The Robust Exact Differentiator Toolbox revisited: Filtering and Discretization Features

Benedikt Andritsch
Institute of Automation and Control
Graz University of Technology
Graz, Austria
bandritsch@tugraz.at

Martin Horn
Christian Doppler Laboratory
for Model Based Control
of Complex Test Bed Systems
Institute of Automation and Control
Graz University of Technology
Graz, Austria
martin.horn@tugraz.at

Stefan Koch
Christian Doppler Laboratory
for Model Based Control
of Complex Test Bed Systems
Institute of Automation and Control
Graz University of Technology
Graz, Austria
stefan.koch@tugraz.at

Helmut Niederwieser
Institute of Automation and Control
Graz University of Technology
BEST - Bioenergy and Sustainable
Technologies GmbH
Graz, Austria
helmut.niederwieser@tugraz.at

Maximilian Wetzlinger
Institute of Automation and Control
Graz University of Technology
Graz, Austria
m.wetzlinger@tugraz.at

Markus Reichhartinger
Institute of Automation and Control
Graz University of Technology
Graz, Austria
markus.reichhartinger@tugraz.at

***Abstract*—An extended version of a Simulink®-block providing on-line differentiation algorithms based on discretized sliding-mode concepts is presented. Based on user-specified settings it computes estimates of the time-derivatives of the input signal up to order ten. Different discrete-time estimation algorithms as well as optional filtering properties can be selected. The paper includes an overview of the implemented algorithms, a detailed explanation of the developed Simulink®-block and two examples. The first example illustrates the application of the toolbox in a numerical simulation environment whereas the second one shows results obtained via an electrical laboratory setup.**

***Index Terms*—on-line discrete-time differentiation, sliding-mode observation, MATLAB®/Simulink®-toolbox, signal filtering**

I. Introduction

In applications of control, signal processing, fault detection, state and parameter estimation on-line differentiation of signals is an important operation. Therein, the signals to be differentiated are captured by analog-to-digital converters and are available for processing within a discrete-time environment such as micro-controllers or programmable logic controllers. In the case of on-line algorithms it is important to provide the computed results within one sampling period. In contrast to offline differentiation, the algorithms should not be computationally expensive in order to save processing power and to reduce performance degrading delays.

Due to the relevance of the topic several approaches for on-line differentiation have been developed. Apart from well-known standard approaches like finite-differences or ideal differentiators in combination with first order lag elements, more advanced observer-based algorithms have proven to be effective tools. In [1] a linear High Gain Differentiator is presented and its behavior in the presence of measurement noise is studied. In [2] a class of nonlinear continuous differentiators, which allow for the exact reconstruction of the derivatives of polynomial functions in the absence of measurement noise within finite time, is considered. For signals with a bounded $(n+1)^{\text{st}}$ derivative and in the absence of measurement noise the Robust Exact Differentiator (RED), see [3], [4], and the Uniform Robust Exact Differentiator (URED), see [5], allow for the exact reconstruction of the first n derivatives within finite time and fixed time, respectively. In order to improve the performance of the RED when differentiating noisy signals, a filtering extension is proposed in [6].

However, the discrete-time implementation of those advanced methods in a digital environment is not straightforward. Furthermore, especially when estimating derivatives of higher order, the tuning of the differentiator parameters is a cumbersome and challenging task.

In order to make the aforementioned algorithms easily accessible and applicable, an on-line differentiation toolbox for Simulink®has been implemented[1].

This work was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant agreement 734832. The financial support by the Christian Doppler Research Association, Austria, the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development, Austria is gratefully acknowledged.

[1]The toolbox is available for download at http://www.reichhartinger.at.

In contrast to the previous versions [7], [8], which implemented a certain discretization scheme of the RED only, the toolbox presented in this paper provides different discrete-time versions of a linear differentiator, a nonlinear but continuous differentiator, the RED and the URED. In addition, recently developed methods including filtering options for the differentiation of noisy signals are implemented. In order to keep the tuning as simple as possible, the choice of the differentiator parameters is reduced to the tuning of a so-called robustness factor. Also in contrast to the previous versions of the toolbox, automatic code generation for embedded hardware is supported. This allows for the direct application of the implemented differentiators in a real-time environment without the need of manually coding the algorithms.

Note that throughout the paper, the notation

$$f^{(k)}(t) = \frac{\mathrm{d}^k f(t)}{\mathrm{d}t^k} \quad \text{and, in particular,} \quad f^{(0)}(t) = f(t) \tag{1}$$

is recurrently used.

II. Methodology - Outline

All methods implemented in the toolbox follow the same basic principle, which is indicated in Fig. 1. The signal $f(t) = f^{(0)}(t)$ to be differentiated n times is considered to be generated by $n+1$ times integration of its $(n+1)^{st}$ derivative $f^{(n+1)}(t)$. The amplitude of $f^{(n+1)}(t)$ is assumed to be bounded by a constant $L \geq 0$, i.e.,

$$|f^{(n+1)}(t)| \leq L \quad \forall t, \tag{2}$$

which is usually satisfied in most practical applications. Typically, $f^{(0)}(t)$ is corrupted by some measurement noise $\eta(t)$, i.e., $\tilde{f}(t) = f^{(0)}(t) + \eta(t)$. Thus, $\tilde{f}(t)$ is regarded as the noisy output of the chain of $n+1$ integrators

$$\begin{aligned}
\frac{\mathrm{d}f^{(0)}}{\mathrm{d}t} &= f^{(1)}, \\
&\vdots \\
\frac{\mathrm{d}f^{(n-1)}}{\mathrm{d}t} &= f^{(n)}, \\
\frac{\mathrm{d}f^{(n)}}{\mathrm{d}t} &= f^{(n+1)},
\end{aligned} \tag{3}$$

with output

$$\tilde{f}(t) = f^{(0)}(t) + \eta(t). \tag{4}$$

The goal is to reconstruct the integrator states $f^{(0)}(t), \dots, f^{(n)}(t)$ of system (3) from the noisy output $\tilde{f}(t)$. However, even in the absence of measurement noise, i.e., $\eta(t) \equiv 0$, this problem is not trivial since the input $f^{(n+1)}(t)$ as well as the initial conditions of the integrators $f^{(0)}(t_0), \dots, f^{(n)}(t_0)$ are unknown. The methods implemented in this toolbox solve this task by using a observer-based approach. The observer

$$\begin{aligned}
\frac{\mathrm{d}z_0}{\mathrm{d}t} &= z_1 - \varphi_0(e)e, \\
&\vdots \\
\frac{\mathrm{d}z_{n-1}}{\mathrm{d}t} &= z_n - \varphi_{n-1}(e)e, \\
\frac{\mathrm{d}z_n}{\mathrm{d}t} &= -\varphi_n(e)e,
\end{aligned} \tag{5}$$

consists of a copy of the "virtual" integrator chain (3) and additional correction terms $\varphi_0(e)e, \dots, \varphi_n(e)e$, which are functions of the observer error $e = z_0 - \tilde{f}(t)$. The variable z_i, $i = 0, \dots, n$, denotes the estimate of the corresponding derivative $f^{(i)}(t)$. The derivative estimation methods implemented in the toolbox differ from each other in the particular choice of the correction terms $\varphi_i(e)$. The different methods ensure convergence of the derivative estimates z_i into a certain neighborhood of the actual derivatives $f^{(i)}(t)$. Depending on the particular choice of the correction terms, different theoretical properties in terms of the convergence behavior, the robustness w.r.t. the unknown input $f^{(n+1)}(t)$ and the sensitivity w.r.t. the measurement noise $\eta(t)$ are achieved.

As already mentioned above, each differentiator algorithm of the toolbox can be enhanced by some filtering characteristics.

A. Filtering Differentiator Design

For design purposes the integrator chain depicted in Fig. 1 is written in state-space representation

$$\Sigma_\mathrm{d} : \begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{f} &= \boldsymbol{A}_\mathrm{d}\boldsymbol{f} + \boldsymbol{b}_\mathrm{d} f^{(n+1)}, \\ \tilde{f} &= \boldsymbol{c}_\mathrm{d}^\mathrm{T}\boldsymbol{f} + \eta \end{cases} \tag{6}$$

with state vector $\boldsymbol{f} := \begin{bmatrix} f^{(0)} & \dots & f^{(n)} \end{bmatrix}^\mathrm{T}$, and $\boldsymbol{c}_\mathrm{d}^\mathrm{T} = \begin{bmatrix} 1\,0 \dots 0 \end{bmatrix}$, $\boldsymbol{b}_\mathrm{d}^\mathrm{T} = \begin{bmatrix} 0 \dots 0\,1 \end{bmatrix}$. The matrix $\boldsymbol{A}_\mathrm{d}$ is an upper-shift matrix of size $n+1$, i.e.,

$$\boldsymbol{A}_\mathrm{d} = \left(\begin{array}{c|c} \boldsymbol{0}_{n\times 1} & \boldsymbol{I}_{n\times n} \\ \hline 0 & \boldsymbol{0}_{1\times n} \end{array}\right) \tag{7}$$

where $\boldsymbol{I}_{n\times n}$ denotes the identity matrix of size n. The output of system Σ_d is connected in series to a filter of order $n_\mathrm{f}+1$ represented by

$$\Sigma_\mathrm{f} : \begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{g} &= \boldsymbol{A}_\mathrm{f}\boldsymbol{g} + \boldsymbol{b}_\mathrm{f}\tilde{f}, \\ g_0 &= \boldsymbol{c}_\mathrm{f}^\mathrm{T}\boldsymbol{g}, \end{cases} \tag{8}$$

where $\boldsymbol{g}$ is the state vector, $\boldsymbol{c}_\mathrm{f}^\mathrm{T} = \begin{bmatrix} 1\,0 \dots 0 \end{bmatrix}$, $\boldsymbol{b}_\mathrm{f}^\mathrm{T} = \begin{bmatrix} 0 \dots 0\,1 \end{bmatrix}$ and $\boldsymbol{A}_\mathrm{f}$ is an upper shift matrix of size n_f. The interconnection of System Σ_d and the filter Σ_f results in

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} \boldsymbol{g} \\ \boldsymbol{f} \end{bmatrix} &= \begin{bmatrix} \boldsymbol{A}_\mathrm{f} & \boldsymbol{b}_\mathrm{f}\boldsymbol{c}_\mathrm{d}^\mathrm{T} \\ \boldsymbol{0} & \boldsymbol{A}_\mathrm{d} \end{bmatrix}\begin{bmatrix} \boldsymbol{g} \\ \boldsymbol{f} \end{bmatrix} + \begin{bmatrix} \boldsymbol{b}_\mathrm{f}\eta \\ \boldsymbol{b}_\mathrm{d} f^{(n+1)} \end{bmatrix}, \\
g_0 &= \begin{bmatrix} \boldsymbol{c}_\mathrm{f}^\mathrm{T} & \boldsymbol{0} \end{bmatrix}\begin{bmatrix} \boldsymbol{g} \\ \boldsymbol{f} \end{bmatrix}.
\end{aligned} \tag{9}$$

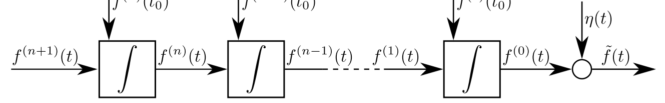


Fig. 1. The signal $f(t) = f^{(0)}(t)$ to be differentiated n times is obtained by $n+1$ times integrating its $(n+1)^{th}$ derivative $f^{(n+1)}(t)$. In practical applications, $f(t)$ is corrupted by measurement noise $\eta(t)$.

As mentioned above a differentiator is obtained by designing an observer for the interconnected system (9). The observer

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{w}\\ \boldsymbol{z}\end{bmatrix}=\begin{bmatrix}\boldsymbol{A}_\mathrm{f} & \boldsymbol{b}_\mathrm{f}\boldsymbol{c}_\mathrm{d}^\mathrm{T}\\ \mathbf{0} & \boldsymbol{A}_\mathrm{d}\end{bmatrix}\begin{bmatrix}\boldsymbol{w}\\ \boldsymbol{z}\end{bmatrix}-\begin{bmatrix}\boldsymbol{\varphi}_\mathrm{f}(\varepsilon_0)\\ \boldsymbol{\varphi}_\mathrm{d}(\varepsilon_0)\end{bmatrix}\varepsilon_0 \tag{10}$$

consists of a copy of system (9) and correction terms $\boldsymbol{\varphi}_\mathrm{f}(\varepsilon_0)$, $\boldsymbol{\varphi}_\mathrm{d}(\varepsilon_0)$ which are functions of the estimation error $\varepsilon_0 = w_0 - g_0 = \boldsymbol{c}_\mathrm{f}^\mathrm{T}(\boldsymbol{w}-\boldsymbol{g})$. Taking into account (10) and (9), the dynamics of the estimation errors $\boldsymbol{\varepsilon} := \boldsymbol{w}-\boldsymbol{g}$ and $\boldsymbol{e} := \boldsymbol{z}-\boldsymbol{f}$ yield

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{\varepsilon}\\ \boldsymbol{e}\end{bmatrix}=\left(\begin{bmatrix}\boldsymbol{A}_\mathrm{f} & \boldsymbol{b}_\mathrm{f}\boldsymbol{c}_\mathrm{d}^\mathrm{T}\\ \mathbf{0} & \boldsymbol{A}_\mathrm{d}\end{bmatrix}-\begin{bmatrix}\boldsymbol{\varphi}_\mathrm{d}(\varepsilon)\boldsymbol{c}_\mathrm{f}^\mathrm{T} & \mathbf{0}\\ \boldsymbol{\varphi}_\mathrm{f}(\varepsilon)\boldsymbol{c}_\mathrm{f}^\mathrm{T} & \mathbf{0}\end{bmatrix}\right)\begin{bmatrix}\boldsymbol{\varepsilon}\\ \boldsymbol{e}\end{bmatrix}-\begin{bmatrix}\boldsymbol{b}_\mathrm{f}\eta\\ \boldsymbol{b}_\mathrm{d}f^{(n+1)}\end{bmatrix}. \tag{11}$$

For implementation purposes the expression for $\frac{\mathrm{d}}{\mathrm{d}t}\varepsilon$ given in (11) is rearranged, by using the second Equation in (6), as

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\varepsilon}=\boldsymbol{A}_\mathrm{f}\boldsymbol{\varepsilon}+\boldsymbol{b}_\mathrm{f}\boldsymbol{c}_\mathrm{d}^\mathrm{T}\boldsymbol{z}-\boldsymbol{\varphi}_\mathrm{f}(\varepsilon_0)\varepsilon_0-\boldsymbol{b}_\mathrm{f}\tilde{f}, \tag{12}$$

which eventually gives, together with the differential equations for $\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{z}$ in (10), the filtering differentiator

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{\varepsilon}\\ \boldsymbol{z}\end{bmatrix}=\begin{bmatrix}\boldsymbol{A}_\mathrm{f} & \boldsymbol{b}_\mathrm{f}\boldsymbol{c}_\mathrm{d}^\mathrm{T}\\ \mathbf{0} & \boldsymbol{A}_\mathrm{d}\end{bmatrix}\begin{bmatrix}\boldsymbol{\varepsilon}\\ \boldsymbol{z}\end{bmatrix}-\begin{bmatrix}\boldsymbol{\varphi}_\mathrm{f}(\varepsilon_0)\\ \boldsymbol{\varphi}_\mathrm{d}(\varepsilon_0)\end{bmatrix}\varepsilon_0-\begin{bmatrix}\boldsymbol{b}_\mathrm{f}\\ \mathbf{0}\end{bmatrix}\tilde{f}. \tag{13}$$

The correction terms are obtained by prescribing $n+n_\mathrm{f}+1$ eigenvalues of the dynamic matrix in (11). In the implementation of this toolbox all eigenvalues are placed at

$$s_i = -r|\varepsilon_0|^{\frac{d}{1-d(n+n_\mathrm{f})}} \qquad i=1,\dots,n+n_\mathrm{f}+1 \tag{14}$$

which yields a family of continuous and discontinuous differentiators as discussed in [2], the parameter $d \in [0, -1]$ represents the homogeneity degree and r is a positive tuning parameter. The choice $d=-1$ gives the robust exact filtering differentiator (see [6]) whereas the choice $d=0$ gives the high gain observer [1] with additional filtering option.

B. Overview of the implemented concepts

The algorithm implemented in the toolbox is designed in an analogous way in discrete-time with sampling time T_s. There, the eigenvalues are assigned to $\lambda_{i,k}$.

Within the toolbox the discretization method can be adjusted via the parameter $m = \{0, 1, 2\}$. The choice

- $m=0$ yields the explicit discretization proposed in [9] and is characterized by
$$\lambda_{i,k} = 1+T_\mathrm{s}s_{i,k} = 1-T_\mathrm{s}r|\varepsilon_{0,k}|^{\frac{d}{1-d(n+n_\mathrm{f})}} \tag{15}$$
with $\varepsilon_{0,k}=\varepsilon_0(kT_\mathrm{s})$.
- whereas $m=1$ corresponds to the matching approach (see [8], [10]) obtained by selecting
$$\lambda_{i,k} = e^{T_\mathrm{s}s_{i,k}} = e^{-T_\mathrm{s}r|\varepsilon_{0,k}|^{\frac{d}{1-d(n+n_\mathrm{f})}}}. \tag{16}$$
- $m=2$ gives the URED, see [11], with the eigenvalues
$$\lambda_{i,k} = \frac{|\varepsilon_{0,k}|^{\frac{1}{n+n_\mathrm{f}+1}}}{T_\mathrm{s}r\mu|\varepsilon_{0,k}|+|\varepsilon_{0,k}|^{\frac{1}{n+n_\mathrm{f}+1}}+T_\mathrm{s}r}. \tag{17}$$

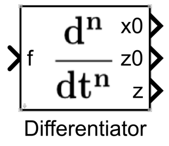


Fig. 2. Appearance of the Simulink block.

where μ is a positive tuning parameter.

With the two parameters m and d a variety of possibly filtering differentiators which have been proposed in literature can be obtained. For example:

- $m=1$, $d=0$ gives the discrete time *High gain observer* as presented in [12]
- $m=0$, $d=-1$ gives the sliding mode based so-called *Generalized Homogeneous Differentiator* presented in [9].
- $m=1$, $d=-1$ gives the sliding mode based so-called *Matching Differentiator* presented in [8], [10],
- $m=2$ gives a semi-implicit discretized URED of arbitrary order which can be obtained by generalizing the ideas presented in [11].

III. The Differentiator Simulink Block

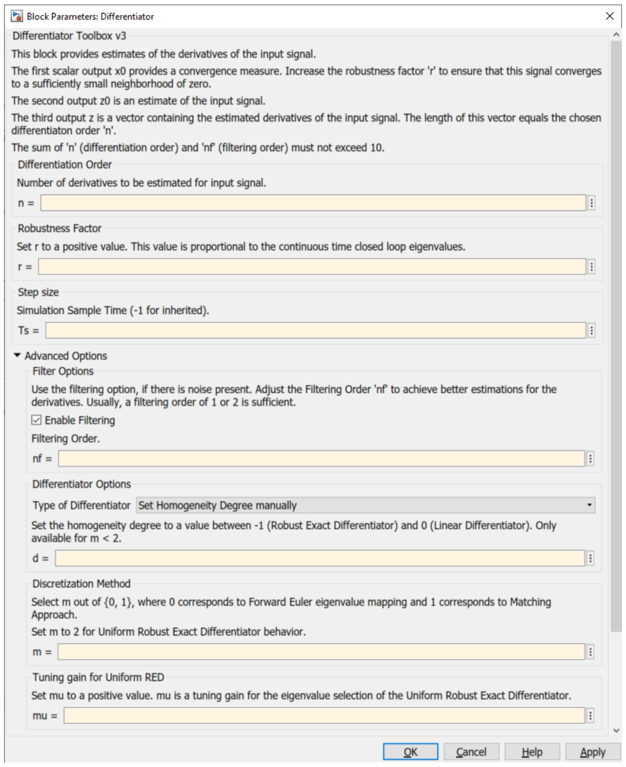
The described methods are implemented in MATLAB®/Simulink®and are available in form of the Simulink®block shown in Fig. 2. This block has one scalar input and three output ports. The first output x_0 provides a convergence measure for tuning the robustness factor parameter r. In the unfiltered case, the output equals the difference $f-z_0$ whereas in the filtered case, it is the negative first state variable, i.e., $-\varepsilon_0$. The second output z_0 is an estimate for the input signal f. The third and last output, the vector $\boldsymbol{z}$, contains the estimates of the n derivatives of the input signal, i.e., z_1 to z_n. The length of $\boldsymbol{z}$ therefore equals the differentiation order n.

Fig. 3 shows the corresponding user interface of the Simulink®block. The parameters to be tuned are the following: The differentiation order n, the robustness factor r, the sample time T_s, and, in case filtering is enabled, the filtering order n_f. Furthermore, there are the differentiator type and, if set to `Set Homogeneity Degree manually`, the homogeneity degree d. Finally, the discretization method can be adjusted via the parameter m, and the tuning gain for `Uniform RED` μ, in the case of $m=2$.

As explained in Section II, the discretization method can be selected by the parameter m which allows the values $0, 1$ or 2.

IV. Toolbox Application Examples

In this section the application of the toolbox is demonstrated by means of two examples. Numerical simulation results obtained by differentiating a signal suggested by literature are shown in Section IV-A. An electric circuit driven by an embedded control unit is used to illustrate the real-world applicability of the toolbox, see Section IV-B.


Block Parameters: Differentiator

Differentiator Toolbox v3

This block provides estimates of the derivatives of the input signal.

The first scalar output x0 provides a convergence measure. Increase the robustness factor 'r' to ensure that this signal converges to a sufficiently small neighborhood of zero.

The second output z0 is an estimate of the input signal.

The third output z is a vector containing the estimated derivatives of the input signal. The length of this vector equals the chosen differentiaton order 'n'.

The sum of 'n' (differentiation order) and 'nf' (filtering order) must not exceed 10.

Differentiation Order

Number of derivatives to be estimated for input signal.

n =

Robustness Factor

Set r to a positive value. This value is proportional to the continuous time closed loop eigenvalues.

r =

Step size

Simulation Sample Time (-1 for inherited).

Ts =

Advanced Options

Filter Options

Use the filtering option, if there is noise present. Adjust the Filtering Order 'nf' to achieve better estimations for the derivatives. Usually, a filtering order of 1 or 2 is sufficient.

[✓] Enable Filtering

Filtering Order.

nf =

Differentiator Options

Type of Differentiator: Set Homogeneity Degree manually

Set the homogeneity degree to a value between -1 (Robust Exact Differentiator) and 0 (Linear Differentiator). Only available for m < 2.

d =

Discretization Method

Select m out of {0, 1}, where 0 corresponds to Forward Euler eigenvalue mapping and 1 corresponds to Matching Approach.

Set m to 2 for Uniform Robust Exact Differentiator behavior.

m =

Tuning gain for Uniform RED

Set mu to a positive value. mu is a tuning gain for the eigenvalue selection of the Uniform Robust Exact Differentiator.

mu =

Fig. 3. User interface of the Simulink®block.

A. Simulation Example

In this section, the signal

$$\begin{aligned}\tilde{f}(t) &= f(t) + \eta(t), \quad f(t) = -0.4\sin t + 0.8\cos(0.8t),\\ \eta(t) &= \cos(10^4 t + 0.7791) + 0.0375\sin^2(100t)\lfloor\cos(100t)\rceil^{-\frac{1}{2}}\\ &\quad - 0.075\lfloor\cos(100t)\rceil^{\frac{3}{2}}, \qquad (18)\end{aligned}$$

which was defined in [6], serves as input signal to the differentiator block with different parameter settings. Note that there exist time instances where $\tilde{f}$ is unbounded. This was introduced in order to investigate the performance of differentiators also for such occurrences. Furthermore, in (18) the function $\lfloor a\rceil^b = |a|^b\operatorname{sign}(a)$ is used. In all simulations, the sample time is $T_\mathrm{s} = 10^{-5}$ s. In simulation scenario a, a RED without using an additional filter is used. Then, in simulation scenario b, the signal is applied to a RED with activated filter. Further, the results for a URED are shown in simulation scenario c and in simulation scenario d, a higher order RED with filter is used. For comparison purposes, in simulation scenario e, an unfiltered linear differentiator is used. The results can be seen in Fig. 4 to 9 respectively.

In all simulation scenarios, the actual signal f is reconstructed with satisfying accuracy despite the measurement noise η, see Fig. 4. Thus, the estimate z_0 for the actual signal f might also be used for signal denoising purposes.

The results of simulation scenario a show a significant deviation in the amplitude in the estimation of the first derivative, $z_{1,a}$, and a large phase shift in the estimation of the second derivative, $z_{2,a}$. Simulation scenario b shows results similar

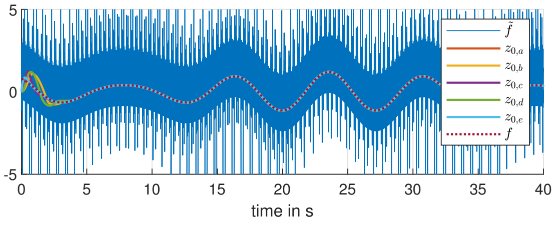


Fig. 4. Input function $\tilde{f}$, actual function f, and the estimates for the input function out of all 5 simulations scenarios (scenario a to scenario e).

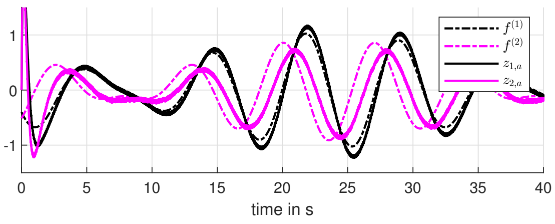


Fig. 5. First and second derivatives in simulation scenario a: Unfiltered RED, with $n = 2$, $n_\mathrm{f} = 0$, $r = 1$, $d = -1$ and $m = 1$.

to those in [6]. One can see only small deviations in the amplitude of $z_{1,b}$ and a small phase shift in $z_{2,b}$ compared to the true derivatives $f^{(1)}$ and $f^{(2)}$ respectively. By using the URED in simulation scenario c, faster convergence of the estimates is achieved. However, larger deviations in the initial transient phase can be observed. In simulation scenario d, the differentiation order is increased to $n = 4$, whereas the filtering order $n_f = 2$ is reduced in comparison to simulation scenarios b and c. Nevertheless, the deviation in amplitude in the estimation of the first derivative $z_{1,d}$ is reduced as well as the phase shift in the second derivative estimation, $z_{2,d}$. The last simulation scenario e was performed using a linear differentiator of order 2 without filter. The continuous-time transfer functions of these linear differentiators for the first

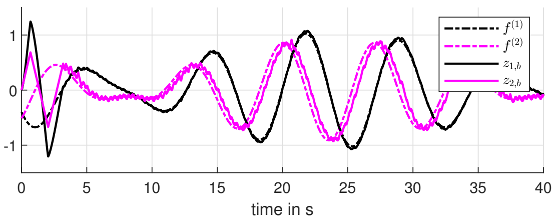


Fig. 6. Derivatives in simulation scenario b: Filtered RED, with $n = 2$, $n_\mathrm{f} = 3$, $r = 1$, $d = -1$ and $m = 1$.

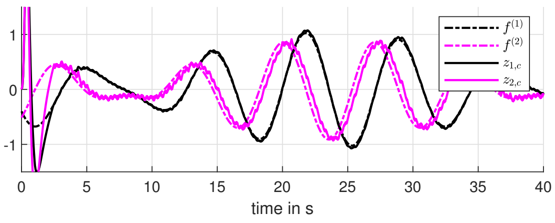


Fig. 7. Derivatives in simulation scenario c: Filtered URED, with $n = 2$, $n_\mathrm{f} = 3$, $r = 1$, $d = -1$, $m = 2$ and $\mu = 5000$.

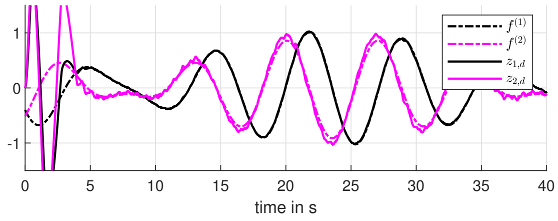


Fig. 8. Derivatives in simulation scenario d: Filtered higher order RED, with $n = 4$, $n_\mathrm{f} = 2$, $r = 1$, $d = -1$ and $m = 1$.

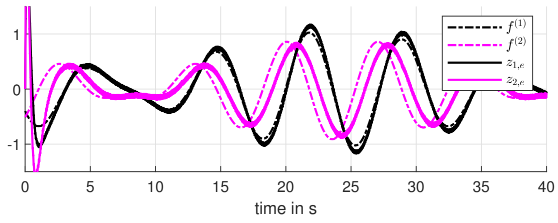


Fig. 9. Derivatives in simulation scenario e: Unfiltered linear differentiator, with $n = 2$, $n_\mathrm{f} = 0$, $r = 4$, $d = 0$ and $m = 1$.

and second derivatives are

$$G_1(s) = \frac{\mathcal{L}\{z_{1,e}(t)\}}{\mathcal{L}\left\{\tilde{f}(t)\right\}} = \frac{48s^2 + 64s}{s^3 + 12s^2 + 48s + 64}$$
$$G_2(s) = \frac{\mathcal{L}\{z_{2,e}(t)\}}{\mathcal{L}\left\{\tilde{f}(t)\right\}} = \frac{64s^2}{s^3 + 12s^2 + 48s + 64}. \quad (19)$$

The results are similar to those obtained by the RED without filter given in simulation scenario a, while showing a smaller phase shift in the estimate for the second derivative, $z_{2,e}$.

B. Application to a series resonator

In this section the presented differentiator toolbox is demonstrated by means of a real world experiment. For this purpose a series resonator as shown in Fig. 10 is used. The differentiation is performed for an on-line reconstruction of the current through the series resonator by differentiating the measured capacitor's voltage.

1) Mathematical Model: The dynamics of the circuit is modeled by

$$L\frac{\mathrm{d}x_1}{\mathrm{d}t} = -Rx_1 - x_2 + u \quad , \quad C\frac{\mathrm{d}x_2}{\mathrm{d}t} = x_1, \quad (20a)$$
$$y = x_2 + \eta, \quad (20b)$$

where R, L and C denote the resistance, the inductance and the capacity of the corresponding electric components, respectively. The voltage u is regarded as the input, the output y is given by the measured voltage x_2 which is corrupted by the noise η, the current through the circuit is denoted by x_1. In the present application the dynamics of the measured voltage x_2 of the capacitor C is used to estimate the current x_1 by

$$\hat{x}_1 = C\frac{\mathrm{d}y}{\mathrm{d}t}, \quad (21)$$

where $\hat{x}_1$ denotes the estimate of x_1, see Equation (20a).

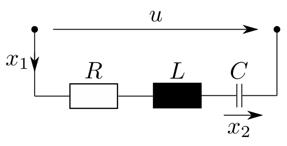


Fig. 10. Schematics of the series resonator used for real world experiments.

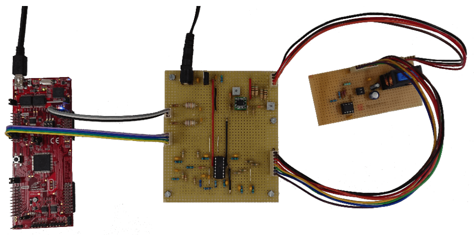


Fig. 11. Experimental setup consisting of a processor board of type *Texas Instruments LAUNCHXL-F28069M* (left), a connector board with a measuring amplifier (middle) and the circuit under investigation (right) [13].

2) Experimental Setup: Fig. 11 shows the experimental setup, which consists of a processor board of type *Texas Instruments LAUNCHXL-F28069M*, a connector board with a measuring amplifier and the RLC-series resonance circuit [13]. The parameters of this resonator are $R = 39.5\,\Omega$, $C = 78\,\mu$F and $L = 0.4$ H. Measurements in the circuit are collected via a MATLAB®/Simulink®interface between a computer and the processor board. This interface and the processor board limit the sampling frequency to less than 1 kHz to retrieve assured measurements without missing data points. Therefore, the sampling frequency was set to $500\,$Hz, leading to a sample time of $T_\mathrm{s} = 2$ ms. Furthermore, only steady-state measurements are considered, i.e., measurements were only recorded after about 15 s.

3) Results: As input source signal to the circuit the signal $u(t) = A\ \sin(wt)$, with the three frequencies $w_a = 2\,\frac{\mathrm{rad}}{\mathrm{s}}$, $w_b = 5\,\frac{\mathrm{rad}}{\mathrm{s}}$ and $w_c = 20\,\frac{\mathrm{rad}}{\mathrm{s}}$ and the three amplitudes $A_a = 1\,$V, $A_b = 1\,$V and $A_c = 2\,$V, respectively is used. These frequencies are far away from the resonance frequency of $(LC)^{-1/2} \approx 179\,\frac{\mathrm{rad}}{\mathrm{s}}$. This results in a bad signal-to-noise-ratio which renders the measurements problematic for direct computation of the current. The goal is to retrieve a better estimation of the current in the circuit by computing the first derivative of the measured signal y, see Equation (21).

In this experiment the following parameters of the differentiator toolbox were chosen: $n = 2$, $n_\mathrm{f} = 1$, $d = -1$, $m = 1$. This choice yields a RED of order $n = 2$ in combination with a filter of order $n_\mathrm{f} = 1$ discretized by the matching approach. The robustness factor r is adjusted depending on the frequency w. For the three different frequencies w_a, w_b and w_c, the robustness factors are $r_a = 2.1$, $r_b = 4.2$, $r_c = 14.0$ respectively. For comparison, also the scaled current measurements are shown. The estimate g is determined using the transfer function $G(s) = \frac{\mathcal{L}\{g(t)\}}{\mathcal{L}\{\tilde{f}(t)\}} = \frac{s}{\frac{s}{10\,w}+1}$ with a cutoff frequency of 10 times the input frequency. Fig. 12 shows the measurements $\tilde{f}$ with the three different input signals.

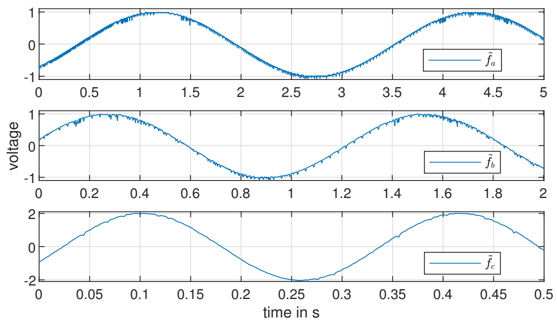


Fig. 12. Measurements of the voltage across the capacitor, $\tilde{f}(t)$, in experiments a to c.

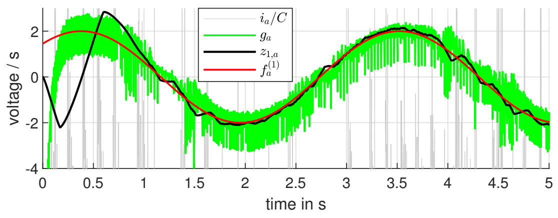


Fig. 13. Results of experiment a: Input signal with frequency $w_a = 2\frac{1}{s}$ and robustness factor $r_a = 2.1$.

The recorded noise is higher in the first two measurements. The results are summarized in Fig. 13 to 15. The analytical derivative $f^{(1)}$ (in red) was determined using the input signal $u(t)$ and the circuit parameters R, L and C. Especially in experiments a and b, the scaled current measurements i/C (in gray) deviate significantly from $f^{(1)}$. Here, it shall be stated, that the resulting figures where cut from above and below. The current i was measured by measuring the voltage across a shunt resistor of $4.8\,\Omega$. In experiments a and b, the amplitude of this shunt voltage is around $2\,\text{mV}$ respective $1\,\text{mV}$. It is noteworthy that such small voltages cannot be measured reliably with this experimental setup. The estimate g (in green) basically tracks $f^{(1)}$ correctly, however there is much noise present. The estimate z_1 for the derivative from the toolbox (in black) matches the analytical derivative $f^{(1)}$ much better in these experiments and contains only little noise. In the last experiment c, the scaled current measurements are much less noisy, but continue to contain a negative bias. The estimate g_c is less noisy, but also differs strongly from $f_c^{(1)}$ at certain time instants.

V. Conclusion

In this paper an updated version of the MATLAB®/Simulink®differentiator toolbox has been

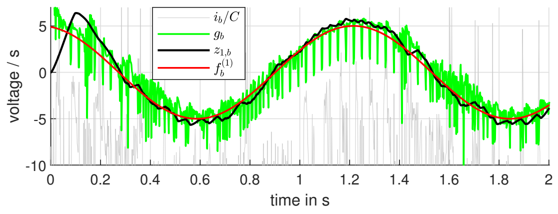


Fig. 14. Results of experiment b: Input signal with frequency $w_b = 5\frac{1}{s}$ and robustness factor $r_b = 4.2$.

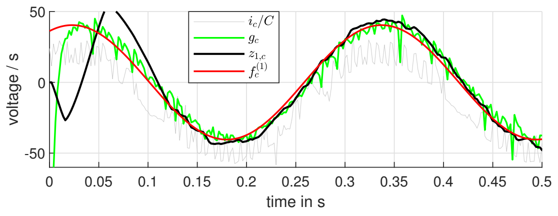


Fig. 15. Results of experiment c: Input signal with frequency $w_c = 20\frac{1}{s}$ and robustness factor $r_c = 14.0$.

presented. The Simulink®toolbox provides on-line estimates of the derivatives of the input signal of arbitrary order. This toolbox supports automatic code generation (in earlier versions this feature was supported upon request only). The tuning of the differentiator toolbox is essentially achieved by the selection of a single parameter (robustness factor) and thus is straightforward. Furthermore, now the toolbox provides a number of advanced options which allow to select a variety of differentiators such as the Robust Exact Differentiator, the Uniform Robust Exact Differentiator or the High Gain Observer. Additionally, the new version contains a filtering option which is beneficial when differentiating noisy signals. Exploiting the filtering option, the toolbox may also be used to perform signal denoising. The application of the toolbox has been demonstrated in simulation as well as in an practical example.

References

[1] L. K. Vasiljevic and H. K. Khalil, “Differentiation with high-gain observers the presence of measurement noise,” in *Proceedings of the 45th IEEE Conference on Decision and Control.* IEEE, 2006.

[2] E. Cruz-Zavala and J. A. Moreno, “Lyapunov functions for continuous and discontinuous differentiators,” *IFAC-PapersOnLine*, vol. 49, no. 18, 2016.

[3] A. Levant, “Robust exact differentiation via sliding mode technique,” *Automatica*, vol. 34, no. 3, 1998.

[4] ——, “Higher-order sliding modes, differentiation and output-feedback control,” *International journal of Control*, vol. 76, no. 9-10, 2003.

[5] E. Cruz-Zavala, J. A. Moreno, and L. M. Fridman, “Uniform robust exact differentiator,” *IEEE Transactions on Automatic Control*, vol. 56, no. 11, 2011.

[6] A. Levant and M. Livne, “Robust exact filtering differentiators,” *European Journal of Control*, vol. 55, 2020.

[7] M. Reichhartinger, S. Spurgeon, M. Forstinger, and M. Wipfler, “A robust exact differentiator toolbox for matlab®/simulink®,” *IFAC-PapersOnLine*, vol. 50, no. 1, 2017.

[8] M. Reichhartinger, S. Koch, H. Niederwieser, and S. K. Spurgeon, “The robust exact differentiator toolbox: Improved discrete-time realization,” in *2018 15th International Workshop on Variable Structure Systems (VSS).* IEEE, 2018.

[9] S. Koch, M. Reichhartinger, M. Horn, and L. Fridman, “Discrete-time implementation of homogeneous differentiators,” *IEEE Transactions on Automatic Control*, vol. 65, no. 2, 2019.

[10] S. Koch and M. Reichhartinger, “Discrete-time equivalent homogeneous differentiators,” in *2018 15th International Workshop on Variable Structure Systems (VSS).* IEEE, 2018.

[11] M. Wetzlinger, M. Reichhartinger, M. Horn, L. Fridman, and J. A. Moreno, “Semi-implicit discretization of the uniform robust exact differentiator,” in *2019 IEEE 58th Conference on Decision and Control (CDC).* IEEE, 2019.

[12] A. M. Dabroom and H. K. Khalil, “Discrete-time implementation of high-gain observers for numerical differentiation,” *International Journal of Control*, vol. 72, no. 17, 1999.

[13] F. Schober, “Design, realisation and control of electrical circuits for laboratory purposes,” Master’s thesis, Graz University of Technology, 2019.