Recurrent Neural Network-Based Joint Chance Constrained Stochastic Model Predictive Control ⋆

Shu-Bo Yang *, **Zukui Li** *

* *Department of Chemical and Materials Engineering, University of Alberta, Edmonton, Canada (e-mail: zukui@ualberta.ca).*

Abstract: A novel recurrent neural network (RNN)-based approach is proposed in this work to handle joint chance-constrained stochastic model predictive control (SMPC) problem. In the proposed approach, the joint chance constraint (JCC) is first reformulated as a quantile-based inequality to reduce the complexity in approximation. Then, the quantile function (QF) in the quantile-based inequality is replaced by the empirical QF using sample average approximation (SAA). Afterwards, the empirical QF is approximated via an RNN-based surrogate model, which is embedded into the SMPC problem formulation to predict quantile values at different sampling instants. By employing the RNN-based approximation, the resulting deterministic optimization problem is finally solved through a nonlinear optimization solver. The proposed approach is applied to a hydrodesulphurisation process to demonstrate its efficiency in handling the SMPC problem.

Keywords: Recurrent neural network, Stochastic model predictive control, Stochastic optimal control, Joint chance constraint, Sample average approximation

1. INTRODUCTION

Model predictive control (MPC) is a popular advanced control methodology in the industry. At each time step, a finite horizon optimal control problem is solved and only the first control input is applied to the system. This process is repeated in a receding horizon fashion. Practical applications of MPC often face uncertainty issues. To deal with the MPC problem under uncertainty, strategies including robust MPC and stochastic MPC have been widely studied. Among them, the stochastic model predictive control (SMPC) problem has the flexibility in controlling solution robustness and has received lots of attention in the past decade, as shown in Mesbah (2016).

In this study, we focus on the SMPC problem with uncertainty existing in constraints. Chance constraint is a commonly used method to tackle the uncertainty in constraints. There are two types of chance constraint: the individual chance constraint (ICC) and the joint chance constraint (JCC) (Li et al., 2002). Although ICCs are relatively easier to handle, they only guarantee that each equation satisfies the constraint to a certain confidence level. The JCC is more general in engineering applications than the ICC since the JCC ensures all constraints are satisfied simultaneously to a certain confidence level that is more natural in an optimization problem (You et al., 2021). However, the JCC is generally difficult to handle as it requires dealing with multidimensional distributions. Thus, joint chance-constrained programming (JCCP) problems are generally solved through approximations. There are two main approximation methods: analytical approximation methods and sampling-based methods which are elaborated in Yuan et al. (2017).

We focus on sampling-based methods in this work because they can avoid overly conservative solutions occurring in analytical approximation methods (van Ackooij et al., 2014). Among all sampling-based methods, the sample average approximation (SAA) (Pagnoncelli et al., 2009) is a widely used method which approximates the JCC by the empirical JCC based on the collected samples and enforces the empirical joint constraint satisfaction probability (JCSP) to be greater than or equal to the target value. The proposed approach in this study is based on the SAA. More specifically, in the proposed method, the JCC is rewritten as the quantile-based form, and the involved quantile function (QF) is approximated by the empirical QF extended from the SAA. Then, the empirical QF is further modelled by a recurrent neural network (RNN)-based surrogate model to enhance the tractability of problem-solving. Finally, the RNN-based model is incorporated into the constraints to predict quantile values at different sampling instants for handling the JCC at different sampling instants.

There are several existing methods for dealing with the SMPC, e.g., stochastic tube approaches, sample-based SMPC approaches, scenario-based SMPC methods, etc. However, those methods have drawbacks that hinder their applications. Stochastic tube approaches are limited to individual chance constraints and unable to address hard input constraints (Heirung et al., 2018). A sample-based SMPC approach suffers from the high computational cost due to the large sample size required for the optimization at each sampling instant (Batina, 2004). The difficulty

⋆ This work was supported by Imperial Oil and the Natural Sciences and Engineering Research Council of Canada (NSERC).

of a scenario-based SMPC method lies in identifying the appropriate number of scenarios that guarantees both constraint satisfaction and manageable computational burden. Additionally, the computation of a scenario-based SMPC method might be time-consuming, as shown in Navia et al. (2014). The proposed method can overcome the above-mentioned disadvantages in the existing SMPC approaches, by employing the RNN technique which possesses powerful approximation ability and high computational efficiency (Mohajerin and Waslander, 2019).

The proposed approach has the following features: 1) it leads to solutions with desired constraint satisfaction probability; 2) the optimal solution can be obtained more efficiently when compared to the sampling-based approach Moen (2015); 3) there is no restriction on the number of constraints involved in the joint chance constraint; 4) it is applicable to both linear and nonlinear SMPCs via exploiting the RNN-based surrogate model. In the rest of the paper, the SMPC problem is presented first, followed by the proposed approaches and a case study.

2. PROBLEM STATEMENT

In this work, we consider the SMPC problem with uncertainty in constraints. The optimization problem in such SMPC is formulated in discrete-time, which is given as:

$$\min_{u_j \in \mathbb{U}} \quad \sum_{j=0}^{K-1} J(u_j) \tag{1a}$$

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

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

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

where j is the index for sampling instants. K is the number of sampling instants. $x_j \in \mathbb{R}^{n_x}$, $u_j \in \mathbb{R}^{n_u}$, and $\xi \in \mathbb{R}^{n_\xi}$ are the system state, input, and uncertainty vector, respectively. $\mathbb{U}$ is the set of feasible inputs. J is the cost function that only depends on u_j. x_{t_0} is the initial system state at t_0 (current sampling instant). In the above discrete-time optimization problem, the prediction horizon covers the sampling instants $j = 0, ..., K$, and the control horizon covers the sampling instants $j = 0, ..., K-1$. x_j indicates the state at a certain sampling instant j ($j = 1, ..., K$) in the prediction horizon, based on the initial state $x_0 = x_{t_0}$. f_D is a function describing the discrete-time system dynamics. Eq. 1c enforces the constraint satisfaction probability at a certain sampling instant j.

3. METHODS

3.1 Quantile reformulation of chance constraint

The general formulations of a ICC and a JCC are given as:

$$\text{ICC: } Pr(g_i(u, \xi) \leq 0) \geq 1 - \varepsilon, \quad i = 1, \cdots, p \tag{2a}$$

$$\text{JCC: } Pr(g_i(u, \xi) \leq 0, i = 1, \cdots, p) \geq 1 - \varepsilon \tag{2b}$$

where $u \in \mathbb{R}^{n_u}$ represents the decision variable vector. $\xi \in \mathbb{R}^{n_\xi}$ is the random parameter vector. For each i, $g_i : \mathbb{R}^{n_u} \times \mathbb{R}^{n_\xi} \to \mathbb{R}$ is a function. The difference between the ICC and the JCC is the position of $(i = 1, \cdots, p)$. Since the JCC in (2b) ensures that all constraints $g_{i=1,...,p}(u, \xi) \leq 0$ are satisfied simultaneously to a certain confidence level $1-\varepsilon$ which is more general than the ICC, we only consider the JCC in this study.

The above JCC can be rewritten as the following form for further reformulation:

$$Pr(\bar{g}(u, \xi) \leq 0) \geq 1 - \varepsilon, \quad \bar{g}(u, \xi) = \max_{i=1,\cdots,p} g_i(u, \xi) \tag{3}$$

Then, consider the definition of the $1 - \varepsilon$ level quantile with a random variable G, which is given as:

$$Q^{1-\varepsilon}(G) = \inf \{\gamma \in \mathbb{R} \mid Pr(G \leq \gamma) \geq 1 - \varepsilon\} \tag{4}$$

Due to the above definition, the rewritten JCC in (3) can be reformulated as the quantile-based form according to the following relationship:

$$Pr(\bar{g}(u, \xi) \leq 0) \geq 1 - \varepsilon \quad \Leftrightarrow \quad Q^{1-\varepsilon}(\bar{g}(u, \xi)) \leq 0 \tag{5}$$

The benefit of rewriting the JCC as the quantile-based form is shown in Fig. 1. As can be seen from Fig. 1, $Q^{1-\varepsilon}(\bar{g}(u, \xi))$ has better convexity and less complexity than $1 - \varepsilon - Pr(\bar{g}(u, \xi) \leq 0)$ in the original JCC. Accordingly, $Q^{1-\varepsilon}(\bar{g}(u, \xi))$ is easier for a surrogate model to approximate than $1 - \varepsilon - Pr(\bar{g}(u, \xi) \leq 0)$ in the original JCC.

Subsequently, $Q^{1-\varepsilon}(\bar{g}(u, \xi))$ in (5) can be approximated by the empirical QF $\widetilde{Q}^{1-\varepsilon}(\bar{g}(u, \xi))$ which is defined as:

$$\widetilde{Q}^{1-\varepsilon}(\bar{g}(u, \xi)) = \inf \left\{ \gamma \mid \frac{1}{N} \sum_{l=1}^{N} \mathbb{I}(\bar{g}(u, \xi_l) \leq \gamma) \geq 1 - \varepsilon \right\} = \bar{g}_{\lceil M \rceil}(u) \tag{6}$$

where M equals to $(1 - \varepsilon)N$, and N is the number of collected samples of ξ. $\mathbb{I}$ is the indicator function defined as:

$$\mathbb{I}(\bar{g}(u, \xi_l) \leq \gamma) = \begin{cases} 0, & \text{for} \quad \bar{g}(u, \xi_l) > \gamma \\ 1, & \text{for} \quad \bar{g}(u, \xi_l) \leq \gamma \end{cases} \tag{7}$$

$\bar{g}_{\lceil M \rceil}(u)$ represents the M-th smallest component of $\{\bar{g}(u, \xi_1), ..., \bar{g}(u, \xi_N)\}$. The definition of the empirical QF shown in (6) is based on the idea of SAA (Pagnoncelli et al., 2009).

By exploiting the above quantile-based reformulation, the discrete-time optimization problem in (1a)-(1d) can be rewritten as:

$$\min_{u_j \in \mathbb{U}} \quad \sum_{j=0}^{K-1} J(u_j) \tag{8a}$$

$$\text{s.t.} \quad x_{j+1} = f_D(x_j, u_j, \xi), \quad j = 0, .., K-1 \tag{8b}$$

$$\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j)) \leq 0, \quad j = 1, .., K \tag{8c}$$

$$x_0 = x_{t_0} \tag{8d}$$

where $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j))$ denotes the empirical QF at a certain sampling instant j, and $\bar{g}(x_j) = \max_{i=1,...,p} g_i(x_j)$.

3.2 RNN-based optimization model

$\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j))$ can be further approximated to reduce the computational complexity. The RNN-based surrogate model is employed in this study. The RNN is a variant of neural networks (NNs), 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

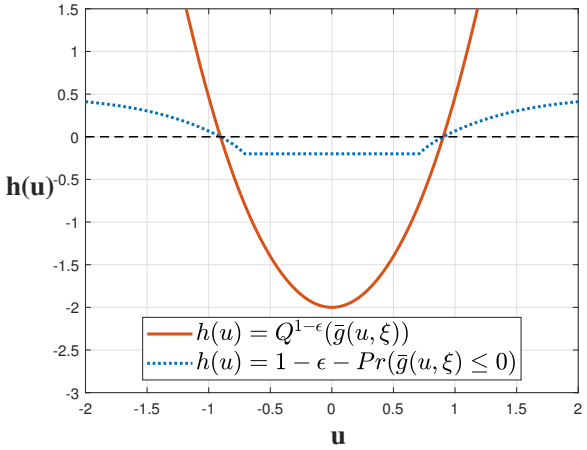


Fig. 1. Comparison between $Q^{1-\varepsilon}(\bar{g}(u,\xi))$ and $1-\varepsilon-Pr(\bar{g}(u,\xi) \leq 0)$. $1-\varepsilon = 0.8$; $\xi = [\xi_1, \xi_2]$; $\xi_1 \sim \mathcal{N}(0,1), \xi_2 \sim U(-2,2)$; $\bar{g}(u,\xi) = \max\{g_1(u,\xi_1), g_2(u,\xi_2)\}$; $g_1(u,\xi_1) = 1.5\xi_1 u^2 - 3$; $g_2(u,\xi_2) = 2\xi_2 u^2 - 2$.

previous time step to the current time step. There are different types of RNN with different structures, e.g., vanilla RNN, Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997), etc. The RNN-based surrogate model in this work is based on the LSTM structure since the LSTM can avoid short-term memory problems in the vanilla RNN and achieve higher accuracy than the GRU (Althelaya et al., 2018). The LSTM generates time-series predictions by updating short-term memory (hidden state) and long-term memory (cell state) at different sampling instants.

The structure of the RNN-based surrogate model used in this study is shown in Fig. 2. The RNN-based model is composed of two NNs and one stacked LSTM (stacked NN-LSTM). The initial system state x_0 is fed into the 2 NNs to produce the initial hidden and cell states for the first LSTM layer which are h_0^1 and C_0^1, respectively. By this means, the stacked NN-LSTM can produce predictions based on different initial conditions. The initial hidden and cell states for the second LSTM layer (h_0^2 and C_0^2, respectively) are set to be zero vectors. The internal structure of an LSTM cell in the first LSTM layer is shown in Fig. 3, which can be interpreted using the following equations:

$$f_{j+1} = \sigma_S(W_f \cdot [h_j^1, u_j] + b_f) \tag{9a}$$
$$in_{j+1} = \sigma_S(W_{in} \cdot [h_j^1, u_j] + b_{in}) \tag{9b}$$
$$\tilde{C}_{j+1} = \sigma_T(W_{\tilde{C}} \cdot [h_j^1, u_j] + b_{\tilde{C}}) \tag{9c}$$
$$o_{j+1} = \sigma_S(W_o \cdot [h_j^1, u_j] + b_o) \tag{9d}$$
$$C_{j+1}^1 = f_{j+1} \circ C_j^1 + in_{j+1} \circ \tilde{C}_{j+1} \tag{9e}$$
$$h_{j+1}^1 = o_{j+1} \circ \sigma_T(C_{j+1}^1) \tag{9f}$$

where σ_S represents the element-wise sigmoid activation function (node "S" in Fig. 3). W_f and b_f are the weight matrix and the bias vector for the forget gate layer, respectively. $[h_j^1, u_j]$ is a vector concatenated from h_j^1 and u_j. σ_T represents the element-wise tanh activation function (node "T" in Fig. 3). W_{in} and $W_{\tilde{C}}$ are weight matrices for the two input gate layers. b_{in} and $b_{\tilde{C}}$ are bias vectors for the two input gate layers. W_o and b_o are the weight matrix and the bias vector for the output gate layer, respectively. The symbol $\circ$ in (9e) and (9f) represents the element-wise multiplication. The LSTM cell in the second LSTM layer has the similar structure. The LSTM cells in the second layer are different from the cells in the first layer based on the following differences: 1) the input becomes h_j^1 instead of u_j; 2) the dimensions of h_j^2 and C_j^2 are different from h_j^1 and C_j^1. The output layer of the stacked NN-LSTM can be described by:

$$\hat{Q}_j^{1-\varepsilon} = \sigma_T(W_{out} \cdot h_j^2 + b_{out}) \tag{10}$$

where $\hat{Q}_j^{1-\varepsilon}$ is the quantile value predicted from the stacked NN-LSTM at the sampling instant j. W_{out} and b_{out} are the weight matrix and the bias vector for the output layer, respectively.

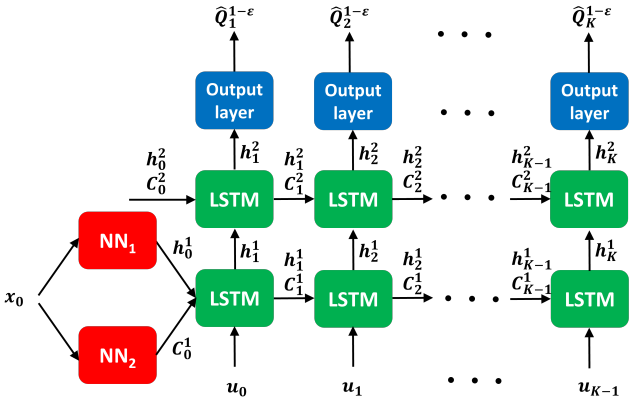


Fig. 2. Schematic diagram of the RNN-based surrogate model employed in this research

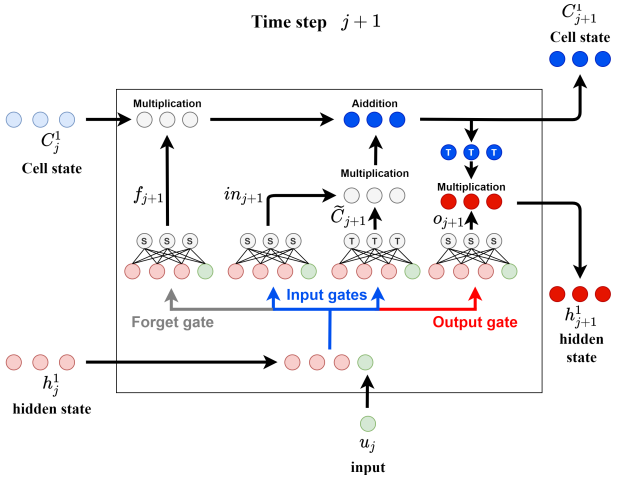


Fig. 3. Schematic diagram for illustrating an LSTM cell

After incorporating the stacked NN-LSTM into the optimization problem (8a)-(8d) to predict the quantile values for (8c) at different sampling instants, the problem can be rewritten as:

$$\min_{u_j \in \mathbb{U}} \quad \sum_{j=0}^{K-1} J(u_j) \tag{11a}$$

$$\text{s.t.} \quad \hat{Q}^{1-\varepsilon}_{j=1,\ldots,K} = NL(u_{j=0,\ldots,K-1}, x_0) \tag{11b}$$

$$\hat{Q}^{1-\varepsilon}_{j} \leq 0, \quad j = 1, .., K \tag{11c}$$

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

where NL denotes the stacked NN-LSTM model. This is a deterministic nonlinear optimization problem that can be solved using a nonlinear solver such as IPOPT (Wächter and Biegler, 2006).

While applying the proposed approach to a SMPC problem, we first solve the problem in (11a)-(11d) to obtain the optimal control sequence based on the given initial system state. Then, only the first component in the optimal control sequence is applied to the controlled system. Afterwards, the updated system state is used as the new initial state for the problem in (11a)-(11d). The above procedure is repeated to control the system.

4. CASE STUDY

The proposed SMPC method is applied to a case study and compared with the Monte Carlo sampling-based method from Moen (2015), to investigate the performance of the proposed approach.

The case study in this work is the SMPC problem of the hydrodesulphurisation process (HDS) shown in Fig. 4. According to Fig. 4, 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 the 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.

The HDS is modelled mathematically based on the following assumptions for simplification:

- Temperatures in R-1 and R-2 are controlled perfectly.
- Pressures in all the streams and units in the HDS are controlled perfectly.
- The hydrodesulphurisation reaction can be described by the first-order model.
- The flash tank T-1 can separate hydrocarbons from hydrogen and sulfide perfectly.
- R-1 and R-2 can be modelled as 1 reactor.

According to the above assumptions, only mass and component balances should be taken into account for modelling the HDS process.

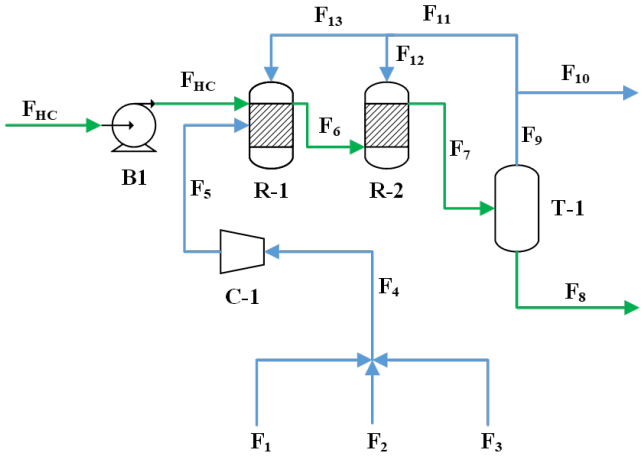


Fig. 4. Schematic diagram of the studied HDS

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

$$\min_{F_1,F_2,F_{10}} \quad \int_{t_0}^{t_f} (C_{H4}X_1F_1 + C_{H3}X_2F_2)\, dt \tag{12a}$$

$$\text{s.t.} \quad \tau \frac{dF_x^{H2}}{dt} + F_x^{H2} = F_{HC}\rho \tag{12b}$$

$$\frac{VP}{ZR_gT}\frac{dX_{H2}}{dt} = F_5X_5 - F_{10}X_{H2} - F_x^{H2} \tag{12c}$$

$$F_5 = F_{10} + F_x^{H2} \tag{12d}$$

$$F_5 = F_1 + F_2 + F_3 \tag{12e}$$

$$F_5X_5 = F_1X_1 + F_2X_2 + F_3X_3 \tag{12f}$$

$$Pr(X_{H2} \geq 0.7, X_5 \geq 0.9) \geq 1 - \varepsilon \tag{12g}$$

$$0 \leq F_1 \leq 1400 \tag{12h}$$

$$0 \leq F_2 \leq 790 \tag{12i}$$

$$0 \leq F_{10} \leq 1500 \tag{12j}$$

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 problem. Note that $t_f - t_0 = 2$. C_{H4} and C_{H3} are unit prices (unit: €/Mmol) of hydrogen from streams F_1 and F_2, respectively. F_1, F_2, and F_{10} are the molar flow rates of the streams F_1, F_2, and F_{10} (unit: kmol/h), respectively. Since the hydrodesulphurisation reaction model is assumed to be first-order, the hydrogen consumption rate F_x^{H2} (unit: kmol/h) in the reactor can be described by Eq. 12b. τ is the time constant of the reaction (unit: h). F_{HC} is the volume flow rate of hydrocarbons fed into the reactor (unit: m^3/h). ρ is a random parameter following the Gaussian distribution ($\rho \sim \mathcal{N}(12.6, 0.4)$), which is the specific consumption rate characteristic of the type of hydrocarbon received (unit: kmol/m^3). Eq. 12c is used to calculate the hydrogen mole fraction in the reactor (X_{H2}). V is the reactor volume (unit: m^3). P denotes the pressure (unit: atm) inside the reactor. Z is the compressibility factor. R_g is the ideal gas constant. T is the temperature (unit: K) inside the reactor. F_5 is the molar flow rate of the stream F_5 (unit: kmol/h). X_5 is the hydrogen mole fraction in stream F_5. The mass balance over the reactor is expressed as Eq. 12d. Eq. 12e is exploited to calculate F_5. The component balance of hydrogen can be described by

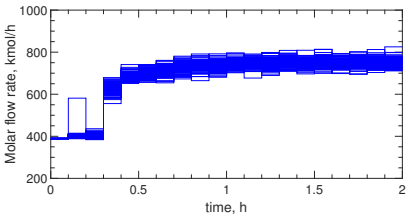
Eq. 12f. X_3 is a random parameter following the Gaussian distribution ($X_3 \sim \mathcal{N}(0.85, 0.013)$), which is the hydrogen fraction corresponding to F_3. $1-\varepsilon$ is the user-defined confidence level, which is set to be 0.8 in this cases study. Eqs. 12h-12j are the bounds for the decision variables of the optimization problem. The values of parameters are found in Moen (2015).

Since the proposed SMPC approach can only be used for discrete-time problems, the objective function and the differential equations in the above optimization problem are discretized through the trapezoidal rule method with 20 time intervals. Then, the training data for the stacked NN-LSTM is generated by the following steps:

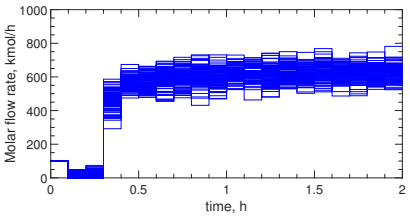
(1) Collect 1000 samples of $[\rho, X_3]$ from the corresponding Gaussian distributions.
(2) Randomly select a sequence of manipulated variables (MVs) (F_1, F_2, and F_{10}) from the uniform distributions between the lower and upper bounds of these MVs.
(3) Calculate different values of X_{H2} and X_5 based on the sequence of MVs and different realizations of $[\rho, X_3]$, through the discretized differential equations mentioned above.
(4) Calculate the 0.8-quantile value of $\bar{g}$ based on X_{H2} and X_5 obtained from the previous step ($\bar{g} = \max\{-X_{H2}+0.7, -X_5+0.9\}$).
(5) Repeat steps 2 $\sim$ 4 to produce different 0.8-quantile values corresponding to different sequences of MVs.
(6) Different 0.8-quantile values paired with different sequences of MVs are respectively used as targets and inputs to train the stacked NN-LSTM.

After training the stacked NN-LSTM (it takes 5300 seconds to train), the stacked NN-LSTM is incorporated into the mentioned discretized optimization problem. Afterwards, the optimization model involving the stacked NN-LSTM can be solved deterministically using IPOPT solver. Regarding the stacked NN-LSTM in this case study, the dimensions of the hidden and cell states in the first LSTM layer are all equal to 50. The dimensions of the hidden and cell states in the second LSTM layer are all equal to 30. The 2 NNs in the stacked NN-LSTM individually have 1 ReLU hidden layer containing 50 neurons.

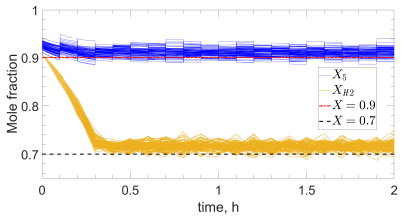
While implementing the proposed method to address this case study, the optimization model involving the stacked NN-LSTM is first solved based on the given initial conditions ($F_x^{H2} = 682.5$ and $X_{H2} = 0.9$). Then, only the first component in the obtained optimal control sequence is applied to the above-mentioned discretized differential equation model based on a realization of $[\rho, X_3]$ to attain new values of F_x^{H2} and X_{H2}. Subsequently, the optimization problem based on the new values of F_x^{H2} and X_{H2} is solved to obtain a new optimal control sequence. Afterwards, the first component in the obtained optimal control sequence is applied to the discretized differential equation model based on a new realization of $[\rho, X_3]$ to attain new values of F_x^{H2} and X_{H2}. After repeating the above procedure for 20 sampling instants, 1 SMPC execution is completed. 100 SMPC simulation experiments are completed in this case study and the obtained results are shown in Fig. 5. Since the optimal sequences of F_2 from the 100 SMPC executions are all equal to 0 for all sampling instants, the



(a) Manipulated variable, F_1



(b) Manipulated variable, F_{10}



(c) Controlled variables, X_5 and X_{H2}

Fig. 5. Results of 100 SMPC executions

results of F_2 are not shown in Fig. 5. Because the hydrogen mole fraction of stream F_2 ($X_2 = 0.931$) is lower than of stream F_1 ($X_1 = 0.991$), using hydrogen from stream F_2 would make the hydrogen mole fraction in the reactor (X_{H2}) more difficult to be above the required value of 0.7. Thus, the optimal results suggest not to use stream F_2. As shown in Fig. 5, the constraint satisfaction probabilities at all sampling instants are above the required confidence level of 0.8 (minimum and maximum values are 0.8314 and 0.9548, respectively), which means that the proposed method can reliably control the HDS in this case study with satisfactory constraint satisfaction for all sampling instants. Notably, the probability of constraint satisfaction at each sampling instant is calculated through the following steps: 1) calculate 100 $\bar{g}$ based on the 100 X_{H2} and X_5 obtained from the 100 SMPC executions; 2) calculate the percentage of the 100 $\bar{g}$ less than or equal to 0, which is the probability of constraint satisfaction at a sampling instant.

The average optimal objective and solution time based on the 100 SMPC executions by using the proposed approach are compared with the optimal objective and solution time attained from the Monte Carlo sampling-based method in Moen (2015). The comparison is shown in Table 1. The solution time corresponding to a method in Table 1 is the overall time of an SMPC implementation for 20 sampling instants. The Monte Carlo sampling-based method in Moen's work is based on 1000 realizations of $[\rho, X_3]$, which is the same as the proposed approach. As can be seen from Table 1, the proposed approach can obtain a better objective value in a much shorter solution time than the sample-based approach in literature.

Table 1: Comparison between the proposed approach and the Monte Carlo sampling-based method in Moen (2015)

Method	Optimal objective value	Solution time (s)
Proposed method	118.8	395
Monte Carlo sampling-based method	132.9	2100

Finally, based on the above discussion, the proposed approach can efficiently achieve the feasible solution for a joint chance-constrained SMPC problem, and the solution from the proposed method is better than the solution from the existing method. Also, the presented method is effective for the SMPC problem with any number of constraints in the JCC since $\bar{g}(u,\xi) = \max_{i=1,\cdots,p} g_i(u,\xi)$ is adopted. Moreover, the presented approach is applicable for both linear and nonlinear SMPC problems because the employed stacked NN-LSTM is capable of handling both linear and nonlinear models. Although generating training data and training the stacked 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 stacked NN-LSTM is trained. The presented method requires the prior knowledge of uncertainty distributions. If the uncertainty distributions in the SMPC problem studied are accessible, the presented approach can be an efficient method for SMPC problem.

5. CONCLUSION

A novel approach based on the RNN surrogate model is proposed to address stochastic model predictive control problems with joint chance constraints. In the proposed method, the quantile-based reformulation is applied to the joint chance constraint and the quantile function is further approximated by a stacked LSTM-based surrogate model (stacked NN-LSTM) which is a hybrid model consisting of feed-forward neural networks and LSTM taking initial conditions and control sequence as input. To handle SMPC, the stacked NN-LSTM can be embedded into the considered optimization problem. When SMPC is executed with this method, the involved optimization problem 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. While comparing with the sample-based method in the literature, the proposed approach can achieve faster and better solutions for an SMPC problem. 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 joint chance constraints involving any number of constraints.

ACKNOWLEDGEMENTS

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

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

Batina, I. (2004). *Model predictive control for stochastic systems by randomized algorithms*. Citeseer.

Heirung, T.A.N., Paulson, J.A., O'Leary, J., and Mesbah, A. (2018). Stochastic model predictive control—how does it work? *Computers & Chemical Engineering*, 114, 158–170.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735–1780.

Li, P., Wendt, M., and Wozny, G. (2002). A probabilistically constrained model predictive controller. *Automatica*, 38(7), 1171–1176.

Mesbah, A. (2016). Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36(6), 30–44.

Moen, M.A. (2015). Stochastic optimisation.

Mohajerin, N. and Waslander, S.L. (2019). Multistep prediction of dynamic systems with recurrent neural networks. *IEEE transactions on neural networks and learning systems*, 30(11), 3370–3383.

Navia, D., Sarabia, D., Gutiérrez, G., Cubillos, F., and de Prada, C. (2014). A comparison between two methods of stochastic optimization for a dynamic hydrogen consuming plant. *Computers & Chemical Engineering*, 63, 219–233.

Pagnoncelli, B.K., Ahmed, S., and Shapiro, A. (2009). Sample average approximation method for chance constrained programming: theory and applications. *Journal of optimization theory and applications*, 142(2), 399–416.

van Ackooij, W., Henrion, R., Möller, A., and Zorgati, R. (2014). Joint chance constrained programming for hydro reservoir management. *Optimization and Engineering*, 15(2), 509–531.

Wächter, A. and Biegler, L.T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1), 25–57.

You, B., Esche, E., Weigert, J., and Repke, J.U. (2021). Joint chance constraint approach based on data-driven models for optimization under uncertainty applied to the williams-otto process. In *Computer Aided Chemical Engineering*, volume 50, 523–528. Elsevier.

Yuan, Y., Li, Z., and Huang, B. (2017). Robust optimization approximation for joint chance constrained optimization problem. *Journal of Global Optimization*, 67(4), 805–827.