Fractional-Order Dynamics in Large Scale Control Systems

Bill Goodwine[1]

Abstract— **Fractional-order differential equations are increasingly being used to model systems in engineering for purposes such as control and health-monitoring. Because of the nature of a fractional derivative, mechanistically fractional order dynamics will most naturally arise when there are non-local features or processes in the dynamics. Even if there are no non-local effects, however, when searching for an approximate model for a very high order system, it is worth asking whether a fractional-order model is better than an integer-order model. This work is motivated by the challenges presented by very large scale systems, which will be increasingly common as integration of the control of formerly decoupled systems occurs such as in cyber-physical systems. Because fractional-order differential equations are more difficult to numerically compute, justifying the use of a fractional-order model is a balance between accuracy of the approximation and ease of computation. This paper constructs large, random networks and compares the accuracy of integer-order and fractional-order models for their dynamics. The main result is that, over the range of parameter values for the system considered, fractional-order models generally provide a more accurate approximation to the response of the system than integer order models. To ensure a fair comparison, both the fractional-order and integer-order models considered had two parameters.**

I. INTRODUCTION

In [1] the author constructed a random scale-free network of masses interconnected with springs and dampers and observed that, when compared to exponential solutions, the step response relationship between randomly selected nodes sometimes had characteristics suggestive of fractional-order systems. This paper extends those results by systematically applying optimization techniques to identify the degree to which integer-order or fractional-order models better approximate the step response of systems of a similar nature. To ensure a fair comparison, the optimization problem is over two decision variables in each case.

While the networks constructed in this paper are made up of springs and dampers, these are not necessarily only mechanical elements. Specifically, for a control system, if the distance between two agents is governed by a PD control law, the dynamic relationship between the masses will be equivalent to the mechanical components. As such, in this paper while we nominally consider a mechanical system, the results are motivated by, and apply to, a large class of control systems as well.

Fractional calculus has a long history in mathematics and a much shorter one in engineering. Of course the idea is that,

*The partial support of the US National Science Foundation under grant No. CMMI 1826079 is gratefully acknowledged.

[1]Bill Goodwine is with the Department of Aerospace & Mechanical Engineering at the University of Notre Dame, Notre Dame, IN, USA. bill@controls.ame.nd.edu

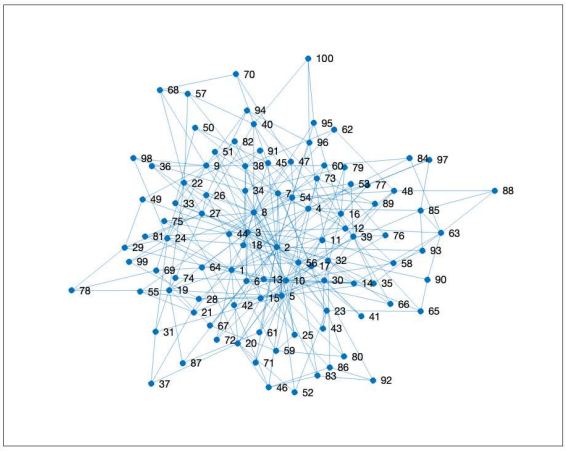

Fig. 1. Example network.

given a function $f(t)$ with integer-order derivatives

$$\frac{\mathrm{d}f}{\mathrm{d}t}(t), \quad \frac{\mathrm{d}^2 f}{\mathrm{d}t^2}(t), \quad \frac{\mathrm{d}^3 f}{\mathrm{d}t^3}(t), \quad \textit{etc.},$$

are there derivatives "in between" with real-valued orders, *e.g.*,

$$\frac{\mathrm{d}^{1/2} f}{\mathrm{d}t^{1/2}}(t), \quad \frac{\mathrm{d}^{1.2} f}{\mathrm{d}t^{1.2}}(t), \quad \textit{etc.}?$$

There are, in fact, many different definitions of the fractional derivative, many of which fundamentally are built upon replacing factorial functions appearing in many integer-order representations of the derivative with gamma functions. Examples include the Riemann–Liouville, Caputo and the Grünwald–Letnikov definitions and the interested reader is referred to the references [2]–[5], for descriptions of each.

In this paper we avoid having to choose a particular definition of the fractional derivative because we focus directly on the solution. The two-parameter Mittag-Leffler function is a generalization of the exponential function and is defined as [6]

$$\mathrm{E}_{\alpha,\beta}(t) = \sum_{k=0}^{\infty} \frac{t^k}{\Gamma(\alpha k + \beta)}, \qquad \alpha, \beta > 0$$

and it is well known that

$$x(t) = t^{\alpha} \mathrm{E}_{\alpha,\alpha+1}(at^{\alpha})$$

is a solution to

$$\frac{\mathrm{d}^{\alpha} x}{\mathrm{d}t^{\alpha}}(t) + ax(t) = 1, \quad x(0) = 0.$$

We will see subsequently that the step response relationship between randomly selected nodes in the network can look like this function, thus motivating searching for the best α and a values to approximate the step response.

The literature on fractional calculus is vast. The interested reader is referred to several books overviewing the topic from a mathematical perspective, such as [7]–[9]. Papers using fractional calculus to model large-scale and infinite order dynamics include, among others, [10]–[12]. Viscoelasticity is an obvious application, and some references include [13], [14], and some prior work by the authors using it in robotics include [15]–[17]. Fractional-order control is one of the more popular areas and two recommended references include [18], [19]. Finally, an excellent review article illustrating the extremely broad range of applications is [20].

The literature on integrated, large-scale and cyber-physical systems is similarly extensive. Formation control and multi-agent control of cyber-physical systems, of course, has a vast literature and representative references include [21]–[37] An important result from this work is that fractional-order dynamics seem to be commonplace in large-scale systems, and as integrated systems become larger and larger, such models may become increasingly useful.

II. NETWORK MODEL AND NUMERICAL EXPERIMENTS

In this paper we consider many networks of masses interconnected by springs and dampers and study their dynamic response. We selected mass-spring-damper networks because they are prototypical and often can be used for models of swarms of robots, complicated mechanical structures, etc. Specifically, they can model the proportional and derivative terms in a PD control law controlling spacing between agents in a formation control problem.

To construct a single network, we randomly[1] construct a networks with a size between 50 and 2050 masses. Each mass is connected to some of the other masses with dampers or springs. Which other masses they are connected to is randomly selected with a bias toward connecting to masses already connected to a large number of other masses. Also whether the connecting element is a spring or damper is randomly selected.

In detail, a single instance of the network was constructed as follows.

1) Set the spring constant value, k.
2) Randomly select the number of masses, N (between 50 and 2050).
3) Randomly select the damping coefficient, b (between 2 and 5).
4) Randomly select the minimum connectivity, *mincon*, for the masses in the system (between 1 and 3).
5) Create a network with *mincon* masses, each of which is connected to each of the other masses with both a spring and damper.
6) Add $N - mincon$ more masses one at a time by connecting it to *mincon* other masses as follows:
   a) Randomly select an existing mass as a potential target mass to connect.
   b) Compute a threshold value equal to a random number between 0 and 1 times the number of currently existing masses. If the potential target mass is connected to more masses than the threshold, then connect to it.
   c) If the threshold connectivity is not met, then randomly select another potential target and repeat.
   d) The connecting element is randomly selected as either a spring or damper.

An example network with $N = 100$ and $mincon = 3$ is illustrated in Figure 1.

The dynamics of each node are such that each mass i has a position, $x_i(t)$ and velocity $\dot{x}_i(t)$, and its dynamics are given by Newton's law. For simplicity in this paper, we constrain the dynamics to be one-dimensional, so that the equation of motion for mass i is

$$\ddot{x}_i(t) = \sum_{m \in \mathscr{K}_i} k\left(x_m(t) - x_i(t)\right) + \sum_{m \in \mathscr{B}_i} b\left(\dot{x}_m(t) - \dot{x}_i(t)\right)$$

where $\mathscr{K}_i$ is the set of neighbors of mass i connected to it with springs and $\mathscr{B}_i$ is the set of neighbors of mass i connected to it with dampers and the value of the mass is 1. The network is illustrated in the plane in Figure 1 for clarity of presentation, but in fact the network dynamics are actually one dimensional.

This results in a network that is approximately scale free because a few of the masses are very highly connected to many other masses; whereas, many of the masses are only connected to a few other masses. In other words, the degree distribution for the network is approximately described by a power law: if $P(k)$ is the fraction of masses connected to k other masses, then $P(k) \sim k^{-\gamma}$. We choose an approximately scale free network because they are relatively common in nature with some desirable attributes [38]. Whether there is a specific connection to that feature of the network, as opposed to simply being large-scale, is the subject of future research efforts. The fact that the network illustrated in Figure 1 is approximately scale free is illustrated in Figure 2.

Having constructed a network, a single experiment consisted of:

1) Create the network according to the recipe previously described.
2) Randomly select the input node. If the input node is only connected to the rest of the network by dampers and no springs, randomly select a different one.
3) Randomly select an output node that is different from the input node.
4) Set all the initial conditions equal to zero, except for the input node, i, where we set $x_i(0) = 1$ and $\dot{x}_i(0) = 0$. Keep the input node fixed throughout the experiment,

[1] All random selections in this paper are made using the Matlab `rand()` function, meaning that the random variable is selected from a probability distribution that is an approximately uniform distribution between the minimum and maximum values.

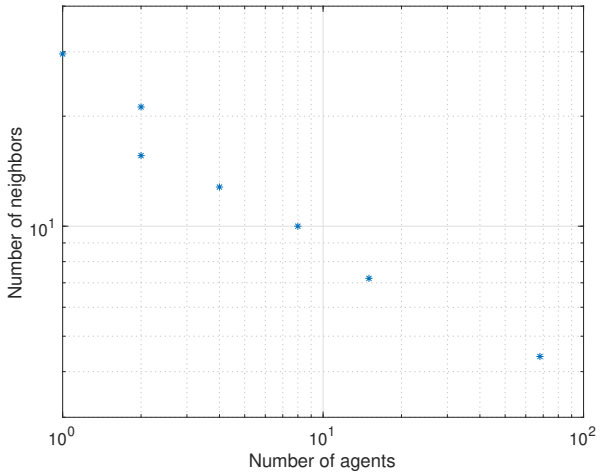

Fig. 2. Number of agents versus degree of connectivity, illustrating that the network is approximately scale free.

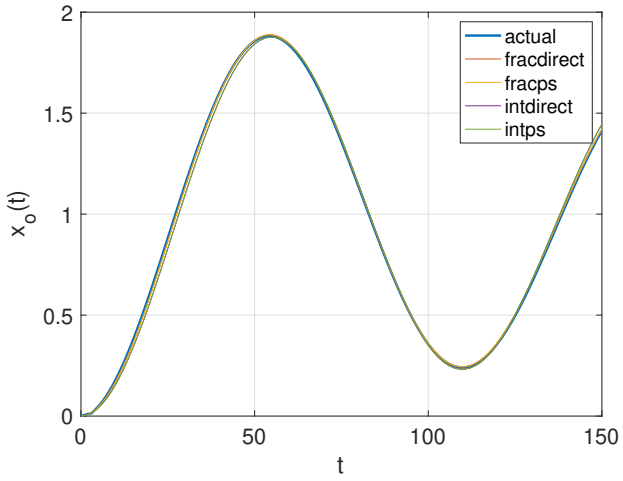

Fig. 3. Step response of selected output mass.

i.e., $x_i(t) = 1$ and $\dot{x}_i(t) = 0$. The fact the input node must be connected to the rest of the network with at least one spring is because with this method of creating the input, there is no force applied to the rest of the network by the input without any springs.

5) Numerically solve the system of $2N$ differential differential equations. We used `ode45()` in Matlab with `RelTol` = 1e-5 and `AbsTol` = 1e-8 for all cases in this paper.
6) Use the optimization methods described in the next section to determine the best fits to the solution of the output node, $x_o(t)$, for a fractional-order model and for an integer order model.

A typical response is illustrated in Figure 3 for a network with $N = 1235$ masses, $k = 5$, $b = 3.957$, input node, $i = 628$ and output node, $o = 1012$. The range of the parameter values described above (for N, b, t, *etc.*) were selected so that the reciprocal of the natural frequency was approximately one half of the time range considered. As described in the next section, we then used two different optimization methods to determine the best fit to this solution for an second-order (integer) model and a fractional-order model.

III. OPTIMIZATION METHODS

For each experiment, we attempt to match solutions to both the integer-order system, $x_n(t)$

$$\frac{\mathrm{d}^2 x_n}{\mathrm{d}t^2}(t) + 2\zeta\omega_n \frac{\mathrm{d}x_n}{\mathrm{d}t}(t) + \omega_n^2 x_n(t) = 1$$

and the fractional-order system, $x_f(t)$

$$\frac{\mathrm{d}^\alpha x_f}{\mathrm{d}t^\alpha}(t) + a x_f(t) = 1$$

with zero initial conditions to the computed $x_o(t)$. Each system has two parameters, (ζ, ω_n) for the integer-order system and (α, a) for the fractional order system. Both of these have closed form solutions, which were used in the optimization, specifically

$$x_n(t) = 1 - e^{-\zeta\omega_n t}\left(\cos\left(\omega_n\sqrt{1-\zeta^2}t\right) + \zeta/\sqrt{1-\zeta^2}\sin\left(\omega_n\sqrt{1-\zeta^2}t\right)\right)$$

if $\zeta < 1$ and

$$x_n(t) - 1 - \frac{p_2}{p_2 - p_1}e^{p_1 t} + \frac{p_1}{p_2 - p_1}e^{p_2 t}$$

when $\zeta \geq 1$ and p_1 and p_2 are the two real roots of $p^2 + 2\zeta\omega_n p + \omega_n^2$. For the fractional equation, the solution is given by the Mittag-Leffler function

$$x_f(t) = t^\alpha E_{\alpha,\alpha+1}(at^\alpha)$$

as described previously.

The solution for $x_o(t)$ is sampled at 100 points evenly spaced in time, and the integer-order, $x_n(t)$, and fractional-order, $x_f(t)$, solutions are evaluated at those same times. The error in each case is the sum of the squares of the difference between x_o and the two models at the 100 points:

$$error_n = \sum_{k=1}^{100}\left(x_o(k\Delta t) - x_n(k\Delta t)\right)^2$$

$$error_f = \sum_{k=1}^{100}\left(x_o(k\Delta t) - x_f(k\Delta t)\right)^2,$$

where $\Delta t = 1.5152$.[2]

In order to ensure that our results are not an artifact of any specific optimization method, we used two different global nonlinear optimization methods to search for the best matches for the fractional- and integer-order models. Both methods balance local refinement with global searching by maintaining a set of candidate solutions. The particle swam

[2]Note that this Δt is NOT the time step used to compute the numerical solution to the differential equations. That time step is adaptive and handled automatically by `ode45()`.

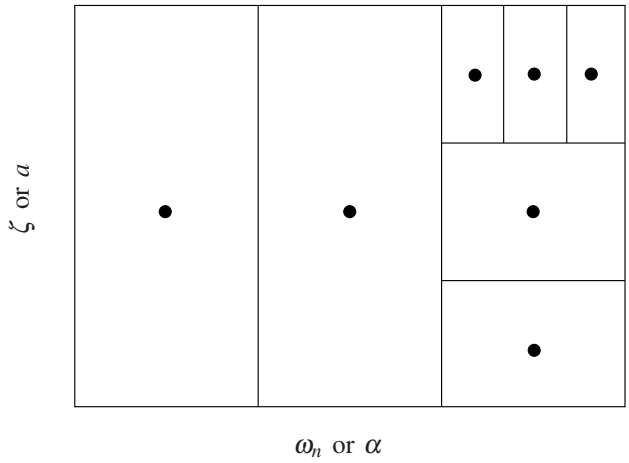

Fig. 4. Dividing rectangles division of the search space.

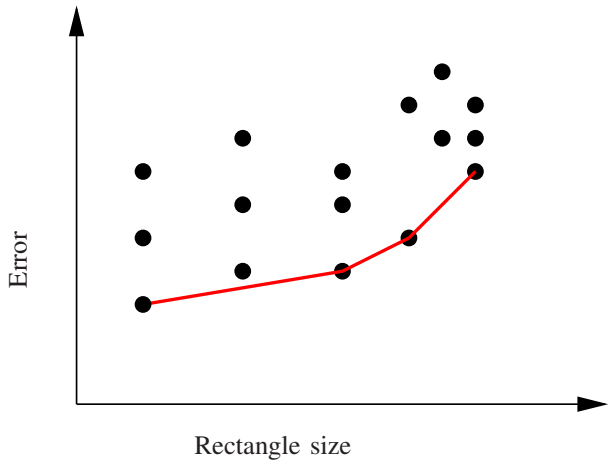

Fig. 5. Dividing rectangles method to select which rectangles to subdivide.

method is easily implemented using a standard Matlab toolbox and contains stochastic elements to the search; whereas, dividing rectangles is a fully deterministic search method.

A. Particle Swam Method

We implement the standard Matlab particle swarm optimization method with all parameter values set to the default except the swarm size is set to 100. The method works by generating initial values and velocities to each candidate solution ("particle"). The error is computed for each particle and the velocity for each particle is updated based upon a combination of its previous velocity, the best solution it has seen and the best solution a randomly selected set of neighbors has seen. The position of each particle is updated according to its velocity and the new errors are computed. There are a variety of criteria to terminate the algorithm including reaching a fixed number of iterations, the relative change in the objective function over a set number of iterations is no longer improving, *etc.*[3]

B. Dividing Rectangles

In contrast to the particle swarm method which includes random elements in the velocity computation for each particle, the dividing rectangles method is completely deterministic (see [39] for a comprehensive description). In the method, the search space is divided into rectangles and the error is computed at the center of each rectangle. At each step, some of the rectangles are subdivided, and the error is computed at the center of the smaller rectangles. The balance between locally refining good solutions and maintaining a global search is struck by plotting the error at the center of each existing rectangle versus the size of each rectangle. The rectangles that are subdivided are those along the lower convex hull of this plot.

As a schematic example, Figure 4 shows the search space divided into three different sized rectangles. The error is computed at the center of each rectangle. A plot of the error versus rectangles size for six different sized rectangles is illustrated in Figure 5. The lower convex hull is illustrated by the red line. The rectangles corresponding to the points along this convex hull are selected to be divided in the next step. This provides a balance between refining the best solution (on the lower left of the convex hull) with larger rectangles containing a lot of unsearched space (upper right of convex hull). This method continues for a fixed number of steps.

[3]https://www.mathworks.com/help/gads/particle-swarm-optimization-algorithm.html

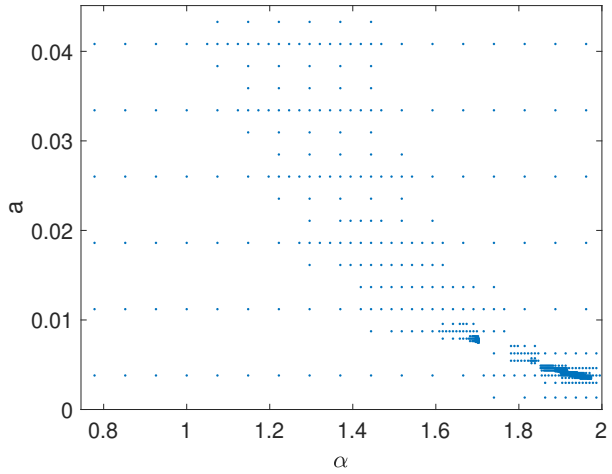

Fig. 6. Dividing rectangles process: each blue dot is the center of a rectangle.

We used the algorithm implementation from [40]. For a typical run, the process results searching for optimal values as illustrated in Figure 6. Each blue dot is the center of a rectangle. The search focusing on potential optimal values is shown by the more densely spaced dots. In all the experiments, for identifying the fractional-order parameters the dividing rectangles was run for 200 iterations. For the integer-order models it was run for 1000 iterations.

IV. RESULTS

For each of the spring constant values of $k = 0.5, 2, 5, 6, 10$ and 25 we constructed 50 networks using the method de-

direct or swarm	k	fractional best (count)	integer best (count)	fractional best (percent)
direct	0.5	9	33	21
swarm	0.5	16	26	38
direct	2.0	18	25	42
swarm	2.0	21	22	49
direct	5.0	26	16	62
swarm	5.0	31	11	74
direct	6.0	23	20	53
swarm	6.0	30	13	70
direct	10	27	12	69
swarm	10	29	10	74
direct	25	35	8	81
swarm	25	35	8	81

TABLE I
SUMMARY OF RESULTS.

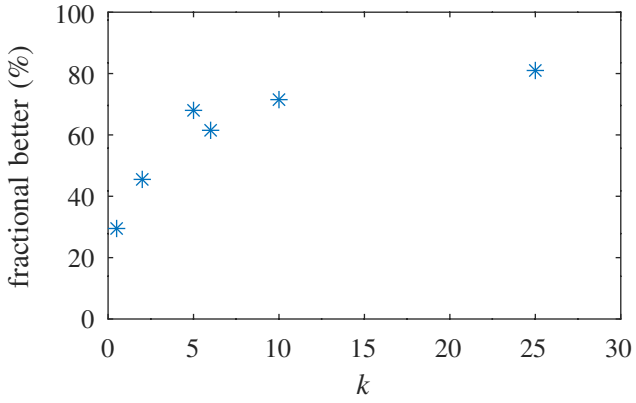

Fig. 7. Percentage of cases where the fractional-order model provides a better fit as a function of k.

scribed in Section II. Each of the 50 networks had a random number of masses, a random damper constant b and a random number of minimum connections. For each network, a random input mass was selected and different random output mass. The initial conditions for the input mass were set to $x(0) = 1$ and $\dot{x}(0) = 0$, and the mass was held fixed in that position. The $2N$ differential equations describing the motion of the masses was solved for $t = 0$ to $t = 150$, which was selected to give a couple of periods of oscillation for the typical motions, or if there were no oscillations, for the characteristic rise time of the system to be about half of that time range. If the input mass that was selected happened to be connected with the rest of the network with only dampers, then no motion results, so in that case a different input mass was randomly selected.

For each of the 50 networks, both the particle swarm and the direct optimization methods were run to attempt to identify the best pairs (ω_n, ζ) and (α, a) for the integer-order and fractional-order cases, respectively. If any of the identified parameter values was on the boundary of the search region, the result was thrown out. Table I summarizes the results. The last column shows the proportion of times the fractional-order models provided a better model for each of the various k values.

One conclusion that is evident from the table is the fact that the proportion of cases where the fractional-order model provides a better fit increases with k, as illustrated in Figure 7. Why this is the case is not yet clear. A typical response for $k = 0.5$ is illustrated in Figure 8 and for $k = 25$ is illustrated in Figure 9, so the fractional cases seems to generally be better when the response is more oscillatory. Over the range of parameter values studied, the fractional-order model provided a more accurate model for the step response than the integer-order model approximately 60% of the time. Therefore, the attributes of the types of cases for more general systems where a more accurate description by using a fractional-order model rather than integer-order is warranted.

Because the number of states was randomly set over a fairly wide range, the time needed for the simulation and

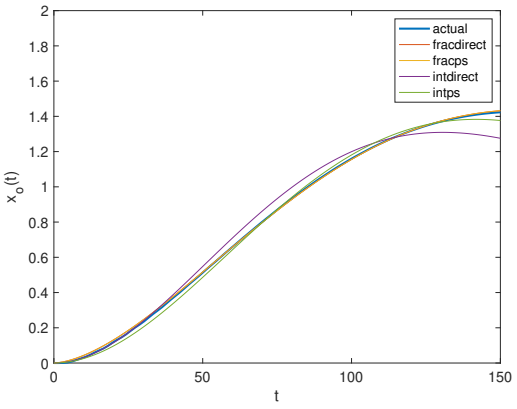

Fig. 8. Typical response for case when $k = 0.5$. Other parameter values for this simulation were $N = 1331$, *mincon* $= 2$ and $b = 2.4$.

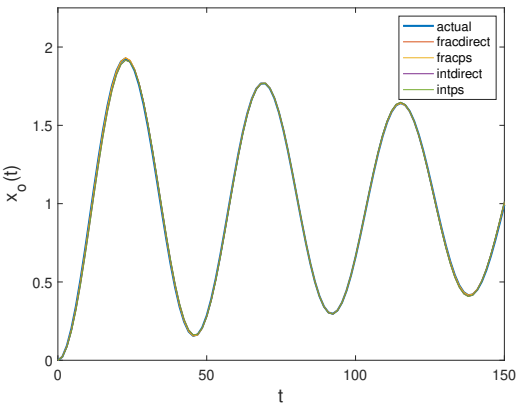

Fig. 9. Typical response for case when $k = 25$. Other parameter values for this simulation were $N = 1157$, *mincon* $= 2$ and $b = 2.5$.

optimizations varied quite a bit. Typically, it would take approximately two minutes of wall clock time to construct the network, solve the $2N$ differential equations, and identify the optimal fractional-order parameters and integer-order parameters using the particle swarm and dividing rectangles methods. The platform was a 2020 Apple M1 (8 GHz with 8 cores) and with 8GB RAM running Matlab R2022b. Therefore, the entirety of the simulations reported in Table I required approximately 9 hours of wall clock time to complete.

V. CONCLUSIONS

This paper showed that fractional-order dynamics are common in dynamic responses of the large scale systems studied in this paper. The system was a network of masses connected by springs and dampers, and the very high-order response $N \sim 2000$ appeared to be "normal" by cursory visual inspection as a second-order step response, but was often better matched with the solution to a fractional-order system. The comparison was fair in the sense that both the integer-order system and fractional-order system had two variables over which the solutions were optimized.

In this paper the obviously conclusion, other than the utility of fractional-order models, was that the fractional-models were better for larger k values. Current research efforts are directed towards a mechanistic explanation for this as well as identifying other correlations between system parameter values and fractional-order dynamics such as, distance in the graph between the input and output nodes, network size, connectivity as well as whether the scale free nature of the network has any bearing on the existence of fractional-order dynamics at all.

REFERENCES

[1] B. Goodwine, "Fractional-order dynamics in a random, approximately scale-free network of agents," in *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, 2014, pp. 1581–1586.

[2] M. Ortigueira, "An introduction to the fractional continuous-time linear systems: the 21st century systems," *Circuits and Systems Magazine, IEEE*, vol. 8, no. 3, pp. 19–26, 2008.

[3] M. D. Ortigueira, *Fractional Calculus for Scientists and Engineers*, ser. Lecture Notes in Electrical Engineering. Springer, 2011, vol. 84.

[4] J. T. Machado, V. Kiryakova, and F. Mainardi, "Recent history of fractional calculus," *Communications in Nonlinear Science and Numerical Simulation*, vol. 16, no. 3, pp. 1140 – 1153, 2011.

[5] S. Das, *Functional Fractional Calculus*. Springer Science & Business Media, 2011.

[6] R. Gorenflo, *Mittag-Leffler functions, related topics and applications*, ser. Springer monographs in mathematics. Heidelberg [Germany]: Springer, 2014.

[7] M. D. Ortigueira, *Fractional Calculus for Scientists and Engineers*, ser. Lecture Notes in Electrical Engineering. Netherlands: Springer Netherlands, 2011, vol. 84.

[8] K. Oldham and J. Spanier, *The Fractional Calculus Theory and Applications of Differentiation and Integration to Arbitrary Order*. Elsevier Science, 1974.

[9] A. Oustaloup, *La d´erivation non enti'ere*. Hermes, 1995.

[10] A.-J. Guel-Cortez, C.-F. Méndez-Barrios, E.-j. Kim, and M. Sen, "Fractional-order controllers for irrational systems," *IET Control Theory & Applications*, vol. 15, no. 7, pp. 965–977, 2021. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/cth2.12095

[11] J. Sabatier, "Beyond the particular case of circuits with geometrically distributed components for approximation of fractional order models: Application to a new class of model for power law type long memory behaviour modelling," *Journal of Advanced Research*, vol. 25, pp. 243–255, 2020, recent Advances in the Fractional-Order Circuits and Systems: Theory, Design and Applications. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2090123220300667

[12] J. Mayes, "Reduction and approximation in large and infinite potential-driven flow networks," Ph.D. dissertation, University of Notre Dame, 2012.

[13] T. C. Doehring, A. D. Freed, E. O. Carew, and I. Vesely, "Fractional Order Viscoelasticity of the Aortic Valve Cusp: An Alternative to Quasilinear Viscoelasticity," *Journal of Biomechanical Engineering*, vol. 127, no. 4, pp. 700–708, 01 2005. [Online]. Available: https://doi.org/10.1115/1.1933900

[14] N. Heymans and J.-C. Bauwens, "Fractal rheological models and fractional differential equations for viscoelastic behavior," *Rheologica Acta*, vol. 33, no. 3, pp. 210–219, 1994. [Online]. Available: https://doi.org/10.1007/BF00437306

[15] B. Goodwine, "Modeling a multi-robot system with fractional-order differential equations," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1763–1768.

[16] K. Leyden and B. Goodwine, "Using fractional-order differential equations for health monitoring of a system of cooperating robots," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 366–371.

[17] K. Leyden, M. Sen, and B. Goodwine, "Large and infinite mass–spring–damper networks," *Journal of Dynamic Systems, Measurement, and Control*, vol. 141, no. 6, Feb 2019, 061005. [Online]. Available: https://doi.org/10.1115/1.4042466

[18] D. Valério and J. S. de Costa, *An Introduction to Fractional Control*. London, United Kingdom: Institution of Engineering and Technology, 2013.

[19] Dingyu Xue, Chunna Zhao, and YangQuan Chen, "Fractional order pid control of a dc-motor with elastic shaft: a case study," in *2006 American Control Conference*, 2006, pp. 3182–3187.

[20] H. Sun, Y. Zhang, D. Baleanu, W. Chen, and Y. Chen, "A new collection of real world applications of fractional calculus in science and engineering," *Communications in Nonlinear Science and Numerical Simulation*, vol. 64, pp. 213–231, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1007570418301308

[21] J. Sztipanovits, X. Koutsoukos, G. Karsai, N. Kottenstette, P. Antsaklis, V. Gupta, B. Goodwine, J. Baras, and S. Wang, "Toward a science of cyber-physical system integration," *Proceedings of the IEEE*, 2011.

[22] J. Julliand, H. Mountassir, and E. Oudot, "Composability, compatibility, compositionality: automatic preservation of timed properties during incremental development," UFR Sciences et Techniques, Tech. Rep., 2007.

[23] B. Goodwine and P. Antsaklis, "Multi-agent compositional stability exploiting system symmetries," *Automatica*, vol. 49, no. 11, pp. 3158–3166, 2013.

[24] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[25] W. Ren, R. W. Beard, and E. M. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control Systems Magazine*, pp. 71–82, April 2007.

[26] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.

[27] N. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials, and coordinated control of groups," in *Proceedings of the 40th IEEE Conference on Decision and Control*, December 2001, pp. 2968–2973.

[28] R. Olfati-Saber and R. M. Murray, "Distributed cooperative control of multiple vehicle formations using structural potential functions," in *Proceedings of the 2002 IFAC World Congress*, July 2002.

[29] M. B. McMickell and B. Goodwine, "Reduction and nonlinear controllability of symmetric distributed systems," *International Journal of Control*, vol. 76, no. 18, pp. 1809–1822, 2003.

[30] I. Suzuki and M. Yamashita, "Distributed anonymous mobile robots: Formation of geometric patterns," *SIAM Journal on Computing*, vol. 28, no. 4, pp. 1347–1363, 1999.

[31] M. B. McMickell and B. Goodwine, "Reduction and controllability of robotic systems with drift," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, May 2002.

[32] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1465– 1476, 2004.
[33] M. B. McMickell and B. Goodwine, "Motion planning for symmetric distributed robotic systems," in *2003 IEEE International Conference on Robotics and Automation*, September 2003.
[34] Y. Ikemoto, Y. Hasegawa, T. Fukuda, and K. Matsuda, "Gradual spatial pattern formation of homogeneous robot group," *Information Sciences*, vol. 171, no. 4, pp. 431–445, 2005.
[35] M. B. McMickell and B. Goodwine, "Motion planning for nonlinear symmetric distributeed robotic systems," *International Journal of Robotics Research*, vol. 26, no. 10, pp. 1025–1041, October 2007.
[36] M. B. McMickell, B. Goodwine, and L. A. Montestruque, "Micabot: A robotic platform for large-scale distributed robotics," in *2003 IEEE International Conference on Robotics and Automation*, September 2003.
[37] M. B. McMickell and B. Goodwine, "Reduction and controllability of symmetric distributed systems with robotic applications," in *International Conference on Intelligent Robotics and Systems*, vol. 3. IEEE/RSJ, October 2001, pp. 1232–1236.
[38] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999. [Online]. Available: https://science.sciencemag.org/content/286/5439/509
[39] D. R. Jones and J. R. R. A. Martins, "The DIRECT algorithm: 25 years later," *Journal of Global Optimization*, vol. 79, p. 521–566, 2021.
[40] M. Björkman and K. Holmström, "Global optimization using the direct algorithm in matlab," *Advanced Modeling and Optimization*, vol. 1, no. 2, pp. 17–37, 1999.