Feasibility and spatial heterogeneity of lineage conversion processes

Carlos Mendonca[1] Shloka Janapaty[1] Michael Tulskikh[1]

[1]Department of Applied Mathematics,
Columbia University

Dec 11, 2023

Who are we?

Carlos Mendonca

- CC
- Interested in developmental and molecular embryogenesis.

Shloka Janapaty

- SEAS
- Interested in theoretical ecology and ecological state transitions.

Michael Tulskikh

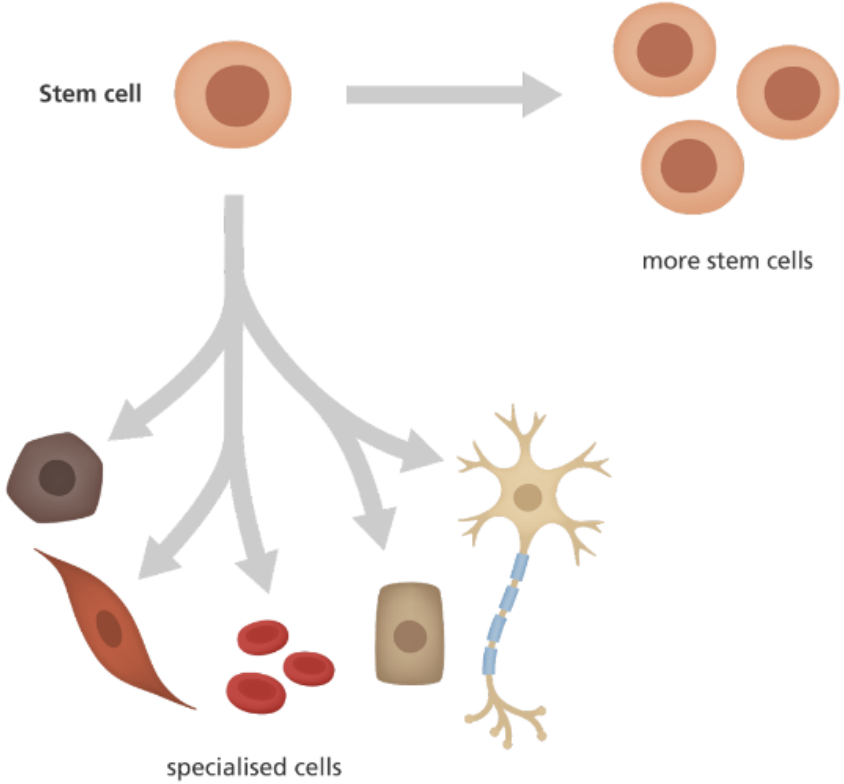
- SEAS
- Interested in PDEs and differential geometry.

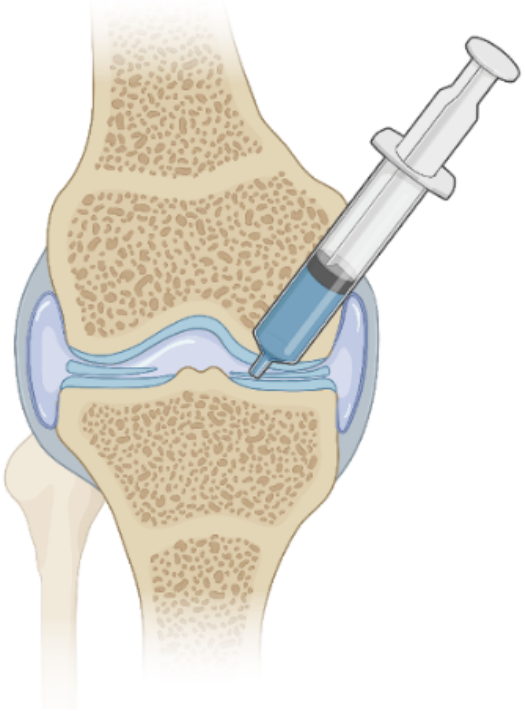
Primary References

1. Joo, J. I., Zhou, J. X., Huang, S., & Cho, K. H. (2018). **Determining relative dynamic stability of cell states using boolean network model.** *Nature Scientific Reports*, 8(1), 12077.
2. Gonze, D., Lahti, L., Raes, J., Faust, K. (2017). **Multi-stability and the origin of microbial community types.** *The ISME journal*, 11(10), 2159-2166.
3. Lu, M., & Hedin, L. O. (2019). **Global plant–symbiont organization and emergence of biogeochemical cycles resolved by evolution-based trait modelling.** *Nature Ecology & Evolution*, 3(2), 239-250.

Table of Contents

What is a Stem Cell[1]



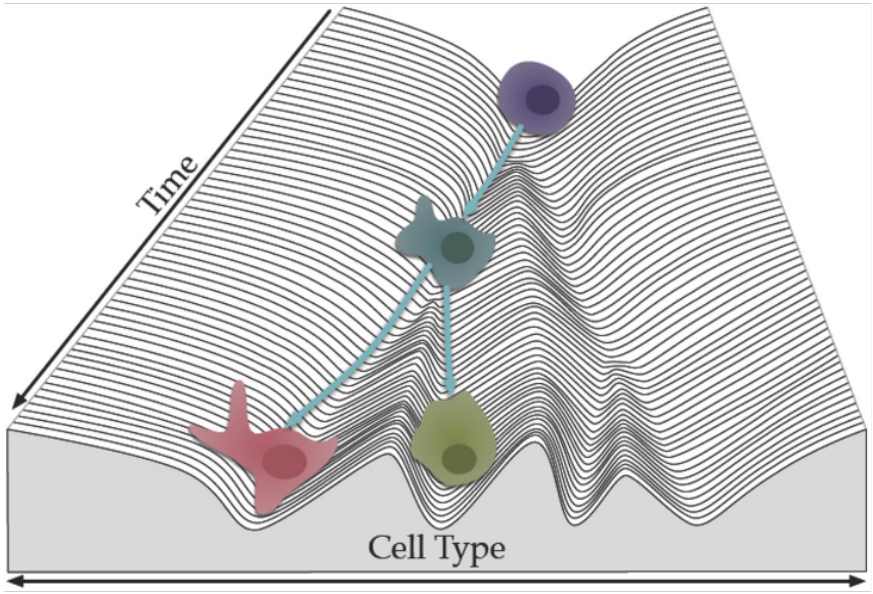


Effective therapy for cartilage repair in osteoarthritis

[1] Liu et al. (2021), *Gels*

Waddington's landscape of cell differentiation[2]

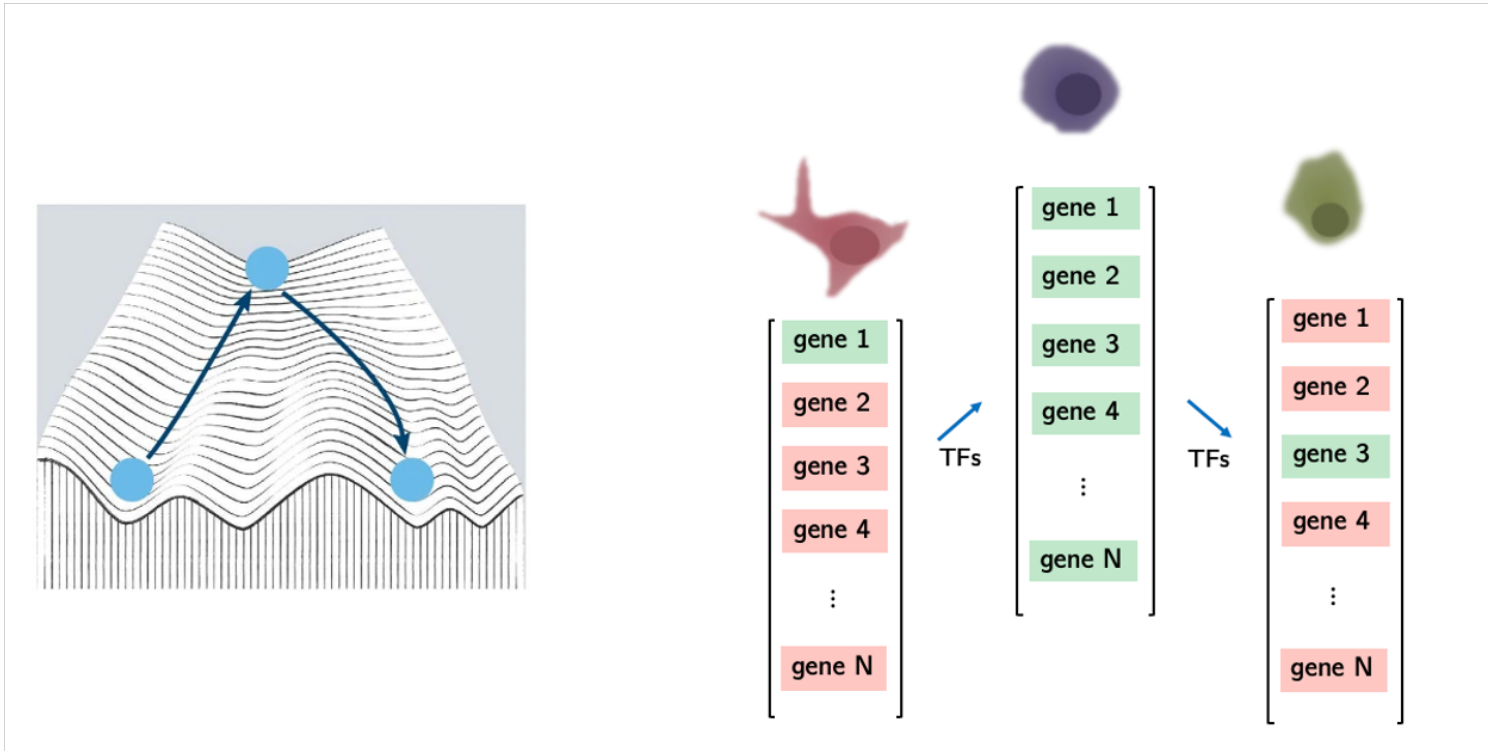
- Epigenetic modifications cause stem cells to descend from regions of high potency to stable, differentiated cell types



[2] Zwiessele and Lawrence. (2021), *BioRXiv*

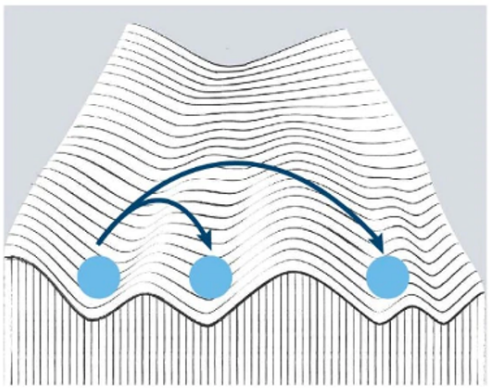
Engineering specific cell types: Pluripotency Reprogramming

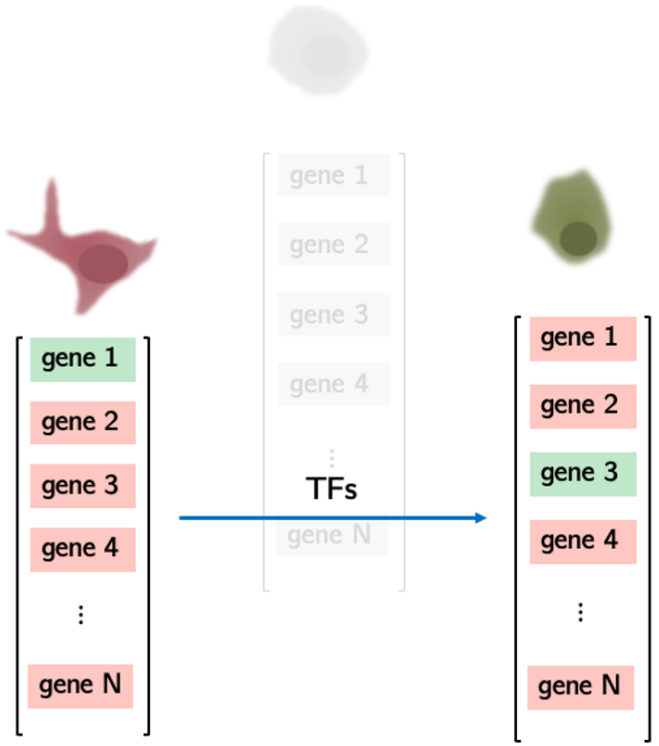
- Achieved through an increase in potential to pluripotent status followed by directed programming to the desired cell fate
- **Problematic due to proliferation of pluripotent cells and risk of off-target tumorigenesis**



Engineering specific cell types: Lineage Conversion

- Lineage conversion directs the donor cell type directly to the target cell type, circumventing the problematic pluripotent stage





Lineage Conversion

Advantages

- Patient specific cell generation
- Bypasses the pluripotent state, thereby decreasing risks of off-target tumorigenesis

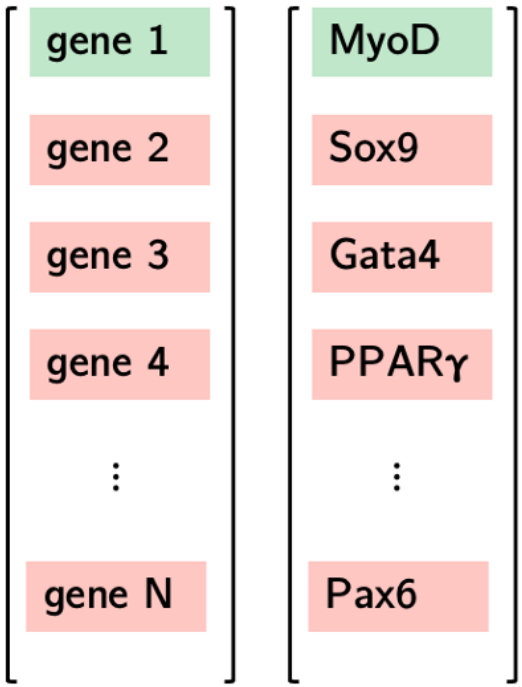
Disadvantages

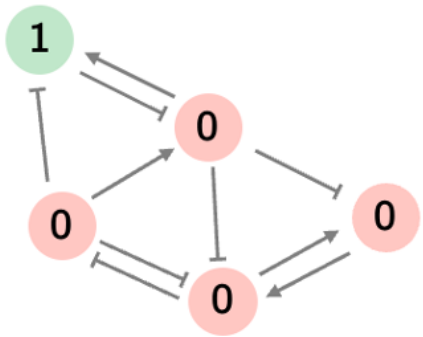
- Limited mechanistic understanding and laborious search for transcription factor cocktail
- Can lead to heterogeneous target tissues

1. How can we better inform the search for novel lineage conversion pathways?
2. Can we develop simple models of differentiation and conversion to characterize spatial heterogeneity of target tissues?

Network of Master Regulatory Genes

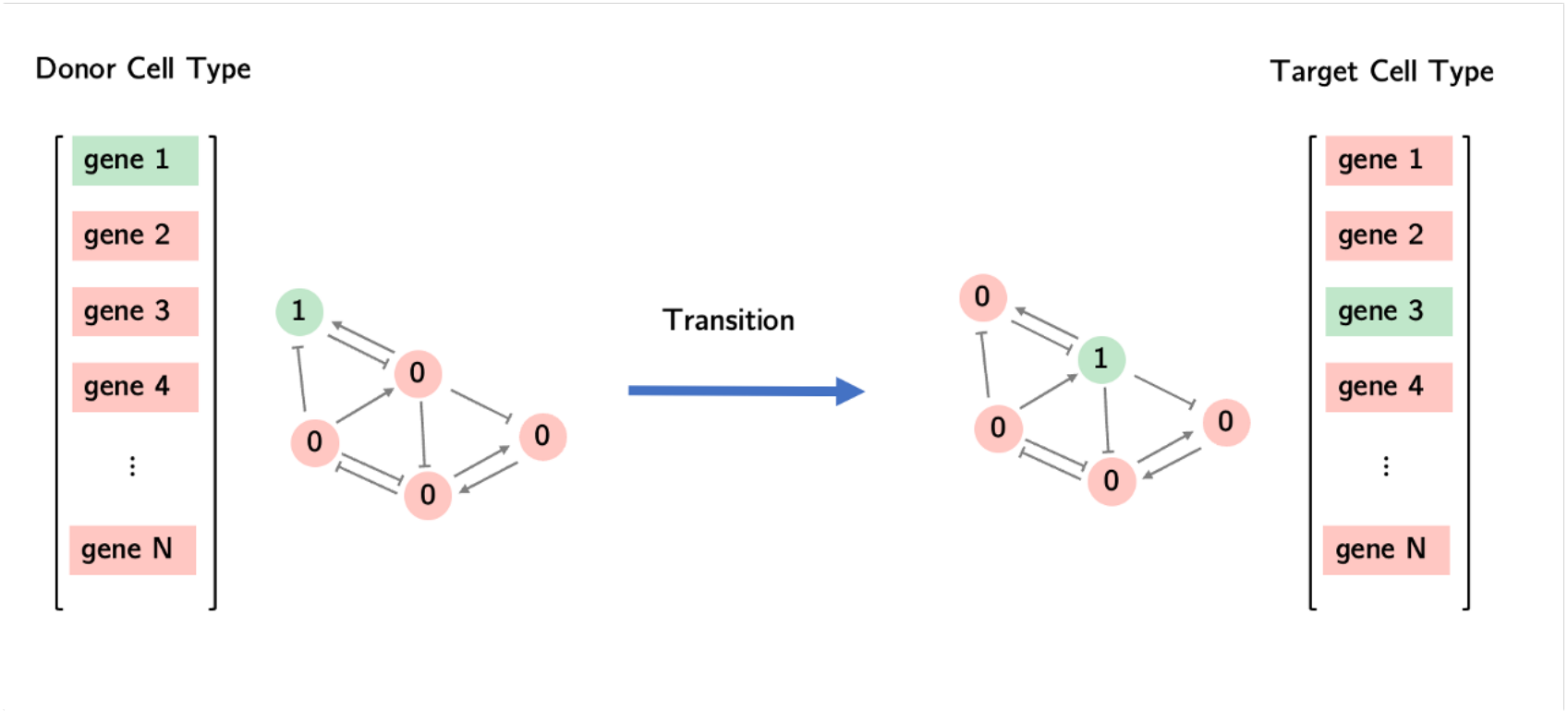
- Inhibiting and activating relationships between interacting genes can be represented in a boolean network





Lineage conversion is transition between stable states

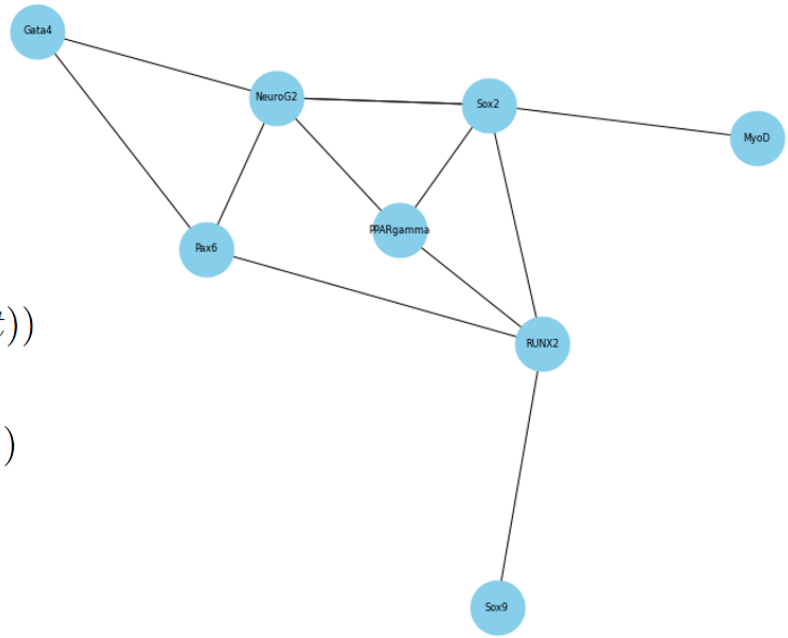
- Lineage conversion can be considered flux which forces the gene regulatory network to transition to a new stable state



Generation of Biologically Relevant Updating Functions

- Biochemical and molecular assays can inform logical relationships between nodes in a gene regulatory network [3]

$$
\begin{aligned}
\mathrm{MyoD}(t+1) &= \mathrm{MyoD}(t) \wedge \mathrm{Sox2}(t) \\
\mathrm{Sox9}(t+1) &= \mathrm{Sox9}(t) \vee \mathrm{RUNX2}(t) \\
\mathrm{Gata4}(t+1) &= \mathrm{Gata4}(t) \wedge \mathrm{NeuroG2}(t) \\
\mathrm{PPAR}\gamma(t+1) &= \mathrm{PPAR}\gamma(t) \vee (\mathrm{NeuroG2}(t) \wedge \neg \mathrm{Sox2}(t)) \\
\mathrm{Sox2}(t+1) &= \mathrm{Sox2}(t) \wedge \neg \mathrm{NeuroG2}(t) \\
\mathrm{RUNX2}(t+1) &= \mathrm{RUNX2}(t) \vee (\mathrm{PPAR}\gamma(t) \wedge \neg \mathrm{Sox2}(t)) \\
\mathrm{NeuroG2}(t+1) &= \mathrm{NeuroG2}(t) \wedge \mathrm{Pax6}(t) \wedge \neg \mathrm{Sox2}(t) \\
\mathrm{Pax6}(t+1) &= \mathrm{Pax6}(t) \wedge (\mathrm{RUNX2}(t) \vee \neg \mathrm{Gata4}(t))
\end{aligned}
$$

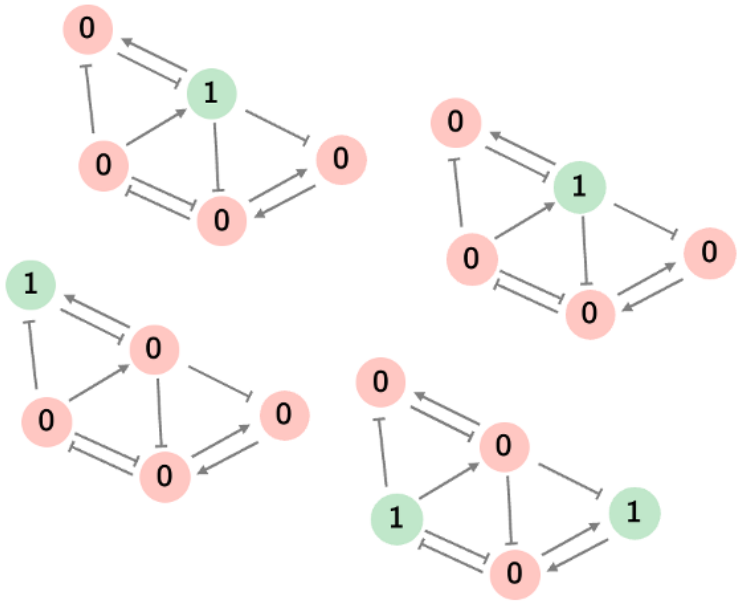


[3] See references on Frame 18-19.

Network of master regulatory genes

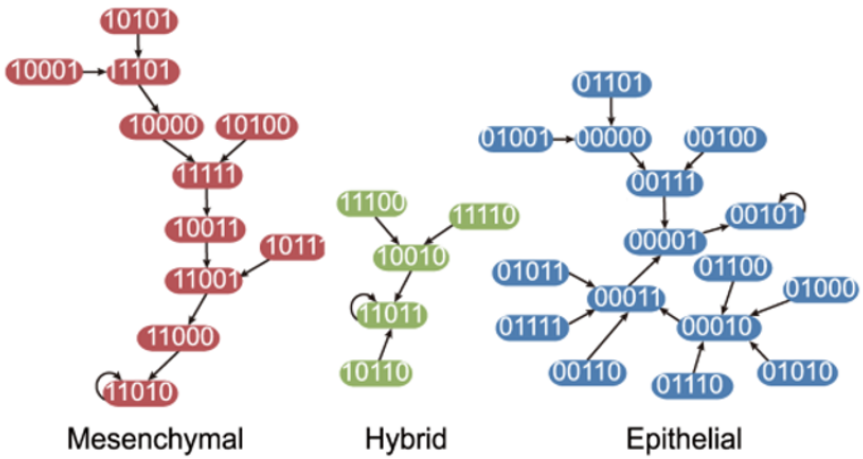
- Evolving the dynamical system generates biologically relevant stable states including adipose, retinal, and cardiac.

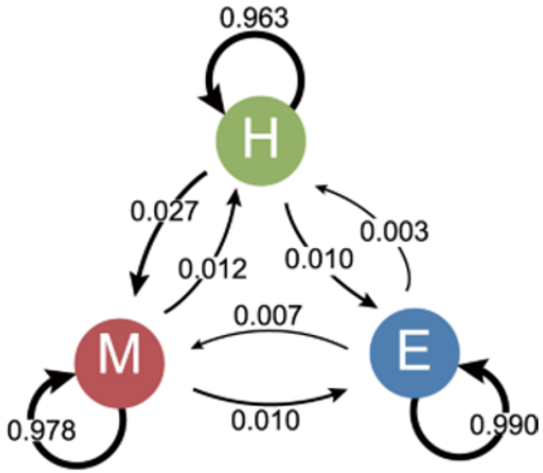
$$
\begin{aligned}
\mathbf{MyoD(t+1)} &= \mathbf{MyoD(t)} \ \wedge \mathbf{Sox2}(t) \\
\text{Sox9}(t+1) &= \text{Sox9}(t) \vee \text{RUNX2}(t) \\
\text{Gata4}(t+1) &= \text{Gata4}(t) \wedge \text{NeuroG2}(t) \\
\text{PPAR}\gamma(t+1) &= \text{PPAR}\gamma(t) \vee (\text{NeuroG2}(t) \wedge \neg\text{Sox2}(t)) \\
\text{Sox2}(t+1) &= \text{Sox2}(t) \wedge \neg\text{NeuroG2}(t) \\
\text{RUNX2}(t+1) &= \text{RUNX2}(t) \vee (\text{PPAR}\gamma(t) \wedge \neg\text{Sox2}(t)) \\
\text{NeuroG2}(t+1) &= \text{NeuroG2}(t) \wedge \text{Pax6}(t) \wedge \neg\text{Sox2}(t) \\
\text{Pax6}(t+1) &= \text{Pax6}(t) \wedge (\text{RUNX2}(t) \vee \neg\text{Gata4}(t))
\end{aligned}
$$



Network of master regulatory genes[4]

- Joo et. al use Monte Carlo methods to calculate basin transition rates between skin cells, transformed cancerous cells, and transitional cells
- We leverage the same technique in determining transition rate between cell states in lineage conversion processes





[4] Joo et. al. (2018), *Nature*

Monte Carlo Simulation for Transition Probabilities

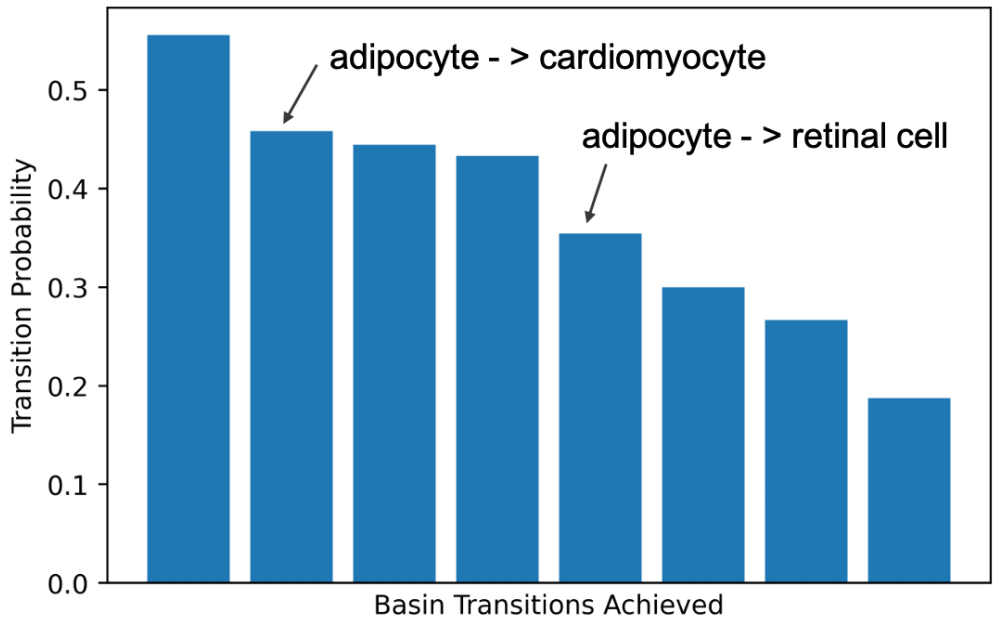
Algorithm Monte Carlo Simulation

Initialize empty transition counts
for i from $i = 0$ to $i = numsamples$ **do**
    Randomly initialize initial states.
    Find stable state $initial\_basin$ by evolving system until attractor is reached.
    if $initial\_basin$ is found **then**
        Perturb the attractor by randomly changing two elements and evolve to the new attractor $final\_state$.
        if $final\_states$ is stable **then**
            Increment transition count from $initial\_basin$ to $final\_states$.

            Calculate transition probabilities from transition counts.

Example Simulation Results

- Transition probabilities for eight different lineage conversion pathways were identified in the simulation
- Several of these, including adipocyte -> cardiomyocyte are non-canonical lineage conversion pathways
- High theoretical transition probabilities may indicate feasibility of transition and motivation for transcription factor search



Simulation Results

References

1. Tapscott, S. J., Davis, R. L., Lassar, A. B., & Weintraub, H. (1990). **MyoD: a regulatory gene of skeletal myogenesis.** Myoblast Transfer Therapy, 3-6.
2. Shi, G., Sohn, K. C., Li, Z., Choi, D. K., Park, Y. M., Kim, J. H. & Lee, J. H. (2013). **Expression and functional role of Sox9 in human epidermal keratinocytes.** PloS one, 8(1), e54355.
3. Lefterova, M. I., Zhang, Y., Steger, D. J., Schupp, M., Schug, J., Cristancho, A., ... & Lazar, M. A. (2008). **PPARγ and C/EBP factors orchestrate adipocyte biology via adjacent binding on a genome-wide scale.** Genes & development, 22(21), 2941-2952.
4. Noack, F., Vangelisti, S., Raffl, G., Carido, M., Diwakar, J., Chong, F., & Bonev, B. (2022). **Multimodal profiling of the transcriptional regulatory landscape of the developing mouse cortex identifies Neurog2 as a key epigenome remodeler.** Nature Neuroscience, 25(2), 154-167.

References

1. Grocott, T., Lozano-Velasco, E., Mok, G. F., & Münsterberg, A. E. (2020). **The Pax6 master control gene initiates spontaneous retinal development via a self-organising Turing network.** Development, 147(24), dev185827.
2. Sonne, S. B., Perrett, R. M., Nielsen, J. E., Baxter, M. A., Kristensen, D. M., Leffers, H. & Rajpert-de-Meyts, E. (2010). **Analysis of SOX2 expression in developing human testis and germ cell neoplasia.** The International journal of developmental biology, 54(4), 755.
3. Sánchez-Luis, E., Joaquín-García, A., Campos-Laborie, F. J., Sánchez-Guijo, F., & De las Rivas, J. (2020). **Deciphering master gene regulators and associated networks of human mesenchymal stromal cells.** Biomolecules, 10(4), 557.

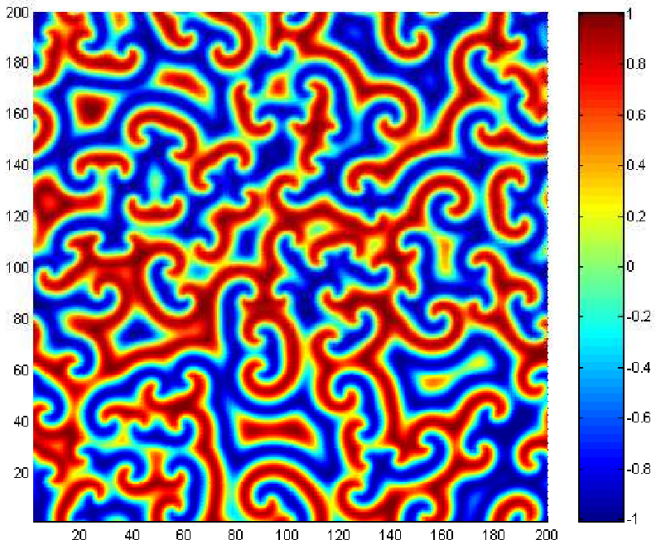
Table of Contents

Introduction to Reaction-Diffusion Models

- **Concept:** Models that describe how the concentration of one or more substances distributed in space changes under the influence of two processes: reaction (chemical) and diffusion (spatial).
- **Applications:** Widely used in various fields such as physics, chemistry, biology, and ecology.
- **Analysis:** Used to understand how small perturbations in a system evolve over time, as well as how certain patterns arise.

Mathematical Formulation of Reaction-Diffusion Models

- **General Form:** $\frac{\partial u}{\partial t} = D\nabla^2 u + f(u)$, where:
  - u represents the concentration of a substance.
  - D is the diffusion coefficient.
  - $\nabla^2 u$ is the Laplacian, representing diffusion.
  - $f(u)$ represents the reaction component.
- **Example:** The Ginzburg–Landau [5] equation $\frac{\partial A}{\partial t} = \nabla^2 A + A - A|A|^2$



[5] Kassam 2003, Oxford Research Archive

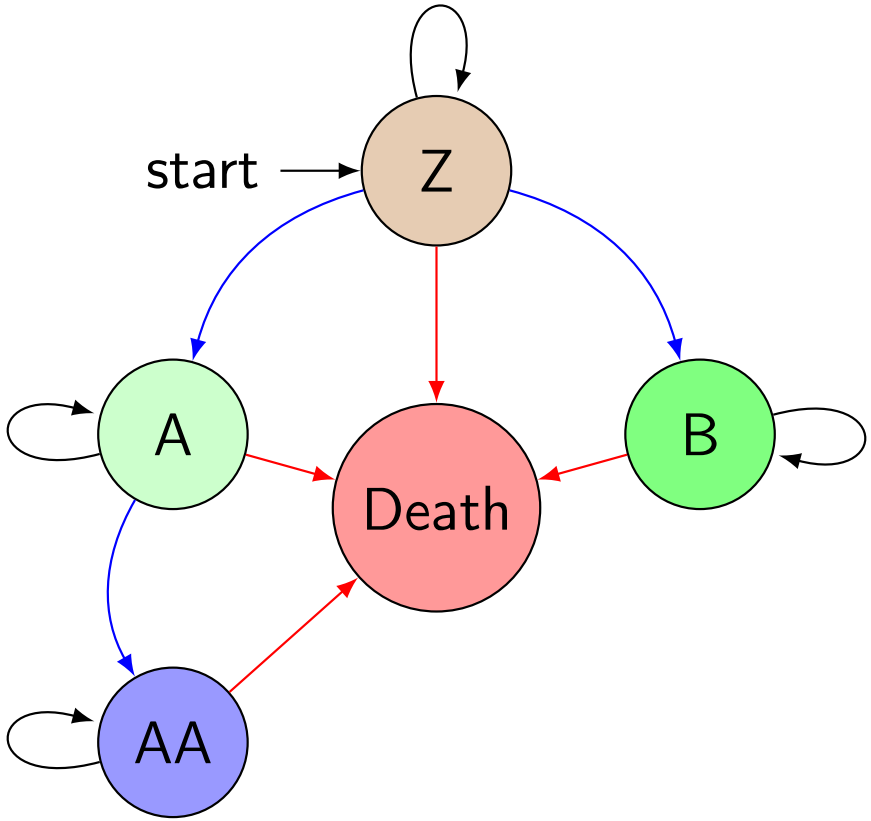
Reaction-Diffusion Models in Biological Systems

Relevance in Biological Systems:

- Explain spatial and temporal patterns in biological systems, such as cellular processes.
- Model the dynamics of population distributions.

Table of Contents

Case 1: Differentiation, Apoptosis [6]



[6] Morris 2016, *Development*

Model 1 Equations

$$\frac{\partial Z}{\partial t} = \text{Growth} - \text{Death} - \text{Diff (Loss)} + \nabla^2 \tag{1}$$

$$\frac{\partial A}{\partial t} = \text{Growth} - \text{Death} + \text{Diff (Gain)} - \text{Diff (Loss)} + \nabla^2 \tag{2}$$

$$\frac{\partial B}{\partial t} = \text{Growth} - \text{Death} + \text{Diff (Gain)} + \nabla^2 \tag{3}$$

$$\frac{\partial AA}{\partial t} = \text{Growth} - \text{Death} + \text{Diff (Gain)} + \nabla^2 \tag{4}$$

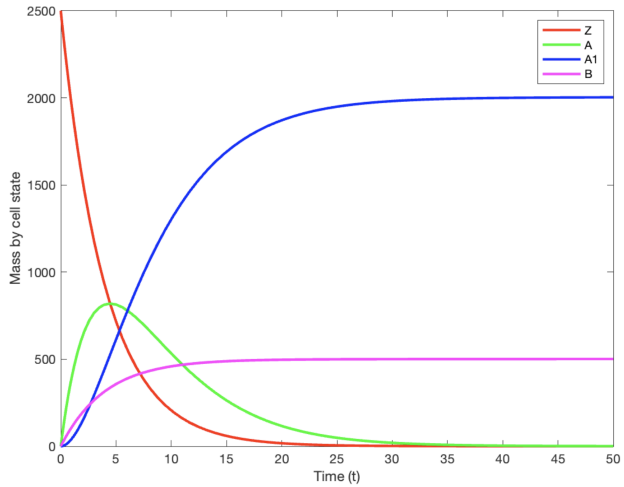
Model 1 Equations

$$\frac{\partial Z}{\partial t} = g_Z - d_Z Z - (p_{ZA} + p_{ZB})Z + D_Z \nabla^2 Z \tag{1}$$

$$\frac{\partial A}{\partial t} = g_A - d_A A + p_{ZA} Z - p_{AAA} A + D_A \nabla^2 A \tag{2}$$

$$\frac{\partial B}{\partial t} = g_B - d_B B + p_{ZB} Z + D_B \nabla^2 B \tag{3}$$

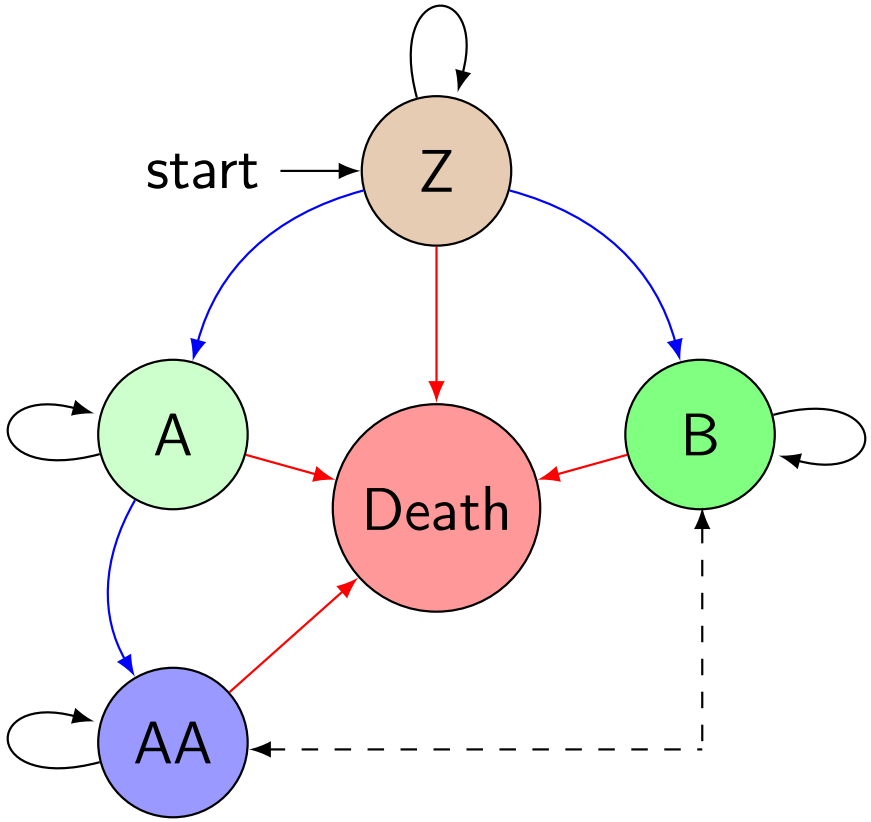
$$\frac{\partial AA}{\partial t} = g_{AA} - d_{AA} AA + p_{AAA} A + D_{AA} \nabla^2 AA \tag{4}$$



Time evolution of (1) differentiation, (2) apoptosis in Model 1

Table of Contents

Case 2: Differentiation, Apoptosis, Lineage Conversion



Model 2 Equations

$$\frac{\partial Z}{\partial t} = \text{Growth} - \text{Death} - \text{Diff (Loss)} + \nabla^2 \quad (1)$$

$$\frac{\partial A}{\partial t} = \text{Growth} - \text{Death} + \text{Diff (Gain)} - \text{Diff (Loss)} + \nabla^2 \quad (2)$$

$$\frac{\partial B}{\partial t} = \text{Growth} - \text{Death} + \text{Diff (Gain)} + \text{LC (Gain)} - \text{LC (Loss)} + \nabla^2 \quad (3)$$

$$\frac{\partial AA}{\partial t} = \text{Growth} - \text{Death} + \text{Diff (Gain)} + \text{LC (Gain)} - \text{LC (Loss)} + \nabla^2 \quad (4)$$

$$\frac{\partial t_{AA}}{\partial t} = -\text{Decay} + \nabla^2 \quad (5)$$

$$\frac{\partial t_B}{\partial t} = -\text{Decay} + \nabla^2 \quad (6)$$

Model 2 Equations

$$\frac{\partial Z}{\partial t} = g_Z - d_Z Z - (p_{ZA} + p_{ZB})Z + D_Z \nabla^2 Z \quad (1)$$

$$\frac{\partial A}{\partial t} = g_A - d_A A + p_{ZA} Z - p_{AAA} A + D_A \nabla^2 A \quad (2)$$

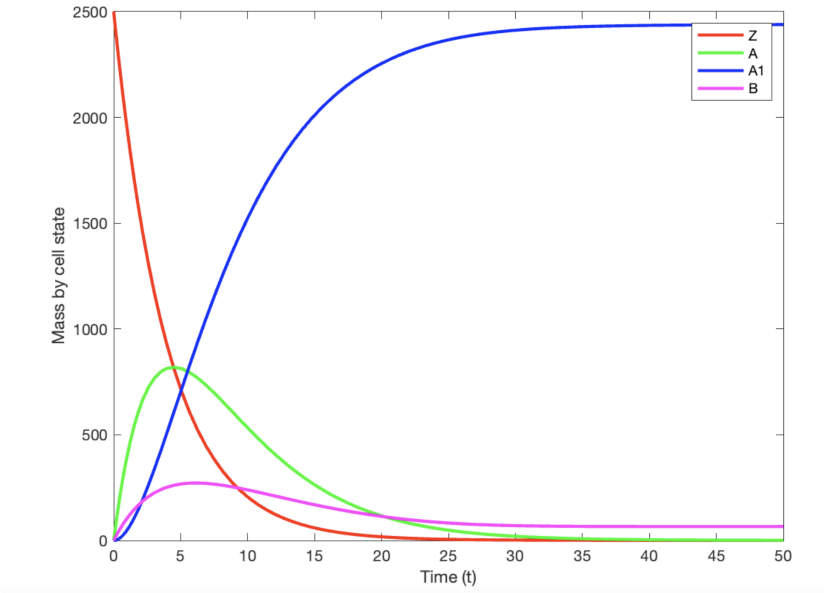
$$\frac{\partial B}{\partial t} = g_B - d_B B + p_{ZB} Z + \mu_B \frac{t_B}{k_B + t_B} AA - \mu_{AA} \frac{t_{AA}}{k_{AA} + t_{AA}} B + + D_B \nabla^2 B \quad (3)$$

$$\frac{\partial AA}{\partial t} = g_{AA} - d_{AA} AA + p_{AAA} A + \mu_{AA} \frac{t_{AA}}{k_{AA} + t_{AA}} B - \mu_B \frac{t_B}{k_B + t_B} AA + D_{AA} \nabla^2 AA \quad (4)$$

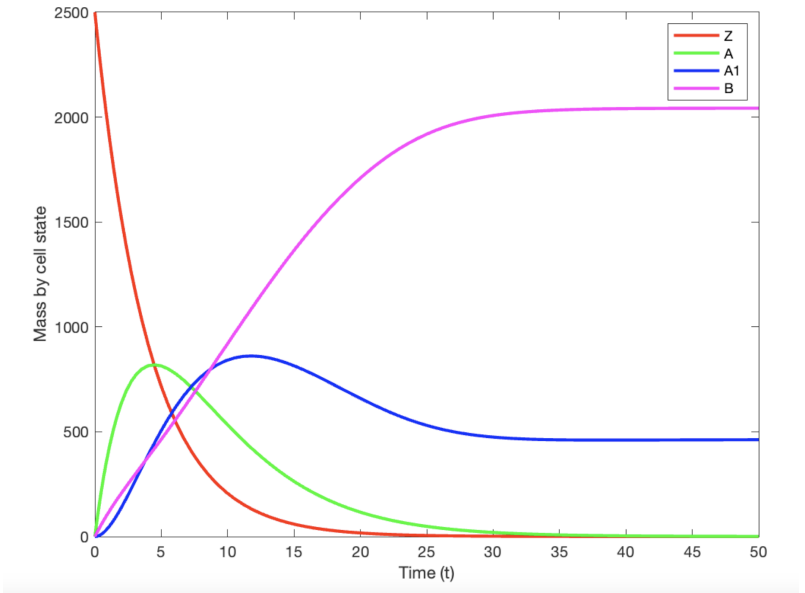
$$\frac{\partial t_{AA}}{\partial t} = - q_{AA} t_{AA} + D_{t\,AA} \nabla^2 t_{AA} \quad (5)$$

$$\frac{\partial t_B}{\partial t} = - q_B t_B + D_{t\,B} \nabla^2 t_B \quad (6)$$

Case 2: Time Evolution



(a) B to AA Lineage Conversion



(b) AA to B Lineage Conversion

(1) Differentiation, (2) Apoptosis, (3) Lineage Conversion

Table of Contents

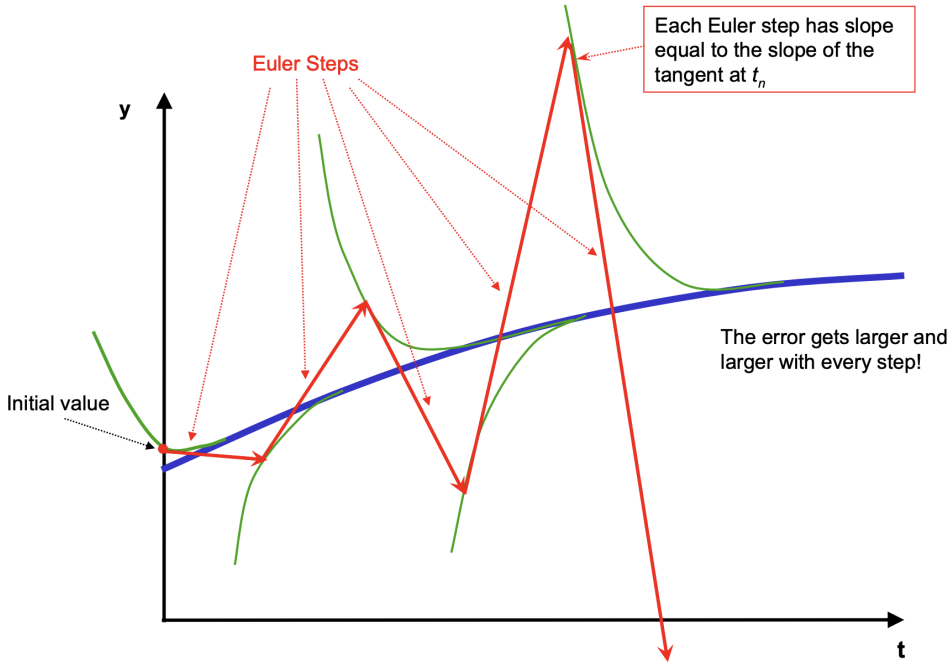
Solving Reaction-Diffusion Equations Numerically

Diffusion equations are numerically **stiff**. [7]

[7] Gear 2007, Numerical Methods for Evolutionary Systems (Lecture 2).

Solving Diffusion Equations Numerically

Diffusion equations are numerically **stiff**.



A stiff PDE is numerically unstable unless the step size is extremely small. [8]

[8] Gear 2007, Numerical Methods for Evolutionary Systems (Lecture 2).

Why is Diffusion stiff?

Diffusive processes spread across space and time with decaying amplitude.

$$u_t = \alpha u_{xx} \tag{1}$$

For example, (1) admits a solution of the form:

$$u(x,t) = Qe^{-at}\sin(kx) \tag{2}$$

A counterpart to (2) is the complex representation of these functions, which can be added to generate a Fourier representation of a general solution of the diffusion equation given in (4),

$$u(x,t) = Qe^{-at}e^{ikx} \tag{3}$$

$$u(x,t) \approx \sum_{k\in K} b_k e^{-\alpha k^2 t} e^{ikx} \tag{4}$$

where K is a set of an infinite k-values.

Why is Diffusion Stiff?

A general solution of the diffusion equation can be built as a linear combination of the basic component

$$e^{-\alpha k^2 t} e^{ikx}. \tag{5}$$

These wave components are solutions of numerical schemes, but the damping factor $A^n = e^{-\alpha k^2 t}$ varies. The exact amplification factor is $A_e = e^{(-\alpha^2 k^2 \Delta t)}$.

Forward Euler: $D_t^+ u = \alpha D_x D_x u$ yields the numerical solution

$$u = (A)^n e^{ikq\Delta x} = \left(1 - 4C \sin^2\left(\frac{k\Delta x}{2}\right)\right)^n e^{ikq\Delta x} \tag{6}$$

which satisfies $A < 1$ always and $A > -1$ only if $C = \frac{\alpha \Delta t}{\Delta x^2} = \frac{1}{2}$. Halving the spatial mesh size reduces Δt by $\frac{1}{4}$. **Stiff**!

Why is Diffusion Stiff?

Backward Euler: $D_t^- u = \alpha D_x D_x u$ yields the numerical solution

$$u = (A)^n e^{ikq\Delta x} = \left(\frac{1}{1 + 4C \sin^2\left(\frac{k\Delta x}{2}\right)} \right)^n e^{ikq\Delta x} \tag{7}$$

which satisfies $0 < A < 1$ always. All numerical wave components are stable for $\Delta t > 0$. [9]

[9] Langtangen and Linge (2013), Finite difference methods for diffusion processes.

BDF Derivation

The BDF method for solving differential equations involves picking certain coefficients a_k in order to minimize the error when approximating the solution to the following IVP:

$$u' = f(t, u) \qquad u(t_0) = u_0$$

The approximation to the solution of this differential equation can be written as:

$$\sum_{k=0}^{s} a_k u_{n+k} = \left. \frac{du}{dt} \right|_{t_{n+s}}$$

- $t_n = t_0 + n\Delta t$, where Δt is the step size.
- $f(t, u)$ is the function representing the system's dynamics.

Table of Contents

BDF Derivation

For example, if we wish to derive a 3rd order BDF, we can see that we need to find coefficients such that:

$$a_0 u_n + a_1 u_{n+1} + a_2 u_{n+2} + a_3 u_{n+3} = \left.\frac{du}{dt}\right|_{t_{n+3}}$$

We start by expressing known values using Taylor expansions around t_{n+3}:

$$u_{n+2} = u_{n+3} + (-\Delta t)\left.\frac{du}{dt}\right|_{t_{n+3}} + \left.\frac{(-\Delta t)^2}{2!}\frac{d^2u}{dt^2}\right|_{t_{n+3}} + \left.\frac{(-\Delta t)^3}{3!}\frac{d^3u}{dt^3}\right|_{t_{n+3}} + \mathcal{O}(\Delta t^4)$$

$$u_{n+1} = u_{n+3} + (-2\Delta t)\left.\frac{du}{dt}\right|_{t_{n+3}} + \left.\frac{(-2\Delta t)^2}{2!}\frac{d^2u}{dt^2}\right|_{t_{n+3}} + \left.\frac{(-2\Delta t)^3}{3!}\frac{d^3u}{dt^3}\right|_{t_{n+3}} + \mathcal{O}(\Delta t^4)$$

$$u_{n} = u_{n+3} + (-3\Delta t)\left.\frac{du}{dt}\right|_{t_{n+3}} + \left.\frac{(-3\Delta t)^2}{2!}\frac{d^2u}{dt^2}\right|_{t_{n+3}} + \left.\frac{(-3\Delta t)^3}{3!}\frac{d^3u}{dt^3}\right|_{t_{n+3}} + \mathcal{O}(\Delta t^4)$$

BDF Derivation

Combining all of these into the original linear combination, we get the following:

$$\begin{aligned}
\left.\frac{du}{dt}\right|_{t_{n+3}} &= \left[a_{n+3} + a_{n+2} + a_{n+1} + a_n\right] u_{n+3} \\
&\quad + \left[(-\Delta t)a_{n+2} + (-2\Delta t)a_{n+1} + (-3\Delta t)a_n\right] \left.\frac{du}{dt}\right|_{t_{n+3}} \\
&\quad + \left[\frac{(-\Delta t)^2}{2!}a_{n+2} + \frac{(-2\Delta t)^2}{2!}a_{n+1} + \frac{(-3\Delta t)^2}{2!}a_n\right] \left.\frac{d^2u}{dt^2}\right|_{t_{n+3}} \\
&\quad + \left[\frac{(-\Delta t)^3}{3!}a_{n+2} + \frac{(-2\Delta t)^3}{3!}a_{n+1} + \frac{(-3\Delta t)^3}{3!}a_n\right] \left.\frac{d^3u}{dt^3}\right|_{t_{n+3}}
\end{aligned}$$

BDF Derivation

From this, we can equate orders of derivatives, and see that for the best approximation we want the coefficients a_n to satisfy the following linear equation, represented in matrix form:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -\Delta t & -2\Delta t & -3\Delta t \\ 0 & \frac{1}{2!}(-\Delta t)^2 & \frac{1}{2!}(-2\Delta t)^2 & \frac{1}{2!}(-3\Delta t)^2 \\ 0 & \frac{1}{3!}(-\Delta t)^3 & \frac{1}{3!}(-2\Delta t)^3 & \frac{1}{3!}(-3\Delta t)^3 \end{bmatrix} \begin{bmatrix} a_{n+3} \\ a_{n+2} \\ a_{n+1} \\ a_n \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

BDF Derivation

Solving this matrix equation using methods from linear algebra, we achieve the final result which says that:

$$\left.\frac{du}{dt}\right|_{t_{n+3}} = \frac{1}{\Delta t}\left(\frac{11}{6}u_{n+3} - 3u_{n+2} + \frac{3}{2}u_{n+1} - \frac{1}{3}u_n\right) + \mathcal{O}(\Delta t^4)$$

Given this formula, we now have a good 3rd order approximation for the solution to the differential equation, as we can use it to slowly step forward in time, using information about current and previous values.

Stability

A multistep linear process is zero-stable if and only if all of the roots of the characteristic polynomial lie within the unit circle. Root that lie on the unit circle must have multiplicity 1.
For our system above, the characteristic polynomial is

$$\rho(z) = \sum_{k=0}^{s} a_k z^k$$

By some simple computations, repeating the derivation for different orders, we can see that the BDF method is not zero stable for any order $s > 6$, so we can only use this method for up to 6th order approximations.

SciPy's BDF System Solver - Part 1

Algorithm Solving the BDF Algebraic System - Part 1

1: **function** solve_bdf_system(fun, t_new, y_predict, c, psi, LU, solve_lu, scale, tol)
2: Initialize deviation d and copy predicted value y
3: Initialize old norm of correction dy_norm_old to None
4: Set convergence flag *converged* to False
5: **for** $k = 0$ to $NEWTON\_MAXITER - 1$ **do**
6: Evaluate f using $fun(t\_new, y)$
7: **if** any element of f is not finite **then**
8: **break**
9: **end if**
10: Compute correction dy by solving linear system
11: Calculate norm of correction dy_norm
12: **if** previous norm dy_norm_old exists **then**
13: Compute rate of convergence *rate*
14: **end if** =0

SciPy's BDF System Solver - Part 2

Algorithm Solving the BDF Algebraic System - Part 2

```
        if convergence rate is poor or error estimate is large then
            break
        end if
        Update y and cumulative deviation d with dy
        if correction is sufficiently small then
            Set converged to True and exit loop
        end if
        Update old norm of correction dy_norm_old
    end for
    return converged, iteration count k + 1, updated y, and deviation d
end function
```

Initialization of SciPy's BDF Solver

Algorithm Initialization

```
function init(fun, t0, y0, t_bound, max_step, rtol, atol, jac, jac_sparsity, vectorized, first_step, extraneous)
    Set solver parameters (max_step, rtol, atol, etc.)
    Validate first step and compute initial Jacobian
    Initialize LU decomposition and solver functions
    Precompute coefficients for BDF
    Allocate memory for differences array D
    Set initial order and steps counters
    Validate Jacobian and sparsity pattern
end function
```

BDF Step Documentation - Part 1

Algorithm BDF Step Documentation - Part 1

```
1:  function step_impl
2:      Check and adjust the step size if it exceeds the maximum
3:      Calculate the new time level and predict the state
4:      Compute scaling factors for error estimation
5:      Initialize variables for convergence checking
6:      Prepare for solving the BDF algebraic system
7:      repeat
8:          Check if the Jacobian matrix needs updating
9:          Solve the BDF system using Newton's method
10:         Update the solution and evaluate convergence
```

BDF Step Documentation - Part 2

Algorithm BDF Step Implementation - Part 2

```
        if solution has not converged and Jacobian is updated then
            break
        end if
        if solution has not converged then
            Reduce the step size and reattempt the solution
        end if
    until convergence is achieved or step size becomes too small
    Update error estimate and adjust step size for next step
    Update BDF order based on error norms
    Update differences array for the next step
    return status of the step
end function
```

Table of Contents

Future Work: Lineage Conversion

Molecular Advancements:

- Development of in vivo lineage conversion

Computational and Mathematical Next Steps

- Characterize parameter dependency of heterogeneity to minimize these effects

References

1. Joo, J. I., Zhou, J. X., Huang, S., & Cho, K. H. (2018). **Determining relative dynamic stability of cell states using boolean network model.** *Nature Scientific Reports*, 8(1), 12077.
2. Gonze, D., Lahti, L., Raes, J., Faust, K. (2017). **Multi-stability and the origin of microbial community types.** *The ISME journal*, 11(10), 2159-2166.
3. Lu, M., & Hedin, L. O. (2019). **Global plant–symbiont organization and emergence of biogeochemical cycles resolved by evolution-based trait modelling.** *Nature Ecology & Evolution*, 3(2), 239-250.

Our Futures

Carlos
Applying to graduate programs in biology

Shloka
Applying to PhD programs in ecology

Michael
Moving to Armenia