A Numerical Algorithm for Reconstructing Continuous-Time Linear Models with Pure Integrators from Their Discrete-Time Equivalents

Zaher M. Kassas and Greg Morrow

***Abstract*— A numerical algorithm for reconstructing continuous-time linear models with pure integrators from their discrete-time equivalents is derived in this paper. The algorithm treats the reconstruction problem in the frequency-domain and establishes a numerically tractable z-domain to s-domain mapping, which avoids the need to software packages with symbolic mathematical solving capabilities. The proposed algorithm can be conveniently implemented using standard building block numerical routines and has been recently employed in a computer-aided control systems design software package. Unlike other approaches, this algorithm is shown to avoid unnecessary computations, which typically result in numerical burden, ambiguity, and singularity. A numerical example is illustrated demonstrating the application of the proposed algorithm.**

I. Introduction

Identification of models is a problem of considerable importance in various disciplines, such as control, economics, and signal processing. The identification process is often done using a digital computer, from samples of input-output data. However, the model is usually of continuous-time (CT) nature, and its dynamical properties are most aptly described in terms of differential equations or transfer functions (TFs). The two basic approaches for identification of CT systems can be categorized into two methods: (i) the indirect method, where first a discrete-time (DT) model is estimated from the sampled data and then an equivalent CT model is calculated; and (ii) the direct method, where a CT model is obtained directly without going through the intermediate step of determining a DT model [1], [2], [3].

Identification of CT linear time invariant (LTI) systems can be done either in the time-domain or in the frequency-domain. Time-domain methods usually determine a DT model of the system, while frequency-domain methods can identify either a DT (z-domain) or a CT (s-domain) model. In frequency-domain based indirect identification, where a z-domain model has to be determined first, the usual underlying assumptions are: (i) the excitation signal is constant between the sampling instants, also known as the zero-order hold (ZOH) assumption, and (ii) the output of the signal is sampled without bandlimiting it by an anti-aliasing filter [4], [5]. If theses assumptions are fulfilled, the s-domain rational TF uniquely determines a corresponding z-domain model through the so-called step-invariant (ZOH) transform.

However, while the ZOH transformation from the s-domain to the z-domain is unique, the other direction is sometimes ambiguous. In [6], it was shown that it is possible to reconstruct a CT rational TF based on the discretized model at several, suitably chosen sampling rates. However, the approach assumed that the TF does not possess multiple poles at the same location.

Since rational TFs can be expanded into partial fractions with poles in the denominators, and since the ZOH transform is linear, the reconstruction of the CT TF from its DT equivalent can be achieved by expanding the DT TF into its partial fractions and applying the inverse ZOH transformation on each. Consequently, all the DT poles can be mapped into their CT counterpart. The zeros cannot be related to each other by a similarly simple mapping, however [7].

Given a rational TF, $H(z)$, corresponding to a sampled representation of a CT TF, $H(s)$, the partial fraction expansion (PFE) of $H(z)$ will be of the form

$$H(z) = \sum_{j=1}^{n} \sum_{i=1}^{m_i} \frac{d_{i,j}}{(z - p_i)^j},$$

where n is the number of poles that $H(z)$ possesses, m_i is the multiplicity of the i^{th} pole p_i, and $\{d_{i,j};\, i = 1, \ldots, m_i;\, j = 1, \ldots, n\}$ are constants. The partial fractions of $H(z)$ will be classified to belong to one of four different classes: fractions with poles at $z = 1$, fractions with poles at $z = 0$, fractions with negative real poles, and fractions with poles elsewhere.

For the well-known problem of reconstructing TF models with negative real poles, since the z-domain to s-domain mapping involves taking the logarithm of the pole p_i, and since $\ln(-p_i) = \ln(p_i) + j(\pi + 2k\pi), k = 0, \pm1, \pm2, \ldots$, the transformation is not unique. Even if we select a certain leaf of the complex logarithm function, we end up with a complex pole without a complex conjugate pair. Kollar et al in [8] suggested an extra manipulation to overcome this issue. By adding a canceling zero/pole pair to the z-domain model, the transformation becomes possible, but then ambiguity appears. The undetermined part in the TF of the resulting s-domain system, as it was shown, is a signal with zero crossings exactly at the sampling instants. These hidden oscillations can be eliminated by applying a “parsimony principle”. For the case where the z-domain system contains poles at the origin, which translates to a delay in the s-domain, several algorithms were suggested [9], [10].

For the case where there are no negative real poles in the z-domain TF nor poles at the origin, and assuming that the sampling radian frequency, Ω, is larger than double the largest imaginary part of the s-domain poles, the z-domain

This work has been supported by National Instruments.

Zaher M. Kassas and Greg Morrow are with National Instruments, Control Design & Simulation R&D, 11500 N. Mopac Expwy., Austin, TX 78759, USA. zkassas@ieee.org and greg.morrow@ni.com

to s-domain transformation is unique and straightforward, analytically. Nevertheless, if a computer-aided control system design (CACSD) software tool is to be employed to perform automatically such mapping, then such a tool should possess symbolic mathematical solver capabilities. This stems from the fact that the z-transform of a rational TF in PFE form involves taking partial derivatives. This paper will focus on the case when the poles are at $z = 1$, which translate to pure integrators in the s-domain. In this respect, a numerical algorithm for establishing z-domain to s-domain mapping is derived. Consequently, this algorithm is employed into ZOH reconstruction of TF models with pure integrators. It is argued that our algorithm can be extended into reconstructing TFs belonging to the fourth class discussed above, i.e. TFs with poles elsewhere.

This paper is organized as follows. Section II reviews relevant alternative algorithms and establishes how our approach is different. Section III derives a numerical algorithm to achieve the z-domain to s-domain mapping. Section IV presents the algorithm for reconstructing CT TF models with pure integrators, given their ZOH DT equivalents. Section V illustrates the application of the proposed algorithm through a numerical example. Concluding remarks and future work are presented in Section VI.

II. Background and Motivation

Most CACSD software packages implement their native algorithms in state-space [11]. As such, the process of reconstructing a CT TF from its DT equivalent involves realizing the TF in SS, reconstructing the CT SS model from the DT representation, and finally converting the reconstructed CT SS model into a TF. In particular, give a CT LTI system defined by the CT system matrices $(\mathbf{A}_c, \mathbf{B}_c, \mathbf{C}_c, \mathbf{D}_c)$ as

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}_c\mathbf{x}(t) + \mathbf{B}_c\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}_c\mathbf{x}(t) + \mathbf{D}_c\mathbf{u}(t), \end{aligned}$$

where $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{B} \in \mathbb{R}^{n\times r}$, $\mathbf{C} \in \mathbb{R}^{m\times n}$, and $\mathbf{D} \in \mathbb{R}^{m\times r}$, the ZOH DT equivalent can be readily shown to be given by

$$\begin{aligned} \mathbf{A}_d &= e^{\mathbf{A}_c T}, \quad \mathbf{C}_d = \mathbf{C}_c, \quad \mathbf{D}_d = \mathbf{D}_c, \\ \mathbf{B}_d &= \int_0^T e^{\mathbf{A}_c \tau} d\tau\, \mathbf{B}_c \;=\; [\mathbf{A}_d - \mathbf{I}]\, \mathbf{A}_c^{-1}\mathbf{B}_c, \end{aligned}$$

where T is the sampling time [12]. Consequently, the reconstruction of the CT system matrices, given their DT counterpart involves performing the matrix logarithm and the matrix inverse operations, each of which may result in an ambiguity. The first ambiguity arises if $\mathbf{A}_d$ has negative real eigenvalues, in which case $\mathbf{A}_c$ would contain complex-valued elements. The second ambiguity arises whenever $\mathbf{A}_d$ has eigenvalues at 1 with any multiplicity, which corresponds to $\mathbf{A}_c$ possessing eigenvalues at 0. This is due to the fact that computing $\mathbf{B}_c$ would involve computing the matrix $[\mathbf{A}_d - \mathbf{I}]^{-1}$, which would be rank-deficient by as many eigenvalues at 1 the matrix $\mathbf{A}_d$ would have. To circumvent this singularity, a common approach is to compute the matrix equivalents through the relationship

$$\begin{bmatrix} \mathbf{A}_d & \mathbf{B}_d \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \exp\left(\begin{bmatrix} \mathbf{A}_c & \mathbf{B}_c \\ \mathbf{0} & \mathbf{0} \end{bmatrix} T \right), \tag{1}$$

which avoids performing the matrix inversion step [13]. Nevertheless, the process of realizing a given TF in SS, performing the computations, and finally converting back the resulting SS model into a TF involves unnecessary intermediate steps along with computationally intensive linear algebra manipulations. Even though several algorithms have been proposed to reduce such numerical burden when computing $\mathbf{A}_c$ and $\mathbf{B}_c$, such as [14], [15], it is desirable if the reconstruction algorithm does not involve the intermediate steps of realizing the system in SS and back to TF nor it involves intensive manipulations. An attempt to formulate a frequency-domain based algorithm that performs the z-domain to s-domain mapping was presented in [16]. However, the approach in this paper is different in several aspects. First, all the numerical "building blocks" of our algorithm are basic looping and simple polynomial manipulations and can be readily found in any standard numerical software, such as LabVIEW. In particular, our algorithm eliminates the need to perform PFE of polynomials of the form $(1-z^{-1})^{-1}$, which is a step that requires non-standard polynomial manipulation building blocks. Moreover, the proposed algorithm is attractive for "in-the-loop" identification, where the identification algorithm could reside on embedded devices, such as processors and field-programmable gate arrays (FPGAs), with limited computational capabilities, and where deploying intensive linear algebra algorithms may not be possible. Second, we derive our algorithm starting from standard Laplace and z-transform tables and show how we can perform the process of taking partial derivatives through simple numerical polynomial manipulations. Consequently, we employ this z-domain to s-domain mapping into reconstructing CT TFs with pure integrators from their ZOH DT counterpart. This algorithm has been thoroughly tested and is now made available through the LabVIEW Control Design and Simulation Module [11] and the LabVIEW System Identification Toolkit [17].

III. Numerical Algorithm for Computing the z-Transform of Pure Integrator Models

The z-transform of an m^{th}-order integrator can be found in standard Laplace and z-transform table to be given by

$$\frac{1}{s^m} \longleftrightarrow \lim_{\alpha\to 0} \frac{(-1)^{m-1}}{(m-1)!} \frac{\partial^{m-1}}{\partial \alpha^{m-1}} \left\{ \frac{z}{z - e^{-\alpha T}} \right\}, \tag{2}$$

where T is the sampling time in seconds [12]. Equation (2) suggests that the s-domain to z-domain mapping cannot be achieved through a simple numerical algorithm, and rather we need to have a symbolic mathematical solver to achieve the transformation from one domain to another. This section will outline how to compute numerically the z-transform of an m^{th}-order integrator.

Proposition 1: The z-transform of an m^{th}-order integrator can be computed numerically from

$$H_m(s) = \frac{1}{s^m} \longleftrightarrow z \sum_{i=1}^{m} \frac{b_{m,i}}{(z-1)^i} = H_m(z), \qquad (3)$$

where the coefficients $\{b_{m,i}\}_{i=1}^m$ are to be computed through the algorithm outline in the proof below.

Proof: The proof will proceed by proving the structure of (3) and then deriving a numerical recursive algorithm for computing the coefficients $\{b_{m,i}\}_{i=1}^m$. First, we will rewrite (3) as

$$H_m(z) = \frac{(-1)^{m-1}}{(m-1)!} \lim_{\alpha \to 0} G_{m-1}(z, \alpha), \qquad (4)$$

where we define the auxiliary function $G_m(z, \alpha)$ as

$$G_m \triangleq \frac{\partial^m}{\partial \alpha^m} \left\{ \frac{z}{z - e^{-\alpha T}} \right\}, \qquad (5)$$

with the initial condition and recursive relationships defined respectively as

$$G_0 \triangleq \frac{z}{z - e^{-\alpha T}} \qquad (6)$$

$$G_i \triangleq \frac{\partial}{\partial \alpha} G_{i-1}, \qquad i = 1, 2, \ldots, \qquad (7)$$

where the dependency of $G_i(z, \alpha)$ on z and α in the preceding equations was dropped for simplicity of notation. From the definitions of (5)-(7) its easy to see that

$$G_1 = C.G_0, \qquad (8)$$

where

$$C \triangleq \frac{-T\, e^{-\alpha T}}{z - e^{-\alpha T}}. \qquad (9)$$

From (7)-(9) we can readily compute $G_2(z, \alpha)$ as

$$G_2 = P_1.G_1, \qquad (10)$$

where $P_1(z, \alpha)$ is defined as

$$P_1 \triangleq 2\,C - T. \qquad (11)$$

Next, we will define the recursion

$$P_i \triangleq \frac{\partial}{\partial \alpha} P_{i-1}, \qquad i = 2, 3, \ldots. \qquad (12)$$

From (7), (10), and (12) we can note that G_i and P_i satisfy the conditions of Lemma 1, except that the indices of the initial conditions differ, i.e. we have the correspondence

$$G_2 = P_1 G_1 \longleftrightarrow g_1 = f_0\, g_0.$$

From Lemma 1, we know that for any $m \in \mathbb{N}$ we have

$$g_m = \sum_{i=0}^{m-1} \binom{m-1}{i} f_{m-1-i}\, g_i. \qquad (13)$$

Replacing g_m with G_{m+1} and f_m with P_{m+1} in (13) we get

$$G_{m+1} = \sum_{i=0}^{m-1} \binom{m-1}{i} P_{m-i}\, G_{i+1} \qquad (14)$$

$$= \sum_{k=1}^{m} \binom{m-1}{k-1} P_{m-k+1}\, G_k, \qquad (15)$$

where we have used the change of summation index $k = i+1$ to go from (14) to (15). Finally, we can rewrite (15) as

$$G_m = \sum_{i=1}^{m-1} \binom{m-2}{i-1} P_{m-i}\, G_i. \qquad (16)$$

Now, we will derive the relationship between P_i and P_{i+1}. In this respect, we will use the argument of induction, which will eventually yield a recursive formula for computing $\{P_i\}_{i=1}^m$. First, we will express (11) as

$$P_1 = \sum_{j=0}^{1} a_{1,j}\, T^{1-j}\, C^j, \qquad (17)$$

where the coefficients $a_{1,j}$ in (17) evaluate to $a_{1,0} = -1$ and $a_{1,1} = 2$. Generalizing (17) to any $i \in \mathbb{N}$, we have the induction hypothesis

$$P_i = \sum_{j=0}^{i} a_{i,j}\, T^{i-j}\, C^j. \qquad (18)$$

Next, we will derive a recursive formula relating P_i to P_{i+1} that will ensure the induction step, i.e. ensuring that if P_i is true, then P_{i+1} is also true. From (12) and (18) we can see that

$$P_{i+1} = \frac{\partial}{\partial \alpha} \left\{ \sum_{j=0}^{i} a_{i,j}\, T^{i-j}\, C^j \right\}$$

$$= \sum_{j=0}^{i} a_{i,j}\, T^{i-j}\, j\, C^{j+1} - \sum_{j=0}^{i} a_{i,j}\, T^{i-j+1}\, j\, C^j. \qquad (19)$$

Using the change of variables $k = j+1$ in the first summation in (19) and splitting the last term of the first summation and the first term of the second summation yields

$$P_{i+1} = \sum_{k=1}^{i} a_{i,k-1}(k-1) T^{i-k+1} C^k - \sum_{j=1}^{i} a_{i,j} T^{i-j+1} j C^j + a_{i,i}\, i\, C^{i+1}. \qquad (20)$$

Now, rewriting the summation in (18) up to $i+1$ yields

$$P_{i+1} = \sum_{j=0}^{i+1} a_{i+1,j}\, T^{i+1-j}\, C^j. \qquad (21)$$

Splitting out the first and last terms of the summation in (21) yields

$$P_{i+1} = \sum_{j=1}^{i} a_{i+1,j}\, T^{i+1-j}\, C^j + a_{i+1,0} T^{i+1} C^0 + a_{i+1,i+1} T^0 C^{i+1}. \qquad (22)$$

For the induction step to hold true, it is required that (22) be equal to (20). Equating the coefficients of equal powers of C in the two expressions gives

$$C^0 : a_{i+1,0} \equiv 0 \qquad (23)$$

$$C^{i+1} : a_{i+1,i+1} \equiv a_{i,i}\, i \qquad (24)$$

$$C^j : a_{i+1,j} \equiv a_{i,j-1}(j-1) - a_{i,j} j, \quad j = 1, \ldots, i. \qquad (25)$$

We can view (23)-(25) as not only necessary for proving the induction step, but also as means for computing recursively P_i for any $i \in \mathbb{N}$. This will facilitate the evaluation of G_m, since P_i and G_m are related through (16). This in turn will facilitate the computation of $H_m(z)$ as it is related to $G_{m-1}(z,\alpha)$ through (4).

The last remaining step is to take the limit as $\alpha \to 0$. First, note that

$$\lim_{\alpha\to 0} C = \frac{-T}{z-1}.$$

Hence, taking the limit as $\alpha \to 0$ of (18) yields

$$\lim_{\alpha\to 0} P_i = \sum_{j=0}^{i} \frac{\tilde{a}_{i,j}}{(z-1)^j}, \tag{26}$$

where

$$\tilde{a}_{i,j} \triangleq (-1)^j T^i a_{i,j}. \tag{27}$$

From (26) we can see that P_i is a polynomial of order i in $(z-1)^{-1}$. Now, taking the limit as $\alpha \to 0$ of G_0 and G_1, respectively, yields

$$\begin{aligned} \lim_{\alpha\to 0} G_0 &= \frac{z}{z-1} \qquad (28)\\ \lim_{\alpha\to 0} G_1 &= \frac{-Tz}{(z-1)^2}. \end{aligned}$$

Using (16) with $m=2$ we can see that G_2 is the product of G_1 and P_1; hence, $\lim_{\alpha\to 0} G_2$ is a polynomial of third order in $(z-1)^{-1}$ times one power of z. More generally, it is easy to see that for any $m \in \mathbb{Z}$, $\lim_{\alpha\to 0} G_m$ is a polynomial of order $(m+1)$ in $(z-1)^{-1}$ times one power of z. Since H_m is related to G_{m-1} through (4), this proves that H_m is a polynomial of order m in $(z-1)^{-1}$ times one power of z, which in turns proves the structure of (3).

Next, we will determine the recursion on $\{b_{m,i}\}_{i=1}^{m}$ in (3), which will complete the proof. From the relationship between H_m and G_{m-1} in (4), it is clear that $\lim_{\alpha\to 0} G_{m-1}$ has an expansion like (3), but its coefficients, which we will denote by $\tilde{b}_{m,i}$, will differ from those of H_m by the factor $\frac{(-1)^{m-1}}{(m-1)!}$. So, we let

$$\lim_{\alpha\to 0} G_{m-1} = z\sum_{i=1}^{m} \frac{\tilde{b}_{m,i}}{(z-1)^i}, \tag{29}$$

where

$$\tilde{b}_{m,i} = \frac{(m-1)!}{(-1)^{m-1}} b_{m,i}. \tag{30}$$

Substituting (26) into (16) we get

$$\begin{aligned} \lim_{\alpha\to 0} G_m &= \sum_{i=0}^{m-1} \binom{m-2}{i-1} \left[\sum_{j=0}^{m-i} \frac{\tilde{a}_{m-i,j}}{(z-1)^j}\right] \lim_{\alpha\to 0} G_i \\ &= \sum_{i=0}^{m-1} \binom{m-2}{i-1} \left[\sum_{j=0}^{m-i} \frac{\tilde{a}_{m-i,j}}{(z-1)^j}\right] \\ &\quad \cdot \left[z\sum_{k=1}^{i+1} \frac{\tilde{b}_{i+1,k}}{(z-1)^k}\right] \qquad (31)\\ &\equiv z\sum_{i=1}^{m+1} \frac{\tilde{b}_{m+1,i}}{(z-1)^i}, \qquad (32) \end{aligned}$$

where we have used (29) to write (31) as (32). Hence, generally, given a desired order $m > 2$, we can evaluate $H_m(z)$ through the following recursive numerical algorithm

1) Starting with $a_{1,0} = -1$ and $a_{1,1} = 2$, calculate $\tilde{a}_{1,0}$ and $\tilde{a}_{1,1}$ using (27). Also, identify that $\tilde{b}_{2,1} = 0$ and $\tilde{b}_{2,2} = -T$ from (28) and (30). Set $n = 2$.
2) Evaluate

$$\begin{aligned} \lim_{\alpha\to 0} G_n &= \sum_{i=0}^{n-1} \binom{n-2}{i-1} \left[\sum_{j=0}^{n-i} \frac{\tilde{a}_{n-i,j}}{(z-1)^j}\right] \qquad (33)\\ &\quad \cdot \left[z\sum_{k=1}^{i+1} \frac{\tilde{b}_{i+1,k}}{(z-1)^k}\right]. \qquad (34) \end{aligned}$$

3) Accumulate the resulting polynomial and identify the coefficients $\left\{\tilde{b}_{n+1,i}\right\}_{i=1}^{n+1}$ corresponding to

$$\lim_{\alpha\to 0} G_n = z\sum_{i=1}^{n+1} \frac{\tilde{b}_{n+1,i}}{(z-1)^i}. \tag{35}$$

4) If $m \equiv n$, stop; otherwise, set $n = n+1$, calculate $\{\tilde{a}_{n-1,j}\}_{j=0}^{n-1}$ using (23)-(25), and go to step (2).
5) Finally, calculate the coefficients $\{b_{m,i}\}_{i=1}^{m}$ from the coefficients $\left\{\tilde{b}_{m,i}\right\}_{i=1}^{m}$ using the relationship (30).

■

IV. ZOH Reconstruction of Models with Pure Integrators

Proposition 2: The ZOH reconstruction of a given TF with m-poles at $z = 1$ is given by

$$\mathrm{ZOH}^{-1}\left\{\frac{1}{(z-1)^m}\right\} = \sum_{j=0}^{m} \frac{c_{m,j}}{s^j}, \tag{36}$$

where the coefficients $\{c_{m,j}\}_{j=0}^{m}$ are to be computed as outlined in the proof below.

Proof: The proof will proceed as follows. Assuming that the proposed structure is true, this will lead to a system of linearly independent equations, which upon solving we can identify the coefficients $\{c_{m,j}\}_{j=0}^{m}$. First, we recall that the so-called step-invariant (or ZOH) transform of a TF $H(s)$ is given by

$$H(z) = \mathrm{ZOH}\{H(s)\} = \frac{z-1}{z}\mathcal{Z}\left\{\frac{H(s)}{s}\right\}, \tag{37}$$

where $\mathcal{Z}\left\{\frac{H(s)}{s}\right\}$ is the shorthand for $\mathcal{Z}\left\{\mathcal{L}^{-1}\left\{\frac{H(s)}{s}\right\}\right\}$, and $\mathcal{L}\{\cdot\}$, $\mathcal{Z}\{\cdot\}$, denote the Laplace and $z-$transforms, respectively [12]. Applying the ZOH transform to both sides of (36) yields

$$\begin{aligned}\frac{1}{(z-1)^m} &= \text{ZOH}\left\{\sum_{j=0}^{m}\frac{c_{m,j}}{s^j}\right\} \\ &= \frac{z-1}{z}\sum_{j=0}^{m}c_{m,j}\,\mathcal{Z}\left\{\frac{1}{s^{j+1}}\right\} \quad (38)\\ &= \frac{z-1}{z}\sum_{j=0}^{m}c_{m,j}\left[z\sum_{i=1}^{j+1}\frac{b_{j+1,i}}{(z-1)^i}\right] \quad (39)\\ &= \sum_{j=0}^{m}c_{m,j}\left[\sum_{i=1}^{m+1}\frac{b_{j+1,i}}{(z-1)^{i-1}}\right] \quad (40)\\ &= \sum_{i=1}^{m+1}\frac{1}{(z-1)^{i-1}}\left[\sum_{j=0}^{m}c_{m,j}\,b_{j+1,i}\right], \quad (41)\end{aligned}$$

where to go from (38) to (39) we have used (3), and to go from (39) to (40) we have used the fact that $b_{j,i}\equiv 0,\ \forall\, i>j$, which enabled us to write the second summation up to $m+1$. Finally, we have rearranged the summations to go from (40) to (41). Using the change of variable $k=i-1$ in the index of the first summation of (41) yields

$$\frac{1}{(z-1)^m}=\sum_{k=0}^{m}\frac{1}{(z-1)^k}\left[\sum_{j=0}^{m}c_{m,j}\,b_{j+1,k+1}\right]. \quad (42)$$

We can readily recognize that for (42) to hold true, we need that all the coefficients in the square brackets of the second summation corresponding to terms with $k<m$ to be zero, whereas the coefficient corresponding to the term with $k=m$ must be 1. Consequently, we can write the linear system

$$\begin{bmatrix} b_{1,1} & b_{2,1} & b_{3,1} & \cdots & b_{m+1,1}\\ 0 & b_{2,2} & b_{3,2} & \cdots & b_{m+1,2}\\ 0 & 0 & b_{3,3} & \cdots & b_{m+1,3}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & 0 & b_{m+1,m+1}\end{bmatrix}\begin{bmatrix} c_{m,0}\\ c_{m,1}\\ c_{m,2}\\ \vdots\\ c_{m,m}\end{bmatrix}=\begin{bmatrix}0\\0\\0\\ \vdots\\ 1\end{bmatrix}. \quad (43)$$

Therefore, we can evaluate the coefficients $\{c_{m,j}\}_{j=0}^{m}$ using the linear system of equations in (43). We can show that the $(m+1)\times(m+1)$ matrix is nonsingular by contradiction. First, we note that in order for the $(m+1)\times(m+1)$ upper triangular matrix to be nonsingular, we need that all the diagonal elements $\{b_{ii}\}_{i=1}^{m+1}$ to be nonzero. The proof by contradiction will proceed as follows. We will assume that one or more coefficient on the diagonal of this matrix is zero and show that this cannot be true. To do so, we note that the diagonal elements $\{b_{ii}\}_{i=1}^{m+1}$, which facilitate the evaluation of the TFs $\{H_i(z)\}_{i=1}^{m+1}$ are related to the coefficients $\left\{\tilde{b}_{ii}\right\}_{i=1}^{m+1}$ through (30), which we used to evaluate the auxiliary TFs $\{G_i(z)\}_{i=1}^{m+2}$, through (32). Moreover, we have showed in the proof of Lemma 1 that for any $m\in\mathbb{Z}$, $\lim_{\alpha\to 0}G_m$ is a polynomial of order $(m+1)$ in $(z-1)^{-1}$. Hence, the coefficient $\tilde{b}_{m+1,m+1}$ has to be nonzero, because if it was zero, then the TF $G_m(z)$ will be a polynomial of order less than $(m+1)$ in $(z-1)^{-1}$. Since this is true for any $m\in\mathbb{Z}$, then none of the coefficients $\left\{\tilde{b}_{ii}\right\}_{i=1}^{m+1}$ can be zero, which completes the proof by contradiction. ∎

V. Example

This section will demonstrate the application of the proposed algorithm into reconstructing CT TF models of systems with pure integrators from their ZOH DT equivalents. In this respect, the algorithms that were presented in Section III and IV were coded numerically and were consequently used to reconstruct CT TF models of systems with integrators up to six multiplicities. Table I shows the values of the coefficients $c_{m,j}$ that correspond to the reconstructed TF in (36), where m is the multiplicity of the integrators, and $j=0,1,\ldots,m$.

TABLE I
ZOH reconstruction of CT TF pure integrator models up to six multiplicities

m	j						
	0	1	2	3	4	5	6
1	0	$\frac{1}{T}$					
2	0	$\frac{-1}{2T}$	$\frac{1}{T^2}$				
3	0	$\frac{1}{3T}$	$\frac{-1}{T^2}$	$\frac{1}{T^3}$			
4	0	$\frac{-1}{4T}$	$\frac{11}{12T^2}$	$\frac{-3}{2T^3}$	$\frac{1}{T^4}$		
5	0	$\frac{1}{5T}$	$\frac{-5}{6T^2}$	$\frac{7}{4T^3}$	$\frac{-2}{T^4}$	$\frac{1}{T^5}$	
6	0	$\frac{-1}{6T}$	$\frac{137}{180T^2}$	$\frac{-15}{8T^3}$	$\frac{17}{6T^4}$	$\frac{-5}{2T^5}$	$\frac{1}{T^6}$

VI. Conclusion

This paper has presented a numerical algorithm for reconstructing CT TF models of systems with pure integrators from their ZOH DT equivalents. The algorithm used basic recursion and simple polynomial manipulations and is well suited for CACSD software. The algorithm performs the reconstruction solely in the frequency-domain; hence avoids intermediate unnecessary steps of realizing the TF in SS and converting back the CT SS model into TF. As such, the algorithm avoids a singularity that may arise if the reconstruction is done in SS, and it avoids employing computationally costly linear algebra manipulations. This work established a simple z-domain to s-domain mapping, where the process of taking partial derivatives is realized through polynomial multiplications. Such mapping was derived from standard Laplace and z-transform tables.

The generalization of this algorithm into reconstructing CT TFs from their ZOH DT equivalents will be investigated in future work. In this respect, the generalized algorithm will be extended to handle poles at arbitrary locations in the z-domain. It is noted that the z-transform of CT TFs with poles at arbitrary locations has the same structure of the z-transform of CT TFs with poles at $s=0$, cf. (2), except for taking the limit of the partial derivatives. Consequently, it is argued that the generalized algorithm will follow the derivations outlined in this work.

APPENDIX

Lemma 1: Given two sequences of m-times differentiable functions, with respect to the variable α, $\{f_i\}_{i=0}^{m}$ and $\{g_i\}_{i=0}^{m}$, satisfying

$$\begin{aligned} f_i &\triangleq \frac{\partial f_{i-1}}{\partial \alpha}, \qquad i = 1, 2, \dots && (44) \\ g_i &\triangleq \frac{\partial g_{i-1}}{\partial \alpha}, \qquad i = 1, 2, \dots && (45) \\ g_1 &= f_0 \cdot g_0, && (46) \end{aligned}$$

then it follows that

$$\begin{aligned} g_m &= \frac{\partial g_{m-1}}{\partial \alpha} \\ &= \sum_{i=0}^{m-1} \binom{m-1}{i} f_{m-1-i}\, g_i, && (47) \end{aligned}$$

where $\binom{m}{i}$ is the binomial coefficient, defined as

$$\binom{m}{i} \triangleq \frac{m!}{(m-i)!\, i!}\,. \qquad (48)$$

Proof: From (44)-(46) it is easy to see that

$$\begin{aligned} g_2 &= f_1 \cdot g_0 + f_0 \cdot g_1 && (49) \\ g_3 &= f_2 \cdot g_0 + 2\, f_1 \cdot g_1 + f_0 \cdot g_2 && (50) \\ g_4 &= f_3 \cdot g_0 + 3\, f_2 \cdot g_1 + 3 f_1 \cdot g_2 + f_0 \cdot g_3. && (51) \end{aligned}$$

Now, consider the first few evaluations of the binomial theorem

$$\begin{aligned} (a+b) &= a^1 .\, b^0 + a^0 .\, b^1 && (52) \\ (a+b)^2 &= a^2 .\, b^0 + 2\, a^1 .\, b^1 + a^0 .\, b^2 && (53) \\ (a+b)^3 &= a^3 .\, b^0 + 3\, a^2 .\, b^1 + 3 a^1 .\, b^2 + a^0 .\, b^3. && (54) \end{aligned}$$

Generally, for any $m \in \mathbb{N}$, the binomial theorem is stated as

$$(a+b)^m = \sum_{i=0}^{m} \binom{m}{i} a^{m-i} b^i. \qquad (55)$$

We can easily see the effect of multiplying both sides of (55) by the term $(a+b)$ as

$$\begin{aligned} (a+b)^{m+1} &= \sum_{i=0}^{m} \binom{m}{i} \left[a^{m-i} b^i\right] (a+b) \\ &= \sum_{i=0}^{m} \binom{m}{i} \left[a^{m-i+1} b^i + a^{m-i} b^{i+1}\right] \\ &= \sum_{i=0}^{m+1} \binom{m+1}{i} a^{m-i+1} b^i. && (56) \end{aligned}$$

Now, consider taking the partial derivative with respect to α of the term $f_{m-i} \cdot g_i$

$$\begin{aligned} \frac{\partial}{\partial \alpha} \{f_{m-i} \cdot g_i\} &= \frac{\partial f_{m-i}}{\partial \alpha} . g_i + f_{m-i} \cdot \frac{\partial g_i}{\partial \alpha} \\ &= f_{m-i+1} \cdot g_i + f_{m-i} \cdot g_{i+1}. && (57) \end{aligned}$$

Consequently, we from (49)-(57) we can establish the following analogy: the process of taking successive partial derivatives with respect to α is equivalent to successive multiplications by a binomial. Hence, we can make the following associations

$$\begin{aligned} f_m &\longleftrightarrow a^m \\ g_i &\longleftrightarrow b^m \\ \frac{\partial}{\partial \alpha} (\centerdot) &\longleftrightarrow \times (a+b) \\ g_1 = f_0 \cdot g_0 &\longleftrightarrow (a+b)^0, \end{aligned}$$

which yields (47). ■

REFERENCES

[1] N. Sinha and G. Lastman, “Identification of continuous-time multivariable systems from sampled data,” *Int. J. Control*, vol. 35, no. 1, p. 117126, 1982.

[2] H. Garnier, M. Mensler, and A. Richard, “Continuous-time model identification from sampled data: implementation issues and performance evaluation,” *Int. J. Control*, vol. 76, no. 13, pp. 1337–1357, 2003.

[3] M. Matsubara, Y. Usui, and S. Sugimoto, “Identification of continuous-time MIMO systems via sampled data,” *Control-Theory and Advanced Technology*, vol. 2, no. 5, pp. 1119–1136, October 2006.

[4] J. Schoukens, R. Pintelon, and H. V. Hamme, “Identification of linear dynamic systems using piecewise constant excitations: use, misuse and alternatives,” *Automatica*, vol. 30, no. 7, pp. 1153–1169, 1994.

[5] L. Ljung, *System Identification Theory for the User*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 1999.

[6] T. Chen and D. Miller, “Reconstruction of continuous-time systems from their discretizations,” *IEEE Trans. on Automatic Control*, vol. 45, no. 10, pp. 1914–1917, 2000.

[7] K. Astrom, P. Hagander, and J. Sternby, “Zeros of sampled systems,” *Automatica*, vol. 20, no. 1, pp. 31–38, 1984.

[8] I. Kollar, G. Franklin, and R. Pintelon, “On the equivalence of z-domain and s-domain models in system identification,” in *Proc. IEEE Instrumentation and Measurement Technology Conference’96*, Brussels, Belgium, June 1996, pp. 14 – 19.

[9] J. Nemeth and I. Kollar, “Step-invariant transform from z- to s-domain: A general framework,” in *Proc. IEEE Instrumentation and Measurement Technology Conference’00*, Baltimore, MD, May 2000, pp. 902 – 907.

[10] A. Kuznetsov and D. Clarke, “Simple numerical algorithms for continuous-to-discrete and discrete-to-continuous conversion of the systems with time delays,” in *10th IFAC Symposium on System Identification Sysid’94*, vol. 3, Copenhagen, Denmark, July 1994, pp. 633–638.

[11] NI, *LabVIEW Control Design Toolkit User Manual*, August, 2007.

[12] G. Franklin, J. Powell, and M. Workman, *Digital Control of Dynamic Systems Systems*, 3rd ed. Menlo Park, CA: Addison Wesley, 1997.

[13] B.Wittenmark, K. Astrom, and K. Arzen, “Computer control: An overview,” in *IFAC Professional Briefs*, Lund Institute of Technology, 2002, pp. 15–16.

[14] L. Shieh, J. Tsai, and S. Lian, “Determining continuous-time state equations from discrete-time state equations via the principal qth root method,” *IEEE Trans. on Automatic Control*, vol. 31, no. 5, pp. 454–457, 1986.

[15] V. Feliu, J. A. Cerrada, and C. Cerrada, “An algorithm to compute the continuous state model from its equivalent discrete model,” *Control-Theory and Advanced Technology*, vol. 4, no. 2, pp. 231–241, 1988.

[16] V. Feliu, “A transformation algorithm for estimating system laplace transform from sampled data,” *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 16, no. 1, pp. 168–173, 1986.

[17] NI, *LabVIEW System Identification Toolkit User Manual*, August, 2006.