ParmEd

Jason Swails, Amber Meeting 2016
Rafal Wiewiora, John Chodera

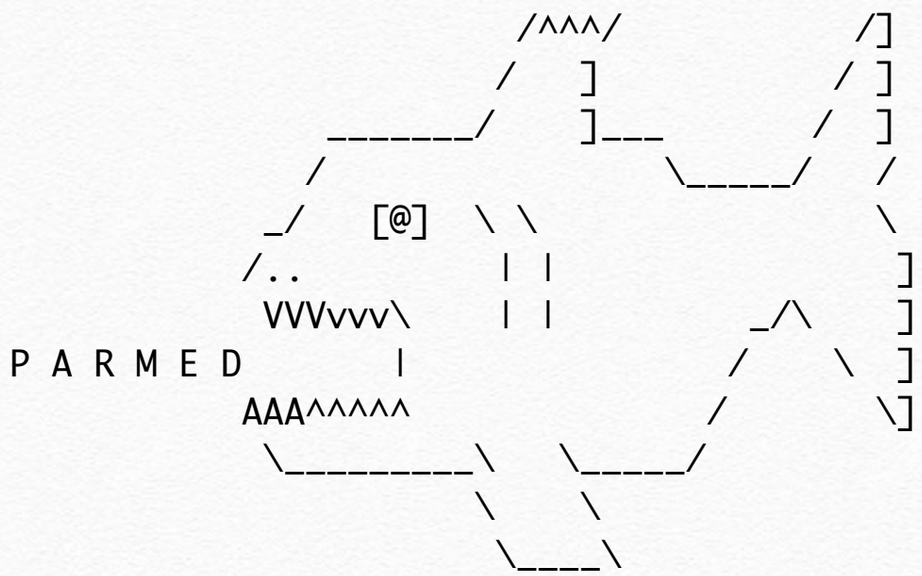
Outline

- What is it?
- Who uses it, and for what?
- What's new?

What is it?

Standalone programs

```
bash$ parmed
```




```
ParmEd: a Parameter file Editor


Reading input from STDIN...
> help

Documented commands (type help <topic>):
========================================
EOF                 changeRadii      minimize        quit
HMassRepartition    checkValidity    netCharge       scale
OpenMM              defineSolvent    outCIF          scee
add12_6_4           deleteBond       outPDB          scnb
addAtomicNumber     deleteDihedral   outparm         setAngle
[snip]
```

API

Easily instantiate and manipulate full MM descriptions of chemical systems.

Support wide range of force fields and potential energy functions.

Read and write dozens of file types from many programs

Dimensional Analysis

```
In [1]: import parmed.unit as u

In [2]: x = 10*u.kilocalorie_per_mole
        x
Out[2]: Quantity(value=10, unit=kilocalorie/mole)

In [3]: x.in_units_of(u.kilojoules_per_mole)
Out[3]: Quantity(value=41.84, unit=kilojoule/mole)

In [4]: k = 10 * u.kilocalories_per_mole / u.angstroms**2
        k
Out[4]: Quantity(value=10, unit=kilocalorie/(angstrom**2*mole))

In [5]: k.value_in_unit(u.kilojoules_per_mole / u.nanometers**2)
Out[5]: 4184.0

In [6]: (1/2*(1*u.grams)*(1*u.nanometers/u.second)**2).value_in_unit(
                    u.kilocalories)
Out[6]: 1.1950286806883367e-25
```

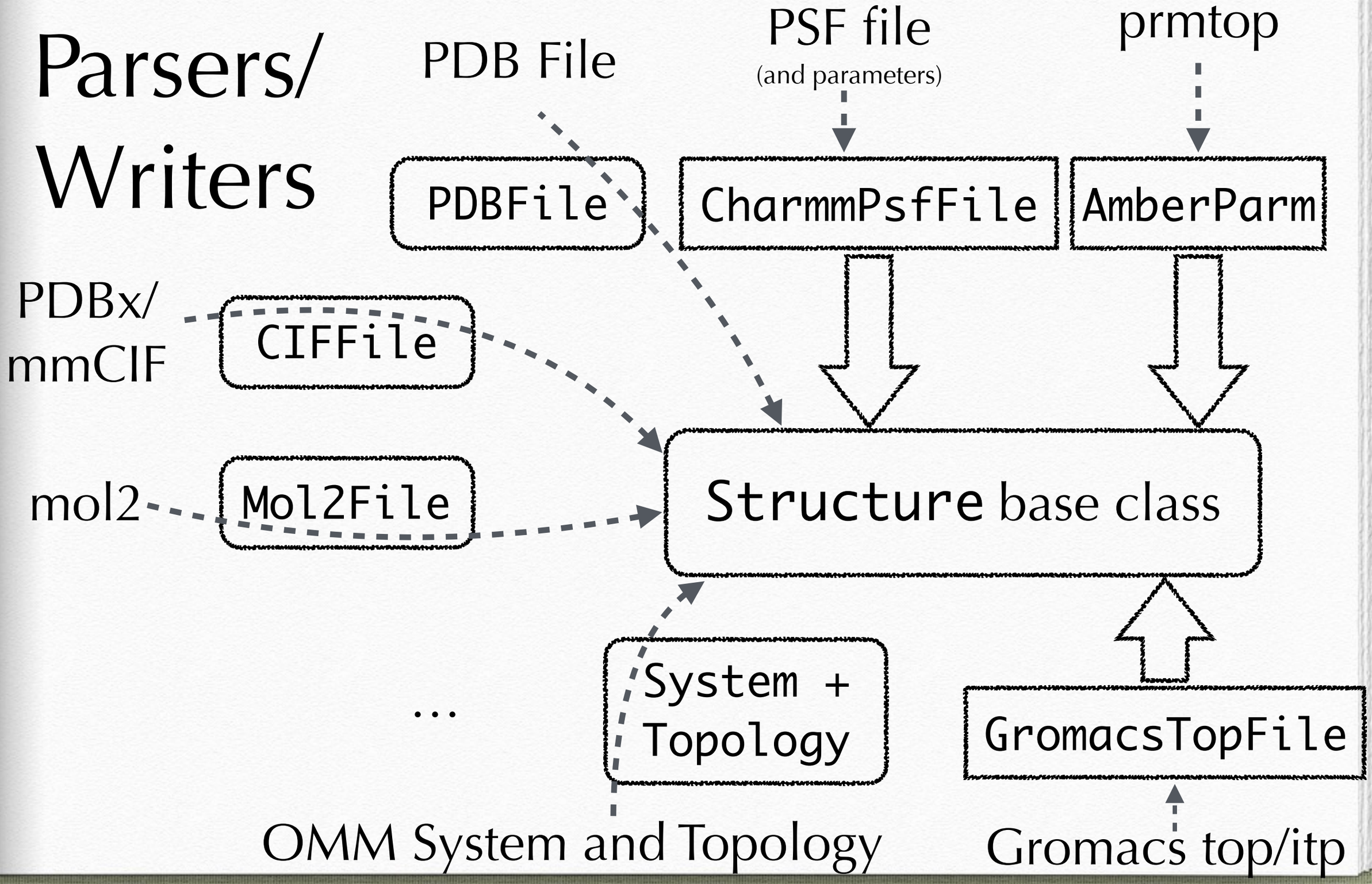
Dimensional Analysis

```
In [7]: (1/2*(1*u.grams)*(1*u.nanometers/u.second**2)).value_in_unit(
                u.kilocalories)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-7-164214fed064> in <module>()
      1 (1/2*(1*u.grams)*(1*u.nanometers/u.second**2)).value_in_unit(
----> 2             u.kilocalories)

/Users/swails/miniconda/envs/py35/lib/python3.5/site-packages/simtk/unit/quantity.py in value_in_unit(self, unit)
    619         Returns underlying value, in the specified units.
    620         """
--> 621         val = self.in_units_of(unit)
    622         if is_quantity(val):
    623             return val._value

/Users/swails/miniconda/envs/py35/lib/python3.5/site-packages/simtk/unit/quantity.py in in_units_of(self, other_unit)
    655         """
    656         if not self.unit.is_compatible(other_unit):
--> 657             raise TypeError('Unit "%s" is not compatible with Unit "%s".' % (self.unit, other_unit))
    658         f = self.unit.conversion_factor_to(other_unit)
    659         return self._change_units_with_factor(other_unit, f)

TypeError: Unit "gram*nanometer/(second**2)" is not compatible with Unit "kilocalorie".
```

Overhauled API

Parsers/ Writers



Supported File Formats

- Amber, chamber, and AMOEBA-style prmtops
- Amber restart/mdcrd
- Amber NetCDF restart and trajectory
- Amber parameter/frcmod/OFF/*leaprc* files
- PDB (with metadata)
- PDBx/mmCIF (with metadata)
- Mol2/Mol3
- CHARMM PSF
- CHARMM RTF, PRM, and STR files
- CHARMM coordinate and *restart* files
- NAMD binary coordinate and velocity files
- GROMACS topology/gro
- Many others

Simple Interface

- Automatic file-type detection upon read

```
In [3]: import parmed as pmd
        pmd.load_file('trx.prmtop')
Out[3]: <AmberParm 1654 atoms; 108 residues; 1670 bonds; parametrized>

In [4]: pmd.load_file('ala_ala_ala.psf')
Out[4]: <CharmmPsfFile 33 atoms; 3 residues; 32 bonds; NOT parametrized>

In [5]: pmd.load_file('tripos9.mol2')
Out[5]: <ResidueTemplate GPN: 34 atoms; 35 bonds; head=N1'; tail=C'>

In [6]: pmd.load_file('2koc.pdb')
Out[6]: <Structure 451 atoms; 14 residues; 0 bonds; PBC (orthogonal); NOT parametrized>

In [7]: pmd.load_file('4LZT.cif')
Out[7]: <Structure 1164 atoms; 274 residues; 0 bonds; PBC (triclinic); NOT parametrized>
```

Simple Interface

- Automatic file-type detection upon write

```
In [10]: import parmed as pmd
         pmd.load_file('trx.prmtop', 'trx.inpcrd').save('sample.pdb', overwrite=True)

In [13]: psf = pmd.load_file('ala_ala_ala.psf')
         psf.coordinates = pmd.namd.NamdBinCoor.read('ala_ala_ala.coor').coordinates
         psf.save('sample.cif', overwrite=True)

In [15]: pmd.load_file('tripos9.mol2', structure=True).save('sample.psf')

In [17]: pmd.load_file('4LZT.cif').save('sample.pdb', overwrite=True)
```

Details of the conversion are automatically handled

Who uses it, and for what?

- sander Python API
- CHARMM-GUI
- Numerous Amber utilities
- OpenMolTools
- Phenix
- InterMol*
- Force Field converters

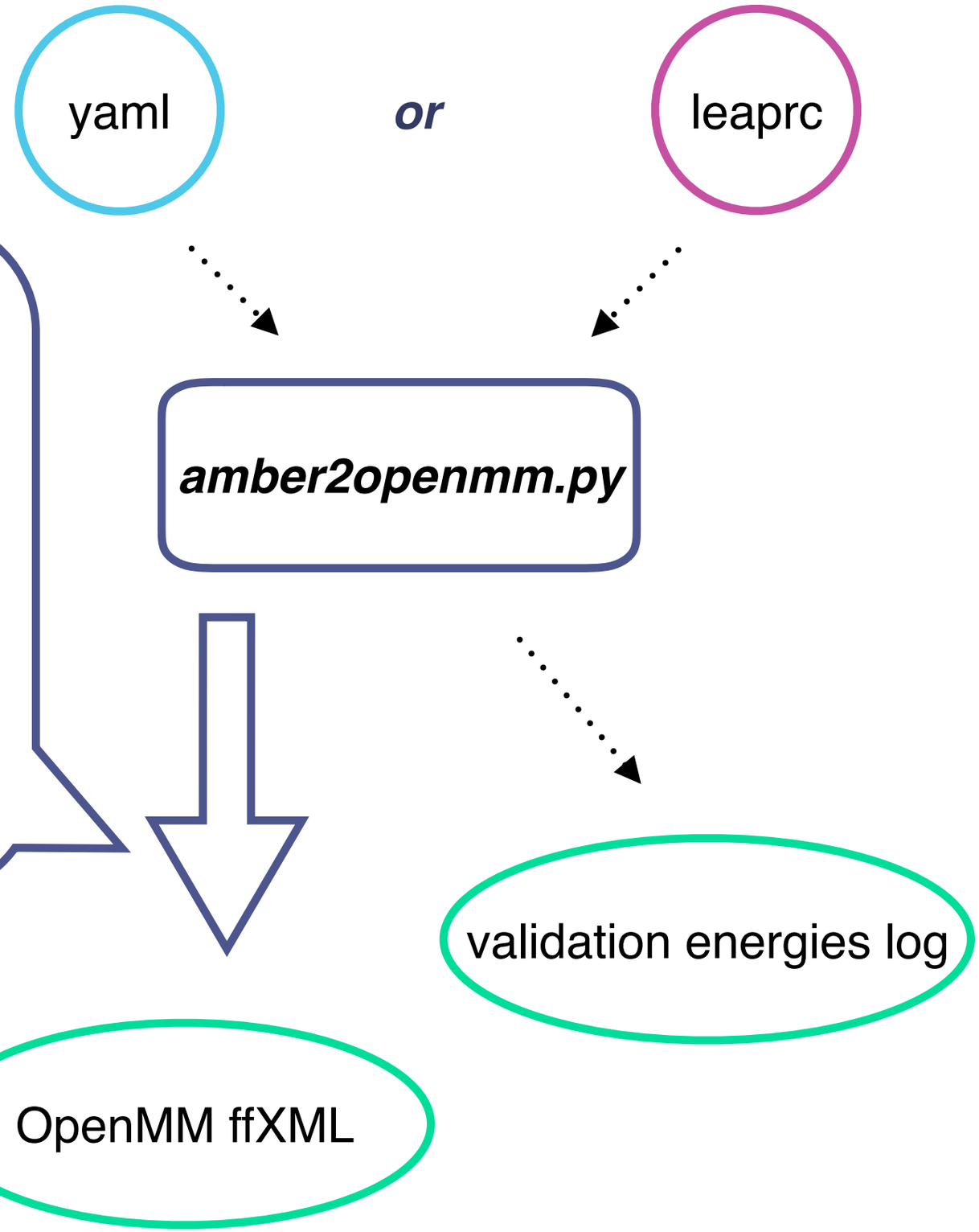
ParmEd as a forcefield converter

- "We" are currently working on porting the current Amber force fields to OpenMM.

- "We" ≈ Rafal Wiewiora

ParmEd-based Amber —>> OpenMM force-field conversion

- *yaml is a master-list for large conversions*
- *ParmEd does the conversion itself - very simple*
- *Extensive energy validations for proteins, nucleic acids, phosphorylated proteins, GAFF and water-ion systems*
- *Relative error in OpenMM energies asserted at 1e-5 tolerance (1e-2 for impropers)*



Rafal P. Wiewiora,
Chodera Lab, MSKCC
@rafwiewiora

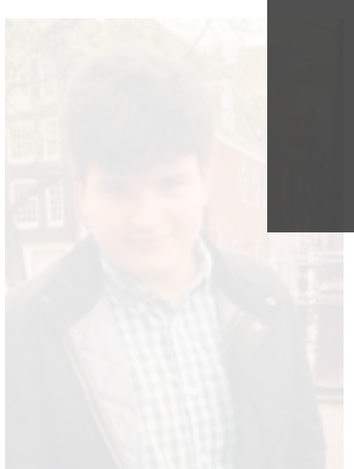
GitHub: for now see https://github.com/choderalab/openmm/pull/15
(to be included in OpenMM 7.1)

ParmEd-based Amber —>> OpenMM force-field conversion

```
# ParmEd - 3-line conversion
params = parmed.amber.AmberParameterSet.from_leaprc('leaprc.ff14SB')
params = parmed.openmm.OpenMMParameterSet.from_parameterset(params)
params.write('ff14SB.xml', provenance=provenance, write_unused=False)
```

- yaml is a master-list for large conversions
- ParmEd does the conversion itself - very simple
- Extensive energy validations for proteins, nucleic acids, phosphorylated proteins, GAFF and other systems
- Relative error in OpenMM energies asserted at 1e-5 tolerance (1e-2 for impropers)

```
# YAML EXAMPLE
- sourcePackage: AmberTools
  sourcePackageVersion: 15
- MODE: LEAPRC
- Source: leaprc.ff14SB
  Reference:
  - >-
    Maier, J.A., Martinez, C., Kasavajhala, K., Wickstrom, L.,
    Hauser, K.E., and Simmerling, C. (2015).
    ff14SB: Improving the Accuracy of Protein Side Chain and
    Backbone Parameters from ff99SB. J. Chem. Theory Comput. 11, 3696-3713.
  Test:
  - protein
  - nucleic
```

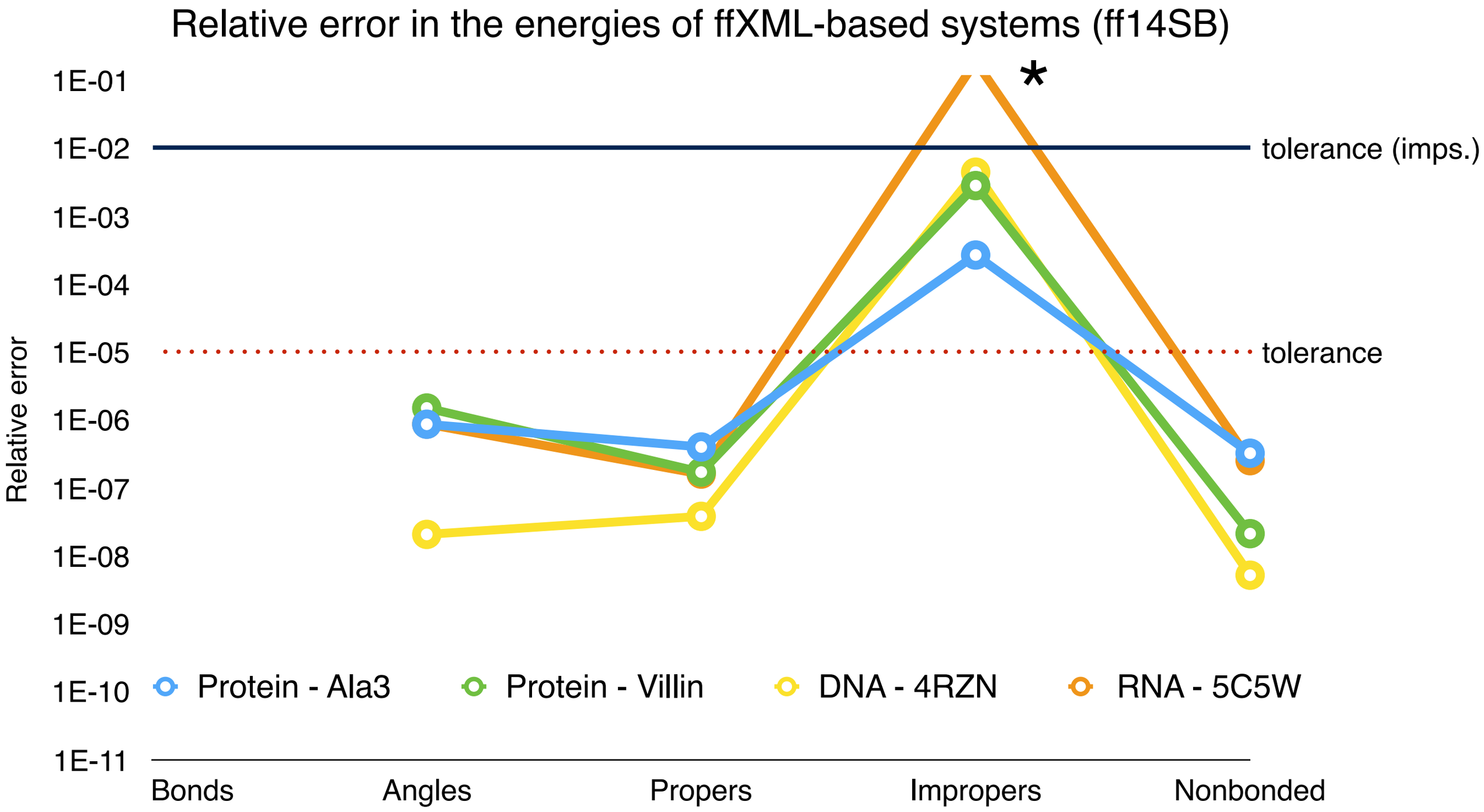


Rafal P. Wiewiora,
Chodera Lab, MSKCC
@rafwiewiora

GitHub:

(to be included in OpenMM 7.1)

ParmEd-based Amber —>> OpenMM force-field conversion
Energy validation



* High relative errors in the impropers are apparently due to a limitation in OpenMM's understanding of how LEaP assigns the order of atoms. Particularly high energy differences are seen in RNA. @rafwiewiora is currently addressing the problem.

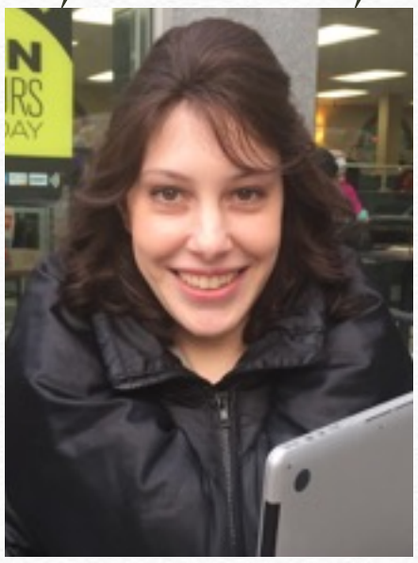
What's new?

- GROMACS support (**`gromber`**)
- Convenience loading/saving functions
- Python 2.7+ with numpy are now requirements
- Reorganized API

Acknowledgements

Chodera's lab; MSKCC, New York, New York

Rafal Wiewiora



Chaya Stern

John Chodera

- Lee-Ping Wang (GROMACS)
- Hai Nguyen
- Christoph Klein
- Dave Case
- David Mobley
- Michael Shirts
- Justin Lemkul (CHARMM validation)
- Wonpil Im and Jumin Lee (CHARMM-GUI)
- Carlos Hernandez (Rosetta)