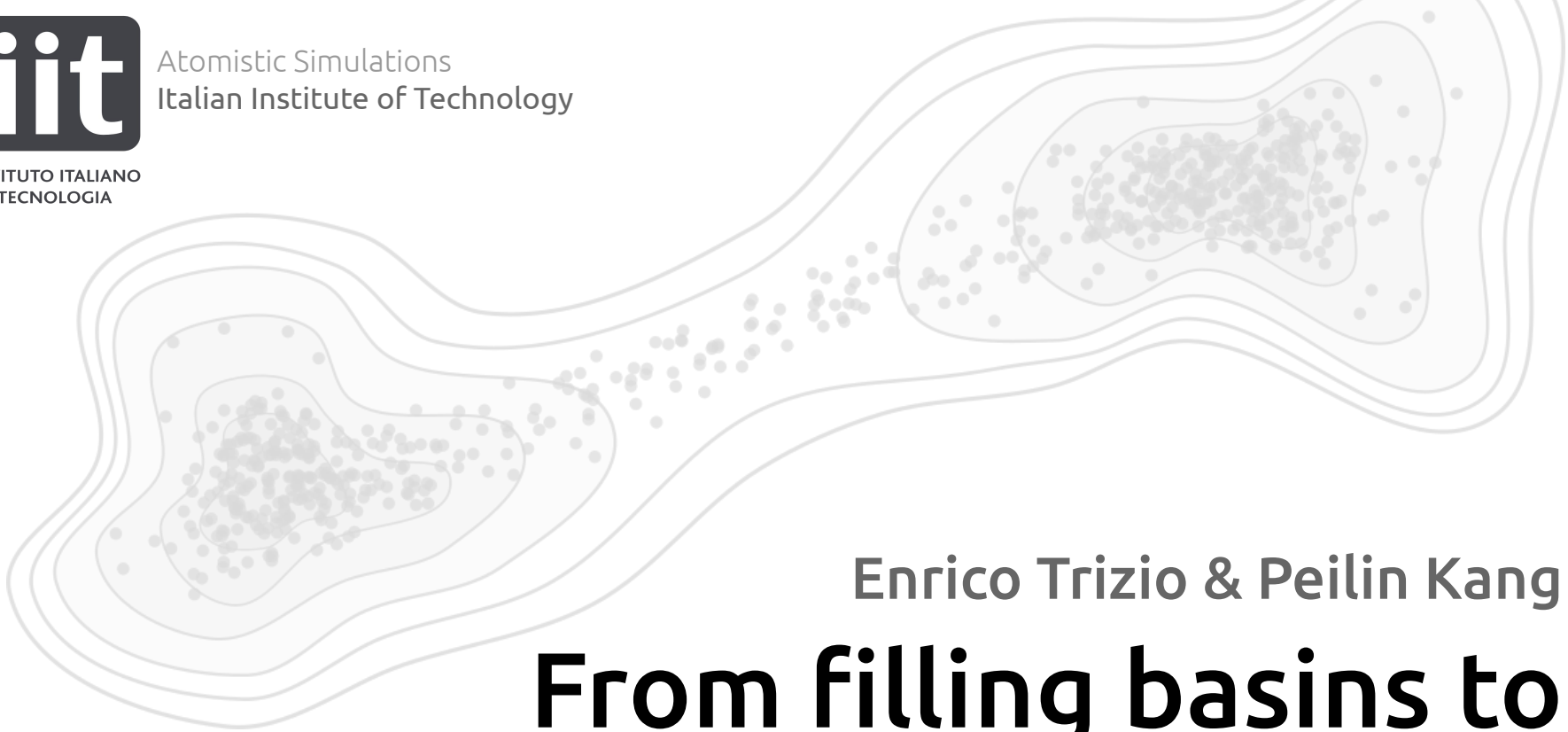
Atomistic Simulations
Italian Institute of Technology

ISTITUTO ITALIANO
DI TECNOLOGIA



Enrico Trizio & Peilin Kang

From filling basins to chopping mountains

Hands on overview

Enhanced sampling

1 Why do we need enhanced sampling?

1 Conventional approaches to enhanced sampling

2 Machine learning collective variables with mlcolvar

Committor wonderland

Machine learning the committor function 3

Transition state oriented enhanced sampling 3

Analyzing the transition state ensemble data 4

Enhanced sampling and collective variables

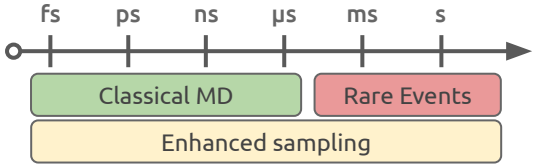
Recall of Pablo's talk

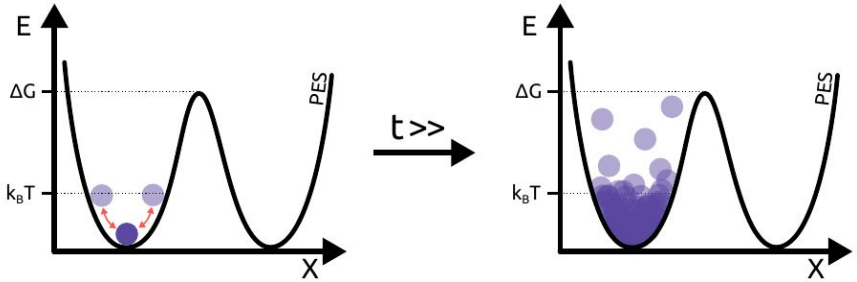
Rare events and (conventional) enhanced sampling

Extending the scope of MD simulations

Standard MD

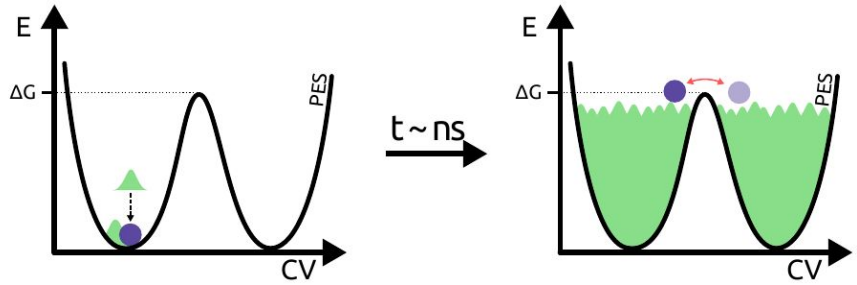
Most of the processes of interest in nature are **rare events** if compared to the timescales of standard MD





Enhanced sampling e.g. Metadynamics[1]

Aims at promoting rare events in simulations. Many methods rely on the addition of **external bias potentials** to reduce free energy barriers.



[1]Laio, Parrinello; PNAS 2002

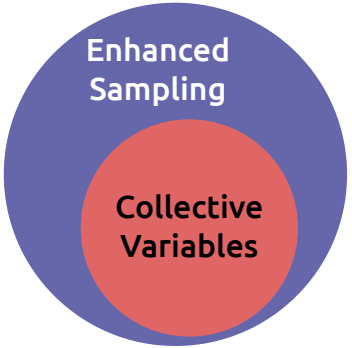
Collective variable based enhanced sampling

Pushing our simulations along the right direction

Collective Variables (CVs)

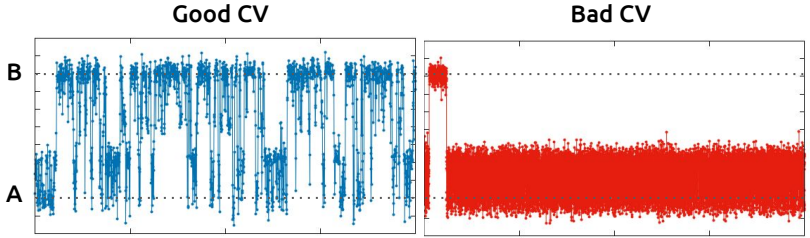
Are functions of the **atomic coordinates** ..... $\mathbf{s}(\mathbf{r}) = (s_1(\mathbf{r}), s_2(\mathbf{r}) \dots s_d(\mathbf{r}))$

- Encode the **relevant modes** of the system
  e.g., drive it through the right path
- **Distinguish** between metastable states
- Operate a **dimensionality reduction**
- **Continuous** and **derivable**!



Enhanced Sampling

Focus on being able to drive the system **across barriers** and observe **transitions**

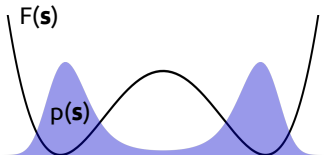


Understanding

Closer to the idea of **order parameters** and/or **reaction coordinates**

$$P(\mathbf{s}) = \int d\mathbf{r}\,\delta[\mathbf{s} - \mathbf{s}(\mathbf{r})]P(\mathbf{r})$$

$$F(\mathbf{s}) = -\frac{1}{\beta}\ln P(\mathbf{s})$$



Building the bias potential

Making our way out of metastable basins

Metadynamics[1] (MetaD)

The free energy landscape is **filled** with repulsive Gaussians centered on the visited points in the CV space

On-the-fly Probability Enhanced Sampling[2-5] (OPES)

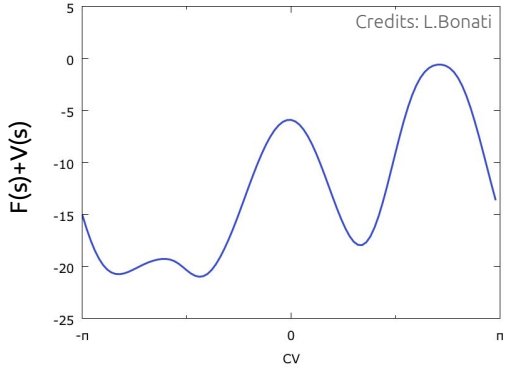
Recent improvement on MetaD, i.e., speed, convergence..
The bias is built on **sampled** and **target probability distributions**

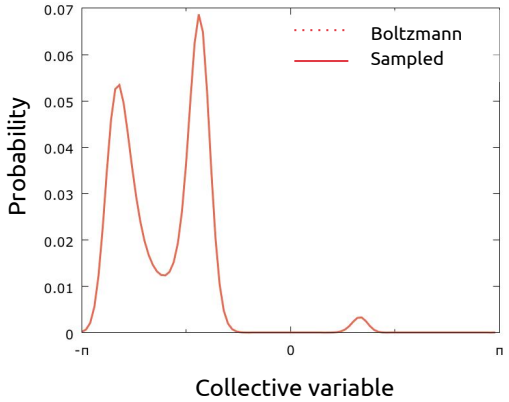
$$V(\mathbf{s}) = \frac{1}{\beta} \ln \frac{P(\mathbf{s})}{p^{tg}(\mathbf{s})}$$

$P(\mathbf{s})$ ..... Sampled probability

$p^{tg}(\mathbf{s})$ ..... Target probability

Different OPES depending on p^{tg}: MetaD[2], Explore[3], Flooding[4], Expanded[5]



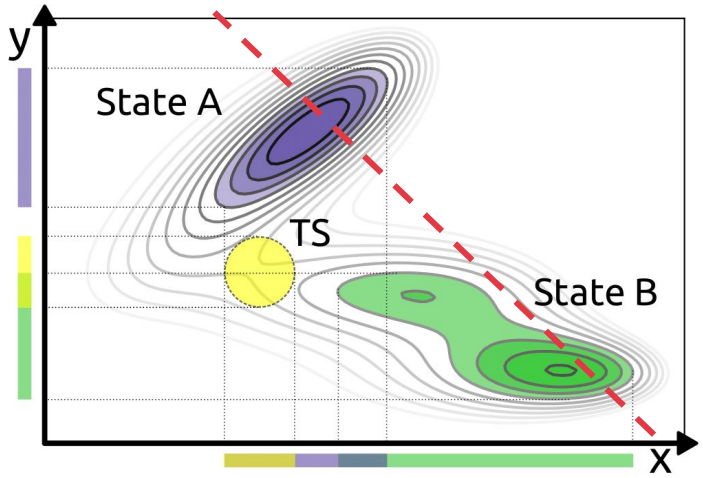


[1]Laio, Parrinello; PNAS 2002 [2]Invernizzi, Parrinello; JPCL 2020 [4]Ray *et al.*; JCTC 2022
[3]Invernizzi, Parrinello; JCTC 2022 [5]Invernizzi *et al.*; Phys. Rev. X 2020

What makes a good CV?

Not so obvious

	Desired properties		
CV	1. Dimensionality reduction	2. Distinguish metastable states	3. Distinguish the TS
x	Yes	So and so	Nope
y	Yes	Yes	Okish
x-y	Yes	Yes	Yes



Data-driven CVs and the mlcolvar library

Methods and applications

Machine learning collective variables

A hot topic in the enhanced sampling community

In the past few year several data-driven approaches have been put forward

Collective variable discovery and enhanced sampling using autoencoders: Innovations in network architecture and error function design

Wei Chen,[1] Aik Rui Tan,[2] and Andrew L. Ferguson[1,2,3,a)]

Reweighted autoencoded variational Bayes for enhanced sampling (RAVE)

João Marcelo Lamim Ribeiro,[1] Pablo Bravo,[2,3] Yihang Wang,[4] and Pratyush Tiwary[1]

Automated design of collective variables using supervised machine learning

Mohammad M. Sultan[1] and Vijay S. Pande[2,a)]

Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics

Christoph Wehmeyer[a)] and Frank Noé[b)]

Collective Variables from Local Fluctuations

Dan Mendels,[†,‡,¶] GiovanniMaria Piccini,[†,‡,¶] and Michele Parrinello*[,†,‡]

Data-Driven Collective Variables for Enhanced Sampling

Luigi Bonati, Valerio Rizzi, and Michele Parrinello*

Collective Variables for Conformational Polymorphism in Molecular Crystals

Oren Elishav,[||] Roy Podgaetsky,[||] Olga Meikler, and Barak Hirshberg*

From Enhanced Sampling to Reaction Profiles

Enrico Trizio and Michele Parrinello*

Transferable Neural Networks for Enhanced Sampling of Protein Dynamics

Mohammad M. Sultan,[†] Hannah K. Wayment-Steele,[†] and Vijay S. Pande*[,‡]

Nonlinear discovery of slow molecular modes using state-free reversible VAMPnets

Wei Chen,[†] Hythem Sidky,[†] and Andrew L. Ferguson[†,‡]

Deep learning collective variables from transition path ensemble

Dhiman Ray,[†] Enrico Trizio,[†,‡] and Michele Parrinello[†,‡]

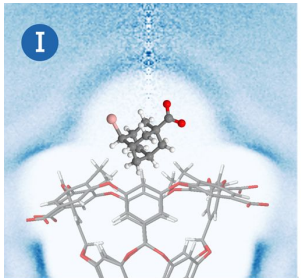
Deep learning the slow modes for rare events sampling

Luigi Bonati[a,b,1], GiovanniMaria Piccini[a], and Michele Parrinello[b,1]

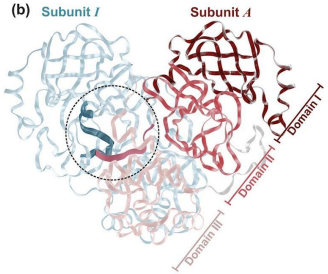
Multitask Machine Learning of Collective Variables for Enhanced Sampling of Rare Events

Lixin Sun,* Jonathan Vandermause, Simon Batzner, Yu Xie, David Clark, Wei Chen, and Boris Kozinsky*

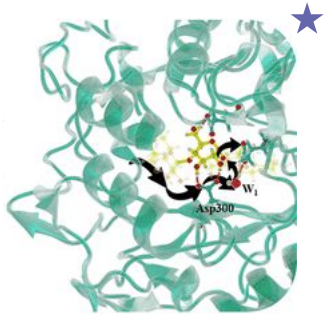
… which have been applied to a wide range of phenomena



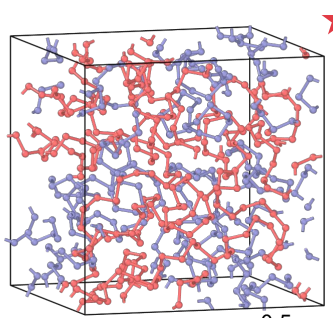
Rizzi *et al*, NatComm 2021



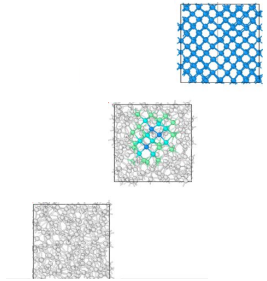
Ansari *et al*, JACS 2021



Das *et al*, ACS Cat. 2023



Yang *et al*, Chem. Sci. 2024



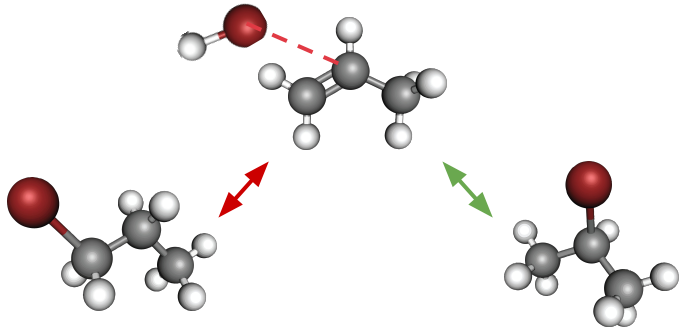
Bonati *et al*, PNAS 2021

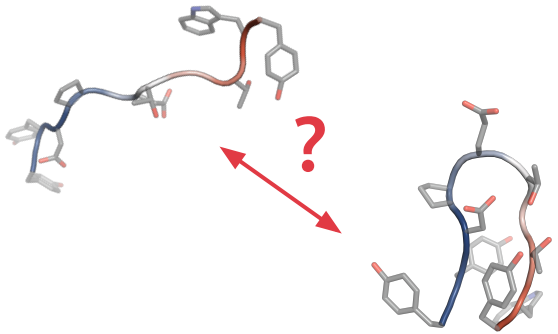
Conventional approach to CVs design

Intuitive but rather limited

Conventionally, CVs were chosen based on physical and chemical **intuition** using simple **physical descriptors**
e.g., distances, angles..

This makes them readily interpretable!





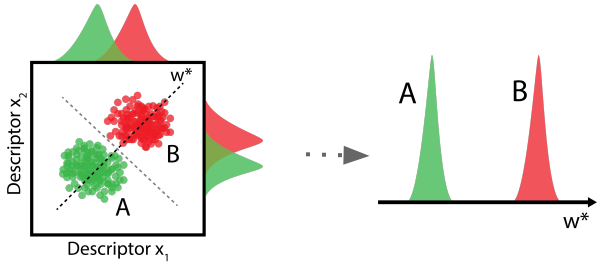
However, they are often too simple and limited

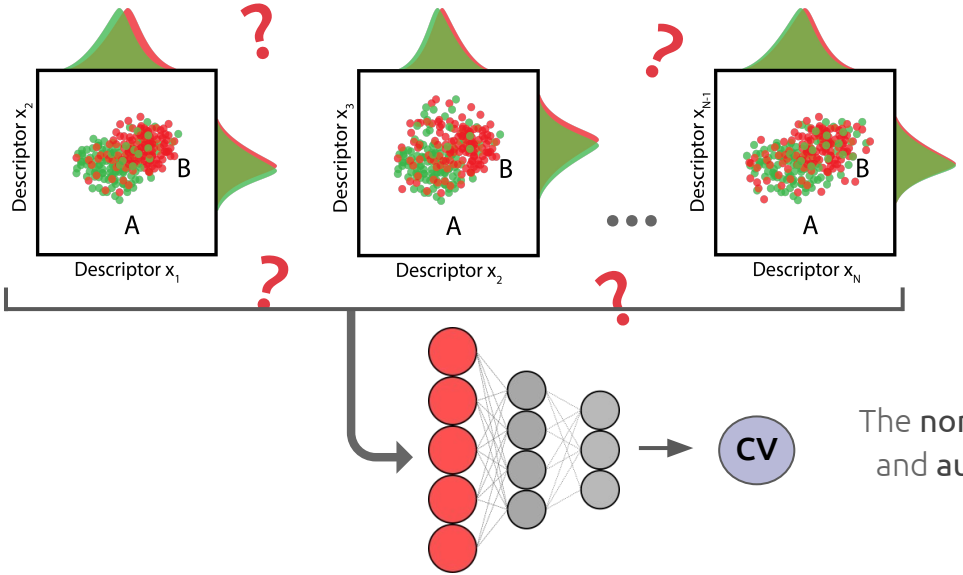
- May be difficult (if not impossible) to identify
- Can easily overlook important variables even on ‘simple’ systems

Data-driven approaches

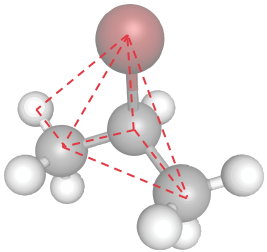
Linear vs non-linear

Linear statistical method like PCA[1] or LDA[2] can be used to meaningfully combine different physical descriptors into **low-dimensional CVs**[3]
e.g. Using a **classification criterion**





But (more than often) also such approaches can fall short when our data become **more complicated** and **highly multi-dimensional**



The **non-linear** features of NN can boost and **automatize** the CV design task and make it **truly data-driven!**

[1]Principal Component Analysis [2]Linear Discriminant Analysis [3]Piccini, Mendels, Parrinello; JCTC 2018

ML-based data-driven approaches

Making the best out of our data

Key ingredients of ML approaches

1. **Dimensionality reduction** ⇒ Unsupervised
2. **Distinguish** metastable **states** ⇒ Supervised
3. Describe the **slow modes** ⇒ Time-informed

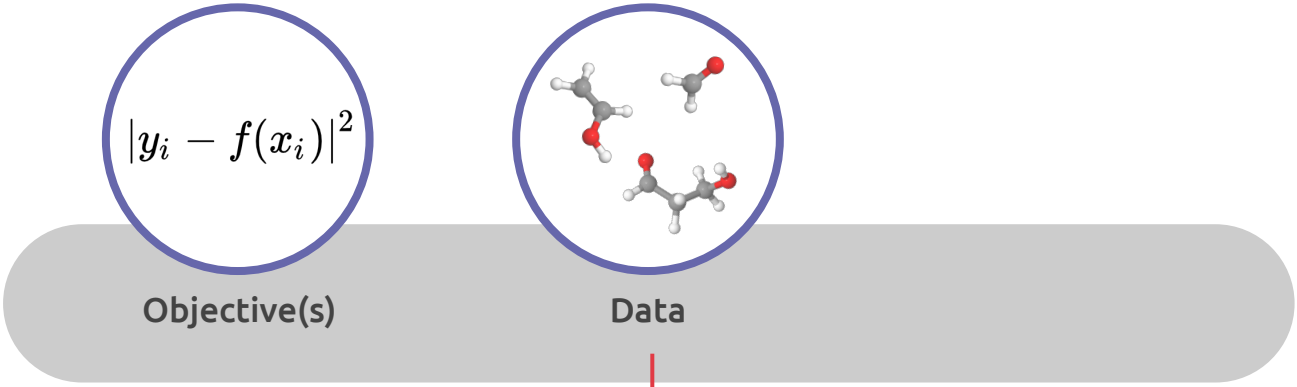
Not always possible to enforce all of them, e.g. due to lack of data

⇒ Often use 1,2 as **surrogates** for 3

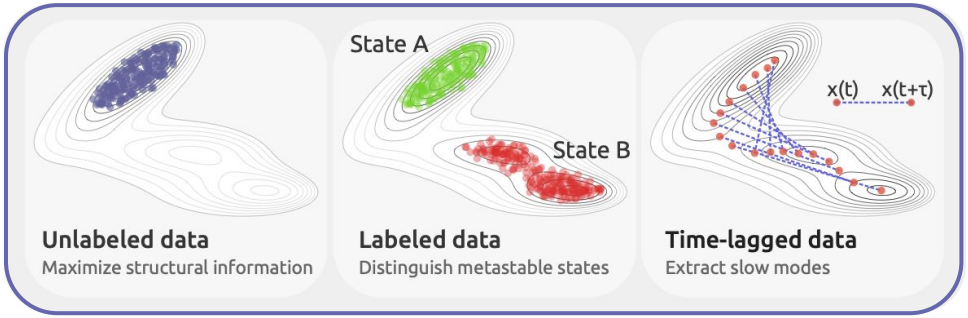
ML-based data-driven approaches

Making our data shine

Key ingredients of ML approaches



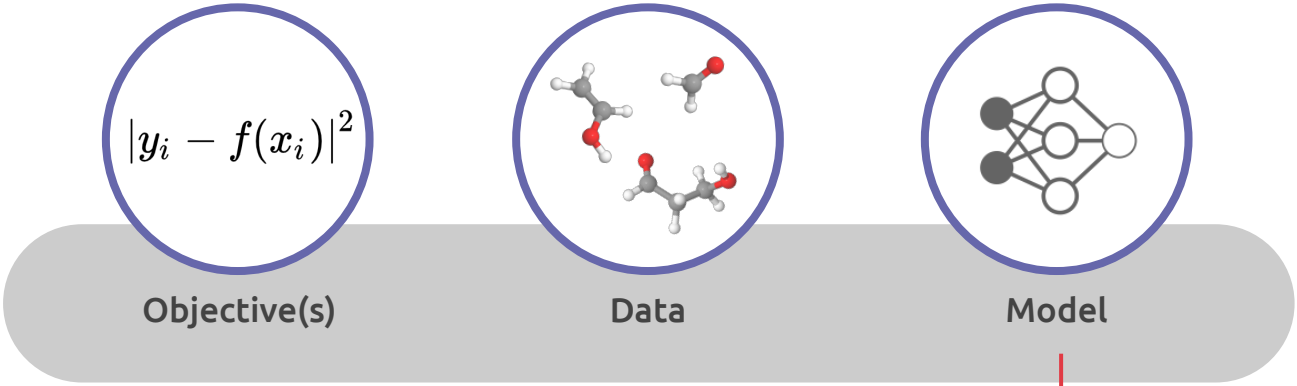
Depending on the available data we can frame the CV design task differently



ML-based data-driven approaches

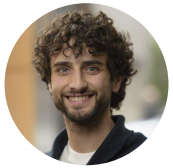
Making our data shine

Key ingredients of ML approaches



We generally take as input set of physical descriptors

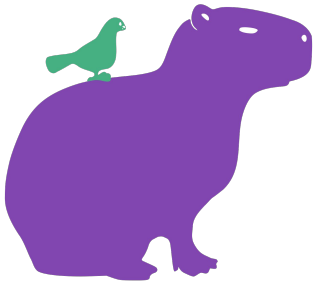
- Continuous and differentiable functions
  - Linear models
  - Neural networks → efficient derivatives via backpropagation
- Trade-off between expressivity and interpretability

The mlcolvar library
A unified framework for ML-CVs

The general framework is the same for most of the methods

Machine Learning Collective Variables[1] (`mlcolvar`): A Python library for data-driven collective-variables (CVs)



Unified framework

Test and utilize (some of) the CVs proposed in the literature

Modular interface

Simplify the development / contamination of approaches

Easy-to-deploy

Use CVs for enhancing sampling via PLUMED

Built on ⇒ PYTORCH PyTorch Lightning PLUMED

User-friendly ⇒ `pip install mlcolvar` `https://mlcolvar.readthedocs.io`

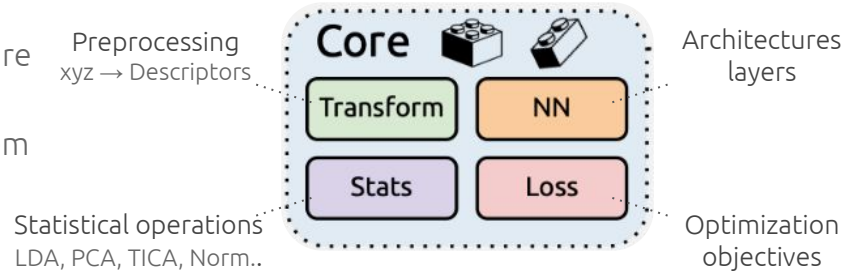
[1]Bonati, Trizio, Rizzi, Parrinello; JCP 159 (2023)

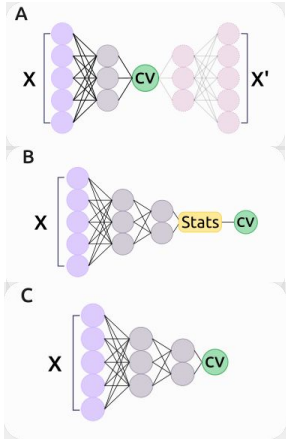
The mlcolvar library

One library to rule them all

Modular interface

The different methods share many components!
In **mlcolvar** we provide them as lego blocks





This way we can provide a unified framework for many methods!

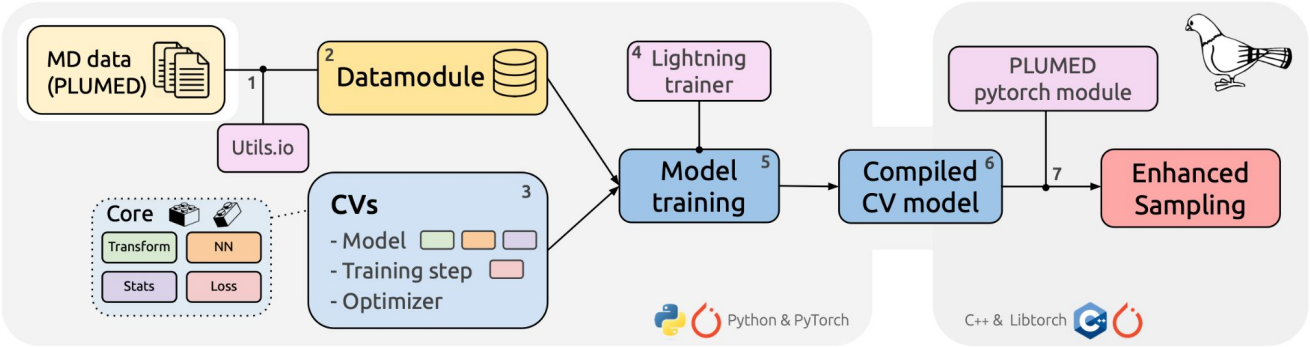
Data	Objective	Method	Architecture
Unlabeled	*Maximize structural information*	PCA	linear
		AutoEncoder (AE)	a
		Variational AE (VAE)	a
		EncoderMap	a
Labeled	*Distinguish metastable states*	LDA	linear
		Deep-LDA	b
		Deep-TDA	c
Time-lagged	*Slow modes*	TICA	linear
		Deep-TICA/SRV	b
	Slow modes + structural information	*Time-lagged AE (TAE)*	a
		Variational Dynamics Encoder (VDE)	a

These are already implemented, but we can play around!

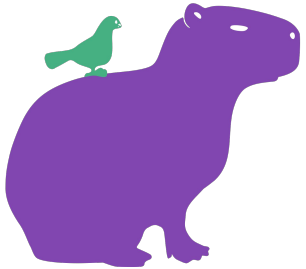
[1]Bonati, Trizio, Rizzi, Parrinello; JCP 159 (2023)

The mlcolvar library

Workflow overview



A minimal example requires only six lines of code!

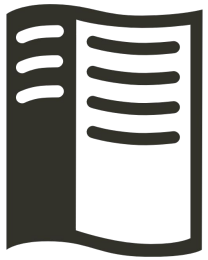


```python
# Setup
import torch,lightning
from mlcolvar.data import DictModule
from mlcolvar.cvs import AutoEncoderCV
from mlcolvar.utils.io import create_dataset_from_files

# 1. Import training data (e.g. PLUMED COLVAR files)
dataset = create_dataset_from_files('./COLVAR')
# 2. Create a Lightning datamodule which splits dataset in train/valid
datamodule = DictModule(dataset, lenghts=[0.8,0.2])
# 3. Choose a model and hyper-parameters
cv_model = AutoEncoderCV(encoder_layers=[45,30,15,2])
# 4. Define a trainer object
trainer = lightning.Trainer(max_epochs=1000)
# 5. Optimize parameters
trainer.fit(cv_model, datamodule)
# 6. Compile the model with TorchScript
cv_model.to_torchscript('model.ptc')
```

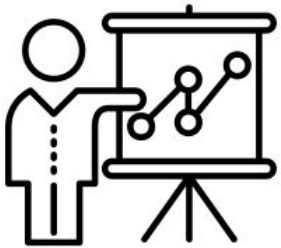
[1]Bonati, **Trizio**, Rizzi, Parrinello; JCP 159 (2023)

User resources

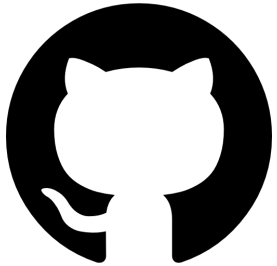


[Documentation]

[Tutorials] colab



[Examples] colab



[GitHub]

Deep Targeted Discriminant Analysis

Deep-TDA for friends

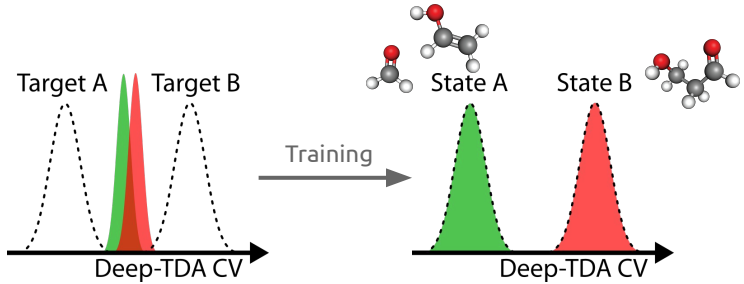
Deep-TDA model[1]

Idea: We know[1,2] how the states should appear in a **good CV space**!

How: **Labeled dataset** from the **metastable basins**

Represent the CV with a **NN** that combines descriptors

Optimize the NN to match a **simple target distribution**

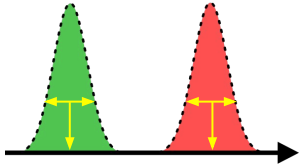


Simple Criterion

Each **state** *k* brings two terms to the **loss function**

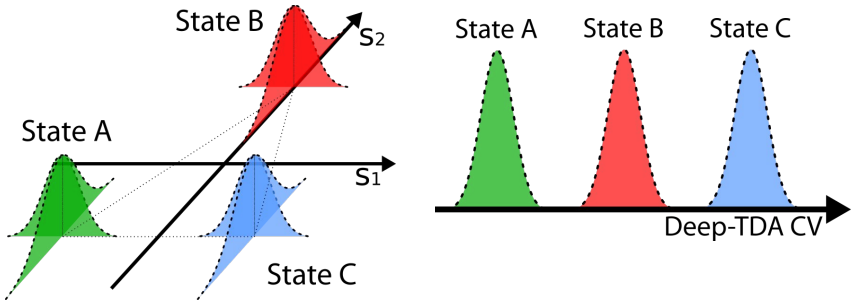
$$L = \alpha \sum_{k}^{N_m} \sum_{l}^{N_d} (\mu_{k,l} - \bar{\mu}_{k,l})^2 + \beta \sum_{k}^{N_m} \sum_{l}^{N_d} (\sigma_{k,l} - \bar{\sigma}_{k,l})^2$$

Center enforcing — Width enforcing



Flexible and multi-dimensional

We can have multiple states and dimensions



[1]Piccini, Mendels, Parrinello; JCTC (2018), [2]Bonati, Rizzi, Parrinello; JPCL (2020), [3]**Trizio**, Parrinello; JPCL (2021)

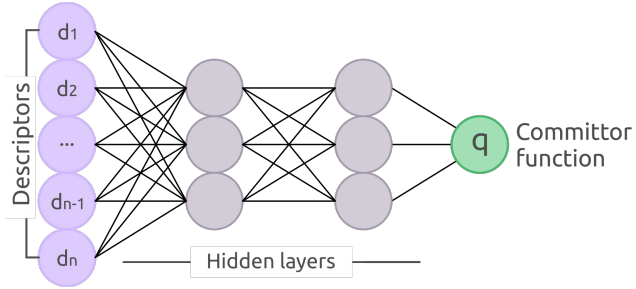
The committor wonderland

Transition state oriented bias potentials

Machine learning the committor function

The machinery

We parametrize the committor function as a **neural network** taking as inputs a set of **physical descriptors**



In the objective function for the optimization we enforce the boundary conditions and the variational principle

$$L = L_v + \alpha L_b$$

$$L_v = \frac{1}{N^n} \sum_i^{N^n} w_i |\nabla_{\mathbf{u}} q(\mathbf{x}_i)|^2$$

Variational loss term
Whole dataset

$$L_b = \frac{1}{N_A} \sum_{i \in A}^{N_A} (q(\mathbf{x}_i))^2 + \frac{1}{N_B} \sum_{i \in B}^{N_B} (q(\mathbf{x}_i) - 1)^2$$

Boundary loss term
Labeled dataset

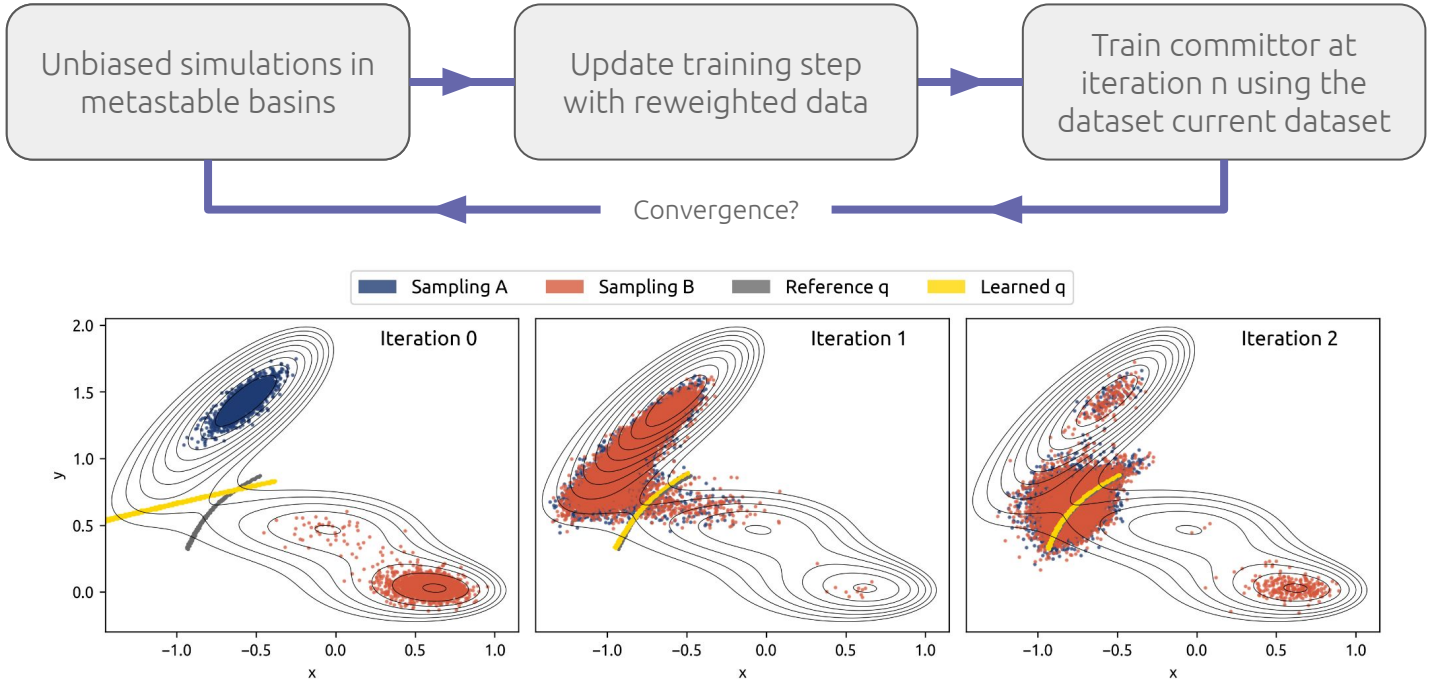
All the code is **open-source** and available through the **mlcolvar library**

mlcolvar

Bonati, Trizio, Rizzi and Parrinello, JCP, 2023

Machine learning the committor function

Making the approach iterative and self-consistent



E. Trizio and M. Parrinello, JPCL, 2021