Complete identification of dynamic physical systems using a five parameter model

Deepyaman Maiti*, Mithun Chakraborty* and Amit Konar

Department of Electronics and Tele-Communication Engineering,
Jadavpur University,
188 Raja S.C. Mallik Road, Kolkata 700 032, India
E-mail: deepyamanmaiti@gmail.com
E-mail: mithun.chakra108@gmail.com
E-mail: konaramit@yahoo.co.in
*Corresponding authors

Abstract: System identification is important in science and engineering. This contribution proposes a simple yet elegant scheme for identifying a fractional order dynamic system based on its observed response to a standard excitation. Our approach consists in obtaining and solving a set of simultaneous linear equations connecting the parameters to yield the desired estimates. To minimise the effect of noise, we have performed an in-depth analysis of the Grunwald-Letnikov definition and, as far as we know, are the first ones to do so. Results show that the proposed method offers a very high degree of accuracy even for erroneous data.

Keywords: differintegral; fractional-order system; Grunwald-Letnikov definition; system identification.


Biographical notes: Deepyaman Maiti is currently in the fourth year of his undergraduate course in Electronics and Telecommunication Engineering at Jadavpur University, Kolkata, India. He is doing his research work under the guidance of Amit Konar. His research interests include fractional order control, system identification techniques, robotics and applications of soft computing algorithms.

Mithun Chakraborty is currently in the fourth year of his undergraduate course in Electronics and Telecommunication Engineering at Jadavpur University, Kolkata, India. He is doing his research work under the guidance of Amit Konar. His research interests include applications of stochastic optimisation algorithms, statistical data mining techniques, control engineering, robotics and cellular automata.

Amit Konar is currently a Professor in the Department of Electronics and Tele-Communication Engineering, Jadavpur University, Kolkata. He is also the Joint Coordinator of the Jadavpur University Excellence Program in Cognitive Science. He has supervised ten PhDs and around 60 MTech theses in the broad area of Artificial Intelligence. He has published over 150 papers and five books from top international publishing houses. He was a recipient of...
the 1997–2000 Career Award for Young Scientists, offered by the All India Council for Technical Education (AICTE) for his significant contributions in teaching and research on Artificial Intelligence. He is also in the Editorial board of several international journals.

1 Introduction

Proper estimation of the parameters of a real process, fractional or otherwise, is a challenge to be encountered in the context of system identification (Norton, 1986; Ljung, 1987; Soderstrom and Stoica, 1988; Johansson, 1992). Accurate knowledge of the parameters of a system is often the first step in designing controllers. Many statistical and geometric methods such as regression models (Draper and Smith, 1981), least squares estimation (Lawson and Hanson, 1974; Wellstead and Edmunds, 1975) and recursive identification (Ljung and Soderstrom, 1983; Goodwin and Sin, 1984) are widely used for real-time parameter estimation.

The problem of parameter estimation becomes more difficult for a fractional order system compared to an integral order one. The real world objects or processes that we need to estimate are generally of fractional order (Torvik and Bagley, 1984). A typical example of a non-integer (fractional) order system is the voltage-current relation of a semi-infinite lossy RC line or diffusion of heat into a semi-infinite solid, where heat flow $q(t)$ is equal to the half-derivative of temperature $T(t)$ (Podlubny et al., 1995):

$$\frac{d^{0.5}T(t)}{dt^{0.5}} = q(t).$$

Electronic systems composed of quality electronic elements are usually of fractional order. However, the usual practice when dealing with such a fractional order process has been to use an integer order approximation. In general, this approximation can cause significant differences between a real system and its mathematical model.

The fractional order of the system was ignored because of the non-existence of simple mathematical tools for the description of such systems. Since major advances have been made in this area recently, it is possible to consider also the real order of the dynamical systems. Such models are more adequate for the description of dynamical systems with distributed parameters than integer-order models with concentrated parameters. With regard to this, in the task of identification, it is necessary to consider also the fractional-order of the dynamical system.

Most classical identification methods cannot cope with fractional order transfer functions. Yet, this challenge must be overcome if we want to design a proper adaptive or self-tuning fractional order controller. In a previous publication (Maiti et al., 2008), we obtained very good results while designing a fractional PID controller for a static fractional order process.

If we can design an accurate ‘Parameter Identifier’ block in Figure 1, then we can have claim to a true self-tuning fractional order controller. As proved elsewhere (Milos and Martin, 2006; Podlubny et al., 2003; Petras, 1999), fractional order controllers are much superior to their traditional integral counterparts. Need for design of adaptive
controllers gives an impetus to finding accurate schemes for system identification. This is the motivation for the present work.

**Figure 1** A general block diagram of a self-tuning controller

Computation of the transfer characteristics of the fractional order dynamic systems has been the subject of several publications (Dorcak et al., 1996; Podlubny, 1994, 1999; Dorcak, 1994; Chen et al., 1990; Hartley and Lorenzo, 2003; Thirry and Jean-Claude, 2004; Djamah et al., 2008; Ikeda et al., 2002; Sabatier et al., 2006; Enacheanu et al., 2006; Venkatesh, 2003), e.g., by numerical methods (Dorcak, 1994), as well as by analytical methods (Podlubny, 1994). In this paper we propose a method for parameter identification of a fractional order system for a chosen structure of the model using fractional calculus theory to obtain simultaneous equations relating the unknown parameters and then solving these equations to obtain accurate estimates. This method enables us to work with the actual fractional order process rather than an integer order approximation. Using it in a system with known parameters will do the verification of the correctness of the identification. We first consider that the fractional powers are constant and display the accuracy of the proposed method both when random corruptions are absent and present. Then we remove this limitation and propose two alternative schemes by which a fractional order system can be completely identified with a high degree of accuracy even in presence of significant quantities of error in the readings.

The remainder of the paper is organised as follows. Section 2 seeks to place our contribution in the context of existing literature on the subject. Section 3 highlights the fundamentals of the Grunwald-Letnikov formula for numerical computation of fractional differintegral. Section 4 details the deterministic method adopted for the identification of a system with known and constant fractional powers $s$. In Section 5, we delve into the intricacies of the Grunwald-Letnikov definition and hence propose a transformation which, when applied to the transfer function, will give accurate parameter estimates even in the presence of significant amounts of error. Section 6 demonstrates, with concrete examples, the application of the proposed scheme to both a noise-free and a noisy system. In Section 7, we put forward and illustrate two novel algorithms for the complete identification of a system, whose fractional powers are not known precisely but vary over a considerably wide range. Section 8 concludes the paper.

### 2 Literature review

It can be realised that classical identification methods cannot be used to deal with fractional order transfer functions, although the basic concept of principle of least squares
(Gauss, 1963) has been successfully used by many researchers. Important researches in the field of fractional order system identification are summarised.

In Chen et al. (1990), an approach to model complex non-linear systems using multi-layered neural networks is presented. This paper investigated the identification of discrete-time nonlinear systems using neural networks with a single hidden layer. New parameter estimation algorithms were derived for the neural network model based on a prediction error formulation.

The identification of fractional and integer-order systems using the concept of continuous order-distribution was proposed in Hartley and Lorenzo (2003). Based on the ability to define systems using continuous order-distributions, it was shown that frequency domain system identification could be performed. Least-squares techniques were applied to discretised order-distributions. Methods for properly discretising the order-distribution were also presented.

A method for modelling, simulation and identification of fractional systems in the time domain was presented in Thierry and Jean-Claude (2004). The fractional system was modelled by a state-space representation, replacing conventional integration by a fractional one with the help of a non-integer integrator. This operator itself was approximated by a \( N \)-dimensional system composed of an integrator and of a phase-lead filter. An output-error technique was used in order to estimate the parameters of the model, including the fractional order \( N \). Finally, this methodology was applied to the modelling of the dynamics of a real heat transfer system.

Modeling, simulation, identification and model reduction of non-integer systems in time domain was discussed in Djamah et al. (2008). The fractional order system simulation was based on a fractional integrator operating on a limited spectral range. This allowed approximating the fractional system by an integer state-space representation of high dimension. An output error method was used to perform the model parameters identification including the fractional order. In this paper, an iterative non-linear programming method was used to perform the approximation reduction.

In Ikeda et al. (2002), system identification methods for flexible structures described by fractional differential equations were proposed. Fractional differential equations systems were proved to be suitable to make precise dynamical model of such flexible structures. The authors proposed the identification methods for the fractional differential equations systems by using FIR- and ARMA-type filters.

The application of fractional system identification to lead acid battery state of charge estimation was studied in Sabatier et al. (2006). The fractional behaviour of lead acid batteries was justified. A fractional system identification method was presented and a new fractional model of the battery was proposed. Based on parameter variations of this model, a state of charge estimation method was presented. Validation tests of this method on unknown state of charge were carried out to highlight that the proposed estimation method gave a state of charge estimation with an error close to 5% whatever the operating temperature.

In Enacheanu et al. (2006), a new modelling of electrical networks based on fractional order systems was proposed. The algorithm of identification of such systems was applied on a particular network presenting a fractal behaviour on a restricted frequency range.

In the literature reviewed, fractional order models for various real-life systems have been suggested which have three or five parameters. For example, Petras and Vinagre (2002) have considered a three-parameter model for a real-life
controlled object (heat solid + pyrometer) with two coefficients and one fractional power in the transfer function. Moreover, Podlubny et al. (1999) have described the dynamics of a heat furnace by a fractional order differential equation whence a five-parameter model can be deduced by simple Laplace transformation. In our paper, we have dealt with a generalised five-parameter model to which real systems like the last one mentioned conform.

In the light of these past works, the unique points of our research can be summarised as follows. The foundation of our thesis is in mathematics, it does not make use of neural networks or any stochastic optimisation algorithms. Although our scheme is designed to identify fractional order systems, it can very well be used to identify an integral order system. Thus, we propose a generalised scheme for system identification. Also, if we consider a system is to be identified which has constant fractional powers (and dynamic coefficients), then we do not need to make use of the least squares principle at all. In fact, even in the case of complete identification, we do not require to know the ranges of variation of the coefficient parameters. It may be understood that this is a significant advantage. Finally, and perhaps most importantly, our method is designed to accurately identify a system even when the readings from the actual system, with which the identification is made, is corrupted to a significant extent. Thus we propose an identification method for a realistic scenario. It is also germane to mention here that, in course of solving the problem in question, we have performed a rigorous and complex mathematical analysis (presented in Section 5), which is an original work in its own right, as far as our knowledge goes.

3 Fractional calculus definitions

The fractional calculus is a generalisation of integration and derivation to non-integer order operators. At first, we generalise the differential and integral operators into one fundamental operator \( _{a}^{D}t^{\alpha} \) where:

\[
_{a}^{D}t^{\alpha} = \begin{cases} 
\frac{d^\alpha}{dt^\alpha}, & \Re(\alpha) > 0 \\
1, & \Re(\alpha) = 0 \\
\int (d\tau)^{-\alpha}, & \Re(\alpha) < 0
\end{cases}
\]  

(1)

The two definitions used for fractional differintegral are the Riemann-Liouville definition (Oldham and Spanier, 1974) and the Grunwald-Letnikov definition (Oldham and Spanier, 1974).

The Riemann-Liouville definition is given as

\[
_{a}^{D}t^{\alpha} f(t) = \frac{1}{\Gamma(n - \alpha)} \frac{d^n}{dt^n} \int_{a}^{t} \frac{f(\tau)}{(t - \tau)^{n-\alpha}} d\tau
\]  

(2)

for \((n-1 < \alpha < n)\) and \(\Gamma(\alpha)\) is Euler’s gamma function.
The Grunwald-Letnikov definition is

\[ D^\alpha f(t) = \lim_{h \to 0} \frac{1}{h^\alpha} \sum_{j=0}^{\left\lfloor \frac{t}{h} \right\rfloor} (-1)^j \binom{\alpha}{j} f(t - jh) \]  

(3)

where \([y]\) means the integer part of \(y\).

Derived from the Grunwald-Letnikov definition, the numerical calculation formula of fractional derivative can be achieved as

\[ _{-L}D^\alpha_t f(t) = h^{-\alpha} \sum_{j=0}^{\left\lfloor \frac{L}{h} \right\rfloor} b_j f(t - jh) = \sum_{j=0}^{N} \Phi_j(\alpha) f(t - jh) \]  

(4)

where \(L\) is the length of memory and \(N = \lfloor L/T \rfloor\). \(T (<< 1 \text{ s})\), the sampling time always replaces the time increment \(h\) during approximation. The weighting coefficients \(b_j\) can be calculated recursively by

\[ b_0 = 1, \quad b_j = \left(1 - \frac{1+\alpha}{j}\right) b_{j-1}, \quad (j \geq 1). \]  

(5)

Clearly, \(b_1 = (-\alpha) b_0 = -\alpha, b_2 = \frac{(1-\alpha)}{2} b_1 = \frac{(-\alpha)(1-\alpha)}{2}, b_3 = \frac{2-\alpha}{3}, b_4 = \frac{(-\alpha)(1-\alpha)(2-\alpha)}{3!}, \ldots \)

In general,

\[ b_n = \frac{1}{n!} (-\alpha)(1-\alpha)(2-\alpha) \ldots [(n-1)-\alpha], \quad n = 0, 1, 2, \ldots \]

The right hand side of equation (4) is nothing but the weighted sum of a very large number of terms, the weight of the \((t - jh)\)th term being \(\Phi_j(\alpha) = h^\alpha b_j\).

Let us call the function

\[ \Phi_n(\alpha) = \frac{1}{n!} (-\alpha)(1-\alpha)(2-\alpha) \ldots [(n-1)-\alpha], \quad n = 0, 1, 2, \ldots, \quad 0 < h < 1, \quad -\infty < \alpha < \infty \]

the Grunwald-Letnikov function of degree \(n\).

4 Process of identification when fractional powers are constant

We have considered a fractional process whose transfer function is of the form

\[ \frac{1}{a_3 s^\alpha + a_2 s^\beta + a_1} \]  

(as illustrated in Figure 2).

Here, the orders of fractionality \(\alpha\) and \(\beta\) are known. The coefficients \(a_1, a_2\) and \(a_3\) are to be estimated. One important advantage of the proposed scheme is that we do not require to know the ranges of variation of \(a_1, a_2\) and \(a_3\). It should be noted that without loss of generality, we may presume the dc gain to be unity so that the dc gain and its possible fluctuations are included in the coefficients \(a_1, a_2\) and \(a_3\).
Therefore,
\[
\frac{C(s)}{R(s)} = \frac{1}{a_1 s^\alpha + a_2 s^\beta + a_3}.
\]

\[\Rightarrow R(s) = a_1 s^\alpha C(s) + a_2 s^\beta C(s) + a_3 C(s).\]

In time domain,
\[
r(t) = a_1 D^\alpha c(t) + a_2 D^\beta c(t) + a_3 c(t)\]
\[
\Rightarrow r(t) = a_1 T^{-\alpha} \sum_{j=0}^{[t/T]} b_j c(t-jT) + a_2 T^{-\beta} \sum_{j=0}^{[t/T]} b_j c(t-jT) + a_3 c(t).\]

The proposed scheme requires sampled input at time instant \(t\) and sampled outputs at time instants \(t, t-T, t-2T, t-3T, \ldots\) Sampled outputs are required for a time length \(L\) previous to \(t\), \(T\) being the sampling time. Calculation of fractional derivatives and integrals requires the past history of the process to be remembered. So, the more the value of \(L\), the better.

Thus, the values of \(D^\alpha c(t)\) and \(D^\beta c(t)\) can be calculated so that equation (6) reduces to the form \(a_1 p + a_2 q + a_3 r = s\), where \(p, q, r, s\) are constants whose values have been determined.

Let us assume that we have a set of sampled outputs \(c(t)\) from the system for unit step test signal.

That is, we have
\[
u(t) = a_1 D^\alpha c(t) + a_2 D^\beta c(t) + a_3 c(t).\]

Now, there are three unknown parameters, namely \(a_1, a_2\) and \(a_3\). So we need three simultaneous equations to solve from them. One equation is (8). We will integrate both sides of equation (8) to get \(\int u(t) dt = \int [a_1 D^\alpha c(t) + a_2 D^\beta c(t) + a_3 c(t)] dt\) which gives us
\[
r(t) = a_1 D^{\alpha-1} c(t) + a_2 D^{\beta-1} c(t) + a_3 D^{-1} c(t)\]

where \(r(t)\) signifies unit ramp input and \(c(t)\) is the output due to unit step input. Thus we have derived a second equation relating \(a_1, a_2\) and \(a_3\).

The third equation will be obtained by integrating both sides of equation (9). This gives us
\[
p(t) = a_1 D^{\alpha-2} c(t) + a_2 D^{\beta-2} c(t) + a_3 D^{-2} c(t)\]

where \(p(t)\) signifies parabolic input and \(c(t)\) is the output due to unit step input.

It can be seen that equations (8)–(10) are distinct equations in \(a_1, a_2\) and \(a_3\). So we can solve them simultaneously to identify the three unknown parameters \(a_1, a_2\) and \(a_3\).
As we have displayed elsewhere, direct application of the above scheme gives very satisfactory results when the readings $c(t)$ are accurate. If we now add an error component $e(t)$ to $c(t)$ to have a distorted output waveform $c(t) = c(t) + e(t)$ from which we want to make our identification, equation (8) will be transformed to

$$u(t) = a_1D^\alpha[c(t) + e(t)] + a_2D^\beta[c(t) + e(t)] + a_3[c(t) + e(t)].$$

(11)

So equation (11) will not give an accurate relation between $a_1$, $a_2$ and $a_3$ due to the presence of the terms $a_1D^\alpha e(t)$, $a_2D^\beta e(t)$ and $a_3 e(t)$. Hence, the equations obtained by applying the transformation $c(t) = c(t) + e(t)$ on equations (9) and (10) will also be inaccurate. Our aim will be to minimise this inaccuracy by reducing the effects of the error waveform as far as possible.

One significant fact we observed is that for the same random error waveform $e(t)$, $D^\alpha e(t) \ll D^\alpha e(t)$ if $\alpha_1 < 0$ and $\alpha_2 > 0$ when if effect, $D^\alpha e(t)$ becomes an integration.

A rigorous mathematical proof explaining this observation is presented in the next section. For now, a philosophical explanation may be put forward as follows. The physical significance of differentiation is the slope of the function we want to differentiate (although this is not strictly the case for fractional differentiation), whereas integration deals with the area under the curve. The random error component that we considered consists of fluctuations having both positive and negative values. Thus, an integration operation over this error component can be expected to yield quite a low value, since many of the areas with opposing signs should nullify the effects of each other. On the other hand integration of the output waveform will give a high positive result, since the output waveform is restricted to assume non-negative values only. So the effect due to the error component is minimised.

However, we cannot say anything definite about the results of differentiation operations; in fact, no pattern can be ascribed to the result obtained by differentiating either the output or the error waveforms.

It is evident that two transformations (integration/differentiation) must be applied to both sides of equation (11) to get the two additional equations in $a_1$, $a_2$ and $a_3$ but the choice of the operations (integration/differentiation) will directly affect the orders of the fractional differintegrals to be computed and these will, in turn, influence the effect that $e(t)$ will have on the values of the unknown quantities determined from this set of linear equations. In other words, we must ascertain the dependence of $D^\alpha x(t)$ on $\alpha$ for $x(t) = c(t)$ or $e(t)$ and this calls for a detailed study of the Grunwald-Letnikov function

$$\Phi_n(\alpha) = \frac{h^{-\alpha}}{n!}(-\alpha)(1-\alpha)(2-\alpha)\ldots((n-1)-\alpha), \quad n = 0, 1, 2, \ldots, 0 < h < 1,$$

$$-\infty < \alpha < \infty.$$

This is dealt with in the following section.

5 Detailed study of Grunwald-Letnikov function $\Phi_n(\alpha)$

For $n = 0$, $\Phi_0(\alpha) = h^{-\alpha}$ which, for $0 < h < 1$, is nothing but a rising exponential function and has a well-known nature:
Complete identification of dynamic physical systems

- \( \Phi_\alpha(\alpha) \) is strictly monotonically increasing from \(-\infty\) at \( \alpha \to -\infty \) to \( +\infty \) at \( \alpha \to +\infty \).

- \( \Phi_0(0) = 1 \).

We are actually interested in the cases where \( n \geq 1 \).

**Case 1: \( \alpha < 0 \)**

We put \( -\alpha = \mu \). Let \( f_n(\mu) = \Phi_\alpha(\alpha) \) for any given \( n \geq 1 \). Then,

\[
\begin{align*}
f_n(\mu) &= \frac{h^\mu}{n!} \mu(1 + \mu)(2 + \mu) \ldots (n - 1) + \mu \\
&= \frac{h^\mu}{n!} g_n(\mu)
\end{align*}
\]

where \( g_n(\mu) = \mu(1 + \mu)(2 + \mu) \ldots (n - 1) + \mu \).

Differentiating \( g_n(\mu) \) with respect to \( \mu \),

\[
g'_n(\mu) = (1 + \mu)(2 + \mu) \ldots (n - 1) + \mu + \mu(2 + \mu)(3 + \mu) \ldots (n - 1) + \mu
\]

\[
= g_n(\mu) \left[ \frac{1}{\mu} + \frac{1}{1 + \mu} + \frac{1}{2 + \mu} + \ldots + \frac{1}{(n - 1) + \mu} \right]
\]

(13)

Thus,

\[
f'_n(\mu) = \frac{h^\mu}{n!} \ln(h) g_n(\mu) + \frac{h^\mu}{n!} g'_n(\mu), \text{ from equation (12)}
\]

(14)

Obviously,

- \( \Psi'_n(\mu) \to \infty \) as \( \mu \to 0 \); \( \Psi'_n(\mu) \to 0 \) as \( \mu \to \infty \);

\( \Psi'_n(\mu) \) is a strictly monotonically decreasing function of \( \mu \) for \( 0 < \mu < \infty \).

Thus, over the interval \( 0 < \mu < \infty \), \( \Psi_\alpha(\mu) \) assumes every possible positive real value once and only once.

So, for any given \( n \), there exists one and only one positive value of \( \mu \) (dependent on \( n \)), say \( \mu_{n,\alpha} \), such that \( \Psi_\alpha(\mu_{n,\alpha}) = \gamma \) (as \( \gamma > 0 \)).
Therefore, \( f'_{n} (\mu_{m,n}) = 0 \).

Now, evidently,

\[
\Psi_{n}(0) = 0 \quad \forall \quad n \geq 1 \quad \text{and} \quad f'_{n}(\mu) > 0 \quad \forall \quad \mu > 0.
\]

Again, as \( \Psi_{n}(\mu) \) is monotonically decreasing for positive \( \mu \) and \( \Psi_{n}(\mu) = \gamma \) at \( \mu = \mu_{m,n} \), we must have

\[
\Psi_{n}(\mu) > \gamma \quad \text{for} \quad 0 < \mu < \mu_{m,n} \quad \text{and} \quad \Psi_{n}(\mu) < \gamma \quad \text{for} \quad \mu > \mu_{m,n}.
\]

Thus, (14)–(16) imply that

\[
f'_{n}(\mu) > 0 \quad \text{for} \quad 0 < \mu < \mu_{m,n} \quad \text{and} \quad f''_{n}(\mu) < 0 \quad \text{for} \quad \mu > \mu_{m,n}.
\]

Moreover,

\[
\lim_{\mu \to \infty} f_{n}(\mu) = \frac{1}{n!} \lim_{\mu \to \infty} \frac{g_{n}(\mu)}{h^{-\mu}},
\]

where \( g_{n}(\mu) \to \infty \) as \( \mu \to \infty \) and, also, \( h^{-\mu} \to \infty \) as \( \mu \to \infty \) since \( 0 < h < 1 \).

Applying L’Hospital’s Rule successively \( n \) times, it can be shown that

\[
\lim_{\mu \to \infty} f_{n}(\mu) = 0.
\]

From equations (15), (17) and (18), we conclude that, for \( 0 \leq \mu < \infty \), \( f_{n}(\mu) \) rises strictly monotonically from zero at \( \mu = 0 \) to a local maximum at \( \mu = \mu_{m,n} \), then decreases strictly monotonically with further increase in \( \mu \) beyond \( \mu_{m,n} \) and asymptotically approaches zero. Thus, for non-negative \( \mu \), the lower bound of \( f_{n}(\mu) \) is 0 and its upper bound is \( f_{n}(\mu_{m,n}) \).

To estimate \( \mu_{m,n} \), we first try to solve, for various possible values of \( n \), the equation

\[
\Psi_{n}(\mu) = \gamma.
\]

For \( n = 1 \), the equation (19) reduces to \( 1/\mu = \gamma \) which has only one root at

\[
\mu = 1/\gamma.
\]

For \( n = 2 \), the equation (19) reduces to \( \gamma \mu^{2} + (\gamma - 2)\mu - 1 = 0 \) with exactly one positive root at

\[
\mu = \frac{1}{\gamma} + \left( \frac{1}{4} + \frac{1}{\gamma^{2}} - \frac{1}{2} \right)^{1/2} \frac{1}{\gamma}.
\]

In general, for \( n > 2 \), we observe that

\[
\Psi_{n} \left( \frac{1}{\gamma} \right) = \gamma + \left[ \frac{1}{1+1/\gamma} + \frac{1}{2+1/\gamma} + \cdots + \frac{1}{(n-1)+1/\gamma} \right] > \gamma \quad \text{as} \quad \gamma > 0
\]
for $\mu < 1/\gamma$,

$$\Psi_n(\mu) = \frac{1}{1+\mu} + \frac{1}{2+\mu} + \ldots + \frac{1}{(n-1)+\mu} > \gamma + \left[ \frac{1}{1+1/\gamma} + \frac{1}{2+1/\gamma} + \ldots + \frac{1}{(n-1)+1/\gamma} \right]$$

$$> \gamma \text{ as } \gamma > 0.$$  \hspace{1cm} (23)

But, as already shown, for any given $n \geq 1$, there exists exactly one value of $\mu$, viz. $\mu_{m,n}$, such that

$$\Psi_n(\mu_{m,n}) = \gamma.$$ \hspace{1cm} (24)

Thus, equation (20) indicates that $\mu_{m,1} = 1/\gamma$ and equations (21)–(24) show that $\mu_{m,n} > 1/\gamma$ for $n > 1$.

Thus,

$$h^{\mu_{m,n}} \leq h^{1/\gamma}, n \geq 1 \text{ as } 0 < h < 1.$$  \hspace{1cm} (25)

In order to provide an estimate of $f_n(\mu_{m,n})$, we apply the A.M.–G.M. inequality on the $n$ distinct positive quantities $\mu_{m,n}, (1 + \mu_{m,n}), (2 + \mu_{m,n}), \ldots, ((n-1) + \mu_{m,n})$ to obtain

$$f_n(\mu_{m,n}) = h^{\mu_{m,n}/n} \mu_{m,1} = h^{1/\gamma} \left( \frac{1}{\gamma} \right) = e^{-1} \gamma = \eta, \text{ say}.$$  \hspace{1cm} (26)

Now, putting back $\mu = -\alpha$ and $f_n(\mu) = \Phi_n(\alpha)$, we can now state that, for $-\infty < \alpha < 0$, $\Phi_n(\alpha)$ rises monotonically from zero at $-\infty$ to a maximum at $\alpha = \alpha_{m,n} = -\beta_{m,n}$ and, with further increase in $\alpha$ it falls monotonically to zero at $\alpha = 0$, i.e., $\Phi_n(\alpha)$ is positive and bounded within 0 and some $\Phi_n(\alpha_{m,n})$. 

Complete identification of dynamic physical systems

199
Figures 3–6 clearly illustrate the nature of $\Phi_n(\alpha)$ for negative $\alpha$.

$$\Phi_{n+1}(\alpha) = \frac{h^{\beta}}{(n+1)!} (-\alpha)(1-\alpha)(2-\alpha)\ldots[(n-1)-\alpha](n-\alpha)$$

$$= \Phi_n(\alpha) \frac{(n-\alpha)}{n+1}$$

$$\therefore \Phi_{n+1}(\alpha) - \Phi_n(\alpha) = \Phi_n(\alpha) \left[ \frac{(n-\alpha)}{n+1} - 1 \right] = \Phi_n(\alpha) \left[ \frac{-\alpha-1}{n+1} \right].$$

As, for $\alpha < 0$, $\Phi_n(\alpha) > 0$ for any positive integer $n$, we conclude that $\Phi_{n+1}(\alpha) < \Phi_n(\alpha)$ for $-1 < \alpha < 0$ but $\Phi_{n+1}(\alpha) > \Phi_n(\alpha)$ for $\alpha < -1 \forall n$, i.e., $\Phi_n(\alpha)$ decreases with increase in $n$ for any $-1 < \alpha < 0$ but increases with increase in $n$ for any $\alpha < -1$. (26)

It is interesting to note that at $\alpha = -1$, $\Phi_n(-1) = (h/n!)(1+1)(2+1)\ldots(n-1+1) = (h/n!)(n!) = h$ $\forall n = 1, 2, 3, \ldots$ so that at $\alpha = -1$, a crossover in the nature of $\Phi_n(\alpha)$ may be said to occur.

This means that, in the interval $-1 < \alpha < 0$, the maximum possible value of $\Phi_n(\alpha)$ occurs for $n = 1$ and is given by $\Phi_1(\alpha_{n+1}) = \eta$.

Again, $\frac{\Phi_{n+1}(\alpha) - \Phi_n(\alpha)}{\Phi_n(\alpha)} = \frac{-\alpha-1}{n+1}$ $\to 0$ as $n \to \infty$, for any finite $\alpha$.

This implies that the relative change in the value of $\Phi_n(\alpha)$, for a given $\alpha$, becomes smaller and smaller as $n$ becomes larger and larger. (27)

Figure 3 Plots of $\Phi_n(\alpha)$ vs. $\alpha$ for $\alpha < 0$, $n = 0, 1$ (see online version for colours)
Complete identification of dynamic physical systems

Figure 4  Plots of $\Phi_n(\alpha)$ vs. $\alpha$ for $\alpha < 0$, $n = 1, 2, 5, 10, 100, 1000$ (see online version for colours)

Figure 5  Plots of $\Phi_n(\alpha)$ vs. $\alpha$ for $\alpha < 0$, $n = 1000, 2000, 5000, 5500, 6000$ (see online version for colours)

Figure 6  Plots of $\Phi_n(\alpha)$ vs. $\alpha$ for $\alpha < 0$, $n = 6000, 7000, 8000, 8500, 9000, 9500, 10000$ (see online version for colours)
Now, getting back to \( D^\alpha x(t) = \sum_{j=0}^{N} \Phi_j(\alpha)x(t-jh) \), we reiterate that, for any negative \( \alpha \), the weights \( \Phi_j(\alpha) \) are all positive.

For \(-1 < \alpha < 0\), the value of \( \Phi_j(\alpha) \) decreases as \( j \) increases, vide equation (26), and this value for any \( j \) can never exceed \( \eta \), vide equation (25). In our case, we take the sampling interval as 0.001 s so that \( \gamma = -\ln(h) = -\ln(0.001) \approx 6.908 \). Thus, \( \eta = \exp(-1)/\gamma \approx 0.0532 << 1 \). For any \( j \), \( e(t-jh) \) is itself very small in magnitude so that multiplication with \( \Phi_j(\alpha) \) will make it even smaller without altering its sign. Finally, as the large number (10,000 in our case) of error-values is randomly distributed in the small interval \([-\varepsilon_{max}, \varepsilon_{max}] \) where \( \varepsilon_{max} << 1 \), positive and negative values occur with nearly the same frequency. This discussion leads to the obvious conclusion that the net effect of the superposition of all these weighted error-values, which is \( D^\alpha e(t) \), can be expected to be very small for \(-1 < \alpha < 0\).

For \( \alpha < -1 \), it is true that the value of \( \Phi_j(\alpha) \) increases as \( j \) increases but actual plots show that, for \( j = 0, 1, 2, \ldots, 10,000 \), this value is never too large: e.g., for negative \( \alpha \), the maximum value of \( \Phi_1(\alpha) \) is 2.80 which occurs at \( \alpha \approx -10.5 \). In fact, for smaller values of \( j \), the weights \( \Phi_j(\alpha) \) are very small (<< 1); for higher values of \( j \), the weights are relatively larger but, as the variation in the value of \( \Phi_j(\alpha) \) decreases with increasing \( j \), vide equation (27), the values of \( \Phi_j(\alpha) \) within a reasonable range of \( j \) are more or less comparable; e.g., at \( \alpha = -4 \), \( \Phi_j(\alpha) \) changes gradually from 0.0360 to 0.0379 (only about 5% change) as \( j \) changes from 6000 to 6100. However, within this interval, the error waveform has 101 samples (random) so that there are more or less evenly distributed positive and negative error-values and, even after multiplication with the respective weights, they will tend to cancel each other, on the average. So, once again, the overall weighted sum is likely to be small.

However, for our system, the output \( c(t) \) is always positive and may assume values greater than unity. Moreover, as all the weights are positive, the weighted sample values (however small owing to the smallness of the weights) will go on accumulating so that the weighted sum may be expected to be substantial.

Thus, we may not be unjustified in expecting that, usually, \( |D^\alpha e(t)| << |D^\alpha c(t)| \) for any negative \( \alpha \).

Case 2: \( \alpha \geq 0 \)

Let us write \( \Phi^\alpha(\alpha) = \frac{h^{-\alpha}}{n!} q(\alpha) \), where \( q(\alpha) = (-\alpha)(1-\alpha)(2-\alpha) \ldots (n-1-\alpha) \) for some given \( n \). Now, for \( \alpha < -n < \alpha < \infty \), \( h^{\alpha} \) (0 < \( h < 1 \)) is always positive and is a strictly monotonically increasing function of \( \alpha \) and, for \( \alpha \geq 0 \), the rate of increase is high (exponential). But, \( q(\alpha) \) has \( n \) zeros at 0, 1, 2, …, \( n-1 \) so that, over the interval \( 0 \leq \alpha \leq n-1 \), \( q(\alpha) \) oscillates on both sides of the \( \alpha \)-axis; in fact, as \( q(\alpha) > 0 \) \( \forall \alpha < 0 \) and \( \alpha = 0 \) is a zero-crossing point of \( q(\alpha) \), we must have \( q(\alpha) < 0 \) for \( 2m < \alpha < 2m+1 \) and \( q(\alpha) > 0 \) for \( 2m+1 < \alpha < 2m+2 \). \( m = 0, 1, 2, \ldots, (n-2)/2 \) or \( (n-3)/2 \) according as \( (n-1) \) is odd or even.

This discussion suggests that, within \( 0 \leq \alpha \leq n-1 \), the function

\[ \Phi_j(\alpha) = \frac{h^{-\alpha}}{n!} q(\alpha) \]

is oscillatory in nature but is modulated by a growing exponential envelope \( h^{-\alpha}/n! \); for \( (n-1) < \alpha < \infty \), \( \Phi_j(\alpha) \) monotonically and sharply increases to \( +\infty \) or decreases to \( -\infty \).
according as \((n - 1)\) is odd or even; i.e., it is unbounded beyond its last zero. However, there do exist some positive values of \(\alpha\) at which the functional values are less than those at certain negative values of \(\alpha\).

Actual plots (Figures 7–10) show that at certain non-negative values of \(\alpha\), the magnitude of \(\Phi_j(\alpha)\) may be extremely large for smaller values of \(j\) although, roughly from \(j = 10\) onwards, the value becomes much smaller for any \(\alpha\). So, it is not unlikely that, in the weighted sum, the terms corresponding to lower values of \(j\) will outweigh those corresponding to higher values of \(j\). Moreover, for positive \(\alpha\), the weights \(\Phi_j(\alpha)\) no longer have the same sign for all \(j\) so that signs of the terms in the weighted summation are no longer determined solely by \(e(t)\) or \(c(t)\). So, \(D^\alpha e(t)\) may work out to a considerable value and may even be comparable to \(D^\alpha c(t)\). In short, for non-negative \(\alpha\), we cannot ascribe any uniform pattern to the weights \(\Phi_j(\alpha)\) and thus cannot make any generalised and conclusive claims about the relative magnitudes of \(D^\alpha e(t)\) and \(D^\alpha c(t)\).

In the light of this discussion, we contend that, to minimise the effect of error, we must ensure that all the fractional differintegrals to be computed are fractional integrations rather than fractional differentiations.

To support our contention, in Table 1, we tabulate the values of \(D^\alpha e(t)\) for ten different sets of \(e(t)\) with \(\alpha = 1.5, 1.2, 0.9, 0.6, 0.3, -0.3, -0.6, -0.9, -1.2, -1.5\). The amplitude of \(e(t)\) varies between \(-0.01\) and \(0.01\). Length of memory = 10 s, i.e., the fractional derivatives are calculated at time \(t = 10\) s. Sampling rate is once in 0.001 s.

The transfer function of our system is \(1/(a_1 s^\alpha + a_2 s^{\beta} + a_3)\), and as we are well aware, \(\alpha, \beta > 0\) for a practical system, so that in equation (11), the orders of derivation \(\alpha, \beta\) are positive.

**Figure 7** Plots of \(\Phi_n(\alpha)\) vs. \(\alpha\) for \(\alpha > 0\), \(n = 0, 1, 2, 3, 4\) (see online version for colours)
Figure 8  Plots of $\Phi_n(\alpha)$ vs. $\alpha$ for $0 < \alpha < 1$, $n = 1, 2, 3, 4, 5, 10$ (see online version for colours)

Figure 9  Plots of $\Phi_n(\alpha)$ vs. $\alpha$ for $1 < \alpha < 2$, $n = 2, 3, 4, 5, 10$ (see online version for colours)

Figure 10  Plots of $\Phi_n(\alpha)$ vs. $\alpha$ for $2 < \alpha < 3$, $n = 3, 4, 5, 10$ (see online version for colours)
To remedy this, we can perform a simple transformation on the transfer function of the system, which we can write as $s^{-\alpha}(a_1s^{\alpha-\beta} + a_2s^{\beta-\alpha} + a_3s^{\alpha-\beta})$, where $(n-1) < \alpha < n$ and $\alpha > \beta$.

Proceeding as before, we can now obtain our three simultaneous equations as:

\begin{align*}
D^{-\alpha}u(t) &= (a_1D^{-\alpha} + a_2D^{\alpha-\beta} + a_3D^{-\beta})[c(t) + e(t)] \\
D^{-\alpha-1}u(t) &= (a_1D^{-\alpha-1} + a_2D^{\alpha-\beta-1} + a_3D^{-\beta-1})[c(t) + e(t)] \\
D^{-\alpha-2}u(t) &= (a_1D^{-\alpha-2} + a_2D^{\alpha-\beta-2} + a_3D^{-\beta-2})[c(t) + e(t)].
\end{align*}

Table 1 Variation of $D^\alpha e(t)$ with $\alpha$ (The 10 sequences $e(t)$ are consecutive and independent)

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$D^\alpha e(t)$ for derivation order $\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5</td>
<td>1.2</td>
</tr>
<tr>
<td>1</td>
<td>-435.7842</td>
</tr>
<tr>
<td>2</td>
<td>-603.6659</td>
</tr>
<tr>
<td>3</td>
<td>424.4136</td>
</tr>
<tr>
<td>4</td>
<td>-256.3730</td>
</tr>
<tr>
<td>5</td>
<td>-107.8138</td>
</tr>
<tr>
<td>6</td>
<td>642.4164</td>
</tr>
<tr>
<td>7</td>
<td>184.7026</td>
</tr>
<tr>
<td>8</td>
<td>-393.9215</td>
</tr>
<tr>
<td>9</td>
<td>-109.5421</td>
</tr>
<tr>
<td>10</td>
<td>-32.4628</td>
</tr>
</tbody>
</table>

It can now be checked that all orders of derivation are now negative so that we will actually be performing fractional order integrations rather than fractional order differentiations.

## 6 Illustration

Let the process whose parameters are to be estimated is $1/(a_1s^{2.23} + a_2s^{0.88} + a_3)$. The input considered is $r(t) = 1$ i.e., unit step.

Synthetic data for $c(t)$ are created using $a_1 = 0.8$, $a_2 = 0.5$ and $a_3 = 1$, i.e., the values of $c(t)$ are obtained at different time instants (using a MATLAB program) assuming a process with transfer function $1/(0.8s^{2.23} + 0.5s^{0.88} + 1)$. The simultaneous equations corresponding to (12)–(14) are

\begin{align*}
D^{3}u(t) &= (a_1D^{-0.77} + a_2D^{-2.12} + a_3D^{-3})[c(t) + e(t)] \\
D^{4}u(t) &= (a_1D^{-1.77} + a_2D^{-3.12} + a_3D^{-4})[c(t) + e(t)] \\
D^{5}u(t) &= (a_1D^{-2.77} + a_2D^{-4.12} + a_3D^{-5})[c(t) + e(t)].
\end{align*}

Length of memory $L = 10$ s and $T = 0.001$ s is used to calculate the fractional derivatives.
We will display the accuracy of identification when the output readings used to calculate the fractional derivatives are ideal and also when they are erroneous to the extent of a random error component in the range \([-0.05, 0.05]\) in each reading. This error component is quite large since the output response is often below unity. The output response of the system for unit step input is shown both in presence and absence of the error component in Figures 11(a)–(c).

**Figure 11(a)** Unit step response of the actual system

![Figure 11(a)](image)

**Figure 11(b)** Random error component

![Figure 11(b)](image)

**Figure 11(c)** Corrupted response due to addition of error component

![Figure 11(c)](image)
6.1 Ideal case: \( e(t) = 0 \) for all \( t \)

The following derivatives are then calculated numerically using equations (4) and (5):

\[
\begin{align*}
D^{-0.77}c(t) &= 6.1777, & D^{-2.12}c(t) &= 51.3011, & D^{-3}c(t) &= 136.1477, \\
D^{-1.77}c(t) &= 32.2818, & D^{-3.12}c(t) &= 152.6826, & D^{-4}c(t) &= 314.8183, \\
D^{-2.77}c(t) &= 108.0207, & D^{-4.12}c(t) &= 342.4005, & D^{-5}c(t) &= 576.6986.
\end{align*}
\]

The set of simultaneous equations is

\[
\begin{bmatrix}
6.1777 & 51.3011 & 136.1477 \\
32.2818 & 152.6826 & 314.8183 \\
108.0207 & 342.4005 & 576.6986
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3
\end{bmatrix}
=
\begin{bmatrix}
166.7167 \\
416.9167 \\
834.1670
\end{bmatrix}.
\]

After solving we have \( a_1 = 0.8001, a_2 = 0.4996, a_3 = 1.0000 \) as the unknown parameters. The errors in estimating them are, respectively, 0.0125, 0.0800 and 0%.

The summation of the square errors of this process model outputs relative to the output data set for unit step input is 0.0030. The unit step responses of the actual and the estimated systems are shown in Figures 12(a) and (b).

**Figure 12(a)** Unit step responses of actual and estimated system for Section 6.1 (ideal case)

---

**Figure 12(b)** Difference (error) between unit step responses of actual and estimated system for Section 6.1
6.2 Non-ideal case: each element in $e(t)$ is between $-0.05$ and $0.05$

To each reading $c(t)$ is added a random error component varying between $-0.05$ and $0.05$. We proceed as before to obtain the set of simultaneous equations as

$$
\begin{bmatrix}
6.1798 & 51.3179 & 136.1948 \\
32.2919 & 152.7357 & 314.9314 \\
108.0577 & 342.5242 & 576.9207 \\
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
\end{bmatrix}
= 
\begin{bmatrix}
166.7167 \\
416.9167 \\
834.1670 \\
\end{bmatrix}
$$

After solving we have $a_1 = 0.7992$, $a_2 = 0.4996$, $a_3 = 0.9996$ as unknown parameters. The errors in estimating them are, respectively, $0.1000$, $0.0800$ and $0.0400\%$.

The summation of the square errors of this process model outputs relative to the actual output data set for unit step input is $0.0062$. The identification is illustrated in Figures 13(a) and (b).

**Figure 13(a)** Unit step responses of actual and estimated system for Section 6.2 (non-ideal case)

[Graph showing unit step responses of actual and estimated system]

**Figure 13(b)** Difference (error) between unit step responses of actual and estimated system for Section 6.2 (non-ideal case)

[Graph showing difference between actual and estimated system responses]
7 Process of identification when fractional powers are varying

We shall propose two algorithms for system identification in this situation and then perform a comparative analysis between them. We will generate possible process models according to some criteria and then choose the optimum model(s) based on their time responses as obtained by simulations. Let us first define a fitness parameter \( F = \sum_{t=0}^{T} (f(t) - g(t))^2 \) for each process model where \( f(t) \) is the set of outputs obtained from the actual process and \( g(t) \) is the set of outputs generated by a possible process model at the corresponding time instants. Lower the value of \( F \) for a model, the better it is.

7.1 Algorithm 1

We will consider that the fractional power \( \alpha \) can vary in the range \( \{\alpha_{\text{min}}, \alpha_{\text{max}}\} \) and \( \beta \) can vary in the range \( \{\beta_{\text{min}}, \beta_{\text{max}}\} \). We will subdivide each of the two ranges \( \alpha_{\text{max}} - \alpha_{\text{min}} \) and \( \beta_{\text{max}} - \beta_{\text{min}} \) into \( m \) and \( n \) intervals, respectively (\( m \) may or may not be equal to \( n \), depending on our design of the interval lengths.) We will consider as the nominal value of \( \alpha \) and \( \beta \) for each interval as its central point.

That is, we may have \( m \) possible values of \( \alpha \):

\[
\alpha_{\text{min}} + \frac{(\alpha_{\text{max}} - \alpha_{\text{min}})}{2m}, \alpha_{\text{min}} + \frac{3(\alpha_{\text{max}} - \alpha_{\text{min}})}{2m}, \alpha_{\text{min}} + \frac{5(\alpha_{\text{max}} - \alpha_{\text{min}})}{2m}, \ldots, \alpha_{\text{min}} + \frac{(2m-1)(\alpha_{\text{max}} - \alpha_{\text{min}})}{2m}.
\]

Likewise we will have \( n \) possible values of \( \beta \):

\[
\beta_{\text{min}} + \frac{(\beta_{\text{max}} - \beta_{\text{min}})}{2n}, \beta_{\text{min}} + \frac{3(\beta_{\text{max}} - \beta_{\text{min}})}{2n}, \beta_{\text{min}} + \frac{5(\beta_{\text{max}} - \beta_{\text{min}})}{2n}, \ldots, \beta_{\text{min}} + \frac{(2n-1)(\beta_{\text{max}} - \beta_{\text{min}})}{2n}.
\]

Thus the fractional powers \( (\alpha, \beta) \) may assume any of the \( m \times n \) values (or be sufficiently close to be an acceptable estimate). Now, for each of the \( m \times n \) possible fractional power combinations we will find out the coefficients \( a_1, a_2, \) and \( a_3 \) by the method of forming and solving simultaneous equations as already illustrated in Sections 3 and 4. Thus we will have \( m \times n \) possible process models. For each of these we will calculate the fitness \( F \) through simulation. We will select the process model with the least value of \( F \), i.e., with highest fitness.

7.1.1 Pseudo code for Algorithm 1

Let \( \alpha_i \) be the nominal values of \( \alpha \) and \( \beta_j \) be the nominal values of \( \beta \) where \( i \) runs from 1 to \( m \) (both values inclusive) and \( j \) from 1 to \( n \) (both values inclusive). The \( \alpha \) and \( \beta \)-values are stored in two arrays of sizes \( m \) and \( n \), respectively.

For \( \alpha = \alpha_i, \beta = \beta_j, a_1, a_2, a_3 \) are calculated and these are stored as the \((i, j)\)th elements of three different matrices of size \( m \times n \) (one matrix for each coefficient).
Finally for $\alpha = \alpha_i$, $\beta = \beta_j$, $a_i = a_i(i, f)$, $a_j = a_j(i, f)$, $a_k = a_k(i, f)$, the corresponding fitness value $F$ is stored as the $(i, j)$th element of an $m \times n$ matrix.

Now the code for Algorithm 1 may be presented as

```
for $i = 1$ to $m$ (increment step = 1)
    $\alpha = \alpha_i$;
    for $j = 1$ to $n$ (increment step = 1)
        $\beta = \beta_j$;
        calculate $a_i, a_j, a_k$ for $\alpha = \alpha_i, \beta = \beta_j$;
        calculate fitness $F$;
    end
end
identify the smallest element of the matrix $F$ as $F(i, j)$;
```

The best identification is $\alpha = \alpha_i$, $\beta = \beta_j$, $a_i = a_i(i, f)$, $a_j = a_j(i, f)$, $a_k = a_k(i, f)$.

### 7.2 Illustration of Algorithm 1

Assume that $\alpha$ varies from 2.0 to 2.4, i.e., $\alpha_{\text{min}} = 2.0$ and $\alpha_{\text{max}} = 2.4$; $\beta$ varies from 0.7 to 1.1, i.e., $\beta_{\text{min}} = 0.7$ and $\beta_{\text{max}} = 1.1$. Also we let $m = n = 20$.

Therefore the 20 possible values of $\alpha$ are 2.01, 2.03, 2.05, ..., 2.37 and 2.39. The 20 possible values for $\beta$ are 0.71, 0.73, 0.75, ..., 1.07 and 1.09.

Then we compute the solution sets of $a_1, a_2, a_3$ for each of the 400 $(\alpha, \beta)$ values and also note the fitness $F$ of each process model. The process model with the best $F$ is our estimated system model.

The estimated parameters, the corresponding percentage errors in estimating them and the fitness values for the ten best models are given below in Table 2.

<table>
<thead>
<tr>
<th>Model sl. no.</th>
<th>Estimated parameters</th>
<th>Percentage errors from actual process</th>
<th>Fitness $F$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$A$</td>
<td>$\beta$</td>
<td>$a_1$</td>
</tr>
<tr>
<td>1</td>
<td>2.23</td>
<td>0.87</td>
<td>0.8039</td>
</tr>
<tr>
<td>2</td>
<td>2.23</td>
<td>0.89</td>
<td>0.7933</td>
</tr>
<tr>
<td>3</td>
<td>2.21</td>
<td>0.85</td>
<td>0.8141</td>
</tr>
<tr>
<td>4</td>
<td>2.25</td>
<td>0.91</td>
<td>0.7823</td>
</tr>
<tr>
<td>5</td>
<td>2.21</td>
<td>0.83</td>
<td>0.8238</td>
</tr>
<tr>
<td>6</td>
<td>2.25</td>
<td>0.93</td>
<td>0.7709</td>
</tr>
<tr>
<td>7</td>
<td>2.21</td>
<td>0.87</td>
<td>0.8040</td>
</tr>
<tr>
<td>8</td>
<td>2.23</td>
<td>0.85</td>
<td>0.8141</td>
</tr>
<tr>
<td>9</td>
<td>2.23</td>
<td>0.91</td>
<td>0.7824</td>
</tr>
<tr>
<td>10</td>
<td>2.25</td>
<td>0.89</td>
<td>0.7934</td>
</tr>
</tbody>
</table>
As we can see, the estimated model is the model with the best fitness, i.e., the first model. The percentage errors in identification of $\alpha$, $\beta$, $a_1$, $a_2$, $a_3$ are respectively 0, 1.1364, 0.4875, 0.4200, 0.2500. The identification is illustrated in Figures 14(a) and (b).

**Figure 14(a)** Unit step responses of actual and best-estimated system for Algorithm 1

**Figure 14(b)** Difference (error) between unit step responses of actual and best-estimated system for Algorithm 1

### 7.3 Algorithm 2

As in Algorithm 1, we will consider that the fractional power $\alpha$ can vary in the range $[\alpha_{\min}, \alpha_{\max}]$ and $\beta$ can vary in the range $[\beta_{\min}, \beta_{\max}]$. We will subdivide each of the two ranges $(\alpha_{\max} - \alpha_{\min})$ and $(\beta_{\max} - \beta_{\min})$ into $m_1$ and $n_1$ intervals respectively. ($m_1$ may or may not be equal to $n_1$, depending on our design of the interval lengths.) We will consider the nominal value of $\alpha$ and $\beta$ for each interval as its central point.

Thus, the fractional powers $(\alpha, \beta)$ may assume any of the $m_1 \times n_1$ values (or be sufficiently close to be an acceptable estimate). Now for each of the $m_1 \times n_1$ possible fractional power combinations we will find out the coefficients $a_1$, $a_2$ and $a_3$ by the method of forming and solving simultaneous equations as already illustrated in Sections 3 and 4. Thus we will have $m_1 \times n_1$ possible process models. For each of these we will calculate the fitness $F$ through simulation. This completes one sub-run.
Let us define the concept of a temporary memory space (buffer) where we will store the best $p_1$ fitness values and the corresponding models. (The choice of $p_1$ depends on us. Obviously $p_1 \geq 1$). Corresponding to the $p_1$ models, we will have $p_1$ sub-intervals where $\alpha$, $\beta$ may lie. We will sub-divide each of the $\alpha$-intervals and $\beta$-intervals into a suitable number of sub-intervals, say $m_2$ and $n_2$. So now we have $p_1 \times m_2 \times n_2$ possible process models. Once again, we shall compute the values of $F$ and store the best $p_2$ values in the buffer. This completes the second sub-run.

In this way, we shall continue the sub-runs until we are satisfied that the value of $F$ is sufficiently good.

The advantage of this algorithm is two-fold. Firstly, it allows us to search more thoroughly within an interval because the search is performed not once but many times. So, we do not have to waste resources searching in an interval that consistently gives poor fitness values. Secondly, because the search is many-layered, for each sub-run we can take low values of $m$ and $n$ and yet obtain better fitness. So this algorithm is more efficient and is of an adaptive nature. Discussions for further improvement of this algorithm will be presented after the illustration.

### 7.3.1 Pseudo code for Algorithm 2

The pseudo code for implementing Algorithm 2 is presented below. The basic principle is same as that of Algorithm 1, that is, for a certain $(\alpha, \beta)$, calculate $a_1, a_2, a_3$ and the fitness $F$. At the end of a run, choose the best fitness(es).

From Step 2 onwards, the matrices $a_1, a_2, a_3$ and $F$ will be three-dimensional of size $m_2 \times n_2 \times p_1, m_3 \times n_3 \times p_2$, etc. The elements of all matrices are overwritten in each iteration of the do-while loop in Step 2. So the last result (which should be the best result so far) remains.

Define $F_{\text{min}}$ as the required level of fitness. If the best value of $F$ is less than $F_{\text{min}}$ at any stage, the algorithm stops.

************************************************************************

**STEP 1**

subdivide $\alpha$ and $\beta$ ranges into $m_1$ and $n_1$ intervals respectively;

choose the central points of these intervals as the nominal values $\alpha_i$ and $\beta_j$ respectively;

for $i = 1$ to $m_1$ (increment step = 1)

$\alpha = \alpha_i$;

for $j = 1$ to $n_1$ (increment step = 1)

$\beta = \beta_j$;

calculate $a_1, a_2, a_3$ for $\alpha = \alpha_i, \beta = \beta_j$;

calculate fitness $F$;

end

end

if (smallest element of $F < F_{\text{min}}$)
then stop;
identify the \( p_1 \) smallest elements of the matrix \( F \);
identify the corresponding \( p_1 \) intervals of \( \alpha \) and \( \beta \) ranges;
goto STEP 2;

STEP 2

\[ t = 2; \]
do
for \( k = 1 \) to \( p_t - 1 \) (increment step = 1)
subdivide \( \alpha \) and \( \beta \) ranges into \( m_t \) and \( n_t \) intervals respectively;
choose the central points of these intervals as the nominal values \( \alpha(i, k) \) and \( \beta(j, k) \) respectively;
end
for \( k = 1 \) to \( p_t - 1 \) (increment step = 1)
for \( i = 1 \) to \( m_t \) (increment step = 1)
\[ \alpha = \alpha(i, k); \]
for \( j = 1 \) to \( n_t \) (increment step = 1)
\[ \beta = \beta(j, k); \]
calculate \( a_1, a_2, a_3 \) for \( \alpha = \alpha(i, k), \beta = \beta(j, k); \]
calculate fitness \( F \);
end
end
identify the \( p_t \) smallest elements of the matrix \( F \);
identify the corresponding \( p_t \) intervals of \( \alpha \) and \( \beta \) ranges;
\[ t = t + 1; \]
while (smallest element of \( F \) > \( F_{\text{min}} \))

If the index of the smallest element of \( F \) is \( (i, j, k) \), the identified model is \( \alpha = \alpha(i, k), \beta = \beta(j, k), a_1 = a_1(i, j, k), a_2 = a_2(i, j, k), a_3 = a_3(i, j, k) \).

7.4 Illustration of Algorithm 2

Assume that \( \alpha \) varies from 2.0 to 2.4, i.e., \( \alpha_{\text{min}} = 2.0 \) and \( \alpha_{\text{max}} = 2.4 \); \( \beta \) varies from 0.7 to 1.1, i.e., \( \beta_{\text{min}} = 0.7 \) and \( \beta_{\text{max}} = 1.1 \).
7.4.1 Sub-run 1

$m_n = n_c = 4$. The $\alpha$-nominal values are 2.05, 2.15, 2.25 and 2.35. The $\beta$-nominal values are 0.75, 0.85, 0.95 and 1.05. These values are tabulated in Table 3.

<table>
<thead>
<tr>
<th>Interval</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td></td>
</tr>
<tr>
<td>2.0–2.1</td>
<td>2.05</td>
</tr>
<tr>
<td>2.1–2.2</td>
<td>2.15</td>
</tr>
<tr>
<td>2.2–2.3</td>
<td>2.25</td>
</tr>
<tr>
<td>2.3–2.4</td>
<td>2.35</td>
</tr>
<tr>
<td>$\beta$</td>
<td></td>
</tr>
<tr>
<td>0.7–0.8</td>
<td>0.75</td>
</tr>
<tr>
<td>0.8–0.9</td>
<td>0.85</td>
</tr>
<tr>
<td>0.9–1.0</td>
<td>0.95</td>
</tr>
<tr>
<td>1.0–1.1</td>
<td>1.05</td>
</tr>
</tbody>
</table>

We have 16 sets of $\alpha, \beta$. We compute the 16 corresponding fitnesses. The best four models are kept in the buffer, i.e., $p_1 = 4$. The four best models for sub-run 1 are given in Table 4:

<table>
<thead>
<tr>
<th>Estimated parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model sl. no.</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

The four $\alpha$- and $\beta$-sub-intervals in which we will search more thoroughly are illustrated in Table 5:

<table>
<thead>
<tr>
<th>The four intervals to be searched in sub-run 2 of Algorithm 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sl. no.</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

The unit step input responses of the actual process and the best-estimated model after sub-run 1 is given below. This identification is illustrated in Figures 15(a) and (b).
7.4.2 Sub-run 2

Each $\alpha$- and $\beta$-sub-interval is divided into 5 sub-sub-intervals, i.e., $m_2 = n_2 = 5$. Hence there will be $p_1 \times m_2 \times n_2 = 4 \times 5 \times 5 = 100$ process models. This is illustrated in Table 6.

<table>
<thead>
<tr>
<th>Set 1</th>
<th>Interval</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>2.20–2.22</td>
<td>2.21</td>
</tr>
<tr>
<td></td>
<td>2.22–2.24</td>
<td>2.23</td>
</tr>
<tr>
<td></td>
<td>2.24–2.26</td>
<td>2.25</td>
</tr>
<tr>
<td></td>
<td>2.26–2.28</td>
<td>2.27</td>
</tr>
<tr>
<td></td>
<td>2.28–2.30</td>
<td>2.29</td>
</tr>
</tbody>
</table>
Table 6  $\alpha, \beta$ intervals and nominal values used in sub-run 2 of Algorithm 2 (continued)

<table>
<thead>
<tr>
<th>Set 1</th>
<th>Interval</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta$</td>
<td>0.90–0.92</td>
<td>0.91</td>
</tr>
<tr>
<td></td>
<td>0.92–0.94</td>
<td>0.93</td>
</tr>
<tr>
<td></td>
<td>0.94–0.96</td>
<td>0.95</td>
</tr>
<tr>
<td></td>
<td>0.96–0.98</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td>0.98–1.00</td>
<td>0.99</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set 2</th>
<th>Interval</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>2.10–2.12</td>
<td>2.11</td>
</tr>
<tr>
<td></td>
<td>2.12–2.14</td>
<td>2.13</td>
</tr>
<tr>
<td></td>
<td>2.14–2.16</td>
<td>2.15</td>
</tr>
<tr>
<td></td>
<td>2.16–2.18</td>
<td>2.17</td>
</tr>
<tr>
<td></td>
<td>2.18–2.20</td>
<td>2.19</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.70–0.72</td>
<td>0.71</td>
</tr>
<tr>
<td></td>
<td>0.72–0.74</td>
<td>0.73</td>
</tr>
<tr>
<td></td>
<td>0.74–0.76</td>
<td>0.75</td>
</tr>
<tr>
<td></td>
<td>0.76–0.78</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>0.78–0.80</td>
<td>0.79</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set 3</th>
<th>Interval</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>2.20–2.22</td>
<td>2.21</td>
</tr>
<tr>
<td></td>
<td>2.22–2.24</td>
<td>2.23</td>
</tr>
<tr>
<td></td>
<td>2.24–2.26</td>
<td>2.25</td>
</tr>
<tr>
<td></td>
<td>2.26–2.28</td>
<td>2.27</td>
</tr>
<tr>
<td></td>
<td>2.28–2.30</td>
<td>2.29</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.80–0.82</td>
<td>0.81</td>
</tr>
<tr>
<td></td>
<td>0.82–0.84</td>
<td>0.83</td>
</tr>
<tr>
<td></td>
<td>0.84–0.86</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>0.86–0.88</td>
<td>0.87</td>
</tr>
<tr>
<td></td>
<td>0.88–0.90</td>
<td>0.89</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set 4</th>
<th>Interval</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>2.10–2.12</td>
<td>2.11</td>
</tr>
<tr>
<td></td>
<td>2.12–2.14</td>
<td>2.13</td>
</tr>
<tr>
<td></td>
<td>2.14–2.16</td>
<td>2.15</td>
</tr>
<tr>
<td></td>
<td>2.16–2.18</td>
<td>2.17</td>
</tr>
<tr>
<td></td>
<td>2.18–2.20</td>
<td>2.19</td>
</tr>
</tbody>
</table>
So we have 100 sets of \((a, \beta)\). We compute the 100 corresponding fitnesses. The best three models are kept in the buffer, i.e., \(p_2 = 3\). The three best models are given in Table 7:

Table 7  
<table>
<thead>
<tr>
<th>Estimated parameters</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(a)</td>
<td>(\beta)</td>
<td>(a_1)</td>
<td>(a_2)</td>
<td>(a_3)</td>
</tr>
<tr>
<td>Model sl. no.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2.23</td>
<td>0.87</td>
<td>0.8039</td>
<td>0.4979</td>
<td>0.9975</td>
</tr>
<tr>
<td>2</td>
<td>2.23</td>
<td>0.89</td>
<td>0.7933</td>
<td>0.5017</td>
<td>1.0013</td>
</tr>
<tr>
<td>3</td>
<td>2.21</td>
<td>0.85</td>
<td>0.8141</td>
<td>0.4866</td>
<td>0.9952</td>
</tr>
</tbody>
</table>

The three \(a\)- and \(\beta\)-sub-intervals in which we will search more thoroughly are given in Table 8:

Table 8  
<table>
<thead>
<tr>
<th>Sl. no.</th>
<th>(a) nominal value</th>
<th>(a) sub-interval</th>
<th>(\beta) nominal value</th>
<th>(\beta) sub-interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.23</td>
<td>2.22–2.24</td>
<td>0.87</td>
<td>0.86–0.88</td>
</tr>
<tr>
<td>2</td>
<td>2.23</td>
<td>2.22–2.24</td>
<td>0.89</td>
<td>0.88–0.90</td>
</tr>
<tr>
<td>3</td>
<td>2.21</td>
<td>2.20–2.22</td>
<td>0.85</td>
<td>0.84–0.86</td>
</tr>
</tbody>
</table>

The unit step input responses of the actual process and the best-estimated model after sub-run 2 is given below. This identification is illustrated in Figures 16(a) and (b).

Figure 16(a)  
Unit step responses of actual and the best-estimated system for Algorithm 2, sub-run 2.
7.4.3 Sub-run 3

Each $\alpha$- and $\beta$-sub-interval is divided into five sub-sub-intervals, i.e., $m_3 = n_3 = 5$. Hence there will be $p_3 \times m_3 \times n_3 = 3 \times 5 \times 5 = 75$ process models. The $\alpha$- and $\beta$-nominal values are calculated as the central points of the respective intervals. So we have 75 sets of $\alpha, \beta$ as illustrated in Table 9.

<table>
<thead>
<tr>
<th>Table 9</th>
<th>$\alpha, \beta$ intervals and nominal values used in sub-run 3 of Algorithm 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set 1</td>
<td>Interval</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>2.220–2.224</td>
</tr>
<tr>
<td></td>
<td>2.224–2.228</td>
</tr>
<tr>
<td></td>
<td>2.228–2.232</td>
</tr>
<tr>
<td></td>
<td>2.232–2.236</td>
</tr>
<tr>
<td></td>
<td>2.236–2.240</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.860–0.864</td>
</tr>
<tr>
<td></td>
<td>0.864–0.868</td>
</tr>
<tr>
<td></td>
<td>0.868–0.872</td>
</tr>
<tr>
<td></td>
<td>0.872–0.876</td>
</tr>
<tr>
<td></td>
<td>0.876–0.880</td>
</tr>
<tr>
<td>Set 2</td>
<td>Interval</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>2.220–2.224</td>
</tr>
<tr>
<td></td>
<td>2.224–2.228</td>
</tr>
<tr>
<td></td>
<td>2.228–2.232</td>
</tr>
<tr>
<td></td>
<td>2.232–2.236</td>
</tr>
<tr>
<td></td>
<td>2.236–2.240</td>
</tr>
</tbody>
</table>
Table 9  $\alpha$, $\beta$ intervals and nominal values used in sub-run 3 of Algorithm 2 (continued)

<table>
<thead>
<tr>
<th>Set 2</th>
<th>Interval</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta$</td>
<td>0.880–0.884</td>
<td>0.882</td>
</tr>
<tr>
<td></td>
<td>0.884–0.888</td>
<td>0.886</td>
</tr>
<tr>
<td></td>
<td>0.888–0.892</td>
<td>0.890</td>
</tr>
<tr>
<td></td>
<td>0.892–0.896</td>
<td>0.894</td>
</tr>
<tr>
<td></td>
<td>0.896–0.900</td>
<td>0.898</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set 3</th>
<th>Interval</th>
<th>Nominal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>2.200–2.204</td>
<td>2.202</td>
</tr>
<tr>
<td></td>
<td>2.204–2.208</td>
<td>2.206</td>
</tr>
<tr>
<td></td>
<td>2.208–2.212</td>
<td>2.210</td>
</tr>
<tr>
<td></td>
<td>2.212–2.216</td>
<td>2.214</td>
</tr>
<tr>
<td></td>
<td>2.216–2.220</td>
<td>2.218</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.840–0.844</td>
<td>0.842</td>
</tr>
<tr>
<td></td>
<td>0.844–0.848</td>
<td>0.846</td>
</tr>
<tr>
<td></td>
<td>0.848–0.852</td>
<td>0.850</td>
</tr>
<tr>
<td></td>
<td>0.852–0.856</td>
<td>0.854</td>
</tr>
<tr>
<td></td>
<td>0.856–0.860</td>
<td>0.858</td>
</tr>
</tbody>
</table>

We compute the 75 corresponding fitnesses. The best three models are kept in the buffer, i.e., $p_3 = 3$. The three best models are given in Table 10:

Table 10  The three best models after sub-run 3 of Algorithm 2

<table>
<thead>
<tr>
<th>Model sl. no.</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>Fitness $F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.230</td>
<td>0.878</td>
<td>0.8014</td>
<td>0.5000</td>
<td>0.9994</td>
<td>0.2273</td>
<td>0.1750</td>
<td>0.0600</td>
<td>0.0205</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2.234</td>
<td>0.886</td>
<td>0.7972</td>
<td>0.5031</td>
<td>1.0066</td>
<td>0.1794</td>
<td>0.6818</td>
<td>0.3500</td>
<td>0.6200</td>
<td>0.0600</td>
<td>0.0228</td>
</tr>
<tr>
<td>3</td>
<td>2.230</td>
<td>0.882</td>
<td>0.7993</td>
<td>0.5008</td>
<td>1.0001</td>
<td>0.2273</td>
<td>0.0875</td>
<td>0.1600</td>
<td>0.0100</td>
<td>0.0252</td>
<td></td>
</tr>
</tbody>
</table>

If we decide to stop at this point, the estimated model is the model with the best fitness, i.e., the first model.

The percentage errors in identification of $\alpha$, $\beta$, $a_1$, $a_2$, $a_3$ are respectively 0, 0.2273, 0.1750, 0, 0.0600.

The unit step input responses of the actual process and the best-estimated model are given below. This identification is illustrated in Figures 17(a) and (b).
7.5 Relative merits of Algorithm 2 over Algorithm 1

First and foremost, we can easily see from the percentage errors of estimated parameters that Algorithm 2 yields much better results. Secondly, for Algorithm 1, we considered $20 \times 20 = 400$ process models. For Algorithm 2, we needed to consider only 191 models. Thus Algorithm 2 is much more efficient. This happens because in Algorithm 2, we first take care to identify the possible sub-ranges of the fractional powers and then fine-search within probable ranges. Algorithm 1 employs only one level of searching, while Algorithm 2 is adaptive and employs intensive searching within specific intervals.

Algorithm 1 is of course simpler to implement, but that remains its only advantage. Algorithm 2 offers better results and is more efficient.
7.6 Possible improvements in Algorithm 2

The way sub-runs 1, 2 and 3 in Algorithm 2 were implemented was as follows. Suppose in a given sub-interval, $\alpha$ assumed the nominal values $\alpha_1, \alpha_2, \alpha_3$ and $\alpha_4$. In the same sub-interval, $\beta$ assumed the nominal values $\beta_1, \beta_2, \beta_3$ and $\beta_4$. We first fixed $\alpha$ at $\alpha_1$, varied $\beta$ from $\beta_1$ to $\beta_4$ and noted down the fitness values. Then we changed $\alpha$ to $\alpha_2$, varied $\beta$ from $\beta_1$ to $\beta_4$ and noted down the fitness values. In this way we continued until $\alpha = \alpha_4$.

A pattern was noticed in the corresponding fitness values. Either the fitness values decreased consistently from $\beta = \beta_1$ to $\beta = \beta_4$, or increased consistently, or first decreased then increased (Table 11). That is to say, a definite pattern was followed. This pattern was noticed in all the searches. Obviously, similar patterns in fitness values would have also been noted if we kept $\beta$ fixed at $\beta_1$ and varied $\alpha$ from $\alpha_1$ to $\alpha_4$ etc.

<table>
<thead>
<tr>
<th>$\alpha, \beta$</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>Fitness F</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.25, 0.81</td>
<td>0.8344</td>
<td>0.4967</td>
<td>0.9834</td>
<td>57.3075</td>
</tr>
<tr>
<td>2.25, 0.83</td>
<td>0.8246</td>
<td>0.4992</td>
<td>0.9878</td>
<td>37.1401</td>
</tr>
<tr>
<td>2.25, 0.85</td>
<td>0.8145</td>
<td>0.5021</td>
<td>0.9921</td>
<td>21.6413</td>
</tr>
<tr>
<td>2.25, 0.87</td>
<td>0.8041</td>
<td>0.5054</td>
<td>0.9961</td>
<td>10.6469</td>
</tr>
<tr>
<td>2.25, 0.89</td>
<td>0.7934</td>
<td>0.5092</td>
<td>0.9999</td>
<td>3.9810</td>
</tr>
<tr>
<td>2.19, 0.81</td>
<td>0.8333</td>
<td>0.4728</td>
<td>0.9887</td>
<td>4.0644</td>
</tr>
<tr>
<td>2.19, 0.83</td>
<td>0.8240</td>
<td>0.4754</td>
<td>0.9928</td>
<td>5.5932</td>
</tr>
<tr>
<td>2.19, 0.85</td>
<td>0.8144</td>
<td>0.4785</td>
<td>0.9968</td>
<td>9.6201</td>
</tr>
<tr>
<td>2.19, 0.87</td>
<td>0.8045</td>
<td>0.4820</td>
<td>1.0006</td>
<td>16.0432</td>
</tr>
<tr>
<td>2.19, 0.89</td>
<td>0.7942</td>
<td>0.4859</td>
<td>1.0042</td>
<td>24.7542</td>
</tr>
<tr>
<td>2.27, 0.91</td>
<td>0.7826</td>
<td>0.5208</td>
<td>1.0023</td>
<td>11.9583</td>
</tr>
<tr>
<td>2.27, 0.93</td>
<td>0.7710</td>
<td>0.5255</td>
<td>1.0059</td>
<td>6.3391</td>
</tr>
<tr>
<td>2.27, 0.95</td>
<td>0.7591</td>
<td>0.5306</td>
<td>1.0093</td>
<td>5.3533</td>
</tr>
<tr>
<td>2.27, 0.97</td>
<td>0.7467</td>
<td>0.5363</td>
<td>1.0125</td>
<td>8.7405</td>
</tr>
<tr>
<td>2.27, 0.99</td>
<td>0.7340</td>
<td>0.5424</td>
<td>1.0157</td>
<td>16.2263</td>
</tr>
</tbody>
</table>

Now, Algorithm 2 can be made even more adaptive and efficient by simply checking whether the fitness value gets better or worse with changing $\beta$ for fixed $\alpha$. If it gets better, then we should continue changing $\beta$ and noting the fitness values. If it is getting worse, we should move on to the next $\alpha$ value. In this fashion, the number of process models generated by Algorithm 2 can be reduced by a factor close to two.
8 Comments and conclusions

An elegant method for the identification of parameters of a fractional order system is proposed. For very accurate results, the value of $L$ (length of memory) should be large, and that of $T$ (the sampling time) small, while calculating the fractional derivation numerically.

The challenge in fractional order system identification is that the fractional powers are not restricted to assume only discrete integral values, but are distributed in a continuous interval. For two integral order systems, identical time domain responses mean identical transfer functions. But for fractional order systems, we often find that a better identification of the actual process has actually a lower fitness than a worse model.

We have made use of a five-parameter system only to illustrate our method but this method can be extended to the case of a system with any number of parameters, at least theoretically, but the increase in the number of parameters is naturally accompanied by an understandable increase in complexity. In fact, our method consists in estimating two distinct types of parameters: system coefficients and fractional powers. The technique of identification of system coefficients is independent of the number of coefficients. However, the difficulty in the identification of the fractional powers may increase significantly with the increase in the number of fractional powers. For such cases (number of fractional powers $\geq 3$), we can apply intelligent search algorithms (such as particle swarm optimisation or differential evolution) instead of the Algorithms 1 and 2 proposed by us. We are currently working on these lines and have already designed the relevant fitness function.

The method of finding a relation between the coefficients by use of fractional calculus renders the application of a complex evolutionary algorithm redundant. Therein lies its merit.

References


Complete identification of dynamic physical systems


