A Robust Exact Differentiator Block for MATLAB®/Simulink®

Markus Reichhartinger
`markus.reichhartinger@tugraz.at`

S.K. Spurgeon
`S.K.Spurgeon@kent.ac.uk`

Institue of Automation and Control
Graz University of Technology
DOI: 10.13140/RG.2.1.3243.4803

the toolbox is available at
click here

July 3, 2016

Contents

1 Download the Simulink® block

The required files and additional information about the block are available at:
www.reichhartinger.at

2 Introduction

Differentiation of signals is a frequently required task in, for example, signal processing devices and control loops. Even in the very simple application of a PID-controller, the derivative of a time-signal, typically the difference between reference signal and variable to be controlled, is required. Many methods have been developed to generate estimates of derivatives up to a certain, practically reasonable order. A comparison of three different methods, namely a B-Spline method, a method based on numerical integration and a sliding-mode based method, is given in [3]. The second approach which is based on numerical integration is outlined briefly in [5], the sliding mode based concept is known from the publications [2, 1, 4]. The differentiation algorithm implemented in the Simulink® block described in this document is proposed in [4]. Some advantages of this algorithm are:

- after a transient time, it provides an exact estimates in the case of a noise-free input signal.
- in the case of a noisy input signal the best possible accuracy is achieved.

3 How to cite this work

If this block is used to obtain results which are included in publications, please use the following reference:

- M. Reichhartinger and S.K. Spurgeon: *A Robust Exact Differentiator Block for MATLAB® / Simulink®*, DOI: 10.13140/RG.2.1.3243.4803, Technical University of Graz, 2016

4 How to setup the Simulink® differentiator block

The following steps have to be performed in order to use the Simulink® differentiator block:

1. Download the `red.zip` file
2. Unzip the folder within `red.zip` corresponding to the MATLAB®/Simulink® version you are using (e.g. `2013b_64Bit_windows`) into the folder which will be used as the working directory of your MATLAB®/Simulink® project.
3. This folder should then contain the files
   - `image.png`
   - `red.mexw64` or `red.mexw32`
   - `redBlk.slx` or `redBlk.mdl`

4. Open MATLAB®/Simulink®
5. Change your current working folder to the folder containing the above mentioned files.
6. Open the `redBlk` Simulink® block diagram
7. You should see a block diagram containing the block as depicted in Figure 1.

The input of the differentiator block is the scalar signal to be differentiated. The differentiator provides two output signals. The first output, which is labeled `c` and is a scalar signal, indicates whether the estimation errors of the differentiator vanish. The second output, which is labeled `x`, is a vector whose length equals the selected order of the differentiator. The elements of this output are the estimates of the derivatives of the input signal in ascending order, i.e. the first element of the output `x` is the estimate of the first derivative, the second element of `x` corresponds to the estimate of the second derivative, etc. If you double click

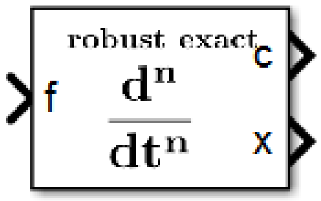
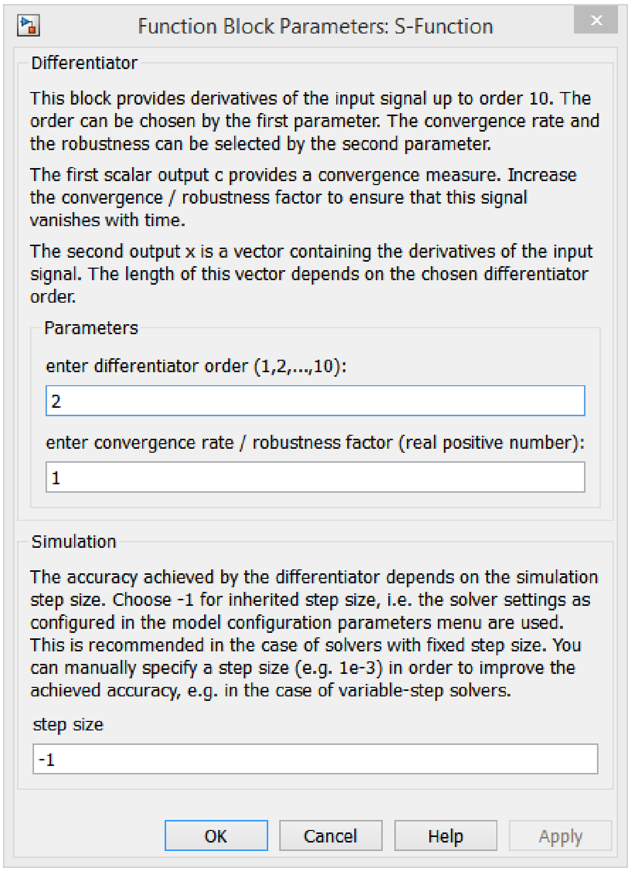


Figure 1: Simulink® block of the robust exact differentiator

the Simulink® block depicted in Figure 1, the function block description appears including the fields to enter the block parameters as depicted in Figure 2. The differentiator block has three parameters:

- the order of the differentiator,
- a convergence rate / robustness factor and
- the step size parameter.

The order of the differentiator has to be selected as an integer number from 1 to maximum order 10. The convergence rate / robustness factor and the step size parameter are positive real numbers.


Function Block Parameters: S-Function

Differentiator

This block provides derivatives of the input signal up to order 10. The order can be chosen by the first parameter. The convergence rate and the robustness can be selected by the second parameter.

The first scalar output c provides a convergence measure. Increase the convergence / robustness factor to ensure that this signal vanishes with time.

The second output x is a vector containing the derivatives of the input signal. The length of this vector depends on the chosen differentiator order.

Parameters

enter differentiator order (1,2,...,10):

2

enter convergence rate / robustness factor (real positive number):

1

Simulation

The accuracy achieved by the differentiator depends on the simulation step size. Choose -1 for inherited step size, i.e. the solver settings as configured in the model configuration parameters menu are used. This is recommended in the case of solvers with fixed step size. You can manually specify a step size (e.g. 1e-3) in order to improve the achieved accuracy, e.g. in the case of variable-step solvers.

step size

-1

Figure 2: Function Block Parameters of the differentiator block

5 Differentiation of a sinusoidal signal

In the subsequent sections a simple example demonstrating the use of the differentiator block is given.

5.1 Obtaining the first and second derivatives

This example shows how to set up a Simulink® simulation differentiating the sinusoidal signal given by $\sin(t + \pi/2)$. To do so, a block diagram as depicted in Figure 3 is implemented.

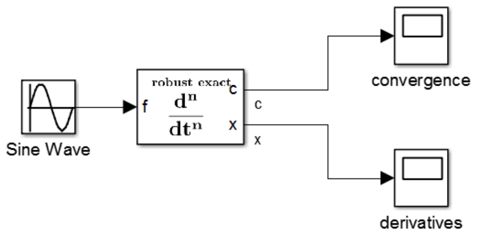


Figure 3: Differentiation of a sinusoidal signal.

The following setting of the differentiator is used:

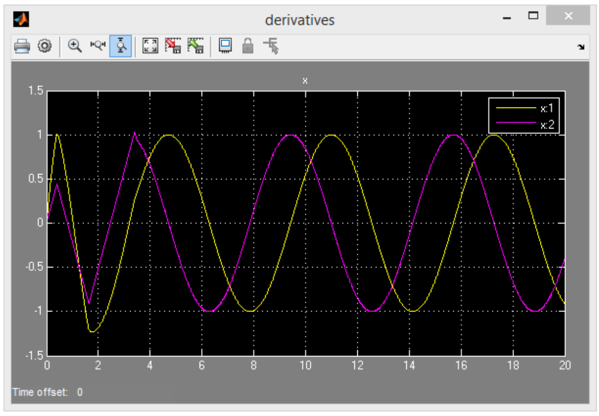
- differentiator order: 2
- convergence rate / robustness factor: 1
- step size: -1

The following Simulink® solver options are used[1]:

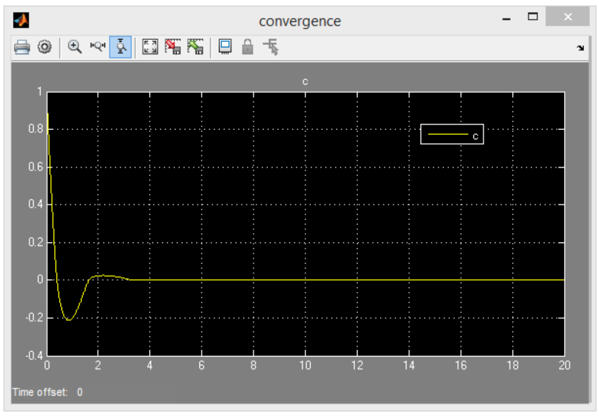
- Stop time: 20
- Type: Fixed-step
- Solver: ode3 (Bogacki-Shampine)
- Fixed-step size (fundamental sample time): 0.001

All the remaining Simulink® options are set to their default values. The results achieved by this simulation are shown in Figure 4.

[1]To enter the window in order to modify this setting use the hot key `Ctrl+E`.

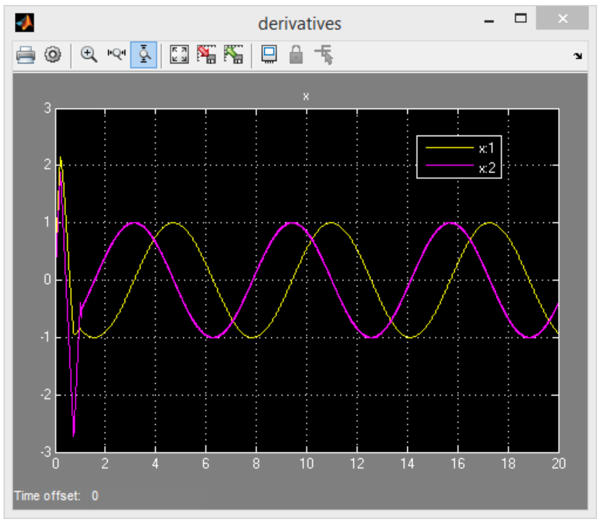


(a) Evolution of the estimated first and second derivative.

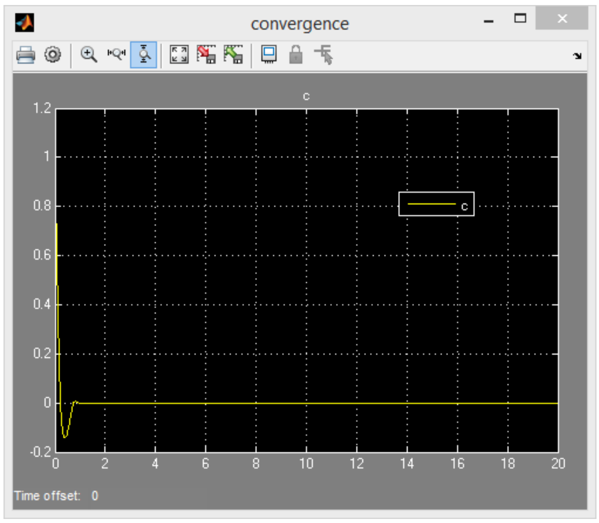


(b) Behaviour of the convergence output of the differentiator.

Figure 4: Simulation results with convergence rate / robustness factor: 1



(a) Evolution of the estimated first and second derivative.



(b) Behaviour of the convergence output of the differentiator.

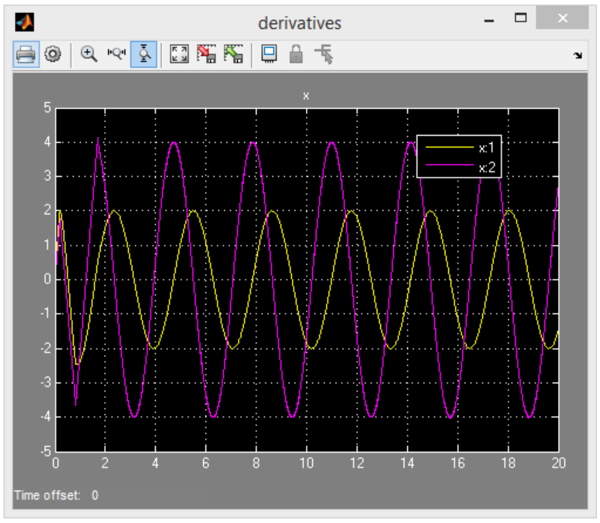
Figure 5: Simulation results with convergence rate / robustness factor: 2

5.2 Adjusting the convergence rate

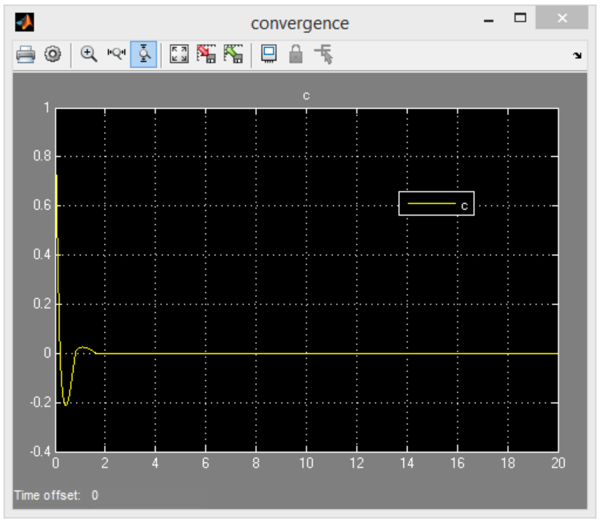
The speed of convergence of the estimation errors may be adjusted to a desired level using the convergence rate / robustness factor. Double click the differentiator block to open the parameter window and modify the convergence rate / robustness factor. Using the same settings of the Simulink® solver and the differentiator as in Section 5.1 and setting the convergence rate / robustness factor to 2 yields the simulation results shown in Figure 5. Note that by increasing the convergence rate, the absolute value of the estimated derivatives during the initial transient phase will typically reach higher values as seen by comparing Figure 4(a) with Figure 5(a). However, by increasing the convergence rate, the time required by the differentiator to converge reduces as seen by comparing Figure 4(b) with Figure 5(b).

5.3 Adjusting the robustness factor

Typically the input signal of the differentiator is not exactly known and therefore it may happen that the convergence signal of the differentiator does not vanish using the default settings of the differentiator block. This phenomenon can be illustrated by changing the input signal used in Section 5.1 from $\sin(t + \pi/2)$ to $\sin(2t + \pi/2)$ so that the frequency of the input signal is doubled. Increasing the robustness factor / convergence rate parameter from 1 to 2 causes the convergence output signal of the differentiator block to vanish. The results of this experiment are shown in Figure 6.



(a) Evolution of the estimated first and second derivative.



(b) Behaviour of the convergence output of the differentiator.

Figure 6: Simulation results with convergence rate / robustness factor: 2, the applied input signal is $\sin(2t + \pi/2)$.

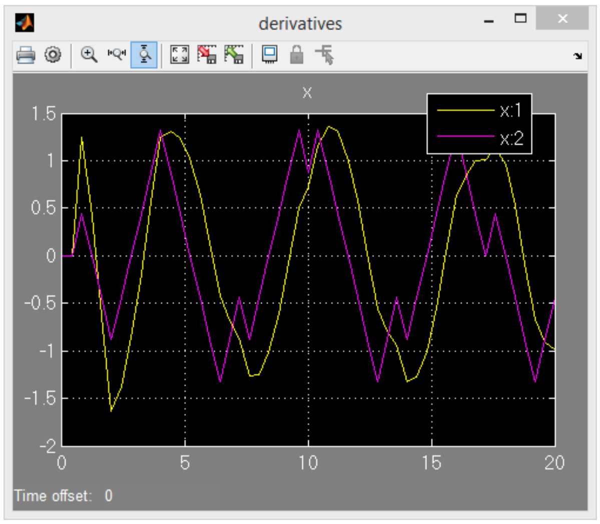
5.4 Impact of the selected solver

It is well known that the accuracy achieved by the differentiator depends on the step size of the solver, see e.g. [4]. This example demonstrates this behaviour. The same Simulink®

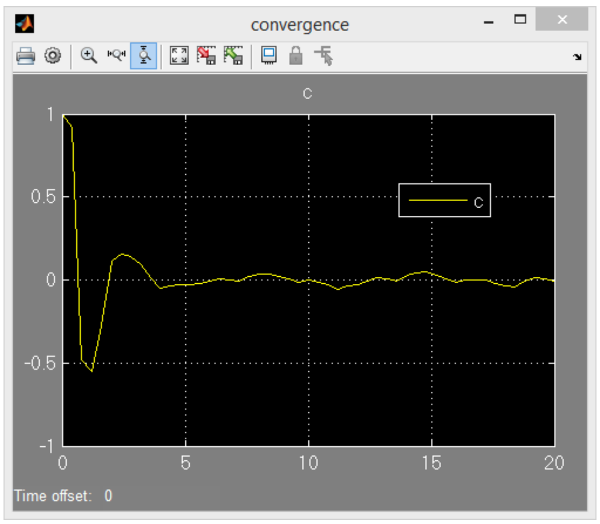
block diagram as used in Section 5.1 is used in this section. However, the solver settings used are now

- Stop time: 20
- Type: Variable-step
- Solver: ode45 (Dormand-Prince)
- Max. step size: auto
- Min. step size: auto
- Initial step size: auto

The simulation results achieved by this setting is shown in Figure 7.



(a) Evolution of the estimated first and second derivative.



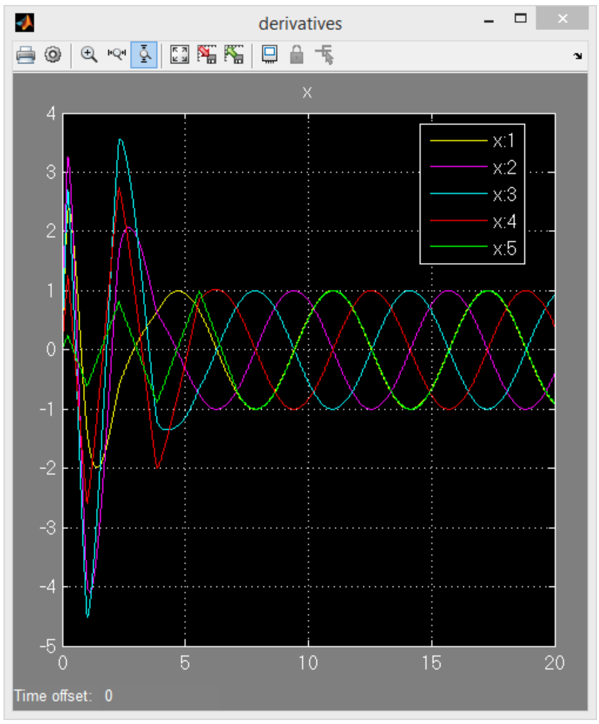
(b) Behaviour of the convergence output of the differentiator.

Figure 7: Simulation results: A Variable-step solver is used. The achieved accuracy is reduced drastically when compared to the results from Section 5.1.

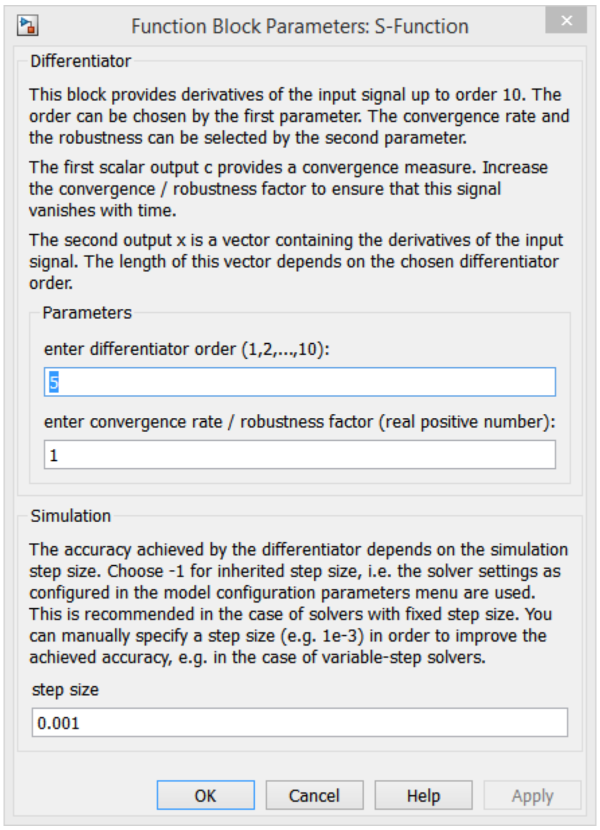
It becomes evident that the accuracy strongly depends on the selected solver and the adjusted step size. In order to improve the accuracy without modifying the solver setting the step size parameter of the differentiator may be used. Changing the step size parameter from `-1` to `0.001` recovers the simulation result as obtained in Section 5.1.

5.5 Obtaining higher derivatives

The order of the differentiator can be adjusted using the parameter window of the differentiator block. The setting of this parameter determines the dimension of the output `x` of the differentiator block. The simulation results depicted in Figure 8 are obtained using a Variable-step solver as used in Section 5.4.



(a) Evolution of the first five derivatives.



Function Block Parameters: S-Function

Differentiator

This block provides derivatives of the input signal up to order 10. The order can be chosen by the first parameter. The convergence rate and the robustness can be selected by the second parameter.

The first scalar output c provides a convergence measure. Increase the convergence / robustness factor to ensure that this signal vanishes with time.

The second output x is a vector containing the derivatives of the input signal. The length of this vector depends on the chosen differentiator order.

Parameters

enter differentiator order (1,2,...,10): 5

enter convergence rate / robustness factor (real positive number): 1

Simulation

The accuracy achieved by the differentiator depends on the simulation step size. Choose -1 for inherited step size, i.e. the solver settings as configured in the model configuration parameters menu are used. This is recommended in the case of solvers with fixed step size. You can manually specify a step size (e.g. 1e-3) in order to improve the achieved accuracy, e.g. in the case of variable-step solvers.

step size: 0.001

(b) parameters used to obtain derivatives up to order five.

Figure 8: Simulation results and differentiator parameter settings used to obtain the first five derivatives of the sinusoidal input signal also used in Section 5.1.

5.6 Impact of noise

In order to demonstrate the possibility of reducing the impact of noise, the block diagram shown in Figure 9 is implemented. It consists of three instances of the differentiator block. Two second order differentiators and one differentiator of order five are supplied with the input signal used in Section 5.1. However, the input signal of one second order differentiator and the fifth order differentiator are corrupted by noise (using the `Uniform Random Number` block with maximum amplitude 0.0001 for a sample time of 0.001).

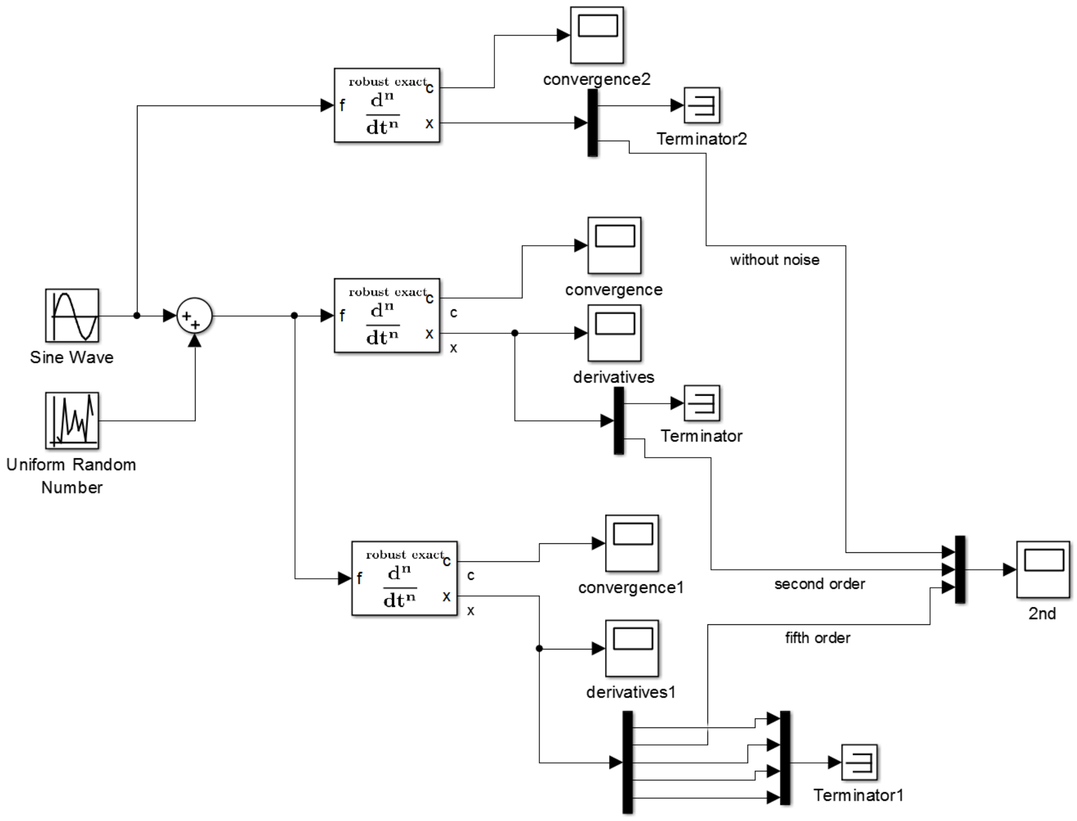


Figure 9: Block diagram implemented showing the reduction in the impact of noise using higher order differentiators

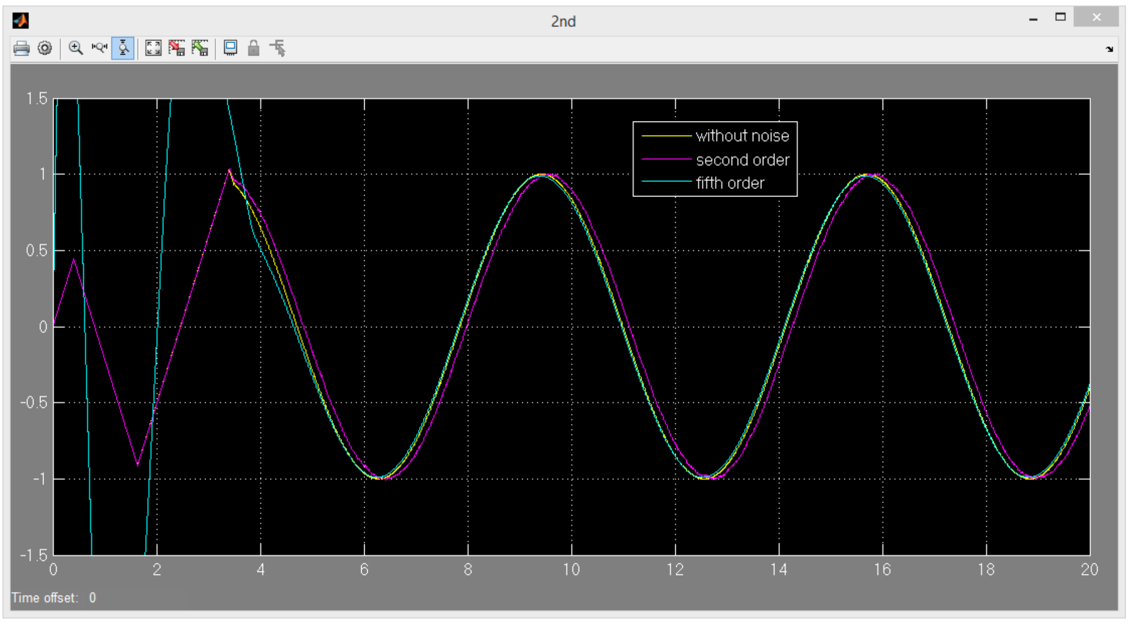


Figure 10: Comparison of the second derivatives obtained by second order differentiators and a fifth order differentiator. The result achieved for the second derivative by the fifth order differentiator is much closer to the result achieved by the second order differentiator which has the noise-free input signal.

6 Copyright and trademark notice

References

[1] A. Levant. Robust exact differentiation via sliding mode technique. *Automatica*, 34(3):379 – 384, 1998.

[2] A. Levant. Higher-order sliding modes, differentiation and output-feedback control. *International Journal of Control*, 76(9-10):924–941, 2003.

[3] K. D. Listmann and Z. Zhao. A comparison of methods for higher-order numerical differentiation. In *European Control Conference (ECC)*, pages 3676–3681, July 2013.

[4] M. Livne and A. Levant. Proper discretization of homogeneous differentiators. *Automatica*, 50(8):2007 – 2014, 2014.

[5] M. Mboup, C. Join, and M. Fliess. A revised look at numerical differentiation with an application to nonlinear feedback control. In *Control Automation, 2007. MED '07. Mediterranean Conference on*, pages 1–6, June 2007.