MITSUBISHI ELECTRIC
Changes for the Better

for a greener tomorrow

MITSUBISHI ELECTRIC RESEARCH LABORATORIES
Cambridge, Massachusetts

Equation-Oriented Languages for Multiphysical Systems

Chris Laughman

18.337

3 October 2018

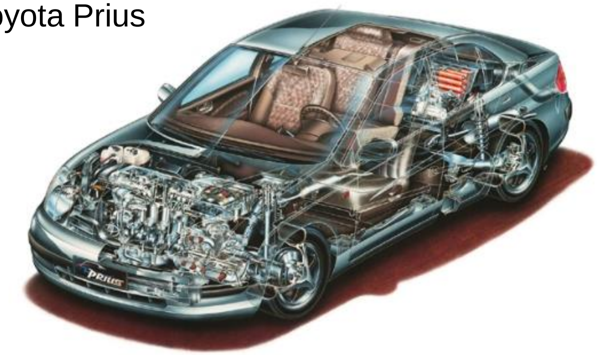
Engineered Systems Are Becoming More Complex

Deutsche Post building

EUI < 100 kWh/m2

U.S. average > 600 kWh/m2

Toyota Prius



We want more out of our engineered systems

- Cheaper, better performance / more efficient, faster time-to-product

Subsystems are also becoming more complex

- More interaction between subsystems at system level
- Increased specialization needed for subsystem engineering

How do we efficiently manage this complexity?

Opportunities

- New tools for large-scale unstructured system science
- The representation used in equation-oriented languages imposes important constraints on the system structure: there are open challenges to adapt and develop mathematical tools in this context
- Leveraging Julia's metaprogramming and multiple dispatch could enable a rich set of potential applications
  - Distributed equation-oriented simulation (HPC)
  - Variable-index DAE systems
  - Multi-rate and sparse DAE solvers and approaches to improve simulation time
  - Leveraging locality in DAE models to improve simulation time and MOR
  - Large-scale estimation, dynamic model reduction, and uncertainty quantification
  - Large-scale EO system optimization (adjoint optimization, PDE-constrained optimization)
- An equation-oriented Julia-based DSL facilitates the application of advances in scientific computing to MBSE

Modelica: A Language for Multiphysical Modeling

What is it?

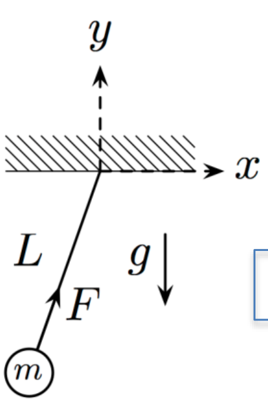
- Equation-based, object-oriented, open standard multiphysical modeling language
- Acausal equation-based model descriptions are compiled to generate causal simulation code
- Industrial use at Toyota, BMW, and United Technologies since mid-1990s

What are its benefits?

- Object-oriented construction facilitates model development (reuse, documentation)
- Many existing model libraries reduce development time
- Acausal declarative language expresses the computation without specifying control flow
- Simulate systems of hybrid differential algebraic equations with >100,000 variables
- Can be interfaced to other programming environments (Matlab, Python, others)

Typical Workflow

Physical System



Mathematical Description

$$m\dot{v}_x = -\left(\frac{x}{L}\right)F$$

$$m\dot{v}_y = -\left(\frac{y}{L}\right)F - mg$$

$$\dot{x} = v_x$$

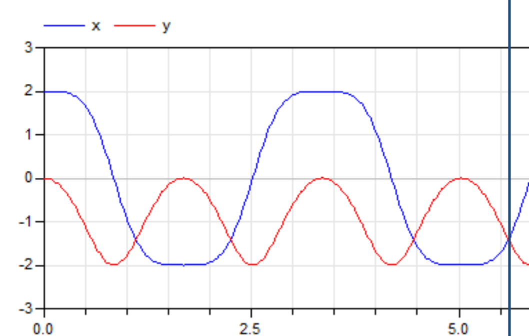
$$\dot{y} = v_y$$

$$\boxed{x^2 + y^2 = L^2}$$

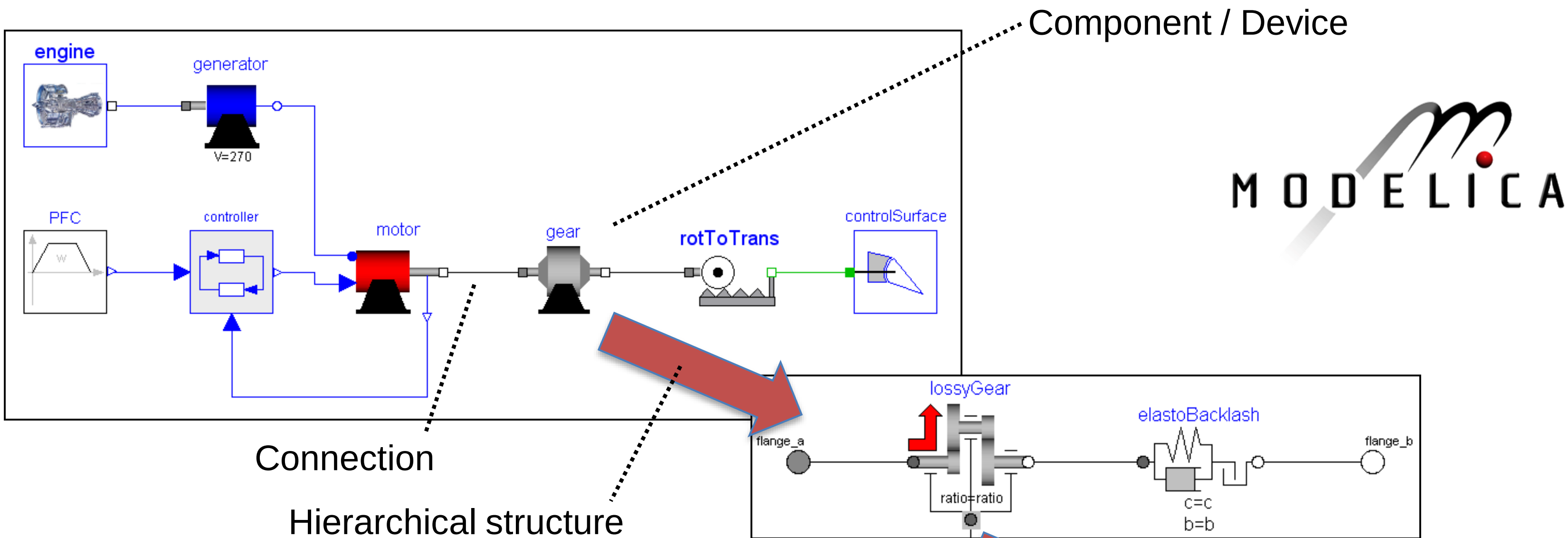
Modelica Model

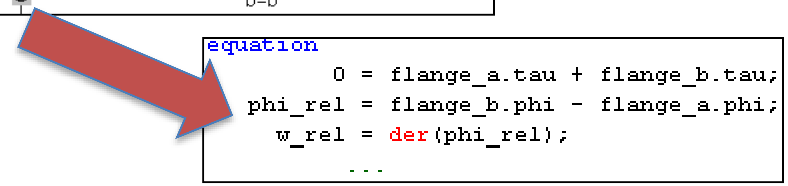
```
model PendulumCartesian
  constant Real PI = 3.14159;
  parameter Real m=1, g=9.81, L=2;
  Real F;
  output Real x(start=2), y(start=0);
  output Real vx, vy;
equation
  m*der(vx) = -(x/L)*F;
  m*der(vy) = -(y/L)*F - m*g;
  der(x) = vx;
  der(y) = vy;
  x^2 + y^2 = L^2;
end PendulumCartesian;
```

Simulation Result



Object-Orientation: User View of Modelica



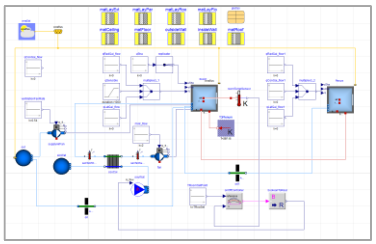


- Each icon represents a physical component
- A connection line represents physical coupling (heat flow, electrical power, fluid flow)
- A component consists of connected subcomponents (using hierarchical structure) and/or equations
- Models of large complex systems can be built up out of mathematical descriptions of simple subsystems

Modelica: Uses Beyond Simulation

- Equation-based system representation supports a variety of uses
- Model-order reduction, control design, optimization, and others

Modelica



```
Buildings.Rooms.MixedAir Room1(
  redeclare package Medium = MediumA,
  AFlo=414.1825,
  hRoo=2.6,
  nConExt=1,
  datConExt(layers={outsideWall},
       A={12.445*2.6},
       til={Buildings.Types.Tilt.Wall},
       azi={Buildings.Types.Azimuth.W}),
  nConExtWin=2,
  datConExtWin(
          layers={outsideWall, outsideWall},
          A={28.0*2.6, 4.0*2.6},
          glaSys={singlePaneGlass, singlePaneGlass},
          hWin={1.6, 1.6},
          wWin={28.0-3.69, 3.515},
          ove(wR={0,0},wL={0,0}, gap={0.1,0.1}, dep={1,1}),
          each fFra=0.1,
          each til=Buildings.Types.Tilt.Wall,
          azi={Buildings.Types.Azimuth.S, Buildings.Types.Azimuth.N}),
  nConPar=1,
  datConPar(layers={insideWall}, each A=100,
       each til=Buildings.Types.Tilt.Wall),
  nConBou=1,
  datConBou(layers={matFloor}, each A=414.1825,
       each til=Buildings.Types.Tilt.Floor),
  nSurBou=1,
  surBou(each A=414.1825,
       each absIR=0.9,
       each absSol=0.9,
       each til=Buildings.Types.Tilt.Ceiling),
  linearizeRadiation = false,
  energyDynamics=Modelica.Fluid.Types.Dynamics.FixedInitial,
  T_start=273.15+24,
  nPorts=6,
  lat=0.6227) "Room model in Tokyo (Latitude = 35.69 degrees North)"
```

DAE

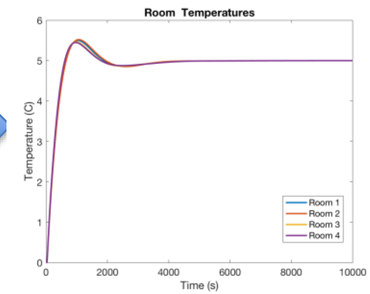
$$0 = F(x, \dot{x}, u)$$

Compiler

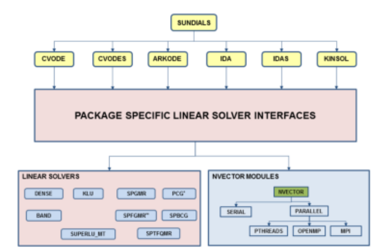
Sparse Symbolic Jacobian

$$J = \frac{\partial F}{\partial x} + \alpha \frac{\partial F}{\partial \dot{x}}$$

Simulation



Solver (IDA)



Feedback control design

- Model reduction
- Freq domain methods
- $H_\infty, H_2, \ldots$
- MPC

Other tools do not support this !

TRNSYS 17 EnergyPlus eQUEST

Case Study: New HVAC Architectures for Next-Gen Buildings

Current building trends:

- Tighter buildings
- Lower loads mean smaller systems
- More integration between envelope and space conditioning
- Distributed generation and storage
- Adaptable to occupants over time
- Collaborative building design

Future HVAC systems will have more subsystems that interact strongly and have less control authority over the space conditions.

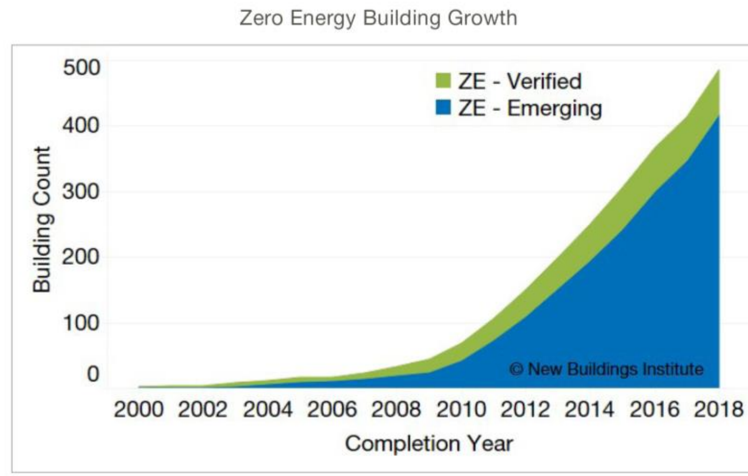
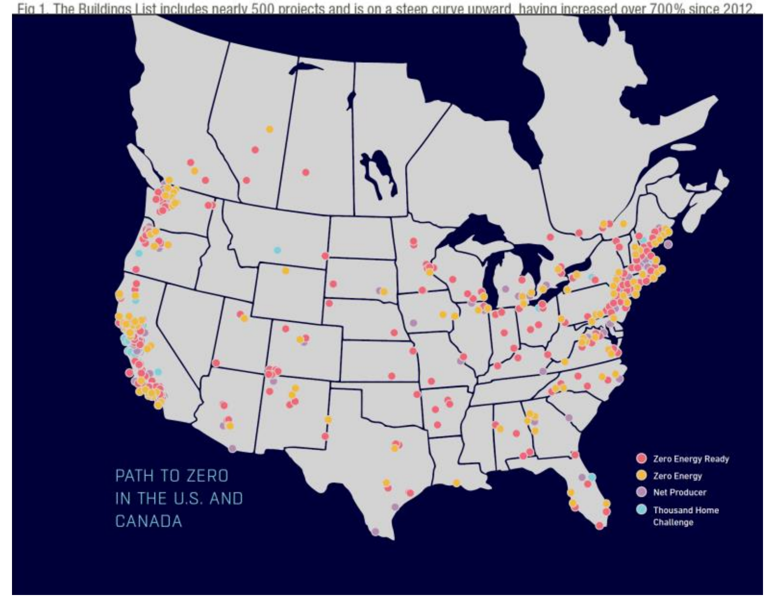
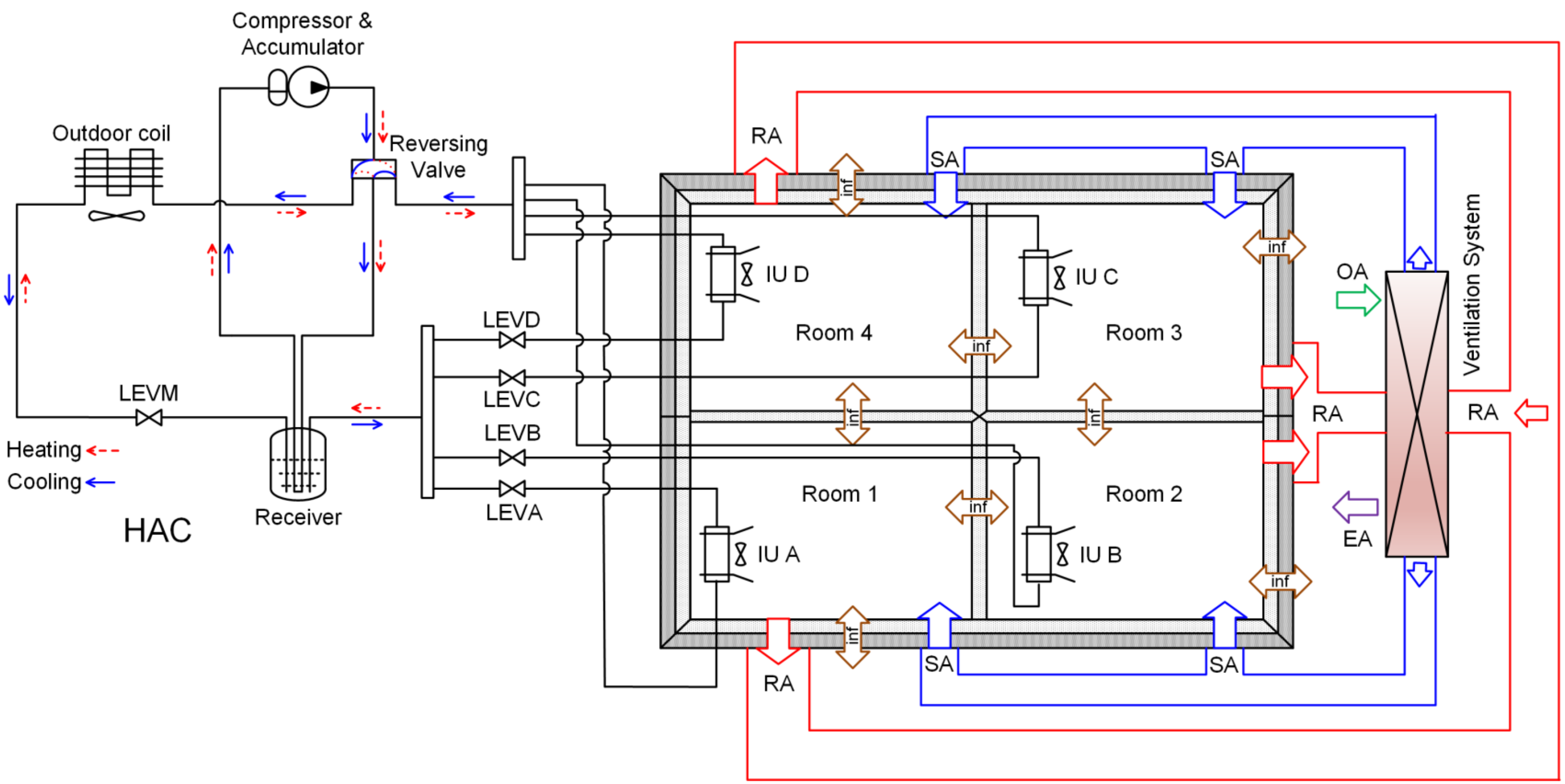


Fig 1. The Buildings List includes nearly 500 projects and is on a steep curve upward, having increased over 700% since 2012.



Integrated Control of Multi-Zone Buildings

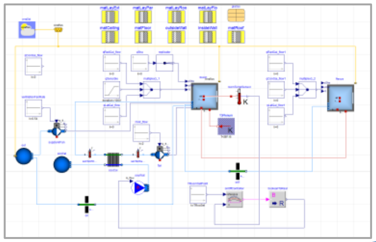


Objectives:

- Study model-based process for dynamic system and controls design
- Evaluate operation of alternate system architectures
  - Compare 3 different ventilation systems

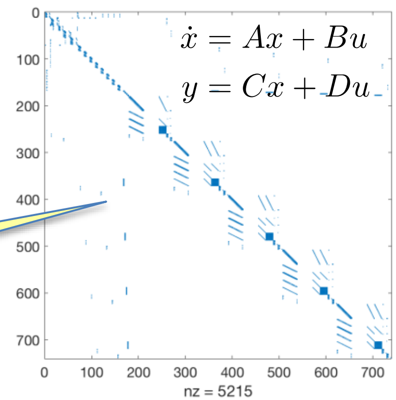
Model-Order Reduction for Control Design

Step 1:
Linearize Modelica



Matlab commands “dcgain”, “bode” etc. **fail**

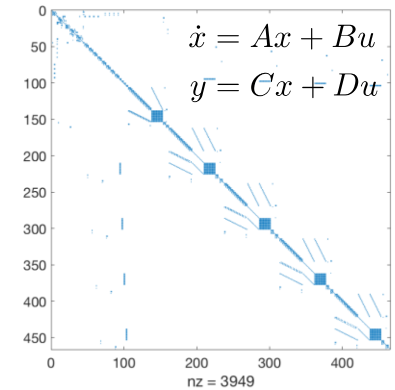
“Raw model”
1000+ states, 99% sparse



Step 2:
Symbolically remove...

- Energy states
- 0 rows and columns

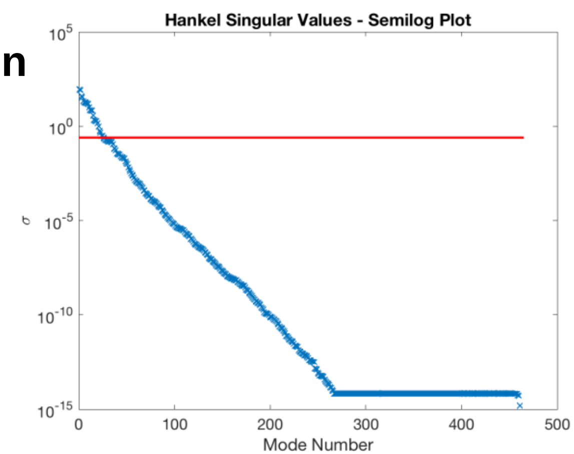
“Full model”
700+ states, 98% sparse



☛ Matlab commands work!

Step 3:
Hankel norm truncation

- Scale inputs, outputs
- Frequency pre-scale
- Modal decomposition
- Remove slow modes
- Balance
- Truncate



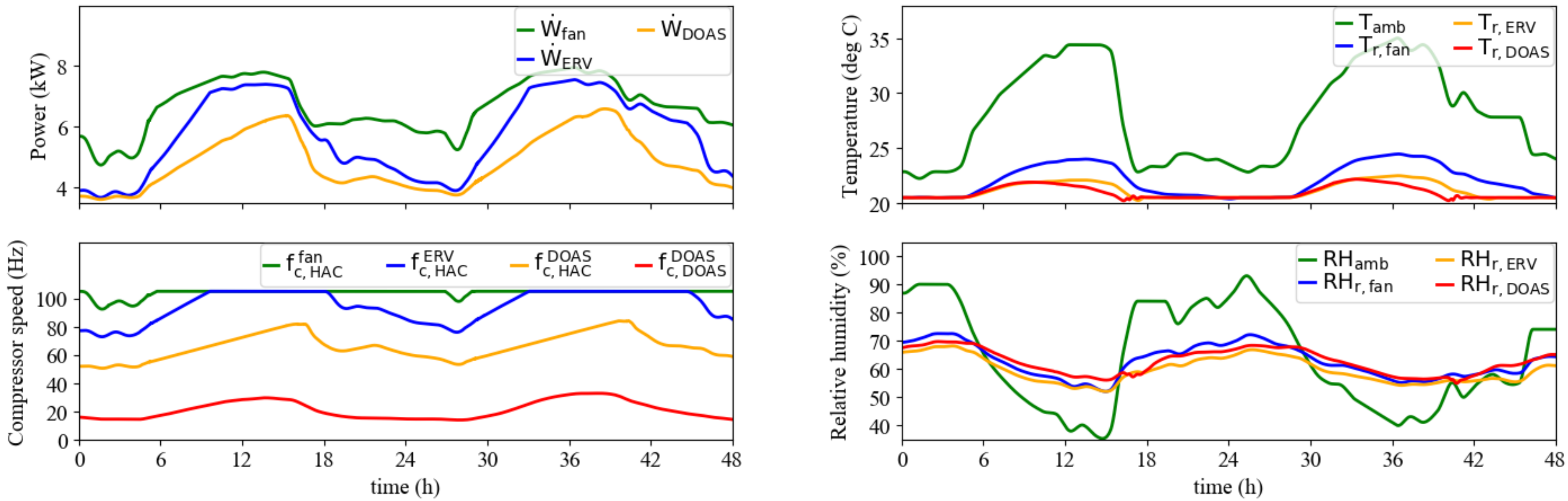
Step 4:
Singular perturbation

- Eliminate fast modes
- Gives DC terms

“Reduced model”
40 states, dense

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$
$$Y = H(j\omega)U$$

Results



- Interactions between the closed-loop systems and the time-varying solar load have a significant effect on the building energy performance

- The ventilation load significantly affects the performance of the VRF system

Ventilation System	Energy Consumption (kWh)
Fan	320.7
ERV	275.8
DOAS	232.7

But I Thought This Was A Julia Talk!

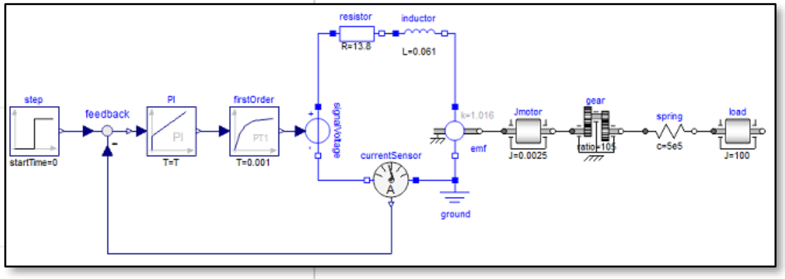
README.md

Modia.jl

build passing | coverage 54% | codecov 54%

Introduction

Modia is a domain specific extension of Julia for **modeling and simulation of physical systems.**



Modelica is almost 20 years old, and has limitations

- All equations are scalarized for symbolic manipulation
- No union types
- Not designed for parallelization
- Language hasn't evolved with recent advances in computer languages

New Julia DSL for this application by original Modelica designers: ***Modia***

Opportunities

- New tools for large-scale unstructured system science
- The representation used in equation-oriented languages imposes important constraints on the system structure: there are open challenges to adapt and develop mathematical tools in this context
- Leveraging Julia's metaprogramming and multiple dispatch could enable a rich set of potential applications
  - Distributed equation-oriented simulation (HPC)
  - Variable-index DAE systems
  - Multi-rate and sparse DAE solvers and approaches to improve simulation time
  - Leveraging locality in DAE models to improve simulation time and MOR
  - Large-scale estimation, dynamic model reduction, and uncertainty quantification
  - Large-scale EO system optimization (adjoint optimization, PDE-constrained optimization)
- An equation-oriented Julia-based DSL facilitates the application of advances in scientific computing to MBSE

Thanks for your time!

Any questions?

Contact:

Chris Laughman

laughman@merl.com