A Robust Exact Differentiator Toolbox for Matlab Simulink: New Discrete-Time Realization

M. Reichhartinger*, S. Koch†, H. Niederwieser‡, S.K. Spurgeon§

***Abstract*—This paper presents a new release of A Robust Exact Differentiator Toolbox for Matlab/Simulink proposed in [1]. This release features a new discrete-time realization of the continuous-time robust exact differentiator. The implemented discretization scheme is less sensitive to gain overestimation and does not suffer from the discretization chattering effect. Hence, the single tuning parameter of the new version of the implemented differentiator is more intuitive to tune. Furthermore, it shows superior estimation performance in the case of large sampling times in comparison to the previous release. This is confirmed by the presented results obtained by numerical simulations and a real world application.**

I. Introduction

Real time differentiation and estimation of noisy signals is a core task in control engineering and signal processing. As documented by a large number of publications, the arbitrary order robust exact differentiator proposed in [2], [3] offers a simple and efficient solution to this problem. The algorithm is based on ideas of sliding mode control. In the absence of noise, the differentiator is capable of providing the exact n^{th} derivative of a time signal $f(t)$, assuming that its $(n+1)^{\text{th}}$ derivative with respect to time t is Lipschitz continuous with known Lipschitz constant L. Hence, after a finite time, the differentiator provides estimates which coincide with the signal f and its first n time derivatives.

Arbitrary order robust exact differentiators have been studied extensively. Their asymptotic estimation accuracy, if the signal f is corrupted by noise, is known to be optimal [2]–[4]. Furthermore, the estimation accuracy of the i^{th} derivative is proportional to τ^{n-i+1}, with $i = 0, 1, \ldots, n$, in the case of differentiating a time sequence generated by sampling the noise-free continuous time signal f with a constant sampling time τ, see also [2]–[4]. In applications where typically both effects are present, the estimation accuracy diminishes with the dominating part, i.e. either with the impact due to noise or due to sampling.

In the majority of applications, the algorithms need to be implemented in a real time environment, requiring a discrete-time realization of the continuous-time differentiator. However, discretization as well as the setting of the n constant tuning parameters of differentiators of orders higher than one is not straightforward. The authors in [5] proposed an extension of the forward Euler scheme such that the discrete-time algorithm preserves the above mentioned best possible asymptotic estimation accuracies. Although this scheme provides a theoretically sound strategy for a proper discretization of differentiators of arbitrary order, it does not address the parameter tuning problem. Typically one of the rare parameter settings of the differentiator is used in the presented simulation studies. In real world applications the setting of the differentiator, however, needs to be adjusted according to the characteristics of the signal to be differentiated. One option to generate a convergent parameter setting is to evaluate recently proposed Lyapunov functions either for certain orders of the differentiator, see [6]–[9], or for the arbitrary order case, see [10]. In order to avoid this cumbersome procedure the toolbox [1] may be used which is an implementation of the tuning paradigm presented in [11]. This allows the tuning effort to be reduced to the intuitive adjustment of a single parameter. The toolbox, which can be used for both numerical simulation studies as well as for real time experiments, is freely available for the MATLAB / SIMULINK environment[1]. The implementation uses the homogeneous discrete-time differentiator (HDD) presented in [5].

Recently developed discretization techniques motivate updating the above mentioned toolbox. Compared to the scheme proposed by [5], the new discrete time realization of the differentiator ensures vanishing estimation errors when the $(n+1)^{\text{th}}$ derivative vanishes and hence, the so-called discretization chattering is avoided, see e.g. [12]. Note that this is not the case when using discretization schemes based on the Euler forward scheme. In this case, a limit cycle like behavior of the estimation errors even in the case when the $(n+1)^{\text{th}}$ derivative vanishes is known to be generated. This feature is essential in for example fault detection tasks, where the $(n+1)^{\text{th}}$ derivative is not equal to zero in faulty cases. The development of the new discretization approach is based on a pseudo linear system representation (see, e.g. [13]) of the estimation error dynamics of the applied differentiators proposed in [11]. This facilitates an eigenvalue based tuning procedure which is also suggested in [11] and forms the

*Institute of Automation and Control, Graz University of Technology, Graz, Austria, e-mail: markus.reichhartinger@tugraz.at

†Christian Doppler Laboratory for Model Based Control of Complex Test Bed Systems, Institute of Automation and Control, Graz University of Technology, Graz, Austria, e-mail: stefan.koch@tugraz.at

‡H. Niederwieser is a master student of Information and Computer Engineering at Graz University of Technology, Graz, Austria

§Department of Electronic & Electrical Engineering, University College London, London, UK e-mail: s.spurgeon@ucl.ac.uk

[1]A version of the toolbox which also supports automatic code generation is available upon request.

basis for recently developed discretization approaches[2] [14]. In particular, a discrete time realization of the continuous time differentiator is generated by exploiting known eigenvalue mappings from linear systems theory. This strategy is outlined in Section II. Simulation studies and results of a real world experiment obtained by the application of the revised version of the differentiator toolbox are presented in Section III. A summary and concluding remarks are given in Section IV.

II. Brief Overview of the Implemented Algorithm

The algorithm implemented in the toolbox is a discrete-time version of the sliding mode based arbitrary order robust exact differentiator presented in [3]. In the so-called non-recursive representation, the differentiator is given by

$$\begin{aligned} \dot{\hat{x}}_i &= k_i \lfloor f - \hat{x}_0 \rceil^{\frac{n-i}{n+1}} + \hat{x}_{i+1}, \quad i = 0, \ldots, n-1 \\ \dot{\hat{x}}_n &= k_n \text{sign}(f - \hat{x}_0), \end{aligned} \tag{1}$$

where it is assumed that the signal to be differentiated, i.e. f, satisfies $f^{(n+1)} \in [-L,\, L]$. The constant parameters of the differentiator are denoted by k_1, k_2, …, k_n so that there are $n+1$ tuning parameters.

A. System Representation

In order to provide an overview of the applied discretization scheme, the differentiator is rewritten as

$$\frac{\mathrm{d}\hat{\boldsymbol{x}}}{\mathrm{d}t} = \boldsymbol{A}\hat{\boldsymbol{x}} + \boldsymbol{\psi}(\sigma_0)\sigma_0, \tag{2}$$

where the vector $\hat{\boldsymbol{x}} = \begin{bmatrix} \hat{x}_0 & \hat{x}_1 & \ldots & \hat{x}_n \end{bmatrix}^{\mathrm{T}}$ contains the estimates $\hat{x}_i$ of f and its time derivatives respectively. The matrix

$$\boldsymbol{A} = \left(\begin{array}{c|c} \boldsymbol{0}_{n\times 1} & \boldsymbol{I}_{n\times n} \\ \hline 0 & \boldsymbol{0}_{1\times n} \end{array}\right) \tag{3}$$

and $\boldsymbol{I}_{n\times n}$ denotes the $(n \times n)$ identity matrix. The vector

$$\boldsymbol{\psi}^{\mathrm{T}}(\sigma_0) = \begin{bmatrix} \psi_0(\sigma_0) & \psi_1(\sigma_0) & \ldots & \psi_n(\sigma_0) \end{bmatrix} \tag{4}$$

contains the injection terms

$$\psi_j(\sigma_0) = k_j |\sigma_0|^{-\frac{j+1}{n+1}}, \quad j = 0 \ldots n, \tag{5}$$

where the estimation error $\sigma_0 = f - \hat{x}_0$. For the purpose of analysis, the noise free signal f is assumed to be generated by a chain of $n+1$ integrators described by the state space model

$$\begin{aligned} \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e}_{n+1} f^{(n+1)}, \\ y &= \boldsymbol{e}_1^{\mathrm{T}} \boldsymbol{x}, \end{aligned} \tag{6}$$

with state vector $\boldsymbol{x} = \begin{bmatrix} x_0 & \ldots & x_n \end{bmatrix}^{\mathrm{T}}$. The vector $\boldsymbol{e}_i$ denotes the i^{th} standard basis vector. Introducing the estimation error vector $\boldsymbol{\sigma} = \boldsymbol{x} - \hat{\boldsymbol{x}}$ with its first component $\sigma_0 = f - \hat{x}_0 = \boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{\sigma}$ allows the estimation error dynamics to be presented

$$\frac{\mathrm{d}\boldsymbol{\sigma}}{\mathrm{d}t} = [\boldsymbol{A} - \boldsymbol{\psi}(\sigma_0)\boldsymbol{e}_1^{\mathrm{T}}]\boldsymbol{\sigma} + \boldsymbol{e}_{n+1} f^{(n+1)}, \tag{7}$$

This is a so-called pseudo linear system representation originally presented within this context in [11]. Furthermore, (7) is a homogeneous system with homogeneity degree -1. Its solutions are understood in the sense of Filippov [15]. The state-dependent dynamic matrix of system (7) can be written as

$$\boldsymbol{A} - \boldsymbol{\psi}(\sigma_0)\boldsymbol{e}_1^{\mathrm{T}} = \begin{bmatrix} -k_0|\sigma_0|^{-\frac{1}{n+1}} & 1 & 0 & \ldots & 0 \\ -k_1|\sigma_0|^{-\frac{2}{n+1}} & 0 & 1 & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ -k_{n-1}|\sigma_0|^{-\frac{n}{n+1}} & 0 & \ldots & \ldots & 1 \\ -k_n|\sigma_0|^{-1} & 0 & \ldots & \ldots & 0 \end{bmatrix} \tag{8}$$

The corresponding characteristic equation is given by

$$\begin{aligned} w(s, \sigma_0) = s^{n+1} + k_0|\sigma_0|^{-\frac{1}{n+1}} s^n + \ldots + \\ + k_{n-1}|\sigma_0|^{-\frac{n}{n+1}} s + k_n|\sigma_0|^{-1} = 0. \end{aligned} \tag{9}$$

In order to compute the eigenvalues s_j of the dynamic matrix given in equation (8), it is beneficial to multiply $w(s, \sigma_0)$ from equation (9) by $|\sigma_0|$, i.e.

$$\begin{aligned} (|\sigma_0|^{\frac{1}{n+1}} s)^{n+1} + k_0(|\sigma_0|^{\frac{1}{n+1}} s)^n + \ldots + \\ + k_{n-1}(|\sigma_0|^{\frac{1}{n+1}} s) + k_n = 0. \end{aligned} \tag{10}$$

From this representation, it is straightforward to see the structure of the eigenvalues

$$s_j = |\sigma_0|^{-\frac{1}{n+1}} p_j. \tag{11}$$

Substituting the eigenvalues s_j into the polynomial w yields

$$w(s_j, \sigma_0) = |\sigma_0|^{-1} \bar{w}(p) \tag{12}$$

with the polynomial

$$\bar{w}(p) = p^{n+1} + k_0 p^n + \ldots + k_{n-1} p + k_n, \tag{13}$$

This shows that the complex numbers p_j are the roots of the polynomial $\bar{w}(p)$. A stable dynamic matrix[3] $\boldsymbol{A} - \boldsymbol{\psi}(\sigma_0)\boldsymbol{e}_1^{\mathrm{T}}$ can be designed by selecting the tuning parameters k_j such that the polynomial $\bar{w}(p)$ is Hurwitz.

B. Discretization technique

The discretization paradigm followed in the implemented version of the toolbox is to establish a new discrete-time realization of the continuous time differentiator by adopting a discretization technique known from linear systems theory. In particular, it is known that an eigenvalue λ corresponding to the system matrix of a linear continuous time, time-invariant system is mapped according to

$$\lambda \to \mathrm{e}^{\lambda\tau}$$

using a zero-order hold discretization technique. The discrete time differentiator implemented in this toolbox aims to generate a discrete time estimation error dynamics characterized by eigenvalues which also satisfy this mapping. This is achieved

[2] The paper [14], which is also submitted to this conference, contains the theory of the discretization approach.

[3] The dynamic matrix in (8) is said to be stable if the real parts of all its eigenvalues s_j are negative for all σ_0.

by the design of a differentiator based on the discrete time system

$$\boldsymbol{x}_{k+1} = \boldsymbol{\Phi}(\tau)\boldsymbol{x}_k + \boldsymbol{h}(\tau) f_k^{(n+1)}, \tag{14}$$

with the state vector $\boldsymbol{x}_k = \begin{bmatrix} x_{0,k} & \dots & x_{n,k} \end{bmatrix}^{\mathrm{T}}$, obtained via zero-order hold discretization of system (6). Here, $f_k^{(n+1)}$ denotes the sampled signal $f^{(n+1)}(t)$ at time instant $t = k\tau$ with $k = 0, 1, 2, \dots$. The discrete time dynamic matrix $\boldsymbol{\Phi}$ and the input vector $\boldsymbol{h}$ of the discrete time system may be obtained using

$$\mathrm{e}^{\begin{bmatrix} \boldsymbol{A} & \boldsymbol{e}_{n+1} \\ \boldsymbol{0} & 0 \end{bmatrix}} = \begin{bmatrix} \boldsymbol{\Phi} & \boldsymbol{h} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{15}$$

The structure of the discrete time system given in (14) motivates the development of a discrete-time realization of the differentiator with structure

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{\Phi}\hat{\boldsymbol{x}}_k + \boldsymbol{\lambda}(\sigma_{0,k})^{\mathrm{T}} \sigma_{0,k}, \tag{16}$$

where

$$\boldsymbol{\lambda}(\sigma_{0,k}) = \begin{bmatrix} \lambda_0(\sigma_{0,k}) & \lambda_1(\sigma_{0,k}) & \dots & \lambda_n(\sigma_{0,k}) \end{bmatrix}^{\mathrm{T}} \tag{17}$$

and

$$\hat{\boldsymbol{x}}_k = \begin{bmatrix} \hat{x}_{0,k} & \dots & \hat{x}_{n,k} \end{bmatrix}^{\mathrm{T}}. \tag{18}$$

The discrete-time estimation error with respect to the signal $f(k\tau)$ is

$$\sigma_{0,k} = x_{0,k} - \hat{x}_{0,k} = f(k\tau) - \hat{x}_{0,k}. \tag{19}$$

Using the discrete-time observer (16) and the discrete-time representation (14), the discrete-time estimation error dynamics can be represented as

$$\boldsymbol{\sigma}_{k+1} = [\boldsymbol{\Phi} - \boldsymbol{\lambda}(\sigma_{0,k})\boldsymbol{e}_1^{\mathrm{T}}]\boldsymbol{\sigma}_k + \boldsymbol{h} f_k^{(n+1)}, \tag{20}$$

where $\boldsymbol{\sigma}_k = \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k$. The idea of the novel discretization technique which is presented in detail in [14] is to design the function $\boldsymbol{\lambda}(\sigma_{0,k})$ such that the eigenvalues of the matrix $[\boldsymbol{\Phi} - \boldsymbol{\lambda}(\sigma_{0,k})\boldsymbol{e}_1^{\mathrm{T}}]$ are located at $z_j = \mathrm{e}^{s_j \tau}$, where s_j are the continuous-time eigenvalues given in equation (11). The eigenvalue placement problem is solved by exploiting Ackerman's formula, where the function $\boldsymbol{\lambda}$ is computed by

$$\boldsymbol{\lambda}(\sigma_{0,k}) = \chi(\boldsymbol{\Phi}, \sigma_{0,k}) \boldsymbol{S}_o^{-1} \boldsymbol{e}_{n+1}, \tag{21}$$

with

$$\chi(\boldsymbol{\Phi}, \sigma_{0,k}) = \prod_{j=0}^{n} \left[\boldsymbol{\Phi} - z_j(\sigma_{0,k})\boldsymbol{I}\right], \tag{22}$$

where

$$z_j(\sigma_{0,k}) = \mathrm{e}^{s_j(\sigma_{0,k})\tau} = \mathrm{e}^{p_j \tau |\sigma_{0,k}|^{-\frac{1}{n+1}}} \tag{23}$$

are the desired closed loop eigenvalues. Hence, the desired characteristic polynomial is given by

$$\prod_{j=0}^{n} \left[\tilde{z} - z_j(\sigma_{0,k})\right]. \tag{24}$$

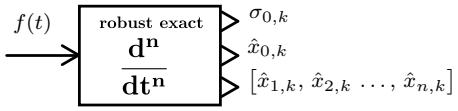

Fig. 1: Block Diagram of the Differentiator Toolbox.

The matrix $\boldsymbol{S}_o$ is the observability matrix, i.e. for the considered system

$$\boldsymbol{S}_o = \begin{bmatrix} \boldsymbol{e}_1^{\mathrm{T}} \\ \boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{\Phi} \\ \vdots \\ \boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{\Phi}^n \end{bmatrix}. \tag{25}$$

The differentiator design is therefore reduced to the selection of appropriate desired closed-loop eigenvalues z_j in equation (22).

C. Tuning of the differentiator parameters

Following the design of the discrete-time differentiator as outlined in Section II-B, it is clear that the tuning of the differentiator can be carried out by specifying $n+1$ roots p_j of the polynomial $\bar{w}(p)$ from equation (13). It is of interest to provide a toolbox implementing the proposed differentiation strategy in order to offer an intuitive and straightforward way to integrate the algorithm into existing simulation and real world implementations. In order to reduce the number of tuning parameters, all roots are selected as $p_j = -c$, where the positive real constant c can be adjusted by the user of the toolbox. This choice of the roots yields the desired closed-loop eigenvalues

$$z_j(\sigma_{0,k}) = \mathrm{e}^{-c\tau |\sigma_{0,k}|^{-\frac{1}{n+1}}}. \tag{26}$$

Following the selection $p_j = -c$ also in the parametrization of the continuous time differentiator yields the parameter $k_n = c^{n+1}$. This is evident from the polynomials (10) and (13) and considering that k_n is equal to the product of the specified eigenvalues. In the case of a known Lipschitz constant L and considering that $k_n > L$ is necessary in order to have an equilibrium of the estimation errors at the origin, it is natural to select $c > L^{\frac{1}{n+1}}$. This is consistent with the strategy adopted in a previous version of the toolbox, where the tuning was also reduced to a single positive real parameter labeled as the convergence rate/robustness factor, see [1]. This selection of the roots therefore ensures that the interface is consistent with the original version of the toolbox. In Fig. 1, the block diagram as it appears in a Matlab/Simulink implementation is given. It is seen that the differentiator requires the signal $f(t)$ as the input signal and that it provides the estimation error $\sigma_{0,k}$ and the estimates $\hat{\boldsymbol{x}}_k$ as output signals. The parameters τ and c can be adjusted as parameters of the differentiator block and the required order n of the differentiator can also be chosen. As in the previous version, differentiator order up to 10 is available.

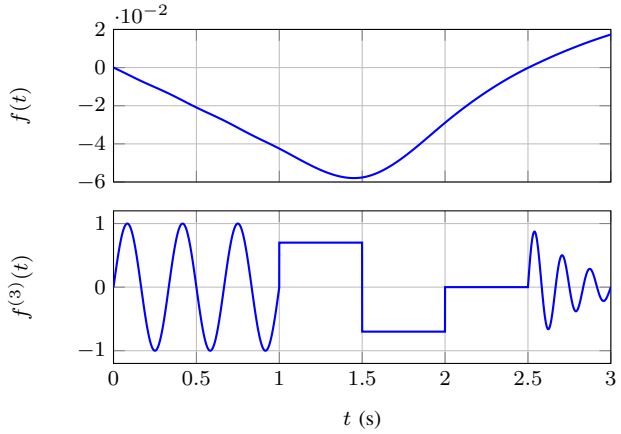

Fig. 2: Signal $f(t)$ to be differentiated and its third derivative.

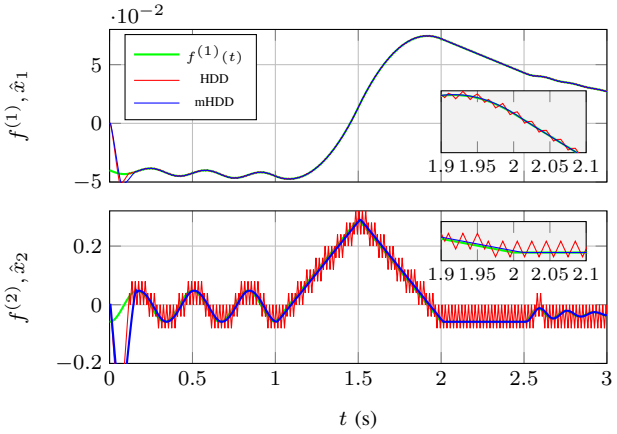

Fig. 3: Comparison of the estimated 1st and 2nd derivatives using both robust exact differentiator schemes with the analytical derivatives.

III. Simulation Examples & Application

The advantages of the discretization scheme exploited in the new release of the toolbox when compared to the implementation in the previous version will now be demonstrated using numerical simulation studies as well as laboratory implementation. In the following simulation examples, the order of the differentiator is set to $n = 2$. Both toolbox versions are used to differentiate the signal $f(t)$. To ensure Lipschitz continuity of the second derivative of the test signal, $f(t)$ is generated by a triple integration of the bounded signal

$$f^{(3)}(t) = \begin{cases} \sin(6\pi t) & 0 \leq t < 1, \\ 0.7 & 1 \leq t < 1.5, \\ -0.7 & 1.5 \leq t < 2, \\ 0 & 2 \leq t < 2.5, \\ \sin(12\pi t)e^{-\frac{t-2.5}{0.3}} & 2.5 \leq t, \end{cases} \tag{27}$$

which satisfies

$$\sup_t |f^{(3)}(t)| \leq 1. \tag{28}$$

Choosing the initial values $f^{(2)}(0) = -0.058$, $f^{(1)}(0) = -0.04$ and $f(0) = 0$ yields the signal $f(t)$ shown in Fig. 2. The convergence rate/robustness factor is set to $c = 2$ in both toolbox versions and the sampling time is chosen as $\tau = 5$ ms. The simulation results are shown in Fig. 3. The results obtained with the HDD, which is implemented in the first toolbox version, induces chattering due to discretisation. This is particularly noticeable in the time period $t \in [2, 2.5)$ where $f^3(t) = 0$. In contrast, the proposed discretization scheme, which is labeled mHDD, exhibits no chattering due to discretization. Plotting the homogeneous norm of the estimation errors, i.e.

$$\|\boldsymbol{\sigma}_k\|_h = \left(|\sigma_{0,k}| + |\sigma_{1,k}|^{\frac{3}{2}} + |\sigma_{2,k}|^3\right)^{\frac{1}{3}}, \tag{29}$$

over time t reveals that in this case the error state variables converge to zero hyper-exponentially, see Fig. 4 at $t = 2$ s. After some transient behavior the estimation errors of the

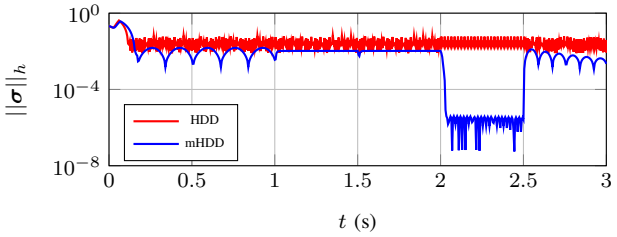

Fig. 4: Comparison of the estimation precision plotted as homogeneous norm of the error variables over time.

HDD enter the so-called real sliding set independent of the disturbance $f^{(3)}(t)$. Due to the presence of discretization chattering, higher accuracy is not achievable even in the case where the $(n+1)^{\text{th}}$ derivative vanishes. With the proposed discretization scheme, the estimation quality is directly related to the perturbation $f^{(3)}$ which is clearly visible during the time span $t \in [2.5, 3]$ where the amplitude of $f^{(3)}$ exponentially decreases with time. As mentioned in Section II-C, the tuning of the differentiator is reduced to the selection of a single parameter, the so-called robustness factor c. The choice $c > L\frac{1}{n+1}$ yields vanishing estimation errors. This is illustrated in Fig. 5. However, as also can be seen in Fig. 5, an overestimation of the parameter c results in a loss of precision in the case of the HDD. This is essentially a disadvantage when differentiating signals without knowledge of L. In such a situation, finding a meaningful value of c may be a challenging task. On the other hand, the proposed scheme ensures that the precision does not suffer from overestimation of the parameter c in the absence of noise. Fig. 5 also illustrates that the differentiator obtained with the proposed discretization scheme requires slightly larger gains in order to outperform the HDD. In a second simulation study, the precision with respect to noise is demonstrated by differentiating the polynomial $f(k\tau) = f_0(k\tau) + \eta_k$, where $f_0(k\tau) = (k\tau)^4 - 5(k\tau)^2 +$

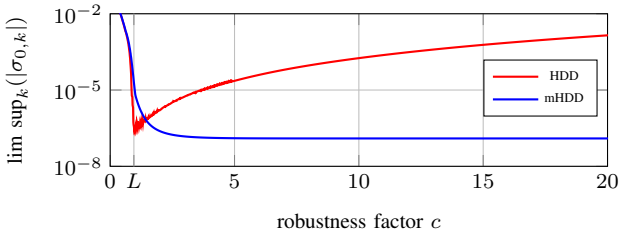

Fig. 5: Maximum observer error w.r.t. an increase of c

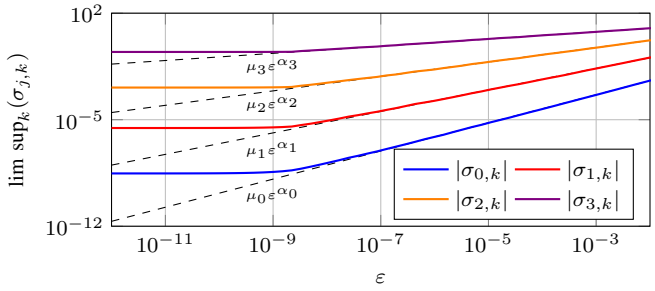

Fig. 6: Estimation accuracy of the proposed differentiator w.r.t. noise.

$2(k\tau)$ where η_k represents uniformly distributed random numbers such that $\eta_k \in \frac{1}{2}[-\varepsilon, \varepsilon]$. The order of the differentiator is $n = 3$ and $c = 2.6$. Fig. 6 shows the precision obtained versus the noise level ε. The black dashed lines represent interpolations of the obtained data. Neglecting the effect of sampling, one expects from theoretical considerations that the steady state estimation error $|\sigma_{j,k}| < \mu_j \varepsilon^{1-\frac{j}{n+1}}$ for positive constants μ_j. From interpolation using data corresponding to $\varepsilon > 10^{-7}$ and the interpolant $\mu_j\varepsilon^{\alpha_j}$, it is obtained that: $\alpha_0 = 0.92$, $\alpha_1 = 0.70$, $\alpha_2 = 0.47$, $\alpha_3 = 0.23$ which is close to the theoretically expected accuracy orders. The computed constants are $\mu_0 = 0.25$, $\mu_1 = 2.9$, $\mu_2 = 14.1$, $\mu_3 = 30.1$.

A. Laboratory Application

Results obtained by applying both versions of the toolbox in a real world application are presented in this paragraph. The task is to estimate the velocity as well as the acceleration of a vertically moving mechanical platform by differentiating the measured position signal. The position of the platform is measured by a potentiometer. Real-time code is generated directly from the corresponding Simulink diagram which includes both differentiator toolboxes. The experiment is conducted for sampling times $\tau = 1$ ms and $\tau = 5$ ms. The acceleration signal is also measured by an accelerometer and is used for validation. The results are shown in the plots provided in Fig. 7. The plots on the left show the results obtained with $\tau = 1$ ms and the plots on the right show the results in the case $\tau = 5$ ms. The robustness factor is set to the same value in both versions. The value $c = 17$, which is tuned heuristically, yields suitable tracking performance, i.e. $|\sigma_{0,k}|$ is sufficiently small after some transient time. For the case $\tau = 1$ ms both versions produce satisfactory estimates, whereas increasing the sampling step size to $\tau = 5$ ms significantly diminishes the estimation precision offered by the HDD. In contrast, the discrete-time version of the differentiator obtained by the proposed discretization scheme produces considerably better results.

IV. Conclusion

A novel discretization approach applied to the robust exact differentiator facilitates the updating of the differentiator toolbox presented in [1]. An overview of this discretization technique is presented. As in the first version of the toolbox, the implementation facilitates tuning of the differentiator. It is straightforward to integrate the resulting block in existing Matlab/Simulink implementations. The automatic code generation option can also be used in conjunction with the toolbox to build source code for real-time execution on control hardware. It is shown that the new release of the toolbox provides particular advantages when sampling rates are low.

References

[1] M. Reichhartinger, S. Spurgeon, M. Forstinger, and M. Wipfler, "A robust exact differentiator toolbox for Matlab/Simulink," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 1711 – 1716, 2017, 20th IFAC World Congress.

[2] A. Levant, "Robust exact differentiation via sliding mode technique," *Automatica*, vol. 34, no. 3, pp. 379 – 384, 1998.

[3] ——, "Higher-order sliding modes, differentiation and output-feedback control," *International Journal of Control*, vol. 76, no. 9-10, pp. 924–941, 2003.

[4] A. Levant, D. Efimov, A. Polyakov, and W. Perruquetti, "Stability and robustness of homogeneous differential inclusions," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, Dec 2016, pp. 7288–7293.

[5] M. Livne and A. Levant, "Proper discretization of homogeneous differentiators," *Automatica*, vol. 50, no. 8, pp. 2007 – 2014, 2014.

[6] J. A. Moreno and M. Osorio, "Strict lyapunov functions for the super-twisting algorithm," *IEEE Transactions on Automatic Control*, vol. 57, no. 4, pp. 1035–1040, April 2012.

[7] R. Seeber and M. Horn, "Stability Proof for a Well-Established Super-Twisting Parameter Setting, accepted for publication in," *Automatica*, 2017.

[8] J. A. Moreno and M. Osorio, "A lyapunov approach to second-order sliding mode controllers and observers," in *2008 47th IEEE Conference on Decision and Control*, Dec 2008, pp. 2856–2861.

[9] F. A. Ortiz Ricardez, T. Sanchez, and J. Moreno, "Smooth lyapunov function and gain design for a second order differentiator," *IEEE Conference on Decision and Control*, pp. 5402–5407, 12 2015.

[10] E. Cruz-Zavala and J. A. Moreno, "Lyapunov functions for continuous and discontinuous differentiators," *IFAC-PapersOnLine*, vol. 49, no. 18, pp. 660 – 665, 2016, 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016.

[11] M. Reichhartinger and S. K. Spurgeon, "An arbitrary order differentiator design paradigm with adaptive gains," *to appear in: International Journal of Control*, 2018.

[12] V. Utkin, J. Guldner, and J. Shi, *Sliding Mode Control in Electro-Mechanical Systems, Second Edition* -. Boca Raton, Fla: CRC Press, 2009.

[13] H. Ghane and M. B. Menhaj, "Eigenstructure-based analysis for nonlinear autonomous systems," *IMA J. Math. Control & Information*, vol. 32, pp. 21–40, 2015.

[14] S. Koch and M. Reichhartinger, "Discrete-time equivalent homogeneous differentiators," *submitted to: 15th International Workshop on Variable Structure Systems and Sliding Mode Control, Graz, Austria*, 2018.

[15] A. Filippov, *Differential Equations with Discontinuous Righthand Sides - Control Systems*. Berlin Heidelberg: Springer Science & Business Media, 2013.

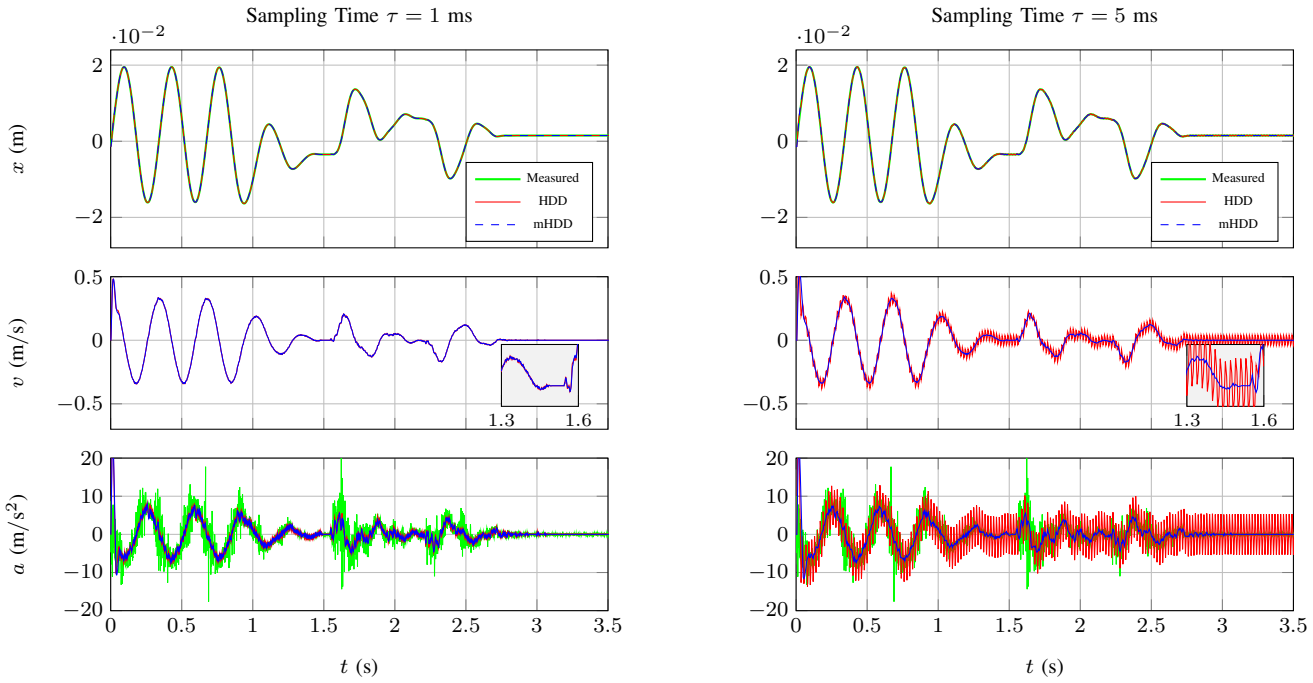

Fig. 7: Comparison of measured and estimated position and acceleration signals using the discussed discrete-time realizations of the differentiator for two different choices of τ.