UCONN
UNIVERSITY OF CONNECTICUT

# Reduced-Space Optimization of Data-Driven Hybrid Models in EAGO.jl


**Matthew Wilhelm, PhD Candidate**

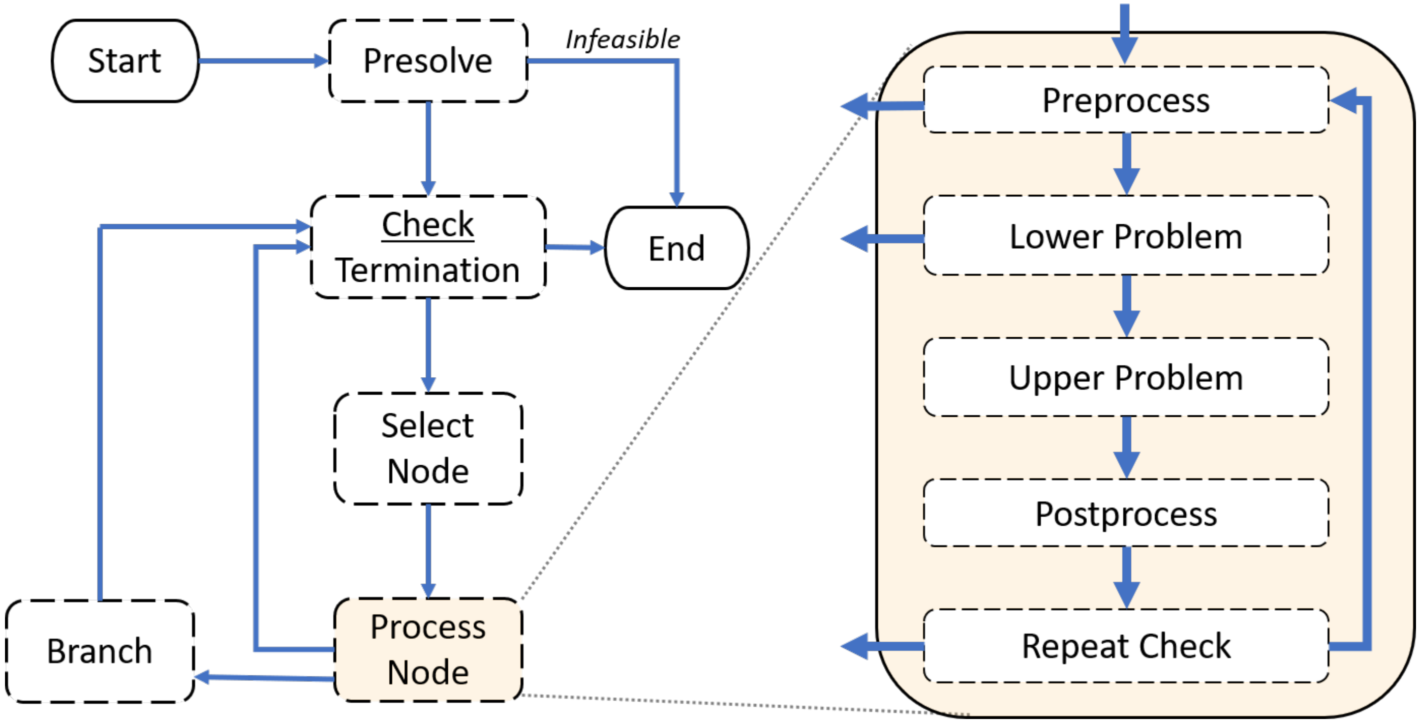
Matthew Stuber, Assistant Professor


informs ANNUAL MEETING
2021 ANAHEIM, CALIFORNIA

Process Systems and Operations Research Laboratory

# EAGO.jl: Global Deterministic Optimization of Simulations

- EAGO.jl: A deterministic global optimizer in JuMP/Julia (for nonconvex MINLP via branch-and-cut)

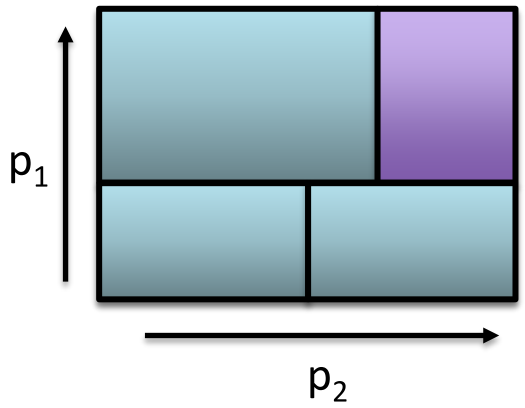



1. Wilhelm, M.E., and Stuber, M.D.. **EAGO.jl: easy advanced global optimization in Julia.** *Optimization Methods and Software*, 1-26.

# EAGO.jl: Global Deterministic Optimization of Simulations

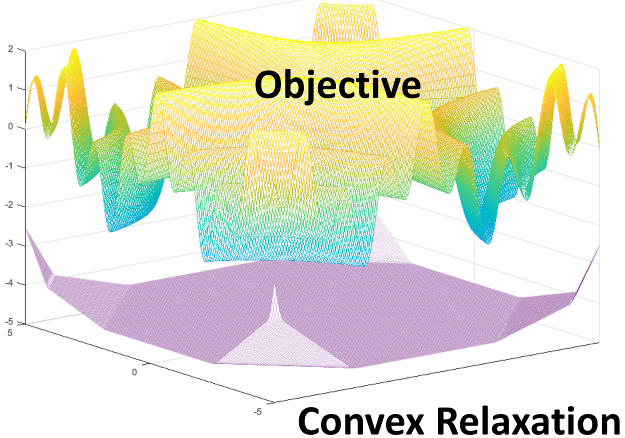
- EAGO.jl: A deterministic global optimizer in JuMP/Julia (for nonconvex MINLP via branch-and-cut)
- Can solve formulations with **user-defined expressions (simulations, etc.)**
- Uses composite relaxation framework which enables expansion to an esoteric set of problems

*(Domain of Active Node)*




Generalized McCormick Operator Arithmetic[2,3]

*(Relaxation of Objective and Constraints)*




2. Mitsos, A, et al. **McCormick-based relaxations of algorithms.** *SIAM Journal on Optimization*, SIAM (2009) 20, 73-601.
3. Scott, JK, et al. **Generalized McCormick relaxations.** *Journal of Global Optimization* 51.4 (2011): 569-606.

# EAGO.jl: McCormick Relaxations

$$\mathbf{y} = \mathbf{f}(\mathbf{g}(\mathbf{x}), \ldots, \mathbf{h}(\mathbf{x}))$$

**Relaxations of g(x) at x in X**

**…**

**Relaxations of h(x) at x in X**

*Apply* **f** *composite relaxation rules*

**Relaxations of f(x) at x in X**

## Akin to automatic differentiation but computes relaxations instead of derivatives

- **EAGO generates relaxations of complicated nonlinear expressions using a McCormick relaxation methodology**[2,3]
- **Intrinsic library of relaxations**
- **Hybrid source-code transformation approach**
  - Method overloading for basic functions.
  - Store and analyze graph structure as well.

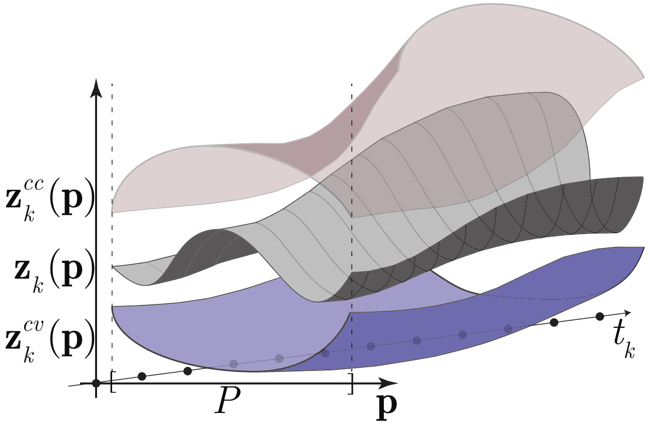
2. Mitsos, A, et al. **McCormick-based relaxations of algorithms.** *SIAM Journal on Optimization*, SIAM (2009) 20, 73-601.
3. Scott, JK, et al. **Generalized McCormick relaxations.** *Journal of Global Optimization* 51.4 (2011): 569-606.

# New Applications

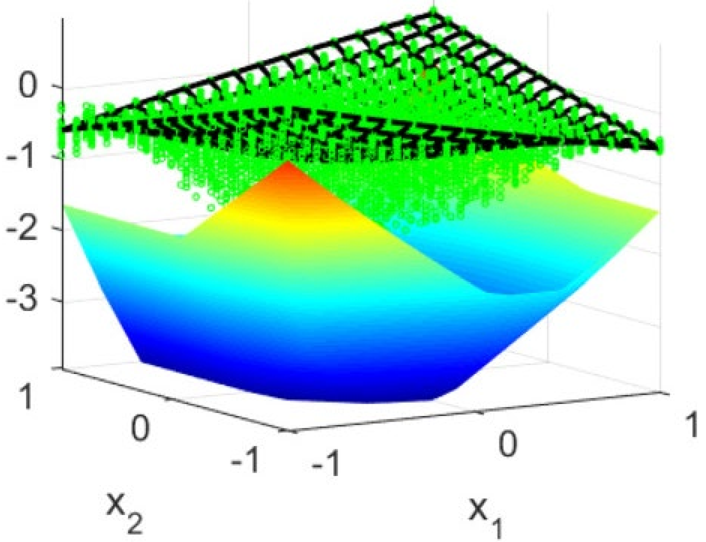
### Implicit Functions[4]

$$\mathbf{x}(\mathbf{p}) \mid \mathbf{h}(\mathbf{x}(\mathbf{p}), \mathbf{p}) = \mathbf{0}$$

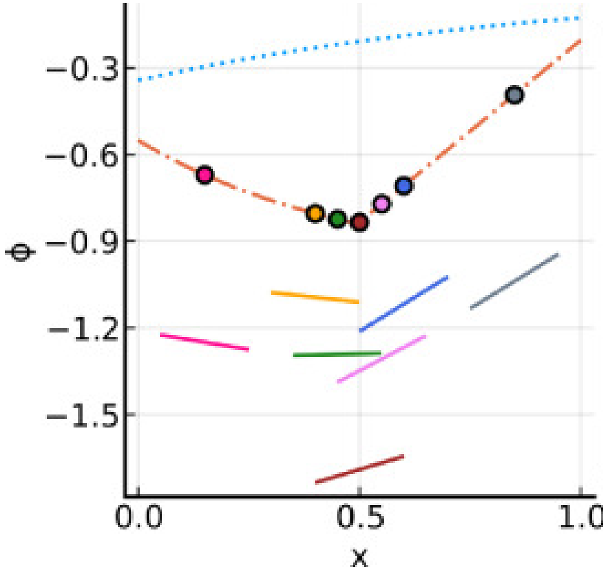
### ODEs and DAEs[5]




### Continuous Random Variables[6]




### Blackbox Functions[7]





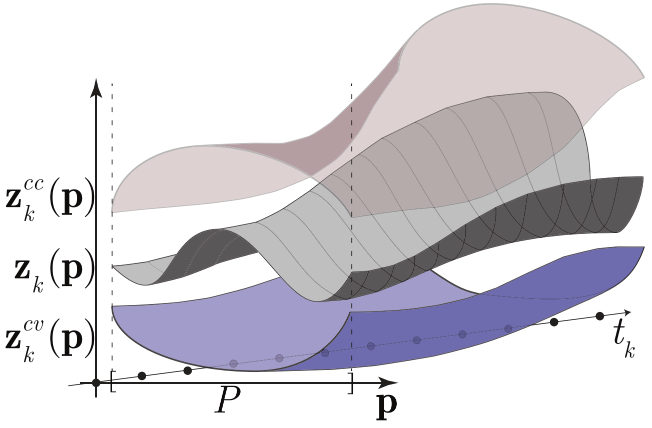
4. Stuber, MD et al. **Convex and concave relaxations of implicit functions.** *Optimization Methods and Software* (2015), 30, 424-460
5. Wilhelm, ME; Le, AV; and Stuber. MD. **Global Optimization of Stiff Dynamical Systems.** *AIChE Journal: Futures Issue,* 65 (12), *2019*
6. Shao, Y and Scott JK. **Convex relaxations for global optimization under uncertainty described by continuous random variables**, *AIChE Journal*, (2018): 3023 – 3033.
7. Song, Y; Cao, H; Mehta, C; and Khan KA. **Bounding Convex Relaxations of Process Models from Below by Tractable Black-Box Sampling**, *Computers & Chemical Engineering*, In Press, (2021).

# New Application s

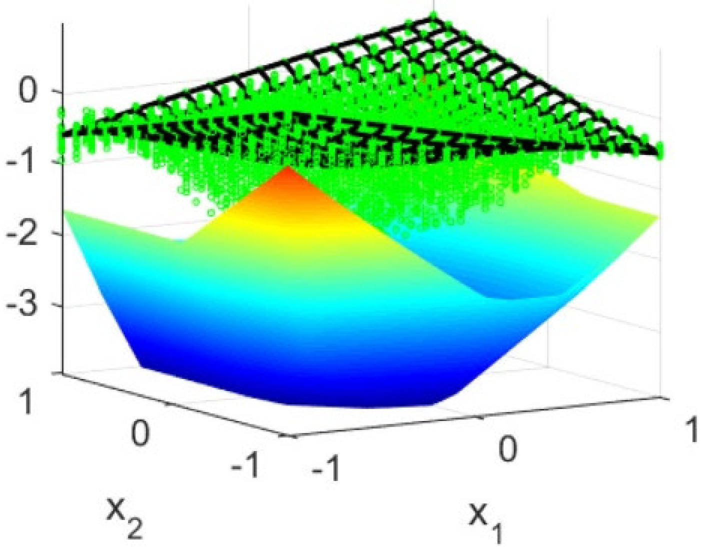
### Implicit Functions[4]

$$\mathbf{x}(\mathbf{p}) \mid \mathbf{h}(\mathbf{x}(\mathbf{p}), \mathbf{p}) = \mathbf{0}$$

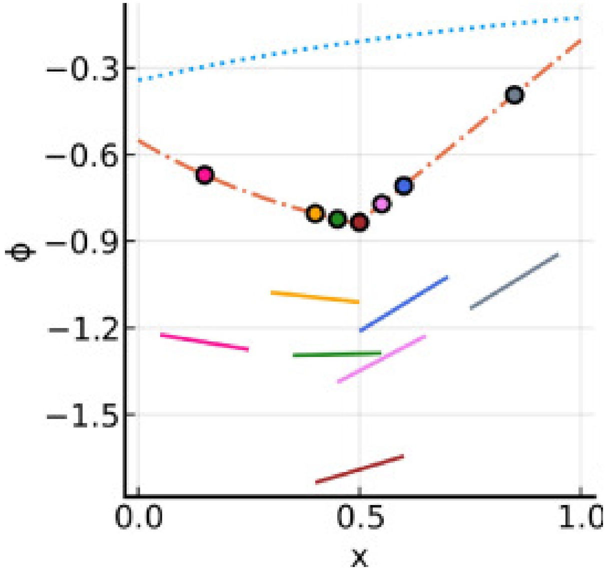
### ODEs and DAEs[5]




### Continuous Random Variables[6]




### Blackbox Functions[7]




4. Stuber, MD et al. **Convex and concave relaxations of implicit functions.** *Optimization Methods and Software* (2015), 30, 424-460
5. Wilhelm, ME; Le, AV; and Stuber. MD. **Global Optimization of Stiff Dynamical Systems.** *AIChE Journal: Futures Issue,* 65 (12), *2019*
6. Shao, Y and Scott JK. **Convex relaxations for global optimization under uncertainty described by continuous random variables**, *AIChE Journal*, (2018): 3023 – 3033.
7. Song, Y; Cao, H; Mehta, C; and Khan KA. **Bounding Convex Relaxations of Process Models from Below by Tractable Black-Box Sampling**, *Computers & Chemical Engineering*, In Press, (2021).

# EAGO.jl: Extended Nonlinear

**New multi-graph representation:**

- Extends JuMP + EAGO backend to introduce auxiliary variables

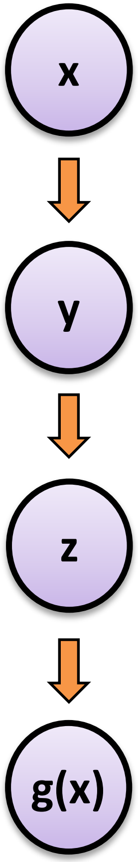
```julia
m = EAGOModel()
@variable(m, -2 <= x[i=1:2] <= 2)
@auxiliary_variable(m, z[i=1:2])
```

- Support for multiple-output subexpressions using new multigraph backend representation.

```julia
function f!(d, y)
    d[1] = y[1]^2 + y[2]
    d[2] = y[2]^2 + y[1]
end
@mimo_expression(m, f!, z, x)
@constraint(m, sum(z) <= 0.0)
```

**Composability of Nonlinear Functions (via JuMP extension):**

- Facilities chaining blocks of general nonlinear expressions (such as implicitly defined functions).



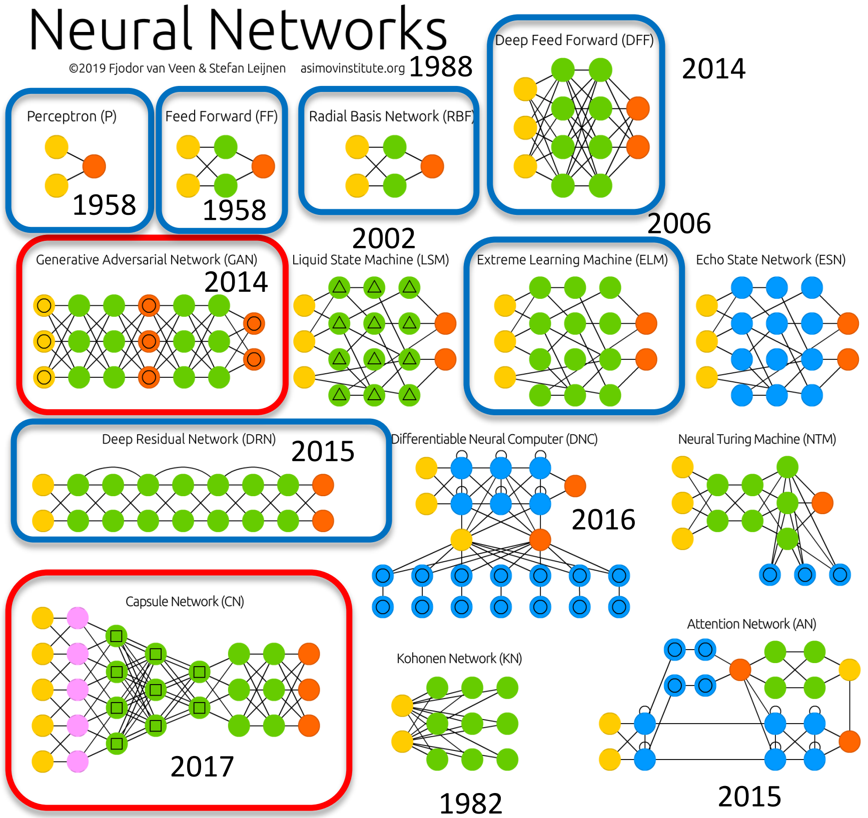

$$\mathbf{h}_1(\mathbf{y},\mathbf{x}) = \mathbf{0}$$
$$\mathbf{y} = \mathbf{y}(\mathbf{x})$$
$$\mathbf{h}_2(\mathbf{z},\mathbf{y}) = \mathbf{0}$$
$$\mathbf{z} = \mathbf{z}(\mathbf{y})$$
$$\mathbf{g}(\mathbf{x}) = \mathbf{f}(\mathbf{z}(\mathbf{y}(\mathbf{x})))$$

# EAGO.jl: Embedded ML

- Currently, EAGO.jl supports ML-models that are embedded with have a factorable representation.
- Additional work underway to natively support embedded Flux.jl models.
- Future support for more complex ML model structures
  - Layers with implicit functions evaluated via fixed-point methods (i.e. deep equilibrium networks).

$$\hat{y}\,(u) = Cx + Du$$
$$x = \phi(Ax + Bu)$$


8. L.E. Ghaoui et al. **Implicit Deep Learning.** https://arxiv.org/pdf/1908.06315.pdf.
9. S. Bai, J.Z. Kolter, and V. Koltun. **Deep Equilibrium Models.** https://arxiv.org/abs/1909.01377.pdf


*Adapted from https://www.asimovinstitute.org/neural-network-zoo/*

# Simple Surrogate Model

***Peaks Function***

$$f(x,y) = 3(1-x)^2 e^{-x^2-(y+1)^2} - 10\left(\frac{x}{5} - x^3 - y^5\right) e^{-x^2-y^2} - \frac{e^{-(x+1)^2-y^2}}{3}$$

```
function f(x,y)
    3*exp(-x^2-(y+1)^2)*(1-x)^2 - 10*(x/5-x^3-y^5)*exp(-x^2-y^2) - exp(-(x+1)^2-y^2)/3
end
```

***Domain***

$$-3 \leq x, y \leq 3$$

```
l = [-3; -3]
u = [3; 3]
```

# Simple Surrogate Model

**Surrogates.jl**
Workflow for Model Development

1. Generate Data

```
xd = sample(n, l, u, SobolSample())
yd = f.(xd)
```

# Simple Surrogate Model

**Surrogates.jl**
Workflow for Model Development

1. Generate Data

```
xd = sample(n, l, u, SobolSample())
yd = f.(xd)
```

2. Specify Model

```
m = Chain(Dense(2,6,tanh), Dense(6,1))
```

# Simple Surrogate Model

**Surrogates.jl**
Workflow for Model Development

1. Generate Data

```
xd = sample(n, l, u, SobolSample())
yd = f.(xd)
```

2. Specify Model

```
m = Chain(Dense(2,6,tanh), Dense(6,1))
```

3. Train Model

```
ns = NeuralSurrogate(xd, yd, l, u, model = m)
surrogate_optimize(f, SRBF(), l, u, ns, SobolSample())
```

# Optimize Surrogate Model

**JuMP + EAGO.jl**
Workflow for Optimization

4. Create Model

```julia
using JuMP, EAGO

m = Model(EAGO.Optimizer)
@variable(m, l[i] <= x[i=1:2] <= u[i])
```

5. Register Function, Specify Problem

```julia
@register(m, :f, 2, f)
@NLobjective(m, Min, f(x[1], x[2]))
```

6. Optimize the problem

```julia
optimize!(m)
```

# Optimize Surrogate Model

- ❑ Dispatch to improved relaxations for a large library of activation functions automatically
- ❑ Some support for "cleaning" script defined models  for compatibility.
- ❑ Compatible with standard global optimization methods

4. Create Model

```
using JuMP, EAGO

m = Model(EAGO.Optimizer)
@variable(m, l[i] <= x[i=1:2] <= u[i])
```

5. Register Function, Specify Problem

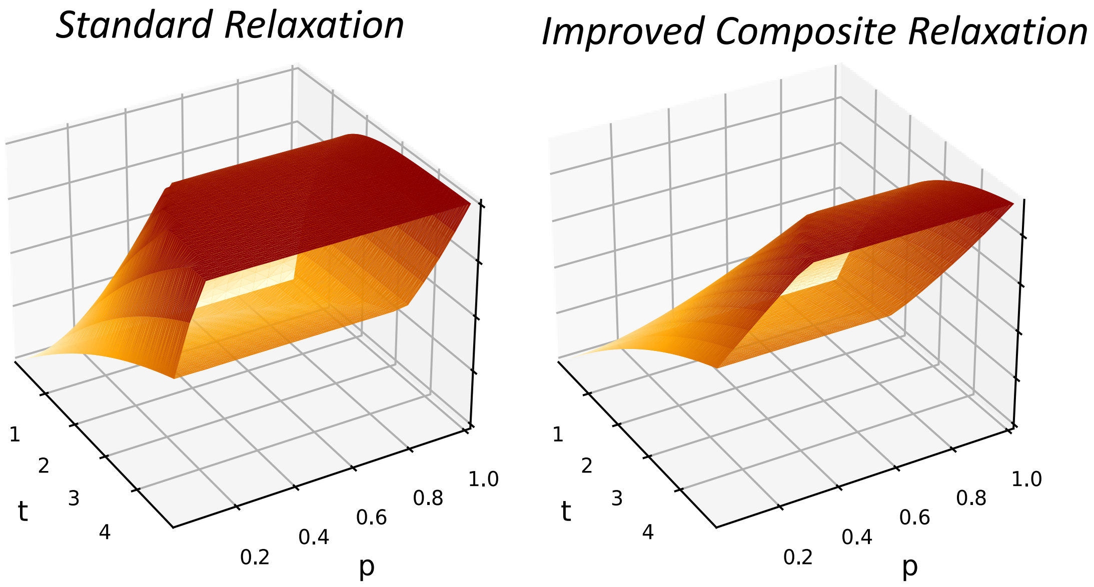
```
@register(m, :f, 2, f)
@NLobjective(m, Min, f(x[1], x[2]))
```

6. Optimize the problem

```
optimize!(m)
```

# Next Steps

- **Support for neural-ODE models.**
- **Improvements to composite relaxation forms (faster global optimization of surrogate models).**
- **Continually add support for additional ML forms.**
    - Relaxation of implicit functions[10]
    - Reverse propagation of relaxations[11]




10. Stuber, MD et al. **Convex and concave relaxations of implicit functions.** *Optimization Methods and Software* (2015), 30, 424-460
11. Wechsung, Achim, et al. **Reverse propagation of McCormick relaxations.** *Journal of Global Optimization* 63.1 (2015): 1-36.

# Acknowledgements

- PSOR Lab @ UCONN
- INFORMS 2021 Organizers
- Funding: National Science Foundation

This material is based upon work supported by the National Science Foundation under Grant 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.

informs ANNUAL MEETING
2021 ANAHEIM, CALIFORNIA

Process Systems and Operations Research Laboratory

UCONN
UNIVERSITY OF CONNECTICUT

# Questions?

Process Systems and Operations Research Laboratory

https://www.psor.uconn.edu

EAGO

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