Los Alamos NATIONAL LABORATORY

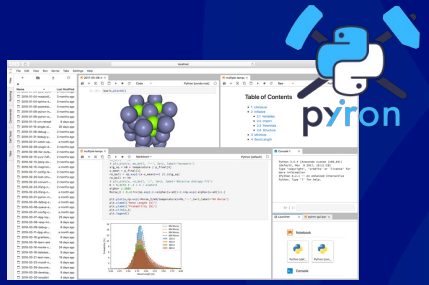
MAX-PLANCK-INSTITUT FÜR EISENFORSCHUNG GmbH

pyiron_base – pyiron without atomistics

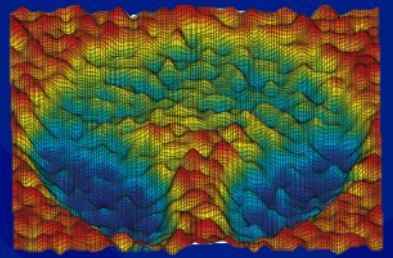
Jan Janssen

07/27/2022

Simulation Code

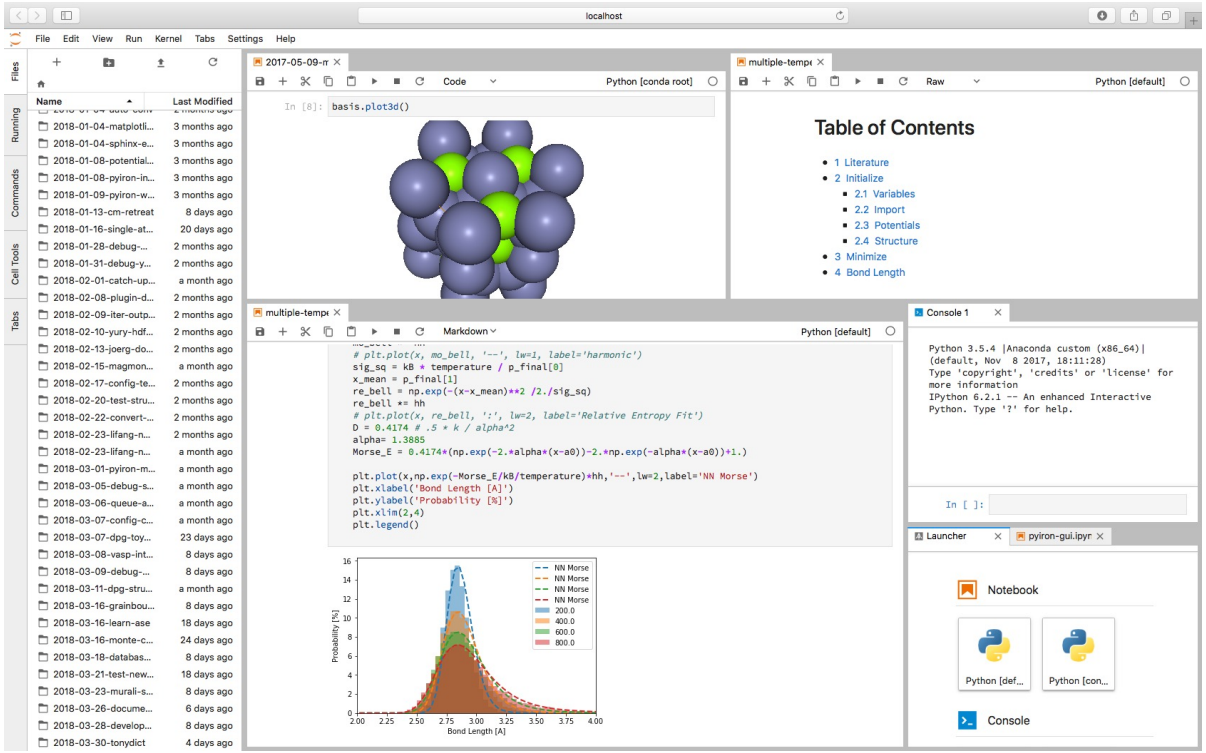


pyiron_base



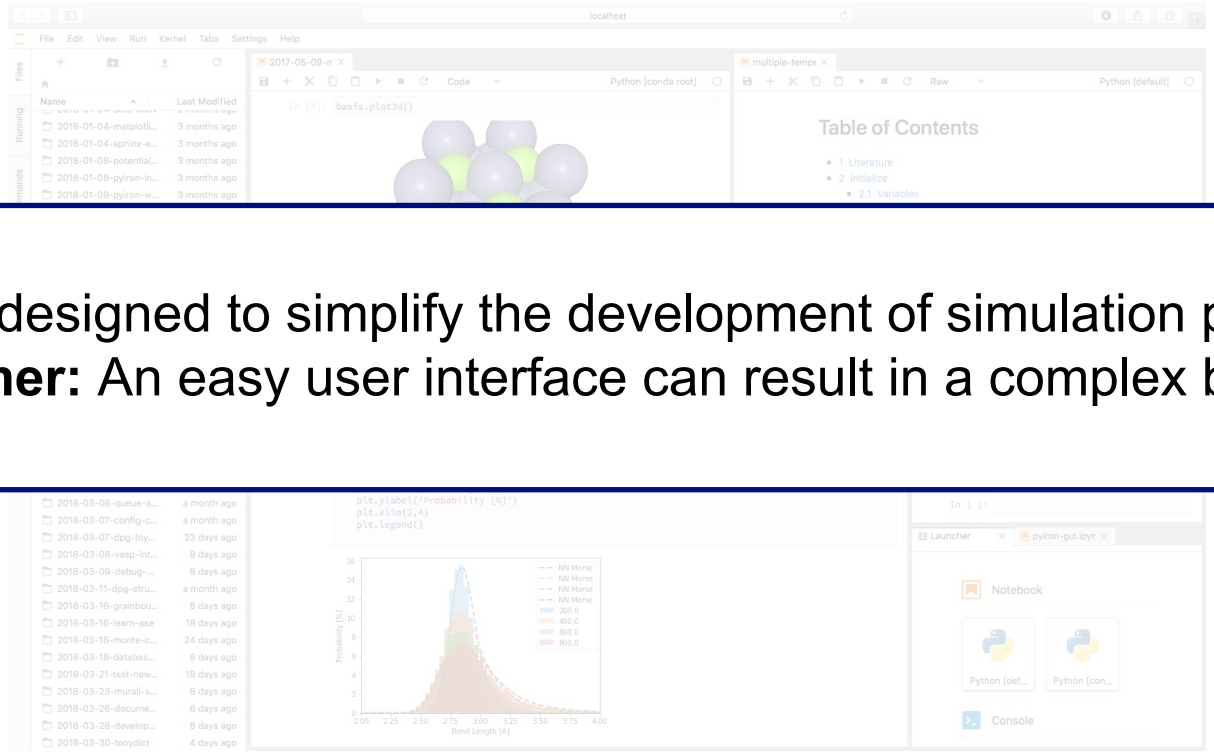
Parameter Study

pyiron: from users to contributors



J. Janssen, et al., Comp. Mat. Sci. 161 (2019) - http://pyiron.org

pyiron: from users to contributors



pyiron is designed to simplify the development of simulation protocols.
Disclaimer: An easy user interface can result in a complex backend.

Implement New Classes

In [1]:
```
from os.path import join
from pyiron_base import TemplateJob, Project
```

In [2]:
```
class ToyJob(TemplateJob):
    def __init__(self, project, job_name):
        super().__init__(project, job_name)
        self.input['input_energy'] = 100
        self.executable = "cat input > output"

    def write_input(self):
        self.input.write_file(
            file_name="input",
            cwd=self.working_directory
        )

    def collect_output(self):
        file = join(self.working_directory, "output")
        with open(file) as f:
            line = f.readlines()[0]
        energy = float(line.split()[1])
        with self.project_hdf5.open("output/generic") as h5out:
            h5out["energy_tot"] = energy
```

Implement New Classes

In [1]:
```
from os.path import join
from pyiron_base import TemplateJob, Project
```

In [2]:
```
class ToyJob(TemplateJob):
    def __init__(self, project, job_name):
        super().__init__(project, job_name)
        self.input['input_energy'] = 100
        self.executable = "cat input > output"

    def write_input(self):
        self.input.write_file(
            file_name="input",
            cwd=self.working_directory
        )

    def collect_output(self):
        file = join(self.working_directory, "output")
        with open(file) as f:
            line = f.readlines()[0]
        energy = float(line.split()[1])
        with self.project_hdf5.open("output/generic") as h5out:
            h5out["energy_tot"] = energy
```

`write_input()`:
- input: python dictionary
- output: write Input files

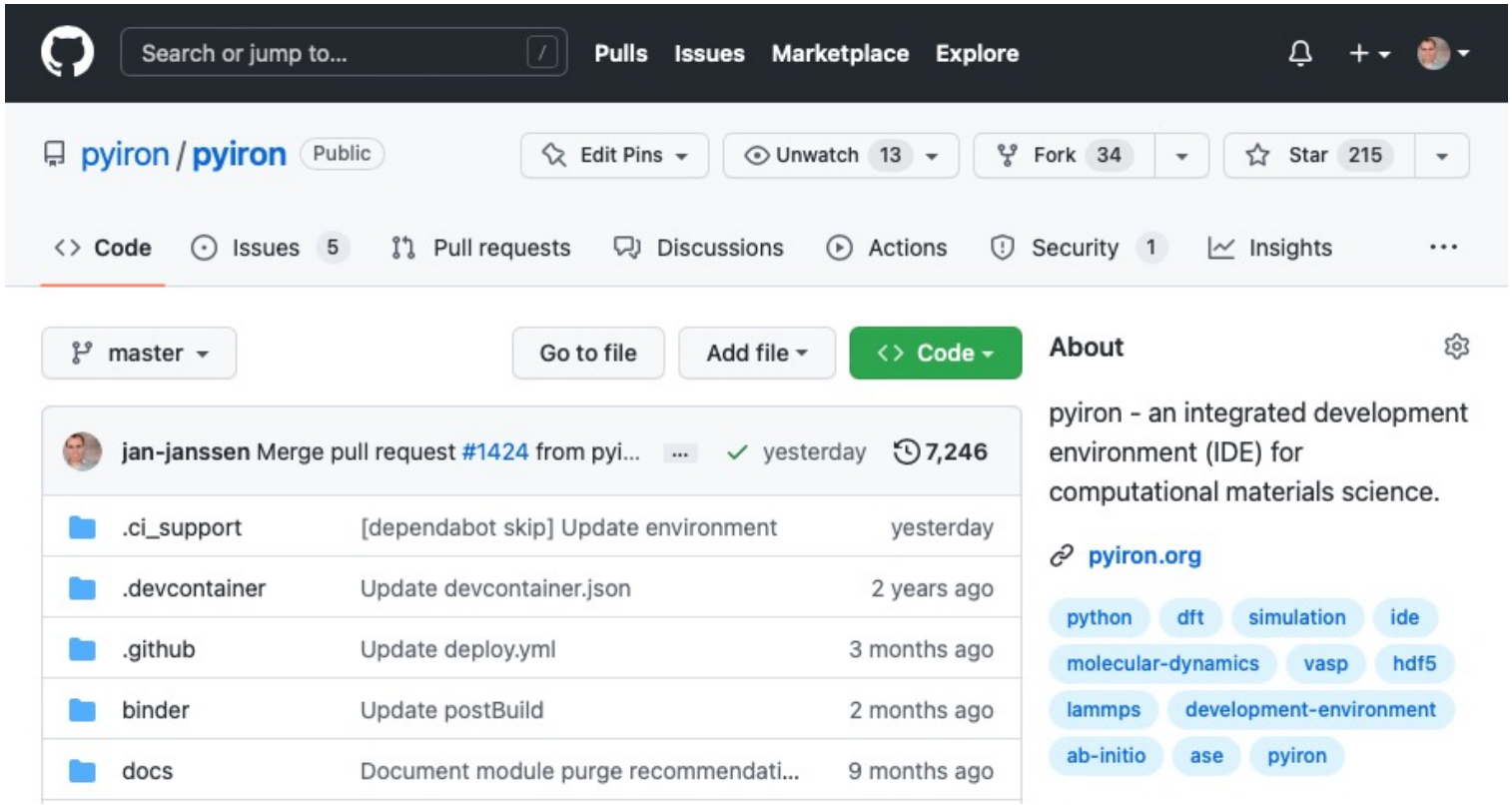
`collect_output()`:
- input: directory of output
- output: python dictionary

Implement New Classes

```
In [1]: from os.path import join
        from pyiron_base import TemplateJob, Project

In [2]: class ToyJob(TemplateJob):
            def __init__(self, project, job_name):
                super().__init__(project, job_name)
                self.input["input_energy"] = 1.0

            def write_input(self):
                file = join(self.working_directory, "input")
                with open(file, "w") as f:
                    f.write("energy " + str(self.input["input_energy"]))

            def collect_output(self):
                file = join(self.working_directory, "output")
                with open(file) as f:
                    line = f.readlines()[0]
                energy = float(line.split()[1])
                with self.project_hdf5.open("output/generic") as h5out:
                    h5out["energy_tot"] = energy
```

In the most basic case only a `write_input()` function and a `collect_output()` function are required to leverage pyiron.

Opensource Development



MAX-PLANCK-INSTITUT FÜR EISENFORSCHUNG

ICAMS
INTERDISCIPLINARY CENTRE FOR ADVANCED MATERIALS SIMULATION

FACHGEBIET MATERIALMODELLIERUNG

Skoltech
Skolkovo Institute of Science and Technology

THE UNIVERSITY OF BRITISH COLUMBIA

CENTER FOR MOLECULAR MODELING

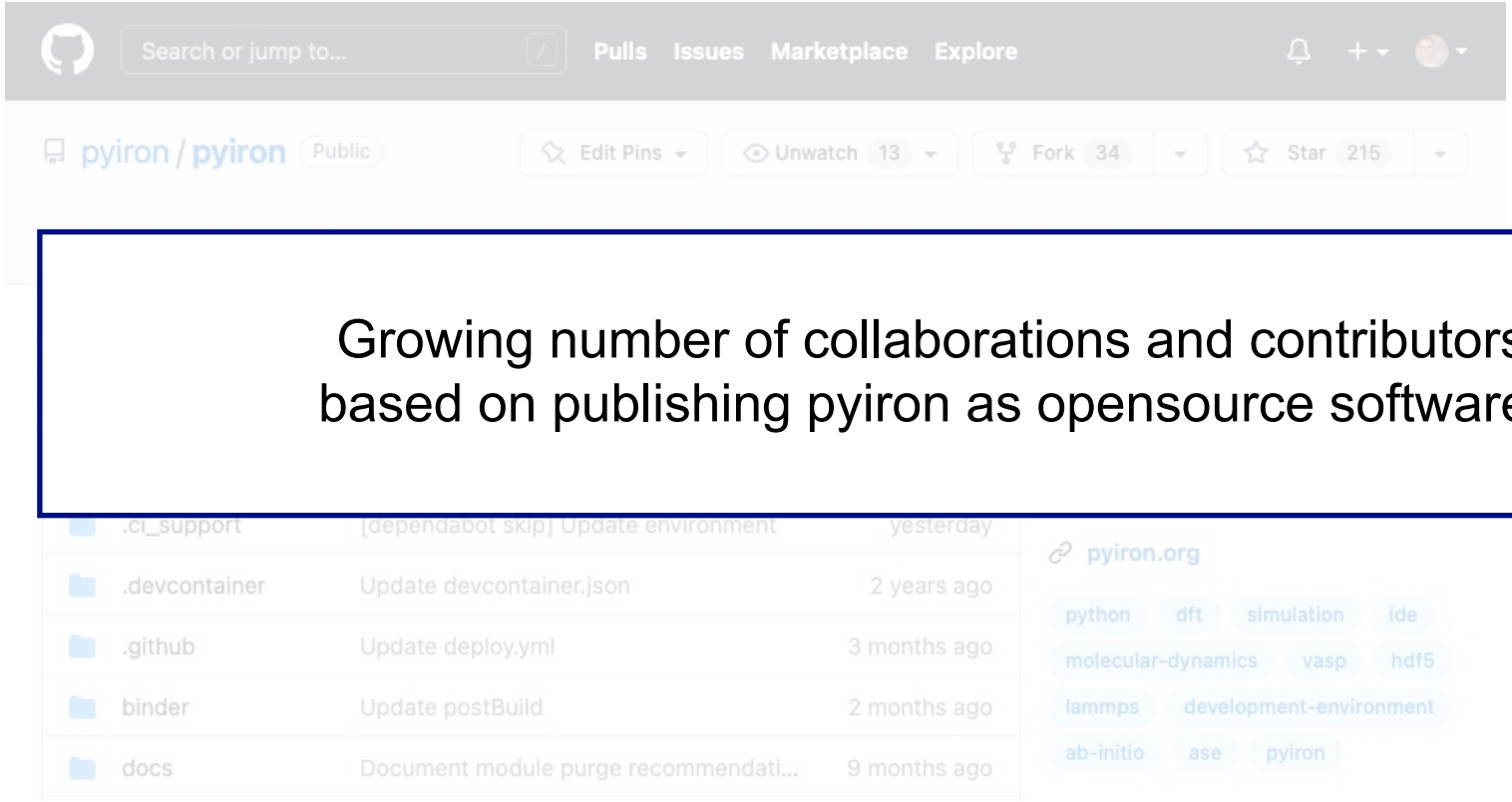
MONTAN UNIVERSITÄT LEOBEN

Los Alamos NATIONAL LABORATORY

MAX-PLANCK-INSTITUT FÜR EISENFORSCHUNG

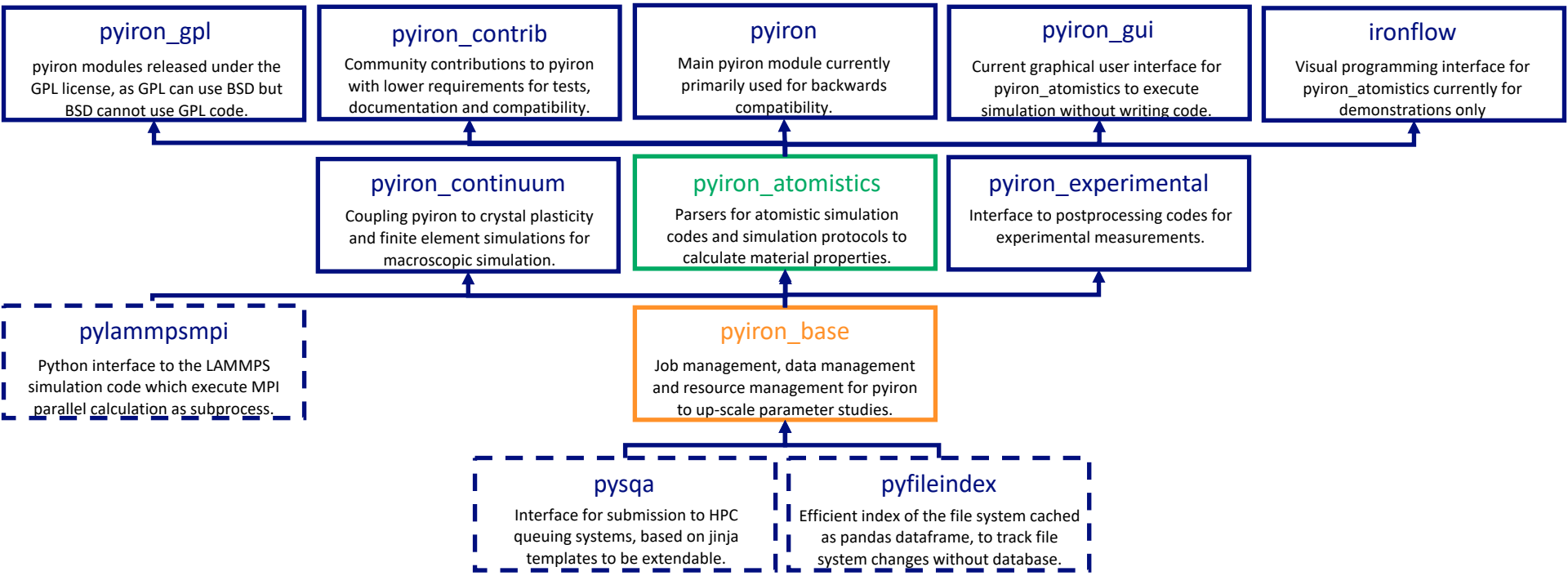
J. Janssen, et al., Comp. Mat. Sci. 161 (2019) - http://pyiron.org

Opensource Development



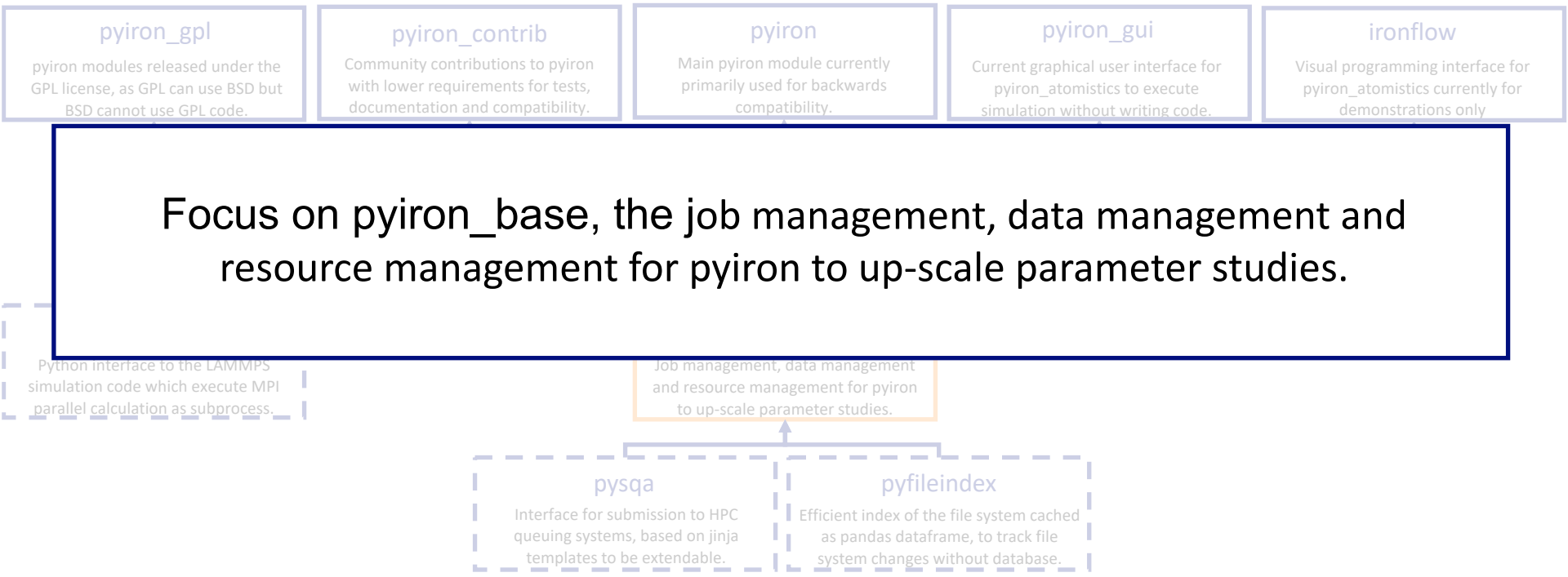
Growing number of collaborations and contributors based on publishing pyiron as opensource software.

pyiron Projects

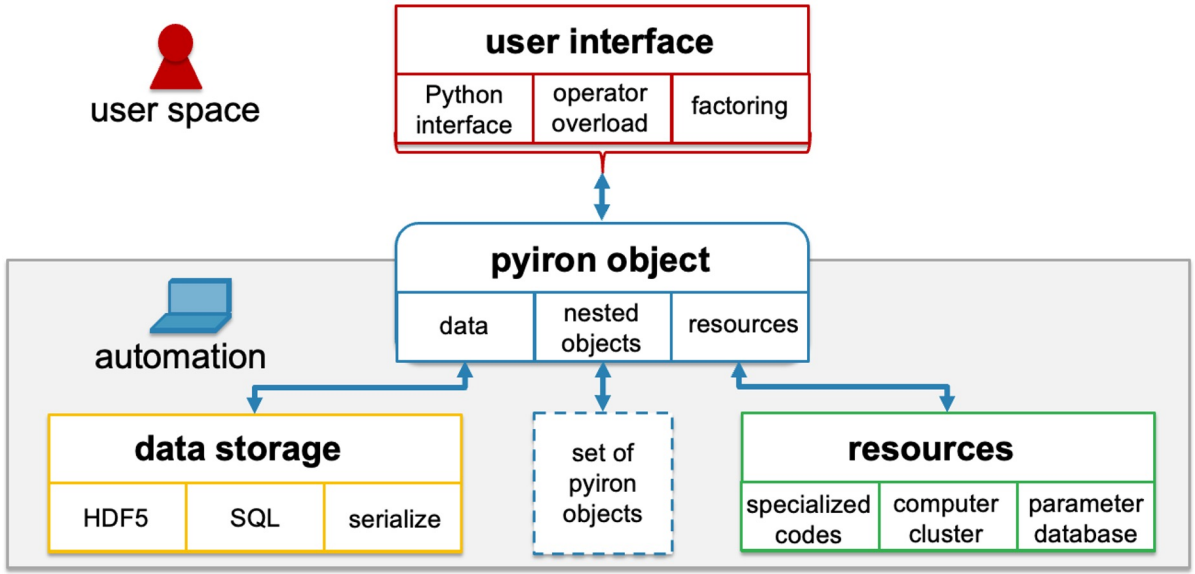


https://github.com/pyiron

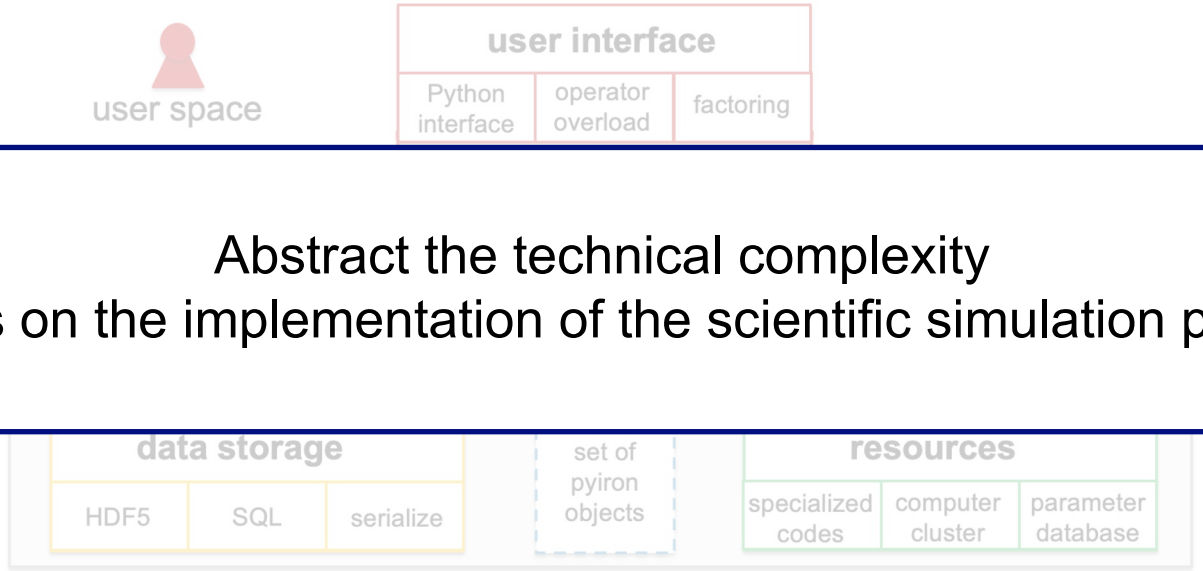
pyiron Projects



pyiron Objects



pyiron Objects



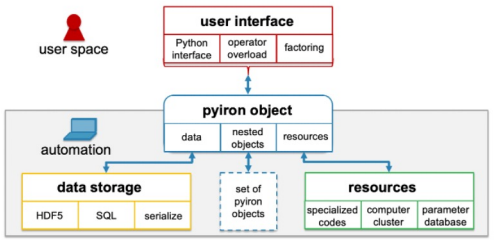
Abstract the technical complexity
to focus on the implementation of the scientific simulation protocol.

Source Code

Summary

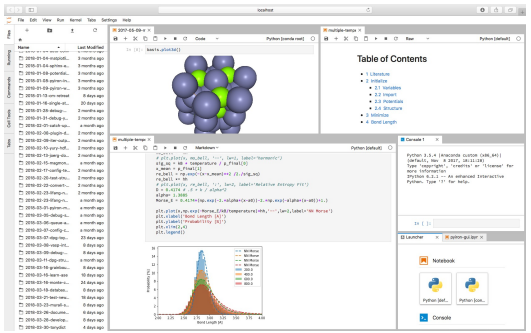
Making the Materials Science Accessible

pyiron objects



Separate the technical and scientific complexity.

pyiron IDE



Develop simulation protocols inside jupyter notebooks.

Application

```
from os.path import join
from pyiron_base import TemplateJob, Project
```

```
class ToyJob(TemplateJob):
    def __init__(self, project, job_name):
        super().__init__(project, job_name)
        self.input['input_energy'] = 100
        self.executable = "cat input > output"

    def write_input(self):
        self.input.write_file(
            file_name="input",
            cwd=self.working_directory
        )

    def collect_output(self):
        file = join(self.working_directory, "output")
        with open(file) as f:
            line = f.readlines()[0]
        energy = float(line.split()[1])
        with self.project_hdf5.open("output/generic") as h5out:
            h5out["energy_tot"] = energy
```

Discover previously unknown relations from parameter studies.

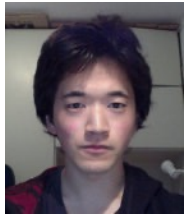
Collaboration

Max Planck Institut für Eisenforschung

Jörg Neugebauer

Tilmann Hickel

Sudarsan Surendralal

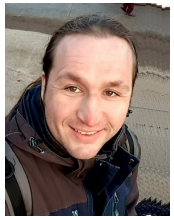


Osamu Waseda

Liam Huber

Marvin Poul

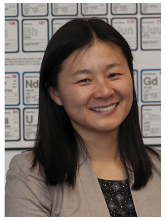
Muhammad Hassani



Niklas Siemer

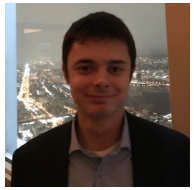
Los Alamos National Laboratory

Danny Perez



Ping Yang

Michael Taylor



Daniel Burrill

Michael Tynes

Collaborators

Blazej Grabowski

Alexander Shapeev

Yury Lysogorskiy

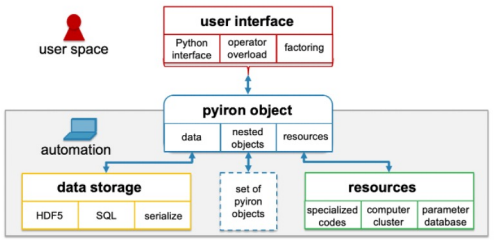
Shoji Ishibashi

…

Summary – Thank you

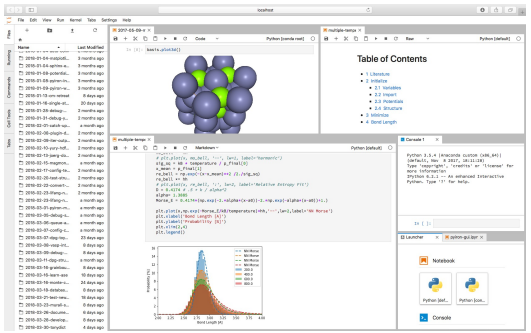
Making the Materials Science Accessible

pyiron objects



Separate the technical and scientific complexity.

pyiron IDE



Develop simulation protocols inside jupyter notebooks.

Application

```
from os.path import join
from pyiron_base import TemplateJob, Project
```

```
class ToyJob(TemplateJob):
    def __init__(self, project, job_name):
        super().__init__(project, job_name)
        self.input['input_energy'] = 100
        self.executable = "cat input > output"

    def write_input(self):
        self.input.write_file(
            file_name="input",
            cwd=self.working_directory
        )

    def collect_output(self):
        file = join(self.working_directory, "output")
        with open(file) as f:
            line = f.readlines()[0]
        energy = float(line.split()[1])
        with self.project_hdf5.open("output/generic") as h5out:
            h5out["energy_tot"] = energy
```

Discover previously unknown relations from parameter studies.