Developing methods for developmental modeling: Learning reduced stochastic dynamics and Algebras of dynamic structures

Eric Mjolsness

University of California, Irvine
http://emj.ics.uci.edu

KITP, UC Santa Barbara
15 August 2019
Updated refs: 10 Sept 2019
Acknowledgements

- Modeling & computing
  - Oliver Ernst (UCSD), Tom Bartol (Salk), Terry Sejnowski (Salk)
  - Cory Scott (UCI), Bob Bird (LANL), Eric Medwedeff (UCI)

- Plant Biology
  - Elliot Meyerowitz (Caltech), Ray Wightman (SLCU)

- Funding: HFSP, NIA/NIH

- Visits: KITP, CNLS, … SLCU
Outline

• This is a talk about methods - computational and mathematical

• Preamble: “Principles in biology” (1 slide)

Machine learning for model reduction: Dynamic Boltzmann Distributions

Algebra of dynamic spatially embedded graphs (structures), as semantics for languages sufficient for bio model reduction

• Epilogue: A conceptual architecture for model stacks (3 slides)

Mappings:
- Semantics
- Reduction of models
- Analysis of models
- Implementation of models

Ψ
ℛ
Α
fläche
Preamble: Some candidate bio “principles”

- **Biophysical**
  - scarce resources: Follow the ... energy, elements/small molecules; information, proximity/access
  - specific feedback inhibition in biosynthesis [Umbarger 1950]
  - co-option of emergent properties (biomechanics, self-organization, phase separation, ... )
  - regeneration of ~modular subunits => robustness
  - dynamic structures (~spatially embedded graphs) recur at all scales

- **Informational**
  - Information bottlenecks are key (e.g. genome; cell-cell signaling; spatial info flow in cell & dev ... )
  - regulation, replication, ... are catalysis by information. Other processes produce/consume information.
  - internal representations (of world, self) are highly functional as reduced models. (E.g. positional info~charts)
  - meta-evolution works (evo of evo; evo of sub-evolutions)

- **Methodological**
  - We’re not smart enough to just think it all through *(but we should try anyway; then use cyborg mode ...)*
  - mathematical/computational **models**, simulations, & analyses are essential tools for understanding ...
  - *but also automated multiscale model stacks* ⇒ numeric (ML) plus symbolic AI needed!
Reduced model examples

- Well-mixed mass action concentration models of biochemical networks
- PDE mass action reaction-diffusion models
- Cell-centered biomechanical models of SAM
- Vertex biomechanical models of animal epithelia
- FEM multi-compartmental biomechanical models
- Mean field theory approaches to X

Analyses:
- topology of biomech models
- phase diagram; bifurcation diagram
Learning reduced stochastic dynamics
Multiple Scales of Synapse

- multiscale modeling of synapse in MCell
- methods vs. problem scale
E.g.: CaMKII Signaling Model

*interacting particles with dynamical state information*

Original model: [Pepke et al. 2010]

Figure 7.1: An MRF model of calcium binding, CaM/CaMKII interaction, and CaMKII dimerization.
**GCCD: Target and Approximate Stochastic Dynamics**

- **Target stoch. dynamics:** Chemical master equation
  \[
  \frac{dp}{dt} = W \cdot p
  \]
  i.e.
  \[
  \frac{d}{dt} p([n_i]) = \sum_{r} p^{(r)} \left( \prod_{j} (n_j - S_j^{(r)}) \right) p([n_i - S_i^{(r)}]) - \sum_{r} p^{(r)} \left( \prod_{j} n_j \right) p([n_i])
  \]

- **Approximation:** Boltzmann/MRF + parameter ODEs
  \[
  \rho(R,t) = \exp \left[ - \sum_{\alpha} \mu_\alpha(t) V_\alpha(R) \right] / \hat{Z}(\mu(t))
  \]
  \[
  \frac{d}{dt} \mu_\alpha = f_\alpha(\mu|\theta) = \sum_A \theta_A f_{\alpha A}(\mu)
  \]

- **Error criterion:** L1-regularized sum squared error
  \[
  S([\theta_A]) = \sum_{a,t,di,s} \left| \frac{d\mu_\alpha(t)}{dt} \right|_{fit[\theta_\alpha A]} - \left| \frac{d\mu_\alpha(t)}{dt} \right|_{BMLA} \right|^2 + \lambda \sum_A |\theta_A|
  \]

- **Name:** Graph-Constrained Correlation Dynamics
  
  - “Graph” = assumed MRF structure graph; “Correlations” =
  \[
  \mu_c V_c(X_c)
  \]
GCCD eg. Synapse model spike train

- Fine scale: rule-based particle methods
- Coarse scale: time-varying Boltzmann distribution

Figure 7.12: Set of ordinary differential equations with learned coefficients (red lines) versus time series of eight MRF parameter values (colored lines) (MCell), spike train.

[Johnson et al., Physical Biology 2015]
Mapping: Model reduction

- **Nonspatial:**
  
  \[ \dot{\rho}(R, t) = \exp \left[ - \sum_{\alpha} \mu_{\alpha}(t) V_{\alpha}(R) / \dot{Z}(\mu(t)) \right] \]
  
  - Graph-Constrained Correlation Dynamics
  - warmup case for ...

- **Spatial generalization:**

  \[ \tilde{\rho}(n, x, \alpha, t) = \frac{1}{Z} \exp \left[ - \sum_{k=1}^{K} \sum_{\langle j \rangle} \nu_k(x_{\langle j \rangle}, \alpha_{\langle j \rangle}, t) \right] , \]
  
  - Dynamic Boltzmannmann distributions
Approximating Statistical Systems by Dynamic Boltzmann Distributions

\[ \dot{p} = Wp \rightarrow \tilde{p} \sim \exp(-E)/Z \]
Analytically construct MaxEnt approx. to MEs describing simple systems.

Identify basis functions
Time evolution of approximating probability dist.

\[ F(\nu) = \alpha_{\text{Diff}}(t) F_{\text{Diff}}(\nu) + \alpha_{A \rightarrow \emptyset}(t) F_{A \rightarrow \emptyset}(\nu) + \ldots \]
ML projections from true systems into reduced space.

Reduced model is \textit{linear} in fine-scale model sum over processes
\[ \Rightarrow \mathcal{R} \text{ respects model structure} \]
MaxEnt Problem

\[ S = \int_0^\infty dt \, \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) \]

w/ \[ \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) = \sum_{n=0}^\infty \int dx \, p \ln \frac{p}{\tilde{p}} \]

\[ \tilde{p}(n, x, \alpha, t) = \frac{1}{Z} \exp \left[ - \sum_{k=1}^K \sum_{\langle j \rangle} \nu_k(x_{\langle j \rangle}, \alpha_{\langle j \rangle}, t) \right] , \]

Variational problem

\[ \frac{\delta S}{\delta F_k[\{\nu_k(x)\}]_{k=1}^K} = 0 \text{ for } k = 1, \ldots, K \text{ at all } x \] (12)

where the variation is with respect to a set of functionals

\[ \dot{\nu}_k(x) = F_k[\{\dot{\nu}_k\}]_{k=1}^K \] (13)

… Higher-order calculus!
Variational Problem: Spatial systems

\[
\frac{\delta S}{\delta F_k[\nu(x)']} = \sum_{k'=1}^{K} \int dx' \int dt \frac{\delta S}{\delta \nu_{k'}(x', t)} \frac{\delta \nu_{k'}(x', t)}{\delta F_k[\nu(x)']} = 0 \quad (19)
\]

(1) \[
\frac{\delta S}{\delta \nu_{k'}(x', t)} = \left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(x' - x_{\langle i \rangle_{k'}^{n}}) \right\rangle_p - \left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(x' - x_{\langle i \rangle_{k'}^{n}}) \right\rangle_{\tilde{p}} \quad (20)
\]

e.g. \( k' = 1 \): \( \left\langle \sum_{i=1}^{n} \delta(x_i - x') \right\rangle \) for all \( x' \)

\( k' = 2 \): \( \left\langle \sum_{i=1}^{n} \sum_{j>i} \delta(x_i - x'_1)\delta(x_j - x'_2) \right\rangle \) for all \( x'_1, x'_2 \)

Need to choose a parametrization for functional!

Diffusion-inspired parametrization

\[ p(x) \sim \exp \left[ -\frac{(x - x_0)^2}{4Dt} \right] \rightarrow \exp[-\nu_1(x)] \]

satisfies: \[ \frac{\partial \nu_1}{\partial t} = D \nabla^2 \nu_1(x) - D(\nabla \nu_1(x))^2 \]

∴ \[ F_k[\nu(x)] = F_k^{(0)} + \sum_{\lambda=1}^{k} F_k^{(1)}(\nabla \nu_\lambda)^2 + \sum_{\lambda=1}^{k} F_k^{(2)}(\nabla^2 \nu_\lambda) \]  \hspace{1cm} (20)

where: \( F = \) some funcs of \( \nu \) on LHS

\[ \frac{\delta S}{\delta F_k^{(0)}} = 0, \quad \frac{\delta S}{\delta F_k^{(1)}} = 0, \quad \frac{\delta S}{\delta F_k^{(2)}} = 0 \]

PDE-constrained Optimization Problem

Minimize \[ \sum_{k'=1}^{K} \int_0^{\infty} dt \left( \left\langle \sum_{\langle i \rangle_{k'}} \delta(x' - x_{\langle i \rangle_{k'}}^n) \right\rangle_p - \left\langle \sum_{\langle i \rangle_{k'}} \delta(x' - x_{\langle i \rangle_{k'}}^n) \right\rangle_{\tilde{p}} \right) \frac{\delta \nu_{k'}(t)}{\delta F} \]  \hspace{1cm} (23)

subject to PDE constraints for \( \delta \nu_{k'}(t)/\delta F \).
Spatial Dynamic Boltzmann Distributions

\[ Z = \sum \sum \exp \left[ \sum_{i=1} h_{\alpha_i}(t) s_i + \sum_{i=1} J_{\alpha_i, \alpha_{i+1}}(t) s_i s_{i+1} \right] \]

Diffusion: $\bar{F}_h$
A$\rightarrow$Ø: $\bar{F}_h$
A+A$\rightarrow$Ø: $\bar{F}_h$
A$\rightarrow$A+A: $\bar{F}_h$
A+A$\rightarrow$A: $\bar{F}_h$

Diffusion: $\bar{F}_j$
A$\rightarrow$Ø: $\bar{F}_j$
A+A$\rightarrow$Ø: $\bar{F}_j$
A$\rightarrow$A+A: $\bar{F}_j$
A+A$\rightarrow$A: $\bar{F}_j$

Initial

Final

Mean (stoch. sim)
Mean (asleep phase)
NN (stoch. sim)
NN (asleep phase)


Slides: Oliver Ernst, Salk
Algorithm 2. Boltzmann machine-style learning of dynamics.

1: Initialize
2: \( \text{Initial } \theta^{(r)} \text{ for all } r. \)
3: Max. integration time \( T. \)
4: A formula for the learning rate \( \lambda. \)
5: Time-series of lattice spins \( \{ s \}(t) \) from stochastic simulations from some known IC \( h_0, J_0. \)
6: Fully visible MRF with NN connections and as many units as lattice sites \( N. \)
7: while not converged do
8: \( \text{Generate trajectory in reduced space:} \)
9: Solve the PDE constraint (52) with IC \( h_0, J_0 \) for \( 0 \leq t \leq T. \)
10: \( \text{Awake phase:} \)
11: Evaluate true moments \( \mu(t), \Delta(t) \) from the Stochastic simulation data \( \{ s \}(t). \)
12: \( \text{Asleep phase:} \)
13: Evaluate moments \( \tilde{\mu}(t), \tilde{\Delta}(t) \) of the Boltzmann distribution by Gibbs sampling.
14: \( \text{Update to decrease objective function:} \)
15: Solve (54) for derivative terms.
16: Update \( \theta^{(s)} \) to decrease the objective function for all \( s \) by taking: \( \theta^{(s)} \rightarrow \theta^{(s)} - \lambda \times (53). \)

\[
\begin{align*}
\frac{dh}{dt} &= F_h(h, J) = \sum_r \theta^{(r)} \tilde{F}_h^{(r)}, \\
\frac{dJ}{dt} &= F_J(h, J) = \sum_r \theta^{(r)} \tilde{F}_J^{(r)}. \\
\frac{\partial}{\partial t} \left( \frac{\partial h(t')}{\partial \theta^{(s)}} \right) &= \tilde{F}_h^{(s)} + \sum_r \theta^{(r)} \frac{\partial \tilde{F}_h^{(r)}}{\partial h} \\
&+ \sum_r \theta^{(r)} \frac{\partial \tilde{F}_h^{(r)}}{\partial J}, \\
\frac{\partial}{\partial t} \left( \frac{\partial J(t')}{\partial \theta^{(s)}} \right) &= \tilde{F}_J^{(s)} + \sum_r \theta^{(r)} \frac{\partial \tilde{F}_J^{(r)}}{\partial h} \\
&+ \sum_r \theta^{(r)} \frac{\partial \tilde{F}_J^{(r)}}{\partial J}, \\
\int_0^\infty dt' \left( \tilde{\mu}(t') - \mu(t') \right) \frac{\partial h(t')}{\partial \theta^{(s)}} \\
&+ \int_0^\infty dt' \left( \tilde{\Delta}(t') - \Delta(t') \right) \frac{\partial J(t')}{\partial \theta^{(s)}} = 0,
\end{align*}
\]

Algorithm 1 Stochastic Gradient Descent for Learning Restricted Boltzmann Machine Dynamics

1: Initialize
2: Parameters $\mathbf{u}_k$ controlling the functions $F_k(\theta; \mathbf{u}_k)$ for all $k = 1, \ldots, K$.
3: Time interval $[t_0, t_f]$, a formula for the learning rate $\lambda$.
4: while not converged do
5: Initialize $\Delta F_{k,i} = 0$ for all $k = 1, \ldots, K$ and parameters $i = 1, \ldots, M_k$.
6: for sample in batch do
7: $\triangleright$ Generate trajectory in reduced space $\theta$:
8: Solve the PDE constraint (27) for $\theta_k(t)$ with a given IC $\theta_{k,0}$ over $t_0 \leq t \leq t_f$, for all $k$.
9: $\triangleright$ Wake phase:
10: Evaluate moments $\mu_k(t)$ of the data for all $k, t$.
11: $\triangleright$ Sleep phase:
12: Evaluate moments $\tilde{\mu}_k(t)$ of the Boltzmann distribution.
13: $\triangleright$ Solve the adjoint system:
14: Solve the adjoint system (31) for $\phi_k(t)$ for all $k, t$.
15: $\triangleright$ Evaluate the objective function:
16: Update $\Delta F_{k,i}$ as the cumulative moving average of the sensitivity equation (30) over the batch.
17: $\triangleright$ Update to decrease objective function:
18: $u_{k,i} \rightarrow u_{k,i} - \lambda \Delta F_{k,i}$ for all $k, i$.

$\frac{dS}{du_{k,i}} = -\int_{t_0}^{t_f} dt \frac{\partial F_k(\theta(t); \mathbf{u}_k)}{\partial u_{k,i}} \phi_k(t)$.

[Ernst, Bartol, Sejnowski, Mjolsness, Phys Rev E 99 063315, 2019]
Benefit of Hidden Units

Network: fratricide + lattice diffusion

\[ \mathcal{R} \]

\[ \text{Ernst, Bartol, Sejnowski, Mjolsness, Phys Rev E 99 063315, 2019} \]
Benefit of Hidden Units

Network: fratricide + lattice diffusion

- Learned DBD ODE RHS, without and with hidden units

![Graphs showing time-evolution functions for the fully visible model (19), using the $Q_4$, $C_1$ finite element parameterization (21) with 5 x 5 evenly spaced cubic cells. Left: Training set of initial points (b, J, K) (cyan) sampled evenly in $[-1,1]$. Stochastic simulations for each initial point are used as training data (learned trajectories shown in black, endpoints in magenta). Other panels: The time evolution functions learned. Bottom row: Hidden layer model (29) and parameterization (21) with the same number of cells as the visible model. Initial points are generated by BM learning the points of the visible model.](image)

$\mathbb{R}^{n \times n}$

MSE of 4th order stats

[Ernst, Bartol, Sejnowski, Mjolsness, Phys Rev E 99 063315, 2019]
Rössler Oscillator in 3D

- Function:

- Learned DBD ODE RHS:

[Ernst, Bartol, Sejnowski, Mjolsness, Phys Rev E 99 063315, 2019]
Rössler Oscillator in 3D

- Learned correlations:
  - [Ernst, Bartol, Sejnowski, Mjolsness, Phys Rev E 99 063315, 2019]

- Learned Configuration

[Image: Graphs and 3D models showing phase plots and configurations]
Learned model reduction maps: Implications

- We can and should seek not models, but model stacks
- simulation = model₀ ↦ model₁ ↦ ... ↦ modelₙ = analysis
- each reduction is conditional
- great computing resources required at all levels - but becoming available
Algebras of dynamic structures
Living matter:  
Tissues at cellular scale

Spring biomechanics:

Voronoi (or power) diagrams fit SAM geometry
Dynamic cell structures in *Drosophila* embryo

Intercalation and convergent extension observed during germ band elongation in *Drosophila* embryo. Note topological rearrangements. [Bertet et al. 2004]
Dynamic bio structures

- geo-cell complexes of bio-cells in tissues
- cytoskeleton
  - supercellular cables
  - axons & dendrites
  - cytonemes
  - ...
- cell-centered and vertex biomechanical models
  - PDE adaptive meshes and grids
Microtubule dynamics


WT data. Also have mutants: *spiral2* and *botero*

Microtubule dynamics


WT data. Also have mutants: spiral2 and botero

Microtubule dynamics

Cortical microtubules in *Arabidopsis* petiole cells.
Movie with Ray Wightman SLCU May 2015

WT data.
Also have mutants: *spiral2* and *botero*

More cortical microtubules, color coded by growth vs shrinkage, in 3D.
From Ray Wightman SLCU 2015.
Bundling or Zippering

Collision catastrophe
Simulated bundling, catastrophe

Dustin Maurer + Francois Nedelec
Simulated bundling, catastrophe

Dustin Maurer + Francois Nedelec
MT fiber

Stochastic Parametrized Graph Grammar

\[
\begin{align*}
(\bullet_1) \langle (x_1, u_1) \rangle &\rightarrow (\bigcirc_1 \rightarrow \bullet_2) \langle (x_1, u_1), (x_2, u_2) \rangle \\
& \quad \text{with } \hat{\rho}_{\text{grow}}([\text{tubulin}]) \mathcal{N}(x_1 - x_2; Lu_1, \sigma) \mathcal{N}(u_2; u_1/(|u_1| + \epsilon), \epsilon), \\
(\bigblackdiamond_1 \rightarrow \bigcirc_2) \langle (x_1, u_1), (x_2, u_2) \rangle &\rightarrow (\bullet_2) \langle (x_2, u_2) \rangle \\
& \quad \text{with } \hat{\rho}_{\text{retract}} \\
(\bigcirc_1 \rightarrow \bigcirc_2 \rightarrow \bigcirc_3) \langle (x_1, u_1), (x_2, u_2), (x_3, u_3), (x_4, u_4) \rangle &\rightarrow \left(\bigcirc_1 \rightarrow \bigtriangleup_2 \rightarrow \bigcirc_3\right) \langle (x_1, u_1), (x_2, u_2), (x_3, u_3), (x_4, u_4) \rangle \\
& \quad \text{with } \hat{\rho}_{\text{bundle}}(|u_2 \cdot u_4|/|\cos \theta_{\text{crit}}|) \exp(-|x_2 - x_4|^2/2L^2) \\
(\bigblackdiamond_1 \rightarrow \bullet_2) \langle (x_1, u_1), (x_2, u_2) \rangle &\leftrightarrow \emptyset \quad \text{with } \left(\hat{\rho}_{\text{retract}}, \hat{\rho}_{\text{nucleate}}([\text{tubulin}]) \mathcal{N}(x; 0, \sigma_{\text{broad}}) \delta_{\text{Dirac}}(|u_1| - 1) \delta_{\text{Dirac}}(u_1 - u_2)\right) \\
(\bullet_1) \langle (x_1, u_1) \rangle &\leftrightarrow \left(\bigblackdiamond_1\right) \langle (x_1, u_1) \rangle \\
& \quad \text{with } \left(\hat{\rho}_{\text{retract}} \leftrightarrow \text{growth}, \hat{\rho}_{\text{growth}} \leftrightarrow \text{retract}\right)
\end{align*}
\]

MT fiber
Stochastic Parametrized Graph Grammar

[Chakrabortty et al.
Current Biology 2018]

+arXiv:1804.11044]
MT fiber

Dynamical Graph Grammar

(hand-transformed from stochastic G.G.)

5.2 MT dynamical graph grammar

// Treadmilling (growth end):
\[ (O_1 \longrightarrow \bullet_2) \langle (l, u), (x_+, u_+) \rangle \rightarrow (O_1 \longrightarrow \bullet_2) \langle (l, u), (x_+ + dx_+, u_+) \rangle \]
\[ \text{solving } dx_+/dt = \dot{\rho}_{\text{grow}}([Y_g])(1 - l/l_{\text{max}})u_+ \]

// Treadmilling (retracting end):
\[ (\bullet_1 \longrightarrow O_2) \langle (x_- u_-), (l, u) \rangle \rightarrow (\bullet_1 \longrightarrow O_2) \langle (x_- + dx_-, u_-), (l, u) \rangle \]
\[ \text{solving } dx_-/dt = \dot{\rho}_{\text{retract}}([Y_r])(l/l_{\text{max}})u \]

// Treadmilling (interior node):
\[ (\bullet_1 \longrightarrow O_2 \longrightarrow \bullet_3) \langle (x_- u_-), (l, u), (x_+, u_+) \rangle \]
\[ \rightarrow (\bullet_1 \longrightarrow O_2 \longrightarrow \bullet_3) \langle (x_- u_-), (l + dl, u), (x_+, u_+) \rangle \]
\[ \text{solving } dl/dt = |dx_+/dt| - |dx_-/dt| = \dot{\rho}_{\text{grow}}([Y_g]) - (\dot{\rho}_{\text{grow}}([Y_g]) + \dot{\rho}_{\text{retract}}([Y_r]))(l/l_{\text{max}}) \]

// Treadmilling (interior node):
\[ (\bullet_1 \longrightarrow O_2 \longrightarrow \bullet_3) \langle (x_- u_-), (l, u), (x_+, u_+) \rangle \]
\[ \rightarrow (\bullet_1 \longrightarrow O_2 \longrightarrow \bullet_3) \langle (x_- u_-), (l + dl, u), (x_+, u_+) \rangle \]
\[ \text{solving } dl/dt = |dx_+/dt| - |dx_-/dt| = 2\dot{\rho}_{\text{grow}}([Y_g])(1 - l/l_{\text{max}})u_+ \]

// Treadmilling (interior node):
\[ (\bullet_1 \longrightarrow O_2 \longrightarrow \bullet_3) \langle (x_- u_-), (l, u), (x_+, u_+) \rangle \]
\[ \rightarrow (\bullet_1 \longrightarrow O_2 \longrightarrow \bullet_3) \langle (x_- u_-), (l + dl, u), (x_+, u_+) \rangle \]
\[ \text{solving } dl/dt = |dx_+/dt| - |dx_-/dt| = 2\dot{\rho}_{\text{retract}}([Y_r])(l/l_{\text{max}})u_- \]

// Fiber collision, exerting continuous force:
\[ (\star_1 \longrightarrow O_2 \longrightarrow \star_3) \langle (x_1, u_1), (l_2, u_2), (x_3, u_3), (l_4, u_4), (x_5, u_5) \rangle \]
\[ \rightarrow (\star_1 \longrightarrow O_2 \longrightarrow \star_3) \langle (x_1, u_1), (l_2, u_2), (x_3, u_3), (l_4 + dl_4, u_4), (x_5 + dx_5, u_5) \rangle \]
\[ \text{solving } \begin{align*}
    dx_5/\text{dt} &= \kappa u_5[\hat{e}_\gamma \exp(-\gamma^2/2\epsilon^2)]\Theta(\epsilon \leq \alpha \leq 1 - \epsilon) \\
    dl_4/\text{dt} &= u_5 \cdot dx_5/\text{dt} = \kappa \hat{e}_\gamma \exp(-\gamma^2/2\epsilon^2)]\Theta(\epsilon \leq \alpha \leq 1 - \epsilon)
\end{align*} \]

where
\[ \begin{align*}
    \gamma &= -[(x_3 - x_1) \times (x_1 - x_5)]_z/[(x_3 - x_1) \times u_5]_z \quad \text{// rel. distance to intersection along } u_5 \\
    \alpha &= -[(x_1 - x_4) \times u_3]_z/[(x_3 - x_1) \times u_3]_z \quad \text{// fractional location of intersection along } u_2
\end{align*} \]
MT fiber

Dynamical Graph Grammar

(hand-transformed from stochastic G.G.)

// (continued)

// Fiber collision, with several alternative discrete outcomes:

\[
\begin{align*}
\rightarrow \left( \begin{array}{c}
\star_1 \quad \circ_2 \quad \star_3 \\
\circ_4 \quad \bullet_5 \\
\end{array} \right) & \preccurlyeq (x_1, u_1), (l_2, u_2), (x_3, u_3), (l_4, u_4), (x_5, u_5) \\
\rightarrow \left( \begin{array}{c}
\star_1 \quad \circ_6 \quad \bullet_2 \quad \circ_7 \quad \star_3 \\
\circ_4 \\
\end{array} \right) & \preccurlyeq (x_1, u_1), (x_2, u_2), (x_3, u_3), (l_4, u_4), \emptyset, (a l_4, u_2), ((1 - a) l_4, u_2) \\
\rightarrow \left( \begin{array}{c}
\star_1 \quad \circ_6 \quad \bullet_2 \quad \circ_7 \quad \star_3 \\
\circ_4 \quad \bullet_5 \\
\end{array} \right) & \preccurlyeq (x_1, u_1), (l_2, u_2), (x_3, u_3), (l_4, u_4), (x_5, u_5) \\
\rightarrow \left( \begin{array}{c}
\star_1 \quad \circ_6 \quad \bullet_2 \quad \circ_7 \quad \star_3 \\
\circ_4 \\
\end{array} \right) & \preccurlyeq (x_1, u_1), ((1 - a) l_2, u_2), (x_3, u_3), (l_4, u_4), \emptyset, \\
& ((1 - a) l_4, u_2), ((1 - a) l_2, u_2), (x_2 + a l_4, u_4), (x_4) \\
\end{align*}
\]

with \( \hat{\rho}_{\text{bundle}}(|u_2 \cdot u_4|/|\cos \theta_{\text{crit}}|) \exp \left(-\frac{\gamma^2}{2\epsilon^2}\right) \Theta(\epsilon \leq a \leq 1 - \epsilon) \)

[Chakrabortty et al. Current Biology 2018]

\[\gamma = -[(x_3 - x_1) \times (x_1 - x_5)]_z/[(x_3 - x_1) \times u_5]_z \quad \text{rel. distance to intersection along } u_5\]

\[a = -[(x_1 - x_5) \times u_5]_z/[(x_3 - x_1) \times u_5]_z \quad \text{fractional location of intersection along } u_2\]

Operator algebra for
Pure stochastic chemical reactions

For reaction/rule \( r \):

\[
\hat{W}_{\{m_i^{(r)}\} \to \{n_i^{(r)}\}} = k^{(r)} \prod_i (\hat{a}_i)^{n_i^{(r)}} (a_i)^{m_i^{(r)}}
\]

\[n_\alpha \in \mathbb{N} : [a_\alpha, \hat{a}_\beta] = \delta_{\alpha\beta} I, \text{ i.e.}\]

\[a_\alpha \hat{a}_\beta = \hat{a}_\beta a_\alpha + \delta_{\alpha\beta} I\alpha\]

\[n_\alpha \in \{0,1\} : a_\alpha \hat{a}_\beta = (1 - \delta_{\alpha\beta}) \hat{a}_\beta a_\alpha + \delta_{\alpha\beta} Z\alpha\]

For reaction/rules \( r_1 \) and \( r_2 \):

\[
\hat{W}_{\{m_i^{(r_2)}\} \to \{n_i^{(r_2)}\}} \hat{W}_{\{m_i^{(r_1)}\} \to \{n_i^{(r_1)}\}} = k^{(r_2)} k^{(r_1)} \sum_{\{l_i=0\ldots \min(m_i^{(r_2)},n_i^{(r_1)})\}} \left( \prod_i \frac{(m_i^{(r_2)})_l (n_i^{(r_1)})_l}{l_i!} \right)
\]

\[\times \hat{W}_{\{(m_i^{(r_1)}+m_i^{(r_2)} - l_i)\} \to \{(n_i^{(r_1)}+n_i^{(r_2)} - l_i)\}}\]

Why: \( \partial_x^m (x^n f(x)) = \text{binomial sum} \)
Lie Algebra for
Pure stochastic chemical reactions

• Rotation group: \([X, Y] = Z + \text{ cyclic}\)

  • Curvature in a Lie group with invariant metric:
  \[
  R(X, Y)Z = \frac{1}{4}[[X, Y], Z]
  \]

• For reaction/rule \(r\):
  \[
  [a_\alpha, \hat{a}_\beta] = \delta_{\alpha\beta} I
  \]

\[
\hat{W}_{\{m_i^{(r)}\} \rightarrow \{n_i^{(r)}\}} = k^{(r)} \prod_i (\hat{a}_i)^{n_i^{(r)}} (a_i)^{m_i^{(r)}}
\]

• For reaction/rules \(r_1\) and \(r_2\):
  \[
  [\hat{W}_{\{m_i^{(r_2)}\} \rightarrow \{n_i^{(r_2)}\}}, \hat{W}_{\{m_i^{(r_1)}\} \rightarrow \{n_i^{(r_1)}\}}] = k^{(r_2)}k^{(r_1)} 
  \sum_{\{l_i = 0 \ldots \min(m_i^{(r_2)}, n_i^{(r_1)})\}} \left[ \left( \prod_i \frac{(m_i^{(r_2)})_l (n_i^{(r_1)})_l}{l_i!} \right) - \left( \prod_i \frac{(m_i^{(r_1)})_l (n_i^{(r_2)})_l}{l_i!} \right) \right] 
  \times \hat{W}_{\{(m_i^{(r_1)}+m_i^{(r_2)}-l_i) \rightarrow (n_i^{(r_1)}+n_i^{(r_2)}-l_i)\}}
  \]

where \((n)_l \equiv \begin{cases} n!/l! & \text{for } l \leq n; \\ 0 & \text{otherwise}. \end{cases}\)
Particle to Structure Dynamics

- **Particle** reactions/transitions, with params

\[ A_1(x_1), A_2(x_2), ..., A_n(x_n) \rightarrow B_1(y_1), B_2(y_2), ..., B_m(y_m) \text{ with } \rho(\{x_i\}, \{y_j\}) \]

\[ \tilde{\omega}_r = \rho_r \sum_{\{x'_i, x_j\}} \prod_{i \in \text{lhs}(r)} \hat{a}(\tau_i, x_i) \prod_{j \in \text{rhs}(r)} a(\tau_j, x_j) \Pr(\{x_i\} | \{x_j\}) \]

(and can integrate ODE rules too)
Particle to Structure Dynamics

- **Particle** reactions/transitions, with params

\[ A_1(x_1), A_2(x_2), ... , A_n(x_n) \rightarrow B_1(y_1), B_2(y_2), ... , B_m(y_m) \text{ with } \rho(\{x_i\}, \{y_j\}) \]

\[ \tilde{\rho}_r = \rho_r \sum_{[x'_i,x_j]} \prod_{i \in \text{rhs}(r)} \tilde{a}(\tau_i, x_i) \prod_{j \in \text{lhs}(r)} a(\tau_j, x_j) \Pr(\{x_i\} | \{x_j\}) \]

(and can integrate ODE rules too)

- **Labelled graph** (structure) transitions

\[ \hat{a}_\alpha^2 = 0 = a_\alpha^2 \]

\[ a_\alpha \hat{a}_\beta = (1 - \delta_{\alpha\beta}) \hat{a}_\beta a_\alpha + \delta_{\alpha\beta} Z_\alpha \]

\[ Z_\alpha \equiv I_\alpha - N_\alpha \]

\[ N_\alpha = \hat{a}_\alpha a_\alpha \]

\[ \hat{a}_{i_1...i_k}(G') = \hat{a}_{i_1...i_k}(G'_{\text{links}}) \hat{a}_{i_1...i_k}(G'_{\text{nodes}}) \]

\[ = \prod_{s',t' \in \text{rhs}(r)} \left( \hat{a}_{i_{s'},i_{t'}} \right)^{\gamma_{s',t'}} \left[ \prod_{t' \in \text{rhs}(r)} \hat{a}_{i_{t'},\lambda_{t'}} \right] \]

\[ a_{i_1...i_k}(G) = a_{i_1...i_k}(G_{\text{links}}) a_{i_1...i_k}(G_{\text{nodes}}) \]

\[ = \prod_{s,t \in \text{lhs}(r)} \left( a_{i_{s},i_{t}} \right)^{\delta_{s,t}} \left[ \prod_{i_{t} \in \text{lhs}(r)} a_{i_{t},\lambda_{t}} \right]. \]

[EM, MFPS Proc., ENTCS 2010]
Graph rewrite rule operators

- $G = \text{LHS labelled graph, } G' = \text{RHS labelled graph}$

$$\dot{O}_r = \frac{1}{k!} \sum_{\{i_1,\ldots,i_k\}} \prod_{c,d \in \text{rhs}(r)} (\hat{a}_{i_c i_d}) g'_{c d} \prod_{c \in \text{rhs}(r)} \hat{a}_{i_c} \lambda'_c \prod_{a, b \in \text{lhs}(r)} (a_{i_a i_b}) g_{a b} \prod_{a \in \text{lhs}(r)} a_{i_a} \lambda_a$$

[EM, MFPS Proc. 2010]

\[
\begin{align*}
\hat{a}_\alpha^2 &= 0 = a_\alpha^2 \\
a_\alpha \hat{a}_\beta &= (1 - \delta_{\alpha \beta}) \hat{a}_\beta a_\alpha + \delta_{\alpha \beta} Z_\alpha \\
Z_\alpha &= I_\alpha - N_\alpha \\
N_\alpha &= \hat{a}_\alpha a_\alpha
\end{align*}
\]
MT Treadmilling Rules

Rule 1: end extension
\[
\begin{align*}
\delta_1 &= 1 \\
\lambda_1 &= 1 \\
\hat{w}_1 &= \sum_{ij} \hat{a}_{i,j_0} \hat{a}_{i,j} \hat{a}_{j,i} \hat{a}_{j,j_0}
\end{align*}
\]

Rule 2: end retraction
\[
\begin{align*}
\delta_2 &= 1 \\
\lambda_2 &= 1 \\
\hat{w}_2 &= \sum_{i,j} \hat{a}_{j,i} \hat{a}_{i,j} \hat{a}_{j,j_0} \hat{a}_{j,j_1}
\end{align*}
\]

\[\hat{w}_1 - \hat{w}_2 \propto \sum_{ij} \hat{a}_{i,j_0} \hat{a}_{i,j} \hat{a}_{j,i} \hat{a}_{j,j_0} \]

\[\approx I\]

(if dangling edges are removed)
Growth vs. Bundling

+ end growth

bundling
Growth vs. Bundling

\[
S_2 \left( \hat{\phi}_2, \hat{\psi}_1 \right) = \hat{\phi}_2 \hat{\psi}_1 - \hat{\phi}_1 \hat{\psi}_2
\]

= \left\{ \begin{array}{c}
\begin{array}{c}
1 \rightarrow 2 \rightarrow 3 \\
\end{array}
\end{array} \right\}

+ \left\{ \begin{array}{c}
\begin{array}{c}
1 \rightarrow 2 \rightarrow 3 \\
\end{array}
\end{array} \right\}

+ \left\{ \begin{array}{c}
\begin{array}{c}
1 \rightarrow 2 \rightarrow 2 \\
\end{array}
\end{array} \right\}

= [\hat{\phi}_{\text{bundling}}, \hat{\phi}_{\text{treadmill}}] + \text{end J}

- expected
- rare
- energetically disfavored
Why operator algebra yields algorithms

- Baker Campbell Hausdorff theorem
  - $\Rightarrow$ operator splitting algorithms e.g. Trotter Product Formula …
  \[ \lim_{n \to \infty} \left[ e^{(t/n)H_0} e^{(t/n)H_1} \right]^n \]

- Time-ordered product expansions $\Rightarrow$
  Stochastic Simulation Algorithm (SSA)
  – [EMj, Phys Bio 2013]

\[ \exp(t (W_0 + W_1)) = \exp(t W_0) \left( \exp\left( \int_0^t \exp(-\tau W_0) W_1 \exp(\tau W_0) d\tau \right) \right)_+ \]
\[ \equiv \exp(t W_0) \left( \exp\left( \int_0^t W_1 \tau d\tau \right) \right)_+ \]

- weighted SSA (wSSA) possible too
Generation of valid algorithms, continued

Approximate algorithms from Operator Exponentials

\[ \begin{align*}
\varepsilon(t^2) &= e^t \varepsilon + \frac{t^2}{2} \varepsilon^2 B + \frac{t^3}{12} (2\varepsilon A^3 + \varepsilon B^3 + 3 \varepsilon B^2 A^2) + O(t^4) \\
\varepsilon(t^2) &= e^{t A/2} e^t B e^{t A/2} + O(t^4) \\
\varepsilon(t^2) &= e^{t A/2} e^{t A/2} e^{t A/2} + O(t^4) \\
\varepsilon(t^2) &= e^{t A/2} e^{t A/2} e^{t A/2} + O(t^4) \\
D_1 &= \frac{1}{24} [A, [A, B]] - \frac{1}{12} [B, [B, A]] \\
D_2 &= \frac{1}{6} [A, [A, B]] - \frac{1}{12} [B, [B, A]] \\
D_3 &= \frac{1}{6} [A, [A, B]] - \frac{1}{12} [B, [B, A]]
\end{align*} \]

[Zassenhaus]

So commutators are key to analyzing errors and minimizing beyond \(O(t^2)\).
Product Theorems

• Semantics: (compositional)


\[ \hat{W}_r \propto \int d\lambda d\lambda' \rho_r(\lambda, \lambda') \sum_{\langle i_1, \ldots, i_k \rangle \neq \emptyset} \hat{a}_{i_1, \ldots, i_k}(G^r_{\text{out}})a_{i_1, \ldots, i_k}(G^r_{\text{in}}) \]

• Calculate product ...
Product Theorems

- **Semantics:** (compositional)

\[ \hat{W}_r \propto \rho_r(\lambda, \chi') \sum_{i_1, \ldots, i_k} \left[ \prod_{p', q' \in \text{rhs}(r)} (\hat{a}_{i_{p', q'}})^{g_{p', q'}} \right] \left[ \prod_{p' \in \text{lhs}(r)} (\hat{a}_{i_{p', \chi'}})^{h_{p'}} \right] \]

- **Product:**

\[ \hat{W}_{r_2} \hat{W}_{r_1} \propto (\rho_{r_1}(\lambda_1, \chi'_1) \rho_{r_2}(\lambda_2, \chi'_2)) \sum_{\{i_1, \ldots, i_k_1\}} \sum_{\{j_1, \ldots, j_k_2\}} \]

\[ \times \left[ \prod_{p', q' \in \text{rhs}(r_2)} (\hat{a}_{i_{p', q'}})^{g_{p', q'}} \right] \left[ \prod_{p' \in \text{lhs}(r_2)} (\hat{a}_{i_{p', \chi'_2}})^{h_{p'}} \right] \]

\[ \times \left[ \prod_{p, q \in \text{lhs}(r_2)} (a_{i_{p, q}})^{g_{p, q}} \right] \left[ \prod_{p \in \text{lhs}(r_2)} (a_{i_{p, \lambda_2}})^{h_{p}} \right] \]

+ a variant which eliminates dangling edges

**Proposition 1** The product of two operators taking the form of Equation (\(*\)) can be rewritten as a signed-integer-weighted sum of expressions taking the same form. The product and the sum are equal, and graph-equivalent, and each is subpermutation-invariant with respect to indexing.
Computed Products and Commutators

• Computation must yield the form:

\[
\hat{\sigma}_{\gamma_1} \cdot \hat{\sigma}_{\gamma_2} = \sum_{\omega} \left( \omega \in \mathbb{Z} \right) \hat{\sigma}_{\omega}^{(\omega)} \rightarrow \omega^{(\omega)}
\]

\[
\left[ \hat{\sigma}_{\gamma_1}, \hat{\sigma}_{\gamma_2} \right] = \sum_{\omega} \left( \omega \in \mathbb{Z} \right) \hat{\sigma}_{\omega}^{(\omega)} \rightarrow \omega^{(\omega)}
\]

• Cf. Heisenberg & rotation-group Lie algebras

• Particular cases simplify further

  • eg. polymerization, dendromers, etc..
  • analysis for compilation?
Algebra of Labelled-Graph Rewrite Rules

\[
\hat{W}_{G_2^{\text{in}} \rightarrow G_2^{\text{out}}} \hat{W}_{G_1^{\text{in}} \rightarrow G_1^{\text{out}}} \simeq \sum_{H \subseteq G_1^{\text{out}}} \sum_{\tilde{H} \subseteq G_2^{\text{in}}} \sum_{l: H \xrightarrow{1-1} \tilde{H}} \hat{W}_{G_1^{\text{in}} \cup (G_2^{\text{in}} \setminus \tilde{H})} \rightarrow G_2^{\text{out}} \cup (G_1^{\text{out}} \setminus H) \mid \text{edge-maximal}
\]

Algebra of Labelled-Graph Rewrite Rules

\[ \hat{W}_{G_2^{\text{in}} \rightarrow G_2^{\text{out}}} \hat{W}_{G_1^{\text{in}} \rightarrow G_1^{\text{out}}} \approx \sum_{H \subseteq G_1^{\text{out}} \sim \tilde{H} \subseteq G_2^{\text{in}}} \sum_{h : H \xrightarrow{1} \tilde{H}} \hat{W}_{G_1^{\text{in}} \cup (G_2^{\text{in}} \setminus \tilde{H}) \rightarrow G_2^{\text{out}} \cup (G_1^{\text{out}} \setminus H)} \]

\[ G_{1;2}^{\text{in \ nodes}} = G_{1}^{\text{in \ nodes}} \cup (G_{2}^{\text{in \ nodes}} \setminus \tilde{H}_{\text{nodes}}) \]
\[ G_{1;2}^{\text{in \ links}} = G_{1}^{\text{in \ links}} \cup h^{-1}(G_{2}^{\text{in \ links}} \setminus \tilde{H}_{\text{links}}) \]
\[ G_{1;2}^{\text{out \ nodes}} = G_{2}^{\text{out \ nodes}} \cup (G_{1}^{\text{out \ nodes}} \setminus H_{\text{nodes}}) \]
\[ G_{1;2}^{\text{out \ links}} = G_{2}^{\text{out \ links}} \cup h^{*}(G_{1}^{\text{out \ links}} \setminus H_{\text{links}}) \]

\[ K_a = G_{a}^{\text{in \ nodes}} \cap G_{a}^{\text{out \ nodes}} \]
\[ K_{1;2} = (K_1 \setminus H_{\text{nodes}} \cup h^{-1}(K_2 \setminus \tilde{H}_{\text{nodes}}) \cup (K_1 \cap h^{-1}(K_2)) \]

\section*{Product Theorems}

- Double pushout semantics:
  
in the category of graphs

- Commutator=0 condition

\begin{itemize}
  \item L, R = Left, Right graphs;
  \item K = shared graph;
  \item G = input, H = output
  \item Eg:
\end{itemize}

\begin{definition}[van Kampen square] A pushout (1) is a van Kampen square if, for any commutative cube (2) with (1) in the bottom and where the back faces are pullbacks, the following statement holds: the top face is a pushout iff the front faces are pullbacks:
\end{definition}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure.png}
\caption{Example of a van Kampen square with pushouts and pullbacks.}
\end{figure}

\begin{fact}[characterization of parallel and sequential independence] Two direct (typed) graph transformations \( G \xrightarrow{f_1} H_1 \) and \( G \xrightarrow{f_2} H_2 \) are parallel independent iff there exist morphisms \( i : L_1 \to D_2 \) and \( j : L_2 \to D_1 \) such that \( f_2 \circ i = m_1 \) and \( f_1 \circ j = m_2 \):
\end{fact}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{example.png}
\caption{Example of parallel independence with morphisms and transformations.}
\end{figure}
Meta-graph grammar for scalable implementation

- Transformation target for spatially embedded labeled graph rewrite dynamics
- For computational reduction to scalable particle codes?

\[
x, y, z: \text{real-valued params}  
a, b, c: \text{discrete-valued params}  
A, B, C: \text{OIDs}  
\]

\[
\text{particle}(A, a, x) \rightarrow \text{itself under an ODE } | a  
\]

\[
\text{particle}(A, a, x), \ \text{particle}(B, b, y) \rightarrow \text{themselves under an ODE } | a, b \text{ for } x  
\]

\[
\text{particle}(A, a, x), \ \text{particle}(B, b, y), \ \text{link}(A, B)  
\rightarrow \text{themselves under an ODE } | a, b \text{ for } x  
\]

\[
\text{particle}(A, a, x) \leftrightarrow \text{particle}(A, a, x), \ \text{particle}(B, b, y)  
\quad \text{with a propensity depending on } x-y, \ a, b  
\]

\[
\text{particle}(A, a, x) \leftrightarrow \text{null with a propensity depending on } x  
\quad \text{(null is non-modeled stuff - but violates conservation)}  
\]

\[
\text{particle}(A, a, x), \ \text{particle}(B, b, y)  
\leftrightarrow \text{particle}(A, a, x), \ \text{particle}(B, b, y), \ \text{link}(A, B)  
\quad \text{with a propensity depending on } x-y, \ a, b  
\]

\[
\text{particle}(A, a, x), \ \text{particle}(B, b, y), \ \text{link}(A, B)  
\leftrightarrow \text{particle}(A, a, x), \ \text{particle}(B, y), \ \text{link}(A, B), \ \text{link}(B, A)  
\]

\[
\text{particle}(A, a, x), \ \text{particle}(B, b, y), \ \text{link}(A, B), \ \text{particle}(C, z), \ \text{link}(B, C)  
\leftrightarrow \text{particle}(A, a, x), \ \text{particle}(B, b, y), \ \text{link}(A, B), \ \text{particle}(C, c, z),  
\quad \text{link}(B, C), \ \text{link}(C, A) \quad \text{with a propensity } (x-y, y-z, z-x | a, b, c)  
\]

other local graph grammar rules
Summit Architecture
(#1 in 2018-9)

- Each node:
  - 2 x 22 cores/CPU ~1 TFlops
  - 6xV100 GPU ~47 TFlops

- 4608 nodes
  - ~200 PFlops
  - ~340 tons

https://en.wikichip.org/wiki/supercomputers/summit
“Cabana” particle sim can be fast

Aaron Scheinberg and XGC team
void evolve_particle_damped(particle_list_t& particles, size_t i)
{
    auto type = particles.slice<Type>();
    auto force_type_A = particles.slice<Type>()(i);
    auto velocity = particles.slice<Velocity>();
    auto position = particles.slice<Position>();
    auto length = particles.slice<Length>();

    if (force_type_A == positive)
    {
        // ith particle, property j (0..2)
        i_1 = nbr_interior.i; j_1 = nbr_interior.j;
        velocity[i,j] = v_plus * (1-length(i_1)/length_max) * u[i,j];
        position[i, j] += velocity(i, j) * delta; //?? + length(i);
        for all nearby other fibers k {
            alpha = - 2dcross((...), (...))/2dcross((...), (...)) ; // 2d cross product
            gamma = - 2dcross((...), (...))/2dcross((...), (...)) ; // 2d cross product
            // directional derivative of kappa * exp(- gamma^2/(2*epsilon^2)):
            velocity[i,j] += kappa * u[i,j] * (-gamma/epsilon^2) * exp(- gamma^2/(2*epsilon^2));
            elongation_speed[i] += v_plus * (1-length(i)/length_max) ;
        }
        position[i, j] += velocity(i, j) * delta; //?? + length(i);
    }
    else if (force_type_A == negative)
    {
        // ith particle, property j (0..2)
        i_1 = nbr_interior.i;
        velocity[i,j] = v_minus * (length(i_1)/length_max) * u[i,j];
        position[i, j] += velocity(i, j) * delta; //?? + length(i);
    }
    else if (force_type_A == intermediate)
    {
        // ith particle, property j (0..2)
        i_1 = nbr_interior.i; j_1 = nbr_interior.j;
        velocity[i,j] = v_plus * (1-length(i_1)/length_max) * u[i,j];
        position[i, j] += velocity(i, j) * delta; //?? + length(i);
        for all nearby other fibers k {
            alpha = - 2dcross((...), (...))/2dcross((...), (...)) ; // 2d cross product
            gamma = - 2dcross((...), (...))/2dcross((...), (...)) ; // 2d cross product
            // directional derivative of kappa * exp(- gamma^2/(2*epsilon^2)):
            velocity[i,j] += kappa * u[i,j] * (-gamma/epsilon^2) * exp(- gamma^2/(2*epsilon^2));
            elongation_speed[i] += v_plus * (1-length(i)/length_max) ;
            elongation_speed[i] += v_minus * (length(i_1)/length_max) ;
        }
        position[i, j] += elongation_speed(i)*delta;
    }
    else if (force_type_A == junction)
    {
        // ith particle, property j (0..2)
    }
    else {
        std::cout << " ??? " << std::endl;
    }
}
Cajete MT: First Light

Eric Medwedeff, UCI
Cajete MT: First Light

Eric Medwedeff, UCI
Eg: Plant gene expression model
Declarative, with cell growth & division

\[
\begin{align*}
&\{\emptyset \rightarrow U, k_1 TIP[t]\}, \{U \rightarrow \emptyset, k_2\}, \{U \rightarrow U, \text{Diffusion}[D_U]\}, \\
&\{\emptyset \rightarrow V, k_3 L_1[t]\}, \{V \rightarrow \emptyset, k_4\}, \{V \rightarrow V, \text{Diffusion}[D_V]\}, \\
&\{\emptyset \equiv Z, k_7, k_8 U[t]\}, \{X \equiv V, \text{GRN}[v_v, T_{vw}, 1, h_v]\}, \\
&\{U, V, W \equiv W, \text{GRN}[v_w, \{T_{uw}, T_{vw}, T_{ww}\}, 1, h_w]\}, \{W \rightarrow \emptyset, k_6 Z[t] + k_9 L_2[t]\} \\
&W \equiv X, \text{GRN}[v_x, T_{wx}, 1, h_x]\}, \{X \rightarrow \emptyset, k_5\}, \{X \rightarrow X, \text{Diffusion}[D_x]\}, \\
&\text{cell} \rightarrow \text{cell}, \text{Grow}[\text{GrowthRate}[\mu, f_\mu], \text{Pressure}[P, f_P], \text{Spring}[k, f_k]], \\
&\text{cell} \rightarrow \text{cell} + \text{cell}, \text{Errera}[\text{cell}, \mu, \sigma}\}
\end{align*}
\]

Cf. L-systems:

[Shapiro et al Frontiers in Plant Science 2013]
Dynamical Grammar example: Root growth

**Cell division**

\[
\{\text{Cell}(x_i, r_i, m_i = 2, a_i, y_i)\} \rightarrow \begin{cases} 
\text{Cell}(x_i, r_i, m_i = 1, a_i, y_i), & \text{Cell}(x_{i+1}, r_{i+1}, m_{i+1} = 1, a_{i+1}, y_{i+1}) \\
\text{spring}(c_i, c_{i+1})\end{cases} \\
\text{s}_{i+1} = \text{spring}(c_i, c_{i+1}) \rightarrow \{c_i, c_{i+1}, s_{i+1}\}
\]

with
\[
\rho_{dv}(y_i) = \left(\frac{y_i}{k_{dv,1}}\right)^{a_{dv,1}}\left(1 + \left(\frac{y_i}{k_{dv,2}}\right)^{a_{dv,2}}\right)
\]

**Active auxin transport**

\[
\{c_i = \text{Cell}(x_i, r_i, m_i, a_i, y_i), \quad c_{i+1} = \text{Cell}(x_{i+1}, r_{i+1}, m_{i+1}, a_{i+1}, y_{i+1})\}, \quad s_{i+1} = \text{spring}(c_i, c_{i+1}) \rightarrow \{c_i, c_{i+1}, s_{i+1}\}
\]

solving
\[
\frac{da_{i+1}}{dt} = -K_0 a_i b(a_i), \quad \frac{da_i}{dt} = K_0 a_i b(a_i)
\]

**Auxin flow from the shoot**

\[
\{c_N = \text{Cell}(x_N, r_N, m_N, a_N, y_N)\} \rightarrow \{c_N\}
\]

solving
\[
\frac{dc_N}{dt} = \alpha_{mit} + \frac{0.17t}{\text{CellCycleTime}}
\]

**Hypothetical substance Y**

solving
\[
\frac{dy_i}{dt} = -y_i(K_{d,y}(a_i) + \frac{v(r_i)}{r_i}), \quad \frac{da_i}{dt} = v(r_i)
\]

\[
K_{d,y}(a_i) = k_{d,y}^0 \left(1 + \left(a_i \frac{k_{d,y}}{a_{d,y}}\right)^{a_{d,y}}\right) \left(1 + \left(\frac{r_i}{k_{d,y}^2}\right)^{a_{d,y}}\right)
\]

Symbolic transformation: \{\text{Reaction}\} \rightarrow \{\text{ODE}\}

- This can be done by meta-rules, in a meta-grammar
- As can many modeling-language extensions & translations
Symbolic model transformations: endless possibilities

• Meta-rules for transforming dynamics rules
  ✓ e.g. Reactions → ~ODEs
  ✓ e.g. detailed balance by arrow reversal
    – generation of ML algorithms from models, > autodiff

• Model reduction by ML (linear combination)
  • structural discovery of fast modes

• Reduction to spatial graph dynamics
  • e.g. adaptive grids by graph rewrite rule

✓ emergent dynamical structures: tissue, cytoskeleton, …
Fields to Structures

- Dynamical Graph Grammars (DGGs):
  - operator addition of reactions, GGs, ODEs;
  - but what about PDEs?

- Fields: PDE differential operator dynamics in $W$

- Approximately eliminate fields by:
  - Cell complexes in PDE (adaptive) meshing / FEMs, FVMs
Geometric meshing: protective manifolds

(a) Initial PLC  (b) PSC with intestine
(c) Initial collar  (d) Final collar  (e) Initial intestine  (f) Final intestine

[Rand and Walkington 2009]
Cf. [Murphy, Mount, & Gable 2001; Engwirda 2016]
Graph Grammars for 2D meshes

• Triangular:

\[
\begin{align*}
&\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
1
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
2
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
3
\end{array}
\end{array}
\end{array}
\quad\langle l_1, l_2, l_3, l_4 \rangle
\end{align*}
\rightarrow
\begin{align*}
&\begin{array}{c}
\begin{array}{c}
1
\end{array}
\begin{array}{c}
4
\end{array}
\begin{array}{c}
6
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
2
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
5
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
3
\end{array}
\end{array}
\end{array}
\quad\langle l_1, l_2, l_3, l_4, \text{max}(l_1, l_2, l_3, l_4) \rangle
\end{align*}
\end{align*}
\]

• Cuboid:

\[
\begin{align*}
&\begin{array}{c}
\begin{array}{c}
1
\end{array}
\begin{array}{c}
2
\end{array}
\begin{array}{c}
3
\end{array}
\end{array}
\quad\langle l_1, l_2, l_3 \rangle
\end{align*}
\rightarrow
\begin{align*}
&\begin{array}{c}
\begin{array}{c}
1
\end{array}
\begin{array}{c}
4
\end{array}
\begin{array}{c}
6
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
2
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
5
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{c}
3
\end{array}
\end{array}
\end{array}
\quad\langle l_1, l_2, l_3, \text{max}(l_1, l_2, l_3) + 1, \text{max}(l_1, l_2, l_3) + 1 \rangle
\end{align*}
\end{align*}
\]

Higher level rewrite rules

- Identify strata

\[ G \xrightarrow{h} G_S \]
\[ \chi_G \xrightarrow{\chi_{G_S}} \tilde{J}_D \]

(\text{Diag})

each inverse image \((\chi_{G_S}^{-1})(d)\) must be a fully disconnected

- Operator algebra semantics for strata and other slices

\[ \rho_{\text{graph }, r}((\kappa, \lambda), (\kappa', \lambda')) = \Theta(P_H(\kappa)) \times \Theta(P_H(\kappa')) \times \rho_{\text{slice }, H, r}(\lambda, \lambda') \]

\[ \mathcal{I}_m \]

Extended objects via slices

using graph homomorphisms

\[ G \xrightarrow{h} G' \]

\[ \varphi_G \quad \varphi_{G'} \]

\[ H = \tilde{\mathbb{N}}, \quad \tilde{J}_D, \quad \tilde{\mathbb{N}}_D^{\text{op}}, \quad \tilde{C}_D, \quad \text{or} \quad \tilde{\mathcal{C}}_D. \]

\( \tilde{\mathbb{N}} = (\mathbb{N}, \text{Successor}) \)

- nonnegative integers \( \{0, 1, \ldots\} \) as vertices,
- with (possibly directed) edges from each integer \( i \) to its immediate successor \( i + 1 \) and to itself;

\[ \tilde{J}_D = \begin{cases} \mathbb{N}_D = \{0, \ldots, D \geq 0\}, \to & \text{directed graphs}; \\ = \text{integers } \{0, \ldots, D\} \text{ with } (i, j) \text{ edge iff } i \geq j; \\ \tilde{K}_D = \tilde{K}_{\{0, \ldots, D\}} \text{ (fully connected w. self-edges)} & \text{undirected graphs} \end{cases} \]

\[ \tilde{\mathbb{N}}_D^{\text{op}} = \begin{cases} \text{integers } \{0, \ldots, D\} \text{ with } (i, j) \text{ edge iff } i = j + 1 \text{ or } i = j & \text{directed graphs}; \\ \text{integers } \{0, \ldots, D\} \text{ with } (i, j) \text{ edge iff } |i - j| \leq 1 & \text{undirected graphs} \end{cases} \]

\[ C_D = \tilde{\mathbb{N}} \sqcup \tilde{J}_D \]

\[ \tilde{\mathcal{C}}_D = \tilde{\mathbb{N}} \sqcup \tilde{\mathbb{N}}_D^{\text{op}} \]

Operator Algebra variants:

\[ \rho_{\text{graph}} r((\kappa, \lambda), (\kappa', \lambda')) = \Theta(P_H(\kappa)) \times \Theta(P_H(\kappa')) \times \rho_{\text{slic}} H, r(\lambda, \lambda') \]

Stratified space of MTs:

Antitubulin labelling in premitotic epidermal cells

\textit{Datura stramonium}


Graded graph

Stratified graph

Abstract cell complex

Graded stratified graph

Graded abstract cell complex

Stratified spaces, not cell complexes, are necessary for cytoskeleton

Left: Antitubulin labelling in premitotic epidermal cells *Datura stramonium*  

Above: Antitubulin labelling at intact cell cortex  
[DeBolt et al., PNAS 2007 supplementary info figure 8A.]

[Smith, Nat Rev MCB 2 2001]
Graph Lineage Definitions

• **Hierarchical Graph Sequence**: a mapping from \( \mathbb{N} \) into some sequence of graphs which obeys the following:
  - \( G_0 \) is the graph with one vertex and one loop on that vertex
  - Edge and vertex cardinality of graphs in the sequence grow at most “exponentially” in some base, \( b \):
    \[ O(b^{1+\epsilon}) \]

• **Graded Graph**: \( G \) is a graded graph if all of the vertices of \( G \) are labeled with non-negative integers such that if \((v_1, v_2)\) is an edge, the labels of \( v_1 \) and \( v_2 \) differ by at most 1.

• **Graph Lineage**: a graded graph where the sequence of \( \Delta L = 0 \) subgraphs is a HGS and the subgraphs with \( \Delta L = 1 \) are a HGS of bipartite graphs. The above is a graph lineage of path graphs of length \( 2^n \).

• **Hierarchitecture**: A graph lineage, used as a model architecture.
Generating Graph Lineages

- One way to generate a graph lineage (or more generally, graded graphs) is via local graph rewrite rules.

- Rules can be applied locally, or to all cells in a graph simultaneously:

  Local Firing
  - Graph labels suppressed, but necessary
  - More:

  Global Firing
Multiscale numerics:
Alg. Multigrid Methods for Graphs

\[ G' \simeq P^T G P \]
Define Graph Process
Directed “Distances”

• Definition requires constrained opt of diffusion operator:

\[
\begin{align*}
D(G_1, G_2|R, \alpha > 0, t) &= \inf_{P|C(P)} ||P\exp(\alpha^{-1/2}tW_1^{(R)}) - \exp(\alpha^{1/2}tW_2^{(R)})P||_F \\
D(G_1, G_2|R, t) &= \inf_{\alpha > 0} D(G_1, G_2|R, \alpha, t)
\end{align*}
\]

• Constraints: orthogonality; sparsity?

\[
C(P) : \quad P^TP = I \quad \text{; restriction, prolongation} ; \quad \max \text{ fanout}(P) \leq (n_{P_{\text{fine}}}/n_{P_{\text{course}}})^s
\]

• Optimize time & time dilation due to graph size:

\[
\tilde{D}(G_1, G_2|R) = \sup_{t > 0} \inf_{\alpha > 0} D(G_1, G_2|R, \alpha, t)
\]

• Can obtain \( P \) at early times (“rigid” vs “flexible” def of \( D \)):

\[
D_{\text{rigid}}(G_1, G_2|R, t) = \inf_{P|C(P)} ||P^*\exp(\alpha^{*}-1/2tW_1^{(R)}) - \exp(\alpha^{1/2}tW_2^{(R)})P^*||_F , \text{ where} \\
(a^*, P^*) = \text{argmin}_{\alpha > 0, P|C(P)} ||\alpha^{-1/2}PW_1^{(R)} - \alpha^{1/2}W_2^{(R)}P||_F
\]

• \( \triangle \leq \) provable with weaker \( \alpha \) : \quad \alpha = \left( \frac{n_1}{n_2} \right)^r

Graph Distance Experiments

- Triangle inequality

- Graph limits

with Cory Scott
MS in prep

Key data type: Stack of models
w. conditional reductions, each model on the spectrum:

• pure chemical reactions
• parameterized object rewrite rules
  – propensity functions
  – differential equations (ordinary, stochastic, delay)
• graph grammar rewrite rules
• graph-limit rewrite rules
  – support PDEs on $R^n$, manifolds, CCs, SSs
• sub-grammar calls, macros, types/inheritance

Epilogue

- Interlevel mappings in “morphodynamics”/dev bio modeling are central to: AI for bio
- Such model reductions can be specified, curated, optimized and learned computationally
  - optimized and learned: Dynamic Boltzmann Distributions, GCCD, machine learning methods
  - specified: ~Dyn Graph Grammar high level languages + graph limits. Microtubule, cell tissue models as test cases.
  - curated: Tschicoma conceptual architecture; Cajete scalable prototype
- Comments? Want to help? emj@uci.edu.
“Tchicoma” Architecture for Mathematical Modeling

• Language meta-hierarchy:  (a DAG with edge labels in a tree)

• Mappings therein: respecting compositional structure

   Enables problem-solving via chaining, theorem-proving
   Foments abstraction via commutation

Conclusions

• Biological model reduction can be achieved by machine learning, in spatial stochastic models (and easier ones). Reaction/diffusion examples.

• Morpho-dynamic spatial structures (and easier models) can be modeled by dynamical graph grammars with operator semantics. Bio-universal; scalability is in progress. MT examples.

• Model stacks are the key data structure for understanding complex bio systems. They require model reduction and bio-universal modeling languages (perhaps as above). They can intersect productively, and could be curated in a proposed conceptual architecture “Tchicoma”.

• Declarative modeling languages with operator algebra semantics can support generic model reduction, hence model stacks.

• In these ways, both symbolic and numeric AI can be brought to bear on understanding complex biological systems at their own scale.
A change of view

• Human, physics-centric viewpoint:

• Computer viewpoint: