PyRosetta:
A Workshop

What is it?

- Python bindings for Mini
- Allows a user to interact with and extend Rosetta using Python – one of the best languages we have today
- Python is widely used by scientists!

Benefits

- Line by line interactivity – absolutely great for new users
- On the fly object inspection – great for debugging
- Tab completion – great for writing code faster and learning the structure of objects
- No compilation, no waiting, rapid prototyping of new protocols

Demo

- First time use of PyRosetta
- start iPython session and initialize Rosetta
- Create a pose, scorefxn, view score
- Create a mover in PyRosetta

Creating PyRosetta Build Environment

Dependencies

- PyGCCXML
  - GCCXML
    - need to modify CXX flags
    - Cmake
- Pyplusplus
- Boost (libboost_python-xgcc*)

How to set up a build environment for PyRosetta

- pyrosetta.org/pyrosetta_build_scripts.tgz
- 2 scripts - create environment & build
- Create env: (create_pyrosetta-env.sh)
  - requirements:
    - wget, cvs, svn
  - Sets up local installation of the PyRosetta Dependencies
- Build
  - move into mini/src/python/bindings and execute
    - sh build_with_options.sh

Adding new mini C++ functions to PyRosetta

- In src/python/bindings look up files: **IncludeDict** and **IncludeDict.new** and find source files that you want to add.

- **IncludeDict.new** (auto generated file) – list of files that were just recently added to mini and were never tested with PyRosetta. ← ***by default automatically excluded from PyRosetta build.***

- **IncludeDict** – list of mini files/dirs + include/exclude flags for PyRosetta:

…

'core' : (True, 999, []),

'core/pose' : (True, 999, []),

'core/pose/Pose.hh' : (True, 999, []),

'core/pose/util.hh' : (True, 999, []),

'core/pose/PDBPoseMap.hh' : (True, 999, []),

…

For your files change the include/exclude flag to ‘True’:

…

‘protocols' : (True, 999, []),

'protocols/MyProtocol' : (False, 999, []),

'protocols/MyProtocol/MyFile.hh‘ : (False, 999, []),

…

…

‘protocols' : (True, 999, []),

'protocols/MyProtocol' : (True, 999, []),

'protocols/MyProtocol/MyFile.hh‘ : (True, 999, []),

…

Rebuild!

- Advanced procedure, exclude individual functions:

  src/python/bindings/exclude.py, find and modify function
  '**mb_exclude**(path, mb, hfile)'

- Advanced procedure, **Templates**: make them concrete and then bind!

PyRosetta: What's fast and what's not?

- Good news: All C++ functions run at **the same speed as mini** release build.
- Python code in general is slow. But if you work on code outside of your main loop – it does not matter!
- For Python: think **of amount of data** you need to process for each step. Small quantities (lists with ~<1,000 elements) is most likely ok.
- Python ←→ C++: Automatic conversion from Python list to C++ vector*<> classes will result in data copy. ➔If speed is a concern then use vector*<> type from the beginning (ie. PyRosetta bound vector*<> implementations).

Always remember the **GOLDEN RULE:**

PREMATURE OPTIMIZATION IS THE ROOT OF ALL EVIL

To speed up PyRosetta development we want:

- Permission to use current mini development sources for PyRosetta public release.

  Currently we bind mostly ‘core’ and only mature part of ‘protocols’ lib, we have no plan’s to bind ‘devel’ for PyRosetta. We release only binary builds, so no source code will go to public.