AI and Theorem-Proving for Science:

Towards AI for mathematical modeling of complex physical and biological systems

with aspects of "Feynman on AI …" book chapter

Eric Mjolsness

University of California, Irvine

http://emj.ics.uci.edu

Artificial Intelligence and Theorem Proving (AITP) Conference
September 2023

Some aspects of the "Feynman on AI" chapter

- Back-story: lunchtime student+TA discussions with RPF in the early to mid-1980s, often including EM, MD
- in which RPF evinced an interest in a machine learning approach to AI, with a progressive sequence of tasks
- [EM, Feynman on Artificial Intelligence and Machine Learning.] - my considered version
- Includes tutorial derivations omitted here

[EM, Feynman on Artificial Intelligence and Machine Learning.
Feynman Lectures on Computation: Anniversary Edition; arXiv:2209.00083]

Symbolic algebraic expressions:
Key to new computing paradigms, and a hidden currency of ML

- Gibbs-Bogolubov-Feynman variational method
  - $F \leq \langle H \rangle_0 - TS_0$
- Hopfield neural network
  - $F \leq E_{\mathrm{MFT}}[v] \equiv -\frac{1}{2}\sum_{i\neq j} T_{ij} v_i v_j - \sum_i h_i v_i + \sum_i \varphi(v_i), \qquad v_i \equiv \langle s_i \rangle_0 \in \{0,1\}$.
  - $\varphi(v) = \frac{1}{\beta}\Big(v \log v + (1-v)\log(1-v)\Big) = -TS_{\mathrm{MFT}}[v]$ => activation $v_i = \tanh\Big[\beta\Big(u_i = \sum_j T_{ij} v_j + h_i\Big)\Big]$.
  - Generalized to multiway 0/1 choice: $\varphi([v_{i\star}]) = \frac{1}{\beta}\sum_a v_{ia} \log v_{ia} = -T\tilde{S}[v]$ => softmax $v_{ia} = e^{\beta u_{ia}} \Big/ \sum_b e^{\beta u_{ib}}$.
- Boltzmann Machine $H_{\mathrm{Ising}}[s] \equiv -\frac{1}{2}\sum_{i\neq j} T_{ij} s_i s_j - \sum_i h_i s_i$; learn by minimizing KL divergence to external p(s).
- Quantum computer, per Feynman $H = \sum_{i=0}^{k-1} q_{i+1}^* q_i A_{i+1}$.
- Transformer neural network: Inner softmax $v_{ia} = e^{\beta u_{ia}} \Big/ \sum_b e^{\beta u_{ib}}$

[EM, Feynman on Artificial Intelligence and Machine Learning.
Feynman Lectures on Computation: Anniversary Edition; arXiv:2209.00083]

Symbolic vs. Neural AI

- Feynman lunchtime discussions, early 1980s:
  - G. Sussman *(Minsky colleague)* vs. (ideas of) J. Hopfield
    - Feynman expressed a NN research program
- Symbolic was ascendant, but experiencing AI winter
- Now reversed:
  - Last 10-15 years: AAAI dominated by ML/neural networks
- Future: Hegelian Synthesis?

[EM, Feynman on Artificial Intelligence and Machine Learning.
Feynman Lectures on Computation: Anniversary Edition; arXiv:2209.00083]

Quoting my own best-effort text …

- "What would Feynman have thought of all all these advances? Remembering his commitment to having a personal, independent point of view on everything, and the impossibility of emulating a great mind, there can be little certainty on this point. Nevertheless it seems to me that: …"

  - "3. He would be enthusiastic about transfer learning, and about the particular neural architectures that have led to success in vision, natural language, and limited combinations thereof. These methods have essentially achieved the goals of the machine learning project that he had in mind - insofar as he had expressed them."

Foundation Models; Transformers

  - "4. Regarding equations as the hidden currency of learning architecture, he might be of two minds: attracted because he was a mathematical master; repelled because the source of his mastery was the ability to deeply visualize the meaning of each equation. Neural network equations for the most part are just not that conceptually deep. In the old AI dichotomy of "scruffy" vs. "neat" research approaches, ML equations might be neat but not neat enough."

Symbolic Applied Math

[EM, Feynman on Artificial Intelligence and Machine Learning.
Feynman Lectures on Computation: Anniversary Edition; arXiv:2209.00083]

Quoting my own best-effort text …

- "What would Feynman have thought of all all these advances? …" <…Caveats! …>

  - "6. He might nevertheless be intrigued by conceptually deeper, less *ad hoc* architectures that connect to physics, such as manifold learning (clearly related to general relativity), and perhaps by attempts to connect neural networks to real neurobiology."

PIML

  - "8. Because of his deep commitment to physics, Feynman would be quite interested in - if also a bit skeptical of - the "physics-informed machine learning" agenda which is now being pursued with many different architectures, representations, and methods for the purpose of doing computational physics, chemistry, and biology. If I had to pick one thing about present-day neural networks, machine learning, and AI that Richard Feynman would be most interested in, this would be it."

  - 9. He would *certainly* have creative and potentially powerful new ideas that are not yet on anybody's list or agenda."

[EM, Feynman on Artificial Intelligence and Machine Learning.
Feynman Lectures on Computation: Anniversary Edition; arXiv:2209.00083]

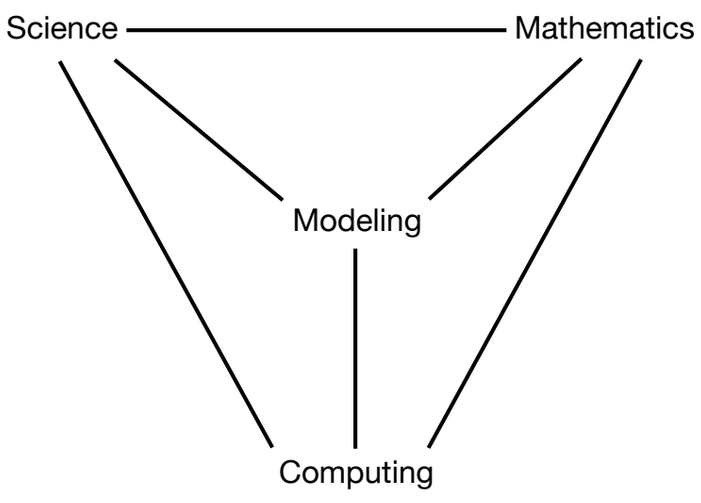
AITP for Science:

some logic in favor

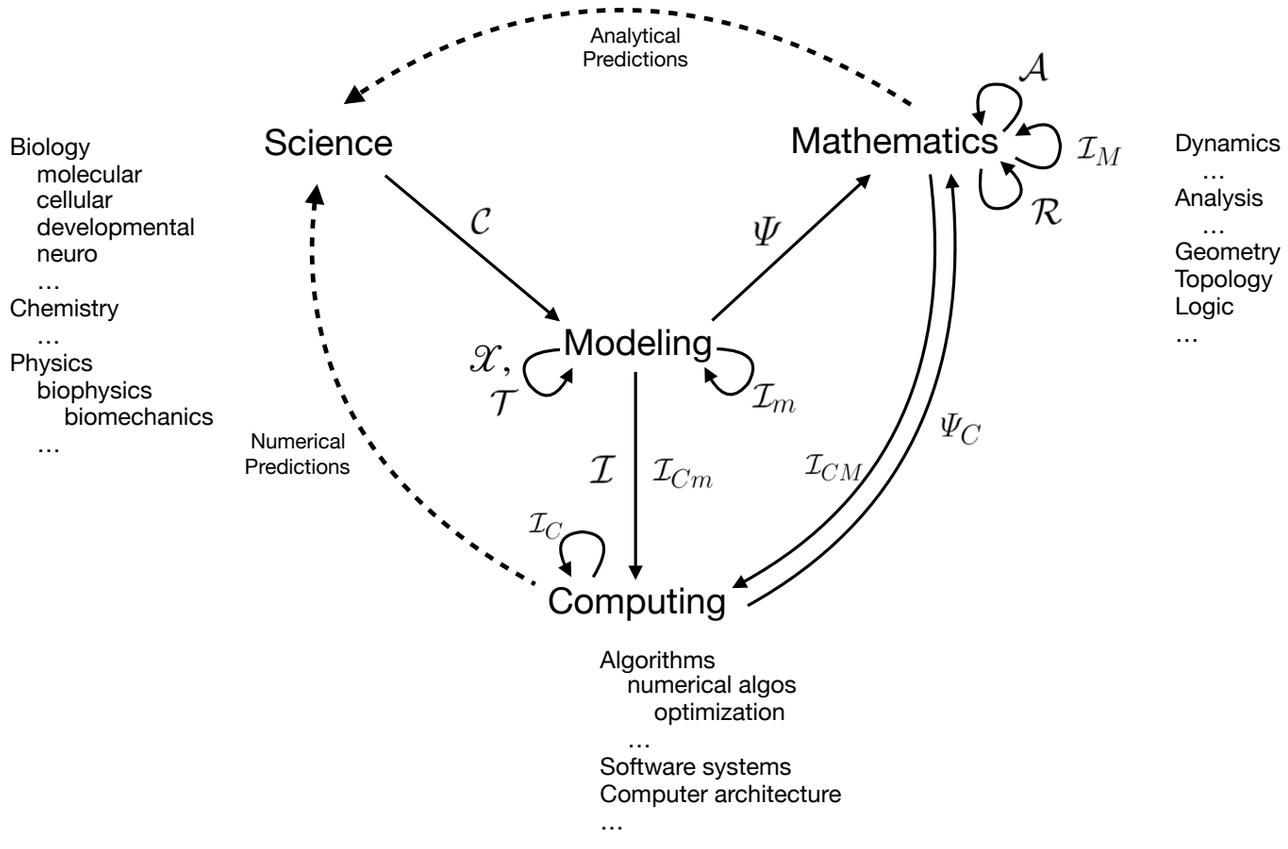
- “AI”: Reliable AI for Science:
  - Representationalism 1st, ML 2nd. (On top vs. on tap)
  - *formal, symbolic* DSLs for modeling …
    - applicable math, sci content, algorithms (model sim, ML, analysis)
- “TP”: verify exact and approximate maps between such languages
  - e.g. dynamics to simulation algs: XDEs, Markov jump processes, hybrids e.g. *dynamical grammars*
  - high-level support for scientific knowledge representation

"Tchicoma" Architecture for Mathematical Modeling

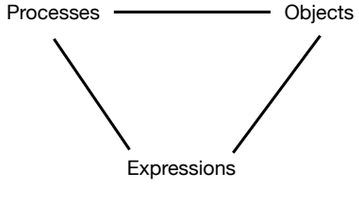
To formalize:



+ mappings and hierarchies:



Each language has:



[EM, Bull. Math Biol. 81:8 Aug 2019
+arXiv:1804.11044]

AITP for Science:

some logic in favor

- "AI": Reliable AI for Science:
  - Representationalism 1st, ML 2nd. (ML on tap, not on top)
  - *formal, symbolic* DSLs for modeling …
    - applicable math, sci content, algorithms (model sim, ML, analysis)
- "TP": verify exact and approximate maps between such languages
  - e.g. dynamics to simulation algs: XDEs, Markov jump processes, hybrids e.g. *dynamical grammars*
  - high-level support for scientific knowledge representation

Some scientific simulation paradigms

- Partial differential equations (PDEs)
  - central to PIML so far
- Stochastic models
  - kinetic theory in physics
  - well-motivated in biology

PDE verification?

- ODEs: Lorenz attractor [Immler 2017]

- Finite element method foundation: Lax-Milgram theorem on unique existence of FEM weak formulation solutions

Master Equation Semantics

$$A_1(x_1),\ A_2(x_2),\ \ \ldots,\ A_n(x_n) \rightarrow B_1(y_1),\ B_2(y_2),\ \ \ldots,\ B_m(y_m)\ \textbf{with}\ \rho(\{x_i\},\ \{y_j\})$$

- RHS, LHS are multisets
- Founded on **stochastic** processes
- Dynamics from the **Master Equation**:

$$\frac{d\mathbf{p}}{dt} = \left(\sum_{processes,r} W_r\right)\cdot \mathbf{p}$$

- Create elementary processes from yet more elementary "Basis operators"
  - Term creation/annihilation operators: for each param value,

$$\hat{a} = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots \\ 1 & 0 & 0 & 0 & \\ 0 & 1 & 0 & 0 & \\ 0 & 0 & 1 & 0 & \\ \vdots & & & \ddots & \ddots \end{pmatrix} = \delta_{n,m+1} \text{ and } a = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots \\ 0 & 0 & 2 & 0 & \\ 0 & 0 & 0 & 3 & \\ 0 & 0 & 0 & 0 & \ddots \\ \vdots & & & & \ddots \end{pmatrix} = m\,\delta_{n+1,m}$$

  - Obeying Heisenberg **operator algebra**

$$[a,\ \hat{a}] \equiv (a\ \ \hat{a} - \hat{a}\,a) = I$$

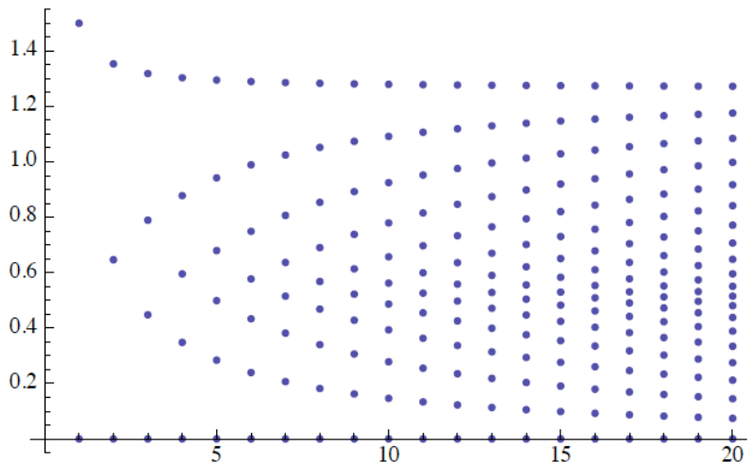
$$[a(x),\ \hat{a}(y)] = \delta(x-y)\left[I + N\,Q\left(N \,\middle|\, n^{(\max)}\right)\right]$$

  - Yet classical, not quantum, probabilities

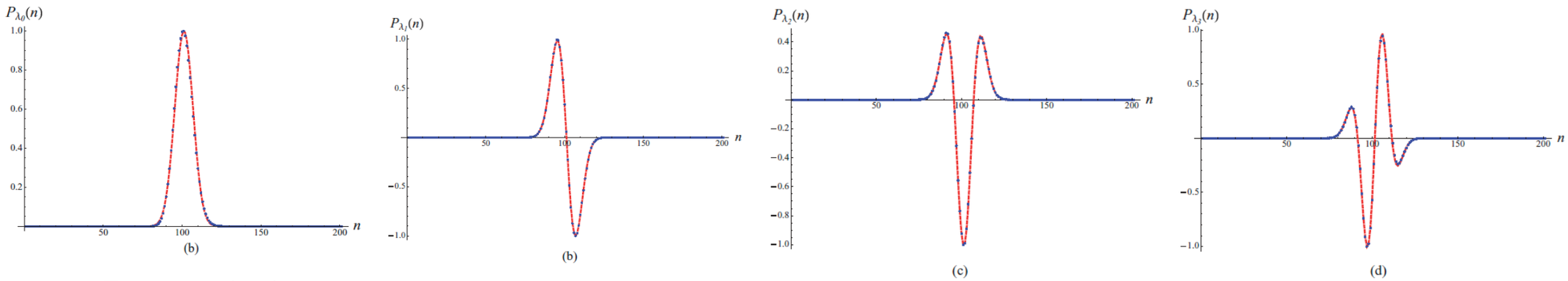
[Doi 1976; ... Mikhailov '81; ...Mattis & Glasser '98; ... EM+GY Annals of Math. and A. I., 47(3-4), January 2007]

Example: $\left\{A + B \xrightarrow{k_f} C,\ C \xrightarrow{k_r} A + B\right\}$

- Operator: $H = k_f\,(\hat{a}_3\, a_1\, a_2 - N_1\, N_2\, I_3) + k_r\,(\hat{a}_1\, \hat{a}_2\, a_3 - I_1\, I_2\, N_3)$
- Spectrum:



- EV's with boundary layer approximation:



Prlambda(nC)

[Mjolsness and Prasad, J. Chem Phys. 2013]

Ψ

Operator algebra for Pure stochastic chemical reactions

- For reaction/rule r:

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

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

$$a_\alpha \hat{a}_\beta = \hat{a}_\beta a_\alpha + \delta_{\alpha\beta} I_\alpha$$

$$n_\alpha \in \{0,1\}: \quad a_\alpha \hat{a}_\beta = (1-\delta_{\alpha\beta}) \hat{a}_\beta a_\alpha + \delta_{\alpha\beta} Z_\alpha$$

- For reaction/rules r_1 and r_2:

where $(n)_l \equiv \begin{cases} n!/(n-l)! & \text{for } l \le n; \\ 0 & \text{otherwise.} \end{cases}$

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

Why: $\partial_x^m (x^n f(x)) =$ binomial sum (+ Weyl algebra)

Generation of valid algorithms

- Compute or sample exp $t\,H$; e.g. Euler's formula

$$e^{t\,H} = \lim_{n\to\infty}\left(1+\frac{t}{n}H\right)^n$$

- Approximate: Trotter Product formula (interleaving):

$$= \lim_{n\to\infty}\left[e^{(t/n)\,H_0}\;\; e^{(t/n)\,H_1}\right]^n$$

- Time-ordered product expansion (TOPE) - convergent Dyson series :

$$e^{t\,H} = \sum_{k=0}^{\infty}\left[\int_0^t dt_k \int_0^{t_k} dt_{k-1}\cdots\int_0^{t_2} dt_1\,\exp((t-t_k)\,H_0)\,H_1\,\exp((t_k-t_{k-1})\,H_0)\cdots H_1\,\exp(t_1\,H_0)\right]$$

[Annals of Math. and A. I., 47(3-4), January 2007]

  - can be used recursively !
  - restate with time-ordering:

$$\exp(t\,(W_0+W_1)) = \exp(t\,W_0)\left(\exp\left(\int_0^t \exp(-\tau\,W_0)\,W_1\,\exp(\tau\,W_0)\,d\tau\right)\right)_+$$

$$\equiv \exp(t\,W_0)\left(\exp\left(\int_0^t W_1\,(\tau)\,d\tau\right)\right)_+$$

[Physical Biology 10, 2013]

- Diagonal/off-diagonal split: obtain *Gillespie's Stochastic Simulation Algorithm* (SSA)

$$\mathcal{W}(I,\,t'\mid J,\,t) \approx \hat{W}_{I,J}\,\exp(-(t'-t)\,D_{J\,J})\,\mathbf{1}\,(t'\geqslant t)$$

$$\Pr(.\mid J,\,k) = \mathcal{W}^k \circ \Pr(.\mid J,\,0) = \left[\hat{W}\,\exp(-\Delta\,t\,D)\,\mathbf{1}\,(\Delta\,t\geqslant 0)\right]^k \circ \Pr(.\mid J,\,0)$$

TOPE and Feynman diagrams

Time-Ordered Product Expansion

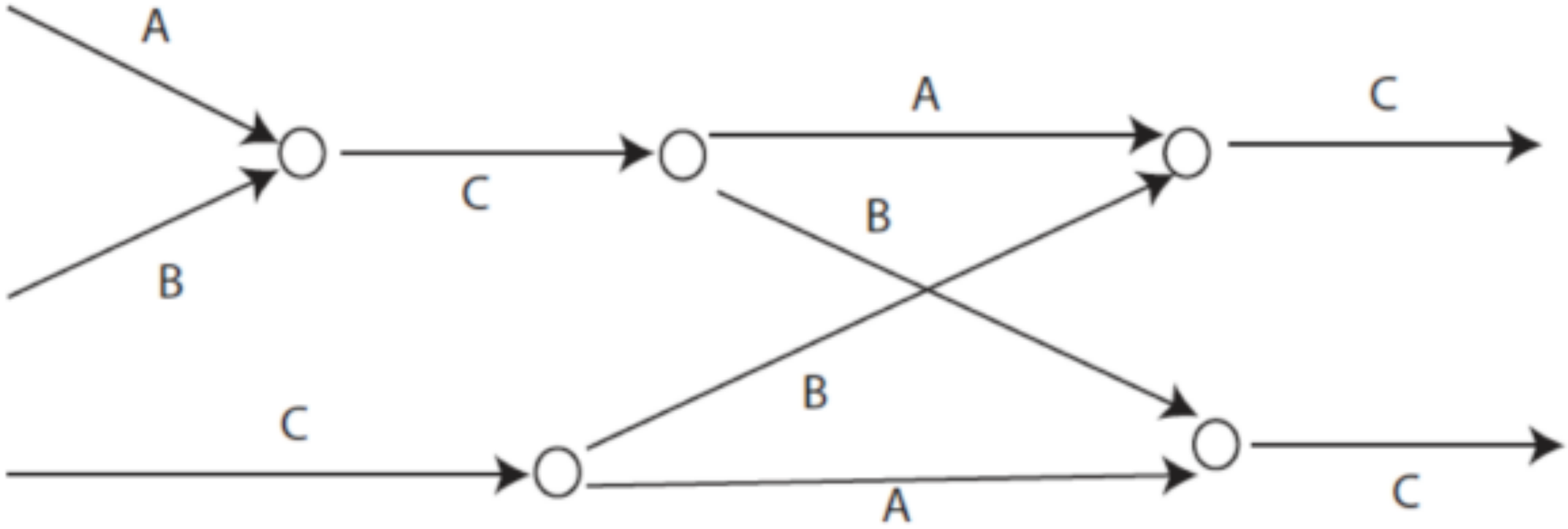


Figure 1. A time history of the reaction $A + B \rightleftharpoons C$. Time flows left to right. Solid blocks represent reaction events, with probability factor $\times W_1$. In between reaction events are unimolecular particle propagators $\exp((t_k - t_{k-1})\, W_0)$, labelled by arrows and particle names (repeated for clarity). This is a non-spatial version of the Lee model in quantum field theory (cf. for example [Bender et al. 2005]).

[Physical Biology 10, 2013]

Time-Ordered Product Expansion (TOPE) verification? 1/2

- "Big operator" derivation
- formal power series, changes of variable, induction on dimension
- notation "represents" permutation symmetries

Line #

$$S(z) = \exp(t\,(H_1 z + H_0)) \cdot p_0 = \sum_{n=0}^{\infty} s_k z^k$$

$$= \sum_{k=0}^{\infty} \frac{z^k}{k!} \left[\partial_z{}^k \exp(t\,(H_1 z + H_0)) \right]_{z=0} \cdot p_0$$

$$= \sum_{k=0}^{\infty} \frac{z^k}{k!} \left[\partial_z{}^k \sum_{l=0}^{\infty} \frac{(t\,(H_1 z + H_0))^l}{l!} \right]_{z=0} \cdot p_0 \qquad (3)$$

$$= \sum_{k=0}^{\infty} \frac{z^k}{k!} \left[\sum_{l=k}^{\infty} \frac{1}{l!} \sum_{\{0 \leqslant i_p \leqslant l-k\} \wedge \sum_{p=0}^{k} i_p = l-k} k! (tH_0)^{i_k} t\hat{H} (tH_0)^{i_{k-1}} \cdots t\hat{H} (tH_0)^{i_0} \right] \cdot p_0 \qquad (4)$$

$$= \sum_{k=0}^{\infty} z^k t^k \left[\sum_{l=0}^{\infty} \frac{1}{(l+k)!} \sum_{\{0 \leqslant i_p \leqslant l\} \wedge \sum_{p=0}^{k} i_p = l} (tH_0)^{i_k} H_1 (tH_0)^{i_{k-1}} \cdots H_1 (tH_0)^{i_0} \right] \cdot p_0$$

$$= \sum_{k=0}^{\infty} z^k t^k \left[\sum_{l=0}^{\infty} \sum_{\{0 \leqslant i_p \leqslant l\} \wedge \sum_{p=0}^{k} i_p = l} \frac{\prod_{p=0}^{k} (i_p)!}{\left(\sum_{p=0}^{k} i_p + k \right)!} \frac{(tH_0)^{i_k}}{(i_k)!} H_1 \frac{(tH_0)^{i_{k-1}}}{(i_{k-1})!} \cdots H_1 \frac{(tH_0)^{i_0}}{(i_0)} \right] \cdot p_0$$

$$= \sum_{k=0}^{\infty} z^l t^l \left[\sum_{\{0 \leqslant i_p \leqslant \infty\}} \frac{\prod_{p=0}^{k} (i_p)!}{\left(\sum_{p=0}^{k} (i_p + 1) - 1 \right)!} \frac{(tH_0)^{i_k}}{(i_k)!} H_1 \frac{(tH_0)^{i_{k-1}}}{(i_{k-1})!} \cdots H_1 \frac{(tH_0)^{i_0}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{k=0}^{\infty} z^k t^k \left[\sum_{\{0 \leqslant i_p \leqslant \infty\}} \frac{\prod_{p=0}^{k} \Gamma(i_p + 1)}{\Gamma\left(\sum_{p=0}^{k} (i_p + 1) \right)} \frac{(tH_0)^{i_k}}{(i_k)!} H_1 \frac{(tH_0)^{i_{k-1}}}{(i_{k-1})!} \cdots H_1 \frac{(tH_0)^{i_0}}{(i_0)!} \right] \cdot p_0$$

Now we use the Multinomial–Dirichlet normalization integral

$$\frac{\prod_{p=0}^{n} \Gamma(i_p + 1)}{\Gamma\left(\sum_{p=0}^{n} (i_p + 1) \right)} = \int_0^1 d\theta_0 \cdots \int_0^1 d\theta_k \delta \left(\sum_{p=0}^{k} \theta_p - 1 \right) \prod_{p=0}^{k} (\theta_p)^{i_p}.$$

[EM+GY Annals of Math. and A. I., 47(3-4), January 2007]

Proof Outline:

TOPE, line 3 => 4, 1/2

EMyaLkress
8/14/23

MyaWkress 2007 Appendix I
Line (3) => (4)

$$(H_0 + zH_1)^\ell = \prod_{m=1}^{\ell} (H_0 + zH_1) \qquad (H_0, H_1 \text{ don't commute})$$

$$= \prod_{m=1}^{\ell} \left(\sum_{J_m=0}^{1} H_0^{1-J_m} (zH_1)^{J_m} \right)$$

$J_m : m \in \{1 \dots \ell\} \longrightarrow \{0,1\}$ — otherwise unconstrained

$$= \sum_{\{J_m \in \{0,1\}\}} \prod_{m=1}^{\ell} \left(H_0^{1-J_m} (zH_1)^{J_m} \right) \qquad (3.3)$$

Find the k^{th} power of z in this expression:

$$\underbrace{\text{Coef}}_{\text{linear in 1st argument}}\left((H_0 + zH_1)^\ell, z^k \right) = \frac{1}{k!} \partial_z^k \left[(H_0 + zH_1)^\ell, z^k \right]_{z=0}$$

$$\dots = \text{Coef}\left(\sum_{\{J_m \in \{0,1\}\}} \prod_{m=1}^{\ell} \left(H_0^{1-J_m} (zH_1)^{J_m} \right), z^k \right)$$

(J_m counts powers $(0,1)$ of z)

$$= \sum_{\left\{ J_m \in \{0,1\} \;\middle|\; \sum_{m=1}^{\ell} J_m = k \right\}} \prod_{m=1}^{\ell} \left(H_0^{1-j_m} H_1^{j_m} \right)$$

// Note: line (3) $\frac{z^k}{n!}$ should be $\frac{z^k}{k!}$

Now introduce run length encoding of $J_m \mid \sum_{m=1}^{\ell} J_m = k$:

Change variables $\{J_m\}_{m=1}^{\ell} \longrightarrow \{i_p\}_{p=0}^{k}$

eg $H_0 H_0 H_0 H_1 H_0 H_0 H_1 H_0 H_0 H_0$
$= H_0^3 H_1 H_0^2 H_1 H_0^3$ $\quad (\ell = 10, k = 2)$

$(J_0 = 0\ J_1 = 0\ J_2 = 0\ J_3 = 1\ J_4 = 0\ J_5 = 0\ J_6 = 1\ J_7 = 0 \dots)$
$\mapsto (i_0 = 3\ i_1 = 2\ i_2 = 3)$ $\quad (J_8 = J_9 = J_{10} = 0)$

power of H_0 between p^{th} & $(p+1)^{th}$ H_1 factors;
or before 1st ($p=0$),
or after k^{th} ($p=k$).

Human Justification for 3.3, ~ the Distributive Law for Big Operators

Emy Lemos
8/23/23

Big operator distributive law

eg. $\prod$ over Σ (could be $\wedge$ over $\cup$, $\otimes$ over $\oplus$, etc)

$\prod_{i=1}^{I} A_i = \prod_{i \in \{1<2<\cdots<I\}} A_i$; $A_i \cdot A_j$ is associative but not commutative, $A_i \neq A_j$ in both cases $\neq$ comm.

base case could be: $i \in \emptyset$:

$\prod_{i=1}^{0} A_i = \prod_{i \in [\,]} = 1$; but we will use $\prod_{i=1}^{1} A_i = \prod_{i \in [\,]} = A_1$;

$\prod_{}^{I} \sum_{}^{J} = \sum_{\text{f functions}} \prod$ by induction on I:

Base case: (optional subscript)

$\prod_{i=1}^{J_1} \left(\sum_{j=1}^{J_i} A_{ij} \right) = \sum_{j=1}^{J_1} A_{1j} = \sum_{J_1=1}^{J_1} A_{1J_1} = \sum_{J_1=1}^{J_1} \left(\prod_{i=1}^{1} A_{iJ_i} \right)$ ✓

(where $J_i \geq 1$) $\quad \{J_k \mid 1 \leq k \leq I=1\}$

Induction step: $I > 1 \Rightarrow I-1 \geq 1$ (where $J_i \geq 1$)

$\prod_{i=1}^{I} \left(\sum_{j=1}^{J_i} A_{ij} \right) = \left[\prod_{i=1}^{I-1} \left(\sum_{j=1}^{J_i} A_{ij} \right) \right] \left(\sum_{J_I=1}^{J_I} A_{iJ_I} \right)$

$= \sum_{j_I=1}^{J_I} \left[\prod_{i=1}^{I-1} \left(\sum_{j=1}^{J_i} A_{ij} \right) \right] A_{iJ_I}$ // distributive law

$= \sum_{j_I=1}^{J_I} \left[\sum_{\{j_1 \ldots j_{I-1}\} \mid j_k \in \{1 \ldots J_k\}} \left(\prod_{i=1}^{I-1} A_{ij_i} \right) \right] A_{iJ_I}$ // induction

$= \sum_{j_I=1}^{J_I} \sum_{\{j_1 \ldots j_{I-1}\} \mid j_k \in [1 \ldots J_k]} \prod_{i=1}^{I} A_{ij_I}$ // distr. law + def of $\prod$

$= \sum_{\{j_1 \ldots j_{I-1}\} \mid j_k \in [1 \ldots J_k]} \sum_{j_I=1}^{J_I} \prod_{i=1}^{I} A_{ij_I}$ // commutative +, $\sum$

$\prod_{i=1}^{I} \left(\sum_{j=1}^{J_i} A_{ij} \right) = \sum_{\{j_1 \ldots j_I\} \mid j_k \in [1 \ldots J_k]} \prod_{i=1}^{I} A_{ij_I}$ // def of $\sum$

QED.

Machine justification for 3.3?

- In Coq: [Bertot et al, "Canonical Big Operators", TPHOLs 2008]

```
Lemma bigA_distr_bigA :
  forall (I J : finType) F,
  \big[*%M/1]_(i : I) \big[+%M/0]_(j : J) F i j =
  \big[+%M/0]_(f : {ffun I -> J}) \big[*%M/1]_(i) F i (f i).
```

- In Lean 4: Mathlib.Algebra.BigOperators.Ring (?)

theorem Finset.prod_sum {α : Type u} {β : Type v} [inst : CommSemiring β] {δ : α → Type u_1} [inst : DecidableEq α] [inst : (a : α) → DecidableEq (δ a)] {s : Finset α} {t : (a : α) → Finset (δ a)} {f : (a : α) → δ a → β} : (Finset.prod s fun a => Finset.sum (t a) fun b => f a b) = Finset.sum (Finset.pi s t) fun p => Finset.prod (Finset.attach s) fun x => f (↑x) (p ↑x (_ : ↑x ∈ s))

The product over a sum can be written as a sum over the product of sets, Finset.Pi.

https://leanprover-community.github.io/mathlib4_docs/Mathlib/Algebra/BigOperators/Ring.html#Finset.prod_sum

Proof Outline:

TOPE,

line 3 => 4,

2/2

Expression for map $\{j\}_0^l \to \{i\}_0^k$ by induction on p:

$p=0$ base case: ("such that")

$$i_0 \in \{0,\dots k\} \dots\dots\; j_1, \dots j_{i_0} = 0 \wedge (j_{i_0+1} = 1 \;\vee\; i_0 = k)$$

$p \geq 1$ induction, up to $p = k-1$:

$$\begin{cases} \sum_{q=0}^{p} (i_q + 1) \leq l \quad // \text{ total length of } \underbrace{H_0^{i_0} H_1}_{i_0+1} \underbrace{H_0^{i_1} H_1}_{i_1+1} \cdots \underbrace{H_0^{i_p} H_1}_{i_p+1} \\ \wedge\; j_{(\sum_{q=0}^{p-1}(i_q+1)+1)}, \dots j_{(\sum_{q=0}^{p}(i_p+1)-1)} = 0 \\ \wedge\; j_{(\sum_{q=0}^{p}(i_q+1))} = 1 \end{cases}$$

$p = k$:

$$\sum_{q=0}^{k-1} (i_q + 1) + i_k = l$$

$$\wedge\; j_{(\sum_{q=0}^{k-1}(i_q+1)+1)}, \dots \underbrace{j_{(\sum_{q=0}^{k}(i_q+1)-1)}}_{\equiv \sum_{q=0}^{k-1}(i_q+1) + i_k = l\,!} = 0$$

Then...

Lemma: $\forall k,\ \{j\}_1^l \to \{i\}_0^k$ is a bijection of discrete finite mappings.

[Verify this – obvious to a human]

Time-Ordered Product Expansion (TOPE) verification? 2/2

- "Big operator" derivation
- formal power series, changes of variable, induction on dimension
- notation "represents" permutation symmetries

Accordingly,

$$S(z) = \sum_{k=0}^{\infty} z^k \; t^k \left[\sum_{\{0 \leqslant i_p \leqslant \infty\}} \int_0^1 d\theta_0 \cdots \int_0^1 d\theta_k \delta\left(\sum_{p=\mathbf{0}}^{k} \theta_p - 1\right) \left(\prod_{p=0}^{k} (\theta_p)^{i_p}\right) \times \frac{(tH_0)^{i_k}}{(i_k)!} H_1 \frac{(tH_0)^{i_{k-1}}}{(i_{k-1})!} \cdots H_1 \frac{(tH_0)^{i_0}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{k=0}^{\infty} z^k \; t^k \left[\sum_{\{0 \leqslant i_p \leqslant \infty\}} \int_0^1 d\theta_0 \cdots \int_0^1 d\theta_k \delta\left(\sum_{p=\mathbf{0}}^{k} \theta_p - 1\right) \times \frac{(\theta_k tH_0)^{i_k}}{(i_k)!} H_1 \frac{(\theta_{k-1} tH_0)^{i_{k-1}}}{(i_{k-1})!} \cdots H_1 \frac{(\theta_0 tH_0)^{i_0}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{k=0}^{\infty} z^k \; t^k \left[\int_0^1 d\theta_0 \cdots \int_0^1 d\theta_k \delta\left(\sum_{p=\mathbf{0}}^{k} \theta_p - 1\right) \sum_{\{0 \leqslant i_0 \leqslant \infty\}} \frac{(\theta_k tH_0)^{i_0}}{(i_k)!} H_1 \times \sum_{\{0 \leqslant i_1 \leqslant \infty\}} \frac{(\theta_{k-1} tH_0)^{i_1}}{(i_{k-1})!} \cdots H_1 \sum_{\{0 \leqslant i_k \leqslant \infty\}} \frac{(\theta_0 tH_0)^{i_k}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{k=0}^{\infty} z^k \; t^k \left[\int_0^1 d\theta_0 \cdots \int_0^1 d\theta_k \delta\left(\sum_{p=\mathbf{0}}^{k} \theta_p - 1\right) \exp(\theta_k tH_0) \times H_1 \exp(\theta_{k-1} tH_0) \cdots H_1 \exp(\theta_0 tH_0) \right] \cdot p_0$$

Thus

$$S(z) = \sum_{k=0}^{\infty} z^k \left[\int_0^t d\tau_0 \cdots \int_0^t d\tau_k \delta\left(\sum_{p=\mathbf{0}}^{k} \tau_p - t\right) \exp(\tau_k H_0) \times H_1 \exp(\tau_{k-1} H_0) \cdots H_1 \exp(\tau_0 H_0) \right] \cdot p_0 \qquad \text{(A59)}$$

In summary (since p_0 was never used in the above calculations),

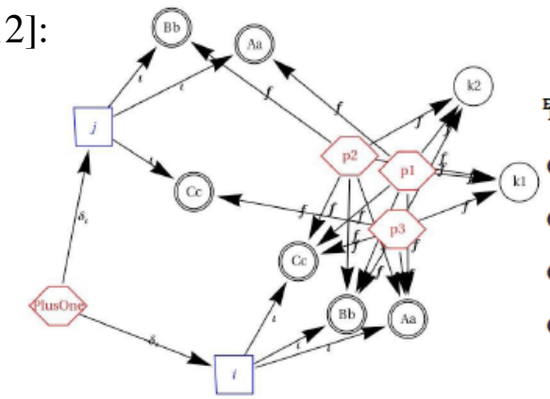
$$\exp(t\,(H_1 + H_0)) = \sum_{k=0}^{\infty} \left[\int_0^t d\tau_0 \cdots \int_0^t d\tau_k \delta\left(\sum_{p=\mathbf{0}}^{k} \tau_p - t\right) \exp(\tau_k H_0) H_1 \exp(\tau_{k-1} H_0) \cdots H_1 \exp(\tau_0 H_0) \right].$$

22

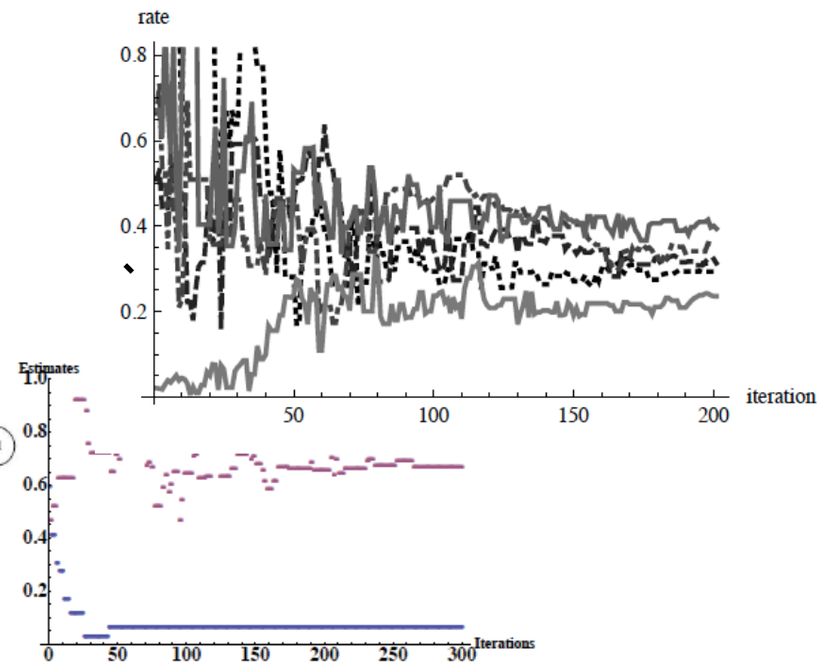
[EM+GY Annals of Math. and A. I., 47(3-4), January 2007]

Learning in stochastic reaction networks

- MCMC [Yosiphon thesis 2009]:
- DD for SDE version [Johnson thesis 2012]:



(a) Model diagram.



(b) Sampled parameter values as a function of iteration number. Annealing causes samples to converge to point estimates near the true values.

- TOPE method
  - [Wang et al., BMC Systems Biology 2010]:

$$\tilde{\rho}_r \frac{\partial}{\partial \tilde{\rho}_r}\left[e^{(t_{s+1}-t_s)\tilde{W}}\right](x(t_{s+1}),x(t_s)) = \sum_{k=0}^{\infty}\int_{t_s}^{t_{s+1}}\cdots\int_{t_s}^{t_{s+1}} d[\tau]_0^n\,\delta\left((t_{s+1}-t_s)-\sum_{p=0}^{n}\tau_p\right)$$

$$\times\left[\sum_{p=1}^{n}\left(e^{-\tau_n\tilde{D}}\,\tilde{W}\ldots e^{-\tau_p\tilde{D}}\left(\tilde{b}_r\,\tilde{W}\right)e^{-\tau_{p-1}\tilde{D}}\ldots e^{-\tau_1\tilde{D}}\,\tilde{W}\,e^{-\tau_0\tilde{D}}\right)(x(t_{s+1}),x(t_s))\right.$$

$$-\tilde{\rho}_r\sum_{p=1}^{n}\left(e^{-\tau_n\tilde{D}}\,\tilde{W}\ldots e^{-\tau_p\tilde{D}}\,\tau_p\,\tilde{D}_r\ldots e^{-\tau_1\tilde{D}}\,\tilde{W}\,e^{-\tau_0\tilde{D}}\right)$$

$$\left.-\tilde{\rho}_r\,e^{-\tau_n\tilde{D}}\,\tilde{W}\ldots e^{-\tau_1\tilde{D}}\,\tilde{W}\,e^{-\tau_0\tilde{D}}\,\tau_0\,\tilde{D}_r\right]$$

$$=\sum_{k=0}^{\infty}\sum_{p=1}^{n}\left\langle\tilde{b}_r(\text{reaction event } p \text{ out of } n)\right\rangle_{\tilde{W},x(t_{s+1}),x(t_s)}$$

$$-\tilde{\rho}_r\sum_{k=0}^{\infty}\sum_{p=0}^{n}\left\langle\tau_p\,\tilde{D}_r(\text{reaction event } p \text{ out of } n)\right\rangle_{\tilde{W},x(t_{s+1}),x(t_s)}$$

Stochastic Simulation

- SSA: $$\mathcal{W}(I, t' \mid J, t) \approx \hat{W}_{I,J} \exp(-(t' - t) D_{J\,J})\, \mathbf{1}(t' \geqslant t)$$

$$\Pr(. \mid J, k) = \mathcal{W}^k \circ \Pr(. \mid J, 0) = \left[\hat{W} \exp(-\Delta t\, D)\, \mathbf{1}(\Delta t \geqslant 0)\right]^k \circ \Pr(. \mid J, 0)$$

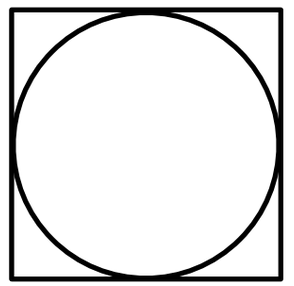
- Exact R-leap:

$$\left[\prod_{k=L-1\searrow 0} \hat{W} \exp(-\tau_k D)\right]_{I_L, I_0} = \frac{(\tilde{D}_{I_0\,L-1})^L}{(\tilde{D}_{I_0\,L-1})^L} \sum_{\{s \mid s_r \in \mathbb{N},\, \sum_r s_r = L\}} \text{Multinomial}(s \mid \boldsymbol{p}, L)$$

$$\times \text{Erlang}\left(\sum_k \tau_k \,\middle|\, L, \tilde{D}_{I_0\,L-1}\right) \text{UniformSimplex}(\boldsymbol{\tau}; L)\, \text{Accept}(s, L, \boldsymbol{\tau})$$

[with Orendorff, Chatelain, Koumoutsakos - J. Chem. Phys. 2009]

Rejection Sampling
(in general)



Rejection sampling allows one to exploit probability bounds in exact sampling, as follows: given a target distribution $P(x)$ and an algorithm for sampling from a related distribution $P'(x)$ and from the uniform distribution $U(u)$ on [0,1], and if

$$P(x) < M\,P'(x)$$

for some constant $M > 1$, then $P(x)$ satisfies

$$P(x) = P'(x)\,\frac{P(x)}{M\,P'(x)} + (1 - 1/M)\,P(x)$$

and therefore also

$$P(x) = \int P'(x')\,dx' \int U(u)\,du \left[\mathbf{1}\left(u < \frac{P(x')}{M\,P'(x')}\right)\cdot\delta(x - x') + \mathbf{1}\left(u \geqslant \frac{P(x')}{M\,P'(x')}\right)\cdot P(x)\right]$$

which constitutes a mixture distribution, that can be applied recursively as needed to sample from $P(x)$. Pseudocode for sampling $P(x)$ according to Equation 13 is as follows (where "//" introduces a comment):

```
while not accepted {
        sample P'(x) and U(u);    // P'(x) only approximates P(x)
        compute Accept(x) = P(x) / (M P'(x));    // acceptance probability
        if  u < Accept(x) then accept x;
}  // now P(x) is sampled exactly
```

Accelerated Rejection Sampling

~ *"Squeeze method" [Marsaglia 1961]*

But what if $P(x)$ is expensive to compute? Then Accept(x) will also be expensive to compute and rejection sampling may be prohibitively expensive, even for a good approximating $P'(x)$. A solution to this problem is possible if a cheap lower bound for $P(x)$ is available. Suppose there is a function $\underline{A}(x)$ such that

$$0 \leqslant \underline{A}(x) \leqslant \text{Accept}(x) \equiv P(x) / (M\, P'(x)) < 1$$

- *Then* Accept(x) is a mixture:

$$\text{Accept}(x) = \underline{A}(x) \cdot 1 + (1 - \underline{A}(x)) \cdot Q(x), \text{ where}$$

$$Q(x) \equiv \left(\frac{\text{Accept}(x) - \underline{A}(x)}{1 - \underline{A}(x)} \right),$$

- and we have the generic pseudocode algorithm:

```
while not accepted {
        sample P'(x) and U(u);   // cheap but approximate
        compute A(x);   // cheap
        if u < A(x) then accept x;
        else {
                compute Accept(x) = P(x) / M P'(x);   // expensive
                compute Q(x) = (Accept(x) – A(x)) / (1 – A(x));   // A(x) < 1 ⇒ 1 – A(x) ≠ 0
                sample U(u);
                if u < Q(x) then accept x;
                else reject x;
        }
}
```

AITP for Science:

some logic in favor

- “AI”: Reliable AI for Science:
  - Representationalism 1st, ML 2nd. (On top vs. on tap)
  - *formal, symbolic* DSLs for modeling …
    - applicable math, sci content, algorithms (model sim, ML, analysis)
- “TP”: verify exact and approximate maps between such languages
  - e.g. dynamics to simulation algs: XDEs, Markov jump processes, hybrids e.g. *dynamical grammars*
  - high-level support for scientific knowledge representation

Variable-binding via operator integration

- Parameterized grammar rules: $\{\tau_i(x_i)\} \rightarrow \{\tau'_j(y_j)\}$ **with** $\rho_r((x_i), (y_j))$
- Parameterized grammar rule operators:

$$\hat{O}_r = \int_{D_{\beta(1)}} \cdots \int_{D_{\beta(c)}} \cdots \left(\prod_c d\mu_{\beta(c)}(X_c) \right) \rho_r \left([x_i(\{X_c\})], [y_j(\{X_c\})] \right)$$
$$\times \left[\prod_{i \in \mathrm{rhs}(r)} \hat{a}_{a(i)}(x_i(\{X_c\})) \right] \left[\prod_{j \in \mathrm{lhs}(r)} a_{b(j)}(y_j(\{X_c\})) \right] \qquad (19)$$

Thus, syntactic variable-binding has the semantics of multiple integration. This is the same result one would get if each rule with variables were replaced with a (finite, countable, or uncountably infinite) set of rules with all possible values substituted in for all the variables, with firing rates weighted by the relevant measure, and running in parallel.

[Annals of Math. and A. I., 47(3-4), January 2007]

- So, object parameters need *measure spaces*

Ψ

Particle to Structure Dynamics

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

$$A_1(x_1),\ A_2(x_2),\ \ \ldots,\ A_n(x_n) \rightarrow B_1(y_1),\ B_2(y_2),\ \ \ldots,\ B_m(y_m)\ \textbf{with}\ \rho(\{x_i\},\ \{y_j\})$$

$$\tilde{O}_r = \rho_r \sum_{\{x'_i, x_j\}} \prod_{i \in \mathrm{rhs}(r)} \hat{a}(\tau_i, x_i) \prod_{j \in \mathrm{lhs}(r)} a(\tau_j, x_j)\, \Pr(\{x_i\} \mid \{x_j\})$$

(and can integrate ODE rules too)

$$[a_{ab}, \hat{a}_{cd}] = I\, \delta_{(ab),(cd)}$$
$$[a_i, a_j] = [\hat{a}_i, \hat{a}_j] = 0$$

Ψ

Particle to Structure Dynamics

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

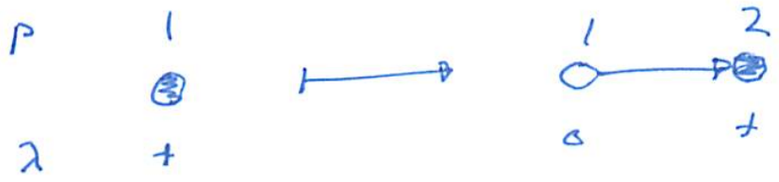
$$A_1(x_1),\ A_2(x_2),\ \ ...,\ A_n(x_n) \to B_1(y_1),\ B_2(y_2),\ \ ...,\ B_m(y_m)\ \textbf{with}\ \rho(\{x_i\},\ \{y_j\})$$

$$\tilde{O}_r = \rho_r \sum_{\{x'_i, x_j\}} \prod_{i \in \mathrm{rhs}(r)} \hat{a}(\tau_i, x_i) \prod_{j \in \mathrm{lhs}(r)} a(\tau_j, x_j)\, \Pr(\{x_i\} \mid \{x_j\})$$

(and can integrate ODE rules too)

$$[a_{ab}, \hat{a}_{cd}] = I\, \delta_{(ab),(cd)}$$
$$[a_a, a_b] = [\hat{a}_a, \hat{a}_b] = 0$$

- ***Labelled graph*** (structure) transitions



$$\hat{W}_r \propto \int d\lambda d\lambda'\, \rho_r(\lambda, \lambda') \sum_{\langle i_1, \ldots i_k \rangle_{\neq}} \hat{a}_{i_1,\ldots i_k}(G^{r\ \mathrm{out}}) a_{i_1,\ldots i_k}(G^{r\ \mathrm{in}})$$

(and can integrate ODE rules too)

[EM, MFPS Proc., ENTCS 2010]

$$\begin{aligned}
\hat{a}_\alpha^2 &= 0 = a_\alpha^2 \\
a_\alpha \hat{a}_\beta &= (1 - \delta_{\alpha\beta}) \hat{a}_\beta a_\alpha + \delta_{\alpha\beta} Z_\alpha \\
Z_\alpha &\equiv I_\alpha - N_\alpha \\
N_\alpha &\equiv \hat{a}_\alpha a_\alpha
\end{aligned}$$

$$\begin{aligned}
\hat{a}_{i_1,\ldots i_k}(G') &= \hat{a}_{i_1,\ldots i_k}(G'_{\mathrm{links}}) \hat{a}_{i_1,\ldots i_k}(G'_{\mathrm{nodes}}) \\
&= \left[\prod_{s',t' \in \mathrm{rhs}(r)} \left(\hat{a}_{i_{s'} i_{t'}} \right)^{g'_{s't'}} \right] \left[\prod_{v' \in \mathrm{rhs}(r)} \hat{a}_{i_{v'} \lambda'_{v'}} \right] \\
a_{i_1,\ldots i_k}(G) &= a_{i_1,\ldots i_k}(G_{\mathrm{links}}) a_{i_1,\ldots i_k}(G_{\mathrm{nodes}}) \\
&= \left[\prod_{s,t \in \mathrm{lhs}(r)} (a_{i_s} i_t)^{g_{st}} \right] \left[\prod_{v \in \mathrm{lhs}(r)} a_{i_v \lambda_v} \right].
\end{aligned}$$

Ψ

Algebra of Labelled-Graph Rewrite Rules

$$\hat{W}_{G^{r2\ \mathrm{in}} \to G^{r2\ \mathrm{out}}} \hat{W}_{G^{r1\ \mathrm{in}} \to G^{r1\ \mathrm{out}}} \simeq \sum_{\substack{H \subseteq G^{r1\ \mathrm{out}} \simeq \tilde{H} \subseteq G^{r2\ \mathrm{in}} \\ | \text{ edge-maximal}}} \sum_{h: H \overset{1\text{-}1}{\hookrightarrow} \tilde{H}} \hat{W}_{G^{r1\ \mathrm{in}} \cup (G^{r2\ \mathrm{in}} \setminus \tilde{H}) \underset{h}{\to} G^{r2\ \mathrm{out}} \cup (G^{r1\ \mathrm{out}} \setminus H)}$$

$$G^{1;2\ \mathrm{in}}_{\mathrm{nodes}} = G^{r_1\ \mathrm{in}}_{\mathrm{nodes}} \dot{\cup} (G^{r_2\ \mathrm{in}}_{\mathrm{nodes}} \setminus \tilde{H}_{\mathrm{nodes}}) \qquad G^{1;2\ \mathrm{out}}_{\mathrm{nodes}} = G^{r_2\ \mathrm{out}}_{\mathrm{nodes}} \dot{\cup} (G^{r_1\ \mathrm{out}}_{\mathrm{nodes}} \setminus H_{\mathrm{nodes}})$$

$$G^{1;2\ \mathrm{in}}_{\mathrm{links}} = G^{r_1\ \mathrm{in}}_{\mathrm{links}} \cup h^{-1\star}(G^{r_2\ \mathrm{in}}_{\mathrm{links}} \setminus \tilde{H}_{\mathrm{links}}) \qquad G^{1;2\ \mathrm{out}}_{\mathrm{links}} = G^{r_2\ \mathrm{out}}_{\mathrm{links}} \cup h^{\star}(G^{r_1\ \mathrm{out}}_{\mathrm{links}} \setminus H_{\mathrm{links}})$$

$$K_a = G^{r_a\ \mathrm{in}}_{\mathrm{nodes}} \cap G^{r_a\ \mathrm{out}}_{\mathrm{nodes}}$$

$$K_{1;2} = (K_1 \setminus H_{\mathrm{nodes}} \ \cup \ h^{-1}(K_2 \setminus \tilde{H}_{\mathrm{nodes}}) \ \cup \ (K_1 \cap h^{-1\star}(K_2))$$

[EM, http://arxiv.org/abs/1909.04118]

Verifying this result by machine would be both valuable, and a big challenge.
It was the longest, trickiest calculation I've ever done by hand.
Possibly we need an easier hand-verification first.

Point of maximum intermediate expression swell

$$
\begin{aligned}
\hat{W}_{r_2}\hat{W}_{r_1} \;=\; & \frac{1}{C_{r_1}(N_{\max\,\text{free}})}\frac{1}{C_{r_2}(N_{\max\,\text{free}})}\rho_{r_1}(\boldsymbol{\lambda}^{(1)},\boldsymbol{\lambda}'^{(1)})\rho_{r_2}(\boldsymbol{\lambda}^{(2)},\boldsymbol{\lambda}'^{(2)}) \\
& \times \sum_{T\subseteq G^{r_1\,\text{in}}_{\text{nodes}}\setminus G^{r_1\,\text{out}}_{\text{nodes}}} \;\; \sum_{\pi:T\overset{1\text{-}1}{\hookrightarrow}G^{r_2\,\text{out}}_{\text{nodes}}\setminus G^{r_2\,\text{in}}_{\text{nodes}}} \\
& \times \sum_{S\subseteq G^{r_1\,\text{out}}_{\text{nodes}}} \;\; \sum_{h:S\overset{1\text{-}1}{\hookrightarrow}G^{r_2\,\text{in}}_{\text{nodes}}} \left[\prod_{i_{5\,0}\in L_2\cap R_1=\mathcal{I}(S)} \delta_{\lambda'^{(1)}_{\mathcal{I}^{-1}(i_{5\,0})},\,\lambda^{(2)}_{h(\mathcal{I}^{-1}(i_{5\,0}))}}\right] \quad \text{// defines } \sum_{H} \text{ and adjusts } \tfrac{1}{C} \\
& \sum_{\mathcal{I}:L_1\cup R_1\overset{1\text{-}1}{\hookrightarrow}\mathcal{U}} \;\; \sum_{\substack{\mathcal{J}:L_2\cup R_2\overset{1\text{-}1}{\hookrightarrow}\mathcal{U} \\ \text{Im}(\mathcal{I})\cap\text{Im}(\mathcal{J})=\mathcal{I}(S)\cup\mathcal{I}(T) \\ \mathcal{I}(S)=\mathcal{J}(h(S))=L_2\cap R_1\;\wedge\;\mathcal{I}(T)=\mathcal{J}(\pi(T))}} \left\{\left[\prod_{(j_1,j_2)\in\mathcal{R}_2}\hat{a}_{(j_1,j_2)}\right]\left[\prod_{(i_1,i_2)\in\mathcal{R}_1\setminus\mathcal{L}_2}\hat{a}_{i_1 i_2}\right]\right\}_{③} \\
& \times \left\{\left[\prod_{j_5\in R_2\setminus(R_1\cap\overline{L_2\cap R_1})=R_2\setminus(R_1\setminus L_2)}\hat{a}_{j_5,\,\lambda'^{(2)}_{\mathcal{J}^{-1}(j_5)}}\right]\left[\prod_{i_{5\,1}\in R_1\setminus L_2=\mathcal{I}(G^{r_1\,\text{out}}_{\text{nodes}}\setminus S)}\hat{a}_{i_{5\,1},\,\lambda'^{(1)}_{\mathcal{I}^{-1}(i_{5\,1})}}\right]\right\}_{④} \\
& \times \left\{\left[\prod_{(j_3,j_4)\in\mathcal{L}_2\setminus\mathcal{R}_1} a_{j_3 j_4}\right]\left[\prod_{(i_3,i_4)\in\mathcal{L}_1} a_{i_3 i_4}\right]\right\}_{⑤} \\
& \times \left\{\left[\prod_{j_{6\,1}\in L_2\setminus R_1=\mathcal{J}(G^{r_2\,\text{in}}_{\text{nodes}}\setminus h^{-1}(S))} a_{j_{6\,1},\,\lambda^{(2)}_{\mathcal{J}^{-1}(j_{6\,1})}}\right]\left[\prod_{i_6\in L_1} a_{i_6,\,\lambda^{(1)}_{\mathcal{I}^{-1}(i_6)}}\right]\right\}_{⑥} \\
& \times \left\{\left[\prod_{(j_7,j_8)\in\mathcal{I}(H_{\text{links}})\setminus\mathcal{L}_1\equiv(\mathcal{L}_2\cap\mathcal{R}_1)\setminus\mathcal{L}_1} Z_{j_7 j_8}\right]\left[\prod_{i_{5\,0\,0}\in\mathcal{I}(H_{\text{nodes}})\setminus L_1=(L_2\cap R_1)\setminus L_1} Y_{i_{5\,0\,0},\,\lambda'^{(1)}_{\mathcal{I}^{-1}(i_{5\,0\,0})}}\right]\right\}_{⑦}
\end{aligned}
$$

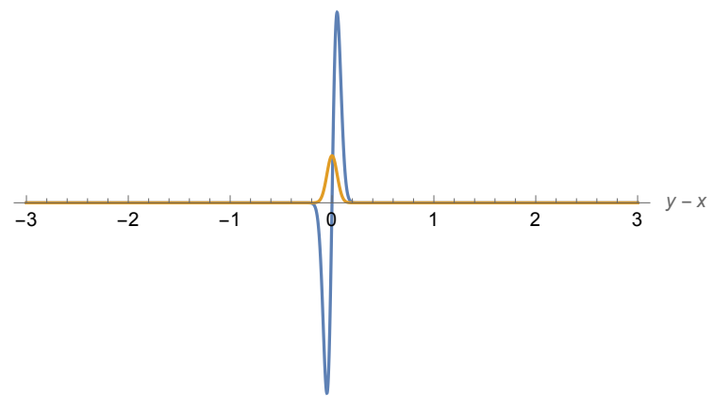
[EM, http://arxiv.org/abs/1909.04118]

Algebra of Labelled-Graph Rewrite Rules: ODE case

Operator for ODE rules:

$$W_{\textbf{ODE}\ 2} = \hat{W}_{\textbf{ODE}\ 2} = \int d\boldsymbol{x}d\boldsymbol{y}\ \rho_2(\boldsymbol{y},\boldsymbol{x}) \sum_{\langle i_1,\dots i_k\rangle_{\neq}} \hat{a}_{i_1,\dots i_k}(G^{(2)}(\boldsymbol{y}))a_{i_1,\dots i_k}(G^{(2)}(\boldsymbol{x}))\ \textbf{, where}$$

$$\rho_2(\boldsymbol{y},\boldsymbol{x}) = -\textbf{grad}_{\boldsymbol{y}} \cdot (\boldsymbol{v}(\boldsymbol{y})\delta(\boldsymbol{y}-\boldsymbol{x})) = -\sum_a \textbf{grad}_{y_a}(v_a(\boldsymbol{y})\prod_b \delta(\boldsymbol{y}-\boldsymbol{x}))\quad .$$



Commutator of ODE & ODE rules:

$$\boldsymbol{v}_{[2\,,1]}(\boldsymbol{x}) = (\boldsymbol{v}_1 \cdot \textbf{grad}_{\boldsymbol{x}})\boldsymbol{v}_2(\boldsymbol{x}) - (\boldsymbol{v}_2 \cdot \textbf{grad}_{\boldsymbol{x}})\boldsymbol{v}_1(\boldsymbol{x})$$

Commutator of GG & ODE rules:

$$[W_{\textbf{ODE}\ (2)}, \hat{W}_{\text{SPG}\ (1)}] = \frac{1}{C_1(N_{\textbf{max free}})} \sum_{h:H\to\tilde{H}} \int d\mu_{(1)h}(X_{h\perp})\ \%$$

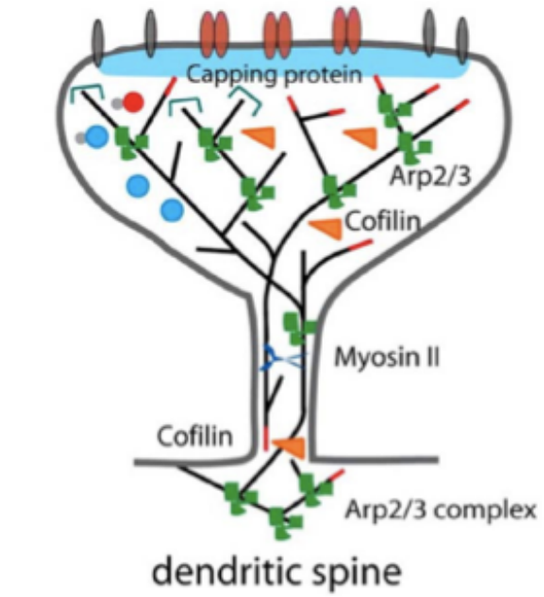
$$\int dz_{h\parallel} \int d\boldsymbol{x} \int d\boldsymbol{y} \quad \rho_2(\boldsymbol{y},\boldsymbol{x})$$

$$\times \Big\{ \rho_1(z_{h\parallel} = \boldsymbol{x}, X_{h\perp}) \hat{W}_{G^{1;2}\,\textbf{in}_{(\tilde{H},\boldsymbol{x},z_{h\perp})\to_h G^{1;2}}\,\textbf{out}_{(H,\boldsymbol{y},z_{h\perp})}}$$

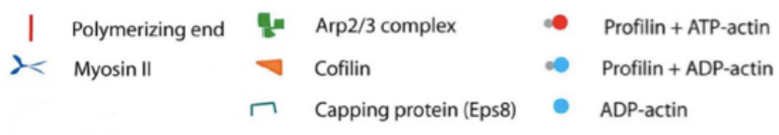
$$-\rho_1(z_{h\parallel} = \boldsymbol{y}, X_{h\perp}) \hat{W}_{G^{2;1}\,\textbf{in}_{(\tilde{H},\boldsymbol{x},z_{h\perp})\to_h G^{2;1}}\,\textbf{out}_{(H,\boldsymbol{y},z_{h\perp})}} \Big\}\quad ,$$

[EM, Front. Syst. Biol., 09 September 2022, **Proposition 1**]

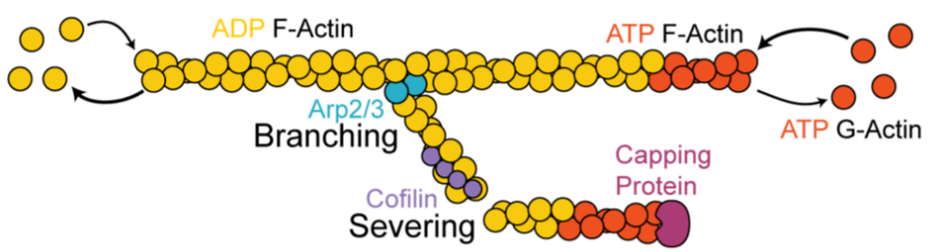
Synapse morphodynamics

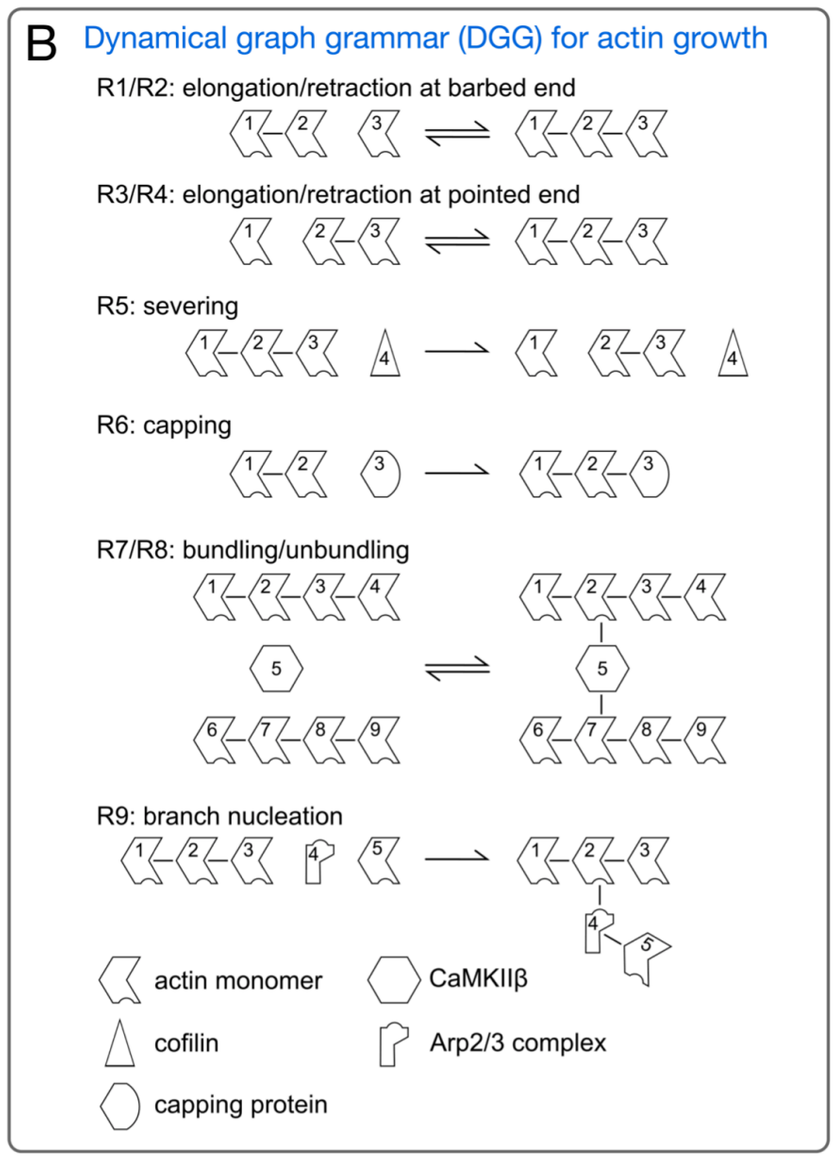


T. Bartol (Salk), modified from: [Hotulainen & Hoogenraad, J. Cell Biology 189, 2010]



(And likewise for axons.)





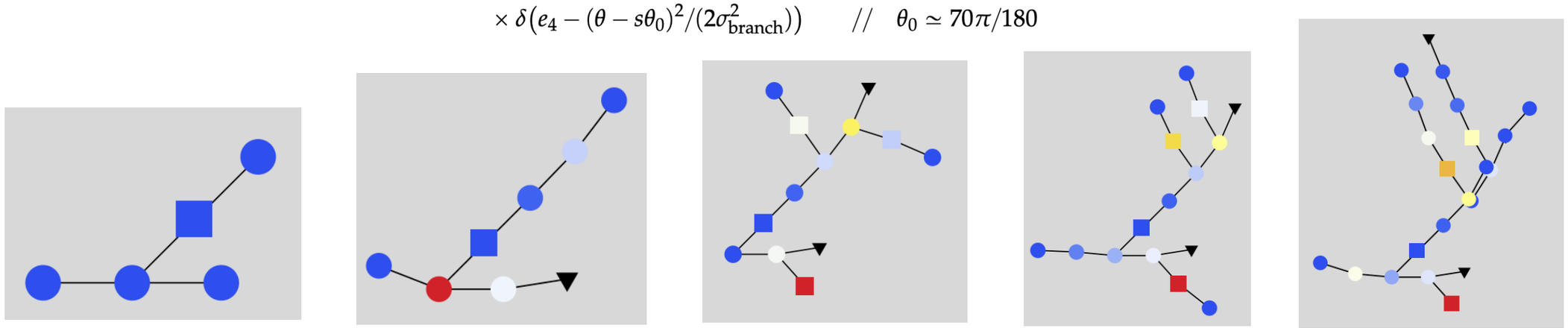
E.g. actin cytoskeleton in synapse

- Branching rule - one of a dozen or so

$$\left(\begin{array}{ccccc} \bigcirc_1 & \longrightarrow & \bigcirc_2 & \longrightarrow & \bigcirc_3 \\ & & \bullet_4 & & \end{array}\right) \langle\!\langle (\boldsymbol{x}_1, e_1), (\boldsymbol{x}_2, e_2), (\boldsymbol{x}_3, e_3) \rangle\!\rangle$$

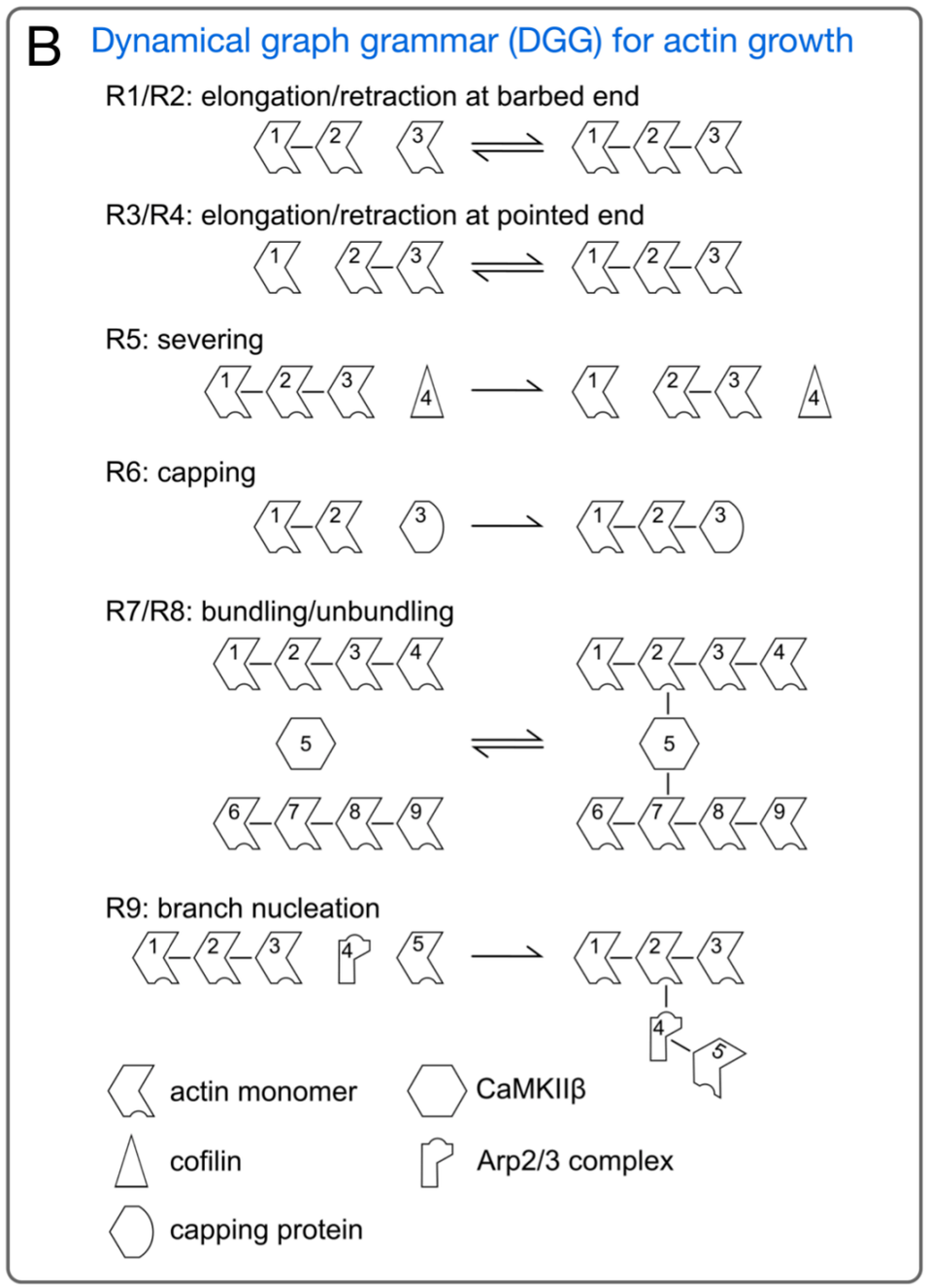
$$\longrightarrow \left(\begin{array}{ccccc} \bigcirc_1 & \longrightarrow & \bigcirc_2 & \longrightarrow & \bigcirc_3 \\ & & & \searrow & \\ & & & & \blacksquare_4 \end{array}\right) \langle\!\langle (\boldsymbol{x}_1, e_1), (\boldsymbol{x}_2, e_2), (\boldsymbol{x}_3, e_3), (\boldsymbol{x}_4, e_4) \rangle\!\rangle$$

$$\textbf{with}\ \ \rho_{\text{branch}} \mathcal{N}(\theta; s\theta_0, \sigma_{\text{branch}})\ \text{Unif}(s \in \{\pm 1\})\ \delta\Big(\boldsymbol{x}_4 - \big(\boldsymbol{x}_2 + R(\theta) \circ (\boldsymbol{x}_2 - \boldsymbol{x}_1)\big)\Big)$$
$$\times\, \delta\big(e_4 - (\theta - s\theta_0)^2 / (2\sigma^2_{\text{branch}})\big) \qquad // \quad \theta_0 \simeq 70\pi/180$$



Plenum DGG simulations: Matthew Hur
Earlier software: Guy Yosiphon and Arthur York

Dendritic spine head DGG Rules



Actin Network Biomechanics Rules

- Stochastic Angle Bending
- Pairwise Lennard-Jones Force
- Tri-Nodal Anisotropic Buckling
- Hessian Boltzmann Sampling
- Euler-Bernoulli Deflection

Spine Head Membrane Rules

- Area, Line Tension, & Helfrich Bending Energy

Biomechanical Rules

- Implement dynamics for combined energy
- $$E_{\text{tot}} = \sum_{ijk}^{N} \underbrace{U_{\text{tri-nodal}}(x_i, x_j, x_k)}_{\text{Anisotropic Buckling}} + \underbrace{U_H(x_i, x_j, x_k)}_{\text{Hessian Thermal Noise}} + \underbrace{U_{\text{LJ-midpoint}}(x_1, x_2, x_3)}_{\text{Stochastic Midpoint Minimization}} + \underbrace{P \sum_{ij}^{M} \frac{x^i y^j - y^i x^j}{2}}_{\text{Membrane Area}} + \underbrace{\Omega \sum_{ijk}^{M} \frac{||x_i - x_j||_2 + ||x_k - x_j||_2}{2}}_{\text{Membrane Length}} + \underbrace{2\kappa \sum_{hijkl}^{M} \frac{\partial H^2}{\partial x_j}(x_h, x_i, x_j, x_k, x_l)}_{\text{Membrane Helfrich Curvature}}$$
- where

$$U_H(x_1, x_2, x_3) = \frac{1}{2}(x_2 - x_1)^T \frac{\partial^2 U_{LJ}(x_2, x_1)}{\partial x_2 \partial x_1}(x_2 - x_1) + \frac{1}{2}(x_2 - x_3)^T \frac{\partial^2 U_{LJ}(x_2, x_3)}{\partial x_2 \partial x_3}(x_2 - x_3)$$

- E.g. (pretty-printed rules) :

Hessian Thermal Noise:

$$(\bigcirc_1 \text{——} \bigcirc_2 \text{——} \bigcirc_3)\,\langle\!\langle (\boldsymbol{x}_1, \boldsymbol{\theta}_1), (\boldsymbol{x}_2, \boldsymbol{\theta}_2), (\boldsymbol{x}_3, \boldsymbol{\theta}_3) \rangle\!\rangle$$

$$\longrightarrow (\bigcirc_1 \text{——} \bigcirc_2 \text{——} \bigcirc_3)\,\langle\!\langle (\boldsymbol{x}_1, \boldsymbol{\theta}_1), (\boldsymbol{x}_2 + \Delta_{\text{H}}, \boldsymbol{\theta}_2), (\boldsymbol{x}_3, \boldsymbol{\theta}_3) \rangle\!\rangle$$

with $k_{\text{biomech}} p(U_{\text{H}}(x_1, x_2 + \Delta_{\text{H}}, x_3) - U_{\text{H}}(x_1, x_2, x_3))$

where $\begin{cases} \Delta_{\text{H}} \sim \mathcal{N}_2(0, \Sigma_H) \\ p(\Delta U(x_1, x_2, x_3)) = \frac{e^{-\frac{\Delta U(x_1, x_2, x_3)}{k_B T}}}{1 + e^{-\frac{\Delta U(x_1, x_2, x_3)}{k_B T}}} \end{cases}$

LJ Anisotropic Buckling:

$$(\bigcirc_1 \text{——} \bigcirc_2 \text{——} \bigcirc_3)\,\langle\!\langle (\boldsymbol{x}_1, \boldsymbol{\theta}_1), (\boldsymbol{x}_2, \boldsymbol{\theta}_2), (\boldsymbol{x}_3, \boldsymbol{\theta}_3) \rangle\!\rangle$$

$$\longrightarrow (\bigcirc_1 \text{——} \bigcirc_2 \text{——} \bigcirc_3)\,\langle\!\langle (\boldsymbol{x}_1, \boldsymbol{\theta}_1), (\boldsymbol{x}_2 + \Delta_{\text{LJ}}(x_2), \boldsymbol{\theta}_2), (\boldsymbol{x}_3, \boldsymbol{\theta}_3) \rangle\!\rangle$$

with $k_{\text{biomech}} p(U_{\text{tri-nodal}}(x_1, x_2 + \Delta_{\text{LJ}}(x_2), x_3) - U_{\text{tri-nodal}}(x_1, x_2, x_3))$

where $\begin{cases} \Delta_{\text{LJ}}(x_1, x_2, x_3) = \text{Clip}(-\frac{1}{\zeta} \frac{\partial U_{\text{tri-nodal}}(x_1, x_2, x_3)}{\partial x_2}) \\ p(\Delta U(x_1, x_2, x_3)) = \frac{e^{-\frac{\Delta U(x_1, x_2, x_3)}{k_B T}}}{1 + e^{-\frac{\Delta U(x_1, x_2, x_3)}{k_B T}}} \end{cases}$

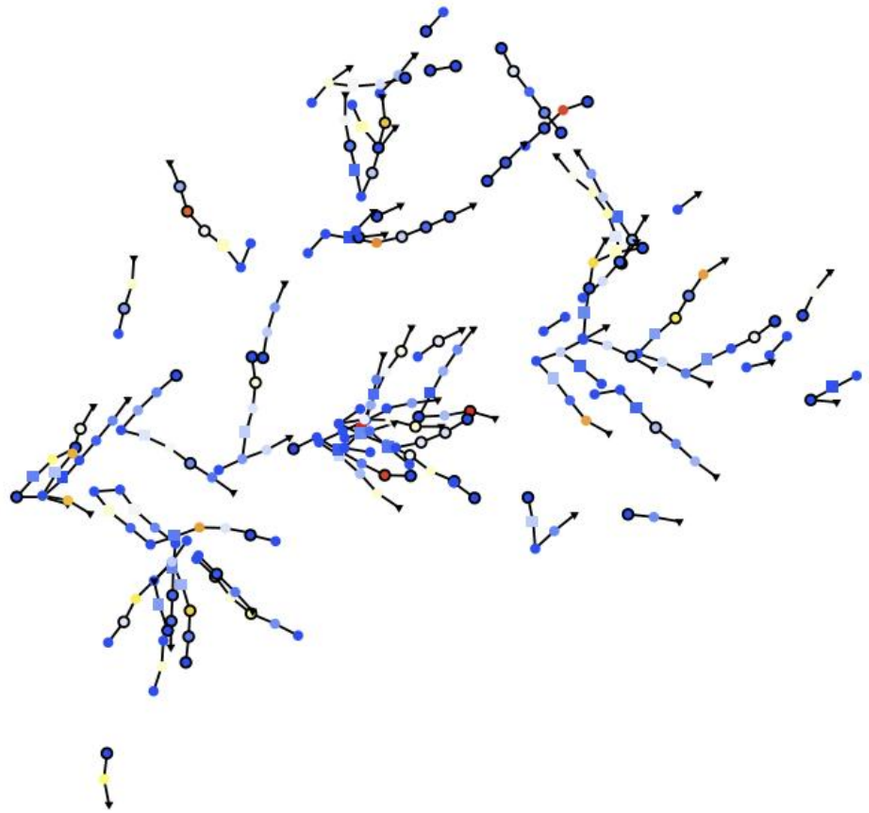
Plenum/Mathematica implementation

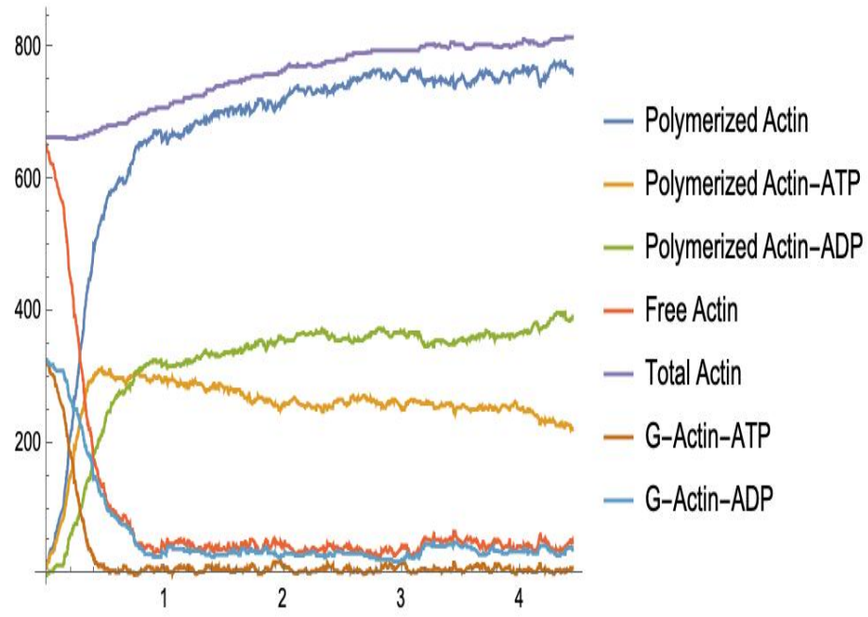
E.g.:
Rule for Hessian Thermal Noise
Boltzmann Sampling of Random Displacements for a Chain of Three Actins:

```
{
  actin[aIDP, coordsP, aIDPP, aID, angleP, CLP, arpIDP, endP, membraneIDP, nucIDP], actin[aID, coords, aIDP, aIDN, angle, CL, arpID, INT, membraneID, nucID],
  actin[aIDN, coordsN, aID, aIDNN, angleN, CLN, arpIDN, endN, membraneIDN, nucIDN]
 } → {
  (*Move position of middle actin and update angle*)
  actin[aID, addVectors[coords, {dx, dy}], aIDP, aIDN,
   middleThetaMovement[coordsP, addVectors[coords, {dx, dy}], coordsN], CL
   , arpID, INT, membraneID, nucID],
  (*Update angle of first actin*)
  actin[aIDP, coordsP, aIDPP, aID, If[endP == INT, angleP + startActinDeltaTheta[coordsP, coords, addVectors[coords, {dx, dy}]], 0], CLP, arpIDP, endP, membraneIDP, nucIDP],
  (*Update angle of last actin*)
  actin[aIDN, coordsN, aID, aIDNN, If[endN == INT, angleN + endActinDeltaTheta[coords, coordsN, addVectors[coords, {dx, dy}]], 0], CLN, arpIDN, endN, membraneIDN, nucIDN]
 },
with[networkFactor * heatBathHessian[coordsP, coords, coordsN, l0, l0, {dx, dy}, step] × grammarPDF[NormalDistribution[0, transverseDeviationA / biomechanicalRate], dx] ×
  grammarPDF[NormalDistribution[0, transverseDeviationA / biomechanicalRate], dy]],
```

2D Actin Network

Network Remodeling Rules Reach Non-Zero Steady-State

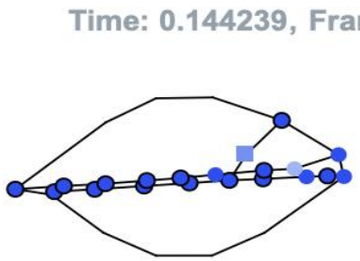


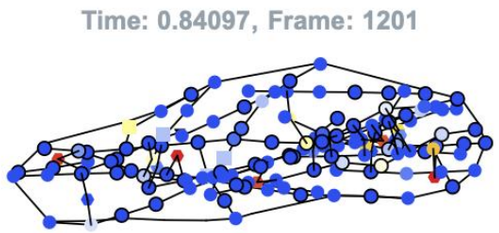


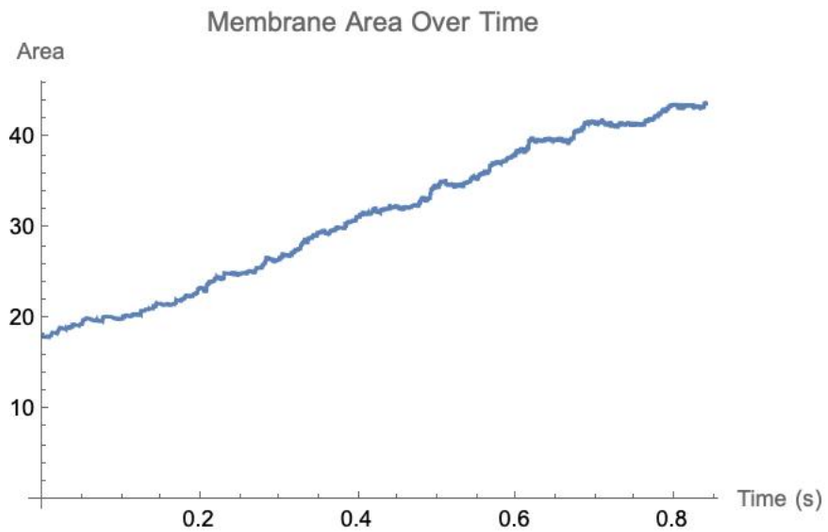
- 0.06 fraction of actin are monomers at steady-state. Addition of membrane would increase this value making it consistent with 0.12 reported in literature
- Average number of actins per branch is 7 in the simulation while it is 8 in the literature

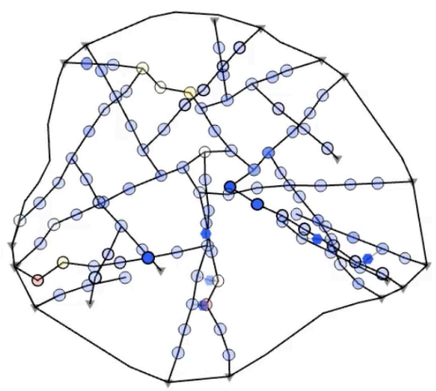
Plenum DGG simulations: Matthew Hur
Earlier software: Guy Yosiphon and Arthur York

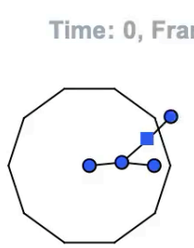
2D Actin Network with Membrane

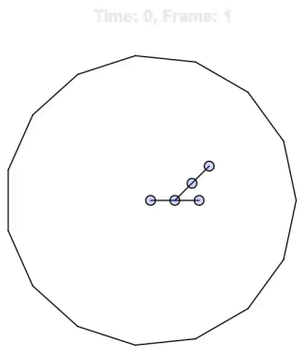








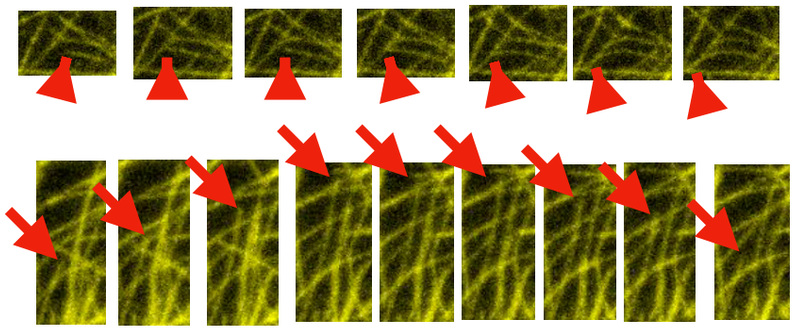




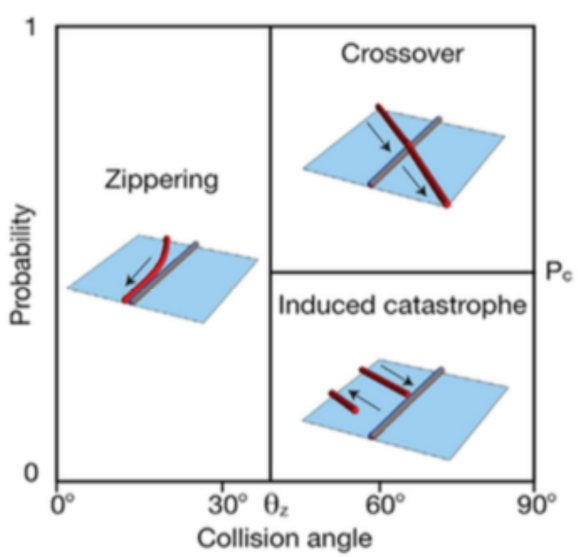
slower biomech rate
higher curvature penalty

Plenum DGG simulations: Matthew Hur
Earlier software: Guy Yosiphon and Arthur York

MT fiber
Stochastic Parametrized Graph Grammar



$$(\bullet_1)\langle\!\langle (x_1,u_1)\rangle\!\rangle \longrightarrow (\circ_1 \longrightarrow \bullet_2)\langle\!\langle (x_1,u_1),(x_2,u_2)\rangle\!\rangle$$
$$\textbf{with}\ \ \hat{\rho}_{\text{grow}}([\mathrm{Y}_g])\mathcal{N}(x_1-x_2;Lu_1,\sigma)\mathcal{N}(u_2;u_1/(|u_1|+\epsilon),\epsilon),$$
$$(\blacksquare_1 \longrightarrow \circ_2)\langle\!\langle (x_1,u_1),(x_2,u_2)\rangle\!\rangle \longrightarrow (\blacksquare_2)\langle\!\langle (x_2,u_2)\rangle\!\rangle$$
$$\textbf{with}\ \ \hat{\rho}_{\text{retract}}([\mathrm{Y}_r])$$
$$\begin{pmatrix} \circ_1 \longrightarrow \circ_2 \longrightarrow \circ_3 \\ \bullet_4 \end{pmatrix}\langle\!\langle (x_1,u_1),(x_2,u_2),(x_3,u_3),(x_4,u_4)\rangle\!\rangle$$
$$\longrightarrow \begin{pmatrix} \circ_1 \longrightarrow \blacktriangle_2 \longrightarrow \circ_3 \\ \circ_4 \end{pmatrix}\langle\!\langle (x_1,u_1),(x_2,u_2),(x_3,u_3),(x_4,u_4)\rangle\!\rangle$$
$$\textbf{with}\ \ \hat{\rho}_{\text{bundle}}(|u_2\cdot u_4|/|\cos\theta_{\text{crit}}|)\exp\left(-|x_2-x_4|^2/2L^2\right)$$
$$\longrightarrow \begin{pmatrix} \circ_1 - \circ_2 - \circ_3 \\ \circ_4 - \blacksquare_5 \end{pmatrix}\langle\!\langle (x_1,u_1),(x_2,u_2),(x_3,u_3),(x_4,u_4)\rangle\!\rangle$$
$$\textbf{with}\ \ \hat{\rho}'_{\text{bundle}}(|u_2\cdot u_4|/|\cos\theta_{\text{crit}}|)\exp\left(-|x_2-x_5|^2/2L^2\right)$$
$$\longrightarrow \begin{pmatrix} & & \bullet_5 \\ & \circ_6 & \\ \circ_1 - & \blacklozenge_2 & - \circ_3 \\ & \circ_4 & \end{pmatrix}\langle\!\langle (x_1,u_1),(x_2,u_2),(x_3,u_3),(x_4,u_4),(x_5,u_5),(x_6,u_6)\rangle\!\rangle$$
$$\textbf{with}\ \ \hat{\rho}''_{\text{bundle}}(|u_2\cdot u_4|/|\cos\theta_{\text{crit}}|)\exp\left(-|x_2-x_5|^2/2L^2\right)$$

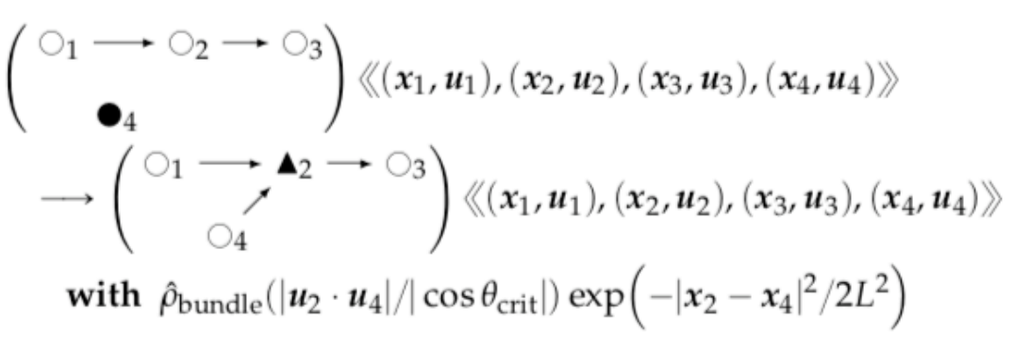


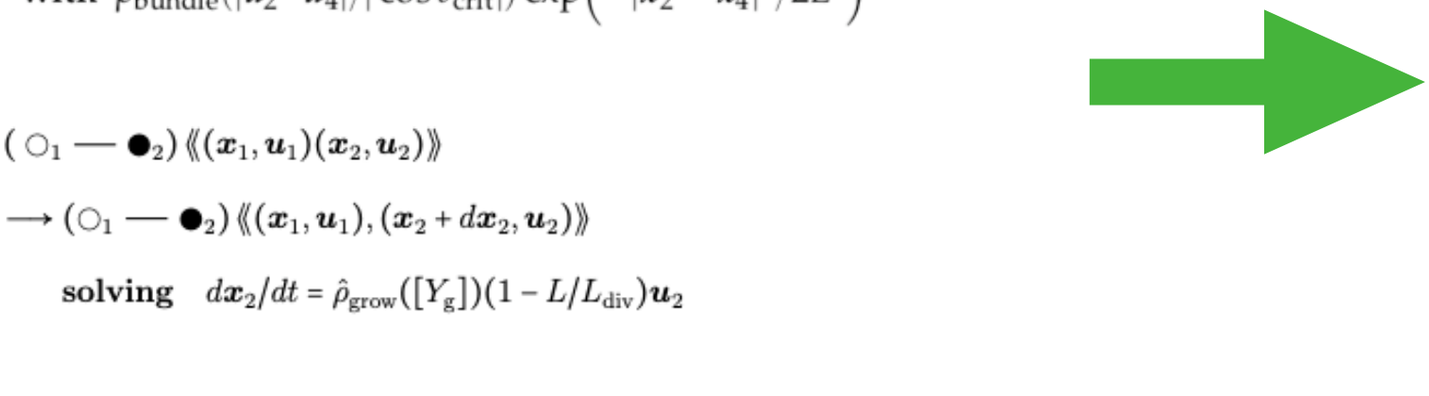
[Chakrabortty et al. Current Biology 2018]

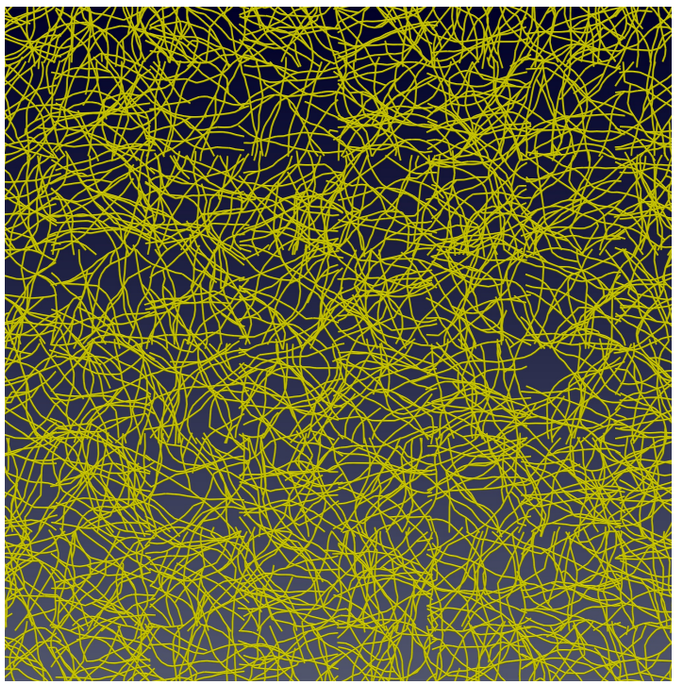
[EM, Bull. Math Biol. 81:8 Aug 2019 +arXiv:1804.11044]

Large-scale simulation

Dynamical graph grammar







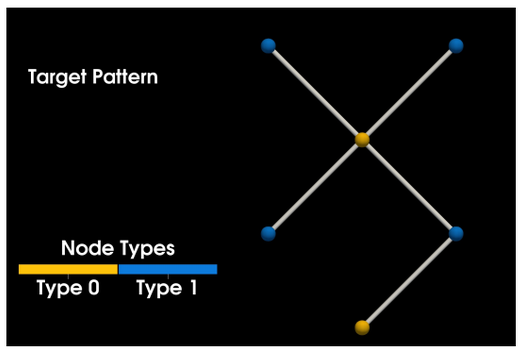
...

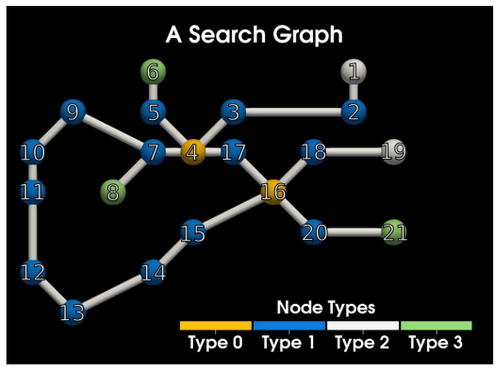
To infer: $L, L_{\text{div}}, \theta_{\text{crit}}$, params for $\rho_{\text{bundle}}, \rho_{\text{grow}}$

[Medwedeff and Mjolsness, Physical Biology, June 2023]

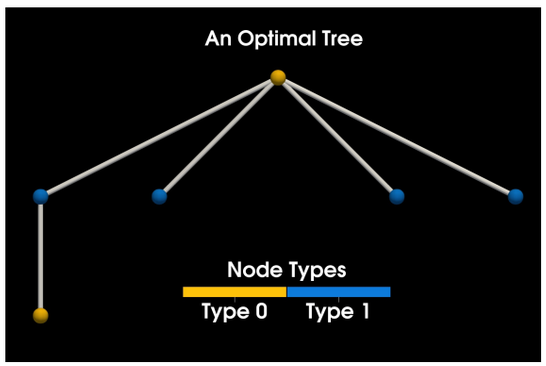
Parallel DGG algorithm:

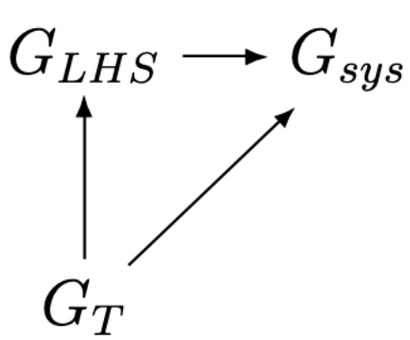
Simulated MT bundling, catastrophe



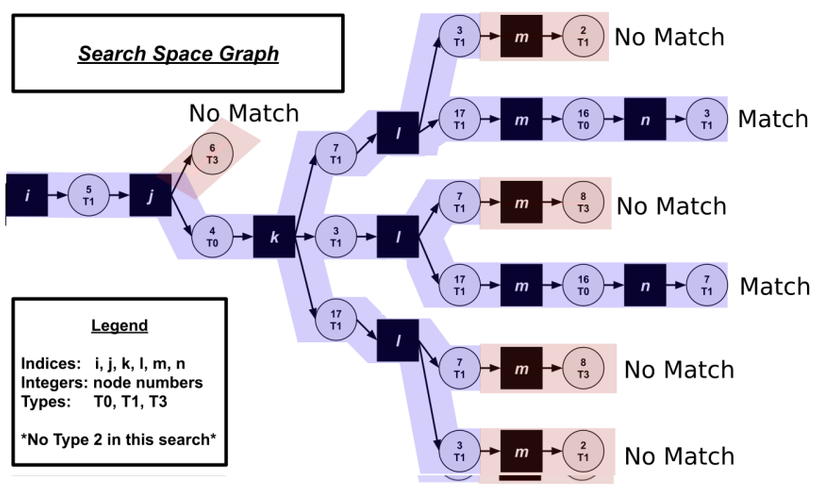


(b) Graph Pattern Target to Find





(b) Optimal Tree Transformation



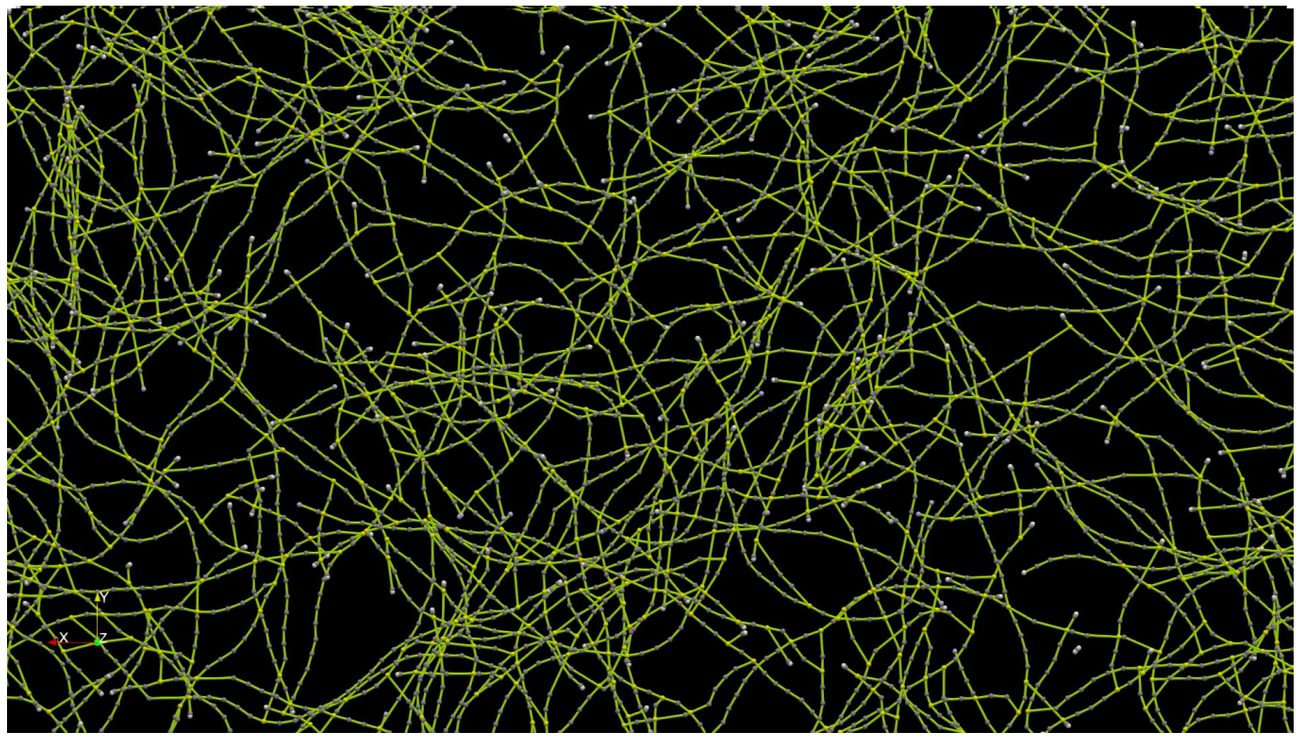


Figure 5.15: Randomly Selected 5x5 Zoomed of 5,000 MTs, T = 400

[Medwedeff and Mjolsness, Physical Biology, June 2023]

Cell Complex Operator Splitting

$$e^{tW} \approx \left(\prod_{d\downarrow} e^{\frac{t}{n} W_{(d)}} \right)^{n \to \infty}$$

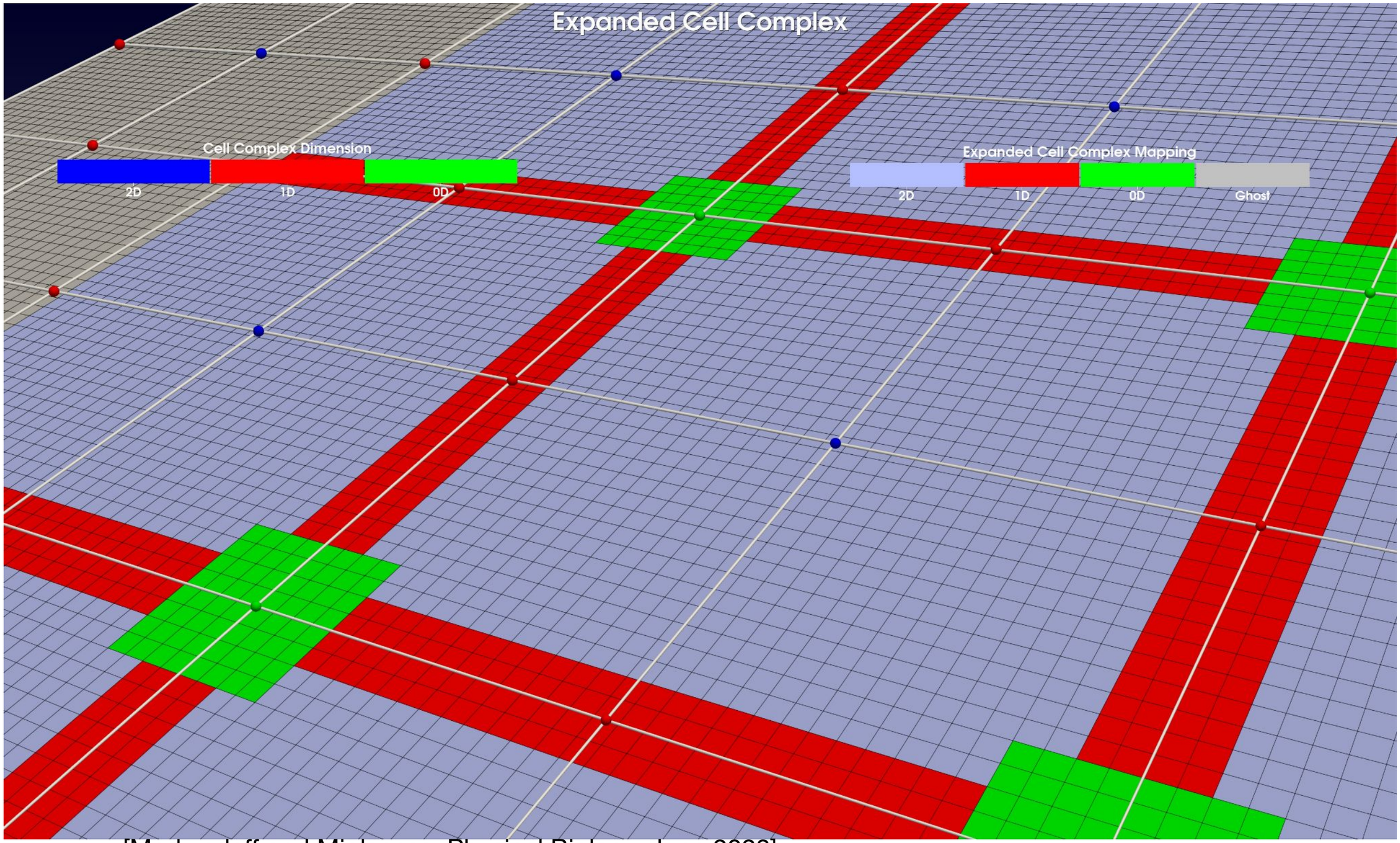
$$e^{t' W_{(d)}} = \prod_{c \subset d} e^{t' W_{(c,d)}} \quad \text{where} \quad \left[W_{(c,d)}, W_{(c',d)} \right] \approx 0 \quad \text{and} \quad t' \equiv \frac{t}{n}$$

$$W_{(c,d)} = \sum_r W_{r,c} \equiv \sum_r \sum_{\left\{ \begin{array}{l} R \mid \varphi(R)=c, \\ R \text{ instantiates } r \end{array} \right\}} W_r(R \mid c, d)$$

- Requires different c of the same d be well-separated.

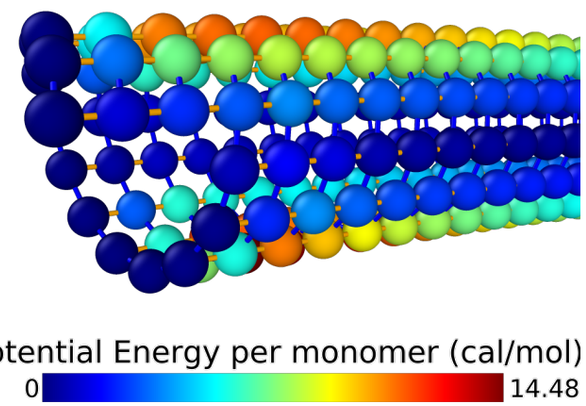
[Medwedeff and Mjolsness, Physical Biology, June 2023]

Expanded Cell Complex, for separation within dimension

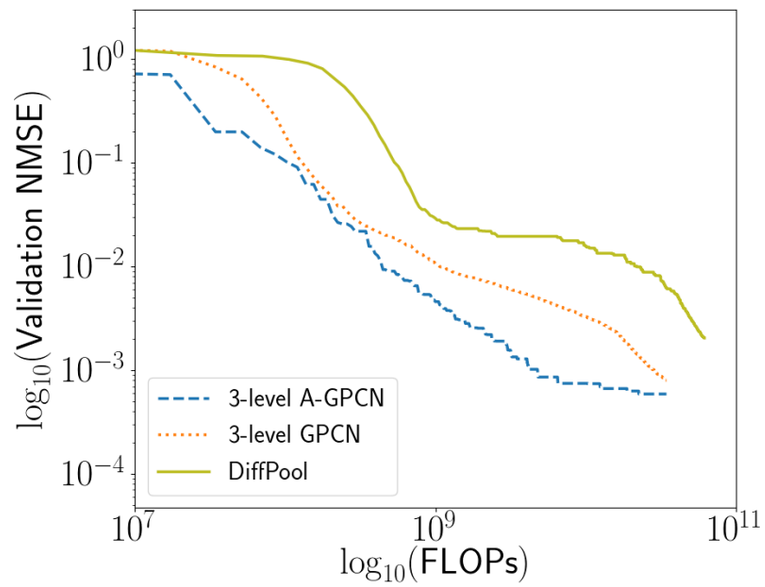


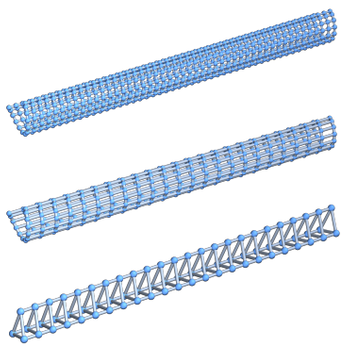
[Medwedeff and Mjolsness, Physical Biology, June 2023]

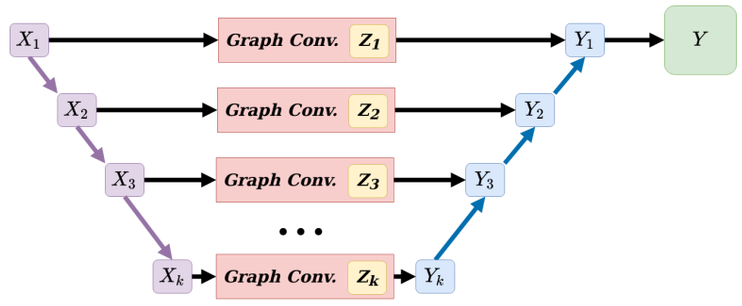
MT MD model reduction

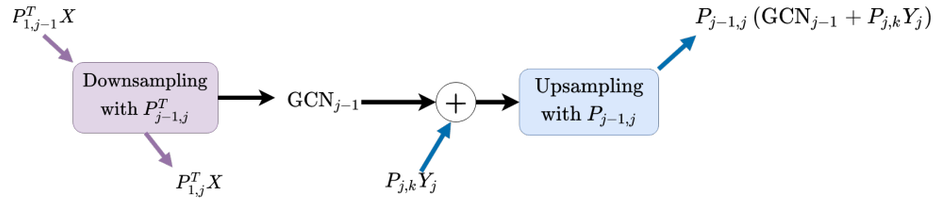


Train on:
LAAMPS simulations









[C. Scott and EM, ML:ST 2020 = arXiv:2002.05842;
cf. SIAM J Sci Comp 2019 = arXiv:1806.05703]

MaxEnt Problem

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

$$\text{w/ } \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) = \sum_{n=0}^{\infty} \int d\boldsymbol{x}\; p \ln \frac{p}{\tilde{p}}$$

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

Variational problem

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

where the variation is with respect to a set of **functionals**

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

… Higher-order calculus!

Slides: Oliver Ernst, Salk

O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

Variational Problem: Spatial systems

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

① ②

$$\frac{\delta S}{\delta \nu_{k'}(\boldsymbol{x}', t)} = \left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(\boldsymbol{x}' - \boldsymbol{x}_{\langle i \rangle_{k'}^{n}}) \right\rangle_{p} - \left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(\boldsymbol{x}' - \boldsymbol{x}_{\langle i \rangle_{k'}^{n}}) \right\rangle_{\tilde{p}} \qquad (20)$$

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

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

Need to choose a parametrization for functional!

O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

Computational problem

PDE-constrained Optimization Problem

$$\text{Minimize} \sum_{k'=1}^{K} \int_0^{\infty} dt \left(\left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(\boldsymbol{x}' - \boldsymbol{x}_{\langle i \rangle_{k'}^{n}}) \right\rangle_{p} - \left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(\boldsymbol{x}' - \boldsymbol{x}_{\langle i \rangle_{k'}^{n}}) \right\rangle_{\tilde{p}} \right) \frac{\delta \nu_{k'}(t)}{\delta F} \quad (23)$$

subject to PDE constraints for $\delta \nu_{k'}(t)/\delta F$.

O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

Adjoint method BMLA-like learning algorithm

Algorithm 1 Stochastic Gradient Descent for Learning Restricted Boltzmann Machine Dynamics

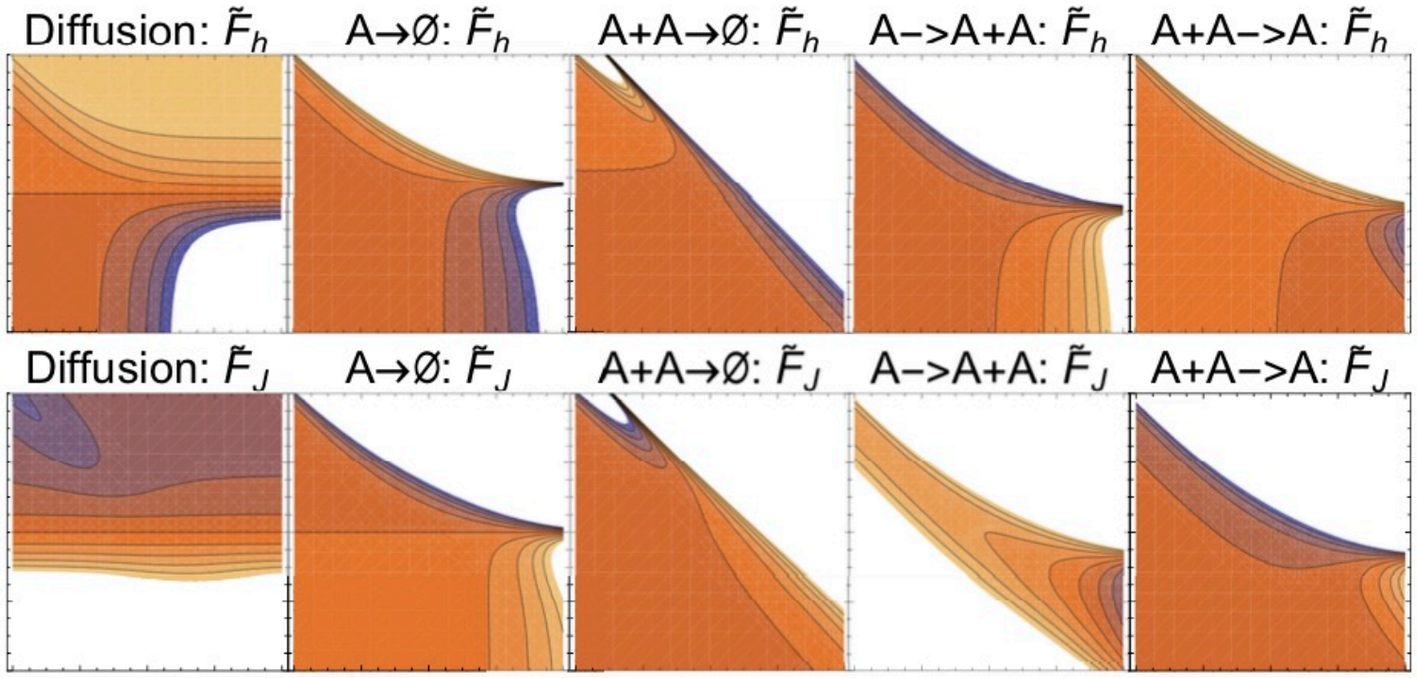
Initialize
  Parameters $\boldsymbol{u}_k$ controlling the functions $F_k(\boldsymbol{\theta}; \boldsymbol{u}_k)$ for all $k = 1, \ldots, K$.
  Time interval $[t_0, t_f]$, a formula for the learning rate λ.
while not converged **do**
  Initialize $\Delta F_{k,i} = 0$ for all $k = 1, \ldots, K$ and parameters $i = 1, \ldots, M_k$.
  for sample in batch **do**
    ▷ *Generate trajectory in reduced space* $\boldsymbol{\theta}$*:*
    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. ⟷ $\frac{d}{dt}\theta_k(t) = F_k(\boldsymbol{\theta}(t); \boldsymbol{u}_k)$
    ▷ *Wake phase:*
    Evaluate moments $\mu_k(t)$ of the data for all k, t.
    ▷ *Sleep phase:*
    Evaluate moments $\tilde{\mu}_k(t)$ of the Boltzmann distribution.
    ▷ *Solve the adjoint system:*
    Solve the adjoint system (31) for $\phi_k(t)$ for all k, t. ⟷ $\frac{d}{dt}\phi_k(t) = \tilde{\mu}_k(t) - \mu_k(t) - \sum_{l=1}^{K} \frac{\partial F_l(\boldsymbol{\theta}(t); \boldsymbol{u}_l)}{\partial \theta_k(t)} \phi_l(t),$
    ▷ *Evaluate the objective function:*
    Update $\Delta F_{k,i}$ as the cumulative moving average of the sensitivity equation (30) over the batch.
  ▷ *Update to decrease objective function:*
  $u_{k,i} \to u_{k,i} - \lambda \Delta F_{k,i}$ for all k, i.

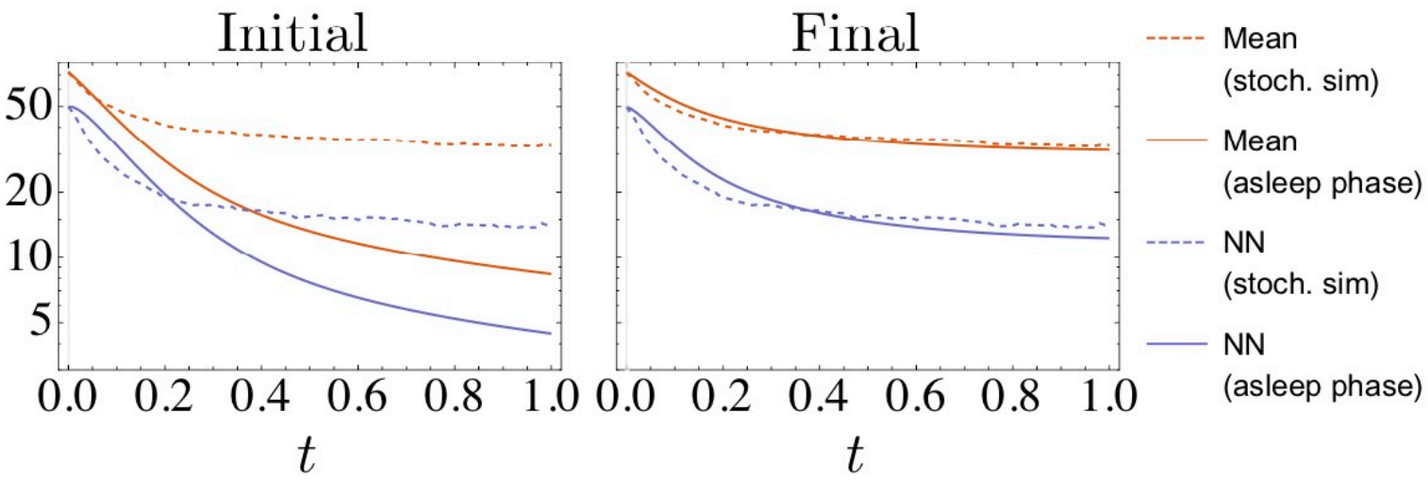
$\frac{dS}{du_{k,i}} \updownarrow = -\int_{t_0}^{t_f} dt \, \frac{\partial F_k(\boldsymbol{\theta}(t); \boldsymbol{u}_k)}{\partial u_{k,i}} \phi_k(t)$

[Ernst, Bartol, Sejnowski, Mjolsness, Phys Rev E 99 063315, 2019]

Spatial Dynamic Boltzmann Distributions

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





Computer algebra for basis functions:

djdtBran =

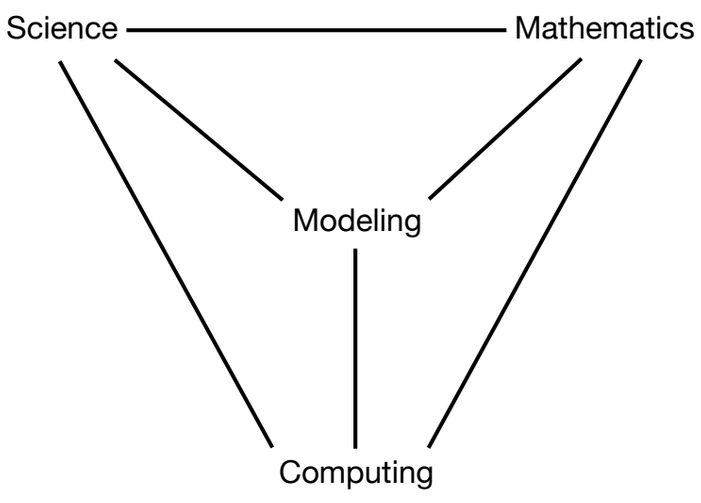
$$\left(e^{-h-j}\left(1+e^{2(h+j)}\left(7+2\sqrt{e^{-2(h+j)}\left(1+e^{h}\left(4+e^{j}\left(-2+e^{h+j}\right)\right)\right)}\right)+e^{h}\left(2+e^{j}\left(1+e^{2(h+j)}\left(-1+2e^{j}\right)\right)\left(1+\sqrt{e^{-2(h+j)}\left(1+e^{h}\left(4+e^{j}\left(-2+e^{h+j}\right)\right)\right)}\right)\right)\right)\Big/\left(4e^{2(h+j)}\,\mathrm{Cosh}[j]\right)\Big)\Big/\left(1+e^{h+j}\left(1+\sqrt{e^{-2(h+j)}\left(1+e^{h}\left(4+e^{j}\left(-2+e^{h+j}\right)\right)\right)}\right)\right)$$

Slides: Oliver Ernst, Salk

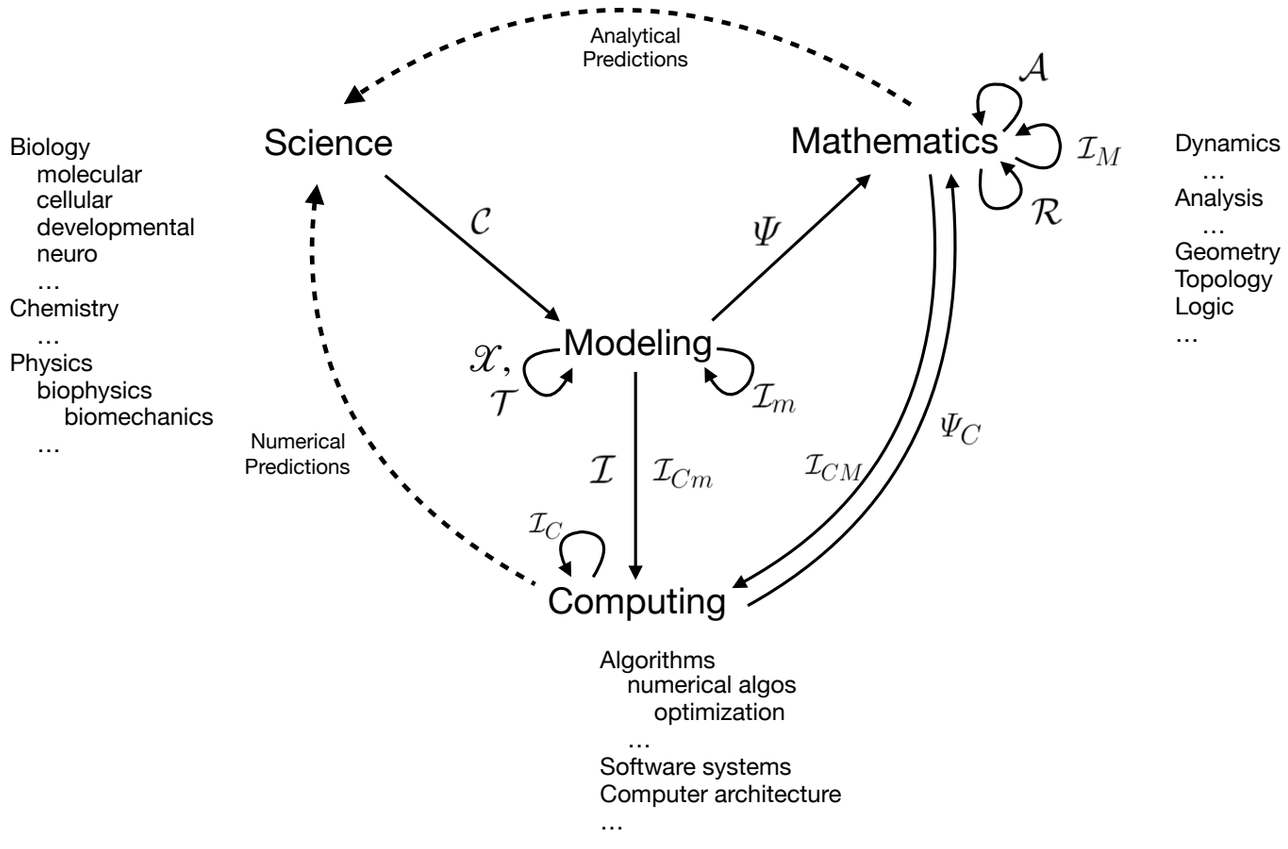
O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

"Tchicoma" Architecture for Mathematical Modeling

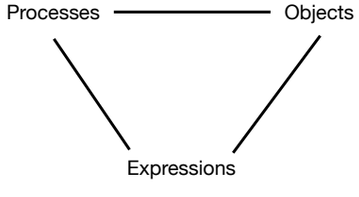
To formalize:



+ mappings and hierarchies:

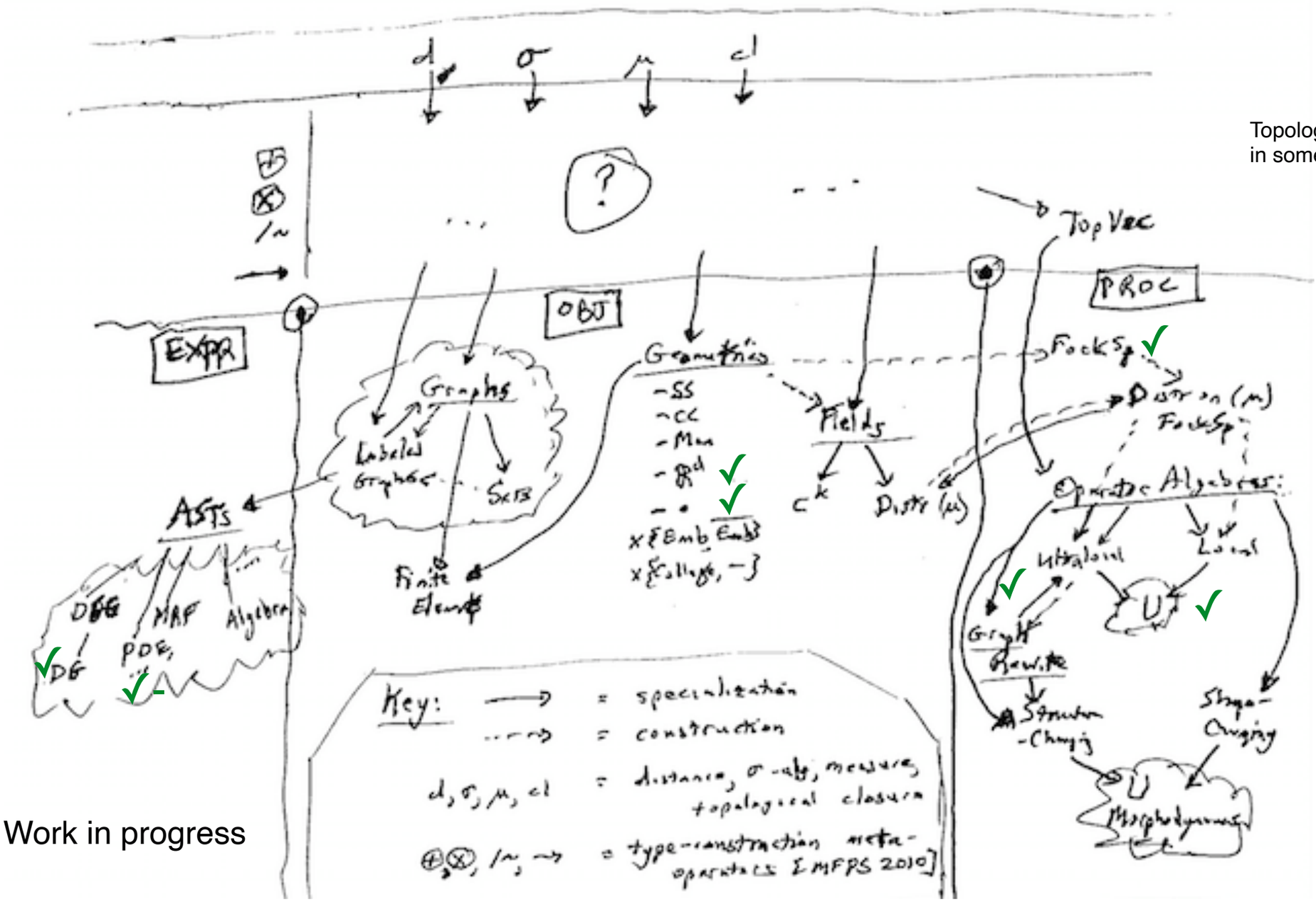


Each language has:



[EM, Bull. Math Biol. 81:8 Aug 2019
+arXiv:1804.11044]

“Eclectic Algebraic Type Theory” for mathematical type hierarchy



Topological categories go here, in some order. Eg. [MFPS 2010]

Work in progress

Eclectic Types

Context: The previous bio modeling languages shown above have a well-developed semantics of scientific “processes” modeled mathematically, but not for extended science-modeling “objects” beyond parameter-labelled graphs (which however covers a lot). E.g. no manifolds, dynamic function spaces, etc.. Such semantics could also undergird machine-assisted proof of some of the “arrows” in the Tchicoma architecture, e.g. algorithm-generation. The hope was that type theory could help with all this. So far no great luck.

Goals:
1. Generalize (loosen) the idea of a single “universal” Cartesian Closed Category as the semantics of a type theory of mathematical objects, to allow many base categories in a progressive curriculum, or none, while licensing a rich collection of type constructor notations to the extent possible. So, “eclectic” about foundations and expressive notations.
2. Define larger units of organization in a “type hierarchy”, to allow for looser coordination in expert development.

Caveat: This is all in the service of automated tools for scientific modeling, so might be (even?) worse for other areas of mathematics.

Eclectic Types

Definition. An ***eclectic type*** t is a triple (C_t, τ_t, L_t), subject to certain constraints, where C_t is a content category whose category-objects are the mathematical objects of eclectic type t, τ_t is a type object in a logical type category A, and L_t is a language equivalent to the minimal subset of the internal language of A that includes τ_t and all its predecessors in a partial ordering < on the objects of A. The constraints on C_t and τ_t are:

(1) finite product types $\tau_t = \prod_p \tau_{\sigma_p}$ in A: [technical condition on C_t] , and likewise for sums; &

(2) function types in A, $\tau_f = \tau_x \to \tau_t$, [technical condition on C_f] .

The partial ordering relation < is related to definability <# within a type hierarchy as follows:

$$<\# \implies < \quad \text{i.e.} \quad \forall a, b\ (a <\# b \implies a < b)$$

This way, required definitional statements can be made in the available sub-language L_t of each type.

Definition. A ***subtype*** (C_s, τ_s, L_s) of an eclectic type (C_t, τ_t, L_t) is an eclectic type that has both a [technical condition on C_s and C_t] and a subtyping relationship $\tau_s <: \tau_t$.

By the Liskov substitution principle, subtype relationships <: imply further such relationships among product types, sum types, and function types (“contravariantly” among the function argument types). This relationship contributes to generativity of the language fragment connected with an eclectic type in a type hierarchy. Likewise, the <# and hence < partial order relations must be consistent with product, sum, and function type constructors.

Definition. Assuming these kinds of consistency relations among <:, <#, and <, then we have a ***type hierarchy***.

Eclectic Types

Definition. In general, <# and < are related not only by type constructor consistency, but also by refinement: $<\# \implies <$. However, <: need not be related to < by refinement. When it is, the preferred relation has the opposite sense:

$$(:> \implies <\# \implies <) \quad \text{i.e.} \quad \forall a, b\ (a :> b \implies a <\# b \implies a < b)\ .$$

A subset of eclectic types, within a type hierarchy, that satisfies this relationship is called a ***type hierarchy module*** or where unambiguous a "*type module*".

Definition. A ***curriculum*** is a type hierarchy together with a set-covering collection of type modules within the type hierarchy, that form a DAG and hence a partial order, under the quotient new-generalization relationship: (<# and <:)/modules .

Spiral development is a special case of a curriculum.

A type module is easily extensible, but only by further definition (old <# new) and specialization (old :> new) which can of course be applied in combinatorial ways. Type hierarchy modules could be fruitful for symbolic AI search algorithms. When in the course of mathematical development new generalization supertypes are required, thus exceeding the limitations of a type module, the options are

(1) to begin a new type module within an existing type hierarchy, preserving existing <# definitional type relationships while potentially also creating a "curriculum" of modules of increasing generalization content, or

(2) to map the old type hierarchy into a new and more encompassing one, thus "restructuring" it.

In this way three levels and timescales of mathematical type-elaboration activity naturally emerge: incremental exploitation within modules, development of new modules, and foundational restructuring.

AITP for Science:

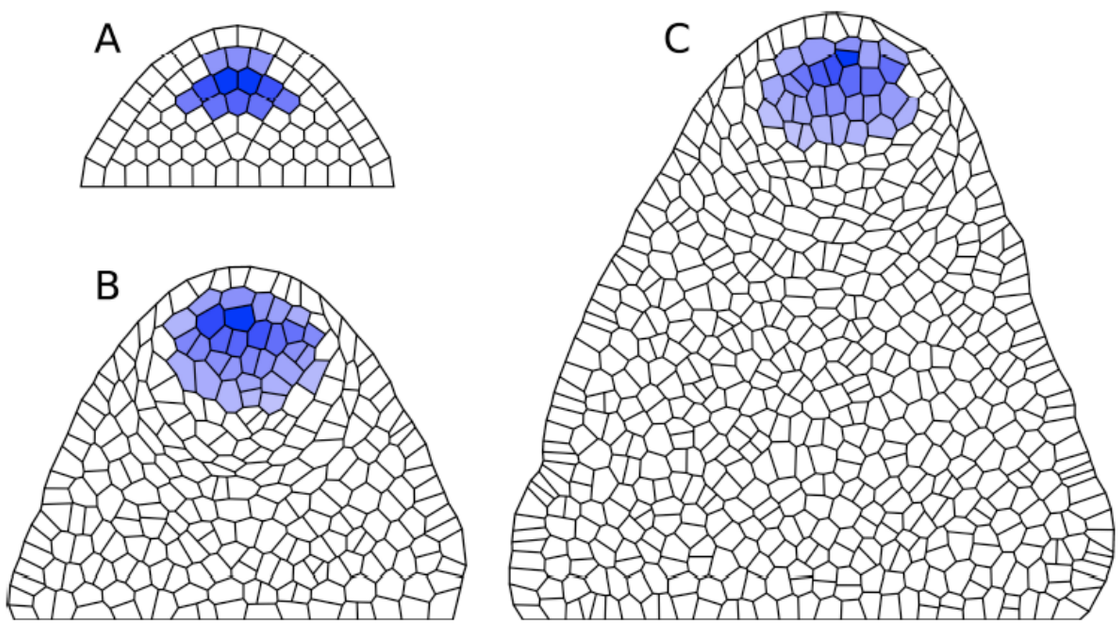
some logic in favor

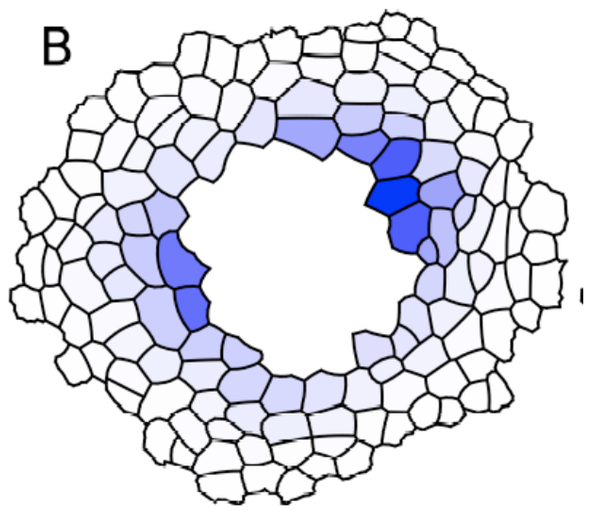
- “AI”: Reliable AI for Science:
  - Representationalism 1st, ML 2nd. (On top vs. on tap)
  - *formal, symbolic* DSLs for modeling …
    - applicable math, sci content, algorithms (model sim, ML, analysis)
- “TP”: verify exact and approximate maps between such languages
  - e.g. dynamics to simulation algs: XDEs, Markov jump processes,
  - high-level support for scientific knowledge representation

Eg: Plant gene expression model

Declarative, with cell growth & division

$$\{\{\emptyset \to \mathtt{U}, \mathtt{k_1 TIP[t]}\}, \{\mathtt{U} \to \emptyset, \mathtt{k_2}\}, \{\mathtt{U} \longrightarrow \mathtt{U}, \mathtt{Diffusion[D_U]}\},$$
$$\{\emptyset \to \mathtt{V}, \mathtt{k_3 L1[t]}\}, \{\mathtt{V} \to \emptyset, \mathtt{k_4}\}, \{\mathtt{V} \longrightarrow \mathtt{V}, \mathtt{Diffusion[D_V]}\},$$
$$\{\emptyset \rightleftarrows \mathtt{Z}, \mathtt{k_7}, \mathtt{k_8 U[t]}\}, \{\mathtt{X} \mapsto \mathtt{V}, \mathtt{GRN[v_V, T_{WV}, 1, h_V]}\},$$
$$\{\{\mathtt{U}, \mathtt{V}, \mathtt{W}\} \mapsto \mathtt{W}, \mathtt{GRN[v_W, \{T_{UW}, T_{VW}, T_{WW}\}, 1, h_W]}\}, \{\mathtt{W} \to \emptyset, \mathtt{k_6 Z[t]} + \mathtt{k_9 L2[t]}\}$$
$$\{\mathtt{W} \mapsto \mathtt{X}, \mathtt{GRN[v_X, T_{WX}, 1, h_X]}\}, \{\mathtt{X} \to \emptyset, \mathtt{k_5}\}, \{\mathtt{X} \longrightarrow \mathtt{X}, \mathtt{Diffusion[D_X]}\},$$
$$\{\mathtt{cell} \longrightarrow \mathtt{cell}, \mathtt{Grow[GrowthRate}[\mu, \mathtt{f}_\mu], \mathtt{Pressure[P, f_P]}, \mathtt{Spring[k, f_k]}\},$$
$$\{\mathtt{cell} \longrightarrow \mathtt{cell} + \mathtt{cell}, \mathtt{Errera[cell}, \mu, \sigma\}\}$$





[Shapiro et al Frontiers in Plant Science 2013]

Abstract

The complexity of biological systems (among others) makes demands on the complexity of the mathematical modeling enterprise that could be satisfied with mathematical artificial intelligence of both symbolic and numerical flavors. Technologies that I think will be fruitful in this regard include:

(1) the use of machine learning to bridge spatiotemporal scales, which I will illustrate with the "Dynamic Boltzmann Distribution" method for learning model reduction of stochastic spatial biochemical networks and the "Graph Prolongation Convolutional Network" approach to course-graining the biophysics of microtubules;

(2) a meta-language for stochastic spatial graph dynamics, "Dynamical Graph Grammars", that can represent structure-changing processes including microtubule dynamics and that has an underlying combinatorial theory related to operator algebras; and

(3) an integrative conceptual architecture of typed symbolic modeling languages and structure-preserving maps between them, including model reduction and implementation maps.

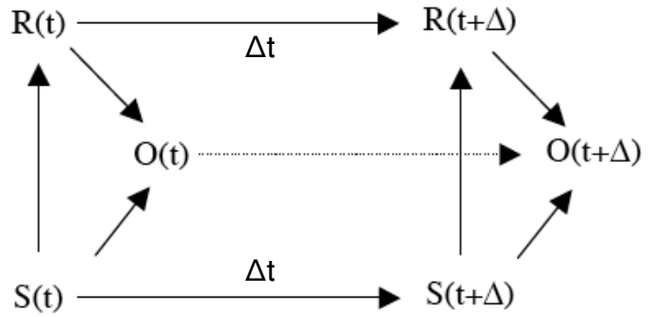
Machine learning for model reduction incorporating stat mech knowledge:

Dynamic Boltzmann Distributions for stochastic reaction-diffusion systems

Oliver Ernst, Salk & UCSD

Model Reduction

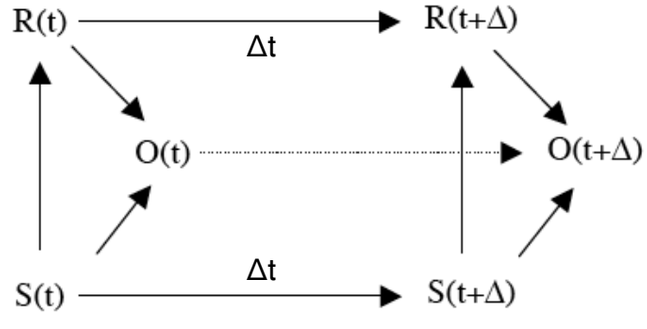
- For …
  - understandability
  - computational escalation
- Commutation:



$$\Psi\mathscr{R} \simeq \mathscr{R}\Psi$$

- To reduce within the paradigm, we need …
  - stochastic + deterministic dynamics
  - dynamic particles, fields, ***and graphs***

Mapping: Model reduction



$$\Psi\mathscr{R} \simeq \mathscr{R}\Psi$$

$$\frac{dp}{dt} = W \cdot p$$

[Johnson, Bartol, Sejnowski, and Mjolsness. Physical Biology 12:4, July 2015]

- Nonspatial: $\hat{p}(R,t) = \exp\left[-\sum_{\alpha}\mu_{\alpha}(t)V_{\alpha}(R)\right]/\hat{Z}(\mu(t))$

  [Johnson, Bartol, Sejnowski, and Mjolsness. Physical Biology 12:4, July 2015]

  - Graph-Constrained Correlation Dynamics
  - warmup case for …
- Spatial generalization: $\tilde{p}(n,\boldsymbol{x},\boldsymbol{\alpha},t) = \frac{1}{Z}\exp\left[-\sum_{k=1}^{K}\sum_{\langle j\rangle}\nu_k(\boldsymbol{x}_{\langle j\rangle},\boldsymbol{\alpha}_{\langle j\rangle},t)\right],$

  - Dynamic Boltzmann distributions

Stat mech knowledge context for ML:
Master eq. at fine scale
Dynamic Boltzmann distributions at coarse scale

MaxEnt Problem

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

$$\text{w/ } \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) = \sum_{n=0}^{\infty} \int d\boldsymbol{x}\; p \ln \frac{p}{\tilde{p}}$$

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

Variational problem

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

where the variation is with respect to a set of **functionals**

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

… Higher-order calculus!

Slides: Oliver Ernst, Salk

O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

Variational Problem: Spatial systems

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

① ②

$$\frac{\delta S}{\delta \nu_{k'}(\boldsymbol{x}', t)} = \left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(\boldsymbol{x}' - \boldsymbol{x}_{\langle i \rangle_{k'}^{n}}) \right\rangle_{p} - \left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(\boldsymbol{x}' - \boldsymbol{x}_{\langle i \rangle_{k'}^{n}}) \right\rangle_{\tilde{p}} \qquad (20)$$

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

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

Need to choose a parametrization for functional!

O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

Computational problem

PDE-constrained Optimization Problem

$$\text{Minimize} \sum_{k'=1}^{K} \int_0^{\infty} dt \left(\left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(\boldsymbol{x}' - \boldsymbol{x}_{\langle i \rangle_{k'}^{n}}) \right\rangle_{p} - \left\langle \sum_{\langle i \rangle_{k'}^{n}} \delta(\boldsymbol{x}' - \boldsymbol{x}_{\langle i \rangle_{k'}^{n}}) \right\rangle_{\tilde{p}} \right) \frac{\delta \nu_{k'}(t)}{\delta F} \quad (23)$$

subject to PDE constraints for $\delta \nu_{k'}(t)/\delta F$.

O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

Linearity of process operators

Proposition

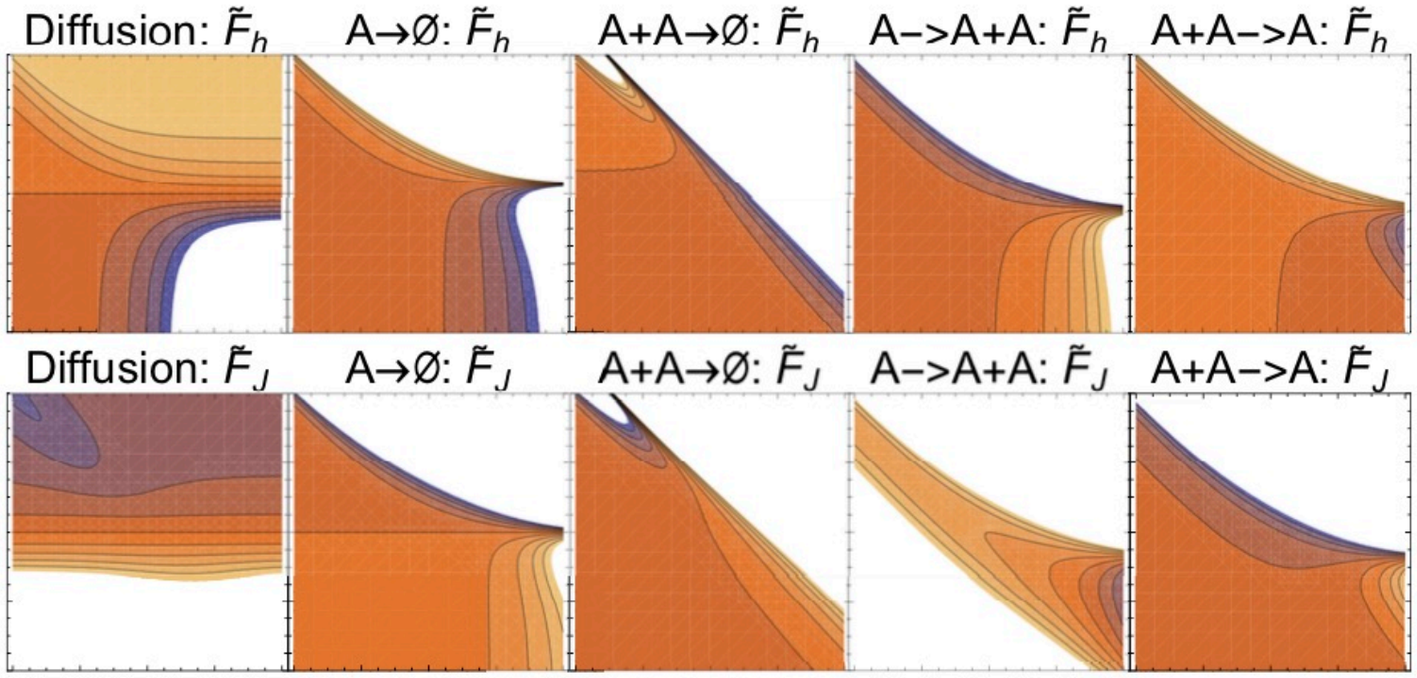
Fix: reaction network $\{\boldsymbol{W}_r\}$ *and graphical model* $\boldsymbol{\nu}$
Then: linearity of CME in reaction operators:
$\dot{\boldsymbol{p}} = \sum_r \boldsymbol{W}^{(r)} \boldsymbol{p}$
extends to the reduced model: $\tilde{F}_{1\alpha} = \sum_r \tilde{F}^{(r)}_{1\alpha}$ *and*
$\tilde{F}_{2\alpha\beta} = \sum_r \tilde{F}^{(r)}_{2\alpha\beta}$

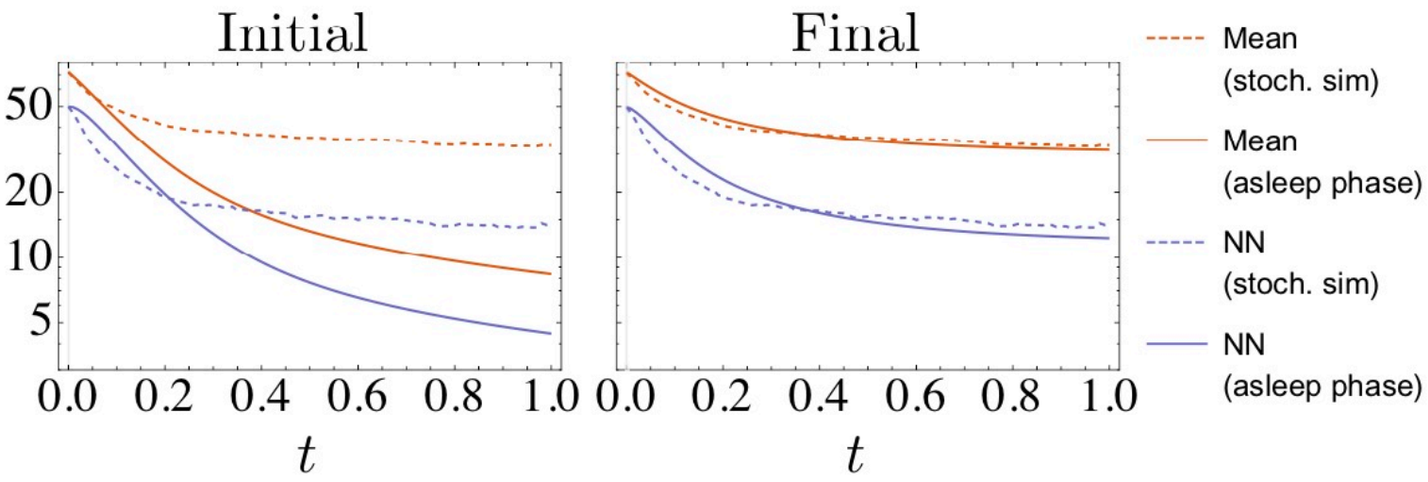
- Reason: Chain rule on MaxEnt optimal inverse statement

O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

Spatial Dynamic Boltzmann Distributions

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





Computer algebra for basis functions:

djdtBran =

$$\left(e^{-h-j}\left(1+e^{2(h+j)}\left(7+2\sqrt{e^{-2(h+j)}\left(1+e^{h}\left(4+e^{j}\left(-2+e^{h+j}\right)\right)\right)}\right)+e^{h}\left(2+e^{j}\left(1+e^{2(h+j)}\left(-1+2e^{j}\right)\right)\left(1+\sqrt{e^{-2(h+j)}\left(1+e^{h}\left(4+e^{j}\left(-2+e^{h+j}\right)\right)\right)}\right)\right)\right)\Big/\left(4e^{2(h+j)}\,\mathrm{Cosh}[j]\right)\Big)\Big/\left(1+e^{h+j}\left(1+\sqrt{e^{-2(h+j)}\left(1+e^{h}\left(4+e^{j}\left(-2+e^{h+j}\right)\right)\right)}\right)\right)$$

Slides: Oliver Ernst, Salk

O. Ernst, T. Bartol, T. Sejnowksi, and E. Mjolsness , J of Chem Phys 149, 034107, July 2018. Also arXiv 1803.01063

Adjoint method BMLA-like learning algorithm

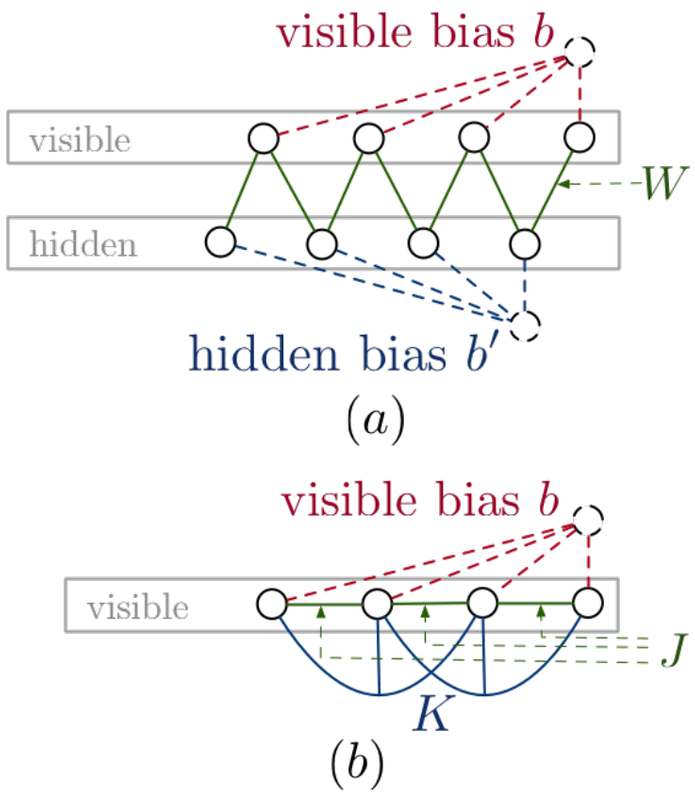
Algorithm 1 Stochastic Gradient Descent for Learning Restricted Boltzmann Machine Dynamics

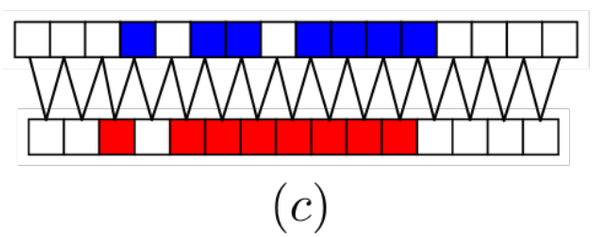
Initialize
  Parameters $\boldsymbol{u}_k$ controlling the functions $F_k(\boldsymbol{\theta}; \boldsymbol{u}_k)$ for all $k = 1, \ldots, K$.
  Time interval $[t_0, t_f]$, a formula for the learning rate λ.
while not converged **do**
  Initialize $\Delta F_{k,i} = 0$ for all $k = 1, \ldots, K$ and parameters $i = 1, \ldots, M_k$.
  for sample in batch **do**
    ▷ *Generate trajectory in reduced space* $\boldsymbol{\theta}$*:*
    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. ⟷ $\frac{d}{dt}\theta_k(t) = F_k(\boldsymbol{\theta}(t); \boldsymbol{u}_k)$
    ▷ *Wake phase:*
    Evaluate moments $\mu_k(t)$ of the data for all k, t.
    ▷ *Sleep phase:*
    Evaluate moments $\tilde{\mu}_k(t)$ of the Boltzmann distribution.
    ▷ *Solve the adjoint system:*
    Solve the adjoint system (31) for $\phi_k(t)$ for all k, t. ⟷ $\frac{d}{dt}\phi_k(t) = \tilde{\mu}_k(t) - \mu_k(t) - \sum_{l=1}^{K} \frac{\partial F_l(\boldsymbol{\theta}(t); \boldsymbol{u}_l)}{\partial \theta_k(t)} \phi_l(t),$
    ▷ *Evaluate the objective function:*
    Update $\Delta F_{k,i}$ as the cumulative moving average of the sensitivity equation (30) over the batch.
  ▷ *Update to decrease objective function:*
  $u_{k,i} \to u_{k,i} - \lambda \Delta F_{k,i}$ for all k, i.

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

Benefit of Hidden Units

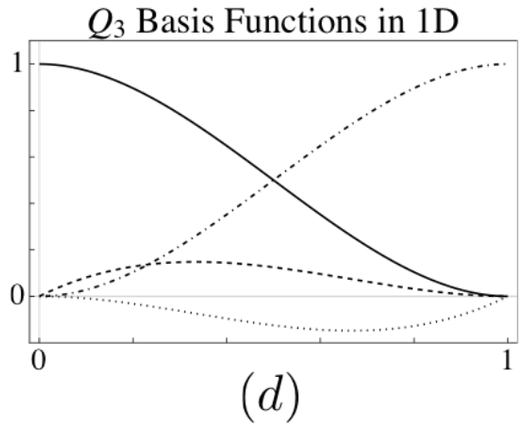
Network: fratricide + lattice diffusion





$$E(\boldsymbol{v},\boldsymbol{h},b(t),W(t),b'(t)) = -b(t)\sum_{i=1}^{N} v_i - b'(t)\sum_{j=1}^{N-1} h_j - W(t)\sum_{i=1}^{N}\sum_{j=i-1,i} v_i h_j,$$

$$\frac{d}{dt}\gamma = F_\gamma(b,b',W;\boldsymbol{u}_\gamma) \quad \text{for} \quad \gamma = b,b',W.$$



$$E(\boldsymbol{v},b(t),J(t),K(t)) = -b(t)\sum_{i=1}^{N} v_i - J(t)\sum_{i=1}^{N-1} v_i v_{i+1} - K(t)\sum_{i=1}^{N-2} v_i v_{i+1} v_{i+2},$$

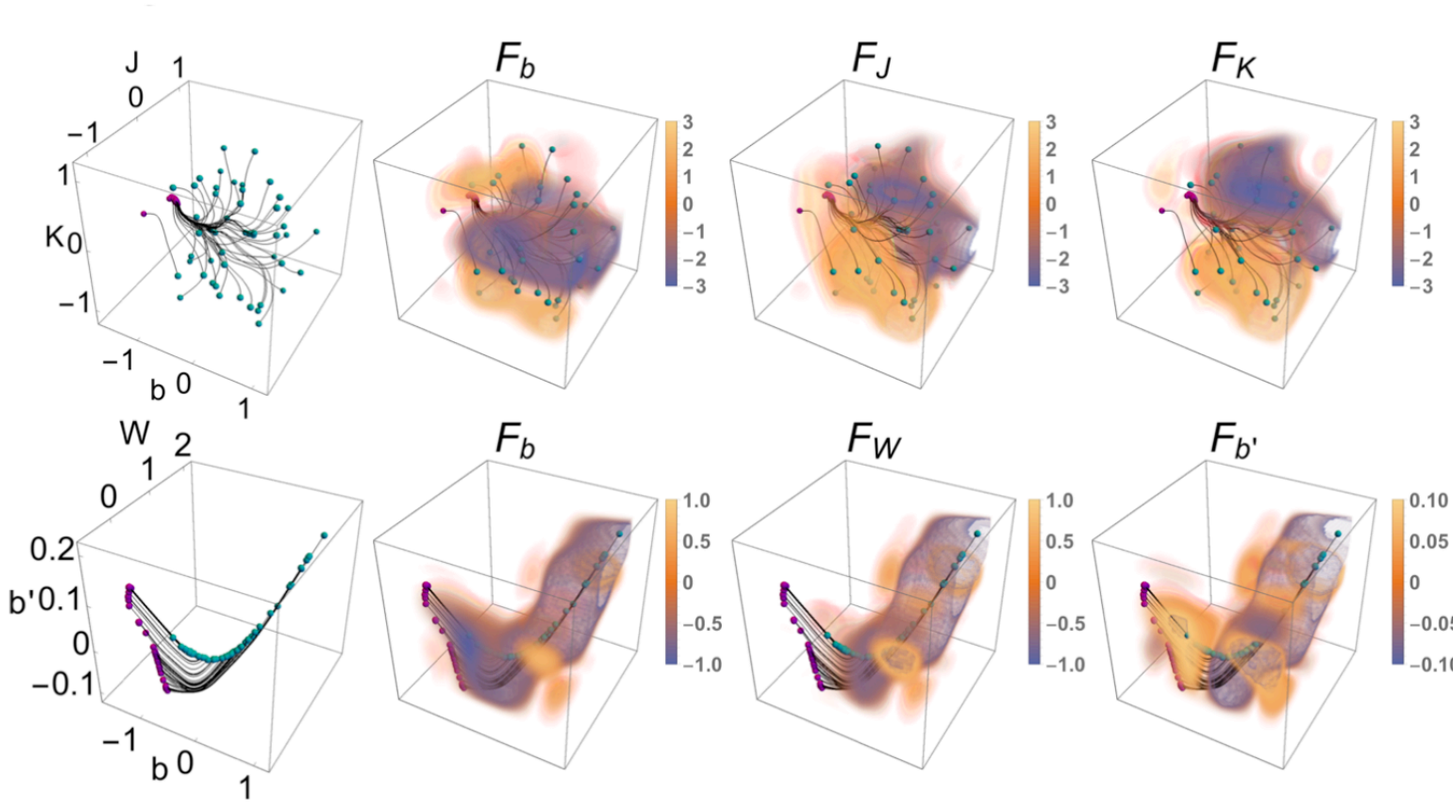
$$\frac{d}{dt}\gamma = F_\gamma(b,J,K;\boldsymbol{u}_\gamma) \quad \text{for} \quad \gamma = b,J,K.$$

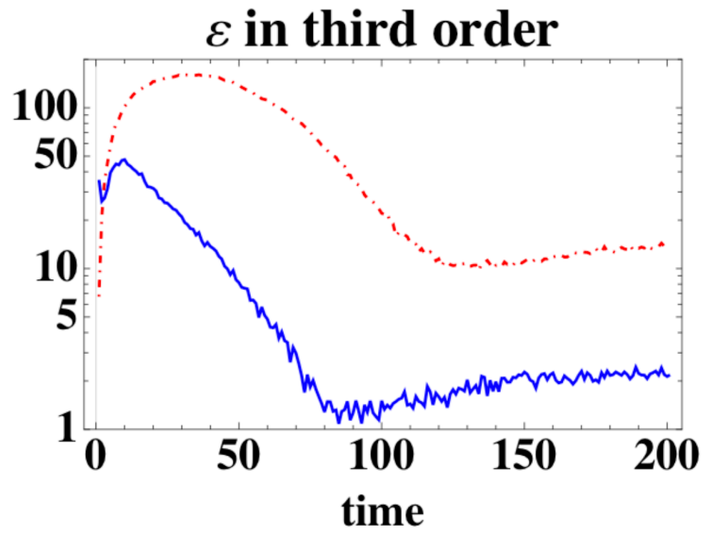
[Ernst, Bartol, Sejnowski, Mjolsness, arXiv:1808.08630 v2 April 2019]

Benefit of Hidden Units

Network: fratricide + lattice diffusion

- Learned DBD ODE RHS, without and with hidden units





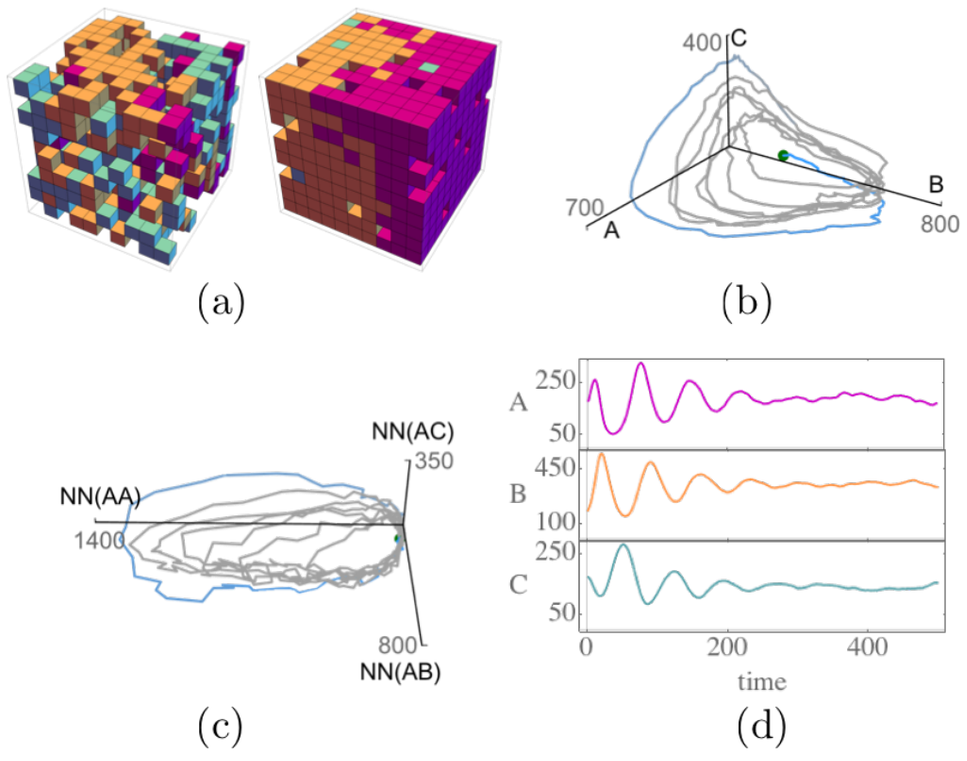
MSE of 3rd order stats

FIG. 2. Top row: Learned time-evolution functions for the fully visible model (38), using the Q_3, C_1 finite-element parametrization (34) with cells of size $0.5 \times 0.5 \times 0.5$ in (b, J, K). Left panel: 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). Middle three panels: The time evolution functions learned, where the heat map indicates the value of F_γ in (38). Right panel: Vertices of the finite-element cells used. Bottom row: Hidden layer model (40) and parametrization (34) with cells of size $0.5 \times 0.5 \times 0.05$ in (b, W, b'). Initial points are generated by BM learning applied to the points of the visible model. Note that the coefficients corresponding to the other seven degrees of freedom at each vertex are also learned (not shown), i.e., the first derivatives in each parameter.

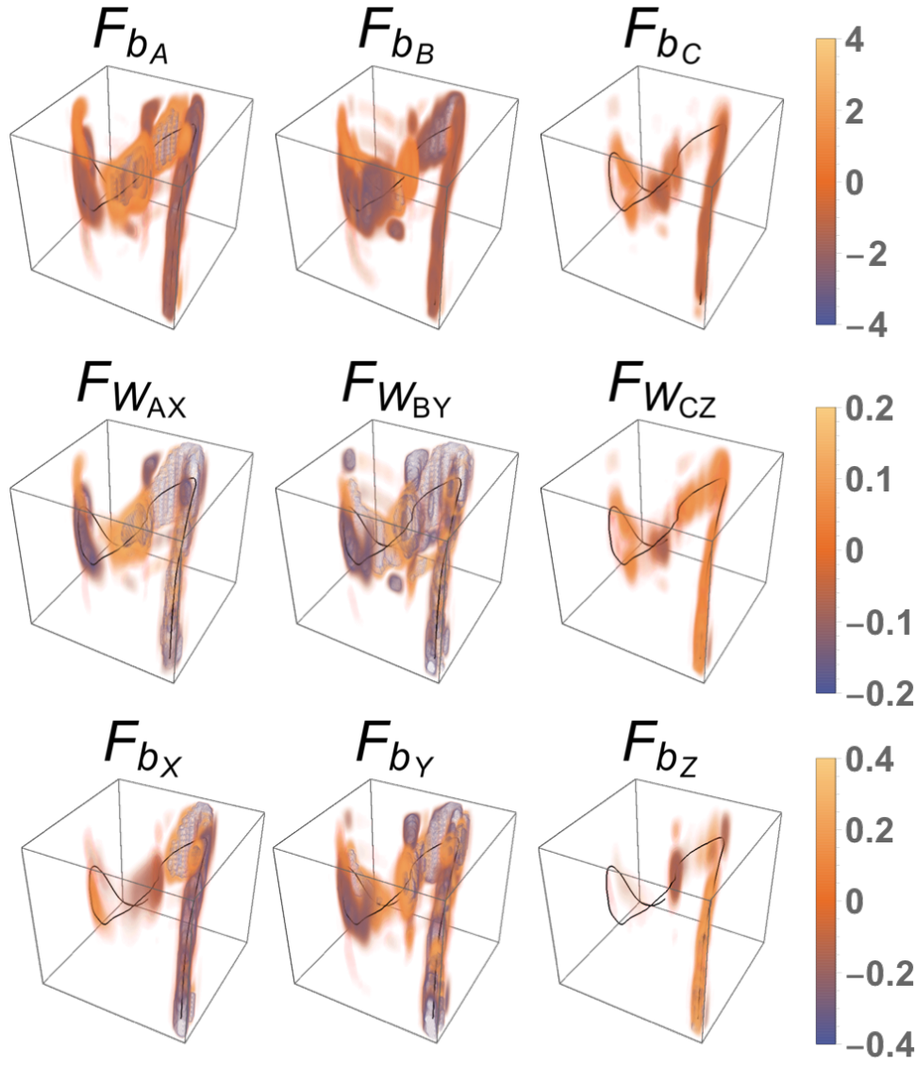
[Ernst, Bartol, Sejnowski, Mjolsness, Phys Rev E 99 063315, 2019]

Rössler Oscillator in 3D

- Function:



- Learned DBD ODE RHS:



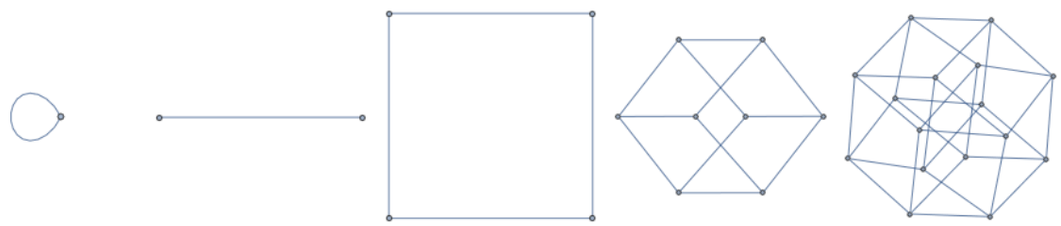
[Ernst, Bartol, Sejnowski, Mjolsness, arXiv:1808.08630 v2 April 2019]

Graph Prolongation Convolutional Networks
for biomechanical model reduction from molecular dynamics

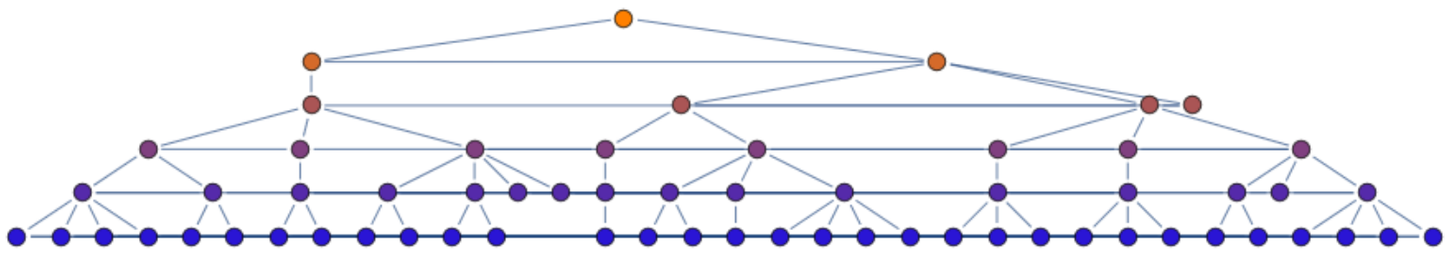
Cory Scott

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*: $O(b^{l1+\epsilon})$



- *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.

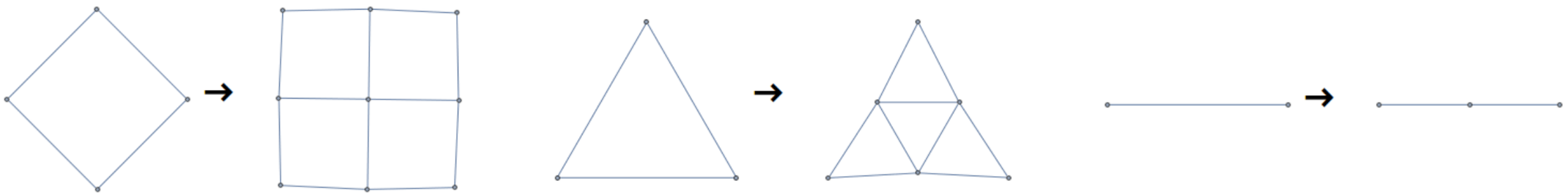


- *Graph Lineage:* a graded graph where the sequence of $\Delta L = 0$ subgraphs is a HGS and the subgraphs with $\Delta L = 1$ are a HGS of bipartite graphs. The above is a graph lineage of path graphs of length 2^n.

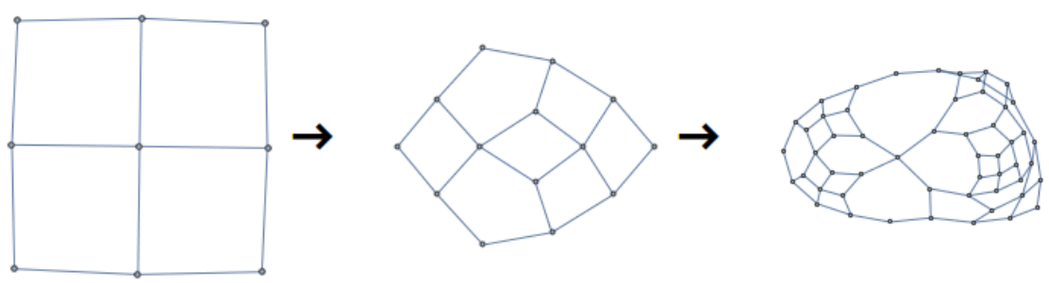
- *Hierarchitecture*: A graph lineage, used as a model architecture.

Generating Graph Lineages

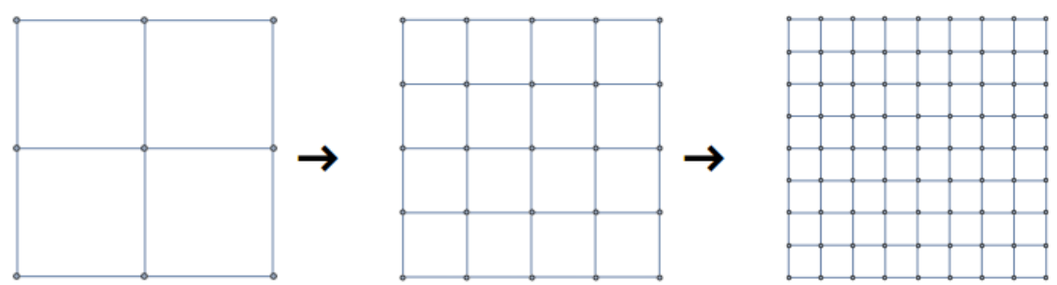
- One way to generate a graph lineage (or more generally, graded graphs) is via local graph rewrite rules.



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

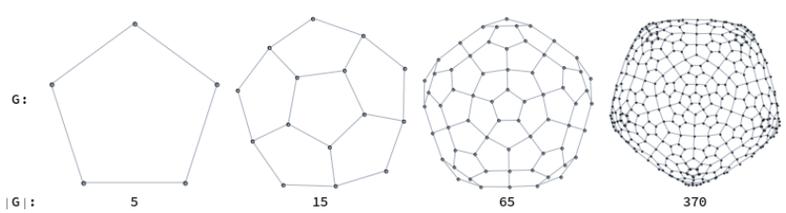


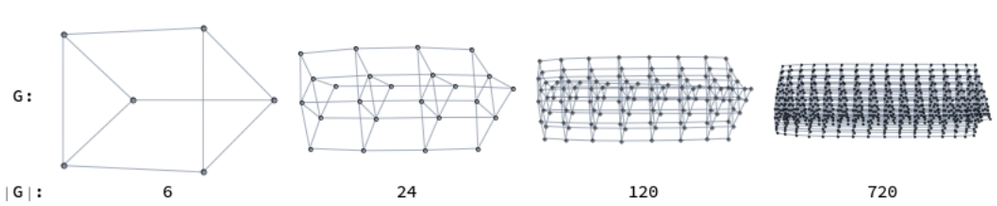
Local Firing



Global Firing

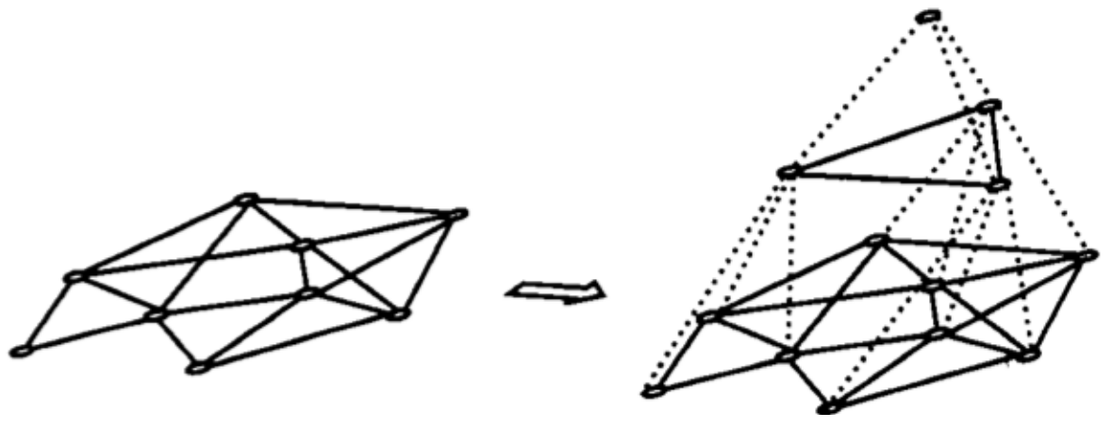
- Graph labels suppressed, but necessary
- More:





Multiscale numerics:
Alg. Multigrid Methods for Graphs

$$W' \simeq P^T W P$$



E.g. in optimization-based neural networks.

[EM et al., IEEE Trans. Neural Networks 1991]

Cf. $MG' \simeq GM$ for graph-matching [Gold et al. Neural Computation 8 1996]

Here: use graph Laplacian $L' \simeq P^T L P$, or $PL' \simeq LP$ with orthogonal $P^T P = I$

Define Graph Process Directed "Distances"

- Definition requires constrained opt of diffusion operator:

$$D(G_1, G_2|R, \alpha > 0, t) = \inf_{P|C(P)} ||P\exp(\alpha^{-1/2} t W_1^{(R)}) - \exp(\alpha^{1/2} t W_2^{(R)}) P||_F$$
$$D(G_1, G_2|R, t) = \inf_{\alpha>0} D(G_1, G_2|R, \alpha, t)$$

- Constraints: orthogonality; sparsity?

$$C(P): \quad \underbrace{P^T P = I}_{\text{restriction.prolongation}} \quad ; \quad \max \text{fanout}(P) \leq (n_{P\text{fine}}/n_{P\text{course}})^s$$

- Optimize time & time dilation due to graph size:

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

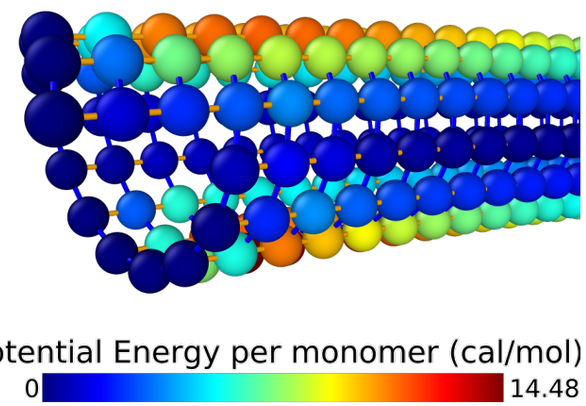
- Can obtain *P* at early times ("rigid" vs "flexible" def of *D*):

$$D_{\text{rigid}}(G_1, G_2|R, t) = \inf_{P|C(P)} ||P^* \exp(\alpha^{*-1/2} t W_1^{(R)}) - \exp(\alpha^{*1/2} t W_2^{(R)}) P^*||_F \text{, where}$$
$$(\alpha^*, P^*) = \text{argmin}_{\alpha>0, P|C(P)} ||\alpha^{-1/2} P W_1^{(R)} - \alpha^{1/2} W_2^{(R)} P||_F$$

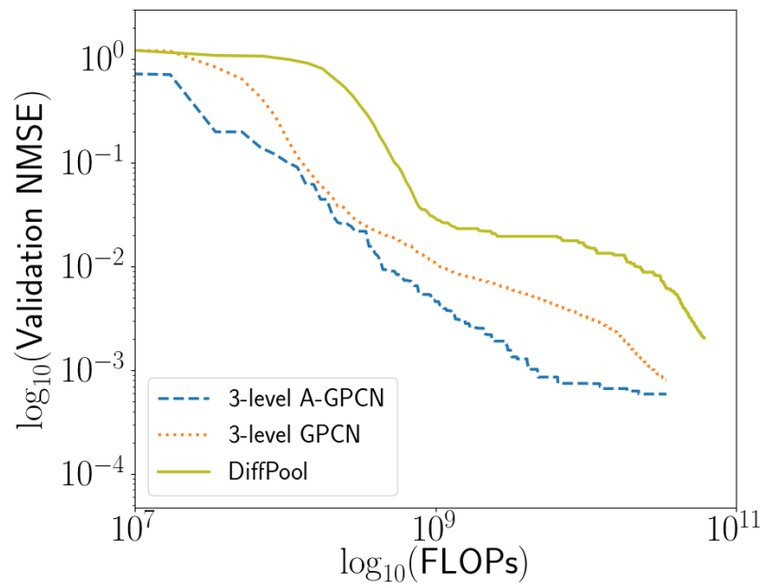
- $\triangle\leq$ provable with weaker α : $\boldsymbol{\alpha = \left(\frac{n_1}{n_2}\right)^r}$

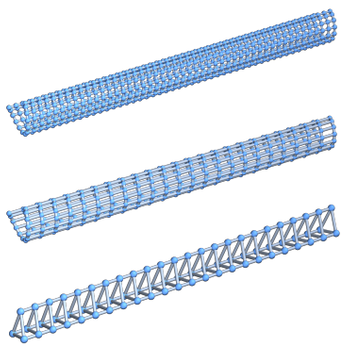
[C. Scott and EM, http://arxiv.org/abs/1909.04203]

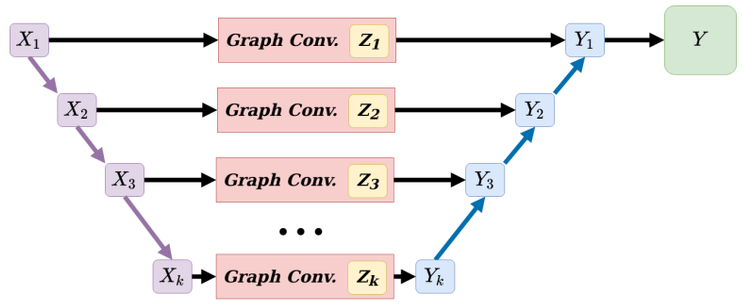
MT MD model reduction

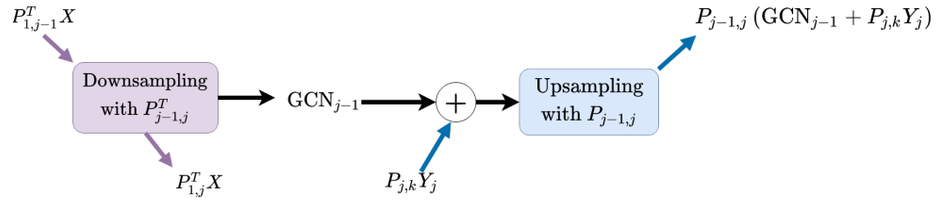


Train on:
LAAMPS simulations









[C. Scott and EM, ML:ST 2020 = arXiv:2002.05842;
cf. SIAM J Sci Comp 2019 = arXiv:1806.05703]