# Replicating Attempt: Implementing the Radius of Curvature as a collective variable using PYCV

This repository is an attempt to replicate the work described in the paper **"PYCV: a PLUMED 2 Module Enabling the Rapid Prototyping of Collective Variables in Python"**, published on 11 October 2019. The goal is to implement a custom collective variable (CV) using Python and integrate it with PLUMED through the PythonCVInterface (PyCV).

## Description

The code provided implements a custom function (`r_f`) and its gradient using the JAX library. The collective variable is calculated based on the relative positions of atoms and is designed to interact with PLUMED via the PyCV module. The `plumedCalculate` function serves as the bridge between the Python function and the PLUMED framework, passing values and gradients back to PLUMED during a molecular dynamics simulation.

## Key Features

- **Custom CV Calculation**: Computes a collective variable using JAX, leveraging automatic differentiation for gradients.
- **PLUMED Integration**: Uses the `plumedCommunications` interface for seamless interaction with PLUMED simulations.

## Requirements

- **PLUMED**: Ensure PLUMED 2 with the PyCV plugin is installed and configured correctly.
- **Python Libraries**:
  - `jax`
  - `jax.numpy`
  - `plumedCommunications`

## How to Use

1. Compile the PyCV plugin for PLUMED if not already done:

```
./standaloneCompile.sh
```

2. Ensure the Python script is in your `PYTHONPATH`.

3. Configure the PLUMED input file (`.dat`) to load the Python function:

```
LOAD FILE=/path/to/PythonCVInterface.so
rc: PYTHONCV IMPORT=curvature ATOMS=1,4,3 Function = r
PRINT ARG=rc FILE=output_colvar.out
RESTRAINT ARG=rc AT=0 KAPPA=0 SLOPE=1
```

4. Run the PLUMED simulation with the appropriate driver command:

```
plumed driver --plumed plumed_input.dat --mf_xtc
    trajectory.xtc --timestep 0.002
```

# References


- The PLUMED Consortium
- **PYCV: a PLUMED 2 Module Enabling the Rapid Prototyping of Collective Variables in Python** (2019)