

ME40064: System Modelling & Simulation

ME50344: Engineering Systems Simulation

Lecture 17

Dr Andrew Cookson
University of Bath, October 2018

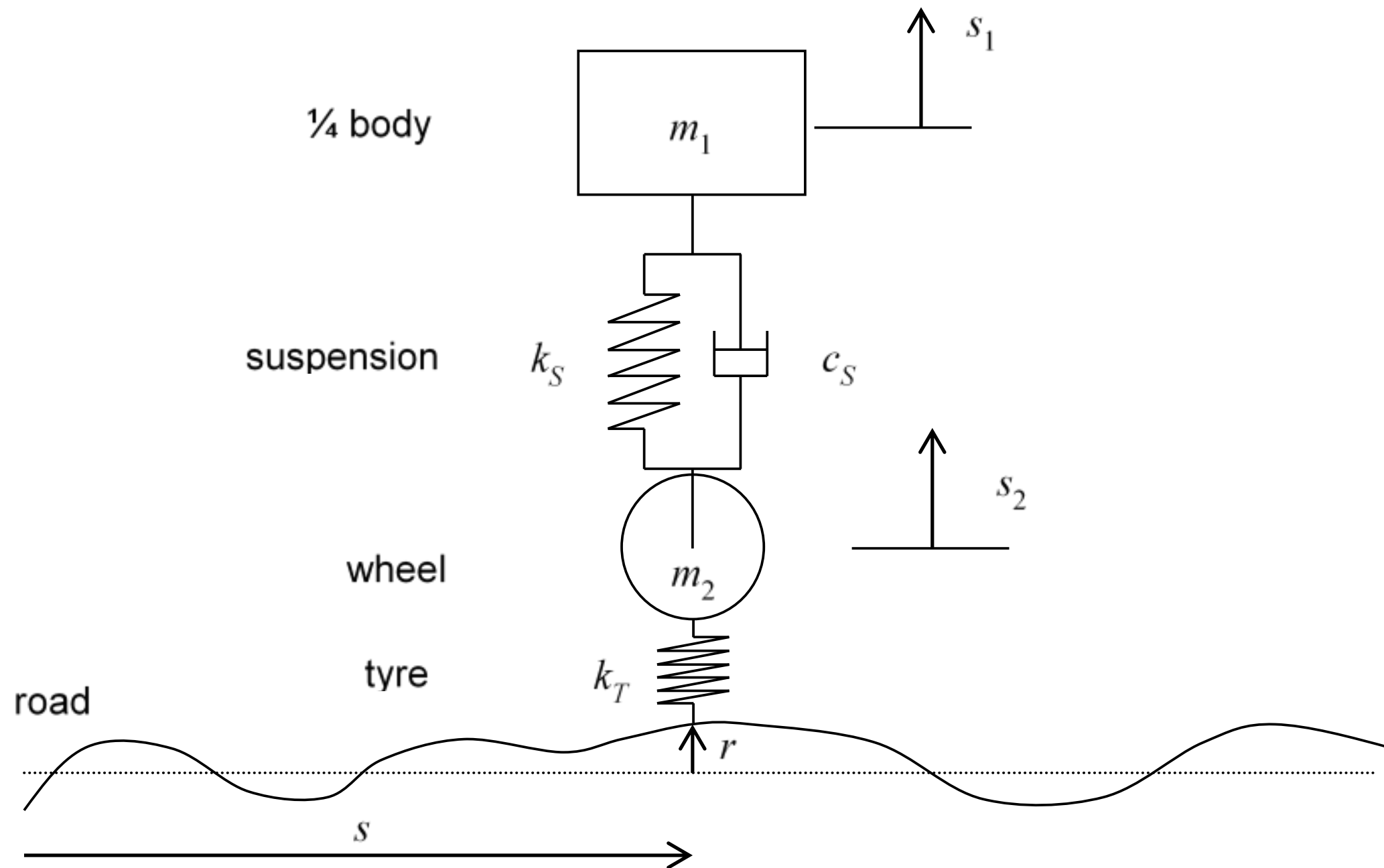
LECTURE 17

Verification Of Simulink Models

- Introduce 1/4 car model
- Reiterate importance of verification
- Understand how to use theoretical methods for verification
- Appreciate advanced tools for verification provided by Simulink

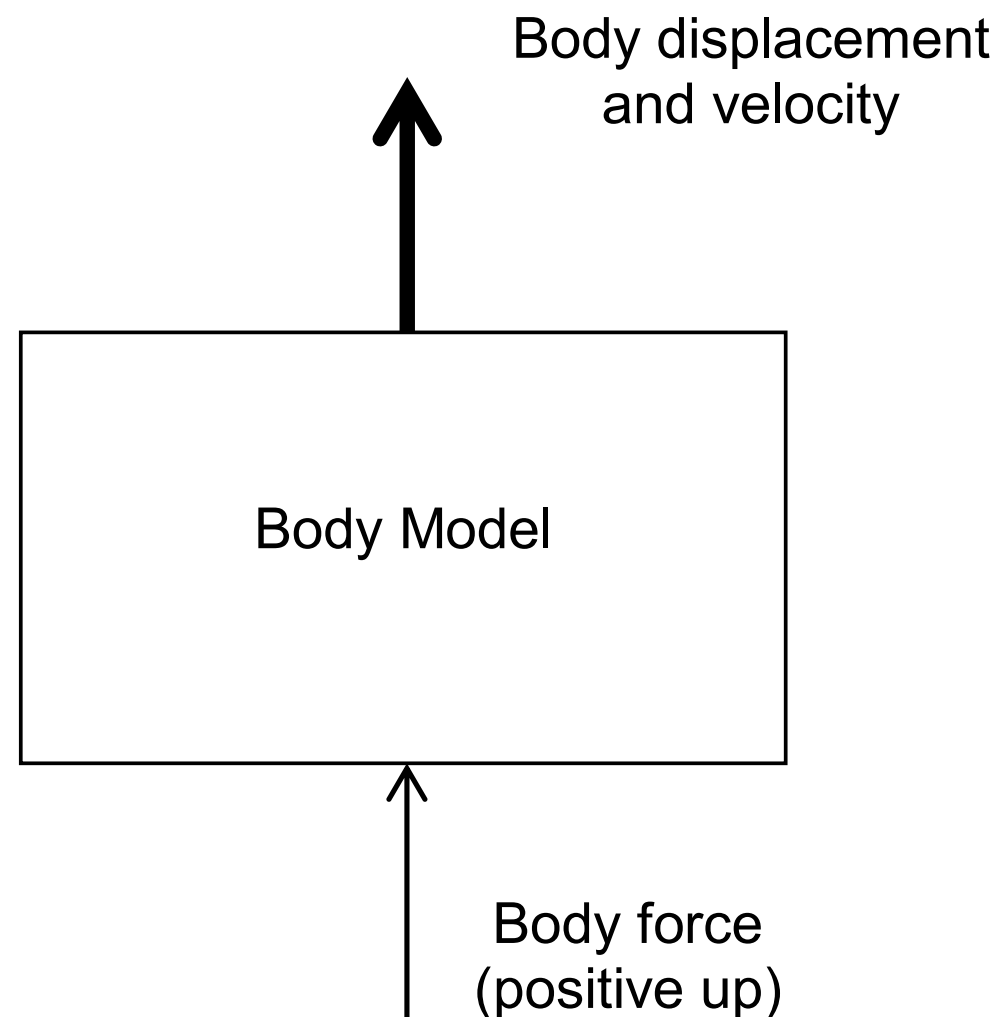
THE 1/4 CAR MODEL

The System



THE 1/4 CAR MODEL

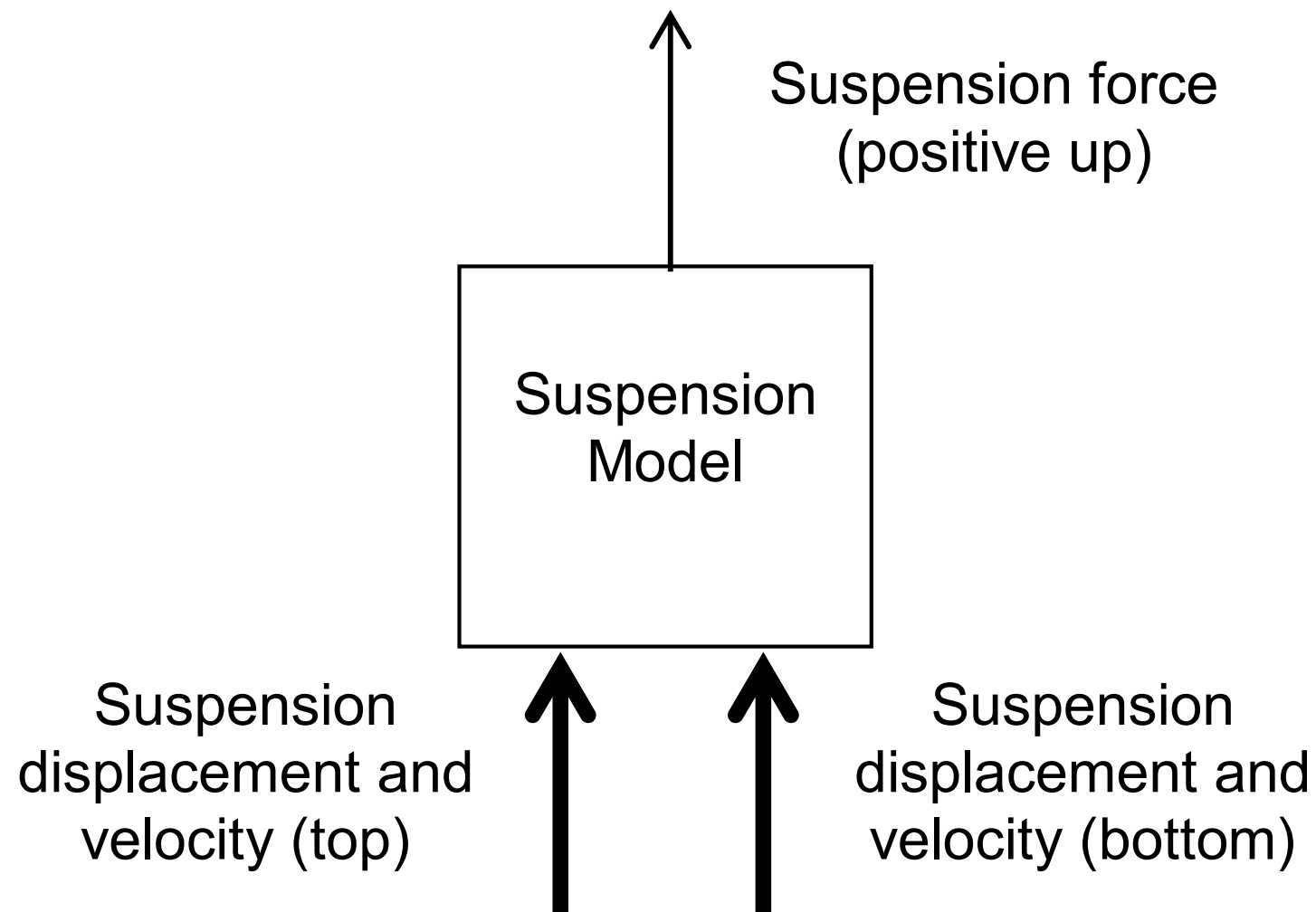
Body Model Subsystem



$$m_1 \ddot{s}_1 = F - m_1 g$$

THE 1/4 CAR MODEL

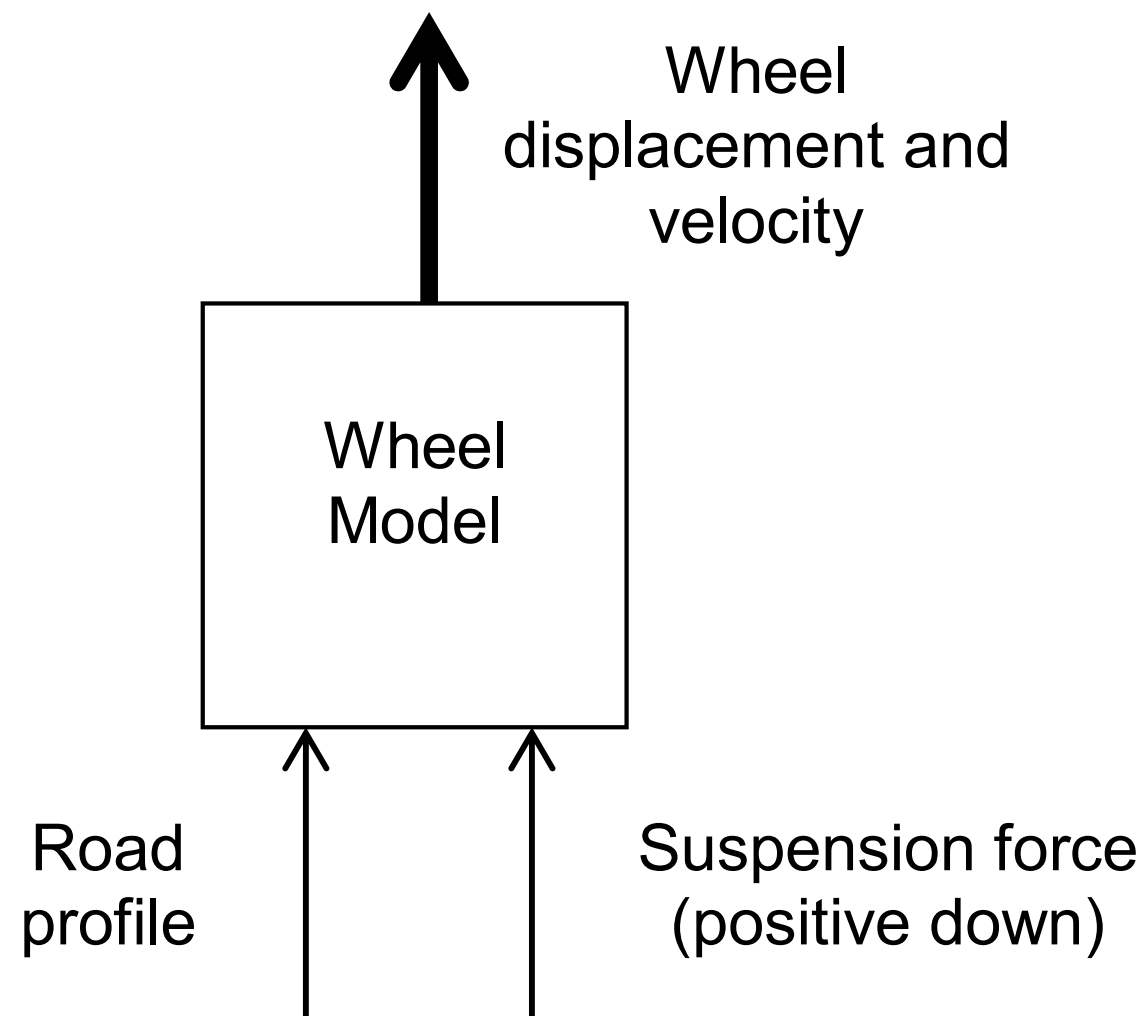
Suspension Model Subsystem



$$F = -k_s (s_1 - s_2) - c_s (\dot{s}_1 - \dot{s}_2)$$

THE 1/4 CAR MODEL

Tyre Model Subsystem



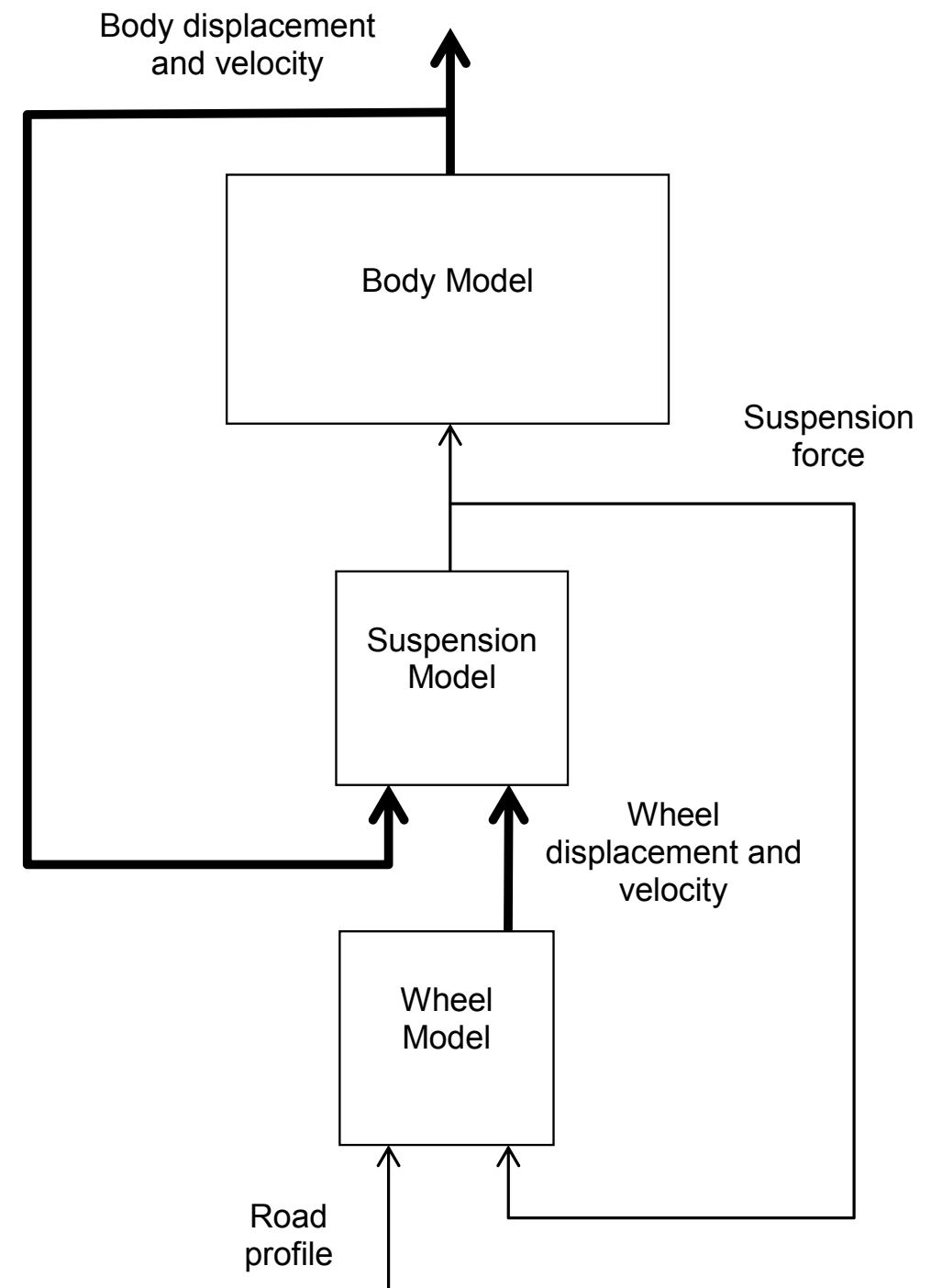
$$m_2 \ddot{s}_2 = -F + k_T (r - s_2) - m_2 g$$

THE 1/4 CAR MODEL

Coupled Together For Whole System

Coupled 1/4 Car Model System

- Note the series and feedback
- The model layout relates to the “physical” model
- As presented, the model is linear
- Could be modified to have nonlinear elements
 - Different bump/rebound rates in the suspension
 - Tyre forces can be due to compression only



SIMULINK VERIFICATION

The Challenge

As you can see the GOOD thing about Simulink is how easy its graphical method makes it to build complex models

SIMULINK VERIFICATION

The Challenge

As you can see the GOOD thing about Simulink is how easy its graphical method makes it to build complex models

However...

SIMULINK VERIFICATION

The Challenge

As you can see the GOOD thing about Simulink is how easy its graphical method makes it to build complex models

However...

The BAD thing about Simulink is how easy its graphical method makes it to build complex models

SIMULINK VERIFICATION

The Aims

Verify that subsystem models and the complete model deliver the correct solutions. As far as possible, test them to show that:

- Solutions are accurate to a desired level. This will depend on the particular time-stepping routine used and associated specifications of relative and absolute tolerances
- All mathematical and logic errors have been eliminated. This is possible by thorough checking of code and assessing the model predictions arising from particular forms of all inputs

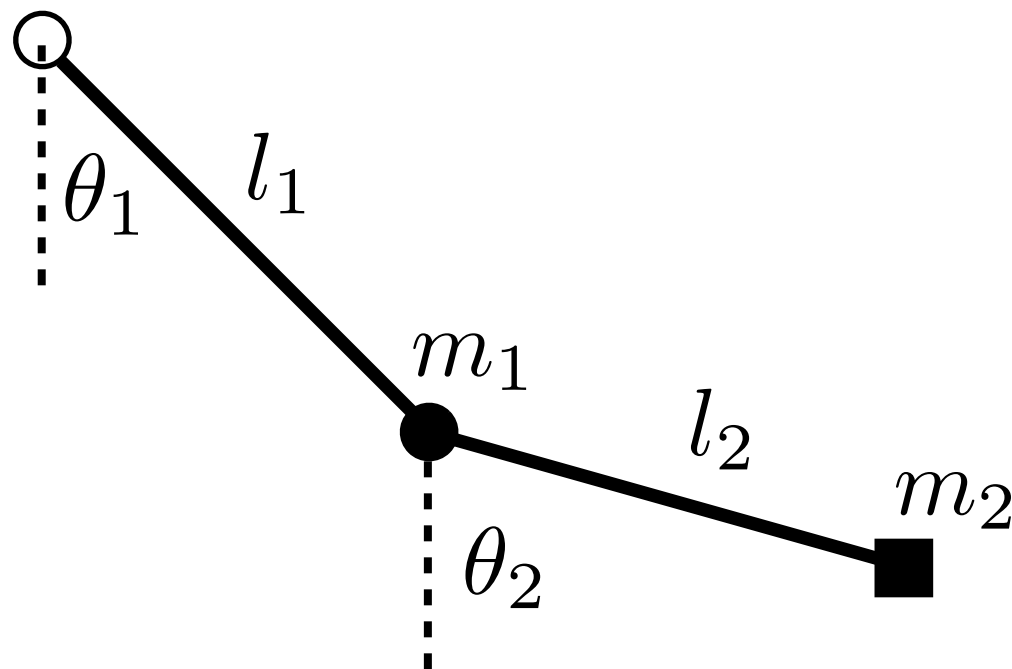
SIMULINK VERIFICATION

What Can We Do?

1. Checking against conservation laws
2. Running simple parameter cases where behaviour can be predicted from simple calculations or analytical solutions
3. Simulink's testing tools

SIMULINK VERIFICATION

Conservation Of Energy Example



Potential Energy + Kinetic Energy = Constant

$$m_1gy_1 + m_2gy_2 + 1/2m_1v_1^2 + 1/2m_2v_2^2 = Constant$$

SIMULINK VERIFICATION

Using 1/4 Car As An Example

Default data:

$$m_1 = 250 \text{ kg}, m_2 = 20 \text{ kg}, k_s = 2 \times 10^4 \text{ N/m}, c_s = 1000 \text{ Ns/m}, k_T = 14 \times 10^4 \text{ N/m}$$

As part of the verification process, we are at liberty to change these parameters for particular tests

We have shown how to write the system in terms of three sub-models

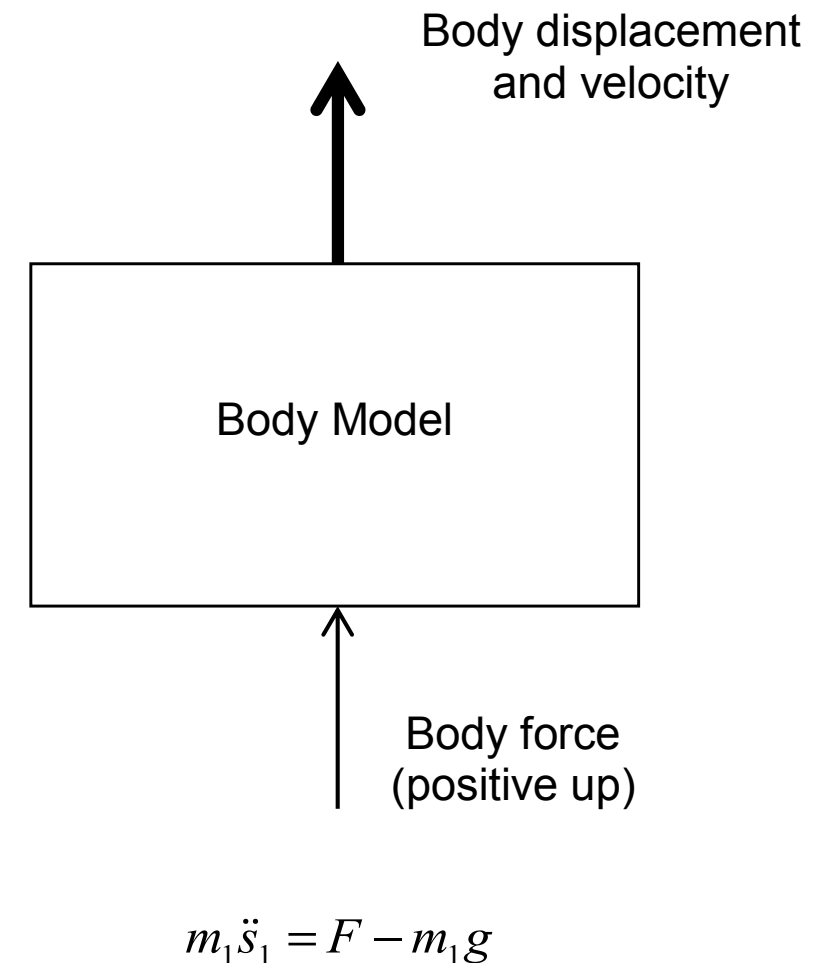
- can test each sub-model individually

Assume linear components

SIMULINK VERIFICATION

Car Body Subsystem

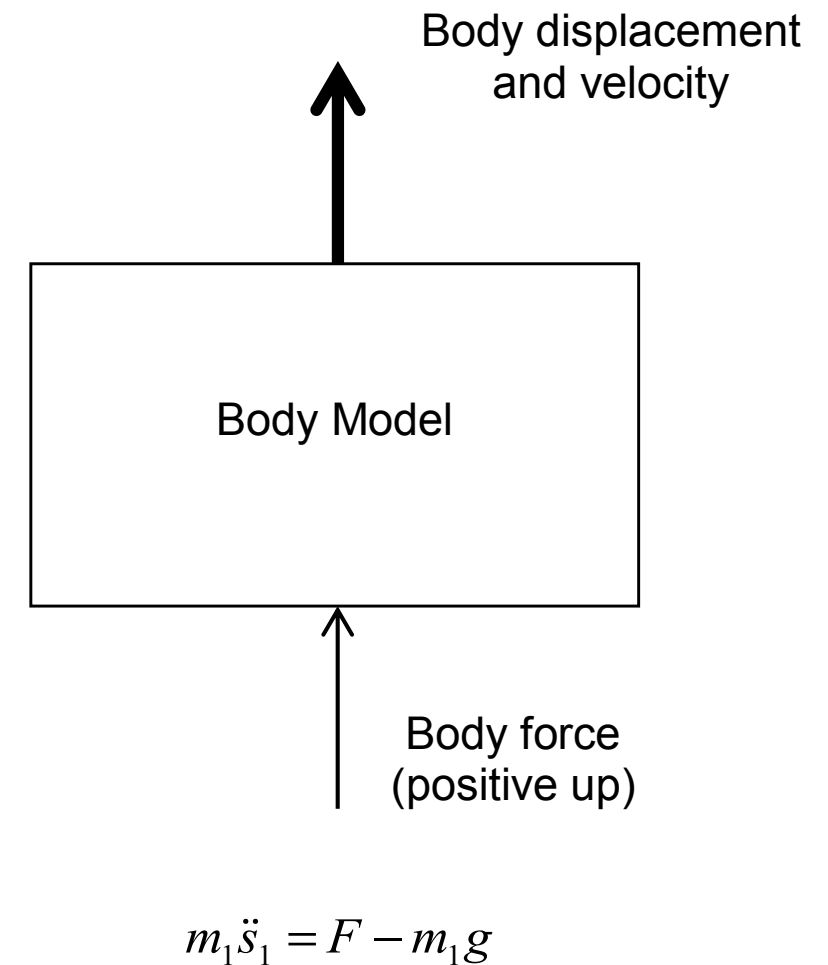
I. Apply zero force as input with zero initial conditions. The expected output is that the body will fall under gravity with a negative parabolic trajectory for body displacement and a corresponding linearly decreasing negative velocity



SIMULINK VERIFICATION

Car Body Subsystem

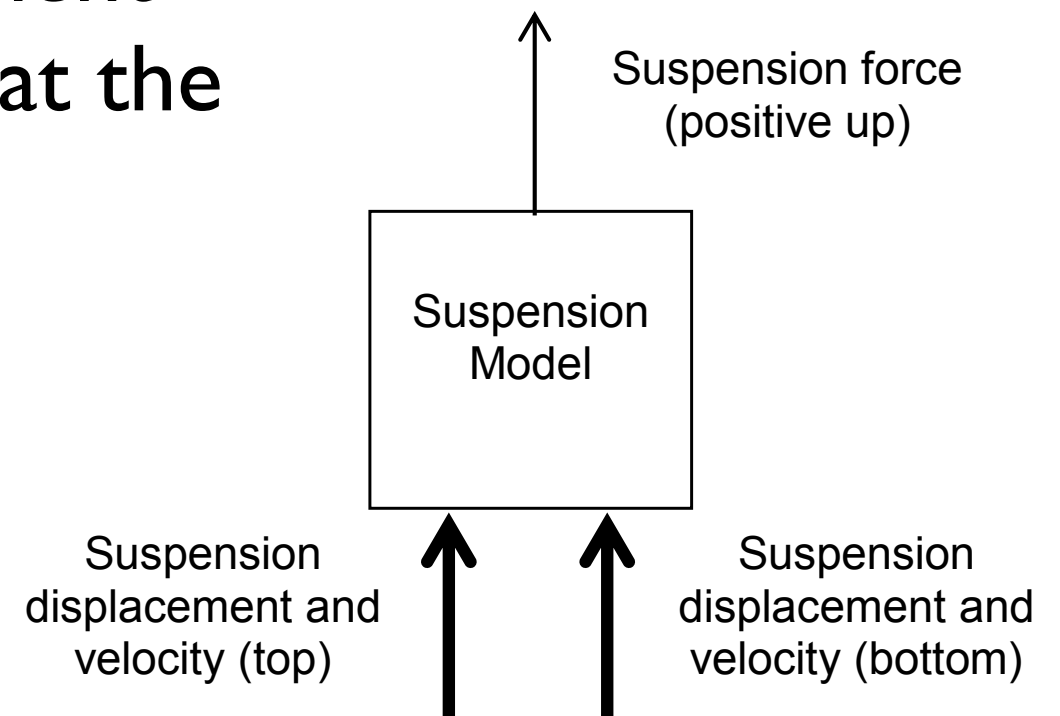
1. Apply zero force as input with zero initial conditions. The expected output is that the body will fall under gravity with a negative parabolic trajectory for body displacement and a corresponding linearly decreasing negative velocity
2. With zero initial conditions, apply a force input that is equal and opposite to the body weight. The expected output is that both displacement and velocity should be zero



SIMULINK VERIFICATION

Suspension Subsystem

I. Apply unit values to each displacement and velocity input in turn, checking that the output force values are as expected

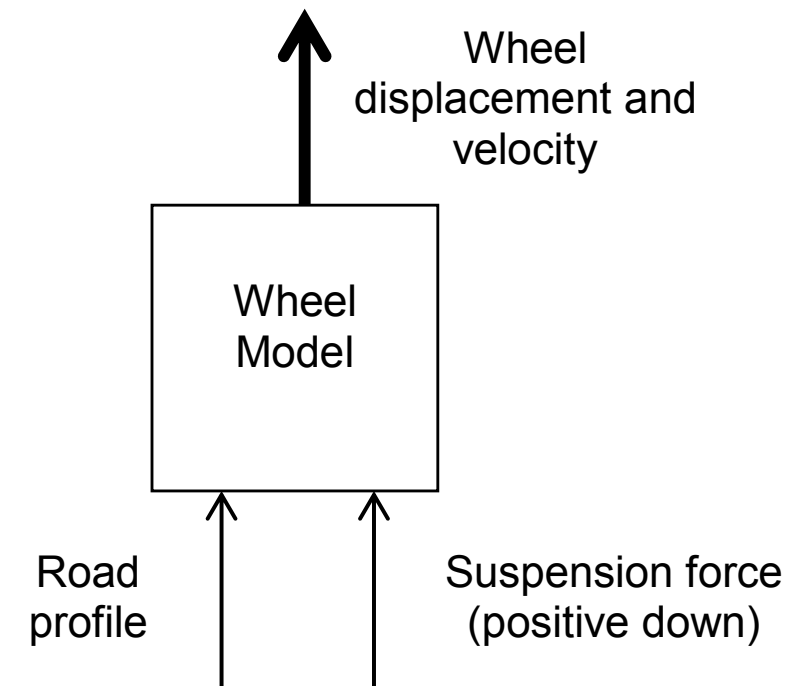


$$F = -k_s(s_1 - s_2) - c_s(\dot{s}_1 - \dot{s}_2)$$

SIMULINK VERIFICATION

Wheel Subsystem

I. With $k_T = 0$, similar tests as for the car body subsystem could be applied

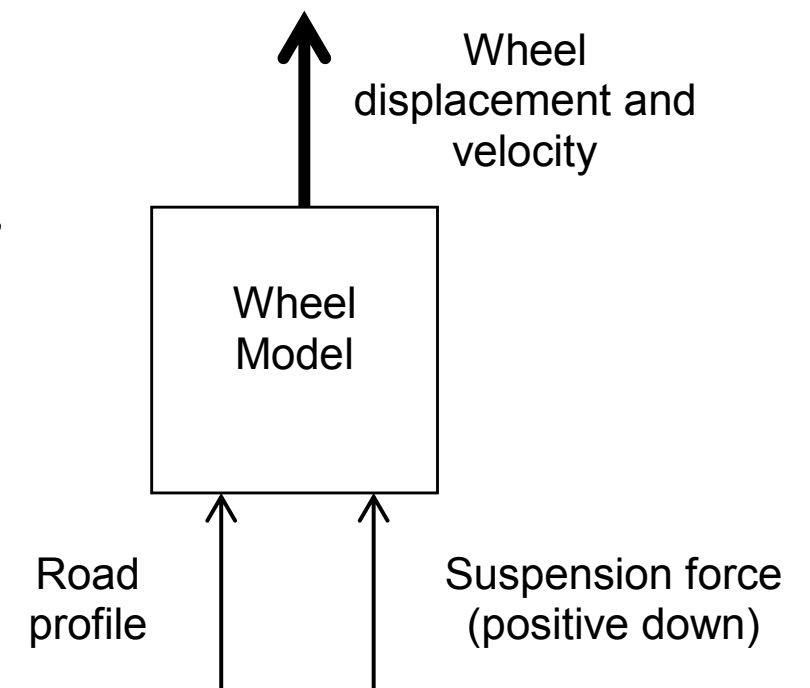


$$m_2 \ddot{s}_2 = -F + k_T(r - s_2) - m_2 g$$

SIMULINK VERIFICATION

Wheel Subsystem

1. With $k_T = 0$, similar tests as for the car body subsystem could be applied
2. With $k_T \neq 0$, $F = 0$, $g = 0$, a step input in r from 0 to 1 should result in oscillations at an angular frequency $\sqrt{k_T / m_2}$, the mean value of s_2 being 1 and the mean value of \dot{s}_2 being 0



$$m_2 \ddot{s}_2 = -F + k_T (r - s_2) - m_2 g$$

Note: ordinary frequency $f = \frac{\omega}{2\pi}$

SIMULINK VERIFICATION

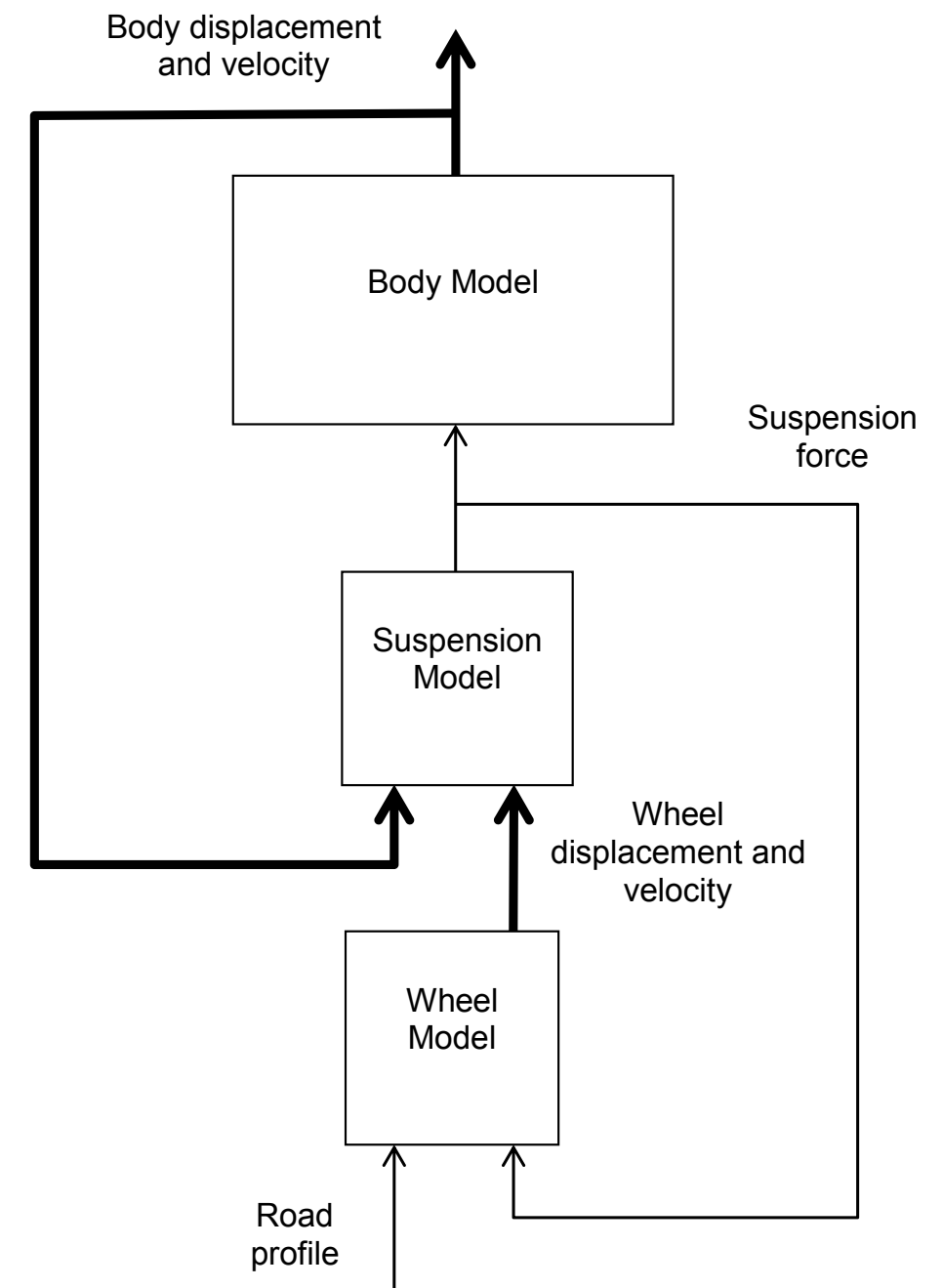
Complete 1/4 Car Model

1. With $r=0$, the wheel should settle to a displacement at which the tyre supports the whole vehicle weight:

$$s_2 = -(m_1 + m_2)g / k_T$$

2. The suspension spring should support the car body weight:

$$s_1 = -m_1 g / k_s + s_2$$



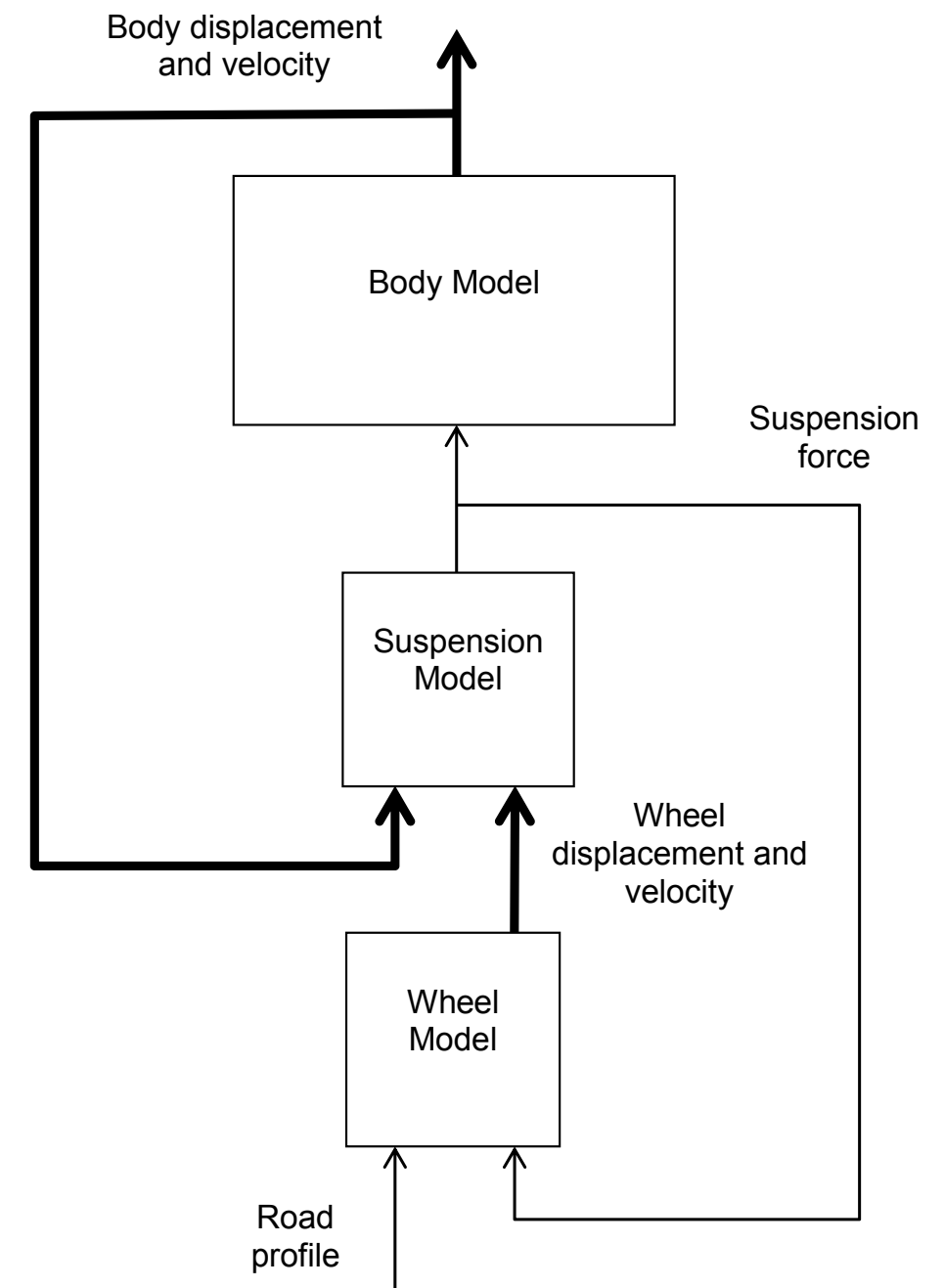
SIMULINK VERIFICATION

Complete 1/4 Car Model

1. Set k_s or c_s to be large values such that the body and wheel masses are effectively a single mass.

Under a step input in r from 0 to 1, both masses should oscillate with a peak amplitude of 1 at the natural frequency:

$$\sqrt{k_T / (m_1 + m_2)}$$



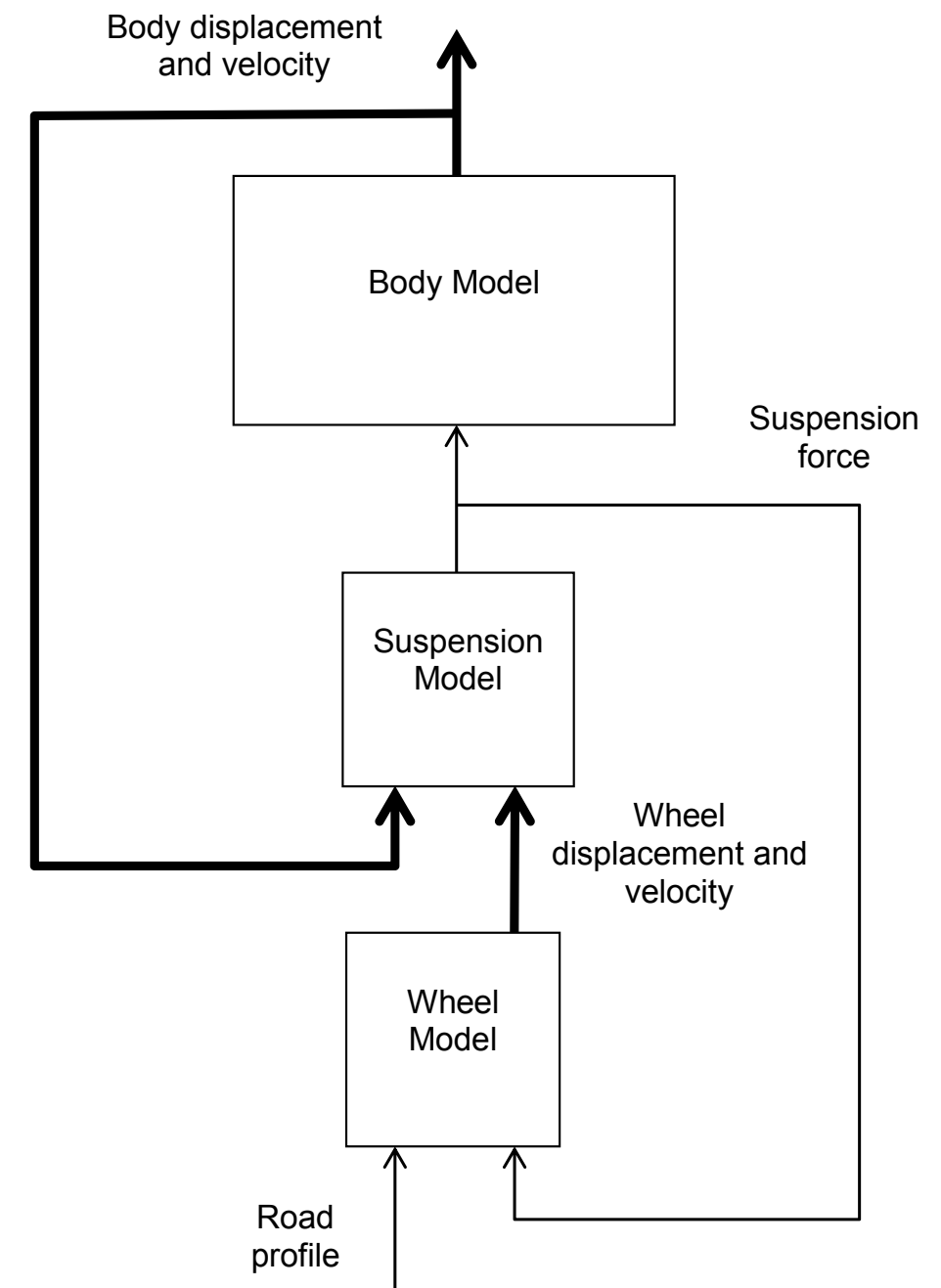
SIMULINK VERIFICATION

Complete 1/4 Car Model

I. Set k_T to be an abnormally high value, m_2 to be very small (non-zero). Under a step input in r from 0 to 1, the body mass should oscillate at the natural frequency $\sqrt{k_S / m_1}$ (if $c_S = 0$) with a peak amplitude of 1.

If a suspension damper has a non-zero rate, the oscillations should decay.

The wheel mass should follow closely the step input.



SIMULINK VERIFICATION

Simulink's Testing Suite

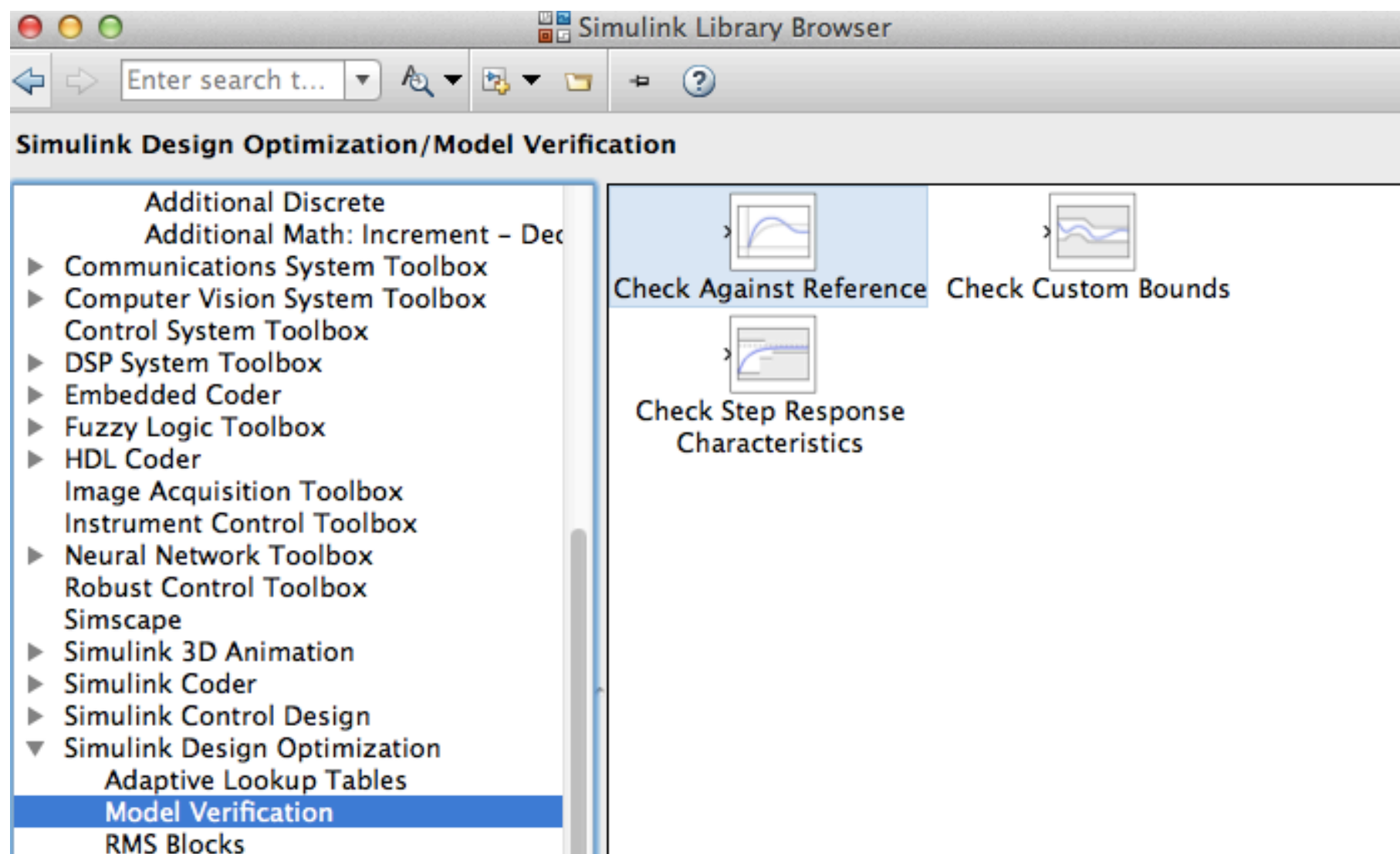
You could check these types of outputs manually or..

SIMULINK VERIFICATION

Simulink's Testing Suite

You could check these types of outputs manually or...

Simulink Design Optimization
Model Verification



SIMULINK VERIFICATION

Simulink's Testing Suite

Model Verification

Check Against Reference

<https://uk.mathworks.com/help/sldo/ref/checkagainstreference.html>

Sink Block Parameters: Check Against Reference

Check Against Reference

Assert that the input signal tracks the specified reference signal.

Bounds Assertion

☒ Include reference signal tracking in assertion

Times (seconds):

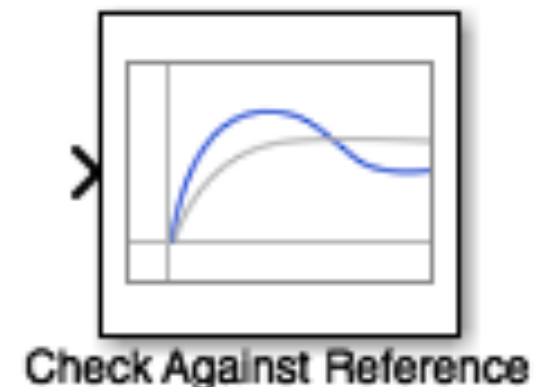
Amplitudes:

Absolute tolerance:

Relative tolerance:

Show Plot ☐ Show plot on block open Response Optimization...

? OK Cancel Help Apply



Data to compare against

Tolerance settings

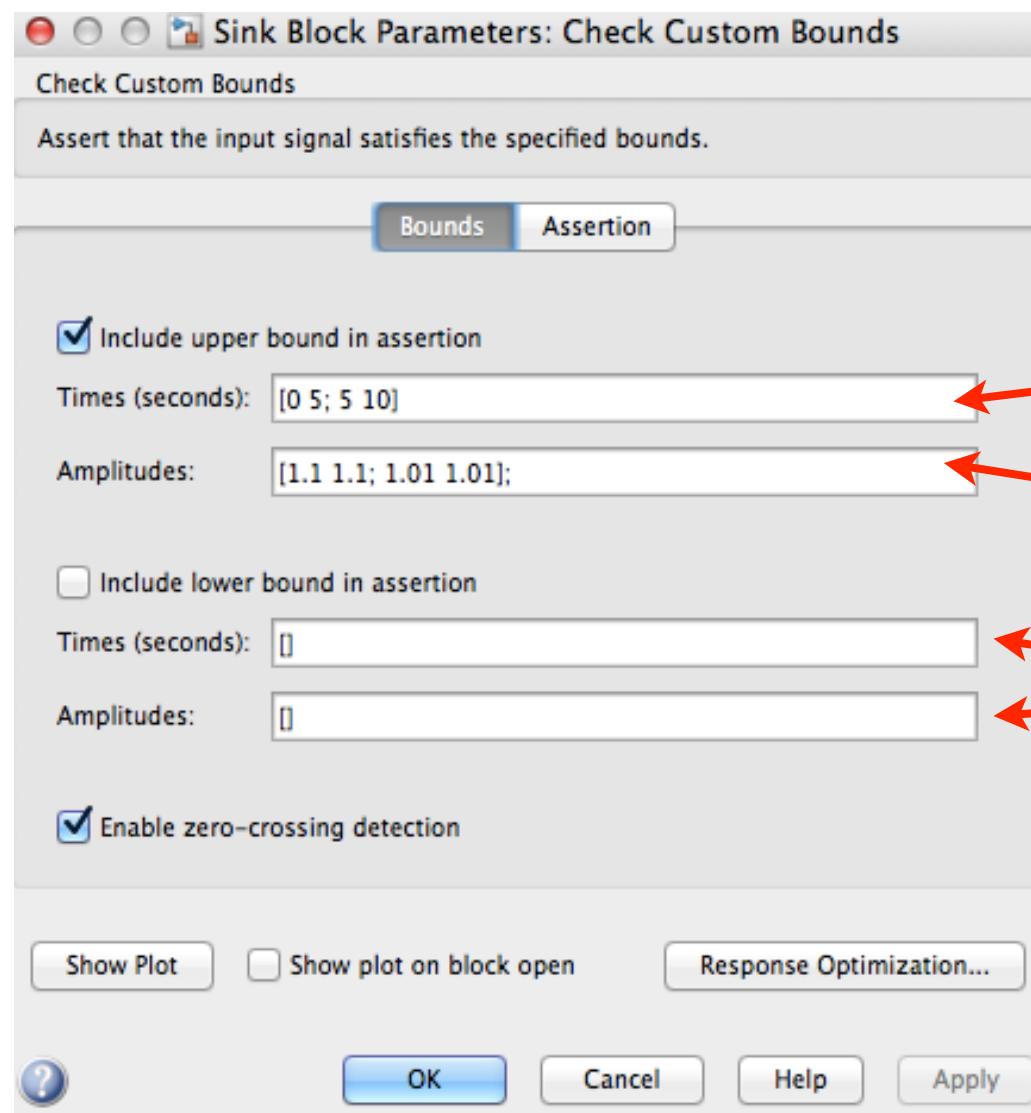
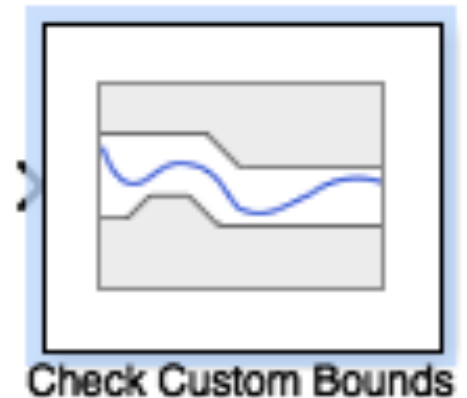
SIMULINK VERIFICATION

Simulink's Testing Suite

Model Verification

Check Custom Bounds

<https://uk.mathworks.com/help/sldo/ref/checkcustombounds.html>

The image shows the 'Sink Block Parameters: Check Custom Bounds' dialog box. It has a title bar with standard window controls and a subtitle 'Check Custom Bounds'. Below the subtitle is a description: 'Assert that the input signal satisfies the specified bounds.' There are two tabs: 'Bounds' (selected) and 'Assertion'. Under the 'Bounds' tab, there are two sections. The first section is for the upper bound, with a checked checkbox 'Include upper bound in assertion'. It contains two input fields: 'Times (seconds):' with the value '[0 5; 5 10]' and 'Amplitudes:' with the value '[1.1 1.1; 1.01 1.01];'. The second section is for the lower bound, with an unchecked checkbox 'Include lower bound in assertion'. It contains two empty input fields: 'Times (seconds):' and 'Amplitudes:'. At the bottom of the dialog, there is a checked checkbox 'Enable zero-crossing detection'. Below this are three buttons: 'Show Plot', 'Show plot on block open' (unchecked), and 'Response Optimization...'. At the very bottom are four buttons: '?', 'OK', 'Cancel', and 'Apply'.

Vector of time values for the bounds to hold

Bound values

Settings for lower bounds

SIMULINK VERIFICATION

Further Reading

Simulink Verification, Validation, and Test:

<https://uk.mathworks.com/solutions/verification-validation.html>

Using Unit Tests in Simulink:

<https://uk.mathworks.com/help/sltest/ug/run-test-files-using-matlab-unit-test.html>