



Article

Rational Approximations of Arbitrary Order: A Survey

José Daniel Colín-Cervantes ¹, Carlos Sánchez-López ^{1,*} , Rocío Ochoa-Montiel ¹ , Delia Torres-Muñoz ² , Carlos Manuel Hernández-Mejía ³ , Luis Abraham Sánchez-Gaspariano ⁴ and Hugo Gustavo González-Hernández ⁵

- ¹ Department of Electronics, Autonomous University of Tlaxcala, Clzda Apizaquito S/N, km. 1.5, Apizaco 90300, Mexico; mw_makarov@hotmail.com (J.D.C.-C.); mariadelrocio.ochoa.m@uatx.mx (R.O.-M.)
- ² Higher Technological Institute of San Martín Texmelucan, Barranca de Pesos S/N, Puebla 74120, Mexico; delets@gmail.com
- ³ Higher Technological Institute of Misantla, Loma del Cojolite S/N, km. 1.8, Misantla 93821, Mexico; cmhernandezm@itsm.edu.mx
- ⁴ Faculty of Electronics Sciences, Autonomous University of Puebla, Av. San Claudio, Puebla 72570, Mexico; luis.sanchezgas@correo.buap.mx
- ⁵ School of Engineering and Sciences, Tecnológico de Monterrey, Ave. Eugenio Garza Sada 2501, Monterrey 64849, Mexico; hgonz@tec.mx
- * Correspondence: carlsanmx@yahoo.com.mx



Citation: Colín-Cervantes, J.D.; Sánchez-López, C.; Ochoa-Montiel, R.; Torres-Muñoz, D.; Hernández-Mejía, C.M.; Sánchez-Gaspariano, L.A.; González-Hernández, H.G. Rational Approximations of Arbitrary Order: A Survey. *Fractal Fract.* **2021**, *5*, 267. <https://doi.org/10.3390/fractalfract5040267>

Academic Editors: Costas Psychalinos, Ahmed Radwan, Blas M. Vinagre and Karabi Biswas

Received: 11 November 2021
Accepted: 4 December 2021
Published: 8 December 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Abstract: This paper deals with the study and analysis of several rational approximations to approach the behavior of arbitrary-order differentiators and integrators in the frequency domain. From the Riemann–Liouville, Grünwald–Letnikov and Caputo basic definitions of arbitrary-order calculus until the reviewed approximation methods, each of them is coded in a Maple 18 environment and their behaviors are compared. For each approximation method, an application example is explained in detail. The advantages and disadvantages of each approximation method are discussed. Afterwards, two model order reduction methods are applied to each rational approximation and assist a posteriori during the synthesis process using analog electronic design or reconfigurable hardware. Examples for each reduction method are discussed, showing the drawbacks and benefits. To wrap up, this survey is very useful for beginners to get started quickly and learn arbitrary-order calculus and then to select and tune the best approximation method for a specific application in the frequency domain. Once the approximation method is selected and the rational transfer function is generated, the order can be reduced by applying a model order reduction method, with the target of facilitating the electronic synthesis.

Keywords: arbitrary-order calculus; Oustaloup's approximation; refined Oustaloup's approximation; Matsuda's approximation; continued fraction expansion approximation; Charef's approximation; Carlson's approximation; curve fitting approximation; modified stability boundary locus approximation; model order reduction

1. Introduction

Arbitrary-order calculus is an important issue that has attracted the attention of research institutes, academic associations, industrial areas and foundry companies. It has shown a better way of modeling and controlling many financial, biological, chemical, physical, electrical and control phenomena [1–3]. Capacitors and inductors are dissipative two-terminal passive elements widely used to describe integer-order differentiators, integrators and state-space systems. However, to date, there is no robust passive element with arbitrary characteristics so that it can be used to describe the behavior of a system of arbitrary-order differential equations that models natural phenomena or human-made systems more realistically. In response to this problem, integer-order rational approximations in the frequency domain were developed to approximate the arbitrary order with low level of error and wide bandwidth, such as Oustaloup's [4], refined Oustaloup's [5], Charef's [6],

Carlson's [7,8], Matsuda's [9], etc. Examples of its use in several applications can be found in the literature. For instance, a synthesis methodology of arbitrary-order chaotic systems was discussed in [10]. The analysis and analog design of arbitrary-order charge/flux controlled memristor emulators of incremental/decremental type was described in [11]. In [3,12,13] the analog design of arbitrary-order proportional-integral-derivative controllers was reported. The arbitrary-order lead/lag compensator was investigated in [14,15]. The design of double exponent arbitrary-order filters and power law filters were investigated in [16,17], respectively, and so on. Note that in each application, a different arbitrary-order approximation method was used; once the rational approximation is deduced, this can be synthesized using analog or digital platforms. Therefore, this survey not only describes and analyzes the main approximation methods reported to date, but two model order reduction methods are also reviewed in order to obtain rational transfer functions of reduced order and with the target of simplifying the electronic synthesis process. Unlike [18,19] and to help beginners get started quickly and learn arbitrary-order calculus, the corresponding Maple 18 code of each approximation method is herein reported.

The paper is organized as follows. Section 2 deals with the three main basic definitions on arbitrary-order calculus. The Maple code for each of them is also introduced in this section. Section 3 describes nine rational approximation methods in the frequency domain. Further, the Maple 18 code for each approximation method is not only presented, but an example of application by each method is also developed. The advantages and disadvantages of each method are also discussed. Section 4 deals with the analysis of two model order reduction methods. Examples for each method are also discussed in this section, showing the drawbacks and benefits. Finally, the results and conclusions are given in Section 5.

2. Basic Definitions

2.1. Arbitrary-Order Integral and Derivative of Riemann–Liouville

The arbitrary-order integral in the sense of Riemann–Liouville is defined as [1,3]

$${}_a J_t^\alpha f(t) = \frac{1}{\Gamma(\alpha)} \int_a^t \frac{f(\tau)}{(t-\tau)^{1-\alpha}} d\tau \quad (1)$$

where a and t are the lower and upper limits of integration, $\Gamma(\cdot)$ is the gamma function, $\alpha \in \mathbb{R}^+$ is the arbitrary order and $f(\tau)$ describes a causal function of time. This first basic definition can easily be code in Maple 18 environment, as described in Table 1.

Table 1. Maple 18 code of (1).

```
restart;
assume(alpha <= 1); additionally(0 < alpha);
AOI_RL := proc(alpha,f)
1/GAMMA(alpha)*int(f*(t-tau)^(alpha-1),tau = 0..t,'AllSolutions') assuming t > 0;
end proc;
```

Let us consider an example on the use of Table 1. Defining $f := \tau^2 - 2\tau + 1$ and $\alpha := 0.65$ we compute the arbitrary-order integral as

$$\begin{aligned} S1 &:= \text{AOI_RL}(\alpha, f); \\ S1 &:= 0.5081609661t^{(53/20)} - 1.346626560t^{(33/20)} + 1.110966912t^{(13/20)} \end{aligned} \quad (2)$$

making a variable change and defining $\alpha = 0.35$ we have

$$\begin{aligned}
 & \text{alpha} := 0.35 : t := \text{tau} : \text{Sa} := \text{S1}; \text{unassign}('t'); \\
 & \text{Sa} := 0.5081609661\tau^{(53/20)} - 1.346626560\tau^{(33/20)} + 1.110966912\tau^{(13/20)} \\
 & \text{S2} := \text{AOI_RL}(\text{alpha}, \text{Sa}); \\
 & \text{S2} := 0.333333334t^3 - t^2 + t
 \end{aligned} \tag{3}$$

the last equation in (3) is the first-order integral of the original function $f(\tau)$ described above and was found by applying (1) twice. Despite the ease of application, the main disadvantage of this basic definition is that the analytical solution of complex functions cannot be found easily and the solution is often a function of other nested functions. Moreover, the inverse process of (1), i.e., the arbitrary-order derivative, is defined as

$$D^\alpha f(t) = \frac{d^p}{dt^p} \left[\frac{1}{\Gamma(p - \alpha)} \int_a^t \frac{f(\tau)}{(t - \tau)^{1+\alpha-p}} d\tau \right] \tag{4}$$

where p is a positive integer such that $p - 1 < \alpha \leq p$. Based on the previous programming code, (4) is coded in Table 2.

Table 2. Maple 18 code of (4).

```

restart:
assume(p-1 < alpha); additionally(alpha <= p);
AOD_RL := proc(alpha, f, p)
diff(1/GAMMA(p-alpha)*int((t-tau)^(p-alpha-1)*f, tau = 0..t), t$ p) assuming t >= 0;
end proc;

```

Let us consider an example on the use of Table 2. Defining $f := \frac{\tau^3}{3} - \tau^2 + \tau$, $p := 1$ and $\alpha := 0.15$ we get

$$\begin{aligned}
 & \text{S1} := \text{AOD_RL}(\text{alpha}, f, p); \\
 & \text{S1} := 0.4011444710t^{(57/20)} - 1.143261742t^{(37/20)} + 1.057517112t^{(17/20)}
 \end{aligned} \tag{5}$$

making, again, a variable change and updating $\alpha = 0.85$, we have

$$\begin{aligned}
 & \text{alpha} := 0.85 : t := \text{tau} : \text{Sa} := \text{S1}; \text{unassign}('t'); \\
 & \text{Sa} := 0.4011444710\tau^{(57/20)} - 1.143261742\tau^{(37/20)} + 1.057517112\tau^{(17/20)} \\
 & \text{S2} := \text{AOD_RL}(\text{alpha}, \text{Sa}, p); \\
 & \text{S2} := 0.99999t^2 - 2t + 1
 \end{aligned} \tag{6}$$

the last equation in (6) is the first-order derivative of $f(\tau)$ mentioned before and going back to the original function used in the first example. Note that (4) was also applied twice. Aside from the aforementioned disadvantage of limiting to simple functions, another disadvantage of (4) is that the initial conditions of the test function cannot be considered. This is a serious problem from a practical engineering standpoint.

2.2. Arbitrary-Order Integral and Derivative of Grünwald–Letnikov

The arbitrary-order integral in the sense of Grünwald–Letnikov is defined as [1,3]

$${}_a J_t^\alpha f(t) = \lim_{h \rightarrow 0} h^\alpha \sum_{p=0}^{\frac{t-a}{h}} \frac{\Gamma(\alpha + p)}{p! \Gamma(\alpha)} f(t - ph) \tag{7}$$

where h is the integration step. This third basic definition is coded in Maple 18 environment as illustrated in Table 3. Defining $\alpha = 0.65$, $h = 0.01$, $tf = 3$, $a = 0$ and $f := t^2 - 2t + 1$, we got the numerical behavior of arbitrary-order integral of the function as

```
alpha := 0.65 : h := 0.01; tf := 3 : a := 0 :
f := subs(t = t - p * h, f) : f := unapply(f, t, p, h) :
S1 := AOI_GL(alpha, h, tf, a) :
```

(8)

the second line of (8) defines a variable change in f . Because (7) is a discrete function, the time-domain response of complex functions can numerically be computed. However, a level of error is glimpsed compared with its continuous part. This error can be reduced, making $h \rightarrow 0$, but the CPU time increases. Therefore, there is a trade-off between accuracy and speed. Another disadvantage is related with the memory length since (7) is an iterative procedure and the sum in its scheme becomes longer and longer. Note that whereas the analytical solution is given at the second line of (2), its numerical response is shown in Figure 1 (blue-line).

Table 3. Maple 18 code of (7).

```
restart:
f := subs(t = t - p*h, f): f := unapply(f, t, p, h):
AOI_GL := proc(alpha, h, tf, a)
local t, p, Sa, S1 := []:
for t from 0 by h to tf do
  Sa := 0:
  for p from 0 by 1 to (t-a)/h do
    Sa := Sa + h^alpha * GAMMA(alpha + p) / (p! * GAMMA(alpha)) * f(t, p, h):
  od:
  S1 := [op(S1), [t, Sa]]:
od: return(S1):
end proc:
```

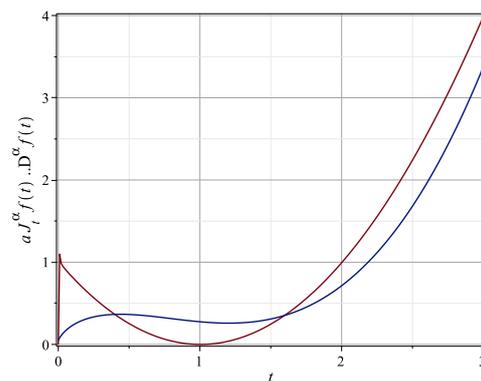


Figure 1. Time-domain response of (8) (blue line) and (10) (red line).

Moreover, the arbitrary-order derivative is defined as

$$D^\alpha f(t) = \lim_{h \rightarrow 0} \frac{1}{h^\alpha} \sum_{p=0}^{\frac{t-a}{h}} (-1)^p \frac{\Gamma(\alpha + 1)}{p! \Gamma(1 + \alpha - p)} f(t - ph) \quad (9)$$

By virtue of the form of (9) and according to Table 3, the Maple 18 code is given in Table 4. Defining $\alpha = 0.65$, $h = 0.01$, $tf = 3$, $a = 0$ and $f := 0.5081609661 * t^{(53/20)} -$

$1.346626560 * t^{(33/20)} + 1.110966912 * t^{(13/20)}$, we obtain the numerical behavior of arbitrary-order derivative of the function as

$$\begin{aligned} & \text{alpha} := 0.65 : h := 0.01; tf := 3 : a := 0 : \\ & f := \text{subs}(t = t - p * h, f) : f := \text{unapply}(f, t, p, h) : \\ & S1 := \text{AOD_GL}(\text{alpha}, h, tf, a) : \end{aligned} \quad (10)$$

The time-domain numerical response is shown in Figure 1 (red-line). Apart from the disadvantages mentioned above, another drawback of (9) is related with the initial conditions of f , which cannot be included.

Table 4. Maple 18 code of (9).

```

restart:
f := subs(t = t - p*h,f): f := unapply(f,t,p,h):
AOD_GL := proc(alpha,h,tf,a)
local t,p,Sa,S1 := []:
for t from 0 by h to tf do
  Sa := 0:
  for p from 0 by 1 to (t-a)/h do
    Sa := Sa + (-1)^p/h^alpha*GAMMA(alpha + 1)/(p!*GAMMA(1 + alpha-p))*f(t,p,h):
  od:
  S1 := [op(S1),[t,Sa]]:
od:return(S1):
end proc:

```

2.3. Arbitrary-Order Derivative of Caputo

The arbitrary-order derivative in the sense of Caputo is defined as [1,3]

$$D^\alpha f(t) = \frac{1}{\Gamma(p - \alpha)} \int_0^t \frac{f^{(p)}(\tau)}{(t - \tau)^{1 + \alpha - p}} d\tau \quad (11)$$

This last basic definition is easily coded in the Maple 18 environment, as given in Table 5.

Table 5. Maple 18 code of (11).

```

restart:
assume(p - 1 < alpha);additionally(alpha <= p);
AOD_C := proc(alpha,f,p)
1/GAMMA(p-alpha)*int(diff(f,tau$p)/(t-tau)^(1 + alpha-p),tau = 0..t) assuming t >= 0;
end proc:

```

Let us consider an example on the use of Table 5. Defining $f := \frac{\tau^3}{3} - \tau^2 + \tau$, $p := 1$ and $\alpha := 0.15$ we compute the arbitrary-order derivative as

$$\begin{aligned} & S1 := \text{AOD_C}(\text{alpha}, f, p); \\ & S1 := 0.4011444709t^{(57/20)} - 1.143261742t^{(37/20)} + 1.057517111t^{(17/20)} \end{aligned} \quad (12)$$

making, again, a variable change and updating $\alpha = 0.85$, we have

$$\begin{aligned} & \text{alpha} := 0.85 : t := \text{tau} : \text{Sa} := S1; \text{unassign}(t); \\ & \text{Sa} := 0.4011444709\tau^{(57/20)} - 1.143261742\tau^{(37/20)} + 1.057517111\tau^{(17/20)} \\ & S2 := \text{AOD_C}(\text{alpha}, \text{Sa}, p); \\ & S2 := 0.99999999t^2 - 2t + 0.99999997 \end{aligned} \quad (13)$$

The last equation of (13) is the first-order derivative of $f(\tau)$ mentioned before, where (11) was also applied twice. A main advantage of this basic definition is that the initial conditions of the test function can be considered, i.e., $f(0), f'(0), \dots, f^{p-1}(0)$. However, the analytical solution of complex functions cannot be easily computed, and the solution is again a function of other nested functions.

Moreover, natural phenomena are often modeled by state-space representation, which is a system of integer- or arbitrary-order differential equations [2]. Once the physical phenomena are modeled, the following task is to find their solution. From an engineering point of view, there are two ways to solve the state-space system: frequency-domain methods [18] and time-domain methods [20]. Note that the main basic definitions of arbitrary-order calculus are of the second type. Other reported methods in the time domain are the Adomian decomposition method [21], predictor–corrector approach based on one-step Adams–Bashforth–Moulton method [22–25] and Chebyshev collocation method [26,27]. However, the main disadvantage of all these methods is that numerical simulations can only be performed, and some of them are not robust enough to be encoded on reconfigurable hardware. In this context, frequency-domain methods are based on the approximation of the transition function $\frac{1}{s^\alpha}$ through pole-zero pairs. Although this type of approximation has two main drawbacks—(1) the approximations of $\frac{1}{s^\alpha}$ is valid only for few values of α and (2) the difference between the original behavior and approximate is still unknown—its major advantage is that integer-order rational polynomials can be synthesized with discrete analog circuits or encoded in reconfigurable hardware and, as a consequence, experimental results can be observed.

3. Rational Approximations in the Frequency Domain

In the analog domain, the expression $Z(s) = \frac{k_0}{s^\alpha}$, where k_0 is a constant called fractance device. Several frequency-domain approximation methods have been reported in the literature that can be used to compute the integer-order rational approximations of the fractance, as described below.

3.1. Oustaloup's Approximation

This approximation method is based on a recursive distribution of zeros and poles into a frequency interval (w_l, w_h) , where w_l and w_h are the low and high transitional frequencies expressed in rad/s [4]. Thus, the target is to approximate the function of the form

$$H(s) = s^\alpha, \alpha \in \mathfrak{R}^+ \quad (14)$$

by an integer-order rational approximation

$$s^\alpha = w_h^\alpha \prod_{k=1}^N \frac{s + z_k}{s + p_k}, \quad 0 < \alpha < 1 \quad (15)$$

The zeros are computed as

$$z_k = w_l \left(\frac{w_h}{w_l} \right)^{\frac{2k-1-\alpha}{2N}} \quad (16)$$

and the poles are computed as

$$p_k = w_l \left(\frac{w_h}{w_l} \right)^{\frac{2k-1+\alpha}{2N}} \quad (17)$$

where N is the approximation order. Equations (15)–(17) are coded in Maple 18 as given in Table 6.

Table 6. Oustaloup's approximation coded in Maple 18 environment.

```

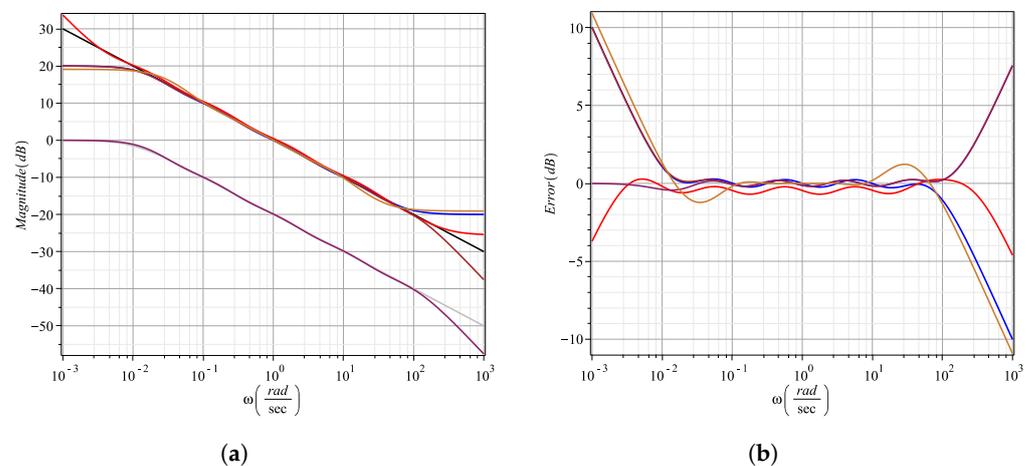
restart:Digits := 6:
Oustaloup := proc(alpha,wl,wh,N)
local p,z,Ha;
z := wl*(wh/wl)^((2*k - 1 - alpha)/(2*N)):
p := wl*(wh/wl)^((2*k - 1 + alpha)/(2*N)):
Ha := wh^alpha*factor(product((s + z(k))/(s + p(k)),k = 1..N)):
return(Ha):
end proc:

```

Let us consider an example on the use of Table 6. Defining $\alpha = 0.5$, $w_l = 0.01$, $w_h = 100$ and $N = 4$, the rational approximation is computed as

$$\begin{aligned}
 & \text{alpha} := 0.5 : \text{wl} := 0.01 : \text{wh} := 100 : \text{N} := 4 : \\
 & \text{Ha} := \text{Oustaloup}(\text{alpha}, \text{wl}, \text{wh}, \text{N}); \\
 & \text{Ha} := 10.0 \frac{(s + 0.0177828)(s + 0.177828)(s + 1.77828)(s + 17.7828)}{(s + 0.0562341)(s + 0.562341)(s + 5.62341)(s + 56.2341)} \quad (18) \\
 & \text{Ha} := \text{expand}(\text{numer}(\text{Ha})) / \text{expand}(\text{denom}(\text{Ha})); \\
 & \text{Ha} := \frac{10s^4 + 197.567s^3 + 354.523s^2 + 62.4761s + 1}{s^4 + 62.4761s^3 + 354.522s^2 + 197.566s + 9.99994}
 \end{aligned}$$

the third line of (18) expresses the rational approximation as a product of poles and zeros, and the fifth line as a quotient of fourth-order rational polynomials. Note that here s^α was approximated by (18), that is, the arbitrary-order derivative. To obtain $1/s^\alpha$, i.e., the arbitrary-order integral, the numerator and denominator of (18) must be interchanged. In this sense, whereas the magnitude and phase responses of the arbitrary-order integral approximation along with the ideal behavior (black line) are shown in Figure 2a,c, the errors for each case are depicted in Figure 2b,d, respectively. All magnitude and phase waveforms along with their errors are colored with a blue line. Advantages of this approximation method are that (1) it allows to obtain rational polynomials in a wide range of frequencies and (2) for the same iteration number, the rational approximation has a lower order compared with other approximation methods. However, its major disadvantage is that the phase error does not decrease in the frequency interval despite increasing the number of iterations. As a consequence, high-order rational approximations are generated and later, this becomes a more complex and tedious task when the rational polynomial is synthesized with discrete analog circuits or encoded with reconfigurable hardware.

**Figure 2.** Cont.

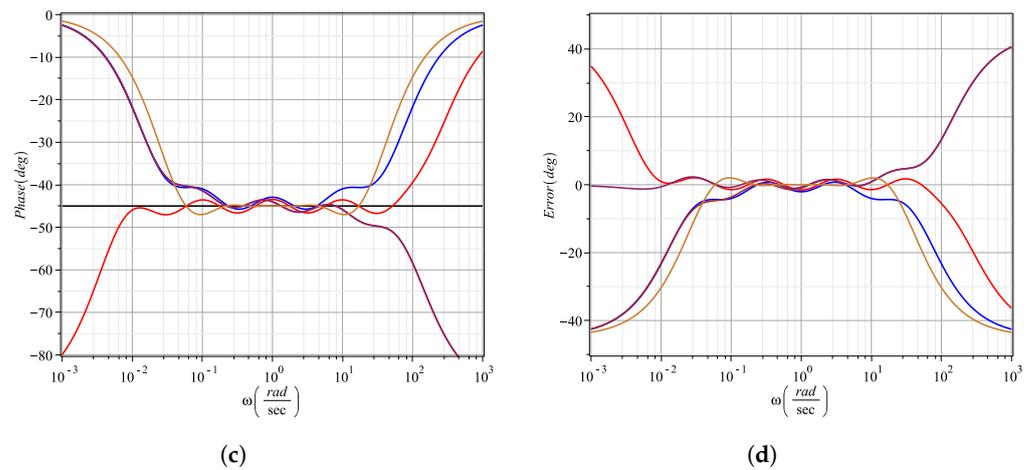


Figure 2. Ideal behavior (black line), Oustaloup’s (blue line), refined Oustaloup’s (red line), Charef’s V1 (brown line), ideal behavior (gray line) of (26), Charef’s V2 (maroon line) and Carlson’s (gold line) approximation for $\frac{1}{s^{0.5}}$. (a) Magnitude response. (b) Magnitude error. (c) Phase response. (d) Phase error.

3.2. Refined Oustaloup’s Approximation

This approximation method is defined as [5]

$$s^\alpha = \left(\frac{d}{b}w_h\right)^\alpha \frac{ds^2 + bw_h s}{d(1-\alpha)s^2 + bw_h s + d\alpha} \prod_{k=1}^N \frac{s + z_k}{s + p_k}, \quad 0 < \alpha < 1 \quad (19)$$

where $b = 10$ and $d = 9$ are adjustment parameters and their values were found by experimentation. The zeros and poles are computed by using (16) and (17). Note that in [5], a slight modification was reported compared with (19), where the bottom index of the product starts from $-N$ instead of 1, as in the original Oustaloup’s notation. Nevertheless, both approximations, reported by Equation (23) in [5] and (19) are equivalent. According to Table 6, (19) is easily coded in the Maple 18 environment as shown in Table 7.

Table 7. Refined Oustaloup’s approximation coded in Maple 18 environment.

```

restart:Digits := 6:
Ref_Oustaloup := proc(alpha,wl,wh,N)
local p,z,Ha,b := 10,d := 9:
z := wl*(wh/wl)^((2*k - 1 - alpha)/(2*N)):
p := wl*(wh/wl)^((2*k - 1 + alpha)/(2*N)):
Ha := factor((d*wh/b)^alpha*(d*s^2 + b*wh*s)/(d*(1 - alpha)*s^2 + b*wh*s + d*alpha))
*factor(product((s + z(k))/(s + p(k)),k = 1..N)):
return(Ha):
end proc:
    
```

Using $\alpha = 0.5$, $w_l = 0.01$, $w_h = 100$ and $N = 4$, the rational approximation is computed as

$$\begin{aligned}
 & \alpha := 0.5 : w_l := 0.01 : w_h := 100 : N := 4 : \\
 & Ha := Ref_Oustaloup(\alpha, w_l, w_h, N); \\
 & Ha := \frac{18.9737s(s + 111.111)(s + 0.0177)(s + 0.17782)(s + 1.77828)(s + 17.7828)}{(s + 222.21)(s + 0.0045)(s + 0.05623)(s + 0.5623)(s + 5.6234)(s + 56.234)} \quad (20) \\
 & Ha := expand(numer(Ha))/expand(denom(Ha)); \\
 & Ha := \frac{18.9737s^6 + 2483.05s^5 + 42323.5s^4 + 74859.0s^3 + 13173.1s^2 + 210.819s}{s^6 + 284.698s^5 + 14239.1s^4 + 79043.0s^3 + 44268.6s^2 + 2419.79s + 9.99994}
 \end{aligned}$$

The magnitude and phase responses of the arbitrary-order integral approximation are illustrated in Figure 2a,c while their errors are shown in Figure 2b,d. All waveforms are colored with a red line. Advantages of this approximation method are that (1) it has a better fitting quality around the frequency band boundaries in both magnitude and phase responses, as can be seen in Figure 2 (red line) and (2) for the same number of iterations, the error decreases compared with the Oustaloup's method. The main disadvantage found is that the order of the rational approximation increases, as one can see in (20), generating all the drawbacks mentioned above.

3.3. Charef's Approximation Version 1

The Charef's approximation method was reported in [6] to approximate the transfer function as a function of singularities of poles and zeros into a frequency bandwidth (w_l, w_h), which depends on an approximation error ϵ (dB) previously defined. Thus, the objective is to approximate the function

$$H(s) = \frac{1}{s^\alpha} \approx \frac{\prod_{k=0}^{N-1} \left(1 + \frac{s}{z}\right)}{\prod_{k=0}^N \left(1 + \frac{s}{p}\right)}, \quad 0 < \alpha < 1 \quad (21)$$

where the coefficients are computed for obtaining a maximum deviation according to ϵ . Thus

$$a = 10^{\frac{\epsilon}{10(1-\alpha)}}, \quad b = 10^{\frac{\epsilon}{10\alpha}}, \quad (22)$$

therefore, the distribution of the poles and zeros becomes

$$p_0 = w_l 10^{\frac{\epsilon}{20\alpha}}, \quad p = (ab)^k p_0, \quad z = a(ab)^k p_0 \quad (23)$$

and the total number of singularities is determined as

$$N = \text{integer} \left(\frac{\log\left(\frac{wh}{p_0}\right)}{\log(ab)} \right) + 1 \quad (24)$$

Here, *integer* indicates the integer part of the mathematical operation. Equations (21)–(24) are coded in Maple 18 as shown in Table 8.

Table 8. Code in Maple 18 environment of (21).

```

restart:Digits := 6:
Charef_V1 := proc(alpha,wl,wh,epsilon)
local p0,p,z,a,b,k,Ha,N;
p0 := wl*10^(epsilon/(20*alpha)):
a := 10^(epsilon/(10*(1-alpha))):
b := 10^(epsilon/(10*alpha)):
N := ceil(log10(wh/p0)/log10(a*b)):
p := (a*b)^k*p0: z := (a*b)^k*a*p0:
Ha := factor(product(1 + s/z,k = 0..N-1)/product(1 + s/p,k = 0..N))*eval(1/s^alpha,s = wl):
return(Ha):
end proc:

```

Using $\alpha = 0.5$, $w_l = 0.01$, $w_h = 100$ and $\epsilon = 2.36$, we obtain $N = 4$ and the rational approximation is computed as

$\alpha := 0.5 : w_l := 0.01 : w_h := 100 : \epsilon := 2.36 :$

$Ha := \text{Charef_V1}(\alpha, w_l, w_h, \epsilon);$

$$Ha := \frac{13.3044(s + 0.0510506)(s + 0.448744)(s + 3.94457)(s + 34.6736)}{(s + 0.0172187)(s + 0.151356)(s + 1.33045)(s + 11.6949)(s + 102.801)} \quad (25)$$

$Ha := \text{expand}(\text{numer}(Ha)) / \text{expand}(\text{denom}(Ha));$

$$Ha := \frac{13.3044s^4 + 520.442s^3 + 2076.76s^2 + 921.238s + 41.6864}{s^5 + 115.995s^4 + 1374.11s^3 + 1828.18s^2 + 273.172s + 4.16861}$$

The magnitude and phase responses of (25) are again illustrated in Figure 2a,b, and their errors are also shown in Figure 2b,d. All frequency responses are colored with a brown line. Advantages of Charef's Version 1 approximation method are that (1) for the upper frequency limit w_h , the magnitude error decreases, whereas for w_l , the magnitude behavior is similar to Oustaloup's approximation, and (2) the number of iterations depends on ϵ . Hence, the error increases or decreases in terms of α -dB. Disadvantages of the approximation method are that (1) in the frequency band boundaries, the phase error increases, as one can be seen in Figure 2d (brown line). In fact, the phase response at the lower frequency limit w_l has a similar behavior to Oustaloup's method. (2) Additionally, for the same number of iterations, the order of the rational approximation is slightly higher than Oustaloup's method, but slightly lower than the refined Oustaloup's method.

3.4. Charef's Approximation Version 2

For this method [6], the goal is to approximate the function

$$H(s) = \frac{1}{(1 + \frac{s}{w_l})^\alpha} \approx \frac{\prod_{k=0}^{N-1} (1 + \frac{s}{z})}{\prod_{k=0}^N (1 + \frac{s}{p})}, \quad 0 < \alpha < 1 \quad (26)$$

Unlike (21), an arbitrary pole is here included to model the frequency behavior of the transfer function. Note that in general, systems usually exhibits finite magnitude at very low frequency. To compute the zeros and poles, (22)–(24) are also used. Therefore, based on Table 8, (26) is coded in Maple 18 as depicted in Table 9.

Table 9. Code in Maple 18 environment of (26).

```
restart:Digits := 6;
Charef_V2 := proc(alpha,wl,wh,epsilon)
local p0,p,z,a,b,k,Ha,N;
p0 := wl*10^(epsilon/(20*alpha));
a := 10^(epsilon/(10*(1-alpha)));
b := 10^(epsilon/(10*alpha));
N := ceil(log10(wh/p0)/log10(a*b));
p := (a*b)^k*p0: z := (a*b)^k*a*p0:
Ha := factor(product(1 + s/z,k = 0..N - 1)/product(1 + s/p,k = 0..N)):
return(Ha):
end proc:
```

Using $\alpha = 0.5$, $w_l = 0.01$, $w_h = 100$ and $\epsilon = 2.36$, we obtain $N = 4$ and the rational approximation is computed as

$\alpha := 0.5 : w_l := 0.01 : w_h := 100 : \epsilon := 2.36 :$

$Ha := \text{Charef_V2}(\alpha, w_l, w_h, \epsilon);$

$$Ha := \frac{1.33044(s + 0.0510506)(s + 0.448744)(s + 3.94457)(s + 34.6736)}{(s + 0.0172187)(s + 0.151356)(s + 1.33045)(s + 11.6949)(s + 102.801)} \quad (27)$$

$Ha := \text{expand}(\text{numer}(Ha)) / \text{expand}(\text{denom}(Ha));$

$$Ha := \frac{1.33044s^4 + 52.0442s^3 + 207.676s^2 + 92.1238s + 4.16864}{s^5 + 115.995s^4 + 1374.11s^3 + 1828.18s^2 + 273.172s + 4.16861}$$

According to Figure 2 (maroon line), one can observe that for w_h the frequency behavior of the magnitude and phase error responses is equal to Charef's Version 1 approximation, whereas for w_l , the magnitude and phase errors have better behavior in comparison with Charef's Version 1 method. Another advantage of this method is that the number of iterations depends on ϵ and, as a consequence, the error is controlled in terms of α -dB. The disadvantages of the approximation method are the same as those described in Charef's Version 1 approximation method. Comparing (25) and (27), it can be seen that the only difference is the gain of the transfer functions.

3.5. Carlson's Approximation

The method proposed in [7] was derived from a regular Newton process used for iterative approximation of the α -th root. The starting point of the method is defined as

$$H(s) = G(s)^\alpha \quad (28)$$

Defining the initial value as $H_0(s) = 1$, the approximated rational function is obtained as

$$H_k(s) = H_{k-1}(s) \frac{(1-\alpha)H_{k-1}(s)^{\frac{1}{\alpha}} + (1+\alpha)G(s)}{(1+\alpha)H_{k-1}(s)^{\frac{1}{\alpha}} + (1-\alpha)G(s)} \quad \forall k = 1 \dots N \quad (29)$$

where $G(s) = s$ defines the derivative and $G(s) = \frac{1}{s}$ defines the integral. The code in Maple 18 environment of (29) is written in Table 10.

Table 10. Code in Maple 18 environment of (29).

```
restart:Digits := 6;
Carlson := proc(alpha,G,N)
local k,Ha := 1;
for k from 1 to N do
  Ha := Ha*((1-alpha)*Ha^(1/alpha) + (1+alpha)*G)/((1+alpha)*Ha^(1/alpha) + (1-alpha)*G);
od;
return(simplify(expand(numer(Ha))/expand(denom(Ha))));
end proc;
```

For this case, we use $\alpha = \frac{1}{2}$, $G(s) = \frac{1}{s}$ and $N = 2$. The rational approximation is obtained as

$$\begin{aligned} & \text{alpha} := 1/2 : G := 1/s : N := 2 : \\ & \text{Ha} := \text{Carlson}(\text{alpha}, G, N); \\ & \text{Ha} := \frac{s^4 + 36s^3 + 126s^2 + 84s + 9}{9s^4 + 84s^3 + 126s^2 + 36s + 1} \\ & \text{PrintSystem}(\text{ZeroPoleGain}(\text{evalf}(\text{Ha}))); \\ & \text{Ha} := \frac{0.111111(s + 32.1634)(s + 3)(s + 0.704088)(s + 0.132474)}{(s + 7.54863)(s + 1.42028)(s + 0.333333)(s + 0.0310912)} \end{aligned} \quad (30)$$

the fourth line in (30) gets the zeros, poles and gain of the function described in the third line and hence, the approximated function is built in terms of poles and zeros, as described in the fifth line. The magnitude and phase responses of the arbitrary-order integral approximation are depicted in Figure 2a,c and the error for each case is shown in Figure 2b,d. Here, a gold color line is used to plot the waveform at each case. The main advantage of this approximation method is that the magnitude and phase errors are almost zero within a narrow bandwidth. However, if one wants to enlarge the bandwidth, it is necessary to increase N . However, if $N = 3$, the order of the generated polynomials is 13 and for $N = 4$ the order increases to 40, which is infeasible from the point of view of electronic synthesis. Other disadvantages found are that (1) there is no control over the frequency

interval of interest (w_l, w_h) , and (2) for some values of $0 < \alpha \leq 1$ with one or two digits, the method generates terms of s raised to a rational exponent. For instance, for $\alpha = \frac{9}{10}$ one obtains

$$H_a := \frac{\left(\frac{s+19}{19s+1}\right)^{1/9} (s^3 + 38s^2 + 361s) + 361s^2 + 6878s + 361}{\left(\frac{s+19}{19s+1}\right)^{1/9} (361s^3 + 6878s^2 + 361s) + 361s^2 + 38s + 1} \quad (31)$$

which becomes a difficult or impossible task during the electronic synthesis process. One possible solution is to compute several rational transfer functions with α different [8]. Thus, we can generate $\frac{1}{s^{9/10}} = \frac{1}{s^{5/10}} \frac{1}{s^{1/5}} \frac{1}{s^{1/5}}$, obtaining

$$H_a := \left(\frac{s^4 + 36s^3 + 126s^2 + 84s + 9}{9s^4 + 84s^3 + 126s^2 + 36s + 1} \right)^* \left(\frac{128s^7 + 2610s^6 + 11367s^5 + 22410s^4 + 23760s^3 + 13752s^2 + 3810s + 288}{288s^7 + 3810s^6 + 13752s^5 + 23760s^4 + 22410s^3 + 11367s^2 + 2610s + 128} \right)^2 \quad (32)$$

(3) If $\alpha \rightarrow 0$, then the order of the generated polynomials increases exponentially, as one can verify in the second term of (32).

3.6. Matsuda's Approximation

This approximation method was suggested in [9], which fits the original function in a set of logarithmically spaced frequencies given by $w = [w_1, w_2, w_3 \dots w_{N+1}]$ and $w_a = [|w_1^\alpha|, |w_2^\alpha|, |w_3^\alpha| \dots |w_{N+1}^\alpha|]$ and by using the following functions

$$\begin{aligned} d_{1,1} &= |w_1^\alpha| & d_{1,2} &= |w_2^\alpha| & d_{1,3} &= |w_3^\alpha| & \dots & d_{1,c} &= |w_c^\alpha| \quad \forall c = 1 \dots N+1 \\ d_{2,2} &= \frac{w_2 - w_1}{d_{1,2} - d_{1,1}} & d_{2,3} &= \frac{w_3 - w_1}{d_{1,3} - d_{1,1}} & & & & & \\ & & d_{3,3} &= \frac{w_3 - w_2}{d_{2,3} - d_{2,2}} & & & & & \\ & & & & & & & & d_{r,c} = \frac{w_c - w_{r-1}}{d_{r-1,c} - d_{r-1,r-1}} \quad \forall r = 2 \dots c \end{aligned} \quad (33)$$

getting a superior triangular matrix as

$$D = \begin{bmatrix} d_{1,1} & d_{1,2} & d_{1,3} & \dots & d_{1,N+1} \\ 0 & d_{2,2} & d_{2,3} & \dots & d_{2,N+1} \\ 0 & 0 & d_{3,3} & \dots & d_{3,N+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & d_{N+1,N+1} \end{bmatrix} \quad (34)$$

where each element of the main diagonal of D is used to recursively approximate the desired function as a continued fraction

$$H_a(s) = d_{1,1} + \frac{s - w_1}{d_{2,2} + \frac{s - w_2}{d_{3,3} + \frac{s - w_3}{d_{4,4} + \dots \frac{s - w_N}{d_{N+1,N+1}}}} \quad (35)$$

All equations mentioned before are coded in Maple 18 as given in Table 11.

Table 11. Matsuda's method coded in Maple 18 environment.

```

restart:Digits := 6:
logspace := proc(a,b,n) evalf(10^ ~ <seq(evalf(a)..evalf(b),evalf((b - a)/(n - 1),7))>) end proc:
Matsuda := proc(alpha,wl,wh,N,k)
local w,wa,d,c,r,Ha,Dp := []:
w := convert(logspace(log10(wl),log10(wh),N + 1),list):
wa := [seq(abs(w[r]^ alpha),r = 1..nops(w))]:
d := matrix(nops(w),nops(w),0):
for c from 1 to N + 1 do
d[1,c] := wa[c]:
for r from 2 to c do
d[r,c] := (w[c] - w[r - 1])/(d[r - 1,c] - d[r - 1,r - 1]):
od:
Dp := [op(Dp),d[c,c]]:
od:
Ha := op(nops(Dp),Dp):
for r from nops(Dp) - 1 by -1 to 1 do
Ha := Dp[r] + (s-w[r])/Ha:
od:return(simplify(k*Ha)):
end proc:

```

The second line of Table 11 defines the *logspace* function. As an example on the use of Table 11, we define $\alpha = -0.5$, $w_l = 0.01$, $w_h = 100$, $N = 8$ and gain $k = 1$. Therefore, the rational approximation is computed as

$$\begin{aligned}
 & \alpha := -0.5 : wl := 0.01 : wh := 100 : N := 8 : k := 1 : \\
 & Ha := Matsuda(\alpha, wl, wh, N, k); \\
 & Ha := \frac{0.456s^4 + 84.07s^3 + 607.77s^2 + 317.944s + 10.3411}{10.3406s^4 + 317.56s^3 + 608.078s^2 + 84.1832s + 0.45518} \quad (36) \\
 & PrintSystem(ZeroPoleGain(evalf(Ha))); \\
 & Ha := \frac{0.111111(s + 32.1634)(s + 3)(s + 0.704088)(s + 0.132474)}{(s + 7.54863)(s + 1.42028)(s + 0.333333)(s + 0.0310912)}
 \end{aligned}$$

The magnitude and phase responses of the approximation are depicted in Figure 3a,c, and the error for each of them is also shown in Figure 3b,d. All frequency responses are colored with a blue line. The main advantage of this approximation method is that the magnitude and phase errors are low within the desired bandwidth and still can be minimized by increasing the number of iterations N . In fact, the order of the rational transfer function is always $N/2$. For this reason, N must be even. Otherwise, improper transfer functions are generated. After several numerical simulations, we recommend using, as a starting point, $N = 8$ in order to obtain magnitude and phase errors within ± 2 dB and $\pm 6^\circ$, respectively. On the other hand, the main disadvantages found are that (1) if N is very large, then high-order rational transfer functions are generated. This fact becomes a tedious task during the electronic synthesis process. (2) Additionally, for the parameters used, we note that from $N = 16$, some positive poles are generated, hence conditioning the stability of the rational approximation.

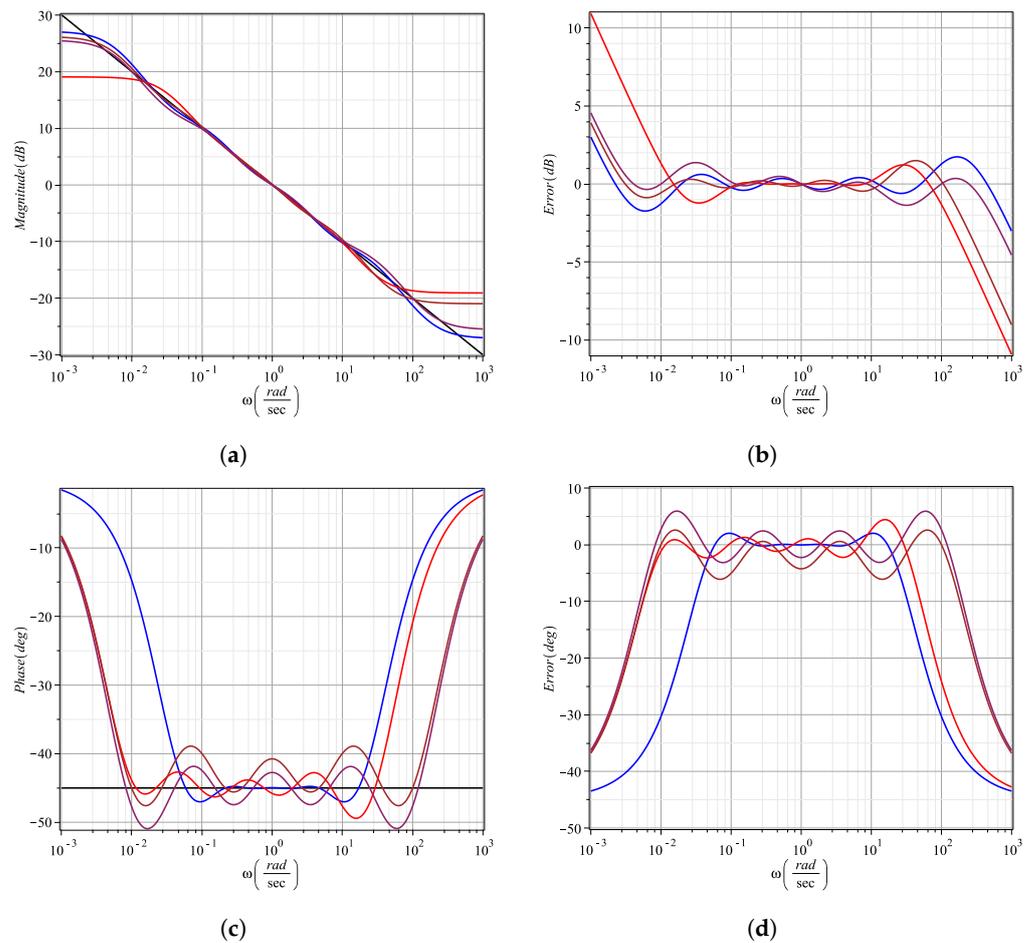


Figure 3. Ideal behavior (black line), Matsuda’s (blue line), continued fraction expansion (red line), curve fitting (brown line) and MSBL (maroon line) approximation for $\frac{1}{s^{0.5}}$. (a) Magnitude response. (b) Magnitude error. (c) Phase response. (d) Phase error.

3.7. Continued Fraction Expansion Approximation

This method is widely used for evaluation of functions since it converges faster than power series expansions within a larger domain in the complex plane [28]. Continued fraction expansion is defined as

$$(1 + x)^\alpha = \frac{1}{1 - \frac{\alpha x}{1 + \frac{(1 + \alpha)x}{2 + \frac{(1 - \alpha)x}{3 + \frac{(2 + \alpha)x}{2 + \frac{(2 - \alpha)x}{5 + \dots \frac{(N + \alpha)x}{2 + \frac{(N - \alpha)x}{2N + 1}}}}}}}} \quad (37)$$

Replacing $x = s - 1$, we can approximate s^α . To obtain the arbitrary-order integral, i.e., $\frac{1}{s^\alpha}$, the numerical value of α must be negative. Table 12 shows the code in Maple 8 of (37).

Table 12. Code in Maple 18 environment of (37).

```

restart:Digits := 6:with(DynamicSystems):with(numtheory):
CFE := proc(alpha,N)
local x,Ha:unassign('x','s'):
Ha := cfrac(cfrac((x + 1)^alpha,x,N)):x := s - 1:
return(expand(numer(eval(Ha)))/expand(denom(eval(Ha)))):
end proc:
    
```

Now, by defining $\alpha = -0.5$ and $N = 8$, the rational transfer function is computed as

$$\begin{aligned}
 & \text{alpha} := -0.5 : N := 8 : \\
 & \text{Ha} := \text{CFE}(\text{alpha}, N); \\
 & \text{Ha} := \frac{s^4 + 36s^3 + 126s^2 + 84s + 9}{9s^4 + 84s^3 + 126s^2 + 36s + 1} \\
 & \text{PrintSystem}(\text{ZeroPoleGain}(\text{evalf}(\text{Ha}))); \\
 & \text{Ha} := \frac{0.111111(s + 32.1634)(s + 3)(s + 0.704088)(s + 0.132474)}{(s + 7.54863)(s + 1.42028)(s + 0.333333)(s + 0.0310912)}
 \end{aligned} \tag{38}$$

Note that for these parameters, (38) is equal to (30), but this is not always the case. The magnitude and phase responses in the frequency domain of the rational transfer function are depicted in Figure 3a,c and the error for each case is shown in Figure 3b,d, respectively. In this case, a red line is used to plot all waveforms. After several numerical simulations, one concluded that the main advantages of this approximation method are that (1) the magnitude and phase errors are almost zero within a narrow bandwidth and (2) the order $= N/2$ of the transfer function holds, where N is always even. In order to extend the bandwidth, N must be increased. Similarly to Matsuda's method, we also recommend use, as starting point, $N = 8$ and hence, the magnitude and phase errors are ± 1.5 and $\pm 5^\circ$, approximately. Moreover, the disadvantages found are that (1) there is no control in the frequency range of interest (w_l, w_h) , (2) if N is odd, then improper transfer functions are generated, and (3) high-order rational polynomials are generated when N becomes very large and consequently affects the electronic synthesis process.

3.8. Curve-Fitting Approximation

This approximation method was reported in [29,30], which is based on obtaining the frequency response data of s^α into a set of logarithmically spaced frequencies given as $w = [w_1, w_2, w_3 \dots w_h] \in (w_l, w_h)$. Next, by iteratively applying the Sanathanan–Koerner least-squares method [31], the integer-order transfer function is approximated from the frequencies response data and defined as

$$\text{Ha}(s) = \frac{P(s)}{Q(s)} \approx \frac{\sum_{n=0}^N p_n s^n}{1 + \sum_{n=1}^N q_n s^n} \approx \frac{P\psi(s)}{1 + Q\phi(s)} \tag{39}$$

where P , Q , $\psi(s)$ and $\phi(s)$ are defined as

$$\begin{aligned}
 P &= [p_0, p_1, \dots, p_N]^T & Q &= [q_0, q_1, \dots, q_N]^T \\
 \psi(s) &= [1, s, \dots, s^N] & \phi(s) &= [s, s^2, \dots, s^N]
 \end{aligned} \tag{40}$$

The target is to obtain each coefficient of P and Q into the desired frequency range w . To do, Levy's linearized cost function along with the Sanathanan–Koerner least-squares iteration are used to minimize the difference between data samples $s^\alpha = (jw_k)^\alpha$ and (39) as

$$\arg \min_{P, Q} \sum_{k=1}^h \left| \frac{P(jw_k)}{Q^{\tau-1}(jw_k)} - \frac{Q^\tau(jw_k)}{Q^{\tau-1}(jw_k)} (jw_k)^\alpha \right|^2 \tag{41}$$

where $\tau = 1 \dots T$ is the iteration step and T is the number of iterations. The curve fitting is achieved when $Q^{\tau-1}(jw_k)$ approaches to $Q^\tau(jw_k)$, and for obtaining a minimal realization of (39), equivalent poles and zeros must be canceled. The last step is to convert the obtained state-space representation to a transfer function. Unfortunately, this method is complex to code in Maple 18, but easy to implement in the Matlab environment, due to the use of *frd* and *fitfrd* functions. However, it is possible to link Maple 18 with Matlab. The first block of Table 13 shows the Maple 18 code to link and call the Matlab script shown in the second block.

Table 13. Code in Maple 18 environment to call the m-file and Matlab script.

<pre> CF := proc(alpha,wl,wh,N) Matlab[setvar]("alpha",alpha):Matlab[setvar]("wl",wl): Matlab[setvar]("wh",wh):Matlab[setvar]("N",N): Matlab[evalM]("CF(alpha,wl,wh,N)"); end proc: </pre>
Matlab script of (39)–(41)
<pre> function Ha = CF(alpha,wl,wh,N) w = logspace(log10(wl),log10(wh)); A = (j*w).^alpha; Ha = fitfrd(frd(A,w),N); [num,den] = ss2tf(Ha.A,Ha.B,Ha.C,Ha.D); Ha = minreal(tf(num,den)) end </pre>

We now define $\alpha = -0.5$, $w_l = 0.01$, $w_h = 100$ and $N = 4$. The rational approximation is computed as

$$\begin{aligned}
& \alpha := -0.5 : wl := 0.01 : wh := 100 : N := 4 : \\
& Ha := CF(\alpha, wl, wh, N); \\
& Ha := \frac{0.08931s^4 + 4.331s^3 + 10.74s^2 + 2.692s + 0.05927}{s^4 + 9.209s^3 + 7.152s^2 + 0.5605s + 0.002899} \quad (42) \\
& PrintSystem(ZeroPoleGain(evalf(Ha))); \\
& Ha := \frac{0.0893100(s + 45.8877)(s + 2.32684)(s + 0.255132)(s + 0.0243616)}{(s + 8.36168)(s + 0.759744)(s + 0.0820100)(s + 0.00556443)}
\end{aligned}$$

The magnitude and phase responses are depicted in Figure 3a,c, and the errors are shown in Figure 3b,d. All the curves are colored with a brown line. The main advantages of this approximation method are that (1) if N is low, for instance $N = 4$, the magnitude error is low into the desired frequency range, ± 2 dB approximately, and the ripples can be minimized by increasing N . (2) If N is large, the phase error becomes very low into the frequency interval. The disadvantages found are that (1) if N is low, for instance $N = 4$, then close to the w_h upper limit, the ripple of the phase error is high, -25° approximately. (2) Additionally, when N increases, the magnitude error at w_h is slightly higher than the error at w_l . Note that if N becomes very large, then the electronic synthesis process could be a tedious and complex task.

3.9. Modified Stability Boundary Locus (MSBL) Fitting Approximation

This approximation method was reported in [19,32], which is an improved version of the stability boundaries locus fitting approximation method [33]. According to [19], the method uses a set of logarithmically spaced frequency sampling points $w_k = [w_1, w_2, w_3 \dots w_h] \in (w_l, w_h)$, for approximating the behavior of s^α . A closed-loop control system composed by the plant, i.e., s^α , and a proportional–integral controller is assumed. Following the analysis of [19,33], the equations of the method are

$$\begin{aligned}
A_{r,k} &= (jw_k)^{r+1} - (jw_k)^{N-r+1} \frac{\cos(\frac{\pi}{2}\alpha)}{w_k^\alpha} - (jw_k)^{N-r} \frac{\sin(\frac{\pi}{2}\alpha)}{w_k^{\alpha-1}} \\
&= Re(A_{r,k}) + Im(A_{r,k}) \\
B_k &= -(jw_k)^{N+1} + jw_k \frac{\cos(\frac{\pi}{2}\alpha)}{w_k^\alpha} + \frac{\sin(\frac{\pi}{2}\alpha)}{w_k^{\alpha-1}} \\
&= Re(B_k) + Im(B_k)
\end{aligned} \quad (43)$$

and the unknown coefficients of the approximate model are computed as

$$C = [a_0, a_1, \dots, a_{N-1}] = \left[\frac{1}{[A_{r,k}]^T} \cdot [B_k]^T \right]^T \quad (44)$$

and the approximated integer-order rational transfer function is given by

$$Ha(s) = \frac{a_0 s^N + a_1 s^{N-1} + \dots + a_{N-1} s + 1}{s^N + a_{N-1} s^{N-1} + \dots + a_1 s + a_0} \quad (45)$$

Equations (43)–(45) are coded in Maple 18 as shown in Table 14.

Table 14. Maple 18 code of MSBL method.

```

restart:Digits := 6:with(LinearAlgebra):
logspace := proc(a,b,n) evalf(10^~ <seq(evalf(a)..evalf(b),evalf((b-a)/(n-1),7))>) end proc:
MSBL := proc(alpha,wl,wh,N)
local A := Matrix(N),B := Vector[row](N),C,k,r,w,Ha:unassign('s'):
if N = 1 then w[1] := wh:
else w := convert(logspace(log10(wl),log10(wh),N),list): fi
for k from 1 to N do
for r from 1 to N do
A[r,k] := (I*w[k])^r-(I*w[k])^(N-r+2)*cos(Pi/2*alpha)/w[k]^alpha
-(I*w[k])^(N-r+1)*sin(Pi/2*alpha)/w[k]^(alpha-1):
A[r,k] := Re(A[r,k]) + Im(A[r,k]):
B[k] := -(I*w[k])^(N+1) + I*w[k]*cos(Pi/2*alpha)/w[k]^alpha
+sin(Pi/2*alpha)/w[k]^(alpha-1):
B[k] := Re(B[k]) + Im(B[k]):
od:
od:unassign('k'):
C := convert(((1/A)^%T.B^%T)^%T,list):
Ha := sort(sum(1/N + C[k+1]*s^(N-k),k = 0..N-1),s,descending)/
sort(sum(s^N/N + C[N-k]*s^(N-k-1),k = 0..N-1),s,descending):
return(Ha):
end proc:

```

By using $\alpha = -0.5$, $w_l = 0.01$, $w_h = 100$ and $N = 4$, the rational approximation is computed as

$$\begin{aligned}
& \alpha := -0.5 : wl := 0.01 : wh := 100 : N := 4 : \\
& Ha := MSBL(\alpha, wl, wh, N); \\
& Ha := \frac{0.05266 * s^4 + 9.46702s^3 + 68.5199s^2 + 34.5218s + 1}{s^4 + 34.5218s^3 + 68.5199s^2 + 9.46702s + 0.05266} \quad (46) \\
& PrintSystem(ZeroPoleGain(evalf(Ha))); \\
& Ha := \frac{0.0526682(s + 172.216)(s + 6.99006)(s + 0.511296)(s + 0.0308478)}{(s + 32.4172)(s + 1.95581)(s + 0.143060)(s + 0.00580665)}
\end{aligned}$$

The magnitude and phase responses are illustrated in Figure 3a,c and the errors are depicted in Figure 3b,d, respectively. For this last method, all curves are colored with a maroon line. The main advantages of this approximation method are that (1) the magnitude and phase responses are improving when N increases and for the parameters described before, the magnitude error is ± 1.5 dB and for the phase error is $\pm 7^\circ$, approximately, and (2) the method can not only be used to approximate arbitrary-order operators, but can also be used to approximate transfer functions. The main disadvantage found is that w_l should be set close to zero since if w_l drifts toward higher frequencies, then the method fails.

It is worth mentioning that other approximation methods of rational transfer functions can be found in the literature, such as the Fourier series and inverse Fourier transform method [20],

vector fitting method [34], Abdelbaki's method [35], Maione's approximation [36], Thiele's approximation [37], El-Khazali's method [38], AbdelAty's method [39]. However, the accuracy of each method is almost the same or less with some method described above. For this reason, these approximation methods are not discussed herein.

4. Model Order Reduction

From the point of view of electronic design, low-order accurate transfer functions are widely required. This is because they are easier to synthesize with analog circuits or be encoded with reconfigurable hardware. In this context, two model order reduction methods are described below.

4.1. Pade's Approximation

The Pade's method approximates a continuous function by a rational function of $[m/n]$ order [40]. Given a function $H_a(s)$ with two integers $0 \leq m, n$ and $n \leq m$, Pade's approximation is defined as

$$P(s) = \frac{\sum_{j=0}^m a_j s^j}{1 + \sum_{k=1}^n b_k s^k} = \frac{a_0 + a_1 s + a_2 s^2 + \dots + a_m s^m}{1 + b_1 s + b_2 s^2 + \dots + b_n s^n} \quad (47)$$

and have as many of their derivatives as possible equal to $s = 0$.

$$\begin{aligned} H_a(0) &= P(0) \\ H_a''(0) &= P''(0) \\ &\vdots \\ H_a^{m+n}(0) &= P^{m+n}(0) \end{aligned} \quad (48)$$

Note that for $k = 0$, the obtained approximation is the Maclaurin expansion for $H_a(s)$. In the Maple 18 environment, Pade's approximation can be computed as given in Table 15.

Table 15. Maple 18 code of Pade's method.

```
restart:Digits := 6:with(SignalProcessing):
Pade := proc(Ha,m,n)
local A,Hp:unassign('s'):
A := series(Ha,s = 0.200):
Hp := convert(A,ratpoly,m,n):
end proc:
```

Applying the algorithm of Table 15 to each rational transfer function associated with each approximation method, we obtain

$$\begin{aligned} P_{Ous}(s) &= \frac{0.100001 + 6.23548s + 34.6971s^2 + 15.7445s^3}{1 + 19.6351s + 33.0972s^2 + 2.83636s^3} \\ P_{Ref_Ous}(s) &= \frac{21.0820s + 464.965s^2 - 91901.8s^3 - 3.53478E6s^4 - 3.52981E7s^5}{1 + 201.551s - 7544.47s^2 - 717282s^3 - 1.39038E7s^4 - 5.7186E7s^5} \\ P_{Charef_V1}(s) &= \frac{10.0001 + 425.430s + 4720.84s^2 + 4514.38s^3}{0.999999 + 85.9742s + 1748.72s^2 + 7433.33s^3 - 1830.34s^4} \\ P_{Charef_V2}(s) &= \frac{1.00001 + 42.5430s + 472.084s^2 + 451.438s^3}{0.999999 + 85.9742s + 1748.72s^2 + 7433.33s^3 - 1830.34s^4} \\ P_{Carlson}(s) &= \frac{8.99996 + 68.5714s - 5.62857s^2 - 72.8571s^3}{0.999999 + 34.2857s + 65.6631s^2 - 83.4526s^3} \end{aligned} \quad (49)$$

$$\begin{aligned}
 P_{Matsuda}(s) &= \frac{22.7187 + 529.917s - 4982.07s^2 - 35993.4s^3}{0.999999 + 177.524s - 86.4722s^2 - 18084.8s^3} \\
 P_{CFE}(s) &= \frac{8.99996 + 68.5714s - 5.62857s^2 - 72.8571s^3}{0.999999 + 34.2857s + 65.6631s^2 - 83.4526s^3} \\
 P_{CF}(s) &= \frac{20.4450 + 455.702s - 16273.1s^2 - 29078.0s^3}{1 + 170.213s - 1931.47s^2 - 40335.3s^3} \\
 P_{MSBL}(s) &= \frac{18.9868 + 467.349s - 4638.09s^2 + 2698.41s^3}{0.999999 + 169.841s - 450.533s^2 - 7149.58s^3}
 \end{aligned}
 \tag{50}$$

As one can observe, from each original rational transfer function, Pade’s method is applied to reduce only an order in each polynomial. According to Figure 4a,b, the magnitude bandwidth is not only reduced, but for some approximation methods, the magnitude error is increased drastically in the desired frequency range (w_l, w_h). This same behavior is observed in the phase responses, as illustrated in Figure 4c,d. Of all of them, the Oustaloup’s, refined Oustaloup’s and Charef’s V2 methods have the least errors one decade down. Therefore, we conclude that Pade’s method fails because there is no approximation control in the frequency interval.

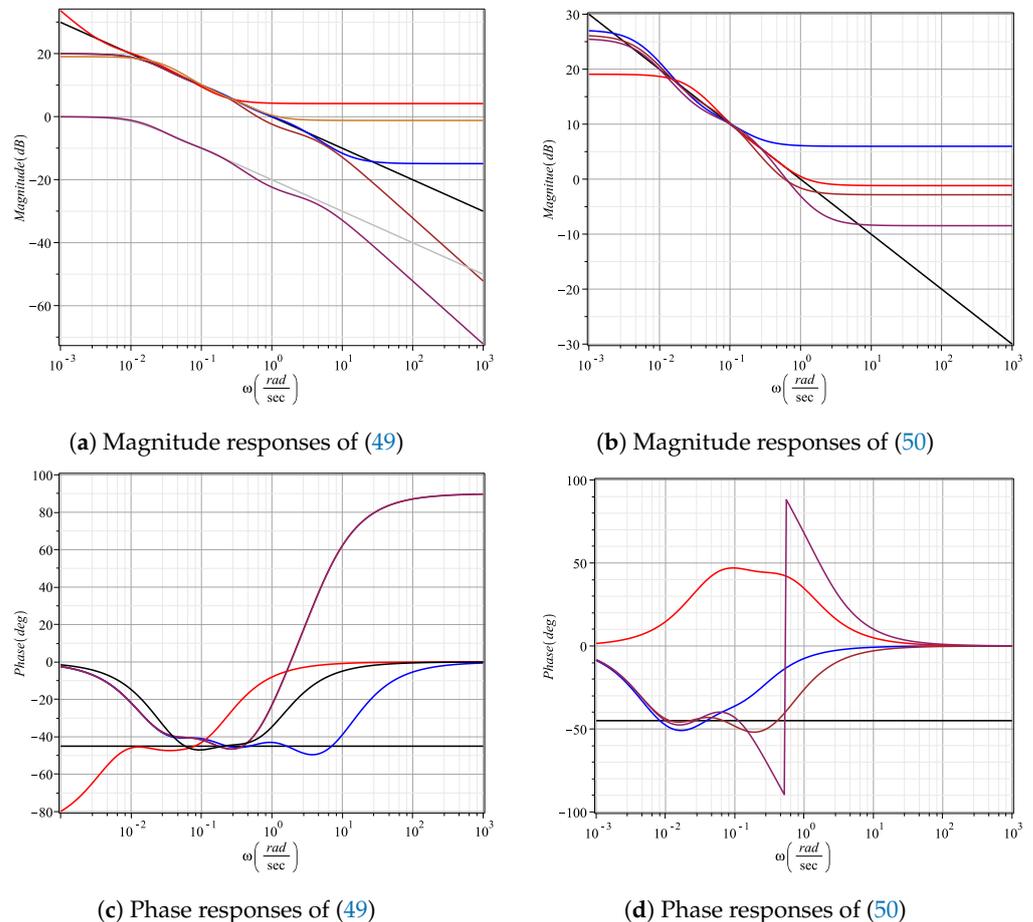


Figure 4. Magnitude and phase responses of reduced order transfer functions by applying Pade’s approximation. (a) Ideal behavior (black line) of $\frac{1}{s^{0.5}}$, Oustaloup’s (blue line), refined Oustaloup’s (red line), Charef’s V1 (brown line), ideal behavior (gray line) of $\frac{1}{1+s^{0.5}}$, Charef’s V2 (maroon line) and Carlson’s (gold line), (b) ideal behavior (black line) of $\frac{1}{s^{0.5}}$, Matsuda’s (blue line), continued fraction expansion (red line), curve fitting (brown line) and MSBL (maroon line), (c) phase responses of (a,d), phase responses of (b).

4.2. Stochastic Balancing Method

The *balred* function of Matlab defines the stochastic balancing related model reduction method [41,42]. Applying this function to each rational transfer function associated with each approximation method as

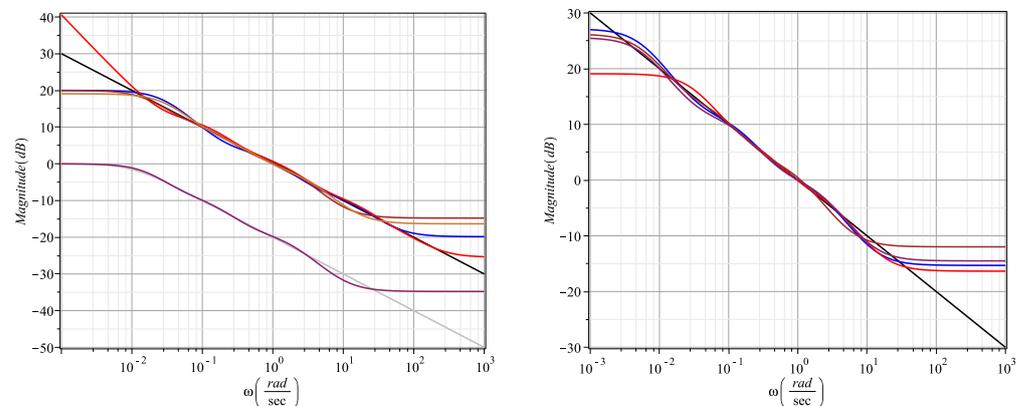
$$Pr = \text{balred}(Ha, R, \text{FreqIntervals}', [wl\ wh]) \quad (51)$$

where Ha is the original transfer function, Pr is the reduced order transfer function and R is the order, we obtain

$$\begin{aligned} Pr_{Ous}(s) &= \frac{9.827s^3 + 151.5s^2 + 143.8s + 4.139}{s^3 + 55.41s^2 + 213.2s + 41.39} \\ Pr_{Ref_Ous}(s) &= \frac{18.77s^5 + 2142s^4 + 33580s^3 + 49910s^2 + 5988s}{s^5 + 2612s^4 + 11780s^3 + 58880s^2 + 25540s + 640.1} \\ Pr_{Charef_v1}(s) &= \frac{0.1826s^3 + 2.012s^2 + 1.052s + 0.04883}{s^3 + 1.99s^2 + .3172s + 0.004883} \\ Pr_{Charef_v2}(s) &= \frac{0.01826s^3 + 0.2012s^2 + 0.1052s + 0.004883}{s^3 + 1.99s^2 + 0.3172s + 0.004883} \\ Pr_{Carlson}(s) &= \frac{0.1524s^3 + 2.604s^2 + 2.737s + 0.3217}{s^3 + 3.518s^2 + 1.257s + 0.03574} \end{aligned} \quad (52)$$

$$\begin{aligned} Pr_{Matsuda}(s) &= \frac{0.1716s^3 + 2.196s^2 + 1.218s + 0.03979}{s^3 + 2.297s^2 + 0.3237s + 0.001751} \\ Pr_{CFE}(s) &= \frac{0.1524s^3 + 2.604s^2 + 2.737s + 0.3217}{s^3 + 3.518s^2 + 1.257s + 0.03574} \\ Pr_{CF}(s) &= \frac{0.2519s^3 + 1.52s^2 + 0.4432s + 0.009953}{s^3 + 1.129s^2 + 0.09365s + 0.0004868} \\ Pr_{MSBL}(s) &= \frac{0.1884s^3 + 2.262s^2 + 1.199s + 0.03487}{s^3 + 2.35s^2 + 0.3298s + 0.001836} \end{aligned} \quad (53)$$

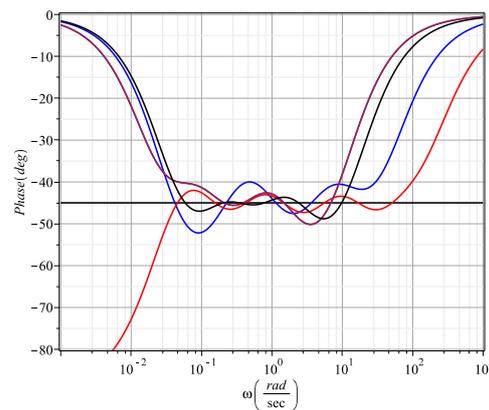
Similarly as before, the stochastic balancing method is applied to reduce an order in each numerator and denominator of the original transfer function. Thus, whereas the magnitude responses of (52) are depicted in Figure 5a, those of (53) are illustrated in Figure 5b. At this stage, one can observe that for Oustaloup's and refined Oustaloup's methods, the bandwidth is slightly kept in the desired frequency range. However, for phase responses, the bandwidth of all approximation methods is reduced, as shown in Figure 5c,d.



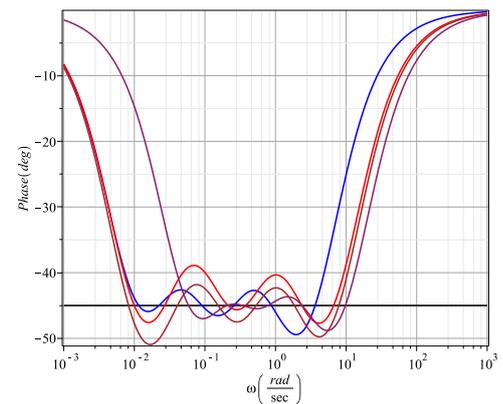
(a) Magnitude responses of (52).

(b) Magnitude responses of (53).

Figure 5. Cont.



(c) Phase responses of (52).



(d) Phase responses of (53).

Figure 5. Magnitude and phase diagrams of reduced order transfer functions by applying stochastic balancing method. (a) Ideal behavior (black line) of $\frac{1}{s^{0.5}}$, Oustaloup's (blue line), refined Oustaloup's (red line), Charef's V1 (brown line), ideal behavior (gray line) of $\frac{1}{1+s^{0.5}}$, Charef's V2 (maroon line) and Carlson's (gold line), (b) ideal behavior (black line) of $\frac{1}{s^{0.5}}$, Matsuda's (blue line), continued fraction expansion (red line), curve fitting (brown line) and MSBL (maroon line), (c) phase responses of (a,d), phase responses of (b).

5. Results and Conclusions

Three basic definitions of arbitrary-order calculus were revised, and the Maple 18 code for each of them along with an application example were shown. Afterwards, the frequency-domain approximation methods were introduced. For each approximation method, not only the Maple 18 code was presented, but an application example was also explained, where the closed form transfer functions were derived for $\alpha = 1/2$. The advantages and disadvantages of each approximation method were also discussed. From the frequency response waveforms, it is evident that curve fitting and MSBL methods are better suited to the ideal behavior. On the one hand, if the required bandwidth for a specific application is narrow, then continued fraction expansion or Matsuda's method can be used. On the other hand, if the application needs to control the approximation error, then Charef's V1 and V2 methods should be used. Furthermore, we also note that in general, the accuracy of the approximation methods depends on N (ϵ for Charef's V1 and V2) and the desired frequency interval. Therefore, for a large bandwidth, N should be increased to obtain the desired accuracy, but if the bandwidth is narrow, then N can be reduced. For all approximation methods, the rational transfer function as a product of poles and zeros was also derived. This is important because each pole-zero pair can be synthesized in the analog domain by using first-order shelving equalizers, and for the case of a pole with gain, a low-pass filter topology can be used. Finally, we showed two model order reduction methods and according to the numerical results, the stochastic balancing method achieves the best reduced-order transfer functions, despite all its disadvantages. As a consequence, the electronic synthesis process will be accelerated.

Author Contributions: Conceptualization, J.D.C.-C. and C.S.-L.; methodology, J.D.C.-C., C.S.-L., R.O.-M., D.T.-M. and C.M.H.-M.; software, J.D.C.-C., C.S.-L. and L.A.S.-G.; validation, L.A.S.-G., H.G.G.-H. and C.S.-L.; formal analysis, J.D.C.-C., C.S.-L., R.O.-M., D.T.-M., C.M.H.-M., L.A.S.-G. and H.G.G.-H.; investigation, J.D.C.-C., C.S.-L., R.O.-M., D.T.-M., C.M.H.-M., L.A.S.-G. and H.G.G.-H.; writing—original draft preparation, C.S.-L., L.A.S.-G. and H.G.G.-H.; writing—review and editing, C.S.-L., R.O.-M. and D.T.-M.; supervision, C.S.-L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Oldham, K.B.; Spanier, J. *The Fractional Calculus: Theory and Applications of Differentiation and Integration to Arbitrary Order*, 1st ed.; Dover Publications: Mineola, NY, USA, 2006.
- Sun, H.; Zhang, Y.; Baleanu, D.; Chen, W.; Chen, Y. A new collection of real world applications of fractional calculus in science and engineering. *Commun. Nonlinear Sci. Numer. Simul.* **2018**, *64*, 213–223. [[CrossRef](#)]
- Petráš, I. *Fractional-Order Nonlinear Systems, Modeling, Analysis and Simulation*, 1st ed.; Springer: London, UK; New York, NY, USA, 2011.
- Oustaloup, A.; Levron, F.; Mathieu, B.; Nanot, F.M. Frequency band complex noninteger differentiator: characterization and synthesis. *Trans. Circuits Syst. I* **2000**, *47*, 25–39 [[CrossRef](#)]
- Xue, D.; Zhao, C.; Chen, Y. A Modified Approximation Method of Fractional Order System. In Proceedings of the International Conference on Mechatronics and Automation, Luoyang, China, 25–28 June 2006; pp. 1043–1048.
- Charef, A.; Sun, H.H.; Tsao, Y.Y.; Onaral, B. Fractal system as represented by singularity function. *Trans. Autom. Control* **1992**, *37*, 1465–1470. [[CrossRef](#)]
- Carlson, G.E.; Halijak, C.A. Approximation of a fractional capacitors $(1/s)^{1/n}$ by a regular Newton process. *Trans. Circuits Theory CT-11* **1964**, *2*, 210–213. [[CrossRef](#)]
- Shrivastava, N.; Varshney, P. Rational Approximation of Fractional Order Systems Using Carlson Method. In Proceedings of the International Conference on Soft Computing Techniques and Implementation, Faridabad, India, 8–10 October 2015; pp. 76–80.
- Matsuda, K.; Fujii, H. H_∞ optimized wave-absorbing control: Analytical and experimental results. *J. Guid. Control Dyn.* **1993**, *16*, 1146–1153. [[CrossRef](#)]
- Sánchez-López, C. An experimental synthesis methodology of fractional-order chaotic attractors. *Nonlinear Dyn.* **2020**, *100*, 3907–3923.
- Sánchez-López, C.; Carbajal-Gómez, V.H.; Carrasco-Aguilar, M.A.; Carro-Pérez, I. Fractional-Order Memristor Emulator Circuits. *Complexity* **2018**, *2018*, 2806976. [[CrossRef](#)]
- Podlubny, I. Fractional-order systems and $PI^\lambda D^\mu$ -controllers. *IEEE Trans. Autom. Control* **1999**, *44*, 208–214. [[CrossRef](#)]
- Muñoz-Montero, C.; García-Jiménez, L.V.; Sánchez-Gaspariano, L.A.; Sánchez-López, C.; González-Díaz, V.R.; Tlelo-Cuautle, E. New alternatives for analog implementation of fractional-order integrators, differentiators and PID controllers based on integer-order integrators. *Nonlinear Dyn.* **2017**, *90*, 241–256. [[CrossRef](#)]
- Raynaud, H.F.; Zergainoh, A. State-space representation for fractional order controllers. *Automatica* **2000**, *36*, 1017–1021. [[CrossRef](#)]
- Monje, C.A.; Vinagre, B.M.; Calderon, A.J.; Feliu, V.; Chen, Y.Q. Auto-tuning of fractional lead-lag compensators. In Proceedings of the 16th IFAC World Congress, Prague, Czech Republic, 4–8 July 2005; pp. 319–324.
- Kapoulea, S.; Psychalinos, C.; Elwakil, A.S. Double exponent fractional-order filters: Approximation methods and realization. *Circuits Syst. Signal Process.* **2021**, *40*, 993–1004. [[CrossRef](#)]
- Kapoulea, S.; Psychalinos, C.; Elwakil, A.S. Power law filters: A new class of fractional-order filters without a fractional-order Laplacian operator. *AEU-Int. J. Electron. Commun.* **2021**, *129*, 153537. [[CrossRef](#)]
- Krishna, B.T. Studies on fractional order differentiators and integrators: A survey. *Signal Process.* **2011**, *91*, 386–426. [[CrossRef](#)]
- Nur-Deniz, F.; Baykant-Alagoz, B.; Tan, N.; Koseoglu, M. Revisiting four approximation methods for fractional order transfer function implementations: Stability preservation, time and frequency response matching analyses. *Annu. Rev. Control* **2020**, *49*, 239–257. [[CrossRef](#)]
- Atherton, D.P.; Tan, N.; Yüce, A. Methods for computing the time response of fractional-order systems. *IET Control Theory Appl.* **2015**, *9*, 817–830. [[CrossRef](#)]
- Cafagna, D.; Grassi, G. Fractional-order Chua's circuit: Time-domain analysis, bifurcation, chaotic behaviour and test for chaos. *Int. J. Bifur. Chaos.* **2008**, *18*, 261–267. [[CrossRef](#)]
- Diethelm, K.; Ford, N.J.; Freed, A.D. A predictor-corrector approach for the numerical solution of fractional differential equations. *Nonlinear Dyn.* **2002**, *29*, 3–22. [[CrossRef](#)]
- Deng, W. Numerical algorithm for the time fractional Fokker-Planck equation. *J. Comput. Phys.* **2007**, *227*, 1510–1522. [[CrossRef](#)]
- Li, C.; Xiong, J.; Li, W.; Tong, Y.; Zeng, Y. Robust synchronization for a class of fractional-order dynamical system via linear state variable. *Indian J. Phys.* **2013**, *87*, 673–678. [[CrossRef](#)]
- Parsa-Moghaddam, B.; Yaghoobi, S.; Tenreiro-Machado, J.A. An extended predictor-corrector algorithm for variable-order fractional delay differential equations. *J. Comput. Nonlinear Dyn.* **2016**, *11*. [[CrossRef](#)]
- Dabiri, A.; Butcher, E.A. Stable fractional Chebyshev differentiation matrix for the numerical solution of multi-order fractional differential equations. *Nonlinear Dyn.* **2017**, *90*, 185–201. [[CrossRef](#)]
- Dabiri, A.; Butcher, E.A. Numerical solution of multi-order fractional differential equations with multiple delays via spectral collocation methods. *Appl. Math. Model.* **2018**, *56*, 424–448. [[CrossRef](#)]
- Khovanskii, A.N. *The Application of Continued Fractions and Their Generalizations to Problems in Approximation Theory*; Noordhoff, Ltd.: Groningen, The Netherlands, 1963; pp. 100–110.
- Bingi, K.; Ibrahim, R.; Karsiti, M.N.; Hassam, S.M.; Harindran, V.R. Frequency response based curve fitting approximation of fractional-order PID controllers. *Int. J. Appl. Math. Comput. Sci.* **2019**, *29*, 311–326. [[CrossRef](#)]

30. Bingi, K.; Ibrahim, R.; Karsiti, M.N.; Hassam, S.M.; Harindran, V.R. *Fractional-Order Systems and PID Controllers*; Springer: Berlin/Heidelberg, Germany, 2020.
31. Sanathanan, C.; Koerner, J. Transfer function synthesis as a ratio of two complex polynomials. *IEEE Trans. Autom. Control* **1963**, *8*, 56–58. [[CrossRef](#)]
32. Koseoglu, M.; Nur-Deniz, F.; Baykant-Alagoz, B.; Alisoy, H. An effective analog circuit design of approximate fractional-order derivative models of M-SBL fitting method. *Int. J. Eng. Sci. Technol.* **2021**, *1*, 1–15. [[CrossRef](#)]
33. Nur-Deniz, F.; Baykant-Alagoz, B.; Tan, N.; Atherton, D.P. An integer order approximation method based on stability boundary locus for fractional order derivative/integrator operators. *ISA Trans.* **2016**, *62*, 154–163. [[CrossRef](#)]
34. Gustavsen, B.; Semlyen, A. Rational approximation of frequency domain responses by vector fitting. *IEEE Trans. Power Deliv.* **1999**, *14*, 1052–1061. [[CrossRef](#)]
35. Djouambi, A.; Charef, A.; Besancon, A.V. Optimal approximation, simulation and analog realization of the fundamental fractional order transfer function. *Int. J. Appl. Math. Comput. Sci.* **2007**, *17*, 455–462. [[CrossRef](#)]
36. Maione, G. Continued fractions approximation of the impulse response of fractional order dynamic systems. *IET Control Theory Appl.* **2008**, *2*, 564–572. [[CrossRef](#)]
37. Maione, G. Closed-Form Rational Approximations of Fractional, Analog and Digital Differentiators/Integrators. *J. Emerg. Sel. Top. Circuits Syst.* **2013**, *3*, 322–329. [[CrossRef](#)]
38. El-Khazali, R. Discretization of fractional-order Laplacian operators. In Proceedings of the 19th IFAC Congress, Cape Town, South Africa, 24–29 August 2014; pp. 2016–2021.
39. AbdelAty, A.M.; Elwakil, A.S.; Radwan, A.G.; Psychalinos, C.; Maundy, B.J. Approximation of the fractional-order Laplacian s^{α} as a weighted sum of first-order high-pass filters. *IEEE Trans. Circuit. Sys. II Express Briefs* **2018**, *65*, 1114–1118.
40. Baker George, A. *Essentials of Padé Approximants*, 1st ed.; Academic Press: New York, NY, USA, 1975.
41. Green, M. A Relative Error Bound for Balanced Stochastic Truncation. *Trans. Autom. Control* **1998**, *62*, 961–965. [[CrossRef](#)]
42. Varga, A. On stochastic balancing related model reduction. In Proceedings of the 39th IEEE Conference on Decision and Control, Sydney, NSW, Australia, 12–15 December 2000; pp. 2385–2390.