



*Inria*



European Research Council  
Established by the European Commission

# Machine learning for coarse-graining molecular systems

**Gabriel STOLTZ**

(CERMICS, Ecole des Ponts & MATHERIALS team, Inria Paris)

*With Zineb Belkacemi (Sanofi & ENPC), Tony Lelièvre (ENPC/Inria)  
and Paraskevi Gkeka (Sanofi)*

MAC-MIGS One-Day Meeting on Mathematical Topics in Machine Learning

- **Molecular systems and basics of statistical physics**
  - **Reaction coordinates** and free energy
  - A (short/biased) review of machine learning approaches for RC
- **Free-energy biasing and iterative learning with autoencoders<sup>1</sup>**
  - **Autoencoders** and their training
  - General presentation of the iterative algorithm
  - Illustration/sanity checks on toy examples
- **Applications to systems of interest** (alanine dipeptide, HSP90)

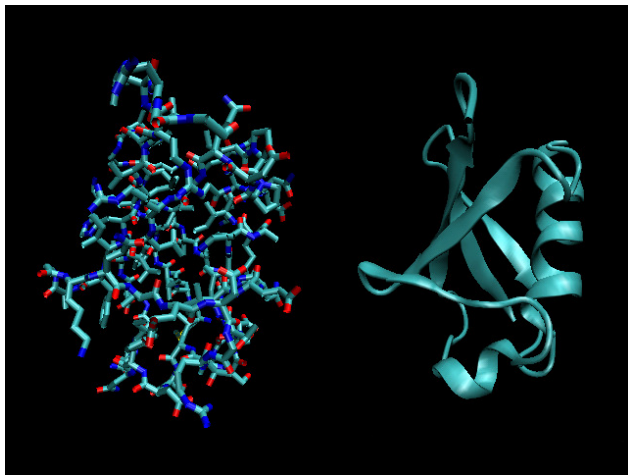
---

<sup>1</sup>Z. Belkacemi, P. Gkeka, T. Lelièvre, G. Stoltz, arXiv preprint **2104.11061**

# Molecular description of systems

# Statistical physics (1)

What is the **structure** of the protein? What are its **typical conformations**, and what are the **transition pathways** from one conformation to another?



## Statistical physics (2)

- **Microstate** of a classical system of  $N$  particles:

$$(q, p) = (q_1, \dots, q_N, p_1, \dots, p_N) \in \mathcal{E} = (a\mathbb{T})^{3N} \times \mathbb{R}^{3N}$$

**Positions**  $q$  (configuration), **momenta**  $p$  (to be thought of as  $M\dot{q}$ )

- **Hamiltonian**  $H(q, p) = V(q) + \sum_{i=1}^N \frac{p_i^2}{2m_i}$  (physics is in  $V$ )

**Macrostate:** Boltzmann–Gibbs probability measure (NVT)

$$\mu(dq dp) = Z_{\text{NVT}}^{-1} e^{-\beta H(q,p)} dq dp, \quad \beta = \frac{1}{k_B T}$$

- Typical evolution equations: Langevin dynamics (friction  $\gamma > 0$ )

$$\begin{cases} dq_t = M^{-1} p_t dt \\ dp_t = -\nabla V(q_t) dt - \gamma M^{-1} p_t dt + \sqrt{2\gamma\beta^{-1}} dW_t \end{cases}$$

# Reaction coordinates (RC) / collective variables (CV)

- **Reaction coordinate**  $\xi : \mathbb{R}^D \rightarrow \mathbb{R}^d$  with  $d \ll D$
- Ideally:  $\xi(q_t)$  captures the **slow** part of the dynamics
- **Free energy** computed on  $\Sigma(z) = \{q \in (a\mathbb{T})^D \mid \xi(q) = z\}$  (foliation)

$$F(z) = -\frac{1}{\beta} \ln \left( \int_{\Sigma(z)} e^{-\beta V(q)} \delta_{\xi(q)-z}(dq) \right)$$

- Various methods: TI, FEP, ABF, metadynamics, etc<sup>2</sup>

---

<sup>2</sup>Lelièvre/Rousset/Stoltz, *Free Energy Computations: A Mathematical Perspective* (Imperial College Press, 2010)

# Some representative approaches for finding RC/CV (1)

- Chemical/physical **intuition** (distances, angles, RMSDs, coordination numbers, etc)
- **Short list of data-oriented approaches** (depending on the data at hand...)
  - [supervised learning] separate metastable states
  - [unsupervised] distinguish linear models (PCA) and nonlinear ones (e.g. based on autoencoders such as **MESA**<sup>3</sup>)
  - [dynamics] operator based approaches (VAC, EDMD, diffusion maps, MSM; incl. tICA and VAMPNets)

(Huge literature! I am not quoting precise references here because the list would be too long)

- Other classifications<sup>4,5</sup> possible, e.g. **slow vs. high variance CV**

---

<sup>3</sup>W. Chen and A.L. Ferguson, *J. Comput. Chem.* 2018; W. Chen, A.R. Tan, and A.L. Ferguson, *J. Chem. Phys.* 2018

<sup>4</sup>P. Gkeka et al., *J. Chem. Theory Comput.* 2020

<sup>5</sup>A. Gliemlo, B. Husic, A. Rodriguez, C. Clementi, F. Noé, A. Laio, *Chem. Rev.* 2021

# Some representative approaches for finding RC/CV (2)

## Methods for Choosing Collective variables

### High-variance CVs

Principal Components  
Analysis (PCA)

Locally Linear  
Embedding (LLE)

Independent Component  
Analysis (ICA)

Laplacian and Hessian  
eigenmaps

Local tangent space  
alignment

Kernel PCA

Nonlinear PCA

Isomap

Diffusion maps

Multidimensional scaling

Semidefinite embedding/  
Maximum variance unfolding

### Available tools for CV identification

Diffusion-Map-directed MD  
(DM-d-MD)

Intrinsic Map Dynamics  
(iMapD)

Smooth and nonlinear datadriven CVs  
(SandCV)

Molecular Enhanced Sampling  
with Autoencoders (MESA)

Rewighted Autoencoded Variations  
Bayes for Enhanced Sampling (RAVE)

REinforcement Learning based on  
Adaptive samPLing (REAP)

### Slow CVs

Variational Approach to Conformational dynamics (VAC)

(extended) Dynamical Mode Decomposition ((E)DMD)

Kernel TICA

Markov State Models (MSM)

Time-lagged autoencoders (TAEs)

Time-lagged Independent Component  
Analysis (TICA)

Deep Canonical Correlation Analysis  
(DCCA)

Variational Dynamics Encoders  
(VDEs)

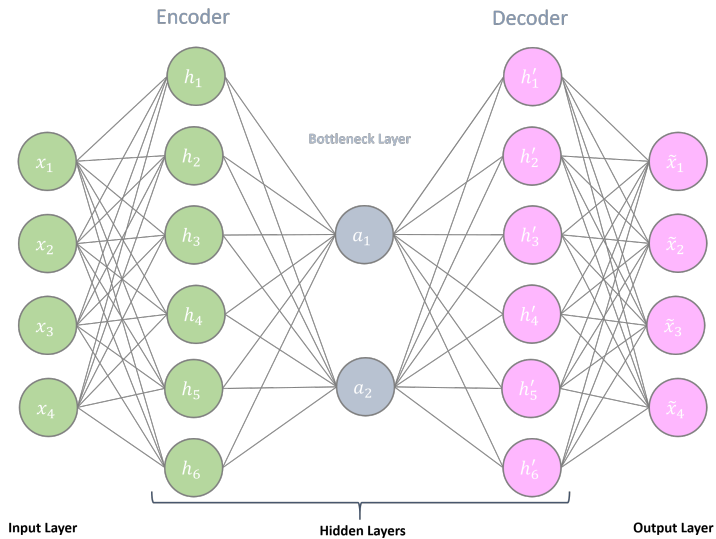
Variational Approach for Markov Processes nets (VAMPnets)

State-free Reversible VAMPnets (SRV)



# Free-energy biasing and iterative learning with autoencoders

# Autoencoders (1)



## Autoencoders (2)

- Data space  $\mathcal{X} \subseteq \mathbb{R}^D$ , bottleneck space  $\mathcal{A} \subseteq \mathbb{R}^d$  with  $d < D$

$$f(x) = f_{\text{dec}}(f_{\text{enc}}(x))$$

where  $f_{\text{enc}} : \mathcal{X} \rightarrow \mathcal{A}$  and  $f_{\text{dec}} : \mathcal{A} \rightarrow \mathcal{X}$

Reaction coordinate = encoder part

$$\xi = f_{\text{enc}}$$

- Fully connected neural network, symmetrical structure,  $2L$  layers
- Parameters  $\mathbf{p} = \{p_k\}_{k=1, \dots, K}$  (bias vectors  $b_\ell$  and weights matrices  $W_\ell$ )

$$f_{\mathbf{p}}(x) = g_{2L} [b_{2L} + W_{2L} \dots g_1 (b_1 + W_1 x)],$$

with activation functions  $g_\ell$  (examples:  $\tanh(x)$ ,  $\max(0, x)$ , etc)

# Training autoencoders

- **Theoretically**: minimization problem in  $\mathcal{P} \subset \mathbb{R}^K$

$$\mathbf{p}_\mu \in \operatorname{argmin}_{\mathbf{p} \in \mathcal{P}} \mathcal{L}(\mu, \mathbf{p}),$$

with **cost function**

$$\mathcal{L}(\mu, \mathbf{p}) = \mathbb{E}_\mu(\|X - f_{\mathbf{p}}(X)\|^2) = \int_{\mathcal{X}} \|x - f_{\mathbf{p}}(x)\|^2 \mu(dx)$$

- In practice, access only to a sample: **minimization of empirical cost**

$$\mathcal{L}(\hat{\mu}, \mathbf{p}) = \frac{1}{N} \sum_{i=1}^N \|x^i - f_{\mathbf{p}}(x^i)\|^2, \quad \hat{\mu} = \frac{1}{N} \sum_{i=1}^N \delta_{x^i}$$

- Typical choices: canonical measure  $\mu$ , data points  $x^i$  postprocessed from positions  $q$  (alignment to reference structure, centering, reduction to backbone carbon atoms, etc)

# Training on modified target measures

- Interesting systems are **metastable** (no spontaneous exploration of phase space)  
Sample according to a biased distribution  $\tilde{\mu}$  (**importance sampling**)
- Need for **reweighting** to learn the correct encoding!

$$w(x) = \frac{\mu(x)}{\tilde{\mu}(x)}$$

- **Minimization problem:** theoretical cost function

$$\mathcal{L}(\mu, \mathbf{p}) = \int_{\mathcal{X}} \|x - f_{\mathbf{p}}(x)\|^2 w(x) \tilde{\mu}(dx),$$

actual cost function

$$\mathcal{L}(\hat{\mu}_{\text{wght}}, \mathbf{p}) = \sum_{i=1}^N \hat{w}_i \|x^i - f_{\mathbf{p}}(x^i)\|^2, \quad \hat{w}_i = \frac{\mu(x^i)/\tilde{\mu}(x^i)}{\sum_{j=1}^N \mu(x^j)/\tilde{\mu}(x^j)}$$

- Only requires the knowledge of  $\mu$  and  $\tilde{\mu}$  up to a multiplicative constant.

# How training is actually performed...

- **Gradient descent with minibatching:** randomly reshuffle data points,

$$\mathbf{p}_r = \mathbf{p}_{r-1} - \eta \nabla_{\mathbf{p}} \mathcal{L}_r(\mathbf{p}_{r-1}), \quad \mathcal{L}_r(p) = \frac{1}{m} \sum_{i=r m+1}^{(r+1)m} \|x^i - f_{\mathbf{p}}(x^i)\|^2$$

One epoch =  $\lceil N/m \rceil$  gradient steps (in order to visit all the data)

- **Actual procedure:**

- Use keras module in python
- Computation of gradient performed with backpropagation
- Optimization in fact performed with Adam algorithm  
(weights summing to 1 to use default optimization parameters)
- “Early stopping” (stop when validation loss no longer improves)

- Many local minima...

# Proof of concept (1)

- **Gaussian distributions**  $\mu_i = \mathcal{N}(0, \Sigma_i)$  with

$$\Sigma_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0.01 \end{pmatrix}, \quad \Sigma_2 = \begin{pmatrix} 0.01 & 0 \\ 0 & 1 \end{pmatrix}$$

Datasets  $\mathcal{D}_i$  of  $N = 10^6$  i.i.d. points

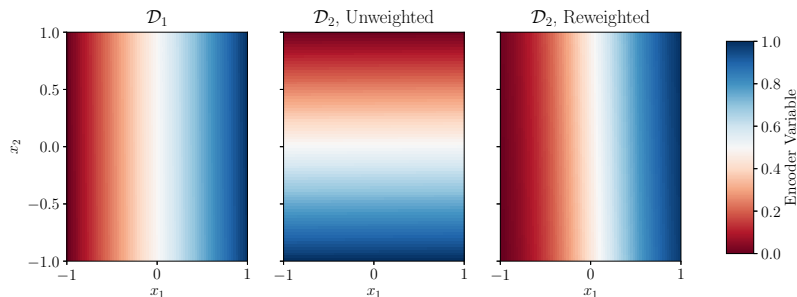
- Autoencoders with 2 layers of resp. 1 and 2 nodes, linear activation functions ( $\simeq$  PCA)

- **Training on:**

- $\mathcal{D}_1$
- $\mathcal{D}_2$
- $\mathcal{D}_2$  with reweighting  $\hat{w}_i \propto \mu_1/\mu_2$

# Proof of concept (2)

## Heat maps of $f_{\text{enc}}$



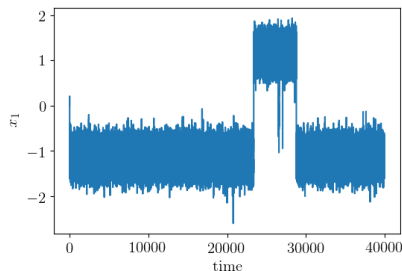
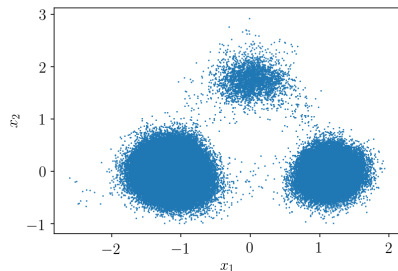
Third encoder very similar to the first: projection on  $x_1$ .  
Second encoder projects on a direction close to  $x_2$ .



# Proof of concept with free energy biasing (1)

## Two dimensional potential (“entropic switch”)<sup>6</sup>

$$V(x_1, x_2) = 3e^{-x_1^2} \left( e^{-(x_2-1/3)^2} - e^{-(x_2-5/3)^2} \right) - 5e^{-x_2^2} \left( e^{-(x_1-1)^2} + e^{-(x_1+1)^2} \right) + 0.2x_1^4 + 0.2(x_2 - 1/3)^4$$



Trajectory from  $q^{j+1} = q^j - \nabla V(q^j)\Delta t + \sqrt{2\beta^{-1}\Delta t}G^j$  for  $\beta = 4$  and  $\Delta t = 10^{-3} \rightarrow$  **metastability** in the  $x_1$  direction

<sup>6</sup>S. Park, M.K. Sener, D. Lu, and K. Schulten (2003)

## Proof of concept with free energy biasing (2)

- **Free energy biasing:** distributions  $Z_i^{-1} \exp(-\beta [V(q) - F_i(\xi_i(q))])$

$$F_1(x_1) = -\frac{1}{\beta} \ln \left( \int_{\mathbb{R}} e^{-\beta V(x_1, x_2)} dx_2 \right), \quad F_2(x_2) = -\beta^{-1} \ln \left( \int_{\mathbb{R}} \dots dx_1 \right)$$

**Three datasets:** unbiased trajectory, trajectories biased using  $F_1$  and  $F_2$

(free energy biased trajectories are shorter but same number of data points  $N = 10^6$ )

- Autoencoders: 2-1-2 topology, activation functions  $\tanh$  (so that RC is in  $[-1, 1]$ ) then identity

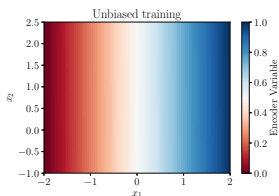
- **Five training scenarios:**

- training on long unbiased trajectory (reference RC)
- $\xi_1$ -biased trajectory, **with** or **without** reweighting
- $\xi_2$ -biased trajectory, **with** or **without** reweighting

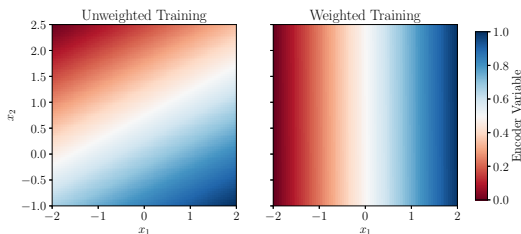
# Proof of concept with free energy biasing (3)

Normalize to compare

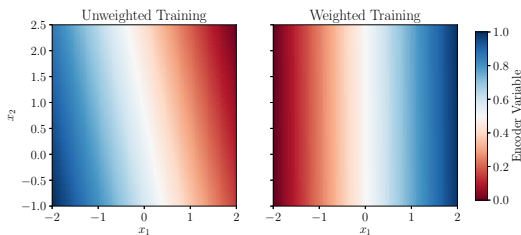
$$\xi_{\text{SAE}}^{\text{norm}}(x) = \frac{\xi_{\text{SAE}}(x) - \xi_{\text{SAE}}^{\text{min}}}{\xi_{\text{SAE}}^{\text{max}} - \xi_{\text{SAE}}^{\text{min}}}$$



Reference RC  
(distinguishes well the 3 wells)



$x_1$ -biased trajectory



$x_2$ -biased trajectory

# Full iterative algorithm (Free Energy Biasing and Iterative Learning with AutoEncoders)

**Input:** Initial condition  $q_0$ , autoencoder topology and initialization parameters  $A_{\text{init}}$ , number of samples  $N$ , simulation procedure  $S$  and **adaptive biasing procedure**  $S_{\text{AB}}$ , maximum number of iterations  $I_{\text{max}}$ , minimum convergence score  $s_{\text{min}}$

## Initialization

Sample  $\text{traj}_0 \leftarrow S(q_0, N)$

Initialize autoencoder  $\text{AE}_0 \leftarrow A_{\text{init}}$

Train  $\text{AE}_0$  on  $\text{traj}_0$  with weights  $(\hat{w}_0, \dots, \hat{w}_N) = (1, \dots, 1)$

Extract the encoder function  $\xi_0 : x \mapsto \xi_0(x)$

## Iterative update of the reaction coordinate

Set  $i \leftarrow 0, s \leftarrow 0$

**While**  $i < I_{\text{max}}$  and  $s < s_{\text{min}}$

Set  $i \leftarrow i + 1$

Sample  $\text{traj}_i, F_i \leftarrow S_{\text{AB}}(q_0, N, \xi_{i-1})$

Compute weights  $\hat{w}_j \propto e^{-\beta F_i(\xi_{i-1}(x^j))}$

Initialize autoencoder  $\text{AE}_i \leftarrow A_{\text{init}}$

Train  $\text{AE}_i$  on  $\text{traj}_i$  with sample weights  $\hat{w}_j$

Extract the encoder function  $\xi_i : x \mapsto \xi_i(x)$

Set  $s \leftarrow \text{regscore}(\xi_{i-1}, \xi_i)$

Set  $\xi_{\text{final}} \leftarrow \xi_i$

Threshold  $s_{\text{min}}$  to be determined

in our case: extended ABF

Convergence metric to be made precise

## Production of output:

Sample  $\text{traj}_{\text{final}}, F_{\text{final}} \leftarrow S_{\text{AB}}(q_0, N_{\text{final}}, \xi_{\text{final}})$  with  $N_{\text{final}}$  large enough to ensure PMF convergence

# Discussion on the convergence criterion

- Check convergence of CV?

Quantify  $\xi_i \approx \Phi(\xi_{i-1})$  for some **monotonic function  $\Phi$**

- Approach: approximate  $\Phi$  by a linear model  $\rightarrow$  **linear regression**

- **Regression score** between  $\xi$  and  $\xi'$

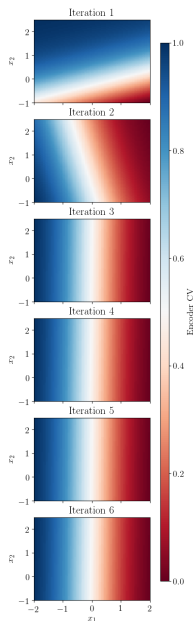
- Two sets of values of RC  $(\xi(q^1), \dots, \xi(q^N))$  and  $(\xi'(q^1), \dots, \xi'(q^N))$
- Match them with a linear model  $M(z) = Wz + b$

- Coefficient of determination  $R^2 = 1 - \frac{\sum_{i=1}^N \|\xi'(q^i) - M(\xi(q^i))\|^2}{\sum_{i=1}^N \|\xi'(q^i) - \bar{\xi}'\|^2}$

- Maximization of  $R^2$  w.r.t.  $W, b$  provides  $\text{regscore}(\xi', \xi)$

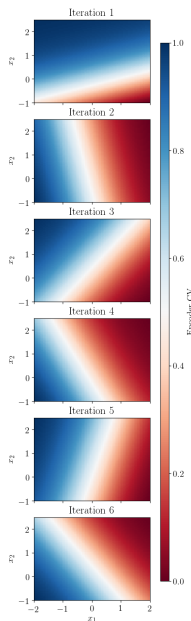
- **Value of  $s_{\min}$**  computed using some bootstrap procedure

# The iterative algorithm on the toy 2D example



**Left:** with reweighting  
Convergence to RC  $\simeq x_1$

**Right:** without reweighting  
No convergence  
(cycles between two RCs)



# Applications to systems of interest

# Alanine dipeptide

- **Molecular dynamics:**

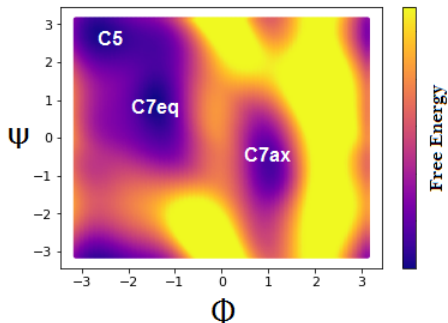
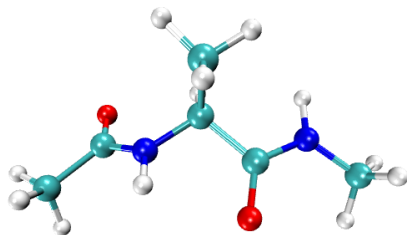
openmm with openmm-plumed to link it with plumed  
colvar module for eABF and computation of free energies  
timestep 1 fs, friction  $\gamma = 1 \text{ ps}^{-1}$  in Langevin dynamics

- **Machine learning:**

keras for autoencoder training

input = carbon backbone (realignment to reference structure and centering)

neural network: topology 24-40-2-40-24, tanh activation functions

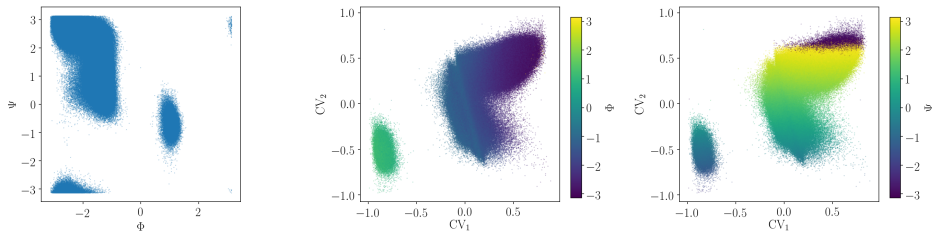




# Ground truth computation

Long trajectory ( $1.5 \mu\text{s}$ ),  $N = 10^6$  (frames saved every 1.5 ps)

RC close to dihedral angles  $\Phi, \Psi$

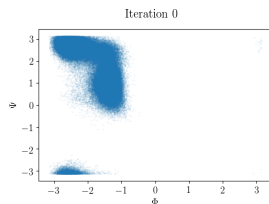


Quantify  $s_{\min} = 0.99$  for  $N = 10^5$  using a bootstrapping procedure

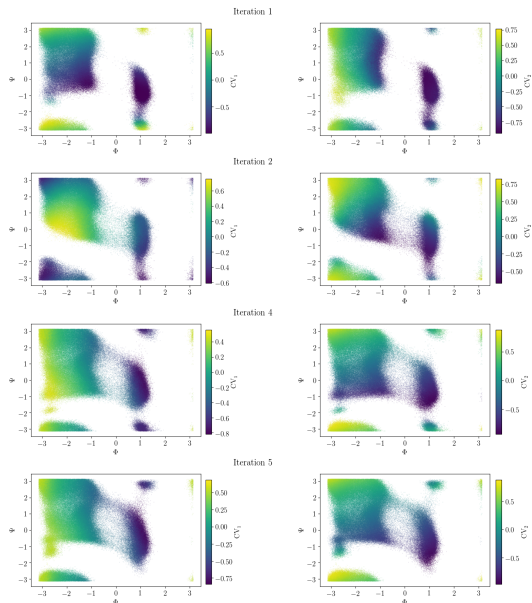
**For the iterative algorithm:** 10 ns per iteration

(compromise between times not too short to allow for convergence of the free energy, and not too large in order to alleviate the computation cost)

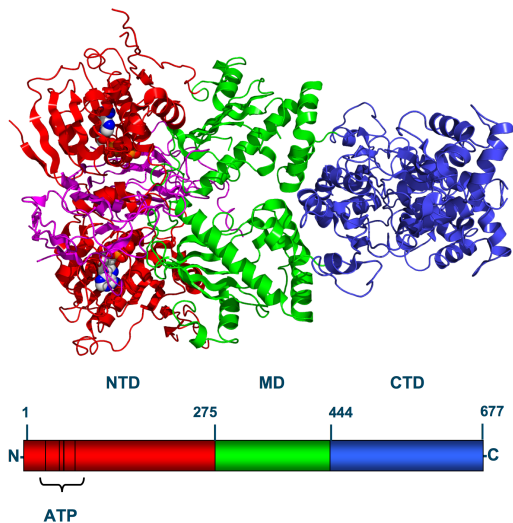
# Results for the iterative algorithm



iter.	regscore	$(\Phi, \Psi)$
0	—	0.922
1	0.872	0.892
2	0.868	0.853
3	0.922	0.973
4	0.999	0.972
5	0.999	0.970
6	0.999	0.971
7	0.999	0.967
8	0.998	0.966
9	0.999	0.968



# HSP90 (work in progress...)

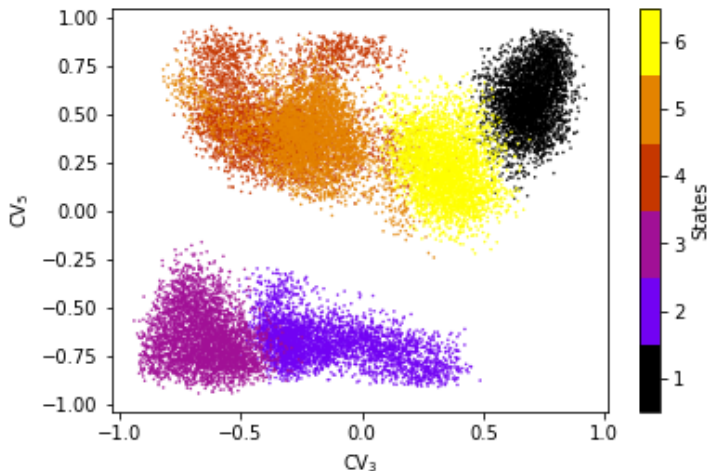


Chaperone protein assisting other proteins to fold properly and stabilizing them against stress, including proteins required for **tumor growth**

→ look for **inhibitors** (e.g. targeting binding region of ATP; focus only on the N-terminus domain )

(picture from [https://en.wikipedia.org/wiki/File:Hsp90\\_schematic\\_2cg9.png](https://en.wikipedia.org/wiki/File:Hsp90_schematic_2cg9.png))

# HSP90 (work in progress...)



6 conformational states, data from  $10 \times 20$  ns trajectories, input features = 621 C carbons, AE topology 621-100-5-100-621