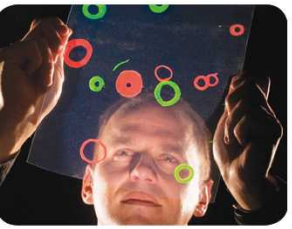


Pyomo 4.0

William E. Hart
Sandia National Laboratories

January 11, 2014

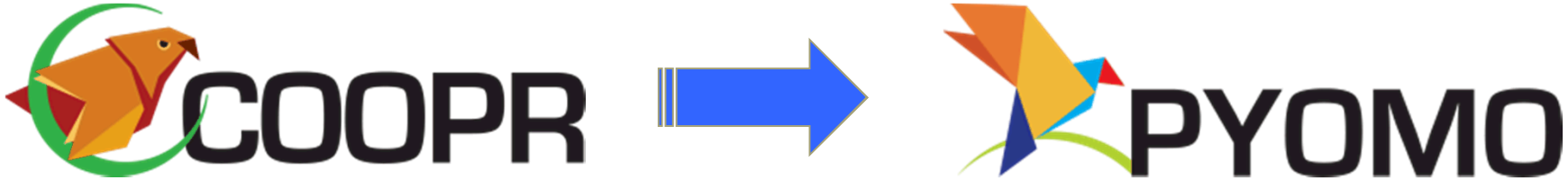
SAND2015-0058C

Sandia National Laboratories

Exceptional service in the national interest

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

What Happened to Coopr?



- Users were installing Coopr but using Pyomo
  - Pyomo modeling extensions were not distinct enough
  - Researchers cited “Coopr/Pyomo”
- Users/Developers were confused by the coopr and pyomo commands
- Developers were coding in Coopr but talking about Pyomo

We need to provide clear branding this project!

Pyomo Overview

Idea: a Pythonic framework for formulating optimization models

- Provide a natural syntax to describe mathematical models
- Formulate large models with a concise syntax
- Separate modeling and data declarations
- Enable data import and export in commonly used formats

Highlights:

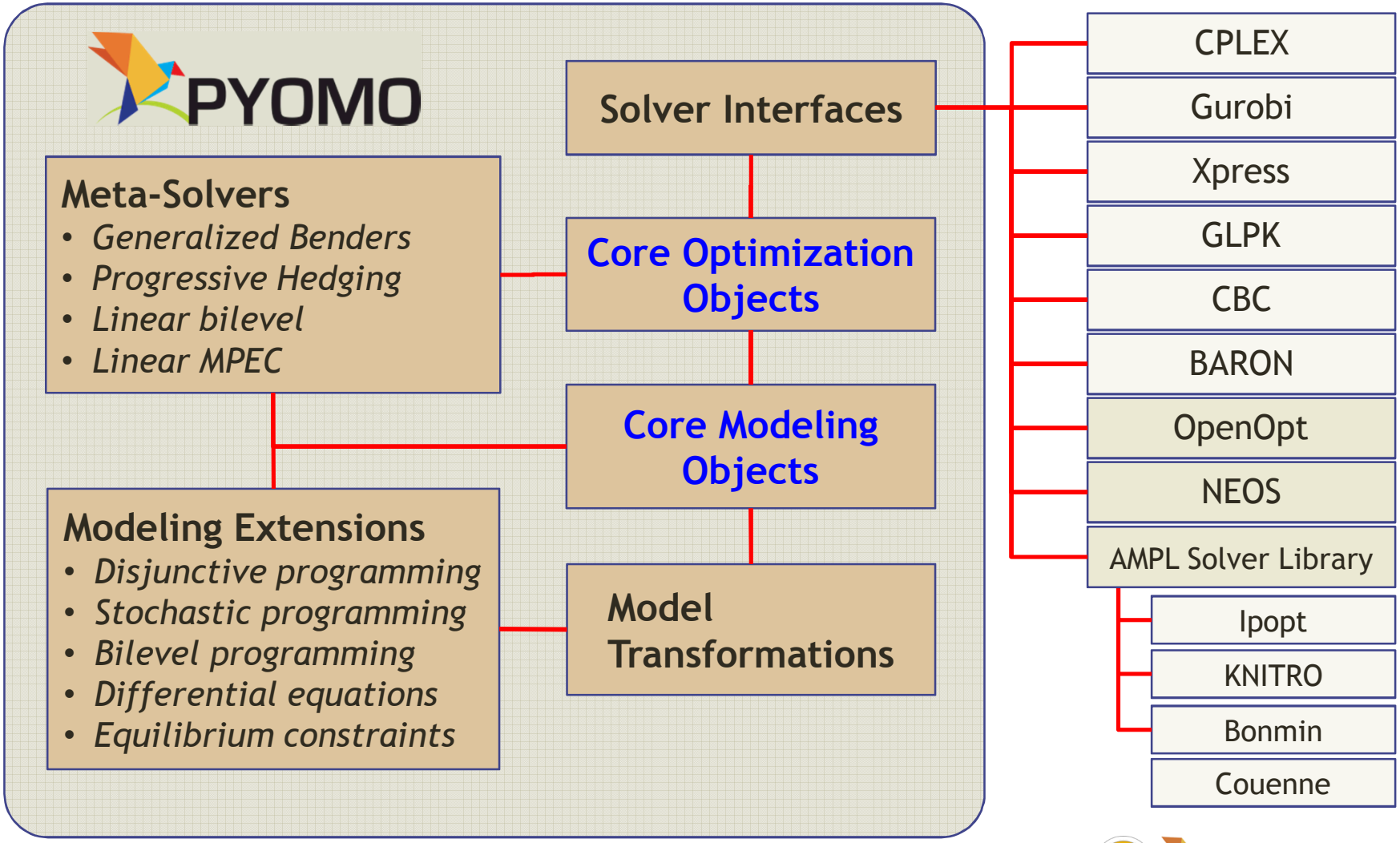
- Python provides a clean, intuitive syntax
- Python scripts provide a flexible context for exploring the structure of Pyomo models

```
# simple.py
from pyomo.environ import *

M = ConcreteModel()
M.x1 = Var()
M.x2 = Var(bounds=(-1,1))
M.x3 = Var(bounds=(1,2))
M.o  = Objective(
       expr=M.x1**2 + (M.x2*M.x3)**4 + \
            M.x1*M.x3 + \
            M.x2*sin(M.x1+M.x3) + M.x2)

model = M
```

Pyomo at a Glance



Who Uses Pyomo?

- Students
  - Rose-Hulman, UC Davis, U Texas, Iowa State, NPS
- Researchers
  - Sandia National Labs, Lawrence Livermore National Lab, Los Alamos National Lab, UC Davis, TAMU, Rose-Hulman, UT, USC, GMU, Iowa State, NCSU, U Washington, NPS, U de Santiago de Chile, U Pisa, Federal Energy Regulatory Agency, …
- Software Projects
  - TEMOA – Energy economy optimization models
  - Minpower – Power systems toolkit
  - Water Security Toolkit – Planning/Response for water contamination
  - SolverStudio – Excel plugin for optimization modeling

What's New in Pyomo 4.0*

- Revisions to the pyomo command
- Modeling enhancements/solvers/transformations
  - Benders decomposition
  - Bilevel solvers
  - MPEC solvers
- Explicit support for model transformations
- Pyomo software engineering
  - Pyomo plugin environment
  - Integrated source tree
  - Robust support for Python 3.x
- A new home page: http://www.pyomo.org

*** A few of these were introduced in Coopr 3.5 and refined in Pyomo 4.0**

Using Pyomo

There are two main ways a user can optimize a Pyomo model:

1. In a script
2. Using the pyomo command

In Pyomo 4.0, the pyomo command integrates a number of separate scripts through subcommands:

- pyomo solve: Optimize a model
  - This replaces the old pyomo script
- pyomo convert: Convert a model into a specified format
  - This replaces scripts like pyomo2nl and pyomo2lp
- pyomo check: Apply syntactic checks on a model file
- pyomo help: Provide help/configuration information

Example: solve subcommand

In previous Coopr releases:

```
pyomo --solver=ipopt model.py
```

In Pyomo 4.0 (*):

```
pyomo solve --solver=ipopt model.py
```

*** For backwards compatibility, the old syntax is supported but not documented.**

New Modeling Capabilities

Bilevel programming

- Modeling components
- Transformations
  - Dualize linear min-max submodel
  - KKT conditions for general submodel

Related Talks (2E)
- W. Hart
  Bilivel programming
- J. Siirola
  Model transformations

Mathematical programming with equilibrium constraints

- Modeling components
- Transformations
  - Convert to standard form
  - Convert to square MCP
  - Nonlinear penalty reformulation
- PATH solver interface

Stochastic Programming

- Generalized Benders solver
- Bounding methods

Model Transformations

Pyomo 4.0 includes a variety of model transformations that can be used to reformulate optimization problems:

- core.linear_dual
- core.nonnegative_vars
- core.radix_linearization
- core.relax_integrality
- bilevel.linear_dual
- dae.collocation_discretization
- dae.finite_difference_discretization
- gdp.bigm
- gdp.bilinear
- gdp.chull
- mpec.simple_disjunction
- mpec.simple_nonlinear
- mpec.standard_form

Example: A MPEC Transformation

A simple MPEC model (S. Dirkse and M. Ferris, 1999):

```
# df.py
from pyomo.environ import *
from pyomo.mpec import *

M    = ConcreteModel()
M.x  = Var(bounds=(-1,2))
M.y  = Var()
M.o  = Objective(expr=(M.x - 1 - M.y)**2)
M.c1 = Constraint(expr=M.x**2 <= 2)
M.c2 = Constraint(expr=(M.x - 1)**2 + (M.y - 1)**2 <= 3)
M.c3 = Complementarity(expr=complements(M.y - M.x**2 + 1 >= 0, M.y >= 0))

model = M
```

The --transform option transforms the model before calling the optimizer:

- pyomo solve --transform=mpec.simple_nonlinear \
  --solver=ipopt df.py

Software Engineering Changes

Direct management of plugins

- Eliminated use of Python plugins in Pyomo
  - These consistently caused code stability issues
- Cleaned up memory usage in Pyomo/PyUtilib plugins
  - Resolved some well-known memory leaks
- Created Pyomo environment package to initialize plugins
  - User must import pyomo.environ

In previous Coopr releases, a model file would include the following import:

```
from coopr.pyomo import *
```

In Pyomo 4.0:

```
from pyomo.environ import *
```

Software Engineering (cont'd)

Pyomo's source is integrated into a single Python package

- Pyomo continues to use subpackages
- Pyomo continues to use Python namespace packages
- The main change is to distribute Pyomo as a single Python package

Impact: Robust/fast Pyomo installations

- Install a scientific Python distribution (www.scipy.org/install.html)
  - *Anaconda, Canopy, Python(x,y), etc.*
- Install with pip, which is packaged with these distributions
  - pip install pyomo
- Optionally install auxilliary packages
  - pip install pyomo.extras
- Install your favorite optimization solvers
  - *This step is not required if you use the NEOS solver interface*

Focusing on Usability

Change	Impact
Rework of plugins environment	• Resolve memory issues • Resolve plugin stability issues
Integrated source tree	• Simplify installation and uninstallation
Subcommands for pyomo	• Simplify user experience • Consolidate help information • Reduce number of command-line scripts
Renaming project	• Clearer branding of this project • Eliminated the coopr command

Some perspective

- These changes are quite disruptive
  - *Existing modeling scripts are inoperable*
- … but core modeling functionality has not changed
  - *These changes require superficial modifications to modeling scripts*
- … and installation of Pyomo should be much more robust
  - *We now rely on a standard Python installation process*

Future Evolution of Pyomo

1. Consolidation of more functionality in the pyomo command
   - Integrate PySP solver scripts (as solvers)
   - Integrate Pyomo utility scripts (e.g. Pyro services)
2. Simplification of the pyomo solve subcommand
   - Idea: express complex optimization solves with a configuration file
     - pyomo solve config.jsn
   - This will simplify the user experience
   - This reflects current user practice (e.g. caching complex pyomo scripts in system shells)
3. Support for other Python implementations
   - PyPy, Jython, IronPython
4. Increased focus on model transformations and meta-solvers
   - Meta-solvers are a natural way to leverage model structure
   - Model transformations will push the development of Pyomo scripting capabilities

For More Information

See the new Pyomo homepage

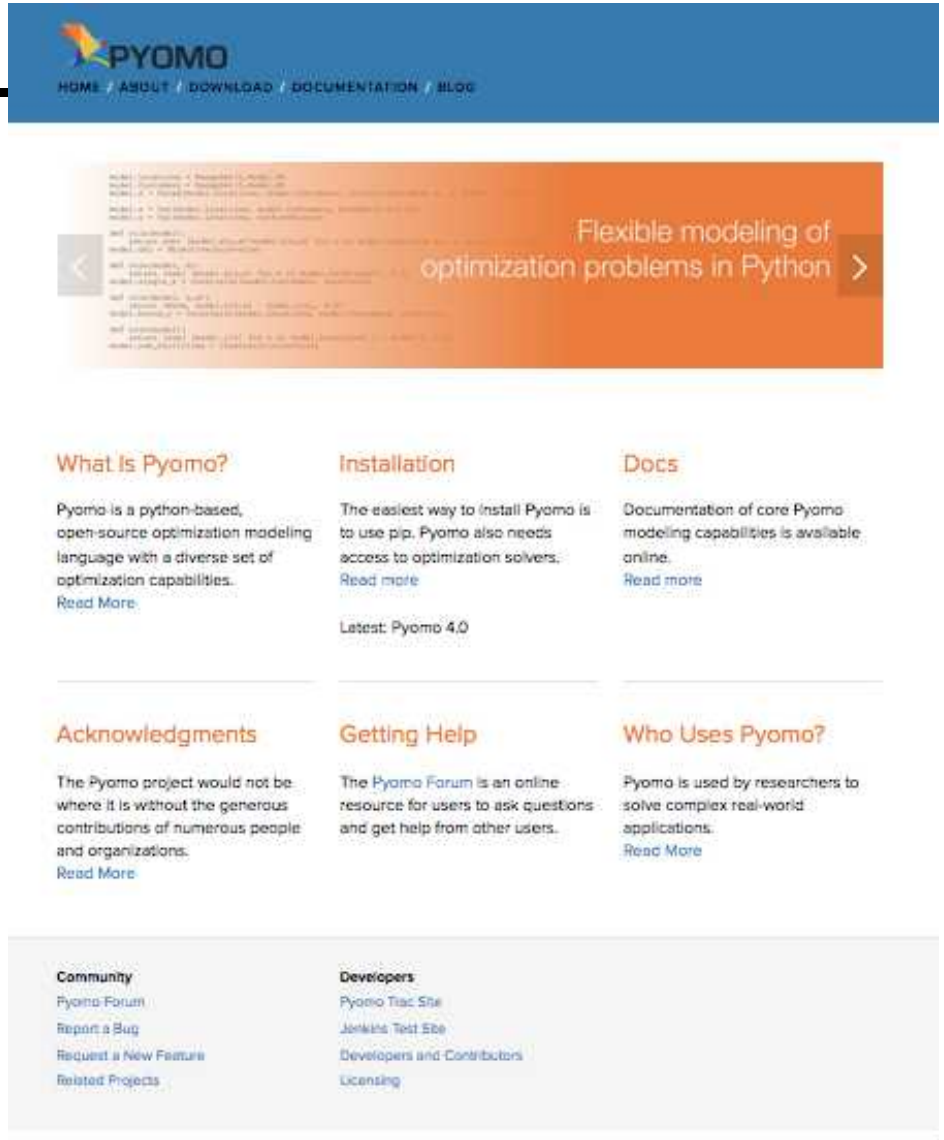
- www.pyomo.org

The Pyomo hompage provides a portal for:

- Online documentation
- Installation instructions
- Help information
- Developer links

Coming soon:

- Blogging about Pyomo capabilities and features
- A gallery of simple examples



Acknowledgements

- Sandia National Laboratories
  - William Hart
  - Jean-Paul Watson
  - John Siirola
  - Francisco Munoz
- University of California, Davis
  - Prof. David L. Woodruff
  - Prof. Roger Wets
- University of Purdue
  - Prof. Carl D. Laird
- Oregon State University
  - Gabe Hackebeil
- Carnegie Mellon University
  - Bethany Nicholson