The Abdus Salam
International Centre
for Theoretical Physics

MyMD. A program with potential…

Workshop on Computer Programming and Advanced Tools for Scientific Research Work

Manuel Proissl on behalf of MyMD Team
Miramare, Trieste

Get the code: git clone https://github.com/Rhouli/ljmd-c.git

PROJECT MANAGEMENT

- With the intention to restructure an existing C version of a MD code, we have decided to turn this into an object-oriented C++ version and applied the following class structure:

Classes	Group members
MyMD (main class), Helper	Rodrigo
Atoms, Integrator	Manuel, Maksim
Force, Pair(_LJ)	Giovanni, Aris
Python interface	Leopold

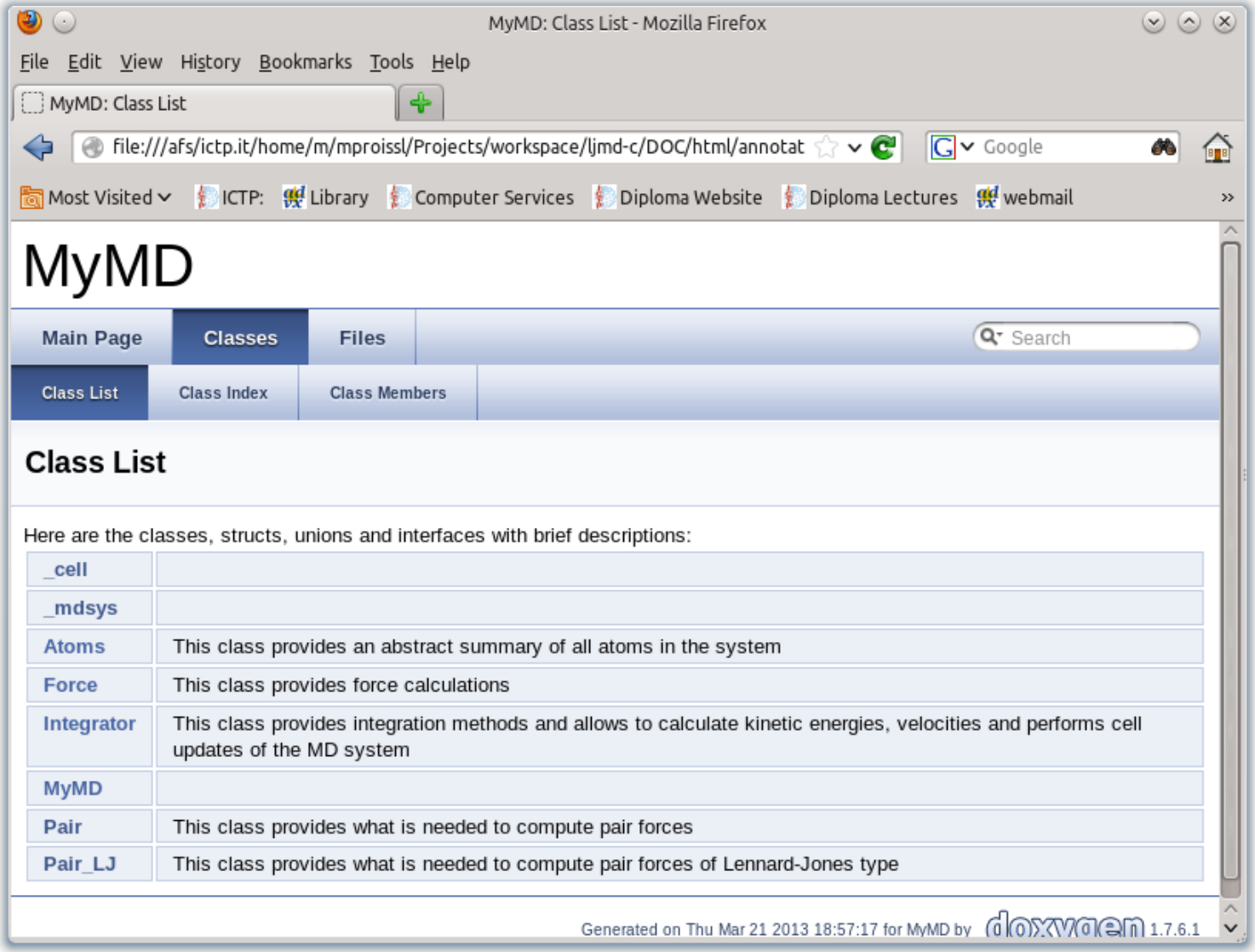
- In addition we have developed a flexible Python interface to run the program with different inputs.

- Working as a group in an amateur-software-company-style:
  - ✓ One main meeting to plan the program structure
  - ✓ Individual work on classes in sub-groups
  - ✓ Several meetings to debug and compile the code
  - ✓ Validating the output with reference data from original C version.

DEVELOPMENT PROBLEMS / STATUS

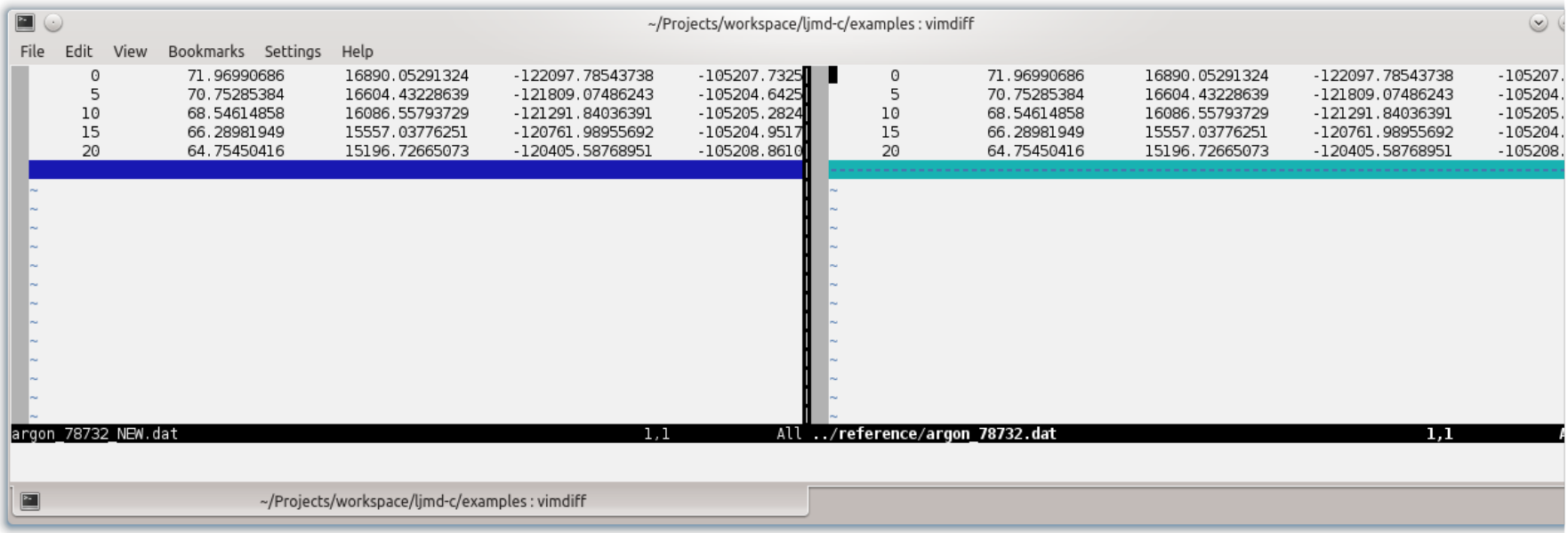
- The baseline: group members with none, a little and a lot C++ experience.
- No coding style conventions were defined.
- Code was only compiled at the end, rather than iteratively.
- Once all pieces were put together, the program “obviously” did not compile.
- Spent one day only on debugging the code and getting it into a working state.
- No improvements in terms of functionality (e.g. adding Morse potential) have been made, BUT the program structure allows now easily extensions!
- A Doxygen website has been created and mostly inline documentation provided.
- A flexible interface in Python has been developed and allows convenient config.

MYMD DOCUMENTATION



MYMD VALIDATION

- Simple validation using reference data and gdb.



- It WORKS!!! (most likely..)

INTERFACE – THE API PROBLEM

- MyMD code to be completely reorganized
  - No fixed C++ API
  - No working code to test against
  - Decisions:
    - Start with ‘API’ = input file format
    - Postpone cython C++ interface to later

INTERFACE – THE TIME PROBLEM

- Coding a usable interface = lot of work
- 4 x ½ day = not a lot time

➢ What can we do?
Check what already has been done for us!

INTERFACE – ASE

Atomic Simulation Environment ¶

The Atomic Simulation Environment (ASE) is the common part of the simulation tools developed at CAMd. ASE provides Python modules for manipulating atoms, analyzing simulations, visualization etc.

> **Note:** The old ASE-2 webpage has moved to http://wiki.fysik.dtu.dk/ase2.

Supported `calculators`:

CALCULATOR CLASS

```
"""This package defines an ASE interface to MyMD

MyMD is an educational project, written during the 2013
Workshop on Computer Programming and Advanced Tools for Scientific Research Work
at ICTP Trieste, Italy.
"""

import os
import sys
import subprocess

import numpy as np

from ase.atoms import Atoms
from ase.calculators.calculator import FileIOCalculator, Parameters, ReadError

import potentials
import io

class FileIOMyMD(FileIOCalculator):
```

EXAMPLE

```
""" Example 03

This example shows how to use the built-in atom viewer
of ASE as well as data visualization with matplotlib.
"""
from ase.lattice.cubic import SimpleCubic
from ase.md.velocitydistribution import MaxwellBoltzmannDistribution
import ase.units as units
import ase.calculators.mymd as mymd

# Set up a cube of 125 Argon atoms in a cube of (15 Angstroms)**3.
myatoms = SimpleCubic('Ar', latticeconstant=3.0, size=(5,5,5))
scalef = 8.31446714569e-7       # adjust velocity units
MaxwellBoltzmannDistribution(myatoms, temp=50*scalef, force_temp=True)
```

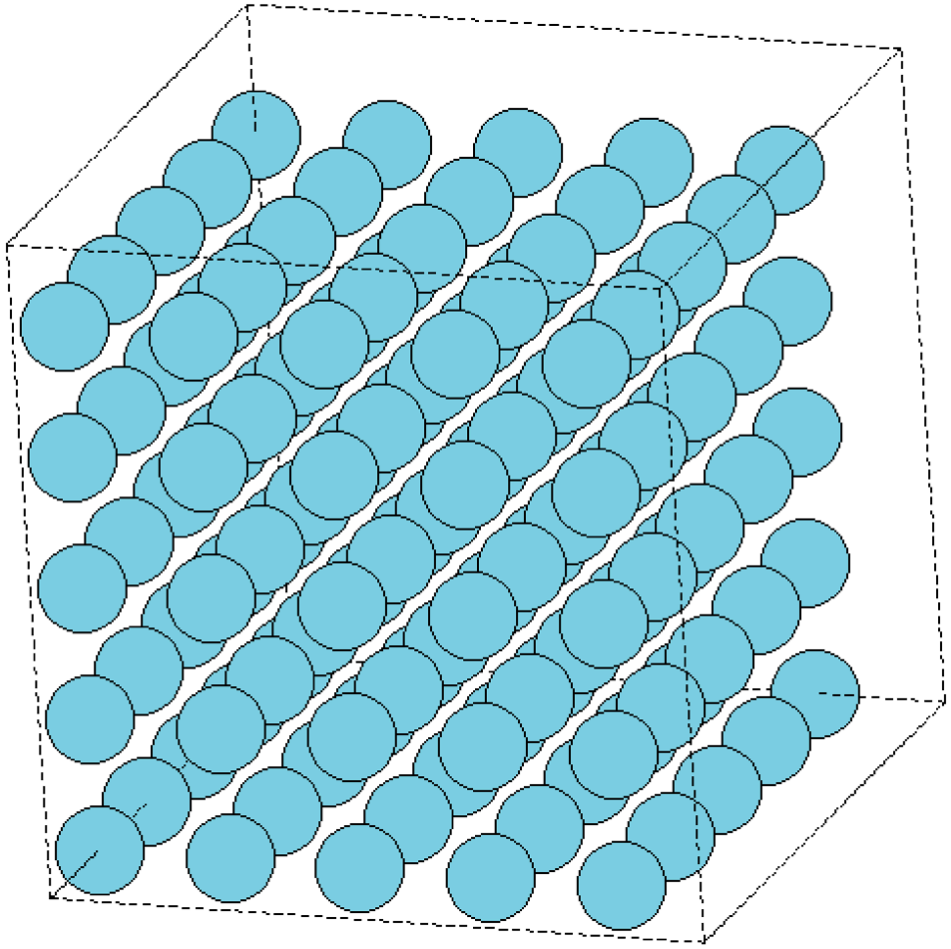
EXAMPLE

```
from ase.visualize import view
view(myatoms)


# Set up MyMD calculator and run
calc  = mymd.FileIOMyMD(label='mymd'
                        nsteps=10000,
                        dt=2.0,
                        nprint=100)
calc.run_md(myatoms)

# View final state of the atoms
view(calc.state)

# View trajectory of the atoms
trajectory = calc.frames.collect('at
view(trajectory)
```



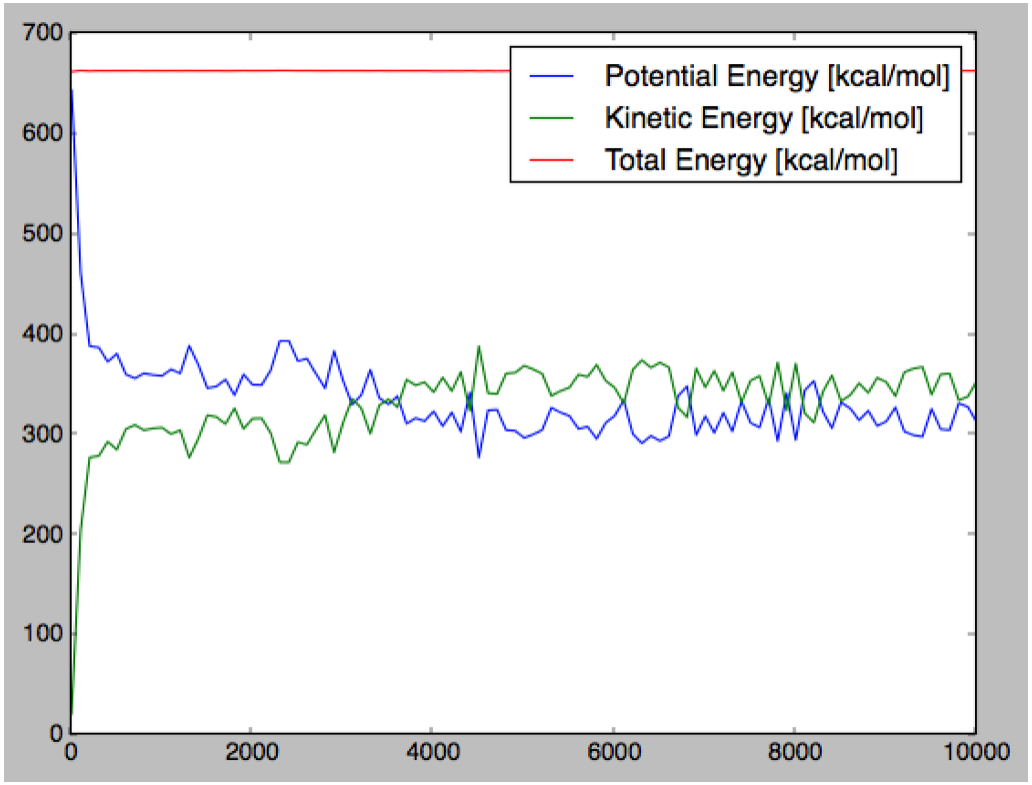
EXAMPLE

```
# Visualize data using matplotlib
import matplotlib.pyplot as plt

n    = calc.frames.collect('index')
ekin = calc.frames.collect('ekin')
epot = calc.frames.collect('epot')
etot = calc.frames.collect('etot')

plt.plot(n,epot, label='Potential Energy [kcal/mol]')
plt.plot(n,ekin, label='Kinetic Energy [kcal/mol]')
plt.plot(n,etot, label='Total Energy [kcal/mol]')
plt.legend()

plt.show()
```



THANKS!

Check out code: git clone https://github.com/Rhouli/ljmd-c.git