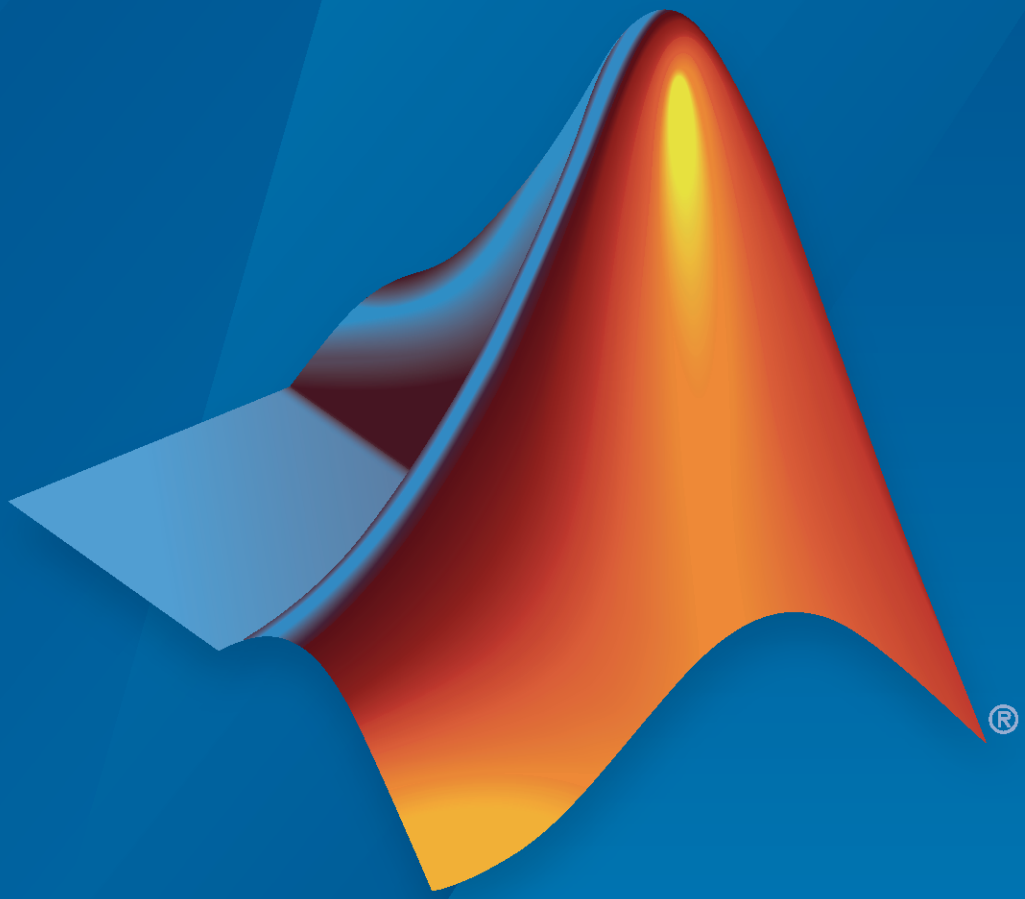
Motor Control Blockset™ Release Notes



MATLAB® & SIMULINK®

MathWorks®

How to Contact MathWorks

Latest news: `www.mathworks.com`

Sales and services: `www.mathworks.com/sales_and_services`

User community: `www.mathworks.com/matlabcentral`

Technical support: `www.mathworks.com/support/contact_us`

Phone: 508-647-7000

The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

Motor Control Blockset™ Release Notes

Contents

R2021a

R2020b

R2021a

Version: 1.2

New Features

Compatibility Considerations

Flux Observer: Implement sensorless control of induction motors

The Flux Observer block now works with induction motors.

This block, available under the Sensorless Estimators library, uses α- and β-axis voltages and currents to compute electrical position, magnetic flux, and electrical torque of either a PMSM or an induction motor. You can also use this block to implement field-oriented control of a brushless DC motor.

The block now uses an internal high-pass filter to remove noise, which results in a more accurate output.

Compatibility Considerations

When you use this version of the block with the previous Motor Control Blockset releases, the block supports only a PMSM.

When you use the older version of this block with the current Motor Control Blockset release, Simulink® configures the block to work with a PMSM by default. However, you can reconfigure the block to work with an induction motor.

IIR Filter: Improve sensorless control performance using high-pass filter

Use the IIR Filter block to implement a discrete infinite impulse response (IIR) high-pass filter.

This block, available under the Signal Management library, can now function as either a low-pass or high-pass filter. You can select the discrete step size to determine and set a cutoff frequency (Hz) for the filter.

When you use this version of the block with the previous Motor Control Blockset releases, the block always runs as a low-pass filter.

When you use the older version of this block with the current Motor Control Blockset release, the block behaves as a low-pass filter by default, but you can reconfigure it as a high-pass filter.

Motor Parameter Estimation: Estimate PMSM parameters without position sensors using F28069M controller

In addition to the LAUNCHXL-F28379D controller (with BOOSTXL-DRV8305 inverter), you can now use the Motor Control Blockset parameter estimation tool with the F28069 controller (with DRV8312-69M-KIT inverter) to determine PMSM parameters by using the sensorless flux observer.

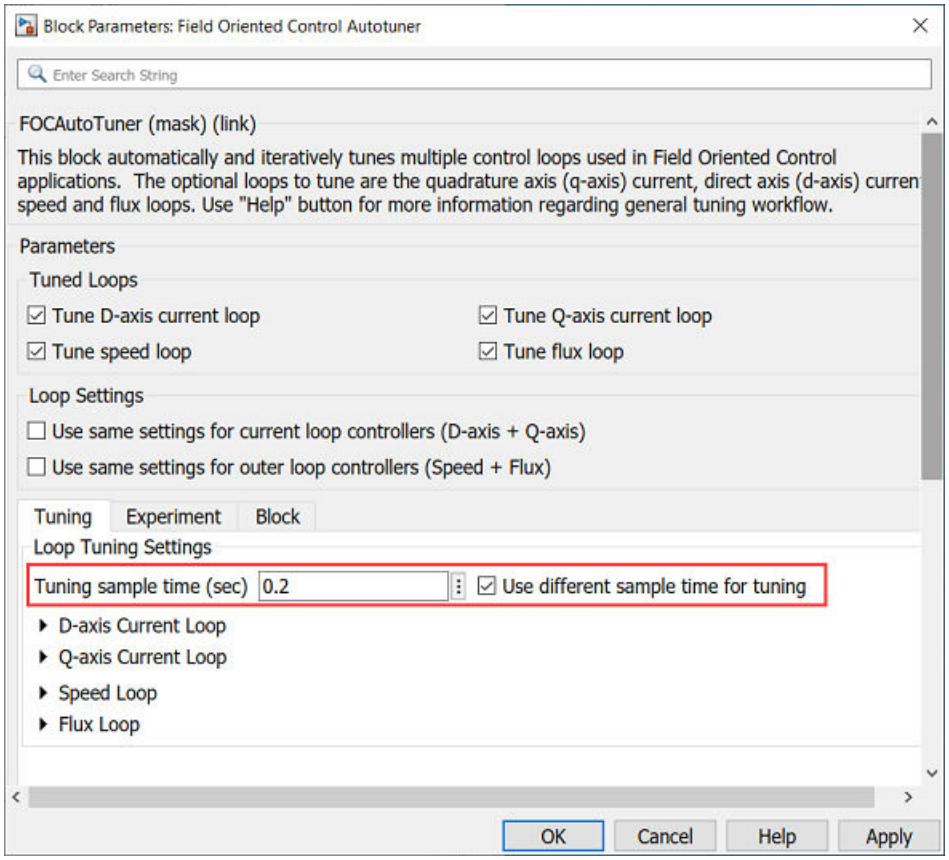
InitFcn Callback For PI Gains: Customize PI gain computation

The enhanced `mcb.internal.SetControllerParameters` function (used by the model initialization script associated with Motor Control Blockset examples) now includes more control parameters to help you customize computation of proportional integral (PI) controller gains.

Field Oriented Control Autotuner Block: Reduce target hardware throughput requirements by running autotuning at slower sample rate than PID controllers

The Field Oriented Control Autotuner block now lets you specify a different sample time for PID gain tuning. Previously, the block inherited the tuning sample time, for each loop, from the sample time specified in the **Controller sample time** parameter for that loop.

When you have a PID controller with a fast sample time and you run the tuning process at the same rate, some hardware might not complete PID gain calculation in a single time step. Therefore, when you have limited processing power and you want to tune controllers with fast sample times, enable the **Use different sample time for tuning** parameter and specify a sample time for tuning in the **Tuning sample time** parameter. For each loop that you tune, after the frequency response estimation experiment ends, controller tuning occurs at the sample time specified in the **Tuning sample time** parameter.



For more information about deploying the Field Oriented Control Autotuner block, see “Tune PI Controllers Using Field Oriented Control Autotuner Block on Real-Time Systems”.

R2020b

Version: 1.1

New Features

Compatibility Considerations

Induction Motors: Design and implement field-oriented control algorithms for three-phase induction machines

Implement field-oriented control (FOC) for AC Induction Motors (ACIM) by using these control algorithm blocks:

- The ACIM Control Reference block generates the reference currents for a control system.
- The ACIM Feed Forward Control block computes the decoupling terms *d* and *q*-axis voltages.
- The ACIM Slip Speed Estimator block estimates the slip speed of AC Induction Motors.
- The ACIM Torque Estimator block estimates torque and power of AC Induction Motors.

These blocks, available under the Controls/Control Reference library, support simulations with discrete and continuous solvers. They also support optimized code generation for both fixed-point and floating-point target systems.

Induction Motors: Model and simulate three-phase induction machines

Use the new Induction Motor model to design and validate motor control algorithms for three-phase induction motors.

This block, available under the Electrical Systems/Motors library, uses the mechanical load (torque or speed) and balanced three-phase voltage inputs to generate the mechanical (speed) and electrical (current) feedback.

BLDC Motors: Design and implement trapezoidal control using Six Step Commutation block

Use the new Six Step Commutation block to run a three-phase permanent magnet brushless DC (BLDC) motor in a 120-degree conduction mode.

The block, available under the Controls/Control Reference library, supports inputs either from Hall sensors or from any other position sensor.

The block uses Hall sensors to obtain the rotor position and compute the commutation sequence. It commutes the inverter switches at every 60 degrees such that the motor delivers maximum torque at a given position feedback. Use the block to implement a less complex but accurate speed control.

Motor Parameter Estimation: Identify PMSM parameters using quadrature encoder or flux observer

Determine motor parameters by using the quadrature encoder sensor or sensorless position algorithms, in addition to the existing Hall sensor based enhanced parameter estimation utility that runs the prebuilt instrumented tests on a motor. For more information, see Estimate Motor Parameters by Using Motor Control Blockset Parameter Estimation Tool.

Vector Plot Block: Visualize and verify motor control algorithms by plotting rotating phasors

Use the new Vector Plot to visualize space vectors corresponding to the different time-varying AC quantities such as voltages, currents, and flux.

The block, available under the Signal Management library, enables you to switch between the stationary and rotating reference frames to provide a graphical representation of the input vectors and better analyze the dynamics.

In addition to plotting the vectors, the block also traces the position history to represent the dynamics in both space and time in the same plot.

You can use the block to visualize the space vectors in different reference frames when designing and implementing the motor control algorithms. The block supports continuous and discrete solvers (including the fixed-step discrete solver) and floating-point simulation.

Sensorless Estimators: Compute more accurate rotor flux estimate

The Flux Observer block is improved in this release.

The enhanced Flux Observer block now computes a precise value of the rotor flux by eliminating the leakage flux from the total magnetic flux, and therefore, calculates the rotor position accurately for a PMSM.

Compatibility Considerations

When you use the older version of this block with the current Motor Control Blockset release, Simulink configures the block to use the default value for the newly added **Stator d-axis inductance (H)** parameter. However, you can change the parameter value.

R2020a

Version: 1.0

New Features

Introducing Motor Control Blockset: Design and implement motor control algorithms

Motor Control Blockset provides reference examples and blocks for developing field-oriented control algorithms for brushless motors. The examples show how to configure a controller model to generate a compact and fast C code for any target microcontroller (by using Embedded Coder®). You can also use the reference examples to generate algorithmic C code and driver code for specific motor control kits.

Reference Examples: Simulate field-oriented control and generate compact and fast C code for implementation on microcontroller (by using Embedded Coder)

Reference examples are setup to implement motor control algorithms for several supported motor control hardware kits.

For more information, see:

- Run 3-Phase AC Motors in Open-loop Control and Calibrate ADC Offset
- Field-Oriented Control of PMSM by Using Hall Sensor
- Field Oriented Control of PMSM by Using Quadrature Encoder
- Dual Motor (Dyno) Control for PMSM
- Use Motor Control Blockset™ to Generate Code for a Custom Target

Sensor Decoders and Sensorless Estimators: Implement sensor-based and sensorless motor control

Use the decoder blocks for Hall, resolver, and quadrature encoder sensors to calculate the position feedback. For more information, see Hall Validity, Hall Speed and Position, Quadrature Decoder, and Resolver Decoder.

Use the sensorless observers to calculate the rotor position. For more information, see Sliding Mode Observer, Flux Observer, and Sensorless Field-Oriented Control of PMSM Using Sliding Mode Observer and Flux Observer.

Controller Autotuning: Automatically tune current and speed loops

Automatically compute the initial PI controller gains for the speed and current loops based on the motor and inverter parameters. Use the Field Oriented Control Autotuner block to tune the speed and current loop gains of field-oriented controllers to achieve the specified bandwidth and phase margin for each loop (by using Simulink Control Design™). For more information, see Estimate Control Gains from Motor Parameters and Design Field-Oriented Control Algorithm.

Motor Parameter Estimation: Identify motor parameters from experiments with motor hardware

Determine these motor parameters by using the prebuilt instrumented tests and parameter estimation dashboard:

- Phase resistance (Rs)
- d and q axis inductances (Ld and Lq)
- Back-EMF constant (Ke)
- Motor inertia (J)
- Friction constant (F)

For more information, see Estimate Motor Parameters by Using Motor Control Blockset Parameter Estimation Tool.

Motor and Inverter Models: Verify control algorithms using closed-loop simulation

Verify control algorithms in closed-loop simulation with linear surface-mount and interior permanent magnet synchronous motor (PMSM) models and average-value inverter models.

For more information, see Create a Model with PMSM Block and Use Motor Parameters, Surface Mount PMSM, Interior PMSM, and Average-Value Inverter.