SAND2022-6985C

Sandia National Laboratories

Exceptional service in the national interest

Pyomo

Optimization Beyond Modeling

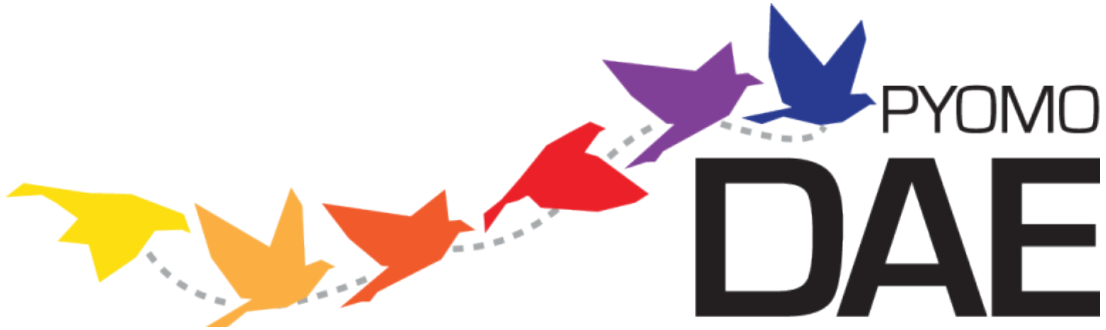
Presenter: Miranda Mundt – mmundt@sandia.gov
Coauthors: Michael Bynum, William Hart, Bethany Nicholson, John Siirola

2022 Tri-lab Advanced Simulation & Computing Sustainable Scientific Software Conference (ASC S^{3}C)

May 24-26, 2022

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Introduction



Pyomo (Python Optimization Modeling) is a Python-based open-source software package that supports a diverse set of optimization capabilities for formulating, solving, and analyzing optimization models.

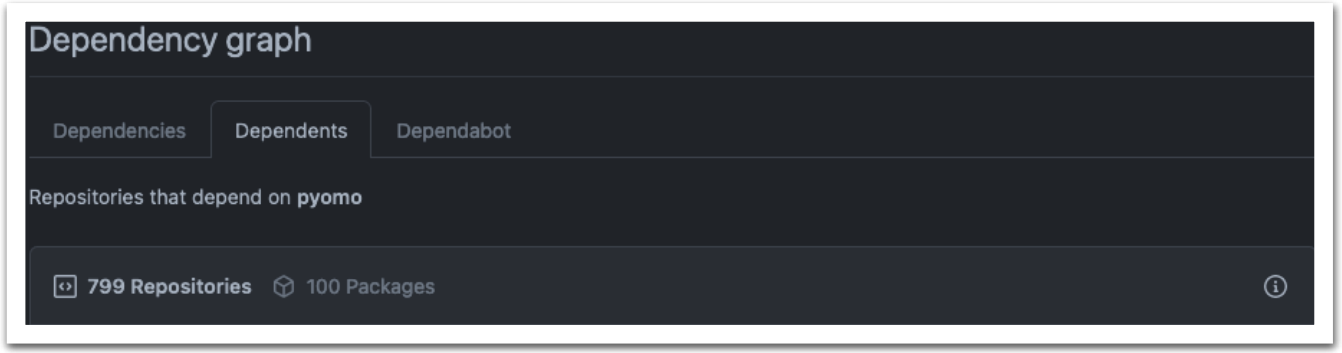
Pyomo supports a wide range of problem types, including:

- Linear programming
- Quadratic programming
- Nonlinear programming
- Mixed-integer linear programming
- Mixed-integer quadratic programming
- Mixed-integer nonlinear programming
- Stochastic programming
- Generalized disjunctive programming
- Differential algebraic equations
- Bilevel programming
- Mathematical programs with equilibrium constraints

Can be found on pypi.org, anaconda.org, and GitHub

History of Pyomo

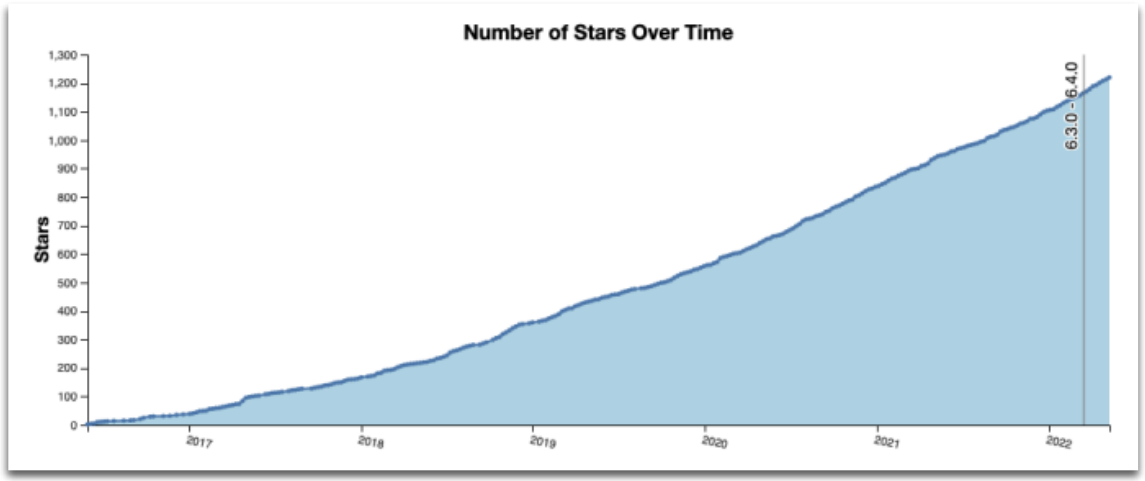
- First released in 2008 as the Coopr software library
- Rebranded as Pyomo around 2011
- Moved to GitHub in mid-2016
- Steady growth in usage and popularity ever since

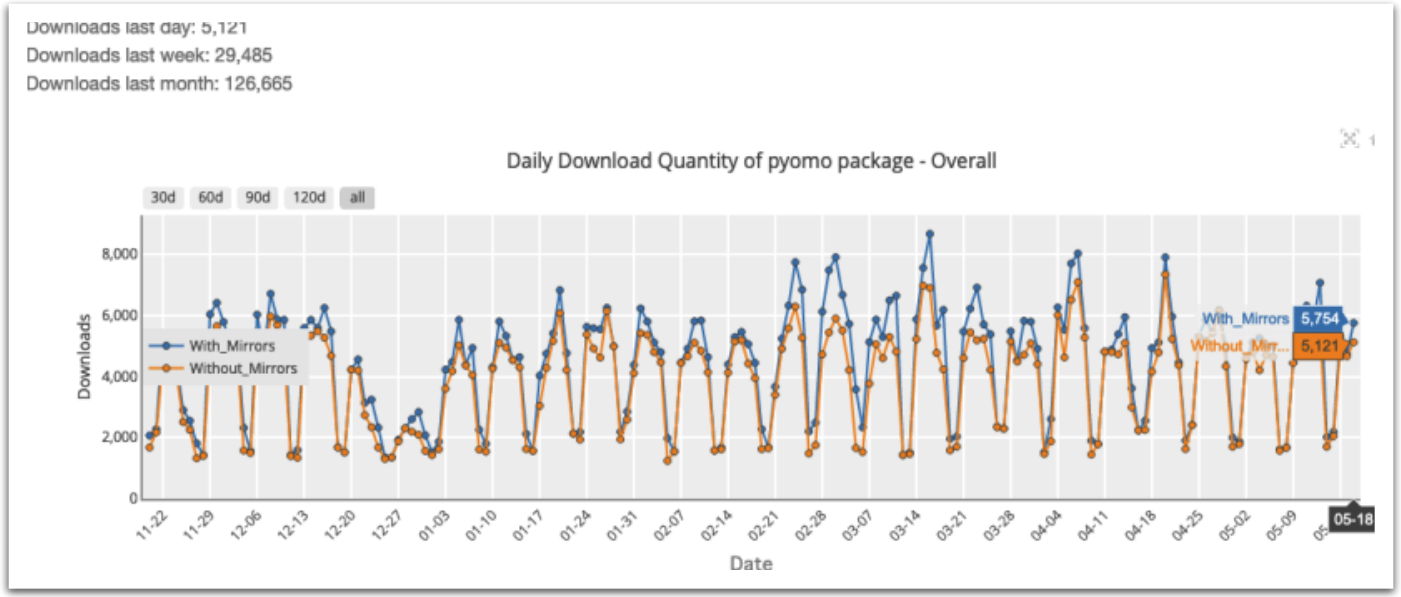


IDAES
Institute for the Design of Advanced Energy Systems

Nevergrad

SCALENE
CPU
MEMORY
GPU





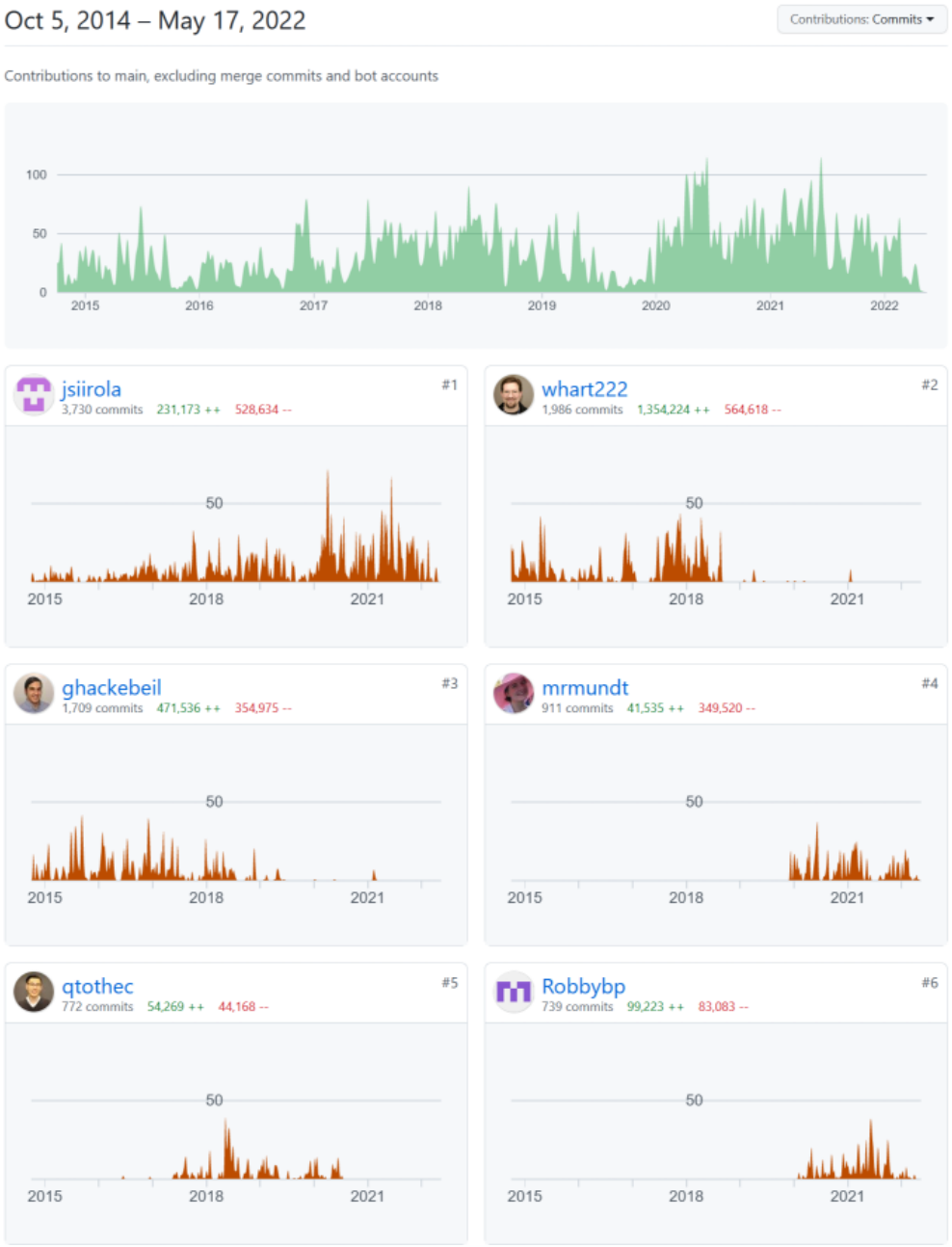
Outline

- Challenges
  - Team composition
  - Software dependencies (upstream and downstream)
  - Automation tools
  - Organization policy requirements
- Resolved Challenges
  - Team composition
  - Downstream dependencies
  - Automation tools
  - Competing priorities – commercial and research interests
- Unresolved Challenges
  - Funding
  - Open-source concerns
  - Upstream Dependency Updates

Challenges

Team Composition

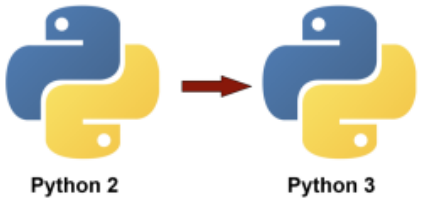
- Core development team (~10 people) spans national labs, industry, and academia
  - 81 total contributors on GitHub
- Team is geographically dispersed over many time zones
  - Dev meetings rarely include more than 5 members of the core dev team
- Development team is constantly changing
  - Few consistent contributors for longer than a few years
- Competing goals and priorities
  - Developers of dependent software projects need stability
  - Classroom users need documentation, examples, and easy installation
  - Research users need fast development and API flexibility



Software Dependencies

Upstream Dependencies

- Dependencies on packages which are not regularly maintained:
  - PyUtilib (a set of python utilities)
  - Nosetests (testing driver)
- Frequently updating optional dependencies causing:
  - Higher technical debt
  - Unreliable testing infrastructure
- Staying updated on new Python versions
  - End-of-life for Python 2
  - API changes in Python 3



Downstream Dependencies

- Institute for the Design of Advanced Energy Systems (IDAES)
  - Relies on Pyomo as part of its core capability
- Other high-traffic or high-impact open source users, interfaces, and research purposes
  - Government stakeholders in DOE/OE, DOE/NA22, NOE/NNSA, DOE/FE, EPA, etc.
  - Optimization solvers (cplex, ipopt, gurobi, etc.)
  - Energy system modelers (IDAES, Calliope)
  - COVID-19 Modelers
  - Research through universities and LDRDs

IDAES
Institute for the Design of Advanced Energy Systems

GUROBI OPTIMIZATION

Automation Tools

From 2016 to 2019, Pyomo simultaneously used three different platforms for automated testing:

- Jenkins
  - Linux (RHEL)
  - Cpython and Pypy (module system)
  - Integration and benchmark testing
    - Licensed optimization solvers
- TravisCI
  - Linux (Ubuntu)
  - Cpython and Pypy (anaconda)
  - Integration testing
- Appveyor
  - Windows
  - Cpython (miniconda)
  - Integration testing

This introduced heavy maintenance debt, and tests for a single Pull Request took an average of 6 hours to fully complete.

Organizational Policy Requirements

- Funding
  - Pyomo has never had a source of direct funding supporting maintenance, CI, user support, documentation, etc.
  - Pyomo was seeded with LDRD funding. Subsequent support has been indirect funding where Pyomo developments supported research capabilities or specific national security applications

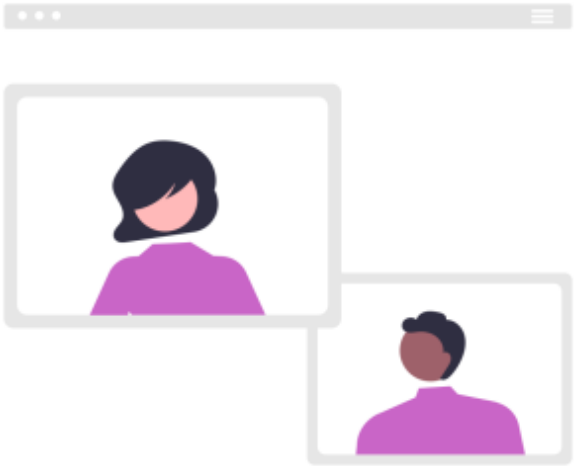
- Open-source at an NNSA national lab
  - In early days of Pyomo, the repository was hosted internally at Sandia making it very challenging to co-develop with external collaborators
  - Pyomo moved to GitHub in 2016 – which helped with these issues
  - New concerns about contributing to or installing open-source software have appeared within the NNSA since 2016 – causing much confusion

National Nuclear Security Administration + open source initiative =

Resolved Challenges

Teaming

- Moved Pyomo development to GitHub in 2016
  - Easier for non-Sandian developers to contribute/access new features
  - Adopted “fork → branch → pull request” workflow for all development
  - Encourage, support, and promote contributions from the community
- Weekly developers call
  - All are invited
  - Designated time for any Pyomo contributor to raise issues/questions with the core dev team
- Established a Pyomo Management Committee (PMC)
  - Developed policies for resolving conflicting development priorities
  - “vote by email” strategy for geographically dispersed team



Separation from Outdated / Not Maintained Dependencies

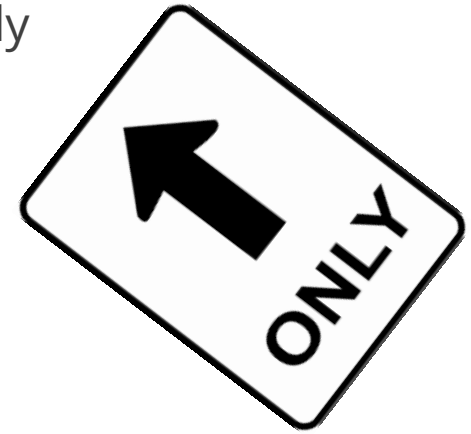
Build / Install Dependencies

Prior to Pyomo 6 series:

- PyUtilib (no active development; maintained by core Pyomo team; many capabilities now exist in Python)
- Enum34
- Ply
- Six

Starting with the Pyomo 6 series:

- Ply



Testing Dependencies

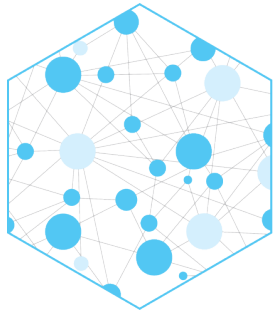
Prior to Pyomo 6.3.0:

- Nosetests (last update: June 2015)
- Specifically designed nosetests plugins for: categories, dynamic test generation
- Fails for Python 3.10 (due to use of removed features)

Starting with Pyomo 6.3.0:

- Pytest (regularly maintained)
- Built-in support for categories and dynamic test generation
- Works for all supported versions of Python

Downstream Software Dependencies



IDAES
Institute for the Design of Advanced Energy Systems

- Close collaboration with IDAES core developers
- Weekly development meetings to align priorities
- Sync release schedules
  - IDAES determines release date
  - Pyomo releases 1-2 weeks prior
  - Release candidate is tested against IDAES before being officially released

Pyomo has interfaces into many open source and commercial solvers

GLPK
ГЛПК

IBM CPLEX

FICO Xpress

GAMS

mosek

We extensively test these interfaces, including collaborating with commercial solvers for access to licenses

Automation Tools

November 2019 → November 2020

Jenkins

GitHub Actions

AppVeyor

Total time: 1-6 hours → 15-60 minutes

Total coverage: ~70% → ~80%

- Jenkins
  - Linux (RHEL)
  - Cpython and Pypy (module system)
  - Integration and benchmark testing
- ~~TravisCI~~ GitHub Actions
  - ~~Linux (Xbuntu)~~ Linux (Ubuntu), Windows, MacOS
  - ~~Cpython and Pypy (apt and anaconda)~~ Cpython and Pypy (pip and anaconda)
  - ~~Integration testing~~ Integration testing for PRs
- ~~Appveyor~~ Automatic wheel creation for distribution
  - ~~Windows~~ Local fork/branch testing
  - Cpython (miniconda)
  - Integration testing

Competing Priorities – Commercial vs. Research

- "fork → branch → pull request" workflow greatly improved code stability **and** development speed
  - Automated testing is done on code contributions BEFORE code is merged into the main branch
  - Every code contribution is reviewed by at least one other developer
  - Codecov check requires automated tests to cover a certain percentage of the lines changed for every pull request
- Pyomo.contrib
  - Designated space for new features/extensions with less rigorous requirements around documentation, testing, and backwards compatibility
- Recognition that developer time is finite
  - Stakeholder funding dictates priority – so some tasks fall by the wayside

Unresolved Challenges

Funding

Successes

- Pyomo funding through IDAES
  - Allows developers to create new functionality and maintain existing capabilities
- LDRD Investments
  - Allow extensions of Pyomo
    - Adversarial optimization (PAO)
    - Optimization and machine learning (OMLT)

Continued Issues

- No direct funding for required activities such as:
  - Performance bottlenecks
  - Modernizations
    - Refactoring / redesigns of antiquated systems
    - Modern-age hardware and infrastructure upgrades
  - Documentation & user support

Open-Source Concerns

Many of the Pyomo developers work within Sandia National Laboratories. As a result, they must consider concerns related to open-source software to balance:

- Risk (security, failure, etc.)
- Technical debt
- Value added

The evolving guidance for NNSA and Sandia open-source packages creates uncertainty about the future of Pyomo and its management, particularly with regards to multi-institutional engagement.

Upstream Dependency Updates

While Pyomo only has one required dependency, there is a large list of optional dependencies which are utilized outside of the “core” Pyomo offering.

Our integration tests frequently break due to constantly changing:

- GitHub Action Images
- Anaconda Environments
- Optional dependency updates

This is both positive and negative. It ensures that we stay current, but causes lots of unintentional churn in our testing system.

Conclusion