Data-Driven Optimization under Uncertainty

by

Shu-Bo Yang

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Process Control

Department of Chemical and Materials Engineering

University of Alberta

Abstract

With the advances made in machine learning and data science, data-driven modeling and optimization techniques have garnered significant attention in recent years. However, despite the availability of various data-driven methods for addressing optimization problems under uncertainty, their practical applicability is often constrained by their own limitations such as high computational costs and reliance on non-trivial assumptions. This thesis aims to investigate and develop novel data-driven approaches to address various optimization problems involving uncertainty and overcome the limitations of existing methods.

First, we propose a novel framework to address process optimization under surrogate model prediction uncertainty. The framework involves approximating an ensemble surrogate model with a computationally efficient mixture density network (MDN) and embedding the MDN into a chance-constrained optimization problem with a mean-variance-type objective. This method reduces the high computational cost seen in other existing methods for optimization considering surrogate model prediction uncertainty. This approach is demonstrated through a numerical example and two case studies, showcasing its capability to solve various optimization problems under surrogate model prediction uncertainty.

Second, we develop a new neural network (NN)-based approach for solving the uncertain optimization problems in joint chance-constrained formulations (joint chance-constrained optimization problems, JCCPs). The approach involves approximating a joint chance constraint (JCC) with a NN and incorporating the NN into the optimization model. This method makes NP-hard JCCPs tractable and deterministically solvable. This method is applied to a process optimization problem to show its performance in solving a nonlinear JCCP. Furthermore, we extend the above method to handle joint chance-constrained stochastic optimal control problems (JCC-SOCPs). We replace the NN with the recurrent neural network (RNN) for the approximation of the JCC in a JCC-SOCP. This method has a much lower computational burden than other commonly used stochastic optimal control approaches. This approach is applied to a numerical SOCP example and a case study to demonstrate its efficacy.

Third, we propose a novel distributionally robust chance-constrained optimization (DRCCP) method to handle JCCPs without knowing true uncertainty distributions. This DRCCP method is based on the kernel ambiguity set established by utilizing the maximum mean discrepancy (MMD). This approach overcomes the restrictions of the popular Wasserstein DRCCP which necessitates complicated assumptions on uncertain constraints. A numerical example and a nonlinear process optimization problem are studied to demonstrate the efficacy of the presented DRCCP method. Subsequently, this DRCCP approach is further combined with a neural network-like deep kernel to enhance its performance. The effectiveness of this deep kernel-based DRCCP is demonstrated by applying it to a case study.

Fourth, we develop another novel DRCCP method to further overcome more limitations of the popular Wasserstein DRCCP. This method is based on the Sinkhorn ambiguity set constructed by using the Sinkhorn distance. This approach outperforms the Wasserstein DRCCP by being assumption-free on uncertain constraints and being able to hedge against more general families of uncertainty distributions. The performance of this method is evaluated through a numerical example and a nonlinear process optimization.

Fifth, we develop an innovative iterative algorithm that can remove outliers and extreme data samples leading to overly conservative DRCCP solutions. The presented algorithm can significantly improve the DRCCP solution quality, and it can simultaneously ensure the feasibility of the DRCCP solution. Moreover, the proposed algorithm is compatible with various DRCCP models such as Wasserstein DRCCP, kernel DRCCP, Sinkhorn DRCCP, etc.

Overall, we make several contributions through this research. From a methodological perspective, we develop several novel data-driven approaches for optimization under uncertainty and significantly overcome the limitations of existing popular methods. From an application perspective, our proposed methods can be applied to various real-world optimization problems such as process optimization and optimal control.

Preface

This thesis was completed under the supervision of Dr. Zukui Li. This is a paper-based thesis. Each chapter provides the necessary background information for its own comprehension. Computer programming formed a significant part of the research, with MATLAB, GAMS, and Python being the primary languages utilized. Rigorous validation of the scripts was carried out in close collaboration between Dr. Zukui Li and myself. The submission details and contributions of each researcher are provided below.

Chapter 2 of this thesis has been published as Shu-Bo Yang, Zukui Li, and Wei Wu. "Data-driven process optimization considering surrogate model prediction uncertainty: A mixture density network-based approach." *Industrial & Engineering Chemistry Research* 60.5 (2021): 2206-2222.

The first half of Chapter 3 of this thesis has been published as Shu-Bo Yang, Jesús Moreira, and Zukui Li. "Joint Chance Constrained Process Optimization through Neural Network Approximation." *Computer Aided Chemical Engineering*. Vol. 49. Elsevier, 2022. 1237-1242.

The second half of Chapter 3 of this thesis has been published as Shu-Bo Yang, Zukui Li, and Jesus Moreira. "A recurrent neural network-based approach for joint chance constrained stochastic optimal control." *Journal of Process Control* 116 (2022): 209-220.

The first half of Chapter 4 of this thesis has been published as Shu-Bo Yang and Zukui Li. "Kernel distributionally robust chance-constrained process optimization." *Computers & Chemical Engineering* 165 (2022): 107953.

The second half of Chapter 4 of this thesis has been published as Shu-Bo Yang and Zukui Li. "Distributionally Robust Chance-Constrained Optimization with Deep Kernel Ambiguity Set." *2022 IEEE International Symposium on Advanced Control of Industrial Processes (AdCONIP)*. IEEE, 2022.

Chapter 5 of this thesis has been submitted as Shu-Bo Yang and Zukui Li. "Distributionally Robust Chance-Constrained Optimization with Sinkhorn Ambiguity Set." Submitted to *AIChE Journal* (minor revision in progress).

Chapter 6 of this thesis is a manuscript under preparation with the title "A Novel Efficient Algorithm for Improving Solution Quality of Distributionally Robust Chance-Constrained Optimization".

In Chapter 2, I am responsible for theory development, algorithmic implementation, computational experiments, result analysis, and writing the manuscript. Dr. Zukui Li is responsible for concept formulation, research supervision, and reviewing as well as editing the manuscript. Dr. Wei Wu is responsible for reviewing and editing the manuscript.

In Chapter 3, I am responsible for theory development, algorithmic implementation, computational experiments, result analysis, and writing the manuscript. Dr. Zukui Li is responsible for concept formulation, research supervision, and reviewing as well as editing the manuscript. Dr. Jesus Moreira is responsible for providing industrial experience and reviewing as well as editing the manuscript.

In Chapters 4, 5, and 6, I am responsible for theory development, algorithmic implementation, computational experiments, result analysis, and writing the manuscript. Dr. Zukui Li is responsible for concept formulation, research supervision, and reviewing as well as editing the manuscript.

Acknowledgements

My utmost gratitude goes to Dr. Zukui Li, my supervisor, for granting me this remarkable chance and providing me with unwavering support and guidance throughout my learning journey in the field of optimization.

I extend my heartfelt appreciation to my beloved family for their affection, encouragement, and motivation that have been instrumental in my achievements.

I would also like to express my sincere thanks to all the staff members of the University of Alberta for fostering a welcoming and conducive atmosphere.

The researchers involved in this study express their deep appreciation to NSERC for the financial support extended during the research endeavor.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Due to the significant development in machine learning and data science, there has been a notable amount of interest in data-driven modeling and optimization in recent years. Although data-driven techniques can enhance the problem-solving efficiency of optimization problems, most of the techniques are not capable of handling uncertainty in an optimization framework. However, uncertainty generally exists in real-world problems. Although some data-driven approaches have been proposed to address optimization under uncertainty [1–4], their practical applicability is frequently restricted by limitations such as high computational costs and reliance on complicated assumptions. As a result, practical and efficient methods for solving optimizations involving uncertainty are still in demand.

Since data-driven models have much lower computational costs than full-order models, they can act as surrogate models to replace the full-order models in optimization problems to reduce computational expense [5]. However, data-driven models inherently contain prediction uncertainty because of the training data set variations [6]. Although there are several existing approaches [6–9] for handling model prediction uncertainty, they are difficult to be incorporated into an optimization formulation to address uncertainty in the optimization framework. This is because of their sophisticated model structures and high computational burdens. This research addresses the surrogate model prediction uncertainty in an optimization problem by proposing a novel machine learning-based approach with high computational efficiency and reliability.

Chance-constrained programming (CCP) [10] is a widely-used approach to address optimization problems involving uncertainty in constraint functions. There are two types of CCP: the individual CCP (ICCP) and the joint CCP (JCCP) [11]. In ICCP, a constraint satisfaction probability is enforced for each uncertain constraint. The JCCP is more general than the ICCP in the sense that it ensures multiple constraints are satisfied jointly to a certain probability [12]. However, a JCCP problem is difficult to solve as it requires dealing with the multidimensional distribution [13]. Although there are several existing approaches to deal with JCCP problems [1, 2, 11, 14], these methods have limited practical applications due to their high computational costs or the utilization of the overly conservative approximation. Two innovative machine learning-based approaches with high efficiency and performance are presented in this work to handle JCCP problems.

One major challenge in solving a JCCP problem is that the true distribution of uncertainty should be known in advance. However, obtaining the true uncertainty distribution is difficult in real-world scenarios. As a solution, distributionally robust chance-constrained programming (DRCCP) [15] is proposed. It aims to solve a JCCP problem by optimizing the objective with the worst-case joint constraint satisfaction probability (JCSP) above an acceptable level, without anticipating the true uncertainty distribution. The existing popular DRCCP approaches have several drawbacks such as unreliable out-of-sample performance, dependence on non-trivial assumptions, failure to hedge against the correct families of uncertainty distributions, etc [16, 17]. This study overcomes the above challenges by presenting two new DRCCP approaches with better performance, reliability, and applicability. Furthermore, a novel iterative algorithm is proposed in this work, which can significantly improve the DRCCP solution quality by removing outliers and extreme data causing overly conservative solutions.

1.2 Preliminaries

1.2.1 Artificial neural network

The artificial neural network (ANN) has a fully connected feed-forward network structure shown in the following schematic diagram:

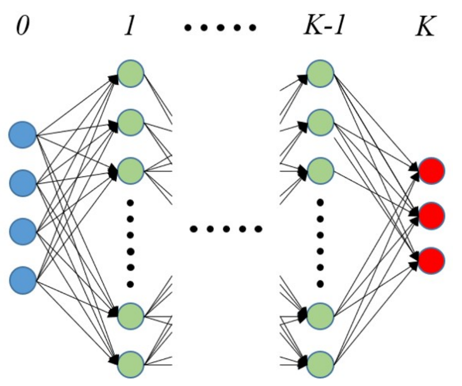


Figure 1.1: Schematic diagram of the fully connected feed-forward artificial neural network

Regarding the above schematic diagram, the fully connected feed-forward ANN maps input variables to output predictions through layers of neurons (the nodes in the schematic diagram). Layer 0 and layer K are the input and output layers, respectively. The rest of the layers are namely hidden layers. Besides the input layer, all the neurons in each layer are fully connected with all the neurons in the previous layer. The output of one neuron in each layer (except the input layer) can be calculated using (1.1):

$$x_j^k = \sigma\left(\sum_{i=1}^{n} W_{ij}^k x_i^{k-1} + b_j^k\right), \qquad k = 1, ..., K \tag{1.1}$$

where i, j, and k symbolize the index of each neuron in the previous layer, the index of each neuron in the current layer, and the index of each layer, respectively. x, W, b, and n represent the output of the neuron, the weights between the current and previous layers, the bias for each neuron in the current layer, and the number of neurons in the previous layer, respectively. σ represents the transfer function (activation function) of each neuron in the current layer.

1.2.2 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 Figure 1.2 which is an unfolded representation of a vanilla RNN for modeling a dynamic system with 3 time steps.

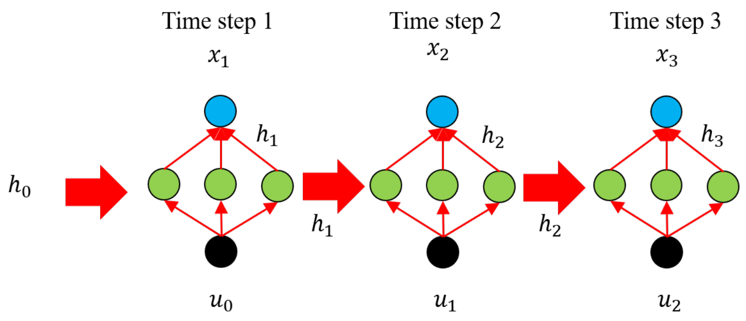


Figure 1.2: Schematic diagram of a vanilla recurrent neural network

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 [18]. At the time step $j+1$ ($j = 0, 1, 2$, j is also the time step index), 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:

$$h_{j+1} = \sigma_h(W_h u_j + U_h h_j + b_h) \tag{1.2a}$$

$$x_{j+1} = \sigma_x(W_x h_{j+1} + b_x) \tag{1.2b}$$

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 [19]. 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 preferable than the GRU when prediction accuracy is critical since the LSTM has more gates and parameters [20]. Thus, the LSTM is used in the study in Section 3.2. A schematic diagram for illustrating an LSTM cell is shown in Figure 1.3. 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 of the LSTM is in Appendix A2.1.

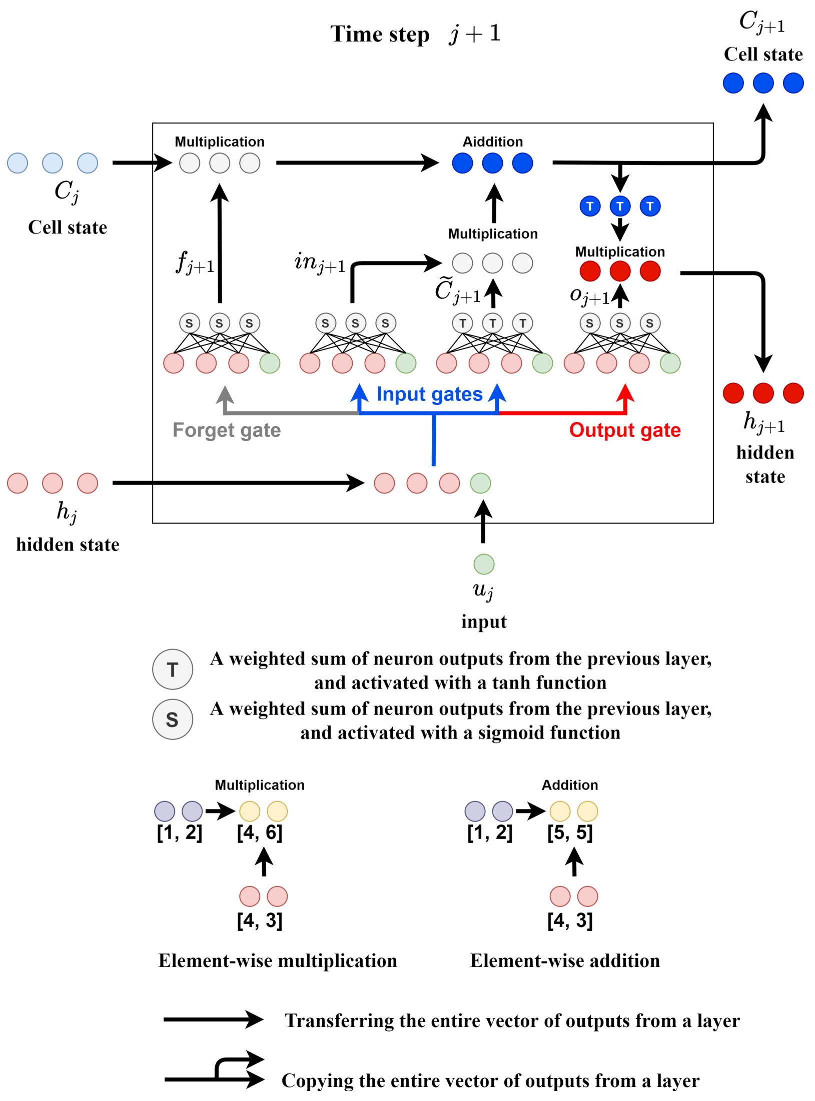

Time step $j+1$
C_{j+1}
Cell state
Multiplication
Aiddition
C_j
Cell state
T
Multiplication
Multiplication
f_{j+1}
in_{j+1}
$\widetilde{C}_{j+1}$
o_{j+1}
S
Input gates
Forget gate
Output gate
h_{j+1}
hidden state
h_j
hidden state
u_j
input
T
A weighted sum of neuron outputs from the previous layer, and activated with a tanh function
S
A weighted sum of neuron outputs from the previous layer, and activated with a sigmoid function
Multiplication
[1, 2]
[4, 6]
[4, 3]
Element-wise multiplication
Addition
[1, 2]
[5, 5]
[4, 3]
Element-wise addition
Transferring the entire vector of outputs from a layer
Copying the entire vector of outputs from a layer

Figure 1.3: Schematic diagram for illustrating an LSTM cell

1.2.3 Chance-constrained programming

When uncertainty occurs in the constraint functions of an optimization problem, it would be difficult to find the reliable optimal decision due to the unexpected random effects that would cause the constraint violation. Under this circumstance, one would rather require the optimal decision to be feasible with a certain probability/chance instead of satisfying constraints with exact limited values. Based on this idea, chance-constrained programming (CCP) [10] is proposed.

There are two divisions of CCP: the individual CCP (ICCP) and the joint CCP (JCCP) [21]. They are illustrated as follows:

- **Individual chance-constrained programming (ICCP):** As to the individual chance-constrained programming, the term "individual" relates to the fact that each of the stochastic constraints is reformulated into a chance constraint individually. The general formulation of a set of individual chance constraints (ICCs) is given as:

  $$Pr(g_i(x,\xi) \leq 0) \geq 1-\delta \quad (i=1,...,w) \tag{1.3}$$

  x and ξ are decision and random vectors, respectively. $g_i(x,\xi) \leq 0$ refers to a finite set of inequalities (constraints) and Pr is a probability measure. The value $1-\delta \in [0,1]$ is the confidence value or the probability level. ICCs are appealing in that they are simple to solve [12] and their linear deterministic inequalities as well as analytical solutions can be obtained easily [22]. The ICCP has been applied for the refinery blend planning [11], process industry scheduling [23], balancing return and risk [24], etc. As to the drawbacks of the ICCP, they only guarantee that each equation satisfies the constraint to a certain confidence level. The model with individual chance constraints is not feasible while constraints as a whole are required to be satisfied simultaneously. In that circumstance, the JCCP is adopted.

- **Joint chance-constrained programming (JCCP):** The JCCP ensures that constraints as a whole are satisfied simultaneously to a certain confidence level. The general formulation of one single joint chance constraint (JCC) is given as:

  $$Pr(g_i(x,\xi) \leq 0 \quad (i=1,...,w)) \geq 1-\delta \tag{1.4}$$

The difference between (1.3) and (1.4) is given by the position of $(i = 1, ..., w)$. The JCCP has been exploited for stochastic optimal control [25], design and planning of chemical supply chains [26], hydro reservoir management [27], etc. Although the JCCP is more general and natural than the ICCP, it is incredibly difficult to solve as it requires dealing with multidimensional distributions. [13] Thus, JCCP problems are generally solved through approximations [28]. More details about the approximation methods for JCCP problems are explained in Section 1.3.2.

1.2.4 Distributionally robust chance-constrained programming

The requirement of knowing the true uncertainty distribution is a major challenge in solving JCCP problems in practice (this is also a major difficulty in solving an ICCP problem in practice). To address this issue, the distributionally robust chance-constrained programming (DRCCP) [29] method has been introduced, which solves a JCCP problem based on the worst-case distribution among all possible data-generating candidate distributions, without estimating the true uncertainty distribution. The DRCCP method optimizes the objective with the worst-case joint constraint satisfaction probability (JCSP) above an acceptable confidence level, over a set of candidate distributions. This set is called the ambiguity set. The ambiguity set may contain certain distributional information about the unknown true distribution, such as moments, structural properties, and domain knowledge obtained from gathered data. In order to ensure the solution quality and robustness, a well-designed ambiguity set should contain the true distribution with high confidence and exclude irrelevant distributions resulting in overly conservative decisions. Additionally, the ambiguity set must be designed in a way that enables the DRCCP problem to be a tractable mathematical model that can be solved using off-the-shelf solvers [30]. More details about the ambiguity set are elaborated in Section 1.3.3.

The general formulation of a DRCCP problem is given as:

$$\min_{x\in\mathcal{X}} \quad f(x) \tag{1.5a}$$

$$\text{s.t.} \quad \min_{\mathbb{P}\in\mathcal{P}(\Xi)} Pr\left(g_i(x,\xi) \leq 0, \forall i = 1, ..., w\right) \geq 1-\delta \tag{1.5b}$$

where x is the decision variable vector with the feasible set $\mathcal{X}$. ξ is a random parameter vector on measurable continuous support Ξ. $\mathcal{P}(\Xi)$ is the ambiguity set on the support Ξ, which contains infinite number of candidate distributions $\mathbb{P}$ ($\mathbb{P}$ are distributions of ξ). Pr is the probability measure. Inequality (1.5b) is the worst-case joint chance constraint (JCC) that ensures the worst-case JCSP greater than or equal to a user-defined confidence level $1 - \delta$. The expression (1.5b) can be rewritten equivalently as the worst-case violation probability form:

$$\max_{\mathbb{P}\in\mathcal{P}(\Xi)} Pr\left(\bigcup_{i=1}^{w} g_i(x,\xi) > 0\right) \leq \delta \tag{1.6}$$

The left-hand side of the above expression represents the worst-case violation probability, which is the maximum probability of constraint violation ($g_1(x,\xi) > 0$ or $g_2(x,\xi) > 0$, or ..., or $g_w(x,\xi) > 0$) under all possible probability distributions in the ambiguity set. The expression (1.6) is more useful for deriving tractable DRCCP formulations. More details about the derivations of tractable DRCCP formulations are explained in Chapter 4.

1.2.5 Wasserstein distance

The Wasserstein distance is a metric that quantifies the dissimilarity between two probability distributions by computing the minimum cost of transporting one distribution to another through an optimal transportation plan. This distance measures the amount of cost necessary to move one distribution to another, taking into account the overall transportation cost based on the optimal transportation plan. The type-q Wasserstein distance $\mathcal{W}_q$ between the distributions $\mathbb{P}$ and $\mathbb{P}_0$ is defined by:

$$\mathcal{W}_q(\mathbb{P},\mathbb{P}_0) := \min_{\pi\in\Pi(\mathbb{P},\mathbb{P}_0)} \left(\int_{\Xi\times\Xi} c(\eta,\xi)^q d\pi(\eta,\xi)\right)^{1/q} \tag{1.7}$$

where random parameter vectors $\eta \sim \mathbb{P}$ and $\xi \sim \mathbb{P}_0$. $\mathbb{P}$ and $\mathbb{P}_0$ are two probability measures based on continuous support Ξ. $c(\eta,\xi)$ denotes the ground cost function and is often evaluated based on a norm $\|\eta - \xi\|$ (e.g., ℓ_1 norm, ℓ_2 norm, etc.). π is called the transportation plan between $\mathbb{P}$ and $\mathbb{P}_0$. $\Pi(\mathbb{P},\mathbb{P}_0)$ is the set of all joint distributions on $\Xi\times\Xi$, whose marginal distributions are $\mathbb{P}$ and $\mathbb{P}_0$. Since the type-1 Wasserstein distance $\mathcal{W}_1$ is the most popular Wasserstein metric [31], all the Wasserstein distances in the rest of this thesis refer to the type-1 Wasserstein distance $\mathcal{W}_1$. $\mathcal{W}_1$ can be equivalently reformulated in the expectation form:

$$\mathcal{W}_1(\mathbb{P}, \mathbb{P}_0) = \inf_{\pi \in \Pi(\mathbb{P}, \mathbb{P}_0)} \mathbb{E}_{(\eta,\xi)\sim\pi} \left[c(\eta, \xi)\right] \tag{1.8}$$

Moreover, the type-1 Wasserstein distance is a special case of the integral probability metrics (IPMs) [32]. An IPM is generally defined as:

$$\Gamma[\mathcal{F}, \mathbb{P}, \mathbb{P}_0] = \max_{f \in \mathcal{F}} \left\{ \int_{\Xi} f(\xi) d\mathbb{P}(\xi) - \int_{\Xi} f(\xi_0) d\mathbb{P}_0(\xi_0) \right\} \tag{1.9}$$

$\mathcal{F}$ is a space of real-valued bounded measurable functions on Ξ. The IPM $\Gamma[\mathcal{F}, \mathbb{P}, \mathbb{P}_0]$ is fully characterized by $\mathcal{F}$. While $\mathcal{F} = \{f : \text{lip}(f) \leq 1\}$, the IPM reduces to the type-1 Wasserstein distance. Similarly, the IPM reduces to the max mean discrepancy (MMD) while $\mathcal{F} = \{f : \|f\|_{\mathcal{H}} \leq 1\}$. The details about the MMD are explained in Subsection 4.1.3.

When two discrete empirical distributions $\hat{\mathbb{P}}$ and $\hat{\mathbb{P}}_0$ are respectively supported on finite samples $\{\eta_n\}_{n=1}^N$ and $\{\xi_m\}_{m=1}^M$, $\mathcal{W}_1$ between $\hat{\mathbb{P}}$ and $\hat{\mathbb{P}}_0$ based on the finite discrete support can be formulated as the following optimal transport problem [31]:

$$\begin{aligned} \mathcal{W}_1 = \min_{\pi \geq 0} \quad & \sum_{n=1}^{N} \sum_{m=1}^{M} \|\eta_n - \xi_m\| \pi_{nm} \\ \text{s.t.} & \sum_{m=1}^{M} \pi_{nm} = \frac{1}{N}, \forall n = 1, ..., N \\ & \sum_{n=1}^{N} \pi_{nm} = \frac{1}{M}, \forall m = 1, ..., M \end{aligned} \tag{1.10}$$

The above optimal transport problem is a linear programming (LP) problem.

1.3 Literature review

1.3.1 Surrogate model prediction uncertainty

To reduce the computational effort of model-based optimization, data-driven surrogate models are commonly used to approximate the original full-order models in optimization frameworks [5]. Data-driven surrogate models have the advantage of being relatively low in complexity and can be constantly updated with new data. For example, the data-driven hinge hyperplane has been adopted as a surrogate model for real-time optimization of complex processes under uncertainties [33]. The nonlinear autoregressive exogenous (NARX) neural network has been utilized as a surrogate model for the optimization of an industrial-scale air and gas compression system [34]. The neural network has been employed to estimate plant gradients for a modifier adaptation method to address plant-model mismatch in real-time optimization of a gas-lifted oil well network [35]. For a comprehensive review of surrogate model-based optimization in the chemical engineering field, readers can refer to [36].

Among various data-driven surrogate models, the artificial neural network (ANN) stands out as a popular and efficient option since the ANN has demonstrated strong performance in approximating non-linear models. For instance, the ANN has been employed for the biodiesel production process modeling, which significantly reduces the process model complexity without sacrificing representativeness [37]. Additionally, ANN can be directly incorporated into optimization problems, as seen in studies on distillation energy efficiency optimization and heat-integrated crude oil distillation system optimization [38, 39]. In both cases, the sigmoid neural networks were used as the surrogate models to enable the complicated process optimization problems can be solved efficiently.

Although ANNs are powerful surrogate models, they heavily rely on the data used for training. This means that the prediction of the ANN model is greatly affected by the variation in the training data. Even with a large data set, the randomness in the collection of data cannot be fully eliminated. Meanwhile, it is challenging to determine if the collected data set is large enough to train an accurate ANN. As a result, the prediction uncertainty caused by the data variation should be considered in ANN surrogate model-based optimization to improve solution robustness.

Various methods have been proposed to estimate the prediction uncertainty of ANNs, including the Bayesian method [8], fuzzy method [9], Monte Carlo simulation method [40], and optimization-based method [41]. However, these approaches are generally difficult to be incorporated into an optimization framework due to their complex structures. Although the ensemble-based method [42] can be integrated into an optimization problem, it has a high computational cost due to the large-scale model involved. For a comprehensive review on existing techniques for handling ANN prediction uncertainty, readers can refer to [43]. Apart from the above approaches, the mixture density network (MDN) [44] is a more efficient and easier alternative to address ANN prediction uncertainty. The MDN can be thought of as a traditional ANN, with the only difference being that the outputs of the MDN are parameters of the prediction distributions (such as the means and standard deviations of ANN predictions) instead of single-point predictions. Therefore, the ANN prediction uncertainty can be estimated by the distribution parameters predicted from the MDN.

The MDN has been applied in multiple practical scenarios, as demonstrated by several studies. Men et al. [45] used an ensemble of MDNs for wind speed and power forecasting, testing the methodology with a data set obtained from a wind farm in Taiwan. Zhang et al. [46] employed the MDN to analyze the uncertainty in wind turbine power output. In Herzallah and Lowe's work [47], the MDN was used to model multi-component distributions for non-linear control problems. Ahangar et al. [48] proposed an innovative voice conversion algorithm based on the MDN. Sarochar et al. [49] generated energy consumption data using the MDN integrated with a multi-layered LSTM network for residential and commercial areas. Lastly, Vakanski et al. [50] evaluated human motions in physical therapy using the MDN, aiming to improve the therapy.

Although the MDN has a wide range of applications and it is a promising approach for estimating the model prediction uncertainty, its application in optimization considering model prediction uncertainty is still rare. Accordingly, the MDN is employed in this research for the proposed novel data-driven method addressing optimization under surrogate model prediction uncertainty. More details are elaborated in Chapter 2

1.3.2 Joint chance-constrained programming

Joint chance-constrained programming (JCCP) is more intuitive and useful than individual chance-constrained programming (ICCP). However, solving a JCCP problem is immensely challenging due to the need to handle complex multidimensional distributions. Accordingly, JCCP problems are generally solved by utilizing approximations. Analytical approximation methods [51] and sampling-based methods [14] are the two main approximation ways for the JCCP. Analytical approximation methods are employed to approximate a JCCP problem by using a deterministic optimization formulation [28]. One such approach is the robust optimization (RO) [52] technique which uses uncertainty sets to address uncertainties and does not require the full information of uncertainty distributions. For a comprehensive explanation of RO, readers can refer to [28]. On the other hand, sampling-based approaches contain scenario approximation and sample average approximation (SAA). The scenario approximation first generates random samples for uncertain parameters, and then it approximates the JCC by using constraints corresponding to each generated sample [53]. However, the infeasibility of the approximated JCC may arise due to the randomness of sample generation [28]. To overcome this issue, the SAA is developed as a generalization of the scenario approximation. In SAA, the random parameters in the problem studied are first sampled, and then the constraint satisfaction probability based on the samples is enforced to be above an acceptable level. SAA is a straightforward and effective approach that can enable chance-constrained problems to be tractable and solvable [14]. For a comprehensive elaboration of both scenario approximation and SAA, readers can refer to [53, 54].

While RO and SAA can be exploited to solve JCCP problems, they have certain drawbacks that limit their practical applications. As to RO, this approach may produce an overly conservative solution if the uncertainty set is not well-designed, and it is difficult to obtain a well-designed uncertainty set in practice [55]. Additionally, the SAA method can be computationally costly because the SAA model contains the indicator function including binary variables, which can lead to increased complexity and longer computation times, especially when the underlying optimization problem is non-linear [56].

To overcome the above restrictions of the existing methods for JCCP problems, two machine learning-based techniques are proposed in this research. More details are explained in Chapter 3.

1.3.3 Distributionally robust chance-constrained optimization

According to the introduction of DRCCP in Section 1.2.4, the ambiguity set is a key element of DRCCP models. Different types of ambiguity sets lead to different DRCCP approaches. Moment-based and metric-based ambiguity sets are commonly used in the field of DRCCP, as discussed in several studies [57–59]. A moment-based ambiguity set includes a collection of distributions that meet specific moment constraints. In order to establish such a set, a certain level of moment information needs to be known beforehand. Moreover, since different distributions may share the same moments, the moment constraints may not be precise enough to exclude irrelevant distributions, potentially leading to overly conservative decisions [60]. Furthermore, even with sufficient data, distributionally robust chance constraints based on moment-based ambiguity sets may not be able to accurately approximate the original chance constraints [61]. Metric-based ambiguity sets are more powerful alternatives to prevent the above limitations. The metric-based ambiguity sets can be thought of as balls in the space of probability distributions. In a metric-based ambiguity set, all candidate distributions are centered around a nominal distribution constructed by gathered samples, within a radius defined by a probability metric. The radius size of the ambiguity set is a user-defined hyper-parameter that determines the degree of conservatism in the solution. The general formulation of a metric-based ambiguity set can be defined as:

$$\mathcal{P}(\Xi) = \{\mathbb{P} : \mathcal{D}(\mathbb{P}, \mathbb{P}_0) \leq \varepsilon\} \tag{1.11}$$

where $\mathcal{P}(\Xi)$ here represents a metric-based ambiguity set. $\mathbb{P}$ and $\mathbb{P}_0$ are candidate and nominal distributions, respectively. $\mathcal{D}$ is a probability metric determining the similarity between $\mathbb{P}$ and $\mathbb{P}_0$. ε is the radius size of the ambiguity set which is a user-defined hyper-parameter.

The ϕ-divergence is a widely-used probability metric for metric-based ambiguity sets [62–64]. Ning and You [65] proposed a well-designed DRCCP approach exploiting a generative adversarial network (GAN) and the ϕ-divergence to construct an ambiguity set. Their approach employs an f-GAN to produce a continuous reference distribution (nominal distribution), with all candidate distributions in the ambiguity set required to be close to the reference distribution based on the ϕ-divergence. This method has been demonstrated to have excellent performance in power systems applications. However, as highlighted in existing literature [16, 66], an ambiguity set constructed using the ϕ-divergence only includes distributions that are absolutely continuous with respect to the nominal distribution. This means that the ϕ-divergence ambiguity set contains only those distributions with the same support as the nominal distribution. Thus, if the nominal distribution is continuous, there are no discrete distributions in the ϕ-divergence ambiguity set, and vice versa. This can be problematic if the true distribution has a different support than the nominal distribution, as the true distribution may not be included in the ambiguity set. Additionally, according to Gao and Kleywegt [66], the ϕ-divergence only considers the relative ratio between two distributions, and therefore cannot capture the distance between them. This limitation may cause irrelevant distributions to be included in the ϕ-divergence ambiguity set, leading to an overly conservative solution.

To address the issues of the ϕ-divergence ambiguity sets, the Wasserstein distance has become a popular alternative for constructing metric-based ambiguity sets [67–69]. The

Wasserstein distance is a metric that quantifies the minimum cost of transforming one probability distribution into another, where the cost is defined as the optimal transport distance obtained from solving the optimal transport problem between the two distributions. The Wasserstein ambiguity set can include both continuous and discrete distributions, making it capable of containing the true distribution with high confidence. Furthermore, because the Wasserstein distance possesses the capability of measuring the distance between two distributions, Wasserstein ambiguity sets are more effective in eliminating irrelevant distributions and preventing excessively conservative decisions compared to ϕ-divergence ambiguity sets [66]. However, existing studies of Wasserstein DRCCP rely on some non-trivial assumptions on uncertain constraints to attain tractable Wasserstein DRCCP models. Most of the current works on the Wasserstein DRCCP [3, 4, 61, 70–73] assume that constraints involving uncertainty are affine in uncertain parameters. The Wasserstein DRCCP method presented in Gu and Wang's research [74] restricts uncertain constraints to be quadratic-convex in uncertainty. The Wasserstein DRCCP approach proposed by Hota et al. [75] limits uncertain constraints to be either concave or convex in uncertainty. Apart from the restrictions on uncertain constraints, the worst-case distribution of the Wasserstein DRCCP is proven to be limited to a discrete distribution supported on at most $M + 1$ samples (M is the number of real gathered samples) [66, 76]. This limitation would cause the Wasserstein DRCCP to hedge against the wrong families of distributions if the true distribution is continuous [17]. The above restrictions seriously hinder the practical applications of the Wasserstein DRCCP.

To overcome the issues of the existing DRCCP approaches, two novel DRCCP methods are presented in this research. Moreover, a new, efficient, and widely compatible algorithm is further proposed to enhance the DRCCP solution quality, which is capable of removing outliers and extreme samples causing overly conservative solutions. More details are elucidated in Chapters 4, 5, and 6.

1.4 Thesis contribution

In this research, several novel data-driven methods have been developed to efficiently address optimization under uncertainty and to surmount the limitations of existing approaches for optimization involving uncertainty. The main contributions of this study are shown below:

- Chapter 2 presents an innovative mixture density network (MDN)-based technique for addressing optimization considering surrogate model prediction uncertainty. This approach outperforms other existing methods due to its higher computational efficiency and better applicability.

- Chapter 3 proposes two machine learning-based techniques to handle steady-state joint chance-constrained programming (JCCP) problems and dynamic joint chance-constrained stochastic optimal control problems (SOCPs). These two techniques have better performance and lower computational costs than other methods in the literature. The uncertainty distributions should be known in advance while using these techniques.

- Chapter 4 presents a new distributionally robust chance-constrained optimization (DRCCP) approach based on the kernel ambiguity set, which requires fewer assumptions and has better performance than the popular Wasserstein-based method. The performance of this proposed method can be further enhanced by combining it with a neural network-like deep kernel. This kernel-based approach can handle JCCP problems without knowing true uncertainty distributions.

- Chapter 5 proposes a new DRCCP approach based on the Sinkhorn ambiguity set. This method can hedge against more general families of uncertainty distributions, requires fewer assumptions, and can achieve a better solution with lower variability than the widely-used Wasserstein-based approach. This Sinkhorn-based method can address JCCP problems without obtaining exact uncertainty distributions.

- Chapter 6 presents a novel algorithm that can exclude outliers and extreme data samples leading to overly conservative DRCCP solutions. This algorithm is able to significantly enhance the DRCCP solution quality, which has wider compatibility with various DRCCP models than other existing similar approaches.

The main contributions of this thesis and the interrelationships between different developed methods are summarized in the following flowchart.

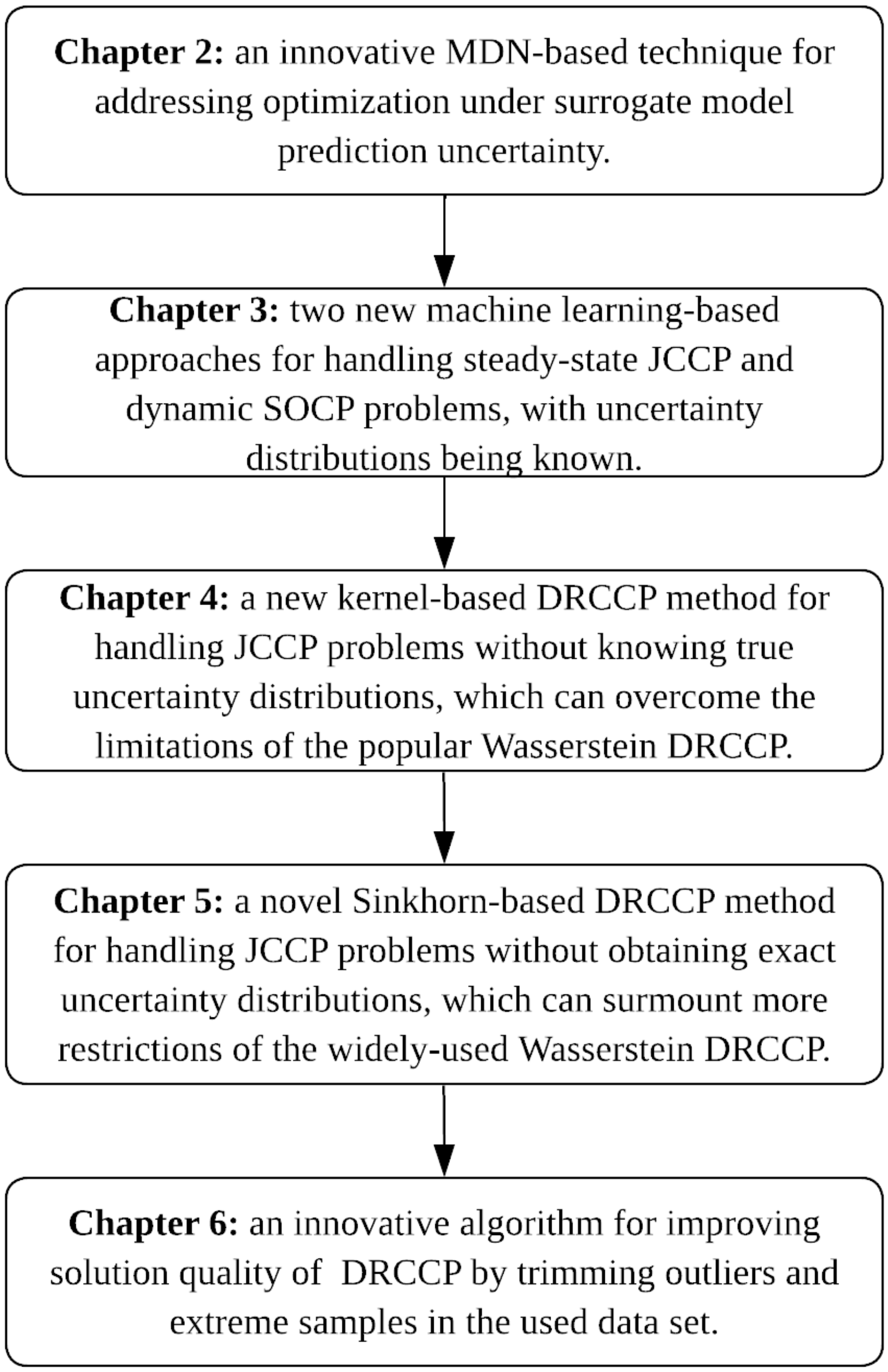

Chapter 2: an innovative MDN-based technique for addressing optimization under surrogate model prediction uncertainty.
Chapter 3: two new machine learning-based approaches for handling steady-state JCCP and dynamic SOCP problems, with uncertainty distributions being known.
Chapter 4: a new kernel-based DRCCP method for handling JCCP problems without knowing true uncertainty distributions, which can overcome the limitations of the popular Wasserstein DRCCP.
Chapter 5: a novel Sinkhorn-based DRCCP method for handling JCCP problems without obtaining exact uncertainty distributions, which can surmount more restrictions of the widely-used Wasserstein DRCCP.
Chapter 6: an innovative algorithm for improving solution quality of DRCCP by trimming outliers and extreme samples in the used data set.

Figure 1.4: Flowchart of this thesis

Chapter 2

Data-Driven Process Optimization Considering Surrogate Model Prediction Uncertainty: A Mixture Density Network-Based Approach

Abstract

The artificial neural network (ANN) can be effectively used as a data-driven surrogate model in process optimization. However, there is a problem that the change of training set leads to prediction uncertainty. A novel framework is proposed in this research to address this issue. In the proposed approach, an ensemble of ReLU ANNs is first trained with different training sets to simulate the prediction uncertainty caused by the training set variation. Then, a mixture density network (MDN) is used to approximate the ReLU ANN ensemble and it is further embedded into a mixed-integer linear optimization problem. The original optimization problem is reformulated into a chance-constrained form with the mean-variance type objective function to address both constraint and objective uncertainties. The proposed approach is applied to a numerical example and two case studies to show its capability of solving complex process optimization problems under the neural network model prediction uncertainty.

2.1 Introduction

With the advances made in machine learning and data science, data-driven modeling and optimization have received lots of attention in recent years. Various machine learning techniques have been used as data-driven surrogate modeling methods to improve the efficiency of solving complex optimization problems. While these techniques have received lots of studies in the past, uncertainties in generated surrogate models have been widely ignored. Therefore, how to handle the uncertainty in the surrogate model and address it in an optimization framework is still an issue worth exploring in depth.

In terms of the data-driven surrogate model, it is utilized to approximate the original full-order model to reduce the computation effort [5] due to its relatively low complexity and ability to be updated constantly with the collected data. For instance, the data-driven hinge hyperplane is adopted as the surrogate model for real-time optimization (RTO) of complex processes under uncertainties [33]. Lee et al. [34] adopted the Nonlinear Autoregressive eXogenous (NARX) neural network to optimize the operating conditions of an industrial-scale air and gas compression system in a commercial terephthalic acid manufacturing plant. The neural network is utilized to estimate the plant gradient for the modifier adaptation method to cope with the plant-model mismatch in the studied RTO of a gas-lifted oil well network [35].

Among numerous types of data-driven surrogate models, the artificial neural network (ANN) is a widely used and efficient surrogate model which has been adopted in several studies of process simulation and optimization [77–81]. ANN is evidenced to have a strong ability for the approximation of the non-linear model. According to Fahmi's research [37], the ANN can significantly reduce the complexity of the highly non-linear biodiesel production process model without losing its representativeness. Furthermore, the ANN can be directly formulated in an optimization problem. For instance, in Osuolale's study [38], the difficult non-linear optimization of the distillation energy efficiency can be simplified and solved efficiently with the use of the sigmoid neural network as the surrogate model. Also, according to Ochoa-Estopier's work [39], an innovative optimization approach for optimizing heat-integrated crude oil distillation systems with the sigmoid neural network embedded is proposed.

On the other hand, since integer decision variables are often involved in the process optimization, many optimization problems in PSE can be modelled as mixed-integer problems. In terms of a mixed-integer optimization problem involving the ANN, it will become a mixed-integer non-linear programming (MINLP) problem if the non-linear activation function, such as the sigmoid function or hyperbolic tangent function, is adopted in the ANN. In order to further simplify the MINLP optimization involving the ANN, the piecewise-linear rectified linear unit (ReLU) activation function can be used to simplify the studied optimization to be a mixed-integer linear programming (MILP) problem. Also, according to Grimstad's study [82], the ReLU ANN can approximate the complex non-linear model accurately. Therefore, based on the mentioned advantages of the ReLU ANN, the ReLU ANN is adopted as the surrogate model in this work for the purpose of simplicity.

However, although ANNs are powerful approaches to approximate non-linear processes, they are data-driven methods that are significantly influenced by the training set variation. The ANN model prediction uncertainty is caused by the training set variation. A large data set cannot guarantee an uncertainty-free prediction since the randomness in data collection cannot be completely avoided by increasing the data set size. Also, it is hard to know whether the collected data set is large and comprehensive enough to train an accurate ANN. Thus, the model prediction uncertainty should be taken into account for the ANN surrogate model-based optimization, in order to improve the solution robustness.

To estimate the prediction uncertainty of the ANN, various methods have been proposed, such as the Bayesian method [7, 8, 83–86], fuzzy method [9], Monte Carlo simulation method [40, 87, 88], and optimization-based method [41]. However, those methods are generally difficult to be incorporated into an optimization problem. Another potential approach to handle the prediction uncertainty is the ensemble-based method [6, 42]. This method can be applied to an optimization problem but its computation cost is very high due to the large-scale model involved in this method. More details about the ensemble-based method are elaborated in Section 2.2.3. In addition to the above-mentioned approaches, the ANN prediction uncertainty can be addressed using a more efficient and easier method called the mixture density network (MDN) [89]. The concept of the MDN is very easy-to-understand. The MDN can be thought of as the traditional ANN, and the only difference between the MDN and traditional ANN is that the outputs of the MDN are parameters of the prediction distributions (e.g., the means and standard deviations of ANN predictions) instead of exact prediction values. Therefore, the ANN prediction uncertainty can be estimated by the distribution parameters predicted from the MDN. More details about the MDN will be illustrated in Section 2.2.4.

The practical application of the MDN has been presented in many studies. According to Men's work [45], an ensemble of MDNs is used for wind speed/power forecasting and the proposed methodology was tested by the data set obtained from a wind farm in Taiwan. In Zhang's study [46], the MDN is adopted for the uncertainty analysis of the power of a wind turbine. Herzallah and Lowe [47] adopted the MDN to model multi-component distributions for exploiting uncertainties in non-linear control problems. As can be seen from Ahangar's work [48], an innovative voice conversion algorithm based on the MDN is proposed in this research. In Sarochar's study [49], the energy consumption data of residential and commercial areas is generated by using the MDN integrated with a multi-layered Long Short-Term Memory (LSTM) network. According to Vakanski's work [50], the MDN is used to evaluate human motions in physical therapy for further improvement in the studied therapy.

In this study, we proposed an innovative optimization method based on ReLU MDN, to address the prediction uncertainty sourced from the variation of the training data set of ReLU ANN. While applying the proposed approach to an optimization problem, an ensemble of ReLU ANNs are first trained individually with different training sets to approximate non-linearities in the studied optimization problem as surrogate models, and to simulate the prediction uncertainty caused by the training set variation. Afterwards, the obtained ReLU ANN ensemble is then approximated by the MDN. Moreover, the studied problem is further reformulated to be the MILP chance-constrained optimization with the mean-variance type objective function and MDN embedded.

Since many challenging PSE optimization problems involve complex process models, the ReLU ANN provides a good candidate for such applications due to its strong flexibility in approximating complex functions and its explicit mixed integer linear formulation. Applications of these MILP surrogate model-based formulations to PSE optimizations are still rare so far. Additionally, an efficient way to address the ANN model prediction uncertainty through an explicit deterministic optimization framework is not found. For the above reasons, the novel MDN-based approach is developed in this work, which has potential applications in wide areas including PSE. The advantage of the proposed approach are as follows: 1) it provides a way to capture solution robustness under the prediction uncertainty, especially when the training data set is limited as demonstrated in Section 2.3.3. 2) The proposed approach relies on solving a single deterministic MILP problem, which is easy for implementation and shows computationally advantage as shown in Section 2.3.2.

The proposed approach is presented in Section 2.2. A numerical illustrating example is presented in Section 2.3, where the robustness and reliability of the optimal solution are studied. Also, the proposed approach is compared with the ANN ensemble-based optimization method in this illustrating example to demonstrate its performance. Moreover, the presented optimization method is applied to two case studies in Section 2.4 to show that robust optimal solutions of complex processes can be achieved under the ANN prediction uncertainty derived from the training set variation. The first case study is to minimize the total annual cost (TAC) of a water-ethanol distillation column with two specification constraints. The second case study is to solve a more complicated process optimization that is the product yield maximization of an ethylene glycol production process with two constraints.

2.2 Methods

In this section, we first present the mixed-integer linear constraint formulation of ReLU ANN and its usage in the deterministic optimization problem. Then, we investigate the prediction uncertainty of ReLU ANN and present the proposed approach of optimization under such uncertainty based on the mixture density network.

2.2.1 ReLU ANN

The ANN is a computational model having a fully connected feed-forward network structure. A detailed explanation of the ANN is in Section 1.2.1. The ANN employing the rectified linear unit (ReLU) transfer function is called the ReLU ANN. The ReLU transfer function is shown as (2.1).

$$\sigma\left(y\right) := \max\left\{0, y\right\} \tag{2.1}$$

For the output layer in the ReLU ANN, the corresponding transfer function of each neuron is given as (2.2).

$$\sigma\left(y\right) := y \tag{2.2}$$

Some important properties of ReLU ANNs that make them attractive as surrogate models in optimization are highlighted below. First, ReLU ANNs are composed of max-affine operators and are piecewise-linear as well as continuous functions [90] which means that a ReLU ANN can be precisely formulated in a MILP problem. In contrast, ANNs composed of other non-linear activation functions, such as sigmoid or hyperbolic functions, can only be expressed approximately in a MILP problem. Thus, ANNs involving non-linear activation functions are usually embedded in MINLP problems instead of MILP problems. Second, the approximation ability of the ReLU ANN can be controlled by adjusting its numbers of layers and neurons in a layer [91]. Finally, many toolboxes are available for designing, modelling, and training ReLU ANNs (e.g., TensorFlow [92], PyTorch [93]). In this research, MATLAB Neural Network Toolbox [94] is utilized for modeling and training different neural networks.

2.2.2 Deterministic optimization with ReLU ANN surrogate model

Incorporating the ReLU ANN as the surrogate model to the optimization framework has been widely studied in the field of machine learning [95–97]. However, such an application is still rare in the field of PSE. Therefore, it is innovative and promising to apply the ReLU ANN for the improvement of solving PSE optimization problems. Before illustrating the optimization involving the ReLU ANN, let's start with the following optimization formulation first:

$$\min_{x} \quad f(x) \tag{2.3a}$$

$$\text{s.t.} \quad g_i(x) \leq 0 \qquad i = 1, ..., w \tag{2.3b}$$

w is the number of inequality constraints. $f(x)$, $g_i(x)$, and x are the objective function, inequality constraints, and decision variables in the optimization problem, respectively. Note that $f(x)$ and $g_i(x)$ are assumed to be complex nonlinear functions that will be approximated by the ReLU ANN. The optimization problem involving a ReLU ANN can be expressed below. The non-linearities in the original optimization formulation ($f(x)$ and $g_i(x)$ given in (2.3a)-(2.3b)) are approximated using the ReLU ANN and corresponding approximations are symbolized as $\hat{f}$ and $\hat{g}_i$, respectively. In the model, symbols with hats represent predictions or approximations computed from the ReLU ANN.

$$
\begin{aligned}
\min \quad & \hat{f} && && \text{(2.4a)}\\
\text{s.t.} \quad & \hat{g}_i \leq 0 && i = 1, ..., w && \text{(2.4b)}\\
& H_s^0 = x_s && s = 1, ..., S && \text{(2.4c)}\\
& a_j^k = \sum_{l=1}^{L} W_{lj}^k H_l^{k-1} + b_j^k && j = 1, ..., J, \quad k = 1, ..., K && \text{(2.4d)}\\
& 0 \leq H_j^k \leq M(1 - z_j^k) && && \text{(2.4e)}\\
& a_j^k \leq H_j^k \leq a_j^k + M z_j^k && && \text{(2.4f)}\\
& z_j^k \in \{0, 1\} && && \text{(2.4g)}\\
& Y_r = \sum_{l=1}^{L} W_{lr}^K H_l^{K-1} + b_r^K && r = 1, ..., R && \text{(2.4h)}\\
& Y_r = [\hat{f}, \hat{g}_i] && && \text{(2.4i)}
\end{aligned}
$$

As to the above formulation, equations (2.4c) and (2.4h) represent the input and output layers of the embedded ReLU ANN, respectively. Hidden layers are expressed as (2.4d)-(2.4g) where the max-affine operators $H_j^k = \max\{0, a_j^k\}$ in the ReLU ANN are expressed through binary variables z_j^k. Equation 2.4i states that outputs of the embedded ReLU ANN are $\hat{f}$ and $\hat{g}_i$. H_s^0 and x_s are outputs of neurons in the input layer and corresponding decision variables, respectively. s is the index of each neuron in the input layer. S is the number of neurons in the input layer. j, k, and l are the index of each neuron in the current layer, the index of the layer, and the index of each neuron in the previous layer, respectively. J, K, and L are the number of neurons in the current layer, the number of layers (excluding the input layer), and the number of neurons in the previous layer, respectively. Note that both J and L are different at different k. For instance, when k equals to 1 in (2.4d), J and L are equivalent to a user-defined value and S, respectively. a_j^k, W_{lj}^k, W_{lr}^K, H_l^{k-1}, H_l^k, H_l^{K-1}, and Y_r are linear combinations of outputs from the previous layer, weights between the current and previous layers, weights between the last hidden layer and output layer, the output of each neuron in the previous layer, the output of each neuron in the current layer, the output of each neuron in the last hidden layer, and the output of each neuron in the output layer, respectively. b_j^k and b_r^K are biases for a certain hidden layer and the output layer, respectively. R is the number of neurons in the output layer. z_j^k are binary variables for max-affine operators in the ReLU ANN. M is a big number. The above formulation enables the direct compilation of the ReLU ANN into a MILP optimization problem.

2.2.3 Methods for addressing ANN prediction uncertainty

To further enhance the robustness of the solution to the optimization involving the ReLU ANN, the ReLU ANN prediction uncertainty caused by the training set variation cannot be ignored. The necessity of taking into account the prediction uncertainty is demonstrated in Section 2.3.3. Several approaches are proposed to deal with the ANN prediction uncertainty, such as the Bayesian method [7, 8, 83–86], fuzzy method [9], Monte Carlo simulation method [40, 87, 88], and optimization-based method [41]. However, these methods are too complicated to be applied to an optimization problem. Another potential approach named the ensemble-based method [6] can be compiled into an optimization model to cope with the prediction uncertainty of the embedded ANN. Thus, the ensemble-based method is selected to be compared with the proposed optimization approach in Section 2.3.2 of this research.

In order to employ the ensemble-based method, an ANN ensemble should be first produced. The ANN ensemble is composed of multiple different ANNs trained separately with different training sets. Such different training sets are sampled from a collected data set utilizing the bootstrap method with replacement. After giving a certain input to the ANN ensemble, various predictions are obtained. Then, the ANN prediction uncertainty originating from the training set variation based on the given input, can be described by the mean and standard deviation of the obtained predictions. Therefore, the ANN ensemble can be incorporated into an optimization framework to address the incorporated ANN prediction uncertainty. The above illustration is the entire conception of the ensemble-based method, and its practical implementation with more details is demonstrated in Section 2.3.2.

2.2.4 Mixture Density Network

Although adopting the ReLU ANN ensemble to handle the prediction uncertainty in an optimization problem is achievable, it may lead to a large-scale optimization model that would be time-consuming to solve, and it is illustrated in Section 2.3.2. A more efficient way to cope with the prediction uncertainty in an optimization framework is to use the mixture density network (MDN) to approximate the ReLU ANN ensemble as well as embedding the MDN in the studied optimization problem. While utilizing the MDN to evaluate the prediction uncertainty in an optimization problem, only one network is embedded in the optimization formulation that makes the computation burden loadable. Based on the above fact, the MDN is a promising approach and it is adopted in this work to handle the ReLU ANN prediction uncertainty originating from the training set variation in studied optimization problems.

The structure of the MDN is shown in Figure 2.1. The mathematical formulation of the MDN is the same as the traditional ANN mentioned in Section 2.2.1. The distinction between the MDN and traditional ANN is that the outputs of the MDN are parameters constructing the prediction distributions (e.g., means and standard deviations of the predictions) instead of exact prediction values. One remark here is that although weight factors are often included in outputs of MDNs [98], they are not considered in this work for the purpose of simplicity. In addition, the ReLU activation function is used in hidden layers of the MDN in this study.

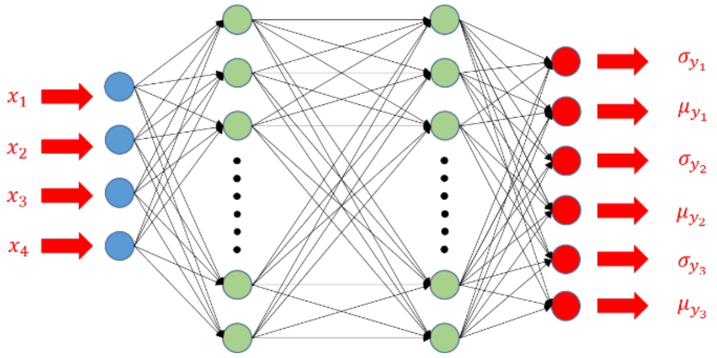


Figure 2.1: Schematic diagram of the mixture density network

To establish the MDN for the estimation of the prediction uncertainty in this work, a ReLU ANN ensemble should be generated first to produce training and validation sets for the MDN. More specifically, after generating the ReLU ANN ensemble, numerous observations are input into the ensemble to produce numerous predictions, and then corresponding means and standard deviations of the predictions can be computed. Then, the obtained means and standard deviations of the predictions paired with corresponding observations are used as training and validation sets for the MDN. Subsequently, the ReLU ANN prediction uncertainty can be estimated with the distributions constructed by the prediction means and standard deviations predicted from the MDN. Such distributions are assumed to be Gaussian distributions in this research and the feasibility of this assumption is elaborated in Section 2.3 and Section 2.4.

2.2.5 Optimization under uncertainty using MDN surrogate model

In order to handle the prediction uncertainty of the ReLU ANN embedded in the optimization framework by using the MDN, the optimization formulation is reformulated from the original form to the chance-constrained form with the mean-variance type objective function which is given below:

$$\begin{aligned} \min \quad & \mu_{\hat{f}} + \lambda\sigma_{\hat{f}} && && (2.5a) \\ \text{s.t.} \quad & Pr(\hat{g}_i \leq 0) \geq 1 - \varepsilon && i = 1, ..., w && (2.5b) \end{aligned}$$

$\mu_{\hat{f}}$, $\sigma_{\hat{f}}$, λ, and Pr are the mean of $\hat{f}$ in (2.4a), standard deviation of $\hat{f}$, weight factor of $\sigma_{\hat{f}}$, and probability measure, respectively. The mean-variance type objective function in (2.5a) is exploited to find a trade-off between solution quality and variability. Since $\mu_{\hat{f}}$ and $\sigma_{\hat{f}}$ could be in different scales, it should be addressed by selecting an appropriate value of λ to balance the two terms in (2.5a). Hence, the λ value can be chosen properly once scales of $\mu_{\hat{f}}$ and $\sigma_{\hat{f}}$ are known. If λ is too small, the solution variability will be overlooked and the solution robustness will be reduced significantly. If λ is too large, it will lead to an overly conservative solution. Both conditions are demonstrated in Sections 2.3.1 and 2.4.1.1. On the other hand, equation (2.5b) means that the probability of satisfying inequality constraints in (2.4b) should be greater than or equal to a certain confidence value $1 - \varepsilon$. For example, if $Pr(\hat{g}_i \leq 0)$ is greater than or equal to 0.9 with a given x, it is predicted that 90 percent or more of predictions $\hat{g}_i$ computed from the ReLU ANNs in the ensemble satisfy the constraint in (2.4b) based on the given x. The predicted probability is related to the means and standard deviations computed from the MDN because the prediction uncertainty is simulated using ReLU ANN ensemble and this ensemble is further approximated by the MDN.

Moreover, equation (2.5b) can be further reformulated into the deterministic form, and the corresponding derivation is shown as follows:

1. Using standardization of the random $\hat{g}_i$:

$$Pr\left(\frac{\hat{g}_i - \mu_{\hat{g}_i}}{\sigma_{\hat{g}_i}} \leq \frac{0 - \mu_{\hat{g}_i}}{\sigma_{\hat{g}_i}}\right) \geq 1 - \varepsilon \tag{2.6}$$

   $\mu_{\hat{g}_i}$ and $\sigma_{\hat{g}_i}$ are the mean and standard deviation of $\hat{g}_i$ based on a given x.

2. Using the cumulative distribution function $\phi(\cdot)$ of the standard normal distribution,

equation (2.6) can be rearranged to be:

$$\phi\left(\frac{0-\mu_{\hat{g}_i}}{\sigma_{\hat{g}_i}}\right) \geq 1-\varepsilon \tag{2.7}$$

3. Subsequently, the inverse cumulative distribution function is applied:

$$\frac{0-\mu_{\hat{g}_i}}{\sigma_{\hat{g}_i}} \geq \phi^{-1}(1-\varepsilon) \Longrightarrow -\mu_{\hat{g}_i} \geq \phi^{-1}(1-\varepsilon)\sigma_{\hat{g}_i} \tag{2.8}$$

Since $\mu_{\hat{f}}$, $\sigma_{\hat{f}}$, $\mu_{\hat{g}_i}$, and $\sigma_{\hat{g}_i}$ are predicted by the MDN, the chance-constrained optimization given in (2.5a)-(2.5b) should incorporate the MDN as well as being reformulated to be a MILP chance-constrained optimization which is given as follows:

$$\begin{aligned}
\min \quad & \mu_{\hat{f}} + \lambda\sigma_{\hat{f}} && && \text{(2.9a)}\\
\text{s.t.} \quad & -\mu_{\hat{g}_i} \geq \phi^{-1}(1-\varepsilon)\sigma_{\hat{g}_i} && i=1,...,w && \text{(2.9b)}\\
& H_s^0 = x_s && s=1,...,S && \text{(2.9c)}\\
& a_j^k = \sum_{l=1}^{L} W_{lj}^k H_l^{k-1} + b_j^k && j=1.,..,J, \quad k=1,...,K && \text{(2.9d)}\\
& 0 \leq H_j^k \leq M(1-z_j^k) && && \text{(2.9e)}\\
& a_j^k \leq H_j^k \leq a_j^k + M z_j^k && && \text{(2.9f)}\\
& z_j^k \in \{0,1\} && && \text{(2.9g)}\\
& Y_r = \sum_{l=1}^{L} W_{lr}^K H_l^{K-1} + b_r^K && r=1,...,R && \text{(2.9h)}\\
& Y_r = [\mu_{\hat{f}}, \sigma_{\hat{f}}, \mu_{\hat{g}_i}, \sigma_{\hat{g}_i}] && && \text{(2.9i)}
\end{aligned}$$

As can be seen from the above formulation, equations (2.9c)-(2.9i) express the embedded MDN in the MILP chance-constrained optimization. More specifically, equations (2.9c)-(2.9h) are the same as (2.4c)-(2.4h), and (2.9i) states that $\mu_{\hat{f}}$, $\sigma_{\hat{f}}$, $\mu_{\hat{g}_i}$, and $\sigma_{\hat{g}_i}$ are the outputs of the MDN.

To briefly summarize the ReLU MDN-based optimization approach mentioned above, the prediction uncertainty is first simulated by an ensemble of ReLU ANNs. Afterwards, such a ReLU ANN ensemble is further approximated by an MDN for the reduction of the computation load, and then the MDN is incorporated into the studied optimization framework. Finally, the studied optimization formulation is further reformulated to be a MILP chance-constrained optimization with the mean-variance type objective function and MDN embedded to enforce the optimal solution robustness under the prediction uncertainty.

2.2.6 Workflow

The procedure for conducting the presented optimization approach is explained below:

1. Collect the data set from the studied system.

2. Generate different training data sets resampled from the collected data set using the bootstrap method (with the replacement). Train multiple different ReLU ANNs using those training sets separately. After training ReLU ANNs, a ReLU ANN ensemble is obtained. The ReLU ANN ensemble is utilized to simulate the prediction uncertainty caused by the training data set variation. One remark here is that other types of activation function can also be used for ANNs in the ensemble to simulate the prediction uncertainty. The ReLU activation function is used for the ANNs in the ensemble in this study since the prediction uncertainty of the ReLU ANN surrogate model is investigated in this research.

3. Input observations into the ReLU ANN ensemble to generate predictions. The means and standard deviations of the generated predictions with respect to corresponding observations are further calculated. Then, calculated means and standard deviations paired with corresponding observations are adopted as training and validation data for the MDN.

4. Train the MDN to approximate the ReLU ANN ensemble with the training data set obtained from step 3. Note that the ReLU activation function is used in hidden layers of the MDN.

5. Incorporate the MDN into the studied optimization framework as well as reformulating the original optimization formulation to the chance-constrained form with the mean-variance type objective function. The chance-constraints and mean-variance type objective function are utilized to cope with constraint satisfaction uncertainty as well as finding a trade-off between solution quality and variability, respectively.

6. Solve the above chance-constrained optimization.

The above procedure is also organized as the flowchart shown in Figure 2.2.

In order to demonstrate that the robustness of the obtained optimal solution from the above workflow can be adjusted by tuning $1 - \varepsilon$ and λ which are the design parameters in the proposed optimization approach, the above-mentioned chance-constrained optimization involving the MDN can be solved several times with different $1 - \varepsilon$ and λ. By this mean, the variation of the optimal solution with respect to different $1 - \varepsilon$ and λ can be observed. Moreover, the obtained optimal decision variables from the chance-constrained optimization can be further plugged into the ReLU ANN ensemble and original optimization formulation for validating predictions of the MDN embedded in the chance-constrained optimization based on different $1 - \varepsilon$ and λ. Note that the optimal decision variables and solutions mentioned in the following part of this work are indicated as the optimal decision variables and solutions gained from the chance-constrained optimization involving the MDN based on different $1 - \varepsilon$ and λ. More related details are illustrated in the following sections.

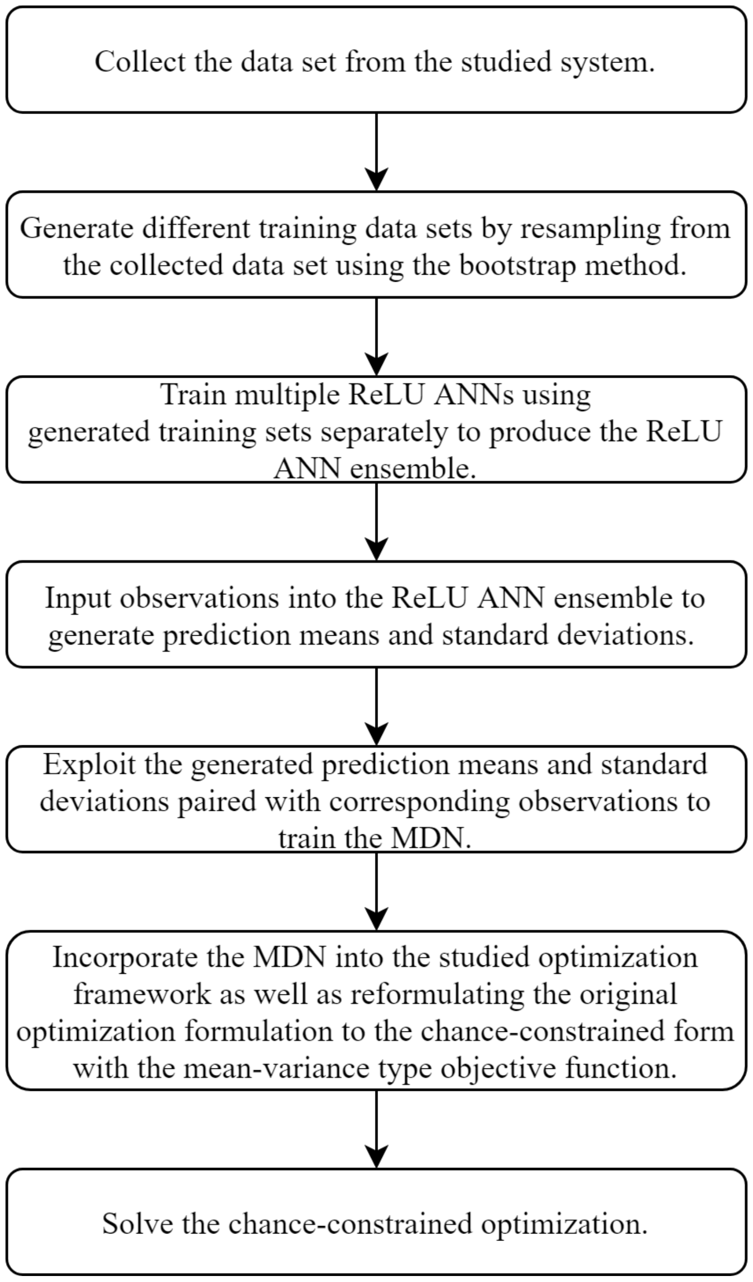

Collect the data set from the studied system.
Generate different training data sets by resampling from the collected data set using the bootstrap method.
Train multiple ReLU ANNs using generated training sets separately to produce the ReLU ANN ensemble.
Input observations into the ReLU ANN ensemble to generate prediction means and standard deviations.
Exploit the generated prediction means and standard deviations paired with corresponding observations to train the MDN.
Incorporate the MDN into the studied optimization framework as well as reformulating the original optimization formulation to the chance-constrained form with the mean-variance type objective function.
Solve the chance-constrained optimization.

Figure 2.2: Flowchart of the proposed optimization approach

2.3 Illustrating example

Consider a numerical constrained non-linear non-convex optimization problem given as:

$$\min_{x,y} \quad f(x,y) = e^x(4x^2 + 2y^2 + 4xy + 2y - 1) \tag{2.10a}$$

$$\text{s.t.} \quad g_1(x,y) = \frac{xy}{2} + (x+2)^2 + \frac{(y-2)^2}{2} \leq 2 \tag{2.10b}$$

$$g_2(x,y) = (x+5)^2 + y^2 \leq 25 \tag{2.10c}$$

$$-5.3 \leq x \leq -0.6 \tag{2.10d}$$

$$0.6 \leq y \leq 5.3 \tag{2.10e}$$

where $f(x,y)$, $g_1(x,y)$, and $g_2(x,y)$ are non-linearities in the above optimization problem. Equations (2.10d) and (2.10e) are bounds for decision variables. The local and global optimums of this problem are already known and are shown in Figure 2.3. The local optimum is $x = -0.8210$ and $y = 0.6696$ with objective value 0.7626. The global optimum is located at $x = -5.2813$ and $y = 4.6815$ with objective value 0.3299.

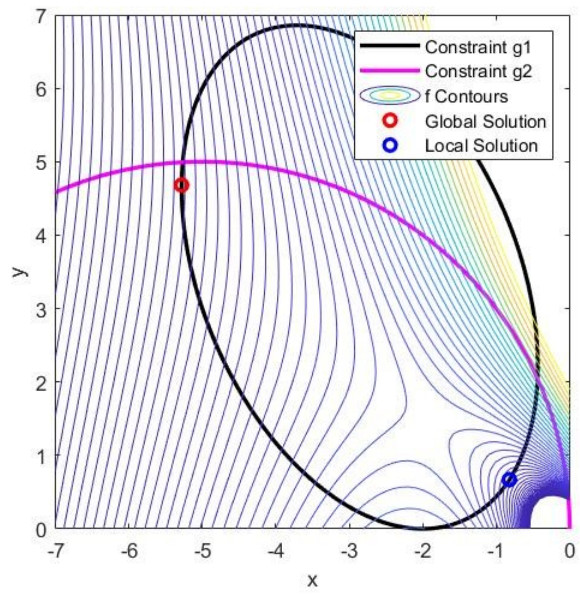


Figure 2.3: Local and global optimums of the illustrating example

To apply the presented ReLU MDN-based optimization approach to this numerical optimization problem, the data for training ReLU ANNs in the ensemble should be first generated. 55696 samples containing the value of $f(x, y)$, $g_1(x, y)$, and $g_2(x, y)$ paired with corresponding decision variables x and y are gathered from (2.10a)-(2.10c) as the data set. This data set is collected with the decision variable x in the range of -5.3 to -0.6 with an interval of 0.02 and decision variable y in the range of 0.6 to 5.3 with an interval of 0.02. Afterwards, as shown in Figure 2.4, 100 different ReLU ANNs are trained individually with 100 different training sets and each training set includes 44557 samples. These 100 training sets are resampled from the mentioned data set using the bootstrap method with replacement. Subsequently, the ReLU ANN ensemble including 100 ReLU ANNs used to approximate non-linearities in this optimization problem is obtained. Next, the ReLU ANN ensemble is adopted to simulate the ReLU ANN prediction uncertainty caused by the training data set variation. The prediction uncertainty is assumed to follow Gaussian distribution and this assumption is validated to be feasible as shown in Figure 2.6 and Figures A1a-A1f in Section A1.1 in Appendix (based on $x = -2.18$ and $y = 2.14$). On the other hand, all ReLU ANNs in the ensemble are composed of 2 hidden layers. The first and second hidden layers of each ReLU ANN include 50 and 30 neurons, respectively. Meanwhile, there are 2 inputs (x and y) and 3 outputs (ReLU ANN predictions of $f(x, y)$, $g_1(x, y)$, and $g_2(x, y)$ which are $\hat{f}$, $\hat{g}_1$, and $\hat{g}_2$, respectively) in each ReLU ANN in the ensemble.

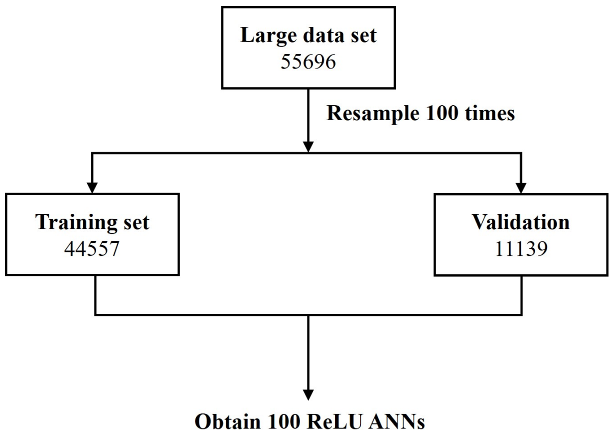


Figure 2.4: Schematic diagram for illustrating the production of the ReLU ANN ensemble in the illustrating example

Since it is not practical to embed the entire ReLU ANN ensemble into an optimization formulation, the ReLU ANN ensemble should be further approximated by the MDN. To generate the training and validation sets for the MDN, 55696 inputs (one input includes one set of x and y) are input into the ReLU ANN ensemble and then $55696 \times 100 \times 3$ predictions are produced due to the 55696 inputs, 100 ReLU ANNs in the ensemble, and 3 outputs of each ReLU ANN in the ensemble. Thus, 55696×3 sets of prediction means and standard deviations are generated as well as paired with corresponding input to form the training and validation sets for the MDN. A schematic diagram for demonstrating the procedure of producing training and validation sets for the MDN is shown in Figure 2.5. As to the network structure of the MDN, two hidden layers are included as well as 50 and 30 neurons in the first and second layers, respectively. Meanwhile, there are 2 inputs (x and y) and 6 outputs (means and standard deviations of 3 ReLU ANN predictions, which are $\mu_{\hat{f}}$, $\sigma_{\hat{f}}$, $\mu_{\hat{g}_1}$, $\sigma_{\hat{g}_1}$, $\mu_{\hat{g}_2}$, and $\sigma_{\hat{g}_2}$) in the MDN. The mean absolute percentage errors (MAPEs) of the trained MDN based on the validation set are shown in Table A1 (Section A1.1 in Appendix).

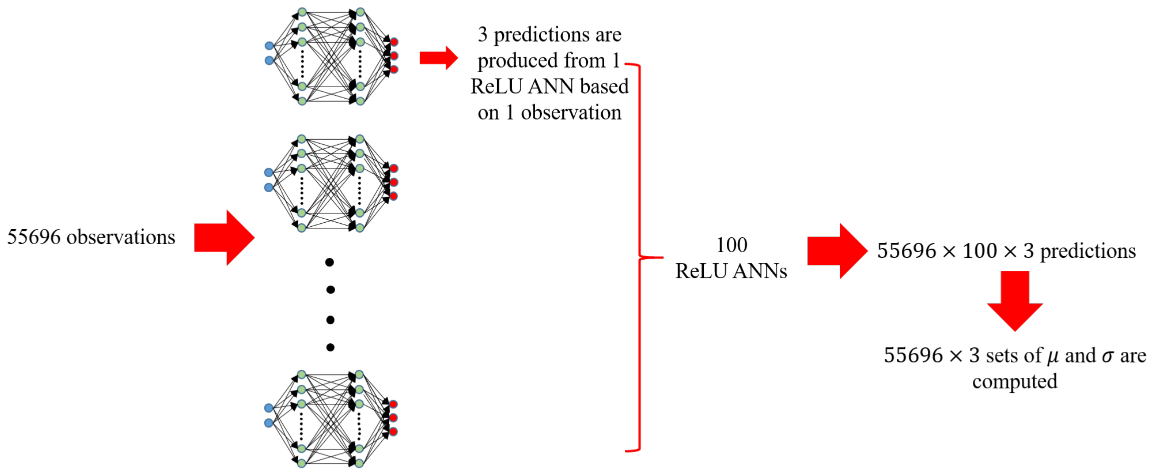

3 predictions are produced from 1 ReLU ANN based on 1 observation
55696 observations
100 ReLU ANNs
55696 × 100 × 3 predictions
55696 × 3 sets of μ and σ are computed

Figure 2.5: Procedure of generating training and validation sets for the MDN

After training the MDN, the original numerical optimization problem is transformed into a chance-constrained optimization with the mean-variance type objective function and trained MDN embedded. The final MILP formulation is reported in Appendix (equations (A1a)-(A1k) in Section A1.1).

The MILP chance-constrained optimization problem is solved several times with different $1-\varepsilon$ and λ to study the optimal solution robustness by tuning these design parameters ($1-\varepsilon$ and λ). The problem is solved utilizing CPLEX solver in GAMS. With respect to different optimal decision variables based on different $1-\varepsilon$ and λ, corresponding prediction means and standard deviations predicted from the MDN are listed in Table A2 (Section A1.1 in Appendix). Afterwards, the optimal decision variables in Table A2 are input into the ReLU ANN ensemble. The means and standard deviations of predictions from the ReLU ANNs in the ensemble with respect to different sets of optimal decision variables, are listed in Table A3 (Section A1.1 in Appendix).

Finally, based on different sets of optimal decision variables, percentages of predictions from the ReLU ANN ensemble satisfying both original constraints in (2.10b) and (2.10c) (constraint satisfaction probability) can be obtained. Moreover, the optimal decision variables are further plugged into the original problem formulation (formulated in (2.10a)-(2.10e)) to observe real conditions of constraint satisfaction as well as robustness of the optimal solutions. These results are listed together in Table A4 (Section A1.1 in Appendix).

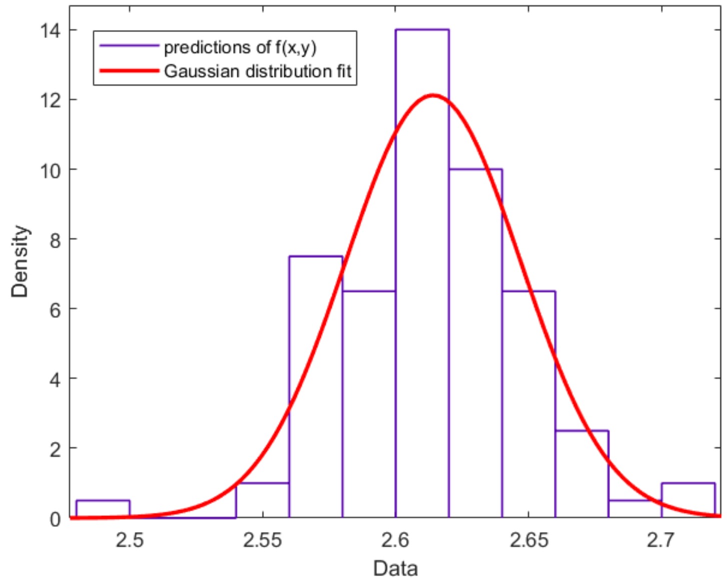


Figure 2.6: Empirical probability distribution of output prediction from ANN ensemble

2.3.1 Discussion

Based on the results obtained from the illustrating example, impacts of varying $1-\varepsilon$ and λ on the optimal solution are discussed and analyzed comprehensively.

The impact of changing λ:

- **Impact of changing λ based on the MDN computation:** According to the MDN computation, $\mu_{\hat{f}}$ increases with λ since the weight of $\sigma_{\hat{f}}$ in (A1a) (Section A1.1 in Appendix) increases and the minimization of equation (A1a) focuses more on minimizing

$\sigma_{\hat{f}}$. Thus, $\mu_{\hat{f}}$ increases and $\sigma_{\hat{f}}$ decreases as λ increases. Moreover, while $\mu_{\hat{f}}$ grows with λ, corresponding $\mu_{\hat{g}_1}$ and $\mu_{\hat{g}_2}$ decrease as well as staying away from the upper limits in (2.10b) and (2.10c), respectively (2 and 25, respectively). The reason is that the global optimum of the original optimization problem locates on the first constraint (in other words, the first constraint is active).

- **Impact of changing λ on the solution robustness:** The optimal solution robustness can be improved by increasing λ due to the above-mentioned reasons. However, too large λ may cause an overly conservative solution because the solution variability is overly weighted in the objective function. As can be seen from Figure 2.7, overly conservative solutions with 100% constraint satisfaction probabilities are gained while $\lambda = 100$. More satisfactory solutions with constraint satisfaction probabilities closer to the required confidence levels, are obtained while $\lambda = 50$. To avoid an overly conservative solution, the ideal λ value should be able to make $\sigma_{\hat{f}}$ have a similar size to $\mu_{\hat{f}}$.
- **Impact of changing λ based on computations of the ReLU ANN ensemble and original problem formulation:** After inputting the optimal decision variables into the ReLU ANN ensemble, changes of $\mu_{\hat{f}}$, $\sigma_{\hat{f}}$, $\mu_{\hat{g}_1}$, and $\mu_{\hat{g}_2}$ based on the ReLU ANN ensemble computation with increasing λ are similar to the changes based on the MDN computation. Also, after inputting the optimal decision variables into the original problem formulation, variations of $f(x, y)$, $g_1(x, y)$, and $g_2(x, y)$ calculated from the original problem formulation with growing λ, are similar to the changes of $\mu_{\hat{f}}$, $\mu_{\hat{g}_1}$, and $\mu_{\hat{g}_2}$ obtained from the MDN and ReLU ANN ensemble. These results are illustrated in Figures A2a-A2i, Figures A3a-A3b, and Tables A2-A4 (Section A1.1 in Appendix).

In summary, the increase of λ leads to a more robust optimal solution and it is also evidenced in Figure 2.7. According to Figure 2.7, the higher the λ is, the higher constraint satisfaction probability can be achieved which means that the more robust solution can be obtained. Finally, the MDN is a reliable approximation approach that can precisely model the output variations of the ReLU ANN ensemble and original problem formulation based on different optimal decision variables with different λ.

The impact of changing ε:

- **Impact of varying $1 - \varepsilon$ on $\mu_{\hat{f}}$, $\mu_{\hat{g}_1}$, $\mu_{\hat{g}_2}$, $f(x, y)$, $g_1(x, y)$, and $g_2(x, y)$:** $1 - \varepsilon$

directly determines the constraint satisfaction probability. It is expected that the higher the $1-\varepsilon$ is, the more robust optimal solution with higher constraint satisfaction probability can be achieved. The direct influence of $1-\varepsilon$ on the optimal solution robustness and constraint satisfaction can be evidenced according to Figure 2.7 that constraint satisfaction probability increases obviously with $1-\varepsilon$. As can be seen from Tables A2 and A3 (Section A1.1 in Appendix), $\mu_{\hat{f}}$ obtained from both MDN and ReLU ANN ensemble increase with $1-\varepsilon$. Also, according to Table A4 (Section A1.1 in Appendix), $f(x,y)$ computed from the original problem formulation based on a certain set of optimal decision variables increases as $1-\varepsilon$ grows. The reason for the above-mentioned changes of $\mu_{\hat{f}}$ and $f(x,y)$ is that the constraint satisfaction probability rises as $1-\varepsilon$ grows. Then, the optimal solution becomes more robust and it increases as well as deviating from the real optimal solution. Meanwhile, $\mu_{\hat{g}_1}$, $\mu_{\hat{g}_2}$, $g_1(x,y)$, and $g_2(x,y)$ computed from the MDN, ReLU ANN ensemble and original problem formulation decrease with incremental $1-\varepsilon$ to move away from the constrained limits due to the increase of $\mu_{\hat{f}}$ and $f(x,y)$ with $1-\varepsilon$.

- **Impact of varying $1-\varepsilon$ on $\sigma_{\hat{f}}$, $\sigma_{\hat{g}_1}$, and $\sigma_{\hat{g}_2}$:** $\sigma_{\hat{f}}$ computed from the MDN decreases slightly with incremental $1-\varepsilon$ that matches the changes of $\sigma_{\hat{f}}$ evaluated from the ReLU ANN ensemble with increasing $1-\varepsilon$. On the other hand, according to Table A2 (Section A1.1 in Appendix), $\sigma_{\hat{g}_1}$ and $\sigma_{\hat{g}_2}$ computed from the MDN do not change significantly while varying $1-\varepsilon$. Moreover, as can be seen from Table A3 (Section A1.1 in Appendix), while λ equals to 1 or 10, $\sigma_{\hat{g}_1}$ and $\sigma_{\hat{g}_2}$ obtained from the ReLU ANN ensemble decrease with incremental $1-\varepsilon$, but they do not change significantly or instead increase slightly with growing $1-\varepsilon$ while λ equals to 50 or 100. According to the above observations, $1-\varepsilon$ has no significant and consistent impact on $\sigma_{\hat{g}_1}$ and $\sigma_{\hat{g}_2}$.

In summary, more robust optimal solution with higher constraint satisfaction probability is achieved by increasing $1-\varepsilon$. In addition, as can be seen from Figures A2a-A3f and Tables A3-A4 (Section A1.1 in Appendix), the MDN is a valid method that can generally reflect the output variations of the ReLU ANN ensemble and original problem formulation based on different optimal decision variables with different $1-\varepsilon$.

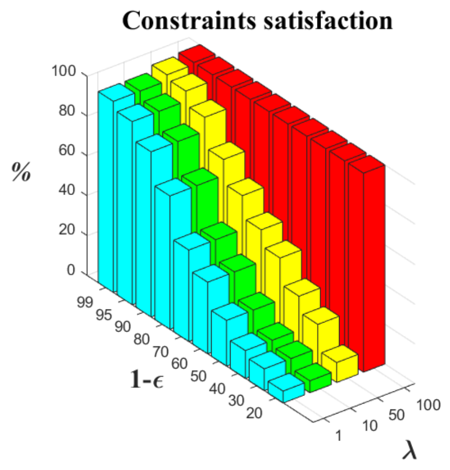


Figure 2.7: Percentages of the predictions from the ReLU ANN ensemble satisfying both constraints in (2.10b) and (2.10c) under different $1 - \varepsilon$ and λ

2.3.2 Comparison with the ensemble-based optimization method

The ensemble-based method [6] is another potential approach to cope with the prediction uncertainty in the illustrating example. The obtained ReLU ANN ensemble in Section 2.3 is directly used for the ensemble-based approach in this section. It is impractical to explicitly express the whole ReLU ANN ensemble into a single problem since it would lead to a very large-scale problem. Therefore, the ReLU ANN ensemble is incorporated into a black box optimization model with the same objective and constraint functions shown in (A1a)-(A1c) (Section A1.1 in Appendix). The black box optimization is solved utilizing the genetic algorithm in MATLAB. $1 - \varepsilon$ and λ are set to be 0.99 and 50 in this section, respectively. The results from both ensemble-based method and proposed MDN-based approach are shown and compared in Table 2.1.

Table 2.1: Results from both ensemble-based method and proposed MDN-based approach

	Optimal decision variables		Predicted means and standard deviations			Validation result and constraint satisfaction probability		Solution time (s)
	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{f}}+\lambda\sigma_{\hat{f}}$	$f(x,y)$	%	
Ensemble-based method	-5.000	4.736	0.395	0.010	0.895	0.395	100	16406
MDN-based approach	-4.904	4.992	0.424	0.012	1.024	0.424	100	27

* λ and $1-\varepsilon$ are 50 and 0.99, respectively.

The $f(x, y)$ values and constraint satisfaction probabilities in Table 2.1 are obtained by plugging corresponding optimal decision variables into the original problem formulation and the ReLU ANN ensemble, respectively. More related details are presented in Section 2.3. According to Table 2.1, the ensemble-based method can attain a solution with lower objective value ($\mu_{\hat{f}} + \lambda\sigma_{\hat{f}}$) and $f(x, y)$ than the MDN-based approach since the MDN-based approach employs the approximation of the ReLU ANN ensemble that might engender prediction errors. However, the ensemble-based method is overwhelmingly more time-consuming than the MDN-based approach.

Based on the above discussion, the MDN-based approach is more competitive than the ensemble-based method because the MDN-based approach can be directly modeled as a MILP problem of moderate size. Furthermore, the solution from the MDN-based approach is still satisfactory and robust, as can be seen from Table 2.1.

2.3.3 Necessity of considering the prediction uncertainty

Since it is generally hard to know whether the collected data is sufficient enough to train an accurate ANN surrogate model, the ANN prediction uncertainty engendered by the training data uncertainty should be considered in an ANN-based optimization, in order to improve the robustness of the optimal solution. In this section, the illustrating example is solved utilizing both ReLU ANN-based method mentioned in Section 2.2.2 (the method without considering the prediction uncertainty) and the proposed MDN-based approach, to demonstrate the importance of considering the prediction uncertainty.

In order to show the impact of the prediction uncertainty on the solution robustness, ReLU ANNs are trained based on smaller data sets than those used in Section 2.3. 30

different 2304-sample data sets are randomly selected from the data set used in Section 2.3, by using bootstrap sampling. First, 10 ReLU ANNs are trained individually on the first 10 data sets. Then, 10 ReLU ANN-based optimizations individually involving 10 different ReLU ANNs are gained via the procedure mentioned in Section 2.2.2. These ReLU ANN-based optimizations are solved utilizing CPLEX in GAMS. Afterwards, 10 MDNs are trained individually based on another 10 data sets. For each MDN, 100 ReLU ANNs are separately trained on different 1843-sample training sets resampled from the corresponding 2304-sample data set utilizing bootstrap sampling. The obtained 100-ReLU ANN ensemble is further approximated by the MDN. Then, 10 MDN-based optimizations individually incorporating 10 different MDNs can be generated, according to the procedure mentioned in Section 2.3. These 10 MDN-based optimizations are solved with $\lambda = 50$ and $1 - \varepsilon = 0.99$. Finally, another 10 MDNs are produced separately based on the remaining data sets according to the same procedure mentioned above. Meanwhile, 10 MDN-based optimizations involving these MDNs are solved with $\lambda = 10$ and $1 - \varepsilon = 0.90$. The solutions gained from the ReLU ANN-based optimizations and the MDN-based optimizations are shown in Figure 2.8.

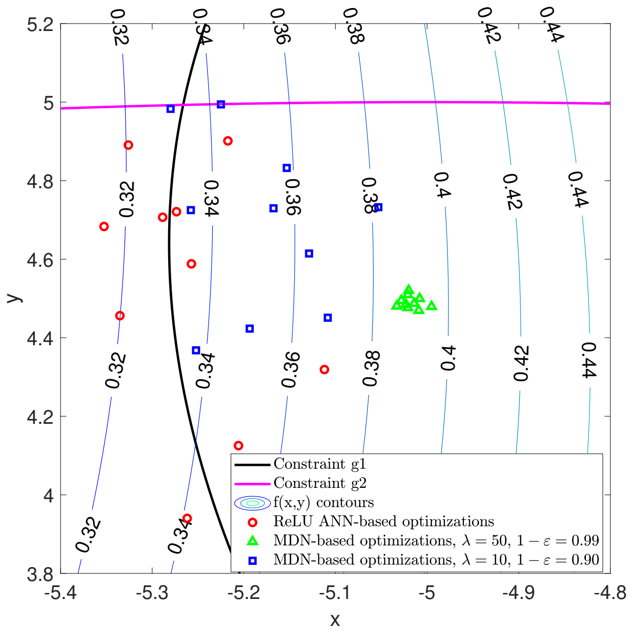


Figure 2.8: Solutions from the ReLU ANN-based optimizations and the MDN-based optimizations

As seen from Figure 2.8, the feasibility of the solutions from the single ReLU-ANN based methods is not guaranteed because of the prediction uncertainty sourced from the training set variation. On the other hand, most of the solutions from the MDN-based optimizations stay in the feasible region with lower variability since the prediction uncertainty is taken into account in this optimization approach. Moreover, by increasing λ and $1 - \varepsilon$ of the MDN-based approach, the solution robustness can be improved and the solution variability can be reduced. Accordingly, the proposed MDN-based optimization approach is more robust to the prediction uncertainty than the ReLU ANN-based optimization, even if the training data is limited. Also, the quality and variability of the solution from the MDN-based method can be adjusted via tuning λ and $1 - \varepsilon$.

2.4 Case studies

The presented ReLU MDN-based optimization approach is applied to two case studies to examine its performance on process optimization applications. The first case study is to minimize the total annual cost (TAC) of a water-ethanol distillation column with two specification constraints. The second one is the product yield maximization of a more complex ethylene glycol production process.

Impacts of varying $1 - \varepsilon$ and λ on the optimal solution robustness are also analysed in the first case study to reinforce the previous statement that optimal solution robustness can be controlled by $1 - \varepsilon$ and λ. For the second case study, the proposed optimization approach is conducted with $1 - \varepsilon = 0.99$ and different λ to demonstrate its ability to reach highly robust optimal solution of a complex non-linear process.

2.4.1 Distillation Process Optimization

Minimizing the TAC of a water-ethanol distillation column is carried out in this case study, The optimization formulation is given as:

$$\min \quad TAC \tag{2.11a}$$

$$\text{s.t.} \quad x_D \geq 0.8 \tag{2.11b}$$

$$Rec \geq 0.9 \tag{2.11c}$$

$$NT - NF \geq 1 \tag{2.11d}$$

$$1 \leq RR \leq 10 \tag{2.11e}$$

$$1 \leq RD \leq 10 \tag{2.11f}$$

$$10 \leq NT \leq 150 \tag{2.11g}$$

$$2 \leq NF \leq 149 \tag{2.11h}$$

RR, RD, NT, and NF are the reflux ratio, reboiler duty, total number of stages, and feed stage number of the water-ethanol distillation column, respectively, which are necessary design variables of a distillation column. TAC, x_D, and Rec in the above formulation all depend on RR, RD, NT, and NF. The unit of TAC in this work is 10^6\$. The unit of RD is GJ/hr. x_D and Rec are ethanol mole fraction and ethanol recovery at the top of the distillation column, respectively. The stage number 1 and stage number NT are stage numbers for the condenser and reboiler, respectively. As a result, the feed stage number NF must be in the range of 2 to $NT - 1$ and it is constrained by (2.11d) and (2.11h). Equations (2.11e)-(2.11h) are bounds for decision variables.

The distillation process in this case study is simulated using Aspen Plus and the flowsheet of this distillation process is shown in Figure 2.9. UNIQ-HOC thermodynamic model [99] is used for the simulation. In terms of the studied distillation column, the reboiler type, condenser type, and stage pressure drop are set to be Kettle, total condenser, and 0.01 psi, respectively. The feed temperature, feed pressure, feed flowrate, and feed composition are set to be 25°C, 1.5 atm, 100 kmol/hr, and 10 mol% ethanol as well as 90 mol% water, respectively. The optimal solution of the problem is obtained from the build-in optimizer in Aspen Plus: $TAC = 0.1492$ with $RR = 2.355$, $RD = 2.071$, $NT = 14$, $NF = 9$, $x_D = 0.8001$, and $Rec = 0.9002$.

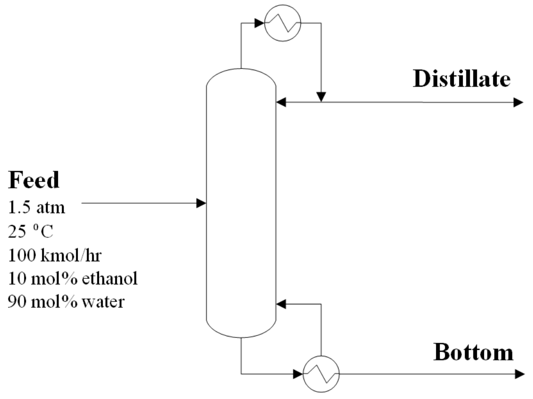


Figure 2.9: Flowsheet of the studied water-ethanol distillation process

The procedure for applying the proposed ReLU MDN-based optimization approach to this case study is the same as the procedure demonstrated in the illustrating example. First of all, the ReLU ANN is used as the surrogate model to approximate the non-linearities (TAC, x_D, and Rec) in the studied optimization formulation. To further enforce the ReLU ANN prediction robustness with the consideration of the prediction uncertainty originating from the training set variation, an ensemble of ReLU ANNs should be established to simulate the prediction uncertainty. First, 120000 samples are gathered from the simulation of the studied distillation process based on the following know-how: x_D and Rec are critical controlled variables which influence the optimal solution robustness significantly. Since x_D and Rec are more sensitive to RR and RD than to NT and NF, according to the simulation results from the original process model, the employed 120000-sample data set is gathered based on the following rules: 1) RR is collected in the range of 1 to 10 with an interval of 0.2; 2) RD is collected in the range of 1 to 10 with an interval of 0.2; 3) NT is collected in the range of 10 to 150 with an interval of 10; 4) For each selected NT in the data set, 3 or 4 stages are selected uniformly in the range of 2 to $NT - 1$ (The stage 1 and stage NT are the condenser and reboiler, respectively) as corresponding NFs. For instance, if $NT = 100$, stages 2, 25, 75, and 99 are selected as corresponding NFs for the data set.

Afterwards, 100 different ReLU ANNs are trained individually with 100 different training sets resampled (by using the bootstrap method) from the 120000-sample data set. Then, the ReLU ANN ensemble including 100 ReLU ANNs for simulating the prediction uncertainty is obtained. The procedure for producing the ReLU ANN ensemble is illustrated in Figure 2.10. Note that the feasibility of assuming the prediction uncertainty as Gaussian distribution in this case study is corroborated in Figures A4a-A4f (Section A1.2 in Appendix).

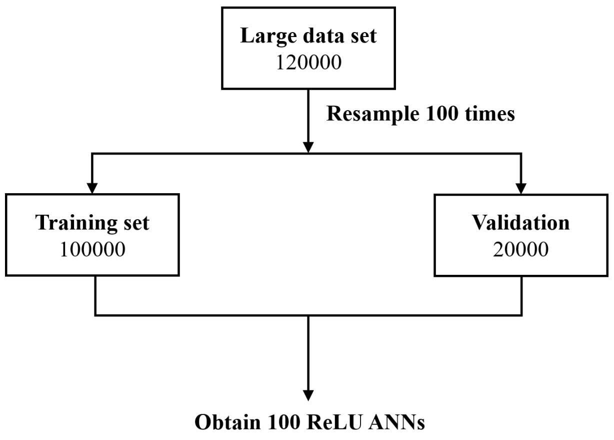


Figure 2.10: Schematic diagram for illustrating the production of the ReLU ANN ensemble in the first case study

Subsequently, after feeding 120000 inputs (one input includes one set of RR, RD, NT, and NF) into the ReLU ANN ensemble, $120000 \times 100 \times 3$ predictions can be obtained due to 120000 observations, 100 ReLU ANNs, and 3 outputs of each ReLU ANN (ReLU ANN predictions of TAC, x_D, and Rec which are $T\hat{A}C$, $\hat{x}_D$, and $\hat{Rec}$, respectively). Then, 120000×3 sets of prediction means and standard deviations can be computed and paired with corresponding observations to produce the training and validation sets for the MDN. Note that both ReLU ANN and MDN in this case study have 4, 50, and 30 neurons in the input layer, first hidden layer, and second hidden layer, respectively. There are 3 and 6 neurons in the output layers of the ReLU ANN and MDN, respectively. The MAPEs of the trained MDN based on the validation set are shown in Table A5 (Section A1.2 in Appendix).

After training the MDN with the training set generated from the ReLU ANN ensemble, the MDN is incorporated into the studied optimization as well as transforming the original optimization into the chance-constrained form. To be more specific, the constraints in (2.11b) and (2.11c) are transformed into chance constraints as well as the objective function in (2.11a) being reformulated to the mean-variance type. The reformulated chance-constrained optimization is given in (A2a)-(A2u) (Section A1.2 in Appendix).

Finally, the chance-constrained optimization is formulated as a MILP problem that can be solved using CPLEX in GAMS. It is solved several times with different $1 - \varepsilon$ and λ to corroborate that the optimal solution robustness can be adjusted via changing these parameters. The obtained optimal RR, RD, NT, NF from the proposed method are in ranges of $2.255 - 3.940$, $2.145 - 2.728$, $16 - 28$, and $14 - 21$, respectively, based on different λ

and $1-\varepsilon$. With respect to different sets of optimal decisions based on different $1-\varepsilon$ and λ, the corresponding predictions of the embedded MDN are listed in Table A6 (Section A1.2 in Appendix). Moreover, these optimal decisions are also input into the ReLU ANN ensemble and the original full-order model of the distillation process simulated using Aspen Plus. The corresponding results are listed in Table A7 and A8 (Section A1.2 in Appendix).

2.4.1.1 Discussion

Based on the results got from the first case study, the impacts of changing $1-\varepsilon$ and λ on the optimal results are analysed in details in this section to reinforce the previous assertion that the optimal solution robustness is controlled by $1-\varepsilon$ and λ. The discussion is divided into several parts listed below and associated detailed computation results are shown in Tables A6-A8 (Section A1.2 in Appendix).

- **Impact of changing λ on $\mu_{T\hat{A}C}$, $\sigma_{T\hat{A}C}$, and TAC:** The larger the λ is, the more robust optimal solution can be achieved. $\mu_{T\hat{A}C}$ and $\sigma_{T\hat{A}C}$ computed from the MDN grows and declines with incremental λ, respectively, because the weight of $\sigma_{T\hat{A}C}$ in (A2a) (Section A1.2 in Appendix) grows and the minimization focuses more on $\sigma_{T\hat{A}C}$. Also, after inputting different sets of optimal decision variables to the ReLU ANN ensemble and the original distillation process model, changes of $\mu_{T\hat{A}C}$ and $\sigma_{T\hat{A}C}$ gained from the ReLU ANN ensemble with increasing λ are similar to the results computed from the MDN. Meanwhile, the variation of TAC calculated from the original process model with growing λ is similar to the changes of $\mu_{T\hat{A}C}$ gained from both MDN and ReLU ANN ensemble. The above-mentioned changes of $\mu_{T\hat{A}C}$, $\sigma_{T\hat{A}C}$, and TAC indicate that the optimal solution increases and deviates from the real optimum with increasing λ. In the meantime, the optimal solution variability is reduced with incremental λ to enhance the solution robustness.

- **Impact of changing λ on $\mu_{\hat{x}_D}$, $\sigma_{\hat{x}_D}$, and x_D:** $\mu_{\hat{x}_D}$ computed from the MDN has no significant variation with increasing λ, but $\mu_{\hat{x}_D}$ gained from the ReLU ANN ensemble as well as x_D calculated from the original process model increase slightly with λ. Although the variations of $\mu_{\hat{x}_D}$ with λ computed from the MDN and ReLU ANN ensemble mismatch slightly, changes of $\sigma_{\hat{x}_D}$ with λ obtained from both MDN and ReLU ANN ensemble are similar. While λ increasing, $\sigma_{\hat{x}_D}$ computed from both MDN and ReLU ANN ensemble decline to reduce the variability of $\hat{x}_D$.

- **Impact of changing λ on the solution robustness:** The optimal solution robustness can be improved by increasing λ due to the above-mentioned reasons. However, as can be seen from Figure 2.13, overly conservative solutions are gained while $\lambda = 100$. In order to avoid overly conservative solution, the appropriate λ value should be chosen to make $\sigma_{T\hat{A}C}$ have a similar size to $\mu_{T\hat{A}C}$.

- **Impact of varying $1 - \varepsilon$ on $\mu_{T\hat{A}C}$, $\mu_{\hat{x}_D}$, $\mu_{\hat{Rec}}$, TAC, and x_D:** $1 - \varepsilon$ significantly influences the robustness and constraint satisfaction probability of the optimal solution. $\mu_{T\hat{A}C}$, $\mu_{\hat{x}_D}$, and $\mu_{\hat{Rec}}$ computed from the MDN grow with $1 - \varepsilon$. The reason is that the higher the $1 - \varepsilon$ is, the more robust optimal solution with higher constraint satisfaction probability can be obtained from the MILP chance-constrained optimization. Furthermore, growing constraint satisfaction percentage leads to increasing $\mu_{\hat{x}_D}$ and $\mu_{\hat{Rec}}$ computed from the MDN to move away from the lower limits (which are 0.8 and 0.9, respectively). Also, note that the TAC of a distillation column increases as the higher purity and recovery of the product (which are x_D and Rec) are attained. Therefore, $\mu_{T\hat{A}C}$ increases with both $\mu_{\hat{x}_D}$ and $\mu_{\hat{Rec}}$ based on the computation of the MDN. Additionally, variations of $\mu_{T\hat{A}C}$ and $\mu_{\hat{x}_D}$ acquired from the ReLU ANN ensemble with incremental $1 - \varepsilon$ are similar to the results gained from the MDN. Also, changes of TAC and x_D calculated from the original process model with increasing $1 - \varepsilon$ is similar to the variations of $\mu_{T\hat{A}C}$ and $\mu_{\hat{x}_D}$ obtained from both MDN and ReLU ANN ensemble with growing $1 - \varepsilon$.

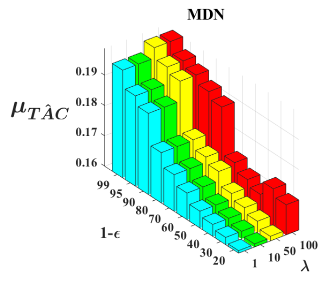
- **Impact of changing $1 - \varepsilon$ on $\sigma_{T\hat{A}C}$ and $\sigma_{\hat{x}_D}$:** $\sigma_{T\hat{A}C}$ and $\sigma_{\hat{x}_D}$ computed from the embedded MDN generally decrease with incremental $1 - \varepsilon$ to reduce variabilities of $T\hat{A}C$ and $\hat{x}_D$. In the meantime, changes of $\sigma_{T\hat{A}C}$ and $\sigma_{\hat{x}_D}$ obtained from the ReLU ANN ensemble is similar to the computation from the MDN.

- **Impact of varying $1 - \varepsilon$ on $\mu_{\hat{Rec}}$, $\sigma_{\hat{Rec}}$, and Rec:** The computation results of $\mu_{\hat{Rec}}$ and $\sigma_{\hat{Rec}}$ from the MDN and ReLU ANN ensemble are mismatched. In particular, variations of $\mu_{\hat{Rec}}$ and $\sigma_{\hat{Rec}}$ computed from the MDN with incremental $1 - \varepsilon$ are opposite to the results obtained from the ReLU ANN ensemble. Nevertheless, as can be seen from Figure 2.13, the constraint satisfaction probability still increases obviously with incremental λ and $1 - \varepsilon$ regardless of the mismatches of $\mu_{\hat{Rec}}$ as well as $\sigma_{\hat{Rec}}$ acquired from the MDN and ReLU ANN ensemble. The reason is that the MDN always underestimates $\mu_{\hat{Rec}}$ based on different λ and $1 - \varepsilon$. Therefore, $\mu_{\hat{Rec}}$ and Rec respectively gained from the ReLU ANN ensemble and the original process model based on different

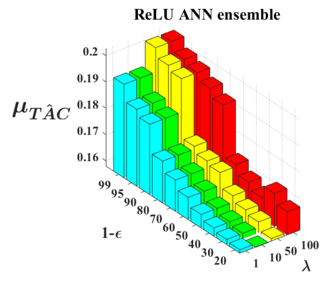
$1-\varepsilon$ and λ are always higher and stay farther away from the lower limit (which is 0.9) than the $\mu_{\hat{Rec}}$ from the MDN.

- **Impacts of changing λ and $1-\varepsilon$ on decision variables:** The obtained optimal RR, RD, NT, NF from the proposed method are in ranges of $2.255-3.940$, $2.145-2.728$, $16-28$, and $14-21$, respectively, based on different λ and $1-\varepsilon$. After plugging these optimal decisions into the original full-order model, corresponding x_D and TAC computed from the original model are in ranges of $0.8200-0.8430$ and $0.1581-0.2031$, respectively. These results are illustrated in Figures 2.11c and 2.11f. While λ is fixed, RR, RD rise as $1-\varepsilon$ grows that leads to the increase of x_D. Since incremental RR and RD give rise to the higher concentration of ethanol at the top and higher steam and cooling water costs, TAC grows accordingly. On the other hand, while $1-\varepsilon$ is fixed, RR, RD, NT increase with increasing λ that makes the growing x_D. Because increasing RR, RD, and NT lead to higher x_D, utility cost (steam and cooling water costs), and capital cost, TAC increases correspondingly. The above-mentioned results are consistent with the previously mentioned fact that higher λ and $1-\varepsilon$ lead to a more robust and conservative solution (also worse objective value) with the larger constraint satisfaction probability.

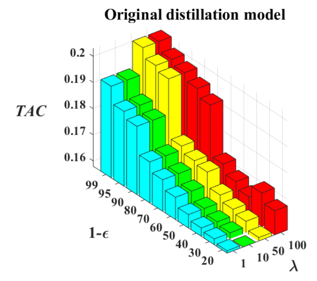
In summary, increasing $1-\varepsilon$ and λ leads to the decrease of $\sigma_{\hat{x}_D}$ and reduces the variability of $\hat{x}_D$. Moreover, increasing $1-\varepsilon$ leads to the increase of $\mu_{\hat{x}_D}$ and the solution move away from the lower limit. In addition, $\mu_{\hat{Rec}}$ computed from the MDN are all underestimated based on different $1-\varepsilon$ and λ that make $\mu_{\hat{Rec}}$ and Rec respectively gained from the ReLU ANN ensemble and the original process model far enough away from the lower limit. Therefore, the $\mu_{\hat{Rec}}$ and $\sigma_{\hat{Rec}}$ computed from the MDN are not influential to the optimal solution robustness. On the other hand, the decision variables RR, RD, and NT grow with increases of λ and $1-\varepsilon$ that leads to a more robust and conservative solution with higher x_D and TAC. Based on the aforementioned, the optimal solution robustness as well as constraint satisfaction probability in this case study can be enhanced with increasing $1-\varepsilon$ and λ. However, a too large value of λ should be avoided to prevent the overly conservative solution.



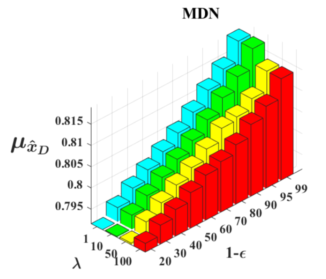
(a) $\mu_{T\hat{A}C}$ computed from the MDN



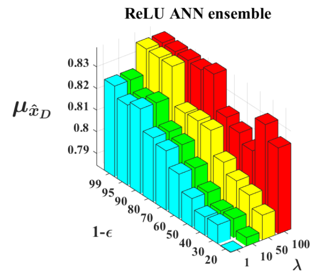
(b) $\mu_{T\hat{A}C}$ computed from the ReLU ANN ensemble



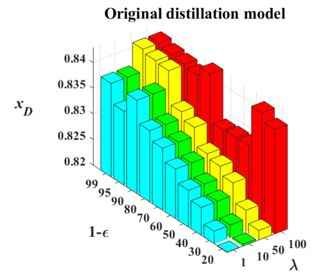
(c) TAC obtained from the original process model



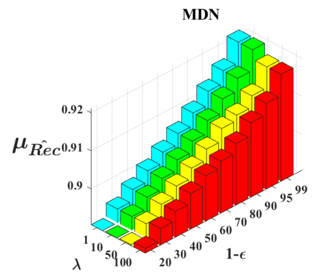
(d) $\mu_{\hat{x}_D}$ computed from the MDN



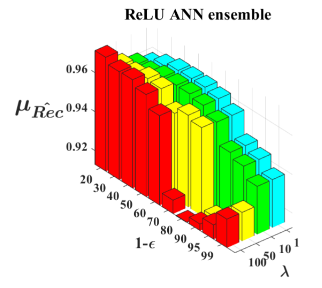
(e) $\mu_{\hat{x}_D}$ computed from the ReLU ANN ensemble



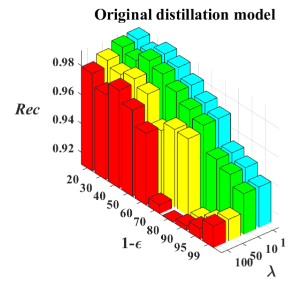
(f) x_D obtained from the original process model



(g) $\mu_{\hat{Rec}}$ computed from the MDN

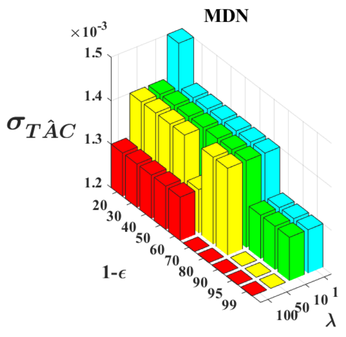


(h) $\mu_{\hat{Rec}}$ computed from the ReLU ANN ensemble

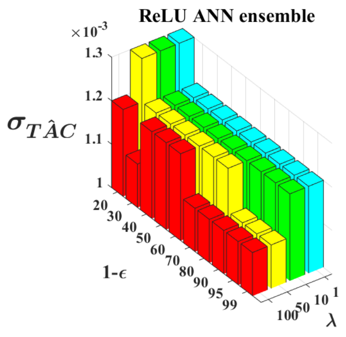


(i) Rec obtained from the original process model

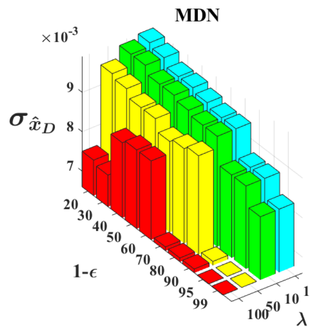
Figure 2.11: Prediction means computed from the MDN and ReLU ANN ensemble as well as results obtained from the original distillation process model with different $1 - \varepsilon$ and λ



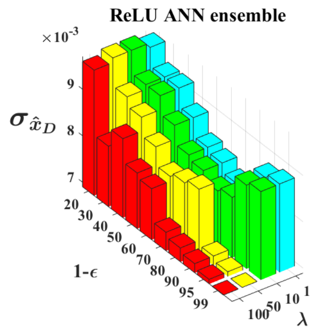
(a) $\sigma_{T\hat{A}C}$ computed from the MDN



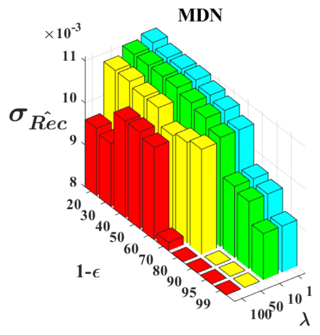
(b) $\sigma_{T\hat{A}C}$ computed from the ReLU ANN ensemble



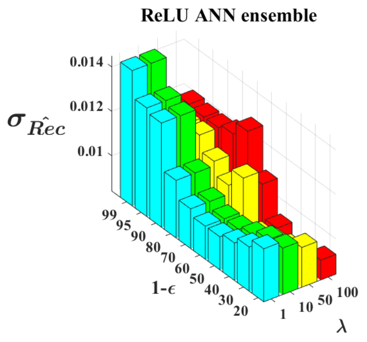
(c) $\sigma_{\hat{x}_D}$ computed from the MDN



(d) $\sigma_{\hat{x}_D}$ computed from the ReLU ANN ensemble



(e) $\sigma_{\hat{Rec}}$ computed from the MDN



(f) $\sigma_{\hat{Rec}}$ computed from the ReLU ANN ensemble

Figure 2.12: Prediction standard deviations computed from the MDN and ReLU ANN ensemble with different $1-\varepsilon$ and λ

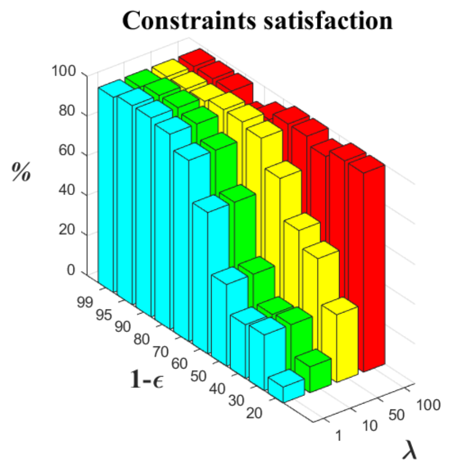


Figure 2.13: Percentages of the predictions from the ReLU ANN ensemble satisfying both constraints in (2.11b) and (2.11c) under different $1 - \varepsilon$ and λ

2.4.2 Optimization of the ethylene glycol production process

In the second case study, the presented optimization approach is applied to a more complicated optimization problem, in order to show its ability to solve complicated process optimization problems.

The optimization problem in this section is to maximize the product yield of the ethylene glycol (EG) production process. In terms of the studied EG production process which is shown in Figure 2.14, ethylene oxide (EO) and water are first fed into the CSTR to produce EG. The reaction equations of synthesizing EG and associated by-products are given as:

$$EO + H_2O \xrightarrow{R_1} EG \tag{2.12a}$$

$$EG + EO \xrightarrow{R_2} DEG \tag{2.12b}$$

$$DEG + EO \xrightarrow{R_3} TEG \tag{2.12c}$$

DEG and TEG are diethylene glycol and triethylene glycol, respectively, which are undesirable by-products. The kinetic models of the above reaction equations are shown below:

$$R_1 = k_1 c_{EO} c_{H_2O} \tag{2.13a}$$
$$R_2 = k_2 c_{EO} c_{EG} \tag{2.13b}$$
$$R_3 = k_3 c_{EO} c_{DEG} \tag{2.13c}$$

with the kinetic parameters $k_1 = 1.8302 \times 10^{10} exp(-\frac{87987}{R_g T})$, $k_2 = 2.1k_1$, and $k_3 = 2.2k_1$ [100]. R_g and T are the ideal gas constant and reaction temperature, respectively. c_{EO}, c_{H_2O}, c_{EG}, and c_{DEG} are molar concentrations of EO, water, EG, and DEG, respectively. The output of the CSTR is connected to the flash separator. The product (EG) is collected from the bottom of the flash separator. Meanwhile, 97% of the top flash vapour stream is recycled to mix with the feed flow. The entire EG production process is simulated using Aspen Plus as well as assuming the vapour-liquid equilibrium as ideal phase behaviour.

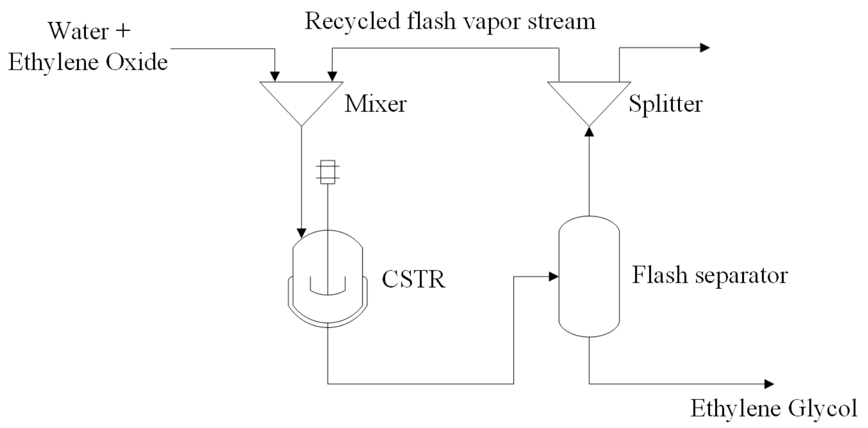


Figure 2.14: Flowsheet of the studied ethylene glycol production process

The objective is to maximize the product yield (EG molar flowrate in the bottom of the flash separator, F_{EG}, with the unit of kmol/hr). The decision variables include the water feed flowrate (F_w, with the unit of kmol/hr), volume of the CSTR (V_{CSTR}, with the unit of m^3), and temperature of the flash separator (T_f, with the unit of K). The EO feed flowrate, the operating temperature of the CSTR, and the fraction of recycled flash vapour stream are fixed at 625 kmol/hr, 295 K, and 0.97, respectively. The corresponding optimization formulation is given as:

$$
\begin{aligned}
\max \quad & F_{EG} && \text{(2.14a)}\\
\text{s.t.} \quad & X_{EO} \geq 0.6 && \text{(2.14b)}\\
& x_{EG} \geq 0.5 && \text{(2.14c)}\\
& 700 \leq F_w \leq 5000 && \text{(2.14d)}\\
& 221 \leq V_{CSTR} \leq 321 && \text{(2.14e)}\\
& 300 \leq T_f \leq 390 && \text{(2.14f)}
\end{aligned}
$$

X_{EO} and x_{EG} are the ethylene oxide conversion rate and the EG mole fraction in the bottom flow of the flash separator, respectively. Equations (2.14d)-(2.14f) are bounds of the decision variables. Except for the constraints and bounds of the decision variables in the above optimization formulation, all the settings of the EG production process in this section are the same as in Kahrs' work [101].

In order to apply the proposed optimization approach to the studied EG production process optimization, an ensemble of ReLU ANNs should be trained to address non-linearities in the process including reaction kinetic models, vapour-liquid equilibrium, and mass balance with recycling flow. 82826 samples are collected from the full-order model simulated by using Aspen Plus. F_w, V_{CSTR}, and T_f are observations in each collected sample. F_{EG}, X_{EO}, and x_{EG} are targets in each sample. 100 ReLU ANNs are trained individually with 100 different training sets. Each training set includes 66261 samples resampled from the 82826-sample data set by using the bootstrap method. Afterwards, 82826 observations are input into the ReLU ANN ensemble to generate $82826 \times 100 \times 3$ predictions and then 82826×3 sets of prediction means and standard deviations are calculated based on the generated predictions. The calculated prediction means and standard deviations are paired with corresponding observations to form the training and validation sets for the MDN. One remark here is that structures of the MDN and ReLU ANNs in this section are the same as in Section 2.3 and Section 2.4.1. The MAPEs of the trained MDN based on the validation set are shown in Table A9 (Section A1.3 in Appendix). After training the MDN, the original optimization problem including (2.14a)-(2.14f) is reformulated to the MILP chance-constrained form with the mean-variance type objective function as well as incorporating the MDN. The reformulated optimization problem is given in (A3a)-(A3u) (Section A1.3 in Appendix).

The MILP chanced-constrained optimization is solved with $1 - \varepsilon = 0.99$ as well as different λ values, and obtained results are shown in Table 2.2. According to Table 2.2, the optimal solution with respect to $\lambda = 100$ is the most robustness with the highest constraint satisfaction probability. One remark here is that F_{EG} decreases with both incremental X_{EO} and x_{EG} according to the simulation results from the full-order model. Therefore, while λ equals to 100, $\mu_{\hat{F}_{EG}}$ is the lowest since $\mu_{\hat{X}_{EO}}$ and $\mu_{\hat{x}_{EG}}$ are the highest to stay away from lower limits of constraints (0.6 and 0.5) for the enhancement of the optimal solution robustness. On the other hand, $\mu_{\hat{F}_{EG}}$, $\mu_{\hat{X}_{EO}}$, and $\mu_{\hat{x}_{EG}}$ computed from the embedded MDN are accurate compared to values gained from the full-order model that are shown in parentheses in Table 2.2 based on the same optimal decision variables. The process optimization in this case study contains more non-linearities (e.g., reaction kinetic models, vapour-liquid equilibrium, and mass balance with recycling flow) than the illustrating example and the first case study, the satisfactory optimal solutions of this case study show that the proposed optimization approach is not only workable for simple optimization problems but also effective for more complicated process optimizations.

Finally, in order to show that the true optimum of the studied EG production process can be approached closely by utilizing the presented optimization approach, the optimal solution gained from the proposed optimization approach is compared to the true optimum mentioned in Kahrs' study [101] based on the same condition. Since there is no constraints for the optimization in Kahrs' work, the MILP chance-constrained optimization in this section is solved with the two chance constraints removed. Also, λ is set to be 0 to neglect the variability of $\hat{F}_{EG}$. The obtained results are shown and compared with the true optimum in Table 2.3. According to Table 2.3, the obtained optimal results from the presented optimization approach is close to the true optimum.

Table 2.2: EG production process optimization results obtained from the proposed optimization approach under different λ

	$\lambda = 1$	$\lambda = 10$	$\lambda = 50$	$\lambda = 100$
$\mu_{\hat{F}_{EG}}$ (F_{EG})	383.293 (388.107)	383.293 (388.107)	383.293 (388.107)	378.092 (379.594)
$\mu_{\hat{X}_{EO}}$ (X_{EO})	0.605 (0.611)	0.605 (0.611)	0.605 (0.611)	0.606 (0.613)
$\mu_{\hat{x}_{EG}}$ (x_{EG})	0.503 (0.502)	0.503 (0.502)	0.503 (0.502)	0.522 (0.517)
constraint satisfaction %	74%	74%	74%	100%
F_w	923.510	923.510	923.510	880.897
V_{CSTR}	321	321	321	321
T_f	364.682	364.682	364.682	366.596

Values in parentheses are full-order model computations based on the same operating conditions.

Above optimal results are obtained with $1 - \varepsilon = 0.99$

Table 2.3: Optimal results from the proposed optimization approach and Kahrs' work

	Optimal results	
	Proposed optimization approach	Kahrs' work [101]
$\mu_{\hat{F}_{EG}}$ / F_{EG}	532.7 (524.1)	527.5
F_w	4753	4864
V_{CSTR}	321	321
T_f	344.3	344.5

Value in the parenthesis is the full-order model computation based on the same operating conditions.

2.5 Conclusion

An innovative optimization method based on ReLU MDN was proposed to address the uncertainty of neural network surrogate model prediction caused by training set variation. In order to apply the proposed approach to the process optimization problem, a data set is first collected from the process studied. Then, an ensemble of ReLU ANNs are trained with different training sets re-sampled from the data set using the bootstrap method. The obtained ReLU ANN ensemble is further used to simulate the prediction uncertainty caused by training set variation. Numerous observations are then fed into the ReLU ANN ensemble to produce predictions. The means and standard deviations of predictions are paired with the corresponding input observations to form the training and validation sets for the MDN training. Subsequently, MDN is used to approximate the ReLU ANN ensemble and to address the prediction uncertainty. The MDN is incorporated into the optimization problem to deal with both constraint and objective uncertainties through a MILP formulation.

According to the results of this work, the proposed approach can effectively model the uncertainty of ANN model prediction. The outputs from the MDN well match the means and standard deviations of predictions from the ReLU ANN ensemble. This means that the MDN is a reliable approach to approximate the ReLU ANN ensemble for addressing the prediction uncertainty. To address the model prediction uncertainty, we can incorporate the MDN into the optimization problem through chance constraint and mean-variance objective. The results exhibited in the illustrating example demonstrate that the presented approach leads to improved solution robustness, even if the training data is limited. Also, the proposed approach is much more efficient than the ensemble-based approach from the literature [6]. Moreover, the results of the case study show that the solution robustness and variability can be effectively adjusted through parameters $1 - \varepsilon$ and λ in the presented optimization approach. However, the potential applications are not restricted to the proposed chance constraint and mean-variance modeling.

As to future work, the proposed optimization approach can be improved under an adaptive sampling framework. In addition, other modeling techniques for addressing uncertainty can be also studied. Finally, an interesting direction worth exploration is to extend the approach to neural networks with different types of activation functions.

Chapter 3

Machine Learning-Based Approaches for Joint Chance-Constrained Optimization

Abstract

This chapter presents two machine learning-based approaches for steady-state joint chance-constrained process optimization and dynamic joint chance-constrained stochastic optimal control. The first method is presented in Section 3.1, which involves using a neural network (NN) to approximate the joint chance constraint (JCC) and integrating the NN into the optimization model. This method enables the NP-hard joint chance-constrained optimization problems to become deterministically solvable and tractable. The effectiveness of this method is demonstrated in a nonlinear process optimization problem. The second approach is introduced in Section 3.2, which extends the first one by utilizing a recurrent neural network (RNN) to handle the JCC in a stochastic optimal control problem (SOCP). This approach significantly reduces computational burden compared to other common stochastic optimal control methods. We apply this approach to a numerical SOCP example and a case study to demonstrate its efficacy.

3.1 Joint Chance Constrained Process Optimization through Neural Network Approximation

3.1.1 Introduction

Practical process optimization often faces uncertainties. Chance-constrained optimization is a popular technique for addressing uncertainty [102]. It enforces that the optimal solution should satisfy the uncertain constraint with a certain probability level. There are two types of chance constraint: the individual chance constraint (ICC) and the joint chance constraint (JCC). The JCC is more general in engineering applications than the ICC since the JCC ensures all constraints to be satisfied simultaneously to a certain confidence level, which is more natural in many applications [103]. However, the JCC is generally difficult to solve as it requires dealing with multidimensional distributions. Thus, joint chance-constrained optimization problems (JCCPs) are generally solved through approximations. There are two main approximation methods: analytical approximation methods and sampling-based methods [28].

In terms of analytical approximation approaches, they are exploited to approximate a chance-constrained problem with a deterministic optimization formulation. Among all the analytical approximation methods, the robust optimization (RO) [104] has been extensively studied recently. The attractiveness of RO approximation is that it does not require the assumption of uncertainty distributions and uncertainties are described employing uncertainty sets. A comprehensive research on different types of uncertainty set is studied by Li [52]. In Chen's work [105], different tractable approximations to individual chance-constrained problems with the use of RO have been proposed, and they are further extended to joint chance-constrained problems. Hong [106] developed a RO approximation for joint chance-constrained problems while data is insufficient to infer the underlying uncertainty distributions. Although the RO approximation is a distribution-free and widely studied method for addressing joint chance-constrained problems, it may lead to an overly conservative solution because the worse-case scenario is taken into account in the RO [27, 104]. The overly conservative solution might satisfy uncertain constraints with the probability much higher than the required confidence level, and deviate significantly from the actual optimum.

The overly conservative solution can be avoided by adopting sampling-based methods to handle chance constraints, if empirical distributions of uncertainties can be obtained from historical data. 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 chance constraint in the problem is approximated by a set of constraints corresponding to each generated sample [53]. However, the randomness of sample generation might cause the infeasibility of the problem with the approximated chance constraint [28]. In order to overcome such the 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 confidence level in the original chance constraint. The SAA can approximate chance constraints tightly. Also, the solution may satisfies the chance constraint strictly with the use of SAA. Furthermore, the SAA is an easy-to-implement approach that can make intractable chance-constrained problems tractable [14, 107]. More details related to the SAA are demonstrated in Section 3.1.2.3.

Based on the concept of SAA, a novel and efficient ReLU artificial neural network (ANN) approximation method is developed in this work to deal with the joint chance constraint. The developed approach is a two-layer approximation method composed of the empirical quantile function (QF) approximation and the ReLU ANN approximation. More specifically, the joint chance constraint is first reformulated to the quantile-based form and then approximated by the empirical QF, due to the better convexity and less complexity of the QF compared to the probability function in the chance constraint. Subsequently, the ReLU ANN is introduced to further approximate the empirical QF because of its strong approximation ability for non-linear functions [108]. Moreover, the ReLU ANN can approximate the stochastic empirical QF as a deterministic MILP model [82]. Finally, the optimization incorporating the proposed ReLU ANN approximation model for addressing the joint chance constraint can be solved by the off-the-shelf solver, such as CPLEX in GAMS.

The proposed ReLU ANN approximation approach is presented in Section 3.1.2. The proposed approach is applied to a case study in Section 3.1.3 to show that the satisfactory optimal solution of a complex process involving uncertainty can be achieved. This case study is to maximize the product yield of the ethylene glycol production process under measurement uncertainty.

3.1.2 Methods

In this section, the joint chance-constrained optimization and its quantile function-based reformulation are illustrated. Then, an approximation approach for dealing with the joint chance constraint is investigated, which is a two-layer approach composed of the empirical quantile function (QF) approximation and the ReLU ANN approximation.

3.1.2.1 Joint chance-constrained optimization

The formulation of a joint chance-constrained optimization is given as:

$$\min_{x} \quad f(x) \tag{3.1a}$$

$$\text{s.t.} \quad Pr(g_i(x,\xi) \leq 0 \quad (i = 1, ..., w)) \geq 1 - \varepsilon \tag{3.1b}$$

where x represents the decision variable with a closed feasible domain $\mathcal{X} \subset \mathbb{R}^{n_x}$. $f : \mathbb{R}^{n_x} \longrightarrow \mathbb{R}$ is the objective function. ξ symbolizes an uncertain parameter vector with the sample space $\Xi \subset \mathbb{R}^{n_\xi}$. The probability measure Pr is well defined on the sigma algebra of Ξ ($\mathcal{B}_\Xi$). The probability density of ξ is indicated as $p_\xi(\xi)$. $g_{i=1,...,w} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_\xi} \rightarrow \mathbb{R}^w$ is a finite set of functions. $1 - \varepsilon$ is the confidence level, and ε denotes the violation probability. The joint chance constraint in (3.1b) ensures that all constraints $g_{i=1,...,w}(x,\xi) \leq 0$ are satisfied simultaneously to a certain confidence level $1-\varepsilon$. f and $g_{i=1,...,w}$ are assumed to be continuous and differentiable with $\forall x \in \mathcal{X}$ and $\forall \xi \in \Xi$. Thus, the continuous multivariate CDF based on the random variables $g_{i=1,...,w}(x,\xi)$ exists.

It is NP-hard to solve the above joint chance-constrained optimization due to the following reasons:

- A joint chance-constrained problem is convex only when the distributions of involved random variables are log-concave [109].
- Usually, joint chance-constrained problems are computationally intractable, even though uncertainty distributions are known [110, 111].

Due to the above-mentioned difficulties, joint chance-constrained optimizations are generally solved through approximations [25, 28].

3.1.2.2 Quantile reformulation of joint chance constraint

$Pr(\omega \leq \gamma), \gamma \in \mathbb{R}$ with the random variable ω can be expressed equivalently in the cumulative distribution function (CDF) form $\phi_\omega(\gamma)$. Furthermore, the $1-\varepsilon$ level quantile of the random variable ω is defined as:

$$Q^{1-\varepsilon}(\omega) = \inf \{\gamma \in \mathbb{R} \mid Pr(\omega \leq \gamma) \geq 1 - \varepsilon\} \tag{3.2}$$

According to the above discussion, the following relationship holds:

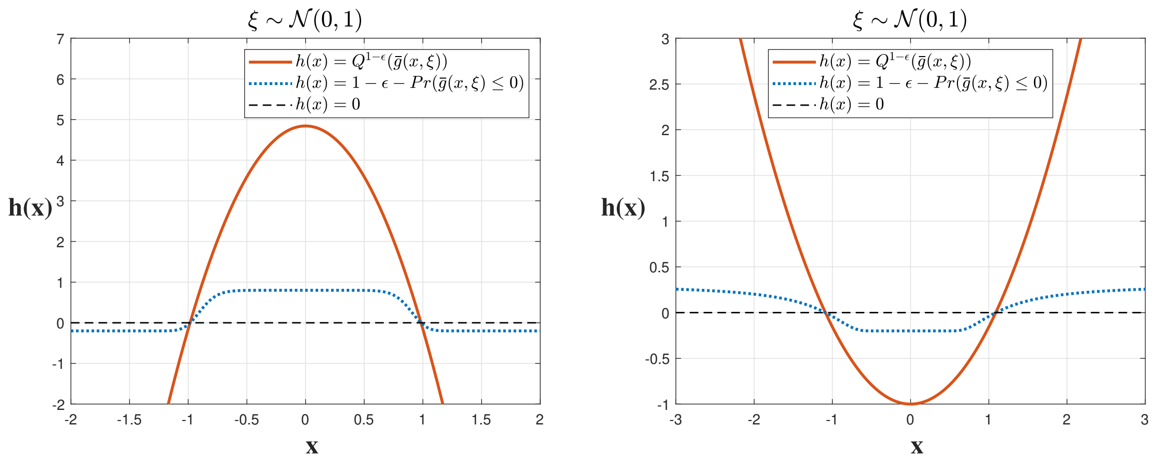
$$Pr(g(x,\xi) \leq \gamma) \geq 1 - \varepsilon \quad \leftrightarrow \quad \phi_g(\gamma, x) \geq 1 - \varepsilon \quad \leftrightarrow \quad Q^{1-\varepsilon}(g(x,\xi)) \leq \gamma \tag{3.3}$$

Because of the above relationship, $Q^{1-\varepsilon}(g(x,\xi)) \leq 0$ is equivalent to $Pr(g(x,\xi) \leq 0) \geq 1-\varepsilon$ and $\phi_g(0,x) \geq 1 - \varepsilon$. Note that $g(x,\xi)$ should be a scalar random variable. Otherwise, $Q^{1-\varepsilon}(g(x,\xi)) \leq 0$ cannot be well defined. For $g(x,\xi)$ being a vector, $\bar{g}(x,\xi) = \max_{i=1,\ldots,w} g_i(x,\xi)$ is used instead.

Accordingly, the original optimization formulation including (3.1a) and (3.1b) can be reformulated equivalently as follows:

$$\begin{aligned} \min_x \quad & f(x) && (3.4a) \\ \text{s.t.} \quad & Q^{1-\varepsilon}(\bar{g}(x,\xi)) \leq 0 && (3.4b) \end{aligned}$$

The benefit of adopting the above QF-based problem reformulation is explained below: Let's first consider two types of joint chance constraint reformulations derived from (3.1b), i.e., $Q^{1-\varepsilon}(\bar{g}(x,\xi)) \leq 0$ and $1-\varepsilon-Pr(\bar{g}(x,\xi) \leq 0) \leq 0$. Since $Q^{1-\varepsilon}(\bar{g}(x,\xi))$ and $1-\varepsilon-Pr(\bar{g}(x,\xi) \leq 0)$ are essentially functions of x, they can be approximated by surrogate models depending on x, for enhancing computation tractability. As can be seen from Figure 3.1, it is much easier to model $Q^{1-\varepsilon}(\bar{g}(x,\xi))$ than $1-\varepsilon-Pr(\bar{g}(x,\xi) \leq 0)$, due to better convexity and less complexity of $Q^{1-\varepsilon}(\bar{g}(x,\xi))$.



(a) For $\bar{g}(x,\xi) = -5x^2 + 4 + \xi$ (b) For $\bar{g}(x,\xi) = \xi x^2 - 1$

Figure 3.1: Comparisons between $1-\varepsilon - Pr(\bar{g}(x,\xi) \leq 0)$ and $Q^{1-\varepsilon}(\bar{g}(x,\xi))$ with $1-\varepsilon = 0.8$

3.1.2.3 ReLU ANN-based approximation of QF

Since the analytical solution of $Q^{1-\varepsilon}(\bar{g}(x,\xi))$ is intractable, $Q^{1-\varepsilon}(\bar{g}(x,\xi))$ should be further approximated and calculated through the empirical method. The sample set $\Xi^N = \{\xi_1, ..., \xi_N\}$ should be gained first by extracting samples independently from Ξ. Then, $\bar{G}^N = \{\bar{g}(x,\xi_1), ..., \bar{g}(x,\xi_N)\}$ can be obtained based on a given x. Subsequently, the CDF $\phi_{\bar{g}}(\gamma, x)$ can be approximated by the empirical CDF shown below:

$$\widetilde{\phi_{\bar{g}}}(\gamma, x) = \frac{1}{N}\sum_{j=1}^{N} \mathbb{I}(\bar{g}(x,\xi_j) \leq \gamma) \tag{3.5}$$

where N is the number of samples, and $\mathbb{I}(\bar{g}(x,\xi_j) \leq \gamma)$ denotes the indicator function written as:

$$\mathbb{I}(\bar{g}(x,\xi_j) \leq \gamma) = \begin{cases} 0, & for \quad \bar{g}(x,\xi_j) > \gamma \\ 1, & for \quad \bar{g}(x,\xi_j) \leq \gamma \end{cases} \tag{3.6}$$

Note that using the empirical CDF in (3.5) to approximate the probability function in (3.1b) is essentially the same as the SAA approach mentioned in Section 3.1.1.

According to (3.2), (3.3), and (3.5), the quantile $Q^{1-\varepsilon}(\bar{g}(x,\xi))$ can be approximated by the empirical quantile defined as:

$$\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,\xi)) = \inf\left\{\gamma \mid \frac{1}{N}\sum_{j=1}^{N}\mathbb{I}(\bar{g}(x,\xi_j) \leq \gamma) \geq 1-\varepsilon\right\} = \bar{g}_{\lceil M \rceil}(x) \tag{3.7}$$

where M equals to $(1-\varepsilon)\times N$, and $\bar{g}_{\lceil M \rceil}(x)$ represents the M-th smallest component of $\bar{G}^N$ with respect to a given x.

In order to compute the stochastic $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,\xi))$ utilizing the deterministic approach, $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,\xi))$ can be further approximated by a ReLU ANN. Since $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,\xi))$ is a function of x, different values of $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,\xi))$ with corresponding x are exploited respectively as targets and observations in the training and validation sets for the ReLU ANN. The schematic diagram of the ReLU ANN for approximating the empirical QF is shown in Figure 3.2:

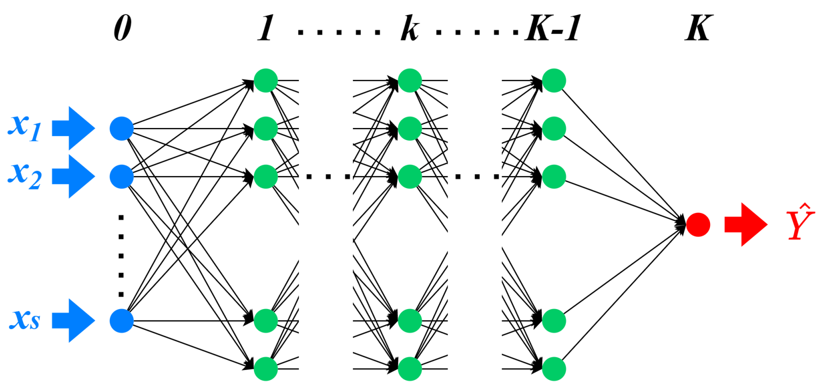


Figure 3.2: Schematic diagram of the ReLU ANN for approximating the empirical quantile

As to Figure 3.2, $x_1, ..., x_s$ are input decision variables. $\hat{Y}$ indicates the quantile value predicted by the ReLU ANN. Layers $1, ..., k, ..., K-1$ denote hidden layers. Layer 0 and layer K are the input and output layers, respectively.

Subsequently, the problem including (3.4a) and (3.4b) can be transformed to the following form with the ReLU ANN embedded to approximate the empirical QF:

$$\min_{x} \quad f(x) \tag{3.8a}$$

$$\text{s.t.} \quad \hat{Y} \leq 0 \tag{3.8b}$$

$$H_s^0 = x_s \qquad s = 1, ..., S \tag{3.8c}$$

$$a_j^k = \sum_{l=1}^{L} W_{lj}^k H_l^{k-1} + b_j^k \qquad j = 1, ..., J \quad k = 1, ..., K \quad l = 1, ..., L \tag{3.8d}$$

$$H_j^k \geq a_j^k \tag{3.8e}$$

$$0 \leq H_j^k \leq \mathcal{M} \tag{3.8f}$$

$$H_j^k \leq \mathcal{M}(1 - dz_j^k) \tag{3.8g}$$

$$H_j^k \leq a_j^k + \mathcal{M}(1 - dl_j^k) \tag{3.8h}$$

$$dz_j^k + dl_j^k = 1 \tag{3.8i}$$

$$dz_j^k, dl_j^k \in \{0, 1\} \tag{3.8j}$$

$$\hat{Y} = \sum_{l=1}^{L} W_l^K H_l^{K-1} + b^K \tag{3.8k}$$

As to the above formulation, equations (3.8c)-(3.8k) express the incorporated ReLU ANN in the optimization framework. Equation (3.8c) represents the input layer. Hidden layers are modelled as (3.8d)-(3.8j). Max-affine operators in the hidden layers are formulated as (3.8e)-(3.8j). Equation (3.8k) describes the output layer of the embedded ReLU ANN. H_s^0 and x_s are neuron outputs from the input layer and corresponding decision variables, respectively. s is the index of each neuron in the input layer. S is the number of neurons in the input layer. j, k, and l are the index of each neuron in the current layer, the index of the layer, and the index of each neuron in the previous layer, respectively. J, K, and L are the number of neurons in the current layer, the number of layers (excluding the input layer), and the number of neurons in the previous layer, respectively. Note that both J and L are different at different k. For instance, when k equals to 1 in (3.8d), J and L are equivalent to a user-defined value and S, respectively. a_j^k, W_{lj}^k, W_l^K, H_l^{k-1}, H_l^k, and H_l^{K-1} are linear combinations of outputs from the previous layer, weights between the current and previous layers, weights between the last hidden layer and output layer, neuron outputs from the previous layer, neuron outputs from the current layer, and neuron outputs from the last hidden layer, respectively. b_j^k and b^K are biases for a certain hidden layer and the output layer, respectively. dz_j^k and dl_j^k are binary variables for max-affine operators in the ReLU

ANN. $\mathcal{M}$ is a big number. If $f(x)$ is linear or approximated by another ReLU ANN, the above optimization formulation will be a mixed-integer linear programming (MILP) problem that can be solved deterministically utilizing off-the-shelf solver such as CPLEX in GAMS.

Notably, since $\bar{g}\left(x,\xi\right) = \max\limits_{i=1,\ldots,w} g_i(x,\xi)$ is used in the quantile-based reformulation, the proposed approach can be applied to the problem with any number of constraints in the JCC. In addition, the proposed method is applicable to different problems, including those with black-box functions. A demonstration of how to implement the proposed approach for solving a JCCP is presented in the next section.

3.1.3 Case study

The case study is the ethylene glycol (EG) production process optimization considering measurement uncertainty. As to the studied EG production process which is shown in Figure 3.3, ethylene oxide (EO) and water are first fed into the CSTR to produce EG. The reaction equations of synthesizing EG and associated by-products are given as:

$$EO + H_2O \xrightarrow{R_1} EG \tag{3.9a}$$

$$EG + EO \xrightarrow{R_2} DEG \tag{3.9b}$$

$$DEG + EO \xrightarrow{R_3} TEG \tag{3.9c}$$

DEG and TEG are diethylene glycol and triethylene glycol, respectively, which are undesirable by-products. The kinetic models of the above reaction equations are shown below:

$$R_1 = k_1 c_{EO} c_{H_2O} \tag{3.10a}$$

$$R_2 = k_2 c_{EO} c_{EG} \tag{3.10b}$$

$$R_3 = k_3 c_{EO} c_{DEG} \tag{3.10c}$$

with the kinetic parameters $k_1 = 1.8302\times10^{10} exp(-\dfrac{87987}{R_g T})$, $k_2 = 2.1k_1$, and $k_3 = 2.2k_1$ [100]. R_g and T are the ideal gas constant and reaction temperature, respectively. c_{EO}, c_{H_2O}, c_{EG}, and c_{DEG} are molar concentrations of EO, water, EG, and DEG, respectively. The output of the CSTR is connected to the flash separator. The product (EG) is collected from the bottom of the flash separator. Meanwhile, 97% of the top flash vapour stream is recycled to mix with the feed flow. The entire EG production process is simulated using Aspen Plus as well as assuming the vapour-liquid equilibrium as ideal phase behaviour. All the above-mentioned settings are the same as the design in Kahrs' work [101].

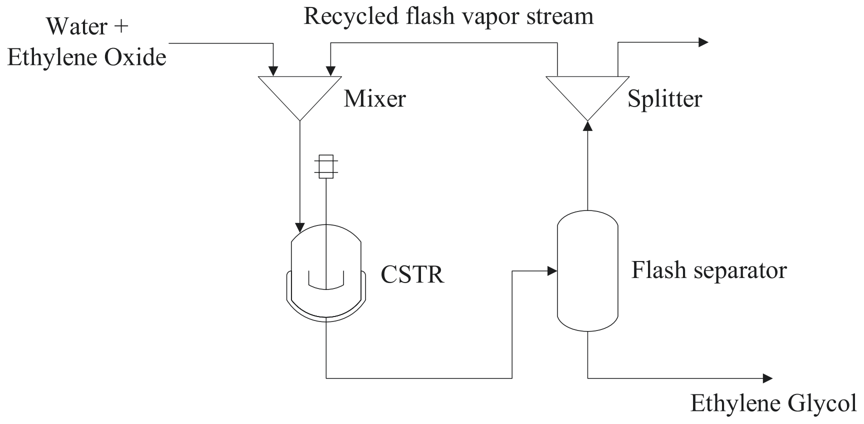


Figure 3.3: Flowsheet of the studied ethylene glycol production process

The process optimization in this section is to maximize the EG yield (EG molar flowrate in the bottom of the flash separator, F_{EG}, unit: kmol/hr). The decision variables include the water feed flowrate (F_w, unit: kmol/hr), the volume of the CSTR (V_{CSTR}, unit: m^3), and temperature of the flash separator (T_f, unit: K). The EO feed flowrate, the operating temperature of the CSTR, and the fraction of recycled flash vapour stream are fixed at 625 kmol/hr, 295 K, and 0.97, respectively. The EO conversion rate (X_{EO}) and the EG mole fraction in the bottom flow of the flash separator (x_{EG}) are both required to be greater or equal to 0.5. To consider measurement uncertainty, two Gaussian noises $\xi_{X_{EO}}$ ($\mathcal{N}(0,1)\times 0.03$) and $\xi_{x_{EG}}$ ($\mathcal{N}(0,1) \times 0.015$) are added to X_{EO} and x_{EG} computed from the full-order model simulated by Aspen Plus, respectively. The corresponding optimization formulation is given as:

$$\max_{F_w, V_{CSTR}, T_f} \quad F_{EG} \tag{3.11a}$$

$$\text{s.t.} \quad Pr\left\{ \begin{array}{l} X_{EO} + \xi_{X_{EO}} \geq 0.5 \\ x_{EG} + \xi_{x_{EG}} \geq 0.5 \end{array} \right\} \geq 1 - \varepsilon \tag{3.11b}$$

$$700 \leq F_w \leq 5000 \tag{3.11c}$$

$$221 \leq V_{CSTR} \leq 321 \tag{3.11d}$$

$$300 \leq T_f \leq 390 \tag{3.11e}$$

Equations (3.11c)-(3.11e) are bounds of decision variables. F_{EG}, X_{EO}, and x_{EG} are dependent variables depending on F_w, V_{CSTR}, and T_f. Equation 3.11b is the joint chance constraint that can be approximated adopting the ReLU ANN approximation approach.

To exploit the ReLU ANN approximation for the joint chance constraint, the above joint chance constraint is first transformed equivalently to the following form:

$$Pr\left\{ \begin{array}{l} -X_{EO} - \xi_{X_{EO}} + 0.5 \leq 0 \\ -x_{EG} - \xi_{x_{EG}} + 0.5 \leq 0 \end{array} \right\} \geq 1 - \varepsilon \tag{3.12}$$

$-X_{EO} - \xi_{X_{EO}} + 0.5$ and $-x_{EG} - \xi_{x_{EG}} + 0.5$ in (3.12) are denoted as $g_{1,C1}$ and $g_{2,C1}$, respectively. The subscript $C1$ is employed for symbols used in this case study. $\bar{g}_{C1}$ can be attained via $\bar{g}_{C1} = \max\{g_{1,C1}, g_{2,C1}\}$, with respect to a certain set of (F_w, V_{CSTR}, T_f). Then, equation (3.12) can be reformulated and approximated by the empirical QF based on (3.7). The number of samples for the empirical QF in this section is 10^3. Since such the empirical QF is a function of (F_w, V_{CSTR}, T_f), it can be further modelled by a ReLU ANN. For a certain value of $1 - \varepsilon$, 2000 samples of empirical QF values and corresponding (F_w, V_{CSTR}, T_f) sets are collected as training and validation data for the ReLU ANN. For the ReLU ANN structure, there are 2 hidden layers in the ReLU ANN, as well as 50 and 30 neurons in the first and second hidden layers, respectively. Afterward, the ReLU ANN is incorporated into the original optimization framework in this case study to replace the joint chance constraint, and the original objective function is modeled by another ReLU ANN (with the same structure as the previous ReLU ANN). As a result, this reformulated optimization model incorporates two ReLU ANNs. Such the reformulated optimization problem is given as:

$$\min_{x} \quad \hat{F}_{EG} \tag{3.13a}$$

$$\text{s.t.} \quad \hat{Y}_{C1} \leq 0 \tag{3.13b}$$

$$700 \leq F_w \leq 5000 \tag{3.13c}$$

$$221 \leq V_{CSTR} \leq 321 \tag{3.13d}$$

$$300 \leq T_f \leq 390 \tag{3.13e}$$

$\hat{F}_{EG}$ and $\hat{Y}_{C1}$ represent the values of F_{EG} and QF predicted by the embedded ReLU ANNs, respectively. The formulations of the incorporated ReLU ANNs are based on (3.8c)-(3.8k), and they are neglected in the above optimization formulation for simplification. Finally, multiple ReLU ANN-based optimizations individually according to different training sets for different $1-\varepsilon$ are solved employing CPLEX in GAMS. The gained optimal results are shown in Table 3.1.

Table 3.1: Optimal results of the EG production optimization w.r.t. different $1-\varepsilon$

$1-\varepsilon$	F_w	V_{CSTR}	T_f	$\hat{F}_{EG}/F_{EG}$	$Pr\left\{\begin{array}{l} X_{EO}+\xi_{X_{EO}} \geq 0.5 \\ x_{EG}+\xi_{x_{EG}} \geq 0.5 \end{array}\right\}$
0.8	1048.887	321	365.415	427.134/426.261	0.8006
0.99	916.229	321	368.191	396.610/395.247	0.9903

$\hat{F}_{EG}/F_{EG}$ are computations from the ReLU ANN/full-order model based on the same operating conditions

According to Table 3.1, the ReLU ANN approximation of F_{EG} is highly accurate. In addition, the achieved optimal solutions are able to satisfy the constraints with the probabilities close to the required confidence levels. Therefore, by using the presented method, the JCCP in this case study can be reliably handled without over-conservatism.

The constraint satisfaction probability based on a certain $1-\varepsilon$ in Table 3.1 is calculated through the following steps: 1) Plug a certain set of optimal decision variables into the full-order model to generate corresponding X_{EO} and x_{EG}; 2) Add the Gaussian noises $\xi_{X_{EO}}$ and $\xi_{x_{EG}}$ to the generated X_{EO} and x_{EG}, respectively; 3) Sample $-X_{EO}-\xi_{X_{EO}}+0.5$ and $-x_{EG}-\xi_{x_{EG}}+0.5$ 10^6 times individually to produce corresponding 10^6-component vectors, which are denoted as $h_{1,C1}$ and $h_{2,C1}$, respectively; 4) Gain the 10^6-component vector $\bar{h}_{C1}$ via $\bar{h}_{C1}=\max\{h_{1,C1}, h_{2,C1}\}$; 5) Calculate the percentage of components in $\bar{h}_{C1}$ less or equal to 0, and such the percentage is equal to the constraint satisfaction probability based on the optimal decision variables with respect to a certain $1 - \varepsilon$.

3.2 A recurrent neural network-based approach for joint chance constrained stochastic optimal control

3.2.1 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 [112].

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 [113]. 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.

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 [114]. 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 are limited to individual chance constraints and unable to address hard input constraints [115]. Sample-based SMPC approaches use sampling methods to realize uncertain parameters for characterizing stochastic system dynamics [116]. The sample-based SMPC based on the Monte Carlo simulation is presented in Moen's work [1]. 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 [117]. The sequential approach proposed in Navia's work [2] is an iterative method utilized to solve the stochastic optimal control problem. In the sequential approach, the inverse mapping method [118] 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 [119, 120]. Among numerous machine learning techniques, recurrent neural networks (RNN) have been widely used for modelling nonlinear dynamic systems [121, 122]. 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 behavior in a way conceptually similar to nonlinear state-space differential equation models [123]. 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 [124]. To further enhance performance on sequential data modeling, more advanced RNNs, such as the Long Short-Term Memory (LSTM) and the Gated Recurrent Unit (GRU), have been developed. Due to outstanding performance on sequence prediction, different types of RNNs have been applied to MPC [125]. Hertneck et al. proposed a neural network-based supervised-learning framework to approximate the robust MPC for enhancing the computational efficiency [126]. Drgoňa et al. [127] exploited deep time-delay neural networks and regression trees to approximate the MPC to reduce the implementation cost. Quan and Chung [128] 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 which is extended from the method in Section 3.1, 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 further 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 modeled 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 [1]. The comparison is shown in the hydrodesulphurisation process case study. Moreover, this approach is applicable to both, stochastic linear and nonlinear dynamic systems, involving any number of stochastic parameters and constraints.

This study is structured as follows. Section 3.2.2 presents the stochastic optimal control problem through a motivating example. Section 3.2.3 discusses the reformulation of the chance constraints through quantile function. Section 3.2.4 presents the proposed RNN approach for stochastic optimal control with the illustrating example studied followed by a chemical industry case study presented in section 3.2.5. Moreover, the extension of the proposed approach to the SMPC of the hydrodesulphurisation process is illustrated in Section 3.2.6.

3.2.2 Stochastic optimal control problem

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

$$\min_{u_j \in \mathbb{U}} \quad \mathbb{E}\left[\sum_{j=0}^{K-1} J(x_j, u_j) + J_f(x_K)\right] \tag{3.14a}$$

$$\text{s.t.} \quad x_{j+1} = f_D(x_j, u_j, \xi) \quad j = 0, .., K-1 \tag{3.14b}$$

$$Pr(g_i(x_j, u_{j-1}, \xi) \leq 0, i = 1, \cdots, w) \geq 1 - \varepsilon, \quad j = 1, .., K \tag{3.14c}$$

$$x_0 = x_{t_0} \tag{3.14d}$$

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 (3.14b) represents the process dynamic model. f_D is a function describing the discrete-time system dynamics. Inequality (3.14c) 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. w 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.

3.2.2.1 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:

$$\min_{u_j} \quad 2 \times \sum_{j=0}^{9} \frac{\Delta t}{2}(\mathbb{E}[x_{1,j}] + \mathbb{E}[x_{1,j+1}]) \tag{3.15a}$$

$$\text{s.t.} \quad x_j = [x_{1,j}, x_{2,j}]\,, \ \ j = 0, ..., 10 \tag{3.15b}$$

$$x_{j+1} = f_D(x_j, u_j, \xi_1, \xi_2), \ \ j = 0, ..., 9 \tag{3.15c}$$

$$Pr(-x_{1,j} - 3.3 \leq 0, -x_{2,j} - 3.3 \leq 0) \geq 1 - \varepsilon, \quad j = 1, ..., 10 \tag{3.15d}$$

$$|u_j| \leq 2, \quad j = 0, ..., 9 \tag{3.15e}$$

$$x_{1,0} = 2, \quad x_{2,0} = 0 \tag{3.15f}$$

$$\xi_1 \sim \mathcal{N}(0, 0.25), \quad \xi_2 \sim U(-0.5, 0.5) \tag{3.15g}$$

$j = 0, ..., 10$ 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 (3.15f) 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 (3.15d) 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 Figure 3.4. 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 Figure 3.4, respectively. The achieved optimal objective value is -2.6561 which is based on $x_{1,DN}$. In Figure 3.4, 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 Figure 3.4 ($x_{1,DU}$ 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 Figure 3.4, 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 (3.15a)-(3.15g).

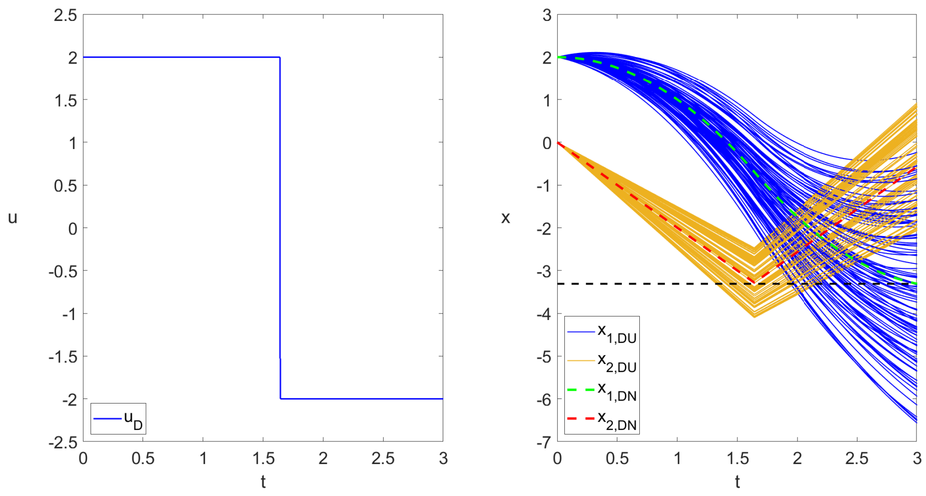


Figure 3.4: The optimal u_j sequence of the deterministic illustrating example (u_D), and the corresponding x_1 and x_2. The subscripts DU and DN are used for the results under different uncertainty scenarios and under the nominal scenario, respectively.

3.2.3 Reformulation and empirical approximation

3.2.3.1 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:

$$Pr(g_i(x, u, \xi) \leq 0, i = 1, \cdots, w) \geq 1 - \varepsilon \tag{3.16}$$

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 (3.16) ensures that all constraints $g_{i=1,\ldots,w}(x, u, \xi) \leq 0$ are satisfied simultaneously to a certain confidence level $1 - \varepsilon$.

As mentioned in Section 3.2.1, since joint chance-constrained problems are difficult to solve [13, 109, 111], 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) $\phi_g(\gamma) = Pr(g(x, u, \xi) \leq \gamma)$, the $1 - \varepsilon$ level quantile of $g(x, u, \xi)$ is defined as [129]:

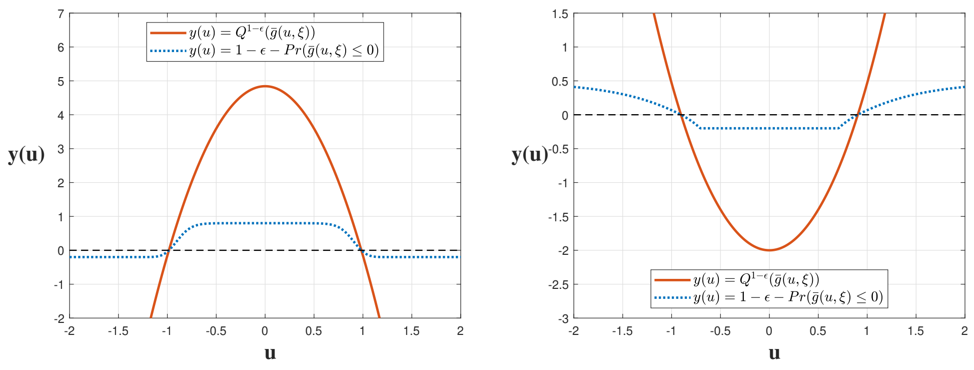
$$\begin{aligned}
&Q^{1-\varepsilon}(g(x, u, \xi)) \\
&= \inf \{\gamma \in \mathbb{R} \mid Pr(g(x, u, \xi) \leq \gamma) \geq 1 - \varepsilon\} \\
&= \inf \{\gamma \in \mathbb{R} \mid \phi_g(\gamma) \geq 1 - \varepsilon\}
\end{aligned} \tag{3.17}$$

According to (3.17), $Q^{1-\varepsilon}(g(x, u, \xi))$ outputs the minimum γ that satisfies $Pr(g(x, u, \xi) \leq \gamma) \geq 1 - \varepsilon$, that is, $\phi_g(\gamma) \geq 1 - \varepsilon$. Thus, the following relationship holds:

$$\begin{aligned}
Pr(g(x, u, \xi) \leq \gamma) \geq 1 - \varepsilon \Leftrightarrow \phi_g(\gamma) \geq 1 - \varepsilon \\
\Leftrightarrow Q^{1-\varepsilon}(g(x, u, \xi)) \leq \gamma
\end{aligned} \tag{3.18}$$

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 (3.16), we define $\bar{g}(x, u, \xi) = \max_{i=1,\ldots,w} g_i(x, u, \xi)$. Note that $Pr(\bar{g}(x, u, \xi) \leq 0)$ is equivalent to $Pr(g_i(x, u, \xi) \leq 0, i = 1, \cdots, w)$ in (3.16). Accordingly, the inequality (3.16) can be equivalently reformulated as the follow:

$$Q^{1-\varepsilon}(\bar{g}(x, u, \xi)) \leq 0 \tag{3.19}$$



(a) (b)

Figure 3.5: Comparisons between $1-\varepsilon-Pr(\bar{g}(u,\xi)\leq 0)$ and $Q^{1-\varepsilon}(\bar{g}(u,\xi))$ 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 $\bar{g}(u,\xi)=\max\{g_1(u,\xi_1),g_2(u,\xi_2)\}$, $g_1(u,\xi_1)=1.5\xi_1u^2-3$, $\xi_1\sim\mathcal{N}(0,1)$, $g_2(u,\xi_2)=2\xi_2u^2-2$, $\xi_2\sim U(-2,2)$. $\bar{g}$ here only depends on u and ξ for simplification and the demonstration purpose.

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 Figure 3.5, $Q^{1-\varepsilon}(\bar{g}(u,\xi))$ 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 an arbitrary number of uncertain constraints.

3.2.3.2 Empirical approximation

While it is challenging to evaluate the quantile function in (3.19) 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 $\Xi^N=\{\xi_1,\ldots,\xi_N\}$ 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:

$$\widetilde{\phi}_{\bar{g}}(\gamma) = \frac{1}{N}\sum_{l=1}^{N}\mathbb{I}(\bar{g}(x,u,\xi_l) \leq \gamma) \tag{3.20}$$

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

$$\mathbb{I}(\bar{g}(x,u,\xi_l) \leq \gamma) = \begin{cases} 0, & \text{for} \quad \bar{g}(x,u,\xi_l) > \gamma \\ 1, & \text{for} \quad \bar{g}(x,u,\xi_l) \leq \gamma \end{cases} \tag{3.21}$$

Note that using the empirical CDF in (3.20) to approximate the probability function in (3.16) is essentially the same as the SAA approach mentioned in Section 3.1.1.

According to (3.17), (3.18), and (3.20), the quantile $Q^{1-\varepsilon}(\bar{g}(x,u,\xi))$ can be approximated by the empirical quantile defined as:

$$\begin{aligned} &\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,u,\xi)) \\ &= \inf\left\{\gamma \mid \frac{1}{N}\sum_{l=1}^{N}\mathbb{I}(\bar{g}(x,u,\xi_l) \leq \gamma) \geq 1-\varepsilon\right\} \\ &= \bar{g}_{\lceil M \rceil}(x,u) \end{aligned} \tag{3.22}$$

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 (3.19) can be approximated by the empirical quantile function in (3.22), and (3.19) can be reformulated as:

$$\widetilde{Q}^{1-\varepsilon}(\bar{g}(x,u,\xi)) \leq 0 \tag{3.23}$$

Note that the above sample-based empirical approximation is also applied to 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, a 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 [130], there is no concrete rule to determine the exact number of samples N that must be considered for the above empirical approximation.

3.2.4 RNN-based stochastic optimal control

According to the quantile-based reformulation for the JCC mentioned in the previous section, the SOCP including (3.14a)-(3.14d) can be reformulated as:

$$\min_{u_j} \quad \mathbb{E}\left[\sum_{j=0}^{K-1} J(x_j, u_j) + J_f(x_K)\right] \tag{3.24a}$$

$$\text{s.t.} \quad x_{j+1} = f_D(x_j, u_j, \xi) \quad j = 0, .., K-1 \tag{3.24b}$$

$$\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j, u_{j-1}, \xi)) \leq 0 \quad j = 1, .., K \tag{3.24c}$$

$$x_0 = x_{t_0} \tag{3.24d}$$

where $\widetilde{Q}^{1-\varepsilon}(\bar{g}(x_j, u_{j-1}, \xi))$ denotes the empirical QF at a certain sampling instant j. $\bar{g}(x_j, u_{j-1}, \xi) = \max\limits_{i=1,\ldots,w} g_i(x_j, u_{j-1}, \xi)$. 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 proposed approach based on the RNN and the quantile-based reformulation is presented. Then, the illustrating example under uncertainty mentioned in Section 3.2.2 is solved in Section 3.2.4.2.

3.2.4.1 RNN-based approach

To reduce the solution complexity of the optimization including (3.24a)-(3.24d), the Long Short-Term Memory (LSTM) is exploited to generate the surrogate model for handling the dynamic model in (3.24b). Meanwhile, this surrogate model is also used to predict quantile values for the inequality (3.24c). Notably, the LSTM is more preferable than the Gated Recurrent Unit (GRU) when prediction accuracy is critical since the LSTM has more gates and parameters [20]. Thus, the LSTM is used in this research. The detailed explanation of RNN and LSTM is in Section 1.2.2 and Appendix A2.1. The illustration of the LSTM for the quantile value prediction is shown in Figure 3.6.

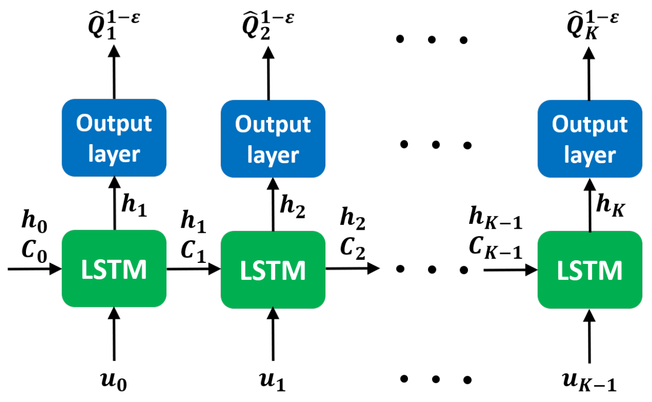


Figure 3.6: Schematic diagram of the LSTM for the quantile value prediction

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 $\hat{Q}_{j+1}^{1-\varepsilon}$ 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 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 (3.24a) were predicted by another LSTM model (the second LSTM model).

The two LSTM models are further incorporated into the SOCP including (3.24a)-(3.24d). The SOCP incorporating the two LSTM models is given as:

$$\min_{u_j} \quad \sum_{j=0}^{K} \hat{E}_j \tag{3.25a}$$

$$\text{s.t.} \quad \hat{Q}_j^{1-\varepsilon} \leq 0 \quad j = 1, ..., K \tag{3.25b}$$

$$\hat{Q}_{j=1,...,K}^{1-\varepsilon} = L_1(u_{j=0,...,K-1}) \tag{3.25c}$$

$$\hat{E}_{j=1,...,K} = L_2(u_{j=0,...,K-1}) \tag{3.25d}$$

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. $\hat{Q}_j^{1-\varepsilon}$ 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 $\hat{Q}_j^{1-\varepsilon}$. 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 Appendix.

3.2.4.2 Illustrating example revisited

The illustrating problem studied in Section 3.2.2 is solved by utilizing the RNN-based approach and demonstrated in the following subsections.

Problem reformulation

The JCC in (3.15d) can be reformulated equivalently as the discretized quantile-based inequality which is given as:

$$\widetilde{Q}_{I,j}^{1-\varepsilon}(\bar{g}_{I,j}) \leq 0 \quad j = 1, ..., 10 \tag{3.26}$$

where $\bar{g}_{I,j} = \max\{g_{1,I,j}, g_{2,I,j}\}$. $g_{1,I,j}$ 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 (3.15a), (3.15c), (3.15d), and (3.26) are elaborated in Appendix. Moreover, the calculated expected values and quantile values are used for the LSTM training mentioned in the following subsection.

LSTM model training

Two LSTMs are employed to address (3.15a) and (3.26). The first LSTM is used to predict the quantile values in (3.26) 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 Figure 3.6. 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 (3.15a) for approximating (3.15a). 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 [131]. 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 Appendix.

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 [92]. The formulation of the optimization involving the two LSTM models and the related details are shown in Appendix.

Results

The attained optimal results are shown in Figure 3.7. Note that u_R in Figure 3.7 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 (3.15d). 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 Appendix.

As can be seen from Figure 3.7, 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 Figure 3.7, to ensure the outcomes under different uncertainty scenarios being feasible with the required probability of 0.8.

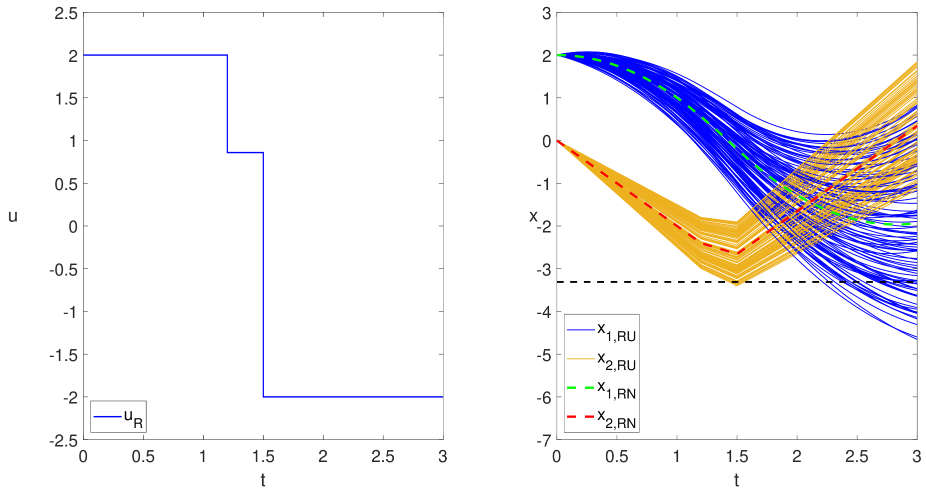


Figure 3.7: The optimal sequence of u_j obtained from the proposed RNN-based approach (u_R), and the corresponding x_1 and x_2. The subscripts RU and RN are used for the results under different uncertainty scenarios and under the nominal scenario, respectively.

3.2.5 Case study: hydrodesulphurisation process

In this case study, the RNN-based approach is applied to the problem of the hydrodesulphurisation process (HDS) taken from [2]. 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 Figure 3.8. As can be seen from Figure 3.8, 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.

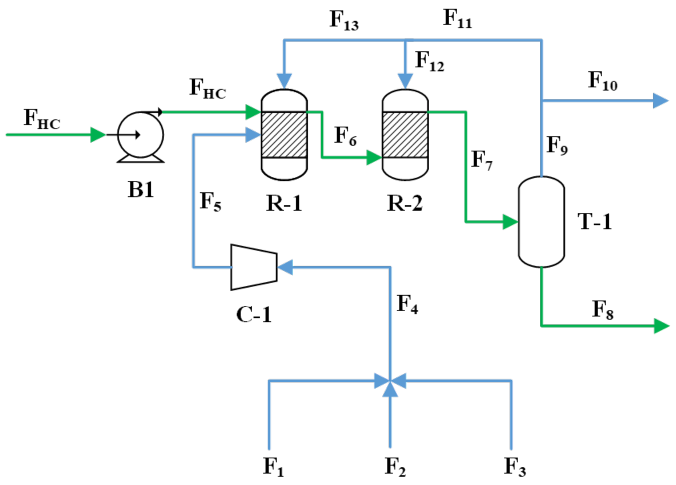


Figure 3.8: Schematic diagram of the studied hydrodesulphurisation process

3.2.5.1 Model

The HDS is modeled 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 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:

$$\min_{\dot{F}_1,\dot{F}_2,\dot{F}_{10}} \quad \int_{t_0}^{t_f} C_{H4}X_1\dot{F}_1 + C_{H3}X_2\dot{F}_2 dt \tag{3.27a}$$

$$\text{s.t.} \quad \tau\frac{d\dot{F}_x^{H2}}{dt} + \dot{F}_x^{H2} = \dot{F}_{HC}\rho \tag{3.27b}$$

$$\frac{VP}{ZR_gT}\frac{dX_{H2}}{dt} = \dot{F}_5X_5 - \dot{F}_{10}X_{H2} - \dot{F}_x^{H2} \tag{3.27c}$$

$$\dot{F}_5 = \dot{F}_{10} + \dot{F}_x^{H2} \tag{3.27d}$$

$$\dot{F}_5 = \dot{F}_1 + \dot{F}_2 + \dot{F}_3 \tag{3.27e}$$

$$\dot{F}_5X_5 = \dot{F}_1X_1 + \dot{F}_2X_2 + \dot{F}_3X_3 \tag{3.27f}$$

$$Pr(X_{H2} \geq 0.7, X_5 \geq 0.9) \geq 1 - \varepsilon \tag{3.27g}$$

$$0 \leq \dot{F}_1 \leq 1400 \tag{3.27h}$$

$$0 \leq \dot{F}_2 \leq 790 \tag{3.27i}$$

$$0 \leq \dot{F}_{10} \leq 1500 \tag{3.27j}$$

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 (3.27b). 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 (3.27c) 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 (3.27d). Equation (3.27e) is exploited to calculate $\dot{F}_5$. The component balance of hydrogen can be described by (3.27f). 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. (3.27h)-(3.27j) are the bounds for the decision variables of the SOCP. The values of parameters in the SOCP are shown in Table 3.2.

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 [1]. 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 research refers to the modified sample-based method.

Table 3.2: Values of parameters in the SOCP including (3.27a)-(3.27j)

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	$(m^3bar)/(Kkmol)$
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] $X_3 \sim \mathcal{N}(0.85, 0.013)$

3.2.5.2 Results and discussion

Deterministic optimal control

To show the impacts of the random parameters ρ and X_3 on the SOCP of the HDS, the optimization including (3.27a)-(3.27j) 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 (3.27g) 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 Figure 3.9. The corresponding optimal results of X_5 and X_{H2} are $X_{5,DN}$ and $X_{H2,DN}$ illustrated in Figure 3.9, respectively. The achieved optimal objective value is 106.8659 € which is based on $\dot{F}_{1,D}$ and $\dot{F}_{2,D}$. In Figure 3.9, 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 Figure 3.9 ($X_{5,DU}$ 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.

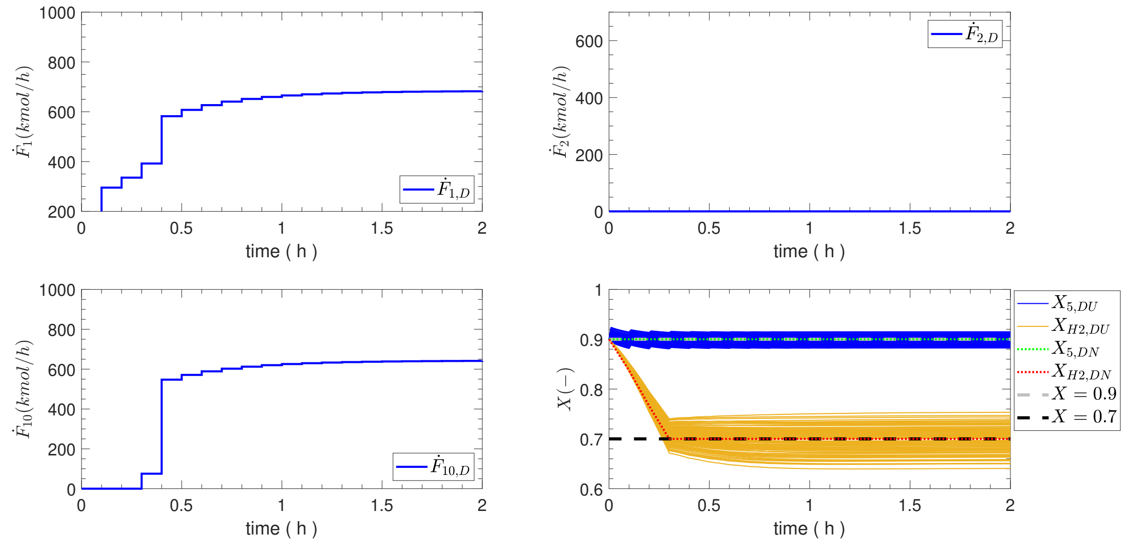


Figure 3.9: The optimal results of the deterministic optimal control of the HDS. The subscripts DU and DN are used for the results under different uncertainty scenarios and under the nominal scenario, respectively.

According to the results shown in Figure 3.9, $X_{5,DU}$ 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.

Stochastic optimal control

The SOCP of the HDS is solved using the proposed RNN-based approach. While using the proposed approach, the SOCP including (3.27a)-(3.27j) should be first discretized. The details of the discretization and the discretized SOCP formulation are illustrated in Appendix. 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 Appendix. By utilizing the proposed approach to solve the SOCP of the HDS, the attained optimal results are shown in Figure 3.10. 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 Appendix. Note that $\dot{F}_{1,R}$, $\dot{F}_{2,R}$, and $\dot{F}_{10,R}$ in Figure 3.10 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.

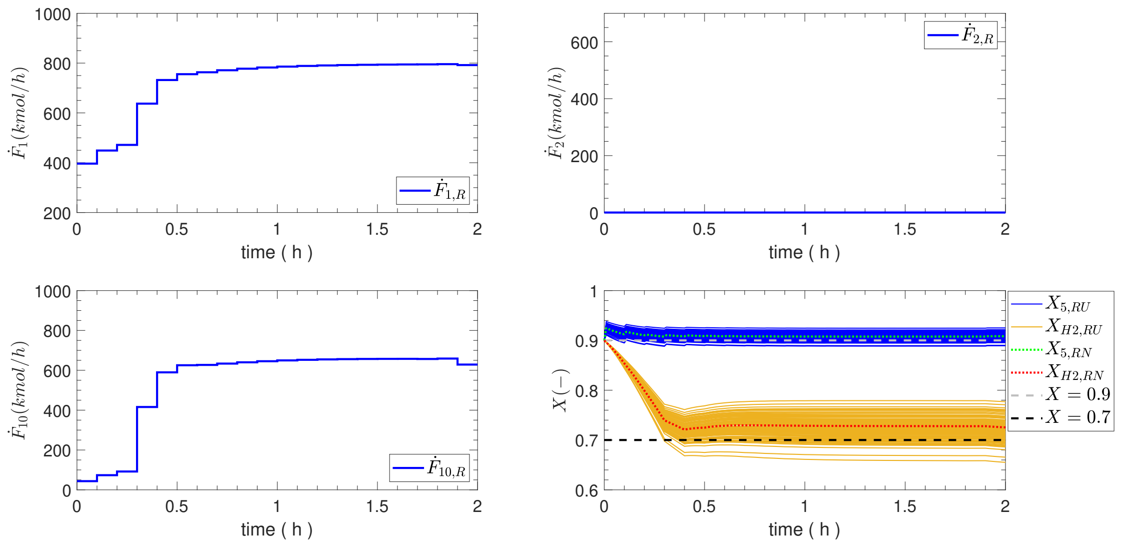


Figure 3.10: The optimal results of the SOCP of the HDS gained from the proposed approach. The subscripts RU and RN are used for the results under different uncertainty scenarios and under the nominal scenario, respectively.

As can be seen from Figure 3.10, 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

Table 3.3: 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

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 from Moen's study [1]. The comparison is organized in Table 3.3. According to Table 3.3, 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.

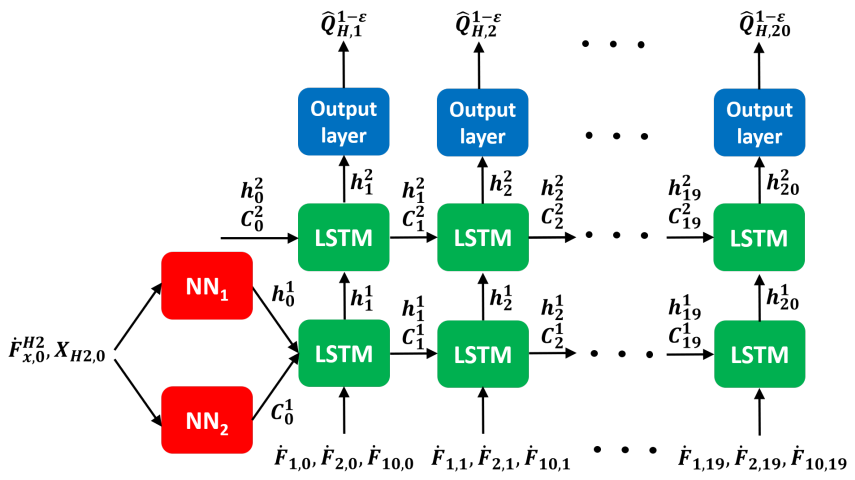
3.2.6 SMPC implementation

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 Figure 3.6 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.

3.2.6.1 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 Figure 3.11. According to Figure 3.11, 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 Appendix.



$\hat{Q}_{H,1}^{1-\varepsilon}$
$\hat{Q}_{H,2}^{1-\varepsilon}$
$\hat{Q}_{H,20}^{1-\varepsilon}$
Output layer
LSTM
NN$_1$
NN$_2$
$\dot{F}_{x,0}^{H2}, X_{H2,0}$
h_0^2 C_0^2
h_0^1 C_0^1
h_1^2 C_1^2 h_2^2 C_2^2 h_{19}^2 C_{19}^2 h_{20}^2
h_1^1 C_1^1 h_2^1 C_2^1 h_{19}^1 C_{19}^1 h_{20}^1
$\dot{F}_{1,0}, \dot{F}_{2,0}, \dot{F}_{10,0}$
$\dot{F}_{1,1}, \dot{F}_{2,1}, \dot{F}_{10,1}$
$\dot{F}_{1,19}, \dot{F}_{2,19}, \dot{F}_{10,19}$

Figure 3.11: 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 Appendix. 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 Appendix. 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 Figure 3.12. The reason for only choosing 100 results is to avoid too many overlapping curves shown in Figure 3.12 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 Figure 3.13. 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 Figure 3.13, 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.

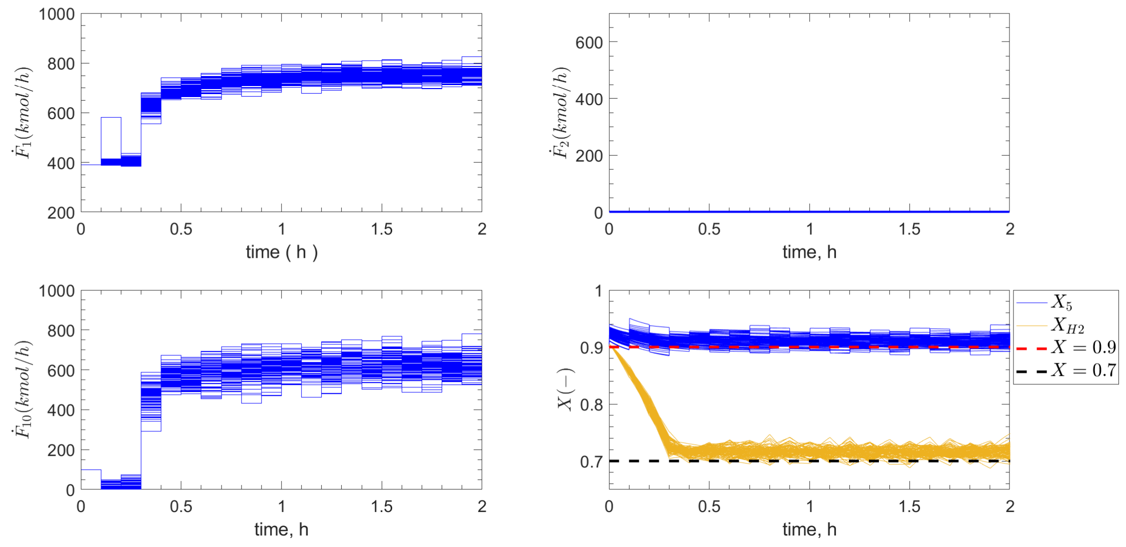


Figure 3.12: 100 SMPC results of the HDS, which are attained by exploiting the proposed RNN-based approach

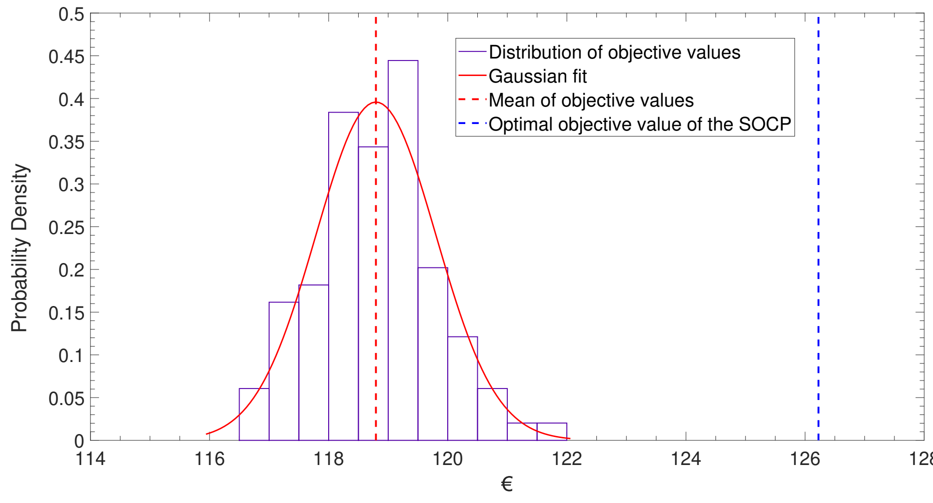


Figure 3.13: The distribution of the objective values gained from the 1000 SMPC implementations

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 [1]. The comparison is organized in Table 3.4. One remark here is that the solution time corresponds to a method in Table 3.4 is the overall time of the SMPC implementation for 20 sampling instants.

Table 3.4: 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 3.4, 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.

3.3 Conclusion

A novel method involving the empirical quantile reformulation of JCC and the ReLU ANN approximation is proposed in Section 3.1 to address JCCPs. This method relies on the sampled data of objective and constraint function values, which can be obtained from explicit process model equations or black-box process simulators. Hence, the method can be used for different process optimization problems, such as problems with explicit models or black-box constraint functions. In the meantime, there is no restriction on the number of constraints in the JCC. The problem is finally converted to the solution of a deterministic MILP problem. Through a case study, it is shown that the developed method can efficiently solve a nonlinear joint chance-constrained process optimization problem without over-conservatism.

In Section 3.2, the above proposed approach is extended to become a new RNN-based method to solve stochastic optimal control and stochastic model predictive control problems with JCCs. In this extended 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 this RNN-based approach can obtain the solution satisfying the confidence level effectively. Compared with the sample-based method modified from the literature, the RNN-based 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. This RNN-based method 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.

Chapter 4

Kernel Distributionally Robust Chance-Constrained Optimization

Abstract

In Section 4.1 of this chapter, a distributionally robust chance-constrained optimization (DR-CCP) method is proposed based on the kernel ambiguity set. The kernel ambiguity set is established via the kernel mean embedding (KME) and the maximum mean discrepancy (MMD) between distributions. The proposed approach can be formulated as two different models. The first one is a mixed-integer model employing the indicator function for handling the joint chance constraint. The second one is a continuous optimization model using the Conditional Value-at-Risk (CVaR) approximation to approximate the indicator function. The proposed method is compared with the popular Wasserstein ambiguity set based approach. A numerical example and a nonlinear process optimization problem are studied to demonstrate its efficacy. In Section 4.2 of this chapter, the presented kernel-based approach is further combined with a neural network-like deep kernel to enhance its performance. The deep kernel-based method is applied to an alkylation process optimization to demonstrate its performance.

4.1 Distributionally Robust Chance-Constrained Optimization with Kernel Ambiguity Set

4.1.1 Introduction

Real-world optimization problems often involve uncertainty. It is hard to obtain reliable optimal decisions for problems under uncertainty since constraint violations may be caused by unexpected random effects. Chance-constrained programming (CCP) is a widely-used method for optimization under uncertainty [10]. While using the CCP, the optimal decision is required to be feasible with a user-defined probability. There are two types of CCP: the individual chance-constrained programming (ICCP) and the joint chance-constrained programming (JCCP) [11, 21]. The ICCP only requires each constraint to be satisfied with its own confidence level. The JCCP which is more natural and general, ensures that all the constraints are satisfied simultaneously to a certain confidence level. Although the JCC can limit the probability of violating any constraints in a problem, it is difficult to solve as it requires dealing with the multidimensional distribution [13]. In addition, a JCCP problem is convex only when the following requirements are satisfied: 1) the original constraints in the joint chance constraint are jointly convex in both decision variables and random parameters. 2) the random parameters follow log-concave distributions [132]. Accordingly, JCCP problems are generally solved using approximation methods, such as the analytical approximation methods and the sampling-based methods [28].

In practical applications, exact distributions of uncertainty are often hard to be known. Thus, to solve a JCCP problem, instead of anticipating the exact distribution of uncertainty, a more practical way is to consider the worst-case distribution from all potential data-generating distributions to obtain the distributionally robust solution. This is the concept of the distributionally robust chance-constrained programming (DRCCP) [70]. In the field of DRCCP, the potential data-generating distributions are also called candidate distributions, and the set of candidate distributions is typically referred to as the ambiguity set [133]. The candidate distributions are learned from historical data and characterized through certain known information (e.g., moments, structure properties, domain knowledge, etc) of the unknown data-generating distribution. The ambiguity set composed of candidate distributions should contain the true data-generating distribution with high confidence, and it should also be small enough to exclude distributions that may lead to overly conservative decisions [134]. Additionally, the ambiguity set should be easily extracted from data, and it should enable a tractable reformulation of the DRCCP as a mathematical program that can be solved using off-the-shelf optimization solver [135].

Among different types of ambiguity sets, the moment-based and metric-based ambiguity sets have received significant attention and been intensively studied [57–59]. A moment-based ambiguity set contains all distributions that satisfy certain moment constraints [133, 136, 137]. A certain level of moment information should be known in advance for constructing a moment-based ambiguity set. In addition, since different distributions might share the same moments, moment constraints in a moment-based ambiguity set might be too loose to exclude distributions that may cause overly conservative solutions [60]. Moreover, distributionally robust chance constraints based on moment-based ambiguity sets are not capable of tightly approximating chance constraints even if sufficient data is available [61]. To avoid the above-mentioned drawbacks, metric-based ambiguity sets are attractive alternatives that define the ambiguity set in the space of probability distributions [138]. More specifically, all candidate distributions in a metric-based ambiguity set are centered around the nominal distribution within a radius determined by the prescribed probability metric. The nominal distribution is established based on historical data. The radius size of the ambiguity set is a user-defined hyper-parameter allowing the user to control the degree of conservatism of the underlying optimization problem [30]. The Wasserstein metric is extensively used for metric-based ambiguity sets. However, the existing studies rely on some nontrivial assumptions on uncertain constraints to obtain tractable reformulations. For instance, in most studies of Wasserstein DRCCPs [3, 4, 61, 70–73], uncertain constraints are assumed to be affine in uncertainty. Although there are a few works proposing Wasserstein DRCCP methods for nonlinear uncertain constraints, they still require some assumptions on uncertain constraints. Gu and Wang [74] proposed a Wasserstein DRCCP approach for nonlinear uncertain constraints which are restricted to be quadratic convex in uncertainty. Hota et al. [75] proposed Wasserstein DRCCP methods for uncertain constraints which are limited to be either concave or convex in uncertainty. Based on the aforementioned, the Wasserstein DRCCP approaches proposed so far rely on nontrivial assumptions on uncertain constraints, that hinder their practical applications. The research on the Wasserstein DRCCP approach for general uncertain constraints has not been found so far.

To overcome the mentioned drawback of the Wasserstein DRCCPs, the DRCCP over the kernel ambiguity set (kernel DRCCP) proposed in this research would be a promising solution because it is compatible with general uncertain constraints without any assumption. The reformulations of the presented kernel DRCCP are unified for arbitrary constraint functions. These tractable reformulations of the kernel DRCCP are expressed as two different optimization models. The first model is a mixed-integer programming model exploiting the indicator function to address the joint constraint satisfaction probability (JCSP) involved in the JCC. To reduce the computational burden of the first model, the second model is a continuous optimization model employing the Conditional Value-at-Risk (CVaR) approximation [139] to approximate the indicator function for handling the JCSP.

The kernel ambiguity set employed in this study is established by using kernel mean embedding (KME) [140] and maximum mean discrepancy (MMD) [141]. Regarding the existing studies on the kernel ambiguity set, the kernel ambiguity set has only been applied to distributionally robust optimization (DRO) problems so far. In [16], the MMD based DRO is studied, and it is connected to the Hilbert norm regularization. In [142], the DROs based on different types of kernel ambiguity sets are developed, e.g., the RKHS norm-ball ambiguity set, polytope kernel ambiguity set, etc. In addition, the DRO over the RKHS norm-ball ambiguity set is applied to the worst-case risk quantification in another Zhu's research [143]. Those existing works only exploit kernel ambiguity sets to address DRO problems which consider distributional ambiguity in objective functions. To the best of the authors' knowledge, the study of the DRCCP based on the kernel ambiguity set has not been found so far. Therefore, the main contribution of this work is the development of a new DRCCP method based on the kernel ambiguity set which possesses significant advantages over other existing DRCCP methods. Regarding the limitation of the proposed DRCCP approach, the presented method is based on the discretized support, and the support discretization approach used in this study is only applicable to rectangular and unbounded support sets (more details are mentioned in Sections 4.1.4, 4.1.7, and 4.1.8). More studies on the more general support discretization method for the presented DRCCP approach should be done in future work.

This research is structured as follows. DRCCP is introduced in Section 4.1.2. The details about the kernel ambiguity set are elaborated in Section 4.1.3. The derivation for the different models of the kernel DRCCP is presented in Sections 4.1.4-4.1.5. The adaption of the presented approach to the problems involving distributionally robust objective functions is shown in Section 4.1.6. In Section 4.1.7, the presented DRCCP method is applied to a numerical example to compare with the popular Wasserstein DRCCP approach. Subsequently, the application of the presented approach to a nonlinear process optimization under uncertainty is demonstrated in Section 4.1.8.

4.1.2 Distributionally robust chance-constrained optimization

As mentioned in Section 4.1.1, joint chance-constrained programming (JCCP) is more comprehensive and intuitive than individual chance-constrained programming (ICCP). Hence, we only focus on the JCCP in this research. A detailed explanation of JCCP and ICCP is in Section 1.2.3. However, the exact probability distribution of uncertainty in conventional chance-constrained programming (CCP) is not always available. Accordingly, to address distributional ambiguity of uncertainty, the ambiguity set containing candidate distributions can be introduced to the CCP to result in the DRCCP. The general formulation of the DRCCP is given as:

$$\min_{x \in \mathcal{X}} \quad f(x) \tag{4.1a}$$

$$\text{s.t.} \quad \min_{\mathbb{P}(\xi) \in \mathcal{P}} Pr(g_i(x, \xi) \leq 0, \quad \forall i = 1, ..., w) \geq 1 - \delta \tag{4.1b}$$

where $\mathbb{P}(\xi)$ is the probability distribution function of ξ. $\mathcal{P}$ is the ambiguity set composed of all candidate distributions. Inequality (4.1b) represents the distributionally robust joint chance constraint (DRJCC) ensuring the worst-case JCSP higher than or equal to the user-defined confidence level $1 - \delta$, based on the ambiguity set.

The expression (4.1b) can be rewritten equivalently as the worst-case violation probability form:

$$\max_{\mathbb{P}(\xi) \in \mathcal{P}} Pr\left(\bigcup_{i=1}^{w} g_i(x, \xi) > 0\right) \leq \delta \tag{4.2}$$

Notably, the left-hand side of the above expression represents the worst-case violation probability, which is the maximum probability of constraint violation ($g_1(x, \xi) > 0$ or $g_2(x, \xi) > 0$, or ..., or $g_w(x, \xi) > 0$) under all possible probability distributions in the ambiguity set. The expression (4.2) is more useful for deriving tractable DRCCP formulations.

4.1.3 Kernel ambiguity set

The kernel ambiguity set which is the core of the proposed Kernel DRCCP, is based on the kernel mean embedding (KME) [140] and the maximum mean discrepancy (MMD) [141] explained below.

A symmetric function $k : \Xi \times \Xi \to \mathbb{R}$ can be called as a positive definite kernel if $\sum_{j=1}^{N} \sum_{l=1}^{N} a_j a_l k(\xi_j, \xi_l) \geq 0$, $\forall N \in \mathbb{Z}^+$, $\forall \{\xi_1, ..., \xi_N\} \subset \Xi$, $\forall a_1, ..., a_N \in \mathbb{R}$ [144]. For simplification, all the kernels mentioned in the following parts of this study refer to positive definite kernels. According to Moore-Aronszajn theorem [145], with a kernel k given, a Hilbert space $\mathcal{H}$ of real-valued functions and a map $\phi : \Xi \to \mathcal{H}$ exist, such that $\langle f, \phi(\xi) \rangle_{\mathcal{H}} = f(\xi)$ for all $f \in \mathcal{H}$ and $\xi \in \Xi$. The Hilbert space $\mathcal{H}$ mentioned above is the so-called reproducing kernel Hilbert space (RKHS) [146] associated with the kernel k, which has the reproducing property, i.e., $\langle f, \phi(\xi) \rangle_{\mathcal{H}} = f(\xi)$ for all $f \in \mathcal{H}$ and $\xi \in \Xi$. Notably, the map ϕ is commonly known as a feature map, and $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the inner product on $\mathcal{H}$. Furthermore, the reproducing property implies that $k(\xi, \xi_0) = \langle \phi(\xi), \phi(\xi_0) \rangle_{\mathcal{H}} = \langle k(\xi, \cdot), k(\xi_0, \cdot) \rangle_{\mathcal{H}}$ for any $\xi, \xi_0 \in \Xi$ [147]. Note that $\phi(\xi)$ can be interchangeably written as $k(\xi, \cdot)$.

The KME is to make non-parametric representations of distributions in an RKHS [145]. More specifically, with a probability distribution $\mathbb{P}$ and a kernel k given, the KME of $\mathbb{P}$ is given as:

$$\mu_{\mathbb{P}}(\cdot) = \int k(\xi, \cdot) d\mathbb{P}(\xi) \tag{4.3}$$

The above expression of the KME can be intuitively deemed as the expectation of the feature map under the distribution $\mathbb{P}$. $\mu_{\mathbb{P}}$ exists in $\mathcal{H}$ if $\mathbb{E}_{\xi \sim \mathbb{P}}[k(\xi, \xi)] < \infty$ [148]. The quality of the KME is highly influenced by the properties of the associated kernel k. If k is a characteristic kernel, the kernel mean map $\mathbb{P} \mapsto \mu_{\mathbb{P}}$ is an injective map that $\mu_{\mathbb{P}}$ is unique for any distribution $\mathbb{P}$ [149]. Furthermore, if k is a universal kernel, the corresponding RKHS can be considered as a rich enough RKHS, and any continuous function can be approximated arbitrarily well by a function in such the RKHS [142]. One remark here is that all universal kernels are also characteristic kernels, but not vice versa, as mentioned in Theorem 8 of [141]. The detailed definitions of characteristic kernels and universal kernels are shown in Definition 3.2 and Definition 3.3 in [140], respectively. Accordingly, we only consider universal kernels in this work due to the above reasons.

By exploiting the KME, the distance between two probability distributions in a RKHS can be determined. More specifically, with a kernel given, $\|\mu_{\mathbb{P}} - \mu_{\mathbb{P}_0}\|_{\mathcal{H}}$ defines the distance between the distributions $\mathbb{P}$ and $\mathbb{P}_0$ in the associated RKHS $\mathcal{H}$ [150]. $\|\mu_{\mathbb{P}} - \mu_{\mathbb{P}_0}\|_{\mathcal{H}}$ is known as the maximum mean discrepancy (MMD) [140]. One remark here is that $\|\cdot\|_{\mathcal{H}}$ is the Hilbert space norm, and $\|f\|_{\mathcal{H}} := \sqrt{\langle f, f \rangle_{\mathcal{H}}}, \forall f \in \mathcal{H}$. As mentioned in Lemma 4 of [141], the MMD metric is a special case of the integral probability metrics (IPMs). The IPM reduces to the

MMD while the associated space of real-valued bounded measurable functions on Ξ (which is $\mathcal{F}$) follows the condition: $\mathcal{F} = \{f : \|f\|_{\mathcal{H}} \leq 1\}$. More details about the IPMs are introduced in Section 1.2.5. The MMD can be expressed in the form with the associated kernel k, which is given as:

$$\|\mu_{\mathbb{P}} - \mu_{\mathbb{P}_0}\|_{\mathcal{H}}^2 = \mathbb{E}_{\xi,\xi' \sim \mathbb{P}}[k(\xi, \xi^{'})] - 2\mathbb{E}_{\xi \sim \mathbb{P}}\mathbb{E}_{\xi_0 \sim \mathbb{P}_0}[k(\xi, \xi_0)] + \mathbb{E}_{\xi_0,\xi_0' \sim \mathbb{P}_0}[k(\xi_0, \xi_0^{'})] \tag{4.4}$$

Note that $\xi^{'}$ and $\xi_0^{'}$ are independent copies of ξ and ξ_0, respectively. The above expression can be estimated by using the empirical estimator with the samples collected from $\mathbb{P}$ and $\mathbb{P}_0$ [141].

The kernel ambiguity set in this work embeds distributions in a RKHS norm-ball via the KME. In the kernel ambiguity set, the KME representations of candidate distributions are centered around the KME representation of the nominal distribution within a RKHS norm ball radius determined by the MMD. If the universal kernel (eg., the Gaussian kernel, the Laplacian kernel, the rational quadratic kernel, etc.) is employed, the corresponding RKHS can be considered as a rich enough RKHS, and the associated kernel ambiguity set may contain the true distribution with high confidence if the RKHS norm-ball radius is large enough [16].

Mathematically, the kernel ambiguity set $\mathcal{K}_{\mathcal{C}}$ in this work is defined as:

$$\mathcal{K}_{\mathcal{C}} = \left\{\mathbb{P} \in \mathcal{P}(\Xi) : \int \phi d\mathbb{P} = \mu, \mu \in \mathcal{C} \subseteq \mathcal{H}\right\}$$

$\mathcal{P}(\Xi)$ denotes the set of all Borel probability measures on support Ξ. Both sides of the constraint $\int \phi d\mathbb{P} = \mu$ are functions in $\mathcal{H}$, where $\mathcal{H}$ is an RKHS whose feature map is ϕ. μ can be viewed as a generalized moment vector (infinite dimension vector), which is constrained to lie within the RKHS norm ball $\mathcal{C} \subseteq \mathcal{H}$. The RKHS norm ball $\mathcal{C}$ is defined as:

$$\mathcal{C} = \{\mu : \left\|\mu - \mu_{\hat{\mathbb{P}}_0}\right\|_{\mathcal{H}} \leq \varepsilon_{\mathcal{H}}\}$$

$\hat{\mathbb{P}}_0$ is the nominal distribution, which is an empirical distribution based on the realizations of uncertainty. $\mu_{\hat{\mathbb{P}}_0}$ is the embedding of the nominal distribution $\hat{\mathbb{P}}_0$. $\varepsilon_{\mathcal{H}}$ is the RKHS norm-ball radius which is a user-defined hyper-parameter. The solution conservatism of the kernel DRCCP can be controlled by adjusting $\varepsilon_{\mathcal{H}}$.

4.1.4 Kernel distributionally robust chance constraints

In this work, the kernel distributionally robust chance constraint (KDRCC) is the DRJCC based on the kernel ambiguity set. The proposed kernel DRCCP can be formulated as different models based on different KDRCC formulations. The different KDRCC formulations are derived from the worst-case violation probability in inequality (4.2). With the kernel ambiguity set, the worst-case violation probability in inequality (4.2) can be rewritten as:

$$
\begin{aligned}
\max \quad & Pr\left(\bigcup_{i=1}^{w} g_i(x,\xi) > 0\right) \\
\text{s.t.} \quad & \left\|\mu - \mu_{\hat{\mathbb{P}}_0}\right\|_{\mathcal{H}} \leq \varepsilon_{\mathcal{H}} \\
& \int \phi(\xi) d\mathbb{P}(\xi) = \mu
\end{aligned}
\tag{4.5}
$$

where $\mu_{\hat{\mathbb{P}}_0}$ is the embedding of the nominal distribution $\hat{\mathbb{P}}_0$. $\varepsilon_{\mathcal{H}}$ is the radius of the RKHS norm ball, which is a user-defined parameter. According to [143], to make problem (4.5) computationally tractable, μ and $\mu_{\mathbb{P}_0}$ in problem (4.5) can be handled using empirical embedding approaches, i.e., $\mu = \sum_{j=1}^{M} \alpha_j \phi(\eta_j)$ and $\mu_{\hat{\mathbb{P}}_0} = \sum_{j=1}^{N} \dfrac{1}{N}\phi(\xi_j)$. $\{\eta_j\}_{j=1}^{M}$ are expansion points used to discretize the support of the candidate distributions, and they are composed of the collected samples $\{\xi_j\}_{j=1}^{N}$ and sampled support points $\{\zeta_j\}_{j=1}^{Y}$. The sampled support points do not have to be from the original data samples [142]. There are several ways to produce sampled support points [142], e.g., selecting grid points evenly from a bounded sample space, generating synthetic points via the convex combinations of collected samples, producing new points by adding perturbations to collected samples, etc. The number of the sampled support points can be treated as a user-defined hyper-parameter. As stated in [142] and [143], the expansion points containing collected samples and sampled support points are exactly the support of the nominal and worst-case distributions, that allow us to anticipate potential distributions with limited data and make problem (4.5) to be distributionally robust. α_j is the probability mass on η_j. Based on the above-mentioned empirical embedding of μ and $\mu_{\hat{\mathbb{P}}_0}$, the formulation (4.5) can be reformulated as:

$$\begin{aligned}
\max_{\alpha} \quad & \sum_{j=1}^{M} \alpha_j \mathbb{I}\left(\bigcup_{i=1}^{w} g_i(x,\eta_j) > 0\right) && \text{(4.6a)}\\
\text{s.t.} \quad & \left\| \sum_{j=1}^{M} \alpha_j \phi(\eta_j) - \sum_{j=1}^{N} \frac{1}{N}\phi(\xi_j) \right\|_{\mathcal{H}} \leq \varepsilon_{\mathcal{H}} && \text{(4.6b)}\\
& \sum_{j=1}^{M} \alpha_j = 1 && \text{(4.6c)}\\
& \alpha_j \geq 0, \quad \forall j = 1, ..., M && \text{(4.6d)}
\end{aligned}$$

The objective (4.6a) is the empirical expression of the worst-case violation probability. The indicator function $\mathbb{I}\left(\bigcup_{i=1}^{w} g_i(x,\eta_j) > 0\right)$ in (4.6a) is defined as:

$$\mathbb{I}\left(\bigcup_{i=1}^{w} g_i(x,\eta_j) > 0\right) = \begin{cases} 0, & \text{for} \quad g_i(x,\eta_j) \leq 0, \forall i = 1, ..., w \\ 1, & \text{for} \quad g_i(x,\eta_j) > 0, \exists i = 1, ..., w \end{cases} \tag{4.7}$$

Inequality (4.6b) can be rewritten as a more tractable form according to (4.4), which is:

$$\alpha^T K_\eta \alpha - 2\frac{1}{N}\alpha^T K_{\eta\xi}\mathbb{1} + \frac{1}{N^2}\mathbb{1}^T K_\xi \mathbb{1} \leq \varepsilon_{\mathcal{H}}^2 \tag{4.8}$$

where $\alpha := [\alpha_1, ..., \alpha_M]^T$ and $\mathbb{1} := [1, ..., 1]^T$. K_η, $K_{\eta\xi}$, and K_ξ are the Gram matrices associated with the kernel k, i.e., $K_\eta := [k(\eta_j, \eta_l)]_{jl}$, $K_{\eta\xi} := [k(\eta_j, \xi_l)]_{jl}$, $K_\xi := [k(\xi_j, \xi_l)]_{jl}$. Then, the optimization including (4.6a)-(4.6d) can be reformulated as:

$$\begin{aligned}
\max_{\alpha} \quad & \mathcal{I}(x)^T \alpha \\
\text{s.t.} \quad & \alpha^T K_\eta \alpha - 2\frac{1}{N}\alpha^T K_{\eta\xi}\mathbb{1} + \frac{1}{N^2}\mathbb{1}^T K_\xi \mathbb{1} \leq \varepsilon_{\mathcal{H}}^2 \\
& \sum_{j=1}^{M} \alpha_j = 1 \\
& \alpha_j \geq 0, \quad \forall j = 1, ..., M
\end{aligned} \tag{4.9}$$

where $\mathcal{I}(x)$ is a column vector defined as:

$$\mathcal{I}(x) = \left[\mathbb{I}\left(\bigcup_{i=1}^{w} g_i\left(x, \eta_1\right) > 0 \right), \cdots, \mathbb{I}\left(\bigcup_{i=1}^{w} g_i\left(x, \eta_M\right) > 0 \right) \right]^{\top}$$

Notably, as mentioned in [143], the problem (4.9) is strictly feasible while expansion points contain collected sample points, i.e. $\{\xi_j\}_{j=1}^{N} \subseteq \{\eta_j\}_{j=1}^{M}$.

The first inequality constraint in problem (4.9) can be rewritten as a second-order cone constraint, and then problem (4.9) can be reformulated as a second-order cone programming (SOCP) problem which is given as:

$$\begin{aligned} \max_{\alpha} \quad & \mathcal{I}(x)^T \alpha \\ \text{s.t.} \quad & \begin{bmatrix} R\alpha \\ \frac{1}{N}\mathbb{1}^T K_{\eta\xi}^T \alpha \\ \frac{1}{N}\mathbb{1}^T K_{\eta\xi}^T \alpha \end{bmatrix} - \begin{bmatrix} \nvdash \\ \frac{1}{2N^2}\mathbb{1}^T K_{\xi} \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2} \\ \frac{1}{2N^2}\mathbb{1}^T K_{\xi} \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2} \end{bmatrix} \in \mathfrak{L}^{M+2} \\ & I\alpha \in \mathbb{R}_{+}^{M} \\ & \mathbb{1}^T \alpha = 1 \end{aligned} \tag{4.10}$$

where $\mathfrak{L}^{M+2}$ denotes the $(M+2)$-dimension Lorenz cone. $\nvdash$ is a column zero vector and I is the identity matrix.

Next, based on the conic duality, the dual formulation of the above SOCP is given as:

$$\begin{aligned} \min_{\psi,\phi,\rho,\nu} \quad & -\nu - \left(\frac{1}{2N^2}\mathbb{1}^T K_{\xi}\mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2} \right)\phi_1 - \left(\frac{1}{2N^2}\mathbb{1}^T K_{\xi}\mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2} \right)\phi_2 \\ \text{s.t.} \quad & \mathbb{1}\nu + R^T\psi + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_1 + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_2 + I^T\rho = -\mathcal{I}(x) \\ & \|\psi\|_2^2 + \phi_1^2 \le \phi_2^2 \\ & \rho_j \ge 0, \quad \forall j = 1, ..., M \end{aligned} \tag{4.11}$$

where $\psi, \phi_1, \phi_2, \rho,$ and ν are dual variables. Notably, since ρ in the above model is non-negative and only exists in the equality constraint, ρ can be removed from the above model.

Therefore, the above model can be rewritten as:

$$\begin{aligned}
\min_{\psi,\phi,\nu} \quad & -\nu - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2}\right)\phi_1 - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2}\right)\phi_2 \\
\text{s.t.} \quad & \mathbb{1}\nu + R^T\psi + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_1 + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_2 \leq -\mathcal{I}(x) \\
& \|\psi\|_2^2 + \phi_1^2 \leq \phi_2^2
\end{aligned} \tag{4.12}$$

Subsequently, by plugging the above dual problem into the DRCCP including (4.1a)-(4.1b) to replace the DRJCC stated in (4.1b), and dropping the min operator, we get

$$\begin{aligned}
\min_{x\in\mathcal{X},\psi,\phi,\nu} \quad & f(x) \\
\text{s.t.} \quad & -\nu - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2}\right)\phi_1 \\
& \qquad - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2}\right)\phi_2 \leq \delta \\
& \mathbb{1}\nu + R^T\psi + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_1 + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_2 \leq -\mathcal{I}(x) \\
& \|\psi\|_2^2 + \phi_1^2 \leq \phi_2^2
\end{aligned} \tag{4.13}$$

The column vector $\mathcal{I}(x)$ can be computed via different methods that convert the above optimization problem into different formulations.

4.1.4.1 Indicator function-based formulation

The column vector $\mathcal{I}(x)$ in problem (4.13) can be computed based on the following mixed-integer model:

$$\mathcal{I}(x) = [\bar{y}_1, ..., \bar{y}_M]^T \tag{4.14a}$$

$$\mathcal{M}(\widetilde{y}_{ij} - 1) + \epsilon \leq g_i(x, \eta_j) \leq \mathcal{M}\widetilde{y}_{ij}, \quad \forall i = 1, ..., w, \forall j = 1, ..., M \tag{4.14b}$$

$$\bar{y}_j - 1 \leq (\sum_{i=1}^{w} \widetilde{y}_{ij}) - 1 \leq w\bar{y}_j - 1, \quad \forall j = 1, ..., M \tag{4.14c}$$

$$\bar{y}_j \in \{0, 1\}, \quad \forall j = 1, ..., M \tag{4.14d}$$

$$\widetilde{y}_{ij} \in \{0, 1\}, \quad \forall i = 1, ..., w, \forall j = 1, ..., M \tag{4.14e}$$

where w and M are numbers of constraints and expansion points, respectively. $\mathcal{M}$ is a big number that $\mathcal{M} \geq g_i(x, \eta_j), \forall i = 1, ..., w, \forall j = 1, ..., M, \forall x \in \mathcal{X}$. ϵ is a small positive number which is set as 10^{-8} in this work. $\bar{y}_j$ and $\widetilde{y}_{ij}$ are 0-1 integer variables, which are defined as:

$$\widetilde{y}_{ij} = \begin{cases} 0, & \text{for} \quad g_i(x, \eta_j) \leq 0 \\ 1, & \text{for} \quad g_i(x, \eta_j) > 0 \end{cases} \tag{4.15}$$

$$\bar{y}_j = \begin{cases} 0, & \text{for} \quad g_i(x, \eta_j) \leq 0, \forall i = 1, ..., w \\ 1, & \text{for} \quad g_i(x, \eta_j) > 0, \exists i = 1, ..., w \end{cases} \tag{4.16}$$

Inequality (4.14b) is used to identify whether the value of $g_i(x, \eta_j)$ is greater than 0, i.e. $\widetilde{y}_{ij} = 1 \Leftrightarrow g_i(x, \eta_j) > 0$. Inequality (4.14c) is utilized to identify whether $g_i(x, \eta_j) > 0, \exists i = 1, ..., w$, i.e. $\bar{y}_j = 1 \Leftrightarrow \sum_{i=1}^{w} \widetilde{y}_{ij} \geq 1 \Leftrightarrow g_i(x, \eta_j) > 0, \exists i = 1, ..., w$. Based on the aforementioned, the problem (4.13) can be rewritten as the following mixed-integer program:

$$\begin{aligned}
\min_{x,\psi,\phi,\nu,\bar{y},\tilde{y}} \quad & f(x) \\
\text{s.t.} \quad & -\nu - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2}\right)\phi_1 \\
& \qquad\qquad - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2}\right)\phi_2 \leq \delta \\
& \mathbb{1}\nu + R^T\psi + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_1 + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_2 \leq -[\bar{y}_1, ..., \bar{y}_M]^T \\
& \|\psi\|_2^2 + \phi_1^2 \leq \phi_2^2 \\
& \bar{y}_j - 1 \leq (\sum_{i=1}^{w} \widetilde{y}_{ij}) - 1 \leq w\bar{y}_j - 1, \quad \forall j = 1, ..., M \\
& \mathcal{M}(\widetilde{y}_{ij} - 1) + \epsilon \leq g_i(x, \eta_j) \leq \mathcal{M}\widetilde{y}_{ij}, \quad \forall i = 1, ..., w, \forall j = 1, ..., M \\
& \bar{y}_j \in \{0, 1\}, \quad \forall j = 1, ..., M \\
& \widetilde{y}_{ij} \in \{0, 1\}, \quad \forall i = 1, ..., w, \forall j = 1, ..., M
\end{aligned} \tag{4.17}$$

The kernel DRCCP based on the above formulation is named the indicator function-based kernel DRCCP (IF-KDRCCP) in this work. If $f(x)$ and $g_i(x)$ are linear functions, the above optimization becomes a mixed integer quadratically constrained optimization (MIQCP) problem which can be solved using off-the-shelf solver such as XPRESS or CPLEX.

4.1.5 Kernel DRCCP with CVaR approximation

To further reduce the computational burden of the IF-KDRCCP in model (4.17), we use the worst-case CVaR inequality to approximate the worst-case violation probability in (4.2). The details about the worst-case CVaR approximation [73, 151].

The worst-case CVaR inequality for approximating inequality (4.2) can be expressed as:

$$\max_{\mathbb{P}(\xi)\in\mathcal{P}} \mathrm{CVaR}\left(\max_{i=1,\dots,w}\{g_i(x,\xi)\}, \delta\right) \le 0 \tag{4.18a}$$

$$\Rightarrow \max_{\mathbb{P}(\xi)\in\mathcal{P}} \min_{\beta\in\mathbb{R}} \left\{\beta + \frac{1}{\delta}\mathbb{E}_{\mathbb{P}(\xi)}\left(\left[\max_{i=1,\dots,w}\{g_i(x,\xi)\} - \beta\right]^+\right)\right\} \le 0 \tag{4.18b}$$

According to Theorem 2 of [151], we suppose that $\mathcal{P}$ is a compact convex set, and thus $\max_{\mathbb{P}(\xi)\in\mathcal{P}}$ as well as $\min_{\beta\in\mathbb{R}}$ in the above inequality can be switched:

$$\min_{\beta\in\mathbb{R}} \left\{\beta + \frac{1}{\delta}\max_{\mathbb{P}(\xi)\in\mathcal{P}}\mathbb{E}_{\mathbb{P}(\xi)}\left(\left[\max_{i=1,\dots,w}\{g_i(x,\xi)\} - \beta\right]^+\right)\right\} \le 0 \tag{4.19}$$

By addressing the distributional ambiguity of the above inequalities with the KME and MMD, $\max_{\mathbb{P}(\xi)\in\mathcal{P}}\mathbb{E}_{\mathbb{P}(\xi)}\left([\max_{i=1,\dots,w}\{g_i(x,\xi)\}-\beta]^+\right)$ in inequality (4.19) can be reformulated as:

$$\begin{aligned}
\max_{\alpha}\quad & G(x)^T\alpha \\
\text{s.t.}\quad & \alpha^T K_\eta \alpha - 2\frac{1}{N}\alpha^T K_{\eta\xi}\mathbb{1} + \frac{1}{N^2}\mathbb{1}^T K_\xi \mathbb{1} \le \varepsilon_{\mathcal{H}}^2 \\
& \sum_{j=1}^{M}\alpha_j = 1 \\
& \alpha_j \ge 0, \quad \forall j = 1, \dots, M
\end{aligned} \tag{4.20}$$

where

$$G(x) = \left[\left[\max_{i=1,\dots,w}\{g_i(x,\xi_1)\} - \beta\right]^+, \dots, \left[\max_{i=1,\dots,w}\{g_i(x,\xi_M)\} - \beta\right]^+\right]^T$$

is a column vector. Notably, since $G(x)$ does not dependent on α, it is deemed as a constant vector in the above formulation.

Because the only difference between problem (4.20) and problem (4.9) is $G(x)^T$ in the objective function, the dual problem of problem (4.20) can be simply obtained according to the steps for deriving formulation (4.12), which is given as:

$$\begin{aligned}
\min_{\psi,\phi,\nu} \quad & -\nu - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2}\right)\phi_1 - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2}\right)\phi_2 \\
\text{s.t.} \quad & \mathbb{1}\nu + R^T\psi + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_1 + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_2 \le -G(x) \\
& \|\psi\|_2^2 + \phi_1^2 \le \phi_2^2
\end{aligned} \tag{4.21}$$

Subsequently, after plugging model (4.21) into inequality (4.19), we have:

$$\begin{aligned}
& \min_{\beta\in\mathbb{R}} \left\{\beta + \frac{1}{\delta}\min_{\phi,\nu}\left[-\nu - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2}\right)\phi_1 \; - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2}\right)\phi_2\right]\right\} \le 0 \\
& \mathbb{1}\nu + R^T\psi + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_1 + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_2 \le -G(x) \\
& \|\psi\|_2^2 + \phi_1^2 \le \phi_2^2
\end{aligned} \tag{4.22}$$

After dropping the min operators in the first constraint and plugging the result into the general DRCCP formulation including (4.1a)-(4.1b) to replace the DRJCC, the kernel DRCCP with CVaR approximation can be written as:

$$\begin{aligned}
\min_{x,\beta,\psi,\phi,\nu} \quad & f(x) \\
\text{s.t.} \quad & \delta\beta - \nu - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2}\right)\phi_1 - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2}\right)\phi_2 \le 0 \\
& \mathbb{1}\nu + R^T\psi + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_1 + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_2 \le -G(x) \\
& \|\psi\|_2^2 + \phi_1^2 \le \phi_2^2 \\
& G(x) = \left[\left[\max_{i=1,\ldots,w}\{g_i(x,\xi_1)\} - \beta\right]^+, \ldots, \left[\max_{i=1,\ldots,w}\{g_i(x,\xi_M)\} - \beta\right]^+\right]^T
\end{aligned} \tag{4.23}$$

To handle the expression of $G(x)$, the above formulation can be rewritten as:

$$\begin{aligned}
\min_{x,\beta,\psi,\phi,\nu,G} \quad & f(x) \\
\text{s.t.} \quad & \delta\beta - \nu - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2}\right)\phi_1 - \left(\frac{1}{2N^2}\mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2}\right)\phi_2 \le 0 \\
& \mathbb{1}\nu + R^T\psi + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_1 + \frac{1}{N}K_{\eta\xi}\mathbb{1}\phi_2 \le -\left[G_1, ..., G_M\right]^T \\
& \|\psi\|_2^2 + \phi_1^2 \le \phi_2^2 \\
& G_j \ge 0, \quad \forall j = 1, ..., M \\
& G_j \ge g_i(x, \xi_j) - \beta, \quad \forall i = 1, ..., w, \forall j = 1, ..., M
\end{aligned} \tag{4.24}$$

If $f(x)$ and $g_i(x, \eta_j)$ are convex in x, the above model would be convex since the employed CVaR approximation is convexity-preserving. The kernel-based DRCCP based on model (4.24) is named as the CVaR-based kernel DRCCP (KCVaR) in this research. If $f(x)$ and $g_i(x)$ are linear functions, the above optimization becomes a quadratically constrained optimization (QCP) problem.

Note that both models (4.17) and (4.24) are compatible with general uncertain constraints $g_i(x, \eta_j)$, and $g_i(x, \eta_j)$ can be non-linear, non-convex, etc. in uncertainty and decision variables. This is a significant superiority of the kernel DRCCP for the practical application, compared to the existing Wasserstein DRCCP approaches which are limited to specific uncertain constraint forms, e.g., affine in uncertainty [3], quadratic convex in uncertainty [74], concave in uncertainty, or convex in uncertainty [75].

4.1.6 Kernel distributionally robust objective

Although this work focuses on distributionally robust chance constraints, the proposed method can be adapted easily to address the distributionally robust objective function in the distributionally robust optimization (DRO). The general formulation of a distributionally robust objective is given as:

$$\min_{x\in\mathcal{X}} \quad \max_{\mathbb{P}(\xi)\in\mathcal{P}} \mathbb{E}_{\mathbb{P}(\xi)}[f(x, \xi)] \tag{4.25}$$

The DRO is a minimization problem involving an inner worst-case expectation problem. The DRO based on the kernel ambiguity set (kernel DRO) is given as:

$$\min_{x} \max_{\alpha} \quad \sum_{j=1}^{M} \alpha_j f(x, \eta_j) \tag{4.26a}$$

$$\text{s.t.} \quad \left\| \sum_{j=1}^{M} \alpha_j \phi(\eta_j) - \sum_{j=1}^{N} \frac{1}{N} \phi(\xi_j) \right\|_{\mathcal{H}} \leq \varepsilon_{\mathcal{H}} \tag{4.26b}$$

$$\sum_{j=1}^{M} \alpha_j = 1 \tag{4.26c}$$

$$\alpha_j \geq 0, \quad \forall j = 1, ..., M \tag{4.26d}$$

the above optimization problem can be rewritten as:

$$\begin{aligned} \min_{x} \max_{\alpha} \quad & \mathcal{L}^T \alpha \\ \text{s.t.} \quad & \alpha^T K_\eta \alpha - 2\frac{1}{N} \alpha^T K_{\eta\xi} \mathbb{1} + \frac{1}{N^2} \mathbb{1}^T K_\xi \mathbb{1} \leq \varepsilon_{\mathcal{H}}^2 \\ & \sum_{j=1}^{M} \alpha_j = 1 \\ & \alpha_j \geq 0, \quad \forall j = 1, ..., M \end{aligned} \tag{4.27}$$

where $\mathcal{L}$ is a column vector: $\mathcal{L} = [f(x, \eta_1), ..., f(x, \eta_M)]$. We use the dual problem of the inner maximization problem and merge the min operators. Then, we can obtain the tractable formulation of the kernel distributionally robust optimization problem:

$$\begin{aligned} \min_{x,\psi,\phi,\nu} \quad & -\nu - \left(\frac{1}{2N^2} \mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} + \frac{1}{2} \right) \phi_1 - \left(\frac{1}{2N^2} \mathbb{1}^T K_\xi \mathbb{1} - \frac{\varepsilon_{\mathcal{H}}^2}{2} - \frac{1}{2} \right) \phi_2 \\ \text{s.t.} \quad & \mathbb{1}\nu + R^T \psi + \frac{1}{N} K_{\eta\xi} \mathbb{1} \phi_1 + \frac{1}{N} K_{\eta\xi} \mathbb{1} \phi_2 \leq -[f(x, \eta_1), ..., f(x, \eta_M)] \\ & \|\psi\|_2^2 + \phi_1^2 \leq \phi_2^2 \end{aligned} \tag{4.28}$$

If $f(x, \eta)$ is linear in x, then the resulting model is a SOCP problem which is readily solvable using off-the-shelf solver.

4.1.7 Numerical example

The presented IF-KDRCCP (model (4.17)) and KCVaR (model (4.24)) approaches are applied to a numerical joint chance-constrained example taken from [14], which is given as:

$$\min_{x_1,x_2} \quad x_1 + x_2 \tag{4.29a}$$

$$\text{s.t.} \quad Pr\left\{ \begin{array}{l} \xi_1 x_1 + x_2 \geq 7 \\ \xi_2 x_1 + x_2 \geq 4 \end{array} \right\} \geq 1 - \delta \tag{4.29b}$$

$$x_1, x_2 \geq 0 \tag{4.29c}$$

The random parameters ξ_1 and ξ_2 follow uniform distributions in [1,4] and $[\frac{1}{3}, 1]$, respectively. The analytical solution to this problem is available:

$$\begin{cases} x_1^* = \dfrac{18}{9+8(1-\delta)}, \quad x_2^* = \dfrac{2(9+28(1-\delta))}{9+8(1-\delta)} & \text{for } \delta \in [0.5, 1] \\ x_1^* = \dfrac{9}{11-9(1-\delta)}, \quad x_2^* = \dfrac{41-36(1-\delta)}{11-9(1-\delta)} & \text{for } \delta \in [0, 0.5] \end{cases}$$

Note that δ in this section is set to be 0.1. Thus, the true solution of the problem is $x_1^* = 3.1034$ and $x_2^* = 2.9655$.

As the comparisons to the proposed approaches, the CVaR-based Wasserstein DRCCP (WCVaR) and the CVaR-based Wasserstein DRCCP under the Bonferroni approximation (WCVaR-B) are also applied to this example. The WCVaR employed for this example uses model (A40) in Appendix, and the related details are explained in A3.1. While using the WCVaR-B, the joint chance constraint in (4.29b) is decomposed into two individual chance constraints, and the violation tolerance δ is distributed equally to the two individual chance constraints, i.e., the violation tolerance for each individual chance constraint is 0.05. Both individual chance constraints are under the Wasserstein ambiguity set. The WCVaR-B employed in this example uses model (21) presented in [73]. While solving this numerical example with the DRCCP methods, we assume that we do not have prior knowledge about the bounds of the uncertainty support. Accordingly, while using the Wasserstein-based DRCCP methods, we employ the above-mentioned models that do not consider information of uncertainty support bounds.

Since the objective and constraint functions in this example are linear, the problem-solving through IF-KDRCCP, KCVaR, WCVaR, and WCVaR-B can be addressed as MIQCP, QCP, QCP, and QCP problems, respectively. All the MIQCP and QCP problems in this research are solved by using XPRESS in GAMS.

The studied example is solved with different sample sizes and different ambiguity set radius sizes (ASRS, which is $\varepsilon_{\mathcal{H}}$ or ε_W). With respect to a certain sample size and ASRS, the problem-solving based on each approach is conducted 100 times based on 100 different sample sets with the same sample size.

For the numerical experiment in this section, we assume that we already have a sample set individually containing 80 samples of ξ_1 and ξ_2. Subsequently, we can generate the sampled support points for the IF-KDRCCP and the KCVaR based on the 80-sample set according to the following steps:

1. Calculate the means (μ_1 and μ_2) and the standard deviations (σ_1 and σ_2) of ξ_1 and ξ_2 in the mentioned 80-sample set.
2. Individually select 10 uniformly distributed grid points from the ranges of $[\mu_1 - 2\sigma_1, \mu_1 + 2\sigma_1]$ and $[\mu_2 - 2\sigma_2, \mu_2 + 2\sigma_2]$.
3. Randomly shuffle both sets of the grid points and concatenate them together to be the set ζ.

We use the same set of sampled support points ζ for all the optimizations based on the kernel-based methods. With the same set of sampled support points, we examine the performance of the kernel-based approaches with respect to different real sample sizes and different ASRSs.

In this numerical example, the IF-KDRCCP and the KCVaR approaches are implemented based on the following universal kernels:

$$\text{Gaussian kernel:} \quad k(\xi,\xi_0) = \exp\left(-\gamma \left\|\xi - \xi_0\right\|_2^2\right), \gamma > 0$$

$$\text{Laplacian kernel:} \quad k(\xi,\xi_0) = \exp\left(-\gamma \left\|\xi - \xi_0\right\|_1\right), \gamma > 0$$

$$\text{Rational quadratic kernel:} \quad k(\xi,\xi_0) = \left(\left\|\xi - \xi_0\right\|_2^2 + c^2\right)^{-b}, b > 0, c > 0$$

Based on experiments for this numerical example, changing hyper-parameters in these kernels (γ, c, b) would not significantly influence the optimal solution. Hence, γ in both Gaussian and

Laplacian kernels are determined using the median heuristic [152] based on the expansion points composed of sample support points (grid points) and the real samples. c and b in the rational quadratic kernel are simply set to be 1 and 1.5, respectively.

4.1.7.1 Effect of sample size and ambiguity set size

According to the above setting, the results obtained from the different DRCCP approaches are shown in Figures 4.1-4.6, where the ASRSs and the sample sizes in the ranges of $0.005 \sim 0.02$ and of $20 \sim 80$, respectively. Note that the JCSPs are calculated through the sample average approximation (SAA) with 10^4 samples.

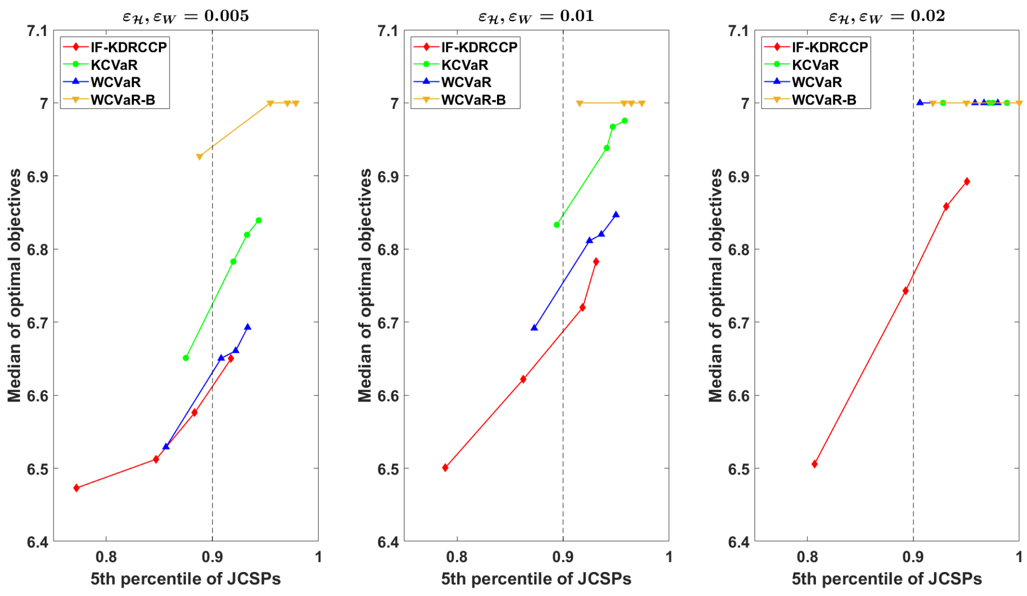


Figure 4.1: Medians of the obtained optimal objectives with respect to the 5th percentiles of the corresponding JCSPs, based on different ASRSs. The markers in each line correspond to sample sizes equal to 20, 40, 60, and 80 in order from left to right. The results from the IF-KDRCCP and the KCVaR are based on the Gaussian kernel. The dash line denotes the target JCSP value of 0.9.

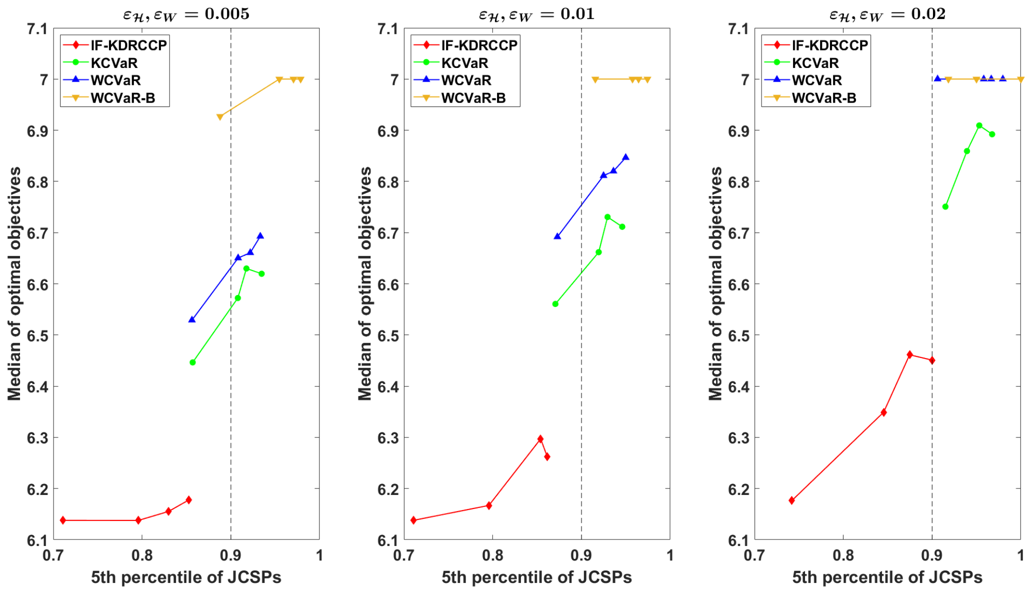


Figure 4.2: Medians of the obtained optimal objectives with respect to the 5th percentiles of the corresponding JCSPs, based on different ambiguity set radius sizes ($\varepsilon_{\mathcal{H}}$ or ε_W). The markers in each line correspond to sample sizes equal to 20, 40, 60, and 80 in order from left to right. The results from the IF-KDRCCP and the KCVaR are based on the Laplacian kernel. The dash line denotes the target JCSP value of 0.9.

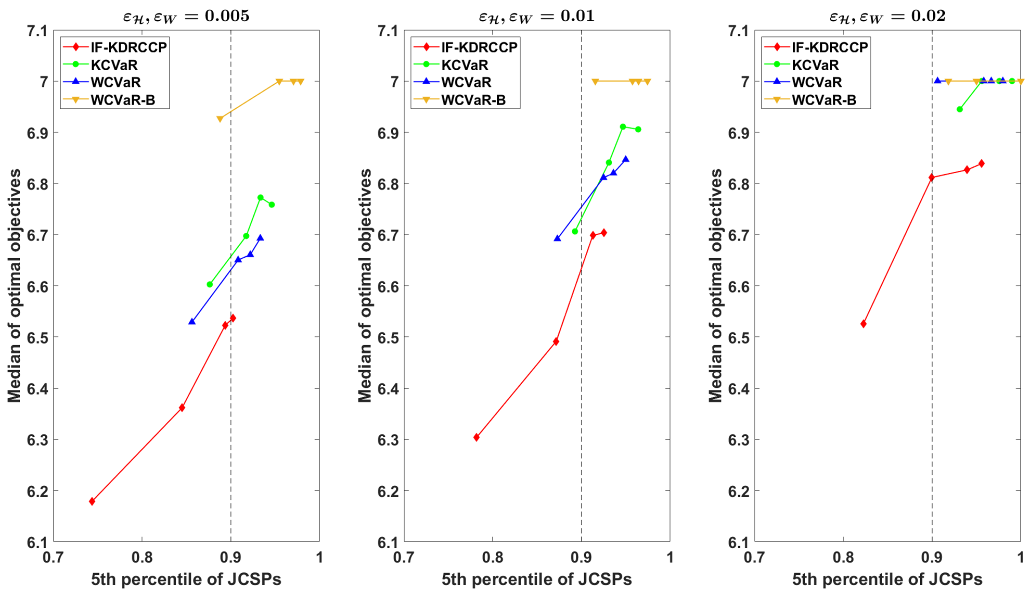


Figure 4.3: Medians of the obtained optimal objectives with respect to the 5th percentiles of the corresponding JCSPs, based on different ambiguity set radius sizes ($\varepsilon_{\mathcal{H}}$ or ε_W). The markers in each line correspond to sample sizes equal to 20, 40, 60, and 80 in order from left to right. The results from the IF-KDRCCP and the KCVaR are based on the rational quadratic kernel. The dash line denotes the target JCSP value of 0.9.

We have the following observations from Figures 4.1-4.3

- The IF-KDRCCP is less conservative than the KCVaR approximation under a fixed ASRS, regardless of the kernel selection.
- Under the same condition, the Laplacian kernel leads to less conservative solution than the other kernels.
- The WCVaR-B is always the more conservative approach. The Bonferroni approximation used in the WCVaR-B is not tight for approximating the original JCC. This leads to overly conservative decisions [3].
- In general, a DRCCP method attains larger objective values as the sample size increases. This attributes to the more extreme scenarios in the larger data set. On the other hand, a larger sample size leads to a reference distribution that is closer to the true distribution, hence the ambiguity set is more inclusive of the true distribution. Therefore, the solution conservativeness of a DRCCP approach might first increase and

then decrease with the increased sample size, as observed from the green lines and the red lines in Figures 4.2-4.3.

- Finally, while the radius of kernel ambiguity set and the Wasserstein ambiguity set is not directly comparable, we can see that the ideal solution is at the bottom right each figure. Hence, we can check the relative position of the curves to compare the kernel DRCCP and Wasserstein DRCCP methods. From this point of view, the IF-KDRCCP dominates other methods.

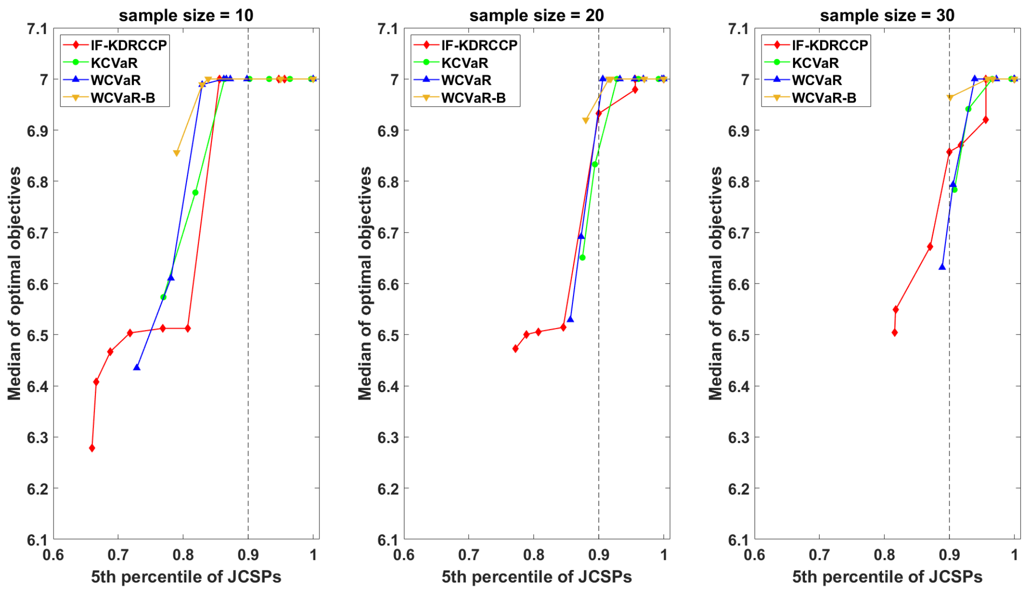


Figure 4.4: Medians of the obtained optimal objectives with respect to the 5th percentiles of the corresponding JCSPs, based on different sample sizes. The markers in each line correspond to ambiguity set radius sizes equal to 0.005, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, and 0.09 in order from left to right. The results from the IF-KDRCCP and the KCVaR are based on the Gaussian kernel. The dash line denotes the target JCSP value of 0.9.

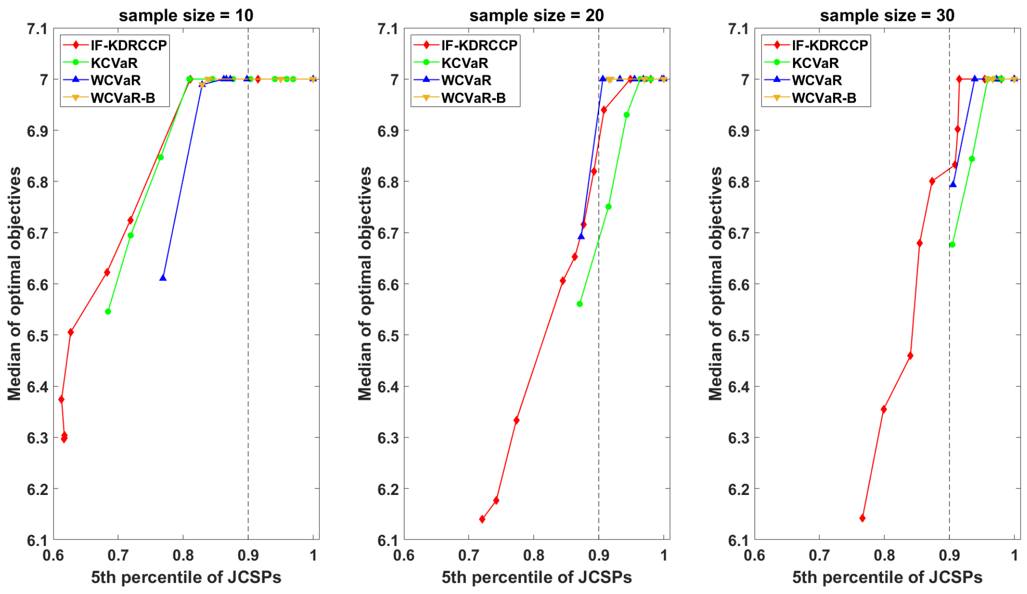


Figure 4.5: Medians of the obtained optimal objectives with respect to the 5th percentiles of the corresponding JCSPs, based on different sample sizes. The markers in each line correspond to ambiguity set radius sizes equal to 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, and 0.1 in order from left to right. The results from the IF-KDRCCP and the KCVaR are based on the Laplacian kernel. The dash line denotes the target JCSP value of 0.9.

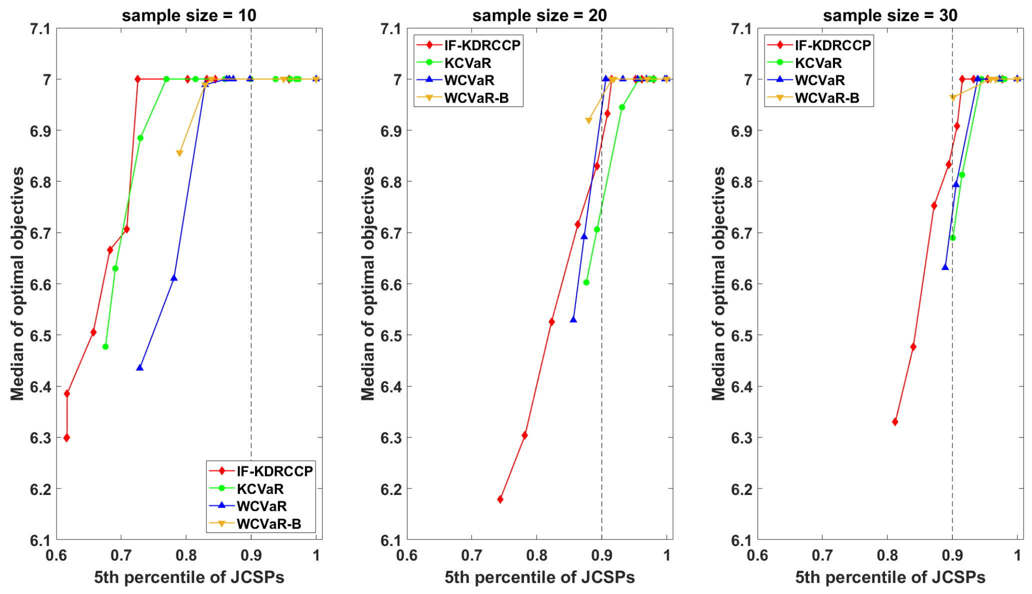


Figure 4.6: Medians of the obtained optimal objectives with respect to the 5th percentiles of the corresponding JCSPs, based on different sample sizes. The markers in each line correspond to ambiguity set radius sizes equal to 0.005, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, and 0.1 in order from left to right. The results from the IF-KDRCCP and the KCVaR are based on the rational quadratic kernel. The dash line denotes the target JCSP value of 0.9.

Figures 4.4-4.6 are used to analyze the performance of the different DRCCP methods based on the same sample size and the different ASRSs. Note that these figures only show the results based on the sample sizes of 10, 20, and 30. While the curves are not separated clearly, in general, the KCVaR and WCVaR demonstrate competitive superior performance. In the meantime, the results from the WCVaR-B generally correspond to most conservative solutions.

4.1.7.2 Tuning of ambiguity set size

In practice, we often face a fixed size of sample set. In such situation, we can tune the ASRSs of the ambiguity set to get least conservative solution that satisfies the JCSP target. Here, we tune the ambiguity set radius of different DRCCP methods to make their 5th percentiles of JCSPs of the obtained optimal solutions to be the target value of 0.9. The corresponding results are shown in Tables 4.1 and 4.2. In these tables, the optimal result and the corresponding tuned ASRS based on a certain sample size and a certain DRCCP approach are obtained through the following steps:

1. Generate 10^4 realizations of $[\xi_1, \xi_2]$ as the validation data set.
2. Take an initial guess of the ASRS. In this numerical example, it is selected from the range of 0.0001 to 0.1.
3. Solve the the numerical example 100 times based on 100 different sample sets with the same sample size.
4. Calculate the JCSPs of the 100 optimal solutions obtained from the previous step, through the SAA based on the validation data set.
5. Find the J-th smallest value of the JCSPs calculated from the previous step (Jth percentile of the JCSPs). This value is denoted as Q^J. J is a user-defined value, and it is set to be 5 in this example.
6. Change the value of the ASRS.
7. Repeat steps $3 \sim 6$ until Q^J is close to $1-\delta$ (Q^J should be $\geq 1-\delta$ and $1-\delta$ is equal to 0.9 in this example). Then, the corresponding value of the ASRS is the tuned ASRS.
8. Based on the tuned ASRS, solve the numerical example 100 times with respect to 100 different sample sets with the same sample size.

Table 4.1: Medians of optimal objectives from different DRCCP methods with respect to different sample sizes.

Sample size	Gaussian		Laplacian		Rational quadratic		WCVaR	WCVaR-B
	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR		
10	7	7	7	7	7	7	7	7
20	6.9324	6.8239	6.8881	***6.6840****	6.8881	6.7420	7	7
30	6.8577	6.7090	6.8326	***6.6354****	6.8577	6.6855	6.8053	6.8321
40	6.7858	6.5318	6.8286	***6.4999****	6.8116	6.5373	6.6504	6.7190

[a] Each value is the median obtained from the 100 optimizations based on 100 different sample sets.

[b] The smallest objective with respect to a fixed sample size is marked by *.

Table 4.2: Tuned ASRSs corresponding to the results shown in Table 4.1.

Sample size	Gaussian		Laplacian		Rational quadratic		WCVaR	WCVaR-B
	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR		
10 samples	0.0630	0.0300	0.1000	0.0690	0.095	0.0610	0.0610	0.0210
20 samples	0.0400	0.0115	0.0744	0.0160	0.0476	0.0120	0.0185	0.0070
30 samples	0.0300	0.0036	0.0600	0.0077	0.0350	0.0048	0.0085	0.0030
40 samples	0.0147	0.0001	0.0483	0.0012	0.0200	0.0009	0.0030	0.0006

According to Table 4.1, with respect to a sample size, the KCVaR based on the Laplacian kernel generally achieves the smaller objectives than based on the other kernels. Furthermore, the KCVaR based on the Laplacian kernel always achieves the smallest objective than the other DRCCP methods with respect to a fixed sample size. In addition, the KCVaR always attains the least conservative objective among all the DRCCP methods with respect to a certain sample size, regardless of the kernel selection. Thus, after tuning the ASRSs, the KCVaR based on any kernel function can outperform the other DRCCP methods. On the other hand, the kernel selection does not have consistent influence to the IF-KDRCCP. Finally, for all the employed DRCCP methods, the obtained objective values decrease with the incremental sample size since more data samples would bring more information to enable a DRCCP method obtaining the solution closer to the true solution. This statement has also been evidenced in [153].

The average solution times of different DRCCP methods are shown in Table 4.3. Based on the different kernels, the KCVaR always has higher computational efficiency than the IF-KDRCCP because the KCVaR model does not contain integer variables which are involved in the IF-KDRCCP. Both WCVaR and WCVaR-B have higher computational efficiency than the kernel-based methods. Moreover, the WCVaR has the highest computational efficiency among all the DRCCP methods.

Table 4.3: Average solution times of different DRCCP methods.

	Gaussian		Laplacian		Rational quadratic		WCVaR	WCVaR-B
	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR		
Average solution time (s)	2.0412	0.7673	1.6595	0.7102	2.0782	0.7624	0.4443	0.5996

[a] For each DRCCP approach, the average solution time is calculated based on 100 optimizations.

[b] Each optimization is based on a set of 40 samples.

Apart from the above discussion, the kernel-based methods have an additional advantage over the Wasserstein-based DRCCPs, which is not shown in the above discussion. Unlike the Wasserstein-based DRCCPs, the kernel-based approaches can be applied to general forms of uncertain constraints. To demonstrate this, the kernel-based methods are applied to a chemical process optimization involving non-linear and non-convex uncertain constraints in the following case study.

4.1.8 Case study

In this section, the proposed kernel-based DRCCP approaches are applied to a chemical process optimization problem under uncertainty. This process optimization problem is taken from [154], and the chemical process is shown in Figure 4.7.

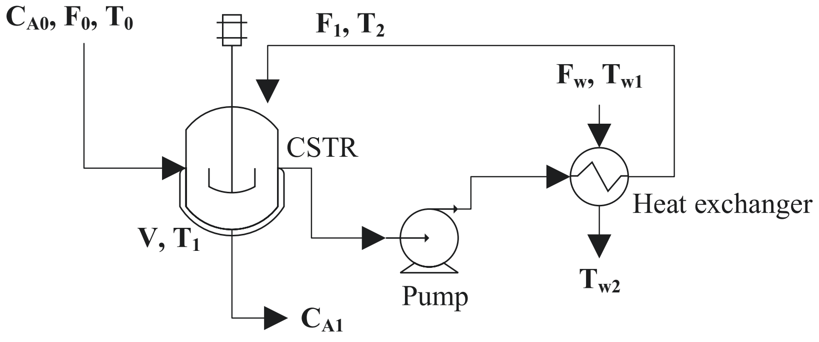


Figure 4.7: Flowsheet of the process studied in the case study

In the process studied, the reaction $A \longrightarrow B$ takes place in the CSTR. One of the outlet streams of the CSTR is pumped to a heat exchanger to be cooled down, and then it is recycled to the CSTR. This chemical process is modeled by the following mathematical formulation:

$$X_A = \frac{Vk_R C_{A0} e^{-\frac{E}{RT_1}}}{F_0 + Vk_R C_{A0} e^{-\frac{E}{RT_1}}} \tag{4.30a}$$

$$Q = \mathcal{A}U \frac{(T_1 - T_{w2}) + (T_2 - T_{w1})}{2} \tag{4.30b}$$

$$T_2 = \frac{2(-\Delta H)F_0 X_A}{\mathcal{A}U} - \frac{2F_0 \rho c_p (T_1 - T_0)}{\mathcal{A}U} - (T_1 - T_{w2}) + T_{w1} \tag{4.30c}$$

$$F_1 = \frac{Q}{\rho c_p (T_1 - T_2)} \tag{4.30d}$$

$$F_w = \frac{Q}{\rho_w c_{pw} (T_{w2} - T_{w1})} \tag{4.30e}$$

C_{A0}, F_0, and T_0 are the concentration of A (kmol/m^3), flowrate (m^3/h), and temperature (K) in the feed flow, respectively. V, T_1, and C_{A1} are the volume (m^3), operating temperature (K), and concentration of A (kmol/m^3) in the CSTR, respectively. F_w, T_{w1}, and T_{w2} are the flowrate (m^3/h), inlet temperature (K), and outlet temperature (K) of the cooling stream, respectively. F_1 and T_2 are the flowrate (m^3/h) and temperature (K) in the recycled flow, respectively. Moreover, X_A, k_R, E, R, Q, $\mathcal{A}$, U, ΔH, ρ, c_p, ρ_w, and c_{pw} are the conversion rate of A in the CSTR, kinetic coefficient of the reaction equation (m^3/(kmol h)), activation energy of the reaction (J/kmol), ideal gas constant (J/(kmol K)), amount of heat (kJ), heat transfer area in the heat exchanger (m^2), overall heat transfer coefficient (kJ/(m^2 h K)), reaction heat (kJ/kmol), density of the recycled stream (kg/m^3), heat capacity of the recycled stream (kJ/(kg K)), density of water (kg/m^3), and heat capacity of water (kJ/(kg K)), respectively. Constant parameters involved in the process model are listed in Table 4.4.

Table 4.4: Deterministic parameters in the process model

Parameter	Value
ρc_p, kJ/(m^3K)	167.4
$\rho_w c_{pw}$, kJ/(m^3K)	4.190
C_{A0}, kmol/m^3	32.04
E/R, K	560
ΔH, kJ/kmol	-23260

In this case study, F_0, T_0, T_{w1}, k_R, and U are uncertain parameters, which follow different

Gaussian distributions with the parameters listed in Table 4.5.

Table 4.5: Means and standard deviations of uncertain parameters

Parameter	Mean	Standard deviation
F_0, m^3/h	45.36	0.1
T_0, K	333	0.02
T_{w1}, K	300	0.03
k_R, m^3/(kmol h)	9.81	0.1
U, kJ/(m^2 h K)	1635	0.1

The studied process optimization is a joint chance-constrained problem given as:

$$\min_{V,\mathcal{A},T_1,T_{w2}} \quad f = 691.2V^{0.7} + 873.6\mathcal{A}^{0.6} + 1.76F_w + 7.056F_1 \tag{4.31a}$$

$$\text{s.t.} \quad \text{Equations } (4.30a) \text{ to } (4.30e)$$

$$Pr\left\{\begin{array}{l} 0.9 - X_A \leq 0 \\ T_2 - T_1 \leq 0 \\ T_{w1} - T_2 + 11.1 \leq 0 \\ T_{w1} - T_{w2} \leq 0 \\ T_2 - 389 \leq 0 \\ 311 - T_2 \leq 0 \end{array}\right\} \geq 1 - \delta \tag{4.31b}$$

$$T_{w2} - T_1 + 11.1 \leq 0 \tag{4.31c}$$

$$311 \leq T_1 \leq 389 \tag{4.31d}$$

$$301 \leq T_{w2} \leq 355 \tag{4.31e}$$

The objective function f represents the overall cost of the studied process, including the capital cost and the operational cost. The unit of the objective function f is \$/year [155]. The violation probability level δ is set to be 0.05 in this case study (the required JCSP is 0.95). For simplification, F_w and F_1 in the objective function correlating with uncertainty are computed based on the nominal scenario (all the uncertain parameters are at their mean values). Notably, the uncertain constraints in 4.31b are nonlinear and non-convex in uncertainty. According to the literature review in Section 4.1.1, the existing Wasserstein-based DRCCP approaches can only be applied to the problems with uncertain constraints affine, quadratic convex, convex, or concave in uncertainty [4, 74, 75]. Therefore, the existing Wasserstein-based DRCCP approaches cannot handle the problem studied in this case study. As a comparison to the presented kernel-based methods, we introduce the SAA-based optimization model for handling the studied process optimization, which is given as:

$$\min_{V,\mathcal{A},T_1,T_{w2}} \quad f = 691.2V^{0.7} + 873.6\mathcal{A}^{0.6} + 1.76F_w + 7.056F_1 \tag{4.32a}$$

$$\text{s.t.} \quad \text{Equations } (4.30a) \text{ to } (4.30e)$$

$$\frac{1}{N}\sum_{j=1}^{N} \mathbb{I}\left(\bigcup_{i=1}^{w} g_i(V,\mathcal{A},T_1,T_{w2},\xi_j) > 0\right) \leq \delta \tag{4.32b}$$

$$T_{w2} - T_1 + 11.1 \leq 0 \tag{4.32c}$$

$$311 \leq T_1 \leq 389 \tag{4.32d}$$

$$301 \leq T_{w2} \leq 355 \tag{4.32e}$$

where N is the number of real samples. w is the number of the uncertain constraints, which is equal to 6 in this case study. $g_i(V,\mathcal{A},T_1,T_{w2},\xi_j)$ denotes an uncertain constraint in (4.31b). ξ is the vector involving uncertain parameters F_0, T_0, T_{w1}, k_R, and U.

The indicator function can be addressed by using the same idea of the mixed-integer model introduced in Section 4.1.4.1. Therefore, the above SAA-based optimization model becomes a mixed-integer nonlinear programming (MINLP) model.

To show the impact of uncertainty on the optimal solution, the studied process optimization is solved deterministically based on the nominal scenario. When solving this deterministic optimization, the joint chance constraint in (4.31b) is decomposed into 6 deterministic constraints. In addition, all variables depending on uncertainty are computed based on the nominal scenario. This deterministic optimization is solved as a nonlinear programming (NLP) problem using KNITRO in GAMS. The obtained solution is: $V = 5.480, \mathcal{A} = 7.199, T_1 = 389, T_{w2} = 355$. The JCSP of this optimal solution is 50.19% which is far away from the required value of 0.95. Therefore, to attain the solution satisfying the constraints simultaneously with high confidence, the uncertainty involved in this process optimization can not be ignored. In this case study, the JCSP of a solution is calculated via the SAA mentioned in Section 4.1.7 based on 10^4 samples of the uncertain parameters, for the testing purpose.

To investigate the general performance of the kernel-based methods and the SAA-based approach in this case study, the studied process optimization handled by a DRCCP method is solved 100 times based on 100 different sample sets. Each set in the 100 sample sets has N samples. The above process based on a DRCCP method is executed with respect to different sample sizes N ($N = 20, 30, 40, 50$ in this case study). The studied process optimization addressed through the IF-KDRCCP, KCVaR, and SAA-based approaches can be solved as MINLP, NLP, and MINLP problems, respectively. All the MINLP and NLP problems in this research are solved by using KNITRO in GAMS.

The 100-time problem-solving with respect to a kernel-based method and a sample size N, is based on the ASRS tuned through the following steps. Note that only one set of collected samples is used in the following steps to make the following tuning method practical.

1. Take an initial guess of the ASRS. The ASRS is selected from the set $\{0.0001, 0.001, 0.01, 0.1, 1\}$.
2. Split the collected samples into $\mathcal{K}$ subsets ($\mathcal{K}$ is set to be 5 in this case study).
3. Select the first subset as the validation set.
4. Merge the remaining $\mathcal{K} - 1$ subsets as the training set.
5. Implement the employed kernel-based DRCCP method on the training set.
6. Calculate the JCSP of the gained optimal solution through the SAA. The attained JCSP is denoted as Q_1.
7. Select the second subset as the validation set and repeat steps $4 \sim 6$ to get Q_2.
8. Repeat the above procedure until $Q_{\mathcal{K}}$ is obtained.
9. Average out $Q_1 \sim Q_{\mathcal{K}}$ to obtain $\bar{Q}$.
10. Change the value of the ASRS.
11. Repeat the above procedure over all the candidate values of the ASRS.
12. The ASRS corresponding to the $\bar{Q}$ which is the closest to $1-\delta$ ($\bar{Q}$ should be $\geq 1-\delta$), is selected for the employed kernel-based DRCCP method to execute the above-mentioned 100-time problem-solving. The tuning process is completed.

In this case study, the sampled support points for the kernel-based methods are generated based on the same idea mentioned in Section 4.1.7. For this case study, 25 sampled support points are generated through the mentioned method. Subsequently, we examine the performance of the kernel-based DRCCP methods based on different sample sizes and the same set of the produced sampled support points. By doing so, we can avoid the impacts of the randomness of the sampled support points selection on the DRCCP performance, to make the following result analysis simpler.

According to the above setting, the obtained optimal results based on the different DRCCP approaches and the different sample sizes are shown in Tables 4.6-4.8. To attain a median value in Table 4.6, 100 optimizations should be completed via a DRCCP approach based on a sample size and the corresponding tuned ASRS shown in Table 4.8. In addition, the 100 optimizations would correspond to 100 JCSPs (reminder: each JCSP is calculated via the SAA based on 10^4 samples for the test purpose in this case study.). The percentage of the JCSPs ≥ 0.95 (the required value in this case study) among the 100 JCSPs (the feasibility percentage) and the minimum JCSP among the 100 JCSPs (MJCSP) are important bases for investigating the reliability of a DRCCP method. These values are shown in Table 4.7, corresponding to the optimal results reported in Table 4.6.

According to Tables 4.6-4.7, the SAA-based method is always the least conservative with the lowest feasibility percentage and MJCSP, with respect to a certain sample size. The SAA-based method is the least conservative and the least reliable approach because it does not consider the distributional ambiguity of uncertainty for enhancing the solution robustness. One remark here is that the median of optimal objectives, the feasibility percentage, and the MJCSP obtained from the SAA-based method increase with the incremental sample size since the larger sample size could contain more extreme data leading to more conservative solutions. As to the kernel-based methods, they can all achieve the feasibility percentages $\geq 95\%$. Thus, we can believe that the kernel-based DRCCP methods are reliable to obtain feasible solutions with high confidence. Based on the similar performance on the solution feasibility and the same sample size, the KCVaR always achieves lower objectives than the IF-KDRCCP, regardless of the kernel selection. On the other hand, with respect to different sample sizes, the kernel selection has no consistent impact on both IF-KDRCCP and KCVaR with the tuned ASRSs. Notably, for both IF-KDRCCP and KCVaR, the objective values decrease with the incremental sample size since a larger data set would bring more information to enable a DRCCP method achieving the solution closer to the true solution.

The average solution times of the different approaches are shown in Table 4.9. The KCVaR is the most computationally efficient among all the methods because the KCVaR model does not include the integer variables which are involved in both IF-KDRCCP and SAA-based method.

Table 4.6: Medians of optimal objectives from different DRCCP methods with respect to different sample sizes.

Sample size	Gaussian		Laplacian		Rational quadratic		SAA
	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	
20	9848.93	9847.83	9851.71	9845.97	9848.66	9845.44	9824.62
30	9848.72	9846.25	9850.17	9844.43	9847.71	9844.16	9829.10
40	9847.82	9845.94	9848.03	9843.33	9847.69	9843.51	9831.41
50	9846.57	9842.38	9846.13	9842.34	9846.63	9842.42	9832.90

Table 4.7: Feasibility percentages and the minimum JCSPs corresponding to the optimal results shown in Table 4.6.

Sample size	Gaussian		Laplacian		Rational quadratic		SAA
	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	
20	96 / 90.04	95 / 88.04	96 / 90.23	95 / 87.74	96 / 89.56	95 / 87.65	23 / 72.39
30	96 / 90.23	95 / 89.05	96 / 90.41	95 / 87.62	96 / 90.01	95 / 87.49	43 / 78.90
40	96 / 90.11	95 / 89.04	96 / 90.17	95 / 87.02	96 / 90.01	95 / 87.34	43 / 79.55
50	96 / 90.14	95 / 88.82	96 / 90.07	95 / 88.63	96 / 90.19	95 / 88.96	50 / 85.13

[a] The percentage values in this table are based on %.

[b] A percentage value in the table is shown as:
feasibility percentage of the 100 optimizations / the minimum JCSP among the 100 JCSPs.

Table 4.8: Tuned ASRSs corresponding to the results shown in Table 4.6.

Sample size	Gaussian		Laplacian		Rational quadratic	
	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR
20	1	0.1	1	0.1	1	0.1
30	1	0.1	1	0.1	1	0.1
40	0.1	0.001	0.1	0.001	0.1	0.001
50	0.001	0.0001	0.001	0.0001	0.001	0.0001

Table 4.9: Average solution times of different methods.

	Gaussian		Laplacian		Rational quadratic		SAA
	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	IF-KDRCCP	KCVaR	
Average solution time (s)	11.6102	0.8737	10.3715	0.8302	11.6749	0.8859	6.0439

[a] For each DRCCP approach, the average solution time is calculated based on 100 optimizations.

[b] Each optimization is based on a set of 50 samples.

4.2 Distributionally Robust Chance-Constrained Optimization with Deep Kernel Ambiguity Set

4.2.1 Introduction

The kernel DRCCP method presented in Section 4.1 can be combined with a neural network-like deep kernel to enhance its performance. The deep kernel is called the multi-layer arc-cosine kernel (MLACK) [156]. The MLACK possesses a deep architecture, which mimics the computation in a multi-layer deep neural network with infinite hidden and output units [157]. Compared to the shallow kernels such as the linear, polynomial, and Gaussian kernels, the MLACK with the deep framework can extract more complex structures and generate more efficient representations by exploiting raw features [158]. Therefore, the MLACK can outperform the shallow kernels for producing representations of distributions in the kernel ambiguity set, which can further enhance the performance of the kernel-based DRCCP approach.

The research of the kernel DRCCP based on the deep kernel is organized as follows. Section 4.2.2 provides a comprehensive explanation of MLACK, while Section 4.2.3 demonstrates the superiority of the deep kernel-based DRCCP over the Gaussian kernel-based DRCCP, by applying both to optimize a nonlinear alkylation process involving uncertainty.

4.2.2 Multi-layer arc-cosine kernel

The multi-layer arc-cosine kernel (MLACK) is the multi-fold composition of arc-cosine kernels (ACKs) [156]. The n-th order ACK (k_n) which is a positive definite kernel, can be formulated as:

$$\begin{aligned}
k_n(x,y) &= \frac{1}{\pi}\,\|x\|^n\,\|y\|^n\,J_n(\theta)\\
J_n(\theta) &= (-1)^n(\sin\theta)^{2n+1}\left(\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\right)^n\left(\frac{\pi-\theta}{\sin\theta}\right)\\
\theta &= \cos^{-1}\left(\frac{x\cdot y}{\|x\|\,\|y\|}\right)
\end{aligned}\tag{4.33}$$

For simplification, we only consider $n = 0, 1$ in this study:

$$\begin{aligned}
J_0(\theta) &= \pi-\theta\\
J_1(\theta) &= \sin\theta+(\pi-\theta)\cos\theta
\end{aligned}$$

ACKs have the following properties: 1) while $n = 0$, the ACK maps inputs to a unit hypersphere in a feature space, with $k_0(x,x) = 1$; 2) while $n = 1$, the ACK preserves the norm of inputs, with $k_1(x,x) = \|x\|^2$; 3) while $n > 1$, ACKs expand the dynamic range of inputs, with $k_n(x,x) \sim \|x\|^{2n}$ [156].

Furthermore, ACKs have the recursive property that makes the multi-fold composition of ACKs a more powerful new kernel [159]. The multi-fold composition of ACKs can be written as:

$$\begin{aligned}
k_n^{\ell+1}(x,y) &= \frac{1}{\pi}\left[k_n^{\ell}(x,x)k_n^{\ell}(y,y)\right]^{n/2}J_n\left(\theta_n^{\ell}\right)\\
\theta_n^{\ell} &= \cos^{-1}\left(k_n^{\ell}(x,y)\left[k_n^{\ell}(x,x)k_n^{\ell}(y,y)\right]^{-1/2}\right)
\end{aligned}\tag{4.34}$$

The above expression is the formulation of the MLACK. ℓ is the layer index. The base case k_n^1 is calculated using (4.33). Notably, n can be different with respect to different ℓ. The MLACK computes the inner product between the output vectors produced from a multi-layer neural network. This neural network has infinite neurons in hidden and output layers [160]. A schematic diagram for illustrating the concept of the MLACK is shown in Figure 4.8. $\Phi(x)$ and $\Phi(y)$ are respectively nonlinear transforms of inputs x and y through the multi-layer neural network. In this neural network, there are $\ell + 1$ layers including the hidden and output layers. The $\ell + 1$-th layer is the output layer. The order n at each layer determines the activation function. For $n = 0$ and $n = 1$, the associated activation functions are the step function and the rectified linear unit (ReLU) function, respectively [161]. Notably, the kernel function used for the kernel ambiguity set is to compute the similarity between two

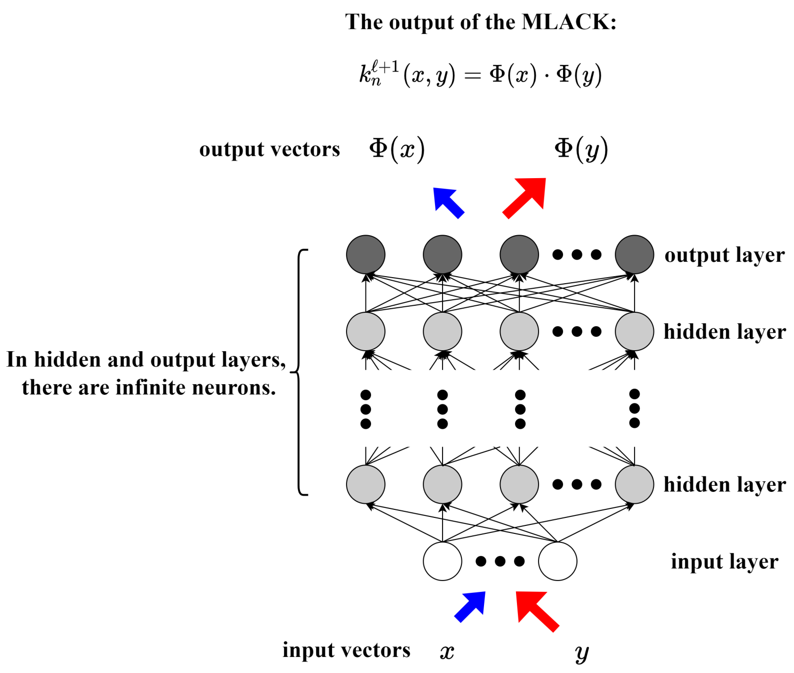


Figure 4.8: Schematic diagram for illustrating the concept of the multi-layer arc-cosine kernel (MLACK)

data points, and the similarity value is generally defined in $[0, 1]$. Therefore, the order n at the output layer of the MLACK is set to be 0 in this work to ensure the MLACK output is in $[0, 1]$. On the other hand, based on our previous computational experiments, setting orders at hidden layers of the MLACK > 1 does not improve or even worsen the performance of the kernel DRCCP. Accordingly, we only consider $n = 0, 1$ for all the layers of the MLACK in this work.

The multi-fold compositions of linear, polynomial, and Gaussian kernels cannot generate more powerful new kernels since these kernels are shallow kernels without the capability of bringing deep architectures [158]. On the contrary, the performance of the MLACK could be improved by increasing the number of layers of the kernel composition [159]. A deep MLACK could capture more complex structures of uncertainty distributions and produce more efficient representations of uncertainty distributions in the RKHS, compared to the shallow kernels. Therefore, the performance of the DRCCP under the kernel ambiguity set could be further enhanced by employing the MLACK.

According to the investigation on kernel DRCCP presented in Section 4.1, the CVaR-based kernel DRCCP (KCVaR) model in (4.24) exhibits significantly better computational efficiency than the indicator function-based kernel DRCCP (IF-KDRCCP) model in (4.17). Hence, in this research work, we have integrated the KCVaR model with the MLACK, which is a simple process. The integration of KCVaR with MLACK involves the computation of Gram matrices K_η, $K_{\eta\xi}$, and K_ξ using MLACK, followed by the incorporation of the calculated Gram matrices into the KCVaR model in (4.24). We can use the same procedure for combining the IF-KDRCCP model with the MLACK.

4.2.3 Case study

In this section, the KCVaR models based on the MLACK (KCVaR-M) and the Gaussian kernel (KCVaR-G) are applied to the alkylation process optimization under uncertainty. This case study is modified from Example 14.3 in [162] by adding random parameters to the original process model. The simplified flowsheet of the alkylation process is shown in Figure 4.9. The reactor feeds contain an olefin stream (100% butane), a pure isobutane recycle stream, a 100% isobutane make-up stream, and an acid catalyst. The spent acid is removed from the reactor. The product stream from the reactor is separated into isobutane and alkylate product using a fractionator.

The objective of the studied optimization is to maximize the total profit based on the alkylate product value (\$0.063/octane-barrel), olefin feed cost (\$5.04/barrel), isobutane recycle cost (\$0.035/barrel), acid addition cost (\$10.00/per thousand pounds), and isobutane makeup cost (\$3.36/barrel).

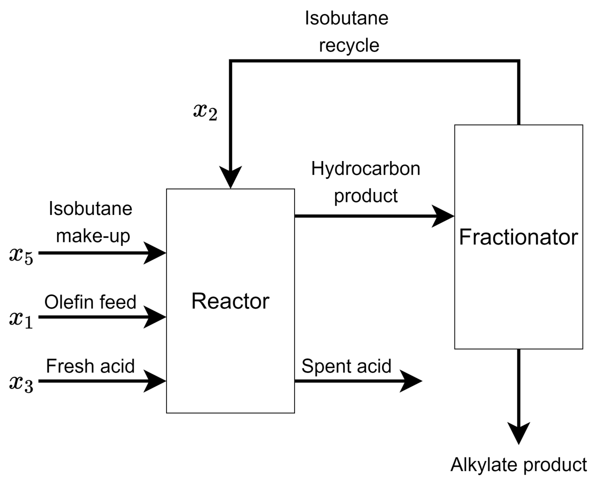


Figure 4.9: Flowsheet of the studied alkylation process

The studied optimization problem is formulated as (4.35a)-(4.35l). $x_1 \sim x_{10}$ are decision variables as explained in Table 4.10. Each of them is associated with some physical bounds. The volumetric balance for the reactor is defined in (4.35g). The alkylate yield x_4 can be expressed as the combination of the olefin feed x_1 and the isobutane makeup x_5, minus the volumetric shrinkage, which is 0.22 of the alkylate yield. Thus, we have $x_4 = x_1 + x_5 - 0.22x_4$, which can be rearranged to obtain $x_5 = 1.22x_4 - x_1$. To calculate the acid strength x_6, we use (4.35h), which assumes that the added acid (with an acid addition rate of x_3) has an acid strength of 98%. The external isobutane-to-olefin ratio x_8 is determined by dividing the sum of the isobutane recycle x_2 and the isobutane makeup x_5 by the olefin feed x_1, as given in (4.35i).

Table 4.10: Variables of the optimization model

Symbol	Variable	Unit
x_1	Olefin feed	barrels/day
x_2	Isobutane recycle	barrels/day
x_3	Acid addition rate	thousands of pounds/day
x_4	Alkylate yield	barrels/day
x_5	Isobutane makeup	barrels/day
x_6	Acid strength	weight percent
x_7	Motor octane number	–
x_8	External isobutane-to-olefin ratio	–
x_9	Acid dilution factor	–
x_{10}	F-4 performance number	–

$$\max_{x} \quad 0.063x_4x_7 - 5.04x_1 - 0.035x_2 - 10x_3 - 3.36x_5 \tag{4.35a}$$

$$\text{s.t.} \quad Pr\left\{\begin{array}{l} x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2) - \dfrac{9}{10}x_4 \geq 0 \\ -x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2) + \dfrac{10}{9}x_4 \geq 0 \\ -\xi_4 + \xi_5 x_7 - \dfrac{9}{10}x_{10} \geq 0 \\ \xi_4 - \xi_5 x_7 + \dfrac{10}{9}x_{10} \geq 0 \end{array}\right\} \geq 0.9 \tag{4.35b}$$

$$86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_6 - 89) - \frac{9}{10}x_7 \geq 0 \tag{4.35c}$$

$$-86.35 - 1.098x_8 + 0.038x_8^2 - 0.325(x_6 - 89) + \frac{10}{9}x_7 \geq 0 \tag{4.35d}$$

$$35.82 - 0.222x_{10} - \frac{9}{10}x_9 \geq 0 \tag{4.35e}$$

$$-35.82 + 0.222x_{10} + \frac{10}{9}x_9 \geq 0 \tag{4.35f}$$

$$x_5 = 1.22x_4 - x_1 \tag{4.35g}$$

$$x_6 = \frac{98000x_3}{x_4x_9 + 1000x_3} \tag{4.35h}$$

$$x_8 = \frac{x_2 + x_5}{x_1} \tag{4.35i}$$

$$0 \leq x_1 \leq 2000, \;\; 0 \leq x_2 \leq 16000, \;\; 0 \leq x_3 \leq 120, \;\; 0 \leq x_4 \leq 5000 \tag{4.35j}$$

$$0 \leq x_5 \leq 2000, \;\; 85 \leq x_6 \leq 93, \;\; 90 \leq x_7 \leq 95, \;\; 3 \leq x_8 \leq 12 \tag{4.35k}$$

$$0.01 \leq x_9 \leq 4, \;\; 145 \leq x_{10} \leq 162 \tag{4.35l}$$

The term $x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2)$ in the first two constraints of the joint chance constraint (4.35b), is the regression model for computing x_4 based on x_1 and x_8. The term $-\xi_4 + \xi_5 x_7$ in the last two constraints of the joint chance constraint, is the regression model for computing x_{10} based on x_7. The inequalities in the joint chance constraint limit the percentage errors between the regression model outputs and the true values to the range $0.9 \sim 1.11$. $\xi_1 \sim \xi_5$ are uncertain parameters in the regression models. They either follow normal distributions $\xi_1 \sim N(1.12, 0.028^2)$, $\xi_4 \sim N(133, 3.325^2)$, or uniform distributions $\xi_2 \sim U(0.1277, 0.1356)$, $\xi_3 \sim U(0.0065, 0.0069)$, $\xi_5 \sim U(2.91, 3.09)$. Inequalities (4.35c)-(4.35f) involve the regression model $86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_6 - 89)$ for x_7, and the regression model $35.82 - 0.222x_{10}$ for x_9. This optimization problem is a nonlinear joint chance-constrained optimization. All the settings for the studied process optimization problem are adopted from [162], except for the percentage error tolerances of the regression models and the uncertain parameters.

The studied problem is handled by the KCVaR-G and KCVaR-M. The problem addressed by the KCVaR-G or KCVaR-M is solved 100 times based on 100 different N-sample sets, with the hyper-parameters tuned through Algorithm 1. The above 100-time problem-solving procedure is similar to the 100-time problem-solving procedure mentioned in Section 4.1.7. A sample supported point for $[\xi_1, \xi_2, \xi_3, \xi_4, \xi_5]$ is denoted as $\left[\xi_1', \xi_2', \xi_3', \xi_4', \xi_5'\right]$, respectively. ξ_1', ξ_2', ξ_3', ξ_4', and ξ_5' are evenly selected from the ranges of 1.0486 to 1.1914, 0.1277 to 0.1356, 0.0065 to 0.0069, 124.5212 to 141.4787, and 2.91 to 3.09, respectively. This selection method of sampled support points can approximately cover the uncertainty domain.

Algorithm 1 Tuning the hyper-parameters for DRCCP

```
1:  Input: $\varepsilon_{\mathcal{H}} \in [0.0001, 0.1]$ : kernel ambiguity set radius size
           $Y \in [10, 20, 30, 40, 50]$ : # of sampled support points
           $L \in \mathbb{Z}_{>0}$ : # of layers in the MLACK
           $n_\ell \in \{0, 1\}$ : the order in each layer of the MLACK
           $tol$ : a positive user-defined tolerance
2:  Data: 100 different $N$-sample sets of $[\xi_1, \xi_2, \xi_3, \xi_4, \xi_5]$,
          a $10^4$-sample set of $[\xi_1, \xi_2, \xi_3, \xi_4, \xi_5]$ as the validation set $\mathcal{S}_V$
3:  Take an initial guess of the hyper-parameters.
4:  while $Q^J - (1-\delta) \geq tol \vee Q^J \leq (1-\delta)$ do
5:      Change the values of hyper-parameters if the iteration # $\geq 2$.
6:      Solve the optimization problem by using a DRCCP method 100 times based on 100
        different $N$-sample sets.
7:      Calculate the JCSPs of the 100 optimal solutions obtained from the previous step,
        through the sample average approximation [14] based on $\mathcal{S}_V$.
8:      Find the $J$-th largest value of the JCSPs from the previous step. This value is denoted
        as $Q^J$.
9:  end while
10: Output: The tuned hyper-parameters
```

The obtained optimal results and the corresponding tuned hyper-parameters are shown in Tables 4.11-4.12, respectively.

Table 4.11: Means and standard deviations of optimal objectives from different methods with respect to different sample sizes

Sample size N	KCVaR-G	KCVaR-M
	mean / standard deviation	
20	2324.37 / 95.90	2350.16 / 136.84
30	2325.99 / 117.23	2352.01 / 125.63
40	2326.46 / 129.05	2353.84 / 125.51
50	2331.92 / 118.18	2355.18 / 121.83

Table 4.12: Tuned hyper-parameters

Sample size N	KCVaR-G	KCVaR-M
	$\varepsilon_{\mathcal{H}}$ / Y / $[n_1, ..., n_\ell]$	
20	0.0001 / 20 / -	0.0009 / 20 / [0 1 1 1 1 0]
30	0.0001 / 20 / -	0.0006 / 20 / [0 1 1 1 1 0]
40	0.0001 / 20 / -	0.0007 / 20 / [0 1 1 1 1 0]
50	0.0001 / 20 / -	0.0002 / 20 / [0 1 1 1 1 0]

[a] $[n_1, ..., n_\ell]$ are the orders of 1st $\sim$ ℓth MLACK layers, respectively.

According to Table 4.11, the KCVaR-M can always achieve better solutions than the KCVaR-G (this case study is a maximization problem where a larger objective is deemed more desirable). The results indicate that the performance of the kernel DRCCP approach is better when using MLACK compared to the traditional Gaussian kernel. This can be explained by the fact that MLACK with its deep architecture can capture more intricate structures of uncertainty distributions and create more effective representations of uncertainty distributions in the RKHS, compared to the shallow Gaussian kernel.

4.3 Conclusions

The DRCCP based on the kernel ambiguity set is proposed in Section 4.1. The kernel ambiguity set is established via the kernel mean embedding and MMD between distributions.

The presented kernel-based DRCCP can be formulated as two different models. The first one is a mixed-integer model involving the indicator function for addressing the JCSP in the JCC, namely the IF-KDRCCP. The second one is more computationally efficient, which is a continuous model employing the CVaR approximation to approximate the indicator function, namely the KCVaR.

The advantages of the kernel ambiguity set-based DRCCP have been demonstrated through examples. After tuning the ASRSs, the KCVaR can outperform the IF-KDRCCP, in the aspects of both solution quality and computational efficiency. Moreover, for dealing with larger nonlinear optimization problems, the KCVaR may be more suitable than the IF-KDRCCP since it would be very time-consuming to solve larger nonlinear problems by using the mixed-integer IF-KDRCCP. The presented kernel-based DRCCP approaches are applicable to general forms of uncertain constraints without any assumption on the constraints. This is a remarkable advantage over the existing DRCCP methods which require non-trivial assumptions for different forms of uncertain constraints.

The kernel DRCCP presented above is further integrated with a deep kernel, the MLACK, to enhance its performance. The MLACK has a deep neural network-like architecture. The MLACK is able to capture more complex structures and generate more efficient representations by exploiting raw features, and it can produce better representations of distributions in the kernel ambiguity set, outperforming shallow kernels such as the linear, polynomial, and Gaussian kernels. According to the results of the case study, the kernel DRCCP approach based on the MLACK can achieve better optimal solutions than the kernel DRCCP based on the traditional Gaussian kernel.

As to future work, the more general support discretization method for the presented kernel-based DRCCP should be studied. Also, a more detailed study on the specific application scenarios of different kernel DRCCP models could be a meaningful research direction. In addition, it is worth to investigate more efficient tuning steps of the kernel ambiguity set radius. Moreover, the proposed approach can be extended to multi-stage adaptive optimization problems.

Chapter 5

Distributionally Robust Chance-Constrained Optimization with Sinkhorn Ambiguity Set

Abstract

A novel distributionally robust chance-constrained optimization (DRCCP) method is proposed in this work based on the Sinkhorn ambiguity set. The Sinkhorn ambiguity set is constructed based on the Sinkhorn distance, which is a variant of the Wasserstein distance with the entropic regularization. The proposed method can hedge against more general families of uncertainty distributions than the Wasserstein ambiguity set-based methods. The presented approach is formulated as a tractable conic model based on the Conditional value-at-risk (CVaR) approximation and the discretized kernel distribution relaxation. This model is compatible with more general uncertain constraints than the Wasserstein-based methods. Accordingly, the presented Sinkhorn DRCCP is a more practical approach that overcomes the limitations of the traditional Wasserstein DRCCP approaches. A numerical example and a nonlinear chemical process optimization case are studied to demonstrate the efficacy of the Sinkhorn DRCCP and its advantages over the Wasserstein DRCCP.

5.1 Introduction

Optimization under uncertainty is crucial for process design, control, planning, and scheduling because in the real world, systems are subject to various uncertain factors such as material availability, market demand, equipment failure, and measurement errors. These uncertainties can have a significant impact on the system's performance, making it challenging to design an optimal solution that meets the desired criteria and constraints. Chance-constrained programming (CCP) is a widely-used method [10] to tackle uncertainty. The CCP enforces the optimal decision for an optimization problem to be feasible with a user-defined confidence level. There are two types of CCP: the individual CCP (ICCP) and the joint CCP (JCCP) [11]. In ICCP, a constraint satisfaction probability is enforced for each uncertain constraint [163]. The JCCP is more general than the ICCP in the sense that it ensures multiple constraints are satisfied jointly to a certain probability [12]. However, a JCCP problem is difficult to solve as it requires dealing with the multidimensional integration [13]. Therefore, JCCP problems are usually solved through approximation methods including analytical approximation methods and sampling-based methods, etc [28].

In practical situations, true distributions of uncertainty are usually hard to be obtained. Thus, solving the JCCP problem based on the worst-case distribution among all potential data-generating distributions is a more robust and practical way. This is essentially the idea of the distributionally robust chance-constrained programming (DRCCP) [15]. The DRCCP aims to optimize the objective with the worst-case joint constraint satisfaction probability above the acceptable level over a set of candidate distributions. Such a set is called the ambiguity set [133]. The ambiguity set might involve certain distributional information of the unknown true distribution (e.g., moments, structure properties, domain knowledge, etc) from collected data. The ambiguity set should be large enough to contain the true distribution with high confidence and a well-designed set can also exclude irrelevant distributions which may cause overly conservative decisions [134]. Moreover, a well-designed ambiguity set should enable the DRCCP problem to be formulated as a tractable mathematical program, which can be solved using standard solvers [135].

In the field of the DRCCP, moment-based ambiguity sets and metric-based ambiguity sets are widely-used [57–59]. A moment-based ambiguity set is composed of all distributions satisfying certain moment constraints [133, 136, 137]. To establish a moment-based ambiguity set, a certain level of moment information should be known beforehand. Additionally, because different distributions might share the same moments, a moment-based ambiguity set might not be adequate to exclude irrelevant distributions that can result in overly conservative solutions [60]. Furthermore, moment-based ambiguity sets are not capable of closely approximating original chance constraints even if sufficient data is available [61]. Metric-based ambiguity sets offer a stronger solution to address the drawbacks mentioned above. In the context of metric-based ambiguity sets, these sets can be conceptualized as balls in the probability distribution space. In particular, all potential distributions in a metric-based ambiguity set are centered around the nominal distribution, which is established based on collected data, and the radius of this set is determined by a probability metric. The size of the radius is a user-defined hyperparameter that can be adjusted to control the level of conservatism in the resulting solution. The ϕ-divergence is a widely-used probability metric for metric-based ambiguity sets [62–65]. Nonetheless, as pointed out by some existing literatures [16, 66], a ϕ-divergence ambiguity set only includes distributions absolutely continuous with respect to the nominal distribution, and thus the ϕ-divergence ambiguity set only contains distributions with the same support as the nominal distribution. This raises the issue that the true distribution may not be included in the ambiguity set if the true and nominal distributions have different supports. Moreover, according to Gao and Kleywegt [66], as the ϕ-divergence only considers the relative ratio between two distributions, it cannot capture the distance between them. This limitation may result in the inclusion of irrelevant distributions in the ϕ-divergence ambiguity set, leading to excessively conservative decision-making. To address the aforementioned issues in constructing metric-based ambiguity sets, the Wasserstein distance has become a widely used alternative [67–69]. This distance is defined as the optimal transport distance obtained from the optimal transport problem between the two distributions. The ambiguity set constructed using the Wasserstein distance (Wasserstein ambiguity set) contains both continuous and discrete distributions, and thus it may include the true distribution with high confidence. Moreover, Wasserstein ambiguity sets are more efficient in excluding irrelevant distributions and avoiding overly conservative decisions than the ϕ-divergence ambiguity sets [66] since the Wasserstein distance has the capability of measuring the distance between two distributions. However, the existing studies rely on some nontrivial assumptions on uncertain constraints to attain tractable Wasserstein DRCCP models. In most existing works of Wasserstein DRCCP [3, 4, 61, 70–73], constraints involving uncertainty are assumed to be affine in uncertainty. In Reference [74], uncertain constraints are restricted to be quadratic convex in uncertainty. In Reference [75], uncertain constraints are limited to be either concave or convex in uncertainty.

The Wasserstein ambiguity set's worst-case distribution is typically constrained to a discrete distribution supported by no more than $M+1$ samples, where M represents the number of real samples collected [66, 76]. This limitation arises because the Wasserstein distance is calculated at a vertex of the polyhedral set of transportation plans [164], resulting in an extreme and sparse transportation plan. As a result, the worst-case distribution of the Wasserstein ambiguity set is based on this sparse optimal transportation plan, leading to a sparse and extreme discrete distribution. If the true distribution is continuous, this limitation could cause the Wasserstein DRCCP to incorrectly hedge against the wrong distribution family, leading to decisions that deviate significantly from the true optimal decision.

To overcome the above limitations of the Wasserstein DRCCP, a novel DRCCP approach based on the Sinkhorn distance [164] is proposed in this work. The Sinkhorn distance is a variant of the Wasserstein distance with the entropic regularization [164]. The entropic regularization prevents the Sinkhorn distance from reaching an extreme and sparse transportation plan. In other words, the entropic regularization smooths out the transportation plan. With the smoother transportation plan, the ambiguity set based on the Sinkhorn distance (Sinkhorn ambiguity set) results in a smoother worst-case distribution that is not limited to a discrete distribution. Accordingly, the proposed DRCCP method based on the Sinkhorn ambiguity set (Sinkhorn DRCCP) is based on a more general worst-case distribution than the Wasserstein DRCCP. The presented approach is formulated as a tractable conic optimization model based on the Conditional value-at-risk (CVaR) approximation[139] and the discretized kernel distribution relaxation [17]. This model does not require any assumptions on uncertain constraints. Based on the above advantages, the Sinkhorn DRCCP presented in this work is more practical for real-world problems than the traditional Wasserstein DRCCP. Regarding the existing studies, the Sinkhorn ambiguity set has only been employed for distributionally robust optimization (DRO) containing uncertainty only in the objective function [17]. To the best of the author's knowledge, the research of the DRCCP containing uncertainty in constraints based on the Sinkhorn ambiguity set has not been found so far.

This work is structured as follows. The background knowledge of the distributionally robust optimization (DRO) including DRCCP is introduced in Section 5.2. To demonstrate one of the advantages of the Sinkhorn DRCCP over the Wasserstein DRCCP, the comparisons between the worst-case distributions from the Sinkhorn and Wasserstein ambiguity sets are presented in Section 5.3. The derivation for the tractable model of the Sinkhorn DRCCP is presented in Section 5.4. In Section 5.5, a numerical example is studied. Subsequently, the application of the proposed approach to a nonlinear process optimization under uncertainty is demonstrated in Section 5.6. Finally, this study is summarized in Section 5.7.

5.2 Distributionally robust optimization

In distributionally robust optimization (DRO), we optimize the worst-case expected value of the objective function over the ambiguity set. The general formulation of the DRO problem is given as:

$$\min_{x\in\mathcal{X}} \max_{\mathbb{P}\in\mathcal{P}(\Xi)} \quad \mathbb{E}_{\mathbb{P}}\left[f(x,\eta)\right] \tag{5.1}$$

where x is the decision variable with the feasible set $\mathcal{X}$ and η is a random parameter vector. $\mathcal{P}(\Xi)$ is the ambiguity set defined with metric $\mathcal{D}$ over support Ξ:

$$\mathcal{P}(\Xi) = \{\mathbb{P} : \mathcal{D}(\mathbb{P},\mathbb{P}_0) \leq \varepsilon\} \tag{5.2}$$

The ambiguity set $\mathcal{P}(\Xi)$ contains a family of candidate probability distributions $\mathbb{P}$ supported on Ξ. A candidate distribution $\mathbb{P}$ has a certain degree of similarity to the nominal distribution $\mathbb{P}_0$. In most practical settings, the nominal distribution $\mathbb{P}_0$ is set to be an empirical distribution $\hat{\mathbb{P}}_0$ established by samples $\{\xi_m\}_{m=1}^M$ (M is the number of collected samples). The (dis)similarity between $\mathbb{P}$ and $\mathbb{P}_0$ is determined by the distance metric $\mathcal{D}$ between $\mathbb{P}$ and $\mathbb{P}_0$. This distance is restricted to be less than or equal to a user-defined parameter ε to exclude irrelevant distributions that may cause overly conservative solutions. Problem (5.1) is to find the optimal x that minimizes the worst-case (maximum) expectation of the objective over $\mathbb{P}$ in $\mathcal{P}(\Xi)$.

The distributionally robust chance-constrained optimization (DRCCP) enforces the worst-case probability of constraint violation below a user-defined tolerance. The general formulation of a DRCCP is given as:

$$\min_{x\in\mathcal{X}} \quad f(x) \tag{5.3a}$$

$$\text{s.t.} \quad \max_{\mathbb{P}\in\mathcal{P}(\Xi)} Pr\left(\bigcup_{i=1}^{w} g_i(x,\eta) > 0\right) \leq \delta \tag{5.3b}$$

The inequality in (5.3b) is the distributionally robust joint chance constraint (DRJCC) enabling the worst-case probability of violating any constraints $g_i(x,\eta) \leq 0$ less or equal to a user-defined tolerance δ.

5.2.1 Wasserstein ambiguity set

The Wasserstein distance is a popular metric for constructing the ambiguity set (Wasserstein ambiguity set). The detailed explanation of the Wasserstein distance is in Section 1.2.5. The Wasserstein ambiguity set based on the type-1 Wasserstein distance $\mathcal{W}_1$, is defined as:

$$\mathcal{P}_{\varepsilon_w} = \{\mathbb{P} : \mathcal{W}_1(\mathbb{P}, \mathbb{P}_0) \leq \varepsilon_w\} \tag{5.4}$$

where ε_w is a user-defined parameter restricting the size of the Wasserstein ambiguity set $\mathcal{P}_{\varepsilon_w}$. While the nominal distribution $\mathbb{P}_0$ is based on the collected data samples $\{\xi_m\}_{m=1}^{M}$, it is proven that the Wasserstein DRO results in a discrete worst-case distribution supported on at most $M+1$ points, as shown in Section 5 of Reference [76] through the Richter-Rogosinski theorem. If the true distribution of uncertainty is a continuous distribution, the Wasserstein DRO cannot hedge against the correct family of distributions. This issue is mainly caused by the reason that the Wasserstein distance (an optimal transport problem) is always solved at a vertex of the polyhedral set of transportation plans [164]. An intuitive visualization of this statement is shown in Figure 5.1. The optimal transportation plan from the Wasserstein distance π_W^* is at the vertex of $\Pi(\mathbb{P}, \mathbb{P}_0)$ (the black polytope) that results in an extreme and sparse transportation plan. On the other hand, the optimal transportation plan from the Sinkhorn distance π_S^* is not restricted to be at the vertices since the Sinkhorn distance is a nonlinear programming (NLP) problem due to the entropic regularization, as shown in the next section.

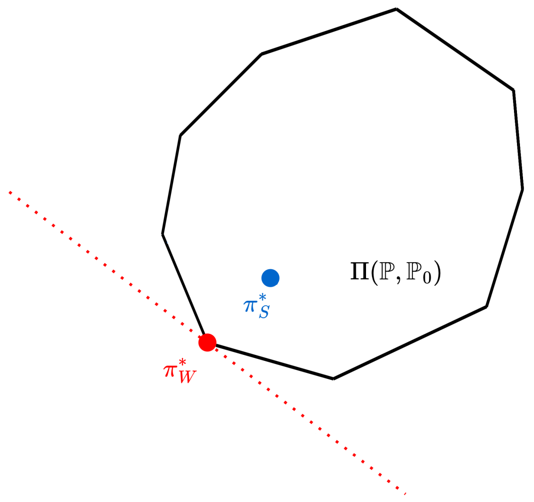


Figure 5.1: Illustration of optimal transport plan from Wasserstein distance and Sinkhorn distance

5.2.2 Sinkhorn ambiguity set

To overcome the above-mentioned limitation of the Wasserstein DRO, the Sinkhorn DRO based on the Sinkhorn ambiguity set is proposed [17]. The Sinkhorn ambiguity set is constructed using the Sinkhorn distance [164], which is a variant of the Wasserstein distance with the entropic regularization. The Sinkhorn distance between $\mathbb{P}$ and $\mathbb{P}_0$ supported on Ξ can be expressed as:

$$\mathcal{W}_{\gamma_s}(\mathbb{P}, \mathbb{P}_0) = \inf_{\pi \in \Pi(\mathbb{P}, \mathbb{P}_0)} \mathbb{E}_{(\eta,\xi)\sim\pi} \{[c(\eta, \xi)] + \gamma_s H(\pi | A \otimes B)\} \tag{5.5}$$

where $\gamma_s \geq 0$ is the regularization parameter. A and B are two reference measures such that $\mathbb{P}$ and $\mathbb{P}_0$ are absolutely continuous with respect to them, respectively. A special case widely studied in the literature is the choice with $A = \mathbb{P}$ and $B = \mathbb{P}_0$. $H(|A \otimes B)$ is the relative entropy of the transport plan π with respect to the product measure $A \otimes B$ (characterized by $(A \otimes B)(X \times Y) = A(X)B(Y)$ for any pair of Borel sets X, Y):

$$H(\pi | A \otimes B) = \int \log \left(\frac{d\pi(\eta, \xi)}{dA(\eta) dB(\xi)} \right) d\pi(\eta, \xi)$$

Note that all notations log in this chapter represent the natural logarithm. According to (1.8) and (5.5), the only difference between $\mathcal{W}_1$ and $\mathcal{W}_{\gamma_s}$ is the entropic regularization term

$\gamma_s H(\pi|\pi|\mathbb{P} \otimes \mathbb{P}_0)$. The entropic regularization smooths out the transportation plan and prevents the Sinkhorn distance from reaching an extreme and sparse transportation plan as shown in Figure 5.1. Based on the smooth transportation plan from the Sinkhorn distance, the Sinkhorn ambiguity set results in a smooth worst-case distribution that is not restricted to be discrete. Accordingly, the Sinkhorn DRO can hedge against more general families of uncertainty distributions than the Wasserstein DRO.

$\mathcal{W}_{\gamma_s}$ between two discrete empirical distributions $\hat{\mathbb{P}}$ and $\hat{\mathbb{P}}_0$ with reference measure $A = \hat{\mathbb{P}}$ and $B = \hat{\mathbb{P}}_0$ can be equivalently formulated as the following regularized optimal transport problem [165]:

$$\begin{aligned} \mathcal{W}_{\gamma_s} = \min_{\pi \geq 0} \quad & \sum_{n=1}^{N}\sum_{m=1}^{M} \|\eta_n - \xi_m\|\pi_{nm} + \gamma_s \pi_{nm} \log(\pi_{nm}) \\ \text{s.t.} & \sum_{m=1}^{M} \pi_{nm} = \frac{1}{N}, \forall n = 1, ..., N \\ & \sum_{n=1}^{N} \pi_{nm} = \frac{1}{M}, \forall m = 1, ..., M \end{aligned} \tag{5.6}$$

The Sinkhorn ambiguity set $\mathcal{P}_{\gamma_s,\varepsilon_s}(\Xi)$ is defined as

$$\mathcal{P}_{\gamma_s,\varepsilon_s} = \{\mathbb{P} : \mathcal{W}_{\gamma_s}(\mathbb{P}, \mathbb{P}_0) \leq \varepsilon_s\} \tag{5.7}$$

where ε_s is the size of the Sinkhorn ambiguity set and γ_s is the regularization parameter.

5.3 Worst-case distribution

In this section, we use a toy example to demonstrate the difference between the worst-case distributions under the Wasserstein ambiguity set and the Sinkhorn ambiguity set. Consider a simple worst-case expectation problem based on an ambiguity set $\mathcal{P}(\Xi)$:

$$\begin{aligned} \max_{\mathbb{P} \in \mathcal{P}} \quad & \mathbb{E}_{\mathbb{P}}[\eta] \\ \text{s.t.} \quad & \mathcal{P} = \{\mathbb{P} : \mathcal{D}(\mathbb{P}, \mathbb{P}_0) \leq \varepsilon\} \end{aligned} \tag{5.8}$$

To make the above problem solvable, we discretize the problem based on finite discrete supports. The candidate and nominal distributions are discretized based on finite samples $\{\eta_n\}_{n=1}^N$ and $\{\xi_m\}_{m=1}^M$, respectively. With Wasserstein ambiguity set defined through (1.10), the discretized version of problem (5.8) can be written as:

$$\begin{aligned} \max_{\pi\geq 0,\alpha\geq 0} \quad & \sum_{n=1}^N \eta_n \alpha_n \\ \text{s.t.} \quad & \sum_{n=1}^N \sum_{m=1}^M \|\eta_n - \xi_m\| \pi_{nm} \leq \varepsilon_w \\ & \sum_{m=1}^M \pi_{nm} = \alpha_n, \forall n = 1, ..., N \\ & \sum_{n=1}^N \pi_{nm} = \frac{1}{M}, \forall m = 1, ..., M \end{aligned} \tag{5.9}$$

It is worth noting that (5.9) and (1.10) differ because of the different purposes of the Wasserstein distances used. In particular, the Wasserstein distance in (5.9) measures the distance between a variable worst-case distribution and a fixed nominal distribution, whereas the Wasserstein distance in (1.10) calculates the distance between two fixed empirical distributions. As problem (5.9) determines the worst-case distribution, the probability masses α_n of the worst-case distribution are variables. Problem (5.9) can be equivalently rewritten as the following after eliminating the variable α

$$\begin{aligned} \max_{\pi\geq 0} \quad & \sum_{n=1}^N \sum_{m=1}^M \eta_n \pi_{nm} \\ \text{s.t.} \quad & \sum_{n=1}^N \sum_{m=1}^M \|\eta_n - \xi_m\| \pi_{nm} \leq \varepsilon_w \\ & \sum_{n=1}^N \pi_{nm} = \frac{1}{M}, \forall m = 1, ..., M \end{aligned} \tag{5.10}$$

Similarly, while the Sinkhorn ambiguity set is used, problem (5.8) can be rewritten as:

$$\begin{aligned}
\max_{\pi \geq 0} \quad & \sum_{n=1}^{N} \sum_{m=1}^{M} \eta_n \pi_{nm} \\
\text{s.t.} \quad & \sum_{n=1}^{N} \sum_{m=1}^{M} \|\eta_n - \xi_m\| \pi_{nm} + \gamma_s \sum_{n=1}^{N} \sum_{m=1}^{M} \pi_{nm} \log(\pi_{nm}) \leq \varepsilon_s \\
& \sum_{n=1}^{N} \pi_{nm} = \frac{1}{M}, \forall m = 1, ..., M
\end{aligned} \tag{5.11}$$

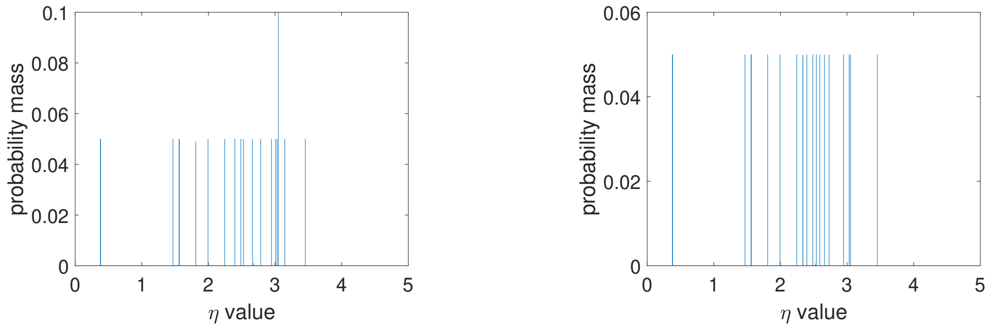
Next, we study the worst-case distribution under different settings. First, we consider the case that the true distribution is the normal distribution $\mathcal{N}(2.5, 0.8333)$. $\{\xi_m\}_{m=1}^{M}$ with $M = 20$ are sampled from $\mathcal{N}(2.5, 0.8333)$. A nominal distribution is supported on $\{\xi_m\}_{m=1}^{M}$. On the other hand, 500 samples are sampled from $U(0, 5)$, and then these 500 samples and $\{\xi_m\}_{m=1}^{20}$ are merged to be $\{\eta_n\}_{n=1}^{N}$ with $N = 520$. Candidate distributions are supported on $\{\eta_n\}_{n=1}^{N}$. Based on $\{\xi_m\}_{m=1}^{20}$ and $\{\eta_n\}_{n=1}^{520}$, the LP problem (5.10) and the NLP problem (5.11) are solved using linprog in Matlab and KNITRO in GAMS, respectively. The results with respect to different hyper-parameters (ε_w for (5.10); γ_s and ε_s for (5.11)) are shown in Figure 5.2 and Figure 5.3. Next, we set the true distribution as an uniform distribution $U(0, 5)$ and repeat the above experiment. The only difference is that the $M = 20$ samples $\{\xi_m\}_{m=1}^{M}$ are from the uniform distribution. The results are shown in Figure 5.4 and Figure 5.5. In those figures, "# points" means the number of points with non-zero probability masses (i.e., $\alpha_n = \sum_{m=1}^{M} \pi_{nm}^* \neq 0$).

As shown in Figures 5.2 and 5.4, the worst-case distributions from the Wasserstein-based method are discrete. Moreover, these worst-case distributions are supported on at most $M+1$ points ($M+1 = 21$ here). The Wasserstein-based method cannot hedge against the correct family of distributions here since the true distribution is continuous.

According to Figures 5.3 and 5.5, while the regularization parameter is relatively larger (e.g., $\gamma_s = 0.1$), the worst-case distributions from the Sinkhorn-based approach tend to be more continuous (smoother). As γ_s decreases, the worst-case distributions from the Sinkhorn-based method is more discrete and similar to the worst-case distributions from the Wasserstein-based method. This is explained by the fact that the Sinkhorn distance converges to the Wasserstein distance as $\gamma_s \longrightarrow 0$.

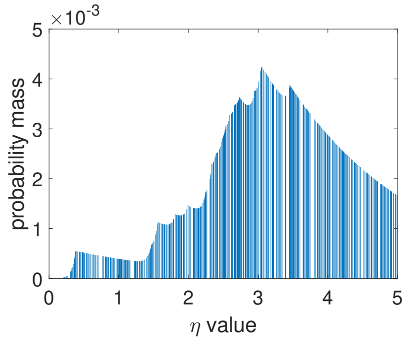
Furthermore, as ε_s decreases, the worst-case distributions from the Sinkhorn-based method also tends to be more discrete (e.g., when $\gamma_s = 10^{-8}$, the number of support points is reduced from 416 to 20). This is due to the fact that as ε_s decreases (the size of the Sinkhorn ambiguity set decreases), the worst-case distributions get closer to the nominal distribution which is the discrete distribution supported on limited points $\{\xi_m\}_{m=1}^M$.

On the other hand, the worst-case distribution from the Wasserstein DRO is restricted to a discrete distribution supported on at most $M+1$ points. Hence, Wasserstein DRO cannot hedge against the right family of distributions if the true distribution is continuous. The Sinkhorn ambiguity set overcomes the above limitation and it results in a smoother worst-case distribution. Accordingly, the Sinkhorn-based approaches can hedge against more general families of uncertainty distributions than the Wasserstein-based methods. By adjusting hyper-parameters γ_s and ε_s, the Sinkhorn-based approach can result in a wider variety of worst-case distributions than the Wasserstein-based methods.

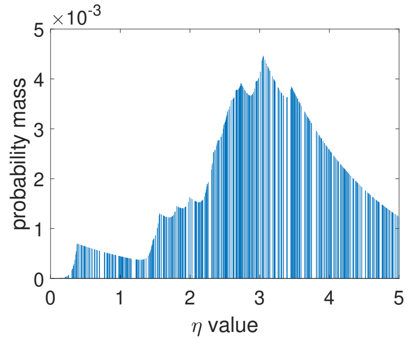


(a) $\varepsilon_w = 0.1$, 20 points (b) $\varepsilon_w = 0.0001$, 21 points

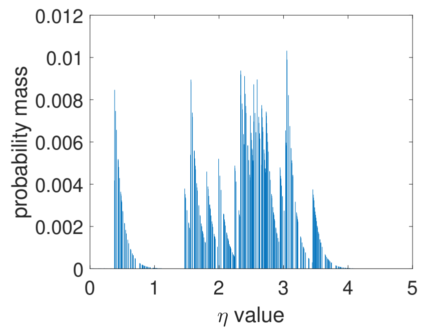
Figure 5.2: Worst-case distributions from Wasserstein ambiguity sets ($\xi \sim \mathcal{N}(2.5, 0.8333)$).



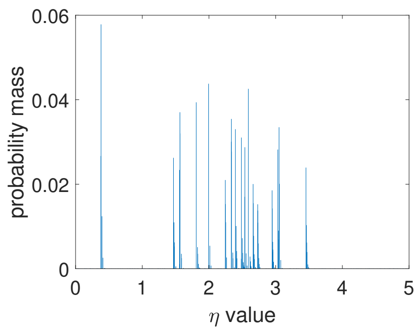
(a) $\gamma_s = 0.1, \varepsilon_s = 0.1$, 520 points



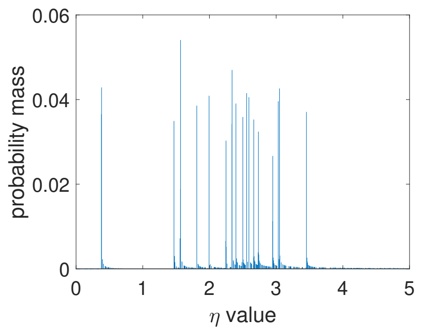
(b) $\gamma_s = 0.1, \varepsilon_s = 0.0001$, 520 points



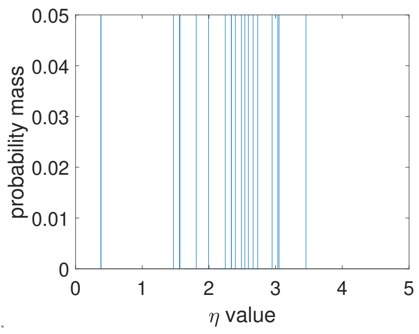
(c) $\gamma_s = 0.001, \varepsilon_s = 0.1$, 482 points



(d) $\gamma_s = 0.001, \varepsilon_s = 0.0001$, 233 points

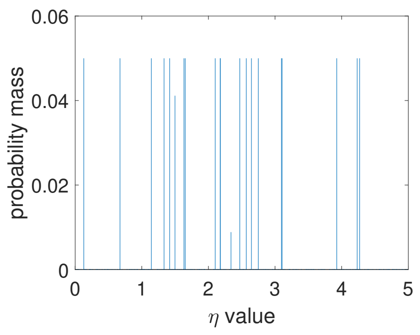


(e) $\gamma_s = 10^{-8}, \varepsilon_s = 0.1$, 416 points

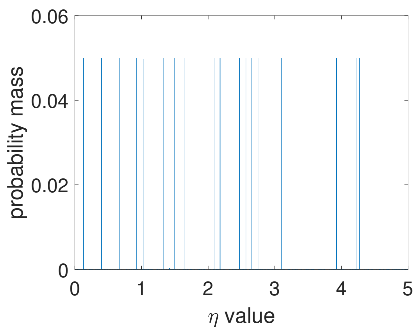


(f) $\gamma_s = 10^{-8}, \varepsilon_s = 0.0001$, 20 points

Figure 5.3: Worst-case distributions from Sinkhorn ambiguity sets ($\xi \sim \mathcal{N}(2.5, 0.8333)$).

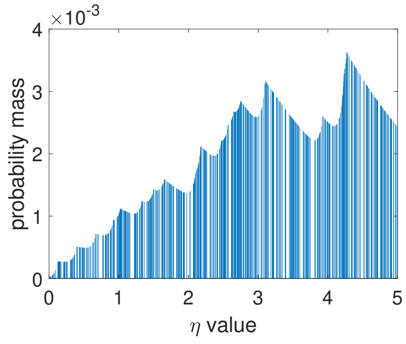


(a) $\varepsilon_w = 0.1$, 21 points

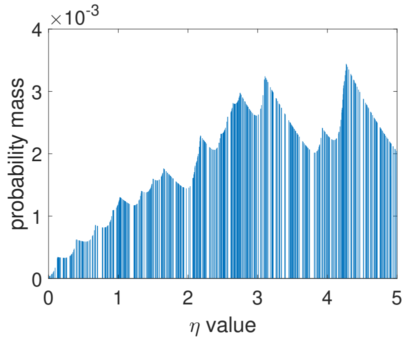


(b) $\varepsilon_w = 0.0001$, 21 points

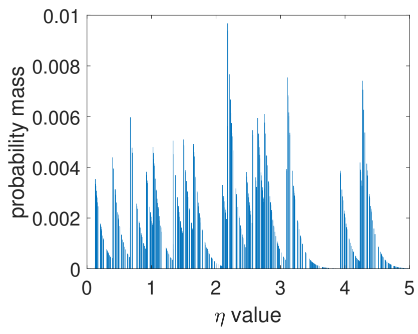
Figure 5.4: Worst-case distributions from Wasserstein ambiguity sets ($\xi \sim U(0,5)$).



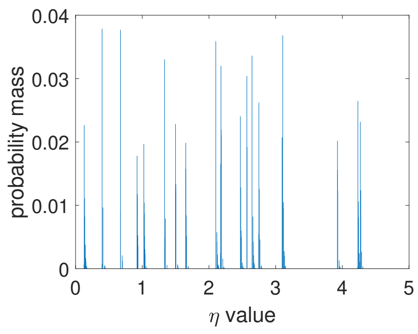
(a) $\gamma_s = 0.1, \varepsilon_s = 0.1$, 520 points



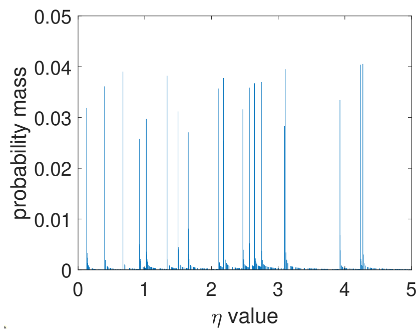
(b) $\gamma_s = 0.1, \varepsilon_s = 0.0001$, 520 points



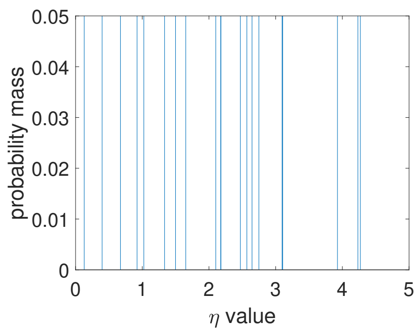
(c) $\gamma_s = 0.001, \varepsilon_s = 0.1$, 511 points



(d) $\gamma_s = 0.001, \varepsilon_s = 0.0001$, 200 points



(e) $\gamma_s = 10^{-8}, \varepsilon_s = 0.1$, 511 points



(f) $\gamma_s = 10^{-8}, \varepsilon_s = 0.0001$, 48 points

Figure 5.5: Worst-case distributions from Sinkhorn ambiguity sets ($\xi \sim U(0,5)$).

5.4 Sinkhorn DRCCP model reformulation

The general formulation of the Sinkhorn DRCCP is given as:

$$\min_{x\in\mathcal{X}} \quad f(x) \tag{5.12a}$$

$$\text{s.t.} \quad \max_{\mathbb{P}\in\mathcal{P}_{\gamma_s,\varepsilon_s}} Pr\left(\bigcup_{i=1}^{w} g_i(x,\eta) > 0\right) \leq \delta \tag{5.12b}$$

$$\mathcal{P}_{\gamma_s,\varepsilon_s} = \{\mathbb{P} : \mathcal{W}_{\gamma_s}(\mathbb{P},\mathbb{P}_0) \leq \varepsilon_s\} \tag{5.12c}$$

where $\mathcal{P}_{\gamma_s,\varepsilon_s}$ is the Sinkhorn ambiguity set with the hyper-parameters γ_s and ε_s. In this section, a tractable reformulation of the above Sinkhorn DRCCP is derived. To begin, the worst-case violation probability in (5.12b) is rewritten as a worst-case expectation problem:

$$\max_{\mathbb{P}\in\mathcal{P}_{\gamma_s,\varepsilon_s}} \quad \mathbb{E}_{\mathbb{P}}\left[\mathbb{I}\left(\max_{i=1,\ldots,w} g_i(x,\eta) > 0\right)\right] \tag{5.13}$$

where $\mathbb{I}$ is the indicator function expressed as:

$$\mathbb{I}\left(\max_{i=1,\ldots,w} g_i(x,\eta) > 0\right) = \begin{cases} 0, & \text{for} \quad \max_{i=1,\ldots,w} g_i(x,\eta) \leq 0 \\ 1, & \text{for} \quad \max_{i=1,\ldots,w} g_i(x,\eta) > 0 \end{cases} \tag{5.14}$$

Accordingly, (5.12b) can be rewritten as:

$$\max_{\mathbb{P}\in\mathcal{P}_{\gamma_s,\varepsilon_s}} \quad \mathbb{E}_{\mathbb{P}}\left[\mathbb{I}\left(\max_{i=1,\ldots,w} g_i(x,\eta) > 0\right)\right] \leq \delta \tag{5.15}$$

To address the nonconvex nature of the indicator function $\mathbb{I}$, the CVaR approximation [139] can be used. According to Reference [73], (5.15) can be rewritten as the constraint below using the CVaR approximation:

$$\max_{\mathbb{P}\in\mathcal{P}_{\gamma_s,\varepsilon_s}} \min_{\beta\in\mathbb{R}} \left\{\beta + \frac{1}{\delta}\mathbb{E}_{\mathbb{P}}\left(\left[\max_{i=1,\ldots,w} g_i(x,\eta) - \beta\right]^+\right)\right\} \leq 0 \tag{5.16}$$

According to Theorem 2 in Reference [151], when $\mathcal{P}_{\gamma_s,\varepsilon_s}$ is a compact convex set, $\max\limits_{\mathbb{P}\in\mathcal{P}_{\gamma_s,\varepsilon_s}}$ and $\min\limits_{\beta\in\mathbb{R}}$ in the above inequality can be switched:

$$\min_{\beta\in\mathbb{R}}\left\{\beta+\frac{1}{\delta}\max_{\mathbb{P}\in\mathcal{P}_{\gamma_s,\varepsilon_s}}\mathbb{E}_{\mathbb{P}}\left(\left[\max_{i=1,\ldots,w} g_i(x,\eta)-\beta\right]^+\right)\right\}\leq 0 \tag{5.17}$$

To make the above formulation tractable, the strong dual of the worst-case expectation in (5.17) is derived in the following paragraphs.

In many practical settings, the nominal distribution $\mathbb{P}_0$ is approximated by an empirical distribution $\hat{\mathbb{P}}_0$ constructed by the collected samples $\{\xi_m\}_{m=1}^{M}$. Accordingly, the worst-case expectation in (5.17) becomes:

$$\begin{aligned}&\max_{\mathbb{P}\in\mathcal{P}_{\gamma_s,\varepsilon_s}}\ \mathbb{E}_{\mathbb{P}}\left(\left[\max_{i=1,\ldots,w} g_i(x,\eta)-\beta\right]^+\right)\\ &\qquad \text{s.t. } \mathcal{P}_{\gamma_s,\varepsilon_s}=\left\{\mathbb{P}:\mathcal{W}_{\gamma_s}(\mathbb{P},\hat{\mathbb{P}}_0)\leq\varepsilon_s\right\}\end{aligned} \tag{5.18}$$

Note that problem (5.18) is not tractable since its decision variable is the distribution function. To tackle this issue, the following discretized version of the strong dual of (5.18) is derived (as shown in the appendix) based on samples $\eta_{n,m}$ generated from a kernel distribution:

$$\begin{aligned}\min_{\lambda,s,a}\quad & \lambda\bar{\varepsilon}_s+\frac{1}{M}\sum_{m=1}^{M}s_m\\ \text{s.t.}\quad & \lambda\gamma_s\geq\frac{1}{N}\sum_{n=1}^{N}a_{n,m},\forall m=1,...,M\\ & \left(\lambda\gamma_s,a_{n,m},\left[\max_{i=1,\ldots,w} g_i(x,\eta_{n,m})-\beta\right]^+-s_m\right)\in\mathcal{K}_{\exp},\forall n=1,...,N,\forall m=1,...,M\\ & \lambda\geq 0,\ \ s\in\mathbb{R}^M,\ \ a\in\mathbb{R}_+^{N\times M}\end{aligned} \tag{5.19}$$

where $\mathcal{K}_{\exp}$ in (5.19) denotes the exponential cone [166]: $\mathcal{K}_{\exp}=\left\{(t,u,v)\in\mathbb{R}_+\times\mathbb{R}_+\times\mathbb{R}:\exp\left(\frac{v}{t}\right)\leq\frac{u}{t}\right\}$ and the constant $\bar{\varepsilon}_s$ is a hyperparameter as defined in the Appendix. Note that $\bar{\varepsilon}_s$ can be considered as a hyper-parameter in the Sinkhorn DRCCP. While tuning the hyper-parameters of the Sinkhorn DRCCP, we can directly adjust $\bar{\varepsilon}_s$ instead of the original radius ε_s.

Based on the above dual formulation, we can replace the worst-case expectation in (5.17) with (5.19), and then merge the min operators:

$$
\begin{aligned}
&\min_{\beta,\lambda,s,a} \quad \beta + \frac{1}{\delta}\left\{\lambda\bar{\varepsilon}_s + \frac{1}{M}\sum_{m=1}^{M} s_m\right\} \leq 0 \\
&\lambda\gamma_s \geq \frac{1}{N}\sum_{n=1}^{N} a_{n,m}, \forall m = 1, ..., M \\
&\left(\lambda\gamma_s, a_{n,m}, \left[\max_{i=1,...,w} g_i(x,\eta_{n,m}) - \beta\right]^+ - s_m\right) \in \mathcal{K}_{\exp}, \forall n = 1, ..., N, \forall m = 1, ..., M \\
&\beta \in \mathbb{R},\;\; \lambda \geq 0,\;\; s \in \mathbb{R}^M,\;\; a \in \mathbb{R}_+^{N\times M}
\end{aligned} \tag{5.20}
$$

Afterwards, the above formulation can be used to replace (5.12b) and (5.12c). Furthermore, the inner min operator in (5.20) can be omitted. Subsequently, to address the expression of $\left[\max_{i=1,...,w} g_i(x,\eta_{n,m}) - \beta\right]^+$, we introduce variables $G_{n,m}$ for the epi-graph reformulation. Then, the original DRCCP problem is tackled by solving the following deterministic optimization problem:

$$
\begin{aligned}
\min_{x,\beta,\lambda,s,a,G} \quad & f(x) \\
\text{s.t.} \quad & \beta + \frac{1}{\delta}\left\{\lambda\bar{\varepsilon}_s + \frac{1}{M}\sum_{m=1}^{M} s_m\right\} \leq 0 \\
& \lambda\gamma_s \geq \frac{1}{N}\sum_{n=1}^{N} a_{n,m},\;\; \forall m = 1, ..., M \\
& (\lambda\gamma_s, a_{n,m}, G_{n,m} - s_m) \in \mathcal{K}_{\exp},\;\; \forall n = 1, ..., N,\;\; \forall m = 1, ..., M \\
& G_{n,m} \geq g_i(x,\eta_{n,m}) - \beta,\;\; \forall i = 1, ..., w,\;\; \forall n = 1, ..., N,\;\; \forall m = 1, ..., M \\
& G_{n,m} \geq 0,\;\; \forall n = 1, ..., N,\;\; \forall m = 1, ..., M \\
& x \in \boldsymbol{\mathcal{X}},\;\; \beta \in \mathbb{R},\;\; \lambda \geq 0,\;\; s \in \mathbb{R}^M,\;\; a \in \mathbb{R}_+^{N\times M}
\end{aligned} \tag{5.21}
$$

Note that the conic constraint in the above model can be rewritten as: $\lambda\gamma_s \exp\left(\frac{G_{n,m} - s_m}{\lambda\gamma_s}\right) \leq a_{n,m}$, $\forall n = 1, ..., N,\;\; \forall m = 1, ..., M$. Also note that model (5.21) does not need any assumptions on the uncertain constraints $g_i(x,\eta)$ in the original DRCCP optimization. When $f(x)$ and $g(x,\eta)$ are convex in x, problem (5.21) becomes a convex conic optimization with good scalability. While increasing sample size M and the number of uncertain constraints w, only the number of constraints having indices m and i would be increased. Since these constraints are convex, the optimization model complexity would not be dramatically increased.

5.5 Numerical example

In this section, the presented Sinkhorn DRCCP is applied to the following numerical example:

$$\min_{x} \quad 0.4x_1 + 0.5x_2 + 0.6x_3 + 1.2x_4 + 1.8x_5 \tag{5.22a}$$

$$\text{s.t.} \quad Pr\left\{\begin{aligned} &\xi_5 - x_5 > 0 \\ &\xi_4 + \xi_5 - x_4 - x_5 > 0 \\ &\xi_1 + \xi_4 + \xi_5 - x_1 - x_4 - x_5 > 0 \\ &\xi_2 + \xi_4 + \xi_5 - x_2 - x_4 - x_5 > 0 \\ &\xi_3 + \xi_4 + \xi_5 - x_3 - x_4 - x_5 > 0 \\ &\xi_1 + \xi_2 + \xi_4 + \xi_5 - x_1 - x_2 - x_4 - x_5 > 0 \\ &\xi_1 + \xi_3 + \xi_4 + \xi_5 - x_1 - x_3 - x_4 - x_5 > 0 \\ &\xi_2 + \xi_3 + \xi_4 + \xi_5 - x_2 - x_3 - x_4 - x_5 > 0 \\ &\xi_1 + \xi_2 + \xi_3 + \xi_4 + \xi_5 - x_1 - x_2 - x_3 - x_4 - x_5 > 0 \end{aligned}\right\} \leq \delta \tag{5.22b}$$

$$0 \leq x_i \leq 1 \quad i = 1, \cdots, 3 \tag{5.22c}$$

$$0 \leq x_4 \leq 2 \tag{5.22d}$$

$$0 \leq x_5 \leq 3 \tag{5.22e}$$

where the violation tolerance δ is set to be 0.1. In other words, the joint constraint satisfaction probability (JCSP) is required to be ≥ 0.9 $(1 - \delta = 0.9)$. $\xi_1 \sim \xi_5$ are correlated random variables which follow a multivariate normal distribution and the marginal distributions are $\xi_1 \sim N(0.8, 0.2^2)$, $\xi_2 \sim N(1.5, 0.3^2)$, $\xi_3 \sim N(1.2, 0.6^2)$, $\xi_4 \sim N(0.5, 0.4^2)$, $\xi_5 \sim N(0.7, 0.3^2)$.

The correlation matrix is given as:

$$R = \begin{pmatrix} 1.0 & -0.5 & 0.0 & 0.3 & -0.5 \\ -0.5 & 1.0 & -0.8 & 0.0 & 0.2 \\ 0.0 & -0.8 & 1.0 & 0.0 & 0.3 \\ 0.3 & 0.0 & 0.0 & 1.0 & 0.0 \\ -0.5 & 0.2 & 0.3 & 0.0 & 1.0 \end{pmatrix} \tag{5.23}$$

As the comparisons to the proposed Sinkhorn DRCCP, the Wasserstein DRCCP and the sample average approximation (SAA)-based method [14] are also applied to the same problem. We use the tractable models in our previous work [167] for the Wasserstein DRCCP and the SAA-based method in this study. The employed models of the Wasserstein DRCCP (based on the type-1 Wasserstein distance with ℓ_2 norm) and the SAA-based method are shown in Section A4.2 in Appendix.

In this numerical example, the Sinkhorn, Wasserstein, and SAA methods lead to the NLP, quadratically constrained programming (QCP), and mixed-integer linear programming (MILP) problems, respectively. They are modeled in GAMS and solved using KNITRO, XPRESS, and CPLEX, respectively.

Since the Sinkhorn DRCCP and the Wasserstein DRCCP have hyper-parameters (γ_s, $\bar{\varepsilon}_s$, and the number of expansion points N in the Sinkhorn DRCCP; ε_w in the Wasserstein DRCCP), these hyper-parameters should be tuned to avoid the over conservatism caused by the inappropriate values of the hyper-parameters. The hyper-parameters are tuned by the following algorithm:

Algorithm 2 Tuning the hyper-parameters for DRCCP

```
1: Input: $\bar{\varepsilon}_s$ or $\varepsilon_w \in [0.0001, 0.1]$
          $\gamma_s \in [0.001, 0.1]$
          $N \in \{10, 15, 20, 25\}$
          $tol$ : a positive user-defined tolerance
2: Data: 100 different $M$-sample sets of $[\xi_1, \xi_2, \xi_3, \xi_4, \xi_5]$,
         a $10^4$-sample set of $[\xi_1, \xi_2, \xi_3, \xi_4, \xi_5]$ as the validation set $\mathcal{S}_V$
3: Take an initial guess of the hyper-parameters.
4: while $Q^J - (1-\delta) \geq tol \vee Q^J \leq (1-\delta)$ do
5:     Change the values of hyper-parameters if the iteration $\# \geq 2$.
6:     Solve the numerical example by using a DRCCP method 100 times based on 100
       different $M$-sample sets.
7:     Calculate the JCSPs of the 100 optimal solutions obtained from the previous step,
       through the sample average approximation [14] based on $\mathcal{S}_V$.
8:     Find the $J$-th largest value of the JCSPs from the previous step. This value is denoted
       as $Q^J$.
9: end while
10: Output: The tuned hyper-parameters
```

The value M for the M-sample sets in Algorithm 2 is set as 5, 10, 20, 30, or 40 in this numerical example. Through Algorithm 2, we can assume that an optimal solution obtained from a DRCCP method with the tuned hyper-parameters would be feasible (the corresponding JCSP $\geq 1-\delta$) with J% probability. J is set to be 95 in this work. The above algorithm is used to show the best performance of a DRCCP method based on the appropriate selection of hyper-parameters, for the demonstration purpose. In our experience, the performance of the Sinkhorn DRCCP is most sensitive to the regularization parameter γ_s, followed by the radius size $\bar{\varepsilon}_s$, and finally the number of expansion points N. Thus, while tuning the hyper-parameters for the Sinkhorn DRCCP, we first tune γ_s. Then, with the fixed chosen γ_s, we tune $\bar{\varepsilon}_s$. Finally, with the fixed chosen γ_s and $\bar{\varepsilon}_s$, we tune N. By these means, we can get the best combination of hyper-parameters for the Sinkhorn DRCCP faster.

The numerical problem is solved 100 times based on 100 different M-sample sets and the tuned hyper-parameters. The mean and the standard deviation of the 100 optimal objectives from the 100-time problem-solving are used to present the general performance of a DRCCP approach, with respect to the sample size M. In addition, the 100 JCSPs correspond to the 100-time problem-solving are calculated based on a test set containing 10^4 samples. Then, the percentage of JCSPs≥ 0.9 (the required value in this example) and the minimum JCSP among the 100 JCSPs are calculated, and they are important bases for investigating the reliability of a DRCCP method. The above-mentioned results are shown in the following tables.

Table 5.1: Means and standard deviations of optimal objectives from different methods with respect to different sample sizes

Sample size M	Sinkhorn DRCCP	Wasserstein DRCCP	SAA
	mean / standard deviation		
5	6.8483 / 0.6628	7.0735 / 0.6762	5.3014 / 0.7146
10	6.5901 / 0.5646	6.7828 / 0.5898	5.3607 / 0.4076
20	6.4773 / 0.4828	6.5559 / 0.4999	5.5034 / 0.3085
30	6.4473 / 0.3748	6.4848 / 0.3844	5.5162 / 0.2630
40	6.3035 / 0.3201	6.3171 / 0.3211	5.5562 / 0.2313

Table 5.2: Feasibility percentages and the minimum JCSPs corresponding to the optimal results shown in Table 5.1

Sample size M	Sinkhorn DRCCP	Wasserstein DRCCP	SAA
	Feasibility percentages / minimum JCSPs		
5	95 / 80.43	95 / 83.03	5 / 31.91
10	95 / 84.46	95 / 86.20	7 / 32.51
20	95 / 85	95 / 82.46	11 / 55.53
30	96 / 88.12	95 / 86.91	11 / 62.16
40	95 / 88.30	95 / 87.88	25 / 67.10

[a] The percentage values in this table are based on %.

Table 5.3: Tuned hyper-parameters corresponding to the results shown in Table 5.1

Sample size M	Sinkhorn DRCCP	Wasserstein DRCCP
	$\bar{\varepsilon}_s/\gamma_s$/N	ε_w
5	0.001/0.059/25	0.04
10	0.001/0.02/15	0.022
20	0.0001/0.014/15	0.011
30	0.0001/0.0072/15	0.0065
40	0.0001/0.0012/15	0.004

Table 5.4: Average solution times of different methods

	Sinkhorn DRCCP	Wasserstein DRCCP	SAA
Average solution time (s)	0.4224	0.2009	0.4946

[a] For each method, the average solution time is calculated based on 100 optimizations.

[b] Each optimization is based on a set of 40 samples.

According to Table 5.1, with respect to a fixed sample size, the SAA-based method always achieves the least conservative mean of optimal objectives among all the approaches. However, as can be seen from Table 5.2, with respect to a fixed sample size, the SAA-based method always has the worst feasibility percentage and minimum JCSP among all the approaches which means that the SAA-based method is the least reliable method. Moreover, the SAA-based method always has very low feasibility percentages (only $5 \sim 25\%$) which means that the SAA-based method is not reliable for handling this example. This is because the SAA-based method does not consider the distributional ambiguity of uncertainty for enhancing the solution robustness. Furthermore, the SAA-based approach lacks tunable hyper-parameters to enhance the reliability of the solutions, which sets it apart from the DRCCP methods.

In Table 5.3, with respect to a certain sample size, the tuned radius value for the Sinkhorn DRCCP is smaller than the tuned radius value for the Wasserstein DRCCP. This is because the worst-case distribution of the Wasserstein DRCCP is restricted to a discrete distribution, while the worst-case distribution of the Sinkhorn DRCCP is more general and can be either continuous or discrete. As a result, the less restricted worst-case distribution of the Sinkhorn DRCCP has more leeway to result in a more conservative solution. If the same radius value is used for both the Sinkhorn and Wasserstein DRCCPs, the Sinkhorn DRCCP would achieve a more conservative solution due to the above reason. This is evidenced based on our previous experiment.

According to the results from the Sinkhorn DRCCP and the Wasserstein DRCCP, both methods can always achieve feasibility percentages $\geq 95\%$ which means that they are reliable to obtain feasible solutions with high confidence if their hyper-parameters are appropriately selected. Furthermore, with respect to a fixed sample size, the Sinkhorn DRCCP can always obtain optimal objectives with better means and smaller standard deviations than the Wasserstein DRCCP. Based on the above discussion, under the same level of reliability, the Sinkhorn DRCCP has better performance than the Wasserstein DRCCP in this example, in the aspect of the solution quality.

As can be seen from Table 5.4, the SAA-based method has the longest average solution time since it involves binary variables causing higher computational cost. The Wasserstein DRCCP has the lower computational burden than the presented Sinkhorn method because the numerical example becomes a QCP problem by using the Wasserstein DRCCP model in (A66) in Appendix which has the lower computational cost than the nonlinear Sinkhorn DRCCP model in (5.21). Although the presented Sinkhorn approach has the higher computational cost than the Wasserstein method, the Sinkhorn approach can outperform the Wasserstein DRCCP by achieving a less conservative solution with lower variability. In addition, the Sinkhorn method does not need any assumptions on the constraints in the joint chance constraint. However, the Wasserstein DRCCP requires nontrivial assumptions on the constraints, e.g., constraints are affine in uncertainty.

5.6 Case study

In this section, the proposed method is applied to the alkylation process optimization under uncertainty. This optimization problem is modified from Example 14.3 in Reference [162] by adding random parameters to the original process model. The simplified flowsheet of the studied process is illustrated in Figure 5.6. The reactor feeds include an olefin stream (100% butane), a pure isobutane recycle stream, a 100% isobutane make-up stream, and an acid catalyst. The spent acid is removed from the reactor. The product stream from the reactor is passed through a fractionator to separate isobutane and the alkylate product.

The objective is to maximize the total profit, which is calculated based on the alkylate product value (\$0.063/octane-barrel), olefin feed cost (\$5.04/barrel), isobutane recycle cost (\$0.035/barrel), acid addition cost (\$10.00/per thousand pounds), and isobutane makeup cost (\$3.36/barrel).

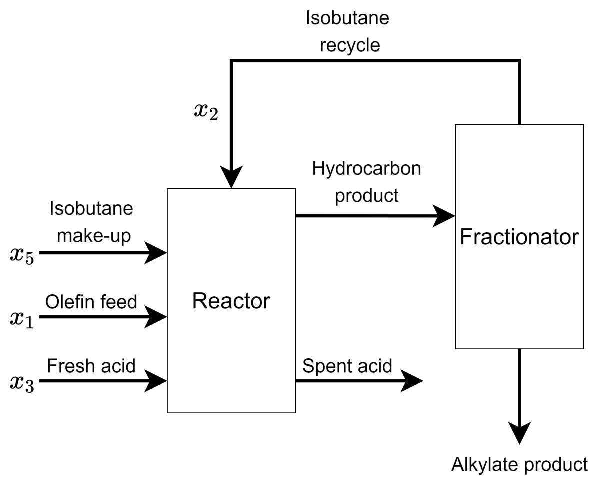


Figure 5.6: Flowsheet of the studied alkylation process

The studied process optimization problem involving the process model is formulated as follows. $x_1 \sim x_{10}$ are variables as explained in Table 5.5. Each of them is associated with some physical bounds. Equation (5.24g) defines the volumetric balance for the reactor. The alkylate yield x_4 is equal to the sum of the olefin feed x_1 and the isobutane makeup x_5 minus the volumetric shrinkage, which 0.22 of alkylate yield: $x_4 = x_1 + x_5 - 0.22x_4 \implies x_5 = 1.22x_4 - x_1$. The acid strength x_6 is calculated through equation (5.24h) based on the assumption that the added acid (acid addition rate is x_3) has acid strength of 98%. The external isobutane-to-olefin ratio x_8 equals to the sum of the isobutane recycle x_2 and the isobutane makeup x_5 divided by the olefin feed x_1, which is calculated via equation (5.24i).

Table 5.5: Variables of the optimization model

Symbol	Variable	Unit
x_1	Olefin feed	barrels/day
x_2	Isobutane recycle	barrels/day
x_3	Acid addition rate	thousands of pounds/day
x_4	Alkylate yield	barrels/day
x_5	Isobutane makeup	barrels/day
x_6	Acid strength	weight percent
x_7	Motor octane number	–
x_8	External isobutane-to-olefin ratio	–
x_9	Acid dilution factor	–
x_{10}	F-4 performance number	–

$$\max_{x} \quad 0.063x_4x_7 - 5.04x_1 - 0.035x_2 - 10x_3 - 3.36x_5 \tag{5.24a}$$

$$\text{s.t.} \quad Pr\left\{\begin{array}{l} x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2) - 0.89x_4 \geq 0 \\ -x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2) + 1.12x_4 \geq 0 \\ -\xi_4 + \xi_5 x_7 - 0.89x_{10} \geq 0 \\ \xi_4 - \xi_5 x_7 + 1.12x_{10} \geq 0 \end{array}\right\} \geq 0.9 \tag{5.24b}$$

$$86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_6 - 89) - 0.89x_7 \geq 0 \tag{5.24c}$$

$$-86.35 - 1.098x_8 + 0.038x_8^2 - 0.325(x_6 - 89) + 1.12x_7 \geq 0 \tag{5.24d}$$

$$35.82 - 0.222x_{10} - 0.89x_9 \geq 0 \tag{5.24e}$$

$$-35.82 + 0.222x_{10} + 1.12x_9 \geq 0 \tag{5.24f}$$

$$x_5 = 1.22x_4 - x_1 \tag{5.24g}$$

$$x_6 = \frac{98000x_3}{x_4x_9 + 1000x_3} \tag{5.24h}$$

$$x_8 = \frac{x_2 + x_5}{x_1} \tag{5.24i}$$

$$0 \leq x_1 \leq 2000, \;\; 0 \leq x_2 \leq 16000, \;\; 0 \leq x_3 \leq 120, \;\; 0 \leq x_4 \leq 5000 \tag{5.24j}$$

$$0 \leq x_5 \leq 2000, \;\; 85 \leq x_6 \leq 93, \;\; 90 \leq x_7 \leq 95, \;\; 3 \leq x_8 \leq 12 \tag{5.24k}$$

$$0.01 \leq x_9 \leq 4, \;\; 145 \leq x_{10} \leq 162 \tag{5.24l}$$

In the first two constraints of the joint chance constraint (5.24b), $x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2)$ is the regression model for computing x_4 by using x_1 and x_8. In the last two constraints of the joint chance constraint, $-\xi_4 + \xi_5 x_7$ is the regression model for computing x_{10} by using x_7. The inequalities restrict the percentage errors between the regression model outputs and the true values to the range $0.89 \sim 1.12$. $\xi_1 \sim \xi_5$ are uncertain parameters in the regression models. They either follow normal distributions $\xi_1 \sim N(1.12, 0.028^2)$, $\xi_4 \sim N(133, 3.325^2)$, or uniform distributions $\xi_2 \sim U(0.1277, 0.1356)$, $\xi_3 \sim U(0.0065, 0.0069)$, $\xi_5 \sim U(2.91, 3.09)$. Inequalities (5.24c)-(5.24f) are based on the regression model $86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_6 - 89)$ for x_7, and the regression model $35.82 - 0.222x_{10}$ for x_9. This optimization problem is a nonlinear joint chance-constrained optimization. Apart from the uncertain parameters and the percentage error tolerances for the regression models, all the settings for the studied process optimization problem are taken from Reference [162].

As the comparisons to the presented Sinkhorn DRCCP, the Wasserstein DRCCP and the SAA-based approach are also applied to this process optimization problem. We use the models (A66) and (A67) in Appendix for the Wasserstein DRCCP and the SAA approach in this case study, respectively. Notably, based on our experiments, the Wasserstein DRCCP would converge to infeasible solutions if the model (A66) is directly applied to this process optimization problem. This is mainly because the uncertain parameter ξ_4 has a much larger value than other uncertain parameters. Wasserstein DRCCP leads to biased worst-case distribution. More specifically, according to (1.7), since ξ_4 has a much larger value than other uncertain parameters, ξ_4 has more impact on the cost function $c(\cdot, \cdot)$. Thus, the Wasserstein distance computation would be dominated by ξ_4 and the other uncertain parameters may be neglected in the distance computation. Meanwhile, as discussed in Section 5.3, the Wasserstein distance generates the extreme transportation plan leading to the extreme worst-case distribution. Accordingly, the biased Wasserstein distance computation dominated by ξ_4 causes the biased extreme transportation plan leading to the extremely biased worst-case distribution. This outcome causes the extremely biased solution which may be infeasible. To overcome this issue, we divide the sample values of ξ_4 by 100 for the Wasserstein DRCCP model. Also, for the Wasserstein DRCCP model, the terms $\xi_5 x_7 - 0.89x_{10}$ and $-\xi_5 x_7 + 1.12x_{10}$ in the third and fourth constraints in (5.24b) are also divided by 100 to balance the scaling of ξ_4. By this scaling process, the Wasserstein DRCCP can converge to feasible solutions. This scaling process has no significant impact on the Sinkhorn DRCCP and SAA methods. This is because the Sinkhorn distance involves the entropic regularization smoothing out the transportation plan that prevents the Sinkhorn distance from getting the extreme transportation plan. Thus, the Sinkhorn ambiguity set is more robust to the data containing extremely different scales. In terms of the SAA method, since the constraint satisfaction is independent of the relative scales of uncertain parameters, the SAA method does not encounter the above-mentioned scale issue. Based on the above reasons, the above-mentioned scaling process is not applied to the Sinkhorn DRCCP and the SAA method in this case study.

The hyper-parameters in both Sinkhorn DRCCP and Wasserstein DRCCP are tuned through Algorithm 2 to avoid the over conservatism caused by the inappropriate selection of the hyper-parameters. After tuning the hyper-parameters, the process optimization problem in this case study is solved 100 times based on different data set as mentioned in Section 5.5 to compare the performances of different DRCCP approaches. This problem is solved as NLP, NLP, and mixed-integer nonlinear programming (MINLP) problems by using the Sinkhorn DRCCP, Wasserstein DRCCP, and SAA approach, respectively. We use KNITRO in GAMS to solve all the optimization problems in this case study. The obtained results are shown in the following tables.

Table 5.6: Means and standard deviations of optimal objectives from different methods with respect to different sample sizes

Sample size M	Sinkhorn DRCCP	Wasserstein DRCCP	SAA
	mean / standard deviation		
10	2528.5 / 100.24	2470.0 / 119.01	2635.5 / 26.95
20	2531.9 / 75.51	2471.5 / 117.01	2633.9 / 37.1493
30	2532.5 / 73.38	2484.1 / 91.87	2623.7 / 44.11
40	2543.7 / 61.69	2500.1 / 86.18	2562.9 / 107.01

Table 5.7: Feasibility percentages and the minimum JCSPs corresponding to the optimal results shown in Table 5.6

Sample size M	Sinkhorn DRCCP	Wasserstein DRCCP	SAA
	Feasibility percentages (%)/ minimum JCSPs (%)		
10	95 / 86.15	95 / 86.19	13 / 45.24
20	95 / 86.43	95 / 86.27	22 / 66.06
30	95 / 87.13	95 / 86.91	24 / 66.39
40	96 / 88.31	96 / 87.92	40 / 72.32

Table 5.8: Tuned hyper-parameters corresponding to the results shown in Table 5.6

Sample size M	Sinkhorn DRCCP	Wasserstein DRCCP
	$\bar{\varepsilon}_s/\gamma_s/\mathrm{N}$	ε_w
10	0.1/0.0013/20	0.00027
20	0.01/0.001/15	0.00015
30	0.01/0.0007/10	0.00014
40	0.01/0.00045/10	0.00006

Table 5.9: Average solution times of different methods (sample size $M = 40$)

	Sinkhorn DRCCP	Wasserstein DRCCP	SAA
Average solution time (s)	0.5726	0.2779	526.81

According to Table 5.6, the SAA-based method always has the worst feasibility percentage and the JCSP is not reaching the target value. This means that the SAA method is not reliable because of not considering the distributional ambiguity of uncertainty.

According to the results from the Sinkhorn DRCCP and the Wasserstein DRCCP, the two methods can always achieve feasibility percentages ≥ 0.95 which means that they are reliable for getting feasible solutions with high confidence if their hyper-parameters are appropriately tuned. Moreover, with respect to a fixed sample size, the Sinkhorn DRCCP can always reach optimal objectives with the less conservative mean and smaller standard deviation than the Wasserstein DRCCP. Accordingly, with the same level of reliability, the Sinkhorn DRCCP outperforms the Wasserstein DRCCP in this case study, in the aspect of the solution quality.

As can be seen from Table 5.9, the SAA method has the longest average solution time because it contains binary variables. The Wasserstein DRCCP has higher computational efficiency than the proposed Sinkhorn DRCCP since the Wasserstein distance does not has the nonlinear entropic regularization term in the Sinkhorn distance. Although the Sinkhorn DR-CCP has slightly lower computational efficiency than the Wasserstein DRCCP, the Sinkhorn DRCCP can outperform the Wasserstein DRCCP by reaching a less conservative solution with lower variability. In addition, the experiments for this case study show that the Sinkhorn DRCCP does not suffer from the scale issue of uncertain parameters.

5.7 Conclusions

The DRCCP based on the Sinkhorn ambiguity set is proposed in this work. The Sinkhorn ambiguity set is constructed based on the Sinkhorn distance which is a variant of the Wasserstein distance with the entropic regularization, which leads to a smooth transportation plan. With the smooth transportation plan, the Sinkhorn ambiguity set can generate smooth worst-case distribution which is not limited to be discrete. Thus, the Sinkhorn DRCCP is based on the more general worst-case distribution than the Wasserstein DRCCP which is restricted to the discrete worst-case distribution supported on at most $M+1$ sample points (M is the number of real collected samples). Also, due to the entropic regularization in the Sinkhorn distance, the Sinkhorn DRCCP is more robust to the data containing features with extremely different scales than the Wasserstein DRCCP. Furthermore, the presented Sinkhorn DRCCP is formulated as a tractable conic model based on the CVaR approximation as well as the discretized kernel distribution relaxation, and this model does not require some assumptions on uncertain constraints as needed by the Wasserstein method. Based on the aforementioned, the Sinkhorn DRCCP is a more practical approach for real-world problems, that overcomes the limitations of the Wasserstein method. As shown by the results of a numerical example and the case study in this work, the Sinkhorn DRCCP outperforms the Wasserstein DRCCP by obtaining less conservative solutions with lower variability.

As to future work, it is worth to investigate a more efficient tuning algorithm for the hyper-parameters in the proposed approach. In addition, the presented method can be extended to multi-stage adaptive optimization problems.

Chapter 6

A Novel Efficient Algorithm for Improving Solution Quality of Distributionally Robust Chance-Constrained Optimization

Abstract

Distributionally robust chance-constrained optimization (DRCCP) is a powerful technique to handle optimization problems involving uncertainty in constraint functions. However, the DRCCP methods are highly sensitive to outliers and extreme samples in the data sets used. The outliers and extreme samples may deteriorate the decision quality of DRCCP. Although there are numerous outlier detection techniques, they are either unable to pinpoint the samples causing overly conservative solutions, or incompatible with DRCCP models. This work proposes a novel, efficient, and widely compatible algorithm that generates a representative subset of the original data set and removes samples causing overly conservative solutions to exclude data points with relatively negative impacts on the DRCCP solution. By solving the DRCCP problem based on the generated subset, the DRCCP solution quality can be enhanced while simultaneously ensuring the solution feasibility. Two linear numerical examples and a nonlinear process optimization problem are studied to demonstrate its efficacy.

6.1 Introduction

Optimization problems in real-world scenarios typically involve uncertainty, making it difficult to arrive at reliable optimal solutions. The presence of unexpected random factors may cause constraint violations, further complicating the problem of obtaining reliable optimal decisions. Chance-constrained programming (CCP) [10] is a widely-used technique for optimization problems involving uncertainty in constraints. In CCP, the optimal decision needs to be feasible according to a user-defined probability. There are two categories of CCP: individual chance-constrained programming (ICCP) and joint chance-constrained programming (JCCP) [11, 21]. ICCP only requires each constraint to be fulfilled with its own level of confidence. On the other hand, JCCP is more comprehensive and ensures that all constraints are met concurrently with a specific confidence level. Although the JCCP can restrict the probability of any constraint violation in a problem, solving it can be challenging as it involves addressing a multidimensional distribution [13]. Moreover, a JCCP problem is convex only when certain conditions are met, such as the original constraints in the joint chance constraint being jointly convex in both decision variables as well as random parameters and the random parameters following log-concave distributions [132]. Consequently, JCCP problems are usually solved using approximation methods like analytical approximations and sampling-based methods [28].

In practical scenarios, it can be challenging to know the exact distributions of uncertainty. Therefore, a more practical approach for solving JCCP problems is to exploit the worst-case distribution from all possible data-generating distributions to obtain the distributionally robust solution, instead of estimating the true uncertainty distribution. This is the idea behind distributionally robust chance-constrained programming (DRCCP) [70]. Within the realm of DRCCP, the possible data-generating distributions are referred to as candidate distributions, and the set of candidate distributions is typically called the ambiguity set [133]. Candidate distributions are inferred from historical data and characterized by prior distributional knowledge, such as moments, structural properties, domain knowledge, etc., of the unknown data-generating distribution. The ambiguity set composed of candidate distributions must have a high degree of confidence in capturing the true data-generating distribution, while also being small enough to exclude distributions that could cause excessively conservative decisions [134]. Moreover, the ambiguity set should be straightforward to derive from the data, and it should facilitate a feasible reformulation of the DRCCP as a mathematical problem that can be effectively solved using commonly available optimization solvers [135].

The moment-based and metric-based ambiguity sets have been extensively studied and received significant attention among various types of ambiguity sets [57–59]. A moment-based ambiguity set consists of all distributions that fulfill specific moment constraints [133, 136, 137]. Prior knowledge of a certain level of moment information is required for constructing a moment-based ambiguity set. However, as various distributions could have the same moments, moment constraints within a moment-based ambiguity set may not be stringent enough to eliminate distributions that could cause excessively conservative decisions [60]. In addition, even when there is sufficient data, distributionally robust chance constraints relying on moments-based ambiguity sets are unable to provide a closely accurate approximation of original chance constraints [61]. In order to overcome the above problems, metric-based ambiguity sets offer an appealing alternative by defining the ambiguity set as a ball in the probability distribution space [138]. In a metric-based ambiguity set, all candidate distributions are centered around a nominal distribution, with a radius determined by the prescribed probability metric. The nominal distribution is established based on historical data, while the size of the radius of the ambiguity set is a user-defined hyper-parameter that enables the user to manage the degree of conservatism in the optimization problem [30]. The Wasserstein distance, Sinkhorn distance, and maximum mean discrepancy are effective probability metrics for establishing metric-based ambiguity sets [17, 167]. The Wasserstein distance measures the distance between two probability distributions, by computing the optimal transport plan to move one distribution into the other while minimizing the transportation cost [168]. The Sinkhorn distance is a variant of the Wasserstein distance with entropic regularization [164]. The maximum mean discrepancy is a measure quantifying the difference between two probability distributions by comparing the means of their embeddings in a reproducing kernel Hilbert space (RKHS) [140]. There are several studies on the DRCCP methods based on the metric-based ambiguity sets constructed by using the above metrics [17, 30, 70, 73, 167]. In this study, we exclusively concentrate on the metric-based ambiguity sets due to their advantages over the moment-based ambiguity sets and their availability of diverse options. Particularly, we focus on the DRCCP method based on the ambiguity set constructed by the Wasserstein distance (Wasserstein DRCCP) in this work, as the Wasserstein DRCCP is the most well-known DRCCP method with satisfactory performance.

Although DRCCP is an effective technique for addressing optimization under uncertainty, it is highly sensitive to outliers and extreme data samples in the used data set. Outliers and extreme data samples may deteriorate the DRCCP performance and lead to overly conservative decisions. Outliers are defined as data points significantly differing from most other data points in a data set. Outliers can come from various sources such as measurement errors, data entry or processing errors, unrepresentative sampling, etc [169, 170]. Extreme data samples can be defined as data values that are plausible but extremely high or low. In other words, data samples with values at the extreme ends of the data distribution are called extreme samples. Extreme samples may not necessarily be outliers. They are extreme but they may still follow the general trend of most samples in the data set. Although there are several outlier detection approaches in the literature [171–173], those approaches cannot pinpoint extreme sample points leading to poor DRCCP solutions. Some methods have been proposed to mitigate the negative impacts of outliers on distributionally robust optimization (DRO). DRO is similar to DRCCP, which optimizes the worst-case expectation of an uncertain objective over an ambiguity set [174]. Zhai et al. [175] developed the Distributional and Outlier Robust Optimization (DORO) framework which is an outlier robust refinement of DRO. This approach is based on a refined risk function preventing DRO from overfitting to potential outliers. Jiang et al. [176] introduced a framework combining DRO with distributional favorable optimization (DFO). This integration enhances the robustness of DRO against outliers while maintaining satisfactory out-of-sample performance. Rahimian et al. [177] proposed a rigorous mathematical approach to identify effective samples which are influential to the optimal solution of the DRO problem based on total variation distance. This approach has the potential to identify the samples that could potentially have negative impacts on the DRO solution. Since the above methods are only designed for DRO which does not involve uncertain constraints, the above approaches do not take into account the joint constraint satisfaction probability (JCSP) of the optimal decision. However, for the DRCCP problems involving joint chance constraints, it is important to ensure the DRCCP solution satisfies the JCSP threshold. Therefore, the above methods designed for DRO are not compatible with DRCCP.

This research presents an innovative algorithm for enhancing the decision quality of DRCCP. This algorithm generates a subset representing the original data set and trims outliers as well as extreme data points to exclude data samples with negative impacts on the DRCCP solution quality. Meanwhile, this algorithm can ensure that the optimal DRCCP decision meets the required JCSP. The algorithm is an iterative approach guided by the DRCCP. The presented algorithm is compatible with arbitrary DRCCP models such as Wasserstein DRCCP, Sinkhorn DRCCP, kernel DRCCP, etc. Since the Wasserstein DRCCP is the most famous DRCCP approach, we use the Wasserstein DRCCP for the proposed algorithm in this study.

This research is structured as follows. The background knowledge of DRCCP and Wasserstein DRCCP is introduced in Section 6.2. The developed algorithm is presented in Section 6.3. In Section 6.4, two numerical examples are studied. Subsequently, the application of the presented algorithm to a nonlinear process optimization under uncertainty is demonstrated in Section 6.5. Finally, this work is summarized in Section 6.6.

6.2 Problem statement

In this section, distributionally robust chance-constrained programming (DRCCP) and Wasserstein DRCCP are introduced.

6.2.1 Distributionally robust chance-constrained optimization

As discussed in Section 6.1, joint chance-constrained programming (JCCP) is more comprehensive than individual chance-constrained programming (ICCP), and thus we only consider JCCP in this work. The details of JCCP and ICCP are elaborated in Section 1.2.3. According to the discussion in Sections 1.2.4 and 6.1, the true uncertainty distribution in a JCCP problem is often unknown. Thus, to handle distributional ambiguity of uncertainty, the ambiguity set composed of candidate distributions can be combined with the JCCP to result in the DRCCP. The general formulation of a DRCCP problem is shown below:

$$\min_{x\in\mathcal{X}} \quad f(x) \tag{6.1a}$$

$$\text{s.t.} \quad \min_{\mathbb{P}\in\mathcal{P}(\Xi)} Pr(g_i(x,\xi) \leq 0, \quad \forall i = 1, ..., w) \geq 1-\delta \tag{6.1b}$$

where $\mathbb{P}$ is the probability distribution of ξ. $\mathcal{P}(\Xi)$ is the ambiguity set composed of all candidate distributions. Inequality (6.1b) is the distributionally robust joint chance constraint (DRJCC) requiring the worst-case JCSP greater than or equal to the user-defined probability level $1-\delta$, over the ambiguity set $\mathcal{P}(\Xi)$.

The inequality (6.1b) can be reformulated equivalently as the worst-case violation probability form:

$$\max_{\mathbb{P}\in\mathcal{P}(\Xi)} Pr\left(\bigcup_{i=1}^{w} g_i(x,\xi) > 0\right) \leq \delta \tag{6.2}$$

The left-hand side of the above inequality is the worst-case violation probability, which is the maximum probability of constraint violation ($g_1(x,\xi) > 0$ or $g_2(x,\xi) > 0$, or ..., or $g_w(x,\xi) > 0$) under all potential probability distributions within the ambiguity set. The inequality (6.2) is more useful for the derivation of tractable DRCCP models.

6.2.2 Wasserstein DRCCP

The general formulation of the Wasserstein DRCCP based on the type-1 Wasserstein distance $\mathcal{W}_1$, is given as:

$$\min_{x\in\mathcal{X}} \quad f(x) \tag{6.3a}$$

$$\text{s.t.} \quad \min_{\mathbb{P}\in\mathcal{P}_{\varepsilon_w}(\Xi)} Pr\left(g_i(x,\xi) \leq 0, \forall i = 1,...,w\right) \geq 1-\delta \tag{6.3b}$$

$$\mathcal{P}_{\varepsilon_w}(\Xi) = \{\mathbb{P} : \mathcal{W}_1(\mathbb{P},\mathbb{P}_0) \leq \varepsilon_w\} \tag{6.3c}$$

Expression (6.3c) defines the Wasserstein ambiguity set $\mathcal{P}_{\varepsilon_w}(\Xi)$ established by using the type-1 Wasserstein distance $\mathcal{W}_1$. The Wasserstein distance is explained in detail in Section 1.2.5. In $\mathcal{P}_{\varepsilon_w}(\Xi)$, all the Wasserstein distances between all the candidate distributions $\mathbb{P}$ and the nominal distribution $\mathbb{P}_0$ should be within a user-defined radius size ε_w. To obtain the tractable model for the above problem, the worst-case joint chance constraint in (6.3b) can be approximated by using the Conditional Value-at-Risk (CVaR) approximation [73]. Then, in practical settings, the nominal distribution can be defined as an empirical distribution constructed by the collected samples $\{\xi_m\}_{m=1}^{M}$ [17], and each sample has equal probability mass $\dfrac{1}{M}$. Afterward, we assume that constraint functions $g_{i=1,\ldots,w}(x,\xi)$ can be expressed as the form affine in uncertainty ($g_i(x,\xi) = h_i(x)^T\xi + h_i^0(x)$). Based on the above settings, the tractable model of the above Wasserstein DRCCP can be formulated as:

$$
\begin{aligned}
\min_{x,\beta,\kappa,z} \quad & f(x) \\
\text{s.t.} \quad & \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{M}\sum_{m=1}^{M} z_m\right] \leq 0 \\
& z_m \geq h_i^0(x) - \beta + h_i(x)^T\xi_m, \quad \forall i = 1,...,w, \forall m = 1,...,M \\
& z_m \geq 0, \quad \forall m = 1,...,M \\
& \|h_i(x)\|_* \leq \kappa, \quad \forall i = 1,...,w
\end{aligned} \tag{6.4}
$$

where β is the variable coming from the CVaR approximation. κ and z are dual variables coming from the derivation of the above model. $\|\cdot\|_*$ is a dual norm. More details about the derivation of the above Wasserstein DRCCP model are explained in [167].

6.3 Proposed algorithm

This research introduces an innovative algorithm that enhances DRCCP decision quality by generating a representative subset of the original data set and eliminating outliers as well as extreme samples. Through both subset generation and removal of outliers and extreme samples, this algorithm improves DRCCP solution quality by shielding a DRCCP model from samples that could potentially reduce its effectiveness. Since not all the outliers and extreme samples would have negative impacts on DRCCP solution quality, this algorithm only removes outliers and extreme samples causing over-conservatism. In addition, this algorithm can simultaneously ensure DRCCP solution feasibility. Moreover, this algorithm is compatible with arbitrary DRCCP models since it does not rely on any specific mathematical property of a DRCCP approach. The proposed algorithm is an iterative algorithm guided by the DRCCP, and it is named the DRCCP Solution Quality Enhancer (DSQE). The DSQE is exhibited below:

Algorithm 3 DRCCP Solution Quality Enhancer (DSQE)

```
 1: Input: $I$: superset (original data set in hand),
           $N$: subset size,
           $B$: number of $k$-means clustering repetitions,
           $T$: parameter used when trimming samples,
           $1 - \delta$: JCSP tolerance
 2: $I' \leftarrow I$                                          ▷ $I'$ is a copy of $I$
 3: JCSP $\leftarrow 1$                                        ▷ Initialize JCSP
 4: while JCSP $\geq 1 - \delta$ do                            ▷ JCSP is evaluated based on $I$
 5:     $C \leftarrow \emptyset^B$                              ▷ Initialize $C$ as an empty set
 6:     for $b = 1 : B$ do
 7:         $C_b \leftarrow$ $k$-means clustering based on $I'$  ▷ $C_b$ is the set of centers of $k$ clusters
                                                                ▷ $C_b$ is saved as an element in $C$
 8:     end for
 9:     Select $C_b$ having the lowest total sum of distances, as the subset $S$.
10:     Solve the DRCCP based on $S$ to get the optimal solution $x^*$.
11:     Set the worst constraint violation based on $x^*$ and $S$, as a threshold $\mathcal{V}$.
12:     Select the samples in $I'$ causing constraint violations (based on $x^*$) worse than $\mathcal{V}$,
        as a set $\mathcal{R}$.
13:     Select $T\%$ samples in $\mathcal{R}$, which cause worst $T\%$ constraint violations.
14:     Remove the samples selected in the previous step from $I'$.
15: end while
16: Output: $x_f^*$: final feasible optimal solution,
            $f_f^*$: optimal objective corresponding to $x_f^*$,
            $S_f^*$: subset $S$ corresponding to $x_f^*$,
            $\mathcal{R}^{all}$: all the samples removed from $I'$
```

The detailed explanation of the above algorithm is shown below: Assuming we have a large enough data set, referred to as the superset I, and user-defined parameters N, B, T, and $1 - \delta$, with N being smaller than the sample size of I. A copy of I, named I', is created for data trimming, while I is used to evaluate the JCSP of the optimal DRCCP decision, which serves as the stopping criterion for the DSQE. By doing so, since I is used for computing the JCSP and remains unchanged, we can guarantee that the latest DRCCP solution is feasible based on the original data set at hand. This is essential because we cannot be entirely confident that a certain sample is completely irrelevant for assessing solution feasibility. To start the algorithm, the stopping criterion JCSP is initially set to a large value, and then a while loop is used to remove samples that lead to overly conservative DRCCP decisions. Within the while loop, we begin by initializing an empty set called C. Next, we perform k-means clustering B times through a for loop. Note that $k = N$. The k-means clustering is implemented based on the normalized I' (mean-centered and scaled to unit variance) since data normalization can improve the accuracy and computational efficiency of the k-means clustering algorithm [178]. The clustering results are reversed back to the original scales (denormalization). For each iteration of the for loop, we store the set of centers of k clusters obtained from the k-means clustering as an element C_b in C. Since the initialization of the k-means clustering involves randomness, the obtained sets of centers in different for loop iterations are different. After the for loop, we select the set of centers that correspond to the lowest total sum of distances (i.e., the sum of distances between all data points and their assigned cluster centers) as the subset S. S containing N samples, is considered the most representative subset for the set I', from the multiple runs of the k-means clustering. Next, the optimal solution x^* is obtained by solving the DRCCP problem based on the selected subset S. We utilize the subset instead of the superset for solving the DRCCP due to the smaller size and great representativeness of the subset. Accordingly, the DSQE can ensure satisfactory computational efficiency while preserving most of the important information in I'. Also, solving the DRCCP using the subset helps prevent the solution from being affected by outliers and extreme samples, as the subset typically does not contain these types of data points. This statement is evidenced in Sections 6.4 and 6.5. After obtaining x^*, we evaluate the worst constraint violation over all data points in S, by computing the expression $\max\limits_{\xi_S \in S} \max\limits_i g_i(x^*, \xi_S)$. The obtained worst constraint violation is denoted as $\mathcal{V}$. Next, we calculate the constraint violations based on x^* and I' by calculating the expression $\max\limits_i g_i(x^*, \xi_{I'}), \xi_{I'} \in I'$, and select the samples in I' ($\xi_{I'}$) that correspond to constraint violations worse than $\mathcal{V}$. The set of these selected samples is denoted as $\mathcal{R}$ ($\max\limits_i g_i(x^*, \xi_{\mathcal{R}}) \geq \mathcal{V}$, $\xi_{\mathcal{R}} \in \mathcal{R}$). We then choose $T\%$ of the samples in $\mathcal{R}$ that cause the worst $T\%$ constraint violations, and remove them from I' to obtain updated I'. Accordingly, the samples leading to extreme constraint violations are removed, and these samples are considered potential outliers and extreme samples in the DSQE. These samples contribute to over-conservatism, as the DRCCP has to produce a highly conservative solution to ensure constraint satisfaction based on these samples causing extreme constraint violations. These samples can be also called overly conservative samples. By removing these samples, we can prevent their negative effects on the generation of the subset. The above process in the while loop is repeated until the stopping criterion JCSP evaluated based on I reaches the tolerance $1 - \delta$. Once the DSQE stops, we obtain the final feasible DRCCP optimal solution and its corresponding optimal objective and subset. Additionally, we also store all the removed samples as an output of the DSQE. The critical steps of the DSQE are summarized in the flowchart illustrated in Figure 6.1.

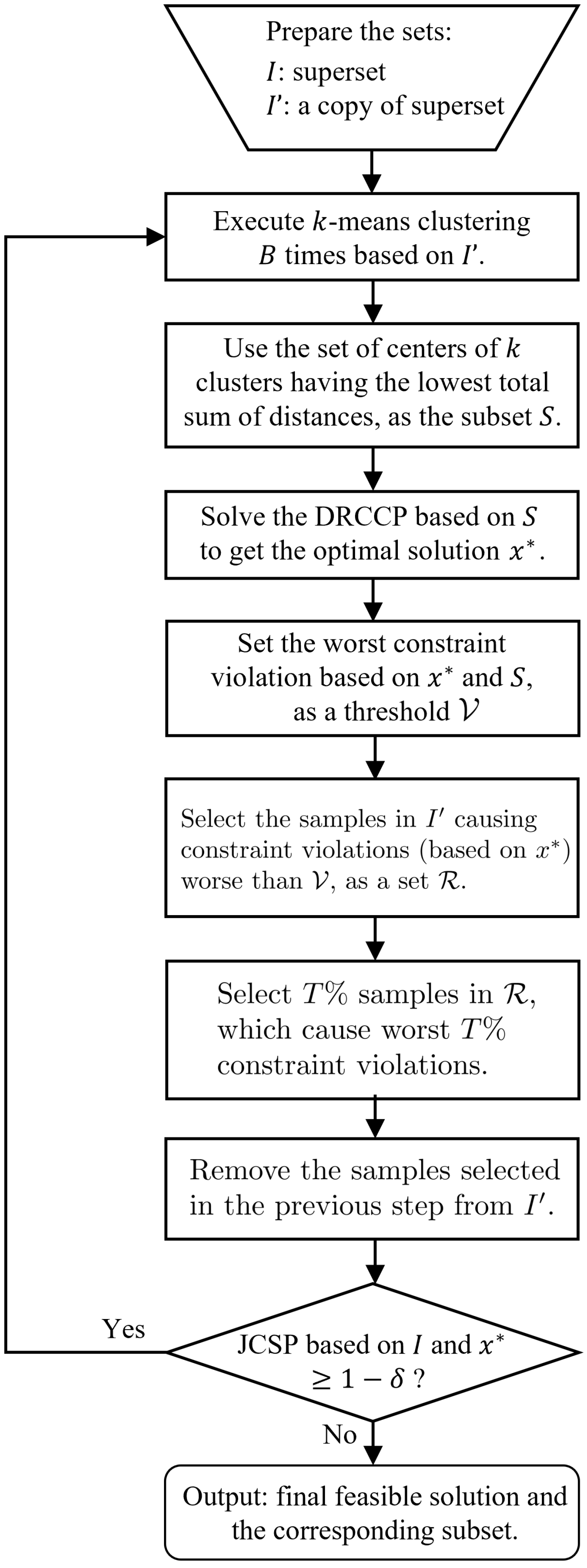

Prepare the sets:
I: superset
I': a copy of superset
Execute k-means clustering B times based on I'.
Use the set of centers of k clusters having the lowest total sum of distances, as the subset S.
Solve the DRCCP based on S to get the optimal solution x^*.
Set the worst constraint violation based on x^* and S, as a threshold $\mathcal{V}$
Select the samples in I' causing constraint violations (based on x^*) worse than $\mathcal{V}$, as a set $\mathcal{R}$.
Select $T\%$ samples in $\mathcal{R}$, which cause worst $T\%$ constraint violations.
Remove the samples selected in the previous step from I'.
JCSP based on I and x^* $\geq 1 - \delta$?
Yes
No
Output: final feasible solution and the corresponding subset.

Figure 6.1: Flowchart of the DSQE algorithm

In this work, the DSQE is applied to two numerical examples and a nonlinear process optimization problem to exhibit its efficacy. More details are demonstrated in the following sections.

6.4 Numerical examples

6.4.1 Numerical example 1

The first studied numerical example is taken from [14], and it is shown below:

$$\begin{aligned} \min_{x_1 \geq 0, x_2 \geq 0} \quad & x_1 + x_2 \\ \text{s.t.} \quad & Pr \left\{ \begin{array}{c} \xi_1 x_1 + x_2 \geq 7 \\ \xi_2 x_1 + x_2 \geq 4 \end{array} \right\} \geq 1 - \delta \end{aligned} \tag{6.5}$$

The random parameters ξ_1 and ξ_2 follow uniform distributions in [1,4] and $[\frac{1}{3}, 1]$, respectively. $1 - \delta$ is set to be 0.9. The true analytical solution is given as: $x_1^* = 3.1034$ and $x_2^* = 2.9655$. The true optimal objective is 6.0689.

A large data set of random parameters is generated from the true uniform distributions. This data set contains 10000 samples. In addition, 100 outliers outside the ranges of the true uniform distributions are added to the data set. The data set including regular samples and outliers is illustrated below:

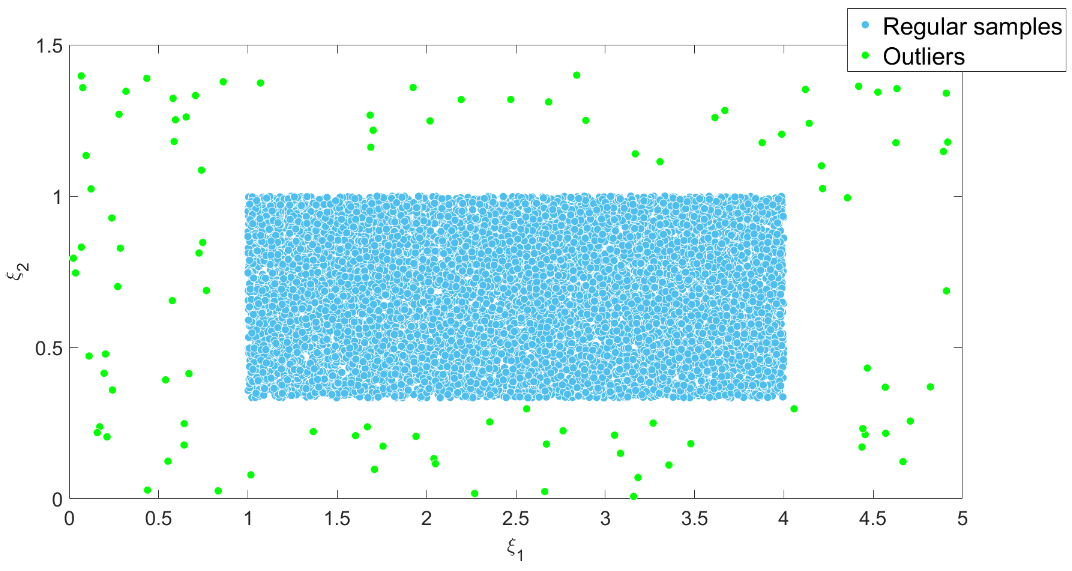


Figure 6.2: Regular samples and outliers (10100 samples in total)

The DSQE is executed for solving the above numerical example based on the data set composed of 10100 samples (10000 regular samples and 100 outliers). For this numerical problem, the parameters of the DSQE are set as follows: $N = 50$, $B = 10$, and $1 - \delta = 0.9$. Our experiment reveals that higher B and N decrease optimal solution variability but lengthen the computation time of DSQE. Based on our experiment, we choose $B = 10$ and $N = 50$ as optimal values for balancing DSQE's performance and computation time. To investigate the general DSQE performance of solving this example, the DSQE is implemented 100 times with respect to a certain value of T (T is one of the parameters in the DSQE) and a certain value of ε_w (ε_w is the radius size of the Wasserstein ambiguity set in the Wasserstein DRCCP). Meanwhile, as the comparisons, the Wasserstein DRCCP is implemented without using the DSQE, based on the entire 10100-sample superset, with respect to different values of ε_w. The obtained results are presented in Tables 6.1-6.3. Notably, all the outcomes presented in the Tables correspond to JCSP values above the required level of 0.9, as validated using the 10100-sample superset and a large test set consisting of 10^6 samples obtained from the true distribution. The large test set is employed purely for testing purposes. Accordingly, the DSQE is able to maintain the solution feasibility when solving this numerical problem.

Table 6.1: Means and standard deviations of the optimal objectives from the DSQE, with respect to different values of T and ε_w

T	Mean/Standard deviation/Average solution time		
	$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
100	6.3769/0.1224/3.6931	6.3669/0.1103/4.0631	6.3461/0.0760/4.6663
50	6.3039/0.0898/6.1075	6.2498/0.0660/7.2766	6.2203/0.0414/8.4829
5	6.2577/0.0665/11.5373	6.2134/0.0479/24.3228	6.1937/0.0205/51.2696

Table 6.2: Average numbers of samples removed by using the DSQE, with respect to different values of T and ε_w

T	Average number of removed samples		
	$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
100	54.84	144.12	570.14
50	163.99	336.67	742.15
5	189.59	381.72	812.90

Table 6.3: Optimal objectives from the Wasserstein DRCCP based on the entire 10100-sample set, with respect to different values of ε_w

Optimal objective/Solution time		
$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
6.6404/1.2430	6.6694/0.8791	6.9552/0.9234

According to Table 6.1, with respect to a fixed value of ε_w, decreasing T results in an improvement in the mean objective value (since this is a minimization problem, a smaller objective value is better) and a decrease in the standard deviation. This is because a smaller T value leads to the removal of fewer samples at each DSQE iteration, allowing the algorithm to exercise more caution when trimming samples and decreasing the probability of eliminating too many samples in one iteration. Consequently, premature termination of the algorithm is prevented, enabling the DSQE to perform more iterations and more accurately remove samples causing over-conservatism. Accordingly, the total number of samples removed by the DSQE is increased while decreasing T, as can be seen from Table 6.2. By eliminating more samples causing over-conservatism, the impact of these samples on the DSQE's final subset generation is reduced, enabling the DRCCP based on the final subset to produce a less conservative solution. Additionally, since the algorithm becomes more careful in choosing which samples to remove as T decreases, in the final DSQE iteration, more relevant and significant samples in the data set are preserved, which enables the final subset generation of the DSQE to be more stable. Therefore, the variability of solutions obtained from the DSQE is reduced. However, it should be noted that there is a trade-off involved in decreasing T. Because the DSQE conducts more iterations with reduced T, the average solution time of the DSQE increases, as can be observed in Table 6.1. On the other hand, with respect to a fixed value of T, increasing ε_w results in an improvement in the mean objective value and a decrease in the standard deviation. The reason for this is that a larger ε_w allows for a larger Wasserstein ambiguity set, which in turn may lead the Wasserstein DRCCP to be based on a worse worst-case distribution, resulting in a more conservative solution. Based on the above, since a more conservative DRCCP solution is generated in each DSQE iteration, the DSQE takes more iterations to reach the JCSP tolerance $1 - \delta$. As more iterations occur, over-conservative samples are increasingly eliminated, as observed in Table 6.2. As a result, the influence of overly conservative samples on the DSQE's final subset production is diminished. This enables the optimal solution attained from the DSQE to be more stable (lower variability) and less conservative. Apart from the above discussion, by comparing the results in Tables 6.1 and 6.3, regardless of the values of T and ε_w, the average optimal objectives obtained from the DSQE are all substantially superior to the optimal objectives obtained from the DRCCP using the entire 10100-sample set. This observation demonstrates that no matter the values of T and ε_w, the DSQE can effectively improve the solution quality for this example.

This paragraph discusses the DSQE's performance in trimming outliers and extreme samples for this numerical example. Figure 6.3 illustrates the data trimming results obtained after a single execution of the DSQE with $T = 5$ and $\varepsilon_w = 0.0001$.

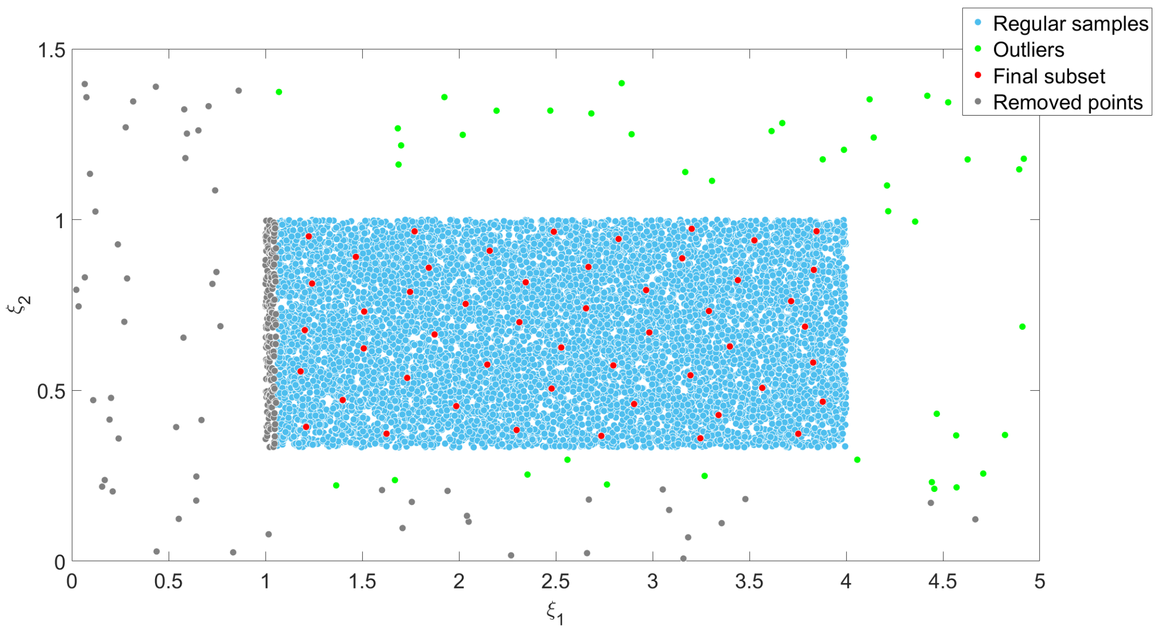


Figure 6.3: Data trimming result of example 1, after implementing the DSQE (only once) with $T = 5$ and $\varepsilon_w = 0.0001$

Figure 6.3 shows that the DSQE removes outliers and regular samples with much smaller values of ξ_1 or ξ_2, as they can cause more extreme constraint violations and worse objectives compared to other points. Specifically, the constraints in the joint chance constraint of Eq. 6.5 require their left-hand-side terms to be greater than or equal to two positive values, and since x_1 and x_2 are non-negative, smaller values of ξ_1 and ξ_2 result in larger x_1 and x_2 values needed to satisfy the constraints. As a consequence, minimizing the objective function $x_1 + x_2$ with samples having much smaller values of ξ_1 or ξ_2 leads to much worse constraint violations, prompting the removal of these samples to improve solution quality. On the other hand, some outliers with larger values of ξ_1 or ξ_2 are not removed as they do not lead to worse constraint violations than other outliers and samples. This shows that the DSQE only removes outliers and samples that have negative impacts on the DRCCP solution, rather than eliminating all outliers.

Finally, the final subset points in Figure 6.3 exclude outliers and extreme samples causing over-conservatism. Generated by the DSQE, these subset points guide the DRCCP execution, resulting in a much less conservative optimal solution compared to the one obtained from the DRCCP based on the entire original dataset (superset).

6.4.2 Numerical example 2

The second studied numerical example is modified from the example in [179], and it is presented below:

$$\begin{aligned} \max_{x_1\geq 0, x_2 \geq 0} \quad & x_1 + x_2 \\ \text{s.t.} \quad & Pr \left\{ \begin{array}{l} \frac{1}{3}x_1 + x_2 \leq \xi_1 \\ \frac{3}{2}x_1 + x_2 \leq \xi_2 \end{array} \right\} \geq 1 - \delta \end{aligned} \tag{6.6}$$

The random parameter vector $[\xi_1, \xi_2]$ follows a two-dimensional Gaussian mixture distribution with 6 components. The means of the six components are presented below:

$$\begin{pmatrix} 3 & 2.5 \\ 4 & 7 \\ 5.5 & 2.5 \\ 1.5 & 9 \\ 4 & 12.5 \\ 6.25 & 9 \end{pmatrix}$$

In the above matrix, the six rows correspond to the six components. The first column corresponds to ξ_1 and the second column corresponds to ξ_2. The covariances of the six components are exhibited below:

$$\text{Cov 1} = \begin{pmatrix} 0.5 & 0.0 \\ 0.0 & 1.0 \end{pmatrix} \quad \text{Cov 2} = \begin{pmatrix} 0.2 & -0.1 \\ -0.1 & 0.6 \end{pmatrix} \quad \text{Cov 3} = \begin{pmatrix} 0.2 & 0.0 \\ 0.0 & 0.8 \end{pmatrix}$$
$$\text{Cov 4} = \begin{pmatrix} 0.25 & 0.0 \\ 0.0 & 1.0 \end{pmatrix} \quad \text{Cov 5} = \begin{pmatrix} 0.2 & 0.1 \\ 0.1 & 0.8 \end{pmatrix} \quad \text{Cov 6} = \begin{pmatrix} 0.3 & -0.1 \\ -0.1 & 0.6 \end{pmatrix}$$

Cov 1 $\sim$ Cov 6 are the covariances of the 6 components.

A large data set of the random parameter vector is produced from the true Gaussian mixture distribution. This data set includes 10000 samples, and it is illustrated below:

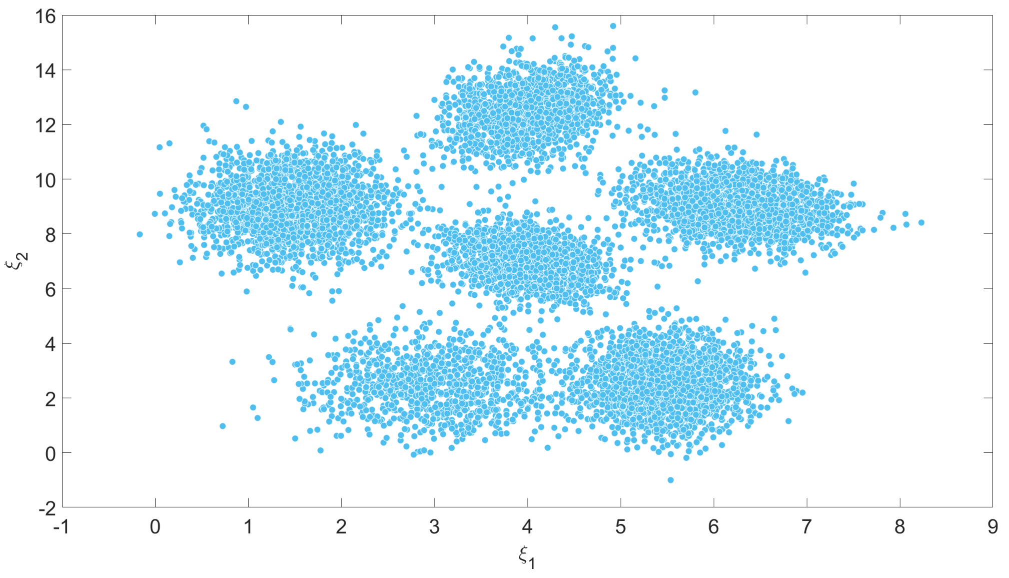


Figure 6.4: Samples from the Gaussian mixture distribution (10000 samples in total)

The DSQE is utilized to solve the given numerical example using a 10000-sample data set. For this example, we set the parameters in the DSQE as follows: $N = 50$, $B = 10$, and $1 - \delta = 0.9$. For this numerical example, our experiment determines that $B = 10$ and $N = 50$ are optimal values for balancing the DSQE's performance and computational efficiency. To evaluate the general DSQE performance for this example, we run the DSQE 100 times with respect to a certain value of T and a certain value of ε_w. For comparison, we also implement the Wasserstein DRCCP without the DSQE, using the complete 10000-sample superset, with respect to different values of ε_w. The results can be found in Tables 6.4 to 6.6. Importantly, all outcomes in Tables 6.4 to 6.6 have JCSP values satisfying the required 0.9 level, as verified by the 10000-sample superset and a large test set of 10^6 samples from the true distribution. The large test set is used only for testing purposes. Accordingly, the DSQE effectively maintains solution feasibility when addressing this numerical problem.

Table 6.4: Means and standard deviations of the optimal objectives from the DSQE, with respect to different values of T and ε_w

T	Mean/Standard deviation/Average solution time		
	$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
100	1.3033/0.1116/2.7769	1.3107/0.1054/2.9670	1.3268/0.0668/3.4022
50	1.3422/0.0584/3.5421	1.3516/0.0545/4.0610	1.3618/0.514/4.0865
5	1.3926/0.0371/18.7090	1.3943/0.0336/20.0894	1.3944/0.0293/23.1362

Table 6.5: Average numbers of samples removed by using the DSQE, with respect to different values of T and ε_w

T	Average number of removed samples		
	$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
100	159.73	175.67	229.56
50	177.04	199.75	271.93
5	259.60	282.30	345.74

Table 6.6: Optimal objectives from the Wasserstein DRCCP based on the entire 10000-sample set, with respect to different values of ε_w

Optimal objective/Solution time		
$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
1.0416/1.4581	1.0089/1.2073	0.9725/1.1862

The obtained results of this numerical example follow a similar trend and share underlying reasons with the first numerical example. Table 6.4 shows that for a fixed ε_w, decreasing T improves the mean objective value (since this is a maximization problem, a larger objective value is better). This occurs because a smaller T leads to the removal of fewer samples per DSQE iteration, allowing the algorithm to be more cautious about trimming data and avoid premature termination. As a result, the DSQE can conduct more iterations to eliminate more samples causing over-conservatism (can be seen from Table 6.5) to enhance the optimal solution. However, more iterations lead to increased computation time. On the other hand, for a fixed value of T, increasing ε_w leads to a better mean objective value. A larger ε_w permits a bigger Wasserstein ambiguity set, which may cause the Wasserstein DRCCP to generate a more conservative solution based on a worse worst-case distribution. As a result, since a more conservative DRCCP solution is generated in each DSQE iteration, the DSQE requires more iterations to reach the JCSP tolerance $1-\delta$. Due to more iterations, more samples causing over-conservatism are removed (can be seen from Table 6.5), which enables the DSQE to achieve a less conservative DRCCP solution in the final iteration. According to the above discussion and the results exhibited in Table 6.5, reducing T and increasing ε_w lead to more DSQE iterations and more samples causing over-conservatism to be removed. Thus, the DRCCP in the final DSQE iteration operates on a more stable subset, leading to a solution with reduced variability. In addition to the above discussion, a comparison of the results in Tables 6.4 and 6.6 reveals that, irrespective of the values of T and ε_w, the average optimal objectives achieved by the DSQE are all considerably better than those obtained from the DRCCP using the complete 10000-sample set. This finding illustrates the DSQE's effectiveness in enhancing solution quality for this example, regardless of the T and ε_w values.

This paragraph examines the DSQE's ability to trim samples for this numerical example. Figure 6.5 displays the data trimming outcome from a single DSQE execution using $T = 5$ and $\varepsilon_w = 0.0001$.

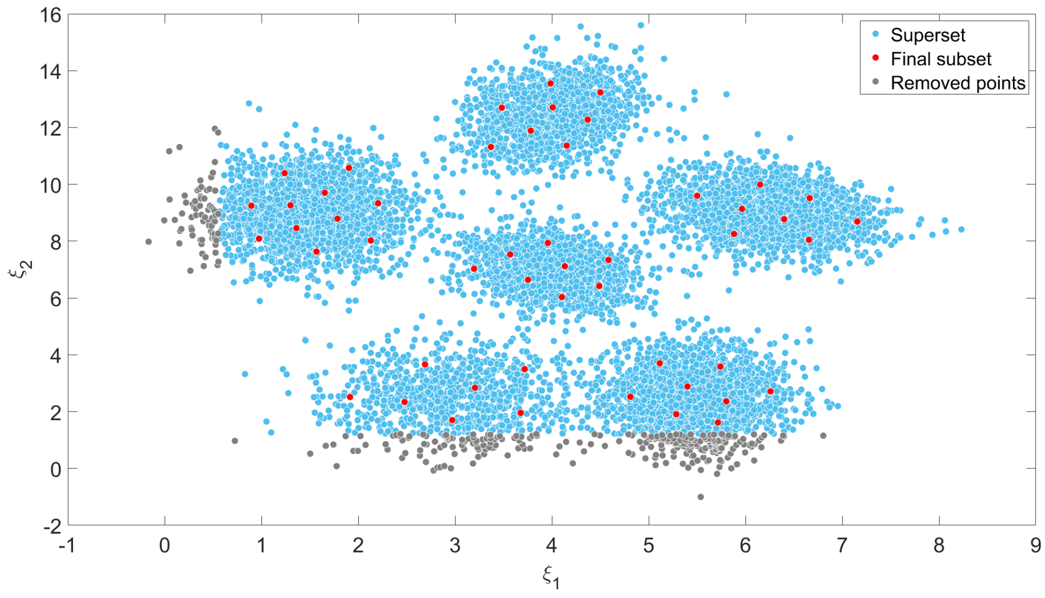


Figure 6.5: Data trimming result of example 2, after implementing the DSQE (only once) with $T = 5$ and $\varepsilon_w = 0.0001$

As can be seen from Figure 6.5, samples with extremely small values of ξ_1 or ξ_2 are removed. This is because those samples would cause more extreme constraint violations and result in a worse objective compared to other samples. More specifically, regarding the constraints in the joint chance constraint in Eq. 6.6, since the left-hand-side terms of the constraints are all positive and they have to be below two random parameters ξ_1 and ξ_2, smaller values of ξ_1 and ξ_2 would lead to smaller x_1 and x_2 values. Then, smaller values of ξ_1 or ξ_2 would lead to the smaller objective value which is equal to $x_1 + x_2$. Accordingly, samples with extremely small values of ξ_1 or ξ_2 would lead to extreme constraint violations when maximizing the objective $x_1 + x_2$, and thus those extreme samples would be removed by the DSQE to improve the solution quality.

Finally, the final subset illustrated in Figure 6.5 does not include outliers and extreme samples causing over-conservatism. This subset directs the DRCCP implementation, resulting in a significantly less conservative optimal solution compared to the one attained from the DRCCP based on the entire original superset.

6.5 Case study

This section applies the proposed DSQE to an alkylation process optimization under uncertainty. This process optimization problem is modified from Example 14.3 in [162] by introducing random parameters into the original process model. The simplified flowsheet of the studied alkylation process is shown in Figure 6.6.

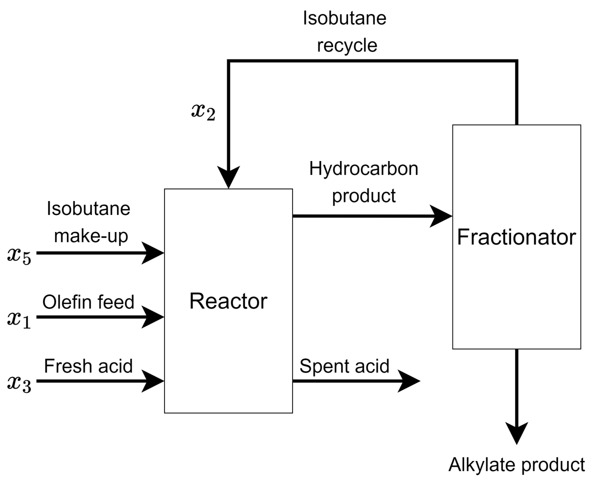


Figure 6.6: Flowsheet of the studied alkylation process

As can be seen from Figure 6.6, the alkylation process is composed of a reactor and a fractionator. The inputs of the reactor are an olefin stream (100% butane), a pure isobutane recycle stream, a 100% isobutane make-up stream, and acid catalyst. The spent acid is removed from the reactor. The product stream of the reactor is passed through a fractionator separating the isobutane and the alkylate product.

This process optimization aims to maximize the total profit determined by the alkylate product value (\$0.063/octane-barrel), olefin feed cost (\$5.04/barrel), isobutane recycle cost (\$0.035/barrel), acid addition cost (\$10.00/per thousand pounds), and isobutane makeup cost (\$3.36/barrel). The studied process optimization problem is given as:

$$\max_{x} \quad 0.063x_4x_7 - 5.04x_1 - 0.035x_2 - 10x_3 - 3.36x_5 \tag{6.7a}$$

$$\text{s.t.} \quad Pr\left\{ \begin{array}{l} x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2) - 0.89x_4 \geq 0 \\ -x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2) + 1.12x_4 \geq 0 \\ -\xi_4 + \xi_5 x_7 - 0.89x_{10} \geq 0 \\ \xi_4 - \xi_5 x_7 + 1.12x_{10} \geq 0 \end{array} \right\} \geq 0.9 \tag{6.7b}$$

$$86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_6 - 89) - 0.89x_7 \geq 0 \tag{6.7c}$$

$$-86.35 - 1.098x_8 + 0.038x_8^2 - 0.325(x_6 - 89) + 1.12x_7 \geq 0 \tag{6.7d}$$

$$35.82 - 0.222x_{10} - 0.89x_9 \geq 0 \tag{6.7e}$$

$$-35.82 + 0.222x_{10} + 1.12x_9 \geq 0 \tag{6.7f}$$

$$x_5 = 1.22x_4 - x_1 \tag{6.7g}$$

$$x_6 = \frac{98000x_3}{x_4x_9 + 1000x_3} \tag{6.7h}$$

$$x_8 = \frac{x_2 + x_5}{x_1} \tag{6.7i}$$

$$0 \leq x_1 \leq 2000, \;\; 0 \leq x_2 \leq 16000, \;\; 0 \leq x_3 \leq 120, \;\; 0 \leq x_4 \leq 5000 \tag{6.7j}$$

$$0 \leq x_5 \leq 2000, \;\; 85 \leq x_6 \leq 93, \;\; 90 \leq x_7 \leq 95, \;\; 3 \leq x_8 \leq 12 \tag{6.7k}$$

$$0.01 \leq x_9 \leq 4, \;\; 145 \leq x_{10} \leq 162 \tag{6.7l}$$

where $x_1 \sim x_{10}$ are decision variables as explained in Table 6.7. Each of them is associated with some physical bounds, as shown in Eqs. 6.7j$\sim$6.7l. Equation 6.7g defines the volumetric balance for the reactor. In Eq. 6.7g, the alkylate yield x_4 is equal to the sum of the olefin feed x_1 and the isobutane makeup x_5, with the volumetric shrinkage which is $0.22x_4$ subtracted from the sum: $x_4 = x_1 + x_5 - 0.22x_4 \Longrightarrow x_5 = 1.22x_4 - x_1$. The acid strength x_6 is calculated through Eq. 6.7h based on the assumption that the added acid (acid addition rate is x_3) has an acid strength of 98%. The external isobutane-to-olefin ratio x_8 is the sum of the isobutane recycle x_2 and the isobutane makeup x_5 divided by the olefin feed x_1, which is calculated via Eq. 6.7i. As to the first two constraints inside the joint chance constraint in Eq. 6.7b, the term $x_1(\xi_1 + \xi_2 x_8 - \xi_3 x_8^2)$ is the regression model for predicting x_4 by utilizing x_1 and x_8. In the last two constraints inside the joint chance constraint, the term $-\xi_4 + \xi_5 x_7$ is the regression model for predicting x_{10} with the use of x_7. The inequalities in the joint chance constraint restrict the percentage errors between the regression model predictions and the true values to the range 0.89 $\sim$ 1.12. $\xi_1 \sim \xi_5$ denote uncertain parameters in the regression models. They either follow normal distributions $\xi_1 \sim N(1.12, 0.028^2)$, $\xi_4 \sim N(133, 3.325^2)$, or uniform distributions $\xi_2 \sim U(0.1277, 0.1356)$, $\xi_3 \sim U(0.0065, 0.0069)$, $\xi_5 \sim U(2.91, 3.09)$. Inequalities 6.7c-6.7f are based on the regression model $86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_6 - 89)$ for predicting x_7, and the regression model $35.82 - 0.222x_{10}$ for predicting x_9, and these inequalities also constrain the percentage errors between the predictions and true values to the range 0.89 $\sim$ 1.12. The above optimization problem is a nonlinear joint chance-constrained optimization. All the settings for the above optimization are taken from [162], with the exception of the uncertain parameters and the percentage error tolerances for the regression models.

Table 6.7: Variables of the optimization model

Symbol	Variable	Unit
x_1	Olefin feed	barrels/day
x_2	Isobutane recycle	barrels/day
x_3	Acid addition rate	thousands of pounds/day
x_4	Alkylate yield	barrels/day
x_5	Isobutane makeup	barrels/day
x_6	Acid strength	weight percent
x_7	Motor octane number	–
x_8	External isobutane-to-olefin ratio	–
x_9	Acid dilution factor	–
x_{10}	F-4 performance number	–

The presented DSQE is implemented for solving the studied process optimization based on a 10000-sample data set. This data set is produced from the true distributions of the uncertain parameters. The hyper-parameters of the DSQE are set as follows, for this case study: $N = 50$, $B = 10$, and $1 - \delta = 0.9$. In this case study, our experiment finds that using $B = 10$ and $N = 50$ achieves the optimal balance between the DSQE's performance and computational efficiency. Again, the DSQE is also executed 100 times with respect to a certain value of T and a certain value of ε_w to investigate the general algorithm performance for this case study. The obtained results are exhibited in Tables 6.8 to 6.10. The JCSPs of the obtained optimal results are evaluated based on both the 10000-sample set and a large 10^6-sample test set. The evaluated JCSPs all satisfy the required value of 0.9. Thus, the

DSQE is able to ensure the solution feasibility for this case study.

Table 6.8: Means and standard deviations of the optimal objectives from the DSQE with respect to different values of T and ε_w

T	Mean/Standard deviation/Average solution time		
	$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
100	2581.4/20.1473/4.1314	2592.3/18.7523/5.3645	2598.6/11.2349/7.6514
50	2585.2/19.2823/4.4176	2600.0/17.1314/8.4847	2622.8/6.6900/11.1550
5	2611.0/12.1866/9.8458	2620.9/7.6046/14.0047	2626.1/5.8717/27.7169

Table 6.9: Average numbers of samples removed by using the DSQE, with respect to different values of T and ε_w

T	Average number of removed samples		
	$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
100	30.26	39.37	43.97
50	37.98	50.89	57.41
5	41.02	61.33	70.26

Table 6.10: Optimal objectives from the Wasserstein DRCCP based on the entire 10100-sample set, with respect to different values of ε_w

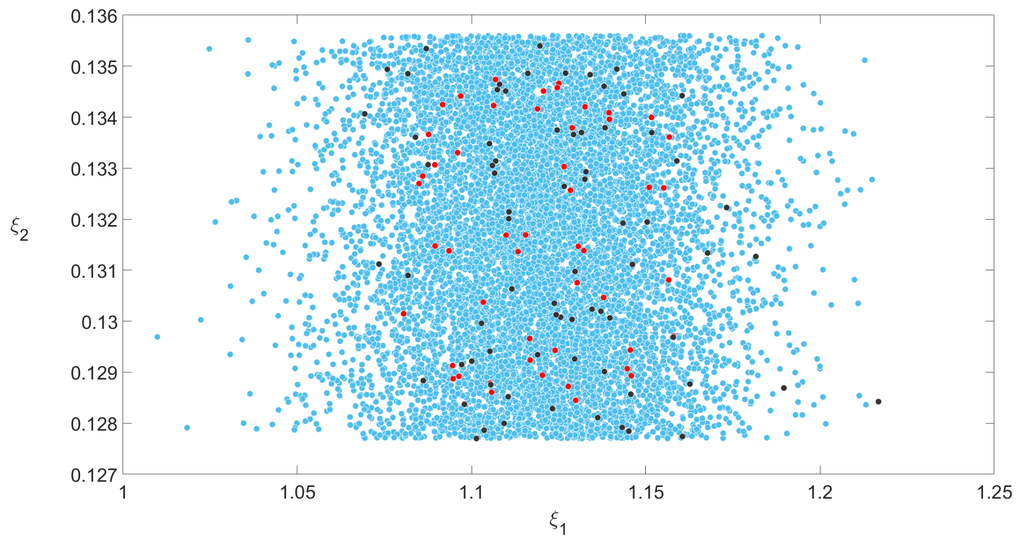
Optimal objective/Solution time		
$\varepsilon_w = 0.0001$	$\varepsilon_w = 0.001$	$\varepsilon_w = 0.01$
2466.5/11.1651	1926.9/11.9823	1660.3/22.242

The outcomes obtained in this case study exhibit a similar trend and share underlying reasons with the numerical examples in Section 6.4. Table 6.8 demonstrates that, for a fixed ε_w, a decrease in T results in a better mean objective value (since this case study is a maximization problem, a larger objective value is preferred). This improvement occurs because a smaller T removes fewer samples per DSQE iteration, allowing the algorithm to be more cautious in trimming data and avoiding premature termination. Consequently, the

DSQE can perform more iterations to eliminate samples contributing to over-conservatism (as seen in Table 6.9), thus enhancing the optimal solution. However, more iterations also lead to increased computation time. Conversely, for a fixed value of T, an increase in ε_w leads to a better mean objective value. A larger ε_w allows for a richer Wasserstein ambiguity set, causing the Wasserstein DRCCP to produce a more conservative solution based on a worse worst-case distribution. As a result, since each DSQE iteration generates a more conservative DRCCP solution, the DSQE requires more iterations to reach the JCSP tolerance $1-\delta$. Due to the increased number of iterations, more samples contributing to over-conservatism are removed (as seen in Table 6.9), enabling the DSQE to achieve a less conservative DRCCP solution in the final iteration. Moreover, according to the above discussion and the results exhibited in Table 6.9, reducing T and increasing ε_w lead to more DSQE iterations and the removal of more samples causing over-conservatism. This results in the DRCCP in the final DSQE iteration operating on a more stable subset and producing a solution with reduced variability. Furthermore, a comparison of the results in Tables 6.8 and 6.10 indicates that, regardless of the values of T and ε_w, the average optimal objectives achieved by the DSQE are significantly better than those obtained from the DRCCP using the complete 10000-sample set. This observation highlights the DSQE's effectiveness in improving solution quality for this case study, irrespective of the T and ε_w values.

The results illustrated in Figures 6.7-6.11 reveal that samples with higher ξ_4 values and lower ξ_5 values tend to be removed. However, it is more challenging to determine which samples, based on their $\xi_1 \sim \xi_3$ values, should be removed. This is because of the following reasons: ξ_4 and ξ_5 are related to the last two constraints in the joint chance constraint (JCC) in (6.7b), and these constraints are linear in both decision and random variables. Conversely, $\xi_1 \sim \xi_3$ are related to the first two constraints in the JCC, and these constraints are non-linear in the decision variables. Based on the above, ξ_4 and ξ_5 are involved in simpler constraints than $\xi_1 \sim \xi_3$. Consequently, the impact of ξ_4 and ξ_5 on constraint violations and solution conservatism is easier to discern than the effect of $\xi_1 \sim \xi_3$ on constraint violations and solution conservatism. Therefore, it is simpler to identify which samples should be removed based on their ξ_4 and ξ_5 values, as indicated by the results presented in Figures 6.7-6.11.

Finally, the final subset depicted in Figures 6.7-6.11 excludes outliers and extreme samples that lead to over-conservatism. This subset guides the DRCCP execution, which in turn produces a considerably less conservative optimal solution compared to the one gained from the DRCCP based on the full original superset.



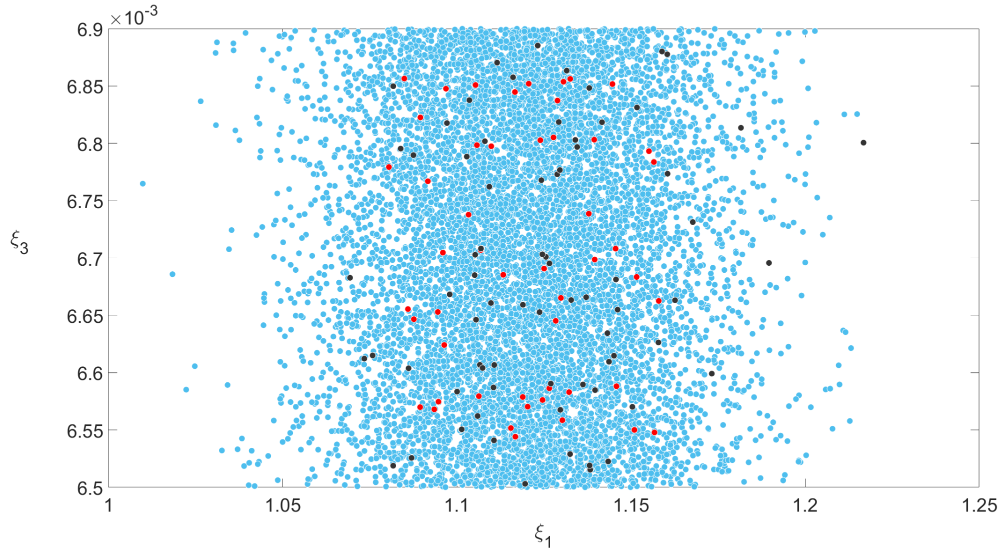


Figure 6.7: Data trimming result 1 of the case study, after implementing the DSQE (only once) with $T = 5$ and $\varepsilon_w = 0.0001$. Blue points: superset; Red points: final subset; Gray points: Removed points.

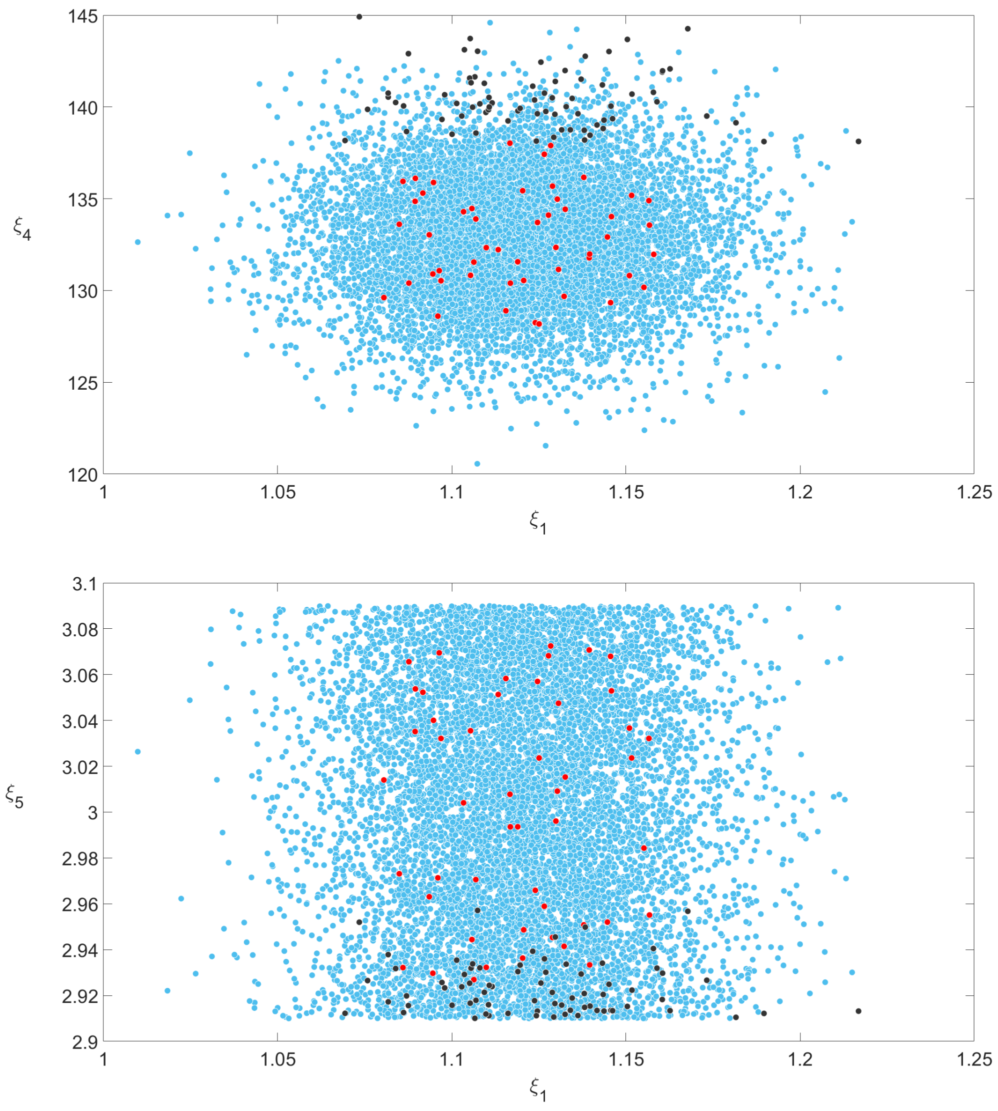


Figure 6.8: Data trimming result 2 of the case study, after implementing the DSQE (only once) with $T = 5$ and $\varepsilon_w = 0.0001$. Blue points: superset; Red points: final subset; Gray points: Removed points.

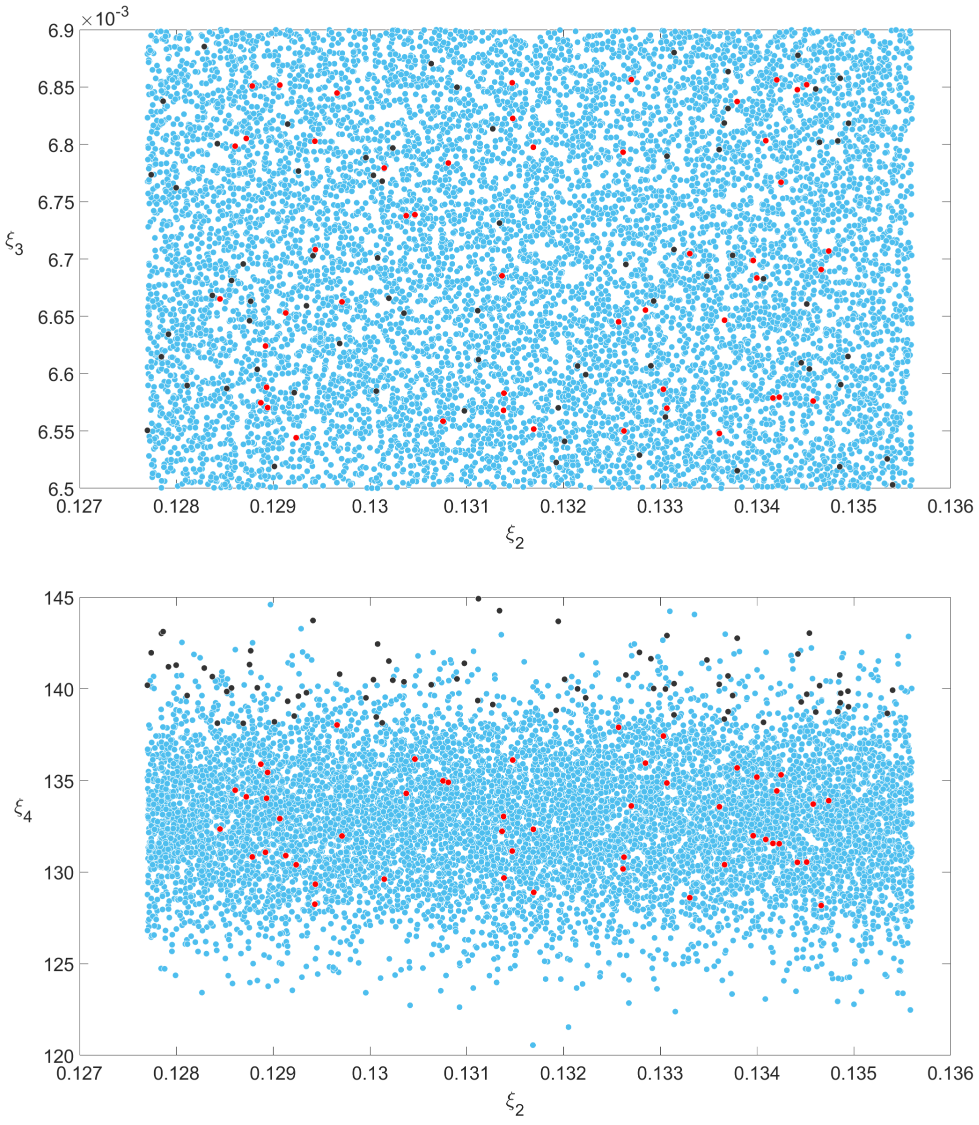


Figure 6.9: Data trimming result 3 of the case study, after implementing the DSQE (only once) with $T = 5$ and $\varepsilon_w = 0.0001$. Blue points: superset; Red points: final subset; Gray points: Removed points.

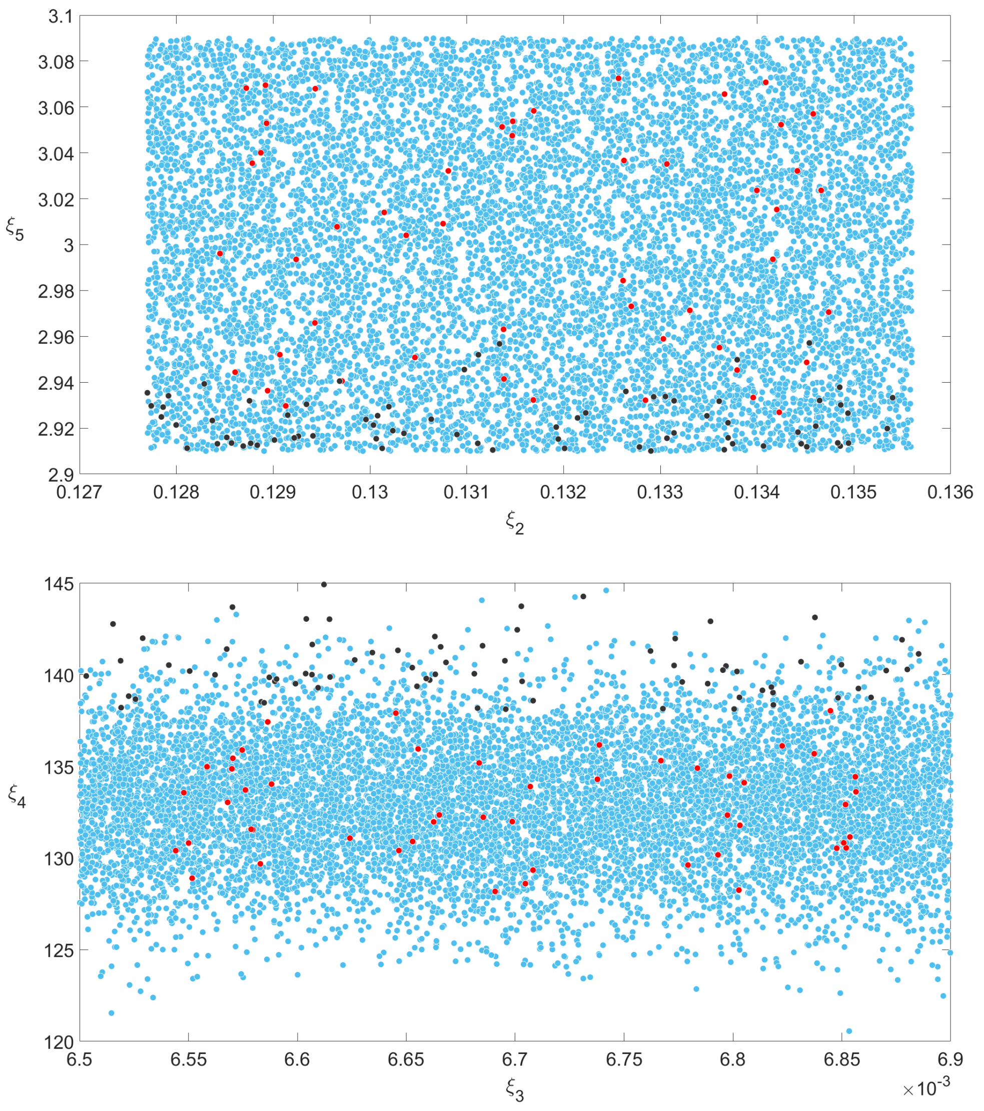
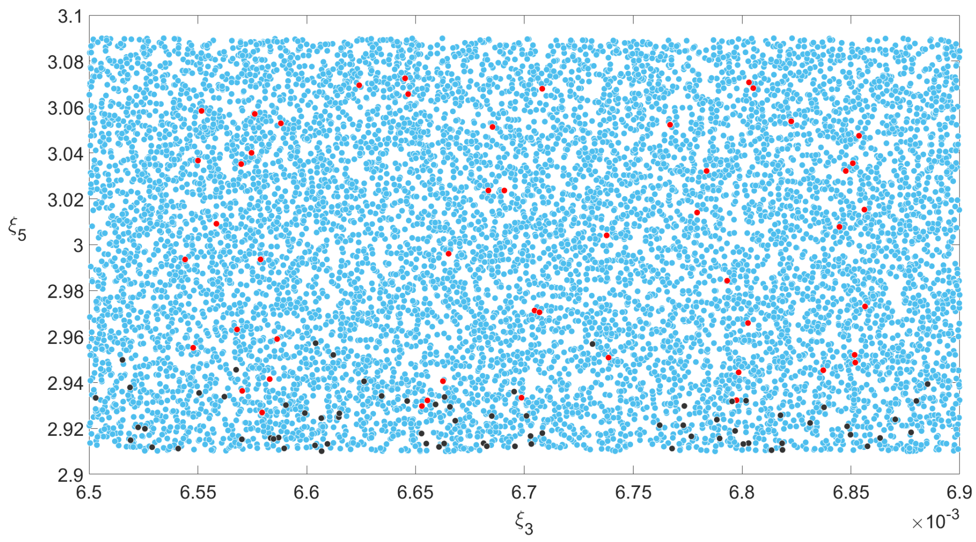


Figure 6.10: Data trimming result 4 of the case study, after implementing the DSQE (only once) with $T = 5$ and $\varepsilon_w = 0.0001$. Blue points: superset; Red points: final subset; Gray points: Removed points.



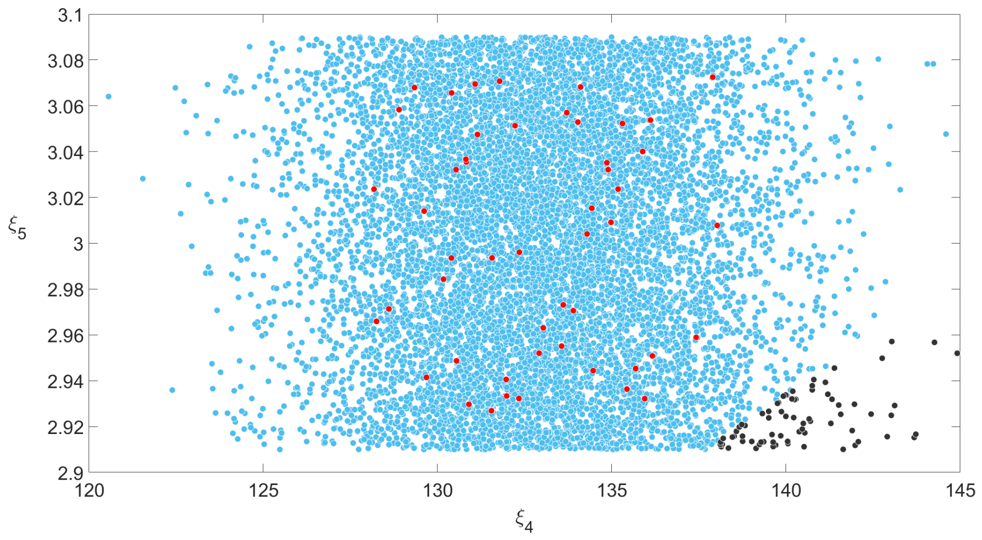


Figure 6.11: Data trimming result 5 of the case study, after implementing the DSQE (only once) with $T = 5$ and $\varepsilon_w = 0.0001$. Blue points: superset; Red points: final subset; Gray points: Removed points.

6.6 Conclusions

A novel and effective algorithm named DRCCP Solution Quality Enhancer (DSQE) is presented in this work, which is designed to enhance the quality of DRCCP solutions. This algorithm is compatible with various DRCCP methods since it does not rely on any specific mathematical property of a DRCCP approach. By generating a representative subset of the original data set and eliminating the samples contributing to extreme constraint violations, the DSQE algorithm can significantly improve the decision quality of DRCCP. In addition, the algorithm can simultaneously ensure the solution feasibility of DRCCP. The efficacy of the DSQE has been demonstrated through the application of two linear numerical examples and a nonlinear process optimization case study. The presented algorithm is capable of removing samples that negatively impact the decision quality, thereby enabling DRCCP to generate improved solutions for the mentioned applications. The obtained results indicate that while adjusting the hyperparameters of both the DSQE and the DRCCP model embedded in the DSQE may impact the DSQE's performance, the DSQE can still attain a significantly better optimal solution than the DRCCP (based on the original data set) without using the DSQE, irrespective of the selected hyper-parameter values.

For future research, it would be valuable to develop a more powerful technique to determine the subset representing the original superset to further enhance the DSQE performance and computational efficiency.

Chapter 7

Conclusion and Future work

7.1 Concluding remarks

This study aims to propose novel data-driven methods that can address optimization under uncertainty and overcome the limitations of existing approaches. To tackle the optimization problems involving surrogate model prediction uncertainty, we develop a new framework based on a computationally efficient mixture density network (MDN). By employing the MDN to address model prediction uncertainty, the presented approach can improve the solution robustness and achieve higher computational efficiency than the ensemble-based method from the literature. We also introduce an innovative neural network-based technique that can enable NP-hard joint chance-constrained problems (JCCPs) to be tractable and deterministic solvable. Furthermore, we extend this technique by replacing the neural network with a recurrent neural network (RNN), to solve joint chance-constrained stochastic optimal control problems (SOCPs). Our RNN-based method has higher computational efficiency and can achieve a better optimal control action compared to existing approaches for SOCPs. Subsequently, we present two new distributionally robust chance-constrained optimization (DRCCP) methods to solve JCCPs when the true distributions of uncertainty are not known. The first approach is based on the kernel ambiguity set, which can achieve better performance and require fewer assumptions than the popular Wasserstein-based method. The second method is based on the Sinkhorn ambiguity set, which can achieve better performance, require fewer assumptions, and hedge against more general families of uncertainty distributions than the widely-used Wasserstein DRCCP. Finally, we develop a novel iterative algorithm to improve the DRCCP solution quality by trimming outliers and extreme data samples in the used data set. This approach can significantly enhance the DRCCP solution quality while ensuring the solution feasibility. This algorithm is compatible with arbitrary DRCCP approaches.

7.2 Future work

This study could be further developed by exploring the following directions:

- The optimization approach proposed in Chapter 2 can be integrated with an adaptive sampling framework to enhance the solution quality. Also, using other modeling techniques for handling uncertainty such as the Gaussian process, the Bayesian neural network, etc., can be studied. Moreover, it would be worthwhile to investigate an intriguing direction of expanding the approach to neural networks with different types of activation functions.

- The RNN-based SOCP method presented in Section 3.2 can be combined with the DRCCP methods proposed in Chapters 4 and 5, to make the presented SOCP method also applies to the scenario that uncertainty distributions are unknown. In addition, a future endeavor would be to integrate the proposed SOCP method with a closed-loop prediction model employing control policy rather than action.

- A more general support discretization method for the kernel DRCCP presented in Chapter 4 should be studied. Also, conducting a more thorough examination of the particular application scenarios of different kernel DRCCP models could offer valuable avenues for future research. Additionally, exploring more efficient tuning steps for the kernel ambiguity set radius is worth considering. Furthermore, the proposed approach has the potential for extension to multi-stage adaptive optimization problems.

- It would be worthwhile to explore a more effective tuning algorithm for the hyperparameters of the Sinkhorn-based method presented in Chapter 5. Additionally, this Sinkhorn-based method can be extended to multi-stage adaptive optimization problems.

- The algorithm presented in Chapter 6 can be enhanced by developing more powerful methods to determine the subset representing the original superset.

- When the approaches proposed in Chapters 2-6 are applied to large-scale chemical processes, the result may be intricate optimization models that could incur substantial computational expenses. The scalability of these methodologies, hence, poses an area requiring diligent future exploration.

- Following the above discussion, despite the computational challenges, the integration of computationally efficient surrogate models into the optimization frameworks offers a promising solution. These surrogate models approximate large-scale chemical processes, substantially reducing computational load. Moreover, future work can focus on developing tailored algorithms that seamlessly integrate these surrogate models into optimization frameworks. Techniques could include iterative solutions or decomposition of the entire large problem into manageable sub-problems, resolved in a multi-stage manner for improved computational efficiency. Such algorithmic adaptations represent an exciting future direction for optimizing large-scale chemical processes.

Bibliography

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

[2] 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," *Computers & Chemical Engineering*, vol. 63, pp. 219–233, 2014.

[3] S. Zymler, D. Kuhn, and B. Rustem, "Distributionally robust joint chance constraints with second-order moment information." *Mathematical Programming*, vol. 137, 2013.

[4] R. Ji and M. A. Lejeune, "Data-driven distributionally robust chance-constrained optimization with wasserstein metric," *Journal of Global Optimization*, vol. 79, no. 4, pp. 779–811, 2021.

[5] S. Li, L. Feng, P. Benner, and A. Seidel-Morgenstern, "Using surrogate models for efficient optimization of simulated moving bed chromatography," *Computers & Chemical Engineering*, vol. 67, pp. 121–132, 2014.

[6] R. Srivastav, K. Sudheer, and I. Chaubey, "A simplified approach to quantifying predictive and parametric uncertainty in artificial neural network hydrologic models," *Water Resources Research*, vol. 43, no. 10, 2007.

[7] M. S. Khan and P. Coulibaly, "Assessing hydrologic impact of climate change with uncertainty estimates: Bayesian neural network approach," *Journal of Hydrometeorology*, vol. 11, no. 2, pp. 482–495, 2010.

[8] X. Zhang, F. Liang, B. Yu, and Z. Zong, "Explicitly integrating parameter, input, and structure uncertainties into bayesian neural networks for probabilistic hydrologic forecasting," *Journal of Hydrology*, vol. 409, no. 3-4, pp. 696–709, 2011.

[9] S. Alvisi and M. Franchini, "Fuzzy neural networks for water level and discharge forecasting with uncertainty," *Environmental Modelling & Software*, vol. 26, no. 4, pp. 523–537, 2011.

[10] A. Charnes and W. W. Cooper, "Management science," *Management Science*, vol. 6, no. 1, pp. 73–79, 1959.

[11] Y. Yang, P. Vayanos, and P. I. Barton, "Chance-constrained optimization for refinery blend planning under uncertainty," *Industrial & Engineering Chemistry Research*, vol. 56, no. 42, pp. 12 139–12 150, 2017.

[12] 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.

[13] 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.

[14] B. K. Pagnoncelli, S. Ahmed, and A. Shapiro, "Sample average approximation method for chance constrained programming: theory and applications," *Journal of Optimization Theory and Applications*, vol. 142, no. 2, pp. 399–416, 2009.

[15] S. Zhao and F. You, "Distributionally robust chance constrained programming with generative adversarial networks (gans)," *AIChE Journal*, vol. 66, no. 6, p. e16963, 2020.

[16] M. Staib and S. Jegelka, "Distributionally robust optimization and generalization in kernel methods," *Advances in Neural Information Processing Systems*, vol. 32, pp. 9134–9144, 2019.

[17] J. Wang, R. Gao, and Y. Xie, "Sinkhorn distributionally robust optimization," *arXiv preprint arXiv:2109.11926*, 2021.

[18] Q. Wu, K. Ding, and B. Huang, "Approach for fault prognosis using recurrent neural network," *Journal of Intelligent Manufacturing*, vol. 31, pp. 1621–1633, 2020.

[19] 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.

[20] 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.

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

[22] P. Li, H. Arellano-Garcia, and G. Wozny, "Chance constrained programming approach to process optimization under uncertainty," *Computers & chemical engineering*, vol. 32, no. 1-2, pp. 25–45, 2008.

[23] Y. Zhang, Y. Feng, and G. Rong, "Data-driven chance constrained and robust optimization under matrix uncertainty," *Industrial & Engineering Chemistry Research*, vol. 55, no. 21, pp. 6145–6160, 2016.

[24] S. Shen, "Using integer programming for balancing return and risk in problems with individual chance constraints," *Computers & Operations Research*, vol. 49, pp. 59–70, 2014.

[25] J. A. Paulson and A. Mesbah, "An efficient method for stochastic optimal control with joint chance constraints for nonlinear systems," *International Journal of Robust and Nonlinear Control*, vol. 29, no. 15, pp. 5017–5037, 2019.

[26] G. Guillén-Gosálbez and I. Grossmann, "A global optimization strategy for the environmentally conscious design of chemical supply chains under uncertainty in the damage assessment model," *Computers & Chemical Engineering*, vol. 34, no. 1, pp. 42–58, 2010.

[27] W. van Ackooij, R. Henrion, A. Möller, and R. Zorgati, "Joint chance constrained programming for hydro reservoir management," *Optimization and Engineering*, vol. 15, no. 2, pp. 509–531, 2014.

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

[29] G. C. Calafiore and L. E. Ghaoui, "On distributionally robust chance-constrained linear programs," *Journal of Optimization Theory and Applications*, vol. 130, pp. 1–22, 2006.

[30] P. Mohajerin Esfahani and D. Kuhn, "Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations," *Mathematical Programming*, vol. 171, no. 1-2, pp. 115–166, 2018.

[31] C. Villani, *Optimal transport: old and new.* Springer, 2009, vol. 338.

[32] A. Müller, "Integral probability metrics and their generating classes of functions," *Advances in applied probability*, pp. 429–443, 1997.

[33] Y. Yang and J. Kelly, "Efficient real time optimization using a data-driven piecewise affine model," *Computers & Chemical Engineering*, vol. 125, pp. 545–557, 2019.

[34] W. J. Lee, J. Na, K. Kim, C.-J. Lee, Y. Lee, and J. M. Lee, "Narx modeling for real-time optimization of air and gas compression systems in chemical processes," *Computers & Chemical Engineering*, vol. 115, pp. 262–274, 2018.

[35] J. O. Matias and J. Jäschke, "Using a neural network for estimating plant gradients in real-time optimization with modifier adaptation," *IFAC-PapersOnLine*, vol. 52-1, pp. 808–813, 2019.

[36] K. McBride and K. Sundmacher, "Overview of surrogate modeling in chemical process engineering," *Chemie Ingenieur Technik*, vol. 91, no. 3, pp. 228–239, 2019.

[37] I. Fahmi and S. Cremaschi, "Process synthesis of biodiesel production plant using artificial neural networks as the surrogate models," *Computers & Chemical Engineering*, vol. 46, pp. 105–123, 2012.

[38] F. N. Osuolale and J. Zhang, "Energy efficiency optimisation for distillation column using artificial neural network models," *Energy*, vol. 106, pp. 562–578, 2016.

[39] L. M. Ochoa-Estopier, M. Jobson, and R. Smith, "Operational optimization of crude oil distillation systems using artificial neural networks," *Computers & Chemical Engineering*, vol. 59, pp. 178–185, 2013.

[40] G. Artigue, A. Johannet, V. Borrell, and S. Pistre, "Flash flood forecasting in poorly gauged basins using neural networks: case study of the gardon de mialet basin (southern france)," *Natural Hazards and Earth System Sciences*, vol. 12, no. 11, pp. 3307–3324, 2012.

[41] K. Kasiviswanathan, R. Cibin, K. Sudheer, and I. Chaubey, "Constructing prediction interval for artificial neural network rainfall runoff models based on ensemble simulations," *Journal of Hydrology*, vol. 499, pp. 275–288, 2013.

[42] K. Kasiviswanathan and K. Sudheer, "Quantification of the predictive uncertainty of artificial neural network based river flow forecast models," *Stochastic Environmental Research and Risk Assessment*, vol. 27, no. 1, pp. 137–146, 2013.

[43] K. Kasiviswanathan, "Methods used for quantifying the prediction uncertainty of artificial neural network based hydrologic models," *Stochastic environmental research and risk assessment*, vol. 31, pp. 1659–1670, 2017.

[44] H. Zen and A. Senior, "Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2014, pp. 3844–3848.

[45] Z. Men, E. Yee, F.-S. Lien, D. Wen, and Y. Chen, "Short-term wind speed and power forecasting using an ensemble of mixture density neural networks," *Renewable Energy*, vol. 87, pp. 203–211, 2016.

[46] J. Zhang, J. Yan, D. Infield, Y. Liu, and F.-S. Lien, "Short-term forecasting and uncertainty analysis of wind turbine power based on long short-term memory network and gaussian mixture model," *Applied Energy*, vol. 241, pp. 229–244, 2019.

[47] R. Herzallah and D. Lowe, "A mixture density network approach to modelling and exploiting uncertainty in nonlinear control problems," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 2, pp. 145–158, 2004.

[48] M. Ahangar, M. Ghorbandoost, S. Sharma, and M. J. Smith, "Voice conversion based on a mixture density network," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA).* IEEE, 2017, pp. 329–333.

[49] J. Sarochar, I. Acharya, H. Riggs, A. Sundararajan, L. Wei, T. Olowu, and A. I. Sarwat, "Synthesizing energy consumption data using a mixture density network integrated with long short term memory," in *2019 IEEE Green Technologies Conference (GreenTech).* IEEE, 2019, pp. 1–4.

[50] A. Vakanski, J. Ferguson, and S. Lee, "Mathematical modeling and evaluation of human motions in physical therapy using mixture density neural networks," *Journal of Physiotherapy & Physical Rehabilitation*, vol. 1, no. 4, 2016.

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

[52] 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," *Industrial & Engineering Chemistry Research*, vol. 50, no. 18, pp. 10 567–10 603, 2011.

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

[54] R. Karimi, J. Cheng, and M. A. Lejeune, "A framework for solving chance-constrained linear matrix inequality programs," *INFORMS Journal on Computing*, vol. 33, no. 3, pp. 1015–1036, 2021.

[55] Y. Li, X. Yang, and F. Liu, "Robust adaptive beamforming for distributed radar based on covariance matrix reconstruction and steering vector estimation," in *2019 IEEE International Conference on Signal, Information and Data Processing (ICSIDP)*. IEEE, 2019, pp. 1–4.

[56] J. Cheng, C. Gicquel, and A. Lisser, "Partial sample average approximation method for chance constrained problems," *Optimization Letters*, vol. 13, pp. 657–672, 2019.

[57] Y. Zhang, S. Shen, and S. A. Erdogan, "Distributionally robust appointment scheduling with moment-based ambiguity set," *Operations Research Letters*, vol. 45, no. 2, pp. 139–144, 2017.

[58] X. Yu and S. Shen, "Multistage distributionally robust mixed-integer programming with decision-dependent moment-based ambiguity sets," *Mathematical Programming*, vol. 196, no. 1-2, pp. 1025–1064, 2022.

[59] S. Lee, H. Kim, and I. Moon, "A data-driven distributionally robust newsvendor model with a wasserstein ambiguity set," *Journal of the Operational Research Society*, pp. 1–19, 2020.

[60] Z. Lu, X. Xu, and Z. Yan, "Data-driven stochastic programming for energy storage system planning in high pv-penetrated distribution network," *International Journal of Electrical Power & Energy Systems*, vol. 123, p. 106326, 2020.

[61] Z. Chen, D. Kuhn, and W. Wiesemann, "Data-driven chance constrained programs over wasserstein balls," *arXiv preprint arXiv:1809.00210*, 2018.

[62] J. Le, X. Liao, L. Zhang, and T. Mao, "Distributionally robust chance constrained planning model for energy storage plants based on kullback–leibler divergence," *Energy Reports.*, vol. 7, pp. 5203–5213, 2021.

[63] Y. Chen, Q. Guo, H. Sun, Z. Li, W. Wu, and Z. Li, "A distributionally robust optimization model for unit commitment based on kullback–leibler divergence," *IEEE Transactions on Power Systems*, vol. 33, no. 5, pp. 5147–5160, 2018.

[64] Z. Hu and L. J. Hong, "Kullback-leibler divergence constrained distributionally robust optimization," *Available at Optimization Online.*, pp. 1695–1724, 2013.

[65] C. Ning and F. You, "Deep learning based distributionally robust joint chance constrained economic dispatch under wind power uncertainty," *IEEE Transactions on Power Systems*, vol. 37, no. 1, pp. 191–203, 2021.

[66] R. Gao and A. Kleywegt, "Distributionally robust stochastic optimization with wasserstein distance," *Mathematics of Operations Research*, 2022.

[67] W. Feng, Y. Feng, and Q. Zhang, "Multistage distributionally robust optimization for integrated production and maintenance scheduling," *AIChE Journal*, vol. 67, no. 9, p. e17329, 2021.

[68] C. Ge, L. Zhang, and Z. Yuan, "Distributionally robust optimization for the closed-loop supply chain design under uncertainty," *AIChE Journal*, vol. 68, no. 12, p. e17909, 2022.

[69] B. Liu, Q. Zhang, and Z. Yuan, "Two-stage distributionally robust optimization for maritime inventory routing," *Computers & Chemical Engineering*, vol. 149, p. 107307, 2021.

[70] W. Xie, "On distributionally robust chance constrained programs with wasserstein distance," *Mathematical Programming*, vol. 186, no. 1-2, pp. 115–155, 2021.

[71] Y. Zhang, S. Shen, and J. L. Mathieu, “Distributionally robust chance-constrained optimal power flow with uncertain renewables and uncertain reserves provided by loads,” *IEEE Transactions on Power Systems*, vol. 32, no. 2, pp. 1378–1388, 2016.

[72] N. Ho-Nguyen, F. Kılınç-Karzan, S. Küçükyavuz, and D. Lee, “Distributionally robust chance-constrained programs with right-hand side uncertainty under wasserstein ambiguity,” *Mathematical Programming*, pp. 1–32, 2021.

[73] B. Liu, Q. Zhang, X. Ge, and Z. Yuan, “Cvar-based approximations of wasserstein distributionally robust chance constraints with application to process scheduling,” *Industrial & Engineering Chemistry Research*, vol. 59, no. 20, pp. 9562–9574, 2020.

[74] Y. Gu and Y. Wang, “Distributionally robust chance-constrained programmings for non-linear uncertainties with wasserstein distance,” *arXiv preprint arXiv:2103.04790*, 2021.

[75] A. R. Hota, A. Cherukuri, and J. Lygeros, “Data-driven chance constrained optimization under wasserstein ambiguity sets,” in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 1501–1506.

[76] M.-C. Yue, D. Kuhn, and W. Wiesemann, “On linear optimization over wasserstein balls,” *Mathematical Programming*, vol. 195, no. 1-2, pp. 1107–1122, 2022.

[77] M. Chambers and C. Mount-Campbell, “Process optimization via neural network metamodeling,” *International Journal of Production Economics*, vol. 79, no. 2, pp. 93–100, 2002.

[78] C. A. O. Nascimento, R. Giudici, and R. Guardani, “Neural network based approach for optimization of industrial chemical processes,” *Computers & Chemical Engineering*, vol. 24, no. 9-10, pp. 2303–2314, 2000.

[79] M. M. Hamed, M. G. Khalafallah, and E. A. Hassanien, “Prediction of wastewater treatment plant performance using artificial neural networks,” *Environmental Modelling & Software*, vol. 19, no. 10, pp. 919–928, 2004.

[80] J. Michalopoulos, S. Papadokonstadakis, G. Arampatzis, and A. Lygeros, “Modelling of an industrial fluid catalytic cracking unit using neural networks,” *Chemical Engineering Research and Design*, vol. 79, no. 2, pp. 137–142, 2001.

[81] P. Rai, G. Majumdar, S. DasGupta, and S. De, "Modeling the performance of batch ultrafiltration of synthetic fruit juice and mosambi juice using artificial neural network," *Journal of Food Engineering*, vol. 71, no. 3, pp. 273–281, 2005.

[82] B. Grimstad and H. Andersson, "Relu networks as surrogate models in mixed-integer linear programs," *Computers & Chemical Engineering*, vol. 131, p. 106580, 2019.

[83] D. J. MacKay, "Bayesian interpolation," *Neural Computation*, vol. 4, no. 3, pp. 415–447, 1992.

[84] R. M. Neal, *Bayesian learning for neural networks.* Springer Science & Business Media, 2012, vol. 118.

[85] M. S. Khan and P. Coulibaly, "Bayesian neural network for rainfall-runoff modeling," *Water Resources Research*, vol. 42, no. 7, 2006.

[86] X. Zhang, F. Liang, R. Srinivasan, and M. Van Liew, "Estimating uncertainty of streamflow simulation using bayesian neural networks," *Water Resources Research*, vol. 45, no. 2, 2009.

[87] Y.-K. Tung and B. C. Yen, *Hydrosystems engineering uncertainty analysis.* McGraw-Hill New York, 2005.

[88] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 42, no. 1, pp. 55–61, 2000.

[89] C. M. Bishop, "Mixture density networks," 1994.

[90] R. Balestriero and R. Baraniuk, "Mad max: Affine spline insights into deep learning," *arXiv preprint arXiv:1805.06576*, 2018.

[91] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. S. Dickstein, "On the expressive power of deep neural networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70.* JMLR. org, 2017, pp. 2847–2854.

[92] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[93] N. Ketkar, "Introduction to pytorch," in *Deep learning with python.* Springer, 2017, pp. 195–208.

[94] M. H. Beale, M. T. Hagan, and H. B. Demuth, "Neural network toolbox user's guide," *The MathWorks*, 2010.

[95] M. Fischetti and J. Jo, "Deep neural networks and mixed integer linear optimization," *Constraints*, vol. 23, no. 3, pp. 296–309, 2018.

[96] T. P. Runarsson, "Approximating probabilistic constraints for surgery scheduling using neural networks," in *International Conference on Machine Learning, Optimization, and Data Science.* Springer, 2019, pp. 643–654.

[97] G. Wu, B. Say, and S. Sanner, "Scalable nonlinear planning with deep neural network learned transition models," *arXiv preprint arXiv:1904.02873*, 2019.

[98] S. Choi, K. Lee, S. Lim, and S. Oh, "Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling," in *2018 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, 2018, pp. 6915–6922.

[99] D. S. Abrams and J. M. Prausnitz, "Statistical thermodynamics of liquid mixtures: a new expression for the excess gibbs energy of partly or completely miscible systems," *AIChE Journal*, vol. 21, no. 1, pp. 116–128, 1975.

[100] W. Parker and J. Prados, "Analog computer design of an ethylene glycol system," *Chem. Eng. Prog.*, vol. 60, no. 6, pp. 74–78, 1964.

[101] O. Kahrs and W. Marquardt, "The validity domain of hybrid models and its application in process optimization," *Chemical Engineering and Processing: Process Intensification*, vol. 46, no. 11, pp. 1054–1066, 2007.

[102] M. Wendt, P. Li, and G. Wozny, "Nonlinear chance-constrained process optimization under uncertainty," *Industrial & Engineering Chemistry Research*, vol. 41, no. 15, pp. 3621–3629, 2002.

[103] B. You, E. Esche, J. Weigert, and J.-U. Repke, "Joint chance constraint approach based on data-driven models for optimization under uncertainty applied to the williams-otto process," in *Computer Aided Chemical Engineering.* Elsevier, 2021, vol. 50, pp. 523–528.

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

[105] W. Chen, M. Sim, J. Sun, and C.-P. Teo, "From cvar to uncertainty set: Implications in joint chance-constrained optimization," *Operations Research*, vol. 58, no. 2, pp. 470–485, 2010.

[106] L. J. Hong, Z. Huang, and H. Lam, "Approximating data-driven joint chance-constrained programs via uncertainty set construction," in *2016 Winter Simulation Conference (WSC).* IEEE, 2016, pp. 389–400.

[107] J. Luedtke and S. Ahmed, "A sample approximation approach for optimization with probabilistic constraints," *SIAM Journal on Optimization*, vol. 19, no. 2, pp. 674–699, 2008.

[108] S. Sonoda and N. Murata, "Neural network with unbounded activation functions is universal approximator," *Applied and Computational Harmonic Analysis*, vol. 43, no. 2, pp. 233–268, 2017.

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

[110] A. Nemirovski and A. Shapiro, "Convex approximations of chance constrained programs," *SIAM Journal on Optimization*, vol. 17, no. 4, pp. 969–996, 2007.

[111] 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.

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

[113] 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.

[114] M. Cannon, B. Kouvaritakis, S. V. Raković, and Q. Cheng, "Stochastic tubes in model predictive control with probabilistic constraints," *IEEE Transactions on Automatic Control*, vol. 56, no. 1, pp. 194–200, 2011.

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

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

[117] G. C. Calafiore and L. Fagiano, "Robust model predictive control via scenario optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 1, pp. 219–224, 2012.

[118] H. Arellano-Garcia and G. Wozny, "Chance constrained optimization of process systems under uncertainty: I. strict monotonicity," *Computers & Chemical Engineering*, vol. 33, no. 10, pp. 1568–1583, 2009.

[119] B. Bhadriraju, A. Narasingam, and J. S.-I. Kwon, "Machine learning-based adaptive model identification of systems: Application to a chemical process," *Chemical Engineering Research and Design*, vol. 152, pp. 372–383, 2019.

[120] 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," *Journal of Process Control*, vol. 86, pp. 30–43, 2020.

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

[122] 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 Transactions on Neural Networks*, vol. 20, no. 1, pp. 45–60, 2008.

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

[124] 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.

[125] 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.

[126] 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.

[127] 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.

[128] 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.

[129] G. Steinbrecher and W. T. Shaw, "Quantile mechanics," *European Journal of Applied Mathematics*, vol. 19, no. 2, pp. 87–112, 2008.

[130] 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 Transactions on Automatic Control*, vol. 60, no. 1, pp. 46–58, 2014.

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

[132] A. Prékopa, "Convexity theory of probabilistic constrained problems," in *Stochastic Programming.* Springer, 1995, pp. 301–317.

[133] W. Wiesemann, D. Kuhn, and M. Sim, "Distributionally robust convex optimization," *Operations Research*, vol. 62, no. 6, pp. 1358–1376, 2014.

[134] H. Sun and H. Xu, "Convergence analysis for distributionally robust optimization and equilibrium problems," *Mathematics of Operations Research*, vol. 41, no. 2, pp. 377–401, 2016.

[135] C. Ning and F. You, "Data-driven wasserstein distributionally robust optimization for biomass with agricultural waste-to-energy network design under uncertainty," *Applied Energy*, vol. 255, p. 113857, 2019.

[136] E. Delage and Y. Ye, "Distributionally robust optimization under moment uncertainty with application to data-driven problems," *Operations Research*, vol. 58, no. 3, pp. 595–612, 2010.

[137] J. Goh and M. Sim, "Distributionally robust optimization and its tractable approximations," *Operations Research*, vol. 58, no. 4-part-1, pp. 902–917, 2010.

[138] D. Wozabal, "A framework for optimization under ambiguity," *Annals of Operations Research*, vol. 193, no. 1, pp. 21–47, 2012.

[139] R. T. Rockafellar, S. Uryasev *et al.*, "Optimization of conditional value-at-risk," *Journal of Risk*, vol. 2, pp. 21–42, 2000.

[140] K. Muandet, K. Fukumizu, B. Sriperumbudur, and B. Schölkopf, "Kernel mean embedding of distributions: A review and beyond," *arXiv preprint arXiv:1605.09522*, 2016.

[141] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 723–773, 2012.

[142] J.-J. Zhu, W. Jitkrittum, M. Diehl, and B. Schölkopf, "Kernel distributionally robust optimization," *arXiv preprint arXiv:2006.06981*, 2020.

[143] J.-J. Zhu and W. Jitkrittum, "Worst-case risk quantification under distributional ambiguity using kernel mean embedding in moment problem," in *2020 59th IEEE Conference on Decision and Control (CDC).* IEEE, 2020, pp. 3457–3463.

[144] J. M. Phillips and S. Venkatasubramanian, "A gentle introduction to the kernel distance," *arXiv preprint arXiv:1103.1625*, 2011.

[145] A. Berlinet and C. Thomas-Agnan, *Reproducing kernel Hilbert spaces in probability and statistics.* Springer Science & Business Media, 2011.

[146] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley, "Reproducing kernel hilbert space, mercer's theorem, eigenfunctions, nystr\" om method, and use of kernels in machine learning: Tutorial and survey," *arXiv preprint arXiv:2106.08443*, 2021.

[147] J.-J. Zhu, K. Muandet, M. Diehl, and B. Schölkopf, "A new distribution-free concept for representing, comparing, and propagating uncertainty in dynamical systems with

kernel probabilistic programming," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 7240–7247, 2020.

[148] A. Smola, A. Gretton, L. Song, and B. Schölkopf, "A hilbert space embedding for distributions," in *International Conference on Algorithmic Learning Theory.* Springer, 2007, pp. 13–31.

[149] B. K. Sriperumbudur, K. Fukumizu, and G. R. Lanckriet, "Universality, characteristic kernels and rkhs embedding of measures." *The Journal of Machine Learning Research*, vol. 12, no. 7, 2011.

[150] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, 2006.

[151] S. Zhu and M. Fukushima, "Worst-case conditional value-at-risk with application to robust portfolio management," *Operations Research*, vol. 57, no. 5, pp. 1155–1168, 2009.

[152] D. Garreau, W. Jitkrittum, and M. Kanagawa, "Large sample analysis of the median heuristic," *arXiv preprint arXiv:1707.07269*, 2017.

[153] H. Lam and Y. Zeng, "Complexity-free generalization via distributionally robust optimization," *arXiv preprint arXiv:2106.11180*, 2021.

[154] T. Lapteva, N. Ziyatdinov, and I. Emel'yanov, "Chemical process design taking into account joint chance constraints," *Theoretical Foundations of Chemical Engineering*, vol. 54, pp. 145–156, 2020.

[155] K. P. Halemane and I. E. Grossmann, "Optimal process design under uncertainty," *AIChE Journal*, vol. 29, no. 3, pp. 425–433, 1983.

[156] Y. Cho and L. Saul, "Kernel methods for deep learning," *Advances in Neural Information Processing Systems*, vol. 22, pp. 342–350, 2009.

[157] C.-C. Cheng and B. Kingsbury, "Arccosine kernels: Acoustic modeling with infinite neural networks," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2011, pp. 5200–5203.

[158] A. Afzal, N. K. Nair, and S. Asharaf, "Deep kernel learning in extreme learning machines," *Pattern Analysis and Applications*, vol. 24, no. 1, pp. 11–19, 2021.

[159] Y. Cho and L. K. Saul, "Analysis and extension of arc-cosine kernels for large margin classification," *arXiv preprint arXiv:1112.3712*, 2011.

[160] Y. Cho, "Kernel methods for deep learning," Ph.D. dissertation, UNIVERSITY OF CALIFORNIA, SAN DIEGO, 2012.

[161] Y. Lyu, "Spherical structured feature maps for kernel approximation," in *International Conference on Machine Learning*. PMLR, 2017, pp. 2256–2264.

[162] T. F. Edgar, D. M. Himmelblau, L. S. Lasdon *et al.*, *Optimization of chemical processes*. McGraw Hill, 2001.

[163] M. T. Kelley, R. Baldick, and M. Baldea, "Demand response scheduling under uncertainty: Chance-constrained framework and application to an air separation unit," *AIChE Journal*, vol. 66, no. 9, p. e16273, 2020.

[164] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[165] A. Pichler and M. Weinhardt, "The nested sinkhorn divergence to learn the nested distance," *Computational Management Science*, vol. 19, no. 2, pp. 269–293, 2022.

[166] V. Chandrasekaran and P. Shah, "Relative entropy optimization and its applications," *Mathematical Programming*, vol. 161, no. 1, pp. 1–32, 2017.

[167] S.-B. Yang and Z. Li, "Kernel distributionally robust chance-constrained process optimization," *Computers & Chemical Engineering*, vol. 165, p. 107953, 2022.

[168] C. Villani and C. Villani, "The wasserstein distances," *Optimal Transport: Old and New*, pp. 93–111, 2009.

[169] W. Li and Y. Wang, "A robust supervised subspace learning approach for output-relevant prediction and detection against outliers," *Journal of Process Control*, vol. 106, pp. 184–194, 2021.

[170] S. Yin, G. Wang, and X. Yang, "Robust pls approach for kpi-related prediction and diagnosis against outliers and missing data," *International Journal of Systems Science*, vol. 45, no. 7, pp. 1375–1382, 2014.

[171] H. Wang, M. J. Bah, and M. Hammad, “Progress in outlier detection techniques: A survey,” *IEEE Access*, vol. 7, pp. 107 964–108 000, 2019.

[172] A. Smiti, “A critical overview of outlier detection methods,” *Computer Science Review*, vol. 38, p. 100306, 2020.

[173] X. Xia, X. Pan, N. Li, X. He, L. Ma, X. Zhang, and N. Ding, “Gan-based anomaly detection: a review,” *Neurocomputing*, 2022.

[174] C. Shang and F. You, “Distributionally robust optimization for planning and scheduling under uncertainty,” *Computers & Chemical Engineering*, vol. 110, pp. 53–68, 2018.

[175] R. Zhai, C. Dan, Z. Kolter, and P. Ravikumar, “Doro: Distributional and outlier robust optimization,” in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 345–12 355.

[176] N. Jiang and W. Xie, “Dfo: A framework for data-driven decision-making with endogenous outliers,” 2022.

[177] H. Rahimian, G. Bayraksan, and T. Homem-de Mello, “Identifying effective scenarios in distributionally robust stochastic programs with total variation distance,” *Mathematical Programming*, vol. 173, no. 1-2, pp. 393–430, 2019.

[178] D. Virmani, S. Taneja, and G. Malhotra, “Normalization based k means clustering algorithm,” *arXiv preprint arXiv:1503.00900*, 2015.

[179] B. A. Calfa, I. E. Grossmann, A. Agarwal, S. J. Bury, and J. M. Wassick, “Data-driven individual and joint chance-constrained optimization via kernel smoothing,” *Computers & Chemical Engineering*, vol. 78, pp. 51–69, 2015.

[180] C. Villani, *Optimal transport: old and new.* Springer Science & Business Media, 2008, vol. 338.

[181] R. G. Bartle, *The elements of integration and Lebesgue measure.* John Wiley & Sons, 2014.

Appendix

A1 Chapter 2 supplementary materials

A1.1 Illustrating example in Chapter 2

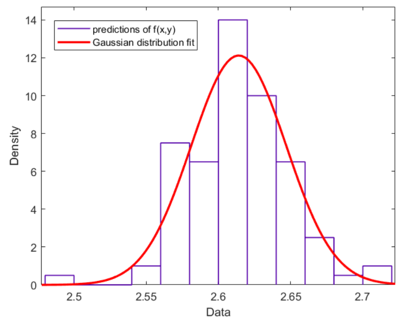
The MILP optimization model with MDN embedded for the illustrating example is given as:

$$
\begin{aligned}
\min \quad & \mu_{\hat{f}} + \lambda \sigma_{\hat{f}} && && \text{(A1a)}\\
\text{s.t.} \quad & 2 - \mu_{\hat{g}_1} \geq \phi^{-1}(1-\varepsilon)\sigma_{\hat{g}_1}, \quad 25 - \mu_{\hat{g}_2} \geq \phi^{-1}(1-\varepsilon)\sigma_{\hat{g}_2} && && \text{(A1b)}\\
& -5.3 \leq x \leq -0.6, \quad 0.6 \leq y \leq 5.3 && && \text{(A1c)}\\
& X_i = [x, y] && i = 1, 2 && \text{(A1d)}\\
& a_j^1 = \sum_{i=1}^{4} W_{ij}^1 X_i + b_j^1, \quad H_j^1 \geq a_j^1, \quad 0 \leq H_j^1 \leq M && j = 1, ..., 50 && \text{(A1e)}\\
& H_j^1 \leq M(1 - dz_j^1), \quad H_j^1 \leq a_j^1 + M(1 - dl_j^1) && && \text{(A1f)}\\
& dz_j^1 + dl_j^1 = 1, \quad dz_j^1, dl_j^1 \in 0, 1 && && \text{(A1g)}\\
& a_m^2 = \sum_{j=1}^{50} W_{jm}^2 H_j^1 + b_m^2, \quad H_m^2 \geq a_m^2, \quad 0 \leq H_m^2 \leq M && m = 1, ..., 30 && \text{(A1h)}\\
& H_m^2 \leq M(1 - dz_m^2), \quad H_m^2 \leq a_m^2 + M(1 - dl_m^2) && && \text{(A1i)}\\
& dz_m^2 + dl_m^2 = 1, \quad dz_m^2, dl_m^2 \in 0, 1 && && \text{(A1j)}\\
& Y_r = \sum_{m=1}^{30} W_{mr}^3 H_m^2 + b_r^3, \quad Y_r = [\mu_{\hat{f}}, \sigma_{\hat{f}}, \mu_{\hat{g}_1}, \sigma_{\hat{g}_1}, \mu_{\hat{g}_2}, \sigma_{\hat{g}_2}] && r = 1, ..., 6 && \text{(A1k)}
\end{aligned}
$$

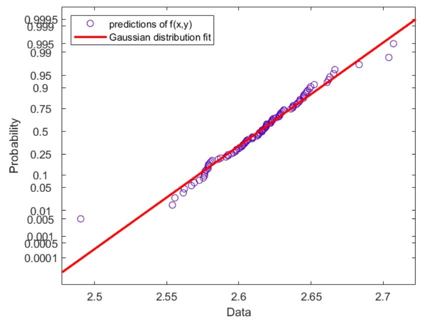
The embedded ReLU MDN is formulated in (A1d)-(A1k). The input layer is described by (A1d). Equations (A1e)-(A1g) represent the first hidden layer. Equations (A1h)-(A1j) express the second hidden layer. The output layer is described by (A1k). $\mu_{\hat{f}}$, $\sigma_{\hat{f}}$, $\mu_{\hat{g}_1}$, $\sigma_{\hat{g}_1}$, $\mu_{\hat{g}_2}$, and $\sigma_{\hat{g}_2}$ are means and standard deviations of the 3 ReLU ANN predictions ($\hat{f}$, $\hat{g}_1$, and $\hat{g}_2$) and they are predicted by the embedded MDN. W^1_{ij}, W^2_{jm}, W^3_{mr}, b^1_j, b^2_m, and b^3_r are the weights between the input and first hidden layers, weights between the first and second hidden layers, weights between the second hidden layer and the output layer, bias for each neuron in the first hidden layer, bias for each neuron in the second hidden layer, and bias for each neuron in the output layer, respectively. H^1_j and H^2_m are the output of each neuron in the first hidden layer and output of each neuron in the second hidden layer, respectively. dz^1_j, dl^1_j, dz^2_m, and dl^2_m are binary variables for max-affine operators in the embedded ReLU MDN. M is a very big number. ε is a constant for determining the confidence value. Equations (A1b) and (A1b) are chance constraints utilized to make the ReLU ANN predictions satisfy constraints given in (2.10b) and (2.10c) (in the main text) with a certain confidence, respectively, under the prediction uncertainty. The derivation of the chance constraint is mentioned in Section 2.2.5 (in the main text).

Table A1: Mean absolute percentage errors of the trained MDN

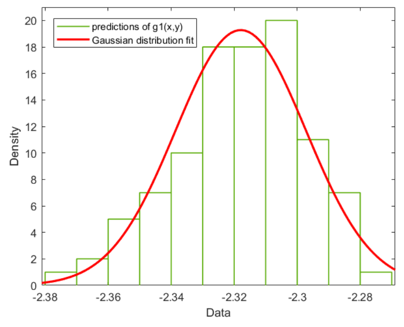
	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
MAPE (%)	0.54	5.22	1.31	4.14	0.07	4.24



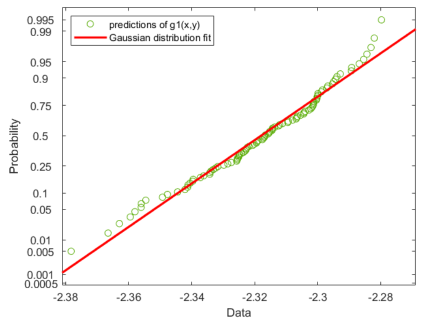
(a) PDF of $\hat{f}$



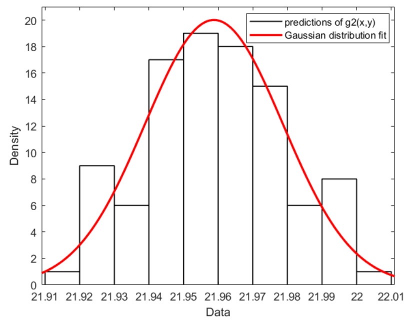
(b) Probability plot of $\hat{f}$



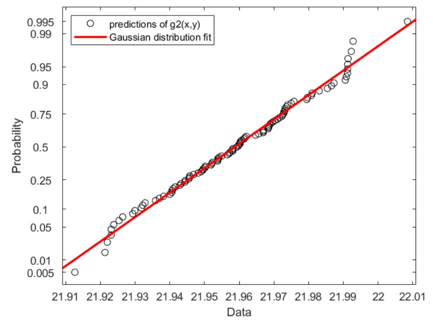
(c) PDF of $\hat{g}_1$



(d) Probability plot of $\hat{g}_1$

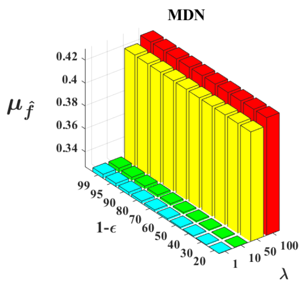


(e) PDF of $\hat{g}_2$

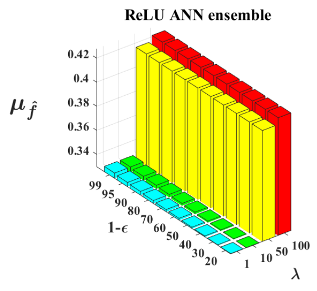


(f) Probability plot of $\hat{g}_2$

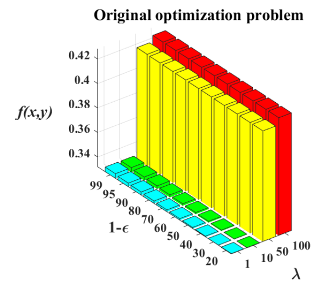
Figure A1: Validation of the prediction uncertainty distribution



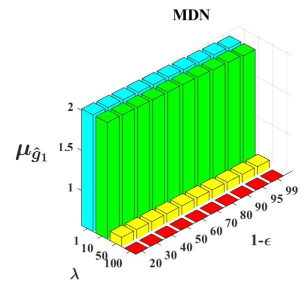
(a) $\mu_{\hat{f}}$ computed from the MDN



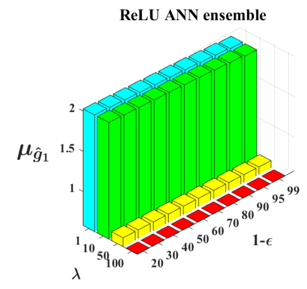
(b) $\mu_{\hat{f}}$ computed from the ReLU ANN ensemble



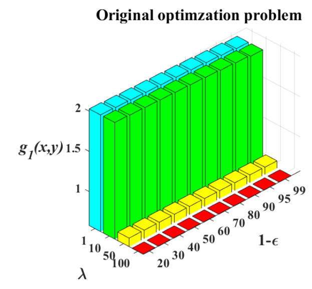
(c) $f(x, y)$ obtained from the original problem formulation



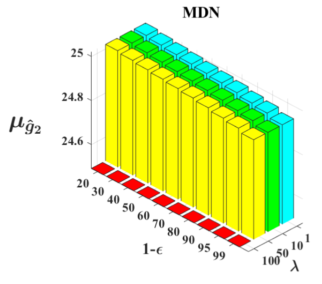
(d) $\mu_{\hat{g}_1}$ computed from the MDN



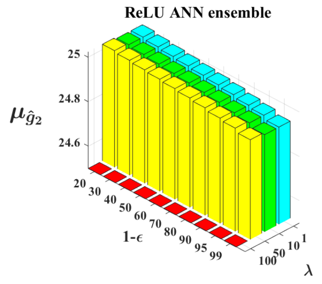
(e) $\mu_{\hat{g}_1}$ computed from the ReLU ANN ensemble



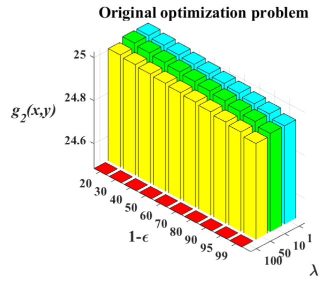
(f) $g_1(x, y)$ obtained from the original problem formulation



(g) $\mu_{\hat{g}_2}$ computed from the MDN

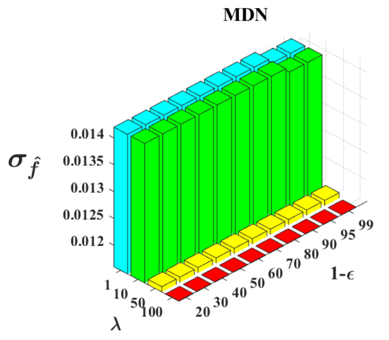


(h) $\mu_{\hat{g}_2}$ computed from the ReLU ANN ensemble

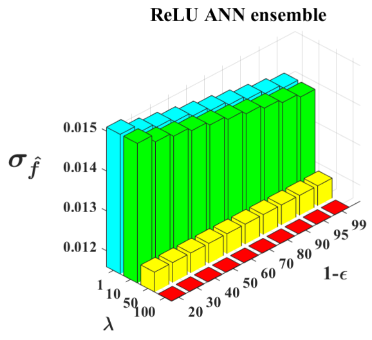


(i) $g_2(x, y)$ obtained from the original problem formulation

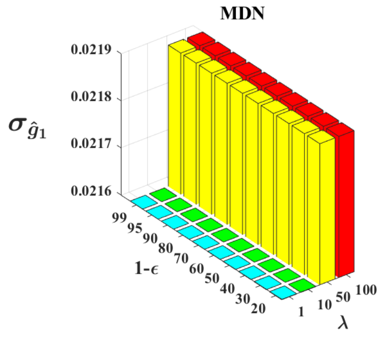
Figure A2: Prediction means computed from the MDN and ReLU ANN ensemble as well as results obtained from the original problem formulation under different $1 - \varepsilon$ and λ



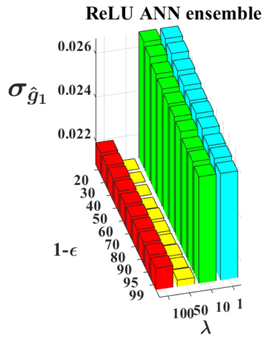
(a) $\sigma_{\hat{f}}$ computed from the MDN



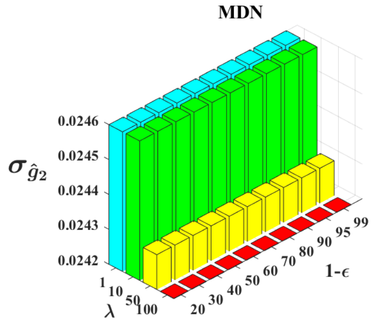
(b) $\sigma_{\hat{f}}$ computed from the ReLU ANN ensemble



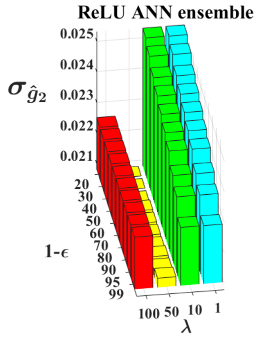
(c) $\sigma_{\hat{g}_1}$ computed from the MDN



(d) $\sigma_{\hat{g}_1}$ computed from the ReLU ANN ensemble



(e) $\sigma_{\hat{g}_2}$ computed from the MDN



(f) $\sigma_{\hat{g}_2}$ computed from the ReLU ANN ensemble

Figure A3: Prediction standard deviations computed from the MDN and ReLU ANN ensemble under different $1 - \varepsilon$ and λ

Table A2: MDN outputs based on different $1-\varepsilon$ and λ

	Confidence value	Optimal decision variables		Predicted means and standard deviations from the MDN					
$\lambda = 1$	$1-\varepsilon$	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
	0.99	-5.2557	4.9877	0.3304	0.0141	1.9497	0.0216	24.9427	0.0246
	0.95	-5.2594	4.9893	0.3298	0.0141	1.9644	0.0216	24.9595	0.0246
	0.9	-5.2614	4.9902	0.3294	0.0142	1.9723	0.0216	24.9684	0.0246
	0.8	-5.2638	4.9912	0.3290	0.0142	1.9818	0.0216	24.9793	0.0246
	0.7	-5.2655	4.9920	0.3287	0.0142	1.9887	0.0216	24.9871	0.0246
	0.6	-5.2670	4.9926	0.3284	0.0142	1.9945	0.0216	24.9938	0.0246
	0.5	-5.2684	4.9932	0.3282	0.0142	2	0.0216	25	0.0246
	0.4	-5.2698	4.9938	0.3280	0.0142	2.0055	0.0216	25.0062	0.0246
	0.3	-5.2713	4.9945	0.3277	0.0142	2.0113	0.0216	25.0129	0.0246
	0.2	-5.2730	4.9952	0.3274	0.0142	2.0182	0.0216	25.0207	0.0246
$\lambda = 10$	$1-\varepsilon$	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
	0.99	-5.2557	4.9877	0.3304	0.0141	1.9497	0.0216	24.9427	0.0246
	0.95	-5.2594	4.9893	0.3298	0.0141	1.9644	0.0216	24.9595	0.0246
	0.9	-5.2614	4.9902	0.3294	0.0142	1.9723	0.0216	24.9684	0.0246
	0.8	-5.2638	4.9912	0.3290	0.0142	1.9818	0.0216	24.9793	0.0246
	0.7	-5.2655	4.9920	0.3287	0.0142	1.9887	0.0216	24.9871	0.0246
	0.6	-5.2670	4.9926	0.3284	0.0142	1.9945	0.0216	24.9938	0.0246
	0.5	-5.2684	4.9932	0.3282	0.0142	2	0.0216	25	0.0246
	0.4	-5.2698	4.9938	0.3280	0.0142	2.0055	0.0216	25.0062	0.0246
	0.3	-5.2713	4.9945	0.3277	0.0142	2.0113	0.0216	25.0129	0.0246
	0.2	-5.2730	4.9952	0.3274	0.0142	2.0182	0.0216	25.0207	0.0246
$\lambda = 50$	$1-\varepsilon$	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
	0.99	-4.9043	4.9921	0.4240	0.0117	0.6629	0.0219	24.9436	0.0243
	0.95	-4.9050	4.9938	0.4237	0.0117	0.6665	0.0219	24.9601	0.0243
	0.9	-4.9054	4.9947	0.4236	0.0117	0.6684	0.0219	24.9689	0.0243
	0.8	-4.9059	4.9957	0.4235	0.0117	0.6708	0.0219	24.9796	0.0243
	0.7	-4.9063	4.9965	0.4234	0.0117	0.6724	0.0219	24.9873	0.0243
	0.6	-4.9066	4.9972	0.4233	0.0117	0.6739	0.0219	24.9939	0.0243
	0.5	-4.9069	4.9978	0.4232	0.0117	0.6752	0.0219	25	0.0243
	0.4	-4.9071	4.9984	0.4231	0.0117	0.6766	0.0219	25.0061	0.0243
	0.3	-4.9074	4.9991	0.4230	0.0117	0.6780	0.0219	25.0127	0.0243
	0.2	-4.9078	4.9998	0.4229	0.0117	0.6797	0.0219	25.0204	0.0243
$\lambda = 100$	$1-\varepsilon$	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
	0.99	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.95	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.9	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.8	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.7	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.6	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.5	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.4	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.3	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242
	0.2	-4.8837	4.9468	0.4300	0.0116	0.5640	0.0219	24.4910	0.0242

Table A3: Results from the ReLU ANN ensemble based on different $1-\varepsilon$ and λ

	Confidence value	Optimal decision variables		Means and standard deviations of ReLU ANN ensemble predictions					
$\lambda = 1$	$1-\varepsilon$	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
	0.99	-5.2557	4.9877	0.3341	0.0141	1.9555	0.0258	24.9367	0.0227
	0.95	-5.2594	4.9893	0.3333	0.0142	1.9709	0.0259	24.9539	0.0233
	0.9	-5.2614	4.9902	0.3328	0.0143	1.9791	0.0259	24.9631	0.0236
	0.8	-5.2638	4.9912	0.3322	0.0144	1.9891	0.0261	24.9743	0.0240
	0.7	-5.2655	4.9920	0.3317	0.0145	1.9963	0.0262	24.9823	0.0242
	0.6	-5.2670	4.9926	0.3314	0.0146	2.0024	0.0262	24.9892	0.0244
	0.5	-5.2684	4.9932	0.3310	0.0147	2.0082	0.0263	24.9956	0.0246
	0.4	-5.2698	4.9938	0.3307	0.0148	2.0139	0.0264	25.0021	0.0248
	0.3	-5.2713	4.9945	0.3303	0.0149	2.0200	0.0266	25.0090	0.0251
	0.2	-5.2730	4.9952	0.3298	0.0151	2.0272	0.0267	25.0170	0.0253
$\lambda = 10$	$1-\varepsilon$	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
	0.99	-5.2557	4.9877	0.3341	0.0141	1.9555	0.0258	24.9367	0.0227
	0.95	-5.2594	4.9893	0.3333	0.0142	1.9709	0.0259	24.9539	0.0233
	0.9	-5.2614	4.9902	0.3328	0.0143	1.9791	0.0259	24.9631	0.0236
	0.8	-5.2638	4.9912	0.3322	0.0144	1.9891	0.0261	24.9743	0.0240
	0.7	-5.2655	4.9920	0.3317	0.0145	1.9963	0.0262	24.9823	0.0242
	0.6	-5.2670	4.9926	0.3314	0.0146	2.0024	0.0262	24.9892	0.0244
	0.5	-5.2684	4.9932	0.3310	0.0147	2.0082	0.0263	24.9956	0.0246
	0.4	-5.2698	4.9938	0.3307	0.0148	2.0139	0.0264	25.0021	0.0248
	0.3	-5.2713	4.9945	0.3303	0.0149	2.0200	0.0266	25.0090	0.0251
	0.2	-5.2730	4.9952	0.3298	0.0151	2.0272	0.0267	25.0170	0.0253
$\lambda = 50$	$1-\varepsilon$	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
	0.99	-4.9043	4.9921	0.4231	0.0121	0.6719	0.0212	24.9480	0.0211
	0.95	-4.9050	4.9938	0.4229	0.0121	0.6754	0.0211	24.9644	0.0210
	0.9	-4.9054	4.9947	0.4228	0.0121	0.6772	0.0211	24.9732	0.0210
	0.8	-4.9059	4.9957	0.4227	0.0121	0.6795	0.0210	24.9838	0.0209
	0.7	-4.9063	4.9965	0.4227	0.0121	0.6811	0.0210	24.9915	0.0209
	0.6	-4.9066	4.9972	0.4226	0.0121	0.6825	0.0210	24.9980	0.0209
	0.5	-4.9069	4.9978	0.4225	0.0121	0.6838	0.0209	25.0042	0.0209
	0.4	-4.9071	4.9984	0.4225	0.0121	0.6851	0.0209	25.0103	0.0208
	0.3	-4.9074	4.9991	0.4224	0.0121	0.6865	0.0209	25.0168	0.0208
	0.2	-4.9078	4.9998	0.4223	0.0121	0.6881	0.0209	25.0245	0.0208
$\lambda = 100$	$1-\varepsilon$	x	y	$\mu_{\hat{f}}$	$\sigma_{\hat{f}}$	$\mu_{\hat{g}_1}$	$\sigma_{\hat{g}_1}$	$\mu_{\hat{g}_2}$	$\sigma_{\hat{g}_2}$
	0.99	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.95	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.9	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.8	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.7	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.6	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.5	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.4	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.3	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225
	0.2	-4.8837	4.9468	0.4279	0.0116	0.5778	0.0219	24.4990	0.0225

Table A4: Results from the ReLU ANN ensemble and original problem formulation based on different $1-\varepsilon$ and λ

	Confidence value	Optimal decision variables		ReLU ANN ensemble constraint satisfaction	Results from the original formulation		
$\lambda = 1$	$1-\varepsilon$	x	y	%	$f(x,y)$	$g_1(x,y)$	$g_2(x,y)$
	0.99	-5.2557	4.9877	96	0.3358	1.9557	24.9423
	0.95	-5.2594	4.9893	92	0.3350	1.9713	24.9604
	0.9	-5.2614	4.9902	83	0.3346	1.9796	24.9701
	0.8	-5.2638	4.9912	67	0.3341	1.9897	24.9818
	0.7	-5.2655	4.9920	46	0.3337	1.9970	24.9903
	0.6	-5.2670	4.9926	36	0.3334	*2.0032	24.9975
	0.5	-5.2684	4.9932	23	0.3331	*2.0090	*25.0043
	0.4	-5.2698	4.9938	14	0.3328	*2.0148	*25.0111
	0.3	-5.2713	4.9945	11	0.3324	*2.0210	*25.0184
	0.2	-5.2730	4.9952	6	0.3321	*2.0283	*25.0269
$\lambda = 10$	$1-\varepsilon$	x	y	%	$f(x,y)$	$g_1(x,y)$	$g_2(x,y)$
	0.99	-5.2557	4.9877	96	0.3358	1.9557	24.9423
	0.95	-5.2594	4.9893	92	0.3350	1.9713	24.9604
	0.9	-5.2614	4.9902	83	0.3346	1.9796	24.9701
	0.8	-5.2638	4.9912	67	0.3341	1.9897	24.9818
	0.7	-5.2655	4.9920	46	0.3337	1.9970	24.9903
	0.6	-5.2670	4.9926	36	0.3334	*2.0032	24.9975
	0.5	-5.2684	4.9932	23	0.3331	*2.0090	*25.0043
	0.4	-5.2698	4.9938	14	0.3328	*2.0148	*25.0111
	0.3	-5.2713	4.9945	11	0.3324	*2.0210	*25.0184
	0.2	-5.2730	4.9952	6	0.3321	*2.0283	*25.0269
$\lambda = 50$	$1-\varepsilon$	x	y	%	$f(x,y)$	$g_1(x,y)$	$g_2(x,y)$
	0.99	-4.9043	4.9921	99	0.4234	0.6699	24.9305
	0.95	-4.9050	4.9938	97	0.4232	0.6733	24.9469
	0.9	-4.9054	4.9947	90	0.4231	0.6751	24.9556
	0.8	-4.9059	4.9957	75	0.4230	0.6773	24.9662
	0.7	-4.9063	4.9965	63	0.4229	0.6788	24.9738
	0.6	-4.9066	4.9972	52	0.4229	0.6802	24.9804
	0.5	-4.9069	4.9978	44	0.4228	0.6815	24.9865
	0.4	-4.9071	4.9984	31	0.4227	0.6827	24.9925
	0.3	-4.9074	4.9991	23	0.4226	0.6841	24.9991
	0.2	-4.9078	4.9998	10	0.4226	0.6856	$*$25.0067
$\lambda = 100$	$1-\varepsilon$	x	y	%	$f(x,y)$	$g_1(x,y)$	$g_2(x,y)$
	0.99	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.95	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.9	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.8	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.7	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.6	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.5	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.4	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.3	-4.8837	4.9468	100	0.4284	0.5781	24.4846
	0.2	-4.8837	4.9468	100	0.4284	0.5781	24.4846

$*$ marks mean that the marked values violate the first or second constraints in the original problem.

A1.2 Distillation process in Chapter 2

The MILP chance-constrained optimization with the mean-variance type objective function and the MDN embedded for the first case study is given as:

$$\min \quad \mu_{T\hat{A}C} + \lambda\sigma_{T\hat{A}C} \tag{A2a}$$

$$\text{s.t.} \quad 0.8 - \mu_{\hat{x}_D} \leq \phi^{-1}(\varepsilon)\sigma_{\hat{x}_D} \tag{A2b}$$

$$0.9 - \mu_{\hat{Rec}} \leq \phi^{-1}(\varepsilon)\sigma_{\hat{Rec}} \tag{A2c}$$

$$NT - NF \geq 1 \tag{A2d}$$

$$1 \leq RR \leq 10 \tag{A2e}$$

$$1 \leq RD \leq 10 \tag{A2f}$$

$$10 \leq NT \leq 150 \tag{A2g}$$

$$2 \leq NF \leq 149 \tag{A2h}$$

$$X_i = [RR, RD, NT, NF] \qquad i = 1, 2, 3, 4 \tag{A2i}$$

$$a_j^1 = \sum_{i=1}^{4} W_{ij}^1 X_i + b_j^1 \qquad j = 1, ..., 50 \tag{A2j}$$

$$H_j^1 \geq a_j^1, \quad 0 \leq H_j^1 \leq M \tag{A2k}$$

$$H_j^1 \leq M(1 - dz_j^1) \tag{A2l}$$

$$H_j^1 \leq a_j^1 + M(1 - dl_j^1) \tag{A2m}$$

$$dz_j^1 + dl_j^1 = 1, \quad dz_j^1, dl_j^1 \in 0, 1 \tag{A2n}$$

$$a_m^2 = \sum_{j=1}^{50} W_{jm}^2 H_j^1 + b_m^2 \qquad m = 1, ..., 30 \tag{A2o}$$

$$H_m^2 \geq a_m^2, \quad 0 \leq H_m^2 \leq M \tag{A2p}$$

$$H_m^2 \leq M(1 - dz_m^2) \tag{A2q}$$

$$H_m^2 \leq a_m^2 + M(1 - dl_m^2) \tag{A2r}$$

$$dz_m^2 + dl_m^2 = 1, \quad dz_m^2, dl_m^2 \in 0, 1 \tag{A2s}$$

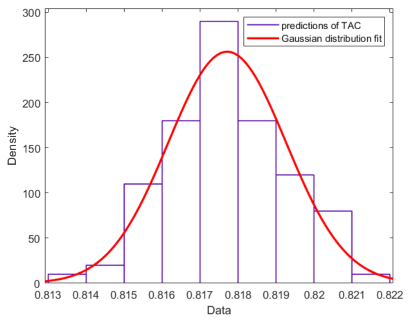
$$Y_r = \sum_{m=1}^{30} W_{mr}^3 H_m^2 + b_r^3 \qquad r = 1, ..., 6 \tag{A2t}$$

$$Y_r = [\mu_{T\hat{A}C}, \sigma_{T\hat{A}C}, \mu_{\hat{x}_D}, \sigma_{\hat{x}_D}, \mu_{\hat{Rec}}, \sigma_{\hat{Rec}}] \tag{A2u}$$

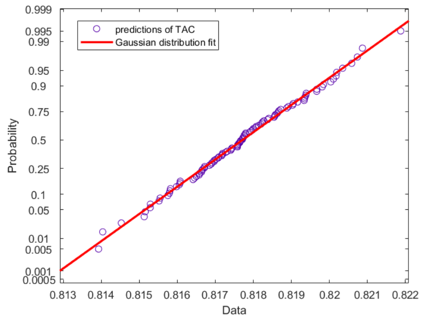
Equations (A2d)-(A2h) are the linear inequality equation for making NF in the reasonable range and the bounds for decision variables. The input layer is described by (A2i). The output layer is expressed as (A2t) and (A2u). $\mu_{T\hat{A}C}$, $\sigma_{T\hat{A}C}$, $\mu_{\hat{x}_D}$, $\sigma_{\hat{x}_D}$, $\mu_{\hat{Rec}}$, and $\sigma_{\hat{Rec}}$ are means and standard deviations of the 3 ReLU ANN predictions ($T\hat{A}C$, $\hat{x}_D$, and $\hat{Rec}$) and they are predicted from the MDN. Equations (A2b) and (A2c) are chance constraints transformed from (2.11b) and (2.11c) (in the main text), respectively. The derivation of the chance constraint is mentioned in Section 2.2.5 (in the main text). Equations (A2j)-(A2s) are the same as (A1e)-(A1j) mentioned in the illustrating example. The inverse cumulative distribution function $\phi^{-1}(\varepsilon)$ in (A2b) and (A2c) is computed based on the assumption of Gaussian distribution because the distribution of predictions from the ReLU ANN ensemble based on a certain observation can be approximated by Gaussian distribution. It is evidenced in Figures A4a-A4f.

Table A5: Mean absolute percentage errors of the trained MDN

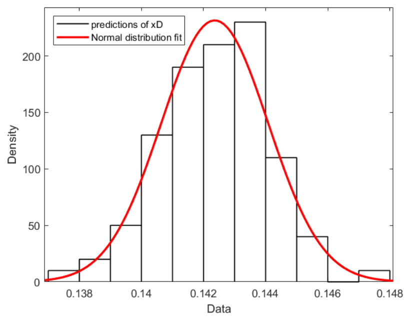
	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
MAPE (%)	1.14	5.21	1.23	6.33	4.74	7.86



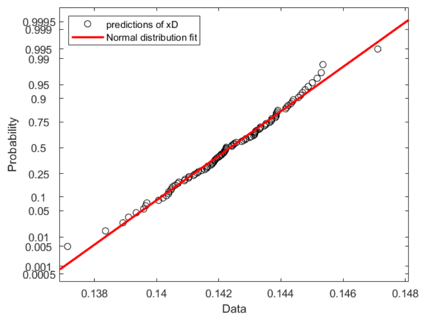
(a) PDF of $T\hat{A}C$



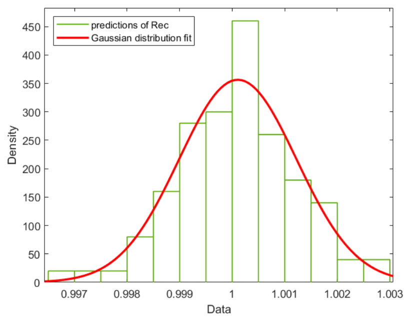
(b) Probability plot of $T\hat{A}C$



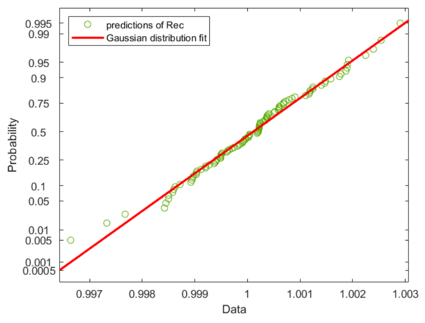
(c) PDF of $\hat{x}_D$



(d) Probability plot of $\hat{x}_D$



(e) PDF of $\hat{Rec}$



(f) Probability plot of $\hat{Rec}$

Figure A4: Prediction uncertainty validation based on $RR = 3.2$, $RD = 6.3957$ GJ/hr, $NT = 150$, and $NF = 2$

Table A6: MDN outputs based on different $1-\varepsilon$ and λ

	Confidence value	Optimal decision variables				Predicted means and standard deviations from the MDN					
$\lambda = 1$	$1-\varepsilon$	RR	RD	NT	NF	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
	0.99	3.689	2.660	16	15	0.1955	0.0013	0.8190	0.0082	0.9212	0.0091
	0.95	3.427	2.571	16	15	0.1898	0.0013	0.8143	0.0087	0.9158	0.0096
	0.9	3.323	2.526	17	15	0.1868	0.0013	0.8112	0.0087	0.9125	0.0097
	0.8	2.865	2.366	18	15	0.1766	0.0014	0.8079	0.0094	0.9089	0.0106
	0.7	2.693	2.301	19	15	0.1721	0.0014	0.8049	0.0094	0.9056	0.0108
	0.6	2.553	2.253	19	15	0.1688	0.0014	0.8024	0.0095	0.9028	0.0109
	0.5	2.419	2.208	19	15	0.1656	0.0014	0.8000	0.0096	0.9000	0.0110
	0.4	2.344	2.182	19	15	0.1640	0.0014	0.7976	0.0096	0.8972	0.0110
	0.3	2.338	2.175	18	14	0.1625	0.0014	0.7949	0.0098	0.8942	0.0110
	0.2	2.310	2.175	17	14	0.1609	0.0015	0.7916	0.0099	0.8906	0.0111
$\lambda = 10$	$1-\varepsilon$	RR	RD	NT	NF	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
	0.99	3.689	2.660	16	15	0.1955	0.0013	0.8190	0.0082	0.9212	0.0091
	0.95	3.427	2.571	16	15	0.1898	0.0013	0.8143	0.0087	0.9158	0.0096
	0.9	3.323	2.526	17	15	0.1868	0.0013	0.8112	0.0087	0.9125	0.0097
	0.8	2.865	2.366	18	15	0.1766	0.0014	0.8079	0.0094	0.9089	0.0106
	0.7	2.693	2.301	19	15	0.1721	0.0014	0.8049	0.0094	0.9056	0.0108
	0.6	2.553	2.253	19	15	0.1688	0.0014	0.8024	0.0095	0.9028	0.0109
	0.5	2.419	2.208	19	15	0.1656	0.0014	0.8000	0.0096	0.9000	0.0110
	0.4	2.344	2.182	19	15	0.1640	0.0014	0.7976	0.0096	0.8972	0.0110
	0.3	2.338	2.175	18	14	0.1625	0.0014	0.7949	0.0098	0.8942	0.0110
	0.2	2.255	2.145	18	14	0.1611	0.0014	0.7918	0.0098	0.8907	0.0110
$\lambda = 50$	$1-\varepsilon$	RR	RD	NT	NF	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
	0.99	3.940	2.728	22	14	0.1989	0.0012	0.8155	0.0066	0.9187	0.0080
	0.95	3.864	2.678	22	14	0.1953	0.0012	0.8109	0.0066	0.9132	0.0080
	0.9	3.824	2.652	22	14	0.1934	0.0012	0.8085	0.0067	0.9102	0.0080
	0.8	2.900	2.370	19	15	0.1768	0.0014	0.8077	0.0091	0.9088	0.0105
	0.7	2.794	2.326	20	15	0.1739	0.0014	0.8047	0.0090	0.9055	0.0104
	0.6	2.771	2.310	20	14	0.1719	0.0013	0.8022	0.0089	0.9026	0.0103
	0.5	2.519	2.233	20	15	0.1674	0.0014	0.8000	0.0092	0.9000	0.0107
	0.4	2.430	2.204	20	15	0.1655	0.0014	0.7977	0.0092	0.8973	0.0107
	0.3	2.403	2.193	19	14	0.1638	0.0014	0.7951	0.0094	0.8943	0.0108
	0.2	2.289	2.151	19	14	0.1616	0.0014	0.7920	0.0095	0.8909	0.0108
$\lambda = 100$	$1-\varepsilon$	RR	RD	NT	NF	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
	0.99	3.940	2.728	22	14	0.1989	0.0012	0.8155	0.0066	0.9187	0.0080
	0.95	3.864	2.678	22	14	0.1953	0.0012	0.8109	0.0066	0.9132	0.0080
	0.9	3.824	2.652	22	14	0.1934	0.0012	0.8085	0.0067	0.9102	0.0080
	0.8	3.750	2.613	22	14	0.1907	0.0012	0.8056	0.0067	0.9068	0.0080
	0.7	3.603	2.561	23	15	0.1884	0.0012	0.8035	0.0067	0.9043	0.0082
	0.6	2.777	2.311	21	15	0.1728	0.0013	0.8022	0.0086	0.9026	0.0102
	0.5	2.610	2.254	22	16	0.1700	0.0013	0.8000	0.0086	0.9000	0.0103
	0.4	2.479	2.209	23	17	0.1679	0.0013	0.7979	0.0085	0.8974	0.0103
	0.3	2.615	2.239	27	20	0.1725	0.0013	0.7961	0.0074	0.8950	0.0095
	0.2	2.446	2.181	28	21	0.1700	0.0013	0.7937	0.0075	0.8919	0.0096

Table A7: Results from the ReLU ANN ensemble based on different $1-\varepsilon$ and λ

	Confidence value	Optimal decision variables				Means and standard deviations of ReLU ANN ensemble predictions					
$\lambda = 1$	$1-\varepsilon$	RR	RD	NT	NF	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
	0.99	3.689	2.660	16	15	0.1935	0.0012	0.8265	0.0088	0.9374	0.0145
	0.95	3.427	2.571	16	15	0.1872	0.0012	0.8216	0.0087	0.9434	0.0133
	0.9	3.323	2.526	17	15	0.1847	0.0012	0.8237	0.0082	0.9461	0.0131
	0.8	2.865	2.366	18	15	0.1739	0.0012	0.8142	0.0081	0.9573	0.0110
	0.7	2.693	2.301	19	15	0.1699	0.0012	0.8121	0.0083	0.9615	0.0102
	0.6	2.553	2.253	19	15	0.1665	0.0012	0.8052	0.0084	0.9630	0.0099
	0.5	2.419	2.208	19	15	0.1632	0.0012	0.7978	0.0088	0.9631	0.0099
	0.4	2.344	2.182	19	15	0.1613	0.0012	0.7932	0.0092	0.9629	0.0101
	0.3	2.338	2.175	18	14	0.1601	0.0012	0.7934	0.0092	0.9608	0.0104
	0.2	2.310	2.175	17	14	0.1594	0.0013	0.7846	0.0096	0.9573	0.0108
$\lambda = 10$	$1-\varepsilon$	RR	RD	NT	NF	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
	0.99	3.689	2.660	16	15	0.1935	0.0012	0.8265	0.0088	0.9374	0.0145
	0.95	3.427	2.571	16	15	0.1872	0.0012	0.8216	0.0087	0.9434	0.0133
	0.9	3.323	2.526	17	15	0.1847	0.0012	0.8237	0.0082	0.9461	0.0131
	0.8	2.865	2.366	18	15	0.1739	0.0012	0.8142	0.0081	0.9573	0.0110
	0.7	2.693	2.301	19	15	0.1699	0.0012	0.8121	0.0083	0.9615	0.0102
	0.6	2.553	2.253	19	15	0.1665	0.0012	0.8052	0.0084	0.9630	0.0099
	0.5	2.419	2.208	19	15	0.1632	0.0012	0.7978	0.0088	0.9631	0.0099
	0.4	2.344	2.182	19	15	0.1613	0.0012	0.7932	0.0092	0.9629	0.0101
	0.3	2.338	2.175	18	14	0.1601	0.0012	0.7934	0.0092	0.9608	0.0104
	0.2	2.255	2.145	18	14	0.1580	0.0013	0.7886	0.0097	0.9605	0.0105
$\lambda = 50$	$1-\varepsilon$	RR	RD	NT	NF	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
	0.99	3.940	2.728	22	14	0.2028	0.0011	0.8380	0.0069	0.9275	0.0120
	0.95	3.864	2.678	22	14	0.1993	0.0011	0.8378	0.0070	0.9202	0.0119
	0.9	3.824	2.652	22	14	0.1974	0.0011	0.8377	0.0071	0.9163	0.0119
	0.8	2.900	2.370	19	15	0.1749	0.0012	0.8200	0.0083	0.9579	0.0112
	0.7	2.794	2.326	20	15	0.1725	0.0012	0.8203	0.0082	0.9599	0.0106
	0.6	2.771	2.310	20	14	0.1713	0.0012	0.8198	0.0081	0.9565	0.0114
	0.5	2.519	2.233	20	15	0.1657	0.0012	0.8089	0.0085	0.9648	0.0093
	0.4	2.430	2.204	20	15	0.1635	0.0012	0.8042	0.0089	0.9656	0.0095
	0.3	2.403	2.193	19	14	0.1621	0.0012	0.8015	0.0091	0.9637	0.0097
	0.2	2.289	2.151	19	14	0.1590	0.0013	0.7961	0.0097	0.9637	0.0102
$\lambda = 100$	$1-\varepsilon$	RR	RD	NT	NF	$\mu_{T\hat{A}C}$	$\sigma_{T\hat{A}C}$	$\mu_{\hat{x}_D}$	$\sigma_{\hat{x}_D}$	$\mu_{\hat{Rec}}$	$\sigma_{\hat{Rec}}$
	0.99	3.940	2.728	22	14	0.2028	0.0011	0.8380	0.0069	0.9275	0.0120
	0.95	3.864	2.678	22	14	0.1993	0.0011	0.8378	0.0070	0.9202	0.0119
	0.9	3.824	2.652	22	14	0.1974	0.0011	0.8377	0.0071	0.9163	0.0119
	0.8	3.750	2.613	22	14	0.1946	0.0011	0.8372	0.0072	0.9128	0.0121
	0.7	3.603	2.561	23	15	0.1915	0.0011	0.8391	0.0073	0.9198	0.0127
	0.6	2.777	2.311	21	15	0.1720	0.0012	0.8230	0.0080	0.9588	0.0108
	0.5	2.610	2.254	22	16	0.1685	0.0012	0.8197	0.0081	0.9653	0.0093
	0.4	2.479	2.209	23	17	0.1659	0.0012	0.8153	0.0086	0.9685	0.0084
	0.3	2.615	2.239	27	20	0.1706	0.0011	0.8313	0.0082	0.9683	0.0086
	0.2	2.446	2.181	28	21	0.1669	0.0012	0.8244	0.0096	0.9713	0.0093

Table A8: Results from the ReLU ANN ensemble and original distillation process model based on different $1-\varepsilon$ and λ

	Confidence value	Optimal decision variables				ReLU ANN ensemble constraint satisfaction	Results from the original model		
$\lambda = 1$	$1-\varepsilon$	RR	RD	NT	NF	%	TAC	x_D	Rec
	0.99	3.689	2.660	16	15	98	0.1930	0.8382	0.9395
	0.95	3.427	2.571	16	15	100	0.1866	0.8344	0.9470
	0.9	3.323	2.526	17	15	100	0.1841	0.8382	0.9515
	0.8	2.865	2.366	18	15	98	0.1737	0.8339	0.9693
	0.7	2.693	2.301	19	15	91	0.1701	0.8321	0.9742
	0.6	2.553	2.253	19	15	71	0.1666	0.8297	0.9806
	0.5	2.419	2.208	19	15	41	0.1632	0.8272	0.9871
	0.4	2.344	2.182	19	15	27	0.1613	0.8255	0.9905
	0.3	2.338	2.175	18	14	28	0.1603	0.8225	0.9847
	0.2	2.310	2.175	17	14	8	0.1590	0.8200	0.9888
$\lambda = 10$	$1-\varepsilon$	RR	RD	NT	NF	%	TAC	x_D	Rec
	0.99	3.689	2.660	16	15	98	0.1930	0.8382	0.9395
	0.95	3.427	2.571	16	15	100	0.1866	0.8344	0.9470
	0.9	3.323	2.526	17	15	100	0.1841	0.8382	0.9515
	0.8	2.865	2.366	18	15	98	0.1737	0.8339	0.9693
	0.7	2.693	2.301	19	15	91	0.1701	0.8321	0.9742
	0.6	2.553	2.253	19	15	71	0.1666	0.8297	0.9806
	0.5	2.419	2.208	19	15	41	0.1632	0.8272	0.9871
	0.4	2.344	2.182	19	15	27	0.1613	0.8255	0.9905
	0.3	2.338	2.175	18	14	28	0.1603	0.8225	0.9847
	0.2	2.255	2.145	18	14	13	0.1581	0.8204	0.9880
$\lambda = 50$	$1-\varepsilon$	RR	RD	NT	NF	%	TAC	x_D	Rec
	0.99	3.940	2.728	22	14	98	0.2031	0.8425	0.9264
	0.95	3.864	2.678	22	14	98	0.1995	0.8419	0.9189
	0.9	3.824	2.652	22	14	98	0.1976	0.8416	0.9148
	0.8	2.900	2.370	19	15	100	0.1751	0.8352	0.9646
	0.7	2.794	2.326	20	15	100	0.1727	0.8338	0.9652
	0.6	2.771	2.310	20	14	98	0.1716	0.8303	0.9585
	0.5	2.519	2.233	20	15	84	0.1660	0.8295	0.9779
	0.4	2.430	2.204	20	15	64	0.1638	0.8279	0.9826
	0.3	2.403	2.193	19	14	56	0.1624	0.8242	0.9797
	0.2	2.289	2.151	19	14	34	0.1594	0.8219	0.9836
$\lambda = 100$	$1-\varepsilon$	RR	RD	NT	NF	%	TAC	x_D	Rec
	0.99	3.940	2.728	22	14	98	0.2031	0.8425	0.9264
	0.95	3.864	2.678	22	14	98	0.1995	0.8419	0.9189
	0.9	3.824	2.652	22	14	98	0.1976	0.8416	0.9148
	0.8	3.750	2.613	22	14	90	0.1948	0.8411	0.9114
	0.7	3.603	2.561	23	15	96	0.1917	0.8430	0.9176
	0.6	2.777	2.311	21	15	100	0.1723	0.8337	0.9613
	0.5	2.610	2.254	22	16	100	0.1688	0.8339	0.9715
	0.4	2.479	2.209	23	17	96	0.1662	0.8340	0.9795
	0.3	2.615	2.239	27	20	100	0.1709	0.8420	0.9707
	0.2	2.446	2.181	28	21	100	0.1673	0.8404	0.9792

A1.3 Ethylene glycol production process in Chapter 2

The MILP chance-constrained optimization with the mean-variance type objective function and the MDN embedded for the second case study is given as:

$$\max \quad \mu_{\hat{F}_{EG}} + \lambda \sigma_{\hat{F}_{EG}} \tag{A3a}$$

$$\text{s.t.} \quad 0.6 - \mu_{\hat{X}_{EO}} \leq \phi^{-1}(\varepsilon)\sigma_{\hat{X}_{EO}} \tag{A3b}$$

$$0.5 - \mu_{\hat{x}_{EG}} \leq \phi^{-1}(\varepsilon)\sigma_{\hat{x}_{EG}} \tag{A3c}$$

$$700 \leq F_w \leq 5000 \tag{A3d}$$

$$221 \leq V_{CSTR} \leq 321 \tag{A3e}$$

$$300 \leq T_f \leq 390 \tag{A3f}$$

$$X_i = [F_w, V_{CSTR}, T_f] \qquad i = 1, 2, 3 \tag{A3g}$$

$$a_j^1 = \sum_{i=1}^{4} W_{ij}^1 X_i + b_j^1 \qquad j = 1, ..., 50 \tag{A3h}$$

$$H_j^1 \geq a_j^1 \tag{A3i}$$

$$0 \leq H_j^1 \leq M \tag{A3j}$$

$$H_j^1 \leq M(1 - dz_j^1) \tag{A3k}$$

$$H_j^1 \leq a_j^1 + M(1 - dl_j^1) \tag{A3l}$$

$$dz_j^1 + dl_j^1 = 1, \quad dz_j^1, dl_j^1 \in 0, 1 \tag{A3m}$$

$$a_m^2 = \sum_{j=1}^{50} W_{jm}^2 H_j^1 + b_m^2 \qquad m = 1, ..., 30 \tag{A3n}$$

$$H_m^2 \geq a_m^2 \tag{A3o}$$

$$0 \leq H_m^2 \leq M \tag{A3p}$$

$$H_m^2 \leq M(1 - dz_m^2) \tag{A3q}$$

$$H_m^2 \leq a_m^2 + M(1 - dl_m^2) \tag{A3r}$$

$$dz_m^2 + dl_m^2 = 1, \quad dz_m^2, dl_m^2 \in 0, 1 \tag{A3s}$$

$$Y_r = \sum_{m=1}^{30} W_{mr}^3 H_m^2 + b_r^3 \qquad r = 1, ..., 6 \tag{A3t}$$

$$Y_r = [\mu_{\hat{F}_{EG}}, \sigma_{\hat{F}_{EG}}, \mu_{\hat{X}_{EO}}, \sigma_{\hat{X}_{EO}}, \mu_{\hat{x}_{EG}}, \sigma_{\hat{x}_{EG}}] \tag{A3u}$$

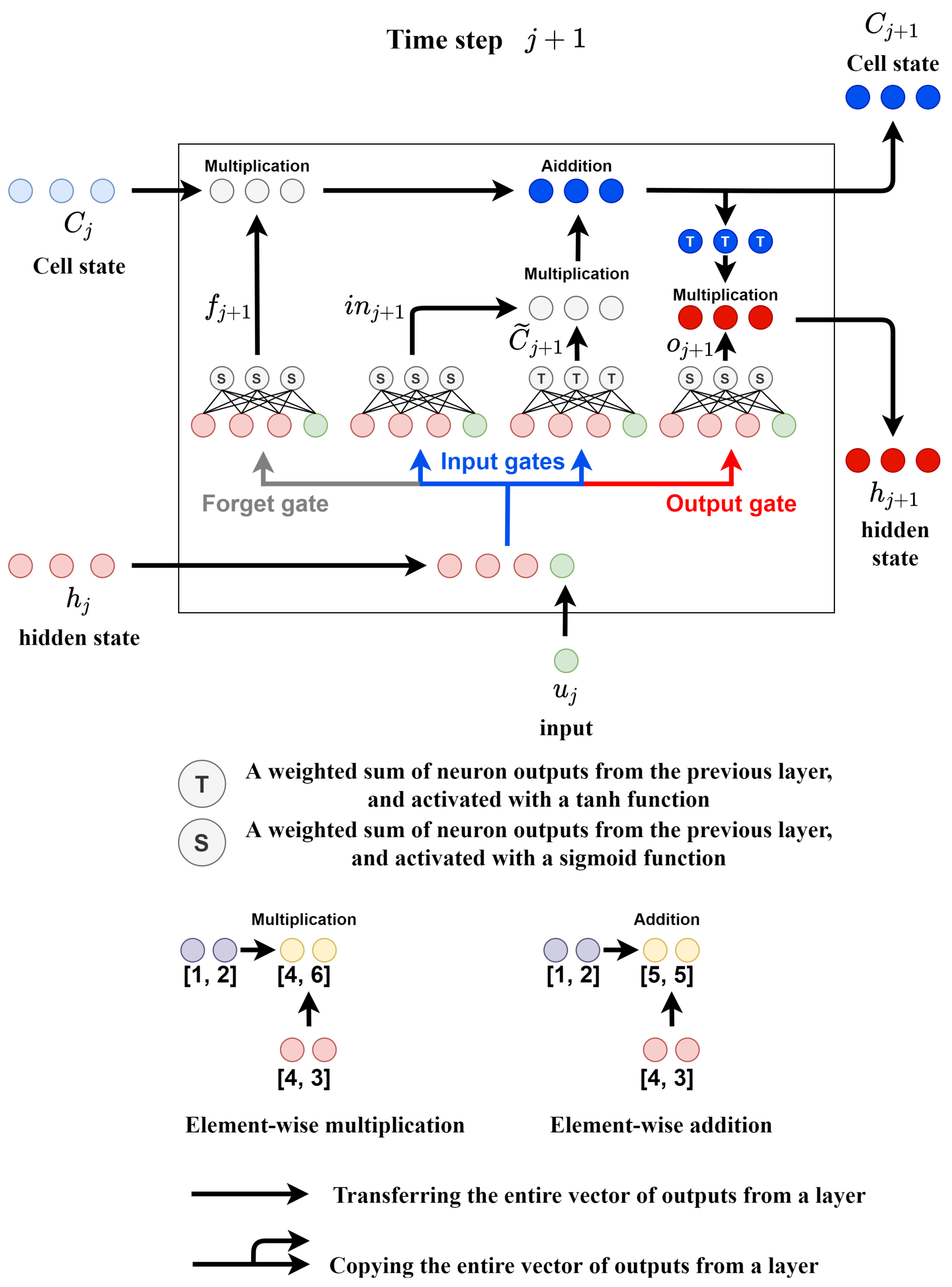
Equations (A3b) and (A3c) are chance constraints transformed from (2.14b) and (2.14c) (in the main text), respectively. The derivation of the chance constraint is mentioned in Section 2.2.5 (in the main text). The embedded MDN is formulated as (A3g)-(A3u). The input layer is expressed as (A3g). The output layer is formulated as (A3t) and (A3u). $\mu_{\hat{F}_{EG}}$, $\sigma_{\hat{F}_{EG}}$, $\mu_{\hat{X}_{EO}}$, $\sigma_{\hat{X}_{EO}}$, $\mu_{\hat{x}_{EG}}$, and $\sigma_{\hat{x}_{EG}}$ are means and standard deviations of the 3 ReLU ANN predictions ($\hat{F}_{EG}$, $\hat{X}_{EO}$, and $\hat{x}_{EG}$) and they are predicted by the embedded MDN. Equations (A3h)-(A3s) are the same as (A1e)-(A1j) mentioned in the illustrating example. The inverse cumulative distribution function $\phi^{-1}(\varepsilon)$ in (A3b) and (A3c) is computed based on the assumption of Gaussian distribution.

Table A9: Mean absolute percentage errors of the trained MDN

	$\mu_{\hat{F}_{EG}}$	$\sigma_{\hat{F}_{EG}}$	$\mu_{\hat{X}_{EO}}$	$\sigma_{\hat{X}_{EO}}$	$\mu_{\hat{x}_{EG}}$	$\sigma_{\hat{x}_{EG}}$
MAPE (%)	2.07	8.43	2.34	9.52	2.11	9.62

A2 Chapter 3 supplementary materials

A2.1 Detailed explanation of the LSTM



Time step $j+1$
C_{j+1}
Cell state
Multiplication
Aiddition
C_j
Cell state
Multiplication
Multiplication
f_{j+1}
in_{j+1}
$\tilde{C}_{j+1}$
o_{j+1}
S
T
Input gates
Forget gate
Output gate
h_{j+1}
hidden state
h_j
hidden state
u_j
input
A weighted sum of neuron outputs from the previous layer, and activated with a tanh function
A weighted sum of neuron outputs from the previous layer, and activated with a sigmoid function
Multiplication
[1, 2]
[4, 6]
[4, 3]
Element-wise multiplication
Addition
[1, 2]
[5, 5]
[4, 3]
Element-wise addition
Transferring the entire vector of outputs from a layer
Copying the entire vector of outputs from a layer

Figure A5: Schematic diagram for illustrating a LSTM cell

As shown in Figure A5, a LSTM cell is composed of the forget gate, input gates, and output gate to remove or add information to the cell state from the previous time step, which is the key to the LSTM. Every LSTM cell has the same structure, weight matrices, and bias vectors. The first step in the LSTM is to determine what information should be removed from the cell state. This determination is made through a sigmoid layer, namely the forget gate layer. The forget gate layer outputs a vector with numbers in the range of 0 and 1, based on the hidden state from the previous time step h_j and the input u_j. The output of the forget gate can be computed by the following equation:

$$f_{j+1} = \sigma_S(W_f \cdot [h_j, u_j] + b_f) \tag{A4}$$

σ_S represents the element-wise sigmoid activation function (node "S" in the figure). W_f and b_f are the weight matrix and the bias vector for the forget gate layer. h_j and u_j are concatenated to be a vector $[h_j, u_j]$, and to be the input of the forget gate. The next step in the LSTM is to determine what information should be stored in the cell state. This determination is made by 2 input gate layers. The first one is to decide which values to be updated through a sigmoid layer based on $[h_j, u_j]$. The second one is to create candidate values in the vector $\tilde{C}_{j+1}$ via a tanh layer. The two input gates can be represented by the following equations:

$$in_{j+1} = \sigma_S(W_{in} \cdot [h_j, u_j] + b_{in}) \tag{A5a}$$

$$\tilde{C}_{j+1} = \sigma_T(W_C \cdot [h_j, u_j] + b_C) \tag{A5b}$$

σ_T represents the element-wise tanh activation function (node "T" in the figure). W_{in} and W_C are weight matrices for the two input gate layers. b_{in} and b_C are bias vectors for the two layers. Subsequently, the element-wise multiplication of C_j (the cell state from the previous time step) and f_{j+1} is carried out to forget determined information, and then the corresponding outcome is added element-wisely by $in_{j+1} \circ \tilde{C}_{j+1}$ to be C_{j+1}. Note that $\circ$ represents the element-wise multiplication. Finally, to generate the hidden state h_{j+1} at the time step $j+1$, $[h_j, u_j]$ is transformed via the output gate layer with the element-wise sigmoid function, and then be multiplied element-wisely by the transformation of C_{j+1} via a tanh layer. The above calculation can be expressed as following equations:

$$C_{j+1} = f_{j+1} \circ C_j + in_{j+1} \circ \tilde{C}_{j+1} \tag{A6a}$$

$$o_{j+1} = \sigma_S(W_o \cdot [h_j, u_j] + b_o) \tag{A6b}$$

$$h_{j+1} = o_{j+1} \circ \sigma_T(C_{j+1}) \tag{A6c}$$

where W_o and b_o are the weight matrix and the bias vector for the output gate layer, respectively. The LSTM forgets and updates information in the cell and hidden states through the above procedure for all time steps to be capable of learning long-sequence temporal data.

A2.2 Illustrating example

A2.2.1 Calculation in the discretized formulation

Objective function

The expected values, $x_{1,j}$, and $x_{2,j}$ in (3.15a) and (3.15d) in the main text, are computed according to the following steps:

1) ξ_1 and ξ_2 are randomly sampled 1000 times to gain two sets which are $\{\xi_{1,1}, ..., \xi_{1,10^3}\}$ and $\{\xi_{2,1}, ..., \xi_{2,10^3}\}$, respectively. Then, the random parameters ξ_1 and ξ_2 are first fixed at values of $\xi_{1,1}$ and $\xi_{2,1}$, respectively.

2) Based on the fixed values of random parameters, x_1 and x_2 are computed from the original form of the dynamic model in (3.15c) in the main text ($\frac{dx_1(t)}{dt} = x_2(t) + \xi_1, \frac{dx_2(t)}{dt} = -u(t) + \xi_2$) from $t = 0$ to $t = 3$ with 1000 time intervals, by using the trapezoidal rule method. The computation of the differential equation model is based on a sequence of u_j which is the discrete form of $u(t)$. A sequence of u_j is implemented to the differential equation model at 10 time points which are $t = 0 \sim 2.7$ with an interval of 0.3. The manipulated variable (MV) $u(t)$ remains constant in the interval between two consecutive time points. For instance, $u(t) = u_0$, $u(t) = u_1$, and $u(t) = u_9$ are at the intervals of $0 \leq t < 0.3$, $0.3 \leq t < 0.6$, and $2.7 \leq t \leq 3.0$, respectively.

3) The computed x_1 and x_2 are extracted individually at 10 time points which are $t = 0.3 \sim 3.0$ with an interval of 0.3. This way, the sequences of $x_{1,j}$ and $x_{2,j}$ can be

obtained for the computation of $-x_{1,j}-3.3$ and $-x_{2,j}-3.3$ in (3.15d) in the main text.

4) 1000 sequences of $x_{1,j}$ and $x_{2,j}$ can be computed by repeating steps $2 \sim 3$, based on different realizations of ξ_1 and ξ_2. The 1000 sequences of $x_{1,j}$ and $x_{2,j}$ can be deemed as two 1000×10 matrices. 1000 rows in each matrix correspond to 1000 computations based on different realizations of ξ_1 and ξ_2, and 10 columns in each matrix correspond to $j = 1, ..., 10$.

5) By taking mean values of the 1000×10 matrix of $x_{1,j}$ through the column direction, the 10-element vector of mean values can be obtained. This 10-element vector is the same as the sequence of expected values in (3.15a) in the main text.

One remark here is that the above calculation is based on a certain sequence of u_j.

Quantile-based inequality

The quantile values in (3.26) in the main text with respect to a certain sequence of u_j, are calculated through the following steps:

1) The 1000×10 matrices of $x_{1,j}$ and $x_{2,j}$ are gained via the procedure mentioned in the previous paragraph.

2) The 1000×10 matrices of $-x_{1,j} - 3.3$ and $-x_{2,j} - 3.3$ are calculated with respect to the gained 1000×10 matrices of $x_{1,j}$ and $x_{2,j}$.

3) The column vectors of the matrices of $-x_{1,j} - 3.3$ and $-x_{2,j} - 3.3$ are 1000-element vectors correspond to different sampling instants j. The 1000-element vectors of $-x_{1,j}-3.3$ and $-x_{2,j} - 3.3$ are denoted as $g_{1,I,j}$ and $g_{2,I,j}$, respectively. The subscript I is employed for the variables related to the calculation of quantile values, expected values, and probabilities in this illustrating example.

4) Based on a certain sampling instant j, $\bar{g}_{I,j}$ can be obtained via $\bar{g}_{I,j} = \max\{g_{1,I,j}, g_{2,I,j}\}$.

5) Based on a certain sampling instant j, the quantile value of $\bar{g}_{I,j}$ can be acquired through $\widetilde{Q}_{I,j}^{1-\varepsilon}(\bar{g}_{I,j}) = \bar{g}_{I,j,\lceil 1000\times 1-\varepsilon \rceil}$. $1-\varepsilon$ is equal to 0.8 in the illustrating example.

A2.2.2 Optimization involving two LSTM models

According to Section 3.2.4.2 in the main text, two LSTM models are utilized to handle the discretized objective and quantile functions in (3.15a) and (3.26) in the main text, respectively. The mean absolute percentage errors (MAPEs) of the two LSTMs based on the testing data sets are shown in the following table.

Table A10: Mean absolute percentage errors (MAPEs) of the 2 LSTMs based on the testing data sets, with respect to the predictions at different sampling instants

j	MAPE of the first LSTM	MAPE of the second LSTM
1	0.0831 %	0.0233 %
2	0.2571 %	0.0265 %
3	0.3206 %	0.0553 %
4	0.4461 %	0.1130 %
5	0.5685 %	0.2163 %
6	0.7262 %	0.8572 %
7	0.9277 %	1.2143 %
8	1.1560 %	1.3064 %
9	1.2785 %	1.4935 %
10	1.3363 %	1.5178 %

After training the two LSTM models, the two models are incorporated into the illustrating problem to address the discretized objective function and quantile-based inequality involved. The optimization involving the two LSTM models is given as:

$$\min_{u_j} \quad \sum_{j=0}^{9} 2 \times \frac{0.3}{2}(\hat{E}_{I,j} + \hat{E}_{I,j+1}) \tag{A7a}$$

$$\text{s.t.} \quad \hat{Q}_{I,j}^{1-\varepsilon} \leq 0 \quad j = 1, ..., 10 \tag{A7b}$$

$$\hat{Q}_{I,j=1,...,10}^{1-\varepsilon} = L_{I,1}(u_{j=0,...,9}) \tag{A7c}$$

$$\hat{E}_{I,j=1,...,10} = L_{I,2}(u_{j=0,...,9}) \tag{A7d}$$

$$|u_j| \leq 2 \quad j = 0, ..., 9 \tag{A7e}$$

$$\hat{E}_{I,0} = 2 \tag{A7f}$$

where $\hat{E}_{I,j}$ is the expected value predicted from the second LSTM at each sampling instant j. Note that $\hat{E}_{I,0}$ is equal to 2 because $\hat{E}_{I,0}$ is based on $x_{1,0}$ which is a constant ($x_{1,0} = 2$).

$\hat{Q}_{I,j}^{1-\varepsilon}$ is the quantile value predicted from the first LSTM at each sampling instant j. $L_{I,1}$ and $L_{I,2}$ are the first and second LSTM models that individually take a sequence of u_j as inputs.

A2.2.3 Supplementary results

In the illustrating example, the detailed results obtained from the proposed RNN-based approach are shown in the following table.

Table A11: Results of the illustrating example from the proposed RNN-based approach

j	u_j	j	$Pr\left\{\begin{array}{c} -x_{1,j} - 3.3 \leq 0 \\ -x_{2,j} - 3.3 \leq 0 \end{array}\right\}$
0	2.0000	1	1.0000
1	2.0000	2	1.0000
2	2.0000	3	1.0000
3	2.0000	4	1.0000
4	0.8590	5	0.9307
5	-2.0000	6	1.0000
6	-2.0000	7	0.9924
7	-2.0000	8	0.9332
8	-2.0000	9	0.8465
9	-2.0000	10	0.8001

[a] Objective value according to the computation of the second LSTM model: -0.5866

[b] True objective value: -0.5872

[c] Solution time: 1.7 seconds

A2.2.4 Calculation of the results

The results shown in Figure 3.7 in the main text are calculated through the following steps:

1) Two 10^3-element sets of ξ_1 and ξ_2 are sampled. Such two sets are different from the two sets mentioned in Section A2.2.1.

2) x_1 and x_2 are computed through the differential equation model ($\frac{dx_1(t)}{dt} = x_2(t) + \xi_1, \frac{dx_2(t)}{dt} = -u(t) + \xi_2$) 10^3 times individually, with respect to the optimal sequence of u_j and the two sets gained from the previous step. One computation of the differential equation model is achieved via the trapezoidal rule method with 1000 time intervals

($t = 0 \sim 3$ with an interval of 0.003), based on the optimal sequence of u_j, a sample of ξ_1, and a sample of ξ_2. Note that the implementation of the optimal u_j sequence here is the same as the implementation of the u_j sequence mentioned in step 2 in the first paragraph of Section A2.2.1. Therefore, the computed x_1 and x_2 are two 1001-element vectors with respect to a pair of ξ_1 and ξ_2 realizations (1000 computed values +1 initial value for each vector). The computed x_1 and x_2 become 1000×1001 matrices with respect to 1000 pairs of ξ_1 and ξ_2 realizations. 1000 rows in each matrix correspond to the 1000 pairs of ξ_1 and ξ_2 realizations, and 1001 columns in each matrix correspond to 1001 time points. These matrices are denoted as $x_{1,RU}$ and $x_{2,RU}$, and 100 rows of each matrix are randomly selected and illustrated in Figure 3.7 in the main text. The reason for only choosing 100 rows is to avoid too many overlapping curves, as shown in Figure 3.7 in the main text, since it will be difficult to interpret with too many overlapping curves. The subscript *RU* is employed for the x_1 and x_2 based on the optimal solution obtained from the proposed RNN-based approach, and under different uncertainty scenarios.

3) $x_{1,RN}$ and $x_{2,RN}$ in Figure 3.7 in the main text are computed through the differential equation model with respect to the optimal sequence of u_j and under the nominal scenario. The subscript *RN* is employed for the x_1 and x_2 based on the optimal solution obtained from the proposed RNN-based approach, and under the nominal scenario. The nominal scenario is based on the mean values of ξ_1 and ξ_2 which are equal to zero. The computation of $x_{1,RN}$ and $x_{2,RN}$ is similar to the computation of $x_{1,RU}$ and $x_{2,RU}$ mentioned in the previous step.

The joint constraint satisfaction probabilities presented in Table A11 are calculated according to the following steps:

1) The column vectors of the matrices of $x_{1,RU}$ and $x_{2,RU}$ are extracted with respect to 10 time points ($t = 0.3 \sim 3$ with an interval of 0.3). These column vectors can be deem as 1000-element vectors of $x_{1,j}$ and $x_{2,j}$ at $j = 1, ..., 10$.

2) The vectors of $x_{1,j}$ and $x_{2,j}$ gained from the previous step are plugged into $-x_{1,j} - 3.3$ and $-x_{2,j} - 3.3$, respectively.

3) For a certain sampling instant j, $-x_{1,j} - 3.3$ and $-x_{2,j} - 3.3$ can be deemed as two 10^3-element vectors, and be denoted as $P_{1,I,j}$ and $P_{2,I,j}$, respectively.

4) For each sampling instant j, the 10^3-element vector $\bar{P}_{I,j}$ is produced via $\bar{P}_{I,j} = \max\{P_{1,I,j}, P_{2,I,j}\}$.

5) For a certain sampling instant j, the percentage of elements in $\bar{P}_{I,j}$ less or equal to zero is calculated, and such the percentage is the joint constraint satisfaction probability at a certain sampling instant shown in Table A11. On the other hand, the objective value first mentioned in the footnote of Table A11 is based on the predictions from the second LSTM mentioned in Section 3.2.4.2 in the main text, with respect to the obtained optimal sequence of u_j. The real objective value also mentioned in the footnote of Table A11 is gained based on the matrix of $x_{1,RU}$. More specifically, the mean values of the matrix $x_{1,RU}$ are taken in the column direction to obtain a 1001-element vector of mean values. Then, the real objective value is calculated according to the trapezoidal rule approximation of (3.15a) in the main text, based on the 1001-element vector of mean values.

A2.3 SOCP of the HDS

A2.3.1 Discretization of the SOCP

To exploit the proposed RNN-based approach to solve the SOCP of the HDS, the objective function and joint chance constraint in (3.27a) and (3.27g) in the main text, respectively, should be discretized first. The discretized objective function and joint chance constraint are given as:

$$\sum_{j=0}^{19}(C_{H4}X_1\dot{F}_{1,j} + C_{H3}X_2\dot{F}_{2,j}) \times 0.1 \tag{A8a}$$

$$Pr(X_{H2,j} \geq 0.7, X_{5,j} \geq 0.9) \geq 1-\varepsilon \quad j = 1, ..., 20 \tag{A8b}$$

For the above equation and inequality, $j = 0, ..., 19$ correspond to $t = t_0 \sim t_f - 0.1$ with an interval of 0.1, and $j = 1, ..., 20$ correspond to $t = t_0 + 0.1 \sim t_f$ with an interval of 0.1. Such relationships between j and t hold throughout the HDS case study. Equation (A8a) is obtained from (3.27a) in the main text through the right rectangular approximation method (RRAM) with 20 time intervals. 0.1 in (A8a) is the time interval.

The values of $X_{H2,j}$ and $X_{5,j}$ in (A8b) are computed according to the following steps:

1) t_0 and t_f are first set to be 0 and 2, respectively. Note that $\dot{F}_x^{H2}$ and X_{H2} at $t = 0$ are respectively equal to 682.5 and 0.9 which are the given initial conditions.

2) ρ and X_3 are randomly sampled 1000 times to gain two sets which are $\{\rho_1, ..., \rho_{1000}\}$ and $\{X_{3,1}, ..., X_{3,1000}\}$, respectively. Then, the random parameters ρ and X_3 are first fixed at the values of ρ_1 and $X_{3,1}$, respectively.

3) Based on the fixed values of random parameters, X_{H2} and X_5 are computed from the process model ((3.27b)-(3.27f) in the main text) from $t_0 = 0$ to $t_f = 2$ with 1000 time intervals, by using the trapezoidal rule method. The computation of the process model is also based on a certain sequence of $\dot{F}_{1,j}$, $\dot{F}_{2,j}$, and $\dot{F}_{10,j}$ which are the manipulated variables (MVs). A sequence of MVs is implemented to the process model at 20 time points which are $t = t_0 \sim t_f - 0.1$ with an interval of 0.1. The MVs remain constant in the interval between two consecutive time points. The implementation of MVs here is similar to the implementation of the u_j sequence mentioned in Section A2.2.1.

4) The computed X_{H2} and X_5 are extracted individually at 20 time points which are $t = t_0 + 0.1 \sim t_f$ with an interval of 0.1. Thus, the sequences of $X_{H2,j}$ and $X_{5,j}$ can be obtained.

5) 1000 sequences of $X_{H2,j}$ and $X_{5,j}$ can be computed by repeating steps $3 \sim 4$, based on different pairs of ρ and X_3 realizations. The 1000 sequences of $X_{H2,j}$ and $X_{5,j}$ can be deemed as two 1000×20 matrices. 1000 rows in each matrix correspond to 1000 computations based on different pairs of ρ and X_3 realizations. 20 columns in each matrix correspond to $j = 1, ..., 20$. One remark here is that the above calculation is based on a certain sequence of MVs and a certain set of initial conditions.

A2.3.2 Joint chance constraint reformulation

The discretized joint chance constraint in (A8b) can be further reformulated equivalently as the discretized quantile-based inequality. Based on a certain sequence of MVs and a certain set of initial conditions, the quantile values in the discretized quantile-based inequality is calculated through the following steps:

1) The 1000×20 matrices of $X_{H2,j}$ and $X_{5,j}$ are gained via the procedure mentioned in the previous subsection.

2) $X_{H2,j} \geq 0.7$ and $X_{5,j} \geq 0.9$ in (A8b) can be rearranged to be $-X_{H2,j} + 0.7 \leq 0$ and $-X_{5,j} + 0.9 \leq 0$, respectively. Then, the 1000×20 matrices of $-X_{H2,j} + 0.7$ and $-X_{5,j} + 0.9$ are calculated with respect to the gained 1000×20 matrices of $X_{H2,j}$ and $X_{5,j}$.

3) The column vectors of matrices of $-X_{H2,j} + 0.7$ and $-X_{5,j} + 0.9$ are 1000-element vectors correspond to different sampling instants j. The 1000-element vectors of $-X_{H2,j} + 0.7$ and $-X_{5,j} + 0.9$ are denoted as $g_{1,H,j}$ and $g_{2,H,j}$, respectively. The subscript H is employed for the variables related to the calculation of quantile values and probabilities in the HDS case study.

4) Based on a certain sampling instant j, $\bar{g}_{H,j}$ can be obtained via $\bar{g}_{H,j} = \max\{g_{1,H,j}, g_{2,H,j}\}$.

5) Based on a certain sampling instant j, the quantile value of $\bar{g}_{H,j}$ can be acquired through $\widetilde{Q}^{1-\varepsilon}_{H,j}(\bar{g}_{H,j}) = \bar{g}_{H,j,\lceil 1000\times(1-\varepsilon)\rceil}$. $1-\varepsilon$ is equal to 0.8 in the HDS case study. Subsequently, the discretized quantile-based inequality in this case study can be formulated as:

$$\widetilde{Q}^{1-\varepsilon}_{H,j}(\bar{g}_{H,j}) \leq 0 \quad j = 1, ..., 20 \tag{A9}$$

A2.3.3 LSTM for the SOCP

To reduce the complexity of the SOCP in the HDS case study and make it deterministically solvable, an LSTM is employed to predict the sequences of quantile values based on different sequences of MVs. This LSTM is a stacked LSTM, and its structure is illustrated as the follow:

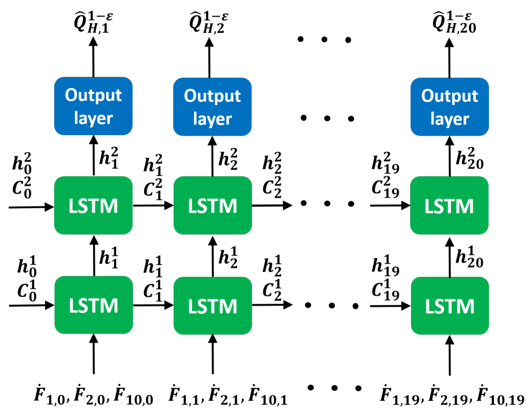

$\hat{Q}^{1-\varepsilon}_{H,1}$
$\hat{Q}^{1-\varepsilon}_{H,2}$
$\hat{Q}^{1-\varepsilon}_{H,20}$
Output layer
LSTM
h_0^2 c_0^2
h_1^2 c_1^2
h_2^2 c_2^2
h_{19}^2 c_{19}^2
h_{20}^2
h_0^1 c_0^1
h_1^1 c_1^1
h_2^1 c_2^1
h_{19}^1 c_{19}^1
h_{20}^1
$\dot{F}_{1,0}, \dot{F}_{2,0}, \dot{F}_{10,0}$
$\dot{F}_{1,1}, \dot{F}_{2,1}, \dot{F}_{10,1}$
$\dot{F}_{1,19}, \dot{F}_{2,19}, \dot{F}_{10,19}$

Figure A6: Schematic diagram of the stacked LSTM used in the HDS case study

As can be seen from the above figure, the stacked LSTM is composed of 2 LSTM layers. The initial hidden state h_0^1 and cell state C_0^1 are fed to the first LSTM cell in the first LSTM layer. The initial hidden state h_0^2 and cell state C_0^2 are fed to the first LSTM cell in the second LSTM layer. The h_0^1, C_0^1, h_0^2, and C_0^2 are set to be zero vectors. In the first LSTM layer, the hidden state h_{j+1}^1 and the cell state C_{j+1}^1 are produced based on h_j^1, C_j^1, $\dot{F}_{1,j}$, $\dot{F}_{2,j}$, and $\dot{F}_{10,j}$. Subsequently, in the second LSTM layer, the hidden state h_{j+1}^2 and the cell state C_{j+1}^2 are generated based on h_{j+1}^1, h_j^2, and C_j^2. Afterwards, for a certain sampling instant $j+1$, h_{j+1}^2 is fed to a output layer to predict the quantile value. The dimensions of h_{j+1}^1 and C_{j+1}^1 are both equal to 50. The dimensions of h_{j+1}^2 and C_{j+1}^2 are both equal to 30. One remark here is that the superscript m in h_{j+1}^m and C_{j+1}^m (m=1,2) is the index for the LSTM layers, instead of the power.

The training, validation, and testing data sets for this LSTM can be produced via the calculation of quantile values mentioned in Section A2.3.2 with respect to different sequences of MVs. The validation and testing data sets are used for the early stopping of the training procedure and evaluating prediction errors, respectively. Afterwards, 10^5, 10^4, and 10^4 samples are produced for training, validation, and testing, respectively. The overall time for data generation is around 3600 seconds. Each sample is composed of a sequence of quantile values paired with the corresponding sequence of MVs. The architecture of the LSTM is determined through the 10-fold cross-validation based on the training set first, and then the LSTM is trained on the entire training set with the use of the validation set for the early stopping. The LSTM is constructed and trained using Keras. The training time for this LSTM is around 1000 seconds. The mean absolute percentage errors (MAPEs) of the LSTM based on the testing data set are shown in the following table:

Table A12: Mean absolute percentage errors (MAPEs) of the LSTM based on the testing data set, with respect to the predictions at different sampling instants

j	1	2	3	4	5
MAPEs	1.07 %	1.27 %	1.52 %	2.48 %	2.66 %
j	6	7	8	9	10
MAPEs	2.78 %	2.91 %	3.15 %	4.28 %	4.53 %
j	11	12	13	14	15
MAPEs	2.88 %	2.97 %	2.85 %	3.78 %	4.23 %
j	16	17	18	19	20
MAPEs	3.76 %	3.91 %	4.12 %	4.78 %	4.62 %

A2.3.4 Optimization incorporating the LSTM

The discretized objective function and the LSTM for predicting quantile values are incorporated into the SOCP in Section 3.2.5.1 in the main text. The SOCP involving the discretized objective function and the LSTM is given as:

$$\min_{\dot{F}_{1,j},\dot{F}_{2,j},\dot{F}_{10,j}} \quad \sum_{j=0}^{19}(C_{H4}X_1\dot{F}_{1,j} + C_{H3}X_2\dot{F}_{2,j}) \times 0.1 \tag{A10a}$$

$$\text{s.t.} \quad \hat{Q}_{H,j}^{1-\varepsilon} \leq 0 \qquad j = 1, ..., 20 \tag{A10b}$$

$$\hat{Q}_{H,j=1,...,20}^{1-\varepsilon} = L_H(\dot{F}_{1,j=0,...19}, \dot{F}_{2,j=0,...19}, \dot{F}_{10,j=0,...19}) \tag{A10c}$$

$$0 \leq \dot{F}_{1,j} \leq 1400 \qquad j = 0, ..., 19 \tag{A10d}$$

$$0 \leq \dot{F}_{2,j} \leq 790 \qquad j = 0, ..., 19 \tag{A10e}$$

$$0 \leq \dot{F}_{10,j} \leq 1500 \qquad j = 0, ..., 19 \tag{A10f}$$

where $\hat{Q}_{H,j}^{1-\varepsilon}$ is the quantile value predicted from the LSTM at a certain sampling instant j. L_H is the LSTM model that takes the sequence of MVs as inputs.

The above SOCP is an optimization problem involving the LSTM model to predict quantile values. It is solved by using IPOPT 0.3.0 in the Python environment. Also, the automatic differentiation (AD) for gaining gradient information is conducted by using TensorFlow. It took 19.7 seconds to solve this SOCP, and the obtained results are shown in Table A13.

A2.3.5 Supplementary results

The detailed results of the SOCP of the HDS attained from the proposed RNN-based approach are shown in the following table.

Table A13: Results of the SOCP of the HDS from the proposed RNN-based approach

j	$F_{1,j}$ $(kmol/h)$	$F_{2,j}$ $(kmol/h)$	$F_{10,j}$ $(kmol/h)$	j	$Pr\left\{\begin{array}{c} X_{H2,j} \geq 0.7 \\ X_{5,j} \geq 0.9 \end{array}\right\}$
0	396.4	0.0	43.6	1	0.9548
1	449.0	0.0	73.6	2	0.9165
2	471.5	0.0	92.0	3	0.8344
3	637.2	0.0	415.7	4	0.8444
4	732.0	0.0	589.7	5	0.8597
5	755.5	0.0	625.4	6	0.8559
6	763.2	0.0	626.9	7	0.8495
7	771.3	0.0	633.7	8	0.8447
8	777.5	0.0	640.1	9	0.8402
9	782.3	0.0	645.5	10	0.8367
10	785.9	0.0	649.6	11	0.8345
11	788.6	0.0	652.6	12	0.8329
12	790.6	0.0	654.6	13	0.8322
13	792.1	0.0	656.0	14	0.8317
14	793.2	0.0	656.9	15	0.8315
15	794.0	0.0	657.4	16	0.8314
16	794.6	0.0	657.5	17	0.8316
17	794.9	0.0	656.8	18	0.8319
18	795.8	0.0	658.9	19	0.8318
19	792.0	0.0	628.8	20	0.8326

A2.3.6 Calculation of the results

The results shown in Figure 3.10 in the main text are calculated through the following steps:

1) Two 1000-element sets of ρ and X_3 are sampled. Such two sets are different from the two sets sampled previously.

2) X_{H2} and X_5 are computed through the process model 1000 times, with respect to the optimal sequence of MVs and the two sets gained from the previous step. One computation of the process model is achieved via the trapezoidal rule method with 1000 time intervals ($t = 0 \sim 2$ with an interval of 0.002), based on the optimal sequence of MVs, a sample of ρ, and a sample of X_3. Therefore, the computed X_{H2} and X_5 are two 1001-element vectors with respect to a pair of ρ and X_3 realizations

(1000 computed values + 1 initial value for each vector). The computed X_{H2} and X_5 become 1000×1001 matrices with respect to 1000 pairs of ρ and X_3 realizations. 1000 rows in each matrix correspond to the 1000 pairs of ρ and X_3 realizations, and 1001 columns in each matrix correspond to 1001 time points. These matrices are denoted as $X_{H2,RU}$ and $X_{5,RU}$, and 100 rows of each matrix are randomly selected and illustrated in Figure 3.10 in the main text. The reason for only choosing 100 rows is to avoid too many overlapping curves shown in the figure because the figure will be difficult to interpret with too many overlapping curves. The subscript RU is employed for the X_{H2} and X_5 based on the optimal solution attained from the proposed approach, and under different uncertainty scenarios.

3) $X_{H2,RN}$ and $X_{5,RN}$ in Figure 3.10 in the main text are computed through the process model with respect to the optimal sequence of MVs and under the nominal scenario ($\rho = 12.6$ and $X_3 = 0.85$). The subscript RN is used for X_{H2} and X_5 based on the optimal solution obtained from the proposed RNN-based approach, and under the nominal scenario. The computation of $X_{H2,RN}$ and $X_{5,RN}$ is in a similar way to the computation of $X_{H2,RU}$ and $X_{5,RU}$ mentioned in the previous step.

The joint constraint satisfaction probabilities presented in Table A13 are calculated according to the following steps:

1) The column vectors of the matrices of $X_{H2,RU}$ and $X_{5,RU}$ are extracted with respect to 20 time points ($t = 0.1 \sim 2$ with an interval of 0.1). These column vectors can be deemed as 1000-element vectors of $X_{H2,j}$ and $X_{5,j}$ at $j = 1, ..., 20$.

2) The vectors of $X_{H2,j}$ and $X_{5,j}$ gained from the previous step are plugged into $-X_{H2,j} + 0.7$ and $-X_{5,j} + 0.9$, respectively.

3) For a certain sampling instant j, $-X_{H2,j} + 0.7$ and $-X_{5,j} + 0.9$ can be deemed as two 1000-element vectors, and be denoted as $P_{1,H,j}$ and $P_{2,H,j}$, respectively.

4) For each sampling instant j, the 1000-element vector $\bar{P}_{H,j}$ is generated through $\bar{P}_{H,j} = \max\{P_{1,H,j}, P_{2,H,j}\}$.

5) For a certain sampling instant j, the percentage of elements in $\bar{P}_{H,j}$ less or equal to 0 is calculated, and such the percentage is the joint constraint satisfaction probability at a certain sampling instant shown in Table A13.

On the other hand, the optimal objective value mentioned in Section 3.2.5.2 in the main text is calculated by plugging the optimal sequence of MVs into (A10a).

A2.4 SMPC implementation of the HDS case study

A2.4.1 NN-LSTM for the SMPC of the HDS

To train, validate, and test the NN-LSTM model for the SMPC of the HDS, 2.4×10^5 data samples are generated with respect to different sequences of MVs and different sets of initial conditions. The overall time for data generation is around 9600 seconds. Each sample consists of a sequence of quantile values paired with a corresponding sequence of MVs and a corresponding set of initial conditions. Out of all the samples, 2×10^5, 2×10^4, and 2×10^4 samples are used for training, validation, and test, respectively. The architecture of the NN-LSTM is determined through the 10-fold cross-validation based on the training set first, and then the NN-LSTM is trained on the entire training set with the use of the validation set for the early stopping. The stacked NN-LSTM is constructed and trained by using the functional API in Keras. The training time for the NN-LSTM is around 5300 seconds. The mean absolute percentage errors (MAPEs) of the NN-LSTM based on the testing data set are shown in the following table.

Table A14: Mean absolute percentage errors (MAPEs) of the NN-LSTM based on the testing data set, with respect to the predictions at different sampling instants

j	1	2	3	4	5
MAPE	1.39 %	1.46 %	1.72 %	2.68 %	2.69 %
j	6	7	8	9	10
MAPE	2.88 %	3.01 %	3.35 %	4.38 %	4.63 %
j	11	12	13	14	15
MAPE	2.98 %	3.17 %	3.25 %	3.89 %	4.51 %
j	16	17	18	19	20
MAPE	4.06 %	4.01 %	4.32 %	4.88 %	4.72 %

After training the NN-LSTM, the NN-LSTM is incorporated into the discretized SOCP of the HDS mentioned in Section A2.3.1, and the corresponding formulation is given as:

$$\min_{\dot{F}_{1,j},\dot{F}_{2,j},\dot{F}_{10,j}} \quad \sum_{j=0}^{19}(C_{H4}X_1\dot{F}_{1,j} + C_{H3}X_2\dot{F}_{2,j}) \times 0.1 \tag{A11a}$$

$$\text{s.t.} \quad \hat{Q}^{1-\varepsilon}_{H,j} \leq 0 \quad j = 1, ..., 20 \tag{A11b}$$

$$\hat{Q}^{1-\varepsilon}_{H,j=1,...,20} = NL_H(\dot{F}_{1,j=0,...,19}, \dot{F}_{2,j=0,...,19}, \dot{F}_{10,j=0,...,19}, \dot{F}^{H2}_{x,0}, X_{H2,0}) \tag{A11c}$$

$$0 \leq \dot{F}_{1,j} \leq 1400 \qquad j = 0, ..., 19 \tag{A11d}$$

$$0 \leq \dot{F}_{2,j} \leq 790 \qquad j = 0, ..., 19 \tag{A11e}$$

$$0 \leq \dot{F}_{10,j} \leq 1500 \qquad j = 0, ..., 19 \tag{A11f}$$

$$\dot{F}^{H2}_{x,0} = \dot{F}^{H2}_{x,t_0} \tag{A11g}$$

$$X_{H2,0} = X_{H2,t_0} \tag{A11h}$$

where $\hat{Q}^{1-\varepsilon}_{H,j}$ is the quantile value predicted from the NN-LSTM at a certain sampling instant j. NL_H is the NN-LSTM model that takes the sequence of MVs and the initial conditions as inputs. $\dot{F}^{H2}_{x,0}$ and $X_{H2,0}$ are the initial conditions based on the measurements at the current sampling instant t_0. $\dot{F}^{H2}_{x,t_0}$ and X_{H2,t_0} are the hydrogen consumption rate and hydrogen mole fraction in the reactor, that are acquired at the current sampling instant t_0, respectively.

A2.4.2 Steps for the SMPC implementation of the HDS

The SMPC implementation of the HDS is carried out according to the following steps:

1) The discretized SOCP mentioned in Section A2.3.1 is first solved at $t = 0$. Since $\dot{F}^{H2}_{x}$ and X_{H2} are respectively equal to 682.5 and 0.9 at $t = 0$, they are used as the initial conditions.

2) The first element of the obtained optimal sequence of MVs are applied to the process model including (3.27b)-(3.27f) in the main text. The computation of the process model is also based on the initial conditions, a realization of ρ, and a realization of X_3. The computation of the process model is carried out through the trapezoidal rule method with 50 time intervals, from $t = 0 \sim 0.1$. The computational results at the final time point are used as the initial conditions for the SOCP at the next sampling instant.

3) The SOCP is solved at $t = 0.1$ with the initial conditions computed from the previous step.

4) The process model is computed in the same way mentioned in step 2, from $t = 0.1 \sim 0.2$. Meanwhile, the computation of the process model is based on the first element of the optimal sequence of MVs obtained from step 3, the initial conditions gained from step 2, a realization of ρ, and a realization of X_3. Again, the computational results at the final time point are used as the initial conditions for the next sampling instant.

5) The above procedure is repeated until $t = 2$. Then, the SMPC implementation of the HDS is completed once.

A2.4.3 Supplementary results of the SMPC implementation

1000 SMPC executions are completed by repeating the procedure illustrated in the previous subsection 1000 times. The gained results are shown in the following table.

Table A15: The joint constraint satisfaction probabilities at different sampling instants based on the 1000 SMPC implementations of the HDS

t (h)	0.1	0.2	0.3	0.4	0.5
$Pr\left\{\begin{matrix} X_{H2} \geq 0.7 \\ X_5 \geq 0.9 \end{matrix}\right\}$	94.03 %	93.06 %	80.19 %	89.11 %	89.60 %
t (h)	0.6	0.7	0.8	0.9	1.0
$Pr\left\{\begin{matrix} X_{H2} \geq 0.7 \\ X_5 \geq 0.9 \end{matrix}\right\}$	87.82 %	89.60 %	89.60 %	87.98 %	88.79 %
t (h)	1.1	1.2	1.3	1.4	1.5
$Pr\left\{\begin{matrix} X_{H2} \geq 0.7 \\ X_5 \geq 0.9 \end{matrix}\right\}$	88.87 %	89.19 %	88.87 %	89.35 %	88.55 %
t (h)	1.6	1.7	1.8	1.9	2.0
$Pr\left\{\begin{matrix} X_{H2} \geq 0.7 \\ X_5 \geq 0.9 \end{matrix}\right\}$	89.35 %	88.15 %	88.39 %	88.87 %	87.90 %

A3 Chapter 4 supplementary materials

A3.1 DRCCP based on Wasserstein ambiguity set

In literature, the DRCCP based on the Wasserstein ambiguity set (Wasserstein DRCCP) has been widely studied. As a comparison to the proposed kernel DRCCP method, we introduce the Wasserstein DRCCP in this section.

A3.1.1 Wasserstein ambiguity set

The Wasserstein ambiguity set $\mathcal{D}_W$ constructed by exploiting the type-1 Wasserstein distance is defined as (A12). The Wasserstein distance is explained in detail in Section 1.2.5.

$$\mathcal{D}_W = \left\{ \mathbb{P} \in \mathcal{P}(\Xi) : W_1(\mathbb{P}, \hat{\mathbb{P}}_0) \leq \varepsilon_W \right\} \tag{A12}$$

where $\mathcal{P}(\Xi)$ is the set of Borel probability measures supported on Ξ, which contains all candidate probability distributions $\mathbb{P}$. The nominal distribution $\hat{\mathbb{P}}_0$ is an empirical distribution which is determined based on the collected samples of the random parameter vector. ε_W is the Wasserstein radius which is a user-defined parameter. The solution conservatism of the DRCCP over the Wasserstein ambiguity set can be controlled by adjusting ε_W. Moreover, to enable a tractable form for the DRCCP based on the Wasserstein ambiguity set, the Wasserstein ambiguity set can be combined with the CVaR approximation to address the DRJCC in the underlying DRCCP. The DRCCP based on the Wasserstein ambiguity set and the CVaR approximation (CVaR-based Wasserstein DRCCP) is explained in detail in A3.1.5. The CVaR-based Wasserstein DRCCP employed in this work relies on the assumption that the constraint functions in the JCC are affine in random parameters, i.e., $g_i(x, \xi) = h_i(x)^T \xi + h_i^0(x)$. Note that all the Wasserstein distances mentioned in the following parts of this study refer to the 1-Wasserstein distance.

A3.1.2 Worst-case expectation based on Wasserstein ambiguity set

To deal with the DRCCP based on the Wasserstein ambiguity set, the worst-case expectation problem based on the Wasserstein ambiguity set should be first studied, which is given as:

$$\begin{aligned} \max_{\mathbb{P}} \quad & \mathbb{E}_{\mathbb{P}}\left[L(\xi)\right] \\ \text{s.t.} \quad & W_1(\mathbb{P}, \hat{\mathbb{P}}_0) \leq \varepsilon_W \end{aligned} \tag{A13}$$

where $L(\xi)$ is the loss function.

The Wasserstein distance W_1 can be formulated equivalently as the following optimal transport problem [180]:

$$\begin{aligned} W_1(\mathbb{P}, \hat{\mathbb{P}}_0) = \min_{\pi \geq 0} & \sum_{h=1}^{H} \sum_{j=1}^{N} \left\| \xi_h - \xi_{0,j} \right\| \pi_{hj} \\ \text{s.t.} \quad & \sum_{h=1}^{H} \pi_{hj} = \frac{1}{N}, \quad \forall j \\ & \sum_{j=1}^{N} \pi_{hj} = p_h, \quad \forall h \end{aligned} \tag{A14}$$

In the above problem, $\mathbb{P}$ and $\hat{\mathbb{P}}_0$ are discrete distributions. The candidate distributions $\mathbb{P}$ containing the true distribution are assumed to be based on the finite support $\Xi = \{\xi_1, ..., \xi_H\}$, with the probability mass p_h for ξ_h. The nominal distribution $\hat{\mathbb{P}}_0$ is based on the collected samples $\{\xi_{0,1}, ..., \xi_{0,N}\}$, and every sample has equal probability mass $\frac{1}{N}$. π_{hj} is the joint probability of ξ_h and $\xi_{0,j}$. Then, by plugging the above optimal transport expression into problem (A13), we have:

$$\begin{aligned}
\max_{p\geq 0,\pi\geq 0} \quad & \sum_{h=1}^{H} L(\xi_h)p_h && \text{(A15a)}\\
\text{s.t.} \quad & \min_{\pi\geq 0} \quad \sum_{h=1}^{H}\sum_{j=1}^{N} \left\|\xi_h-\xi_{0,j}\right\| \pi_{hj} \leq \varepsilon_W && \text{(A15b)}\\
& \qquad \text{s.t.} \quad \sum_{h=1}^{H} \pi_{hj} = \frac{1}{N}, \quad \forall j && \text{(A15c)}\\
& \qquad\qquad\quad \sum_{j=1}^{N} \pi_{hj} = p_h, \quad \forall h && \text{(A15d)}
\end{aligned}$$

By replacing p_h in (A15a) with the expression shown in (A15d) and omitting the min operator in inequality (A15b), we can obtain:

$$\begin{aligned}
\max_{\pi\geq 0} \quad & \sum_{h=1}^{H}\sum_{j=1}^{N} L(\xi_h)\pi_{hj} && \text{(A16a)}\\
\text{s.t.} \quad & \sum_{h=1}^{H}\sum_{j=1}^{N} \left\|\xi_h-\xi_{0,j}\right\| \pi_{hj} \leq \varepsilon_W && \text{(A16b)}\\
& \sum_{h=1}^{H} \pi_{hj} = \frac{1}{N}, \quad \forall j && \text{(A16c)}
\end{aligned}$$

Subsequently, after introducing dual variables κ and z_j corresponding respectively to inequality (A16b) and equation (A16c), the dual problem of the above optimization can be gained:

$$\begin{aligned}
\min_{\kappa\geq 0, z_j} \quad & \kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j \\
\text{s.t.} \quad & z_j \geq L(\xi_h) - \kappa\left\|\xi_h-\xi_{0,j}\right\|, \quad \forall j, \forall h && \text{(A17)}
\end{aligned}$$

Note that $\kappa \geq 0$ since κ corresponds to the inequality constraint. The above problem can be rewritten as:

$$\begin{aligned} \min_{\kappa \geq 0, z_j} \quad & \kappa \varepsilon_W + \frac{1}{N} \sum_{j=1}^{N} z_j \\ \text{s.t.} \quad & z_j \geq \max_{h} L(\xi_h) - \kappa \left\| \xi_h - \xi_{0,j} \right\|, \quad \forall j \end{aligned} \tag{A18}$$

While $H \to \infty$, infinite number of samples are considered that the above optimization model is extended to the continuous support Ξ. Accordingly, the above optimization can be rewritten as:

$$\min_{\kappa \geq 0, z_j} \quad \kappa \varepsilon_W + \frac{1}{N} \sum_{j=1}^{N} z_j \tag{A19a}$$

$$\text{s.t.} \quad z_j \geq \max_{\xi \in \Xi} L(\xi) - \kappa \left\| \xi - \xi_{0,j} \right\|, \quad \forall j \tag{A19b}$$

According to the definition of dual norm, we can obtain $\kappa \left\| \xi - \xi_{0,j} \right\| = \max_{\|V_j\|_* \leq \kappa} \langle V_j, \xi - \xi_{0,j} \rangle$. Thus, $-\kappa \left\| \xi - \xi_{0,j} \right\|$ in the above problem can be replaced with $\min_{\|V_j\|_* \leq \kappa} -\langle V_j, \xi - \xi_{0,j} \rangle$. Afterwards, inequality (A19b) can be rewritten as $z_j \geq \max_{\xi \in \Xi} L(\xi) + \min_{\|V_j\|_* \leq \kappa} -\langle V_j, \xi - \xi_{0,j} \rangle$. Moreover, the min operator on the right-hand side of this inequality can be eliminated, and then we plug this rewritten inequality back into inequality (A19b) to attain the following problem:

$$\min_{\kappa \geq 0, z_j, V_j} \quad \kappa \varepsilon_W + \frac{1}{N} \sum_{j=1}^{N} z_j \tag{A20a}$$

$$\text{s.t.} \quad z_j \geq \max_{\xi \in \Xi} L(\xi) - \langle V_j, \xi - \xi_{0,j} \rangle, \quad \forall j = 1, ..., N \tag{A20b}$$

$$\left\| V_j \right\|_* \leq \kappa, \quad \forall j = 1, ..., N \tag{A20c}$$

The condition $\kappa \geq 0$ in the above optimization can be removed because of inequality (A20c). Also, $\langle V_j, \xi - \xi_{0,j} \rangle$ in inequality (A20b) can be rewritten equivalently as $\langle V_j, \xi \rangle - \langle V_j, \xi_{0,j} \rangle$.

Subsequently, the above problem can be reformulated as:

$$\min_{\kappa, z_j, V_j} \quad \kappa \varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j \tag{A21a}$$

$$\text{s.t.} \quad z_j \geq \max_{\xi \in \Xi} L(\xi) - \langle V_j, \xi \rangle + \langle V_j, \xi_{0,j} \rangle\,, \quad \forall j = 1, ..., N \tag{A21b}$$

$$\|V_j\|_* \leq \kappa, \quad \forall j = 1, ..., N \tag{A21c}$$

A3.1.3 Special case 1

The model shown above can be further simplified if the loss function is a convex piece-wise linear function $L(\xi) = \max_{i=1,...,w} a_i^T \xi + b_i$ and the support set is $\Xi \subseteq \mathbb{R}^w$. After plugging the piece-wise linear loss function into inequality (A21b), we have

$$\max_{\xi \in \Xi} \max_{i=1,...,w} a_i^T \xi + b_i - \langle V_j, \xi \rangle + \langle V_j, \xi_{0,j} \rangle \leq z_j, \forall j = 1, ..., N$$

This expression can be further rearranged as :

$$\max_{\xi \in \Xi} (a_i^T \xi - \langle V_j, \xi \rangle) + b_i + \langle V_j, \xi_{0,j} \rangle \leq z_j, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \tag{A22}$$

For each i and each j, the maximization in the above inequality is a linear programming (LP) problem given as:

$$\max_{\xi \in \Xi} \quad (a_i - V_j)^T \xi \tag{A23}$$

Since the above LP problem is unbounded that would give infinite value if $a_i - V_j \neq 0$, $a_i - V_j$ should be equal to zero to make (A23) bounded. Accordingly, inequality (A22) can be simplified as:

$$b_i + \langle a_i, \xi_{0,j} \rangle \leq z_j, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \tag{A24}$$

After replacing inequality (A21b) with inequality (A24) and plugging $a_i = V_j$ into inequality (A21c), the optimization problem in equation (A21a)-(A21c) can be reformulated as a convex problem which is given as:

$$
\begin{aligned}
\min_{\kappa, z_j} \quad & \kappa \varepsilon_W + \frac{1}{N} \sum_{j=1}^{N} z_j \\
\text{s.t.} \quad & b_i + a_i^T \xi_{0,j} \leq z_j, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \\
& \|a_i\|_* \leq \kappa, \quad \forall i = 1, ..., w, \forall j = 1, ..., N
\end{aligned} \tag{A25}
$$

The above formulation of the worst-case expectation problem is based on the assumptions that the loss function $L(\xi)$ is piece-wise linear and the support set is $\Xi \subseteq \mathbb{R}^w$.

A3.1.4 Special case 2

In addition to the convex piece-wise linear loss function assumption made in the previous subsection (special case 1) $L(\xi) = \max_{i=1,...,w} a_i^T \xi + b_i$, if the support set is further assumed polyhedral $\Xi = \{\xi \in \mathbb{R}^w : C\xi \leq d\}$, the maximization for each i and each j in inequality (A22) can be formulated as a LP problem given as:

$$
\max_{\xi \in \Xi} \quad (a_i - V_j)^T \xi \tag{A26a}
$$

$$
\text{s.t.} \quad C\xi \leq d \tag{A26b}
$$

After introducing a dual variable vector γ_{ij}, the dual problem of the above LP is given as:

$$
\begin{aligned}
\min_{\gamma_{ij}} \quad & d^T \gamma_{ij} \\
\text{s.t.} \quad & C^T \gamma_{ij} = a_i - V_j \\
& \gamma_{ij} \geq 0
\end{aligned} \tag{A27}
$$

Due to the strong duality, problem (A27) can replace the maximization in inequality (A22), and then inequality (A22) can be reformulated as follows:

$$
\min_{\gamma_{ij}} \quad d^T \gamma_{ij} + b_i + \langle V_j, \xi_{0,j} \rangle \leq z_j, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \tag{A28a}
$$

$$
C^T \gamma_{ij} = a_i - V_j, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \tag{A28b}
$$

$$
\gamma_{ij} \geq 0, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \tag{A28c}
$$

We can drop the min operator in inequality (A28a). By further replacing V_j with $a_i - C^T\gamma_{ij}$ according to (A28b), the formulation including (A28a)-(A28c) can be reformulated as:

$$b_i + a_i^T\xi_{0,j} + \gamma_{ij}^T(d - C\xi_{0,j}) \le z_j, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \tag{A29a}$$

$$\gamma_{ij} \ge 0, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \tag{A29b}$$

By replacing inequalities (A21b) and (A21c) with the above inequalities, the optimization problem in (A21a)-(A21c) can be formulated as a convex problem which is given as:

$$\begin{aligned} \min_{\kappa, z_j, \gamma_{ij}} \quad & \kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j \\ \text{s.t.} \quad & b_i + a_i^T\xi_{0,j} + \gamma_{ij}^T(d - C\xi_{0,j}) \le z_j, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \\ & \left\|a_i - C^T\gamma_{ij}\right\|_* \le \kappa, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \\ & \gamma_{ij} \ge 0, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \end{aligned} \tag{A30}$$

The above formulation of the worst-case expectation problem is based on the assumptions that the loss function $L(\xi)$ is piece-wise linear and the support set is polyhedral.

A3.1.5 DRCCP based on Wasserstein ambiguity set: CVaR method

To exploit the CVaR approximation to address the DRJCC, let's begin with the DRJCC expressed as:

$$\max_{\mathbb{P}(\xi)\in\mathcal{P}} Pr\left(\max_{i=1,...,w}\{g_i(x,\xi)\} > 0\right) \le \delta \tag{A31}$$

We apply an upper bounding function $\Phi\left(\frac{1}{-\beta}\max_{i=1,...,w}\{g_i(x,\xi)\}\right) = \left(\frac{1}{-\beta}\max_{i=1,...,w}\{g_i(x,\xi)\} + 1\right)^+$ (with $\beta < 0$) for $\mathbb{I}\left(\max_{i=1,...,w}\{g_i(x,\xi)\}\right)$. Assuming that $g_i(x,\xi)$ can be expressed as the affine form $h_i(x)^T\xi + h_i^0(x)$, the CVaR approximation for inequality (A31) is given as:

$$\max_{\mathbb{P}(\xi)\in\mathcal{P}}\min_{\beta\in\mathbb{R}}\left\{\beta + \frac{1}{\delta}\left\{-\beta\mathbb{E}_{\mathbb{P}(\xi)}\left(\left[\frac{1}{-\beta}\max_{i=1,...,w}\left\{(h_i(x)^T\xi + h_i^0(x))\right\} + 1\right]^+\right)\right\}\right\} \le 0 \tag{A32a}$$

$$\Rightarrow \max_{\mathbb{P}(\xi)\in\mathcal{P}}\min_{\beta\in\mathbb{R}}\left\{\beta + \frac{1}{\delta}\left\{\mathbb{E}_{\mathbb{P}(\xi)}\left(\left[\max_{i=1,...,w}\left\{(h_i(x)^T\xi + h_i^0(x))\right\} - \beta\right]^+\right)\right\}\right\} \le 0 \tag{A32b}$$

$\max_{\mathbb{P}(\xi)\in\mathcal{P}}$ and $\min_{\beta\in\mathbb{R}}$ in inequality (A32b) can be switched, and then inequality (A32b) can be rewritten as:

$$\min_{\beta\in\mathbb{R}} \left\{ \beta + \frac{1}{\delta} \max_{\mathbb{P}(\xi)\in\mathcal{P}} \mathbb{E}_{\mathbb{P}(\xi)} \left(\left[\max_{i=1,\ldots,w} \left\{ (h_i(x)^T\xi + h_i^0(x)) \right\} - \beta \right]^+ \right) \right\} \leq 0 \tag{A33}$$

Subsequently, by exploiting the formulation including (A21a)-(A21c), the inner problem $\max_{\mathbb{P}(\xi)\in\mathcal{P}} \mathbb{E}_{\mathbb{P}(\xi)} \left(\left[\max_{i=1,\ldots,w} \left\{ (h_i(x)^T\xi + h_i^0(x)) \right\} - \beta \right]^+ \right)$ in the above inequality can be reformulated as:

$$\min_{\kappa, z_j, V_j} \quad \kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j \tag{A34a}$$

$$\text{s.t.} \quad z_j \geq \max_{\xi\in\Xi} \left[\max_{i=1,\ldots,w} \left\{ (h_i(x)^T\xi + h_i^0(x)) \right\} - \beta \right]^+ - \langle V_j, \xi \rangle + \langle V_j, \xi_{0,j} \rangle, \quad \forall j = 1, ..., N \tag{A34b}$$

$$\|V_j\|_* \leq \kappa, \quad \forall j = 1, ..., N \tag{A34c}$$

Through replacing the worst-case expectation problem in inequality (A33) with the above formulation, the left-hand side of inequality (A33) can be rewritten as:

$$\min_{\beta} \quad \left\{ \beta + \frac{1}{\delta} \min_{\kappa, z_j, V_j} \left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j \right] \right\} \tag{A35a}$$

$$\text{s.t.} \quad z_j \geq \max_{\xi\in\Xi} \left[\max_{i=1,\ldots,w} \left\{ (h_i(x)^T\xi + h_i^0(x)) \right\} - \beta \right]^+ - \langle V_j, \xi \rangle + \langle V_j, \xi_{0,j} \rangle, \quad \forall j = 1, ..., N \tag{A35b}$$

$$\|V_j\|_* \leq \kappa, \quad \forall j = 1, ..., N \tag{A35c}$$

The above optimization problem can be rearranged as:

$$\min_{\beta,\kappa,z_j,V_j} \quad \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \tag{A36a}$$

$$\begin{aligned} \text{s.t.} \quad & z_j \geq \max_{\xi\in\Xi}\left[\max_{i=1,...,w}\left\{(h_i(x)^T\xi + h_i^0(x))\right\} - \beta\right]^+ \\ & \qquad - \langle V_j, \xi\rangle + \langle V_j, \xi_{0,j}\rangle, \quad \forall j = 1,...,N \qquad \text{(A36b)} \\ & \|V_j\|_* \leq \kappa, \quad \forall j = 1,...,N \qquad \text{(A36c)} \end{aligned}$$

In inequality (A36b), $\max_{\xi\in\Xi}\left[\max_{i=1,...,w}\left\{(h_i(x)^T\xi + h_i^0(x))\right\} - \beta\right]^+$ is a convex piece-wise linear function with respect to ξ. Further notice that

$$\left[\max_{i=1,...,w}\left\{(h_i(x)^T\xi + h_i^0(x))\right\} - \beta\right]^+ = \max\left\{\max_{i=1,...,w}\left\{(h_i(x)^T\xi + h_i^0(x))\right\} - \beta, 0\right\},$$

So, $\left[\max_{i=1,...,w}\left\{(h_i(x)^T\xi + h_i^0(x))\right\} - \beta\right]^+$ can be written in the form of $L(\xi) = \max_{i=1,...,w} a_i^T\xi + b_i$, with $a_1 = h_1(x), b_1 = h_1^0(x) - \beta, ..., a_w = h_w(x), b_w = h_w^0(x) - \beta, a_{w+1} = 0, b_{w+1} = 0$

If the support set of uncertainty $\Xi \subseteq \mathbb{R}^w$, inequalities (A36b)-(A36c) can be transformed to be the form shown in (A25). Then, by plugging $a_1 = h_1(x), b_1 = h_1^0(x) - \beta, ..., a_w = h_w(x), b_w = h_w^0(x) - \beta, a_{w+1} = 0, b_{w+1} = 0$ into the transformed inequalities, the model including (A36a)-(A36c) can be reformulated as the follows:

$$\begin{aligned} \min_{\beta,\kappa,z_j} \quad & \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \\ \text{s.t.} \quad & z_j \geq h_i^0(x) - \beta + h_i(x)^T\xi_{0,j}, \quad \forall i = 1,...,w, \forall j = 1,...,N \\ & z_j \geq b_{w+1} + a_{w+1}^T\xi_{0,j}, \quad \forall j = 1,...,N \\ & \|h_i(x)\|_* \leq \kappa, \quad \forall i = 1,...,w \\ & \|a_{w+1}(x)\|_* \leq \kappa \end{aligned} \tag{A37}$$

Subsequently, because $a_{w+1} = 0$ and $b_{w+1} = 0$, $a_{w+1}^T\xi + b_{w+1} = 0$. Then, formulation A37 can be rewritten as:

$$
\begin{aligned}
\min_{\beta,\kappa,z_j} \quad & \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] && \text{(A38a)}\\
\text{s.t.} \quad & z_j \geq h_i^0(x) - \beta + h_i(x)^T \xi_{0,j}, \quad \forall i = 1, ..., w, \forall j = 1, ..., N && \text{(A38b)}\\
& z_j \geq 0, \quad \forall j = 1, ..., N && \text{(A38c)}\\
& \|h_i(x)\|_* \leq \kappa, \quad \forall i = 1, ..., w && \text{(A38d)}\\
& \kappa \geq 0 && \text{(A38e)}
\end{aligned}
$$

Inequality (A38e) is redundant because of inequality (A38d), and thus inequality (A38e) can be removed. Then, by replacing the worst-case CVaR in inequality (A33) with the above optimization model and omitting the min operator in (A38a), we can obtain the following reformulation of inequality (A33):

$$
\begin{aligned}
& \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \leq 0\\
& z_j \geq h_i^0(x) - \beta + h_i(x)^T \xi_{0,j}, \quad \forall i = 1, ..., w, \forall j = 1, ..., N\\
& z_j \geq 0, \quad \forall j = 1, ..., N\\
& \|h_i(x)\|_* \leq \kappa, \quad \forall i = 1, ..., w \qquad \text{(A39)}
\end{aligned}
$$

Afterwards, we can attain the formulation of the DRCCP by plugging formulation (A39) into the DRCCP including (4.1a)-(4.1b) to replace the DRJCC stated in (4.1b). Therefore, the attained model of the DRCCP based on the Wasserstein ambiguity set and the CVaR approximation is given as:

$$
\begin{aligned}
\min_{x,\beta,\kappa,z_j} \quad & f(x)\\
\text{s.t.} \quad & \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \leq 0\\
& z_j \geq h_i^0(x) - \beta + h_i(x)^T \xi_{0,j}, \quad \forall i = 1, ..., w, \forall j = 1, ..., N\\
& z_j \geq 0, \quad \forall j = 1, ..., N\\
& \|h_i(x)\|_* \leq \kappa, \quad \forall i = 1, ..., w && \text{(A40)}
\end{aligned}
$$

The above formulation of the CVaR-based Wasserstein DRCCP is based on the following assumptions: 1) The constraint function involves convex piece-wise linearity; 2) The constraints in the joint chance constraint are affine in the uncertain parameters; 3) The support set is $\Xi \subseteq \mathbb{R}^w$.

On the other hand, if the support set of uncertainty is polyhedral ($\Xi = \{\xi \in \mathbb{R}^w : C\xi \leq d\}$), inequalities (A36b)-(A36c) can be transformed to be the forms of the inequalities shown in formulation (A30). Subsequently, by plugging $a_1 = h_1(x), b_1 = h_1^0(x) - \beta, ..., a_w = h_w(x), b_w = h_w^0(x) - \beta, a_{w+1} = 0, b_{w+1} = 0$ into the transformed inequalities, the model including (A36a)-(A36c) can be reformulated as the follows:

$$\begin{aligned}
\min_{\beta,\kappa,z_j,\gamma_{ij}} \quad & \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \\
\text{s.t.} \quad & z_j \geq h_i^0(x) - \beta + h_i(x)^T\xi_{0,j} + \gamma_{ij}^T(d - C\xi_{0,j}), \\
& \qquad\qquad\qquad\qquad \forall i = 1,...,w, \forall j = 1,...,N \\
& z_j \geq b_{w+1} + a_{w+1}^T\xi_{0,j} + \gamma_{w+1,j}^T(d - C\xi_{0,j}), \quad \forall j = 1,...,N \\
& \left\|h_i(x) - C^T\gamma_{ij}\right\|_* \leq \kappa, \quad \forall i = 1,...,w, \forall j = 1,...,N \\
& \left\|a_{w+1} - C^T\gamma_{w+1,j}\right\|_* \leq \kappa, \quad \forall j = 1,...,N \\
& \gamma_{ij} \geq 0, \quad \forall i = 1,...,w, j = 1,...,N \\
& \gamma_{w+1,j} \geq 0, \quad \forall j = 1,...,N
\end{aligned} \tag{A41}$$

Since $a_{w+1} = 0$ and $b_{w+1} = 0$, the above formulation can be rewritten as:

$$\min_{\beta,\kappa,z_j,\gamma_{ij}} \quad \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \tag{A42a}$$
$$\text{s.t.} \quad z_j \geq h_i^0(x) - \beta + h_i(x)^T\xi_{0,j} + \gamma_{ij}^T(d - C\xi_{0,j}), \tag{A42b}$$
$$\forall i = 1,...,w, \forall j = 1,...,N \tag{A42c}$$
$$z_j \geq \gamma_{w+1,j}^T(d - C\xi_{0,j}), \quad \forall j = 1,...,N \tag{A42d}$$
$$\left\|h_i(x) - C^T\gamma_{ij}\right\|_* \leq \kappa, \quad \forall i = 1,...,w, \forall j = 1,...,N \tag{A42e}$$
$$\left\|-C^T\gamma_{w+1,j}\right\|_* \leq \kappa, \quad \forall j = 1,...,N \tag{A42f}$$
$$\gamma_{ij} \geq 0, \quad \forall i = 1,...,w, j = 1,...,N \tag{A42g}$$
$$\gamma_{w+1,j} \geq 0, \quad \forall j = 1,...,N \tag{A42h}$$

Since $\gamma_{w+1,j}^T(d - C\xi_{0,j})$ in inequality (A42d) and $\left\|-C^T\gamma_{w+1,j}\right\|_*$ in inequality (A42f) are non-negative, the feasible set constructed by inequalities (A42d), (A42f), and (A42h), are the same as the feasible set constructed by $z_j \geq 0$, $\kappa \geq 0$, and $\gamma_{w+1,j} = 0$. Moreover, $\kappa \geq 0$ is redundant due to inequality (A42e). Thus, the above optimization model can be reduced as:

$$\min_{\beta,\kappa,z_j,\gamma_{ij}} \quad \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \tag{A43a}$$

$$\text{s.t.} \quad z_j \geq h_i^0(x) - \beta + h_i(x)^T\xi_{0,j} + \gamma_{ij}^T(d - C\xi_{0,j}), \quad \forall i = 1,...,w, \forall j = 1,...,N \tag{A43b}$$

$$\left\|h_i(x) - C^T\gamma_{ij}\right\|_* \leq \kappa, \quad \forall i = 1,...,w, \forall j = 1,...,N \tag{A43c}$$

$$\gamma_{ij} \geq 0, \quad \forall i = 1,...,w, j = 1,...,N \tag{A43d}$$

$$z_j \geq 0, \quad \forall j = 1,...,N \tag{A43e}$$

By replacing the worst-case CVaR in inequality (A33) with the above optimization model and omitting the min operator in (A43a), we can obtain the following reformulation of inequality (A33):

$$\begin{aligned}
&\beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \leq 0 \\
&z_j \geq h_i^0(x) - \beta + h_i(x)^T\xi_{0,j} + \gamma_{ij}^T(d - C\xi_{0,j}), \\
&\qquad\qquad \forall i = 1,...,w, \forall j = 1,...,N \\
&\left\|h_i(x) - C^T\gamma_{ij}\right\|_* \leq \kappa, \quad \forall i = 1,...,w, \forall j = 1,...,N \\
&\gamma_{ij} \geq 0, \quad \forall i = 1,...,w, j = 1,...,N \\
&z_j \geq 0, \quad \forall j = 1,...,N
\end{aligned} \tag{A44}$$

Finally, by plugging formulation (A44) into the DRCCP including (4.1a)-(4.1b), we can have the CVaR-based Wasserstein DRCCP formulation given as:

$$\begin{aligned}
\min_{x,\beta,\kappa,z_j,\gamma_{ij}} \quad & f(x) \\
\text{s.t.} \quad & \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{N}\sum_{j=1}^{N} z_j\right] \leq 0 \\
& z_j \geq h_i^0(x) - \beta + h_i(x)^T \xi_{0,j} + \gamma_{ij}^T(d - C\xi_{0,j}), \\
& \qquad\qquad\qquad\qquad \forall i = 1, ..., w, \forall j = 1, ..., N \\
& \left\| h_i(x) - C^T\gamma_{ij} \right\|_* \leq \kappa, \quad \forall i = 1, ..., w, \forall j = 1, ..., N \\
& \gamma_{ij} \geq 0, \quad \forall i = 1, ..., w, j = 1, ..., N \\
& z_j \geq 0, \quad \forall j = 1, ..., N
\end{aligned} \tag{A45}$$

The above formulation of the CVaR-based Wasserstein DRCCP is based on the following assumptions: 1) The worst-case expectation problem in the CVaR approximation involves piece-wise linearity; 2) The constraints in the joint chance constraint are affine in the random parameters; 3) The support set is polyhedral.

A4 Chapter 5 supplementary materials

A4.1 Dual of the worst-case expectation problem

We first consider a worst-case expectation problem based on the Sinkhorn ambiguity set $\mathcal{P}_{\gamma_s,\varepsilon_s}$ over support Ξ:

$$\max_{\mathbb{P}\in\mathcal{P}_{\gamma_s,\varepsilon_s}} \quad \mathbb{E}_{\mathbb{P}}\left[\mathcal{L}(x,\eta)\right] \tag{A46a}$$

$$\text{s.t.} \quad \mathcal{P}_{\gamma_s,\varepsilon_s} = \left\{\mathbb{P} : \mathcal{W}_{\gamma_s}(\mathbb{P},\hat{\mathbb{P}}_0) \le \varepsilon_s\right\} \tag{A46b}$$

where $\mathcal{L}$ is the loss function. x is the decision variable for the minimization problem outside the worst-case expectation problem, and it is treated as a parameter here. $\mathbb{P}$ and $\hat{\mathbb{P}}_0$ are the candidate and nominal distributions, respectively. $\mathbb{P}$ and $\hat{\mathbb{P}}_0$ are based on random parameter vectors η and ξ, respectively. $\eta \in \Xi$ and $\xi \in \Xi$. The nominal distribution is an empirical distribution established by the collected data points $\{\xi_m\}_{m=1}^{M}$, i.e., $\hat{\mathbb{P}}_0 = \dfrac{1}{M}\sum_{m=1}^{M}\delta_{\xi_m}$ (δ_{ξ_m} denotes the one-point probability distribution supported on $\{\xi_m\}$). According to the definition of the Sinkhorn distance in Equation 5.5 (in the main text) and based on the choice of reference measure $B = \hat{\mathbb{P}}_0$, the above model is equivalent to

$$\begin{aligned}\max_{\mathbb{P}} \quad & \mathbb{E}_{\mathbb{P}}\left[\mathcal{L}(x,\eta)\right]\\ \text{s.t.} \quad & \mathbb{E}_{\pi}\left[c(\eta,\xi) + \gamma_s \log\left(\frac{d\pi(\eta,\xi)}{dA(\eta)d\hat{\mathbb{P}}_0(\xi)}\right)\right] \le \varepsilon_s\end{aligned} \tag{A47}$$

The joint distribution $\pi = \dfrac{1}{M}\sum_{m=1}^{M}\delta_{\xi_m}\otimes\pi_m$, where π_m is the conditional distribution of π given the first marginal of π equals ξ_m. $c(\cdot,\cdot)$ is the same cost function as used in (1.7) and (5.5) (in the main text). $A(\eta)$ is a Lebesgue measure [181] supported on Ξ that enables all the candidate distributions $\mathbb{P}$ absolutely continuous with respect to $A(\eta)$, to ensure the entropic regularization in the Sinkhorn distance for the Sinkhorn ambiguity set well-defined [17]. Accordingly, the expectation in (A47) can be rewritten as:

$$\frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\pi_m}\left[c(\eta,\xi_m) + \gamma_s \log\left(\frac{d\pi_m(\eta)}{dA(\eta)}\right)\right] \tag{A48}$$

Introduce a kernel probability distribution $\mathbb{Q}_m$

$$d\mathbb{Q}_m(\eta) := \frac{\exp(-c(\eta,\xi_m)/\gamma_s)}{\int \exp(-c(\tau,\xi_m)/\gamma_s)dA(\tau)} dA(\eta) \tag{A49}$$

and use the following change-of-measure identity [64]

$$\log\left(\frac{d\pi_m(\eta)}{dA(\eta)}\right) = \log\left(\frac{d\mathbb{Q}_m(\eta)d\pi_m(\eta)}{dA(\eta)d\mathbb{Q}_m(\eta)}\right) = \log\left(\frac{d\mathbb{Q}_m(\eta)}{dA(\eta)}\right) + \log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right) \tag{A50}$$

(A48) is converted to

$$\frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\pi_m}\left[c(\eta,\xi_m) + \gamma_s \log\left(\frac{d\mathbb{Q}_m(\eta)}{dA(\eta)}\right) + \gamma_s \log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)\right] \tag{A51}$$

Then (A51) becomes:

$$\frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\pi_m}\left[c(\eta,\xi_m) + \gamma_s \log\left(\frac{\exp\left(-c(\eta,\xi_m)/\gamma_s\right)}{\int \exp\left(-c(\tau,\xi_m)/\gamma_s\right) dA(\tau)}\right) + \gamma_s \log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)\right] \tag{A52}$$

By combining the first two terms in the expectation, we can get:

$$\frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\pi_m}\left[-\gamma_s \log\left(\int \exp\left(-c(\eta,\xi_m)/\gamma_s\right) dA(\eta)\right) + \gamma_s \log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)\right] \tag{A53}$$

After plugging the above formulation into (A47) and rearranging the inequality, we can gain:

$$\frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\pi_m}\left[\gamma_s \log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)\right] \le \varepsilon_s + \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\pi_m}\left[\gamma_s \log\left(\int \exp\left(-c(\eta,\xi_m)/\gamma_s\right) dA(\eta)\right)\right] \tag{A54}$$

In the right hand side of the above inequality, $\int \exp\left(-c(\eta,\xi_m)/\gamma_s\right) dA(\eta)$ is essentially an expected value with respect to η. Thus, the second term in the right hand side of the above inequality is equal to $\frac{1}{M}\sum_{m=1}^{M} \gamma_s \log\left(\int \exp\left(-c(\eta,\xi_m)/\gamma_s\right) dA(\eta)\right)$.

Define the following new variable $\bar{\varepsilon}_s$ to replace the right hand side

$$\bar{\varepsilon}_s := \varepsilon_s + \frac{1}{M}\sum_{m=1}^{M} \gamma_s \log\left(\int \exp\left(\frac{-c(\eta,\xi_m)}{\gamma_s}\right) dA(\eta)\right) \tag{A55}$$

Then, the constraint in (A46b) can be reformulated as:

$$\frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\pi_m}\left[\gamma_s \log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)\right] \leq \bar{\varepsilon}_s \tag{A56}$$

On the other hand, the objective function in (A46a) can be reformulated as:

$$\max_{\pi_m \in \mathcal{P}_{\gamma_s,\varepsilon_s}, m=1,\ldots,M} \quad \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\pi_m}\left[\mathcal{L}(x,\eta)\right]$$

Then, the worst-case expectation problem including (A46a)-(A46b) becomes:

$$\begin{aligned}\max_{\pi_m \in \mathcal{P}_{\gamma_s,\varepsilon_s}, m=1,\ldots,M} \quad & \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\pi_m}\left[\mathcal{L}(x,\eta)\right] \\ \text{s.t.} \quad & \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\pi_m}\left[\gamma_s \log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)\right] \leq \bar{\varepsilon}_s\end{aligned} \tag{A57}$$

Let $L_m(\eta) = \dfrac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}$. Then

$$\begin{aligned}\mathbb{E}_{\pi_m}\left[\mathcal{L}(x,\eta)\right] &= \int_{\Xi}\mathcal{L}(x,\eta)d\pi_m(\eta) = \int_{\Xi}\mathcal{L}(x,\eta)\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}d\mathbb{Q}_m(\eta) \\ &= \int_{\Xi}\mathcal{L}(x,\eta)L_m(\eta)d\mathbb{Q}_m(\eta) = \mathbb{E}_{\mathbb{Q}_m}\left[\mathcal{L}(x,\eta)L_m(\eta)\right] \\ \mathbb{E}_{\pi_m}\left[\log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)\right] &= \int_{\Xi}\log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)d\pi_m(\eta) = \int_{\Xi}\log\left(\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}\right)\frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)}d\mathbb{Q}_m(\eta) \\ &= \int_{\Xi}\log\left(L_m(\eta)\right)L_m(\eta)d\mathbb{Q}_m(\eta) = \mathbb{E}_{\mathbb{Q}_m}\left[\log(L_m(\eta))L_m(\eta)\right]\end{aligned} \tag{A58}$$

Based on the above, (A57) can be rewritten as:

$$\begin{aligned}\max_{L_m(\eta)\in\mathbb{L}, m=1,\ldots,M} \quad & \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\mathbb{Q}_m}\left[\mathcal{L}(x,\eta)L_m(\eta)\right] \\ \text{s.t.} \quad & \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\mathbb{Q}_m}\left[\gamma_s \log\left(L_m(\eta)\right)L_m(\eta)\right] \leq \bar{\varepsilon}_s\end{aligned} \tag{A59}$$

Note that $\mathbb{L} = \{L_m(\eta) \in \mathbb{L}(\mathbb{Q}_m(\eta)) : \mathbb{E}_{\mathbb{Q}_m}\left[L_m(\eta)\right] = 1, L_m(\eta) \geq 0, \forall m = 1, ..., M\}$. Note that

$\mathbb{E}_{\mathbb{Q}_m}[L_m(\eta)] = 1$ because $\mathbb{E}_{\mathbb{Q}_m}[L_m(\eta)] = \int \frac{d\pi_m(\eta)}{d\mathbb{Q}_m(\eta)} d\mathbb{Q}_m(\eta) = \int d\pi_m(\eta) = 1$. Since the above optimization is a maximization problem involving the linear objective function and the convex constraint, the above problem is convex. Thus, the strong dual problem of the above optimization can be obtained:

$$\min_{\lambda \geq 0} \max_{L_m(\eta) \in \mathbb{L}, m=1,\dots,M} \left\{ \lambda \bar{\varepsilon}_s + \frac{1}{M} \sum_{m=1}^{M} \mathbb{E}_{\mathbb{Q}_m} \left[\mathcal{L}(x,\eta) L_m(\eta) - \lambda \gamma_s \log\left(L_m(\eta)\right) L_m(\eta) \right] \right\} \tag{A60}$$

where $\lambda \geq 0$ is the dual variable corresponding to the inequality constraint in (A59). Let's first focus on the inner maximization of (A60):

$$\begin{aligned} \max_{L_m(\eta) \in \mathbb{L}^0} \quad & \mathbb{E}_{\mathbb{Q}_m} \left[\mathcal{L}(x,\eta) L_m(\eta) - \lambda \gamma_s \log\left(L_m(\eta)\right) L_m(\eta) \right] \\ \text{s.t.} \quad & \mathbb{E}_{\mathbb{Q}_m} [L_m(\eta)] = 1 \end{aligned} \tag{A61}$$

Notably, $\mathbb{L}^0 = \{L_m(\eta) \in \mathbb{L}(\mathbb{Q}_m(\eta)) : L_m(\eta) \geq 0, \forall m = 1, \dots, M\}$. Since problem (A61) is a maximization problem involving the concave objective function and the affine equality constraint, it is a convex optimization problem. Then, the strong dual problem of (A61) is given as:

$$\min_{\psi_m} \max_{L_m(\eta) \in \mathbb{L}^0} \mathbb{E}_{\mathbb{Q}_m} \left[\mathcal{L}(x,\eta) L_m(\eta) - \lambda \gamma_s \log\left(L_m(\eta)\right) L_m(\eta) + \psi_m (L_m(\eta) - 1) \right] \tag{A62}$$

where ψ_m is the dual variable corresponding to the equality constraint in (A61). Since the inner maximization problem in (A62) is unconstrained, its optimality condition is

$$\begin{aligned} & \frac{\partial \left\{ \mathcal{L}(x,\eta) L_m(\eta) - \lambda \gamma_s \log\left(L_m(\eta)\right) L_m(\eta) + \psi_m (L_m(\eta) - 1) \right\}}{\partial L_m(\eta)} = 0 \\ & \mathcal{L}(x,\eta) - \lambda \gamma_s \log\left(L_m(\eta)\right) - \lambda \gamma_s + \psi_m = 0 \\ & \Longrightarrow L_m^*(\eta, \psi_m^*) = \exp\left(\frac{\mathcal{L}(x,\eta) + \psi_m^* - \lambda \gamma_s}{\lambda \gamma_s} \right) \end{aligned}$$

L_m^* and ψ_m^* are optimums of L_m and ψ_m, respectively. After plugging the expression of

$L_m^*(\eta, \psi_m^*)$ into the equality constraint in (A61), we can attain:

$$\begin{aligned}
&\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)+\psi_m^*-\lambda\gamma_s}{\lambda\gamma_s}\right)\right]=1\\
&\Longrightarrow \mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\exp\left(\frac{\psi_m^*-\lambda\gamma_s}{\lambda\gamma_s}\right)=1\\
&\Longrightarrow \mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]=\exp\left(\frac{\lambda\gamma_s-\psi_m^*}{\lambda\gamma_s}\right)\\
&\Longrightarrow \log\left\{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\right\}=\frac{\lambda\gamma_s-\psi_m^*}{\lambda\gamma_s}\\
&\Longrightarrow \psi_m^*=-\lambda\gamma_s\log\left\{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\right\}+\lambda\gamma_s
\end{aligned}$$

We can plug the above expression of ψ_m^* back to the expression of $L_m^*(\eta, \psi_m^*)$ to get:

$$L_m^*(\eta,\psi_m^*)=\exp\left(\frac{\mathcal{L}(x,\eta)-\lambda\gamma_s\log\left\{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\right\}+\lambda\gamma_s-\lambda\gamma_s}{\lambda\gamma_s}\right)=\frac{\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)}{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]}$$

We plug the above expression of $L_m^*(\eta, \psi_m^*)$ into the objective function of (A61), which becomes:

$$\begin{aligned}
&\mathbb{E}_{\mathbb{Q}_m}\left[\mathcal{L}(x,\eta)\frac{\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)}{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]}-\lambda\gamma_s\log\left(\frac{\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)}{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]}\right)\frac{\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)}{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]}\right]\\
&=\mathbb{E}_{\mathbb{Q}_m}\left[\lambda\gamma_s\log\left(\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\right)\frac{\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)}{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]}\right]\\
&=\frac{\lambda\gamma_s\log\left(\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\right)}{\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]}\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\\
&=\lambda\gamma_s\log\left(\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\right)
\end{aligned}$$

Since problem (A61) is the sub-problem of problem (A60), we can plug $\lambda\gamma_s\log\left(\mathbb{E}_{\mathbb{Q}_m}\left[\exp\left(\frac{\mathcal{L}(x,\eta)}{\lambda\gamma_s}\right)\right]\right)$

back to problem (A60). Hence, the problem (A60) becomes a more tractable reformulation:

$$\min_{\lambda \geq 0} \quad \lambda \bar{\varepsilon}_s + \frac{1}{M} \sum_{m=1}^{M} \lambda \gamma_s \log \left(\mathbb{E}_{\mathbb{Q}_m} \left[\exp \left(\frac{\mathcal{L}(x, \eta)}{\lambda \gamma_s} \right) \right] \right) \tag{A63}$$

We introduce epi-graphical variables s_m to reformulate problem (A63) as:

$$\min_{\lambda \geq 0, s} \quad \lambda \bar{\varepsilon}_s + \frac{1}{M} \sum_{m=1}^{M} s_m \tag{A64a}$$

$$\text{s.t.} \quad \lambda \gamma_s \log \left(\mathbb{E}_{\mathbb{Q}_m} \left[\exp \left(\frac{\mathcal{L}(x, \eta)}{\lambda \gamma_s} \right) \right] \right) \leq s_m, \forall m = 1, ..., M \tag{A64b}$$

The constraint (A64b) can also be rewritten as:

$$\begin{aligned} &\mathbb{E}_{\mathbb{Q}_m} \left[\exp \left(\frac{\mathcal{L}(x, \eta)}{\lambda \gamma_s} \right) \right] \leq \exp \left(\frac{s_m}{\lambda \gamma_s} \right) \\ &\Longrightarrow 1 \geq \mathbb{E}_{\mathbb{Q}_m} \left[\exp \left(\frac{\mathcal{L}(x, \eta) - s_m}{\lambda \gamma_s} \right) \right] \\ &\Longrightarrow \lambda \gamma_s \geq \mathbb{E}_{\mathbb{Q}_m} \left[\lambda \gamma_s \exp \left(\frac{\mathcal{L}(x, \eta) - s_m}{\lambda \gamma_s} \right) \right] \end{aligned}$$

Till this point, the strong dual problem of (5.18) (in the main text) is derived without assuming that the support Ξ is discrete. In the next subsection, the dual problem is discretized to get a more tractable formulation.

A4.1.1 Discrete approximation of the dual problem

To enhance the computational tractability and efficiency, we approximate $\mathbb{Q}_m$ by an empirical distribution $\hat{\mathbb{Q}}_m$ established based on samples $\{\eta_{n,m}\}_{n=1}^{N}$. Those samples are called expansion points and they can be sampled from the kernel probability distribution $d\mathbb{Q}_m(\eta) := \frac{\exp(-c(\eta, \xi_m)/\gamma_s)}{\int \exp(-c(\tau, \xi_m)/\gamma_s) dA(\tau)} dA(\eta)$. This step is called the discretized kernel distribution relaxation since the original support is relaxed to be a discrete support based on the finite discrete points sampled from a kernel distribution. Note that this relaxation is not a conservative approximation since it only provides an empirical estimation of the original support and does not yield a bound for the original problem. According to (A49) and [17], $\mathbb{Q}_m$ becomes a Gaussian distribution $\mathcal{N}(\xi_m, \gamma_s I_d)$ if the cost function $c(\cdot, \cdot)$ is selected to be $\frac{1}{2}\| \cdot - \cdot \|_2^2$ and

$\Xi = \mathbb{R}^d$. I_d is a $d \times d$ identity matrix. $\{\eta_{n,m}\}_{n=1}^{N}$ can be sampled from $\mathcal{N}(\xi_m, \gamma_s I_d)$.

With the above discretization based approximation, the inequality becomes:

$$\lambda\gamma_s \geq \frac{1}{N}\sum_{n=1}^{N} \lambda\gamma_s \exp\left(\frac{\mathcal{L}(x, \eta_{n,m}) - s_m}{\lambda\gamma_s}\right), \forall m = 1, ..., M$$

or, equivalently

$$\begin{aligned}
&\lambda\gamma_s \geq \frac{1}{N}\sum_{n=1}^{N} a_{n,m}, \forall m = 1, ..., M \\
&a_{n,m} \geq \lambda\gamma_s \exp\left(\frac{\mathcal{L}(x, \eta_{n,m}) - s_m}{\lambda\gamma_s}\right), \forall n = 1, ..., N, \forall m = 1, ..., M
\end{aligned}$$

The last constraint is essentially $(\lambda\gamma_s, a_{n,m}, \mathcal{L}(x, \eta_{n,m}) - s_m) \in \mathcal{K}_{\text{exp}}$, where $\mathcal{K}_{\text{exp}}$ denotes the exponential cone [166]. Finally, the optimization model including (A64a)-(A64b) can be rewritten as:

$$\min_{\lambda, s, a} \quad \lambda\bar{\varepsilon}_s + \frac{1}{M}\sum_{m=1}^{M} s_m \tag{A65a}$$

$$\text{s.t.} \quad \lambda\gamma_s \geq \frac{1}{N}\sum_{n=1}^{N} a_{n,m}, \forall m = 1, ..., M \tag{A65b}$$

$$(\lambda\gamma_s, a_{n,m}, \mathcal{L}(x, \eta_{n,m}) - s_m) \in \mathcal{K}_{\text{exp}}, \forall n = 1, ..., N, \forall m = 1, ..., M \tag{A65c}$$

$$\lambda \geq 0, s \in \mathbb{R}^M, a \in \mathbb{R}_+^{N \times M} \tag{A65d}$$

A4.2 Employed Wasserstein DRCCP model and SAA model

The Wasserstein DRCCP model employed in this work is based on the type-1 Wasserstein distance with ℓ_2 norm, and it is given as:

$$\begin{aligned}
\min_{x,\beta,\kappa,z} \quad & f(x) \\
\text{s.t.} \quad & \beta + \frac{1}{\delta}\left[\kappa\varepsilon_W + \frac{1}{M}\sum_{m=1}^{M} z_m\right] \le 0 \\
& z_m \ge h_i^0(x) - \beta + h_i(x)^T \xi_m, \quad \forall i = 1, ..., w, \forall m = 1, ..., M \\
& z_m \ge 0, \quad \forall m = 1, ..., M \\
& \|h_i(x)\|_* \le \kappa, \quad \forall i = 1, ..., w
\end{aligned} \tag{A66}$$

where ε_W is the radius size of the Wasserstein ambiguity set. Since this Wasserstein DRCCP model also employs the CVaR approximation, β in the above model also comes from the CVaR approximation. δ is the user-defined tolerance for the JCSP. While using the above Wasserstein DRCCP model, a constraint function $g_i(x, \xi_m)$ in the original problem based on a collected sample ξ_m must be able to be expressed as the form affine in uncertainty: $g_i(x, \xi_m) = h_i(x)^T \xi_m + h_i^0(x)$. $\| \cdot \|_*$ is a dual norm. Since we use ℓ_2 norm in this research, the dual norm in the above model is still a ℓ_2 norm (the dual norm of a ℓ_2 norm is still a ℓ_2 norm).

The SAA model employed in this work is given as:

$$\begin{aligned}
\min_{x} \quad & f(x) \\
\text{s.t.} \quad & \frac{1}{M}\sum_{m=1}^{M} \mathbb{I}\left(\bigcup_{i=1}^{w} g_i(x, \xi_m) > 0\right) \le \delta
\end{aligned} \tag{A67}$$

where $\mathbb{I}$ is the indicator function defined as:

$$\mathbb{I}\left(\bigcup_{i=1}^{w} g_i(x, \xi_m) > 0\right) = \begin{cases} 0, & \text{for} \quad g_i(x, \xi_m) \le 0, \forall i = 1, ..., w \\ 1, & \text{for} \quad g_i(x, \xi_m) > 0, \exists i = 1, ..., w \end{cases} \tag{A68}$$

The above formulation based on a sample ξ_m can be reformulated as the following mixed-integer model:

$$
\begin{aligned}
&\frac{1}{M}\sum_{m=1}^{M}\bar{y}_m \le \delta \\
&\mathcal{M}(\widetilde{y}_{im}-1)+\epsilon \le g_i(x,\xi_m) \le \mathcal{M}\widetilde{y}_{im}, \quad \forall i, m \\
&\bar{y}_m - 1 \le (\sum_{i=1}^{w}\widetilde{y}_{im}) - 1 \le w\bar{y}_m - 1, \quad \forall m \\
&\bar{y}_m \in \{0,1\}, \widetilde{y}_{im} \in \{0,1\}
\end{aligned}
\tag{A69}
$$

where $\bar{y}_m$ and $\widetilde{y}_{im}$ are binary variables. $\mathcal{M}$ is a big number that $\mathcal{M} \ge g_i(x,\xi_m), \forall i = 1, ..., w, \forall m = 1, ..., M, \forall x \in \mathcal{X}$. ϵ is a small positive number which is set as 10^{-8} in this work.