Recent Developments in EAGO.jl (Easy Advanced Global Optimization in Julia)

Robert Gottlieb, PhD Student
Matthew Wilhelm, PhD
Pengfei Xu, PhD Student
Matthew Stuber, Assistant Professor

November 15th, 2022
Importance of Global Optimization

Physically Meaningful Computations¹

Safety-Critical Systems²

Metastable Point

Equilibrium

Better quality solutions than local methods

Global Optimization

- Nonconvex MINLP formulations naturally arise in many applications
- MINLP solvers generally rely on some variation of spatial branch-and-bound\(^3,4\)
- Relaxed subproblems are used to compute bounds and are often derived from relaxed functions\(^3,4\)

EAGO.jl

Deterministic global optimizer

- High performance solver
- Open-source and free for non-commercial use
- Designed for user-defined functions and routines

Language/Solver Capabilities

- Performance for low-level routines
- Multiple dispatch & contextual programming allow for ready extensibility (e.g., GPU parallelism)
- Ease of setup and distribution

McCormick Relaxations of Factorable Functions

\[ y = f(g(x), \ldots, h(x)) \]

**Auxiliary Variable Method**

\[
\begin{align*}
    z_1 &= g(x) \\
    \vdots \\
    z_n &= h(x) \\
    y &= f(z_1, \ldots, z_n)
\end{align*}
\]

**McCormick-Based Relaxations**

1. Relaxations of \( g \) at \( x \) on \( X \)
2. Apply composite relaxation rules
3. Relaxations of \( f \) at \( (x, z) \) on \( (X, Z) \)
4. Relaxations of \( h \) at \( x \) on \( X \)

Reduced Space Relaxations

Implicit Functions

\[ x(p) \mid h(x(p), p) = 0 \]

ODEs and DAEs

Continental Random Variables

Blackbox Functions

Original EAGO Capabilities

Extendable Support for Script Defined Functions

Parameter Estimation for Nonideal Two-Liquid Mixture

Performant Subroutines

- Nonconvex MINLP & SIP solver
- Bounds Tightening Routines:
  - Optimization-based
  - Feasibility-based
- Preprocessing Routines:
  - Algebraic Rearrangements
  - Subexpression Elimination
  - Regular Problem Classification

\[
\begin{align*}
\min_{p \in \mathcal{P}} & \sum_{i} \left( C_{P}^{mod}(T_i, x_i, p) - C_{P}^{exp}(T_i, x_i) \right) \\
C_{P}^{mod}(T_i, x_i, p) &= -T_i \frac{\partial^2 G}{\partial T^2} \bigg|_p (T_i, x_i, p)
\end{align*}
\]
Additional Features

### Built-in Convexity Detection
(Disciplined Convex Programming\(^{11}\), etc.)

- \(\log(\sqrt{x} + 1)\) (VEX_CONCAVE MONO_NONDECR)
- \(\sqrt{x} + 1\) (VEX_CONCAVE MONO_NONDECR)
- \(\sqrt{x}\) (VEX_CONCAVE MONO_NONDECR)
- \(x\) (VEX_AFFINE MONO_NONDECR)

### Global Dynamic Optimization

\[
\frac{dx}{dt} = \exp(p) \sin(x)(2 - x),
\]
\[
x(0) = 1, \quad p \in [0.01, 1], \quad t \in [0, 5]
\]

**DynamicBounds.jl\(^{12}\)**


### Improved Composite Bilinear Relaxations

- **McCormick Relaxation\(^5\)**
- **Multivariate McCormick Relaxation\(^{13}\)**
- A priori Calculation (Subgradient)
- A priori Calculation (Max-Concave)
Extensibility

Seamless Integration with Defaults

- Any function can be replaced with a user-defined routine
- EAGO automatically incorporates new routines with default operation

Applications:
- Test new bounding methods
- Solve unique problem types
- Modify existing methods
- […]

\[
\min_{x \in X} \sin(x_1)x_2^2 - \cos(x_3)/x_4
\]

In [1]:
```python
using EAGO, IntervalArithmetic
```

In [2]:
```python
struct IntervalExt <: EAGO.ExtensionType end
```

In [3]:
```python
import EAGO; lower_problem!

function lower_problem!(t::IntervalExt, x::EAGO.GlobalOptimizer)
    # Retrieve bounds at current node
    n = x._current_node
    lower = n.lower_variable_bounds
    upper = n.upper_variable_bounds

    # Define X for the node and compute the interval extension
    x_value = Interval.(lower, upper)
    F = sin(x_value[1])*x_value[2]^2-cos(x_value[3])/x_value[4]
    x._lower_objective_value = F.lo
    x._lower_solution = IntervalArithmetic.mid.(x_value)
    x._lower_feasibility = true
    return
end
```
What’s New in EAGO

- Documentation
- Improvements to McCormick.jl
- Parallelization
- Maintenance
Updates in progress:

- Updated Jupyter notebook examples: [https://github.com/PSORLab/EAGO-notebooks](https://github.com/PSORLab/EAGO-notebooks)
- Overhaul of documentation website
  - Quick-start guide
  - Use examples with varying complexity
  - Guide to creating extensions
- New docstrings for all user-facing functions
- Fixes for all incomplete and outdated docstrings
Improvements to McCormick Relaxation Library

New Publication:


- Tighter relaxations of many common ANN activation functions
- Implemented within McCormick.jl
- Substantial speedup compared to naïve relaxations
- Enables global optimization with embedded ANN surrogate models
Parallelization Efforts

Behind-the-scenes development:

- First deterministic global optimization algorithm to be implemented on a GPU
- Packages that extend EAGO to parallelize some aspects of B&B
- Come see us at FOCAPO / CPC 2023!
Maintenance: JuMP and MOI

Current Status

v1.1.1
Nonlinear functionality handled by JuMP

Update in Progress

v1.4.0
Nonlinear functionality handled by MathOptInterface abstraction layer
Conclusions

EAGO — An extensible deterministic global optimizer
- High performance solver
- Designed for user-defined functions and routines
- Open-source and free for non-commercial use

Future Outlook
- Parallel computing capability
- Regular updates to promote ease of use
- Performance improvements in core EAGO algorithms
- Exploring compiled versions for use with GAMS
Acknowledgements

Members of the Process Systems and Operations Research Laboratory at the University of Connecticut (https://psor.uconn.edu/)

Funding:
National Science Foundation, Award No.: 1932723

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.
Questions?

Process Systems and Operations Research Laboratory

https://www.psor.uconn.edu

https://www.github.com/PSORLab/EAGO.jl