fractal and fractional

MDPI

Article

A Fractional-Order On-Line Self Optimizing Control Framework and a Benchmark Control System Accelerated Using Fractional-Order Stochasticity †

Jairo Viola and YangQuan Chen *

School of Engineering, University of California, Merced, 5200 North Lake Road, Merced, CA 95340, USA
* Correspondence: ychen53@ucmerced.edu
† This manuscript expand the results presented on the paper J. Viola and Y. Chen, "An Accelerated Self Optimizing Control Framework for Smart Process Control Using Fractional Order Stochasticity", 2021 9th International Conference on Control, Mechatronics and Automation (ICCMA), 11–14 November 2021; pp. 104–109, doi:10.1109/ICCMA54375.2021.9646222.

Abstract: This paper presents a design and evaluation of a fractional-order self optimizing control (FOSOC) architecture for process control. It is based on a real-time derivative-free optimization layer that adjusts the parameters of a discrete-time fractional-order proportional integral (FOPI) controller according to an economic cost function. A simulation benchmark is designed to assess the performance of the FOSOC controller based on a first order plus dead time system. Similarly, an acceleration mechanism is proposed for the fractional-order self optimizing control framework employing fractional-order Gaussian noise with long-range dependence given by the Hurst exponent. The obtained results show that the FOSOC controller can improve the system closed-loop response under different operating conditions and reduce the convergence time of the real-time derivative-free optimization algorithm by using fractional-order stochasticity.

Keywords: self optimizing control; discrete FOPI controller; globalized constrained Nelder–Mead algorithm; fractional-order stochasticity

Citation: Viola, J.; Chen, Y. A Fractional-Order On-Line Self Optimizing Control Framework and a Benchmark Control System Accelerated Using Fractional-Order Stochasticity. *Fractal Fract.* **2022**, *6*, 549. https://doi.org/10.3390/fractalfract6100549

Academic Editor: Martin Čech

Received: 19 August 2022
Accepted: 19 September 2022
Published: 28 September 2022

1. Introduction

Industry 4.0 requires smart controls to ensure closed-loop stability and improve system performance based on its real-time prognostic analytics. Thus, control strategies such as self optimizing control (SOC) can be used to design smart control systems. The SOC has different approaches. One of them is the control variable optimization as proposed by [1–4]. In this type of SOC, the system complexity is high due to hundreds of inputs and outputs on the system that need to be optimized, where the SOC controller performs offline optimization to determine the most suitable control variables.

Similarly, the SOC can also be extended for controller closed-loop parameter adjustment on systems with limited knowledge of its dynamics. It includes controllers such as Extremum seeking, maximum power point tracking (MPPT), run to run, iterative learning, or real-time optimization algorithms [5–7]. In these cases, the performance is assessed in an online configuration to find the controller parameters that satisfy operation specifications given a set of constraints.

In addition, the extremum seeking control (ESC) methods rely on a model-free approach to determine the optimal controller parameters [5,8]. Some ESC methods have fractional-order versions [9–11] employed on applications such as impedance matching, gas sensing, or maximum power tracking of photovoltaic arrays. Thus, the system performance is improved based on an economical cost function and using fractional-order derivatives or stochasticity.

Regarding real-time optimization algorithms, some work uses optimization to find the optimal controller parameters based on an economical cost function. This includes solving classic quadratic regulator problems or model predictive control methods [12–15], which requires a well-defined model of the system in terms of its governing equations or estimations based on neural networks [7].

In the particular case of fractional-order PI (FOPI) or PID (FOPID) controllers, there are adaptive control approaches that perform optimization using metaheuristic algorithms and fuzzy logic to determine the optimal gains for the controller depending on the operating zone of the system [16–25]. However, these optimization methods are executed offline due to the high computational load required for these methods to converge into an optimal solution. It makes this type of optimization impractical from a real-time optimization point of view, especially if the degrees of freedom (DOF) of the controller are of higher order ($DOF > 5$). In addition, the optimization methods noted above require significant time and computational resources to provide an optimal solution that satisfies the desired performance specifications. Therefore, accelerating the convergence of the optimization algorithms to the optimal solution is fundamental to self optimizing control to ensure faster response from the initial tuning and after an external disturbance or parametric uncertainty exists in the system.

In that sense, fractional-order calculus has shown the capability of improving optimization processes by using more optimal randomness such as Lévy flights, fractional Gaussian noise, or alpha-stable distributions, which are heavy tail distributions with long-range dependence (LRD) properties. Applications of more optimal optimization using fractional order randomness includes improved extremum seeking control and maximum power point tracking [9,26], metaheuristic optimization [27,28], accelerated gradient descend [29] stochastic configuration networks [30], or more optimal consensus [31].

This paper presents a new fractional-order self optimizing control (FOSOC) framework based on derivative-free optimization algorithms for the performance improvement of a stable closed-loop fractional-order system with online tuning of the system controller parameters according to a performance cost function. The gradient-free globalized constrained Nelder–Mead (GCNM) algorithm is proposed as an optimization method for the FOSOC controller. In addition, an acceleration method for the GCNM is developed employing fractional-order stochasticity. It uses the fractional Gaussian noise with long-range dependence (LRD) given by the Hurst exponent to perform the probabilistic restart on the GCNM optimization algorithm.

A simulation benchmark is designed for the performance assessment of the FOSOC controller using a first order plus dead time (FOPDT) system with a fractional-order proportional integral (FOPI) controller. A discrete-time implementation of the FOPI controller is employed to enable real-time optimization of the approximation proposed by [32]. The initial conditions for the FOPI controller parameters are calculated using the FMIGO method proposed by [33]. The FOSOC controller is evaluated on three scenarios for the FOPDT—a time constant dominated, balanced, and delay dominated system—for a periodic reference signal with a set of performance indices to obtain a quantitative performance comparison of the strategy. The accelerated GCNM for the FOPI controller is evaluated under the same scenarios to determine the LRD that increases the GCNM convergence speed. This paper is the extended application of the SOC control framework presented in [34,35] for fractional-order systems with accelerated convergence produced by fractional-order stochasticity with LRD. The main contributions of this paper are:

- Introducing a fractional-order self optimization control framework using derivative-free optimization algorithms to improve the closed-loop performance of a system with a FOPI controller;
- The development of a more optimal fractional-order self optimizing controller using fractional-order stochasticity with enhanced convergence properties towards the performance improvement of the stable closed-loop system performance employing a FOPI controller.

This paper is structured as follows. Section 2 introduces the parallel self-optimizing control framework. Section 3 defines the GCNM optimization algorithm. Section 4 presents the discrete implementation of the fractional order PI controller using the practical approximation presented in [36]. Section 5 describes the SOC benchmark designed in Matlab/Simulink to assess the SOC FOPI controller. Section 6 describes the FOSOC acceleration method using fractional-order stochasticity. Finally, conclusions and future work are presented.

2. Fractional-Order Self Optimizing Control Framework

The proposed FOSOC control architecture is shown in Figure 1. As can be observed, the SOC acts as a high optimization layer taking the system reference r, error signal e, and output y to find the optimal values of the controller $c(s)$, corresponding to a FOPI controller (1) with proportional gain k_p, integral gain k_i, and fractional-order integration order λ. A FOPDT system is selected as test system $p(s)$ (2), where K, τ, and L are the system gain, time constant, and delay, respectively. The FOSOC cost function is given by (3) and (4), where T_s is the system settling time, OV is the overshoot percentage, $\mu = [k_p, k_i, \lambda]$, A and B are the maximum overshoot and settling time, $k_{p_{min,max}}, k_{i_{min,max}}, \lambda_{min,max}$ are the limits for the FOPI gains, and $W_{1,2,3}$ are the weights for the overshoot, settling time, and the integral square error index, respectively. In this case, the FOSOC controller executes an optimization step after one cycle of the periodic reference signal r.

$$c(s) = k_p + \frac{k_i}{s^\lambda} \tag{1}$$

$$p(s) = \frac{K}{\tau s+1} e^{-Ls} \tag{2}$$

$$\min_{\mu \in \mathbb{R}} J = W_1\, OV(\mu) + W_2\, T_s(\mu) + W_3 \int_0^t e^2(t,\mu)\, dt, \tag{3}$$

subject to:

$$OV(\mu) < A;\ \ T_s(\mu) < B,$$
$$k_{p_{min}} \leq k_p \leq k_{p_{max}},\ \ k_{i_{min}} \leq k_i \leq k_{i_{max}},\ \ \lambda_{min} \leq \lambda \leq \lambda_{max}. \tag{4}$$

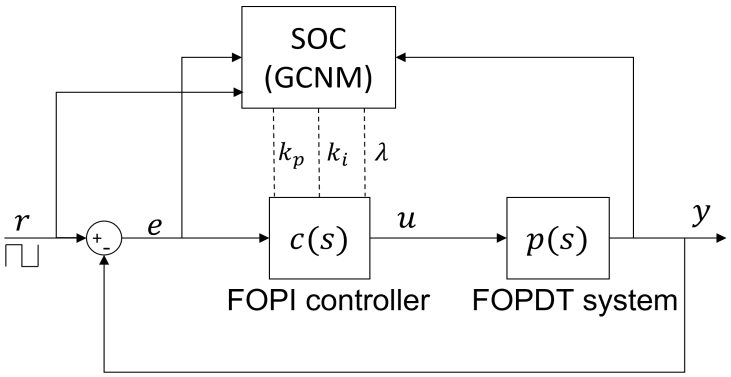

Figure 1. Proposed self optimizing control architecture with FOPI controller.

3. Globalized Constrained Nelder–Mead Optimization Algorithm

The globalized constrained Nelder–Mead algorithm (GCNM) shown in Figure 2 is employed as optimization algorithm for the FOSOC control. It is a modification of the classic Nelder–Mead (NM) [37], which searches for a global optimum based on a set of constraints. The GCNM method follows the same steps of evaluation α, reflection β, contraction γ, expansion, and shirking δ of the NM to create the simplex shape of $n+1$ vertices, where n is the number of optimization outputs.

The GCNM algorithm introduces a probabilistic restart mechanism that reinitializes the search from a different random initial condition to prevent falling into a local minimum.

Initially, the probabilistic restart evaluates if the cost function (3) reaches a steady value. For this, the algorithm evaluates if the standard deviation of the last m values of the simplex centroid is below a threshold ϵ. If this is true, then the optimization is in the steady state, and the constraints are evaluated. If at least one of the constraints is not satisfied, the GCNM restarts the searching on a new random point, assigning a new set of initial conditions among the parameter space defined for the problem. In this case, the new random initial points for the GCNM optimization method are selected using a normal distribution. Notice that the GCNM probabilistic restart can use different stochasticity, as will be shown in Section 6, where it is replaced by the fractional order Gaussian noise with LRD.

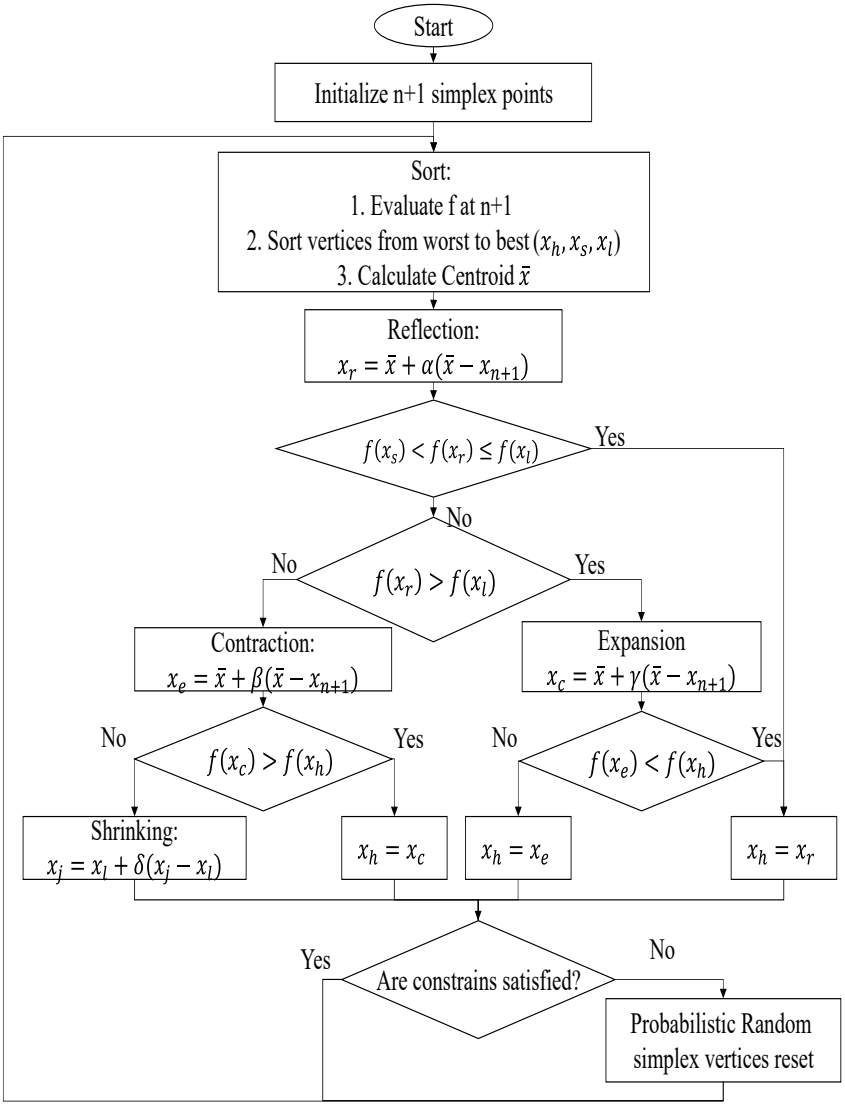

Figure 2. Globalized constrained Nelder–Mead algorithm flowchart.

One of the main advantages of the GCNM algorithm is that it can be used for any system with little or no knowledge of its dynamical behavior. In addition, considering the sequential structure of the GCNM algorithm and the low-computational complexity of the algorithm, it is adapted for real-time execution, conditioning the computation of each of the operations to a period of the reference signal r. Other advantages of the GCNM optimization algorithm include:

- It is a derivative-free model optimization method, which can be employed for any system with or without prior knowledge of the system model.

- The GCNM optimization is immune to noise (no gradient estimation), enhancing the robustness of the optimization process.
- By using probabilistic restart, the GCNM optimization algorithm is less susceptible to falling in a local minimum when searching for an optimal solution.
- The cost function is evaluated online directly using the real process, unlike several optimization approaches that perform offline optimization before real testing.
- The GCNM can be implemented in any embedded device with low computational cost. Please see its real-time implementation for solving the sphere function problem in Arduino https://tinyurl.com/5h3j5r5c (accessed on 10 July 2022).
- In the case of the SOC framework presented in this paper, the noise robustness, as well as the global search behavior, allows a wider exploration of the optimization space for the FOPI controller, with faster convergence than other model-free methods such as metaheuristics or grid search.

4. Discrete FOPI Controller Implementation

The discrete implementation of the FOPI controller (1) is used to perform real-time changes on the controller gains k_p, k_i, and integration order λ. According to [32], the fractional derivative given by the Riemann–Liouville definition (5) with order β can be represented in a discrete form, using the prewarped Tustin definition (6), where T is the sampling time and w_c is the gain crossover frequency.

$$D_t^{\lambda} f(t) = \frac{1}{\Gamma(n-\lambda)} \frac{d^n}{dt^n} \int_a^t \frac{f(\tau)}{(t-\tau)^{\lambda-n+1}} d\tau \tag{5}$$

$$s^{\lambda} = \left(\frac{w_c}{tan(w_c T/2)} \times \frac{1-z^{-1}}{1+z^{-1}} \right)^{\lambda} = M^{\lambda} \times \left(\frac{1-z^{-1}}{1+z^{-1}} \right)^{\lambda}. \tag{6}$$

The Taylor series approximation of (6) is given by (7), where $w = z^{-1}$, N is the order of truncation of the Taylor series, and the coefficients $f_k(\beta)$ are defined in (8). In this paper, the s^{β} operator has an order of truncation $N = 6$.The guideline for selecting the truncation order $N = 6$ for the fractional-order controller is given by the fact that, after several evaluations and real experimentation for lower truncation orders performed in [36], the response of the discrete FOPI controller was not reliable and robust for $N < 6$. In addition, the use of $N = 6$ truncation allows for a faster and memory efficient implementation of the FOPI controller on embedded devices unlike a higher truncation.

$$\left(M \frac{1-z^{-1}}{1+z^{-1}} \right)^{\lambda} = M^{\lambda} \sum_{k=0}^{N} f_k(\lambda) w^k, \tag{7}$$

$$f_k(\lambda) = \frac{1}{k!} \frac{d^k}{dw^k} \left(\frac{1-w}{1+w} \right)^{\lambda} \Bigg|_{w=0}. \tag{8}$$

The infinite gain of integral term $s^{-\lambda}$ of the FOPI controller can be reduced by rewriting it as $s^{-\lambda} = \frac{1}{s} s^{1-\lambda}$, whose Taylor series is given by (9), where, $w = z^{-1}$, $\frac{z+1}{z-1}$ corresponds to the Tustin approximation for the integral term $\frac{1}{s}$, and the parameters $f_N(\lambda) = f_N(1-\lambda) = [1, \; -2\lambda, \; 2\lambda^2, \; -\left(\frac{4}{3}\lambda^3 + \frac{2}{3}\lambda\right), \; +\left(\frac{2}{3}\lambda^4 + \frac{4}{3}\lambda^2\right), \; -\left(\frac{4}{15}\lambda^5 + \frac{4}{3}\lambda^3 + \frac{2}{5}\lambda\right), \; \left(\frac{4}{45}\lambda^6 + \frac{8}{9}\lambda^4 + \frac{46}{45}\lambda^2\right)]$ for $N = [0, 1, \ldots, 6]$. The difference equation for the FOPI controller is given by (10), where $u_p(z)$ and $u_i(z)$ are defined in (11) and $a_{0:5} = f_N(\lambda)$. Further details of the discrete fractional-order PI controller can be found at [36].

$$s^{-\lambda} = M^{-\lambda} \frac{z+1}{z-1} \sum_{k=0}^{N} f_k(1-\lambda) z^{-k} \tag{9}$$

$$u(z) = u_p(z) + u_i(z), \tag{10}$$

$$\begin{aligned} u_p(k) = ke(k);\ \ u_i(k) = \left[\frac{k_i}{\alpha^{\lambda}}(e(k) - e(k-1))\right] - (a_5 - 1)u_i(k-1) \\ -(a_4 - a_5)u_i(k-2) - (a_3 - a_4)u_i(k-3) - (a_2 - a_3)u_i(k-4) \\ -(a_1 - a_2)u_i(k-5) - (a_0 - a_1)u_i(k-6) + a_0 u_i(k-7). \end{aligned} \tag{11}$$

5. An SOC Benchmark for Smart Process Control

The simulation benchmark shown in Figure 3 was built in Matlab/Simulink to evaluate FOSOC controller performance. The simulation benchmark is composed of (1) the GCNM controller and (2) the FOPDT system (12) with the discrete FOPI controller implemented using Stateflow. The FOPDT system is normalized and evaluated in this benchmark, following Remark 11.1.3 presented in Appendix A [38]. The FOSOC is tested under three conditions of the FOPDT system: lag dominated ($L = 0.1$), balanced time ($L = 1$), and delay dominated ($L = 10$). A closed-loop stable FOPI controller is designed as the initial condition for each case using the FMIGO method [33]. In addition, the FOSOC benchmark uses as performance indices the GCNM convergence time, the closed-loop overshoot and settling time, the root mean square value (RMS) of the control action, the integral square error (ISE), the integral absolute error (IAE), and the root mean square error (RMSE) during the benchmark execution time given by (13). The code for the benchmark can be downloaded from https://github.com/tartanus/FOSOCBenchmark, accessed on 10 July 2022. In addition, there is a 300s period reference square signal r. The benchmark parameters are shown in Table 1. Notice that the maximum and minimum constraint values of K_p, K_i, and λ change from each test to ensure the stability of the closed-loop system. For all the benchmark tests, the cost function weights are $W_1 = 5$, $W_2 = 0.1$, $W_3 = 0.1$. Considering that (3) is composed of several optimization objectives, the selection of the weights is based on an extensive simulation analysis of the case study in order to provide a magnitude balance among the cost function terms. It is important to notice that the cost function (3) can be improved by adding a L2 norm or sparse regularization term to improve the optimization landscape.

$$P_o(s) = \frac{1}{s+1}e^{-Ls}, \tag{12}$$

$$RMS = \int_0^{\infty} u(t)dt;\ ISE = \int_0^{\infty} e^2(t)dt;\ IAE = \int_0^{\infty} |e(t)|dt;\ RMSE = \sqrt{\int_0^{\infty} e^2(t)dt}. \tag{13}$$

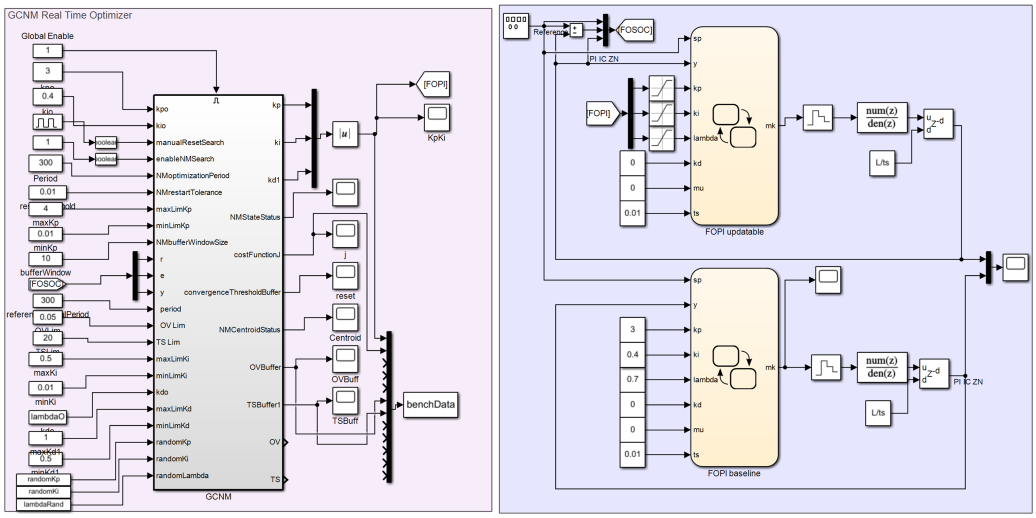

Figure 3. FOSOC benchmark in Matlab/Simulink and fractional-order PI controller implementation using Stateflow.

The time response of the FOSOC controller for the FOPDT system with $L = [0.1,\ 1,\ 10]$ s are shown in Figures 4–6. As can be observed, the FOSOC controller satisfies the overshoot and settling time specifications with a convergence time closer to 2000 s after performing some probabilistic restarts, with an improved performance compared with the initial condition IC given by the FOPI parameters obtained using the FMIGO [39] tuning method, shown in the last columns of Table 1 for each value of L. It corresponds to the starting point for the FOPI optimization obtained by using the FMIGO tuning method for the FOPI controller proposed in [40]. This is also the case for the values for λ. Notice that for each period of the reference signal, one candidate solution provided by the GCNM algorithm is evaluated, which derives in a longer optimization period due to system behavior. Similarly, the performance metrics obtained for the three tests with $L = 0.1,\ 1,\ 10$ are shown in Table 2. In addition, the FOSOC convergence time increases with the deadtime, with the overshoot and settling time satisfying the desired specifications given in Table 1. The other performance indices are similar for the three tests. Thus, we can say that the FOSOC control framework shows promising results for its application in process control towards smart control systems.

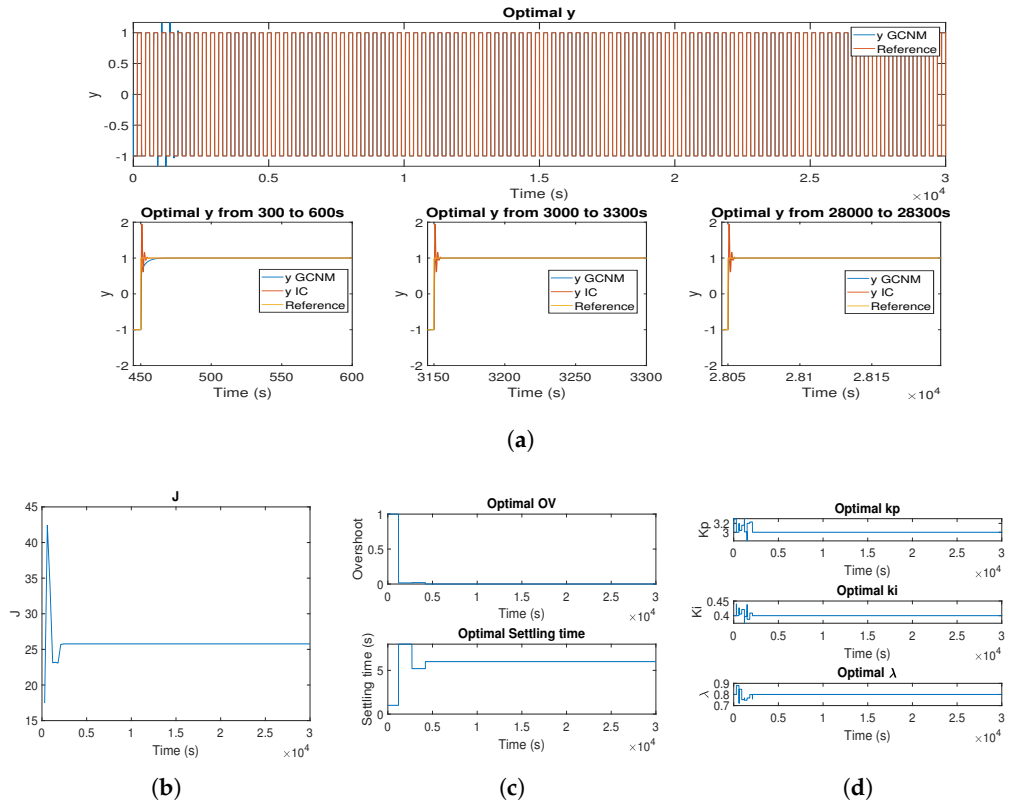

Figure 4. FOSOC controller: (**a**) time response, (**b**) cost function, (**c**) overshoot and settling time, and (**d**) FOPI gains evolution for $L = 0.1$ s.

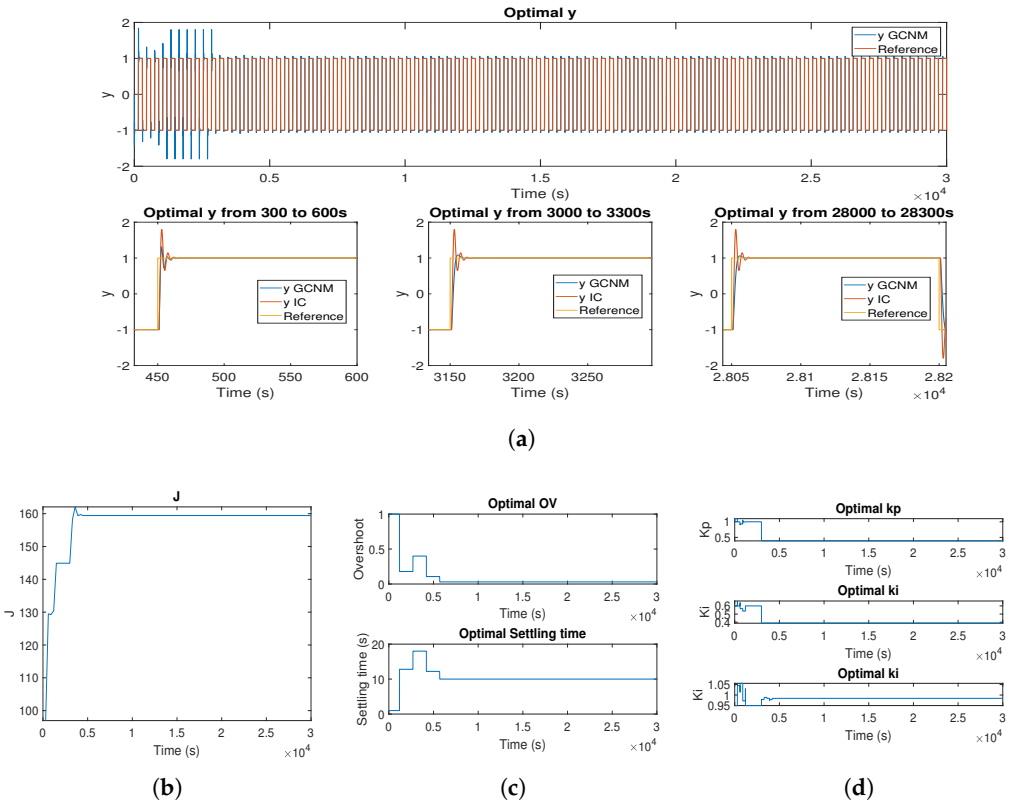

Figure 5. FOSOC controller: (**a**) time response, (**b**) cost function, (**c**) overshoot and settling time, and (**d**) FOPI gains evolution for $L = 1$ s.

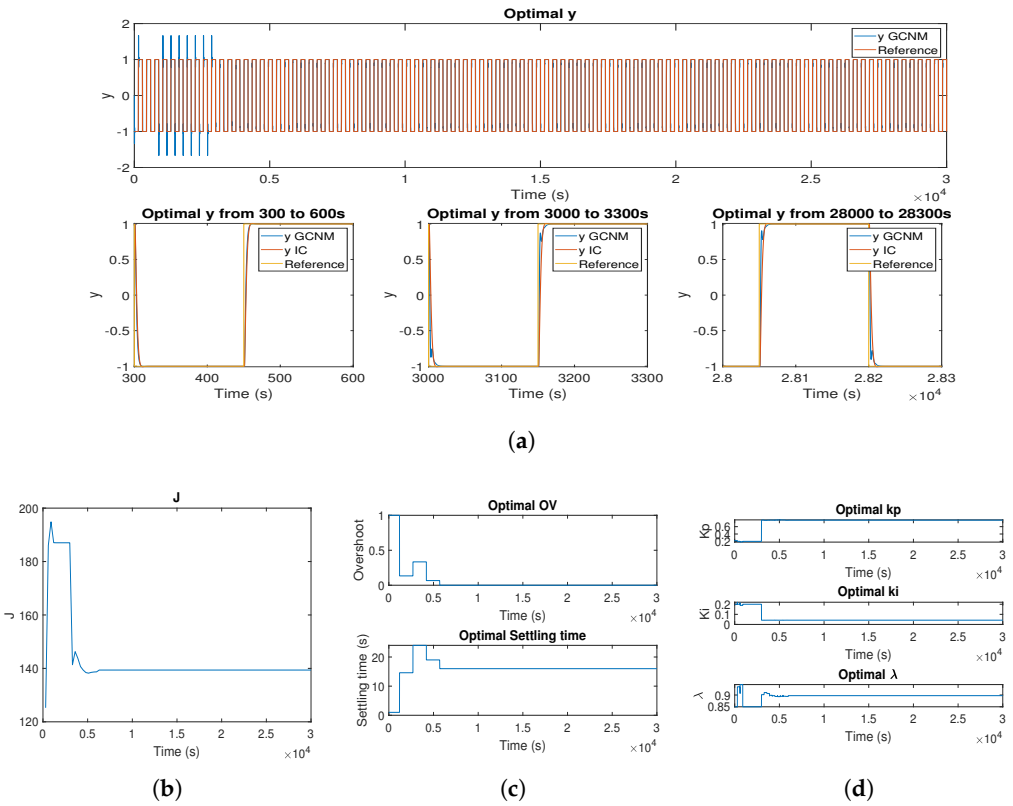

Figure 6. FOSOC controller: (**a**) time response, (**b**) cost function, (**c**) overshoot and settling time, and (**d**) FOPI gains evolution for $L = 10$ s.

Table 1. SOC benchmark configuration parameters.

Delay L (s)	K_p		K_i		λ		Settling Time (s)	OV (%)	FMIGO FOPI		
	min	max	min	max	min	max			K_p	K_i	λ
0.1	0.01	4	0.01	0.5	0.7	1.1	20	5	3.26	8.07	0.7
1	0.01	0.5	0.01	0.1	0.7	1.1	20	5	0.2977	0.0886	1
10	0.01	0.5	0.01	0.2	0.7	1.1	50	5	0.3275	0.1618	1.1

Table 2. SOC benchmark performance indices.

Dead Time L (s)	SOC Convergence Time (s)	OV	Settling Time (s)	RMS	ISE	IAE	RMSE
L = 0.1	2000	0.01	10	0.9934	0.03143	0.989	0.095
L = 1	4500	0.02	12	0.977	0.045	0.973	0.0958
L = 10	4500	0.06	21	0.967	0.41	0.94	0.984

6. Accelerating Self Optimizing Control for FOPI Controller with Fractional-Order Stochasticity

As stated before, the GCNM algorithm uses a normal distribution to pick up the new initial conditions after each probabilistic restart. However, this process can be performed by using a fractional-order Gaussian distribution to increase the convergence speed of the algorithm. The fractional-order Gaussian noise can be represented as the change in Brownian motion step defined by the Riemann–Liouville fractional integral (14), where $dB(s)$ is the general definition of white noise, $\Gamma(\cdot)$ is the gamma function, and H is the Hurst exponent, which indicates the LRD property of the random disturbance signal [41,42]. According to the value of H, the fractional-order randomness can represent a Brownian motion if $H = 0.5$, positively correlated if $0.5 < H < 1$, and negatively correlated if $0 < H < 0.5$.

$$B_H(t) = \frac{1}{\Gamma(H + 1/2)} \int_0^t (t - s)^{H-0.5} dB(s). \tag{14}$$

The optimal randomness evaluation is performed using the normalized FOPDT system (12) for three different delay values $L = 0.1, 1, 10$ corresponding to the time constant dominated, balanced time, and delay time dominated behaviors of the system. For each delay value, the Hurst exponent of the fractional-order Gaussian noise is evaluated as H = 0.1:0.1:0.9, where $H = 0.5$ corresponds to the normal distribution and the remaining H values to the fractional-order noise. A total of 50 evaluations are performed for each value of H for the different delay values. Thus, the average value for each performance indicator will be considered as the reference performance index for the system. As an example, Figure 7 shows the sequences for $[k_p, k_i, \lambda]$ used for the SOC benchmark evaluation with $L = 1$ s with Hurst exponent $H = 0.5$.

The SOC controller is tested for the three scenarios defined in the previous section with normal and fractional-order Gaussian noise. As an example, Figure 8 shows the performance of the SOC controller for $L = 0.1$ s with a Hurst exponent $H = 0.5$ that is equivalent to the normal distribution. As can be observed, the PI controller begins with the values provided by the FMIGO tuning and then, after each execution cycle, the GCNM performs a real-time optimization that searches for the optimal values of the FOPI controller gains until the desired performance specifications are satisfied according to the cost function.

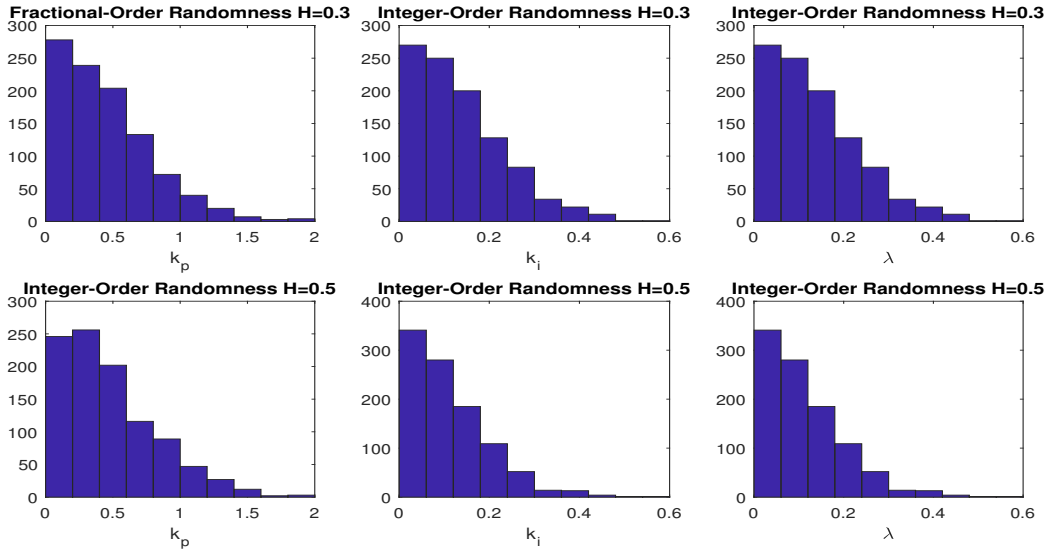

Figure 7. Fractional order randomness employed for the SOC controller with $H = 0.3,\ 0.5$.

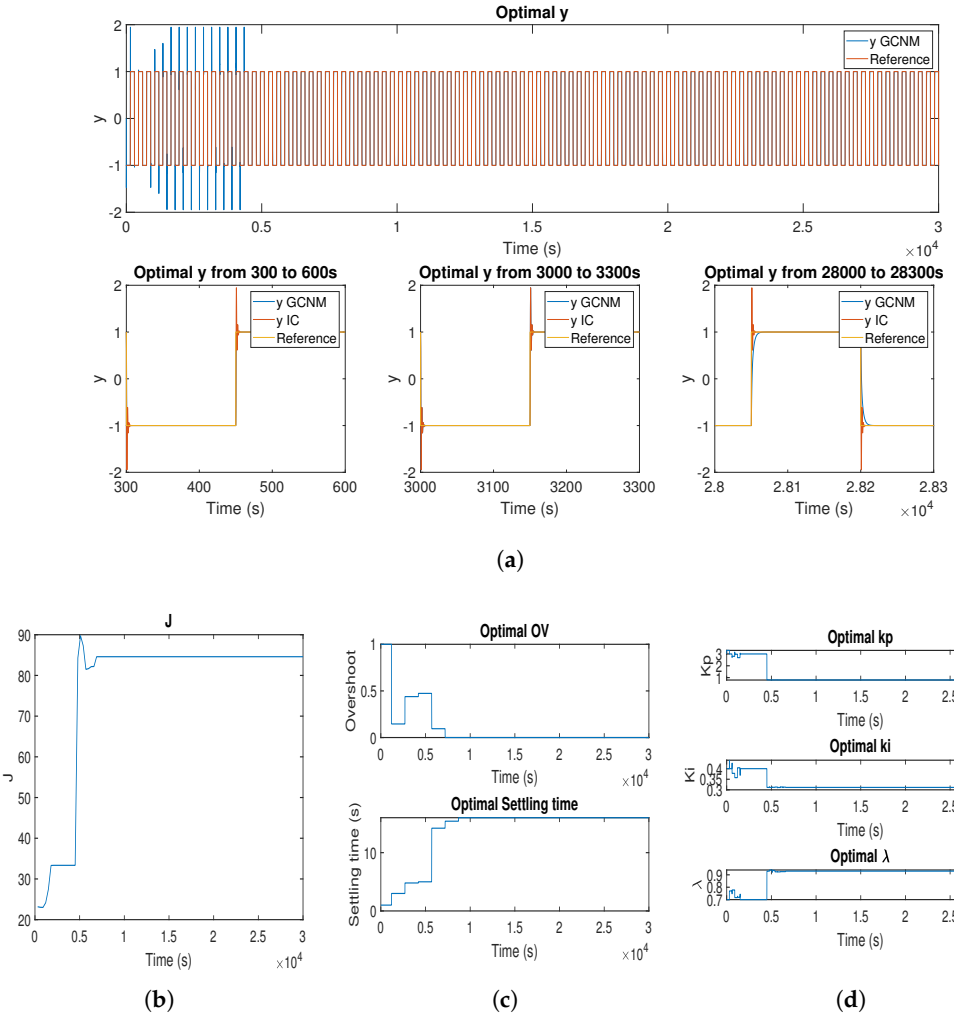

Figure 8. SOC controller: (**a**) time response, (**b**) cost function, (**c**) overshoot and settling time, and (**d**) PI gains evolution for $L = 0.1$ with $H = 0.7$.

The average performance of the SOC controller with GCNM algorithm for $L = 0.1, 1, 10$ regarding to the mean overall convergence time, closed-loop settling time, and overshoot

of the GCNM algorithm after 50 iterations is shown in Figure 9 for the different Hurst exponents H. As can be observed, the GCNM convergence time is reduced by 10% when the fractional-order randomness has a negative LRD (H = 0.2 to 0.4), reducing the algorithm convergence time. Similarly, Figure 9 shows the closed-loop settling time and overshoot of the system, which indicates that for all H the optimization conditions are satisfied and at $H = 0.3$ has the lowest values for settling time.

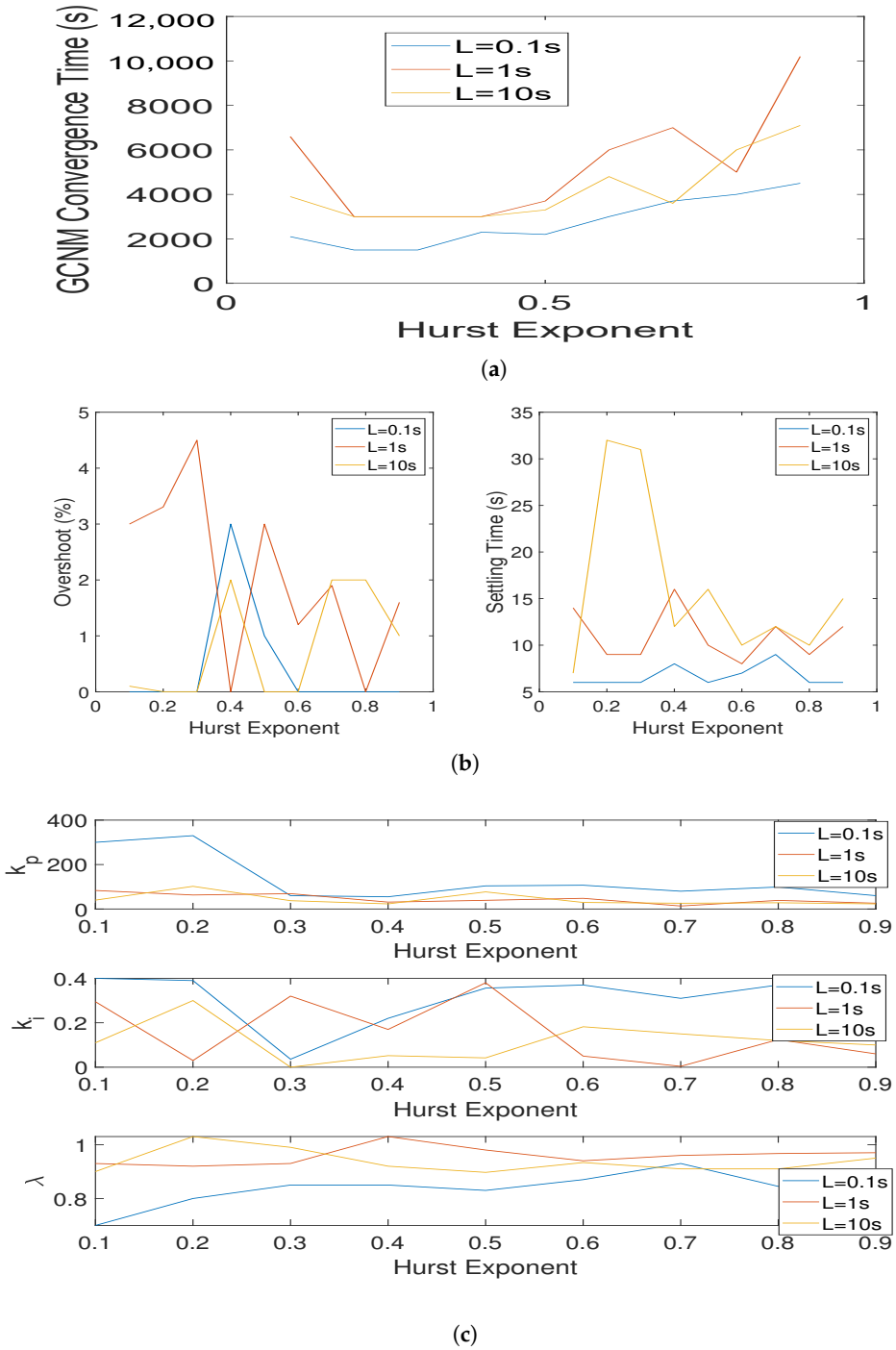

Figure 9. SOC controller: (**a**) overall convergence time, (**b**) overshoot and settling time, and (**c**) final PI gains after 50 iterations for $L = 0.1, 1, 10$.

In the case of the balanced system $L = 1$, the mean overall convergence of the GCNM convergence time is reduced by 20% when fractional-order randomness is employed, reaching its minimum value at $H = 0.3$. Regarding to the overshoot and settling time,

the minimum values are reached at $H = 0.3$ and $H = 0.4$, respectively. However, in all the cases for the fractional order randomness, the performance specifications are satisfied.

For the delay dominated system $L = 10$, the mean overall convergence of the GCNM convergence time is reduced by 19%, with fractional-order randomness reaching its minimum value at $H = 0.3$. In the case of the overshoot and settling time, the minimum values are reached at $H = 0.3$ and $H = 0.4$, respectively, again satisfying the performance specifications for all the Hurst exponents. For this particular case, the overall convergence time of the algorithm is reduced due to a strong lower boundary on the integral gain required to keep the system stability caused by the bigger delay of the system.

Finally, the average RMS, RMSE, ISE, and IAE values for the three tests $L = 0.1, 1, 10$ after 50 repetitions are similar with integer or fractional order randomness, indicating that the optimization process is able to achieve a similar performance for the different cases according to the economic cost function (3). Therefore, and based on the obtained results, we can say that the fractional-order randomness is able to reduce the overall convergence time of the SOC controller in the range of 10% to 20% compared with the integer-order Gaussian noise. It means that the fractional-order randomness with negative LRD can improve the self optimizing control performance by accelerating the convergence of the optimization process without modifying the search algorithm or the SOC controller structure. The fractional-order randomness can be considered as a suitable alternative to improve the performance not only of real-time SOC controllers but also any searching process.

7. Comparison of the FOSOC GCNM Controller with Other Self-Tuning PID Control Methods in the Literature

The proposed fractional-order SOC control for FOPI controllers is compared with several other self-tuning control methods in the literature. Table 3 presents the comparison analyzing several criteria of the self-tuning controllers, including the cost function used, the type of controller, tuning methods, number of iterations, and if the control approach is offline or real-time. As can be observed, in most of the cases, the self-tuning controllers relying on optimization algorithms are developed on an offline setup, except for the SOC GCNM control method. The number of iterations for these algorithms is between 10 and 50. Something important about the iterations on the GCNM is that each iteration should be accounted for when the GCNM performs a complete operation of reflection, contraction, expansion, or shrinking after evaluating the n + 1 vertices of the simplex. Thus, in the FOPI case, each iteration will take approximately 2000 s. For these reasons, the iterations required to converge the GCNM SOC on the first two rows of Table 3 are considerably faster than the other methods.

For the population-based optimization algorithms, one iteration is considered when all the candidate solutions are evaluated at one step. Each iteration involves several function evaluations, increasing the time required to perform the optimization. If these function evaluations were performed over the real system in the SOC configuration proposed in this paper, for example, using genetic algorithms or any other population-based algorithm, the overall convergence time of the algorithm would be much longer than the one resulting from the SOC GCNM shown in the second column of Table 2. For these reasons, the GCNM optimization algorithm with SOC provides a much more reasonable convergence time than population-based algorithms.

In the case of adaptive algorithms, such as sliding surface control or real-time identification, knowledge of the model is required to determine an appropriate tuning of the controller parameters, which can be hard to obtain if the system is complex and constantly changing.

In the particular case of extremum seeking control (ESC), it behaves similarly to the GCNM SOC. The iterations are related in a similar way to the function evaluations, which have a better performance compared with the population-based algorithms.

Therefore, it is possible to say that the GCNM SOC control is a suitable alternative for the PID/FOPID controller optimization in real-time when little or no system knowledge is

available, with much better performance than population-based algorithms executed on a real-time setup.

Table 3. Comparison of self tuning methods for PI, PID, FOPI, and FOPID controllers.

Paper	Year	Author	Controller	Cost Function	Optimization Method	Iterations	Real-Time /Offline
Proposed method		Viola and Chen	FOPI	Squared error overshoot settling time	SOC GCNM	1	real-time
[43]	2021	Viola and Chen	PID PI	Squared error overshoot settling time	SOC GCNM	3	real-time
[18]	2016	Farhan	FOPI fuzzy	Error	Gradient based optimization (metaheuristic)	30	offline
[20]	2019	Hekimoğlu	FOPID	ITAE	CHaSO Algorithm	30	offline
[19]	2021	Izci and Ekinci	FOPID PID	ITAE	Harris Hawk	40	offline
[44]	2006	Kilingsworh and Krtics	PID	IAE ITAE ITSE	Extremum seeking control	13	real-time
[45]	2006	Suardiaz	PID	Online identification	Pole placement and online identification	-	real-time
[46]	2021	Rodriguez Abreo	PID	Settling time squared error phase margin	Genetic algorithm	-	offline
[47]	2012	J. L. Meza	PID fuzzy	Error velocity	Fuzzy logic	-	real-time
[48]	2014	Kumar	PID	Square error	Cukoo search	-	offline
[49]	2003	Chen	PID	Phase and gain margin	Relay feedback control	-	real-time
[50]	2013	Moradi	PID	Error	Sliding surface control	-	real-time
[51]	2022	Baz et al.	PID	Error	Fuzzy logic functions	-	real-time
[52]	2018	Oliveira	PID	ITAE	Particle swarm optimization	50	offline
[53]	2021	Vanchinathan	FOPID	ISE	Artificial Bee Colony	10	offline
[54]	2008	Altinten	PID	IAE	Genetic algorithm	30	offline

8. Conclusions and Future Work

This paper presented a FOSOC framework to improve a closed-loop system performance. A benchmark was built in Matlab/Simulink to evaluate the FOSOC controller performance using a FOPI controller under three different tests. In addition, a more optimal self optimizing control (SOC) employing fractional-order Gaussian noise (fGn) with long range dependence (LRD) to improve the optimization convergence time of the GCNM (globalized constrained Nelder–Mead) algorithm was presented. The obtained results show that the FOSOC controller enhances the system closed-loop performance and ensures stability and robustness for different dead times. Additionally, the accelerated SOC using fractional order stochasticity was evaluated using fractional Gaussian noise with different LRD given by the Hurst exponent. It can be noticed that using negative LRD ($H = [0.2, 0.3]$), the convergence time of the SOC is reduced significantly compared with classic Gaussian noise. It indicates that fractional-order stochasticity is more efficient in improving the SOC convergence speed. It is also a suitable alternative for other optimization algorithms

that heavily rely on random events, such as several metaheuristics or stochastic gradient methods. In future work, the implementation of the FOSOC algorithm and its extension to a fractional-order PID controller will be proposed and experimentally validated with thermal systems.

Author Contributions: Conceptualization, J.V. and Y.C.; methodology, Y.C.; software, J.V.; validation, J.V. and Y.C.; formal analysis, Y.C.; investigation, J.V.; resources, J.V. and Y.C.; data curation, J.V.; writing—original draft preparation, J.V.; writing—review and editing, J.V. and Y.C.; visualization, J.V.; supervision, Y.C.; project administration, Y.C.; funding acquisition, Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: https://github.com/tartanus/FOSOCBenchmark, accessed on 10 July 2022.

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

Appendix A. Remark 11.31 [40]

The FOPDT system (A1) can be normalized as follows:

$$p(s) = \frac{K}{\tau s + 1} e^{-Ls} \tag{A1}$$

$$P_n(s) = \frac{1}{\tau s + 1} e^{-L/\tau} \tau s = \frac{1}{s' + 1} e^{-L's'}, \tag{A2}$$

where $s' = \tau s$ and $L' = L/T$. The parameter K in (2) can be normalized as 1, as the steady-state gain of the plant can always be used as part of the gain of the PID controller. Therefore, if the complete information of the achievable set of specifications phase margin ϕ_m and gain crossover frequency w_c is collected for the standard form of the control system plant below:

$$P_o(s) = \frac{1}{s + 1} e^{-Ls}. \tag{A3}$$

where L is equal to L' in (A2), then the complete achievable region of the specifications ϕ_m and w_c can be easily found for the normalized FOPDT system (A2), with the proportional change of the w_c axis, $w_c = w\ c/T$ as $s' = Ts$.

References

1. Skogestad, S. Plantwide control: The search for the self-optimizing control structure. *J. Process Control* **2000**, *10*, 487–507. [CrossRef]
2. Francisco, M.; Vega, P.; Skogestad, S. Nonlinear offset free MPC for self-optimizing control in wastewater treatment plants. In Proceedings of the 19th International Conference on System Theory, Control and Computing (ICSTCC), Cheile Gradistei, Romania, 14–16 October 2015; pp. 390–395.
3. Reza, M.; Mehdi, M.; Panahi, M. Convex reformulations for self-optimizing control optimization problem : Linear Matrix Inequality approach. *J. Process Control* **2022**, *116*, 172–184. [CrossRef]
4. Ye, L.; Cao, Y.; Yang, S. Global self-optimizing control with active-set changes: A polynomial chaos approach. *Comput. Chem. Eng.* **2022**, *159*, 107662. [CrossRef]
5. Bariyur, K.; Krstic, M. *Real-Time Optimization by Extremum-Seeking Control*; Wiley-Interscience: Hoboken, NJ, USA 2003; p. 230. [CrossRef]
6. Ahn, H.S.; Chen, Y.Q.; Moore, K.L. Iterative learning control: Brief survey and categorization. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2007**, *37*, 1099–1121. [CrossRef]
7. Çıtmacı, B.; Luo, J.; Jang, J.B.; Canuso, V.; Richard, D.; Ren, Y.M.; Morales-Guio, C.G.; Christofides, P.D. Machine learning-based ethylene concentration estimation, real-time optimization and feedback control of an experimental electrochemical reactor. *Chem. Eng. Res. Des.* **2022**, *185*, 87–107. [CrossRef]

8. Yin, C.; Chen, Y.; Cheng, Y.; Zhong, S.M.; Tian, L. Maximum power point tracking in photovoltaic system through extremum seeking control with fo switching technique. In Proceedings of the 2015 IEEE 54th Annual Conference on Decision and Control (CDC), Osaka, Japan, 15–18 December 2015; Volume 9, pp. 5629–5634. [CrossRef]
9. Hollenbeck, D.; Chen, Y.Q. A more optimal stochastic extremum seeking control using fractional dithering for a class of smooth convex functions. *IFAC-PapersOnLine* **2020**, *53*, 3737–3742. [CrossRef]
10. Malek, H.; Chen, Y. A single-stage three-phase grid-connected photovoltaic system with fractional order MPPT. In Proceedings of the IEEE Applied Power Electronics Conference and Exposition (APEC), Fort Worth, TX, USA, 16–20 March 2014; pp. 1793–1798. [CrossRef]
11. Viola, J.; Hollenbeck, D.; Rodriguez, C.; Chen, Y. Fractional-Order Stochastic Extremum Seeking Control with Dithering Noise for Plasma Impedance Matching. In Proceedings of the 2021 IEEE Conference on Control Technology and Applications (CCTA), San Diego, CA, USA, 9–11 August 2021; pp. 247–252.
12. Tani, T.; Matsuo, K. Robust closed-loop real-time optimization for refinery utility plant with model predictive control for constraint handling. In Proceedings of the 2009 IEEE International Conference on Industrial Technology, Churchill, VIC, Australia, 10–13 February 2009. [CrossRef]
13. Zgraggen, A.U.; Fagiano, L.; Morari, M. Real-time optimization and adaptation of the crosswind flight of tethered wings for airborne wind energy. *IEEE Trans. Control Syst. Technol.* **2015**, *23*, 434–448. [CrossRef]
14. Shuofeng, Z.; Amini, M.R.; Sun, J.; Mi, C. A Two-Layer Real-Time Optimization Control Strategy for Integrated Battery Thermal Management and HVAC System in Connected and Automated HEVs. *IEEE Trans. Veh. Technol.* **2021**, *70*, 6567–6576. [CrossRef]
15. Cimini, G.; Bernardini, D.; Levijoki, S.; Bemporad, A. Embedded Model Predictive Control with Certified Real-Time Optimization for Synchronous Motors. *IEEE Trans. Control Syst. Technol.* **2021**, *29*, 893–900. [CrossRef]
16. Altbawi, S.M.A.; Mokhtar, A.S.B.; Jumani, T.A.; Khan, I.; Hamadneh, N.N.; Khan, A. Optimal design of Fractional order PID controller based Automatic voltage regulator system using gradient-based optimization algorithm. *J. King Saud Univ.-Eng. Sci.* **2021**. [CrossRef]
17. Liu, Y.; Fan, K.; Ouyang, Q. Intelligent Traction Control Method Based on Model Predictive Fuzzy PID Control and Online Optimization for Permanent Magnetic Maglev Trains. *IEEE Access* **2021**, *9*, 29032–29046. [CrossRef]
18. Farhan, M.; Ullah, N.; Ahmed, N. A single parameter self-tune fractional order PIλ controller for second order uncertain dynamical system. In Proceedings of the ICET 2016—2016 International Conference on Emerging Technologies, Islamabad, Pakistan, 18–19 October 2016; pp. 3–8. [CrossRef]
19. Izci, D.; Ekinci, S. An Efficient FOPID Controller Design for Vehicle Cruise Control System Using HHO Algorithm. In Proceedings of the HORA 2021—3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications, Ankara, Turkey, 11–13 June 2021. [CrossRef]
20. Hekimoglu, B. Optimal Tuning of Fractional Order PID Controller for DC Motor Speed Control via Chaotic Atom Search Optimization Algorithm. *IEEE Access* **2019**, *7*, 38100–38114. [CrossRef]
21. Xu, C.; Liao, M.; Li, P.; Yao, L.; Qin, Q.; Shang, Y. Chaos Control for a Fractional-Order Jerk System via Time Delay Feedback Controller and Mixed Controller. *Fractal Fract.* **2021**, *5*, 257. [CrossRef]
22. Chen, H.; Xie, W.; Chen, X.; Han, J.; Aït-Ahmed, N.; Zhou, Z.; Tang, T.; Benbouzid, M. Fractional-Order PI Control of DFIG-Based Tidal Stream Turbine. *J. Mar. Sci. Eng.* **2020**, *8*, 309. [CrossRef]
23. Xu, K.; Chen, L.; Wang, M.; Lopes, A.M.; Tenreiro Machado, J.A.; Zhai, H. Improved Decentralized Fractional PD Control of Structure Vibrations. *Mathematics* **2020**, *8*, 326. [CrossRef]
24. Alam, M.S.; Al-Ismail, F.S.; Abido, M.A. PV/Wind-Integrated Low-Inertia System Frequency Control: PSO-Optimized Fractional-Order PI-Based SMES Approach. *Sustainability* **2021**, *13*, 7622. [CrossRef]
25. Gao, Z. A Tuning Method via Borges Derivative of a Neural Network-Based Discrete-Time Fractional-Order PID Controller with Hausdorff Difference and Hausdorff Sum. *Fractal Fract.* **2021**, *5*, 23. [CrossRef]
26. Yin, C.; Chen, Y.; Zhong, S.M. Fractional-order sliding mode based extremum seeking control of a class of nonlinear systems. *Automatica* **2014**, *50*, 3173–3181. [CrossRef]
27. Wei, J.; Chen, Y.Q.; Yu, Y.; Chen, Y. Optimal randomness in swarm-based search. *Mathematics* **2019**, *7*, 828. [CrossRef]
28. Ahmed, W.A.E.M.; Mageed, H.M.A.; Mohamed, S.A.E.; Saleh, A.A. Fractional order Darwinian particle swarm optimization for parameters identification of solar PV cells and modules. *Alex. Eng. J.* **2021** , *6*, 1249–1263. [CrossRef]
29. Chen, Y.; Gao, Q.; Wei, Y.; Wang, Y. Study on fractional order gradient methods. *Appl. Math. Comput.* **2017**, *314*, 310–321. [CrossRef]
30. Niu, H.; Wei, J.; Chen, Y. Optimal randomness for stochastic configuration network (SCN) with heavy-tailed distributions. *Entropy* **2021**, *23*, 56. [CrossRef] [PubMed]
31. Zhang, G.; Chen, Y.Q. More Informed Random Sample Consensus. In Proceedings of the 2020 8th International Conference on Control, Mechatronics and Automation (ICCMA 2020), Moscow, Russia, 6–8 November 2020; pp. 197–201. [CrossRef]
32. Merrikh-Bayat, F.; Mirebrahimi, N.; Khalili, M.R. Discrete-time fractional-order PID controller: Definition, tuning, digital realization and some applications. *Int. J. Control. Autom. Syst.* **2015**, *13*, 81–90. [CrossRef]
33. Monje, C.A.; Chen, Y.Q.; Feliu-Batlle, V.; Xue, D.; Vinagre, B.M. *Fractional-Order Systems and Controls Fundamentals and Applications*; Springer: London, UK, 2010; p. 430.

34. Viola, J.; Chen, Y. A Fractional-Order On-line Self Optimizing Control Framework and a Benchmark Control System. In Proceedings of the International Conference on Fractional Differentiation and its Applications (ICFDA 2020), Warsaw, Poland, 6–8 September 2021; p. 6.
35. Viola, J.; Chen, Y. An Accelerated Self Optimizing Control Framework for Smart Process Control Using Fractional Order Stochasticity. In Proceedings of the 2021 9th International Conference on Control, Mechatronics and Automation (ICCMA), Luxembourg, 11–14 November 2021; pp. 104–109. [CrossRef]
36. Viola, J.; Angel, L.; Sebastian, J.M. Design and robust performance evaluation of a fractional order PID controller applied to a DC motor. *IEEE/CAA J. Autom. Sin.* **2017**, *4*, 304–314. [CrossRef]
37. Gao, F.; Han, L. Implementing the Nelder-Mead simplex algorithm with adaptive parameters. *Comput. Optim. Appl.* **2014**, 51, 259—277 . [CrossRef]
38. Xue, D.; Chen, Y.; Atherton, D. *Linear Feedback Control, Analysis and Design with MATLAB–Advances in Design and Control*; SIAM: Philadelphia, PA, USA, 2007; p. 354.
39. Chen, Y.Q.; Bhaskaran, T.; Xue, D. Practical tuning rule development for fractional order proportional and integral controllers. *J. Comput. Nonlinear Dyn.* **2008**, *3*, 021403. [CrossRef]
40. Luo, Y.; Chen, Y.Q. *Fractional Order Motion Controls*; Wiley: Chichester, UK, 2012; p. 456.
41. Magin, R.L. Fractional calculus in bioengineering. *Crit. Rev. Biomed. Eng.* **2004**, *32*, 1–104. [CrossRef]
42. Sheng, H.; Chen, Y.; Qiu, T. *Fractional Processes and Fractional-Order Signal Processing Techniques and Applications*; Springer: Berlin/Heidelberg, Germany, 2012; p. 322.
43. Viola, J.; Chen, Y. A Self Optimizing Control Framework and a Benchmark for Smart Process Control. In Proceedings of the 2021 3rd International Conference on Industrial Artificial Intelligence (IAI), Shenyang, China, 8–11 November 2021; pp. 1–6. [CrossRef]
44. Killingsworth, N.; Krstic, M. PID tuning using extremum seeking: Online, model-free performance optimization. *IEEE Control Syst. Mag.* **2006**, *26*, 70–79. [CrossRef]
45. Suardíaz Muro, J.; Al-Hadithi, B.M.; García, A.I. Controlador auto-parametrizable dedicado al control del caudal de un fluido. *IEEE Lat. Am. Trans.* **2006**, *4*, 332–338. [CrossRef]
46. Rodriguez-Abreo, O.; Rodriguez-Resendiz, J.; Fuentes-Silva, C.; Hernandez-Alvarado, R.; Falcon, M.D.C.P.T. Self-Tuning Neural Network PID with Dynamic Response Control. *IEEE Access* **2021**, *9*, 65206–65215. [CrossRef]
47. Meza, J.L.; Santibáñez, V.; Soto, R.; Llama, M.A. Fuzzy self-tuning PID semiglobal regulator for robot manipulators. *IEEE Trans. Ind. Electron.* **2012**, *59*, 2709–2717. [CrossRef]
48. Kashyap, A.K.; Parhi, D.R. Particle Swarm Optimization aided PID gait controller design for a humanoid robot. *ISA Trans.* **2021**, *114*, 306–330. [CrossRef] [PubMed]
49. Relay Feedback Tuning of Robust PID Controllers With Iso-Damping Property. *Proc. IEEE Conf. Decis. Control* **2003**, *3*, 2180–2185.
50. Moradi, M. Self-tuning PID controller to three-axis stabilization of a satellite with unknown parameters. *Int. J. Non-Linear Mech.* **2013**, *49*, 50–56. [CrossRef]
51. Baz, R.; Majdoub, K.E.; Giri, F.; Taouni, A. Self-tuning fuzzy PID speed controller for quarter electric vehicle driven by In-wheel BLDC motor and Pacejka's tire model. *IFAC-PapersOnLine* **2022**, *55*, 598–603. [CrossRef]
52. de Moura Oliveira, P. Design of Digital PID Controllers using Particle Swarm Optimization: A Video Based Teaching Experiment. *IFAC-PapersOnLine* **2018**, *51*, 298–303. [CrossRef]
53. Vanchinathan, K.; Selvaganesan, N. Adaptive fractional order PID controller tuning for brushless DC motor using Artificial Bee Colony algorithm. *Results Control Optim.* **2021**, *4*, 100032. [CrossRef]
54. Altınten, A.; Ketevanlioğlu, F.; Erdoğan, S.; Hapoğlu, H.; Alpbaz, M. Self-tuning PID control of jacketed batch polystyrene reactor using genetic algorithm. *Chem. Eng. J.* **2008**, *138*, 490–497. [CrossRef]