MODELING OF AN ANTI-LOCK BRAKING SYSTEM

Name: Shrikant Dudhane.

Contents

Introduction

The target of ABS (Antilock braking system) is to produce the biggest conceivable braking power progressively while keeping the vehicle stable and avoiding excessive wheel slippage. ABS works when the braking force is more than the force of adhesion.
Antilock braking mechanism enhances the vehicle steadiness and steering ability to stop a vehicle wheel without locking and minimizing stopping distance.
The ABS screens the pace of every wheel to identify locking. When it recognizes sudden breaking, it will discharge breaking pressure for a moment and then continue optimum braking pressure to each wheel, by repeating this procedure in brief time frame, it upgrades steering control amid sudden stops.
This report shows how to model a simple model for an Anti-Lock Braking System (ABS). It simulates the dynamic behavior of a vehicle under hard braking conditions. The model represents a single wheel, which may be replicated a number of times to create a model for a multi-wheel vehicle.

Analysis and Physics

The wheel rotates with an initial angular speed that corresponds to the vehicle speed before the brakes are applied. We used separate integrators to compute wheel angular speed and vehicle speed. We use two speeds to calculate slip, which is determined by Equation 1. Note that we introduce vehicle speed expressed as an angular velocity.

$W_v = \frac{V}{R}$ (equals the wheel angular speed if there is no slip)

$$W_v = \frac{V_v}{R_r}$$

$$\text{Slip} = 1 - \frac{W_w}{W_v}$$

W_v = vehicle speed divided wheel radius

V_v = vehicle linear velocity

R_r = wheel radius

W_w = wheel angular velocity

From these expressions, we see that slip is zero when wheel speed and vehicle speed are equal, and slip equals one when the wheel is locked. A desirable slip value is 0.2, which means that the number of wheel revolutions equals 0.8 times the number of revolutions under non braking conditions with the same vehicle velocity. This maximizes the adhesion between the tire and road and minimizes the stopping distance with the available friction.

Modeling

The friction coefficient between the tire and the road surface, mu, is an empirical function of slip, known as the mu-slip curve. We created mu-slip curves by passing MATLAB variables into the block diagram using a Simulink lookup table. The model multiplies the friction coefficient, mu, by the weight on the wheel, W, to yield the frictional force, Ff, acting on the circumference of the tire. Ff is divided by the vehicle mass to produce the vehicle deceleration, which the model integrates to obtain vehicle velocity.

n this model, we used an ideal anti-lock braking controller, that uses 'bang-bang' control based upon the error between actual slip and desired slip. We set the desired slip to the value of slip at which the mu-slip curve reaches a peak value, this being the optimum value for minimum braking distance

Note: In an actual vehicle, the slip cannot be measured directly, so this control algorithm is not practical. It is used in this example to illustrate the conceptual construction of such a simulation model. The real engineering value of a simulation like this is to show the potential of the control concept prior to addressing the specific issues of implementation.

Model of anti-lock braking system in Simulink

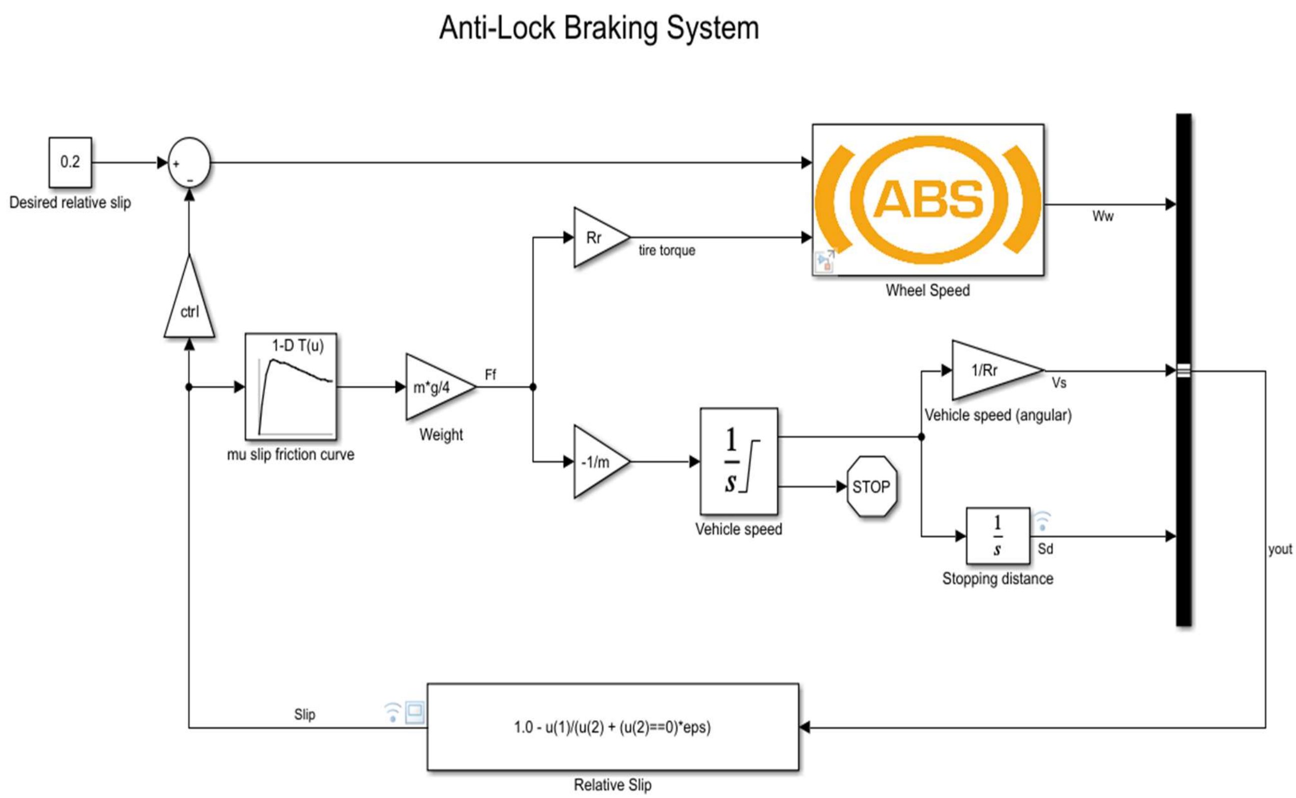


Figure 1: Anti-Lock Braking (ABS) Model

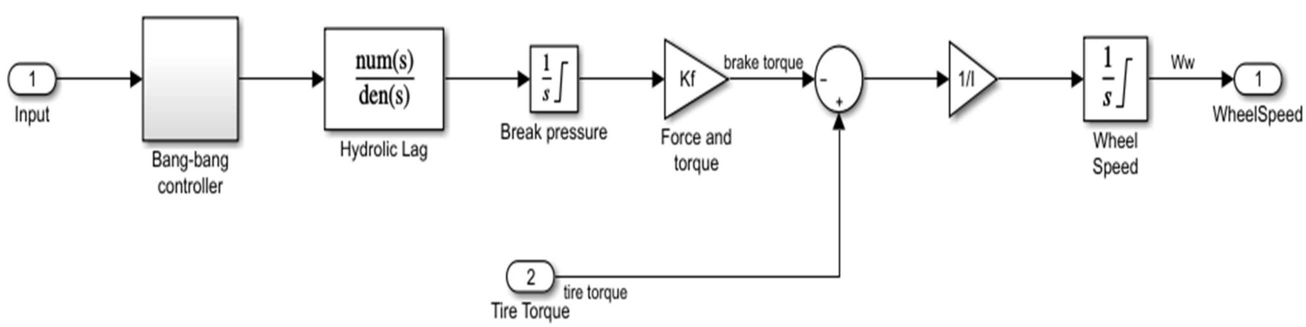


Figure 2: Wheel Speed **subsystem**

This subsystem gives us the wheel slip, the desired wheel slip, and the tire torque, this subsystem calculates the wheel angular speed.

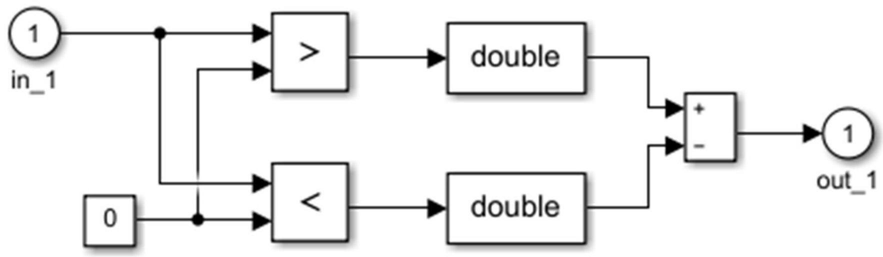


Figure 3: bang bang controller

To control the rate of change of brake pressure, the model subtracts actual slip from the desired slip and feeds this signal into a bang-bang control (+1 or -1, depending on the sign of the error, see Figure 2). This on/off rate passes through a first-order lag that represents the delay associated with the hydraulic lines of the brake system. The model then integrates the filtered rate to yield the actual brake pressure. The resulting signal, multiplied by the piston area and radius with respect to the wheel (Kf), is the brake torque applied to the wheel.

The model multiplies the frictional force on the wheel by the wheel radius (Rr) to give the accelerating torque of the road surface on the wheel. The brake torque is subtracted to give the net torque on the wheel. Dividing the net torque by the wheel rotational inertia, I, yields the wheel acceleration, which is then integrated to provide wheel velocity. In order to keep the wheel speed and vehicle speed positive, limited integrators are used in this model.

Running the Simulation in ABS Mode

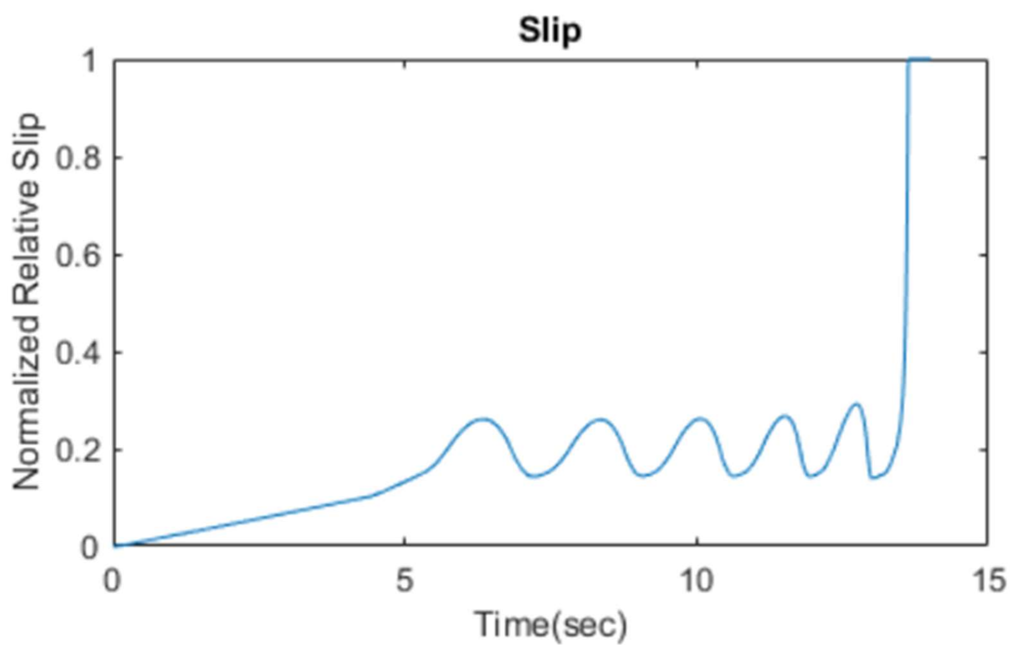
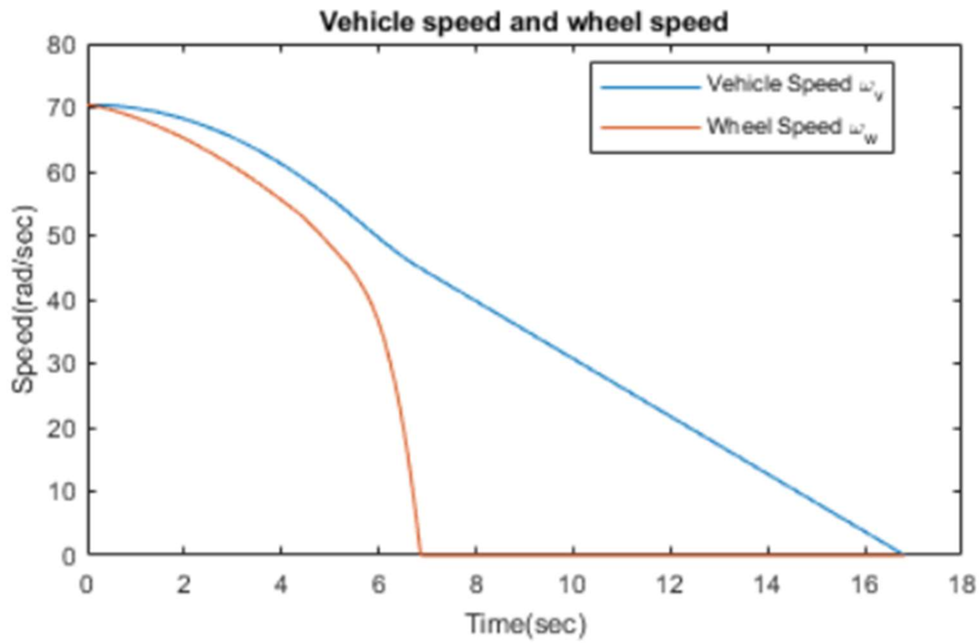


Figure 4: Running the simulation with ABS

Running the Simulation without ABS



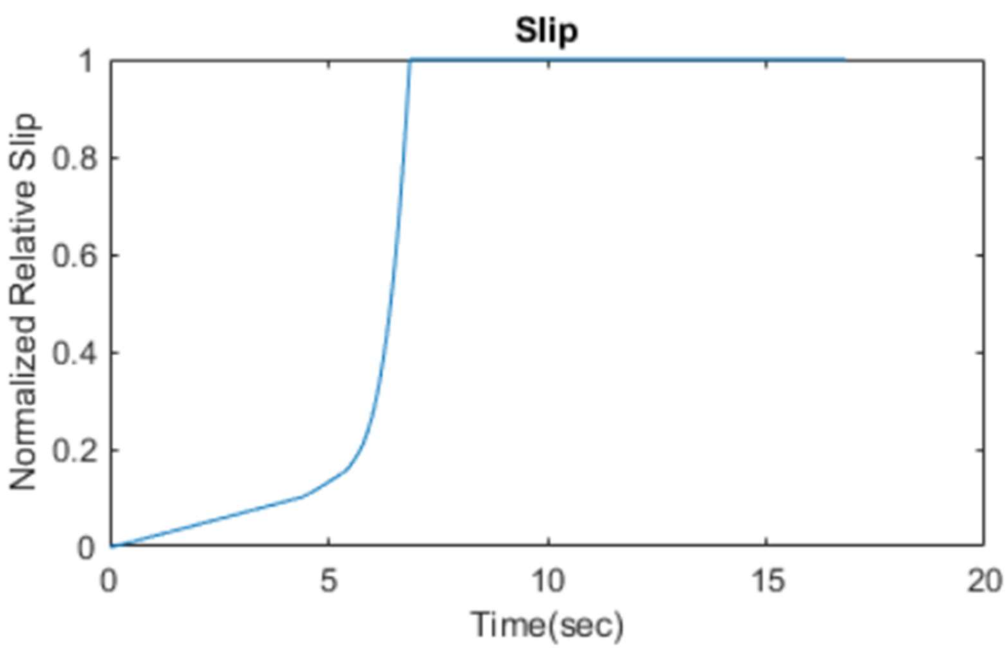


Figure 5: Running the simulation without ABS

Braking With ABS Versus Braking Without ABS

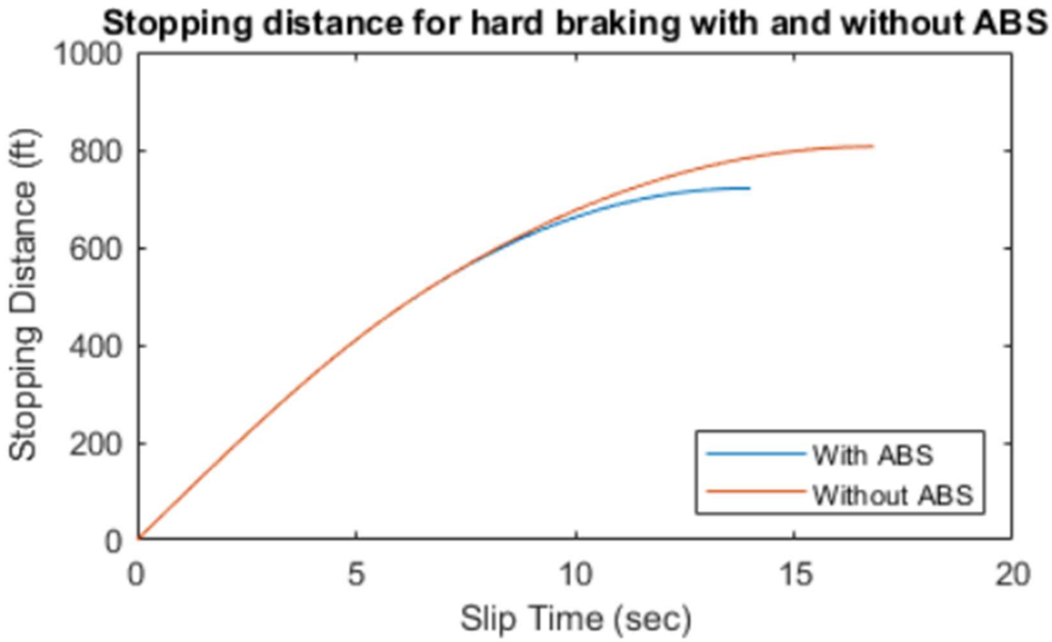


Figure 6: Braking with and without ABS

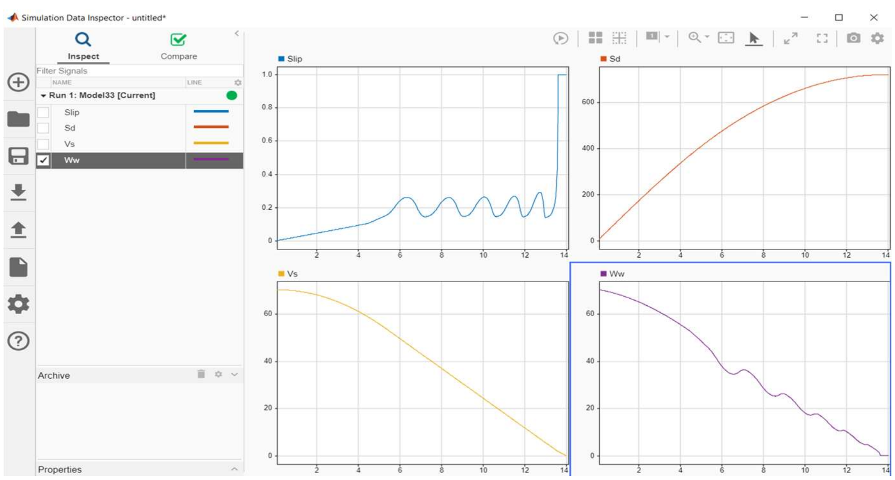
Highlights of the model

1.Data inspector:

Inspect and compare data and simulation results to validate and iterate model designs.

The Simulation Data Inspector visualizes and compares multiple kinds of data.

Using the Simulation Data Inspector, you can inspect and compare time series data at multiple stages of your workflow. This example workflow shows how the Simulation Data Inspector supports all stages of the design cycle

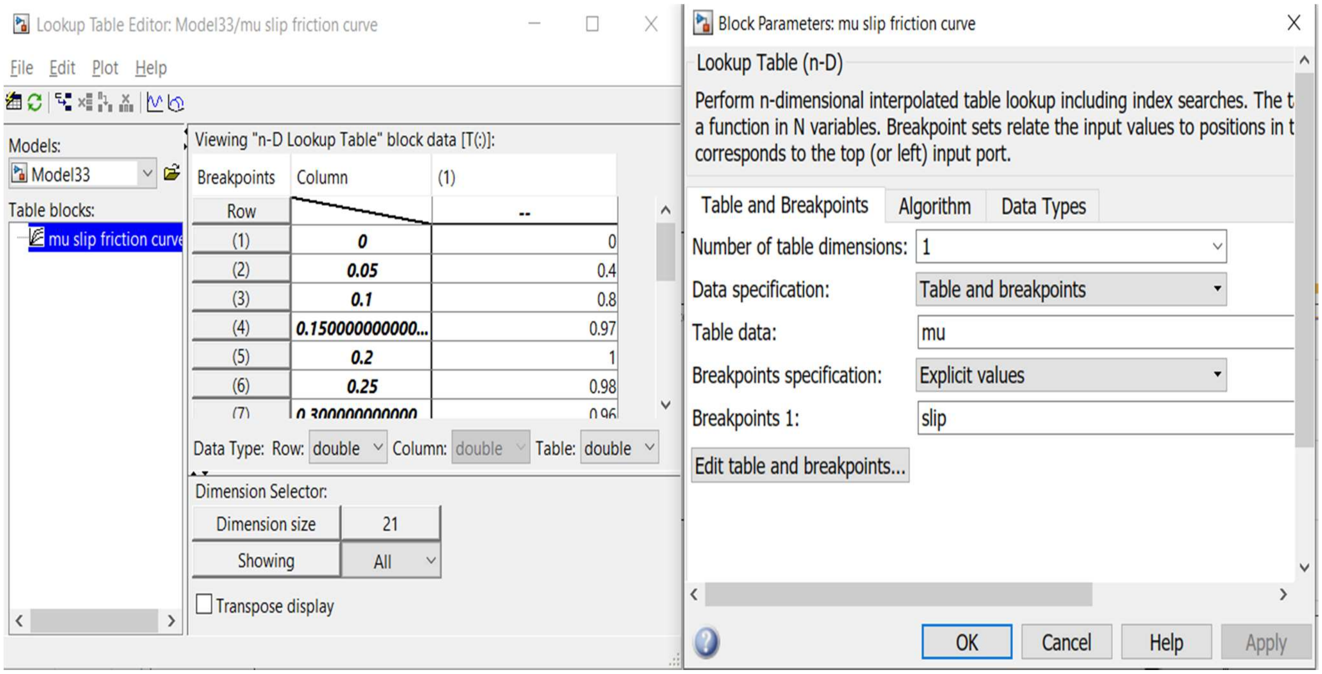


2. Look-up table:

The 1-D, 2-D, and n-D Lookup Table blocks evaluate a sampled representation of a function in N variables.

The block maps inputs to an output value by looking up or interpolating a table of values you define with block parameters. The block supports flat (constant), linear (linear point-slope), Lagrange (linear Lagrange), nearest, cubic-spline, and Akima spline interpolation methods. You can apply these methods to a table of any dimension from 1 through 30.

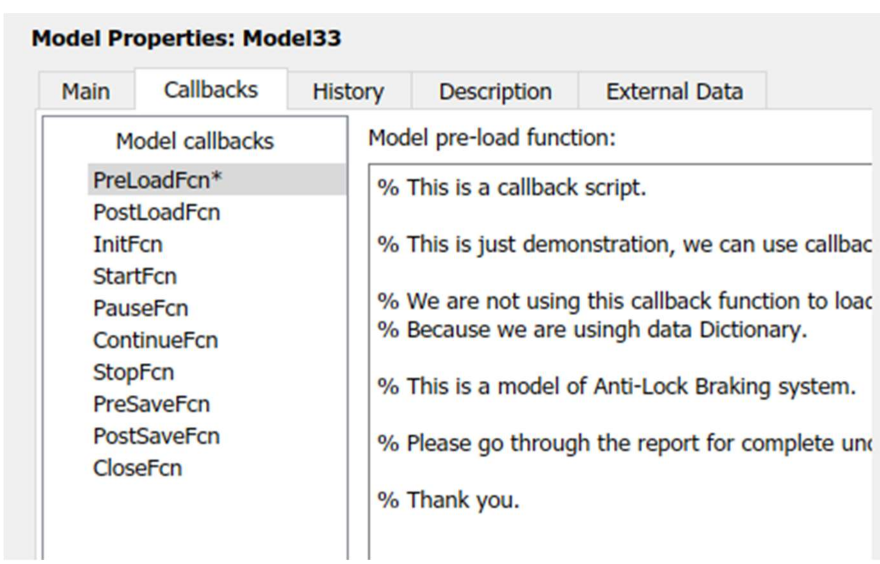
In the following block, the first input identifies the first dimension (row) breakpoints, the second input identifies the second dimension (column) breakpoints, and so on.



3. Callbacks:

Model callbacks execute at specified action points, for example after you load or save the model.

You can set most of the same callbacks for libraries. Only the callbacks that can execute for a library are available to set for a library. For example, you cannot set the InitFcn callback for a library, which is called as part of simulation, because you cannot simulate a library.



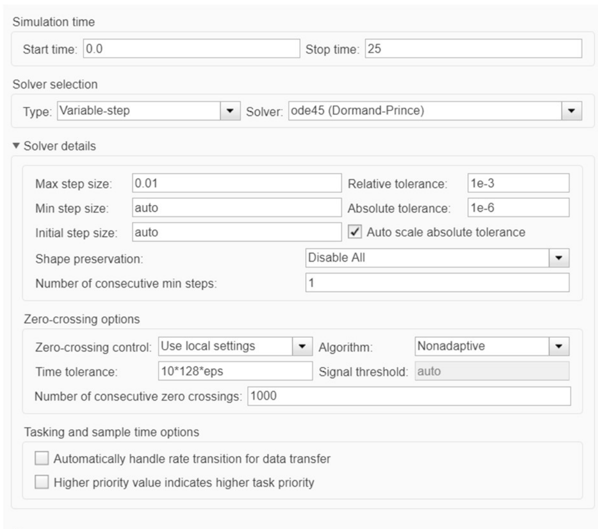
4. Solver selection strategy:

Variable-step Solver.

Ode45(dormand-prince)

Computes the model's state at the next time step using an explicit Runge-Kutta (4,5) formula (the Dormand-Prince pair) for numerical integration.

ode45 is a one-step solver, and therefore only needs the solution at the preceding time point.



Conclusion

This model shows how you can use Simulink to simulate a braking system under the action of an ABS controller. The controller in this example is idealized, but you can use any proposed control algorithm in its place to evaluate the system's performance.

For a hardware-in-the-loop braking system simulation, you can remove the 'bang-bang' controller and run the equations of motion on real-time hardware to emulate the wheel and vehicle dynamics.