Machine learning and data science in soft materials design and engineering

IPAM MLP Tutorial Talk
9 September 2019
Andrew Ferguson, U. Chicago
Three waves of AI

1. Expert systems

Conventional rule based if-then algorithms programmed with expert knowledge to mimic human decision making.

2. Machine learning

Algorithms that train themselves to learn the rules from the data.

3. Explainable AI (XAI) + Physics-aware AI (PAI)

Algorithms that explain their actions and/or respect and exploit physical laws.
AI in molecular and materials science

- AI particularly valuable for:
  1. high-dimensional and/or complex systems that foil human intuition
  2. large conformational or combinatorial search spaces
  3. inverse problems — data but not models, goals but not mechanisms

Chemical discovery (i, ii)
— teaching machines chemical intuition and search

Medical diagnosis (i, iii)
— pathology identification, intervention design

Materials engineering (i, ii, iii)
— inverse materials design via the what (and the why)

Reaction engineering (ii, iii)
— optimizing conditions, predictive (retro)synthesis
Data science & domain science

- Data-driven inquiry emerging as a "fourth pillar" of science — knowledge discovery from data (KDD)

- Success is contingent on integration of data science paradigms and tools with domain specific knowledge and expertise (thermo, QM, rxn eng, ...)

A. Agrawal and A. Choudhary APL Mater. 4 053208 (2016)
Nonlinear machine learning in simulations of soft and biological materials

J. Wang* and A. L. Ferguson*bc

*Department of Physics, University of Illinois Urbana-Champaign, Urbana, IL, USA; bDepartment of Materials Science and Engineering, University of Illinois Urbana-Champaign, Urbana, IL, USA; cDepartment of Chemical and Biomolecular Engineering, University of Illinois Urbana-Champaign, Urbana, IL, USA

ABSTRACT
Interpretable parameterisations of free energy landscapes for soft and biological materials calculated from molecular simulation require the availability of ‘good’ collective variables (CVs) capable of discriminating the metastable states of the system and the barriers between them. If these CVs are coincident with the slow collective modes governing the long-time dynamical evolution, then they also furnish good coordinates in which to perform enhanced sampling to surmount high free energy barriers and efficiently explore and recover the landscape. Non-linear manifold learning techniques provide a means to systematically extract such CVs from molecular simulation trajectories by identifying and extracting low-dimensional manifolds lying latent within the high-dimensional coordinate space. We survey recent advances in data-driven CV discovery and enhanced sampling using non-linear manifold learning, describe the mathematical and theoretical underpinnings of these techniques, and present illustrative examples to molecular folding and colloidal self-assembly. We close with our outlook and perspective on future advances in this rapidly evolving field.

ARTICLE HISTORY
Received 11 August 2017
Accepted 29 October 2017

KEYWORDS
Enhanced sampling, free energy landscapes; non-linear manifold learning; protein folding; self-assembly

Review articles
1. Classical molecular dynamics in 15 minutes

2. Enhanced sampling in molecular simulation
   [ML-driven search of conformational space]
   - collective variable (CV) discovery
   - accelerated sampling in molecular simulations
   - **APPLICATION**: enhanced sampling of protein folding w/ auto-encoding ANNs

3. Data-driven design of self-assembling π-conjugated oligopeptides
   [DS-driven search of chemical space]
   - surrogate model construction
   - high-throughput virtual screening
   - **APPLICATION**: oligopeptide discovery w/ coarse-grained molecular simulation, variational autoencoders, Gaussian process regression, and active learning
Acknowledgements

Collab: Prof. J.D. Tovar (JHU)
       Prof. Howard Katz (JHU)
       Prof. André Schleife (UIUC)

GS:  Wei Chen
     Yutao Ma
     Kirill Shmilovich
     Beryl Zhang
     Xinran Lian
     Nick Rego
     Shiqi Chen

UG:  Olivia Dunne
     Gillian Shen
     Joseph Aulicino

Post-Docs: Dr. Hythem Sidky
           Dr. Brandon Peters
           Dr. Mingfei Zhao

Ferguson Laboratory

http://andrewferguson.uchicago.edu

Collab:
Prof. J.D. Tovar (JHU)
Prof. Howard Katz (JHU)
Prof. André Schleife (UIUC)

GS:  Wei Chen
     Yutao Ma
     Kirill Shmilovich
     Beryl Zhang
     Xinran Lian
     Nick Rego
     Shiqi Chen

UG:  Olivia Dunne
     Gillian Shen
     Joseph Aulicino

Post-Docs: Dr. Hythem Sidky
           Dr. Brandon Peters
           Dr. Mingfei Zhao

Ferguson Laboratory

http://andrewferguson.uchicago.edu
1. Classical molecular dynamics in 15 minutes

2. ANN accelerated sampling of molecular free energy landscapes [ML-driven search of conformational space]

3. Data-driven design of π-conjugated oligopeptides [DS-driven search of chemical space]
What is molecular dynamics?

- A computational microscope

- An experiment on a computer

- A simulation of the classical mechanics of atoms
**Milestones in MD**

1957
**Alder & Wainwright**
First MD simulation of hard sphere fluid

1960
**Gibson et al.**
Simulation of Cu radiation damage
Gibson, J.B., Goland, A.N., Milgram, M., and Vineyard, G.H. Phys. Rev. 120 1229 (1960)

1964
**Rahman**
First simulation of liquid Ar using realistic potential

1974
**Rahman & Stillinger**
First simulation of liquid water

1977
**McCammon et al.**
First protein simulation (BPTI) [8.8ps]

1994
**York et al.**
BPTI hydrated xtal [1ns]

2010
**Shaw et al.**
BPTI in water [1μs]

1998
**Duan & Kollman**
Villin headpiece in water [1μs]
The fundamental idea

- MD simulates atomic motions using classical mechanics
- Running a simulation is like cooking - just follow the recipe
- Three ingredients:

  1. An initial system configuration
     \[ [\vec{r}(t = 0), \vec{v}(t = 0)] \]

  2. A (classical) interaction potential for the system
     \[ V(\vec{r}) \]

  3. A way to integrate \( F = ma \)
     \[ r(t + \delta t) = 2r(t) - r(t - \delta t) + a(t)\delta t^2 \]
The fundamental idea

- Laplace’s Demon / “The Clockwork Universe”

“Given for one instant an intelligence which could comprehend all the forces by which nature is animated and the respective positions of the beings which compose it, if moreover this intelligence were vast enough to submit these data to analysis, it would embrace in the same formula both the movements of the largest bodies in the universe and those of the lightest atom; to it nothing would be uncertain, and the future as the past would be present to its eyes.”

- Pierre Simon de Laplace (1749-1827)

This is essentially molecular dynamics
Ingredient 1: Initial configuration

- Specification of initial atomic coordinates and velocities
- Classical mechanics is deterministic: initial state and interaction rules fully specify the system’s future*.
- Wind up Laplace’s clockwork universe and — in principle — a “vast intelligence” could compute the future of the system.
- Our intelligence is insufficiently vast — the equations are hard! — and thus we resort to numerical simulation.

* neglecting numerical integration errors and finite precision (i.e., uncertainty)

www.ks.uiuc.edu
Initial configurations can be generated “by hand” or short scripts for simple systems (e.g., liquid Ar, bulk Al).

Software tools for complex systems (e.g., proteins, complex defect structures):

- PRODRG (http://davapc1.bioch.dundee.ac.uk/prodrg/)
- ATP (http://compbio.biosci.uq.edu.au/atb/)
- PyMOL (http://www.pymol.org/)
- Chimera (http://www.cgl.ucsf.edu/chimera/)

Common protein structures are in Protein Data Bank (PDB) (www.rcsb.org/pdb)
Initializing velocities

- Bad idea to start atoms from rest (absolute zero = 0 K) due to thermal shock upon starting simulation

- Standard approach is to draw velocities randomly from a Maxwell-Boltzmann distribution at the temperature, $T$

$$f_v(v_x, v_y, v_z) = \left( \frac{m}{2\pi kT} \right)^{3/2} \exp \left[ -\frac{m(v_x^2 + v_y^2 + v_z^2)}{2kT} \right]$$
The net force acting on each atom in the system is a result of its interactions with all other atoms.

These interaction amount to a set of rules known as a force field or interaction potential.

Accurate, robust, and transferable force fields are critical to perform physically realistic molecular simulations.

Force field development is an academic industry.

- **Metals:** EAM (Daw & Baskes), MEAM (Baskes)
- **Biomolecules:** Amber (Kollman, UCSF), GROMOS (U. Groningen), CHARMM (Karplus, Harvard), OPLS (Jorgensen, Yale), MARTINI [coarse grained] (Marrink, U. Groningen)
- **Polymers:** TraPPE (Siepmann, U. Minnesota), MM2 (Allinger, UGA)
- **Water:** SPC (Berendsen), SPC/E (Berendsen), TIPnP(Jorgensen), ST2 (Stillinger & Rahman)
- **General:** DREIDING (Mayo et al.), DISCOVER(Rappe et al.), UFF (Hagler et al.)
The potential energy of the system is a complicated function of atomic coordinates (this is why we have to simulate numerically rather than calculate analytically).

The net force on atom $i$ is the negative gradient of the potential energy with respect to the atomic coordinates:

$$ F_i = -\nabla_i [V(r_1, r_2, \ldots, r_N)] $$

$$ a_i = \frac{F_i}{m_i} $$

The potential energy is typically broken into four parts:

$$ V(\vec{r}) = V_{\text{bonded}} + V_{\text{non-bonded}} + V_{\text{restraints}} + V_{\text{field}} $$
In the GROMOS:96 force field:

- Fourth power potential
- Harmonic potential

### 4.2.1 Bond stretching

The simplest improper dihedral potential is a harmonic potential; it is plotted in Fig. 4.2.11 Improper dihedrals

Improper dihedrals are meant to keep planar groups.

### 4.2.6 Cosine based angle potential

The bond angle vibration between a triplet of atoms is represented by a harmonic potential, and the form is the same as the appropriate angle vibration.

Proper dihedrals: Ryckaert-Bellemans function

Improper dihedrals: periodic type

The numbering of multiple parameters is convenient. Type 9 allows multiple potential functions to be applied automatically to a single dihedral when multiple parameters are defined for the same atom types in the input in topology files. Angles are given in degrees and force constants in kJ mole\(^{-1}\) rad\(^{-2}\). Taxonomic interactions must be included in the potential function for reasons of computational efficiency. Written as:

\[
V_b (r_{ij}) = \frac{1}{2} k_b r_{ij} (r_{ij} - b_{ij})^2
\]

\[
V_a (\theta_{ijk}) = \frac{1}{2} k^a \theta_{ijk} (\theta_{ijk} - \theta^0_{ijk})^2
\]

\[
V_{rb} (\phi_{ijkl}) = \sum_{n=0}^{5} C_n (\cos(\psi))^n
\]

\[
V_{id} (\xi_{ijkl}) = \frac{1}{2} k_\xi (\xi_{ijkl} - \xi_0)^2
\]
Non-bonded

- Approximate full $n$-body interactions as pairwise additive for simplicity and computational efficiency

- van der Waals

- Coulomb

\[
V_{LJ}(r_{ij}) = 4\epsilon \left[ \left( \frac{\sigma}{r_{ij}} \right)^{12} - \left( \frac{\sigma}{r_{ij}} \right)^{6} \right]
\]

\[
V_{Coul}(r_{ij}) = \frac{1}{4\pi\varepsilon_0} \frac{q_i q_j}{r_{ij}}
\]


http://guweb2.gonzaga.edu/faculty/cronk/chemistry/images/graph-electrostatic-PE-alt.gif
Restraints can be part of, or supplemental, to a force field.

Many applications, common uses include:
- fixed bond lengths and angles (esp. for light atoms)
- artificially immobilize part of the system (e.g., rigid walls or boundary condition)
Fields are commonly used to model:
1. external potentials (e.g., electric field, flow field)
2. continuum solvation (no explicit solvent molecules)
Ingredient 3: Integrators

- [initial atomic coordinates and velocities] + [force field]  
  $\Rightarrow$ entire future (and past!) modeled by $F=ma$

- Analytical solutions for the dynamical evolution cannot be computed for all but the simplest systems (>2 body).

- Solve Newton’s equations by numerical integration  
  $\Rightarrow$ computers ideally suited to rapid, repetitive calculations

- Solving by hand would require thousands of years
Many possible integration algorithms exist (e.g., explicit/implicit Euler, Gear predictor-corrector, $n^{th}$ order Runge-Kutta, Beeman, Newmark-beta)

The method of choice is the **Verlet algorithm**

- fast
- simple
- low-memory
- stable
- time-reversible
- symplectic (phase space volume & E conserving)

✗ poor accuracy for large time steps ($\Delta t$ must be small)

First recorded use by Delambre in 1791
Popularized in MD by Loup Verlet in 1967
Verlet algorithm

- Derived from Taylor series:

\[ r(t + \delta t) = r(t) + \dot{r}(t)\delta t + \frac{1}{2} \ddot{r}(t)\delta t^2 + \ldots \]

\[ = r(t) + v(t)\delta t + \frac{1}{2} a(t)\delta t^2 + \ldots \]

\[ r(t - \delta t) = r(t) - \dot{r}(t)\delta t + \frac{1}{2} \ddot{r}(t)\delta t^2 + \ldots \]

\[ = r(t) - v(t)\delta t + \frac{1}{2} a(t)\delta t^2 + \ldots \]

\[ r(t + \delta t) = 2r(t) - r(t - \delta t) + a(t)\delta t^2 + \mathcal{O} (\delta t^4) \]

\[ v(t) = \frac{r(t + \delta t) - r(t - \delta t)}{2\delta t} + \mathcal{O} (\delta t^2) \]

\[ a_i = \frac{F_i}{m_i} \]
Time-reversibility

Higher order integration algorithms have higher per step accuracy, enabling longer time steps and faster simulations (e.g., Runge-Kutta, Gear predictor-corrector)

But, do not respect time reversibility of Newton's equations causing energy drift and error accumulation

http://einstein.drexel.edu/courses/Comp_Phys/Integrators/leapfrog/errors.gif
Simulation overview

1. **START**
2. Initialize Positions and Velocities
3. Calculate forces for all molecules using Potential
4. Apply Thermostat and Volume Changes
5. Update Position and Velocities
6. Analyze the Data
7. **Till Termination Condition**
<table>
<thead>
<tr>
<th>Software</th>
<th>License</th>
<th>Website</th>
</tr>
</thead>
<tbody>
<tr>
<td>GROMACS</td>
<td>FREE</td>
<td><a href="http://www.gromacs.org">www.gromacs.org</a></td>
</tr>
<tr>
<td>CHARMM</td>
<td></td>
<td><a href="http://www.charmm.org">www.charmm.org</a></td>
</tr>
<tr>
<td>AMBER</td>
<td></td>
<td><a href="http://www.ambermd.org">www.ambermd.org</a></td>
</tr>
<tr>
<td>NAMD</td>
<td>FREE</td>
<td><a href="http://www.ks.uiuc.edu">www.ks.uiuc.edu</a></td>
</tr>
<tr>
<td>Desmond</td>
<td>FREE</td>
<td><a href="http://www.deshawresearch.com">www.deshawresearch.com</a></td>
</tr>
<tr>
<td>LAMMPS</td>
<td>FREE</td>
<td><a href="http://lammps.sandia.gov">http://lammps.sandia.gov</a></td>
</tr>
<tr>
<td>HOOMD-blue</td>
<td>FREE</td>
<td><a href="http://codeblue.umich.edu/hoomd-blue/">http://codeblue.umich.edu/hoomd-blue/</a></td>
</tr>
<tr>
<td>Folding@home</td>
<td>FREE</td>
<td><a href="http://folding.stanford.edu">http://folding.stanford.edu</a></td>
</tr>
<tr>
<td>OpenMM</td>
<td>FREE</td>
<td><a href="http://openmm.org">http://openmm.org</a></td>
</tr>
</tbody>
</table>
1. Classical molecular dynamics in 15 minutes

2. ANN accelerated sampling of molecular free energy landscapes [ML-driven search of conformational space]

3. Data-driven design of π-conjugated oligopeptides [DS-driven search of chemical space]
Limitations of molecular simulation

“Two limitations in existing simulations are the approximations in the potential energy functions and the lengths of the simulations. The first introduces systematic errors and the second statistical errors.”


1. Accurate force fields

2. Sampling configurational space
Limitations of molecular simulation

“Two limitations in existing simulations are the approximations in the potential energy functions and the lengths of the simulations. The first introduces systematic errors and the second statistical errors.”


1. Accurate force fields

2. Sampling configurational space
Accelerated sampling techniques partition largely into two classes:

- **Tempering techniques**
  - Simulated annealing
  - Multicanonical algorithm
  - Replica exchange
  - Hamiltonian exchange
  - Parallel tempering
  ...

- **Collective variable biasing**
  - Umbrella sampling
  - Hyperdynamics
  - Metadynamics
  - Adiabatic free energy dynamics (AFED)
  - Temperature accelerated dynamics (TAD)
  - Temperature accelerated MD (TAMD)
  - Adaptive force biasing
  ...

- **Tempering** modifies T or Hamiltonian to accelerate barrier crossing → substantial CPU time expended on conditions not of direct interest

- **CV biasing** efficiently directs sampling along relevant order parameters → presupposes *a priori* availability of “good” CVs
Accelerated sampling

Accelerated sampling techniques partition largely into two classes:

- **Tempering techniques**
  - Simulated annealing
  - Multicanonical algorithm
  - Replica exchange
  - Hamiltonian exchange
  - Parallel tempering
  - ...

- **Collective variable biasing**
  - Umbrella sampling
  - Hyperdynamics
  - Metadynamics
  - Adiabatic free energy dynamics (AFED)
  - Temperature accelerated dynamics (TAD)
  - Temperature accelerated MD (TAMD)
  - Adaptive force biasing
  - ...

- **Tempering** modifies T or Hamiltonian to accelerate barrier crossing
  ➔ substantial CPU time expended on conditions not of direct interest

- **CV biasing** efficiently directs sampling along relevant order parameters
  ➔ presupposes *a priori* availability of “good” CVs

Automated CV discovery

- Given a simulation trajectory data mining / dimensionality reduction can discover “good” CVs that:
  
  1. Separate metastable system states
  2. Characterize important large-scale or slow conformational motions
  3. Are explicit differentiable functions of atomic coordinates

- (3) is required to propagate CV biases to atomic forces

\[ f_{i}^{\text{tot}} = f_{i}^{U} + f_{i}^{B}. \]

**Linear dim red**
(e.g., PCA, MDS)
- Many required to separate states
- Poor descriptors of molecular motion
- Explicit function of atomic coords

**Nonlinear dim red**
(e.g., LLE, Isomap, dMaps)
- Parsimonious state separation
- Coincident with large-scale motions
- Unknown mapping to atomic coords
"[n]o method can presently extract reaction coordinates on the fly during MD simulations and at the same time use them to enhance the sampling of the configurational space"


**CV biasing: The chicken and the egg**

Good CVs required to drive sampling of configurational space (chicken)

Trajectories with good sampling needed to discover good CVs (egg)
Interleaved CV discovery and biased simulation

Biased simulations in current CVs to expand exploration

1. Nonlinear learning to update CV estimate

- **Biasing step frustrated by absence of CV mapping to atomic coords**

- Approximate solutions:
  1. Correlate data-mined CVs with physical variables in which to do biasing
  2. Select from (linear combinations of) known CVs
  3. Use CVs not for biasing but smart initialization of new runs
  4. Approximate CVs with functional fit or by “landmarks” in CV embedding


Selection among **known** CVs

REAP (Shukla et al.)

RAVE (Tiwary et al.)

\[ r^K(c_m) = \sum_{i=1}^{k} w_i s^i \frac{|\langle \theta_i(c_m) \rangle - \langle \theta_i(C) \rangle|}{\sigma_i(C)} \]

\[ D_{KL}(P(z) \| P(\chi)) = \sum_{i} P^u(z_i) \log \frac{P^u(z_i)}{P^u(\chi_i)} \]

Nonlinear dim red + smart initialization

DM-d-MD (Clementi et al.)

iMapD (Kevrekidis et al.)
Auto-associative neural networks (autoencoders)

- **Autoencoders** unique among unsupervised nonlinear dimensionality reduction tools in furnishing **explicit** and **differentiable** latent space map

![Autoencoder Diagram](https://inspirehep.net/record/1252540/files/autoencoder.png)

**Neuron activation function**

\[ y_k^{(i)} = f^{(i)}(x_k^{(i)}) \]

- Input to node \( k \) in layer \( i \)
- Activation function (e.g., tanh)
- Output of node \( k \) in layer \( i \)

**Weighted sums between layers**

\[ x_k^{(i)} = b_k^{(i)} + \sum_j w_{jk}^{(i-1)} y_j^{(i-1)} \]

- Bias to node \( k \) in layer \( i \)
- Weight from node \( j \) in layer \( (i-1) \) and node \( k \) in layer \( i \)
Auto-associative neural networks (autoencoders)

- **Autoencoders** unique among unsupervised nonlinear dimensionality reduction tools in furnishing **explicit** and **differentiable** latent space map

![Diagram of autoencoder](https://inspirehep.net/record/1252540/files/autoencoder.png)

- Idea is to discover and parameterize with CVs a low-dim manifold from which atomic coordinates can be approximately reconstructed

\[
\Theta_{\text{proj}} : \mathcal{H} \rightarrow \mathcal{L}
\]

\[
\hat{z}_q = (\Theta_{\text{rec}} \circ \Theta_{\text{proj}})z_q
\]
Implementing the bias

- Generically apply bias through artificial potential in CVs

\[ P(r^N) = \frac{e^{-\beta E(r^N)}}{Z} = \frac{e^{-\beta[H(r^N)+V(CV(r^N))]} \cdot Z}{Z} \]

where CVs are explicit and differentiable functions of atomic coords

\[ CV_i = CV_i(r^N) \text{ ugly, but explicit, function of input atomic coords and autoencoder weights, biases, and activation functions} \]
Implementing the bias

- Generically apply bias through artificial potential in CVs

\[ P(r^N) = \frac{e^{-\beta E(r^N)}}{Z} = \frac{e^{-\beta[H(r^N)+V(CV(r^N))]} }{Z} \]

where CVs are explicit and differentiable functions of atomic coords

\[ CV_i = CV_i(r^N) \quad \text{ugly, but explicit, function of input atomic coords and autoencoder weights, biases, and activation functions} \]

- Perform biased MD by analytically propagating CV bias into atomic forces

**Energy**

\[ E(r^N) = H(r^N) + V(CV(r^N)) \]

**Force**

\[ f_i(r^N) = -\nabla_{r_i} H(r^N) - \frac{\partial V}{\partial CV} \nabla_{r_i} CV(r^N) \]
• Autoencoders permit biased simulation directly in the discovered CVs

• Interleaved on-the-fly learning and biasing:
  Online biasing implemented in OpenMM as custom force plugin
  Offline autoencoder training over trajectory using Pytorch Python libraries
Alanine dipeptide in vacuum (Amber99sb)
Alanine dipeptide in vacuum (Amber99sb)
Alanine dipeptide in vacuum (Amber99sb)
Al Alanine dipeptide in vacuum (Amber99sb)

- MESA **converges within 10 iterations** to quantitatively accurate FES
- Autoencoder discovers correct 4D flat torus topology with two periodic collective variables \{\Phi, \Psi\}
- Timings on single Intel i7-5820K CPU core:

  10 × training 21-40-2-40-21 networks w/ Q=1500 & N=16 1200 s
  1 × 800 ps unbiased simulation 12 s
  75 × 10 ps biased simulations 130 s
  22 CPU-mins
Open-source availability

https://github.com/weiHelloWorld/accelerated_sampling_with_autoencoder

https://github.com/weiHelloWorld/ANN_Force
1. Classical molecular dynamics in 15 minutes

2. ANN accelerated sampling of molecular free energy landscapes [ML-driven search of conformational space]

3. Data-driven design of $\pi$-conjugated oligopeptides [DS-driven search of chemical space]
Supramolecular biocompatible optoelectronics

- Synthetic $\pi$-conjugated peptides can self-assemble into 10-100 nm fibers
- Fibers possess emergent optical and electronic functionality due to e-$\pi$ delocalization along overlapping $p$ orbitals
- Absorption of UV light produces transient electric fields, exciton generation, and organic photovoltaic activity

B.A. Thurston, E.P. Shapera, J.D. Tovar, A. Schleife, and ALF (submitted, 2019)
Sequence–structure–function relation

- Peptide-wing and π-core sequence programs self-assembly behavior
- Self-assembled structure governs optical and electronic function

B.A. Thurston and ALF Mol. Sim. 44 11 930-945 (2018)
Coarse-grained MD of oligopeptide assembly

- Coarse-grained MARTINI bead-level representation of oligopeptides
- Compromise between accuracy and speed — can predict aggregation of hundreds of oligopeptides over microseconds

96 × DGAG-OPV3-GAGD
GROMACS 2018
Martini w/ explicit non-polarizable water
T = 298 K, P = 1 bar
t = 3,000 ns (100 h wall time)
Coarse-grained MD of oligopeptide assembly

- Coarse-grained MARTINI bead-level representation of oligopeptides
- Compromise between accuracy and speed — can predict aggregation of hundreds of oligopeptides over microseconds

96 × DGAG-OPV3-GAGD
GROMACS 2018
Martini w/ explicit non-polarizable water
T = 298 K, P = 1 bar
t = 3,000 ns (100 h wall time)
Coarse-grained MD of oligopeptide assembly

- Coarse-grained MARTINI bead-level representation of oligopeptides
- Compromise between accuracy and speed — can predict aggregation of hundreds of oligopeptides over microseconds

The curse of dimensionality:

The DXXX-Π-XXXD family comprises $20^3 = 8,000$ sequences for each Π core

$DXXXX-Π-XXXXD \implies 20^4 = 160,000$

$DXXXXX-Π-XXXXXD \implies 20^5 = 3,200,000$

...

Trial-and-improvement AA or CG simulation too slow for high-throughput virtual screening and rational design

Martini w/ explicit non-polarizable water

$T = 298 \text{ K}, P = 1 \text{ bar}$

$t = 3,000 \text{ ns (100 h wall time)}$

**Machine learning can help**

1. **Unsupervised** nonlinear learning of low-dimensional oligopeptides representations using **variational autoencoders**

2. **Supervised** learning of sequence—morphology relation using **Gaussian process regression**

3. **Active learning** to optimally deploy computational effort to explore oligopeptide sequence space
Machine learning can help

1. **Unsupervised** nonlinear learning of low-dimensional oligopeptides representations using variational autoencoders

   Learn featurization

2. **Supervised** learning of sequence—morphology relation using Gaussian process regression

   Estimate fitness

3. **Active learning** to optimally deploy computational effort to explore oligopeptide sequence space

   Explore sequence space

---

R. Gomez-Bombarelli, J.N. Wei, ..., and A. Aspuru-Guzik  

---

© 2018 American Chemical Society

Cite This:  
ACS Cent. Sci. 2018, 4, 268−276  
DOI: 10.1021/acscentsci.7b00572

---

* Ryan P. Adams,  
† Benjamín Recht,  
‡ Rafael Gömez-Bombarelli,  
́ Jennifer N. Wei,  
⊥ Dennis Sheberla,  
¶ Jose Miguel Hernández-Lobato,  
§ David Duvenaud,  
* Ryan P. Adams,  
† Benjamín Recht,  
‡ Rafael Gömez-Bombarelli,  
́ Jennifer N. Wei,  
⊥ Dennis Sheberla,  
¶ Jose Miguel Hernández-Lobato,  
§ David Duvenaud

---

Supporting Information

This is an open access article published under an ACS AuthorChoice License

Read the Supporting Information online.
Variational autoencoders comprise two linked deep neural networks — The **encoder** $\Phi$ learns to project samples $x$ into a low-dim latent space $z$ — The **decoder** $\theta$ reconstructs samples $x$ from latent space vectors $z$

Trained to reconstruct its own inputs (i.e., auto-encode) the VAE performs unsupervised nonlinear dimensionality reduction.
Learn oligopeptide featurization

- The latent space is regularized to a Gaussian for mathematical convenience — the encoder infers \((\mu, \sigma)\) for each input
- A trained VAE is **generative** — decoder can hallucinate new samples from arbitrary latent space vectors sampled from the latent space
Learn oligopeptide featurization

- We train VAEs to learn latent space embeddings = essential featurizations of all $20^n$ oligopeptides for a given $\Pi$ core
- Represent oligopeptides to VAE as:
  (i) vector of Martini bead types (composition)
  (ii) bead adjacency matrix (molecular topology)
We train VAEs to learn latent space embeddings = essential featurizations of all $20^n$ oligopeptides for a given $\Pi$ core.

Represent oligopeptides to VAE as:

(i) vector of Martini bead types (composition)
(ii) bead adjacency matrix (molecular topology)
Learn oligopeptide featurization

- We train VAEs to learn latent space embeddings = essential featurizations of all $20^n$ oligopeptides for a given $\Pi$ core
- Construct and train VAEs in TensorFlow to minimize loss function error on cross-validation partitions
Regress oligopeptide fitness over latent space

- VAE latent space provides 3D featurization of oligopeptides
  - leading order description of oligopeptide composition and structure
  - embeds similar oligopeptides close together
• Run Martini CG simulations for $O(10)$ randomly selected oligopeptides
• "Fitness" is **number of inter-core contacts** in self-assembled aggregate — more core contacts $\Rightarrow$ better $p$ orbital overlap and e$^-$/h$^+$ paths
• Construct supervised learning of **Gaussian process regression** model — fitness $= f$(VAE latent space)

GPR assumes data $\{y\}$ can be represented as a sample from a multivariate Gaussian distribution over $x$

$$
\begin{align*}
\text{training data} & \quad \begin{bmatrix} y \\ y_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K & K_*^T \\ K_* & K_{**} \end{bmatrix}\right), \\
\text{predictions} & \quad \mathbb{E}[y_*] = K_*K^{-1}y, \\
& \quad \text{covariance}
\end{align*}
$$

Conditional probability of new datum $y^*$ given training $\{y\}$ follows Gaussian

$$
\begin{align*}
\mathbb{E}[y_*] &= K_*K^{-1}y \\
\text{var}(y_*) &= K_{**} - K_*K^{-1}K_*^T
\end{align*}
$$

C. E. Rasmussen & C. K. I. Williams, Gaussian Processes for Machine Learning, the MIT Press, 2006
• Use GPR to inform **optimal traversal of sequence space** in virtuous cycle
  — GPR strengthens with samples ⇔ better guidance from GPR

• **Active learning** paradigm to select new oligopeptides to simulate from GPR
  — **exploit**: best candidate picked by GPR
  — **explore**: sample candidates where GPR has maximum uncertainty

C. E. Rasmussen & C. K. I. Williams, Gaussian Processes for Machine Learning, the MIT Press, 2006
Optimally sample sequence space

- **Expected improvement** (EI) acquisition function balances exploit / explore
- Select next oligopeptide to simulate as that which maximizes EI

---

Putting it all together...

**GOAL:** Computationally identify optimal assembling DXXX-OPV3-XXXD oligopeptides

---

1. **Unsupervised VAE (re)training and latent space embedding**
2. **Supervised GPR (re)training over VAE latent space**
3. **Active learning of next best oligopeptides to simulate**

---

0. **CG MD simulations**
   - measure core-core contacts
   - $R1: O(10); R2+: 3-4$

---

**peptide family to screen**

![DXXX-OPV3-XXXD oligopeptide](image)
Stopping criteria

- Determine convergences by monitoring GPR model performance
- Terminate when model predictions stop changing with additional samples
- Model converges after 24 rounds, 186 chemistries, 558 μs of simulation

**Cross validated R2 score on observed data**

**Bhattacharya distance $D_B$ between GPR posteriors**

(change in posterior with added samples)
Stopping criteria

- Determine convergences by monitoring GPR model performance
- Terminate when model predictions stop changing with additional samples
- Model converges after 24 rounds, 186 chemistries, 558 μs of simulation

Cross validated R2 score on observed data

Bhattacharya distance $D_B$ between GPR posteriors (change in posterior with added samples)

ML model (GPR) capable of identifying optimal oligopeptides after active learning sampling of only 2.3% of accessible sequence space
Optimal candidates

- Top 10 candidates identified and validated throughout 25 rounds of active learning protocol

\[
\text{average degree of core-core contacts interaction graph} \\
(\text{higher is better})
\]

<table>
<thead>
<tr>
<th>Chemistry (DXXX)</th>
<th>(\langle \kappa_i \rangle)</th>
<th>Round #</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEAA</td>
<td>6.067</td>
<td>1</td>
</tr>
<tr>
<td>DDAI</td>
<td>6.034</td>
<td>0</td>
</tr>
<tr>
<td>DIAM</td>
<td>6.008</td>
<td>17</td>
</tr>
<tr>
<td>DVAA</td>
<td>5.950</td>
<td>9</td>
</tr>
<tr>
<td>DAAV</td>
<td>5.925</td>
<td>19</td>
</tr>
<tr>
<td>DGLG</td>
<td>5.922</td>
<td>20</td>
</tr>
<tr>
<td>DAEA</td>
<td>5.920</td>
<td>25</td>
</tr>
<tr>
<td>DAGI</td>
<td>5.900</td>
<td>21</td>
</tr>
<tr>
<td>DGIG</td>
<td>5.883</td>
<td>25</td>
</tr>
<tr>
<td>DEAL</td>
<td>5.880</td>
<td>23</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>DGAG</td>
<td>5.540</td>
<td>0</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>DFAG</td>
<td>4.984</td>
<td>0</td>
</tr>
</tbody>
</table>
Three classes of oligopeptides

**TRAPPERS**

DFAG

**ASSEMBLERS**

DGAG

**FRAGMENTORS**

DWWW

5 clusters formed
3 clusters formed
14 clusters formed
Lo-dim viz of assembly pathways

- Diffusion map dimensionality reduction over interaction graphs reveals mechanistic partitioning of the three classes identified by active learning:
  - Assemblers
  - Trappers
  - Fragmentors
  - Monomers
Lo-dim viz of assembly pathways

- Diffusion map dimensionality reduction over interaction graphs reveals mechanistic partitioning of the three classes identified by active learning:
  - Assemblers
  - Trappers
  - Fragmentors
  - Monomers