

Psi4: THEN AND NOW

ANDREW C. SIMMONETT

JUSTIN M. TURNER

LORI A. BURNS

DEVELOPERS MEETING, UGA

4 NOVEMBER 2016

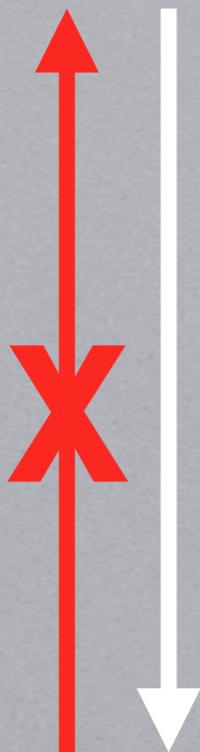
PART I: CONTRIBUTING TO Psi4

ANDY C. SIMMONETT

USING GITHUB'S PULL REQUESTS

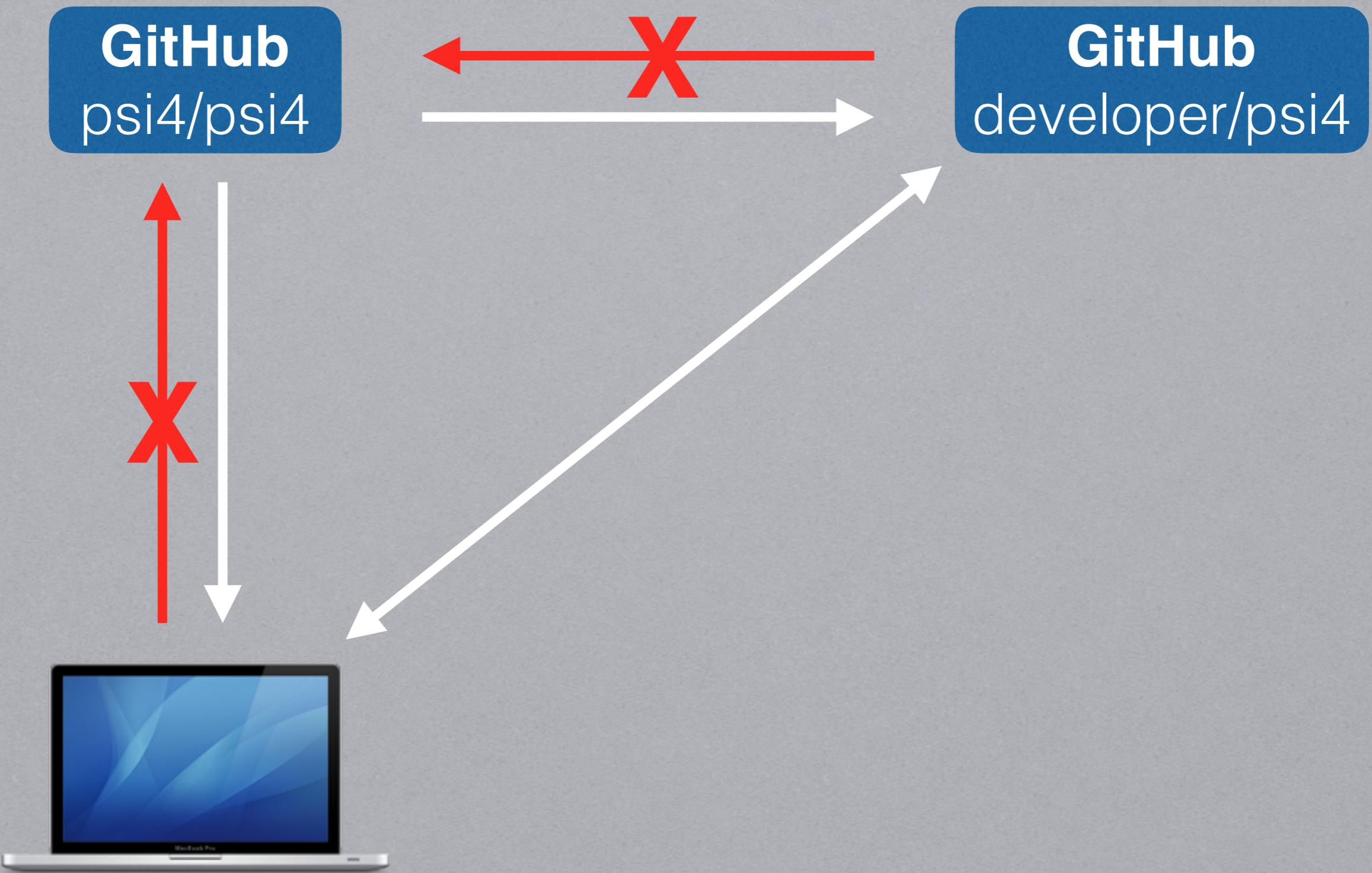
DIRECT PUSHES TO MASTER NO LONGER ALLOWED

GitHub
psi4/psi4



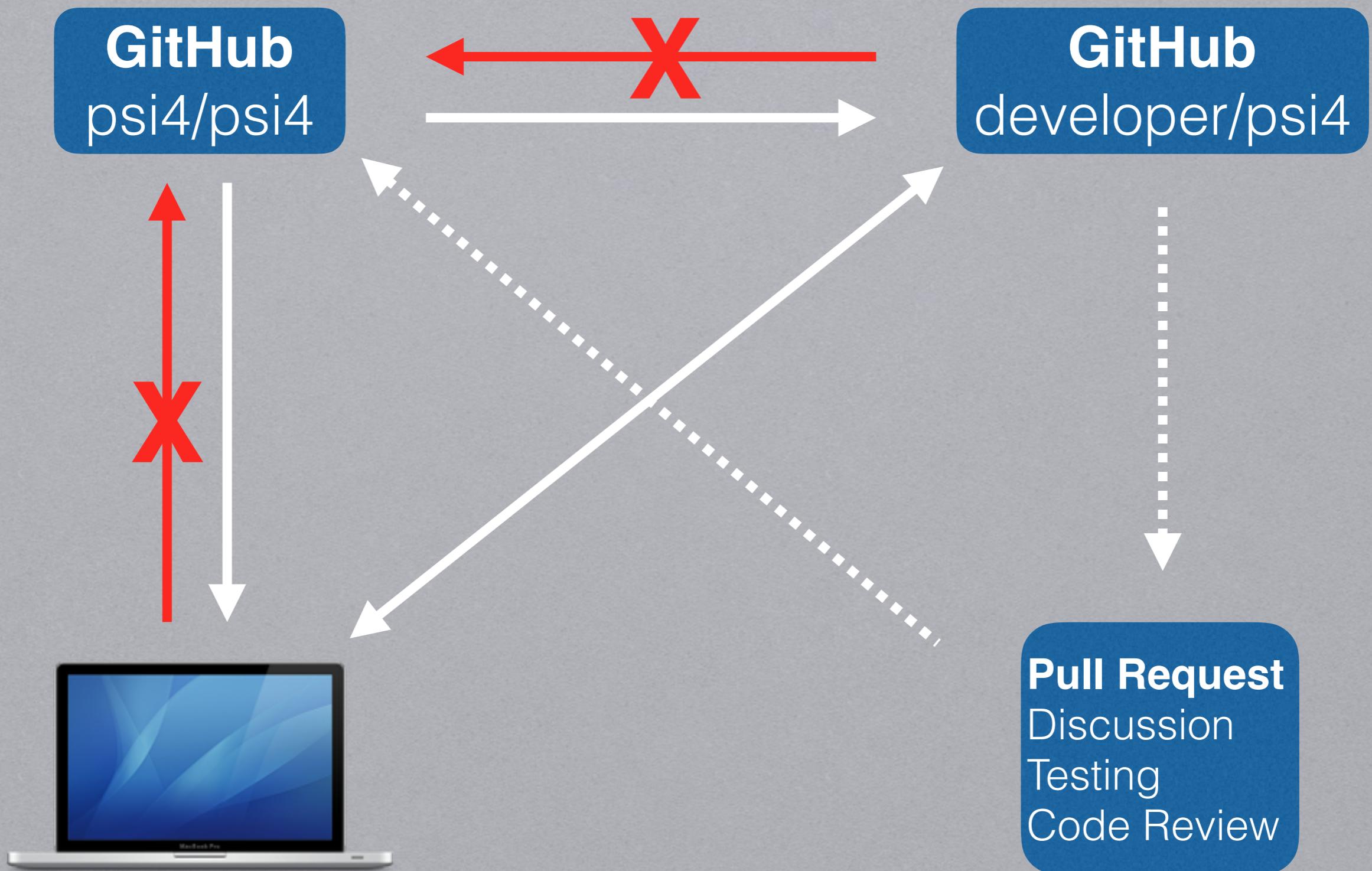
USING GITHUB'S PULL REQUESTS

DIRECT PUSHES TO MASTER NO LONGER ALLOWED

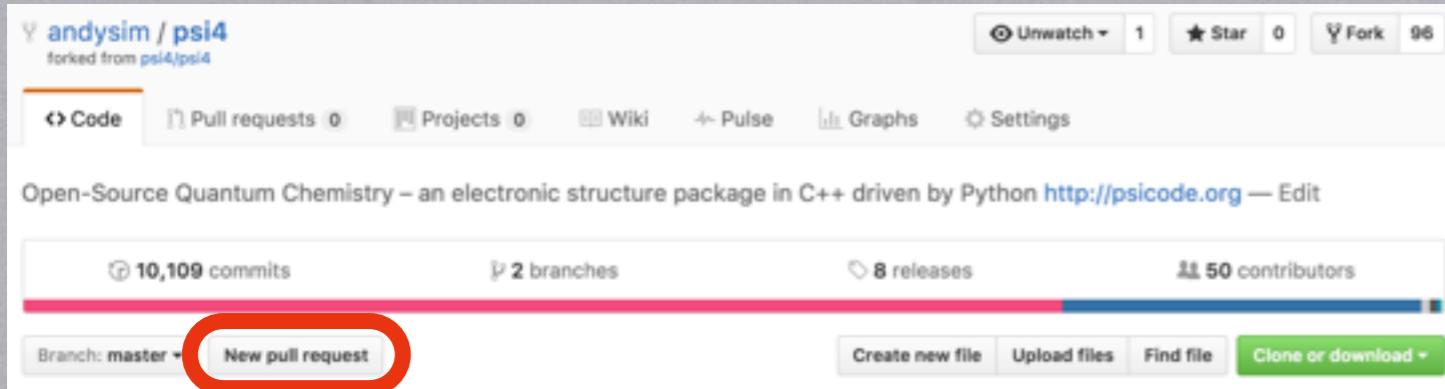


USING GITHUB'S PULL REQUESTS

DIRECT PUSHES TO MASTER NO LONGER ALLOWED



ISSUING A PULL REQUEST FROM YOUR PERSONAL GITHUB HOMEPAGE



The PR info screen

Launching a PR

A screenshot of a GitHub pull request (PR) details page. At the top, it shows the repository information: 'andysim commented on Jul 5 • edited' and 'psi4 member'. Below this, the 'Description' section contains the text: 'Adds analytic RHF Hessians. Still much cleanup/fixing to be done, but I thought I'd open a PR to allow others to chime in.' The 'Todos' section lists four items, all of which are checked: 'Add tests.', 'Hook into driver (with help from @loriab).', 'Add batching to allow fock derivatives to be computed without using too much memory.', and 'Fix bug in spherical harmonic second derivative integrals.' The 'Questions' section contains one question: 'Should we just merge this in when UHF is working, or should we also push to get the XC Hessian terms, @dgasmith / @robparrish ? I don't know how involved the XC terms will be, but I'm happy to help get them cranked out.' The 'Status' section has one item checked: 'Ready to go'.

INTERACTIVITY OF PULL REQUESTS

GETTING FEEDBACK FROM OTHER DEVELOPERS

A screenshot of a pull request review interface. The top bar shows the file path "cmake/Psi4Macros.cmake". Below the file path, there is a code diff with several changes highlighted in red and green. A blue button with a plus sign is visible on line 64. The right side of the interface has standard review buttons: "View", "Comment", and "Edit".

```
3  cmake/Psi4Macros.cmake
@@ -60,14 +60,15 @@ macro(psi4_add_module binlib libname sources)
 60  60
 61  61      # binary modules explicitly compiled into psi4.so
 62  62      if(${binlib} MATCHES bin)
 63  -      set_property(GLOBAL APPEND PROPERTY LIBLIST ${libname})
 63  +      set_property(GLOBAL APPEND PROPERTY BINLIST ${libname})
 64  64 + endif()
 65  65
 66  66      set(depend_name "${ARGN}")
 67  67      foreach(name_i IN LISTS depend_name)
 68  68          target_link_libraries(${libname} PRIVATE ${name_i})
 69  69      endforeach()
 70  70      target_link_libraries(${libname} PRIVATE pybind11::pybind11)
 71  71 + target_link_libraries(${libname} PRIVATE ${LAPACKBLAS_LIBRARIES})
 71  72  endmacro()
```

The review screen

A screenshot of a pull request conversation screen. At the top, there is a commit message: "fix empty strings preventing dft from printing for the last year and ..." with a checkmark and the ID "1f2f0da". Below the commit message, there are two comments:

- dgasmith commented on Mar 18**: "@ryanmrichard Is this something that can be fixed with the parallel printer so that we might avoid this in the future?"
- dgasmith commented on the diff on Mar 18**:
src/lib/libfunctional/functional.cc
View full changes
... ... @@ -38,8 +38,8 @@ void Functional::common_init()
38 38 gpa_ = false;
39 39 meta_ = false;
40 40 name_ = "";

Below the code snippet, there are two more comments:

- dgasmith on Mar 18**: "psi4 member" "Should we add a default of "" here also? Names should be set, but is not guaranteed."
- lorlab on Mar 18**: "psi4 member" "It'd be pretty hard to build a functional and use it w/o setting names. But out of an abundance of caution, some spaces would probably be good."

At the bottom, there is a "Reply..." button.

The conversation screen

AUTOMATIC CODE TESTING

SEEING WHICH COMMITS BREAK THE TESTS

The screenshot shows a GitHub pull request interface. At the top, a comment from **dgasmith** is visible, stating "Rebased through USAPTO changes." Below this, a list of commits by **loriab** and others is shown, each with its commit message, author, date, and a color-coded status indicator (green checkmark for passing, red X for failing). A detailed list of these commits follows:

- update samples a9e3334
- patch up docs build for inversion b6b90fe
- de-magic methods 8d722c0
- INV: Started pure python test cases 8dfa552
- INV: Fixed travis, adds explicit occupation test cases 29fcb9d
- INV: Helps to add the changes to runtest 0266429
- initial stab at atomic basis sets for SAD 4da92c6
- mod atomic basis for mints2, x2c1 788a4ee
- INV: SAD guess should be working again. 31ddbc0
- INV: Removed old C-side pyconstruct members and fixed datadir input b6e44ef

At the bottom of the list, another comment from **dgasmith** says: "Getting pretty close to merging this in. Failing tests case should be limited to the following:" followed by a list of failing test cases: 22 - cc14 (Failed), 155 - docs-psimod (Failed), 187 - mints9 (Failed).

Text at the very bottom states: "Docs and mints both require changes to the test case. cc14 remains the only undiagnosed test failure."

CONTINUOUS INTEGRATION TOOLS

TRAVIS / DISTELLI / CONDA-BUILD

psi4 / psi4  build passing

Current Branches Build History Pull Requests > **Build #581** More options 

✓ Merge pull request #479 from jgonthier/USAPT
Open-shell SAPT0
Commit f23b5f1
Compare 7ed9a5f..f23b5f1
Daniel Smith authored GitHub committed

-o #581 passed
Elapsed time 50 min 50 sec
Total time 2 hrs 58 min 21 sec
13 days ago

Restart build

Build Jobs

✓ # 581.1	 Compiler: clang C++	 CXX_COMPILER='clang++-3.6' C_COMPILER='clang' 35 min 40 sec
✓ # 581.3	 Compiler: gcc C++	 CXX_COMPILER='g++-4.9' C_COMPILER='gcc-4.9' 43 min 20 sec
✓ # 581.4	 Compiler: gcc C++	 CXX_COMPILER='g++-6' C_COMPILER='gcc-6' For 50 min 48 sec
✓ # 581.5	 Compiler: gcc C++	 CXX_COMPILER='g++-6' C_COMPILER='gcc-6' For 47 min 57 sec

Allowed Failures ?

! # 581.2	 Compiler: clang C++	 CXX_COMPILER='clang++-3.9' C_COMPILER='clang' 36 sec
-----------	---	--

ACCEPTING/MERGING A PR AFTER TEAM APPROVAL

The screenshot shows a GitHub pull request interface. At the top, a green checkmark icon and the commit hash `84b8214` are visible. Below this, the title of the pull request is `CIWave: Quick MCSCF docs`. The pull request has received three LGTM comments from team members:

- andysim** commented 10 days ago: "I love the python driver and the extra docs. Fantastic stuff! LGTM"
- jturney** commented 10 days ago: "LGTM"
- loriab** commented 10 days ago: "LGTM"

At the bottom of the list of comments, there is a summary of the merge action:

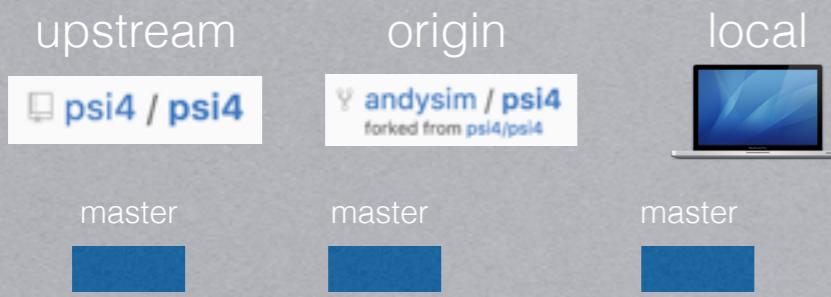
loriab merged commit `cdfd396` into `psi4:master` 10 days ago

Details of the merge check results:

- 3 checks passed
 - ✓ **approvals/lgtm** this commit looks good
 - ✓ **continuous-integration/Distelli** [Details](#)
 - ✓ **continuous-integration/travis-ci/pr** The Tra... [Details](#)

WORKFLOW FOR CONTRIBUTING

MAKING THE CHANGES/SETTING UP THE PR



1) `git clone git@github.com:andysim/psi4`
`git remote add upstream git@github.com:psi4/psi4`

WORKFLOW FOR CONTRIBUTING

MAKING THE CHANGES/SETTING UP THE PR

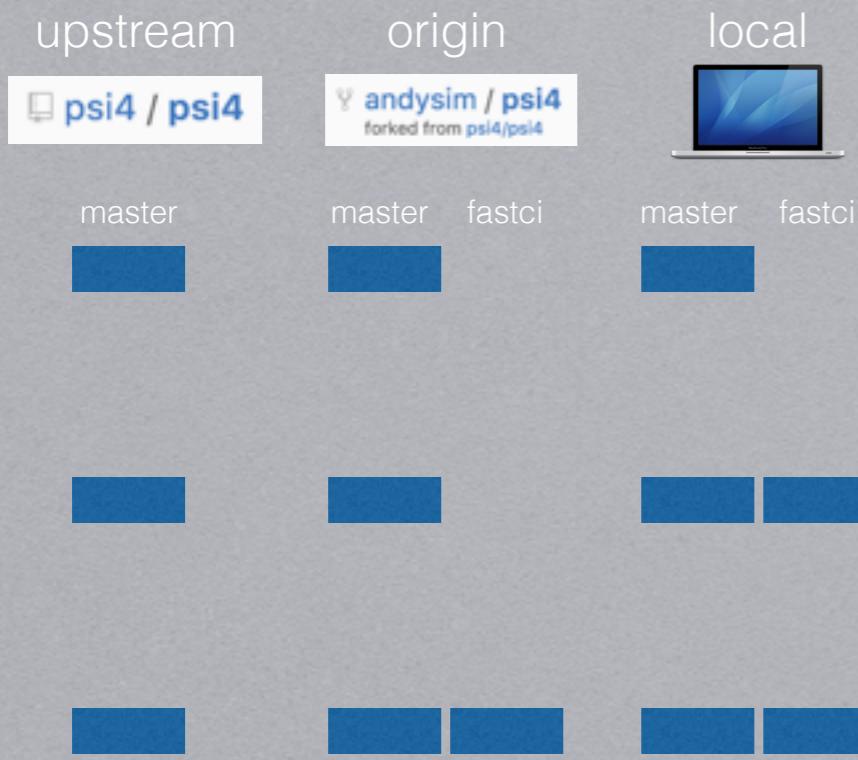


1) `git clone git@github.com:andysim/psi4`
`git remote add upstream git@github.com:psi4/psi4`

2) `git checkout -b fastci`

WORKFLOW FOR CONTRIBUTING

MAKING THE CHANGES/SETTING UP THE PR



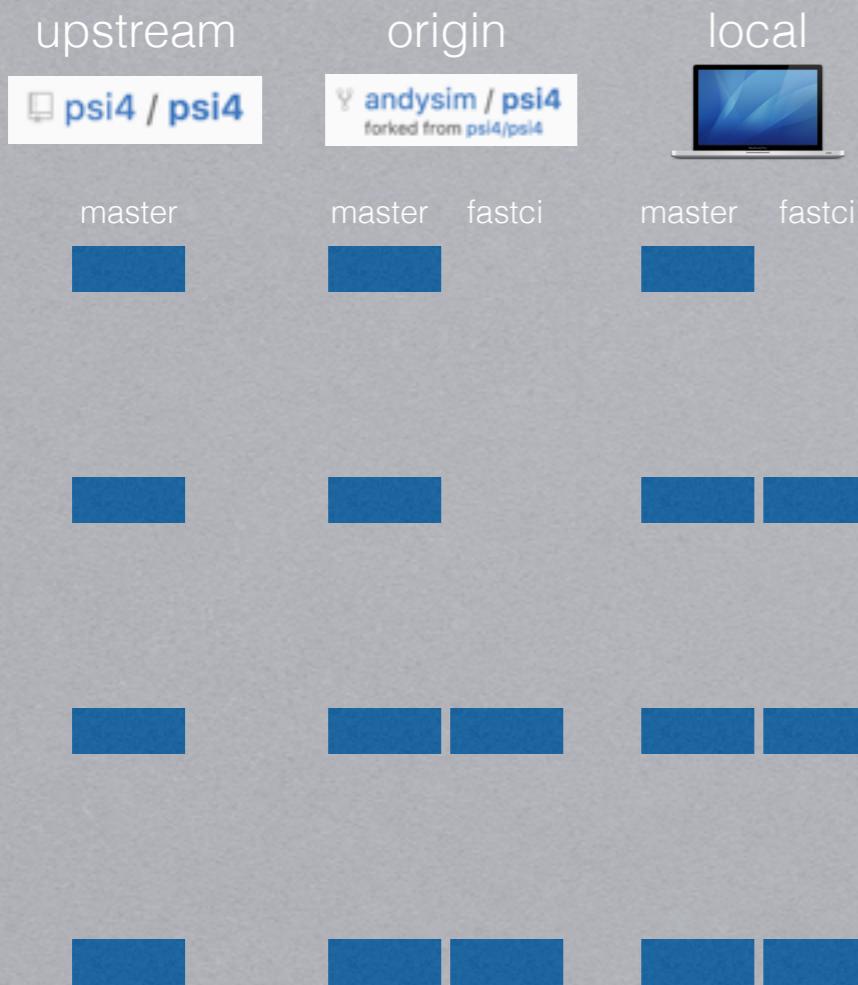
1) `git clone git@github.com:andysim/psi4`
`git remote add upstream git@github.com:psi4/psi4`

2) `git checkout -b fastci`

3) `git push origin fastci`

WORKFLOW FOR CONTRIBUTING

MAKING THE CHANGES/SETTING UP THE PR



1) `git clone git@github.com:andysim/psi4`
`git remote add upstream git@github.com:psi4/psi4`

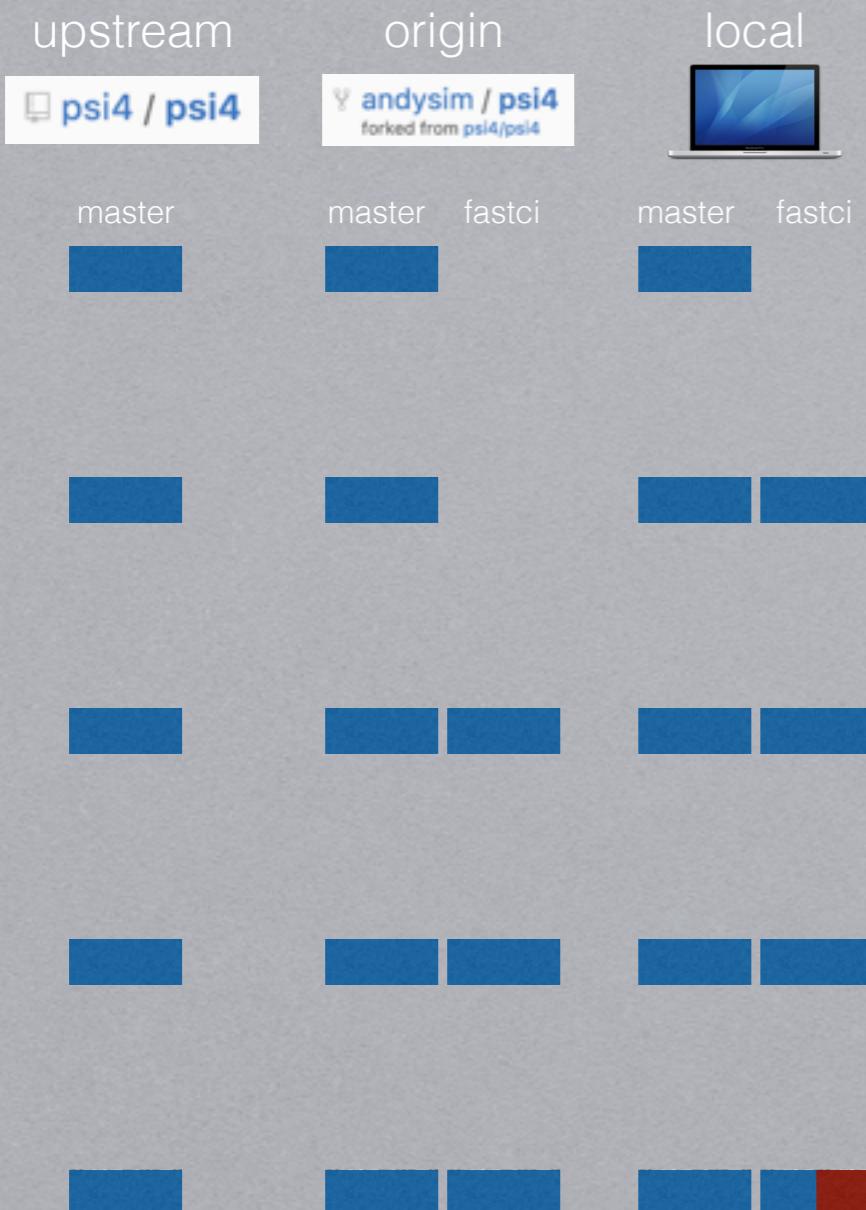
2) `git checkout -b fastci`

3) `git push origin fastci`

4) 

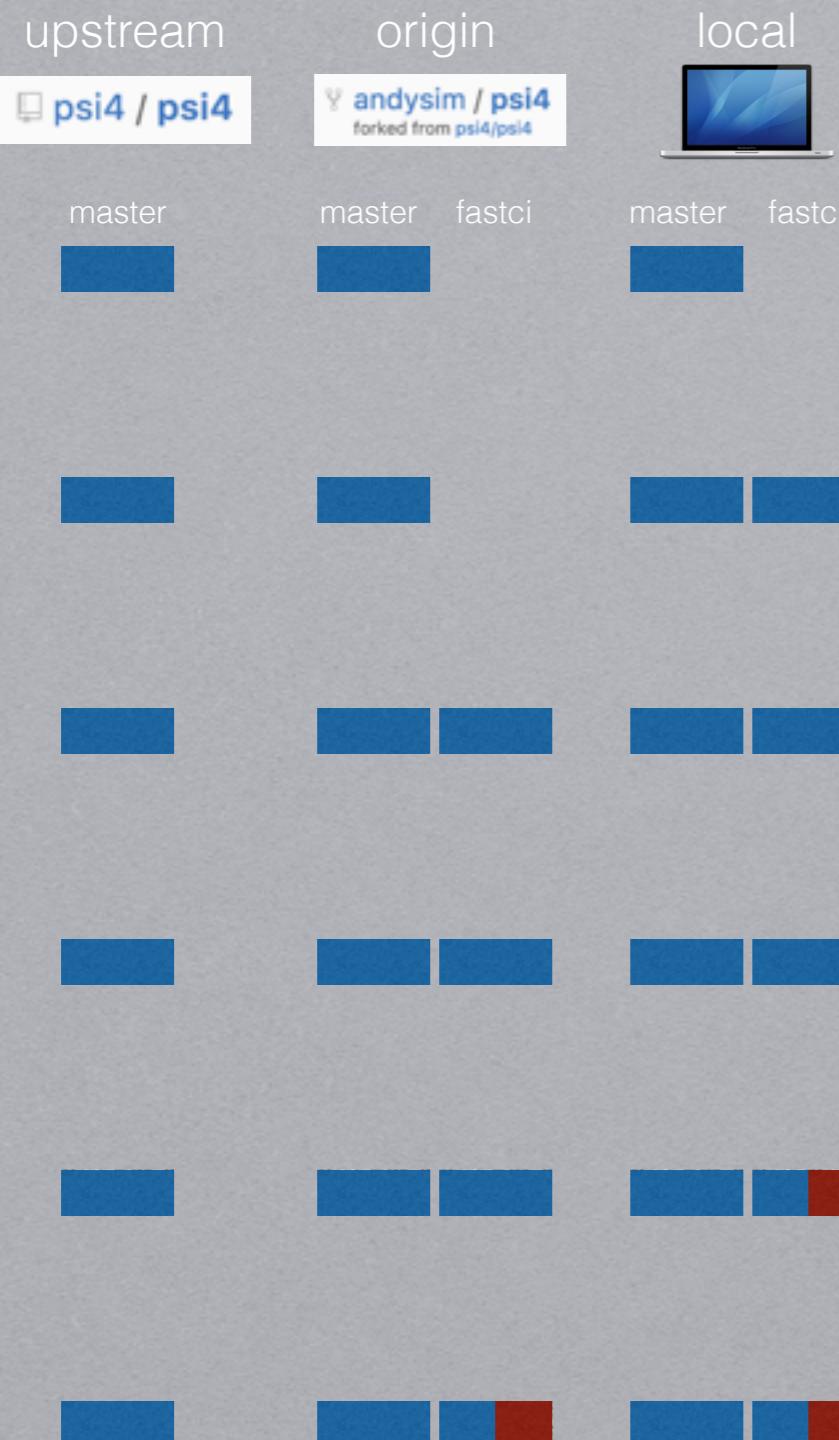
WORKFLOW FOR CONTRIBUTING

MAKING THE CHANGES/SETTING UP THE PR



WORKFLOW FOR CONTRIBUTING

MAKING THE CHANGES/SETTING UP THE PR



1) `git clone git@github.com:andysim/psi4`
`git remote add upstream git@github.com:psi4/psi4`

2) `git checkout -b fastci`

3) `git push origin fastci`

4)

5) Add new code on fastci branch

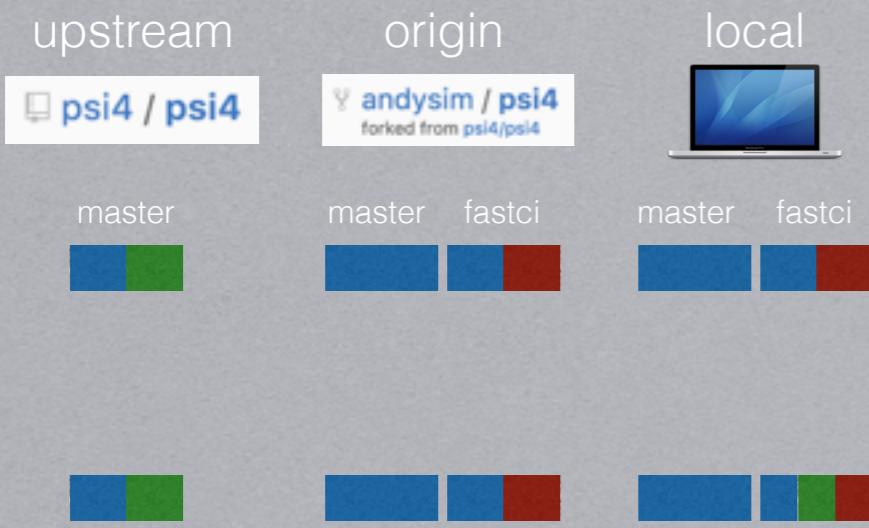
6) `git push origin fastci`

WORKFLOW FOR CONTRIBUTING SYNCHRONIZING / CLEANING REPOSITORIES



7) New feature added on master by somebody else (via a PR)

WORKFLOW FOR CONTRIBUTING SYNCHRONIZING / CLEANING REPOSITORIES

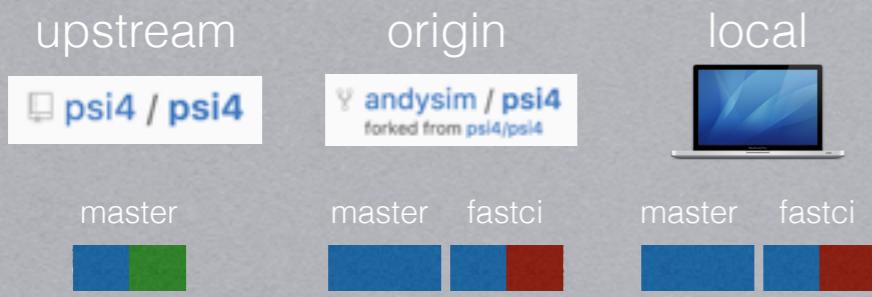


7) New feature added on master by somebody else (via a PR)

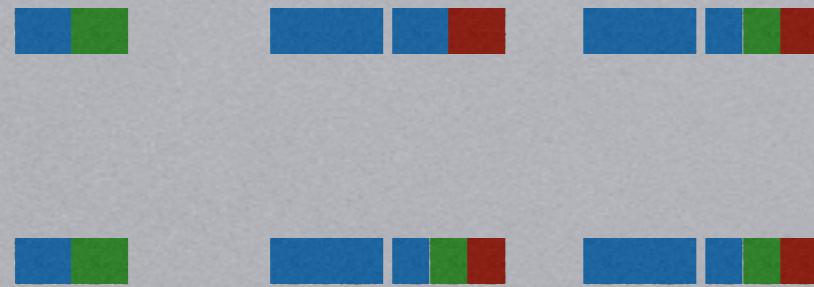
8) git pull --rebase upstream master

WORKFLOW FOR CONTRIBUTING

SYNCHRONIZING / CLEANING REPOSITORIES



7) New feature added on master by somebody else (via a PR)



8) git pull --rebase upstream master



9) git push origin fastci (may need to add --force)

WORKFLOW FOR CONTRIBUTING

SYNCHRONIZING / CLEANING REPOSITORIES



7) New feature added on master by somebody else (via a PR)



8) git pull --rebase upstream master

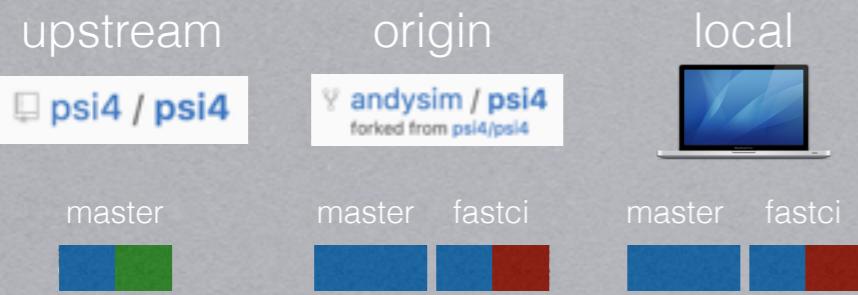


9) git push origin fastci (may need to add --force)



10) PR accepted, after “ready to go” checked, tests, LGTM approval

WORKFLOW FOR CONTRIBUTING SYNCHRONIZING / CLEANING REPOSITORIES



7) New feature added on master by somebody else (via a PR)



8) git pull --rebase upstream master



9) git push origin fastci (may need to add --force)



10) PR accepted, after “ready to go” checked, tests, LGTM approval

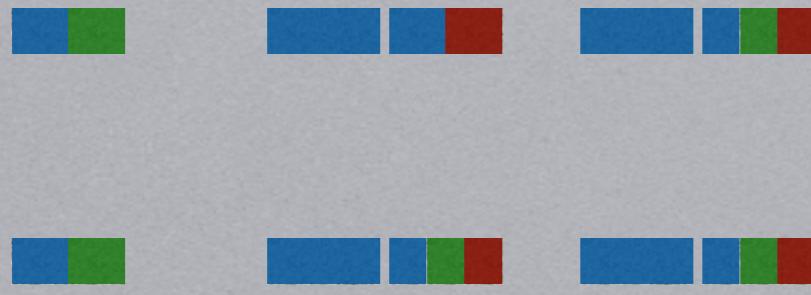


git checkout master
11) git pull --rebase upstream master
git push origin master

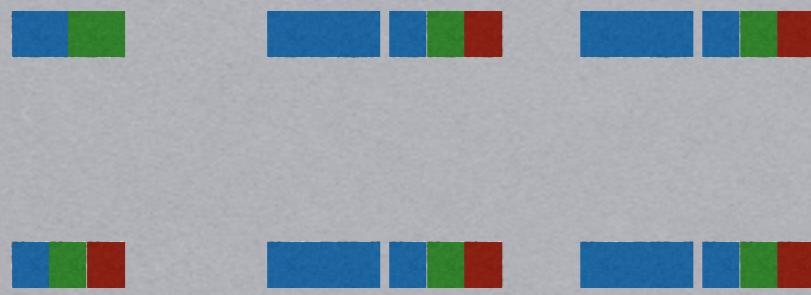
WORKFLOW FOR CONTRIBUTING SYNCHRONIZING / CLEANING REPOSITORIES



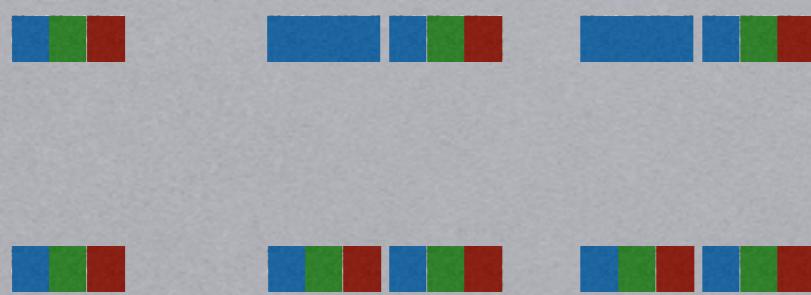
7) New feature added on master by somebody else (via a PR)



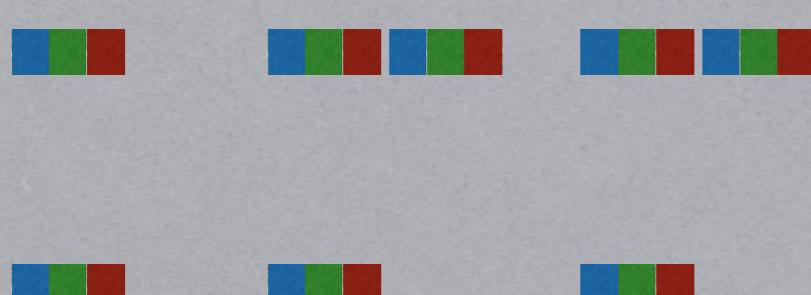
8) git pull --rebase upstream master



9) git push origin fastci (may need to add --force)



10) PR accepted, after “ready to go” checked, tests, LGTM approval



git checkout master
11) git pull --rebase upstream master
git push origin master



12) git branch -d fastci
git push origin :fastci

ADVANTAGES OF THE NEW WORKFLOW

- **For feature developers:-**
 - Feedback from the Psi4 team during development.
 - No fear of breaking the code for others.
 - Easy to see whether each commit breaks anything.
 - Intuitive for those familiar with other GitHub projects.
- **For the Psi4 team:-**
 - Easy way to track what people are working on.
 - Easy vetting / testing / merging of new contributions.
 - Code review means more consistent style.
- **For end users:-**
 - Code from GitHub should always work.

PART II: CONDA & BRANCHES

LORI A. BURNS

DOCUMENTATION REFRESH

HTTP://PSICODE.ORG/PSI4MANUAL/MASTER/INDEX.HTML

The screenshot shows a web browser window with the Psi4 logo and navigation bar. The main content is the 'INSTALLATION AND RUNTIME CONFIGURATION' page. The 'Obtaining Psi4' section is highlighted, and the 'Installing from Binary' sub-section is expanded, showing options like 'Binary Distribution' (Psi4conda Installer, Conda Proficients).

- API docs like **help(psi4)**
- NumPy docstrings in driver
- Sphinx (and Perl) only build dep

The screenshot shows a web browser window with the Psi4 logo and navigation bar. The main content is the 'PROGRAMMERS' MANUAL' page. It lists several sections: Contributions, Programming, Adding Add-Ons, and Psi4 API.

- Contributions: Intro to Programming in Psi4
 - Plugins: Adding New Functionality to Psi4
 - Documentation
 - PsiPEP: Plans and Practices to Organize Psi4
- Programming: Using the Core Libraries
 - LibOptions: globals, locals, has_changed and all that
 - Adding Methods to Driver
- Adding Add-Ons
- Psi4 API: Linking C++ and Python
 - psi4.core Module
 - psi4.driver Package

DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS

<http://psicode.org/psi4manual/master/index.html>

HOW TO EDIT



WHAT TO EDIT

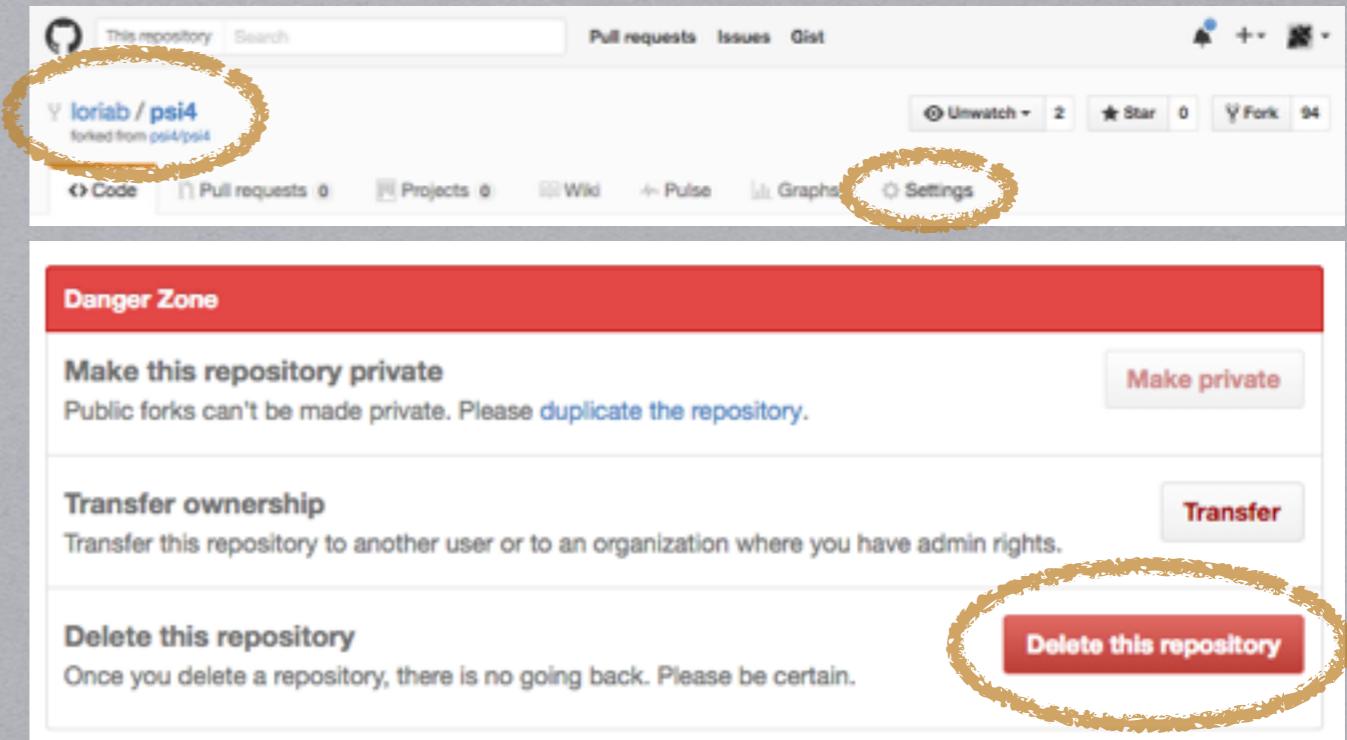
DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS

<http://psicode.org/psi4manual/master/index.html>

HOW TO EDIT

1. Delete any existing GH fork



WHAT TO EDIT



DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS

<http://psicode.org/psi4manual/master/index.html>

HOW TO EDIT



1. Delete any existing GH fork

2. Create a new fork, perhaps by editing a file online (no clone) from **docs/sphinxman/source/**

The screenshot shows a GitHub repository for "psi4 / psi4". The user is trying to edit the file "psi4 / doc / sphinxman / source / cubeprop.rst". A message says "You need to fork this repository to propose changes". A green button labeled "Fork this repository and propose changes" is highlighted with a yellow circle. On the right, a "Propose file change" dialog box is open, showing a "sample docs mod" and a "Propose file change" button at the bottom, also highlighted with a yellow circle.

DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS

<http://psicode.org/psi4manual/master/index.html>

HOW TO EDIT



1. Delete any existing GH fork
2. Create a new fork, perhaps by editing a file online (no clone) from **docs/sphinxman/source/**
3. Submit a pull request from a *branch* of your fork

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.

A screenshot of a GitHub comparison interface. It shows a dropdown menu for 'base fork' set to 'psi4/psi4', 'base' set to 'master', and 'head fork' circled in orange set to 'cdsgroup/psi4'. Below it, a green button labeled 'Create pull request' is also circled in orange. A note says '✓ Able to merge. These branches can be automatically merged.' and a link to 'Discuss and review the changes in this comparison with others.'

WHAT TO EDIT

DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS

<http://psicode.org/psi4manual/master/index.html>

HOW TO EDIT

1. Delete any existing GH fork
2. Create a new fork, perhaps by editing a file online (no clone) from **docs/sphinxman/source/**
3. Submit a pull request from a *branch* of your fork
4. If conflicts arise, clone and rebase to resolve them, independently or with advice



WHAT TO EDIT

DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS

<http://psicode.org/psi4manual/master/index.html>

HOW TO EDIT

1. Delete any existing GH fork
2. Create a new fork, perhaps by editing a file online (no clone) from **docs/sphinxman/source/**
3. Submit a pull request from a *branch* of your fork
4. If conflicts arise, clone and rebase to resolve them, independently or with advice
5. Once PR accepted, collect your sticker sheet

WHAT TO EDIT



DOCUMENTATION CHALLENGE

A PLOY TO GET RID OF STALE FORKS

<http://psicode.org/psi4manual/master/index.html>

HOW TO EDIT



1. Delete any existing GH fork
2. Create a new fork, perhaps by editing a file online (no clone) from **docs/sphinxman/source/**
3. Submit a pull request from a *branch* of your fork
4. If conflicts arise, clone and rebase to resolve them, independently or with advice
5. Once PR accepted, collect your sticker sheet

WHAT TO EDIT

- Anything that has confused you in narrative docs
- Any broken link
- Any docstring in driver
- Boost used 1-indexing for args, while Pybind11 uses 0-indexing, so many docstrings need adjusting in **psi4/src/core.cc**

2015		binary	docs
Tool	Linux	conda-build	Sphinx
Resource	Linux	psinet	psinet
Matrix	Linux	1 generic comp/gnu, py 2	
When		nightly crontab at 2AM	nightly crontab at 2AM
Runs		full tests AM 6; all must pass	full code scan all must build
Produces		Psi4 core conda package anaconda.org/psi4/psi4/files	Narrative & API Docs for Users & Py psicode.org/psi4manual/master/

PSINET (AKA. SPINET)



MACPSINET



2016		CI	binary	installer	stats	docs
Tool	L	Travis	conda-build	conda constructor	Codecov	Sphinx
Resource	M	Distelli				
Matrix	L	5 comp/gnu, py 2	psinet	psinet	NYI, switch fr/mac	psinet
Matrix	M	1 comp/gnu, py 2				
When		commit every latest commit	nightly crontab at 2AM	occasionally manually at new tag	occasionally manually	nightly crontab at 2AM
Runs		quicktests AM 4	full tests AM 6; all must pass	full tests AM 6; all must pass	full+processing AM 5	
Produces		Pass/Fail notice on GitHub commit github.com/psi4/psi4/pulls	Psi4 core conda package anaconda.org/psi4/psi4/files	Installer: Miniconda + P4core + Addons psicode.org/downloads2.html	Coverage Dash- board, Sunburst codecov.io/gh/psi4/psi4	Sphinx & Doxygen Autodocumenting psicode.org/psi4manual

POST-PROCESSING

DFTD3 Grimme

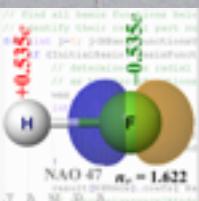
dispersion correction
Fortran executable

GCP Grimme

small-basis correction
Fortran executable

GDMA Stone

multipole analysis
Fortran library



JANPA Nikolaienko

NBO bond analysis
Java executable

resp2 McGibbon

RESP charges
C++ Psi4 plugin

UPSTREAM

CHARMMing Woodcock

QM/MM web server
Python/HTML executable; calls Psi4



Pulsar Pritchard & Richards

datastructure interoperability
C++ framework; calls Psi4

WebMO Polik & Schmidt

GUI/web server
Perl/Java executable; calls Psi4



QC METHODS

Ambit Turney

tensor manipulations
Python/C++ library

Cfour Stanton & Gauss

$\langle CC|CC \rangle$ CC/MBPT/properties/etc.
Fortran executable

CheMPS2 Wouters

$\bigtriangleup_{he[M][P][S]}^2$ DMRG theory
Python/C++ library

DKH Wolf & Reiher

relativistic correction
Fortran library

Gaussian



many-body expansion
Fortran executable

MRCC Kállay



arbitrary order CC/CI
Fortran executable

v2rdm_casscf DePrince

active-space SCF
Fortran/C++ Psi4 plugin

libxc Marques

exchange-correlation fnctls.
C library

INTEGRALS

ERD Flocke

2e⁻ integrals
Fortran library

libint Valeev

2e⁻ integrals
C library

simint Pritchard

2e⁻ integrals
C++ library

SOLVATION



ADF Visscher

FDE solvation
Fortran/C executable



libefp Kaliman & Slipchenko

fragment potentials
C library



PCMSolver DiRemigio

solvation
Fortran library

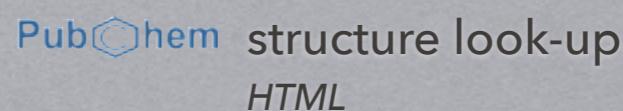
STRUCTURE & VIS



MOLDEN Schaftenaar

orbital/density visualization
Fortran executable

PubChem NIH



structure look-up
HTML

POST-PROCESSING

DFTD3 Grimme 

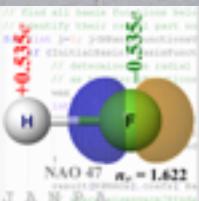
dispersion correction
Fortran executable

GCP Grimme

small-basis correction
Fortran executable

GDMA Stone 

multipole analysis
Fortran library



JANPA Nikolaienko

NBO bond analysis
Java executable

resp2 McGibbon

RESP charges
C++ Psi4 plugin

UPSTREAM

CHARMMing Woodcock

QM/MM web server
Python/HTML executable; calls Psi4

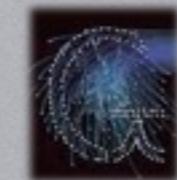


Pulsar Pritchard & Richards

datastructure interoperability
C++ framework; calls Psi4

WebMO Polik & Schmidt 

GUI/web server
Perl/Java executable; calls Psi4



Gaussian

many-body expansion
Fortran executable

MRCC Kállay 

arbitrary order CC/CI
Fortran executable

v2rdm_casscf DePrince 

active-space SCF
Fortran/C++ Psi4 plugin

libxc Marques

exchange-correlation fnctls.
C library

QC METHODS

Ambit Turney

tensor manipulations
Python/C++ library

Cfour Stanton & Gauss

$\langle CC|CC \rangle$ CC/MBPT/properties/etc.
Fortran executable

CheMPS2 Wouters 

$\langle he[M][P][S] \rangle_2$ DMRG theory
Python/C++ library

DKH Wolf & Reiher 

relativistic correction
Fortran library



Gaussian

many-body expansion
Fortran executable

MRCC Kállay 

arbitrary order CC/CI
Fortran executable

v2rdm_casscf DePrince 

active-space SCF
Fortran/C++ Psi4 plugin

libxc Marques

exchange-correlation fnctls.
C library

INTEGRALS

ERD Flocke

2e⁻ integrals
Fortran library

libint Valeev 

2e⁻ integrals
C library

simint Pritchard

2e⁻ integrals
C++ library

SOLVATION



ADF Visscher

FDE solvation
Fortran/C executable

libefp Kaliman & Slipchenko 

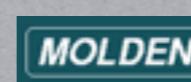
fragment potentials
C library



PCMSolver DiRemigio 

solvation
Fortran library

STRUCTURE & VIS



MOLDEN Schaftenaar

orbital/density visualization
Fortran executable

PubChem NIH 



structure look-up
HTML



Psi4 actively performs build and integration testing

POST-PROCESSING

DFTD3 Grimme 

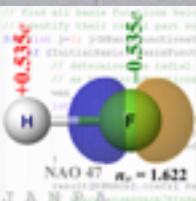
dispersion correction
Fortran executable

GCP Grimme

small-basis correction
Fortran executable

GDMA Stone 

multipole analysis
Fortran library



JANPA Nikolaienko

NBO bond analysis
Java executable

resp2 McGibbon

RESP charges
C++ Psi4 plugin

UPSTREAM

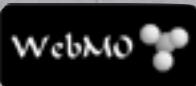
CHARMMing Woodcock

QM/MM web server
Python/HTML executable; calls Psi4



Pulsar Pritchard & Richards

datastructure interoperability
C++ framework; calls Psi4



WebMO Polik & Schmidt 

GUI/web server
Perl/Java executable; calls Psi4

QC METHODS

Ambit Turney

tensor manipulations
Python/C++ library

Cfour Stanton & Gauss

$\langle CC|CC \rangle$ CC/MBPT/properties/etc.
Fortran executable

CheMPS2 Wouters 

$\langle he[M][P][S] \rangle^2$ DMRG theory
Python/C++ library

DKH Wolf & Reiher 

relativistic correction
Fortran library

Gaussian



many-body expansion
Fortran executable

MRCC Kállay 



arbitrary order CC/CI
Fortran executable

v2rdm_casscf DePrince 

active-space SCF
Fortran/C++ Psi4 plugin

libxc Marques

exchange-correlation fnctls.
C library

INTEGRALS

ERD Flocke

2e⁻ integrals
Fortran library

libint Valeev 

2e⁻ integrals
C library

simint Pritchard

2e⁻ integrals
C++ library

SOLVATION



ADF Visscher

FDE solvation
Fortran/C executable



libefp Kaliman & Slipchenko 

fragment potentials
C library

PCMSolver DiRemigio 

solvation
Fortran library

STRUCTURE & VIS



MOLDEN Schaftenaar

orbital/density visualization
Fortran executable

PubChem NIH 



structure look-up
HTML



Psi4 actively performs build and integration testing



library and integrated into Psi4 CMake

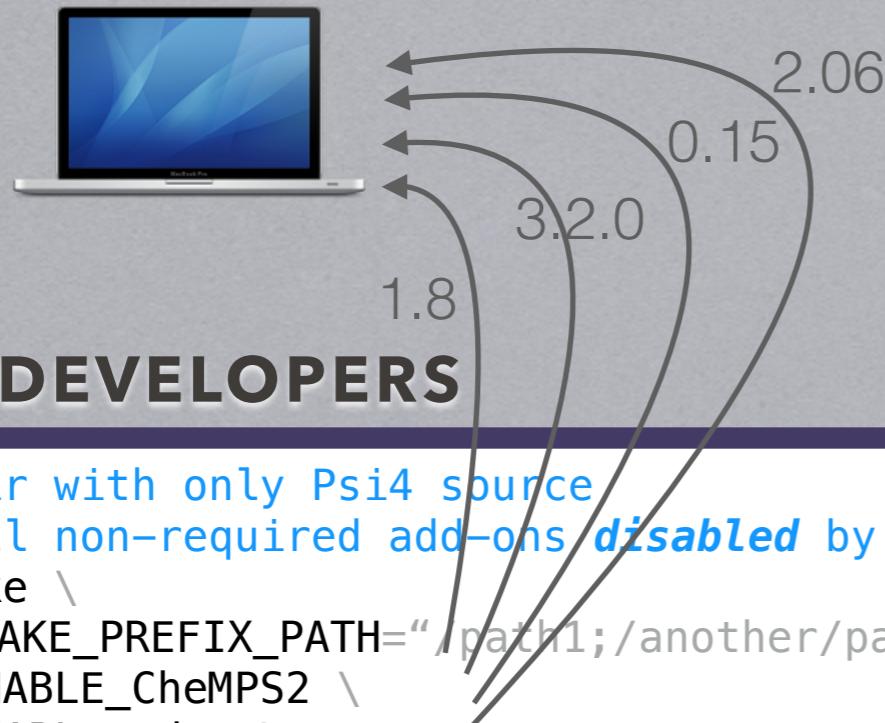
INTEGRATIONS FOR ALL

CORE DEVELOPERS

```
>>> # dir with only Psi4 source
>>> # all non-required add-ons disabled by default
>>> cmake \
    -DCMAKE_PREFIX_PATH="/path1;/another/path2" \
    -DENABLE_CheMPS2 \
    -DENABLE_gdma \
    ...
    # CMake downloads external projects' source from internet
>>> make
>>> ctest -L addons
Psi4 core..... PASSED
Psi4 + DFTD3 executable..... PASSED
Psi4 + CheMPS2 library..... PASSED
Psi4 + PCMSolver library..... PASSED
Psi4 + V2RDM_CASSCF plugin..... PASSED
Psi4 + Ambit library..... PASSED
Psi4 library..... PASSED
>>> # working Psi4 executable complete with add-ons, customized for local
hardware & software
```

INTEGRATIONS FOR ALL

seeking *built* in **CMAKE_PREFIX_PATH**



CORE DEVELOPERS

```
>>> # dir with only Psi4 source
>>> # all non-required add-ons disabled by default
>>> cmake \
    -DCMAKE_PREFIX_PATH="/path1;/another/path2" \
    -DENABLE_CheMPS2 \
    -DENABLE_gdma \
    ...
    # CMake downloads external projects' source from internet
>>> make
>>> ctest -L addons
Psi4 core..... PASSED
Psi4 + DFTD3 executable..... PASSED
Psi4 + CheMPS2 library..... PASSED
Psi4 + PCMSolver library..... PASSED
Psi4 + V2RDM_CASSCF plugin..... PASSED
Psi4 + Ambit library..... PASSED
Psi4 library..... PASSED
>>> # working Psi4 executable complete with add-ons, customized for local
hardware & software
```



INTEGRATIONS FOR ALL

seeking *built* in **CMAKE_PREFIX_PATH**



1.8

CORE DEVELOPERS

```
>>> # dir with only Psi4 source
>>> # all non-required add-ons disabled by default
>>> cmake \
    -DCMAKE_PREFIX_PATH="/path1;/another/path2" \
    -DENABLE_CheMPS2 \
    -DENABLE_gdma \
    ...
    # CMake downloads external projects' source from internet
>>> make
>>> ctest -L addons
Psi4 core..... PASSED
Psi4 + DFTD3 executable..... PASSED
Psi4 + CheMPS2 library..... PASSED
Psi4 + PCMSolver library..... PASSED
Psi4 + V2RDM_CASSCF plugin..... PASSED
Psi4 + Ambit library..... PASSED
Psi4 library..... PASSED
>>> # working Psi4 executable complete with add-ons, customized for local
hardware & software
```

3.2.0

0.15

2.06

seeking *source* from internet

research homepage

GitHub

GitHub/psi4

BitBucket



INTEGRATIONS FOR ALL

linking *built* in **CMAKE_PREFIX_PATH**



library

CORE DEVELOPERS

```
>>> # dir with only Psi4 source
>>> # all non-required add-ons disabled by default
>>> cmake \
    -DCMAKE_PREFIX_PATH="path1;/another/path2" \
    -DENABLE_CheMPS2 \
    -DENABLE_gdma \
    ...
    # CMake downloads external projects' source from internet
>>> make
>>> ctest -L addons
Psi4 core..... PASSED
Psi4 + DFTD3 executable..... PASSED
Psi4 + CheMPS2 library..... PASSED
Psi4 + PCMSolver library..... PASSED
Psi4 + V2RDM_CASSCF plugin..... PASSED
Psi4 + Ambit library..... PASSED
Psi4 library..... PASSED
>>> # working Psi4 executable complete with add-ons, customized for local
hardware & software
```



INTEGRATIONS FOR ALL

ENTER CONDA

- nothing so far has saved compile time, unless you built all those pkgs separately (and compatibly)
- you're probably not using many of the addons, so why bother maintaining their build
- conda's already got them built. why not reuse?

Package Repository for psi4

Packages	Files	Install Instructions	History
Filters			
Type: conda	Access: all	Label: main	
Package Name	Access	Summary	Updated
psi4	None	open-source quantum chemistry	2016-11-03
libint	public	E. Valeev and J. Fermann's two-body Gaussian molecular integrals	2016-11-03
chemps2	None	S. Wouters' spin-adapted implementation of DMRG for ab initio quantum chemistry	2016-10-30
pychemps2	public	python interface to S. Wouters' spin-adapted implementation of DMRG for ab initio quantum chemistry	2016-10-30
liberd	public	N. Flocke and V. Lotrich's Gaussian molecular integrals for AcesIII	2016-10-22
pymol	None	Schrödinger's Python-based molecular visualization system	2016-10-20
libint_am4	public	No Summary	2016-08-31
pcmsolver	public	An API for the Polarizable Continuum Model	2016-08-23
dftd3	None	S. Grimme's dispersion correction for DFT, Hartree-Fock, and semi-empirical quantum chemical methods	2016-08-23
gcp	public	S. Grimme's geometrical counterpoise correction for DFT and Hartree-Fock quantum chemical methods	2016-08-23
v2rdm_casscf	public	E. DePrince's variational 2-RDM-driven CASSCF plugin to Psi4	2016-08-23
ambit	public	No Summary	2016-08-23
psi4metapackage	None	add-ons and build dependencies for Psi4	2016-08-23
boost	public	No Summary	2016-08-23
omp5	public	Intel OpenMP redistributable library	2016-08-23
gcc-5	public	The GNU Compiler Collection	2016-08-23

INTEGRATIONS FOR ALL

The screenshot shows the Anaconda Cloud profile page for the user 'Psi4'. The profile section displays the user's name, a blue Psi4 logo, and their contribution history. Below the profile are their email address (psi4aiqc+binstar@gmail.com) and website (http://www.psiicode.org). The 'Packages' section lists 22 packages, including psi4, gdma, libefp, libint, chemps2, pychemps2, dkh, liberd, pymol, and libint_am4, each with a timestamp. The 'Environments' section shows two environments: 'linux_psi4_devel' and 'macos_psi4_devel', both marked as 'active'.

```
>>> # install a mini/anaconda
>>> # update it (conda update conda)
>>> # if "conda env create" below fails, conda install anaconda-client -or-
>>> # download file from site and save as environment.yml
>>> cat environment.yml
name: linux_psi4_devel
channels: !python/tuple
- !python/unicode 'defaults'
dependencies:
- mkl=11.3.3=0
- numpy=1.11.2=py27_0
- psi4/label/test::gdma=2.2.06=0
- psi4/label/test::psi4=1.1a2.dev3+20605d7=py27_0
- psi4::dftd3=3.2.0=7
- psi4::gcc-5=5.2.0=1
- psi4::libint=1.1.6=0
- python=2.7.12=1
...
>>> conda env create psi4/linux_psi4_devel (or macos_psi4_devel)
>>> source activate linux_psi4_devel
>>> # working Psi4 executable complete with add-ons
```

yml file

INTEGRATIONS FOR ALL

linking *built* in **CMAKE_PREFIX_PATH**



CORE DEVELOPERS

```
>>> # dir with only Psi4 source
>>> export PREFIX=/home/miniconda/envs/linux_psi4-devel
>>> cmake \
-DCMAKE_PREFIX_PATH="${PREFIX}" \
-DENABLE_CheMPS2=ON \
-ENABLE_gdma=ON \
-DMAX_AM_ERI=6 \
-DPYTHON_EXECUTABLE=${PREFIX}/bin/python2.7 \
...
# CMake collects mostly built conda libraries, internet src as fallback
>>> make
>>> ctest -L addons
Psi4 core..... PASSED
Psi4 + DFTD3 executable..... PASSED
Psi4 + CheMPS2 library..... PASSED
Psi4 + PCMSolver library..... PASSED
Psi4 + V2RDM_CASSCF plugin..... PASSED
Psi4 + Ambit library..... PASSED
Psi4 library..... PASSED
>>> # working Psi4 executable complete with add-ons, Psi4 core customized
for local hardware & software
```

downloading **source** from internet

research homepage

GitHub

BitBucket

GitHub/psi4



INTEGRATIONS FOR ALL

linking *built* in **CMAKE_PREFIX_PATH**



CORE DEVELOPERS

```
>>> # dir with only Psi4 source
>>> export PREFIX=/home/miniconda/envs/linux_psi4-devel
>>> cmake \
-> -DCMAKE_PREFIX_PATH=$PREFIX \
-> -DENABLE_CheMP2 \
-> -DENABLE_gdm \
-> -DMAX_AM_ERI=6 \
-> -DPYTHON_EXECUTABLE=$PREFIX/bin/python2.7 \
-> ... \
-> # CMake config
>>> make
>>> ctest -L add
Psi4 core...
Psi4 + DFTD3
Psi4 + CheMP2
Psi4 + PCMSO
Psi4 + V2RDM
Psi4 + Ambit
Psi4 library
>>> # working Psi4 executable compiled with gcc 4.8, V2RDM core disabled
for local hardware & software
```

```
>>> export PREFIX=/home/user/miniconda/envs/linux_psi4-devel
      # cmake itself
>>> ${PREFIX}/bin/cmake \
-> -DCMAKE_PREFIX_PATH="$PREFIX" \
-> -DENABLE_gdma=ON \
-> -DMAX_AM_ERI=6 \
-> -DPYTHON_EXECUTABLE=${PREFIX}/bin/python2.7 \
-> # c/c++ compilers (linux only)
-> -DCMAKE_C_COMPILER=${PREFIX}/bin/gcc \
-> -DCMAKE_CXX_COMPILER=${PREFIX}/bin/g++ \
-> # fortran compilers (linux & mac)
-> -DCMAKE_Fortran_COMPILER=${PREFIX}/bin/gfortran \
-> # sphinx for docs build
-> -DSPHINX_ROOT=${PREFIX} \
-> # gcc c++11 libraries, if compiler needs them
-> -DCMAKE_C_FLAGS="-gcc-name=${PREFIX}/bin/gcc" \
-> -DCMAKE_CXX_FLAGS="-gcc-name=${PREFIX}/bin/gcc -gxx-name=${PREFIX}/bin/g++"
```

downloading **source** from internet

research homepage

GitHub

BitBucket

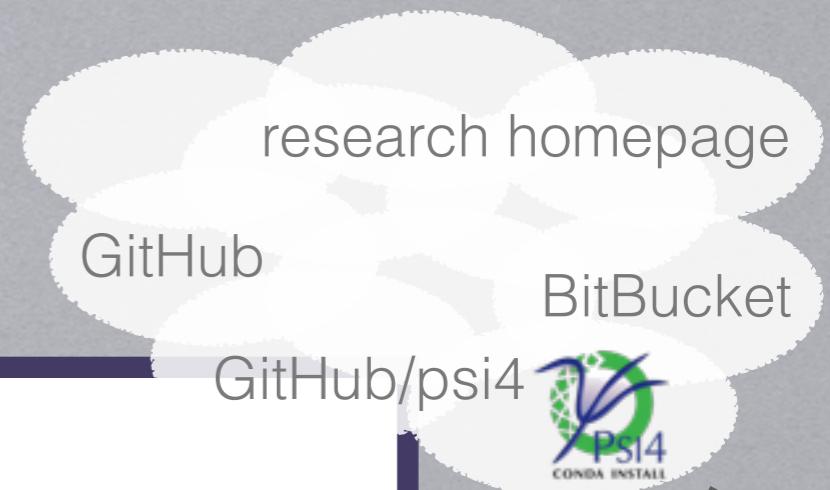
GitHub/psi4



INTEGRATIONS FOR ALL

USERS

```
>>> # download Psi4 installer  
>>> bash Psi4conda-latest.sh  
...  
Psi4 core..... PASSED  
Psi4 + DFTD3 executable..... PASSED  
Psi4 + CheMPS2 library..... PASSED  
Psi4 + PCMSolver library..... PASSED  
Psi4 + V2RDM_CASSCF plugin..... PASSED  
Psi4 + Ambit library..... PASSED  
Psi4 library..... PASSED  
>>> # working Psi4 executable complete with add-ons
```



PERIPHERAL DEVELOPERS

```
>>> # working Psi4 executable complete with add-ons (from above)  
>>> conda install gcc  
>>> psi4 -new-plugin mygreatcode  
>>> cd mygreatcode  
>>> make  
Attention! This SCF may be density-fitted.  
>>> # working compiling development environment ready for extension
```

VERSIONING

PRAY I DON'T ALTER IT ANY FURTHER

V DEVELOPMENT

SNAPSHOT

on master; passes quicktests, may break features;



FORMAT `M.m+1.devN`



UPDATE VIA `conda update psi4 -c psi4/label/level`

V PRE-RELEASE

on master; passes quicktests; conda devel channel



ALPHA much of the milestone is in



RC much a plausible release candidate



conda devel channel

V RELEASE

on master or maintenance; all tests pass, never less capable than its predecessor

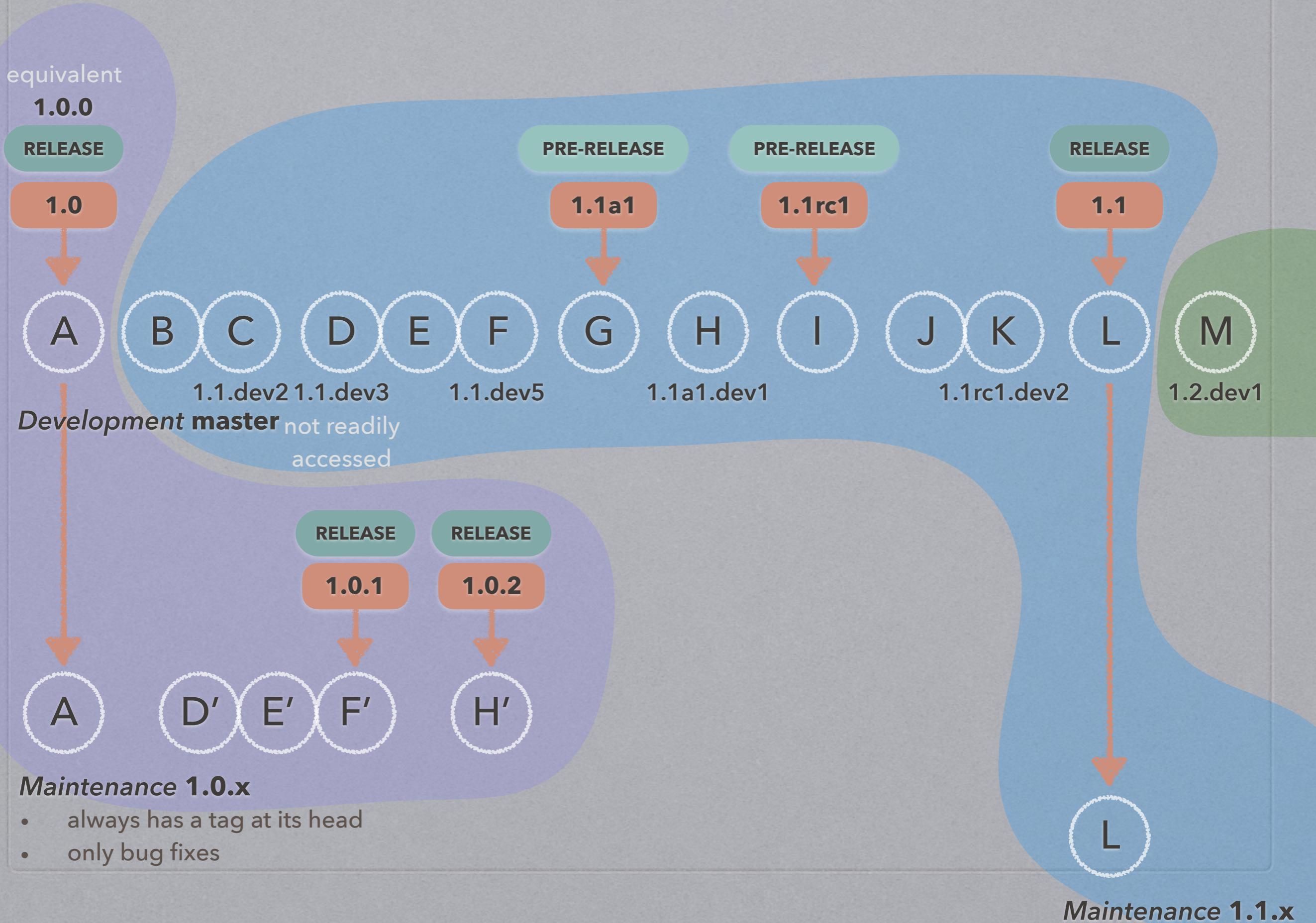


FORMAT master (`M.m`) or maintenance (`M.m.p`)



UPDATE VIA `conda update psi4`

- want to be able to break things on psi4/psi4 master occasionally (e.g., plugins, mints9) without those commits being a released version (implied by 1.0.101)
- want to be able to do bug fixes on releases (pulled off as 1.0.x) without naming conflicts)
- want to accommodate those who only want stable versions
- you can ignore tags if you like. but locally computed v will be wrong



PART III: PLUGINS & LAYOUT

JUSTIN M. TURNEY

PLUGINS

Plugins continue to be the best way to add new functionality to Psi4.

Your code is developed as a standalone entity, which is compiled independently of Psi4, but can still make use of Psi4's vast library.

Plugins are loaded at runtime by Psi4 from any location.

PLUGINS

```
% psi4 --help
--new-plugin NEW_PLUGIN
    Creates a new directory with files for writing a new
    plugin.
--new-plugin-template
{aointegrals,basic,dfmp2,mointegrals,scf,sointegrals,wavefunction}
    New plugin template to use.

% psi4 --new-plugin mydfmp2 --new-plugin-template dfmp2
-- Creating "mydfmp2" with "dfmp2" template. -----
==> Created plugin files:
__init__.py, CMakeLists.txt, doc.rst, input.dat, plugin.cc, pymodule.py
```

The ability to create just a Makefile for an existing plugin as been removed.

PLUGINS

```
project(mydfmp2 CXX)
find_package(psi4 1.0 REQUIRED)
add_psi4_plugin(mydfmp2 plugin.cc)
```

```
export CMAKE_PREFIX_PATH=\
/Users/jturney/Install/psi4
```

- Plugins now make use of CMake for configuring and building.
- If you install Psi4 into a non-standard location you will likely need to define CMAKE_PREFIX_PATH

PLUGINS

```
% cd mydfmp2

% cmake .
-- The CXX compiler identification is AppleClang 8.0.0.8000042
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Psi4 include directory: /Users/jturney/Install/psi4/include
-- Psi4 executable: /Users/jturney/Install/psi4/bin/psi4
-- Psi4 core library: /Users/jturney/Install/psi4/lib/psi4/core.so
-- Python executable: /Users/jturney/anaconda3/bin/python
-- Configuring done
-- Generating done
-- Build files have been written to: /Users/jturney/mydfmp2

% make

% psi4
```

PART IV: DIRECTORY LAYOUT

JUSTIN M. TURNEY

BINARIES AND LIBRARIES

MERGED

C. 2015

```
src/lib/  
libdiis/  
libdpd/  
libfock/  
libmints/  
libpsio/  
...  
src/bin/  
ccenergy/  
detci/  
dfmp2/  
dmrg/  
occ/  
sapt/  
scf/  
...
```



NOW

```
psi4/src/psi4/  
ccenergy/  
detci/  
dfmp2/  
dmrg/  
libdiis/  
libdpd/  
libfock/  
libmints/  
libpsio/  
occ/  
sapt/  
scf/  
...
```

- **WHAT** modules/libraries of C++ code, written by us
- **WHY** bin/lib distinguished by names; mirror installed header paths
- **NEW** *binaries* get built; *libraries* get their headers installed

C++ CORE

RAISED ABOVE MODULES AND CLEANED

C. 2015

```
src/bin/psi4/  
clean.cc  
export blas lapack.cc  
export functional.cc  
export mints.cc  
export oeprop.cc  
...  
psi4.cc  
psi4.h  
psi_start.cc  
psi_stop.cc  
python.cc  
read_options.cc  
script.cc  
script.h  
set_memory.cc
```



NOW

```
psi4/src/  
core.cc  
export blas lapack.cc  
export functional.cc  
export mints.cc  
export oeprop.cc  
...  
psi4/  
read_options.cc
```

- **WHAT** module-running code
- **WHY** sanity and CMake like dependencies as subdirectories
- **NEW** **python.cc** (`BOOST_PYTHON_MODULE(psi4)`) becomes **core.cc** (`PYBIND11_PLUGIN(core)`); most else gone

PSIDATADIR

SPLIT BY LANGUAGE, PURPOSE, & PROJECT

C. 2015

```
lib/  
basis/  
databases/  
efpfrag/  
fsapt/  
grids/  
plugin/  
psi4/  
python/  
quadratures/  
scripts/
```



NOW

```
psi4/share/psi4/  
basis/  
databases/  
fsapt/  
grids/  
plugin/  
quadratures/  
scripts/  
  
psi4/  
driver/
```

- **WHAT** the pieces that didn't need compilation
- **WHY** separate python and data; mirror GNU install directory structure in source
- **NEW** **python/** becomes **driver/**; envvar PSI4DATADIR defunct; efp evicted

HEADERS WINNOWED AND NAMESPACE

C. 2015

```
include/
Z_to_element.h
chkpt_params.h
compiler.h
cov_radii.h
element_to_Z.h
exception.h
fragment.h
masses.h
molecular_system.h
physconst.h
process.h
psi4-dec.h
psi4-def.h
psifiles.h
psitypes.h
...
```



NOW

```
psi4/include/psi4/
masses.h
physconst.h
pragma.h
psi4-dec.h
psifiles.h
pybind11.h

psi4/src/psi4/libmints/
element_to_Z.h
psi4/src/psi4/optking/
cov_radii.h
psi4/src/psi4/libpsi4util/
exception.h
psi4/src/psi4/libparallel/
process.h
```

- **WHAT** global (supra-module) headers
- **WHY** mirror GNU install directory structure in source
- **NEW** now actually installed! useable by plugins

EXTERNAL PROJECTS

TENTACLES SEVERED

boost/
boost_1_57_0.tar.bz2

cmake/
FindCHEMPS2.cmake
FindPCMSolver.cmake
ConfigPCMSolver.cmake
ConfigChemp2.cmake
ConfigBoost.cmake

interfaces/

libefp/

lib/

efpfrag/

src/lib/

libderiv/

liberd/

libint/

libmints/dkh2-dkh4_main.F90

C. 2015



NOW

external/
ambit/CMakeLists.txt
chemps2/CMakeLists.txt
dkh/CMakeLists.txt
gdma/CMakeLists.txt
gtfock/CMakeLists.txt
libefp/CMakeLists.txt
liberd/CMakeLists.txt
libint/CMakeLists.txt
pcmsolver/CMakeLists.txt
pybind11/CMakeLists.txt

- **WHAT** everything we didn't write that we depend on or use as enhancement
- **WHY** good practice, sanity for maintainers (LAB), and avoid possessing others' code
- **NEW** no external code stored locally, even tarballs & share; "config" CMake proj., not "module"

VERSION, HEADER, CL-ARGS

SPLIT BY LANGUAGE, PURPOSE, & PROJECT

c. 2015

src/bin/psi4/
gitversion.py
version.cc
psi_start.cc



NOW

psi4/
run_psi4.py.in
versioner.py
metadata.py

```
Psi4: An Open-Source Ab Initio Electronic Structure Package
Psi4 1.0 Driver

Git: Rev dirty

J. M. Turney, A. C. Simmonett, R. M. Parrish, E. G. Hohenstein,
F. A. Evangelista, J. T. Fermann, B. J. Mintz, L. A. Burns, J. J. Wilke,
M. L. Abrams, N. J. Russ, M. L. Leininger, C. L. Janssen, E. T. Seidl,
W. D. Allen, H. F. Schaefer, R. A. King, E. F. Valeev, C. D. Sherrill,
and T. D. Crawford, WIREs Comput. Mol. Sci. 2, 556-565 (2012)
(doi: 10.1002/wcms.93)

Additional Contributions by
A. E. DePrince, U. Bozkaya, A. Yu. Sokolov, D. G. A. Smith, R. Di Remigio,
R. M. Richard, J. F. Gomthier, H. R. McAlexander, M. Saitow, and
B. P. Pritchard

Psi4 started on: Monday, 17 October 2016 18:48AM
Process ID: 68892
PSIDATADIR: /Users/leriaab/linux/psihub/psi4inversion/psi4/objdir1/stage/Use
Inversion/install-psi4/share/psi4
Memory: 500.0 MiB
Threads: 1
```

```
usage: psi4 [-h] [-i INPUT] [-o OUTPUT] [-v] [-V] [-k] [-m] [-r] [-s SCRATCH]
            [-a] [-l PSIDATADIR] [-n NTHREAD] [-p PREFIX] [--inplace]
            [--new-plugin NEW_PLUGIN]
            [--new-plugin-makefile NEW_PLUGIN_MAKEFILE]

A hybrid C++/Python quantum chemistry module.

optional arguments:
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        Input file name. Default input.dat.
  -o OUTPUT, --output OUTPUT
                        Redirect output elsewhere. Default filename.out if
                        input is filename.filename.out if input is filename.in
                        or filename.dat output.dat if input is input.dat
  -v, --verbose          Print a lot of information.
  -V, --version          Print version information.
  -k, --skip-preprocessor
                        Skips input preprocessing. Expert mode.
  -m, --messy             Leave temporary files after the run is completed.
  -r, --restart           Number to be used instead of process id.
  -s SCRATCH, --scratch SCRATCH
                        Psi4 scratch directory to use.
  -a, --append            Append results to output file. Default Truncate first
  -l PSIDATADIR, --psidatadir PSIDATADIR
                        Specify where to look for the Psi data directory.
                        Overrides PSIDATADIR.
  -n NTHREAD, --nthread NTHREAD
                        Number of threads to use (overrides OMP_NUM_THREADS)
  -p PREFIX, --prefix PREFIX
                        Prefix name for psi files. Default psi
  --inplace              Runs psi4 from the source directory. !Warning! expert
                        option.
  --new-plugin NEW_PLUGIN
                        Creates a new directory with files for writing a new
                        plugin. You can specify an additional argument that
                        specifies a template to use, for example --new-plugin
                        name +mointegrals
  --new-plugin-makefile NEW_PLUGIN_MAKEFILE
                        Creates Makefile that can be used to compileplugins.
                        The Makefile is placed in the currentdirectory.
```

- **WHAT** initial run-time probes
- **WHY** everyone from Py, CM, GNU has versioning prefs; easier to control env through Py
- **NEW** wait for versioning lightning talk

DIRECTORY STRUCTURE

**CONCERNING
C-SIDE CORE**

**CONCERNING
PSI4+EXT
SUPERBUILD**

/
LICENSE (GPLv2+)
cmake/ (superbuild build support)
doc/ (documentation)
external/ (external projects)
media/ (logos)
plugins/ (model plugins)
psi4/ (Psi4 proj. in superbuild)
samples/ (tests less CMake)
tests/ (Psi4 core & add-on tests)

**CONCERNING
PSI4 WHOLE**

psi4/
`__init__.py` (import Py- & C-sides)
`driver/` (Py-side Psi4)
`import_core.py` (import C- details)
`include/` (C-side headers)
`(build config help)`
`e.in` (CM detection)
`(simulate psi4 exe)`
`(text data library)`
 (C-side Psi4)
`reader & versioning`

psi4/src/

`core.cc` (make py-mod, top exports)
`export_mints.cc` (module exports)
`export blas lapack.cc` ("")
`al.cc` ("")
`c` ("")
 (module source)
 (module options)

DIRECTORY STRUCTURE

NOT MEANT TO BE CRUEL

psi4/psi4/src/psi4

- constrained by GitHub repository name
- contains the CMake project

psi4/psi4/src/psi4

- constrained by Python module name
- contains the Python project

psi4/psi4/src/psi4

- constrained by GNU project name
- contains the C++ project
- new namespace pad dirs, but tab-able

psi4/psi4/share/psi4

psi4/psi4/include/psi4

PART V: A BRIEF HISTORY OF VERSIONING Psi4

OR, WHY THINGS KEEP GETTING MORE COMPLICATED

LORI A. BURNS

IN THE BEGINNING

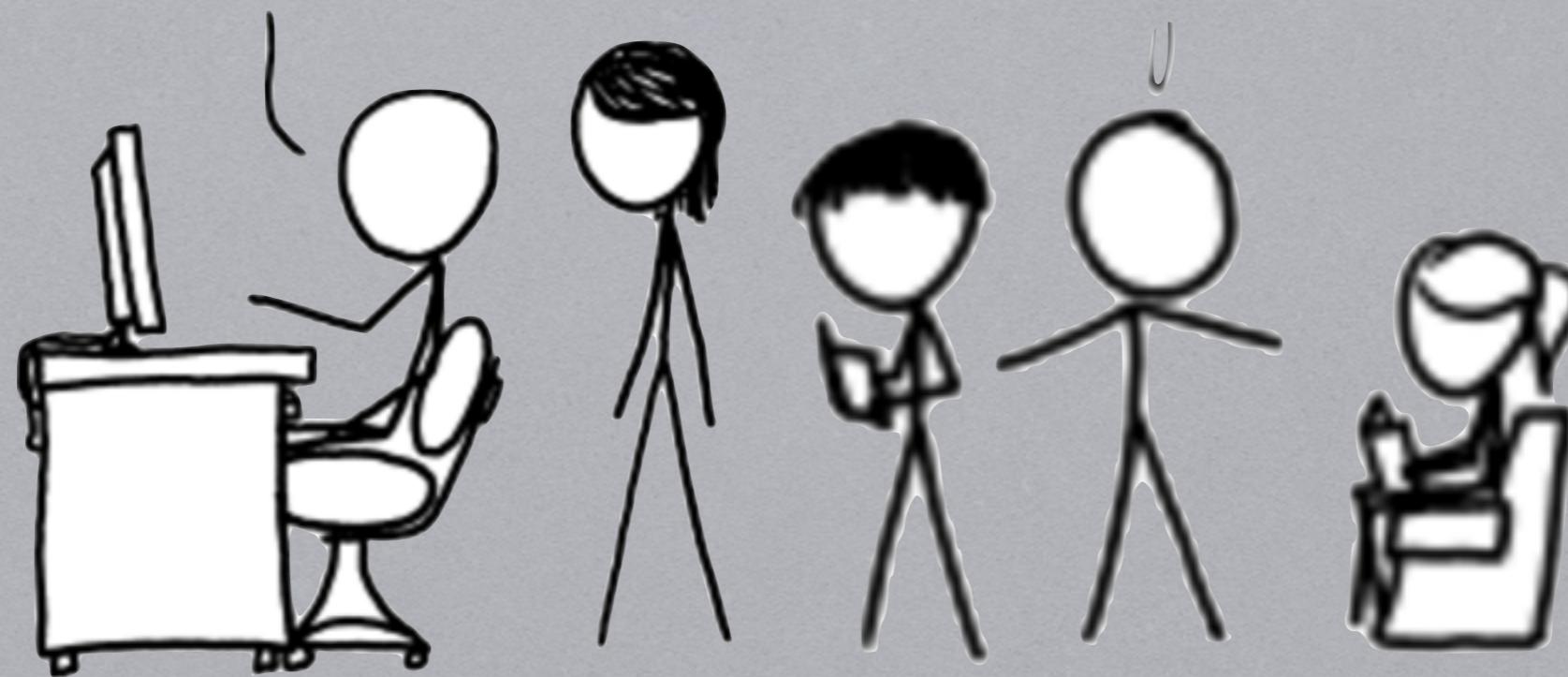
JET NEEDED NO VERSIONING



- **CONSTRAINTS** None: only one person. Plus, SVN numbers sequentially.

THE INTERREGNUM

TRUE DEVELOPERS ONLY NEED GIT HASHES



- **CONSTRAINTS** Minimal: only need to ID the code so git hash suffices.

VERSIONING CIRCA 2015

ENTER CONDA & SORTABILITY



- CONSTRAINTS `conda update psi4` needs a sortable version to act upon

VERSIONING CIRCA 2015

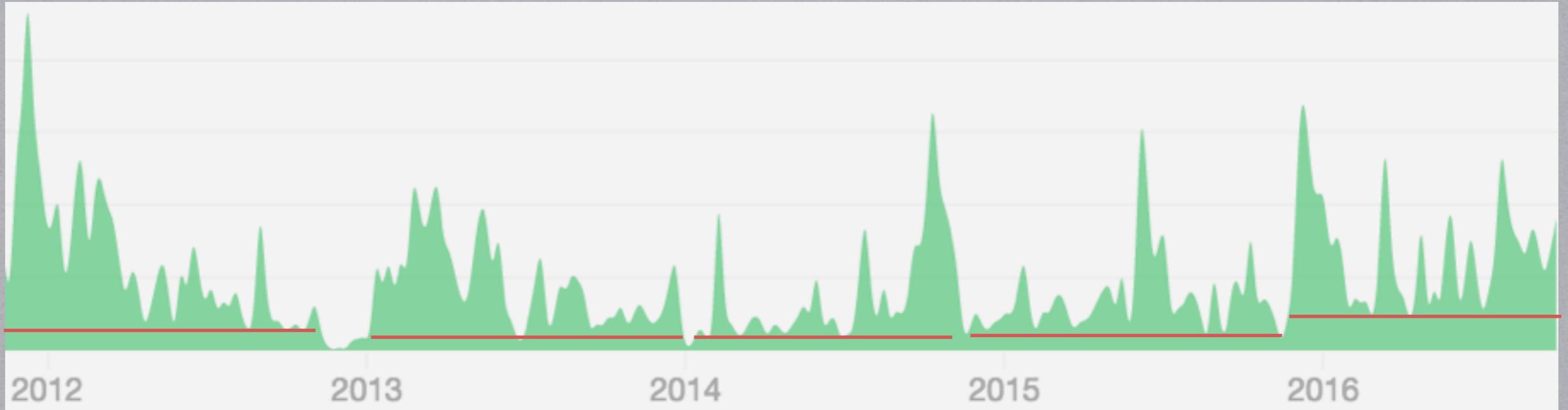
ENTER CONDA & SORTABILITY



- CONSTRAINTS `conda update psi4` needs a sortable version to act upon

VERSIONING IN THE AGE OF CHANGE

I HAVE ALTERED THE VERSIONING

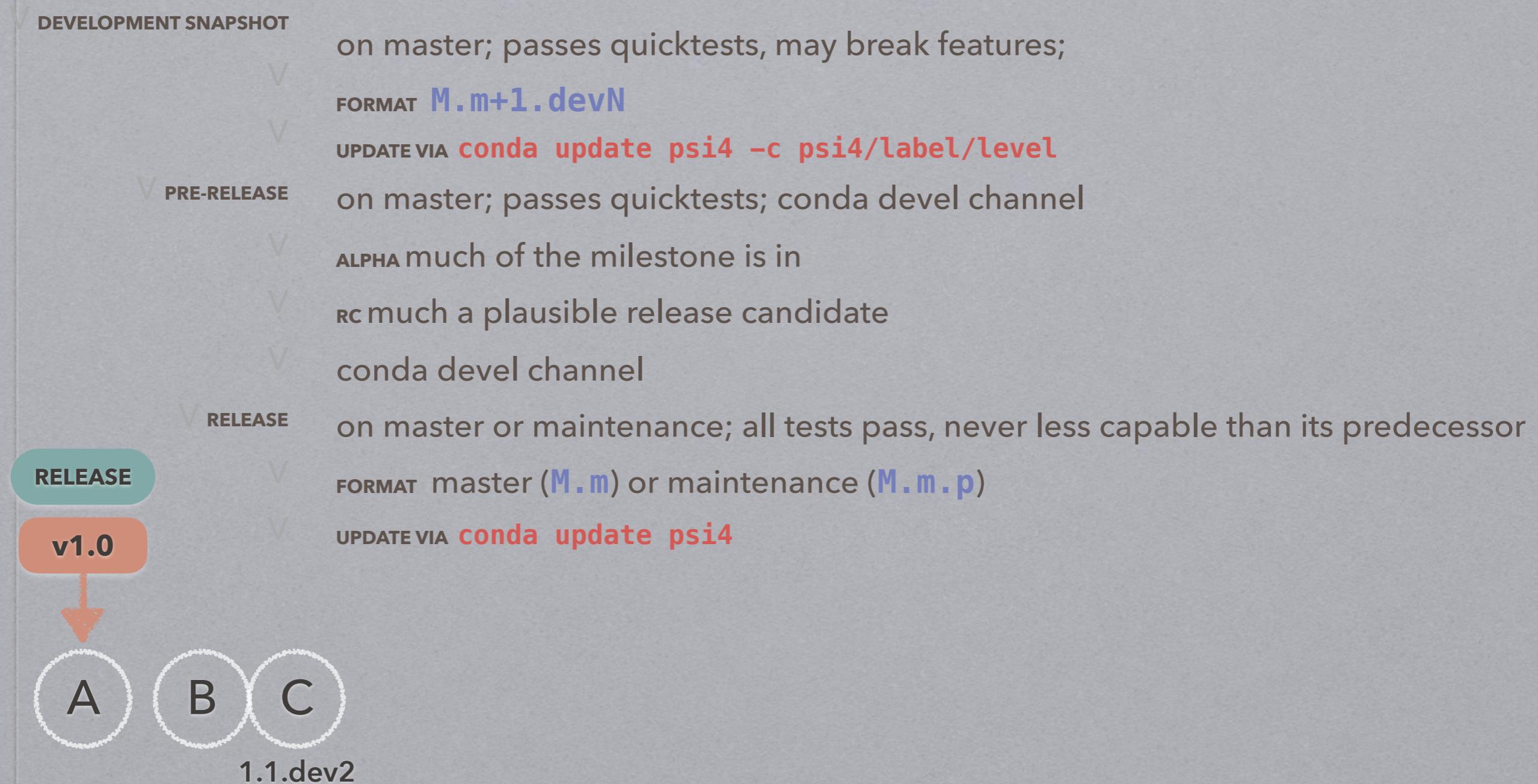


- CONSTRAINTS
 - want to break things on master occasionally (e.g., plugins, cc14) without implying a released version
 - want to do bug fixes on tagged releases without naming conflicts
 - want to accommodate users who only want stable versions

THE FINE PRINT: you can ignore tags if you like. but locally computed version will be undefined

VERSIONING

PRAY I DON'T ALTER IT ANY FURTHER



DEVELOPMENT SNAPSHOT

PRE-RELEASE

RELEASE

on master; passes quicktests, may break features; **M.m+1.devN** format; **conda update psi4 -c psi4/label/devel**
on master; passes quicktests; conda devel channel
on master (**M.m**) or maintenance (**M.m.p**) branch; all tests pass, always excepting deprecations, never less capable
than its predecessor; **conda update psi4**

equivalent

1.0.0

RELEASE

v1.0



1.1.dev2

Development master not readily accessed



1.1.dev3

1.1.dev5

PRE-RELEASE

v1.1a1



1.1a1.dev1

PRE-RELEASE

v1.1rc1



1.1rc1.dev2

RELEASE

v1.1

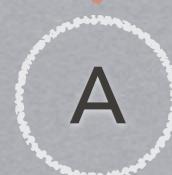


1.2.dev1

Development master not readily accessed

RELEASE

v1.0.1



E'

F'

RELEASE

v1.0.2



Maintenance 1.0.x

- always has a tag at its head
- only bug fixes



Maintenance 1.1.x

DEVELOPMENT SNAPSHOT

on master; passes quicktests, may break features; **M.m+1.devN** format; **conda update psi4 -c psi4/label/devel**
on master; passes quicktests; conda devel channel
on master (**M.m**) or maintenance (**M.m.p**) branch; all tests pass, always excepting deprecations, never less capable
than its predecessor; **conda update psi4**

equivalent

1.0.0

RELEASE

v1.0

A

B

C

D

E

F

G

H

I

J

K

L

M

1.1.dev2

1.1.dev3

1.1.dev5

1.1a1.dev1

1.1rc1.dev2

1.2.dev1

Development master not readily
accessed

RELEASE

v1.0.1

RELEASE

v1.0.2

A

D'

E'

F'

H'

Maintenance 1.0.x

- always has a tag at its head
- only bug fixes

L

Maintenance 1.1.x