2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Learnable Adaptive and Robust Controller for a Two Particle Carbonate Precipitation Process

Mikhail Kakanov[1], Sandesh Athni Hiremath[1], Andreas Voigt[2], Kai Sundmacher[2] and Naim Bajcinca[1]

Abstract— **In this work, we introduce a novel learning-based controller tailored for autonomous control of a batch-type precipitation process involving calcium and magnesium carbonates. The process takes in fluid containing valuable materials such as Ca^{+2} and Mg^{+2} ions, along with impurities and seed particles, to facilitate the sequential precipitation of these ions into their respective carbonates. The controller's goal is to attain a specified size of the precipitated particles under different process uncertainties. Here the residence time, i.e. the time allowed for the ions to remain in fluid phase, is used as the manipulation variable. The controller is designed as a solution to a stochastic optimal control problem and implemented using machine learning techniques. For the prediction model, we use convolutional neural networks (CNN) and for the control synthesis, we use a type of recurrent neural networks (RNNs). The designed control is learnable, adaptable to varying process dynamics and robust to random disturbances in the process, thus resulting in a learnable adaptive, and robust controller (LARC). The effectiveness of LARC is validated through different simulation-based tests.**

Index Terms— **stochastic control, learning-based control, process control, particulate processes.**

I. Introduction

Addressing the global challenge of climate change necessitates innovative solutions for reducing greenhouse gas emissions and permanently sequestering carbon dioxide. Carbon mineralization, a concept initially proposed in 1990 [1], offers a promising avenue for storing CO_2 as environmentally benign and stable carbonates. This approach presents an opportunity to establish a permanent and leakage-free method for CO_2 disposal. Among the myriad of alkaline earth metals found in nature, calcium and magnesium stand out as the most abundant, making them ideal candidates for carbonate formation [2]. Natural minerals, such as Antigorite, Lizardite, Forsterite, Augite, and Wollastonite, along with industrial waste streams like waste cement, coal fly ash, steelmaking slag, platinum group mineral mine tailings, and red gypsum, can be employed as feedstocks for mineral carbonation [3], [4], [5]. This research project explores the possibilities and challenges of ex-situ mineral carbonation, a process that holds the potential to transform CO_2 into a stable and sustainable resource.

This work was supported by DFG priority program (SPP 2364) - Autonomous processes in particle technology - 504852622

[1]The authors are with the Department of Mechanical and Process Engineering, RPTU Kaiserslautern, Gottlieb-Daimler-Straße 42, 67663 Kaiserslautern, Germany. (Email:{m.kakanov, sandesh.hiremath, naim.bajcinca}@mv.rptu.de).

[2]The authors are with the Faculty of Process and Systems Engineering Institute of Process Engineering Universitätsplatz 2, 39106, Magdeburg, Germany. (Email:{andreas.voigt, kai.sundmacher}@ovgu.de).

In the realm of carbon mineralization and the control of carbonate precipitation, our research delves into the intricacies of distributed chemical processes. These encompass diffusion-convection-reaction processes, where the control of spatially distributed profiles is paramount. More often than not chemical processes are characterized by the interplay of macroscopic and microscopic phenomena, which directly influence the control of material properties. Specifically, within the context of carbonate precipitation process, our primary concern revolves around the precise control of particle size distribution (PSD), a pivotal factor in optimizing the desired outcomes of carbonate formation and, ultimately, sustainable carbon capture and storage solutions.

In the domain of controlling distributed chemical processes, process models are typically represented as hyperbolic/parabolic partial differential equations (PDEs), coupled PDEs and molecular dynamics/Monte-Carlo models, and integro-differential equations (population balances). The field of control for such systems has witnessed significant advancements. Presently, methodologies for control encompass a range of techniques. Nonlinear parabolic PDEs, as documented in [6], are harnessed for deriving low-order ordinary differential equation (ODE) models via Galerkin's method and the approximation of inertial manifolds. This allows for the synthesis of nonlinear and robust controllers [7], as well as the optimization of actuator/sensor placement through dynamic optimization techniques, as exemplified by [8]. Moreover, control strategies have extended to encompass other categories of distributed systems, including particulate processes and fluid dynamic systems [9]. In parallel, the control of distributed parameter systems has yielded diverse approaches, with developments extending to hyperbolic PDEs, passivity-based control methods [10], backstepping boundary control strategies [11], and predictive control techniques [12], [13].

The above mentioned control methodology, though not exhaustive, can be broadly classified as classical/conventional control methods and some of the main challenges of such methods are as follows. Firstly, the issue of robustness is the most prominent one. Since particulate processes are inherently effected by noisy perturbations, classical methods struggle to cope thus rendering them less effective in adapting to real-time variations. Secondly, the problem of autonomy arises due to the reliance on ex-situ measurements and actuation, leading to a lack of real-time feedback mechanisms. This deficiency necessitates frequent system re-identification, thus requiring human intervention, thus impeding the desired level of production throughput. Thirdly, the limitation of classical control methods lies in their static or dynamic optimization formulations, which frequently hinder their ability to learn from data and performs repeated calculations for varying input conditions. This stands in contrast to statistical optimization formulations, which offer a broader horizon for learning from empirical observations.

The success of machine learning techniques, particularly deep neural networks (DNN), has significantly impacted natural and engineering sciences, including control of particulate processes. DNN-based reinforcement learning (RL) has also gained interest in control domains. The Deep-Q-Network (DQN) technique is an example of this approach [14], which has been successfully applied in various industrial contexts, including optimizing multi-stage precipitation processes for zinc product purity [15]. However, DNNs and RL methods have limitations, such as concerns about mathematical explainability and risk quantification. This has led to the exploration of model-based RL approaches, which require specific dynamical models and numerical schemes for simulations tailored to each process [16], [17], which further leads enables DNNs to serve as numerical solvers for simulating complex systems [18]. DNN-based ML techniques offer a promising opportunity to address particulate process control challenges and generalize control strategies across processes, enabling organizations to achieve high levels of autonomy.

In this work, we present a novel control method for the control of a two-particle precipitation process that is robust to process noise and model imperfections and adaptive to the variability of the process dynamics. To this end, we present a suitable stochastic partial differential equation (SPDE) to model the evolution of particle size distribution (PSD) and introduce a corresponding stochastic optimal control problem (SOCP). To solve the SOCP we employ ML techniques where we use DNNs for control synthesis as well as predicting the process dynamics. More specifically, for the former we employ a Gated Recurrent Unit (GRU) based RNN architecture, and for the latter CNN based UNet architecture.

The structure of this article is as follows: Section III provides a formal problem formulation including process modeling in Section III-A and predictive control formulation in Section III-B. In Section IV, we delve into the design concepts of the learning-based control system followed by details about dataset generation and network architecture in Section IV-A and Section IV-B respectively. Moving forward, Section V offers an insightful discussion of the outcomes derived from the numerical simulations, and lastly, in Section VI, we summarize the findings, emphasize their significance, and provide concluding remarks.

II. Notation and Preliminaries

The space of real, natural, and integer numbers are denoted as $\mathbb{R}$, $\mathbb{N}$ and $\mathbb{Z}$ respectively. The space of positive real numbers is denoted as $\mathbb{R}^+$. The set of real $k \times m$ matrices is denoted by $\mathbb{R}^{k\times m}$. The transpose $A \in \mathbb{R}^{k\times m}$ are denoted by $A^\top$. $\mathrm{diag(f)} \in \mathrm{R^{d\times d}}$ denotes the diagonal matrix formed from vector $f \in \mathbb{R}^d$. For a function $f : [0, T] \times [0, S] \to \mathbb{R}$, $(t, x) \mapsto f(t, x)$, $f_t(\cdot)$ is a short hand notation for $f(t, \cdot)$. A vector function $F : [0, T] \times [0, S] \to \mathbb{R}^d$ is also denoted as $F = [F^1, \cdots, F^d]^\top$ where $F^i : [0, T] \times [0, S] \in \mathbb{R}$ for $i \in \{1, \cdots, d\}$. The zero matrix of appropriate dimensions is denoted by 0. The space of Lebesgue square-integrable functions $f : [0, T] \to [a, b]$ with $[a, b] \in \mathbb{R}$ and $T > 0$ is denoted as $L^2([0, T]; [a, b])$. Depending on where $f \in L^2([0, T]; \mathbb{R}^d)$ of $f \in \mathbb{R}^d$, the norm $\|f\|_2$ either denotes the L^2 norm or the Eucledian norm. For a number $x \in \mathbb{R}$, $|x|$ denotes the absolute value of x. For $\mathcal{X}$ a metric space and $X \in \mathcal{X}$ a random variable (rv) then $\mathbb{E}^{\mathcal{X}}[Y]$ denotes expectation of Y with respect to a distribution defined on the space $\mathcal{X}$ via X. $X \sim \mathrm{Unif}([a, b])$ denotes a rv X that is uniformly distributed over the interval $[a, b] \in \mathbb{R}$. Analogously $X \in \mathbb{R}^d$ and $X \sim \mathrm{Unif}([a, b]^d)$ denotes uniformly distributed rv X over $d > 0$ dimensional hypercube.

III. Problem formulation

This work is focused on the control of a particulate process that is operated in batch mode. The process of interest is the precipitation of two carbonate substances namely $\mathrm{CaCO_3}$ and $\mathrm{MgCO_3}$ from an aqueous solution containing the substances as cations i.e. in their respective ion form namely $\mathrm{Ca^{2+}}$ and $\mathrm{Mg^{2+}}$ respectively. The ion-based solution is placed in a reaction tank or vessel which is externally mixed with seed particles and carbon dioxide gas $\mathrm{CO_2}$. Further, the solution is externally influenced by factors such as pH, temperature, and pressure which ultimately impacts the reaction kinetics of the metal ions leading to formation of $\mathrm{CaCO_3}$ and $\mathrm{MgCO_3}$. The kinetics of the process can be described by the following reactions:

$$\begin{aligned}
&\mathrm{CO_2\,(g)} \to \mathrm{CO_2\,(aq)}\\
&\mathrm{CO_2\,(aq)} + \mathrm{H_2O(i)} \to \mathrm{HCO_3{}^-(aq)} + \mathrm{H^+(aq)}\\
&\mathrm{HCO_3{}^-(aq)} \to \mathrm{CO_3{}^{2-}(aq)} + \mathrm{H^+(aq)}\\
&\mathrm{OH^-(aq)} + \mathrm{H^+(aq)} \to \mathrm{H_2O(i)}\\
&\mathrm{Mg^{2+}(aq)} + \mathrm{CO_3{}^{2-}(aq)} \to \mathrm{MgCO_3(s)}\\
&\mathrm{Ca^{2+}(aq)} + \mathrm{CO_3{}^{2-}(aq)} \to \mathrm{CaCO_3(s)}
\end{aligned} \tag{1}$$

This sequence of reactions characterizes the chemical process of carbonate substances which then precipitates out from the solution as a solid material. Altogether, this is referred to as the carbonate precipitation batch process (CPBP). Notably, it is essential to remark that calcium carbonate $\mathrm{CaCO_3}$ and magnesium carbonate $\mathrm{MgCO_3}$ have different precipitation properties with the former in general having a higher precipitation rate compared to the latter [19]. Thus under suitable operational conditions, i.e. process dynamics condition, one could aim to selectively precipitate one substance before the other, preferably $\mathrm{CaCO_3}$ first and then $\mathrm{MgCO_3}$ later.

A. Process modelling

For the control of selective precipitation of $\mathrm{MgCO_3}$ and $\mathrm{CaCO_3}$ we model the underlying precipitation process by a coupled system of ordinary and partial differential equations. Here for the sake of simplicity as well as due to the restrictions in experimental setting, we make several simplifications. Firstly, we consider that both Mg^{2+} and Ca^{2+} ions do not interact with each other in any way, thus allowing us to decouple the dynamical equations for the two. Secondly, we neglect the process of nucleation as a result of which the birth term can be neglected. Thirdly, the effects of thermodynamic factors such as temperature and pressure are also ignored. Based on these simplifications, the precipitation of CaCO_3 and MgCO_3 are treated independently of each other. Following these considerations, we employ population balance equation (PBE) to model the evolution of particle size distribution (PSD) of the respective carbonates. The PBEs are in turn coupled with the evolution of the respective ion concentrations which are in turn modeled using ODEs. Let $f^1 : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$ and $f^2 : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$ represent the PSD of CaCO_3 and MgCO_3 respectively, with $(t, x) \mapsto f^j(t, x)$ denoting the density of particles having the size $x > 0$ at time $t \geq 0$, for $j \in \{1, 2\}$. Similarly, let $c^1 : \mathbb{R}^+ \to \mathbb{R}^+$ and $c^2 : \mathbb{R}^+ \to \mathbb{R}^+$ represents the concentration of Ca^{2+} and Mg^{2+} ions in the solution. Based on this notation and letting $j \in \{1, 2\}$, the simplified model for precipitation takes the following form:

$$\begin{aligned}
\frac{\partial f^j}{\partial t}(t,x) &= \frac{f_0^j(x) - f^j(t,x)}{\tau^j(t)} - G^j(t)\frac{\partial f^j}{\partial x}(t,x), \quad t, x > 0 \\
\frac{dc^j}{dt}(t) &= \frac{c_0^j - c^j(t)}{\tau^j(t)} - \rho k_v^j G^j(t) M^j(t), \quad t > 0 \\
G^j(t) &= k_g^j \left(\frac{c^j(t)}{c_{\text{sat}}^j} - 1\right)^{1.5}, \\
B^j(t) &= \rho^j k_b^j \left(\frac{c^j(t)}{c_{\text{sat}}^j} - 1\right)^{2.5} M^j(t), \\
M^j(t) &= \int_0^\infty x^2 f^j(t,x)dx, \\
f^j(0,x) &= f_0^j(x), \quad c^j(0) = c_0^j \quad f^j(t,0) = \frac{B^j(t)}{G^j(t)}.
\end{aligned} \tag{2}$$

As already mentioned above, for $j \in 1, 2$, $f^j(t,x) \propto \frac{1}{m^4}$ denotes the PSD of j-th substance, $x \propto \mu m$ signifies the size of particles, $c^j(t) \propto \frac{mol}{l}$ represents the concentration of j-th species of ion, $G^j(t) \propto \frac{60m}{min}$ characterizes the growth rate of the j-th substance, and $M^j(t) \propto \frac{1}{m}$ is the second moment of $f^j(t,x)$, indicative of the surface area of the growing carbonate particle. The term $\tau^j(t) \propto min$ stands for the residence time of the j-th species, indicating the duration for which the j-th species of ions remain in the system. Since τ^1 and τ^2 are taken to be independent, they provide the option to selectively precipitate one substance before the other under suitable process operational conditions. Additionally, for each j-th ion species, $\rho^j \propto \frac{mol}{m^3}$ represents density constant, $c_{\text{sat}}^j \propto \frac{mol}{l}$ corresponds to the saturation constant, $k_g^j \propto \frac{60\mu m}{min}$ signifies the growth rate coefficient and $k_v^j = 1$ is the particle shape coefficient.

The introduction of stochastic elements into the model is essential for accounting for the inherent imperfections in the model, the impurity of solutions, and measurement errors. To account for these factors, we represent the system (2) by incorporating a standard Brownian motion as a stochastic noise term for the ion concentration equation, thus allowing us to consider the effects of thermodynamic fluctuations of kinetic reactions as well as for the impurities in the ion concentrations. Altogether, due to the coupling with a PBE, we obtain, in the general sense, Stochastic Partial Differential Equation (SPDE). For $t \in (0, T]$, the SPDE version of (2) is given as

$$\begin{aligned}
dC_t &= [U_t(C_0 - C_t) - \boldsymbol{\rho} K_v \boldsymbol{G}(t)\boldsymbol{M}(t)]\,dt + \boldsymbol{\sigma}(t)C_t dW_t, \\
\partial_t F_t &= U_t(F_0 - F_t) - \boldsymbol{G}(t)\partial_x F_t, \\
F_0 &= F(0), \quad F(t,0) = \left[\frac{B^1}{G^1}, \frac{B^2}{G^2}\right]^\top, \quad C(0) = [c_0^1, c_0^2]^\top,
\end{aligned} \tag{3}$$

where, $F_t = [f^1(t,\cdot), f^2(t,\cdot)]^\top$, $C_t = [c^1(t), c^2(t)]^\top$,

$$\begin{aligned}
U_t &= \operatorname{diag}([\mathrm{u_t^1}, \mathrm{u_t^2}]^\top),\ \mathrm{u_t^j} = \frac{1}{\tau^{\mathrm{j}}(\mathrm{t})}, \\
K_v &= \operatorname{diag}([k_v^1, k_v^2]^\top), K_b = \operatorname{diag}([k_b^1, k_b^2]^\top), \\
K_g &= \operatorname{diag}([k_g^1, k_g^2]^\top),\ C_{\text{sat}} = \operatorname{diag}([c_{\text{sat}}^1, c_{\text{sat}}^2]^\top), \\
\boldsymbol{\rho} &= \operatorname{diag}([\rho^1, \rho^2]^\top),\ \boldsymbol{G}(t) = \operatorname{diag}([G^1(t), G^2(t)]^\top), \\
\boldsymbol{\sigma}(t) &= \operatorname{diag}([\sigma^1, \sigma^2]^\top), \boldsymbol{M}(t) = \operatorname{diag}([M^1(t), M^2(t)]^\top).
\end{aligned}$$

Here $(W_t)_{t\geq 0} = ([W_t^1, W_t^2]^\top)_{t\geq 0}$ is standard Brownian motion (standard Wiener process) on $\mathbb{R}^2$ and defined on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$, and $\mathcal{F}_t \subset \mathcal{F}$ denotes the filtration generated by W_t. Furthermore, the components W_t^1 and W_t^2 of W_t are independent of each other for all $t \geq 0$. For the sake of notational simplicity and convenience, we shall write (3) compactly as an abstract Cauchy problem

$$\begin{aligned}
dX_t &= [A(X_t, U_t; \boldsymbol{\theta})X_t + H(X_t, U_t; \boldsymbol{\theta})]dt + \Sigma_t dW_t \\
X(0) &= X_0 \\
\boldsymbol{A}(X_t, U_t; \boldsymbol{\theta}) &:= \begin{bmatrix} -\boldsymbol{G}(t)\partial_x - \operatorname{diag}(\mathrm{U_t}) & 0 \\ 0 & -\operatorname{diag}(\mathrm{U_t}) \end{bmatrix}, \\
X_t &:= [F_t^\top, C_t^\top]^\top, \quad \Sigma_t := \begin{bmatrix} 0 & 0 \\ 0 & \boldsymbol{\sigma}(t) \end{bmatrix}, \\
\boldsymbol{H}_t &:= \begin{bmatrix} \operatorname{diag}(\mathrm{U_t F_0}) \\ \operatorname{diag}(\mathrm{U_t C_0}) - \boldsymbol{\rho}\mathrm{K_v}\boldsymbol{G}(\mathrm{t})\boldsymbol{M}(\mathrm{t}) \end{bmatrix}, \\
\boldsymbol{\theta} &:= [k_v^1, k_v^2, k_g^1, k_g^2, k_b^1, k_b^2, \rho^1, \rho^2, c_{\text{sat}}^1, c_{\text{sat}}^2]^\top.
\end{aligned} \tag{4}$$

Here $\boldsymbol{A}$ denotes an unbounded nonlinear operator whose domain adheres to the boundary conditions, $\boldsymbol{H}_t$ denotes the nonlinear reaction terms, $\boldsymbol{\theta}$ denotes the vector of process parameters, and U_t denotes the control.

B. *Predictive control formulation*

Our objective is to design an effective control strategy for the task of selective precipitation of carbonate precipitation batch process. For this, we formulate a stochastic optimal control problem (SOCP) based on an underlying objective/cost function. The latter takes the following generic form, capable of incorporating the corresponding performance objectives of the process.

$$J(\mathrm{x}, U; \boldsymbol{\theta}, Y) = \mathbb{E}_{\mathrm{x}}\left[\int_0^T L(X_s, U_s)ds + g(X_T, Y_T)\right],$$

where $\mathbb{E}_{\mathrm{x}}[\cdot] := \mathbb{E}[\cdot|X_0 = \mathrm{x}]$. Here, the cost function comprises two components: a running cost $L(X_t, U_t, Y_t)$, and a terminal cost $g(X_T, Y_T)$, with Y representing the reference or the target state. In the current implementation, these terms have the following specific form:

$$L(x, u; y) := \alpha_u \|u\|_2^2,\ \alpha_u := 0.01,$$
$$g(x, y) := \alpha_T \|x - y\|_2^2,\ \alpha_T := 1.$$

The weights $\alpha_u \propto$ min and $\alpha_T \propto m^4$. Thus, the resulting total cost is a non-dimensional number. Based on (5) and letting $\mathcal{U} := L_2([0,T]; (u_{\min}, u_{\max})^2)$ with $u_{\min}, u_{\max} \in (0, \infty)$, the corresponding SOCP for computing an optimal control policy is given as

$$U^* = \arg\min_{U \in \mathcal{U}} J(\mathrm{x}, U; \boldsymbol{\theta}, Y),\ \text{such that (4) holds.}$$

Thus given, $(\mathrm{x}, \boldsymbol{\theta}, Y)$, the solution U^* to (5) can be viewed as a mapping $(\mathrm{x}, \boldsymbol{\theta}, Y) \mapsto U^*$. This mapping in turn defines a (optimal) control law that is represented as

$$U^* = \pi^*(\mathrm{x}, \boldsymbol{\theta}, Y) \in \mathcal{U} \tag{5}$$

The standard techniques for solving (5) involve successive state and adjoint equation resolutions until convergence. However, due to the computational inefficiency of these methods, especially when dealing with the stochastic PDE in (3), alternatives like the Hamilton-Jacobi-Bellman equation become resource-intensive [20]. To address this, a statistical approach is preferred, offering twofold benefits: it provides optimal control mapping, reduces the need for repetitive and costly SOCP solving, and allows for broader automation potential by handling diverse inputs. Consequently, in the following section, a statistical optimization formulation and its DNN-based implementation will be presented.

IV. DNN implementation

Denoting $\mathcal{X}$ as the input space for (5), with $\mathcal{X} \ni \xi = (\mathrm{x}, \boldsymbol{\theta}, Y)$, our focus is to create a map $\xi \mapsto \Lambda(\xi)$ yielding the SOCP solution $\Lambda(\xi) = \pi^*(\xi)$. We represent $\Lambda(\xi)$ using a parameterized estimator $\hat{\lambda} \mapsto \Lambda(\xi; \hat{\lambda})$, $\lambda \in \mathbb{R}^l$, $l \gg 0$, ensuring $\Lambda(\xi; \hat{\lambda}) = \pi^(\xi)$ for nearly every $\xi \in \mathcal{X}$. A weaker criterion is to attain $\mathbb{E}^{\mathcal{X}}[\Lambda(\xi; \hat{\lambda})] = \mathbb{E}^{\mathcal{X}}[\pi^(\xi)]$, demanding $\Lambda(\cdot; \hat{\lambda})$ to be an unbiased estimator of $\pi^(\cdot)$. This leads to the statistical optimization problem (SP)

$$\hat{\lambda} = \arg\min_{\lambda \in \mathbb{R}^l} \mathbb{E}^{\mathcal{X}}[J(\mathrm{x}, \Lambda(\xi; \lambda); \boldsymbol{\theta}, Y)].$$

Analogously, given a set of $\mathbb{X} = \{\xi^k\}_{k=\{1,;m\}}$ of m input samples from $\mathcal{X}$, the discrete SP takes the following form

$$\hat{\lambda}^m = \arg\min_{\lambda \in \mathbb{R}^l} \frac{1}{m} \sum_{k=1}^{m} J(\mathrm{x}^k, \Lambda(\xi^k; \lambda); \boldsymbol{\theta}^k, Y^k).$$

$J(\mathrm{x}, \Lambda(\xi; \lambda); \boldsymbol{\theta}, Y)$, after temporal discretization, is given as

$$J(\mathrm{x}, \Lambda(\xi; \lambda); \boldsymbol{\theta}, Y) \approx \mathbb{E}_{\mathrm{x}} \left[\sum_{i=1}^{N} \|\hat{U}_{t_i}\|_2^2 + \|X_{t_N}(\hat{U}) - Y_{t_N}\|_2^2 \right]$$
$$\approx \frac{1}{n} \sum_{q=1}^{n} \sum_{i=1}^{N} \|\hat{U}_{t_i}^q\|_2^2 + \|X_{t_N}^q(\Lambda) - Y_{t_N}^q\|_2^2.$$

Since the input samples ξ are independent of the process noise $(W_t)_{t \geq 0}$ we can consider the k-th input sample in turn contains n process noise variations so as to obtain $M = m \times n$ samples altogether. Based on this we can combine the input-averaging and the noise averaging to obtain the following SP

$$\hat{\lambda}^M = \arg\min_{\lambda \in \mathbb{R}^l} \frac{1}{M} \sum_{k=1}^{M} \sum_{i=1}^{N} \|\hat{U}_{t_i}^k\|_2^2 + \|X_{t_N}^k(\hat{U}^k) - Y_{t_N}^k\|_2^2,$$
$$= \arg\min_{\lambda \in \mathbb{R}^l} \mathbb{L}_{\Lambda}^M(\lambda; \mathbb{X}) \tag{6}$$

where $\Lambda(\xi^k; \hat{\lambda}) = \hat{U}_{t_0:t_{N-1}}^k = [U_{t_0}^k, \cdots, U_{t_{N-1}}^k]^\top$ and $X_{t_1:t_N}^k(\hat{U}_{t_0:t_{N-1}}^k)$ is the predicted state of the process under the control policy $\hat{U}_{t_0:t_{N-1}}^k$. Following this, the parameterized optimal control policy function $\Lambda(\cdot; \lambda)$ is implemented as a deep neural network (DNN) whose internal parameter λ is determined as a solution to the SP (6). This requires the of use stochastic approximation schemes [21] to computationally determine the suitable minimizer $\hat{\lambda}$ which is attributed to the task network training. Since determining $\hat{\lambda}$ requires predicting the states X_{t_i} along the prediction horizon $[0, T]$, which can be done using discrete numerical integration of (4). Alternatively, another network Φ can be used to predict the evolution of the process along the horizon $[0, T]$ for the given input $\zeta := (\mathrm{x}, \boldsymbol{\theta}, U)$ consisting of the initial value x, process parameters $\boldsymbol{\theta}$ and the control value U. Based on the similar idea for obtaining Λ, we shall implement a mapping $\zeta \mapsto \Phi(\zeta)$ such that $\Phi(\zeta) = X_{t_1:t_N} = [X_1, \cdots, X_{t_N}]^\top$ is the solution to the (4) for the given data ζ. By parameterizing the map $\Phi(\cdot; \varphi)$ in terms of $\varphi \in \mathbb{R}^p$, for $p \gg 0$, Φ is realized by statistically optimizing over the dataset $\mathbb{Y} = \{\zeta^k\}_{k=1}^M$ which is formally stated as follows:

$$\hat{\varphi}^M = \arg\min_{\varphi \in R} \frac{1}{M} \sum_{k=1}^{M} \sum_{i=1}^{N} \|\hat{X}_{t_i}^k - X_{t_i}^k\|_2^2 \tag{7}$$
$$= \arg\min_{\varphi \in \mathbb{R}^p} \mathbb{L}_{\Phi}^M(\varphi; \mathbb{Y})$$

where, $\Phi(\zeta^k; \varphi) = \hat{X}_{t_1:t_N}^k = [\hat{X}_{t_i}^k, \cdots, \hat{X}_{t_N}^k]^\top$ and $X_{t_1:t_N}^k = [X_1^k, \cdots, X_{t_N}^k]^\top = \mathbb{I}(\zeta^k)$ with $\mathbb{I}$ being the numerical integrator, such as Euler-Maruyama and Meilstein [22], of (4). Having obtained a sufficiently robust and generalized Φ, it can be used as the predictor model for training the Λ network. This provides the following advantages: (i) the controller network Λ has only the black-box view of the process behavior which makes the network more flexible to cope with process variability. (ii) It avoids the need for explicitly solving for the process dynamics thus speeds up the controller training process. (iii) Provides the option for tuning the process network Φ in order to adapt to the measured data. It further provides the flexibility for online and closed-loop tuning of Φ. Based on this the estimation problem of $\hat{\lambda}$ takes the following form

$$\hat{\lambda}^M = \arg\min_{\lambda \in \mathbb{R}^l} \frac{1}{M} \sum_{k=1}^{M} \sum_{i=1}^{N} \|\hat{U}_{t_i}^k\|_2^2 + \|\Phi(\cdots, \hat{U}_{t_N}^k) - Y_{t_N}^k\|_2^2$$
$$= \arg\min_{\lambda \in \mathbb{R}^l} \mathbb{L}_{\Lambda}^M(\lambda; \mathbb{X}) \tag{8}$$

Variable	min value	max value	Unit
$\tau^j, j \in \{1,2\}$	10	500	min
x	0	1	μm
$\rho^j, j \in \{1,2\}$	0.01	0.05	g/cm^3
$c_0^j, j \in \{1,2\}$	0.00125	0.05	mol/l
$k_g^j, j \in \{1,2\}$	1	15	$\mu m/s$
$\mu_0^j, j \in \{1,2\}$	0.2	0.8	μm
$c_{\text{sat}}^j, j \in \{1,2\}$	1e-05	1e-05	mol/l
$k_b^j, j \in \{1,2\}$	0	0	1
$k_v^j, j \in \{1,2\}$	1	1	1

TABLE I: Variable interval bounds.

In view of (7) and (8) the composition of the two networks Λ and Φ can be symbolically represented as $\Lambda(\Phi)$ which basically represents the learning based robust and adaptive controller which in short is called as the LARC network. Next, we shall discuss briefly the dataset generation and network architecture for the process and the controller network Φ and Λ respectively.

A. Dataset generation

For training the process network Φ and controller network Λ, two separate datasets are generated. For the Φ network, a synthetic dataset $\mathbb{Y} = \{\zeta^k\}_{k=1}^M$ is obtained by sampling uniformly from fixed intervals spanned by the minimum and maximum values provided in Table I. More specifically, each component of the k-th input sample ζ^k is obtained in the following way. The initial particle size distribution $F_0(x) = [\exp(-(x-\mu_0^1)^2), \exp(-(x-\mu_0^2)^2)]^\top$ with $\mu_0^j \sim \text{Unif}([\mu_{0,\min}^j, \mu_{0,\max}^j])$, initial ion concentration $C_0 \sim \text{Unif}([c_{0,\min}^1, c_{0,\max}^1] \times [c_{0,\min}^1, c_{0,\max}^1])$, process parameters $\boldsymbol{\theta} \sim \text{Unif}([\boldsymbol{\theta}_{\min}, \boldsymbol{\theta}_{\max}])$ and control signal $U \sim \text{Unif}([\frac{1}{\tau_{\min}^1}, \frac{1}{\tau_{\max}^1}]^2)$. Here we have introduced two notational simplifications. Firstly we have ignored the superscript k corresponding to the k-th input sample. Secondly, the variables $\boldsymbol{\theta}_{\min}$ and $\boldsymbol{\theta}_{\max}$ are used to denote the minimum and maximum values, respectively, of the involved components of the parameter vector $\boldsymbol{\theta}$. Similar convention also applies to the other variables mentioned above. Furthermore, the variables K_g, $\boldsymbol{\rho}$, and U are taken to be time-varying parameters thus they are further uniformly sampled along the time (prediction) horizon. The choice of time-varying process parameters thus enables us to construct an adaptive prediction model Φ.

For the Λ network, the dataset $\mathbb{X} = \{\xi^k\}_{k=1}^M$ is synthetically generated by sampling uniformly from the fixed interval spanned by the minimum and maximum values provided in Table I for the respective variables. Additionally, the terminal target PSD $\bar{F} := Y_{t_N}$ is obtained using the relation $\bar{F}(x) = [\exp(-(x-\mu_T^1)^2), \exp(-(x-\mu_T^2)^2)]^\top$ where $\mu_T^j \sim \mu_0^j + 0.35(1-\mu_0^j)\text{Unif}([0,1])$ for $j \in \{1,2\}$. The size of both the datasets is given by M which is equal $15K$. To ensure unbiased training testing and validation, the datasets are split into three disjoint sets namely training, testing, and validation sets as per the splitting ratio $3:1:1$. This basically provides $9K$ samples for training and $3K$ samples for testing and validation each.

B. DNN Architecture

Since the process dynamics are defined by an SPDE involving independent variables: time (t) and particle size (x), for the network Φ, focusing on predictions within finite time $t \in [0,T]$ and particle sizes $x \in [0,S]$, we consider the joint space $\mathfrak{D}_T = [0,T] \times [0,S]$ to predict PSD F_t on $\mathfrak{D}_T$ and C_t on $[0,T]$. Utilizing this perspective, we employ a UNet architecture due to its effective multiscale design ideal for 2D feature identification and localization [23]. UNet's adaptability to limited datasets aids faster training and fine-tuning, beneficial for active online learning in closed-loop operations. Accordingly, the implemented Φ network is comprised of 27 layers, with input/output dimensions of 50×57 and 50×50, totaling $17.3M$ parameters.

For the control synthesis model Λ, we make use of a network structure suitable for identifying sequential correlations. This motivates us to use RNN-like architecture and specifically we make use of Gated Recurrent Units (GRUs) based network. With fewer gating mechanisms than other RNNs such as LSTMs, GRUs enable more efficient training processes with reduced computational load [24]. Furthermore, the streamlined update and reset gates allow selective memory operations, aiding in better information retention and mitigating the vanishing/exploding gradient problem. These factors make GRUs effective in capturing essential temporal dependencies in sequential data. Accordingly, we designed the Λ network to be composed of 50 layers with input and output dimensions being 50×16 and 50×2 respectively. As a result, the network has in total $1.9K$ trainable parameters.

C. Training

For the training of the process and controller networks, we make use of the finite-sample loss functionals $\mathfrak{L}_\Phi^M$ and $\mathfrak{L}_\Lambda^M$ respectively, which are as given in (9). These functionals are motivated by the SP formulation provided in (7) and (8). Additionally, the loss terms are supplemented with the terms $\mathbb{L}_{\text{mode},\varphi}^M(\varphi; \mathbb{X})$ and $\mathbb{L}_{\text{mode},\lambda}^M(\lambda; \mathbb{X})$ respectively, in order to facilitate better alignment of the mode of the estimated PSD $\hat{F}_t$ with that of the target PSD $\bar{F}_t$ for all $t > 0$.

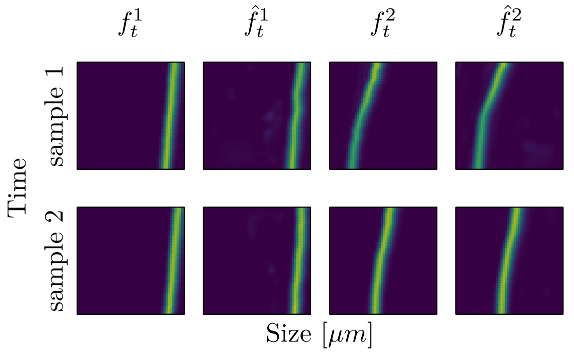
$$\mathfrak{L}_\Phi^M(\varphi) = \mathbb{L}_\Phi^M(\varphi; \mathbb{Y}) + \mathbb{L}_{\text{mode},\varphi}^M(\varphi; \mathbb{Y}) \tag{9}$$

$$\mathfrak{L}_\Lambda^M(\lambda) = \mathbb{L}_\Lambda^M(\lambda; \mathbb{X}) + \mathbb{L}_{\text{mode},\lambda}^M(\lambda; \mathbb{X}) \tag{10}$$

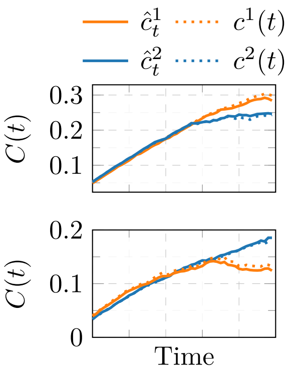
$$\mathbb{L}_{\text{mode},\varphi}^M(\varphi; \mathbb{Y}) := \frac{1}{M}\sum_{k=1}^M \sum_{i=1}^N \|\text{mode}(\hat{F}_{t_i}^k) - \text{mode}(\bar{F}_{t_i}^k)\|_2^2$$

$$\mathbb{L}_{\text{mode},\lambda}^M(\lambda; \mathbb{X}) := \frac{1}{M}\sum_{k=1}^M \sum_{i=1}^N \max(0, \text{mode}(\hat{F}_{t_i}^k) - \text{mode}(\bar{F}_{t_i}^k))$$

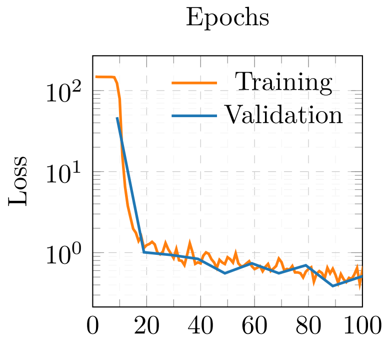
Based on this we make use of the Algorithms 1 and 2 to determine the most likely parameters $\hat{\varphi}$ and $\hat{\lambda}$ of the process and controller networks Φ and Λ, respectively that is able to best explain the observations made available through the datasets $\mathbb{Y}$ and $\mathbb{X}$ respectively. Since training $\Lambda(\Phi)$ is dependent on Φ, the latter is trained first and then used, while keeping learned value $\hat{\varphi}$ fixed, for the training of Λ. Based on Fig. 1(c) we see that after 100 epochs of training over a training set of 9000 samples, the process network Φ is able to fit the training dataset quite well evident in the loss reduction of order 10^2. Furthermore, since the reduction



(a) Particle size distribution



(b) Concentration



(c) Loss

Fig. 1: Sample outputs of the process network Φ and its training and validation loss.

of the validation loss, obtained over a validation set of 3000 samples, is also in the same order we infer that there is practically minimal over-fitting. The latter can also be confirmed by qualitatively inspecting the predictions of Φ shown in Fig. 1(a) and 1(b). In the former i.e. Fig. 1(a), the PSDs are depicted as grayscale images with y-axis denoting time in $mins$ and x-axis denoting particle size in μm. The plots on the top row correspond to $\mathrm{CaCO_3}$ while the bottom plots correspond to $\mathrm{MgCO_3}$. Based on this we can see that the predicted PSD evolution pattern $\hat{F}$ matches that of the reference PSD $\bar{F}$ obtained by numerical integration. In Fig. 1(b), the predicted ion-concentrations $\hat{C} = [\hat{c}^1, \hat{c}^2]^\top$ are depicted in solid lines (orange for Ca^{2+} and blue for Mg^{2+}) while the reference profile obtained from numerical integration is denoted in dashed line with similar substance specific color coding.

Algorithm 1: process neural network $\hat{X}_{t_1:t_N} := \Phi(X_{t_0}, \boldsymbol{\theta}_{t_0:t_{N-1}}, U_{t_0:t_{N-1}})$

```
Data: Dataset $\mathbb{Y} = \{\zeta^k\}_{k=1}^M$, consist of $M$ samples
      $\zeta^k = (\mathrm{x}^k, \boldsymbol{\theta}^k_{t_0:t_{N-1}}, U^k_{t_0:t_{N-1}})$, batch size
      $B = 250$, $[\sigma^1, \sigma^2] = [.1, .1]$,
      $\boldsymbol{\theta}^k_{t_i} = (K^k_{b,t_i}, K^k_{g,t_i}, K^k_{v,t_i}, \boldsymbol{\rho}^k_{t_i}, C^k_{sat,t_i})$.
input : Initial of neural network parameters $\varphi^0$ and
        Wiener increments $Z_{t_0:t_{N-1}}$, $\Sigma_{t_0:t_{N-1}}$
1  for $k = 1, 2, \ldots, L_{epoch}$ do
2      $\hat{X}^k_{t_1:t_N} = \Phi(\zeta^k; \varphi^{k-1})$
3      for $m = 1, 2, \ldots, B$ do
4          for $i = 0, 1, \ldots, N-1$ do
5              $\boldsymbol{G}_{t_i} = K_{g,t_i}\left(\frac{C_{t_i}}{C_{\text{sat}}} - 1\right)^{1.5}$
6              $\boldsymbol{M}_{t_i} = \frac{1}{n_x}\sum_0^{n_x} xx^T F_{t_i}$
7              $X_{t_{i+1}} = X_{t_i} + [A(X_{t_i}, U_{t_i}; \boldsymbol{\theta}_{t_i})X_{t_i} +$
               $H(X_t, U_t; \boldsymbol{\theta}_t)]dt + \sqrt{dt}\Sigma_{t_i} X_{t_i} Z_{t_i}$
8      if Training then
9          $\mathfrak{L}^B(\varphi^{k-1})$ as per (7);
10         $\varphi^k \leftarrow \arg\min_{\varphi \in \mathbb{R}^p} \mathbb{L}(\varphi)$
return: $\hat{X}_{t_1:t_N}$
```

Based on the plots we see that the predictions and reference profiles match very well. As a consequence, the growth coefficients (related via (2)) also matches (not shown plots) thereby ensuring qualitatively correct evolution profiles of the predicted PSD. Next, with the Φ model fixed, we trained the controller network again for 100 epochs to obtain good loss reduction over the training set (of 9000 samples) with almost zero over-fitting on the validation set (of 3000 samples) as shown in Fig. 2 (c). Qualitative results of the synthesized controls can be inferred by inspecting Fig. 2(a) and 2(b) which show the state of the simulated process after applying the synthesized control of Λ to two different input samples from the test set.

Algorithm 2: Controller neural network $\hat{U}_{t_0:t_{N-1}} := \Lambda(X_{t_0}, \boldsymbol{\theta}_{t_0:t_{N-1}}, Y_{t_1:t_N})$

```
Data: Dataset $\mathbb{X} = \{\xi^k\}_{k=1}^M$, , consist of $M$ samples
      $\xi^k = (\mathrm{x}^k, \boldsymbol{\theta}^k_{t_0:t_{N-1}}, Y^k_{t_1:t_N})$, batch size
      $B = 250$, $[\sigma^1, \sigma^2] = [.1, .1]$,
      $\boldsymbol{\theta}^k_{t_i} = (K^k_{b,t_i}, K^k_{g,t_i}, K^k_{v,t_i}, \boldsymbol{\rho}^k_{t_i}, C^k_{sat,t_i})$.
input : Initial of neural network parameters $\lambda^0$
1  for $k = 1, 2, \ldots, L_{epoch}$ do
2      $\hat{U}^k_{t_0:t_{N-1}} = \Lambda(\xi^k; \lambda^{k-1})$
3      $\hat{X}^k_{t_1:t_N} = \Phi(\mathrm{x}^k, \boldsymbol{\theta}^k_{t_0:t_{N-1}}, \hat{U}^k_{t_0:t_{N-1}}; \hat{\varphi})$
4      if Training then
5          $\mathfrak{L}^B(\lambda^{k-1})$ as per (7);
6          $\lambda^k \leftarrow \arg\min_{\lambda \in \mathbb{R}^l} \mathbb{L}(\lambda)$
return: $\hat{U}_{t_0:t_{N-1}}$
```

Based on this we see that the single-step control is able to drive the PSD towards the target and the growth rate has a decaying profile indicating that the process would slow down and eventually stop as it reaches the target PSD. Also, another interesting aspect to point out is the synthesized control signal $\hat{U}(t)$ i.e. $[\hat{u}_t^1, \hat{u}_t^2]^\top$ is such that its components $\hat{u}_t^j := \frac{1}{\tau^j(t)}$ for $j \in \{1, 2\}$ have non-intersection graphs. This indicates that the controller is able to select one of the substances to precipitate faster by keeping its residence time shorter while making the other particle precipitate slower by prescribing a larger residence time. Furthermore, the controller is correctly able to prioritize $\mathrm{CaCO_3}$ over $\mathrm{MgCO_3}$ thereby achieving faster overall throughput of the required carbonate sizes. The selective precipitation is also evident from the faster decay in the growth rate G^1 of $\mathrm{CaCO_3}$ in comparison to the growth rate G^2 of $\mathrm{MgCO_3}$.

V. Numerical Experiments

In this section, numerical experiments are conducted to study the adaptiveness and robustness of our LARC model $\Lambda(\Phi)$ for the task of controlling the CPBP. Firstly, we conducted robustness tests for the process network Φ where we fed the network with noisy parameter $\tilde{\boldsymbol{\theta}}_t$ obtained as a noisy perturbation of the nominal value $\boldsymbol{\theta}_t$. More specifically, $\tilde{\boldsymbol{\theta}}_t = \boldsymbol{\theta}_t(1+\delta z)$ for $\mathbb{R}^d \ni z \sim \mathcal{N}(0,1)$, with d being the dimension of $\boldsymbol{\theta}_t$. Letting $F_T = [f_T^1, f_T^2]^\top$ denote the reference PSD obtained via numerical integration of (4) (via $\mathbb{I}$) with $\boldsymbol{\theta}_t$ as the parameter and letting $\hat{F}_T(\delta) = [\hat{f}_T^1(\delta), \hat{f}_T^2(\delta)]^\top$ denote the prediction of Φ for delta perturbed $\boldsymbol{\theta}$, we denote the mean absolute error (MAE) $E_\Phi(\delta) = [E_\Phi^1(\delta), E_\Phi^2(\delta)]^\top$, with $E_\Phi^j(\delta) := \frac{1}{B}\sum_{k=1}^{B} |\bar{f}_T^{j,k} - \hat{f}_T^{j,k}(\delta)|$, for $j \in \{1,2\}$, $\delta \geq 0$ and batch size $B = 100$. Based on this the plot of MAE E_Φ for increasing values of δ is as shown in Fig. 4(a). From this, we see that the trained model shows sub-linear increase in error and the error for large perturbation of order 10 is still within tolerable region of .05. Also, the variation in the error across different batches, indicated by the confidence bands shown as shaded regions around the solid line, is fairly narrow. Altogether, we can infer that the Φ network is robust to uncertain process dynamics and is able to predict the behavior of the process with good order of accuracy. Furthermore, since the Φ has a broad input (parameter) range (refer Table I) it can provide stable predictions even for large growth parameters $\boldsymbol{G}$ which can usually be problematic due to the for classical numerical integration due to the numerical instability of discrete advection operator [25], [26]. These two factors combined with the generalizability property of a DNN renders the Φ model adaptable to the varying as well as uncertain dynamics of the process. Next we performed a similar robustness experiment with the controller network Λ where we sequentially fed the controller with perturbed feedback $\tilde{X}_t = [\tilde{F}_t, \tilde{C}_t]$ obtained from a numerically simulated process with increasing intensity of the process noise parameter $\sigma_t^j = \sigma$, $j \in \{1,2\}$. This corresponds to a closed-loop simulation with a noisy simulated process model which we ran for 200 time steps with sampling time $t_s = .1$, which in total corresponds to $T = 20\mathrm{mins}$ of sequential batch control. Based on this we computed the MAE $E_\Lambda(\sigma) := [E_\Lambda^1(\sigma), E_\Lambda^2(\sigma)]^\top$ with $E_\Lambda^j(\sigma) := \sum_{k=1}^{B} |\hat{f}_T^{j,k}(\sigma) - \bar{f}_T^{j,k}|$, for $j \in \{1,2\}$ of the final controlled PSD $\hat{F}_T(\sigma)$ from the prescribed target PSD $\bar{F}_T$ on the test set with batch size $B = 100$. The errors so obtained for different values of σ are as shown in Fig. 4(b). The error plot indicates that the controller is able to drive the process toward the prescribed target PSD even when being operated under increasing intensity of noisy feedback. Even with higher than 5 fold increase in the noise intensity, the MAE $E_\Lambda(\sigma)$ was less than the tolerable threshold of .07. Furthermore, the variation in error across different batches, indicated by the confidence bands shown as shaded regions around the solid line, is also fairly narrow. These observations indicate the robustness of the controller to external disturbances. Altogether, from Fig. 4(a),(b) we can infer that $\Lambda(\Phi)$ is an adaptive and robust controller. Following this we applied it for the control of simulated precipitation process with default noise intensity of $\boldsymbol{\sigma} = \mathrm{diag}([.1,.1]^\top)$ for a fixed initial and target PSD F_0 and F_T respectively. We performed sequential batch operations for 100 steps with sampling time $t_s = .1$ which corresponds to a total time of $T = 30\mathrm{mins}$. The results of the simulation are as shown in Fig. 3, which shows that $\Lambda(\Phi)$ was able to achieve the required PSD in roughly $30\mathrm{mins}$.

VI. Conclusions

In this work, we have introduced a novel learning-based adaptive and robust controller (LARC) $\Lambda(\Phi)$ for the task of controlling the CPBP in a selective manner. By the use of deep neural network architectures, including UNet for predicting the particle size distribution and GRU for generating control sequences, we were able to obtain fairly robust and adaptive controller. The numerical experiments indicate that Φ model not only provides accurate predictions of both particle size distribution and ion concentration but is also adaptive and robust to varying process parameters, thus demonstrating its potential for effective prediction of the real process dynamics. The ability of the controller $\Lambda(\Phi)$ to drive the system from an initial distribution to a predefined target, even in presence of moderate amount of noise, highlights its practical efficacy. Furthermore, the decoupled process model and control signals enable selective precipitation, and the use of Φ as the prediction model facilitates active online learning based on real data. Based on these, some of the planned work for the near future involves (i) incorporating measured data in the training, (ii) adapting the process model with a more realistic coupling involving particle interactions conditioned on different operating conditions such as pH, temperature, and pressure and finally (iii) to also incorporate, the aforementioned, operational variables as control variables.

References

[1] W. Seifritz, "CO2 disposal by means of silicates," *Nature*, vol. 345, no. 6275, pp. 486–486, jun 1990.

[2] F. Goff and K. S. Lackner, "Carbon Dioxide Sequestering Using Ultramafic Rocks," *Environmental Geosciences*, vol. 5, no. 3, pp. 89–102, sep 1998.

[3] A. A. Olajire, "A review of mineral carbonation technology in sequestration of CO2," *Journal of Petroleum Science and Engineering*, vol. 109, pp. 364–392, sep 2013.

[4] E. R. Bobicki, Q. Liu, Z. Xu, and H. Zeng, "Carbon capture and storage using alkaline industrial wastes," *Progress in Energy and Combustion Science*, vol. 38, no. 2, pp. 302–320, apr 2012.

[5] A. Sanna, M. Uibu, G. Caramanna, R. Kuusik, and M. M. Maroto-Valer, "A review of mineral carbonation technologies to sequester CO 2," *Chem. Soc. Rev.*, vol. 43, no. 23, pp. 8049–8080, 2014.

[6] P. D. Christofides, *Nonlinear and Robust Control of PDE Systems*, ser. Systems & Control: Foundations & Applications. Boston, MA: Birkhäuser Boston, 2001.

[7] T. Y. Chiu and P. D. Christofides, "Robust control of particulate processes using uncertain population balances," *AIChE Journal*, vol. 46, no. 2, pp. 266–280, feb 2000.

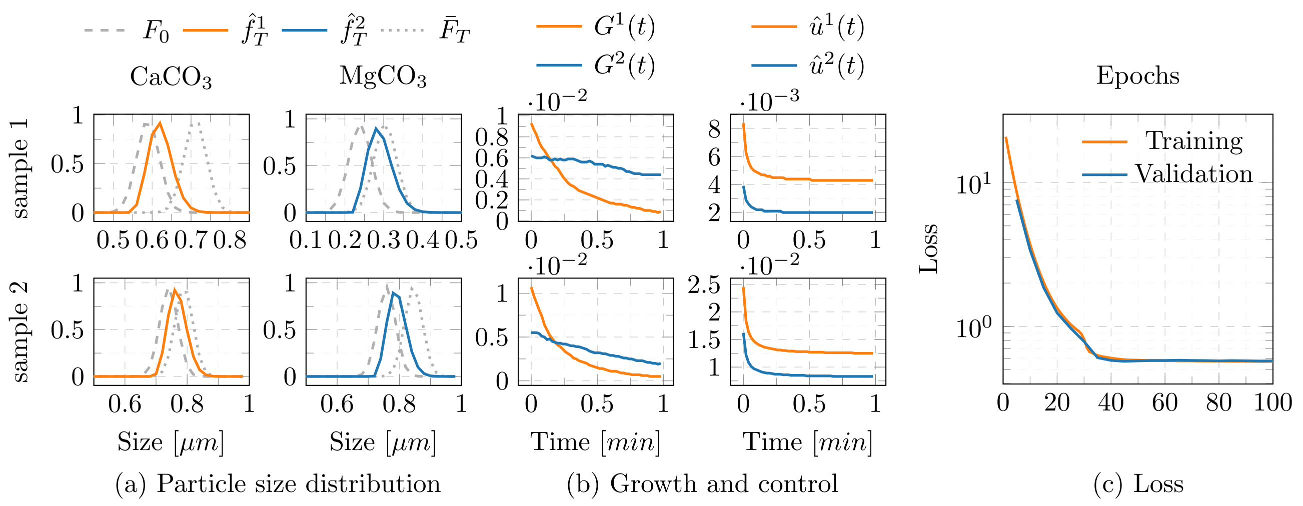


Fig. 2: Sample outputs of the LARC network $\Lambda(\Phi)$ and its training and validation loss.

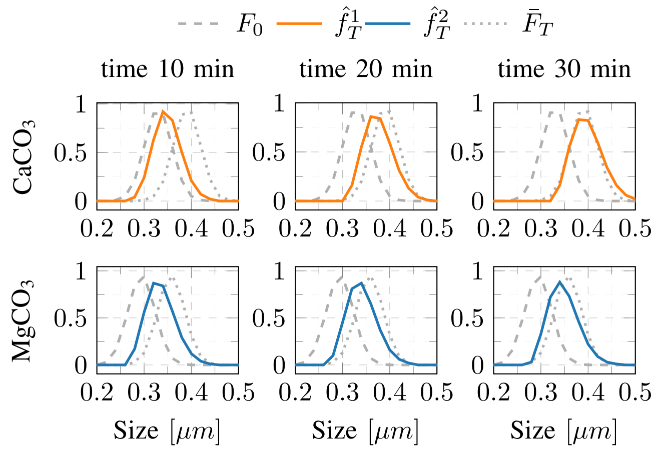


Fig. 3: Batch control of the precipitation process

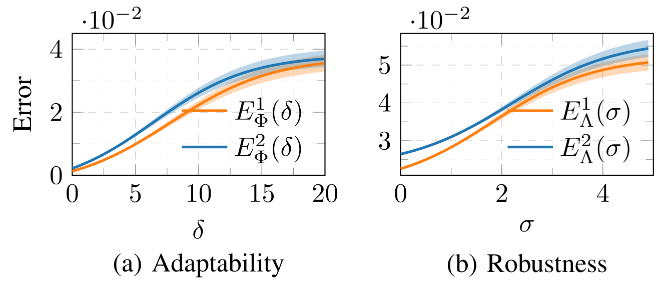


Fig. 4: Adaptability and robustness tests for Φ and $\Lambda(\Phi)$

[8] A. Armaou and M. A. Demetriou, "Optimal actuator/sensor placement for linear parabolic PDEs using spatial H2 norm," *Chemical Engineering Science*, vol. 61, no. 22, pp. 7351–7367, 2006.

[9] A. K. Rajagopalan, S. Bötschi, M. Morari, and M. Mazzotti, "Feedback Control for the Size and Shape Evolution of Needle-like Crystals in Suspension. III. Wet Milling," *Crystal Growth and Design*, vol. 19, no. 5, pp. 2845–2861, 2019.

[10] B. Ydstie, "Passivity based control via the second law," *Computers & Chemical Engineering*, vol. 26, no. 7, pp. 1037–1048, 2002.

[11] M. Krstic and A. Smyshlyaev, *Boundary Control of PDEs*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2008.

[12] N. Chen, J. Dai, X. Zhou, Q. Yang, and W. Gui, "Distributed Model Predictive Control of Iron Precipitation Process by Goethite Based on Dual Iterative Method," *International Journal of Control, Automation and Systems*, vol. 17, no. 5, pp. 1233–1245, may 2019.

[13] D. Shi, N. H. El-Farra, M. Li, P. Mhaskar, and P. D. Christofides, "Predictive control of particle size distribution in particulate processes," *Chemical Engineering Science*, vol. 61, no. 1, pp. 268–281, jan 2006.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013.

[15] N. Chen, S. Luo, J. Dai, B. Luo, and W. Gui, "Optimal Control of Iron-Removal Systems Based on Off-Policy Reinforcement Learning," *IEEE Access*, vol. 8, pp. 149 730–149 740, 2020.

[16] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Data-driven discovery of partial differential equations," *Science Advances*, vol. 3, no. 4, apr 2017.

[17] M. Raissi, "Deep hidden physics models: Deep learning of nonlinear partial differential equations," *The Journal of Machine Learning Research*, vol. 19, no. 1, pp. 932–955, 2018.

[18] J. Han, A. Jentzen, and W. E, "Solving high-dimensional partial differential equations using deep learning," *Proceedings of the National Academy of Sciences*, vol. 115, no. 34, pp. 8505–8510, aug 2018.

[19] *Investigations on the CO2 Corrosion of Mild Steel in the Presence of Magnesium and Calcium Ions*, ser. NACE CORROSION, vol. All Days, 06 2020.

[20] J. Han, A. Jentzen, and W. E, "Solving high-dimensional partial differential equations using deep learning," *Proceedings of the National Academy of Sciences*, vol. 115, no. 34, pp. 8505–8510, aug 2018.

[21] M. Crowder, *Stochastic Approximation: A Dynamical Systems Viewpoint*. Springer, aug 2009, vol. 77, no. 2.

[22] A. Jentzen and P. E. Kloeden, *Taylor Approximations for Stochastic Partial Differential Equations*. Society for Industrial and Applied Mathematics, jan 2011, no. 83.

[23] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.

[24] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

[25] R. Courant, K. Friedrichs, and H. Lewy, "Über die partiellen Differenzengleichungen der mathematischen Physik," *Mathematische Annalen*, vol. 100, no. 1, pp. 32–74, dec 1928.

[26] J. Manzanero, G. Rubio, E. Ferrer, and E. Valero, "Dispersion-Dissipation Analysis for Advection Problems with Nonconstant Coefficients: Applications to Discontinuous Galerkin Formulations," *SIAM Journal on Scientific Computing*, vol. 40, no. 2, pp. A747–A768, January 2018.