PSO-Based Adaptive NMPC for Uranium Extraction-Scrubbing Operation in Spent Nuclear Fuel Treatment Process *

Duc-Tri Vo *, ** **Ionela Prodan** * **Laurent Lefèvre** *
Vincent Vanel ** **Sylvain Costenoble** ** **Binh Dinh** **

*Univ. Grenoble Alpes, Grenoble INP, LCIS, 26000, Valence, France
(e-mail: {duc-tri.vo, ionela.prodan,
laurent.lefevre}@lcis.grenoble-inp.fr).*
** *CEA, DES, ISEC, DMRC, Univ Montpellier, Marcoule, France
(e-mail: {vincent.vanel, sylvain.costenoble, binh.dinh}@cea.fr)*

Abstract: This paper addresses the particularities of adaptive optimal control of the uranium extraction-scrubbing operation in the PUREX process. The process dynamics are nonlinear, high dimensional, and have limited online measurements. In addition, analysis and developments are based on a qualified simulation program called PAREX, which was validated with laboratory and industrial data. The control objective is to stabilize the process at a desired solvent saturation level, guaranteeing constraints and handling disturbances. The developed control strategy relies on optimization-based methods for computing control inputs and estimates, i.e., Nonlinear Model Predictive Control (NMPC) and Nonlinear Moving Horizon Estimation (NMHE). The designs of these two associated algorithms are tailored for this process's particular dynamics and are implemented through an enhanced Particle Swarm Optimization (PSO) to guarantee constraint satisfaction. Software-in-the-loop simulations using PAREX show that the designed control scheme effectively satisfies control objectives and guarantees constraints during operation.

Keywords: PUREX, PAREX, Liquid-liquid Extraction, Nonlinear Model Predictive Control, Moving Horizon Estimation, Particle Swarm Optimization, Derivative-free Optimization.

1. INTRODUCTION

1.1 Motivation

Spent nuclear fuels from nuclear reactors predominantly comprise 95% uranium and 1% plutonium, both of which are amenable to recovery. The remaining 4%, characterized by its pronounced radioactivity, is classified as the ultimate waste. The PUREX process, denoting "Plutonium, Uranium, Reduction, EXtraction," was conceived to effectuate the recovery and purification of uranium and plutonium from spent nuclear fuels (Vaudano (2008)). It finds application at the La Hague nuclear fuel reprocessing plant in northern France, owing to its ability to recover these materials with an elevated degree of purity. This procedural approach facilitates the recycling of uranium and plutonium to produce fresh fuel resources while curtailing the volume of ultimate waste. Therefore, the PUREX process advances sustainable fuel cycles and nuclear waste management paradigms.

One of the primary objectives of the PUREX process is to attain a notable recovery efficiency for uranium and plutonium while concurrently achieving a high decontamination factor of those elements toward the fission products.

This objective, performed to several liquid-liquid extraction steps, can be realized through the deliberate manipulation of solvent saturation, accomplished by modulating the flow rate of the feed solution, i.e., the quantity of uranium introduced into the process can be increased or decreased as necessary. This study directs its attention to the extraction-scrubbing operation, which is the first step within the extraction cycles and substantially influences the overall efficiency of the process.

1.2 PUREX Modelling: The PAREX Code

The French Alternative Energies and Atomic Energy Commission (CEA) developed a simulation software, PAREX, which has the ability to emulate the extraction operations within the PUREX process. This computational tool was validated through rigorous comparison with laboratory and industrial-scale experimental data, affirming its proficiency in effectively replicating PUREX process operations under transient and steady-state conditions, as reported by Bisson et al. (2016). PAREX finds application at the La Hague facility, aiming to optimize processes, troubleshoot operational issues, and conduct safety analyses (Bisson et al. (2016)). Within this study, the PAREX code is the primary tool employed for system dynamics analysis and the formulation of control strategies. Since the process dynamics are complicated, we aim to integrate the PAREX simulator directly into the control scheme (as the predictor

and the estimator). This way, we can benefit from the qualified mathematical model and the efficient, robust numerical solver employed in PAREX.

1.3 Literature Review

To the best of our knowledge, the research landscape concerning this specific process control is notably limited. A study within the same domain is reported in Vo et al. (2023), wherein a tracking Nonlinear Model Predictive Control (NMPC) strategy was designed to maximize process production. The simulation outcomes, both under nominal and perturbed conditions, demonstrated the efficacy of NMPC in stabilizing the process at its optimal working condition. However, this work used a simplified model for both NMPC developments and process simulation, which had errors compared to the qualified model in PAREX and the actual process. In addition, all the states and disturbances were assumed to be perfectly measured, which is a strong assumption compared to practical implementation conditions. Finally, to broaden the view, studies on the control of similar processes can be found in Vo et al. (2023) and the references therein.

Our particular control problem is nonlinear and subject to hard constraints. Thus, Model Predictive Control, a well-known control method in academia and industry (Qin and Badgwell (2003)), is a promising candidate. The basic idea of MPC is to find the control inputs in a prediction horizon that minimize a cost function. In other words, MPC solves an optimization problem online to obtain optimal control inputs. For tracking problems, the cost function is usually chosen to represent the tracking error in the prediction horizon. However, another challenge in this study stems from the limited availability of online measurements: only one state is measured online. The choice of measurements is made by considering sensitivity analysis (to have measurements on the most sensitive points), type of device (pulsed column, centrifugal extractor, mixer-settler), and cost. It is possible to carry out several measurement points online, but this study focuses on the case with one measurement.

Consequently, there arises a need for an estimator to compute estimations of states and parameters using the available online measurements. These estimated values are subsequently integrated into the NMPC as feedback, constituting a configuration commonly denoted as "output feedback MPC" or "adaptive MPC" (Jabarivelisdeh et al. (2020); Valipour and Ricardez-Sandoval (2022)). A promising candidate for this purpose is the optimization-driven Moving Horizon Estimation (MHE) method (Rawlings (2013)). MHE estimates states and parameters simultaneously by minimizing the errors between nominal outputs and online measurements. It has widespread application across diverse domains, particularly within process control (Liu et al. (2018); Tuveri et al. (2023)). MPC and MHE theory, computation, and implementation are discussed systematically and comprehensively in James and David (2022).

Implementing MPC and MHE involves solving their optimization problems. Thus, an optimization algorithm is required. Particle Swarm Optimization (PSO) is a derivative-free, population-based optimization algorithm inspired by natural collective behavior observed in birds and fish, as initially proposed by Kennedy and Eberhart (1995). Within the PSO framework, each particle position is represented as a vector containing a candidate solution for the optimization problem. Initially, particles are distributed in the search space. Then, a cost value is attributed to each particle based on their position. Subsequently, particles undergo iterative movements in pursuit of the optimal position associated with the lowest cost. Successful applications of PSO in nonlinear control and estimation are reported in Schwaab et al. (2008); Zietkiewicz et al. (2020); Gbadega and Sun (2022). An essential feature of PSO is its independence from the necessity of function gradients, thereby obviating the need for an explicit analytical representation of the cost function and system dynamics. This characteristic aligns with our control objectives, where all computations are based on the "black box" PAREX.

1.4 Contributions and Paper Organization

This paper introduces a non-linear adaptive control strategy tailored to the PUREX process's uranium extraction-scrubbing operation. The proposed control approach combines non-linear Model Predictive Control, Moving Horizon Estimation, and Particle Swarm Optimization. Notably, the control strategy relies on the qualified simulation software PAREX as the primary computational instrument, as no explicit mathematical model equations is required. Briefly, our main contributions are:

- proposing an adaptive control strategy based on NMPC and MHE to account for constraints satisfaction (e.g., uranium concentration in the fission product, equipment limits), setpoint tracking (i.e., aqueous uranium concentration at stage 9 settler), and parameters uncertainty (i.e., variation in the fresh solvent flow rate);
- proposing an enhanced PSO implementation for both the NMPC and MHE algorithms that can efficiently handle constraints on states, overshoots, control inputs, rate of changes of control inputs, and parameter estimation;
- study the effectiveness of the proposed control strategy through software-in-the-loop (SIL) testing using the validated PAREX simulator developed by CEA.

The paper is organized as follows. The process dynamics and the control problem setup is presented in Section 2. Then, the control strategy for the control problem is developed in Section 3. Section 4 presents our enhanced PSO algorithm to solve the optimization problems in the control strategy. Simulation results of nominal and disturbed case are presented in Section 5. Finally, conclusions are stated in Section 6.

Notations: Vectors are denoted by bold lowercase letters, and matrices are denoted by capital letters. $\mathbf{I}$, $\mathbf{0}$ denotes identity and zero matrices of appropriate dimensions. $\mathbf{x}_n$ denotes the nth element of $\mathbf{x}$. $\mathbf{x}_{n:m}$ denotes a vector slice from the nth to m^{th} (included) elements of $\mathbf{x}$. $\|\mathbf{x}\|_{\mathbf{Q}} := \mathbf{x}^T \mathbf{Q} \mathbf{x}$. $\mathbb{N}_{m:n} := \{i \in \mathbb{N} | m \leq i < n\}$. $|\mathcal{C}|$ denotes the cardinality of the set $\mathcal{C}$. $U(\mathbf{a}, \mathbf{p})$ denotes a uniformly random distribution within bounds $\mathbf{a}$, $\mathbf{p}$. Inequalities between vectors are element-wise.

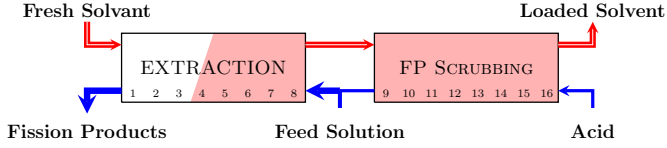# 2. URANIUM EXTRACTION-SCRUBBING OPERATION

Fig. 1. Extraction - fission products (FP) scrubbing step (Vo et al. (2023)).

Fig. 1 illustrates the fundamental architecture of our process, characterized by a sequential arrangement of 16 mixer-settler stages. The mathematical model within the PAREX platform was developed following a similar approach as the simplified one in the work by Vo et al. (2023). The main difference is the inclusion of specific experimentally derived parameters. Consequently, we adopt the discretized state space representation (1) to derive the control strategy. It is worth noting that while the definition of $\mathbf{f}$ is not explicitly given, its evaluation can be obtained utilizing PAREX.

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), u(k), q(k)), \qquad (1a)$$
$$y(k+1) = \mathbf{x}_{25}, \qquad (1b)$$

where:

- $\mathbf{x} \in \mathbb{R}^{128 \times 1}$, uranium and acid concentrations in aqueous and organic phases of each stage, are system states;
- $u = A_F$, the feed solution flow rate, is the manipulated variable;
- $q = O_E$, the solvent flow rate, is the unknown disturbance. This type of disturbance can appear when there is a failure of mechanical coupling, pumps or flow meters in the system;
- $y = [U]_9^{\mathrm{aqD}}$, the aqueous uranium concentration at stage 9, is the controlled variable. This choice is based on the fact that this variable can effectively indicate the solvent saturation level, and its value is compatible with the uranium concentration sensor's operation range.

Multiple time scales: The control problem has to handle tasks at different time scales: i) first, it is necessary to read the measurements, which should be done *online* with sufficient small sampling time, T_{meas}, to obtain relevant feedback from the system; ii) second, is to detect any system disturbances by comparing measurements with simulation results. This task is done with an estimation sampling time T_{esti}; iii) lastly, it is necessary to find the system's optimal control inputs, which generally requires the highest computation time. In addition, since the process dynamics is slow, the control sampling time T_{ctrl} can be chosen as the largest sampling time, i.e., $T_{\mathrm{ctrl}} \geq T_{\mathrm{esti}} \gg T_{\mathrm{meas}}$. For convinience in the discrete control scheme, we denote $N_{\mathrm{ctrl}} = T_{\mathrm{ctrl}}/T_{\mathrm{meas}}$ and $N_{\mathrm{esti}} = T_{\mathrm{esti}}/T_{\mathrm{meas}}$.

2.1 Solvent Saturation and Desired Set-point

Saturation is a non-linear characteristic of this process dynamics, as illustrated in Fig. 2. Once the system is over-saturated, i.e., $A_F > A_F^*$, $[U]_1^{\mathrm{aqD}}$ increases drastically. Thus, uranium will go into extraction raffinates, which must be avoided. In addition, the system can quickly become over-saturated with a smaller safety margin, especially with uncertainties. Therefore, keeping the system working in the under-saturation region but close to the critical condition is desirable. The choice of the set point depends on the expected performances (recovery rate, decontamination factor, the specifics of the chemical system (extractant, media, concentration), and the feed solution composition). Latter in the simulations, we choose $[U]_{9,\mathrm{set}}^{\mathrm{aqD}} = 0.5[U]_9^{\mathrm{aqD}*}$ as illustrated in Fig. 2.
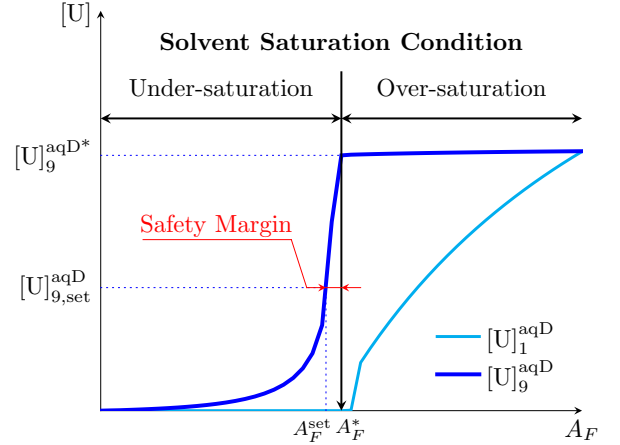

Fig. 2. Steady state relationship of feed solution flow rates and uranium concentrations.

2.2 Problem Statement

The control problem is stated as follows: based on the PAREX Code and the state space representation (1), develop a control system that stabilizes $y = [U]_9^{\mathrm{aqD}}$ at a desired set point $y_{\mathrm{set}} = [U]_{9,\mathrm{set}}^{\mathrm{aqD}}$ in the presence of disturbance. Additionally, constraint on uranium concentration in the extraction raffinates $[U]_1^{\mathrm{aqD}}$ (2a), overshoots OS (2b), control inputs A_F (2c), and rates of control inputs (2d) must be guaranteed during operation. Note that $\Delta_{A_F}^{\mathrm{max}}$ is the maximum allowable variation in A_F and (2d) is considered as a soft constraint and can be relaxed to guarantee the hard constraints (2a)-(2c):

$$[U]_1^{\mathrm{aqD}} \leq [U]_{1,\mathrm{tol}}^{\mathrm{aqD}}, \qquad (2a)$$
$$\mathrm{OS} = y/y_{\mathrm{set}} - 1 \leq \mathrm{OS}^{\mathrm{max}}, \qquad (2b)$$
$$A_F^{\mathrm{min}} \leq A_F \leq A_F^{\mathrm{max}}, \qquad (2c)$$
$$|A_F(k+1) - A_F(k)| \leq \Delta_{A_F}^{\mathrm{max}}. \qquad (2d)$$

Furthermore, in our control problem, the steady tracking error e is not necessarily zero, which is hard to achieve in the presence of uncertainties. We accept that the system is stabilized if the tracking error is below a predefined tolerance (3), where $\varepsilon_{\mathrm{ss}} \in (0, 1)$ is a relative factor:

$$e = |y - y_{\mathrm{set}}| \leq \varepsilon_{\mathrm{ss}} y_{\mathrm{set}}. \qquad (3)$$

As we will see, this choice helps reduce MPC's computational cost and, thus, makes the practical implementation of the control strategy more efficient.
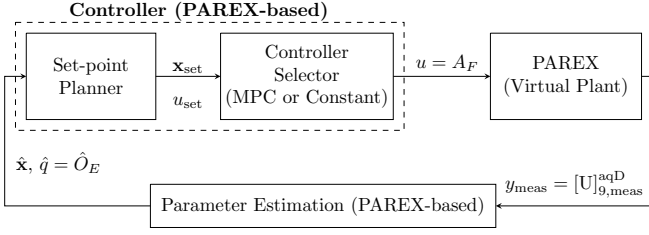
3. CONTROL STRATEGY

Fig. 3. Proposed MPC-MHE control strategy using the PAREX simulator in the loop.

This section presents the developed control strategy illustrated in Fig. 3. First, the online measurement y_{meas} is read and then sent to the Moving Horizon Estimator (MHE) for state and parameter estimations. After that, estimated states and parameters are used to compute the control input u. We have a controller selector inside the controller block that allows us to turn off MPC and use constant control inputs under some conditions. This behavior allows us to reduce computation costs while adhering to control objectives and constraints.

3.1 MHE for Parameter Estimation

As shown in Fig. 3 we delineate in the following the MHE procedure for estimating the solvent flow rate, that is, the O_E parameter in (1) and then compute the estimated state values. This approach avoids solving a large-scale optimization problem since the system is already high dimensional with 128 states. Firstly, denote e_{esti} as the estimation errors, i.e., the deviation between the measurements y_{meas}, and the estimated value $\hat{y}$:

$$e_{\mathrm{esti}} = y_{\mathrm{meas}} - \hat{y}. \tag{4}$$

Note that, $y = [U]_9^{\mathrm{aqD}}$ is the aqueous uranium concentration at stage 9. Denote $\mathcal{K}^E = \{k_{-N_e}, \ldots, k\}$ as the set of co-incident points at which we want to minimize e_{esti}. Regarding the set $\mathcal{K}^E$, at the first time step k_{-N_e}, the state value must be good enough since $\mathbf{x}\left(k_{-N_e}\right)$ is the initial condition for the optimization problem (6). In this work, we choose k_{-N_e} corresponding to the latest window of size N_w: $[k_{-N_e} - N_w, k_{-N_e}]$ in which e_{esti} is always below a tolerance ε_{-N_e}:

$$\max_{i \in [k_{-N_e} - N_w, k_{-N_e}]} |e_{\mathrm{esti}}| \leq \varepsilon_{-N_e}. \tag{5}$$

Denote $k_{-1} := k$ as the rest of the time steps, i.e. $k_{-N_e+1}, \ldots, k_{-2}$, which are evenly spaced samples on the interval $[k_{-N_e}, k_{-1}]$. Hence, the estimated parameter $\hat{q}(k)$ can be obtained by solving the following optimization problem, $\forall i \in \mathcal{K}_k^E$:

$$\min_{\{\hat{q}(i)\}_{i \in \mathcal{K}_k^E}} \sum_{i \in \mathcal{K}_k^E} \alpha_i e_{\mathrm{esti}}^2, \tag{6a}$$

subject to:

$$\hat{\mathbf{x}}(i+1) = \mathbf{f}(\hat{\mathbf{x}}(i), u(i), \hat{q}(i)), \tag{6b}$$

$$\hat{y}(i) = \hat{\mathbf{x}}_{25}(i), \tag{6c}$$

where α_i is the weighting factor and $\mathcal{K}_k^E$ is the set of co-incident points at which e_{esti} is minimized. Putting high

weighting factors on the latest co-incident points is preferable since the latest feedback is much more important.

One observation is that estimating $q = O_E$ by solving (6) is costly, and the computational cost can be reduced by setting $\hat{q}(k) = \hat{q}(k-1)$ if the following condition holds:

$$|e_{\mathrm{esti}}| \leq \varepsilon_{\mathrm{active}}, \tag{7}$$

where $\varepsilon_{\mathrm{active}}$ represents the acceptable deviation between the estimated value and online feedback.

The algorithm for implementing the parameter estimation is briefly summarized as follows. First, we read the measurements up to time k. After, we check if (7) holds so that we can set $\hat{q}(k) = \hat{q}(k-1)$. Otherwise, we compute $\hat{q}$ using the MHE method, which includes constructing κ^E and solving (6). The estimated state corresponding to $\hat{q}$ is then obtained using PAREX.

3.2 Controller Design

This subsection introduces the set-point tracking NMPC and the controller selector as shown in Fig 3. Firstly, PAREX is used to find the set point $(\mathbf{x}_{\mathrm{set}}, u_{\mathrm{set}})$ by solving the steady-state equation, where $y_{\mathrm{set}} = \mathbf{x}_{\mathrm{set}, 25}$:

$$\mathbf{f}(\mathbf{x}(k), u(k), \hat{q}(k))|_{y(k)=y_{\mathrm{set}}} = 0. \tag{8}$$

The control sampling time T_{ctrl} is significantly larger than the measurement sampling time T_{meas}. Hence, to describe the MPC algorithm, denote $\mathcal{K}^C$ the set of time steps at which we compute the control input using MPC:

$$\mathcal{K}^C := \{k | (k \mod N_{\mathrm{ctrl}}) = 1, \forall k \in \mathbb{N}_{1:N_{\mathrm{sim}}}\}, \tag{9}$$

where N_{sim} denotes the total simulation time. Then, the set of control time steps in the prediction horizon at time step k, $\mathcal{K}_k^C$, is given as

$$\mathcal{K}_k^C := \{k_p \in \mathcal{K}^C \mid k \leq k_p \leq k + N_p N_{\mathrm{ctrl}}\}, \tag{10}$$

with N_p denoting the prediction horizon. At the control time step $k \in \mathcal{K}^C$, given the estimated initial state $\hat{\mathbf{x}}(k)$, denote $\mathbf{x}(i|k)$ the predicted state at time step j driven by the predicted control inputs $\{u(j|k)\}_{j=k}^i$, $\forall i \in \mathbb{Z}^+$, $i > k$. Assuming that the control input u is constant over the interval $[jk, jk + N_{\mathrm{ctrl}} - 1]$ $\forall j \in \mathbb{N}_{0:N_p-1}$ and denoting the tracking error for the state by $\tilde{\mathbf{x}} := \mathbf{x} - \mathbf{x}_{\mathrm{set}}$ and for the input by $\tilde{u} := u - u_{\mathrm{set}}$, then we define the quadratic cost function as:

$$\ell_{\mathrm{MPC}} = \sum_{i \in \mathcal{K}_k^C} \left(\|\tilde{\mathbf{x}}(i|k)\|_{\mathbf{Q}}^2 + \|\tilde{u}(i|k)\|_{\mathbf{R}}^2 \right)$$
$$+ \sum_{i \in \mathcal{K}_k^C} \|u(i|k) - u(i-1|k)\|_{\mathbf{S}}^2 + \|\tilde{\mathbf{x}}(N+1|k)\|_{\mathbf{P}}^2, \tag{11}$$

where $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{P}$, $\mathbf{S}$ denote symmetric positive definite weighting matrices of appropriate dimensions. The optimization problem is defined as:

$$\min_{\{u(i|k)\}_{i \in \mathcal{K}_k^C}} \ell_{\mathrm{MPC}}, \tag{12a}$$

subject to:

$$\mathbf{x}(j+1|k) = \mathbf{f}(\mathbf{x}(j|k), u(j|k), \hat{q}(k)), \tag{12b}$$

$$\mathbf{x}(k|k) = \hat{\mathbf{x}}(k), \tag{12c}$$

$$u(k-1|k) := u(k-1), \tag{12d}$$

$$\mathbf{x}(i|k) \geq 0, \tag{12e}$$

$$[U]_1^{\mathrm{aqD}}(i|k) = \mathbf{x}_{17}(i|k) \leq [U]_{1,\mathrm{tol}}^{\mathrm{aqD}}, \tag{12f}$$

$$OS(i|k) \leq y(i|k)/y_{\text{set}} - 1 \leq OS^{\max}, \qquad (12g)$$
$$A_F^{\min} \leq u(i|k) \leq A_F^{\max}, \qquad (12h)$$
$$|u(i|k) - u(i-1|k)| \leq \Delta A_F^{\max}. \qquad (12i)$$

To reduce the computation cost of NMPC, we propose the *Controller Selector*, which allows to switch to a constant controller $u(k) = u_{\text{set}}$ without solving the NMPC optimization problem. The switching to the constant controller is done if by predicting over the next N_{p2} time steps, the condition (3) holds, i.e.:

$$|y(i|k) - y_{\text{set}}| \leq \varepsilon_{\text{ss}} y_{\text{set}}, \forall i \in \mathcal{K}_{k2}^C, \qquad (13a)$$
$$\mathcal{K}_{k2}^C := \left\{ k_p \in \mathcal{K}^C \mid k \leq k_p \leq k + N_{p2} N_{\text{ctrl}} \right\}. \qquad (13b)$$

From our experience, N_{p2} is preferred to be much larger than N_p to ensure that the system is about to be steady.

The controller design is summarized as follows: We first get the estimated values from the MHE. If condition (13) holds, we can consider $u(k) = u_{\text{set}}$. Otherwise we obtain $u(k)$ by solving the NMPC optimization problem (12).

4. PARTICLE SWARM OPTIMIZATION FOR THE PROPOSED MPC AND MHE DESIGNS

With the PAREX simulator integrated into the control loop as in Fig. 3, we propose the use of PSO to solve the NMPC and MHE optimization problems developed in the previous section. The advantage is that PSO does not require an explicit declaration of the mathematical equations of the model; it allows the use of the "black box" PAREX to solve optimization problems. In this section, we developed and extended the PSO formulation to accommodate the NMPC and MHE designs. The PSO implementation will be analyzed here for the MPC optimization problem (12) and can be similarly applied for solving the MHE optimization problem (6). Since multiple notations are used, we will first describe the framework of the PSO algorithm in Sec. 4.1, then present our extensions later in Sec. 4.2.

4.1 Global-best Guaranteed Convergence PSO

Basic Global-best PSO Assume that the swarm has N_{part} particles. At iteration i, denote the nth particle's position and cost by $\mathbf{p}_n^{(i)}$, and $\ell_n^{(i)} := \ell(\mathbf{p}_n^{(i)})$ respectively. In addition, denote $\bar{\mathbf{p}}^{(i)}$, $\bar{\ell}_n^{(i)}$ the *personal best* position and cost of the nth particle; $\bar{\bar{\mathbf{p}}}^{(i)}$, $\bar{\bar{\ell}}_n^{(i)}$ the *global best* position and cost of the swarm:

$$\bar{\mathbf{p}}_n^{(i)} := \underset{\mathbf{p}=\mathbf{p}_n^{(j)}, j \in \mathbb{N}_{0:i+1}}{\operatorname{argmin}} \ell(\mathbf{p}), \quad \bar{\ell}_n^{(i)} := \ell\left(\bar{\mathbf{p}}_n^{(i)}\right), \quad (14a)$$

$$\bar{\bar{\mathbf{p}}}^{(i)} := \underset{\substack{\mathbf{p}=\mathbf{p}_n^{(j)}, \\ j \in \mathbb{N}_{0:i+1}, n \in \mathbb{N}_{0:N_{\text{part}}}}}{\operatorname{argmin}} \ell(\mathbf{p}), \quad \bar{\bar{\ell}}_n^{(i)} := \ell\left(\bar{\bar{\mathbf{p}}}^{(i)}\right). \quad (14b)$$

At each iteration, particle positions are updated with velocity velocities $\mathbf{v}_n^{(i)}$ given in (15a). Particles velocities are computed using (15b), with $\mathbf{r}_1^{(i)}, \mathbf{r}_2^{(i)} \sim U(\mathbf{0}, \mathbf{1})$, $w^{(i)} \in (0,1)$ is the inertia weight factor, and $c_1^{(i)}, c_2^{(i)} > 0$ are acceleration constants. In this work, the linear decreasing inertia weight and time-varying acceleration constants (15c)-(15e) are used, with $w_i = 0.9$, $w_f = 0.4$, $c_{1i} = c_{2f} =$

2.5, $c_{1f} = c_{2i} = 0.5$ as suggested in Naka et al. (2001) and Ratnaweera et al. (2004). Note that the velocity should be clamped within a predefined range to avoid large updates that make the particle go out of the search space or miss a potential search region. Initially, particle positions are scattered *uniformly* in the search space, and velocities are set to zero.

$$\mathbf{p}_n^{(i+1)} = \mathbf{p}_n^{(i)} + \mathbf{v}_n^{(i)}, \qquad (15a)$$
$$\mathbf{v}_n^{(i)} = w^{(i)} \mathbf{v}_n^{(i-1)} + c_1^{(i)} \mathbf{r}_1^{(i)} \left(\bar{\mathbf{p}} - \mathbf{p}_n^{(i)} \right)$$
$$+ c_2^{(i)} \mathbf{r}_2^{(i)} \left(\bar{\bar{\mathbf{p}}}^{(i)} - \mathbf{p}_n^{(i)} \right), \qquad (15b)$$
$$w^{(i)} = (w_f - w_i) \cdot (i_{\max} - i) / i_{\max} + w_f, \qquad (15c)$$
$$c_1^{(i)} = (c_{1f} - c_{1i}) i / i_{\max} + c_{1i}, \qquad (15d)$$
$$c_2^{(i)} = (c_{2f} - c_{2i}) i / i_{\max} + c_{2i}, \qquad (15e)$$

Guaranteed Convergence PSO In the basic PSO formulation, the $\bar{n}^{\text{th}}$ particle that has the *global best position* may become *stagnant* when $\mathbf{p}_{\bar{n}}^{(i)} = \bar{\mathbf{p}}_n^{(i)} = \bar{\bar{\mathbf{p}}}^{(i)}$, which leads to $\mathbf{v}_{\bar{n}}^{(i)} = w^{(i)} \mathbf{v}_n^{(i-1)} \to \mathbf{0}$. Consequently, this particle may stop searching for a better position, and the swarm tends to converge to $\bar{\bar{\mathbf{p}}}^{(i)}$, which may not be a minimum. To avoid this phenomenon, we update $\mathbf{p}_{\bar{n}}^{(i+1)}$ with the velocity given in (16), where $n_s(i)$ and $n_f(i)$ are the number of successive successes and failures up to iteration i (Engelbrecht (2007)). A success is defined when $\ell\left(\bar{\bar{\mathbf{p}}}^{(i+1)}\right) < \ell\left(\bar{\bar{\mathbf{p}}}^{(i)}\right)$.

$$\mathbf{v}_{\bar{n}}^{(i+1)} = -\mathbf{p}_{\bar{n}}^{(i)} + \bar{\bar{\mathbf{p}}}^{(i)} + w^{(i)} \mathbf{v}^{(i)} + \rho^{(i)} \left(1 - 2\mathbf{r}_2^{(i)} \right) \quad (16a)$$

$$\rho^{(i+1)} = \begin{cases} 2\rho^{(i)}, & n_s(i) > \epsilon_s, \\ 0.5\rho^{(i)}, & n_f(i) > \epsilon_f, \\ \rho^{(i)}, & otherwise. \end{cases} \qquad (16b)$$

Termination conditions For the searching process to be terminated, the swarm *cluster rate* r_{clus} must be larger than a minimum value:

$$r_{\text{clus}} = |\mathcal{C}|/N_{\text{part}} \geq r_{\text{clus}}^{\min}. \qquad (17)$$

The computing procedure for r_{clus} can be found in the work of Van Den Bergh and Engelbrecht (2002). We can avoid stopping the searching procedure prematurely by ensuring (17). Once (17) holds, the searching process is terminated if one of the following conditions is satisfied:

$$d\ell^{(i)} = \max_{j \in \{i-N_{\text{ter}}, \dots, i\}} \left| \bar{\bar{\ell}}^{(j)} - \bar{\bar{\ell}}^{(j-1)} \right| / \bar{\bar{\ell}}^{(i)} \leq \epsilon_{d\bar{\ell}}, \quad (18a)$$

$$v_{\max}^{(i)} = \max_{\substack{j \in \{i-N_{\text{ter}}, \dots, i\} \\ n \in \{0, N_{\text{part}}-1\}}} \left\| \mathbf{v}_n^{(j)} \right\| \leq \epsilon_v, \quad (18b)$$

$$\bar{\bar{\ell}}(i) \leq \epsilon_{\bar{\ell}}, \qquad (18c)$$
$$i = i_{\max}. \qquad (18d)$$

Alg. 1 describes the overall computing procedure. Firstly, the search space, then at each iteration, we: i) re-initialize the swarm using Alg. 2 and Alg. 3 to guarantee constraints; ii) apply the *Global-best Guaranteed Convergence PSO (gbest-GCPSO)* method for convergence guarantees; iii) remove duplicated particles, and iv) check if terminating condition holds.

Algorithm 1: gbest-GCPSO-based NMPC

---

foreach $j \in \mathbb{N}_{0:N_p}$ **do**
  $\mathbf{p}_{n,j}^{\mathrm{ub},(i)} \leftarrow \max\left\{\mathbf{p}^{\max}, \mathbf{p}_{n,j-1}^{(i),\mathrm{ub}} + \Delta_p\right\}$;
  $\mathbf{p}_{n,j}^{\mathrm{lb},(i)} \leftarrow \min\left\{\mathbf{p}^{\min}, \mathbf{p}_{n,j-1}^{(i),\mathrm{lb}} - \Delta_p\right\}$;
end
$\mathbf{p}_n^{(0)} \sim U\left(\mathbf{p}^{\mathrm{lb}}, \mathbf{p}^{\mathrm{ub}}\right)$; $\mathbf{v}_n^{(0)} = \mathbf{0}$;
for $i \in \mathbb{N}_{1:i_{max}}$ **do**
  for $n \in \mathbb{N}_{0:N_{part}}$ **do** $\mathbf{p}^{(i)} \leftarrow$ Alg. 2;
  $\bar{\bar{\mathbf{p}}}^{(i)}, \bar{\bar{\ell}}^{(i)} \leftarrow$ (14); $\rho^{(i)} \leftarrow$ (16b);
  for $n \in \mathbb{N}_{0:N_{part}}$ **do** $\mathbf{v}_n^{(i)} \leftarrow$ (15b);
  Remove duplicated particles;
  if (17) & *any in* (18) *holds* **then return** $\bar{\bar{\mathbf{p}}}^{(i)}$;
end

Algorithm 2: Swarm re-initialization

---

for $n \in \mathbb{N}_{0:N_{part}}$ **do**
  $\mathbf{p}_n^{(i)} \leftarrow$ (16a) and Alg. 3;
  while $\left\|\mathbf{v}_n^{(i)}\right\| \neq 0$ *and **not*** (12f)-(12g) **do**
    $\mathbf{v}_n^{(i)} \leftarrow \mathbf{v}_n^{(i)}/2$;
    $\mathbf{p}_n^{(i)} \leftarrow$ (16a) and Alg. 3;
  end
  while (12f)-(12g) **do**
    if $\mathbf{p}^{lb} \geq \mathbf{p}^{min}$ **then** $\mathbf{p}^{\mathrm{lb}} \leftarrow \mathbf{p}^{\mathrm{lb}} - \Delta_p$;
    else if $\mathbf{p}^{ub} \geq \mathbf{p}^{min}$ **then** $\mathbf{p}^{\mathrm{ub}} \leftarrow \mathbf{p}^{\mathrm{ub}} - \Delta_p$;
    else $\mathbf{p}_n^{(i)} \leftarrow \mathbf{p}^{\min}$;
  end
  $\mathbf{v}_n^{(i-1)} \leftarrow \mathbf{p}_n^{(i)} - \mathbf{p}_n^{(i-1)}$;
end

4.2 Proposed PSO extensions for MPC and MHE

In this subsection, we present the extensions tailored for this process dynamics after an in-depth analysis, as will be described later. We propose techniques for search-space initialization, swarm re-initialization, and duplicated particle removal.

Search-space initialization: For the MPC optimization problem (12) the search space can be limited based on the previous value of the input $u(k-1)$ (i.e., the initial guess), the input and its rate of change constraints (12h)-(12i), $\mathbf{p}^{\min} := A_F^{\min} + \mathbf{0}$, $\mathbf{p}^{\max} := A_F^{\max} + \mathbf{0}$, $\Delta_p := \Delta A_F^{\max} + \mathbf{0}$. Denote $\mathbf{p}_{n,j}$ the jth element of the position vector of particle n, the search space, i.e., the particle position bounds can be computed as shown at the beginning of Alg. 1. By narrowing the search space, we guarantee (12h)-(12i) and improve the optimum searching efficiency.

Swarm re-initialization for constraints guarantee (Alg. 2): The state constraints (12e)-(12g) require more careful observations than those on control inputs, which can be guaranteed by applying saturation as detailed in Alg. 3. Analyzing the process dynamics shows that guaranteeing on (12f)-(12g) can be achieved by reducing the control input magnitude. Therefore, reducing particle position can help reduce the constraint violation. To do this, we first try to reduce the velocity magnitude. Once the velocity is too small, and the constraints are not guaranteed, we relax the soft constraint (12i) and re-initialize the swarm with reduced bounds until: i) all particle positions satisfy (12f)-(12i), or ii) $\mathbf{p}^{\mathrm{lb}} = \mathbf{p}^{\mathrm{ub}} = \mathbf{p}^{\min}$. This algorithm is illustrated in Fig. 4, where the particles are shown in blue if constraints are guaranteed and in red if constraints are violated. The green dot denotes the initial guess of the optimization problem. We can see that the search space is narrowed after each time of re-initialization until all particles can guarantee constraints, $u(k) = A_F^{\min}$ in Fig. 4.

Duplicated particles removal: Duplicated particles are defined as those whose current position, best personal position, and velocity are the same. These particles will have the same behaviors in future iterations. Therefore, removing the duplicates will improve the computation cost and allow us to solve the optimization problem efficiently.

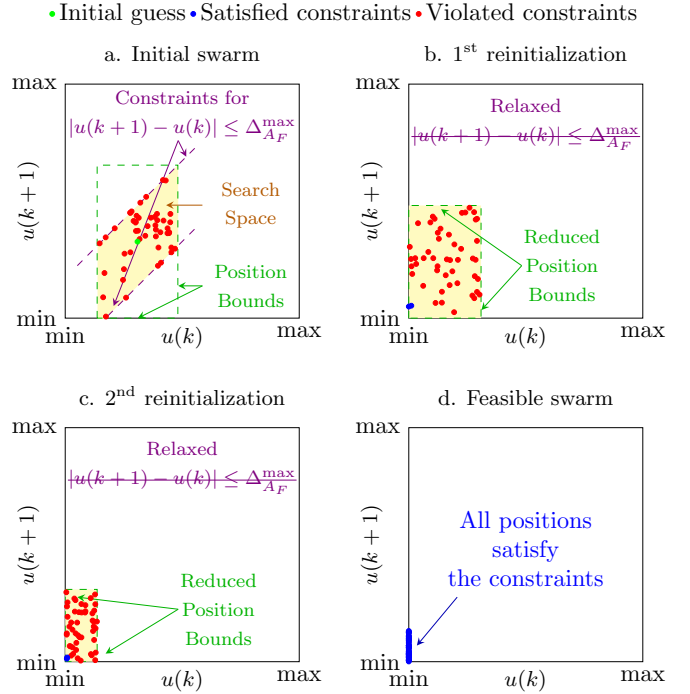· Initial guess · Satisfied constraints · Violated constraints

Fig. 4. Particle swarm re-initialization to guarantee constraints at $t = 4$ h in case (I), Sec. 5.

Algorithm 3: Guarantee position constraints.

$\mathbf{p}_{n,-1}^{(i)} \leftarrow u(k-1)$;
for $j \in \mathbb{N}_{0:N_p}$ **do**
  if $\mathbf{p}_{n,j}^{(i)} < \mathbf{p}^{lb}$ **then** $\mathbf{p}_{n,j}^{(i)} \leftarrow \mathbf{p}^{\mathrm{lb}}$;
  if $\mathbf{p}_{n,j}^{(i)} > \mathbf{p}^{ub}$ **then** $\mathbf{p}_{n,j}^{(i)} \leftarrow \mathbf{p}^{\mathrm{ub}}$;
  if $\mathbf{p}_{n,j}^{(i)} - \mathbf{p}_{n,j-1}^{(i)} > \Delta_p$ **then** $\mathbf{p}_{n,j}^{(i)} \leftarrow \mathbf{p}_{n,j-1}^{(i)} + \Delta_p$;
  if $\mathbf{p}_{n,j}^{(i)} - \mathbf{p}_{n,j-1}^{(i)} < -\Delta_p$ **then** $\mathbf{p}_{n,j}^{(i)} \leftarrow \mathbf{p}_{n,j-1}^{(i)} - \Delta_p$;
  $\mathbf{v}_{n,j}^{(i-1)} \leftarrow \mathbf{p}_{n,j}^{(i)} - \mathbf{p}_{n,j}^{(i-1)}$
end

return $\mathbf{p}_n^{(i)}$, $\mathbf{v}_{n,j}^{(i-1)}$

5. CASE STUDIES

This section presents the simulation results of the developed control strategy. Firstly, we study the performance of the developed MPC on the full-state feedback nominal system. Secondly, we study the disturbed system with a 30% decrease in O_E. For all simulations, we assume that the system is at a steady state at the beginning and contains only nitric acid and tributyl phosphate with nominal parameters, i.e., $[U]^{\mathrm{aq}}_F = 0$. Then, from $t = 0$, $[U]^{\mathrm{aq}}_F$ is set to the nominal value. The set point is chosen as $[U]^{\mathrm{aqD}}_{9,\mathrm{set}} = 0.5[U]^{\mathrm{aqD*}}_9$ as shown in Fig. 2. Simulation parameters are given in Tab. 1 and each simulation.

5.1 Nominal case

Simulation results for nominal the PSO-based MPC are summarized in Tab. 2. As shown in Fig. 5 and Fig. 6, all MPC formulations can effectively stabilize the system at the desired operating condition. It can be seen that there is a period (from 4 h to 6 h) that the Alg. 2 is active to avoid constraint violations. In addition, it is shown that with a longer prediction horizon, MPC can better handle the constraints. For instance, in case (IIIa), MPC relaxes the soft constraint (12i) sooner compared to case (Ia) and (IIa) to avoid overshooting. As a result, this case has the lowest overshoot and settling time. Comparing Fig. 5 and Fig. 6 shows that a good trade-off in the cost function (11) also helps to avoid constraints violation and improve control performance. With larger values of $\mathbf{R}, \mathbf{S}$, (12) return a solution that is closer to u_{set}. However, if $\mathbf{R}, \mathbf{S}$ are too large, (12) will always return $u^*(k) = u_{\mathrm{set}}$, which is in fact the open loop controller.

Table 1. Overall simulation parameters.

Parameter	Value	Parameter	Value
T_{meas}	0.1 h	N_{part}	50
T_{esti}	0.5 h	w_i	0.9
T_{ctrl}	0.5 h	w_f	0.4
$[U]^{\mathrm{aqD}}_{9,\mathrm{set}}$	$0.5[U]^{\mathrm{aqD*}}_9$	$c_{1i} = c_{2f}$	2.5
$\mathrm{OS}^{\max}$	20 %	$c_{1f} = c_{2i}$	0.5
$\varepsilon_{\mathrm{ss}}$	5 %	Δ_p (MPC)	$\Delta^{\max}_{A_F}$
N_w	5	Δ_p (MHE)	∞
N_e	2	$r^{\min}_{\mathrm{clus}}$	70 %
ε_{-N_e}	$0.1\%[U]^{\mathrm{aqD}}_{9,\mathrm{set}}$	N_{ter}	5
$\varepsilon_{\mathrm{active}}$	$1\%[U]^{\mathrm{aqD}}_{9,\mathrm{set}}$	ϵ_{dl}	1E-5
N_{p2}	10	ϵ_{v}	0.001
$\mathbf{Q} = \mathbf{P}$	$\mathbf{I}$	ϵ_{l}	1E-4
		$i_{\max}$	100

Table 2. Results for PSO-based NMPC.

Case	(Ia)	(IIa)	(IIIa)	(Ib)	(IIb)	(IIIb)
$\mathbf{S} = \mathbf{R}$		$0.001/u_{\mathrm{set}}\mathbf{I}$			$0.01/u_{\mathrm{set}}\mathbf{I}$	
N_p	2	3	5	2	3	5
Max OS	19.7%	13.5%	4.6%	14.1 %	7.3 %	2.1 %
Settling time	7.4 h	5.7 h	5 h	5.5 h	4.8 h	3.6 h

5.2 Disturbed case

In this subsection, we study the effects of disturbances on the system when a $\pm 30\%$ variation of O_E (denoted
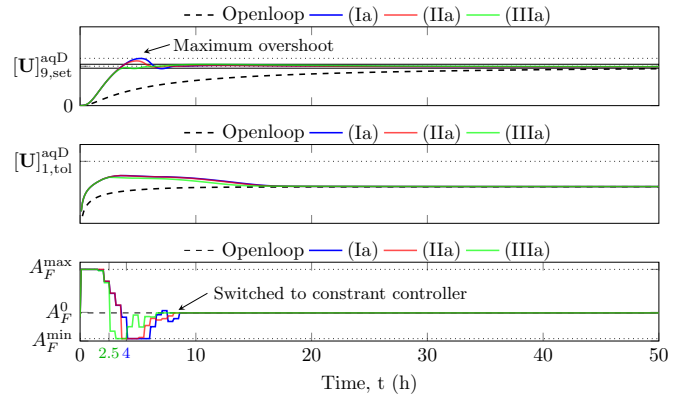

Fig. 5. Simulation results for nominal PSO-based NMPC with $\mathbf{S} = \mathbf{R} = 0.001/u_{\mathrm{set}}\mathbf{I}$.
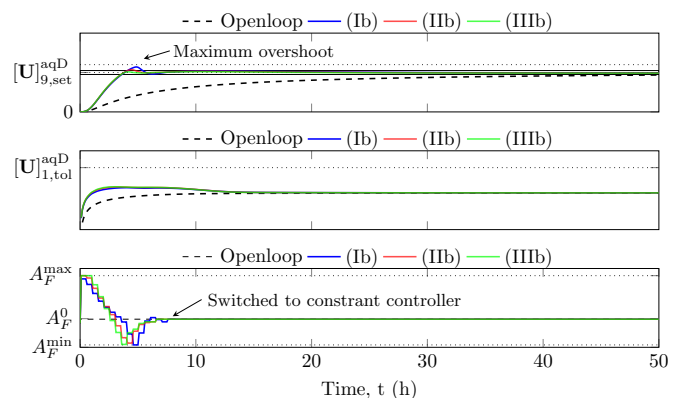

Fig. 6. Simulation results for nominal PSO-based NMPC with $\mathbf{S} = \mathbf{R} = 0.01/u_{\mathrm{set}}\mathbf{I}$.

as p in (1)) appears. We use the MPC with $N_p = 3$, $\mathbf{S} = \mathbf{R} = 0.01/u_{\mathrm{set}}\mathbf{I}$ integrated with and MHE with parameters given in Tab. 1. The simulation results are shown in Fig. 7 and Fig. 8, with O^0_E as the nominal value of O_E and the disturbances are assumed to appear at $t = 10$ h. It can be seen that without MHE, MPC will continue to make predictions with its old parameter O^0_E. Consequently, the control input is kept constant, as in case (IIb). Thus, it fails to keep the output around the set point. In addition, even with MHE, it takes time for the effects of disturbances to appear in the measurements, i.e., e_{esti} becomes sufficiently large. As a result, there is a short period just after $t = 10$ h in which MPC does not predict the process evolution well since it keeps using the old value of O_E. This period results in an oscillation after $t = 10$ h. However, thanks to the estimator, MHE, O_E can be estimated and fed back to MPC. Finally, the controller can stabilize the system within the tolerance band (3).

6. CONCLUSION

This paper presents a nonlinear optimal adaptive control strategy for the uranium extraction-scrubbing operation in the PUREX process. The strategy was developed based on optimization-based control and estimation methods, i.e., nonlinear Model Predictive Control (NMPC) and nonlinear Moving Horizon Estimation (MHE). The methods were used for set-point tracking application to ensure the solvent saturation level of the process and parameter estima-
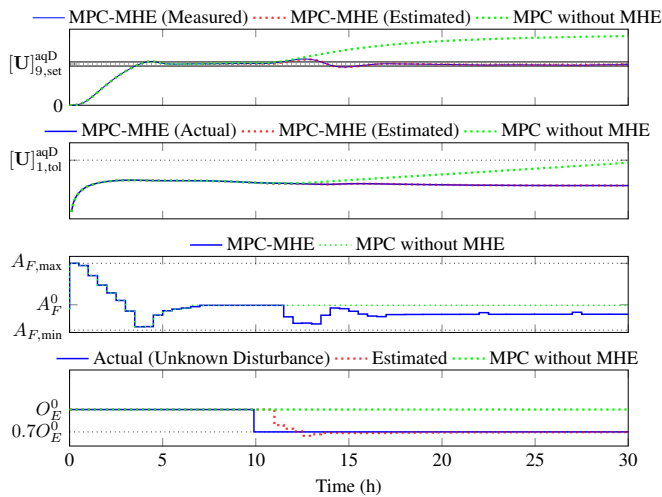
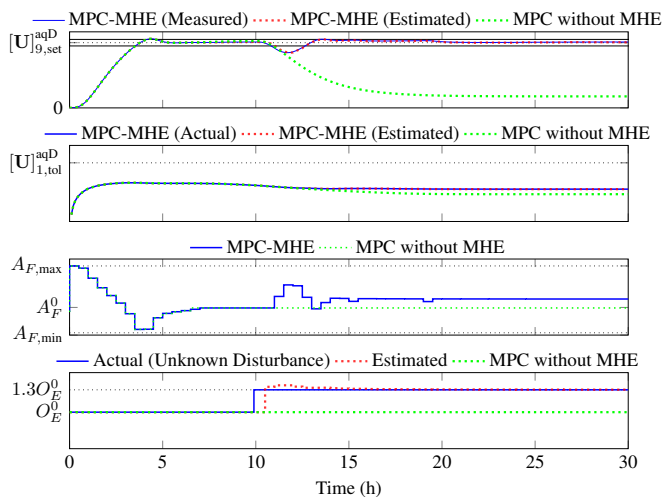Fig. 7. Simulation results for PSO-based adaptive NMPC under an unknown 30% decrease of O_E.

Fig. 8. Simulation result for PSO-based adaptive NMPC under an unknown 30% increase of O_E.

tion since the solvent flow rate is an unknown disturbance. The NMPC and MHE optimization problems are solved using the Particle Swarm Optimization (PSO) algorithm, which allows the use of PAREX directly without requiring explicit mathematical equations. Multiple simulations were done using the PAREX code, the qualified simulation program that allows steady and transient calculations of the process. Simulation results show that the developed control strategy stabilizes the system at the desired operation condition, disregarding disturbances. Future developments, including studies on stability, robustness, uncertainties handling, and experimental implementation, will be studied in June 2024.

REFERENCES

Bisson, J., Dinh, B., Huron, P., and Huel, C. (2016). PAREX, A Numerical Code in the Service of La Hague Plant Operations. *Procedia Chemistry*, 21, 117–124.

Engelbrecht, A.P. (2007). *Computational Intelligence: An Introduction.* Wiley. doi:10.1002/9780470512517.

Gbadega, P.A. and Sun, Y. (2022). A hybrid constrained particle swarm optimization-model predictive control (CPSO-MPC) algorithm for storage energy management optimization problem in micro-grid. *Energy Reports*, 8, 692–708.

Jabarivelisdeh, B., Carius, L., Findeisen, R., and Waldherr, S. (2020). Adaptive predictive control of bioprocesses with constraint-based modeling and estimation. *Computers & Chemical Engineering*, 135, 106744.

James, R. and David, M. (2022). *Model Predictive Control: Theory, Computation, and Design.* Nob Hill Publishing.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*. IEEE.

Liu, J., Su, Q., Moreno, M., Laird, C., Nagy, Z., and Reklaitis, G. (2018). Robust state estimation of feeding-blending systems in continuous pharmaceutical manufacturing. *Chemical Engineering Research and Design*, 134, 140–153.

Naka, S., Genji, T., Yura, T., and Fukuyama, Y. (2001). Practical distribution state estimation using hybrid particle swarm optimization. In *2001 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No.01CH37194)*. IEEE.

Qin, S.J. and Badgwell, T.A. (2003). A Survey of Industrial Model Predictive Control Technology. *Control Engineering Practice*, 11, 733–764.

Ratnaweera, A., Halgamuge, S., and Watson, H. (2004). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, 8(3), 240–255.

Rawlings, J.B. (2013). Moving horizon estimation. In *Encyclopedia of Systems and Control*, 1–7. Springer London.

Schwaab, M., Evaristo Chalbaud Biscaia, J., Monteiro, J.L., and Pinto, J.C. (2008). Nonlinear parameter estimation through particle swarm optimization. *Chemical Engineering Science*, 63(6), 1542–1552.

Tuveri, A., Nakama, C.S., Matias, J., Holck, H.E., Jäschke, J., Imsland, L., and Bar, N. (2023). A regularized moving horizon estimator for combined state and parameter estimation in a bioprocess experimental application. *Computers and Chemical Engineering*, 172, 108183.

Valipour, M. and Ricardez-Sandoval, L.A. (2022). A Robust Moving Horizon Estimation under Unknown Distributions of Process or Measurement Noises. *Computers & Chemical Engineering*, 157.

Van Den Bergh, F. and Engelbrecht, A.P. (2002). *An Analysis of Particle Swarm Optimizers.* Ph.D. thesis, ZAF. AAI0804353.

Vaudano, A. (2008). Overview of Treatment Processes. In *Monograph - Treatment and recycling of spent nuclear fuel.* Le Moniteur.

Vo, D.T., Prodan, I., Lefèvre, L., Vanel, V., Costenoble, S., and Dinh, B. (2023). Nonlinear model predictive control for uranium extraction-scrubbing operation in spent nuclear fuel treatment process. In *Proceedings of the 20th International Conference on Informatics in Control, Automation and Robotics.* doi: 10.5220/0012180700003543.

Zietkiewicz, J., Kozierski, P., and Giernacki, W. (2020). Particle swarm optimisation in nonlinear model predictive control, comprehensive simulation study for two selected problems. *International Journal of Control*, 94(10), 2623–2639.