A Recurrent Neural Network-Based Approach for Joint Chance Constrained Stochastic Optimal Control

Shu-Bo Yang[a], Zukui Li[a,*] and Jesús Moreira

[a]Department of Chemical and Materials Engineering, University of Alberta, 9211 116 St, Edmonton, T6G1H9, Alberta, Canada
[b]Corporate Planning, Imperial Oil, 505 Quarry Park Blvd SE, Calgary, T2C5N1, Alberta, Canada

ARTICLE INFO

Keywords:
Optimal control
Recurrent neural network
Stochastic model predictive control
Joint chance-constrained optimization
Surrogate modelling
Sample average approximation

ABSTRACT

A recurrent neural network (RNN)-based approach is proposed in this paper to handle joint chance-constrained stochastic optimal control problems (SOCP) and stochastic model predictive control (SMPC) implementations. In the proposed approach, the joint chance constraint (JCC) in a SOCP is first reformulated as a quantile-based inequality. Then, the sample average approximation (SAA) method is used to build the RNN-based surrogate model for the quantile function. Afterwards, the RNN-based model is embedded into the probabilistic constraint of the SOCP. Subsequently, the SOCP involving the RNN-based model can be solved using a deterministic nonlinear optimization solver. Moreover, while applying the proposed approach to the SMPC, the SOCP involving the RNN-based model is solved repeatedly at different sampling instants, based on different initial system states. The proposed approach is applied to a numerical illustrating example and a chemical process case study to demonstrate its capability of handling the SOCP and the SMPC implementation.

1. Introduction

In optimal control problems, the aim is to find a control sequence for a dynamic system over a period of time (possibly infinite) such that an associated reward (or cost) objective function is maximized (or minimized). Model predictive control (MPC) is an advanced control methodology that is based on optimal control. At each time step, a shorter finite horizon problem that approximates the original long horizon optimal control problem is solved, and only the first input is applied to the system. This process is repeated in a receding horizon fashion [1].

Deterministic optimal control and MPC approaches rely on deterministic models, without considering uncertainty in controlled systems. However, uncertainty generally exists in real-world problems. Due to uncertainty in the system dynamics, stochastic optimal control is needed. Similarly, stochastic MPC (SMPC) is a strategy to cope with MPC problems under uncertainty [2]. In those problems, it is common to use the expected value of the reward (cost) function and to enforce some operating constraints to be satisfied with certain confidence levels through chance constraints. Under chance constraints, optimal decisions are required to satisfy the constraints with a certain probability target. There are two types of chance constraint: the individual chance constraint and the joint chance constraint [3].

As to the individual chance constraint, the term "individual" relates to the fact that each of the stochastic constraints is reformulated into a chance constraint individually. Individual chance constraints are relatively easier to handle [4]. In terms of the drawback, they only guarantee that each equation satisfies the constraint to a certain confidence level. In terms of the joint chance constraint (JCC), it ensures that a set of constraints are satisfied simultaneously to a certain confidence level, which is a more realistic modeling of uncertain constraints in many engineering applications. However, JCCs are more difficult to handle and they are generally solved through approximations [5]. Analytical approximation methods [6] and sampling-based methods [7] are two main ways for the JCC approximation. Analytical approximation approaches are exploited to approximate a joint chance-constrained problem with a deterministic optimization formulation [8]. For example, the robust optimization (RO) approach [9] is one analytical approximation method that does not require the full information of uncertainty distributions, and uncertainties are described using uncertainty sets [10]. Sampling-based approaches include the scenario approximation and the sample average approximation. The main concept of the scenario approximation is to generate samples of the random parameters in the problem studied, and then the JCC in the problem is approximated by a set of constraints corresponding to each generated sample [11]. However, the randomness of sample generation might cause the infeasibility of the problem with the approximated JCC [8]. To overcome this drawback, the sample average approximation (SAA) is developed as the generalization of the scenario approximation. In the SAA method, the random parameters in the studied problem are first sampled. Then, the constraint satisfaction probability based on the samples is enforced to be greater or equal to the defined confidence level. SAA is an easy-to-implement approach that can make chance-constrained problems tractable [7].

By utilizing different methods to address uncertainty and chance constraints, different SMPC approaches have been developed. Stochastic tube approaches are commonly used for linear SMPC systems [12]. Stochastic tube approaches exploit a state feedback control law with a fixed feedback gain to minimize an infinite-horizon value function subject to individual chance constraints. Stochastic tube approaches

*Corresponding author
zukui@ualberta.ca (Z. Li)
ORCID(s): 0000-0002-8332-8436 (Z. Li)

are limited to individual chance constraints and unable to address hard input constraints [13]. Sample-based SMPC approaches use sampling methods to realize uncertain parameters for characterizing stochastic system dynamics [14]. The sample-based SMPC based on the Monte Carlo simulation is presented in Moen's work [15]. The main drawback of a sample-based SMPC approach is the high computational cost due to the large sample size required for the optimization at each sampling instant. The scenario-based SMPC method employs the scenario approach for the involved stochastic optimization [16]. The sequential approach proposed in Navia's work [17] is an iterative method utilized to solve the stochastic optimal control problem. In the sequential approach, the inverse mapping method [18] is exploited to evaluate the probabilities in chance constraints. Reliable optimal decisions can be attained using the sequential approach but its long solution time hinders its application to real SMPCs. To overcome the above drawbacks in the existing SMPC approaches, a more applicable SMPC approach with higher computation efficiency should be developed for both linear and nonlinear MPC problems under uncertainty.

Machine learning has received lots of attention in system identification and control recently [19, 20]. Among numerous machine learning techniques, recurrent neural networks (RNN) have been widely used for modelling nonlinear dynamic systems [21, 22, 23]. Unlike one-way connections between neurons in feed-forward neural networks, feedback loops exist in an RNN architecture, that introduces the past information from previous inputs to the computation of the hidden state in the current time step. Hidden states act as a memory of an RNN and capture dynamic behaviour in a way conceptually similar to nonlinear state-space differential equation models [24]. An RNN is capable of processing sequential data and generating a sequence of multi-step ahead predictions. Because of the above-mentioned "memory feature" of RNNs, RNNs can outperform the traditional feed-forward neural networks while addressing sequential data or modeling dynamic systems [25]. To further enhance performance on sequential data modelling, more advanced RNNs, such as the long short-term memory (LSTM) and the gated recurrent unit (GRU) [26], have been developed. Due to outstanding performance on sequence prediction, different types of RNNs have been applied to MPC [27]. Hertneck et al. proposed a neural network-based supervised-learning framework to approximate the robust MPC for enhancing the computational efficiency [28]. Drgoňa et al. [29] exploited deep time-delay neural networks and regression trees to approximate the MPC to reduce the implementation cost. Quan and Chung [30] employed the RNN to approximate the MPC for reducing the computational burden. Although the machine learning-based approximation for MPC problems has been extensively studied, its application to the joint chance-constrained SMPC is very limited so far. Therefore, a computationally efficient joint chance-constrained SMPC approach based on the RNN approximation is proposed in this work.

In this study, a new RNN-based approach is proposed to deal with the stochastic optimal control problem (SOCP) and the SMPC implementation. In this study, the JCC is reformulated as a quantile-based inequality. The quantile function (QF) in the inequality is furthered approximated by the empirical QF based on the SAA method. Then, since the empirical QF is still a stochastic and complex function, it is further modelled by an RNN-based model to reduce complexity and to make the SOCP deterministically solvable. At each sampling instant, only one quantile value should be predicted by the RNN-based model to examine the constraint satisfaction, regardless of the number of constraints incorporated in the JCC. The proposed approach exploits the strong temporal prediction ability and high computation efficiency of the RNN to handle SOCPs and SMPC implementations.

Different from the full approximation of the whole optimization problem, the proposed method only approximates the joint chance constraint (JCC) through RNN. From this point of view, the reliability of the solution depends on the accuracy of the JCC approximation accuracy. While we don't provide a theoretical guarantee on the solution feasibility, the proposed approach can efficiently obtain solutions with satisfactory constraint satisfaction probabilities for both SOCP and SMPC implementation. This approach can determine the optimal decision efficiently with a shorter solution time, compared to the sample-based approach from the literature [15]. The comparison is shown in the hydrodesulphurisation process case study. Moreover, this approach is applicable for both, stochastic linear and nonlinear dynamic systems, involving any number of stochastic parameters and constraints.

This manuscript is structured as follows. Section 2 presents the stochastic optimal control problem through a motivating example. Section 3 discusses the reformulation of the chance constraints through quantile function. Section 4 presents the proposed RNN approach for stochastic optimal control with the illustrating example studied followed by a chemical industry case study presented in section 5. Moreover, the extension of the proposed approach to the SMPC of the hydrodesulphurisation process is illustrated in Section 6.

2. Stochastic optimal control problem

The stochastic optimal control problem (SOCP) studied in this work is formulated in discrete time, which is given as:

$$\min_{u_j \in \mathbb{U}} \quad \mathbb{E}\left[\sum_{j=0}^{K-1} J(x_j, u_j) + J_f(x_K)\right] \tag{1a}$$

$$\text{s.t.} \quad x_{j+1} = f_D(x_j, u_j, \xi) \quad j = 0, .., K-1 \tag{1b}$$

$$Pr(g_i(x_j, u_{j-1}, \xi) \le 0, i = 1, \cdots, p) \ge 1 - \varepsilon \quad j = 1, .., K \tag{1c}$$

$$x_0 = x_{t_0} \tag{1d}$$

where j is the index for sampling instants. K is the number of sampling instants. In the above discrete-time SOCP, the

prediction horizon covers the sampling instants $j = 0, ..., K$, and the control horizon covers the sampling instants $j = 0, ..., K-1$. $x_j \in \mathbb{R}^{n_x}$ indicates the state at a certain sampling instant j ($j = 1, ..., K$) in the prediction horizon, based on the initial state $x_0 = x_{t_0}$. $u_j \in \mathbb{R}^{n_u}$ is the system input and $\mathbb{U}$ is the set of feasible inputs. $\xi \in \mathbb{R}^{n_\xi}$ is the uncertainty vector. $\mathbb{E}[\cdot]$ denotes the expectation. J is the cost function and J_f denotes the terminal cost. Equation (1b) represents the process dynamic model. f_D is a function describing the discrete-time system dynamics. Inequality (1c) denotes a JCC which enforces the probability of constraint satisfaction at a certain sampling instant j. $Pr(\cdot)$ represents the probability measure. $g_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_\xi} \to \mathbb{R}$ is a constraint function. p is the number of constraints involved in the JCC. $1-\varepsilon$ is the target probability of constraint satisfaction ($\varepsilon \in [0, 1]$). While there are other options to group JCCs, we apply this popular JCC setting in this work. The proposed RNN-based approach is based on the above discrete SOCP.

Illustrating example

A numerical SOCP example involving the differential equation model: $\frac{dx_1(t)}{dt} = x_2(t) + \xi_1$, $\frac{dx_2(t)}{dt} = -u(t) + \xi_2$, is used as an illustrating example. The involved differential equation model is discretized using the trapezoidal rule from time $t = 0 \sim 3$ with 10 time intervals. The values of x_1 and x_2 at 10 particular sampling instants ($t = 0.3 \sim 3$ with an time interval of 0.3) are selected for the problem-solving of the discrete-time formulation of the illustrating example. The discrete-time form of the differential equation model is denoted as f_D based on the x_1 and x_2 at the 10 sampling instants. The discrete-time formulation of the illustrating example is given as:

$$\min_{u_j} \quad 2 \times \sum_{j=0}^{9} \frac{\Delta t}{2} (\mathbb{E}[x_{1,j}] + \mathbb{E}[x_{1,j+1}]) \tag{2a}$$

$$\text{s.t.} \quad x_j = \left[x_{1,j}, x_{2,j}\right], \quad j = 0, ..., 10 \tag{2b}$$

$$x_{j+1} = f_D(x_j, u_j, \xi_1, \xi_2), \quad j = 0, ..., 9 \tag{2c}$$

$$Pr(-x_{1,j} - 3.3 \le 0, -x_{2,j} - 3.3 \le 0) \ge 1 - \varepsilon, \quad j = 1, ..., 10 \tag{2d}$$

$$|u_j| \le 2, \quad j = 0, ..., 9 \tag{2e}$$

$$x_{1,0} = 2, \quad x_{2,0} = 0 \tag{2f}$$

$$\xi_1 \sim \mathcal{N}(0, 0.25), \quad \xi_2 \sim U(-0.5, 0.5) \tag{2g}$$

$j = 0, ..., 10$ correspond to $t = 0 \sim 3$ in the differential equation model with the time interval Δt=0.3. $x_{1,0}$ and $x_{2,0}$ in (2f) represent the initial states. ξ_1 and ξ_2 are two random parameters, following Gaussian and uniform distributions, respectively. $1-\varepsilon$ is set to be 0.8 in this illustrating example.

To show the effect of the random parameters ξ_1 and ξ_2, the above optimization is solved deterministically under the nominal scenarios where ξ_1 and ξ_2 are fixed at mean values of zero. The JCC in (2d) is decomposed into two deterministic constraints. CPLEX is used as the solver. The obtained optimal sequence of u_j is the u_D exhibited in Fig. 1. The subscript D is employed for the optimal solution of the deterministic optimization under the nominal scenario. The corresponding optimal results of x_1 and x_2 are $x_{1,DN}$ and $x_{2,DN}$ shown in Fig. 1, respectively. The achieved optimal objective value is -2.6561 which is based on $x_{1,DN}$. In Fig. 1, each solid curve is the x_1 or x_2 calculated based on the optimal u_j sequence and a set of sampled realizations of ξ_1 and ξ_2. Thus, the solid curves in Fig. 1 ($x_{1,DU}$ and $x_{2,DU}$) are x_1 and x_2 calculated based on the optimal solution and under different uncertainty scenarios. Notably, the subscript DU is employed for the x_1 and x_2 based on the deterministic optimal solution and under different uncertainty scenarios. Also, the subscript DN is used for the x_1 and x_2 based on the deterministic optimal solution and under the nominal scenario.

As can be seen from Fig. 1, while $x_{1,DN}$ and $x_{2,DN}$ touch the constraint limit, $x_{1,DU}$ and $x_{2,DU}$ under many uncertainty scenarios violate the constraints significantly. This is because the deterministic illustrating example is solved without considering different uncertainty scenarios. Then, the robustness of the optimal solution is not achieved because of the neglect of uncertainty. Accordingly, the random effects of ξ_1 and ξ_2 has to be considered in the SOCP including (2a)-(2g).

3. Reformulation and empirical approximation

Quantile reformulation of chance constraint

The quantile-based reformulation of the JCC is the basis of the RNN-based approach. To illustrate the quantile-based reformulation, let's consider a JCC given as:

$$Pr(g_i(x, u, \xi) \le 0, i = 1, \cdots, p) \ge 1 - \varepsilon \tag{3}$$

where $x \in \mathbb{R}^{n_x}$ denotes the system state, $u \in \mathbb{R}^{n_u}$ represents the decision variable vector, $\xi \in \mathbb{R}^{n_\xi}$ is the random parameter vector. For each i, $g_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_\xi} \to \mathbb{R}$ is a function. The JCC in (3) ensures that all constraints $g_{i=1,...,p}(x, u, \xi) \le 0$ are satisfied simultaneously to a certain confidence level $1 - \varepsilon$.

As mentioned in Section 1, since joint chance-constrained problems are difficult to solve [5, 31, 32], they are generally solved through approximations. To improve the approximation of the JCC, the quantile-based reformulation for the JCC is employed. The quantile-based reformulation is given as follows. Consider a single constraint function $g(x, u, \xi)$ as a random variable g with cumulative distribution function (CDF) $\phi_g(\gamma) = Pr(g(x, u, \xi) \le \gamma)$, the $1 - \varepsilon$ level quantile of $g(x, u, \xi)$ is defined as [33]:

$$\begin{aligned} &Q^{1-\varepsilon}(g(x, u, \xi)) \\ &= \inf \{\gamma \in \mathbb{R} \mid Pr(g(x, u, \xi) \le \gamma) \ge 1 - \varepsilon\} \\ &= \inf \left\{\gamma \in \mathbb{R} \mid \phi_g(\gamma) \ge 1 - \varepsilon\right\} \end{aligned} \tag{4}$$

According to (4), $Q^{1-\varepsilon}(g(x, u, \xi))$ outputs the minimum γ that satisfies $Pr(g(x, u, \xi) \le \gamma) \ge 1-\varepsilon$, that is, $\phi_g(\gamma) \ge 1-\varepsilon$.

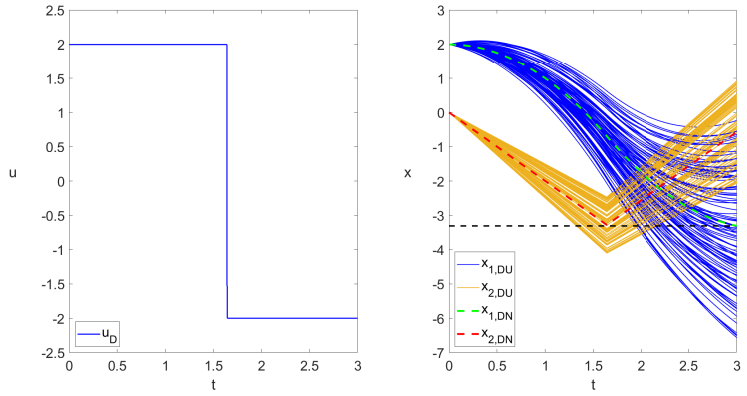

Fig. 1: The optimal u_j sequence of the deterministic illustrating example (u_D), and the corresponding x_1 and x_2. The subscripts DU and DN are used for the results under different uncertainty scenarios and under the nominal scenario, respectively.

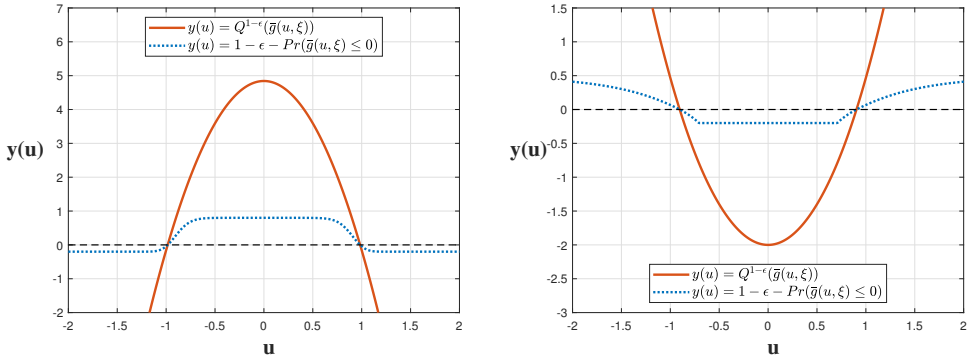

Fig. 2: Comparisons between $1-\varepsilon-Pr(\bar{g}(u,\xi)\leq 0)$ and $Q^{1-\epsilon}(\bar{g}(u,\xi))$ with $1-\varepsilon=0.8$. (a) Single chance constraint with $\bar{g}(u,\xi)=-5u^2+4+\xi$, $\xi\sim\mathcal{N}(0,1)$; (b) JCC with $\bar{g}(u,\xi)=\max\left\{g_1(u,\xi_1),g_2(u,\xi_2)\right\}$, $g_1(u,\xi_1)=1.5\xi_1u^2-3$, $\xi_1\sim\mathcal{N}(0,1)$, $g_2(u,\xi_2)=2\xi_2u^2-2$, $\xi_2\sim U(-2,2)$. $\bar{g}$ here only depends on u and ξ for simplification and the demonstration purpose.

Thus, the following relationship holds:

$$Pr(g(x,u,\xi)\leq\gamma)\geq 1-\varepsilon\Leftrightarrow\phi_g(\gamma)\geq 1-\varepsilon \Leftrightarrow Q^{1-\varepsilon}(g(x,u,\xi))\leq\gamma \tag{5}$$

Based on the above fact, the corresponding single chance constraint $Pr(g(x,u,\xi)\leq 0)\geq 1-\varepsilon$ has quantile reformulation $Q^{1-\varepsilon}(g(x,u,\xi))\leq 0$. To carry out the quantile-based reformulation of JCC in (3), we define $\bar{g}(x,u,\xi)=\max_{i=1,\ldots,p} g_i(x,u,\xi)$. Note that $Pr(\bar{g}(x,u,\xi)\leq 0)$ is equivalent to $Pr(g_i(x,u,\xi)\leq 0, i=1,\cdots,p)$ in (3). Accordingly, the inequality (3) can be equivalently reformulated as the follow:

$$Q^{1-\varepsilon}(\bar{g}(x,u,\xi))\leq 0 \tag{6}$$

To demonstrate the benefit of adopting the above quantile-based reformulation, let's first consider a simple form of $\bar{g}$ which only depends on u and ξ ($\bar{g}(u,\xi)$) for the demonstration purpose. Based on the discussion in the previous paragraph, two equivalent reformulations of the JCC based on $\bar{g}(u,\xi)$ can be obtained: $Q^{1-\varepsilon}(\bar{g}(u,\xi))\leq 0$ and $1-\varepsilon-Pr(\bar{g}(u,\xi)\leq 0)\leq 0$ (it is obtained by moving $Pr(\bar{g}(u,\xi)$ in the JCC $Pr(\bar{g}(u,\xi)\geq 1-\varepsilon$ to the right-hand side). The left-hand-side values of the two inequalities are represented as $y(u)$ since $Q^{1-\varepsilon}(\bar{g}(u,\xi))$ and $1-\varepsilon-Pr(\bar{g}(u,\xi)\leq 0)$ are essentially functions of u. $Q^{1-\varepsilon}(\bar{g}(u,\xi))$ and $1-\varepsilon-Pr(\bar{g}(u,\xi)\leq 0)$ can be approximated by deterministic surrogate models depending on u, for enhancing computational efficiency and tractability. As can be seen from Fig. 2, $Q^{1-\varepsilon}(\bar{g}(u,\xi))$ are convex or concave functions of u. On the contrary, $1-\varepsilon-Pr(\bar{g}(u,\xi)\leq 0)$ are non-convex or non-concave functions of u. Accordingly, since $Q^{1-\varepsilon}(\bar{g}(u,\xi))$ possess better convexity or concavity with less complex shapes than $1-\varepsilon-Pr(\bar{g}(u,\xi)\leq 0)$, it is easier to model $Q^{1-\varepsilon}(\bar{g}(u,\xi))$ than $1-\varepsilon-Pr(\bar{g}(u,\xi)\leq 0)$. The quantile-based reformulation will be used for addressing a JCC with arbitrary number of uncertain constraints.

Empirical approximation

While it is challenging to evaluate the quantile function in (6) for multi-dimensional uncertainty, $Q^{1-\varepsilon}(\bar{g}(x,u,\xi))$ can be approximated and calculated through an empirical method. To carry out the empirical method, a sample set

$\Omega^N = \{\xi_1, ..., \xi_N\}$ is generated. Then, the set of function values $\{\bar{g}(x,u,\xi_1), ..., \bar{g}(x,u,\xi_N)\}$ can be easily evaluated based on fixed x and u. Subsequently, the CDF $\phi_{\bar{g}}(\gamma)$ can be approximated by the empirical CDF:

$$\widetilde{\phi}_{\bar{g}}(\gamma) = \frac{1}{N}\sum_{l=1}^{N} \mathbb{I}(\bar{g}(x,u,\xi_l) \leq \gamma) \tag{7}$$

where N is the number of samples, and $\mathbb{I}(\bar{g}(x,u,\xi_l) \leq \gamma)$ is an indicator function defined as:

$$\mathbb{I}(\bar{g}(x,u,\xi_l) \leq \gamma) = \begin{cases} 0, & \text{for} \quad \bar{g}(x,u,\xi_l) > \gamma \\ 1, & \text{for} \quad \bar{g}(x,u,\xi_l) \leq \gamma \end{cases} \tag{8}$$

Note that using the empirical CDF in (7) to approximate the probability function in (3) is essentially the same as the SAA approach mentioned in Section 1.

According to (4), (5), and (7), the quantile $Q^{1-\varepsilon}(\bar{g}(x,u,\xi))$ can be approximated by the empirical quantile defined as:

$$\begin{aligned} &\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,u,\xi)) \\ &= \inf\left\{\gamma \mid \frac{1}{N}\sum_{l=1}^{N} \mathbb{I}(\bar{g}(x,u,\xi_l) \leq \gamma) \geq 1-\varepsilon\right\} \\ &= \bar{g}_{\lceil M \rceil}(x,u) \end{aligned} \tag{9}$$

where M equals to $(1-\varepsilon)\times N$, and $\bar{g}_{\lceil M \rceil}(x,u)$ represents the $\lceil M \rceil$-th smallest component of $\bar{G}^N$ with respect to given x and u. Thus, the quantile function in (6) can be approximated by the empirical quantile function in (9), and (6) can be reformulated as:

$$\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,u,\xi)) \leq 0 \tag{10}$$

Note that the above sample-based empirical approximation is also applied for the expectation objective in this work.

To have an accurate empirical approximation, the collected samples should approximately cover the entire range of the uncertainty distribution. Additionally, more complex distribution requires more samples. For instance, if the distribution of uncertainty ξ is a mixture Gaussian distribution, more samples are required to have an accurate empirical approximation than if the distribution of uncertainty ξ is just a simple Gaussian distribution. On the other hand, unlike the scenario approach [34], there is no concrete rule to determine the exact number of samples N that must be considered for the above empirical approximation.

4. RNN-based stochastic optimal control

According to the quantile-based reformulation for the JCC mentioned in the previous section, the SOCP including (1a)-(1d) can be reformulated as:

$$\min_{u_j} \quad \mathbb{E}\left[\sum_{j=0}^{K-1} J(x_j,u_j) + J_f(x_K)\right] \tag{11a}$$

$$\text{s.t.} \quad x_{j+1} = f_D(x_j,u_j,\xi) \quad j = 0,..,K-1 \tag{11b}$$

$$\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j,u_{j-1},\xi)) \leq 0 \quad j = 1,..,K \tag{11c}$$

$$x_0 = x_{t_0} \tag{11d}$$

where $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j,u_{j-1},\xi))$ denotes the empirical QF at a certain sampling instant j. $\bar{g}(x_j,u_{j-1},\xi) = \max_{i=1,...,p} g_i(x_j,u_{j-1},\xi)$. Since $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j,u_{j-1},\xi))$ is still a stochastic and complex function of u_j, RNN-based approaches can be exploited to approximate $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j,u_{j-1},\xi))$ to reduce the complexity and make the above SOCP deterministically solvable.

In the subsequent subsections, the RNN is first illustrated. Then, the proposed approach based on the RNN and the quantile-based reformulation is presented. Finally, the illustrating example under uncertainty mentioned in Section 2 is solved in Section 4.3.

4.1. Recurrent neural network

The recurrent neural network (RNN) is a variant of neural networks, which is capable of learning sequential data. An RNN can model a discrete-time dynamic system through the feedback of the hidden state from the previous time step to the current time step. A vanilla RNN is shown in Fig. 3 which is an unfolded representation of a vanilla RNN for modelling a dynamic system with 3 time steps. This vanilla RNN has 1 hidden layer with 3 neurons. The hidden layer at each time step is also called the cell. Every cell has the same hidden layer structure, set of weights, and set of biases. $u_{j=0,1,2}$ and $x_{j=1,2,3}$ are the input sequence and the output sequence of the RNN, respectively, which are 1-dimensional sequences with 3 time steps individually. h_0 is the initial hidden state and $h_1 \sim h_3$ are hidden states computed from the hidden layer at different time steps. $h_0 \sim h_3$ are 3-element vectors because there are 3 neurons in the hidden layer. h_0 is generally set as a zero vector [35]. At the time step $j+1$ ($j = 0,1,2$), the inputs for a neuron in the hidden layer are the hidden state from the previous time step h_j and the input u_j. A vanilla RNN can be interpreted as the equations shown below:

$$h_{j+1} = \sigma_h(W_h u_j + U_h h_j + b_h) \tag{12a}$$

$$x_{j+1} = \sigma_x(W_x h_{j+1} + b_x) \tag{12b}$$

where W_h and U_h are the weight matrices for neurons in the hidden layer. W_x is the weight matrix for neurons in the output layer. b_h and b_x are the bias vectors for neurons in the hidden and output layers, respectively. σ_h and σ_x are the activation functions for neurons in the hidden and output layers, respectively. The subscripts h and x are indices for neurons in the hidden and output layers, respectively.

The vanilla RNN can work well with moderate sequence size, but it fails to retain information when the sequence given is long. This is known as the short-term memory problem of the vanilla RNN [36]. To tackle such a problem, more advanced RNNs have been developed. The most popular advanced RNNs are the Long short-term memory (LSTM) and the Gated Recurrent Unit (GRU). The LSTM is more

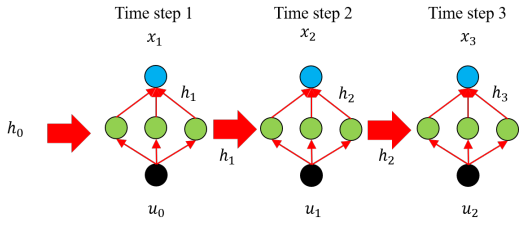

Fig. 3: Schematic diagram of a vanilla recurrent neural network

preferable than the GRU when prediction accuracy is critical since the LSTM has more gates and parameters [37]. Thus, the LSTM is used in this study. A schematic diagram for illustrating an LSTM cell is shown in Fig. 4. The difference between an LSTM and a vanilla RNN is that the structure of an LSTM cell is much more complex than of a vanilla RNN cell. More specifically, there are several gates composed of several neurons in an LSTM cell to add or remove information in states. A more detailed explanation about the LSTM is in the Supplementary Materials.

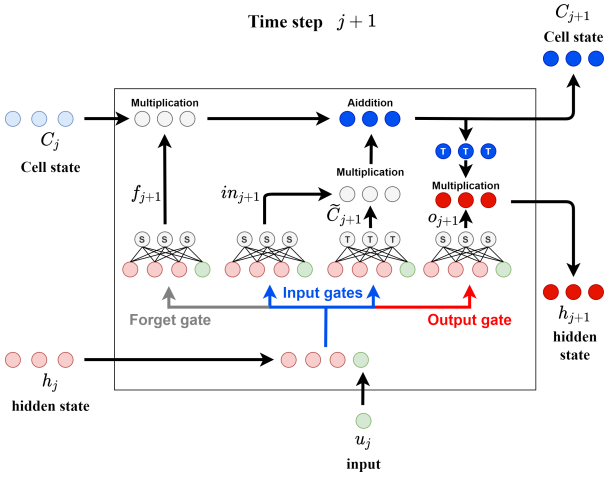

Fig. 4: Schematic diagram for illustrating an LSTM cell

4.2. RNN-based approach

To reduce the solution complexity of the optimization including (11a)-(11d), the LSTM is exploited to generate the surrogate model for handling the dynamic model in (11b). Meanwhile, this surrogate model is also used to predict quantile values for the inequality (11c). The illustration of the LSTM for the quantile value prediction is shown in Fig. 5. where h_0 and C_0 are the initial hidden state and the initial cell state, respectively. These initial states are set to be zero vectors. $h_1 \sim h_K$ and $C_1 \sim C_K$ are the hidden states and the cell states computed by the LSTM at corresponding sampling instants. At each sampling instant $j+1$, the predicted quantile value $\hat{Q}_{j+1}^{1-\epsilon}$ is output from the output layer employing the corresponding hidden state from the LSTM cell as the input.

A similar method can be used to handle the expectation objective: we use the same LSTM network structure and

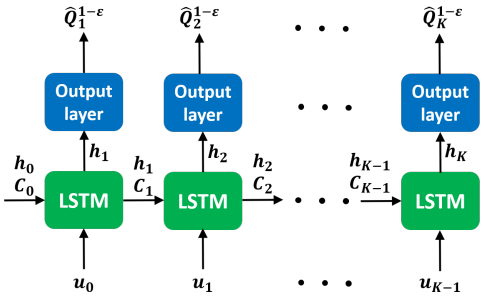

Fig. 5: Schematic diagram of the LSTM for the quantile value prediction

use the output of each LSTM cell as the approximated expectation at each sampling instant. The quantile function and the expectation can be approximated through the same LSTM model or two different LSTM models. For achieving higher surrogate model accuracy, the expectations in (11a) were predicted by another LSTM model (the second LSTM model).

The two LSTM models are further incorporated into the SOCP including (11a)-(11d). The SOCP incorporating the two LSTM models is given as:

$$\min_{u_j} \quad \sum_{j=0}^{K} \hat{E}_j \tag{13a}$$

$$\text{s.t.} \quad \hat{Q}_j^{1-\epsilon} \leq 0 \quad j = 1, ..., K \tag{13b}$$

$$\hat{Q}_{j=1,...,K}^{1-\epsilon} = L_1(u_{j=0,...,K-1}) \tag{13c}$$

$$\hat{E}_{j=1,...,K} = L_2(u_{j=0,...,K-1}) \tag{13d}$$

where $\hat{E}_j$ is the expected value predicted from the second LSTM at a certain sampling instant j. $\hat{E}_{j=0}$ is a constant based on initial conditions. $\hat{Q}_j^{1-\epsilon}$ is the quantile value predicted from the first LSTM at a certain sampling instant j. L_1 and L_2 are the first and second LSTM models that individually take the sequence of u_j as inputs. The above optimization is an optimization problem involving two LSTM models to compute $\hat{E}_j$ and $\hat{Q}_j^{1-\epsilon}$. According to the above optimization formulation, the SOCP can be solved deterministically due to the deterministic LSTM approximation. The training set for the first LSTM is generated by computing quantile values based on the randomly selected input sequences. Similarly, the training set for the second LSTM is produced by calculating expected values based on the randomly selected input sequences. More details about the generation of training sets are elucidated in the following sections and the Supplementary Materials.

5. Illustrating example revisited

The illustrating problem studied in Section 2 is solved by utilizing the RNN-based approach and demonstrated in the following subsections.

4.3.1. Problem reformulation

The JCC in (2d) can be reformulated equivalently as the discretized quantile-based inequality which is given as:

$$\widetilde{Q}_{I,j}^{1-\varepsilon}(\bar{g}_{I,j}) \leq 0 \quad j = 1, ..., 10 \tag{14}$$

where $\bar{g}_{I,j} = \max\left\{g_{1,I,j}, g_{2,I,j}\right\}$. $g_{1,I,j}$ and $g_{2,I,j}$ denotes $-x_{1,j} - 3.3$ and $-x_{2,j} - 3.3$, respectively.

Calculation of expected values, $x_{1,j}$, $x_{2,j}$, $\bar{g}_{I,j}$, and quantile values ($\widetilde{Q}_{I,j}^{1-\varepsilon}(\bar{g}_{I,j})$) in (2a), (2c), (2d), and (14) are elaborated in the Supplementary Materials. Moreover, the calculated expected values and quantile values are used for the LSTM training mentioned in the following subsection.

4.3.2. LSTM model training

Two LSTMs are employed to address (2a) and (14). The first LSTM is used to predict the quantile values in (14) based on different sequences of u_j. In total, 8×10^4, 10^4, and 10^4 samples are generated for training, validation and testing, respectively. The overall time for sample generation is around 1200 seconds. Each sample is composed of a sequence of quantile values paired with the corresponding sequence of u_j. This LSTM has the same structure as the one shown in Fig. 5. Also, this LSTM has 50 units (dimensions of the hidden state and the cell state are equal to 50 individually) in each cell. Then, another LSTM (the second LSTM) is exploited to predict the expected values in (2a) for approximating (2a). The second LSTM has the same structure as the first one. Same sample number setting is used for training, validating, and testing the second LSTM. The overall time for sample generation for the second LSTM is also around 1200 seconds. The architectures of the above two LSTMs are determined through the 10-fold cross-validations based on the training sets first, and then the LSTMs are trained on the entire training sets with the use of the validation sets for the early stopping. The two LSTMs are constructed and trained utilizing Keras [38]. The training time for each LSTM is around 500 seconds. The mean absolute percentage errors (MAPEs) of the two LSTMs based on the testing sets are below 2%. More details are shown in the Supplementary Materials.

Afterwards, the two trained LSTM models are incorporated into the discretized optimization formulation of the illustrating problem. Such optimization involving the two LSTM models is solved using IPOPT 0.3.0 in the Python environment, with solution time being 1.7 seconds. Meanwhile, the automatic differentiation (AD) for evaluating the gradient of the objective function and Jacobian of the constraint, is carried out via TensorFlow [39]. The formulation of the optimization involving the two LSTM models and the related details are shown in the Supplementary Materials.

4.3.3. Results

The attained optimal results are shown in Fig. 6. Note that u_R in Fig. 6 is the optimal sequence of u_j obtained from the proposed RNN-based approach. The subscript R is used for the optimal solution attained from the proposed RNN-based approach. The obtained optimal results satisfy the JCC in (2d). Based on the attained optimal u_j sequence, the true objective value and the objective value according to the computation of the second LSTM model are -0.5866 and -0.5872, respectively. More details about the above-mentioned results are elaborated in the Supplementary Materials.

As can be seen from Fig. 6, the majorities of $x_{1,RU}$ and $x_{2,RU}$ under different uncertainty scenarios are above the lower bound -3.3 since $x_{1,RN}$ and $x_{2,RN}$ are away from the lower bound. More specifically, $x_{1,RN}$ and $x_{2,RN}$ are gained under the nominal scenario which appears with higher probability than other scenarios. Accordingly, $x_{1,RN}$ and $x_{2,RN}$ should be higher than the lower bound significantly as they are shown in Fig. 6, to ensure the outcomes under different uncertainty scenarios being feasible with the required probability of 0.8.

5. Case study: hydrodesulphurisation process

In this case study, the RNN-based approach is applied to the problem of the hydrodesulphurisation process (HDS) taken from [17]. Both SOCP and SMPC implementation of the HDS are handled through the proposed approach, and they are illustrated and discussed comprehensively in the following sections.

The studied HDS is part of a large refinery process, which is exploited to remove sulphur from a hydrocarbon flow. Sulphur is removed from the hydrocarbon flow through a two-stage fixed bed reactor. The hydrocarbon flow is mixed with hydrogen gas in the reactor. Sulphur in the hydrocarbon flow reacts with hydrogen to be hydrogen sulfide. A schematic diagram of the HDS is exhibited in Fig. 7. As can be seen from Fig. 7, there are 3 hydrogen feed streams, namely F_1, F_2, and F_3. The 3 streams are mixed and fed to a compressor (C-1), to keep the inlet pressure of the reactor constant. The hydrocarbon flow is denoted as F_{HC}. The reactor for hydrodesulphurisation is a two-stage reactor. R-1 and R-2 are the first stage reactor and the second stage reactor, respectively. The outlet flow of R-2 (F_7) is fed into a flash tank (T-1) to separate hydrocarbons from hydrogen and sulfide gas. Then, the separated hydrocarbons are collected from the product flow F_8. The flow at the top of T-1 (F_9) is recycled to R-1 and R-2 partially, and the rest leaves the HDS through a purge stream denoted as F_{10}. The operation of the HDS should satisfy the following constraints: the hydrogen mole fractions in both R-1 and R-2 (X_{H2}) should be maintained above 0.7 to avoid catalyst deactivation; the hydrogen mole fraction in stream F_5 (X_5) should be kept above 0.9 because of the requirement of C-1.

5.1. Model

The HDS is modelled mathematically based on the following assumptions for simplification: 1) Temperatures in R-1 and R-2 are controlled perfectly. 2) Pressures in all the streams and units in the HDS are controlled perfectly. 3) The hydrodesulphurisation reaction can be described by the first-order model. 4) The flash tank T-1 can separate hydrocarbons from hydrogen and sulfide perfectly. 5) R-1

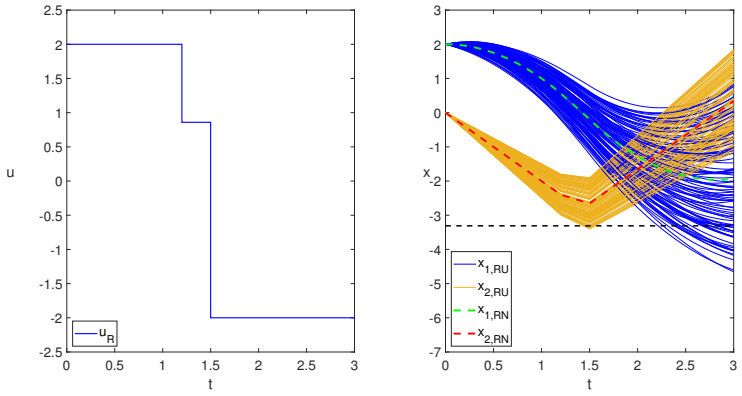

Fig. 6: The optimal sequence of u_j obtained from the proposed RNN-based approach (u_R), and the corresponding x_1 and x_2. The subscripts RU and RN are used for the results under different uncertainty scenarios and under the nominal scenario, respectively.

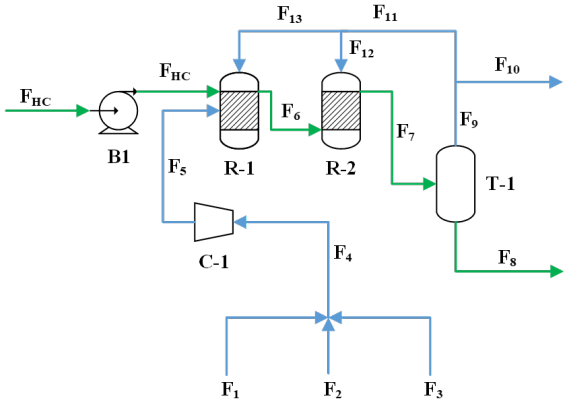

Fig. 7: Schematic diagram of the studied hydrodesulphurisation process

and R-2 can be modelled as 1 reactor. According to these assumptions, only mass and component balances should be taken into account for modelling the HDS process.

The optimal control objective is to minimize the cost of hydrogen usage from streams F_1 and F_2 under the above-mentioned constraints under uncertainty. The SOCP of this case study is formulated as:

$$\min_{\dot{F}_1,\dot{F}_2,\dot{F}_{10}} \quad \int_{t_0}^{t_f} C_{H4}X_1\dot{F}_1 + C_{H3}X_2\dot{F}_2 dt \tag{15a}$$

$$\text{s.t.} \quad \tau\frac{d\dot{F}_x^{H2}}{dt} + \dot{F}_x^{H2} = \dot{F}_{HC}\rho \tag{15b}$$

$$\frac{VP}{ZR_gT}\frac{dX_{H2}}{dt} = \dot{F}_5X_5 - \dot{F}_{10}X_{H2} - \dot{F}_x^{H2} \tag{15c}$$

$$\dot{F}_5 = \dot{F}_{10} + \dot{F}_x^{H2} \tag{15d}$$

$$\dot{F}_5 = \dot{F}_1 + \dot{F}_2 + \dot{F}_3 \tag{15e}$$

$$\dot{F}_5X_5 = \dot{F}_1X_1 + \dot{F}_2X_2 + \dot{F}_3X_3 \tag{15f}$$

$$Pr(X_{H2} \geq 0.7, X_5 \geq 0.9) \geq 1-\varepsilon \tag{15g}$$

$$0 \leq \dot{F}_1 \leq 1400 \tag{15h}$$

$$0 \leq \dot{F}_2 \leq 790 \tag{15i}$$

$$0 \leq \dot{F}_{10} \leq 1500 \tag{15j}$$

t is time (unit: h). t_0 is the current sampling instant that the current system state is acquired. t_f is the end sampling instant in the above SOCP. Note that the values of t_0 and t_f are varied in the SOCP at different sampling instants, and $t_f - t_0 = 2$. C_{H4} and C_{H3} are unit prices of hydrogen from streams F_1 and F_2, respectively. $\dot{F}_1$, $\dot{F}_2$, and $\dot{F}_{10}$ are the mole flow rates of the streams F_1, F_2, and F_{10}, respectively. Also, $\dot{F}_1$, $\dot{F}_2$, and $\dot{F}_{10}$ are decision variables of the SOCP. Since the hydrodesulphurisation reaction model is assumed to be first-order, the hydrogen consumption rate $\dot{F}_x^{H2}$ in the reactor can be described by (15b). τ is the time constant of the reaction. $\dot{F}_{HC}$ is the volume flow rate of hydrocarbons fed into the reactor. ρ is a random parameter following the Gaussian distribution. Equation (15c) is used to calculate the hydrogen mole fraction in the reactor (X_{H2}). V is the reactor volume. P denotes the pressure inside the reactor. Z is the compressibility factor. R_g is the ideal gas constant. T is the temperature inside the reactor. $\dot{F}_5$ is the mole flow rate of the stream F_5. X_5 is the hydrogen mole fraction in stream F_5. The mass balance over the reactor is expressed as (15d). Equation (15e) is exploited to calculate $\dot{F}_5$. The component balance of hydrogen can be described by (15f). X_3 is a random parameter following the Gaussian distribution. $1-\varepsilon$ is the user-defined confidence level, which is set to be 0.8 in this case study. (15h)-(15j) are the bounds for the decision variables of the SOCP. The values of parameters in the SOCP are shown in Table 1.

The SOCP and SMPC implementation of the HDS are handled by using both proposed RNN-based approach and the sample-based approach modified from Moen's work [15]. Notably, the original sample-based approach in the literature exploited the individual chance constraints to approximate the JCC that transforms the original joint chance-constrained HDS SOCP to be an individual chance-constrained problem. Accordingly, to have fair comparisons between the sample-based approach and the proposed approach, the sample-based method from the literature is modified to be directly

applicable to joint chance-constrained problems. In the modified sample-based approach, the joint constraint satisfaction probability (JCSP) at each sampling instant is computed through the following steps: 1) The HDS model is simulated multiple times based on different realizations of uncertain parameters to obtain different values of X_{H2} and X_5; 2) The JCSP is computed using SAA based on the X_{H2} and X_5 gained from the previous step. The multiple simulations of the HDS model are treated as a black-box model to compute the JCSP, and this black-box model is incorporated into the HDS SOCP. Thus, the HDS SOCP becomes a black-box joint chance-constrained optimization problem while using the modified sample-based method. The modified sample-based approach is essentially based on the idea of the Monte Carlo simulation which is the same as the main idea of the original sample-based method from the literature. For simplification, the sample-based method mentioned in the following parts of this paper refers to the modified sample-based method.

Table 1: Values of parameters in the SOCP including (15a)-(15j)

Parameter	Value	Unit
C_{H4}	88.1	$€/Mmol$
C_{H3}	77	$€/Mmol$
τ	0.3	h
$\dot{F}_{HC}$	102	m^3/h
V	100	m^3
P	68.901	bar
Z	1	-
R_g	0.08314	$(m^3 bar)/(K kmol)$
T	623.15	K
X_1	0.991	-
X_2	0.931	-
$\dot{F}_x^{H2}$ at $t=0$	682.5	$kmol/h$
X_{H2} at $t=0$	0.9	-

[a] $\rho \sim \mathcal{N}(12.6, 0.4)$ (unit: $kmol/m^3$)
[b] $X_3 \sim \mathcal{N}(0.85, 0.013)$

5.2. Results and discussion

5.2.1. Deterministic optimal control

To show the impacts of the random parameters ρ and X_3 on the SOCP of the HDS, the optimization including (15a)-(15j) is solved deterministically under the nominal scenario that ρ and X_3 are fixed at their mean values (12.6 and 0.85, respectively). Meanwhile, the JCC in (15g) is decomposed into two deterministic constraints for X_5 and X_{H2}. Such deterministic optimization is solved numerically through the trapezoidal rule method with 20 time intervals. More specifically, the involved integral objective function and the process model are approximated via the trapezoidal rule method with 20 intervals. Meanwhile, IPOPT is used as the solver. The obtained optimal MVs are $\dot{F}_{1,D}$, $\dot{F}_{2,D}$, and $\dot{F}_{10,D}$ shown in Fig. 8. The corresponding optimal results of X_5 and X_{H2} are $X_{5,DN}$ and $X_{H2,DN}$ illustrated in Fig. 8, respectively. The achieved optimal objective value is 106.8659 € which is based on $\dot{F}_{1,D}$ and $\dot{F}_{2,D}$. In Fig. 8, each solid curve is the X_5 or X_{H2} calculated based on the optimal sequence of MVs and a pair of ρ and X_3 realizations. Thus, the solid curves in Fig. 8 ($X_{5,DU}$ and $X_{H2,DU}$) are X_5 and X_{H2} calculated based on the optimal solution and under different uncertainty scenarios. Note that the subscript DU is employed for the X_5 and X_{H2} based on the deterministic optimal solution and under different uncertainty scenarios. Additionally, the subscript DN is employed for the X_5 and X_{H2} based on the deterministic optimal solution and under the nominal scenario. According to the results shown in Fig. 8, $X_{5,DU}$ and $X_{H2,DU}$ violate the constraints under many uncertainty scenarios. This is because the deterministic optimal control problem under the nominal scenario is solved without considering different uncertainty scenarios. Therefore, X_5 and X_{H2} under the nominal scenario ($X_{5,DN}$ and $X_{H2,DN}$) which appears with higher probability than other scenarios, are allowed to activate the constraints to minimize the objective value. Then, the robustness of the optimal solution is reduced because of the neglect of uncertainty. Based on the above discussion, the random effects of ρ and X_3 are not negligible for addressing the stochastic problem in this case study.

5.2.2. Stochastic optimal control

The SOCP of the HDS is solved using the proposed RNN-based approach. While using the proposed approach, the SOCP including (15a)-(15j) should be first discretized. The details of the discretization and the discretized SOCP formulation are illustrated in the Supplementary Materials. The MAPEs of the LSTM model used for handling the SOCP of the HDS are below 5% (based on the testing set). More details about this LSTM and the optimization involving this LSTM are elucidated in the Supplementary Materials. By utilizing the proposed approach to solve the SOCP of the HDS, the attained optimal results are shown in Fig. 9. The optimal objective value is 126.2254 € and the solution time is 19.7 seconds. The optimal results satisfy the JCC for all sampling instants. More details about the results are illustrated in the Supplementary Materials. Note that $\dot{F}_{1,R}$, $\dot{F}_{2,R}$, and $\dot{F}_{10,R}$ in Fig. 9 are exactly the optimal sequence of MVs obtained from the proposed RNN-based approach and the subscript R is used for the optimal solution attained from the RNN-based approach.

As can be seen from Fig. 9, the majorities of $X_{5,RU}$ and $X_{H2,RU}$ under different uncertainty scenarios are above corresponding lower bounds which are 0.9 and 0.7, respectively. This is because $X_{5,RN}$ and $X_{H2,RN}$ gained under the nominal scenario which appears with higher probability than other scenarios, are away from the lower bounds. Therefore, the obtained outcomes are feasible with the required confidence level of 0.8. Also, since $X_{5,RN}$ and $X_{H2,RN}$ are higher than the constraints significantly to ensure the solution robustness, the objective value (126.2254 €) is higher than the objective value obtained from the deterministic optimal control problem of this case study (106.8659 €).

For the SOCP of the HDS, the optimal solution obtained from the proposed approach is compared with the optimal solution attained from the sample-based approach modified

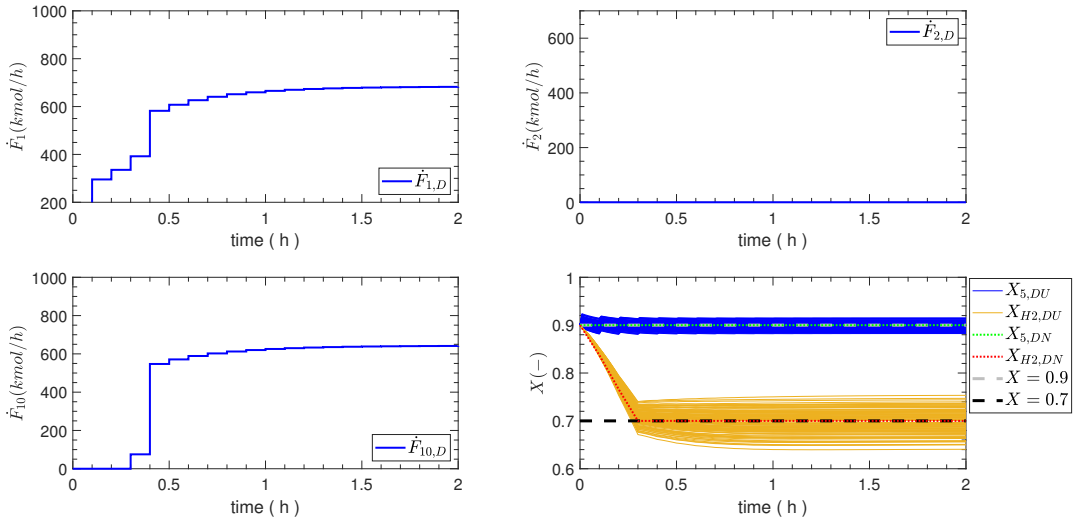

Fig. 8: The optimal results of the deterministic optimal control of the HDS. The subscripts DU and DN are used for the results under different uncertainty scenarios and under the nominal scenario, respectively.

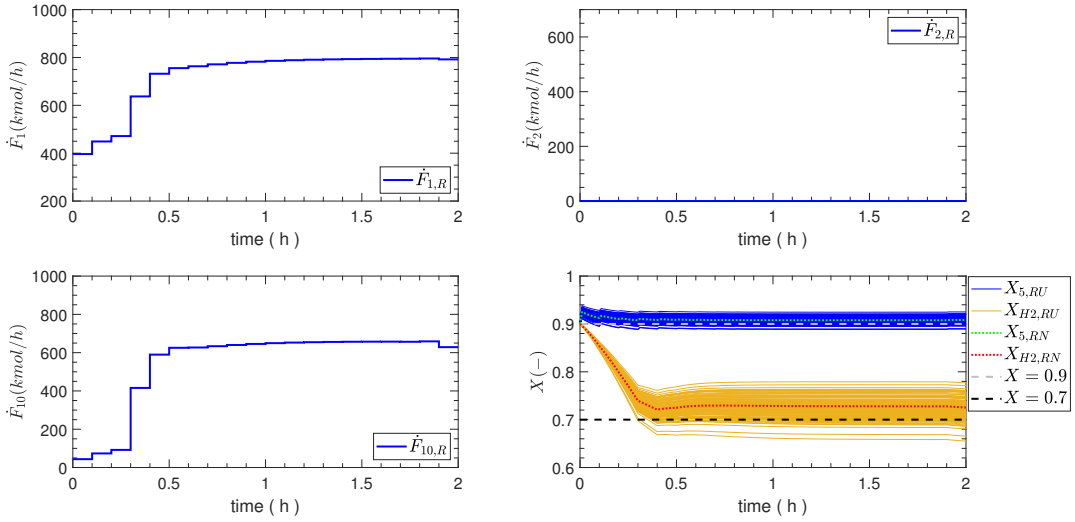

Fig. 9: The optimal results of the SOCP of the HDS gained from the proposed approach. The subscripts RU and RN are used for the results under different uncertainty scenarios and under the nominal scenario, respectively.

from Moen's study [15]. The comparison is organized in Table 2.

Table 2: Comparison between the results of the SOCP of the HDS

	Objective value (€)	Solution time (s)
Proposed approach	126.2254	19.7
Sample-based method with 1000 Monte Carlo simulations	128.7322	317.5

According to Table 2, the objective value from the proposed method is only slightly better than from the sample-based method. However, the proposed approach has a much shorter solution time than the sample-based method. This is because the sample-based method has to run 1000 Monte Carlo simulations to compute the JCSP during the optimization. Therefore, a lot of time is spent on the Monte Carlo simulations that restrict the efficiency of the sample-based method. As to the proposed approach, the stochastic process model and the JCC in the original SOCP are handled by the incorporated LSTM. Through this method, the original stochastic optimization model can be approximated by the LSTM surrogate model to reduce the complexity and be deterministically solvable.

Since solving a SOCP is the element of SMPC implementation, studying the solution time of solving a SOCP is a good starting method to examine the efficiency of

a control approach. Therefore, we firstly investigated the solution times of the above two methods for solving the SOCP to deduce the performance of the two methods for the SMPC implementation. Then, we confirm the actual control performance of the two methods by conducting the SMPC experiments elucidated below.

6. SMPC implementation

To deal with the SMPC problem using the proposed RNN-based approach, the incorporated LSTM model in the proposed approach should be improved. More specifically, since the LSTM given by Fig. 5 only uses the sequence of u_j as input, it can only be used to solve the SOCP with fixed initial conditions. However, when SMPC is implemented, SOCP is based on different initial conditions at different sampling instants. To this end, a hybrid model consisting of two feed-forward neural networks and one LSTM (NN-LSTM) is proposed to capture the dynamics between quantile values, u_j, and different initial conditions. In the NN-LSTM, the initial conditions are transformed to be the initial hidden and cell states of the LSTM to contribute initial information of system dynamics. Accordingly, the two neural networks are employed to learn the non-linear transformations from the initial conditions to the initial hidden and cell states of the LSTM. The nonlinear transformations through the neural networks are necessary since the dimensions of the initial conditions may not match the dimensions of the initial hidden and cell states of the LSTM model. Finally, the NN-LSTM is incorporated into the SOCP for the SMPC implementation in this study. More details are elaborated in the following subsection.

6.1. SMPC implementation of the HDS case study

In this case study, the NN-LSTM used for the SMPC implementation is a stacked NN-LSTM with the structure shown in Fig. 10. According to Fig. 10, the initial conditions are first fed into two neural networks (NN_1 and NN_2) to generate the initial hidden state h_0^1 and the initial cell state C_0^1. Each neural network only has 1 input layer and 1 hidden layer. There are 50 neurons in the hidden layer of each neural network. Thus, the dimensions of h_0^1 and C_0^1 are both equal to 50. Then, h_0^1 and C_0^1 are fed to the first LSTM cell in the first LSTM layer. In the first LSTM layer, h_{j+1}^1 and C_{j+1}^1 are computed based on h_j^1, C_j^1, $\dot{F}_{1,j}$, $\dot{F}_{2,j}$, and $\dot{F}_{10,j}$. The dimensions of h_{j+1}^1 and C_{j+1}^1 are both equal to 50. In the second LSTM layer, h_0^2 and C_0^2 are set to be zero vectors, and their dimensions are both equal to 30. Meanwhile, h_{j+1}^2 and C_{j+1}^2 are computed based on h_j^2, C_j^2, and h_{j+1}^1. The dimensions of h_{j+1}^2 and C_{j+1}^2 are both equal to 30. For each sampling instant $j+1$, the quantile value ($\hat{Q}_{H,j+1}^{1-\varepsilon}$) is computed from the output layer by employing h_{j+1}^2 as the input.

Quantile values corresponding to different input sequences and different initial conditions are generated as the training data for the stacked NN-LSTM. Then the stacked NN-LSTM is trained to learn the dynamics between control input sequences, initial conditions, and quantile values. This enables the modeling of SOCP in each moving window of the SMPC implementation. More details about the data generation for the stacked NN-LSTM are explained in the Supplementary Materials.

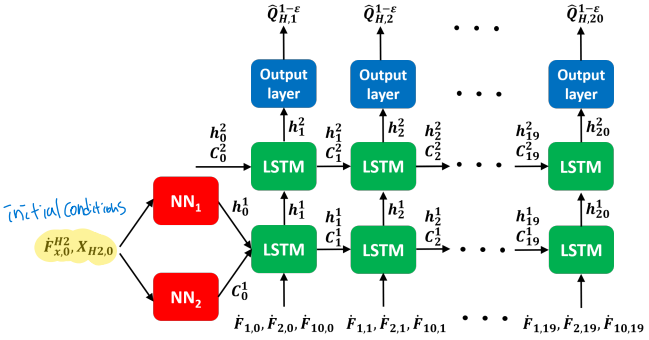

Fig. 10: Schematic diagram of the NN-LSTM used for the SMPC implementation of the HDS case study

Subsequently, the stacked NN-LSTM is incorporated into the discretized SOCP of the HDS after training. The MAPEs of this NN-LSTM are below 5% (based on the testing set). More details about this NN-LSTM and the SOCP formulation involving this NN-LSTM are elucidated in the Supplementary Materials. For one SMPC implementation of the HDS, the SOCP involving the NN-LSTM is solved repeatedly at different sampling instants with different initial conditions. More details are explained in the Supplementary Materials. The SMPC implementation is conducted 1000 times repeatedly. Each SMPC implementation is executed with a realization of ρ and a realization of X_3 at each sampling instant. The corresponding results based on the 1000 SMPC executions are shown in the Supplementary Materials. Also, 100 SMPC results are randomly selected from the 1000 SMPC executions and illustrated in Fig. 11. The reason for only choosing 100 results is to avoid too many overlapping curves shown in Fig. 11 because the figure will be difficult to interpret with too many overlapping curves. Moreover, the distribution of the objective values gained from the 1000 implementations is shown in Fig. 12. The mean value of the distribution of the objective values is 118.7938 €.

The attained SMPC results are feasible with the probabilities higher than the required level of 0.8 for all sampling instants. Also, as can be seen from Fig. 12, all the objective values attained from the SMPC executions are lower than the objective value obtained from the SOCP. This is because the optimal MVs are updated based on the obtained system state at each sampling instant during each SMPC implementation. However, the SOCP is solved only based on a set of initial conditions. Therefore, the SMPC can achieve a better objective value under uncertainty because more state information is available during the SMPC execution.

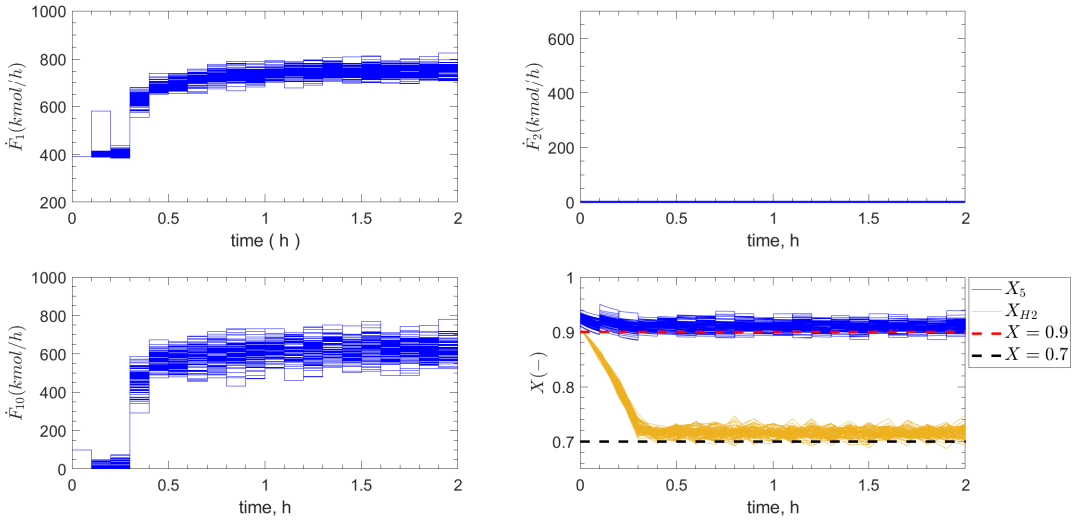

Fig. 11: 100 SMPC results of the HDS, which are attained by exploiting the proposed RNN-based approach

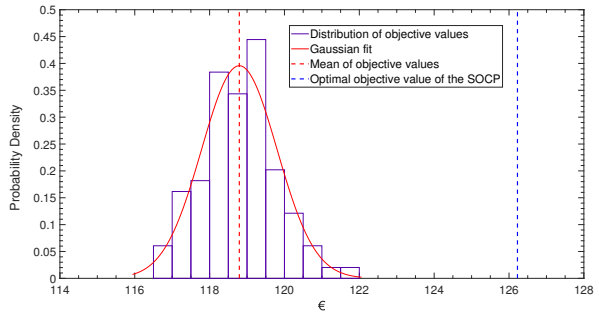

Fig. 12: The distribution of the objective values gained from the 1000 SMPC implementations

The SMPC results attained from the proposed approach are compared with the SMPC results gained from the sample-based method modified from Moen's work [15]. The comparison is organized in Table 3. One remark here is that the solution time corresponds to a method in Table 3 is the overall time of the SMPC implementation for 20 sampling instants.

Table 3: Comparison between the results of the SMPC of the HDS

	Objective value (€)	Solution time (s)
Proposed approach	118.7938	395
Sample-based method with 1000 Monte Carlo simulations	126.6641	12193
Sample-based method with 100 Monte Carlo simulations	126.6772	2949

According to Table 3, the presented approach can obtain a better objective value in a shorter solution time than the sample-based approach. Although the sample-based approach can be accelerated significantly by reducing the number of Monte Carlo simulations (a minimum of 100 Monte Carlo simulations are required to achieve satisfactory SMPC results), it still requires a longer solution time. Thus, the proposed approach is a more attractive method to address SMPC problems.

Finally, based on the above discussion, the proposed RNN-based approach can efficiently solve a joint chance-constrained SMPC problem with satisfactory constraint satisfaction probability and solution quality. Moreover, the proposed approach is applicable for both linear and nonlinear SMPC problems because the employed NN-LSTM surrogate model is capable of handling both linear and nonlinear models. Although generating data for the NN-LSTM and training the NN-LSTM require a lot of time, these time-consuming steps do not hinder the application of the proposed approach to SMPC. This is because these time-consuming steps are completed offline before executing SMPC. In other words, the proposed approach is applied to an SMPC after the NN-LSTM is trained.

7. Conclusion

A new approach based on RNN is proposed to solve stochastic optimal control and stochastic model predictive control problems with JCCs. In the proposed method, the quantile-based reformulation is applied to the JCC and the quantile function is further approximated by an LSTM-based surrogate model. To handle SMPC, NN-LSTM can be embedded into the SOCP, where a hybrid model consisting of feed-forward neural networks and LSTM takes initial conditions and control sequence as input. When SMPC is executed with this method, the SOCP is repeatedly solved at different sampling instants based on the updated initial states.

The results show that the proposed approach can obtain the solution satisfying the confidence level effectively. Compared with the sample-based method modified from the literature, the proposed approach can achieve faster and better solutions for both SOCP and SMPC implementation. Furthermore, broad flexibility is also an important feature of the proposed approach. The approach presented in this work, can be applied to both, linear and nonlinear SMPCs with any number of JCCs.

This work can be further investigated with the following research directions. First, uncertainty distributions may be not available, and a data-driven distributionally robust framework is a possible approach. Second, in the SMPC application, we only studied the open-loop prediction model. A closed-loop prediction model with control policy instead of action involved will be one of the future works.

Acknowledgment

The authors gratefully acknowledge the financial support from the University Research Awards program of Imperial Oil and the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] Z. Wu, H. Durand, and P. D. Christofides, "Safe economic model predictive control of nonlinear systems," *Syst. Control. Lett.*, vol. 118, pp. 69–76, 2018.

[2] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems Magazine*, vol. 36, no. 6, pp. 30–44, 2016.

[3] P. Li, M. Wendt, and G. Wozny, "A probabilistically constrained model predictive controller," *Automatica*, vol. 38, no. 7, pp. 1171–1176, 2002.

[4] M. Ono and B. C. Williams, "Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint," in *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 3427–3432.

[5] W. Van Ackooij, R. Zorgati, R. Henrion, and A. Möller, "Chance constrained programming and its applications to energy management," *Stochastic Optimization—Seeing the Optimal for the Uncertain*, pp. 291–320, 2011.

[6] A. Geletu, M. Klöppel, A. Hoffmann, and P. Li, "A tractable approximation of non-convex chance constrained optimization with non-gaussian uncertainties," *Eng. Optimiz.*, vol. 47, no. 4, pp. 495–520, 2015.

[7] B. K. Pagnoncelli, S. Ahmed, and A. Shapiro, "Sample average approximation method for chance constrained programming: theory and applications," *J. Optim. Theory Appl.*, vol. 142, no. 2, pp. 399–416, 2009.

[8] Y. Yuan, Z. Li, and B. Huang, "Robust optimization approximation for joint chance constrained optimization problem," *J. Global Optim.*, vol. 67, no. 4, pp. 805–827, 2017.

[9] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust optimization*. Princeton University Press, 2009, vol. 28.

[10] Z. Li, R. Ding, and C. A. Floudas, "A comparative theoretical and computational study on robust counterpart optimization: I. robust linear optimization and robust mixed integer linear optimization," *Ind. Eng. Chem. Res.*, vol. 50, no. 18, pp. 10 567–10 603, 2011.

[11] A. Nemirovski and A. Shapiro, "Scenario approximations of chance constraints," in *Probabilistic and randomized methods for design under uncertainty*. Springer, 2006, pp. 3–47.

[12] M. Cannon, B. Kouvaritakis, S. V. Raković, and Q. Cheng, "Stochastic tubes in model predictive control with probabilistic constraints," *IEEE Trans. Automat. Contr.*, vol. 56, no. 1, pp. 194–200, 2011.

[13] T. A. N. Heirung, J. A. Paulson, J. O'Leary, and A. Mesbah, "Stochastic model predictive control—how does it work?" *Comput. Chem. Eng.*, vol. 114, pp. 158–170, 2018.

[14] I. Batina, *Model predictive control for stochastic systems by randomized algorithms*. Citeseer, 2004.

[15] M. A. Moen, "Stochastic optimisation," 2015.

[16] G. C. Calafiore and L. Fagiano, "Robust model predictive control via scenario optimization," *IEEE Trans. Automat. Contr.*, vol. 58, no. 1, pp. 219–224, 2012.

[17] D. Navia, D. Sarabia, G. Gutiérrez, F. Cubillos, and C. de Prada, "A comparison between two methods of stochastic optimization for a dynamic hydrogen consuming plant," *Comput. Chem. Eng.*, vol. 63, pp. 219–233, 2014.

[18] H. Arellano-Garcia and G. Wozny, "Chance constrained optimization of process systems under uncertainty: I. strict monotonicity," *Comput. Chem. Eng.*, vol. 33, no. 10, pp. 1568–1583, 2009.

[19] B. Bhadriraju, A. Narasingam, and J. S.-I. Kwon, "Machine learning-based adaptive model identification of systems: Application to a chemical process," *Chem. Eng. Res. Des.*, vol. 152, pp. 372–383, 2019.

[20] B. Sun, C. Yang, Y. Wang, W. Gui, I. Craig, and L. Olivier, "A comprehensive hybrid first principles/machine learning modeling framework for complex industrial processes," *J. Process Control*, vol. 86, pp. 30–43, 2020.

[21] Y. H. Kim, F. L. Lewis, and C. T. Abdallah, "A dynamic recurrent neural-network-based adaptive observer for a class of nonlinear systems," *Automatica*, vol. 33, no. 8, pp. 1539–1543, 1997.

[22] T. W. Chow and Y. Fang, "A recurrent neural-network-based real-time learning control strategy applying to nonlinear systems with unknown dynamics," *IEEE Trans. Ind. Electron.*, vol. 45, no. 1, pp. 151–161, 1998.

[23] H. A. Talebi, K. Khorasani, and S. Tafazoli, "A recurrent neural-network-based sensor and actuator fault detection and isolation for nonlinear systems with application to the satellite's attitude control subsystem," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 45–60, 2008.

[24] Z. Wu, A. Tran, D. Rincon, and P. D. Christofides, "Machine learning-based predictive control of nonlinear processes. part i: theory," *AIChE J.*, vol. 65, no. 11, p. e16729, 2019.

[25] H. Chung and K.-s. Shin, "Genetic algorithm-optimized long short-term memory network for stock market prediction," *Sustainability*, vol. 10, no. 10, p. 3765, 2018.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural. Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] T. Baumeister, S. L. Brunton, and J. N. Kutz, "Deep learning and model predictive control for self-tuning mode-locked lasers," *JOSA B*, vol. 35, no. 3, pp. 617–626, 2018.

[28] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer, "Learning an approximate model predictive controller with guarantees," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 543–548, 2018.

[29] J. Drgoňa, D. Picard, M. Kvasnica, and L. Helsen, "Approximate model predictive building control via machine learning," *Applied Energy*, vol. 218, pp. 199–216, 2018.

[30] Y. S. Quan and C. C. Chung, "Approximate model predictive control with recurrent neural network for autonomous driving vehicles," in *2019 58th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*. IEEE, 2019, pp. 1076–1081.

[31] S. Ahmed and A. Shapiro, "Solving chance-constrained stochastic programs via sampling and integer programming," in *State-of-the-art decision-making tools in the information-intensive age*. Informs, 2008, pp. 261–269.

[32] A. Prékopa, *Stochastic programming*. Springer Science & Business Media, 2013, vol. 324.

[33] G. Steinbrecher and W. T. Shaw, "Quantile mechanics," *Eur. J. Appl. Math.*, vol. 19, no. 2, pp. 87–112, 2008.

[34] P. M. Esfahani, T. Sutter, and J. Lygeros, "Performance bounds for the scenario approach and an extension to a class of non-convex programs," *IEEE Trans. Automat. Contr.*, vol. 60, no. 1, pp. 46–58, 2014.

[35] Q. Wu, K. Ding, and B. Huang, "Approach for fault prognosis using recurrent neural network," *J. Intell. Manuf.*, pp. 1–13, 2018.

[36] U. Saini, R. Kumar, V. Jain, and M. Krishnajith, "Univariant time series forecasting of agriculture load by using lstm and gru rnns," in *2020 IEEE Students Conference on Engineering & Systems (SCES)*. IEEE, 2020, pp. 1–6.

[37] K. A. Althelaya, E.-S. M. El-Alfy, and S. Mohammed, "Stock market forecast using multivariate analysis with bidirectional and stacked (lstm, gru)," in *2018 21st Saudi Computer Society National Computer Conference (NCC)*. IEEE, 2018, pp. 1–7.

[38] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[39] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.