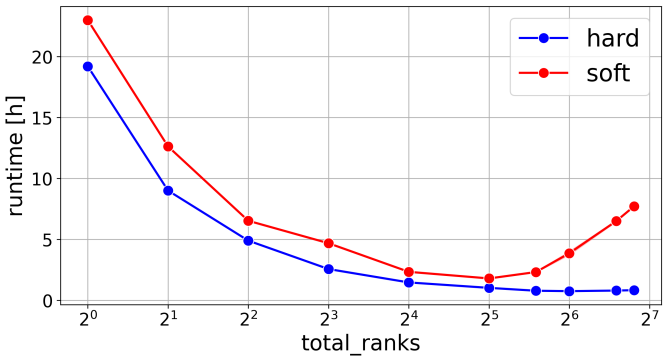
[1] `CoolMUC-4` is part of the Linux Cluster at the Leibniz Supercomputing Centre (LRZ). It consists of 106 compute nodes equipped with *Intel Xeon Platinum 8480+ (Sapphire Rapids)* CPUs, each providing 112 cores. For details, see https://doku.lrz.de/coolmuc-4-1082337877.html.

B. Strong Scaling

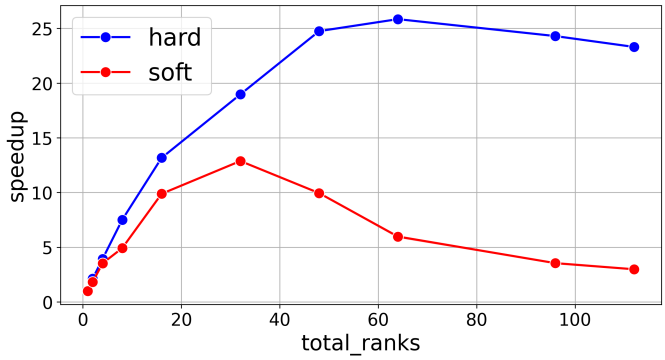
To assess parallel performance, we analyze runtime and speedup as functions of CPU core count. In the strong-scaling tests, colonies grow from a single cell to a radius of 100.

We find that the hard model consistently outperforms the soft model across all core counts, being faster by a median factor of 1.83 (ranging from 1.20 at 1 core to 9.36 at 112 cores, see Figure 14a). This performance advantage likely stems from the hard model's significantly larger stable timesteps, which allow more computational work per communication cycle. As shown in Figure 14b, the hard model achieves reasonable speedup up to approximately 64 cores, while the soft model scales well only up to about 32 cores.

The decrease in performance at higher ranks arises from the domain decomposition. At very high core counts, each rank handles only a narrow angular slice of the colony (see Figure 13), so communication increasingly dominates runtime. With less local work per rank, ghost particle exchanges and global PETSc matrix operations (in the hard model) become more frequent and costly, especially when small timesteps Δt are used. This imbalance reduces scaling efficiency and leads to longer runtimes at high core counts.



(a) Runtime to reach colony radius of 100 versus CPU cores. The hard model is consistently faster, with the performance gap widening at high core counts.



(b) Speedup versus CPU cores. The hard and soft models scale reasonably up to 64 and 32 cores, respectively.

Figure 14: Strong scaling comparisons between hard and soft collision models. (a) Runtime vs. CPU cores. (b) Speedup vs. CPU cores.

C. Adaptive Timestep Analysis

A major factor contributing to the soft model's poor parallel performance is its very small critical timestep, determined by Equation 19. As the colony grows, Δt must be drastically reduced to maintain stable simulations (see Figure 15).

As described in the previous section, tiny timesteps cause each rank to advance particle positions only minimally before initiating costly MPI communication, including ghost particle exchanges and neighbor list reconstruction. As a result, these communication and update operations dominate total simulation time, leading to poor scaling at high core counts.

In contrast, the hard model maintains a mostly constant timestep that is significantly larger than that of the soft model. This allows each rank to advance particles substantially between communication steps, effectively amortizing the cost of MPI exchanges over more computational work. Consequently, the hard model sustains a favorable computation-to-communication ratio, explaining its better scaling at moderate to high core counts.

As shown in Figure 15, the hard model stabilizes around $\Delta t_{\text{hard}} \approx 3 \cdot 10^{-4}$, while the soft model steadily decreases to $\Delta t_{\text{soft}} \approx 10^{-5}$. The ratio $\Delta t_{\text{hard}}/\Delta t_{\text{soft}} \approx 35$ indicates that the hard model can take roughly 35 times larger steps than the soft model in dense colonies, requiring far fewer steps to simulate the same physical time. Notably, for the hard model with $\lambda = 10^{-2}$, the critical timestep even increases over time as interior cell growth slows (see Figure 12) and the overall median velocity u_m decreases.

Using adaptive timestepping provides both stability and efficiency by adjusting the timestep to the colony's current dynamics. A fixed timestep cannot achieve this balance: steps that are too small waste computational effort when interactions are sparse, while steps that are too large can lead to numerical instability in dense regions. This adaptivity enables long-term simulations to proceed efficiently without compromising accuracy or stability.

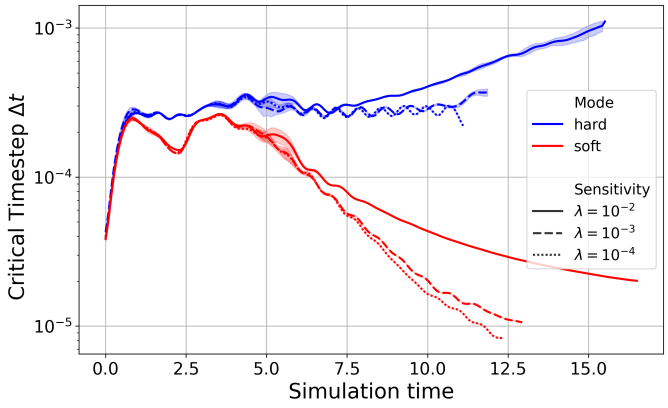


Figure 15: Adaptive timestep Δt for both collision models. The soft model requires a drastic reduction of Δt ($\Delta t_{\text{soft}} \approx 10^{-5}$) to maintain stability, whereas the hard model maintains a significantly larger timestep ($\Delta t_{\text{hard}} \approx 3 \cdot 10^{-4}$), allowing more computation per communication step.

D. Impact of Adaptive Timestep on BBPGD Performance

The efficiency of the BBPGD solver for the hard model also depends critically on the timestep size. The timestep Δt determines both the number of BBPGD iterations per timestep as well as the total number of timesteps required for the simulation. This relationship produces a characteristic U-shaped runtime curve as a function of the CFL number and corresponding timestep, as shown in Figure 16.

The total runtime to reach a colony radius of 100 on 112 CPU cores exhibits a pronounced minimum at CFL ≈ 2, corresponding to cells moving roughly twice the solver tolerance ϵ per timestep. At small CFL values (CFL $\ll 1$), each timestep requires relatively few solver iterations due to accurate linearization (Equation 14) and minimal particle overlaps, but many timesteps are required, making the simulation inefficient. At large CFL values (CFL $\gg 3$), linearization assumptions break down as cells move too far per step, dramatically increasing solver iterations, and in extreme cases (CFL > 3.7), the solver may fail to converge entirely, as indicated by the red shaded region in Figure 16.

The average number of solver iterations per timestep, also plotted in Figure 16, increases roughly linearly with the CFL factor at low values, following iterations $\approx 1800 \cdot$ CFL. At the optimal CFL, around 3600 iterations are required per physical step, after which BBPGD iteration counts rise sharply.

By selecting an appropriate CFL factor, adaptive timestepping maximizes physical progress per unit of computational effort, thereby minimizing total runtime. The timestep, determined by the CFL factor, represents a balance between relying on more constraint solver iterations per step and performing a larger number of total simulation steps. Optimizing this balance allows the hard model to achieve higher performance and improved scaling.

All benchmarks in the previous section used a conservative CFL factor of 0.5, indicating that runtimes could be improved even more by using a larger CFL value closer to the optimal point.

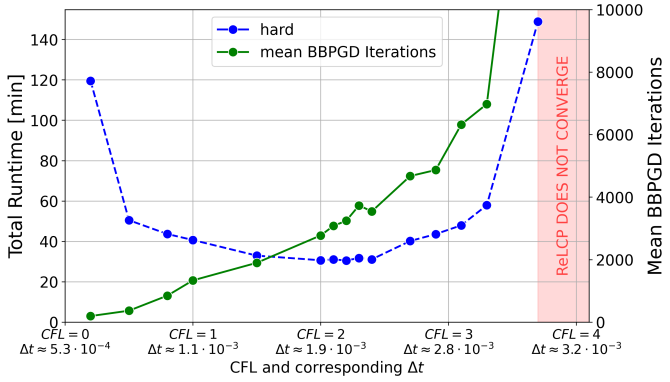


Figure 16: Total runtime to reach a colony radius of 100 versus CFL number on 112 cores. The red region indicates CFL values where the ReLCP procedure fails to converge.

E. Insights into BBPGD Iterations

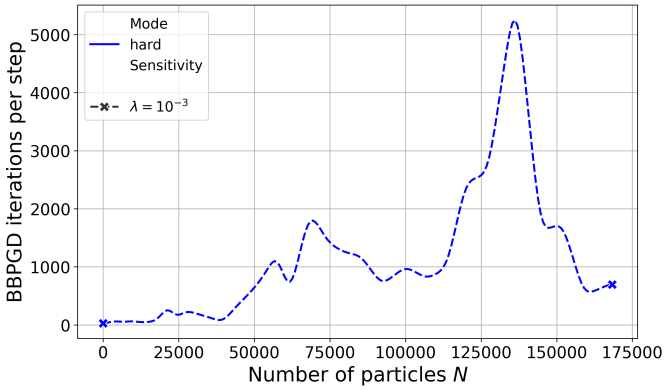
This section examines the performance of the Barzilai-Borwein projected gradient descent (BBPGD) algorithm used in the hard collision model to solve the constrained optimization problem. Since the BBPGD algorithm is the most computationally demanding part of the simulation, understanding its efficiency and convergence is important. In colonies with radius $R \approx 100$, where numerous contact constraints must be satisfied, BBPGD and its associated matrix-vector operations account for roughly 85% of the total simulation time and therefore largely determine overall performance.

A key metric is the number of BBPGD iterations required per simulation step and how this quantity scales with system size. Figure 17a shows that iteration count remains mostly on the order of $\mathcal{O}(1000)$, with occasional peaks reaching $\mathcal{O}(5000)$ during critical moments such as large-scale rearrangements or near jamming transitions. This count includes additional BBPGD calls from the ReLCP [22] procedure, which requires up to 6 iterations to fully resolve remaining overlaps. The typical order of magnitude for the BBPGD iteration count is consistent with related studies [6], even though our model's collision-growth coupling introduces additional complexity.

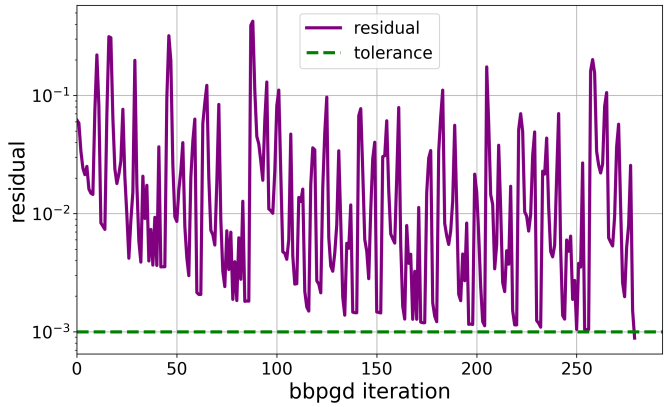
The convergence behavior of a single BBPGD step is illustrated in Figure 17b. The solver reaches the desired tolerance of $\epsilon = 10^{-3}$ after approximately 280 iterations, with the residual directly corresponding to the maximum overlap between any two cells in the colony. The residual exhibits a highly oscillatory pattern, a well-known characteristic of BBPGD resulting from adaptive step-size selection [26], [32].

Since we use the ReLCP procedure [22], each simulation step may involve multiple BBPGD iterations until all overlaps are resolved. The overall energy evolution during this convergence process is shown in Figure 17c, illustrating that the solver steadily reduces the system's energy until the current overlaps are eliminated. Once this state is reached, new contacts are detected, the system's energy increases, and the process repeats. The system ultimately converges to a feasible, overlap-free configuration within six ReLCP iterations, requiring a total of 960 BBPGD steps. The plot also shows the total number of constraints resolved during the BBPGD iterations, which grows with each new ReLCP iteration. For a colony of size $R \approx 100$ with 36,244 cells, roughly 100,000 new constraints are generated per ReLCP iteration, reaching a maximum of about 600,000 constraints in the final phase.

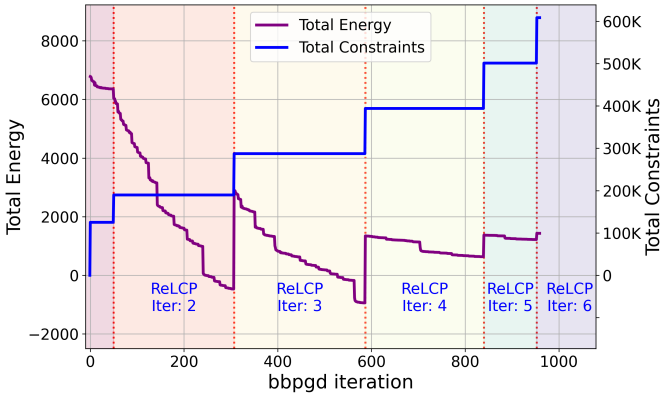
There remains considerable potential to further improve the performance of BBPGD. As demonstrated in subsection VII-D, the timestep size critically influences solver efficiency. Future work could investigate more advanced adaptive schemes that directly leverage solver feedback, for instance by adjusting Δt based on the number of BBPGD iterations or the ReLCP convergence rate. Moreover, warm-start techniques that initialize the solver with multipliers from previous timesteps could help reduce iteration counts.



(a) Number of BBPGD iterations per timestep as a function of particle count. Iterations remain roughly $\mathcal{O}(1000)$ on average, with occasional spikes up to $\mathcal{O}(5000)$.



(b) BBPGD residuals over iterations for a single timestep. The solver converges to tolerance $\epsilon = 10^{-3}$ within approximately 280 iterations, showing highly oscillatory behavior. The residual directly represents the current maximum overlap between any two cells in the colony.



(c) System energy (Equation 16) throughout the ReLCP procedure during a single timestep. The energy consistently decreases as BBPGD iterations progress. After each set of constraints is resolved, new overlaps are detected, and the process repeats until a feasible configuration is reached within six ReLCP iterations.

Figure 17: BBPGD convergence analysis for a typical timestep in a colony of size $R \approx 100$ with 36244 cells. (a) BBPGD iterations per timestep versus number of particles. (b) Residuals over iterations. (c) System energy over iterations.

F. Maximum Attainable Colony Size

In this section, we explore the largest colony sizes that can be simulated within a fixed computational budget of 24 hours on 112 CPU cores of the `CoolMUC-4` cluster, using a stress sensitivity of $\lambda = 10^{-3}$. At these extreme scales, only the hard model is considered, as the soft model suffers from severe overcrowding and excessive cell overlap, which renders its results physically unreliable.

Figure A1 shows a representative snapshot of the hard model colony at its maximum simulated size of $R = 260$, corresponding to approximately 301,116 distinct bacterial cells. The colony displays well-defined concentric density bands and a smooth outer boundary, demonstrating that the solver, contact-handling scheme, and domain decomposition remain stable and physically consistent even at this extreme scale.

The computational cost of reaching such colony sizes is summarized in Figure 18. The wall time grows rapidly with colony radius and closely follows a fifth-degree polynomial fit given by $T_{\text{wall}}\,[\text{h}] \approx 1.94 \cdot 10^{-11}\, R^5 + 0.37$. Since the number of particles scales approximately as $N \propto R^2$, this implies a total runtime scaling of $T_{\text{wall}}\,[\text{h}] \propto N^{2.5}$.

The solver workload is illustrated in Figure 19a, which displays the number of BBPGD iterations per timestep. At this scale, each timestep requires resolving about 5,131,551 contact constraints, distributed across 8 ReLCP phases. The number of BBPGD iterations grows roughly linearly with the number of particles, although two pronounced spikes of up to 25,000 iterations occur during the simulation.

Figure 19b presents the radial distribution of the average cell length at $R = 260$. Clear wave-like patterns with a characteristic wavelength of $\lambda \approx 25$ are observed, corresponding to the ring spacing visible in Figure A1.

Finally, Figure 19c shows the corresponding radial stress distribution. In contrast to the results in Figure 11, the simulated stress profile closely follows the analytical prediction of Weady et al. [22], indicating that at this scale, the continuum assumptions are well satisfied and discretization noise is largely suppressed.

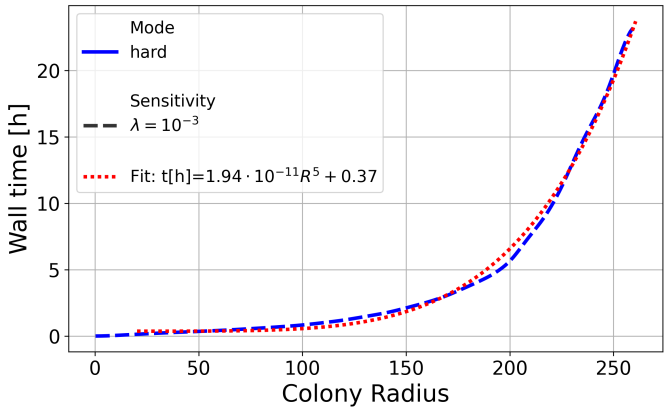
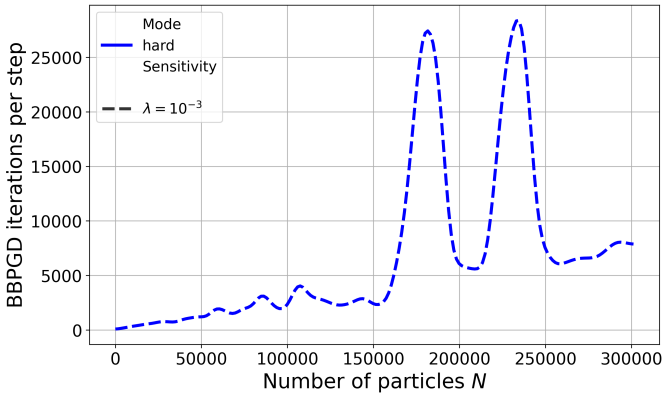
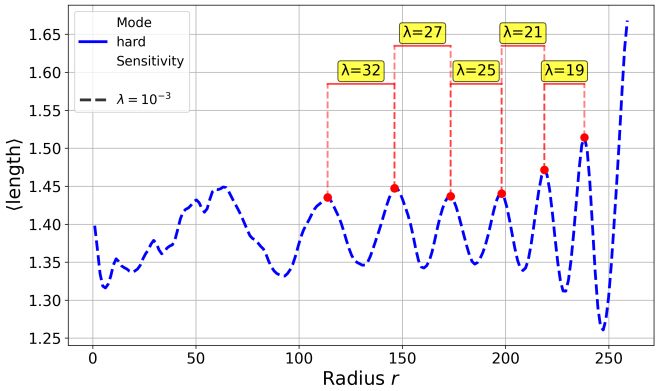


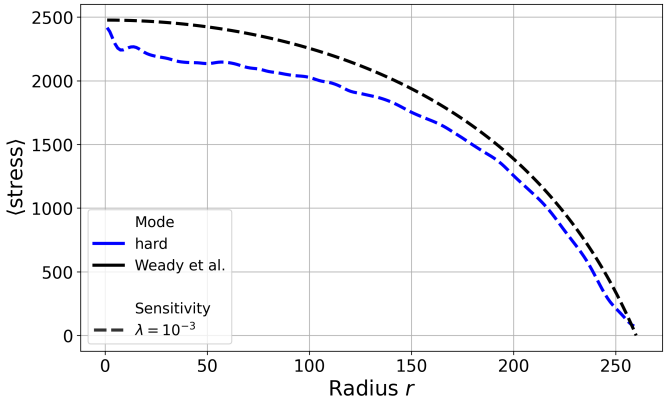
Figure 18: Wall time as a function of colony radius. The wall time roughly follows $T_{\text{wall}}\,[\text{h}] \approx 1.94 \cdot 10^{-11}\, R^5 + 0.37$.



(a) BBPGD iterations per timestep as a function of particle count for the hard model. Each timestep at $R = 260$ resolves about 5,131,551 constraints over 8 ReLCP phases, requiring on average 7,800 iterations, with occasional peaks up to 25,000.



(b) Radial distribution of the average cell length for the hard model at $R = 260$. The oscillatory pattern has a characteristic wavelength of $\lambda \approx 25$, corresponding to the radial ring spacing visible in Figure A1.



(c) Radial stress profile of the hard model at $R = 260$. The simulated stress closely follows the analytical prediction from [22], confirming global mechanical balance across the colony.

Figure 19: Analysis of maximum colony size for the hard model under a fixed 24-hour computational budget. (a) Solver iterations as a function of particle count. (b) Radial oscillations in cell length and ring spacing. (c) Radial stress distribution compared to analytical predictions.

VIII. Discussion and Conclusion

This work presented a systematic comparison of hard (constraint-based) and soft (potential-based) collision models for simulating proliferating cell collectives within a unified computational framework, yielding key insights into the trade-offs between these two approaches.

Both models successfully reproduce the concentric ring patterns and microdomain formation observed in experimental bacterial colonies at small scales, demonstrating that these phenomena emerge from growth-mechanics feedback and are robust to the specific details of collision resolution. This robustness justifies the use of computationally simpler, soft, models for studies focused on colony-level pattern formation at small radii.

At the microscopic level, however, the models differ significantly. The hard model enforces strict non-overlap, maintaining a realistic packing fraction of about 0.9 and producing reliable stress distributions and microdomain structures. In contrast, the soft model allows significant, unphysical overlap, with packing fractions exceeding 5 in colony centers. This leads to distorted microdomain morphology and elongated, unrealistic cell bundles, particularly in large colonies or under low stress-sensitivity conditions. These artifacts limit the soft model's applicability for detailed analyses of cell-scale organization or mechanical stress distributions.

Performance benchmarking shows that the hard model consistently outperforms the soft model across all colony sizes, achieving up to 9.36× faster runtimes on 112 cores. This advantage arises from its ability to use timesteps roughly 30 times larger ($\Delta t_{\text{hard}} \approx 3{\cdot}10^{-4}$ vs. $\Delta t_{\text{soft}} \approx 10^{-5}$), which amortizes communication costs over more computational work. The hard model also exhibits better parallel scaling, maintaining reasonable speedup up to 64 cores, whereas the soft model scales poorly beyond 32 cores. Within a fixed 24-hour computational budget on 112 cores, the hard model simulated colonies up to $R \approx 260$ with approximately 301,116 cells while preserving physical accuracy.

The use of an adaptive timestepping algorithm based on the Courant-Friedrichs-Lewy condition proved essential for numerical stability, dynamically selecting timesteps suited to the colony state. For the hard model, an optimal CFL factor of approximately 2 was identified, balancing solver iterations per step against the total number of steps and minimizing runtime.

In summary, the hard model is strongly preferred for nearly all applications, as it enforces realistic packing densities, yields accurate stress distributions, and provides superior computational performance. The soft model may still be useful for exploratory studies of very small colonies ($R \lesssim 50$) at high stress-sensitivity, where packing artifacts are limited and only pattern formation is of interest. However, its severe physical limitations, including unphysical overcrowding, distorted stress distributions, and artificial cell bundles, make it unsuitable for analysis or large-scale simulations.

IX. Future Work

The computational challenges identified in this work suggest several promising research directions to enhance the scale and efficiency of simulating proliferating cell collectives. While our current implementation provides a solid foundation, significant optimizations remain to exploit modern HPC architectures fully.

A. Improved Parallel Scalability

Communication overhead at high core counts limits scalability for both hard and soft models. Future work should focus on improving communication patterns and domain decomposition to reduce MPI bottlenecks and better balance workloads across ranks. Strategies could include asynchronous communication, overlapping computation with communication, and hierarchical or adaptive decomposition schemes that account for particle density and colony growth. Hybrid MPI+OpenMP parallelism could further reduce communication costs by allowing multiple threads per rank to share memory. However, PETSc does not fully support hybrid operations, so alternative libraries or frameworks may need to be considered.

B. GPU Acceleration via PETSc

For the hard model, the primary computational bottleneck at large scales is the BBPGD solver. Building on the work of [7], future work could leverage PETSc's built-in GPU support to offload linear algebra operations to GPUs with minimal code changes. This approach has been shown to provide significant performance improvements in large-scale simulations where the constraint solver dominates runtime.

C. Addressing Interpenetration in the Soft Model

Future work should explore strategies to mitigate interpenetration in the soft model. Promising approaches include overlap-aware adaptive timestepping, which reduces Δt when overlaps exceed specified thresholds, or hybrid methods that combine soft potentials with periodic constraint projections to remove accumulated overlaps. Additionally, the soft model's pairwise force calculations could benefit from integration with specialized molecular dynamics libraries such as AutoPas [33], [34], which provide optimized neighbor search algorithms, hardware-aware traversal schemes, and automatic parameter tuning.

Acknowledgments

We gratefully acknowledge the computational and data resources, as well as the support provided by the Leibniz Supercomputing Centre (www.lrz.de). All benchmarks in this work were carried out on the `CoolMUC-4` cluster, which is part of the Linux Cluster at LRZ. Further information is available at https://doku.lrz.de/coolmuc-4-1082337877.html.

REFERENCES

[1] S. Weady, B. Palmer, A. Lamson, T. Kim, R. Farhadifar, and M. J. Shelley, "Mechanics and morphology of proliferating cell collectives with self-inhibiting growth," *Phys. Rev. Lett.*, vol. 133, p. 158402, Oct 2024. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.133.158402

[2] R. Wittmann, G. H. P. Nguyen, H. Löwen, F. J. Schwarzendahl, and A. Sengupta, "Collective mechano-response dynamically tunes cell-size distributions in growing bacterial colonies," *Communications Physics*, vol. 6, no. 1, p. 331, 2023. [Online]. Available: https://doi.org/10.1038/s42005-023-01449-w

[3] Y. Yamazaki, T. Ikeda, H. Shimada, and F. Hiramatsu, "Periodic growth of bacterial colonies," *Physica D: Nonlinear Phenomena*, vol. 205, no. 1, pp. 136–153, 2005, synchronization and Pattern Formation in Nonlinear Systems: New Developments and Future Perspectives. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167278905000321

[4] F. A. Bankole, B. Badu-Apraku, A. O. Salami, T. D. O. Falade, R. Bandyopadhyay, and A. Ortega-Beltran, "Variation in the morphology and effector profiles of exserohilum turcicum isolates associated with the northern corn leaf blight of maize in nigeria," *BMC Plant Biology*, vol. 23, no. 1, p. 386, 2023. [Online]. Available: https://doi.org/10.1186/s12870-023-04385-7

[5] M. R. Warren, H. Sun, Y. Yan, J. Cremer, B. Li, and T. Hwa, "Spatiotemporal establishment of dense bacterial colonies growing on hard agar," *eLife*, vol. 8, p. e41093, mar 2019. [Online]. Available: https://doi.org/10.7554/eLife.41093

[6] W. Yan, H. Zhang, and M. J. Shelley, "Computing collision stress in assemblies of active spherocylinders: Applications of a fast and generic geometric method," *The Journal of Chemical Physics*, vol. 150, no. 6, p. 064109, 02 2019. [Online]. Available: https://doi.org/10.1063/1.5080433

[7] A. Tasora, D. Negrut, and M. Anitescu, "Large-scale parallel multi-body dynamics with frictional contact on the graphical processing unit," *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics*, vol. 222, no. 4, pp. 315–326, 2008. [Online]. Available: https://doi.org/10.1243/14644193JMBD154

[8] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified particle physics for real-time applications," *ACM Trans. Graph.*, vol. 33, no. 4, Jul. 2014. [Online]. Available: https://doi.org/10.1145/2601097.2601152

[9] Y. Li, T. Du, K. Wu, J. Xu, and W. Matusik, "Diffcloth: Differentiable cloth simulation with dry frictional contact," *CoRR*, vol. abs/2106.05306, 2021. [Online]. Available: https://arxiv.org/abs/2106.05306

[10] Z. Ferguson, M. Li, T. Schneider, F. Gil-Ureta, T. Langlois, C. Jiang, D. Zorin, D. M. Kaufman, and D. Panozzo, "Intersection-free rigid body dynamics," *ACM Trans. Graph.*, vol. 40, no. 4, Jul. 2021. [Online]. Available: https://doi.org/10.1145/3450626.3459802

[11] T. J. Rudge, P. J. Steiner, A. Phillips, and J. Haseloff, "Computational modeling of synthetic microbial biofilms," *ACS Synthetic Biology*, vol. 1, no. 8, pp. 345–352, 2012. [Online]. Available: https://doi.org/10.1021/sb300031n

[12] A. Blanchard and T. Lu, "Bacterial social interactions drive the emergence of differential spatial colony structures," *BMC systems biology*, vol. 9, p. 59, 09 2015.

[13] P. Ghosh, J. Mondal, E. Ben-Jacob, and H. Levine, "Mechanically-driven phase separation in a growing bacterial colony," *Proceedings of the National Academy of Sciences*, vol. 112, no. 17, pp. E2166–E2173, 2015. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.1504948112

[14] Z. You, D. J. G. Pearce, A. Sengupta, and L. Giomi, "Geometry and mechanics of microdomains in growing bacterial colonies," *Phys. Rev. X*, vol. 8, p. 031065, Sep 2018. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevX.8.031065

[15] M. T. Khan, J. Cammann, A. Sengupta, E. Renzi, and M. G. Mazza, "Toward a realistic model of multilayered bacterial colonies," *Condensed Matter Physics*, vol. 27, no. 1, p. 13802, Mar. 2024. [Online]. Available: http://dx.doi.org/10.5488/CMP.27.13802

[16] Z. You, D. J. G. Pearce, and L. Giomi, "Confinement-induced self-organization in growing bacterial colonies," *Science Advances*, vol. 7, no. 4, Jan. 2021. [Online]. Available: http://dx.doi.org/10.1126/sciadv.abc8685

[17] A. Valdez, H. Sun, H. H. Weiss, and I. Aranson, "Biomechanical modeling of spatiotemporal bacteria-phage competition," *Communications Physics*, vol. 8, no. 1, p. 139, 2025. [Online]. Available: https://doi.org/10.1038/s42005-025-02078-1

[18] T. J. Rudge, F. Federici, P. J. Steiner, A. Kan, and J. Haseloff, "Cell polarity-driven instability generates self-organized, fractal patterning of cell layers," *ACS Synthetic Biology*, vol. 2, no. 12, pp. 705–714, 2013, pMID: 23688051. [Online]. Available: https://doi.org/10.1021/sb400030p

[19] B. Langeslay and G. Juarez, "Microdomains and stress distributions in bacterial monolayers on curved interfaces," *Soft Matter*, vol. 19, no. 20, p. 36053613, 2023. [Online]. Available: http://dx.doi.org/10.1039/D2SM01498J

[20] D. Eberly, "Robust computation of distance between line segments," https://www.geometrictools.com/, 2018, https://www.geometrictools.com/GTE/Applications/Application.h. [Online]. Available: https://www.geometrictools.com/Documentation/DistanceLine3Line3.pdf

[21] S. S. Datta, "Life at low reynolds number isn't such a drag," 2024. [Online]. Available: https://arxiv.org/abs/2410.20648

[22] S. Weady, B. Palmer, A. Lamson, T. Kim, R. Farhadifar, and M. J. Shelley, "Supplementary material: Mechanics and morphology of proliferating cell collectives with self-inhibiting growth," New York, NY, USA, 2024.

[23] W. Yan, S. Ansari, A. Lamson, M. A. Glaser, R. Blackwell, M. D. Betterton, and M. Shelley, "Toward the cellular-scale simulation of motor-driven cytoskeletal assemblies," *eLife*, vol. 11, p. e74160, may 2022. [Online]. Available: https://doi.org/10.7554/eLife.74160

[24] T. J. Rudge, P. J. Steiner, A. Phillips, and J. Haseloff, "Mathematical description of cellmodeller," https://github.com/cellmodeller/CellModeller/blob/master/Doc/Maths/math.pdf, 2015.

[25] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. New York, NY: Springer, 2006.

[26] Y.-H. Dai and R. Fletcher, "Projected barzilai–borwein methods for large-scale box-constrained quadratic programming," *Numerische Mathematik*, vol. 100, pp. 21–47, March 2005, received 17 October 2003, Revised 19 August 2005, Published 16 February 2005. [Online]. Available: https://doi.org/10.1007/s00211-004-0569-y

[27] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, and H. Zhang, "PETSc Web page," http://www.mcs.anl.gov/petsc, 2015. [Online]. Available: http://www.mcs.anl.gov/petsc

[28] R. Courant, K. Friedrichs, and H. Lewy, "Über die partiellen differenzengleichungen der mathematischen physik," *Mathematische Annalen*, vol. 100, no. 1, pp. 32–74, 1928. [Online]. Available: https://doi.org/10.1007/BF01448839

[29] J. Ahrens, B. Geveci, and C. Law, "Paraview: An end-user tool for large data visualization," in *Visualization Handbook*. Elsevier, 2005.

[30] O. Hallatschek, P. Hersen, S. Ramanathan, and D. R. Nelson, "Genetic drift at expanding frontiers promotes gene segregation," *Proceedings of the National Academy of Sciences*, vol. 104, no. 50, pp. 19 926–19 930, 2007. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.0710150104

[31] A. Giometto, D. R. Nelson, and A. W. Murray, "Physical interactions reduce the power of natural selection in growing yeast colonies," *Proceedings of the National Academy of Sciences*, vol. 115, no. 45, pp. 11 448–11 453, 2018. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.1809587115

[32] M. Schneider, "On non-stationary polarization methods in fft-based computational micromechanics," *International Journal for Numerical Methods in Engineering*, vol. 122, no. 22, pp. 6800–6821, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.6812

[33] F. A. Gratl, S. Seckler, N. Tchipev, H.-J. Bungartz, and P. Neumann, "Autopas: Auto-tuning for particle simulations," in *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2019, pp. 748–757.

[34] S. J. Newcome, F. A. Gratl, P. Neumann, and H.-J. Bungartz, "Towards auto-tuning multi-site molecular dynamics simulations with autopas," *Journal of Computational and Applied Mathematics*, vol. 433, p. 115278, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377042723002224

APPENDIX

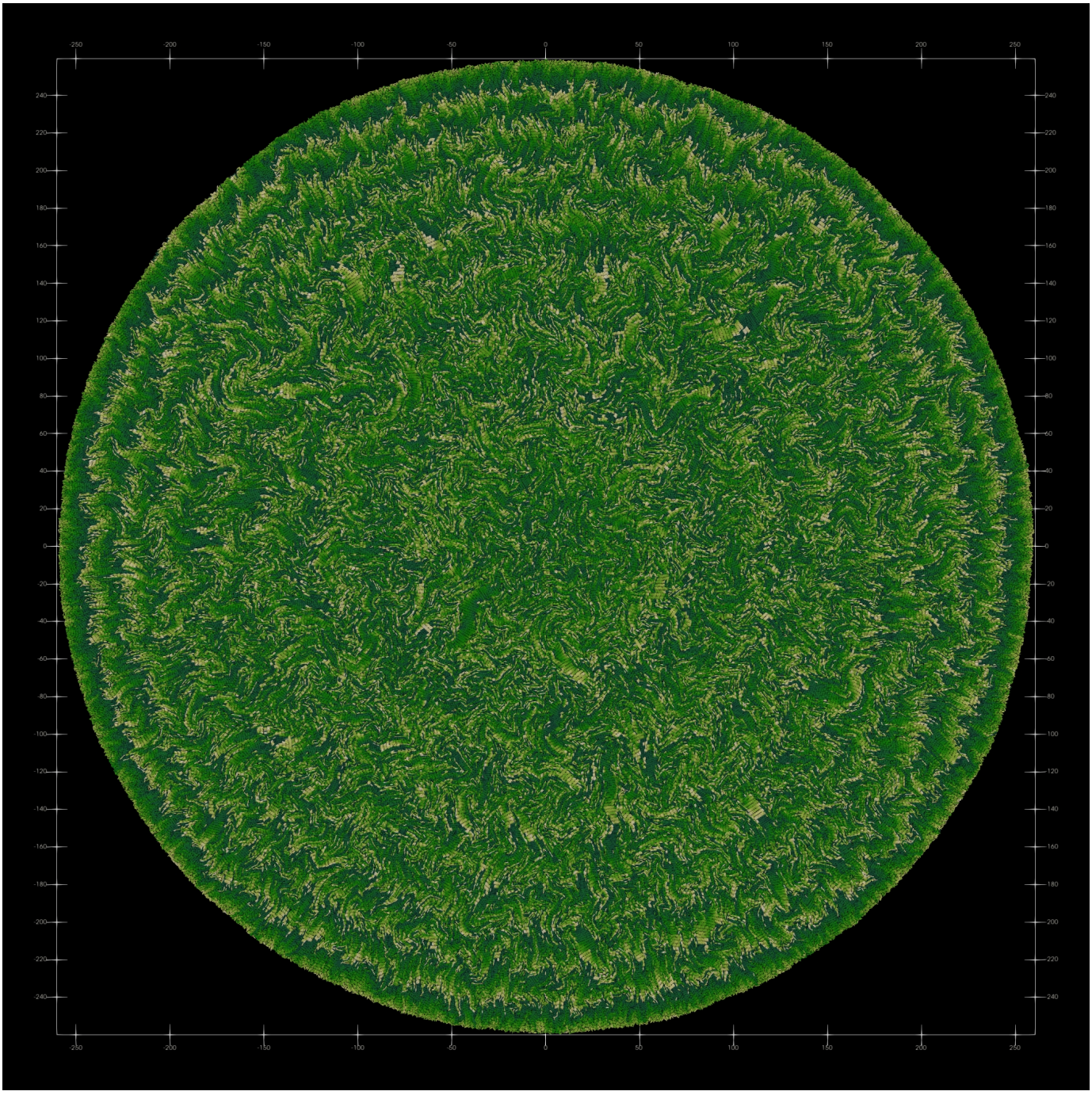


Figure A1: Snapshot of largest attainable colony size using the hard collision model with $\lambda = 10^{-3}$ after 24h of wall time on 112 ranks on the CoolMUC-4 cluster. The colony has a radius of approximately $R \approx 260$ and consists of 301,116 cells. The color indicates the length of the cells (short cells are darker, longer cells are lighter). Clear concentric ring patterns are visible, similar to those observed in smaller colonies (See Figure 4).

SUPPLEMENTARY MATERIALS

All high-resolution figures and supplementary videos are available online at https://home.cit.tum.de/~ler/bacteria/.