Python ecosystem for scientific computing with ABINIT: challenges and opportunities

M. Giantomassi and the AbiPy group

Frejus, May 9, 2017

- Python package for:
  - generating input files automatically
  - post-processing results stored in netcdf files
  - creating and executing workflows (relaxations, phonons, GW, BSE…)

- Why Python?
  - relatively easy to use and to learn
  - can implement logic to drive high-performance/high-throughput jobs
  - great support for science (numpy, scipy, pandas, matplotlib…)
  - interactive environment (ipython, jupyter notebooks, GUIs)
  - pymatgen ecosystem and materials project database …

pymatgen

Computational Materials Science 68 (2013) 314–319

Contents lists available at SciVerse ScienceDirect

Computational Materials Science

journal homepage: www.elsevier.com/locate/commatsci

Python Materials Genomics (pymatgen): A robust, open-source python library for materials analysis

Shyue Ping Ong [a,*], William Davidson Richards [a], Anubhav Jain [b], Geoffroy Hautier [c], Michael Kocher [b], Shreyas Cholia [b], Dan Gunter [b], Vincent L. Chevrier [d], Kristin A. Persson [b], Gerbrand Ceder [a]

- Classes for the representation of Molecules and Structures
- Structure manipulation
- CIF file and XYZ format support
- IO capabilities to manipulate many VASP and ABINIT input and output files
- Tools to generate and view compositional and grand canonical phase diagrams
- Electronic structure analyses
- Integration with the Materials Project Database (REST API)

http://pypi.python.org/pypi/pymatgen

abipy

Table Of Contents

This Page

Show Source

Quick search

Go

pypi package 0.2.0 | build passing | coverage 82% | license GPL

Release: 0.2.0
Date: May 03, 2017

Getting Started

- Feature Overview
- Getting AbiPy
- What's new in abipy

pip or conda packages

Post-processing

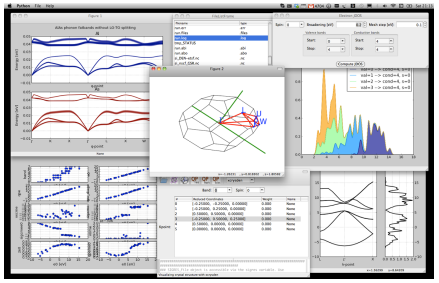
- Command line tools
- plot Examples

$ abistruct.py
$ abiopen.py
$ abicomp.py
$ abirun.py
$ abidoc.py

Calculations

- TaskManager

40 examples (electrons, fatbands, phonons, SCR.nc, WFK.nc, grunesein …



API

- API documentation
- The AbiPy Developers' Guide

Indices and tables

- Index
- Module Index
- Search Page

Explains how to connect AbiPy with Abinit

License

Fork me on GitHub

8 Jupyter notebooks with lessons

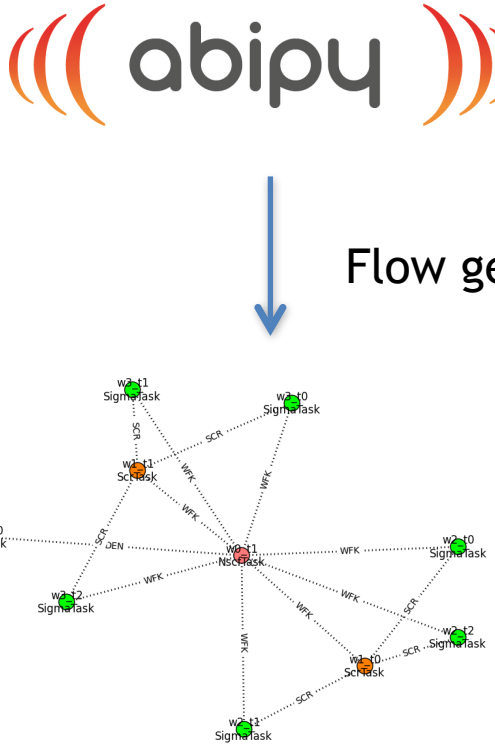
http://abinit.github.io/abipy

AbiPy Design

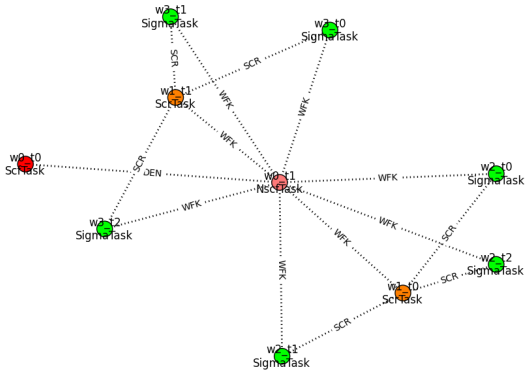
- AbiPy communicates with Abinit via yaml docs and netcdf files:
  - portable, can support “unconventional" machines e.g. BlueGeneQ
  - yaml “events” found in the logfile trigger handlers (error-handlers, restart …)
  - yaml docs with autoparal configurations, independent perturbations …
  - netcdf results + metadata used to implement post-processing/worflows

- Tight integration between Python and Abinit:
  - AbiPy invokes Abinit at run-time to get critical parameters
  - Programmatic interface to optimize/modify resources (mem, cpus, timelimit) at runtime to respond to software failure
  - AbiPy pass runtime parameters to Abinit e.g —time-limit

Two different workflow models (with/without database)

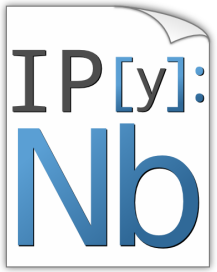


Flow generation

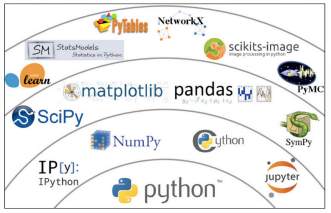


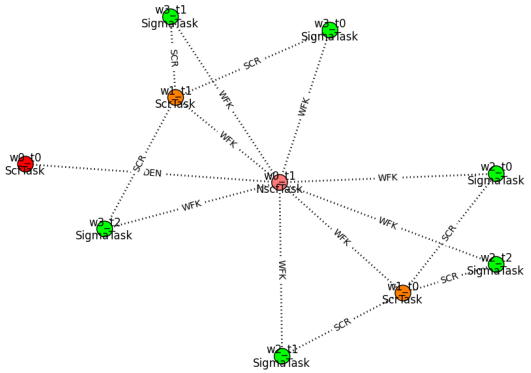
Scheduling the flow on the cluster
Error handling
Restart/increase resources …

SLURM
PBS/Torque
Sun Grid Engine
IBM LoadLeveler



Interactive analysis and presentation of results

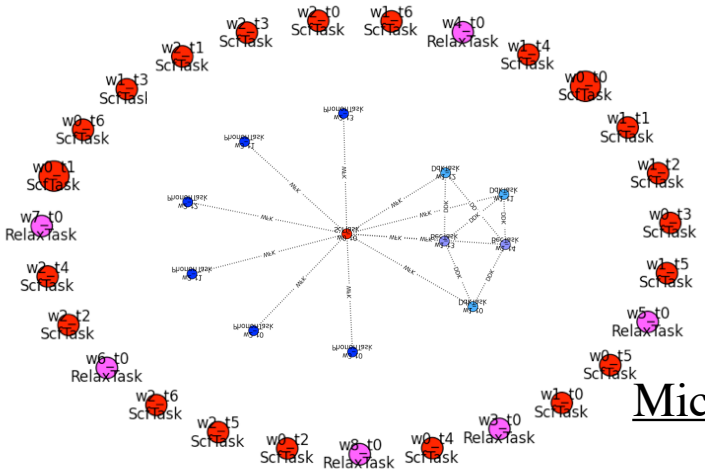




Storing results

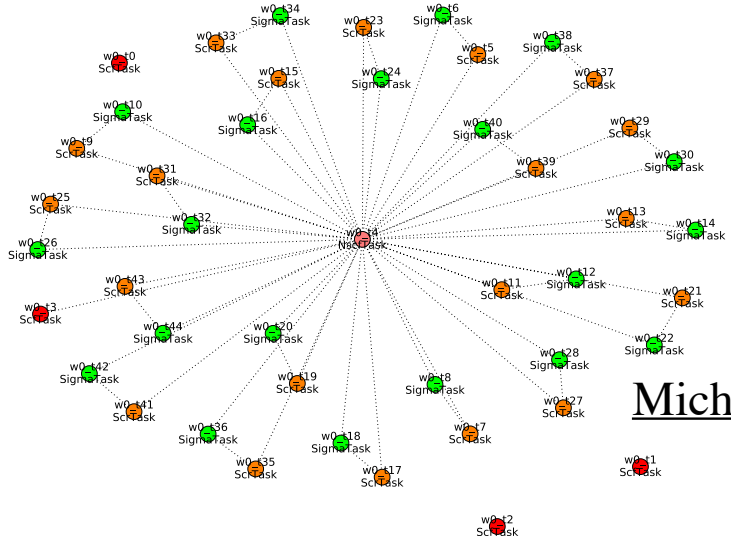
Abinit abipy FireWorks

PseudoDojo



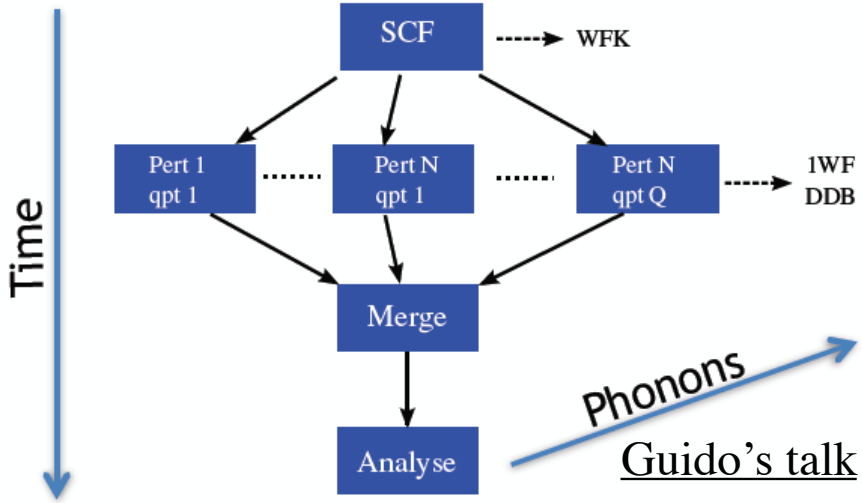
Michiel's talk

G_0W_0 for ~100 solids



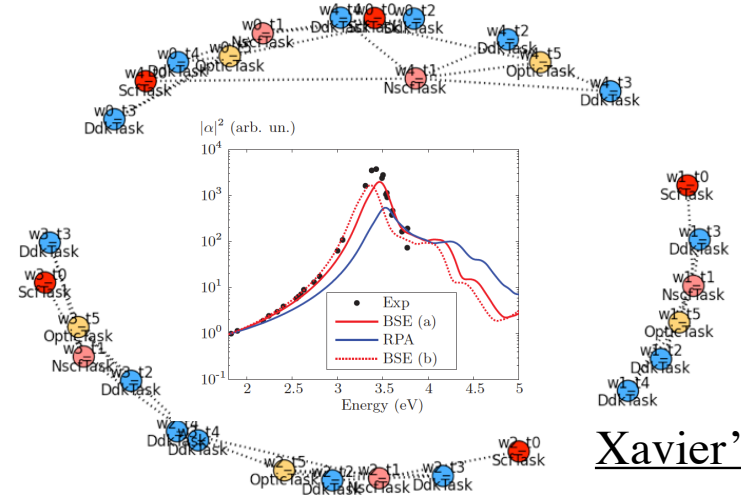
Michiel's talk

HT DFPT



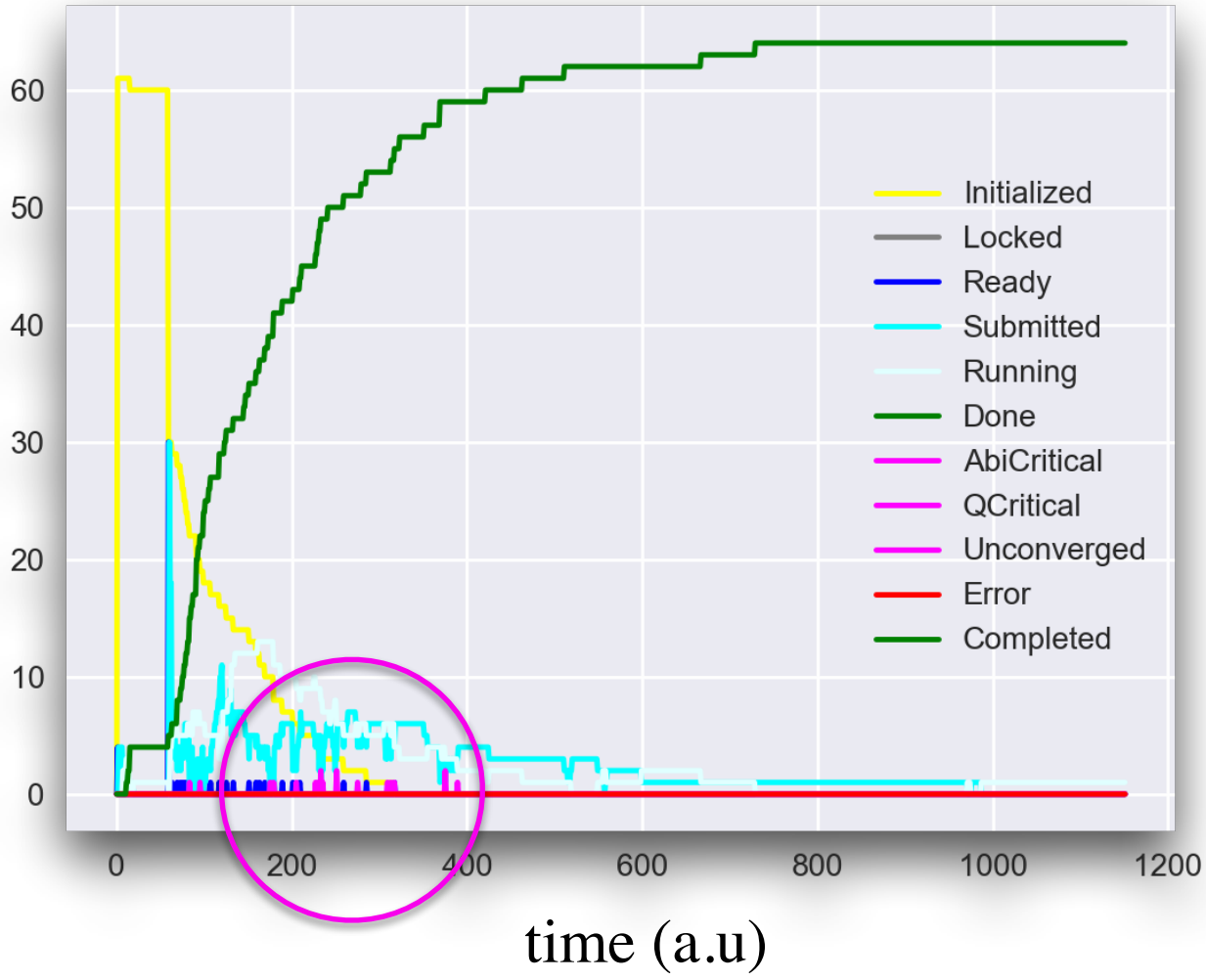
Guido's talk

Raman with frozen phonon and RPA/BSE



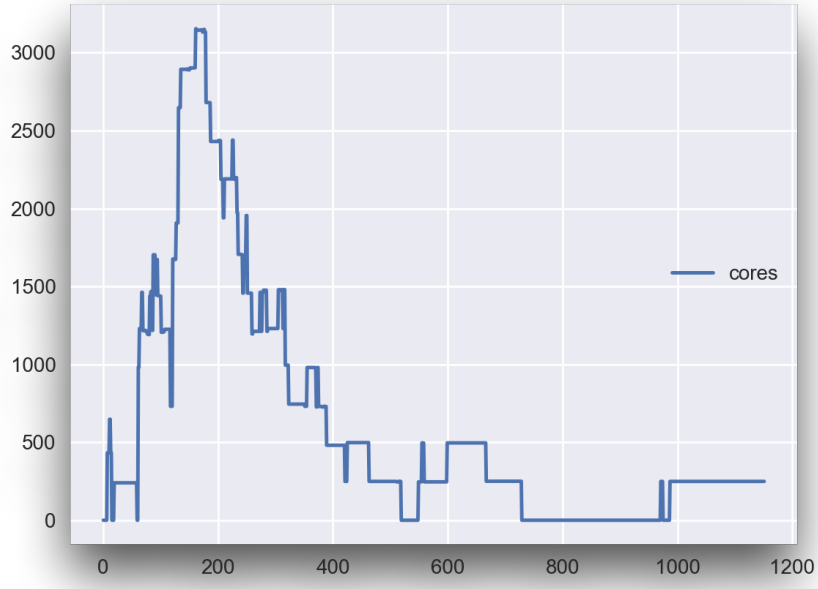
Xavier's talk

Evolution of tasks in a single GW flow



time (a.u)

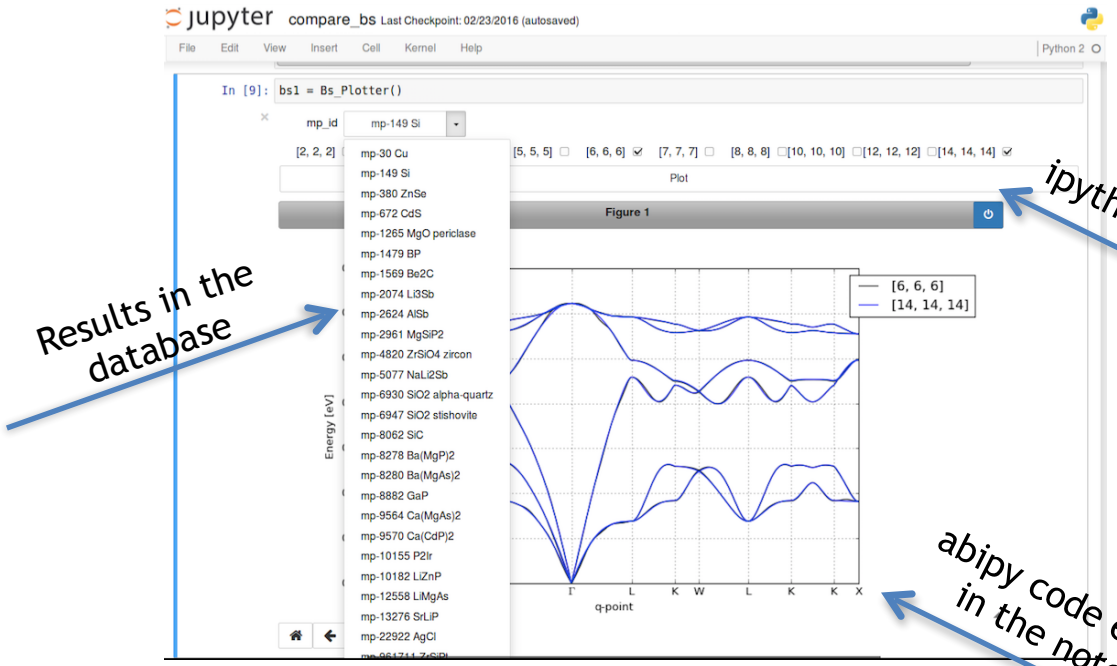
Tot #cpus used by a single GW flow



time (a.u)

jupyter

- Web application to create and share documents with live code, equations, visualizations and explanatory text
- Can produce rich output such as images, videos, LaTeX, and slides for talks!
- Make data analysis easier to share and reproduce. Keep detailed records of work
- Interactive widgets can be used to manipulate and visualize data in realtime
- Some researchers are even using notebooks as supplementary material (e.g. pseudo-dojo paper)

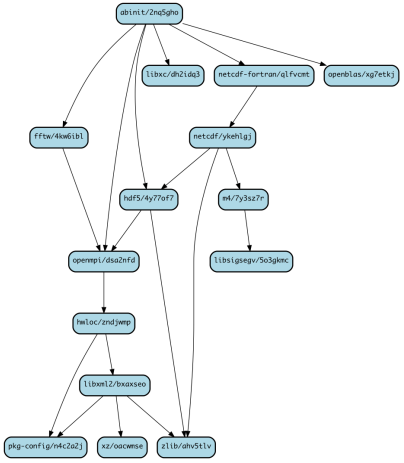


$ abiopen.py FILE -nb

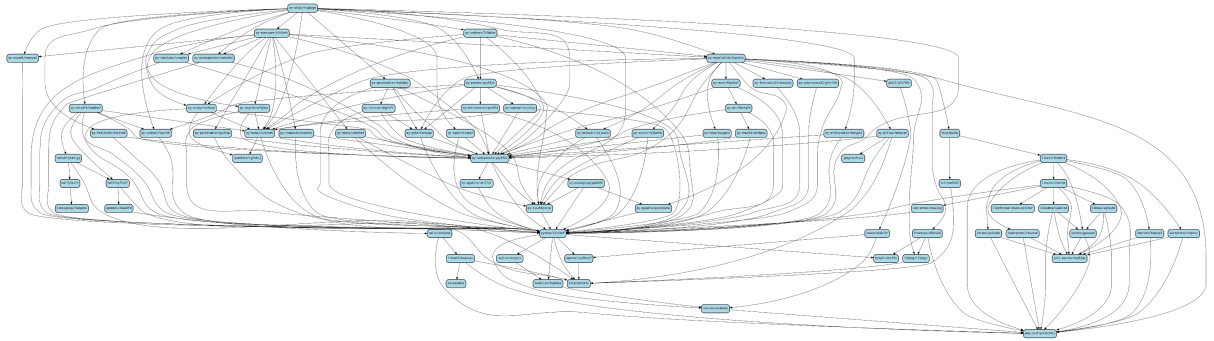
Challenges Issues

- Improve the accessibility of these tools to researchers and end-users
- Test/validate the Abinit/AbiPy interface and the workflows to guarantee:
    - i) Backward compatibility, software quality, scientific reproducibility
    - ii) That new developments don't break what is already available
- Make the python ecosystem coexist with the HPC software stack

Abinit (HPC libraries)



Abipy + Pymatgen
(usually gcc-toolchain)



Vendor-provided HPC libraries are not always compatible with PyData stack
(unless you love `configure && make` all the libraries you need!)

From: https://computation.llnl.gov/newsroom/flexible-package-manager-hpc-software

A Flexible Package Manager for HPC Software

FRIDAY, FEBRUARY 19, 2016

High-performance computing (HPC) software is becoming **increasingly complex**, quickly outpacing the capabilities of existing software management tools. To support scientific applications, system administrators and developers frequently build, install, and support **different configurations of math and physics libraries and other software** on the same HPC system. Those applications are later rebuilt to fix bugs and to support new operating system versions, compilers (special programs that turn programming language into code that can be used by a computer's processor), message passing interface (MPI) versions, and other dependency libraries. **Forcing all application teams to use a single, standard software stack […] is infeasible, but managing many software configurations and versions for all users on a single system is a time-consuming task for supercomputing staff.**

Existing tools can automate portions of this process, but they either cannot manage installation of multiple versions and configurations, or they require numerous configuration files for each software version, leading to organizational and maintenance issues.

- Handle multiple versions via modules
- Module load abinit_version_gcc-toolchain

https://github.com/hpcugent/easybuild

https://github.com/llnl/spack

- Uses RPATH linking (each package knows where to find its dependencies)
- $ spack activate abinit (abinit**+hdf5@8.2.2**)

★ Support for multiple versions and configurations of software

★ Installing a new version does not break existing installations, many configurations can coexist

- Recipes for Abinit/AbiPy are already available
- The Abinit test farm could use these tools to extend the set of tests and/or keep an history of the different builds
- What about desktop computers? What if I want to have full control of my software stack without having to install these HPC tools?

https://docs.continuum.io/anaconda

ANACONDA

- Package manager + environment manager (conda)
- Python distribution and collection of over 720 open source packages

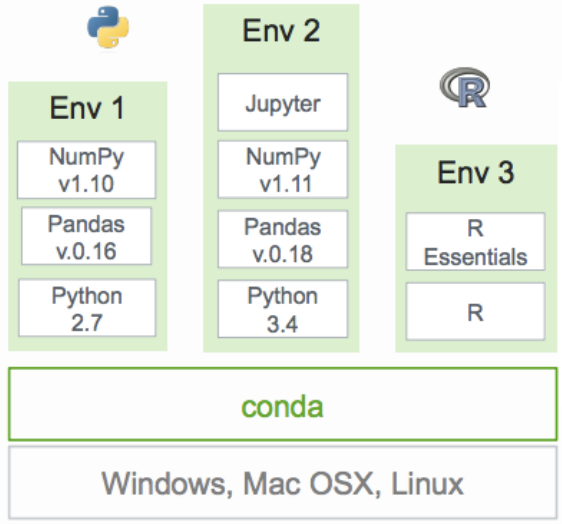
CONDA

- Created for Python programs, but can package and distribute any software
- Keeps track of the dependencies between packages and platforms
- A conda environment is a directory that contains a specific collection of conda packages that you have installed.
- You can easily activate or deactivate (switch between) these environments.
- Conda packages are downloaded from remote channels, which are simply URLs to directories containing conda packages

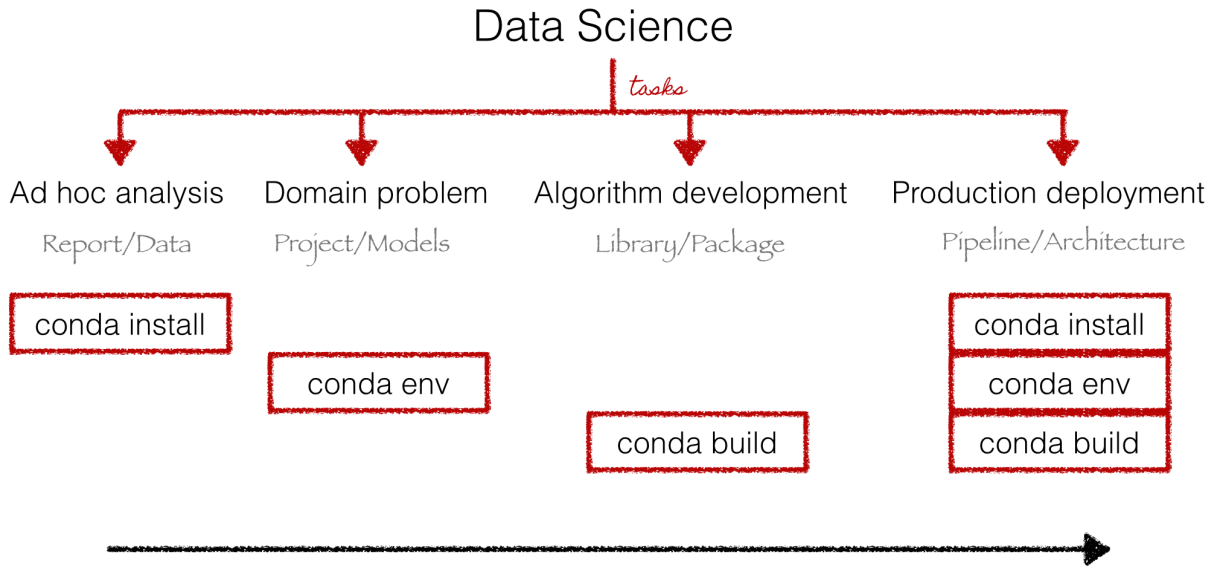
Conda: Package and Environment Management

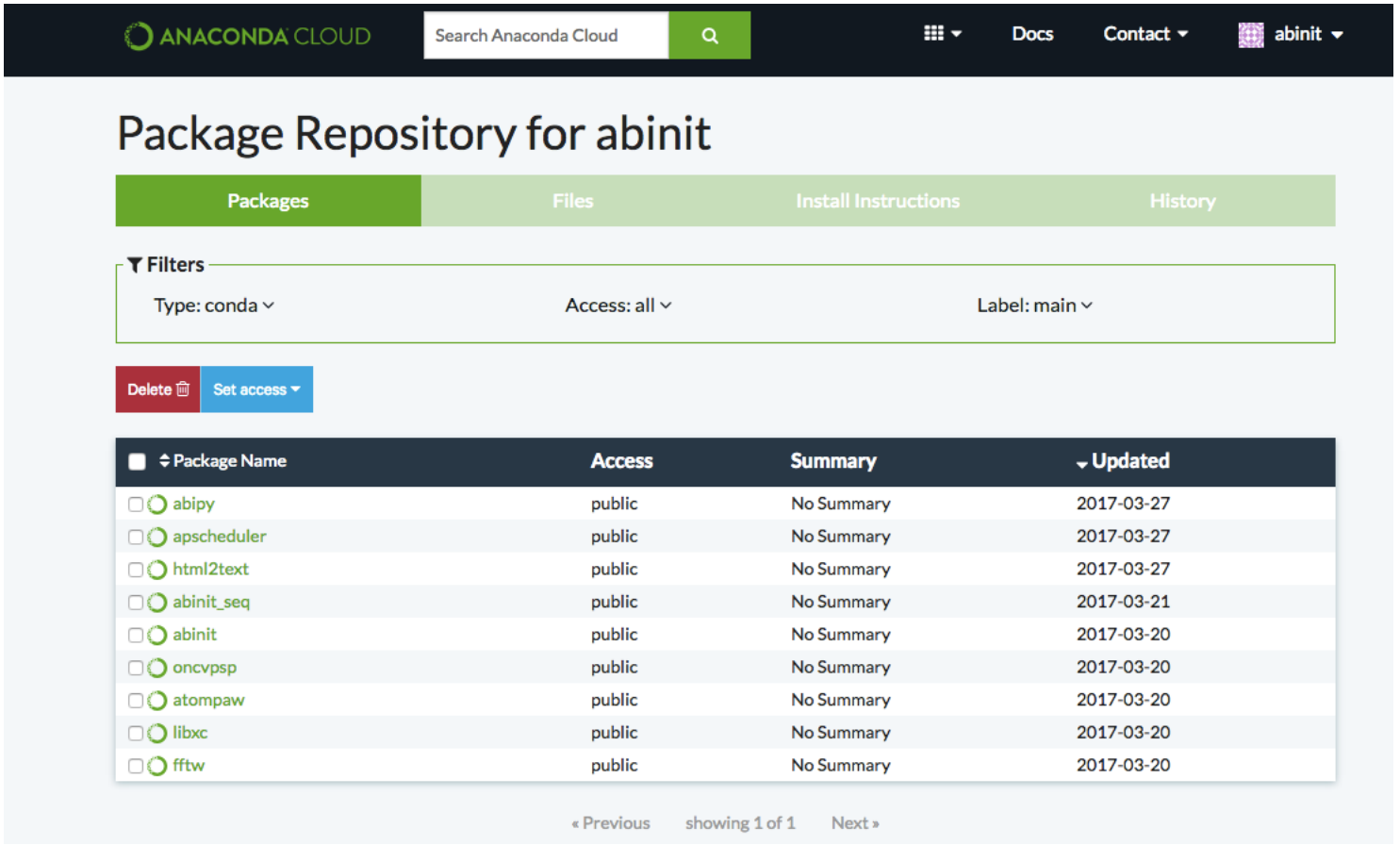
http://chdoig.github.io/pydata2015-dallas-conda

- Install packages
- Update packages
- Create sandboxes: Conda environments
- Conda environments: Critical for reproducibility, collaboration & scale

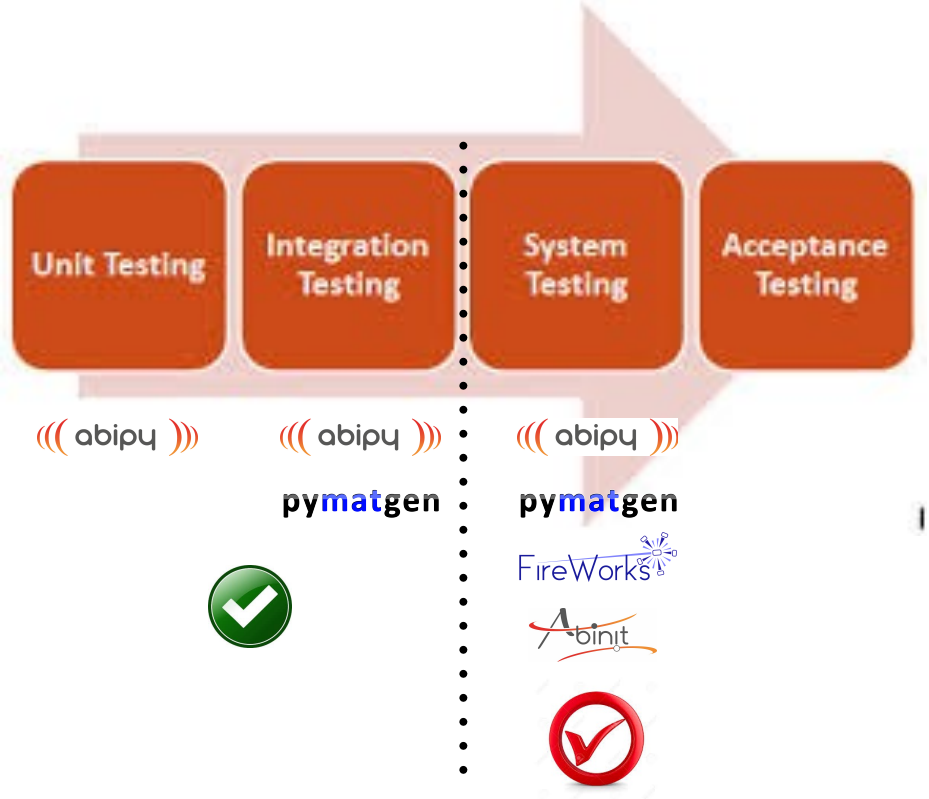


conda	pip (+ virtualenv)
language agnostic	python packages
handles environments natively	virtualenv
installs binaries	compiles from source
general purpose environments	python environments



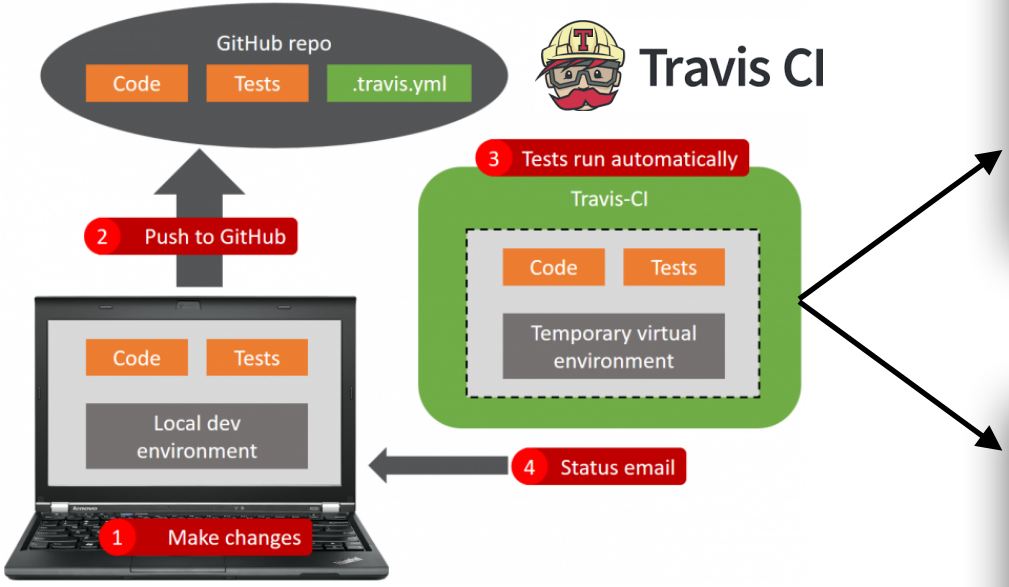


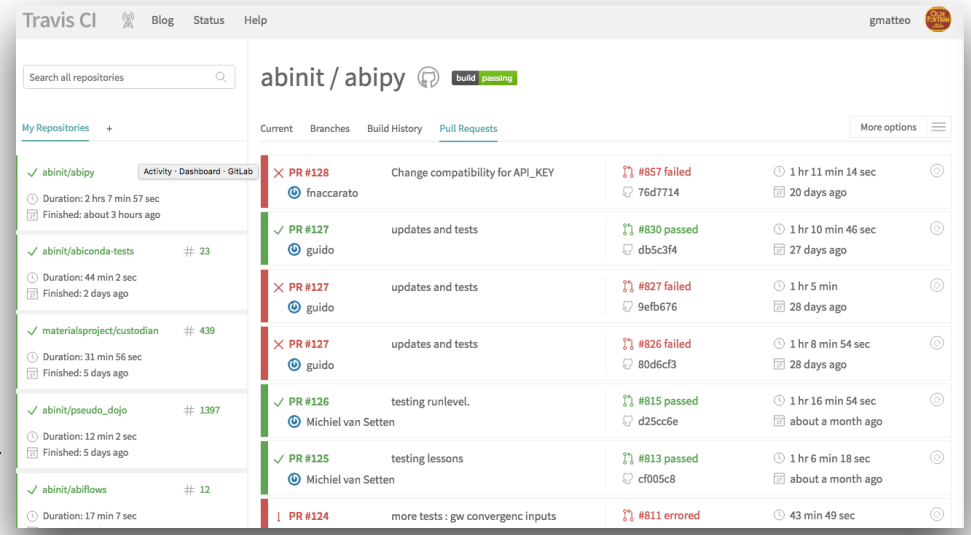
- Install abinit from scratch in 5 steps (see https://github.com/abinit/abiconda)
- Linux-64 and Osx-64 (sequential or parallel version)
- Can be used for schools and tutorials (not recommended for HPC)
- Cornerstone of the continuous integration infrastructure used by AbiPy …

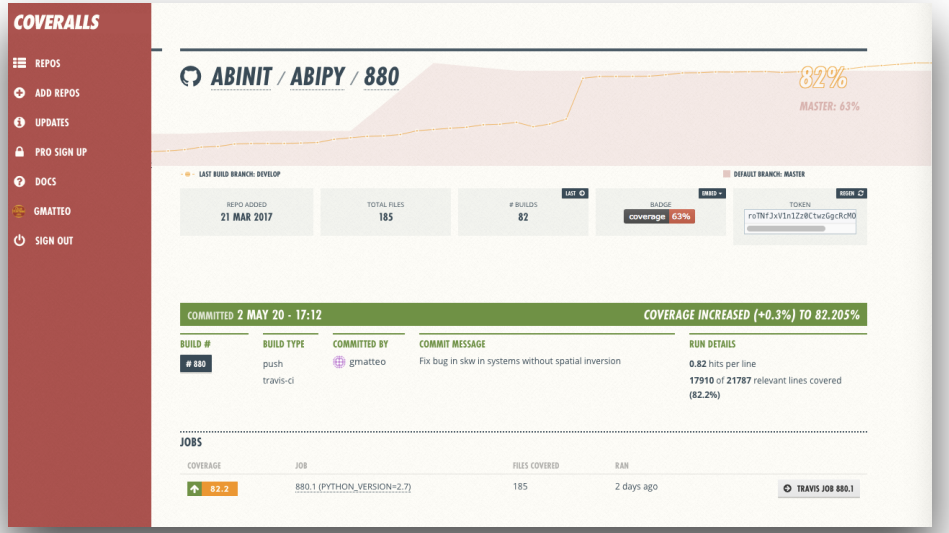


- Testing scientific software has never been easy (roundoff errors, parallelism …)
- Testing frameworks for automatic calculations is even more complicated:
  - Lots of "futures"
  - python code assumes a well-defined "response protocol" (output files in a given format, error messages …)
  - In principle one should validate the code in a typical production env (databases, resource managers, MPI execution in "exotic" architectures …)

Level1: unit tests and CI

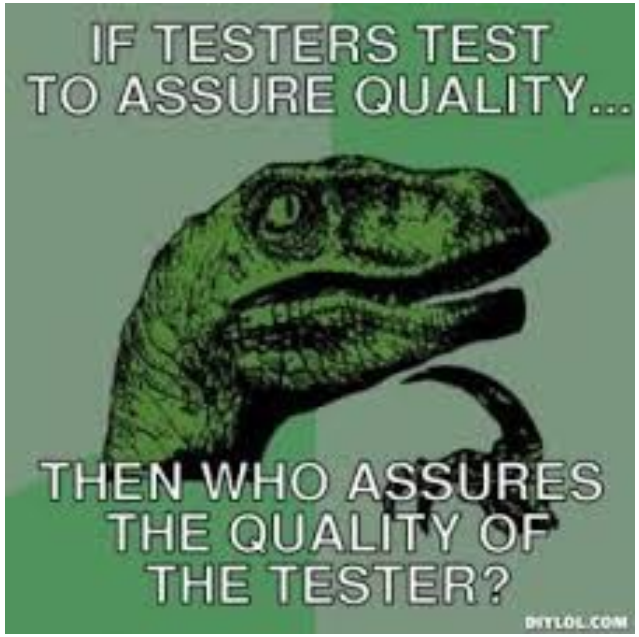






- 82% coverage
- abinit8.2.2 from conda vs Abipy@github:
  - ✓ validate input files
  - ✓ execute small workflows

Level2: integration with other python packages and software deployment



Goals:

- github repository to periodically test software integration
- pip/conda/github-repos with abinit8.2.2 binary from abinit channel

abiconda-tests:

```
Abinitbuild:
Abinit Build Information:
    Abinit version: 8.2.2
    MPI: True, MPI-IO: True, OpenMP: False
    Netcdf: True

Abipy Scheduler:
PyFlowScheduler, Pid: 2194
Scheduler options: {'seconds': 10, 'hours': 0, 'weeks': 0, 'minutes': 0, 'days': 0}

Installed packages:
Package         Version
--------------  ----------------------
system          Linux
python_version  2.7.13
numpy           1.12.1
scipy           0.19.0
netCDF4         1.2.7
apscheduler     2.1.0
pydispatch      2.0.5
yaml            3.12
pymatgen        4.7.4
matplotlib      2.0.0 (backend: Qt5Agg)


Abipy requirements are properly configured

Running small flow in workdir: /tmp/tmpWxnGY8

[Mon May  1 14:59:52 2017] Number of launches: 1

Work #0: <BandStructureWork, node_id=1, workdir=../../../../../tmp/tmpWxnGY8/w0>, Finalized=False
+--------+-------------+-------------+-------------+-----------+----------+----------------+----------+-----------+
| Task   | Status      | Queue       | MPI|Omp|Gb  | Warn|Com  | Class    | Sub|Rest|Corr  | Time     |   Node_ID |
+--------+-------------+-------------+-------------+-----------+----------+----------------+----------+-----------+
| w0_t0  | Submitted   | 2257@travis | 2|  1|2.0    | NA|NA     | ScfTask  | (1, 0, 0)      | 0:00:00Q |         2 |
+--------+-------------+-------------+-------------+-----------+----------+----------------+----------+-----------+
| w0_t1  | Initialized | None        | 1|  1|2.0    | NA|NA     | NscfTask | (0, 0, 0)      | None     |         3 |
+--------+-------------+-------------+-------------+-----------+----------+----------------+----------+-----------+

[Mon May  1 15:00:02 2017] Number of launches: 1

Work #0: <BandStructureWork, node_id=1, workdir=../../../../../tmp/tmpWxnGY8/w0>, Finalized=False
+--------+-----------+-------------+-------------+-----------+----------+----------------+----------+-----------+
| Task   | Status    | Queue       | MPI|Omp|Gb  | Warn|Com  | Class    | Sub|Rest|Corr  | Time     |   Node_ID |
+--------+-----------+-------------+-------------+-----------+----------+----------------+----------+-----------+
| w0_t0  | Completed | 2257@travis | 2|  1|2.0    | 4|  0     | ScfTask  | (1, 0, 0)      | 0:00:02R |         2 |
+--------+-----------+-------------+-------------+-----------+----------+----------------+----------+-----------+
| w0_t1  | Submitted | 2295@travis | 2|  1|2.0    | NA|NA     | NscfTask | (1, 0, 0)      | 0:00:00Q |         3 |
+--------+-----------+-------------+-------------+-----------+----------+----------------+----------+-----------+
```

Next step: monitor integration between Abinit@gitlab and AbiPy-stable

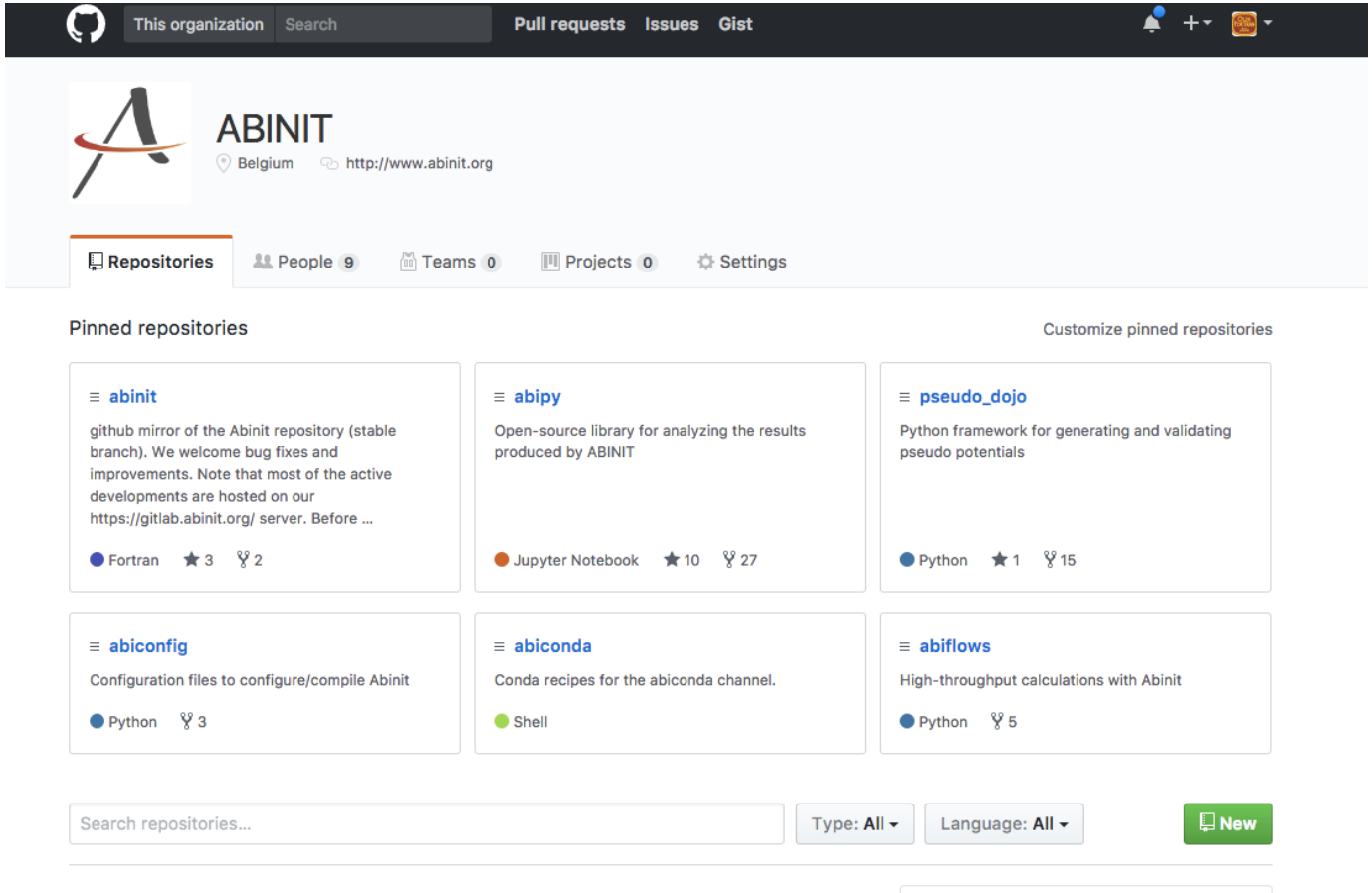
Goals:

- Buildbot slave to run AbiPy tests with Abinit-dev when merge-request is opened on gitlab
- Detect possible regressions or incompatibilities before we go in production

➡ Scientific code must be allowed to change and evolve to tackle new problems

➡ But now there's an entire ecosystem around Abinit and big changes must be rationalised and planned in advance (e.g. Abinit7 —> Abinit8)

Thank your for your attention



- pseudo_dojo: python package providing pseudos/workflows for tests
- abiconfig: configuration files for clusters
- abiconda: recipes to build abinit/abipy (matsci.sh provides pymatgen packages)
- abiflows: Fireworks workflows
- abinit_issues: