MATLAB EXPO 2019

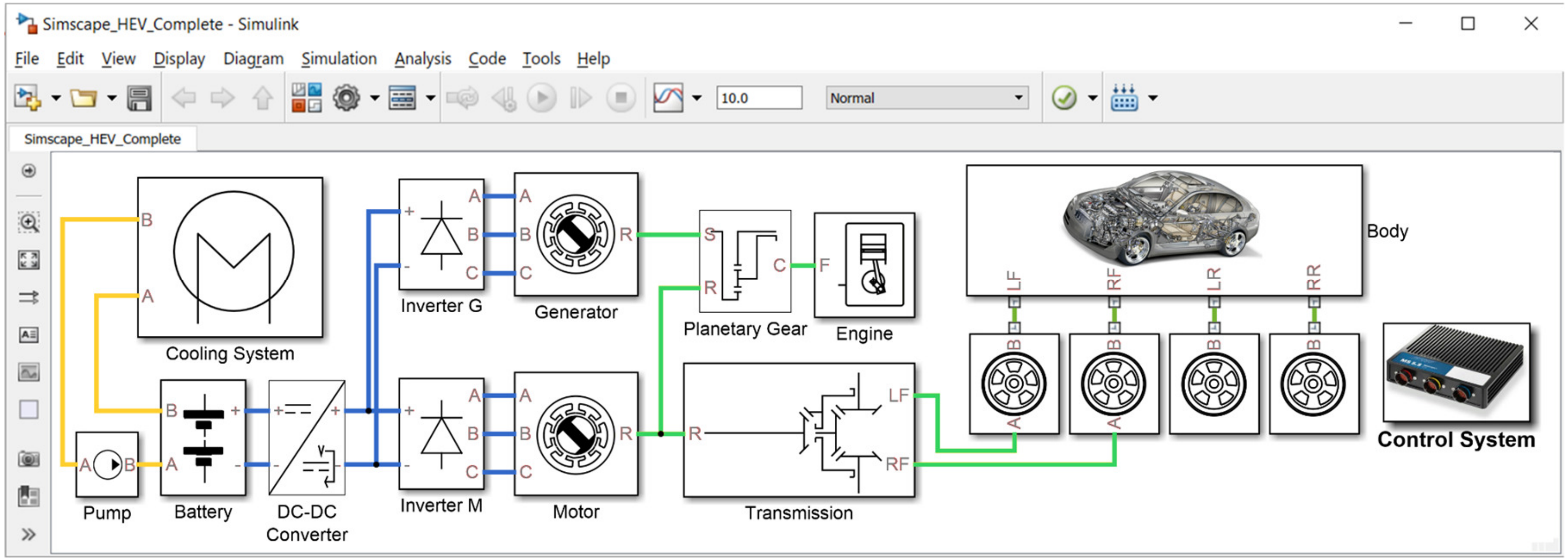
Developing Fit-for-Purpose Simscape™ Models to Support System and Control Design

Rick Hyde

Overview

- Matching engineering design tasks to models
- Examples
  - Hybrid vehicle powertrain - *with focus on need for multiple models*
  - Photovoltaic system - *with focus on plant design and control*
- Tutorial – faulted DC motor

Models must be matched to the engineering design task

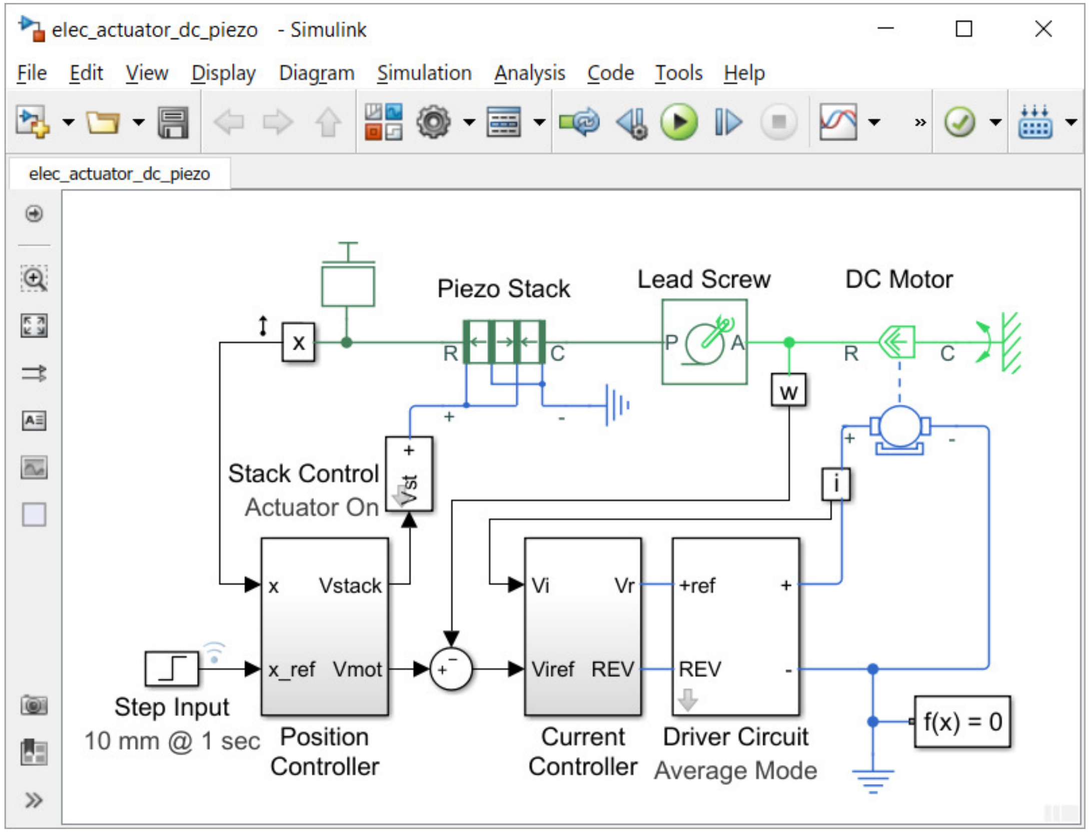


Design task: Predict & optimize system performance

Model requirements:

- Faster than real-time for drive cycle analysis
- Component efficiency information as function of operating point

Models must be matched to the engineering design task

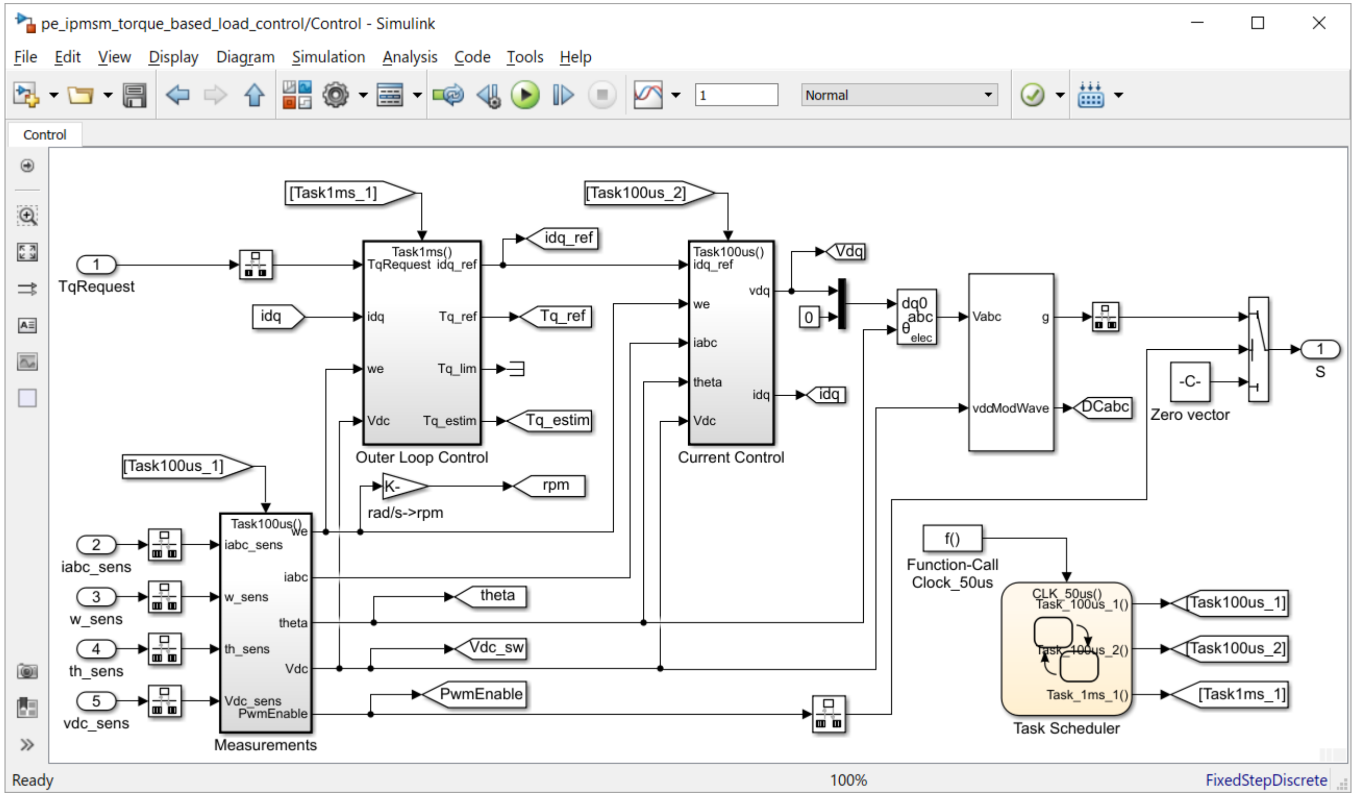


Design task: Assess a disruptive/novel actuation technology

Model requirements:

- Torque-speed characteristics
- Linearizable->frequency response
- Predict losses and heat
- Assess failure modes

Models must be matched to the engineering design task

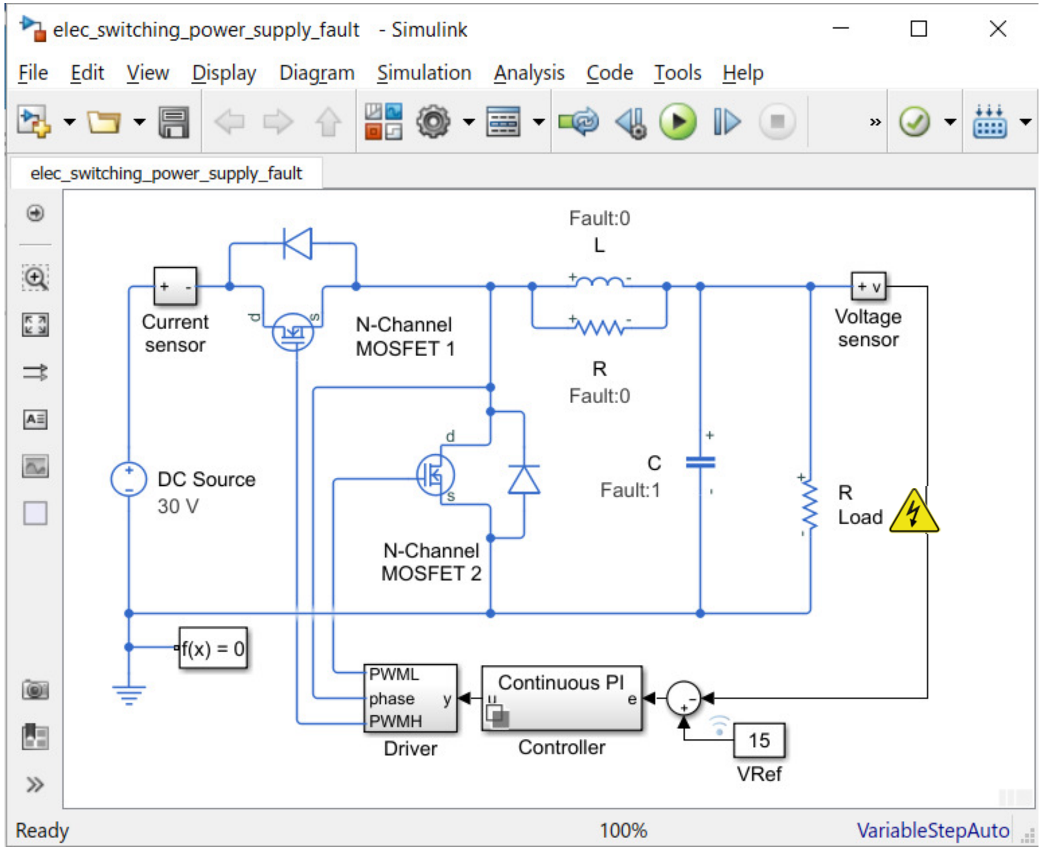


Design task: control system design

Model requirements:

- Linearizable
- Include *relevant* dynamics
- Key tolerances & uncertainties defined
- Physics readily understandable (no black boxes)

Models must be matched to the engineering design task



Design task: reliability assessment

Model requirements:

- Operating limits
- Tolerances
- Fault behaviours
- Behavioural fault triggering

MathWorks is committed to supporting diverse modelling requirements

Datasheet-driven parameterization

Measurement-based parameterization

Design data import from specialist tool

Abstractable to mapped model

Configurable fidelity

Architecturally reconfigurable

Capture operating limits and apply tolerances

Faultable

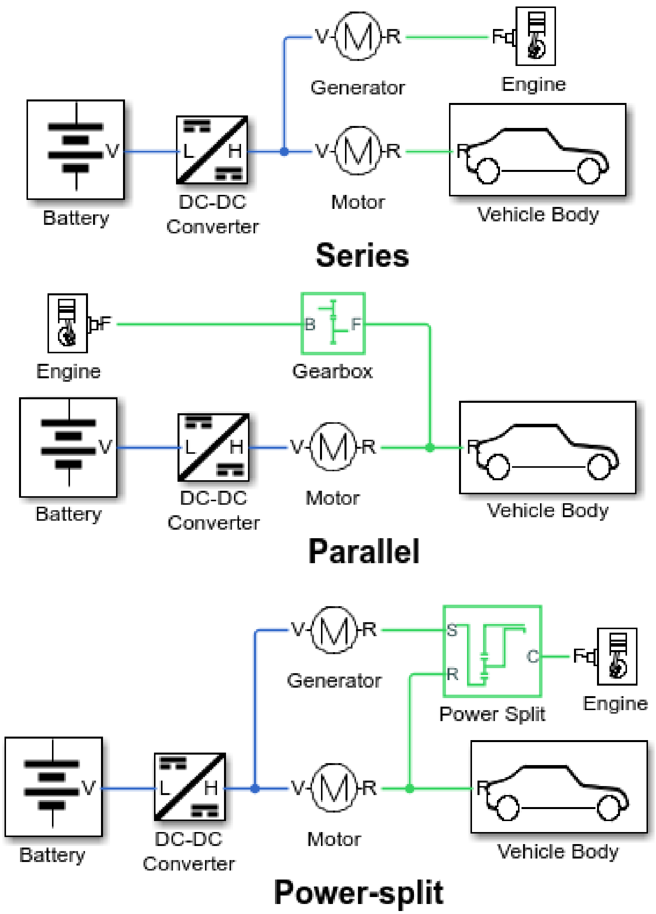
Linearizable

Deployable

Overview

- Matching models to engineering design tasks
- Examples
  - Hybrid vehicle powertrain - *with focus on need for multiple models*
  - Photovoltaic system - *with focus on plant design and control*
- Tutorial – faulted DC motor

Hybrid vehicle powertrain example

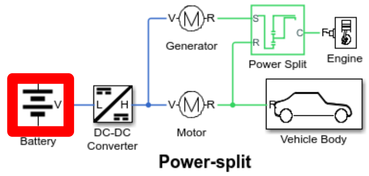


Architecture selection

- Compare series, parallel and power-split
- Re-use same set of subsystems (battery, motor, engine etc)
- i.e. an *architecturally reconfigurable* model
- ***Enabled by Simscape™ physical connections***

1. Trade-off studies/select architecture
2. Optimize selected architecture
3. Detailed component design
4. Design validation

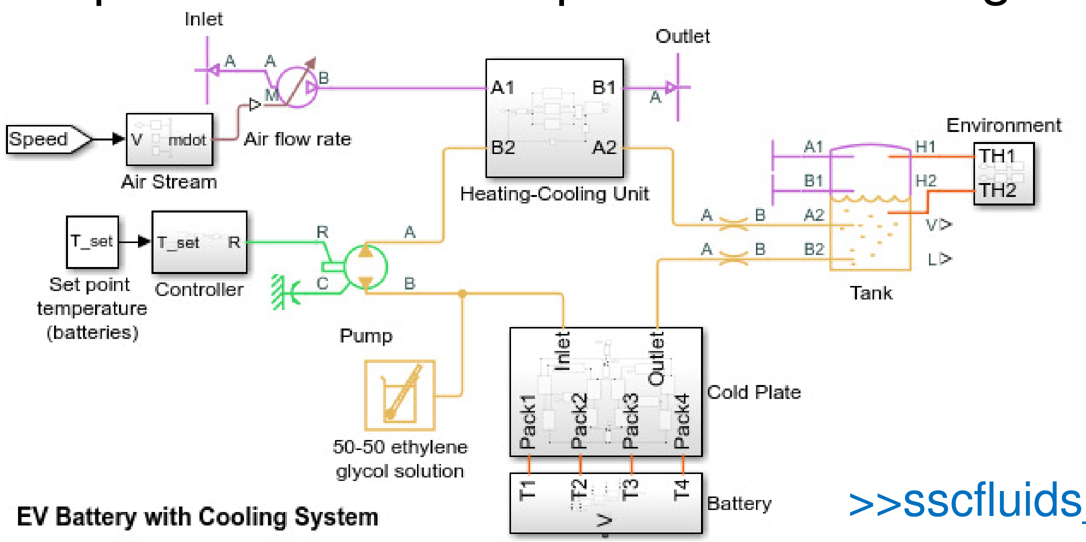
Hybrid vehicle powertrain example



Battery

- *Datasheet-driven* blocks from Simscape Electrical™

- Option to model temperature & cooling



200 x real-time

>>sscfluids_ev_battery_cooling

1.Trade-off studies/select architecture

2. Optimize selected architecture

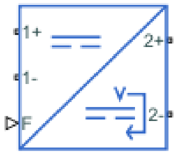
3. Detailed component design

4. Design validation

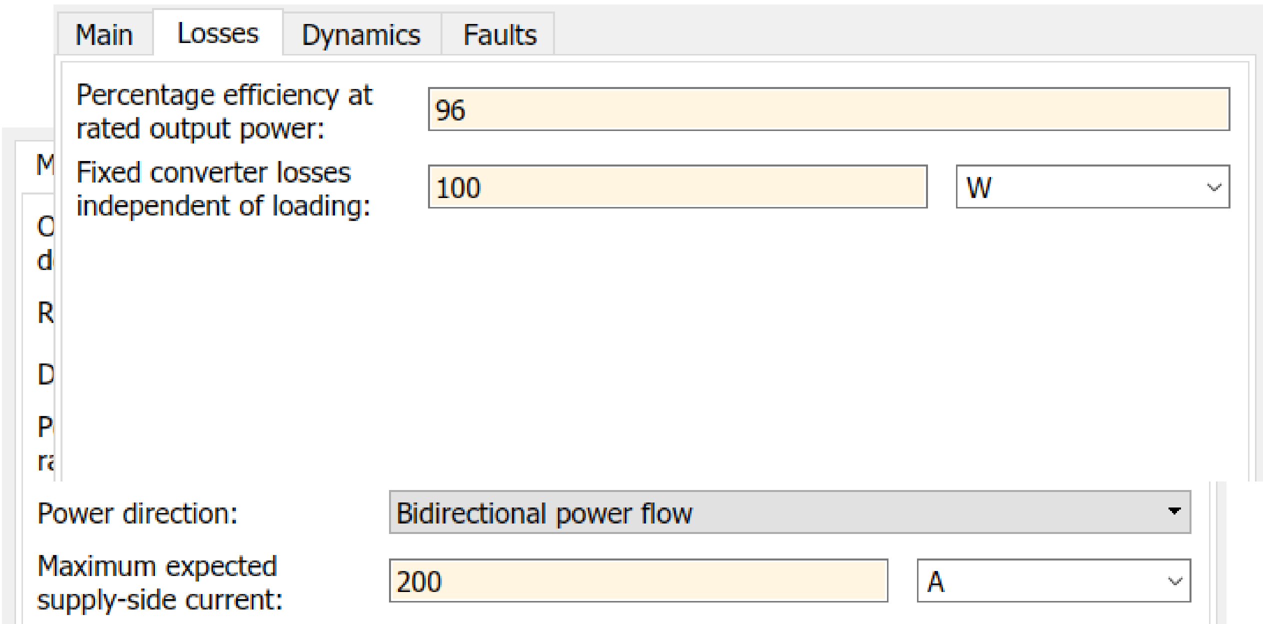
Hybrid vehicle powertrain example

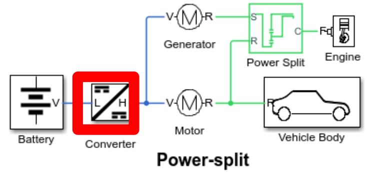
DC-DC Converter

- *Datasheet-driven* blocks from Simscape Electrical™



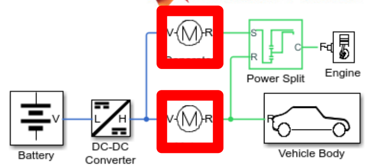
DC-DC Converter





1. Trade-off studies/select architecture
2. Optimize selected architecture
3. Detailed component design
4. Design validation

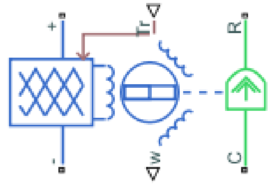
Hybrid vehicle powertrain example



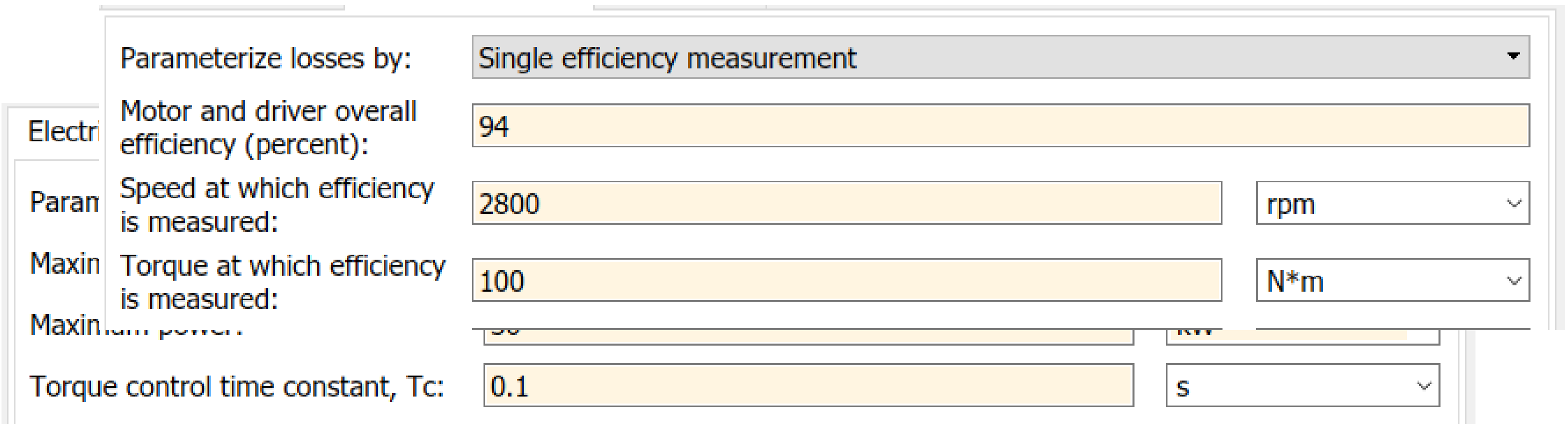
Power-split

PMSM Motor and PMSM Generator

- *Datasheet-driven* blocks from Simscape Electrical™



Simplified PMSM Drive



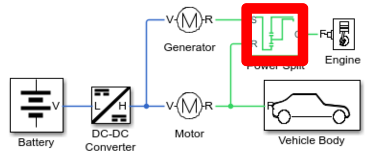
1.Trade-off studies/select architecture

2. Optimize selected architecture

3. Detailed component design

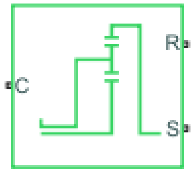
4. Design validation

Hybrid vehicle powertrain example

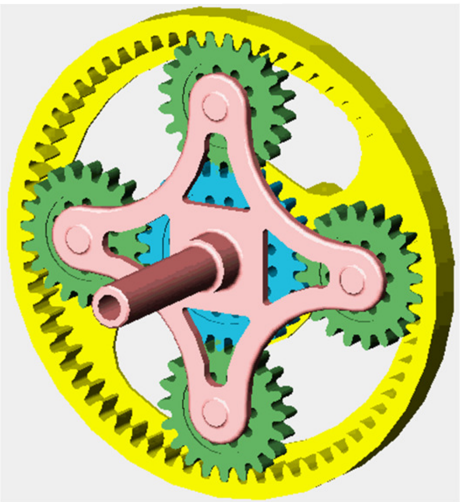


Power-Split (planetary gear)

- *Datasheet-driven* blocks from Simscape Driveline™



Planetary Gear



Main | Meshing Losses | Viscous Losses | Inertia

Friction model: Constant efficiency

Sun-planet and ring-planet ordinary efficiencies: [.96, .98]

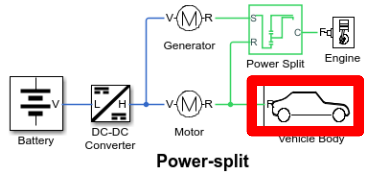
Table 1: Overall efficiency of the basic 2 d.o.f. basic gear train [3, 4].

Input links	Driven links	Overall efficiency η_E
[illegible]	[illegible]	$\frac{1}{[\ldots - 1)]\,\eta_{i(k-j)}}$ [illegible]
j,k	i	$\frac{1}{\left[1+\frac{\omega_k(1-R)}{\omega_j R}\right]\eta_{k(j-i)} + \left[1+\frac{\omega_j R}{\omega_k(1-R)}\right]\eta_{j(k-i)}}$
i,j	k	$\frac{1}{\left[1-\frac{\omega_i}{\omega_j R}\right]\eta_{i(j-k)} + \left[1-\frac{\omega_j R}{\omega_i}\right]\eta_{j(i-k)}}$
i	k,j	$\frac{\eta_{k(i-j)}}{1+\frac{\omega_k(1-R)}{\omega_j R}} + \frac{\eta_{j(i-k)}}{1+\frac{\omega_j R}{\omega_k(1-R)}}$

"A Review of Formulas for the Mechanical Efficiency Analysis of Two Degrees-of-Freedom Epicyclic Gear Trains", Pennestri, Valentini

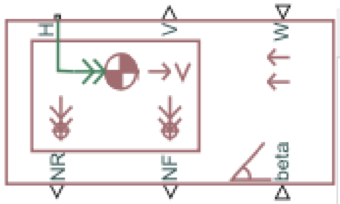
1. Trade-off studies/select architecture
2. Optimize selected architecture
3. Detailed component design
4. Design validation

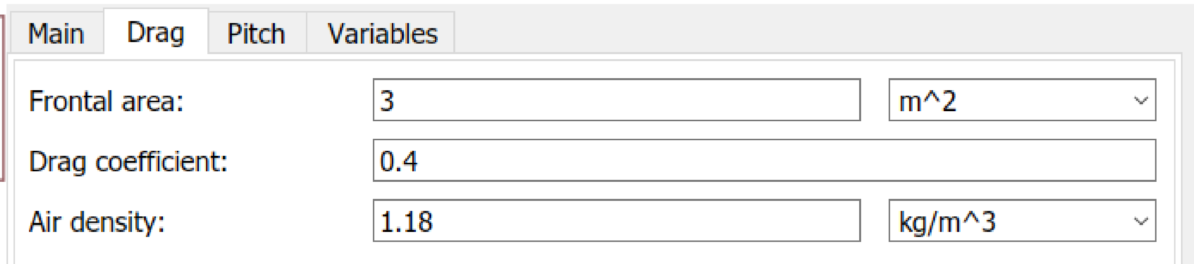
Hybrid vehicle powertrain example



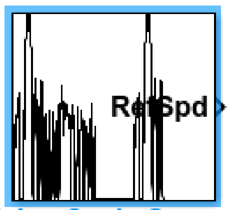
Vehicle and Drive Cycle

- System-level blocks from Simscape Driveline™

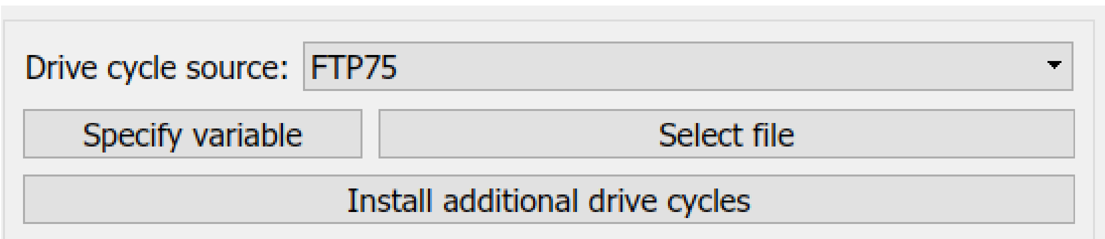




- Automotive-specific blocks from Powertrain Blockset™



Drive Cycle Source
FTP75 (2474 seconds, Cyclic)



1. Trade-off studies/select architecture

2. Optimize selected architecture

3. Detailed component design

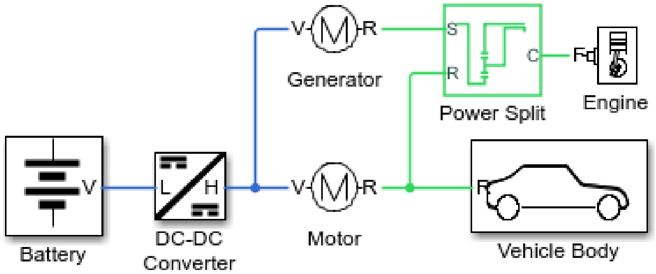
4. Design validation

Hybrid vehicle powertrain example

Optimize selected architecture

- Sensitivity of drive cycle efficiency to key design parameters e.g. battery weight
- Direct optimization of key design parameters

Enabled by MATLAB® scripting and Optimization and Global Optimization Toolboxes



Power-split

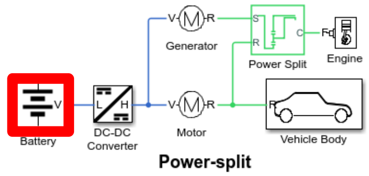
1. Trade-off studies/select architecture

2. Optimize selected architecture

3. Detailed component design

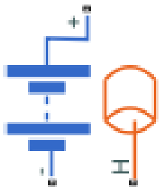
4. Design validation

Hybrid vehicle powertrain example

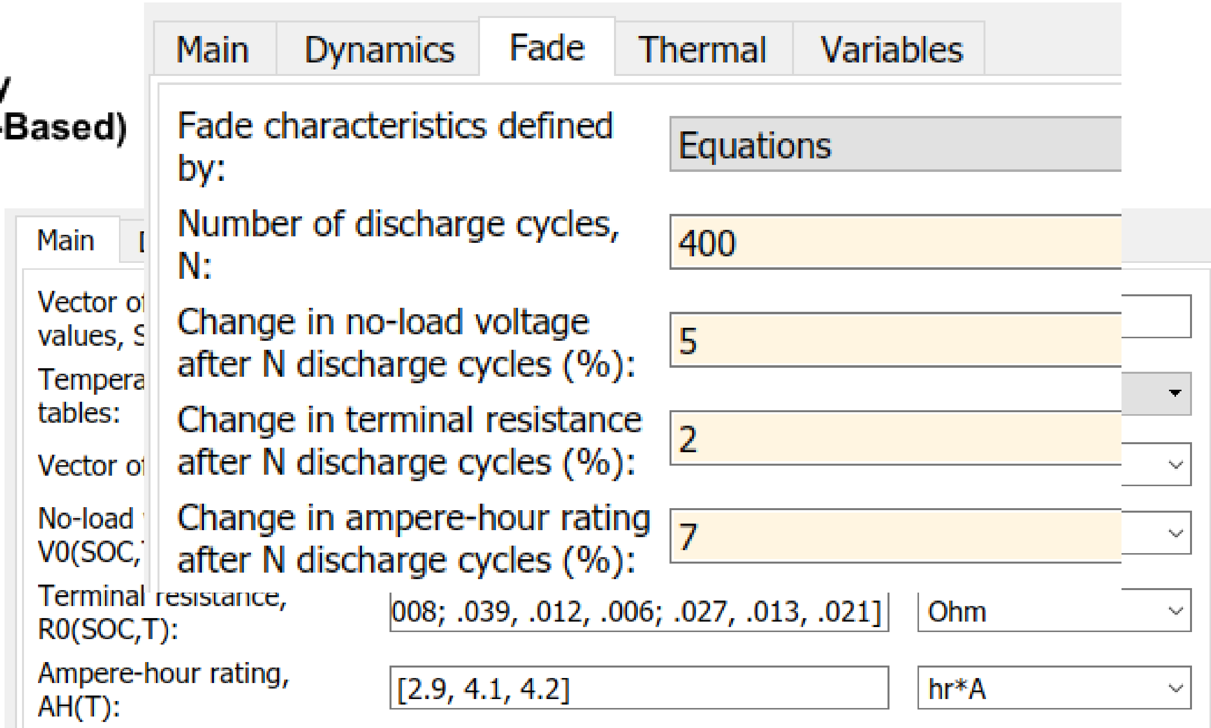


Battery

- *Measurement-based* blocks from Simscape Electrical™



Battery (Table-Based)



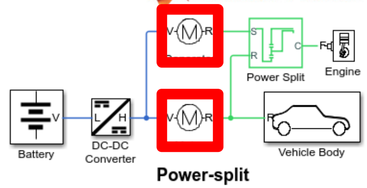
1.Trade-off studies/select architecture

2. Optimize selected architecture

3. Detailed component design

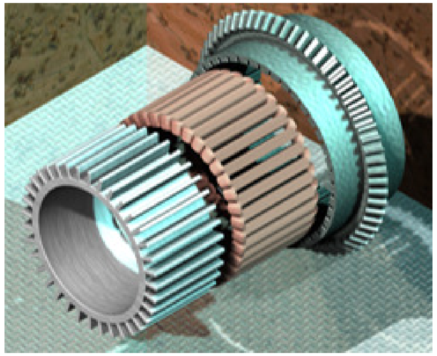
4. Design validation

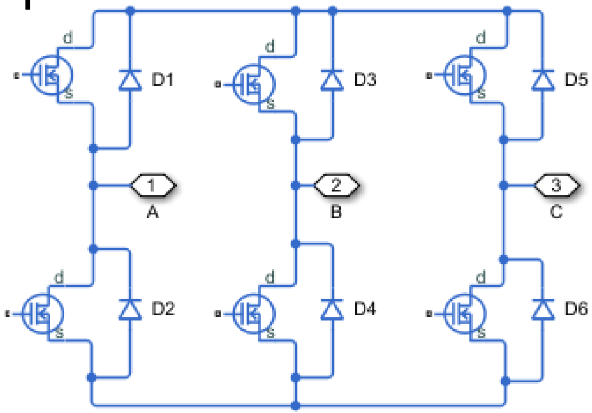
Hybrid vehicle powertrain example

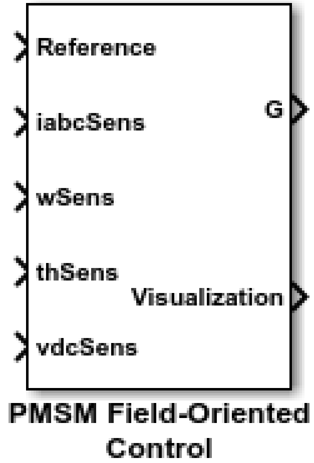


PMSM Drives

- Three constituent parts







- Modelling fidelity must be sufficient to:
  - Predict overall drive efficiency
  - Predict electrical harmonics at DC supply
  - Predict torque ripple applied to load
  - Ensure components stay within operating limits

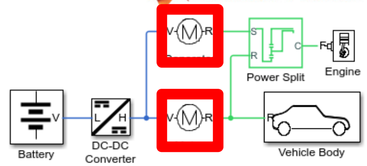
1. Trade-off studies/select architecture

2. Optimize selected architecture

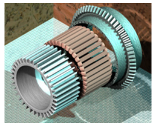
3. Detailed component design

4. Design validation

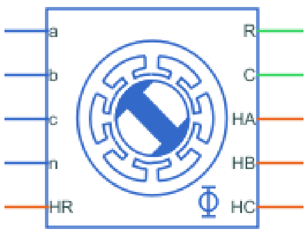
Hybrid vehicle powertrain example



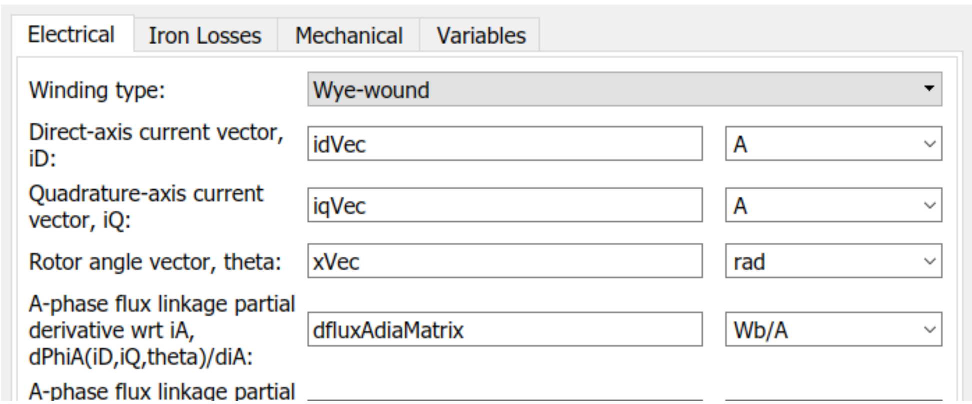
PMSM Motor



- Magnetic finite element level model required to:
  - Determine flux linkage as function of currents and rotor angle
  - Determine iron losses as function of load and speed
- Import into Simscape using the Simscape Electrical™ FEM-Parameterized PMSM block *(Design data import from specialist tool)*



FEM-Parameterized PMSM



1. Trade-off studies/select architecture

↓

2. Optimize selected architecture

↓

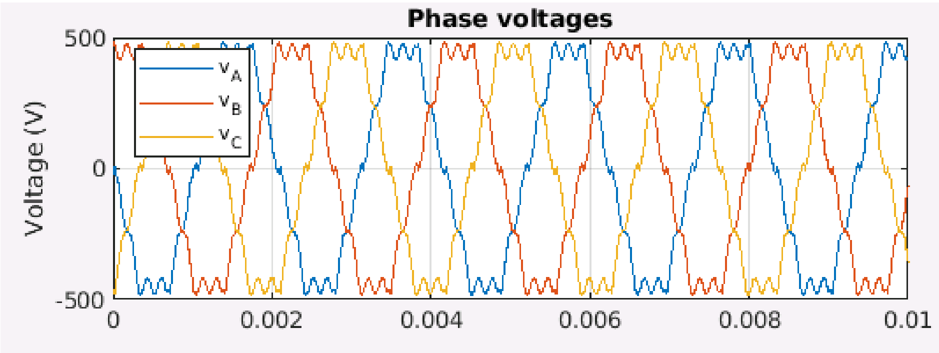
3. Detailed component design

↓

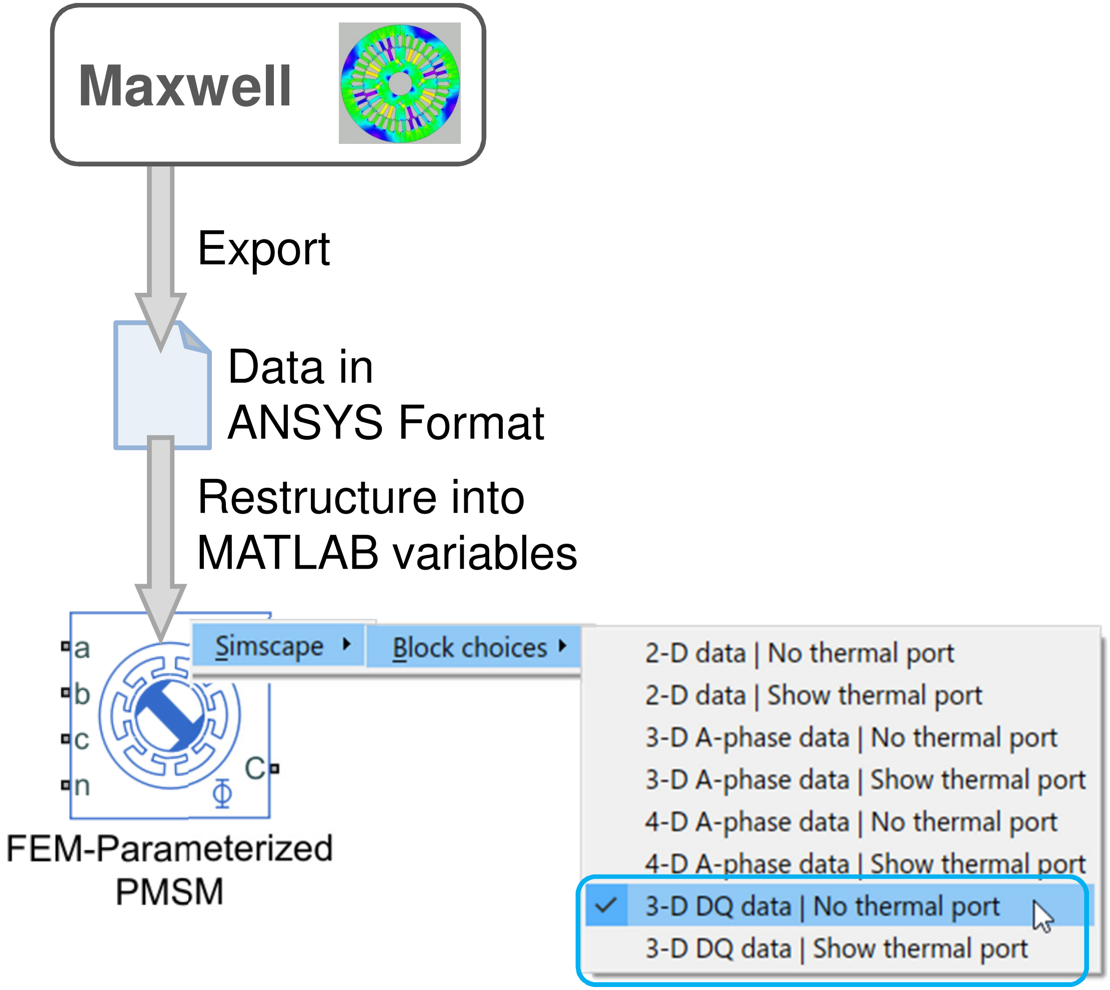
4. Design validation

Simscape Electrical

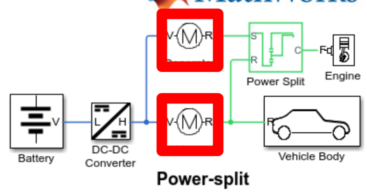
Import from Maxwell (ANSYS)




```
>> elec_import_fem_maxwell

>> elec_import_fem_motorcad
```

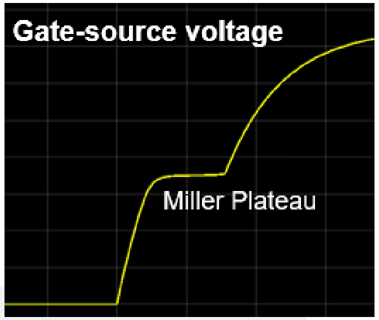


Hybrid vehicle powertrain example



Semiconductor switching devices

- Artefacts that must be modelled to predict timing and losses:
  - Non-linear I-V characteristics
  - Non-linear charge characteristics
- Modelling options
  - Tabulated I-V and capacitance



Main | Junction Capacitance | Temperature Dependence

Parameterization:	Specify tabulated input, reverse transfer and output capacitance	
Input capacitance, Cies:	[80 40 32 28 27.5 27 26.5 26.5 26.5]	nF
Reverse transfer capacitance, Cres:	[55 9 5.5 3.1 2.5 2.1 1.9 1.8 1.7]	nF
Output capacitance, Coes:	[55 20 12 8 6 4.8 4 3.5 3.1]	nF
Corresponding collector-emitter voltages:	[0 1 2 5 10 15 20 25 30]	V

  - Imported SPICE subcircuits

1.Trade-off studies/select architecture

2. Optimize selected architecture

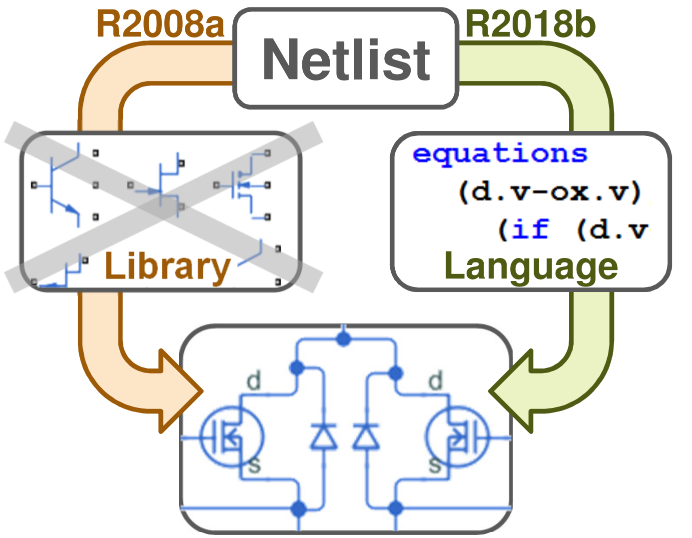
3. Detailed component design

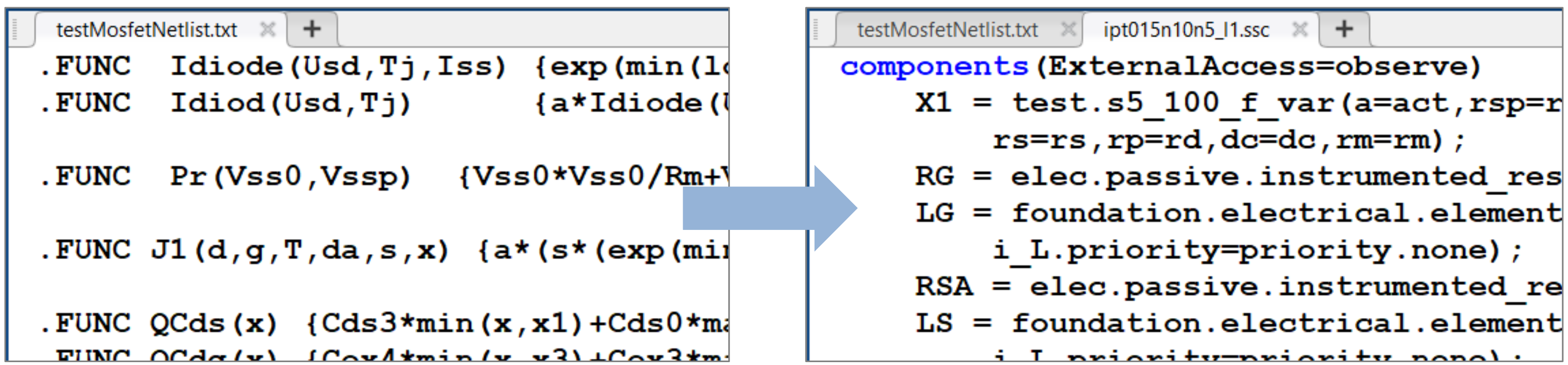
4. Design validation

Simscape Electrical

SPICE Import

- Convert SPICE models into Simscape components
  - Command **`subcircuit2ssc`**
  - Converts subcircuit definition to a Simscape Language component



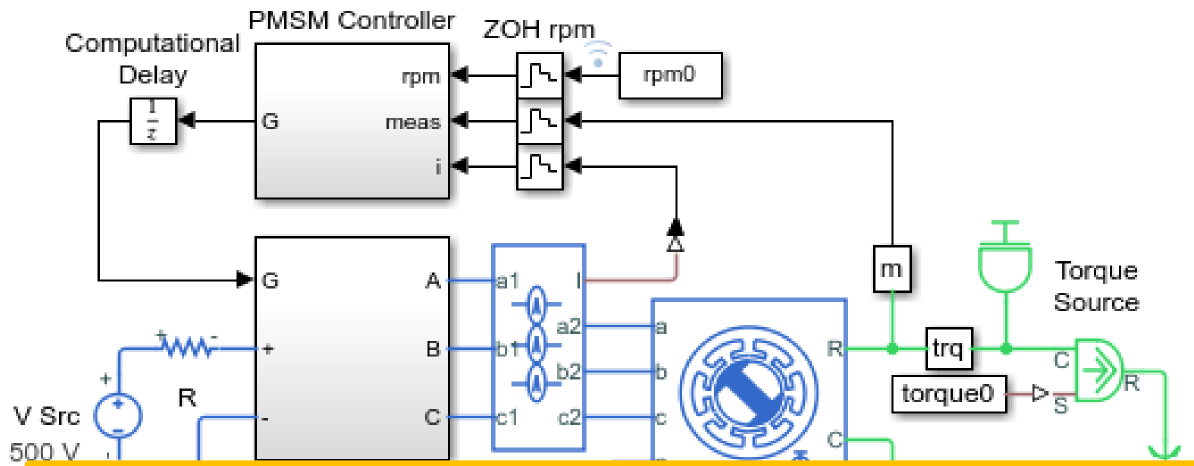



```
.FUNC   Idiode(Usd,Tj,Iss) {exp(min(l
.FUNC   Idiod(Usd,Tj)      {a*Idiode(U

.FUNC   Pr(Vss0,Vssp)   {Vss0*Vss0/Rm+V

.FUNC J1(d,g,T,da,s,x) {a*(s*(exp(mi

.FUNC QCds(x) {Cds3*min(x,x1)+Cds0*ma
```

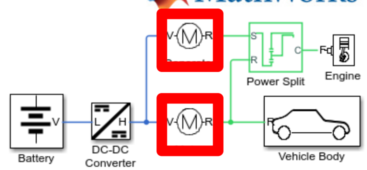
```
components(ExternalAccess=observe)
    X1 = test.s5_100_f_var(a=act,rsp=r
        rs=rs,rp=rd,dc=dc,rm=rm);
    RG = elec.passive.instrumented_res
    LG = foundation.electrical.element
        i_L.priority=priority.none);
    RSA = elec.passive.instrumented_re
    LS = foundation.electrical.element
```

Hybrid vehicle powertrain example

PMSM Drive Test Harness




```
>> ee_motor_pmsm_drive
Running the PMSM Drive Test Harness model to generate simulation data
Efficiency = 86.3198% when speed = 1400rpm and torque = 200Nm
Losses in watts by component are as follows:
                        LoggingNode                                Power

    ______________________________________________________    _____

    {'ee_motor_pmsm_drive.FEM_Parameterized_PMSM'            }    3562.4
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch1.IGBT.transistor'}     161.9
```



Power-split

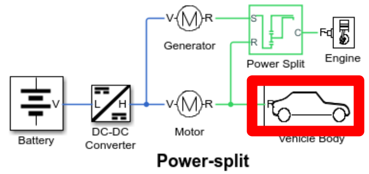
1. Trade-off studies/select architecture

2. Optimize selected architecture

3. Detailed component design

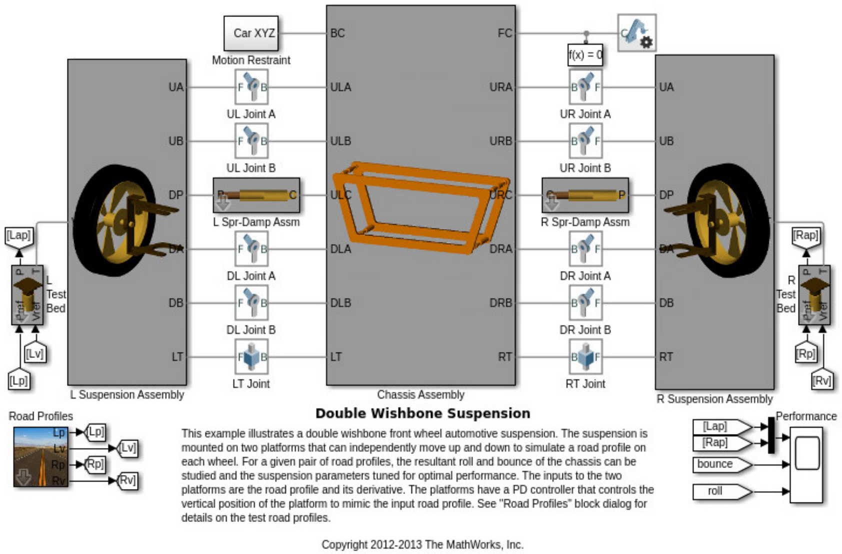
4. Design validation

Hybrid vehicle powertrain example



Vehicle

- Simscape Multibody™
  - Steering and suspension design
  - Determine torque ripple coupling to steering and chassis




```
>> sm_double_wishbone_suspension
```

1. Trade-off studies/select architecture

2. Optimize selected architecture

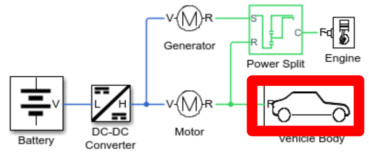
3. Detailed component design

4. Design validation

Hybrid vehicle powertrain example

Design validation

- Revise system-level model (***Abstractable to mapped model)***
  - Update drive cycle efficiencies, predicted range etc.
- Validate detailed components deployed in full vehicle simulation
  - Components must stay within permitted operating points ***(Capture operating limits and apply tolerances)***
  - Assess impact of detailed component behaviour at system level ***(Configurable fidelity)***
- HIL and simulator testing
  - Simulink Coder deploys controllers and physical models to hardware ***(Deployable)***
  - Abstraction of chassis if needed using Vehicle Dynamics Blockset™ ***(Configurable fidelity)***



Power-split

1.Trade-off studies/select architecture

2. Optimize selected architecture

3. Detailed component design

4. Design validation

Overview

- Matching models to engineering design tasks
- Examples
  - Hybrid vehicle powertrain - *with focus on need for multiple models*
  - Photovoltaic system - *with focus on plant design and control*
- Tutorial – faulted DC motor

MathWorks is committed to supporting diverse modelling requirements

Datasheet-driven parameterization

Measurement-based parameterization

Design data import from specialist tool

Abstractable to mapped model

Configurable fidelity

Architecturally reconfigurable

Capture operating limits and apply tolerances

Faultable

Linearizable

Deployable

Overview

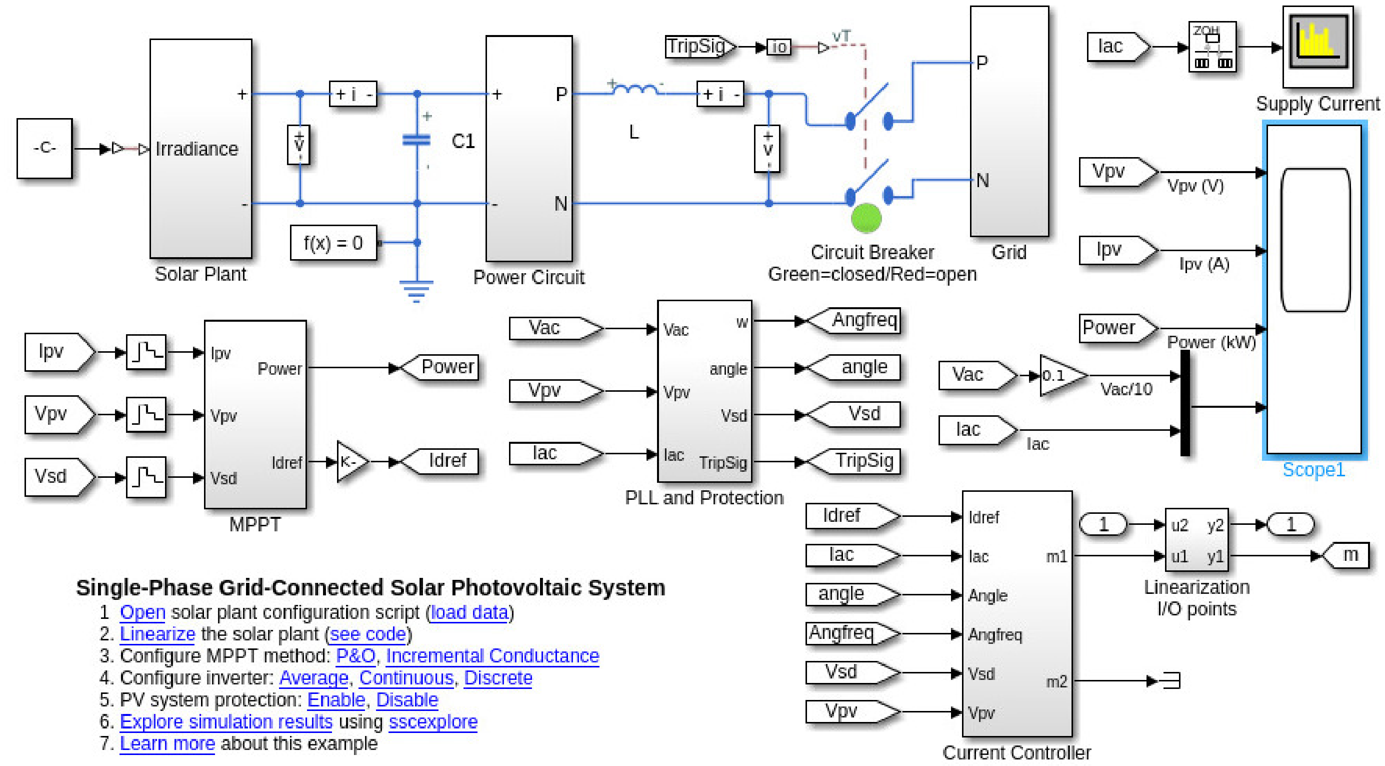
- Matching models to engineering design tasks

- Examples
  - Hybrid vehicle powertrain - *with focus on need for multiple models*
  - Photovoltaic system - *with focus on plant design and control*

- Tutorial – faulted DC motor

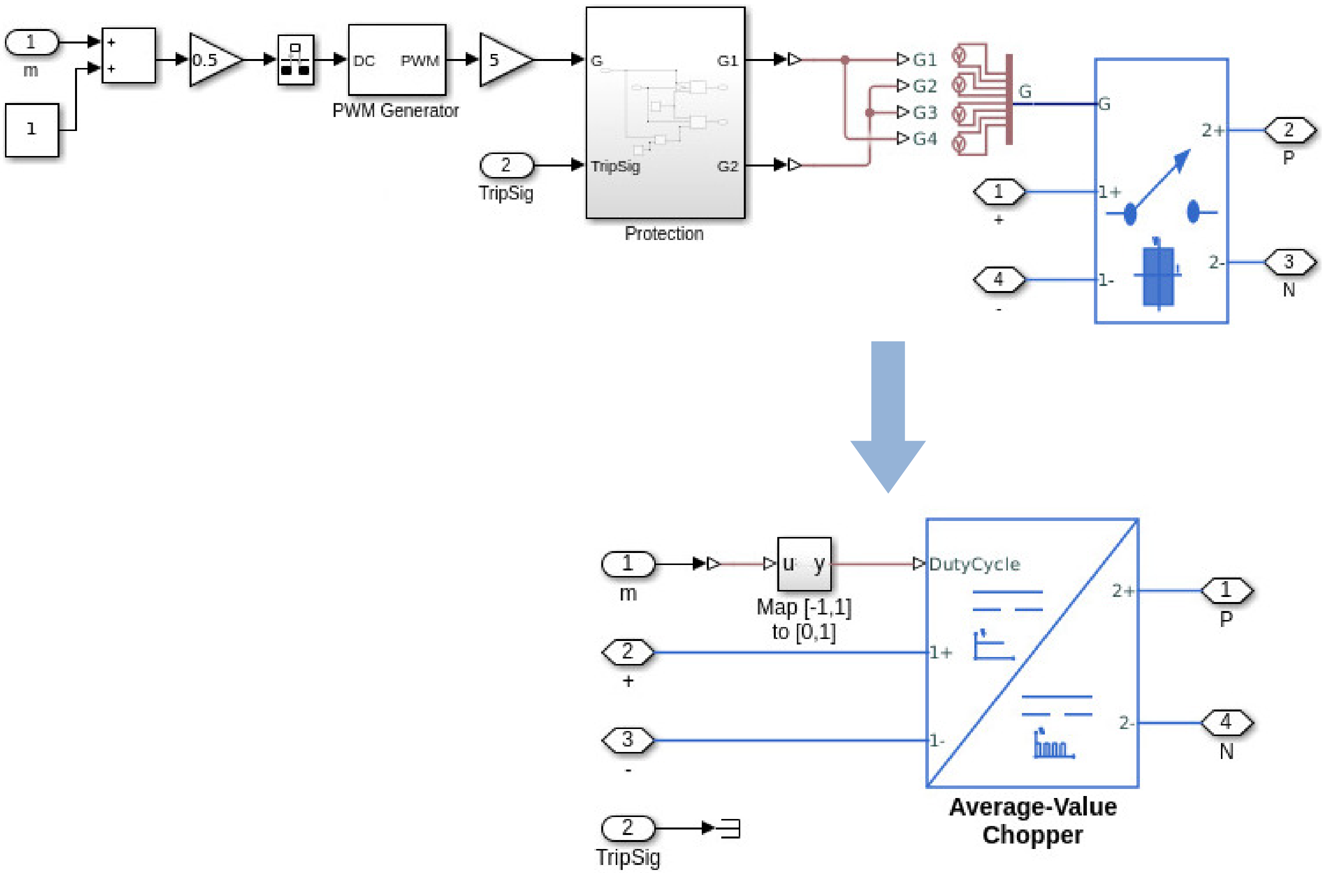
Photovoltaic generator

- **Task:** Assess/check stability margins
- **Solution #1**: Linearize and view Bode plot using Simulink Control Design
- **Solution #2**: Run a frequency-response identification exercise



Photovoltaic generator

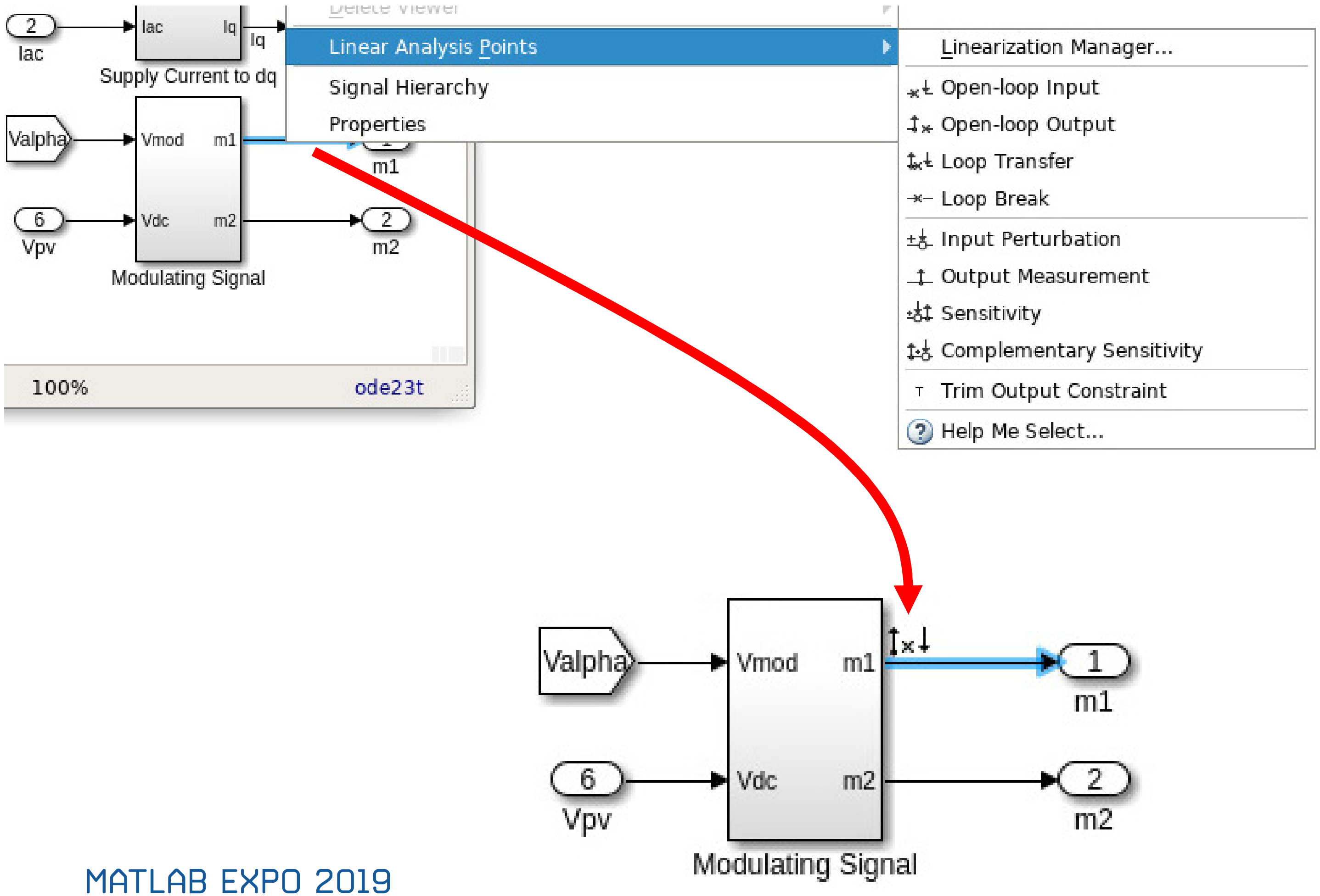
Solution #1: Linearize and view Bode plot using Simulink Control Design



1. Remove switching
   - Simscape Electrical average-value converters
2. Add linearization I/O
   - Simulink Control Design analysis points
3. Define operating point
   - Simulate to desired operating point
4. Linearize
   - Launch Simulink Control Design UI

Photovoltaic generator

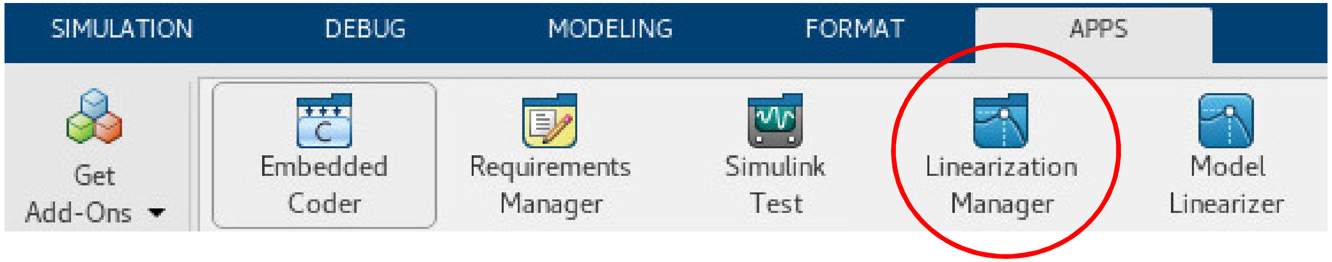
Solution #1: Linearize and view Bode plot using Simulink Control Design

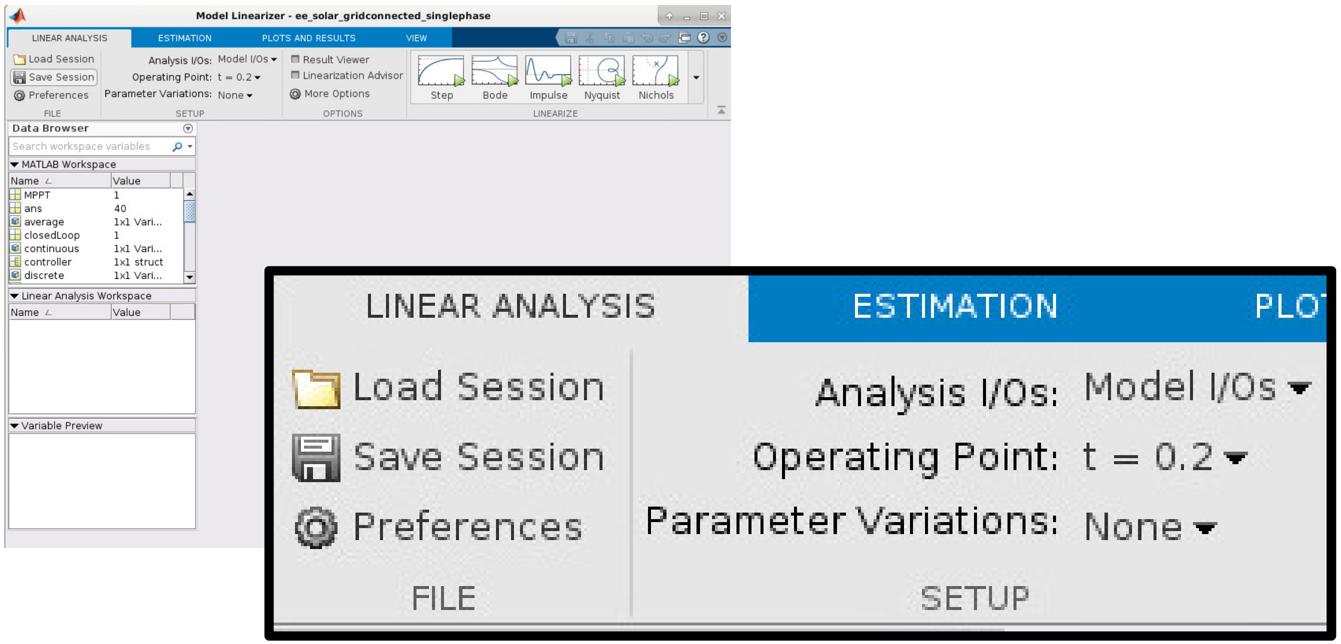


1. Remove switching
   - Simscape Electrical average-value converters
2. Add linearization I/O
   - Simulink Control Design linear analysis points
3. Define operating point
   - Simulate to desired operating point
4. Linearize
   - Launch Simulink Control Design UI

Photovoltaic generator

Solution #1: Linearize and view Bode plot using Simulink Control Design

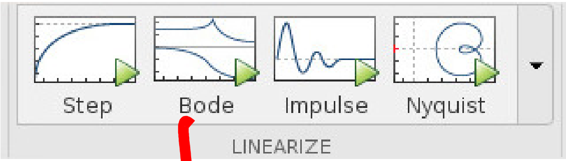


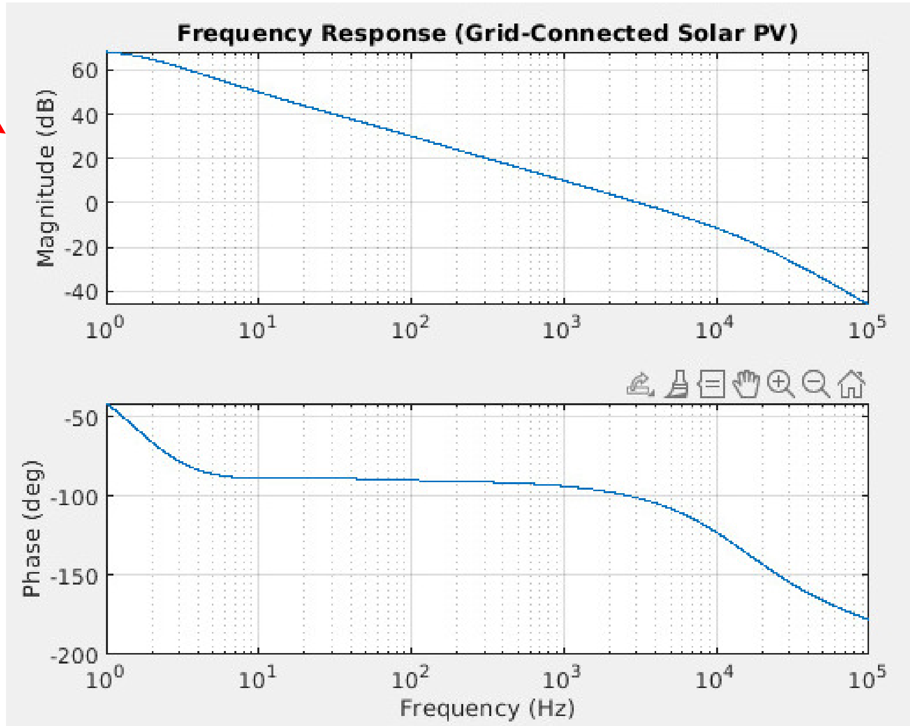


1. Remove switching
   – Simscape Electrical average-value converters
2. Add linearization I/O
   – Simulink Control Design analysis points
3. Define operating point
   – Simulate to desired operating point
4. Linearize
   – Launch Simulink Control Design Model Linearizer UI

Photovoltaic generator

Solution #1: Linearize and view Bode plot using Simulink Control Design

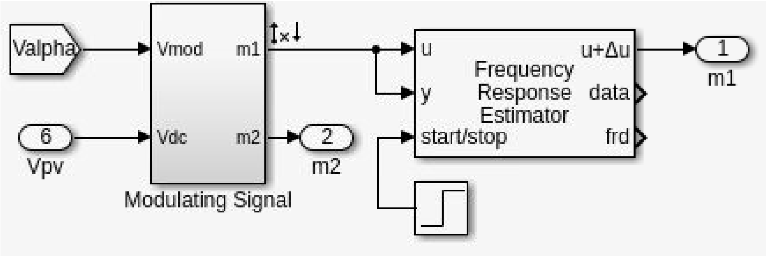


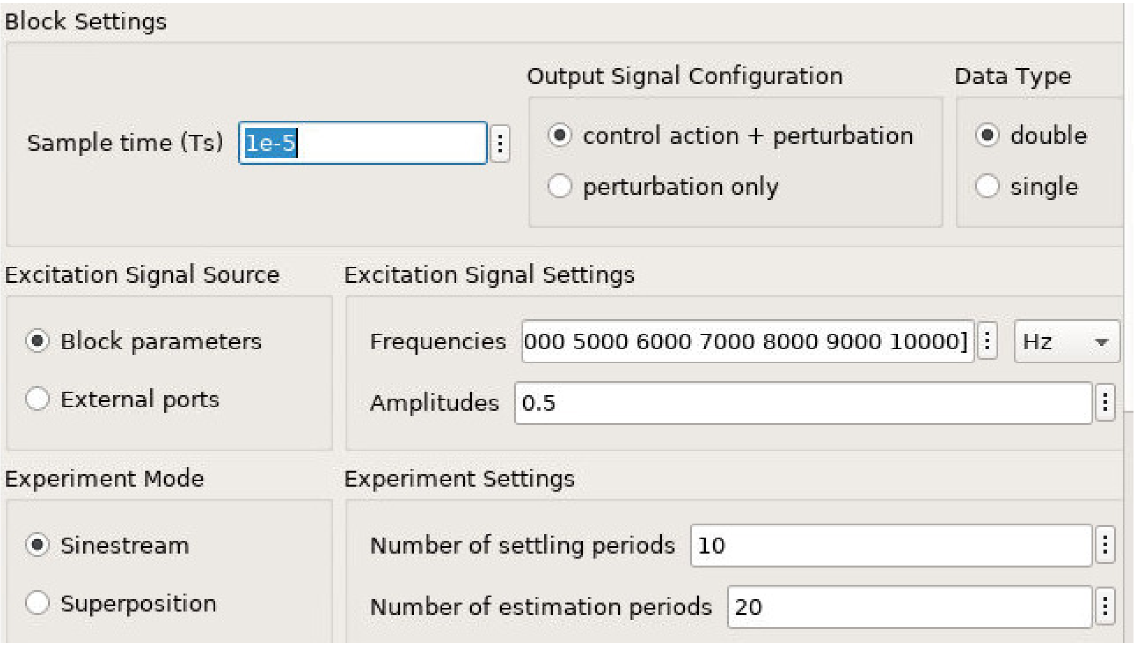


1. Remove switching
   - Simscape Electrical average-value converters
2. Add linearization I/O
   - Simulink Control Design analysis points
3. Define operating point
   - Simulate to desired operating point
4. Linearize
   - Launch Simulink Control Design Model Linearizer UI

Photovoltaic generator

Solution #2: Run a frequency-response identification exercise

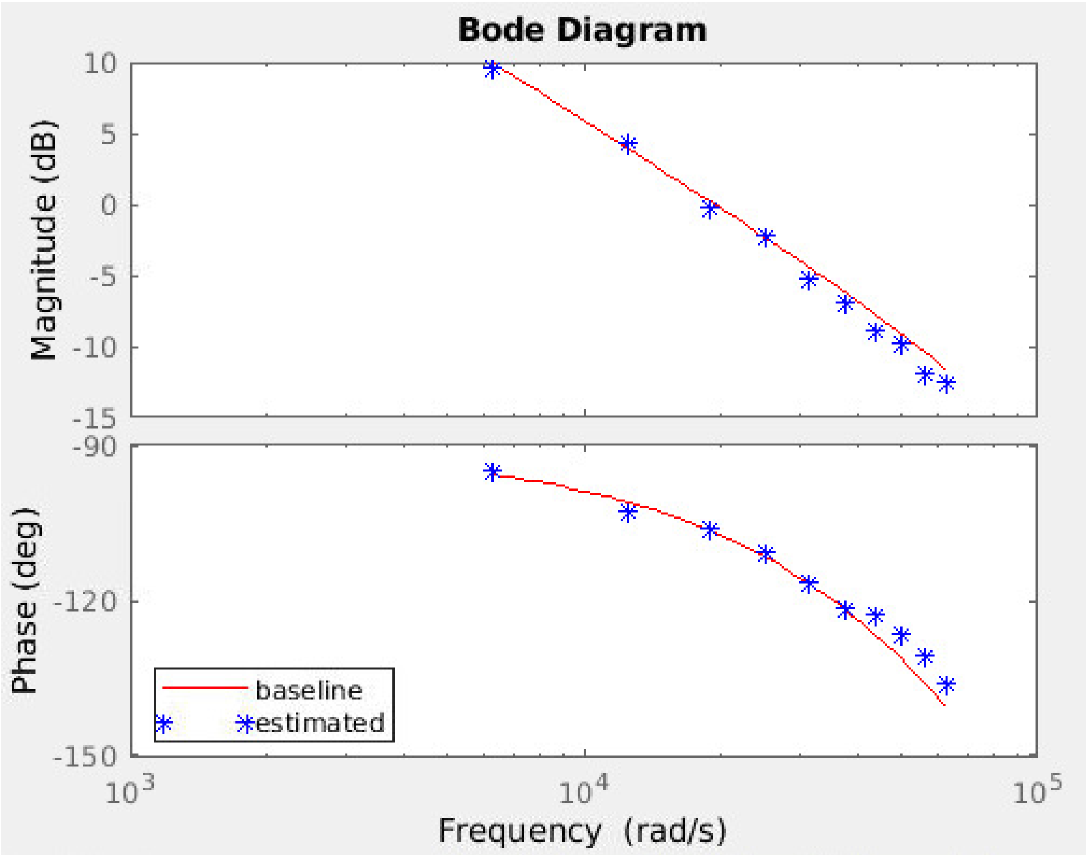




1. Add a Simulink Control Design *Frequency Response Estimator* block.
2. Configure for frequency points and sample time.
3. Run the model.

Photovoltaic generator

Solution #2: Run a frequency-response identification exercise

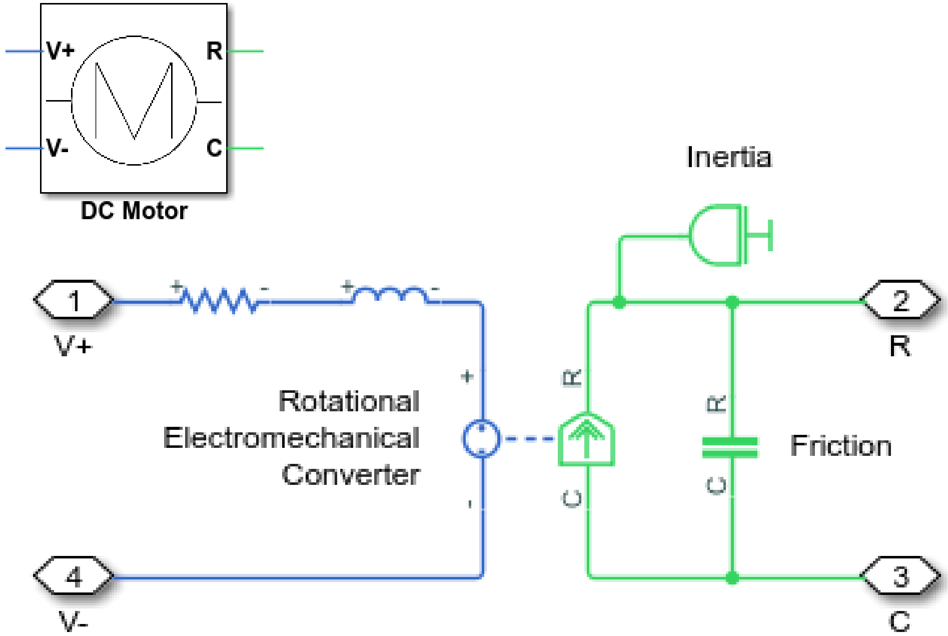


1. Add a Simulink Control Design *Frequency Response Estimator* block.
2. Configure for frequency points and sample time.
3. Run the model.

Overview

- Matching models to engineering design tasks
- Examples
  - Hybrid vehicle powertrain - *with focus on need for multiple models*
  - Photovoltaic system - *with focus on plant design and control*
- Tutorial – faulted DC motor

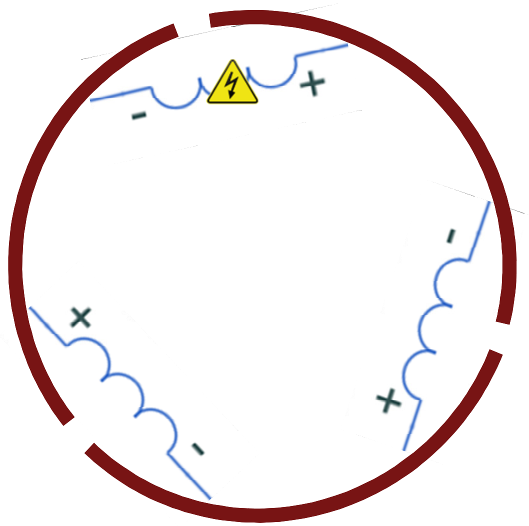
Faultable DC Motor



- Requirements
  - Normal behaviour as example ssc_dcmotor
  - Faulted behaviour: one winding open-circuit

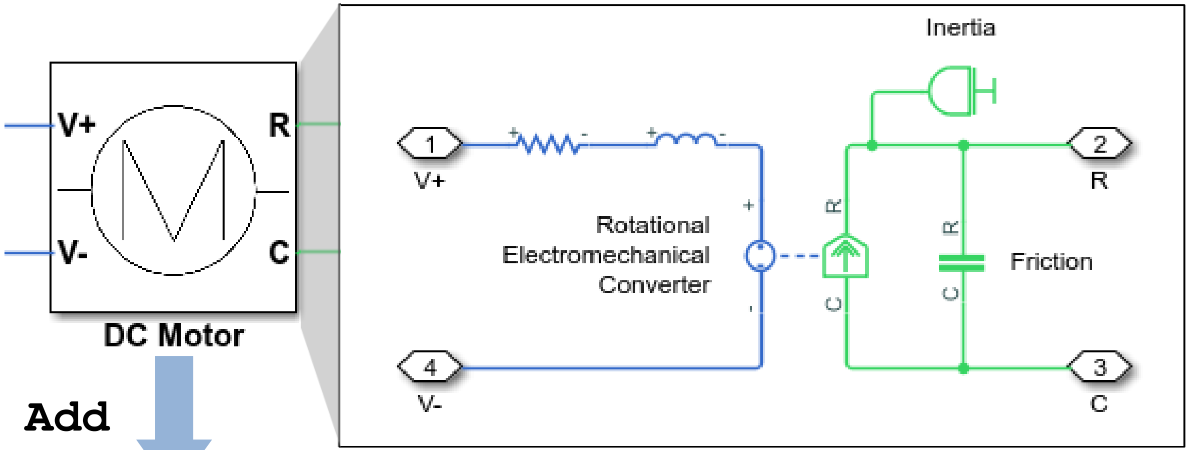
- Non-requirements
  - Transients and electrical noise due to brushing

- **Solution**
  1. Convert DC Motor subsystem to a Simscape component
  2. Modify the component to be faultable

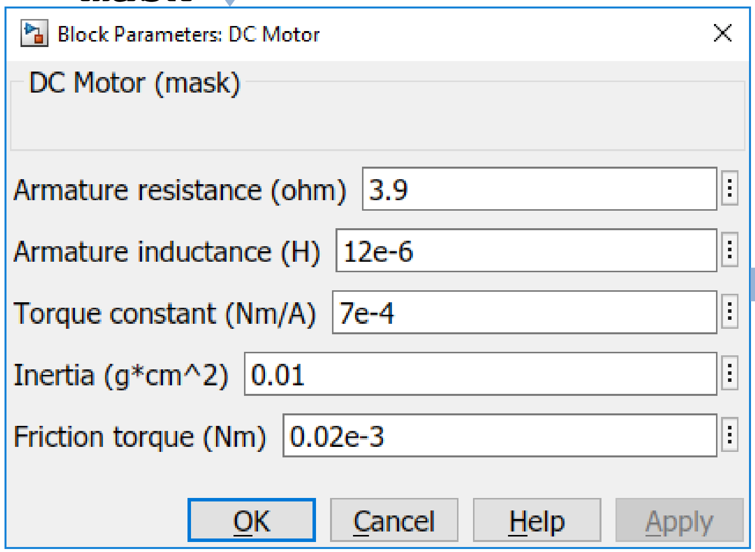


Convert subsystem to Simscape component

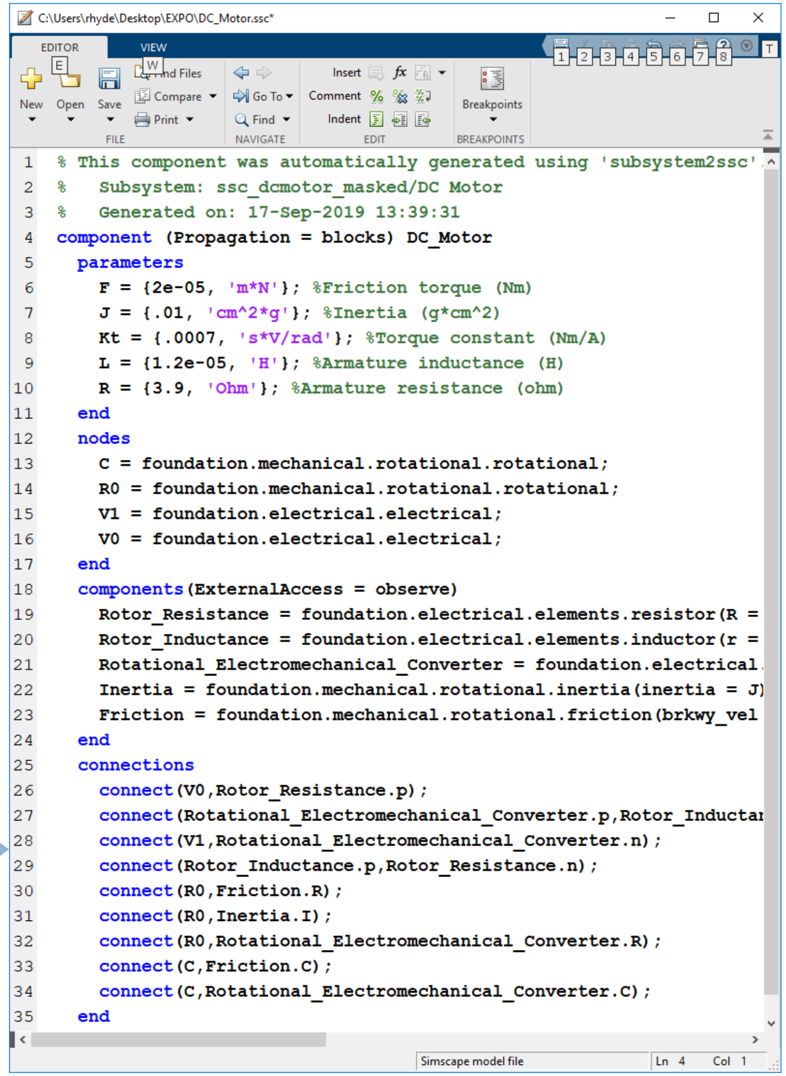
`subsystem2ssc` Function



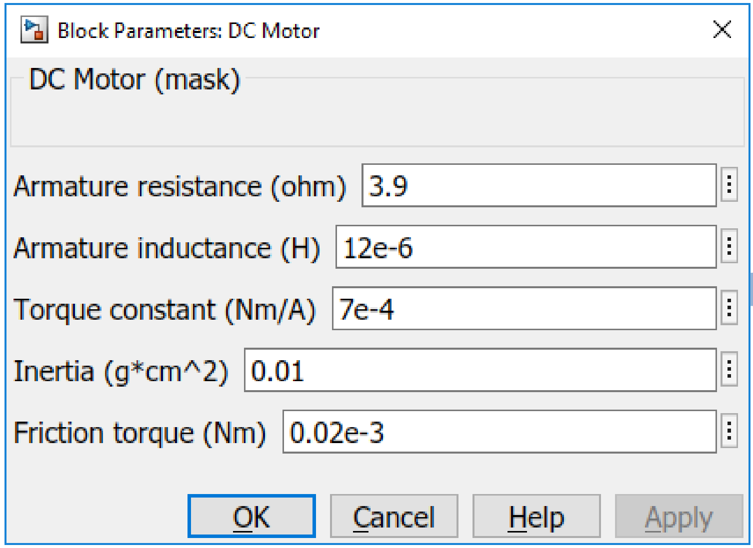
Add mask



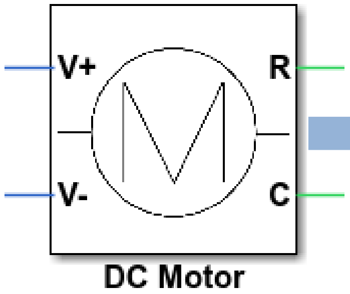
`subsystem2ssc(gcb)`



Convert subsystem to Simscape component

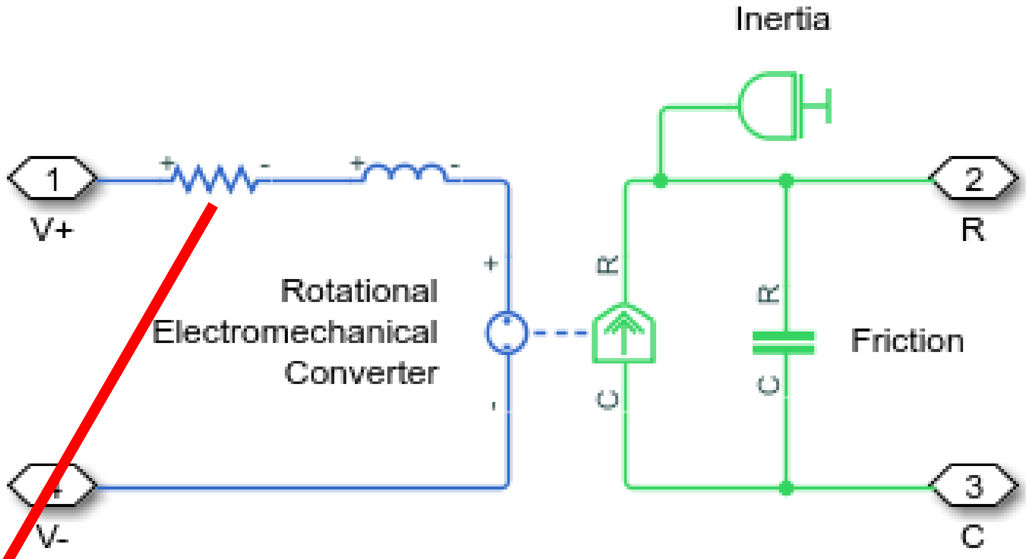



```
parameters
  F = {2e-05, 'm*N'}; %Friction torque (Nm)
  J = {.01, 'cm^2*g'}; %Inertia (g*cm^2)
  Kt = {.0007, 's*V/rad'}; %Torque constant (Nm/A)
  L = {1.2e-05, 'H'}; %Armature inductance (H)
  R = {3.9, 'Ohm'}; %Armature resistance (ohm)
end
```



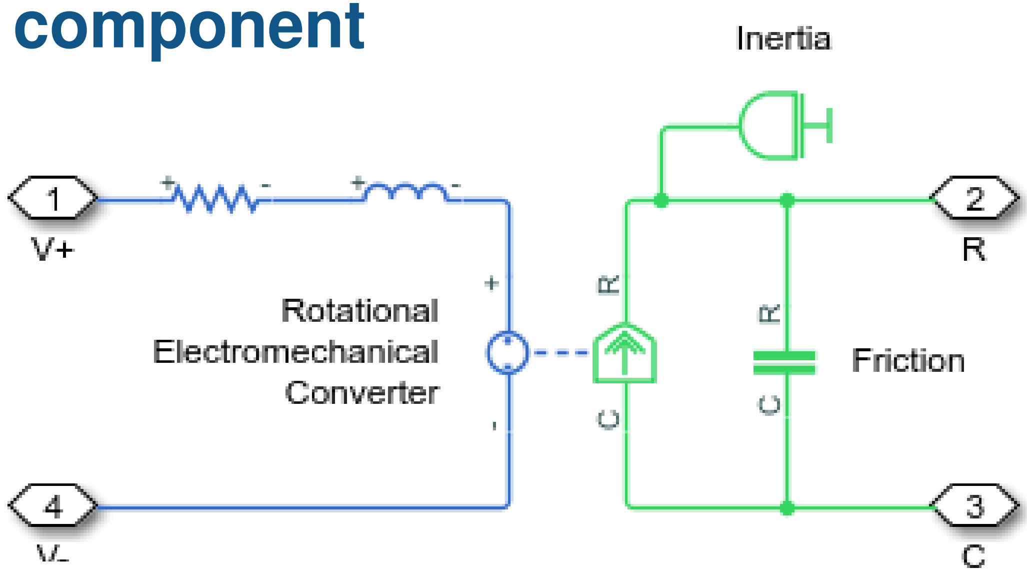

```
nodes
  C = foundation.mechanical.rotational.rotational;
  R0 = foundation.mechanical.rotational.rotational;
  V1 = foundation.electrical.electrical;
  V0 = foundation.electrical.electrical;
end
```

Convert subsystem to Simscape component




```
components(ExternalAccess = observe)
  Rotor_Resistance = foundation.electrical.elements.resistor(R = R);
  Rotor_Inductance = foundation.electrical.elements.inductor(l = L);
  Rotational_Electromechanical_Converter = foundation.electrical.elements.rotational_converter(K = Kt);
  Inertia = foundation.mechanical.rotational.inertia(inertia = J);
  Friction = foundation.mechanical.rotational.friction(brkwy_trq = F, Col_trq = F);
end
```

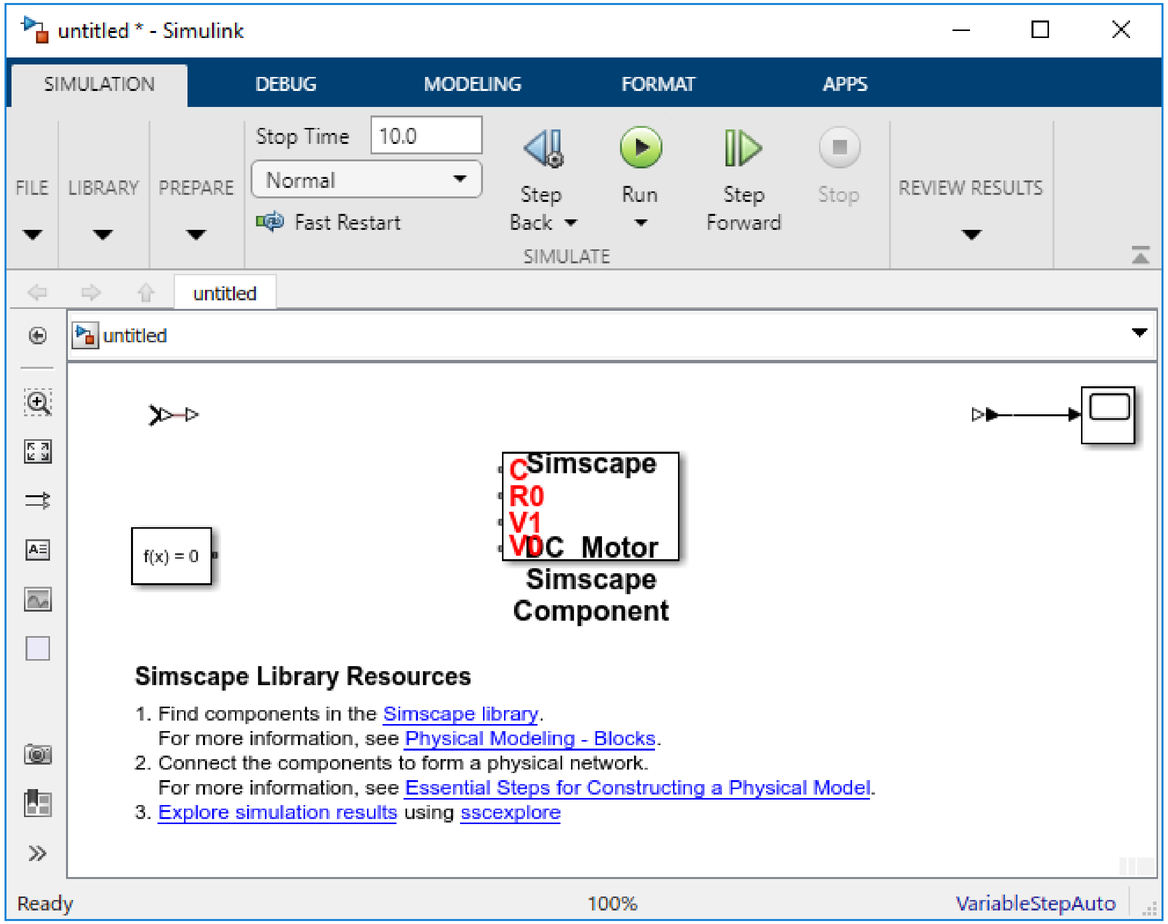
Convert subsystem to Simscape component

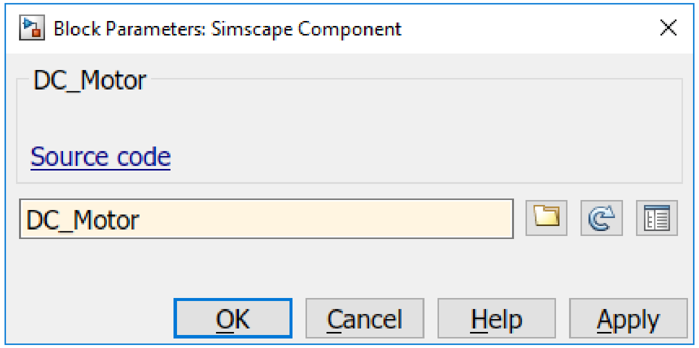


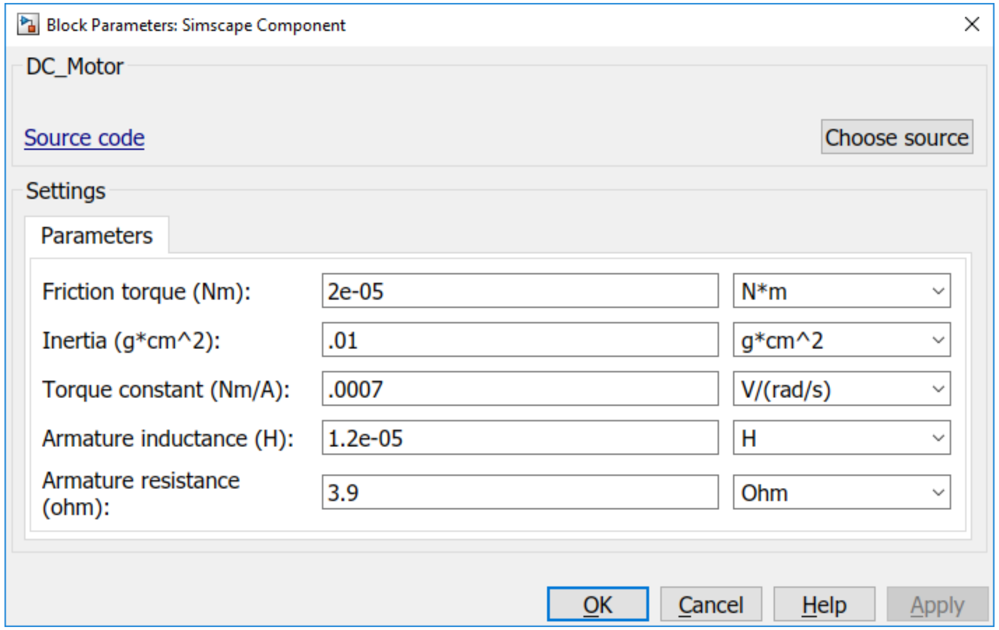

```
connections
  connect(V0,Rotor_Resistance.p);
  connect(Rotational_Electromechanical_Converter.p,Rotor_Inductance.n);
  connect(V1,Rotational_Electromechanical_Converter.n);
  connect(Rotor_Inductance.p,Rotor_Resistance.n);
  connect(R0,Friction.R);
  connect(R0,Inertia.I);
  connect(R0,Rotational_Electromechanical_Converter.R);
  connect(C,Friction.C);
  connect(C,Rotational_Electromechanical_Converter.C);
end
```

Convert subsystem to Simscape component

Use Simscape Component block to instantiate



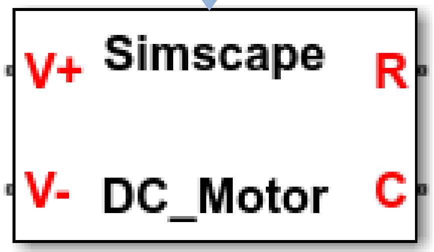




Convert subsystem to Simscape component

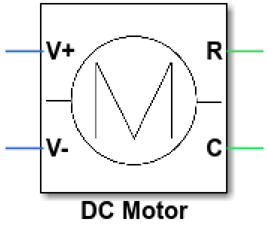
Label and order the ports

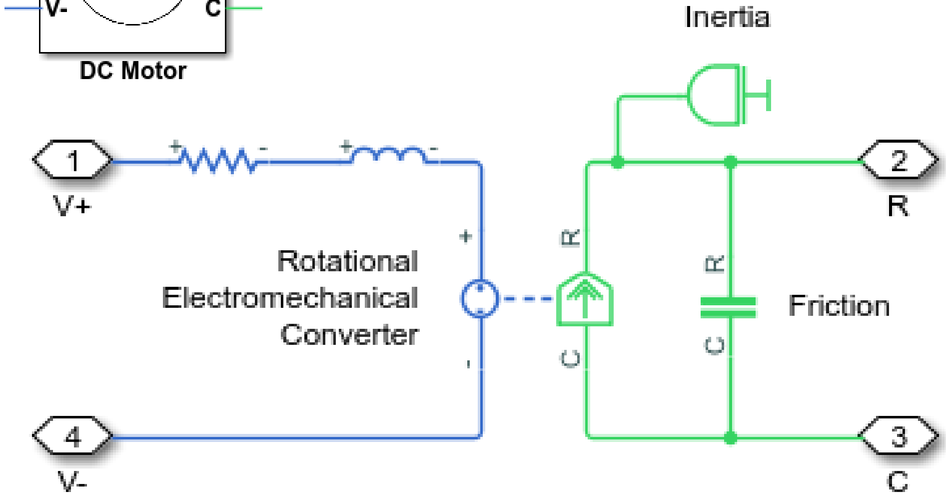
```
nodes
  R0 = foundation.mechanical.rotational.rotational; % R:right
  C = foundation.mechanical.rotational.rotational; % C:right
  V0 = foundation.electrical.electrical; % V+:left
  V1 = foundation.electrical.electrical; % V-:left
end
```



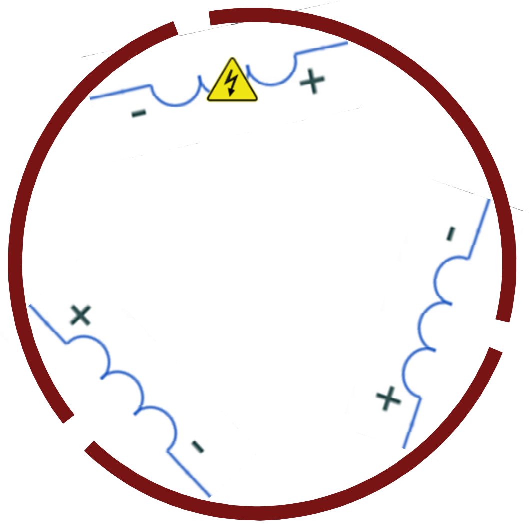
Simscape Component

Faultable DC Motor

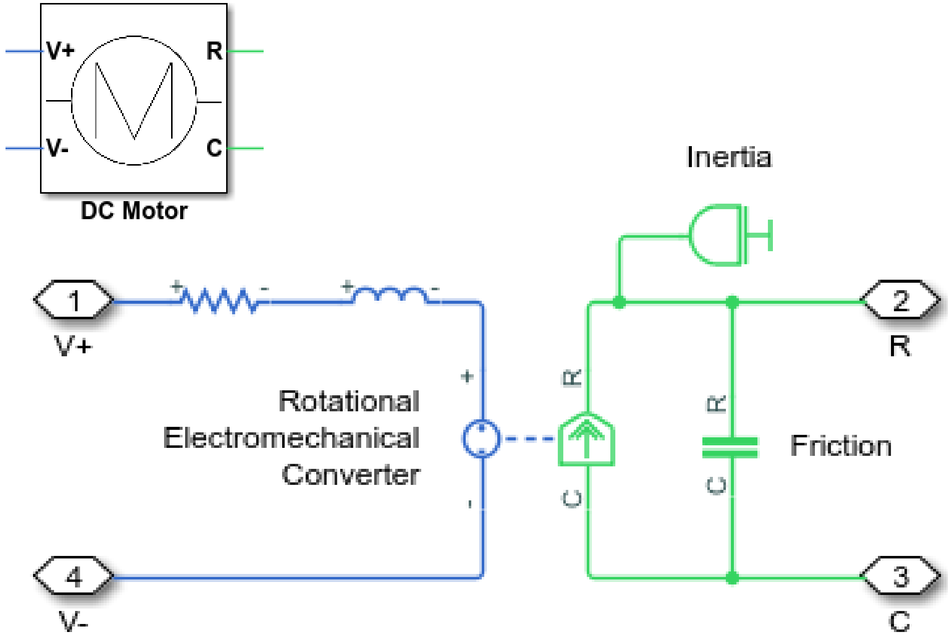




- Requirements
  - Normal behaviour as example ssc_dcmotor
  - Faulted behaviour: one winding open-circuit
- Non-requirements
  - Transients and electrical noise due to brushing
- **Solution**
  1. Convert DC Motor subsystem to a Simscape component
  2. Modify the component to be faultable



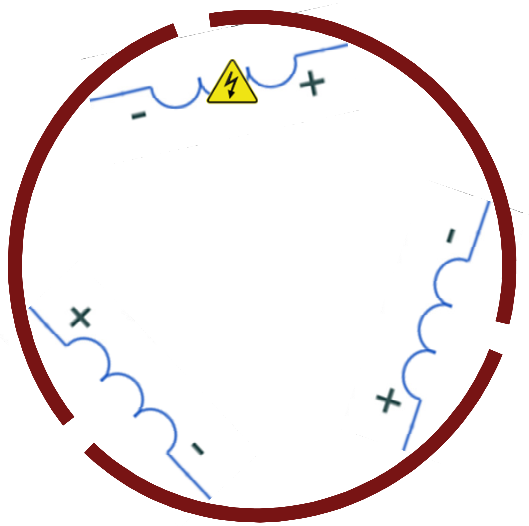
Faultable DC Motor



- Requirements
  - Normal behaviour as example ssc_dcmotor
  - Faulted behaviour: one winding open-circuit

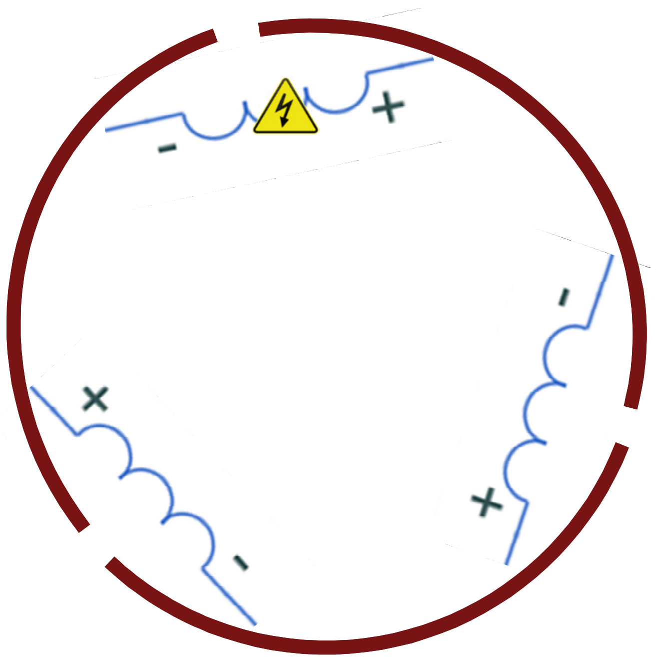
- Non-requirements
  - Transients and electrical noise due to brushing

- **Solution**
  1. Convert DC Motor subsystem to a Simscape component
  2. Modify the component to be faultable



Modify the custom component to be faultable

Requirements & functional design



- Fault behaviour
  - Path seen by brushes goes open circuit for 120 degrees of rotation

- Implementation
  - Represent this with a custom resistor with resistance dependent on rotor angle

Custom resistor

R2019b

```
>> sscnewfile('FaultableResistor'
```

C:\Users\rhyde\Rick\FaultableResistor.ssc*

```
component FaultableResistor < foundation.electrica
% Resistor
% The voltage-current (V-I) relationship for a li
% where R is the constant resistance in ohms.

% Copyright 2005-2016 The MathWorks, Inc.

parameters
    R = { 1, 'Ohm' }; % Resistance
end

equations
    assert(R>=0);
    v == R*i;
end

end
```

Simscape model file Ln 5 Col 1

C:\Program Files\MATLAB\R2019b\toolbox\physmod\simscape\library\m\+foundation\+electric...

```
component(Hidden=true) branch
% Electrical Branch
% Copyright 2005-2013 The MathWorks, Inc.
    nodes
        p = foundation.electrical.electrical; % +:left
        n = foundation.electrical.electrical; % -:right
    end
    variables
        i = { 0, 'A' }; % Current
        v = { 0, 'V' }; % Voltage
    end
    branches
        i : p.i -> n.i;
    end
    equations
        v == p.v - n.v;
    end
end
```

Simscape model file Ln 12 Col 13

Custom resistor

C:\Users\rhyde\Rick\FaultableResistor.ssc*

```
component FaultableResistor < foundation.electrical.branch
% Resistor
% The voltage-current (V-I) relationship for a linear resistor is V=I*R,
% where R is the constant resistance in ohms.

% Copyright 2005-2016 The MathWorks, Inc.

parameters
    R = { 1, 'Ohm' }; % Resistance
end

equations
    assert(R>=0);
    v == R*i;
end

end
```

- To do:
  1. Add a variable for rotor angle.
  2. Add a parameter for faulted resistance.
  3. Make resistance a function of rotor angle.
  4. Add logic to enable the fault.

Custom resistor

```
variables
    rotor_angle = {0, 'deg'}; % Rotor angle
end
```

- To do:
  1. Add a variable for rotor angle.
  2. Add a parameter for faulted resistance.
  3. Make resistance a function of rotor angle.
  4. Add logic to enable the fault.

Custom resistor

```
parameters
    R = { 1, 'Ohm' }; % Resistance
    Rfaulted = { 1, 'MOhm' }; % Faulted resistance
end
```

- To do:
  1. Add a variable for rotor angle.
  2. Add a parameter for faulted resistance.
  3. Make resistance a function of rotor angle.
  4. Add logic to enable the fault.

MathWorks

Custom resistor

```
equations
    let
        open_circuit = if gt(sin(rotor_angle),sin(pi/6)), 1 else 0 end;
    in
        v == (1-open_circuit)*R*i +open_circuit*Rfaulted*i;
    end
end
```

open_circuit is zero for 1/3 of a revolution, otherwise one.

Switch between armature resistance and faulted resistance values

Advanced: note use of "gt" rather than ">" in conditional.

This takes out zero crossings which would cause numeric issues around zero speed.

- To do:
  1. Add a variable for rotor angle.
  2. Add a parameter for faulted resistance.
  3. Make resistance a function of rotor angle.
  4. Add logic to enable the fault.

Custom resistor

```
parameters
    R = { 1, 'Ohm' }; % Resistance
    Rfaulted = { 1, 'MOhm' }; % Faulted resistance
    tFault = { inf, 's' }; % Time at which to fault
end
```

```
    if time<tFault
        v == R*i;
    else
        let
            open_circuit = if gt(sin(rotor_angle),sin(pi/6)), 1 else 0 end;
        in
            v == (1-open_circuit)*R*i + open_circuit*Rfaulted*i;
        end
    end
```

4. Add logic to enable the fault.

Custom resistor

```
component FaultableResistor < foundation.electrical.branch
% Resistor
% The voltage-current (V-I) relationship for a linear resistor is V=I*R,
% where R is the constant resistance in ohms.

% Copyright 2005-2016 The MathWorks, Inc.

parameters
    R = { 1, 'Ohm' }; % Resistance
    Rfaulted = { 1, 'MOhm' }; % Faulted resistance
    tFault = { inf, 's' }; % Time at which to fault
end

variables
    rotor_angle = {0, 'deg'}; % Rotor angle
end

equations
    assert(R>=0);
    if time<tFault
        v == R*i;
    else
        let
            open_circuit = if gt(sin(rotor_angle),sin(pi/6)), 1 else 0 end;
        in
            v == (1-open_circuit)*R*i + open_circuit*Rfaulted*i;
        end
    end
end

end
```

Integrate custom resistor into custom DC motor

```
components(ExternalAccess = observe)
  Rotor_Resistance = foundation.electrical.elements.resistor(R = R);
```

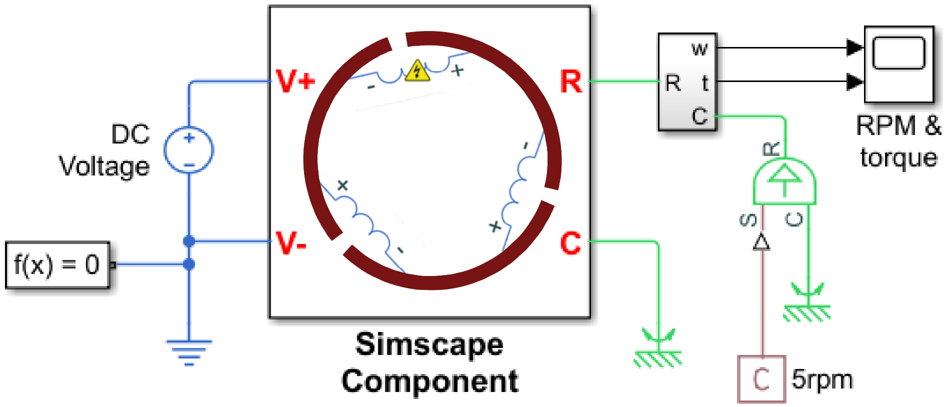
```
components(ExternalAccess = observe)
  Rotor_Resistance = FaultableResistor(R = R, Rfaulted=Rfaulted, tFault=tFault);
```

```
parameters
  F = {2e-05, 'm*N'}; %Friction torque (Nm)
  J = {.01, 'cm^2*g'}; %Inertia (g*cm^2)
  Kt = {.0007, 's*V/rad'}; %Torque constant (Nm/A)
  L = {1.2e-05, 'H'}; %Armature inductance (H)
  R = {3.9, 'Ohm'}; %Armature resistance (ohm)
  Rfaulted = { 1, 'MOhm' }; % Faulted resistance
  tFault = { inf, 's' }; % Time at which to fault
end
```

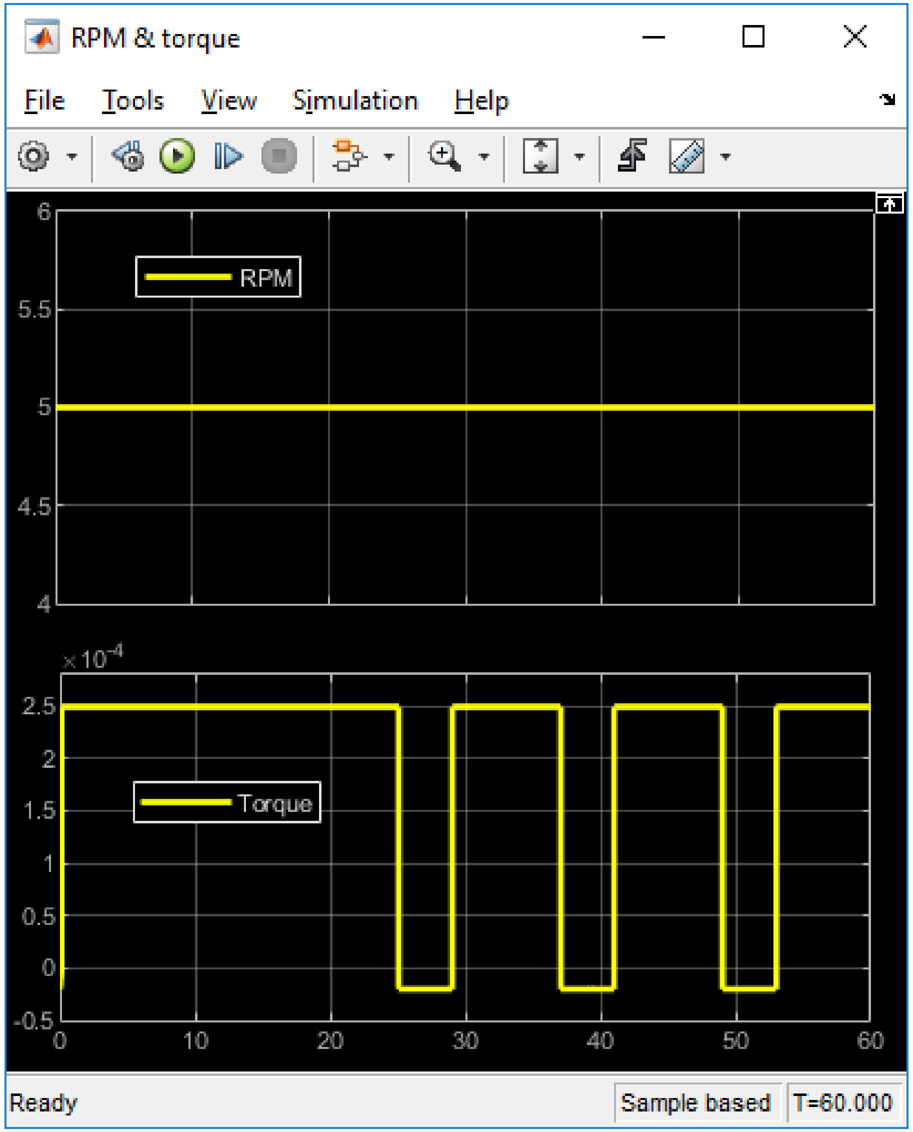
```
equations
    Rotor_Resistance.rotor_angle.der == Inertia.w;
end
```

- To do:
  1. Replace standard resistor with the faultable resistor.
  2. Pass faultable resistor parameters to DC motor interface.
  3. Add an equation that defines rotor angle.
  4. Create a test harness and validate.

Faultable DC Motor – Test Harness



Test harness for faultable behaviour



- Test:
  - Drive motor at fixed 5rpm
  - Measure torque

Summary

- Matching models to engineering design tasks

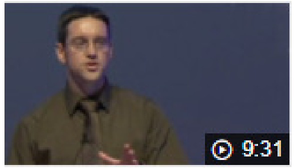
- Examples
  - Hybrid vehicle powertrain - *with focus on need for multiple models*
  - Photovoltaic system - *with focus on plant design and control*

- Tutorial – faulted DC motor

How to find out more

- MathWorks physical modelling page:
  - https://www.mathworks.com/solutions/physical-modeling.html

- Steve Miller's introduction video
  - https://www.mathworks.com/videos/physical-modeling-introduction-75883.html



- MATLAB Central File Exchange
  - https://www.mathworks.com/matlabcentral/fileexchange/

- Hybrid vehicle example:
  - https://uk.mathworks.com/matlabcentral/fileexchange/28441-hybrid-electric-vehicle-model-in-simulink