





European Research Council

Established by the European Commission

# Constructing collective variables using Machine Learning and free energy biased simulations

**Tony LELIEVRE, Thomas PIGEON and Gabriel STOLTZ**

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

Winter school GdR MatBio, 8-12 November 2021, CIRM

# Outline

- **A (short/biased) review of machine learning approaches for CV**

- **Free-energy biasing and iterative learning with autoencoders**[1]
  - Autoencoders: definition, training, interpretation
  - Extended adaptive biasing force method
  - General presentation of the iterative algorithm
  - Illustration/sanity checks on toy examples

- **Applications to systems of interest** (alanine dipeptide, chignolin, HSP90)

[1] Z. Belkacemi, P. Gkeka, T. Lelièvre, G. Stoltz, arXiv preprint **2104.11061**

# (A biased perspective on some) References

- **ML reviews in MD**
  - A. Gliemlo, B. Husic, A. Rodriguez, C. Clementi, F. Noé, A. Laio, *Chem. Rev.* **121**(16), 9722-9758 (2021)
  - M. Chen, *Eur. Phys. J. B* **94**, 211 (2021)
  - P. Gkeka *et al.*, *J. Chem. Theory Comput.* **16**(8), 4757-4775 (2020)
  - F. Noé, A. Tkatchenko, K.-R. Müller, C. Clementi, *Annu. Rev. Phys. Chem.* **71**, 361-390 (2020)
  - A.L. Ferguson, *J. Phys.: Condens. Matter* **30**, 04300 (2018)

- **More general ML references**
  - P. Mehta, M. Bukov, C.-H. Wang, A.G.R.Day, C. Richardson, C.K.Fisher, D.J. Schwab, A high-bias, low-variance introduction to Machine Learning for physicists, *Physics Reports* **810**, 1-124 (2019)
  - I. Goodfellow, Y. Bengio, A. Courville *Deep Learning* (MIT Press, 2016) `http://www.deeplearningbook.org`

# ML approaches for finding CV

# Some representative approaches for finding 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/static] distinguish linear models (PCA) and nonlinear ones (e.g. based on autoencoders such as MESA[2])
  - [unsupervised/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[3,4] possible, e.g. **slow vs. high variance CV**

---

[2] W. Chen and A.L. Ferguson, *J. Comput. Chem.* 2018; W. Chen, A.R. Tan, and A.L. Ferguson, *J. Chem. Phys.* 2018

[3] P. Gkeka et al., *J. Chem. Theory Comput.* (2020)

[4] A. Gliemlo *et al.*, *Annu. Rev. Phys. Chem.* (2021)

# Some representative approaches for finding 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)

Reweighted 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)

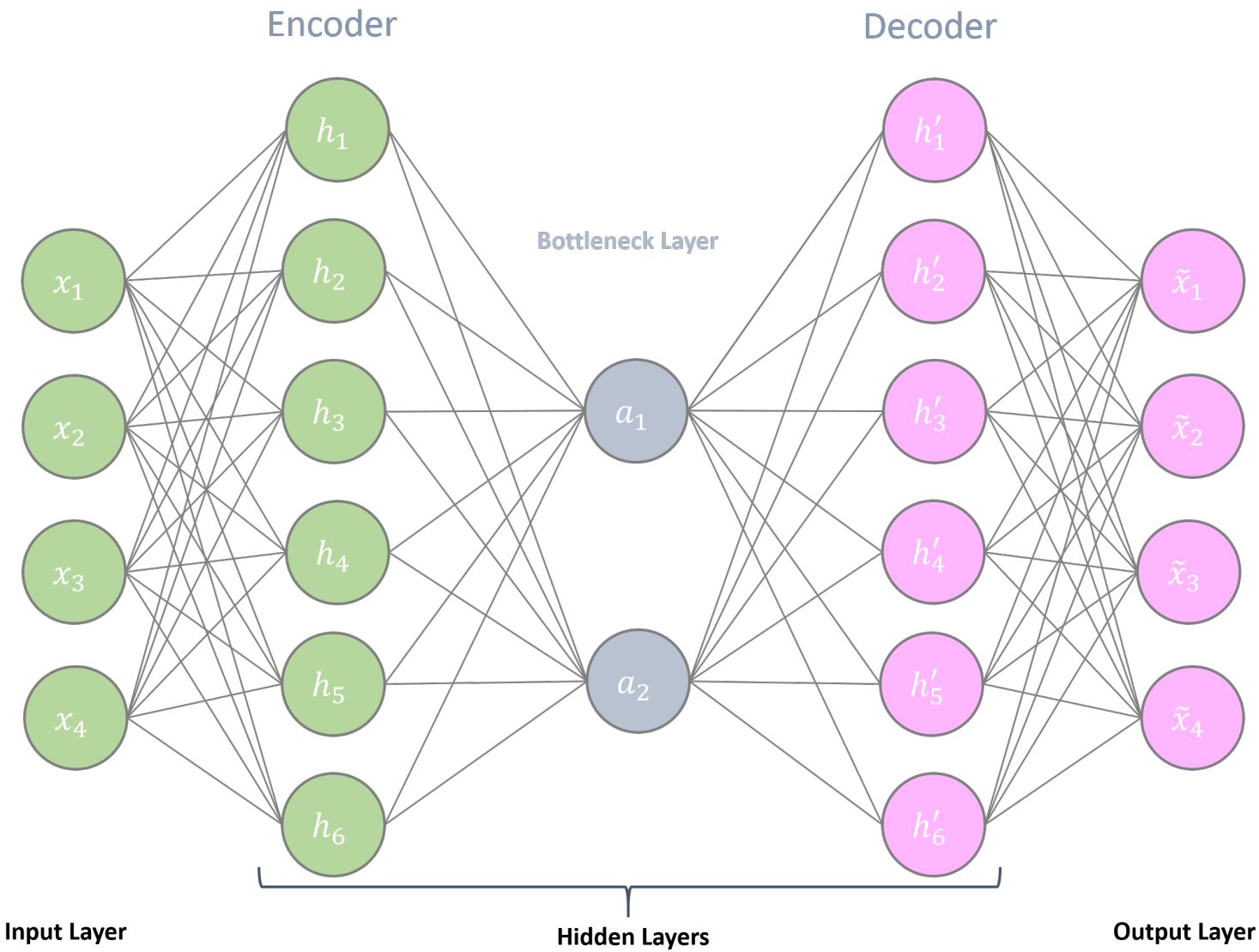
Variational Dynamics Encoders (VDEs)

Variational Approach for Markov Processes nets (VAMPnets)

State-free Reversible VAMPnets (SRV)

# CV construction with autoencoders

# Autoencoders (1)



# Autoencoders (2)

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

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

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

**Collective variable = 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 $W_\ell$)

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

with activation functions $g_\ell$

(examples: $\tanh(x)$, ReLU $\max(0, x)$, sigmoid $\sigma(x) = 1/(1 + \mathrm{e}^{-x})$, 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**

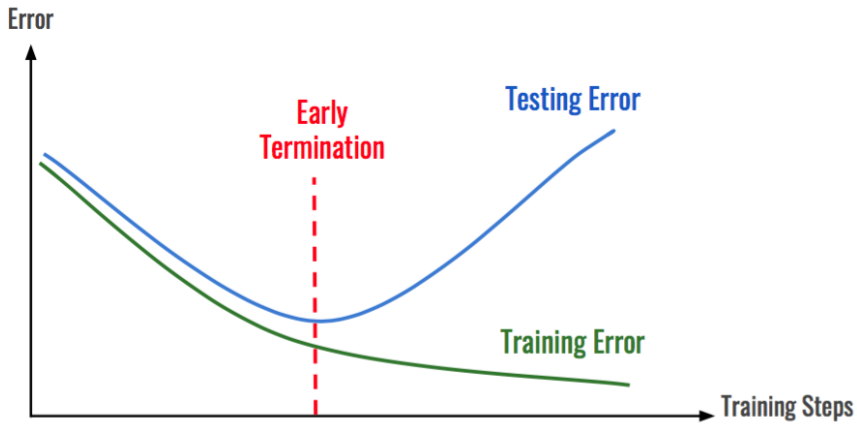
$$\mathcal{L}(\hat{\mu}, \mathbf{p}) = \frac{1}{N}\sum_{i=1}^{N} \|x^i - f_{\mathbf{p}}(x^i)\|^2, \qquad \hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} \delta_{x^i}$$

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

# Some elements on training neural networks (1/2)

- Many local minima...

- **Actual procedure:**
  - Separate data set into training/validation: optimize on training set
  - "Early stopping": stop when validation loss no longer improves[5]




  - Computation of gradient performed with backpropagation
  - Choice of optimization method[6], here Adam

[5] See Section 7.8 in [Goodfellow/Bengio/Courville]
[6] See Chapter 8 in [Goodfellow/Bengio/Courville]

# Some elements on training neural networks (2/2)

- **Stochastic gradient descent (SGD)**: mini-batching (randomness)

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

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

Stochastic approximation of the true gradient[7]

$$\nabla_{\mathbf{p}} \mathcal{L}_r(\mathbf{p}) = \nabla_{\mathbf{p}} \mathcal{L}(\mathbf{p}) + \sqrt{\varepsilon(r)} \Sigma(\mathbf{p})^{1/2} Z, \quad Z \approx \mathcal{N}(0, \mathrm{Id}), \quad \varepsilon(r) \approx \frac{N^2}{m}$$

- **Adam (Adaptive Moment)**[8]: uses a moving average of gradients, normalized by a moving average of gradient mean squares (element-wise) $\rightarrow$ more robust to anisotropy

... but current debates for DeepNN about why SGD better for generalization although Adam better for optimization

[7] See e.g. Section 2.2 in I. Sekkat, G. Stoltz, *arXiv preprint* **2105.10347**

[8] Kingma/Ba, 2015 (arXiv preprint **1412.6980**); Balles/Hennig, *ICML* (2018)

# A variance interpretation of autoencoders (Wei Zhang)

- Total variance $\boxed{\mathrm{Var}(X) = \mathrm{Var}\left[\mathbb{E}(X|f_{\mathrm{enc}}(X))\right] + \mathbb{E}\left[\mathrm{Var}(X|f_{\mathrm{enc}}(X))\right]}$

Proof: Use $\mathbb{E}\left[\mathbb{E}(h(X)|f_{\mathrm{enc}}(X))\right] = \mathbb{E}(h(X))$ to write $\mathrm{Var}\left[\mathbb{E}(X|f_{\mathrm{enc}}(X))\right] = \mathbb{E}\left[\mathbb{E}(X|f_{\mathrm{enc}}(X))^2\right] - \mathbb{E}(X)^2$ and $\mathbb{E}\left[\mathrm{Var}(X|f_{\mathrm{enc}}(X))\right] = \mathbb{E}(X^2) - \mathbb{E}\left[\mathbb{E}(X|f_{\mathrm{enc}}(X))^2\right]$, the sum of these terms being $\mathbb{E}(X^2) - \mathbb{E}(X)^2$.

- Training w.r.t. decoder part performed analytically in principle as

$$\min_{f_{\mathrm{enc}}} \left[ \min_{f_{\mathrm{dec}}} \int_{\mathcal{X}} |x - f_{\mathrm{dec}}(f_{\mathrm{enc}}(x))|^2 \, \mu(dx) \right] = \min_{f_{\mathrm{enc}}} \ell(f_{\mathrm{enc}})$$

with
$$\ell(f_{\mathrm{enc}}) = \mathbb{E}\left[ \Big( X - \mathbb{E}(X|f_{\mathrm{enc}}(X)) \Big)^2 \right] = \mathbb{E}\left[\mathrm{Var}(X|f_{\mathrm{enc}}(X))\right]$$
$$= \mathrm{Var}(X) - \mathrm{Var}\left[\mathbb{E}(X|f_{\mathrm{enc}}(X))\right]$$

- Minimizing $\ell(f_{\mathrm{enc}})$ equivalent to maximizing $\mathrm{Var}\left[\mathbb{E}(X|f_{\mathrm{enc}}(X))\right]$
= minimizing intraclass dispersion vs. maximizing interclass dispersion
= small spread of data points for $f_{\mathrm{enc}}$ given around the mean vs. the mean values associated with $f_{\mathrm{enc}}$ given should be as spread out as possible

# Free energy biasing for complex CV

# Extended systems

- Computing $\nabla\xi$ already difficult, higher order derivatives is worse
- **Extended system** strategy : $V_{\mathrm{ext}}(q,\lambda) = V(q) + \frac{\kappa}{2}\big(\xi(q)-\lambda\big)^2$
- Free energy for the (simple) collective variable $\xi_{\mathrm{ext}}(q,\lambda) = \lambda$

$$
\begin{aligned}
F_\kappa(z) &= -\frac{1}{\beta}\ln\int_{\mathcal{D}} \mathrm{e}^{-\beta V_{\mathrm{ext}}(q,z)}\,dq + C \\
&= -\frac{1}{\beta}\ln\int\left(\int_{\Sigma(\zeta)} \mathrm{e}^{-\beta V(q)}\delta_{\xi(q)-\zeta}(dq)\right)\mathrm{e}^{-\beta\kappa(\zeta-z)^2/2}\,d\zeta + C \\
&= -\frac{1}{\beta}\ln\int \mathrm{e}^{-\beta F(\zeta)}\chi_\kappa(z-\zeta)\,d\zeta + \widetilde{C}, \qquad \chi_\kappa(s) = \left(\frac{\beta\kappa}{2\pi}\right)^{d/2}\mathrm{e}^{-\beta\kappa s^2/2} \\
&\xrightarrow[\kappa\to+\infty]{} F(z)
\end{aligned}
$$

Calls for taking $\kappa$ large

# Extended ABF

**Extended overdamped Langevin dynamics** ($\kappa$ limits $\Delta t$...)

$$\begin{cases} dq_t = \Big[ -\nabla V(q_t) + \kappa(\xi(q_t) - \lambda_t)\nabla\xi(q_t) \Big] dt + \sqrt{2\beta^{-1}} dW_t^q \\ d\lambda_t = -[\lambda_t - \xi(q_t)]\, dt + \sqrt{2\beta^{-1}}\, dW_t^\lambda \end{cases}$$

## Extended ABF overdamped Langevin dynamics

$$\begin{cases} dq_t = \Big[ -\nabla V(q_t) + \kappa(\xi(q_t) - \lambda_t)\nabla\xi(q_t) \Big] dt + \sqrt{2\beta^{-1}} dW_t^q \\ d\lambda_t = [\xi(q_t) - \mathbb{E}(\xi(q_t) \,|\, \lambda_t)]\, dt + \sqrt{2\beta^{-1}}\, dW_t^\lambda \end{cases}$$

In practice, $\mathbb{E}(\xi(q_t) \,|\, \lambda_t)$ is estimated by $\Xi_t(\lambda_t)$ with

$$\Xi_t(\Lambda) = \frac{\displaystyle\int_0^t \delta_\varepsilon(\lambda_s - \Lambda)\xi(q_s)\, ds}{\max\left(\eta, \displaystyle\int_0^t \delta_\varepsilon(\lambda_s - \Lambda)\, ds\right)}$$

# Unbiased estimate of the free energy in eABF

- Stationarity: configurations distributed according to $\mathrm{e}^{-\beta(V_{\mathrm{ext}}(q,\lambda)-F_\kappa(\lambda))}$

$$\rho(z,\lambda) = Z_\kappa^{-1} \exp\left(-\beta\left[F(z) + \frac{\kappa}{2}(z-\lambda)^2 - F_\kappa(\lambda)\right]\right)$$

- **Unbiased estimator of the mean force (CZAR)**[9]

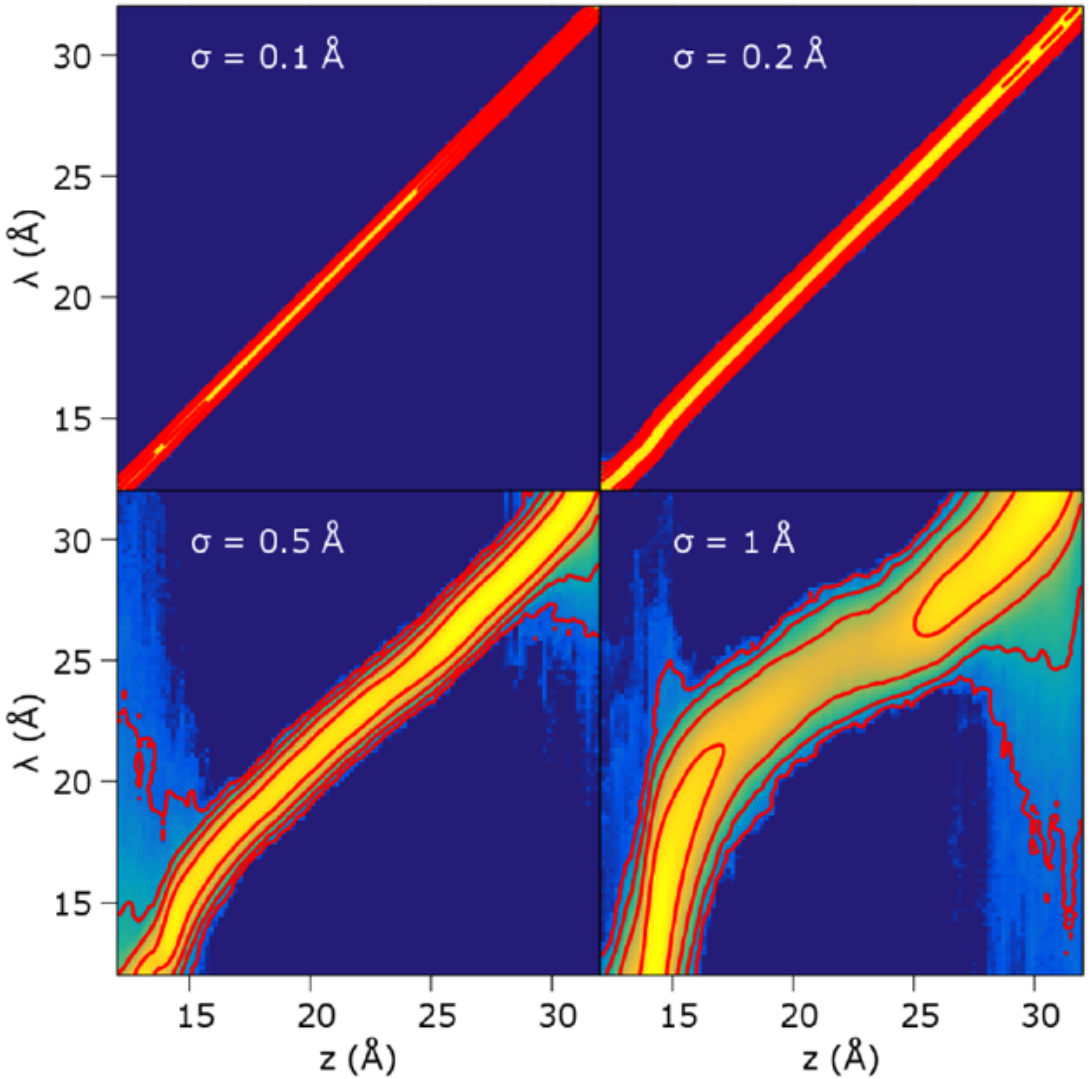
$$F'(z) = -\frac{1}{\beta}\frac{d[\ln \overline{\rho}(z)]}{dz} + \kappa(\langle\lambda\rangle_z - z)$$

with $\overline{\rho}(z) = \int \rho(z,\lambda)\, d\lambda$ and $\langle\lambda\rangle_z = \dfrac{1}{\overline{\rho}(z)}\displaystyle\int \lambda\rho(z,\lambda)\, d\lambda$ (conditional dist.)

Proof: start from $F'(z) = -\dfrac{1}{\beta}\dfrac{\partial_z \rho(z,\lambda)}{\rho(z,\lambda)} - \kappa(z-\lambda)$, multiply both sides of the equality by $\rho(z,\lambda)/\overline{\rho}(z)$ and integrate with respect to $\lambda$

[9] A. Lesage, T. Lelièvre, G. Stoltz and J. Hénin, *J. Phys. Chem. B* (2017)

# Joint distribution of $(\lambda, z)$ (deca-alanine)




logarithmic scale

$$\sigma^2 = \frac{1}{\beta\kappa}$$

Marginal distribution in $\lambda$ nearly uniform (as expected)

# Iterative free energy biasing/ autoencoder learning

# Training on modified target measures

- Interesting systems are metastable (no spontaneous exploration of phase space)
Sample according to a biased distribution $\widetilde{\mu}$ (importance sampling)

- Need for reweighting to learn the correct encoding!

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

- **Minimization problem:** theoretical cost function

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

actual cost function

$$\mathcal{L}(\widehat{\mu}_{\mathsf{wght}}, \mathbf{p}) = \sum_{i=1}^{N} \widehat{w}_i \|x^i - f_{\mathbf{p}}(x^i)\|^2, \qquad \widehat{w}_i = \frac{\mu(x^i)/\widetilde{\mu}(x^i)}{\sum_{j=1}^{N} \mu(x^j)/\widetilde{\mu}(x^j)}$$

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

- Stochastic gradients in training: sampling with replacement according to multinomial distribution

# Proof of concept (1)

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

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

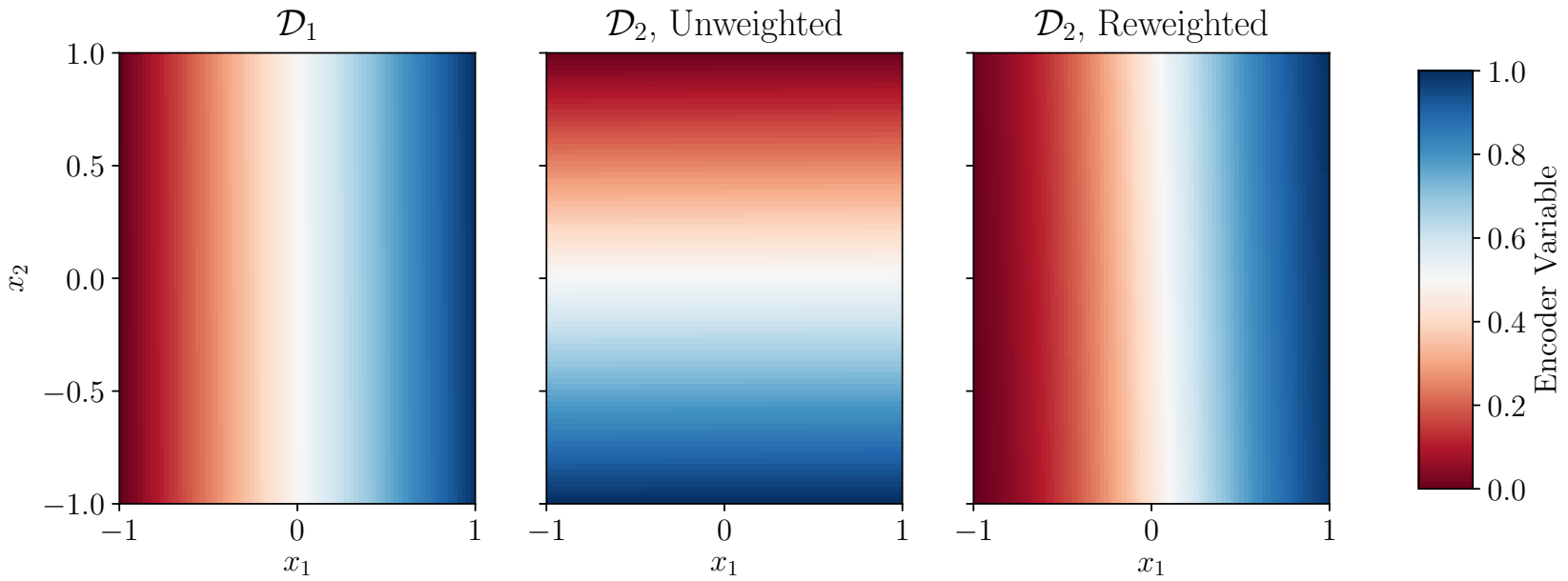
Datasets $\mathcal{D}_i$ of $N = 10^6$ i.i.d. points

- 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 $\widehat{w}_i \propto \mu_1/\mu_2$

# Proof of concept (2)

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

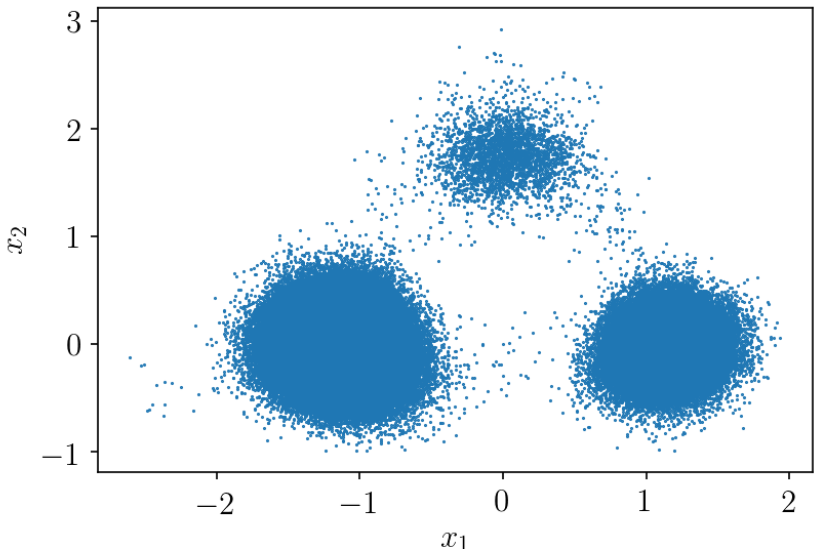


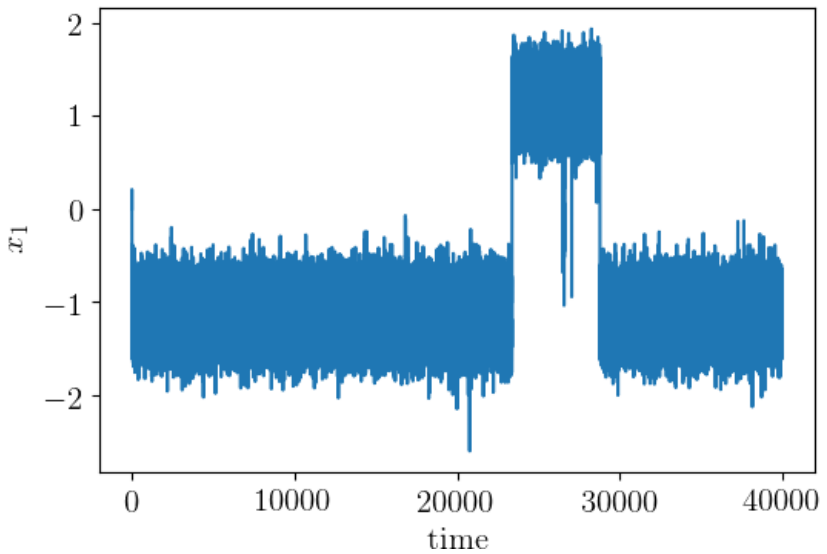

Third encoder very similar to the first one: projection on $x_1$.
Second encoder projects on a direction close to $x_2$.

# Proof of concept with free energy biasing (1)

**Two dimensional potential** ("entropic switch")[10]

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




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 $\Delta t = 10^{-3}$ $\longrightarrow$ metastability in the $x_1$ direction

[10] S. Park, M.K. Sener, D. Lu, and K. Schulten (2003)

# Proof of concept with free energy biasing (2)

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

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

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

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

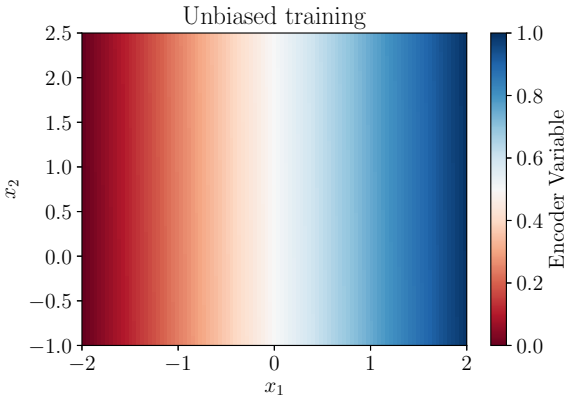
- Autoencoders: 2-1-2 topology, activation functions $\tanh$ (so that CV is in $[-1, 1]$) then identity

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

# Proof of concept with free energy biasing (3)

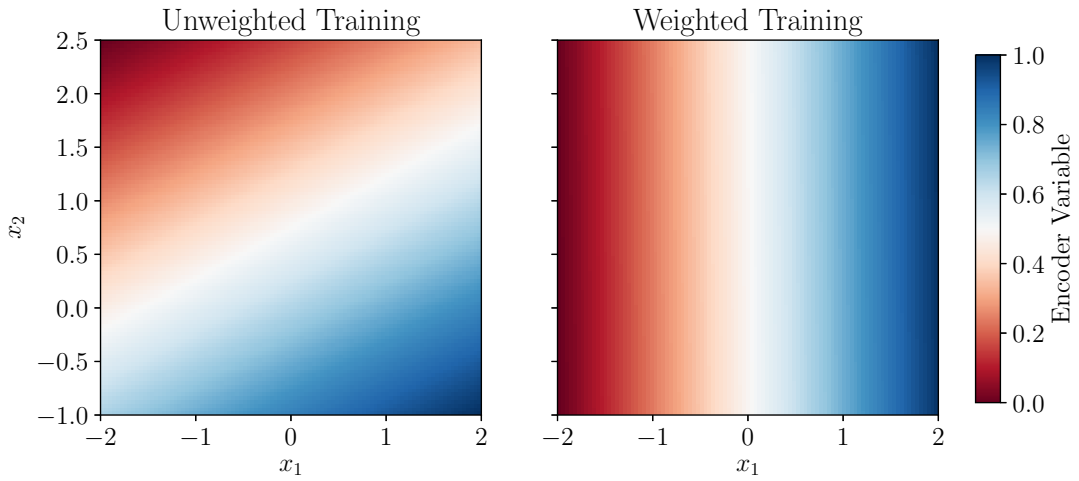
Normalize to compare

$$\xi_{\mathrm{AE}}^{\mathrm{norm}}(x) = \frac{\xi_{\mathrm{AE}}(x) - \xi_{\mathrm{AE}}^{\min}}{\xi_{\mathrm{AE}}^{\max} - \xi_{\mathrm{AE}}^{\min}}$$

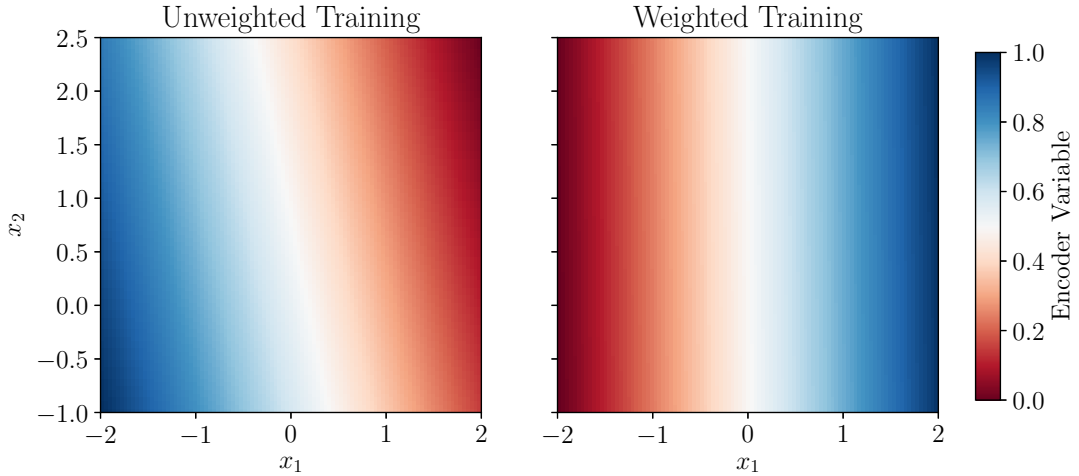



Reference CV

(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 $I_{\max}$, minimum convergence score $s_{\min}$

**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 $(\widehat{w}_0, \dots, \widehat{w}_N) = (1, \dots 1)$

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

**Iterative update of the collective variable**

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

**While** $i < I_{\max}$ and $s < s_{\min}$ — Threshold $s_{\min}$ to be determined

- Set $i \leftarrow i + 1$
- Sample $\text{traj}_i$, $F_i \leftarrow S_{\text{AB}}(q_0, N, \xi_{i-1})$ — in our case: extended ABF
- Compute weights $\widehat{w}_j \propto \mathrm{e}^{-\beta F_i(\xi_{i-1}(x^j))}$
- Initialize autoencoder $\text{AE}_i \leftarrow A_{\text{init}}$
- Train $\text{AE}_i$ on $\text{traj}_i$ with sample weights $\widehat{w}_j$
- Extract the encoder function $\xi_i : x \mapsto \xi_i(x)$
- Set $s \leftarrow \text{regscore}(\xi_{i-1}, \xi_i)$ — Convergence metric to be made precise

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

**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 (1/2)

- Check convergence of CV?

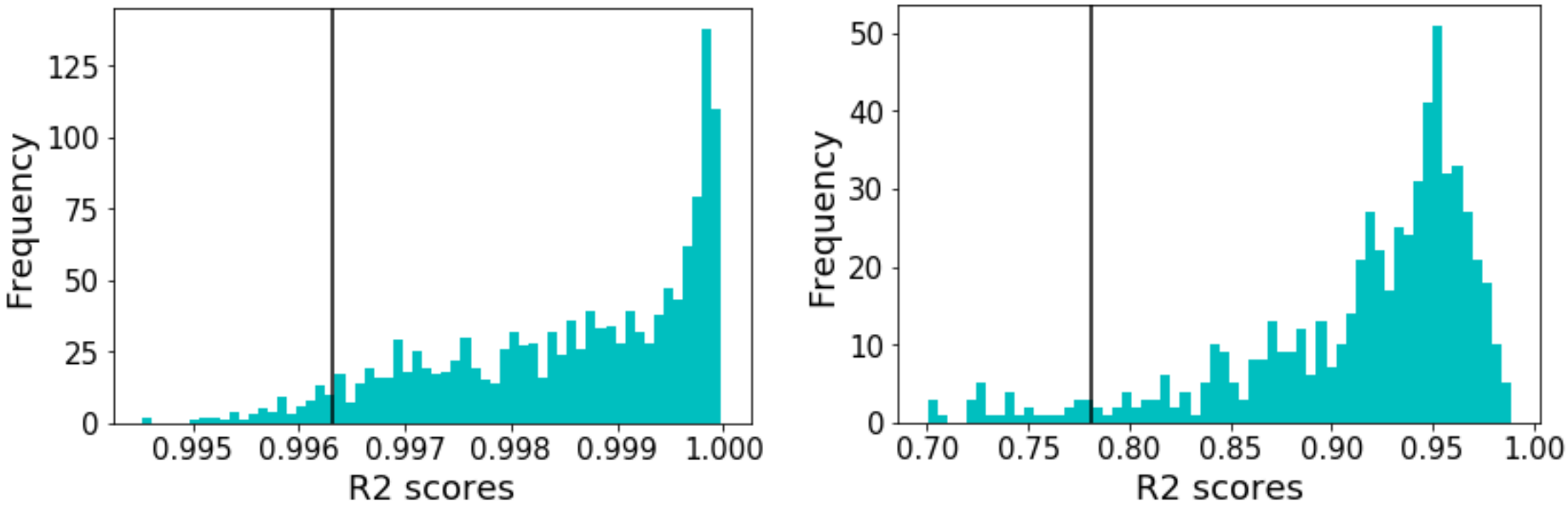
Quantify $\xi_i \approx \Phi(\xi_{i-1})$ for some monotonic function $\Phi$

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

- **Regression score** between $\xi$ and $\xi'$
  - Two sets of values of CV $(\xi(q^1), \dots, \xi(q^N))$ and $(\xi'(q^1), \dots, \xi'(q^N))$
  - Match them with a linear model $M(z) = Wz + b$
  - Coefficient of determination $R^2 = 1 - \dfrac{\sum_{i=1}^{N} \left\| \xi'(q^i) - M(\xi(q^i)) \right\|^2}{\sum_{i=1}^{N} \left\| \xi'(q^i) - \bar{\xi'} \right\|^2}$
  - Maximization of $R^2$ w.r.t. $W, b$ provides regscore$(\xi', \xi)$

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

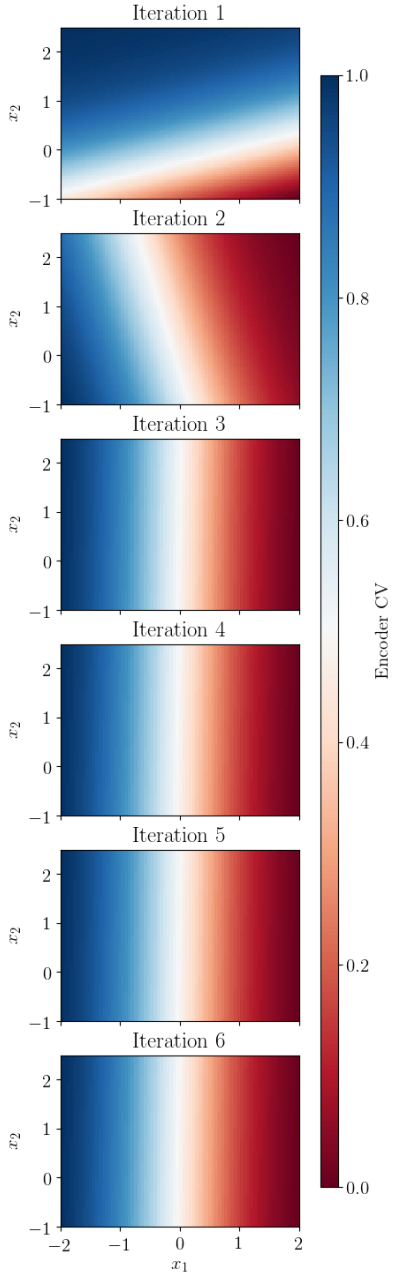
# Discussion on the convergence criterion (2/2)




Histogram of the $R^2$ scores obtained using subsets of $N = 10^5$ points out of $10^6$ points (vertical black line = 5% percentile).

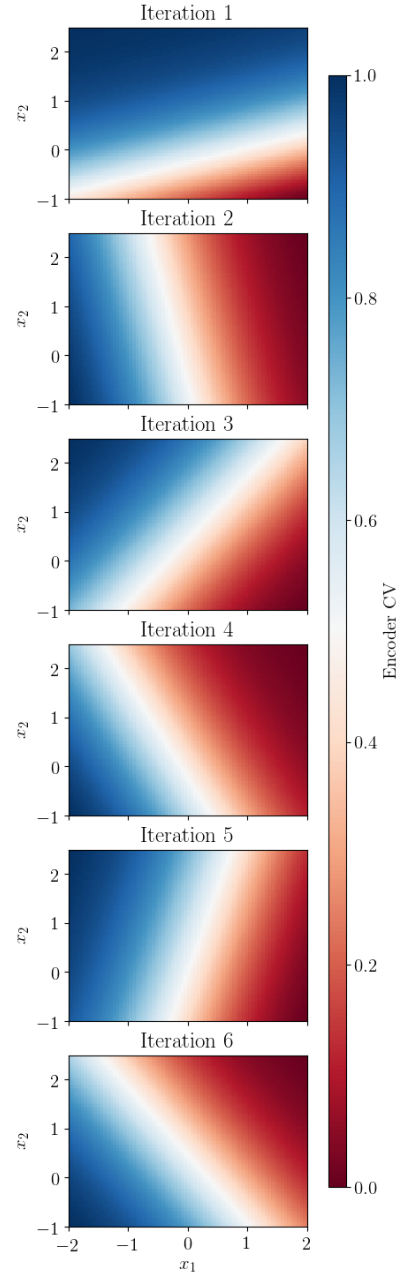
(Left: Alanine dipeptide. Right: Chignolin)

# The iterative algorithm on the toy 2D example




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

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



# Applications

# to systems of interest

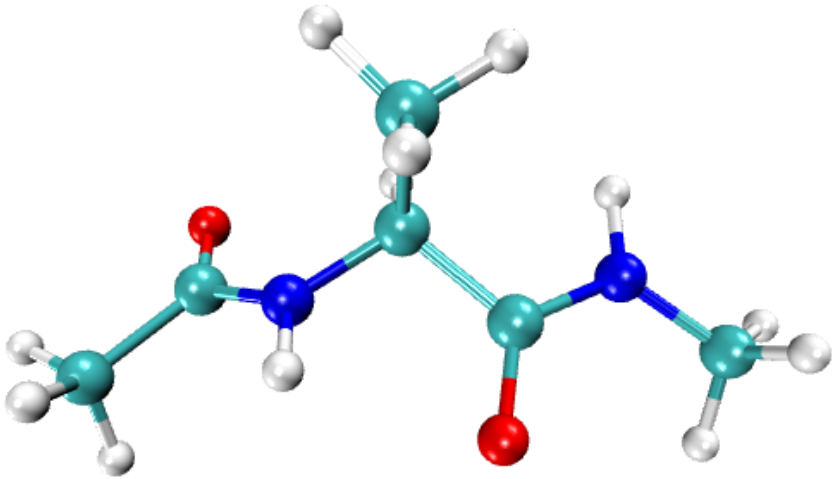
# Alanine dipeptide

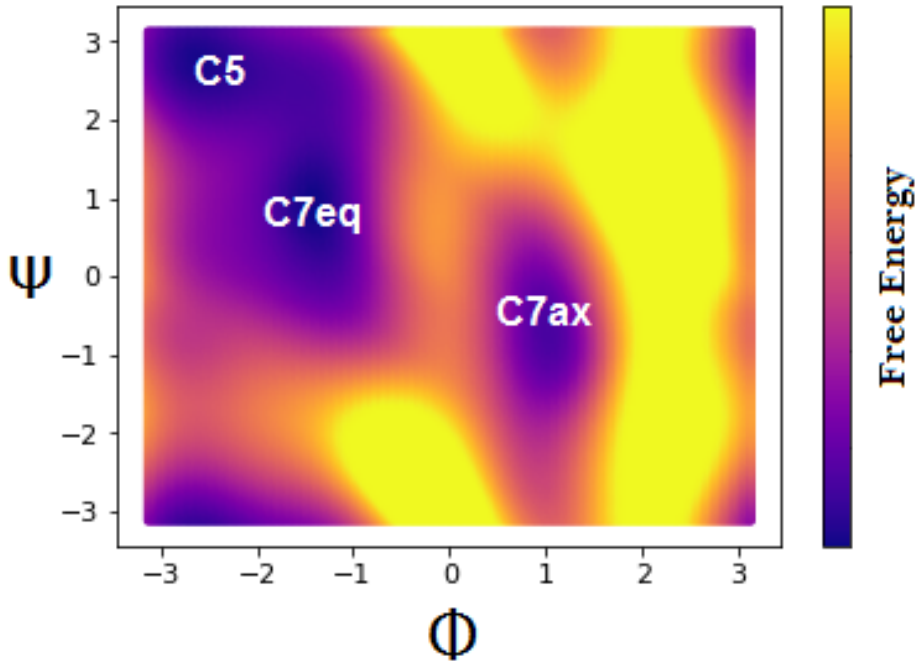
- **Molecular dynamics:**

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

- **Machine learning:**

`keras` for autoencoder training
input = carbon backbone (realignement to reference structure and centering)
neural network: topology 24-40-2-40-24, tanh activation functions

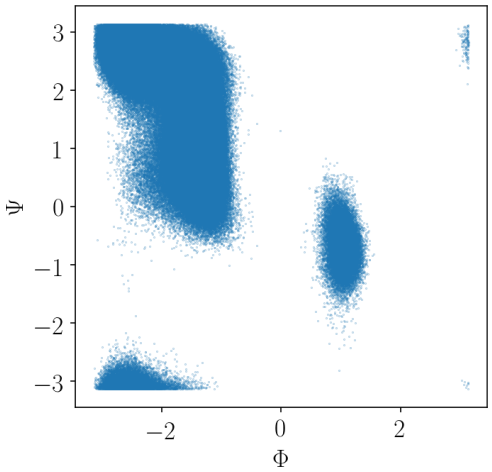


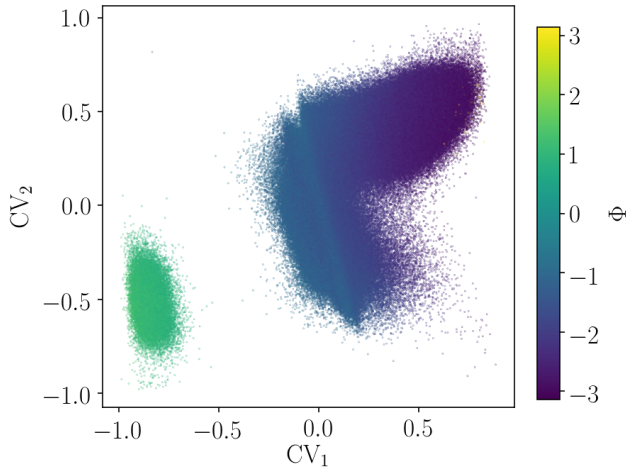


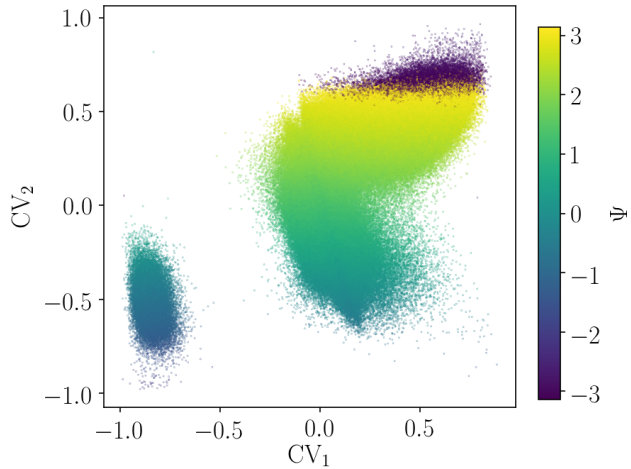
# Ground truth computation

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

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



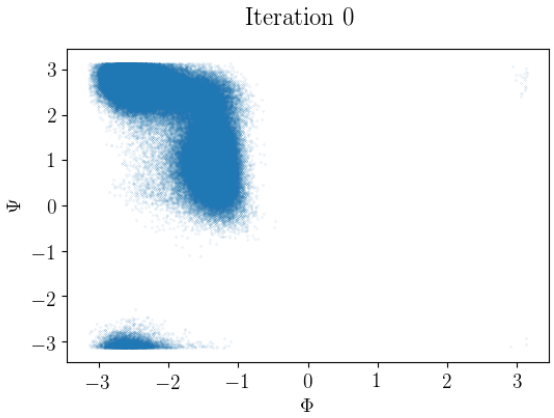




Quantify $s_{\min} = 0.99$ for $N = 10^5$ using a bootstraping procedure

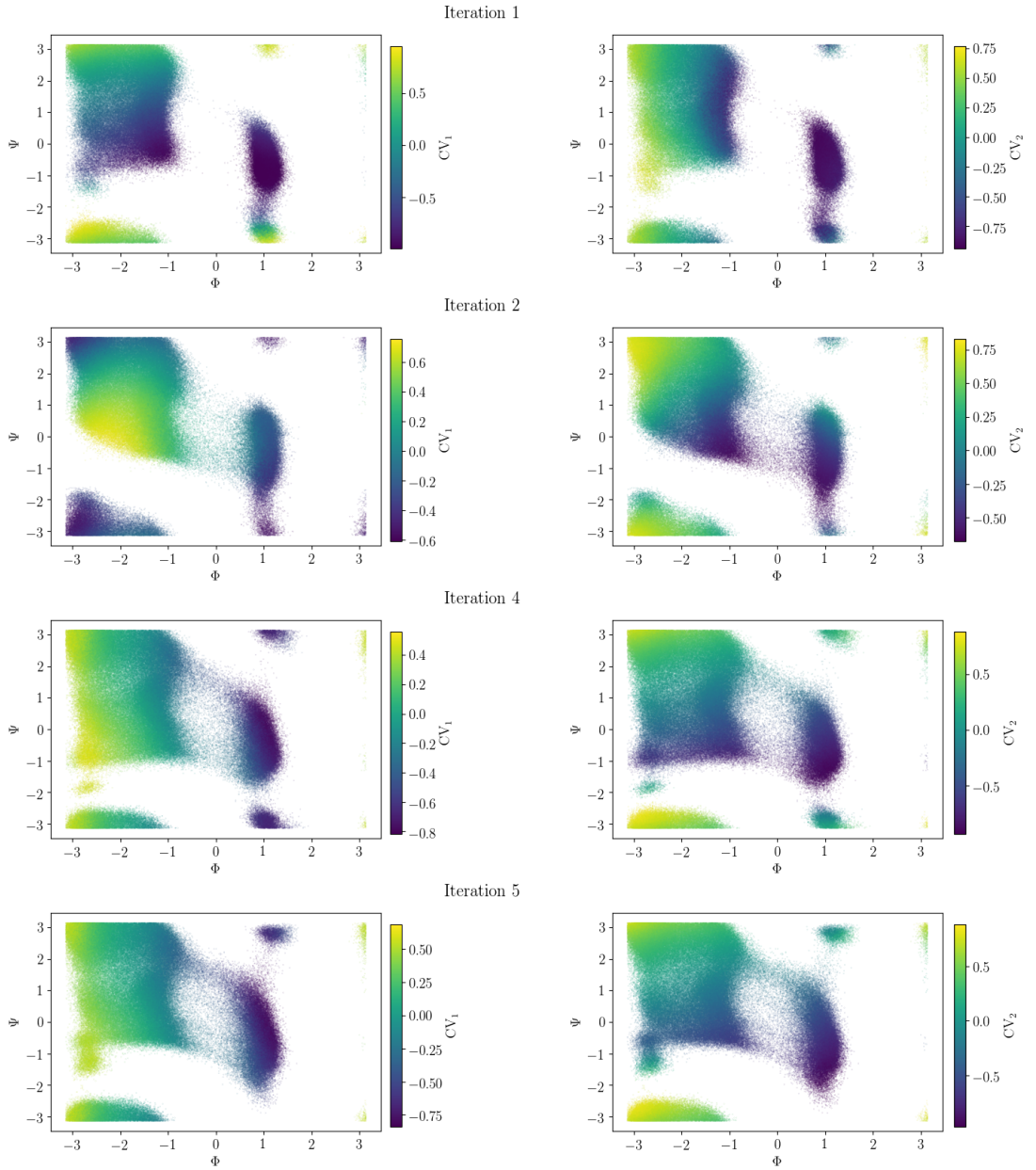
**For the iterative algorithm:** 10 ns per iteration

(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)

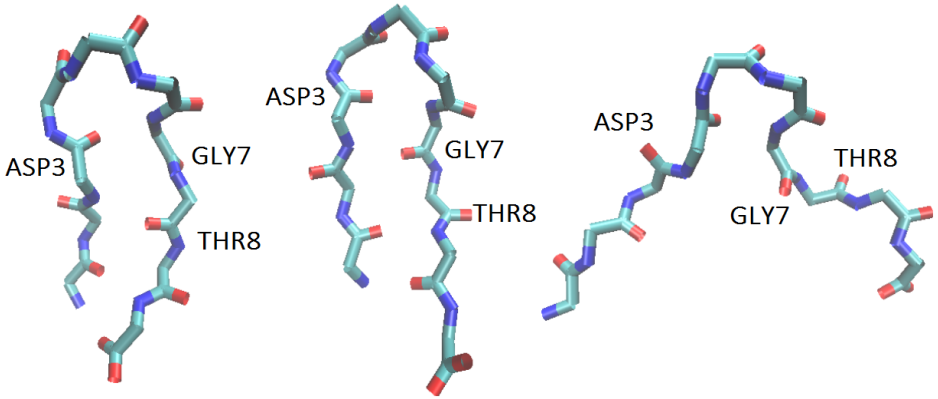
# Results for the iterative algorithm



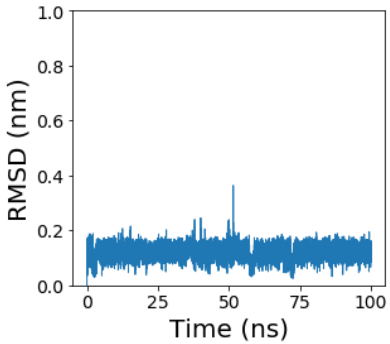

| iter. | regscore | $(\Phi, \Psi)$ |
|---|---|---|
| 0 | – | 0.922 |
| 1 | 0.872 | 0.892 |
| 2 | 0.868 | 0.853 |
| 3 | 0.922 | 0.973 |
| 4 | 0.999 | 0.972 |
| 5 | 0.999 | 0.970 |
| 6 | 0.999 | 0.971 |
| 7 | 0.999 | 0.967 |
| 8 | 0.998 | 0.966 |
| 9 | 0.999 | 0.968 |

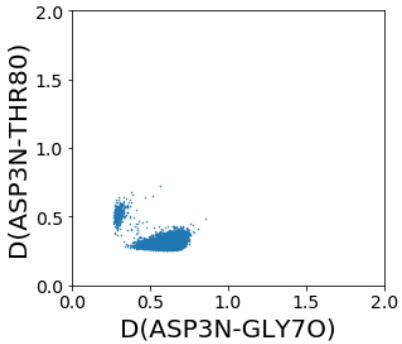


# Chignolin (Folded/misfolded/unfolded states)

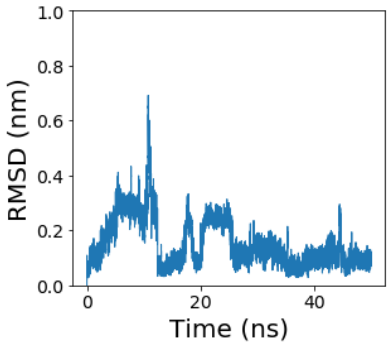


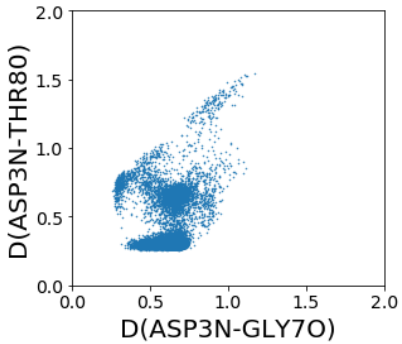

Iteration 0



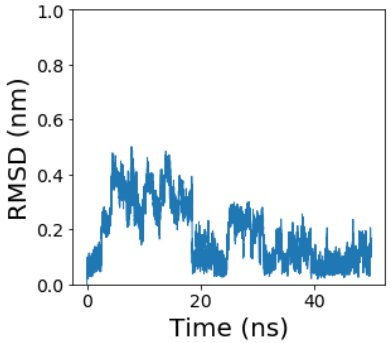



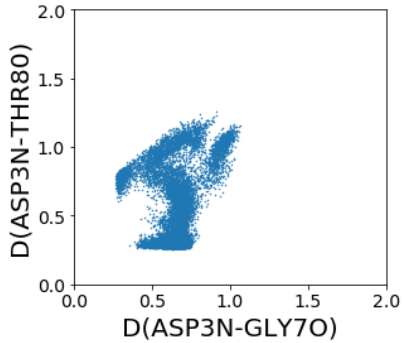

Iteration 1



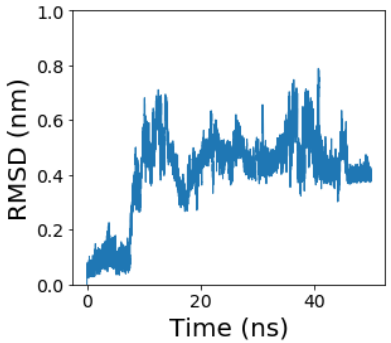



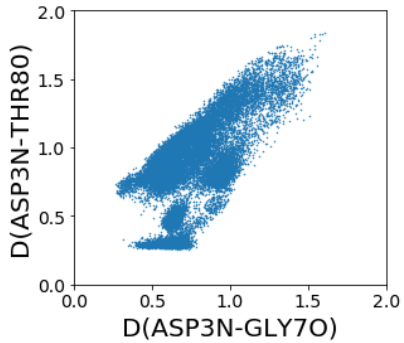

Iteration 3



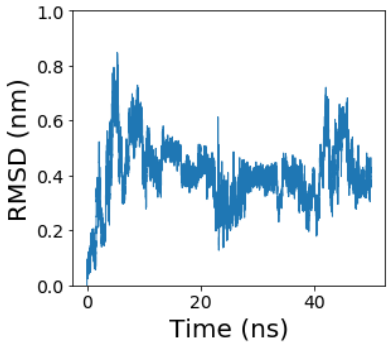



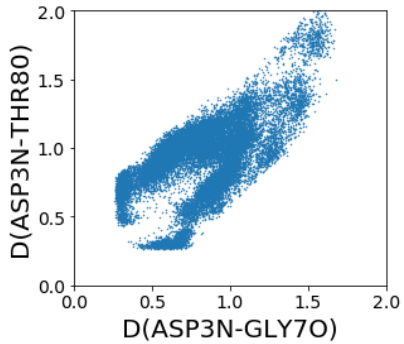

Iteration 5



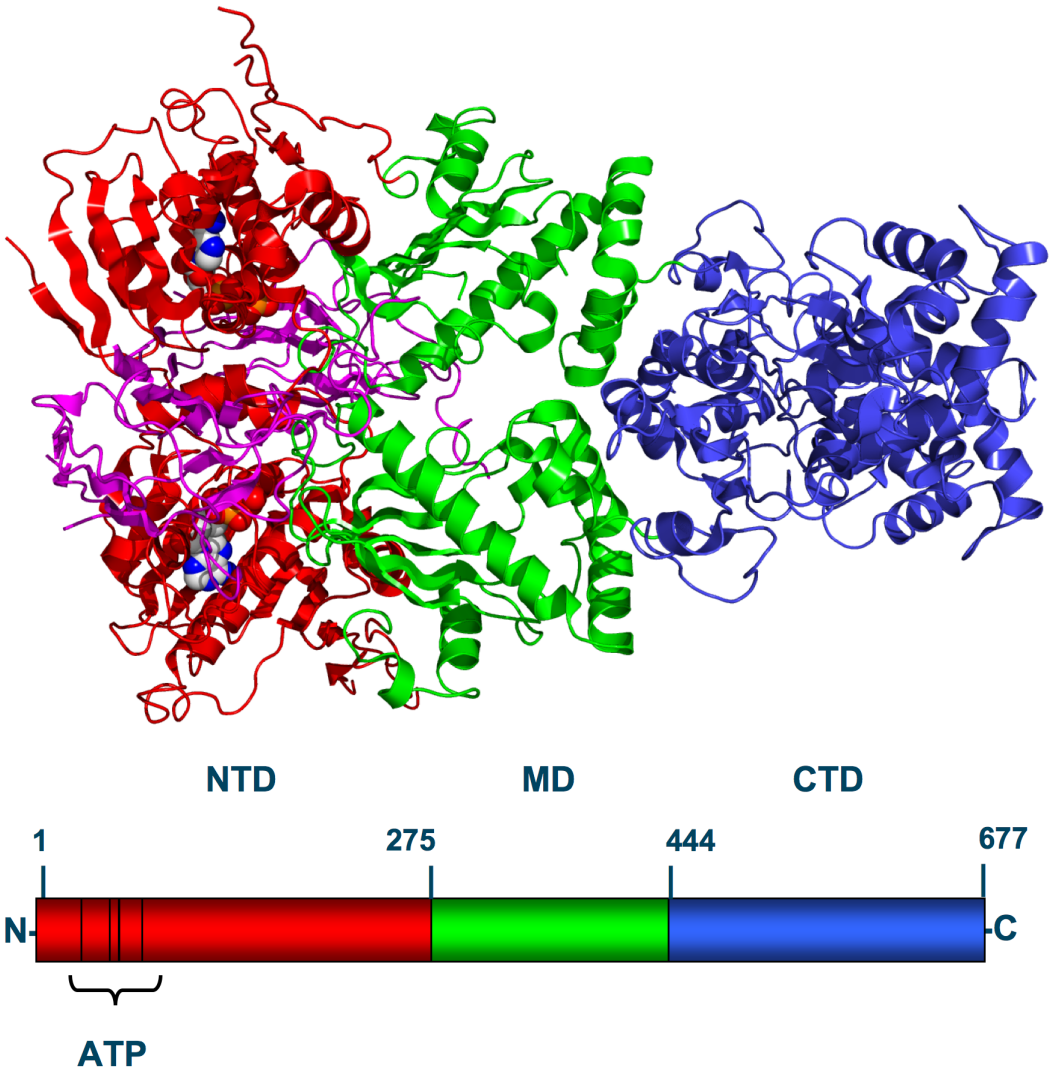




Iteration 6



# HSP90 (work in progress...)

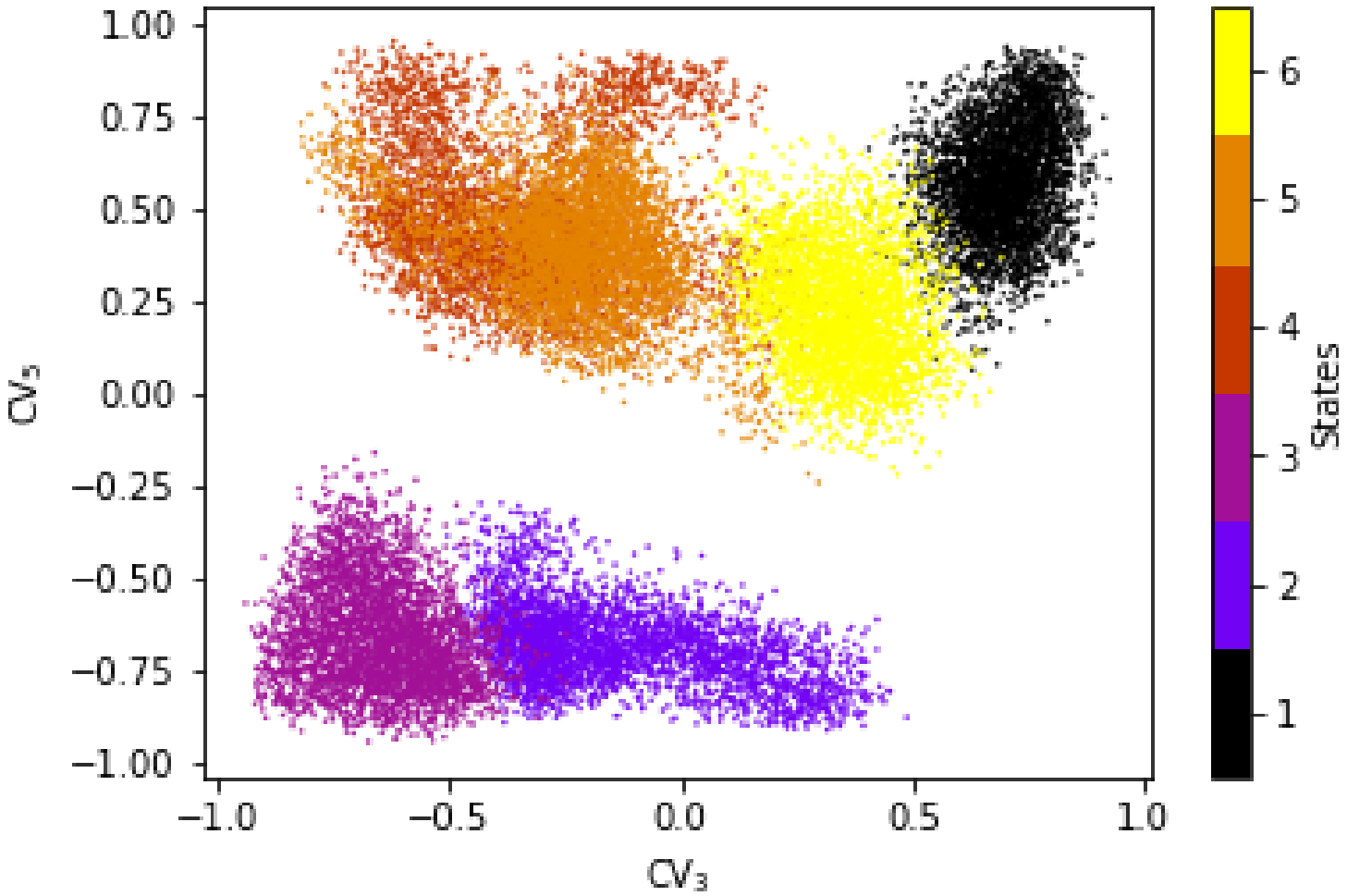



(picture from https://en.wikipedia.org/wiki/File:Hsp90_schematic_2cg9.png)

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

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

# 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