PPDyn

Release 0.0.2

Sayan Adhikari (UiO), Rupak Mukherjee (PPPL)

Sep 24, 2022

CONTENTS:

1 Installation **1**
- 1.1 Prerequisites . . . 1
- 1.2 Procedure . . . 1
  - 1.2.1 Using *pip* from PyPI . . . 1
  - 1.2.2 Using *git clone* from GitHub . . . 1

2 Getting Started **3**
- 2.1 PyPI build . . . 3
- 2.2 GNU Make build . . . 3

3 Setting up parameters for simulation **5**

4 Contributing to PPDyn **7**
- 4.1 Our Development Process . . . 7
- 4.2 Pull Requests . . . 7
- 4.3 Contributions under the MIT Software License . . . 7
- 4.4 Issues . . . 7
- 4.5 Coding Style . . . 8

5 Physics Model **9**

6 Indices and tables **11**

CHAPTER

ONE

INSTALLATION

1.1 Prerequisites

- Python3 or higher
- GNU Make (If you are installing from GitHub)
- git (If you are installing from GitHub)

1.2 Procedure

PPDyn can be installed in two ways.

1. *Using pip from PyPI*
2. *Using git clone from GitHub*

1.2.1 Using *pip* from PyPI

```
pip install PPDyn
```

1.2.2 Using *git clone* from GitHub

First make a clone of the master branch using the following command

```
git clone https://github.com/sayanadhikari/PPDyn.git
```

Then enter inside the *PPDyn* directory

```
cd PPDyn
```

Now complile and built the *PPDyn* code

```
make all
```

CHAPTER

TWO

GETTING STARTED

2.1 PyPI build

If you used PyPI to build the project, Download the input template to your working directory

```
wget https://raw.githubusercontent.com/sayanadhikari/PPDyn/main/input.ini
```

Now, either create a python script in your working directory or use your python console

```
from PPDyn import ppdyn
from PPDyn.ppdplot import animate
import time

start = time.time()
ppdyn(input)
end = time.time()
print("Elapsed (after compilation) = %s"%(end - start)+" seconds")
animate()
```

2.2 GNU Make build

If you used GNU Make to build the project, upon successful compilation, run the code using following command

```
ppdyn --i input.ini
```

CHAPTER

THREE

SETTING UP PARAMETERS FOR SIMULATION

Edit the *input.ini* or make your own *.ini* and run the code. The basic structure of *input.ini* is provided below,

```
;
; @file   input.ini
; @brief  PPDyn inputfile.
;
scope = default

[simbox]
Lx  = 10.0    ; System length in X
Ly  = 10.0    ; System length in Y
Lz  = 10.0    ; System length in Z

[particles]
N     = 100     ; Number of particles
Vxmax = 1.0     ; Maximum velocity in X
Vymax = 1.0     ; Maximum velocity in Y
Vzmax = 1.0     ; Maximum velocity in Z
Temp  = 0.010   ;

[boundary]
btype = periodic ; Type of boundary

[time]
tmax  = 50.0    ; Final time
dt    = 0.010   ; time step size

[diagnostics]
dumpPeriod  = 10    ; Data dump period
dumpData    = False
[options]
parallelMode  = True  ;set to false to disable parallel
```

CHAPTER

FOUR

CONTRIBUTING TO PPDYN

We want to make contributing to this project as easy and transparent as possible.

4.1 Our Development Process

We use GitHub to sync code to and from our internal repository. We'll use GitHub to track issues and feature requests, as well as accept pull requests.

4.2 Pull Requests

We actively welcome your pull requests.

- Fork the repo and create your branch from master.
- If you've added code that should be tested, add tests.
- Ensure the test suite passes.
- Make sure your code lints.

4.3 Contributions under the MIT Software License

When you submit code changes, your submissions are understood to be under the same MIT License that covers the project. Feel free to contact the maintainers if that's a concern.

4.4 Issues

We use GitHub issues to track public bugs. Please ensure your description is clear and has sufficient instructions to be able to reproduce the issue.

4.5 Coding Style

- tabs for indentation rather than 2 spaces
- 80 character line length

Important: By contributing to PPDyn, you agree that your contributions will be licensed under its MIT License.

Note: This document was adapted from the open-source contribution guidelines for Facebook's Draft

CHAPTER

FIVE

PHYSICS MODEL

Molecular Dynamics simulation (MD) is a powerful method to study atomic and molecular processes. In plasma physics, it has vast applications starting from fusion plasma to dusty plasma. It solves the equation of motion using Newton equations of motion (or Langevin dynamics which includes energy dissipation through an additional friction term):

$$\frac{\partial^2}{\partial t^2}\vec{r}_i = \frac{1}{m_i}\vec{f}_i\vec{f}_i = -\frac{\partial}{\partial r}V(\vec{r}_1(t), \vec{r}_i(t),\ ...,\ \vec{r}_N(t))$$

where $\vec{r}_i(t)$ is the position of atom i at time t with mass m_i, and V is the interaction potential between all N involved species.

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search