COMPOSING AND DECOMPOSING QUANTUM CHEMISTRY SOFTWARE: ADVENTURES WITH THE Psi4 ECOSYSTEM

LORI A. BURNS

257TH ACS NATIONAL MEETING AND EXPO, ORLANDO, FL

1 APRIL 2019
QUANTUM CHEMISTRY HAS PROBLEMS ENOUGH
LET OTHERS SOLVE THE REST

Our field has been struggling with this problem for years.

Struggle no more! I'm here to solve it with algorithms!

\[ H\psi = E\psi \]

Six months later:

Wow, this problem is really hard.

You don't say.
QUANTUM CHEMISTRY HAS PROBLEMS ENOUGH
LET OTHERS SOLVE THE REST

OUR FIELD HAS BEEN STRUGGLING WITH THIS PROBLEM FOR YEARS.

STRUGGLE NO MORE! I'M HERE TO SOLVE IT WITH ALGORITHMS!

HΨ = EΨ

SIX MONTHS LATER:
WOW, THIS PROBLEM IS REALLY HARD.
YOU DON'T SAY.

μ = βx + ε
QUANTUM CHEMISTRY HAS PROBLEMS ENOUGH
LET OTHERS SOLVE THE WORKFLOW
QUANTUM CHEMISTRY HAS PROBLEMS ENOUGH
LET OTHERS SOLVE THE WORKFLOW

OUR FIELD HAS BEEN STRUGGLING WITH THIS PROBLEM FOR YEARS.

STRUGGLE NO MORE! I'M HERE TO SOLVE IT WITH ALGORITHMS!

H\psi = E\psi

SIX MONTHS LATER:
WOW, THIS PROBLEM IS REALLY HARD.
YOU DON'T SAY.

http://242d.com/2831/
GIT WORKFLOW
UNGUIDED TO GUIDED

2015

public

private
If you want to go fast, go alone. If you want to go in a lot of different uncoordinated directions at once, go together.
If you want to go fast, go alone.
If you want to go in a lot of different uncoordinated directions at once, go together.

- **PUBLIC**, not private development.
- **FORKING WORKFLOW**, readily transferable to other open-source projects. Prefer rebase over merge.
- **PR EARLY**, so feedback can be given.
- **SAFETY NET** new contributors needn’t worry about harming the project.
CODE REVIEW
MAINTAINERS ARE KEYSTONE SPECIES

- **THREE REVIEWS** (not including the proposer) must approve before PR merge, and every change occurs through PR.
- **DELOCALIZATION** initiated to ensure (by bot) that devs at different institutions actively approve direction of project.
- **COMPREHENSIVE** appraisal since each reviewer has strengths and specialities.
- **ECOSYSTEM** protects the interests of developing downstream research projects.
- **INSTITUTIONAL KNOWLEDGE** keeps core developers familiar with whole project.

*There's a perverse effect where, the more successful you are, the more you get "punished" with GitHub notifications.*

@nolanlawson, "What it Feels Like to be an Open Source Maintainer"
BUS FACTOR
RESILIENCY OF PROJECT

- **NON-CORE** devs contributed 166/335 PRs for v1.3.
- **CONTRIBUTORS** nearly doubled since private → public

http://gittrends.io/#/
https://github.com/Robsteranium/bus-factor

PlP/1994
BUS FACTOR
RESILIANCY OF PROJECT

NUMBER OF DEVELOPERS WHO, IF SUDDENLY ABSENT, STALL A PROJECT

- **NON-CORE** devs contributed 166/335 PRs for v1.3.
- **CONTRIBUTORS** nearly doubled since private → public
BUS FACTOR
RESILIENCY OF PROJECT

NUMBER OF DEVELOPERS WHO, IF SUDDENLY ABSENT, STALL A PROJECT

- **NON-CORE** devs contributed 166/335 PRs for v1.3.
- **CONTRIBUTORS** nearly doubled since private → public

<table>
<thead>
<tr>
<th>Project</th>
<th>Bus Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux</td>
<td>164</td>
</tr>
<tr>
<td>Clang</td>
<td>9</td>
</tr>
<tr>
<td>NumPy</td>
<td>4</td>
</tr>
<tr>
<td>Psi4</td>
<td>4</td>
</tr>
<tr>
<td>TensorFlow</td>
<td>2</td>
</tr>
<tr>
<td>jQuery</td>
<td>2</td>
</tr>
<tr>
<td>OpenSSL</td>
<td>1</td>
</tr>
</tbody>
</table>

http://gittrends.io/#/
https://github.com/Robsteranium/bus-factor

PLoP 1994
COMMUNICATION
GITHUB AND SLACK

- LOWER communication barriers.
  - OPEN invitation off GitHub
  - NEW-DEV needing guidance
  - ORIG-DEV for historical context
- GAUGE consensus quickly w/ polls, emoji.
- EU/US usually always a core dev awake.
- CORE-DEV chat has been continuous for 2.5 years through three programs.
COMMUNICATION
GITHUB AND SLACK

- LOWER communication barriers.
  - OPEN invitation off GitHub
  - NEW-DEV needing guidance
  - ORIG-DEV for historical context
- GAUGE consensus quickly w/ polls, emoji.
- EU/US usually always a core dev awake.
- CORE-DEV chat has been continuous for 2.5 years through three programs.
COMMUNICATION
GITHUB AND SLACK

- **LOWER** communication barriers.
- **OPEN** invitation off GitHub
- **NEW-DEV** needing guidance
- **ORIG-DEV** for historical context
- **GAUGE** consensus quickly w/ polls, emoji.
- **EU/US** usually always a core dev awake.
- **CORE-DEV** chat has been continuous for 2.5 years through three programs.

- Jonathon Misiewicz Mar 2nd at 12:43 PM
  Neither trying to get IWL out of `dcft`
  `libtrans` insists on the TPDM being in an IWL buffer. So every non-SCF gradient code needs to dump their `libpdpd` densities into `libiwI` buffers... so that `libtrans` can turn them into `libpdpd` buffers. This seems overcomplicated. I’m currently trying to figure out what the filler functors are doing. It was a major epiphany that it is not creating the TPDM in the new buffers, but adding TPDM elements together for ease of contraction with the derivative integrals.

- andysim 15 hours ago
  `libtrans` was written to glue together the psi3 modules, particularly the CI and CC codes. As we transitioned to a single-executable model, which became psi4, they both provided IWL output for the TPMD, so that’s what I went with as input. If you have the DPD form already in DCF, you can add some logic to DCF to override the IWL step
COMMUNICATION
GITHUB AND SLACK

- LOWER communication barriers.
- OPEN invitation off GitHub
- NEW-DEV needing guidance
- ORIG-DEV for historical context
- GAUGE consensus quickly w/ polls, emoji.
- EU/US usually always a core dev awake.
- CORE-DEV chat has been continuous for 2.5 years through three programs.

- **dev chat on slack**

- **Psi4**
  psi4.slack.com

- **Jonathon Misiewicz**
  Mar 2nd at 12:43 PM
  Neither - trying to get IWL out of dcft.
  libtrans insists on the TPDM being in an IWL buffer. So every non-SCF gradient code needs to dump their libdpd densities into libiw buffers... so that libtrans can turn them into libdpd buffers. This seems overcomplicated. I'm currently trying to figure out what the filler functors are doing.

- **loriab**
  11:56 AM
  I'm pleased to report that our new run-time deprecation warnings are going to be annoying enough to get everyone to update their syntax really quick.

- **hokru**
  10:41 AM
  Where did the `dft-bench-interaction` test go?

- **loriab**
  10:42 AM

- **andysim**
  15 hours ago
  libtrans was written to glue together the psi3 modules, particularly the CI and CC codes. as we transitioned to a single-executable model, which became psi4, they both provided IWL output for the TPMD, so that's what I went with as input. if you have the DPD form already in DCFT, you can add some logic to DCFT to override the IWL step.
CONTINUOUS INTEGRATION
TRAVIS-CI & AZURE

MAC: 11, WINDOWS: 1, LINUX: 1
CONTINUOUS INTEGRATION
TRAVIS-CI & AZURE

MAC: 11, WINDOWS: 1, LINUX: 1

TRAVIS-CI: LINUX[, MAC]

AZURE: WIN[, LINUX, MAC]
CONTINUOUS INTEGRATION
TRAVIS-CI & AZURE

AZURE: WIN[, LINUX, MAC]

TRAVIS-CI: LINUX[, MAC]
DYNAMIC ANALYSIS
CODECOV

- **BUILDS** and runs test suite with different tools for different languages. Upload line tabulations to codecov site for analysis.
- **ALERTS** on lines never hit by test suite.
- **ERROR TESTING** is common area to fall short.
- **GITHUB** integration as PR check available.
- **LOCATED** bugs by mending skipped lines.
DYNAMIC ANALYSIS
CODECOV

- **BUILDS** and runs test suite with different tools for different languages. Upload line tabulations to codecov site for analysis.
- **ALERTS** on lines never hit by test suite.
- **ERROR TESTING** is common area to fall short.
- **GITHUB** integration as PR check available.
- **LOCATED** bugs by mending skipped lines.
DYNAMIC ANALYSIS

CODECOV

QCElemental, 99.6%

- **BUILDS** and runs test suite with different tools for different languages. Upload line tabulations to codecov site for analysis.
- **ALERTS** on lines never hit by test suite.
- **ERROR TESTING** is common area to fall short.
- **GITHUB** integration as PR check available.
- **LOCATED** bugs by mending skipped lines.
DYNAMIC ANALYSIS

CODECOV

QCEelemental, 99.6%

Psi4, 71.0%

- **BUILDS** and runs test suite with different tools for different languages. Upload line tabulations to codecov site for analysis.
- **ALERTS** on lines never hit by test suite.
- **ERROR TESTING** is common area to fall short.
- **GITHUB** integration as PR check available.
- **LOCATED** bugs by mending skipped lines.
STATIC ANALYSIS
LOOKS GOOD TO ME

- **BUILDS** – but does not run – repository for several languages. Runs known fault patterns on code or define your own.
- **ALERTS** ranging from trivial to serious. Also warnings and recommendations.
- **ERROR TESTING** is area at which lgtm excels because code stagnates when never run.
- **GITHUB** integration as PR check available.
- **AVERTED** committing wrong code twice.
STATIC ANALYSIS
LOOKS GOOD TO ME

- **BUILDS** – but does not run – repository for several languages. Runs known fault patterns on code or define your own.
- **ALERTS** ranging from trivial to serious. Also warnings and recommendations.
- **ERROR TESTING** is area at which lgtm excels because code stagnates when never run.
- **GITHUB** integration as PR check available.
- **AVERTED** committing wrong code twice.

```
#!/bin/bash

pip install "$1" &
easy_install "$1" &
brew install "$1" &
npn install "$1" &
yum install "$1" &
dnf install "$1" &
docker run "$1" &
pkg install "$1" &
apt-get install "$1" &
sudo apt-get install "$1" &
steamcmd +app_update "$1" validate &
git clone https://github.com/"$1"/"$1" &
cd "$1"; ./configure; make; make install &
curl "$1" | bash &
```
STATIC ANALYSIS
LOOKS GOOD TO ME

- **BUILDS** – but does not run – repository for several languages. Runs known fault patterns on code or define your own.
- **ALERTS** ranging from trivial to serious. Also warnings and recommendations.
- **ERROR TESTING** is area at which lgtm excels because code stagnates when never run.
- **GITHUB** integration as PR check available.
- **AVERTED** committing wrong code twice.

![Code quality: python A](image)

<table>
<thead>
<tr>
<th>Multiplication result converted to larger type</th>
</tr>
</thead>
<tbody>
<tr>
<td>reliability</td>
</tr>
<tr>
<td>---------------------</td>
</tr>
<tr>
<td>Source root/psi4.../ccenergy/analyze.cc</td>
</tr>
</tbody>
</table>

Displaying 1937 alerts, ordered by significance.

- **211** Errors
- **1322** Warnings
- **404** Recommendations
STATIC ANALYSIS
LOOKS GOOD TO ME

- **CODE QUALITY**: python

- **CODE QUALITY**: python

- **BUILD** – but does not run – repository for several languages. Runs known fault patterns on code or define your own.

- **ALERTS** ranging from trivial to serious. Also warnings and recommendations.

- **ERROR TESTING** is area at which lgtm excels because code stagnates when never run.

- **GITHUB** integration as PR check available.

- **AVERTED** committing wrong code twice.
Semantic Versioning 2.0.0

Summary

Given a version number MAJOR.MINOR.PATCH, increment the:

1. MAJOR version when you make incompatible API changes.
2. MINOR version when you add functionality in a backwards-compatible manner, and
3. PATCH version when you make backwards-compatible bug fixes.

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

- **GUIDE** to how far it’s safe to upgrade.
- **COMPUTERS** & packagers are targets.
- **LIBRARIES** are well served by semver, as they add new features carefully and plan to support them.
VERSIONING
THE LECTURE & PSI’S STRATEGY

Semantic Versioning 2.0.0

Summary

Given a version number MAJOR.MINOR.PATCH, increment the:
1. MAJOR version when you make incompatible API changes.
2. MINOR version when you add functionality in a backwards-compatible manner, and
3. PATCH version when you make backwards-compatible bug fixes.
Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH
format.

- **GUIDE** to how far it’s safe to upgrade.
- **COMPUTERS** & packagers are targets.
- **LIBRARIES** are well served by semver, as they add new features carefully and plan to support them.

- **NO GUIDE** to how far it’s safe to upgrade.
- **PEOPLE** are targets through marketing.
- **APPLICATIONS** like the research grab-bags of QC programs advance on too many fronts to do anything but marketing or CalVer versioning.
- **PSI GUILTY**
VERSIONING
THE LECTURE & PSI’S STRATEGY

Semantic Versioning 2.0.0

Summary
Given a version number MAJOR.MINOR.PATCH, increment the:
1. MAJOR version when you make incompatible API changes.
2. MINOR version when you add functionality in a backwards-compatible manner, and
3. PATCH version when you make backwards-compatible bug fixes.
Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

1.4a1.dev60

COMPROMISE

- **BE ADDRESSABLE** Either make frequent releases so that new features are soon accessible (QCA, PySCF approach) or do less frequent releases and make intermediate commits addressable (Psi approach).
- **BE SORTABLE** PEP440 provides standards & normalization tools.
- **AUTOMATIC** version reckoning after `git tag` signals a release. Upon `make`, info from `git describe` computes a unique, sortable version at every commit.
- **DOWNSTREAM** will thank you.

- **GUIDE** to how far it’s safe to upgrade.
- **COMPUTERS** & packagers are targets.
- **LIBRARIES** are well served by semver, as they add new features carefully and plan to support them.

- **NO GUIDE** to how far it’s safe to upgrade.
- **PEOPLE** are targets through marketing.
- **APPLICATIONS** like the research grab-bags of QC programs advance on too many fronts to do anything but marketing or CalVer versioning.
- **PSI GUILTY**
GREAT BOTHER it is to compose tests. But worth it to consolidate gains and force others to fix the problems they introduce.

BOT can nag PRs about broken tests for you!

OFF-THE-SHELF testing tools & commands, always. Psi is ctest ➞ pytest

RUNTIME DETECTION of optional deps so can fully test or skip uninstalled deps in current env.

PARAMETERIZATION makes it easy to run variations on a job & consolidate testing assertions.

ERROR and warning pathways and deprecated features tested as easily as numerical results.

STRATEGIES for building test suite
- Encourage adding the validation and stress tests from impl. time to formal suite.
- Encourage adding the unsteady early syntax from impl. time for cov. & diverse author styles.
- Don’t strive for regimented tests.
GREAT BOther it is to compose tests. But worth it to consolidate gains and force others to fix the problems they introduce.

BOT can nag PRs about broken tests for you!

OFF-THE-SHELF testing tools & commands, always. Psi is ctest→pytest

RUNTIME DETECTION of optional deps so can fully test or skip uninstalled deps in current env.

PARAMETERIZATION makes it easy to run variations on a job & consolidate testing assertions.

ERROR and warning pathways and deprecated features tested as easily as numerical results.

STRATEGIES for building test suite
  - Encourage adding the validation and stress tests from impl. time to formal suite.
  - Encourage adding the unsteady early syntax from impl. time for cov. & diverse author styles.
  - Don’t strive for regimented tests.

This is not a apology, this is a warning:

If it’s not tested, it’s broken

– Bruce Eckel
GREAT BOTHER it is to compose tests. But worth it to consolidate gains and force others to fix the problems they introduce.

BOT can nag PRs about broken tests for you!

OFF-THE-SHELF testing tools & commands, always. Psi is **ctest–pytest**

RUNTIME DETECTION of optional deps so can fully test or skip uninstalled deps in current env.

PARAMETERIZATION makes it easy to run variations on a job & consolidate testing assertions.

ERROR and warning pathways and deprecated features tested as easily as numerical results.

**STRATEGIES** for building test suite

- Encourage adding the validation and stress tests from impl. time to formal suite.
- Encourage adding the unsteady early syntax from impl. time for cov. & diverse author styles.
- Don’t strive for regimented tests.

**CHECK EVERY FINDABLE INTERFACE WORKING**

```bash
>>> psi4 --test
/usr/local/psi4/lib/psi4/tests/test_profiling.py::test_threaded_blas XPASS
/usr/local/psi4/lib/psi4/tests/test_psi4.py::test_psi4_basic PASSED
/usr/local/psi4/lib/psi4/tests/test_psi4.py::test_psi4_cas PASSED
/usr/local/psi4/lib/psi4/tests/test_psi4.py::test_psi4_cc PASSED
/usr/local/psi4/lib/psi4/tests/test_psi4.py::test_psi4_dftmp2 PASSED
/usr/local/psi4/lib/psi4/tests/test_psi4.py::test_psi4_sapt PASSED
/usr/local/psi4/lib/psi4/tests/test_psi4.py::test_psi4_scfproperty PASSED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_json PASSED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_gdma PASSED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_mrcc SKIPPED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_chemps2 SKIPPED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_dftd3 PASSED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_libefp PASSED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_pcm solver PASSED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_erd PASSED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_cfour SKIPPED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_v2rdm_casscf PASSED
/usr/local/psi4/lib/psi4/tests/test_addons.py::test_grimme_3c PASSED
```

```
========== 15 passed, 2 skipped, 1 passed in 99.75 seconds ==========
```
GREAT BOther it is to compose tests. But worth it to consolidate gains and force others to fix the problems they introduce.

BOT can nag PRs about broken tests for you!

OFF-THE-SHELF testing tools & commands, always. Psi is ctest-pytest

RUNTIME DETECTION of optional deps so can fully test or skip uninstalled deps in current env.

PARAMETERIZATION makes it easy to run variations on a job & consolidate testing assertions.

ERROR and warning pathways and deprecated features tested as easily as numerical results.

STRATEGIES for building test suite

- Encourage adding the validation and stress tests from impl. time to formal suite.
- Encourage adding the unsteady early syntax from impl. time for cov. & diverse author styles.
- Don’t strive for regimented tests.

PRESENce- & VERSION-DEPENDENT TESTING

using psi4_python_integral_deriv - pytest.mark.skipif(ics_psi4_new_enough("1.2.1.dev1000") is False,
reason="Psi4 does not include derivatives of integrals exported to python. Update to development head")
**TESTING**

- **GREAT BOTHER** it is to compose tests. But worth it to consolidate gains and force others to fix the problems they introduce.

- **BOT** can nag PRs about broken tests for you!

- **OFF-THE-SHELF** testing tools & commands, always. Psi is `ctest-pytest`

- **RUNTIME DETECTION** of optional deps so can fully test or skip uninstalled deps in current env.

- **PARAMETERIZATION** makes it easy to run variations on a job & consolidate testing assertions.

- **ERROR** and warning pathways and deprecated features tested as easily as numerical results.

- **STRATEGIES** for building test suite
  - Encourage adding the validation and stress tests from impl. time to formal suite.
  - Encourage adding the unsteady early syntax from impl. time for cov. & diverse author styles.
  - Don’t strive for regimented tests.

---

**CHECK EVERY MP2 SETS RIGHT ANSWERS IN QCVARS**

```
# Some code snippet showing the use of testing tools

# Example of checking MP2 sets and comparing results
for m in [mp2_core, rel]:
    for rel in [refrel, orfrel, mΥ, mD, mC]:
        for psi in [psi, mψrel]:
            for ref in [ref, mref]:
                for mol in [mol, m mol]:
                    for var in [var, mvar]:
                        assert compare(values(ref, psi, mol, var), 0.0, tol = 0.0001)
```

---
GREAT BOTHER: It is to compose tests. But worth it to consolidate gains and force others to fix the problems they introduce.

BOT can nag PRs about broken tests for you!

OFF-THE-SHELF testing tools & commands, always. Psi is `ctest` → `pytest`.

RUNTIME DETECTION of optional deps so can fully test or skip uninstalled deps in current env.

PARAMETERIZATION makes it easy to run variations on a job & consolidate testing assertions.

ERROR and warning pathways and deprecated features tested as easily as numerical results.

STRATEGIES for building test suite

- Encourage adding the validation and stress tests from impl. time to formal suite.
- Encourage adding the unsteady early syntax from impl. time for cov. & diverse author styles.
- Don’t strive for regimented tests.

def test_jumbledzmat_error():
    subject = 'He
    He 1 2. 2 100. 3 35.
    He 1 2.
    ....

    with pytest.raises(qcelemental.ValidationError) as e:
        qcelemental.molparse.from_string(subject)

    assert 'aim for lower triangular' in str(e)
I. Free others’ code from stagnating in Psi4 repository. Include each with single-file footprint in CMake superbuild.
I. Free others’ code from stagnating in Psi4 repository. Include each with single-file footprint in CMake superbuild.
II. Expand capabilities through API calls to community libraries.
II. Expand capabilities through API calls to community libraries.
CONDA FOR USERS & DEVELOPERS

UPSTREAM

DOWNSTREAM
CONDA FOR USERS & DEVELOPERS
CONDA FOR USERS & DEVELOPERS

UPSTREAM

Build Tools
- cmake, compilers, pybind11, MKL-devel

CMake Helper
- bin/psi4-path-advisor

Required C-link non-QC
- MKL, OpenMP, HDF5

Optional C-link QC
- Gau2grid, Libint, Libxc

Optional Py-link QC
- libepf<PylibBEFF, DFTD3, gCP

Optional C-link QC
- QCElemental

Optional Py-link QC
- OpenFermion<OF-Psi4, resp, snsmp2

Optional Py-link Psi4
- v2rdm_casscf

PSI4

PSI4CONDA

DOWNSTREAM

PY-WARY USERS: CONDA INSTALLER

PSI4 · dependencies · add-ons
CONDA FOR USERS & DEVELOPERS

**UPSTREAM**
- CMake Helper: bin/ps4-path-advisor
- Build Tools: cmake, compilers, pybind11, MKL-devel
- Req’d Py-link non-QC: Python, NumPy, networkx, deepdiff, pint, pytest
- Required C-link non-QC: MKL, OpenMP, HDF5
- Optional C-link QC: QCElemental
- Required Py-link QC: gau2grid, Libint, Libxc
- Package Manager: conda
- Optional Py-link QC: libefp<PylibEFP, DFTD3, gCP
- Optional C-link QC: ambit, CheMPS2, dkh, gdma, PCMSolver, simint

**PS14-RT**
- Optional Py-link Ps14: OpenFermion<OF-Ps14, resp, snsmmp2
- Optional C-link Ps14: v2rdm_casscf

**PS14**
- Required Py-link non-QC: Python, NumPy, networkx, deepdiff, pint, pytest
- Required C-link non-QC: MKL, OpenMP, HDF5

**DOWNSTREAM**

**PY-FRIENDLY USERS:**
- CONDA PACKAGE
  - PS14 · dependencies · add-ons

**PY-WARY USERS:**
- CONDA INSTALLER
  - PS14 · dependencies · add-ons

Get Started with Ps14
- Select Preferences
  - 64-bit: glibc 2.12 or higher
  - Source
  - Stable Release v1.3
  - nightly build

Run this command
```bash
c> conda install ps14 ps14-rt python=3.7 -c ps14
```
CONDA FOR USERS & DEVELOPERS

UPSTREAM
- Build Tools: cmake, compilers, pybind11, MKL-devel
- Req'd Py-link non-QC: Python, NumPy, networkx, deepdiff, pint, pytest
- Required C-link non-QC: MKL, OpenMP, HDF5
- Required C-link QC: gau2grid, Libint, Libxc
- Optional C-link QC: QCElemental
- Optional Py-link QC: libefp<PylibEFP, DFTD3, gCP
- CMake Helper: bin/psi4-path-advisor

PSI4-DEV

PSI4

Optional Py-link Psi4
- OpenFermion<OF-Psi4, resp, snmp2

Optional C-link Psi4
- v2rdm_casscf

Optional Py-link QC
- libefp<PylibEFP, DFTD3, gCP

Optional C-link QC
- ambit, CheMPS2, dkh, gdma, PCMSolver, simint

PY-FRIENDLY USERS:
- CONDA PACKAGE
  PSI4 · dependencies · add-ons

PY-WARY USERS:
- CONDA INSTALLER
  PSI4 · dependencies · add-ons

CORE DEVELOPERS:
- CONDA-ENABLED SUPERBUILD
  deps · dev tools · add-ons · cmake hints

DOWNSTREAM
- Package Manager: conda
  - v2rdm_casscf
CONDA FOR USERS & DEVELOPERS

UPSTREAM
- Build Tools: cmake, compilers, pybind11, MKL-devel
- Required C-link non-QC: MKL, OpenMP, HDF5
- Required Py-link QC: gau2grid, Libint, Libxc
- CMake Helper: bin/psi4-path-advisor
- Req’d Py-link non-QC: Python, NumPy, networkx, deepdiff, pint, pytest
- Required Py-link QC: QCElemental
- Optional C-link QC: ambit, CheMP52, dkh, gdma, PCMSolver, simint
- Optional C-link Psi4: v2rdm_casscf
- Package Manager: conda
- Optional Py-link QC: libefp<PylibEFP, DFTD3, gCP
- Optional Py-link Psi4: OpenFermion<OF-Psi4, resp, snsmp2

PSI4-DEV

PSI4

Downstream
- PSI4: dev tools · cmake hints
- PSICONDA

Peripherale Devel’s:
Conda-Facilitated Plugins
PSI4 · dev tools · cmake hints

Core Developers:
Conda-Enabled Superbuild
deps · dev tools · add-ons · cmake hints

Py-Friendly Users:
Conda Package
PSI4 · dependencies · add-ons

Py-Wary Users:
Conda Installer
PSI4 · dependencies · add-ons
PRODUCTION DISTRIBUTIONS

- LAPACK dynamically link runtime-multiarch MKL direct from Intel.
- MULTIARCH libs optimized for both modern & legacy arch via icpc flags.
- STANDARDS easy to distribute with gcc7.3, so devs can use c++14.
- COMPATIBILITY conda compilers have sysroots with old glibc so useable with slow-moving Linux OS. Binary-compatible w/ defaults & conda-forge channels.

STANDARDS

- SHINY THINGS advance standards liberally.
  - balance with user ease
  - balance with distribution ease
  - balance with not imposing version freeze on other projects
- PYTHON 3.6+ after mid-2018.
- C++ 14 by Dec 2018.
- CONDA COMPILERS
  - MAC Clang 4.0.1 for C, C++; GCC Fortran.
  - WIN MSVC for C, C++; IFORT for Fortran.
DEclarative interface
ALLOWS TRANSPARENT REFACTORIZATION BETWEEN MODULES, LANGS, REPOS

- **ENERGY()**, **gradient()**, **optimize()**, **hessian()**, **frequency()** are the five "user-facing" functions through which 99+% of QC is run in Psi, so easy to guess command. Minimal entry points

- **BEST-PRACTICE OPTIONS** for basis sets, convergence, implementation, algorithm are added at driver layer.

- **PY/C++** interface layer below the user layer makes it simple to shift methods between languages without disturbing user.
  - **PY → C++** when a reference implementation is optimized in compiled language.
  - **C++ → PY** when a legacy code is refactored so that logic in Python and intensive parts in compiled.

- **E.G. MOLECULE** parsing
  - C++ with Boost regex on str (v1.0)
  - C++ with C++11 regex on str (v1.1)
  - Py dict initializing class (v1.2)
  - Py dict from external module initializing class (v1.3)

- **E.G. B3LYP**
  - In-house fctl + lib3index DF + Wfn::KS (v1.1)
  - Libxc fctl + lib3index DF + Wfn::KS
  - Libxc fctl + lib3index DF + Wfn::SCF
  - Libxc fctl + DF_HELPER DF + Wfn::SCF (v1.2)

- **psi4.energy('mp2')**
- **psi4.energy('mp2/cc-pvtz')**
- **psi4.energy('mp2/cc-pv[dt]z')**
- **psi4.gradient('mp2/cc-pv[dt]', bsse_type='cp')**
- **psi4.optimize('mp2/cc-pv[dt]', bsse_type='uncp', dertype=0)**
DECLARATIVE INTERFACE
ALLOWS TRANSPARENT REFACTORING BETWEEN MODULES, LANGS, REPOS

ENERGY(), gradient(), optimize(), hessian(), frequency() are the five “user-facing” functions through which 99+% of QC is run in Psi, so easy to guess command. Minimal entry points

BEST-PRACTICE OPTIONS for basis sets, convergence, implementation, algorithm are added at driver layer.

PY/C++ interface layer below the user layer makes it simple to shift methods btwn languages without disturbing user.

PY → C++ when a reference implementation is optimized in compiled language.

C++ → PY when a legacy code is refactored so that logic in Python and intensive parts in compiled.

E.G. MOLECULE parsing

C++ with Boost regex on str (v1.0)
C++ with C++11 regex on str (v1.1)
Py dict initializing class (v1.2)
Py dict from external module initializing class (v1.3)

E.G. B3LYP

in-house fctl + lib3index DF + Wfn::KS (v1.1)
Libxc fctl + lib3index DF + Wfn::KS
Libxc fctl + lib3index DF + Wfn::SCF
Libxc fctl + DF_Helper DF + Wfn::SCF (v1.2)

REVOLUTIONS

wavefunction passing - localizing, in-memory, giving user access to calc innards – 2016
molecule passing - globals avoidance – 2016
recursivedriver, minimal entry points – 2016
CMakeRewrite – 2016
dependency ejection (build-wise) – 2016
KillTheBoost – 2016
HistoryRewrite – 2016
pysidescf – 2018
molparse – 2018
qcvar localization - globals avoidance, definition coherency – 2018-present
distributeddriver – 2018-present
dependency ejection (interface-wise to QCA stack) – 2019-present
theBeheading – upcoming
options passing - globals avoidance – upcoming
**QCSchema**

FULL-FLEDGED SINGLE JOB SPEC

**QCJOB DICT**

- **CHARACTERISTICS**
  - describes single QC step in unified language
  - DICT, non-serializable, conforms to schema
  - filled by translator functions

- **CONTENTS**
  - **DIRECTIONS** QCprog, method, basis, deriv level
  - **OPTIONS** multilevel, history, Py-format values
  - **RESOURCES** exe loc, scratch, mem, threads
  - **OUTPUTS** stdout, qcvars, interpret-time errors

DISCUSSION HERE

VALIDATING PYTHON OBJECTS HERE
QCSchema
FULL-FLEDGED SINGLE JOB SPEC

QCJOB DICT

- CHARACTERISTICS
  - describes single QC step in unified language
  - DICT, non-serializable, conforms to schema
  - filled by translator functions

- CONTENTS
  - DIRECTIONS QCprog, method, basis, deriv level
  - OPTIONS multilevel, history, Py-format values
  - RESOURCES exe loc, scratch, mem, threads
  - OUTPUTS stdout, qcvars, interpret-time errors

---

```
{
  'driver': 'gradient',
  'model': {'method': 'ccsd(t)',
             'basis': 'cc-pvdz'},
  'molecule': {'atomic_numbers': [8, 1, 1],
               'geometry': [0.0, 0.0, -0.06563853809917809, ...],
               'fix_symmetry': 'Cs'},
  'keywords': {'freeze_core': True,
               ...}
  'r_convergence': 8},

  'provenance': None,
  'success': None
}
```
QCSchema
FULL-FLEDGED SINGLE JOB SPEC

QCJOB DICT

- CHARACTERISTICS
  - describes single QC step in unified language
  - DICT, non-serializable, conforms to schema
  - filled by translator functions

- CONTENTS
  - DIRECTIONS QCprog, method, basis, deriv level
  - OPTIONS multilevel, history, Py-format values
  - RESOURCES exe loc, scratch, mem, threads
  - OUTPUTS stdout, qcvars, interpret-time errors

---

{'driver': 'DITTO', 'model': 'DITTO', 'molecule': 'DITTO', 'keywords': 'DITTO', 'provenance': {'by': 'Psi4', 'func': 'run_json', 'version': '1.3'}, 'raw_output': '
A miracle come to pass. The CC iterations have converged.

  CCSD(T) energy                           -76.320175532159

Cfour scratch file GRD has been read

  3        0.0000000000
  8.0000000000        0.0000000000        0.0000000000        0.0006534633
  1.0000000000        0.0000000000       -0.0003521950       -0.0003267317
  1.0000000000        0.0000000000        0.0003521950       -0.0003267317

extras': {'qcvars': {
    'CCSD CORRELATION ENERGY': -0.2503305491879538,
    'CCSD(T) TOTAL ENERGY': -76.320175532159,
    'CURRENT GRADIENT': [[[0.0, 0.0, 0.0], [0.0, -0.0003522, -0.00032673]], [0.0, 0.0003522, -0.00032673]]}}

'success': True
**QCSchema**

**FULL-FLEDGED SINGLE JOB SPEC**

**QCJOB DICT**
- **CHARACTERISTICS**
  - describes single QC step in unified language
  - DICT, non-serializable, conforms to schema
  - filled by translator functions
- **CONTENTS**
  - DIRECTIONS QCprog, method, basis, deriv level
  - OPTIONS multilevel, history, Py-format values
  - RESOURCES exe loc, scratch, mem, threads
  - OUTPUTS stdout, qcvars, interpret-time errors

---

**Psi4**

C++/PYTHON LIBRARY

**Python Module w/C++ Extension**

---

**PSITHON**

**PSIAPI**

**INPUT PARSER**

**JSON**

**YAML**
III. Narrow data connections by replacing API calls with JSON call.
III. Narrow data connections by replacing API calls with JSON call.
III. Narrow data connections by replacing API calls with JSON call.

QElemental
- **FUNDAMENTALS** context-switchable API to NIST periodic table, NIST CODATA, MOLECULE parsing, validation and enforcement.
- **QCSCHEMA** validation and enforcement.
- **MOLECULE** parsing, validation, export.
- **context-switchable** API context.

QCEngine
- **QCSCHEMA** runner. Either light wrapper for projects that speak QCSchema or input-writer/executor/output-parser.
- **SUBPROCESS** runner utilities.
- **ECOSYSTEM** testing utilities.
- **COMPASS** runner manager & collector.
- **DEP** reader & collector.

Testing and generic utilities.

UPSTREAM
- **libefp**
- **DFTD3**
- **geomeTRIC**

FUNDAMENTALS
- **context-switchable** API

MOLECULE
- **parsing**, validation, export.

QCSCHEMA
- **validation** and enforcement.

ECOSYSTEM
- **testing** utilities.

III. Narrow data connections by replacing API calls with JSON call.

DEP
- **reading DEP**
III. Narrow data connections by replacing API calls with JSON call.
DISTRIBUTED DRIVER

```python
def energy (mtd):
def gradient (mtd):
def hessian (mtd):
```
class SingleResult():

molecule & method & func unchanged. return json

PLAN

ASM

Return analytic energy, gradient, or Hessian.
class **NBodyComputer** ()::

Separate molecule into subsystems. CP, noCP, VMFC basis. method & func unchanged.

```
for frag in fragments: return json
```

Assemble n-body & interaction results from fragments.

class **SingleResult** ()::

```
molecule & method & func unchanged. return json
```

Return analytic energy, gradient, or Hessian.
def energy (mtd):

def gradient (mtd):

def hessian (mtd):

class NBodyComputer ():

Separate molecule into subsystems. CP, noCP, VMFC basis. method & func unchanged.

for frag in fragments: return json

Assemble n-body & interaction results from fragments.

class CBSComputer ():

Separate method into method, basis, & extrapolations. molecule & func unchanged.

for mc in modelchems: return json

Assemble extrapolations & total results from modelchems.

class SingleResult ():

molecule & method & func unchanged. return json

Return analytic energy, gradient, or Hessian.
class NBodyComputer ():

PLAN
Separate molecule into subsystems. CP, noCP, VMFC basis.
method & func unchanged.

```python
def energy (mtd):
    for frag in fragments: return json
```

ASM
Assemble n-body & interaction results from fragments.

class CBSComputer ():

PLAN
Separate method into method, basis, & extrapolations.
molecule & func unchanged.

```python
def hessian (mtd):
    for mc in modelchems: return json
```

ASM
Assemble extrapolations & total results from modelchems.

class FinDifComputer ():

PLAN
Displace molecule according to stencil.
Reference molecule & func unchanged.

```python
def hessian (mtd):
    for disp in displacements: return json
```

ASM
Assemble derivative results from displacements.

class SingleResult ():

PLAN
molecule & method & func unchanged. return json

ASM
Return analytic energy, gradient, or Hessian.
class NBodyComputer ():
  Separate molecule into subsystems. CP, noCP, VMFC basis. method & func unchanged.
  Assemble n-body & interaction results from fragments.

class CBSComputer ():
  Separate method into method, basis, & extrapolations. molecule & func unchanged.
  Assemble extrapolations & total results from modelchems.

class FinDifComputer ():
  Displace molecule according to stencil. Reference molecule & func unchanged.
  Assemble derivative results from displacements.

class SingleResult ():
  molecule & method & func unchanged. return json
  Return analytic energy, gradient, or Hessian.
**DISTRIBUTED DRIVER**

```python
def energy(mtd):
    def gradient(mtd):
        def hessian(mtd):
            class FinDifComputer():
                Displace molecule according to stencil.
                Reference molecule & func unchanged.
                Assemble derivative results from displacements.
            class CBSComputer():
                Separate method into method, basis, & extrapolations.
                molecule & func unchanged.
                Assemble extrapolations & total results from modelchems.
            class NBodyComputer():
                Separate molecule into subsystems. CP, noCP, VMFC basis.
                method & func unchanged.
                Assemble n-body & interaction results from fragments.
            class SingleResult():
                molecule & method & func unchanged.
                return json
```

**PSI4**

C++/PYTHON LIBRARY

LGPL-3.0

---

**PLAN**

Separate molecule into subsystems. CP, noCP, VMFC basis.
method & func unchanged.

```python
for frag in fragments: return json
```

**ASM**

Assemble derivative results from displacements.

---

**PLAN**

Separate method into method, basis, & extrapolations.
method & func unchanged.

```python
for mc in modelchems: return json
```

**ASM**

Assemble extrapolations & total results from modelchems.

---

**PLAN**

Displace molecule according to stencil.
Reference molecule & func unchanged.

```python
for disp in displacements: return json
```

**ASM**

Assemble n-body & interaction results from fragments.

---

**PLAN**

molecule & method & func unchanged. return json

**ASM**

Return analytic energy, gradient, or Hessian.

---

**JSON:** QCSchema

---

**MP2 TOTAL ENERGY/cc-pVTZ**

**MP2 TOTAL ENERGY/cc-pVQZ**

---

**Single FinDif**

**Single NBody**

**CBS**

**Single**

---

**exion**

---

**abc**

---

**def**

---

**gh**
def energy (mtd):

def gradient (mtd):

def hessian (mtd):

class FinDifComputer ():
Displace molecule according to stencil. Reference molecule & func unchanged.

Assemble derivative results from displacements.

class CBSComputer ():
Separate method into method, basis, & extrapolations. molecule & func unchanged.

Assemble extrapolations & total results from modelchems.

class NBodyComputer ():
Separate molecule into subsystems. CP, noCP, VMFC basis. method & func unchanged.

Assemble n-body & interaction results from fragments.

class SingleResult ():
molecule & method & func unchanged. return json

Return analytic energy, gradient, or Hessian.
DISTRIBUTED DRIVER

def energy (mtd):
    def gradient (mtd):
        def hessian (mtd):

class FinDifComputer ():
    Displace molecule according to stencil.
    Reference molecule & func unchanged.
    Assemble derivative results from displacements.

class CBSComputer ():
    Separate method into method, basis, & extrapolations.
    molecule & func unchanged.
    Assemble extrapolations & total results from modelchems.

NBodyComputer ()

class SingleResult ()

DISTRIBUTED DRIVER

```python
def energy(mtd):

def gradient(mtd):

def hessian(mtd):
```

**class FinDifComputer:**
- Displace molecule according to stencil.
- Reference molecule & func unchanged.
- Assemble derivative results from displacements.

**class CBSComputer:**
- Separate method into method, basis, & extrapolations.
- molecule & func unchanged.
- Assemble extrapolations & total results from modelchems.

**class NBodyComputer:**
- Separate molecule into subsystems. CP, noCP, VMFC basis.
- method & func unchanged.
- Assemble n-body & interaction results from fragments.

**class SingleResult:**
- molecule & method & func unchanged. return json
- Return analytic energy, gradient, or Hessian.
class NBodyComputer ():

PLAN
Separate molecule into subsystems. CP, noCP, VMFC basis. method & func unchanged.

for frag in fragments: return json

ASM
Assemble n-body & interaction results from fragments.

class CBSComputer ():

PLAN
Separate method into method, basis, & extrapolations. molecule & func unchanged.

'mp2/cc-pv[tq]z' MP2 TOTAL ENERGY/cc-pVTZ
for mc in modelchems: return json

ASM
Assemble extrapolations & total results from modelchems.

class FinDiffComputer ():

PLAN
Displace molecule according to stencil. Reference molecule & func unchanged.

for disp in displacements: return json

ASM
Assemble derivative results from displacements.

class SingleResult ():

PLAN
molecule & method & func unchanged. return json

ASM
Return analytic energy, gradient, or Hessian.
Because … works.

def energy(mtd):

def gradient(mtd):

def hessian(mtd):

class FinDifComputer():
    Displace molecule according to stencil.
    Reference molecule & func unchanged.
    Assemble derivative results from displacements.

class CBSComputer():
    Separate method into method, basis, & extrapolations.
    molecule & func unchanged.
    Assemble extrapolations & total results from modelchems.

class NBodyComputer():
    Separate molecule into subsystems. CP, noCP, VMFC basis.
    method & func unchanged.
    for frag in fragments: return json

    Assemble n-body & interaction results from fragments.

class SingleResult():
    molecule & method & func unchanged.
    return json

Return analytic energy, gradient, or Hessian.
psi4.optimize('HF/cc-pv[d,t]z', bsse_type='cp', molecule=)

def psi_model(coords):
    dimer = psi4.ani4.prim('HF/cc-pv[D,T]Z', bsse_type='CP', molecule=dimer, return_plan=True, return_total_data=True)
    plan.compute(client)
    snowflake.await_results()
    ret = plan.get_results(client)
    return (ret['extras']['qcvars']['CURRENT ENERGY'],
            np.array(ret['extras']['qcvars']['CURRENT GRADIENT']).reshape(-1, 3))
psi4.optimize('HF/cc-pv[d,t]z', bsse_type='cp', molecule= )
IV. Polish separable tools from Psi4 and release to community.
IV. Polish separable tools from Psi4 and release to community.
IV. Polish separable tools from Psi4 and release to community.
IV. Polish separable tools from Psi4 and release to community.
qcdb.set_options({
    'memory': '35 gb',
    'e_convergence': 1e-11,
    'scf__d_convergence': 1e-9,
    'nwchem_ccsd__maxiter': 1e-9,
    'psi4_mp2_type': 'conv',
    'psi4_scf_type': 'direct',
    'psi4_df_scf_guess': 'false',
})

```python
# ccstq correction: (CCSDTQ - CCSD(T)) / cc-pVDZ
qcdb.set_options({'four_dropno': [1,]})
E_csdtdq = float[jrec['qcvars']]/[CCSDTQ TOTAL ENERGY].data
E_csdtdp = float[jrec['qcvars']]/[CCSDTQ TOTAL ENERGY].data
qcdb.set_options({'four_dropno': None})
de_csdtdq[i] = E_csdtdq - E_csdtdp
print('ccsdqd Correction=de_csdtdq[-1] Har. {{1+1}/(npoints)} ~')
# base calculation: CCSD(T) cc-pCV[Q5]Z
qcdb.set_options({'memory': '30 gb'})
E_base[i] = E
print('base Energy=E Har. {{1+1}/(npoints)} ~')
# basis set correction: MP3 / (aug-cc-pCV[56]Z - cc-pCV[Q5]Z)
E large, = ccdb.energy('p4-mp2/aug-cc-pCV[Q5]Z', return_wfn=True)
de_basis[i] = E large - E small
print('basis Correction=de_basis[-1] Har. {{1+1}/(npoints)} ~')
# relativistic correction: (X2C-CCSD(T) - CCSD(T)) / cc-pVTZ-DK
qcdb.set_options({'psi4_relativistic': 'x2c'})
E_x2c_on[i] = ccdb.energy('p4-ccsd/aug-cc-pVTZ-DK', return_wfn=True)
de_x2c_on[i] = E_x2c_on - E_x2c_off
print('Relativistic Correction=de_x2c[-1] Har. {{1+3}/(npoints)} ~')
# fci correction: (F CI - CCSD(T)) / cc-pVDZ
E_fci[i] = ccdb.energy('gms-ccsd(fci)/cc-pVDZ', return_wfn=True)
de_fci[i] = E_fci - E
print('F CI Correction=de_fci[-1] Har. {{1+1}/(npoints)} ~')
```

```
phys_consts_tot_fci = psi4.diatomic.anharmonicity(R_arr, E_tot_fci)
```
QCA & QCDB ACKNOWLEDGEMENTS

QCArchive
- Daniel Smith
  - MolSSI
- Levi Naden
  - MolSSI
- Theresa Windus
  - Iowa State
- Jiyoun Lee
  - Iowa State
- Annabelle Lolinco
  - Iowa State
- Mark Gordon
  - Iowa State
- Nuwan de Silva

CFour
- John Stanton
  - UFL
- Devin Matthews
  - UTexas
- David Sherrill
  - GaTech
- Daniel Smith
  - MolSSI
- Zach Glick
  - GaTech
- Asim Alenaizan
  - GaTech
- Zach Glick
  - GaTech

NWChem

Psi4

Demos

GAMESS

QCA

PSI4

GAMESS

CFOUR

DEMOS

GAMESS

QMacie

Psi4
QCA & QCDB Acknowledgements

QCArchive
- Daniel Smith
  - MolSSI
- Levi Naden
  - MolSSI
- Theresa Windus
  - Iowa State
- Jiyoung Lee
  - Iowa State
- Annabelle Lolinco
  - Iowa State
- Mark Gordon
  - Iowa State
- Nuwan de Silva
  - MolSSI

CFOUR
- John Stanton
  - UFL
- Devin Matthews
  - UTexas
- David Sherrill
  - GaTech
- Daniel Smith
  - MolSSI

NWChem

GAMESS

Psi4

SHERRILL GROUP
QCA & QCDB ACKNOWLEDGEMENTS

Daniel Smith
MolSSI

Levi Naden
MolSSI

Theresa Windus
Iowa State

Ben Pritchard
MolSSI
QCA & QCDB ACKNOWLEDGEMENTS

Daniel Smith
MolSSI

Levi Naden
MolSSI

Theresa Windus
Iowa State

Ben Pritchard
MolSSI