Annals of Operations Research (2022) 312:1261–1305
https://doi.org/10.1007/s10479-020-03871-7

ORIGINAL RESEARCH

Designing a hybrid reinforcement learning based algorithm with application in prediction of the COVID-19 pandemic in Quebec

Soheyl Khalilpourazari[1,2] · Hossein Hashemi Doulabi[1,2]

Accepted: 7 November 2020 / Published online: 3 January 2021

Abstract
World Health Organization (WHO) stated COVID-19 as a pandemic in March 2020. Since then, 26,795,847 cases have been reported worldwide, and 878,963 lost their lives due to the illness by September 3, 2020. Prediction of the COVID-19 pandemic will enable policymakers to optimize the use of healthcare system capacity and resource allocation to minimize the fatality rate. In this research, we design a novel hybrid reinforcement learning-based algorithm capable of solving complex optimization problems. We apply our algorithm to several well-known benchmarks and show that the proposed methodology provides quality solutions for most complex benchmarks. Besides, we show the dominance of the offered method over state-of-the-art methods through several measures. Moreover, to demonstrate the suggested method's efficiency in optimizing real-world problems, we implement our approach to the most recent data from Quebec, Canada, to predict the COVID-19 outbreak. Our algorithm, combined with the most recent mathematical model for COVID-19 pandemic prediction, accurately reflected the future trend of the pandemic with a mean square error of 6.29E−06. Furthermore, we generate several scenarios for deepening our insight into pandemic growth. We determine essential factors and deliver various managerial insights to help policymakers making decisions regarding future social measures.

Keywords COVID-19 pandemic · SARS-Cov-2 · Reinforcement learning · SIDARTHE · Machine learning

✉ Soheyl Khalilpourazari
soheyl.khalilpourazari@mail.concordia.ca

Hossein Hashemi Doulabi
hossein.hashemi@concordia.ca

[1] Department of Mechanical, Industrial and Aerospace Engineering, Concordia University, Montreal, Canada

[2] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT), Montreal, Canada

1 Introduction

Researchers use optimization in nearly every study area. Optimization remains a fundamental challenge in science and engineering, primarily because of the difficulty of real-world problems and the limitations of traditional methods. Randomization Search Algorithms (RSAs) are among the most flexible and most efficient methodologies to solve complicated problems. These algorithms are mostly polynomial-time algorithms and have significantly lower computational complexity. As one of the most commonly used RSAs, metaheuristics are algorithms that are inspired by natural phenomena to perform optimization. Metaheuristics perform very well in exploring the feasible region and evade local optimum using effective movement processes.

Healthcare science is one of the top research areas in which metaheuristics have been widely applied to. Using these algorithms, scientists can optimize healthcare systems significantly in terms of several objectives, including minimizing cost, waiting time, service time, delivery time, and maximizing reliability or customer satisfaction. In December 2019, a novel strain of coronavirus called SARS-Cov-2 discovered in China. The virus causes COVID-19, a severe respiratory disease. Regardless of primary measures applied by the government of China, the disease spread quickly to many countries leading to 26,795,847 infected cases and 878,963 deaths. Currently, there are no effective medications and vaccines for the disease. However, the effectiveness of some treatment options is under study via clinical trials (Health Canada 2020; WHO 2020). Although most people with mild COVID-19 symptoms recover independently, some other people with severe and critical symptoms need hospitalization in wards and Intensive Care Units (Public Health Authority of Canada 2020). However, due to the limited capacity of the healthcare system, it is impossible to admit all the patients in hospitals.

Efficient modeling and prediction of the COVID-19 pandemic will meaningfully aid the officials and healthcare experts in making decisions to stop the spread of the disease. Besides, by forecasting the upcoming trend of the epidemic, we can also optimally allocate resources to hospitals that will avoid equipment shortages and save patients' lives. Prediction of the COVID-19's trend is challenging because of its uncertain nature and complication. Recently, scientists have provided a novel model to simulate the COVID-19 pandemic called SIDARTHE and was initially offered in research by Giordano et al. (2020) published in *Nature Medicine*. The researchers highlighted the efficiency of the proposed formulation in modeling the pandemic growth. Nevertheless, they emphasized that solving the presented set of differential equations is difficult because of the exceptional characteristics of the model.

In the current research, we offer a novel search methodology that will solve many complex optimization problems very efficiently in a short time. Our algorithm simultaneously benefits from the advantages of machine learning (ML) and evolutionary computation (EC). In our research, we propose a hybrid algorithm that involves a reinforcement learning (RL) technique as the main engine and several ECs as updating operators. The learning process achieved by the RL method accelerates the algorithm and enables us to resolve complicated large-scale problems. The procedure utilizes several operators to enhance the exploration and exploitation capabilities of the algorithm. These features help the proposed process to sidestep local optimum while exploiting the solution space intelligently. We implement the algorithm on several well-known recently developed benchmarks to show its efficiency in solving such complex problems. Moreover, we highlighted significant differences in the performance of the method comparing to other methodologies using robust statistical tests.

Furthermore, we use the proposed algorithm to model and predict the COVID-19 pandemic in Quebec, Canada, that has the most cases of COVID-19 in the country. Our results accurately predict the peak in the number of infected cases of COVID-19 in the province. The outcomes also determine the peak of the number of cases that develop life-threatening symptoms that will require hospitalization. We also perform complex sensitivity analyses to portray future scenarios that enable us to provide detailed information to policymakers and healthcare professionals. In our study, we also measure the effectiveness of implementing measures such as social distancing and partial lockdown on pandemic growth.

The rest of the current study is prepared as follows: In Sect. 2, we deliver a comprehensive review of existing research on the topic. In Sect. 3, we offer a novel algorithm to resolve many complex problems using RL and EC. In Sect. 4, we apply our algorithm to several well-known benchmark functions. We assess the performance of the suggested approach and compare its efficiency to state-of-the-art methods. In Sect. 5, we implement our method to model and predict the COVID-19 pandemic in Quebec, Canada. In Sect. 6, we perform sensitivity analyses and provide valuable managerial insights to fight the COVID-19 pandemic. In Sect. 7, we conclude the paper.

2 Survey on research conducted

The core idea behind most EC algorithms is to follow a swarm intelligence that is inspired by animal behavior and natural phenomenon (Khalilpourazari and Pasandideh 2019). Mirjalili and Lewis (2016) categorized metaheuristics into four main groups: evolutionary algorithms (EAs), physics-based algorithms (PAs), swarm algorithms (SAs), and machine learning-based algorithms (MLAs). EAs imitate the evolution procedure in nature to solve complex problems. PAs utilize laws of physics that enable this family of algorithms to handle complicated problems. On the other hand, SAs simulate the swarm behavior of many individuals in a group. Also, MLAs use artificial intelligence and machine learning to enhance the performance of the previous families of algorithms in terms of exploration and exploitation. Table 1 provides some of the most advanced algorithms on metaheuristics developed in recent years.

Many researchers used these algorithms to solve complex optimization problems in different fields (Hoursan et al. 2020; Tirkolaee et al. 2020a, b; Sangaiah et al. 2020; Lotfi et al. 2018, 2019, 2020; Zare Mehrjerdi and Lotfi 2019; Khalilpourazari et al. 2020c; Hashemi Doulabi et al. 2020a, b). Defined by the no free lunch (NFL) theorem, we can logically prove that no single method performs optimally in resolving all problems (Adam et al. 2019; Wolpert and Macready 1997). Any metaheuristic algorithm may perform well in some benchmarks but weak in others. This theorem makes this field of study highly interesting for researchers searching for an algorithm that performs promising in many benchmarks. In our algorithm, we consider several ECs and operators and let an RL method decide which algorithm to use to relocate each element. Besides, learning during the optimization process will significantly reduce the computational burden and improve the quality of the results. The learning process accelerates the algorithm due to the fact that using the learning process over iterations, the algorithm adapts its operators to perform the best for each problem.

Moreover, we consider a proper framework to maintain a decent equilibrium between exploration and exploitation of the feasible region over generations of the algorithm to avoid local optima. We show the efficiency of our algorithm on the most complex benchmarks in the literature. Besides, we apply the suggested algorithm to predict the COVID-19 pandemic

Table 1 Classification of the metaheuristics

EAs	PAs	SAs	Other algorithms	MLAs
Genetic algorithms (GA) (Holland 1992)	Small-world optimization algorithm (SWOA) (Du et al. 2006)	Particle swarm optimization (PSO) (Eberhart and Kennedy 1995)	Stochastic fractal search (SFS) (Salimi 2015)	Hybrid Q-learning based algorithm (this paper)
Genetic programming (GP) (Koza and Koza 1992)	Curved space optimization (CSO) (Moghaddam et al. 2012).	Grasshopper optimization algorithm (Saremi et al. 2017)	Sine–cosine algorithm (SCA) (Mirjalili 2016a, b)	
Degree-descending search strategy (DDS) (Cui et al. 2018)	Charged system search (CSS) (Kaveh and Talatahari 2010)	Ant lion optimization algorithm (ALO) (Mirjalili 2015a)	Water cycle algorithm (WCA) (Eskandar et al. 2012)	
Biogeography based optimizer (BBO) (Simon 2008)	Multi-verse optimization (MVO) algorithm (Mirjalili et al. 2016a, b)	Crow search algorithm (CSA) (Askarzadeh 2016)	Virus colony search (Li et al. 2016)	
Differential evolution (DE) (Price 2013)	Black hole mechanics optimization (BHMO) (Kaveh et al. 2020)	Salp swarm algorithm (SSA) (Mirjalili et al. 2017)	Gradient-based optimizer (GBO) (hmadianfar et al. 2020)	
Estimation of distribution algorithm (EDA) (Wang et al. 2013)	Galaxy-based search algorithm (GBSA) (Kaveh and Dadras 2017)	Grey Wolf optimizer (GWO) (Mirjalili et al. 2014)	Lightning search algorithm (LSA) (Shareef et al. 2015)	
Evolution strategy (ES) (Rechenberg 1978)	Simulated annealing (SA) (Kirkpatrick et al. 1983; CERBY 1985)	Dragonfly algorithm (DA) (Mirjalili 2016a, b)	Coronavirus optimization algorithm (COA) (Martínez-Álvarez et al. 2020)	

Table 1 continued

EAs	PAs	SAs	Other algorithms	MLAs
Evolutionary programming (EP) (Fogel et al. 1966)	Gravitational search algorithm (GSA) (Rashedi et al. 2009)	Cuckoo search (CS) (Yang and Deb 2009)	Sine–cosine Crow search algorithm (SCCSA) (Khalilpourazari and Pasandideh 2019)	
	Central force optimization (CFO) (Formato 2007)	Whale optimization algorithm (WOA) (Mirjalili and Lewis 2016)	Water cycle Moth Flame optimization (WCMFO) (Khalilpourazari and Khalilpourazary 2019)	
	Black hole (BH) algorithm (Hatamlou 2013)	Artificial bee colony (ABC) (Karaboga and Basturk 2007)		
	Thermal exchange optimization (Kaveh and Dadras 2017)	Moth-flame optimization (MFO) (Mirjalili 2015a, b)		
		Dynamic Virtual Bats Algorithm (DVB) (Topal and Altun 2016)		

in Quebec, Canada. Our outcomes express that the designed algorithm robustly predicts the future trends of the pandemic.

3 Algorithm development

Metaheuristics work based on randomization. By randomization, we mean that these algorithms use random stepsizes while updating each particle's position in the solution space. Metaheuristics use unique operators and strategies to update the position of each particle (solution). The efficiency of these operators and algorithms significantly depends on the solution space of the problem. For instance, some algorithms follow a direct updating procedure, such as water cycle algorithm (WCA), while some other encircle the best solution to update the position of a given particle, such as Grey–Wolf optimizer (GWO). Each of these operators and moving strategies has unique advantages that enable the algorithms to perform well in optimizing specific problems. Therefore, developing new algorithms that could efficiently solve a higher number of optimization problems is essential.

In this study, we use several operators (moving strategies) from various algorithms to update the particles' position in the solution space. For updating each particle, we have to choose an operator from the given set of operators. Determining the best strategy and the most efficient operator for any given optimization problem is computationally challenging. Therefore, we use a reinforcement learning method that learns and optimizes the choice of operators during the optimization process to achieve optimal performance. In the following, we first describe all the features of the offered algorithm, and then we define the algorithm in a unique structure.

3.1 Q-learning

Reinforcement learning (RL) that approximates dynamic programming, and neuro-dynamic programming, is a type of machine learning which determines the best actions in a specific environment to maximize a reward (Bertsekas 2019). One of the main features of reinforcement learning is that the agent receives a reward or punishment after executing an action. The RL continues to interact with the environment to achieve the optimal policy via trial and error.

The Q-Learning is one of the most efficient reinforcement learning algorithms that determine an optimal policy by evaluating taken actions using the environment. Q-learning is a way to optimize solutions in a Markov decision process (MDP) problem (Akhtar and Farrukh 2017). The Q-learning algorithm aims to maximize the anticipated reward by determining the optimal state-action pairs. The algorithm uses a $Q(s, a)$ table where s_t is the state and a_t is the action at time step t, and Q is the cumulative reward matrix. The algorithm updates the components of the Q-table $Q(s, a)$ iteratively using Eq. (1).

$$Q_{(t+1)}(s_t, a_t) = Q_t(s_t, a_t) + \epsilon_t(r_t + \gamma max(Q_t(s_{t+1}, a_{t+1})) - Q_t(s_t, a_t)), \tag{1}$$

In Eq. (1), ϵ_t denotes the learning rate parameter and r_t is the obtained reward/punishment from the current action. Besides, the expression γ is the scaling factor. One of the main challenges in designing an efficient Q-learning algorithm is in determining the importance of the information gained throughout interactions with the environment. For instance, assigning a value close to 1 to the learning rate parameter means that we consider higher importance for the recent information gained. To optimize the value of the learning rate parameter throughout iterations, we use an adaptive methodology that uses Eq. (2) to intelligently tune the learning rate parameter to explore and exploit the search region (Zamli et al. 2018).

$$\epsilon_t = 1 - 0.9\frac{t}{MT}, \tag{2}$$

In Eq. (2), t is the iteration index, and MT is the maximum generation. In this research, we consider a reward value of 1 if the current action improves the solution quality of the particle; otherwise, we consider a punishment value of − 1. Based on the given illustrations, we present the Pseudo-code of the Q-learning in Algorithm 1.

Algorithm 1: Pseudo Code for the Q-Learning Algorithm

```
1: input states, actions, gamma, and initial Q(s, a) table;
2: randomly select an initial state;
3: while stopping criteria not met do
4:     select the best action from the Q-table;
5:     execute the action and get a reward/punishment;
6:     determine the max Q-value for the next state;
7:     update a(t);
8:     update the Q-table;
9:     update the current state, s(t) = s(t + 1);
10: end
11: return the Q-table;
```

In this research, we consider several efficient operators from different algorithms and let the Q-learning algorithm determine the best action throughout the optimization process to modify the location of each particle in the feasible space. In the following subsections, we present the operators in detail.

3.2 Grey–Wolf optimizer

Similar to other swarm intelligence-based systems, GWO initially generates a set of primary solutions. Then, it sorts the solutions regarding their fitness and considers the three best solutions as the dominant wolves (alpha, beta, and delta). The following equations imitate the encircling behavior of the grey wolves around prey:

$$\vec{D} = \left|\vec{C}\vec{X}_p(t) - \vec{X}(t)\right|, \tag{3}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A}\vec{D}, \tag{4}$$

In Eqs. (3) and (4), t represents the iteration index and $\vec{A}$ and $\vec{C}$ characterize location vectors of target and other grey wolves. $\vec{X}_p(t)$ and $\vec{X}(t)$ are the position of the prey and grey wolf, respectively. These coefficients are calculated as follows:

$$\vec{A} = 2a_1\vec{r}_1 - a_1 \tag{5}$$

$$\vec{C} = 2\vec{r}_2, \tag{6}$$

In Eqs. (5) and (6), a_1 decreases over iterations from 2 to 0 and $\vec{r}_1$ and $\vec{r}_2$ are random numbers. After encircling the prey, the wolves start the hunting process. To mathematically

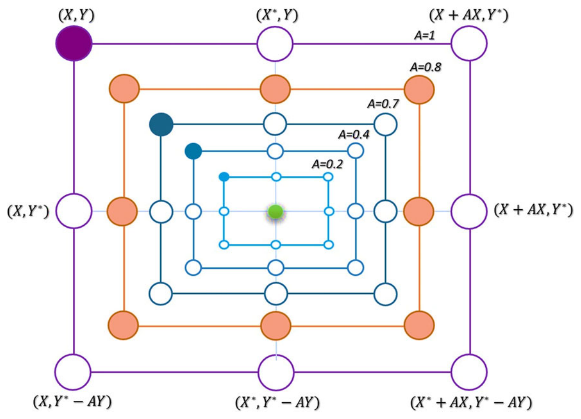


Fig. 1 Hunting behavior in GWO

express the movements of grey wolves in the hunting process, we consider that the alpha, beta, and delta have superior knowledge of the probable position of the target (possible optimal solution of the problem). In this framework, the following formulas are recommended to mimic the hunting process (Khalilpourazari et al. 2020a):

$$\vec{D}_\alpha = \left|\vec{C}_1\vec{X}_\alpha - \vec{X}\right|,\ \vec{D}_\beta = \left|\vec{C}_2\vec{X}_\beta - \vec{X}\right|,\ \vec{D}_\delta = \left|\vec{C}_3\vec{X}_\delta - \vec{X}\right|, \tag{7}$$

$$\bar{X}_1 = \vec{X}_\alpha - \vec{A}_1\vec{D}_\alpha,\quad \bar{X}_2 = \vec{X}_\beta - \vec{A}_2\vec{D}_\beta,\quad \bar{X}_3 = \vec{X}_\delta - \vec{A}_3\vec{D}_\delta, \tag{8}$$

$$\bar{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{9}$$

Figure 1 depicts a graphical interpretation of the hunting action in 2D space.

3.3 Sine–cosine algorithm

SCA is a newly developed search procedure that mimics the sine and cosine like movements in the feasible space to modify the elements using Eq. (10):

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times \sin(r_2) \times \left|r_3 P_i^t - X_i^t\right|, r_4 < 0.5 \\ X_i^t + r_1 \times \cos(r_2) \times \left|r_3 P_i^t - X_i^t\right|, r_4 \geq 0.5 \end{cases} \tag{10}$$

In Eq. (10), X_i^t is the present location $P_i^t$is the location of the best particle and r_1, r_2, r_3 are random numbers in (0,1]. Throughout iterations, r_1 displays the movement path, r_2 controls the moving distance, r_3 guarantees a suitable equilibrium among underline or deemphasize the desalination, and r_4 selects a sine or cosine measure for updating procedure. Figure 2 shows the movement behavior of the sine and cosine actions.

SCA uses sine and cosine movements intelligently to evade local optima. In addition to adjusting the particles' movements during the solution process, SCA reduces the value of r_1 parameter using the below formula to sustain a proper equilibrium between exploration and exploitation as follows:

$$r_1 = a_2 - t\frac{a_2}{T} \tag{11}$$

In Eq. (11), t displays present repetition,a_2 is a constant, and $T$is the maximum generation.

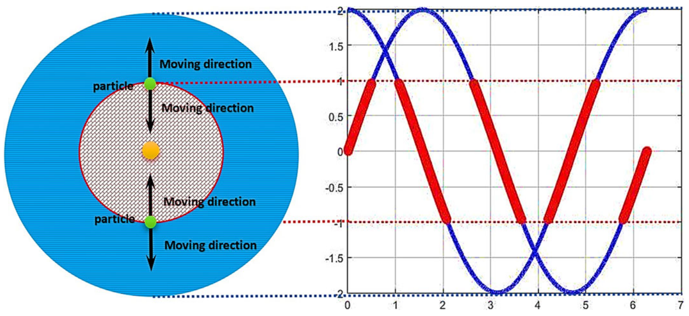


Fig. 2 Updating procedure in SCA

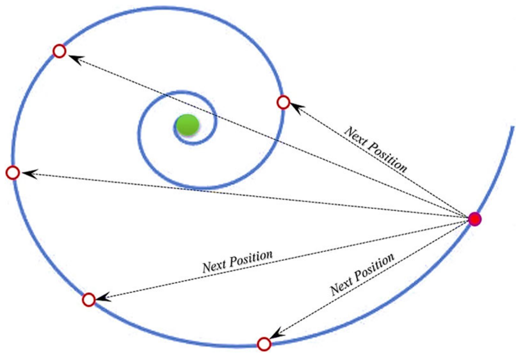


Fig. 3 The spiral fly path of the moths around the flame

3.4 Moth-flame optimization

Like most of the optimization paradigms, MFO initiates the optimization by creating random solutions. Then, it mimics the spiral flying actions of the moths around light sources using a logarithmic spiral function as follows:

$$IP_i^{X+1} = \left|F_i - IP_i^X\right| e^{bt} \cos(2\pi t) + F_i, \tag{12}$$

In Eq. (12), t is a constant in [− 1, 1], and b is a constant for determining the form of the logarithmic spiral. F_i is the flame (the best solution), IP_i^X is the moth, and $\left|F_i - IP_i^X\right|$ calculates the distance between the moth and flame. Figure 3 shows the spiral movement around the flame.

In order to maintain a suitable balance among its exploration and exploitation, the MFO algorithm reduces the search radius using the following equations (Khalilpourazari et al. 2019; Mohammadi and Khalilpourazari 2017):

$$a_3 = -1 + t\left(\frac{-1}{MT}\right) \tag{13}$$

$$tt = (a_3 - 1) \times rand + 1 \tag{14}$$

In Eqs. (13) and (14), t displays present repetition, and MT is the maximum generation.

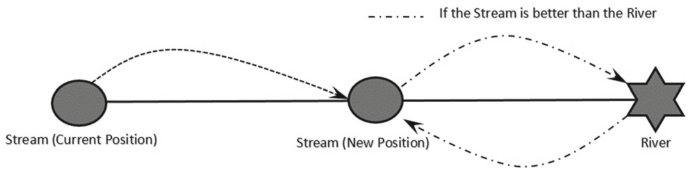

If the Stream is better than the River
Stream (Current Position)
Stream (New Position)
River

Fig. 4 Updating procedure in WCA

3.5 Particle swarm optimization

Particle swarm optimization (PSO) is one of the most efficient procedures for optimization, and it performs promising in solving many complex problems. PSO is a population-based algorithm that uses Eq. (15) to update the location of a given particle in the solution space:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{15}$$

In Eq. (15), $x_i(t)$ presents the current location of the particle and $v_i(t+1)$ determines the velocity of the particle. v_i vector is the main component of the updating operator that is calculated as follows:

$$v_i(t+1) = \omega v_i(t) + C_1 r_1 (p_i(t) - x_i(t)) + C_2 r_2 (G(t) - x_i(t)) \tag{16}$$

In Eq. (16), r_1, r_2 are random numbers in (0, 1], C_1, C_2 are coefficients, $p_i(t)$ is the best solution found by the same solution so far, and $G(t)$ is the best solution attained so far.

3.6 Water cycle algorithm

Water cycle algorithm (WCA) is one of the best algorithms for solving complex problems. WCA is a population-based nature-inspired metaheuristic that mimics the flow of streams to rivers and sea to perform optimization.

$$x^{i+1}_{current} = x^{i}_{current} + C\left(x^{i}_{best\_sol} - x^{i}_{current}\right) \tag{17}$$

In Eq. (17), C is a random value. We used the updating operator on the WCA as one of the means to update the location of a given particle in the feasible space. Figure 4 represents the updating procedure in WCA.

3.7 Gaussian walks and Lévy flight

In this subsection, we use the leading operators of the stochastic fractal search (SFS) offered by Salimi (2015). SFS utilizes an important scientific property called "fractal". Fractals are complicated geometric shapes that generally have a "fractional dimension," resulting in self-similarity. SFS follows diffusion limited aggregation (DLA), which is an efficient technique to create fractals. To simulate the DLA, we use the Gaussian walk and Lévy flight. Figure 5 presents a fractal shape produced through the DLA scheme.

We use the following equation to simulate the diffusion process in the DLA method.

$$x_i^q = x_i + \beta \times Gaussian(|BP|, \sigma) - \left(\varepsilon \times BP - \varepsilon' \times x_i\right), \tag{18}$$

In Eq. (18), q describes the number of new solutions created through the diffusion of each particle, σ is the standard deviation of the Gaussian walk, and BP is the best solution. Also,

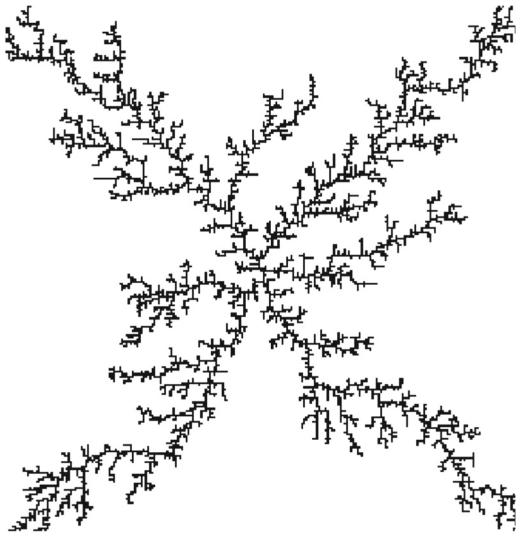


Fig. 5 A fractal produced through the DLA method

x_i^q denotes new particles produced via diffusion process and x_i is the ith solution. Besides, ε' and ε are randomly generated numbers in (0, 1]. Element σ is defined using the below formula (Khalilpourazari et al. 2020b):

$$\sigma = \left| \frac{\log(g)}{g} \times (P_i - BP) \right|, \tag{19}$$

where $\frac{\log(g)}{g}$ decreases the length of Gaussian jumps over iterations. Element g is the iteration number and P_i is the current position of the particle.

In order to enhance exploration and randomness in the population, we also apply a Lévy flight based updating procedure to the particle under consideration as follows:

$$X_{new}^i = X_c^i + X_c^i \otimes Levy(D) \quad for \;\; i = 1, \ldots, m \tag{20}$$

The expression X_{new}^i is the new location and X_c^i is the current location of the particle, respectively. We calculate the Lévy flight using Eq. (13).

$$Levy(x) = \frac{0.01 \times \sigma \times r_1}{|r_2|^{\frac{1}{\beta}}}, \tag{21}$$

where $r_1$and r_2 are random numbers. In Eq. (13) σ is obtained as follows:

$$\sigma = \left(\frac{\Gamma(1+\beta)\sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right)\beta 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}}. \tag{22}$$

3.8 The developed hybrid Q-learning based algorithm

In this section, we propose a novel Hybrid Q-learning based algorithm (HQLA) to solve complicated optimization problems. The idea behind this algorithm is to design a solution methodology that is capable of solving complicated problems by adapting its operators to any solution space. We designed an optimization procedure that can benefit from the advantages of several algorithms. The process can use any movement strategy based on each updating operator. However, a reinforcement learning based algorithm (Q-learning) selects the best action in each iteration for each particle. When optimization begins, the algorithm performs several random actions to evaluate the efficiency of each type of operator. As the iterations continue, Q-learning learns how to employ different actions to achieve the best possible solution. For each action, we consider a reward equal to 1 if the current operators improve the solution quality; Otherwise, the algorithm assigns a punishment value of − 1 to the action. The Pseudo-code of the algorithm is available in Algorithm 2.

Algorithm 2: Pseudo Code for the HQLA

```
1.  input parameters of the algorithm;
2.  create a set of randomly generated solutions;
3.  while stopping criteria not met
4.    check for infeasibility of the particles;
5.    bring infeasible particles to the feasible solution space;
6.    sort the solutions based on their fitness value;
7.    for each particle
8.      if it is the first iteration
9.        select a random action;
10.     else
11.       select the action using the Q-table;
12.     end
13.     if the action is GWO
14.       use GWO operators to update the position of the particle;
15.     else if the action is SFS
16.       use SFS operators to update the position of the particle;
17.     else if the action is WCA
18.       use WCA operators to update the position of the particle;
19.     else if the action is PSO
20.       use PSO operators to update the position of the particle;
21.     else if the action is MFO
22.       use MFO operators to update the position of the particle;
23.     else if the action is SCA
24.       use SCA operators to update the position of the particle;
25.     end
26.   check for infeasibility of the particle;
27.   bring infeasible particle to the feasible solution space;
28.   calculate the objective function value of the particle;
29.   determine the reward/punishment value;
30.   determine the max Q-value for the next state;
31.   update a(t);
32.   update the Q-table;
33.   update the current state, s(t) = s(t + 1);
34.   end
35. end
36. return the best solution
```

4 Results and discussion

Metaheuristics are approximation algorithms that are based on randomized movements. Therefore, their performance may differ from one problem to another. Thus, to show the efficiency of a metaheuristic algorithm, we should apply them to many benchmark functions. For this purpose, we use 29 benchmarks, including Unimodal, multimodal, fixed dimensional multimodal, and hybrid composite functions that are among the most complex benchmarks in the literature. For more details regarding these benchmarks, see "Appendix 1".

We compare our algorithm to state-of-the-art methods, including Crow Search Algorithm (CSA), Artificial Bee Colony (ABC), Cuckoo Search (CS), Genetic Algorithm (GA), Moth-Flame optimization (MFO), Gravitational Search Algorithm (GSA), and Dragonfly Algorithm (DA). In order to solve the benchmark functions, we considered 15,000 Number of Function Evaluations (NFEs) to perform a reasonable assessment. Besides, to draw a reliable conclusion, we apply each algorithm on each benchmark 30 times, and report mean, standard deviation, worst and best results. Table 2 provides more details about the values of the main parameters of the algorithms.

In the first step, we assess the performance of the HQLA on unimodal benchmarks (F1–F7). Figure 6 depicts a 2D representation of these benchmark functions.

Unimodal benchmarks do not have several local optima and are considered to assess the exploration ability of metaheuristic algorithms. This set of test suites are difficult to solve since the algorithms should first locate the global optima approximately and then perform exploitation to provide the best approximation of the location of the global optima. Table 3 provides detailed information on the performance of the algorithms in unimodal benchmarks. Based on the results of Table 3, we observe that the HQLA performs the best in most of the benchmarks. In F1–F5 and F7, the HQLA outperforms all other algorithms by obtaining the best possible solution for all benchmarks. In F6, HQLA ranks third in providing the best solution for this benchmark. Besides, HQLA provides the lowest standard deviation that shows very low variability in the performance of this algorithm. Moreover, considering the boxplot of the results of Table 3 present in Fig. 7a–c, it becomes apparent that HQLA has the lowest and narrowest boxplot among the algorithms.

Furthermore, Fig. 8a–c, that present the convergence plots of the methods, disclose that the HQLA can maintain a perfect balance among exploration and exploitation of the solution space. In Fig. 8a–c, we observe that the HQLA can continually improve the best solution attained in each iteration by choosing the best operator to explore the solution space. The learning process in HQLA enables the algorithm to determine the best operator to change the location of the particles in the solution region by evaluating the efficiency of each updating mechanism. Based on these observations, we conclude that HQLA is a reliable technique for this family of benchmark functions.

The second and third family of the benchmarks are multimodal and fixed-dimension multimodal benchmarks. These benchmarks contain several local optima that make the solution process a complicated task. To perform well in solving these benchmark functions, the algorithms should maintain an excellent balance among exploration and exploitation. This will help the algorithms avoid local optimum. Figures 9 and 10 shows a schematic view of these benchmark functions in 2D.

We provide detailed information on the performance of the algorithms on solving multimodal and fixed-dimension multimodal benchmarks in Tables 4 and 5.

Based on the results, we observe that in F8–F12, F14, and F16–F23 (14 out of 16) benchmark functions, the HQLA outperforms other algorithms considering average, standard deviation, best and worst values over 30 repetitions. In some of these benchmarks, such as F9 and F11, the standard deviation of the results provided by HQLA is zero. In these benchmark functions, the HQLA achieves global optima in all the repetitions. These results indicate that HQLA is a robust and reliable algorithm in solving complex optimization problems. HQLA is able to choose between several operators that enable the algorithms to explore

Table 2 The values of the parameters of the algorithms

Algorithm	Parameter	Value	Algorithm	Parameter	Value
HQLA	Number of initial solutions	30	GA	Cross over probability	0.9
	ω	Decreases linearly from 0.9 to 0.4		Mutation probability	0.005
	C_2	2		Number of initial solutions	30
	C_1	2	GSA	Number of initial solutions	30
	a_1	Decreases linearly from 2 to 0		G_0	1
	a_2	Decreases linearly from 2 to 0		α	20
	a_3	Decreases linearly from -1 to -2	MFO	a_3	Decreases linearly from -1 to -2
CSA	Number of initial solutions	30		Number of initial solutions	30
DA	Number of initial solutions	30	CS	Discovery rate of alien solutions	0.25
ABC	Number of initial solutions	30			

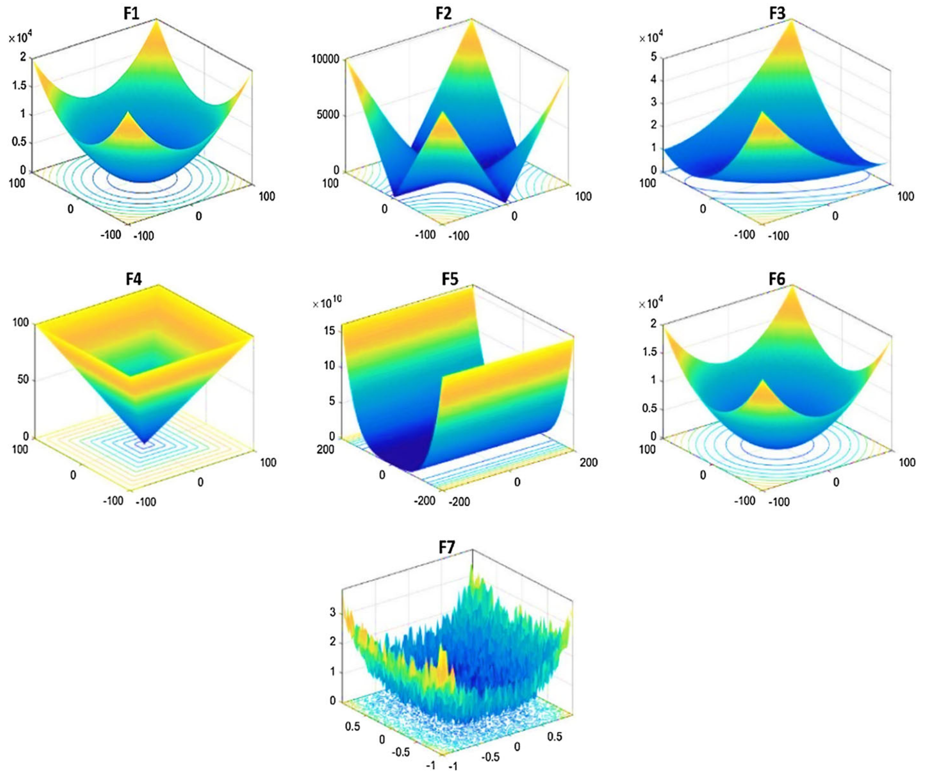


Fig. 6 2D representation of F1–F7

and exploit the resolution region intelligently. The learning process in the HQLA helps the algorithm evaluate the efficiency of the operators and discover the most efficient operator (action) for any problem. Besides, Fig. 8a–c depict the perfect balance among exploration and exploitation in the performance of the HQLA throughout iterations. Moreover, in F13 and F15, the HQLA ranked second among all algorithms, which shows its high capability in solving optimization problems. Furthermore, Fig. 7a–c, disclose that the boxplot of the outcomes of the HQLA is narrower and lower than any other algorithm that highlights the superiority of the proposed methodology over existing approaches.

The last family of the benchmark functions is hybrid composite benchmarks that are the most challenging (Liang et al. 2005). Figure 11 presents a graphical representation of these benchmark functions in 2D.

Computational results of solving this family of benchmark functions using each algorithm are available in Table 6. Considering the results, we observe that the HQLA provides the best results and outperforms other algorithms in solving hybrid benchmarks. Table 6 shows that the HQLA obtained the lowest average, standard deviation, and best values for these benchmark functions while maintaining the lowest worst value. Besides, Fig. 13 approves this statement by showing that the boxplot of the HQLA is narrower and lower than any other algorithms. Moreover, Fig. 12 shows that the HQLA adjusts exploration and exploitation by intelligently choosing the best operators in each iteration. Based on Fig. 12, we observe that the HQLA can improve the best solution consistently throughout iterations.

Table 3 Computational outcomes of the algorithms in solving unimodal benchmark functions

		HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
F1	Average	**2.14E−66**	2.25E−05	6.10E−02	6.02E+00	3.68E−05	6.34E−14	2.53E−18	1.21E−02
	Std Dev	**6.92E−66**	2.78E−05	0.056501	16.50088	1.82E−05	1.1E−13	9.7E−19	0.009841
	Worst	**3.20E−65**	1.23E−04	2.37E−01	8.36E+01	9.72E−05	5.61E−13	5.91E−18	3.37E−02
	Best	**8.88E−105**	1.98E−06	5.96E−03	0.00E+00	1.63E−05	6.96E−16	1.35E−18	2.61E−04
F2	Average	**6.95E−37**	3.81E−03	7.54E−02	1.69E+00	1.16E−02	6.67E−01	4.93E−09	1.58E−02
	Std Dev	**2.5E−36**	0.002926	0.035631	2.274098	0.002961	2.494438	1.12E−09	0.009469
	Worst	**1.31E−35**	1.37E−02	1.69E−01	1.09E+01	1.66E−02	1.00E+01	7.85E−09	3.81E−02
	Best	**4.31E−57**	6.10E−04	2.15E−02	1.33E−01	6.33E−03	2.78E−10	3.21E−09	1.99E−03
F3	Average	**5.39E−37**	1.68E−02	1.62E+03	1.96E+02	7.00E−02	1.67E+02	3.31E+00	6.13E+01
	Std Dev	**2.24E−36**	0.025618	396.8719	580.7061	0.025593	897.525	3.383667	27.55541
	Worst	**1.25E−35**	1.34E−01	2.51E+03	3.20E+03	1.42E−01	5.00E+03	1.31E+01	1.32E+02
	Best	**1.61E−44**	7.80E−04	7.52E+02	4.76E−02	3.33E−02	2.05E−05	6.14E−02	1.72E+01
F4	Average	**2.64E−25**	1.08E−02	1.76E+01	1.30E+00	2.59E−01	3.76E−02	1.23E−09	6.21E−01
	Std Dev	**6.16E−25**	0.008445	4.709691	1.290182	0.068563	0.137389	2.11E−10	0.162887
	Worst	**2.98E−24**	3.46E−02	2.44E+01	5.21E+00	4.23E−01	7.47E−01	1.76E−09	9.66E−01

Table 3 continued

		HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
	Best	**1.76E−28**	1.98E−03	8.29E+00	0.00E+00	1.19E−01	1.45E−04	8.47E−10	2.86E−01
F5	Average	**5.34E+00**	6.45E+00	8.06E+01	3.66E+03	5.92E+00	3.18E+03	6.91E+00	3.65E+01
	Std Dev	**0.227996**	2.09398	33.02036	16096.55	2.065037	16131.83	0.19993	38.87984
	Worst	**5.93E+00**	9.38E+00	1.58E+02	9.01E+04	1.00E+01	9.00E+04	7.55E+00	1.33E+02
	Best	**4.88E+00**	3.55E−01	1.74E+01	6.94E+00	3.03E+00	1.16E+00	6.46E+00	2.68E+00
F6	Average	5.36E−06	1.56E−05	5.46E−02	5.47E+00	3.06E−05	7.57E−14	**2.60E−18**	1.40E−02
	Std Dev	1.66E−06	1.16E−05	0.043146	25.99325	1.65E−05	1.51E−13	**8.35E−19**	0.009801
	Worst	1.01E−05	4.10E−05	1.41E−01	1.45E+02	6.71E−05	7.85E−13	**5.09E−18**	3.67E−02
	Best	2.42E−06	1.09E−06	7.47E−03	2.22E−06	7.31E−06	1.91E−16	**1.33E−18**	6.39E−04
F7	Average	**2.87E−04**	2.72E−03	9.01E−02	1.74E−02	1.06E−02	5.82E−03	5.63E−03	2.88E−03
	Std Dev	**0.000182**	0.001551	0.036762	0.013218	0.00359	0.003094	0.002614	0.001706
	Worst	**7.50E−04**	6.82E−03	1.61E−01	5.44E−02	1.72E−02	1.60E−02	1.18E−02	7.37E−03
	Best	**2.15E−05**	8.77E−04	3.30E−02	2.43E−03	3.95E−03	2.02E−03	8.37E−04	6.79E−04

(a)

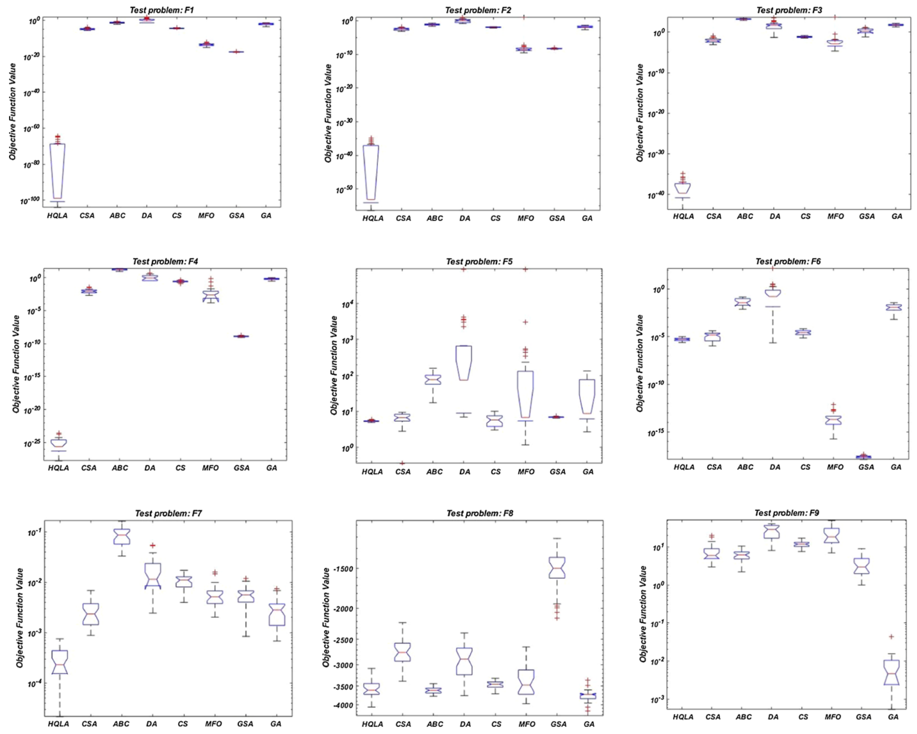
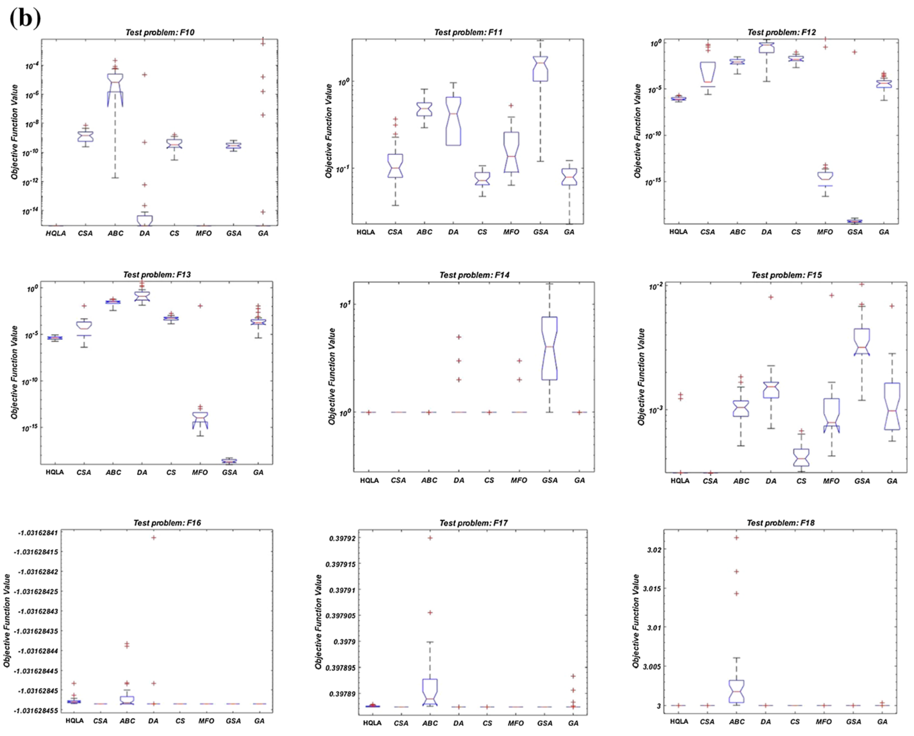


Fig. 7 **a** Boxplot of the results in F1–F9 benchmarks. **b** Boxplot of the results in F10–F18 benchmarks. **c** Boxplot of the results in F19–F23 benchmarks

Although we showed that the HQLA outperforms other state-of-the-art methods in terms of solution performance, we apply Friedman's test, which is a powerful statistical test, to show the statistical superiority of our algorithm to other methods. Table 7 presents the results of Friedman's test. Based on the outcomes, the average rank of the HQLA is 3.036207, which is far smaller than that of other algorithms. Therefore, from a statistical point of view, HQLA performs significantly better than all other algorithms. We note that we perform the Friedman's test at a 95 percent confidence level.

5 Multi-criteria parameter estimation and curve fitting

Quebec is one of Canada's provinces that is dealing with the COVID-19 epidemic triggered by the SARS-CoV-2 virus. The province has been reported the most COVI-19 cases that account for more than 63,713 confirmed cases of COVID-19 and 5770 death cases by the disease. On April 29, 2020, Quebec hospitals announced that the healthcare system capacity could not respond to the influx of the COVID-19 patients to the hospitals (Weeks and Tu Thanh 2020; The Globe and Mail n.d.). On June 28, 2020, Montreal's Emergency Rooms (ERs) reported near capacity status due to limited resources (Fahmy and Ross 2020; CTV News. n.d.). These highlight the need for a methodology that could accurately predict the future trend of the pandemic in the province that enables the policymakers to optimize the



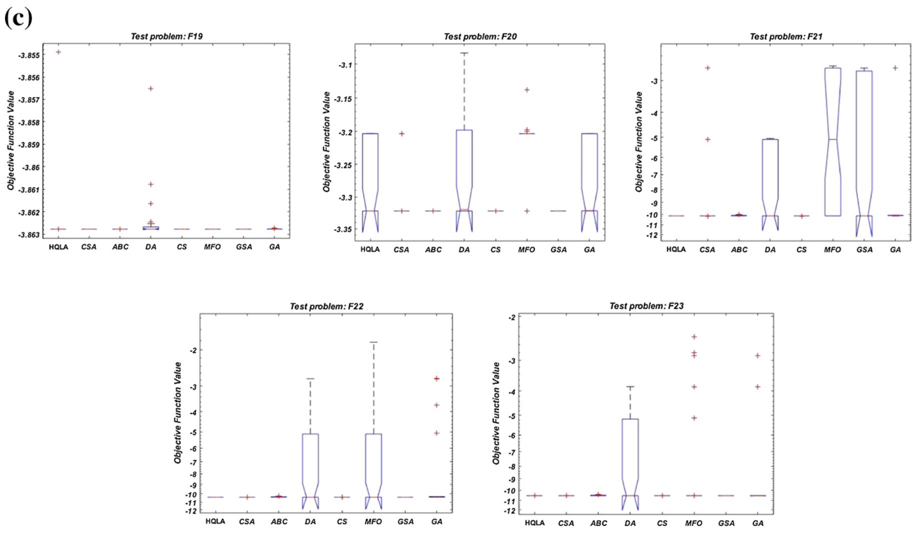


Fig. 7 continued

resource allocation to avoid loss of lives, as many scientists declared that resource shortages such as ventilator shortages are the difference between life and death for patients (Kliff et al. 020; The New York Time. n.d.). Developing new methodologies to predict pandemic growth

(a)

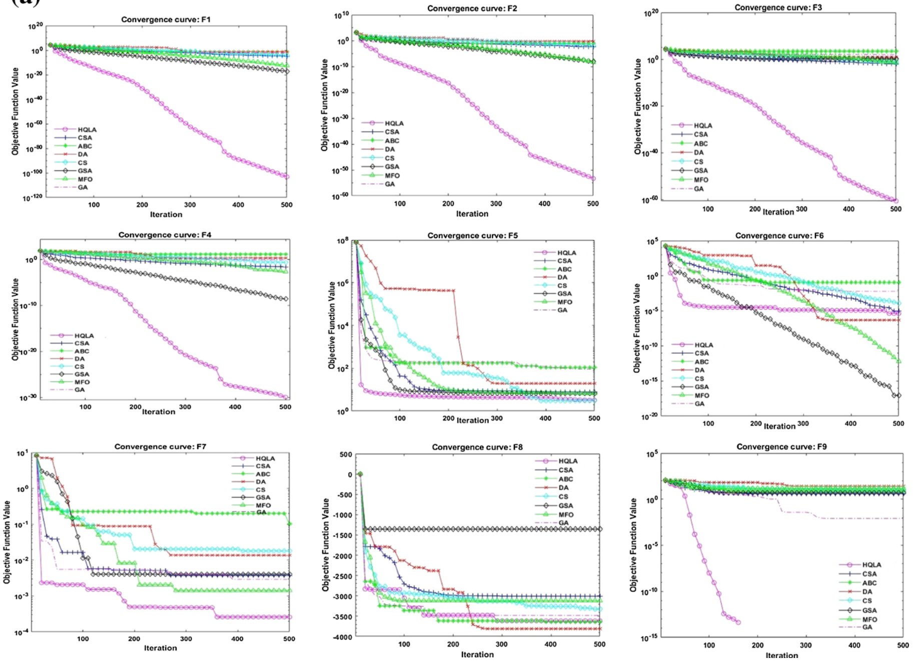


Fig. 8 **a** Convergence plot of the algorithms. **b** Convergence plot of the algorithms. **c** Convergence plot of the algorithms

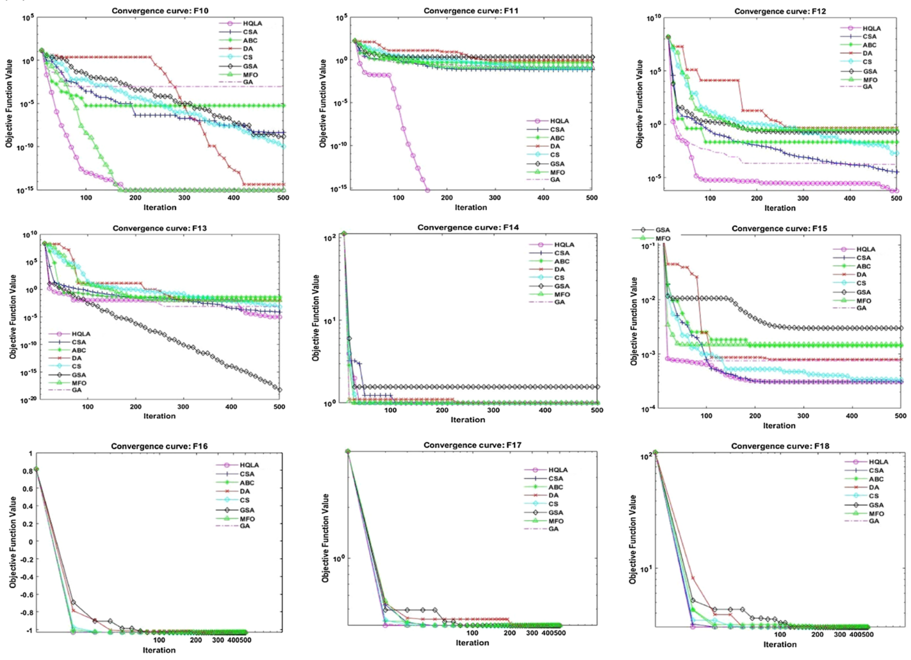
is essential to optimize resource allocation and determine the optimal time to implement lockdown measures. On the other hand, we could optimize resource allocation for life-threatening cases admitted to ICUs and reduce the disease's fatality rate.

In this section, we use the most recent and accurate model called SIDARTHE, presented by Giordano et al. (2020) published in *Nature Medicine*. The scientists showed that using the model, we could predict the future trend of the pandemic accurately. However, the solution to the problem is a cumbersome task. For more information on the SIDARTHE model, see "Appendix 2". In this research, we used the SIDARTHE model and applied it to real data from Quebec, Canada. In order to solve the model, we utilize HQLA. Figure 14 shows the convergence plot of HQLA throughout iterations. We divided the period (from January 25, 2020 (day 1) to July 19, 2020 (day 176)) into six stages as follows in which the Quebec government applied specific restrictions to control the pandemic:

- Stage 1 (from January 25, 2020, to March 15, 2020)
- Stage 2 (from March 15 to March 24)
- Stage 3 (from March 24 to March 28)
- Stage 4 (from March 28 to April 2)
- Stage 5 (from April 2 to April 13)
- Stage 6 (after April 13)

We note that we considered the sum of mean square errors as the objective function value for our model, and its optimal value for our case study is 6.29E−06. Based on the outcomes, we observe that the HQLA can solve the problem very efficiently. Table 8 shows detailed statistics about the optimized factors of the model. Now that the results of Table 8 are the

(b)



(c)

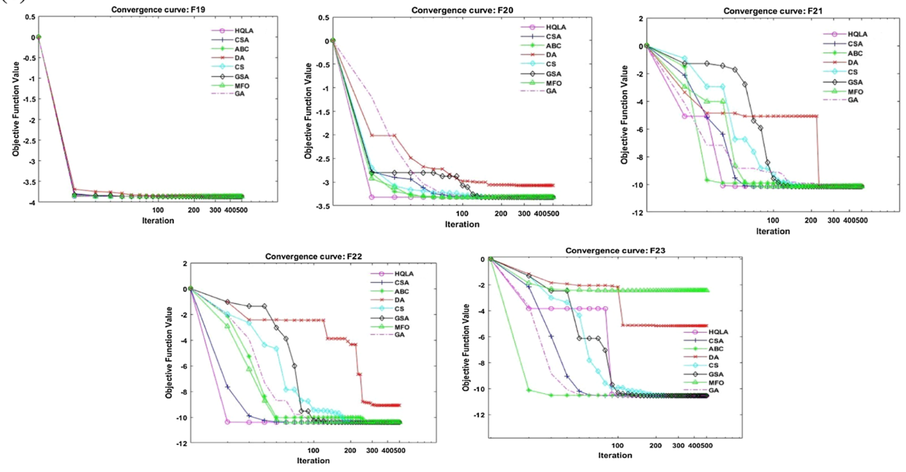


Fig. 8 continued

output of the optimization process and solving the SIDARTHE model for Quebec data using HQLA.

In the following, we compared our results with actual data from Quebec to validate our results. Figure 15 shows the predicted and actual data from Quebec. In Fig. 15, we accurately predict the number of infected cases, recovered cases, and cumulative diagnosed cases. Based on the outcomes, we conclude that the HQLA is an efficient solution methodology for the problem.

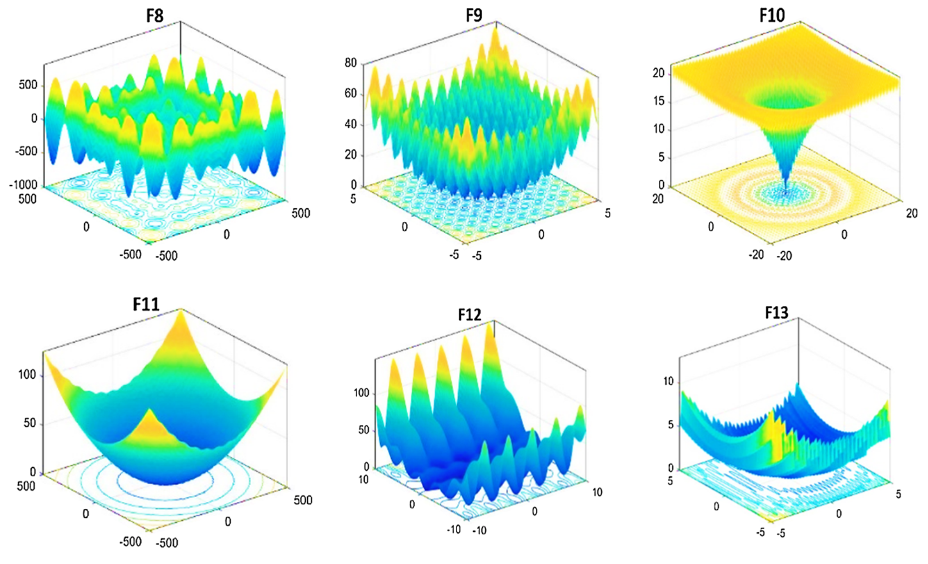


Fig. 9 2D representation of F8–F13

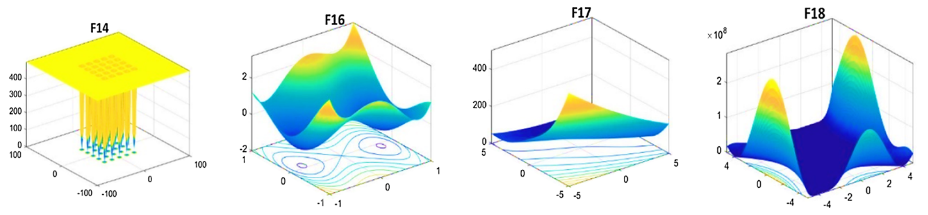


Fig. 10 2D representation of F14–F18

As mentioned earlier, we separated the planning horizon into six phases in which the Quebec government announced detailed limitations to control the epidemic. The first case of COVID-19 was detected in Quebec, Canada, on February 27, 2020 (Lapierre 2020). In the first phase, the transmission rate was considered low. Based on our results, we observe that the transmission rates were low at the first stage, and the reproduction rate was $R_0 = 1.0998$. Quebec province first announced a state of emergency on March 12, 2020. We consider March 15, 2020, to March 24, 2020, as the second phase of the pandemic due to a drastic increase in the number of cases. On March 15, the Quebec government ordered the closure of all recreational and entertainment facilities (Gouvernement Du Québec 2020).

Following the quick progress in the number of infected cases, on March 27, Montreal declared a local state of emergency, and the Quebec government ordered the closure of all universities and schools. Besides, On March 20, the province banned indoor gatherings. In the second phase of the pandemic, our study estimates a reproduction rate of $R_0 = 3.8028$ for Quebec province. From March 24, 2020, to March 28, 2020, Quebec received more COVID-19 test kits enabling the healthcare authorities to perform more tests and determine the infected cases. In this phase, we approximate the reproduction rate equal to $R_0 = 4.6096$.

From March 28, 2020, to April 2, 2020, strict actions were taken by the government decreased the reproduction rate to $R_0 = 1.1193$. The reproduction rate dropped over the next period to $R_0 = 1.0248$ from April 2, 2020, to April 13, 2020. After April 13, 2020,

Table 4 Computational outcomes of the algorithms in solving multimodal benchmark functions

		HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
F8	Average	**− 3.59E+03**	− 2.74E+03	− 3.61E+03	− 2.98E+03	− 3.46E+03	− 3.42E+03	− 1.56E+03	− 3.75E+03
	Std Dev	**214.4446**	283.4627	90.24338	376.9813	98.34133	375.2893	244.7475	155.5952
	Worst	**− 3.08E+03**	− 2.22E+03	− 3.44E+03	− 2.39E+03	− 3.31E+03	− 2.64E+03	− 1.21E+03	− 3.36E+03
	Best	**− 4.07E+03**	− 3.38E+03	− 3.77E+03	− 3.75E+03	− 3.70E+03	− 3.97E+03	− 2.15E+03	− 4.19E+03
F9	Average	**0.00E+00**	7.69E+00	6.16E+00	2.64E+01	1.18E+01	2.18E+01	3.78E+00	7.36E−03
	Std Dev	**0**	4.149784	1.863144	10.72399	2.272461	11.97422	2.06159	0.007922
	Worst	**0.00E+00**	1.99E+01	1.05E+01	4.08E+01	1.71E+01	4.97E+01	8.95E+00	4.34E−02
	Best	**0.00E+00**	2.98E+00	2.22E+00	8.02E+00	7.53E+00	6.96E+00	9.95E−01	5.32E−04
F10	Average	**8.88E−16**	1.83E−09	2.26E−05	7.51E−07	4.88E−10	8.88E−16	3.19E−10	3.10E−04
	Std Dev	**9.86E−32**	1.49E−09	4.01E−05	4.04E−06	3.96E−10	9.86E−32	1.43E−10	0.001217
	Worst	**8.88E−16**	7.16E−09	2.09E−04	2.25E−05	1.66E−09	8.88E−16	6.78E−10	6.10E−03
	Best	**8.88E−16**	2.45E−10	1.78E−12	8.88E−16	3.02E−11	8.88E−16	1.22E−10	8.88E−16
F11	Average	**0.00E+00**	1.28E−01	5.01E−01	4.05E−01	7.53E−02	1.78E−01	1.52E+00	8.03E−02
	Std Dev	**0**	0.077065	0.128705	0.256938	0.015921	0.111694	0.78624	0.024518
	Worst	**0.00E+00**	3.71E−01	8.17E−01	9.68E−01	1.07E−01	5.27E−01	2.95E+00	1.23E−01

Table 4 continued

		HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
	Best	**0.00E+00**	3.75E−02	2.94E−01	0.00E+00	4.77E−02	6.40E−02	1.21E−01	2.29E−02
F12	Average	**9.15E−07**	9.39E−02	1.05E−02	6.54E−01	2.41E−02	9.36E−02	3.18E−03	8.95E−05
	Std Dev	**3.92E−07**	0.187187	0.00722	0.588009	0.020125	0.449848	0.017111	0.00012
	Worst	**1.95E−06**	6.25E−01	3.00E−02	2.17E+00	9.70E−02	2.50E+00	9.53E−02	5.19E−04
	Best	**4.02E−07**	2.52E−06	4.20E−04	6.49E−05	2.07E−03	2.41E−17	2.46E−20	5.89E−07
F13	Average	4.40E−06	1.90E−03	3.14E−02	7.02E−01	5.93E−04	1.83E−03	**2.64E−19**	8.44E−04
	Std Dev	1.68E−06	0.004102	0.013352	1.387596	0.000331	0.004095	**1.13E−19**	0.00221
	Worst	8.88E−06	1.11E−02	6.68E−02	5.22E+00	1.87E−03	1.10E−02	**5.20E−19**	1.13E−02
	Best	1.83E−06	4.27E−07	3.75E−03	1.38E−02	1.40E−04	1.21E−16	**1.01E−19**	4.31E−06
F14	Average	**9.98E−01**	9.98E−01	9.98E−01	1.30E+00	9.98E−01	1.16E+00	5.00E+00	9.98E−01
	Std Dev	**4.61E−12**	3.33E−16	2.41E−09	0.853122	3.07E−15	0.450142	3.794397	1.13E−10
	Worst	**9.98E−01**	9.98E−01	9.98E−01	4.95E+00	9.98E−01	2.98E+00	1.54E+01	9.98E−01
	Best	**9.98E−01**	9.98E−01	9.98E−01	9.98E−01	9.98E−01	9.98E−01	9.98E−01	9.98E−01

Table 5 Computational outcomes of the algorithms in solving multimodal benchmarks

		HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
F15	Average	3.72E−04	**3.07E−04**	1.07E−03	1.73E−03	4.29E−04	1.37E−03	3.74E−03	1.42E−03
	Std Dev	0.000241	**1.89E−13**	0.000279	0.00127	9.41E−05	0.001882	0.001868	0.001202
	Worst	1.32E−03	**3.07E−04**	1.83E−03	8.09E−03	6.75E−04	8.33E−03	1.02E−02	6.85E−03
	Best	3.07E−04	**3.07E−04**	5.09E−04	7.03E−04	3.16E−04	4.22E−04	1.18E−03	5.56E−04
F16	Average	**− 1.03E+00**	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00
	Std Dev	**9.35E−10**	0	3.75E−09	7.56E−09	0	0	0	0
	Worst	**− 1.03E+00**	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00
	Best	**− 1.03E+00**	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00	− 1.03E+00
F17	Average	**3.98E−01**	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01
	Std Dev	**1.45E−07**	1.11E−16	6.8E−06	5.18E−10	1.84E−14	1.11E−16	1.11E−16	1.2E−06
	Worst	**3.98E−01**	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01
	Best	**3.98E−01**	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01
F18	Average	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00
	Std Dev	**1.28E−07**	4.7E−15	0.005074	1.74E−06	4.97E−15	2.52E−15	3.35E−15	5.38E−05
	Worst	**3.00E+00**	3.00E+00	3.02E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00
	Best	**3.00E+00**	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00

Table 5 continued

		HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
F19	Average	**− 3.86E+00**	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00
	Std Dev	**0.001966**	2.66E−15	3.84E−09	0.001171	2.66E−15	2.66E−15	2.66E−15	9.07E−06
	Worst	**− 3.85E+00**	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00
	Best	**− 3.86E+00**	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00	− 3.86E+00
F20	Average	− 3.28E+00	**− 3.32E+00**	− 3.32E+00	− 3.27E+00	− 3.32E+00	− 3.21E+00	− 3.32E+00	− 3.28E+00
	Std Dev	0.056061	**0.021345**	2.86E−05	0.075844	2.94E−07	0.038462	1.33E−15	0.05714
	Worst	− 3.20E+00	**− 3.20E+00**	− 3.32E+00	− 3.08E+00	− 3.32E+00	− 3.14E+00	− 3.32E+00	− 3.20E+00
	Best	− 3.32E+00	**− 3.32E+00**	− 3.32E+00	− 3.32E+00	− 3.32E+00	− 3.32E+00	− 3.32E+00	− 3.32E+00
F21	Average	**− 1.02E+01**	− 9.74E+00	− 1.01E+01	− 8.47E+00	− 1.02E+01	− 6.65E+00	− 7.62E+00	− 8.64E+00
	Std Dev	**0.00034**	1.592751	0.024798	2.383475	4.19E−07	3.389785	3.414204	2.977379
	Worst	**− 1.02E+01**	− 2.68E+00	− 1.00E+01	− 5.06E+00	− 1.02E+01	− 2.63E+00	− 2.68E+00	− 2.68E+00
	Best	**− 1.02E+01**	− 1.02E+01	− 1.02E+01	− 1.02E+01	− 1.02E+01	− 1.02E+01	− 1.02E+01	− 1.01E+01
F22	Average	**− 1.04E+01**	− 1.04E+01	− 1.04E+01	− 7.96E+00	− 1.04E+01	− 8.67E+00	− 1.04E+01	− 8.74E+00
	Std Dev	**0.000252**	4.39E−13	0.034619	2.851794	7.13E−07	2.936026	0	2.992735
	Worst	**− 1.04E+01**	− 1.04E+01	− 1.02E+01	− 2.77E+00	− 1.04E+01	− 1.84E+00	− 1.04E+01	− 2.75E+00
	Best	**− 1.04E+01**	− 1.04E+01	− 1.04E+01	− 1.04E+01	− 1.04E+01	− 1.04E+01	− 1.04E+01	− 1.04E+01
F23	Average	**− 1.05E+01**	− 1.05E+01	− 1.05E+01	− 8.52E+00	− 1.05E+01	− 8.84E+00	− 1.05E+01	− 9.58E+00
	Std Dev	**0.000296**	1.65E−11	0.032012	2.661454	2.18E−05	3.106381	8.88E−15	2.447214
	Worst	**− 1.05E+01**	− 1.05E+01	− 1.04E+01	− 3.84E+00	− 1.05E+01	− 2.42E+00	− 1.05E+01	− 2.87E+00
	Best	**− 1.05E+01**	− 1.05E+01	− 1.05E+01	− 1.05E+01	− 1.05E+01	− 1.05E+01	− 1.05E+01	− 1.05E+01

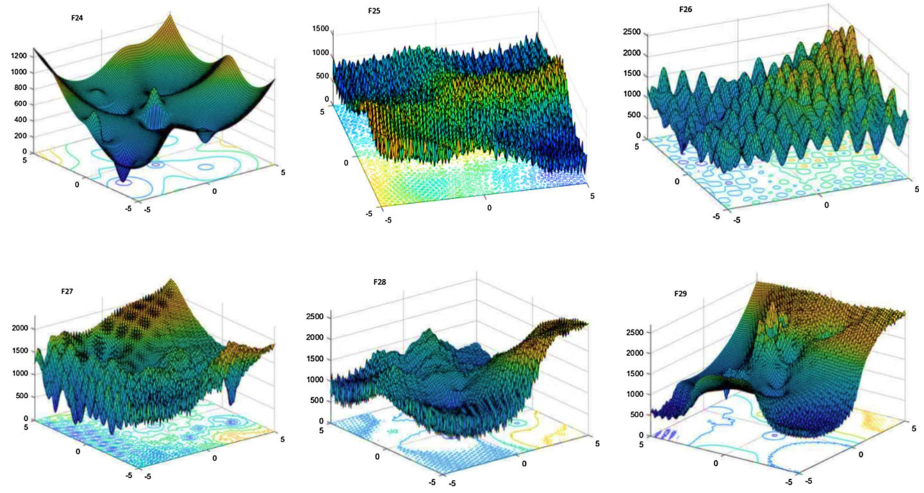


Fig. 11 Two-dimension view of the hybrid composite functions

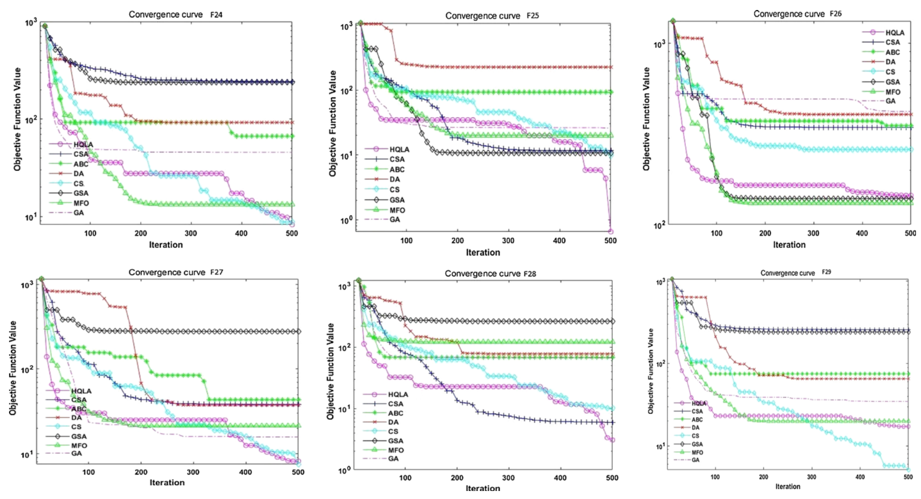


Fig. 12 Convergence plots for algorithms in solving composite problems

the reproduction rate was considered to be $R_0 = 0.7782$. In order to present the progress of the pandemic in the next few months, we extended our prediction to the next 365 days, as is shown in Fig. 16a, b.

Based on our results, considering the current social distancing and limitations, we will experience a significant decrease in the number of cases in the next few months if and only if strict measures such as partial lockdown remain in place for the next few months.

Table 6 Computational outcomes of the algorithms in solving hybrid composite benchmarks

		HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
F24	Average	**6.286974**	89.47798	74.9472	127.8248	11.7614	102.5087	229.0856	129.6113
	Std Dev	**2.737264**	103.3183	14.01152	116.2458	2.422529	67.72289	76.66939	146.3853
	Worst	**9.41E+00**	2.48E+02	1.02E+02	4.62E+02	1.84E+01	1.78E+02	2.81E+02	5.04E+02
	Best	**5.48E−01**	6.91E+00	5.77E+01	4.92E+01	8.76E+00	4.45E+00	2.86E+00	1.65E+01
F25	Average	**4.051293**	41.6118	74.98699	92.44491	34.89759	89.83728	202.1975	64.50657
	Std Dev	**1.305113**	36.35972	12.03202	94.10962	53.20973	87.77274	150.2123	83.70854
	Worst	**5.78E+00**	1.03E+02	9.35E+01	2.34E+02	1.81E+02	2.57E+02	5.34E+02	2.37E+02
	Best	**1.64E+00**	3.22E+00	5.10E+01	6.72E+00	5.45E+00	1.76E−14	1.37E+00	2.91E+00
F26	Average	**5.800523**	89.67129	75.67468	143.2063	11.27073	70.93509	241.6218	188.6492
	Std Dev	**2.535319**	104.9326	12.05648	146.979	2.933045	69.55739	81.02934	162.9257
	Worst	**9.23E+00**	2.54E+02	8.96E+01	5.31E+02	1.71E+01	2.25E+02	2.88E+02	5.15E+02
	Best	**3.27E−01**	1.35E+00	5.12E+01	2.82E+01	7.68E+00	6.34E+00	2.90E+00	2.50E+01
F27	Average	**5.824902**	35.16891	77.9421	161.152	9.838745	76.15984	186.7724	98.29543
	Std Dev	**2.358736**	68.64897	16.40409	151.9523	2.625766	81.88245	123.7938	77.67603
	Worst	**8.89E+00**	2.41E+02	1.00E+02	5.50E+02	1.31E+01	2.26E+02	3.34E+02	2.37E+02
	Best	**1.85E+00**	4.13E+00	4.27E+01	4.24E+01	3.38E+00	5.33E+00	4.40E+00	1.80E+01

Table 6 continued

		HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
F28	Average	**5.907383**	104.8916	67.73344	105.7646	11.92527	102.206	202.6379	85.30539
	Std Dev	**2.128054**	114.5205	14.15686	66.0598	2.57178	75.31781	137.0966	63.38328
	Worst	**9.12E+00**	2.68E+02	9.49E+01	2.37E+02	1.66E+01	2.26E+02	4.31E+02	2.35E+02
	Best	**2.34E+00**	8.17E+00	4.91E+01	4.95E+01	8.13E+00	7.10E+00	5.02E+00	1.69E+01
F29	Average	**5.907383**	104.8916	67.73344	105.7646	11.92527	102.206	202.6379	85.30539
	Std Dev	**2.128054**	114.5205	14.15686	66.0598	2.57178	75.31781	137.0966	63.38328
	Worst	**9.12E+00**	2.68E+02	9.49E+01	2.37E+02	1.66E+01	2.26E+02	4.31E+02	2.35E+02
	Best	**2.34E+00**	8.17E+00	4.91E+01	4.95E+01	8.13E+00	7.10E+00	5.02E+00	1.69E+01

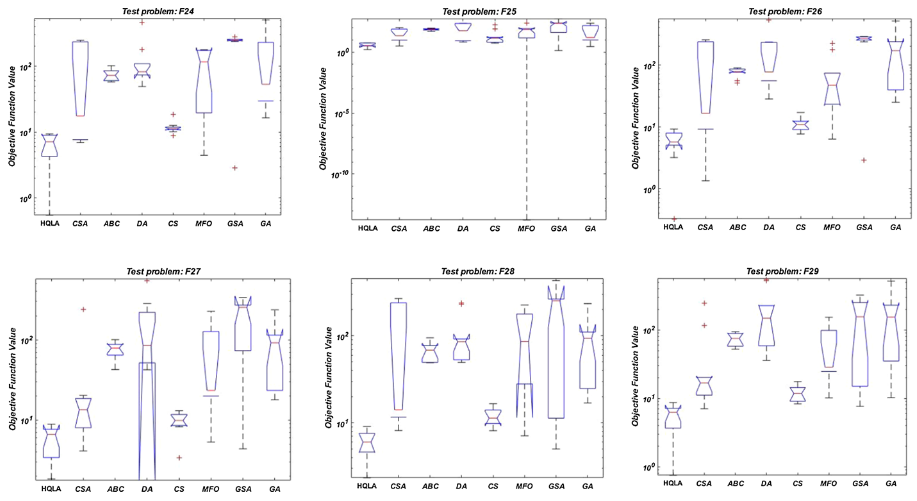


Fig. 13 Boxplot of the results in composite benchmarks

6 Sensitivity analyses and managerial insights

In the previous section, we reflected the pandemic growth over the next few months, considering the current partial lockdown and closure measures. However, in May 2020, the Quebec government ordered a gradual reopening of the businesses. Consequently, it is vital to discover how the reopening of the businesses will impact forthcoming circumstances. In this section, we examine the consequence of variation in transmission rates on the progress of the pandemic. Therefore, we augmented the parameters α, β, γ, δ, and ε and explore their impact on the number of infected, recovered, cumulative diagnosed, and death cases. We portray the outcomes in Figs. 17, 18, 19 and 20. The outcomes prove that the parameter α has a substantial effect on the pandemic growth, and increasing this parameter increases the number of infected, recovered, cumulative diagnosed, and death cases. Besides, increasing β, γ, and δ increases the number of infected, recovered, cumulative diagnosed, and death cases. Moreover, increasing the parameter ε remarkably reduces the number of infected, recovered, cumulative diagnosed, and death cases. Therefore, to decrease the spread of the virus and stop the pandemic, we need to reduce the transmission rate of the infection by applying significant social distancing and behavioral measures while increasing the detection rate of asymptomatic cases.

Based on our results, we observe that from day 250, we will see an increase in the number of infected cases from October 1, 2020, in Quebec by limiting the lockdown measures. Therefore, starting October 1, 2020, Quebec will experience a significant increase in the number of infected cases.

7 Conclusion

COVID-19 is a severe health threat, and the condition is growing every day. Therefore, presenting new procedures to model and predict the COVID-19 pandemic is vital. Prediction of the COVID-19 pandemic will enable policymakers to optimize the use of healthcare system

Table 7 Results of the Friedman's test

	HQLA	CSA	ABC	DA	CS	MFO	GSA	GA
F1	1.033333	4.233333	7.233333	7.266667	4.9	3.033333	2.033333	6.266667
F2	1	4.1	6.966667	7.9	5.3	2.633333	2.7	5.4
F3	1	2.8	7.933333	6.166667	3.933333	2.433333	5.133333	6.6
F4	1.1	3.833333	8	5.966667	5.2	3.433333	2.1	6.366667
F5	2.133333	3.666667	7.1	6.833333	2.8	4.466667	4.033333	4.966667
F6	3.366667	3.966667	7.233333	7.3	4.766667	2	1	6.366667
F7	1	2.833333	7.966667	6.4	6.1	4.333333	4.533333	2.833333
F8	3.266667	6.6	2.8	5.733333	4.166667	3.833333	8	1.6
F9	1	4.833333	4.333333	7.466667	6.166667	6.9	3.3	2
F10	2.25	6.666667	7.8	3.066667	5.533333	2.25	5.233333	3.2
F11	1.016667	3.8	6.666667	5.983333	2.866667	4.733333	7.633333	3.3
F12	2.933333	5.2	6.033333	7.566667	6.533333	2.333333	1.2	4.2
F13	2.933333	4.3	7.066667	7.933333	5.433333	2.633333	1	4.7
F14	5.333333	2.533333	6.7	3.216667	2.766667	3.2	7.966667	4.283333
F15	2.233333	1	5.3	6.4	3.066667	4.9	7.666667	5.433333
F16	7.6	3.45	7.266667	3.883333	3.45	3.45	3.45	3.45
F17	6.9	2.816667	7.9	3.3	5.166667	2.816667	2.816667	4.283333
F18	6.9	4.233333	8	3.916667	3.416667	2.166667	4.466667	2.9
F19	6.466667	2.533333	5.233333	6.666667	2.533333	2.533333	2.533333	7.5
F20	5.366667	2.3	4.833333	6.166667	3.133333	6.516667	1.05	6.633333
F21	4.633333	2.566667	6	4.133333	3.566667	4.8	3.566667	6.733333
F22	4.95	2.866667	6.7	5.233333	4.466667	3.283333	1.566667	6.933333
F23	5.833333	2.4	7.166667	5.5	4.333333	3.166667	1.766667	5.833333
F24	1.2	4.5	5	5.9	2.6	4.9	7.1	4.8
F25	1.6	4	5.6	5.4	4.1	5	6.1	4.2
F26	1.5	3.9	5.4	5.6	2.5	4.2	7	5.9
F27	1.4	3.4	5.9	6.1	2.6	4.7	6.1	5.8
F28	1.1	4.8	4.9	5.5	2.6	5.5	6.2	5.4
F29	1	3.5	5.5	6.4	2.6	5.2	5.6	6.2
Average	3.036207	3.711494	6.363218	5.824138	4.02069	3.839655	4.236207	4.968391

capacity and resource allocation to minimize the fatality rate. In this research, we design a new hybrid reinforcement learning-based algorithm capable of solving complex optimization problems. We applied our algorithm to several well-known benchmarks and show that the presented methodology provides high-quality solutions for the most complex problems in the literature. Besides, we showed the superiority of the offered method to state-of-the-art methods through several measures.

Moreover, to demonstrate the efficiency of the proposed methodology in optimizing real-world problems, we implemented our approach to the most recent data from Quebec, Canada, to predict the COVID-19 outbreak. Our algorithm, combined with the most recent mathematical model for COVID-19 pandemic prediction, accurately reflected the future trend of the pandemic. Furthermore, we analyzed several scenarios for deepening our insight into

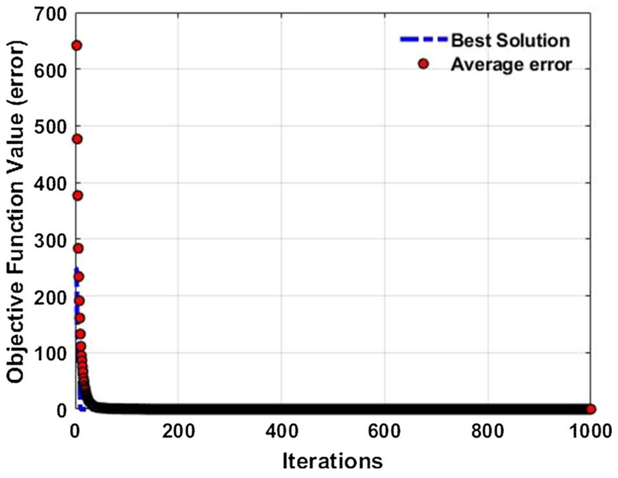


Fig. 14 Performance of HQLA in solving the SIDARTHE problem

Table 8 Results of fitting the model to real-data for Quebec

Parameters	Stages					
	January 25–March 15	Mar 15–Mar 24	Mar 24–Mar 28	Mar 28–Apr 2	Apr 2–Apr 13	After Apr 13
α	0.11807	0.421243	0.421243	0.08988	0.088024	0.088024
β	0.002927	0.000752	0.000752	0.000233	0.000233	0.000233
δ	0.002927	0.000752	0.000752	0.000233	0.000233	0.000233
γ	0.055155	0.208933	0.208933	0.071847	0.031887	0.031887
ε	0.039123	0.039123	0.013056	0.013056	0.013056	0.042366
ζ	0.084829	0.084829	0.084829	0.021848	0.021848	0.02182
η	0.084829	0.084829	0.084829	0.021848	0.021848	0.02182
θ	0.108995	0.108995	0.108995	0.108995	0.108995	0.108995
λ	0.02434	0.02434	0.02434	0.056476	0.056476	0.056476
κ	0.013091	0.013091	0.013091	0.017058	0.017058	0.017181
ξ	0.013091	0.013091	0.013091	0.017058	0.017058	0.017181
ρ	0.02434	0.02434	0.02434	0.017058	0.017058	0.017181
σ	0.013091	0.013091	0.013091	0.017058	0.017058	0.000218
μ	0.003922	0.003922	0.003922	0.00269	0.00269	0.00269
υ	0.031303	0.031303	0.031303	0.028218	0.028218	0.028218
τ	0.009446	0.009446	0.009446	0.009446	0.009446	0.009446

pandemic growth. We determined essential factors and delivered various managerial insights to help policymakers making decisions regarding future social measures.

Our results showed that the transmission rate caused by the interaction of a susceptible case with an asymptomatic case has the most significant effect on future trends. Increasing this parameter can significantly increase the number of infected, recovered, cumulative diagnosed,

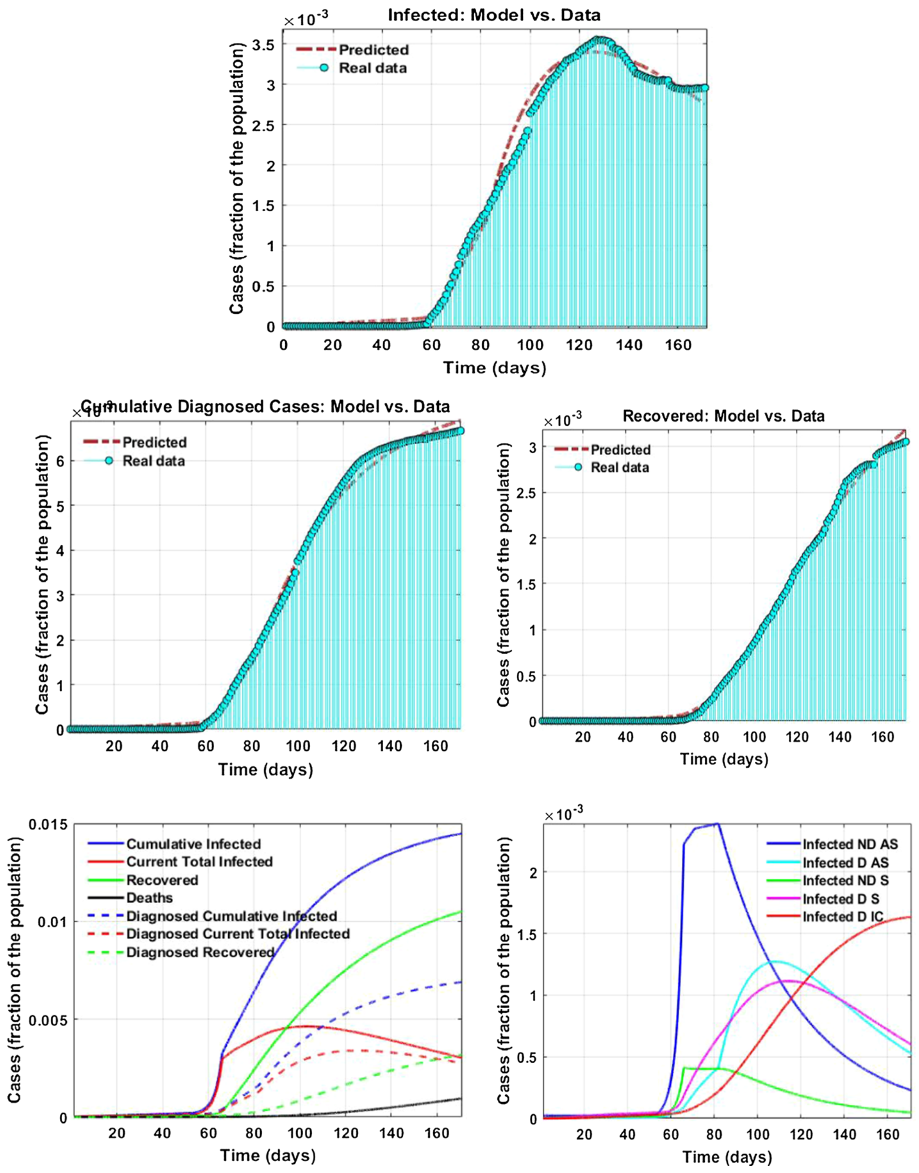


Fig. 15 Prediction versus data using HQLA. Non-diagnosed asymptomatic (ND AS), diagnosed asymptomatic (D AS), non-diagnosed symptomatic (ND S), diagnosed symptomatic (DS), and diagnosed with life-threatening symptoms (D IC)

and death cases. Besides, increasing the transmission rate due to contact of a susceptible case with a diagnosed, ailing, and recognized case increases the number of infected, recovered, cumulative diagnosed, and death cases. Moreover, increasing the parameter ε remarkably reduces the number of infected, recovered, cumulative diagnosed, and death cases. Therefore, to decrease the spread of the virus and stop the pandemic, we need to reduce the transmission

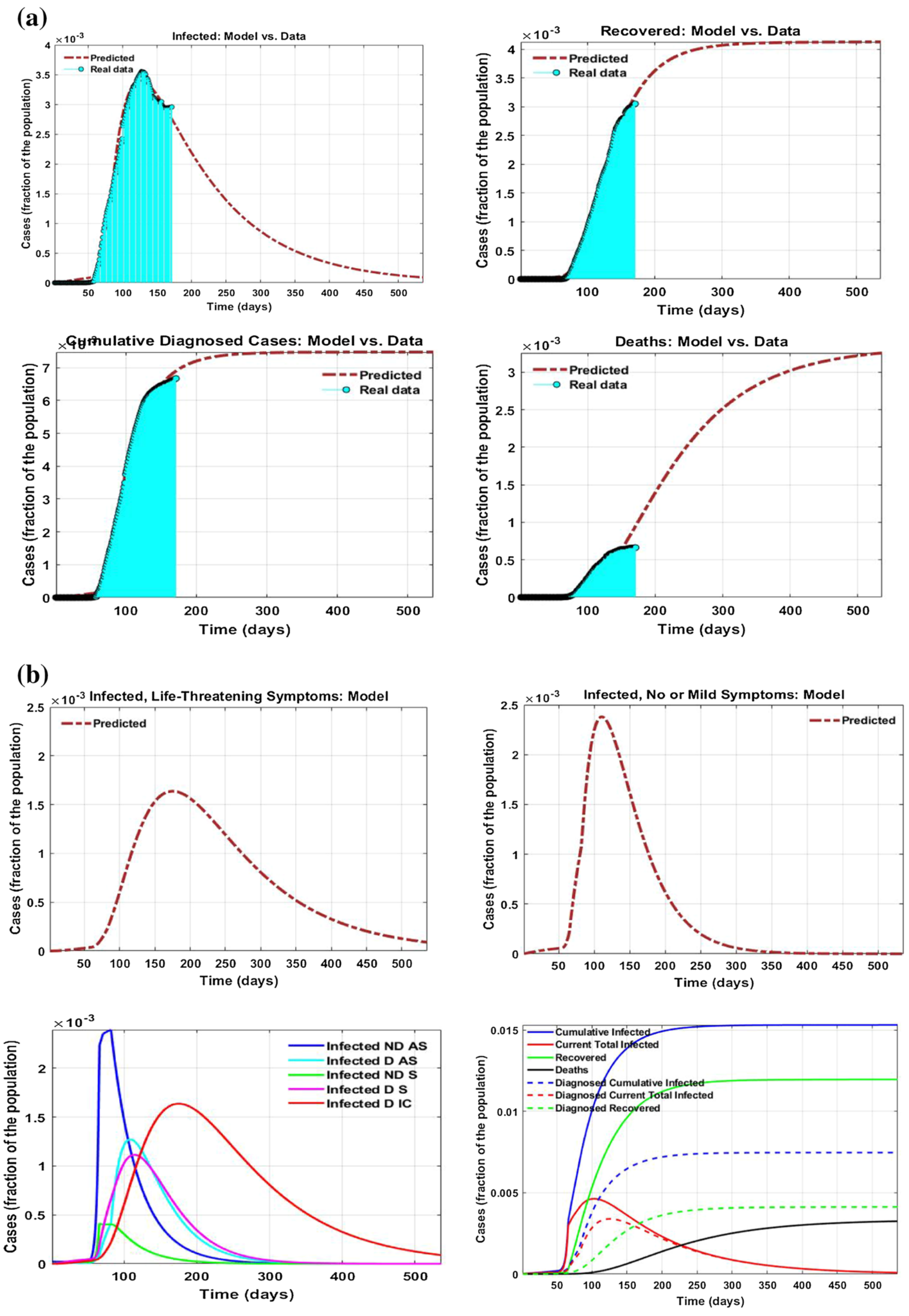


Fig. 16 **a** Prediction of future cases using SIDARTHE and HQLA. **b** Prediction of future cases using SIDARTHE and HQLA. Non-diagnosed asymptomatic (ND AS), diagnosed asymptomatic (D AS), non-diagnosed symptomatic (ND S), diagnosed symptomatic (DS), and diagnosed with life-threatening symptoms (D IC)

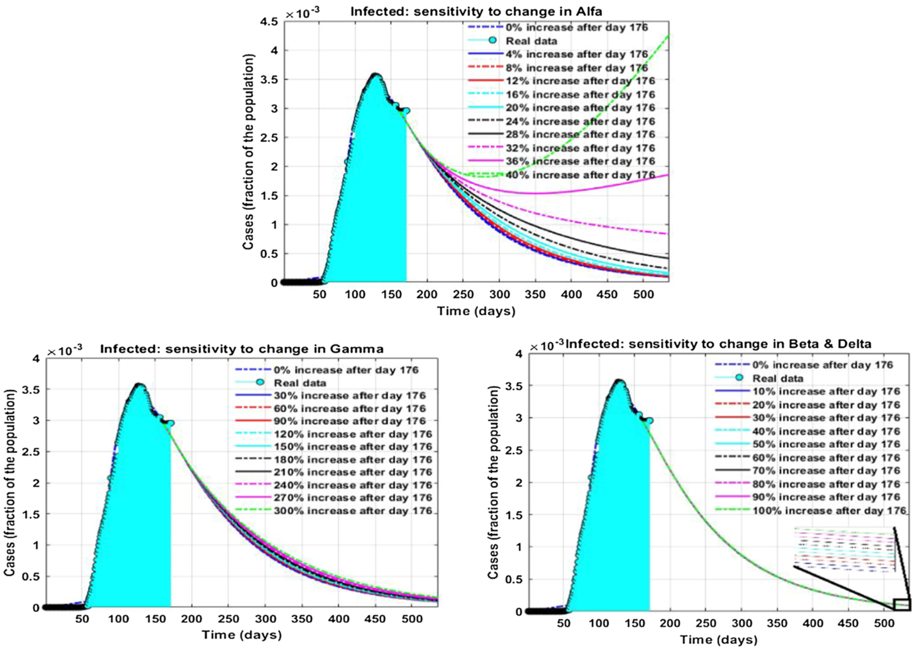


Fig. 17 Future scenarios of the infected cases in Quebec

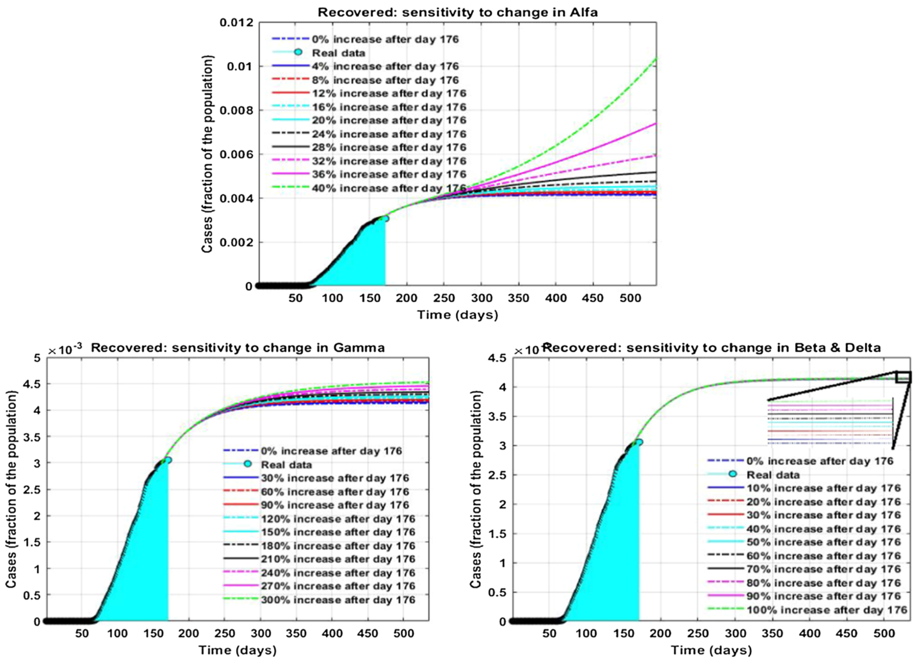


Fig. 18 Future scenarios of the cumulative diagnosed cases in Quebec

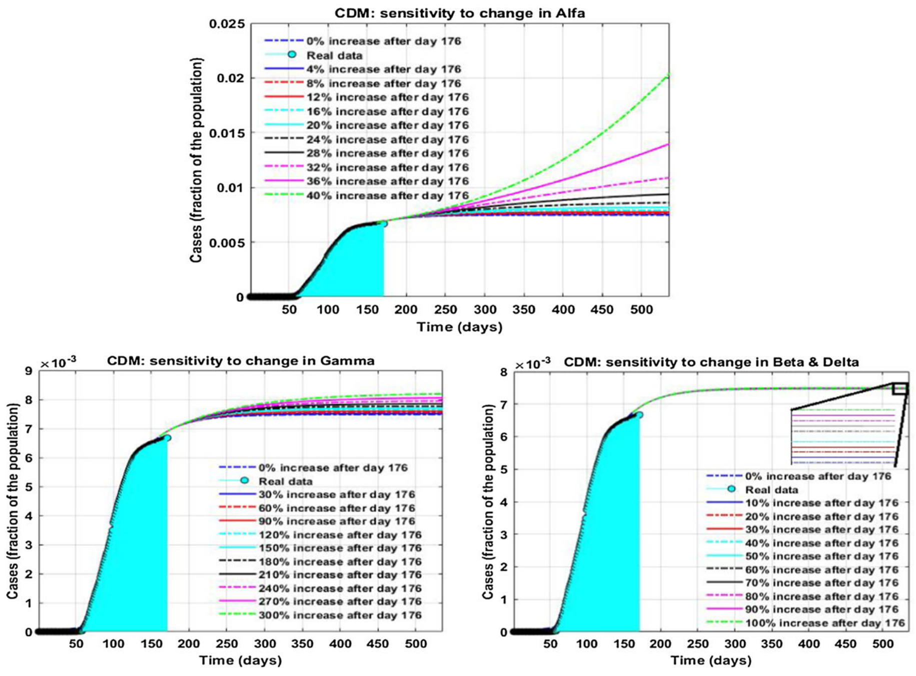


Fig. 19 Future scenarios of the recovered cases in Quebec

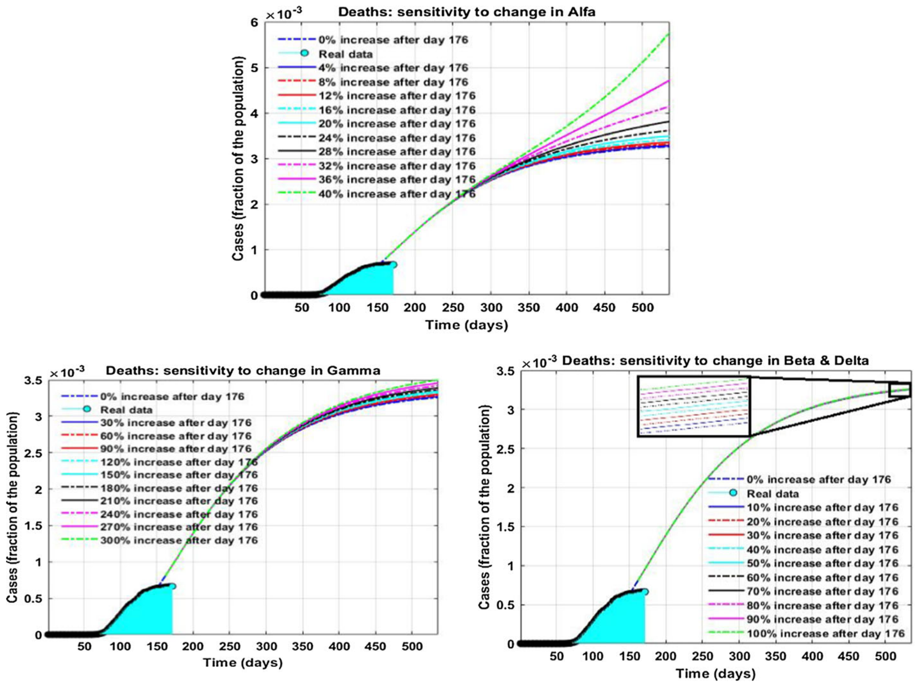


Fig. 20 Future scenarios of the recovered cases in Quebec

rate of the infection by applying significant social distancing and behavioral measures while increasing the detection rate of asymptomatic cases.

As future research, from a mathematical modeling viewpoint, it would be worthwhile to consider stochasticity in the proposed model (Belen et al. 2011; Özmen et al. 2011; Weber et al. 2011). Besides, it would be interesting to use the proposed model to plan for resource management during the pandemic to decrease the fatality rate by increasing the healthcare system capacity. Based on the predictions, healthcare managers can plan for testing kit allocation to test centers. Also, it would be worthwhile to consider the age, medical condition, and gender of the infected cases in the model from a modeling perspective. Moreover, it would also be interesting to use the proposed model to plan for managing healthcare resources such as personal protective equipment (PPE) and ventilators during the pandemic.

Appendix 1

See Table A1.

Table A1 The unimodal benchmark functions

Function	Range	$f_{\min}$
$f_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^{10}$	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10, 10]^{10}$	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	$[-100, 100]^{10}$	0
$f_4(x) = \max_i \{\lvert x_i \rvert, 1 \leq i \leq n\}$	$[-100, 100]^{10}$	0
$f_5(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30, 30]^{10}$	0
$f_6(x) = \sum_{i=1}^{n} \left(\left[x_i + 0.5 \right] \right)^2$	$[-100, 100]^{10}$	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	$[-1.28, 1.28]^{10}$	0
$F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	$[-500, 500]^{10}$	-418.9829×5
$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	$[-5.12, 5.12]^{10}$	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	$[-32, 32]^{10}$	0

Table A1 continued

Function	Range	$f_{\min}$
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^{10}$	0
$F_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}\{(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})]\right.$	$[-50, 50]^{10}$	0
$\left. + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)\right.$		
$y_i = 1 + \frac{x_i + 1}{4}$		
$F_{13}(x) = 0.1\left\{\sin^2(3\pi x_i) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)]\right.$	$[-50, 50]^{10}$	0
$\left. + (x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)\right.$		
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6})^{-1}\right)$	$[-65, 65]^2$	1
$F_{15}(x) = \sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right)^2$	$[-5, 5]^4$	0.00030
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 = \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	$[-5, 5]^2$	-1.0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	$[-5, 5]^2$	0.398

Table A1 continued

Function	Range	$f_{\min}$
$F_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$ $\times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	$[-2, 2]^2$	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	$[1, 3]^3$	− 3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	$[0, 1]^6$	− 3.32
$F_{21}(x) = -\sum_{i=1}^{5} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	− 10.1532
$F_{22}(x) = -\sum_{i=1}^{7} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	− 10.4028
$F_{22}(x) = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	− 10.5363

Appendix 2

In this appendix, we describe the SIDARTHE model proposed by Giordano et al. (2020). The model considers eight states for people, including Susceptible, Infected, Diagnosed, Ailing, Recognized, Threatened, Healed, Extinct. The model reflects cases in different situations, including symptomatic, asymptomatic, detected, undetected, and life-threatening symptoms that required ICU admission. The suggested model includes eight ordinary differential equations to show the progress of the pandemic over time. We use the following notations, as in Table A2, to present the model.

Based on the notations mentioned above, Giordano et al. (2020) presented the following model.

$$\dot{S}(t) = -S(t)(\alpha I(t) + \beta D(t) + \gamma A(t) + \delta R(t)) \tag{23}$$

$$\dot{I}(t) = S(t)(\alpha I(t) + \beta D(t) + \gamma A(t) + \delta R(t)) - (\varepsilon + \zeta + \lambda)I(t) \tag{24}$$

Table A2 Sets, parameters, and state variables of the model

Indices	
t	State index
Parameters	
α	Transmission rate due to contact of a susceptible case with an infected case
β	Transmission rate due to contact of a susceptible case with a diagnosed case
δ	Transmission rate due to contact of a susceptible case with an ailing case
γ	Transmission rate due to contact of a susceptible case with a recognized case
ε	The rate of detecting asymptomatic cases
ζ	The probability that an infected case is aware of being infected
η	The probability that an infected case is unaware of being infected
θ	The detection rate of symptomatic cases
$\lambda, \kappa, \xi, \rho, \sigma$	The recovery rate of the five categories of infected cases
μ	The probability that an undetected/detected infected case shows life-threatening symptoms
υ	The probability that a detected infected case develops life-threatening symptoms
τ	Mortality rate
State variables	
$S(t)$	The fraction of susceptible (not infected) cases in the population
$I(t)$	The fraction of infected (infected and undetected cases without symptoms) cases in the population
$D(t)$	Fraction of diagnosed (infected and detected cases without symptoms) cases in the population
$A(t)$	The fraction of ailing (infected and undetected cases with symptoms) cases in the population
$R(t)$	The fraction of recognized (infected and detected cases with symptoms) cases in the population
$T(t)$	The fraction of threatened (infected detected cases that developed life-threatening symptoms) cases in the population
$H(t)$	The fraction of recovered cases in the population
$E(t)$	The fraction of death cases in the population

$$\dot{D}(t) = \varepsilon I(t) - (\eta + \rho)D(t) \tag{25}$$

$$\dot{A}(t) = \zeta I(t) - (\theta + \mu + \kappa)A(t) \tag{26}$$

$$\dot{R}(t) = \eta D(t) + \theta A(t) - (\nu + \xi)R(t) \tag{27}$$

$$\dot{T}(t) = \mu A(t) + \nu R(t) - (\sigma + \tau)T(t) \tag{28}$$

$$\dot{H}(t) = \lambda I(t) + \rho D(t) + \kappa A(t) + \xi R(t) + \sigma T(t) \tag{29}$$

$$\dot{E}(t) = \tau T(t) \tag{30}$$

References

Adam, S. P., Alexandropoulos, S. A. N., Pardalos, P. M., & Vrahatis, M. N. (2019). No free lunch theorem: A review. In *Approximation and optimization* (pp. 57–82). Springer, Cham.

Ahmadianfar, I., Bozorg-Haddad, O., & Chu, X. (2020). Gradient-based optimizer: A new metaheuristic optimization algorithm. *Information Sciences*, *540*, 131–159.

Akhtar, E., & Farrukh, S. M. (2017). Practical Reinforcement Learning: Develop self-evolving, intelligent agents with OpenAI Gym, Python and Java. Packt Publishing.

Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Computers & Structures, 169,* 1–12.

Belen, S., Kropat, E., & Weber, G. W. (2011). On the classical Maki–Thompson rumour model in continuous time. *Central European Journal of Operations Research, 19*(1), 1–17.

Bertsekas, D. P. (2019). *Reinforcement learning and optimal control*. Belmont, MA: Athena Scientific.

CARLY WEEKS & TU THANH HA. (n.d.). *Quebec hospitals struggling with influx of COVID-19 patients even as province moves to reopen—The Globe and Mail*. Retrieved November 1, 2020, from https://www.theglobeandmail.com/canada/article-quebec-hospitals-struggling-with-influx-of-covid-19-patients-even-as/.

Cerby, V. (1985). Thermodynamical approach to the travelling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications, 45,* 41–51.

Coronavirus Disease (COVID-19) in Québec | Gouvernement du Québec. (n.d.). Retrieved October 11, 2020, from https://www.quebec.ca/en/health/health-issues/a-z/2019-coronavirus/.

Cui, L., Hu, H., Yu, S., Yan, Q., Ming, Z., Wen, Z., et al. (2018). DDSE: A novel evolutionary algorithm based on degree-descending search strategy for influence maximization in social networks. *Journal of Network and Computer Applications, 103,* 119–130.

Du, H., Wu, X., & Zhuang, J. (2006). Small-world optimization algorithm for function optimization. In *International conference on natural computation* (pp. 264–273). Berlin: Springer.

Eberhart, R., & Kennedy, J. (1995, November). Particle swarm optimization. In Proceedings of the IEEE international conference on neural networks (Vol. 4, pp. 1942-1948). Citeseer.

Eskandar, H., Sadollah, A., Bahreininejad, A., & Hamdi, M. (2012). Water cycle algorithm—A novel metaheuristic optimization method for solving constrained engineering optimization problems. *Computers & Structures, 110,* 151–166.

Fahmy, G., & Ross, S. (2020). CTV News Montreal Digital Reporter Montreal ERs are near capacity with delayed health problems, a worrying sign as second wave looms. Montreal. https://montreal.ctvnews.ca/montreal-ers-are-near-capacity-with-delayed-health-problems-a-worrying-sign-as-second-wave-looms-1.5003710?cache=%3FclipId%3D68597.

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966). *Artificial intelligence through simulated evolution*. London: Wiley.

Formato, R. A. (2007). Central force optimization. *Progress in Electromagnetics Research, 77,* 425–491.

Giordano, G., Blanchini, F., Bruno, R., Colaneri, P., Di Filippo, A., Di Matteo, A., et al. (2020). Modelling the COVID-19 epidemic and implementation of population-wide interventions in Italy. *Nature Medicine*, *1*(26), 855–860.

Hashemi Doulabi, H., Jaillet, P., Pesant, G., & Rousseau, L. M. (2020a). Exploiting the structure of two-stage robust optimization models with exponential scenarios. *INFORMS Journal on Computing*. https://doi.org/10.1287/ijoc.2019.0928.

Hashemi Doulabi, H., Pesant, G., & Rousseau, L. M. (2020b). Vehicle routing problems with synchronized visits and stochastic travel and service times: Applications in healthcare. *Transportation Science, 54*(4), 1053–1072.

Hatamlou, A. (2013). Black hole: A new heuristic optimization approach for data clustering. *Information Sciences, 222,* 175–184.

Health Canada. (2020). Drugs and vaccines for COVID-19: Authorized clinical trials [Decisions]. Aem. https://www.canada.ca/en/health-canada/services/drugs-health-products/covid19-industry/drugs-vaccines-treatments/list-authorized-trials.html.

Holland, J. H. (1992). Genetic algorithms. *Scientific American, 267*(1), 66–73.

Hoursan, H., Farahmand, F., & Ahmadian, M. T. (2020). A three-dimensional statistical volume element for histology informed micromechanical modeling of brain white matter. *Annals of Biomedical Engineering, 48*(4), 1337–1353.

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization, 39*(3), 459–471.

Kaveh, A., & Dadras, A. (2017). A novel meta-heuristic optimization algorithm: Thermal exchange optimization. *Advances in Engineering Software, 110,* 69–84.

Kaveh, A., Seddighian, M. R., & Ghanadpour, E. (2020). Black Hole Mechanics Optimization: A novel meta-heuristic algorithm. *Asian Journal of Civil Engineering*, *21*(7), 1129–1149.

Kaveh, A., & Talatahari, S. (2010). A novel heuristic optimization method: charged system search. *Acta Mechanica, 213*(3–4), 267–289.

Khalilpourazari, S., & Khalilpourazary, S. (2019). An efficient hybrid algorithm based on Water Cycle and Moth-Flame Optimization algorithms for solving numerical and constrained engineering optimization problems. *Soft Computing, 23*(5), 1699–1722.

Khalilpourazari, S., Khalilpourazary, S., Çiftçioğlu, A. Ö., & Weber, G. W. (2020a). Designing energy-efficient high-precision multi-pass turning processes via robust optimization and artificial intelligence. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-020-01648-0.

Khalilpourazari, S., Naderi, B., & Khalilpourazary, S. (2020b). Multi-objective stochastic fractal search: A powerful algorithm for solving complex multi-objective optimization problems. *Soft Computing, 24*(4), 3037–3066.

Khalilpourazari, S., & Pasandideh, S. H. R. (2019). Sine–cosine crow search algorithm: Theory and applications. *Neural Computing and Applications*,*32*, 7725–7742.

Khalilpourazari, S., Pasandideh, S. H. R., & Niaki, S. T. A. (2019). Optimizing a multi-item economic order quantity problem with imperfect items, inspection errors, and backorders. *Soft Computing, 23*(22), 11671–11698.

Khalilpourazari, S., Soltanzadeh, S., Weber, G. W., & Roy, S. K. (2020c). Designing an efficient blood supply chain network in crisis: Neural learning, optimization and case study. *Annals of Operations Research, 289*(1), 123–152.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). *Optimization by simulated annealing. science, 220*(4598), 671–680.

Kliff, S., Satariano, A., Silver-Greenberg, J., & Kulish, N. (2020). There aren't enough ventilators to cope with the coronavirus—The New York Times. (n.d.). Retrieved November 1, 2020, from https://www.nytimes.com/2020/03/18/business/coronavirus-ventilator-shortage.html.

Koza, J. R., & Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural selection* (Vol. 1). London: MIT Press.

Lapierre, M. (2020). *Quebec's first case of COVID-19 has been confirmed*. Montreal. https://montreal.ctvnews.ca/quebec-s-first-case-of-covid-19-has-been-confirmed-1.4831088.

Li, M. D., Zhao, H., Weng, X. W., & Han, T. (2016). A novel nature-inspired algorithm for optimization: Virus colony search. *Advances in Engineering Software, 92,* 65–88.

Liang, J. J., Suganthan, P. N., & Deb, K. (2005). Novel composition test functions for numerical global optimization. In *Proceedings 2005 IEEE swarm intelligence symposium, 2005*. SIS 2005. (pp. 68–75). IEEE.

Lotfi, R., Mehrjerdi, Y. Z., Pishvaee, M. S., Sadeghieh, A., & Weber, G. W. (2019). A robust optimization model for sustainable and resilient closed-loop supply chain network design considering conditional value at risk. *Numerical Algebra, Control and Optimization.* https://doi.org/10.3934/naco.2020023.

Lotfi, R., Mostafaeipour, A., Mardani, N., & Mardani, S. (2018). Investigation of wind farm location planning by considering budget constraints. *International Journal of Sustainable Energy, 37*(8), 799–817.

Lotfi, R., Weber, G. W., Sajadifar, S. M., & Mardani, N. (2020). Interdependent demand in the two-period newsvendor problem. *Journal of Industrial and Management Optimization, 16*(1), 117.

Martínez-Álvarez, F., Asencio-Cortés, G., Torres, J. F., Gutiérrez-Avilés, D., Melgar-García, L., Pérez-Chacón, R.,… & Troncoso, A. (2020). Coronavirus Optimization Algorithm: A bioinspired metaheuristic based on the COVID-19 propagation model. arXiv preprint arXiv:2003.13633.

Mirjalili, S. (2015a). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems, 89,* 228–249.

Mirjalili, S. (2015b). The ant lion optimizer. *Advances in Engineering Software, 83,* 80–98.

Mirjalili, S. (2016a). Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications, 27*(4), 1053–1073.

Mirjalili, S. (2016b). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems, 96,* 120–133.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67.

Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software, 114,* 163–191.

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016a). Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Computing and Applications, 27*(2), 495–513.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf optimizer. *Advances in Engineering Software, 69,* 46–61.

Mirjalili, S., Saremi, S., Mirjalili, S. M., & Coelho, L. D. S. (2016b). Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Systems with Applications, 47,* 106–119.

Moghaddam, F. F., Moghaddam, R. F., & Cheriet, M. (2012). Curved space optimization: A random search based on general relativity theory. arXiv preprint arXiv:1208.2214.

Mohammadi, M., & Khalilpourazari, S. (2017). Minimizing makespan in a single machine scheduling problem with deteriorating jobs and learning effects. In *Proceedings of the 6th international conference on software and computer applications* (pp. 310–315).

Özmen, A., Weber, G. W., Batmaz, İ., & Kropat, E. (2011). RCMARS: Robustification of CMARS with different scenarios under polyhedral uncertainty set. *Communications in Nonlinear Science and Numerical Simulation, 16*(12), 4780–4787.

Price, K. V. (2013). Differential evolution. In *Handbook of optimization* (pp. 187–214). Berlin: Springer.

Public Health Authority of Canada. (2020). Coronavirus disease (COVID-19): Symptoms and treatment [Education and awareness]. Aem. https://www.canada.ca/en/public-health/services/diseases/2019-novel-coronavirus-infection/symptoms.html.

Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences, 179*(13), 2232–2248.

Rechenberg, I. (1978). Evolutions strategien. In *Simulations methoden in der Medizin und Biologie* (pp. 83–114). Berlin: Springer.

Salimi, H. (2015). Stochastic fractal search: A powerful metaheuristic algorithm. *Knowledge-Based Systems, 75,* 1–18.

Sangaiah, A. K., Goli, A., Tirkolaee, E. B., Ranjbar-Bourani, M., Pandey, H. M., & Zhang, W. (2020). Big data-driven cognitive computing system for optimization of social media analytics. *IEEE Access, 8,* 82215–82226.

Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper optimisation algorithm: Theory and application. *Advances in Engineering Software, 105,* 30–47.

Shareef, H., Ibrahim, A. A., & Mutlag, A. H. (2015). Lightning search algorithm. *Applied Soft Computing, 36,* 315–333.

Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation, 12*(6), 702–713.

Tirkolaee, E. B., Goli, A., Faridnia, A., Soltani, M., & Weber, G. W. (2020a). Multi-objective optimization for the reliable pollution-routing problem with cross-dock selection using Pareto-based algorithms. *Journal of Cleaner Production, 276,* 122927.

Tirkolaee, E. B., Goli, A., & Weber, G. W. (2020b). Fuzzy mathematical programming and self-adaptive artificial fish swarm algorithm for just-in-time energy-aware flow shop scheduling problem with outsourcing option. *IEEE Transactions on Fuzzy Systems*, *28*(11), 2772–2783.

Topal, A. O., & Altun, O. (2016). A novel meta-heuristic algorithm: Dynamic virtual bats algorithm. *Information Sciences, 354,* 222–235.

Wang, S. Y., Wang, L., Liu, M., & Xu, Y. (2013). An effective estimation of distribution algorithm for solving the distributed permutation flow-shop scheduling problem. *International Journal of Production Economics, 145*(1), 387–396.

Weber, G. W., Defterli, O., Gök, S. Z. A., & Kropat, E. (2011). Modeling, inference and optimization of regulatory networks based on time series data. *European Journal of Operational Research, 211*(1), 1–14.

WHO. (2020). Q&A on coronaviruses. Retrieved October 11, 2020 from https://www.who.int/news-room/q-a-detail/q-a-coronaviruses.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1,* 67–82.

Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. In *2009 World congress on nature & biologically inspired computing (NaBIC)* (pp. 210–214). IEEE.

Zamli, K. Z., Din, F., Ahmed, B. S., & Bures, M. (2018). A hybrid Q-learning sine–cosine-based strategy for addressing the combinatorial test suite minimization problem. *PLoS ONE, 13*(5), e0195675.

Zare Mehrjerdi, Y., & Lotfi, R. (2019). Development of a mathematical model for sustainable closed-loop supply chain with efficiency and resilience systematic framework. *International Journal of Supply and Operations Management, 6*(4), 360–388.