Modeling and simulation of multi-domain physical systems

Shripad Chandrachood
Sr. Application Engineer
MathWorks – Natick, MA
Motivation

- Many real-world physical systems are multi-domain: Combination of Electrical, Mechanical, Hydraulic etc.

- Successful controller development requires thorough and accurate understanding of the physical system
Motivation

- Many real-world physical systems are multi-domain: Combination of Electrical, Mechanical, Hydraulic etc.

- Successful controller development requires thorough and accurate understanding of the physical system
Motivation

- Many real-world physical systems are multi-domain: Combination of Electrical, Mechanical, Hydraulic etc.

- Successful controller development requires thorough and accurate understanding of the physical system
Case Study: Wind Turbine Modeling
Agenda

- System-level simulation of a wind turbine
- Benefits of the Model-Based Design process
- Modeling multi-domain physical systems using Simscape
- Overview of MathWorks physical modeling tools
- Testing system integration
Key Points

- Efficient design of wind turbines requires a smooth, continuous development process

- The ability to easily adjust the level of model fidelity enables earlier detection of integration issues

- Automatically documenting tests can speed up design iterations and provide necessary proof of system performance
Traditional Development Process

Requirements and Specifications

- Requirements are unclear, incomplete, and not integrated in design process

Design and Implementation

- Separate simulation models are difficult to integrate

Integration and Test

- Errors are found too late in the process using expensive prototypes
Model-Based Design Process

Simulation Model
- Control
- Electrical
- Embedded Software
- Mechanical

Requirements and Specifications

Save time by developing in a single simulation environment

Produce better designs by continuously comparing design and specification

Lower costs by using HIL tests and fewer hardware prototypes
Demo: Wind Turbine System-Level Simulation
Linking Specification and Design

Situation:

Wind Turbine Req.
1. Control System
2. Electrical System
3. Mechanical System

Simulation Model

Problem: Matching design to specification is difficult

Solution: Use Simulink Verification and Validation to link the design and specification
Detect System Integration Issues In Simulation

**Model:**
- Mechanical
- Hydraulic
- Electrical
- Controls
- Supervisory Logic
- Park
- Spin
- Drag
- Lift
- Wind
- Actuator (Ideal)
- Actuator (Realistic)
- System (Include)
- System (Ignore)

**Problem:** Test for system integration issues before building hardware prototypes

**Solution:** Use the Simulink environment to integrate the separate systems in simulation
Simulating plant and controller in one environment allows you to optimize system-level performance.
- Automate tuning process using optimization algorithms
- Accelerate process using parallel computing
Detect Integration Issues Earlier

- Controls engineers and domain specialists can work together to **detect integration issues in simulation**
  - Convert plant models to C code for hardware-in-the-loop tests
  - Distribute models to other internal users without extra licenses
  - Distribute models to external users while protecting IP
Build Accurate Models Quickly

- Requires less domain knowledge and programming effort than traditional methods
  - Spend more time developing, less time modeling
MathWorks Physical Modeling Products

- **Multibody mechanics (3-D)**
- **Electrical power systems**
- **SimPowerSystems**
- **SimMechanics**
- **SimDriveline**
- **SimHydraulics**
- **SimElectronics**
- **SimScape**

**Simscape**
- Mechanical
- Hydraulic
- Electrical
- Thermal
- Pneumatic
- Magnetic

Custom Domains via Simscape Language

**Multidomain physical systems**

**Fluid power and control**

**Electromechanical and electronic systems**
MathWorks Physical Modeling Products

- SimMechanics
- SimDriveline
- SimHydraulics
- SimElectronics
- SimPowerSystems

Simscape

- Mechanical
- Hydraulic
- Electrical
- Thermal
- Pneumatic
- Magnetic
Case Study: Wind Turbine Modeling

- Blades
- Wind
- Pitch
- Lift, Drag
- Rotor Speed
- Hub
- Nacelle
- Geartrain
- Generator
- Tower
- Grid
- Generator Speed
- Yaw
Wind turbines have mechanical and electrical components and are enabled by power electronics.
Model the Generator and Electrical Grid

Model:

Problem: Model the generator and electrical grid for the wind turbine

Solution: Use SimPowerSystems built in components for induction, synchronous generators, or wind turbine modules
Model the Utility Grid

- Model localized power distribution grids that include synchronous and asynchronous machines, transformers, transmission line dynamics and a variety of specialized FACTS devices
- Use SimPowerSystems *Discrete Simulation Type* for detailed transient analysis or switch to *Phasor Simulation Type* for steady state analysis
Model the Geartrain in SimDriveline

Problem: Model the geartrain of the wind turbine within the Simulink environment to determine the torque loads.

Solution: Use SimDriveline to model the geartrain.
Model DC to DC Power Converters

- Construct, test and re-use multiple power electronic converter topologies quickly and efficiently

---

**Buck (step-down) Converter**

**Boost (step-up) Converter**

**Buck-Boost Converter**
Model DC to DC Power Converters

- Balance model fidelity and simulation speed according to your needs

**SimPowerSystems**
- Piecewise linear systems solution
- Multiphase bridges and pulse generators
- Transient and harmonic analysis
- Faster simulation

**SimElectronics**
- Nonlinear simultaneous equations solution
- Include temperature effects
- SPICE level switching device models
- Detailed simulation
Model DC to AC Power Inverters

- Build complex, multi-phase, multi-level inverter circuits using the Universal Bridge from the SimPowerSystems library

- Use the built-in tools in SimPowerSystems to perform harmonic analysis directly on your simulation model

- Use average voltage models or ideal switching algorithms for control design and faster simulation
Mechanical Design: Import the Entire Design Directly from a CAD Tool

The translator automatically generates the SimMechanics model using CAD information.
Mechanical Design: Using SimMechanics Link to Import from a CAD Tool

- Automatically create SimMechanics models from a CAD assembly
  - Converts mass and inertia to rigid bodies
  - Converts mate definitions to joints
  - Creates STL files for use with SimMechanics visualization
- Directly connects SolidWorks, ProEngineer and Inventor
- Public API for other CAD tools
- Free download from www.mathworks.com
  - Requires MATLAB
Mechanical Design: Using SimMechanics Link to Import from a CAD Tool

- SimMechanics Link can import XML file to:
  - Create a new model
  - Add to existing model
  - Update an existing model
  - Replace a portion of an existing model

- Engineering teams can collaborate on the same system without doing redundant work
Problem: Model the blade pitch actuation linkage in the Simulink environment

Solution: Use SimMechanics to model the mechanical linkage
Flexible Bodies in SimMechanics

- **Lumped parameter approach**
  - Chain of rigid elements connected by spring/dampers

- **Finite Element Approach**
  - Export eigenmodes from FE program and import into Simulink
  - Superimpose deflection due to flexibility onto rigid body motion
Flexible Blades in SimMechanics

Model:

Problem: Model the blades as flexible cantilevers in the Simulink environment.

Solution: Use SimMechanics to model the flexible body with two approaches (lumped parameter, imported FEA modes).
Wind Turbine Control Systems

- **Blade pitch control system**
  - Adjust pitch angle to regulate rotational speed

- **Supervisory control system**
  - Analyze operating conditions to determine state of turbine to enable/disable operation
Controlling Rotor Speed Using the Pitch Angle

Model:

Problem: Control the pitch angle so that the generator shaft spins at nominal speed

Solution: Use Simulink to determine the pitch angle by controlling the angle of attack
Integrating Simscape with Simulink

Controller (SL/SF) + Plant (Simscape)

Input signal unit: psi

Input signal unit: mm

Hydraulic Pressure Source

Simulink-PS Converter

Ideal Translational Motion Sensor

PS-Simulink Converter
Automatically Document Tests And Results

Situation:

Design Change → Test → Evaluate Results → Document

Problem: Evaluate test results quickly to make design changes and document the results

Solution: Use Simulink Report Generator to automatically document tests and results
Automatically Generating C Code

Model:

<table>
<thead>
<tr>
<th>P1</th>
<th>P2</th>
<th>Px</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>...</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Problem: Minimize the simulation time on a parameter sweep to determine controller parameters.

Solution: Use Simulink Coder to create standalone executable.
Test Controller Hardware and Performance of Generated Control Code

**Problem:** Test control algorithm on actual controller hardware without using prototypes

**Solution:** Generate C code for controller and plant to run hardware-in-the-loop tests
HIL Test Setup

**HIL System**

Anemometer measures wind speed and provides input signal to HIL system.

Bachmann electronic M1 hardware controller runs turbine model in real time.

Pitch command is communicated via CAN to pitch actuators.
Pitch angle is sensed and sent back to main controller via CAN.

Model is configured for HIL tests, converted to C code, and downloaded to controller.
Model-Based Design

Plant Modeling ➔ Control Design ➔ Testing

- SimMechanics
- Simscape
- SimElectronics
- SimPowerSystems
- Simulink Design Optimization
- Optimization Toolbox

- Simulink Control Design
- Control System Toolbox
- Simulink Design Optimization
- Optimization Toolbox

- xPC Target
- MATLAB Coder
- Simulink Coder
- DSP System Toolbox

MATLAB & Simulink
Summary

- Optimize system performance
  - Develop in a single environment
  - No cosimulation

- Find problems before building hardware using HIL

- Discover integration problems using simulation
  - No cosimulation

- Create accurate, reusable plant models quickly and easily
More on Physical System Modeling…
Modeling Physical Systems With MathWorks Products

Modeling Approaches

First Principles Modeling

Data-Driven Modeling

Programming (MATLAB, C)
Block Diagram (Simulink)
Modeling Language (Simscape language)
Symbolic Methods (Symbolic Math Toolbox)

Physical Networks (Simscape and other Physical Modeling products)

Statistical Methods (Model Based Calibration Toolbox)
System Identification (System Identification Toolbox)

Neural Networks (Neural Network Toolbox)
Parameter Tuning (Simulink Design Optimization)
**Modeling Approaches**

- **First Principles**
  - Programming
  - Block Diagram
  - Modeling Language
  - Symbolic Methods

- **Data-Driven**

  - **Purpose:** Explore design or physical parameters
  - **Requirements:**
    - Physics of system are well-known
    - **System-level** equations can be derived and implemented
Modeling Approaches

First Principles

- Programming
- Block Diagram
- Modeling Language
- Symbolic Methods

Data-Driven

- Physical Networks

- Purpose: Explore design or physical parameters
- Requirements:
  - Physics of system are well-known
  - Component-level models exist or can be created
Modeling Approaches

- **Purpose:** Model an existing design (real or virtual)
- **Requirements:**
  - Relevant set of measured data is available
  - Design and physical parameters will not be changed
Modeling Approaches

First Principles

Data-Driven

Parameter Tuning

- **Purpose:** Ensuring parameter values are accurate
- **Requirements:**
  - Relevant set of measured data is available
  - Physically meaningful parameters can be automatically tuned

\[ k = ? \]
\[ b = ? \]
Modeling Approaches

- Use the approach appropriate for your situation
- Each of these approaches can be leveraged using MathWorks products
Simscape and Physical Modeling Products

Simscape

- Mechanical
- Hydraulic
- Electrical
- Thermal
- Pneumatic
- Magnetic

Custom Domains via Simscape Language

Multidomain physical systems

Electrical power systems

- SimPowerSystems
- SimElectronics

Fluid power and control

- SimHydraulics

Multibody mechanics (3-D)

- SimMechanics

Mechanical systems (1-D)

- SimDriveline

Electromechanical and electronic systems

- SimElectronics
Quickly Build Intuitive Models

Models that reflect system Structure are easier to understand, modify, and reuse.
Quickly Build Intuitive Models

Models that reflect system structure are easier to understand, modify, and reuse.
Electrical Systems in Equations

\[ v = K_e \omega + i_m R_{\text{wind}} + L_{\text{wind}} \frac{di_m}{dt} \]

\[ T = K_t i_m - D \omega - J \frac{d\omega}{dt} \]
Simscape model advantages
- Easier to read than equations
- Quicker to create
- More intuitive – easier to explain to other engineers

\[ v = K_v \omega + i_m R_{\text{wind}} + L_{\text{wind}} \frac{di_m}{dt} \]

\[ T = K_t i_m - D_\omega - J \frac{d\omega}{dt} \]
Physical network approach (Simscape) vs. Signal Flow approach (Simulink)

- **Simulink Blocks** are causal
  - Transfer functions
  - Input and output ports (signal flow)
  - Graphically model **system equations**

- **PhysMod “Blocks”** are acausal
  - Bi-directional **energy flow**
  - Domain-specific physical ports (electrical, hydraulic…)
  - Graphically model **system topology**
Demo: Signal Flow vs. Network Approach

Electrical example: RC circuit

\[ i = \frac{v}{r} \quad \text{Simulink (Block diagram) Model} \]

\[ i = c \frac{dv}{dt} \quad \text{Simscape (Network) Model} \]

\[ \omega_p = \frac{1}{RC} \]

Simulation Results:
Signal Flow vs. Network Approach

with hardware interactions

Cascading two filters gives different results … Why?
Signal Flow vs. Network Approach

Modeling interactions

Simscape model

\[ v_{in} \quad i_{in} \quad c \quad i_{out} \quad v_{out} \]

Original Simulink model:

No interactions
No conservation

System interactions

\[
\frac{v_{in} - v_{out}}{R} = i_{in}
\]

\[
v_{out} = \frac{1}{C} \int (i_{in} - i_{out}) dt
\]

New Simulink model …

… with interactions
Signal Flow vs. Network Approach

Summary

- **Signal Flow (Simulink)**
  - No interaction between blocks unless explicitly specified
  - The user must be aware the interactions exist *and* model the effect
  - System changes require model reformulation

- **Network Approach (Simscape)**
  - System of equations automatically created and solved by simulator
  - Enforces conservation (KCL, Newton’s Law, etc…) at every junction
  - Topology changes easy to make, interactions occur automatically
Simscape Pre-defined Domains

- Simscape generalizes conservation laws across physical domains
  - $\Sigma i = 0$ at every junction (where $i$ is the through variable)
- Simscape pre-defined domains:

<table>
<thead>
<tr>
<th>Port Type</th>
<th>Across Variable</th>
<th>Through Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical</td>
<td>Voltage</td>
<td>Current</td>
</tr>
<tr>
<td>Hydraulic</td>
<td>Pressure</td>
<td>Flow rate</td>
</tr>
<tr>
<td>Mechanical (rotational)</td>
<td>Angular velocity</td>
<td>Torque</td>
</tr>
<tr>
<td>Mechanical (translational)</td>
<td>Translational velocity</td>
<td>Force</td>
</tr>
<tr>
<td>Pneumatic</td>
<td>Pressure and temperature</td>
<td>Mass flow rate and heat flow</td>
</tr>
<tr>
<td>Thermal</td>
<td>Temperature</td>
<td>Heat flow</td>
</tr>
<tr>
<td>Magnetic</td>
<td>MMF</td>
<td>Flux</td>
</tr>
</tbody>
</table>
Simscape Language Example: DC Motor

- DC motor with inertia load

Motor Equations:

\[ v = K_e \omega + i_m R_{\text{wind}} + L_{\text{wind}} \frac{di_m}{dt} \]

\[ Tq = -K_t i_m + D \omega + J \frac{d\omega}{dt} \]

- \( v \) = Terminal voltage
- \( K_t \) = Torque constant
- \( K_e \) = Back emf constant
- \( \omega \) = Shaft speed
- \( i_m \) = Motor current
- \( R_{\text{wind}} \) = Winding resistance
- \( L_{\text{wind}} \) = Winding inductance
- \( Tq \) = Output torque
- \( D \) = Damping coefficient
- \( J \) = Moment of inertia
Simscape Language Example: DC Motor

**component dc_pm**

**nodes**

```
p = foundation.electrical.electrical;  % p:left
n = foundation.electrical.electrical;  % n:left
r = foundation.mechanical.rotational.rotational;  % r:right
c = foundation.mechanical.rotational.rotational;  % c:right
```

**parameters**

```
Kt = {10 'N*m/A'};  % Torque constant
Ke = {10 'V/(rad/s)'};  % Back EMF Constant
Rwind = {1 'Ohm'};  % Winding Resistance
Lwind = {1e-3 'H'};  % Winding Inductance
J = {1 'kg*m^2'};  % Motor Inertia
B = {1 'N*m/(rad/s)'};  % Motor Damping
```

**variables**

```
theta = { 0, 'rad'};  % Angular Displacement
tq = { 0, 'N*m'};  % Torque through variable
w = { 0, 'rad/s'};  % Ang Vel across variable
i = { 0, 'A'};  % Current through variable
v = { 0, 'V'};  % Voltage across variable
```

**equation**

```
w == theta.der;

v == Ke*w + i*Rwind + Lwind*i.der;  % Motor
tq == -Kt*i + B*w + J*w.der;  % equations
```

**Physical Connections**

**Motor Parameters**

**Internal Variables**

**Motor Equations**
Simscape Language Overview

- Provides first principles modeling at component level
- Enables control over model fidelity
  - Add non-linear effects
  - Include domain interactions (i.e. thermal effects)
  - Supports conditional statements
- Leverage the foundation library source code as starting point
Benefits of Network Modeling Approach

- Network analysis techniques used to solve system
- Burden on simulator not the user to formulate system equations
- Fidelity of system model is determined by component equations
- Network representation easy for non-experts to follow and understand
Multi-Domain System in Simscape

1. Voltage is applied to motor
2. Motor generates torque to spin pump
3. Pump supplies pressure to the hydraulic valve
4. Actuator is used to open/close the valve and direct fluid flow
5. Hydraulic cylinder moves left or right depending on valve position
6. Mechanical load provides opposing force to cylinder
7. Motion sensor feeds position back to controller
8. Control unit provides set point
9. Position control is achieved
10. No system equations derived!
1. Voltage is applied to motor
2. Motor generates torque to spin pump
3. Pump supplies pressure to the hydraulic valve
4. Actuator is used to open/close the valve and direct fluid flow
5. Hydraulic cylinder moves left or right depending on valve position
6. Mechanical load provides opposing force to cylinder
7. Motion sensor feeds position back to controller
8. Control unit provides set point
9. Position control is achieved
10. No system equations derived!
Summary

- Optimize system performance
  - Develop in a single environment
  - No cosimulation

- Find problems before building hardware using HIL

- Discover integration problems using simulation
  - No cosimulation

- Create accurate, reusable plant models quickly and easily
Q&A